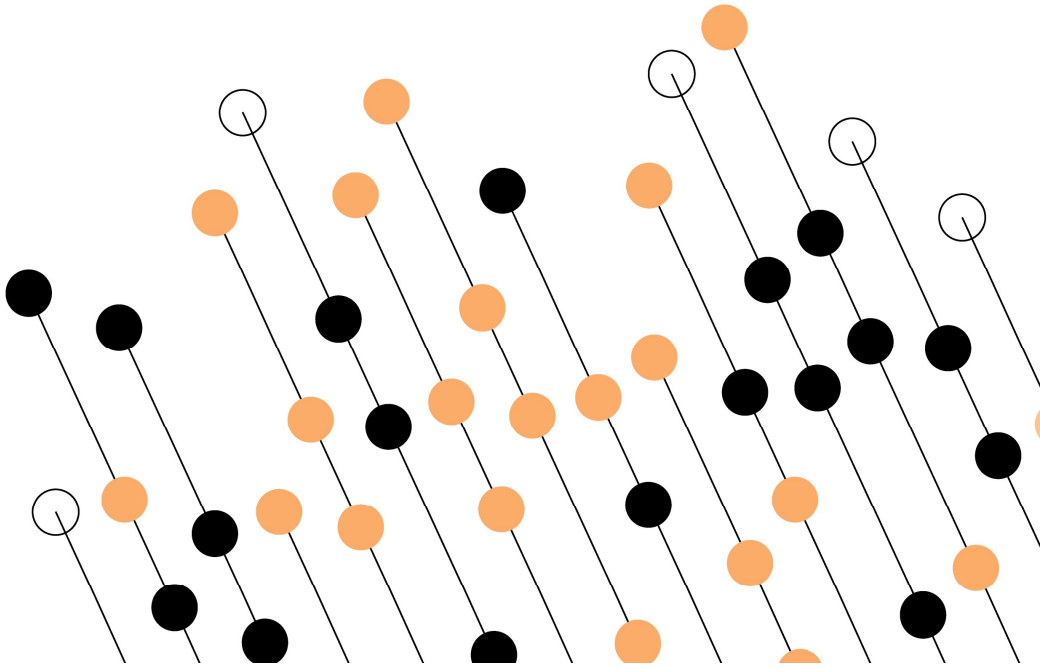
SQL: Views



View

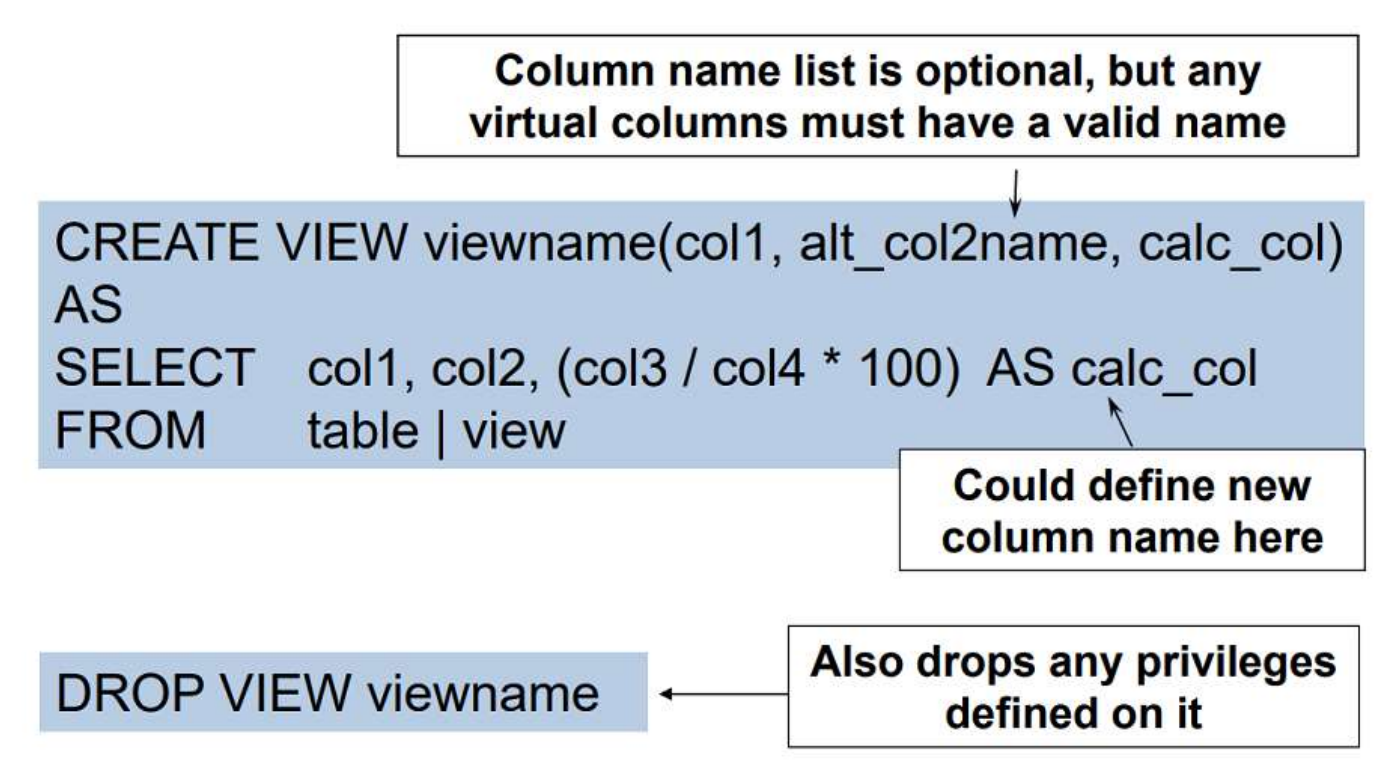
A **view** is a select query which has been pre-created and stored within the database.

- The view will not give any performance advantages over the original query.
- For data retrieval, a view can be used in the same way as an ordinary table.
- You can simplify the use of a difficult query for other users where they do not have to provide the full query again, but can refer to the view by name.
- As the views are called within select queries, by default views cannot include ORDER BY clause when they are created. This is to avoid an expensive sort operation. ORDER BY however can be applied to a view when it is used.

View: advantages

- Stores SELECT statements.
- Acts like a table.
- Reusable.
- Simplifies complex queries.
- Limits access to sensitive data.

Create / drop view



Creating views

Northwind's pricing team needs to work on a regular basis with the individual and the average prices of the active (not discontinued) products from different categories.

Let's create a query that would provide all necessary data.

```
SELECT c.CategoryName, p.ProductName, p.UnitPrice
FROM Products p
        JOIN Categories c ON p.CategoryID = c.CategoryID
WHERE p.Discontinued = 0
ORDER BY CategoryName,ProductName
```

Creating views

We can save the query as a view and thus we will be able to use it repetitively. Each time we run it, the result set will contain the current data (i.e. all changes to products and their prices will be there).

```
CREATE VIEW ProductsByCategory AS
SELECT c.CategoryName, p.ProductName, p.UnitPrice
FROM Products p
JOIN Categories c ON p.CategoryID = c.CategoryID
WHERE p.Discontinued = 0
-- ORDER BY CategoryName,ProductName
```

Why is ORDER BY not included in the view?

Using views

Once the view is created, we can run queries on it as if it is an ordinary table. Here are a couple of examples.

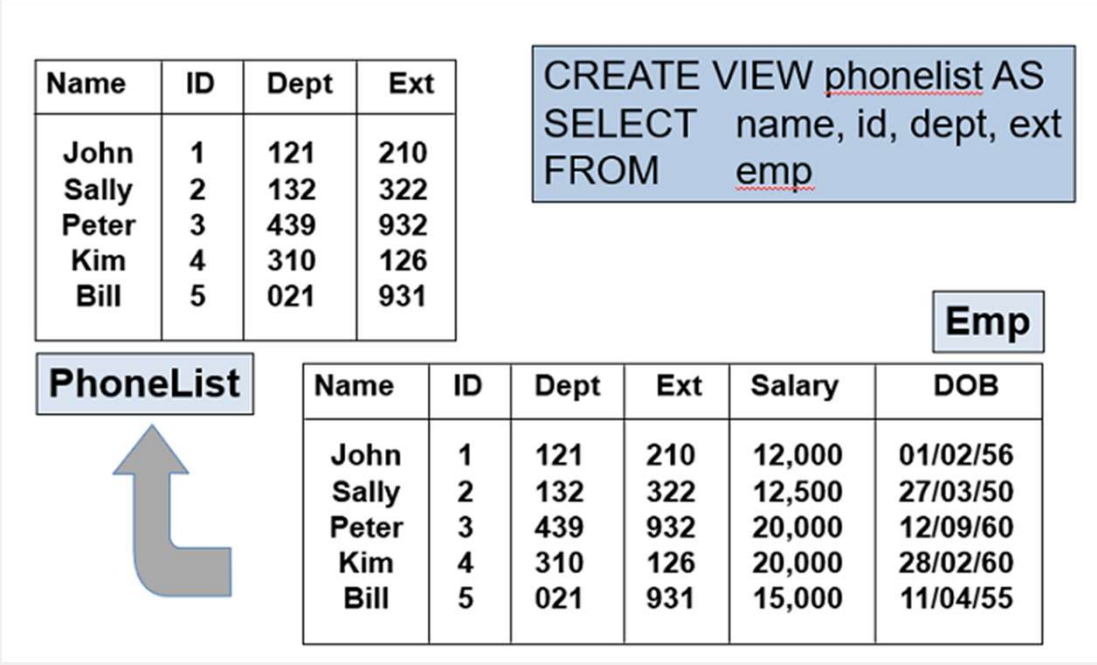
```
SELECT CategoryName,ProductName, UnitPrice
FROM ProductsByCategory
ORDER BY CategoryName,UnitPrice DESC

SELECT CategoryName,AVG(UnitPrice) AS AvCatPrice
FROM ProductsByCategory
GROUP BY CategoryName
HAVING AVG(UnitPrice) > 20
ORDER BY CategoryName,AvCatPrice DESC
```

Much easier!

Views and data security

In most systems there will be some data that should be generally available and some that should only be accessible by specific users. Views can be given their own access privileges, separate from those of the underlying tables.

In the above example, the privilege to select from the **PhoneList** view could be given to everyone and the access to the **Emp** table could be restricted to staff from the personnel department.



Views with check option

By adding the **WITH CHECK OPTION** to the view definition, the engine will ensure that data being added via the view is accessible via the same view.

- This will not allow entry of non-dept3 staff
- Any data inserted or updated via the view must satisfy the view's underlying WHERE clause

```
CREATE VIEW dept3_staff AS
SELECT      *
FROM        salesperson
WHERE       dept_no = 3
WITH CHECK OPTION
```

View restrictions

A view cannot be used for INSERT or UPDATE if it:

- contains the keyword DISTINCT.
- contains a sub-query, a GROUP BY or a HAVING clause.

A view cannot be used for INSERT if it:

- omits any NOT NULL columns that do not have defaults.
- contains calculated or aggregate columns.

Views based on joins typically only allow one underlying table at a time to be updated.