# Exercise 2- SQL - SELECT Statement

## Objectives

The primary objective of this lab is to gain familiarisation with the SQL SELECT statement.

## Reference material

This practical is based on material in the chapter.

## Overview

In this exercise you will use the QAStore database created in the previous lab and write SQL SELECT statements to query the contents of its tables.

## Estimated duration

The estimated duration for this lab is 45 minutes.

## Completed solutions

Solutions to this lab can be found in *CoursewareFolder*\2 \Solutions.

## Step by step

Please follow the instructions below, reading CAREFULLY at all times as the questions have been thoughtfully worded.

1. Start SQL Server Management studio from the Windows Start button.
2. Enter .\SQLEXPRESS as the Server name in the Connect to Server dialog box.
3. Choose QAStore from the drop down list of available databases located on the standard toolbar to ensure that it is selected as the current database.
4. Click the New Query button on the standard toolbar.
5. Enter the following code into the query window:

SELECT \* FROM salesperson

1. Execute the query and note that the entire content of the table appears in the results pane.
2. We need to investigate how to restrict the set of chosen columns. Choose any ONE of the 5 tables that exist in the database and focus your eyes on the column names of that table. Now SELECT any THREE columns of your choice from that table and write a SELECT statement that returns just the contents of these columns.
3. Your code should resemble this structure:

SELECT columnA, columnB, columnC

FROM tableZ

Where you decide what tableZ, columnA, columnB, columnC are!!

An Example, but you should choose something different

SELECT emp\_no, fname, lname

FROM salesperson

emp\_no fname lname

----------- --------------- ---------------

10 Alan Brick

20 Billy Custard

30 Chris Digger

40 Dick Ernst

50 Ernest Flipper

60 Fred Goalie

### Query 1

Write a query that selects all the salespeople who meet the following conditions:

* We would like you to display (SELECT) 2 columns only.
* These are the SURNAMES and the QUARTERLY target of each salesperson.
* the sales\_targets of the salespeople are YEARLY targets.
* One of the 2 columns you have just SELECTed is a Calculated Column. This column should have a heading (ALIAS) of 'Quarterly Target'.
* We would also like the rows to be sorted by these quarterly targets, but with the largest value appearing first.

If you do it right then the results should look like the following:

lname Quarterly Target

--------------- ------------------

Custard 3.500000

Goalie 3.250000

Flipper 3.000000

Ernst 2.750000

Brick 2.250000

Digger 1.750000

### Query 2

Write a query that displays 3 columns, each salesperson's employee number, dept number and initials.

Note: The initials are not in the table but can be retrieved from the firstname and lastname. They should be displayed as one value separated by a fullstop, eg. Alan Brick should be ‘A.B’. The answer set should have a column heading saying 'Initials'.

**Reminder:** the function 'SUBSTRING' has EXACTLY 3 **mandatory** (comma-separated) parameters.

If you do it right then the results should look like the following:

emp\_no dept\_no Initials

----------- ----------- --------

10 1 A.B

20 2 B.C

30 2 C.D

40 3 D.E

50 3 E.F

60 3 F.G

### Query 3

Write a query that lists the employee numbers of those salespeople who have managed to sell something. We don’t want any duplicates!!

If you do it right then the results should look like the following:

emp\_no

---------

10

50

60

## Using a 'WHERE' clause to restrict rows

NOTE: If you are not instructed which columns to 'SELECT' then you choose, or put '\*'. This part of the lab is testing your ability to write a 'WHERE' clause.

### Query 4

Write a query that displays only those sales whose order value is in the range 10 to 30 inclusive. Note which employee(s) made these 2 sales.

If you do it right then the results should look (a bit) like the following:

order emp company contact order order description

no no no code value date

----------- ----------- ----------- ------------ ----------- ------------

300 60 2000 OO 12 2006-07-14 ScanPRO 4800 Scanner

600 50 3000 PP 27 2006-05-23 Complete Desktop

Publishing System

### Query 5

Amend your answer to part 1 so that the WHERE clause further restricts these sales to just those made by either employee 10 or employee 50.

If you do it right then the results should look (a bit) like the following:

order emp company contact order order description

no no no code value date

----------- ----------- ----------- ------------ ----------- ------------

600 50 3000 PP 27 2006-05-23 Complete Desktop

Publishing System

Did you just use the word 'OR' in the last bit of code? If you did, and used no parentheses i.e. '('and')', then you asked the wrong question although you might appear to have the correct answer!! Hopefully you used the 'IN' word which avoided using 'OR'. If you didn’t, ensure you now do.

This SQL is INCORRECT, as it would show all sales by emp\_no 50 regardless of value:

SELECT \*

FROM sale

WHERE order\_value BETWEEN 10 AND 30

AND emp\_no = 10 OR emp\_no = 50

### Query 6

Write a query that firstly produces a list of departments whose name contains 'SYS'. Get this working first!! (depts 2 & 4 should appear).

Now amend the query to ALSO include any depts whose sales\_target is less than 10. Get that working now (depts 1, 2 & 4 should appear).

Amend the query further to reflect the requirement that we DO NOT want to see under ANY circumstances a row in the answer set that has 'Barbara Banana' as manager (depts 1 & 4 ONLY should now appear).

If you have the wrong answer, OR if you have used no parentheses, then revisit your code!!

If you do it right then the results should look like the following:

dept\_no dept\_name manager sales\_target

----------- -------------------- --------------------

1 Animal Products Adam Apricot 5.00

4 Desktop Systems Diver Dan 5.00

## When you have time

### Query 7

Write a query that lists the dept\_no/county combinations of salespeople. Again, no duplicate entries please.

If you do it right then the results should look like the following:

dept\_no county

----------- ---------------

1 Surrey

2 Hampshire

3 London

3 Surrey

### Query 8

Write a query that lists just the month numbers of the sales that have been made in ascending sequence. **Hint**: Use the DatePart function

If you do it right then the results should look like the following:

month

-----------

1

5

5

6

7

7

8

12

### Query 9

Write a query that displays contact names and tel numbers in Inner London. Inner London is '0207'. Note: tel numbers will never be stored as numeric columns because you don't ever want to do arithmetic on them! The correct answer is 3 rows, not 2 and not 4.

Note: You need to work around some very 'iffy'/'dodgy'/'bad' data.

If you do it right then the results should look like the following:

name tel

-------------------- -------------------------

Munching Mike (0207)223-9887

Ollie Octopus 0207-341-566670 ext 10

Sammy Snake 0207-988-0777

If you are not happy with your answer, then don't worry, we will REVIEW after practical!

### Query 10

Write a query that lists sales made in May and July (any year). Use the MS SQL Server date function called 'DATEPART'.

If you do it right then the results should look like the following:

order\_no order\_date

----------------------------------------------

200 2006-05-01 00:00:00.000

300 2006-07-14 00:00:00.000

500 2006-07-23 00:00:00.000

600 2006-05-23 00:00:00.000

## Dealing with nulls

### Query 11

Produce a list of people who have a real (non null) post code value.

If you do it right then the results should look like the following:

emp\_no post\_code

----------- ----------

10 RT8 8LP

20 RF3 9UD

30 W45 TY3

50 CR1 2GH

### Query 12

Now we want a similar result set, but **all** the salespeople must appear. Display emp number and post code of the sales people, but if the post code is 'unknown' then the string 'Post Code Unknown' should be displayed instead. Make use of the function 'COALESCE', ensuring the 2nd column (calculated) has a col heading. This answer set should display only 2 COLUMNS (but all 6 rows), as that is all the question asks for!

If you do it right then the results should look like the following:

emp\_no Postcode

----------- -----------------

10 RT8 8LP

20 RF3 9UD

30 W45 TY3

40 Post Code Unknown

50 CR1 2GH

60 Post Code Unknown

### Query 13

Run only the SELECT/FROM of this precoded query and then estimate how many rows will be returned if you then run it with the WHERE clause added. Now uncomment the 'WHERE' and run all 3 lines. Did you estimate correctly?

SELECT emp\_no, post\_code

FROM salesperson

--WHERE post\_code <> 'RT8 8LP'

Can you see how you could get 5 rows to appear?

SELECT emp\_no, post\_code

FROM salesperson

WHERE post\_code <> 'RT8 8LP' OR ????????????????

If you do it right then the results should look like the following:

emp\_no post\_code

----------- ----------

20 RF3 9UD

30 W45 TY3

40 NULL

50 CR1 2GH

60 NULL

## Limiting the number of rows using TOP

### Query 14

Produce a list of the emp\_no, order\_no and order\_value for all the sales made in descending order of order\_value.

If you do it right then the results should look like the following:

emp\_no order\_no order\_value  
----------- ----------- -----------  
50 600 27  
60 300 12  
60 100 7  
60 200 6  
10 400 5  
10 700 3  
60 800 3  
60 500 2

Now limit the query to return the first row only. Make use of the TOP operator.

### Query 15

Use the same initial query as above produce a list for the sales in ascending order of order\_value.

If you do it right then the results should look like the following:

emp\_no order\_no order\_value  
----------- ----------- -----------  
60 500 2

10 700 3

60 800 3

10 400 5

60 200 6

60 100 7

60 300 12

50 600 27

Now we want to return the lowest two orders. Modify the query to do this.

If your output does not contain three rows as below – ask yourself what you would expect.

emp\_no order\_no order\_value  
----------- ----------- -----------  
60 500 2

10 700 3

60 800 3

What are the values of the lowest two orders? The answer is orders with the value 2 and 3. What orders have these values – orders 500, 700, 800. How would I achieve that using SQL.

## Combining results using UNION

### Query 16

Produce a list of order\_no, emp\_no, order\_value and calculate a new order\_value that will be the order\_value as listed multiplied by two (and give that column a suitable name) for all products sold to be employee 10.

You should produce a list that looks like.

order\_no emp\_no order\_value NewOrderValue

----------- ----------- ----------- -------------

400 10 5 10

700 10 3 6

Now write a similar query that for employee 60 returns the same columns, but the new order value will now be the order\_value as listed multiplied by three.

You should produce a list that looks like.

order\_no emp\_no order\_value NewOrderValue

----------- ----------- ----------- -------------100 60 7 21

200 60 6 18

300 60 12 36

500 60 2 6

800 60 3 9

Now write a query that returns all the remaining orders (i.e. not for employee 10 or 60) that keeps returns the new order value as the same value of the order value as listed.

You should produce a list that looks like.

order\_no emp\_no order\_value NewOrderValue

----------- ----------- ----------- -------------600 50 27 27

Now combine these three queries into one. Hint – use the UNION statement.

You should produce a list that looks like.

order\_no emp\_no order\_value NewOrderValue

----------- ----------- ----------- -------------

100 60 7 21

200 60 6 18

300 60 12 36

400 10 5 10

500 60 2 6

600 50 27 27

700 10 3 6

800 60 3 9

**Query 17**

Take your query above an order it by emp\_no (in ascending order) and within each emp\_no by order\_value (also in ascending order).

Your output should look like.

order\_no emp\_no order\_value NewOrderValue

----------- ----------- ----------- -------------

700 10 3 6

400 10 5 10

600 50 27 27

500 60 2 6

800 60 3 9

200 60 6 18

100 60 7 21

300 60 12 36

## Computing each row

### Query 18

We want to produce the same output as in query 16 or 17 but without using a UNION statement. Hint: use a CASE statement.