Lab 04 – Multi-Table Queries and Views

# Objective:

The purpose of this lab is to introduce students to querying data from multiple tables. Relationships are used in relational databases to reduce redundant and repetitive data, but it is necessary to reconnect these tables when extracting data and obtaining information. Students will be able to:

* produce query results containing data from multiple tables using joins and demonstrate their knowledge of inner, outer and full joins.
* To actively troubleshoot queries to handle potentially ambiguous fields across multiple tables through the use of aliases.
* Students learn to create and modify views.

# Submission:

***Your submission will be a single text-based .SQL file with the solutions provided. Any other file format will not be accepted and will get a 0.***

DBS211\_L04\_FirstName\_LastName.sql

Your submission needs to include a comment header block and be commented to include the questions and the solutions. Make sure every SQL statement terminates with a semicolon.

Example Submission

|  |
| --- |
| -- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  -- Name: Your Name  -- ID: #########  -- Date: The current date  -- Purpose: Lab 04 DBS211  -- \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  -- Q1 SOLUTION --  SELECT \* FROM TABLE;  -- Q2 SOLUTION –  SELECT \* FROM TABLE; |

Locate, select, and submit the file to the Lab 04 link. Proper use of Aliases should be demonstrated wherever applicable as discussed in class lectures. Any error in queries/sql file will result in getting 0 for the erroneous answer.

Please note: All the queries to be created in this Lab are expected to be using the Join operation as demonstrated in the class. The Joins used must be of ANSI-92 (new syntax). Unnecessary use of extra tables in Joins would lead to marks deduction.

# Tasks (Each question carries 1/2 marks):

## Select data from multiple tables / Create Views

1. Create a query that displays all payments made by customers from Canada keeping in mind the below points.
   * Sort the output by Customer Number.
   * Only display the Customer Number, Customer Name, Payment Date and Amount.
   * Make sure the date is displayed clearly to know what date it is (i.e. in the format Apr 2, 2019)
2. Create a query that shows all those employees who have not been assigned to any customer. Display only the employee number, employee full name (combination of first and last name separated by a space) and email. Sort the results by employee full name.
3. Create a view (***vw\_customer\_order***) to list all orders with the following data for all customers:

Customer Number, order number, order date, product name, quantity ordered, and price for each product in every order.

1. Using the ***vw\_customer\_order*** view, display the order information for customer number 124. Sort the output based on order number and then order line number (yes, Order Line Number is not in the view). You are free to choose the columns to display.
2. Create a query that displays the customer number, contact first name, contact last name, phone, and credit limits for all customers who do not have any orders.
3. Create a view ***vw\_employee\_manager*** to display all the information of all employees and the full name of their managers if there is any manager that the employee reports to. Include all employees, including those who do not report to anyone.
4. Modify the ***vw\_employee\_manager*** view so that the view returns all information of all employees and the full name of their managers only for employees who have a manager. Do not DROP and recreate the view – modify it.
5. Drop both the ***vw\_customer\_order*** and ***vw\_employee\_manager*** views.