Introduction to Databases CA- Beauty Retailer

Catherine McIlroy, Student No 23173190

Table of Contents

Introduction

This report will detail the design, creation, population and manipulation of a relational database for an online beauty retailer.

The company is a US based small business, with a team of 20 employees. The objective of the business is to provide a diverse selection of products for sale which cover a wide price range, thereby appealing to customers from a variety of economic backgrounds. The company aims to ship all orders within a maximum of 3 days after transaction completion, thereby providing a reliable and efficient service to its customers.

The product categories include make-up, skincare, body care and haircare. These include products from budget, mid-range and luxury brands, with price points ranging from $6-$95.

When signing up for an account or placing an order, customers must provide an email and contact telephone number. Customer contact details are held in the business database, and these are used to notify existing customers of new product listings, sales, or discounts.

The database will also track inventory levels, allowing the company to efficiently manage stock and ensure products are available for purchase. This real-time inventory management system helps to minimise backorders and optimise the customer shopping experience, with the aim of enhancing overall customer satisfaction and loyalty.

Entities

This section covers Part 1.2 and Part 1.3 of the CA.

The below tables display the distinct entities of the business along with their attributes, attribute data types, primary keys and foreign keys (if applicable).

All tables are presented in 3rd normal form.

products

product_id (**PK**)	varchar (20)
product_description	varchar (200)
unit_price	double
category	enum (Skin, Body, Hair, Make-up)
supplier_id (**FK** references suppliers.supplier_id)	varchar (20)
qty_on_hand	int

transactions

transaction_id (**PK**)	varchar (20)
transaction_date	datetime
order_id (**FK** references orders.order_id)	varchar (20)
payment_method	enum (Mastercard, Visa, Paypal)
transaction_status	enum (Completed, Cancelled)
employee_id (**FK** references employees.employee_id)	varchar (20)

customers

customer_id (**PK**)	varchar (20)
first_name	text (50)
last_name	text (50)
address_id (**FK** references addresses.address_id)	varchar (20)
customer_email	varchar (200)
customer_phone	varchar (20)

employees

employee_id (**PK**)	varchar (20)
first_name	text (50)
last_name	text (50)
employee_email	varchar (200)
employee_phone	varchar (20)
contract_type	enum (Full-time, Part-time)
salary	int

suppliers

supplier_id (**PK**)
supplier_name
account_no
supplier_email
supplier_phone

orders

order_id (**PK**)
customer_id (**FK** references customers.customer_id)
created_at
total_price
fulfilled_at
shipped_at

order_items

order_item_id (**PK**)
order_id (**FK** references orders.order_id)
product_id (**FK** references products.product_id)
product_qty

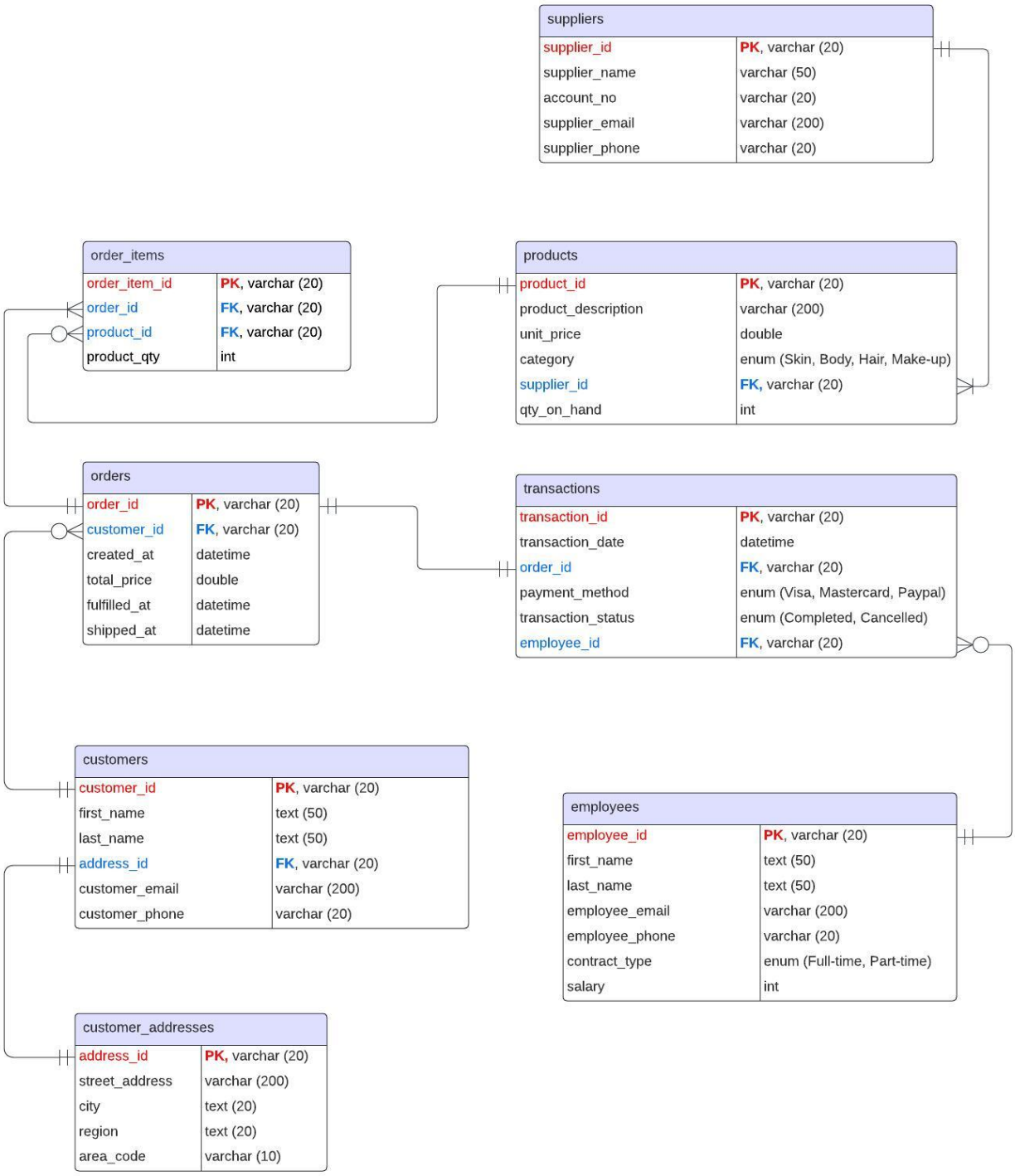
customer_addresses

address_id (**PK**)
street_address
city
region
area_code

Entity Relationship Diagram (ERD)

This section covers Part 1.4 and Part 1.5 of the CA.

The below ERD displays the database tables and the relationships between them, and details multiplicities using crow's feet notation.



The following relationships and cardinalities can be observed from the above diagram:

- **customers to customer_addresses**
  - linked on address_id, which is the primary key in the customer_addresses table and a foreign key in the customers table.
  - each record in the customer_addresses table must be associated with one record in the customers table, and vice versa (i.e. each address must be associated with one customer, and each customer must be associated with one address).
- **customers to orders**
  - linked on customer_id, which is the primary key in the customers table and a foreign key in the orders table.
  - each record in the orders table must be associated with one record in the customers table (i.e. an order must be placed by one customer).
  - each record in the customers table can be associated with zero to many records in the orders table (i.e. each customer can place 0+ orders).
- **orders to transactions**
  - linked on order_id, which is the primary key in the orders table and a foreign key in the transactions table.
  - each record in the orders table must be associated with one record in the transactions table, and vice versa (i.e. each order must be associated with the corresponding transaction, and each transaction must be associated with one order).

- **transactions to employees**
  - linked on employee_id, which is the primary key in the employees table and a foreign key in the transactions table.
  - each record in the transactions table must be associated with one record in the employees table (i.e. each transaction must be carried out by one employee).
  - each record in the employees table can be associated with zero to many records in the transactions table (i.e. each employee can carry out 0+ transactions).
- **orders to order_items**
  - linked on order_id, which is the primary key in the orders table and a foreign key in the order_items table.
  - each record in the orders table can be associated with one or more records in the order_items table (i.e. each order must contain a minimum of 1 item, but can contain many).
  - each record in the order_items table must be associated with one record in the orders table (i.e. each item must be part of one order).
- **order_items to products**
  - linked on product_id, which is the primary key in the products table and a foreign key in the orders table.
  - each record in the products table can be associated with zero to many records in the order_items table (i.e. a listed product might not be included in any orders, or it may be included in many).
  - each record in the order_items table must be associated with one record in the products table (i.e. the item must be one of the listed products).

- **products to suppliers**
  - linked on supplier_id, which is the primary key in the suppliers table and a foreign key in the products table.
  - each record in the products table must be associated with one record in the suppliers table (i.e. each listed product must have one supplier).
  - each record in the suppliers table can be associated with one or many records in the products table (i.e. each supplier must be associated with a minimum of one product listing, but can be the supplier for multiple products).

Database Creation (Data Definition Language)

The database detailed in the previous sections “Entities” and “Entity Relationship Diagram (ERD)” was then created using the following DDL command.

```sql
CREATE DATABASE beauty_retailer;
```

Table Creation (Data Definition Language)

Following creation of the database, the next step involved creation of the previously specified tables. This was carried out using DDL commands as shown below.

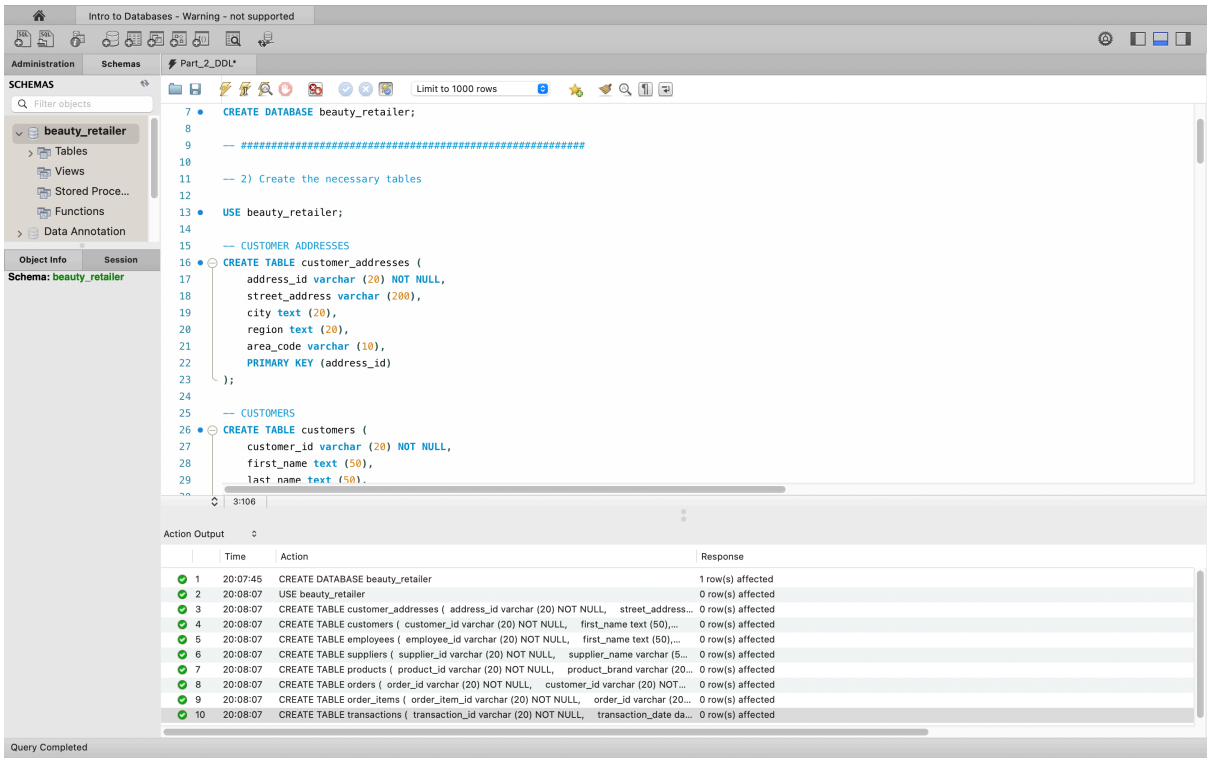
```sql
USE beauty_retailer;


CREATE TABLE customer_addresses (
        address_id varchar (20) NOT NULL,
        street_address varchar (200),
        city text (20),
        region text (20),
        area_code varchar (10),
        PRIMARY KEY (address_id)
);


CREATE TABLE customers (
        customer_id varchar (20) NOT NULL,
        first_name text (50),
        last_name text (50),
        address_id varchar (20),
        customer_email varchar (200),
        customer_phone varchar (20),
        PRIMARY KEY (customer_id),
        FOREIGN KEY (address_id) REFERENCES customer_addresses(address_id)
);
```

```
CREATE TABLE employees (
        employee_id varchar (20) NOT NULL,
        first_name text (50),
        last_name text (50),
        employee_email varchar (200),
        employee_phone varchar (20),
        contract_type enum ("Full-time", "Part-time"),
        salary int,
        PRIMARY KEY (employee_id)
);

CREATE TABLE suppliers (
        supplier_id varchar (20) NOT NULL,
        supplier_name varchar (50),
        account_no varchar (50),
        supplier_email varchar (200),
        supplier_phone varchar (20),
        PRIMARY KEY (supplier_id)
);
```

```
CREATE TABLE products (
        product_id varchar (20) NOT NULL,
        product_description varchar (200),
        unit_price double,
        category enum ("Skin", "Body", "Hair", "Make-up"),
        supplier_id varchar (20) NOT NULL,
        qty_on_hand int,
        PRIMARY KEY (product_id),
        FOREIGN KEY (supplier_id) REFERENCES suppliers(supplier_id)
);

CREATE TABLE orders (
        order_id varchar (20) NOT NULL,
        customer_id varchar (20) NOT NULL,
        created_at datetime,
        total_price double,
        fulfilled_at datetime,
        shipped_at datetime,
        PRIMARY KEY (order_id),
        FOREIGN KEY (customer_id) REFERENCES customers(customer_id)
);
```

```
CREATE TABLE order_items (
        order_item_id varchar (20) NOT NULL,
        order_id varchar (20),
        product_id varchar (20),
        product_qty int,
        PRIMARY KEY (order_item_id),
        FOREIGN KEY (order_id) REFERENCES orders(order_id),
        FOREIGN KEY (product_id) REFERENCES products(product_id)
);

CREATE TABLE transactions (
        transaction_id varchar (20) NOT NULL,
        transaction_date datetime,
        order_id varchar (20),
        payment_method enum ("Visa", "Mastercard", "Paypal"),
        transaction_status enum ("Completed", "Cancelled"),
        employee_id varchar (20),
        PRIMARY KEY (transaction_id),
        FOREIGN KEY (employee_id) REFERENCES employees(employee_id),
        FOREIGN KEY (order_id) REFERENCES orders(order_id)
);
```

These commands executed successfully, resulting in creation of all tables in accordance with the design detailed in the previous sections. A screenshot of the action output is included below.



Database Population (Data Manipulation Language)

This section covers both Part 2.3 and Part 2.4 of the CA.

The tables created in the previous section “Table Creation (Data Definition Language)” were each populated with a minimum of five rows of data using DML insert statements (Part 2.3). These statements are shown below.

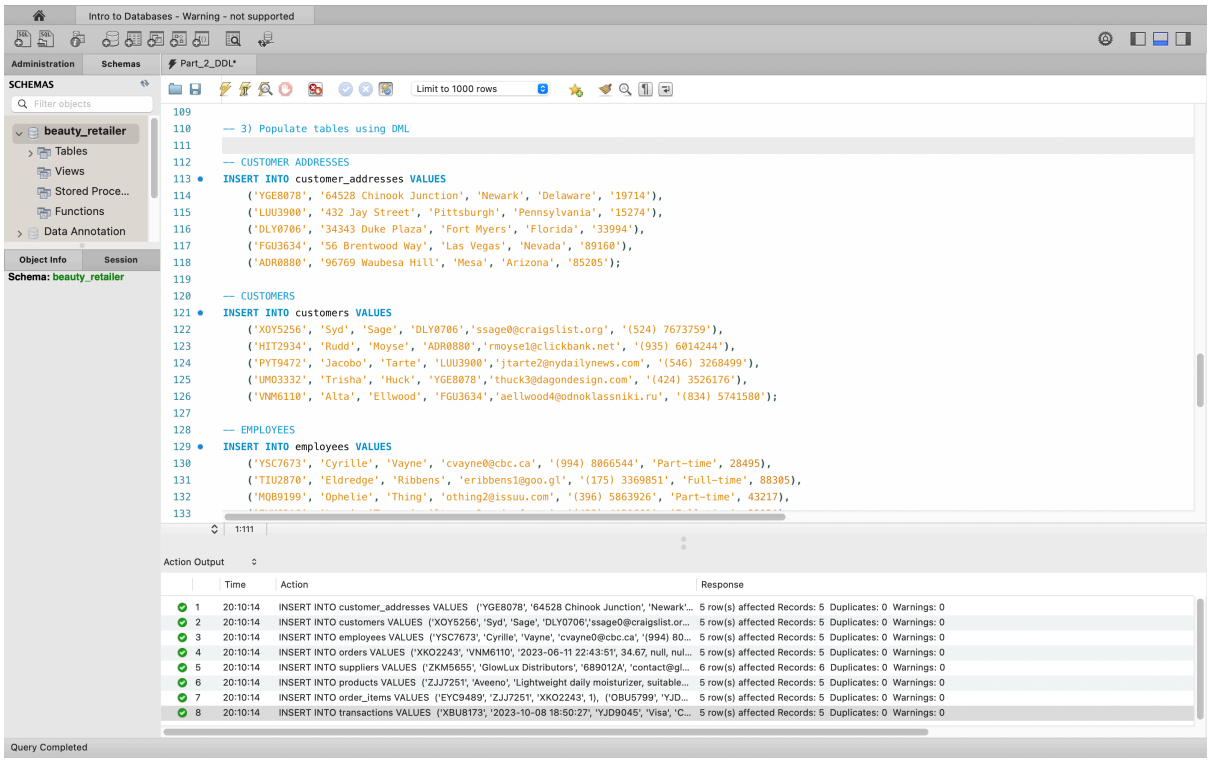
Note: The following values were created using a combination of www.mockaroo.com and ChatGPT 3.5. All data is fictional with the exception of the products(product_description) values, which refer to real products.

```sql
INSERT INTO customer_addresses VALUES
    ('YGE8078', '64528 Chinook Junction', 'Newark', 'Delaware', '19714'),
    ('LUU3900', '432 Jay Street', 'Pittsburgh', 'Pennsylvania', '15274'),
    ('DLY0706', '34343 Duke Plaza', 'Fort Myers', 'Florida', '33994'),
    ('FGU3634', '56 Brentwood Way', 'Las Vegas', 'Nevada', '89160'),
    ('ADR0880', '96769 Waubesa Hill', 'Mesa', 'Arizona', '85205');

INSERT INTO customers VALUES
    ('XOY5256', 'Syd', 'Sage', 'DLY0706','ssage0@craigslist.org', '(524) 7673759'),
    ('HIT2934', 'Rudd', 'Moyse', 'ADR0880','rmoyse1@clickbank.net', '(935) 6014244'),
    ('PYT9472', 'Jacobo', 'Tarte', 'LUU3900','jtarte2@nydailynews.com', '(546) 3268499'),
    ('UMO3332', 'Trisha', 'Huck', 'YGE8078','thuck3@dagondesign.com', '(424) 3526176'),
    ('VNM6110', 'Alta', 'Ellwood', 'FGU3634','aellwood4@odnoklassniki.ru', '(834) 5741580');
```

```
INSERT INTO employees VALUES

        ('YSC7673', 'Cyrille', 'Vayne', 'cvayne0@cbc.ca', '(994) 8066544', 'Part-time',
28495),

    ('TIU2870', 'Eldredge', 'Ribbens', 'eribbens1@goo.gl', '(175) 3369851', 'Full-time',
88305),

    ('MQB9199', 'Ophelie', 'Thing', 'othing2@issuu.com', '(396) 5863926', 'Part-time',
43217),

    ('EHM6216', 'Levy', 'Turney', 'lturney3@unicef.org', '(435) 4120669', 'Full-time',
32921),

    ('KYM5242', 'Reinwald', 'Senner', 'rsenner4@technorati.com', '(269) 6407233',
'Full-time', 61666);


INSERT INTO orders VALUES

        ('XKO2243', 'VNM6110', '2023-06-11 22:43:51', 34.67, null, null),

    ('YCZ8807', 'UMO3332', '2023-09-25 06:02:35', 88.26, '2023-09-25 09:33:57',
'2023-09-25 15:56:39'),

    ('ZJJ7251', 'XOY5256', '2023-02-10 19:27:05', 11.25, '2023-02-11 09:05:32',
'2023-02-11 16:14:36'),

    ('HBE5030', 'HIT2934', '2023-11-19 07:29:58', 9.94, '2023-11-19 11:48:17',
'2023-11-19 16:32:56'),

    ('YJD9045', 'PYT9472', '2023-10-08 18:48:32', 31.24, '2023-10-09 10:02:11',
'2023-10-09 16:48:17');


INSERT INTO suppliers VALUES

        ('ZKM5655', 'GlowLux Distributors', '689012A',
'contact@glowluxdistributors.com', '(432) 4429887'),

        ('JYN8318', 'Dermalogica Wholesale', '24412533',
'wholesale@dermalogica.com', '(318) 9427212'),

        ('BRU1439', 'Beauty Source Distributors', 'TW27882',
'info@beautysourcedistributors.com', '(976) 5385727'),

        ('LVB7431', 'Nuxe Wholesale', 'A51875393', 'wholesale@nuxe.com', '(795)
2492220'),

        ('ZWP4499', 'Beauty Hub', '24329', 'info@beautyhub.com', '(323) 5587896'),

        ('AAL4559', 'Glamour Goods Ltd.', '60MJ9659', 'sales@glamourgoods.com',
'(328) 3581860');
```

```
INSERT INTO products VALUES

        ('ZJJ7251', 'Aveeno Daily Moisturizing Lotion', 9.80, 'Body', 'BRU1439', 24),

        ('YCZ8807', 'Dermalogica Special Cleansing Gel', 31.24, 'Skin', 'JYN8318', 8),

        ('HBE5030', 'Nuxe Huile Prodigieuse Multi-Purpose Dry Oil', 17.33, 'Body',
'LVB7431', 14),

        ('YJD9045', 'Maybelline Fit Me Matte + Poreless Foundation', 9.94, 'Make-up',
'AAL4559', 22),

        ('XKO2243', 'Redken Extreme Anti-Snap Leave-In Treatment', 11.25, 'Hair',
'ZWP4499', 17);


INSERT INTO order_items VALUES

        ('EYC9489', 'ZJJ7251', 'XKO2243', 1),

        ('OBU5799', 'YJD9045', 'YCZ8807', 1),

        ('EON3113', 'XKO2243', 'HBE5030', 2),

        ('UIG0823', 'YCZ8807', 'ZJJ7251', 9),

        ('NNO9305', 'HBE5030', 'YJD9045', 1);


INSERT INTO transactions VALUES

        ('XBU8173', '2023-10-08 18:50:27', 'YJD9045', 'Visa', 'Completed',
'MQB9199'),

    ('FUG7223', '2023-11-19 07:33:14', 'HBE5030', 'Visa', 'Completed', 'MQB9199'),

    ('MMV2192', '2023-06-11 22:45:19', 'XKO2243', 'Paypal', 'Cancelled', 'YSC7673'),

    ('QDW2915', '2023-02-10 19:31:02', 'ZJJ7251', 'Mastercard', 'Completed',
'KYM5242'),

    ('LIV8153', '2023-09-25 06:05:49', 'YCZ8807', 'Paypal', 'Completed', 'TIU2870');


```

These commands executed successfully, resulting in population of all tables with the data included in the insert statements. A screenshot of the action output is included below.



The database was then further populated with a larger dataset by using the MySQL Table Data Import Wizard to import CSV files (or JSON in the case of the products table, as the Import Wizard could not read this in CSV format).

The corresponding tables and JSON code may be found in the Appendix section.

Select Statement with Join

This section answers Part 3.1 of the CA.

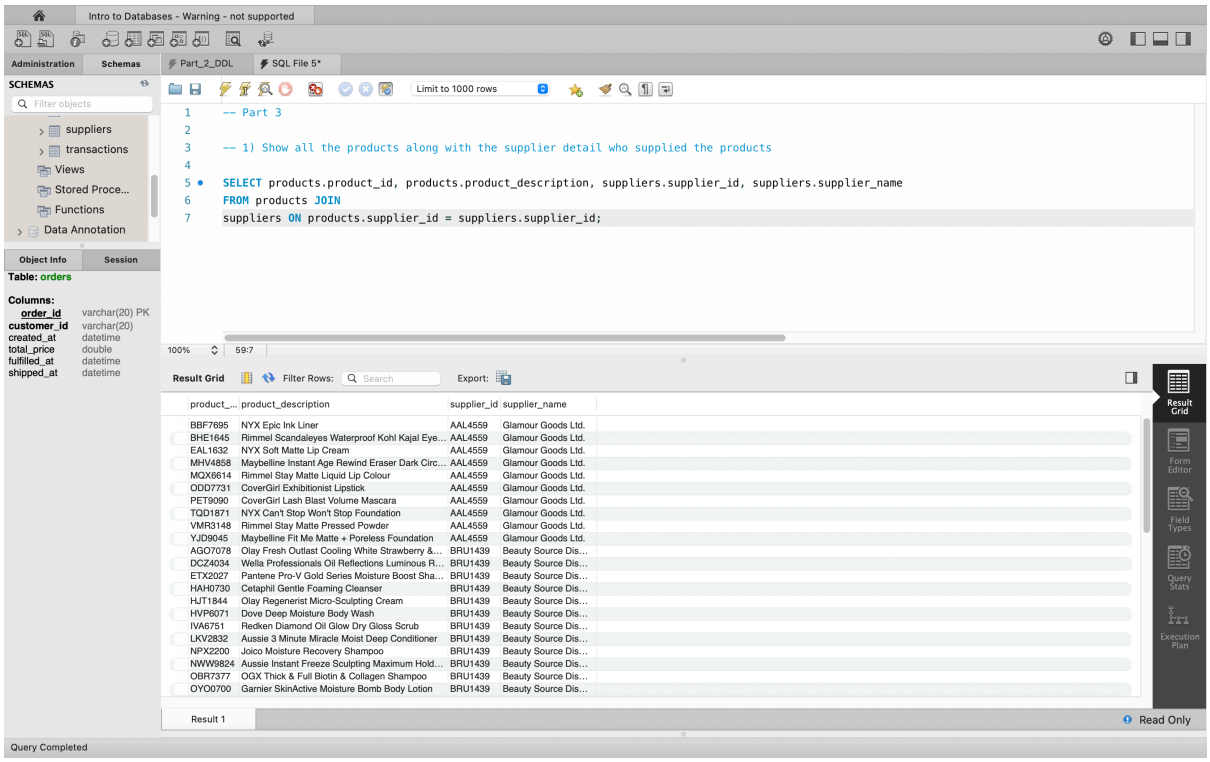
The objective was to display all of the products, along with details of the supplier for each product. In order to achieve this, I have used a SELECT statement with a JOIN.

I have joined the products table with the suppliers table using the supplier_id attribute which is present in both tables (primary key in the suppliers table and foreign key in the products table).

This query will display the product_id and product_description from the products table, and the supplier_id and supplier_name from the suppliers table, thereby fulfilling the task instructions.

The SQL query and a screenshot of the action output are included below.

```sql
SELECT products.product_id, products.product_description, suppliers.supplier_id, suppliers.supplier_name

FROM products JOIN

suppliers ON products.supplier_id = suppliers.supplier_id;
```



Stored Procedure

This section answers Part 3.2 of the CA.

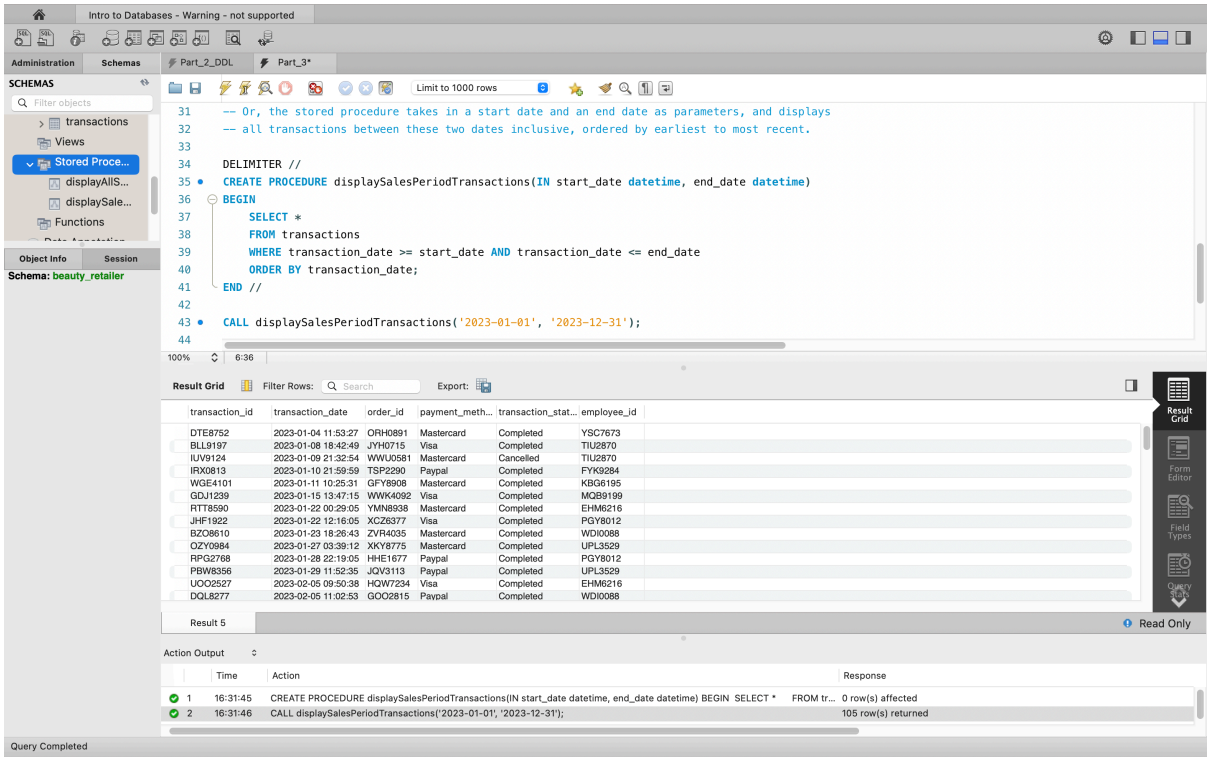
This question can potentially be interpreted in two slightly different ways. The objective was to create a stored procedure which takes the start and end dates for sales and displays all transactions between these dates.

I would interpret this as meaning a stored procedure which takes in two parameters from the user, a start date (datetime) and an end date (datetime). The procedure would contain a SELECT statement which would display all records from the transactions table where the transaction_date is greater than or equal to the start date, and less than or equal to the end date. For convenience, these records are ordered from earliest to most recent transaction_date.

The SQL code for this can be seen below.

```sql
DELIMITER //

CREATE PROCEDURE displaySalesPeriodTransactions(IN start_date datetime, end_date datetime)

BEGIN

    SELECT *

    FROM transactions

    WHERE transaction_date >= start_date AND transaction_date <= end_date

    ORDER BY transaction_date;

END //


CALL displaySalesPeriodTransactions('2023-01-01', '2023-12-31');
```

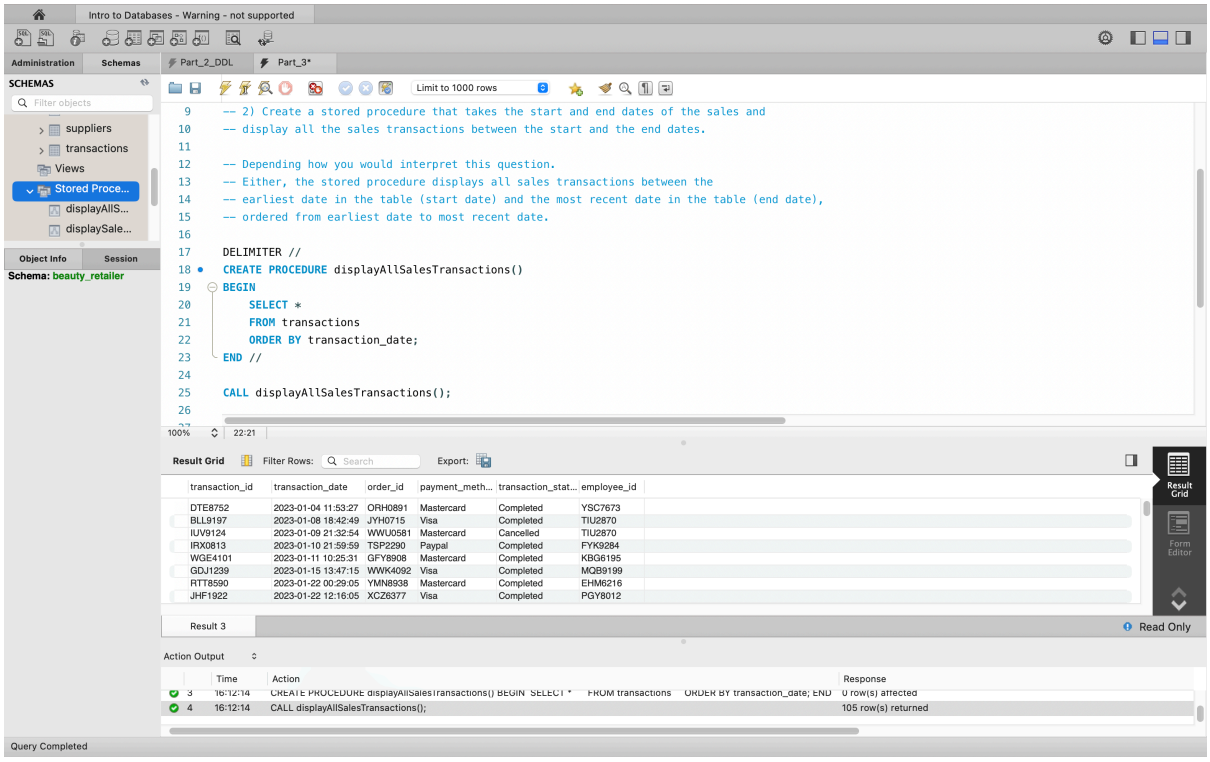
The action output from this code execution can be seen below.



The second way in which this question can be interpreted is to create a stored procedure which includes a SELECT statement to display all records from the transactions table, ordered from earliest to most recent transaction_date. This will effectively display all sales from the start date (earliest transaction_date) to the end date (most recent transaction_date).

The SQL code and a screenshot of the corresponding action output for this can be seen below.

```sql
DELIMITER //

CREATE PROCEDURE displayAllSalesTransactions()

BEGIN

    SELECT *

    FROM transactions

    ORDER BY transaction_date;

END //


CALL displayAllSalesTransactions();
```



View

This section answers Part 3.3 of the CA.

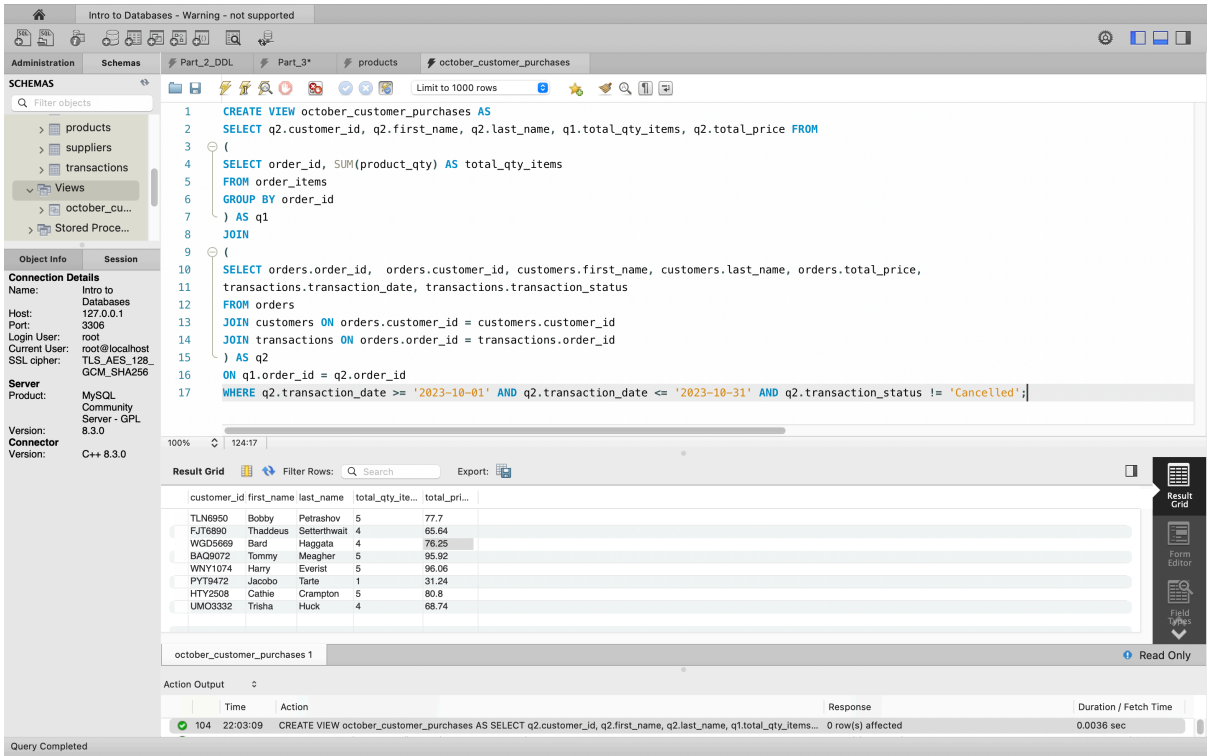
The objective of this task was to create a view displaying customers who made purchases in October 2023, the total number of items purchased by each of these customers in this month, and the total accumulated price of these purchases.

In order to do this, I used nested SELECT and JOIN statements.

Inside one statement I used the SUM function and GROUP BY statement to calculate the total quantity of items associated with each unique order_id in the order_items table. Inside the second statement I extracted the relevant data from the orders, customers and transactions tables.

I then joined the results of these queries in an outer SELECT statement, and filtered the results to display completed transactions which took place in October 2023 only.

The SQL code for this can be seen below, along with a screenshot of the action output.



```sql

CREATE VIEW october_customer_purchases AS

SELECT q2.customer_id, q2.first_name, q2.last_name, q1.total_qty_items,
q2.total_price FROM

(

SELECT order_id, SUM(product_qty) AS total_qty_items

FROM order_items

GROUP BY order_id

) AS q1

JOIN

(

SELECT orders.order_id,  orders.customer_id, customers.first_name,
customers.last_name, orders.total_price, transactions.transaction_date,
transactions.transaction_status

FROM orders

JOIN customers ON orders.customer_id = customers.customer_id

JOIN transactions ON orders.order_id = transactions.order_id

) AS q2

ON q1.order_id = q2.order_id

WHERE q2.transaction_date >= '2023-10-01' AND q2.transaction_date <=
'2023-10-31' AND q2.transaction_status != 'Cancelled';

```

Trigger (Stock Adjustment)

This section answers Part 3.4 of the CA.

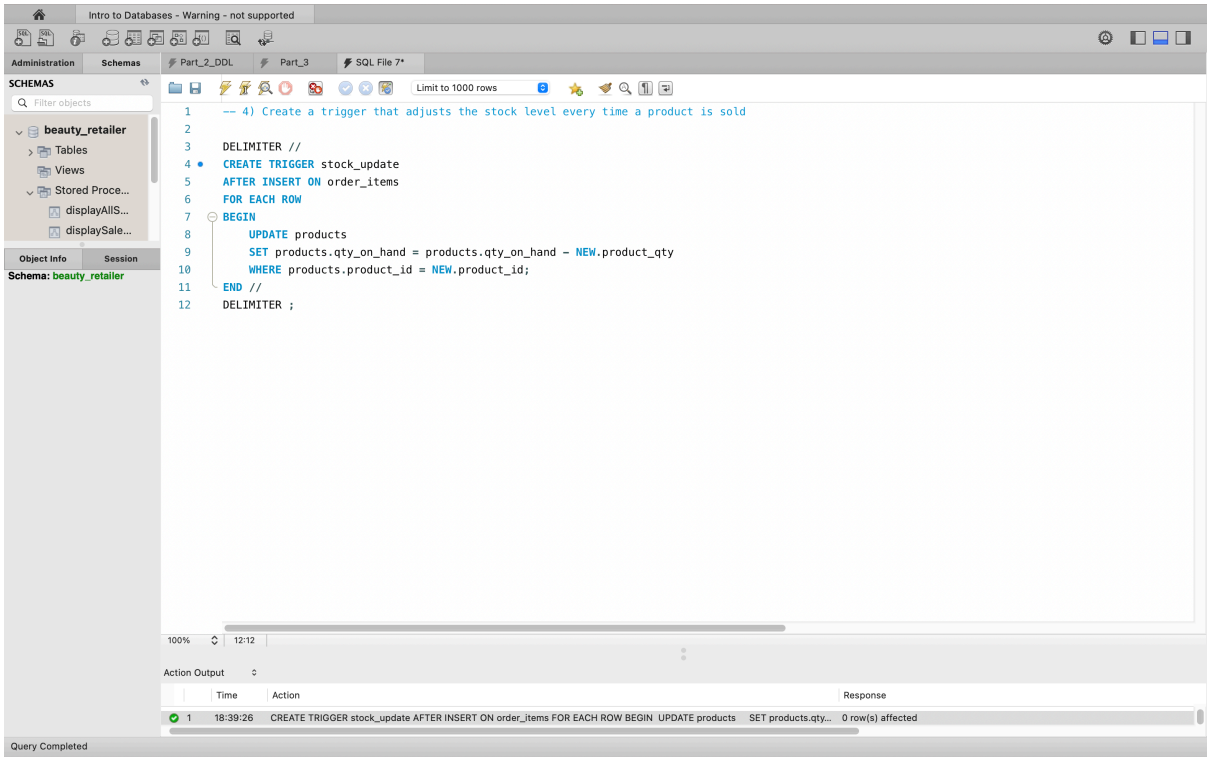
The objective of this task was to create a trigger which will adjust the associated quantity on hand whenever a product is sold. This will keep stock inventory levels correct and up to date.

This was achieved by creating a trigger which will update the products table after a new record is added to the order_items table. The product_qty value of the new order_items record (NEW.product_qty) will be subtracted from the existing qty_on_hand in the products table for the record with the corresponding product_id.

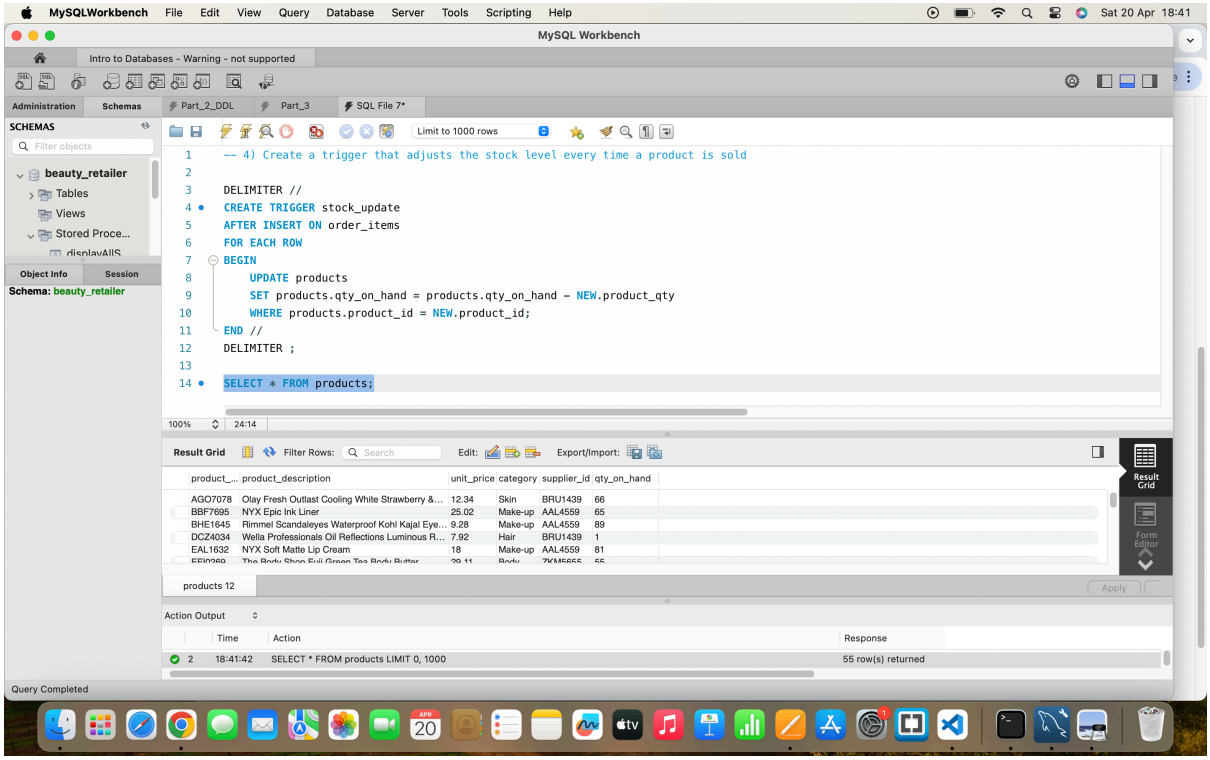
This SQL statement is shown below.

```sql
DELIMITER //
CREATE TRIGGER stock_update
AFTER INSERT ON order_items
FOR EACH ROW
BEGIN
    UPDATE products
    SET products.qty_on_hand = products.qty_on_hand - NEW.product_qty
    WHERE products.product_id = NEW.product_id;
END //
DELIMITER ;
```

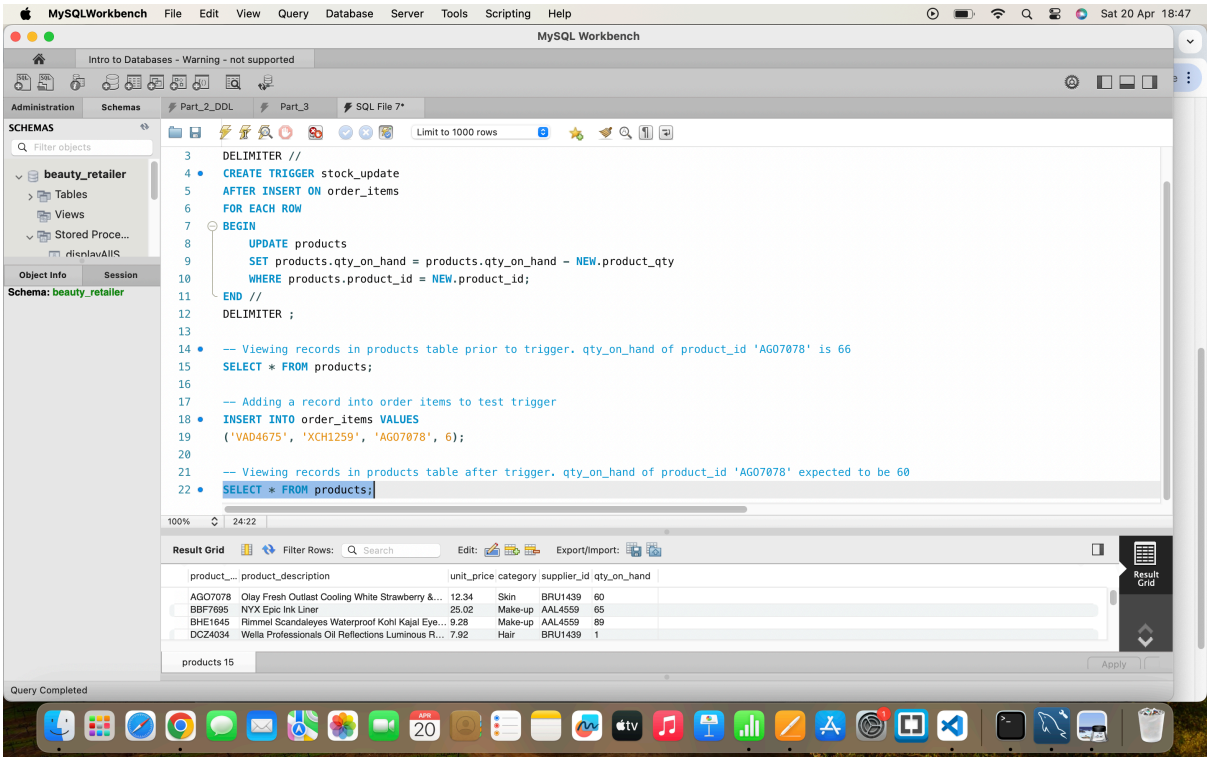
Included below is a screenshot of the action output after executing the above statement, and two screenshots testing the trigger to ensure it is working properly.



1) Action output after code execution



2) Displaying qty_on_hand levels before creation of new order_items record (pre-trigger execution). Note qty_on_hand of product_id AGO7078 is currently 66.



3) Displaying qty_on_hand levels following creation of new order_items record (post-trigger execution). Note qty_on_hand of product_id AGO7078 is now 60.

Annual Sales Report (Group By with Roll-up)

This section answers Part 3.5 of the CA.

The objective of this task was to create an annual sales report for the business for the year 2023. This report needed to showcase the total number of products sold, and the total accumulated price of these products, for each calendar month. I have also included an annual total value to display the total of these attributes for the entire year.

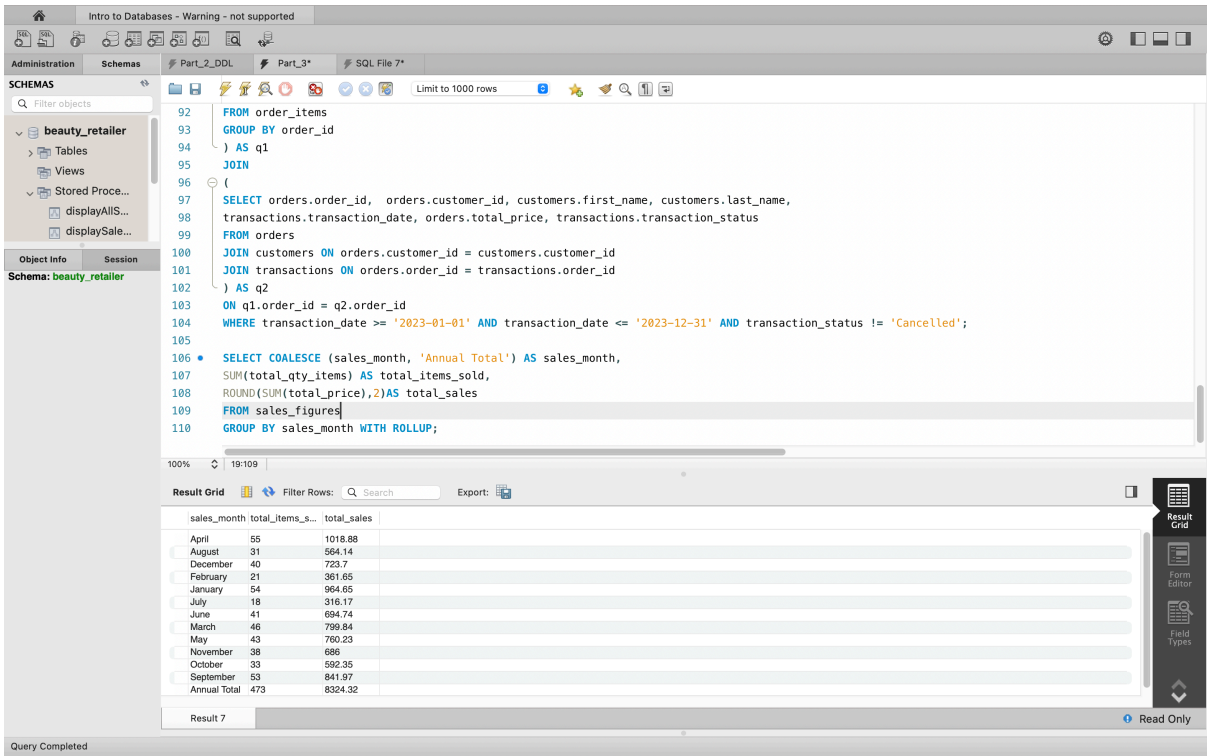
In order to do this, I first created a temporary table which would hold the relevant information needed for the query. The SQL code for this statement is shown below.

```sql
CREATE TEMPORARY TABLE sales_figures

SELECT q1.order_id, MONTHNAME(q2.transaction_date) as sales_month,
q1.total_qty_items, q2.total_price FROM

(

SELECT order_id, SUM(product_qty) AS total_qty_items

FROM order_items

GROUP BY order_id

) AS q1

JOIN

(

SELECT orders.order_id,  orders.customer_id, customers.first_name,
customers.last_name, transactions.transaction_date, orders.total_price,
transactions.transaction_status

FROM orders

JOIN customers ON orders.customer_id = customers.customer_id

JOIN transactions ON orders.order_id = transactions.order_id

) AS q2

ON q1.order_id = q2.order_id

WHERE transaction_date >= '2023-01-01' AND transaction_date <= '2023-12-31'
AND transaction_status != 'Cancelled';
```

I then used the SUM and ROUND functions (inside a SELECT statement) to calculate the total items sold and the total price rounded to two decimal places, and grouped these results by calendar month. I used a ROLLUP to include the annual total.

The SQL code for this procedure is included below, along with a screenshot of the output.

```sql
SELECT COALESCE (sales_month, 'Annual Total') AS sales_month,
SUM(total_qty_items) AS total_items_sold,
ROUND(SUM(total_price),2)AS total_sales
FROM sales_figures
GROUP BY sales_month WITH ROLLUP;
```



Percentage Sales Growth

This section answers Part 3.6 of the CA.

The objective of this task was to display the percentage sales growth of the business from the first month of opening to the most recent month. Since this database includes data relating to the year 2023 only, the months used for this calculation are January 2023 and December 2023.

The formula for calculating percentage sales growth is as follows:

Sales Growth = ((Most Recent Month Total Sales - 1st Month Total Sales) / 1st Month Total Sales) x 100

In order to calculate this, I need a query which will perform three operations:

1) Calculate the difference between the total sales figure for December 2023 and the total sales figure for January 2023.

2) Divide this answer by the total sales figure for January 2023.

3) Multiply the previous answer by 100.

I was not able to write a single query which would perform all three of these tasks. I engineered a workaround by creating a series of three temporary tables, followed by the final SELECT statement.

The first temporary table (sales_jan_dec) holds the total sales figures for January and December 2023 respectively, and draws information from the previous temporary sales_figures table (see previous section, “Annual Sales Report (Group By with Roll-up)”). The code for this can be seen below.

```sql
CREATE TEMPORARY TABLE sales_jan_dec

SELECT * FROM

(SELECT sales_month, ROUND(SUM(total_price),2) AS monthly_sales

FROM sales_figures

GROUP BY sales_month

) AS q1

WHERE sales_month = 'January' OR sales_month = 'December';
```

The second temporary table (sales_growth) draws data from the first temporary table (sales_jan_dec) and uses the LAG function to calculate the difference between the December 2023 and January 2023 total sales figures. It holds this value in the sales_difference column. The corresponding SQL code is displayed below.

```sql
CREATE TEMPORARY TABLE sales_growth

SELECT sales_month, monthly_sales,

monthly_sales-LAG(monthly_sales) OVER (ORDER BY sales_month desc) AS sales_difference

FROM sales_jan_dec;
```

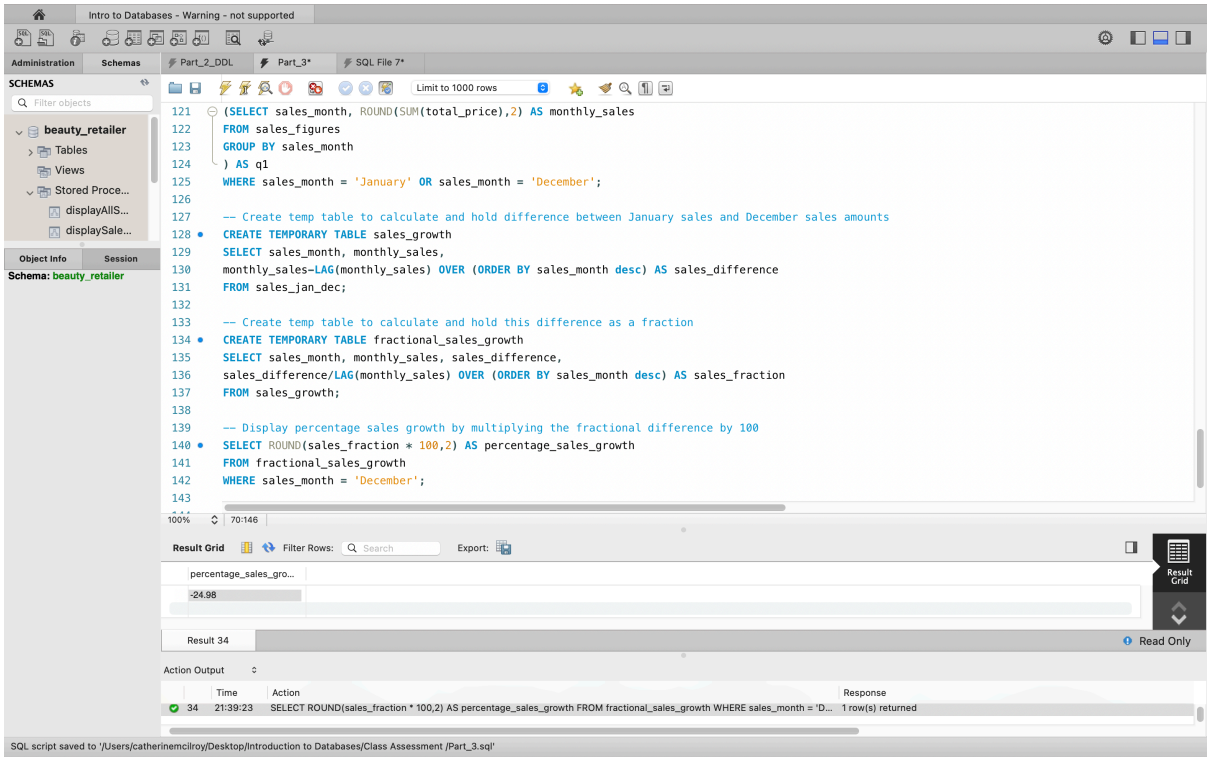
The third temporary table (fractional_sales_growth) draws data from the second temporary table (sales_growth) and once again uses the LAG function to divide the difference between the December 2023 and January 2023 total sales figures, by the January 2023 total sales figures. This results in a fractional sales growth value which is held in the sales_fraction column. The corresponding SQL code is displayed below.

```sql
CREATE TEMPORARY TABLE fractional_sales_growth

SELECT sales_month, monthly_sales, sales_difference,

sales_difference/LAG(monthly_sales) OVER (ORDER BY sales_month desc) AS sales_fraction

FROM sales_growth;
```

This sales_fraction value can now be extracted from the third temporary table (fractional_sales_growth), and multiplied by 100 within a SELECT statement to display the percentage sales growth figure. This value has been rounded to

two decimal figures using the ROUND function, for clarity. The code for this query is included below, along with a screenshot of the output.

```sql
SELECT ROUND(sales_fraction * 100,2) AS percentage_sales_growth
FROM fractional_sales_growth
WHERE sales_month = 'December';
```



From this output, it can be seen that the sales growth in this case is negative, with a value of -24.98%. This indicates that total sales were significantly lower in December 2023 than they were in January 2023.

Delete Statement

This section answers Part 3.7 of the CA.

The objective of this task was to write a query which would delete all customers from the database who have never made a purchase.

In order to do this, I have created a temporary table called customers_qty_orders, which will hold the customer_id, and will calculate and hold the total number of fulfilled orders associated with that customer (total_orders).

The total number of fulfilled orders is calculated using a CASE… ELSE expression. If the fulfilled_at field is null (indicating that the order was never completed) then the total_orders value remains unchanged, otherwise (fulfilled_at field is not null, indicating that the purchase was completed) the total_orders value is incremented by 1. These results are grouped by customer_id so that each total_orders value is associated with a specific customer.

The SQL code for this is shown below.

```sql
CREATE TEMPORARY TABLE customers_qty_orders

SELECT customer_id,

    SUM(CASE

        WHEN fulfilled_at IS NULL THEN 0

        ELSE 1

        END)

    AS total_orders

    FROM orders

    GROUP BY customer_id;
```

Following creation of this temporary table, a LEFT JOIN statement is used inside the DELETE statement to join the customers table with the newly created customers_qty_orders temporary table.

Records in the customers table are deleted only where customers_qty_orders(total_orders) value is null, indicating that the customer has never made a purchase.

The SQL code for this statement is shown below. No screenshot is included, since this query was not executed.

```sql
DELETE customers FROM customers

LEFT JOIN customers_qty_orders ON customers.customer_id =
customers_qty_orders.customer_id

WHERE customers_qty_orders.total_orders IS NULL;
```

Python Pandas and Graphs

This section answers Part 3.8 of the CA.

The objective of this task was to use a reporting tool to connect to the database and create a graphical representation of the percentage sales growth calculated in Part 3.6 (“Percentage Sales Growth”, page 29).

I have chosen to use Python with the Pandas and Matplotlib libraries, since I am familiar with these. The full Python script can be found in the Appendix section.

I first established a connection to the database using the Python mysql.connector module. I then ran the following query:

```python
query = """SELECT q1.order_id, MONTHNAME(q2.transaction_date) 
as sales_month, q1.total_qty_items, q2.total_price FROM 
(
SELECT order_id, SUM(product_qty) AS total_qty_items
FROM order_items
GROUP BY order_id
) AS q1
JOIN
(
SELECT orders.order_id,  orders.customer_id, 
customers.first_name, customers.last_name, 
transactions.transaction_date, orders.total_price, 
transactions.transaction_status
FROM orders
JOIN customers ON orders.customer_id = customers.customer_id
JOIN transactions ON orders.order_id = transactions.order_id
) AS q2
ON q1.order_id = q2.order_id
WHERE transaction_date >= '2023-01-01' AND transaction_date <= 
'2023-12-31' AND transaction_status != 'Cancelled';"""
```

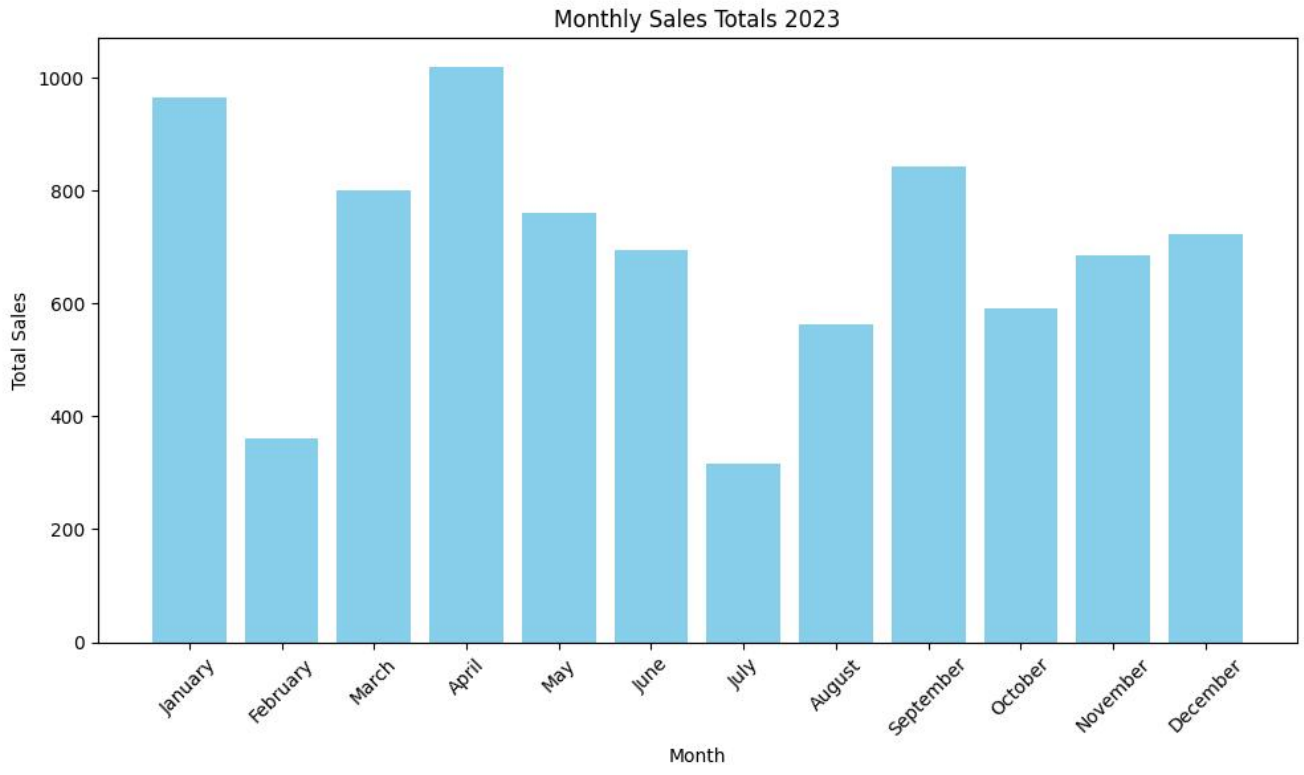
The results of this query were transferred into a Pandas DataFrame (`sales_figures_df) for further analysis. I then ran the following code to calculate the total price and group by sales month:

``` python
monthly_sales_totals = sales_figures_df.groupby('sales_month')
['total_price'].sum()
```

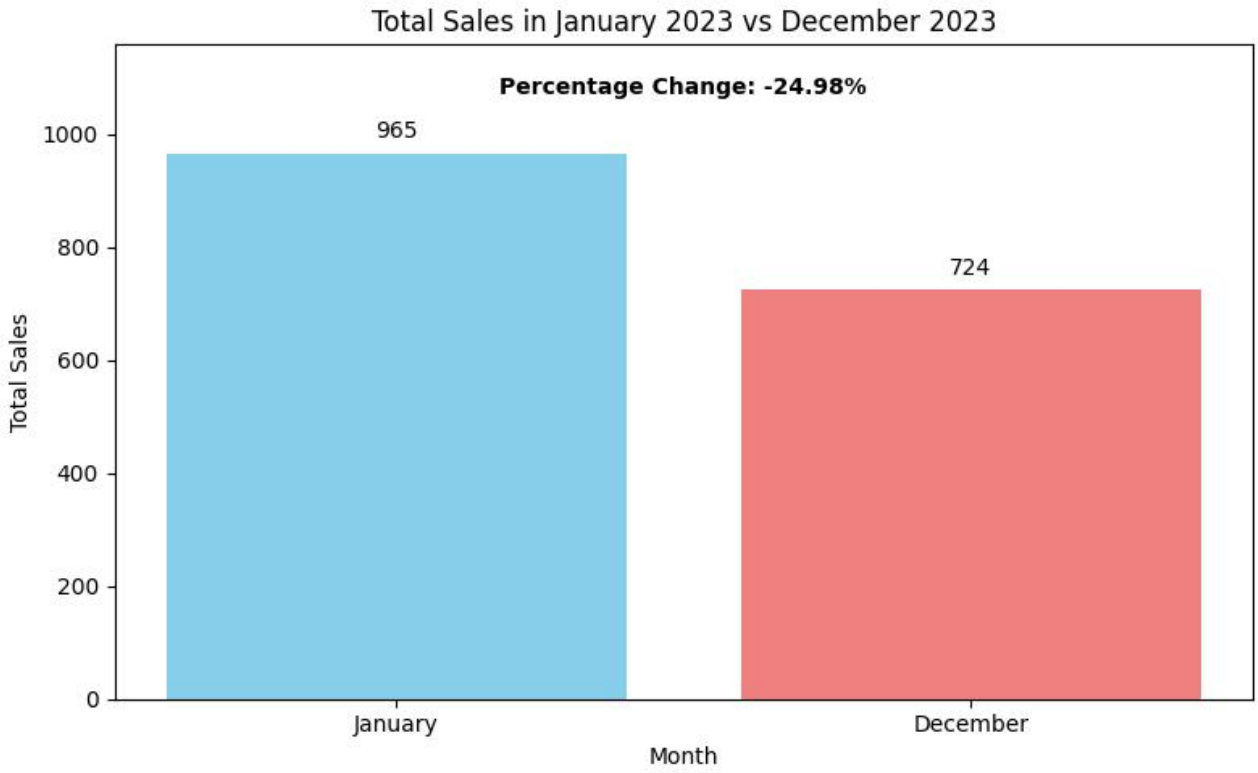
And finally, I calculated the percentage sales growth using the following line of code:

```python
percentage_change = ((december_sales - january_sales) / january_sales) * 100
```

I then used Matplotlib to plot a bar chart showing the total sales figures for each month, and a second bar chart displaying the total sales figures for January 2023 and December 2023 only, along with the percentage difference between these. These graphs are shown below.



Histogram displaying the total monthly sales figure for each month of 2023



Histogram displaying the total sales figures for January 2023 and December 2023, along with percentage change.

References

- OpenAI (2024). *ChatGPT*. [online] chat.openai.com. Available at: https://chat.openai.com. Accessed 11/04/2024.

- Mockaroo (n.d.). Mockaroo - Random Data Generator and API Mocking Tool | JSON / CSV / SQL / Excel. [online] www.mockaroo.com. Available at: https://www.mockaroo.com/. Accessed 11/04/2024.

- Lucidchart. (n.d.). *Online Diagram Software & Visual Solution*. [online] Available at: https://lucidchart.com.

FULL SQL SCRIPT FOR PART 2

```sql
-- Part 2

-- ##########################################################

-- 1) Create the database

CREATE DATABASE beauty_retailer;

--
##########################################################customers##
####

-- 2) Create the necessary tables

USE beauty_retailer;

-- CUSTOMER ADDRESSES
CREATE TABLE customer_addresses (
    address_id varchar (20) NOT NULL,
    street_address varchar (200),
    city text (20),
    region text (20),
    area_code varchar (10),
    PRIMARY KEY (address_id)
);

-- CUSTOMERS
CREATE TABLE customers (
    customer_id varchar (20) NOT NULL,
    first_name text (50),
    last_name text (50),
    address_id varchar (20),
    customer_email varchar (200),
    customer_phone varchar (20),
    PRIMARY KEY (customer_id),
    FOREIGN KEY (address_id) REFERENCES
customer_addresses(address_id)
);

-- EMPLOYEES
CREATE TABLE employees (
    employee_id varchar (20) NOT NULL,
    first_name text (50),
```

```
    last_name text (50),
    employee_email varchar (200),
    employee_phone varchar (20),
    contract_type enum ("Full-time", "Part-time"),
    salary int,
    PRIMARY KEY (employee_id)
);

-- SUPPLIERS
CREATE TABLE suppliers (
    supplier_id varchar (20) NOT NULL,
    supplier_name varchar (50),
    account_no varchar (50),
    supplier_email varchar (200),
    supplier_phone varchar (20),
    PRIMARY KEY (supplier_id)
);

-- PRODUCTS
CREATE TABLE products (
    product_id varchar (20) NOT NULL,
    product_description varchar (200),
    unit_price double,
    category enum ("Skin", "Body", "Hair", "Make-up"),
    supplier_id varchar (20) NOT NULL,
    qty_on_hand int,
    PRIMARY KEY (product_id),
    FOREIGN KEY (supplier_id) REFERENCES
suppliers(supplier_id)
);

-- ORDERS
CREATE TABLE orders (
    order_id varchar (20) NOT NULL,
    customer_id varchar (20) NOT NULL,
    created_at datetime,
    total_price double,
    fulfilled_at datetime,
    shipped_at datetime,
    PRIMARY KEY (order_id),
    FOREIGN KEY (customer_id) REFERENCES
customers(customer_id)
);

-- ORDER ITEMS
CREATE TABLE order_items (
    order_item_id varchar (20) NOT NULL,
    order_id varchar (20),
    product_id varchar (20),
```

```
    product_qty int,
    PRIMARY KEY (order_item_id),
    FOREIGN KEY (order_id) REFERENCES orders(order_id),
    FOREIGN KEY (product_id) REFERENCES products(product_id)
);

-- TRANSACTIONS
CREATE TABLE transactions (
    transaction_id varchar (20) NOT NULL,
    transaction_date datetime,
    order_id varchar (20),
    payment_method enum ("Visa", "Mastercard", "Paypal"),
    transaction_status enum ("Completed", "Cancelled"),
    employee_id varchar (20),
    PRIMARY KEY (transaction_id),
    FOREIGN KEY (employee_id) REFERENCES
employees(employee_id),
    FOREIGN KEY (order_id) REFERENCES orders(order_id)
);

--
##############################################################
###########

-- 3) Populate tables using DML

-- CUSTOMER ADDRESSES
INSERT INTO customer_addresses VALUES
    ('YGE8078', '64528 Chinook Junction', 'Newark',
'Delaware', '19714'),
    ('LUU3900', '432 Jay Street', 'Pittsburgh',
'Pennsylvania', '15274'),
    ('DLY0706', '34343 Duke Plaza', 'Fort Myers', 'Florida',
'33994'),
    ('FGU3634', '56 Brentwood Way', 'Las Vegas', 'Nevada',
'89160'),
    ('ADR0880', '96769 Waubesa Hill', 'Mesa', 'Arizona',
'85205');

-- CUSTOMERS
INSERT INTO customers VALUES
    ('XOY5256', 'Syd', 'Sage',
'DLY0706','ssage0@craigslist.org', '(524) 7673759'),
    ('HIT2934', 'Rudd', 'Moyse',
'ADR0880','rmoyse1@clickbank.net', '(935) 6014244'),
    ('PYT9472', 'Jacobo', 'Tarte',
'LUU3900','jtarte2@nydailynews.com', '(546) 3268499'),
    ('UMO3332', 'Trisha', 'Huck',
'YGE8078','thuck3@dagondesign.com', '(424) 3526176'),
```

```
    ('VNM6110', 'Alta', 'Ellwood',
'FGU3634','aellwood4@odnoklassniki.ru', '(834) 5741580');

-- EMPLOYEES
INSERT INTO employees VALUES
    ('YSC7673', 'Cyrille', 'Vayne', 'cvayne0@cbc.ca', '(994)
8066544', 'Part-time', 28495),
    ('TIU2870', 'Eldredge', 'Ribbens', 'eribbens1@goo.gl',
'(175) 3369851', 'Full-time', 88305),
    ('MQB9199', 'Ophelie', 'Thing', 'othing2@issuu.com',
'(396) 5863926', 'Part-time', 43217),
    ('EHM6216', 'Levy', 'Turney', 'lturney3@unicef.org',
'(435) 4120669', 'Full-time', 32921),
    ('KYM5242', 'Reinwald', 'Senner',
'rsenner4@technorati.com', '(269) 6407233', 'Full-time',
61666);

-- ORDERS
INSERT INTO orders VALUES
    ('XKO2243', 'VNM6110', '2023-06-11 22:43:51', 34.67, null,
null),
    ('YCZ8807', 'UMO3332', '2023-09-25 06:02:35', 88.26,
'2023-09-25 09:33:57', '2023-09-25 15:56:39'),
    ('ZJJ7251', 'XOY5256', '2023-02-10 19:27:05', 11.25,
'2023-02-11 09:05:32', '2023-02-11 16:14:36'),
    ('HBE5030', 'HIT2934', '2023-11-19 07:29:58', 9.94,
'2023-11-19 11:48:17', '2023-11-19 16:32:56'),
    ('YJD9045', 'PYT9472', '2023-10-08 18:48:32', 31.24,
'2023-10-09 10:02:11', '2023-10-09 16:48:17');

-- SUPPLIERS
INSERT INTO suppliers VALUES
    ('ZKM5655', 'GlowLux Distributors', '689012A',
'contact@glowluxdistributors.com', '(432) 4429887'),
    ('JYN8318', 'Dermalogica Wholesale', '24412533',
'wholesale@dermalogica.com', '(318) 9427212'),
    ('BRU1439', 'Beauty Source Distributors', 'TW27882',
'info@beautysourcedistributors.com', '(976) 5385727'),
    ('LVB7431', 'Nuxe Wholesale', 'A51875393',
'wholesale@nuxe.com', '(795) 2492220'),
    ('ZWP4499', 'Beauty Hub', '24329', 'info@beautyhub.com',
'(323) 5587896'),
    ('AAL4559', 'Glamour Goods Ltd.', '60MJ9659',
'sales@glamourgoods.com', '(328) 3581860');

-- PRODUCTS
INSERT INTO products VALUES
    ('ZJJ7251', 'Aveeno Daily Moisturizing Lotion', 9.80,
'Body', 'BRU1439', 24),
```

```
    ('YCZ8807', 'Dermalogica Special Cleansing Gel', 31.24,
'Skin', 'JYN8318', 8),
    ('HBE5030', 'Nuxe Huile Prodigieuse Multi-Purpose Dry
Oil', 17.33, 'Body', 'LVB7431', 14),
    ('YJD9045', 'Maybelline Fit Me Matte + Poreless
Foundation', 9.94, 'Make-up', 'AAL4559', 22),
    ('XKO2243', 'Redken Extreme Anti-Snap Leave-In Treatment',
11.25, 'Hair', 'ZWP4499', 17);

-- ORDER_ITEMS
INSERT INTO order_items VALUES
    ('EYC9489', 'ZJJ7251', 'XKO2243', 1),
    ('OBU5799', 'YJD9045', 'YCZ8807', 1),
    ('EON3113', 'XKO2243', 'HBE5030', 2),
    ('UIG0823', 'YCZ8807', 'ZJJ7251', 9),
    ('NNO9305', 'HBE5030', 'YJD9045', 1);

-- TRANSACTIONS
INSERT INTO transactions VALUES
    ('XBU8173', '2023-10-08 18:50:27', 'YJD9045', 'Visa',
'Completed', 'MQB9199'),
    ('FUG7223', '2023-11-19 07:33:14', 'HBE5030', 'Visa',
'Completed', 'MQB9199'),
    ('MMV2192', '2023-06-11 22:45:19', 'XKO2243', 'Paypal',
'Cancelled', 'YSC7673'),
    ('QDW2915', '2023-02-10 19:31:02', 'ZJJ7251',
'Mastercard', 'Completed', 'KYM5242'),
    ('LIV8153', '2023-09-25 06:05:49', 'YCZ8807', 'Paypal',
'Completed', 'TIU2870');
```

FULL SQL SCRIPT FOR PART 3

```sql
-- Part 3

--
#############################################################
#########################

-- 1) Show all the products along with the supplier detail who
supplied the products

SELECT products.product_id, products.product_description,
suppliers.supplier_id, suppliers.supplier_name
FROM products JOIN
suppliers ON products.supplier_id = suppliers.supplier_id;

--
#############################################################
#########################

-- 2) Create a stored procedure that takes the start and end
dates of the sales and
-- display all the sales transactions between the start and
the end dates.

-- Depending how you would interpret this question.
-- Either, the stored procedure displays all sales
transactions between the
-- earliest date in the table (start date) and the most recent
date in the table (end date),
-- ordered from earliest date to most recent date.

DELIMITER //
CREATE PROCEDURE displayAllSalesTransactions()
BEGIN
    SELECT *
    FROM transactions
    ORDER BY transaction_date;
END //

CALL displayAllSalesTransactions();

-- Or, the stored procedure takes in a start date and an end
date as parameters, and displays
-- all transactions between these two dates inclusive, ordered
by earliest to most recent.
```

```
DELIMITER //
CREATE PROCEDURE displaySalesPeriodTransactions(IN start_date
datetime, end_date datetime)
BEGIN
    SELECT *
    FROM transactions
    WHERE transaction_date >= start_date AND transaction_date
<= end_date
    ORDER BY transaction_date;
END //

CALL displaySalesPeriodTransactions('2023-01-01',
'2023-12-31');

--
##########################################################################
##########################

-- 3) Create a view that shows the total number of items a
customer buys from the business
-- in October 2023 along with the total price (use group by)

CREATE VIEW october_customer_purchases AS
SELECT q2.customer_id, q2.first_name, q2.last_name,
q1.total_qty_items, q2.total_price FROM
(
SELECT order_id, SUM(product_qty) AS total_qty_items
FROM order_items
GROUP BY order_id
) AS q1
JOIN
(
SELECT orders.order_id,  orders.customer_id,
customers.first_name, customers.last_name, orders.total_price,
transactions.transaction_date, transactions.transaction_status
FROM orders
JOIN customers ON orders.customer_id = customers.customer_id
JOIN transactions ON orders.order_id = transactions.order_id
) AS q2
ON q1.order_id = q2.order_id
WHERE q2.transaction_date >= '2023-10-01' AND
q2.transaction_date <= '2023-10-31' AND
q2.transaction_status != 'Cancelled';

--
##########################################################################
##########################
```

```sql
-- 4) Create a trigger that adjusts the stock level every time
a product is sold

DELIMITER //
CREATE TRIGGER stock_update
AFTER INSERT ON order_items
FOR EACH ROW
BEGIN
    UPDATE products
    SET products.qty_on_hand = products.qty_on_hand -
NEW.product_qty
    WHERE products.product_id = NEW.product_id;
END //
DELIMITER ;

--
##############################################################
##########################

-- 5) Create a report of the annual sales (2023) of the
business showing the total number of
-- products sold and the total price sold every month (use A
group by with roll-up)

CREATE TEMPORARY TABLE sales_figures
SELECT q1.order_id, MONTHNAME(q2.transaction_date) as
sales_month, q1.total_qty_items, q2.total_price FROM
(
SELECT order_id, SUM(product_qty) AS total_qty_items
FROM order_items
GROUP BY order_id
) AS q1
JOIN
(
SELECT orders.order_id,  orders.customer_id,
customers.first_name, customers.last_name,
transactions.transaction_date, orders.total_price,
transactions.transaction_status
FROM orders
JOIN customers ON orders.customer_id = customers.customer_id
JOIN transactions ON orders.order_id = transactions.order_id
) AS q2
ON q1.order_id = q2.order_id
WHERE transaction_date >= '2023-01-01' AND transaction_date <=
'2023-12-31' AND transaction_status != 'Cancelled';

SELECT COALESCE (sales_month, 'Annual Total') AS sales_month,
SUM(total_qty_items) AS total_items_sold,
ROUND(SUM(total_price),2)AS total_sales
```

```sql
FROM sales_figures
GROUP BY sales_month WITH ROLLUP;

-- 
##########################################################################
##########################

-- 6) Display the growth in sales/services (as a percentage)
for your business, from the 1st month of opening until now.

-- Sales Growth = (Most Recent Month Total Sales - 1st Month
Total Sales / 1st Month Total Sales) x 100

-- Create temp table to hold total monthly sales for January,
and December respectively
CREATE TEMPORARY TABLE sales_jan_dec
SELECT * FROM
(SELECT sales_month, ROUND(SUM(total_price),2) AS
monthly_sales
FROM sales_figures
GROUP BY sales_month
) AS q1
WHERE sales_month = 'January' OR sales_month = 'December';

-- Create temp table to calculate and hold difference between
January sales and December sales amounts
CREATE TEMPORARY TABLE sales_growth
SELECT sales_month, monthly_sales,
monthly_sales-LAG(monthly_sales) OVER (ORDER BY sales_month
desc) AS sales_difference
FROM sales_jan_dec;

-- Create table to calculate and hold this difference as a
fraction
CREATE TEMPORARY TABLE fractional_sales_growth
SELECT sales_month, monthly_sales, sales_difference,
sales_difference/LAG(monthly_sales) OVER (ORDER BY sales_month
desc) AS sales_fraction
FROM sales_growth;

-- Display percentage sales growth by multiplying the
fractional difference by 100
SELECT ROUND(sales_fraction * 100,2) AS
percentage_sales_growth
FROM fractional_sales_growth
WHERE sales_month = 'December';
```

```
--
##############################################################
##########################

-- 7) Delete all customers who never buy a product from the
business

-- Create a temp table with the customer IDs of customers with
at least one order fulfilled
CREATE TEMPORARY TABLE customers_qty_orders
SELECT customer_id,
    SUM(CASE
        WHEN fulfilled_at IS NULL THEN 0
        ELSE 1
        END)
    AS total_orders
    FROM orders
    GROUP BY customer_id;

-- Join the temp table with the customers table, and delete
all customers where
-- total_orders is null, i.e. they have never placed an order
DELETE customers FROM customers
LEFT JOIN customers_qty_orders ON customers.customer_id =
customers_qty_orders.customer_id
WHERE customers_qty_orders.total_orders IS NULL;
```

FULL PYTHON SCRIPT FOR PART 3.8

```python
# Use a reporting tool of your choice (e.g. Excel) to connect
to your data
# and create a graphical representation of the result of
question 6 in Part 3.

# import necessary modules
import mysql.connector
import pandas as pd
import matplotlib.pyplot as plt

# establish a connection to the database
db = mysql.connector.connect(
  host="localhost",
  user="root",
  password="password",
  database="beauty_retailer"
)

cursor = db.cursor()

# define the query to be run
query = """SELECT q1.order_id, MONTHNAME(q2.transaction_date)
as sales_month, q1.total_qty_items, q2.total_price FROM
(
SELECT order_id, SUM(product_qty) AS total_qty_items
FROM order_items
GROUP BY order_id
) AS q1
JOIN
(
SELECT orders.order_id,  orders.customer_id,
customers.first_name, customers.last_name,
transactions.transaction_date, orders.total_price,
transactions.transaction_status
FROM orders
JOIN customers ON orders.customer_id = customers.customer_id
JOIN transactions ON orders.order_id = transactions.order_id
) AS q2
ON q1.order_id = q2.order_id
WHERE transaction_date >= '2023-01-01' AND transaction_date <=
'2023-12-31' AND transaction_status != 'Cancelled';"""

# execute the query
cursor.execute(query)
```

```
# import the results of the query into a Pandas DataFrame
sales_figures_df = pd.DataFrame(cursor.fetchall())

# close the connection to the database
cursor.close()

# set DataFrame column names
sales_figures_df.columns =
["order_id","sales_month","total_qty_items","total_price"]

# calculate monthly total sales figures
monthly_sales_totals = sales_figures_df.groupby('sales_month')
['total_price'].sum()

# define the order of months
month_order = ['January', 'February', 'March', 'April', 'May',
'June', 'July', 'August', 'September', 'October', 'November',
'December']

# convert the index (month names) to a categorical data type
with the defined order
monthly_sales_totals.index =
pd.Categorical(monthly_sales_totals.index,
categories=month_order, ordered=True)

# sort the monthly sales totals DataFrame based on the
categorical order
monthly_sales_totals = monthly_sales_totals.sort_index()

# plot the bar chart of monthly sales totals
plt.figure(figsize=(10, 6))
plt.bar(monthly_sales_totals.index,
monthly_sales_totals.values, color='skyblue')
plt.xlabel('Month')
plt.ylabel('Total Sales')
plt.title('Monthly Sales Totals 2023')
plt.xticks(rotation=45)  # rotate x-axis labels for better
readability
plt.tight_layout()  # adjust layout to prevent clipping of
labels
plt.show()

# get the total sales for January and December
january_sales = monthly_sales_totals.loc['January']
december_sales = monthly_sales_totals.loc['December']

# calculate the percentage change
percentage_change = ((december_sales - january_sales) /
january_sales) * 100
```

```
# plot the bar graph of January vs December sales totals
plt.figure(figsize=(8, 5))
bars = plt.bar(['January', 'December'], [january_sales,
december_sales], color=['skyblue', 'lightcoral'])
plt.xlabel('Month')
plt.ylabel('Total Sales')
plt.title('Total Sales in January 2023 vs December 2023')
plt.ylim(0, max(january_sales, december_sales) * 1.2)  #
adjust ylim to give space for the text

# add text above each bar showing monthly sales total
for bar in bars:
    yval = bar.get_height()
    plt.text(bar.get_x() + bar.get_width()/2, yval +
0.02*max(january_sales, december_sales), round(yval),
ha='center', va='bottom')

# add text showing percentage change
plt.text(0.5, max(january_sales, december_sales) * 1.1,
f'Percentage Change: {percentage_change:.2f}%', ha='center',
va='bottom', fontsize=10, fontweight='bold')

plt.tight_layout()
plt.show()
```

TABLE OF DATA IMPORTED INTO customer_addresses

customer_addresses

address_id	street_address	city	region	area_code
DID7238	98917 Badeau Crossing	Jackson	Mississippi	39204
ING8434	33 Farmco Court	Omaha	Nebraska	68105
JNA1273	7535 3rd Alley	Atlanta	Georgia	31119
XQN9488	74718 Kedzie Crossing	Montgomery	Alabama	36109
BUR5695	5692 Clarendon Pass	Warren	Michigan	48092
NUN9935	0025 Surrey Avenue	Moreno Valley	California	92555
RKJ3513	50 Hermina Hill	Kissimmee	Florida	34745
JWA9846	5707 Delaware Trail	Atlanta	Georgia	30380
UXV9678	79 Thierer Crossing	Buffalo	New York	14269
BFZ9603	3989 Anderson Trail	Green Bay	Wisconsin	54305
LHC7614	8524 Annamark Junction	Cincinnati	Ohio	45223
AER5906	42 Portage Hill	Kansas City	Missouri	64187
JQB8634	9318 Harbort Lane	Pittsburgh	Pennsylvania	15255
YHH7010	3 Stone Corner Road	Fort Wayne	Indiana	46825
ZAP5208	958 Continental Road	Amarillo	Texas	79188
CMM7000	57 Straubel Plaza	Erie	Pennsylvania	16550
NJK4005	2 Arkansas Plaza	Little Rock	Arkansas	72204
EZU9216	7907 Jackson Alley	Miami	Florida	33190
SMB0047	986 Manley Plaza	Juneau	Alaska	99812
QSU4011	29 Larry Junction	Boston	Massachusetts	2109
JFC4907	16 Garrison Avenue	Lake Charles	Louisiana	70616
XTN5014	9697 Dawn Street	Charleston	West Virginia	25313
DIX1200	6 Mitchell Crossing	Detroit	Michigan	48242
NFI6723	1 Ridgeview Court	Charlotte	North Carolina	28299
MFB7042	12 Scoville Place	Fort Wayne	Indiana	46825
GGV0794	325 Hovde Way	Aurora	Illinois	60505
VLS6721	7 Cascade Plaza	Virginia Beach	Virginia	23454
ENY7712	6 Logan Plaza	Pasadena	California	91199
EUL2576	10309 Fremont Plaza	Richmond	Virginia	23242
CON0942	1898 Hoepker Way	Norwalk	Connecticut	6859
OTO1629	367 Norway Maple Terrace	Charlotte	North Carolina	28299
OUX4472	28196 Homewood Point	Mobile	Alabama	36628

TABLE OF DATA IMPORTED INTO customers

customers

customer_id	first_name	last_name	address_id	customer_email	customer_phone
YSU6749	Gaby	Hellmer	JWA9846	ghellmer0@examiner.com	(394) 3450237
UYE8994	Karna	Duckerin	LHC7614	kduckerin1@craigslist.org	(732) 1886846
VYV1199	Lester	Culp	JNA1273	lculp2@themeforest.net	(388) 9602588
FJT6890	Thaddeus	Setterthwait	YHH7010	tsetterthwait3@simplemachines.org	(827) 4508015
WUJ3083	Silvanus	Shipley	JFC4907	sshipley4@artisteer.com	(738) 1319226
TLN6950	Bobby	Petrashov	BFZ9603	bpetrashov5@wufoo.com	(578) 2595002
CHP6181	Deny	Sapwell	VLS6721	dsapwell6@webmd.com	(854) 8478474
BZM3857	Kamilah	Groven	CON0942	kgroven7@nba.com	(226) 2198128
NGM6346	Sergio	Satterlee	OTO1629	ssatterlee8@fastcompany.com	(758) 4753910
TUP9847	Iormina	Conroy	DID7238	iconroy9@gnu.org	(565) 6931279
GWM2589	Ally	Wesson	VBS1115	awessona@naver.com	(166) 8509638
JFS4924	Amberly	Bess	PVY4946	abessb@goodreads.com	(450) 8092506
GFX5649	Arv	Oughtright	LNX3166	aoughtrightc@imgur.com	(347) 6216676
AQW6767	Vaclav	Falkingham	ENY7712	vfalkinghamd@zimbio.com	(957) 9531578
DFK9646	Dana	Kender	ING8434	dkendere@wikia.com	(539) 1953801
NUQ8103	Dorita	Peak	FTC1479	dpeakf@rakuten.co.jp	(198) 4282440
PXJ7422	Chloe	Blaza	CMM7000	cblazag@gmpg.org	(527) 5076461
GDW6513	Judon	Eltun	UXV9678	jeltunh@xrea.com	(525) 5857335
BKK0556	Cris	Healks	BHH2261	chealksi@delicious.com	(736) 3856808
XOR6605	Bernie	Ralston	ZAP5208	bralstonj@drupal.org	(121) 4260881
UIG2027	Aleksandr	Bohlens	XTN5014	abohlensk@godaddy.com	(876) 8822637
EJR6771	Geri	McGeagh	XQN9488	gmcgeaghl@rambler.ru	(648) 3944208
OWW0857	Meyer	Gasparro	EUL2576	mgasparrom@state.tx.us	(879) 8713974
KJP4738	Emmerich	Spradbery	GGV0794	espradberyn@amazon.de	(908) 7045036
EDT4156	Dana	Lippingwell	IPK3290	dlippingwello@google.co.uk	(189) 7257301
RQY5898	Roarke	Fritche	DIX1200	rfritchep@exblog.jp	(596) 9569781
BAQ9072	Tommy	Meagher	FSH9131	tmeagherq@cnn.com	(421) 5672834
ZZA0678	Kaylyn	Flinn	ROJ1116	kflinnr@free.fr	(709) 1633919
DAN6727	Jules	Gilbeart	NUN9935	jgilbearts@sphinn.com	(238) 6968718

customers (continued)

WNY1074	Harry	Everist	YGQ0029	heveristt@g.co	(556) 5291572
YJW9205	Berna	Sighart	DOH0059	bsighartu@sciencedaily.com	(242) 4090004
HTY2508	Cathie	Crampton	SMB0047	ccramptonv@imgur.com	(281) 7178906
CQP8886	Papagena	Shills	FDT4632	pshillsw@google.pl	(630) 9106475
JFW6309	Goldie	Iliff	WPW2308	giliffx@bing.com	(607) 8115884
VLP0909	Ceciley	Siddeley	OUX4472	csiddeleyy@smugmug.com	(924) 6828883
HOE2306	Lydon	Travis	KMM3767	ltravisz@twitter.com	(983) 5371756
FON0205	Chickie	Joynson	EZU9216	cjoynson10@biblegateway.com	(608) 3876180
BVJ0033	Nels	Fellis	NHA5006	nfellis11@bigcartel.com	(203) 6939025
QST5108	Viva	MacRanald	CQI4183	vmacranald12@cbc.ca	(386) 8697899
WGD5669	Bard	Haggata	NFI6723	bhaggata13@deviantart.com	(163) 1192516
JLK1922	Melvin	Tydeman	OFH8359	mtydeman14@oracle.com	(240) 1938470
REL4866	Idalina	MacMaykin	SER8687	imacmaykin15@vistaprint.com	(727) 2621184
XAS5234	Vernen	Alabaster	BUR5695	valabaster16@g.co	(192) 4666400
YOE7299	Batholomew	Ramsell	JQB8634	bramsell17@shutterfly.com	(957) 3825033
VJQ3919	Jayme	Daldry	MFB7042	jdaldry18@state.gov	(332) 7801728
GBF1112	Jacquette	Fann	XCI7628	jfann19@wsj.com	(201) 1590548
BOU8525	Sadella	Elderton	QSU4011	selderton1a@cbc.ca	(864) 6307420
ABB1713	Pauly	Lucia	RKJ3513	plucia1b@sogou.com	(900) 9281163
LQF2777	Andre	Caven	NJK4005	acaven1c@omniture.com	(250) 7655290
XLM3332	Al	Cirlos	AER5906	acirlos1d@nhs.uk	(931) 6980320

TABLE OF DATA IMPORTED INTO employees

employees

employee_id	first_name	last_name	employee_email	employee_phone	contract_type	salary
EMC3367	Faye	Anetts	fanetts0@usa.gov	(988) 1836711	Full-time	88990
WDI0088	Hal	Jeandet	hjeandet1@digg.com	(440) 3469661	Part-time	95463
XBV9882	Vanda	Colvine	vcolvine2@ow.ly	(550) 9846323	Part-time	50945
LRS8751	Hillie	Zanetti	hzanetti3@webmd.com	(741) 3434882	Part-time	57376
UPL3529	Halsy	Banville	hbanville4@yandex.ru	(308) 8139187	Part-time	30345
HUK7721	Anthia	Ishaki	aishaki5@nifty.com	(911) 7221133	Part-time	38805
PGY8012	Kathie	McPhelimy	kmcphelimy6@jigsy.com	(900) 1365066	Part-time	31260
UKA6948	Kip	Pfeffer	kpfeffer7@typepad.com	(721) 4725968	Full-time	47097
YVP0526	Buckie	Hawkswood	bhawkswood8@ameblo.jp	(965) 4144022	Part-time	68887
MTO3962	Gerrie	Shemilt	gshemilt9@newyorker.com	(420) 5326269	Full-time	90942
KBG6195	Roma	Golder	rgoldera@artisteer.com	(657) 1302649	Full-time	60551
HGI8774	Michelina	Oldman	moldmanb@istockphoto.com	(254) 2897217	Full-time	21575
FYK9284	Yuri	Caulier	ycaulierc@eventbrite.com	(451) 3243834	Full-time	92238
CMJ2218	Tiena	Dunnet	tdunnetd@ebay.com	(509) 9196899	Full-time	95576
RTQ9621	Darcey	Nurdin	dnurdine@liveinternet.ru	(600) 9789562	Full-time	34410

TABLE OF DATA IMPORTED INTO order_items

order_items

order_item_id	order_id	product_id	product_qty
PYM8587	XCH1259	TQD1871	2
OBR5714	XCH1259	OYO0700	1
OFS5487	XCH1259	MQX6614	1
CUO2492	ZON4632	MHV4858	3
ROQ5055	ZON4632	VMR3148	1
JPO2484	GOO2815	JXE3292	2
KRM9129	GOO2815	HVP6071	2
DFA6791	GOO2815	TQD1871	1
MXI0371	XKY8775	TXO9914	3
FXZ7033	XKY8775	OYO0700	1
FYM8003	YHI8553	MQX6614	2
FBM3576	YHI8553	OYO0700	1
BFY2849	YHI8553	HVP6071	1
YPX0580	UCB8992	TQD1871	1
XXS3751	UCB8992	JXE3292	1
NZQ1458	UCB8992	VMR3148	2
LDR1614	KSL3815	OYO0700	2
ROC8442	KSL3815	EAL1632	1
RYE4795	XNN4941	OYO0700	3
OKO3147	XNN4941	TQD1871	2
HKL7869	XNN4941	JXE3292	1
AHU3521	WWU0581	HVP6071	2
DDS0061	WWU0581	MQX6614	2
LLN6277	WWU0581	OYO0700	1
AES7879	UJH8557	EAL1632	2
QIE2624	UJH8557	VMR3148	1
LEG7139	ZVR4035	MQX6614	1
FQH0270	ZVR4035	TQD1871	2
XKO2813	ZVR4035	OYO0700	1

order_items (continued part 1)

ZRC6563	ZVR4035	HVP6071	1
ZQT3094	AVM3211	JXE3292	2
VBL7976	AVM3211	OYO0700	1
DJZ2318	TSP2290	EAL1632	1
CCR9682	TSP2290	OYO0700	2
VIL5515	TSP2290	JXE3292	1
BAW0501	XCZ6377	OYO0700	1
OUM9120	XCZ6377	TQD1871	2
VAN8100	XCZ6377	JXE3292	2
PGW6067	YWF8858	EAL1632	2
NPN4790	YWF8858	OYO0700	2
BBO6095	YWF8858	JXE3292	1
CPO9062	NPN1579	TQD1871	2
DSZ9828	NPN1579	MQX6614	2
VTZ7651	NPN1579	OYO0700	1
ENV2730	ABY1276	OYO0700	2
BTC5202	ABY1276	TQD1871	2
NHJ8761	ABY1276	JXE3292	1
SJQ5003	KDZ0027	EAL1632	1
LOA0238	KDZ0027	MQX6614	2
UQU3187	KDZ0027	OYO0700	2
TFE5839	WIC6142	OYO0700	1
WGY4979	WIC6142	TQD1871	1
JMT3675	WIC6142	JXE3292	2
FWB5581	WIC6142	HVP6071	1
LKH9005	SYN6507	EAL1632	1
HLL9650	SYN6507	OYO0700	2
ZYV7646	SYN6507	JXE3292	2
OZS0653	IYJ8061	TQD1871	2
KNR7722	IYJ8061	OYO0700	2
QCG4573	IYJ8061	JXE3292	1
CEN7408	JYH0715	EAL1632	2
YJF1975	JYH0715	OYO0700	1
PGW3073	JYH0715	JXE3292	2

order_items(continued part 2)

KSQ7047	IRT5508	OYO0700	2
SVF7217	IRT5508	TQD1871	1
MLB0927	IRT5508	JXE3292	2
EES8246	HQW7234	MQX6614	2
FKG5034	HQW7234	OYO0700	2
HQP7979	HQW7234	HVP6071	1
XSY6055	QZQ7375	EAL1632	2
RCM3282	QZQ7375	OYO0700	1
JVZ0928	QZQ7375	JXE3292	2
PGD0147	SWK9062	OYO0700	1
UQS4880	SWK9062	TQD1871	2
WDP6940	SWK9062	JXE3292	2
QIY2678	CIP8540	OYO0700	1
CNG0285	CIP8540	EAL1632	1
UDL4974	CIP8540	JXE3292	2
DIU7660	CIP8540	HVP6071	2
MWB8573	AOW9891	MQX6614	1
PII1270	AOW9891	OYO0700	2
AAX0306	AOW9891	EAL1632	2
KTI2286	SJS7468	JXE3292	2
JDT5267	SJS7468	OYO0700	2
UTP2475	SJS7468	TQD1871	1
AVB3932	KBN3694	EAL1632	1
RNP8816	KBN3694	MQX6614	2
KAQ5496	KBN3694	OYO0700	2
LCV4787	KBN3694	HVP6071	1
BKX0890	LTO1939	OYO0700	1
WPU5622	LTO1939	EAL1632	2
LNT9701	LTO1939	JXE3292	2
VKP0590	OWF1375	EAL1632	2
XDP8812	OWF1375	OYO0700	2
DCG0198	OWF1375	JXE3292	2
OSL7706	QXA9511	EAL1632	2
NLC4391	QXA9511	OYO0700	1
RWJ0341	QXA9511	JXE3292	2

order_items (continued part 3)

ZOB3433	MPV6830	TQD1871	2
RXJ8467	MPV6830	OYO0700	2
ILG4897	MPV6830	JXE3292	1
QQN8128	UJO2256	EAL1632	2
RFM0950	UJO2256	OYO0700	1
PMI6661	UJO2256	JXE3292	2
WLA3989	DYK2145	OYO0700	3
ORD2435	DYK2145	TQD1871	2
MVX8449	DYK2145	JXE3292	1
IZX3188	JXP6936	OYO0700	1
QRF3459	JXP6936	EAL1632	2
JJY7857	JXP6936	JXE3292	2
HUV4522	AGM0946	OYO0700	1
DWB8100	AGM0946	TQD1871	2
AVL1911	AGM0946	JXE3292	2
GMI5075	DVW9406	OYO0700	2
JGB9329	DVW9406	EAL1632	1
OVF1698	DVW9406	JXE3292	2
KBG2952	YKF2765	TQD1871	2
ZDG2979	YKF2765	OYO0700	2
PWV0195	YKF2765	JXE3292	1
BJB6899	DUG2232	TQD1871	2
NXV1938	DUG2232	OYO0700	1
IDQ0183	DUG2232	JXE3292	2
BNW8654	EPI2686	MQX6614	2
BDR5435	EPI2686	OYO0700	1
BTP5585	EPI2686	EAL1632	1
GEC3117	DCV2217	EAL1632	1
LMJ6431	DCV2217	OYO0700	1
OQX6538	DCV2217	JXE3292	2
ISM3621	WWK4092	TQD1871	2
YGU7513	WWK4092	OYO0700	2
VFD8342	WWK4092	MQX6614	1
KCJ3733	APW5105	EAL1632	2
JSR7383	APW5105	OYO0700	2
WYO0891	APW5105	JXE3292	1

order_items (continued part 4)

WXF2837	CZB9315	TQD1871	2
RBR3036	CZB9315	OYO0700	2
TLZ6242	CZB9315	JXE3292	1
BBX2485	SVD9310	EAL1632	1
DJT3431	SVD9310	OYO0700	2
JGU2444	SVD9310	JXE3292	2
SBE9681	GFY8908	OYO0700	2
BLM1417	GFY8908	TQD1871	2
PTM1285	GFY8908	JXE3292	2
KFH4663	EYE9155	TQD1871	2
PHM4595	EYE9155	OYO0700	2
MQF3055	EYE9155	JXE3292	1
TSM9854	PHE3007	OYO0700	2
RIR4286	PHE3007	EAL1632	2
KDL7802	PHE3007	VMR3148	1
FEU0364	QBP6211	OYO0700	2
GOR0550	QBP6211	EAL1632	2
ZFE0161	QBP6211	JXE3292	1
FCR4467	JQV3113	TQD1871	1
HLA4657	JQV3113	OYO0700	2
BDD2054	JQV3113	JXE3292	2
LXG6389	WBS3190	TQD1871	2
VCA7568	WBS3190	OYO0700	1
UHV8568	WBS3190	MQX6614	1
PBY5176	ICL2952	MHV4858	3
KCB7557	ICL2952	VMR3148	1
UZO6903	GLO3857	JXE3292	2
HDH8606	GLO3857	HVP6071	2
WHI8962	GLO3857	TQD1871	1
OET5909	MNR7324	TXO9914	3
UNA9206	MNR7324	OYO0700	1
PQB6683	CPZ7432	MQX6614	2
BPV6324	CPZ7432	OYO0700	1
PXT0486	CPZ7432	HVP6071	1
MQL7553	HNW8866	TQD1871	1

order_items (continued part 5)

ZWS7535	HNW8866	JXE3292	1
YSV4551	HNW8866	VMR3148	2
YNH1805	ORH0891	OYO0700	2
GYE2511	ORH0891	EAL1632	1
GHM2133	HHE1677	OYO0700	3
BWH9144	HHE1677	TQD1871	2
ADE6451	HHE1677	JXE3292	1
KOX4318	LOC2874	HVP6071	2
EXW2948	LOC2874	MQX6614	2
ZYY4527	LOC2874	OYO0700	1
PWO3768	QPM0049	EAL1632	2
KKX6042	QPM0049	VMR3148	1
JLR9067	IVO8127	MQX6614	1
LJK9800	IVO8127	TQD1871	2
LKU5515	IVO8127	OYO0700	1
QZM7172	IVO8127	HVP6071	1
BXT5618	KTG4876	JXE3292	2
LXC0316	KTG4876	OYO0700	1
OWX1025	WQY7550	EAL1632	1
YXD1716	WQY7550	OYO0700	2
WLB8327	WQY7550	JXE3292	1
LKB5520	EEP1833	OYO0700	1
AEU8527	EEP1833	TQD1871	2
MFJ9431	EEP1833	JXE3292	2
QEL9504	VRS5794	EAL1632	2
HTP4546	VRS5794	OYO0700	2
PJX8436	VRS5794	JXE3292	1
NON5187	BVW5294	TQD1871	2
DAO2911	BVW5294	MQX6614	2
WNN9283	BVW5294	OYO0700	1
LSK2267	BFW0036	OYO0700	2
AYA3654	BFW0036	TQD1871	2
IBP7805	BFW0036	JXE3292	1
WHS9964	DWO0404	EAL1632	1
IUU6334	DWO0404	MQX6614	2

order_items (continued part 6)

WJH1858	DWO0404	OYO0700	2
CFY3818	BKI6440	OYO0700	1
XSH4484	BKI6440	TQD1871	1
OMJ6793	BKI6440	JXE3292	2
HDF3531	BKI6440	HVP6071	1
MWX5225	IRF0016	EAL1632	1
FNE9538	IRF0016	OYO0700	2
VIM3774	IRF0016	JXE3292	2
IJE8634	OMX7836	TQD1871	2
EHW5348	OMX7836	OYO0700	2
EPS4054	OMX7836	JXE3292	1
KTZ0938	VTO8072	EAL1632	2
WMC4493	VTO8072	OYO0700	1
RPK9457	VTO8072	JXE3292	2
GCA9317	IOW5476	OYO0700	2
MFV0059	IOW5476	TQD1871	1
USM0597	IOW5476	JXE3292	2
MZV8949	LOX4355	MQX6614	2
WGW4002	LOX4355	OYO0700	2
JIJ9187	LOX4355	HVP6071	1
BHN8585	LFZ4494	EAL1632	2
BJD2018	LFZ4494	OYO0700	1
PKZ2815	LFZ4494	JXE3292	2
AJG1848	IWS0251	OYO0700	1
UTB6343	IWS0251	TQD1871	2
ESX4012	IWS0251	JXE3292	2
IDE8230	YQM7306	OYO0700	1
TNR0974	YQM7306	EAL1632	1
GTC5290	YQM7306	JXE3292	2
YHP2103	YQM7306	HVP6071	2
PGE6489	AUJ4925	MQX6614	1
BDD4166	AUJ4925	OYO0700	2
BPO3717	AUJ4925	EAL1632	2
OZL7061	CPL0919	JXE3292	2
OGP0874	CPL0919	OYO0700	2

order_items (continued part 7)

SLZ2157	CPL0919	TQD1871	1
NNU8095	JMV9265	EAL1632	1
ZTU1775	JMV9265	MQX6614	2
BEE1504	JMV9265	OYO0700	2
ZFE1436	JMV9265	HVP6071	1
EQQ691	WXP5516	OYO0700	1
KWX4592	WXP5516	EAL1632	2
ZKC3608	WXP5516	JXE3292	2
OGV5918	MOA5259	EAL1632	2
KSE2960	MOA5259	OYO0700	2
FCB9319	MOA5259	JXE3292	2
PVT0773	ZVT7472	EAL1632	2
UZV1753	ZVT7472	OYO0700	1
GQM0085	ZVT7472	JXE3292	2
CSQ5085	ASQ7165	TQD1871	2
IOE8399	ASQ7165	OYO0700	2
VTB7140	ASQ7165	JXE3292	1
PUC6512	JWA5441	EAL1632	2
TMP9096	JWA5441	OYO0700	1
BDG0917	JWA5441	JXE3292	2
LOT5233	YMN8938	OYO0700	3
OME7857	YMN8938	TQD1871	2
FPM7748	YMN8938	JXE3292	1
QST0633	GMB4874	OYO0700	1
LDU3889	GMB4874	EAL1632	2
TFS5909	GMB4874	JXE3292	2
CGU9954	ZEE3650	OYO0700	1
SVA5536	ZEE3650	TQD1871	2
OAF2037	ZEE3650	JXE3292	2
MMZ2102	KHO4228	OYO0700	2
QVM6875	KHO4228	EAL1632	1
DSE3964	KHO4228	JXE3292	2
NBY1500	XYU1488	TQD1871	2
LIS9026	XYU1488	OYO0700	2
BYX5887	XYU1488	JXE3292	1

order_items (continued part 8)

LGO0853	PNE9540	TQD1871	2
AGT1940	PNE9540	OYO0700	1
HAI1522	PNE9540	JXE3292	2
NTR3852	YBM7844	MQX6614	2
LAZ9403	YBM7844	OYO0700	1
KVP9834	YBM7844	EAL1632	1
PSF3571	LJB2637	EAL1632	1
NJI0588	LJB2637	OYO0700	1
RQA2323	LJB2637	JXE3292	2
CNB4805	WVL6557	TQD1871	2
DMF3028	WVL6557	OYO0700	2
RHA3291	WVL6557	MQX6614	1
IWP3456	OJA1455	EAL1632	2
WNE0434	OJA1455	OYO0700	2
XNS3782	OJA1455	JXE3292	1
BTL2982	JRL1883	TQD1871	2
HSV8751	JRL1883	OYO0700	2
WGI8252	JRL1883	JXE3292	1
ECP4945	EXU6344	EAL1632	1
BSC3226	EXU6344	OYO0700	2
WDX0497	EXU6344	JXE3292	2
AUP0354	YJV6348	OYO0700	2
ZZR2047	YJV6348	TQD1871	2
UCL0647	YJV6348	JXE3292	2

TABLE OF DATA IMPORTED INTO orders

orders

order_id	customer_id	created_at	total_price	fulfilled_at	shipped_at
XCH1259	EDT4156	2023-04-09 06:44:50	84.14	2023-04-09 12:33:56	2023-04-09 17:21:48
ZON4632	XLM3332	2023-11-14 15:29:28	57.80		
GOO2815	YJW9205	2023-02-05 10:56:29	105.36	2023-02-05 14:20:32	2023-02-05 17:02:27
XKY8775	YJW9205	2023-01-27 03:34:57	70.02	2023-01-27 11:05:48	2023-01-27 12:08:59
YHI8553	DAN6727	2023-06-24 17:45:01	65.64	2023-06-25 10:25:59	2023-06-25 14:27:43
UCB8992	WGD5669	2023-10-21 14:46:46	76.25	2023-10-23 10:55:28	2023-10-23 17:52:13
KSL3815	GWM2589	2023-09-04 06:35:44	58.51	2023-09-04 11:40:45	2023-09-04 16:37:47
XNN4941	TLN6950	2023-06-25 14:55:18	109.40	2023-06-26 12:15:37	2023-06-27 15:10:03
WWU0581	KJP4738	2023-01-09 21:29:56	73.02		
UJH8557	WNY1074	2023-07-16 17:50:13	59.45	2023-07-17 10:35:57	2023-07-17 17:23:09
ZVR4035	PYT9472	2023-01-23 18:23:59	83.86	2023-01-24 15:20:22	2023-01-25 09:12:54
AVM3211	WUJ3083	2023-04-14 19:00:30	88.50	2023-04-15 15:45:03	2023-04-16 09:17:02
TSP2290	XAS5234	2023-01-10 21:56:48	85.04	2023-01-11 14:10:56	2023-01-11 17:55:22
XCZ6377	OWW0857	2023-01-22 12:12:28	93.46	2023-01-23 10:10:39	2023-01-23 17:08:47
YWF8858	XLM3332	2023-05-30 00:37:41	94.58	2023-05-30 13:20:14	2023-05-30 16:31:56
NPN1579	HOE2306	2023-05-14 14:58:45	70.02	2023-05-15 10:45:33	2023-05-15 15:04:02
ABY1276	GFX5649	2023-11-26 03:04:24	95.70	2023-11-27 13:30:49	2023-11-27 17:52:45
KDZ0027	TLN6950	2023-11-05 06:43:57	69.62	2023-11-06 10:15:02	2023-11-06 15:02:18
WIC6142	RQY5898	2023-06-25 16:26:28	91.22	2023-06-26 12:40:57	2023-06-26 17:23:52
SYN6507	PYT9472	2023-05-18 20:12:05	86.76	2023-05-19 16:30:24	2023-05-20 09:23:47
IYJ8061	HOE2306	2023-09-10 10:50:02	98.04	2023-09-11 15:50:40	2023-09-11 16:52:49
JYH0715	VJQ3919	2023-01-08 18:39:46	91.72	2023-01-09 14:40:22	2023-01-09 17:14:39
IRT5508	FON0205	2023-03-12 04:13:16	89.44	2023-03-13 15:00:12	2023-03-13 17:52:59
HQW7234	DAN6727	2023-02-05 09:46:58	78.70	2023-02-05 15:35:29	2023-02-05 17:56:12
QZQ7375	PYT9472	2023-04-20 16:30:14	79.10	2023-04-20 12:20:48	2023-04-20 15:18:06
SWK9062	XLM3332	2023-08-25 13:08:03	85.62	2023-08-25 16:05:09	2023-08-26 10:16:59
CIP8540	GFX5649	2023-03-22 19:40:42	96.52	2023-03-23 14:35:54	2023-03-23 17:37:52
AOW9891	EDT4156	2023-06-09 13:01:36	72.88	2023-06-10 10:30:20	2023-06-10 11:48:12
SJS7468	WNY1074	2023-06-09 19:34:29	100.30		
KBN3694	DAN6727	2023-04-05 18:17:40	74.18	2023-04-06 13:40:26	2023-04-06 17:31:31
LTO1939	HIT2934	2023-11-26 15:47:05	90.22	2023-11-27 14:50:37	2023-11-27 17:33:48
OWF1375	VJQ3919	2023-03-20 19:35:43	103.72	2023-03-21 14:10:15	2023-03-21 16:11:49

orders (continued part 1)

QXA9511	WNY1074	2023-10-19 21:27:20	96.06	2023-10-21 09:20:33	2023-10-21 10:47:31
MPV6830	UYE8994	2023-12-30 04:46:08	91.62	2023-12-30 11:40:19	2023-12-30 14:46:59
UJO2256	REL4866	2023-11-18 17:12:26	94.26	2023-11-19 15:50:28	2023-11-19 15:57:12
DYK2145	JFS4924	2023-09-28 07:54:30	99.70	2023-09-28 12:30:44	2023-09-28 15:17:28
JXP6936	WUJ3083	2023-05-08 00:57:38	90.52	2023-05-08 12:14:52	2023-05-08 15:08:23
AGM0946	ABB1713	2023-09-26 21:35:27	92.52	2023-09-27 11:35:12	2023-09-27 15:03:52
DVW9406	BAQ9072	2023-10-23 20:13:49	95.92	2023-10-24 10:13:03	2023-10-24 16:17:19
YKF2765	ZZA0678	2023-09-09 01:23:48	74.26	2023-09-09 13:35:47	2023-09-09 15:39:04
DUG2232	YSU6749	2023-05-21 21:22:25	85.26	2023-05-22 11:22:03	2023-05-22 16:54:23
EPI2686	UMO3332	2023-10-29 12:33:19	68.74	2023-10-29 14:56:22	2023-10-29 17:58:29
DCV2217	WNY1074	2023-03-18 10:39:02	87.86	2023-03-18 15:31:08	2023-03-18 17:56:48
WWK4092	BKK0556	2023-01-15 13:43:00	77.70	2023-01-16 09:27:14	2023-01-16 14:47:11
APW5105	BKK0556	2023-08-19 07:28:29	99.12	2023-08-19 10:45:36	2023-08-19 15:02:49
CZB9315	YOE7299	2023-09-02 12:50:24	87.42	2023-09-02 17:42:54	2023-09-03 09:17:22
SVD9310	PYT9472	2023-12-28 17:11:36	102.02	2023-12-30 11:12:59	2023-12-30 15:37:17
GFY8908	LQF2777	2023-01-11 10:21:24	102.92	2023-01-11 12:33:21	2023-01-11 17:46:11
EYE9155	UMO3332	2023-09-05 19:24:00	87.62	2023-09-07 13:54:43	2023-09-07 15:14:14
PHE3007	HTY2508	2023-10-29 18:56:54	80.80	2023-10-30 14:20:57	2023-10-30 17:12:07
QBP6211	VNM6110	2023-02-18 06:51:14	78.92	2023-02-18 15:48:09	2023-02-18 17:58:02
JQV3113	PXJ7422	2023-01-29 11:48:13	92.32	2023-01-29 16:15:32	2023-01-29 17:55:38
WBS3190	PYT9472	2023-04-03 03:32:14	84.14	2023-04-03 09:10:28	2023-04-03 09:55:22
ICL2952	BKK0556	2023-11-02 02:12:09	57.80	2023-11-02 10:37:51	2023-11-02 14:31:59
GLO3857	VNM6110	2023-08-05 19:15:39	105.36	2023-08-06 11:59:13	2023-08-06 16:52:10
MNR7324	GWM2589	2023-03-23 14:14:58	70.02	2023-03-23 17:50:55	2023-03-24 09:12:05
CPZ7432	FJT6890	2023-10-22 04:31:41	65.64	2023-10-22 13:46:49	2023-10-22 17:42:09
HNW8866	WUJ3083	2023-10-20 23:29:18	76.25		
ORH0891	GFX5649	2023-01-04 11:48:30	58.51	2023-01-04 15:04:11	2023-01-04 17:01:57
HHE1677	PXJ7422	2023-01-28 22:14:11	109.40	2023-01-30 16:26:33	2023-01-30 17:54:13
LOC2874	BOU8525	2023-12-13 02:17:01	73.02	2023-12-13 16:45:49	2023-12-13 17:14:42
QPM0049	CHP6181	2023-05-03 02:06:10	59.45	2023-05-03 11:47:39	2023-05-03 15:14:37
IVO8127	HIT2934	2023-04-20 01:45:22	83.86	2023-04-20 10:30:25	2023-04-20 14:01:28
KTG4876	BVJ0033	2023-11-14 18:03:36	88.50	2023-11-15 12:15:52	2023-11-15 16:47:23
WQY7550	RQY5898	2023-06-20 03:27:31	85.04	2023-06-20 15:20:34	2023-06-20 17:21:31

EEP1833	REL4866	2023-06-07 17:00:57	93.46	2023-06-08 09:58:07	2023-06-08 10:15:47
VRS5794	OWW0857	2023-12-26 05:38:27	94.58	2023-12-28 13:33:06	2023-12-28 15:03:16
BVW5294	ZZA0678	2023-09-10 15:55:22	70.02	2023-09-11 09:21:14	2023-09-11 10:56:59
BFW0036	HTY2508	2023-07-13 01:46:24	95.70	2023-07-13 12:35:52	2023-07-13 15:17:42
DWO0404	WNY1074	2023-03-20 13:48:58	69.62	2023-03-20 17:11:41	2023-03-21 09:48:49
BKI6440	JLK1922	2023-12-09 05:16:52	91.22	2023-12-09 11:10:39	2023-12-09 15:17:53
IRF0016	XOY5256	2023-07-08 06:43:41	86.76	2023-07-08 15:49:29	2023-07-08 17:10:08
OMX7836	CQP8886	2023-08-08 23:07:23	98.04		
VTO8072	HIT2934	2023-12-25 09:18:40	91.72	2023-12-28 13:57:04	2023-12-28 15:52:24
IOW5476	HOE2306	2023-11-20 15:58:42	89.44	2023-11-21 09:42:53	2023-11-21 10:29:27
LOX4355	VLP0909	2023-08-26 04:03:19	78.70	2023-08-26 10:15:07	2023-08-26 14:05:52
LFZ4494	WGD5669	2023-05-10 20:26:33	79.10	2023-05-11 11:38:20	2023-05-11 14:18:07
IWS0251	ABB1713	2023-09-27 00:44:17	85.62	2023-09-27 13:24:45	2023-09-27 15:59:49
YQM7306	UYE8994	2023-03-04 21:25:25	96.52	2023-03-06 12:59:32	2023-03-06 15:52:03
AUJ4925	HOE2306	2023-04-29 11:18:00	72.88	2023-04-29 16:33:24	2023-04-29 17:00:01
CPL0919	WUJ3083	2023-04-15 05:49:54	100.30	2023-04-15 14:47:58	2023-04-15 17:42:00
JMV9265	PXJ7422	2023-06-14 18:47:40	74.18	2023-06-15 15:10:11	2023-06-15 17:01:51
WXP5516	ZZA0678	2023-03-10 02:13:24	90.22	2023-03-10 09:27:36	2023-03-10 14:22:39
MOA5259	WGD5669	2023-08-08 00:38:16	103.72	2023-08-08 12:38:15	2023-08-08 14:02:53
ZVT7472	UYE8994	2023-04-26 16:46:49	96.06	2023-04-27 10:50:49	2023-04-27 15:01:42
ASQ7165	JFW6309	2023-08-20 10:44:52	91.62	2023-08-20 17:29:38	2023-08-21 09:27:00
JWA5441	QST5108	2023-12-25 11:59:02	94.26	2023-12-28 11:15:02	2023-12-28 15:17:12
YMN8938	GWM2589	2023-01-22 00:23:48	99.70	2023-01-22 13:59:28	2023-01-22 14:58:05
GMB4874	JLK1922	2023-11-25 14:13:37	90.52	2023-11-25 17:07:07	2023-11-26 10:02:00
ZEE3650	KJP4738	2023-05-22 16:41:19	92.52	2023-05-23 09:52:51	2023-05-23 10:15:55
KHO4228	TLN6950	2023-03-27 16:02:06	95.92	2023-03-29 09:10:20	2023-03-29 10:52:27
XYU1488	BVJ0033	2023-07-31 04:34:53	74.26	2023-07-31 16:06:25	2023-07-31 17:23:06
PNE9540	FJT6890	2023-12-27 13:59:32	85.26	2023-12-29 15:44:54	2023-12-29 15:52:17
YBM7844	CHP6181	2023-04-05 02:44:54	68.74	2023-04-05 16:06:29	2023-04-05 17:18:02
LJB2637	CQP8886	2023-04-29 05:59:57	87.86	2023-04-29 11:56:46	2023-04-29 15:51:14
WVL6557	TLN6950	2023-10-21 07:13:03	77.70	2023-10-21 14:43:12	2023-10-21 17:41:29
OJA1455	VLP0909	2023-04-23 19:48:21	99.12	2023-04-24 13:57:42	2023-04-24 17:12:07
JRL1883	GFX5649	2023-02-18 07:38:04	87.42	2023-02-18 13:19:59	2023-02-18 16:02:02
EXU6344	YJW9205	2023-05-08 22:14:07	102.02	2023-05-10 10:33:33	2023-05-10 14:03:27
YJV6348	OWW0857	2023-06-10 14:44:34	102.92	2023-06-11 14:37:10	2023-06-11 15:03:01

JSON DATA IMPORTED INTO products

```json
[{"product_id":"JSE8099","product_description":"Kérastase Elixir Ultime L'Huile Original Hair Oil","unit_price":"28.99","category":"Hair","supplier_id":"ZWP4499","qty_on_hand":"32"},
{"product_id":"MHV4858","product_description":"Maybelline Instant Age Rewind Eraser Dark Circles Concealer","unit_price":"15.42","category":"Make-up","supplier_id":"AAL4559","qty_on_hand":"51"},
{"product_id":"EAL1632","product_description":"NYX Soft Matte Lip Cream","unit_price":"18.00","category":"Make-up","supplier_id":"AAL4559","qty_on_hand":"81"},
{"product_id":"XIV9557","product_description":"Wella Professionals Fusion Intense Repair Shampoo","unit_price":"9.89","category":"Hair","supplier_id":"BRU1439","qty_on_hand":"58"},
{"product_id":"IVA6751","product_description":"Redken Diamond Oil Glow Dry Gloss Scrub","unit_price":"9.44","category":"Hair","supplier_id":"BRU1439","qty_on_hand":"23"},
{"product_id":"TQD1871","product_description":"NYX Can't Stop Won't Stop Foundation","unit_price":"25.82","category":"Make-up","supplier_id":"AAL4559","qty_on_hand":"62"},
{"product_id":"MQX6614","product_description":"Rimmel Stay Matte Liquid Lip Colour","unit_price":"9.02","category":"Make-up","supplier_id":"AAL4559","qty_on_hand":"90"},
{"product_id":"PUC1476","product_description":"Neutrogena Hydro Boost Water Gel","unit_price":"9.87","category":"Skin","supplier_id":"BRU1439","qty_on_hand":"91"},
{"product_id":"ETX2027","product_description":"Pantene Pro-V Gold Series Moisture Boost Shampoo","unit_price":"13.68","category":"Hair","supplier_id":"BRU1439","qty_on_hand":"35"},
{"product_id":"TXO9914","product_description":"Kérastase Nutritive Fondant Magistral Conditioner","unit_price":"23.51","category":"Hair","supplier_id":"ZWP4499","qty_on_hand":"7"},
{"product_id":"ODD7731","product_description":"CoverGirl Exhibitionist Lipstick","unit_price":"12.95","category":"Make-up","supplier_id":"AAL4559","qty_on_hand":"13"},
{"product_id":"XME8860","product_description":"OGX Moroccan Curling Perfection Defining Cream","unit_price":"13.19","category":"Hair","supplier_id":"BRU1439","qty_on_hand":"17"},
{"product_id":"XHC4646","product_description":"SheaMoisture
```

```
Coconut & Hibiscus Curl & Shine
Shampoo","unit_price":"15.32","category":"Hair","supplier_id":
"BRU1439","qty_on_hand":"41"},
{"product_id":"VOM1356","product_description":"Dove Nourishing
Body Care Shea Butter Cream
Oil","unit_price":"9.13","category":"Body","supplier_id":"BRU1
439","qty_on_hand":"49"},
{"product_id":"BHE1645","product_description":"Rimmel
Scandaleyes Waterproof Kohl Kajal
Eyeliner","unit_price":"9.28","category":"Make-
up","supplier_id":"AAL4559","qty_on_hand":"89"},
{"product_id":"UJW9149","product_description":"Estée Lauder
Advanced Night Repair Synchronized Recovery Complex
II","unit_price":"65.10","category":"Skin","supplier_id":"ZKM5
655","qty_on_hand":"70"},
{"product_id":"LKV2832","product_description":"Aussie 3 Minute
Miracle Moist Deep
Conditioner","unit_price":"9.17","category":"Hair","supplier_i
d":"BRU1439","qty_on_hand":"76"},
{"product_id":"HVP6071","product_description":"Dove Deep
Moisture Body
Wash","unit_price":"12.80","category":"Body","supplier_id":"BR
U1439","qty_on_hand":"99"},
{"product_id":"KFS1137","product_description":"Lancôme Hypnôse
Drama Instant Full Volume
Mascara","unit_price":"37.20","category":"Make-
up","supplier_id":"ZKM5655","qty_on_hand":"36"},
{"product_id":"OYO0700","product_description":"Garnier
SkinActive Moisture Bomb Body
Lotion","unit_price":"14.64","category":"Body","supplier_id":"
BRU1439","qty_on_hand":"1"},
{"product_id":"IFR0468","product_description":"Nuxe Crème
Prodigieuse Boost Energising Priming
Concentrate","unit_price":"25.75","category":"Skin","supplier_
id":"LVB7431","qty_on_hand":"83"},
{"product_id":"NWW9824","product_description":"Aussie Instant
Freeze Sculpting Maximum Hold Hair
Gel","unit_price":"10.65","category":"Hair","supplier_id":"BRU
1439","qty_on_hand":"27"},
{"product_id":"HPP1147","product_description":"Nexxus
Emergencée Reconstructing
Treatment","unit_price":"21.14","category":"Hair","supplier_id
":"ZWP4499","qty_on_hand":"10"},
{"product_id":"JXE3292","product_description":"Estée Lauder
Advanced Night Repair Eye Supercharged
Complex","unit_price":"47.24","category":"Skin","supplier_id":
"ZKM5655","qty_on_hand":"8"},
{"product_id":"HAH0730","product_description":"Cetaphil Gentle
Foaming
```

```
Cleanser","unit_price":"9.57","category":"Skin","supplier_id":
"BRU1439","qty_on_hand":"55"},
{"product_id":"VMR3148","product_description":"Rimmel Stay
Matte Pressed Powder","unit_price":"16.80","category":"Make-
up","supplier_id":"AAL4559","qty_on_hand":"10"},
{"product_id":"VPQ6059","product_description":"Kérastase
Resistance Bain Force Architecte
Shampoo","unit_price":"30.15","category":"Hair","supplier_id":
"ZWP4499","qty_on_hand":"25"},
{"product_id":"HJT1844","product_description":"Olay Regenerist
Micro-Sculpting
Cream","unit_price":"15.77","category":"Skin","supplier_id":"B
RU1439","qty_on_hand":"13"},
{"product_id":"FJY7336","product_description":"Paul Mitchell
Awapuhi Wild Ginger Moisturizing Lather
Shampoo","unit_price":"38.69","category":"Hair","supplier_id":
"ZWP4499","qty_on_hand":"49"},
{"product_id":"QGE2196","product_description":"Garnier
SkinActive Hydra Bomb Sheet
Masks","unit_price":"10.02","category":"Skin","supplier_id":"B
RU1439","qty_on_hand":"65"},
{"product_id":"STE4119","product_description":"Clinique Take
The Day Off Cleansing
Balm","unit_price":"34.04","category":"Skin","supplier_id":"ZK
M5655","qty_on_hand":"20"},
{"product_id":"BBF7695","product_description":"NYX Epic Ink
Liner","unit_price":"25.02","category":"Make-
up","supplier_id":"AAL4559","qty_on_hand":"65"},
{"product_id":"UUC3968","product_description":"Dermalogica
Daily
Microfoliant","unit_price":"42.07","category":"Skin","supplier
_id":"JYN8318","qty_on_hand":"17"},
{"product_id":"MSC3408","product_description":"Nuxe Crème
Fraîche de Beauté Moisturizing
Cream","unit_price":"30.89","category":"Skin","supplier_id":"L
VB7431","qty_on_hand":"54"},
{"product_id":"JPG0960","product_description":"Dermalogica
MultiVitamin Power
Firm","unit_price":"93.06","category":"Skin","supplier_id":"JY
N8318","qty_on_hand":"24"},
{"product_id":"LDH8063","product_description":"The Body Shop
Shea Body
Butter","unit_price":"18.13","category":"Body","supplier_id":"
ZKM5655","qty_on_hand":"6"},
{"product_id":"QWA0412","product_description":"Olay Luminous
Whip Face
Moisturizer","unit_price":"9.24","category":"Skin","supplier_i
d":"BRU1439","qty_on_hand":"50"},
{"product_id":"QTJ1106","product_description":"Olaplex No. 0
```

```
Intensive Bond Building Hair
Treatment","unit_price":"36.16","category":"Hair","supplier_id
":"ZWP4499","qty_on_hand":"43"},
{"product_id":"RDZ0257","product_description":"Clinique Chubby
Stick Moisturizing Lip Colour
Balm","unit_price":"26.35","category":"Make-
up","supplier_id":"ZKM5655","qty_on_hand":"72"},
{"product_id":"NPX2200","product_description":"Joico Moisture
Recovery
Shampoo","unit_price":"6.15","category":"Hair","supplier_id":"
BRU1439","qty_on_hand":"64"},
{"product_id":"UKD1663","product_description":"Nexxus Color
Assure Long Lasting Vibrancy Deep Moisture
Masque","unit_price":"25.60","category":"Hair","supplier_id":"
ZWP4499","qty_on_hand":"29"},
{"product_id":"EFI0269","product_description":"The Body Shop
Fuji Green Tea Body
Butter","unit_price":"29.11","category":"Body","supplier_id":"
ZKM5655","qty_on_hand":"55"},
{"product_id":"JXK6218","product_description":"Estée Lauder
Pure Color Envy Sculpting
Lipstick","unit_price":"23.34","category":"Make-
up","supplier_id":"ZKM5655","qty_on_hand":"5"},
{"product_id":"UVJ3293","product_description":"Joico K-PAK
Deep-Penetrating
Reconstructor","unit_price":"14.70","category":"Hair","supplie
r_id":"BRU1439","qty_on_hand":"86"},
{"product_id":"DCZ4034","product_description":"Wella
Professionals Oil Reflections Luminous Reboost
Mask","unit_price":"7.92","category":"Hair","supplier_id":"BRU
1439","qty_on_hand":"1"},
{"product_id":"AGO7078","product_description":"Olay Fresh
Outlast Cooling White Strawberry & Mint Body
Wash","unit_price":"12.34","category":"Skin","supplier_id":"BR
U1439","qty_on_hand":"66"},
{"product_id":"OBR7377","product_description":"OGX Thick &
Full Biotin & Collagen
Shampoo","unit_price":"13.59","category":"Hair","supplier_id":
"BRU1439","qty_on_hand":"48"},
{"product_id":"UTK4133","product_description":"MoroccanOil
Treatment
Original","unit_price":"16.71","category":"Hair","supplier_id"
:"ZWP4499","qty_on_hand":"3"},
{"product_id":"PET9090","product_description":"CoverGirl Lash
Blast Volume Mascara","unit_price":"15.60","category":"Make-
up","supplier_id":"AAL4559","qty_on_hand":"3"},
{"product_id":"GDM0638","product_description":"Olaplex No. 8
Bond Intense Moisture
```

Mask","unit_price":"39.29","category":"Hair","supplier_id":"ZWP4499","qty_on_hand":"93"}]

```
```

TABLE OF DATA IMPORTED INTO transactions

transactions

transaction_id	transaction_date	order_id	payment_method	transaction_status	employee_id
VEX2917	2023-04-09 06:46:03	XCH1259	Visa	Completed	UKA6948
WBB0537	2023-11-14 15:32:10	ZON4632	Visa	Cancelled	YVP0526
DQL8277	2023-02-05 11:02:53	GOO2815	Paypal	Completed	WDI0088
OZY0984	2023-01-27 03:39:12	XKY8775	Mastercard	Completed	UPL3529
ZCY4027	2023-06-24 17:49:25	YHI8553	Visa	Completed	KBG6195
UGN2771	2023-10-21 14:49:36	UCB8992	Mastercard	Completed	HGI8774
JIT9849	2023-09-04 06:38:39	KSL3815	Visa	Completed	EHM6216
QRJ6750	2023-06-25 14:59:49	XNN4941	Visa	Completed	XBV9882
IUV9124	2023-01-09 21:32:54	WWU0581	Mastercard	Cancelled	TIU2870
LXD2261	2023-07-16 17:52:52	UJH8557	Mastercard	Completed	CMJ2218
BZO8610	2023-01-23 18:26:43	ZVR4035	Mastercard	Completed	WDI0088
ESS8857	2023-04-14 19:03:46	AVM3211	Visa	Completed	EHM6216
IRX0813	2023-01-10 21:59:59	TSP2290	Paypal	Completed	FYK9284
JHF1922	2023-01-22 12:16:05	XCZ6377	Visa	Completed	PGY8012
GYI3230	2023-05-30 00:41:19	YWF8858	Visa	Completed	MTO3962
MGX9268	2023-05-14 15:03:35	NPN1579	Visa	Completed	EHM6216
LKW2450	2023-11-26 03:08:47	ABY1276	Visa	Completed	RTQ9621
OEI2798	2023-11-05 06:47:18	KDZ0027	Mastercard	Completed	KBG6195
WKI5321	2023-06-25 16:29:47	WIC6142	Mastercard	Completed	KYM5242
BEJ9818	2023-05-18 20:15:54	SYN6507	Visa	Completed	LRS8751
ERM4153	2023-09-10 10:53:48	IYJ8061	Paypal	Completed	XBV9882
BLL9197	2023-01-08 18:42:49	JYH0715	Visa	Completed	TIU2870
ROF8180	2023-03-12 04:17:25	IRT5508	Visa	Completed	MTO3962
UOO2527	2023-02-05 09:50:38	HQW7234	Visa	Completed	EHM6216
LCN1143	2023-04-20 16:33:46	QZQ7375	Paypal	Completed	XBV9882
IJN6370	2023-08-25 13:10:49	SWK9062	Visa	Completed	LRS8751
ICN5918	2023-03-22 19:43:57	CIP8540	Mastercard	Completed	HGI8774
WQA5165	2023-06-09 13:04:50	AOW9891	Mastercard	Completed	UKA6948
GCK6877	2023-06-09 19:37:56	SJS7468	Paypal	Cancelled	MTO3962
YDU2414	2023-04-05 18:20:52	KBN3694	Visa	Completed	FYK9284
HUQ8737	2023-11-26 15:50:27	LTO1939	Visa	Completed	TIU2870
DNN7162	2023-03-20 19:39:54	OWF1375	Visa	Completed	XBV9882

transactions (continued part 1)

WYM9872	2023-10-19 21:31:15	QXA9511	Visa	Completed	EMC3367
WUP4588	2023-12-30 04:50:02	MPV6830	Visa	Completed	LRS8751
JTE9021	2023-11-18 17:17:09	UJO2256	Visa	Completed	WDI0088
WTF4863	2023-09-28 07:58:44	DYK2145	Paypal	Completed	HUK7721
UZE1858	2023-05-08 01:02:20	JXP6936	Mastercard	Completed	EMC3367
TNP0721	2023-09-26 21:38:56	AGM0946	Mastercard	Completed	HUK7721
VKW5213	2023-10-23 20:17:31	DVW9406	Visa	Completed	HUK7721
PXZ6067	2023-09-09 01:27:03	YKF2765	Mastercard	Completed	YVP0526
ZGW0332	2023-05-21 21:26:14	DUG2232	Visa	Completed	MTO3962
ZTC1290	2023-10-29 12:37:28	EPI2686	Visa	Completed	MQB9199
FUA7216	2023-03-18 10:43:57	DCV2217	Visa	Completed	XBV9882
GDJ1239	2023-01-15 13:47:15	WWK4092	Visa	Completed	MQB9199
AKA4134	2023-08-19 07:32:43	APW5105	Mastercard	Completed	UKA6948
ZND7763	2023-09-02 12:54:39	CZB9315	Visa	Completed	YSC7673
DRW6055	2023-12-28 17:15:59	SVD9310	Mastercard	Completed	TIU2870
WGE4101	2023-01-11 10:25:31	GFY8908	Mastercard	Completed	KBG6195
HZW9649	2023-09-05 19:28:37	EYE9155	Visa	Completed	CMJ2218
YBE9652	2023-10-29 19:01:46	PHE3007	Visa	Completed	HUK7721
BOU8704	2023-02-18 06:55:07	QBP6211	Visa	Completed	FYK9284
PBW8356	2023-01-29 11:52:35	JQV3113	Paypal	Completed	UPL3529
LWK4894	2023-04-03 03:36:10	WBS3190	Paypal	Completed	XBV9882
NRX6499	2023-11-02 02:15:51	ICL2952	Paypal	Completed	UKA6948
RTS9267	2023-08-05 19:20:05	GLO3857	Visa	Completed	XBV9882
XBQ0579	2023-03-23 14:19:41	MNR7324	Visa	Completed	HUK7721
RBX7842	2023-10-22 04:36:10	CPZ7432	Visa	Completed	YSC7673
IZL9895	2023-10-20 23:34:44	HNW8866	Visa	Cancelled	KYM5242
DTE8752	2023-01-04 11:53:27	ORH0891	Mastercard	Completed	YSC7673
RPG2768	2023-01-28 22:19:05	HHE1677	Paypal	Completed	PGY8012
JHN8470	2023-12-13 02:20:45	LOC2874	Mastercard	Completed	EHM6216
WAA9427	2023-05-03 02:11:15	QPM0049	Paypal	Completed	PGY8012
XGX1276	2023-04-20 01:50:07	IVO8127	Visa	Completed	HGI8774
XFE4047	2023-11-14 18:08:04	KTG4876	Visa	Completed	XBV9882
INA6973	2023-06-20 03:32:27	WQY7550	Paypal	Completed	FYK9284

transactions (continued part 2)

IVG6027	2023-06-07 17:05:46	EEP1833	Mastercard	Completed	MQB9199
RXD7441	2023-12-26 05:43:38	VRS5794	Visa	Completed	EMC3367
YEK5339	2023-09-10 16:00:18	BVW5294	Paypal	Completed	XBV9882
LNV6229	2023-07-13 01:51:50	BFW0036	Mastercard	Completed	LRS8751
TFG5794	2023-03-20 13:53:50	DWO0404	Visa	Completed	FYK9284
HUW3782	2023-12-09 05:21:36	BKI6440	Paypal	Completed	TIU2870
RXU2164	2023-07-08 06:48:14	IRF0016	Visa	Completed	HUK7721
ADF2977	2023-08-08 23:12:10	OMX7836	Visa	Cancelled	YVP0526
CNR9928	2023-12-25 09:23:32	VTO8072	Mastercard	Completed	HGI8774
BMZ3281	2023-11-20 16:03:02	IOW5476	Mastercard	Completed	PGY8012
AWE4937	2023-08-26 04:08:08	LOX4355	Mastercard	Completed	YSC7673
JZW3016	2023-05-10 20:31:28	LFZ4494	Mastercard	Completed	UPL3529
UFQ9684	2023-09-27 00:49:54	IWS0251	Mastercard	Completed	WDI0088
IAH7858	2023-03-04 21:30:15	YQM7306	Mastercard	Completed	YSC7673
UWR5419	2023-04-29 11:22:55	AUJ4925	Mastercard	Completed	EHM6216
JQU5343	2023-04-15 05:54:37	CPL0919	Visa	Completed	WDI0088
ZRR4482	2023-06-14 18:52:36	JMV9265	Mastercard	Completed	CMJ2218
IGX3900	2023-03-10 02:18:06	WXP5516	Visa	Completed	MQB9199
QKO9330	2023-08-08 00:43:59	MOA5259	Visa	Completed	EHM6216
LZK5548	2023-04-26 16:51:42	ZVT7472	Paypal	Completed	KYM5242
OCT1961	2023-08-20 10:49:38	ASQ7165	Mastercard	Completed	MQB9199
GIR8112	2023-12-25 12:04:07	JWA5441	Paypal	Completed	YVP0526
RTT8590	2023-01-22 00:29:05	YMN8938	Mastercard	Completed	EHM6216
TTC2622	2023-11-25 14:20:19	GMB4874	Visa	Completed	PGY8012
XWQ1858	2023-05-22 16:46:05	ZEE3650	Visa	Completed	PGY8012
HTP3562	2023-03-27 16:07:39	KHO4228	Visa	Completed	EHM6216
OTX0490	2023-07-31 04:39:33	XYU1488	Visa	Completed	LRS8751
VBL7764	2023-12-27 14:04:21	PNE9540	Paypal	Completed	HGI8774
DSC0165	2023-04-05 02:50:13	YBM7844	Mastercard	Completed	KYM5242
QKE6157	2023-04-29 06:05:27	LJB2637	Visa	Completed	LRS8751
EML3733	2023-10-21 07:18:36	WVL6557	Visa	Completed	KYM5242
OBF6906	2023-04-23 19:53:30	OJA1455	Visa	Completed	LRS8751
HQI8050	2023-02-18 07:43:16	JRL1883	Visa	Completed	YVP0526
HFQ1471	2023-05-08 22:19:41	EXU6344	Visa	Completed	XBV9882
KHE4033	2023-06-10 14:49:16	YJV6348	Paypal	Completed	UPL3529