SQL Server Views

Views in SQL Server are virtual tables based on the result set of an SQL statement. They simplify complex queries, improve readability, and can help with security by restricting access to specific columns or rows in tables.

1. Creating a New View

To create a view in SQL Server, use the `CREATE VIEW` statement, which defines the view name and the SELECT query that it represents. Here's an example:

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

Alternatively, in SQL Server Management Studio (SSMS):
1. Navigate to Object Explorer > Databases > YourDatabase > Views.
2. Right-click on Views and select New View....
3. Use the graphical interface to select tables, columns, and add conditions, then save.

2. Renaming a View

You can rename a view in SSMS or by using Transact-SQL (T-SQL):
- In SSMS:
  1. Navigate to Object Explorer > Views.
  2. Right-click the view and select Rename.
- Using T-SQL:

```
EXEC sp_rename 'old_view_name', 'new_view_name';
```

3. Listing Views in SQL Server

To list all views in a database, you can query the `sys.views` or `INFORMATION_SCHEMA.VIEWS` system catalog views:

```
SELECT name FROM sys.views;
Or:
SELECT TABLE_NAME FROM INFORMATION_SCHEMA.VIEWS;
```

4. Getting View Information

To get details about a specific view, such as its definition, you can use `sp_helptext`:

```
EXEC sp_helptext 'view_name';
```

Or to check the columns of a view:

```
SELECT COLUMN_NAME, DATA_TYPE
FROM INFORMATION_SCHEMA.COLUMNS
WHERE TABLE_NAME = 'view_name';
```

5. Removing a View

To delete a view, use the `DROP VIEW` statement:

```
DROP VIEW view_name;
```

To remove multiple views:

```
DROP VIEW view_name1, view_name2;
```

6. Creating an Indexed View

Indexed views can improve performance, especially when data modifications in the underlying tables are infrequent. To create an indexed view:

1. Ensure SET Options: Before creating an indexed view, ensure that specific `SET` options are enabled.

```
SET ANSI_NULLS ON;
SET QUOTED_IDENTIFIER ON;
```

2. Create the View with SCHEMABINDING: This ensures the view is bound to the schema of the underlying tables.

```
CREATE VIEW view_name
WITH SCHEMABINDING
AS
SELECT column1, column2
FROM table_name
WHERE condition;
```

3. Create a Unique Clustered Index:

```
CREATE UNIQUE CLUSTERED INDEX IX_View ON view_name (column1);
```

Indexed views can be highly beneficial for performance when used on tables with a low frequency of updates, as maintaining the index can impact performance if the underlying data changes frequently.