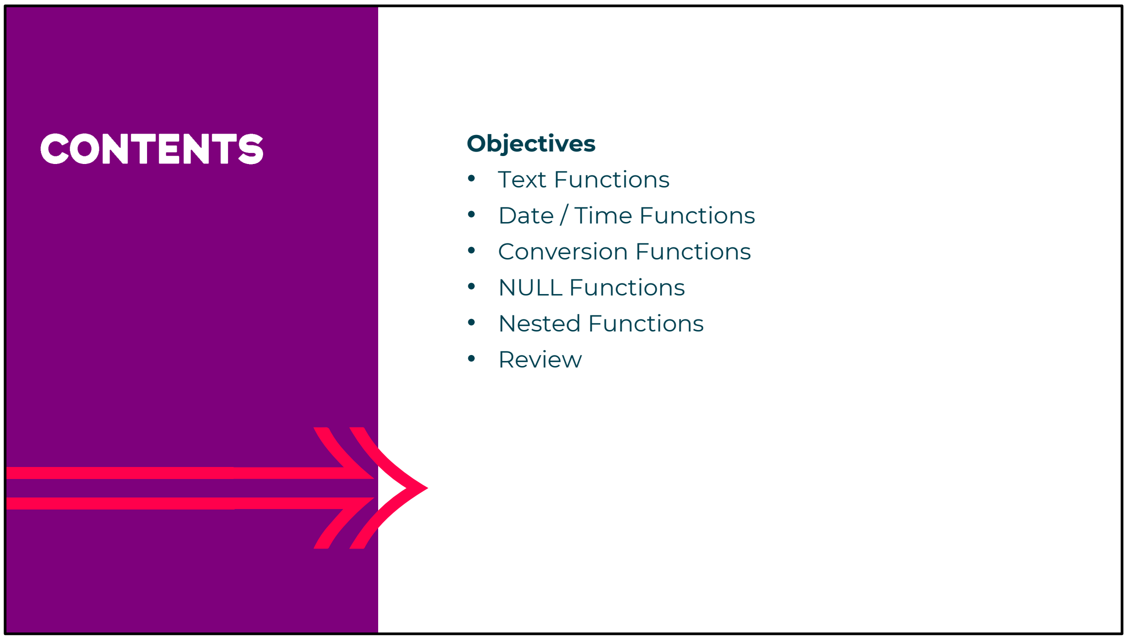
SQL Server

Functions

This module will enable the learner to write SQL that makes use of common functions.
Specifically the learner will be able to:

- Create queries that use text, date, time, conversion and rounding functions.
- Create queries that use functions to handle NULL values.
- Create a query that uses nested functions.

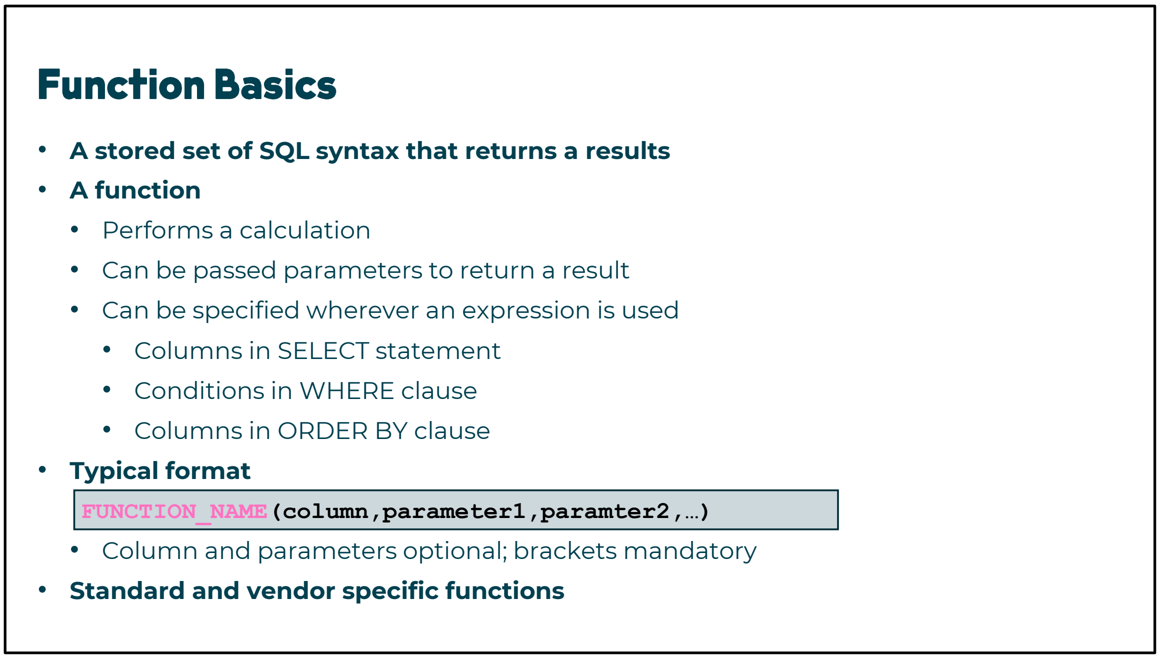
Code examples will be used to explain and demonstrate the content and for leaners to imitate. Supporting exercises will provide practice opportunities. At the end of the module there is a short quiz to review the topic.



CONTENTS

This module provides an introduction to functions to calculate and manipulate data. The topics covered are:

- Objectives – What are the Learning Outcomes and Assessment Criteria for the module.
- Text Functions – To manipulate and search text data.
- Date / Time Functions – To manipulate date and time data and get the current date and time.
- Conversion Functions – To change data from one data type to another.
- NULL Functions – Handling non specified data values.
- Nested Functions – Passing the results of one function to another.
- Review – A short recap of the module including a quiz.



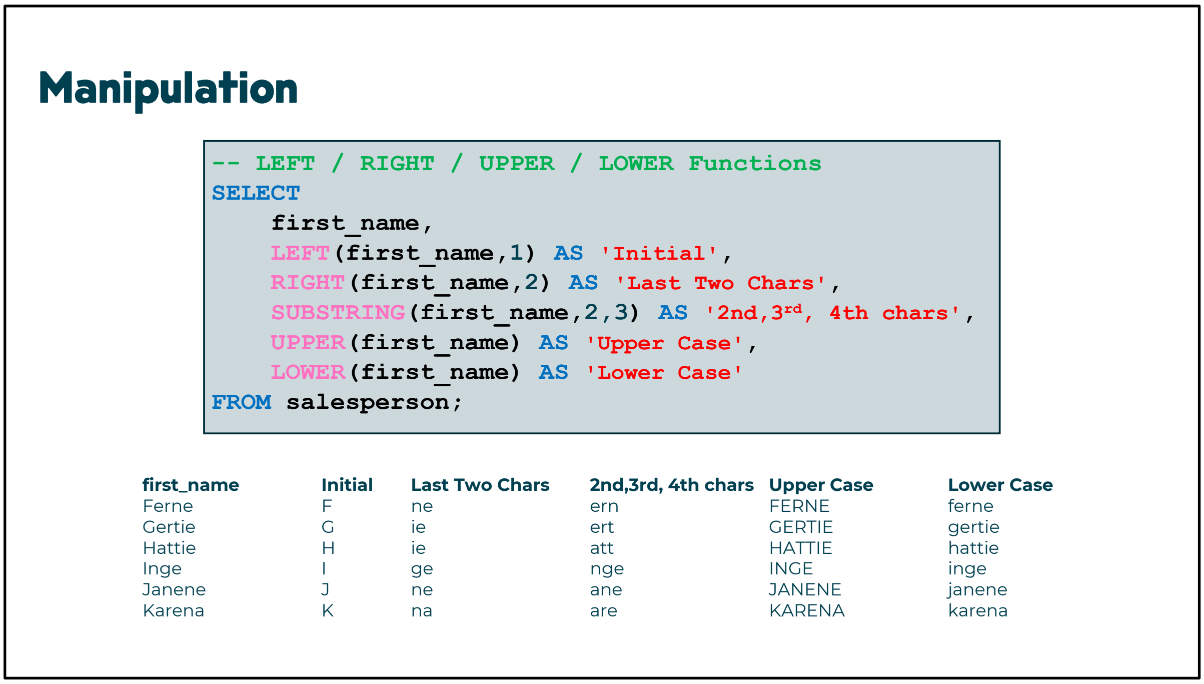
Function Basics

- **A stored set of SQL syntax that returns a results**
- **A function**
  - Performs a calculation
  - Can be passed parameters to return a result
  - Can be specified wherever an expression is used
    - Columns in SELECT statement
    - Conditions in WHERE clause
    - Columns in ORDER BY clause
- **Typical format**

```
FUNCTION_NAME(column,parameter1,paramter2,…)
```

  - Column and parameters optional; brackets mandatory
- **Standard and vendor specific functions**

Functions are stored sets of SQL statements that return a result.
Most functions are referred to as “scalar” functions and return a single value but some functions, known as “table-valued functions” return tables of data and work like parameterised views. This module focuses on scalar functions. Table-Value Functions are beyond the scope of this course.
Functions usually require parameters to be passed to them, which normally include the name of the column that we want to perform the calculation against and possibly some other information.
We can use them anywhere that SQL is expecting an expression, that is to say in SELECT lists, WHERE clauses, ORDER BY clauses, etc.
Unlike Stored Procedures, we don’t need to tell SQL to EXECUTE a function but SQL does need to be able to work out that we are calling one, so the function’s name is always followed by an open parenthesis “(“, then the parameters we’re passing, then a close parenthesis “)”, even if we don’t have any parameters to pass to it. The brackets are not optional.
The following few pages list and discuss some of the more commonly-used functions. There are many, many more available in SQL. Some are standard to all SQL products; others are specific to particular vendors.



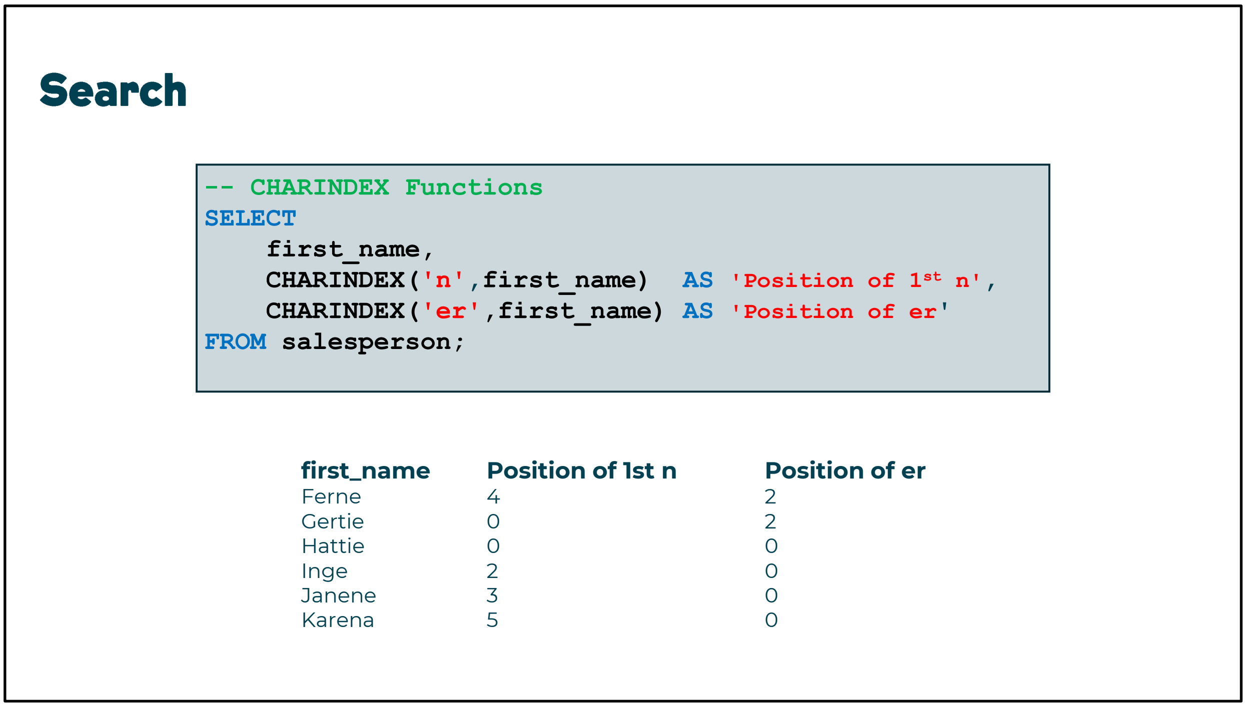
Manipulation

```sql
-- LEFT / RIGHT / UPPER / LOWER Functions
SELECT
    first_name,
    LEFT(first_name,1) AS 'Initial',
    RIGHT(first_name,2) AS 'Last Two Chars',
    SUBSTRING(first_name,2,3) AS '2nd,3rd, 4th chars',
    UPPER(first_name) AS 'Upper Case',
    LOWER(first_name) AS 'Lower Case'
FROM salesperson;
```

```
first_name   Initial   Last Two Chars   2nd,3rd, 4th chars   Upper Case   Lower Case
Ferne        F         ne               ern                  FERNE        ferne
Gertie       G         ie               ert                  GERTIE       gertie
Hattie       H         ie               att                  HATTIE       hattie
Inge         I         ge               nge                  INGE         inge
Janene       J         ne               ane                  JANENE       janene
Karena       K         na               are                  KARENA       karena
```

There are a number of text functions that manipulate the data in a specified column; some of the which require additional parameters:

- LEFT(column,n) will display the first n characters of the specified column.
- RIGHT(column,n) will display the last n characters of the specified column
- SUBSTRING(column,s,n) will display n characters of the specified column starting from position s.
- UPPER(column) will display the specified column as upper case.
- LOWER(column) will display the specified column as upper case.



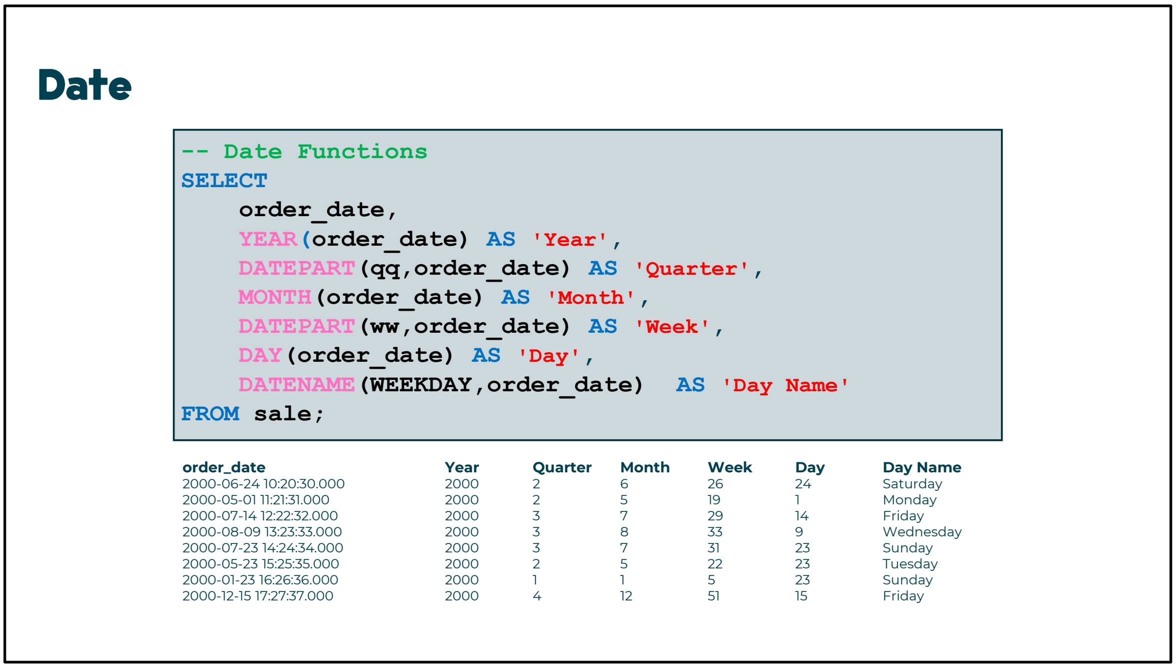
Search

```
-- CHARINDEX Functions
SELECT
    first_name,
    CHARINDEX('n',first_name)  AS 'Position of 1st n',
    CHARINDEX('er',first_name) AS 'Position of er'
FROM salesperson;
```

```
first_name    Position of 1st n     Position of er
Ferne         4                     2
Gertie        0                     2
Hattie        0                     0
Inge          2                     0
Janene        3                     0
Karena        5                     0
```

Other text functions can be used to search for a given string within the data for a specified column:

- CHARINDEX(column,'string') will display the position of the first character of the string if the string is found in the specified column or a 0 if it is not found.



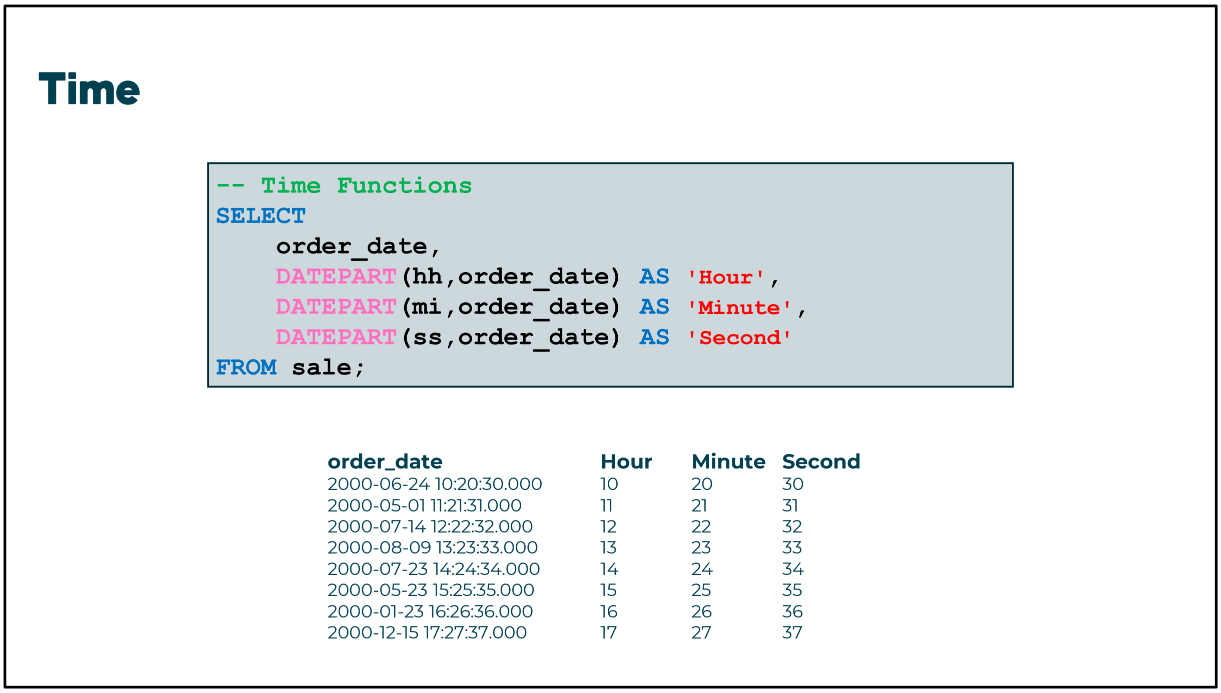
Date

```sql
-- Date Functions
SELECT
    order_date,
    YEAR(order_date) AS 'Year',
    DATEPART(qq,order_date) AS 'Quarter',
    MONTH(order_date) AS 'Month',
    DATEPART(ww,order_date) AS 'Week',
    DAY(order_date) AS 'Day',
    DATENAME(WEEKDAY,order_date)  AS 'Day Name'
FROM sale;
```

```
order_date                Year   Quarter  Month  Week  Day  Day Name
2000-06-24 10:20:30.000   2000   2        6      26    24   Saturday
2000-05-01 11:21:31.000   2000   2        5      19    1    Monday
2000-07-14 12:22:32.000   2000   3        7      29    14   Friday
2000-08-09 13:23:33.000   2000   3        8      33    9    Wednesday
2000-07-23 14:24:34.000   2000   3        7      31    23   Sunday
2000-05-23 15:25:35.000   2000   2        5      22    23   Tuesday
2000-01-23 16:26:36.000   2000   1        1      5     23   Sunday
2000-12-15 17:27:37.000   2000   4        12     51    15   Friday
```

There are a number of date functions that manipulate the data in a specified date / time column:

- YEAR(date_column) will display the year of the date.
- DATEPART(qq,date_column) will display the quarter of the date.
- MONTH(date_column) will display the month of the date.
- DATEPART(ww,date_column) will display the week of the date.
- DAY(date_column) will display the day of the date.
- DATENAME(WEEKDAY,date_column) will display the name of the day of the date.
- DATEPART(dw,date_column) will display the number of day in the week of the date.



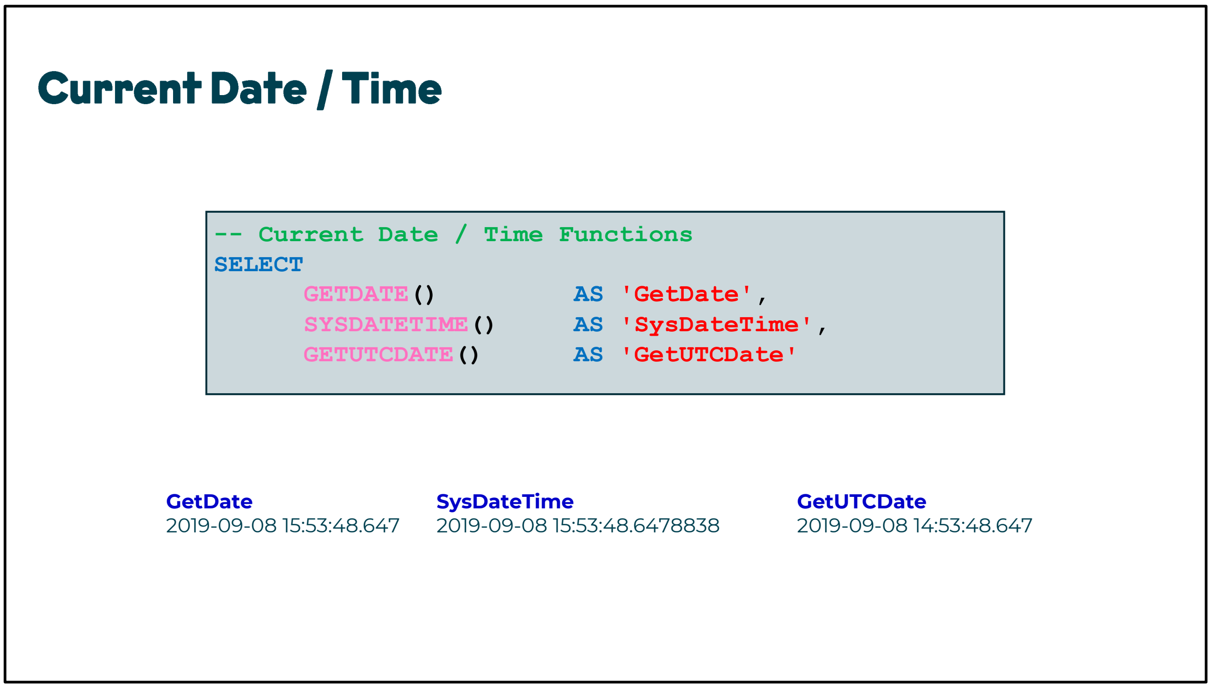
Time

```
-- Time Functions
SELECT
    order_date,
    DATEPART(hh,order_date) AS 'Hour',
    DATEPART(mi,order_date) AS 'Minute',
    DATEPART(ss,order_date) AS 'Second'
FROM sale;
```

```
order_date                Hour   Minute  Second
2000-06-24 10:20:30.000   10     20      30
2000-05-01 11:21:31.000   11     21      31
2000-07-14 12:22:32.000   12     22      32
2000-08-09 13:23:33.000   13     23      33
2000-07-23 14:24:34.000   14     24      34
2000-05-23 15:25:35.000   15     25      35
2000-01-23 16:26:36.000   16     26      36
2000-12-15 17:27:37.000   17     27      37
```

Likewise there are a number of time functions that manipulate the data in a specified date / time column:

- DATEPART(hh,time_column) will display the hour of the time.
- DATEPART(mi,time_column) will display the minute of the time.
- DATEPART(ss,time_column) will display the second of the time.



Current Date / Time

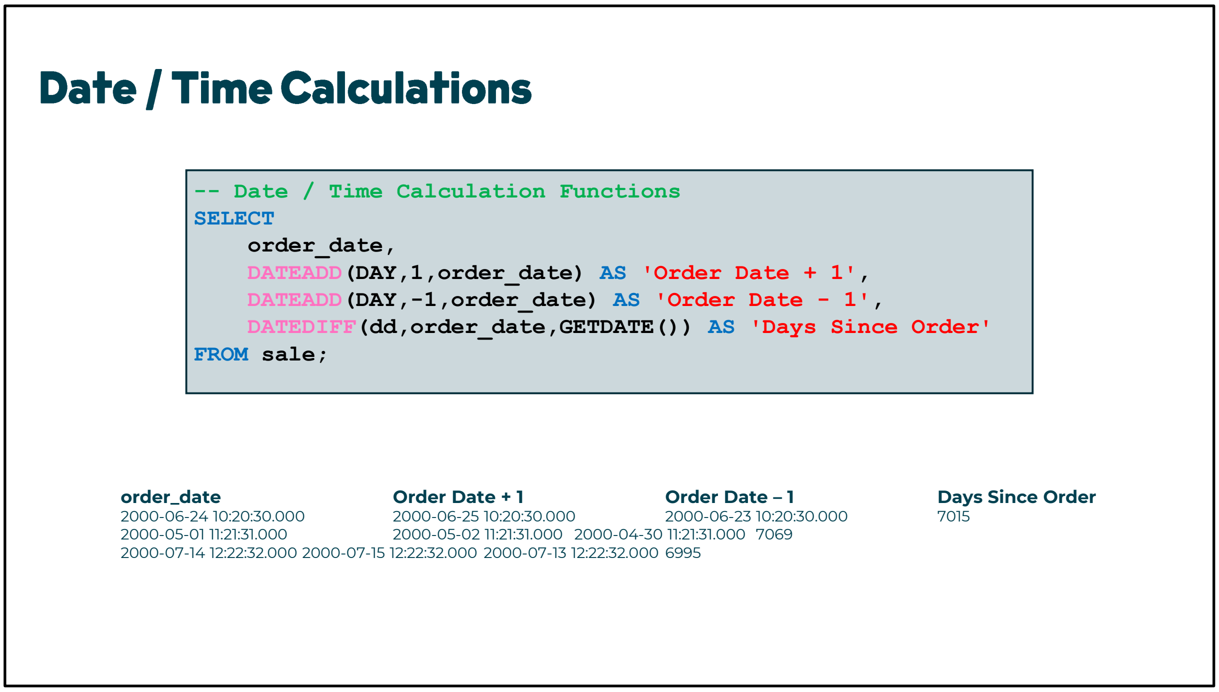
```sql
-- Current Date / Time Functions
SELECT
      GETDATE()          AS 'GetDate',
      SYSDATETIME()      AS 'SysDateTime',
      GETUTCDATE()       AS 'GetUTCDate'
```

```
GetDate                   SysDateTime                      GetUTCDate
2019-09-08 15:53:48.647   2019-09-08 15:53:48.6478838      2019-09-08 14:53:48.647
```

There are number of date / time functions that will provide the current date and time:

- GETDATE() will display the current date and time.
- SYSDATETIME() will display the time at which the function executes.
- GETUTCDATE() will display the current UTC date and time.

Note that these are examples of functions that have no column or parameters specified but still need the opening and closing brackets.



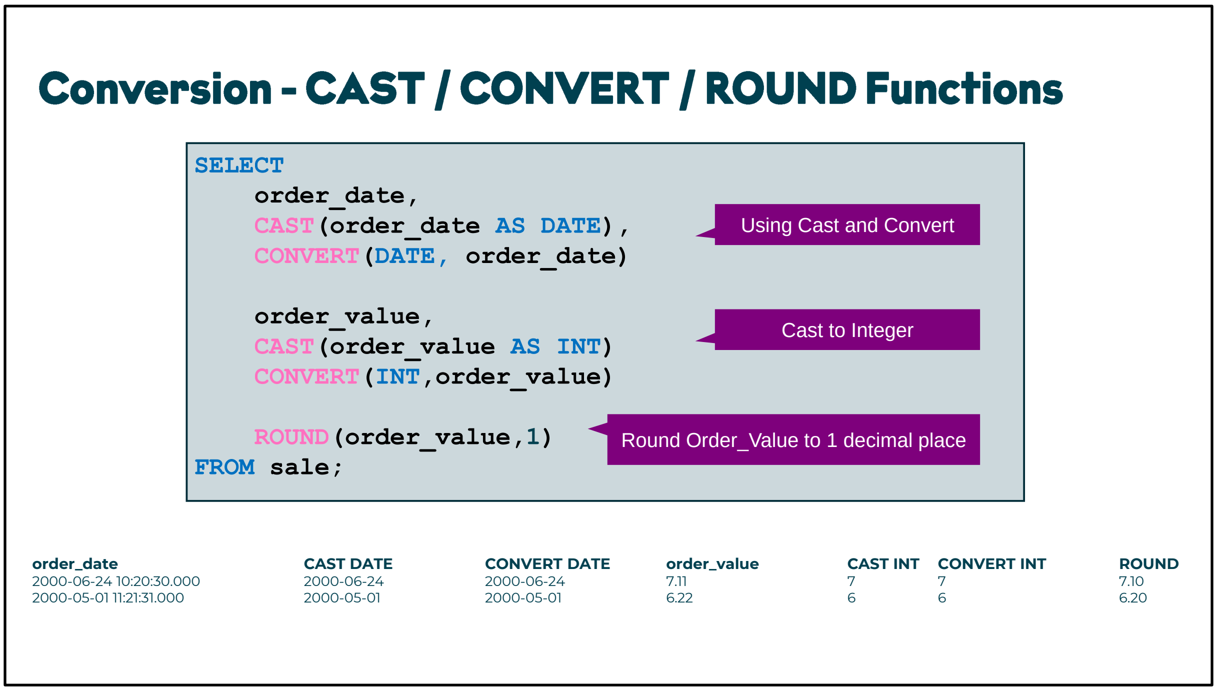
Date / Time Calculations

```sql
-- Date / Time Calculation Functions
SELECT
    order_date,
    DATEADD(DAY,1,order_date) AS 'Order Date + 1',
    DATEADD(DAY,-1,order_date) AS 'Order Date - 1',
    DATEDIFF(dd,order_date,GETDATE()) AS 'Days Since Order'
FROM sale;
```

```
order_date                  Order Date + 1              Order Date – 1              Days Since Order
2000-06-24 10:20:30.000     2000-06-25 10:20:30.000     2000-06-23 10:20:30.000     7015
2000-05-01 11:21:31.000     2000-05-02 11:21:31.000  2000-04-30 11:21:31.000  7069
2000-07-14 12:22:32.000 2000-07-15 12:22:32.000 2000-07-13 12:22:32.000 6995
```

There are also date / time functions that do various date / time calculations:

- DATEADD(UNIT,n,date_column) will display the specified date plus n of the specified time unit.
    - UNIT can be SECOND, MINUTE, HOUR, DAY, etc.
    - For a full list of units see: https://docs.microsoft.com/en-us/sql/t-sql/functions/dateadd-transact-sql
- DATEDIFF(UNIT,date_column1,date_column2) will display the difference between the two specified dates.



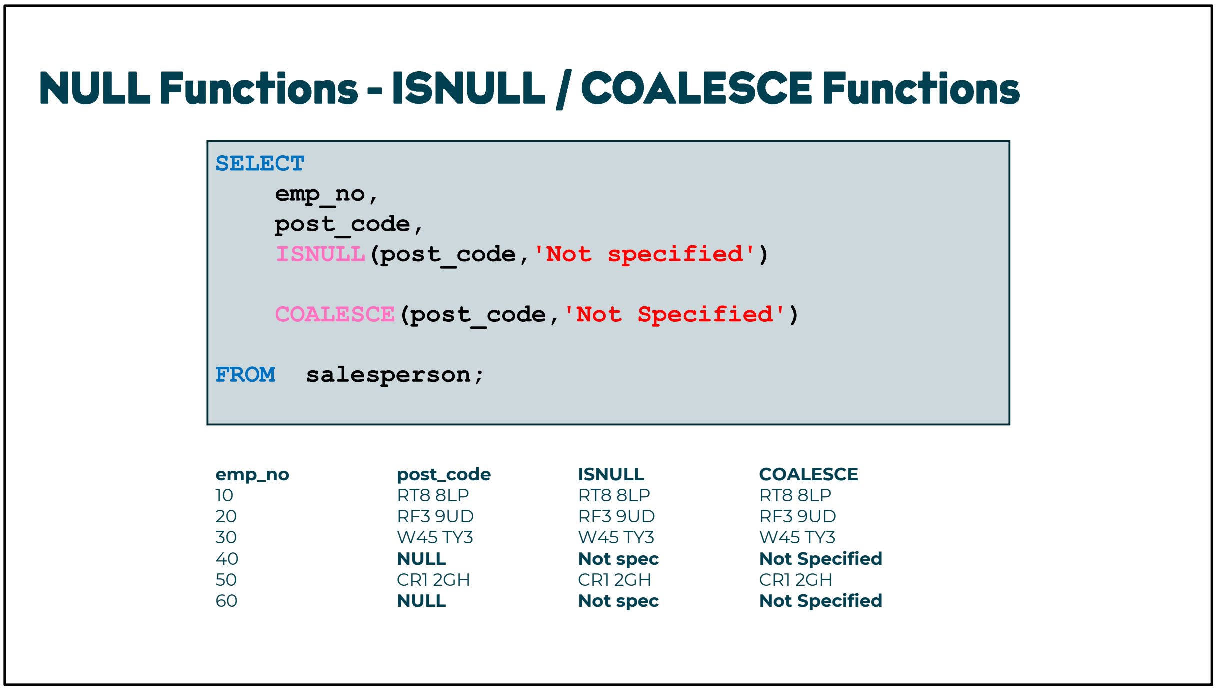
Conversion - CAST / CONVERT / ROUND Functions

```
SELECT
    order_date,
    CAST(order_date AS DATE),
    CONVERT(DATE, order_date)

    order_value,
    CAST(order_value AS INT)
    CONVERT(INT,order_value)

    ROUND(order_value,1)
FROM sale;
```

```
order_date                CAST DATE    CONVERT DATE   order_value   CAST INT   CONVERT INT   ROUND
2000-06-24 10:20:30.000   2000-06-24   2000-06-24     7.11          7          7             7.10
2000-05-01 11:21:31.000   2000-05-01   2000-05-01     6.22          6          6             6.20
```

Conversion functions change the data in a column from one type to another. This is often done to enable the changed data to be used in other functions that only work with data of that type:

- CAST(column AS TYPE) will convert the data in the column specified to the type specified.
  - TYPE can be INT, FLOAT, DATE, TIME, etc.
  - For a full list of types see: http://www.tutorialspoint.com/SQL Server/SQL Server-data-types.htm
  - Note SIGNED converts a value to a SIGNED type, which is a signed 64-bit integer.
- CONVERT(column, TYPE) will convert the data in the column specified to the type specified.
- ROUND(column,n) will round the value in the column to n decimal places.
  - As per normal rules of mathematics a 5 and above will round up and 0 to 5 stay the same.



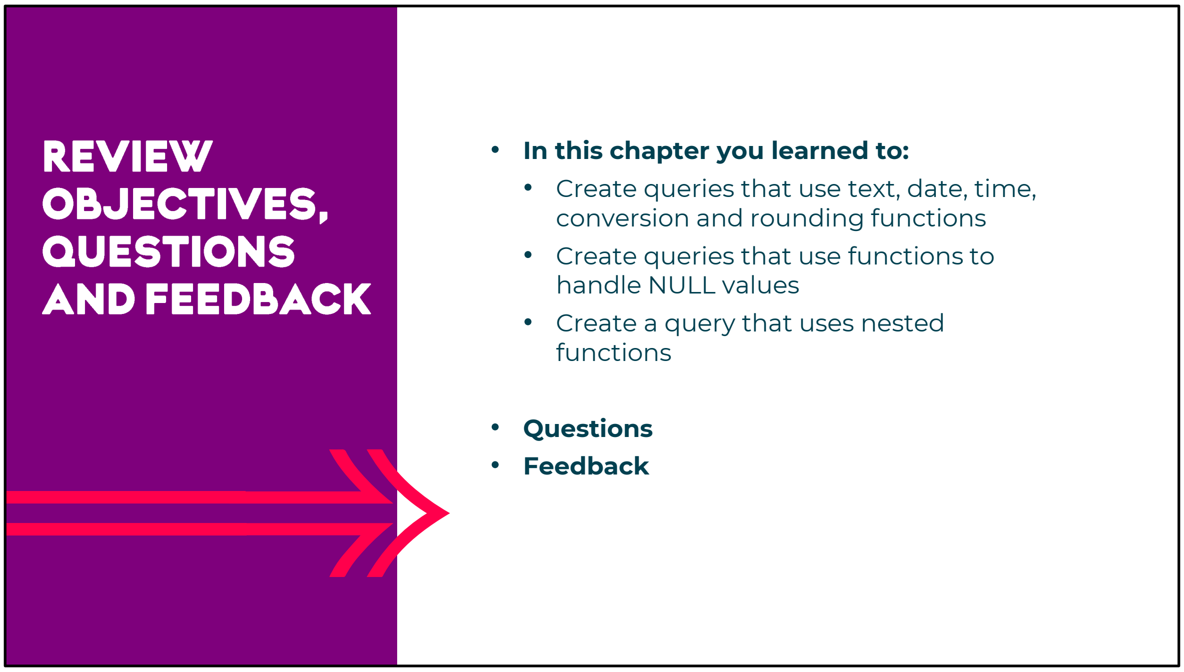
NULL Functions - ISNULL / COALESCE Functions

```sql
SELECT
    emp_no,
    post_code,
    ISNULL(post_code,'Not specified')

    COALESCE(post_code,'Not Specified')

FROM  salesperson;
```

```
emp_no      post_code   ISNULL      COALESCE
10          RT8 8LP     RT8 8LP     RT8 8LP
20          RF3 9UD     RF3 9UD     RF3 9UD
30          W45 TY3     W45 TY3     W45 TY3
40          NULL        Not spec    Not Specified
50          CR1 2GH     CR1 2GH     CR1 2GH
60          NULL        Not spec    Not Specified
```

NULL functions can be used to manipulate NULL values:

- ISNULL(column) will test whether a value is NULL displaying the value if it is populated or the specified ‘string’ if NULL.
- COALESCE(column1,column2,….’String’) will display the value in the first column if it is non-NULL; if it is NULL It will check the second column if specified and display that value if it is non-NULL; This will continue for the number of columns specified. If all are NULL a final ‘string’ is displayed.



REVIEW OBJECTIVES, QUESTIONS AND FEEDBACK

- **In this chapter you learned to:**
  - Create queries that use text, date, time, conversion and rounding functions
  - Create queries that use functions to handle NULL values
  - Create a query that uses nested functions
- **Questions**
- **Feedback**

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

- 3. Write SQL that makes use of common functions.

Also the Assessment Criteria? As a learner can you:

- 3.1 Create queries that use text, date, time, conversion and rounding functions.
- 3.2 Create queries that use functions to handle NULL values.
- 3.3 Create a query that uses nested functions.

Do you have any further questions on the content covered or feedback on the module?

POWERING POTENTIAL

Thank you