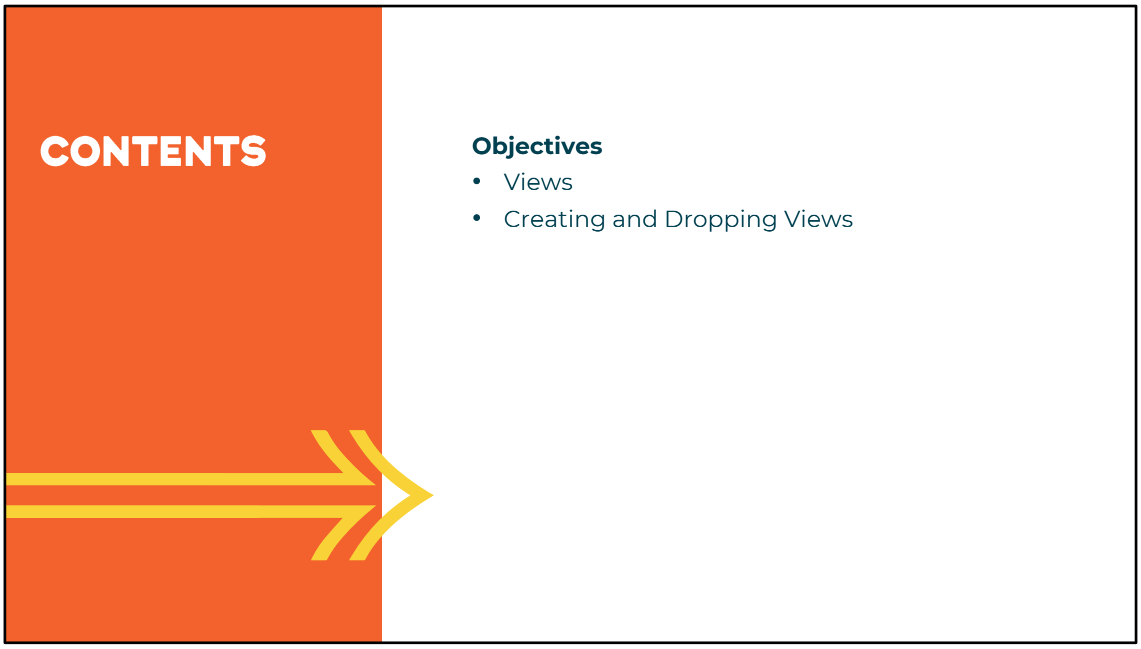
SQL Server

Maintaining Database Objects

This module will enable the learner to write SQL to maintain database objects. Specifically the learner will be able to:

- Use CREATE, ALTER and DROP to maintain table including column, Primary and Foreign Key constraints.
- Use CREATE and DROP to maintain single column, multiple columns and unique indexes.
- Use CREATE, ALTER and DROP to maintain views with and without the check option.
- Use SELECT, INSERT, UPDATE and DELETE to maintain records using a view.
- Use CREATE and DROP to maintain users and GRANT and REVOKE to maintain security options.

Code examples will be used to explain and demonstrate the content and for leaners to imitate. Supporting exercises will provide practice opportunities. At the end of the module there is a short quiz to review the topic.



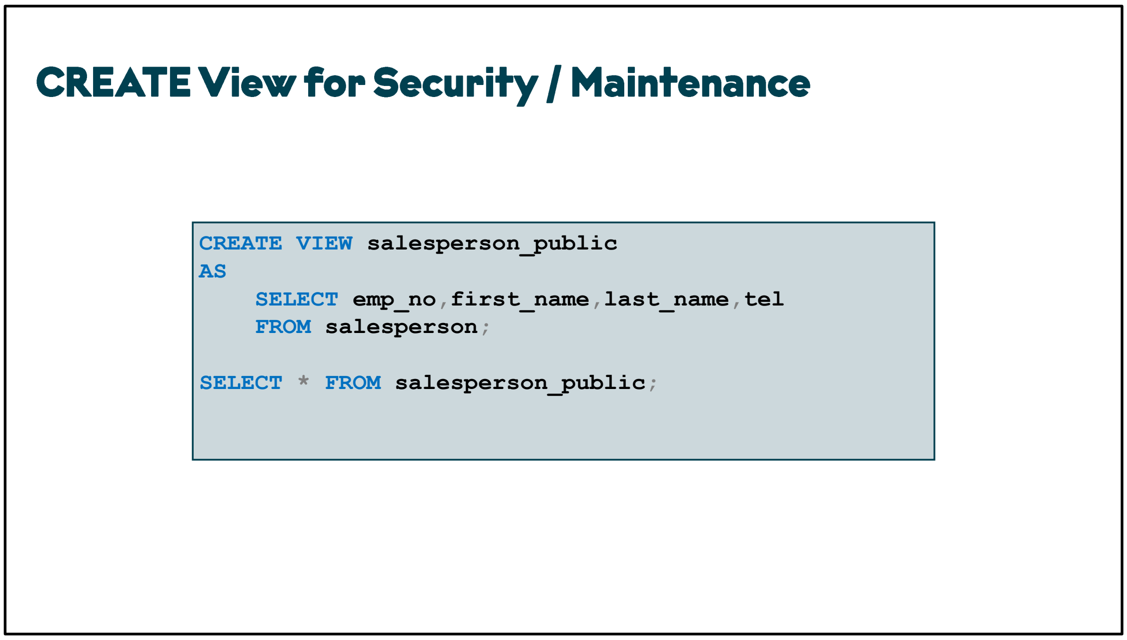
CONTENTS

Objectives

- Views
- Creating and Dropping Views

A relational database is made up of several components:

- Tables are where all the data in a database is stored.
- Indexes are used to order the records in those tables to improve the speed that records can be processed for given queries.
- Views are virtual tables based on the result-set of an SQL statement. A view contains rows and columns which are the fields from one or more real tables in the database. Note that no data is actually stored in the view.
- Users and Security controls which people have access to a database and what actions they can do when they have that access.

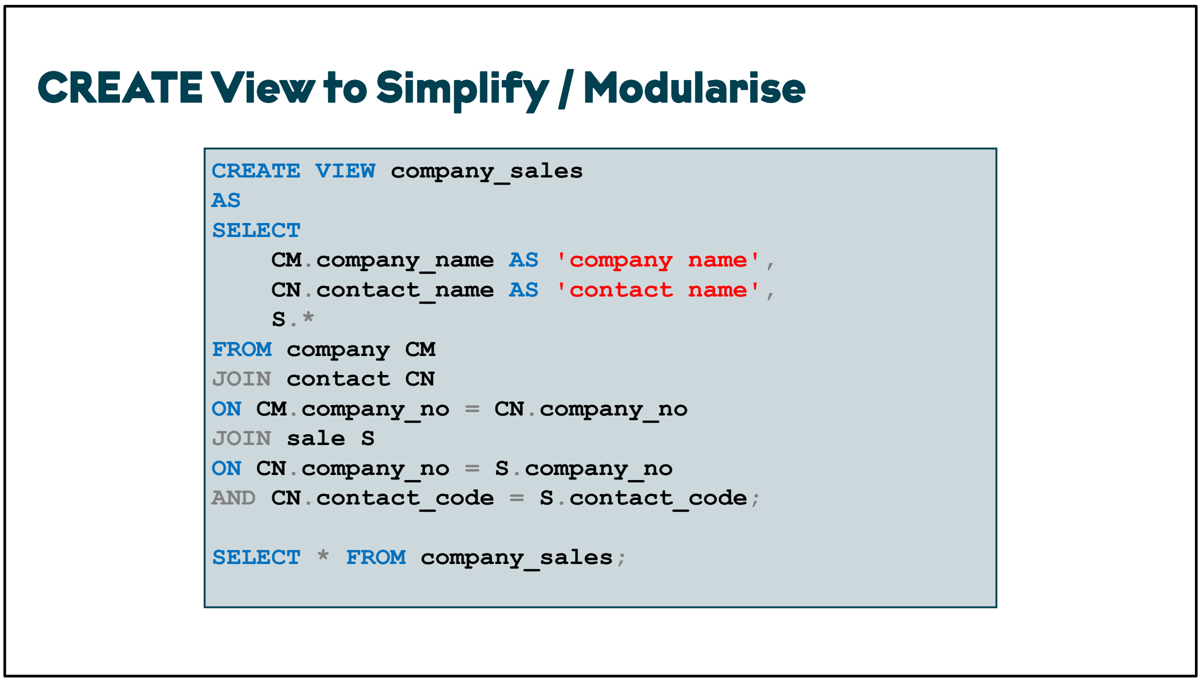


CREATE View for Security / Maintenance

```
CREATE VIEW salesperson_public
AS
    SELECT emp_no,first_name,last_name,tel
    FROM salesperson;

SELECT * FROM salesperson_public;
```

In this case the view is simply a subset of columns from a given table. This could be for security purposes. If columns on the table are confidential a view can be created of just the non-confidential columns. Then by restricting specific users to just use the view they can only access the non-confidential data and not the confidential data. Other users with appropriate access can still view and maintain both the confidential and non-confidential data using the table.
To access the data the view is simply specified as a table would be in a query.



CREATE View to Simplify / Modularise

```sql
CREATE VIEW company_sales
AS
SELECT
    CM.company_name AS 'company name',
    CN.contact_name AS 'contact name',
    S.*
FROM company CM
JOIN contact CN
ON CM.company_no = CN.company_no
JOIN sale S
ON CN.company_no = S.company_no
AND CN.contact_code = S.contact_code;

SELECT * FROM company_sales;
```

A view can be used to simplify a complex or modularise a frequent piece of SQL. In this case joining data from three tables. Once the view has been created using the full SQL the data can be subsequently accessed as if it is a single table.

ALTER View

```
ALTER VIEW salesperson_public
AS
   SELECT emp_no,first_name,last_name,tel,salary
   FROM salesperson;
```

ALTER VIEW can be used to amend the definition of a view.

DROP View

```
DROP VIEW salesperson_public;
```

DROP can be used to delete view. Note the underlying records on the table or tables that make up the view will not be deleted as the view is just a logical representation of the underlying data.

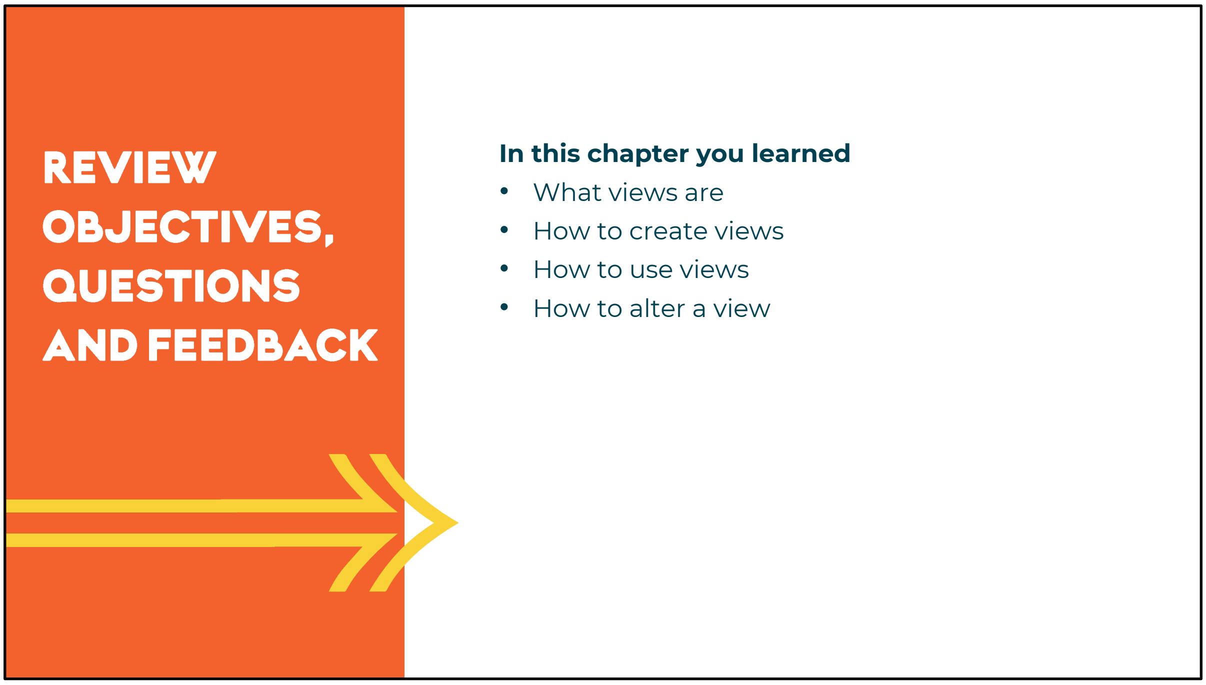
Using a view to update the underlying data

```
It is possible to use a view as a table.

You can update the table which the view refers to.

Please see the annex of this chapter for information.
```

A view can be created with a WHERE clause that restricts the records that can be accessed by the view. In this case the view is again a subset of columns from the salesperson table but restricted to just those for the specified dept_no.
The SELECT using the view should just display the specified columns and only records for salesperson’s in dept_no equal to 3.



REVIEW OBJECTIVES, QUESTIONS AND FEEDBACK

In this chapter you learned

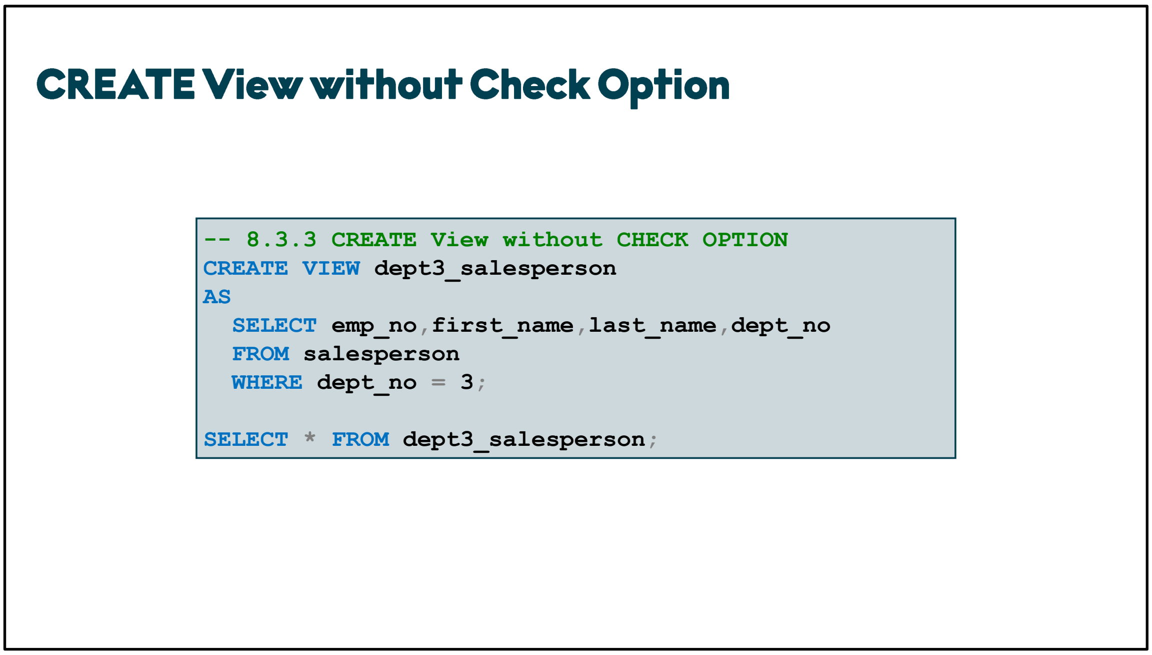
- What views are
- How to create views
- How to use views
- How to alter a view

Have we achieved the Learning Outcomes for this module? As a learner can you:

- 8. Write SQL to maintain database objects.

Also the Assessment Criteria? As a learner can you:

- 8.1 Use CREATE, ALTER and DROP to maintain table including column, Primary and Foreign Key constraints.
- 8.2 Use CREATE and DROP to maintain single column, multiple columns and unique indexes.
- 8.3 Use CREATE, ALTER and DROP to maintain views with and without the check option.
- 8.4 Use SELECT, INSERT, UPDATE and DELETE to maintain records using a view.
- 8.5 Use CREATE and DROP to maintain users and GRANT and REVOKE to maintain security options.



CREATE View without Check Option

```sql
-- 8.3.3 CREATE View without CHECK OPTION
CREATE VIEW dept3_salesperson
AS
  SELECT emp_no,first_name,last_name,dept_no
  FROM salesperson
  WHERE dept_no = 3;

SELECT * FROM dept3_salesperson;
```

A view can be created with a WHERE clause that restricts the records that can be accessed by the view. In this case the view is again a subset of columns from the salesperson table but restricted to just those for the specified dept_no.
The SELECT using the view should just display the specified columns and only records for salesperson’s in dept_no equal to 3.

INSERT Records into View

```
-- 8.3.4 INSERT Records into View
INSERT INTO dept3_salesperson
   (emp_no,first_name,last_name,dept_no)
    VALUES (70,'Leslie','Latchford',3);

INSERT INTO dept3_salesperson
   (emp_no,first_name,last_name,dept_no)
    VALUES (80,'Marcus','Matthew',4);

SELECT * FROM salesperson;
SELECT * FROM dept3_salesperson;
```

The view can be used to maintain the data on table.
Even though the view has been defined to only include records for a specified dept_no it allows both records, including the second for another dept_no, to be added to it.
This might be what is required but to many it makes no sense given the definition of the view. As we will see the check option can be used to enforce the definition of the view when data is maintained using it.

CREATE View with Check Option

```
-- 8.3.5 CREATE View with CHECK OPTION
DROP VIEW dept3_salesperson;

CREATE VIEW dept3_salesperson
AS
   SELECT emp_no,first_name,last_name,dept_no
   FROM salesperson
   WHERE dept_no = 3
   WITH CHECK OPTION;

SELECT * FROM dept3_salesperson;
```

DROP the previous view and re-CREATE it, this time with the check option at the end of the definition.

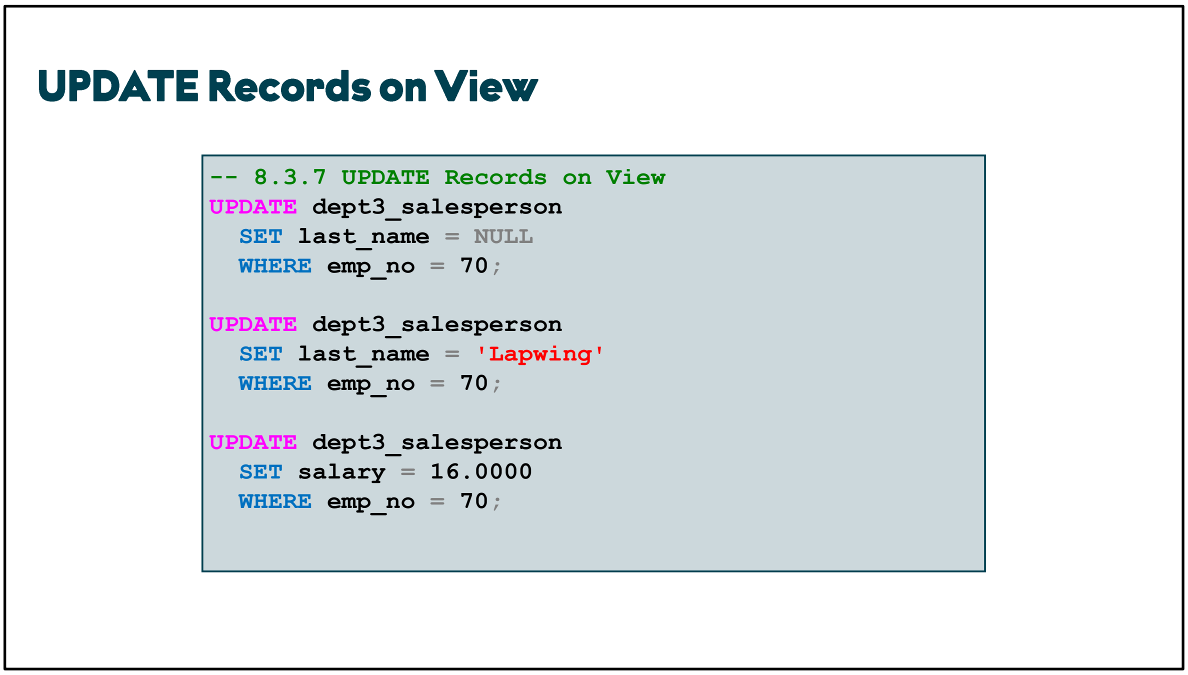
INSERT Records into View

```
-- 8.3.6 INSERT Records into View
INSERT INTO dept3_salesperson
   (emp_no,first_name,last_name,dept_no)
    VALUES (90,'Norman','Nutfield',3);

INSERT INTO dept3_salesperson
   (emp_no,first_name,dept_no)
    VALUES (100,'Peter',4);

SELECT * FROM salesperson;
SELECT * FROM dept3_salesperson;
```

This time when the records are added, the first will be stopped as it is not for the dept_no specified for the view and enforces by the check constraint.



UPDATE Records on View

```sql
-- 8.3.7 UPDATE Records on View
UPDATE dept3_salesperson
  SET last_name = NULL
  WHERE emp_no = 70;

UPDATE dept3_salesperson
  SET last_name = 'Lapwing'
  WHERE emp_no = 70;

UPDATE dept3_salesperson
  SET salary = 16.0000
  WHERE emp_no = 70;
```

The view can be used to amend or UPDATE records.

DELETE Records from View

```
-- 8.3.8 DELETE Records from View
DELETE FROM dept3_salesperson
WHERE emp_no = 70;
```

Likewise the view can be used to remove or DELETE records.

POWERING POTENTIAL

Thank you