Lab 03 – SQL

Single Table Queries/DML Statements

# **Objectives**

The purpose of this lab is to start learning SQL by writing basic SELECT and DML statements involving a single table. You will learn to create basic CRUD statements (queries as well as insert, update and delete).

**Submission**

***Your submission will be a single SQL file with the solutions provided. (with a .sql file extension)***

Using comments to number the question answers, write the SQL code to complete the following tasks.

Example Submission

|  |
| --- |
| -- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  -- Name: Your Name  -- ID: #########  -- Date: The current date  -- Purpose: Lab 03 DBS211 – Part 1  -- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  -- Q1 SOLUTION --  SELECT \* FROM TABLE; -- Your solution  -- Q2 Solution –  SELECT \* FROM TABLE; -- Your solution |

* Save questions 1 to 10 in Part 1 as a script named: DBS211\_L03\_LastName\_Part1.sql
* Save questions 1 to 8 in Part 2 as a script named: DBS211\_L03\_LastName\_Part2.sql
* Locate, select, and submit both files to the Lab 03 link.

## **Setup**

Create a new worksheet in SQL developer to write your solutions of each part and add an appropriate comment header that includes your name, student id, the date and the purpose of the file (i.e. DBS211 – Lab 03).

Immediately under the comment header, enter the following line and then execute it:

SET AUTOCOMMIT ON;

You will need to execute this statement each time you login to the server until the completion of this lab.

## **Style Guide**

Your SQL should be written using the standard coding style:

* all keywords are to be upper case,
* all user-defined names are to be lower case, (example: table and field names)
* there should be a carriage return before each major part of the SQL statements (i.e. before SELECT, FROM, WHERE and ORDER BY)

See the following sample:

SELECT columns

FROM tables

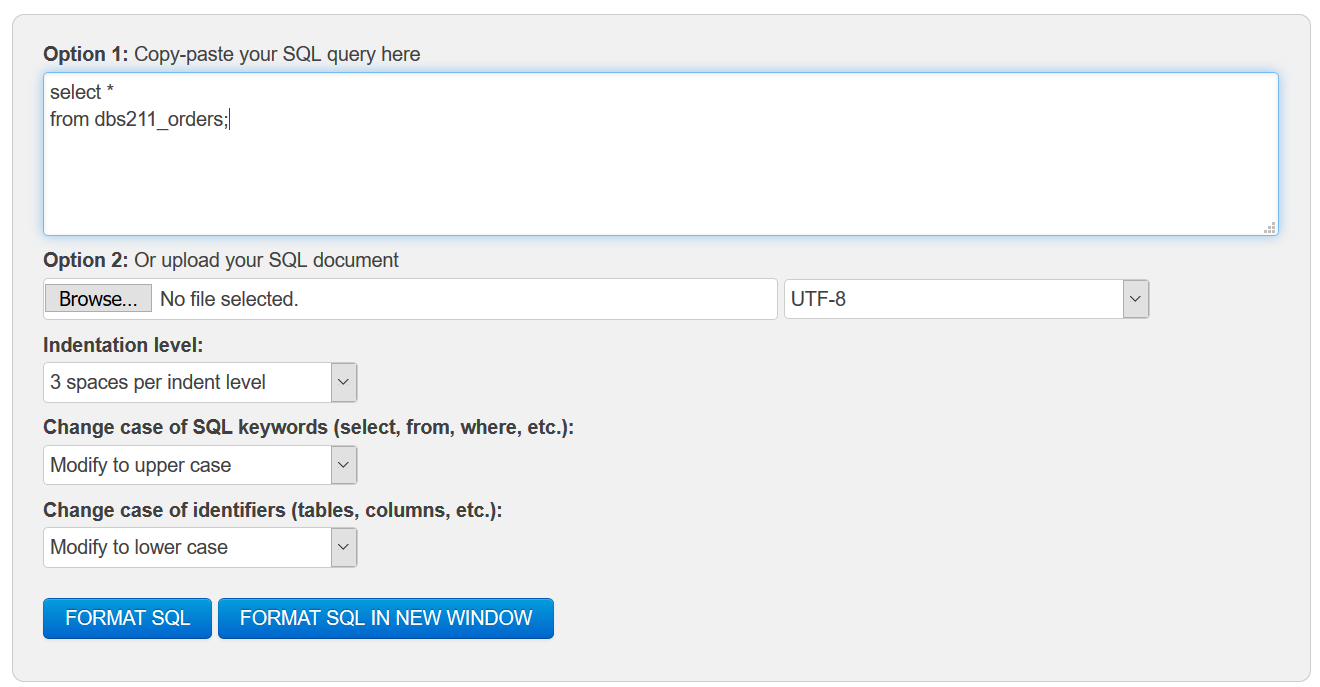
WHERE conditions

ORDER BY column1, column2;

To save time, you can write all SQL statement in your SQL developer. To make sure that your SQL statements style follows the standard SQL style guide, copy and paste your SQL statement onto the following website and click on “FORMAT SQL” or “FORMAT SQL IN NEW WINDOW”.

<https://www.freeformatter.com/sql-formatter.html#ad-output>

You can also upload your SQL file. See the setting in the following image. Have SQL keywords (SELECT, INSERT, UPDATE, etc) uppercase and user defined objects and identifiers (tables, columns, etc.) lowercase.



**Marking Scheme**

**Part 1**

|  |  |  |  |
| --- | --- | --- | --- |
| **Question** | **Weight** | **Question** | **Weight** |
| 1 | 6% | 6 | 6% |
| 2 | 6% | 7 | 6% |
| 3 | 6% | 8 | 6% |
| 4 | 6% | 9 | 6% |
| 5 | 6% | 10 | 6% |

**Part 2**

|  |  |  |  |
| --- | --- | --- | --- |
| **Question** | **Weight** | **Question** | **Weight** |
| 1 | 5% | 5 | 5% |
| 2 | 5% | 6 | 5% |
| 3 | 5% | 7 | 5% |
| 4 | 5% | 8 | 5% |

**Grade Policy**

* Submissions with errors do not get any marks. (They get zero.)
  + Execute your *.sql* file using the “Run Script” button to make sure there is no errors in your file.
* If your result in a question does not match the sample output results, you do not get any marks.
* You do not receive marks for the missing solutions.

**Tasks:**

**Part 1 (SQL SELECT)**

There are 6 questions in Part 1.

***The score of each question is 1.***

***Queries with errors ro incorrect queries will not receive any marks.***

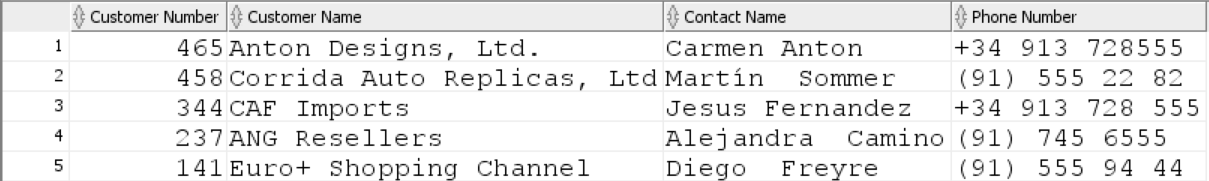
***If your query is partially correct, you will receive half of the score.***

Save the answers for questions in Part 1 as a script named **DBS211\_L03\_LastName\_Part1.sql**.

Remember to comment the question number for each solution.

1. Display customer number, customer name, contact first name and contact last name (in the following format), and phone for all customers in Madrid. (**hint**: be wary of case sensitivity) *Sort the result based on the customer number from high to low.*

Your output must be like the following output.



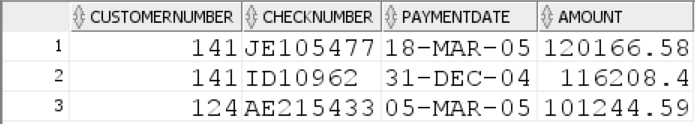
1. Display customer number for customers who have payments. Do not included any repeated   
   values. *Sort the result based on the customer number.* (**Hints**: How do you know a customer has made a payment? You will need to access only one table for this query)

*Show only the first 10 rows.*

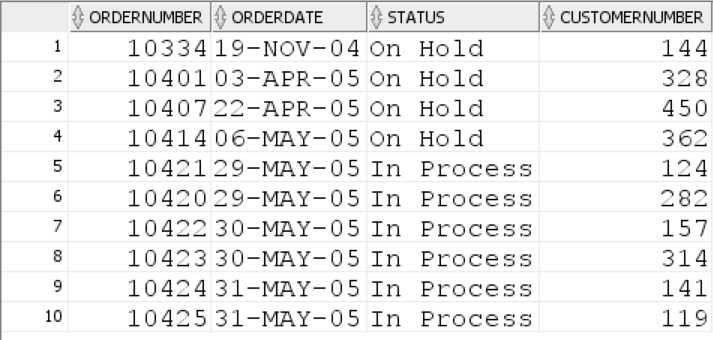


1. List customer numbers, check number, and amount for customers whose payment amount is greater than $100,000 and the payment date is not in year 2003.

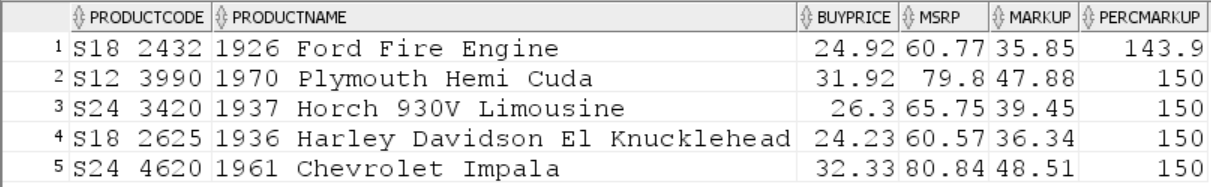
Sort the output by top payments amount first.



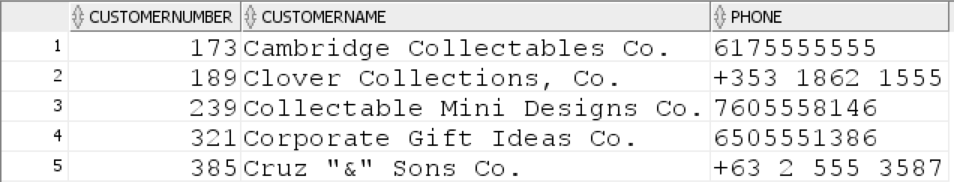
1. Display *order number, order date, status, and customer number* for all orders that are in process. Display also the orders on hold. *Sort the result according to order date.*



1. The company needs to know the percentage markup for each product sold. Produce a query that outputs the ProductCode, ProductName, BuyPrice, MSRP in addition to
   1. The difference between MSRP and BuyPrice (i.e. MSRP-BuyPrice) called *markup*
   2. The percentage markup (100 \* calculated by difference / BuyPrice) called *percmarkup*rounded to 1 decimal place.
   3. *Sort the result according to percmarkup.*
   4. *Show products with percmarkup greater than 140.*



1. Display *customer number, customer name and phone* for all customers whose customer name starts with letter ***c*** (both lowercase and uppercase) and ends with ‘Co.’. *Sort the result according to customer number.*



**Part 2 (DML, Data Manipulation Language)**

***There are 8 questions in Part 2.***

***The score of each question is 0.5.***

***Answers with errors ro incorrect answers will not receive any marks.***

Save the answers for questions in Part 2 as a script named **DBS211\_L03\_LastName\_Part2.sql**.

Remember to comment the question number for each solution.

Before you start answering questions in this section, first create a table like dbs211\_employees named employees using the following statement:

**CREATE TABLE employees AS**

**SELECT \***

**FROM dbs211\_employees;**

Please run the following ALTER statements:

**ALTER TABLE employees  
ADD CONSTRAINT employees\_pk  
PRIMARY KEY (employeenumber);**

**ALTER TABLE employees  
ADD CONSTRAINT employees\_fk  
FOREIGN KEY (REPORTSTO)  
REFERENCES employees (employeenumber);**

Answer the following questions using the ***employees*** table.

1. Create a statement that will insert yourself as an employee of the company in the employees table. ()
   1. Use a unique employee number of your choice (***Hint:*** *Find the highest value of the employee number in the employees table, increase the value by one and use it as your employee number.*)

*To find the highest value of the employee number you can sort the rows in the descending order. The first row will then contain the highest value.*

*Or, you can run the following statement (Do not include this statement in your submission.)*

**SELECT max(employeenumber)**

**FROM employees;**

*This statement returns the maximum value of the employee number in table employees*.

* 1. Use your school email address
  2. *Your extension is ‘x32325’*
  3. Your job title will be “*Accountant*”
  4. Office code will be 4
  5. You will report to employee 1088

1. Create a query that displays your, and only your employee data from ***employees***.
2. Create a statement to update your job title to “Head Cashier”.

***Hint:*** *Be careful. You may update other rows or all rows in the employee table. You only need to update one row which belongs to you and update your job title. Make sure that your query updates only one employee using a WHERE clause.*

1. Create a statement to insert another fictional employee into the ***employees*** table. This employee will be a “Cashier” and will report to you. Make up fake data for the other fields.
2. Create a statement to remove yourself from the employees table. Did it work? If not, why?
3. Create a statement to remove the fake employee from the database and then rerun the statement to remove yourself. Did it work?
4. Create a **single** statement that will insert both yourself and the fake employee at the same time. This time you and the fake employee will report to 1088.
5. Create a **single** statement to remove both yourself and the fake employee from the ***employees*** table.

**Write the following sratement to remove the *employees* table at the end of your queries.**

**DROP TABLE employees;**