DBMS Project Deadline4

REPORT

Writing and executing SQL queries

Siddharth Gupta 2021355

Tony Thomas 2021360

Queries:

1. **Query to authenticate the user and fetch the customer name for welcome message in case authenticated**
   78793 and 'PtSHC6' are the login_id and password entered by the user. If authenticated, it returns the first and last name of the customer for the welcome message. If not, it returns an empty row (this also shows one-one and total relation shared by these two entities)

   ```
   SELECT C.first_name, C.last_name
   FROM Customer C, Authentication_credentials A
   WHERE A.login_id=78793 AND A.password='PtSHC6' AND
   C.customer_id=A.customer_id;
   ```

2. **Query to search a product by name and return all products having that word anywhere in its name (checking for substring)**

   ```
   SELECT P.name, P.description, P.price
   FROM Product P
   WHERE P.name LIKE "%Wine%";
   ```

3. **This query will return all the order items for a given order along with the order_date and order status based on order_id. This will be useful while viewing the order history**

   ```
   SELECT OI.*, O.order_date, O.order_status
   FROM Order_item OI, `Order` O
   WHERE OI.order_id = O.order_id AND O.order_id='ORDER117';
   ```

4. **Query to update the password, here two update queries have been write so as to keep the data unaffected**

```
UPDATE Authentication_credentials A
SET A.password="newPwd"
WHERE A.login_id=78793 AND A.password='PtSHC6' AND
A.customer_id='CUST001';

UPDATE Authentication_credentials A
SET A.password="PtSHC6"
WHERE A.login_id=78793 AND A.password='newPwd' AND
A.customer_id='CUST001';
```

5. **Query to delete cart_item from a customer's cart, to simulate the delete button against a product one has added in one's cart**

```
DELETE FROM cart_item C
WHERE C.cart_id="CART447" AND C.product_id="PROD248";
```

6. **Add a new cart_item to a customer's cart, similar to adding to cart function for products on any online shopping site**

```
INSERT INTO cart_item
(cart_id, product_id, quantity)
VALUES
('CART447', 'PROD248', 35);
```

7. **Query to get all orders with coupon discounts greater than a number**

```
SELECT C.first_name, O.order_id, O.order_date, O.net_amount,
CO.percentage_discount
FROM Customer C, `Order` O, Coupon CO
WHERE C.customer_id = O.customer_id
AND O.coupon_id = CO.coupon_id
AND O.coupon_id
IN (SELECT COU.coupon_id
FROM Coupon COU
WHERE COU.percentage_discount >=20);
```

8. **Increase the quantity of a particular product by 1 in a customer's cart (to simulate the increment done when a customer clicks on the plus button against the cart item in their cart)**

```
UPDATE Cart_item CI INNER JOIN Customer C ON CI.cart_id = C.cart_id
SET CI.quantity = CI.quantity + 1
WHERE CI.product_id = 'PROD124'
AND C.customer_id ='CUST141';

UPDATE Cart_item CI INNER JOIN Customer C ON CI.cart_id = C.cart_id
SET CI.quantity = CI.quantity - 1
WHERE CI.product_id = 'PROD124'
AND C.customer_id ='CUST141';
```

9. **Query to get total amount spent by a customer on the store**

```
SELECT SUM(O.net_amount)
FROM `Order` O, Customer C
WHERE C.customer_id = O.customer_id
AND C.customer_id = 'CUST408';
```

10. **If the admin wants to know a list of all customers who have spent a total amount of money greater than Rs.7000 and sort this in descending order of amount spent.**
The group by command in this query can be used in other ways as well, we could have grouped by pincode to analyse which area is pending the most (to improve delivery services) or least (to advertise or market)

```
SELECT C.customer_id, C.first_name, C.last_name, SUM(O.net_amount)
FROM `Order` O, Customer C
WHERE O.customer_id = C.customer_id
GROUP BY O.customer_id
HAVING SUM(O.net_amount) > 7000
ORDER BY SUM(O.net_amount) desc;
```

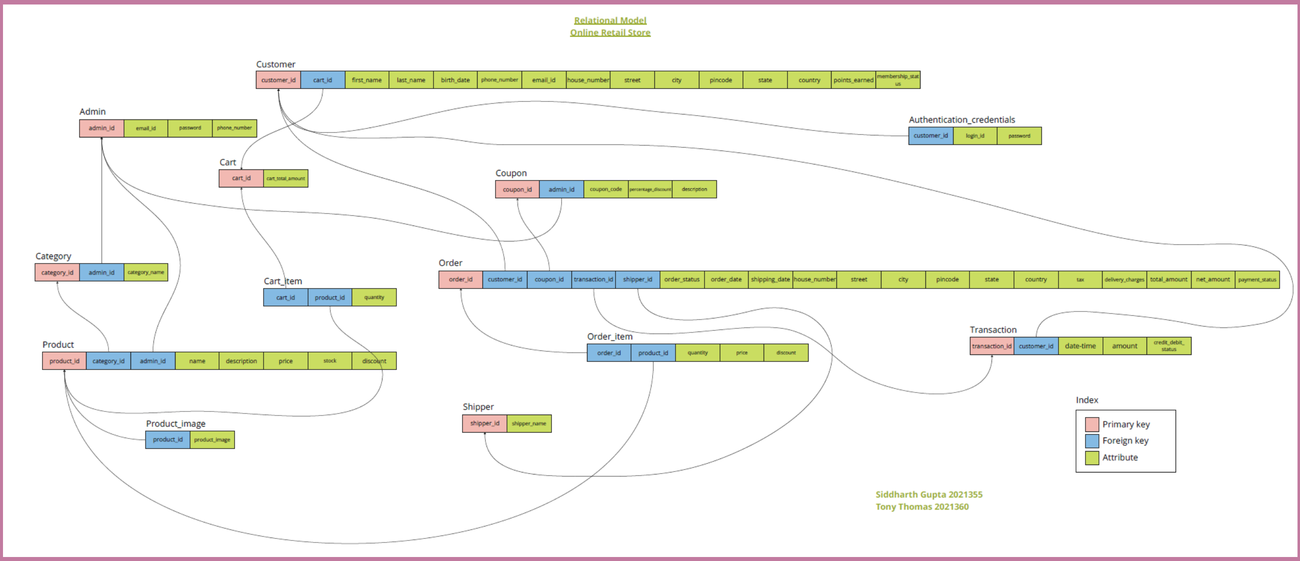
11. **Query to get average amount spent by customers per order and sort it in ascending order**

```
SELECT C.customer_id, C.first_name, C.last_name, AVG(O.net_amount)
FROM `Order` O, Customer C
WHERE O.customer_id = C.customer_id
GROUP BY O.customer_id
ORDER BY AVG(O.net_amount) asc;
```

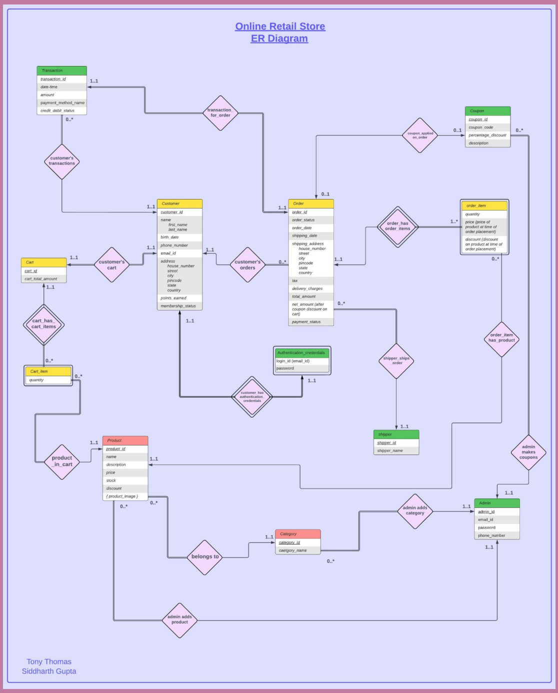
12. **To get the order with maximum amount for the given customer. This customer had 2 orders of amounts 10013 and 8645, so 10013 was returned**

```
SELECT C.customer_id, C.first_name, C.last_name, MAX(O.net_amount)
FROM `Order` O, Customer C
WHERE O.customer_id = C.customer_id AND C.customer_id = 'CUST408';
```

The following show the relational diagram:



Our ER Diagram showing the relations between entities, one-one, many-one, many-many and the strict cardinalities:



Online Retail Store
ER Diagram
Transaction
transaction_ for_order
customer's transactions
Coupon
coupon_applied on_order
Customer
Order
order_has order_items
Order_item
Cart
customer's cart
customer's orders
cart_has_ cart_items
Cart_item
Authentication_credentials
customer_has authentication_ credentials
shipper_ships _order
Shipper
order_item has_product
admin makes coupons
product _in_cart
Product
admin adds category
Admin
Category
belongs to
admin adds product
Tony Thomas
Siddharth Gupta