

**Company’s database management system**

[Document subtitle]



November 19, 2023

Dhyey Bhatt

**RECORDS INSERTED**:

**1. Add a new customer in Customer table with name “George” and id as “2” followed by other details as shown in the image below.**

A screenshot of a computer

Description automatically generated

* 1. **Addition of at least 2 accounts related to George in Accounts table.**

🡺Here, inside the query the customer ID is used as foreign key inside the account table, that is given id as “2” which represented the accounts used by the customer named “George” which was inserted in the record previously using insert into command.

A screenshot of a computer

Description automatically generated

**2. Addition of a new branch with location (City) as ‘New Market’ and added few orders in orders table associated with this branch.**

🡺Here, the Branch created is given the id as “2” as shown in the image below using insert into command.

A screenshot of a computer

Description automatically generated

🡺Here the new orders added are given the Customer id as “2”

A screenshot of a computer

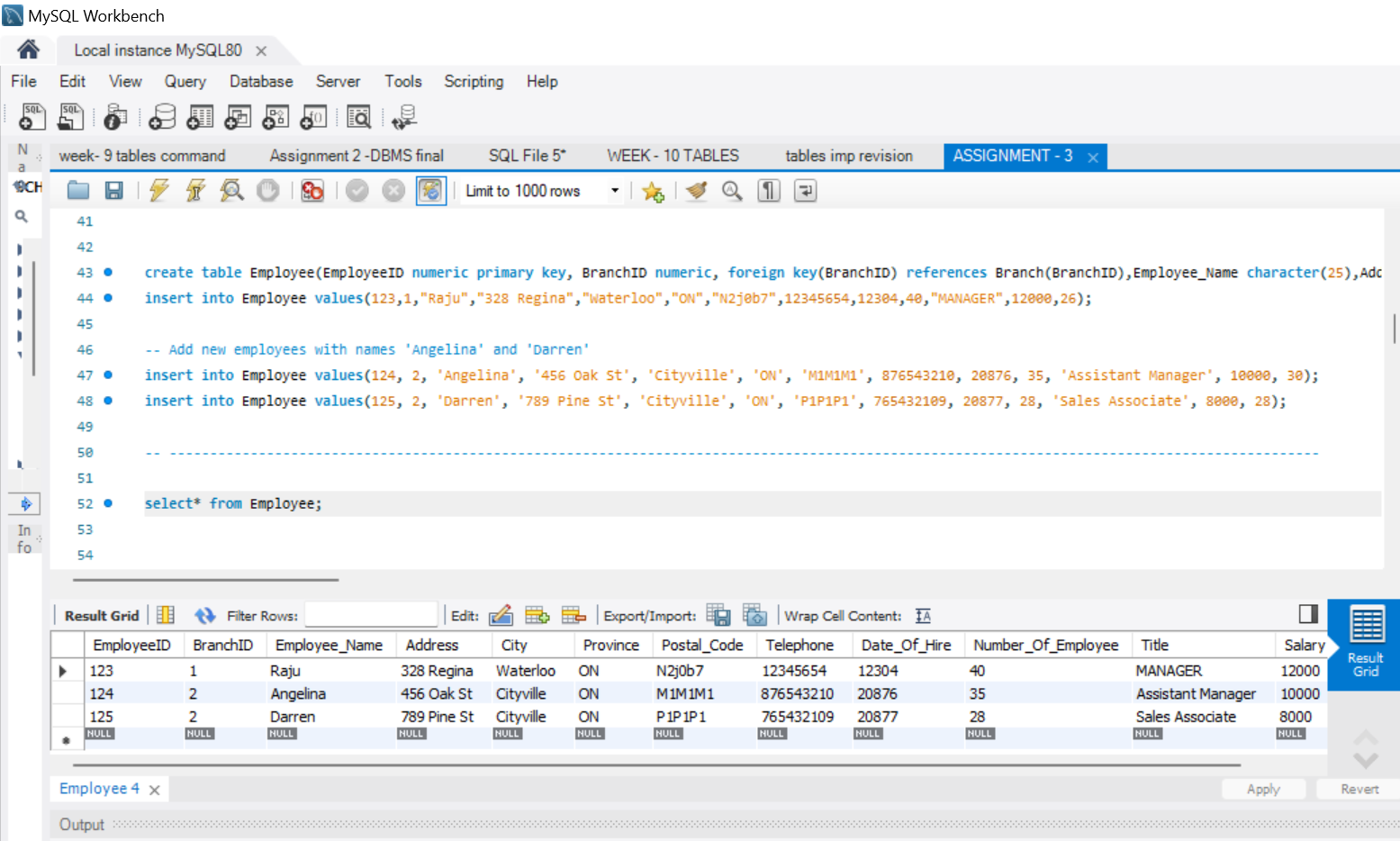
Description automatically generated

🡺The customer id “2” is placing order from the new branch with Branch\_Id as “2” (“new market”)

A screenshot of a computer

Description automatically generated

**3. Added new employees with names ‘Angelina’ and ‘Darren’**



🡺 Addition of at least a dependent “Aria” 🡺 “Angelina” and “Daniel” 🡺”Darren” for each of new employees created, inside the dependent relation table.

A screenshot of a computer

Description automatically generated

**4. Added few records in Branch table.**

( No need to add any employees or any orders associated with it.) since this condition does not qualify the business rules mentioned in the previous assignment, I have added this to make sure that business requirements are matched properly.

A screenshot of a computer

Description automatically generated

🡺I have added employee as well because each branch has at least one employee as mentioned in business rule.

A screenshot of a computer

Description automatically generated

**Now let’s move on to queries:**

1. **Listed the information of all accounts with the name of the customers holding those accounts.**

A screenshot of a computer

Description automatically generated

The query is retrieving information from two tables, **Accounts** (aliased as **a**) and **Customer** (aliased as **c**), using an inner join. The join is performed based on the common column **CustomerID** between these tables.

Here's a step-by-step explanation:

1. **Select Statement**: **SELECT a.\*, c.Customer\_Name**
   * This part of the query specifies the columns to be included in the result set.
   * All columns from the **Accounts** table (**a**) are selected.
   * And the **Customer\_Name** column from the **Customer** table (**c**).
2. **FROM Accounts a INNER JOIN Customer c ON a.CustomerID = c.CustomerID**
   * The query indicates the tables involved as well as the condition for joining those tables.
   * Data from the **Accounts** table (**a**) and the **Customer** table (**c**) is selected as stated.
   * The **INNER JOIN** keyword which is used verifies that only rows with matching **CustomerID** values in both tables are included in the result shown at the end.
3. **Join Condition**: **ON a.CustomerID = c.CustomerID**
   * The **CustomerID** column is used to match records between the two tables.
4. **Result Set**: The result set includes all columns from the **Accounts** table (**a**) and the **Customer\_Name** column from the **Customer** table (**c**)

**====================================================================================**

**2. Listed all the orders placed from a branch whose location (city – branch name) has space in its name and also displayed location (Branch – name) with it.**

A screenshot of a computer

Description automatically generated

1. 🡺**Select Statement**: **SELECT ob.\*, b.Branch\_Name**
   * The query specifies the columns to be included in the result set at the end.
   * All columns from the **Order\_Branch** table (**ob**) is selected and then **Branch\_Name** column from the **Branch** table (**b**) is selected as well.
2. **FROM Order\_Branch ob INNER JOIN Branch b ON ob.OrderID = b.BranchID**
   * The query indicates the tables involved and the condition for joining them.
   * Data from the **Order\_Branch** table (**ob**) and the **Branch** table (**b**) as well are selected.
   * The **INNER JOIN** keyword that is used ensures that only rows with matching **OrderID** in **Order\_Branch** and **BranchID** in **Branch** are included in the result at the end.
3. **Join Condition**: **ON ob.OrderID = b.BranchID**
   * This specifies the condition for joining the tables.
4. **Where Clause**: **WHERE b.Branch\_Name LIKE '% %'**
   * This part filters the results based on a condition.
   * Those rows where the **Branch\_Name** in the **Branch** table contains at least one space are selected.
5. **Result Set**: The final result set includes all columns from the **Order\_Branch** table (**ob**) and the **Branch\_Name** column from the **Branch** table (**b**) where the specified condition is met.

**3. Printed the information of dependents with the employee names with whom they are associated for the employees whose name starts with character ‘A’ or ‘D’.**

A screenshot of a computer

Description automatically generated

1. 🡺**Select Statement**: **SELECT d.\*, e.Employee\_Name**
   * The query specifies the columns to be included in the result set at the end.
   * It selects all columns from the **Dependent** table (**d**)
   * And the **Employee\_Name** column from the **Employee** table (**e**).
2. **FROM Dependent d INNER JOIN Employee e ON d.EmployeeID = e.EmployeeID**
   * The query indicates the tables involved and the condition for joining them as well.
   * It selects data from the **Dependent** table (**d**)
   * And the data from **Employee** table (**e**).
   * The **INNER JOIN** keyword that is used ensures that only rows with matching **EmployeeID** in both tables are included in the result.
3. **Join Condition**: **ON d.EmployeeID = e.EmployeeID**
   * This is the condition for joining the tables.
   * It connects the **EmployeeID** column in the **Dependent** table with the **EmployeeID** column in the **Employee** table.
4. **WHERE e.Employee\_Name LIKE 'A%' OR e.Employee\_Name LIKE 'D%' is the query.**
   * Those rows where the **Employee\_Name** in the **Employee** table (**e**) starts with either 'A' or 'D' to satisfy the requirements are selected.
5. **Result Set**: The result set includes all columns from the **Dependent** table (**d**) as well as the **Employee\_Name** column from the **Employee** table (**e**).

**4. Listed all the orders with their order ids and all the items included in those orders in ascending order of order ids. Item names are appearing every row with order ids.**

**A screenshot of a computer

Description automatically generated**

**Here, order id 3 and 4 have ordered same item with id “3” or “washing machine”.**

1. **SELECT ol.OrderID, ol.ItemID, i.Item\_Description**
   * The query shows the columns to be included in the result set.
   * It selects the OrderID and ItemID columns from the Order\_List table (ol) and along with these the Item\_Description column from the Item table (i).
2. **FROM Order\_List ol INNER JOIN Item i ON ol.ItemID = i.ItemID**
   * This part of the query indicates the tables involved and the condition for joining them.
   * It selects data from the Order\_List table (ol) and the Item table (i).
   * The INNER JOIN keyword which is used, ensures that only rows with matching ItemID in both tables are included in the result.
3. **ON ol.ItemID = i.ItemID**
   * This specifies the condition for joining the tables.
   * The ItemID column in the Order\_List table with the ItemID column in the Item table are connected in turn.
4. **ORDER BY ol.OrderID**
   * This part of the query sorts the result set in ascending order based on the OrderID column.
5. **Result Set:** The result set includes columns from the Order\_List table (ol) and the Item table (i) where there are matching ItemID values. The result is sorted by OrderID in ascending order.

**=====================================================================================**

**5. Listed names of the employees working in a ‘Kitchener-Waterloo’ branch with their branch location**

**A screenshot of a computer

Description automatically generated**

1. **SELECT e.Employee\_Name, b.Branch\_Name**
   * This part of the query specifies the columns to be included in the result set.
   * It selects the Employee\_Name column from the Employee table (e) and the Branch\_Name column from the Branch table (b) as well.
2. **FROM Employee e INNER JOIN Branch b ON e.BranchID = b.BranchID**
   * The query indicates the tables involved and the condition in order to join them.
   * It selects data from the Employee table (e)
   * And selects the data from the Branch table (b) as well.
   * The INNER JOIN keyword which is used ensures that only rows with matching BranchID in both tables are included in the result at the end**.**
3. **ON e.BranchID = b.BranchID**
   * This connects the BranchID column in the Employee table with the BranchID column in the Branch table.
4. **WHERE b.Branch\_Name = 'Kitchener-Waterloo'**
   * Those rows where the Branch\_Name in the Branch table (b) is equal to 'Kitchener-Waterloo' as mentioned in the condition are selected.
5. **Result Set:** The result set includes the Employee\_Name and Branch\_Name columns for employees who work in the 'Kitchener-Waterloo' branch as stated.

**6. List all the branches that has no employees associated with it.**

**A screenshot of a computer

Description automatically generated**

**Since this violates the business rule none of the branch is printed or shown as every branch has at least one employee.**

1. **SELECT b.\***
   * The query specifies the columns to be included in the result set.
   * Selecting all columns from the Branch table (b).
2. **FROM Branch b LEFT JOIN Employee e ON b.BranchID = e.BranchID**
   * The query indicates the tables involved in the condition to join them.
   * Data from the Branch table (b) and the Employee table (e) is selected.
   * The LEFT JOIN keyword ensures that all rows from the Branch table are included in the result, (in consideration, regardless of whether there are matching rows in the Employee table).
3. **ON b.BranchID = e.BranchID**
   * The condition for joining the tables connects the BranchID column in the Branch table (b) with the BranchID column in the Employee table(e).
4. **WHERE e.EmployeeID IS NULL**
   * Those rows where there is no matching employee (EmployeeID IS NULL), meaning there are no employees associated with that branch are selected.
5. **Result Set:** The result set includes all columns from the Branch table (b) for branches that have no employees associated with them.