

Lomas: The Hidden Data Oasis behind the Mist

The Lomas Service

Lomas aims to empower research, inform policy development and drive innovation across sectors, all while upholding the highest standards of data confidentiality.

It is a novel **open-source platform** designed to realize the potential of data held by public administrations, developed by the Data Science Competence Center (DSCC) of the Swiss Federal Statistical Office. It enables authorized users, such as approved researchers and government analysts, to **execute algorithms on datasets without directly accessing the data**.

Differential Privacy (DP)

DP is a framework for **releasing data products while controlling the risk of disclosure of personal data**.

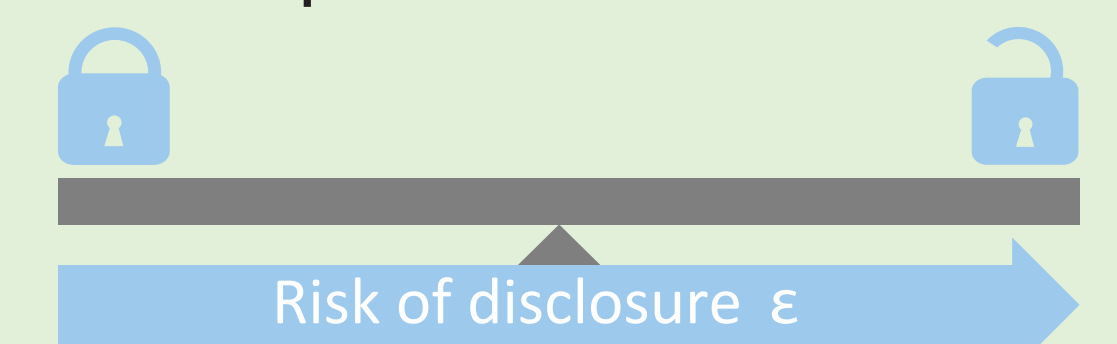
Pure DP Mechanism

$$sens = \max |q(S) - q(S')|$$
$$Q(S) = q(S) + Lap\left(0, \frac{sens}{\epsilon}\right)$$

Formal Guarantee

$$\frac{\Pr[M(S) \in O]}{\Pr[M(S') \in O]} \leq e^\epsilon$$

- $q(S)$: result of query q on dataset S
- $Q(S)$: mechanism Q on dataset S
- S' : dataset S plus or minus one record
- O : an output



Eyes-Off Analysis with Differential Privacy

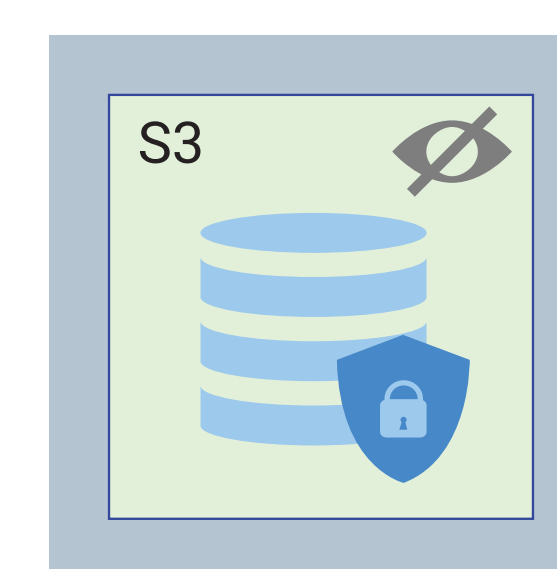
Users **NEVER** see the private data. They can get DP protected results by sending requests to the server using one of the available DP libraries. To prepare their DP algorithm, they can retrieve metadata and dummy dataframes (randomly generated to simulate the sensitive data) through a request to the server.

To query a private dataset, a set of conditions must be met:

- The user is authorized to query the dataset and has enough remaining privacy-loss budget to make the query
- The query is executed by one of the available DP libraries:
 - **Smartnoise-SQL** for SQL-like queries
 - **Smartnoise-Synth** for generating synthetic datasets
 - **OpenDP** for summary statistics
 - **DiffPrivLib** for training Machine Learning models

Users can also perform other types of requests to get information about their privacy-loss budget and get the parameters and results of past queries.

Data tier

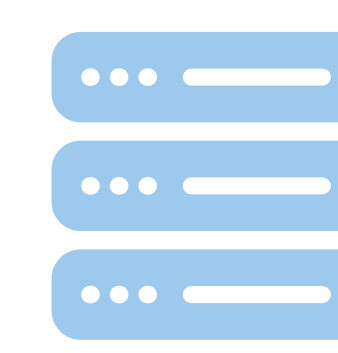


Penguin Dataset	
island	bill_length
A	35.7
B	36.1
A	39.7
...	...

Stored private datasets

Data connectors allowing the server application to retrieve private datasets from various sources

Logic tier



Application server



Administration database server

API allowing communication between the server application and client applications

Client tier



Lomas' Disclosure Control Logic

Administration database

Store information about users and datasets

Metadata of dataset:

```
{ "max_ids": 1,
  "columns": {
    "island": {
      "type": "string", "cardinality": 2,
      "categories": ["A", "B"]
    },
    "bill_length": {
      "type": "float",
      "lower": 30.0, "upper": 65.0
    }
  }
}
```

Data source:

```
{ "dataset_name": "PENGUIN",
  "database_type": "S3_DB",
  "s3_bucket": "animals",
  "s3_key": "penguins.csv",
  "endpoint_url": "http://minio:9000"
}
```

User (access and budget):

```
{ "user_name": "Dr. Antartica",
  "datasets_list": {
    "dataset_name": "PENGUIN",
    "initial_epsilon": 10.0,
    "initial_delta": 0.005,
    "total_spent_epsilon": 0.0,
    "total_spent_delta": 0.0
  }
}
```

Archives (past queries)

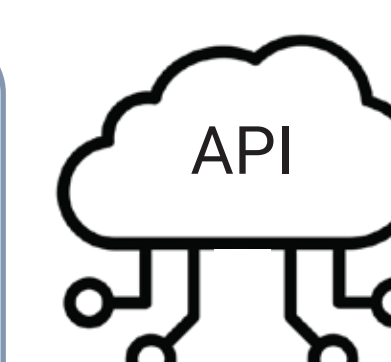
Server application

Receive HTTP requests from the client, process them and execute DP queries

```
app = FastAPI(lifespan=lifespan)
```

```
@app.post("/get_dummy")
• Check user access to dataset
• Get metadata from AdminDB (metadata)
• Make dummy df based on metadata
```

```
@app.post("/dp_query")
1. Check user access to dataset
2. Compute budget of pipeline (metadata)
3. Check user has enough budget
4. Get private data using data source
5. Apply DP query (the result is DP)
6. Update user remaining budget
7. Save query and response in archives
8. Return response to user via API
```



Request

Response

/get_dummy

dummy_df

/dp_query

mean

Client application

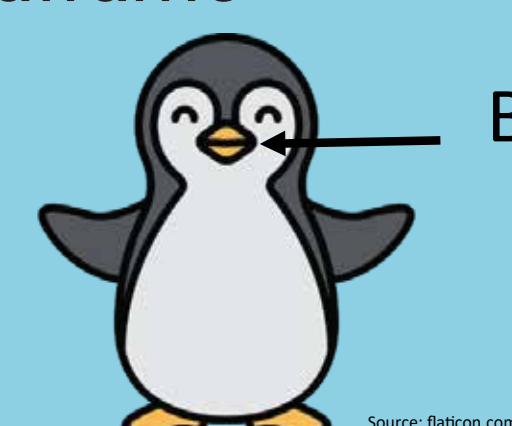
Preprocess user input, send request, receive and postprocess response

```
from lomas_client import Client
client = Client(
    url, Ice, PENGUIN
)
response = client.query(param)
```

```
dummy = client.get_dummy(nb_row=4)
```

island	bill_length
B	34.8
A	36.4
B	34.8
B	60.8

Dummy Penguin Dataframe



```
import opendp as dp
import polars as pl
plan = dummy.select(
    pl.col("bill_length").dp.mean()
)
mean = client.opendp_query(plan)
```