CSCI-5408

DATA MANAGEMENT, WAREHOUSING, & ANALYTICS

LAB ASSIGNMENT - 4

Banner ID: B00977669

GitLab Assignment Link:
https://git.cs.dal.ca/saji/csci5408_w24_b00977669_christin_saji

Table of Contents

Task 1:

Set up a simple e-commerce database system with the following tables:

User table (in **Local** DB) with attributes: id, name, email, phone, address

Query –

```
CREATE TABLE User (
   id INT AUTO_INCREMENT PRIMARY KEY,
   name VARCHAR(20),
   email VARCHAR(30),
   phone VARCHAR(25),
   address VARCHAR(50)
);
```

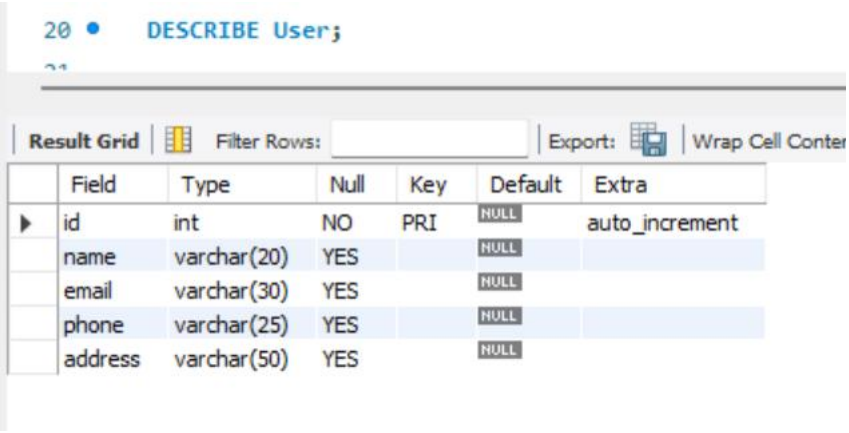


Figure 1 User: Database (local DB)

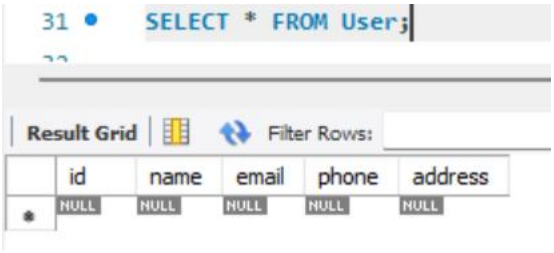


Figure 2 User Table with no data

Order_info table (in **Local** DB) with attributes: order_id, user_id, item_name, quantity, order_date

Query –

```
CREATE TABLE Order_info (
   order_id INT AUTO_INCREMENT PRIMARY KEY,
   user_id INT,
   item_name VARCHAR(30),
   quantity INT,
   order_date DATE
);
```

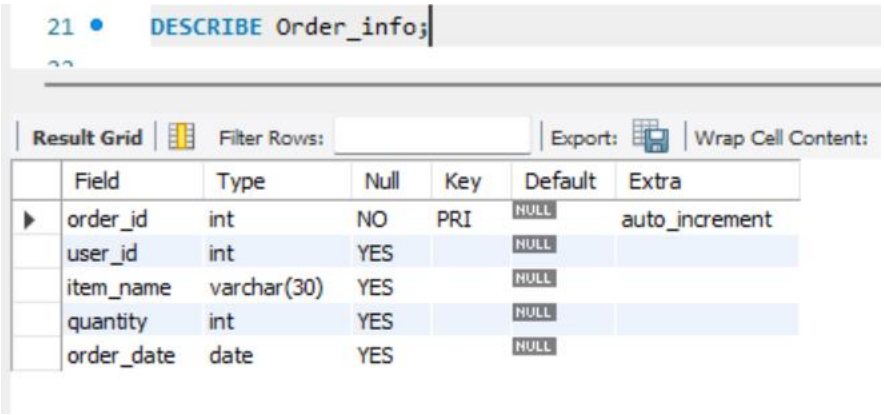


Figure 3 Order_info: Database (local DB)

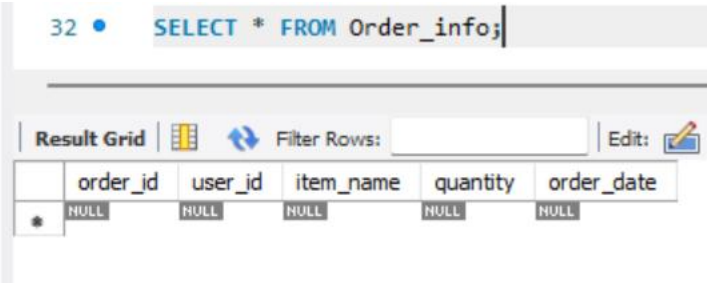


Figure 4 Order_info Table with no data

Inventory table (in remote **GCP** DB) with attributes: item_id, item_name, available_quantity

Query –

```
CREATE TABLE Inventory (
   item_id INT AUTO_INCREMENT PRIMARY KEY,
   item_name VARCHAR(30),
   available_quantity  INT
);
```

```
mysql> DESCRIBE Inventory;
+--------------------+-------------+------+-----+---------+----------------+
| Field              | Type        | Null | Key | Default | Extra          |
+--------------------+-------------+------+-----+---------+----------------+
| item_id            | int         | NO   | PRI | NULL    | auto_increment |
| item_name          | varchar(30) | YES  |     | NULL    |                |
| available_quantity | int         | YES  |     | NULL    |                |
+--------------------+-------------+------+-----+---------+----------------+
3 rows in set (0.21 sec)
```

Figure 5 Inventory: Database (remote DB)

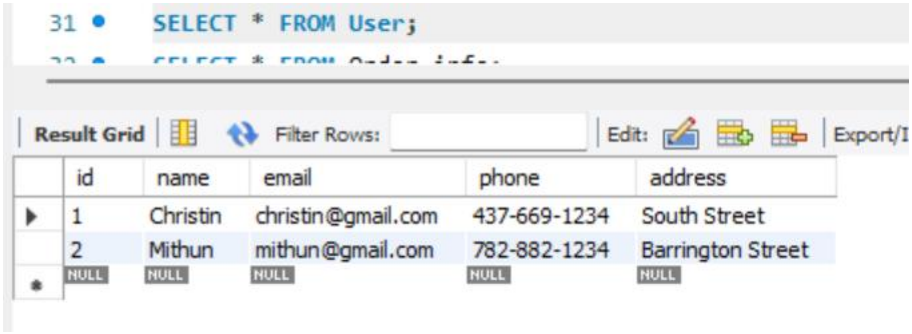
Task 2:

Insert some sample/dummy data in the above tables.

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

('Christin', 'christin@gmail.com', '437-669-1234', 'South Street'),

('Mithun', 'mithun@gmail.com', '782-882-1234', 'Barrington Street');



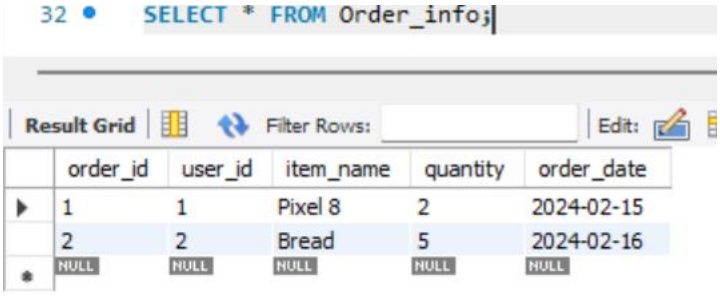
id	name	email	phone	address
1	Christin	christin@gmail.com	437-669-1234	South Street
2	Mithun	mithun@gmail.com	782-882-1234	Barrington Street
NULL	NULL	NULL	NULL	NULL

Figure 6 User Table with dummy data

INSERT INTO Order_info (user_id, item_name, quantity, order_date) VALUES

(1 , 'Pixel 8', 2, '2024-02-15'),

(2 , 'Bread', 5, '2024-02-16');



order_id	user_id	item_name	quantity	order_date
1	1	Pixel 8	2	2024-02-15
2	2	Bread	5	2024-02-16
NULL	NULL	NULL	NULL	NULL

Figure 7 Order_info Table with dummy data

INSERT INTO Inventory (item_name, available_quantity) VALUES

('Pixel 8', 48),

('Bread', 56),

('Lenovo Laptop', 27),

('iPhone 15', 39);

```
mysql> SELECT * FROM Inventory;
+---------+---------------+--------------------+
| item_id | item_name     | available_quantity |
+---------+---------------+--------------------+
|       1 | Pixel 8       |                 48 |
|       2 | Bread         |                 56 |
|       3 | Lenovo Laptop |                 27 |
|       4 | iPhone 15     |                 39 |
+---------+---------------+--------------------+
4 rows in set (0.21 sec)
```

Figure 8 Inventory Table with dummy data

Task 3:

Write a Java program that does the following:

A. Fetch and display all item details from the remote database table (Inventory table).
B. User enters details about item for which an order is to be placed.
C. Insert the order details into local database table (Order_info table).
D. Upon successful insertion, Update the item's quantity information in the remote database table (Inventory table).

Ques. Explanation about the program flow and what each component does.

Ans. Step 1: First, we initialize some variables to connect to the local and remote database using JDBC.

Step 2: Create a main method that executes three functions: 'displayInventory()' to display the inventory details from the remote database, 'userOrder()' to get information from the user for the order, and 'insertOrderDetails(order)' which takes the order object from the users and adds the order details to 'Order_info' table. If the order is inserted successfully, we then invoke 'updateInventory(order)' which updates the remote database based on the number of products ordered.

Step 3: 'displayInventory()' – Establish connection with the remote database. Execute the query to gather all the records from the 'Inventory' table. Iterate over the ResultSet to display each record from the table. Calculate execution time using 'System.currentTimeMillis()'.

Step 4: 'userOrder()' – A simple function to ask for the user's input to get the order details, and finally, store the input in an 'Order' object.

Step 5: 'insertOrderDetails(Order order)' – Establish connection with the local database. Prepare a SQL insertion operation on the 'Order_info' table. If the operation is successful, 'updateInventory(order)' is invoked. Calculate execution time.

Step 6: 'updateInventory(Order order)' – Establish connection with the remote database. Prepare a SQL update operation on the 'Inventory' table. Print the operation status accordingly. Finally, print the execution time for the operation.

Screenshots of the tables after the program execution & user selection.

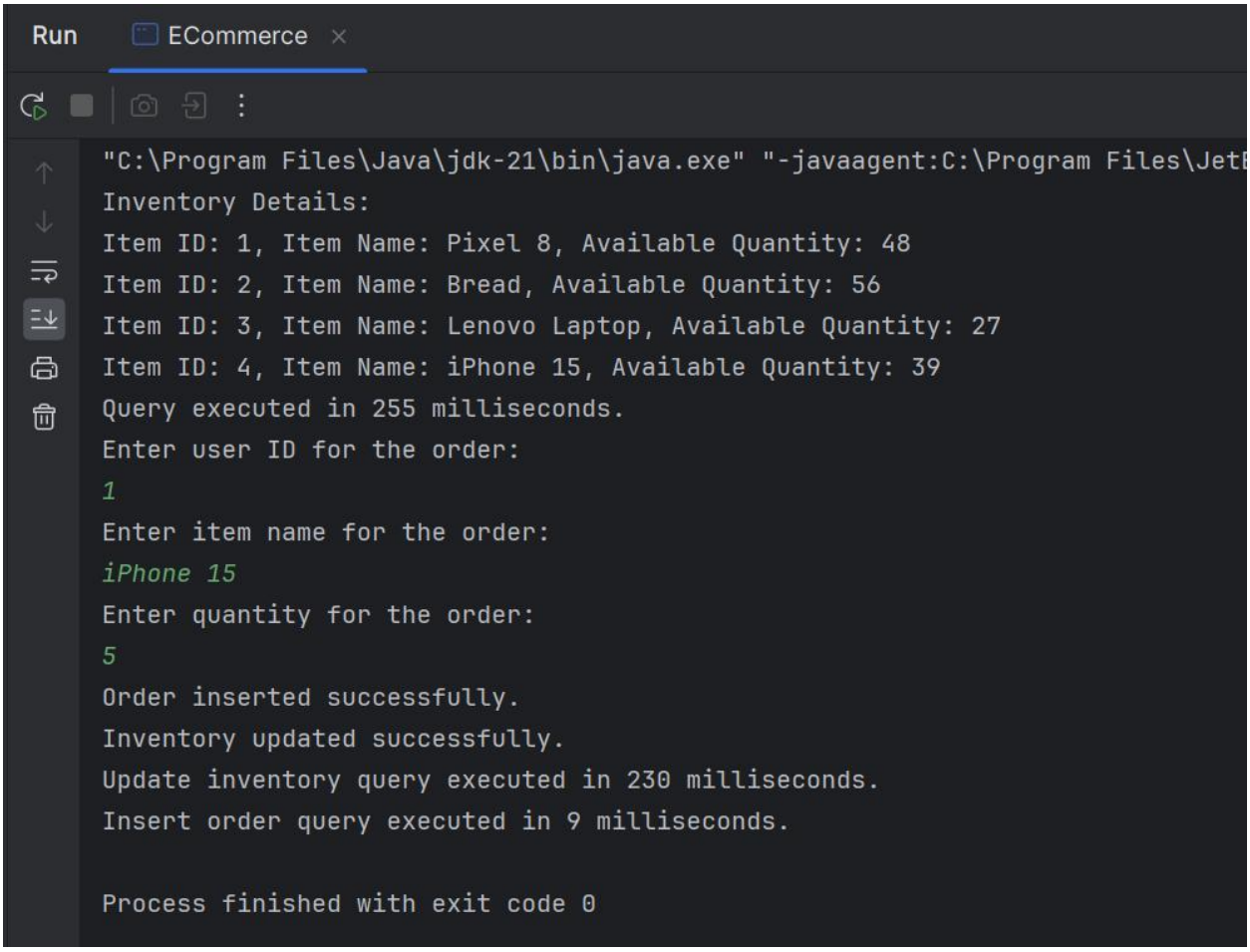


Figure 9 Java Program Output

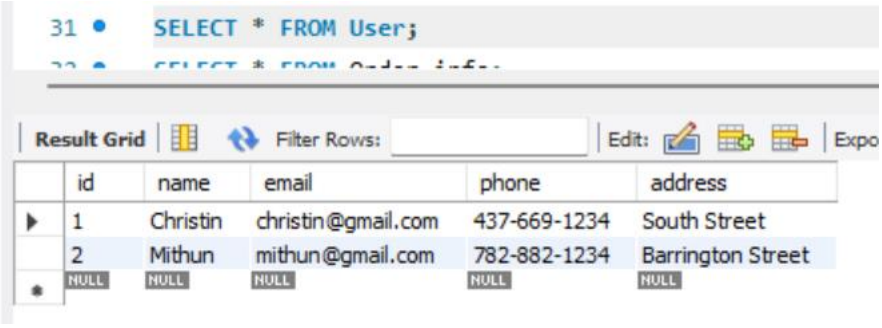


Figure 10 User Table after the program execution

```
32 •  SELECT * FROM Order_info;
```

Result Grid | Filter Rows: | Edit:

order_id	user_id	item_name	quantity	order_date
1	1	Pixel 8	2	2024-02-15
2	2	Bread	5	2024-02-16
3	1	iPhone 15	5	2024-02-17
NULL	NULL	NULL	NULL	NULL

Figure 11 Order_info Table after the program execution

```
mysql> SELECT * FROM Inventory;
+---------+---------------+--------------------+
| item_id | item_name     | available_quantity |
+---------+---------------+--------------------+
|       1 | Pixel 8       |                 48 |
|       2 | Bread         |                 56 |
|       3 | Lenovo Laptop |                 27 |
|       4 | iPhone 15     |                 34 |
+---------+---------------+--------------------+
4 rows in set (0.20 sec)
```

Figure 12 Inventory Table after the program execution

Task 4:

Calculate & display the SQL query execution time in milliseconds at STEPS-A, C & D.

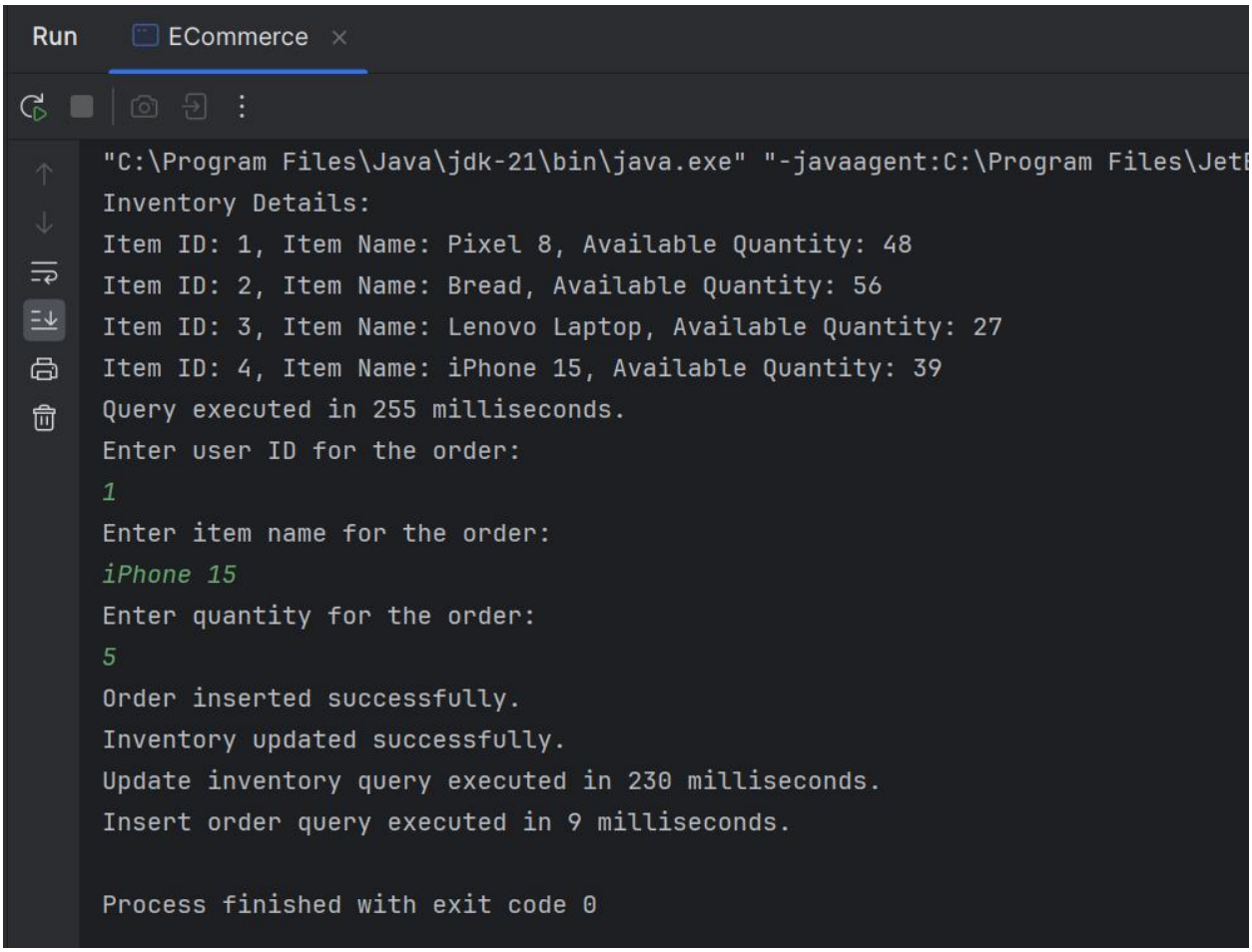


Figure 13 Java Program Output containing the query execution time in milliseconds

Ques. Provide brief explanation about why there is a difference in the execution time for performing operation on local database and on remote database.

Ans. Local databases are accessed on the same machine, which means there is less distance for data to travel. In contrast, remote databases are hosted in a cloud environment, such as Google Cloud Platform (GCP), and are in a specific geographic location. Because of this, data needs to travel over the internet, which can be affected by network latency. The closer the remote database is to the accessing machine, the lower the latency.