**Team 4 - Chapter 5 – Images and Photography Studio:**

**Creating and Manipulating a Relational Database for Team Project:**

1. Client (clientId, firstName, lastName), street, city, state, zip, areaCode, phoneNumber)
2. Meeting (clientId, meetDate, meetTime, repName)
3. Photographer (empId, firstName, lastName, street, city, state, zip, areaCode, number)
4. Booking (clientId, bookingDate, bookingTime, duration, type, empId1, empId2)
5. PackageMenu (packageNo, numWallet, albumType, albumPages, albumCover, price)
6. Job (contractNo, type, eventName, location, clientId , date, time, duration, cost, empId1, empId2, packageNoChosen, totalCost, amtPaid, amtDue)
7. Proof (contractNo, proofNo, quality)
8. Order (orderNo, dateOrdered, totalAmount, packageNoOrdered, contractNo)
9. OrderItem (orderNo, proofNo, size, quantity, dateDelivered)
10. Payment (contractNo, datePaid, payType, amount)
11. EmergencyContact(Clientid,eventid,Emergencyfirstname,Emergencylastname,Emergencyphonenumber)

**Step 5.1 – Review and update the data dictionary and list of assumptions (as needed).**

1. Client Table:

- clientId (Primary Key) - INTEGER

- firstName - VARCHAR(50)

- lastName - VARCHAR(50)

- street - VARCHAR(100)

- city - VARCHAR(50)

- state - VARCHAR(50)

- zip - VARCHAR(10)

- areaCode - VARCHAR(10)

- phoneNumber - VARCHAR(15)

2. Meeting Table:

- clientId (Foreign Key) - INTEGER

- meetDate - DATE

- meetTime - TIMESTAMP

- repName - VARCHAR(100)

3. Photographer Table:

- empId (Primary Key) - INTEGER

- firstName - VARCHAR(50)

- lastName - VARCHAR(50)

- street - VARCHAR(100)

- city - VARCHAR(50)

- state - VARCHAR(50)

- zip - VARCHAR(10)

- areaCode - VARCHAR(10)

- phonenumber - VARCHAR(15)

4. Booking Table:

- clientId (Foreign Key) - INTEGER

- bookingDate - DATE

- bookingTime - TIME

- duration - INTEGER

- type - VARCHAR(50)

- empId1 (Foreign Key) - INTEGER

- empId2 (Foreign Key) - INTEGER

5. PackageMenu Table:

- packageNo (Primary Key) - INTEGER

- numWallet - INTEGER

- albumType - VARCHAR(50)

- albumPages - INTEGER

- albumCover - VARCHAR(50)

- price - DECIMAL(10, 2)

6. Job Table:

- contractNo (Primary Key) - INTEGER

- type - VARCHAR(50)

- eventName - VARCHAR(100)

- location - VARCHAR(100)

- clientId (Foreign Key) - INTEGER

- date - DATE

- time - TIME

- duration - INTEGER

- cost - DECIMAL(10, 2)

- empId1 (Foreign Key) - INTEGER

- empId2 (Foreign Key) - INTEGER

- packageNoChosen (Foreign Key) - INTEGER

- totalCost - DECIMAL(10, 2)

- amtPaid - DECIMAL(10, 2)

- amtDue - DECIMAL(10, 2)

7. Proof Table:

- contractNo (Foreign Key) - INTEGER

- proofNo – INTEGER- Primarykey

- quality - VARCHAR(50)

8. Order Table:

- orderNo (Primary Key) - INTEGER

- dateOrdered - DATE

- totalAmount - DECIMAL(10, 2)

- packageNoOrdered (Foreign Key) - INTEGER

- contractNo (Foreign Key) - INTEGER

9. OrderItem Table:

- orderNo (Foreign Key) - INTEGER

- proofNo (Foreign Key) - INTEGER

- size - VARCHAR(50)

- quantity - INTEGER

- dateDelivered - DATE

10. Payment Table:

- contractNo (Foreign Key) - INTEGER

- datePaid - DATE

- payType - VARCHAR(50)

- amount - DECIMAL(10, 2)

11. Emergency Contact Table:

- Clientid (Foreign Key) - INTEGER

- eventid - INTEGER

- Emergencyfirstname - VARCHAR(50)

- Emergencylastname - VARCHAR(50)

- Emergencyphonenumber - VARCHAR(15)

Constraints:

- Primary Key: Denotes the primary key attribute for each table.

- Foreign Key: Denotes foreign key attributes, referencing other tables.

- VARCHAR(x): Denotes variable-length character data types with a maximum length of 'x'.

- INTEGER: Denotes integer data types.

- DECIMAL(x, y): Denotes decimal numbers with 'x' total digits and 'y' decimal places.

- DATE: Denotes date data types.

- TIME: Denotes time data types.

**Step 5.2 – Design SQL statements to create all tables needed to implement the design. Then create the tables in the database.**

**Tables Created: Client, Meeting, Photographer, Booking, PackageMenu, Job, Proof, Orders, OrderItem, Payment, EmergencyContact.**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

****

**A screenshot of a computer program

Description automatically generated**

**A screen shot of a computer

Description automatically generated**

**A screenshot of a computer program

Description automatically generated**

**A screenshot of a computer program

Description automatically generated**

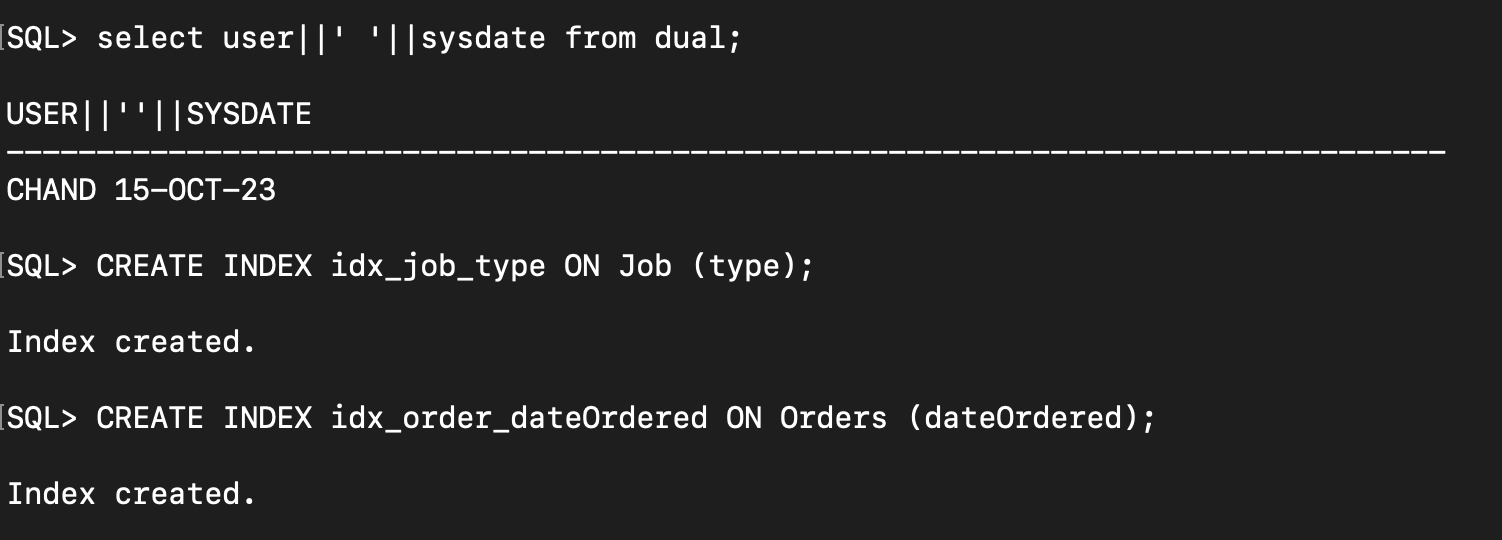
**A screen shot of a computer

Description automatically generated**

**A screen shot of a computer

Description automatically generated**

**Step 5.3 – Design SQL statements to create indexes for foreign keys and for any other columns that will be used most often for queries. Then execute the SQL statements in the database.**

****

**A screen shot of a black and white screen

Description automatically generated**

**Step 5.4 – Design SQL statements to insert at least five records in each table, preserving all constraints.** **Then insert the records into the tables.**

1. **Inserting records into Client Table:**

A screenshot of a computer

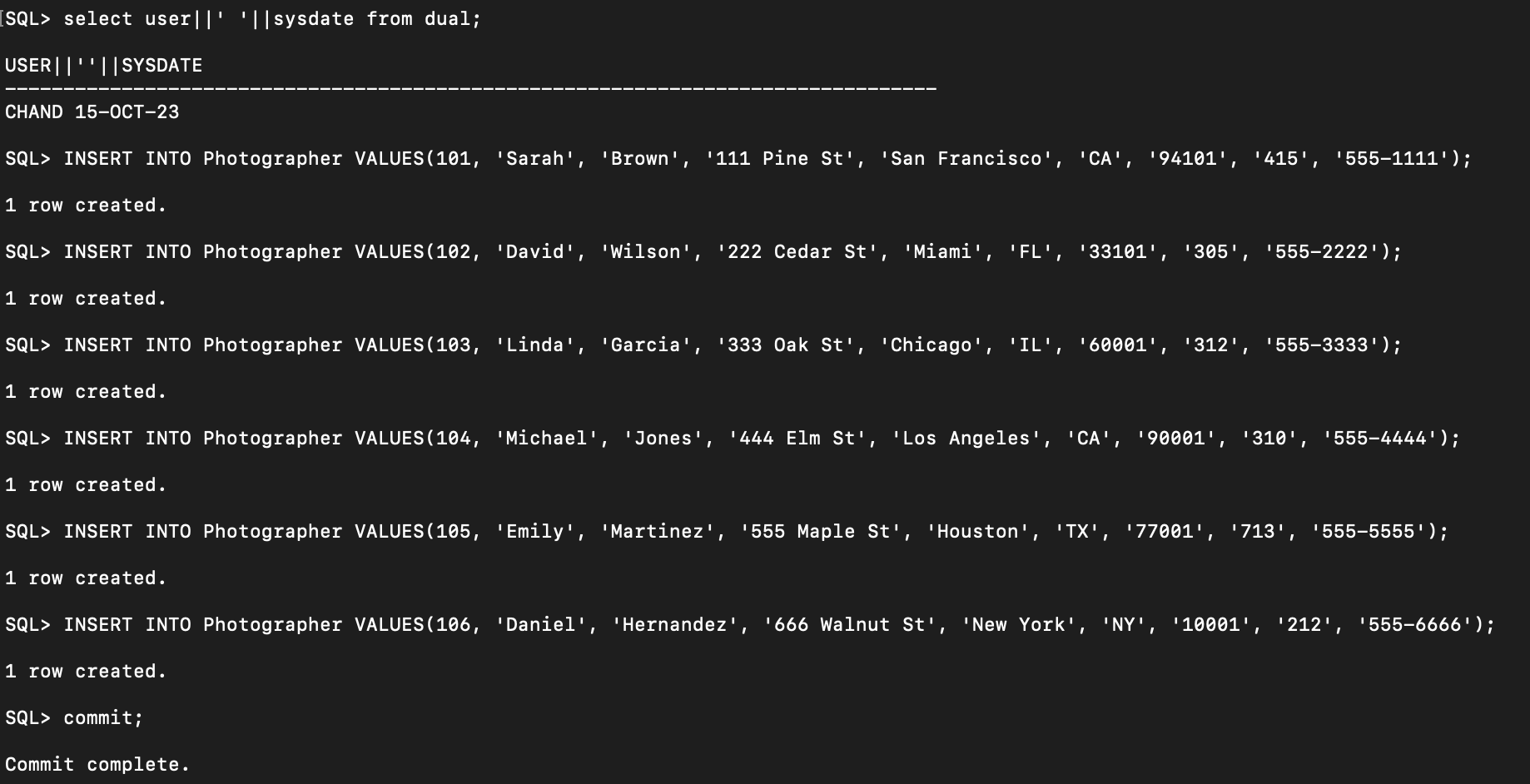
Description automatically generated

1. **Inserting records into Meeting:**

**A screenshot of a computer

Description automatically generated**

1. **Inserting records into Photographer Table:**

****

1. **Inserting records into Booking Table:**

**A screenshot of a computer

Description automatically generated**

1. **Inserting records into PackageMenu Table:**

**A screenshot of a computer program

Description automatically generated**

1. **Inserting records into Job Table:**

**A screenshot of a computer

Description automatically generated**

1. **Inserting records into Proof Table:**

**A screenshot of a computer program

Description automatically generated**

1. **Inserting records into Orders Table:**

**A screenshot of a computer

Description automatically generated**

1. **Inserting records into OrderItem:**

**A screenshot of a computer program

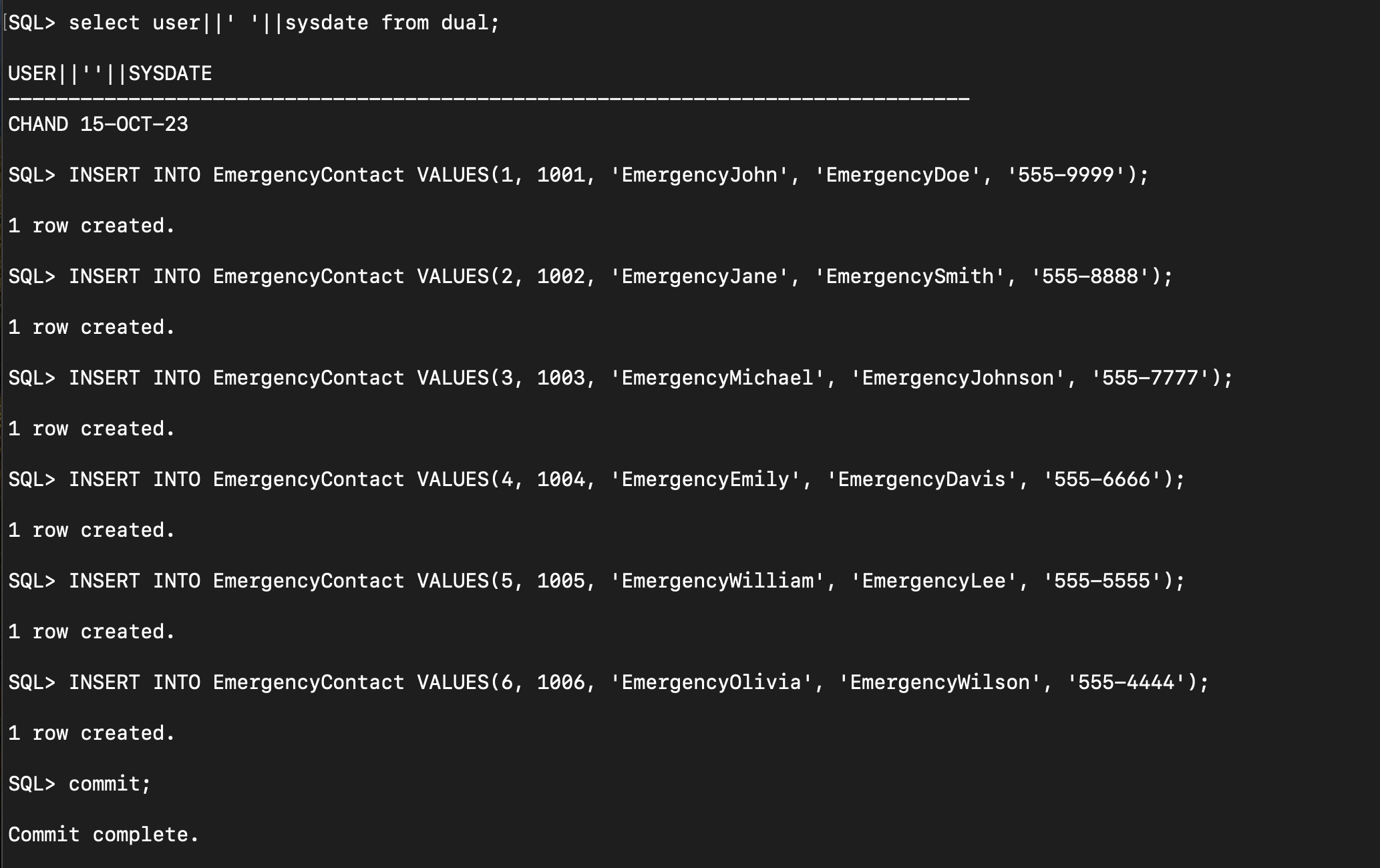
Description automatically generated**

1. **Inserting records into Payment:**

**A screenshot of a computer program

Description automatically generated**

1. **Inserting records into EmergencyContact:**

****

**Step 5.5 – Design SQL statements that will process five non-routine requests for information from the database. Then execute the SQL statements in the database.** Show your work by providing screenshots of executing the SQL statements in the database along with the results.

1. **List Clients with Outstanding Payments: To retrieve a list of clients with outstanding payments (where amtDue is greater than 0):**

**Sql> SELECT C.clientId, C.firstName, C.lastName, J.amtDue FROM Client C JOIN Job J ON C.clientId = J.clientId WHERE J.amtDue > 0;**

**A screenshot of a computer

Description automatically generated**

1. **Find the Most Popular Album Type:**

**Sql> SELECT albumType, COUNT(\*) AS popularity FROM PackageMenu JOIN Job ON PackageMenu.packageNo = Job.packageNoChosen GROUP BY albumType ORDER BY COUNT(\*) DESC FETCH FIRST 1 ROWS ONLY;**

**A black screen with white text

Description automatically generated**

1. **Retrieve Total Revenue by Photographer:**

**Sql> SELECT Photographer.empId, Photographer.firstName, Photographer.lastName, SUM(Job.totalCost) AS totalRevenue FROM Photographer LEFT JOIN Job ON Photographer.empId = Job.empId1 OR Photographer.empId = Job.empId2 GROUP BY Photographer.empId, Photographer.firstName, Photographer.lastName**

**A screenshot of a computer

Description automatically generated**

1. **List Clients Who Haven't Booked a Job:**

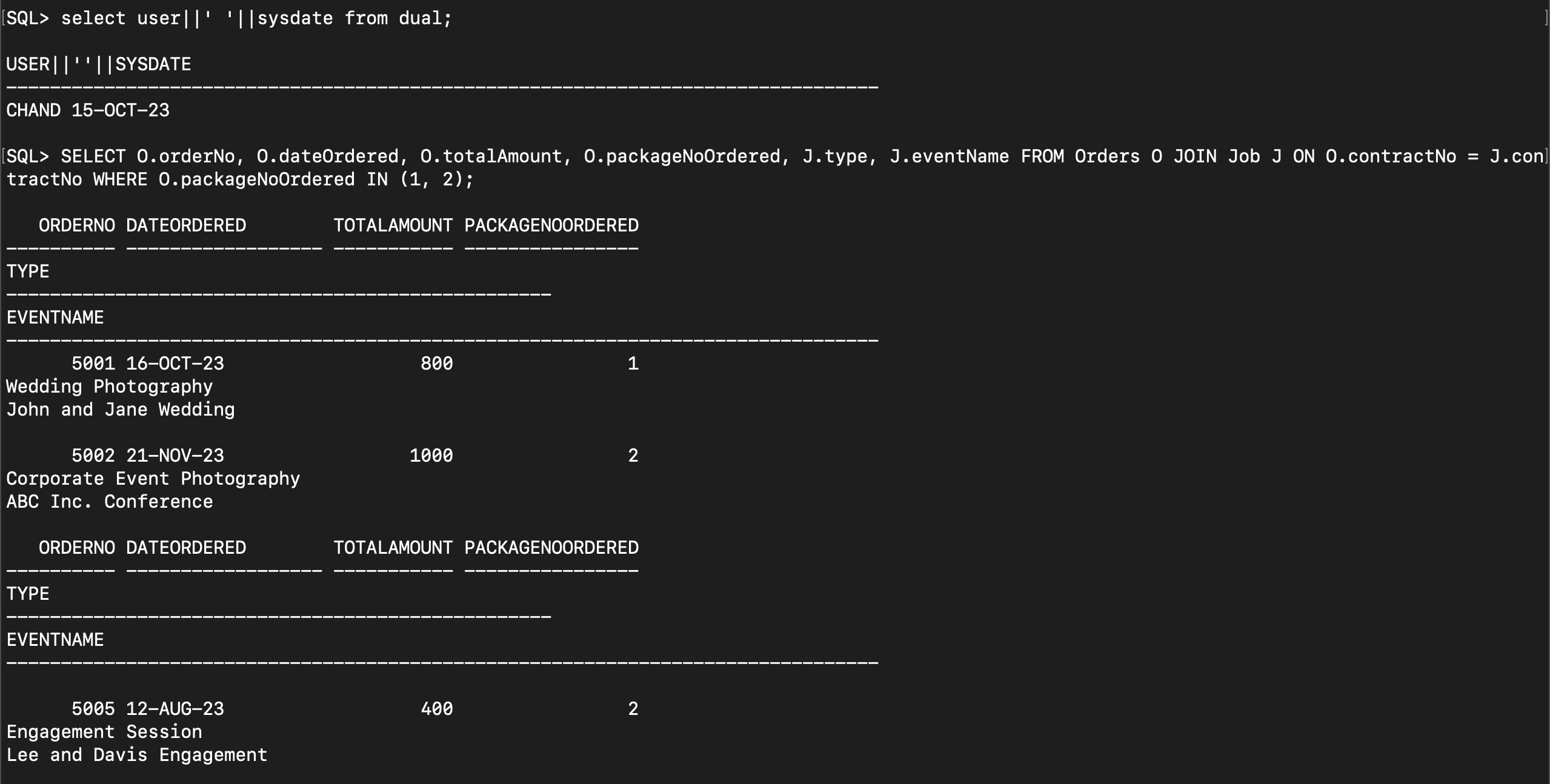
**Sql> SELECT clientId, firstName, lastName FROM Client WHERE clientId NOT IN (SELECT DISTINCT clientId FROM Job);**

**A black screen with white text

Description automatically generated**

1. **Retrieve Details of Orders with Specific Package Numbers:**

**Sql> SELECT O.orderNo, O.dateOrdered, O.totalAmount, O.packageNoOrdered, J.type, J.eventName FROM Orders O JOIN Job J ON O.contractNo = J.contractNo WHERE O.packageNoOrdered IN (1, 2);**

****

**Step 5.6 – Design one trigger for your project. Then create the trigger in the database.**

1. **Create the orders1 table:**

**A screen shot of a computer

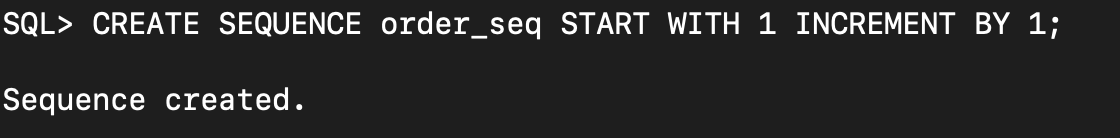
Description automatically generated**

1. **Create the orders\_total table:**

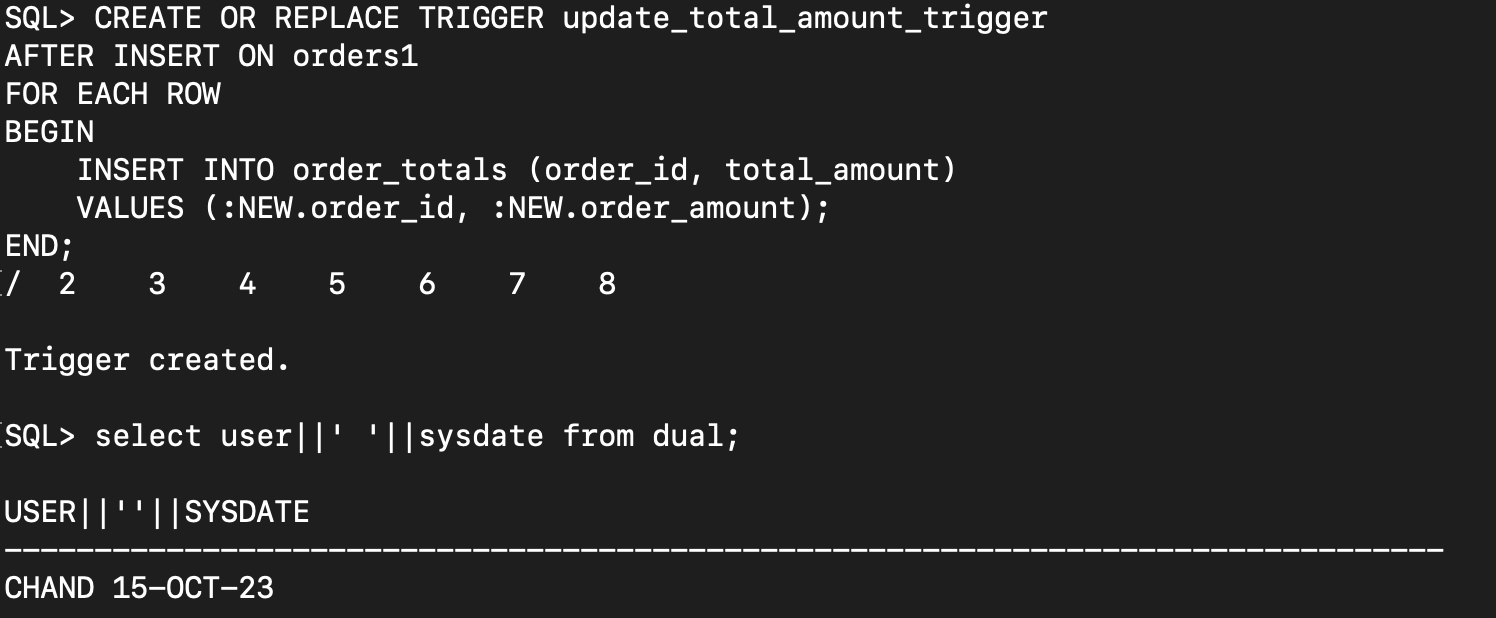
**A screen shot of a computer code

Description automatically generated**

1. **Creating Sequence:**

****

1. **Creating a trigger to update the total\_amount in order\_totals when a new order is inserted:**

****

**Step 5.7 – Design and execute SQL statements to demonstrate that the trigger is working as expected.**

A screenshot of a computer

Description automatically generated