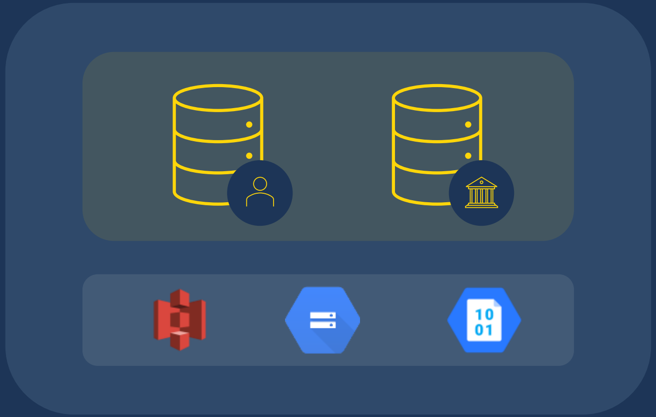
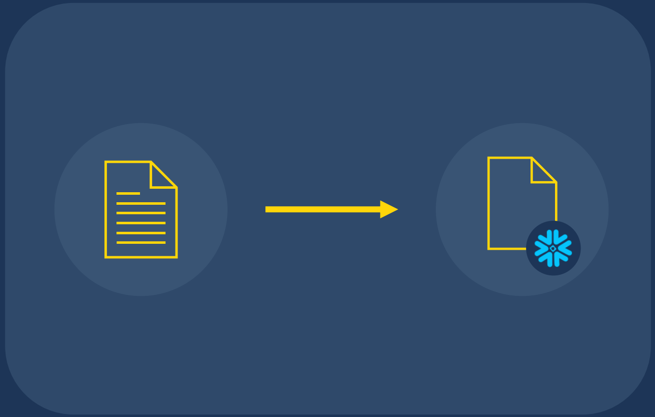
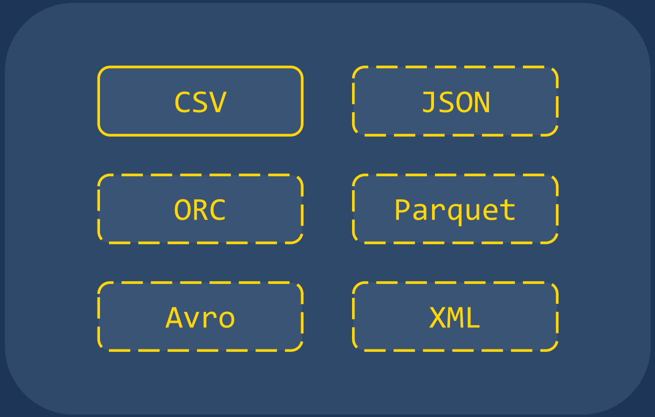
# Storage Layer Overview

# Storage Summary







CSV

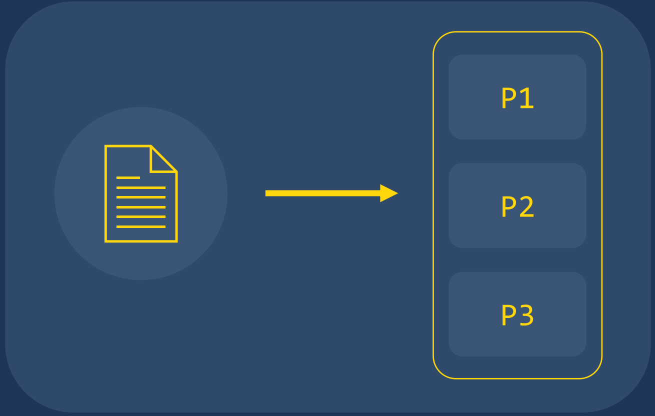
JSON

ORC

Parquet

Avro

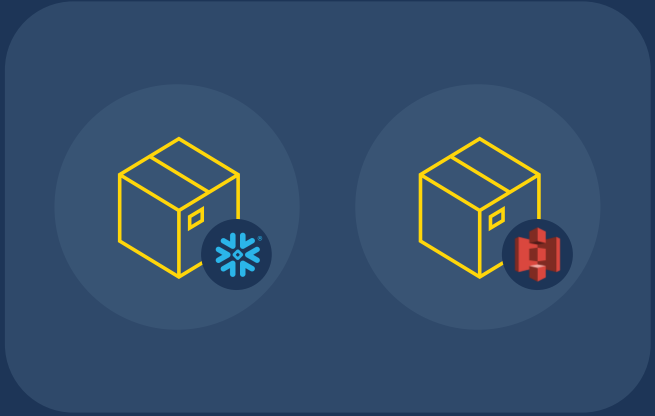
XML



P1

P2

P3

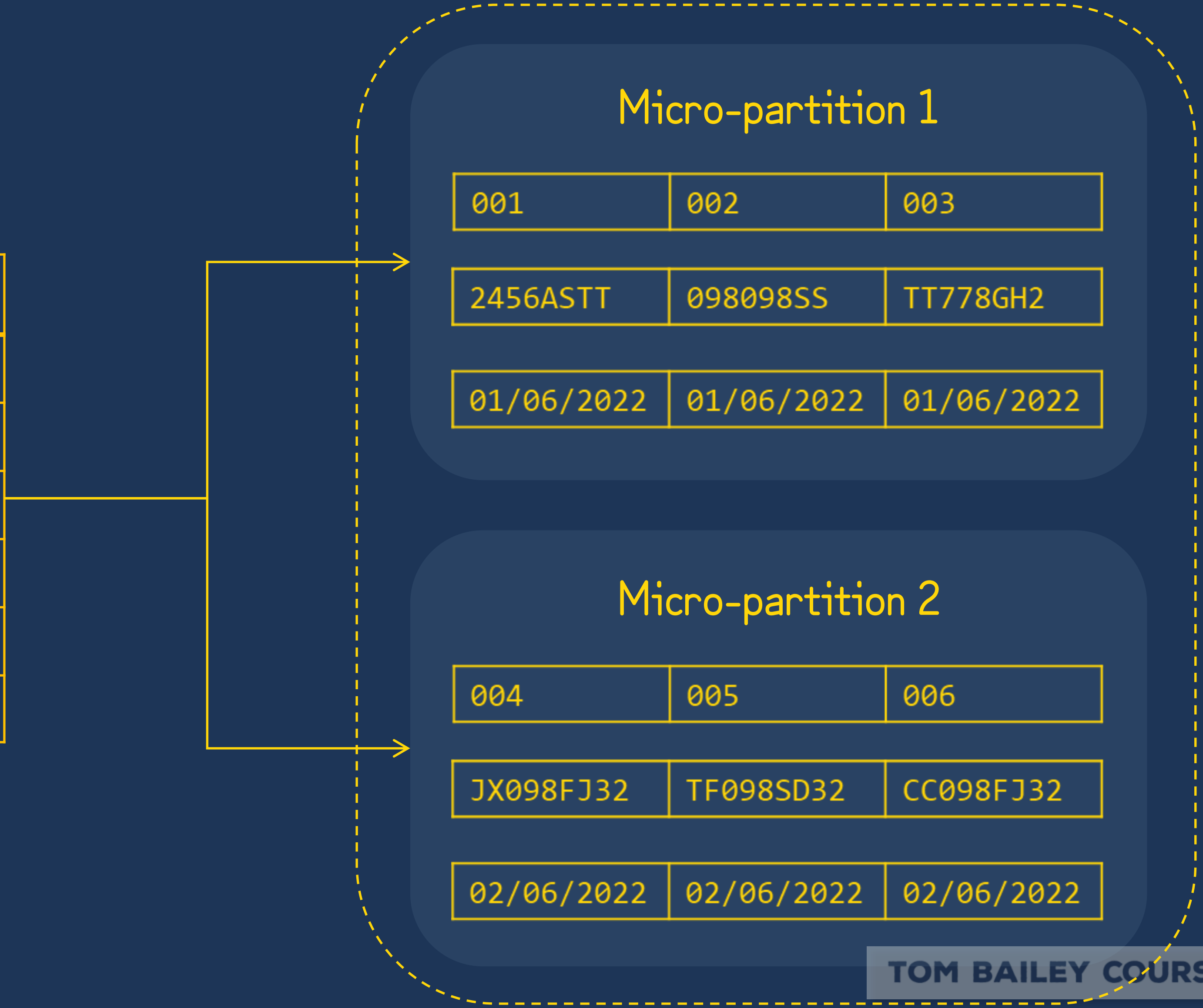


# Micro-partitions

# Micro-partitions

| order_id | item_id | order_date |
|---|---|---|
| 001 | 2456ASTT | 01/06/2022 |
| 002 | 098098SS | 01/06/2022 |
| 003 | TT778GH2 | 01/06/2022 |
| 004 | JX098FJ32 | 02/06/2022 |
| 005 | TF098SD32 | 02/06/2022 |
| 006 | CC098FJ32 | 02/06/2022 |

```
COPY INTO MY_CSV_TABLE FROM
@MY_STAGE/orders.csv
```



Micro-partition 1

| | | |
|---|---|---|
| 001 | 002 | 003 |
| 2456ASTT | 098098SS | TT778GH2 |
| 01/06/2022 | 01/06/2022 | 01/06/2022 |

Micro-partition 2

| | | |
|---|---|---|
| 004 | 005 | 006 |
| JX098FJ32 | TF098SD32 | CC098FJ32 |
| 02/06/2022 | 02/06/2022 | 02/06/2022 |

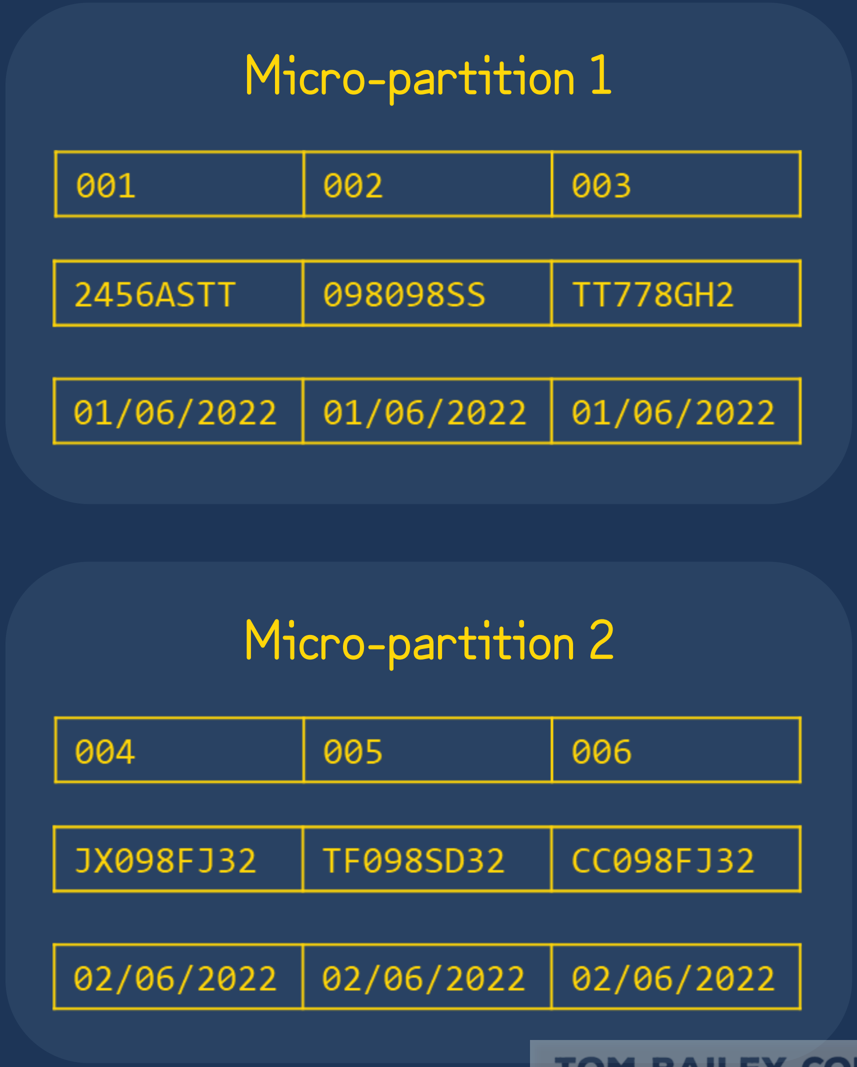
# Micro-partitions

Snowflake partitions along the natural ordering of the input data as it is inserted/loaded.

Micro-partitions are the physical file stored in blob storage and they range in size from **50-500mb** of uncompressed data.

Micro-partitions undergo a reorganisation process into the Snowflake columnar data format.

Micro-partitions are immutable, they are write once and read many.



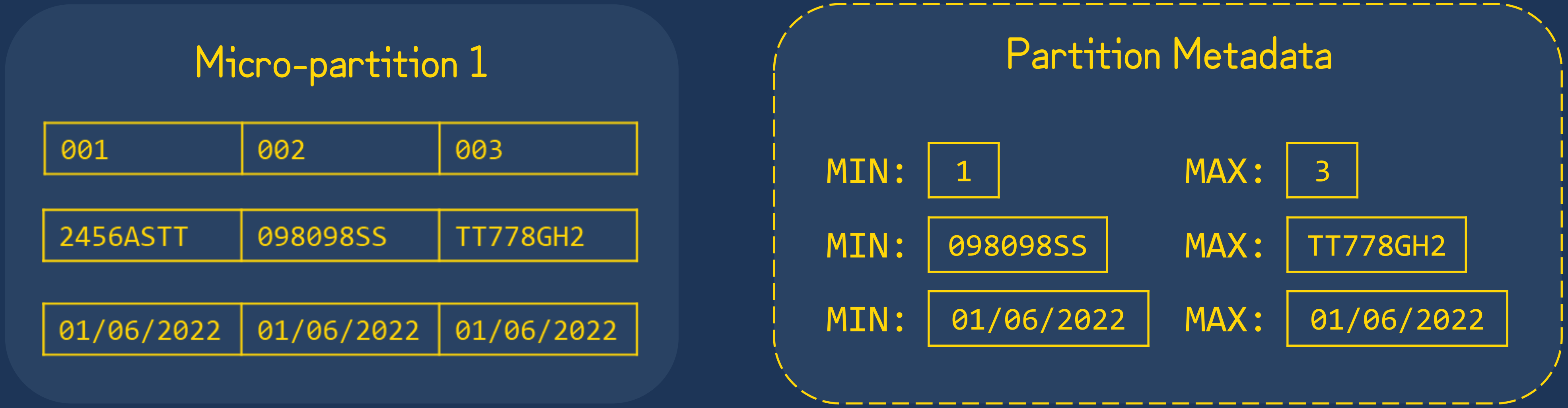
## Micro-partition 1

| | | |
|---|---|---|
| 001 | 002 | 003 |
| 2456ASTT | 098098SS | TT778GH2 |
| 01/06/2022 | 01/06/2022 | 01/06/2022 |

## Micro-partition 2

| | | |
|---|---|---|
| 004 | 005 | 006 |
| JX098FJ32 | TF098SD32 | CC098FJ32 |
| 02/06/2022 | 02/06/2022 | 02/06/2022 |

# Micro-partition Metadata



Micro-partition 1

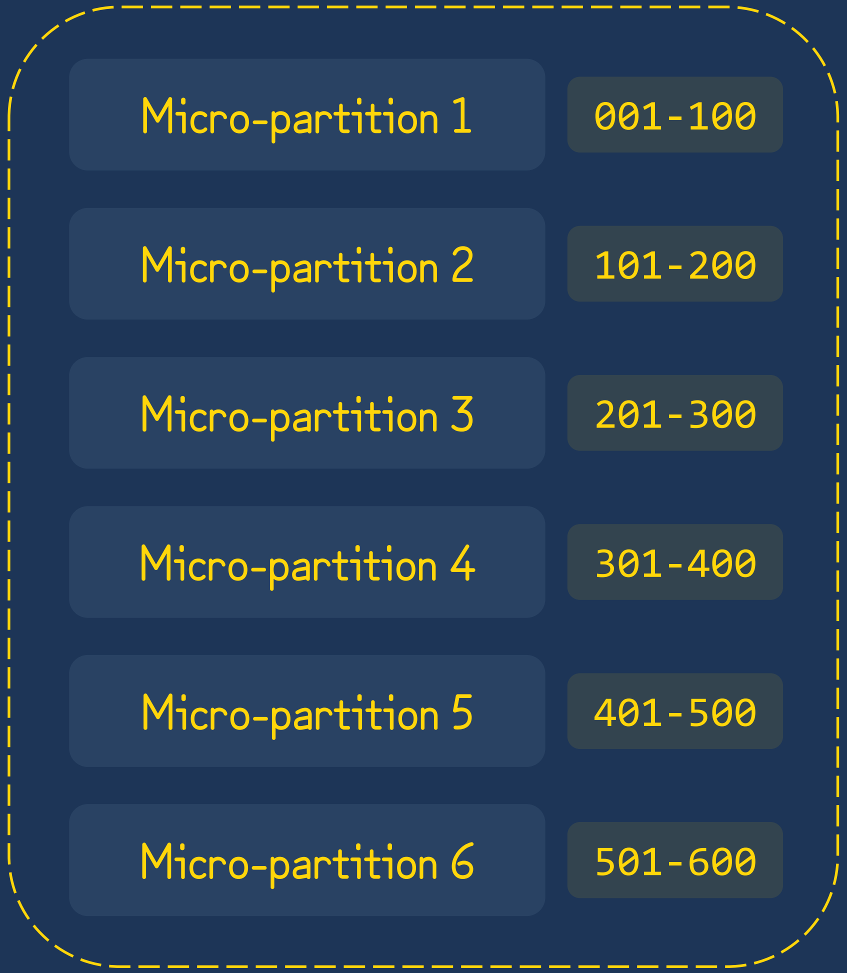
| | | |
|---|---|---|
| 001 | 002 | 003 |
| 2456ASTT | 098098SS | TT778GH2 |
| 01/06/2022 | 01/06/2022 | 01/06/2022 |

Partition Metadata

| | | | |
|---|---|---|---|
| MIN: | 1 | MAX: | 3 |
| MIN: | 098098SS | MAX: | TT778GH2 |
| MIN: | 01/06/2022 | MAX: | 01/06/2022 |

# Micro-partition Pruning

MY_CSV_TABLE



| | |
|---|---|
| Micro-partition 1 | 001-100 |
| Micro-partition 2 | 101-200 |
| Micro-partition 3 | 201-300 |
| Micro-partition 4 | 301-400 |
| Micro-partition 5 | 401-500 |
| Micro-partition 6 | 501-600 |

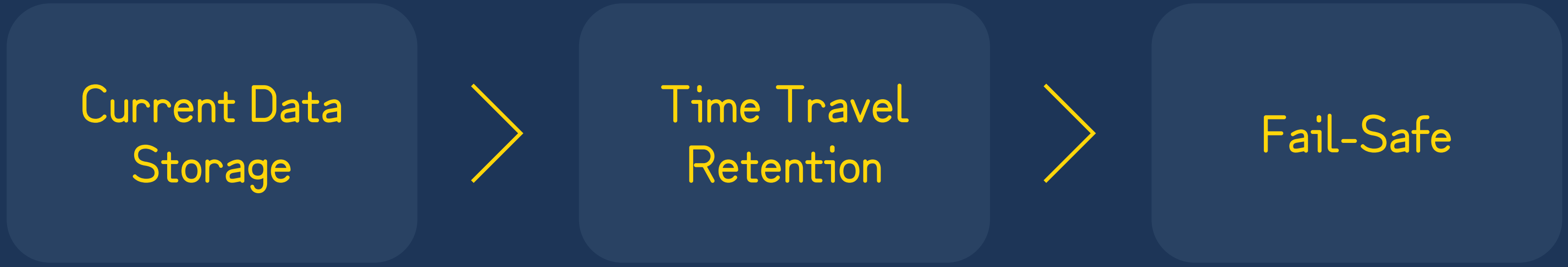
Micro-partition metadata allows Snowflake to optimize a query by first checking the min-max metadata of a column and discarding micro-partitions from the query plan that are not required.

```
SELECT ORDER_ID, ITEM_ID
FROM MY_CSV_TABLE
WHERE ORDER_ID > 360 AND ORDER_ID < 460;
```

The metadata is typically considerably smaller than the actual data, speeding up query time.

# Time Travel & Fail-Safe

# Data Lifecycle

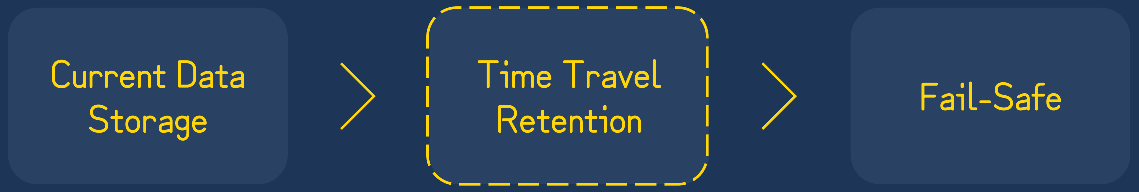


Current Data Storage

Time Travel Retention

Fail-Safe

# Time Travel



Time Travel enables users to restore objects such as tables, schemas and databases that have been removed.

```
UNDROP DATABASE MY_DATABASE;
```

Time Travel enables users to analyse historical data by querying it at points in the past.

```
SELECT * FROM MY_TABLE AT(TIMESTAMP =>
TO_TIMESTAMP('2021-01-01'));
```

Time Travel enables users to create clones of objects from a point in the past.

```
CREATE TABLE MY_TABLE_CLONE CLONE MY_TABLE
AT (TIMESTAMP => TO_TIMESTAMP('2021-01-01'));
```

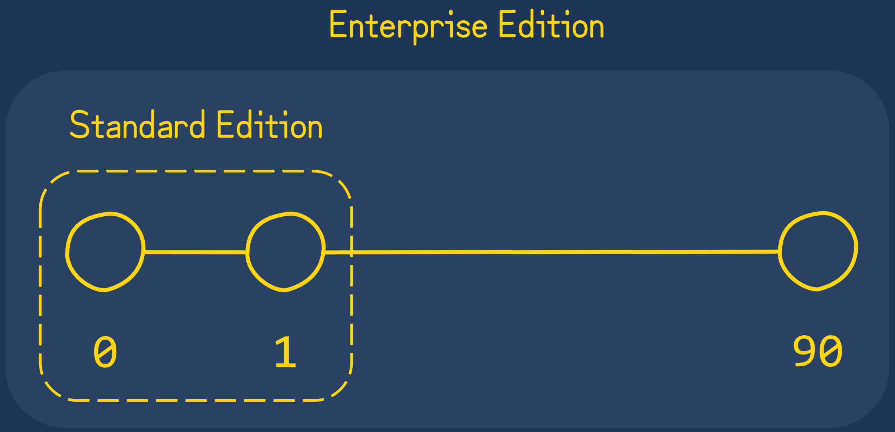
# Time Travel Retention Period

The Time Travel retention period is configured with the parameter: DATA_RETENTION_TIME_IN_DAYS.

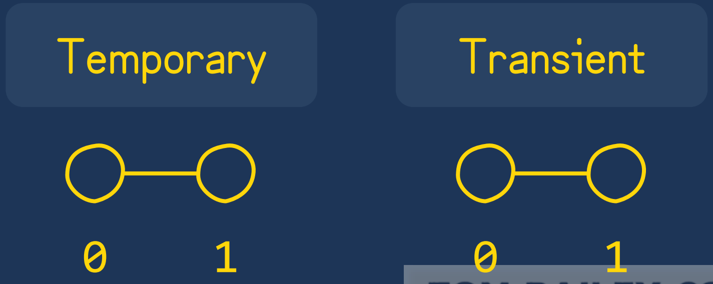
```
ALTER DATABASE MY_DB
SET DATA_RETENTION_TIME_IN_DAYS=90;
```

The default retention period on the account, database, schema and table level is **1 day**.

On the Standard edition of Snowflake the minimum value is 0 and maximum is 1 day and for Enterprise and higher the maximum is increase from 1 to 90.



Temporary and transient objects can have a max retention period of 1 day across all editions.



# Accessing Data In Time Travel

## AT

```
SELECT * FROM MY_TABLE
AT(STATEMENT =>
'01a00686-0000-0c47');
```

The AT keyword allows you to capture historical data inclusive of all changes made by a statement or transaction up until that point.

Three parameters are available to specify a point in the past:

- TIMESTAMP
- OFFSET
- STATEMENT

## BEFORE

```
SELECT * FROM MY_TABLE
BEFORE(STATEMENT =>
'01a00686-0000-0c47');
```

The BEFORE keyword allows you to select historical data from a table up to, but not including any changes made by a specified statement or transaction.

One parameter is available to specify a point in the past:

- STATEMENT

## UNDROP

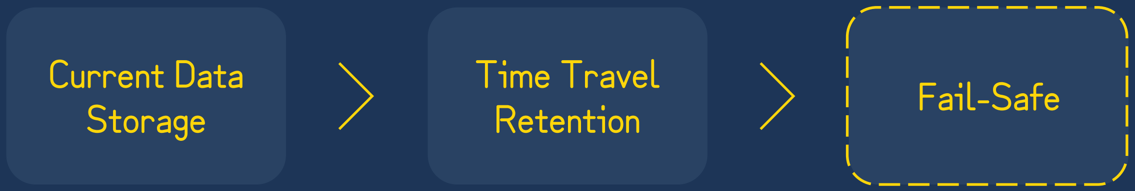
```
UNDROP TABLE MY_TABLE;
UNDROP SCHEMA MY_SCHEMA;
UNDROP DATABASE MY_DATABASE;
```

The UNDROP keyword can be used to restore the most recent version of a dropped table, schema or database.

If an object of the same name already exists an error is returned.

To view dropped objects you can use:
SHOW TABLES HISTORY;

# Fail-safe

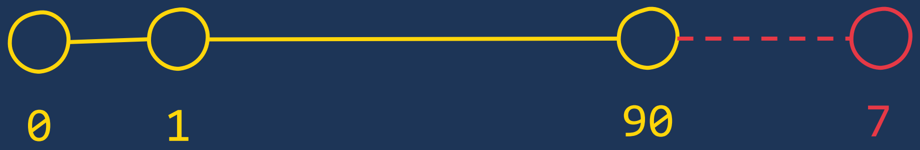


Current Data Storage

Time Travel Retention

Fail-Safe

Fail-safe is a non-configurable period of 7 days in which historical data can be recovered by contacting Snowflake support.



It could take several hours or days for Snowflake to complete recovery.

Fail-safe is only enabled for permanent objects, not temporary or transient.

# Cloning

# Cloning

```
CREATE TABLE MY_TABLE (
COL_1 NUMBER COMMENT 'COLUMN ONE',
COL_2 STRING COMMENT 'COLUMN TWO',
);

CREATE TABLE MY_TABLE_CLONE CLONE MY_TABLE;
```

```
CREATE STAGE MY_EXT_STAGE
  URL='S3://RAW/FILES/'
  CREDENTIALS=();

CREATE STAGE MY_EXT_STAGE_CLONE CLONE
MY_EXT_STAGE;
```

```
CREATE FILE FORMAT MY_FF
TYPE=JSON;

CREATE FILE FORMAT MY_FF_CLONE CLONE MY_FF;
```

Cloning is the process of creating a copy of an existing object within a Snowflake account.

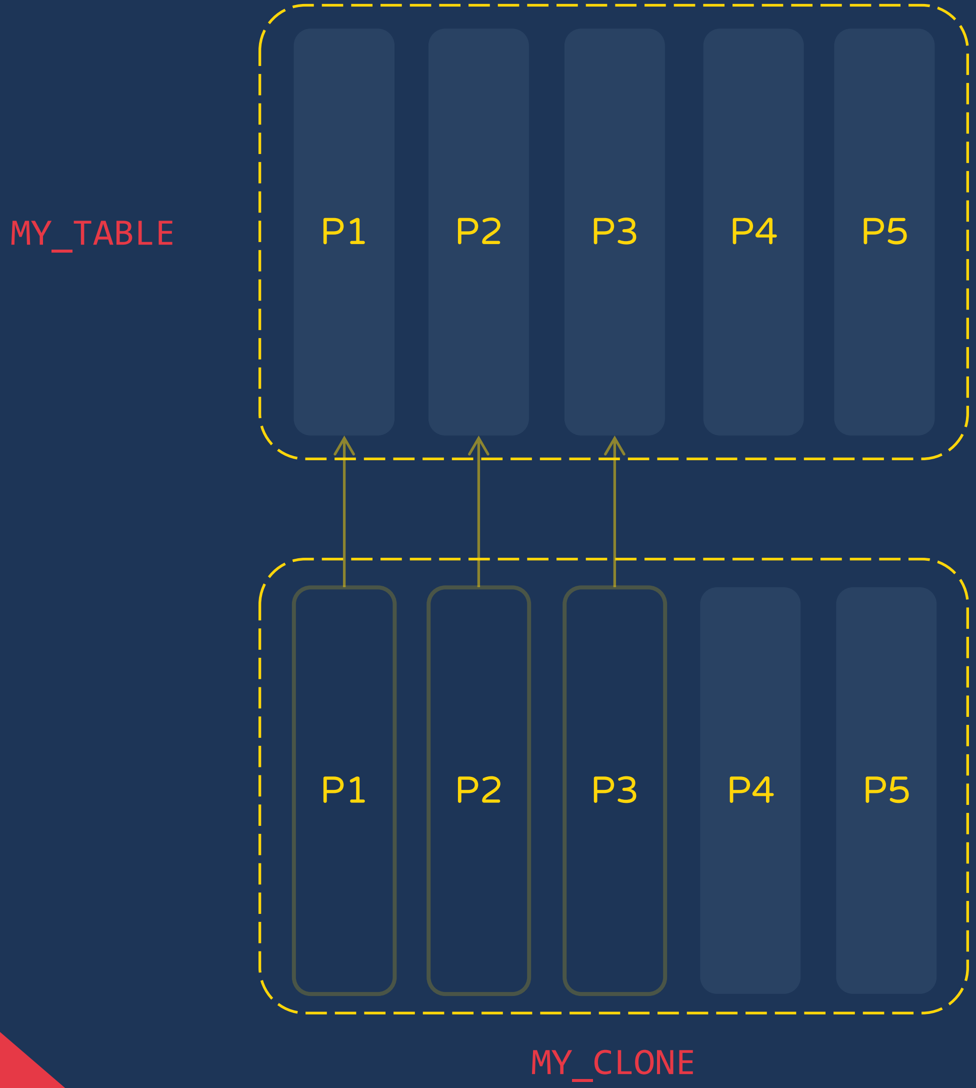
Users can clone:

- DATABASES
- SCHEMAS
- TABLES
- STREAMS
- STAGES
- FILE FORMATS
- SEQUENCES
- TASKS
- PIPES (reference external stage only)

Cloning is a metadata only operation, copying the properties, structure and configuration of it's source.

Cloning does not contribute to storage costs until data is modified or new data is added to the clone.

# Zero-copy Cloning



```
CREATE TABLE MY_CLONE CLONE MY_TABLE;
```

Changes made after the point of cloning then start to create additional micro-partitions.

Changes made to the source or the clone are not reflected between each other, they are completely independent.

Clones can be cloned with nearly no limits.

Because cloning is a meta-data only operation it's very quick, enabling interesting use-case, such as rapid integration testing.

# Cloning Rules

1

A cloned object does not retain the privileges of the source object, with the exception of tables.

2

Cloning is recursive for databases and schemas.

3

External tables and internal named stages are never cloned.

4

A cloned table does not contain the load history of the source table.

5

Temporary and transient tables can only be cloned as temporary or transient tables, not permanent tables.

# Cloning With Time Travel

```
CREATE TABLE MY_TABLE_CLONE CLONE MY_TABLE
AT (TIMESTAMP => TO_TIMESTAMP(‘2022-01-01'));
```

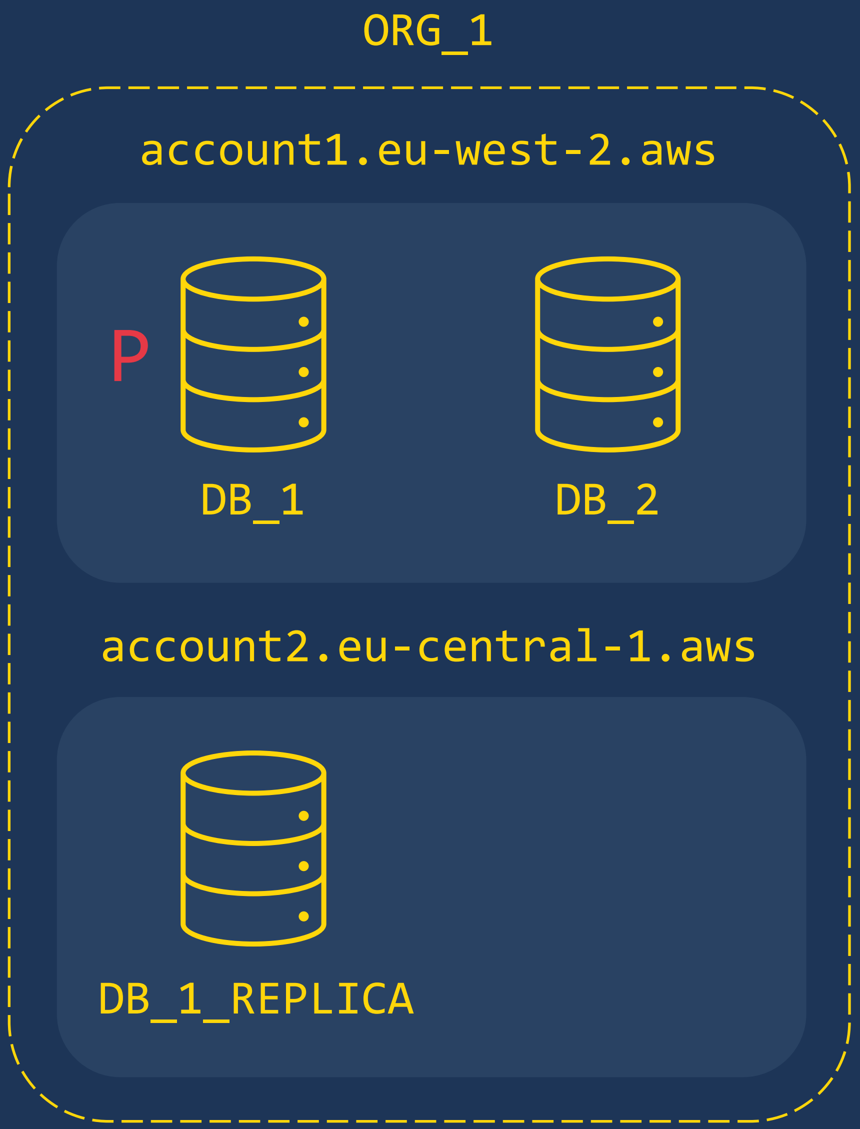
Time Travel and Cloning features can be combined to create a clone of an existing database, schema non-temporary table and stream at a point within their retention period.

```
CREATE TABLE MY_TABLE_CLONE CLONE MY_TABLE
AT (OFFSET => -60*30);
```

If the source object did not exist at the time specified in the AT | BEFORE parameter an error is thrown.

# Replication

# Replication



Replication is a feature in Snowflake which enables replicating databases between Snowflake accounts within an organization.

A database is selected to serve as the primary database from which secondary databases can be created in other accounts:

```
ALTER DATABASE DB_1
ENABLE REPLICATION TO ACCOUNTS ORG1.account2;
```

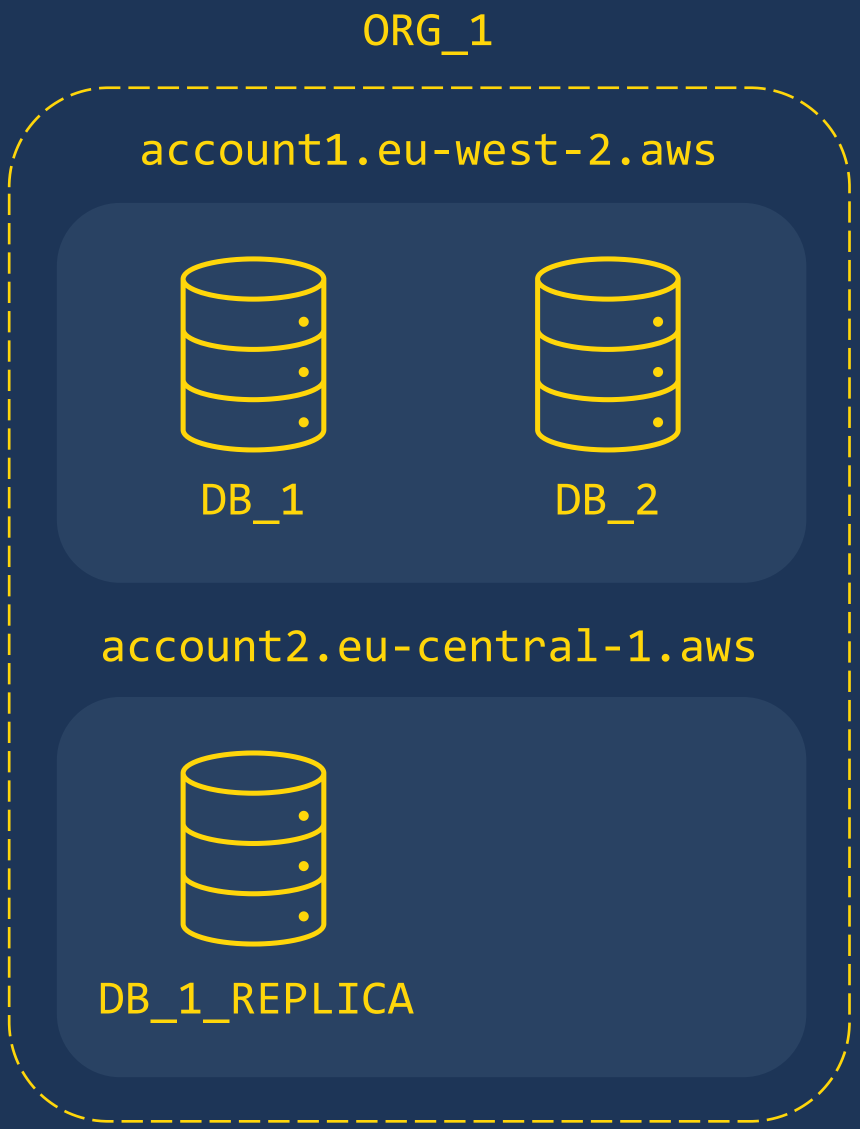
When a primary database is replicated a snapshot of it's database objects and data are transferred to the secondary database:

```
CREATE DATABASE DB_1_REPLICA
AS REPLICA OF ORG_1.account1.DB_1;
```

The secondary database can be periodically refreshed:

```
ALTER DATABASE DB_1_REPLICA REFRESH;
```

# Replication



ORG_1

account1.eu-west-2.aws

DB_1

DB_2

account2.eu-central-1.aws

DB_1_REPLICA

1

External Tables, Stages, Pipes, Streams and Tasks are not currently replicated.

2

Refreshing a secondary database can be automated by configuring a task object to run the refresh command on a schedule.

3

Only databases and some of their child objects can be replicated, not users, roles, warehouses, resource monitors or shares.

4

Privileges granted to database objects are not replicated to the secondary database.

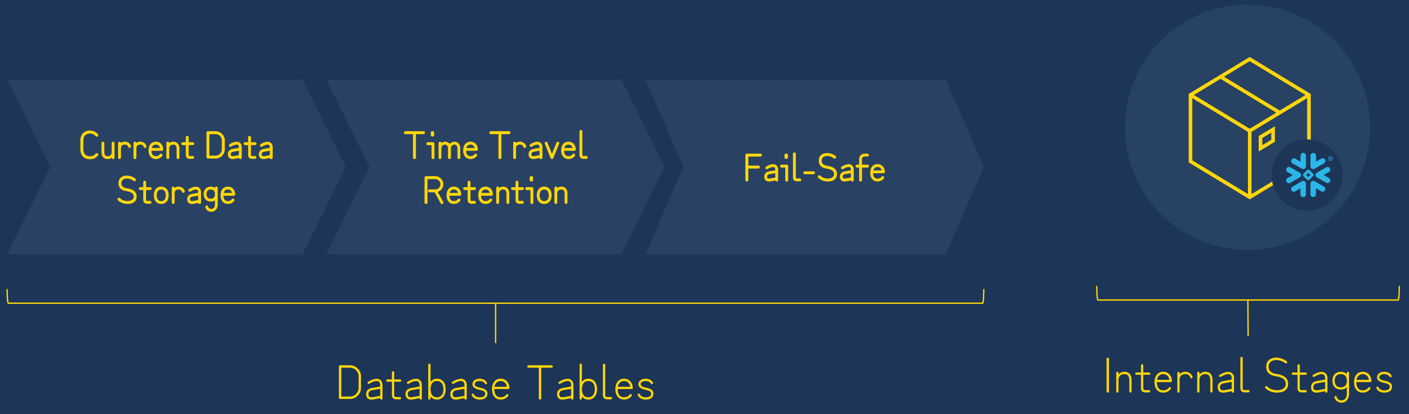
5

Billing for database replication is based on data transfer and compute resources.

# Storage Billing

# Storage Billing

Data storage cost is calculated monthly based on the average number of on-disk bytes for all data stored each day in a Snowflake account.



The monthly costs for storing data in Snowflake is based on a flat rate per Terabyte.

$42 per Terabyte per month for customers deployed in AWS – EU (London)

# Data Storage Pricing

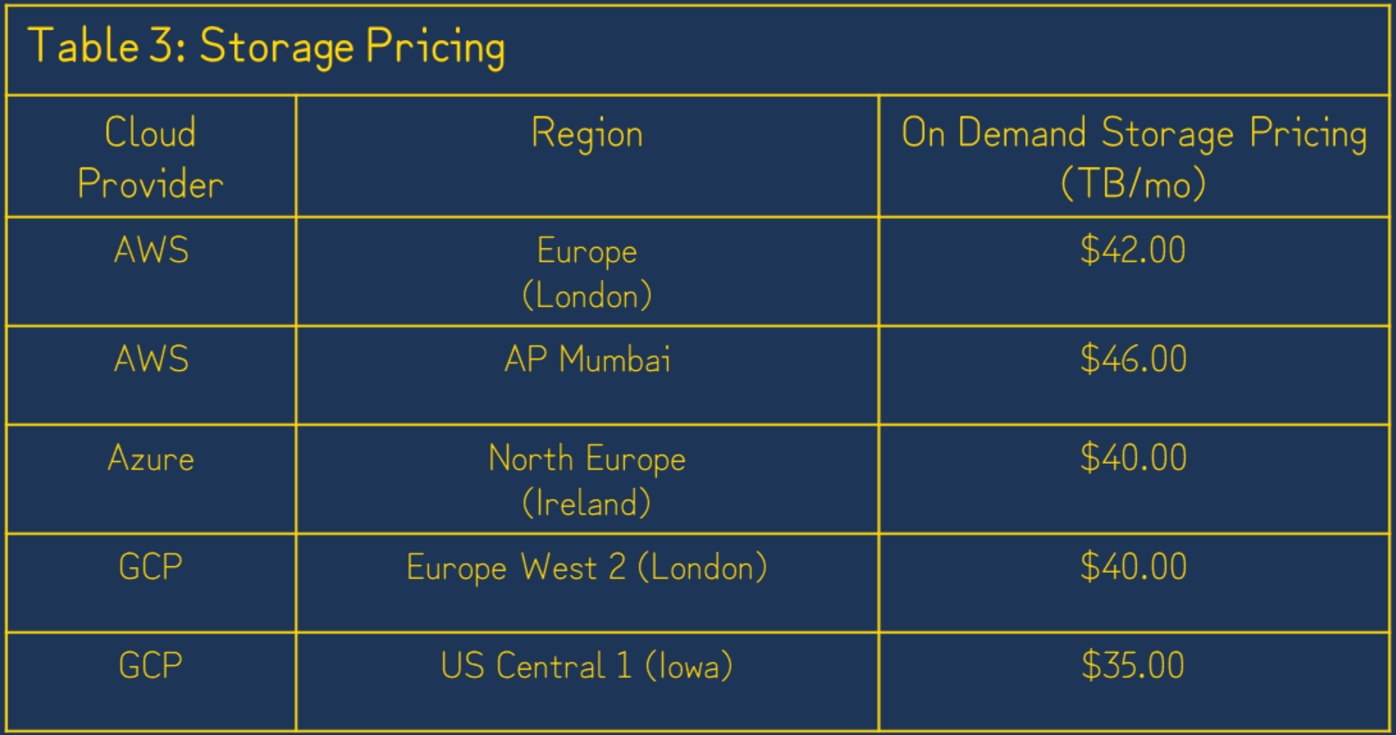


Table 3: Storage Pricing

| Cloud Provider | Region | On Demand Storage Pricing (TB/mo) |
|---|---|---|
| AWS | Europe (London) | $42.00 |
| AWS | AP Mumbai | $46.00 |
| Azure | North Europe (Ireland) | $40.00 |
| GCP | Europe West 2 (London) | $40.00 |
| GCP | US Central 1 (Iowa) | $35.00 |

ⓘ Storage cost is determined by **Cloud Provider**, **Region** & **Pricing Plan.**

# Data Storage Billing Scenarios

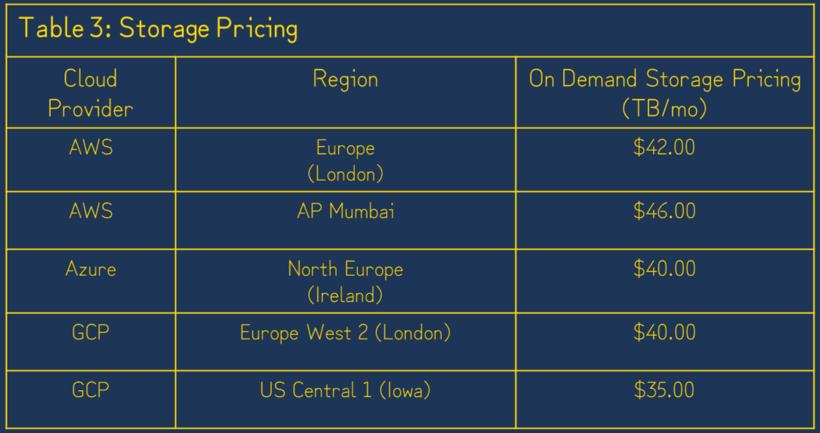


Table 3: Storage Pricing

| Cloud Provider | Region | On Demand Storage Pricing (TB/mo) |
|---|---|---|
| AWS | Europe (London) | $42.00 |
| AWS | AP Mumbai | $46.00 |
| Azure | North Europe (Ireland) | $40.00 |
| GCP | Europe West 2 (London) | $40.00 |
| GCP | US Central 1 (Iowa) | $35.00 |

Storage

If an average of 15TB is stored during a month on account deployed in AWS Europe (London) Region it will cost $630.00 (Nov 2021)

If an average of 14TB is stored during a month on account deployed in AWS AP Mumbai Region it will cost $644.00 (Nov 2021)

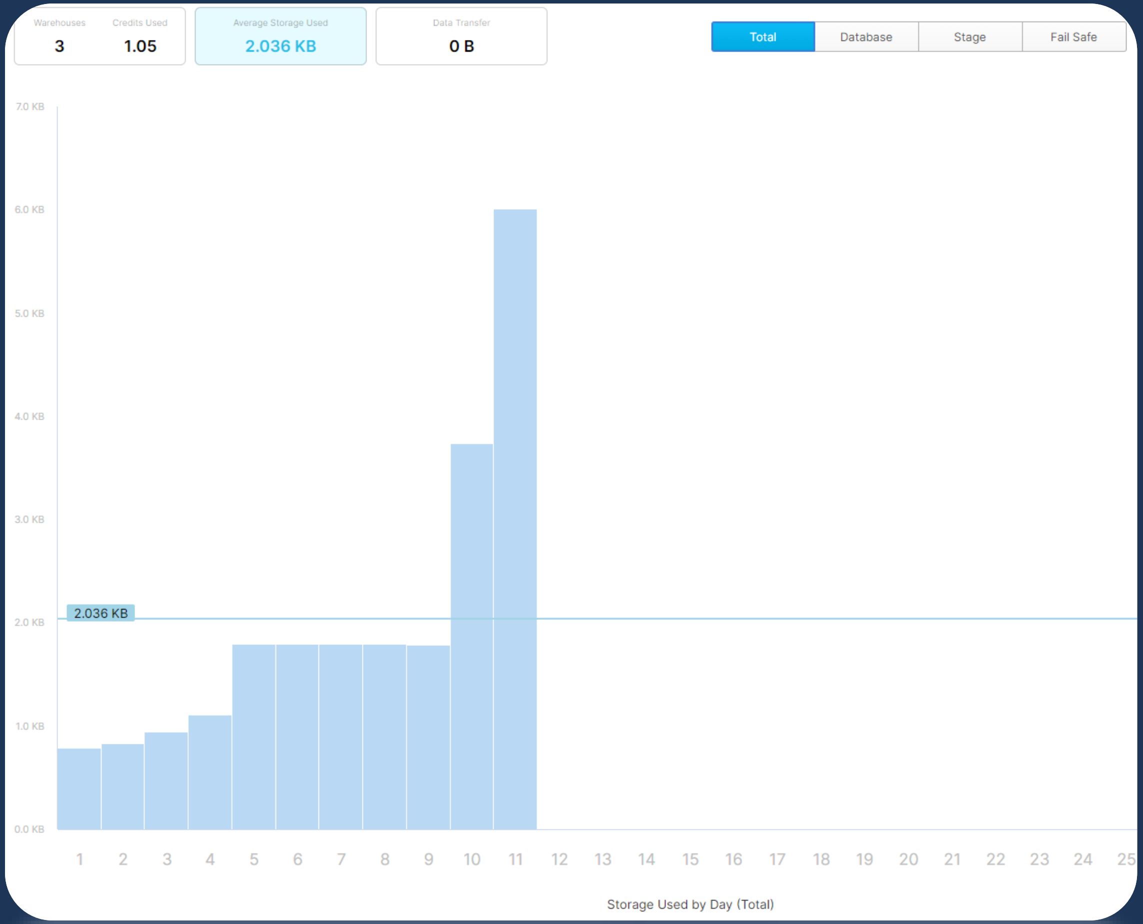
# Storage Monitoring

Data storage usage can be monitored from the Classic and Snowsight user interfaces.

Equivalent functionality can be achieved via the account usage views and information schema views and table functions.

Account Usage Views:

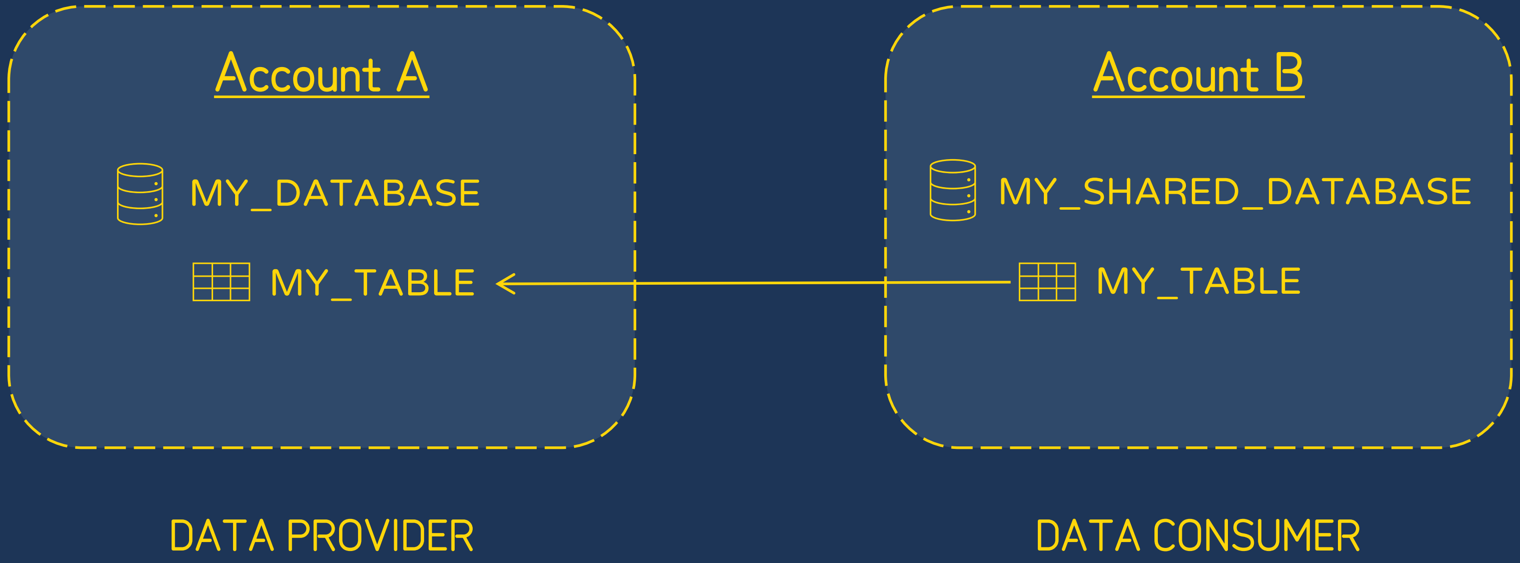
- DATABASE_STORAGE_USAGE
- STAGE_STORAGE_USAGE_HISTORY
- TABLE_STORAGE_METRICS



# Data Sharing

# Secure Data Sharing

Secure Data Sharing allows an account to provide read-only access to selected database objects to other Snowflake accounts without transferring data.



# Secure Data Sharing

Sharing is enabled with an account-level **SHARE** object. It is created by a data provider and consists of two key configurations:

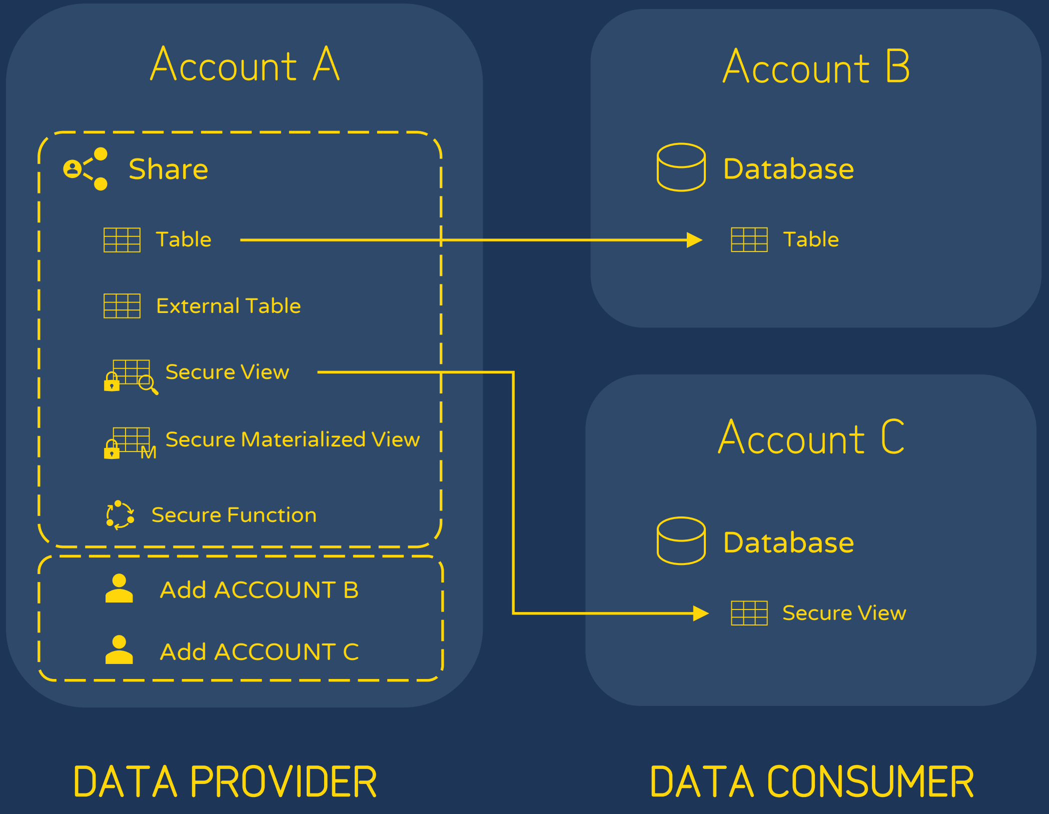
- Grants on database objects
- Consumer account definitions

An account can share the following database objects:

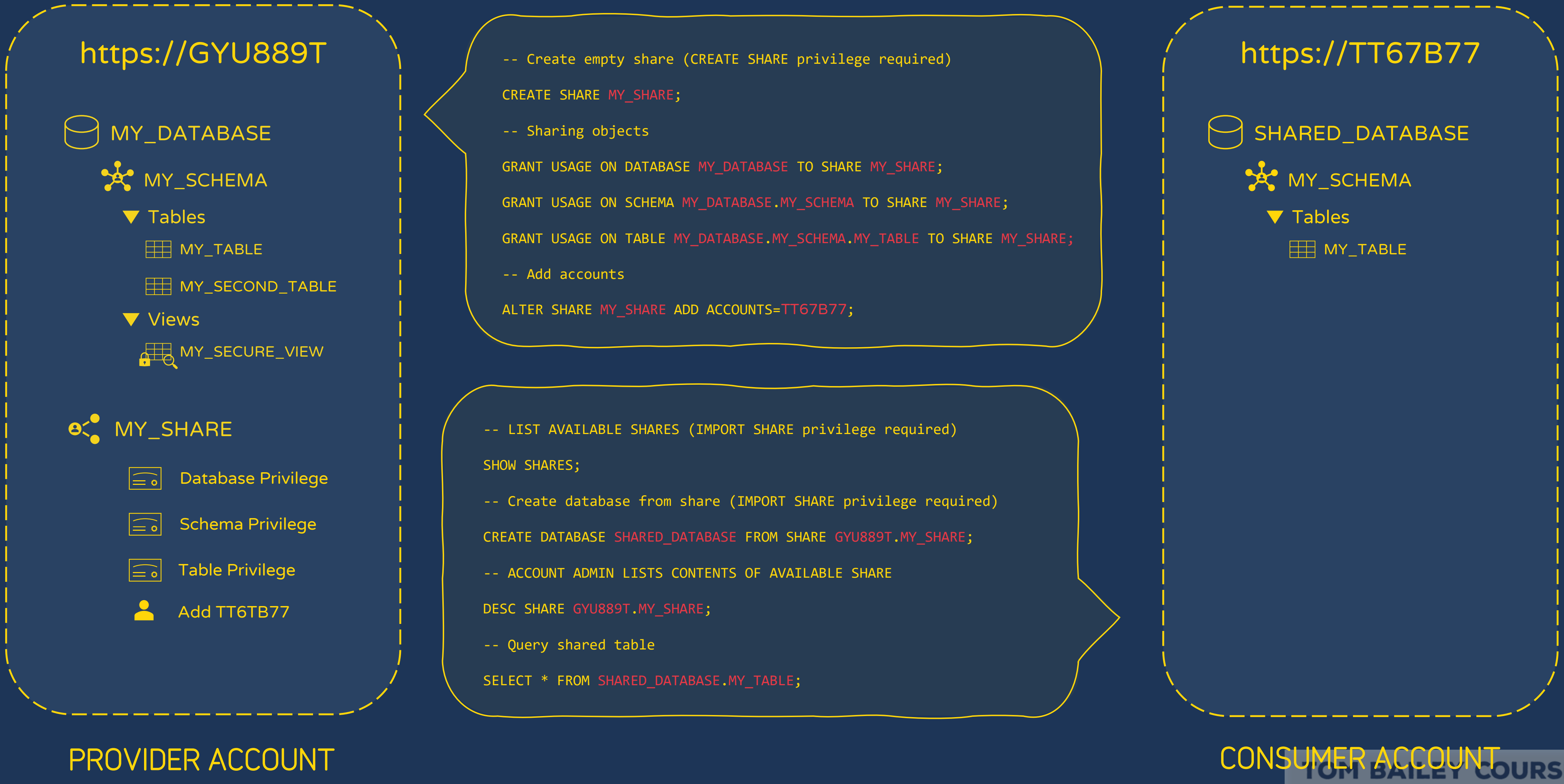
- Tables
- External tables
- Secure views
- Secure materialized views
- Secure UDFs

Data consumers create a database from a SHARE which contains the read-only database objects granted by the data provider.

Sharing is not available on the **VPS edition** of Snowflake.



# Secure Data Sharing Example



https://GYU889T

MY_DATABASE

MY_SCHEMA

▼ Tables

MY_TABLE

MY_SECOND_TABLE

▼ Views

MY_SECURE_VIEW

MY_SHARE

Database Privilege

Schema Privilege

Table Privilege

Add TT6TB77

PROVIDER ACCOUNT

```
-- Create empty share (CREATE SHARE privilege required)

CREATE SHARE MY_SHARE;

-- Sharing objects

GRANT USAGE ON DATABASE MY_DATABASE TO SHARE MY_SHARE;

GRANT USAGE ON SCHEMA MY_DATABASE.MY_SCHEMA TO SHARE MY_SHARE;

GRANT USAGE ON TABLE MY_DATABASE.MY_SCHEMA.MY_TABLE TO SHARE MY_SHARE;

-- Add accounts

ALTER SHARE MY_SHARE ADD ACCOUNTS=TT67B77;
```

```
-- LIST AVAILABLE SHARES (IMPORT SHARE privilege required)

SHOW SHARES;

-- Create database from share (IMPORT SHARE privilege required)

CREATE DATABASE SHARED_DATABASE FROM SHARE GYU889T.MY_SHARE;

-- ACCOUNT ADMIN LISTS CONTENTS OF AVAILABLE SHARE

DESC SHARE GYU889T.MY_SHARE;

-- Query shared table

SELECT * FROM SHARED_DATABASE.MY_TABLE;
```

https://TT67B77

SHARED_DATABASE

MY_SCHEMA

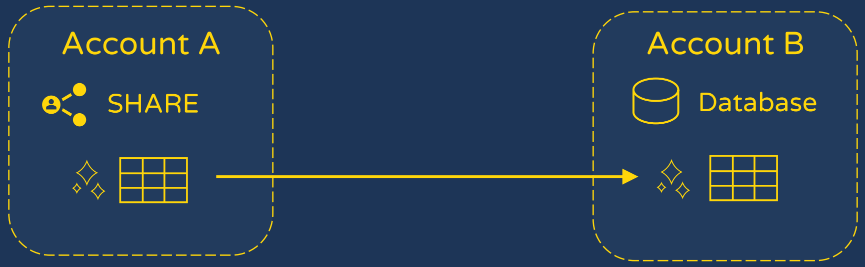
▼ Tables

MY_TABLE

CONSUMER ACCOUNT

# Data Provider

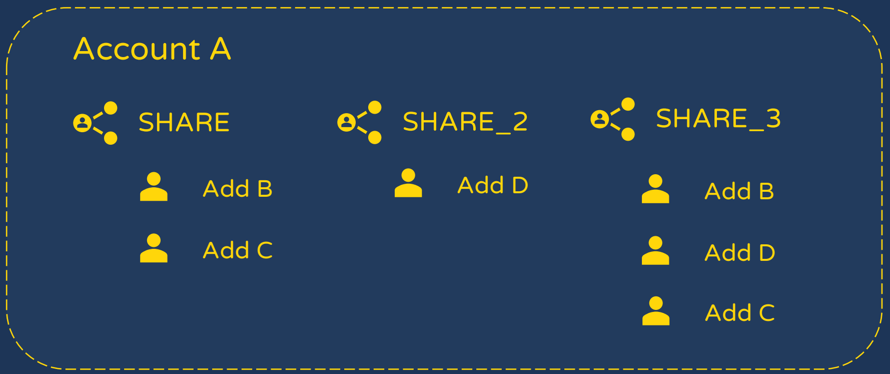
Database objects added to a share become immediately available to all consumers.



Only **one** database can be added per share.

```
SQL compilation error: Database 'TEST_DB' does
not belong to the database that is being shared.
```

No hard limits on the number of shares you can create or the number of accounts you can add to a share.

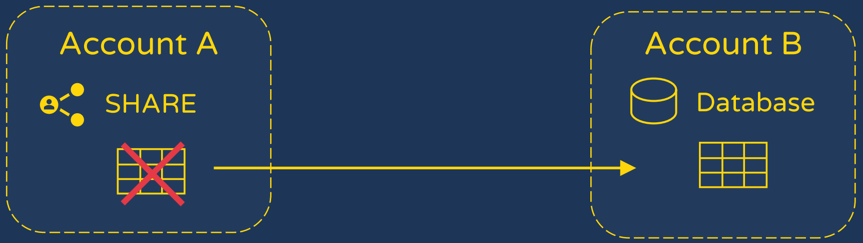


# Data Provider

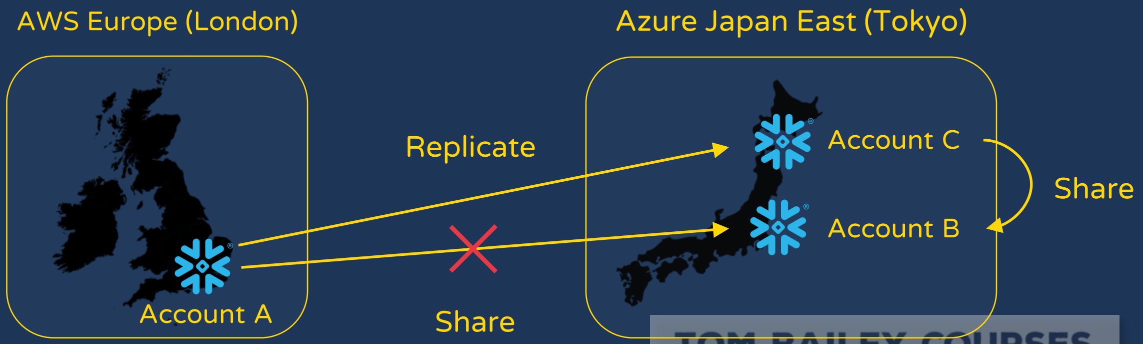
To execute a create share command a user must have a role with the CREATE SHARE privilege granted.

CREATE SHARE

Access to a share or database objects in a share can be revoked at any time.



A share can only be granted to accounts in the same region and cloud provider as the data provider account.



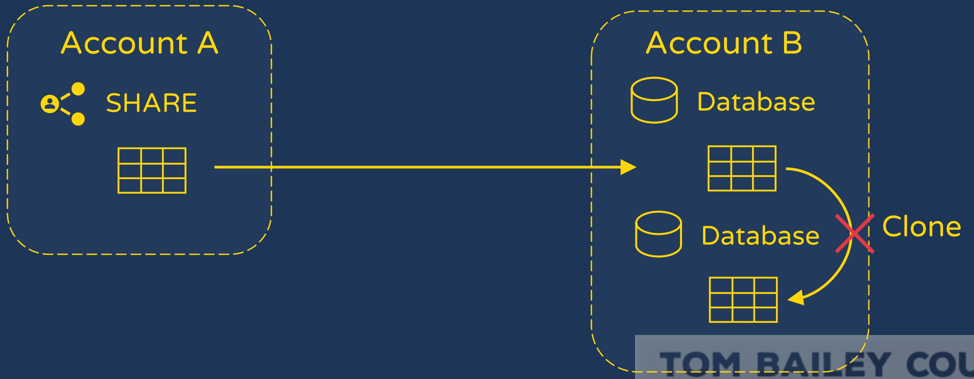
# Data Consumer

Only one database can be created per share.

```
Importing more than once is not supported. Database
is already imported as 'SNOWFLAKE_SAMPLE_DATA’.
```

A data consumer cannot use the Time Travel feature on shared database objects.

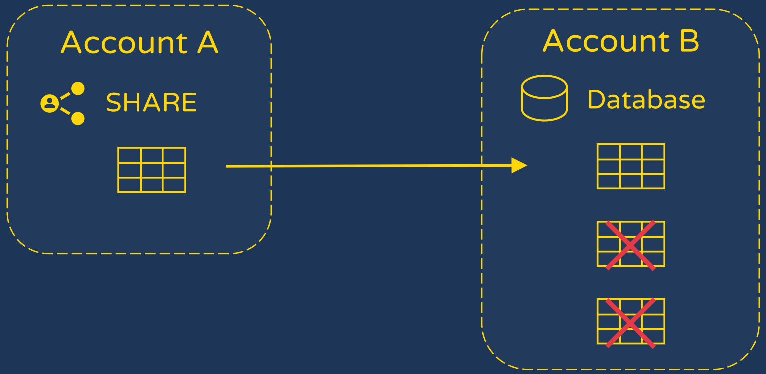
A data consumer cannot create a clone of a shared database or database objects.



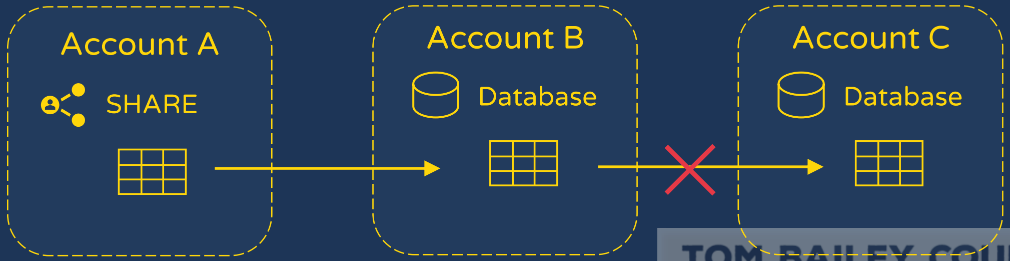
# Data Consumer

To create a database from a share a user must have a role with the IMPORT SHARE privilege granted.

Data consumers cannot create objects inside the shared database.



Data consumers cannot reshare shared database objects.



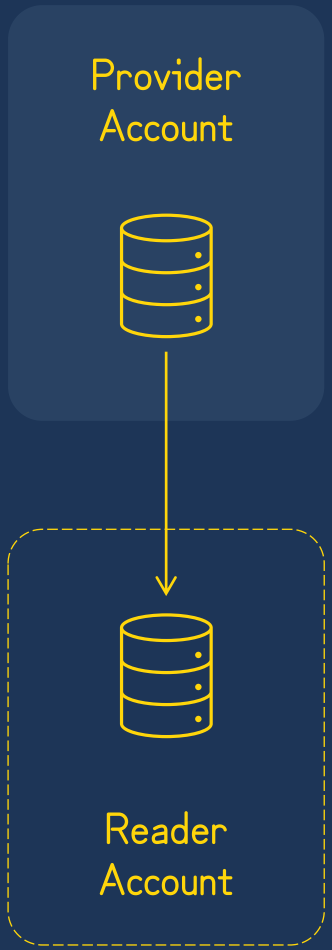
# Reader Account

A reader account provides the ability for non-Snowflake customers to gain access to a providers data.

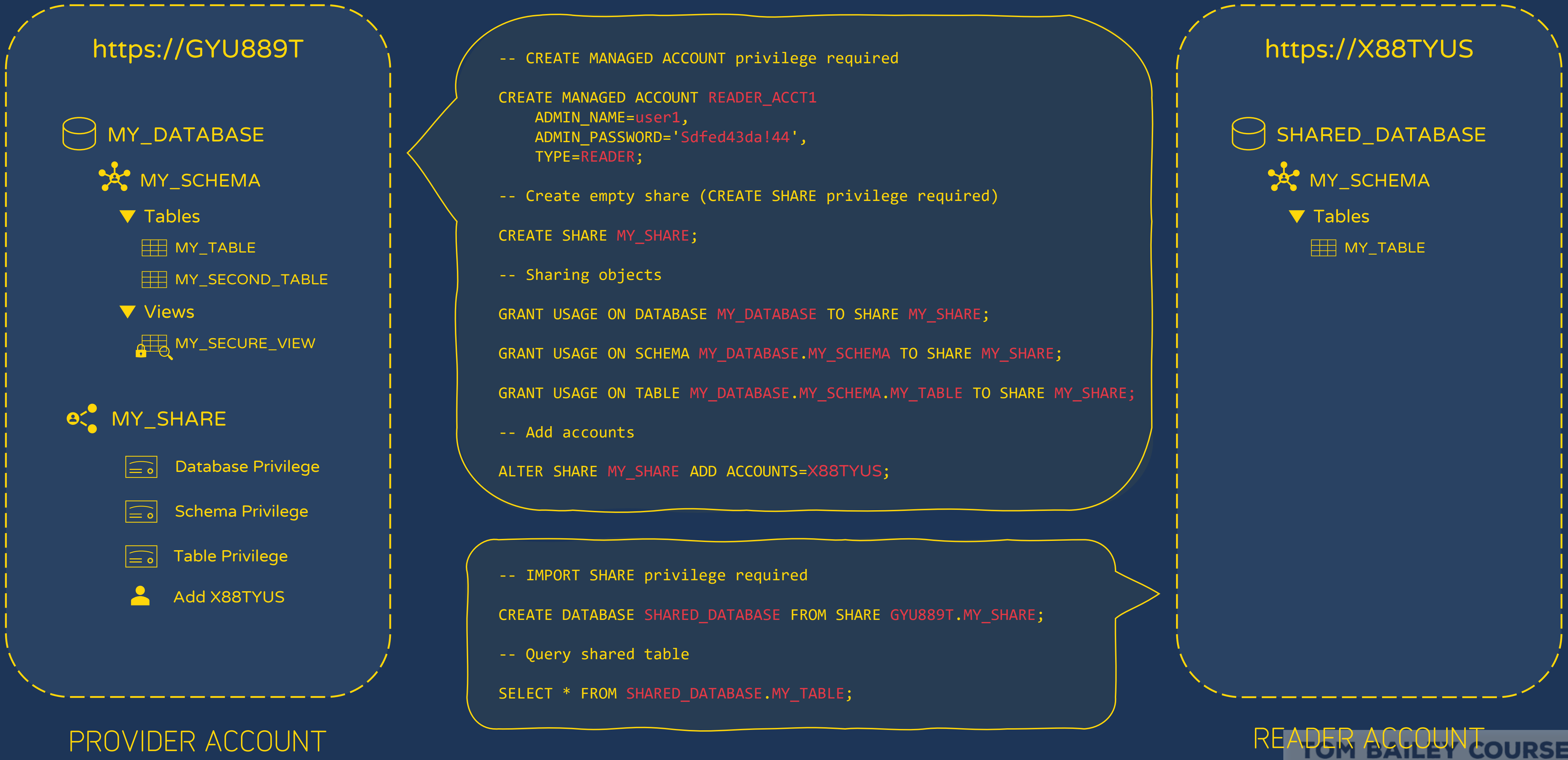
Reader accounts can't insert data or copy data into an account.

A reader account can only consume data from the provider account that created it.

Provider accounts assume responsibility for the reader account they create and are billed for their usage.



# Reader Account



https://GYU889T

MY_DATABASE

MY_SCHEMA

▼ Tables

MY_TABLE

MY_SECOND_TABLE

▼ Views

MY_SECURE_VIEW

MY_SHARE

Database Privilege

Schema Privilege

Table Privilege

Add X88TYUS

PROVIDER ACCOUNT

```
-- CREATE MANAGED ACCOUNT privilege required

CREATE MANAGED ACCOUNT READER_ACCT1
    ADMIN_NAME=user1,
    ADMIN_PASSWORD='Sdfed43da!44',
    TYPE=READER;

-- Create empty share (CREATE SHARE privilege required)

CREATE SHARE MY_SHARE;

-- Sharing objects

GRANT USAGE ON DATABASE MY_DATABASE TO SHARE MY_SHARE;

GRANT USAGE ON SCHEMA MY_DATABASE.MY_SCHEMA TO SHARE MY_SHARE;

GRANT USAGE ON TABLE MY_DATABASE.MY_SCHEMA.MY_TABLE TO SHARE MY_SHARE;

-- Add accounts

ALTER SHARE MY_SHARE ADD ACCOUNTS=X88TYUS;
```

```
-- IMPORT SHARE privilege required

CREATE DATABASE SHARED_DATABASE FROM SHARE GYU889T.MY_SHARE;

-- Query shared table

SELECT * FROM SHARED_DATABASE.MY_TABLE;
```

https://X88TYUS

SHARED_DATABASE

MY_SCHEMA

▼ Tables

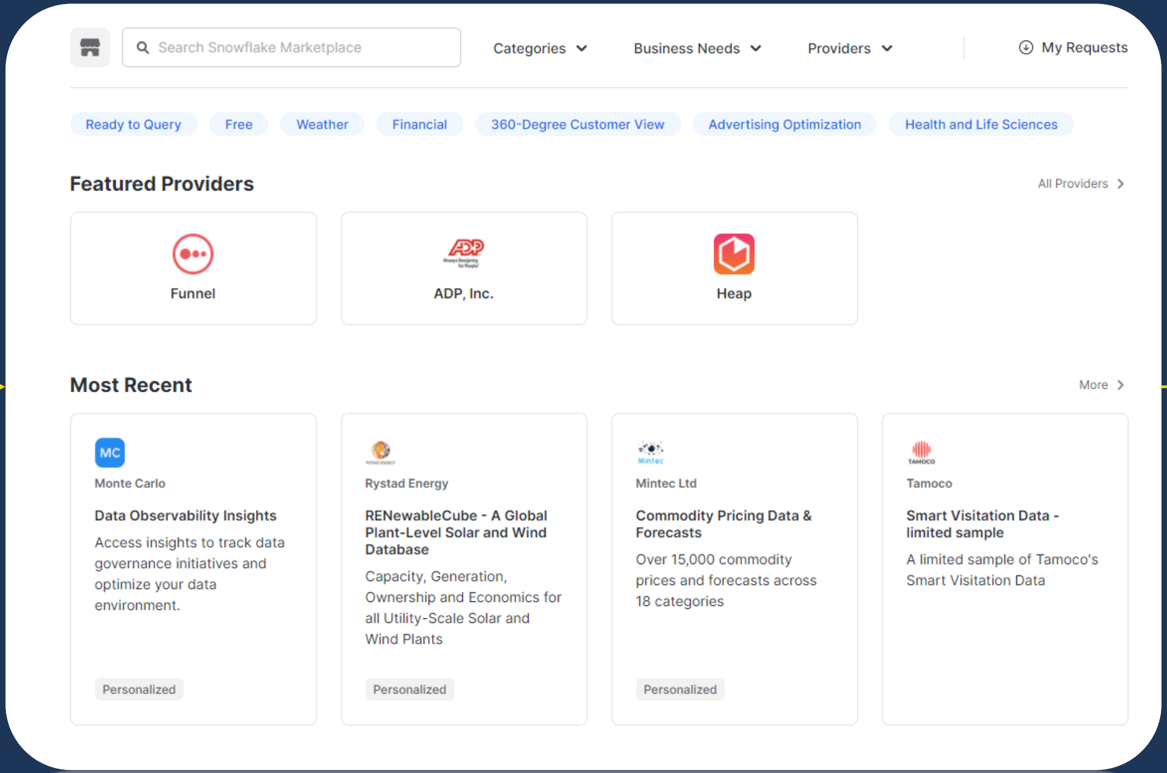
MY_TABLE

READER ACCOUNT

# Data Marketplace

Portal for browsing publicly available third-party shares.

Data Provider



Search Snowflake Marketplace

Categories ⌄ Business Needs ⌄ Providers ⌄ My Requests

Ready to Query · Free · Weather · Financial · 360-Degree Customer View · Advertising Optimization · Health and Life Sciences

**Featured Providers** All Providers ›

Funnel

ADP, Inc.

Heap

**Most Recent** More ›

Monte Carlo
**Data Observability Insights**
Access insights to track data governance initiatives and optimize your data environment.
Personalized

Rystad Energy
**RENewableCube - A Global Plant-Level Solar and Wind Database**
Capacity, Generation, Ownership and Economics for all Utility-Scale Solar and Wind Plants
Personalized

Mintec Ltd
**Commodity Pricing Data & Forecasts**
Over 15,000 commodity prices and forecasts across 18 categories
Personalized

Tamoco
**Smart Visitation Data - limited sample**
A limited sample of Tamoco's Smart Visitation Data

Data Consumer

⇒ Via the Provider Studio Data Provides can list **standard** or **personalised** shares.

⇒ Third-party dataset immediately available to query from account without transformation.

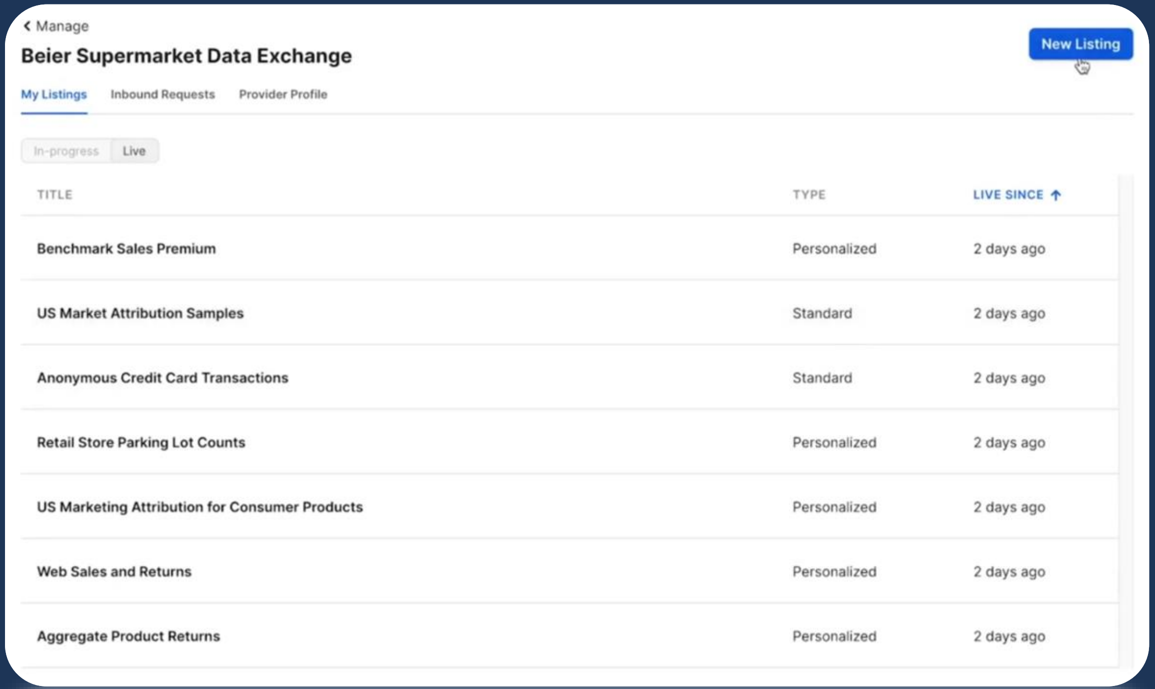
# Video 22: Data Exchange

# Data Exchange

A Data Exchange is a private version of the data marketplace for accounts to provide and consume data.

A Data Exchange for a Snowflake account is set up with Snowflake support by providing a name and description.

The Snowflake account hosting a Data Exchange is the Data Exchange Admin and can manage members, designate members as providers and consumers and manage listing requests.



Beier Supermarket Data Exchange

| TITLE | TYPE | LIVE SINCE ↑ |
|---|---|---|
| Benchmark Sales Premium | Personalized | 2 days ago |
| US Market Attribution Samples | Standard | 2 days ago |
| Anonymous Credit Card Transactions | Standard | 2 days ago |
| Retail Store Parking Lot Counts | Personalized | 2 days ago |
| US Marketing Attribution for Consumer Products | Personalized | 2 days ago |
| Web Sales and Returns | Personalized | 2 days ago |
| Aggregate Product Returns | Personalized | 2 days ago |