



WHITE PAPER

Data Masking 101

A Comprehensive Guide

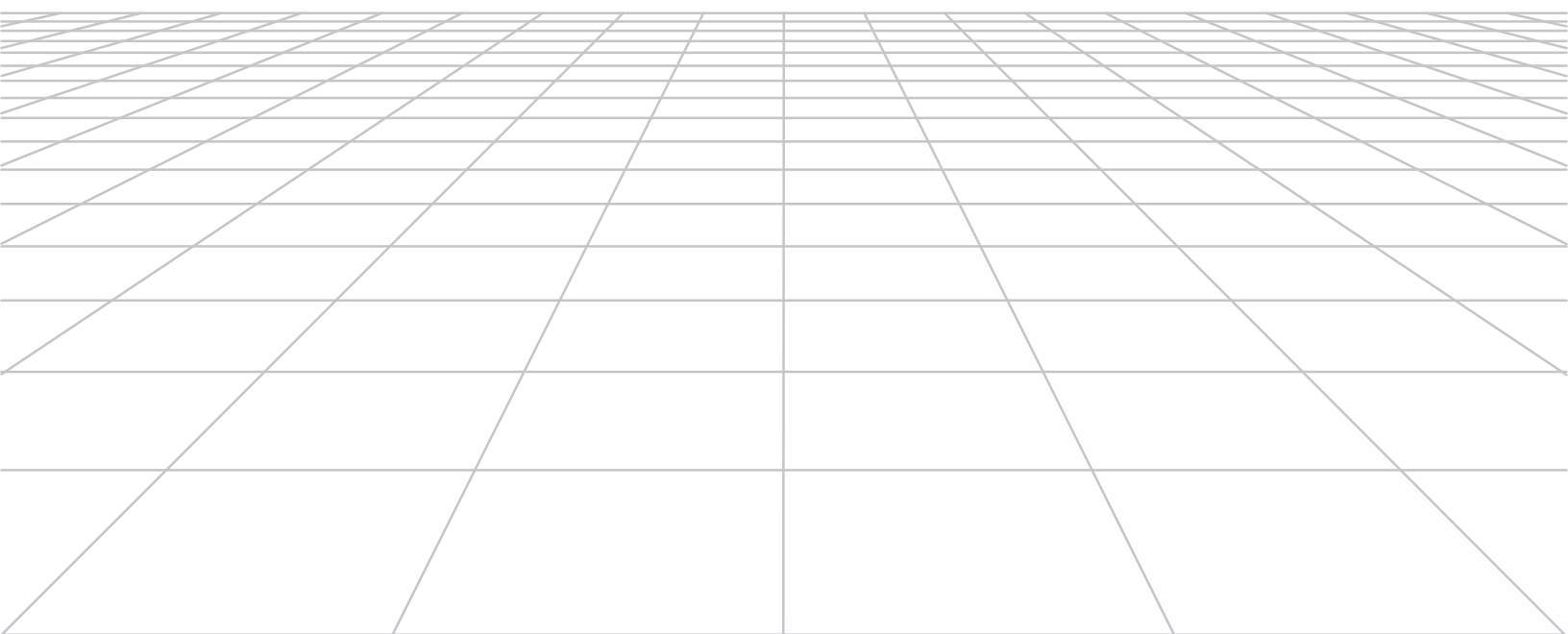


Table of Contents

Introduction	3
What is Data Masking and Why is it Important?	4
Approaches for the Application of Data Masking	6
Popular Data Masking Techniques	8
Best Practices for Masking Sensitive Data	10
How to Implement Secure and Lasting Data Masking	14

Introduction

Sensitive data is everywhere, generated and collected constantly in a world that is always online. This data can include personally identifiable information (PII) like credit card numbers, personal health information (PHI) like biometric data, and other forms of data that should remain private. When an organization stores that data for use, it has both a legal and ethical responsibility to keep sensitive information safe from a potential breach or leak.

According to Immuta's [2023 State of Data Engineering Survey](#), data and IT teams expect their organizations to house more than two-thirds (68%) of their data in the cloud by 2024, but 61% report facing challenges keeping this sensitive cloud data secure. [Similar recent studies](#) have shown that more than 64% of financial services organizations have over 1,000 sensitive data files accessible to their entire employee roster. Effectively protecting sensitive data requires balancing access and protection. Organizations can under-protect data by leaving the information overly accessible, or over-protect by completely locking data down. In either case, the data is not being protected and leveraged effectively.

Ensuring proper access to sensitive data, without compromising security or accessibility, is essential. But at an even more basic level, proactively de-identifying sensitive data within storage and compute environments establishes a base layer of security in the event of breach. Privacy enhancing technologies (PETs) like data masking make this foundational level of security achievable.

In this white paper, we've gathered a range of essential information about data masking. We break down basic definitions, masking types and techniques, best practices, and more. This guide will help you answer the question:

How can I effectively use data masking to secure my organization's sensitive data?

What is Data Masking and Why is it Important?

Data masking, also referred to as **data obfuscation**, is a form of **data access control** that alters existing sensitive information in a data set to make it unidentifiable – but still potentially usable for analytics. This process allows sensitive data to be stored and accessed, while maintaining the anonymity and safety of the information involved.

Data masking comprises a family of obfuscation techniques for controlling information disclosure. The choice of specific technique often depends upon the intended application. In determining which technique to apply, one must consider:

1. **Data Disclosure Risks:** What fields and portions of values are sensitive? Who are the downstream recipients? What could an attacker infer from this information? Who may be harmed by the release of this information and how?
2. **Analytical Use Cases vs Masking Characteristics:** Are downstream recipients and processes sensitive to formatting? Is there any information – such as the portion of a credit card number that encodes the issuing bank – that must be preserved downstream?
3. **Governance:** Do compliance and/or regulatory frameworks such as **GDPR**, **CCPA**, and **HIPAA** apply? What restrictions, if any, do these frameworks place on the use or release of the data? Can masking lower the operational classification of data processing activity, thereby reducing compliance burden or allowing for broader sharing? What masking methods are acceptable for the specified application?

Outside of its general recognition as an effective data security measure, why should you bother to implement masking techniques on sensitive data?

For one, the continued rise of data use and sharing in business and government is increasing the risk of data breaches and leaks. According to the **Identity Theft Resource Center (ITRC)**, 83% of the 1,862 data breaches in 2021 involved sensitive data. In an age where data breaches can impact organizations as large and (seemingly) secure as **Facebook** and **LinkedIn**, it is crucial that companies incorporate masking techniques into their data storage capabilities to maintain consumer safety.

Beyond general safety, customers also want to be able to trust the organizations with which they interact and get assurances that their data is being used for legitimate business purposes. There is now **wide public support** for online data privacy and intense **legislative effort** to adopt new or modernize rules in this space. Consumers now expect that the organizations to whom they disclose their personal information will act as good stewards of their data. If this trust is betrayed and data is not sufficiently safeguarded against misuse, it can severely damage consumer confidence and relationships.

Masking techniques are a must to be able to both reduce the likelihood of data breaches and achieve data minimization and purpose limitation, in particular when personal information is processed for secondary purposes such as data analytics. Not only will masking personal identifiers and/or sensitive data help your organization protect consumers' privacy through heightened confidentiality and unlinkability across processing activities, but it will help maintain a high level of trust that your company will do right with their data.

The screenshot displays the Immuta Global Data Policy Builder interface. The main section is titled "Global Data Policy Builder" and includes links for "Add Certification" and "SQL Support Matrix". The policy is named "Mask PII". The configuration is as follows:

- What is the name of this policy?** Mask PII
- How should this policy protect the data?**
 - Mask:** everyone except
 - columns tagged:** Discovered > PII (with a dropdown arrow and a close button)
 - add another tag (optional):** (empty field)
 - using hashing:** using hashing (with a dropdown arrow)
 - for:** for (with a dropdown arrow)
 - when user:** everyone except
 - is acting under purpose:** Use Case Outside De-identification (with a dropdown arrow)
 - + Add Another Condition:** (button)
 - Enter Rationale for Policy (Optional):** (text area)
 - Add:** (button)
- Where should this policy be applied?**
 - On data sources:** On data sources (with a dropdown arrow)
 - with columns tagged:** with columns tagged (with a dropdown arrow)
 - Discovered > PII:** Discovered > PII (with a dropdown arrow)

A dropdown menu for "using hashing" is open, showing the following options:

- All Masking Types
- using hashing
- with reversibility
- by making null
- using a constant
- using a regex
- by rounding
- with format preserving masking
- with K-Anonymization
- using randomized response
- using the custom function

An example of a masking policy in Immuta.

Approaches for the Application of Data Masking

The application of data masking has taken two major forms over time, each best suited for different scenarios or data environments. By examining the strengths and weaknesses of these types of data masking, you can evaluate which should best suit your organization's unique needs.

Static Data Masking (SDM)

Static data masking (SDM) masks data at rest rather than in active use. By creating a copy of an existing data set and scrubbing it of all sensitive and/or personally identifiable information (PII), this data can then be stored, shared, and accessed for use without putting sensitive information at risk.

The most important aspect of SDM is that it makes a copy of existing data. This means that the masked output of the SDM process is detached from the initial data, with no connections tying the two together. This static characteristic of the data means it will not see updates unless you create another new, more current copy.

When to Use Static Data Masking

Static data masking is best suited for environments where data is only used for a single purpose and does not change over time. Software and application development or training are examples of environments when SDM is useful. When creating a new tool or application, developers need to test their software with data that is realistic enough to be treated in the same way real data would be.

Since static data masking scrubs real data sets of all sensitive information, it strikes the balance between utility and security in a testing environment. Developers can run tests that respond in a realistic fashion without having to worry that the data could be exposed or used for the wrong reasons. However, when large data sizes and/or combinations of different levels of access are introduced, this approach becomes greatly hindered by an inability to scale with ease. Because of this, static data masking is not recommended for analytical use cases because “live” data is required, where updates are real-time available and not hindered by stagnant data.

Dynamic Data Masking (DDM)

Dynamic data masking (DDM) does not require moving or copying data. Instead, it takes the more agile approach of applying masking techniques at query time. DDM applies the same types of data masking techniques as SDM, but does so without needing to separate the data from its original source.

This maintains a single source of truth for the data set, rather than making multiple copies of scrubbed and masked data for various uses. DDM helps teams avoid the pitfalls of confusion and data silos that arise from creating many unnecessary copies of the data. Most importantly, since it is never a copy, the data remains “live” and updated, which is critical for analytical use cases.

When to Use Dynamic Data Masking

Of the types of data masking, dynamic data masking may be the most widely-applicable. Since this type of masking is actively enforced at query time, it is not limited based on where the data is stored or copied to – it is a “live” view of the data. It also allows for more complex logic across varying sets of users, since masking is applied at runtime rather than requiring the creation of a copy for every scenario like with SDM. Because of this, DDM supports more complex policy scenarios and use cases, including dynamically retaining or destroying referential integrity.

Since dynamic masking maintains a single source of truth for data sets and does not require copying data, this “live” view is perfect for analytical use cases. Compliance is also much easier to manage because many more complex policy scenarios can be handled without making hundreds of copies of your data. What’s more, instead of manually maintaining and auditing numerous copies of a data set, policy is enforced and monitored/audited in a single consistent location.

Popular Data Masking Techniques

As the types of data masking developed, different methods and techniques for carrying out the process proliferated. Whether done in a static or dynamic manner, a wide range of data masking techniques can now be used to effectively secure and protect PII in your data environment. Some of the most popular masking techniques include:

- **Replace with NULL:** This function replaces any value in a column with a NULL. When this policy is applied, the underlying data will appear to be NULL. This removes any identifiability from the column, at the cost of removing all utility. It is useful to apply this policy to numeric or text attributes which have a high re-identification risk, but little analytic value. These can include names, personal identifiers, etc.
- **Replace with Constant:** This function replaces any value in a column with a specified value. When this policy is applied, the underlying data will appear to be a constant. This masking carries the same privacy and utility guarantees as "Replace with NULL."
- **Replace with REGEX:** This function uses a regular expression replacement to replace all or a portion of an attribute. For example it may be appropriate to reveal only the first three digits of a US Zip Code, reducing the identifiability of the disclosed zip code. REGEX replacement allows for some groupings to be maintained, while providing greater ambiguity to the disclosed value. This masking technique is useful when the underlying data has some consistent structure (such as a 5 digit zip code or a 9 digit social security number or a 16 digit credit card number), the masked data represents some re-identification risk, and a regular expression can be used to mask the underlying data to be less identifiable.
- **Replace with Hashing:** This function masks the values with an irreversible hash, which is consistent for the same value throughout the data source, so you can count or track the specific values, but not know the true raw value. This is appropriate for cases where the underlying value is sensitive, but there is a need to segment the population. Such attributes could be addresses, time segments, countries, etc. It is important to note that hashing is susceptible to inference attacks based on prior knowledge of the population distribution. For example if "state" is hashed, and the dataset is a sample across the United States, then an adversary could assume that the most frequently occurring hash value is California. As such it most secure to use the hashing mask on attributes which are evenly distributed across a population
- **Mask with Reversibility:** This function also masks in a way that an authorized user can "unmask" a value, thereby revealing the value to an authorized user. "Masking with Reversibility" is appropriate when there is a need to obscure a value, while allowing an authorized user to recover the underlying value. All of the same use cases and caveats which apply to "Replace with Hashing" apply to this function, to include the inference attacks based on prior knowledge of population distribution. Additionally, reversibly masked fields can leak the length of their contents, so it is important to consider whether or not this may be an attack vector for applications involving its use.

- **Mask with Format Preserving Masking:** This function also masks using a reversible function, but does so in a way that preserves the underlying structure of a given value. This means that the length and type of a value are maintained. This is appropriate when the masked value should appear in the same format as the underlying value. Examples of this would include social security numbers and credit card numbers where “Mask with Format Preserving Masking” would return masked values consistent with credit cards, or social security numbers, respectively. The original value can also be recovered by an authorized user. There is larger overhead with this masking type, and it should only be used when format is critically valuable, such as in situations when an engineer is building an application where downstream systems validate content. In almost all analytical use cases, format should not matter, so don’t do it solely because it “looks nicer.”
- **Randomized Response:** This policy randomizes the displayed value in an effort to make the true value uncertain, but maintains some analytic utility. The randomization is applied differently to categorical and quantitative values. In both cases, the noise is tunable, meaning that the noise can be increased to enhance privacy or reduced to preserve more analytic value.
 - **Categorical Randomized Response:** Categorical values are randomized by replacing a value with some non-zero probability. Not all values are randomized, and the consumer of the data is not told which values are randomized and which ones remain unchanged. Values are replaced by selecting a different value uniformly at random from among the domain of possible values. For example, if a randomized response policy were applied to a “state” column, a person’s residency could flip from Maryland to Virginia. While this would be tragic for this person, it would provide ambiguity to the actual state of residency. This policy is appropriate, for example, when obscuring sensitive values such as medical diagnosis or survey responses.
 - **Datetime and Numeric Randomized Response:** Datetime and Numeric randomized response apply a tunable, unbiased noise term to the nominal value. This can obscure the underlying value, while reducing the impact of the induced noise in aggregate. This can be applied to sensitive numerical attributes such as salary, age, or treatment dates.
- **Rounding:** Masking via rounding rounds or truncates numeric or datetime values to a fixed precision. An example would be mapping 3.14159 to 3. This policy is appropriate when it is important to maintain some analytic value of a quantity, but not at its native precision. Rounding is applied differently between numerics and datetime.
 - **Numeric Rounding:** This policy maps the nominal value to the ceiling of some specified bandwidth.
 - **Date/Time Rounding:** This policy truncates the precision of a datetime value to some user defined precision. MINUTE, HOUR, DAY, MONTH, and YEAR are the supported precisions.
- **k-Anonymization:** Masking through **k-Anonymization** can be viewed as a type of masking policy, but it is in reality a measure of re-identification risk over a dataset. Rather than applying to a single attribute, k-Anonymization measures how many rows share a common set of values. By using a combination of “Rounding” and “NULL” masking policies over multiple columns, the dataset is masked so that the rows contain at least “K” records, where K is a positive integer. This means that attributes will only be disclosed when there are a sufficient number of observations. This provides the anonymity of crowds, so that individual rows are made indistinct from each other. This reduces the re-identification risk by making it uncertain which record corresponds to a specific person. This policy is appropriate to apply over indirect identifiers such as zip code, gender, age, etc. Each of these generally are not uniquely linked to an individual, but in combination can be associated with a single person.

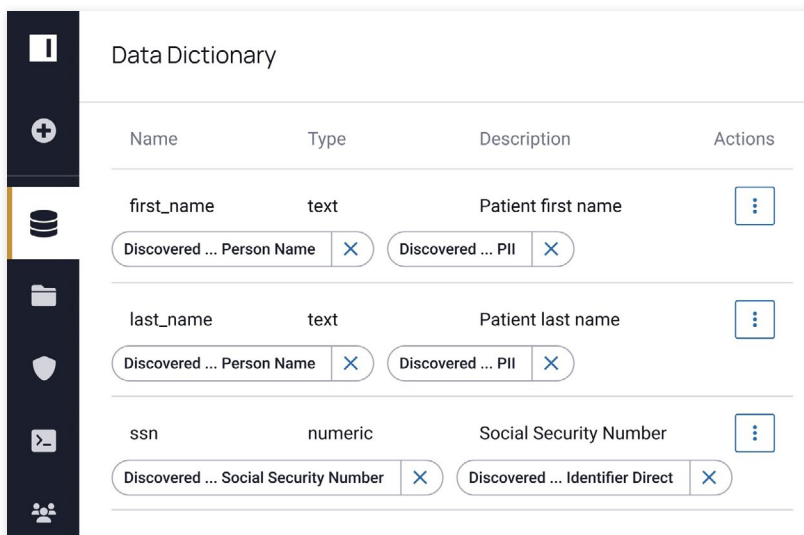
Best Practices for Masking Sensitive Data

There are certain factors that must be considered in order to choose a model that is both secure and efficient. These best practices highlight some of the key considerations you should think about while building out your data masking strategy.

Identify Your Sensitive Data

In order for masking to be effective, it's integral to understand what data exists in your storage and analysis environments. And to choose the proper masking type and technique, you need to know what kind of data you're masking. Is it credit card numbers, addresses, or BMI data in a healthcare system's data set? Each of these can be masked in ways that guarantee their security and proper compliance with the relevant **data regulations**.

The easiest way to maintain consistent, up-to-date knowledge of your data is to facilitate **data discovery** and classification as sensitive data is introduced to your data stack. This gives data teams visibility and control over the type of data in their possession, and where it is being stored and analyzed. Teams can then better understand their data in the context of the regulations they are subject to, as well as the users who need to access sensitive data. Aggregating this information helps determine the who/what/where/when/why of the masking.



The screenshot shows the Immuta Data Dictionary interface. On the left is a dark sidebar with icons for a document, a plus sign, a database, a folder, a shield, a magnifying glass, and a group of people. The main area is titled 'Data Dictionary' and contains a table with four columns: Name, Type, Description, and Actions. The table lists three data fields: 'first_name' (text, Patient first name), 'last_name' (text, Patient last name), and 'ssn' (numeric, Social Security Number). Each field has one or more discovery tags below it, such as 'Discovered ... Person Name' or 'Discovered ... PII', each with a blue 'X' icon to remove it. An Actions column with a three-dot menu icon is on the right of each row.

Name	Type	Description	Actions
first_name	text	Patient first name	
Discovered ... Person Name			
Discovered ... PII			
last_name	text	Patient last name	
Discovered ... Person Name			
Discovered ... PII			
ssn	numeric	Social Security Number	
Discovered ... Social Security Number			
Discovered ... Identifier Direct			

Sensitive Data Discovery occurs in Immuta.

Understand the Analytical Use Cases vs Masking Characteristics

In addition to understanding what you are protecting, you also need to understand how it's going to be used. This is what is commonly termed the privacy vs. utility trade off – you must decide how much utility you wish to provide from the masked data as compared to the privacy risk it entails. You should consider if your masking methods:

- **Preserve Equality and Grouping:** Does the masking function preserve equality? Yes if equal values remain equal under the masking while unequal values remain unequal. This implies that counting statistics are also preserved. Put more simply, each value will be masked to the same value consistently without colliding with others.
- **Preserve Range Statistics:** Is the number of data values falling in a particular range preserved? For strings this can be interpreted as the number of strings falling between any two values by alphabetical order.
- **Preserve Value Locality:** Does the masked value need to be near to the raw value? An example would be someone's IP address. If the IP address is used to geolocate a device, it may be necessary that the masked remain consistent with geolocation. This property may be important for analytic purposes.
- **Preserve Averages:** Is $\text{avg}(\text{mask}(v))$ expected to be near $\text{avg}(v)$?
- **Preserve Message Length:** Is the length of the masked value equal to the length of the original value?
- **Preserve Reversibility:** Does there exist a process for qualified individuals to reveal the original input value? Ignoring the fact this is always possible through policy exceptions.
- **Preserve Appearance:** Does the output masked value belong to the set of valid column values? For example, consider a masking function which outputs phone numbers when given phone numbers. Here, NULL values are not counted against this property.
- **Are Applicable to Numeric Data:** The masking function can be applied to numeric values.
- **Provide Deniability of Record Content:** A (possibly identified) person can plausibly attribute the appearance of the value to the masking function. This is a desirable property of masking functions which retain analytic utility, as such functions must necessarily leak information about the original value. Fields masked with these functions provide strong protections against value inference attacks.
- **Are Suitable for De-Identification:** Masking functions which can be used to obscure record identifiers, hiding data subject identities and preventing future linking against other identified data.
- **Allow for Column-Value Determinism:** Repeated values in the same column mask to a common output.
- **Introduce NULLs:** The masking function may, under normal or irregular circumstances, return NULL values.

TECHNIQUE	NULLing	Constant	REGEX	String Hashing	Reversible Masking	Integer Format Preserving Masking	Character-based Format Preserving Masking	Pattern Based Format Preserving Masking	Categorical Randomized Response	Numerical Randomized Response	Categorical k-Anonymization	Numerical k-Anonymization	Rounding
Preserves Equality and Grouping	✗	✗	⊖ ¹	✓	✓	✓	✓	✓	✗	✗	⊖ ²	✗	✗
Preserves Range Statistics	✗	✗	⊖	✓	✓	✓	✓	✓	⊖ ³	⊖ ⁴	✗	⊖ ⁵	⊖ ⁶
Preserves Value Locality	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓ ⁷	✓	⊖ ⁸	✓
Preserves Averages	N/A	N/A	N/A	N/A	N/A	✗	N/A	N/A	N/A	✓	N/A	✗	⊖ ⁹
Preserves Message Length	✗	✗	⊖	✗	✗	✗	✓	⊖ ¹⁰	✗	N/A	✗	N/A	N/A
Reversible	✗	✗	✗	✗	✓	✓	✓	✓	✗	✗	✗	✗	✗
Preserves Appearance	✗	✗	⊖	✗	✗	✓	✓	✓	✓	✓	✓	✓	✓
Applicable to Numeric Data	✗	✗	✗	✗	✗	✓	✗	⊖ ¹¹	✓	✓	✗	✓	✓
Provides deniability of record content	✓	✓	⊖	⊖ ¹²	✗	✗	✗	✗	✓	✓	✗	✗	✗
Suitable for De-Identification	✓	✓	⊖	⊖ ¹³	⊖ ¹⁴	⊖ ¹⁵	⊖ ¹⁶	⊖ ¹⁷	✗	✗	✓	✓	✗
Column Value Determinism	✓	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✗	✓
Introduces NULLs	✓	✗	✗	✗	✗	⊖	⊖	⊖	✓	✗	✓	✓	✗

LEGEND



Yes, the masking function has this characteristic



No, the masking function does not exhibit this characteristic.



Yes, but with some caveats and assumptions. See footnotes where appropriate.

¹ Custom Masking Functions are any function supported by the underlying database. As a result the specific characteristics of the masking are entirely dependent on the SQL function being used to mask.

² Regular expression masking is highly dependent on the replacement value.

³ k-Anonymization will preserve grouping but only for groups which have sufficiently high counts. As a result low count groups will be NULLed out

^{4, 5} Approximate value counts can be recovered via correction factor. The error is tunable by choice of privacy parameters

⁶ k-Anonymization will preserve grouping but only for groups which have sufficiently high counts. Low count groups will be NULLed out

⁷ Approximate with tunable error by choice of rounding parameters.

⁸ Yes in expectation with tunable error by choice of privacy parameters.

⁹ k-Anonymization will preserve value locality, but only for groups which have sufficiently high counts. Low count groups will be NULLed out

¹⁰ Approximate with tunable error by choice of rounding parameters.

¹¹ Yes if every value matching the pattern has the same length, otherwise no.

¹² Yes if the pattern consists only of numeric strings of digits, otherwise no.

^{13, 14, 15} Yes if values are unique to a record. Otherwise, record content may be inferible through frequency analysis.

^{16, 17, 18} Yes if values are unique to a record, and the range of format values covers all possible values.

Consider Referential Integrity

Referential integrity means that two or more tables can be joined on a common column or set of columns because the data in both sets match.

In some cases, you may want to preserve referential integrity even when data is masked. In other cases, you may want referential integrity destroyed in order to block “toxic” combinations of data that could result in privacy leaks. Masking techniques such as hashing and reversible masking provide the ability through salting and encryption keys to retain or destroy referential integrity. If this is done dynamically using DDM, it can be very powerful.

Consider Governance and its Costs

Compliance frameworks and regulation – such as GDPR, CCPA, HIPAA – may govern the handling of specific categories of information, placing restrictions on the processing and dissemination of data. It is therefore necessary to understand any applicable governance requirements.

This is important not only because frameworks often suggest or dictate masking approaches for governed categories, but also because the masking of select elements may lower the operational classification of data processing activity, thereby reducing compliance burden or allowing for broader sharing. In such cases, costly processes such as review and audit may be reduced or eliminated, lowering operational costs and time to value, and increasing the data's overall availability.

Ensure Repeatability and the Ability to Scale

One could argue that this is the most essential part of creating a lasting **data masking standard** for your organization. Data masking should be viewed as a long-term solution to protecting and securing your data from breach and ensuring compliant analytics, so solutions should therefore be implemented only if they have long-term potential.

The foundations of any masking standard should be built in a way that allows for repeatability and scalability. Masking techniques should be applicable to any new data in perpetuity, without needing to be overhauled or greatly adjusted. As data evolves and multiplies, the techniques used to protect it must be able to keep up. This means that masking techniques should be chosen and implemented only if they can be successful for your data needs both now and in the future.

How to Implement Secure and Lasting Data Masking

It can seem hard to imagine a cut-and-dry masking standard that universally meets the modern range of applications and use cases. By examining data masking's various types, techniques, and best practices against your organization's needs, you can build a system that is best suited for your goals.

Modern data use will only become increasingly complex, and masking methods must be able to maintain their effectiveness and security while allowing the flexibility for organizations to evolve and grow. Immuta's **Data Security Platform** protects data while also providing fast, secure access. Our platform automates the discovery and classification of sensitive data, makes it simple to write and enforce data policies and apply data masking and other privacy enhancing techniques, and continuously monitors data usage for risks such as insider threats.

By applying dynamic data masking at query-time, Immuta maintains privacy without slowing time-to-data or requiring unnecessary copies. PETs like k-anonymization, randomized response, encryption, and others, also provide the consistent, holistic, and scalable masking capabilities required for current and future data use.

For a hands-on look at how data masking is dynamically applied through policies, try our self-guided Immuta walkthrough demo.

About Immuta

Immuta enables organizations to unlock value from their cloud data by protecting it and providing secure access. The Immuta Data Security Platform provides sensitive data discovery, security and access control, data activity monitoring, and has deep integrations with the leading cloud data platforms. Immuta is now trusted by Fortune 500 companies and government agencies around the world to secure their data. Founded in 2015, Immuta is headquartered in Boston, MA.

