snowfl

views serve a variety of purposes, including combining, segregating, and protecting data. For example, you can create s

What is a View?

- 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.
- Views 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:

Show all of the types of medical problems for each patient:

Continued…

- A view can be used almost anywhere that a table can be used (joins, subqueries, etc.). For example, using the views created above:
- Show all of the types of medical problems for each patients:

```
select distinct diagnosis from doctor_view;
+--------------------+
| DIAGNOSIS          |
|--------------------|
| Industrial Disease |
| python bite        |
```

Show all the cost of each treatment (without showing personally identifying information about specific patients):

Continued...

- Show all the cost of each treatment (without showing personally identifying information about specific patients):

```
select treatment, cost
    from doctor_view as dv, accountant_view as av
    where av.patient_id = dv.patient_id;
+---------------------------+----------+
| TREATMENT                 |     COST |
|---------------------------+----------|
| a week of peace and quiet |  2000.00 |
| anti-venom                | 70000.00 |
+---------------------------+----------+
```

snowflake

ANALYTICS WITH ANAND

Show the cost of each medical problem for each patient (without showing personally identifying information about specific patients):

Types of Views

Snowflake supports two types of views:

- Non-materialized views (usually simply referred to as "views")
- Materialized views.

Show the cost of each medical procedure for each patient (without showing personally identifying information about specific patients):

NON-MATERIALIZED VIEW

The term “view” generically refers to all types of views; however, the term is used here to refer specifically to non-materialized views.

A view is basically a named definition of a query.

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.

snowflake

ANALYTICS WITH ANAND

Show the cost of each medical problem for each patient (without showing personally identifying information about specific patients):

MATERIALIZED VIEW

- A materialized view is a pre-computed data set derived from a query specification (the SELECT in the view definition) and stored for later use.
- Although a materialized view is named as though it were a type of view, in many ways it behaves more like a table.
- Because the data is pre-computed, querying a materialized view is faster than executing a query against the base table of the 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.
- This performance difference can be significant when a query is run frequently or is sufficiently complex.
- As a result, materialized views can speed up expensive aggregation, projection, and selection operations, especially those that run frequently and that run on large data sets.
- In addition, materialized views have some restrictions that non-materialized views do not have.

Show the cost per patient (without showing personally identifying information about specific patients):

Materialized Views Can Improve Performance

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.

Show the cost types for each patient (without showing personally identifying information about specific patients):

MATERIALIZED VIEW CONT...

- Materialized views are designed to improve query performance for workloads composed of common, repeated query patterns
- However, materializing intermediate results incurs additional costs.
- As such, before creating any materialized views, you should consider whether the costs are offset by the savings from re-using these results frequently enough.
- **For more Info** : Working with Materialized Views – Snowflake Documentation

snowflake

ANALYTICS WITH ANAND

Secure Views

- Both non-materialized and materialized views can be defined as *secure*.
- Secure views have advantages over standard views, including improved data privacy and data sharing; however, they also have some performance impacts to take into consideration.
- For More Info Check Documentation : Working with Secure Views – Snowflake Documentation

Show the cost of each medical problem for each patient (without showing personally identifying information about specific patients):

Advantages of Views

Views Enable Writing More Modular Code

- Views help you to write clearer, more modular SQL code.
- For example, suppose that your hospital database has a table listing information about all employees.
- You can create views to make it convenient to extract information about only the medical staff or only the maintenance staff.
- You can even create hierarchies of views.
- For example, you can create one view for the doctors, and one for the nurses, and then create the medical_staff view by referring to the doctors view and nurses view:

Show all of the types of medical (without showing personally identifying information about specific patients):

Advantages of Views

```
create table employees (id integer, title varchar);
insert into employees (id, title) values
    (1, 'doctor'),
    (2, 'nurse'),
    (3, 'janitor')
    ;

create view doctors as select * from employees where title = 'doctor';
create view nurses as select * from employees where title = 'nurse';
create view medical_staff as
    select * from doctors
    union
    select * from nurses
    ;
```

```
select *
    from medical_staff
    order by id;
+----+---------+
| ID | TITLE   |
|----+---------|
|  1 | doctor  |
|  2 | nurse   |
+----+---------+
```

snowflake

ANALYTICS WITH ANAND

Show the cost of each medical procedure for each patient (without showing personally identifying information about specific patients):

Advantages of Views

- In many cases, rather than writing one large and difficult-to-understand query, you can decompose the query into smaller pieces, and create a view for each of those pieces.
- This not only makes the code easier to understand, but in many cases it also makes the code easier to debug because you can debug one view at a time, rather than the entire query.
- One view can be referenced by many different queries, so views help increase code re-use.

Views Allow Granting Access to a Subset of a Table

Views allow you to grant access to just a portion of the data in a table(s).

For example, suppose that you have a table of medical patient records.

The medical staff should have access to all of the medical information (for example, diagnosis) but not the financial information (for example, the patient's credit card number).

The accounting staff should have access to the billing-related information, such as the costs of each of the prescriptions given to the patient, but not to the private medical data, such as diagnosis of a mental health condition.

You can create two separate views, one for the medical staff, and one for the billing staff, so that each of those roles sees only the information needed to perform their jobs.

Views allow this because you can grant privileges on a particular view to a particular role, without the grantee role having privileges on the table(s) underlying the view.

In the medical example:

- The medical staff would not have privileges on the data table(s), but would have privileges on the view showing diagnosis and treatment.
- The accounting staff would not have privileges on the data table(s), but would have privileges on the view showing billing information.

Limitations on Views

- The definition for a view cannot be updated (i.e. you cannot use ALTER VIEW or ALTER MATERIALIZED VIEW to change the definition of a view).

- To change a view definition, you must recreate the view with the new definition.

- Changes to a table are not automatically propagated to views created on that table.

  For example, if you drop a column in a table, the views on that table might become invalid.

- Views are read-only (i.e. you cannot execute DML commands directly on a view). However, you can use a view in a subquery within a DML statement that updates the underlying base table.

```
delete from hospital_table
    where cost > (select avg(cost) from accountant_view);
```

thank you!

snowflake

ANALYTICS WITH ANAND