

S2.1: SQL 1

Create Your Oracle Account

We are going to be using a cloud version of Oracle for this module, so you will be able to use it on and off campus.

Go to <https://apex.oracle.com/en/> to create a free account. From the Home page, click the 'Start for Free Today' button, and then choose the 'Free APEX Workspace' link. You will need to create the following details, and should make them easy to remember, so I suggest using your UCLan username and email address:

Workspace name: your UCLan user name, e.g. jallen17

Username: your UCLan email address, e.g. jallen17@uclan.ac.uk

Password: I'll leave that up to you, e.g. mypassword

Creating Tables and Loading Data

We shall now create two tables and load some data into them. There are 4 text files containing the scripts for this (on Blackboard). One file creates a table called DEPT (for Department). One file creates a table called EMP (for employee). The third file loads the data into the DEPT table, and the last file loads the data into the EMP table. Carry out the following instructions in this order:

1. Click on and go to the SQL Workshop.
2. Click on and go to the SQL Commands window.
3. Copy and paste the contents of 'create dept table.txt' into the SQL Commands window and press run.
4. Copy and paste the contents of 'create emp table.txt' into the SQL Commands window and press run.
5. Copy and paste the contents of 'insert data into dept table.txt' into the SQL Commands window and press run.
6. Copy and paste the contents of 'insert data into emp table.txt' into the SQL Commands window and press run.

These commands create the table structures and load the following data. Each file has to be ran separately, and once ran correctly the script can be deleted from the command window.

EMP:

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7839	KING	PRESIDENT	-	11/17/2001	82000	-	10
7782	CLARK	MANAGER	7839	06/09/2009	2750	-	10

					0		
7934	MILLER	CLERK	7782	01/23/2006	13200	-	10
7566	JONES	MANAGER	7839	04/02/2012	26850	-	20
7788	SCOTT	ANALYST	7566	12/09/2008	19500	-	20
7902	FORD	ANALYST	7566	12/03/2014	21500	-	20
7369	SMITH	CLERK	7902	12/17/2002	13750	-	20
7876	ADAMS	CLERK	7788	01/12/2011	11900	-	20
7698	BLAKE	MANAGER	7839	05/01/2005	24000	-	30
7499	ALLEN	SALESMAN	7698	02/20/2006	19000	6300	30
7521	WARD	SALESMAN	7698	02/22/2018	18500	4500	30
7654	MARTIN	SALESMAN	7698	09/28/2008	15650	3400	30
7844	TURNER	SALESMAN	7698	09/08/2011	18500	0	30
7900	JAMES	CLERK	7698	12/03/1998	11950	-	30
4422	STEVENS	MANAGER	7839	02/14/2019	24750	-	50
3258	GREEN	SALESMAN	4422	07/24/2003	18200	2750	50
6548	BARNES	CLERK	4422	01/16/2018	11950	-	50

DEPT:

DEPTNO	DNAME	LOC
10	ACCOUNTING	LONDON
20	RESEARCH	PRESTON
30	SALES	LIVERPOOL
40	OPERATIONS	MANCHESTER
50	MARKETING	LUTON

Note that MGR is the Empno of the employee's manager. King, as the president, doesn't have a manager.

To see the table structure (aka data dictionary or metadata) you can use the DESC (describe) function:
The command `DESC emp;` would give the following result:

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
EMP	EMPNO	NUMBER	-	4	0	1	-	-	-
	ENAME	VARCHAR2	10	-	-	-	✓	-	-
	JOB	VARCHAR2	9	-	-	-	✓	-	-
	MGR	NUMBER	-	4	0	-	✓	-	-
	HIREDATE	DATE	7	-	-	-	✓	-	-
	SAL	NUMBER	-	7	2	-	✓	-	-
	COMM	NUMBER	-	7	2	-	✓	-	-
	DEPTNO	NUMBER	-	2	0	-	✓	-	-

Using the Apex interface, you can look at the tables directly, and have slightly different functionality to what this workbook tells you, but you will need to be able to query the tables using SQL as described in this worksheet.

NOTE: There are some SQL exercises at the end of this worksheet, however, read through all of the pages prior to attempting any of the questions. It is also very important to try out all the examples as you will be able to see the output which will help you gain a better understanding of how SQL works.

Ok, so now we have our tables created and some data loaded to them lets have a go at querying the data...

Selecting Data from Tables

The **SELECT** statement is the workhorse of query processing. The basic statement is:

SELECT COLUMN(S) FROM TABLENAME;

This is the minimum amount of detail which must be entered for a **SELECT** statement to work.

Try the following examples:

SELECT * FROM emp;	Provides a listing of all the data (all columns) in the EMP table.
SELECT ename FROM emp;	Gives a list of all the employee names found in the emp table.
SELECT dname, loc FROM dept;	gives department names and locations.
SELECT job FROM emp;	Lists all the jobs in the emp table even if they appear more than once.
SELECT DISTINCT job FROM emp;	List all the jobs in the EMP table eliminating duplicates.
SELECT job, deptno FROM emp;	Lists the combination of jobs and departments for every row of the emp table.
SELECT DISTINCT job, deptno FROM emp;	List all the combinations of job and department in the EMP table eliminating duplicates.

The WHERE Clause

A **WHERE** clause causes a 'search' to be made, and only those rows that meet the search condition are retrieved.

A **WHERE** clause condition can use any of the following comparison operators:

Operator	meaning	Example
=	equal to	SELECT * FROM emp WHERE ename = 'JONES'; (Note that the data is case sensitive, this would not find 'Jones')
!= <>	not equal to (all do	SELECT * FROM emp WHERE ename != 'FORD';

=	the same)	
>	greater than	SELECT * FROM emp WHERE sal > 15000;
>=	greater than or equal to	SELECT * FROM emp WHERE sal >= 12000;
<	less than	SELECT * FROM emp WHERE sal < 15000;
<=	less than or equal to	SELECT * FROM emp WHERE sal <= 12000;

or special SQL operators:

BETWEEN low AND high	SELECT * FROM emp WHERE sal BETWEEN 10000 AND 15000;
IN (VALUE1, VALUE2, VALUE3.....)	SELECT * FROM emp WHERE job IN ('CLERK', 'ANALYST');
LIKE 'string picture' use '%' and '_' as wildcards within a string picture. Each _ acts as one character.	SELECT * FROM emp WHERE ename LIKE '%A%'; % is for any number of characters Selects all employees with an 'A' in their name. Starting with 'A' would be 'A%' and ending with 'A' would be '%A'
IS NULL (IS may only be used with NULL's. This means the variable has no value. You cannot use = NULL)	SELECT ename FROM emp WHERE comm IS NULL;

You can also **NOT** any of the above expressions (used for negation purposes).

Note: Oracle is case sensitive for data items in speech marks!

Try the following examples:

List the names, numbers and departments of all the Clerks.	SELECT ename, empno, deptno FROM emp WHERE job = 'CLERK';
Find the employee's name, salary and commission, whose commission is greater than their salary.	SELECT ename, sal, comm FROM emp WHERE comm > sal;
Finds the name, job role and salary of employees who earn between 12,000 and 14,000	SELECT ename, job, sal FROM emp WHERE sal BETWEEN 12000 AND 14000;

Selecting rows within a range, the WHERE clause can have a low-value and a high-value associated with it, these values represent the bottom and top of the required range.

NOT BETWEEN means that only rows that are outside the range will be selected.

Finds the employees who are clerks, analysts, or salesmen.	SELECT ename FROM emp WHERE job IN ('CLERK', 'ANALYST', 'SALESMAN');
NOT IN would list those employees whose jobs are not in the list.	SELECT ename FROM emp WHERE job NOT IN ('CLERK', 'ANALYST', 'SALESMAN');
Finds the departments that employees called Ford work in.	SELECT ename, deptno FROM emp WHERE ename = 'FORD';
Finds employee names that have an A as the 3rd letter i.e. Blake, Clark etc. (Note - there are 2 underscores before the A)	SELECT ename, deptno FROM emp WHERE ename LIKE '__A%';

NOTE: Although dates are alphanumeric (with quotes), you can do arithmetic on them

e.g. SYSDATE + 1 is tomorrow
SYSDATE - 1 is yesterday

Multiple search conditions may be used in a select statement, linked by either **AND** (both statements must be true for a row to be selected) or **OR** (only one condition must be true for a row to be selected). **AND** and **OR** may be combined to produce complex search conditions and for clarity and reliability should be parenthesised to force precedence. Otherwise, normal computing rules apply.

Try the following:

Find everyone whose job title is manager, and all the clerks in department 10	SELECT * FROM emp WHERE job = 'MANAGER' OR job = 'CLERK' AND deptno = 10; SELECT * FROM emp WHERE job = 'MANAGER' OR (job = 'CLERK' AND deptno = 10); (use of parentheses to clarify.)
Find all the managers or clerks in department 10.	SELECT * FROM emp WHERE (job = 'MANAGER' OR job = 'CLERK') AND deptno = 10;
Any group of search conditions can be negated by enclosing the statement in parentheses and	SELECT * FROM emp WHERE NOT (job = 'MANAGER'

preceding them with NOT.

Find anyone who is neither a manager nor a clerk but is in department 10.

**OR job = 'CLERK')
AND deptno = 10;**

The ORDER BY Clause

By default, Oracle will display rows of data in a totally unordered way. The **ORDER BY** clause should be used to impose an ordering of the rows retrieved by a query and should always be placed last in the query (or query block).

Try this:

SELECT * FROM emp (notice how the EMPNO values are unordered).

Now try it with an ORDER BY:

**SELECT * FROM emp
ORDER BY empno (now the EMPNO values have been ordered).**

The use of **ORDER BY** causes data to be sorted (by default) as follows:

NUMERICS ascending order by value

DATES chronological order

CHAR alphabetically

The keyword **DESC** causes the sort to be reversed.

A strange thing to note is that **NULL** values in a sorted column will always be sorted high, i.e. they will be first when values are sorted in descending order and last when sorted in ascending order.

Shows details of employees with earliest hiredates first.	SELECT empno, ename, hiredate FROM emp ORDER BY hiredate;
To order all employees by job, and within job, put them in descending salary order	SELECT job, sal, ename FROM emp ORDER BY job, sal DESC;
Lists employees in salary order (salary is the 3rd item in the SELECT list)	SELECT ename, job, sal, comm, deptno FROM emp ORDER BY 3;

Exercises

1. List all information about the employees.
2. List the employee name, employee number, salary and department number for all employees.

3. List details of employees in departments 10 and 30.
4. List all the different jobs that employees do, eliminating duplicates.
5. List the names of the employees who earn less than £20,000.
6. What is the name, job title and employee number of any person in department 20 who earns more than £25000?
7. Find all details for employees that are clerks or salesmen.
8. List any clerks who is not in department 10.
9. List the details of department 40.
10. Find everyone whose job is Salesman and all the Analysts in department 20.
11. List the employees whose name ends with "s" in alphabetical order.
12. List the name, employee number, department number and hire date of all salesmen in department 30 who have a salary more than £18,000.
13. List the employee name, department and salary of all employees who earn between £15,000 and £20,000.
14. Find the name of the President.
15. List the employees whose names have TH or LL in them.
16. List the name, job, salary, hiredate, and department number of all employees by alphabetical order of name.
17. List the names, jobs, salaries and commissions of all employees who do not have managers.
18. List only those employees who receive commission (including 0 commission).
19. Find the name, job, salary, hiredate and department number of all employees in ascending order by their salaries.
20. List all salesmen in descending order by commission divided by their salary.
21. Order employees in department 30 who receive commission, in ascending order by commission.
22. What is the name, job title and employee number of any person in department 20 who earns more than £25000?

Now have a go at writing some questions for other things you might want to find out from the data. After you have solved them with queries, swap with the person next to you to see if they can write queries for them too.

Use any remaining lab time looking at the SQL resources on [W3 Schools](#) and trying out some of the examples.

Any work not finished should be completed in your own time before the next class.