
بسم الله الرحمن الرحيم

DATABASE MID-TERM PRESENTATION

TOPIC: VIEWS

- **Created and Presented by:**
- **Malik.M.Huzaifa (65)**
- **Hamza (47)**
- **Mubashir Ali (73)**
- **Janib Ali (61)**
- **M.Mujeeb (87)**

Contents:

- What is Views
- Type of Views
- Importance of Views
- Uses of Views
- Some Exemplary Programs
- Conclusion

What is a View?

SQL has a special version of tables called View, which is a virtual table that is compiled in runtime. A View is just an SQL statement, and the data associated with it is not physically stored in the view but is stored in the base tables of it.

It can contain all the rows and columns of a table or only a few selected rows and columns if there is a need to restrict the access. Depending on the written SQL query used to create the view, it can be created from one or many tables.

Views can be used to structure data in ways for users to find it natural, simplify complex queries, restrict access to data, and summarize data from several tables to create reports.

Managing Views

There are different aspects related to managing views, which are defined here.

Creating view: Views can be created using the “create view” statement. The view is defined by a query that references materialized views, tables, or other views.

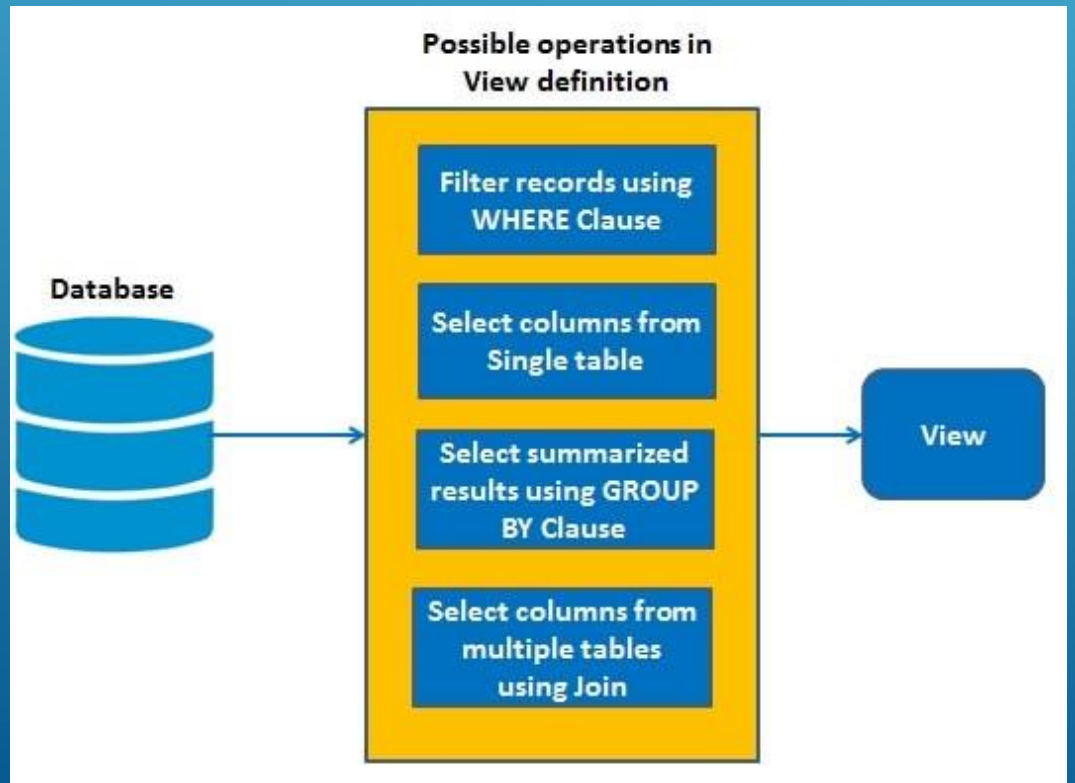
Renaming view: Views can be renamed, and it should be ensured that all objects that reference the old name of the view now should have a new name.

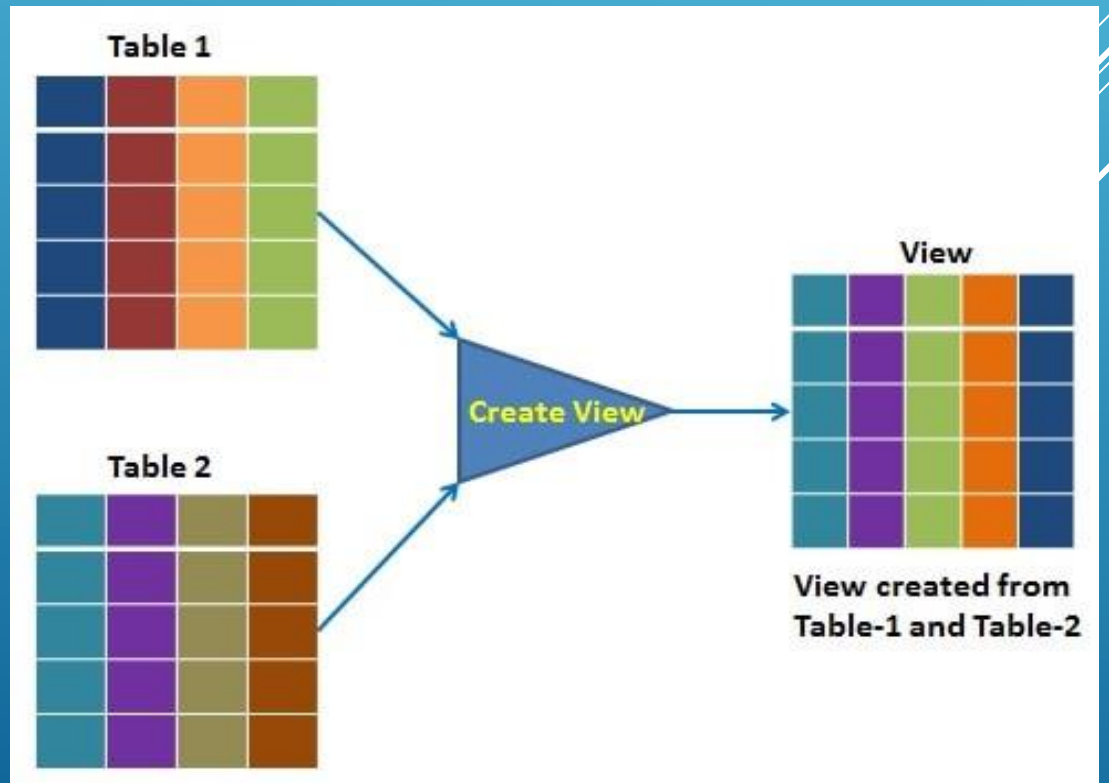
Listing views: All the views in the SQL Server Database can be listed by querying the system catalog view.

Removing view: Using the “drop view” statement, an existing view can be removed.

Static and Dynamic Views

Dynamic views contain data from one or two tables and automatically include all columns of the specified table or tables. Dynamic views can be updated dynamically when related or extended objects are created or modified. Dynamic views can be updated manually when related objects or extended objects are created or modified.
Static views contain data from several tables and the necessary columns of these tables must be specified in the SELECT and WHERE clauses of the static view.





Types of Views

There are two types of views in the SQL Server, namely System Defined Views and User Defined Views.

1. System Defined Views

The System Defined Views are predefined views that already exist in the SQL Server database, such as Tempdb, Master, and temp. Each of the databases has its own properties and functions.

The template database for all User Defined views is from the Master database. It contains many predefined views that are templates for tables and other databases. It contains nearly 230 of the predefined views.

System Defined Views will be automatically attached to all User Defined databases. And these provide information about the database, tables, and all the properties of the database and tables. There are three types of System defined views, Information Schema, Catalog View, and Dynamic Management View.

❑ Information Schema

There are twenty different schema views in the SQL server. They are used to display the physical information of the database, such as tables, constraints, columns, and views. This view starts with INFORMATION_SCHEMA and followed by the View Name. INFORMATION_SCHEMA.CHECK_CONSTRAINTS is used to receive information about any constraint available in the database.

A constraint is used on a particular column in a table to ensure that certain data rules are followed for the column. INFORMATION_SCHEMA.COLUMNS is used to receive information about the table columns such as table name, column name, the position of the column, default value, etc. To return the views present in the current database, INFORMATION_SCHEMA.VIEWS is used.

❑ Catalog View

These are used to return information used by the SQL server. Catalog views provide an efficient way to obtain, present, and transform custom forms of information. But they do not include any information about backup, replication, or maintenance plans, etc. These views are used to access metadata of databases, and the names and column names are descriptive, helping a user to query what is expected.

❑ Dynamic Management View

These were introduced in the SQL server in 2005. The administer can get information about the server state to diagnose problems, monitor the health of the server instance, and tune performance through these views. The Server-scoped Dynamic Management View is only stored in the Master database, whereas the Database-scoped Dynamic Management View is stored in each database.

Results | Messages

	TABLE_CATALOG	TABLE_SCHEMA	TABLE_NAME	CONSTRAINT_CATALOG	CONSTRAINT_SCHEMA	CONSTRAINT_NAME
1	practice	dbo	Employee	practice	dbo	PK_Employee
2	practice	dbo	Course	practice	dbo	PK_Course
3	practice	dbo	salary	practice	dbo	My_Constraint
4	practice	dbo	pkt	practice	dbo	PK_pkt
5	practice	dbo	salary	practice	dbo	My_Constraint3
6	practice	dbo	salary	practice	dbo	My_Constraint31
7	practice	dbo	fkt	practice	dbo	FK_fkt_pktpankaj
8	practice	dbo	tab	practice	dbo	UQ__tab__C2B02CE4B4BB56BF
9	practice	dbo	tab	practice	dbo	UQ__tab__DF906FE5DED35E48
10	practice	dbo	tab	practice	dbo	UQ__tab__DC5021AB42E468E7

2. User Defined Views

These are the types of views that are defined by the users. There are two types under User Defined views, Simple View and Complex View.

❑ Simple View

These views can only contain a single base table or can be created only from one table. Group functions such as MAX(), COUNT(), etc., cannot be used here,

and it does not contain groups of data.

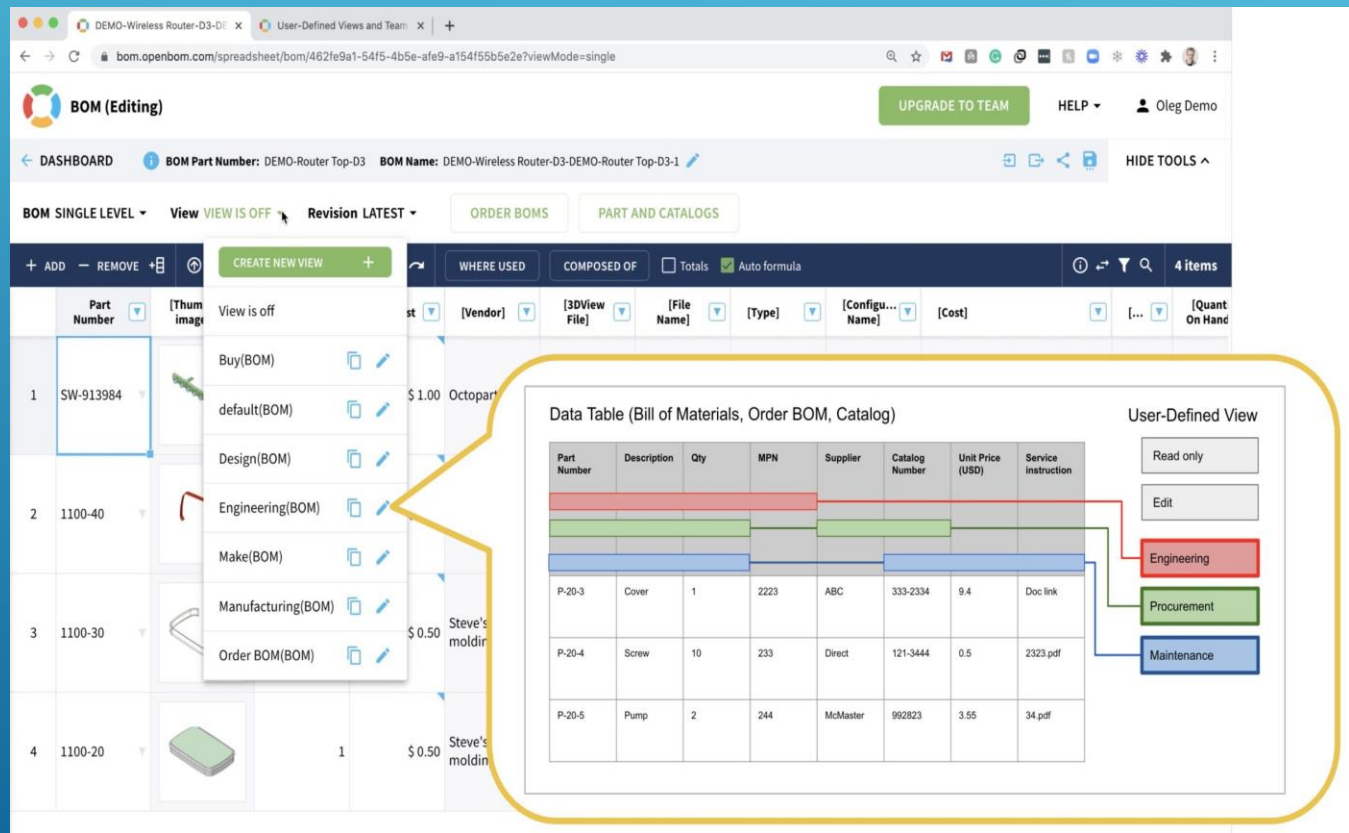
By using Simple View, DML operations can be performed.

Insert, delete, and update are directly possible, but Simple

View does not contain group by, pseudocolumn like rownum,

distinct, columns defined by expressions. Simple view also does

not include NOT NULL columns from the base tables.



❑ Complex View

These views can contain more than one base table or can be constructed on more than one base table, and they contain a group by clause, join conditions, an order by clause. Group functions can be used here, and it contains groups of data. Complex views cannot always be used to perform DML operations.

Insert, delete, and update cannot be applied directly on complex views. But unlike Simple Views, Complex Views can contain group by, pseudocolumn like rownum, distinct, columns defined by expressions. NOT NULL columns can be included in complex views while they are not selected by the Simple View.

There are other views, such as Inline View and Materialized View. The inline view is based on a subquery in FROM clause, the subquery creates a temporary table, and this simplifies the complex query.

These views are used to write complex SQL queries without the join and subqueries operations. The materialized view stores the definition and even the data. Replicas of data are created by storing it physically. This view reduces the processing time for regenerating the whole data.

Importance of Views

To the database, the View is the same as a real table for a user with the set of column names and row data. SQL creates a custom view by giving the View the same name as a table name and store a definition of the View in the database.

A view is used for security purposes in the database and acts as an intermediate between real tables schema and programmability. It also restricts the user from viewing specific columns and rows; Views always represent custom output, which is mentioned in the query and returns that data defined in the query at the time of creation.

Views also provide these specific features:

- **Consistency**

  Views always represent the same presentation, even when we perform normalizations to the Tables schema or if we rename the tables or restructure the table.

- **Data Integrity**

  When a data is viewed or entered by a View in a database, the DBMS will automatically check the data to ensure that it meets the specified integrity constraints.

- **Security**

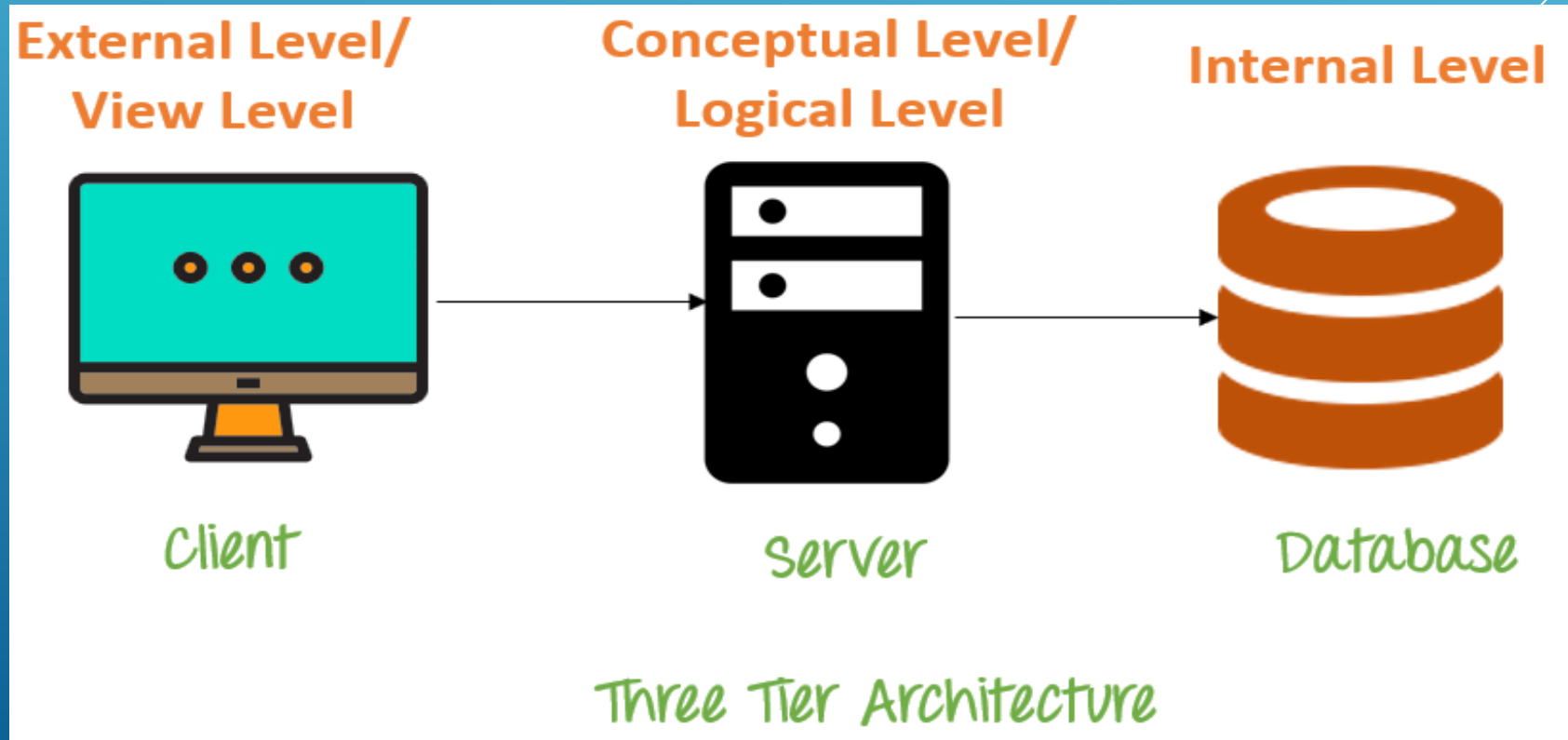
  Views provide security from unauthorized access to data. Each user is given permission to access the database from a View that contains a specified data that a user is authorized to see.

Structural Unity

Views provide a custom view of database structure, which represents a database same as a set of a virtual or custom table that is useful for a user and presents only relevant data.

Data Independence

Views provide data independence to an application as the application depends on a view but not on a real table. Therefore, any change in the table will not affect the application, and an application will always be independent of the table schema design.



Uses of Views

- **Uses of a View :**
  A good database should contain views due to the given reasons.

1. **Restricting data access –**
   Views provide an additional level of table security by restricting access to a predetermined set of rows and columns of a table.
2. **Hiding data complexity –**
   A view can hide the complexity that exists in a multiple table join.
3. **Simplify commands for the user –**
   Views allows the user to select information from multiple tables without requiring the users to actually know how to perform a join.
4. **Store complex queries –**
   Views can be used to store complex queries.
5. **Rename Columns –**
   Views can also be used to rename the columns without affecting the base tables provided the number of columns in view must match the number of columns specified in select statement. Thus, renaming helps to hide the names of the columns of the base tables.
6. **Multiple view facility –**
   Different views can be created on the same table for different users.

Some Exemplary Programs:

- 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;

SELECT * FROM DetailsView;
```

Output:

NAME	ADDRESS
Mubashir	Sir-e-Ghat
Janib	Latifabad
Huzaifa	Hala Naka
Hamza	Railway Station Road
Mujeeb	Phuleli

Some Exemplary Programs:

- In this example, 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;

SELECT * FROM StudentNames;
```

Output:

S-ID	NAMES
73	Mubashir
61	Janib
65	Huzaifa
47	Hamza
87	Mujeeb

Some Exemplary Programs:

- In this example we will create a View named MarksView from two 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 StudentDetails.NAME, StudentDetails.ADDRESS,
StudentMarks.MARKS
FROM StudentDetails, StudentMarks
WHERE StudentDetails.NAME = StudentMarks.NAME;

SELECT * FROM MarksView;
```

Output:

NAME	ADDRESS	MARKS
Mubashir	Sir-e-Ghat	80
Janib	Latifabad	95
Huzaifa	Hala Naka	78
Hamza	Railway Station Road	87
Mujeeb	Phuleli	99

Conclusion

This fully presentation describes the types of views in SQL. A View in SQL is defined and explained in detail, and the different ways in which Views are managed is also defined. The different types of views in SQL, such as System Defined Views and User Defined Views, are described in detail along with the various subtypes under each type, with the Uses of Views and some Exemplary Programs to construct View in Database/SQL.

I hope you All enjoyed and gained some prior knowledge with my presentation!

THANK YOU!