Online Retail Store

Aayush Ranjan (2021003)
Aishiki Bhattacharya (2021007)

Scope:
Today, due to the extremely fast-paced lives of people, it can be exhausting and time-consuming to visit different stores at different locations. Visiting a market and choosing the right product is cumbersome and may take several hours. Also, due to the recent COVID-19 pandemic, people prefer to get products delivered to their doorstep as quickly as possible.

Therefore, to cater to the needs of the public, we provide complete management solutions to manage information on customers, sellers, delivery agents and products efficiently.

A2Z gives the admins a forum to showcase their products of various categories to customers in a hassle-free and systematic way. Customers can conveniently browse products, manage items in the cart and buy as per their preferred payment method. Delivery agents are assigned orders by the admin, which are supposed to be delivered within the stipulated time to provide an excellent experience to customers. Each of the objects is identified by a unique attribute, i.e., their primary key - Customer_ID, Admin_ID, Delivery_Agent_ID, Order_ID, Product_ID and so on.
This application will enable the admin, delivery agent and customer to interact and coordinate efficiently with each other, resulting in a pleasant experience for all.

TEN SQL QUERIES :-

1) **Give the contact no of delivery person who is delivering an order to the respective customer**

```
SELECT contact FROM agent_contact WHERE agent_id in (SELECT
agent_id FROM delivery_agent WHERE order_id in (SELECT
order_id FROM `order` WHERE customer_id=13));
```

2) **Filter the products according to prices(ascending order)**

```
SELECT * FROM product ORDER BY price;
```

3) **List a particular user's orders belonging to the given "category"**

```
SELECT product_id, name FROM product WHERE product_id IN
(SELECT product_id FROM contains WHERE customer_id=76) AND
category_id=19;
```

4) **Find the customer query status for any given user.**

```
SELECT description, status FROM customer_query WHERE
customer_id=76;
```

5) **Find the order to return ratio for a particular customer.**

```
CREATE VIEW customer_ord AS SELECT COUNT(*) AS orders FROM
`order` INNER JOIN customer ON
`order`.customer_id=customer.customer_id WHERE
customer.customer_id=76;

CREATE VIEW customer_ret AS SELECT COUNT(*) AS returns FROM
`return` INNER JOIN customer ON
customer.customer_id=`return`.customer_id WHERE
customer.customer_id=76;

SELECT * FROM customer_ord NATURAL JOIN customer_ret;
```

6) **Display reviews/ratings of a particular product**

```
SELECT Stars FROM review WHERE Product_Id=75;
```

7) **Display all the products that belong to same category together**

```
SELECT * FROM product LEFT JOIN category USING
(category_id) ORDER BY category.category_id DESC;
```

8) Display the list of most regular customers in our store

```
DROP VIEW frequency;
DROP VIEW maxfrequency;
CREATE VIEW frequency AS select customer_id, COUNT(*) freq
FROM payment GROUP BY customer_id;
CREATE VIEW maxfrequency AS SELECT * FROM frequency WHERE
freq IN (SELECT MAX(freq) FROM frequency);
SELECT * FROM customer INNER JOIN maxfrequency ON
customer.customer_id=maxfrequency.customer_id;
```

9) Resolve all queries of a customer

```
UPDATE customer_query SET status="resolved" WHERE
customer_id=30;
```

10) Display the maximum price of products under a given category

```
SELECT max(price) FROM product GROUP BY category_id;
```

11) Display the list of products under a given category on the basis of increasing price from top to bottom.

```
SELECT product_id, name, price FROM product WHERE
product_id IN (SELECT product_id FROM product WHERE
Category_Id=19) ORDER BY price;
```

12) Display the number of products under each category

```
SELECT COUNT(product_id) AS
Num_of_products_in_each_Category FROM products GROUP BY
category_id
```

13) Display the Customer_Id, Name, Payment_Id, Price, Status for a particular customer

```
SELECT Customer.Customer_Id, Customer.Name,
Payment.Payment_Id, Payment.Price, Payment.Status FROM
Payment INNER JOIN Customer ON
Payment.Customer_id=Customer.Customer_id;
```

14) The number of products of that have been sold belonging each admin

```
SELECT admin_id, count(product_id) FROM supplies WHERE
product_id IN (SELECT contains.product_id FROM contains
INNER JOIN supplies ON
supplies.product_id=contains.product_id WHERE cart_id IN
(SELECT cart_id FROM `order`)) GROUP BY admin_id;
```

15) Display list of all products with their categories that are a part of a cart grouped according to cart_id

```
SELECT * FROM contains INNER JOIN product ON
contains.product_Id=product.product_id INNER JOIN category
USING (category_id) ORDER BY cart_id;
```

RELATIONAL MODEL