SQL sample document

Project I

1. Sample code for creating views:

1. OrderDetails

```
CREATE VIEW OrderDetailsView AS
  SELECT
    o.orderID,
    o.userID,
CONCAT(p.productName, ' (', s.sizeName, ', ', c.colorName, ')') AS productDescription,
oi.quantity,
p.price,
(oi.quantity * p.price) AS itemTotal
FROM orderitems oi
JOIN orders o ON oi.orderID = o.orderID
JOIN productvariants pv ON oi.productVariantID = pv.productVariantID
JOIN products p ON pv.productID = p.productID
JOIN sizes s ON pv.sizeID = s.sizeID
JOIN colors c ON pv.colorID = c.colorID
WHERE oi.isDeleted = FALSE;
```

2. MergeOrders

```
CREATE VIEW MergedOrdersView AS
SELECT
o.orderID,
GROUP_CONCAT(oi.quantity) AS quantities,
GROUP_CONCAT(p.name) AS product_names,
GROUP_CONCAT(s.sizeName) AS sizes,
GROUP_CONCAT(pv.color) AS colors, -
SUM(oi.quantity * p.price) AS total_price -
FROM
orders o
JOIN
orderitems oi ON o.orderID = oi.orderID
JOIN
productvariant pv ON oi.productVariantID = pv.productVariantID
JOIN
product p ON pv.productID = p.productID
JOIN
size s ON pv.sizeID = s.sizeID
GROUP BY
o.orderID;
```

3. UserAccountsView

```
CREATE VIEW UserAccountsView AS
SELECT
userID,
firstName,
lastName,
email
FROM Users
WHERE isDeleted = FALSE;
```

4.UserOrderHistoryView

```
CREATE VIEW UserOrderHistoryView AS
SELECT
```

```
o.orderID,
 o.userID,
 o.orderDate,
o.status AS orderStatus,
p.paymentID,
p.paymentMethod AS paymentMethod,
p.paymentStatus AS paymentStatus,
        p.transactionDate,
        a.city,
         a.country
        FROM orders o
        LEFT JOIN payments p ON o.orderID = p.orderID
        LEFT JOIN addresses a ON o.addressID = a.addressID
        WHERE o.isDeleted = FALSE;
```

5. InventoryMonotoringView

```
        CREATE VIEW InventoryMonitoringView AS
         SELECT
         pv.product_variant_id,
         pv.product_id,
         p.product_name,
         pv.variant_name,
         pv.stock_quantity
        FROM
                product_variants pv
        JOIN
                products p ON pv.product_id = p.product_id
        WHERE
```

```
            pv.stock_quantity < 20;
```

2. **Sample code for creating the tables:**

```
CREATE TABLE Users (
  userID INT AUTO_INCREMENT PRIMARY KEY,
  firstName VARCHAR(50) NOT NULL,
  lastName VARCHAR(50) NOT NULL,
  email VARCHAR(100) NOT NULL UNIQUE,
  password VARCHAR(255) NOT NULL,
  isDeleted tinyINT, NOT NULL,
  CHECK (email REGEXP '^[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\\.[A-Za-z]{2,}$')
);

CREATE TABLE Payments (
  paymentID INT PRIMARY KEY AUTO_INCREMENT,
  orderID INT NOT NULL UNIQUE,
  method ENUM('Credit Card', 'Debit Card', 'PayPal', 'Bank Transfer', 'Cash on Delivery') NOT
NULL,
  status ENUM('Pending', 'Completed', 'Failed', 'Refunded') NOT NULL DEFAULT 'Pending',
  transactionDate DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
  FOREIGN KEY (orderID) REFERENCES Orders(orderID)
    ON DELETE CASCADE
    ON UPDATE CASCADE
);
```

3. **Sample code for Relational Algebra:**

```
SELECT * FROM products
WHERE price > 100;
```

```
SELECT productName, price FROM products;

SELECT * FROM colors
CROSS JOIN sizes;

SELECT userID, firstName, lastName FROM users
WHERE userID NOT IN (
  SELECT userID FROM orders
);
```

Sample code for granting permissions to a user:

```
GRANT SELECT ON UserAccountsView TO 'support_user'@'localhost';
GRANT SELECT ON UserOrderHistoryView TO 'support_user'@'localhost';
GRANT SELECT ON OrderDetailsView TO 'support_user'@'localhost';
```

4. **Sample code for a procedure:**

```
IF EXISTS (
    SELECT 1
    FROM orderitems oi
    JOIN orders o ON oi.orderID = o.orderID
    WHERE oi.productVariantID = input_productVariantID
    AND o.status = 'Processing'
) THEN
    -- Raise an error if the product variant is in a processing order
    SIGNAL SQLSTATE '45000'
    SET MESSAGE_TEXT = 'Cannot delete: Product variant is linked to a processing order';
ELSE
```

```
    -- Get the productID of this variant
    SELECT productID INTO productID_var FROM productvariants WHERE productVariantID
= input_productVariantID;


    -- Soft delete the product variant
    UPDATE productvariants
    SET isDeleted = TRUE, stock = 0
    WHERE productVariantID = input_productVariantID;


    -- Check if all variants of this product are now deleted
    IF NOT EXISTS (
        SELECT 1 FROM productvariants
        WHERE productID = productID_var AND isDeleted = FALSE
    ) THEN
        -- If no active variants remain, soft delete the product
        UPDATE products
        SET isDeleted = TRUE
        WHERE productID = productID_var;
    END IF;
END IF;
END
```