
Lab Week - 1

Writing basic SQL statements

Writing basic SQL statements

- In this session:
 - Objectives of SQL
 - Capabilities of SQL SELECT statements
 - Writing SQL statements
 - Execution of a basic SELECT statement
 - Arithmetic Operator precedence
 - Nulls and Aliases
 - Concatenation
 - Literal strings
 - Limiting outputs

Objectives of SQL

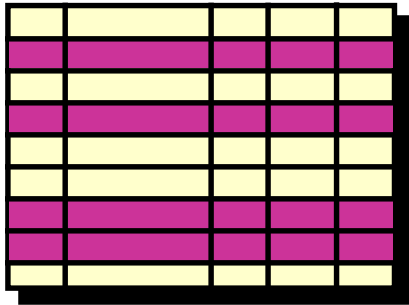
- Ideally, database language should allow user to:
 - create the database and relation structures;
 - perform insertion, modification, deletion of data from relations;
 - perform simple and complex queries.
- Must perform these tasks with minimal user effort and command structure/syntax must be easy to learn.
- It must be portable.

Objectives of SQL

- SQL is relatively easy to learn:
 - it is non-procedural - you specify *what* information you require, rather than *how* to get it;
 - it is essentially free-format.

Capabilities of SQL SELECT Statements

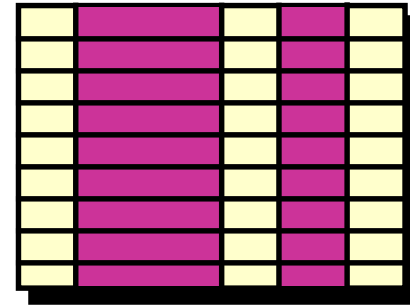
Selection



A 10x5 grid representing a table. The first, third, fifth, seventh, and ninth rows are highlighted in pink, while the second, fourth, sixth, eighth, and tenth rows are yellow.

Table 1

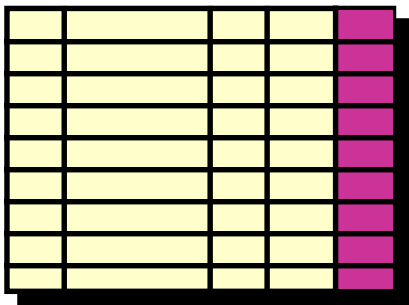
Projection



A 10x5 grid representing a table. The first, third, fifth, seventh, and ninth columns are highlighted in pink, while the second, fourth, sixth, eighth, and tenth columns are yellow.

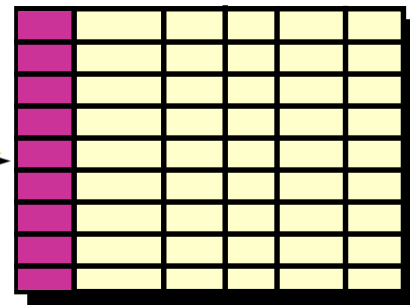
Table 1

Join



A 10x5 grid representing a table. The first, third, fifth, seventh, and ninth columns are highlighted in pink, while the second, fourth, sixth, eighth, and tenth columns are yellow.

Table 1



A 10x5 grid representing a table. The first, third, fifth, seventh, and ninth rows are highlighted in pink, while the second, fourth, sixth, eighth, and tenth rows are yellow.

Table 2

Basic SELECT Statement

- SELECT identifies what columns.
- FROM identifies which table.

```
SELECT    [DISTINCT] {*, column [alias],...}  
FROM      table;
```

Writing SQL Statements

- SQL statement consists of *reserved words* and *user-defined words*.
 - Reserved words are a fixed part of SQL and must be spelt exactly as required and cannot be split across lines.
 - User-defined words are made up by user and represent names of various database objects such as relations, columns, views.

Writing SQL Statements

- Most components of an SQL statement are *case insensitive*, except for literal character data.
- More readable with indentation and lineation:
 - Each clause should begin on a new line.
 - Start of a clause should line up with start of other clauses.
 - If clause has several parts, should each appear on a separate line and be indented under start of clause.

Selecting All Columns

```
SELECT *  
FROM departments ;
```

DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
10	Administration	200	1700
20	Marketing	201	1800
50	Shipping	124	1500
60	IT	103	1400
80	Sales	149	2500
90	Executive	100	1700
110	Accounting	205	1700
190	Contracting		1700

8 rows selected.

Selecting Specific Columns

```
SELECT department_id, location_id
FROM departments;
```

DEPARTMENT_ID	LOCATION_ID
10	1700
20	1800
50	1500
60	1400
80	2500
90	1700
110	1700
190	1700

8 rows selected.

Arithmetic Expressions

- Create expressions on NUMBER and DATE data by using arithmetic operators.

Operator	Description
+	Add
-	Subtract
*	Multiply
/	Divide

Using Arithmetic Operators

```
SELECT last_name, salary, salary + 300
FROM   employees;
```

LAST_NAME	SALARY	SALARY+300
King	24000	24300
Kochhar	17000	17300
De Haan	17000	17300
Hunold	9000	9300
Ernst	6000	6300

■■■
20 rows selected.

Operator Precedence

- Multiplication and division take priority over addition and subtraction.
- Operators of the same priority are evaluated from left to right.
- Parentheses are used to force prioritised evaluation and to clarify statements.



Operator Precedence

```
SELECT last_name, salary, 12*salary+100
FROM employees;
```

1

LAST_NAME	SALARY	12*SALARY+100
King	24000	288100
Kochhar	17000	204100
De Haan	17000	204100

■ ■ ■

20 rows selected.

```
SELECT last_name, salary, 12*(salary+100)
FROM employees;
```

2

LAST_NAME	SALARY	12*(SALARY+100)
King	24000	289200
Kochhar	17000	205200
De Haan	17000	205200

■ ■ ■

20 rows selected.

Defining a Null

- A null is a value that is unavailable, unassigned, unknown, or inapplicable.
- A null is not the same as a zero or a blank space.

```
SELECT last_name, job_id, salary, commission_pct  
FROM employees;
```

LAST_NAME	JOB_ID	SALARY	COMMISSION_PCT
King	AD_PRES	24000	
Kochhar	AD_VP	17000	
...			
Zlotkey	SA_MAN	10500	.2
Abel	SA_REP	11000	.3
Taylor	SA_REP	8600	.2
...			
Gietz	AC_ACCOUNT	8300	

20 rows selected.

Nulls in Arithmetic Expressions

- Arithmetic expressions containing a null value evaluate to null.

```
SELECT last_name, 12*salary*commission_pct  
FROM employees;
```

Kochhar	
King	
LAST_NAME	12*SALARY*COMMISSION_PCT
...	
Zlotkey	25200
Abel	39600
Taylor	20640
...	
Gietz	

20 rows selected.

Defining a Column Alias

- Renames a column heading
- Is useful with calculations
- Immediately follows column name; **optional** AS keyword between column name and alias
- Requires double quotation marks if it contains spaces or special characters or is case sensitive.

Using Column Aliases

```
SELECT last_name AS name, commission_pct comm
FROM employees;
```

NAME	COMM
King	
Kochhar	
De Haan	

...

20 rows selected.

```
SELECT last_name "Name", salary*12 "Annual Salary"
FROM employees;
```

Name	Annual Salary
King	288000
Kochhar	204000
De Haan	204000

...

Concatenation Operator

- Concatenates columns or character strings to other columns
- Is represented by two vertical bars (||)
- Creates a resultant column that is a character expression

Using the Concatenation Operator

```
SELECT    last_name||job_id AS "Employees"  
FROM      employees;
```

Employees
KingAD_PRES
KochharAD_VP
De HaanAD_VP

■ ■ ■

20 rows selected.

Literal Character Strings

- A literal is a character, expression, or number included in the SELECT list.
- Date and character literal values must be enclosed within single quotation marks.
- Each character string is output once for each row returned.

Using Literal Character Strings

```
SELECT last_name || ' is a ' || job_id  
        AS "Employee Details"  
FROM    employees;
```

Employee Details
King is a AD_PRES
Kochhar is a AD_VP
De Haan is a AD_VP
Hunold is a IT_PROG
Ernst is a IT_PROG
Lorentz is a IT_PROG
Mourgos is a ST_MAN
Rajs is a ST_CLERK

...

20 rows selected.

Duplicate Rows

- The default display of queries is all rows, including duplicate rows.

```
SELECT department_id  
FROM employees;
```

1

DEPARTMENT_ID	
	90
	90
	90

...

20 rows selected.

```
SELECT DISTINCT department_id  
FROM employees;
```

2

DEPARTMENT_ID	
	10
	20
	50

...

Limiting Rows Using a Selection


EMPLOYEES

EMPLOYEE_ID	LAST_NAME	JOB_ID	DEPARTMENT_ID
100	King	AD_PRES	90
101	Kochhar	AD_VP	90
102	De Haan	AD_VP	90
103	Hunold	IT_PROG	60
104	Ernst	IT_PROG	60
107	Lorentz	IT_PROG	60
124	Mourgos	ST_MAN	50

...

20 rows selected.

**“retrieve all
employees in
department 90”**



EMPLOYEE_ID	LAST_NAME	JOB_ID	DEPARTMENT_ID
100	King	AD_PRES	90
101	Kochhar	AD_VP	90
102	De Haan	AD_VP	90

Limiting Rows Selected

- Restrict the rows returned by using the WHERE clause.
- The WHERE clause follows the FROM clause.

```
SELECT          [DISTINCT] { * | column [alias], ... }  
FROM            table  
[WHERE          condition(s) ] ;
```

Using the WHERE Clause

```
SELECT employee_id, last_name, job_id, department_id
FROM   employees
WHERE  department_id = 90 ;
```

EMPLOYEE_ID	LAST_NAME	JOB_ID	DEPARTMENT_ID
100	King	AD_PRES	90
101	Kochhar	AD_VP	90
102	De Haan	AD_VP	90

Using the WHERE Clause - Another example

```
SELECT last_name, job_id, department_no  
FROM employees  
WHERE job_id = 'CLERK' ;
```

ENAME	JOB	DEPTNO
-----	-----	-----
JAMES	CLERK	30
SMITH	CLERK	20
ADAMS	CLERK	20
MILLER	CLERK	10

Displaying Table Structure

- Use the *iSQL*Plus* DESCRIBE command to display the structure of a table:

```
DESC[RIBE] tablename
```

Displaying Table Structure

```
DESCRIBE employees
```

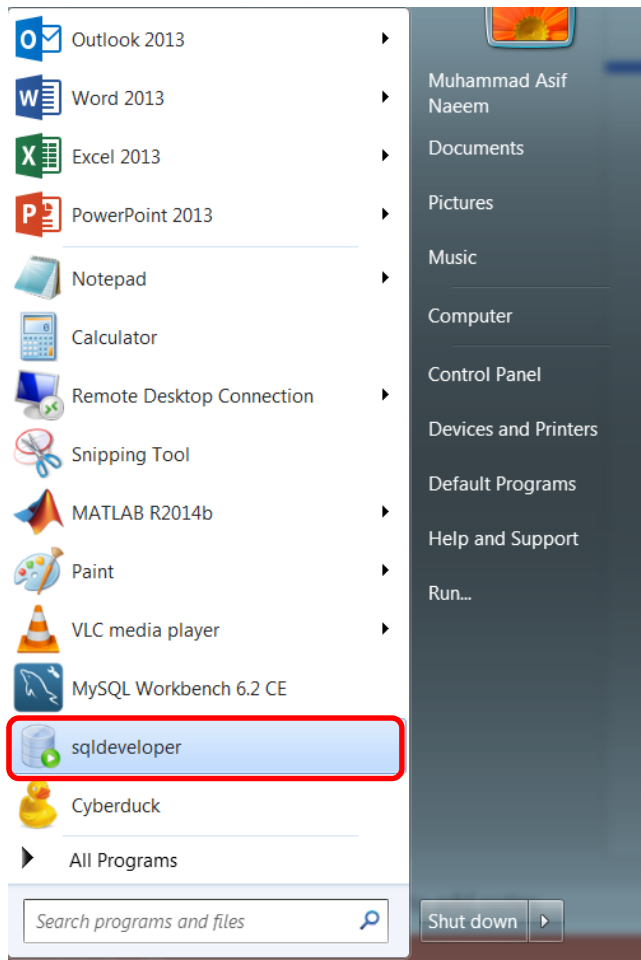
Name	Null?	Type
EMPLOYEE_ID	NOT NULL	NUMBER(6)
FIRST_NAME		VARCHAR2(20)
LAST_NAME	NOT NULL	VARCHAR2(25)
EMAIL	NOT NULL	VARCHAR2(25)
PHONE_NUMBER		VARCHAR2(20)
HIRE_DATE	NOT NULL	DATE
JOB_ID	NOT NULL	VARCHAR2(10)
SALARY		NUMBER(8,2)
COMMISSION_PCT		NUMBER(2,2)
MANAGER_ID		NUMBER(6)
DEPARTMENT_ID		NUMBER(4)

SQL and SQL Plus

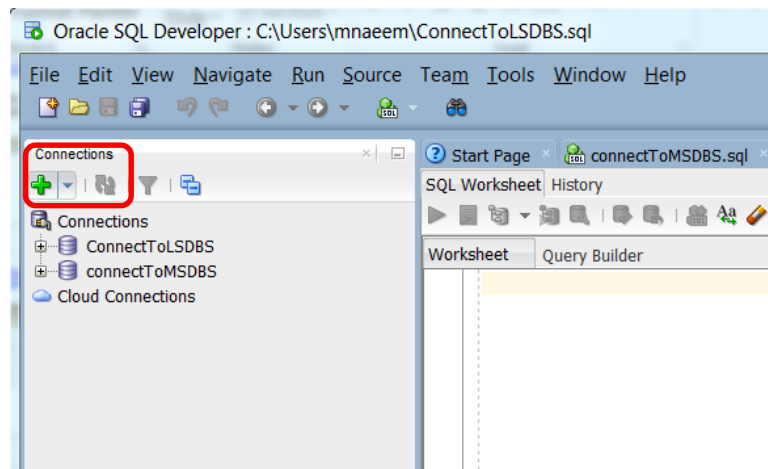
- SQL
 - A command language for communication with the database server from any tool or application
- SQL Developer
 - Is an Oracle tool that recognizes and submits SQL statements to the Oracle server for execution and contains its own command language

How to access SQL Developer?

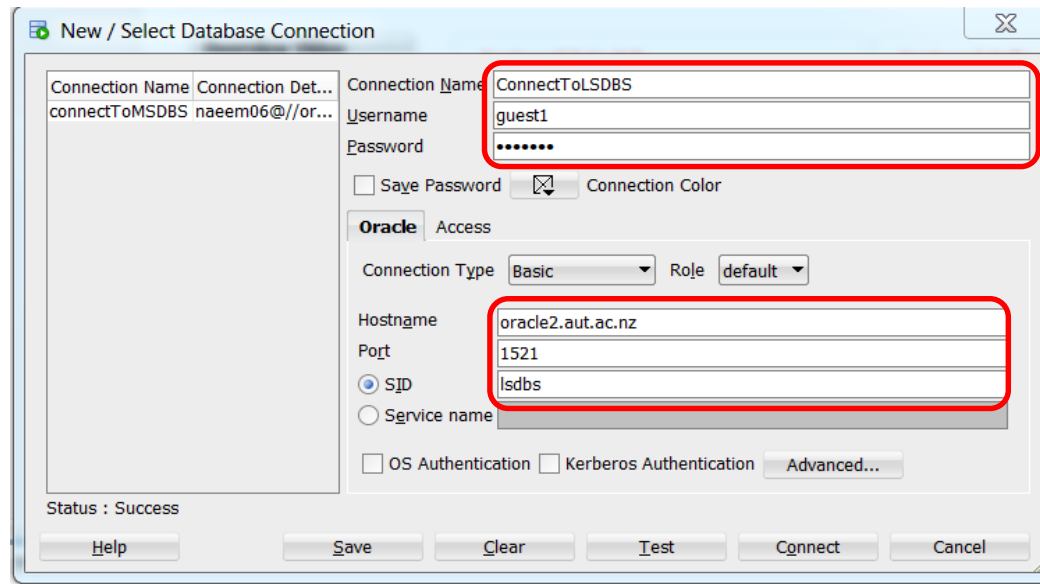
Step 1



Step 2



Step 3



Lab Activities

- Complete SQL exercise
- Using the scripts provided on AUTOnline create the Hotel and HR schemas, using SQL Developer.