# An introduction to data privacy and anonymization:

Data privacy is a fundamental right of every individual in this day and age. With people’s online presence increasing, data privacy is important now more than ever as handling large amounts of data and protecting the rights of an individual is a complex task

Data privacy is the combined responsibility of both the users and the other party

Data Anonymization is a form of privatising the data before making it public or sending it to the intended audience.Data anonymization refers to masking sensitive user data in a way that the identity of the user is maintained and if the data is released, the data can not be traced back to the user. Data anonymization is the tradeoff between usability of the data in terms of statistical parameters and privacy of the users and to maintain the same statistical models and domain knowledge are combined to make sure that the anonymization is done properly.

## Why is data anonymization needed?

One of the biggest examples which states the importance of anonymizing the data is the 2006 AOL data search leak. Here are some interesting statistics about the data leak:

* On 4th of August 2006, search data of approximately 650,000 users along with 20 Million search results were leaked
* The data was removed relatively quickly on 7th of August 2006
* The AOL did not identify the users in the data as the names of the users were not explicitly mentioned in the data
* However, a popular newspaper magazine called New York times were able to identify the users by cross referencing them with other sources like phone book listing

Another popular data breach is that of the Netflix where the data was leaked and the researchers at the university were able to trace it back to the users

A popular data breach at the time of writing this report was that of the UNC chapel hill where the tax from info was sent to wrong people. Tax forms had a lot of sensitive info which created a lot of chaos and the privacy of the users was being sacrificed

Although proper care should be taken that the data does not get leaked in the first place, it is equally necessary to make sure that if the data gets leaked, the sensitive information is not revealed to the users.

### To ensure the privacy of the data, a five parameter framework is used:

1. Ensure that the data is safe
2. Ensure that the people working on the data are safe
3. Ensure that the scope of the project is viable
4. Ensure that the proper compliant standards are set up to ensure safety in place
5. Disclose of the output data can be monitored to ensure that sensitive data is not leaked

Simply removing the identifying attributes from the data is not enough nor is masking them as they can often be traced back to the original source.

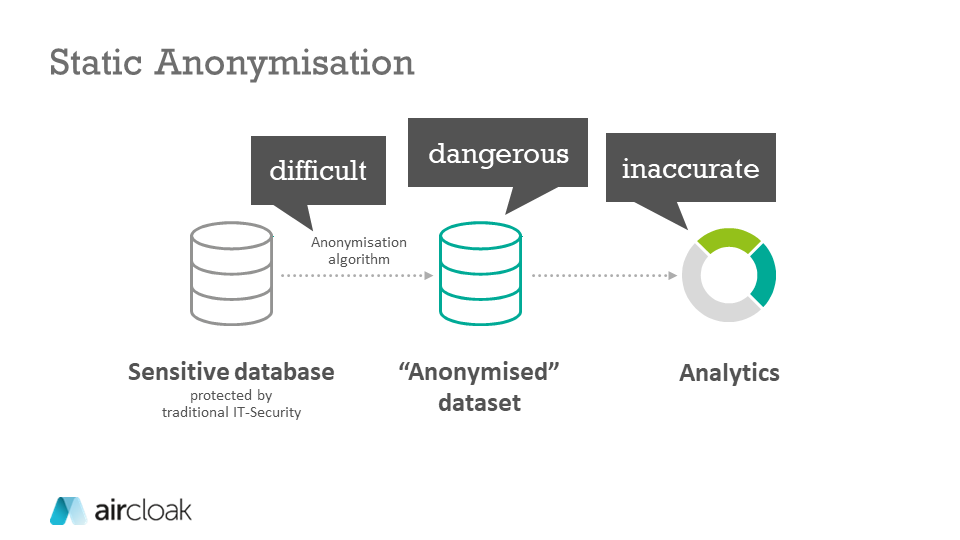
Types of anonymization

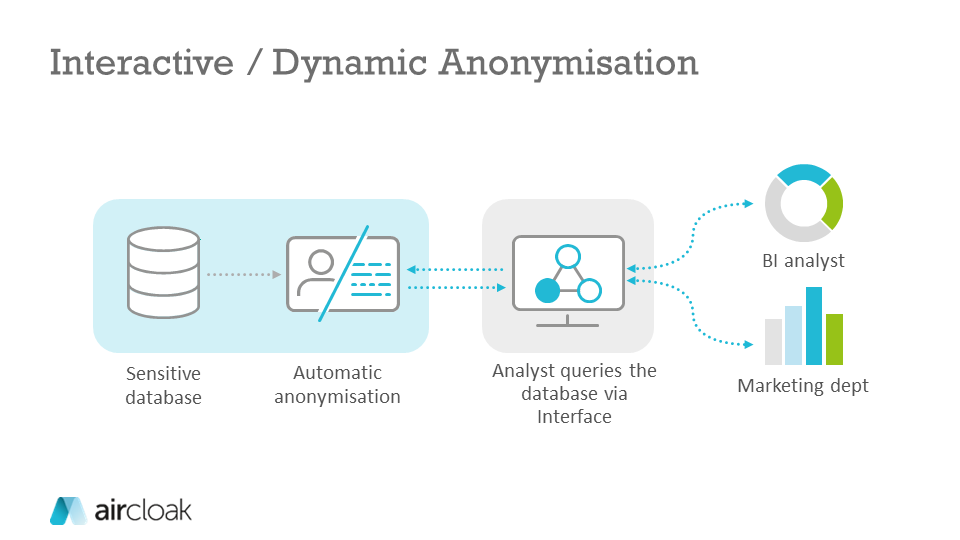
There are two types of anonymization:

1. Static
2. Dynamic

Static anonymization: Static anonymization refers to the anonymization of the data all at once and then the data is being released to the public or the third party source or vendor. In static anonymization, often a subset of the original data is released after anonymization to the users. Popular static anonymization tools and softwares are ARX, Amnesia etc

Dynamic Anonymization: Dynamic anonymization refers to the anonymization of the data using queries. Often the full dataset is released to the public and the anonymization of the dataset is happening in real-time. Dynamic anonymization is considered more reliable than the static one because static anonymization needs to specify the anonymization techniques very carefully otherwise the subset of the data can be extracted multiple times to paint the picture of the actual data. Some popular dynamic anonymization techniques are: The popular R package diffpriv, Google’s RAPPOR etc





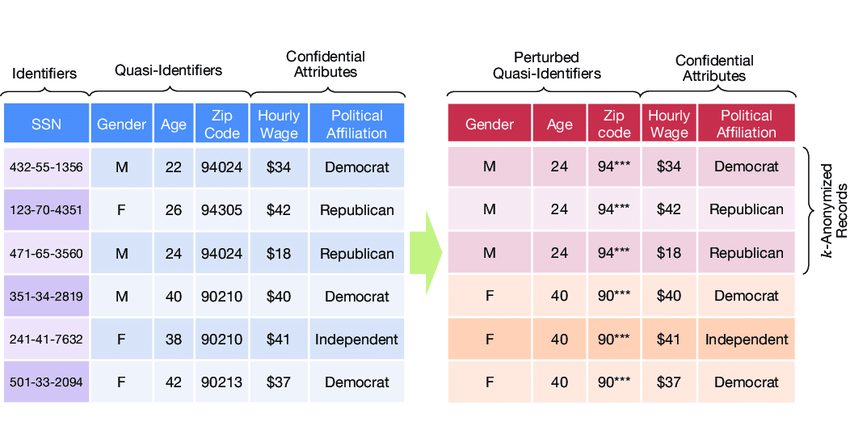


Figure 1 The process of anonymization

# The task at Hand:

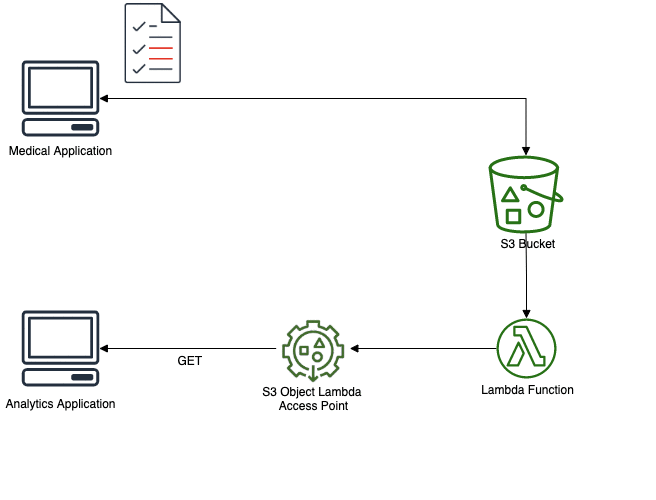
Assuming that I am working for a hospital, I have a lot of patient data with me that needs to be analysed. This data can be used to train complex machine learning models, for evaluation purposes and also understand more about the patients. However, before handing off the data to the analysts or developers, the middle man needs to come in and make sure that no sensitive information from the data is being leaked since the data might have the patient's name, address etc. The goal of the project is to anonymize the data such that no sensitive information of the patient is leaked and or can be traced back to the user AND making sure that the data is still statistically significant to draw conclusions from it. Once the solution has been implemented, the solution needs to be deployed on aws along with the database and end to end pipeline solution needs to be presented that is both effective and cost efficient.

Note: We already know the schema of the data and the format of the data is not going to change, we can expect new data but even that will follow the schema

## Proposed solution 1:

### Data Anonymization using s3 object Lambda

Technical diagram



**Explanation**

**About aws S3:**

S3 is an object based storage service from amazon that stores data in key-value pairs. S3 is often used to store large amounts of static data

**About aws lambda:**

AWS lambda is an event-driven, fully managed service by aws that lets you run your code in response to event triggers. With lambda, you do not have to worry about managing infrastructure or any other hardware, lambda lets you focus simply on your development and lambda can be used to connect to pretty much any aws service or any service outside of the aws

One of the first things that came to my mind when I heard of databases was s3. S3 is an AWS service which is an object based storage system which has a 5TB limitation for storage, hence it is virtually limitless storage. My proposed solution has a medical facing application which has the data available for access to the users which need the confidential data e.g medical professionals, in-house analysts etc. The data is then passed onto the s3 bucket where the data can have as many copies as required and then the data gets anonymized using a lambda function which has been provided additional capability using the s3 lambda access points and then the data is sent to the analytics application for the analysis.

This is a wonderful approach for the legacy data as it allows greater flexibility regarding the anonymization process and the techniques used for anonymization. Moreover, since we already know the structure of the data, having a fixed schema and static anonymization will save us time, cost and complexity.

S3 also has versioning capability which we can use to roll back to the previous version in case of disaster, s3 is also reliable with more than 99% availability and with pre planned compute and resources, we can save money as well

**Components:**

S3 bucket to store our database

Lambda function as cron job

Python script to trigger the lambda

Medical facing application

Analytics application

**Feedback from the mentor and drawbacks:**

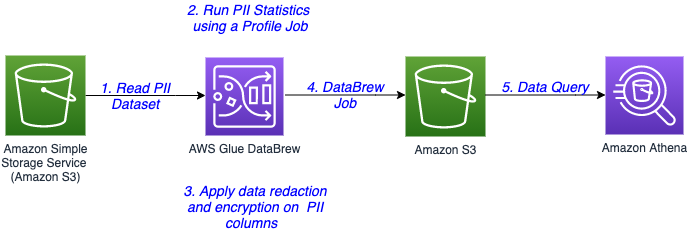
The given solution would work fantastic for data which is legacy and not updated frequently but is used a lot for analytics. E.g The census data (it is retrieved a lot but solemnly updated as it is released once every 10 years) However, for data such as medical applications or customer data or logs which are updated frequently( often multiple times a day) it is simply

Not sustainable enough. The system is also not very scalable and has a single point of failure, the architecture is linear and monolithic. Adding versioning capacity in s3 requires additional cost and Running multiple s3 buckets, a CRON job and on top of that an aws interface which lets users get access to the analytics capability is not cost optimised;

## Proposed solution 2:

Anonymization of the data using aws databrew

**Diagram**

****

**Components:**

1. **AWS S3:** As explained in the previous solution, s3 is an object based service used to store the data
2. **AWS Glue DataBrew:** Is a service by amazon that aids data scientists and machine learning engineers in cleaning and normalising the data. It has a lot of pre-built functions known as jobs for cleaning and transforming the data. You can also transform the data manually in their excel-like console and save the steps to transforming the data as a “job” so that the process can be automated
3. **AWS Athena**: Amazon Athena is a serverless, interactive analytics service built on open-source frameworks, supporting open-table and file formats. Athena provides a simplified, flexible way to analyse petabytes of data where it lives. Analyse data or build applications from an Amazon Simple Storage Service (S3) data lake and 25-plus data sources, including on-premises data sources or other cloud systems using SQL or Python. Athena is built on open-source Trino and Presto engines and Apache Spark frameworks, with no provisioning or configuration effort required.

**Explanation:**

The solution to this is fairly straightforward as the data in question to be anonymized is stored in an s3 bucket, we can enable versioning in s3 to make sure that the data is safe. Moreover, macie can be used to detect sensitive data from the already present data. The data from s3 is then loaded into the aws glue data brew which has jobs which mask the sensitive PII column-wise and after the data is masked, the masked data is then stored into s3 which creates a external table on top of athena to query and interact with the dataset

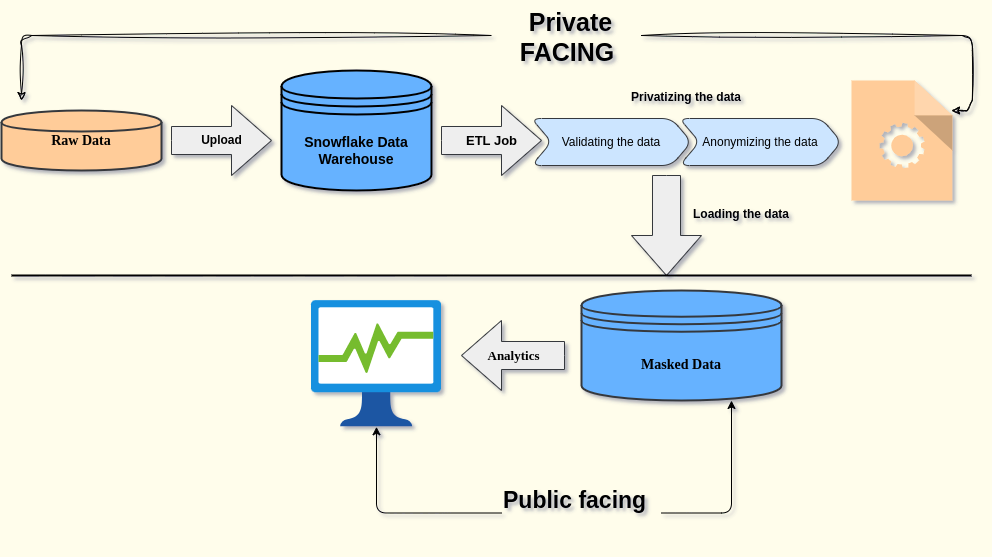
**Feedback from the mentor and drawbacks:**

Using aws macie to detect sensitive data might work for use cases where the user might have accidentally entered the sensitive data such as credit card information inside the s3 bucket. In our use case, the data that we want to mask is just not sensitive data, but the data that can be used in conjunction with other data to identify the users as well which is something where macie fails. There also might be legal and ethical concerns with macie. Additionally, the aws data brew has limited support for masking the sensitive data and if the attacker knows the method or the technique using which the data was masked, it might be easy for the attacker to re-identify the data. Overall, we need a solution that is fast, has versioning capability, is able to handle legacy data and also uses a predefined algorithm to anonymize a data with a known schema while being cost optimised

## Proposed solution 3

### Data Anonymization Pipeline using Snowflake and airflow

Diagram



**Components:**

**Snowflake:** Snowflake is a SaaS data warehouse solution designed solely for the cloud, it supports popular cloud providers like gcp, aws and azure. The snowflake can be used to build data warehouse and data lake right from your browser as snowflake is a managed service

**PYARXAAS:** Is a popular python module which is used as a static anonymization tool. Pyarxaas provides a wrapper to access functions for your local arx instance.

**Apache Airflow:** Apache airflow is an open source, etl tool that is used to automate the data loading and data transforming pipeline

**Explanation:**

The following solution provides an end to end pipeline for anonymizing data that gives users the flexibility to scale the solution as per their demand or data. Here is the process of pipeline:

1. Takes in the raw data either from the source or from a sql database. What makes the solution truly customizable is that snowflake can take in data from multiple sources Snowflake natively supports AVRo, Parquet, CSV, JSON and ORC hence data from pretty much any source can be taken
2. The raw data is then loaded into the snowflake to create a data warehouse. The access to the given snowflake is heavily restricted. This step is also customizable and scalable as if you want to give certain users access to the raw data warehouse, you can simply assign their role access and they can be managed using snowflake organisation. Thus, reducing the costs and increasing the flexibility
3. After that, an ETL job is run on the data warehouse which validates the data if need, sets the hierarchy of anonymization for the data and actually anonymizes the data. The anonymization being used here is a function written manually and will be triggered after a certain time when the data is loaded in the data warehouse. Here, since we already know the project columns and all the details about the data being present inside we will use static anonymization instead of dynamic but dynamic anonymization can be used as well
4. The anonymized data is then stored in another snowflake data table or this can be also be connected to the cloud provider of your choice but an interesting thing about snowflake is that it really lets you create stunning visualisations using the snowflake console itself
5. The masked data which is not public facing can be used to generate charts, analysis and visualisation

**Comments from the mentor and drawbacks:**

One of the biggest advantages of this solution is the scalability in terms of compute and storage. SInce snowflake is a hybrid between the shared nothing and shared everything architecture, it is really easy to scale up or down. The manual data anonymization gives the users greater flexibility to implement their own algorithm. However, one potential challenge to solve is trying to anonymize the data automatically without having the schema of the data. This is still a challenge since pyarxaas is a static anonymization algorithm

### References:

<https://aws.amazon.com/blogs/big-data/introducing-pii-data-identification-and-handling-using-aws-glue-databrew/>

<https://aircloak.com/data-anonymisation-software-differences-between-static-and-interactive-anonymisation/>