fingerTips

Module-9

Views in SQL

1) Views in SQL

In SQL, a view is a virtual table that is based on the result set of a SELECT statement. It contains rows and columns just like a regular table, but its contents are derived from one or more tables in the database. Views are created by selecting a subset of columns or rows from one or more tables and saving the query as a view in the database.

Views in SQL are a powerful tool for simplifying complex queries, improving security, optimizing performance, and providing a consistent interface to the data in a database.

So, a view is:

i. Database object
ii. Created over a SQL query
iii. Doesn't store data
iv. Treated as a virtual table

2) Difference between a view and a table

In SQL, a table and a view are two different database objects with different characteristics and purposes.

A table is a collection of data stored in rows and columns, where each column represents a field and each row represents a record. Tables are the primary means of storing data in a database. They can be used to store, manipulate, and retrieve data.

On the other hand, a view is a virtual table that does not store data physically but retrieves data from one or more tables. A view is a query that is saved as an object in the database and can be referenced like a table. Views can be used to simplify complex queries, to provide a layer of abstraction over the underlying data, and to restrict access to certain data.

Here are some key differences between a view and a table:

i. Data Storage: A table is a physical object that stores data, while a view is a logical object that does not store data but

retrieves it from one or more tables.

ii. Data Modification: A table can be modified by inserting, updating, or deleting data, while a view cannot be directly modified. Changes made to the data in a view will affect the underlying tables.

iii. Data Retrieval: Both tables and views can be used to retrieve data from a database, but views are typically used to simplify complex queries and to provide a layer of abstraction over the underlying data.

iv. Data Security: Views can be used to restrict access to certain data by allowing users to see only the data they have permission to access. Tables do not offer this level of security.

3) Some reasons of creating views in SQL

There are several reasons we might want to use views in SQL:

i. Simplify complex queries: Views can be used to simplify complex queries that involve multiple tables by creating a virtual table that combines data from different tables into a single result set.

ii. Enhance security: Views can be used to restrict access to sensitive data by giving users access to only certain columns in a table or by limiting the rows that are returned by a query.

iii. Improve performance: Views can be used to pre-compute the results of a complex query and store them as a virtual table, which can improve performance when the query is executed repeatedly.

iv. Provide a consistent interface: Views can be used to provide a consistent interface to the data in the database, even if the underlying schema or table structure changes.

4) What are some applications of views in SQL

Some common applications of views in SQL include:

i. Reporting: Views can be used to create reports that combine data from multiple tables or databases, and

provide a consistent interface to the data.

ii. Data analysis: Views can be used to simplify complex queries and perform data analysis tasks, such as aggregating data or filtering out certain rows or columns.

iii. Security: Views can be used to restrict access to sensitive data by giving users access to only certain columns or rows in a table.

iv. Performance optimization: Views can be used to pre-compute the results of a complex query and store them as a virtual table, which can improve performance when the query is executed repeatedly.

5) Create a view containing details of users and the number of times they were delivered food in less than 30 minutes

Create a table first:

```
select u.*, count(*)
from orders1 o join users1 u
on o.user_id=u.user_id
where o.delivery_time<30
group by user_id;
```

Now, create view:

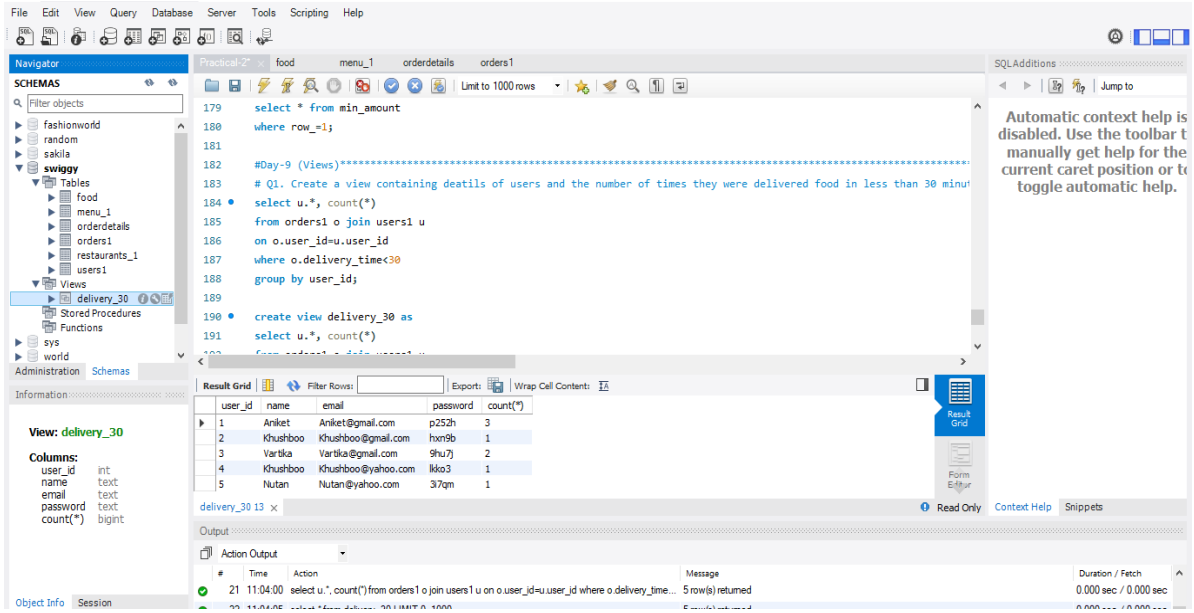
```
create view delivery_30 as
select u.*, count(*)
from orders1 o join users1 u
on o.user_id=u.user_id
where o.delivery_time<30
group by user_id;
```

To see view:

```
select * from delivery_30;
```

Note: It can be used by Swiggy to analyze delivery timings and improve the number of counts of deliveries done in less than 30 minutes.

In LHS we can see, a view is created:



6) Create a view containing details of all those users who ordered food at-least once for rupees 500 or more

Create a table first:

```
select * from users1 where user_id in
(select o.user_id
from orders1 o join users1 u
on o.user_id=u.user_id
where o.amount>=500);
```

Now, create view:

```
create view amount_500 as
select * from users1 where user_id in
(select o.user_id
from orders1 o join users1 u
on o.user_id=u.user_id
where o.amount>=500);
```

To see view:

```
select * from amount_500;
```

7) Create a view containing details of all those restaurants offering food for less than 120

Create a table first:

```
select * from restaurants_1 where r_id in
(select m.r_id
from menu_1 m join restaurants_1 r
on m.r_id=r.r_id
where m.price<120);
```

Now, create view:

```
Create view price_120 as
select * from restaurants_1 where r_id in
(select m.r_id
from menu_1 m join restaurants_1 r
on m.r_id=r.r_id
where m.price<120);
```

To see view:

```
select * from price_120;
```

8) Drop a view in SQL

To drop a view in SQL:

```
drop view price_120;
```