Loading Data

Loading Data

BULK LOADING

- ✓ Most frequent method
- ✓ Uses warehouses
- ✓ Loading from stages
- ✓ COPY command
- ✓ Transformations possible

CONTINUOUS LOADING

- ✓ Designed to load small volumes of data
- ✓ Automatically once they are added to stages
- ✓ Lates results for analysis
- ✓ Snowpipe (Serverless feature)

Understanding Stages

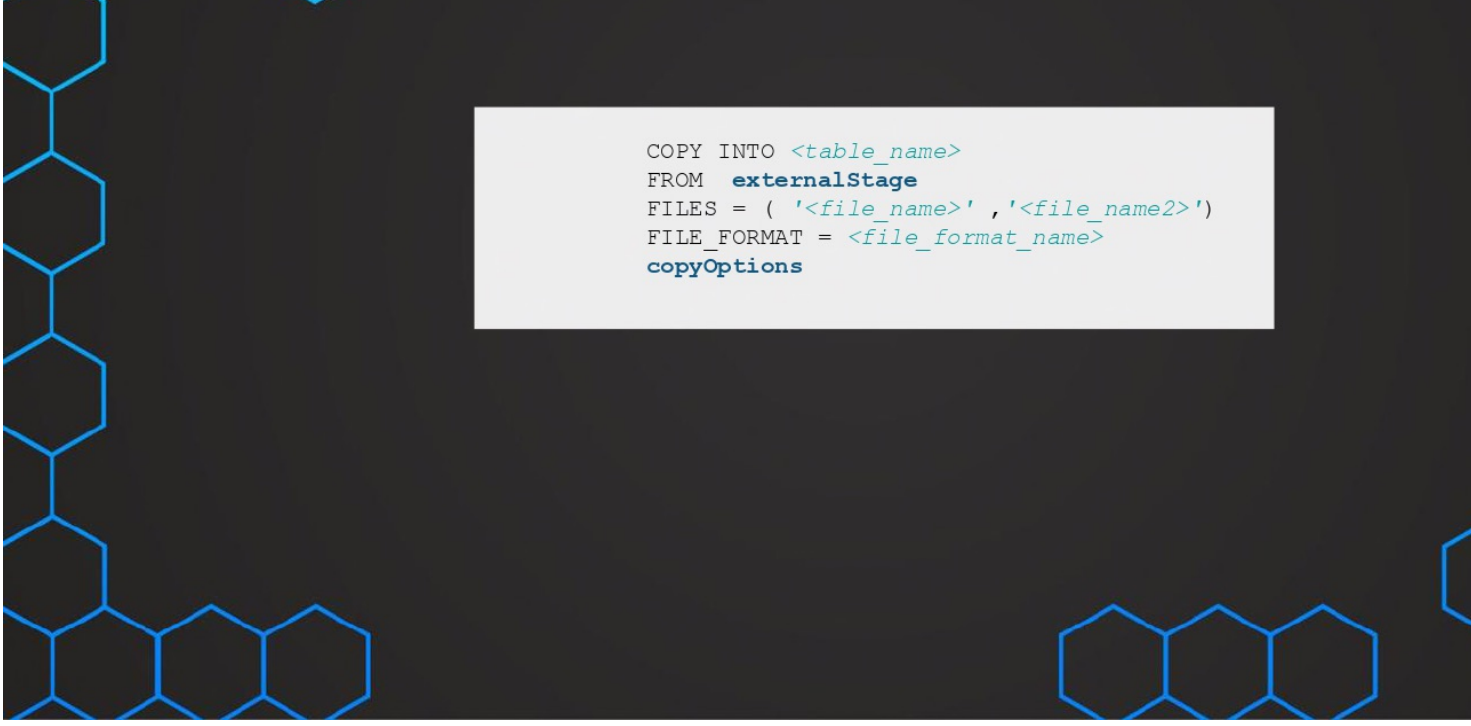
External Stage

- ✓ External cloud provider
  - S3
  - Google Cloud Plattform
  - Microsoft Azure
- ✓ Database object created in Schema
- ✓ CREATE STAGE (URL, access settings)

Note: Additional costs may apply if region/platform differs

Internal Stage

- ✓ Local storage maintained by Snowflake


COPY INTO <table_name>
FROM externalStage
FILES = ('<file_name>' ,'<file_name2>')
FILE_FORMAT = <file_format_name>
copyOptions

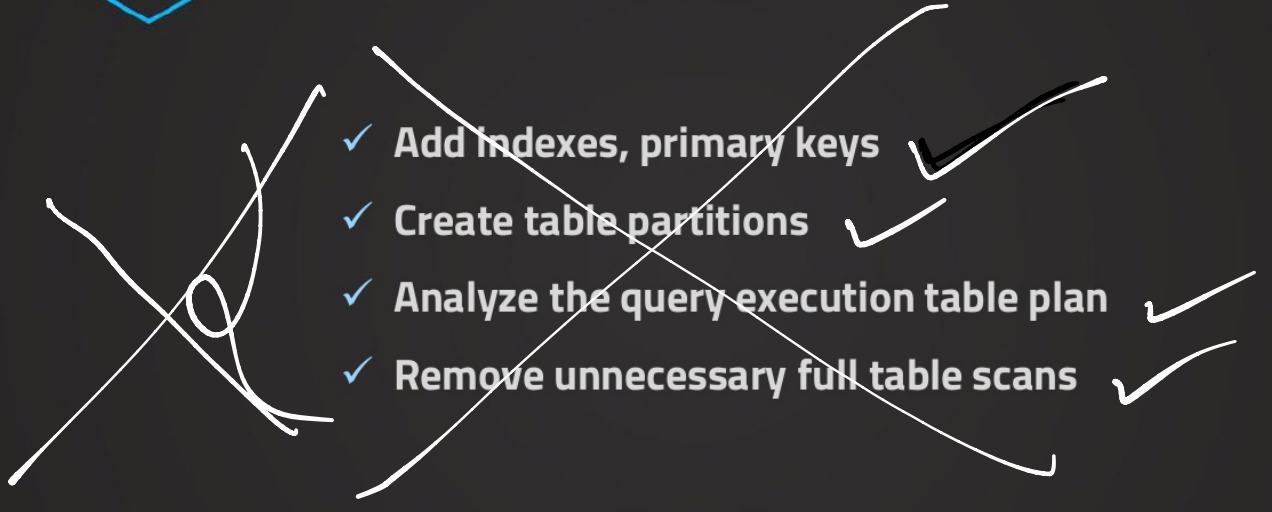
Performance Optimization

Performance Optimization

Make queries run faster

Save costs

Performance Optimization



- ✓ Add indexes, primary keys
- ✓ Create table partitions
- ✓ Analyze the query execution table plan
- ✓ Remove unnecessary full table scans

What is our job?

- ✓ Assigning appropriate data types
- ✓ Sizing virtual warehouses
- ✓ Cluster keys

Performance aspects

Dedicated virtual warehouses

- ✓ Separated according to different workloads

Scaling Up

- ✓ For known patterns of high work load

Scaling Out

- ✓ Dynamically fo unknown patterns of work load

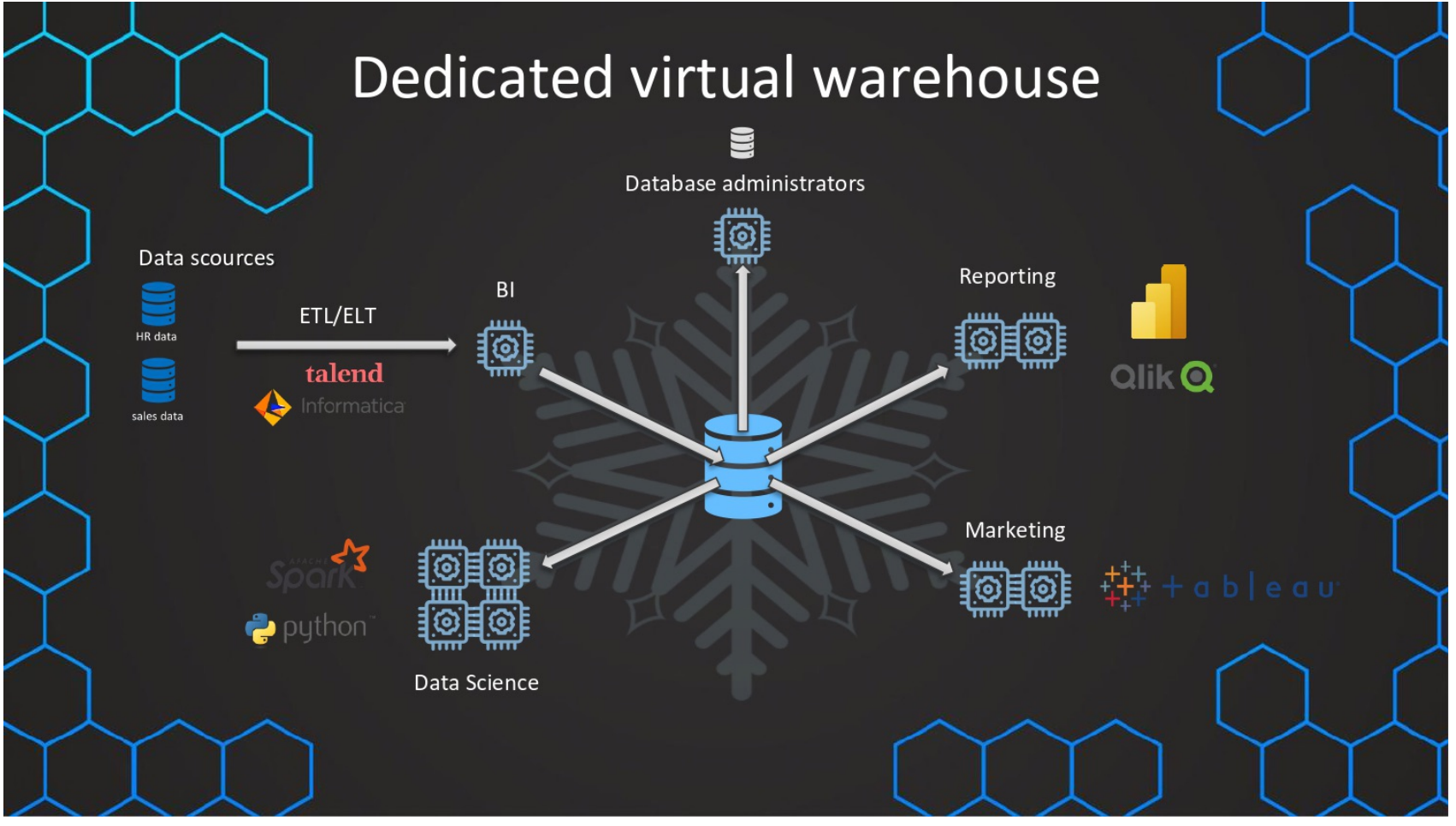
Maximize Cache Usage

- ✓ Automatic caching can be maximized

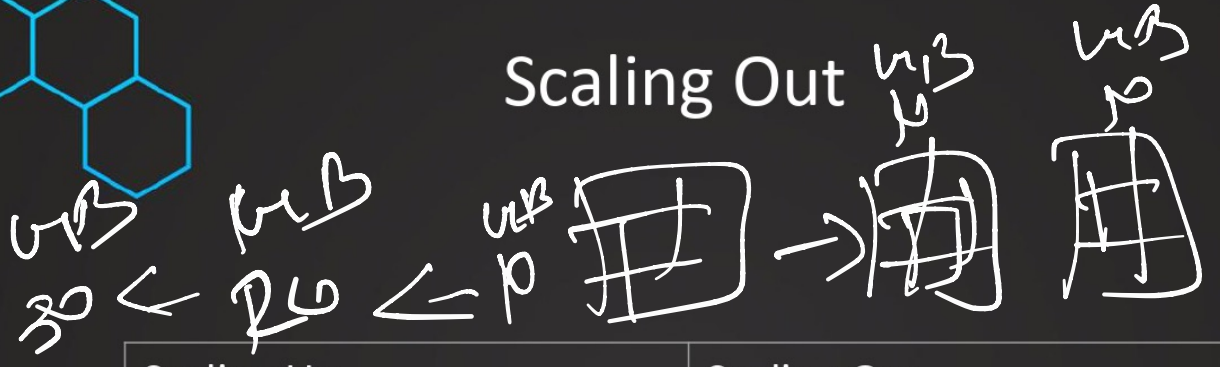
Cluster Keys

- ✓ For large tables

Dedicated virtual warehouse



Scaling Out



Scaling Up	Scaling Out
Increasing the size of virtual warehouses	Using addition warehouses/ Multi-Cluster warehouses
More complex query	More concurrent users/queries

Caching

- ✓ Automatical process to speed up the queries
- ✓ If query is executed twice, results are cached and can be re-used
- ✓ Results are cached for 24 hours or until underlaying data has changed

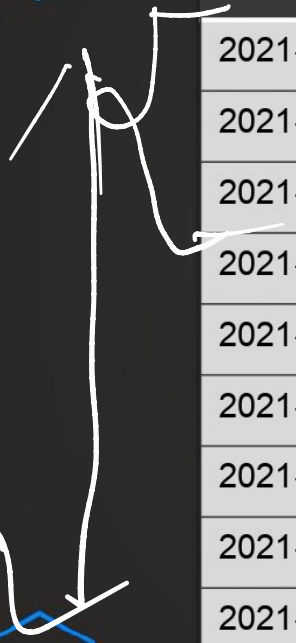
Clustering in Snowflake

- ✓ Snowflake automatically maintains these cluster keys
- ✓ In general Snowflake produces well-clustered tables
- ✓ Cluster keys are not always ideal and can change over time
- ✓ Manually customize these cluster keys

What is a cluster key?

- ✓ Subset of rows to locate the data in micro-partions
- ✓ For large tables this improves the scan efficiency in our queries

What is a cluster key?



Event Date	Event ID	Customers	City
2021-03-12	134584	…	…
2021-12-04	134586	…	…
2021-11-04	134588	…	…
2021-04-05	134589	…	…
2021-06-07	134594	…	…
2021-07-03	134597	…	…
2021-03-04	134598	…	…
2021-08-03	134599	…	…
2021-08-04	134601	…	…

What is a cluster key?

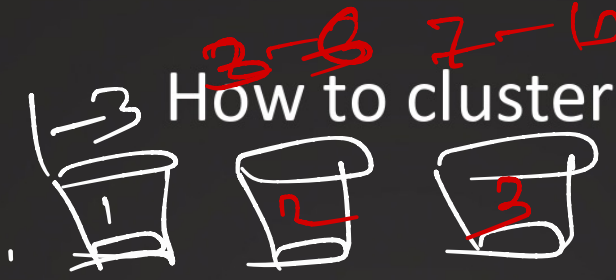
Event Date	Event ID	Customers	City
2021-03-12	134584	…	…
2021-12-04	134586	…	…
2021-11-04	134588	…	…
2021-04-05	134589	…	…
2021-06-07	134594	…	…
2021-07-03	134597	…	…
2021-03-04	134598	…	…
2021-08-03	134599	…	…
2021-08-04	134601	…	…

Event Date	Event ID	Customers	City
2021-03-12	134584	…	…
2021-12-04	134586	…	…
2021-11-04	134588	…	…
2021-04-05	134589	…	…
2021-06-07	134594	…	…
2021-07-03	134597	…	…
2021-03-04	134598	…	…
2021-08-03	134599	…	…
2021-08-04	134601	…	…

When to cluster?

- ✓ Clustering is not for all tables
- ✓ Mainly very large tables of multiple terabytes can benefit

How to cluster?



- ✓ Columns that are used most frequently in WHERE-clauses (often date columns for event tables)
- ✓ If you typically use filters on two columns then the table can also benefit from two cluster keys
- ✓ Column that is frequently used in Joins
- ✓ Large enough number of distinct values to enable effective grouping
  Small enough number of distinct values to allow effective grouping

Clustering in Snowflake

```
CREATE TABLE <name> ... CLUSTER BY ( <column1> [ , <column2> ... ] )
```

```
CREATE TABLE <name> ... CLUSTER BY ( <expression> )
```

```
ALTER TABLE <name> CLUSTER BY ( <expr1> [ , <expr2> ... ] )
```

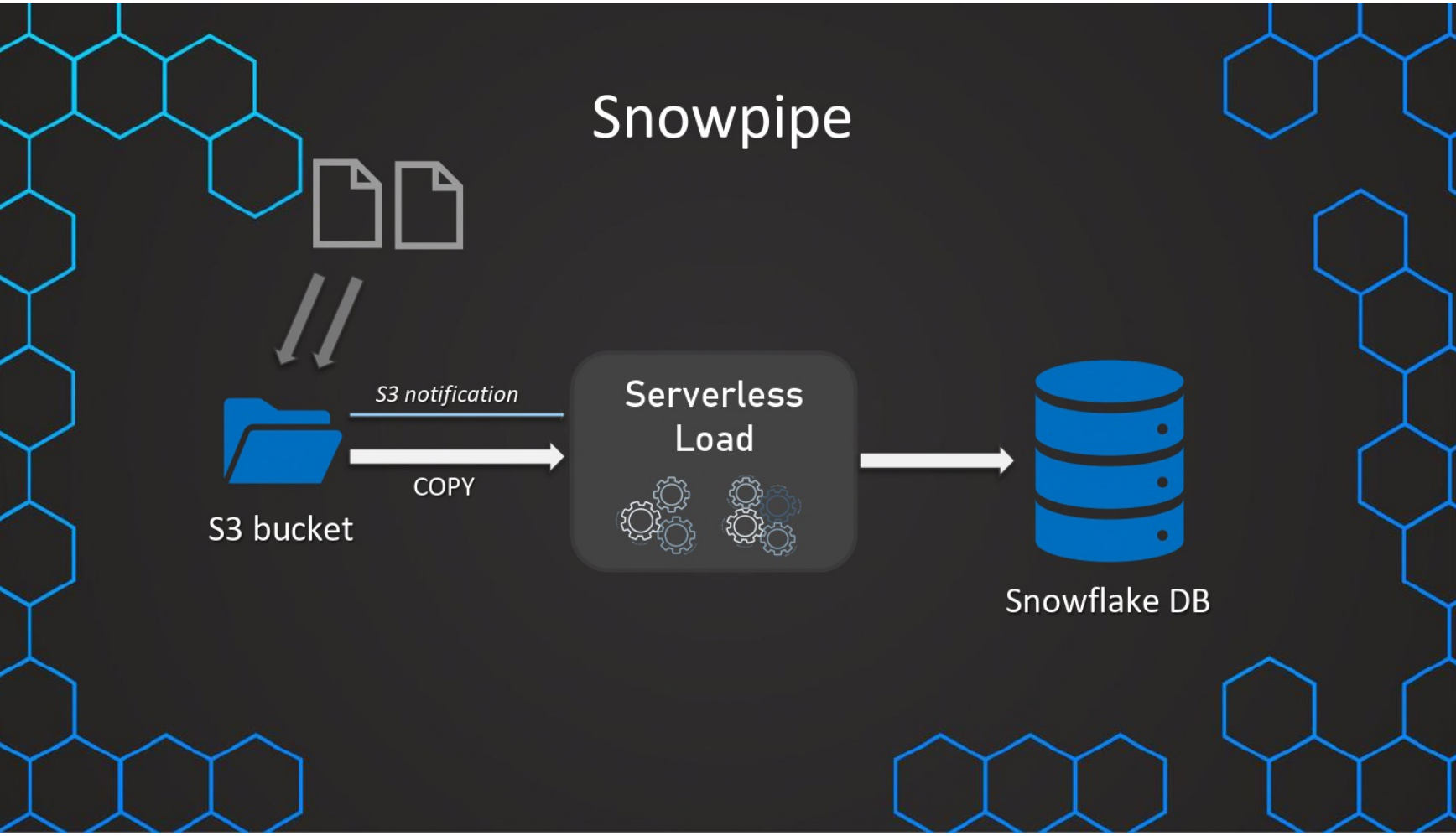
```
ALTER TABLE <name> DROP CLUSTERING KEY
```

Snowpipe

What is Snowpipe?

- ✓ Enables loading once a file appears in a bucket
- ✓ If data needs to be available immediately for analysis
- ✓ Snowpipe uses serverless features instead of warehouses

Snowpipe


S3 notification
COPY
S3 bucket
Serverless Load
Snowflake DB

Setting up Snowpipe

Create Stage ✓ To have the connection

Test COPY COMMAND ✓ To make sure it works

Create Pipe ✓ Create pipe as object with COPY COMMAND

S3 Notification ✓ To trigger snowpipe

Time Travel

Standard

- ✓ Time travel up to 1 day

Enterprise

- ✓ Time travel up to 90 days

Business Critical

- ✓ Time travel up to 90 days

Virtual Private

- ✓ Time travel up to 90 days

RETENTION PERIODE DEFAULT = 1

Continuous Data Protection Lifecycle

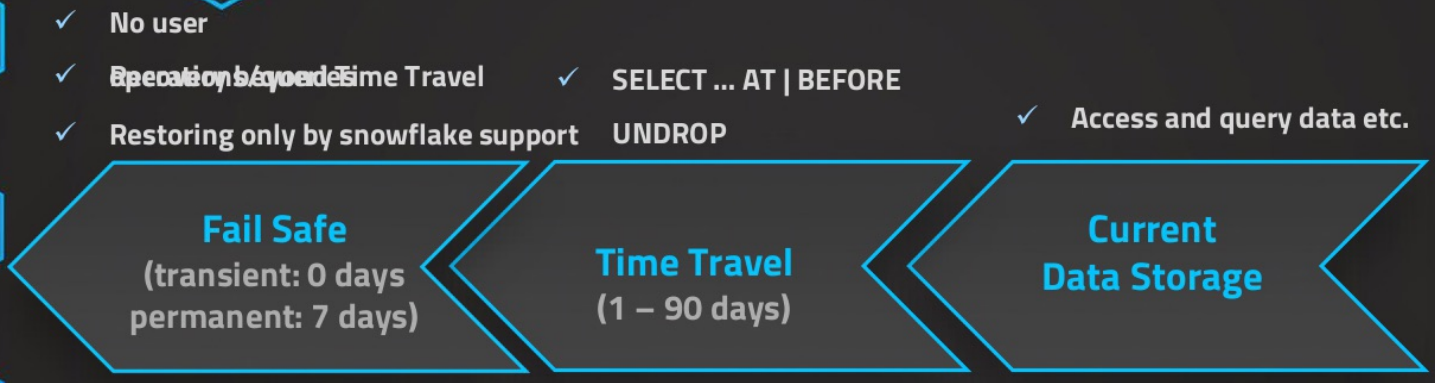


Table Types

Table types

Permanent data

Until dropped

Permanent

`CREATE TABLE`

- ✓ Time Travel Retention Period
  0 – 90 days
- ✓ Fail Safe

Only for data that does not need to be protected

Until dropped

Transient

`CREATE TRANSIENT TABLE`

- ✓ Time Travel Retention Period
  0 – 1 day
- × Fail Safe

Non-permanent data

Only in session

Temporary

`CREATE TEMPORARY TABLE`

- ✓ Time Travel Retention Period
  0 – 1 day
- × Fail Safe

Table types notes

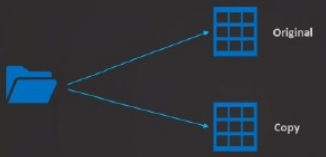
- ✓ Types are also available for other database objects (database, schema etc.)
- ✓ For temporary table no naming conflicts with permanent/transient tables!

Other tables will be effectively hidden!

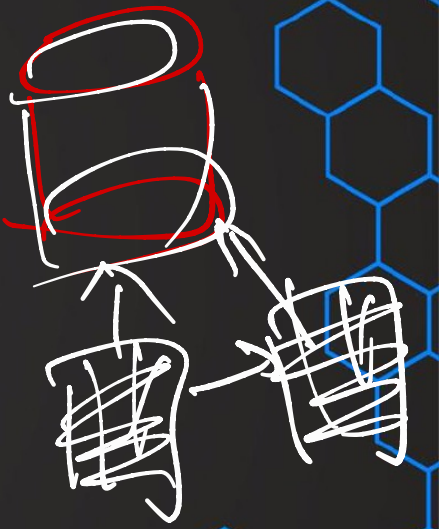
Zero Copy Cloning

Zero-Copy Cloning

- ✓ Create copies of a database, a schema or a table



- ✓ Cloned object is independent from original table
- ✓ Easy to copy all meta data & improved storage management
- ✓ Creating backups for development purposes
- ✓ Works with time travel also



Zero-Copy Cloning

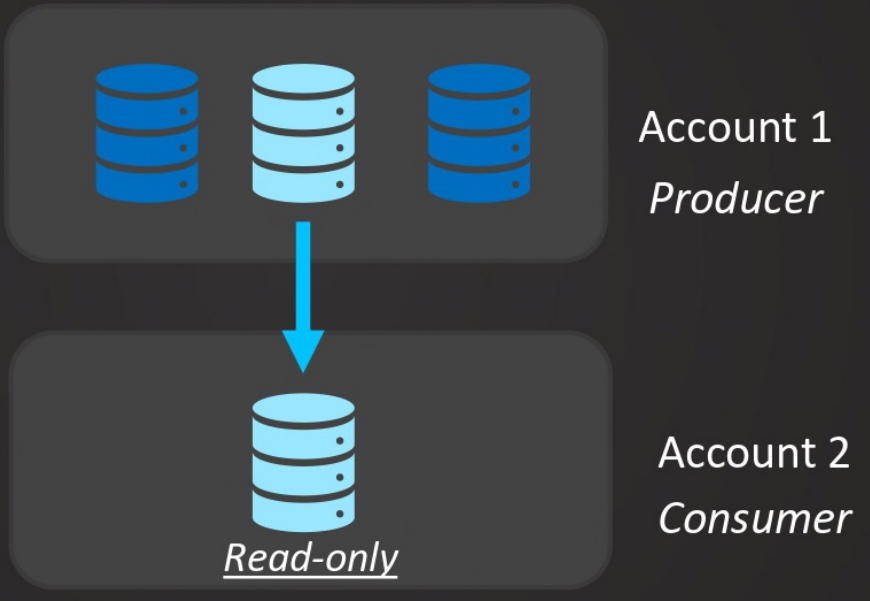
```
CREATE TABLE <table_name> ...
  CLONE  <source_table_name>
  BEFORE ( TIMESTAMP => <timestamp> )
```

Data Sharing

Data Sharing

- ✓ Usually this can be also a rather complicated process
- ✓ Data sharing without actual copy of the data & uptodate
- ✓ Shared data can be consumed by the own compute resources
- ✓ Non-Snowflake-Users can also access through a reader account

Data Sharing



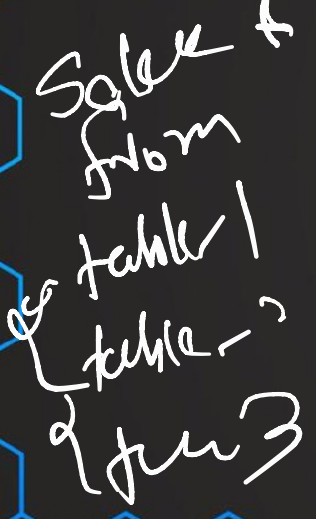
Materialized Views

Materialized views

- ✓ We have a view that is queried frequently and that a long time to be processed
- × Bad user experience
- × More compute consumption

What is a materialized view?

- ✓ Use any SELECT-statement to create this MV
- ✓ Results will be stored in a seperate table and this will be updated automatically based on the base table

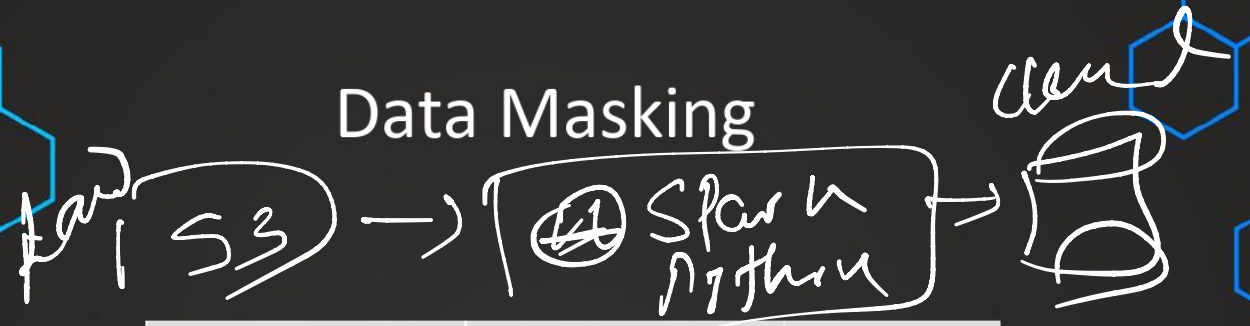


When to use MV?

- ✓ View would take a long time to be processed and is used frequently
- ✓ Underlaying data is change not frequently and on a rather irregular basis

Data Masking

Data Masking



FULL_NAME	EMAIL	PHONE
Lewiss MacDwyer	lmacdwyer0@un.org	262-665-9168
Ty Pettingall	tpettingall1@mayoclinic.com	734-987-7120
Marlee Spadazzi	mspadazzi2@txnews.com	867-946-3659

FULL_NAME	EMAIL	PHONE
L*******	l*******	##-###-##
T*******	t*******	##-###-##
M*******	m*******	##-###-##