CSCI 5408

DATA MANAGEMENT AND WAREHOUSING

LAB - 4

Banner ID: B00984406

GitLab Assignment Link: <https://git.cs.dal.ca/jems/csci5408_s24_b00984406_jems_patel.git>

**Table of Contents**

1. DDL Comands…………………………………….........................................................................3

2. DML Commands…...………………………………………………………………………………….7

3. Screenshots………….………………………………………………………………...………………9

4. Time Taken Explanation.………………….…………………………………….……….….………12

**DDL Commands**

**Create a database called “local\_lab4”:**

CREATE DATABASE IF NOT EXISTS local\_lab4;

A screenshot of a computer

Description automatically generated   
Figure 1.1: Database local\_lab4 created

**Create a database called “remote\_lab4”:**

CREATE DATABASE IF NOT EXIST remote\_lab4;

A screenshot of a computer

Description automatically generated  
Figure 1.2: Database remote\_lab4 created

**Create a table “user”:**

CREATE TABLE User (

id INT PRIMARY KEY AUTO\_INCREMENT,

name VARCHAR(100) NOT NULL,

email VARCHAR(100) UNIQUE NOT NULL,

phone VARCHAR(15),

address VARCHAR(255)

);–––

A screenshot of a computer

Description automatically generated

Figure 1.3: user table created

**Create a table “Order\_info”:**

CREATE TABLE Order\_info (

order\_id INT PRIMARY KEY AUTO\_INCREMENT,

user\_id INT,

item\_name VARCHAR(100) NOT NULL,

quantity INT NOT NULL,

order\_date DATETIME NOT NULL

);

A screenshot of a computer

Description automatically generated

Figure 1.4: Order\_info table created

**Create a table “Inventory”:**

CREATE TABLE Inventory(

item\_id INT PRIMARY KEY,

item\_name VARCHAR(15),

available\_quantity INT

)

A screenshot of a computer

Description automatically generated

Figure 1.5: Inventory table created

**DML Commands**

**Inserting the data into the table “user”(local\_lab4):**

INSERT INTO User (name, email, phone, address) VALUES

('JEMS', 'jems007patel@gmail.com', '7828825653', '1881 Brunswick Street'),

('JOLI', 'joli130@gmail.com', '7855572425', '2001 Brunswick Street');

A screenshot of a computer

Description automatically generated

Figure 2.1: inserting data into the customer\_details table

**Inserting the data into the table “order\_info”(local\_lab):**

(INSERT INTO Order\_info (user\_id, item\_name, quantity, order\_date)

VALUES (1, 'Laptop', 6, now());

INSERT INTO Order\_info (user\_id, item\_name, quantity, order\_date)

VALUES (2, 'Mobile', 10, now());

A screenshot of a computer

Description automatically generated

Figure 2.2: inserting data into the order\_info table

**Inserting the data into the table “Inventory” (remote\_lab):**

INSERT INTO Inventory (item\_id, item\_name, available\_quantity)

VALUES

(1, 'Laptop', 10),

(2, 'Phone', 15);

A screenshot of a computer

Description automatically generated

Figure 2.3: inserting data into the Inventory

**Step 3**

**Explanation about your program flow and what each component does.**

**Program flow**

Step 1 – Connection of the system with localdb and with remotedb.

Step 2 – It executes a query to retrieve data from the inventory table of remotedb and returns

the fields. It is coordinated with time and the time it takes is also shown on the

terminal.

Step 3 – The following step includes taking input for order placement from the users.

Step 4 – update the order\_info table in localdb by inserting new order information.

Step 5 – And now the final step is done which is to update the inventory table present in the

remoteDB.

**Order\_info table before and after order was placed (locally):**

**Before:**

A screenshot of a computer

Description automatically generated

Figure 3.1: order\_info table before order placed

**After:**

**A screenshot of a computer

Description automatically generated**

Figure 3.2: order\_info table after order placed

A computer screen shot of a black screen

Description automatically generated

Figure 3.3: Output of java code in CLI

**Inventory table before and after order was placed (remotely):**

**Before:**

A screenshot of a computer

Description automatically generated

Figure 3.4: Inventory table before order placed

**After:**

A screenshot of a computer

Description automatically generated

Figure 3.5: Inventory table after order placed

**A screenshot of a computer program

Description automatically generated**

Figure 3.6: Inventory table before order placed

**Step 4**

**Provide brief explanation about why there is a difference in the execution time for**

**performing operation on local database and on remote database.**

The time taken when performing operations on the local database rather than the remote database is mainly attributed by the system architecture and the latency resulting from connectivity.

In the case of handling local databases, all the transactions take place on the same computer where the database exists. As a result, data is computed and collected from within the local system resource thus making it run faster in most cases. Since no data traverses through the network, the issue of latency is not prominent and there is optimized speed for the operations to be conducted.

On the other hand, when working with a remote database, operations entail the client –server communication on a network with a database server. This only added more variables that may cause a direct impact on the Execution time; these include time needed to traverse through the network, time required to go through network bandwidth, and others because of possible congestion on the network. Information must be sent from the client to the server and back to the client, which can cause time lags that can be exasperating, particularly over long distances or ineffective networking.

Additionally, remote databases can be placed on third party, isolated hosts that may be accessible by anyone or on commercial providers’ data centers that can possess different performance characteristics. This can affect the general interaction with the database server and can lead to variability in time taken when compared to running the application locally from a separate access database.

In general, the role of the differences in the timing of local and remote Database Systems depends on how efficient the local resources are used in contrast with the considerations of time and resource overhead for the interaction with a network and a multi-tier server architecture in the case of a remote Database System.

**NOTE:**  
  
**In the Java code, I use a simple IntelliJ project, and I have also provided the MySQL Connector JAR file in the GitLab repository.**