REPORT

☑ 과 목 명 : 데이터베이스응용

☑ 학 과 : 컴퓨터공학과

☑ 담당교수 : 김덕령교수님

☑ 이 름 : 장동민

☑ 제 출 일 : 2023 / 12 / 10

목차

소개

A. 소개

이 프로그램은 쇼핑몰을 운영하기 위한 데이터베이스를 구축하고 관리하는데 사용됩니다. 데이터베이스는 쇼핑몰의 제품, 주문, 고객 등의 정보를 효율적으로 저장하고 관리하기 위해 사용되는 시스템 입니다.

먼저, 쇼핑몰 데이터베이스 테이블 생성 코드는 데이터베이스를 생성하고 테이블을 정의하는 SQL (Structured Query Language) 코드입니다.
이 코드를 사용하여 데이터베이스에 필요한 테이블을 생성하고 각 테이블의 컬럼과 관계를 정의합니다.

예를 들어, "제품" 테이블 에는 제품의 이름, 가격, 재고량 등의 정보를 저장할 수 있습니다.
이 코드는 데이터베이스를 구축할 때 필요한 기본적인 작업을 수행합니다.

다음으로, 논리적 모델링은 데이터베이스의 구조를 정의하는 과정입니다.
쇼핑몰 데이터베이스의 논리적 모델링은 테이블 간의 관계를 설계하고 데이터베이스의 엔티티(개체) 와 속성을 결정하는 작업을 포함합니다.
이 과정에서는 엔티티, 속성, 관계를 시각적으로 표현합니다.

마지막으로, ERD는 개체-관계 다이어그램(Entity-Relationship Diagram)의 약자로, 데이터베이스의 구조와 관계를 시각적으로 표현한 도구입니다.
ERD를 사용하면 데이터베이스의 테이블, 엔티티, 속성, 관계 등을 쉽게 이해할 수 있습니다. ERD는 다양한 기호와 표기법을 사용하여 데이터베이스의 구조를 그림으로 나타내며, 엔티티와 관계의 속성과 제약조건을 표현할 수 있습니다.

이 프로그램은 쇼핑몰 데이터베이스를 구축하고 관리하기 위해
테이블 생성 코드와 논리적 모델링, ERD를 제공합니다.
이를 통해 쇼핑몰 운영에 필요한 데이터를 효율적으로 저장하고 관리할 수 있습니다. 프로그램은 사용자가 원하는 테이블과 속성을 선택하여 자동으로 코드를 생성하고, ERD를 시각적으로 표현하여 데이터베이스 구조를 이해하기 쉽게 도와줍니다.

더 나아가, 이 프로그램은 쇼핑몰 운영에 필요한 다양한 작업을 지원하여 효율적이고 안정적인 데이터베이스 관리를 돕는 것이 이 프로그램의 목표입니다.

요구사항 분석

A. 운영자 관점

1.1 입점 몰의 수 및 정보 관리

쇼핑몰 플랫폼 운영자는 다수의 쇼핑몰을 운영하며,
각 쇼핑몰에 대한 체계적인 정보 관리가 필수입니다.
각 입점 몰은 Mall_ID 및 Mall_Name과 같은 속성으로 식별되며,
데이터베이스에서 관리 대상 쇼핑몰의 수를 정확히 파악합니다.
이를 통해 운영자는 효과적인 몰 관리 및 서비스 제공을 가능하게 합니다.

1.2 사용료 정책 및 결제 처리

운영자는 각 입점 몰에 대한 사용료 정책을 수립하여 투명하게 관리합니다.
이 정책은 판매액을 기반으로 하며, 입점 기간에 따라 다르게 책정됩니다.
Fee_ID, Mall_ID(FK), Fee_Amount 등의 정보를 활용하여,
사용료 청구는 결제 처리 및 정산 과정을 거쳐 이루어집니다.
이로써 입점 몰과의 금전 거래가 효율적으로 이뤄지며, 투명한 금융 거래가
이루어집니다.

1.3 쇼핑몰 정보 관리

운영자는 각 쇼핑몰의 정보를 상세히 기록하여 플랫폼 전반의 효율성을
높입니다. 쇼핑몰 정보에는 쇼핑몰의 이름, 입점일, 연락처, 플랫폼 사용료 등이
포함되어 있습니다. 이 정보를 체계적으로 관리함으로써 운영자는
쇼핑몰들 간의 성과 및 활동을 비교 분석하고,
향후 전략을 계획하는 데에 유용하게 활용할 수 있습니다.

1.4 판매 기록 관리

쇼핑몰 운영자는 각 쇼핑몰의 판매 기록을 철저히 관리하여 비즈니스 성과를 분석합니다. 상품별 월간 판매액, 고객별 월간 판매액, 년간 총 판매액 등의 자료를 효과적으로 기록하여 운영자는 판매 동향을 신속하게 파악하고, 적절한 전략을 수립할 수 있습니다.

1.5 순수 이익금 계산

운영자는 총 매출액에서 플랫폼 사용료를 차감하여 년간 순수 이익금을 계산합니다. 이는 플랫폼 전반의 수익성을 정확히 평가하고, 비즈니스 모델을 최적화하기 위한 기초 자료로 활용됩니다.
이러한 데이터 분석은 장기적인 전략 수립에 큰 도움을 줍니다.

B. 사장 관점

2.1 물품 관리

쇼핑몰 입점 사장은 효과적인 상품 관리를 위해 Product 테이블을 활용합니다. 각 상품의 이름, 가격, 재고 등을 체계적으로 기록하여 입점 몰 내에서의 상품 운영을 원활하게 합니다.

2.2 판매 관리

주문 처리, 배송 관리, 주문 상태 업데이트 등의 판매 관련 업무를 효율적으로 수행하기 위해 Order 테이블을 활용합니다.
이는 Product, User와의 관계를 통해 이뤄집니다.
입점 사장은 이를 통해 주문 처리 및 고객 서비스를 효과적으로 운영할 수 있습니다.

2.3 사용자 관리

회원 정보, 로그인, 주문 이력 등을 체계적으로 관리하기 위해 User 테이블을 사용합니다. 사용자의 정보를 보호하고 효과적으로 서비스를 제공하기 위한 핵심적인 데이터베이스 구조입니다.
이를 통해 입점 사장은 고객과의 관계를 관리하고 향상시킬 수 있습니다.

2.4 재고 관리

입고, 출고, 재고 수량 등을 관리하여 상품의 유효한 관리를 수행합니다.
Inventory 테이블을 통해 각 상품의 입고, 출고 기록 및 재고 현황을 정확하게 기록하여 입점 사장은 상품 공급 및 재고 조절에 대한 최적의 전략을 수립할 수 있습니다.

C. 사용자 관점

3.1 쇼핑 내역

사용자는 자신의 주문 이력과 구매 상세 정보를 조회할 수 있습니다.
Shopping History 테이블을 통해 Order_ID(FK), User_ID(FK), Purchase_Date 등의 정보를 효과적으로 관리하여 사용자는 손쉽게 과거 주문 내역을 확인할 수 있습니다.

3.2 포인트 내역

사용자가 적립한 포인트 정보를 효과적으로 추적하기 위해 Point History 테이블을 활용합니다. 사용자에게 마일리지 혜택을 제공하고 효과적인 포인트 관리를 수행하여 사용자는 자신의 혜택을 쉽게 확인하고 활용할 수 있습니다.

3.3 사용자 구매 및 마일리지 정보

사용자는 자신의 월간 구매액, 구매 이력, 월간 마일리지 자료를 효과적으로 관리합니다. 이는 사용자에게 제공되는 혜택 및 쇼핑 행태에 대한 데이터로 사용자는 개인의 구매 패턴을 파악하고 더 나은 쇼핑 경험을 만들 수 있습니다.

D. 요금 청구

운영자는 입점 쇼핑몰의 판매액을 기반으로 사용료를 청구합니다.
0~1년차는 1년치 판매액 * 0.1%, 1~2년차는 0.2%, 3년차 이상은 1%로 책정됩니다. 이를 통해 운영자는 플랫폼의 성장과 함께 입점 쇼핑몰에게 공정한 사용료를 부과하며, 플랫폼의 지속적인 수익을 보장합니다.

E. 마일리지 관리

사용자의 마일리지를 어느 회사에서 얼마나 쌓였는지와 총 마일리지를 효과적으로 관리합니다. 이는 사용자에게 제공되는 혜택을 투명하게 보여주며, 사용자는 자신의 마일리지 현황을 손쉽게 확인할 수 있습니다.
이를 통해 마일리지 제도는 사용자 유입과 충성도 향상에 기여합니다.

F. 판매 기록 관리

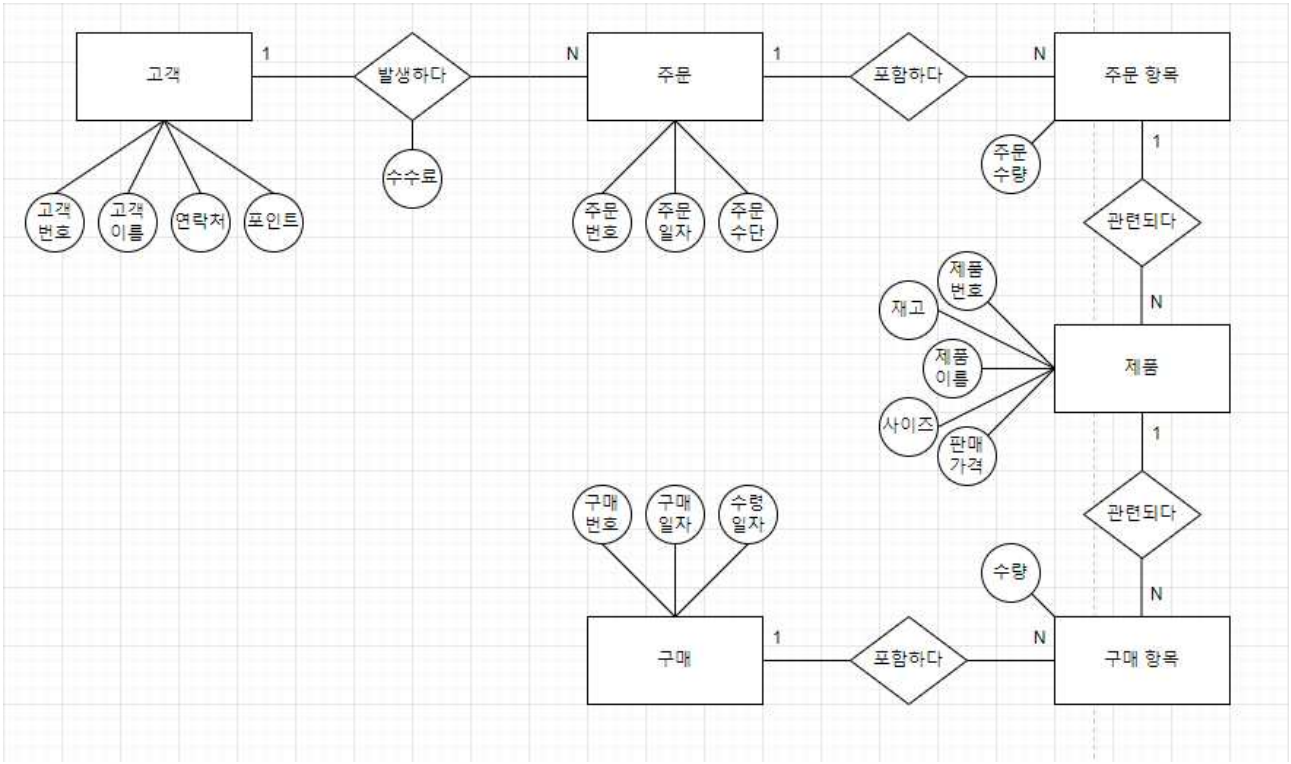
쇼핑몰 사장은 상품별 월간 판매액, 고객별 월간 판매액, 년간 총 판매액의 자료를 기록합니다. 이는 사장에게 특정 상품의 인기와 고객들의 선호도를 파악하는데 도움을 주며,
향후 상품 전략 및 마케팅 계획을 수립하는데 활용됩니다.
판매 기록의 체계적인 관리는 쇼핑몰 사장에게 비즈니스 운영에 대한 인사이트를 제공합니다.

G. 순수 이익금 계산

년간 순수 이익금은 총 매출액에서 플랫폼 사용료를 차감하여 계산됩니다.
이는 쇼핑몰 사장에게 실제로 얼마의 이익을 얻었는지를 정확하게 파악할 수 있는 기준으로 제공되며, 플랫폼 사용료에 대한 투명성을 확보합니다.
이를 통해 사장은 수익성 개선 및 향후 경영 전략을 수립하는 데에 활용합니다.

모델링

A. 개념적 모델링


고객
1
발생하다
N
주문
1
포함하다
N
주문 항목
고객 번호
고객 이름
연락처
포인트
수수료
주문 번호
주문 일자
주문 수단
주문 수량
1
관련되다
N
제품 번호
재고
제품 이름
사이즈
판매 가격
제품
1
관련되다
N
구매 번호
구매 일자
수령 일자
수량
구매
1
포함하다
N
구매 항목

B. 논리적 모델링

1. Mall (쇼핑몰)
   - Mall_ID: INT (주키)
   - Mall_Name: VARCHAR(255) NOT NULL (쇼핑몰 이름)
   - Join_Date: DATE NOT NULL (가입일)
   - Contact_Number: VARCHAR(20) NOT NULL (연락처)

2 Userr (사용자)
   - User_ID: INT (주키)
   - Username: VARCHAR(255) NOT NULL (사용자명)
   - Email: VARCHAR(255) NOT NULL (이메일)

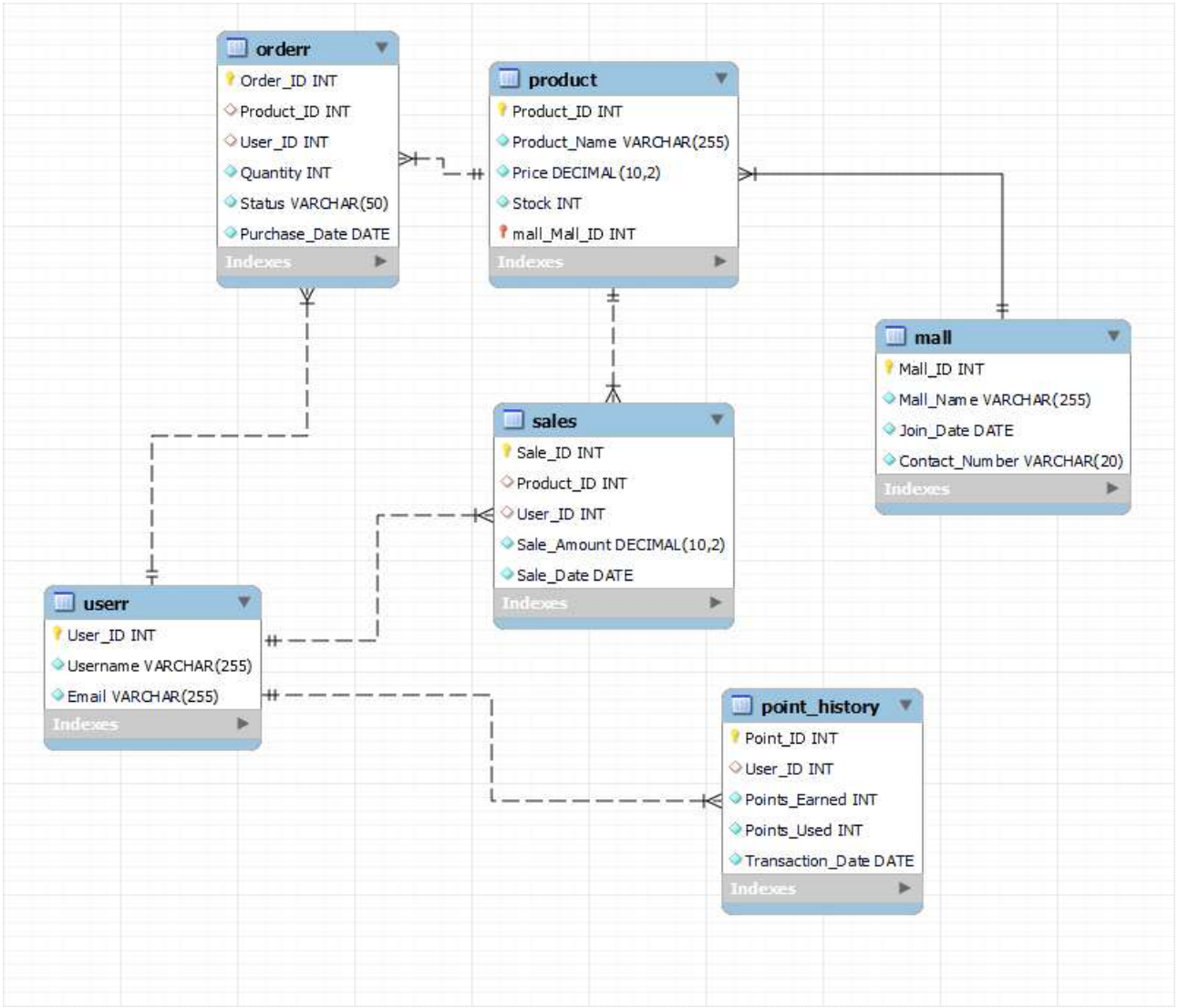
3. Product (상품)
   - Product_ID: INT (주키)
   - Product_Name: VARCHAR(255) NOT NULL (상품명)
   - Price: DECIMAL(10, 2) NOT NULL (가격)
   - Stock: INT NOT NULL (재고)

4. Orderr (주문)
   - Order_ID: INT (주키)
   - Product_ID: INT (외래키 - Product 테이블의 Product_ID 참조)
   - User_ID: INT (외래키 - Userr 테이블의 User_ID 참조)
   - Quantity: INT NOT NULL (수량)
   - Status: VARCHAR(50) NOT NULL (주문 상태)
   - Purchase_Date: DATE NOT NULL (구매일)
   - FOREIGN KEY (Product_ID) REFERENCES Product(Product_ID)
   - FOREIGN KEY (User_ID) REFERENCES Userr(User_ID)

5. Sales (판매)
   - Sale_ID: INT (주키)
   - Product_ID: INT (외래키 - Product 테이블의 Product_ID 참조)
   - User_ID: INT (외래키 - Userr 테이블의 User_ID 참조)
   - Sale_Amount: DECIMAL(10, 2) NOT NULL (판매 금액)
   - Sale_Date: DATE NOT NULL (판매일)
   - FOREIGN KEY (Product_ID) REFERENCES Product(Product_ID)
   - FOREIGN KEY (User_ID) REFERENCES Userr(User_ID)

6. Point_History (포인트 이력)
   - Point_ID: INT (주키)
   - User_ID: INT (외래키 - Userr 테이블의 User_ID 참조)
   - Points_Earned: INT NOT NULL (획득한 포인트)
   - Points_Used: INT NOT NULL (사용한 포인트)
   - Transaction_Date: DATE NOT NULL (거래일)
   - FOREIGN KEY (User_ID) REFERENCES Userr(User_ID)

C. 물리적 모델링


orderr
Order_ID INT
Product_ID INT
User_ID INT
Quantity INT
Status VARCHAR(50)
Purchase_Date DATE
Indexes
product
Product_ID INT
Product_Name VARCHAR(255)
Price DECIMAL(10,2)
Stock INT
mall_Mall_ID INT
Indexes
mall
Mall_ID INT
Mall_Name VARCHAR(255)
Join_Date DATE
Contact_Number VARCHAR(20)
Indexes
sales
Sale_ID INT
Product_ID INT
User_ID INT
Sale_Amount DECIMAL(10,2)
Sale_Date DATE
Indexes
userr
User_ID INT
Username VARCHAR(255)
Email VARCHAR(255)
Indexes
point_history
Point_ID INT
User_ID INT
Points_Earned INT
Points_Used INT
Transaction_Date DATE
Indexes

테이블

A. 테이블 생성 SQL

```
CREATE TABLE Mall (
    Mall_ID INT PRIMARY KEY,
    Mall_Name VARCHAR(255) NOT NULL,
    Join_Date DATE NOT NULL,
    Contact_Number VARCHAR(20) NOT NULL
);
```

```
CREATE TABLE Userr (
    User_ID INT PRIMARY KEY,
    Username VARCHAR(255) NOT NULL,
    Email VARCHAR(255) NOT NULL
);
```

```
CREATE TABLE Product (
    Product_ID INT PRIMARY KEY,
    Product_Name VARCHAR(255) NOT NULL,
    Price DECIMAL(10, 2) NOT NULL,
    Stock INT NOT NULL
);
```

```
CREATE TABLE Orderr (
    Order_ID INT PRIMARY KEY,
    Product_ID INT,
    User_ID INT,
    Quantity INT NOT NULL,
    Status VARCHAR(50) NOT NULL,
    Purchase_Date DATE NOT NULL,
    FOREIGN KEY (Product_ID) REFERENCES Product(Product_ID),
    FOREIGN KEY (User_ID) REFERENCES Userr(User_ID)
);

CREATE TABLE Sales (
    Sale_ID INT PRIMARY KEY,
    Product_ID INT,
    User_ID INT,
    Sale_Amount DECIMAL(10, 2) NOT NULL,
    Sale_Date DATE NOT NULL,
    FOREIGN KEY (Product_ID) REFERENCES Product(Product_ID),
    FOREIGN KEY (User_ID) REFERENCES Userr(User_ID)
);

CREATE TABLE Point_History (
    Point_ID INT PRIMARY KEY,
    User_ID INT,
    Points_Earned INT NOT NULL,
    Points_Used INT NOT NULL,
    Transaction_Date DATE NOT NULL,
    FOREIGN KEY (User_ID) REFERENCES Userr(User_ID)
);
```

B. 코드 설명

Mall 테이블

Mall_ID: 쇼핑몰의 고유 식별자로 사용되는 정수형 필드입니다.
Mall_Name: 쇼핑몰의 이름을 저장하는 문자열 필드입니다.
Join_Date: 쇼핑몰의 가입일을 저장하는 날짜 형식 필드입니다.
Contact_Number: 쇼핑몰의 연락처 번호를 저장하는 문자열 필드입니다.

Userr 테이블

User_ID: 사용자의 고유 식별자로 사용되는 정수형 필드입니다.
Username: 사용자의 이름을 저장하는 문자열 필드입니다.
Email: 사용자의 이메일 주소를 저장하는 문자열 필드입니다.

Product 테이블

Product_ID: 제품의 고유 식별자로 사용되는 정수형 필드입니다.
Product_Name: 제품의 이름을 저장하는 문자열 필드입니다.
Price: 제품의 가격을 저장하는 실수형 필드입니다.
Stock: 제품의 재고량을 저장하는 정수형 필드입니다.

Orderr 테이블

Order_ID: 주문의 고유 식별자로 사용되는 정수형 필드입니다.
Product_ID: 주문된 제품의 식별자를 저장하는 정수형 필드입니다.
Product 테이블의 Product_ID 필드와 외래 키 관계를 가집니다.
User_ID: 주문한 사용자의 식별자를 저장하는 정수형 필드입니다.
Userr 테이블의 User_ID 필드와 외래 키 관계를 가집니다.
Quantity: 주문한 제품의 수량을 저장하는 정수형 필드입니다.
Status: 주문의 상태를 저장하는 문자열 필드입니다.
Purchase_Date: 주문이 발생한 날짜를 저장하는 날짜 형식 필드입니다.

Sales 테이블

Sale_ID: 판매 기록의 고유 식별자로 사용되는 정수형 필드입니다.
Product_ID: 판매된 제품의 식별자를 저장하는 정수형 필드입니다.
Product 테이블의 Product_ID 필드와 외래 키 관계를 가집니다.
User_ID: 판매한 사용자의 식별자를 저장하는 정수형 필드입니다.
Userr 테이블의 User_ID 필드와 외래 키 관계를 가집니다.
Sale_Amount: 판매된 제품의 금액을 저장하는 실수형 필드입니다.
Sale_Date: 판매가 발생한 날짜를 저장하는 날짜 형식 필드입니다.

Point_History 테이블

Point_ID: 포인트 기록의 고유 식별자로 사용되는 정수형 필드입니다.
User_ID: 포인트가 적립되거나 사용된 사용자의 식별자를 저장하는 정수형 필드입니다.
Userr 테이블의 User_ID 필드와 외래 키 관계를 가집니다.
Points_Earned: 적립된 포인트의 양을 저장하는 정수형 필드입니다.
Points_Used: 사용된 포인트의 양을 저장하는 정수형 필드입니다.
Transaction_Date: 포인트 거래가 발생한 날짜를 저장하는 날짜 형식 필드입니다.

예제 데이터

A. 예제 데이터 SQL

쇼핑몰 데이터 삽입

```
INSERT INTO Mall (Mall_ID, Mall_Name, Join_Date, Contact_Number) VALUES
(1, '아디다스 신발', '2021-01-01', '032-123-1311'),
(2, '디올', '2022-02-15', '032-123-1312'),
(3, '오니츠카 타이거', '2023-07-01', '032-123-1313'),
(4, '아레나 수영복', '2021-03-15', '032-123-1314'),
(5, '컨버스', '2021-01-01', '032-123-1315'),
(6, '프로스펙스', '2023-08-15', '032-123-1316'),
(7, '구찌', '2021-11-01', '032-123-1317'),
(8, '켈빈 클라인', '2022-05-15', '032-123-1318'),
(9, '라코스테 핸드백', '2021-02-01', '032-123-1319'),
(10, '다이슨', '2022-12-15', '032-123-1320'),
(11, '필립스', '2021-01-01', '032-123-1321'),
(12, 'LG 전자', '2022-08-15', '032-123-1322');
```

사용자 데이터 삽입

```
INSERT INTO Userr (User_ID, Username, Email) VALUES
(1, '고주현', 'gojuhyun@daum.net'),
(2, '장동민', 'dongmin08@naver.com'),
(3, '윤승민', 'yoon@nate.com');
```

상품 데이터 삽입

```
INSERT INTO Product (Product_ID, Product_Name, Price, Stock) VALUES
(1, '아디다스 그랜드 코트 K 운동화', 99000, 45),
(2, '디올 WALK N DIOR 스니커즈', 235000, 235),
(3, '오니츠카 멕시코 66', 150000, 7),
(4, '아레나 여성 반신 수영복', 60000, 60),
(5, '컨버스 척테일러 올스타', 82000, 144),
(6, '프로스펙스 베어리 씬', 55000, 100),
(7, '구찌 타이거 gg 스몰 토트백', 1480000, 500),
(8, '켈빈 클라인 청바지', 299900, 200),
(9, '라코스테 플랫 핸드백', 599000, 59),
(10, '다이슨 V12 청소기', 135000, 40),
(11, '필립스 에어프라이어 3000', 100000, 5),
(12, 'LG전자 퓨리케어 공기청정기', 350000, 50);
```

판매 데이터 삽입

```
INSERT INTO Sales (Sale_ID, Product_ID, User_ID, Sale_Amount, Sale_Date)
VALUES
(1, 1, 1, 10000, '2022-11-05'),
(2, 2, 1, 15000, '2022-11-05'),
(3, 3, 2, 8000, '2022-11-05'),
(4, 4, 2, 12000, '2022-11-05'),
(5, 5, 3, 5000, '2022-11-06'),
(6, 6, 1, 10000, '2022-12-06'),
(7, 7, 1, 15000, '2022-12-06'),
(8, 8, 2, 8000, '2022-12-07'),
(9, 9, 2, 12000, '2022-12-07'),
(10, 10, 3, 5000, '2022-12-07');
```

마일리지 사용 기록 데이터 삽입

```
INSERT INTO Point_History (Point_ID, User_ID, Points_Earned, Points_Used, Transaction_Date) VALUES
(1, 1, 500, 300, '2022-11-05'),
(2, 1, 700, 400, '2022-11-05'),
(3, 2, 400, 200, '2022-11-05'),
(4, 2, 600, 300, '2022-11-05'),
(5, 3, 200, 100, '2022-11-06'),
(6, 1, 500, 300, '2022-12-06'),
(7, 1, 700, 400, '2022-12-06'),
(8, 2, 400, 200, '2022-12-07'),
(9, 2, 600, 300, '2022-12-07'),
(10, 3, 200, 100, '2022-12-07');
```

주문 데이터 삽입

```
INSERT INTO Orderr (Order_ID, Product_ID, User_ID, Quantity, Status, Purchase_Date) VALUES
(1, 1, 1, 2, 'Completed', '2023-11-05'),
(2, 2, 1, 1, 'Completed', '2023-11-05'),
(3, 3, 2, 3, 'In Progress', '2023-11-05'),
(4, 4, 2, 2, 'In Progress', '2023-11-05'),
(5, 5, 3, 1, 'Completed', '2023-11-06'),
(6, 6, 1, 2, 'Completed', '2023-12-06'),
(7, 7, 1, 1, 'Completed', '2023-12-06'),
(8, 8, 2, 3, 'In Progress', '2023-12-07'),
(9, 9, 2, 2, 'In Progress', '2023-12-07'),
(10, 10, 3, 1, 'Completed', '2023-12-07');
```

요구 사항

A. 요구 사항 SQL

1. 플랫폼 운영자는 입점 쇼핑몰의 1년치 판매액에 대한 플랫폼 사용료를 청구합니다.

입점 0년 ~ 1년차는 1년치 판매액 * 0.1%

입점 1년 ~ 2년차는 1년치 판매액 * 0.2%

입점 3년차 이상은 1년치 판매액 * 1%

```
SELECT
    Mall_ID AS '쇼핑몰 ID',
    Mall_Name AS '쇼핑몰 이름',
    SUM(Sale_Amount) *
        CASE
            WHEN DATEDIFF(CURDATE(), Join_Date) <= 365 THEN 0.001
            WHEN DATEDIFF(CURDATE(), Join_Date) <= 730 THEN 0.002
            ELSE 0.01
        END AS '플랫폼 사용료'
FROM
    Mall
JOIN
    Sales ON Mall.Mall_ID = Sales.Product_ID
GROUP BY
    Mall_ID, Mall_Name;
```

2. 입점한 쇼핑몰의 정보 (이름, 입점일, 연락처, 품목, 누적된 플랫폼 사용료) 가 필요합니다.

```
SELECT
    Mall_Name AS '쇼핑몰 이름',
    Join_Date AS '입점일',
    Contact_Number AS '연락처',
    GROUP_CONCAT(DISTINCT Product_Name) AS '판매 품목',
    SUM(Sale_Amount *
        CASE
            WHEN DATEDIFF(CURDATE(), Join_Date) <= 365 THEN 0.001
            WHEN DATEDIFF(CURDATE(), Join_Date) <= 730 THEN 0.002
            ELSE 0.01
        END) AS '누적 플랫폼 사용료'
FROM
    Mall
JOIN
    Sales ON Mall.Mall_ID = Sales.Product_ID
JOIN
    Product ON Sales.Product_ID = Product.Product_ID
GROUP BY
    Mall_Name, Join_Date, Contact_Number;
```

3. 쇼핑몰 사장은 상품별 월간 판매액의 자료가 필요합니다.

```
SELECT
    Product.Product_ID AS '상품 ID',
    Product.Product_Name AS '상품 이름',
    MONTH(Sales.Sale_Date) AS '월',
    YEAR(Sales.Sale_Date) AS '년도',
    SUM(Sales.Sale_Amount) AS '월간 판매액'
FROM
    Sales
JOIN
    Product ON Sales.Product_ID = Product.Product_ID
GROUP BY
    Product.Product_ID, Product.Product_Name, MONTH(Sales.Sale_Date),
YEAR(Sales.Sale_Date);
```

4. 쇼핑몰 사장은 고객별 월간 판매액의 자료가 필요합니다.

```
SELECT
    Userr.User_ID AS '고객 ID',
    Userr.Username AS '고객 이름',
    MONTH(Sale_Date) AS '월',
    YEAR(Sale_Date) AS '년도',
    SUM(Sale_Amount) AS '월간 판매액'
FROM
    Userr
JOIN
    Sales ON Userr.User_ID = Sales.User_ID
GROUP BY
    Userr.User_ID, Userr.Username, MONTH(Sale_Date), YEAR(Sale_Date)
ORDER BY MONTH(Sale_Date);
```

5. 쇼핑몰 사장은 년간 판매액의 자료가 필요합니다.

```
SELECT
    YEAR(Sale_Date) AS '년도',
    SUM(Sale_Amount) AS '연간 판매액'
FROM
    Sales
GROUP BY
    YEAR(Sale_Date);
```

6. 고객의 마일리지 중 사용한 마일리지와 사용하지 않은 마일리지를 관리합니다.

```
SELECT
    User_ID AS '사용자 ID',
    SUM(Points_Earned) AS '총 적립 마일리지',
    SUM(Points_Used) AS '총 사용 마일리지',
    SUM(Points_Earned - Points_Used) AS '남은 마일리지'
FROM
    Point_History
GROUP BY
    User_ID;
```

7. 연간 순수 이익금은 총 매출액에서 플랫폼 사용료를 차감한 값입니다.

```
SELECT
    YEAR(Sale_Date) AS '년도',
    SUM(Sale_Amount) AS '총 매출',
    SUM(Sale_Amount) -
        SUM(Sale_Amount *
            CASE
                WHEN DATEDIFF(CURDATE(), Join_Date) <= 365 THEN 0.001
                WHEN DATEDIFF(CURDATE(), Join_Date) <= 730 THEN 0.002
                ELSE 0.01
            END) AS '순이익'
FROM
    Sales
JOIN
    Mall ON Sales.Product_ID = Mall.Mall_ID
GROUP BY
    YEAR(Sale_Date);
```

8. 사용자는 자신의 월간 구매액 자료가 필요합니다.

```
SELECT
    YEAR(Purchase_Date) AS '년도',
    MONTH(Purchase_Date) AS '월',
    SUM(Quantity * Price) AS '월간 구매액'
FROM
    Orderr
JOIN
    Product ON Orderr.Product_ID = Product.Product_ID
WHERE
    User_ID = 1
GROUP BY
    YEAR(Purchase_Date), MONTH(Purchase_Date);
```

9. 사용자는 자신의 구매 이력 자료가 필요합니다.

```
SELECT
    Orderr.Order_ID AS '주문 ID',
    Product_Name AS '상품 이름',
    Quantity AS '수량',
    Status AS '주문 상태',
    Purchase_Date AS '구매일'
FROM
    Orderr
JOIN
    Product ON Orderr.Product_ID = Product.Product_ID
WHERE
    User_ID = 1
ORDER BY
    Purchase_Date DESC;
```

10. 사용자는 자신의 월간 마일리지 자료가 필요합니다.

```
SELECT
    User_ID AS '사용자 ID',
    MONTH(Transaction_Date) AS '월',
    YEAR(Transaction_Date) AS '년도',
    SUM(Points_Earned) AS '월간 적립 마일리지',
    SUM(Points_Used) AS '월간 사용 마일리지'
FROM
    Point_History
WHERE
    User_ID = 1
GROUP BY
    User_ID, MONTH(Transaction_Date), YEAR(Transaction_Date);
```

B. 실행 결과

1. 플랫폼 운영자는 입점 쇼핑몰의 1년치 판매액에 대한 플랫폼 사용료를 청구합니다.

   입점 0년 ~ 1년차는 1년치 판매액 * 0.1%

   입점 1년 ~ 2년차는 1년치 판매액 * 0.2%

   입점 3년차 이상은 1년치 판매액 * 1%

	쇼핑몰 ID	쇼핑몰 이름	플랫폼 사용료
▶	1	아디다스 신발	100.00000
	2	디올	30.00000
	3	오니츠카 타이거	8.00000
	4	아레나 수영복	120.00000
	5	컨버스	50.00000
	6	프로스펙스	10.00000
	7	구찌	150.00000
	8	켈빈 클라인	16.00000
	9	라코스테 핸드백	120.00000
	10	다이슨	5.00000

2. 입점한 쇼핑몰의 정보 (이름, 입점일, 연락처, 품목, 누적된 플랫폼 사용료) 가 필요합니다.

	쇼핑몰 이름	입점일	연락처	판매 품목	누적 플랫폼 사용료
▶	구찌	2021-11-01	032-123-1317	구찌 타이거 gg 스몰 토트백	150.00000
	다이슨	2022-12-15	032-123-1320	다이슨 V12 청소기	5.00000
	디올	2022-02-15	032-123-1312	디올 WALK N DIOR 스니커즈	30.00000
	라코스테 핸드백	2021-02-01	032-123-1319	라코스테 플랫 핸드백	120.00000
	아디다스 신발	2021-01-01	032-123-1311	아디다스 그랜드 코트 K 운동화	100.00000
	아레나 수영복	2021-03-15	032-123-1314	아레나 여성 반신 수영복	120.00000
	오니츠카 타이거	2023-07-01	032-123-1313	오니츠카 멕시코 66	8.00000
	컨버스	2021-01-01	032-123-1315	컨버스 척테일러 올스타	50.00000
	켈빈 클라인	2022-05-15	032-123-1318	켈빈 클라인 청바지	16.00000
	프로스펙스	2023-08-15	032-123-1316	프로스펙스 베어리 씬	10.00000

3. 쇼핑몰 사장은 상품별 월간 판매액의 자료가 필요합니다.

	상품 ID	상품 이름	월	년도	월간 판매액
▶	1	아디다스 그랜드 코트 K 운동화	11	2022	10000.00
	2	디올 WALK N DIOR 스니커즈	11	2022	15000.00
	3	오니츠카 멕시코 66	11	2022	8000.00
	4	아레나 여성 반신 수영복	11	2022	12000.00
	5	컨버스 척테일러 올스타	11	2022	5000.00
	6	프로스펙스 베어리 씬	12	2022	10000.00
	7	구찌 타이거 gg 스몰 토트백	12	2022	15000.00
	8	켈빈 클라인 청바지	12	2022	8000.00
	9	라코스테 플랫 핸드백	12	2022	12000.00
	10	다이슨 V12 청소기	12	2022	5000.00

4. 쇼핑몰 사장은 고객별 월간 판매액의 자료가 필요합니다.

	고객 ID	고객 이름	월	년도	월간 판매액
▶	1	고주현	11	2022	25000.00
	2	장동민	11	2022	20000.00
	3	윤승민	11	2022	5000.00
	1	고주현	12	2022	25000.00
	2	장동민	12	2022	20000.00
	3	윤승민	12	2022	5000.00

5. 쇼핑몰 사장은 년간 판매액의 자료가 필요합니다.

	년도	연간 판매액
▶	2022	100000.00

6. 고객의 마일리지 중 사용한 마일리지와 사용하지 않은 마일리지를 관리합니다.

	사용자 ID	총 적립 마일리지	총 사용 마일리지	남은 마일리지
▶	1	2400	1400	1000
	2	2000	1000	1000
	3	400	200	200

7. 연간 순수 이익금은 총 매출액에서 플랫폼 사용료를 차감한 값입니다.

	년도	총 매출	순이익
▶	2022	100000.00	99391.00000

8. 사용자는 자신의 월간 구매액 자료가 필요합니다. (User_ID : 1)

	년도	월	월간 구매액
▶	2023	11	433000.00
	2023	12	1590000.00

9. 사용자는 자신의 구매 이력 자료가 필요합니다. (User_ID : 1)

	주문 ID	상품 이름	수량	주문 상태	구매일
▶	6	프로스펙스 베어리 씬	2	Completed	2023-12-06
	7	구찌 타이거 gg 스몰 토트백	1	Completed	2023-12-06
	1	아디다스 그랜드 코트 K 운동화	2	Completed	2023-11-05
	2	디올 WALK N DIOR 스니커즈	1	Completed	2023-11-05

10. 사용자는 자신의 월간 마일리지 자료가 필요합니다. (User_ID : 1)

	사용자 ID	월	년도	월간 적립 마일리지	월간 사용 마일리지
▶	1	11	2022	1200	700
	1	12	2022	1200	700

프로시저

A. 프로시저 SQL

1. 플랫폼 운영자는 입점 쇼핑몰의 1년치 판매액에 대한 플랫폼 사용료를 청구합니다.

입점 0년 ~ 1년차는 1년치 판매액 * 0.1%

입점 1년 ~ 2년차는 1년치 판매액 * 0.2%

입점 3년차 이상은 1년치 판매액 * 1%

```
    DELIMITER //
CREATE PROCEDURE GetPlatformFees()
BEGIN
    SELECT
        Mall_ID AS '쇼핑몰 ID',
        Mall_Name AS '쇼핑몰 이름',
        SUM(Sale_Amount) *
            CASE
                WHEN DATEDIFF(CURDATE(), Join_Date) <= 365 THEN 0.001
                WHEN DATEDIFF(CURDATE(), Join_Date) <= 730 THEN 0.002
                ELSE 0.01
            END AS '플랫폼 사용료'
    FROM
        Mall
    JOIN
        Sales ON Mall.Mall_ID = Sales.Product_ID
    GROUP BY
        Mall_ID, Mall_Name;
END //
```

2. 입점한 쇼핑몰의 정보 (이름, 입점일, 연락처, 품목, 누적된 플랫폼 사용료) 가 필요합니다.

```
    DELIMITER //
CREATE PROCEDURE GetMallInformation()
BEGIN
    SELECT
        Mall_Name AS '쇼핑몰 이름',
        Join_Date AS '입점일',
        Contact_Number AS '연락처',
        GROUP_CONCAT(DISTINCT Product_Name) AS '판매 품목',
        SUM(Sale_Amount *
            CASE
                WHEN DATEDIFF(CURDATE(), Join_Date) <= 365 THEN 0.001
                WHEN DATEDIFF(CURDATE(), Join_Date) <= 730 THEN 0.002
                ELSE 0.01
            END) AS '누적 플랫폼 사용료'
    FROM
        Mall
    JOIN
        Sales ON Mall.Mall_ID = Sales.Product_ID
    JOIN
        Product ON Sales.Product_ID = Product.Product_ID
    GROUP BY
        Mall_Name, Join_Date, Contact_Number;
END //
```

3. 쇼핑몰 사장은 상품별 월간 판매액의 자료가 필요합니다.

```
    DELIMITER //
CREATE PROCEDURE GetMonthlySalesByProduct()
BEGIN
    SELECT
        Product.Product_ID AS '상품 ID',
        Product.Product_Name AS '상품 이름',
        MONTH(Sales.Sale_Date) AS '월',
        YEAR(Sales.Sale_Date) AS '년도',
        SUM(Sales.Sale_Amount) AS '월간 판매액'
    FROM
        Sales
    JOIN
        Product ON Sales.Product_ID = Product.Product_ID
    GROUP BY
        Product.Product_ID, Product.Product_Name, MONTH(Sales.Sale_Date),
YEAR(Sales.Sale_Date);
END //
```

4. 쇼핑몰 사장은 고객별 월간 판매액의 자료가 필요합니다.

```
    DELIMITER //
CREATE PROCEDURE GetMonthlySalesByUser()
BEGIN
    SELECT
        Userr.User_ID AS '고객 ID',
        Userr.Username AS '고객 이름',
        MONTH(Sale_Date) AS '월',
        YEAR(Sale_Date) AS '년도',
        SUM(Sale_Amount) AS '월간 판매액'
    FROM
        Userr
    JOIN
        Sales ON Userr.User_ID = Sales.User_ID
    GROUP BY
        Userr.User_ID, Userr.Username, MONTH(Sale_Date), YEAR(Sale_Date)
ORDER BY MONTH(Sale_Date);
END //
```

5. 쇼핑몰 사장은 년간 판매액의 자료가 필요합니다.

```
    DELIMITER //
CREATE PROCEDURE GetYearlySales()
BEGIN
    SELECT
        YEAR(Sale_Date) AS '년도',
        SUM(Sale_Amount) AS '연간 판매액'
    FROM
        Sales
    GROUP BY
        YEAR(Sale_Date);
END //
```

6. 고객의 마일리지 중 사용한 마일리지와 사용하지 않은 마일리지를 관리합니다.

```
    DELIMITER //
CREATE PROCEDURE GetPointsManagement()
BEGIN
    SELECT
        User_ID AS '사용자 ID',
        SUM(Points_Earned) AS '총 적립 마일리지',
        SUM(Points_Used) AS '총 사용 마일리지',
        SUM(Points_Earned - Points_Used) AS '남은 마일리지'
    FROM
        Point_History
    GROUP BY
        User_ID;
END //
```

7. 연간 순수 이익금은 총 매출액에서 플랫폼 사용료를 차감한 값입니다.

```
    DELIMITER //
CREATE PROCEDURE GetAnnualNetProfits()
BEGIN
    SELECT
        YEAR(Sale_Date) AS '년도',
        SUM(Sale_Amount) AS '총 매출',
        SUM(Sale_Amount) -
            SUM(Sale_Amount *
                CASE
                    WHEN DATEDIFF(CURDATE(), Join_Date) <= 365 THEN
0.001
                    WHEN DATEDIFF(CURDATE(), Join_Date) <= 730 THEN
0.002
                    ELSE 0.01
                END) AS '순이익'
    FROM
        Sales
    JOIN
        Mall ON Sales.Product_ID = Mall.Mall_ID
    GROUP BY
        YEAR(Sale_Date);
END //
```

8. 사용자는 자신의 월간 구매액 자료가 필요합니다.

```
        DELIMITER //
CREATE PROCEDURE GetUserMonthlyPurchaseAmount(IN userID INT)
BEGIN
    SELECT
        YEAR(Purchase_Date) AS '년도',
        MONTH(Purchase_Date) AS '월',
        SUM(Quantity * Price) AS '월간 구매액'
    FROM
        Orderr
    JOIN
        Product ON Orderr.Product_ID = Product.Product_ID
    WHERE
        User_ID = userID
    GROUP BY
        YEAR(Purchase_Date), MONTH(Purchase_Date);
END //
DELIMITER ;
```

9. 사용자는 자신의 구매 이력 자료가 필요합니다.

```
    DELIMITER //
CREATE PROCEDURE GetUserPurchaseHistory()
BEGIN
    SELECT
        Orderr.Order_ID AS '주문 ID',
        Product_Name AS '상품 이름',
        Quantity AS '수량',
        Status AS '주문 상태',
        Purchase_Date AS '구매일'
    FROM
        Orderr
    JOIN
        Product ON Orderr.Product_ID = Product.Product_ID
    WHERE
        User_ID = 1
    ORDER BY
        Purchase_Date DESC;
END //
```

10. 사용자는 자신의 월간 마일리지 자료가 필요합니다.

```
    DELIMITER //
CREATE PROCEDURE GetMonthlyPointsByUser()
BEGIN
    SELECT
        User_ID AS '사용자 ID',
        MONTH(Transaction_Date) AS '월',
        YEAR(Transaction_Date) AS '년도',
        SUM(Points_Earned) AS '월간 적립 마일리지',
        SUM(Points_Used) AS '월간 사용 마일리지'
    FROM
        Point_History
    WHERE
        User_ID = 1
    GROUP BY
        User_ID, MONTH(Transaction_Date), YEAR(Transaction_Date);
END //
```

```
-- 1. 플랫폼 사용료 조회 프로시저 호출
CALL GetPlatformFees();

-- 2. 입점한 쇼핑몰 정보 조회 프로시저 호출
CALL GetMallInformation();

-- 3. 쇼핑몰 상품별 월간 판매액 조회 프로시저 호출
CALL GetMonthlySalesByProduct();

-- 4. 사용자별 월간 판매액 조회 프로시저 호출
CALL GetMonthlySalesByUser();

-- 5. 년간 판매액 조회 프로시저 호출
CALL GetYearlySales();

-- 6. 마일리지 관리 조회 프로시저 호출
CALL GetPointsManagement();

-- 7. 연간 순수 이익금 조회 프로시저 호출
CALL GetAnnualNetProfits();

-- 8. 사용자의 월간 마일리지 조회 프로시저 호출 (사용자 ID = 1)
CALL GetUserMonthlyPurchaseAmount(1);

-- 9. 사용자의 구매 이력 조회 프로시저 호출 (사용자 ID = 1)
CALL GetUserPurchaseHistory();

-- 10. 사용자의 월간 마일리지 조회 프로시저 호출 (사용자 ID = 1)
CALL GetMonthlyPointsByUser();
```

소감문

A. 소감문

이 프로젝트는 저에게 많은 도전과 성취감을 안겨주었으며,
저의 역량을 향상시키는 좋은 기회였습니다.
이렇게 길게 쓰는 것이 조금 부담스러울 수 있겠지만,
저의 소감을 성심성의껏 전해드리기 위해 최선을 다하겠습니다.

프로젝트 진행 중에는 다양한 어려움과 문제가 있었습니다.
데이터 충돌, 오류 등등 다양한 상황에서 해결책을 찾기 위해 노력하였습니다.
이런 경험을 통해 팀워크의 중요성과 협업의 가치를 깨닫게 되었습니다.

프로젝트를 진행하면서 가장 큰 보람은 완성된 시스템을 사용해보는 순간이었습니다.
이전에는 수작업으로 처리해야 했던 많은 업무들이 자동화되어
효율적으로 처리될 수 있는 지경에 이르렀습니다.
또한, 데이터베이스를 통해 다양한 분석과 통계를 수행하여
쇼핑몰의 판매 동향과 고객 행태를 파악할 수 있었습니다.
이를 통해 쇼핑몰 운영에 대한 통찰력을 얻을 수 있었고,
더 나은 전략을 수립할 수 있었습니다.

마지막으로, 이 프로젝트를 통해 저는 데이터베이스 관리와 개발에 대한
깊은 이해를 얻었습니다.
이러한 경험은 제 미래에 큰 도움이 될 것이라고 믿습니다.
또한, 동료들과의 협업과 소통 능력, 문제 해결 능력 등
다양한 역량도 향상 시킬 수 있었습니다.

이 프로젝트를 진행하면서 많은 어려움과 고난을 겪었지만,
그만큼 더 큰 성취감과 보람을 느낄 수 있었습니다.
이 프로젝트를 통해 새로운 도전에 대한 용기와 자신감을 얻을 수 있었습니다.
앞으로도 더 많은 프로젝트를 수행하고 발전해 나갈 것입니다. 감사합니다.