OCTOPIZE
MIMETHIK DATA

Avatars

Documentation

Privacy-Preserving Synthetic Data
Complete Reference Guide

© Octopize — All rights reserved

About This Document

Complete reference for **Avatars**, Octopize's privacy-preserving synthetic data platform.

CHAPTERS

Overview · Principles · Legal · Methods · Metrics · User Guide · Self-Training · Deploying · FAQ · Support

AUDIENCE

Data scientists · Privacy officers · DevOps engineers · Legal & compliance teams

docs.octopize.io · Patent WO2020144118

Contents

Overview 7
- Avatars, a Privacy-Preserving Revolution 7
- Versatile Applications 7
- Key Benefits 7
- How It Works 7
- Privacy & Security 8
- Data Utility Without Compromise 8
- General work flow 8

Principles 11
- Key Uses 11
- Legal Considerations 11
- Methodology 11
- Evaluation Metrics 11
- Legal 13
- Methods 42
- Metrics 80

User Guide 145
- Choose your path 👇 145
- Quickstart 146
- Anonymization Guide 151
- Database Runs (Beta) 164
- Set Your Preferences 182
- License & Limits 185
- Crash Reporting 187

Self-Training Guide 190
- 📚 Recommended Learning Path 190

Deploying 193
- SaaS 194

Frequently Asked Questions (FAQ) 279
- ❓ Can we revert to the original individual from an anonymized dataset? 279
- ❓ Can the method handle small datasets (a few dozen records)? 279
- ❓ Does the method support free text anonymization? 279
- ❓ Can images be anonymized using this method? 279
- ❓ Does anonymization require user consent? 280
- ❓ How do we assess whether a means is reasonable? 280
- ❓ Can I anonymize my entire database at once for multiple use cases? 281
- ❓ Does the method support wide matrices? 281
- ❓ Does the method reduce dataset variance? 281
- ❓ Can certain results be preserved exactly in the anonymized dataset? 281
- ❓ What level of bias does the method introduce? 281
- ❓ Why was a patient-centric model chosen for anonymization? 282
- ❓ Can the method retain the characteristics of small clusters of individuals? 282

❓ When is my data considered anonymous? 282
❓ Can this method be used for data augmentation? 283
❓ How does the method handle missing data? 283
❓ What projection models are used in the anonymization process? 283
❓ How do you handle computed variables and business rules? 283
❓ Who is responsible in case of improper data anonymisation? 284
Support 286

© Octopize — All rights reserved

© Octopize — All rights reserved

01

Overview

Overview

Understand the value and inner workings of **Avatars in less than 5 minutes**.

Ready to dive in? Skip to the quickstart guide and start using Avatars right away.

Avatars, a Privacy-Preserving Revolution

Handling personal data is challenging due to strict regulations that limit secondary usage. The **Avatar Method** transforms sensitive data into synthetic, anonymized datasets that keeps the exact same structure, enabling safe and compliant data usage without compromising utility.

Versatile Applications

The Avatar Method is ideal for a wide range of use cases, including:

- **Data Anonymization** - Unlock secondary uses of personal data while ensuring compliance with privacy regulations.
- **Data Augmentation** - Enrich datasets for machine learning and analytics without exposing sensitive information.
- **Privacy Assessment** - Evaluate and mitigate re-identification risks in datasets.
- **Desensitization** - Share strategic data insights without revealing its source identity.
- **Pseudonymization** *(Beta)* - Automatically replace direct identifiers (names, emails, phone numbers…) with realistic or hashed substitutes while preserving referential integrity across tables. Learn more →

Key Benefits

✅ **Regulatory Compliance** – Aligns with privacy regulations to facilitate responsible data usage (GDPR and AI Act).

✅ **Unlocks Data Potential** – Since avatarized data is no longer classified as personal, new use cases and business opportunities emerge, accelerating access to valuable data while respecting privacy.

✅ **Retains Analytical Value** – Maintains the same structure, granularity, and statistical properties as the original data.

✅ **Quantified Privacy** – Privacy risk is measurable using dedicated metrics, allowing adaptation to specific data usage contexts.

How It Works

Avatars are generated through a rigorous yet explainable anonymization process:

1. **Input:** Pseudonymized data (without direct identifiers). Alternatively, the platform can handle pseudonymization for you as an integrated step.
2. **Transformation:** The Avatar Method applies advanced anonymization techniques while preserving statistical relevance.
3. **Output:** Synthetic, anonymous avatars with the same analytical properties as the original dataset. If pseudonymization was configured, direct identifiers are replaced with synthetic substitutes in the output.

© Octopize — All rights reserved

ORIGINAL DATASET

ID Patient	Age	Gender	Height	Inclusion Date
0001	43	Male	183	22/08/2021
0002	25	Female	176	17/08/2021
0003	38	Female	165	20/08/2021
0004	52	Male	174	19/08/2021
0005	79	Female	162	20/08/2021

→

AVATAR DATASET

ID Patient	Age	Gender	Height	Inclusion Date
#104	42	Male	179	21/08/2021
#205	23	Female	174	16/08/2021
#167	35	Male	168	19/08/2021
#492	56	Female	176	20/08/2021
#049	81	Female	165	22/08/2021

Privacy & Security

Ensuring privacy means mitigating re-identification risks. The Avatar Method now provides **two complementary reports** to assess and contextualize this risk:

1. **Technical Assessment**: A quantitative report measuring the residual re-identification risk based on EDPB criteria (Singling out, Linkability, Inference) and metrics mimicking **state-of-the-art attack scenarios**. 📄 See an example Technical Assessment report
2. **Impact Assessment (PIA)**: A contextual evaluation judging the likelihood and severity of these attacks based on data sensitivity and the usage environment.
   📄 See an example Impact Assessment report

These reports are designed to be provided and archived by the **company's Data Protection Officer (DPO)**, offering a complete vision to decide if the residual risk is reasonable.

Learn more about our privacy guarantees →

Data Utility Without Compromise

Because avatars mirror the structure and statistical integrity of original data, they can be used with the same **analytics, AI models, and business intelligence tools**. Adjustments ensure the best trade-off between privacy and usability.

- **Preservation of general indicators:** Key statistical measures such as means, standard deviations, and distributions are maintained.
- **Preservation of simple relationships:** Linear relationships, correlations, and statistical tests remain consistent.
- **Preservation of complex relationships:** Avatars retain structures useful for machine learning, clustering, and predictive modeling.
- **Minimal impact on outliers:** Extreme values are adjusted to preserve statistical coherence.

Explore data utility metrics →

General work flow

1. **Prepare your data** – Configure pseudonymization for direct identifiers (or remove the ones you don't need) and ensure a structured format (Single table, Database or Timeseries). The more you know about what data you need and what information you need to preserve, the better.

© Octopize — All rights reserved

2. **Process with the Avatar Method** – Our technology anonymizes data while maintaining its analytical value.
3. **Review the anonymization report** – Gain insights into transformation quality and compliance allowing for iterative adjustments to achieve the most suitable results for your specific data usage context.

🚀 Unlock the potential of your data—**compliantly, securely, and efficiently.**

© Octopize — All rights reserved

02

Principles

Principles

The Avatar method is a privacy-preserving approach designed to generate synthetic datasets that retain the statistical properties of original data while ensuring compliance with privacy regulations. This page provides an overview of the core technical and legal principles underlying the method and serves as a gateway to more detailed sections.

Key Uses

The Avatar method operates at the intersection of **data privacy, statistical integrity, and regulatory compliance**. Its primary goal is to enable the safe use of sensitive data by transforming it into a synthetic equivalent that is no longer considered personally identifiable while maintaining its analytical value. Recently the method has also been used for its data augmentation ability.

Legal Considerations

The regulatory landscape plays a crucial role in defining what constitutes **personal data** and determining the legal status of anonymization. This section explains:

- How different regulations (e.g., GDPR, CCPA) define **personal data**.
- The legal constraints associated with using such data.
- How anonymization affects **data usage permissions** under various legal frameworks.

Understand the legal framework →

Methodology

Understanding the **technical process** behind data anonymization is essential to ensure trust and transparency. The methodology section provides:

- A detailed explanation of each anonymization technique employed regarding your data structure.
- A transparent view of the steps taken to protect privacy while preserving data utility.
- A detailed list of data type handled for each use case.

Learn more about the methodology →

Evaluation Metrics

Assessing the quality of synthetic data requires rigorous evaluation based on well-defined metrics. This section covers:

- The **criteria** used to measure the quality and privacy of generated data.
- The **technical and mathematical principles** behind these metrics.
- How these metrics can be improved to fit your use case.

Explore the evaluation metrics →

To fully grasp the implications and benefits of the Avatar method, we encourage you to explore the dedicated sections on **methodology, evaluation metrics, and legal considerations**. By doing so, you will gain a comprehensive understanding

© Octopize — All rights reserved

of how synthetic data can be leveraged while adhering to privacy regulations and maintaining analytical relevance.

© Octopize — All rights reserved

Legal

The legal management of **personal data** is a crucial aspect of data processing, ensuring compliance with various regulations while balancing privacy protection and data utility. Personal data regulations exist to establish rights for individuals and responsibilities for organizations handling such data.

Several **key principles** govern the legal processing of personal data:

- **Lawfulness, Fairness, and Transparency**: Data must be processed lawfully, with clear and legitimate purposes, and individuals must be informed.
- **Purpose Limitation**: Data collection and processing must serve specific, well-defined purposes.
- **Data Minimization**: Only necessary data should be collected and retained.
- **Accuracy**: Data must be kept accurate and up to date.
- **Storage Limitation**: Personal data should not be kept longer than necessary.
- **Integrity and Confidentiality**: Organizations must ensure security and protection against unauthorized access.
- **Accountability**: Organizations must be able to demonstrate compliance with applicable laws.

Data protection regulations vary across jurisdictions, with key frameworks including:

- **European Artificial Intelligence Act (AI Act)** – European Union
- **General Data Protection Regulation (GDPR)** – European Union
- **UK General Data Protection Regulation (UK GDPR)** – United Kingdom
- **European Health Data Space (EHDS)** – European Union (health data)
- **California Consumer Privacy Act (CCPA)** – California, USA
- **Quebec Privacy Laws (Law 25 and Law 5)** – Quebec, Canada
- **Canadian Privacy Law (PIPEDA)** – Canada
- **Health Insurance Portability and Accountability Act (HIPAA)** – United States (health data)
- **Japan Privacy Law (APPI)** – Japan
- **New Swiss Federal Act on Data Protection (nFADP / nLPD)** - Switzerland

Ensuring compliance with these regulations is essential for organizations that process personal data, particularly when handling **sensitive data**, performing **data anonymization**, or transferring data across borders.

© Octopize — All rights reserved

European Artificial Intelligence Act (AI Act)

Jump straight to Avatar compliance regarding the AI Act

The **Artificial Intelligence Act (AI Act)** is the world's first comprehensive legislation on AI. Entering into force in August 2024, it aims to ensure that AI systems used in the EU are safe, transparent, traceable, non-discriminatory, and environmentally friendly, while encouraging innovation.

Risk-Based Approach

The AI Act classifies AI systems according to the level of risk they pose to individuals' rights and safety:

- **Unacceptable Risk**: Prohibited systems (e.g., social scoring, cognitive manipulation).
- **High Risk**: Systems used in critical infrastructure, education, employment, essential services, or law enforcement.
- **Limited Risk**: Transparency obligations (e.g., chatbots).
- **Minimal Risk**: The vast majority of systems (e.g., video games, spam filters).

Data Governance for High-Risk Systems

This is the core regulation for model developers. According to **Article 10 of the AI Act**, high-risk AI systems trained on data must comply with strict data governance and management requirements.

> *"Training, validation and testing data sets shall be subject to appropriate data governance and management practices."*

These practices must ensure that data sets are:

- **Relevant, representative, free of errors and complete** (Art. 10, paragraph 3).
- **Examined to detect possible biases** (Art. 10, paragraph 2, f).

The Challenge of Using Personal Data (GDPR vs AI Act)

Training models on real data poses a double legal challenge:

1. **GDPR Compliance**: Processing personal data for training requires a solid legal basis and compliance with the minimization principle.
2. **AI Act Compliance**: Article 10 requires high-quality and representative data sets, which often necessitates large volumes of varied data.

Article 4a sets out a notable exception regarding sensitive data (special categories of personal data). It was introduced by Regulation (EU) 2026/1744 (Digital Omnibus on AI), in force since 27 July 2026, which deleted the former Article 10(5):

> *"To the extent strictly necessary to ensure bias detection and correction in relation to high-risk AI systems […], providers of such systems may exceptionally process special categories of personal data, subject to appropriate safeguards for the fundamental rights and freedoms of natural persons."*

© Octopize — All rights reserved

This exception is subject to cumulative conditions. The first one is that *"the bias detection and correction cannot be effectively fulfilled by processing other data, including synthetic or anonymised data"* (Art. 4a(1)(a)). The records of processing activities must also explain *"why that objective could not be achieved by processing other data"* (Art. 4a(1)(f)). Article 4a(2) extends these conditions to providers and deployers of other AI systems and models, and to deployers of high-risk AI systems.

The text therefore explicitly requires synthetic or anonymised data to be considered first. Using raw personal data remains a high risk factor for non-compliance.

This also constitutes a major security risk identified by ANSSI (French National Cybersecurity Agency). In its security recommendations for generative AI systems, the agency highlights that trained models can memorize and regurgitate sensitive data in the event of an attack. To reduce these risks, ANSSI explicitly mentions the use of synthetic data as a relevant protective measure, recommending that models be trained only with data whose sensitivity is consistent with the access rights of end-users (Recommendation R18).

Training high-risk models therefore raises a business challenge: how to freely exploit the full potential of data without exposing the company to legal or security risks.

Regulatory Sandboxes

Article 53 and following establish regulatory sandboxes for AI. These are controlled environments that facilitate the development and testing of innovative AI systems under the supervision of competent authorities before they are placed on the market.

The goal is to enable innovation while ensuring compliance with the GDPR and the AI Act. Access to quality data in these environments is crucial. The provision of anonymous data facilitates the implementation of these sandboxes.

Legal Status of the Avatar Method under the AI Act

> **The Avatar method reconciles the quality requirements of the AI Act with privacy imperatives (GDPR). The method is therefore perfectly suited for training high-risk systems.**

Octopize's synthetic data generation method positions itself as a strategic compliance lever for AI providers. It allows meeting the requirements of Article 10 while activating the value of personal data. The anonymous nature of the generated data unlocks the full business potential of the original data by removing barriers related to personal data processing.

1. Compliance through Anonymity (Exiting GDPR)

Since the avatar method allows to generate **anonymous data** (non-personal), the generated data sets are no longer subject to the GDPR. This allows Data Scientists to use this data for training, validation, and testing without requiring the

© Octopize — All rights reserved

consent of the original individuals, thus lifting legal barriers and speeding up innovation.

2. Quality and Representativeness (Art. 10 §3)
The AI Act requires data to be "relevant, representative, and complete." Unlike classic anonymization techniques (such as masking or strong generalization) which destroy data utility, the **Avatar method preserves original statistical properties**. Models trained on our synthetic data perform equivalently to those trained on real data, guaranteeing the technical compliance required by Europe.

3. Bias Correction (Art. 10 §2)
Synthetic data generation allows for rebalancing data sets (e.g., increasing the representation of a minority class) to mitigate algorithmic biases. This directly addresses the obligation to detect and correct biases imposed by the AI Act, while protecting the privacy of the individuals concerned.

In summary, using the Avatar method for the AI Act allows you to:

- **Eliminate legal risks** related to the use of personal data (GDPR).
- **Guarantee model performance** thanks to statistical preservation.
- **Demonstrate responsible data governance**, a key criterion for CE certification of high-risk AI systems.
- **Accelerate Time to Market** by reducing the R&D cycle by 50% (access to data often takes 6 to 18 months).
- **Increase AI Performance**. The Avatar method allows for the generation of augmented and balanced datasets, improving model robustness and accuracy.
- **Reduce Operational Costs**. Using anonymous synthetic data lowers cybersecurity costs and secure storage expenses.

© Octopize — All rights reserved

California Consumer Privacy Act (CCPA)

Jump straight to Avatar compliance regarding CCPA

The **California Consumer Privacy Act (CCPA)** is a data privacy law that grants California residents enhanced rights over their personal information. It applies to businesses operating in California **or** serving California residents.

Who Must Comply?

A business must comply with CCPA if it meets **at least one** of the following criteria:

- Annual gross revenue **exceeds $25 million**.
- It processes the personal data of **50,000+ consumers, households, or devices** per year.
- It derives **50% or more** of its annual revenue from selling consumers' personal data.

Consumer Rights under CCPA

The CCPA provides California residents with several key **privacy rights**:

- **Right to Know**: Consumers can request details about what personal data a business collects and how it is used.
- **Right to Delete**: Consumers can request deletion of their personal data, with some exceptions.
- **Right to Opt-Out**: Consumers can opt out of their data being sold to third parties.
- **Right to Non-Discrimination**: Businesses cannot discriminate against users exercising their privacy rights.

Definition of Personal Information under CCPA

Under CCPA, **personal information** includes:

- Identifiers (name, address, email, IP address)
- Commercial data (purchase history, transactions)
- Biometric information
- Internet activity (browsing history, search history)
- Geolocation data
- Employment or education data

Pseudonymization, Anonymization, and CCPA Compliance

The **CCPA does not apply to anonymized or de-identified data**, as long as businesses ensure there is no reasonable way to re-identify individuals.

To be considered **de-identified**, data must not be linkable to a specific consumer and must be handled with technical safeguards preventing re-identification.

Legal Status of the Avatar Method

The Avatar method offers a **privacy by design** approach that specifically aim to prevent re-identification. It also offers a privacy evaluation toolkit to assess the residual re-identification risk. The use of the method and the synthetic data generated with it are therefore compatible with the CCPA regulation.

© Octopize — All rights reserved

General Data Protection Regulation (GDPR)

Jump straight to Avatar compliance regarding GDPR

The **General Data Protection Regulation (GDPR)** is a comprehensive data privacy law enacted by the European Union to protect individuals' personal data. It applies to all organizations that process personal data of **EU residents**, regardless of where the organization is based.

What is Personal Data?

According to the GDPR, **personal data** refers to:

> *"Any information relating to an identified or identifiable natural person ('data subject'); an identifiable natural person is one who can be identified, directly or indirectly, in particular by reference to an identifier such as a name, an identification number, location data, an online identifier, or to one or more factors specific to the physical, physiological, genetic, mental, economic, cultural or social identity of that natural person."*

Examples: Name, email, phone number, IP address, financial information, health records.

Legal Basis for Processing Personal Data

Under Article 6 of the GDPR, processing personal data is only lawful if it meets one of the following conditions:

- **Consent**: The individual has given explicit consent.
- **Contractual necessity**: Processing is necessary for contract performance.
- **Legal obligation**: Compliance with legal requirements.
- **Vital interests**: Processing is required to protect life or health.
- **Public interest**: Processing is necessary for tasks carried out in the public interest.
- **Legitimate interests**: The organization has a legitimate reason, provided it does not override the individual's rights.

Pseudonymization vs. Anonymization

- **Pseudonymization**: Data is modified so that it cannot be attributed to a person **without additional information**. However, pseudonymized data is still considered **personal data** under GDPR.
- **Anonymization**: Data is altered so that it is **irreversibly unidentifiable**. Anonymized data **is no longer subject to GDPR**.

See our blog article for more details →

Processing of Sensitive Data

Article 9 of the GDPR defines **sensitive data** as information relating to race, religion, political opinions, genetics, biometrics, health, or sexual orientation. Processing such data is generally **prohibited**, except under specific conditions:

- **Explicit consent**
- **Medical or scientific research**
- **Public health reasons**

© Octopize — All rights reserved

- **Legal claims**

Anonymization Process Status
The **European Data Protection Board (EDPB)**, formerly known as **Working Party 29 (WP29)**, considers that:

> "The anonymization process, which involves processing personal data to render it anonymous, constitutes **further processing**."

This interpretation highlights that anonymization is not an initial data processing activity but rather an additional transformation applied to personal data.

> ⚠️ **Draft guidelines**
>
> The EDPB Guidelines 02/2026 on anonymisation are a **draft** under public consultation until 30 October 2026. They are not final and may change before their final adoption. The points below that refer to them describe the draft.

The EDPB's draft Guidelines 02/2026 on anonymisation reinforce a more precise reading: the central question is not whether a transformation has been applied, but whether the resulting dataset remains **identifiable in the relevant context**, taking into account:

- the **nature and sensitivity of the data**,
- the **likely attackers and their means**,
- the **context of use** and data-sharing environment,
- and the **residual risk of re-identification** after technical and organizational safeguards.

This is a **contextual and risk-based assessment**, not a simple yes/no test.

A crucial point of the draft guidelines is that **a criterion does not need to be "100% protected" in an absolute sense for the dataset to be considered sufficiently anonymous in context**. The question is not whether a metric is perfect, but whether the **residual re-identification risk remains low enough** for the actual use case, taking into account the attacker model, available external knowledge, and the safeguards applied.

Similarly, **a weak result on one criterion does not automatically mean that the data is not anonymous**. A single metric may indicate a higher risk in a specific attack scenario, but the conclusion must be made in context: whether the scenario is realistic, whether attacker capabilities and background knowledge are credible, and whether additional safeguards reduce the residual risk to an acceptable level.

The draft guidelines also clarify that the **inference criterion** is not a vague or abstract concept. It refers to the realistic ability to infer information about an individual or a small group using available external information, probabilistic reasoning, and modern analytical methods. It is therefore assessed through realistic attack models rather than hypothetical impossibilities.

© Octopize — All rights reserved

In the draft EDPB guidelines, the assessment of anonymisation is typically framed through two complementary approaches:

- **Simplified approach**: a conservative, worst-case evaluation used when the data is exposed to a broad, external, or poorly controlled audience. It is the appropriate reference point for strict-sharing scenarios such as open data or broad dissemination, where the controller reasonably assumes a motivated attacker with access to external information.
- **Contextual approach**: a case-specific evaluation tailored to the actual recipient, purpose, data sensitivity, available background knowledge, safeguards, and environment of use. This is the more precise approach when the dataset is shared under controlled access conditions, with a known recipient and a defined purpose.

The two approaches are not contradictory: the simplified one provides a robust fallback for high-risk distribution, while the contextual one allows a more nuanced assessment when the real conditions of use are known and controlled.

Under **Article 5 of the GDPR** (link), personal data cannot be processed for purposes incompatible with those initially intended.

In other words, any **new processing purpose** must be **compatible** with the original purpose communicated to the data subjects at the time of collection. If deemed compatible, further processing is permissible under GDPR regulations.

Reasonable Means and the Attack Model

Recital 26 of the GDPR defines anonymous data as data where the person is **not or no longer identifiable**, taking into account **all the means reasonably likely to be used**, considering **cost, time, and the technological means available at the time of processing**.

This is a key point: the standard is not “all imaginable means in the abstract”, nor is it a requirement for absolute theoretical impossibility. It is a **reasonable attack model** test.

The EDPB’s draft guidance on anonymisation is directly relevant here: see the section on the means “reasonably likely to be used” in the EDPB Guidelines 02/2026 on anonymisation (draft, public consultation until 30 October 2026).

In practice, the question is therefore:

- who is the likely attacker?
- what external information is reasonably accessible?
- what linkage possibilities exist?
- what is the cost and time required to exploit them?
- what tools and methods are available to a motivated actor?

This means that a dataset can be considered anonymous in a controlled, purpose-limited environment even if a different, broader, more open distribution model would require a stricter assessment. The assessment is therefore **contextual** and depends on the actual use case and safeguards.

© Octopize — All rights reserved

This is also why the concept of **reasonable means** should be treated as a full dimension of the legal assessment, not as a side remark. It is central to the evaluation of singling out, linkability, and inference.

For a practical checklist on how this is assessed in real-world use cases, see How do we assess whether a means is reasonable?.

Legal Recognition of Anonymization

The concept of **"personal data"** as defined in Article 4(1) of the GDPR is not unlimited: it requires that the data subject be **identified or identifiable**. The **Court of Justice of the European Union (CJEU)** reaffirmed this principle (*SRB v. EDPS*, September 4, 2025), confirming that identifiability remains the central criterion for qualification.

Recital 26 of the GDPR specifies that information is considered anonymous where "**the data subject is not or no longer identifiable**," taking into account **all the means reasonably likely to be used**, considering **cost, time required**, and the **technological means available at the time of processing**.

This is the key update in the draft guidelines: data are not considered anonymous simply because an anonymization step was applied. They are considered anonymous only when the **residual risk of re-identification remains sufficiently low in the actual context of use**.

The legal test therefore focuses on **reasonable means in context**, not on whether the data is impossible to attack under any conceivable circumstance. This is why the assessment must be grounded in a realistic attacker model, a documented risk analysis, and the actual conditions of sharing.

Therefore, the status of data as anonymous depends on: - the **nature of the data**, - their **level of sensitivity**, - the **context of use**, - and the **realistic attack scenarios** that could be used against them.

An anonymization process must therefore be **assessed and documented** to demonstrate that the residual risk of re-identification remains reasonable in light of the context and purposes.

The technical assessment of risk should take into account the three criteria defined by the **EDPB**: - **Singling out**: the possibility of isolating a single individual within a dataset, - **Linkability**: the possibility of linking records or datasets concerning the same individual, - **Inference**: the possibility of deducing new, sensitive information about an individual.

In addition, a **PIA (Privacy Impact Assessment)** must be carried out to evaluate the **likelihood of potential attacks** against the anonymization process, as well as their **impact** if successful. In simple terms, the technical report evaluates the **residual re-identification risk** posed by certain threats, while the PIA measures the **probability and severity** of those threats in the given context.

Where residual risk is considered too high, **additional technical and organizational safeguards** can be implemented to further strengthen the

© Octopize — All rights reserved

anonymization process. These may include controlled access, encryption, time-limited access, data minimization, or filtering of high-risk variables.

When all these criteria are satisfied for the intended use case, the resulting data can be considered **outside the scope of the GDPR**.

Legal Status of the Avatar Method

> **The synthetic data generated using the Avatar method is no longer considered personal data and therefore fall outside the scope of the GDPR. Here's why:**

The **EDPB** (formerly WP29) acknowledges that anonymization, as further processing of personal data, **can be considered compatible** with the original processing purpose. However, this is contingent on the anonymization process meeting the criteria outlined in **Opinion 05/2014** (link).

The anonymization method developed by **Octopize** has been awarded a **CNIL compliance certificate**, confirming its classification as an **authentic anonymization technique**.

Furthermore, anonymization **enhances data security** by making identification **irreversible**. Therefore, it is categorized as a **technical and organizational security measure**, not a substantial modification of data processing.

- **Anonymization of personal health data** can be considered a **compatible processing activity** aligned with the **original purposes** of data collection.
- **Anonymization does not constitute a substantial modification** of data processing under GDPR.
- **No additional notification or consent** from data subjects is required when implementing anonymization.
- **Avatarized data** resulting from this anonymization process is no longer classified as **personal data** and thus falls **outside the scope of the GDPR**.

The data generated using the Avatar Method are not considered anonymous by default. Their anonymity is systematically and automatically **evaluated and documented** through a rigorous process involving: - A **technical report** measuring the residual re-identification risk based on EDPB criteria (Singling out, Linkability, Inference) and metrics simulating **state-of-the-art attack scenarios**. - A **Privacy Impact Assessment (PIA)** evaluating the **likelihood and severity** of these attacks based on data sensitivity and usage context.

© Octopize — All rights reserved

UK General Data Protection Regulation (UK GDPR)

Jump straight to Avatar compliance regarding UK GDPR

The **UK General Data Protection Regulation** (UK GDPR), tailored by the Data Protection Act 2018 (DPA 2018), is the comprehensive data privacy law governing the processing of personal data in the United Kingdom. It applies to all organizations that process personal data of **UK residents**, regardless of where the organization is based.

The UK GDPR has been substantially amended by the Data (Use and Access) Act 2025 (DUAA), which received Royal Assent on 19 June 2025. Most of its data protection changes came into force on **5 February 2026** (SI 2026/82). This page reflects the text in force.

Since **30 September 2026**, the Information Commissioner's Office (ICO) has been replaced by the **Information Commission** (SI 2026/1015). The guidance cited below was published under the ICO name and is still hosted on ico.org.uk.

The European Commission renewed its adequacy decision for the UK on 19 December 2025, until 27 December 2031 (ICO). Personal data can therefore still flow from the EU to the UK without additional transfer tools.

What is Personal Data?

According to Article 4(1) of the UK GDPR **personal data** refers to: > *"'personal data' means any information relating to an identified or identifiable natural person ('data subject'); an identifiable natural person is one who can be identified, directly or indirectly, in particular by reference to an identifier such as a name, an identification number, location data, an online identifier or to one or more factors specific to the physical, physiological, genetic, mental, economic, cultural or social identity of that natural person;"*

Examples: Name, email, phone number, IP address, financial information, health records.

Legal Basis for Processing Personal Data

Under Article 6 of the UK GDPR, processing personal data is only lawful if it meets one of the following conditions:

- **Consent**: The individual has given explicit consent.
- **Contractual necessity**: Processing is necessary for contract performance.
- **Legal obligation**: Compliance with UK legal requirements.
- **Vital interests**: Processing is required to protect life or health.
- **Public interest**: Processing is necessary for tasks carried out in the public interest or the exercise of official authority.
- **Recognised legitimate interests**: Processing is necessary for one of the purposes listed in Annex 1 (e.g. crime prevention, safeguarding, emergencies). Added by the DUAA; not available to public authorities performing their tasks.
- **Legitimate interests**: The organization has a legitimate reason, provided it does not override the individual's fundamental rights.

Pseudonymization vs. Anonymization

- Pseudonymization:

© Octopize — All rights reserved

> *"means the processing of personal data in such a manner that the personal data can no longer be attributed to a specific data subject without the use of additional information, provided that such additional information is kept separately and is subject to technical and organisational measures to ensure that the personal data are not attributed to an identified or identifiable natural person;"*

Pseudonymized data **is still personal data**.

- Anonymization: Data is altered so that people are **not, or no longer, identifiable**. Anonymous information **is no longer subject to UK GDPR**.

Processing of Special Category Data

Article 9 of the UK GDPR defines **Special category data** as information relating to race, ethnic origin, political opinions, religious or philosophical beliefs, trade union membership, genetics, biometrics, health, or sexual orientation. Processing such data is prohibited except under specific conditions outlined in the UK GDPR and Schedule 1 of the DPA 2018, such as:

- **Explicit consent**
- **Scientific or historical research and statistics**, carried out in accordance with Article 84B
- **Public health reasons**
- **Legal claims**

Anonymization Process Status

The ICO states that anonymization itself **counts as processing** of personal data (ICO, Introduction to anonymisation):

> *"The end result (the anonymous information) is not subject to data protection law, but the procedure (anonymisation) is. […] This means that you must comply with data protection requirements for your anonymisation process. This includes ensuring you have a lawful basis for it and you clearly define your purpose(s) and provide people with information about it. In general it is likely that applying anonymisation techniques to the personal data you hold will be fair and lawful."*

Under Article 5(1)(b) of the UK GDPR, personal data cannot be further processed in a manner incompatible with the purposes for which it was collected. Since the DUAA, compatibility is assessed under Article 8A, which lists the factors to take into account (link between purposes, context, nature of the data, consequences, safeguards such as encryption or pseudonymisation).

For research, archiving and statistical (RAS) purposes, the law is explicit:

- Article 84B(1)(b) allows processing for RAS purposes when it is *"carried out in order to convert the personal data into information which can be processed in*

© Octopize — All rights reserved

a manner which does not permit the identification of a data subject", i.e. anonymization.
- Article 8A(3)(b) treats processing carried out in accordance with Article 84B for RAS purposes as **compatible** with the original purpose.
- Article 84C requires appropriate safeguards, including technical and organisational measures to respect data minimisation, such as pseudonymisation.

Legal Recognition of Anonymization

The concept of **personal data** requires that the data subject be identified or identifiable. Recital 26 specifies that information is considered anonymous where the data subject is **not or no longer identifiable,** taking into account all the means reasonably likely to be used, considering cost, time required, and available technology.

The ICO published its final anonymisation and pseudonymisation guidance on 28 March 2025. The ICO notes that it is under review following the DUAA. It applies a risk-based approach to anonymization, notably the **Motivated Intruder Test** (How do we ensure anonymisation is effective?). The status of data as anonymous depends on whether it is reasonably likely that a competent, motivated person could re-identify an individual, considering:

- the nature and sensitivity of the data,
- the context of use and who has access to the data,
- the availability of other public or external datasets.

An anonymization process must be assessed and documented to demonstrate that the residual risk of re-identification is sufficiently remote. The technical assessment evaluates risks such as:

- **Singling out**: isolating a single individual within a dataset,
- **Linkability**: linking two datasets concerning the same individual,
- **Inference**: deducing new, sensitive information about an individual.

In addition, a **Data Protection Impact Assessment** (DPIA) is required where processing is likely to result in a high risk. It can be used to evaluate the likelihood of potential attacks against the anonymization process, as well as their impact. If the residual risk is too high, additional technical and organizational safeguards must be implemented. When these standards are met, the data falls outside the scope of the UK GDPR.

Draft Guidance on Anonymisation, Pseudonymisation and Research

> ⚠️ **Draft**
>
> This guidance is a **consultation draft**, not final guidance. The consultation runs from 24 August to 19 October 2026.

The draft guidance explains how to apply anonymisation and pseudonymisation when relying on the RAS provisions. Its main points for synthetic data are:

© Octopize — All rights reserved

- Before processing personal data for research, organizations must consider whether **anonymous information** can be used instead.
- Identifiability is assessed **in context**: the data, who can access it, what other information they hold, the **release model** (public release, defined groups, internal use) and the means reasonably likely to be used.
- Synthetic data *"can reduce risks, but it is not automatically anonymous, especially if it is generated from personal data"*. For synthetic data generated from personal data, the motivated intruder test should cover **membership inference** and **attribute disclosure** attacks.
- Organizations should state what the synthetic data is suitable for, label it clearly, and document how identifiability was assessed and how safeguards will be reviewed.

Legal Status of the Avatar Method

Synthetic data generated using the Avatar method can fall outside the scope of the UK GDPR when its residual re-identification risk is assessed as sufficiently remote in its context of use. Here's why:

- Converting personal data into anonymous information is expressly recognized for RAS purposes (Article 84B(1)(b)) and treated as compatible further processing (Article 8A(3)(b)). For other purposes, compatibility is assessed under Article 8A, and the ICO considers anonymization generally likely to be fair and lawful.
- The anonymization method developed by **Octopize** holds a CNIL compliance certificate (EU). The UK has no equivalent certification, but its evaluation methodology follows the risk-based framework described by the ICO.
- Anonymization can reduce identification risk and function as a technical safeguard, but whether data is anonymous depends on whether re-identification risk is sufficiently remote in its context of use.
- Like any processing, anonymization requires a lawful basis and must be covered by the information given to data subjects. It does not require a separate consent.
- Avatarized data that passes this assessment is not classified as personal data.

The data generated using the Avatar Method are not considered anonymous by default. Their anonymity is **systematically** evaluated and documented through a rigorous process involving:

- A **technical report** measuring the residual re-identification risk (Singling out, Linkability and Inference) using metrics that simulate state-of-the-art attack scenarios. These include the Membership Inference and Attribute Inference attacks named in the ICO draft research guidance. The report provides the technical evidence for the **Motivated Intruder Test**.
- A **Data Protection Impact Assessment** (DPIA) evaluating the likelihood and severity of these attacks based on data sensitivity, data recipients (the release model) and usage context.

© Octopize — All rights reserved

European Health Data Space (EHDS)

Jump straight to Avatar compliance regarding the EHDS

The **European Health Data Space (EHDS)** is established by Regulation (EU) 2025/327 of 11 February 2025, published in the Official Journal on 5 March 2025. It covers two uses of electronic health data:

- **Primary use**: access by natural persons to their own health data and exchange of these data between healthcare providers across the EU.
- **Secondary use**: reuse of health data for research, innovation, policymaking, statistics, education and the improvement of care.

This page focuses on **secondary use** (Chapter IV of the regulation), where anonymised and synthetic data play a central role.

Timeline

The regulation is in force but applies in stages (Article 105):

- **26 March 2027**: general date of application. By then, Member States must inform the Commission of the health data access bodies they have designated (Article 55(6)).
- **26 March 2029**: the rules on secondary use (Chapter IV) apply to most categories of health data.
- **26 March 2031**: they apply to the remaining categories listed in Article 51(1), points (b), (f), (g), (m) and (p): determinants of health, genetic, genomic and other omic data, clinical trial data and data from research cohorts.

Implementing acts are being adopted in the meantime. For example, Implementing Regulation (EU) 2026/2098 sets the minimum metadata that health data holders must provide to describe their datasets for secondary use, based on the HealthDCAT-AP profile. It applies from 26 March 2029.

Purposes of Secondary Use

Health data access bodies may only grant access to health data for the purposes listed in **Article 53(1)**, including:

- public interest in public or occupational health,
- policymaking and regulatory activities,
- official statistics,
- education and teaching in the health or care sectors,
- scientific research related to the health or care sectors, including **development and innovation activities** for products or services and the **training, testing and evaluation of algorithms**, including AI systems,
- improvement of the delivery of care.

Access is granted by a health data access body through a **data permit** (Article 68) or, for anonymised statistical answers, a **health data request** (Article 69).

Anonymised Data First

The EHDS makes anonymised data the default format for secondary use. According to **Article 66(2)**:

© Octopize — All rights reserved

> *"Health data access bodies shall provide electronic health data in an anonymised format, where the purpose of processing by the health data user can be achieved with such data, taking into account the information provided by the health data user."*

Pseudonymised data are the exception. They are only provided where the health data user *"has sufficiently demonstrated that the purpose of processing cannot be achieved with anonymised data"* (Article 66(3)). The access application must include this justification (Article 67(2), point (e)).

The regulation also asks for state-of-the-art techniques. When providing access to a pseudonymised or anonymised dataset, a health data access body *"should use state-of-the-art pseudonymisation or anonymisation technology and standards, ensuring to the maximum extent possible that natural persons cannot be re-identified by health data users"* (Recital 72). Health data users are prohibited from re-identifying or attempting to re-identify the persons concerned (Article 61(3)).

Recital 92 adds that some categories of health data remain sensitive even once anonymised, for instance for rare diseases, because a residual re-identification risk can remain *"even where state-of-the-art anonymisation techniques are used"*.

Secure Processing Environments

Health data are only made available through a **secure processing environment** (Article 73(1)). Only non-personal data can leave it. According to **Article 73(2)**:

> *"Health data access bodies shall review the electronic health data included in a download request to ensure that health data users are only able to download non-personal electronic health data, including electronic health data in an anonymised statistical format, from the secure processing environment."*

Right to Opt Out

Natural persons have the right to opt out, at any time and without giving reasons, from the processing of their personal electronic health data for secondary use (**Article 71(1)**). Member States may provide for limited exceptions for public interest purposes (Article 71(4)).

Each Member State sets up its own health data access bodies and the national rules needed to apply the regulation.

Legal Status of the Avatar Method

The EHDS does not define anonymisation itself: whether a dataset is anonymous is assessed under the GDPR, taking into account all the means reasonably likely to be used (see the GDPR page). The regulation does not mention synthetic data either.

Within that framework, the Avatar method addresses several needs created by the EHDS:

© Octopize — All rights reserved

- **Anonymised data first (Article 66)**: health data holders and health data access bodies need anonymised datasets that are useful enough for the requested purpose. The Avatar method generates synthetic data that preserve the statistical properties of the original data, so that more requests can be served without pseudonymised data.
- **Outputs of secure processing environments (Article 73)**: only non-personal data can be downloaded. Avatar data whose anonymity has been demonstrated are non-personal data and can therefore leave the environment, unlike pseudonymised extracts.
- **State-of-the-art anonymisation and residual risk (Recitals 72 and 92)**: each avatarization comes with a privacy evaluation that measures the residual re-identification risk, including for rare records, to document that the generated data are anonymous in their context of use.

Generating avatars processes the original personal data. This step therefore remains subject to the GDPR and to the EHDS rules that apply to personal data, including the right to opt out: data of persons who have opted out should be excluded upstream, before generation.

© Octopize — All rights reserved

Quebec Privacy Laws (Law 25 and Law 5)

Jump straight to Avatar compliance regarding Quebec Laws

Data privacy in the Canadian province of Quebec is primarily governed by two key legislative frameworks: - **Law 25** (formerly Bill 64): A broad and strict reform of Quebec's privacy legislation applicable to all private and public organizations handling personal data of Quebec residents. - **Law 5**: A specialized law governing the collection, use, and disclosure of health and social services information, applying specifically to healthcare institutions, professionals, and the technology providers processing health data on their behalf.

What is Personal Information?

Unlike the GDPR's "personal data", Quebec law uses the term **Personal Information (PI)**. According to Section 2 of the Private Sector Act, personal information refers to: > *"Any information which relates to a natural person and directly or indirectly allows that person to be identified."*

Examples: Name, address, phone number, email, IP address, geolocation, employment or financial data, demographic, or behavioral data.

Legal Basis and Organizational Obligations

Quebec's framework differs from the GDPR's six legal bases. Under Law 25, the primary legal basis for processing personal information is **Consent**, which must be clear, free, informed, and given for specific purposes.

Organizations can process PI without consent only under strict exceptions, such as: - **Contractual necessity**: When necessary to conclude or execute a contract. - **Legal obligation**: Compliance with provincial or federal laws. - **Study, research, or statistics**: Under strict conditions, including ensuring data is de-identified. - **Consistent purposes**: If the new use has a direct and relevant connection to the original purpose.

To ensure accountability, Law 25 also requires organizations to appoint a **Privacy Officer** (automatically the CEO unless explicitly delegated), maintain a privacy governance policy, and ensure strict access logging.

De-identification vs. Anonymization

Quebec law introduces a strict legal distinction between these two concepts:

- **De-identification** (equivalent to Pseudonymization): According to the Act, information is de-identified if it *"no longer allows the person concerned to be directly identified."* De-identified data **is still considered personal information** and remains subject to the law.
- **Anonymization**: According to Section 17 of the Act, data is anonymized if it is, at all times, reasonably foreseeable in the circumstances that it **irreversibly** no longer allows the person to be identified directly or indirectly. Anonymized information **is no longer subject to the Act**.

Processing of Sensitive Information and Health Data (Law 5)

Under Law 25, personal information is considered **sensitive** if, due to its nature or the context of its use, it entails a high level of reasonable expectation of privacy.

© Octopize — All rights reserved

Law 5 creates a highly regulated regime specifically for health information. Health information refers to any identifying data related to: - A person's physical or mental health - Medical consultations or diagnoses - Test results and treatments - Social services received

Using such sensitive data for research purposes is permitted but heavily regulated. It generally requires a Privacy Impact Assessment (PIA), strict security measures, and, in the case of health data, approval from a Research Ethics Board.

Sanctions and Enforcement
Compliance is critical, as the **Commission d'accès à l'information (CAI)** enforces both laws with severe penalties for mishandling data: - **Law 25**: Administrative penalties up to **$10 million CAD or 2% of global turnover**. Criminal fines can reach **$25 million CAD or 4% of global turnover** for serious violations. - **Law 5**: Specific fines up to **$150,000 CAD** per offense for the mishandling of health information.

Anonymization Process Status
Under Quebec law, once the purposes for which personal information was collected are achieved, organizations must destroy the information. However, both Law 25 and Law 5 explicitly recognize that **anonymization is an acceptable alternative to destruction**. Organizations may anonymize data to use it for serious and legitimate purposes (like research), provided the process eliminates any reasonable risk of re-identification.

Converting personal information into anonymized data is a legally recognized action for the lifecycle management of data, provided it adheres to the **Regulation respecting the anonymization of personal information**.

Legal Recognition of Anonymization
The Government of Quebec's Regulation respecting the anonymization of personal information sets strict criteria. The status of data as anonymous depends on a continuous assessment. The organization must: - Remove all direct identifiers. - Conduct an analysis of the re-identification risks, considering the nature and sensitivity of the data, the context of use, and the availability of other datasets.

To demonstrate that the residual risk is practically zero, the technical assessment must evaluate universally recognized risks: - **Singling out**: isolating a single individual within a dataset. - **Linkability**: linking two datasets concerning the same individual. - **Inference**: deducing new, sensitive information about an individual.

In addition, a **Privacy Impact Assessment** (PIA / *Évaluation des facteurs relatifs à la vie privée - EFVP*) is legally required prior to the anonymization process to demonstrate that the methodology effectively mitigates risks. When these standards are met, the resulting data falls completely outside the scope of the privacy laws.

© Octopize — All rights reserved

Legal Status of the Avatar Method

> **The synthetic data generated using the Avatar method is no longer considered personal or health information and thus falls completely outside the scope of Law 25 and Law 5.**

The Avatar method provides a **privacy-by-design framework** that perfectly aligns with Quebec's regulatory requirements. Here's why:

- Law 25 and Law 5 explicitly recognize anonymization as a valid, permanent transformation of personal information.
- The Avatar method generates **statistically realistic but fully anonymous synthetic datasets**. It systematically removes direct identifiers and evaluates the data against recognized best practices, including those expected by the CAI.
- Avatarized data does not require **consent**, **notification**, or subsequent **PIAs** for secondary use, as it no longer constitutes identifiable information. **No additional legal obligations** apply under Quebec privacy law when using these datasets.

The data generated using the Avatar Method are not considered anonymous by default. Their anonymity is **systematically** evaluated and documented through a rigorous process involving:

- A **technical report** (accessible via our privacy evaluation toolkit) measuring the residual re-identification risk against the highest international and Quebec regulatory standards (Singling out, Linkability, and Inference).
- A **Privacy Impact Assessment** (PIA / EFVP) evaluating the likelihood and severity of potential attacks based on data sensitivity and usage context.

📚 References

- Law 25 (formerly Bill 64) – CAI summary (EN)
- Law 5 – Health Information Protection (FR)
- Legis Québec – Act respecting health and social services information
- CNIL – Certification of anonymization (FR)

© Octopize — All rights reserved

Canadian Privacy Law (PIPEDA)

Jump straight to Avatar compliance regarding PIPEDA

Data privacy at the federal level in Canada is governed by the **Personal Information Protection and Electronic Documents Act (PIPEDA)**. It applies to private-sector organizations across Canada that collect, use, or disclose personal information in the course of a commercial activity (except in provinces with their own substantially similar privacy laws, such as Quebec, British Columbia, and Alberta).

What is Personal Information?

Under Part 1, Section 2(1) of PIPEDA, **Personal Information** is defined concisely as: > *"Information about an identifiable individual."*

The Office of the Privacy Commissioner of Canada (OPC) clarifies that information will be about an identifiable individual where there is a **serious possibility that an individual could be identified** through the use of that information, alone or in combination with other information.

Legal Basis for Processing Personal Information

Unlike the GDPR, PIPEDA does not feature a list of "legal bases" like legitimate interest. PIPEDA is built on 10 Fair Information Principles (Schedule 1 of the Act).

The cornerstone of lawful processing under PIPEDA is **Consent** (Principle 4.3): > *"The knowledge and consent of the individual are required for the collection, use, or disclosure of personal information, except where inappropriate."*

Exceptions to Consent: Section 7 of PIPEDA outlines strict exceptions where consent is not required, such as for legal investigations, medical emergencies, or specific statistical and scholarly research (under Section 7(2)(c) and 7(3)(f)).

Pseudonymization vs. Anonymization

Important Note: PIPEDA (in its current enacted form) **does not legally define** the terms "pseudonymization", "de-identification", or "anonymization".

Instead of statutory definitions, the distinction relies on the OPC's official interpretation of what constitutes "an identifiable individual": - **Pseudonymized/ De-identified data**: The OPC states in its guidance on de-identification that if data is merely stripped of direct identifiers (like names) but individuals can still be re-identified by linking it with other data, it **remains personal information** subject to PIPEDA. - **Anonymized data**: According to the OPC, true anonymization implies that the risk of re-identification is virtually zero. When information is altered to the point that it can no longer be linked to an identifiable individual, **PIPEDA no longer applies**.

Processing of Sensitive Data

PIPEDA does not create a separate legal category with prohibited uses for sensitive data (unlike GDPR's Article 9). However, Principle 4.3.4 explicitly states that: > *"Although some information (for example, medical records and income records) is almost always considered to be sensitive, any information can be sensitive, depending on the context."*

© Octopize — All rights reserved

The sensitivity of the data dictates the form of consent required. For sensitive data, the OPC expects **express (opt-in) consent**.

Anonymization Process Status and Legal Recognition
Because PIPEDA does not codify anonymization, compliance relies on demonstrating that the data falls outside the definition of Section 2(1) (i.e., it is no longer about an identifiable individual).

The OPC expects organizations to assess the **risk of re-identification**. In its guidance, the OPC refers to the "serious possibility" test established by the Federal Court of Canada (*Gordon v. Canada*, 2008). Data is only considered anonymous if there is **no serious possibility** of identifying an individual.

To prove this, the technical assessment must evaluate recognized vulnerabilities: - **Singling out** - **Linkability** - **Inference**

Legal Status of the Avatar Method
The synthetic data generated using the Avatar method is no longer considered personal information and therefore falls entirely outside the scope of PIPEDA. Here's why:

- Under PIPEDA, the threshold for data to be exempt from the law is the elimination of any "serious possibility" of re-identification (*Gordon v. Canada* standard).
- The anonymization method developed by **Octopize** systematically removes direct identifiers and replaces them with fully synthetic, statistically representative avatars.
- This process ensures that no individual can be singled out, linked, or inferred, reducing the risk of re-identification to practically zero.
- Because the resulting Avatarized data is no longer "information about an identifiable individual" (Section 2(1)), it does not require **knowledge and consent** (Principle 4.3) for secondary uses or disclosures.

The anonymity of the data is **systematically** evaluated and documented through a rigorous process involving:

- A **technical report** measuring the residual re-identification risk (Singling out, Linkability, and Inference) using metrics that simulate state-of-the-art attack scenarios to meet the OPC's strict expectations.
- A **Privacy Impact Assessment** (PIA) evaluating the likelihood and severity of these attacks based on data sensitivity and usage context.

Official References: - PIPEDA (Full Text - Justice Laws Website) - OPC Guidance on the privacy implications of de-identification

© Octopize — All rights reserved

Japan Privacy Law (APPI)

Jump straight to Avatar compliance regarding APPI

Data privacy in Japan is governed by the **Act on the Protection of Personal Information (APPI)**. It applies to all business operators handling the personal information of individuals in Japan, as enforced by the Personal Information Protection Commission (PPC).

> ⚠️ **2026 amendment**
>
> The APPI was amended in 2026 (Act No. 56 of 2026). The PPC is drafting the implementing cabinet order and rules (overview, in Japanese). Among other changes, the amendment removes the need for consent when personal data are used only to create statistical information, including AI development that qualifies as such, and prohibits the improper use and unlawful acquisition of pseudonymously and anonymously processed information that contains contact details such as an address, a phone number, an email address or a cookie ID (PPC, 14 October 2026, in Japanese). This page describes the APPI as currently applied. It will be updated once the amendment applies.

What is Personal Information?

According to Article 2(1) of the APPI, **Personal Information** is strictly defined as information relating to a living individual that: > *(i) can identify a specific individual by name, date of birth or other description contained in such information (including such information as will allow easy reference to other information and will thereby enable the identification of a specific individual); or* > *(ii) contains an individual identification code.*

Legal Basis for Processing Personal Information

The APPI does not use a system of "legal bases" (such as legitimate interest or contractual necessity). Instead, lawful processing is governed by **Purpose Specification and Restriction on Provision to a Third Party**:

- **Specification of Purpose (Article 17)**: A business operator must specify the utilization purpose as explicitly as possible and must not handle personal information beyond the scope necessary to achieve that purpose (Article 18) without obtaining the principal's consent.
- **Third-Party Provision (Article 27)**: With limited exceptions, a business operator **shall not provide personal data to a third party without obtaining the prior consent of the principal**.

Pseudonymization vs. Anonymization

The APPI explicitly defines two strict statutory categories for transformed data in Article 2:

- **Pseudonymously Processed Information (Article 2(5))**: Information relating to an individual that can be produced from processing personal information so that the individual **cannot be identified unless collated with other information**. Pseudonymized data is restricted to internal use; it **cannot be provided to third**

© Octopize — All rights reserved

parties (Article 41), but operators are exempt from certain obligations like responding to access requests.
- **Anonymously Processed Information (Article 2(6))**: Information relating to an individual that can be produced from processing personal information so that **a specific individual cannot be identified and the personal information cannot be restored**.

Processing of Special Care-Required Information (Sensitive Data)
Under **Article 2(3)** of the APPI, **Special care-required personal information** comprises data revealing a principal's race, creed, social status, medical history, criminal record, or fact of having suffered damage by a crime.

Under **Article 20(2)**, acquiring this sensitive data explicitly requires the **prior consent** of the principal, except in very specific circumstances (e.g., protecting human life). Furthermore, the opt-out exception for third-party provision (Article 27(2)) cannot be applied to special care-required information.

Anonymization Process Status and Legal Recognition
To legally qualify as **Anonymously Processed Information**, the data must be processed according to strict standards prescribed by the PPC rules. Under **Article 43**, a business operator must ensure the process is **irreversible**.

The PPC Guidelines specify that producing Anonymously Processed Information requires: 1. Deleting descriptions that can identify a specific individual. 2. Deleting all individual identification codes. 3. Deleting codes linking the information to other datasets. 4. Deleting idiosyncratic (peculiar) descriptions. 5. Taking appropriate measures to prevent the restoration of the original personal information.

Furthermore, under **Article 43(3)**, when an operator produces such data, they must publicly announce the categories of information it contains. Under **Article 43(5)**, the operator is strictly prohibited from collating the anonymized data with other information to identify the original individual.

If these strict statutory conditions are met, the **Anonymously Processed Information can be freely provided to third parties without the individual's consent**.

Legal Status of the Avatar Method
The synthetic data generated using the Avatar method fulfills and exceeds the requirements for **Anonymously Processed Information** under the APPI, rendering it safe for secondary use and third-party provision without individual consent. Here's why:

- **Article 2(6) and Article 43** of the APPI allow the sharing and secondary use of data if it is irreversibly transformed so that the original personal information cannot be restored.
- The anonymization method developed by **Octopize** systematically deletes all direct and idiosyncratic identifiers. It replaces original records with fully synthetic, statistically representative avatars, making the restoration of the original personal information mathematically and technically impossible.

© Octopize — All rights reserved

- Because the process is entirely irreversible and prevents re-identification, Avatarized datasets do not trigger the consent requirements for third-party provision under Article 27.

The anonymity of the data is **systematically** evaluated and documented through a rigorous process involving:

- A **technical report** measuring the residual re-identification risk against universal and APPI standards (Singling out, Linkability, and Inference) using metrics that simulate state-of-the-art attack scenarios, proving that restoration is impossible.
- A **Privacy Impact Assessment** evaluating the likelihood and severity of these attacks, ensuring compliance with the strict non-re-identification rules enforced by the PPC.

Official References: - Act on the Protection of Personal Information (APPI) - English Translation via Japanese Law Translation Database - Personal Information Protection Commission (PPC) Japan

© Octopize — All rights reserved

Health Insurance Portability and Accountability Act (HIPAA)

Jump straight to Avatar compliance regarding HIPAA

The **Health Insurance Portability and Accountability Act (HIPAA)** is a U.S. federal law that regulates the use, storage, and transmission of **protected health information (PHI)**.

What is Protected Health Information (PHI)?

PHI includes any **individually identifiable health information**:

- Medical history
- Diagnoses and treatment information
- Prescription records
- Laboratory results
- Insurance details

HIPAA Covered Entities and Business Associates

HIPAA applies to:

- **Healthcare providers** (hospitals, clinics, doctors)
- **Health insurance companies**
- **Third-party vendors** handling PHI (business associates)

De-Identification and HIPAA Compliance

HIPAA allows **two main methods** for **de-identifying** PHI so that it is no longer regulated:

1. **Safe Harbor Method**: Removing 18 specific identifiers.
   - Names
   - Geographic subdivisions smaller than a state (e.g., street, city, ZIP code), excluding the first 3 ZIP digits when populations are >20,000
   - All dates directly related to an individual (except year), and all ages over 89 (can be grouped as “90 or older”)
   - Telephone numbers
   - Fax numbers
   - Email addresses
   - Social Security numbers
   - Medical record numbers
   - Health plan beneficiary numbers
   - Account numbers
   - Certificate/license numbers
   - Vehicle identifiers and license plate numbers
   - Device identifiers and serial numbers
   - Web URLs
   - IP addresses
   - Biometric identifiers (e.g., fingerprints, voiceprints)
   - Full-face photos and similar images
   - Any other unique identifying number, characteristic, or code (except as allowed in the “Re-identification” section)

© Octopize — All rights reserved

2. **Expert Determination**: A qualified expert certifies that data has minimal risk of re-identification.

Once data is **de-identified**, it is no longer considered **PHI** and is exempt from HIPAA restrictions.

Legal Status of the Avatar Method

The Avatar method offers a **privacy by design** approach that specifically aim to prevent re-identification. Direct identifiers are no longer present in the data generated in accordance with the **Safe Harbor Method**. The method also offers a privacy evaluation toolkit to assess the residual re-identification risk. This evaluation aligns with the recommended **Expert Determination**. The use of the method and the synthetic data generated with it are therefore compatible with the HIPAA regulation.

© Octopize — All rights reserved

New Swiss Federal Act on Data Protection (nFADP / nLPD)

Jump directly to Avatar Method compliance under the nFADP

The **New Swiss Federal Act on Data Protection (nFADP)**, or **nLPD** in French, is Switzerland's modernized legal framework for the protection of personal data. It has been in effect since **September 1, 2023**, and applies to all organizations processing **personal data of individuals located in Switzerland**, even if the organization is based abroad.

What is personal data?

Under the nFADP, **personal data** is defined as:

> *"All information relating to an identified or identifiable natural person."*

Examples: name, email address, AVS number (social security), IP address, location data, health information, etc.

This definition is broadly similar to that of the GDPR, though more concise in wording.

Legal basis for data processing

Unlike the GDPR, the nFADP does **not require a specific legal basis** for each processing activity. However, it mandates compliance with **core principles**, including:

- **Lawfulness of processing**
- **Good faith and proportionality**
- **Defined purpose**
- **Data accuracy**
- **Data security**
- **Transparency toward data subjects**

Explicit consent is required in certain situations, particularly when processing sensitive personal data or for specific cross-border transfers.

Processing of sensitive personal data

The nFADP defines **sensitive data** as including:

- Health data
- Religious, philosophical, or political views
- Trade union membership
- Intimate or private matters
- Criminal or administrative proceedings and sanctions
- Social welfare measures

Such data may only be processed if:

- **Explicit consent** is obtained
- A **clear legal basis** permits it
- There is an **overriding public or private interest**, and appropriate safeguards are in place

© Octopize — All rights reserved

Status of the anonymization process

According to the Swiss Federal Data Protection and Information Commissioner (FDPIC), data that has been anonymized such that individuals are **no longer identifiable, directly or indirectly**, is **no longer considered personal data** and thus **falls outside the scope of the nFADP**.

In other words, anonymization is recognized as a **security measure**, **not a secondary data processing activity** requiring new legal grounds or renewed data subject notification—**if it complies with the principles of the law**.

The nFADP requires that any reuse of personal data respects the **original processing purpose**, unless:

- The new processing is compatible with the original purpose
- Adequate security measures are implemented
- There is an overriding legal or public interest

Anonymization, when used to **enhance data protection**, is generally seen as **compatible with the original purpose**.

Unlike the GDPR, the nFADP does not explicitly mention "compatible further processing" or "substantial modification," but similar principles apply by analogy, particularly:

- Adherence to the **principle of transparency**
- Consideration of **risks to the personality or fundamental rights of the data subject**
- **No new notification** is required if the change is a security-enhancing technical measure (such as anonymization)

Legal status of the Avatar Method

> **Synthetic data generated through the Avatar method is no longer considered personal data under the nFADP.**

The anonymization technique developed by **Octopize** follows a certified process that ensures **irreversible de-identification**. It fully complies with nFADP expectations:

- It **removes all identifiable links** to individuals.
- It qualifies as a **data security measure** recognized by the FDPIC.
- It **does not require renewed consent or data subject notification** when aligned with the original purpose.

In summary:

- Anonymization through the Avatar Method is **compliant with the nFADP**.
- The resulting data is **no longer personal**.
- **No additional consent or notice** is required when applying this anonymization.
- **Avatarized data** is therefore **outside the scope of the nFADP**.

© Octopize — All rights reserved

Methods

Providing high-quality synthetic data that balances privacy and utility is at the heart of the Octopize methodology. Our goal is to make the generation of such high-quality data as simple and accessible as possible for every user—no matter their technical background. Because data comes in many forms, a single anonymization method simply isn't enough. That's why we've developed a set of specialized techniques—each tailored to a specific data structure—to ensure adaptability across all situations.

We distinguish three main data structures, each requiring a dedicated framework to use our algorithms.

Note: All methods developed regarding a specific data structure are compatible with each other. We can for instance apply both Single Tabular and Time Series anonymization within a Multi Table anonymization context.

1. Single Tabular Data

For datasets where each row corresponds to a unique individual, the **single-table framework** is the most suitable. Simple tabular data was the first use case we addressed historically. Over time, we have developed multiple methodologies to handle this structure, offering flexibility depending on the context and privacy needs.

- The standard **Avatar method**, remains our go-to approach. this method efficiently preserves statistical properties while ensuring privacy at the individual level.

  🔗 Learn more about Single Tabular Data anonymization
- We also implemented **AvatarFastDP**, a variant of the Avatar method that generates synthetic data with **differential privacy** mechanisms.

  🔗 Learn more about Differential Privacy methods

2. Multi-Table (Relational) Data

When information is spread across multiple linked tables (e.g., patient records and medical visits), a **multi-table framework** is required. This method maintains consistency across relationships while anonymizing both individual tables and their interconnections.

🔗 Learn more about Multi-Table Data anonymization

When the tables live in a **database**, Avatar can read them there and write the anonymized copy into another database, keys and joins included.

🔗 Learn more about Database anonymization

3. Time Series Data

For datasets where values evolve over time (e.g., sensor readings, medical signals), a **time-series-adapted method** should be applied. This framework ensures that trends, patterns, and temporal dependencies are preserved while anonymizing sensitive information.

🔗 Learn more about Time Series Data anonymization

© Octopize — All rights reserved

The Avatar methodology is designed to generate **synthetic data** that maintain the structure, granularity, and statistical relevance of the original data while ensuring compliance with privacy regulations. We have built multiple frameworks to help you achieve both **data protection** and **analytical integrity** for any data structure.

© Octopize — All rights reserved

Single Tabular Data

This document describes the simple tabular data anonymization method developed by Octopize. This approach enables the anonymization of individual-level data, where each row corresponds to an independent individual. The method has been openly published[1] in scientific literature and is also protected by a patent[2]. Its uniqueness lies in its individual-centric approach, ensuring high utility preservation while providing proven privacy.

Scope of application

This anonymization method operates under the following conditions:

Input Data: Requires pseudonymized data, meaning at this point there are no longer direct identifiers in the dataset.

Supported Data Types:

- Continuous variables (integer or float)
- Textual categorical variables (no matter the number of categories)
- Numeric categorical variables
- Boolean values
- Dates
- Missing data (the original distribution pattern of missing values will be preserved)

Scalability: Can process datasets ranging from a few dozen to several billion records efficiently. Large volumes of data are handled using a batch-parallelized approach, ensuring fast computation times.

Output

Avatars have the following properties with regard to the original data:

- ☒ **Same structure:** Same number of variables with the same encoding and variable names.
- ☒ **Same granularity:** Same number of records (1-for-1 mapping).
- ☒ **Similar aggregates:** Expect similar averages, sums, medians, etc., across all variables.
- ☒ **Similar links:** Similar correlations between variables.
- ☒ **Similar clustering properties:** Multivariate information within the dataset is preserved.

[1] Guillaudeux, M., Rousseau, O., Petot, J. et al. Patient-centric synthetic data generation, no reason to risk re-identification in biomedical data analysis. npj Digit. Med. 6, 37 (2023). https://doi.org/10.1038/s41746-023-00771-5

[2] Procédé de création d'avatars pour protéger des données sensibles, Big Data Santé, FR

© Octopize — All rights reserved

ORIGINAL DATASET

ID Patient	Age	Gender	Height	Inclusion Date
0001	43	Male	183	22/08/2021
0002	25	Female	176	17/08/2021
0003	38	Female	165	20/08/2021
0004	52	Male	174	19/08/2021
0005	79	Female	162	20/08/2021

→

AVATAR DATASET

ID Patient	Age	Gender	Height	Inclusion Date
#104	42	Male	179	21/08/2021
#205	23	Female	174	16/08/2021
#167	35	Male	168	19/08/2021
#492	56	Female	176	20/08/2021
#049	81	Female	165	22/08/2021

Core Algorithm

Global approach

In short, the core of the Avatar method follows three major steps.

1. Input: the input data takes the form of a pseudonymized tabular dataset.
2. The core of the Avatar method:
   - a. Individual observations are projected into a complete multidimensional space.
   - b. Using the first ncp dimensions of this space, pairwise distances are computed between all sensitive individual observations to find the k nearest neighbors used to define a local area.
   - c. For each individual, a single avatar is pseudo-stochastically drawn in their local area. k, ncp, and other parameters drive the randomness and information content of the simulations.
   - d. The properties of the avatar dataset are evaluated by computing re-identification and signal retention metrics.
3. Output: the avatars are reverse-transformed from their coordinates in the projected space into values in the original space corresponding to the initial dataset. Synthetic observations (rows) are shuffled to remove the link between the original individuals and the avatars.

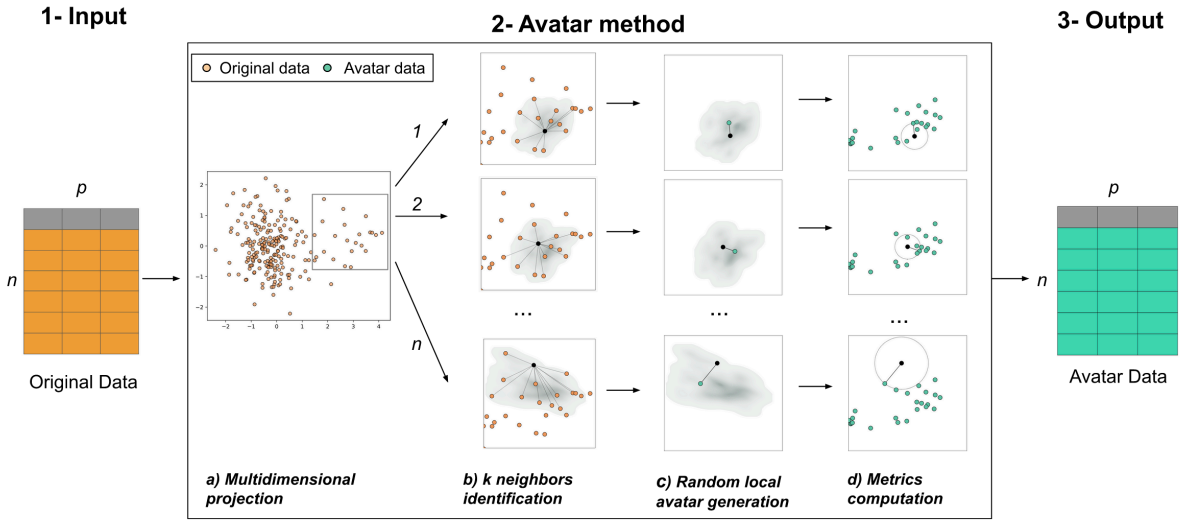


Figure 1: Method

© Octopize — All rights reserved

Figure 1: the Avatar method uses local modeling to stochastically generate a synthetic individual, termed an avatar.

Legend: (1) Original pseudonymized sensitive data. (2) The anonymization consists of four steps: (2a) individuals are projected in a multidimensional space; (2b) pairwise distances are computed to find the k nearest neighbors (here $k = 12$) in a reduced space; (2c) a synthetic individual is pseudo-randomly generated in the subspace defined by the neighbors; (2d) privacy and utility metrics are evaluated. (3) Output of the synthetic dataset.

Synthetic coordinate generation

At step 2c. an avatar is generated for an individual at the weighted barycenter of its neighbors.

The weight of each neighbor is defined as follows:

$$\forall k \in [1, ..., K], P_k = \frac{1}{D_k} \cdot R_k \cdot C_k$$

Where:

- D_k: distance between the individual and its neighbor k,
- $R_k \sim \xi(1)$: a random weight following an exponential distribution,
- $C_k = (\frac{1}{2})^{j_k}$: a contribution, where j_k is the value at index k of the randomly shuffled vector $[1, ..., K]$.

The weights are then normalized for their values to range between 0 and 1 as:

$$W_k = \frac{P_k}{\sum_{i=1}^{K} P_i}$$

The avatar is generated at the weighted center of the k nearest neighbors:

$$x' = \sum_{k=1}^{K} W_k . x_k$$

where,

- x' is the new coordinates of the individual's avatar.
- W_k are the weights of its neighbors.
- x_k the coordinates of its neighbors.

The Avatar coordinates are then inverse transformed in the original space using the inverse operation of the factor analysis technique used in step 2a.

Singularity of the approach

This anonymization method stands out due to its transparent and explainable design, allowing full control over the anonymization process and its evaluation. By generating data within a locally projected modeling space, the approach ensures the creation of plausible synthetic individuals preserving multivariate relationships.

Additionally, the introduction of controlled randomness guarantees the irreversible nature of the anonymization process. The individual-centric approach further

© Octopize — All rights reserved

enhances the assessment of residual risk, providing a more comprehensive evaluation of the privacy protection of the generated data.

Cite this page

If you use the avatar method in your research, please cite as follow:

Guillaudeux, M., Rousseau, O., Petot, J. et al. Patient-centric synthetic data generation, no reason to risk re-identification in biomedical data analysis. npj Digit. Med. 6, 37 (2023). https://doi.org/10.1038/s41746-023-00771-5

© Octopize — All rights reserved

Differential Privacy Methods

Alongside the standard Avatar method, Octopize offers **AvatarFastDP**, a synthesis method that uses differential privacy (DP) mechanisms. Both methods share the same preprocessing pipeline but differ in how synthetic data is generated in the latent space.

This page explains what each method does, when to use it, and how to configure AvatarFastDP.

Shared Pipeline

Regardless of the method chosen, every avatarization follows the same four steps:

1. **Load & type** — Data is loaded, columns are typed (int, float, category…), and missing values are optionally imputed.
2. **Project** — Data is projected into a latent space (via FAMD, Factor Analysis of Mixed Data, for tabular data containing both numerical and categorical data). The number of principal components is controlled by the `ncp` parameter.
3. **Synthesize** — Synthetic coordinates are generated in the latent space. **This is the step that differs between the two methods.**
4. **Back-project** — Synthetic coordinates are projected back to the original space, producing the final avatar dataset.

The following parameters are common to both methods: `ncp`, `imputation`, `use_categorical_reduction`, `exclude_variable_names`, `column_weights`. See the Run Anonymization page for details on each.

Method Comparison

Aspect	Avatar	AvatarFastDP
Main parameter	`k` (neighbors)	`fast_dp_epsilon` (ε privacy budget)
DP guarantee	None (heuristic privacy)	Uses DP mechanisms, but not end-to-end DP
Underlying method	KNN in latent space	Gaussian / GMM / Histogram in latent space
Speed	Fast	Fast
Best for	General use, fine control via `k`	Quick DP-flavored synthesis
Supports multi-table	Yes	Yes
Supports time series	Yes	No

> **i Mutually exclusive**
>
> The two methods are **mutually exclusive**. You set exactly one of `k` or `fast_dp_epsilon` when configuring parameters. Setting both raises an error.

© Octopize — All rights reserved

When to Use Which

- **Avatar (standard)** — Your go-to choice for most use cases. It offers fine-grained control over the privacy/utility trade-off via the `k` parameter and works with all data structures (tabular, multi-table, time series).
- **AvatarFastDP** — Choose this when you want **fast, DP-flavored synthesis** for experimentation or when runtime matters. It applies DP noise during synthesis but does not provide an end-to-end DP guarantee.

AvatarFastDP

AvatarFastDP is a **fast, lightweight DP synthesis** method. It fits a statistical distribution on the projected coordinates and samples from a noisy version of it.

It applies calibrated DP noise during synthesis, but the projection and back-projection steps are **not** covered by the DP guarantee. It is therefore **not end-to-end differentially private**.

Parameters

The table lists the keys under `avatarization_fast_dp` in a YAML configuration. In the Python SDK, `epsilon` is set with `fast_dp_epsilon`.

© Octopize — All rights reserved

Parameter	Type	Required	Default	Description
`epsilon`	float	**Yes**	—	Total privacy budget. Must be > 0.
`mechanism`	string	No	`"gaussian"`	One of `"gaussian"`, `"gmm"`, or `"histogram"`.
`gmm_n_components`	int	No	`5`	Number of Gaussian components. Only used when `mechanism="gmm"`.
`histogram_n_bins`	int	No	`10`	Number of bins per dimension. Only used when `mechanism="histogram"`.
`bounds_percentile`	float	No	`5.0`	Symmetric percentile for inferring data bounds (e.g., 5.0 uses the 5th–95th percentile range). Must be in (0, 50). Used by `"gaussian"` and `"histogram"`.

Python SDK

```
# Default gaussian mechanism
runner.set_parameters("my_table", fast_dp_epsilon=1.0)
```

📝

Mechanism-specific options (`mechanism`, `gmm_n_components`, etc.) are not yet exposed in `set_parameters()` — they use their default values. Advanced configuration is possible via YAML.

YAML Configuration

```
kind: avatarization-parameters
spec:
  schema: schema
```

© Octopize — All rights reserved

```yaml
avatarization_fast_dp:
  my_table:
    epsilon: 1.5
    mechanism: gmm
    gmm_n_components: 3
```

Mechanisms

AvatarFastDP offers three mechanism choices for how synthesis is performed in the latent space.

Gaussian (default)

Fits a **multivariate Gaussian** distribution (mean vector + covariance matrix) on the latent coordinates and adds calibrated Laplace noise to the data sensibility.

- The privacy budget is split between mean estimation (30%) and covariance estimation (70%).
- Data bounds are inferred from empirical percentiles (controlled by `bounds_percentile`) and used to calibrate noise sensitivity.
- The noisy covariance is symmetrized and regularized to remain positive definite.
- Synthetic points are drawn from a multivariate normal with the noisy parameters.

Best for: **unimodal** distributions in the latent space.

GMM (Gaussian Mixture Model)

Fits a **standard GMM** using Expectation-Maximization, then perturbs the fitted parameters (weights, means, covariances) with Laplace noise.

- Budget split: 20% weights, 40% means, 40% covariances.
- Noisy weights are clipped to non-negative values and renormalized. Covariances are symmetrized and regularized.
- Controlled by `gmm_n_components` (default: 5).

Best for: data with **multimodal** distributions in the latent space.

Histogram

Fits **independent per-dimension histograms** with DP noise on bin counts. This mechanism assumes independence between latent dimensions.

- Budget is split equally across dimensions.
- Each dimension gets `histogram_n_bins` fixed-width bins. Laplace noise is added to counts, which are then clipped and normalized.
- Synthetic values are drawn by selecting a bin from the noisy histogram and sampling uniformly within it.

Best for: **simple, exploratory** use cases. The independence assumption between dimensions is strong and may reduce utility on complex data.

© Octopize — All rights reserved

Relational Database

Relational data refers to multiple tables with hierarchical relationships between them. This structure is referenced as **Multi-table** data.

A simple tabular anonymization approach applied independently to each table would be insufficient. Indeed, both privacy and utility depend not only on the individual tables but also on the relationships between them. For easy and effective handling of this data structure, we have developed a dedicated framework. This framework ensures the protection and preservation of inter-table relationships while maintaining data utility and privacy within each table.



Example of structural database with 3 tables. Inter-table privacy risk is illustrated with patient 1 being the only one having 2 visits. Inter-table utility is illustrated with women having on average more visits than men.

Scope of application

The multi-table anonymization method shares the same scope as the single-table approach, meaning it supports the following data types:

Supported Data Types:

- Continuous variables (integer or float)
- Textual categorical variables (no matter the number of categories)
- Numeric categorical variables
- Boolean values
- Dates
- Missing data (the original distribution pattern of missing values will be preserved)

Specificity:

- all tables should contain one primary key
- all tables should be linked to another table using a primary key and a foreign key

© Octopize — All rights reserved

- primary key should be one column and not a combination of multiple columns
  Scalability:
  This method can efficiently process datasets ranging from a few dozen to several billion records. Large volumes of data are handled using a batch-parallelized approach, ensuring fast computation times.

However, modeling the relationships between tables requires **additional constraints** to be met to ensure both privacy protection and structural consistency.

Each table must contain at least one identifier variable, known as a **primary key**, which holds a unique value for each row. Relationships between tables are established through **foreign keys**, which are identifier variables that reference the primary key of another table. Unlike primary keys, foreign keys do not need to be unique within their table, allowing multiple rows to link to the same referenced entity.

VISIT DATA

id	patient_id	doctor_id	day
1	1	1	Monday
2	1	2	Wednesday
3	2	3	Thursday
4	3	2	Monday

primary key **foreign keys**

example of primary key and foreign key in a child table

Output
Such as Single Tabular anonymization, avatars from Multi-table anonymization have the following properties with regard to the original data:

- ☒ **Same structure:** Same number of variables with the same encoding and variable names.
- ☒ **Same granularity:** Same number of records (1-for-1 mapping).
- ☒ **Similar aggregates:** Expect similar averages, sums, medians, etc., across all variables.
- ☒ **Similar links:** Similar correlations between variables.
- ☒ **Similar clustering properties:** Multivariate information within the dataset is preserved.

But also:

© Octopize — All rights reserved

- ☒ **Similar inter-tables relationships:** Multivariate information between tables is preserved.

Core algorithm

Definitions

We call **parent table**, the table having its primary key referred to in the other table. We call **child table**, the table having a foreign key referring to the other table. Finally, we call **individual table** a table where each row refers to a unique individual. In our example above, *PATIENT_DATA* and *DOCTOR_DATA* are individual tables. A **link** is defined between a parent table with a primary key and a child table with a foreign key referring to this primary key.

Algorithm

The anonymization of a multi table database is performed on each link between tables starting from top to bottom.

The top table is defined as the one with no parent table. For each link, the process is achieved in three main stages. - **Enrichment** - **Avatarization** - **Matching**

For each link between tables (e.g. between *PATIENT_DATA* and *VISIT_DATA*):

1.Enrichment

Broadcast the information of the parent table in the child table. The join is based on parent table primary key and child table foreign key referring to the parent table.

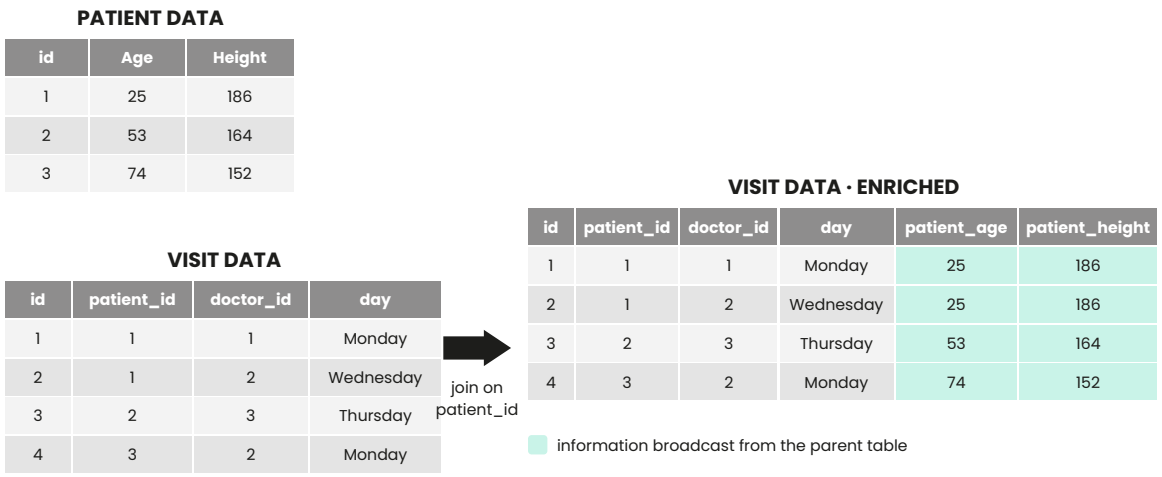


Illustration of enrichment step performing bottom information broadcast.

2.Avatarization

- Anonymize parent table using tabular anonymization described here.
- Anonymize enriched child table also using tabular anonymization.

© Octopize — All rights reserved

PATIENT DATA

id	Age	Height
1	25	186
2	53	164
3	74	152

tabular anonymization

AVATAR PATIENT DATA

id	Age	Height
1	27	186
2	57	164
3	71	152

VISIT DATA · ENRICHED

id	day	patient_age	patient_height
1	Monday	25	186
2	Wednesday	25	186
3	Thursday	53	164
4	Monday	74	152

tabular anonymization

AVATAR VISIT DATA · ENRICHED

id	day	patient_age	patient_height
1	Monday	29	183
2	Monday	72	154
3	Thursday	56	167
4	Wednesday	26	180

Illustration of tabular avatarization step for both parent and child enriched table.

3.Matching

Regarding the nature, the sensitivity and the volume of data we developed three methodology to perform a matching.

- minimum distance assignment

```
    - Common information of avatar parent table and avatar child enriched table are
transformed into coordinates using principal component method (Principal Component
Analysis or Factor Analysis of Mixed Data)
    - Distances are computed between coordinates from parent table and avatar child
enriched table.
    - Each record of the avatar child enriched table is associated with the closest
record of the avatar parent table to recreate the parent foreign key in the avatar
child enriched table.
    - Columns broadcasted from parent table to child table are dropped to create the
final avatar child table
```

© Octopize — All rights reserved

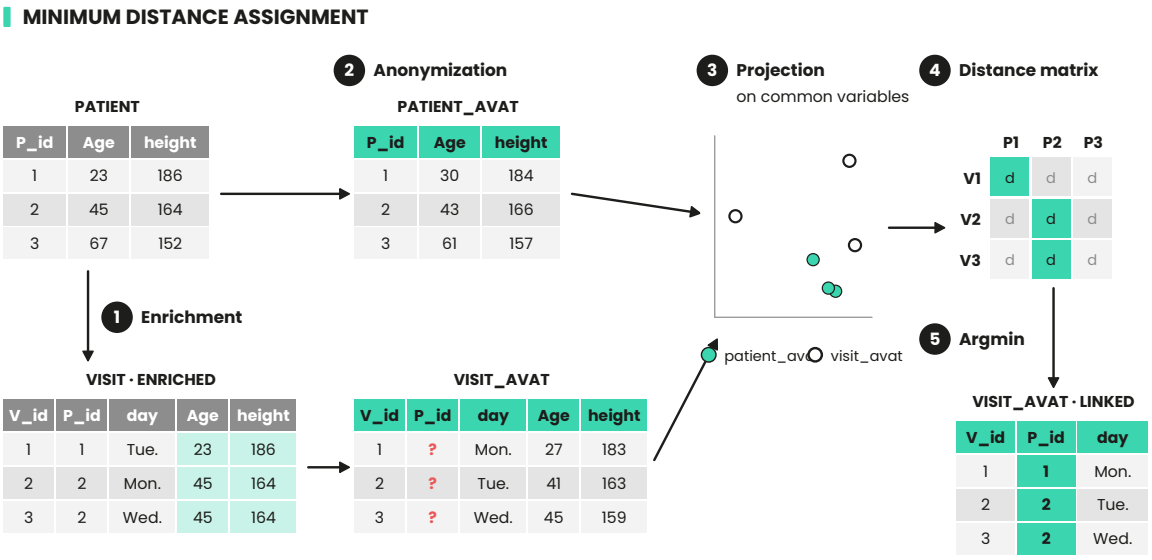
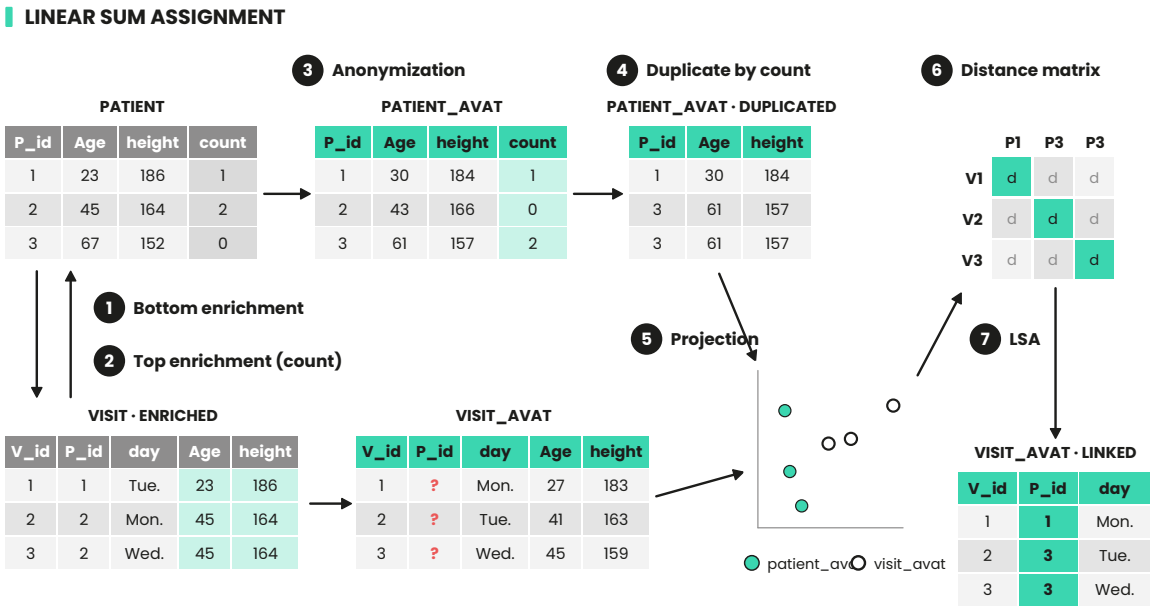


Illustration of minimum distance assignment methodology to recreate plausible match between parent and child table

- linear sum assignment

- Count of child events per parent id is avatarized in the parent table
- avatar rows of the parent table are duplicated according to avatarized count. This step is mandatory for the linear sum assignment algorithm as it applies to datasets of same shape.
- Common information of avatar parent duplicated table and avatar child enriched table are transformed into coordinates using principal component method (Principal Component Analysis or Factor Analysis of Mixed Data)
- Distances are computed between coordinates from parent table and avatar child enriched table.
- A linear sum assignment algorithm is applied to recreate the link between tables. each row from avatar parent duplicated table is associated with an avatarized visit following an optimization algorithm to minimize the total distance.
- Columns broadcasted from parent table to child table are dropped to create the final avatar child table

© Octopize — All rights reserved



One-to-one assignment minimizing the total distance; each avatar patient keeps its avatarized number of visits.

Illustration of linear sum assignment methodology to recreate plausible match between parent and child table

- original order assignment

```
    - This methodology consists in keeping the exact same link as original thus
avatarized visits of a specific individual will be reassigned to its avatar.
```

Use one of the three linking processes to generate plausible matches between parent table and child table information.

Advantages and drawbacks of different matching methods

Minimum distance assignment This is the methodology that ensures the highest level of privacy. It is faster than the LSA method, but has poorer preservation of aggregate distributions between tables, e.g. number of visits per patient.

Linear sum assignment This methodology shows the best balance between privacy and utility preservation. It preserves well aggregate distributions between tables while protecting rare events e.g. a patient having an abnormal number of visits. Because it is using an optimization algorithm, this methodology is also the slowest among the three.

Original order (sensitive) This methodology is fast and ensures a perfect preservation of aggregate distributions between tables. However, it is also the methodology that provides the least privacy guarantees.

Metrics computation

Since both privacy and utility are influenced not only by tables but also by the relationships between them, we calculate various approaches to assess risk at both the table level and their connections.

© Octopize — All rights reserved

Privacy metrics

To evaluate privacy, we use the metrics defined here. Given the characteristics of the multi-table context, we have identified three distinct evaluation scenarios :

Individual-Level Table

For each individual level table (`patient` and `doctor` in our example) we assess privacy through different scenarios :

STANDALONE

This scenario assesses the privacy of a table by computing privacy metrics directly on the table.

TO TOP ENRICHED

This scenario evaluates the protection of individuals in a table considering the information from its child table. The individual-level table is enriched with aggregated information from its child table, and privacy metrics are computed on this enriched table.

> **i Example**
>
> The `patient` table may show a low re-identification risk. However, if the linked `visit` table reveals that a specific patient has a unique number of visits, that count can put the patient at risk. By aggregating these visits information in the `patient` table, we can assess whether this additional information compromises patient privacy.

TO BOTTOM INFORMATION PROPAGATED

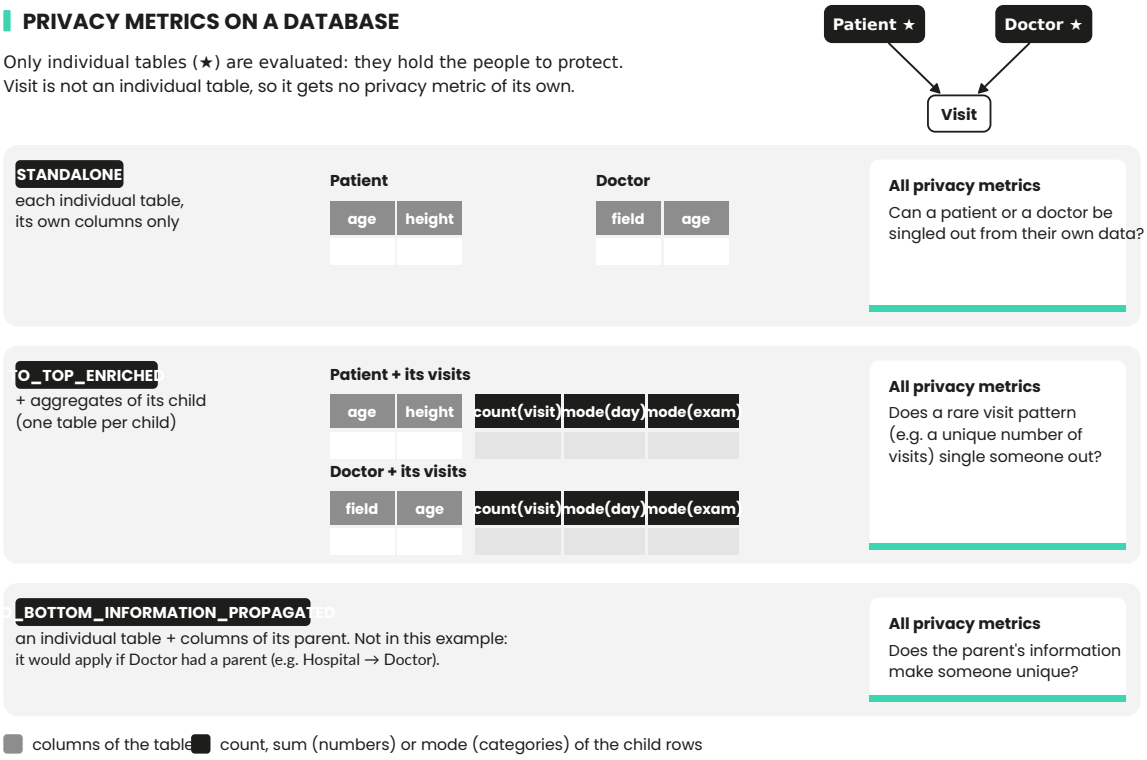
This scenario evaluates the protection of individuals in a table considering the information from its parent table. The individual-level table is enriched with information from its parent table, and privacy metrics are computed on this enriched table.

> **i Example**
>
> Consider an `hospital` table linked to the `doctor` table. If a specific hospital has only one doctor specialized in a rare field, the specialty information propagated from the `hospital` table to the `doctor` table could make that doctor unique. By propagating this information, we can assess whether the relationship compromises the doctor's privacy.

As a table can have parents and children, multiple scenarios are considered. > **i Example** > > Below is a representation of the tables for which privacy metrics are

© Octopize — All rights reserved

computed following the different scenarios: > >



Signal metrics

To evaluate signal, we use the metrics defined here. To evaluate the utility level of avatarization on the tables and relationships between table we defined 3 types of scenario :

Every table

STANDALONE : Signal metrics are computed for each table individually, without considering relationships with other tables.

Parent table

TO_TOP_ENRICHED : This scenario assesses the strength of the relationship between a parent table and its child table. The parent table is enriched with aggregated information from its child table, and we compute the hellinger distance. This process is repeated for each child of the parent table.

Child table

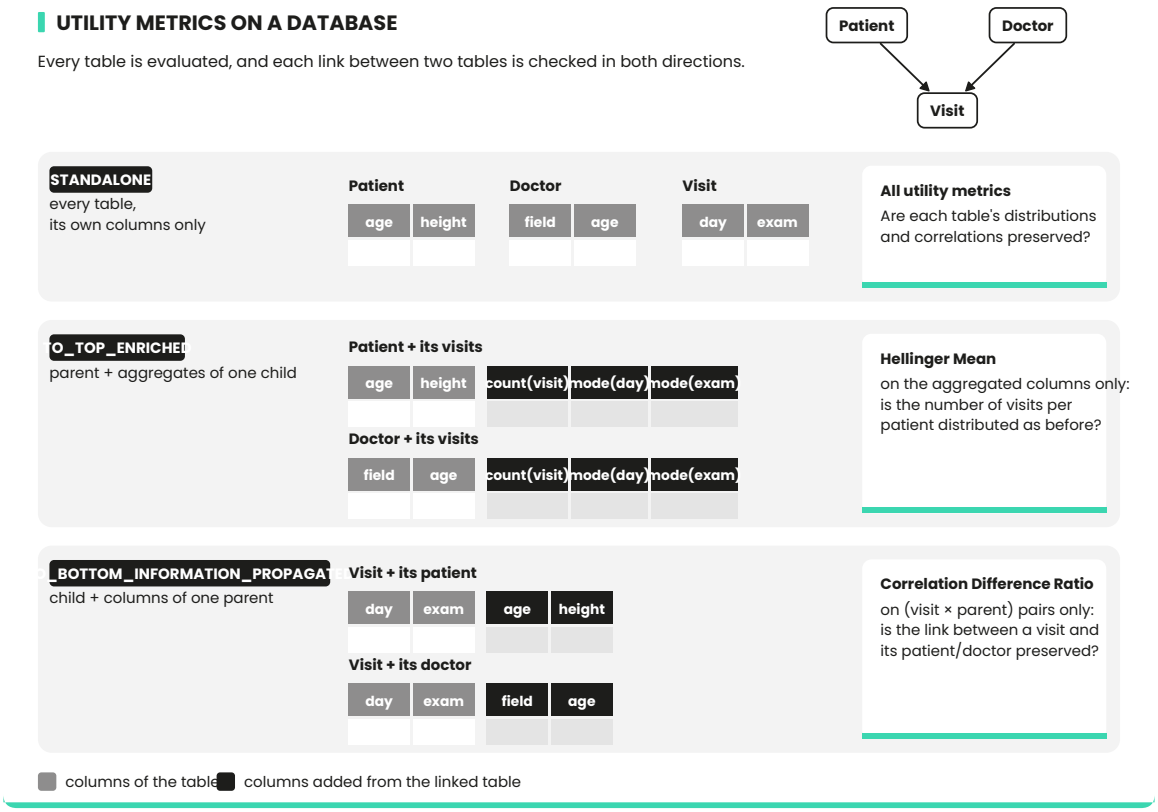
TO_BOTTOM_INFORMATION_PROPAGATED : This scenario evaluates the link between a child table and its parent table. The child table is enriched with broadcasted information from its parent table, and we compute the correlation difference ratio to assess the utility of the link.

As a table can have parents and children, multiple scenarios are considered.

i Example

© Octopize — All rights reserved

In our study case, here are the tables for which signal metrics are computed following the different scenarios :



Interpreting metrics

When evaluating privacy and signal metrics in a multi-table context, you must look beyond tables and consider the relationships between them. We provide a set of metrics to ensure you have the full context needed to judge anonymization quality.

While multi-table analysis can be complex, the following guidelines and summary table example will help you interpret your results and refine your avatarization parameters to balance privacy and utility.

The summary table highlights meta-metrics for a quick health check of your dataset. For a deeper dive into specific indicators, you can refer to detailed metrics results. This allows you to identify potential risks at a glance.

The rows represent the individual level table (the individuals you are protecting), while the columns represent all tables that could potentially leak information about those individuals.

ℹ Example

© Octopize — All rights reserved

Table Name	Patient	Doctor	Visit
Patient	STANDALONE(Patient)		TO_TOP_ENRICHED(Patient, Visit)
Doctor		STANDALONE(Doctor)	TO_TOP_ENRICHED(Doctor, Visit)

Table Name	Patient	Doctor	Visit
Patient	90%		60%
Doctor		85%	90%

In the table above, the `patient` table has a high standalone privacy score of 90%. However, when analyzed with the `visit` table, the score drops to 60%.

💡

Improve Your Results
If you notice a significant drop in privacy due to relationships between tables, consider the following adjustments:

- **Change the Matching Method**: Try using a different assignment method, such as linear sum assignment or minimum distance assignment, to see if it better balances privacy and utility.
- **Increase K**: Increase the k parameter in the source of the risk (in this case, the `visit` table).

Example Usage
For a hands-on walkthrough of multi-table anonymization, check out our Python notebook tutorial:

👉 Multi-table Tutorial

© Octopize — All rights reserved

Database Anonymization (Beta)

> ⚠️ **Beta Feature**
>
> Database anonymization is currently in **beta**. The API and behavior may change in future releases. Try it first on a database where a mistake has no consequence, for example one of the demo databases.

Avatar can read a database, anonymize it, and write the result into another database. The result is a working copy of a production database: the same tables, the same joins and about the same number of rows, without any real individual in it.

This page explains what Avatar does to each table and each variable of the database. It builds on the relational database method, which anonymizes linked tables together, and on pseudonymization, which replaces direct identifiers. To run one, see Database Runs in the user guide.

What Happens to Each Table

Each table gets one **treatment**:

Treatment	The table in the target
copy	the same rows, unchanged. Use it for a lookup table.
pseudonymize	the same rows. The personal values are replaced by fake values.
anonymize	synthetic rows, generated with the Avatar method.
purge	the table is there, with no rows. Use it for an audit log.
ignore	the table is not in the target at all.

purge and **ignore** are different on purpose. After **purge**, an application can query the table and find it empty. After **ignore**, the table does not exist.

Avatar never proposes **purge** or **ignore**: only you know that an audit log must be emptied.

What Happens to Each Variable

The web client and the Python client call a variable a **column**, as databases do. Inside a table, each variable gets one **action**:

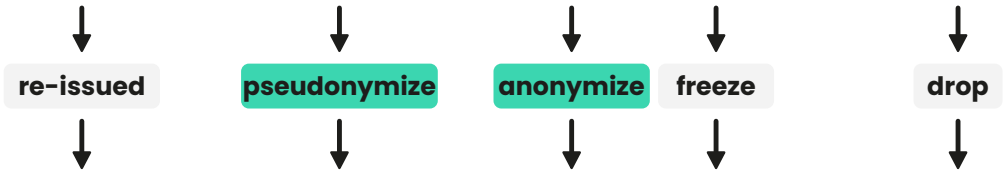
Action	The values in the target
anonymize	synthetic values, drawn from the distribution of the variable.
pseudonymize	fake values of the same kind (an email for an email), applied consistently.
freeze	the original values, unchanged.
drop	the variable is not in the target at all.

© Octopize — All rights reserved

WHAT HAPPENS TO EACH VARIABLE

Source table: customer

id	email	age	status	comment
17	ana.ruiz@example.com	34	active	called twice
18	li.chen@example.com	52	closed	prefers email
19	sam.okafor@example.com	41	active	moved in 2024



Target table: customer

id	email	age	status	comment
8107	noa.martin@example.net	37	active	not in the target
8108	eli.dubois@example.net	49	closed	
8109	mia.lopez@example.net	44	active	

To choose, ask what the users of the copy need from the variable:

- A value that must look real but belong to no individual, such as a name or an email: **pseudonymize**.
- A number or a category whose distribution matters, such as an amount or an age: **anonymize**.
- A code that an application filters on, such as a currency or a status, and that identifies no individual: **freeze**.
- A value that no user of the copy needs, such as a free-text comment: **drop**.

⚠️ Anonymization Versus Pseudonymization

Anonymization **resamples**: it draws values from the real distribution. A real value can appear in the copy, attached to a different individual. That is correct for an amount and wrong for an email address. Direct identifiers must be **pseudonymized**: see Pseudonymization.

⚠️ Privacy Risk of Frozen Variables

A frozen variable keeps its original values in the copy. Freeze only a variable that identifies no individual, on its own or combined with other variables. A frozen variable that identifies someone, or that can single out an individual

© Octopize — All rights reserved

together with other variables, leaks personal data into the copy. See Privacy metrics for the risks of singling out, linkability and inference.

A binary variable, such as a photograph or a scanned document, is emptied in the copy by default.

What Avatar Proposes

Classification reads the name of each variable first. A variable called `email` holds an email. When no rule applies, a model makes a guess, with a confidence. The proposal follows three rules:

What Avatar found	What it proposes
a variable classified as personal	**pseudonymize**
a primary key or a foreign key	nothing: the run handles it
any other variable	**anonymize**

A table where no variable is recognized is proposed for **anonymization**, not for copy. A table can identify individuals through a combination of variables that no rule detects.

You only need to confirm the variables that the classifier had an opinion about. Rejecting a proposal is also a decision, and Avatar records it as one.

Keys and Joins

Primary keys and foreign keys take none of the variable actions. The run **re-issues** every primary key: it replaces each key by another value and changes every foreign key that points to it. All joins still work in the target.

The consequence: key **values** change. If an application or a test fixture queries `WHERE country_code = 'FR'` and `country_code` is a key, it will not find the row. Keeping a key value unchanged is not available yet. Freezing a key is refused, because a frozen foreign key whose parent moved is a broken join that goes unnoticed.

Sequences

If a variable gets its values from a sequence (for example a PostgreSQL `SERIAL` column), the sequence in the target does not know about the copied rows. Ask the run to **continue the sequence** after the copied rows, or the first row that your application inserts collides with a copied one. You can also restart the sequence, or start it at a value of your choice.

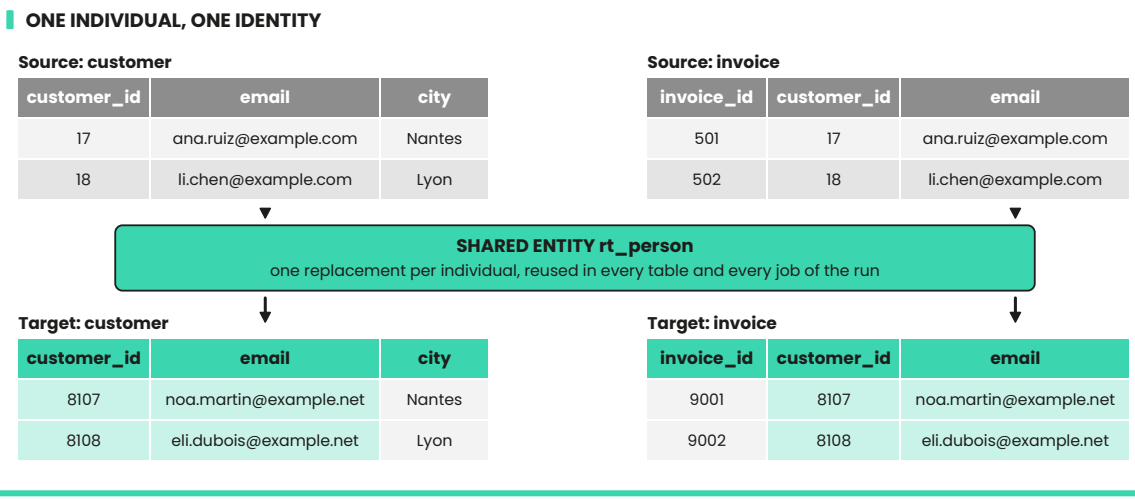
The Python client and the YAML configuration set sequences. The web client does not set them yet.

One Individual, One Identity: Shared Entities

A customer's email can be in the `customer` table and in the `invoice` table. If the two variables are pseudonymized separately, the same individual gets two identities, and the joins that a test database must exercise break.

A **shared entity** says that several tables describe the same thing (an individual, a product, a site) and names the variable that identifies it. The run replaces it **once**, and uses the same replacement in every table and in every job of the run.

© Octopize — All rights reserved



The web client calls a shared entity a **reference table**.

A shared entity can also span **several databases**: for example, a CRM and a billing system that describe the same customers. Run them together, in one run, so that a customer gets the same fake identity in both copies.

> ⚠️ **Identities Depend on the Set of Databases**
>
> A replacement depends on the full set of entities that the run collects. If you add a third database to the run later, every fake identity changes, in all the copies. A refreshed set of copies with an extra source shares no individual with the previous one.

Groups of Tables

Tables linked by foreign keys must be anonymized together, as for any relational database. Tables that are not linked are separate datasets that happen to be in the same database. Avatar finds these **groups** and runs **one job per group**. You do not arrange this yourself.

Status, failure and retry are per group. A failed group does not undo the work of the other groups. All the jobs of a run use the same seed, so that they agree about every shared entity. Each group produces its own privacy and utility metrics and reports.

In Practice

- Database Runs: how a run proceeds, and what the target database must satisfy.
- Run a Database Copy in the Web Client
- Run a Database Copy with the Python Client

© Octopize — All rights reserved

Time Series Data

Time series data is a specific type of dataset where a variable is recorded multiple times over a given period. Examples include sensor readings, financial data, and medical signals such as electrocardiograms (ECGs).

Unlike traditional tabular data, time series data consists of multiple records (rows) per individual. Crucially, valuable insights often emerge from the evolution of values over time rather than from individual observations. For example, a sudden increase in blood pressure is only meaningful when analyzed in relation to previous readings.

A dataset typically contains multiple time series variables, each with distinct recording frequencies and durations. For instance, a vehicle may collect telemetry data from multiple sensors, each with different sampling rates.

If a time series variable contains only a few measurements per individual (e.g., fewer than five), it may be preferable to structure it as standard tabular data by assigning each measurement to a separate column or to use the multi table approach. The method described here applies to time series datasets with a **larger number of time steps**.

To be fully informative, time series data is often accompanied by **static tabular data** (e.g., patient age, gender, or device specifications). This complementary dataset is referred to as **standard data** and can be anonymized alongside the time series variables.

Avatars can accommodate multiple time series variables, even with differing lengths (time between first and last recordings) and frequencies (number of observations per second). However, when time series variables differ significantly in length and frequency, they should be stored in separate datasets.

Also note that the Time Series approach could be used embedded within a multi table anonymization (standard data being the parent patient data for instance and the time series itself being the child table.)

Scope of application

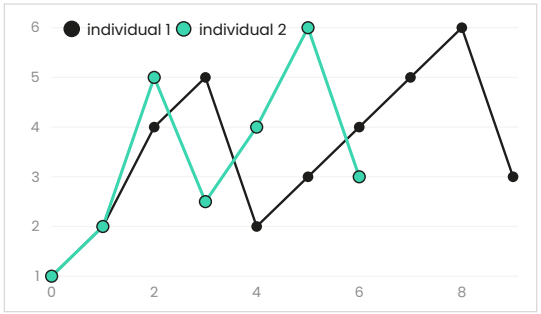
To ensure effective anonymization, time series data must meet the following conditions:

- **Numeric Data Only:** Time series variables must be numerical (categorical data is not supported).
- **No Missing Values:** Time series variables must not contain missing values, though missing values are allowed in the accompanying standard data.
- **Registered Time Series:** Within a dataset, all time series must be **registered**, meaning:
  - Every variable has a value for each time step.
  - The time steps are the same across all variables.
  - Time series at a given time step must be comparable.

  We provide a tool to perform this registration prior to the anonymization.

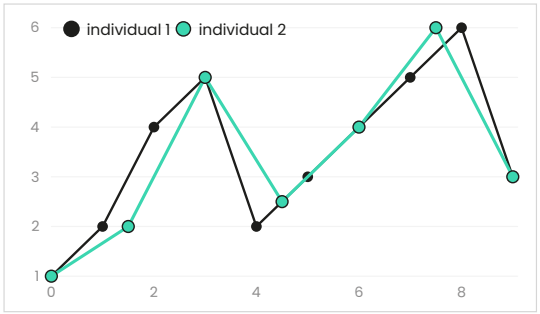
© Octopize — All rights reserved

NON-REGISTERED



non-registered time steps

REGISTERED



registered time steps

- **Regular Time Intervals:** Time series data must be evenly spaced over time (i.e., fixed intervals between observations). We also provide a tool for this.
- **Dataset Size Constraint:** The number of individuals must be at least equal to the product of the number of time steps and the number of time series variables.
- **Mandatory Identifier Variables:** Time series data must contain:
  - An **individual identifier** that is the primary is caseof using time series approach with multi-table.
  - A **time variable**
  - The variables to be anonymized

  This flat data format follows a standard convention for time series representation.

Identifier	Time step	Variables to anonymize		
ID	**t**	**variable 1**	**variable 2**	**...**
1	0	10	0.3	...
1	1	12	0.1	...
1	2	14	0.4	...
2	0	8	1.4	...
...	...	...	...	...

© Octopize — All rights reserved

Scalability:

This method can efficiently process datasets ranging from a few dozen to several thousands of time steps for multiple time series. Larger volumes of data are handled using a batch-parallelized approach, ensuring fast computation times.

Cyclic Data Specificities

Cyclic data, where patterns repeat periodically (e.g., ECG signals), is **not natively supported** by this method. This is because cycles of different lengths across individuals lead to unregistered cycles, distorting the anonymized output.

To properly anonymize cyclic data, cycles should first be extracted and processed individually before reconstructing the full signal. If this preprocessing step is not feasible, anonymization at the cycle level is recommended.

Our scientific publication [^1] details an anonymization approach for ECG signals.

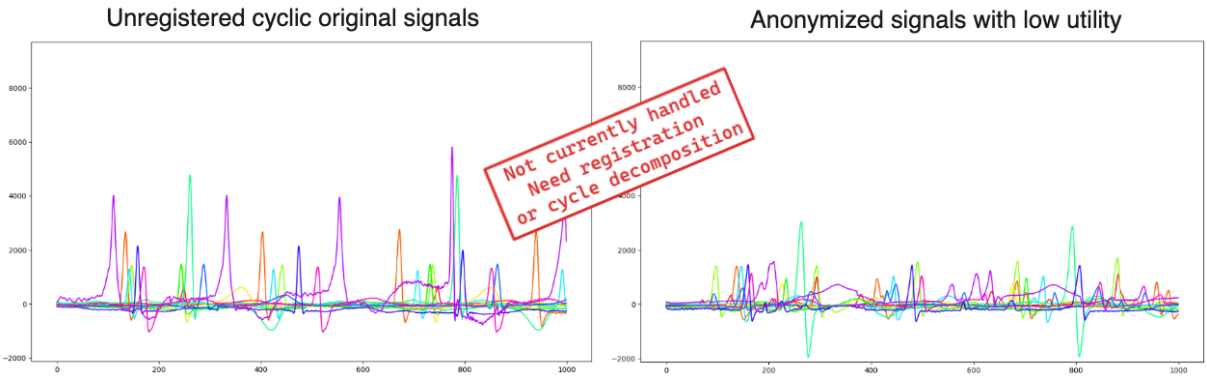


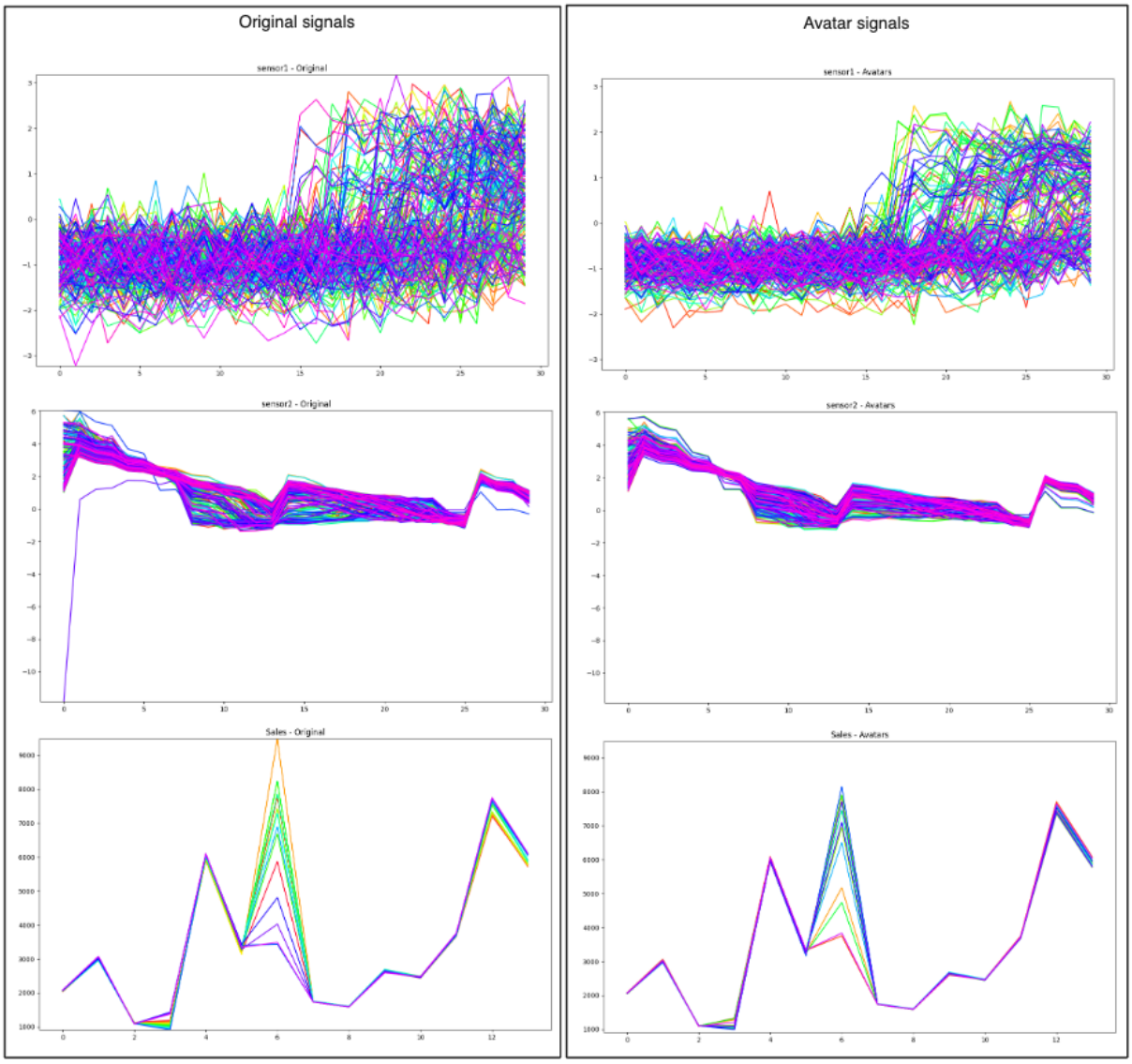
Figure 2: Example of cyclic signals handled without registration or cycle decomposition

Output

Synthetic data return by this method have the following properties:

- ☒ **Similar granularit:y** Same number of time steps as after registration. Note that data could be unregistered afterwards using same given tools as the preprocess.
- ☒ **Similar time patterns:** Univariate information is preserved.
- ☒ **Similar pattern clustering:** Multivariate information is preserved.

© Octopize — All rights reserved



Example of synthetic time-series output versus original data

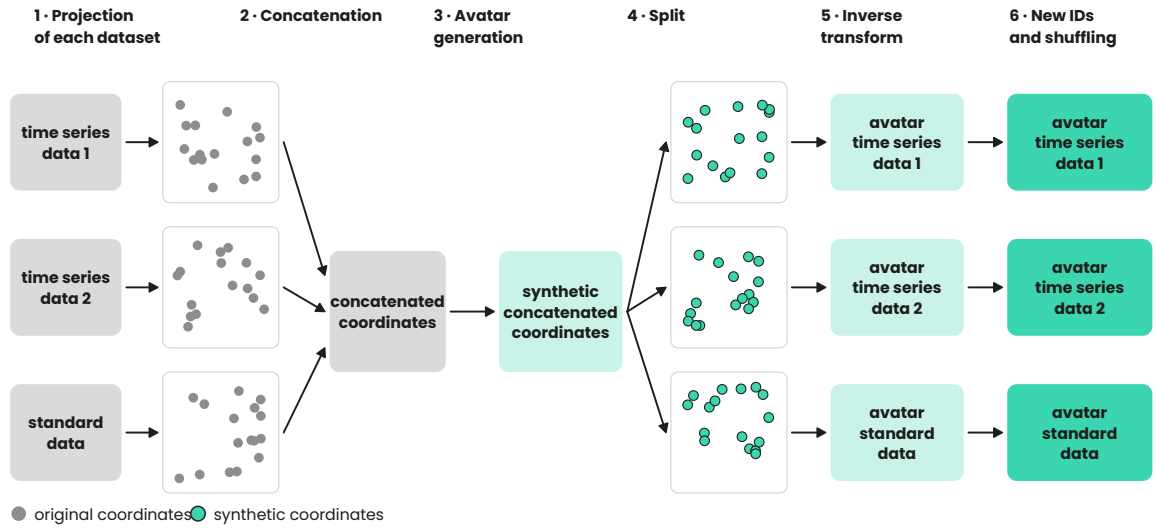
Core Algorithm

To anonymize time series data, the overall process is similar to the one of standard tabular data described here, to the exception of the use of time series specific projection.

1. The first step is to project the data from each table in a multidimensional space where one row of data represents one individual. The standard data is projected like in the case of standard tabular data using a principal component method (Principal Component Analysis or Factor Analysis of Mixed Data). In order to preserve the relationships, trends and any other information that may exist between readings of a variable over time, specific projection methods (FFT, FPCA,...) are used for time series data. Currently, projection of such data is performed using Functional Principal Component Analysis (FPCA).
2. Once each dataset is projected into its own space, the projected data from the different datasets is concatenated in order to yield a single set of coordinates. Following concatenation, one row contains information coming from all datasets about a specific individual.

© Octopize — All rights reserved

3. (Optional): In case of multiple time-series, if we are willing to keep multivariate information between time series we can either project all time series using Multivariate FPCA or by applying a PCA on concatenate coordinates.
4. After concatenation, synthetic coordinates are generated as explained here.
5. The synthetic coordinates are split so that projected variables are grouped according to the input datasets
6. The projection is reverse-transformed so that the synthetic data is expressed in its original form.
7. Finally, individual identifiers are re-generated and datasets shuffled



Example Usage

For a hands-on walkthrough of time series table anonymization, check out our Python notebook tutorial:

👉 Time Series Tutorial

© Octopize — All rights reserved

Data Augmentation

Data augmentation with Avatar leverages the anonymization method's stochastic properties to generate additional synthetic individuals that preserve the statistical characteristics of your original dataset. This capability addresses common challenges in machine learning and data analysis, such as class imbalance and insufficient sample sizes, while maintaining privacy guarantees.



Data augmentation is designed to work seamlessly with Avatar's **tabular**, **multi-table**, and **time series** methodologies, providing flexible augmentation capabilities across different data structures.

Overview

The Avatar method's randomized local modeling process, originally designed for privacy protection, naturally enables controlled data augmentation. Unlike traditional augmentation techniques that may introduce unrealistic values or compromise privacy, Avatar-based augmentation:

- **Preserves statistical properties** – Maintains distributions, correlations, and multivariate relationships
- **Stays within data boundaries** – Generated values remain within the original data space (e.g., ages 20-60 stay within that range)
- **Ensures privacy** – Augmented data inherits the same privacy guarantees as standard avatarization
- **Maintains data structure** – Keeps the same variables, encoding, and granularity

When to Use Data Augmentation

Data augmentation with Avatar is particularly valuable in the following scenarios:

1. Class Imbalance

When your dataset has underrepresented classes that can negatively impact model training and prediction accuracy. For example:

- Binary classification with 10% minority class samples
- Multi-class datasets where some categories have insufficient examples
- Medical datasets where rare conditions are underrepresented

2. Insufficient Sample Size

When you need more data to:

- Train robust machine learning models
- Perform statistical analyses with adequate statistical power
- Test model performance with larger datasets
- Generate representative datasets for development and testing

3. Privacy-Preserving Dataset Expansion

When you need to:

- Share larger datasets with partners while maintaining privacy

© Octopize — All rights reserved

- Expand datasets without collecting additional sensitive data

Augmentation Strategies
Avatar offers four distinct augmentation strategies, each designed for specific use cases:

1. Multiplier Augmentation
Strategy: float value > 1

Purpose: Multiplies the entire dataset size by a specified factor while preserving class proportions.

When to use:

- Dataset is too small overall, regardless of class distribution
- You want to preserve existing class proportions
- Testing model performance with scaled datasets
- Generating multiple independent synthetic datasets

Result: Using a multiplier of 3, a dataset with 1,000 samples becomes 3,000 samples with identical distributions and class proportions.

Result: Using the following proportions `{"Class A": 2, "Class B": 1.5}`. Class A is doubled, class B increased by 50%, and other classes remain unchanged.

2. Minority Class Balancing
Strategy: `minority`

Purpose: Increases the minority class to match the second-smallest class size in classification problems.

When to use:

- Binary classification with significant class imbalance
- You want to balance the minority class
- The minority class has too few samples for effective model training

Result: If your dataset has 400 samples of class A and 100 samples of class B, the augmented dataset will have approximately 400 samples of each class.

3. Not-Majority Class Balancing
Strategy: `not_majority`

Purpose: Balances all non-majority classes to match the largest class in multi-class scenarios.

When to use:

- Multi-class classification with multiple underrepresented classes
- You want all classes to have similar representation
- Training algorithms sensitive to class imbalance

Result: If your dataset has classes with 500, 200, and 150 samples respectively, the augmented dataset will have approximately 500 samples for each class.

© Octopize — All rights reserved

4. Custom Multipliers (Python client only)

Strategy: Dictionary mapping classes to custom multiplier

Purpose: Apply specific augmentation factors to individual classes for fine-grained control.

When to use:

- You have specific requirements for class distribution
- Standard strategies don't match your needs
- You want to apply different multiplication factors to different classes

Strategy Comparison

Strategy	Purpose	Target Column Required	Preserves Proportions	Python client Parameters
`multiplier`	Scale entire dataset	No	Yes	`data_augmentation_strategy` (float > 1)
`minority`	Balance binary classes	Yes	No	`data_augmentation_strategy=minority`, `data_augmentation_target_column`
`not_majority`	Balance all classes	Yes	No	`data_augmentation_strategy=not-majority`, `data_augmentation_target_column`
`custom multiplier`	Fine-grained control	Yes	Custom	`data_augmentation_strategy` (dict), `data_augmentation_target_column`

Privacy Implications

Data augmentation affects privacy evaluation in different ways. Privacy evaluation of augmented data relies on the same set of metrics used for traditional anonymization but requires custom evaluation strategies to assess its impact

Computation Types

When augmentation is applied, privacy metrics are computed across multiple scenarios:

© Octopize — All rights reserved

Computation Type	Original Data used	Avatar Data used	Privacy Question
STANDALONE	All originals	All avatars	Can any avatar record compromise the privacy of its corresponding original?
BARYCENTERED	All originals	All avatar data, barycentered by original record	Can the aggregation of all avatars for an original record compromise its privacy?
CLASS_BALANCING_BARYCENTER	Only augmented originals	Only augmented avatars, barycentered by original record	Does augmentation increase privacy risk for originals that were augmented multiple times?

Privacy Guarantees

- **Same fundamental privacy** – Augmented data maintains the same privacy guarantees as standard avatarization
- **Additional evaluation** – The `CLASS_BALANCING_BARYCENTER` and `BARYCENTERED` computations specifically assess risks introduced by creating multiple avatars from the same original
- **Risk assessment** – Privacy metrics quantify the re-identification risk for both original and augmented records

Example Usage

For a hands-on walkthrough of data augmentation, check out our Python notebook tutorial:

👉 Data Augmentation Tutorial

Further Resources

- **Core methodology:** Single Tabular Data for underlying algorithm details
- **Privacy metrics:** Privacy Metrics for understanding privacy evaluation

© Octopize — All rights reserved

Pseudonymization (Beta)

> ⚠️ **Beta Feature**
>
> Pseudonymization is currently in **beta**. The API and behavior may change in future releases.

Pseudonymization is a data protection technique that replaces personally identifiable information (PII) with artificial identifiers, reducing the risk of direct re-identification while preserving data structure. In the Avatar platform, pseudonymization is available as an **optional post-processing step** that complements the anonymization process.

> 📝
>
> Pseudonymization and anonymization serve different but complementary purposes. Anonymization (via the Avatar method) protects against re-identification through statistical transformation, while pseudonymization replaces direct identifiers with substitutes. Used together, they provide layered privacy protection.

Why Pseudonymize?

When working with sensitive datasets, direct identifiers such as names, email addresses, or social security numbers must be handled carefully. While the Avatar anonymization method operates on pseudonymized data (without direct identifiers), you may need to:

- **Preserve PII columns in the output** with realistic but fake replacements for testing or demonstration purposes
- **Maintain referential integrity** across related tables — ensuring that a patient ID in a child table still correctly references its parent
- **Comply with data handling policies** that require PII to be systematically replaced rather than simply removed
- **Streamline your workflow** by handling pseudonymization and anonymization in a single pipeline

How It Works

Pseudonymization in the Avatar platform is applied **after** the anonymization process completes. The pipeline works as follows:

1. **Before anonymization**: PII columns are extracted and set aside — they do not participate in the Avatar method's statistical modeling
2. **Anonymization**: The Avatar method generates synthetic data based on the non-PII variables
3. **After anonymization**: Each PII column is pseudonymized using the configured strategy, and the resulting values are reattached to the anonymized output
4. **Output**: The final dataset contains anonymized statistical data alongside pseudonymized identifiers

© Octopize — All rights reserved

This separation ensures that the anonymization quality is unaffected by the presence of direct identifiers.

Pseudonymization Strategies

Six strategy classes are available, each suited to different use cases. All strategies (except `HashSha256Strategy` and `ConstantStrategy`) accept a `consistent` parameter: when `True` (the default), the same source value always maps to the same replacement within a pipeline run, preserving referential integrity across tables.

`FakeDataStrategy` — Realistic Synthetic Values

Generates plausible fake values that match the type of PII. For example, an email column produces realistic email addresses, a name column produces realistic names.

- **Best for**: Testing, demonstrations, and scenarios where output data must look realistic
- **Requires**: The type of PII must be specified via `pii_type` (e.g., `PiiType.EMAIL`, `PiiType.FIRST_NAME`)
- **Example**: `FakeDataStrategy(pii_type=PiiType.EMAIL)`

`high_variability` parameter

The `high_variability` parameter controls the trade-off between speed and uniqueness of the generated values:

- **`high_variability=False`** (default): Avatar samples from a pre-generated pool of fake values using indexing. This is significantly faster, but different source values can receive the same fake value (collision possible).
- **`high_variability=True`**: Avatar generates a unique fake value for each row or for each unique source value (depending on `consistent`). This avoids collisions but is slower.

Use the default (`False`) in most cases. Set `high_variability=True` only when you need to guarantee that no two distinct source values receive the same fake replacement. Note that this is subject to the limits of the underlying library

`HashSha256Strategy` — Deterministic Hashing

Replaces each value with its SHA-256 cryptographic hash. The same input always produces the same output.

- **Best for**: Consistent key replacement across datasets and runs
- **Behavior**: Deterministic — identical inputs always yield identical outputs
- **Note**: Hashed values are not human-readable. Avoid this strategy for low-entropy columns (e.g. short codes or integers) as they are vulnerable to brute-force dictionary attacks
- **Example**: `HashSha256Strategy()`

`Uuid4Strategy` — Random Unique Identifiers

Replaces each value with a randomly generated UUID (version 4).

- **Best for**: Primary keys and identifiers where uniqueness matters more than readability
- **Example**: `Uuid4Strategy()`

© Octopize — All rights reserved

`ConstantStrategy` — Fixed Replacement

Replaces every value in the column with the same fixed string, regardless of the original content.

- **Best for**: Fully suppressing or redacting a column
- **Example**: `ConstantStrategy(value="REDACTED")`

`IntegerStrategy` — Pseudonymous Integer IDs

Maps each unique source value to a unique pseudonymous integer. The mapping is randomized so that the original sort order is not preserved.

- **Best for**: Numeric identifiers (e.g. customer IDs, account numbers) when downstream code expects an integer type
- **Example**: `IntegerStrategy()`

`SpecificIdStrategy` — Structured Pattern-Based IDs

Generates structured identifiers from a user-defined pattern, combining literal characters with placeholders.

Placeholder	Description
`#`	Random digit (0–9)
`?`	Random letter (case controlled by `letter_case`)
`^`	Random alphanumeric character
`{{col}}`	Value of another column for the same row

The `letter_case` parameter (a `SpecificIdLetterCase` value) controls letter casing: `UPPER`, `LOWER`, or `BOTH` (default).

- **Best for**: Domain-specific identifiers with a required format (e.g. employee codes, reference numbers)
- **Example**: `SpecificIdStrategy(pattern="EMP-####")`, `SpecificIdStrategy(pattern="{{dept}}-???##", letter_case=SpecificIdLetterCase.UPPER)`

Supported PII Types

The platform recognizes the following types of personally identifiable information. These types guide the `FakeDataStrategy` in generating appropriate replacement values:

© Octopize — All rights reserved

PII Type	Description	Example Original	Example Pseudonymized (FakeDataStrategy)
Email	Email addresses	jane.doe@corp.com	sarah.miller@example.net
First Name	Given names	Jane	Sarah
Last Name	Family names	Doe	Miller
Full Name	Complete names	Jane Doe	Sarah Miller
Phone	Phone numbers	+33 6 12 34 56 78	+33 6 98 76 54 32
SSN	Social security numbers	1 85 12 75 108 042	2 91 03 69 205 118
Address	Physical addresses	12 Rue de Rivoli	45 Avenue des Champs
Free Text	Arbitrary text fields	Patient note…	Lorem ipsum…

💡 The PII type only needs to be specified when using `FakeDataStrategy`. All other strategies work independently of the PII type.

Multi-Table Referential Integrity

When working with multi-table datasets (e.g., a patients table linked to a visits table), pseudonymization automatically maintains **referential integrity**:

- A pseudonymized primary key in the parent table is consistently applied to matching foreign keys in child tables
- This ensures that relationships between tables remain valid after pseudonymization
- Child table foreign keys automatically inherit the parent's pseudonymization mapping — no additional configuration is needed

For example, if `patient_id` in the patients table is pseudonymized with UUID, the same mapping is applied to `patient_id` in the visits table, preserving the parent-child relationship.

Getting Started

Pseudonymization is configured per column when setting up your avatarization parameters. You pass a strategy object for each column that should be pseudonymized — no configuration is needed for foreign key columns, which automatically inherit their parent's mapping.

For technical documentation, code examples, and step-by-step tutorials, refer to the **Python Client documentation**:

- User Guide — How to pseudonymize PII columns
- Tutorial — Pseudonymization walkthrough

© Octopize — All rights reserved

Batch Processing

Avatar processes large datasets in **batches**. A dataset of any size can be anonymized, and the memory a job needs depends on the batch size, not on the size of the dataset.

A dataset smaller than one batch is processed in a single batch. In that case, nothing on this page changes your results.

How it works

1. **Cut.** The dataset is divided into batches of a fixed number of records. The batch size is a setting of the deployment, not a job parameter.
2. **Random assignment.** Records are assigned to batches at random. Each batch is a representative sample of the whole dataset, even when the file is sorted by date, identifier or class.
3. **Anonymization.** Each batch is anonymized independently. The neighbours used to generate an avatar come from the same batch as the original record.
4. **Assembly.** The batches are put back together. The avatars keep the order of the original records, so the 1-for-1 mapping is preserved. The shuffled output is shuffled across the whole dataset, not within each batch.

Multi-table and time series data

A batch never separates records that must be modeled together:

- In a multi-table dataset, a parent record and its child records are always in the same batch. This depends on valid primary and foreign keys.
- In a time series dataset, an individual and all their time series are always in the same batch.

An individual with a very large number of linked records is kept whole. Its batch can then be larger than the configured size.

Metrics

The dataset checks run on every record. The privacy and utility metrics are computed on the first batch, which is a representative sample. Each avatar is compared with the original records of the batch it was generated from. See the technical assessment report.

What this means for you

For users:

- There is no parameter to set. Batch processing applies automatically.
- The same dataset with the same seed gives the same batches and the same results.
- With data augmentation, the additional records of a batch are generated from the original records of that batch.

For administrators of a self-hosted deployment:

- The memory that a job uses depends mainly on the batch size. A larger dataset takes more time, but does not need proportionally more memory.
- Contact Octopize before you change the batch size: it changes the memory needs and the number of neighbours available to each record.

© Octopize — All rights reserved

Metrics

Synthetic data evaluation

Synthetic anonymous data generation aims to faithfully replicate the characteristics of the original training data while ensuring the privacy of the individuals from whom the information originates. To ensure optimal quality of the generated data, it is crucial to quantify both **privacy** and the preservation of **utility** in the synthetic dataset.

We have designed a set of metrics to evaluate these two aspects. These metrics provide an objective measure of the level of privacy protection across multiple criteria while verifying that the synthetic data retains analytical value comparable to the original dataset across various evaluation levels.

More details on each set of metrics here:

- Privacy
- Utility

The Importance of Context

The **context of use**, the data **sensitivity** and the **utility preservation goal** are critical considerations in any anonymization process. The context defines the specific objectives and potential threats to mitigate, shaping the methods and parameters employed. For instance, data intended for research purposes may require a different approach than data shared publicly. Similarly, the sensitivity of the data—whether tied to its nature (e.g., health, financial) or its granularity—dictates the level of protection needed. Finally specific uses may require to keep highly granular yet very re-identifying information. Effective anonymization must account for all these dimensions to ensure the optimal result both in terms of privacy and utility.

© Octopize — All rights reserved

Privacy

The technical evaluation of synthetic data anonymity is based on three fundamental criteria defined here by the European Data Protection Board (EDPB, formerly known as WP29). These criteria remain central to assessing the regulatory compliance of an anonymized dataset, but the EDPB's draft Guidelines 02/2026 on anonymisation also stress that the question is **contextual and risk-based**.

In other words, the goal is not to determine a binary status once and for all, but to assess the **residual re-identification risk** for the actual use case, the recipient, and the realistic attack scenarios that could be carried out.

This is also why the EDPB framing is not a "100% protection for every metric" model. A framework can still be considered acceptable when, overall, the residual risk remains low enough for the intended use, even if one metric is weaker under a specific attack scenario. Conversely, a poor result on a specific criterion is not by itself a legal verdict: it triggers a contextual review of the risk, the attacker model, and the safeguards that can reduce the residual risk.

This is in line with recent machine-learning research. A position paper presented at ICML 2026 argues that the privacy of synthetic data should be treated as an explicit claim, with a stated threat model, that can be tested and contested, rather than as a property assumed from the generation process. It also notes that rare and minority records are the most exposed (Zhao, Januszewicz and Jung, *Position: Privacy Is a Claim, Not a Property of Synthetic Data*). The metrics below follow that approach by simulating explicit attack scenarios.

Reasonable means and the attacker model

The legal test is not based on whether a dataset is theoretically impossible to attack under any imaginable circumstance. It is based on whether a **reasonable attacker**, with the **means reasonably likely to be used** in the real world, could identify individuals.

This includes:

- access to public or semi-public external datasets,
- standard linkage methods,
- open registries or administrative records,
- a realistic level of technical capability and time investment,
- and the actual constraints of the data-sharing environment.

A dataset may therefore be considered acceptable for a controlled or purpose-limited use, while requiring a stricter assessment for open or large-scale distribution.

> ⚠️ **Draft guidelines**
>
> The EDPB Guidelines 02/2026 on anonymisation are a **draft** under public consultation until 30 October 2026. They are not final and may change before their final adoption. The points below that refer to them describe the draft.

© Octopize — All rights reserved

The relevant EDPB source is the draft anonymisation guidance on the means that are "reasonably likely to be used": EDPB Guidelines 02/2026 on anonymisation.

For a practical checklist on how this is assessed, see How do we assess whether a means is reasonable?.

The draft EDPB guidelines distinguish two practical ways to do this:

- **Simplified approach**: a conservative, worst-case assessment, suitable for high-risk scenarios such as open data or broad external sharing. It asks whether a motivated attacker, using publicly available information or common external sources, could realistically re-identify a record.
- **Contextual approach**: a more precise assessment tailored to the real conditions of use, including recipient, purpose, data sensitivity, available background knowledge, and security measures. This is the approach used when the dataset is shared under controlled-access conditions or for a known, limited use case.

The three criteria are:

- **Singling Out / Record Isolation**: the risk that an attacker can isolate a record or a rare attribute combination in a dataset and, using reasonably available background knowledge, attribute it to a specific physical individual.
- **Linkability**: the ability to connect individuals to an external data source containing shared characteristics.
- **Inference**: the possibility of deducing, with significant probability, information about individuals from the anonymized dataset.

This terminology is useful to keep in mind: in modern privacy literature, "record isolation" is often used to describe the same underlying mechanism as "singling out," but the regulatory concept remains "singling out" when the isolated record can be associated with a real person.

Evaluation of the Criteria

To assess these criteria, we have developed metrics that simulate attack scenarios inspired by state-of-the-art practices. These tests systematically verify the dataset's compliance with each criterion.

The regulations encourage a comprehensive evaluation that includes:

1. **"Worst-case" scenarios**: Extreme cases, though unlikely, where privacy risks are maximized.
2. **Likely but less risky scenarios**: Realistic situations where attack probabilities are higher but the impacts remain moderate.

More details about our privacy metrics here:

- Singling out
- Linkability
- Inference

All privacy metrics were designed following a **maximization** objective where high values stands for high privacy and conversely, except Attribute Inference, which reports the attacker's accuracy (lower is better).

© Octopize — All rights reserved

All these metrics are then automatically consolidated into a **technical report assessing the residual re-identification risk**.

Learn more about the technical assessment →

Risk Analysis Report (PIA)

In addition to the technical assessment, we generate a **Privacy Impact Assessment** (PIA) risk analysis report to help determine whether the residual risk is compatible with the intended use.

Where the technical evaluation quantifies a theoretical risk, the PIA evaluates the **likelihood of attacks occurring** and their real-world **impact** (data sensitivity, trust in recipient, etc.).

Learn more about the impact assessment →

The Importance of Context

Risk evaluation heavily depends on the **context of use**:

- Sharing low-sensitivity data internally with a trusted third party presents minimal risks.
- Conversely, sharing sensitive data as open data significantly amplifies risks and requires more stringent evaluation and mitigation measures.

Anonymization is **adapted to the risk associated with the use** of the data. Risk is mainly defined by the **sensitivity** of the data and the **level of trust** placed on the recipient. The result of the metrics provides quality control downstream of the anonymization and ensures that it is impossible in practice for an attacker to re-identify the generated data using reasonable means. Metric results can therefore be considered satisfactory for one use and insufficient for another.

Importantly, data anonymity is **not a binary status** — it depends on the context of use. Measuring privacy metrics allows to have a technical overview of the **residual reidentification risk**. If the residual risk remains too high regarding the context of use, additional security safeguards can be applied to reinforce protection and ensure regulatory compliance. These may include:

- Encryption of synthetic data at rest and in transit,
- Making data available within a secure controlled-access environment,
- Time-limited access or expiring data access sessions,
- User authentication and auditing mechanisms,
- Applying data minimization or filtering by population subgroups,
- Restricting re-identification-sensitive variables or removing uncommon outliers.

By following these principles, our tools enable you to quantify privacy risks and ensure robust compliance with regulatory requirements.

© Octopize — All rights reserved

Singling out

Singling out is a fundamental criterion in evaluating the anonymity of synthetic data. It refers to the risk that an individual can be uniquely identified within a dataset based on the available information.

> **i Terminology note**
>
> In current privacy literature, this mechanism is often described as record isolation. In practice, it refers to the same idea as singling out: an attacker can isolate a row or a rare combination of attributes within a dataset. The criterion becomes relevant only when that isolated record can be linked to a specific physical individual using reasonably available background knowledge.
>
> In other words, isolating a rare attribute combination in a table is not, by itself, a privacy failure if it cannot be associated with a real person. The risk becomes material when the isolated combination can realistically be tied to one person.

> **i Example**
>
> Consider a dataset containing employee information within an organization. An attacker who knows an employee's department, age, and seniority could use this information to isolate the individual, especially if these attributes make them unique—for example, being the only employee with more than 10 years of seniority in a small team.

List of metrics measuring singling out criteria

- Hidden rate
- Local Cloaking
- Distance to closest
- Closest distances ratio
- Closest Rate
- Column direct match protection
- Row direct match protection
- Membership Inference Attack

Factors Influencing the Risk of Singling Out

The risk of singling out increases proportionally with the amount and precision of information available to an attacker. The more contextual data (e.g., age, location, profession) an attacker possesses, the more likely they are to isolate an individual in the dataset.

This criterion is particularly critical for individuals with **remarkable values** (*outliers*), meaning rare or atypical characteristics. For instance, someone living in a sparsely populated area or having a unique combination of traits in the dataset is far more vulnerable to singling out.

Importance of Mitigation

To limit the risks of singling out, the metrics we have developed systematically assess the likelihood that an attacker, with varying levels of prior knowledge, could

© Octopize — All rights reserved

isolate an individual. These analyses help identify and address vulnerabilities before the data is used or shared.

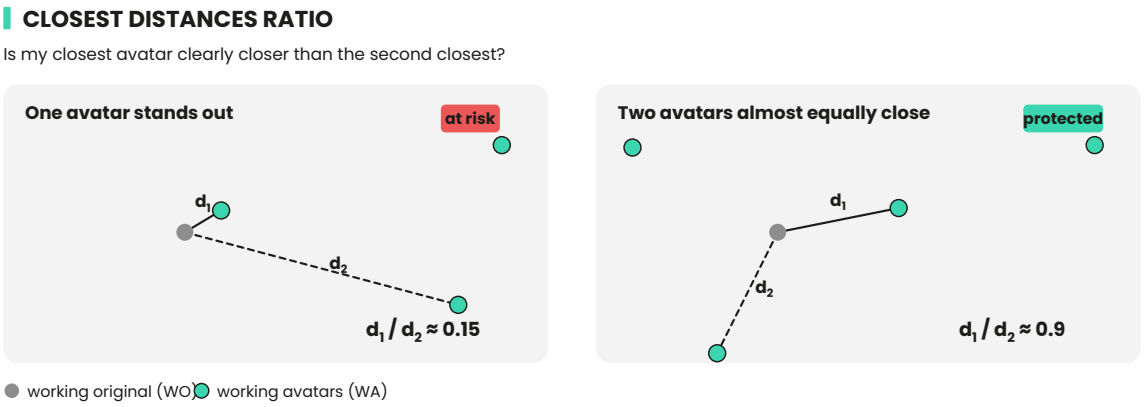
© Octopize — All rights reserved

Closest Distances Ratio

The **Closest Distances Ratio** is a **local density metric**.
It computes the **median ratio** between the Euclidean distance separating an individual from their **closest avatar**, and the distance separating that same individual from their **second closest avatar**.

For interpretability, this value is compared to a reference value.
The final score is the ratio between the **Closest Distances Ratio** computed on the avatars and the one computed on a reference dataset.
This comparison makes it possible to evaluate whether avatars reproduce a local density around original individuals that is similar to what is observed in the reference data.

The approach is also applied to **outliers**, by computing the **Closest Distances Ratio** for individuals who exhibit low local density (isolated points).



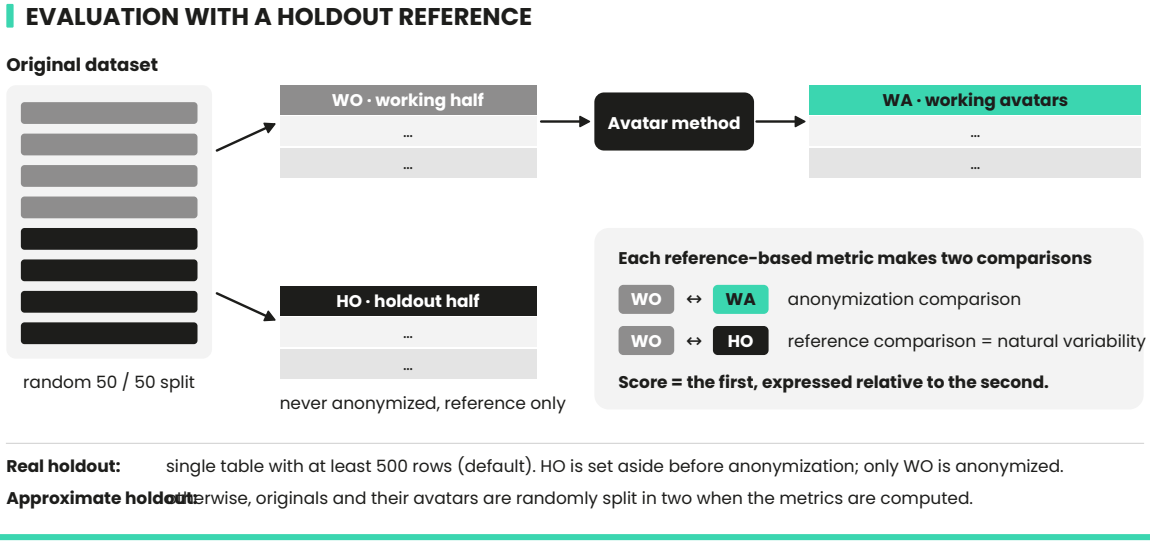
📝 Working half (WO), holdout half (HO), working avatars (WA)

This is a **reference-based** metric: it compares the anonymization (**WO** vs its avatars **WA**) against the natural variability of the real data (**WO** vs a held-out reference **HO**).

- **WO** — *working half original*: the part of the original data that is anonymized.
- **WA** — *working avatars*: the avatars generated from **WO**.
- **HO** — *holdout half original*: a second, untouched sample of real records, used only as a reference.

On datasets large enough (by default ≥ 500 rows) a **real holdout** is set aside before anonymization; on smaller ones an **approximate holdout** is drawn by randomly splitting the data when the metrics are computed. See the "Evaluation with a holdout reference" section in the Technical Assessment report.

© Octopize — All rights reserved



Comparison Strategy (Holdout)

To compute the final score returned to the user, we perform the following steps:

1. Obtain the working half **WO** and a disjoint holdout half **HO** from the real data (a real or approximate holdout — see the note above).
2. Take the working avatars **WA**, generated from **WO**.
3. Compute the **Closest Distances Ratio** on **WO** using **WA** as the attack base (both median and outliers).
4. Compute the **Closest Distances Ratio** on **WO** using **HO** as the attack base (both median and outliers).
5. Compute the ratio between the two resulting values.

This produces two outputs:

- The ratio of the **median Closest Distances Ratio** between avatars and the reference data.
- The ratio of the **outlier Closest Distances Ratio** between avatars and the reference data.

Interpreting the Closest Distances Ratio

The **Closest Distances Ratio** ranges from **0 to 1**:

- A value close to **0** indicates that the distance between an individual and their **closest neighbor** is **much smaller** than the distance to their second closest neighbor.
  This suggests a **strong ability** to form distinct individual–avatar pairs, which is unfavorable for **privacy protection**.
- A value close to **1** means that both distances are **similar**, preventing reliable identification of a unique association between an original individual and an avatar.
  This indicates **good avatar dissociation**, which is beneficial for privacy.

The **ratio value** returned to the user is interpreted as follows:

© Octopize — All rights reserved

- A value close to **100** or higher indicates that the local density of avatars relative to the original data is similar to that of the reference data, suggesting that avatars do **not** enable reliable linkage between original records and synthetic ones.
- A value significantly **below 100** means that avatars form more isolated pairs with original data than what naturally occurs in the dataset's variability — indicating an **increased re-identification risk**.

These two values can be interpreted similarly for both the median result and the outlier result.

i Example

Consider a dataset where the **Closest Distances Ratio** is **0.1**.
This means that for a given individual, the distance to their closest avatar is on average ten times smaller than the distance to their second closest avatar.
This creates a strong possibility of pairing the avatar with that individual, representing a **high re-identification risk**.

If the **Closest Distances Ratio** ratio is **50%**, this means that the local density formed between an original record and its closest avatar is twice as likely to produce distinct pairs compared to what is observed in the reference data — suggesting an **increased risk of re-identification**.

Metric Specificity
This metric is most meaningful when **combined** with the Distance To Closest metric.
A dataset is considered at risk when **both metrics are low**:

- A low **Distance To Closest** indicates that avatars are very close to original data.
- A low **Closest Distances Ratio** means that individual–avatar pairs are **easily identifiable**, which **increases re-identification risk**.

Formula
For each working original $x_i \in WO$, with $d_{1,i}$ and $d_{2,i}$ the distances to its closest and second closest record in the attack base:

$$r_i = \frac{d_{1,i}}{d_{2,i}} \quad (r_i = 1 \ \text{ if } \ d_{2,i} = 0)$$

The ratio is computed once with the working avatars as attack base (r_i^{WA}) and once with the holdout (r_i^{HO}):

$$\text{Closest Distances Ratio} = 100 \times \frac{\text{median}_i\, r_i^{WA}}{\text{median}_i\, r_i^{HO}} \qquad \text{outlier} = 100 \times \frac{P_{2.5}\left(r^{WA}\right)}{P_{2.5}\left(r^{HO}\right)}$$

$P_{2.5}$ is the 2.5th percentile (ratios closest to 0). When the reference value is 0, the score is set to 100. Distances are Euclidean, computed in the projected space of the anonymization (first ncp components).

© Octopize — All rights reserved

How to Improve This Metric?

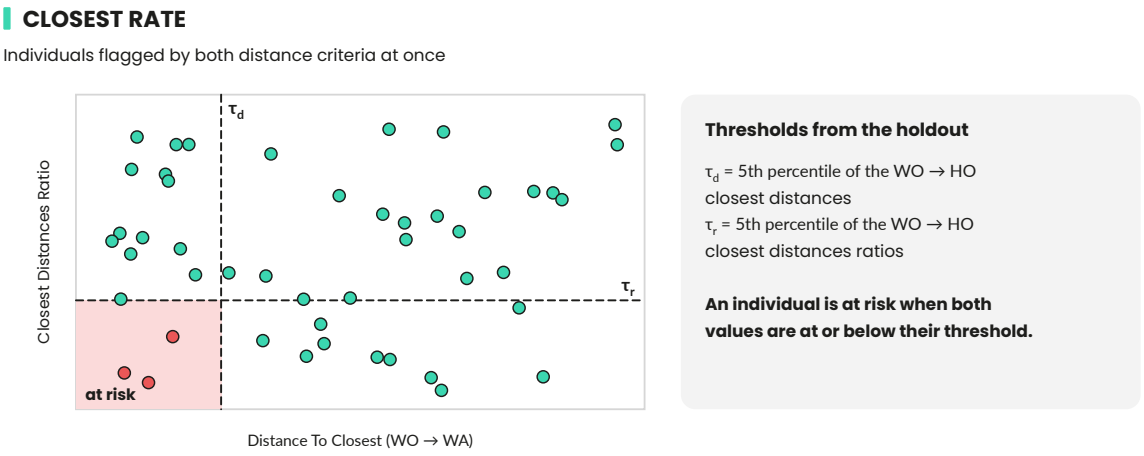
To improve the **Closest Distances Ratio**, it is recommended to **increase the value of *k*** — the number of neighbors used in synthetic data generation.

By increasing **k**, avatars are **more evenly distributed**, which increases the relative distance between an individual and their neighbors and **reduces the likelihood of creating duplicated or overly isolated pairs** in the dataset.

© Octopize — All rights reserved

Closest Rate

The **Closest Rate** metric combines the results of the **Distance to Closest** and **Closest Distances Ratio** metrics to determine the percentage of individuals whose most similar avatar is not too close, and whose original data/avatar pair is not easily distinguishable from the rest of the dataset. It provides an overall measure of the dissociation between the original data and their avatars in the synthetic dataset.

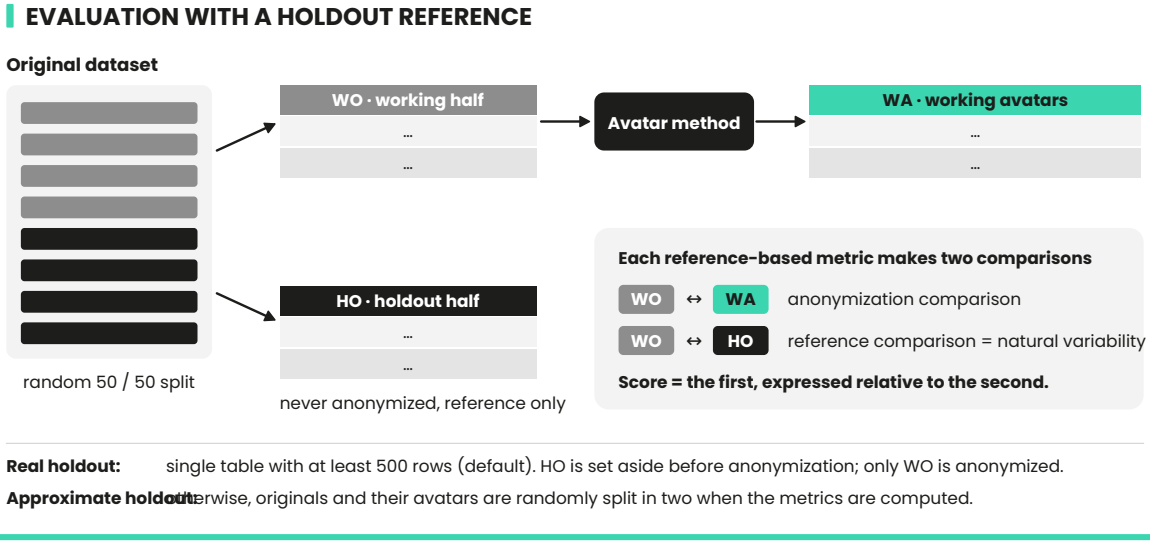


📝 Working half (WO), holdout half (HO), working avatars (WA)

Like the two metrics it combines, the **Closest Rate** is **reference-based**: it compares the anonymization (**WO**, the anonymized *working half*, vs its avatars **WA**) against the natural variability of the real data (**WO** vs a held-out reference **HO**, the untouched *holdout half*).

On datasets large enough (by default ≥ 500 rows) a **real holdout** is set aside before anonymization; on smaller ones an **approximate holdout** is drawn by randomly splitting the data when the metrics are computed. See the "Evaluation with a holdout reference" section in the Technical Assessment report.

© Octopize — All rights reserved



Interpreting Closest Rate

The **Closest Rate** measures how well avatars are separated from the original data while ensuring that the original data/avatar pair is not easily distinguishable from other avatars in the dataset. This composite measure reflects both the distance between the individual and their closest avatar, as well as the distinction between the original data/avatar pair and the rest of the generated avatars.

i Example

Consider a dataset with a **Closest Rate** of 85%. This means that for 85% of individuals, their most similar avatar is not too close to other avatars, and their original data/avatar pair is not easily distinguishable from other data points. In contrast, a low **Closest Rate** could indicate that a significant portion of individuals has avatars that are very similar or easily identifiable, which would increase the risk of re-identification.

Metric Specificity

The **Closest Rate** is particularly useful for providing a global measure of the anonymization quality in synthetic data generation. By combining the results of **Distance to Closest** and **Closest Distances Ratio**, this metric offers an overall view of the separation between the original data and their avatars, helping to assess the risk of re-identification.

Formula

Using the values of Distance To Closest (d_i^{WA}, d_i^{HO}) and Closest Distances Ratio (r_i^{WA}, r_i^{HO}), two thresholds are taken from the holdout reference, with $q = 5$ by default:

$$\tau_d = Q_{q\%}\left(d^{HO}\right) \qquad \tau_r = Q_{q\%}\left(r^{HO}\right)$$

An individual is at risk when both of its values are at or below these thresholds:

© Octopize — All rights reserved

$$R = \{\, i : d_i^{WA} \leq \tau_d \text{ and } r_i^{WA} \leq \tau_r \,\} \qquad \text{Closest Rate} = 100 \times \left(1 - \frac{|\,R\,|}{n}\right)$$

$Q_{q\%}$ is the q-th percentile (midpoint interpolation) and n the number of individuals for which a ratio is available.

How to Improve This Metric?

To improve the **Closest Rate**, it is recommended to increase the value of **k**, which represents the number of neighbors considered when generating the synthetic data. A higher **k** increases the separation between avatars and the original data, reducing the risk that avatars are too close to the actual data or easily confused with other avatars in the dataset.

© Octopize — All rights reserved

Column Direct Match Protection

The **Column Direct Match Protection** metric evaluates a simpler attack scenario, where the goal is to check if a single column in the dataset allows for direct linking between an individual and their avatar. It measures the ability of each column to protect against re-identification by assessing if the most re-identifying column can isolate individuals and their avatars.

The metric is accompanied by a list of the most re-identifying columns, ranked in descending order of re-identification risk.

COLUMN DIRECT MATCH PROTECTION

Can a single column be used as an identifier?

Age

original	avatar
43	42
25	25
38	35
52	56
79	81

matches m = 1 · unique values u = 5

80%

Blood type

original	avatar
A	A
O	O
O	A
B	O
A	A

matches m = 3 · unique values u = 2

76%

Record no.

original	avatar
101	101
102	102
103	103
104	104
105	105

matches m = 5 · unique values u = 5

0%

Per column: 100 × (1 - m × u / n²), n = 5 rows. The metric reports the lowest column (here Record no.: 0%).

A forgotten identifier makes every row match, so the column is flagged.

Interpreting Column Direct Match Protection

The **Column Direct Match Protection** returns the percentage of individuals whose data cannot be directly re-identified from the most re-identifying column in the dataset. A high percentage indicates that the most re-identifying column does not allow for re-identifying individuals, while a value close to **0** suggests that one or more columns contain information specific enough to re-identify individuals, such as a forgotten unique identifier.

i Example

Imagine a dataset where the **Column Direct Match Protection** is 10%. This means that 10% of the individuals in this dataset cannot be directly re-identified from the most re-identifying column. A score this low could indicate that a personal identifier was omitted during the anonymization process, allowing some individuals to be re-identified.

Metric Specificity

This metric is particularly useful when deciding to fix certain values during the anonymization process. It helps measure the impact of this decision on **privacy** and assess if certain variables might compromise anonymization. It is crucial to verify that columns do not contain quasi-identifying values or direct identifiers that could be used to re-identify individuals.

© Octopize — All rights reserved

Formula

For each column c, with x_{ic} the original value and y_{ic} the avatar value at the same row position:

$$m_c = \sum_{i=1}^{n} \mathbb{1}[x_{ic} = y_{ic}] \qquad u_c = \#\,\text{distinct avatar values of } \; c$$

$$P_c = 100 \times \left(1 - \frac{m_c\, u_c}{n^2}\right) \qquad \text{Column Direct Match Protection} = \min_c P_c$$

Weighting by u_c means that matches only weigh heavily for columns with many distinct values, such as identifiers. The columns whose P_c falls below the target are listed with the result.

How to Improve This Metric?

To improve the **Column Direct Match Protection**, it is recommended to ensure that no identifiers or quasi-identifying values remain in the dataset being anonymized. It is also important to include all variables that could be used for direct identification of individuals in the anonymization process. By eliminating or modifying these variables, you strengthen the protection against re-identification.

© Octopize — All rights reserved

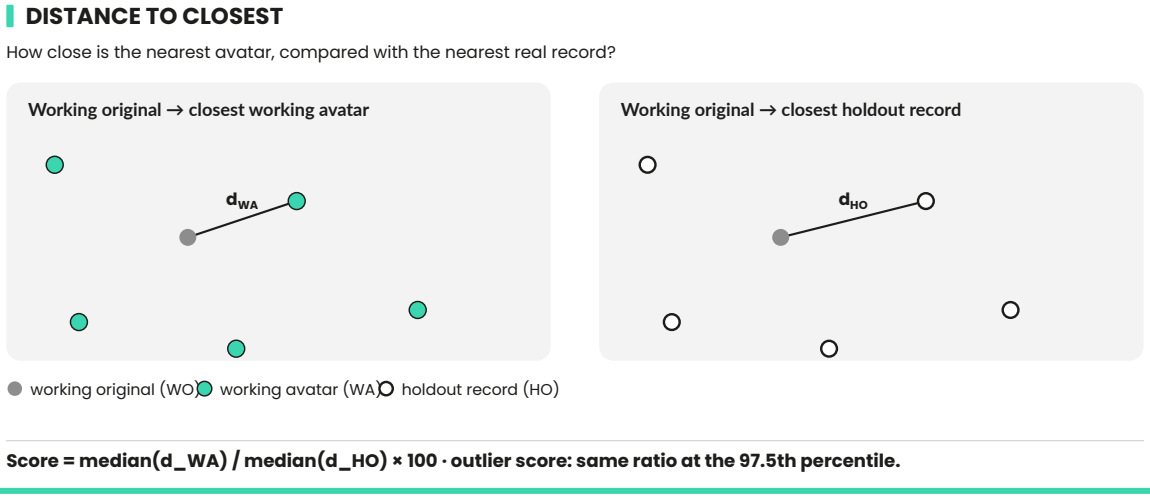
Distance To Closest

The **Distance To Closest** metric computes the **median Euclidean distance** between an individual and the closest avatar in the dataset, without taking into account the link between the original data point and its corresponding avatar.

This metric provides insight into how close the synthetic data lies to the distribution of the original data, while avoiding direct one-to-one connections between original individuals and their avatars.

For interpretability, the obtained value is compared to a reference value.
The final score is the ratio between the **Distance To Closest** computed on the avatars and the one computed on a reference dataset.
This comparison makes it possible to evaluate whether the proximity of avatars to original individuals is similar to what is observed in the reference data.

The same approach is applied to **outliers**, by computing the **Distance To Closest** for individuals with large distances to their closest neighbor (isolated points).



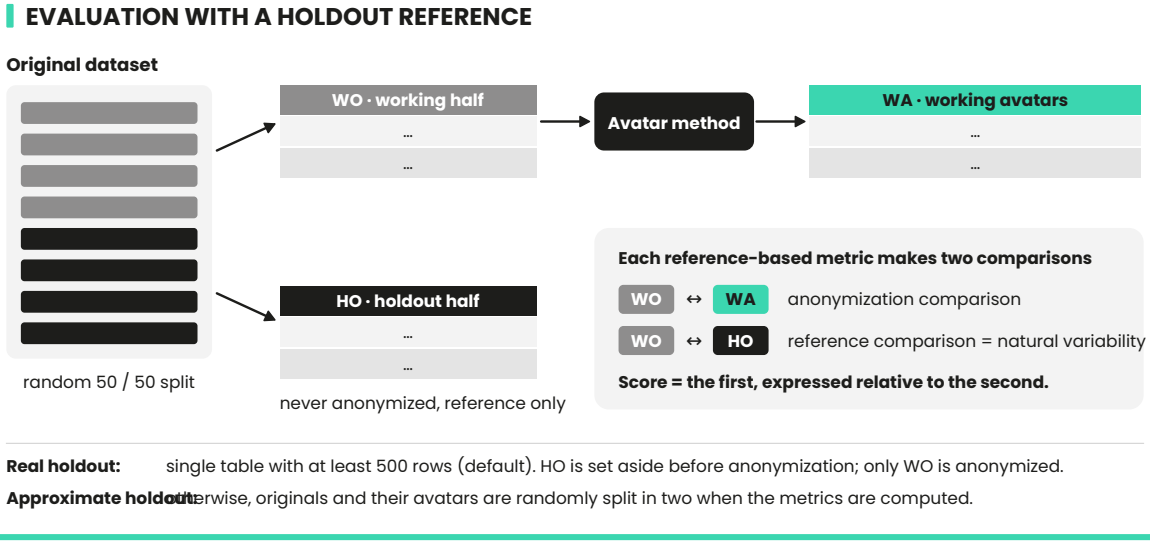
📝 Working half (WO), holdout half (HO), working avatars (WA)

This is a **reference-based** metric: it compares the anonymization (**WO** vs its avatars **WA**) against the natural variability of the real data (**WO** vs a held-out reference **HO**).

- **WO** — *working half original*: the part of the original data that is anonymized.
- **WA** — *working avatars*: the avatars generated from **WO**.
- **HO** — *holdout half original*: a second, untouched sample of real records, used only as a reference.

On datasets large enough (by default ≥ 500 rows) a **real holdout** is set aside before anonymization; on smaller ones an **approximate holdout** is drawn by randomly splitting the data when the metrics are computed. See the "Evaluation with a holdout reference" section in the Technical Assessment report.

© Octopize — All rights reserved



Comparison Strategy (Holdout)

To compute the final score returned to the user, the following steps are performed:

1. Obtain the working half **WO** and a disjoint holdout half **HO** from the real data (a real or approximate holdout — see the note above).
2. Take the working avatars **WA**, generated from **WO**.
3. Compute the **Distance To Closest** on **WO** using **WA** as the attack base (both median and outliers).
4. Compute the **Distance To Closest** on **WO** using **HO** as the attack base (both median and outliers).
5. Compute the ratio between the two resulting values.

The output consists of:

- The ratio of the **median Distance To Closest** between the avatars and the reference data.
- The ratio of the **outlier Distance To Closest** between the avatars and the reference data.

Interpreting Distance To Closest

The **Distance To Closest** metric ranges from 0 to infinity:

- A **low value** indicates that avatars are very close to the original data, which may suggest an increased re-identification risk.
- A **high value** means that avatars are far from the original data, which is favorable for privacy protection.

The ratio returned to the user is interpreted as follows:

- A value close to **100** or higher indicates that avatar proximity is similar to what is naturally observed in the reference data. This suggests that the avatar generation process does **not** enable reliable associations between original data points and their avatars.

© Octopize — All rights reserved

- A value significantly **below 100** indicates that avatars are substantially closer to the original data than expected from the dataset's natural variability, which suggests an **increased re-identification risk**.

These interpretations apply to both the median and the outlier values.

> ℹ **Example**
>
> Suppose the median **Distance To Closest** between the original data and the avatars is **1**, while the median distance between the same original data and the reference dataset is **2**.
> The ratio returned to the user is therefore **50%**.
>
> In other words, avatars are on average **twice as close** to the original data as what is observed in the dataset's natural variability.
> This represents an **increased re-identification risk**.

Metric Specificity

This metric is most meaningful when **used jointly** with the Closest Distances Ratio metric.
A dataset is considered at risk when **both values are low**:

- A low **Distance To Closest** indicates that avatars are very close to the original individuals.
- A low **Closest Distances Ratio** means that individual–avatar pairs are **easily identifiable**, which **increases re-identification risk**.

Formula

For each working original $x_i \in WO$, the distance to its closest working avatar and to its closest holdout record:

$$d_i^{WA} = \min_{y \in WA} d(x_i, y) \qquad d_i^{HO} = \min_{h \in HO} d(x_i, h)$$

$$\text{Distance To Closest} = 100 \times \frac{\text{median}_i\, d_i^{WA}}{\text{median}_i\, d_i^{HO}} \qquad \text{outlier} = 100 \times \frac{P_{97.5}\left(\, d^{WA}\,\right)}{P_{97.5}\left(\, d^{HO}\,\right)}$$

$P_{97.5}$ is the 97.5th percentile (the most isolated individuals). When the reference value is 0, the score is set to 100. Distances are Euclidean, computed in the projected space of the anonymization (first ncp components).

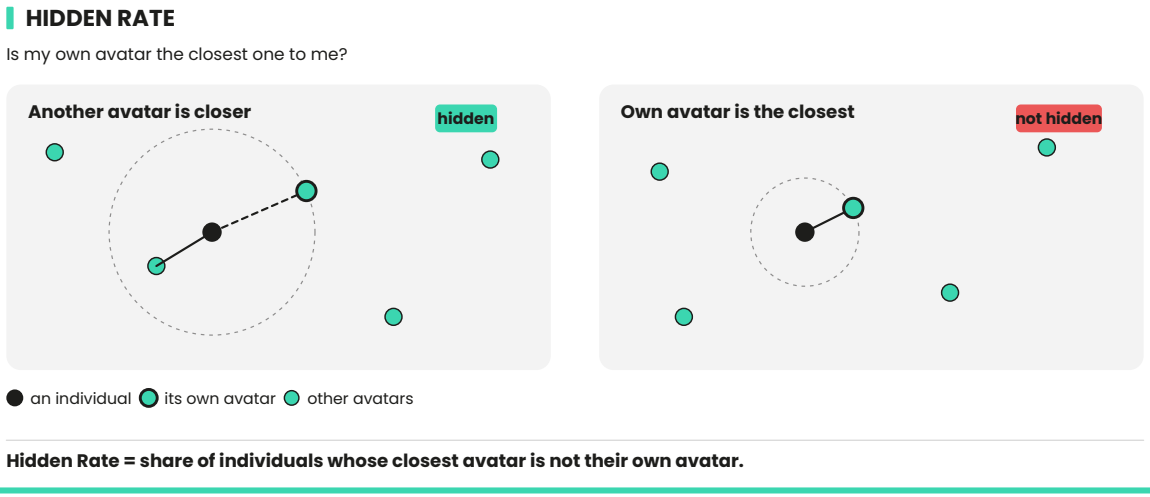
How to Improve This Metric?

The most effective way to improve the **Distance To Closest** metric is to **increase the *k* parameter**, which defines the number of neighbors considered during synthetic data generation.
A higher value of **k** produces a more homogeneous distribution of avatars across the dataset, increasing the distance between avatars and original individuals and reducing the likelihood of direct linkage.

© Octopize — All rights reserved

Hidden Rate

The **Hidden Rate** metric is a measure specific to the Avatar method. It is designed to address a *distance-based record linkage* attack scenario. This metric evaluates the loss of the link between an original data point and its avatar in the context of a method where each individual generates their own synthetic data. The **Hidden Rate** can also be seen as a safeguard that ensures there has been no algorithmic flaw and that the link between the original data and the synthetic data has been effectively broken.



Interpreting Hidden Rate

The **Hidden Rate** can be interpreted as follows: it represents the percentage of individuals for whom the generated avatar is not the most similar entry in the dataset.

> **i Example**
>
> If a dataset has a **Hidden Rate** of 80%, this means that for 80% of individuals, the generated avatar is not the closest match in the dataset. However, this does not imply that the remaining 20% are at risk, as the individuals concerned are chosen randomly and remain unknown to an attacker.

Metric specificity

This metric is crucial for quantifying the effectiveness of the Avatar method in dissociating original data points from the generated avatars. By increasing the **Hidden Rate**, it becomes more challenging for an attacker to link an individual to their avatar with certainty, thereby strengthening the protection of personal data.

Formula

For each original record x_i $(i = 1, ..., n)$ with its own avatar $y_{a(i)}$, let c_i be the number of other avatars that are at least as close as its own avatar:

$$c_i = \#\left\{ j \neq a(i) : d(x_i, y_j) \leq d(x_i, y_{a(i)}) \right\}$$

© Octopize — All rights reserved

$$\text{Hidden Rate} = 100 \times \left(1 - \frac{\#\{\, i : c_i = 0 \,\}}{n}\right)$$

An individual counts against the metric only when its own avatar is strictly its single closest avatar. Distances are Euclidean, computed in the projected space of the anonymization (first ncp components).

How to Improve This Metric?

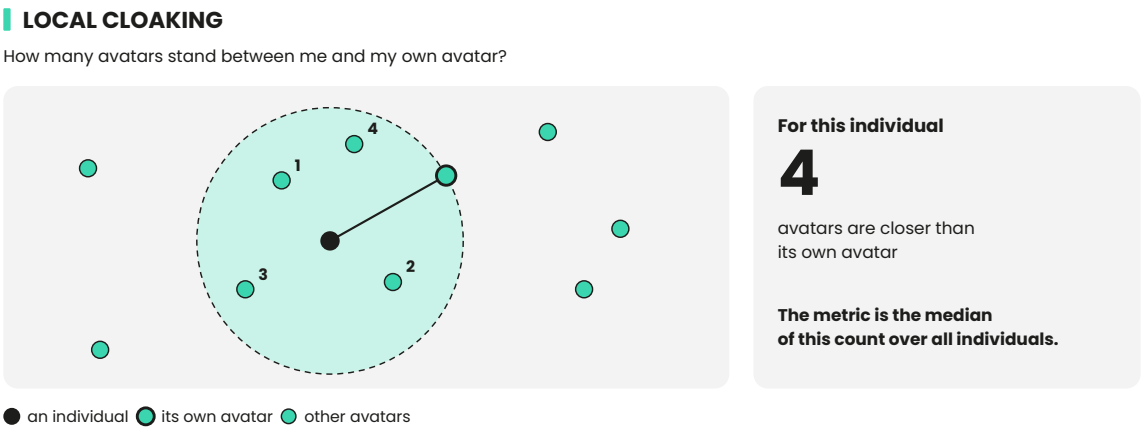
Improving the **Hidden Rate** metric can be achieved by increasing the **k** parameter, which represents the number of neighbors considered during the generation of synthetic data. A higher **k** value increases the relative distance between an original data point and its avatar, thereby strengthening dissociation and protection against *record linkage* attacks.

© Octopize — All rights reserved

Local cloaking

The **Local Cloaking** metric is specific to the Avatar method. It could be seen as a density metric. It is designed to address a *distance-based record linkage* attack scenario. This metric evaluates the loss of the link between an original data point and its avatar in a context where each individual generates their own synthetic data.

It quantifies how isolated an individual's avatar is from their original data point by measuring the median number of other avatars positioned between the original data point and its avatar in the distance space.



Interpreting Local Cloaking

Local Cloaking can be interpreted as follows: it returns the median number of avatars located between an original data point and its avatar.

> **i Example**
>
> Suppose a dataset has a **Local Cloaking** value of 5. This means that, between an original data point and its avatar, there is a median of 5 other avatars that are closer to the original data point. This result reflects the dissociation between the original data and its avatar, making it more difficult for an attacker to establish a direct link.

Metric Specificity

This metric is particularly favorable for high-density datasets, meaning datasets where many individuals share similar characteristics. In such cases, the number of nearby avatars increases, further enhancing the dissociation between the original data points and their avatars.

Formula

With c_i the number of avatars other than its own that are at least as close to x_i as its own avatar $y_{a(i)}$:

$$c_i = \#\left\{ j \neq a(i) : d(x_i, y_j) \leq d(x_i, y_{a(i)}) \right\} \qquad \text{Local Cloaking} = \text{median}_i c_i$$

© Octopize — All rights reserved

Distances are Euclidean, computed in the projected space of the anonymization (first ncp components).

How to Improve This Metric?

Improving the **Local Cloaking** metric can be achieved by increasing the **k** parameter, which represents the number of neighbors considered during the generation of synthetic data. A higher **k** value increases the relative distance between an original data point and its avatar, thereby strengthening dissociation and protection against *record linkage* attacks.

© Octopize — All rights reserved

Row Direct Match Protection

The **Row Direct Match Protection** metric evaluates a simple attack scenario by checking the number of **unique** rows (records) that are identical between the original data and the synthetic data. The goal is to measure how distinct the synthetic data can be from the original data, ensuring that the original records cannot be directly re-identified. We focus here on unique records, as duplicates can be considered protected with respect to the singling out criterion.

ROW DIRECT MATCH PROTECTION

Does any unique original row appear unchanged among the avatars?

ORIGINAL

Age	Height	Weight	Blood type
43	183	78	A
25	176	64	O
38	165	59	B
52	174	81	A
61	162	70	O

identical row

AVATARS

Age	Height	Weight	Blood type
42	179	75	A
38	165	59	B
27	174	66	O
56	176	80	A
64	165	72	O

Here 1 of 5 original rows is found as is: protection = 100 × (1 - 1/5) = 80%. Duplicated original rows are not counted.

Interpreting Row direct match protection

The **Row Direct Match Protection** is expressed as a percentage and returns the proportion of records from the original data that are not exactly present in the synthetic data. A **100%** score means that none of the original data records are present as is in the synthetic data, which is a positive indicator of privacy protection. On the other hand, low values indicate that some records from the original data are reproduced in the synthetic data, thus increasing the risk of re-identification.

i Example

Imagine a dataset where the **Row Direct Match Protection** is 95%. This means that 95% of the records from the original data do not exactly appear in the synthetic data. In other words, only 5% of the records are exactly reproduced, which represents a reduced risk of re-identification. A low score (close to 0) would indicate that many rows from the original data are present in the synthetic data, which would compromise confidentiality.

Metric Specificity

This metric is particularly useful for verifying the reliability of the local generation model used by the anonymization method. The better the model, the fewer direct matches there should be between the original and synthetic data. Additionally, higher values for this metric are easier to achieve for datasets with a large number of columns, as the probability of replicating an entire row is reduced.

© Octopize — All rights reserved

Formula

Let U be the original rows that occur exactly once in the original data (duplicated rows protect each other and are left out), and A the set of distinct avatar rows:

$$\text{Row Direct Match Protection} = 100 \times \left(1 - \frac{\#\{u \in U : u \in A\}}{n}\right)$$

A match requires equality on every column. n is the total number of original rows.

How to Improve This Metric?

Improvement of the **Row Direct Match Protection** can be achieved by increasing the **k** parameter, which represents the number of neighbors considered during the generation of synthetic data. A higher **k** parameter helps increase the local modeling space for each synthetic data point, thereby increasing the likelihood of generating synthetic data that is different from existing data.

© Octopize — All rights reserved

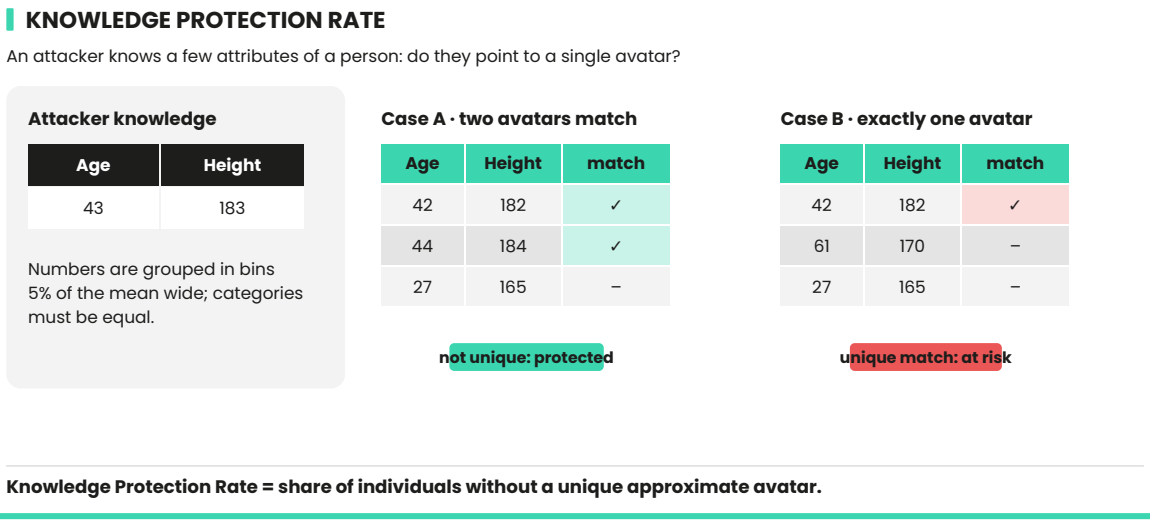
Knowledge Protection Rate

The **Knowledge Protection Rate** metric evaluates more probable and common attack scenarios, in which an attacker has **partial knowledge** of the original dataset.
It measures the system's ability to protect the specific information of an individual against **distance-based linkage attacks**.

The metric quantifies the percentage of individuals for whom the attacker cannot successfully link an original record to a **unique approximate avatar**, based on the partial information they possess.
Uniqueness is essential: if multiple avatars match the attacker's knowledge, the attacker cannot establish a certain link, which protects the individual with respect to the *singling out* criterion.

A tolerance threshold is applied to determine whether re-identification should be considered successful.
An avatar is considered **approximating** if:

- it shares the **same categorical values**, and
- its **numerical values fall within a 5% tolerance interval**.



Interpreting the Knowledge Protection Rate

The **Knowledge Protection Rate** returns the percentage of original individuals who **do not** have a unique approximate avatar in the synthetic dataset, based on a subset of variables likely to be known to the attacker (e.g., demographic information).

ℹ Example

Suppose a dataset yields a **Knowledge Protection Rate** of **92%**.
This means that for 92% of original individuals, the attacker cannot identify a unique approximate avatar in the synthetic data using the demographic information they possess.
Such a result indicates **strong protection** against distance-based linkage attacks.

© Octopize — All rights reserved

Metric Specificity

This metric is particularly relevant in scenarios where the attacker has partial prior knowledge of the original data — a situation that frequently occurs in real-world contexts.
By focusing on demographic information, it evaluates the robustness of avatars against **more targeted and realistic** attack strategies.

Formula

The metric uses only the variables known to the attacker. Each numeric variable v is cut into bins of width

$$\text{step}_v = 0.05 \times \bar{x}_v \quad \text{over} \quad \left[\min x_v - \frac{\sigma_v}{10}, \max x_v + \frac{\sigma_v}{10}\right]$$

(computed on the original records that occur once), and categorical values are kept as they are. Let $b(\cdot)$ be a record after binning, U the original records that occur exactly once, and B_1 the binned avatar profiles that occur exactly once:

$$S = \{\, x \in U : b(x) \in B_1 \,\} \qquad \text{Knowledge Protection Rate} = 100 \times \left(1 - \frac{|\,S\,|}{n}\right)$$

where n is the total number of original records.

How to Improve This Metric?

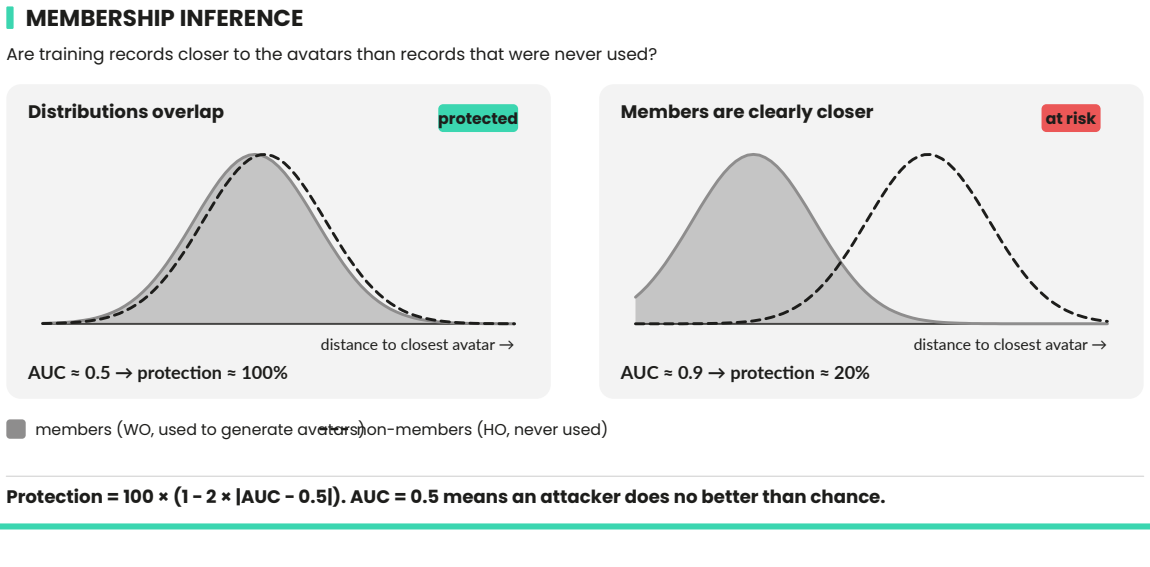
To improve the **Knowledge Protection Rate**, it is recommended to **increase the *k* parameter**, which defines the number of neighbors considered during synthetic data generation.
A higher **k** value increases the relative distance between original data points and their corresponding avatars, thereby reinforcing dissociation and enhancing protection against distance-based linkage attacks.

© Octopize — All rights reserved

Membership Inference Attack (MIA)

The **Membership Inference Attack (MIA)** metric evaluates whether an attacker can determine if a specific individual was part of the training data used to generate avatars. It measures how close that individual is to the nearest avatar compared to individuals who were **not** part of the training data.

Unlike the Hidden Rate, which relies on a binary indicator (whether the closest avatar is the individual's own), the MIA metric uses the full distribution of distances to detect subtle membership leakage. Two datasets can have the same Hidden Rate yet exhibit very different levels of real membership leakage.



Membership inference in synthetic data

This is not merely a question of whether a synthetic record matches a real record one-to-one. The core issue is whether an attacker can determine, with a strong statistical signal, that a specific person was present in the original training set or dataset used to generate the synthetic data.

The technical mechanism is usually linked to overfitting or memorization of atypical individuals. When a model or a synthetic data generator memorizes a rare profile too faithfully, the generated data or model outputs preserve a recognizable statistical trace of that person. In other words, the attack does not require an exact copy; a distinctive combination of variables may be enough to make the person stand out.

Example: medical research dataset

- Context: a hospital publishes a synthetic dataset derived from a real study on an experimental treatment for HIV. The goal is to enable researchers to perform statistical analysis without accessing patient-level records.
- Attacker knowledge: a neighbor knows that Marc attended the hospital regularly in 2025 and has access to a few public facts (male, 48, postal code 35000, blood group O+).
- Attack: the attacker inspects the synthetic dataset and notices an abnormal concentration of highly specific profiles combining exactly: male, 48, postal code 35000, blood group O+, with highly precise biomarkers.

© Octopize — All rights reserved

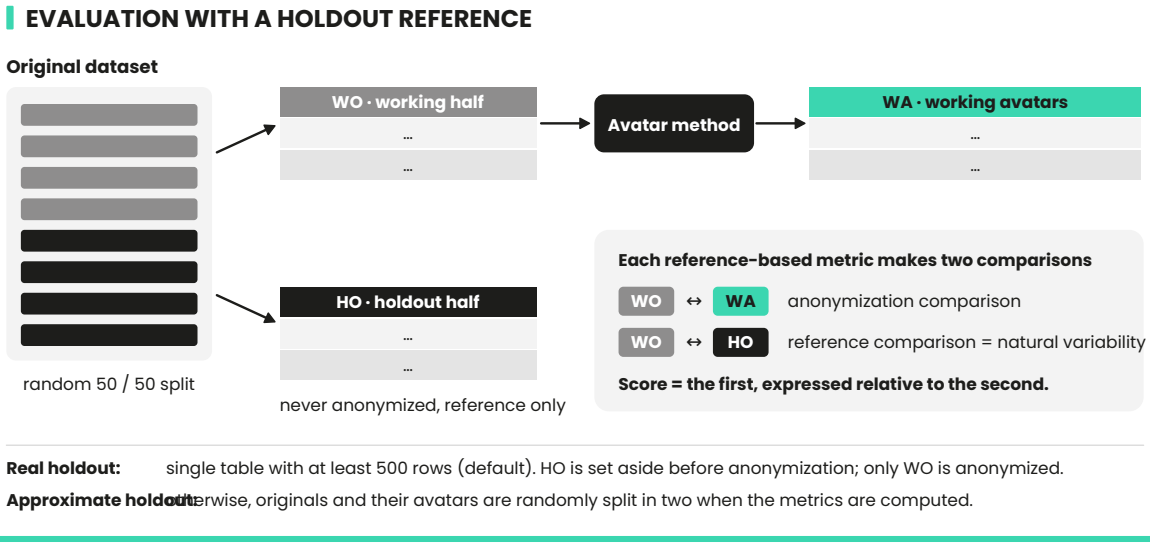
- Result: the attacker can infer with strong statistical confidence that Marc was part of the original cohort. This reveals an extremely sensitive health attribute: Marc has HIV.

This is precisely the risk captured by the membership inference criterion: the synthetic data has preserved enough of the original signal to reveal whether a concrete individual was a member of the source dataset.

How It Works

Two populations are compared:

- **Members** — original records that were used to generate avatars (working originals).
- **Non-members** — hold-out records that were **not** used during avatar generation.



For each record, the **Euclidean distance to its nearest avatar** is computed. If members are systematically closer to avatars than non-members, an attacker can exploit that distance gap to infer membership.

The strength of this signal is quantified as an AUC-ROC (via the Mann-Whitney U statistic), which is then converted to a protection rate:

$$\text{protection_rate} = (1 - 2 \times |\text{AUC} - 0.5|) \times 100$$

Here, AUC is expressed on a 0–1 scale (e.g., AUC = 0.5 corresponds to 50%).

- **AUC = 0.5 (50%)** → protection rate = **100%** (random guessing — perfect privacy)
- **AUC = 0 or 1 (0% or 100%)** → protection rate = **0%** (perfect attack — no privacy)
  ## Raw vs. Normalized Variants

Two variants of the metric are computed:

- **Raw MIA**: uses raw Euclidean distances to the nearest avatar.
- **Normalized MIA**: divides each distance by a baseline distance to the opposite population (member distances divided by distance to nearest non-member, and vice versa).

© Octopize — All rights reserved

The normalized variant accounts for local density in the feature space. In sparse regions, even large distances may be unusually close, while in dense regions, short distances may be typical. Normalization effectively asks *"is this distance unusual for this region?"* rather than *"is this distance large in absolute terms?"*

The normalized protection rate is generally more robust and should be preferred for interpretation.

Interpreting Membership Inference Attack

The **MIA protection rate** ranges from 0 to 100:

- A protection rate close to **100%** means that the distance distributions of members and non-members are indistinguishable — an attacker cannot exploit proximity to avatars to infer membership. This is the ideal outcome.
- A protection rate close to **0%** means there is a clear distance gap between members and non-members, making membership inference trivially possible.

This metric relates to the **singling out** criterion: it measures whether an attacker can determine if a specific individual contributed to the training set.

Metric Specificity

Advantages:

- Detects membership leakage that simpler metrics (like Hidden Rate) may miss, by analyzing full distance distributions rather than relying on a binary indicator.
- Based on a well-established statistical framework (AUC-ROC via the Mann-Whitney U statistic).
- The normalized variant accounts for local density variations in the feature space, making the metric robust across heterogeneous data regions.

Limitations:

- Requires a hold-out set, which reduces the amount of data available for avatar generation.
- Results may vary depending on the size and representativeness of the hold-out set.
- In very small datasets, the statistical power of the underlying test may be limited.

Formula

Members are the working originals WO (size n_m), non-members the holdout HO (size n_h). Each record gets its distance to the closest working avatar:

$$d_i = \min_{y \in WA} d(x_i, y)$$

With R_h the sum of the ranks of the non-member distances in the pooled, ascending sample (ties get their average rank), the AUC is the Mann-Whitney statistic:

$$\mathrm{AUC} = \frac{R_h - \frac{n_h(n_h+1)}{2}}{n_m\, n_h} \qquad \text{protection rate} = 100 \times (1 - 2\,|\mathrm{AUC} - 0.5|)$$

© Octopize — All rights reserved

For the **normalized** variant, each distance is divided by the distance b_i to the closest record of the opposite population (a random subsample of size $\min(n_m, n_h)$): $s_i = d_i/(b_i + \varepsilon)$, and the same AUC is computed on s_i. Distances are Euclidean, computed in the projected space of the anonymization (first ncp components).

How to Improve This Metric?

To improve the **MIA protection rate**, you can increase the **k** parameter, which controls the number of neighbors considered during avatar generation. A higher **k** value increases the distance between members and their nearest avatars, making the distance distributions of members and non-members more similar.

You should also ensure that the hold-out set is sufficiently large and representative of the overall population, as a biased or too-small hold-out can distort the metric.

> ⚠️ **Hold-out requirement**
>
> The MIA metric requires a **hold-out set**, a subset of individuals that was **not** used during avatar generation. This hold-out set serves as the non-member population against which membership leakage is measured.

© Octopize — All rights reserved

Linkability

Linkability is a fundamental criterion in evaluating the anonymity of synthetic data. It refers to the risk that two or more records within a dataset—or across datasets—can be linked to the same individual based on shared attributes.

> ℹ **Example**
>
> Consider two datasets: one containing medical records and another containing shopping habits. An attacker who knows that an individual purchased a specific health-related product may link that information with medical conditions listed in the other dataset if overlapping attributes, such as age, gender, or location, are shared between the datasets. This connection can lead to the identification of sensitive personal information.

List of Metrics Measuring Linkability Criteria

- Linkability Protection Rate

Factors Influencing the Risk of Linkability

The risk of linkability increases when datasets share a significant number of overlapping attributes, especially if these attributes are specific or unique. Attributes like exact birth dates, rare job titles, or detailed geographic information can create strong links between records.

Additionally, the use of highly structured datasets with limited variability across records can amplify the risk. For example, a dataset containing many individuals with identical demographic traits can create patterns that are easier to connect between different datasets.

This criterion is particularly relevant for individuals whose records appear in multiple datasets. Linking multiple data sources could reveal a more complete profile of the individual, increasing the risk of re-identification.

Importance of Mitigation

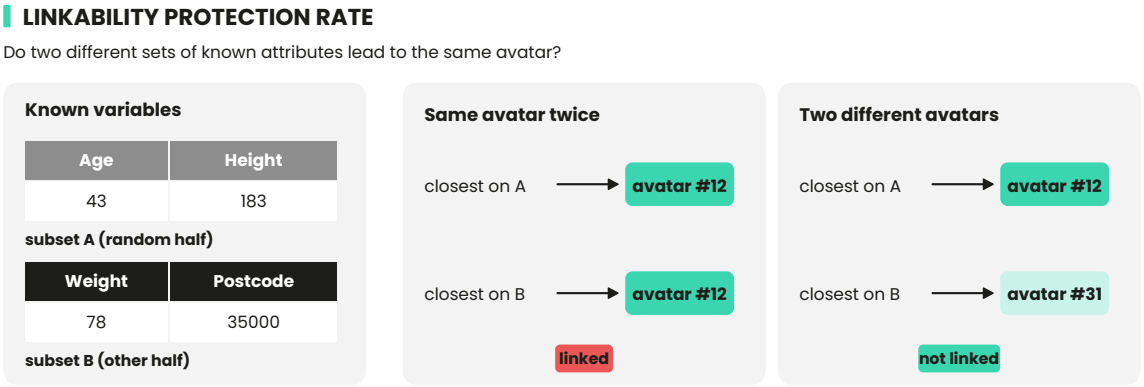
To limit the risks of linkability, the metrics we have developed systematically assess the probability that records can be connected across datasets based on shared attributes. These analyses help identify overlaps and reduce the likelihood of linkages by introducing controlled variability or obfuscation in synthetic data. By addressing these vulnerabilities proactively, the Avatar method ensures greater protection of individual privacy.

© Octopize — All rights reserved

Linkability Protection Rate

The **Linkability Protection Rate** metric is specifically designed to evaluate the **linkability criterion** defined by the European Data Protection Board (EDPB).
It quantifies the risk of **correlation** in synthetic datasets by **simulating scenarios** in which different subsets of variables are used to establish matches between avatars and their corresponding original records.
This metric directly addresses the **risk of cross-linking**, providing a robust measure of privacy protection.

The simulated scenario consists of selecting **two distinct subsets of variables** that could realistically be available in an external database (such as demographic information).
Each original individual from both subsets is then matched to their closest avatar within the feature space defined by the selected variables.
If an individual is linked to the **same avatar** across both subsets, this reveals a **correlation risk** for that individual.



Interpreting the Linkability Protection Rate

The **Linkability Protection Rate** is expressed as a **percentage**, where **higher values indicate stronger protection**.
A score of **100%** means that **no avatar was linked to the same original record** across the different variable subsets — demonstrating complete dissociation between the synthetic and original data.
Conversely, **lower values** indicate a greater **linkability risk**, as some avatars maintain a **stable association** with their original record even when the linking variables change.

i Example

Consider a dataset where the **Linkability Protection Rate** is **85%**.
This means that **85% of avatars** could **not** be linked back to the same original record when tested using two different subsets of variables.
The remaining **15%**, however, sustained a consistent match with the same

© Octopize — All rights reserved

individual, revealing a **potential vulnerability** in the synthetic data generation process.

Metric Specificity

This metric is particularly useful for datasets containing **attributes likely to overlap** across different contexts or domains.
By simulating a **correlation attack scenario**, in which different variable subsets are used to link avatars to original records, it provides a **concrete measure of robustness** for synthetic data.
It evaluates how effectively the Avatar method prevents **exploitable correlations** that could compromise anonymity by facilitating linkages between multiple datasets.

Formula

The known variables K are shuffled and split into two halves K_A and K_B (K_A takes the extra variable when $|\ K\ |$ is odd). For each original record x_i, the closest avatar is searched on each half with the Gower distance, ties being broken at random:

$$j_A(i) = \arg\min_j G_{K_A}(x_i, y_j) \qquad j_B(i) = \arg\min_j G_{K_B}(x_i, y_j)$$

$$G_V(x,y) = \frac{1}{|\ V\ |}\sum_{v\in V}\delta_v(x,y), \qquad \delta_v = \begin{cases} \frac{|\ x_v - y_v\ |}{\max_v - \min_v} & \text{numeric} \quad v \\ \mathbb{1}[x_v \neq y_v] & \text{categorical} \quad v \end{cases}$$

$$\text{Linkability Protection Rate} = 100 \times \left(1 - \frac{\#\{\, i : j_A(i) = j_B(i)\,\}}{n}\right)$$

Numeric ranges are taken over originals and avatars together. The metric is not computed with fewer than two known variables.

How to Improve This Metric?

To improve the **Linkability Protection Rate**, it is recommended to **increase the value of the *k* parameter** — the **number of neighbors** considered during synthetic data generation.
A **higher k** increases the **relative distance** between an original record and its avatar, strengthening **dissociation** and enhancing resistance to *linkability* attacks, especially in contexts involving **multiple data sources**.

© Octopize — All rights reserved

Inference

The risk of **inference** refers to the phenomenon where seemingly anonymous or harmless information can be used to deduce sensitive attributes about an individual. Unlike individualization, where the goal is to uniquely identify an individual, inference involves drawing conclusions from a subset of data without needing to directly know the person's identity. A dataset may be free from individualization for a set of attributes, yet still be vulnerable to inference attacks.

> **i Example**
>
> Imagine you learn that your 25-year-old neighbor, who is 1.80 meters tall, recently participated in a clinical study. You access a database containing information about participants in this study, and find three individuals who match these characteristics. While there is no **individualization risk** – as you cannot know which of the three is your neighbor – you notice that all three participants are sick. You can then **infer** that your neighbor is probably sick because the data suggests a strong correlation between these characteristics and the illness.

List of metrics measuring singling out criteria

- Attribute Inference

Factors Influencing the Risk of Inference

The risk of inference is directly related to the density of the data and the precision of the information collected. Even if a dataset does not directly identify an individual, it can still allow for additional information about that individual to be inferred, enriching an attacker's knowledge base.

Importance of Mitigation

To limit the risks of inference, it is crucial to understand that even a dataset that appears anonymous at first glance can provide enough information to allow inferences about individuals. The metrics developed in our system evaluate not only the risk of individualization but also the risks of inference, helping to better protect sensitive data. Mitigating this risk requires considering interactions between different variables and adopting a more sophisticated approach to ensure that no sensitive conclusions can be drawn from the available data.

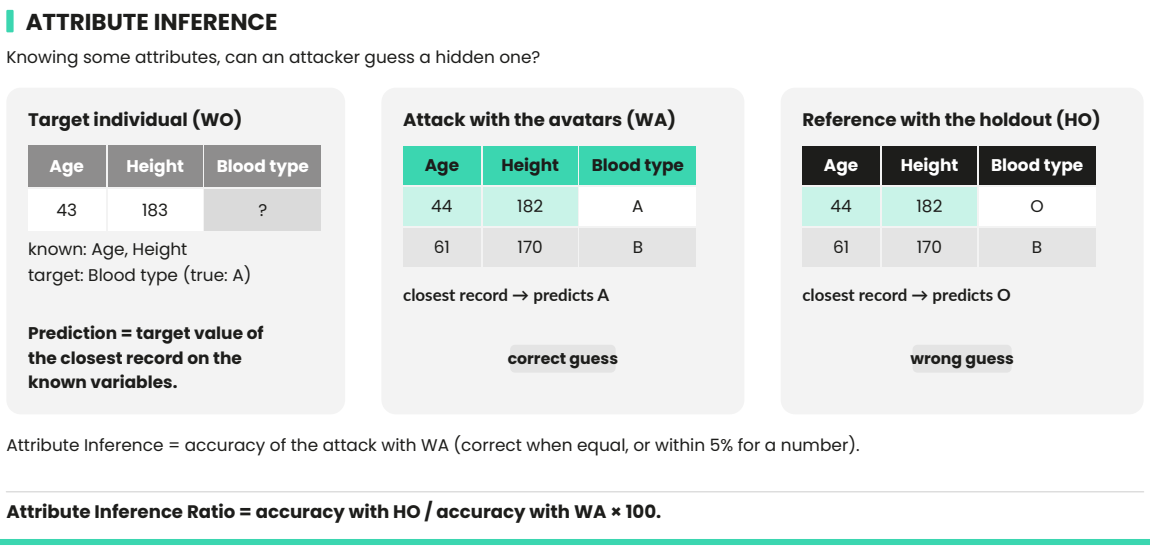
It's important to note that for inference to be considered present, the inferred information must be certain. For instance, determining that an individual known to be a smoker has a 90% chance of contracting lung cancer would not be considered inference.

© Octopize — All rights reserved

Attribute Inference

The **Attribute Inference** metric evaluates the risk that an attacker could **successfully deduce the value** of a target variable using a subset of known attributes.
An inference is considered successful when the predicted modality matches the true modality for a categorical variable, or when the predicted value is sufficiently close to the real value for a continuous variable (~5%).

In practice, each individual is assigned the target value of the synthetic record that is most similar in terms of attribute proximity. The resulting prediction performance constitutes the **Attribute Inference** score.
This metric is particularly relevant in attribute inference attack scenarios, where adversaries leverage publicly available or easily accessible demographic data to infer sensitive categorical information about individuals.



Inferred correlation and attribute inference in synthetic data

This risk is broader than membership inference. It concerns the ability to infer a sensitive attribute about a person even when there is no direct identifier, no direct row match, and no explicit linkage to a name or record in the source data. The attack leverages statistical correlations preserved by the synthetic data or by a model.

The essential point is that synthetic data can keep micro-correlations that are real in the original population. Even where there is no direct one-to-one mapping, a small set of known attributes can be used to identify a narrow subpopulation in which a hidden attribute is overwhelmingly likely.

Example A: inferred financial attribute from a synthetic dataset

- Context: a bank generates a synthetic dataset reflecting customer financial behavior for a risk-analysis hackathon.
- Attacker knowledge: the attacker knows that Valérie is 34, lives in Cesson-Sévigné, is an engineer, and has two young children.
- Attack: the attacker filters the synthetic dataset to the intersection: age 30–35, location Cesson-Sévigné, profession engineer, children = 2.

© Octopize — All rights reserved

- Result: within that very narrow group, 98% of the synthetic records show a high debt ratio or credit purchase pattern.
- Conclusion: the synthetic data preserves a real correlation specific to a small subgroup. The attacker can infer with near-certainty that Valérie is over-indebted, even though neither her name nor her identifiers were directly transmitted.

Example B: prompt-based inference on an AI model

- Context: a company trains an internal LLM on performance reviews and manager notes to assist HR in drafting job descriptions.
- Attack: an employee submits the prompt: "Draft a summary of the managerial feedback for a senior project manager in the Marketing team, who joined the company in 2022 and had a long sick leave in the spring."
- Result: the model generates text that reconstructs highly specific elements of the real evaluation of one individual, revealing a medical or personal situation that was never intended to be disclosed.
- Legal and technical interpretation: the prompt uses secondary attributes to isolate a highly specific individual. The model is effectively leaking a sensitive attribute via statistical completion.

This is the core challenge for synthetic data: when the synthetic output is too faithful to the underlying population, it preserves the micro-correlations that enable membership and attribute inference. When it is too smooth, it may lose utility for analysis and ML applications. A robust anonymization evaluation therefore needs to demonstrate that such inferences are no longer realistically feasible in context, not simply that there is no direct one-to-one mapping.

Comparison Strategy (Holdout)

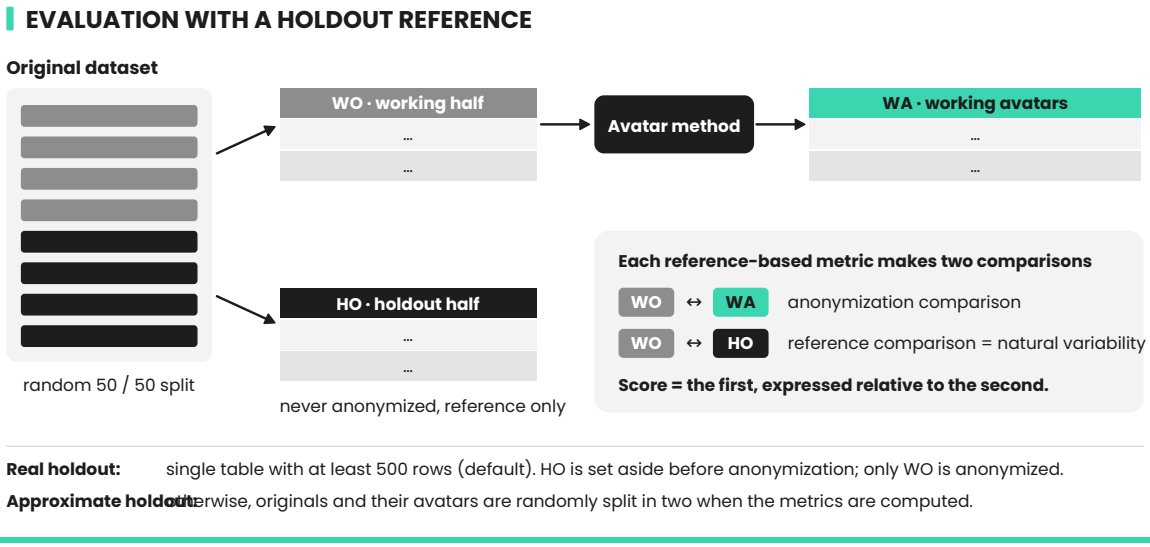
To improve interpretability, this metric is divided into two sub-metrics:

- **Attribute Inference**: evaluates the inference risk for a given target variable as described above.
- **Attribute Inference Ratio**: compares the inference performance obtained using avatars against that of a reference model.

> 📝 **Working half (WO), holdout half (HO), working avatars (WA)**
>
> The **Attribute Inference Ratio** is **reference-based**: it compares inference using the anonymization (**WO** vs its avatars **WA**) against inference using the natural variability of the real data (**WO** vs a held-out reference **HO**).
>
> - **WO** — *working half original*: the part of the original data that is anonymized.
> - **WA** — *working avatars*: the avatars generated from **WO**.
> - **HO** — *holdout half original*: a second, untouched sample of real records, used only as a reference.
>
> On datasets large enough (by default ≥ 500 rows) a **real holdout** is set aside before anonymization; on smaller ones an **approximate holdout** is drawn by randomly splitting the data when the metrics are computed. See the

© Octopize — All rights reserved

"Evaluation with a holdout reference" section in the Technical Assessment report.



The comparison is performed using the following steps:

1. Obtain the working half **WO** and a disjoint holdout half **HO** from the real data (a real or approximate holdout — see the note above).
2. Take the working avatars **WA**, generated from **WO**.
3. Compute the **Attribute Inference** metric on **WO** using **WA** as the attack base.
4. Compute the **Attribute Inference** metric on **WO** using **HO** as the attack base.
5. Compute the ratio between the two resulting performance scores.

Metric Interpretation

The **Attribute Inference** metric is expressed as the **accuracy of the attacker** (in %):

- A value of **0%** means that **no inference** by the attacker succeeded.
- Higher values indicate higher attacker accuracy and thus a **greater inference risk**.

Unlike the other privacy metrics, **lower values mean stronger privacy protection** for this metric.

The **Attribute Inference Ratio** is interpreted as follows:

- A value close to **100%** indicates that inference performance using avatars is similar to that obtained using real data. The anonymization process does **not** increase inference risk.
- A value below **100%** means that avatars **increase** the attacker's ability to correctly infer the target variable.

ℹ Example

If the **Attribute Inference** value is **20%**, attackers correctly predict the target for **20%** of individuals, which corresponds to a **low inference risk**.

© Octopize — All rights reserved

Conversely, a value of **90%** means attackers reach **90% accuracy**, indicating a **high inference risk** and the need for further privacy-enhancing adjustments.

If the **Attribute Inference Ratio** is **50%**, this means that using avatars **doubles** the attacker's capacity to correctly infer the target variable compared to using real data.

Metric Specificity

These metrics are essential to assess the **ability of the Avatar method to protect against attribute inference attacks**.
For proper interpretation, both **Attribute Inference** and **Attribute Inference Ratio** must be considered together.
For example, the **Attribute Inference Ratio** may indicate that avatars significantly increase attacker performance, yet if the **Attribute Inference** score remains low, the **overall inference risk** can still be low.

Formula

The known variables are projected (factor analysis fitted on WO, $\min(5, |\ K\ |)$ components). For each working original $x_i \in WO$, the prediction $\hat{t}_i$ is the target value of its nearest neighbour in the attack base (WA, or HO for the reference). A prediction is correct when

$$\mathrm{ok}_i = \begin{cases} \mathbb{1}\left[\hat{t}_i = t_i\right] & \text{categorical target} \\ \mathbb{1}\left[\,|\ \hat{t}_i - t_i\ | \le 0.05\,|\ t_i\ |\,\right] & \text{numeric target} \end{cases}$$

(for $t_i = 0$, the tolerance is 0.05 times the mean absolute non-zero target). Then

$$\mathrm{Acc}_{WA} = \frac{100}{n}\sum_i \mathrm{ok}_i^{WA} \qquad \mathrm{Acc}_{HO} = \frac{100}{n}\sum_i \mathrm{ok}_i^{HO}$$

$$\text{Attribute Inference} = \mathrm{Acc}_{WA} \qquad \text{Attribute Inference Ratio} = 100 \times \frac{\mathrm{Acc}_{HO}}{\mathrm{Acc}_{WA}}$$

The ratio is not defined when $\mathrm{Acc}_{WA} = 0$.

How to Improve This Metric?

To improve these metrics, begin by identifying the **demographic variables that are most predictive** of the target variable.
Next, analyze whether groups of individuals sharing key demographic attributes **tend to exhibit similar target values**.

If such clusters exist, the **k** parameter (the number of neighbors used during synthetic data generation) should be set to a value **greater than the size of these clusters**.
This increases **variability among individuals with similar target values**, reduces inference risk, and lowers the attacker accuracy reported by the metric—thereby **strengthening privacy protection**.

© Octopize — All rights reserved

Utility

The evaluation of synthetic data quality hinges on a delicate balance between privacy and utility. While privacy ensures that individuals cannot be identified or linked, utility focuses on the preservation of the original dataset's statistical properties, enabling synthetic data to be a reliable substitute for real data.

Evaluation of Utility

To assess the utility of synthetic data, we adopt a framework based on three generic levels of metrics. These levels offer a comprehensive evaluation of how well the synthetic data retains the essential characteristics of the original dataset:

- Univariate Utility: Measures the similarity of individual variable distributions between the original and synthetic datasets.
- Bivariate Utility: Evaluates the preservation of pairwise correlations between variables.
- Multivariate Utility: Assesses the retention of the dataset's global structure, including clustering patterns and overall data relationships.

More details about our privacy metrics here:

- Univariate
- Bivariate
- Multivariate

While these three levels provide a generic foundation for utility evaluation, we strongly encourage the incorporation of use-case-specific metrics tailored to the needs of the dataset and its intended applications. Custom metrics allow for a more targeted assessment of the synthetic data's utility by focusing on specific aspects or features crucial to the use case.

The metrics are designed with a minimization objective: lower values indicate better utility, meaning that the synthetic data closely mirrors the statistical properties of the original dataset.

The Importance of Context

Preserving utility is most effective when the desired use of the synthetic data is clearly defined. Adding custom metrics specific to the dataset or its application enhances the evaluation process. For example:

```
A healthcare dataset may prioritize metrics that assess the retention of diagnostic
trends.
A financial dataset may focus on the preservation of volatility or correlations
between financial indicators.
```

Utility vs. Privacy: The Balance

Utility must always be evaluated in tandem with privacy to ensure that the synthetic data fulfills its purpose without compromising anonymity. As with privacy, the context of use is critical in determining the adequacy of utility:

```
For exploratory data analysis, retaining global trends and distributions may suffice.
For training machine learning models, the preservation of multivariate structures and
complex interactions is often necessary.
```

© Octopize — All rights reserved

By systematically evaluating utility across these three levels and incorporating use-case-specific metrics, our tools empower users to ensure that synthetic data is both privacy-compliant and fit for its intended purpose. This dual focus ensures the creation of synthetic datasets that are secure, reliable, and practical for a wide range of applications.

© Octopize — All rights reserved

Univariate

Univariate utility measures how well the statistical properties of individual variables are preserved in the transformed dataset. This evaluation ensures that the fundamental characteristics of each variable—such as its distribution shape, presence of outliers, and skewness—are maintained after data transformation.

Univariate utility is crucial for preserving the integrity of analyses that rely on single-variable statistics, such as descriptive statistics, hypothesis testing, and univariate predictive models.

> **i Example**
>
> Consider a dataset containing individuals' ages. In the original dataset, the age distribution follows a **bimodal Gaussian shape**, reflecting two distinct subgroups: a younger group (e.g., students or early-career professionals) and an older group (e.g., retirees).
>
> When transforming the dataset, it is important to check whether the resulting age distribution retains its bimodal nature, as well as key statistical properties like the **mean and variance**. If the transformation alters the distribution significantly—e.g., by merging the two peaks into a single mode or shifting the mean—it could lead to misleading analyses and incorrect conclusions.

List of Metrics Measuring Univariate Utility

- Hellinger Distance
- Kolmogorov-Smirnov Test

Factors Influencing Univariate Utility

Several factors can impact the preservation of univariate statistical properties:

- **Outliers**: Extreme values can significantly influence statistical properties such as the mean and variance. Ensuring that outliers are not overly smoothed out while ensuring privacy or exaggerated in the transformed dataset is crucial.
- **Skewed Variables**: Variables with asymmetric distributions (e.g., income, transaction amounts) require careful handling to avoid distorting their statistical properties.
- **Low Data Availability**: Variables with a small number of observations may have unstable distributions, making it harder to preserve their statistical properties.
- **Missing Data**: A high percentage of missing values can lead to biases if the missingness pattern is not correctly accounted for during transformation.

Understanding these factors helps in selecting appropriate transformation techniques and ensuring the utility of the dataset for downstream analyses.

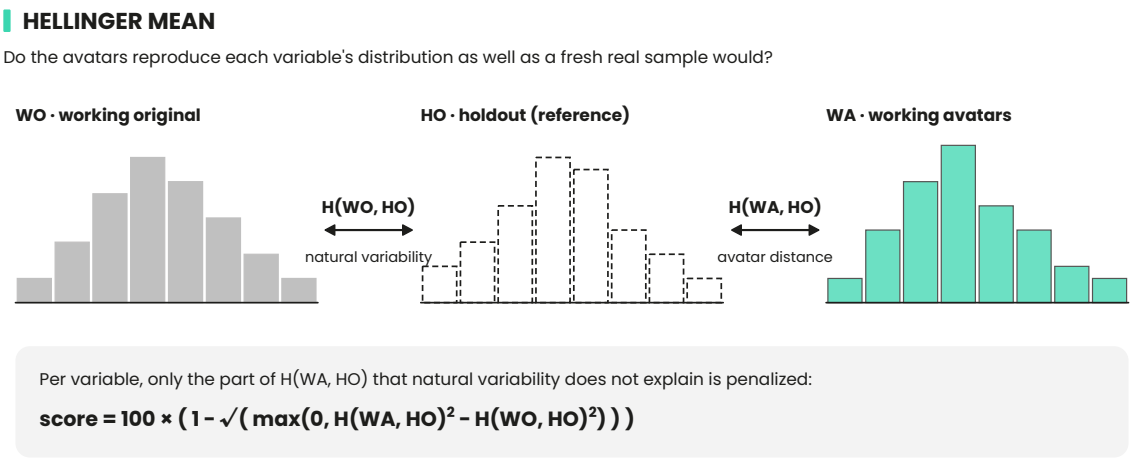
© Octopize — All rights reserved

Hellinger Mean

The **Hellinger Mean** compares how far the avatars' distributions are from a held-out half of the original dataset with how far the two halves of the original dataset are from each other. For interpretability, it uses the natural variability of the data as a reference.

For both comparisons, it computes the Hellinger distance variable by variable, where Hellinger distance measures the divergence between two probability distributions (0 = identical, 1 = completely different).

This metric applies to both categorical and continuous variables. For continuous variables, the Hellinger distance is estimated using histogram binning, while for categorical variables, it is directly computed from observed frequencies.



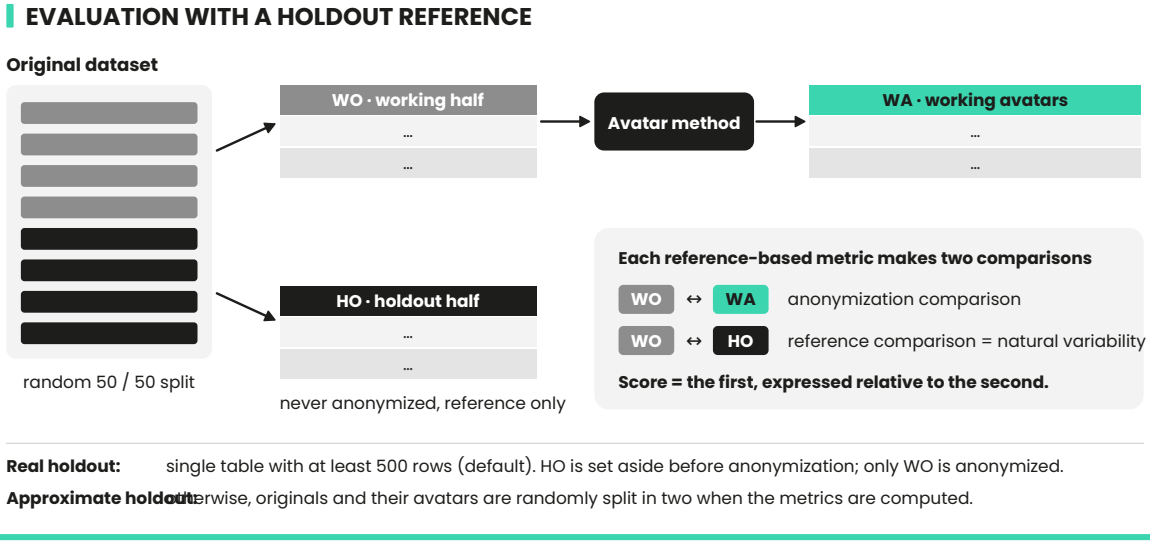
📝 Working half (WO), holdout half (HO), working avatars (WA)

This is a **reference-based** metric: it compares the anonymization (**WO** vs its avatars **WA**) against the natural variability of the real data (**WO** vs a held-out reference **HO**).

- **WO** — *working half original*: the part of the original data that is anonymized.
- **WA** — *working avatars*: the avatars generated from **WO**.
- **HO** — *holdout half original*: a second, untouched sample of real records, used only as a reference.

On datasets large enough (by default ≥ 500 rows) a **real holdout** is set aside before anonymization; on smaller ones an **approximate holdout** is drawn by randomly splitting the data when the metrics are computed. See the "Evaluation with a holdout reference" section in the Technical Assessment report.

© Octopize — All rights reserved



Comparison Strategy (Holdout)

To compute the final score returned to the user, we perform the following steps:

1. Obtain the working half **WO** and a disjoint holdout half **HO** from the real data (a real or approximate holdout — see the note above).
2. Take the working avatars **WA**, generated from **WO**.
3. For each variable, compute the Hellinger distance between **WO** and **HO** (reference distance: the natural variability).
4. For each variable, compute the Hellinger distance between **WA** and **HO** (avatar distance). Both distances are measured against **HO**, because the avatars are generated from **WO** and inherit its sampling error.
5. Score each variable as `100 × (1 − √max(0, avatar_distance² − reference_distance²))`, then average the scores over the variables (see the formula below).

Interpreting the Hellinger Mean

- The closer to 100, the more avatars preserve the natural variability of the dataset.
- Values close to 100 indicate that avatars differ from original data as much as original data differ from each other — good signal retention.
- Values below 100 indicate that avatar distributions diverge more from original distributions than the natural variability would suggest — signal degradation.

ℹ Example

Consider a variable where the reference distance (between the two halves of the original data) is 0.05, and the avatar distance is 0.10. The part not explained by natural variability is $\sqrt{(0.10^2 - 0.05^2)} \approx 0.087$, so the variable scores $100 \times (1 - 0.087) \approx 91$. An avatar distance equal to the reference distance would score 100.

Metric Specificity

Advantages:

© Octopize — All rights reserved

- Captures the overall shape of distributions for both continuous and categorical variables.
- Uses the dataset's own natural variability as a reference, making the result more interpretable than a raw distance.
- Provides a bounded score between 0 and 100.

Limitations:

- Sensitive to the choice of binning when estimating distributions for continuous variables.
- May not fully capture local distribution differences.
- Can be impacted by sample size variations between datasets.

Formula

For one variable with probability mass functions p and q over the same bins b:

$$H(p,q) = \frac{1}{\sqrt{2}}\sqrt{\sum_b \left(\sqrt{p_b} - \sqrt{q_b}\right)^2}$$

For each variable v, the avatar distance is measured against the holdout, and only the part that the natural variability does not explain is penalized:

$$\text{score}_v = 100 \times \left(1 - \sqrt{\max\left(0, H(WA,HO)^2 - H(WO,HO)^2\right)}\right)$$

$$\text{Hellinger Mean} = \frac{1}{V}\sum_{v=1}^{V} \text{score}_v$$

Numeric variables are binned on a common grid of $\min\left(\ 200, \max(20, \text{round}\ (\frac{3}{4}(\max - \min))\ \right)$ bins: equal-width bins, or equal-count bins when the values are significantly skewed. Categorical variables use their observed frequencies.

How to Improve This Metric?

To improve the **Hellinger Mean** score, you can adjust parameter **k** to help align distributions more accurately. You can also apply weighted projections by assigning higher importance to critical variables during data generation. Finally, you can use targeted data processing by applying transformations based on variable characteristics.

© Octopize — All rights reserved

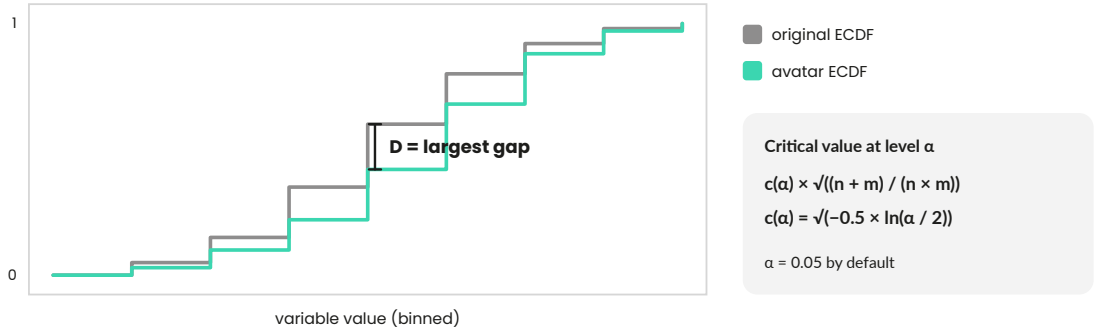
Kolmogorov-Smirnov

The **Kolmogorov-Smirnov** (KS) test is a non-parametric statistical test that compares two empirical cumulative distribution functions (ECDFs) to determine whether they originate from the same distribution. It calculates the maximum absolute difference between the ECDFs of the original and synthetic datasets.

The KS test is particularly useful for datasets with small sample sizes. In the implementation, the cumulative distributions are computed on the same bins as the other univariate metrics (see the formula below).

KOLMOGOROV-SMIRNOV

Largest gap between the two cumulative distributions



A smaller D means the avatar distribution is closer to the original one.

Interpreting Kolmogorov-Smirnov

The KS test provides a p-value that indicates whether the null hypothesis (the distributions are identical) can be rejected:

- p > 0.05: No significant difference between distributions.
- p < 0.05: Suggests the synthetic data differs from the original distribution.

Note: The alpha risk in a statistical test is the probability of incorrectly rejecting the null hypothesis when it is actually true. In simple terms, it means detecting an effect or a difference that does not actually exist. For example, in a medical trial, an alpha risk of 5% means there is a 5% chance of concluding that a treatment works when it actually has no real effect. Lowering alpha reduces this risk but increases the chance of missing a real effect.

A smaller KS statistic implies better similarity.

ℹ Example

Consider a dataset with a customer transaction amount variable. The KS test helps determine if the synthetic data maintains the same cumulative distribution. If the synthetic dataset underestimates high-value transactions, the KS statistic will be larger, indicating a divergence from the original data.

Metric Specificity

Advantages:

- Comes with a statistical decision threshold, making it useful for small datasets.

© Octopize — All rights reserved

- Can detect subtle differences in cumulative distributions.

Limitations:

- More sensitive to small sample variations.
- Less intuitive than bounded metrics like Hellinger distance.

Formula

With p and q the binned distributions of the variable in the original (n values) and avatar (m values) data, and F, G their cumulative sums:

$$D = \max_b |F_b - G_b| \qquad D_\alpha = c(\alpha)\sqrt{\frac{n+m}{n\,m}}, \quad c(\alpha) = \sqrt{-\frac{1}{2}\ln\left(\frac{\alpha}{2}\right)}$$

The distributions are considered different at level α (0.05 by default) when $D > D_\alpha$. Numeric variables are binned on a common grid of $\min\left(\ 200, \max(20, \text{round}\ (\frac{3}{4}(\max - \min)))\ \right)$ bins: equal-width bins, or equal-count bins when the values are significantly skewed. Categorical variables use their observed frequencies.

How to Improve This Metric?

To reduce the KS statistics of **Kolmogorov-Smirnov** test and enhance similarity, you can adjust parameter **k** to help align distributions more accurately. You can also apply weighted projections by assigning higher importance to critical variables during data generation. Finally, you can use targeted data processing by apply transformations based on variable characteristics.

© Octopize — All rights reserved

Bivariate

Bivariate utility measures how well the relationships between pairs of variables are preserved in the transformed dataset. This evaluation ensures that key statistical dependencies, such as correlations and mutual information, remain intact after data transformation.

Preserving bivariate utility is crucial for analyses that rely on variable interactions, such as predictive modeling, feature selection, and statistical inference. If the relationships between variables are distorted, downstream analyses may lead to incorrect conclusions, affecting both statistical accuracy and model performance.

> **i Example**
>
> Consider a dataset containing age and income variables. In the original dataset, there is a positive correlation: older individuals tend to have higher incomes due to career progression.
>
> If the transformed dataset fails to preserve this correlation—e.g., by introducing random noise that weakens the relationship or altering the income distribution independently of age—this could mislead economic or sociological studies. Similarly, models trained on the transformed data might fail to generalize to real-world scenarios due to the loss of meaningful variable interactions.

List of Metrics Measuring Bivariate Utility

- Pearson Correlation Coefficient
- Pairwise Correlation Difference
- Correlation Difference Ratio
- Mutual Information and Variation Information

Factors Influencing Bivariate Utility

Several factors impact the preservation of statistical relationships between variables:

- **Non-linear Dependencies**: Traditional correlation measures like Pearson's coefficient capture only linear relationships. If the dataset contains non-linear dependencies, they may not be well preserved unless more sophisticated techniques are applied.
- **Categorical Variables**: Relationships between categorical variables (e.g., job sector and education level) may not be well represented by standard correlation metrics and require specific treatment.
- **Small Sample Sizes**: With a limited number of observations, estimates of correlation and mutual information can be unstable, making it harder to retain bivariate structures.
- **High Missing Data Rates**: If one variable in a pair has many missing values, the computed relationships may be biased or artificially weakened in the transformed dataset.
- **Feature Engineering Effects**: Some transformation techniques may introduce artificial dependencies or remove subtle yet meaningful relationships between variables.

© Octopize — All rights reserved

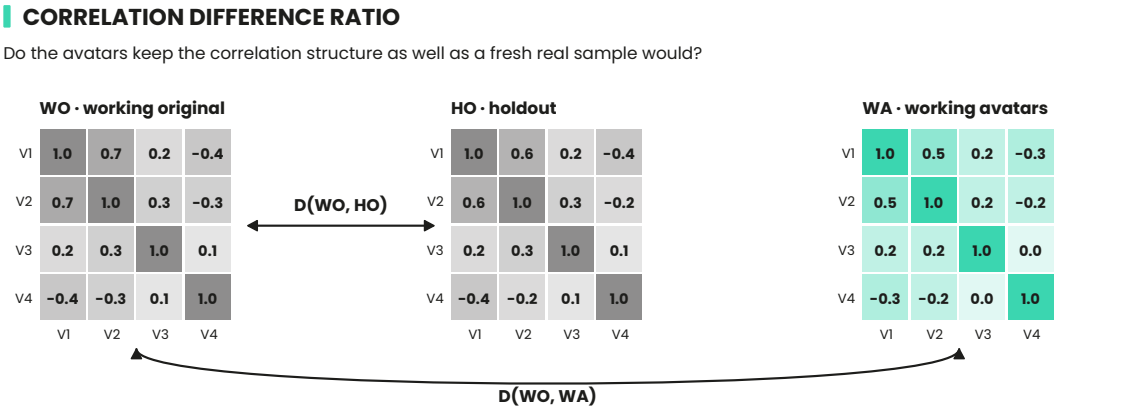
Understanding these factors is key to selecting appropriate transformation methods and ensuring the dataset retains its analytical value.

© Octopize — All rights reserved

Correlation Difference Ratio

The **Correlation Difference Ratio** compares the correlation structure preservation between one half of the original dataset and its avatarized version against the correlation structure between the two halves of the original dataset. For interpretability, it uses the natural variability of the data as a reference.

For both comparisons, it computes the mean absolute difference in correlation coefficients across all pairs of numerical variables.



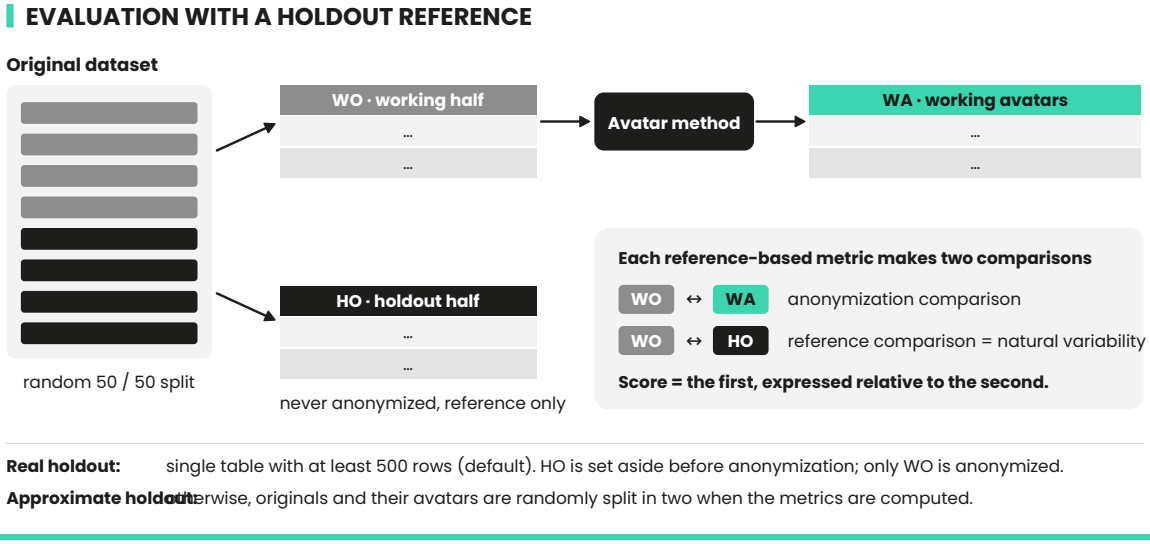
📝 Working half (WO), holdout half (HO), working avatars (WA)

This is a **reference-based** metric: it compares the anonymization (**WO** vs its avatars **WA**) against the natural variability of the real data (**WO** vs a held-out reference **HO**).

- **WO** — *working half original*: the part of the original data that is anonymized.
- **WA** — *working avatars*: the avatars generated from **WO**.
- **HO** — *holdout half original*: a second, untouched sample of real records, used only as a reference.

On datasets large enough (by default ≥ 500 rows) a **real holdout** is set aside before anonymization; on smaller ones an **approximate holdout** is drawn by randomly splitting the data when the metrics are computed. See the “Evaluation with a holdout reference” section in the Technical Assessment report.

© Octopize — All rights reserved



Comparison Strategy (Holdout)

To compute the final score returned to the user, we perform the following steps:

1. Obtain the working half **WO** and a disjoint holdout half **HO** from the real data (a real or approximate holdout — see the note above).
2. Take the working avatars **WA**, generated from **WO**.
3. Compute the mean absolute difference in correlation coefficients between **WO** and **WA** (avatar difference).
4. Compute the mean absolute difference in correlation coefficients between **WO** and **HO** (reference difference).
5. Compute the ratio: `reference_difference / avatar_difference × 100`, capped at 100.

Interpreting the Correlation Difference Ratio

The result estimates the **percentage of the reference correlation preservation obtained through avatarization**:

- The closer to 100, the more avatars preserve the natural correlation structure of the dataset.
- Values close to 100 indicate that avatar correlations differ from original correlations as much as the two halves of original data differ from each other — good correlation retention.
- Values below 100 indicate that avatar correlations diverge more from original correlations than the natural variability would suggest — correlation structure degradation.

Metric Specificity

Advantages:

- Uses the dataset's own natural variability as a reference, making the result more interpretable than a raw difference.
- Provides a bounded score between 0 and 100.
- Intuitive percentage-based interpretation.

Limitations:

© Octopize — All rights reserved

- Only applies to numerical variables using Pearson correlation.
- Sensitive to small correlation values (e.g., a change from 0.05 to 0.15 results in a large percentage difference).
- Does not account for non-linear relationships.
- Can be impacted by sample size variations between datasets.
- More likely to fluctuate on small datasets (fewer than 200 rows), where sampling variance can make the score less stable and harder to interpret.

Formula

For two Pearson correlation matrices A and B over p numeric variables, the mean difference over pairs of variables, in percent:

$$\Delta(A, B) = \frac{1}{p(p-1)} \sum_{i \neq j} 100 \times \frac{|A_{ij} - B_{ij}|}{2}$$

With N the number of rows of WO, the reference is scaled up on large datasets, where the difference between two real samples shrinks:

$$\kappa = \begin{cases} \sqrt{N/2000} & N > 2000 \\ 1 & \text{otherwise} \end{cases} \qquad \text{Correlation Difference Ratio} = \min\left(100, 100 \times \frac{\kappa\, \Delta(R^{WO}, R^{HO})}{\Delta(R^{WO}, R^{WA})}\right)$$

The score is 100 when $\Delta(R^{WO}, R^{WA}) = 0$.

How to Improve This Metric?

To improve the **Correlation Difference Ratio** score, you can adjust parameter **k** to help align correlations more accurately. You can also apply weighted projections by assigning higher importance to critical variables during data generation. Finally, you can use targeted data processing by applying transformations based on variable characteristics. Please consider using alternative metrics (e.g., mutual information) if the data contains non-linear relationships or categorical variables.

© Octopize — All rights reserved

Mutual Information

Mutual Information (MI) measures the dependency between two variables by quantifying the amount of information gained about one variable by knowing the other. Unlike Pearson correlation, MI works with both numerical and categorical variables. It is based on probability distributions rather than direct correlations allowing to capture non-linear dependencies.

Mathematically, MI is defined as:

$ MI(X, Y) = *{x X}* {y Y} p(x, y) ( ) $

where:

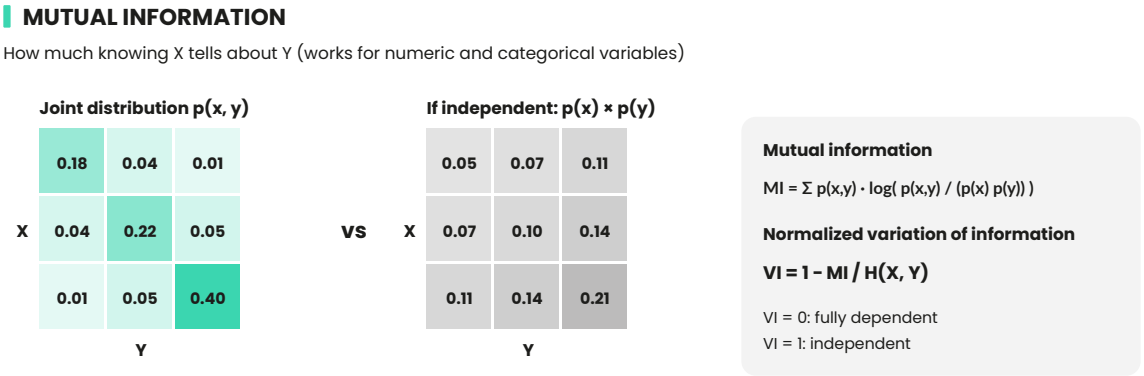
- $ p(x, y) $ is the joint probability mass function (PMF) of X and Y.
- $ p(x) $ and $ p(y) $ are the marginal PMFs of X and Y.

Variation Information (VI) is a derived measure from MI, which normalizes dependency values between 0 (full dependency) and 1 (complete independence). Mathematically, VI is defined as:

$$VI(X,Y) = 1 - \frac{MI(X,Y)}{H(X,Y)}$$

where $H(X,Y)$ is the joint Shannon entropy of X and Y:

$$H(X,Y) = -\sum_{x\in X}\sum_{y\in Y} p(x,y)\log p(x,y)$$



The further p(x, y) is from p(x) × p(y), the higher the MI. Compare it on original and avatars.

Interpreting Mutual and Variation Information

- High MI: Strong dependency between variables.
- Low MI: Weak dependency.
- VI = 0: One variable fully explains the other.
- VI = 1: The variables are completely independent.

i Example

Consider a dataset with two variables:

© Octopize — All rights reserved

- "Hours Studied" (X)
- "Exam Score" (Y)

In the original dataset, suppose **Mutual Information** between these variables is 0.85, indicating a strong relationship.

However, after transformation, if MI drops to 0.40, it suggests that a significant amount of information is lost, making the transformed dataset less reliable for predicting exam scores.

For Variation Information:

- If $ VI_{} = 0.2 $ and $ VI_{} = 0.6 $, it means that the transformed dataset has weakened the dependency between hours studied and exam scores.

Metric Specificity

Advantages:

- Unlike Pearson correlation, works for categorical and numerical data.
- Detects non-linear relationships.
- Not affected by outliers.

Limitations:

- Computationally more intensive.
- Harder to interpret than correlation coefficients (non linear scale).

Formula

With $p(x, y)$ the joint probability mass function of two variables (a 20 × 20 grid for two numeric variables; for a numeric and a categorical variable, a 20-bin histogram of the numeric variable within each category; observed frequencies for two categorical variables), and natural logarithms:

$$MI(X,Y) = \sum_x \sum_y p(x,y) \ln \frac{p(x,y)}{p(x)\,p(y)} \qquad H(X,Y) = -\sum_x \sum_y p(x,y) \ln p(x,y)$$

$$VI(X,Y) = 1 - \frac{MI(X,Y)}{H(X,Y)} \quad (VI = 0 \ \text{ if } \ H(X,Y) = 0)$$

To compare two datasets, the variation of information matrices are compared over the $p(p-1)/2$ pairs of variables:

$$\text{VI score} = \frac{2}{p(p-1)} \sum_{i>j} \left| VI_{ij}^{\text{orig}} - VI_{ij}^{\text{avatar}} \right|$$

How to Improve This Metric?

To enhance **Mutual Information** similarity, you can adjust parameter **k** to help align distributions more accurately. You can also apply weighted projections by assigning higher importance to critical variables during data generation. Finally, you can use targeted data processing by apply transformations based on variable characteristics. Please consider using alternative metrics (e.g., mutual information) if the data contains non-linear relationships or categorical variables.

© Octopize — All rights reserved

Pairwise Correlation Difference

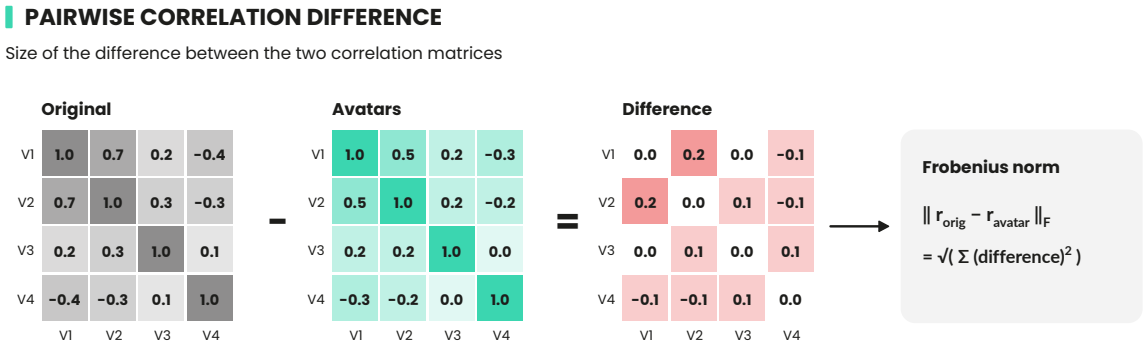
The **Pairwise Correlation Difference** (PCD) quantifies the overall difference in correlation structures between the original and transformed datasets. It is computed as the matrix norm of the difference between the Pearson correlation matrices of both datasets.

Mathematically:

$ PCD = || r_{} - r_{} || $

where $ r_{} $ and $ r_{} $ are the correlation matrices of the original and transformed datasets, respectively.

A smaller PCD value indicates that the correlation structures in the transformed dataset are close to the original dataset.



Interpreting Pairwise Correlation Difference

- **PCD = 0**: The transformed dataset has identical correlations to the original.
- **Low PCD**: The transformed dataset retains most correlation structures.
- **High PCD**: The correlations between variables have significantly changed, which may impact analysis reliability.

ℹ Example

Suppose a dataset contains age, income, and education level. If the original dataset has high correlations (e.g., age vs. income = 0.65, education vs. income = 0.50), but the transformed dataset has lower correlations (0.30 and 0.20, respectively), the PCD metric would indicate a significant loss of correlation structure.

Metric Specificity

Advantages:

- Captures overall structural correlation differences rather than just single variable pairs.
- Easy to interpret: smaller values mean better correlation preservation.
- Useful when analyzing datasets with multiple correlated variables.

© Octopize — All rights reserved

Limitations:

- Sensitive to transformations affecting multiple variables simultaneously.
- Does not capture non-linear relationships (only Pearson-based correlations).
- Ignores categorical variable relationships.

Formula

With R^{orig} and R^{avatar} the Pearson correlation matrices of the original and avatar data (missing differences are set to 0):

$$\text{PCD} = \| R^{\text{orig}} - R^{\text{avatar}} \|_F = \sqrt{\sum_i \sum_j \left(R_{ij}^{\text{orig}} - R_{ij}^{\text{avatar}} \right)^2}$$

How to Improve This Metric?

To lower **Pairwise Correlation Difference** similarity, you can adjust parameter **k** to help align distributions more accurately. You can also apply weighted projections by assigning higher importance to critical variables during data generation. Finally, you can use targeted data processing by apply transformations based on variable characteristics.

© Octopize — All rights reserved

Pearson Correlation

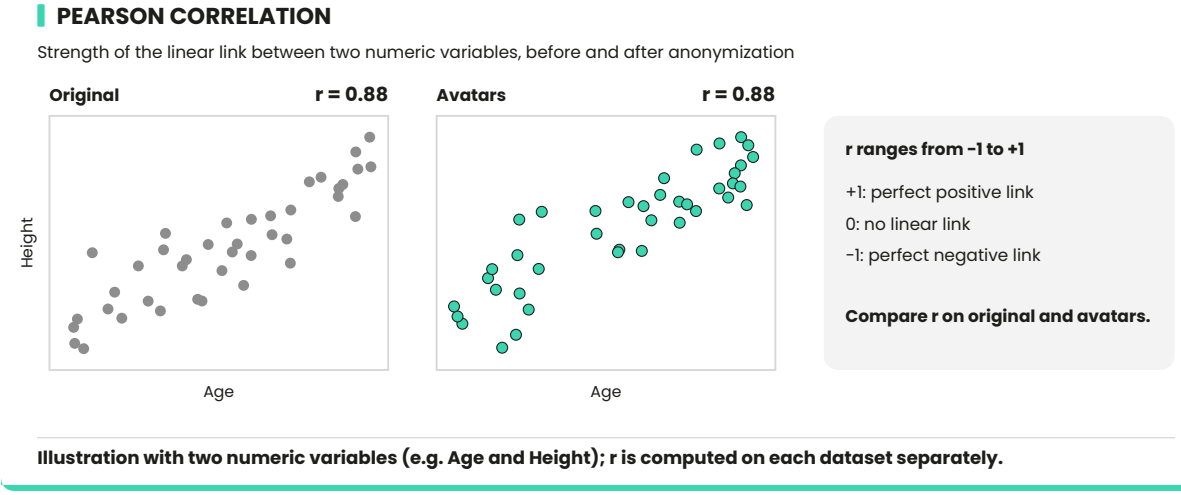
Pearson's correlation coefficient (denoted as **ρ or r**) measures the **linear relationship** between two numerical variables. It quantifies how strongly and in which direction one variable varies as the other changes. Mathematically, it is defined as:

$$r(X,Y) = \frac{\sum(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum(X_i - \bar{X})^2}\sqrt{\sum(Y_i - \bar{Y})^2}}$$

where X and Y are two numerical variables, and $\bar{X}$, $\bar{Y}$ are their respective means.

Values range from **-1 to 1**:

- **+1**: Perfect positive correlation (as one variable increases, the other does too)
- **0**: No correlation (no linear relationship)
- **-1**: Perfect negative correlation (as one variable increases, the other decreases)



Interpreting Pearson Correlation Coefficient

A high absolute value (close to 1) indicates a strong linear relationship, while values near 0 suggest a weak or nonexistent linear relationship. However, **Pearson correlation** only captures linear dependencies. If two variables have a non-linear relationship (e.g., quadratic or exponential), **Pearson's coefficient** may be close to 0 even if the variables are strongly related.

i Example

Consider a dataset containing age and income. If older individuals tend to have higher incomes, **Pearson's coefficient** might be 0.65, indicating a moderately strong positive linear relationship.

If the transformed dataset has a **Pearson coefficient** of 0.20 between the same variables, it suggests that the transformation disrupted the original correlation, which could impact analyses based on income prediction.

Metric Specificity

Advantages:

© Octopize — All rights reserved

- Simple and widely used in statistics and machine learning.
- Provides an intuitive measure of linear relationships.
- Easy to compute and interpret.

Limitations:

- Restricted to linear dependencies**: It cannot detect non-linear relationships.
- Only applicable to continuous numerical variables.
- Sensitive to outliers, which can significantly alter correlation values.

Formula

The Pearson correlation matrix R is computed on the numeric variables of each dataset (constant or empty columns are removed first):

$$R_{XY} = \frac{\sum_i (X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_i (X_i - \bar{X})^2} \sqrt{\sum_i (Y_i - \bar{Y})^2}}$$

The original matrix R^{orig} and the avatar matrix R^{avatar} are then compared, for instance with the Correlation Difference Ratio or the Pairwise Correlation Difference.

How to Improve This Metric?

To enhance **Pearson coefficient** similarity, you can adjust parameter **k** to help align distributions more accurately. You can also apply weighted projections by assigning higher importance to critical variables during data generation. Finally, you can use targeted data processing by apply transformations based on variable characteristics. Please consider using alternative metrics (e.g., mutual information) if the data contains non-linear relationships or categorical variables.

© Octopize — All rights reserved

Multivariate

Multivariate utility measures how well the joint relationships across multiple variables are preserved in the transformed dataset. This evaluation ensures that complex interactions, higher-order dependencies, and multi-variable structures remain intact after data transformation.

Unlike univariate and bivariate utility, which focus on single-variable and pairwise relationships, multivariate utility assesses the global statistical structure of the dataset. It is particularly important for high-dimensional analyses, such as machine learning models, clustering, and multivariate statistical tests.

Preserving multivariate utility is critical for applications that rely on the interplay of multiple variables, such as predictive modeling, dimensionality reduction, and latent structure learning. If the transformation disrupts these multi-variable interactions, it can lead to biased models, incorrect statistical inferences, and reduced explainability.

ℹ Example

Consider a dataset containing demographic, economic, and behavioral variables, such as:

- **Age** (numerical)
- **Income** (numerical)
- **Education Level** (categorical)
- **Spending Score** (numerical)

In the original dataset, there may be a latent structure where:

- Higher education levels are associated with higher incomes.
- Income and age are positively correlated due to career progression.
- Spending scores tend to vary depending on income level and age group.

If a data transformation distorts these relationships—e.g., by independently modifying each variable without maintaining their joint distribution—it can break the underlying structure of the dataset. This could lead to:

- Incorrect clustering results (e.g., grouping high-income and low-income individuals together).
- Biased predictive models (e.g., a spending prediction model failing to capture real-world behaviors).
- Distorted dimensionality reduction (e.g., PCA or t-SNE failing to reflect true variable relationships).

Ensuring multivariate utility allows for faithful data analysis and robust model generalization.

© Octopize — All rights reserved

List of Metrics Measuring Multivariate Utility

- Multivariate Hellinger Distance

Factors Influencing Multivariate Utility

Several factors can impact the preservation of multivariate statistical structures:

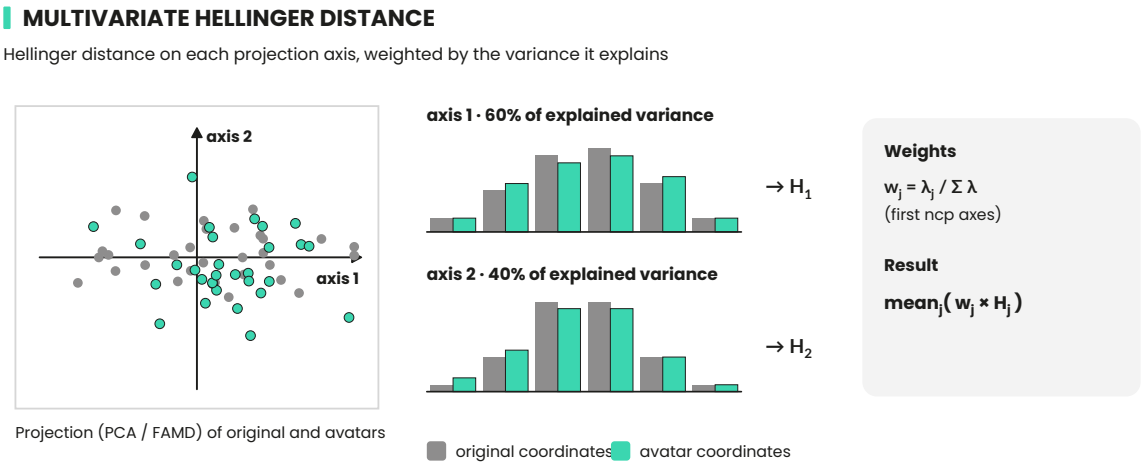
- **Higher-Order Dependencies**: Some datasets contain interactions that emerge only when multiple variables are considered together (e.g., nonlinear interactions between demographics, spending, and credit scores). Standard correlation metrics may fail to capture these.
- **Dimensionality Reduction Effects**: Many transformation methods, such as PCA-based or autoencoder-driven techniques, project data into a lower-dimensional space. If this process is too aggressive, important multivariate relationships may be lost.
- **Non-Linear Relationships**: While bivariate correlations measure linear dependencies, many real-world datasets contain complex, nonlinear dependencies that require more sophisticated techniques for preservation.
- **Feature Importance in Predictive Models**: Some variables contribute more to predictive performance than others. If transformation disrupts these key features, the overall predictive power of models may decline.
- **Sparse or Imbalanced Data**: When certain variable combinations are rare (e.g., low-income individuals with high spending scores), transformation methods may struggle to preserve such **minority patterns**.

Understanding these factors is key to ensuring the dataset remains representative of the real-world data and suitable for downstream analyses.

© Octopize — All rights reserved

Multivariate Hellinger Distance

The **Hellinger distance** is a statistical measure used to quantify the similarity between two probability distributions. It builds on the concept of the Hellinger distance and applies it to the distributions along projection axes, weighted by the percentage of variance explained by each axis. It measures the distance between the Probability Mass Functions (PMFs) of the original dataset and its synthetic counterpart. The distance ranges from 0 (identical distributions) to 1 (completely disjoint distributions).



Interpreting Hellinger Distance

A low **Multivariate Hellinger Distance** indicates that the multivariate structure of the anonymized dataset is similar to that of the original data. Higher values suggest that the relationships between variables have been statistically altered, which may affect the usability of the data.

Typically:

- A value above 0.1 suggests that distributions on projection exis are almost identical.
- A value between 0.1 and 0.3 suggests high fidelity of the generated data.
- Values above 0.3 indicate distortions due to a lack of fidelity.

i Example

Consider a dataset in which three distinct clusters of individuals can be identified, each characterized by significantly different values across a set of variables. A low **Multivariate Hellinger Distance** will indicate that this cluster structure has been well preserved. Conversely, a high distance will suggest that this information has been altered or even lost.

Metric Specificity

Advantages:

- Captures the preservation of multivariate clustering statistics.

© Octopize — All rights reserved

- Not sensitive to dimensionality, as it computes a weighted average of distances per dimension based on the percentage of variance explained.
- Provides a bounded measure between 0 and 1, making it interpretable.

Limitations:

- Less sensitive on datasets with weak or no clear structure.

Formula

Original and avatar data are projected on the same model. With λ_j the share of variance explained by component j, over the q components that explain some variance:

$$w_j = \frac{\lambda_j}{\sum_{k=1}^{q} \lambda_k} \qquad H_j = H\left(p_j^{\text{orig}}, p_j^{\text{avatar}}\right)$$

where p_j is the binned distribution of the coordinates on component j and H the Hellinger distance. The reported value is the mean of the weighted distances:

$$\text{Multivariate Hellinger Distance} = \frac{1}{q} \sum_{j=1}^{q} w_j \, H_j$$

How to Improve This Metric?

To reduce the **Multivariate Hellinger distance** and enhance similarity, you can adjust parameter **k** to help align distributions more accurately. You can also apply weighted projections by assigning higher importance to critical variables during data generation. Finally, you can use targeted data processing by apply transformations based on variable characteristics.

© Octopize — All rights reserved

Technical Assessment

The technical assessment focuses on the **residual re-identification risk** of a dataset. This report is purely technical and relies on the three criteria defined by the EDPB (European Data Protection Board) that are evaluated using privacy metrics.

These state-of-the-art metrics mimic attack scenarios to concretely evaluate:

- **Singling out**
- **Linkability**
- **Inference**

While crucial, this report can be complex to interpret for legal or non-technical profiles. Its goal is to provide a quantitative measure of re-identification risk against various attack scenarios.

Evaluation with a holdout reference

Several privacy and utility (signal) metrics are **reference-based**: they measure how much the anonymization changes the data *relative to the data's own natural variability*. To provide that reference, the original dataset is divided into two halves:

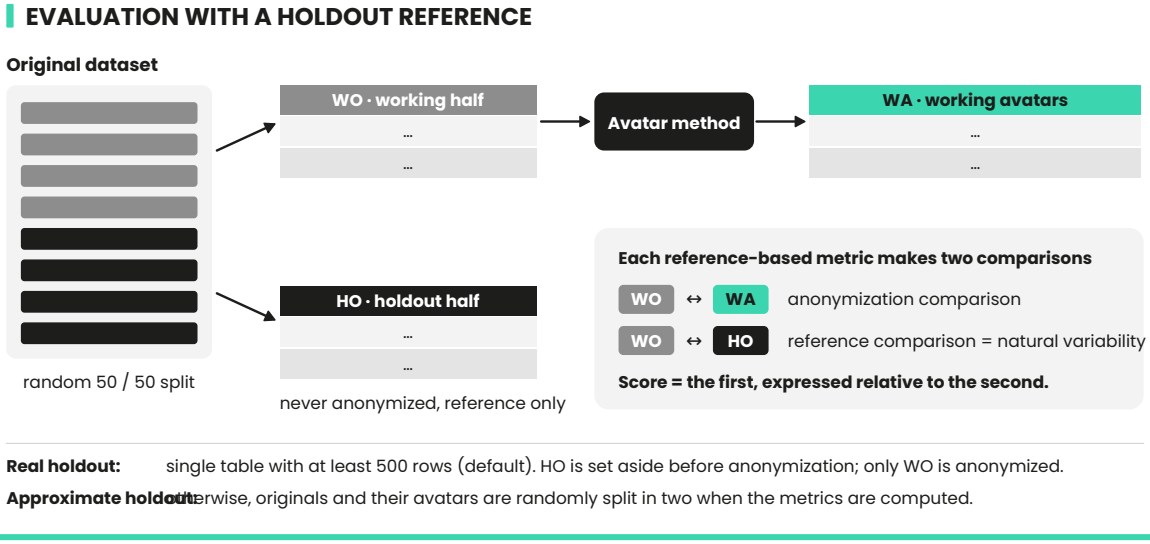
- **Working half (WO)** — the half that is anonymized. Its anonymized counterpart is the **working avatars (WA)**.
- **Holdout half (HO)** — a second, untouched sample of real records, set aside purely as a reference. It is never anonymized.

Each reference-based metric then makes two comparisons and expresses one relative to the other:

- the **anonymization comparison** — WO against WA (real records against their avatars), and
- the **reference comparison** — WO against HO (two independent samples of real data).

Because WO and HO are two disjoint samples of the *same* population, the WO-vs-HO comparison captures the difference expected between any two samples of the real data — its natural variability. Reporting the anonymization comparison relative to this reference is what makes the metric interpretable: a value near 100% means the avatars differ from the originals about as much as two samples of real data differ from each other.

© Octopize — All rights reserved



Real vs approximate holdout

- **Real holdout** — when the dataset is large enough (by default, at least 500 rows), a genuine holdout is set aside *before* anonymization: only WO is anonymized and HO is never seen by the algorithm. This is the most faithful reference and the preferred mode.
- **Approximate holdout** — for smaller datasets, where reserving a real holdout would leave too few rows to anonymize, the reference is approximated by randomly splitting the available original data in two when the metrics are computed. This reuses data the algorithm has already seen, so the reference is slightly optimistic.

The generated technical report states, next to each table's results, which strategy was used.

Which metrics use the holdout reference

Reference-based (computed on WO / WA / HO): Distance to closest, Closest distances ratio, Closest rate, Membership inference and Attribute inference (privacy); Hellinger mean and Correlation difference ratio (utility).

Computed on the whole dataset vs its avatars (no holdout reference): Hidden rate, Local cloaking, Direct match protection, Knowledge protection and Linkability protection.

📝

When a dataset is too large to anonymize in a single pass, it can be split into several batches to improve efficiency — each batch is anonymized independently. The full dataset is anonymized across all batches, but these metrics are computed on the first batch only, as a representative sample. See Batch Processing.

Report Example

📄 See an example technical report

© Octopize — All rights reserved

Impact Assessment (PIA)

The Impact Assessment (or *Privacy Impact Assessment*) complements the technical assessment. Where the technical report measures a theoretical re-identification risk, the impact assessment evaluates the **likelihood** and **severity** of this risk in the real world.

This report takes into account the global context of data usage: - **Probability of occurrence**: Are the elements needed to perform an attack reasonably accessible? - **Impact**: What would be the consequences if an attack were successful, considering data sensitivity? - **Trust level**: Who are the data recipients? - **Data subjects**: Who are the individuals concerned? - **Demographic variables**: How many variables could be used for attacks?

Reasonable Means and the Attacker Model

The impact assessment should not be reduced to a theoretical "all possible attacks" exercise. The relevant question is whether a **reasonable attacker**, with the **means reasonably likely to be used** in the real world, could re-identify individuals from the released dataset.

This includes:

- access to external records or public information,
- realistic linkage strategies,
- practical technical capabilities and time investment,
- and the actual security context of sharing.

This is why the assessment is contextual: the same dataset may be acceptable under controlled access and rejected for broad external release.

For a practical checklist on how "reasonable means" is evaluated in real cases, see How do we assess whether a means is reasonable?.

Decision Making

These two reports are complementary and allow for a complete vision for the final decision: **is the residual re-identification risk reasonable regarding the context?**

- **If yes**: Data is considered anonymous in this context.
- **If no**: Additional measures can be added (encryption, limited sharing, etc.).

Customization Possibility

The automatically generated impact assessment report adapts based on the data usage context, sensitivity, and trust level provided by the user. The format of this report is also intentionally editable (docx) to meet the specific customization and arbitration needs of each organization or industry sector.

Report Example

📄 See an example Impact Assessment report

© Octopize — All rights reserved

03

User Guide

User Guide

Welcome to the **User Guide** section of the Avatars documentation.
This is where theory turns into practice.

Whether you're here for a quick hands-on experience or looking to deeply integrate Avatars into your data workflows, this guide will show you how to use the tool efficiently and securely.

Choose your path 👇

We offer two complementary ways to learn how to use Avatars, depending on your needs and level of familiarity.

🚀 Quickstart – *Get started in under 5 minutes*

If you're new to Avatars or just want to try it out quickly, we recommend starting with one of our Quickstart guides.
They'll walk you through a simple example using either the **Web client** or the **Python client**, so you can see how it works—fast.

👉 Go to the Quickstart

🔍 Step-by-Step – *Master the tool with detailed instructions*

If you're planning a more advanced or production-grade use of Avatars, the step-by-step guide is for you.
It covers the entire anonymization process in detail, broken down into three key stages:

1. **Prepare** – Understand data requirements and how to format your dataset
2. **Run** – Launch the anonymization using the Web or Python client
3. **Enhance** – Evaluate quality, review privacy reports, and export your synthetic data

👉 Go to the Step-by-Step guide

Whichever path you choose, you can switch at any time or explore both.
Let's get started! ✨

© Octopize — All rights reserved

Quickstart

Welcome to the quickstart guide for using **Avatar**! This tutorial will help you get started with both the Python client and the Web client. Choose the section that fits your needs below:

- Python Client
- Web Client

Python Client

The **Python Client** allows you to interact with the Octopize API directly within your Python environment. Here's how to quickly get started with Avatarization on a single table:

Installation

To get started, you'll need an Octopize account. Once your credentials are set up, you can use the Python client with any IDE of your choice (e.g., VS Code, PyCharm, Jupyter) or simply run it from the command line.

You can install the package from PyPI. If you are migrating from the Github Releases to PyPI (prior to 0.7.3, May 7th 2024), make sure to uninstall the old package first.

```
pip uninstall avatars
```

Then, install the package from PyPI. Choose the latest version at https://pypi.org/project/octopize.avatar/

➤ With pip:

```
pip install octopize.avatar
```

➤ With poetry:

```
poetry add octopize.avatar
```

Step 1: Setup and Authentication

Start by importing the necessary libraries and setting up your authentication using an API key.

```
from avatars.manager import Manager

# Initialize the Manager with an API key (the base URL will be handled automatically
by default)
manager = Manager(api_key=os.environ.get("AVATAR_API_KEY"))
```

Step 2: Uploading Your Data

Once authenticated, you can create a `runner` to upload your data table to the server. It is now recommended to assign a unique name (`set_name`) to your project. In this example, we'll use a CSV file `wbcd.csv`.

```
from avatars.models import ReportLanguage

# Initialize the runner
runner = manager.create_runner(set_name="test_wbcd",
report_language=ReportLanguage.EN)
```

© Octopize — All rights reserved

```
# Add the table to the runner
runner.add_table("wbcd", "fixtures/wbcd.csv")
```

The wbcd table is now ready for anonymization.

Step 3: Avatarization Parameters and Execution

Next, define the **avatarization parameters** and run the process. Parameters can be set automatically or manually.

```
# Set anonymization parameters automatically and display them
runner.advise_parameters()
runner.print_parameters()

# OR Set anonymization parameters manually
runner.set_parameters("wbcd", k=10)  # The k value allows you to adjust the privacy
level

# Run the anonymization pipeline
runner.run()

# Get all results after running the anonymization
results = runner.get_all_results()
```

Here, `k=10` is the number of nearest neighbors used in the neighbor-based method. You can adjust this value based on your desired privacy level. To learn more about anonymization parameters, please refer here.

Step 4: Retrieve Avatarized Data

Once the anonymization is complete, you can retrieve and inspect the avatarized data:

```
# Print a preview of the avatarized data
print("Avatar data:")
print(runner.shuffled("wbcd").head())
```

Step 5: Retrieve Privacy and Signal Metrics

Octopize provides privacy and signal metrics to evaluate the quality of the synthetic data produced:

```
# Print privacy metrics
for key, value in runner.privacy_metrics("wbcd").items():
    print(f"{key}: {value}")

# Print signal metrics
for key, value in runner.signal_metrics("wbcd").items():
    print(f"{key}: {value}")
```

Step 6: Download Avatarization Report

Once you have found the optimal set of parameters, you can download two reports. A technical report that compiles privacy and utility metrics as well as an impact analysis report.

```
from avatar_yaml.models.parameters import ReportType

# Download the residual re-identification risk assessment report
runner.download_report('my_report.pdf')
```

© Octopize — All rights reserved

```
# Download the impact analysis report
runner.download_report("my_report_pia.docx", report_type=ReportType.PIA)
```

Once you've followed these steps, you're ready to explore further and fine-tune your anonymization process!

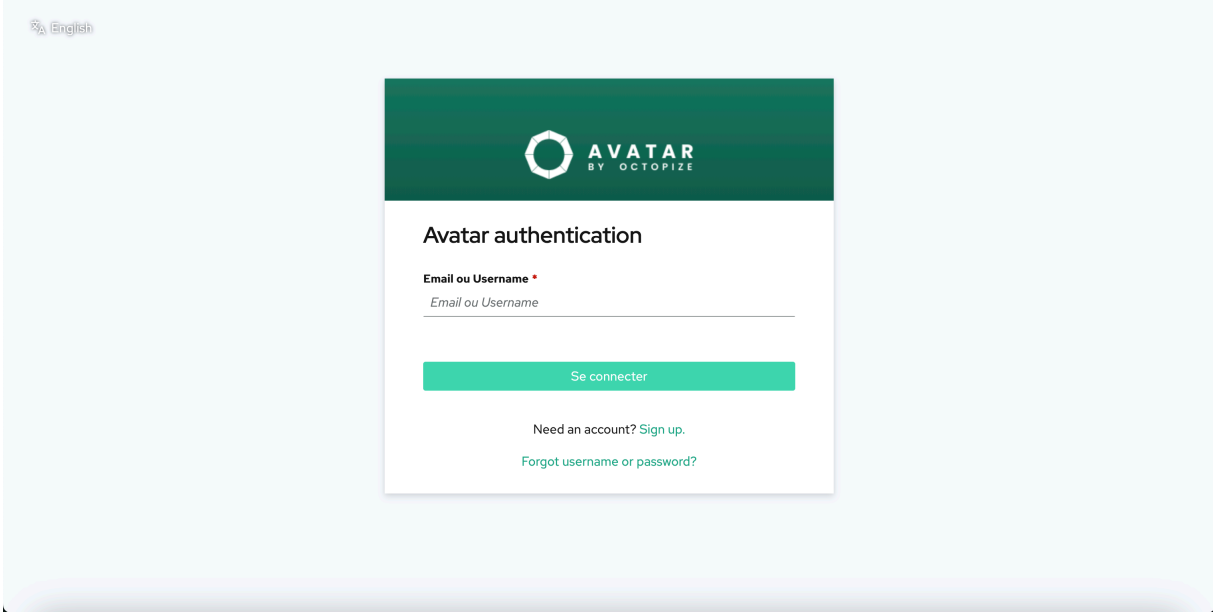
For a deeper, step-by-step walkthrough, please refer to our Step-by-Step Guide.

Web client

The Web Client offers a simpler, button-click interface for users who prefer to interact with Octopize through a graphical interface. You can follow the same anonymization steps without writing any code.

Step 1 — Log in to the Web Platform

Navigate to the Octopize web portal and enter your credentials to authenticate. After logging in, you will be redirected to the dashboard.



Step 2 - Upload your data

- Click on the "Create a Job" button to begin.
- You can then choose from several preconfigurations based on your needs. For anonymization using the Avatar method, click on "Avatar Anonymization".
- Give your project a clear and recognizable name — this will make it easier to locate later.
- You can optionally define information about your use case context. This will help tailor the impact analysis report generated in the final step.
- On the upload page, click the "+" button under Load dataset(s) to import the datasets you wish to anonymize. For first-time users, we recommend using one of the example datasets provided.
- Once everything is configured, click "Next" to continue. An automatic data formatting step should then start.

© Octopize — All rights reserved

🎬 *This step is demonstrated in a video. Watch it in the online documentation at docs.octopize.io.*

Step 3: Set Parameters and Run Avatarization

Once uploaded you will have an overview of your tables.

For each table: - Select the variables to include, remove, or freeze in the anonymization and verify their data types are correct. - In multi-table setups, specify which variables act as identifiers — primary keys or foreign keys — to maintain relational integrity. - Mark variables likely to be known by a potential attacker and adjust their influence in the projection space accordingly. - Once everything is set, click “Next” to proceed to the parameters configuration. If you’re working with a single table, you’ll proceed directly to the next step. Otherwise, you’ll need to define the relationships between your tables before continuing. - An automatic parameter preconfiguration step should start. - Depending on your use case, you can configure the parameters for: - Anonymization - Augmentation - Privacy metrics - Once you’re done with the configuration, click “Next”.

💡 *Tip: Enable expert mode to access a wider range of advanced parameters.*

🎬 *This step is demonstrated in a video. Watch it in the online documentation at docs.octopize.io.*

Step 4: Retrieve Results and Avatarized Data

You’re almost there!

- You can download the YAML configuration file as well as the Python script corresponding to the parameters you just defined — perfect for reproducibility or future automation.
- Click “Run Now” to launch the selected jobs: anonymization, privacy metrics calculation, utility metrics calculation, and report generation.
- Once finished, click on “Metric Details” to access the results page.
- From this page, you will be able to:
  - **View** privacy and utility scores,
  - **Download** the generated avatars,
  - **Visualize** statistical comparisons between your original data and the created synthetic data,
  - **Retrieve** the complete anonymization reports (technical + impact analysis) for your documentation or compliance needs.

💡 *Tip: Enable expert mode to access all advanced results.*

🎬 *This step is demonstrated in a video. Watch it in the online documentation at docs.octopize.io.*

© Octopize — All rights reserved

📚 **Want to go further?**
Download our complete suite of tutorials for the Python client — featuring advanced capabilities and various use cases, from multi-table anonymization to time series and beyond.

👉 Download the full Python tutorials

© Octopize — All rights reserved

Anonymization Guide

Welcome to the **Anonymization Guide**.

This section is designed for users who want to make the most of the Avatars anonymization tool—whether you're aiming for optimal data utility, compliance, or integration into a production workflow.

We'll walk you through the full pipeline of an anonymization project, divided into three clear stages:

🟢 1. Prepare

Data preparation is the most impactful step in the entire anonymization process. A well-prepared data leads to significantly better results in terms of both privacy and utility.

In this step, you'll learn:

- How to format your data for compatibility with the Avatar method
- Required preprocessing rules (e.g., file format, data types, structure)
- Recommended best practices to improve output quality
- How to handle special cases like relational databases or time series
- Which preprocessing steps can be automated directly in the tool using built-in processors

The goal: make high-quality anonymization **easier and more effective** right from the start.

👉 Go to Prepare

🔵 2. Run

Once your data is ready, it's time to run the anonymization process.

In this step, you'll learn how to:

- Configure the anonymization parameters (with the help of the built-in Advisor tool)
- Launch the process using either the **Web Client** or the **Python Client**
- Generate and download reports (technical and impact) to evaluate privacy and utility metrics

🔧 Whether you're integrating anonymization into a pipeline or just exploring via the interface, both clients are fully supported.

To guide your experience:

- The **Web Client** is perfect for quick experimentation and non-technical users.
- The **Python Client** is designed for advanced use cases, integration into ETL workflows, and fine-tuned control.

👉 Go to Run

© Octopize — All rights reserved

🟣 3. Enhance

After the synthetic data has been created, this final stage helps you assess and refine the result.

You'll learn how to:

- Evaluate data quality and fidelity
- Review technical assessment and impact assessment reports
- Export and reuse your synthetic dataset

👉 Go to Enhance

Each stage includes practical instructions, tips, and references to help you succeed.

Take your time—or jump directly to the part you need.

Data Augmentation Guide

Looking to expand your dataset, the **Data Augmentation** feature allows you to generate additional synthetic records while preserving the statistical properties of your original data.

This is useful when:

- You need more data for training machine learning models
- Your dataset is too small to achieve reliable statistical analysis
- You want to balance underrepresented classes in your data

👉 Go to Data Augmentation

© Octopize — All rights reserved

Step 1: Prepare your data

Preparing your dataset before anonymization is a crucial step to ensure high-quality results. This step has the most significant impact on the quality of the generated dataset. Since the anonymization process itself is relatively short, we recommend dedicating most of the project time to data preparation.

This guide outlines the necessary and recommended steps to structure your data for optimal anonymization. Some preprocessing steps can be performed directly within the tool.

Required Preprocessing Steps

To ensure compatibility with the Avatar anonymization process, your dataset **must** adhere to the following requirements:

- **Handle Direct Identifiers**: Names, Social Security Numbers, email addresses and other direct identifiers must not be anonymized as regular variables. Configure pseudonymization for these columns: they are set aside before anonymization and replaced with substitutes in the output. Keep primary keys and foreign keys and declare them as identifiers in multi-table setups: the platform manages them and preserves the links between tables. Remove only the identifier columns you do not need in the output.
- **Supported File Formats**: Provide data in **Parquet**, **CSV** or **Pandas DataFrame** format. If using CSV, ensure it is comma-separated.
- **Standardize Date Format**: Convert all date values to `YYYY-MM-DD` format.
- **Standardize Missing Format**: Convert all missing values to the same format.
- **Structured Data Only**: The method supports structured, tabular data. Free text, audio, or video content should be removed or transformed beforehand.

Recommended Preprocessing Steps

While not mandatory, the following best practices enhance anonymization quality:

- **Remove Computed Variables and recompute afterwards**: Avoid derived variables that can introduce redundancy and impact anonymization quality. (e.g., BMI calculated from weight & height)
- **Remove redundant information**: If columns contain duplicated or derivable information, keep only one of them. For example, if you have both “city” and “region” columns, remove the region since it can be inferred from the city name.
- **Express Relationships as Differences**: Where relevant, express relationships between variables as differences rather than absolute values. For example, an end date should be expressed as the number of days since a start date to ensure the chronological consistency.
- **Transform Low-Cardinality Numeric Columns**: Convert numeric columns with fewer than 20 unique values into categorical variables.
- **Regroup unique modalities**: Convert all unique modalities of a categorical variable into a single to avoid projection distortion and prevent from membership inference attack weakness.
- **Ensure you don’t have mixed datatypes**: When dealing with a mixed data types variable (e.g., “< 0”, 1, 2, 3), the method will automatically consider it as a categorical variable. Please convert to the correct data type beforehand.

© Octopize — All rights reserved

- **Split high-dimensional datasets**: When working with datasets containing many continuous variables (more than 200), consider splitting the anonymization into multiple smaller batches of correlated variables to improve quality.
- **Split datasets with heterogeneous populations**: If your dataset combines two or more distinct groups of individuals, consider splitting it into separate batches (each with a sufficient number of records) to improve modeling quality during anonymization.

The Avatar solution includes built-in tools, called Processors, to assist with these preprocessing steps. You can configure the following processors:

Configurable Processors (optional, for specific use cases):

- Relative Difference Processor — Express numeric variables as proportions of a total
- Inter-Record Range Difference Processor — Preserve start/end date relationships and temporal ordering

Automatic Processors (applied transparently without configuration):

- Automatic Processing — Learn about automatic data handling: missing data, datetime conversion, type preservation, decimal precision, skewed variable transformation, and more

Multi-Table Recommendations:

- **Merge one-to-one relationships**: When two tables have a one-to-one relationship, consider joining them into a single table to simplify the schema.
- **Split large heterogeneous tables**: If a table combines multiple sources of information, split it into separate tables with clear purposes. For example, divide a comprehensive “visit” table into separate “patient” and “visit” tables.
- **Choose linkage parameters carefully**: Select linkage parameters based on the cross-table relationships you need to preserve in the anonymized data.

Specific Preprocessing for Relational Databases and Time Series

For use cases involving relational databases or time series (i.e., when multiple records exist per individual), tables must adhere to the following structural constraints:

Each table must contain at least one identifier variable, known as a **primary key**, which holds a unique value for each row.

Relationships between tables are established through **foreign keys**, which reference the primary key of another table. Unlike primary keys, foreign keys do not need to be unique within their table, allowing multiple rows to link to the same referenced entity.

Recommended Dimensionality

The more horizontally structured a dataset is (i.e., the higher the ratio of the number of variables to the number of records), the more likely it is that statistical properties will be affected at an equal privacy level. This is particularly relevant for datasets with few individuals but many variables. To mitigate this, we recommend including only the variables necessary for the identified use case rather than

© Octopize — All rights reserved

attempting to anonymize a dataset intended for multiple use cases simultaneously.

Next Steps: Run the Anonymization

Once your data is prepared, proceed to the Run section to execute the anonymization process. Keep in mind that optimizing an anonymization process is iterative. After evaluating the anonymized results, you may refine the preparation step to further improve the data quality and privacy balance.

© Octopize — All rights reserved

Automatic Processing

Some automatic processors are applied automatically during the avatarization pipeline. They handle data transformations, type preservation, and missing value management without requiring explicit user configuration.

Power Transformations Processor (Box-Cox)

Description

Applies **Box-Cox transformation** to skewed (long-tailed) continuous variables to normalize their distribution. This improves statistical modeling in the avatarization algorithm.

- **When**: Automatically applied to detected skewed variables during preprocessing
- **Why**: Skewed data can distort the anonymization quality; normalized data improves results
- **Reversal**: The transformation is reversed during post-processing to restore original distribution shape

---

Datetime Processor

Description

Automatically handles **date/timestamp** columns: - **Preprocess**: Converts dates → Unix timestamps (seconds since epoch) - **Anonymize**: Works with numeric timestamps - **Postprocess**: Restores dates to original `YYYY-MM-DD` format

Ensures temporal variables can be used for projections.

To Categorical Processor (Automatic Dtype Advice)

Description

Automatically recommends converting numeric variables with < 20 unique values to categorical.

This is part of the `ToCategoricalDataTypeAdviceRule` that runs during the "advice" phase. Variables detected as low-cardinality numeric are treated as categorical to improve anonymization quality.

- **Threshold**: 20 unique values
- **When**: During data preprocessing analysis phase (automatic)
- **Benefit**: Better statistical modeling, prevents Box-Cox issues on variables that should be categorical

© Octopize — All rights reserved

Inter-Record Range Difference Processor

The **Inter-Record Range Difference Processor** (also called **Range Difference Processor**) transforms pairs of start/end columns into an internal representation that preserves their relationships. This is useful for temporal data, date ranges, and ordered pairs.

Description

This processor handles data where pairs of variables represent a range or interval: - Start/end dates: `start_date` and `end_date` - Start/end times: `start_time` and `end_time` - Lower/upper bounds: `min_value` and `max_value`

The processor transforms these pairs into an internal representation that: 1. **Preserves the temporal or logical order** (start < end) 2. **Maintains the distance between the two values** 3. **Allows avatarization** while keeping the structure intact 4. **Automatically sorts by the start variable**

Use case: When your data contains time periods, events with duration, or ordered pairs that must maintain their relative ordering.

Concrete example

Consider a dataset with event data:

```
person_id | event_name | start_date | end_date
1         | vacation   | 2024-01-15 | 2024-01-22
2         | project    | 2024-02-01 | 2024-03-15
3         | conference | 2024-03-20 | 2024-03-25
```

Problem without processor: During avatarization, `start_date` and `end_date` might be transformed independently, potentially creating impossible dates like `end_date < start_date`.

After applying Inter-Record Range Difference Processor:

```
Preprocessing step:
  - Records the original start_date values for each person_id
  - Calculates the duration: end_date - start_date
  - Sorts by start_date internally

During avatarization:
  - Dates are anonymized as a single unit (start_date and duration are preserved
together)

Post-processing step:
  - Reconstructs end_date = avatarized_start_date + original_duration
  - Ensures end_date > start_date always holds
```

Result: Avatars maintain valid date ranges with preserved temporal logic.

How to Use it in Web

In the web interface, the processor is not available yet. You can either use the Python API or process your data on your own before uploading it to the web interface.

How to use it in python

Please refer to the Python documentation for usage details.

© Octopize — All rights reserved

Key Points

- **Order matters**: Processors are applied in the order specified. If combining multiple processors, the start/end processor should typically be applied before others that modify time-based columns.
- **Multiple ranges**: You can apply this processor to multiple start/end pairs by creating separate processor instances.
- **Non-date ranges**: The processor also works with numeric ranges (e.g., `min_score` and `max_score`), as long as the start < end relationship is logical.

© Octopize — All rights reserved

Relative Difference Processor

The **Relative Difference Processor** expresses a numeric variable as a relative difference to the sum of reference variables. This is useful for preserving mathematical relationships between correlated variables.

Description

This processor is designed for scenarios where one variable represents a proportion or share of a total. For example: - `minutes_in_game = f(minutes_played_home + minutes_played_away + minutes_on_bench)` - `income_per_category = f(total_income)`

Instead of treating `minutes_in_game` as an independent variable, this processor expresses it as:

```
minutes_in_game_ratio = minutes_in_game / (minutes_played_home + minutes_played_away + minutes_on_bench)
```

Use case: Preserve the mathematical/logical relationships in your data during avatarization. The ratio is maintained in avatars, ensuring consistency.

Concrete example

Imagine a sports dataset:

```
player_id | minutes_played_home | minutes_played_away | minutes_on_bench | minutes_in_game
1         | 1500                | 800                 | 200              | 2300
2         | 1200                | 1000                | 100              | 2200
3         | 900                 | 1100                | 250              | 2150
```

The `minutes_in_game` is the sum of the three reference variables. Without the processor, these relationships might not be preserved during avatarization.

After applying Relative Difference Processor:

```
During preprocessing:
  minutes_in_game_ratio = minutes_in_game / (minutes_played_home + minutes_played_away + minutes_on_bench)

  player_id | ratio
  1         | 2300 / 2500 = 0.92
  2         | 2200 / 2300 = 0.956
  3         | 2150 / 2250 = 0.955
```

The ratio is avatarized, then **reversed** during post-processing:

```
During post-processing:
  minutes_in_game_avatar = ratio_avatar × (sum of reference variables in avatar)
```

This ensures the mathematical relationship is preserved in the final avatars.

How to Use it in Web

In the web interface, the processor is not available yet. You can either use the Python API or process your data on your own before uploading it to the web interface.

How to use it in python

Please refer to the Python documentation for usage details.

© Octopize — All rights reserved

Step 2: Run Anonymization

Congratulations! Your dataset is now ready for anonymization. This page explains how to perform anonymization using the different clients provided by Octopize. For each client, we will demonstrate the anonymization process, metric calculation, and report generation using a sample dataset.

Anonymization Parameters

This section lists all parameters that influence the anonymization process.
To facilitate the first iteration, the method includes a built-in tool, further described as Advisor, that generates a predefined set of parameters based on the dataset's characteristics.

> ⚠️ **Mutually Exclusive Methods**
>
> The parameters `k` and `fast_dp_epsilon` correspond to two **distinct anonymization methods** (Avatar and AvatarFastDP). **You must set exactly one of them** — they cannot be combined. See Differential Privacy Methods for guidance on choosing the right method.

- **`k`**: An integer representing the number of nearest neighbors to consider (standard Avatar method). The larger the value relative to the dataset size, the higher the level of protection, but at the cost of increased variance loss.
- **`fast_dp_epsilon`**: A float setting the privacy budget for AvatarFastDP (fast DP-flavored synthesis). Must be > 0.
  - The mechanism and its settings (`mechanism`, `gmm_n_components`, `histogram_n_bins`, `bounds_percentile`) are not arguments of `set_parameters()`. Set them in YAML under `avatarization_fast_dp`. See Differential Privacy Methods.
- **`column_weights`**: Defines the importance of each variable during the projection process. Higher values result in greater separation between individuals based on that variable.
- **`ncp`**: Sets the number of dimensions to consider for the KNN algorithm. Lower values prioritize well-represented variables in the projection.
- **`use_categorical_reduction`**: A boolean that enables the transformation of categorical variables into a latent numerical space before projection. This reduces computational cost while preserving most of the original variance.
- **`imputation`**: Used when anonymizing datasets with missing values. If no imputation parameters are provided, missing data will be automatically handled.
  - **`method`**: The imputation method. Available options: `mode`, `median`, `mean`, `knn`, `fast_knn`. If `knn` or `fast_knn` is used, the parameters `k` and `training_fraction` will be applied; otherwise, they will be ignored.
  - **`k`**: Number of nearest neighbors to consider for KNN imputation.
  - **`training_fraction`**: Fraction of the dataset used to train the KNN imputer. Smaller values improve performance.

© Octopize — All rights reserved

- **`exclude_variable_names`**: Allows the exclusion of some variables from anonymization before reassigning them probabilistically. This is useful for categorical variables with a high number of unique values.
  - **`exclude_replacement_strategy`**: The method for reassigning excluded variables. Options:
    - `ExcludeVariablesMethod.COORDINATE_SIMILARITY`: Matches excluded variable values from the most similar original record.
    - `ExcludeVariablesMethod.ROW_ORDER`: Assigns each avatar record the excluded variable values from its original record.
- **`seed`**: An integer used to set a random seed for reproducibility. **Do not set this for production use cases**.

Python Client

It is ideal for integrating anonymization into industrial ETL processes or using it directly after a preprocessing script.

Example Usage

See the GitHub tutorial to explore parameters and features.

Web Client

A web-based interface that allows users to interact with the API using a simple button-click approach.
Ideal for non-data scientists as it provides a quick and intuitive experience, though it is less suitable for industrial automation.

Interpretation and Iteration

Congratulations! You have completed your first end-to-end anonymization process.
You can now proceed to the Enhance page to interpret and improve the results.

If this is your final iteration, be sure to generate and store the anonymization report. This report serves as a risk analysis document, which can be archived in the DPO's processing registry.

© Octopize — All rights reserved

Step 3: Enhance your results

This is the final step of the anonymization process.

This phase is about analyzing the results of your anonymization run and improving them — if necessary — to reach the optimal trade-off between **privacy** and **utility** according to your use case.
It's an **iterative process**: by understanding the results, identifying what needs adjustment, and refining either the dataset preparation or anonymization parameters, you can progressively fine-tune your configuration.

Understand the Results

After the run is complete, you can explore the **privacy and utility metrics** to evaluate the quality of your synthetic data.

➡️ See Metrics Explained

These metrics help you verify whether the anonymization strength is sufficient **while retaining the necessary statistical properties** for your use case.

You can also **complement the built-in metrics** with additional signal-specific evaluations if you have prior knowledge about key indicators you want to preserve.
The more precisely you define your expectations, the better you can guide the anonymization process toward your goals.

Identify What Needs to Be Improved and How

There are two main types of adjustments you might want to make:

- **Improve privacy**
  For example, to ensure that no individual can be identified or singled out from the synthetic dataset.
- **Improve utility**
  For instance, to better preserve distributions, correlations, or predictive value.

Your next move depends on which of these you're prioritizing.

🛠️ Adapt Data Preparation and/or Parametrization

Improving your results can be done in two main ways:

- **Optimize your data preparation**
  See Prepare your data
  This might involve feature selection, variable transformation, or better handling of missing values through processors.
- **Adjust anonymization parameters**
  See Run configuration
  You can change the privacy settings, influence weights, or methodology parameters.

© Octopize — All rights reserved

How to Improve Privacy

- Increase the value of k
- Include as many variables as possible in the anonymization process (avoid freezing too many)
- Reduce the influence of quasi-identifiers likely to be known by attackers

How to Improve Utility

- Lower the value of k to allow better data fidelity
- Freeze key variables you want to preserve as-is
- Use the column weight parameter to give more influence to important variables in the projection space
- Improve data preprocessing through the use of processors

🔁 Re-run and Re-evaluate

Once you've made adjustments, you can re-launch the process using your updated setup.
Evaluate the new results, and repeat if necessary — until you reach the balance that best suits your context.

Anonymization is not one-size-fits-all: it's about **finding the right trade-off for your specific needs**.

© Octopize — All rights reserved

Database Runs (Beta)

> ⚠️ **Beta Feature**
>
> Database runs are currently in **beta**. The API and behavior may change in future releases. Try a run first on a database where a mistake has no consequence, for example one of the demo databases.

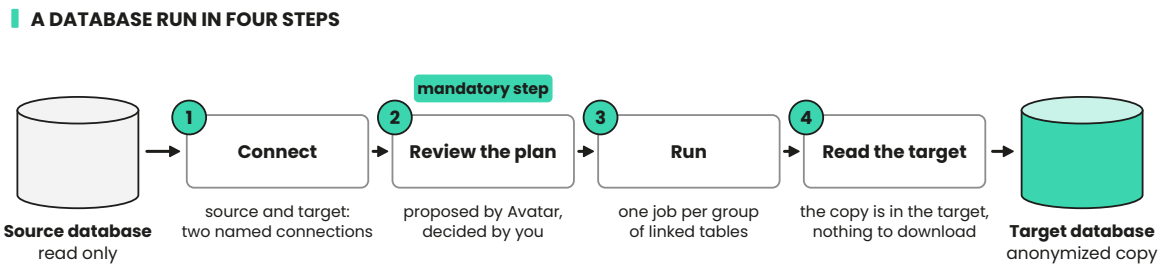
A **database run** reads a database, anonymizes it with the Avatar method, and writes the result into another database. You get a working copy of a production database: the same tables, the same joins and about the same number of rows, without any real individual in it.

Use it to give a test, development or analytics team a database they can query like the real one, without a CSV export in the middle.

> ℹ **Requires the Database Connector**
>
> Database runs need the database connector, an optional component of the platform. It is enabled on the Octopize SaaS. On a self-hosted installation it is off by default: ask your administrator to enable it. Every other feature works without it.

How a Database Run Works

A database run has four steps. The second step is mandatory.



1. **Connect**: you choose a **source** and a **target**. Each one is a named connection stored by Avatar. A job carries the name of a connection, never its password.
2. **Review the plan**: Avatar reads the structure of the source (**schema discovery**) and proposes what to do with each table and each variable (**classification**). You read the proposal, correct it, and confirm it. A run that contains a proposal that was not confirmed is refused when it starts.
3. **Run**: Avatar copies the database. Tables linked by foreign keys are anonymized together, in one job. Each group of linked tables is a separate job.
4. **Read the target**: the result is the target database itself. There is no file to download. Each job also produces the usual privacy and utility metrics and reports.

You can do all of this from the web client or from the Python client.

© Octopize — All rights reserved

What Happens to the Data

Each table gets a **treatment** (copy, pseudonymize, anonymize, purge or ignore) and each variable an **action** (anonymize, pseudonymize, freeze or drop). Avatar proposes them from the names and the contents of the variables; you confirm or correct the proposal. Keys are re-issued so that every join still works, and a **shared entity** gives one individual the same fake identity in every table.

The rules behind these choices, and what they mean for privacy, are described in Database Anonymization. Read it before your first run: in particular, a frozen variable keeps its real values in the copy.

Requirements for the Target

- The target is a **different** database from the source. A run refuses a target that is the same connection as the source, but two different connections that point at the same database are not detected: check this yourself.
- The target uses the **same engine** as the source: PostgreSQL into PostgreSQL, MySQL into MySQL. Avatar does not translate a schema between two database engines.
- The target **already has the schema**: every table that the run writes to, including the tables you purge, must exist in the target. A run does not create tables in a database it reaches through the database connector, and refuses to start when a table is missing, naming it. Create the schema first, for example with `pg_dump --schema-only` restored into the target. The demo targets already have the schema of their demo source.
- A target table that already holds rows is refused, unless you ask the run to empty the target first (**Empty this target before writing** in the web client). The Python client does not offer this option yet.

Every check happens before the first row is written. The error messages name tables and variables, never values.

How the Target Is Emptied

When you ask for the target to be emptied, the run deletes the rows of the tables it writes to, and only of those tables. It never deletes a table. The rows are deleted with `DELETE`, children before parents, in one transaction. ClickHouse uses `TRUNCATE TABLE` on each table, and MongoDB deletes the documents of each collection.

The run refuses to empty the target when a table it does not write to has a foreign key with an `ON DELETE` action (such as `CASCADE` or `SET NULL`) into a table it writes to: deleting the rows would change a table that the run was not asked to touch.

Credits

A database run is charged per table: each group costs **1 credit per table that it anonymizes or pseudonymizes**. Tables that are copied, purged or ignored are free. Schema discovery, the extraction of the rows, and the resolution of shared entities are free. See Credits.

Where to Go Next

- Database Anonymization: what happens to each table, each variable and each key.

© Octopize — All rights reserved

- Database Connections: what you can connect to, and how to add a connection.
- Run a Database Copy in the Web Client
- Run a Database Copy with the Python Client
- For administrators: enable the database connector on a self-hosted Kubernetes installation.

© Octopize — All rights reserved

Database Connections

A database run never carries a password. It carries the **name** of a connection, and Avatar stores the rest. This page explains what you can connect to, who can see a connection, and how to add, test, rename and delete one.

You can manage connections in the web client, in **Account > Database connections**, or from the Python client. They are the same connections.

Supported Engines

Engine	Kind	Status
PostgreSQL	Tables (SQL)	supported
MySQL	Tables (SQL)	supported
MariaDB	Tables (SQL)	supported
Oracle	Tables (SQL)	supported
SQL Server	Tables (SQL)	supported
ClickHouse	Tables (SQL)	supported
MongoDB	Documents (NoSQL)	supported
Snowflake	Tables (SQL)	preview
Google BigQuery	Tables (SQL)	preview, source only
IBM Db2	Tables (SQL)	preview, only on some installations
SQLite	Tables (SQL)	demo databases only

Your installation can offer fewer engines. The web client shows only the engines that your installation can reach. **Preview** means the engine is offered but not yet proven against a real server: try it first on a database where a mistake has no consequence.

Every engine, the demo databases included, is opened by the database connector. Without it, an installation reaches no database, PostgreSQL included.

In MongoDB, each collection is read as a table. Its variables are the top-level fields found in a sample of 500 documents, `_id` is the key, and nested values arrive as JSON text. The Avatar method cannot model a JSON value: drop the nested fields that the users of the copy do not need. A collection has no foreign keys, so each collection is anonymized on its own. To give one individual the same identity in several collections, declare a shared entity.

Each engine has its own form. You never write a connection string yourself.

© Octopize — All rights reserved

Engine	Fields	Behind **More options**
PostgreSQL	host, port, database, user, password	SSL mode: `prefer` (default), `disable`, `require`, `verify-ca`, `verify-full`
ClickHouse	host, port, database, user, password	TLS (off by default)
MongoDB	host, port, database, user, password	auth source, replica set, TLS (on by default)
Snowflake	account, user, password, database, schema (optional), warehouse	role
Google BigQuery	project, dataset, service-account key	

The other engines ask for a host, a port, a database, a user and a password. A port is a number between 1 and 65535. **Test before saving** stays disabled until every required field is valid.

Connection Scopes

A connection has a **scope**, which says who can see it:

Scope	Who can use it	Who can change it
Just me	you only. Administrators do not see it.	you
Whole organization	every member of your organization	the administrators of the organization
Deployment	every user of the installation	your operator, in the configuration of the installation. An administrator can take it out of use for the organization.

In the web client, a new connection is **Just me** by default. Only an administrator can create a **Whole organization** connection. The Python client and the API default to the organization scope: a user who is not an administrator must pass `scope="user"`, or the connection is refused.

A deployment connection has no **Edit** button. Only an administrator sees **Delete** next to it, and deleting it only takes it out of use for the organization.

Write-Only Credentials

The password and the other credentials of a connection are sent once, stored encrypted, and never returned: not by the web client, not by the API, and not in a job. A listing shows the host and the database name, enough to tell two connections apart, and nothing you can connect with.

When a job runs, its worker does not get the password. It gets a short-lived token for one connection. A separate service, the **database connector**, presents the

© Octopize — All rights reserved

token, receives the connection string from the API, and opens the database. The rows go from your database into the storage of the job.

Add a Connection

1. Go to **Account > Database connections** and click **New connection**. You can also click **New connection** in the source and target pickers when you create a database run.
2. Choose the **Type of database** and fill in its form.
3. Choose a **Name**, for example `crm_prod`. The web client suggests one from the database field. A name has at most 63 characters: letters, digits, underscores, hyphens and dots, and it starts with a letter or a digit. Jobs and saved configurations refer to a connection by its name.
4. Choose a **Display name** (optional), for example *CRM (production)*. It is only what users see in a picker.
5. Choose **Who can use this connection**: **Just me** or **Whole organization**.
6. Click **Test before saving**, then **Save**.

Test a Connection

Click **Test** next to a connection. Avatar opens it from the server and reports three facts:

- Is the database **reachable**?
- Does it have a **schema** (tables)?
- Is it **empty**?

These are the three facts that a database run checks on a target, so test a target before you start a run. When the test fails, the message says which kind of failure it is: the database is unreachable, the credentials are wrong, the database does not exist, or a privilege is missing. The message never repeats the connection string.

Rename, Relabel, Replace or Delete a Connection

Click **Edit** next to a connection.

- **Relabel**: change the display name at any time. Nothing else changes.
- **Rename**: jobs already submitted are not affected. A saved configuration that uses the old name will not find the connection, and neither will a re-run of an earlier run (**Re-run this group**, **Re-run every group**, **Adjust and rerun**): jobs carry the name of a connection, not the connection itself.
- **Replace**: to change the host or the password, click **Replace details...** and fill in every field again. The name stays. Avatar never sends the stored credentials back, so there is nothing to edit in place.
- **Delete**: Avatar forgets the connection and its credentials. Jobs already submitted keep working, but a saved configuration or a re-run that names the connection will no longer find it. You can delete your own connections and, as an administrator, the connections of your organization. For a deployment connection, an administrator can only take it out of use for the organization.

Demo Databases

To try a database run before you connect your own databases, click **Add demo connections** in **Account > Database connections**. The wizard also offers it in the

© Octopize — All rights reserved

Read from picker, as long as you have no connection yet. The button only appears when your installation offers the demo databases. They are added as **Just me** connections. Two demo databases are available, each with a source and an empty target:

Database	Content
Northwind	a trading company: 13 tables, 600,000 order lines
Chinook	a music store: 11 tables

The demo sources are read only and shared by every user. Each run gets its own empty copy of the demo target, so two users can run at the same time without interfering. A demo target is deleted 24 hours after its last write, and when the database connector restarts. A demo target already has the schema of its demo source, so it only accepts a copy of that source.

Good Practice

- Give the **source** connection a database role that can only **read**.
- Give the **target** connection a role that can write and delete rows. A run does not create tables, so the target role does not need to.
- Never point a target at a production database. A run refuses a target that holds rows unless you ask for it to be emptied first. It refuses a target that is the **same connection** as the source, but it cannot tell that two different connections point at the same database: check that yourself.

For Administrators

Database connections need the database connector. On a self-hosted installation, your operator enables it: see Database connector. On the Octopize SaaS, the connector can only reach databases on a public address.

© Octopize — All rights reserved

Run a Database Copy in the Web Client

This page follows the job wizard of the web client for a database run, step by step. Read Database Runs and Database Anonymization first for the concepts.

> **i Availability**
>
> The **Database** option appears only when your installation runs the database connector. If you see *Database anonymization is not enabled on this deployment*, ask your administrator. The web client marks the option as **Alpha**: the screens and the results can change between releases.

Before You Start

You need two connections: a source to read from and a target to write to. The target must already have the schema of the source: see Requirements for the Target. To try the feature first, click **Add demo connections**. The wizard offers it in the **Read from** picker as long as you have no connection yet.

Step 1: Use Case

Choose the use case, as for any job.

Step 2: Name & Input

Under **What are you anonymizing?**, choose **Database**. Give the job a name, fill in the job context, and click **Create Job**.

Step 3: Connect a Database

1. In **Read from**, choose the source connection.
2. In **Write to**, choose the target connection. The picker only offers connections of the same engine as the source. For a demo source, it offers the demo targets.
3. Click **Test the connections**. Avatar opens both from the server and reports what it found. Do not skip this: the checks on an empty target and on a connection that cannot be opened only run once the connections have been tested.
4. If the target already holds rows and you want to replace them, check **Empty this target before writing**. Every row in the tables this copy writes to is deleted first. Only those tables are touched.

Each picker also has **New connection** and **Manage connections**, so you can add a connection without leaving the wizard.

Next stays disabled while one of these problems remains:

- a source or a target is not chosen;
- the target is the same connection as the source;
- the target is not empty, and **Empty this target before writing** is not checked;
- the same target is used for two sources;
- a connection could not be opened;
- the engine is not reachable from this installation.

Several Databases

To copy several databases in one run, click **Add another database** and choose a second source and target. Do this only when the same individuals are in several

© Octopize — All rights reserved

databases: the run then gives each individual the same fake identity in every copy.

> ⚠️ **Identities Depend on the Set of Databases**
>
> Adding a database to the run changes every generated name in the other copies too.

Step 3b: Confirm the Schema
Avatar reads the source and proposes a treatment for each table. Nothing is copied and nothing is anonymized at this stage. Discovery starts on its own. If it fails, click **Try discovery again**.

For each table, choose **What happens to this table**:

Option	Result in the target
Pseudonymize / anonymize — each column decides which	the table is processed; you choose per variable in step 5
Copy as-is	the rows, unchanged
Purge — keep the table, drop the rows	the table, with no rows
Ignore — leave it out entirely	no table

Use the search box, the **Filter** and **Apply to** to treat many tables at once.

Open a table to see its variables, its keys (`key → table.column`) and what classification proposes for each variable, with the rule or the model that proposed it. The list is read-only: the choices per variable are made in step 5. A key to a table that you ignore or purge is marked as having no join left to preserve.

When the run copies several databases, each database has its own tab.

Read a Subset of the Rows
In an open table, **Extraction parameters > Rows read** limits the number of rows the copy reads. Leave it empty to read every row. The field is offered on tables that are processed or copied.

A table that references another kept table has no **Rows read** field. It shows *Follows the tables it references*: it receives only the rows that point at the rows that were read, so the subset stays consistent.

Warnings to Read
- *The schema says nothing about …, so they will not be in the copy at all.*
- *… is copied as it stands, so … arrive with their real values. A classifier had proposed replacing them.* Check that this is what you want.

> ⚠️ **Empty Tables Stop the Run**
>
> *These tables have no rows, so they cannot be anonymized* is not a warning but an error, shown in red. **Next** stays enabled, but the whole run fails when it

© Octopize — All rights reserved

starts. Set those tables to **Copy as-is**: an empty table is then copied unchanged.

Step 4: Groups & Identities

The page shows the **groups of related tables** built from your plan. Each group is one job. A purged or ignored table is in no group. Ignoring or purging a table can split a group in two: the tables on each side of it are no longer linked, and become two jobs.

When the run copies several databases, you can declare a **reference table** (optional). A reference table says that several tables describe the same individuals, so that each individual keeps one identity everywhere. Click **Declare a reference table** and answer three questions:

1. **What is this reference table called?** For example `rt_person`.
2. **What identifies the person, and what is it called in each database?** Choose, for each database, the table and the variable that holds the identifier.
3. **Which values are replaced?** Add the variables to replace, for example `email`.

You can also list **key values that are not a person**, such as `0` or `SYSTEM`. Click **Show key values** to compare a few real values from each source: the key must hold the same thing in every table. Click **Add this reference table** to confirm.

With one database, the choices per variable of step 5 are enough.

Step 5: Configure Tables

Each group of related tables has its own tab. For each variable, choose:

Button	Result
Anon	synthetic values, drawn from the distribution
Pseudo	fake values of the same kind, applied consistently
Freeze	the original values, unchanged
Drop	the variable is not in the copy

Freeze keeps real values in the copy. Use it only for a variable that identifies no individual: see Privacy Risk of Frozen Variables.

Key variables show a **key** or **foreign key** label and take no treatment: the run re-issues them and keeps every join working.

Avatar advises parameters for each group before the step is ready.

The Reference Table Tab

If you declared a reference table in step 4, it has its own **Reference table** tab. There you choose how each of its values is replaced. It is resolved once, before any group runs: each individual gets one replacement, and that value is what every group writes. The variables it owns are left out of the modelling of every group.

In steps 6 and 7, the tab is read-only: a reference table has no links and no `k` of its own.

© Octopize — All rights reserved

Sequences
The web client does not set sequences yet. Set them from the Python client, or in the YAML configuration of the run. The results page shows what the run did to the sequence of each table.

Step 6: Link Datasets
For each group, check the links between its tables, as for a relational database run. A group with one table needs no link. The step is skipped when the whole run has a single table.

Step 7: Configure Job
For each group, set the anonymization parameters (`k`, `ncp`) and the metrics. This step is skipped when no variable is left to anonymize, and the steps after it are renumbered.

Step 8: Run Job
The page shows the generated configuration (YAML) and an equivalent Python script. Click **Run now** →.

Read the Results
The anonymized data is in the target database. The job page shows:

- **Whole run**: the number of tables and rows written to the target, the source and target of each database, the result of each reference table, and one line per group.
- **One tab per group**: its tasks, the tables asked for and delivered (with their treatment, row count, sequence and what happened to each variable), and the privacy and utility metrics of the group. A table can be marked *refused* or *nothing written*. A group where every variable is a key or is pseudonymized has nothing to measure: the page says so, and the tables are still copied to the target.
- **Reference table** (when you declared one): how many individuals were resolved and from how many rows, how many appear in only one source, the values that disagree between sources, the rows that could not be matched, and the individuals that appeared after the reference table was resolved. Many individuals in only one source usually means that the key does not hold the same values in every table.
- **Reports**: one Technical Assessment and one Impact Assessment per group, because each group is its own avatarization. **Download all reports (.zip)** gets them all.

If a table did not reach the target, the page warns that rows referencing it now point at nothing.

Re-run or Cancel
Re-run this group and **Re-run every group** start the same configuration again. They only write into an empty target: empty the target, or recreate it, first.

Adjust and rerun reloads the whole configuration of the run, including every database of a multi-database run, and opens the wizard at **Configure tables**. Use it to change a choice and run again.

© Octopize — All rights reserved

All three find the source and the target by the name of their connection. If a connection was renamed or deleted since the first run, the re-run cannot find it.

The jobs list shows one entry per database run, with the tasks of every group, and offers **Download YAML configuration**.

Cancel run stops every group, and **Cancel this group** stops one. The tables already written stay in the target, so the target may be partially filled. Empty it before you start the run again.

© Octopize — All rights reserved

Run a Database Copy with the Python Client

This page shows a database run from the Python client, from the connections to the results. Read Database Runs and Database Anonymization first for the concepts.

Database runs need the Python client `octopize.avatar` **1.30.0** or later.

Set Up

```
from avatars import Manager, shared_entity

manager = Manager(api_key=os.environ["AVATAR_API_KEY"])
runner = manager.create_runner("crm-test-estate", seed=20260908)
```

Give the runner a `seed`. A run can be several jobs, one per group of tables, and they must agree about every shared entity.

Copy One Database

Declare the Source and the Target

The source and the target are **names** of connections. You never write a connection string in a run.

```
SOURCE, TARGET = "sample-crm", "sample-crm-test"

runner.add_database(source=SOURCE, target=TARGET)
```

To copy only some tables, pass `tables=["customer", "invoice"]`. Without it, the run copies every table it finds.

The target must already have the schema, and hold no rows: the Python client neither creates tables nor empties a target. See Requirements for the Target. The Python client also has no option to read only a subset of the rows.

Get the Proposed Plan

Schema discovery is a job. It reads the source, proposes a plan, and stops. Nothing is copied and nothing is anonymized.

```
plan = runner.proposed_plan()

print(plan.summary())
# 1 anonymize, 3 pseudonymize; 6 column(s) awaiting review

print(plan.needs_review())
# {'audit_log': ['actor_email'], 'country': ['name'],
#  'customer': ['address', 'email', 'full_name', 'phone']}
```

Read the proposals before you decide:

```
for proposal in plan.proposals():
    print(proposal)

# customer.email: EMAIL (by rules, NOT reviewed)
# customer.full_name: FULL_NAME (by rules, NOT reviewed)
# customer.city: ADDRESS (by rules, NOT reviewed)
```

© Octopize — All rights reserved

A proposal made **by rules** comes from the name of the variable. A proposal made **by model** comes with a confidence: read each one.

The plan also shows the groups of linked tables. Each group is one job, named after the table that the most links touch:

```
print(plan.clustered())
# {'group-audit_log': ['audit_log'], 'group-customer': ['country', 'customer',
'invoice']}
```

A plan that you build by hand, with `SchemaPlan.for_tables()`, records no group: all its tables are in `group-1`.

Review the Plan

Each method records a decision and returns the plan, so you can chain them:

```
plan = (
    plan.confirm_all()                      # agree with the classifier
    .reject("customer", "city")             # disagree: keep the real cities
    .purge("audit_log")                     # keep the table, remove the rows
    .continue_sequence("customer", "id")    # continue numbering after the copied rows
)

plan.is_reviewed()  # True: the job will be accepted
```

Method	What it does
`confirm(table, *columns)`	agrees with the classifier for these columns
`confirm_all()`	agrees with every proposal. Read `proposals()` first.
`reject(table, *columns)`	keeps these variables unchanged (frozen). Recorded as a decision.
`replace(table, column, strategy)`	pseudonymizes a variable with the strategy you choose
`anonymize_columns(table, *columns)`	synthesizes the values from the distribution of the variable
`freeze_columns(table, *columns)`	keeps the values unchanged
`drop_columns(table, *columns)`	removes the variables from the copy
`copy(*tables)`	copies the tables unchanged
`pseudonymize(*tables)`	keeps the rows, replaces the personal values
`anonymize(*tables)`	anonymizes the tables with the Avatar method
`purge(*tables)`	keeps the tables, with no rows
`ignore(*tables)`	leaves the tables out of the copy
`continue_sequence(table, column)`	continues a sequence after the copied rows
`restart_sequence(table, column)`	restarts a sequence from the beginning (for a purged table)
`start_sequence_at(table, column, n)`	starts a sequence at `n`

A misspelled name is refused immediately, with the names that exist:

© Octopize — All rights reserved

```
plan.purge("audit_logs")
# UnknownTableError: this plan has no table 'audit_logs';
#                    it has: audit_log, country, customer, invoice
```

A key variable takes no variable action. Asking is refused with the reason:

```
plan.freeze_columns("customer", "country_code")
# KeyColumnError: customer.country_code is a key, so it cannot be frozen ...
```

`freeze_columns()` and `reject()` keep real values in the copy. Use them only for a variable that identifies no individual: see Privacy Risk of Frozen Variables.

Declare Shared Entities

If several tables describe the same individuals, declare a shared entity so that each individual gets one replacement everywhere:

```
runner.add_database(
    source=SOURCE,
    target=TARGET,
    plan=plan,
    shared_entities=[
        shared_entity(
            "rt_person",
            tables=["customer", "invoice"],
            key="customer_id",
            replace=["email"],
        )
    ],
)
```

The replacement strategy comes from the name of the attribute: `email` gets realistic fake emails. Pass `strategy=` to choose another one.

Set the Parameters and Run

Each table that the run anonymizes needs its anonymization parameters, as in any other run. The plan knows which tables those are:

```
for table in plan.participating():
    runner.set_parameters(table, k=5)

runner.run()
```

Read the Results

The anonymized data is in the target database. There is nothing to download.

```
print(runner.statuses())
# {'group-customer/standard': 'finished', 'group-customer/signal_metrics':
'finished',
#  'group-customer/privacy_metrics': 'finished', 'group-customer/report': 'finished',
#  'group-audit_log/standard': 'finished', ...}

print(runner.cluster_sets())
# {'group-customer': '6acc42e5-...', 'group-audit_log': '28ef22bc-...'}
```

Each group is its own set of jobs, with its own metrics and reports. When the run has several groups, `runner.get_all_results()` does not wait for them. Read the results of each group from its own set:

© Octopize — All rights reserved

```
for group, set_name in runner.cluster_sets().items():
    results = manager.create_runner_from_id(set_name).get_all_results()
```

A run with a single group is not split: `runner.cluster_sets()` is empty, and `runner.get_all_results()` returns its results. When the run declares shared entities and has several groups, `statuses()` also lists `reference-tables/reference`, the job that resolves them. Each job reports what it wrote: the tables, the number of rows, a foreign key that a purge left with no parent, and a table that the plan did not mention.

> ⚠️ **Copy Without a Plan**
>
> `add_database()` without `plan=` is a plain copy: every table, unchanged. Always pass a reviewed plan.

Copy Several Databases Together

Use this when the same individuals are in more than one database, for example a CRM and a billing system. If everything that is shared is in one database, use `add_database()`.

Name each copy, so that a shared entity can say which database a table is in:

```
runner = manager.create_runner("full-test-estate", seed=20260908)

runner.add_databases(
    copies={
        "crm": ("crm_prod", "crm_test"),
        "billing": ("billing_prod", "billing_test"),
    }
)

plans = {
    "crm": runner.proposed_plan("crm").confirm_all().purge("audit_log"),
    "billing": runner.proposed_plan("billing").confirm_all(),
}
```

Then declare the shared entity with `across`. The identifying variable often has a different name in each database, so you name it per database:

```
runner.add_databases(
    copies={
        "crm": ("crm_prod", "crm_test"),
        "billing": ("billing_prod", "billing_test"),
    },
    plans=plans,
    shared_entities=[
        shared_entity(
            "rt_person",
            across={
                "crm": ("customer", "person_id"),
                "billing": ("invoice", "customer_ref"),
            },
            replace=["email"],
        )
    ],
```

© Octopize — All rights reserved

```
)

runner.run()
runner.statuses()
```

In a run of several databases, each group is named after its copy, for example `crm_group-customer` and `billing_group-invoice`.

Before you run, compare the key values on both sides. If one side holds UUIDs and the other holds integers, you chose the wrong variable.

Every plan is checked and every target validated before a row is written. Each database is written by its own jobs, so one database can be written while another fails. A retry of the failed one is safe: it collects the same entities, in the same order, with the same seed, and gives the same replacements.

Manage Connections from Python

The connections that you see in the web client are also available from the Python client:

```
from avatars.models import ConnectionDraft, ConnectionPatch

connections = manager.auth_client.connections

# Which engines can this deployment reach?
for engine in connections.list_engines().engines:
    print(engine.label, engine.available, engine.preview)

# Which connections can I use?
for connection in connections.list_connections().connections:
    print(connection.name, connection.engine, connection.scope.value)

# Add a personal connection. Without scope="user", the API uses the
# organization scope, which only an administrator can create.
connections.put_connection(
    ConnectionDraft(
        engine="postgresql",
        label="Scratch database",
        scope="user",
        fields={
            "host": "db.internal",
            "port": "5432",
            "database": "scratch",
            "username": "avatar",
            "password": os.environ["SCRATCH_DB_PASSWORD"],
        },
    ),
    "scratch",
)

# Test it: reachable, has a schema, is empty
print(connections.test_connection("scratch"))
# reachable=True engine='postgresql' has_schema=True is_empty=False table_count=12

# Rename or relabel it, then delete it
```

© Octopize — All rights reserved

```
connections.patch_connection(ConnectionPatch(label="Scratch (practice)"), "scratch")
connections.delete_connection("scratch")
```

`list_engines()` also describes the form fields of each engine (`engine.fields`). See Database Connections for the rules on scopes and credentials.

The client has four more calls on connections. The request models are in `avatars.models`.

Call	What it does
`test_draft_connection(ConnectionDraft(...))`	tests a connection before you store it
`list_connection_presets()`	lists the demo databases and the other ready-made connections that your installation offers
`add_connection_presets(ConnectionPresetRequest(kinds="demo", scope="user"))`	adds the demo connections, as **Add demo connections** does
`sample_connection(ConnectionSampleRequest(columns={...}, limit=5), name)`	reads a few real values of some variables, to check that a key holds the same thing everywhere

`sample_connection()` reads 1 to 10 tables, 1 to 5 variables per table, and at most 20 values.

Tutorials

The Python client repository has three notebooks on database runs:

- Tutorial 10: a test database
- Tutorial 11: several databases at once
- Tutorial 12: database connections

© Octopize — All rights reserved

Set Your Preferences

You can configure default preferences that will be automatically applied to every anonymization run. These settings help tailor the format and output of the generated documents to your specific needs.

Setting Preferences

Via the Web Client

1. Navigate to **Account** → **User Preferences** in the web interface
2. Configure your default preferences
3. Save your changes

These preferences will apply to all future runs initiated through the web client.

Via the Python Client

Set preferences when creating a runner using `manager.create_runner()`:

```
from avatars.client import Client
from avatars.models import DataRecipient, DataType, DataSubject, SensitivityLevel,
ReportLanguage

client = Client()
manager = client.managers.get("my-manager")

runner = manager.create_runner(
    max_distribution_plots=20,
    pia_data_recipient=DataRecipient.INTERNAL,
    pia_data_type=DataType.HEALTH,
    pia_data_subject=DataSubject.PATIENTS,
    pia_sensitivity_level=SensitivityLevel.HIGH,
    report_language=ReportLanguage.EN
)
```

Available Preferences

max_distribution_plots

Type: `int | None`
Default: `None`

Maximum number of distribution plots to include in the quality report. Set to limit report size when working with datasets containing many columns.

report_language

Type: `ReportLanguage`
Default: `ReportLanguage.EN`

Language for the generated quality report.

Available values: - `ReportLanguage.EN` - English - `ReportLanguage.FR` - French

Privacy Impact Assessment (PIA) Metadata

The following preferences help document the context of your data processing for privacy assessment and compliance purposes. These values are included in the anonymization report.

© Octopize — All rights reserved

pia_sensitivity_level

Type: `SensitivityLevel`
Default: `SensitivityLevel.UNDEFINED`

Evaluation of the sensitivity level of the personal data being processed, based on the nature of the data and potential risks to data subjects.

Available values: - `SensitivityLevel.VERY_HIGH` - Highly sensitive data with severe re-identification risks - `SensitivityLevel.HIGH` - Sensitive personal data (e.g., GDPR Art. 9 special categories) - `SensitivityLevel.MEDIUM` - Personal data with moderate sensitivity - `SensitivityLevel.LOW` - Low sensitivity demographic data - `SensitivityLevel.VERY_LOW` - Minimal sensitivity data - `SensitivityLevel.NEGLIGIBLE` - Non-sensitive data - `SensitivityLevel.UNDEFINED` - Sensitivity not yet assessed

The chosen value will directly influence the content of the automatically generated impact analysis alongside the anonymization, and will help you document the associated risks and mitigation measures for your project.



Sensitive personal data under GDPR Art. 9 (health, racial/ethnic origin, political opinions, religious beliefs, genetic data, biometric data, etc.) typically require `VERY_HIGH` or `HIGH` sensitivity levels.

pia_data_recipient

Type: `DataRecipient`
Default: `DataRecipient.UNKNOWN`

Categories of recipients who will have access to the anonymized data.

Available values: - `DataRecipient.INTERNAL` - Internal stakeholders within the same organization - `DataRecipient.CONTRACTUAL_THIRDPARTY` - Third parties with contractual relationship (research partners, vendors) - `DataRecipient.TRUSTED_THIRDPARTY` - Trusted third parties under confidentiality agreements - `DataRecipient.OPENDATA` - General public through open data repositories - `DataRecipient.OUTSIDE_EU` - Entities outside the European Union - `DataRecipient.UNKNOWN` - Recipients not yet identified

The chosen value will allow generating a targeted paragraph in the privacy impact analysis report.

pia_data_subject

Type: `DataSubject`
Default: `DataSubject.UNKNOWN`

Categories of individuals whose personal data are being processed.

Available values: - `DataSubject.PATIENTS` - Patients in healthcare or clinical research contexts - `DataSubject.EMPLOYEES` - Employees, job applicants, or contractors - `DataSubject.CLIENTS` - Clients, customers, or insured persons - `DataSubject.USERS` - Users of digital, public, or mobility services - `DataSubject.STUDENTS` - Students in educational institutions - `DataSubject.UNKNOWN` - Data subjects not yet categorized

© Octopize — All rights reserved

The chosen value will allow generating a targeted paragraph in the privacy impact analysis report.

pia_data_type

Type: `DataType`
Default: `DataType.UNKNOWN`

Categories of personal data being processed, based on the sector and context.

Available values: - `DataType.HEALTH` - Health-related datasets with clinical and demographic variables - `DataType.HR` - Human resources data (employment history, remuneration, performance) - `DataType.INSURANCE` - Insurance policyholder and claims data - `DataType.FINANCE` - Financial service data (transactions, accounts, credit ratings) - `DataType.MOBILITY` - Transport and mobility service data (geolocation, travel routes) - `DataType.EDUCATION` - Educational data (academic performance, attendance) - `DataType.UNKNOWN` - Data type not yet determined

The chosen value will allow generating a targeted paragraph in the privacy impact analysis report.

Example: Complete Configuration

```
runner = manager.create_runner(
    max_distribution_plots=15,
    pia_data_recipient=DataRecipient.CONTRACTUAL_THIRDPARTY,
    pia_data_type=DataType.HEALTH,
    pia_data_subject=DataSubject.PATIENTS,
    pia_sensitivity_level=SensitivityLevel.VERY_HIGH,
    report_language=ReportLanguage.FR
)
```

This configuration: - Limits distribution plots to 15 - Indicates data shared with contractual research partners - Specifies health data about patients - Marks data as very high sensitivity - Generates reports in French

© Octopize — All rights reserved

License & Limits

Each organization has a license. It decides which features its users can use (the **scopes**), how large a dataset can be, how many **credits** the organization starts with, and how long the license lasts. The same rules apply in SaaS and on self-hosted deployments.

Plans

	Free Trial	Starter	Professional	Expert
Duration	30 days	1 year	1 year	1 year
Initial credits	10	100	250	500
Max rows per dataset	10,000	100,000	2,000,000	Unlimited
Max columns per dataset	100	100	200	Unlimited
Scopes				
Single Table	✓	✓	✓	✓
Report	✓	✓	✓	✓
Multi-Table	✓		✓	✓
Data Augmentation	✓		✓	✓
Time Series	✓			✓
Privacy Assessment	✓			✓
Differential Privacy	✓		✓	✓
Geolocation				✓

The Free Trial is short and has few credits, but opens most scopes so you can evaluate them. See octopize.io for pricing.

Scopes

Scope	Unlocks
Single Table	Standard tabular data anonymization
Multi-Table	Relational data anonymization, with linked tables
Time Series	Temporal data anonymization
Data Augmentation	Generating additional synthetic records, for example to balance classes
Differential Privacy	DP-based synthesis method (AvatarFastDP)
Privacy Assessment	Privacy metrics computation
Report	Technical Assessment and Impact Assessment report generation
Geolocation	Geolocation-specific anonymization and privacy evaluation

Using a feature outside your scopes returns an error that names the scope it requires.

© Octopize — All rights reserved

Credits

Credits are shared by everyone in the organization. Each avatarization costs **1 credit**, whatever the size of the dataset. Privacy metrics, signal metrics and reports are free.

A database run costs **1 credit per table that it anonymizes or pseudonymizes**. Tables that it copies, purges or ignores are free, and so is schema discovery.

When the organization runs out of credits, new avatarizations are refused until credits are added. Contact support@octopize.io to add credits or change your plan.

Limits

- **Dataset size**: each dataset must fit within the row and column limits of the plan.
- **Retention**: datasets and results are deleted automatically after a retention period, set per organization. Download your results to keep them.
- **Expiry**: when the license expires, the organization is disabled within a day, and its users can no longer use the platform. Contact support to renew it.

Checking your license

Your user profile lists the scopes enabled for your account. From the Python client, with an API key:

```
from avatars.client import ApiClient

client = ApiClient(base_url="https://your-avatar-instance.com/api", api_key="your-api-key")
me = client.users.get_me()  # the scopes are part of the user profile
```

On a self-hosted deployment

The license of an organization is the `license` attribute of its group in Authentik, for example `license: expert`. See User management. Users who sign up by themselves get the license set by `AVATAR_AUTHENTIK_BLUEPRINT_SELF_SERVICE_LICENSE`, see the Docker configuration.

© Octopize — All rights reserved

Crash Reporting

When an unhandled exception occurs while using the Avatar Python client, a crash report is automatically generated to help diagnose the issue.

What Happens When an Error Occurs

If your script encounters an unexpected error, you will see:

1. The standard Python traceback (the familiar error message format)
2. A message indicating that a crash report has been saved on your local device only.

Example output:

```
Traceback (most recent call last):
  File "my_script.py", line 42, in <module>
    runner.run()
  File ".../avatars/runner.py", line 667, in run
    ...
ValueError: Some error message

[!] A crash report has been generated at '/path/to/crash_report.txt'.
    Please include this file if you want to be assisted at support@octopize.io.
```

What's in the Crash Report

The crash report (`crash_report.txt`) contains detailed information to help our support team diagnose your issue:

- **System Information**: Your operating system, Python version, and working directory
- **Exception Details**: The full error type and message
- **Stack Trace**: The complete call stack with source code context for your code
- **Your Script**: The full content of your script file
- **Configuration**: The YAML configuration of any Runner instances you created

Privacy and Legal Notice

> ⚠️ **Important Privacy Information**
>
> The crash report includes your script's source code and configuration files. Before sharing a crash report with Octopize support:
>
> - **Review the file contents** to ensure you are comfortable sharing this information
> - **Check for sensitive data** such as API keys, passwords, file paths, or proprietary business logic. The majority should have been removed.
> - **Remove any confidential information** if necessary before sending
>
> Octopize collects this information solely for the purpose of diagnosing and resolving technical issues. We handle all crash reports confidentially and in accordance with our privacy policy and applicable data protection regulations. The crash report is NOT sent automatically, you remain completely in control.

© Octopize — All rights reserved

What is NOT Included

The crash report does **not** contain:

- Your actual data or datasets
- Database credentials or connection strings (unless explicitly written in your script)
- Environment variables (unless explicitly referenced in your script)

Getting Help

If you encounter an issue:

1. Check the error message in your terminal for immediate guidance
2. If you need assistance, email support@octopize.io and attach the `crash_report.txt` file
3. The crash report contains all the information needed to help you quickly

© Octopize — All rights reserved

04

Self-Training Guide

Self-Training Guide

If you've landed here, you're looking to train yourself independently in using the **Avatar anonymization method**. To help you along the way, we've crafted a recommended learning path. Follow it in order for a structured and progressive experience.

📚 Recommended Learning Path

1. 🧭 General Introduction

Start here to get familiar with the Avatar method.

- 🔗 Overview
- ❓ We also recommend keeping the FAQ handy as you progress through your learning.

2. ⚖️ Legal Context & Anonymization Status

Understand how Avatar fits within data protection laws and compliance frameworks.

- 🔗 Legal & Compliance Overview

3. 🛠️ Technical Explanation (Optional)

If you're curious or need to go deeper into the inner workings of the tool.

- 🔗 How the Method Works

4. 💡 Methodology in Action — Simple Use Case

Follow a concrete walkthrough of the anonymization pipeline applied to a basic dataset.

- 🔗 Quickstart Tutorial

5. 📏 Evaluating Anonymized Data Quality

🔐 Privacy Metrics

- **Individualization**: How unique are the anonymized records?
- **Correlation**: Are statistical relationships preserved?
- **Inference**: Could an attacker still guess hidden info?

📊 Utility Metrics

- **Unimodal**: Are single-variable distributions preserved?
- **Bimodal**: Are pairwise relationships maintained?
- **Multimodal**: Are complex patterns between multiple variables retained?
- 🔗 Metrics Explained

© Octopize — All rights reserved

6. 🧠 Theoretical Application

When and why to use Avatar? Gain a conceptual understanding of its use cases.

- 🔗 See Methodology Details

7. 🧪 Practical Application

Apply everything you've learned to your own data.

- 🔗 Step-by-Step User Guide

By completing this journey, you'll have all the foundations needed to confidently leverage the Avatar method for robust, regulation-aligned data anonymization.

© Octopize — All rights reserved

05

Deploying

Deploying

Avatar runs either on Octopize's infrastructure (SaaS) or on yours (self-hosted). Every option gives you the same platform: the web client, the Python client and the API.

	SaaS	Self-hosted with Docker	Self-hosted with Kubernetes
Who runs it	Octopize	You	You
What you bring	A browser and HTTPS access	One Linux server with Docker	A cluster, an ingress controller, a StorageClass, a PostgreSQL server
How it is set up	Octopize creates your account	`octopize-deploy-tool` generates a Docker Compose stack	The Avatar Helm chart and your values file
PostgreSQL	Managed by Octopize	Runs as a container	External: you point the chart at your server
Identity provider	Managed by Octopize	Authentik, set up automatically	Authentik, with a manual first-admin step
Database runs	Enabled, databases on a public address only	Not yet: contact support	With the database connector
Start here	SaaS	Docker installation	Kubernetes installation

Docker is the path all current on-premise deployments use. Pick it unless you specifically want to run Avatar in a Kubernetes cluster.

Both self-hosted paths share the same requirements (TLS certificate, SMTP server, registry credentials, network flows) and the same user management, because both use Authentik as the identity provider.

For how the platform is built and what it sends to Octopize, see the reference pages: Architecture, Security and Data sent to Octopize.

© Octopize — All rights reserved

SaaS

In SaaS (Software as a Service) mode, Octopize runs the platform for you on Scaleway. There is nothing to install.

Before you start

- ☐ Your users can reach the platform over HTTPS (port 443).
- ☐ The `octopize.io` domain is allowed by your email and antispam filters: account setup and password reset links are sent from it.
- ☐ Your users have a recent browser, with JavaScript enabled.

Getting access

There are two ways in. Both lead to the same platform; they differ in the license your organization gets.

Self-service

To try Avatar on your own:

1. Create an account with your work email address, or click **Sign up** on the login page.
2. Confirm it from the email you receive, and choose a password.
3. Log in at www.octopize.app.

A self-service account starts on the **Free Trial** license: 30 days and 10 credits. To keep going after that, contact Octopize to move your organization to a paid plan.

For your organization

When your organization subscribes through Octopize's sales team:

1. Octopize creates your organization, with the license you subscribed to, and its first admin account.
2. The admin receives an email with a link to set their password. The link expires after a while: ask support for a new one if needed.
3. The admin logs in at www.octopize.app and invites the rest of the team.

Using the API

The API is served on the same domain as the web client, under `/api`: `https://www.octopize.app/api`. For scripts, use an API key rather than an email and password.

Roles

- **User**: creates and manages their own datasets and avatarization jobs.
- **Admin**: everything a user can do, plus managing the organization's user accounts.

Getting help

Use the chat in the application or on octopize.io, or email support@octopize.io. For how the platform protects your data, see Security.

Next, follow the User Guide.

© Octopize — All rights reserved

Requirements

This page lists what both self-hosted paths need: hardware, what Octopize gives you, what you prepare, and the network flows to open. What is specific to a path, such as Docker on the server or the cluster components for Kubernetes, is on that path's installation page.

Hardware

The same capacity is needed whether Avatar runs on one Docker server or across the nodes of a Kubernetes cluster:

Resource	Minimum	Recommended
CPU	4 cores, x86_64, 2.30 GHz	More, for faster jobs
Memory	32 GB	64 GB
Disk	100 GB	More for large datasets
Network	100 MB/s	More for large datasets

Provided by Octopize

- **Registry credentials** for `quay.io`, which hosts the private Avatar images. You receive them when the contract is signed.
- **Telemetry credentials** (an S3 bucket name, an access key ID and a secret), if you enable usage telemetry.
- **SMTP credentials**, if you choose to send email through Octopize's mail service (AWS SES) rather than your own server.

Prepared by you

- **A domain name** that points at the server or the load balancer, for example `avatar.company.com`. Every public URL (web client, API, storage, SSO) is derived from it.
- **A TLS certificate for that domain**, with its private key, in PEM format. The Docker installation page explains how to convert a `.pfx` bundle or generate a self-signed certificate for a test deployment.
- **An SMTP server** that Avatar can send email through: host, port, credentials if needed, and the TLS mode (STARTTLS or implicit TLS). Email is required: users receive their account setup and password reset links by email. Raw IP addresses are not accepted as the SMTP host.

Network flows

Inbound

Port	From	Used for
443	User browsers and clients	The web client, the API, single sign-on, and dataset upload and download (S3 endpoint)

© Octopize — All rights reserved

Outbound

Destination	Port	When	Used for
`quay.io`	443	Installation and upgrades	Avatar images (`avatar-service-api`, `avatar-web`), the Helm chart and the deploy tool image
`docker.io`	443	Installation and upgrades	Supporting images (PostgreSQL, Nginx, SeaweedFS)
`ghcr.io`	443	Installation and upgrades	Authentik images
`pypi.org`	443	Installation and upgrades	The `octopize-deploy-tool` package (Docker path only)
Your SMTP server	25, 465 or 587	Always	Account setup, password reset and notification emails
`sentry.io`	443	Optional	Error reporting
`license.octopize.io`	443	Optional	Client compatibility mapping (download only)
The telemetry S3 endpoint	443	Optional	Usage telemetry, when enabled
Your source and target databases	Their own ports (5432, 3306, 1521, 1433, 8123…)	Optional, Kubernetes only	Database runs, from the database connector pod

The installation and upgrade flows can be closed once the deployment is running. The optional flows can be turned off entirely, see Air-gapped deployments.

Container images

The back end is written in Python (`dask`, `polars`, `pyarrow`, `pandas`, `numpy`), and the web client in `Next.js` with `TypeScript`. Sizes as of writing:

© Octopize — All rights reserved

Image	Registry	Size	Notes
`avatar-service-api`	`quay.io`	3.00 GB	Private
`avatar-web`	`quay.io`	0.30 GB	Private
`authentik`	`ghcr.io`	1.00 GB	
`postgres`	`docker.io`	0.40 GB	
`seaweedfs`	`docker.io`	0.20 GB	
`nginx`	`docker.io`	0.20 GB	Docker path only
`avatar-connector`	`quay.io`		Private. Optional, Kubernetes only: the database connector

Plan for about 6 GB of disk space per Avatar version, and prune old images after upgrades.

Air-gapped deployments

Avatar runs without internet access. Before installing:

- ☐ Mirror the container images to a registry the deployment can reach, or load them onto the server.
- ☐ Transfer the installer: the `octopize-deploy-tool` package or its image for Docker, or the Helm chart archive (`helm pull … --untar`) for Kubernetes.
- ☐ Have the TLS certificate available locally.
- ☐ Make sure an internal SMTP server is reachable.

Then turn off the optional outbound flows:

Flow	Docker	Kubernetes
Error reporting (Sentry)	`IS_SENTRY_ENABLED: false`	`api.config.sentry.enabled: false`
Client compatibility mapping	`COMPATIBILITY_MAPPING_URL: "disabled"`	`api.config.compatibilityMappingUrl: "disabled"`
Usage telemetry	Leave the telemetry values empty	Leave the `telemetry.s3` values empty

On Docker, set these in your configuration file. On Kubernetes, see also pointing the chart at a mirror.

For mirroring the Authentik images, see the Authentik air-gapped documentation.

© Octopize — All rights reserved

Installation

The Docker installation runs the whole platform on one Linux server, as a Docker Compose stack. You do not write that stack yourself: `octopize-deploy-tool` asks you a few questions, generates every file (`.env`, `docker-compose.yml`, the Nginx configuration, the secrets and the Authentik setup), and starts it.

Before you start

Go through the requirements shared by both self-hosted paths: hardware, registry credentials, domain name, TLS certificate, SMTP server and network flows. Then check the server itself:

- ☐ Linux. Ubuntu 22.04 LTS or later is recommended.
- ☐ A recent `docker` with the Compose plugin (as of writing, `docker -v` = `29.4.3` and `docker compose version` = `v5.1.3`), and `curl`.
- ☐ The deploying user is in the `docker` group, see Docker's post-installation steps.
- ☐ Python 3.13 or newer to run the deploy tool. Without it, run the tool from its Docker image.

TLS certificate

The deploy tool asks for the path of a PEM certificate (the full chain) and of its unencrypted private key. Give it absolute paths, and restrict access to the key:

```
chmod 600 /etc/ssl/avatar/server.key
chmod 644 /etc/ssl/avatar/fullchain.crt
```

Install

All commands use `./app/avatar` as the deployment directory. Pick any directory, and use the same one for every later command.

1. Log in to the registry

Run the `docker login` command that Octopize sent you. It looks like:

```
docker login -u='octopize+CUSTOMER_NAME' quay.io
```

2. Install the deploy tool

```
pip install octopize-deploy-tool
```

Install it in a virtual environment: most distributions refuse installs into the system Python. You can also run it without installing it, with uv: `uvx octopize-deploy-tool --help`.

3. Generate the configuration

```
octopize-deploy-tool install --app-root ./app/avatar
```

The tool asks for:

- the public domain name, an environment name (for example `mycompany-prod`) and your organization name,
- the paths to the TLS certificate and private key,
- the SMTP server settings, or whether to use Octopize's mail service,
- the email address of the Authentik administrator (`akadmin`),
- whether to enable usage telemetry, and its credentials if you have them.

© Octopize — All rights reserved

It generates every secret itself. Nothing is started yet. To install without prompts, see Installing without prompts.

4. Start the stack

```
octopize-deploy-tool start --app-root ./app/avatar
```

The first start pulls the images, creates the Docker volumes and the databases, and sets up Authentik. It takes a few minutes.

5. Create the first users

```
octopize-deploy-tool tasks create-users --app-root ./app/avatar \
  --admins admin@company.com \
  --users alice@company.com,bob@company.com
```

Each user receives an email with a link to set their password. See User management to add users later.

6. Verify the deployment

```
octopize-deploy-tool tasks provision-admin --app-root ./app/avatar
octopize-deploy-tool verify --app-root ./app/avatar --test-email you@company.com
```

`provision-admin` creates the admin API key that `verify` uses. `verify` checks the API health and sends a test email. Then log in at `https://<your-domain>/` with one of the accounts you created.

If a check fails, see Verification & Troubleshooting.

Where the configuration lives

`octopize-deploy-tool` keeps the configuration in the deployment directory (`./app/avatar` in these pages):

```
./app/avatar/
├── user_config/              # yours: the tool never overwrites it
│   ├── config.example.yaml   # commented reference, written by `install`
│   └── config.yaml           # optional, your values
├── .deployment_states/       # one slot per configuration (v1, v2, …), kept for
rollback
│   └── currently_deployed -> v2
└── generated_files/          # what Docker Compose runs: generated, do not edit
    ├── .env
    ├── docker-compose.yml
    ├── nginx/nginx.conf
    ├── authentik/
    └── .secrets/
```

You set values in the questionnaire or in a YAML file under `user_config/`. The tool renders them into `generated_files/`. **Do not edit `generated_files/` by hand**: every `update` regenerates it, and a manual edit blocks the next update until you resolve the conflict.

Changing a value

Put only the values you want to change in a YAML file, and pass it to `update`. Everything else is carried forward:

```
yaml title="change.yaml" SMTP_HOST: smtp.company.com SMTP_PORT: 587
```

© Octopize — All rights reserved

```
octopize-deploy-tool update --app-root ./app/avatar --from-config change.yaml
octopize-deploy-tool start  --app-root ./app/avatar
```

`update` shows the diff before applying it. `start` recreates the containers that changed.

Installing without prompts

To install without prompts, for example from automation, start from `user_config/config.example.yaml`. This minimal file is enough:

```
yaml title="user_config/config.yaml" PUBLIC_URL: avatar.company.com ENV_NAME:
company-prod ORGANIZATION_NAME: MyCompany NGINX_SSL_CERTIFICATE_PATH: /etc/ssl/
avatar/fullchain.crt NGINX_SSL_CERTIFICATE_KEY_PATH: /etc/ssl/avatar/server.key
```

```
octopize-deploy-tool install --app-root ./app/avatar \
  --from-config ./app/avatar/user_config/config.yaml --non-interactive
```

`ORGANIZATION_NAME` is required, although `config.example.yaml` does not list it.

Running the deploy tool from Docker

If Python 3.13 is not available, or PyPI is blocked while `quay.io` is not, use the tool's image. It needs the Docker socket and the Compose plugin mounted:

```
docker run -it --rm --user "$(id -u):$(id -g)" --group-add "$(stat -c '%g' /var/run/
docker.sock)" \
  -v "$(pwd)/app/avatar:/app-root" -v /var/run/docker.sock:/var/run/docker.sock \
  -v "$(which docker):/usr/local/bin/docker:ro" -v "$CP:$CP:ro" \
  quay.io/octopize/deploy-tool:latest install --app-root /app-root
```

Replace `install` with any other command. At the prompts, type **host** paths (for example the TLS certificate), since Docker Compose reads them on the host. The deploy tool README has a ready-to-copy block for each command.

Next steps

- To change a setting later, see Changing a value. Every variable is listed in Configuration.
- To move to a new Avatar version, see Upgrade & Rollback.
- For an air-gapped server, see Air-gapped deployments.

© Octopize — All rights reserved

Configuration

This page lists the environment variables of a Docker deployment. They end up in `generated_files/.env`. Do not edit that file: set the values through the deploy tool's questionnaire or a configuration file, as described in Changing a value. Values marked *Generated* are derived or created by the tool.

Core

Variable	Default	Description
`PUBLIC_URL`	*Required*	Bare domain name (no `https://`), for example `avatar.company.com`. Every public URL is derived from it.
`ENV_NAME`	*Required*	Identifies the deployment for support. Lowercase letters, numbers and hyphens, for example `company-prod`.
`ORGANIZATION_NAME`	*Required*	Name of your organization.

Service URLs and endpoints

All of these are derived from `PUBLIC_URL`. They are listed so you can recognise them in `.env`.

Variable	Example	Description
`AVATAR_DOMAIN`	`avatar.company.com`	The bare domain, used by Nginx and Authentik.
`AVATAR_API_URL`	`https://avatar.company.com/api`	Where users reach the API.
`AVATAR_WEB_CLIENT_URL`	`https://avatar.company.com`	Where users reach the web client.
`AVATAR_STORAGE_ENDPOINT_PUBLIC_URL`	`https://avatar.company.com/storage`	Public S3 endpoint for uploads and downloads.
`AVATAR_STORAGE_ENDPOINT_INTERNAL_URL`	`http://s3:8333`	S3 endpoint inside the Compose network.
`SSO_PROVIDER_URL`	`https://avatar.company.com/sso`	Public base URL of Authentik.

Variable	Default	Description
`COMPATIBILITY_MAPPING_URL`	`https://license.octopize.io/compatibility/compatibility_mapping.json`	Where the API downloads the client compatibility mapping. Set it to `disabled` to make no outgoing request. An empty value falls back to the default.

© Octopize — All rights reserved

TLS and Nginx

Variable	Default	Description
NGINX_SSL_CERTIFICATE_PATH	./tls/server.fullchain.crt	Host path of the PEM full-chain certificate. Use an absolute path.
NGINX_SSL_CERTIFICATE_KEY_PATH	./tls/private/server.decrypted.key	Host path of the unencrypted private key. Use an absolute path.
NGINX_TLS_ENABLED	true	Set to `false` only for a local evaluation.
NGINX_HTTP_PORT	8080	Port Nginx listens on when TLS is disabled.

Storage

Variable	Default	Description
SHARED_STORAGE_PATH	s3://{user_id}/{expire_days}d	Where datasets and generated files are stored, in the bundled SeaweedFS.

Datasets and generated files are deleted after 30 days by default. The retention period is set per organization: ask Octopize support to change it.

The storage encryption key and the S3 admin keys are generated into `generated_files/.secrets/` and cannot be configured. Back them up with the rest of the deployment directory: without `storage_encryption_key`, the stored files cannot be read.

SeaweedFS admin password

The SeaweedFS storage has an admin password for the user `admin`. It is in `generated_files/.secrets/seaweedfs_admin_password`. The tool generates it on `install`, and on the first `update` of a deployment that does not have one.

To set your own password, or to rotate it, give the new value to `update`. Then restart the `admin` service, because `start` does not restart it for this change:

```
echo "seaweedfs_admin_password: $(openssl rand -hex 32)" > rotate.yaml
octopize-deploy-tool update --app-root ./app/avatar --from-config rotate.yaml
octopize-deploy-tool start --app-root ./app/avatar
docker compose -f ./app/avatar/generated_files/docker-compose.yml restart admin
rm rotate.yaml
```

Database

PostgreSQL runs as the `db` service, with its data in the `avatar_postgres_data` volume. It holds two databases, Avatar's and Authentik's. Their names, roles and passwords are generated into `generated_files/.secrets/`:

© Octopize — All rights reserved

Secret file	Typical value	Description
`db_name`	`avatar`	Avatar database
`db_user` / `db_password`	`avatar`	Role the API connects as
`db_admin_user` / `db_admin_password`	`avatar_dba`	PostgreSQL superuser, used to create the databases and for backups
`authentik_database_name`	`authentik`	Authentik database
`authentik_database_user` / `authentik_database_password`	`authentik`	Role Authentik connects as

Service versions

Each release of the deploy tool pins tested versions. Override them only when Octopize asks you to.

Variable	Description
`AVATAR_API_VERSION`	Avatar API image tag
`AVATAR_WEB_VERSION`	Avatar web client image tag
`AVATAR_SEAWEEDFS_VERSION`	SeaweedFS image tag
`AVATAR_AUTHENTIK_VERSION`	Authentik image tag

Logging and monitoring

Variable	Default	Description
`IS_SENTRY_ENABLED`	`true`	Send internal errors to Sentry. See Data sent to Octopize.
`LOG_LEVEL`	`INFO`	One of `DEBUG`, `INFO`, `WARNING`, `ERROR`, `CRITICAL`.
`USE_CONSOLE_LOGGING`	`false`	Human-readable logs instead of JSON.

Email

Variable	Default	Description
`SMTP_HOST`	*Required*	SMTP server host name. Raw IP addresses are not accepted.
`SMTP_PORT`	`587`	`587` (STARTTLS), `465` (implicit TLS) or `25` (plain).
`SMTP_USE_TLS`	`true`	Connect over TLS.
`SMTP_START_TLS`	`false`	Upgrade a plain connection to TLS, for example for Office 365.
`SMTP_VERIFY`	`true`	Verify the server's certificate. Set to `false` for a self-signed SMTP certificate.
`SMTP_SENDER_EMAIL`	`noreply@octopize.io`	The "from" address. Often the same as `SMTP_USERNAME`.
`SMTP_USERNAME`	*None*	If the server requires authentication.
`SMTP_PASSWORD`	*None*	If the server requires authentication.

Example for an internal relay without TLS:

```
SMTP_HOST: smtp.company.internal
SMTP_PORT: 25
```

© Octopize — All rights reserved

```
SMTP_USE_TLS: false
SMTP_VERIFY: false
```

Telemetry

Telemetry sends usage metadata to Octopize, never datasets. It lets Octopize support you faster and keep track of the versions you run. See why, and what is sent. It is enabled once all the values below are set. Octopize provides the bucket name and the two credentials.

Variable / secret	Default	Description
`TELEMETRY_S3_ENDPOINT_URL`	`https://s3.fr-par.scw.cloud`	Telemetry S3 endpoint
`TELEMETRY_S3_REGION`	`fr-par`	Telemetry S3 region
`TELEMETRY_S3_BUCKET_NAME`	*Provided by Octopize*	Your telemetry bucket
`telemetry_s3_access_key_id`	*Provided by Octopize*	Secret file
`telemetry_s3_secret_access_key`	*Provided by Octopize*	Secret file

The questionnaire offers to enable telemetry on `install` and on every `update` while it is off. If you do not have the credentials yet, skip them, then run `update` once you receive them.

Single sign-on (Authentik)

The tool sets up Authentik on the first `start`, from a bundled blueprint. You normally only choose the administrator's email address. The other values are generated, and listed here so you can recognise them.

Bootstrap

Variable	Description
`AUTHENTIK_BOOTSTRAP_EMAIL`	Email address of the Authentik administrator (`akadmin`). Asked during `install`.
`AUTHENTIK_BOOTSTRAP_PASSWORD`	*Generated.* Password of `akadmin`.
`AUTHENTIK_BOOTSTRAP_TOKEN`	*Generated.* Authentik API token, used by `tasks create-users` and by the API to revoke the keys of disabled users.

© Octopize — All rights reserved

Blueprint variables

Variable	Default	Description
`AVATAR_AUTHENTIK_BLUEPRINT_DOMAIN`	*Generated*	Same as `AVATAR_DOMAIN`.
`AVATAR_AUTHENTIK_BLUEPRINT_CLIENT_ID`	*Generated*	OAuth2 client ID. The API reads the same value as `SSO_CLIENT_ID`.
`AVATAR_AUTHENTIK_BLUEPRINT_CLIENT_SECRET`	*Generated*	OAuth2 client secret. The API reads the same value as `SSO_CLIENT_SECRET`.
`AVATAR_AUTHENTIK_BLUEPRINT_API_REDIRECT_URI`	*Generated*	`https://<AVATAR_DOMAIN>/api/login/sso/auth`.
`AVATAR_AUTHENTIK_BLUEPRINT_API_WEBHOOK_URL`	*Generated*	Where Authentik notifies the API that a user was disabled or deleted, so their API keys are revoked.
`AVATAR_AUTHENTIK_BLUEPRINT_SELF_SERVICE_LICENSE`	`expert`	License given to users who sign up themselves: `expert` (all features) or `free_trial` (limited runs and features).

`SSO_PROVIDER_APP_NAME` is `avatar-api` and must not change.

© Octopize — All rights reserved

Upgrade & Rollback

Each release of `octopize-deploy-tool` bundles the templates and the image versions of one Avatar release. Upgrading Avatar therefore means upgrading the tool, then letting it regenerate and restart the stack.

> **i Deployment set up by hand, or before Authentik?**
>
> If your deployment directory has `.env` and `.secrets/` at its top level and no `.deployment_states/`, run the one-time Legacy Migration first.

Before you upgrade

- ☐ Read the release notes of the target version.
- ☐ Plan a maintenance window: the stack restarts, which usually takes 5 to 15 minutes, plus the time for database migrations.
- ☐ Check that you have about 6 GB of free disk space for the new images.
- ☐ Back up the databases and the deployment directory.

Backup

Back up two things: **both databases** (Avatar's and Authentik's) and **the whole deployment directory**, which holds the secrets, including the storage encryption key. The backup files are written next to the deployment directory:

```
cd ./app/avatar/generated_files
STAMP=$(date +%Y%m%d-%H%M%S)
DB_ADMIN=$(cat .secrets/db_admin_user)

# Both databases, as SQL that drops and recreates their objects on restore
for DB in $(cat .secrets/db_name) $(cat .secrets/authentik_database_name); do
  docker compose exec -T db pg_dump -U "$DB_ADMIN" --clean --if-exists -d "$DB" \
    > "../../avatar-backup-$STAMP-$DB.sql"
done

# The deployment directory: configuration, every rendered version, and the secrets
tar czf "../../avatar-backup-$STAMP-app.tar.gz" -C ../.. avatar
```

Store these files somewhere safe: they contain every credential of the deployment.

Datasets in SeaweedFS are temporary and expire on their own, so they are not part of the backup. Users download their results.

For production, schedule the database backup, keep at least 30 days of backups, and test a restore regularly.

Upgrade

1. Upgrade the deploy tool

```
pip install --upgrade octopize-deploy-tool
```

If you run the tool from Docker, pull `quay.io/octopize/deploy-tool:latest` instead.

2. Regenerate the configuration

```
octopize-deploy-tool update --app-root ./app/avatar
```

© Octopize — All rights reserved

Your values are carried forward. The tool renders the new templates into a new version slot, shows the diff, starting with the service versions that change, and applies it. It does not restart anything yet.

The deploy tool releases after 3.5.1 need the new secret `seaweedfs_admin_password`. `update` generates it. To give your own value, see SeaweedFS admin password.

`update` ends by listing which Python client versions work with the new API. Tell your users before they next run a job if their client needs upgrading.

3. Restart the stack

```
octopize-deploy-tool start --app-root ./app/avatar
```

This pulls the new images, recreates the containers and re-applies the Authentik blueprint. The API runs the database migrations as it starts.

4. Verify

```
octopize-deploy-tool verify --app-root ./app/avatar --test-email you@company.com
```

Then log in through the web client and run a small avatarization. See Verification & Troubleshooting if anything fails.

Authentik moves one release line per upgrade

Authentik upgrades its database one release line at a time (2026.2, then 2026.5, then 2026.8), and from 2026.8 it refuses to start on a database that skipped one. From deploy tool 3.6.0, `update` moves Authentik at most one release line from the version that your Authentik database has reached. A new installation gets the latest Authentik directly.

A deployment on an Authentik older than 2026.5 therefore needs one upgrade for each release line. For example, from 2026.2:

1. The first `update` gives Authentik 2026.5.7. Run `start` and `verify` as above. They record that Authentik runs 2026.5.7.
2. Run the upgrade again. This `update` gives Authentik 2026.8.

The diff that `update` shows lists the Authentik version that it chose. `verify` also checks that Authentik runs the version of the deployment, and fails if it does not.

> ⚠️ **Do not skip a release line**
>
> If you set `AVATAR_AUTHENTIK_VERSION` in your configuration, keep it at most one release line ahead of the version that Authentik runs. Otherwise Authentik stops at start with `Major version skips are not allowed`. `update` warns you and names the version to use. Authentik cannot downgrade its database: back it up before each upgrade.

Rollback

Roll back if the API does not become healthy, a migration fails, or login, avatarization or storage is broken and cannot be fixed within the maintenance window. First collect diagnostics, so the issue can be investigated afterwards.

© Octopize — All rights reserved

Database migrations are not reversible: a database migrated forward does not work with the previous API version. A rollback therefore restores the databases from the backup.

```
# 1. Stop the stack
octopize-deploy-tool stop --app-root ./app/avatar

# 2. Put the previous rendered version back in place (list the slots with: ls ./app/
avatar/.deployment_states)
octopize-deploy-tool apply --app-root ./app/avatar --version v1

# 3. Restore both databases
cd ./app/avatar/generated_files
docker compose up -d db
DB_ADMIN=$(cat .secrets/db_admin_user)
for DB in $(cat .secrets/db_name) $(cat .secrets/authentik_database_name); do
  docker compose exec -T db psql -U "$DB_ADMIN" -d "$DB" < "../../avatar-backup-
<STAMP>-$DB.sql"
done
cd -

# 4. Start the previous version
octopize-deploy-tool start --app-root ./app/avatar
```

Replace `v1` with the slot that was deployed before the upgrade, and `<STAMP>` with the timestamp of your backup. Then verify the deployment.

If the deployment directory itself is damaged, restore it from the `tar.gz` archive first.

© Octopize — All rights reserved

Verification & Troubleshooting

Use these checks after an installation or an upgrade. `docker compose` commands run from the `generated_files/` directory:

```
cd ./app/avatar/generated_files
```

Quick verification

```
octopize-deploy-tool tasks provision-admin --app-root ./app/avatar
octopize-deploy-tool verify --app-root ./app/avatar --test-email you@company.com
```

`provision-admin` creates an admin API key for the `akadmin` account, in `generated_files/.secrets/avatar_api_key`. It only needs to run once. `verify` then checks the API with that key and sends a test email through Authentik.

Manual checks

Containers

```
docker compose ps
```

Every service should be `running` or `healthy`, none restarting. The services are `api`, `web`, `nginx`, `db`, `dask-scheduler`, `dask-worker`, `master`, `volume`, `filer`, `s3`, `authentik_server` and `authentik_worker`. `init-db`, `init-storage` and `authentik_init_db` exit once they have finished.

API health

```
curl -k https://<your-domain>/api/health
```

This returns `{"message":"ok"}` and needs no authentication. For a full check of the configuration, call `/health/config-full-check` with the admin API key:

```
curl -s -k https://<your-domain>/api/health/config-full-check \
  -H "Authorization: api-key-v1 $(cat .secrets/avatar_api_key)" | python3 -m
json.tool
```

It checks the environment and secrets, the database and pending migrations, storage access, email, the web login page, the workers and telemetry. It returns `200` when everything passes, `207` with warnings and `500` when a check fails. The body details each check.

Email

Emails are sent by Authentik:

```
docker compose exec authentik_worker ak test_email you@company.com
```

Storage encryption

Check that SeaweedFS can read and write the encrypted buckets:

```
docker compose logs -t s3 --tail=100
```

SSE-S3 errors mean that `.secrets/storage_encryption_key` does not match the key the filer holds. Compare them:

```
docker compose exec master weed shell
> fs.cat /etc/s3/sse_kek
```

© Octopize — All rights reserved

If they differ, put the filer's value back in your configuration and restart the stack. Never keep a new key: every stored object would become unreadable. See the SeaweedFS documentation.

Login

1. Open `https://<your-domain>/`.
2. Log in with an account created by `tasks create-users`, or as `akadmin` with the `AUTHENTIK_BOOTSTRAP_EMAIL` and `AUTHENTIK_BOOTSTRAP_PASSWORD` from `.env`.
3. Check that you land back in the Avatar interface and that it shows the expected version.

Sample avatarization

Upload a small dataset, run an avatarization, and download the result. This is the only check that exercises the workers and the storage end to end.

© Octopize — All rights reserved

If a check fails

Symptom	Fix
`verify` reports "API key not found"	Run `tasks provision-admin` first.
Containers start but cannot reach each other	Docker uses an HTTP proxy. Add the `noProxy` setting, then recreate the containers.
`install` stops: "Volume 'avatar_postgres_data' already exists"	The server holds data from an earlier deployment. Use `update` on that deployment's directory to keep it, or `uninstall` it to discard it.
`install` or `update` says the directory is "pre-versioned"	Run the Legacy Migration.
`update` stops: "Apply is blocked until conflicts are resolved"	Someone edited `generated_files/`. Move those edits into a configuration file, then accept the new render.
Pulling images fails (`unauthorized`, `denied`)	Run the `docker login quay.io` command Octopize sent you again.
The API never becomes healthy after an upgrade	Read `docker compose logs api`: a failed database migration is reported there. See Rollback.
Writes to storage fail, SSE-S3 errors in the `s3` logs	The storage encryption key changed. See Storage encryption.
No email arrives	Check the SMTP settings and that the server can reach your SMTP server, then rerun `ak test_email`.
SSO changes do not take effect after `update`	Run `octopize-deploy-tool tasks reconcile-blueprint --app-root ./app/avatar`.
You are locked out of Authentik	See Recovering admin access.
The disk is full	Remove unused images with `docker image prune -a`, or move Docker's storage.

Add `--verbosity 2` to any deploy tool command for detailed output.

🚨 Starting over

Use `octopize-deploy-tool uninstall`, which asks for confirmation and removes the containers, the volumes and the deployment files together. It deletes every dataset and user account. Do not run `docker compose down --volumes` by hand: it deletes the database but leaves the deployment files behind.

© Octopize — All rights reserved

Collecting diagnostic information

Before contacting Octopize support, collect:

```
docker compose ps > diagnostics-ps.txt
docker compose logs --tail=20000 > diagnostics-logs.txt 2>&1
cp .env diagnostics-env.txt
```

> ⚠️
>
> Remove passwords, tokens and secrets from these files before sharing them.

Send them to support@octopize.io with a description of the issue.

© Octopize — All rights reserved

Legacy Migration

> **i When is this needed?**
>
> Only for a deployment whose directory has `.env` and `.secrets/` at its top level and no `.deployment_states/`: one set up by hand, or by a deploy tool older than 2.7.0. Such deployments also predate Authentik and keep their users in the Avatar database.
>
> Every other deployment follows Upgrade & Rollback.

The migration runs once. It moves the configuration into the layout managed by `octopize-deploy-tool`, then imports the existing users, organizations and licenses into Authentik. Afterwards, the deployment upgrades like any other.

1. Back up the legacy deployment
Back up the database and copy the whole legacy directory. The backup commands apply, with the paths of the legacy directory.

2. Migrate the configuration
Install the latest deploy tool, then point it at the legacy directory and at a new, empty deployment directory:

```
octopize-deploy-tool migrate \
  --app-root ./app/avatar-new \
  --from-legacy ./app/avatar-old
```

`migrate` reads the values from the legacy `.env` and `.secrets/` and asks for the missing ones. It then shows the diff, starting with the service versions that change. Expect a large diff: the layout changes and every service jumps to the current release.

> **💡 Non-interactive runs**
>
> Before running with `--non-interactive`, check that the legacy `.env` sets `NGINX_SSL_CERTIFICATE_PATH` and `NGINX_SSL_CERTIFICATE_KEY_PATH` to paths that exist.

3. Start and import the users

```
octopize-deploy-tool start --app-root ./app/avatar-new
```

The first `start` after a `migrate` offers to import the users into Authentik. It previews every operation and asks for confirmation. Accept it.

If you declined, or the import failed partway, run it later. It is safe to re-run, since users already imported are skipped:

```
# Preview without changing anything
octopize-deploy-tool tasks authentik-migrate --from-docker --app-root ./app/avatar-new --dry-run

# Import
octopize-deploy-tool tasks authentik-migrate --from-docker --app-root ./app/avatar-new
```

© Octopize — All rights reserved

```
# Retry only the operations that failed
octopize-deploy-tool tasks authentik-migrate --from-docker --app-root ./app/avatar-
new \
  --from-log migration_log.jsonl --failed-only
```

4. Verify

Follow Verification & Troubleshooting. Existing users log in through Authentik from now on. To add new ones, see User management.

If the migration fails, stop the new deployment, restore the database backup, and bring the legacy deployment back up from its directory.

© Octopize — All rights reserved

Installation

The Avatar platform is packaged as a Helm chart, so it can be deployed into a Kubernetes cluster you already operate. All the architecture components — the API, the web client, the Dask scheduler and workers, the object store and the Authentik identity provider — are installed by that single chart.

Follow the steps below to deploy the Avatar platform into your cluster.

> **i What the chart does and does not do**
>
> The chart deliberately does not create cluster infrastructure. It expects an ingress controller, a StorageClass, a PostgreSQL server and pull credentials to exist already, and it creates everything that runs on top of them — including the Avatar and Authentik databases. The prerequisites below list the split in full.

Cluster prerequisites

Go through the requirements shared by both self-hosted paths, then check that everything in this list exists before you install.

- Kubernetes cluster
  - ☐ An ingress-nginx controller, with a LoadBalancer (or NodePort, or MetalLB) in front of it
  - ☐ A default `StorageClass` — the object store runs as StatefulSets with PersistentVolumeClaims
  - ☐ A CNI that enforces `NetworkPolicy` (Calico, Cilium, …), if you want network isolation
  - ☐ `kubectl` and `helm` (v3) configured against the cluster
- Network and TLS configuration
  - ☐ A domain name pointing at the load balancer. Every public URL the chart hands the API, the web client and Authentik is derived from this one name.
  - ☐ A `kubernetes.io/tls` Secret holding the certificate for that domain, in the release namespace. The chart never creates it.
  - ☐ If that certificate was issued by a private or internal Certificate Authority, the CA certificate itself — it has to be passed to the API as `api.caBundle`, or logins fail
  - ☐ An SMTP server reachable from the cluster, for password resets and invitations
  - ☐ Octopize has provided you with your quay.io credentials
- Database
  - ☐ A PostgreSQL server reachable from the cluster
  - ☐ An administrative account on it. Both the Avatar database and Authentik's are created *for* you, using that account.
- Network
  - ☐ The flows listed in the shared Requirements are open. The Helm chart artifact is pulled from `quay.io` as well.

If you run an ingress controller other than ingress-nginx, set `ingress.enabled: false` and create your own Ingress objects. The URLs stay the same either way.

© Octopize — All rights reserved

Container images

The images and their sizes are listed in Requirements.

The `avatar-service-api` and `avatar-web` images are private, and so is `avatar-connector` if you enable the database connector. Octopize provides you with the `quay.io` credentials upon contract signature; the chart takes them as a base64-encoded Docker `config.json` in `imageCredentials.dockerConfigJson`, and creates the `docker-local-pull-secret` Secret from it:

```
base64 -w0 < ~/.docker/config.json
```

Prepare a values file

The chart ships a complete, working example values file. Pull the chart and copy it:

```
helm pull oci://quay.io/octopize/helm/services-api/avatar --version <version> --untar
cp avatar/example-values.yaml my-values.yaml
```

Every `PLACEHOLDER_` in that file has to be replaced, and each one carries the command that generates it.

The archive also contains a script that makes the credentials for you. Run it, and give the file that it writes to Helm after your own values file:

```
cd avatar && bash scripts/generate-secrets.sh
```

Use `bash`, and not `./`. Helm packages each file with mode 0644, so the copy in the archive is not executable. Helm obeys the last `--values` file, so `avatar-secrets.yaml` replaces these keys in `my-values.yaml`. Keep the keys out of `my-values.yaml`, give the generated file to each upgrade, and keep it. See Generating the secrets.

The script does not make the values that belong to your own infrastructure. You still set these yourself. At a minimum, set:

Value	What it is
`ingress.host`	The domain the platform is served on
`ingress.tls.secretName`	An existing `kubernetes.io/tls` Secret for that domain
`imageCredentials.dockerConfigJson`	base64 of a Docker `config.json` with `quay.io/octopize` credentials
`api.image.tag`	The API image tag
`web.image.tag`	The web client image tag
`database.host` / `database.port`	Your PostgreSQL server
`database.adminUsername` / `database.adminPassword`	Admin credentials, used once to create the databases
`api.config.organizationName`	Your organization's name
`smtp.host` and the other SMTP values	Your SMTP server

See Configuration for the full values reference.

© Octopize — All rights reserved

For a namespace enforcing the `restricted` Pod Security Standard, apply the `example-values-restricted.yaml` overlay on top of your own file with a second `--values` — see Restricted Pod Security Standard.

> ⚠️ **Two values have to agree with something else**
>
> - `dask.scheduler.image.tag` and `dask.worker.image.tag` must equal `api.image.tag`.
> - `storage.sseKek` is generated once and never changed. When migrating an existing SeaweedFS, copy the value out of the old filer — see SeaweedFS encryption. Write it into `avatar-secrets.yaml` if you used the script, because Helm obeys that file last.

You set the four `sso` values one time. The API authenticates with `clientId` and `clientSecret`, and the bundled Authentik blueprint registers the same OAuth provider. `apiToken` and `webhookSecret` revoke the API keys of a user that you disable or delete in Authentik. Helm cannot write into the values of a subchart, so the `AVATAR_AUTHENTIK_BLUEPRINT_*` entries under `authentik.global.env` refer to the objects that the chart writes these four values into. Leave those entries as `example-values.yaml` has them. See SSO.

> ⚠️ **A certificate from a private CA needs `api.caBundle`**
>
> If your domain's certificate was issued by an internal or self-signed authority, every login fails with `CERTIFICATE_VERIFY_FAILED` until you set `api.caBundle`.

Install

```
helm install avatar oci://quay.io/octopize/helm/services-api/avatar \
  --version <version> \
  --namespace avatar --create-namespace \
  --values my-values.yaml
```

A missing required value fails the render with a message naming it, so an install that reports success has at least been given a complete configuration. The chart then prints the routing table for the deployment.

Watch it come up

```
kubectl -n avatar get pods -w
```

The Authentik pods sit in `Init:0/1` until their database answers. That is the `wait-for-db` init container doing its job, not a hang — read the bootstrap Job's logs to see where it is:

```
kubectl -n avatar get jobs
kubectl -n avatar logs job/avatar-authentik-db-init-<chart-version>
```

The Job is cleaned up after it finishes (`authentikDatabase.ttlSecondsAfterFinished`), so read its logs while it is still there.

© Octopize — All rights reserved

Create the first Authentik admin

> 🚨 **Required, and not automated**
>
> An install where nobody has been through this flow has an identity provider nobody can administer.

Unlike the Docker installation, the chart does not set the `AUTHENTIK_BOOTSTRAP_*` variables — Octopize does not hold your identity provider's admin credential. Go through Authentik's own flow instead:

`https://<ingress.host>/sso/if/flow/initial-setup/`

Set the `akadmin` password there. That account then administers users, groups and the OAuth provider the Avatar API talks to — see User management.

Do it promptly: until you do, the flow is reachable by anyone who can reach the deployment. It stops working once the first admin exists.

Log in

Browse to `https://<ingress.host>/web`. You are sent to Authentik to log in, and land back in the Avatar interface.

Next steps

Once the stack is running, confirm everything is healthy by following the Verification & Troubleshooting guide.

To let your users copy and anonymize databases with database runs, enable the database connector. It is off by default.

Trying it out on a local cluster

For an evaluation, the same values file installs on a single-node cluster — minikube, kind, k3s — with a handful of changes. There is nothing Avatar-specific about them; they are what a laptop cannot provide:

- Enable an ingress controller and a default StorageClass. On minikube: `minikube addons enable ingress default-storageclass storage-provisioner`.
- Point `ingress.host` at a name that resolves to the cluster — `<cluster-ip>.nip.io` works without touching DNS.
- Set `ingress.tls.enabled: false` and leave `ingress.tls.secretName` empty. Every URL then uses `http://` — fine for an evaluation, never for production.
- Run a PostgreSQL in the cluster and point `database.host` at its Service.
- Point the SMTP values at a local mail catcher. Mail has to work even locally, or user creation fails.
- Shrink `seaweedfs.volume.dataDirs[0].size` and `dask.worker.replicas`, and match the `-allowedOrigins` arguments to your `http://` host.
- Clear the SeaweedFS pod anti-affinity (`seaweedfs.master.affinity`, `.volume.affinity`, `.filer.affinity` set to `""`), which a single-node cluster cannot satisfy.

Everything else is unchanged from the example values file. The first-admin flow is still required.

© Octopize — All rights reserved

Air-gapped deployments

First go through the air-gapped checklist shared by both paths: mirrored images, the chart archive (`helm pull … --untar`, then install from the local directory), a local certificate and an internal SMTP server.

Then point the images at your mirror with `global.imageRegistry`:

```
global:
  imageRegistry: registry.internal.example
```

It covers the API, the web client and SeaweedFS. Four more take their registry as part of a larger string and have to be pointed at the mirror by hand: `dask.scheduler.image.repository`, `dask.worker.image.repository`, `authentik.global.image.repository` and `authentikDatabase.image`.

Then turn off the two outbound calls the platform would otherwise make:

```
api:
  config:
    compatibilityMappingUrl: "disabled"
    sentry:
      enabled: false
```

For detailed information about mirroring the Authentik images, refer to the Authentik air-gapped documentation.

© Octopize — All rights reserved

Configuration

The chart is configured from a Helm values file. `example-values.yaml`, shipped inside the chart, is a complete working example — copy it and replace the placeholders, as described in Installation.

Where a value matches an environment variable on the Docker deployment, this page links to the Docker configuration reference rather than repeating it.

`_Required_` means the platform will not work until you set it.

Global and common values

Value	Default value	Description
`envName`	*None*	Environment name, shown for support purposes. Same as `ENV_NAME`.
`imageCredentials.dockerConfigJson`	*Required*	base64 of a Docker `config.json` with `quay.io/octopize` credentials. Octopize provides them.
`global.imageRegistry`	*None*	Pull every image the chart owns from this registry instead. Use it for a mirrored install.
`global.imagePullSecrets`	`[]`	Existing pull Secrets to add to every pod. Entries are maps: `- name: my-registry-creds`.
`nameOverride`	*None*	Value of the `app.kubernetes.io/name` label on every object.
`commonLabels`	{}	Extra labels on every object the chart creates.
`commonAnnotations`	{}	Extra annotations on every object the chart creates.

Generating the secrets

The chart archive contains a script that makes these values for you. Run it in the directory that `helm pull --untar` wrote:

```
bash scripts/generate-secrets.sh
```

Use `bash`, and not `./`. Helm packages each file with mode 0644, so the copy in the archive is not executable.

The script writes `avatar-secrets.yaml`. Give that file to Helm after your own values file, and give it to each upgrade:

```
helm install avatar . --values my-values.yaml --values avatar-secrets.yaml
```

Helm obeys the last `--values` file, so the generated file replaces these keys in your own file. Keep the keys out of your own file. If you already have one of these credentials, write it into the generated file. A `storage.sseKek` that protects stored objects is the usual example.

Keep the generated file. A second run of the script writes a new file. It does not replace a file that exists, because `storage.sseKek` cannot change after the first install.

The script does not make the values that belong to your own infrastructure. These are the PostgreSQL admin credentials, the registry pull secret, the SMTP relay, the

© Octopize — All rights reserved

host names, and the credentials of an external S3 endpoint. The script lists them when it stops.

You can also make each value yourself. Each secret that you must invent is 32 random bytes. We write them in hexadecimal, which gives 64 characters, but the chart does not require that. Only `storage.sseKek` and `databaseConnections.encryptionKeys` have a constraint, and the other values accept any text that is not empty.

```
python -c "import secrets; print(secrets.token_hex(nbytes=32), end='')"
openssl rand -hex 32
```

© Octopize — All rights reserved

Value	Constraint	If you change it later
`api.auth.jwtSecretKey`	None	Each user must log in again. No stored data is lost
`database.password`	Any text that PostgreSQL accepts as a password	Change it on the server first. See Rotating a database password
`storage.admin.accessKeyId`	None	Safe with the in-chart SeaweedFS, which the chart writes from this value. With your own `existingConfigSecret`, change both sides at the same time
`storage.admin.secretAccessKey`	None	As above
`storage.sseKek`	**64 lowercase hexadecimal characters.** The chart checks this only when it writes the Secret itself	**Never change it.** Each stored object becomes unreadable
`storage.seaweedfsAdminPassword`	None	Restart the SeaweedFS admin server: `kubectl rollout restart statefulset/seaweedfs-admin`
`sso.clientId`	None. It is not a secret, but you make it in the same way	Restart the Authentik server and worker, then apply the blueprint again
`sso.clientSecret`	None	Restart the Authentik server and worker, then apply the blueprint again
`sso.apiToken`	None	Restart the Authentik server and worker, then apply the blueprint again
`sso.webhookSecret`	None	Restart the Authentik server and worker, then apply the blueprint again
`authentik.authentik.secret_key`	None.	Each user must log in to Authentik again
`authentik.authentik.postgresql.password`	Any text that PostgreSQL accepts as a password	Change it on the server first. See Rotating a database password
`connector.secret`	None. Only read when `connector.enabled` is `true`	The tokens of running database jobs become invalid. See Database connector
`databaseConnections.encryptionKeys`	A Fernet key, not a hex string. See Database connector	Put the new key first and keep the old one. See Database connector

`database.adminPassword` is not in the list, because where it comes from depends on your server. If you create the admin account for Avatar, make its password like the others. If you point the chart at an account that already exists, such as a managed-PostgreSQL admin or a role that a DBA provisioned, use the password that it already has. The chart never creates this role and never changes its password. To rotate it, change it on the server, then update the value.

© Octopize — All rights reserved

You set the four SSO values one time, under `sso`. Helm cannot write into the values of a subchart, so the `AVATAR_AUTHENTIK_BLUEPRINT_*` entries under `authentik.global.env` refer to the objects that the chart writes these values into. The blueprint thus reads back the value that you set. The chart compares the two references and refuses to render if they do not agree. To keep these values out of the file, see Using existing Secrets.

Using existing Secrets

By default you write every credential in your values file. The chart can read each one from a Secret instead. Create that Secret in the namespace before the install, with external-secrets, sealed-secrets, Vault or `kubectl create secret`. The chart only reads it, so your values file holds no credential and you can keep it in version control.

`example-values-existing-secrets.yaml`, shipped in the chart, is a complete install with no credential in a values file. Add it after your own:

```
helm install avatar oci://quay.io/octopize/helm/services-api/avatar \
  --version <version> --namespace avatar --create-namespace \
  --values my-values.yaml \
  --values example-values-existing-secrets.yaml
```

Each credential takes one of four routes: the seven blocks the chart owns, image pull, Authentik's own settings, or SeaweedFS. The last three exist because Helm cannot write into the values of a subchart.

The blocks the chart owns

Seven blocks hold credentials: `database`, `api.auth`, `storage`, `sso`, `smtp`, `telemetry.s3` and `authentikDatabase`. Each takes the plain values, `existingSecret` to name the Secret, and `secretKeys` to map each credential to a key in it.

```
database:
  password: "" # empty: the Secret below supplies it
  adminPassword: ""
  existingSecret: avatar-secrets
  secretKeys:
    password: db-password # the key in that Secret
    adminPassword: db-admin-password
```

One Secret can serve every block, and each block can also have its own. Without `secretKeys`, each block looks for these key names:

© Octopize — All rights reserved

Block	Credential	Key it looks for
`database`	`password`, `adminPassword`	`db-password`, `db-admin-password`
`api.auth`	`jwtSecretKey`	`authjwt-secret-key`
`storage`	`admin.accessKeyId`, `admin.secretAccessKey`	`storage-access-key-id`, `storage-secret-access-key`
`sso`	`clientSecret`, `apiToken`, `webhookSecret`	`sso-client-secret`, `sso-api-token`, `sso-webhook-secret`
`smtp`	`password`	`smtp-password`
`telemetry.s3`	`accessKeyId`, `secretAccessKey`	`telemetry-s3-access-key-id`, `telemetry-s3-secret-access-key`
`authentikDatabase`	Authentik's database password	`authentik-db-password`

The database connector has two more blocks: `connector` (key `connector-secret`) and `databaseConnections` (keys `database-connections` and `connection-encryption-keys`). They take `existingSecret` and `secretKeys` in the same way.

Four rules apply:

- Each key that `secretKeys` names must exist in the Secret. If a key is absent, the pods that mount it do not start.
- To keep one credential in the values file, make its `secretKeys` entry empty.
- If you set a plain value and `existingSecret` together, the Secret wins and the plain value is dropped. The chart gives no warning, so empty the values that the Secret supplies.
- A new value in a Secret does not restart the pods. Do `kubectl rollout restart` on the API and the Authentik server and worker. For a database password, change it on the server first. See Rotating a database password.

To set `database.existingSecret` or `authentikDatabase.existingSecret` on a release that already exists, also increase the chart version. See Verification & Troubleshooting.

Image pull credentials

Set `global.imagePullSecrets` to pull Secrets that you created, and empty `imageCredentials.dockerConfigJson`. It reaches every pod of the chart and the Authentik and SeaweedFS pods, but not Dask, which names its pull Secrets itself. Each entry must be a map, not a bare name. The chart fails the render on a bare name only while Authentik is enabled.

```yaml
imageCredentials:
  dockerConfigJson: ""
global:
  imagePullSecrets:
    - name: my-registry-creds
dask:
  scheduler:
    image:
      pullSecrets:
        - name: my-registry-creds
  worker:
    image:
```

© Octopize — All rights reserved

```
    pullSecrets:
      - name: my-registry-creds
```

Authentik

Authentik reads `AUTHENTIK_SECRET_KEY`, `AUTHENTIK_POSTGRESQL__PASSWORD` and `AUTHENTIK_EMAIL__PASSWORD` from its environment. Add an `authentik.global.env` entry with a `secretKeyRef` for each one, and empty the related `authentik.authentik.*` setting. The subchart then keeps that setting out of its own Secret, which frees the variable name.

```
authentik:
  global:
    env:
      # Helm replaces this list. Write again each entry your install needs.
      - name: AUTHENTIK_SECRET_KEY
        valueFrom:
          secretKeyRef:
            name: avatar-secrets
            key: authentik-secret-key
      - name: AUTHENTIK_POSTGRESQL__PASSWORD
        valueFrom:
          secretKeyRef:
            name: avatar-secrets
            key: authentik-db-password
  authentik:
    secret_key: ""
    postgresql:
      password: ""
```

`authentikDatabase.existingSecret` is the Avatar side of the same database password. It tells the bootstrap Job and the wait-for-database init containers where to read it.

SeaweedFS

SeaweedFS reads its admin identity from one JSON document, not from a key for each credential. Build that Secret manually. The same Secret can also hold the SeaweedFS admin user and password:

```
kubectl create secret generic avatar-seaweedfs-secrets \
  --from-literal=sse-kek=<64 hex characters> \
  --from-literal=admin-user=admin \
  --from-literal=admin-password=<password> \
  --from-literal=seaweedfs_s3_config='{"identities":
[{"name":"power_user","credentials":
[{"accessKey":"...","secretKey":"..."}],"actions":["Admin"]}]}'
```

```
storage:
  sseKek: "" # empty: the entry below supplies it
  seaweedfsAdminPassword: "" # empty: admin.secret below supplies it
seaweedfs:
  admin:
    secret:
      existingSecret: avatar-seaweedfs-secrets
      userKey: admin-user
      pwKey: admin-password
  s3:
```

© Octopize — All rights reserved

```
    existingConfigSecret: avatar-seaweedfs-secrets
    extraEnvironmentVars:
      WEED_S3_SSE_KEK:
        secretKeyRef:
          name: avatar-seaweedfs-secrets
          key: sse-kek
```

The access key and the secret key in that JSON document are the same pair that `storage.existingSecret` gives to the API. The two must agree, and the chart refuses to render if a credential stays in a place that would win without a message.

After you change the admin password in the Secret, run `kubectl rollout restart statefulset/seaweedfs-admin`.

Overriding a value the chart sets

Three of the ConfigMaps that the chart renders hold its runtime configuration. Their keys become environment variables in the containers. The others hold Authentik's branding, templates and translations, which are files and not settings.

ConfigMap	Read by	Made from
`avatar-api-config`	the API pods, the sync-tables CronJob, the Dask scheduler, the `helm test` Pods	`ingress`, `envName`, `database`, `storage`, `sso`, `smtp`, `telemetry`, `api.config`
`avatar-web-config`	the web pods	`ingress`, `storage`, `web.config`, `envName`
`avatar-authentik-config`	the Authentik server and worker	`ingress`, `api.service.port`, `sso.trustedProxyCidrs`

Do not edit these objects with `kubectl`. The next `helm upgrade` renders them again from your values. To change a key, set the value that the chart makes it from. The tables on this page give those values. If the chart has no value for that key, override the key on the container.

On the workloads the chart owns

`api`, `web`, `syncTables` and `tests` each take an `extraEnvVars` list in the Kubernetes `env` shape. The chart writes its own entries first, then adds this list. The API server accepts duplicate names and defines no order, but the kubelet builds the environment in sequence, so the container gets the last entry with a given name. An entry here therefore replaces the value from the ConfigMap.

This example points `DB_HOST` at a ConfigMap of your own:

```
api:
  extraEnvVars:
    - name: DB_HOST
      valueFrom:
        configMapKeyRef:
          name: my-database-config
          key: host
```

© Octopize — All rights reserved

The same list also replaces a credential. The API reads its credentials from files that the chart mounts, but an environment variable has a higher priority than those files. An entry with the name of a credential thus wins:

```
api:
  extraEnvVars:
    - name: DB_PASSWORD
      valueFrom:
        secretKeyRef:
          name: my-database-credentials
          key: password
```

An empty value does not override: the API ignores an empty environment variable and uses the mounted file.

Use the same list for a variable that the chart does not have at all, such as an `HTTPS_PROXY`.

Two more points. `api.extraEnvVars` also goes to each init container of the chart, which are the database setup and the CA-bundle merge, because their environment must be correct before the migrations start. And Helm does not process these values as templates, so a `{{` in a value is text, not syntax.

On the subcharts

`extraEnvVars` does not go to Authentik, Dask or SeaweedFS. Each subchart has its own key:

Subchart	Key	Shape
`authentik`	`authentik.global.env` for both pods, `authentik.server.env` or `authentik.worker.env` for one	a list of `env` entries with `value` or `valueFrom`
`dask`	`dask.scheduler.env`, `dask.worker.env`	the same list shape
`seaweedfs`	`seaweedfs.global.seaweedfs.extraEnvironmentVars` for all components, `seaweedfs.<component>.extraEnvironmentVars` for one	a map of variable names: a string becomes `value`, a map becomes `valueFrom`

- **Authentik.** `avatar-authentik-config` arrives through `authentik.global.envFrom`. Kubernetes gives `env` a higher priority than `envFrom`, so a `global.env` entry with the same name replaces the key. If you override `envFrom`, keep the `configMapRef` entry. Without it, Authentik loses the domain that it registers the Avatar application under.
- **Dask.** `dask.scheduler.env` already reads `ENV_NAME` and `IS_SENTRY_ENABLED` from `avatar-api-config` with `configMapKeyRef`. Read any other key of that ConfigMap, or a key of your own Secret, in the same way.
- **SeaweedFS.** The `extraEnvironmentVars` of a component replaces the global entry with the same name. A map value becomes a `valueFrom`, as `WEED_S3_SSE_KEK` above shows.

```
dask:
  worker:
    env:
```

© Octopize — All rights reserved

```
    - name: LOG_LEVEL
      valueFrom:
        configMapKeyRef:
          name: avatar-api-config
          key: LOG_LEVEL
```

⚠️ Helm replaces an `env` list, it does not merge it

If your values file sets `authentik.global.env`, `dask.scheduler.env` or `dask.worker.env`, write again each entry that the shipped `values.yaml` sets. If you do not, the pods lose those entries.

ℹ A container reads its environment once, at start

A change to your own ConfigMap or Secret does not restart anything. The `checksum` annotations of the chart cover only what the chart renders. Do `kubectl rollout restart` on the workloads that read the value.

Database

You supply the server. The chart creates the databases, the roles and the extensions on it.

Value	Default value	Description
`database.host`	*Required*	Hostname of the PostgreSQL server.
`database.port`	`5432`	Port of that server.
`database.name`	`avatar`	Name of the Avatar database. Same as `DB_NAME`.
`database.username`	`avatar`	Role that the API connects as. Same as `DB_USER`.
`database.password`	*Required*	Password of that role.
`database.adminUsername`	*Required*	Admin account that creates the roles and the databases. Not mounted into the running API.
`database.adminPassword`	*Required*	Password of the admin account.

© Octopize — All rights reserved

What the chart creates

Object	Default name	Created by	Connects as
The Avatar role	`database.username`, `avatar`	the `init-avatar-api` init container, which runs `bin/dbtool.py setup`	`database.adminUsername`
The Avatar database, its tables and the first organization	`database.name`, `avatar`	the same init container	`database.adminUsername`
The Authentik role	`authentik.authentik.postgresql.user`, `authentik`	the `avatar-authentik-db-init-<chart-version>` Job	`database.adminUsername`
The Authentik database, with the `uuid-ossp` and `citext` extensions	`authentik.authentik.postgresql.name`, `authentik`	the same Job	`database.adminUsername`
The Authentik tables	—	the Authentik server, at the first start	`authentik.authentik.postgresql.user`

Both steps are idempotent and run again at each rollout, because the init container is part of the API pod template and the name of the Job contains the chart version. The init container also migrates the Avatar schema at an upgrade: it runs the migrations on an existing database, and creates the tables on a new one.

`authentikDatabase.createDatabase: false` stops the Job. Use it when a DBA already created the Authentik role and database, or when no account available to the cluster can create one. The Avatar database has no equivalent switch, because the init container also does the migrations. Create in advance what you want to, and the init container finds it.

Which role does what

Role	Does	Used by
`database.adminUsername`	creates the roles and databases, sets up the schema, migrates	the `init-avatar-api` init container and the Authentik bootstrap Job
`database.username`	each query that Avatar does at run time	the API container, the sync-tables CronJob and the `helm test` Pods
`authentik.authentik.postgresql.user`	the schema and the queries of Authentik	the Authentik server and worker

The admin password is not in the Secret of the running API. It goes to the init container and the bootstrap Job only, so a running API container cannot read it.

What each role needs

`database.adminUsername` is the only account that you must create. It does not have to be a superuser.

© Octopize — All rights reserved

It needs	Because
`LOGIN`	It connects to the server.
`CONNECT` on `template1`	The setup runs its server-wide statements there. It also looks for the role and the database there, even when the two already exist. If your server does not permit this, set `DB_MASTER_NAME` in `api.extraEnvVars`.
`CONNECT` on `postgres`	The Authentik bootstrap Job does the same from the `postgres` database.
`CREATEROLE`	It creates the two roles. Not necessary if you create them.
`CREATEDB`	It creates the two databases. Not necessary if you create them.
Ownership of the Avatar database and of its `public` schema	The setup creates the tables there, runs the migrations, and does `REVOKE CREATE ON SCHEMA public FROM PUBLIC`. `CREATE DATABASE` without an `OWNER` clause gives it the database. On PostgreSQL 15 and later that carries the `public` schema too. Before 15 the schema stays with the bootstrap superuser and you must hand it over.

`database.username` needs `LOGIN` only. The setup gives it `SELECT`, `INSERT`, `UPDATE` and `DELETE` on each table in `public`, and does this again at each rollout, so a migration that adds a table does not leave it without permissions. `CONNECT` comes from the default grant of PostgreSQL to `PUBLIC`. The role never gets `CREATE` and never does DDL.

The Authentik role owns the Authentik database and runs the Authentik migrations. The chart gives it no permission on the Avatar database.

Rotating a database password

The chart sets the password of a role one time, when it creates the role. It never changes it after that. The setup creates the Avatar role only when the role is absent, and the bootstrap Job creates the Authentik role only when it is absent. Neither one does `ALTER ROLE`.

A new value in `database.password` therefore does not reach PostgreSQL. It only changes the password that the API sends. The two values no longer agree, so the API cannot log in. The `init-avatar-api` init container stops at its connection check, and the pod does not start.

Change the password on the server first:

```
ALTER ROLE avatar WITH PASSWORD '<new password>';
```

Then set the new value, or write it into the Secret that holds it, and restart the workloads that read it:

```
kubectl rollout restart deployment/avatar-api
```

The sync-tables CronJob and the `helm test` Pods read the same password. They get it at their next run, so they need no restart.

© Octopize — All rights reserved

`authentik.authentik.postgresql.password` works in the same way. Run `ALTER ROLE authentik WITH PASSWORD '<new password>'`, then restart `deployment/avatar-authentik-server` and `deployment/avatar-authentik-worker`.

> **i This failure is loud, and it loses nothing**
>
> The API does not serve with a password that does not work, and no data changes. To recover, run the `ALTER ROLE` above, or put the old value back.

Creating the databases manually

The SQL below does what the chart does. Run it as a superuser, or as any account that can create roles and databases on that server.

```
-- 1. The admin account that you give to the chart. You can remove CREATEDB and
--    CREATEROLE if you also do steps 2 and 3.
CREATE ROLE avatar_dba WITH LOGIN PASSWORD '<database.adminPassword>' CREATEDB
CREATEROLE;

-- 2. The Avatar role and database. Optional: the setup creates the two if they
--    are absent. Create them here to remove CREATEDB and CREATEROLE above.
CREATE ROLE avatar WITH LOGIN PASSWORD '<database.password>';
CREATE DATABASE avatar OWNER avatar_dba;

-- 3. The Authentik role and database. This is what
--    authentikDatabase.createDatabase: false expects to find.
CREATE ROLE authentik WITH LOGIN PASSWORD
'<authentik.authentik.postgresql.password>';
CREATE DATABASE authentik OWNER authentik;
\connect authentik
CREATE EXTENSION IF NOT EXISTS "uuid-ossp";
CREATE EXTENSION IF NOT EXISTS "citext";
```

> **i PostgreSQL 14 and earlier**
>
> The bootstrap superuser owns the `public` schema, not the owner of the database. The setup does not fail without this step. Its `REVOKE CREATE ON SCHEMA public FROM PUBLIC` prints `WARNING: no privileges could be revoked for "public"` and continues, so the install succeeds and that one hardening step is skipped with no error. Give the ownership to the admin account:
>
> ```
> \connect avatar
> ALTER SCHEMA public OWNER TO avatar_dba;
> ```

These values then describe what you created:

```yaml
database:
  host: postgres.example.com
  port: "5432"
  name: avatar
  username: avatar
  password: <database.password>
  adminUsername: avatar_dba
  adminPassword: <database.adminPassword>
```

© Octopize — All rights reserved

```yaml
authentikDatabase:
  createDatabase: false # only if you did step 3

authentik:
  authentik:
    postgresql:
      host: postgres.example.com
      name: authentik
      user: authentik
      password: <authentik.authentik.postgresql.password>
```

Keep the passwords in Secrets, not in that file. See Using existing Secrets.

Ingress and the URL contract

One host, and which path on it serves what:

Path	Served by	Prefix
`/api`	The API	Stripped
`/web`	The web client	Kept
`/storage`	The S3 gateway	Stripped
`/sso`, `/`	Authentik	Kept

`ingress.host` is the only place a deployment's domain is set. Every public URL the chart hands the API, the web client and Authentik comes from it.

© Octopize — All rights reserved

Value	Default value	Description
`ingress.enabled`	`true`	Create the Ingress objects. Set it `false` to route to the services yourself.
`ingress.host`	*Required*	The domain the platform is served on.
`ingress.authentikHost`	`ingress.host`	Give Authentik its own domain.
`ingress.className`	`nginx`	IngressClass. The routing needs ingress-nginx; on another controller set `ingress.enabled: false`.
`ingress.tls.enabled`	`true`	Serve over TLS, and derive every public URL as `https`.
`ingress.tls.secretName`	*None*	Name of an existing `kubernetes.io/tls` Secret. Left empty, the controller serves its own certificate.
`ingress.annotations`	`{}`	Extra annotations on every Ingress the chart creates.
`ingress.proxyBodySize`	`1024m`	Upload limit. Raise it for larger datasets.
`ingress.proxyBufferSize`	`16k`	Response buffer size.
`ingress.proxyBuffersNumber`	`4`	Number of proxy buffers.
`ingress.wwwRedirect`	`false`	Redirect the apex domain to its `www.` counterpart.
`ingress.api.path`	`/api`	Path the API is served under. Each route also takes its own `annotations`.
`ingress.web.path`	`/web`	Path the web client is served under.
`ingress.sso.path`	`/sso`	Path Authentik is served under. Must match `authentik.authentik.web.path`.
`ingress.storage.path`	`/storage`	Path the object store is served under.
`ingress.storage.service.name`	*None*	Point the storage route at your own Service instead of the in-chart SeaweedFS.
`ingress.storage.service.port`	`8333`	Port of that service.

Setting up the ingress

With ingress-nginx, which is what the chart generates for:

1. Point DNS for your domain at the ingress controller's load balancer.
2. Create a `kubernetes.io/tls` Secret for that domain in the release namespace. The chart never creates one.
3. Set `ingress.host` and `ingress.tls.secretName`, and leave `ingress.enabled: true` and `ingress.className: nginx`.

The chart then creates the four routes above, and every public URL follows from `ingress.host`.

Using another ingress controller

Set `ingress.enabled: false` and keep `ingress.host` set — the chart still derives every URL from it. Then create your own routes for the four paths above, with:

- the prefix stripped on `/api` and `/storage`, kept on `/web` and `/sso`
- `x-forwarded-prefix: /storage` on the storage route
- a request body-size limit large enough for your dataset uploads

© Octopize — All rights reserved

- `X-Forwarded-For` replaced, not appended to, and the pod network of your controller in `sso.trustedProxyCidrs`

`ingress.className` only applies to the objects the chart creates, and those carry ingress-nginx annotations — pointing it at another controller does not make them work.

If you also set `networkPolicy.enabled`, fill in `networkPolicy.ingressController` with your controller's pod labels.

API

Value	Default value	Description
`api.image.registry`	`quay.io`	Registry the API image is pulled from.
`api.image.repository`	`octopize/avatar-service-api`	Repository of the API image.
`api.image.tag`	`3.4.0`	API version. Set it to the version Octopize gives you; avoid `latest`.
`api.image.digest`	*None*	Pull by digest instead of by tag.
`api.image.pullPolicy`	`IfNotPresent`	Image pull policy.
`api.replicaCount`	`1`	API replicas. Keep it at `1` — see Upgrading.
`api.resources`	`4Gi` / `1000m`, requests = limits	Compute resources of the API container.
`api.containerPort`	`8000`	Port the API listens on.
`api.service.type`	`ClusterIP`	Type of the API Service.
`api.service.port`	`8000`	Port the API Service exposes.
`api.service.annotations`	`{}`	Annotations on the API Service.
`api.livenessProbe.failureThreshold`	`3`	Failures before the container restarts. `periodSeconds` and `timeoutSeconds` sit alongside it.
`api.readinessProbe.failureThreshold`	`3`	Failures before the pod leaves the Service. Same companions.

`api` also takes the extension points for scheduling, security and injection.

Application settings

Every URL below is derived from the `ingress` block when left empty. Set one only to pin it.

© Octopize — All rights reserved

Value	Default value	Description
`api.auth.jwtSecretKey`	*Required*	JWT signing key. Generate the same way.
`api.config.organizationName`	*Required*	Name of your organization. Same as `ORGANIZATION_NAME`.
`api.config.baseUrl`	derived	Public API URL. Same as `AVATAR_API_URL`.
`api.config.webClientUrl`	derived	Public web client URL, used in emails. Same as `AVATAR_WEB_CLIENT_URL`.
`api.config.log_level`	`INFO`	Same as `LOG_LEVEL`.
`api.config.useConsoleLogging`	`false`	Human-readable logs instead of structured JSON. Same as `USE_CONSOLE_LOGGING`.
`api.config.sentry.enabled`	`false`	Report errors to Sentry. Set `dsn` to your own project before turning it on.
`api.config.sentry.dsn`	*None*	Sentry project the errors go to.
`api.config.compatibilityMappingUrl`	`https://license.octopize.io/compatibility/compatibility_mapping.json`	Where the API downloads the client-compatibility mapping from. Left empty it uses that URL. Set it to `"disabled"` behind a firewall that blocks outgoing connections, and the API serves the copy shipped in its image.

TLS and private certificate authorities

Value	Default value	Description
`api.caBundle`	*None*	PEM-encoded CA certificate(s) to trust, on top of the ones the image already has.

⚠️ A certificate from a private CA needs `api.caBundle`

If your domain's certificate was issued by an internal or self-signed authority, every login fails with `CERTIFICATE_VERIFY_FAILED` until you set it:

```
api:
  caBundle: |
    -----BEGIN CERTIFICATE-----
    ...
    -----END CERTIFICATE-----
```

It reaches the API pods and the sync-tables CronJob, but not the Dask workers — see Compute.

© Octopize — All rights reserved

Web client

Value	Default value	Description
`web.image.registry`	`quay.io`	Registry the web image is pulled from.
`web.image.repository`	`octopize/avatar-web`	Repository of the web image.
`web.image.tag`	`0.68.0`	Web client version. Octopize gives it to you alongside the API version.
`web.image.digest`	*None*	Pull by digest instead of by tag.
`web.image.pullPolicy`	`IfNotPresent`	Image pull policy.
`web.replicaCount`	`1`	Web client replicas.
`web.resources`	`2Gi` / `500m`, requests = limits	Compute resources of the web container.
`web.containerPort`	`3000`	Port the web client listens on.
`web.service.type`	`ClusterIP`	Type of the web Service.
`web.service.port`	`3000`	Port the web Service exposes.
`web.service.annotations`	`{}`	Annotations on the web Service.
`web.livenessProbe.failureThreshold`	`3`	Failures before the container restarts.
`web.readinessProbe.failureThreshold`	`3`	Failures before the pod leaves the Service.

`web` takes the same extension points as `api`.

The URLs under `web.config` — `publicUrl`, `apiPublicUrl`, `storagePublicUrl`, `authentikUrl`, `ssoUrl` — are derived from the `ingress` block when left empty. Set one only to pin it. `web.config.apiInternalUrl` and `web.config.storageInternalUrl` are cluster-internal and rarely need changing.

i

The environment name the interface displays comes from `envName`.

Fitting the chart to your cluster

Each workload takes the same set of optional values, all empty by default:

© Octopize — All rights reserved

Key (per workload)	Set it for
`nodeSelector`, `tolerations`, `affinity`, `topologySpreadConstraints`	A labelled or tainted node pool, spreading over zones
`priorityClassName`	Controlling what gets evicted first when a node fills up
`podSecurityContext`, `containerSecurityContext`	A Pod Security Standard, or a policy engine enforcing one
`extraEnvVars`	Overriding a value the chart sets
`extraVolumes`, `extraVolumeMounts`	An injected config, a writable path, a mounted certificate
`initContainers`, `sidecars`	A secrets agent, a logging or service-mesh sidecar
`podAnnotations`, `podLabels`	What a sidecar injector, a scraper or a policy engine keys off
`updateStrategy`	How a rollout replaces pods
`extraPodSpec`	Any pod-spec field the chart does not expose

`api` and `web` take all of them. `syncTables` takes all but `updateStrategy`. `authentikDatabase` takes the scheduling keys, `priorityClassName`, `extraPodSpec` and the two security contexts.

`app.kubernetes.io/component` cannot be set in `podLabels`.

The subcharts have their own keys for the same fields: `authentik.server.securityContext` and `authentik.server.containerSecurityContext`, `seaweedfs.<component>.podSecurityContext`, `dask.worker.nodeSelector`. Their environment variables have their own keys too — see On the subcharts.

Restricted Pod Security Standard

`example-values-restricted.yaml`, shipped in the chart, is a ready-made overlay. Apply it on top of your own values file:

```
helm install avatar oci://quay.io/octopize/helm/services-api/avatar \
  --version <version> --namespace avatar --create-namespace \
  --values my-values.yaml \
  --values example-values-restricted.yaml
```

Two things to keep if you write your own instead:

- Set `runAsUser` alongside `runAsNonRoot: true`, or the container will not start. It differs per image: `1000` for the API, sync-tables and Dask, and `1001` with group `65533` for the web client. Giving the web client `1000` fails quietly — it serves, but every cache write fails.
- Leave `readOnlyRootFilesystem` unset unless you also add a writable mount, which is what `extraVolumes` and `extraVolumeMounts` are for.

⚠️ The Dask subchart cannot run under this profile

© Octopize — All rights reserved

Its pods are rejected by a `restricted` namespace whatever you configure, so Dask has to run somewhere that admits it. Everything else the chart deploys is covered by the overlay.

Object storage

Where datasets and results live, and the credentials the API reaches them with.

Value	Default value	Description
`storage.sharedPath`	in-chart SeaweedFS	Where datasets and generated files live. Same as `SHARED_STORAGE_PATH`.
`storage.endpoint.host`	in-chart SeaweedFS	Cluster-internal host of the S3 gateway.
`storage.endpoint.port`	`8333`	Port of that endpoint.
`storage.admin.accessKeyId`	*Required*	Identity the API authenticates to the object store as. Treat it as a root credential.
`storage.admin.secretAccessKey`	*Required*	Its secret.
`storage.sseKek`	*Required*	Key-encryption key for encryption at rest. Required when `seaweedfs.enabled`.
`storage.seaweedfsAdminPassword`	*Required*	SeaweedFS admin password. Required when `seaweedfs.enabled`, unless `seaweedfs.admin.secret.existingSecret` names your own Secret. See SeaweedFS.

🚨 Generate `storage.sseKek` once and never change it

A new key makes every stored object undecryptable. When migrating an existing SeaweedFS, copy the value out of the old filer — see SeaweedFS encryption.

In-chart SeaweedFS

Value	Default value	Description
`seaweedfs.enabled`	`false`	Deploy the in-cluster object store. `example-values.yaml` sets it `true`.
`seaweedfs.volume.dataDirs`	one 20Gi PVC	The platform's disk budget. Raise it for the data you expect to hold.
`seaweedfs.filer.data`	one 10Gi PVC	Filer metadata. This is the volume to back up.
`seaweedfs.filer.extraArgs`	see `example-values.yaml`	Overriding it replaces the chart's list, so repeat the `-ip` argument it sets.
`seaweedfs.s3.extraArgs`	see `example-values.yaml`	Same. Add `-allowedOrigins` to narrow browser CORS to the web client.
`seaweedfs.networkPolicy.enabled`	`false`	Must equal `networkPolicy.enabled`.

© Octopize — All rights reserved

⚠️ **SeaweedFS allows every browser origin until you set `-allowedOrigins`**

Neither the filer nor the S3 gateway is given the argument by default, and SeaweedFS then serves the object store to any origin. Add it to both, listing the web client URL:

```
seaweedfs:
  filer:
    extraArgs:
      - "-ip=${POD_NAME}.seaweedfs-filer.${NAMESPACE}.svc.cluster.local"
      - "-allowedOrigins=https://avatar.mycompany.com"
  s3:
    extraArgs:
      - "-iam.readOnly=false"
      - "-allowedOrigins=https://avatar.mycompany.com"
```

Helm cannot template a subchart's `extraArgs`, so the value cannot be derived from `api.config.webClientUrl` and has to be written out.

SSO

Value	Default value	Description
`sso.clientId`	*Required*	OAuth client id. The blueprint reads it back from the `avatar-api-config` ConfigMap.
`sso.clientSecret`	*Required*	OAuth client secret. The blueprint reads it back from the `api` Secret.
`sso.providerApplication`	`avatar-api`	Leave it alone. Any other value breaks login.
`sso.providerUrl`	derived	OIDC issuer.
`sso.apiToken`	*Required*	Authentik management-API token. The blueprint reads it back from the `api` Secret.
`sso.webhookSecret`	*Required*	Webhook shared secret. The blueprint reads it back from the `api` Secret.
`sso.trustedProxyCidrs`	every private range	Networks that Authentik takes `X-Forwarded-Proto` and `X-Forwarded-For` from. See The proxy in front of Authentik.

💡

You choose all four. Authentik issues none of them. Make them as described in Generating the secrets. You set each one time. The blueprint reads them back from the objects that the chart writes them into.

`sso.apiToken` and `sso.webhookSecret` are what revoke the API keys of a user you disable or delete in Authentik: the webhook secret authenticates Authentik telling the API about the event, and the token lets the API confirm that user's status before revoking anything. API keys are checked against Avatar's own database rather than against Authentik, so without them a user who has lost their single sign-on

© Octopize — All rights reserved

keeps working API keys. The chart refuses to render when either is missing, rather than installing a deployment where that enforcement is silently absent.

The proxy in front of Authentik

Authentik reads the scheme and the client address of a request from `X-Forwarded-Proto` and `X-Forwarded-For`, but only when the connection comes from an address in `sso.trustedProxyCidrs`. That address is the pod of your ingress controller. From Authentik 2026.8, Authentik ignores both headers when they come from any other address.

The default holds every private range. It covers the pod network of the usual Kubernetes networks, including `100.64.0.0/10` on Scaleway Kapsule:

```
sso:
  trustedProxyCidrs:
    - 10.0.0.0/8
    - 172.16.0.0/12
    - 192.168.0.0/16
    - 100.64.0.0/10
    - 127.0.0.0/8
    - fc00::/7
    - fe80::/10
    - ::1/128
```

If your pod network is public, add it. A list that you set replaces the default, so keep the ranges that you still need. An empty list trusts no address: it does not fall back to the Authentik default.

Authentik reads this value only at start, and `helm upgrade` does not restart it. After each change, run `kubectl -n avatar rollout restart deploy/avatar-authentik-server`.

Replace `X-Forwarded-For` at the ingress controller

Authentik takes the **first** address of `X-Forwarded-For` as the client address. It records that address and gives it to its policies. Your ingress controller must replace that header with the address that it receives the connection from. If it appends to a header that the client sent, the client chooses the address that Authentik records. On ingress-nginx, do one of these:

- Leave `use-forwarded-headers` off. This is the ingress-nginx default.
- If you need `use-forwarded-headers`, set `proxy-real-ip-cidr` to the address of your load balancer only.
- Use the PROXY protocol between the load balancer and the ingress controller.

Any pod in a trusted network that reaches the Authentik server can also set that address. Do not expose the Authentik server outside the cluster through a Service of its own. `networkPolicy.enabled` limits the pods that reach it. To trust only the ingress controller, list the addresses of its pods in `sso.trustedProxyCidrs`.

The login returns to its first screen

When the ingress controller is outside `sso.trustedProxyCidrs`, Authentik takes every request for plain HTTP. It builds `http://` URLs and sets its session cookie without the `Secure` flag. The login then returns to its first screen after each email address. A

© Octopize — All rights reserved

Chromium browser shows only an error icon, because the page asks for its API over `http://`. To fix it:

1. Find the address of the ingress controller pod:

   ```
   kubectl get pods -A -o wide
   ```

2. Add its network to `sso.trustedProxyCidrs`, and keep the ranges that you still need.
3. Run `helm upgrade`, then restart the Authentik server:

   ```
   kubectl -n avatar rollout restart deploy/avatar-authentik-server
   ```

4. Log in again, and read the log of the Authentik server:

   ```
   kubectl -n avatar logs deploy/avatar-authentik-server --tail=100
   ```

   A login request shows `"scheme": "https"`, and its `"remote"` is the address of the client, not the address of the ingress controller pod.

SMTP

Left empty, anything that would send a message fails.

Value	Default value	Docker equivalent
`smtp.host`	*None*	`SMTP_HOST`
`smtp.port`	587	`SMTP_PORT`
`smtp.useTls`	`true`	`SMTP_USE_TLS`
`smtp.startTls`	`false`	`SMTP_START_TLS`
`smtp.verify`	`true`	`SMTP_VERIFY`
`smtp.senderEmail`	*None*	`SMTP_SENDER_EMAIL`
`smtp.username`	*None*	`SMTP_USERNAME`
`smtp.password`	*None*	`SMTP_PASSWORD`

Authentik sends its own mail and is configured separately under `authentik.authentik.email`. Set both.

Telemetry

Telemetry lets Octopize support you faster, follow the versions you run and monitor license usage. Only system data, job metadata and user data (email and organization) are sent — **no sensitive data and no datasets**. Octopize provides the bucket name and credentials. See Data sent to Octopize for why we recommend it and what is sent.

Value	Docker equivalent
`telemetry.s3.endpointUrl`	`TELEMETRY_S3_ENDPOINT_URL`
`telemetry.s3.region`	`TELEMETRY_S3_REGION`
`telemetry.s3.bucketName`	`TELEMETRY_S3_BUCKET_NAME`
`telemetry.s3.accessKeyId`	`telemetry_s3_access_key_id`
`telemetry.s3.secretAccessKey`	`telemetry_s3_secret_access_key`

The export runs as the sync-tables CronJob, so it also needs `syncTables.enabled`.

© Octopize — All rights reserved

Scheduled jobs

Value	Default value	Description
`syncTables.enabled`	`false`	Enable the table-synchronisation CronJob.
`syncTables.schedule`	`0 1 * * *`	Cron schedule expression.
`syncTables.failureEmail`	*None*	Address to notify on failure. Required for the CronJob to be created.

This CronJob is what exports telemetry, so enable it if you send telemetry. It runs the API image, so `api.caBundle` reaches it too, and it takes the full set of extension points under `syncTables.*`.

Authentik

Avatar embeds Authentik as its identity provider. It configures itself on first start; the only manual step is the first admin account.

Value	Default value	Description
`authentik.enabled`	`true`	Whether to deploy Authentik.
`authentik.global.image.tag`	pinned by the chart	Authentik image tag.
`authentik.global.env`	see below	Blueprint variables.
`authentik.authentik.secret_key`	*Required*	Signing key for Authentik sessions.
`authentik.authentik.log_level`	`info`	Authentik log level.
`authentik.authentik.web.path`	`/sso/`	Must match `ingress.sso.path` with a trailing slash.
`authentik.authentik.postgresql.host`	*Required*	PostgreSQL server for Authentik's own database.
`authentik.authentik.postgresql.port`	`5432`	Its port.
`authentik.authentik.postgresql.name`	`authentik`	Database name.
`authentik.authentik.postgresql.user`	`authentik`	Role name.
`authentik.authentik.postgresql.password`	*Required*	Password for that role.
`authentik.authentik.email.*`	see `example-values.yaml`	SMTP settings Authentik sends its own mail through.

The chart derives the domain and redirect URIs from `ingress.host`, so only these entries under `authentik.global.env` are left to set. They are described in full in Blueprint variables. Helm replaces a list rather than merging into it, so a values file that sets `authentik.global.env` at all has to restate every entry it needs.

© Octopize — All rights reserved

Environment variable	Description
AVATAR_AUTHENTIK_BLUEPRINT_CLIENT_ID	Refers to sso.clientId. Leave it as it is.
AVATAR_AUTHENTIK_BLUEPRINT_CLIENT_SECRET	Refers to sso.clientSecret. Leave it as it is.
AVATAR_AUTHENTIK_BLUEPRINT_SELF_SERVICE_LICENSE	License type assigned to self-service signup users.
AVATAR_AUTHENTIK_BLUEPRINT_API_TOKEN	Refers to sso.apiToken. Leave it as it is.
AVATAR_AUTHENTIK_BLUEPRINT_WEBHOOK_SECRET	Refers to sso.webhookSecret. Leave it as it is.
_GOOGLE_CLIENT_ID / _GOOGLE_CLIENT_SECRET	Optional. Set both to add a Google OAuth source.

Each of them can take a valueFrom.secretKeyRef in place of value. See Authentik. Use the same list to override any other Authentik setting, including a key that the chart makes itself. See On the subcharts.

AVATAR_AUTHENTIK_BLUEPRINT_API_WEBHOOK_URL is not in that list. The chart derives it from api.service.port and delivers it to the Authentik pods through the avatar-authentik-config ConfigMap named in authentik.global.envFrom, so it survives the list replacement above and needs no entry. An entry of that name still wins if you add one — Kubernetes gives env precedence over envFrom — but one restated with an empty value would switch the user-event webhook off, and the chart refuses to render rather than let that pass unnoticed.

Bootstrapping the Authentik database

A Job creates Authentik's role, database and extensions on first install.

Value	Default value	Description
authentikDatabase.createDatabase	true	Set false when a DBA has already created them.
authentikDatabase.image	postgres:17-alpine	Anything with a psql client at least as new as the server.
authentikDatabase.imagePullPolicy	IfNotPresent	Image pull policy for the Job.
authentikDatabase.backoffLimit	5	Retries before the Job gives up.
authentikDatabase.ttlSecondsAfterFinished	3600	How long the finished Job stays around to be read with kubectl logs.

It also takes nodeSelector, tolerations, priorityClassName and the two security contexts — see Fitting the chart to your cluster.

Compute

The Dask scheduler and workers run the avatarization jobs.

© Octopize — All rights reserved

Value	Default value	Description
`dask.enabled`	`true`	Whether to deploy the Dask scheduler and workers.
`dask.scheduler.image.repository`	`quay.io/octopize/avatar-service-api`	Scheduler image. Override it for a mirrored install.
`dask.scheduler.image.tag`	`3.4.0`	**Must equal `api.image.tag`.**
`dask.worker.image.repository`	`quay.io/octopize/avatar-service-api`	Worker image.
`dask.worker.image.tag`	`3.4.0`	**Must equal `api.image.tag`.**
`dask.worker.replicas`	`2`	Number of workers. The main dial for avatarization throughput.
`dask.worker.resources`	requests `2Gi` / `1` CPU, limits `32Gi` / `1` CPU	Raise the memory limit for large datasets.
`dask.jupyter.enabled`	`false`	Leave it off.

i

The Dask pods pull with `docker-local-pull-secret`. If you bring your own pull secret through `global.imagePullSecrets`, edit `dask.{scheduler,worker}.image.pullSecrets` to match.

⚠️ `api.caBundle` does not reach the Dask workers

If a worker talks TLS to anything using your private CA, point the workers at it with the block commented out under `dask.worker` in `values.yaml`. There the CA bundle replaces the trust store, so it has to carry any public roots the worker also needs.

Dask is also the one component that cannot run under a restricted Pod Security Standard.

Network isolation

Value	Default value	Description
`networkPolicy.enabled`	`false`	Create `NetworkPolicy` objects. Needs a CNI that enforces them.
`networkPolicy.ingressController.podSelector`	`{}`	Pod labels of your ingress controller, so it is allowed to reach the platform.
`networkPolicy.ingressController.namespaceSelector`	`{}`	Only needed when the ingress controller runs in a different namespace than the release.

`seaweedfs.networkPolicy.enabled` must be set to the same value; the chart refuses to render otherwise.

© Octopize — All rights reserved

Upgrading

Upgrading a Helm-installed Avatar deployment is a single `helm upgrade` against the same values file you installed with. This guide covers the safe procedure, and the one place where Kubernetes differs from the Docker path in a way you need to know about before you start.

> **i Recovery is by re-installing, not by rolling back**
>
> `helm rollback` is untested against this chart. Plan your recovery around re-installing the previous chart version from a database backup — see Rolling back.

Prerequisites

Before starting the upgrade, ensure you have:

- `kubectl` and `helm` access to the cluster and the release namespace
- The values file you installed with, up to date with any changes made since
- The target chart version and the matching image tags from Octopize
- **Network connectivity to pull the new images** (or pre-mirrored images for air-gapped deployments):
  - Egress on port 443 to `quay.io` for the Avatar images and the chart artifact
  - Egress on port 443 to `docker.io` for supporting images
  - Egress on port 443 to `ghcr.io` for Authentik images
- Enough scheduling headroom for a rolling update of the web and Dask deployments

Pre-upgrade checklist

All items should be verified before proceeding.

- ☐ Current chart version and image tags documented (`helm -n avatar list`)
- ☐ Release notes for the target version reviewed
- ☐ Full backup of the PostgreSQL database completed
- ☐ Your values file backed up, and its diff against the deployed release reviewed (`helm -n avatar get values avatar`)
- ☐ Maintenance window scheduled and communicated to users
- ☐ Recovery plan reviewed and understood (see Rolling back)

Check the current version

```
helm -n avatar list
kubectl -n avatar get deploy -o jsonpath='{range .items[*]}{.metadata.name}{"\t"}
{.spec.template.spec.containers[0].image}{"\n"}{end}'
```

Back up the database

Take a snapshot, or `pg_dump` against `database.host`. Back up **both** the Avatar database (`database.name`) and Authentik's (`authentik.authentik.postgresql.name`) — restoring one without the other leaves logins broken.

If you rely on the in-chart SeaweedFS, snapshot its PersistentVolumes as well.

© Octopize — All rights reserved

Set the SeaweedFS admin password

The chart releases after 0.0.97 need a SeaweedFS admin password. Without it, the chart does not render. Before you upgrade, add `storage.seaweedfsAdminPassword` to the file of generated secrets that you keep, for example `avatar-secrets.yaml`:

```
storage:
  seaweedfsAdminPassword: "<output of: openssl rand -hex 32>"
```

To read the password from a Secret of your own instead, see SeaweedFS.

Upgrade to chart 0.1.0 through chart 0.0.102

Chart 0.1.0 runs Authentik 2026.8, which starts only on a database that already ran Authentik 2026.5. Authentik upgrades its database one release line at a time, and cannot downgrade it.

- From chart 0.0.101 or older, upgrade to chart **0.0.102** first, which runs Authentik 2026.5. Wait until Authentik is ready on it:

  ```
  kubectl -n avatar rollout status deploy/avatar-authentik-server
  ```

  Then upgrade to 0.1.0, with the same procedure.
- A new installation can install chart 0.1.0 directly.
- If you skip chart 0.0.102, the Authentik server stops at start with `Major version skips are not allowed`, before it changes the database. Upgrade to chart 0.0.102, let Authentik start, then upgrade to 0.1.0 again.
- Back up Authentik's database before each of the two upgrades.

Chart 0.1.0 also gives Authentik its base URL, `AUTHENTIK_WEB__BASE_URL`, from `ingress.authentikHost` or `ingress.host`. Authentik requires a base URL from 2026.11.

Restart Authentik after the release that adds `sso.trustedProxyCidrs`

The chart release that adds `sso.trustedProxyCidrs` gives Authentik the networks that it takes `X-Forwarded-Proto` and `X-Forwarded-For` from. Its release notes name it. Authentik reads this value only at start, and `helm upgrade` does not restart it. After you upgrade to that release, and after each change to `sso.trustedProxyCidrs`, restart the Authentik server:

```
kubectl -n avatar rollout restart deploy/avatar-authentik-server
kubectl -n avatar rollout status deploy/avatar-authentik-server
```

Until it restarts, an Authentik server on a pod network outside the Authentik default, for example `100.64.0.0/10` on Scaleway Kapsule, takes every request for plain HTTP, and the login returns to its first screen. See The login returns to its first screen.

Running the upgrade

Step 1: update the image tags in your values file

Three values move together on every version bump:

```
api:
  image:
    tag: "<new-api-version>"

web:
```

© Octopize — All rights reserved

```
  image:
    tag: "<new-web-version>"

dask:
  scheduler:
    image:
      tag: "<new-api-version>" # must equal api.image.tag
  worker:
    image:
      tag: "<new-api-version>" # must equal api.image.tag
```

If you run the database connector, its image has its own version line: `connector.image.tag` does not follow `api.image.tag`. Raise it when the release notes ask you to.

Step 2: preview the change

Rendering the new chart with your values file catches a missing or contradictory value before anything in the cluster changes:

```
helm upgrade avatar oci://quay.io/octopize/helm/services-api/avatar \
  --version <new-version> \
  --namespace avatar \
  --values my-values.yaml \
  --dry-run
```

Step 3: upgrade

```
helm upgrade avatar oci://quay.io/octopize/helm/services-api/avatar \
  --version <new-version> \
  --namespace avatar \
  --values my-values.yaml
```

Two things happen automatically as the new pods roll out:

- **Database migrations**, run by the API's init container. A failure shows up as the API pod staying in `Init`, with the old pod still serving traffic.
- **The Authentik bootstrap Job re-runs.** It does nothing when the database already exists.

Step 4: monitor the rollout

```
kubectl -n avatar get pods -w
kubectl -n avatar rollout status deploy/avatar-api
kubectl -n avatar logs deploy/avatar-api -c init-avatar-api
```

Step 5: verify the deployment

Run through the Verification & Troubleshooting checks — at minimum the API health check, the login flow test and a sample avatarization.

Post-upgrade verification checklist

- ☐ All pods `Running`, none in a restart loop
- ☐ `avatar-authentik-db-init-<new-chart-version>` Job `Complete`
- ☐ Authentik server restarted, if the release adds `sso.trustedProxyCidrs` or you changed it
- ☐ API health endpoint returns OK with the correct version
- ☐ User login succeeds through Authentik

© Octopize — All rights reserved

- ☐ Sample avatarization job completes successfully
- ☐ If the database connector is enabled, its pod is `Running` and **Test** succeeds on a demo connection
- ☐ Object storage is accessible, with no SSE-S3 errors
- ☐ PDF report generation works
- ☐ Email sending works

Keep the API at one replica

Leave `api.replicaCount` at `1` across the upgrade, and in general. More than one API pod is not currently supported.

Rolling back

Plan recovery as a fresh install of the previous version:

1. Restore the PostgreSQL databases from the backup taken before the upgrade. Migrations are not reversible, so a database migrated forward will not work with the older API image.
2. Re-run `helm upgrade` with the **previous** chart version and image tags, from the values file you backed up.
3. Verify with the Verification & Troubleshooting checks.

Before doing any of that, collect diagnostic information so the issue can be investigated afterwards.

> ⚠️
>
> If you are on a managed PostgreSQL where you cannot restore in place, restore to a new database and repoint `database.host` / `database.name` rather than upgrading over a database that has already been migrated forward.

Best practices

- Test upgrades against a staging cluster first
- Upgrade during low-usage periods
- Keep the values file in version control, so the diff between two releases is reviewable
- Keep the upgrade output and the init-container logs for troubleshooting

Getting help

If you encounter issues during the upgrade:

1. **Check the logs**: the API init container first, then the Authentik bootstrap Job
2. **Verify the configuration**: `helm -n avatar get values avatar` against your values file
3. **Collect diagnostics**: follow the diagnostic collection procedure
4. **Contact support**: reach out to Octopize support with the specific error messages and the diagnostic files

For emergency recovery assistance, contact Octopize support immediately with your backup files ready.

© Octopize — All rights reserved

Verification & Troubleshooting

This page describes common verification and troubleshooting procedures for Avatar deployments installed with the Helm chart. These procedures apply after both an initial installation and an upgrade.

All commands below assume the release is in the `avatar` namespace. Adjust `-n` if you installed it elsewhere.

Quick verification

Everything is running

```
kubectl -n avatar get pods
kubectl -n avatar get jobs
kubectl -n avatar get ingress
```

Expect the API, web, Dask scheduler and worker, Authentik server and worker pods `Running`, the SeaweedFS StatefulSets `Running` if `seaweedfs.enabled` is true, the `avatar-connector` pod `Running` if `connector.enabled` is true, and the `avatar-authentik-db-init-<chart-version>` Job `Complete`.

API health

The API exposes an unauthenticated health endpoint through the ingress:

```
curl https://<your-domain>/api/health
```

This should return `{"message":"ok"}`. If it fails, check from inside the cluster to tell an ingress or TLS problem from an API one:

```
kubectl -n avatar run curl --rm -it --image=curlimages/curl --restart=Never \
  -- curl -s http://avatar-api:8000/health
```

Manual testing

Deep verification

Login flow test

1. Open `https://<your-domain>/web` in a browser
2. You should be redirected to Authentik, at `https://<your-domain>/sso/`
3. Log in with the `akadmin` account you created during the first-admin flow
4. Verify you land back in the Avatar interface and that it displays the correct version

A successful round trip confirms that the Authentik blueprint registered the OAuth provider with the `sso.clientId` and `sso.clientSecret` that the API uses.

Sample avatarization test

1. Upload a small test dataset
2. Configure and run an avatarization job
3. Verify the job completes successfully
4. Download and inspect the results

This is the only check that exercises the Dask workers and the object store end to end.

© Octopize — All rights reserved

If a check fails

Start with the symptom table below, which covers the failures specific to a Helm install.

© Octopize — All rights reserved

Symptom	Fix
`helm install` fails naming a value	A required value is empty. The message says which.
Authentik pods stuck in `Init:0/1`	The database is not reachable. Check the bootstrap Job's logs, then `database.host` and the admin credentials.
The bootstrap Job fails on `GRANT`	The Authentik role already exists. Set `authentikDatabase.createDatabase: false` and create the database by hand.
Login fails at the token exchange	The registered provider holds an older `sso.clientId` / `sso.clientSecret`. Restart the Authentik server and worker, then apply the blueprint again.
Login fails with `CERTIFICATE_VERIFY_FAILED`	Set your CA certificate as `api.caBundle`.
Login fails, and `/sso/application/o/<slug>/.well-known/openid-configuration` returns 404	`sso.providerAppName` was overridden. Remove the override.
`/sso/if/flow/initial-setup/` returns 404	An admin already exists. Log in as them instead.
A user disabled in Authentik still has working API keys	Their keys go at the next periodic reconcile rather than at once, so the webhook is not reaching the API. Check that the Authentik worker has `AVATAR_AUTHENTIK_BLUEPRINT_API_WEBHOOK_URL` in its environment.
Dask pods in `InvalidImageName`	A Dask image tag is empty. Set it to `api.image.tag`.
Writes to a user bucket fail with `InternalError`	`storage.sseKek` is unset or was changed — see SeaweedFS encryption.
Pods in `ImagePullBackOff`	`imageCredentials.dockerConfigJson` is missing, malformed, or does not cover `quay.io/octopize`.
Nothing reaches the platform with `networkPolicy.enabled`	`networkPolicy.ingressController.podSelector` does not match your controller's pods, or it is in another namespace and `namespaceSelector` is unset.
SeaweedFS StatefulSets stuck `Pending`	No default `StorageClass`, or it cannot satisfy the requested size.
Pods rejected with `violates PodSecurity "restricted"`	Start from `example-values-restricted.yaml` — see Restricted Pod Security Standard. Dask pods are always rejected.
`cannot verify user is non-root`	Set `runAsUser` alongside `runAsNonRoot: true` — `1000` for the API, `1001` for the web client.
`CERTIFICATE_VERIFY_FAILED` inside a Dask worker	`api.caBundle` does not reach the workers. See Compute.
A pod stays `Pending` after setting `podSelector` or `tolerations`	Each workload takes its own. Check the one that is rendered.
`authentikDatabase.createDatabase` is unrecognised	Your chart predates this option. Create the database by hand, or bump the version.

© Octopize - All rights reserved 4 November 2026 - docs.octopize.io

Creating or recovering a user

Avatar users are managed entirely through Authentik — there is no separate internal user store. See User management, including how to recover admin access if you are locked out.

Collecting diagnostic information

If an issue cannot be resolved, collect the following before contacting Octopize support:

```
# Pod status and recent events
kubectl -n avatar get pods -o wide > diagnostics-pods.txt
kubectl -n avatar get events --sort-by=.lastTimestamp > diagnostics-events.txt

# Logs from the Avatar components
for d in $(kubectl -n avatar get deploy -o name); do
  kubectl -n avatar logs "$d" --all-containers --prefix --tail=20000
done > diagnostics-logs.txt 2>&1

# The rendered manifests, with the values that produced them
helm -n avatar get manifest avatar > diagnostics-manifest.yaml
```

> ⚠️
>
> Review diagnostic files for sensitive data (passwords, tokens, secrets) before sharing them with support. The rendered manifests contain the Secret objects the chart creates.

Provide these files along with a description of the issue when contacting Octopize support.

© Octopize — All rights reserved

Database connector (optional)

> 📝 **Optional component**
>
> The database connector is **optional** and **off by default**. Avatar works fully without it: datasets uploaded as files (CSV, Parquet) are anonymized as usual. Enable it only if your users need database runs.

Database runs read a database and write the anonymized copy into another database. The workers that run the jobs do not open these databases themselves. A separate service, the **database connector**, holds every database driver and opens the databases. The API stores the connection strings; the connector receives one only for the time of a job; the workers never see one.

This page explains how to enable the connector on a Kubernetes installation. The connector is off by default. Without it, database runs reach no database, PostgreSQL included, and the demo databases are not offered.

> ℹ **Docker Compose**
>
> The deployment tool for Docker Compose does not install the connector yet. To use database runs on a Docker Compose installation, contact Octopize support.

How it works

```
worker ── short-lived token ──▶ connector ──▶ your database
                                    │
                                    └── "what does this name mean?" ──▶ API (internal
port)
```

- A job carries the **name** of a connection and a token signed by the API. The token is valid for one connection, for one user, and for a limited time.
- The connector checks the token, then asks the API for the connection string on an **internal port**. That port is served by its own Service (`avatar-api-internal`), which no Ingress refers to, and a network policy limits it to the connector. The public port of the API does not serve this route.
- The connector keeps the connection string in memory for 30 seconds. It writes nothing to disk.
- Rows go from your database to the object storage of the job. They do not go through the API.
- The connector has no query endpoint. It lists the structure of a database, reads tables, writes into tables that already exist in a target, and deletes the rows of the tables that a run names. It does not create tables: the target must already have the schema. Error messages give a category (for example *the database refused the credentials*), never a host or a credential.

© Octopize — All rights reserved

Supported engines

Engine	Connection string	Status
PostgreSQL	`postgresql://user:password@host:5432/database`	supported
MySQL	`mysql://user:password@host:3306/database`	supported
MariaDB	`mariadb://user:password@host:3306/database`	supported
Oracle	`oracle://user:password@host:1521/SERVICE_NAME`	supported (thin driver, no Oracle client)
SQL Server	`mssql://user:password@host:1433/database`	supported
ClickHouse	`clickhouse://user:password@host:8123/database`	supported (add `?secure=true` for 8443)
MongoDB	`mongodb://…` or `mongodb+srv://…`	supported (collections read as tables)
Snowflake	`snowflake://user:password@account/database/schema?warehouse=…&role=…`	preview
Google BigQuery	`bigquery://project/dataset?credentials_base64=…`	preview, read only (source only)
IBM Db2	`db2://user:password@host:50000/database`	preview, not in the published image

A **preview** engine is offered, but it has not been proven against a real server yet. The web application marks it as a preview. The connector reports to the API the engines whose driver it carries, and the API only offers those.

IBM Db2 needs a driver that adds 135 MB and runs on amd64 only, so the published image leaves it out. To offer Db2, build the image yourself from the connector sources with `--build-arg WITH_DB2=1 --platform linux/amd64`, or contact Octopize support.

The image also contains two small demo databases (Northwind and Chinook). Your users can add them from **Account › Database connections** and try a database run before they connect their own databases. A copy of a demo target is deleted 24 hours after its last write, and when the connector restarts.

Before you start

- **Chart version 0.0.100 or later.** Version 0.0.99 added the connector, and 0.0.100 fixed its image pull secret.
- **API version 3.18.0 or later.** It is the first release with the internal port that the connector asks for connection strings.
- **Network policies.** The chart refuses to render the connector unless `networkPolicy.enabled` is `true`, because only the network policies keep the connector from reaching the rest of the cluster. Your cluster needs a CNI that enforces NetworkPolicy, such as Calico or Cilium.
- **Access to the image.** The image `quay.io/octopize/avatar-connector` is private. The chart pulls it with the same image pull credentials as the other Avatar images.

© Octopize — All rights reserved

Enable the connector

1. Generate the secrets

The connector needs two secrets:

- `connector.secret`: shared by the API and the connector. It signs the tokens that jobs carry, and the connector authenticates to the API with it.
- `databaseConnections.encryptionKeys`: the Fernet key that encrypts the connections that your users and organizations store. Without it, users cannot store their own connections (the API answers 503), but the deployment connections still work.

`scripts/generate-secrets.sh` creates both, with the other secrets of the chart. See Generating the secrets. To make them yourself:

```
# connector.secret
python -c "import secrets; print(secrets.token_hex(), end='')"
# databaseConnections.encryptionKeys
python -c "from cryptography.fernet import Fernet;
print(Fernet.generate_key().decode(), end='')"
```

Use a key manager instead of a Fernet key

The API can encrypt the stored connections with a key in a KMS-compatible key manager (AWS KMS, or a compatible service such as Scaleway Key Manager) instead of a Fernet key. The API never holds that key: each connection gets its own data key, which only the key manager can decrypt, and every decryption is subject to the key's policy and audit trail. The chart has no values for this yet; set the API environment variables through `api.extraEnvVars`:

Variable	Value
`CONNECTION_CIPHER`	`aws_kms`. Chosen automatically when `CONNECTION_KMS_KEY_ID` is set
`CONNECTION_KMS_KEY_ID`	The key ID, ARN or alias
`CONNECTION_KMS_REGION`	The region of the key
`CONNECTION_KMS_ENDPOINT_URL`	The endpoint of a KMS-compatible service. Leave it unset for AWS
`CONNECTION_KMS_ACCESS_KEY_ID`, `CONNECTION_KMS_SECRET_ACCESS_KEY`	Optional. Without them, the API uses the standard AWS credential chain

Connections already stored with a Fernet key stay readable as long as that key stays set, so you can move from one to the other.

Bring your own Secrets

If you bring your own Kubernetes Secrets, use `connector.existingSecret` (key `connector-secret`) and `databaseConnections.existingSecret` (keys `database-connections` and `connection-encryption-keys`). Change the key names with `connector.secretKeys.*` and `databaseConnections.secretKeys.*`.

2. Turn the connector on

Add this to your values file. Pin the image tag: the chart refuses to render when both `connector.image.tag` and `connector.image.digest` are empty.

© Octopize — All rights reserved

```yaml
networkPolicy:
  enabled: true

connector:
  enabled: true
  image:
    tag: "0.4.0"
```

Then upgrade the release:

```
helm upgrade avatar <chart> --values my-values.yaml --values avatar-secrets.yaml
```

The chart then runs the connector, opens the internal port of the API, and adds the network policies between them.

3. Let the connector reach your databases

The connector has its own network policy. By default, it can reach:

- DNS;
- the internal port of the API;
- the in-chart SeaweedFS, when it is enabled;
- the **public internet**, except the private, shared, link-local and loopback ranges (`10.0.0.0/8`, `172.16.0.0/12`, `192.168.0.0/16`, `100.64.0.0/10`, `169.254.0.0/16`, `127.0.0.0/8`).

A database on a public address works without any change. A database on a private network (behind a VPN or a peering) needs its network in `allowedDatabaseCIDRs`:

```yaml
connector:
  networkPolicy:
    allowedDatabaseCIDRs:
      - 10.20.0.0/16
```

An external object storage on a private address must be listed there too, because the connector writes the tables it reads into the object storage of the job.

If your DNS pods or your object storage pods do not carry the default labels, set `connector.networkPolicy.dnsSelector` (default `k8s-app: kube-dns`) and `connector.networkPolicy.storageSelector` (default `app.kubernetes.io/name: seaweedfs`, `app.kubernetes.io/component: s3`).

4. Check it works

Log in to the web application as any user, go to **Account > Database connections**, click **Add demo connections**, then **Test** next to the demo source. If the test reports the database as reachable, the API, the connector and the network policies work together. Then create a test connection to one of your own databases.

Declare connections for every user (optional)

Users can store their own connections from the web application or the Python client. You can also declare connections for the whole deployment. Every user sees them, and nobody can change them from the application:

```yaml
databaseConnections:
  connections:
    - name: crm-prod
      label: CRM (production)
```

© Octopize — All rights reserved

```
    dsn: "postgresql://avatar_reader:<password>@crm.internal:5432/crm"
  - name: crm-test
    label: CRM (test)
    dsn: "postgresql://avatar_writer:<password>@crm-test.internal:5432/crm"
```

Each entry has a `name`, a `dsn` in the format of Supported engines, and an optional `label`, which is what users see in the pickers.

These values contain passwords. Keep them in a Kubernetes Secret (`databaseConnections.existingSecret`, key `database-connections`, as a JSON list) instead of a values file.

> ⚠️ **Workers read the API Secret by name**
>
> The workers read the deployment connections from the Secret `api`, key `database_connections`. A Secret named in `databaseConnections.existingSecret` does not reach them on its own: point the `DATABASE_CONNECTIONS` entry of `dask.worker.env` at it as well.

Give the source a role that can only **read**. Give the target a role that can write and delete rows. The target must already have the schema: the run does not create tables.

© Octopize — All rights reserved

Other settings

Value	Default	Description
`connector.replicaCount`	`1`	Number of connector pods. The connector is stateless: you can scale it freely.
`connector.tokenTtlSeconds`	`21600`	How long the token of a job stays valid. Set it longer than your longest job.
`connector.storageEndpoints`	`[]`	`host:port` addresses the connector can write extracted tables to. Empty allows any. We recommend the `host:port` of your object storage, as the workers name it.
`connector.internalPort`	`8001`	Internal port of the API that the connector asks for connection strings. Must differ from `api.containerPort`.
`connector.containerPort`	`8080`	Port the connector listens on. Its Service is always `ClusterIP`.
`connector.resources`	requests 250m / 512 MiB, limits 2 CPU / 2 GiB	Compute resources of the connector container.
`connector.extraEnvVars`	`[]`	Extra environment variables of the connector container.
`connector.nodeSelector`, `tolerations`, `affinity`, `podAnnotations`, `podLabels`	empty	Scheduling and metadata of the connector pods.

The connector runs as a non-root user (UID 10001) with a read-only root file system. The complete list of `connector.*` and `databaseConnections.*` values is in the `README.md` of the chart.

Rotate the secrets

- **`connector.secret`**: when you change it, the tokens of the jobs that are running become invalid. Rotate it when no database job runs.
- **`databaseConnections.encryptionKeys`**: a comma-separated list. The first key encrypts, and all the keys decrypt. Put the new key first, wait until every stored connection is written again, then remove the old key.

© Octopize — All rights reserved

User management

Avatar users live in Authentik, the identity provider of both self-hosted paths. There is no separate user store in Avatar.

With the deploy tool (Docker)

`octopize-deploy-tool tasks create-users` creates the accounts in Authentik, puts them in the right group, and emails each user a link to set their password. Run it once the stack is up. It is safe to re-run: existing accounts are skipped.

```
octopize-deploy-tool tasks create-users --app-root ./app/avatar \
  --admins alice@company.com \
  --users bob@company.com,carol@company.com
```

`--no-email` creates the accounts without sending the email. If the email fails, the account still exists: send the link from the Authentik interface, as in step 3.

In the Authentik admin interface

Use this on Kubernetes, or when you need finer control than the command offers. Log in at `https://<your-domain>/sso/` with an account that is a superuser or has the *Avatar Tenant Admin* role, for example `akadmin`. The workflow is:

1. **Create a group** for the organization.
2. **Create the users** inside that group.
3. **Send each user a recovery email** so they set their own password.

See also the Authentik documentation on users, groups and roles.

1. Creating a group

In authentik, groups are the primary mechanism for managing permissions. Each customer company should have its own group. The group name should match the **email domain** of that customer (e.g. `facebook` for a user with `john@facebook.com`). License type is controlled via a group-level attribute.

1.1 Navigate to the Groups page

In the left-hand sidebar, expand **Directory** and click **Groups**.

© Octopize — All rights reserved

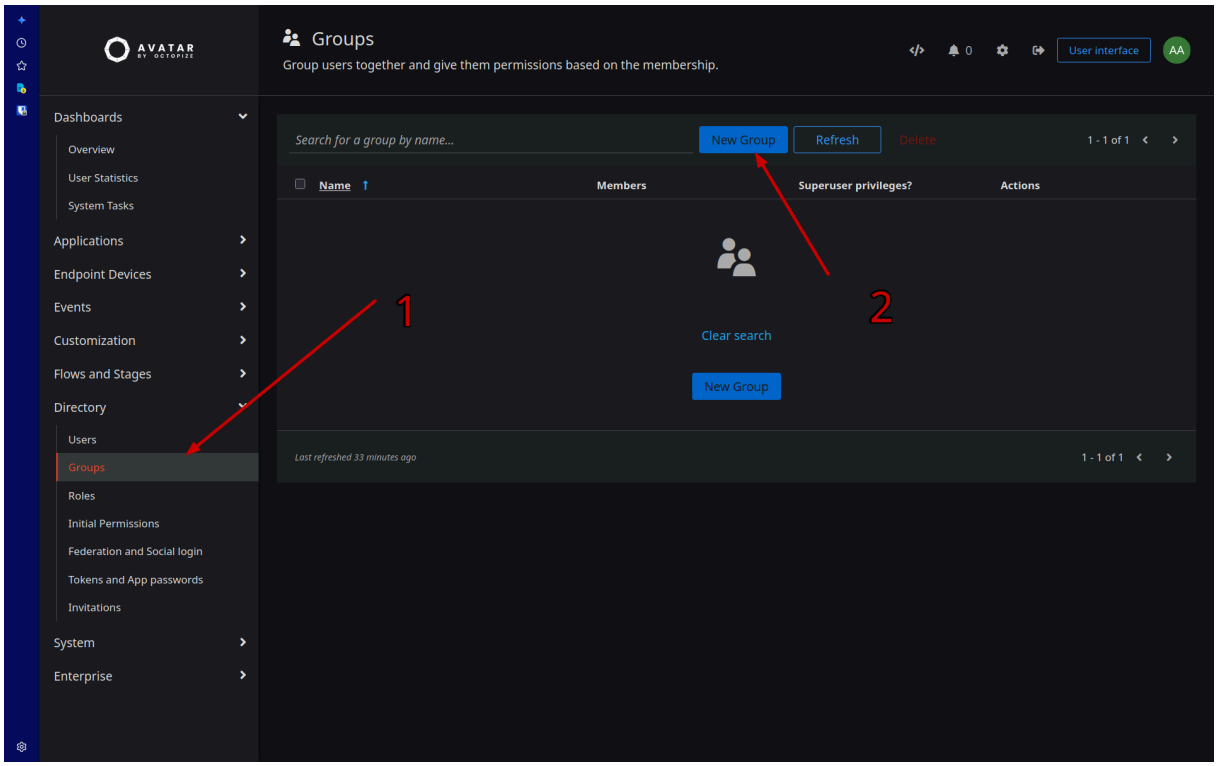


Figure 3: Groups page — navigate and click New Group

(1) Click **Groups** in the sidebar under **Directory**. **(2)** Click the **New Group** button in the top-right area of the list.

1.2 Fill in the New Group form

The **New Group** dialog will appear.

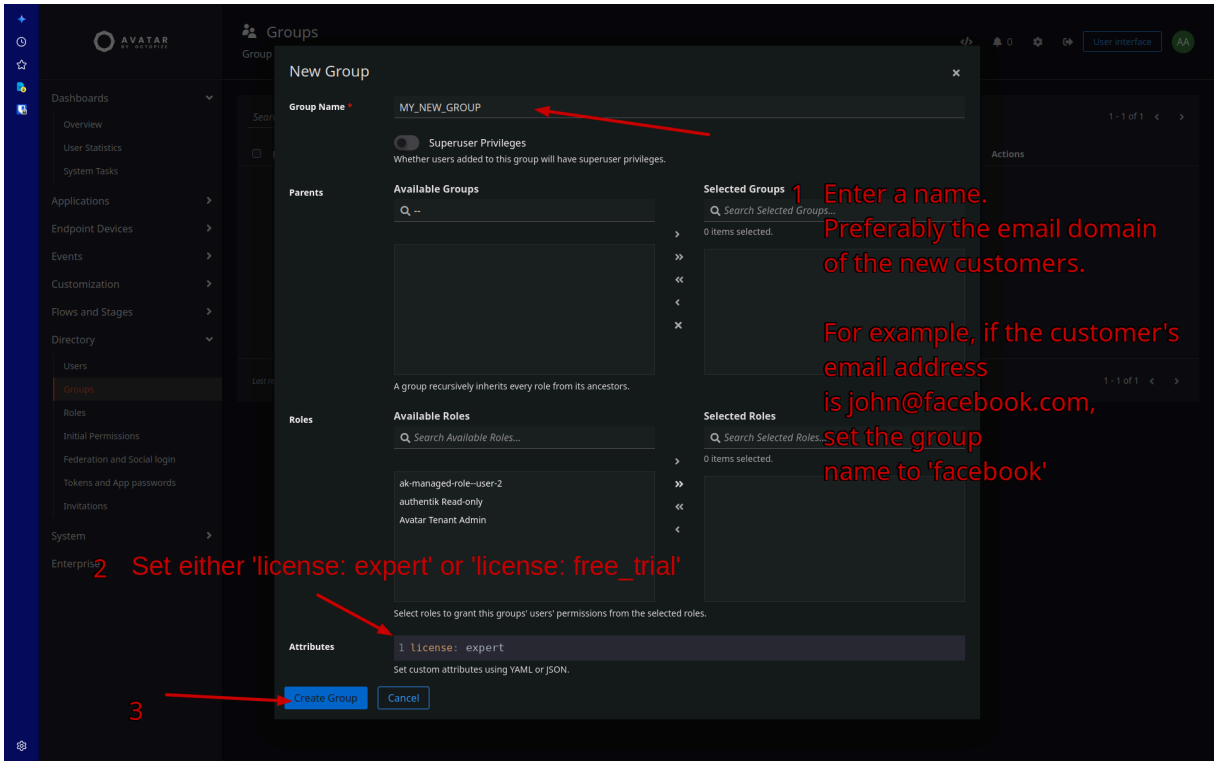


Figure 4: New Group creation form

© Octopize — All rights reserved

Fill in the fields as follows:

Field	What to enter
Group Name	The customer's email domain, e.g. `facebook`. See the annotation in the screenshot.
Superuser Privileges	Leave **disabled** unless explicitly required.
Parents / Roles	Leave empty for standard customer groups.
Attributes	Set the license type: `license: expert` for paying customers, `license: free_trial` for trial/demo access.

(1) Enter the group name — preferably the email domain of the new customer. **(2)** In the **Attributes** field, enter either `license: expert` or `license: free_trial` in YAML format. **(3)** Click **Create Group**.

> **Why the email domain?** Using the domain as the group name makes it easy to correlate groups with customer organizations, and matches what we do with Google OAuth Login.

2. Creating users

Users are created from within the group they belong to. This ensures they are automatically associated with the correct group on creation — no need to assign group membership separately.

2.1 Open the group and go to the Users tab

After creating the group, it will appear in the Groups list. Click on the **group name** to open it.

© Octopize — All rights reserved

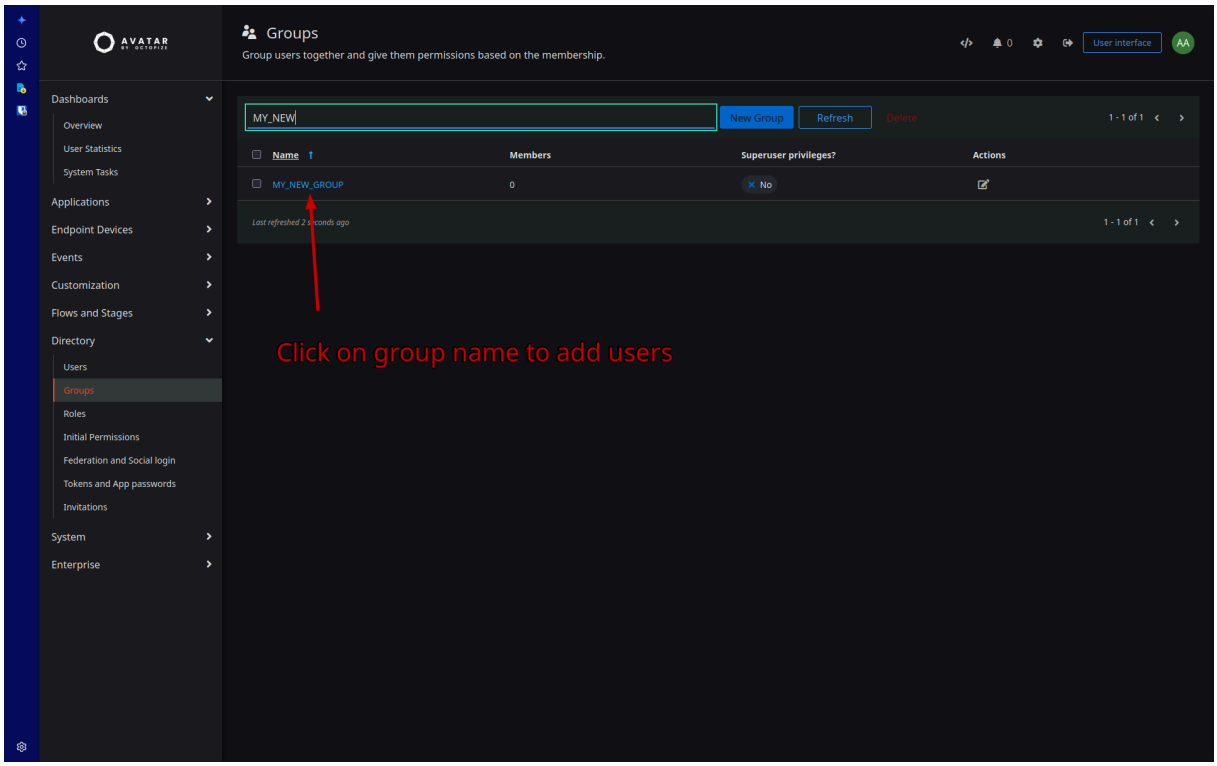


Figure 5: Group list — click group name to manage users

The group detail page will open. Click the **Users** tab at the top.

2.2 Add a new user

Inside the Users tab, click the **Add new user** dropdown button, then select **New user...**.

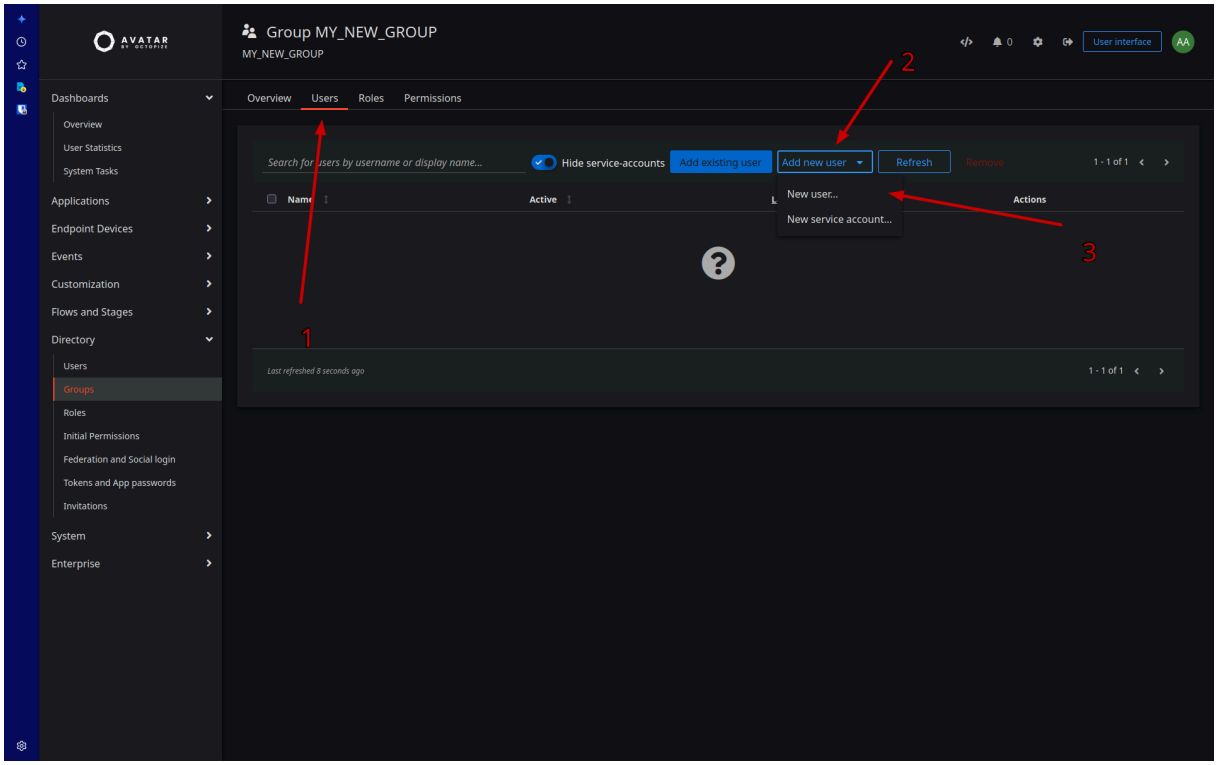


Figure 6: Group Users tab — Add new user dropdown

© Octopize — All rights reserved

(1) You are now in the **Users** tab of the group. **(2)** Click the **Add new user** dropdown button. **(3)** Select **New user…** from the dropdown (not *New service account…*).

2.3 Fill in the New User form

The **New User** dialog will appear, pre-associated with the current group.

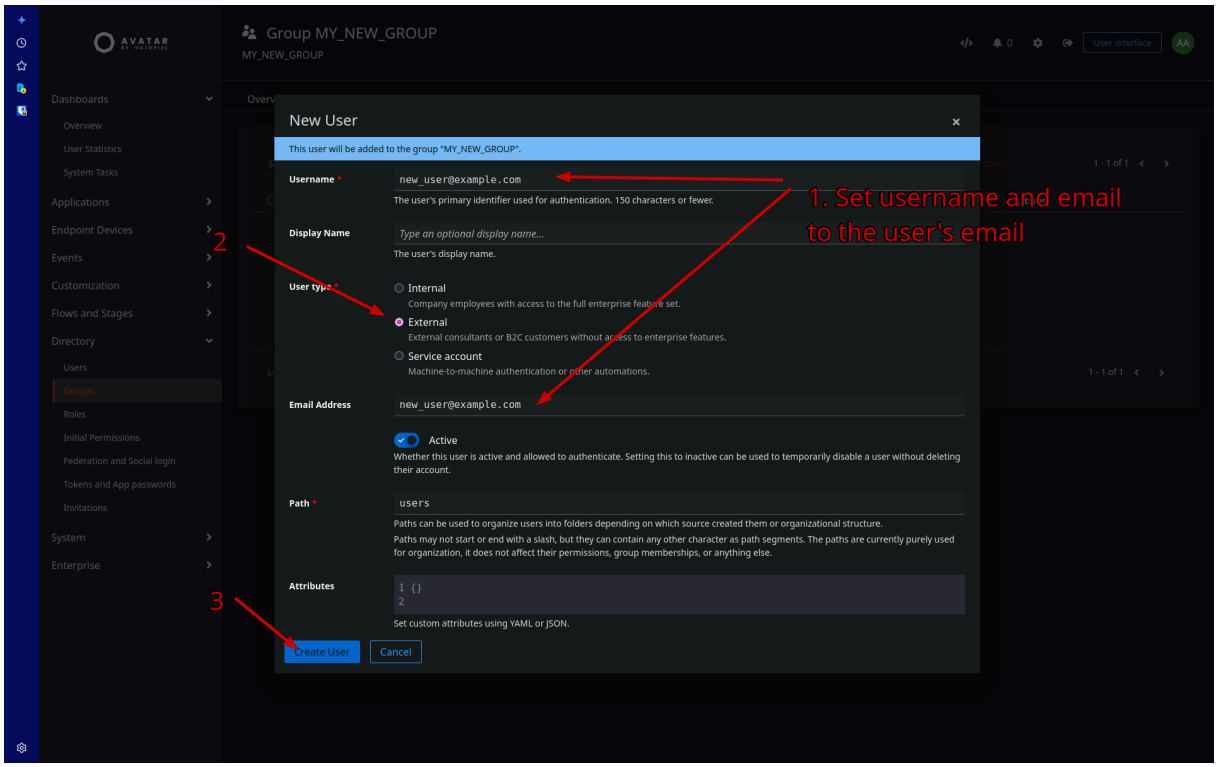


Figure 7: New User creation form

Fill in the fields as follows:

Field	What to enter
Username	The user's **email address** (e.g. `john@facebook.com`). This is their primary login identifier.
Display Name	Optional — can be left blank or set to the user's full name.
User type	Select **External** for customers (B2C users without access to enterprise features).
Email Address	Same as the username — the user's email address.
Active	Leave enabled.
Path	Leave as `users`.
Attributes	Leave empty unless specific per-user attributes are needed.

(1) Set both **Username** and **Email Address** to the user's email address. **(2)** Select the **External** user type. **(3)** Click **Create User**.

© Octopize — All rights reserved

User types: Choose *Internal* only for Octopize employees with full enterprise access. *External* is the correct type for all customers and external collaborators. See the authentik user types docs for more detail.

2.4 Confirm user creation

After clicking **Create User**, you will be redirected back to the group's Users tab. The new user will appear in the list with **Active: Yes**, and a confirmation toast will appear at the bottom of the screen.

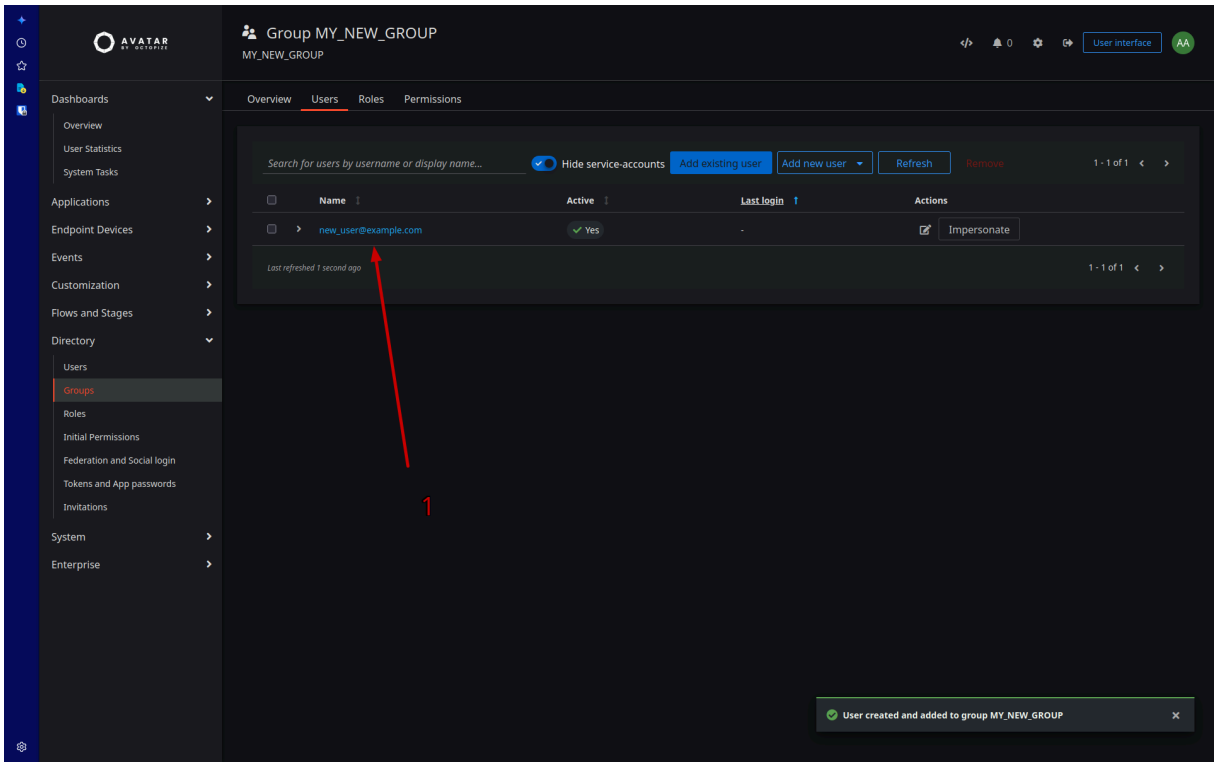


Figure 8: User created and added to group confirmation

(1) The user `new_user@example.com` is now listed as a member of the group and is active.

3. Sending the welcome email

The user has been created without a password. You must send them a **recovery link** so they can set their own password and log in for the first time.

3.1 Open the user's profile

Click on the username in the group's Users list to navigate to their profile page.

© Octopize — All rights reserved

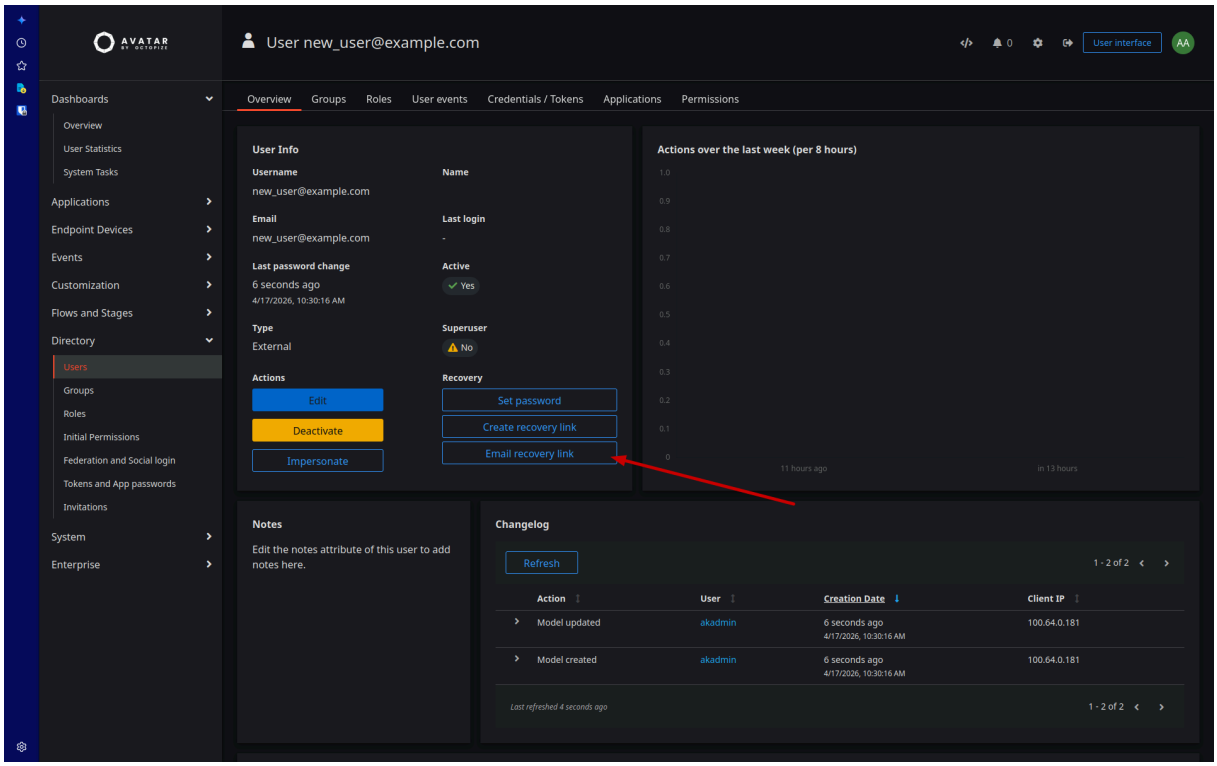


Figure 9: User overview page — Email recovery link button

On the **Overview** tab of the user profile, locate the **Recovery** section in the top-right of the User Info card, and click **Email recovery link**.

3.2 Select the email stage

A **Send recovery link to user** dialog will appear. Click the **Email stage** dropdown and select `avatar-recovery-email`.

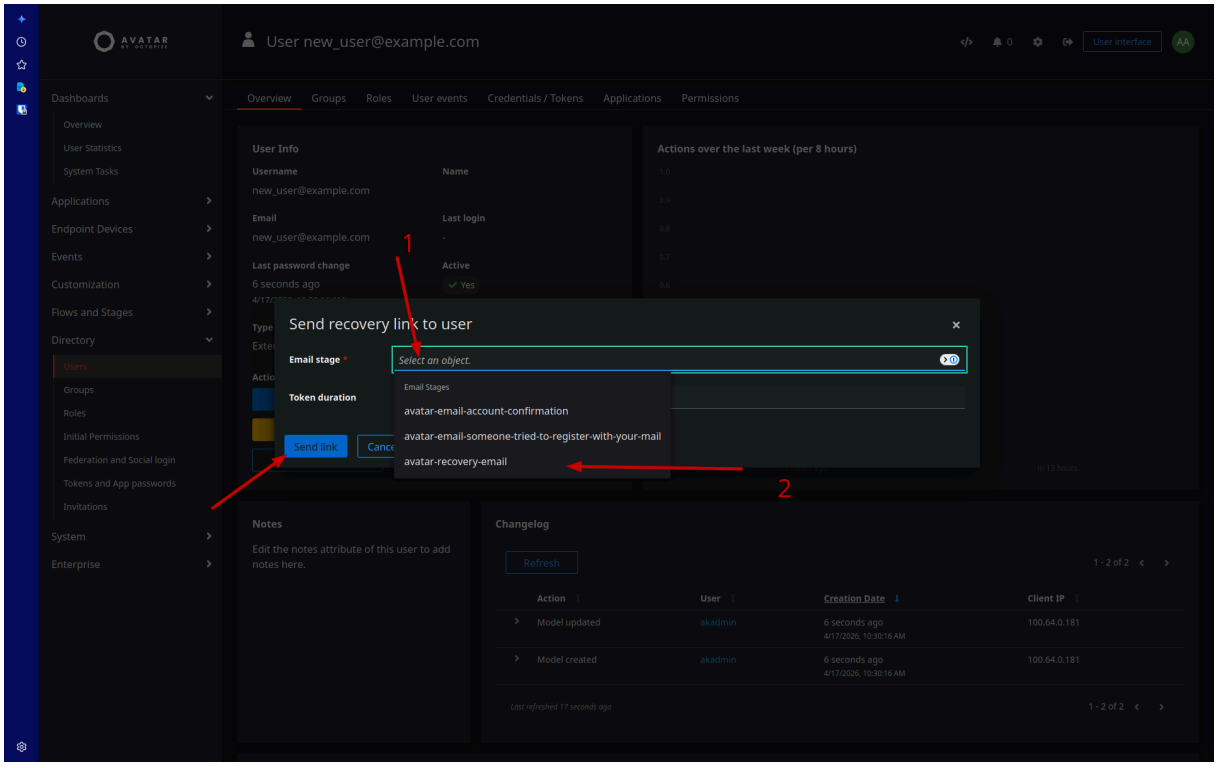


Figure 10: Send recovery link — select avatar-recovery-email stage

© Octopize — All rights reserved

(1) Click the **Email stage** field. **(2)** Select `avatar-recovery-email` from the list of available email stages.

> The `avatar-recovery-email` stage uses the branded Avatar email template. Do not select `avatar-email-account-confirmation` or other stages for this purpose.

3.3 Set the token duration and send

After selecting the email stage, review the **Token duration** (default: `days=7`). Adjust if necessary, then click **Send link**.

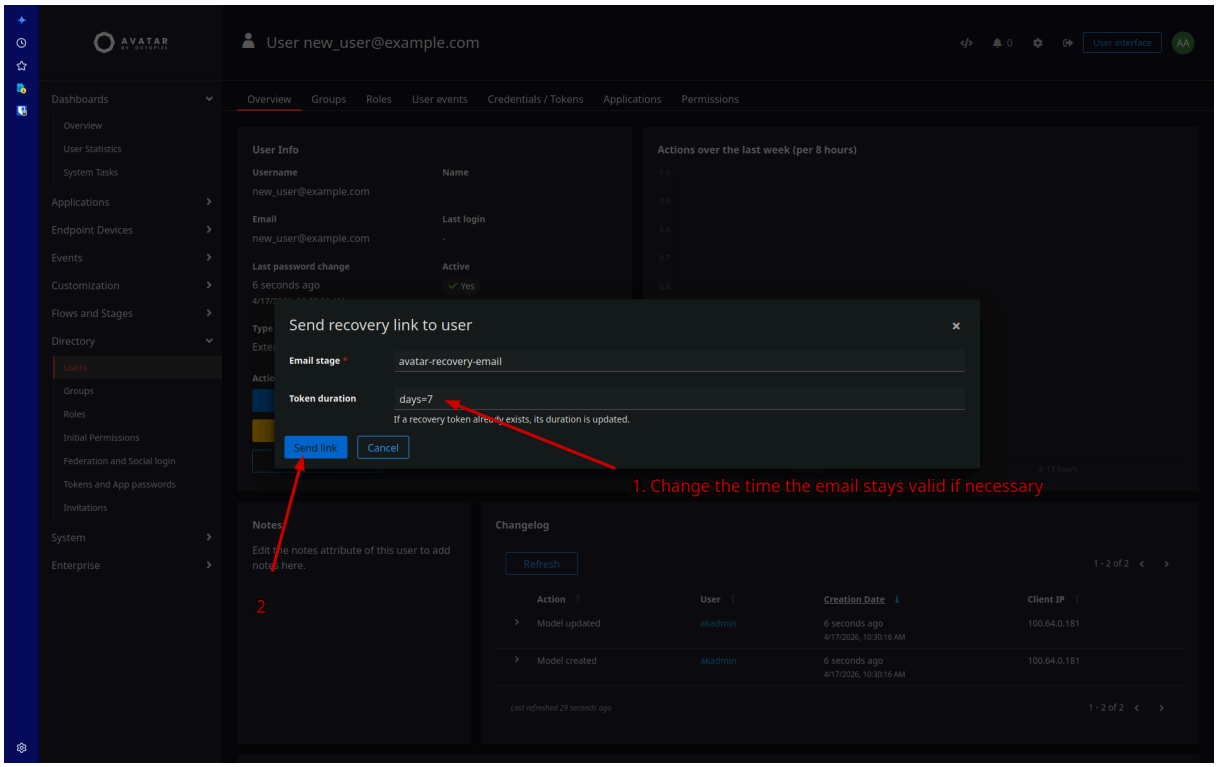


Figure 11: Send recovery link — set token duration and send

(1) Optionally adjust the **Token duration** — `days=7` is the standard validity window. **(2)** Click **Send link**.

The user will receive an email with a link that lets them set their password. The link is valid for the duration specified.

© Octopize — All rights reserved

Summary

Step	Action	Location
1	Navigate to **Directory → Groups**	Sidebar
2	Click **New Group**, set name (email domain) + `license` attribute	Groups page
3	Click the group name, go to **Users** tab	Group detail
4	**Add new user → New user…**, fill in email + select *External*	Group Users tab
5	On the user's profile, click **Email recovery link**	User Overview
6	Select `avatar-recovery-email`, set duration, click **Send link**	Recovery dialog

Further reading

- authentik — Manage Users
- authentik — Manage Groups
- authentik — Manage Roles
- authentik — Email Stage

Recovering admin access

If you lost the `akadmin` password and no other admin account exists, generate a one-time recovery link from the Authentik worker:

```
# Docker, from generated_files/
docker compose exec authentik_worker ak create_recovery_key 1 akadmin

# Kubernetes (the deployment name is prefixed with the Helm release name)
kubectl -n avatar exec deploy/<authentik-worker-deployment> -- ak create_recovery_key
1 akadmin
```

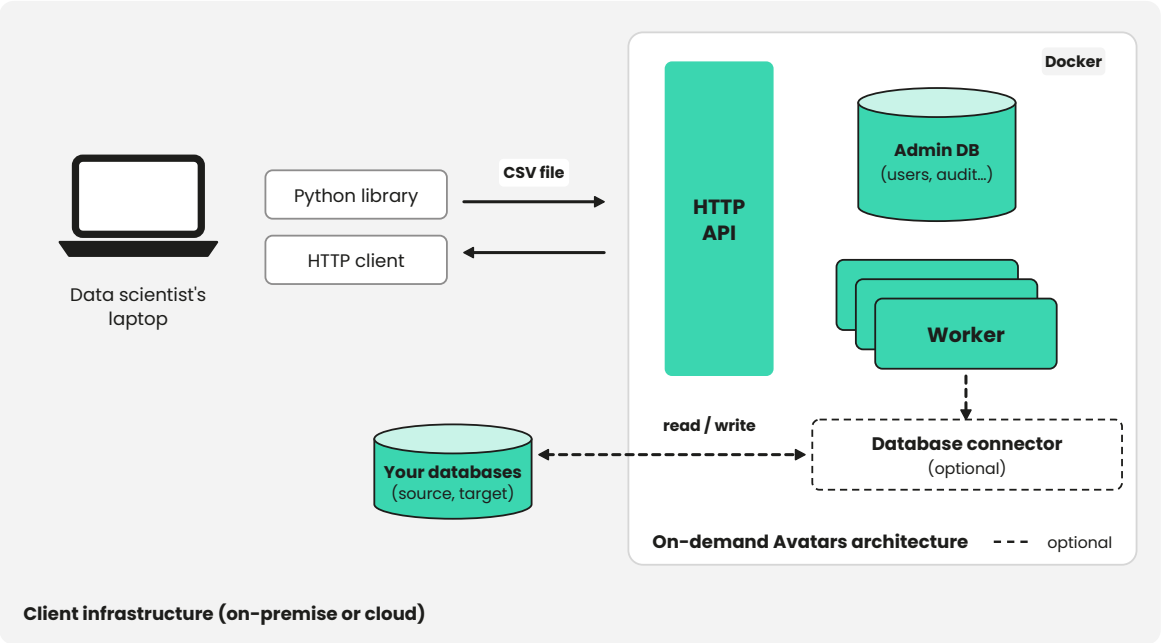
It prints a URL, valid for one year here, that lets you set a new password for `akadmin`. See the Authentik login recovery documentation.

© Octopize — All rights reserved

Architecture

Avatars have two main components:

- The platform which can be used as a SaaS or deployed on your premises.
- The client code which is installed locally.



A simplified view of the Avatars architecture

Client

The Avatars platform exposes an OpenAPI-compliant HTTP API using JSON. Consequently it can be consumed from any HTTP client: curl, Python's `requests` library, etc.

We also provide the following language-specific clients:

- Web client
- Python

Platform components

When deployed on-premise, the following components will be included:

- Stateless components
  - An HTTP proxy (Nginx) providing HTTPS termination with TLS certificate management.
  - An API server exposing an HTTP/JSON interface.
  - A dask scheduler and workers running the anonymization jobs.
  - A web GUI that provides a simplified interface into the anonymization process.
  - An OpenIDProvider (authentik) to manage users, tenants, and Single-Sign-On.
  - **Optional**: a database connector that opens the source and target databases of database runs. It is the only component that carries database drivers. It is never exposed outside the cluster. It is off by default, and everything else works without it.

© Octopize — All rights reserved

- Stateful components
  - A PostgreSQL database to store metadata (datasets, jobs) and configuration (users). We recommend using a recent version (14 as of writing) although older versions will work just as well as we don't use recent PostgreSQL features.
  - A file storage component compatible with S3, we are currently using seaweedfs.

Diagram — see the online documentation for the interactive version.

Dashed: the optional database connector, only needed for database runs. It is off by default.

© Octopize — All rights reserved

Security

We consider security as a critical requirement for all our features. We follow security-by-design practices.

You can find a security-oriented architecture diagram below. For more details, see Architecture

Diagram — see the online documentation for the interactive version.

Dashed: the optional database connector, only needed for database runs. It is off by default.

Confidentiality-wise:

- All traffic transits over an **encrypted transport** (HTTPS). We use Nginx to provide best-in-class security (configurable TLS certificate management, memory safety, hardened TLS stack).
  - As of writing, Nginx terminates HTTPS, which means that traffic between Nginx and the API server uses HTTP. This can be switched to HTTPS if necessary.
  - For On-premise deployment, you can provide your own TLS certificates.

In order to allow handling of very large datasets, data uploads and downloads are done through dedicated S3 endpoints. A dedicated authorization request has to be done for both reading and writing data on that endpoint.

- All files (original and avatar datasets) are kept for the configured retention period (TTL) and **automatically deleted** thereafter.

User authentication and authorization

All user actions are authenticated, stored and can be audited.

- Authentication requires a valid username and password. Password are stored using a secure mechanism (secure hash algorithm with a unique salt for each password + a secret pepper for the whole codebase).
- We use JWT (JSON Web Tokens) for the credentials mechanism. They are set to expire after a configurable amount of time, by default 1 hour. For more information about JWT, checkout jwt.io.
- We support two roles: `Admin` and `User`.

Typical usage data flows

Here's a typical data flow:

1. The user logs into the API (using a straight HTTPS client or via one of our client libraries) and receives a temporary JWT (JSON Web Tokens) for subsequent requests.
2. The user uploads the datasets (CSV, parquet) containing pseudonymized data to the S3 Storage endpoints, usually via one of our client libraries (web, python). This transfer is encrypted in transit through HTTPS.
3. The file is stored in storage. The file will be deleted after the configured retention period (TTL).

© Octopize — All rights reserved

4. The user can run multiple anonymization jobs on a given dataset. They can download the avatar dataset (again, through HTTPS) on their machine to conduct further analysis.
   - The avatar dataset will be deleted after a configurable amount of time (see above).

A database run reads its data from a database instead of a file:

1. The user picks a source and a target among the connections stored by Avatar. The job carries their **names**, never a password.
2. The worker hands a short-lived token to the database connector, which reads the source tables and writes them straight to the object storage of the job. The rows do not go through the API.
3. The anonymized tables are written into the target database, through the connector. Nothing is left to download, and the files of the job follow the usual retention period.

Database connections

Database runs need credentials for your databases. They are handled so that no job, no worker and no user can read them back:

- **Write-only credentials.** A connection's password is sent once, then never returned: not by the API, not by the web client, not in a job. A listing shows the host and the database name only.
- **Encrypted at rest.** The API stores each connection encrypted, with a key kept outside its database, so a database dump alone opens nothing. The key is either a Fernet key from the deployment secrets, or a key in a KMS-compatible key manager (AWS KMS, Scaleway Key Manager) that the API never holds. With KMS, each connection is bound to its organization and its name: a value copied into another row is refused.
- **Scoped.** A connection belongs to one user, to one organization, or to the deployment. See Connection scopes.
- **Workers never hold a password.** A job carries the name of a connection and a token signed by the API, valid for that connection, that user, and a limited time (6 hours by default).
- **One component holds the drivers.** Only the database connector opens your databases. It asks the API for the connection string on an internal port that the public port does not serve, that no Ingress refers to, and that a network policy limits to the connector. It keeps the string in memory for 30 seconds and writes nothing to disk.
- **No query endpoint.** The connector lists a schema, reads tables, writes into existing tables of a target, and deletes the rows of the tables that a run names. It never drops a table, and runs no query that a user writes. Its error messages give a category, never a host or a credential.
- **Contained network.** The connector is never exposed outside the cluster. Its network policy lets it reach DNS, the API's internal port, the object storage, and the public internet, but not the cluster's private networks, the platform's database or the cloud metadata service. A database on a private network must be allowed explicitly, network by network.

© Octopize — All rights reserved

On your side, give the source connection a database role that can only read, give the target a role that can only write and delete rows, and never point a target at a production database.

Network flows
Users reach the platform on a single inbound port, 443 (HTTPS), for the web client, the API, single sign-on and dataset transfers. The outbound flows are listed in Requirements. When the database connector is enabled, it also opens connections to the databases that your users connect to. The optional ones, and exactly what they send, are described in Data sent to Octopize. All of them can be turned off for an air-gapped deployment.

Security limits
The following global limits can be configured to protect the API:

- Max request body size: 1 GB (for dataset upload)
- Global rate limit: 1 request per 100 ms per client per endpoint

What is installed on the user endpoints?
The end-user will only install a client library for their programming language (e.g. Python) to streamline Avatar API usage.

Those libraries are open source and are audited using static analysis:

- https://github.com/octopize/avatar-python

What to backup?
Back up the PostgreSQL databases (Avatar's and Authentik's: users and metadata) and the deployment secrets, including the storage encryption key and, if you use database runs, the key that encrypts the stored connections (`databaseConnections.encryptionKeys`). Without that key, every stored connection is unreadable and must be entered again. See Backup.

The file storage is temporary and not meant to keep data. Once anonymized, the avatar dataset needs to be downloaded and stored somewhere else.

Development practices
We follow best practices when it comes to building secure-by-design software:

- Code reviews
- Static code analysis (using `bandit`)
- Automated dependency analysis (using `dependabot` and `quay.io`)
- Continuous integration
- Automated and manual testing (we monitor code coverage)
- Continuous developer training.
- Threat monitoring
- Our Github organization requires 2FA and commit signing.

© Octopize — All rights reserved

A self-hosted deployment makes at most three outgoing connections to services outside your infrastructure. Usage events and API requests are logged locally, and do not leave it.

Flow	Destination	Direction	Default	To turn it off
Error monitoring	`sentry.io`	Upload	On	`IS_SENTRY_ENABLED: false`
Client compatibility mapping	`license.octopize.io`	Download only	On	`COMPATIBILITY_MAPPING_URL: "disabled"`
Usage telemetry	An S3 bucket	Upload	Off until Octopize gives you credentials	Leave the telemetry values empty

No dataset and no payload content is ever sent. Only metadata. The Kubernetes equivalents of these settings are in the chart configuration.

Error monitoring (`sentry.io`)

Internal errors of the Avatar API are sent to Sentry, where Octopize analyses them.

- **Why**: find and fix production issues quickly, and keep the platform stable and secure.
- **What**: Python stack traces, the exception message and environment information (OS, installed packages). **No user data and no personal information**: breadcrumbs and local variables are disabled.
- **Confidentiality**: encrypted in transit and at rest. Retained 90 days.
- **Destination**: `sentry.io`, port 443.

Sentry is ISO 27001 certified and offers the Standard Contractual Clauses for GDPR compliance in its Data Processing Addendum. See the Sentry trust center.

We recommend leaving it on. To turn it off, set `IS_SENTRY_ENABLED` to `false` in the Docker configuration.

Client compatibility mapping (`license.octopize.io`)

The API downloads the table of which Avatar client versions work with which API versions, at startup and then every 5 minutes. The request carries no information about your deployment.

- **Why**: keep accepting client libraries released after your API image was built, instead of rejecting them.
- **What**: an HTTPS GET of `https://license.octopize.io/compatibility/compatibility_mapping.json`. **Nothing is sent.**
- **Destination**: `license.octopize.io`, port 443.

If the download fails, the API uses the copy shipped in its image, and only clients newer than that image go unrecognised. To make no request at all, set `COMPATIBILITY_MAPPING_URL` to `disabled` (Docker) or `api.config.compatibilityMappingUrl` to `"disabled"` (Kubernetes). An empty value does not disable it: the API falls back to the default URL.

© Octopize — All rights reserved

Usage telemetry (S3)

Telemetry periodically exports a small set of metadata to an S3 bucket dedicated to your deployment. It starts once Octopize has given you the bucket name and its credentials.

We recommend enabling it, because it is what lets Octopize:

- **Support you faster.** When you report an issue, Octopize already sees which jobs failed, when, and on which version. There is no back-and-forth to collect logs and versions.
- **Follow the versions you run.** Octopize knows which API and web client versions each deployment runs. It can warn you about a fix or a security update that concerns your version, tell you when your users' Python clients will stop being compatible, and plan your next upgrade with you.
- **Monitor license usage**, so your plan matches what you use.

What it sends:

- **What**: selected database tables, exported as Parquet files: system data (such as the deployed versions), job metadata (such as the job kind and name) and account data (username, email and organization). **No dataset and no payload content.** A detailed list of the exported fields is available on request.
- **Confidentiality**: encrypted in transit and at rest.
- **Destination**: the S3 endpoint set in `TELEMETRY_S3_ENDPOINT_URL`, port 443.

To enable it, see the Docker or Kubernetes configuration.

© Octopize — All rights reserved

API Keys

API keys provide a secure, long-lived authentication method for programmatic access to the Avatar API. They are ideal for automated pipelines, CI/CD integrations, and service-to-service communication where interactive login is not practical.

Overview

Unlike username/password authentication that produces short-lived JWT tokens, API keys:

- Are **long-lived** with configurable expiration (30 to 3,650 days)
- Can be **revoked** at any time without affecting the user's password
- Are **scoped to a user** and inherit that user's permissions
- Are suitable for **headless/automated** workflows

Creating an API Key

Python Client

```
client = httpx.Client(base_url="https://your-avatar-instance.com/api")
# First authenticate with username/password
response = client.post("/login", json={"username": "user@example.com", "password":
"your-password"})
token = response.json()["access_token"]

# Create an API key
response = client.post(
    "/api-keys",
    headers={"Authorization": f"Bearer {token}"},
    json={
        "name": "my-etl-pipeline",
        "expiration_days": 365
    }
)
api_key = response.json()
print(f"API Key: {api_key['key']}")  # Save this — it won't be shown again
```

Parameters

Parameter	Type	Required	Description
`name`	string	Yes	A descriptive name for the key (max 256 characters)
`expiration_days`	integer	Yes	Key lifetime. Allowed values: `30`, `60`, `120`, `365`, `3650`

⚠️

The full API key value is only returned **once** at creation time. Store it securely — it cannot be retrieved later.

© Octopize — All rights reserved

Using an API Key

Once created, send the API key in the `Authorization` header with the `api-key-v1` scheme:

```
curl -H "Authorization: api-key-v1 <your-api-key>" \
  https://your-avatar-instance.com/api/health
```

```
client = httpx.Client(
    base_url="https://your-avatar-instance.com/api",
    headers={"Authorization": "api-key-v1 <your-api-key>"}
)
response = client.get("/health")
```

Managing API Keys

List All Keys

Retrieve all API keys associated with your account:

```
GET /api-keys
```

Returns key metadata (name, creation date, expiration) but **not** the key value itself.

Get Key Details

```
GET /api-keys/{api_key_id}
```

Revoke a Specific Key

```
DELETE /api-keys/{api_key_id}
```

Revoke All Keys

Revoke every API key associated with your account:

```
DELETE /api-keys
```

Admin: Revoke Keys for Other Users

Administrators can revoke keys by user ID or specific key ID:

```
DELETE /api-keys/admin?user_id={user_id}
DELETE /api-keys/admin?api_key_id={api_key_id}
```

Expiration Policies

Duration	Days	Recommended Use
1 month	30	Temporary access, testing
2 months	60	Short-term projects
4 months	120	Quarterly rotation
1 year	365	Standard automation
10 years	3,650	Long-lived service accounts



© Octopize — All rights reserved

We recommend rotating API keys periodically. Use shorter expiration periods for keys with broader access and longer periods for tightly scoped automation.

Security Best Practices

- **Never commit API keys** to version control
- **Use environment variables** to store keys in automated pipelines
- **Rotate keys regularly** and revoke unused ones
- **Use descriptive names** so you can identify what each key is used for
- **Monitor usage** through the admin panel to detect anomalies

© Octopize — All rights reserved

06

Frequently Asked Questions (FAQ)

Frequently Asked Questions (FAQ)

> ⚠️ **Work in Progress**
> This page is under active development. Information may be incomplete or subject to change.

This section gathers answers to the most frequently asked questions about the Avatar anonymization method. Anonymization is a nuanced process influenced by legal context, dataset structure, and intended use. Below, we clarify key points and best practices to help you get the most out of the tool.

❓ Can we revert to the original individual from an anonymized dataset?

No, reversing an anonymized record to retrieve the original individual is not possible.
To qualify as anonymization, a process must be irreversible under legal standards. To achieve this, the Avatar method introduces stochastic variation during the local modeling of each synthetic record. See more in the methodology section.

❓ Can the method handle small datasets (a few dozen records)?

Yes, the method supports small datasets — but with a few caveats:

- A small number of individuals increases the risk of reidentification due to uniqueness.
  At the same privacy level, fewer statistical properties can be preserved compared to a larger dataset.
- Avatar-generated data reflects the confidence interval of the source data.
  With fewer records, this interval is wider, meaning results may vary more across iterations.

❓ Does the method support free text anonymization?

Not directly. Free text is unstructured and, as of now, lacks a clear legal framework to define what constitutes anonymous textual data.

However, it is possible to **structure the text using NLP**, anonymize the structured version with Avatar, and then regenerate text (if needed) using a language model. This ensures personal data isn't used to train any model.
We have successfully implemented this workflow in past projects — contact us to learn more.

❓ Can images be anonymized using this method?

Not at this stage. There is currently no clear legal definition for what makes an image anonymous.

© Octopize — All rights reserved

However, you can anonymize **structured data linked to pseudonymized images**, then later **reassociate** them via probabilistic matching.
We've used this framework in real-world use cases — reach out for more details.

❓ Does anonymization require user consent?

Not necessarily.
When anonymization is done properly and demonstrably, it's considered compatible with the original purpose of data collection under GDPR.

The Avatar method automatically provides **two complementary reports** (technical assessment and impact assessment) that can be shared with your DPO.
Therefore, **explicit user consent is not required** for anonymization using Avatar.

❓ How do we assess whether a means is reasonable?

The EDPB does not require a means to be impossible in the abstract. It asks whether a means is reasonably likely to be used in the real world, taking into account cost, time, technical feasibility, and the actual environment of disclosure.

The assessment must therefore consider several practical points:

- **Chain of means**: a recipient does not need to be able to execute the attack himself. If he has the technical or financial ability to call on a specialist or third party to do it, that is a reasonable means.
- **Multiplicity of actors**: the analysis is not limited to the legitimate recipient. It must include disloyal colleagues, neighbors or relatives, investigative journalists, public authorities, questionable companies, and cybercriminals.
- **No "no motivation" assumption**: the EDPB rejects the idea that anonymity can be concluded just because a potential actor has "no interest" in re-identifying someone.
- **Illegal means are not automatically irrelevant**: compliance by legitimate actors can only be presumed. One cannot conclude that a means is unreasonable simply because it may be unlawful.
- **Contractual clauses are not enough**: a contractual prohibition on re-identification is useful, but it is only a complement to technical and organizational safeguards and never sufficient by itself to establish anonymity.

In short, the question is not whether a hostile actor would be willing to invest effort, but whether such effort is realistically possible and proportionate to the value of the information. This is the central logic of the "reasonable means" test.

For the official source, see the EDPB guidance on anonymisation: EDPB Guidelines 02/2026 on anonymisation.

© Octopize — All rights reserved

❓ Can I anonymize my entire database at once for multiple use cases?

Yes, the tool supports anonymizing an entire database in one go. A database run reads a database, anonymizes every group of linked tables, and writes the result into another database.
However, we recommend splitting your anonymization projects by **use case** whenever possible.

This enables you to **fine-tune the privacy/utility balance** for each specific purpose, resulting in better-quality avatars.
Learn more in the step-by-step guide.

❓ Does the method support wide matrices?

Yes — the method can handle datasets with more variables than records. However, this configuration increases reidentification risk, as each record becomes highly unique.

At equal privacy levels, **fewer statistical properties can be preserved** in wide matrices than in datasets with more individuals than variables.

❓ Does the method reduce dataset variance?

Slightly, yes.
To protect unique records, the method automatically **centers individuals with rare values**, leading to **a small reduction in variance**.

However, weak signals — like those in **small clusters or rare events** — are preserved, ensuring the dataset remains usable for analysis and modeling.

❓ Can certain results be preserved exactly in the anonymized dataset?

Yes, the method can be configured to **preserve specific values exactly**.
This naturally impacts privacy, but that impact is **quantified by the metrics in the report**, helping you decide whether the trade-off is reasonable for your context.

❓ What level of bias does the method introduce?

Results from synthetic data typically fall within the **confidence interval of the original data**.

- **Less structured signals** (e.g., noisy or less robust patterns) tend to introduce more variability.
- **Well-structured and stable signals** are preserved almost identically.

The method involves randomness, so multiple iterations may be performed to get closer to your **key indicators** — especially when you know what to target in advance.

© Octopize — All rights reserved

❓ Why was a patient-centric model chosen for anonymization?

The patient-centric approach was chosen to ensure **transparency**, **explainability**, and **control**. Unlike generative AI models, this design lets data scientists fully understand and guide the anonymization process, reducing risks such as model misbehavior or hidden reidentification leaks. It also makes it easier to explain how data is protected to individuals. This model further enables **unique, individual-level privacy metrics** not possible in aggregate approaches.

❓ Can the method retain the characteristics of small clusters of individuals?

Yes — the method can preserve rare but non-unique patterns in the data. Unique individuals are automatically **recentered** to prevent reidentification risk, which also helps **clean** the dataset. However, if a small group shares a rare trait, you can preserve that cluster by adjusting the anonymization parameter k to be lower than the group size. This makes the method suitable for analyzing rare events or subpopulations without sacrificing privacy.

❓ When is my data considered anonymous?

Data anonymity is not a binary status — it depends on the context of use, the sensitivity of the data, and the realism of the attack scenarios that could be used against it. To assess this, you must evaluate the **residual re-identification risk** using criteria defined by the EDPB. The Avatar method supports both the **simplified approach** and the **contextual approach** described in official EDPB guidance.

The **simplified approach** is the conservative, worst-case view used for strict-sharing scenarios such as open data or broad external distribution. The **contextual approach** is the case-specific assessment used when the actual recipient, purpose, environment, and safeguards are known and controlled.

The Avatar method provides automatic risk metrics and recommended thresholds for strict use cases like open data. These metrics are consolidated into a **technical report** that quantifies the residual risk and helps document the assessment.

Learn more in the Privacy Metrics section

In addition, a **PIA (Privacy Impact Assessment)** is provided to evaluate the **likelihood of potential attacks** against the anonymization process, as well as their **impact** if successful. In simple terms, the technical report evaluates the **residual re-identification risk** posed by certain threats, while the PIA measures the **probability and severity** of those threats in the given context.

The recent EDPB guidance reinforces that a dataset can only be considered anonymous for a given use case when the residual risk remains sufficiently low and the technical and organizational safeguards are proportionate to the risk. Importantly, this is not a “100% protection on every metric” test: the question is

© Octopize — All rights reserved

whether the residual risk is acceptable in context, and whether the chosen safeguards and assumptions remain proportionate to the realistic attack model.

Where residual risk is considered too high, **additional technical and organizational safeguards** can be implemented to further strengthen the anonymization process.

❓ Can this method be used for data augmentation?

We developed parameters that allow you to generate synthetic datasets larger than the original while preserving statistical properties. You can either augment the size of your dataset or rebalance modalities for underrepresented groups. We also calculate specific privacy metrics for augmented data to ensure that reidentification risks remain acceptable. Learn more in the data augmentation section.

❓ How does the method handle missing data?

The method natively supports missing data and can preserve the same missingness patterns (MAR, MNAR, etc.) in the synthetic dataset. This is useful when the **absence of data itself carries meaning**. Alternatively, it is also possible to generate a version of the synthetic data without any missing values if needed for downstream tasks.

❓ What projection models are used in the anonymization process?

Projection models are selected automatically based on your dataset:

- **PCA** for numeric data
- **MCA** for categorical data
- **FAMD** for mixed datasets
- **FPCA** or **Fourier transforms** for time series

All chosen methods are reversible to suit the methodology described for the avatar approach. This allow the output synthetic data to have the same structural and encoding properties as the original data. Non-reversible projection methods like t-SNE or UMAP can't be used for this reason.

❓ How do you handle computed variables and business rules?

We recommend excluding computed variables (e.g., BMI) from the anonymization process **only when both the computed variable and its input components** (e.g., height and weight) are present in the dataset. In such cases, it's preferable to anonymize only the components and **recalculate the derived variable afterward**. This ensures logical consistency and reduces redundancy. Additionally, the tool includes processors to enforce business rules, such as sequential dates or fixed-sum constraints across variables.

© Octopize — All rights reserved

❓ Who is responsible in case of improper data anonymisation?

As a data processor, we provide an anonymisation tool along with a technical assessment of residual re-identification risks. This enables the data controller (i.e. the user of the tool) to make an informed decision based on the specific context of data use.

Under the General Data Protection Regulation (GDPR), the data controller remains ultimately responsible for ensuring the lawfulness and effectiveness of anonymisation. The technical report supports that decision, but the legal accountability remains with the controller.

See Article 24 GDPR – Responsibility of the controller

💡 More questions coming soon — and feel free to contact us if your specific case isn't covered here.

- Does the method support genomic data?
- Data Characteristics and Impact
- Are there examples comparing original and anonymized datasets?
- How can I anonymize a data stream?
- How do you handle anonymization in relational databases?
- How does the method anonymize time series data?
- Can GPS traces be anonymized?
- How do you handle computed variables and business rules?

© Octopize — All rights reserved

07

Support

Support

Our team will be available to support your use cases on an ongoing basis. You can contact us via:

- Email to an engineer: support@octopize.io
- Phone (refer to your Octopize contact)



If you encounter an error, the Avatar Python client automatically generates a crash report. See the Crash Reporting page for more information about what's included and how to share it with our support team.

© Octopize — All rights reserved

OCTOPIZE
MIMETHIK DATA

Octopize

Proven privacy: unlocked data.

docs.octopize.io
support@octopize.io
octopize.io

© Octopize — All rights reserved. The Avatar anonymization method is protected by patent WO2020144118.