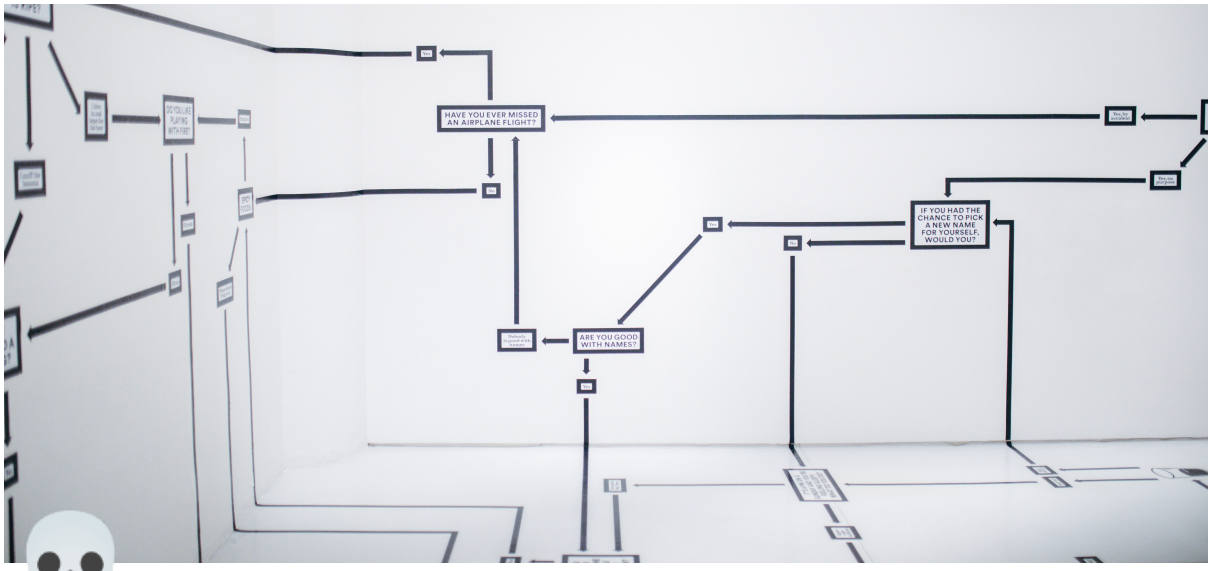

SCHEMA & VIEWS

SCHEMA

- Databases and schemas are used to organize ‘data store’ in Snowflake.
- A schema is a logical grouping of database objects(talble, vies, etc.). Each schema belongs to a single database.
- Together, a database and schema comprise a `namespace` in snowflake.

Database Schemas

A database schema defines how data is organized within a relational database.

Schemas commonly use visual representations to communicate the architecture of the database

Types of Schemas

1. Conceptual

- Conceptual schemas offer a big-picture view of what the system will contain, how it will be organized, and which business rules are involved. Conceptual models are usually created as part of the process of gathering initial project requirements.

2. Logical

- Logical database schemas are less abstract, compared to conceptual schemas. They clearly define schema objects with information, such as table names, field names, entity relationships, and integrity constraints—i.e. any rules that govern the database. However, they do not typically include any technical requirements.

3. Physical

- Physical database schemas provide the technical information that the logical database schema type lacks in addition to the contextual information, such as table names, field names, entity relationships, et cetera. That is, it also includes the syntax that will be used to create these data structures within disk storage.

VIEW

```
# A view is a subset of a database that is generated from a user query and
  gets stored as a permanent object.

# Views serve a variety of purposes, including combining, segregating,
  and protecting data.

# A view is a virtual table whose contents are defined by a query.
```

- A view allows the result of a query to be accessed as if it were a table. The query is specified in the CREATE VIEW statement.
- View serve a variety of purposes, including combining, segregating, and protecting data. For Example you can create separate views that meet the needs of different types of employees, such as Doctors and accountants at a hospital.

```
CREATE TABLE hospital_table (patient_id INTEGER,
                             patient_name VARCHAR,
                             billing_address VARCHAR,
                             diagnosis VARCHAR,
                             treatment VARCHAR,
                             cost NUMBER(10,2));
INSERT INTO hospital_table
        (patient_ID, patient_name, billing_address, diagnosis, treatment, cost)
    VALUES
        (1, 'Mark Knopfler', '1982 Telegraph Road', 'Industrial Disease',
            'a week of peace and quiet', 2000.00),
        (2, 'Guido van Rossum', '37 Florida St.', 'python bite', 'anti-venom',
            70000.00)
        ;
```

- Creating View for doctor and accountant

```
CREATE VIEW doctor_view AS
    SELECT patient_ID, patient_name, diagnosis, treatment FROM hospital_table;

CREATE VIEW accountant_view AS
    SELECT patient_ID, patient_name, billing_address, cost FROM hospital_table;
```

- Show all types of medical problems for each patient:

```
SELECT DISTINCT diagnosis FROM doctor_view;
```

- So, for security purpose we use view, because doctor doesn't need billing address of patient and accountant doesn't need treatment of patient, so by view we are protecting our data.

Database VIEW

A database view is a subset of a database and is based on a query that runs on one or more database tables. Database views are saved in the databse as named queries and can be used to save frequently used, complex queries.

Type of Views

1. Non-materialized views (referred to as "views")

- A non-materialized view's results are created by executing the query at the time that the view is referenced in a query. The results are not stored for future use. Performance is slower than with materialized views. Non-materialized views are the most common type of view.

  Any query expression that returns a valid result can be used to create a non-materialized view, such as:

  - Selecting some (or all) columns in a table.
  - Selecting a specific range of data in table columns.
  - Joining data from two or more tables.

  A non-materialized view can be recursive

2. Materialized view

- A materialized view’s results are stored, almost as though the results were a table. This allows faster access, but requires storage space and active maintenance, both of which incur additional ***costs***.
- In addition, materialized views have some restrictions that non-materialized views do not have.
- Materialized Views are designed to improve performance.
- Materialized Views contain a copy of a subset of the data in a table. Depending upon the amount of data in the table and in the materialized view, scanning the materialized view can be much faster than scanning the table.
- Materialized views also support clustering, and you can create multiple materialized views on the same data, with each materialized view being clustered on a different column, so that different queries can each run on the view with the best clustering for that query.

Views in SQL

- Views in SQL are considered as a virtual table. A view also contains rows and columns.
- To create the view, we can select the fields from one or more tables present in the database.
- A view can either have specific rows based on certain condition or all the rows of a table.