Use case-1

Building a cart analysis for Myph

The aim is to enhance consumer selection options for Myph's new phone range through cart analysis, enabling Personalized product recommendations and insights into buying Patterns.

1. Data Model Amendment:

Modify the product data model to include a collection of documents for each category, storing the category path in a category tree.

2. Cart analysis

→ Use Association Rule Mining (like Apriori or FP-Growth) for analyzing consumer selection patterns and developing different selection options.

→ Implement algorithms to detect outlier/surplus selections in the cart, potentially using statistical methods or machine learning techniques for anomaly detection.

3. Relational Database Application:

→ Assess if a relational database can handle transactions for cart analysis, considering aspects like data consistency, ACID properties, and recovery mechanisms.

→ for recovery in carting and commerce, Relational databases use transaction logs and mechanisms like roll back for ensuring data integrity.

1. List all products with category info

```
SELECT p.product_id , p.title, p.price ,
c.category_name
FROM product p
JOIN category c ON p category_id = c.category_id ;
```

2. Find low stock products

```
SELECT title , stock_quantity
FROM product
WHERE stock_quantity <10 ;
```

3 Get all items in a particular user's cart

```
SELECT ci.quantity ,p.title , p price ,
(ci.quantity * p.price) AS total
FROM cartItem ci
JOIN Product p ON ci.product_id = p.product_id
WHERE c.user_id =101 ;
```

4. Find most frequently added products (popular items)

```
SELECT ptitle , SUM (ci.quantity) AS
total_added
FROM cartItem ci
JOIN product p ON ci.product_id = p.product_id
GROUP BY Pproduct_id
ORDER BY total_added DESC
LIMIT 5;
```

Tables structure :

Products : contains product details (id, menu-id, title, description, stock-quantity, pricing-info, category-id)

- categories : contains category details (id, name, path-in-category-tree)
- category-documents : for storing documents related to categories (id, category-id, document-path)

1. Category Table

```
CREATE TABLE category (
      category_id INT PRIMARY KEY
AUTO_INCREMENT,
      category_name VARCHAR (100),
      Parent_category_id INT NULL,
      FOREIGN KEY (parent_category_id)
REFERENCES category (category_id)
);
```

2. product Table

```
CREATE TABLE product (
     Product_id INT PRIMARY KEY
AUTO-INCREMENT,
    title VARCHAR (255),
    description TEXT,
    stock_quantity INT,
    Price DECIMAL (10,2),
    Category_id INT ,
FOREIGN KEY (category-id ) REFERENCES
category(category_id)
 );
```

3 Cart table

```
CREATE TABLE Cart(
    Cart_id INT PRIMARY KEY
AUTO_INCREMENT,
    User_id INT,
    created-at DATETIME DEFAULT
CURRENT_TIMESTAMP
);
```

4. CartItem Table

```
CREATE TABLE cartItem(
    cart_item_id INT PRIMARY KEY
AUTO_INCREMENT ,
    cart_id INT
    Product_id INT,
    quantity INT,
    FOREIGN KEY (cart-id ) REFERENCES
cart (cart_id),
    FOREIGN KEY (product_id ) REFERENCES
Product(product_id)
 );
```

5. Detect outliers - products added in unusually high quantity

```
SELECT p.title, ci.quantity
FROM CartItem ci
JOIN Product p ON ci.product_id = p.product_id
WHERE ci.quantity > (
    SELECT AVG(quantity)+2* STDDEV(quantity) FROM
 CartItem );
```

6. Recover cart data (e.g., abandoned carts)

```
SELECT c.cart_id, c.user_id,
COUNT(ci.cart_item_id) AS items_in_cart
FROM Cart c
LEFT JOIN CartItem ci ON c.cart_id = ci.cart_id
WHERE c.created_at < NOW() - INTERVAL 1
 DAY
GROUP BY c.cart_id;
```