# Chapter 1- SQL - Joins

## Objectives

The primary objective of this lab is to gain familiarisation with SQL Joins

## Reference material

This practical is based on material in the *SQL - Joins* chapter.

## Overview

In this exercise you will use the QAStore database created earlier and write SQL SELECT statements that employ JOIN clauses 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*\1 SQL - Joins\Solutions.

## Practical 1 – INNER JOINS

## 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.

### Query 1

SELECT \* FROM dept

SELECT \* FROM salesperson

SELECT \* FROM sale

SELECT \* FROM company

SELECT \* FROM contact

Write a query that displays the order number, order value and the NAME of the company each sale was made to.

* If you are not sure what to JOIN on then run the 'sale' and 'company' SELECT statements (listed above) and choose a sale at random and decide the name of the company it was made to and write the code to mimic what you just did.
* Use TABLE ALIASES, please, a good habit to get into early on.
* Sort the answer set by company name. (8 rows)
* Notice all 4 companies have been sold to (this may be relevant later).

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

order\_no order\_value name

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

100 7 Happy Heaters PLC

400 5 Happy Heaters PLC

300 12 Icicle Igloos PLC

700 3 Icicle Igloos PLC

800 3 Judo Jeans PLC

600 27 Judo Jeans PLC

200 6 Judo Jeans PLC

500 2 Kipper Kickers Inc

### Query 2

Write a totally new query that starts off by showing all sales displaying from each sale the order\_value and description columns.

Now, additionally, on the LEFT HAND SIDE of the display show the full name and dept number of the person who made the sale. Use TABLE ALIASES please.

Notice the 8 sales have been made by 2 depts (3 people) only and you get an 8 row result set (this may be relevant later).

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

fname lname dept\_no order\_value description

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

Fred Goalie 3 7 IBM Thinkpad 755CE

Fred Goalie 3 6 MS Office Professional \* 30

Fred Goalie 3 12 S25 canPRO 4800 Scanner

Alan Brick 1 5 Modems and Cables etc

Fred Goalie 3 2 Laser printer

Ernest Flipper 3 27 Complete Desktop Publishing System

Alan Brick 1 3 SQL Server 7.0 20 user licence

Fred Goalie 3 3 Printer cartridges

### Query 3

### Now Copy/Paste your code from Query2 and ADD the emp\_no of the person who made the sale as a FIRST column and also sort the sales by this emp\_no.

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

emp\_no fname lname dept\_no ord\_val description

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

10 Alan Brick 1 5 Modems and Cables etc

10 Alan Brick 1 3 SQL Server 7.0 20 user licence

50 Ernest Flipper 3 27 Complete Desktop Publishing System

60 Fred Goalie 3 3 Printer cartridges

60 Fred Goalie 3 2 Laser printer

60 Fred Goalie 3 7 IBM Thinkpad 755CE

60 Fred Goalie 3 6 MS Office Professional \* 30

60 Fred Goalie 3 12 25 ScanPRO 4800 Scanner

### Query 4

Copy/Paste your code from Query 3.

Make 2 changes to your code:

* Firstly, add the MANAGER of the seller of the sale as an extra FIRST column.

It is now broken, so get this working. If you do it right then the results should look like the following:

manager empno fname lname dept\_no ordval description

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

Paul Peach 60 Fred Goalie 3 7 IBM Thinkpad 755CE

Paul Peach 60 Fred Goalie 3 6 MS Office Professional...

Paul Peach 60 Fred Goalie 3 12 25 ScanPRO 4800 Scanner

Adam Apricot 10 Alan Brick 1 5 Modems and Cables etc

Paul Peach 60 Fred Goalie 3 2 Laser printer

Paul Peach 50 Ernest Flipper 3 27 Complete Desktop...

Adam Apricot 10 Alan Brick 1 3 SQL Server 7.0 20 user...

Paul Peach 60 Fred Goalie 3 3 Printer cartridges

* Secondly, restrict the answer set so that it lists only those sales that contain the text 'printer' in their description. (2 rows)

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

manager empno fname lname dept\_no ordval description

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

Paul Peach 60 Fred Goalie 3 2 Laser printer

Paul Peach 60 Fred Goalie 3 3 Printer cartridges

### Query 5

Write a totally new query that displays each contact's name and their company's name. (you should get 10 rows)

* Make sure you JOIN on the thing they share!!
* Note how many contacts (1, 2, 3, 3) are in each company as you will 'count' them in code later.

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

Contact Company

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

Munching Mike Happy Heaters PLC

Naughty Nick Icicle Igloos PLC

Ollie Octopus Icicle Igloos PLC

Purposeful Peter Judo Jeans PLC

Quentin Quail Judo Jeans PLC

Robber Red Judo Jeans PLC

Sammy Snake Kipper Kickers Inc

Terrible Tim Kipper Kickers Inc

Uppy Umbrella Kipper Kickers Inc

### For Further Investigation:

### Query 6

Managers like to telephone contacts after 'big' sales. It's a sort of 'compliance' requirement. A big sale is defined as one where the value of the order is greater than 50% of the salesperson's target, i.e. somebody hits half their yearly target in one deal.

You need to complete the following query that lists the manager, the contact name and the contact telephone number that the manager must call. You must put in the join predicates and complete the Where clause.

SELECT manager, C.name, C.tel

FROM dept D INNER JOIN salesperson SP

ON ?? = ??

INNER JOIN sale S

ON ?? = ??

INNER JOIN contact C

on ??????????????????????

WHERE ?? > .5 \* ??

If you are NOT getting a 4 row result set, there are 2 things that MIGHT help you spot the problem.

1. If you are getting 7 rows, try adding the order\_no column to the select list, re-run and look closely.
2. If you get 5 rows, recognize that any contacts who have not been sold to should not appear in result set.

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

manager name tel

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

Paul Peach Munching Mike (0207)223-9887

Paul Peach Ollie Octopus 0207-341-566670 ext 10

Adam Apricot Munching Mike (0207)223-9887

Paul Peach Purposeful Peter 0131 324545 ext 213

Important: When you get this working, can you accurately predict how many rows the result set will contain when you run it WITHOUT the WHERE clause? I.e. natural JOIN of the 4 tables.

Did you guess correctly? The answer is 8 rows, it is driven by the number of rows in 'sale' (the 'many' table).

How many rows would you expect to get if you ran your code after these 3 INSERTs happened?

INSERT INTO dept VALUES (20, 'Dept 20', 'Dept 20 Manager', 20)

INSERT INTO salesperson VALUES (100, 'Pete', 'Pitstop', 2, 5, NULL, NULL, NULL, NULL)

INSERT INTO contact VALUES (4000, 'ZZ', 'Zinedine Zidane', 'Celebrity', NULL, NULL)

Work it out first then run the 3 INSERTS followed by the SELECT statement and see if you were correct.

The answer should still be 8. Adding a dept with no people, a person with no sales, or a contact who has not been sold too will not affect the result of an INNER JOIN with sale

Now decide how many rows you would get if you now ran this statement:

INSERT INTO sale VALUES (900, 10, 1000, 'MM', 3, '05-12-2006', 'Metal 3\*2 Desk')

Work it out first then run the INSERT followed by the SELECT statement and see if you were correct.

Answer should be 9 as the sale table has 9 rows

Here are the DELETE statements (run them if you ran any of the 4 INSERTs above).

DELETE FROM salesperson WHERE emp\_no = 100

DELETE FROM dept WHERE dept\_no = 20

DELETE FROM contact WHERE name = 'Zinedine Zidane'

DELETE FROM sale WHERE order\_no = 900

## Practical 2 – OUTER JOINS

### Query 1

This first exercise is done largely as a 'tutorial' learning exercise, you practise outer joins later.

Run the following 3 precoded queries.

* NOTE - there are 5 depts with 5 managers.
* NOTE - the salespeople are in 3 different depts.
* NOTE - the 3rd query fails to list 'Diver Dan' because he manages a 'dept' with no people.

SELECT dept\_no, manager

FROM dept

Should give:

dept\_no manager

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

1 Adam Apricot

2 Barbara Banana

3 Paul Peach

4 Diver Dan

5 Xavier Xylophone

SELECT DISTINCT dept\_no AS 'Distinct list of depts that people are in'

FROM salesperson

Should give:

Distinct list of depts that people are in

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

1

2

3

SELECT D.dept\_no, manager, lname

FROM salesperson SP INNER JOIN dept D

ON SP.dept\_no = D.dept\_no

Should give:

dept\_no manager lname

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

1 Adam Apricot Brick

2 Barbara Banana Custard

2 Barbara Banana Digger

3 Paul Peach Ernst

3 Paul Peach Flipper

3 Paul Peach Goalie

### Query 2

Copy the the 3rd precoded query from above and change the word INNER to the word RIGHT.

This is called an outer JOIN, inserting the word RIGHT means literally "include every row from the table on the RIGHT of the word JOIN", (even if there is no matching row in the table on the LEFT).

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

dept\_no manager lname

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

1 Adam Apricot Brick

2 Barbara Banana Custard

2 Barbara Banana Digger

3 Paul Peach Ernst

3 Paul Peach Flipper

3 Paul Peach Goalie

4 Diver Dan NULL

5 Xavier Xylophone NULL

Now 'Coalesce' the lname column to display 'Nobody in this dept' where appropriate using the following template as a guide:

SELECT D.dept\_no, manager, COALESCE(????, ??????????????) as ??????

FROM salesperson SP RIGHT JOIN dept D

ON SP.dept\_no = D.dept\_no

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

dept\_no manager lname

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

1 Adam Apricot Brick

2 Barbara Banana Custard

2 Barbara Banana Digger

3 Paul Peach Ernst

3 Paul Peach Flipper

3 Paul Peach Goalie

4 Diver Dan Nobody in this dept

5 Xavier Xylophone Nobody in this dept

### Recognize that

FROM dept D LEFT JOIN salesperson SP

### would give the same result as

FROM salesperson SP RIGHT JOIN dept D

### Query 3

### Ask yourself whether this FROM clause makes any sense?

FROM dept D RIGHT JOIN salesperson SP

### Does this translate into:

### "Show me all the people even one's in a non existent dept"?

### Hopefully the referential integrity between the tables will ensure that if you have depts 1-4 only, that there is no one in dept 5 or 6 or 87.

### But can a salesperson be in no (NULL) dept at all? Well, it depends whether 'dept\_no' of 'salesperson' is a NULLable (optional) column or not.

### In your schema/table it IS an optional column so the following INSERT (try it) will run ok.

INSERT INTO salesperson(emp\_no, fname, lname, dept\_no)

VALUES (70, 'Monica', 'Ell', NULL)

The following INNER JOIN query will not discover her

SELECT manager, COALESCE(lname, 'Nobody in this dept') AS Surname

FROM salesperson SP INNER JOIN dept D

ON SP.dept\_no = D.dept\_no

Try it now, it should produce the following:

manager Surname

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

Adam Apricot Brick

Barbara Banana Custard

Barbara Banana Digger

Paul Peach Ernst

Paul Peach Flipper

Paul Peach Goalie

Nor will this Outer JOIN

SELECT manager, COALESCE(lname, 'Nobody in this dept') AS Surname

FROM salesperson SP RIGHT JOIN dept D

ON SP.dept\_no = D.dept\_no

Try it now, it should produce the following:

manager Surname

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

Adam Apricot Brick

Barbara Banana Custard

Barbara Banana Digger

Paul Peach Ernst

Paul Peach Flipper

Paul Peach Goalie

Diver Dan Nobody in this dept

Xavier Xylophone Nobody in this dept

But this one will, note dept D RIGHT JOIN salesperson SP

SELECT COALESCE(manager,'Has no manager') AS Manager, lname

FROM dept D RIGHT JOIN salesperson SP

ON SP.dept\_no = D.dept\_no

Try it now, it should produce the following:

Manager lname

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

Adam Apricot Brick

Barbara Banana Custard

Barbara Banana Digger

Paul Peach Ernst

Paul Peach Flipper

Paul Peach Goalie

Has no manager Ell

### Query 4

‘Can I do a LEFT & a RIGHT JOIN at the same time?’ is a question often asked and the answer is yes, it’s called a FULL JOIN.

Try the following:

SELECT COALESCE(manager, '\*\* Has no manager \*\*')

AS Manager,

COALESCE(lname, '\*\* Empty dept \*\*')

AS Surname

FROM dept D FULL JOIN salesperson SP

ON SP.dept\_no = D.dept\_no

It should produce the following:

Manager Surname

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

Adam Apricot Brick

Barbara Banana Custard

Barbara Banana Digger

Paul Peach Ernst

Paul Peach Flipper

Paul Peach Goalie

Diver Dan \*\* Empty dept \*\*

Xavier Xylophone \*\* Empty dept \*\*

\*\* Has no manager \*\* Ell

But if you were now to include 'dept\_no' in this query, should you choose SP.dept\_no or D.dept\_no, because one of the rows is surely going to have a NULL dept\_no?

Well you could include either and COALESCE it to remove the NULL, but there is one problem. Namely, the 'COALESCE' function requires that both parameters are of the same data type,

So,

* COALESCE(D.dept\_no, 0 ) would be valid syntax but produce a misleading '0' in results.
* COALESCE(D.dept\_no, 'n/a') would fail as 1st arg is numeric, but 2nd arg is not.

The solution is to convert the 'dept\_no' into a character string.

-- In SQL Server this would be:

COALESCE(STR(D.dept\_no, 2), 'n/a') meaning convert D.dept\_no into a 2 character string.

Try the following:

SELECT COALESCE(STR(D.dept\_no,2),'N/A') AS Dept\_no,

COALESCE(manager,'\*\* Has no manager \*\*') AS Manager,

COALESCE(lname,'\*\* Empty dept \*\*') AS Surname

FROM dept D FULL JOIN salesperson SP

ON SP.dept\_no = D.dept\_no

The result should be:

Dept\_no Manager Surname

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

1 Adam Apricot Brick

2 Barbara Banana Custard

2 Barbara Banana Digger

3 Paul Peach Ernst

3 Paul Peach Flipper

3 Paul Peach Goalie

4 Diver Dan \*\* Empty dept \*\*

5 Xavier Xylophone \*\* Empty dept \*\*

N/A \*\* Has no manager \*\* Ell

### Query 5

The odd thing is, what happens when you add the following 'WHERE' clause to this 'Left' JOIN.

SELECT D.dept\_no,

COALESCE(manager,'\*\* Has no manager \*\*') AS

Manager,

COALESCE(lname,'\*\* Empty dept \*\*') AS Surname

FROM dept D LEFT JOIN salesperson SP

ON SP.dept\_no = D.dept\_no

WHERE SP.dept\_no IS NULL

Try it now, it should produce the following:

dept\_no Manager Surname

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

4 Diver Dan \*\* Empty dept \*\*

5 Xavier Xylophone \*\* Empty dept \*\*

You have just seen a technique (there are others, as we will see) of how to find the depts that have no salespeople by JOINing Dept to Salesperson, including all the 'spares', and then just retaining the 'spares'.

The mistake you must not make is to write this 'WHERE' clause:

WHERE SP.post\_code IS NULL

Why?

Because then you will be listing managers of depts (and their employees) that EITHER

1. have no people in them, or
2. have a person, but a person who has no post\_code.

Try the following:

SELECT D.dept\_no,

COALESCE(manager,'\*\* Has no manager \*\*') AS Manager,

COALESCE(lname,'\*\* Empty dept \*\*') AS Surname

FROM dept D LEFT JOIN salesperson SP

ON SP.dept\_no = D.dept\_no

WHERE SP.post\_code IS NULL -- no good, must be a primary key(not NULL) column for safety

dept\_no Manager Surname

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

3 Paul Peach Ernst

3 Paul Peach Goalie

4 Diver Dan \*\* Empty dept \*\*

5 Xavier Xylophone \*\* Empty dept \*\*

What you have just learnt is ANSI 92 Outer JOIN Syntax.

LEFT, RIGHT, FULL mean LEFT OUTER, RIGHT OUTER, FULL OUTER respectively.

Your DBMS may allow you to leave out the word 'OUTER'; SQL Server does, Access doesn’t.