View in SQL

What is a View in SQL?

A view in SQL is like a virtual table. It doesn't store data itself but shows data from other tables based on a saved SQL query. Each time you use a view, it runs the query and shows the results.

Key Benefits of Views:

- Simplifies complex queries by hiding joins and filters inside a single object.
- Improves security by limiting access to certain data.
- Customizes data display for different users or purposes.

CREATE VIEWS in SQL

We can create a view using **CREATE VIEW** statement. A View can be created from a single table or multiple tables.

Syntax:

```
CREATE VIEW view_name AS

SELECT column1, column2.....

FROM table_name

WHERE condition;
```

Key Terms:

view_name: Name for the View

table_name: Name of the table

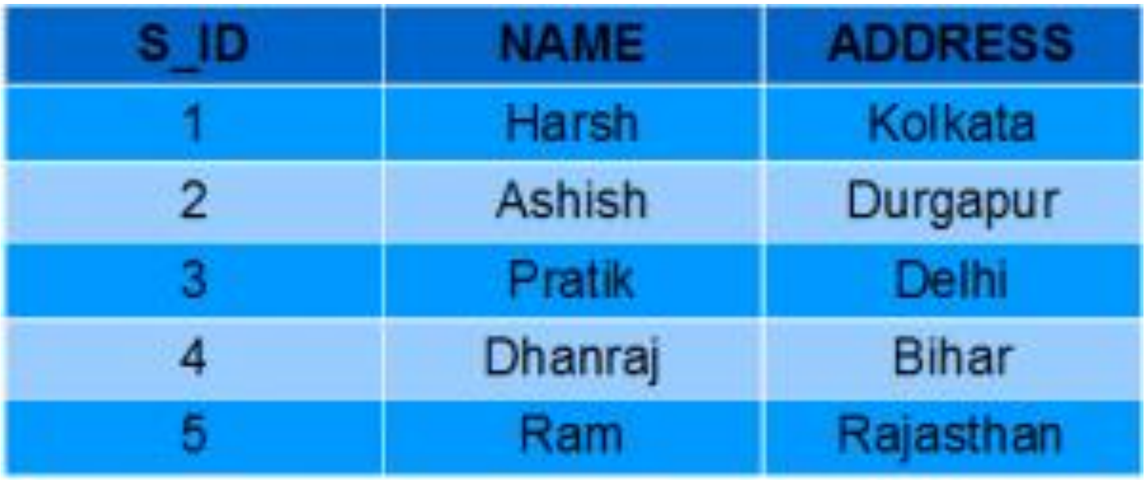
condition: Condition to select rows

Examples

Student Details

```
-- Create StudentDetails table
CREATE TABLE StudentDetails (
  S_ID INT PRIMARY KEY,
  NAME VARCHAR(255),
  ADDRESS VARCHAR(255)
);

INSERT INTO StudentDetails (S_ID, NAME, ADDRESS)
VALUES
  (1, 'Harsh', 'Kolkata'),
  (2, 'Ashish', 'Durgapur'),
  (3, 'Pratik', 'Delhi'),
  (4, 'Dhanraj', 'Bihar'),
  (5, 'Ram', 'Rajasthan');
```

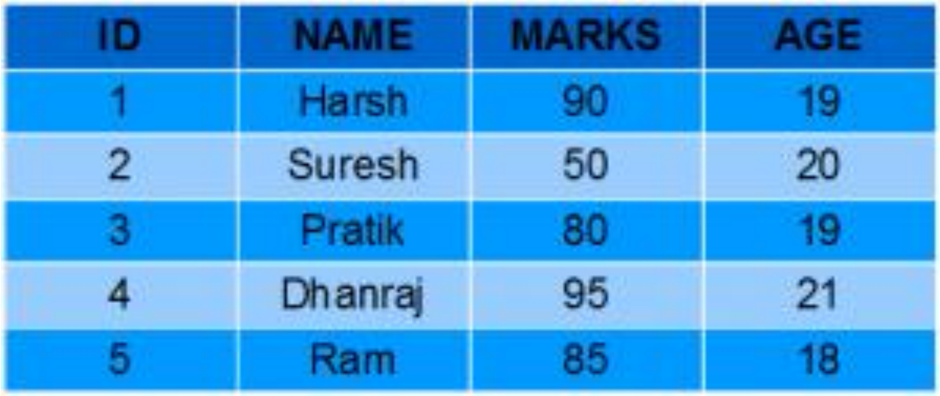


S_ID	NAME	ADDRESS
1	Harsh	Kolkata
2	Ashish	Durgapur
3	Pratik	Delhi
4	Dhanraj	Bihar
5	Ram	Rajasthan

Student Marks

```
-- Create StudentMarks table
CREATE TABLE StudentMarks (
  ID INT PRIMARY KEY,
  NAME VARCHAR(255),
  Marks INT,
  Age INT
);

INSERT INTO StudentMarks (ID, NAME, Marks, Age)
VALUES
  (1, 'Harsh', 90, 19),
  (2, 'Suresh', 50, 20),
  (3, 'Pratik', 80, 19),
  (4, 'Dhanraj', 95, 21),
  (5, 'Ram', 85, 18);
```



ID	NAME	MARKS	AGE
1	Harsh	90	19
2	Suresh	50	20
3	Pratik	80	19
4	Dhanraj	95	21
5	Ram	85	18

Example 1: Creating a Simple View from a Single Table

In this example, we will create a View named DetailsView from the table StudentDetails.

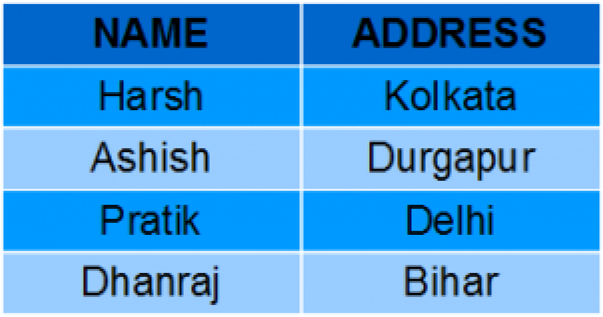
Query:

```
CREATE VIEW DetailsView AS
SELECT NAME, ADDRESS
FROM StudentDetails
WHERE S_ID < 5;
```

Use the below query to retrieve the data from this view

```
SELECT * FROM DetailsView;
```

Output:



NAME	ADDRESS
Harsh	Kolkata
Ashish	Durgapur
Pratik	Delhi
Dhanraj	Bihar

Here, we will create a view named StudentNames from the table StudentDetails.

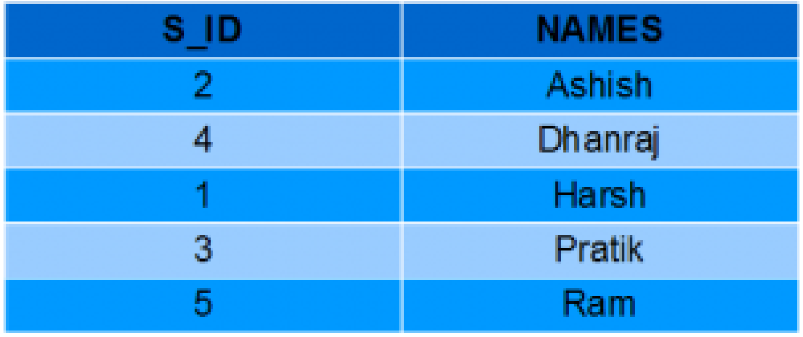
Query:

```
CREATE VIEW StudentNames AS
SELECT S_ID, NAME
FROM StudentDetails
ORDER BY NAME;
```

If we now query the view as,

```
SELECT * FROM StudentNames;
```

Output:



S_ID	NAMES
2	Ashish
4	Dhanraj
1	Harsh
3	Pratik
5	Ram

Example 2: Creating a View From Multiple Tables

In this example we will create a View MarksView that combines data from both tables StudentDetails and StudentMarks. To create a View from multiple tables we can simply include multiple tables in the SELECT statement.

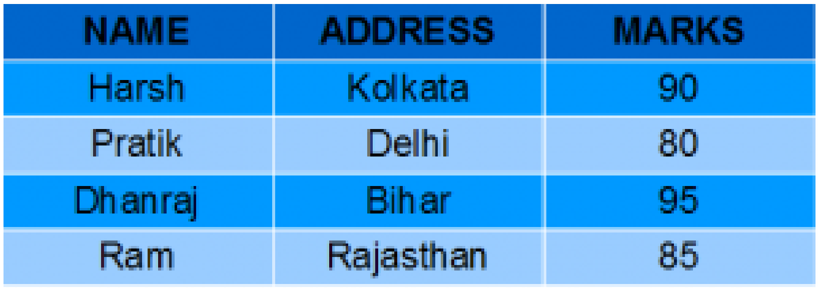
Query:

```
CREATE VIEW MarksView AS
SELECT sd.NAME, sd.ADDRESS, sm.MARKS
FROM StudentDetails sd, StudentMarks sm
WHERE sd.NAME = sm.NAME;
```

To display data of View MarksView:

```
    SELECT * FROM MarksView;
```

Output:



NAME	ADDRESS	MARKS
Harsh	Kolkata	90
Pratik	Delhi	80
Dhanraj	Bihar	95
Ram	Rajasthan	85

Managing Views: Listing, Updating, and Deleting

1. Listing all Views in a Database

We can list all the Views in a database, using the **SHOW FULL TABLES** statement. A View can be created from a single table or multiple tables

```
 USE "database_name";
SHOW FULL TABLES WHERE table_type LIKE "%VIEW";
```

2. Deleting a View

SQL allows us to delete an existing View. We can delete or drop View using the DROP statement. Here's how to remove the MarksView:

```
DROP VIEW view_name;
```

- **Example:** In this example, we are deleting the View MarksView.

```
DROP VIEW MarksView;
```

3. Updating a View Definition

If we want to update the existing data within the view, use the **UPDATE** statement.

```
UPDATE view_name
SET column1 = value1, column2 = value2...., columnN = valueN
WHERE [condition];
```

If you want to update the view definition without affecting the data, use the **CREATE OR REPLACE VIEW** statement. For example, let's add the Age column to the MarksView:

```
CREATE OR REPLACE VIEW view_name AS
SELECT column1, column2, ...
FROM table_name
WHERE condition;
```

Note: Not all views can be updated using the **UPDATE** statement.

Rules to Update Views in SQL:

Certain conditions need to be satisfied to update a view. If any of these conditions are not met, the view cannot be updated.

1. The SELECT statement which is used to create the view should not include GROUP BY clause or ORDER BY clause.

2. The SELECT statement should not have the DISTINCT keyword.

3. The View should have all NOT NULL values.

4. The view should not be created using nested queries or complex queries.

5. The view should be created from a single table. If the view is created using multiple tables, then we will not be allowed to update the view.