이커머스 데이터베이스 성능 최적화 보고서

이커머스 데이터베이스 성능 최적화 보고서

※소스에 쿼리구현은 전부 이루어지지 않았으며 필요하다고 판단한 부분에 대한 처리

1. 프로젝트 개요

- **환경**: Spring Boot + MySQL
- **분석 대상**: 이커머스 데이터베이스 쿼리 성능
- **목적**: 조회 성능 저하 요인 분석 및 최적화 방안 제시

2. 성능 이슈 식별

2.1 상품 타입, 가격 필터링 조회

문제점:

- 검색 시 Full Table Scan 발생
- 카테고리별 필터링 성능 저하

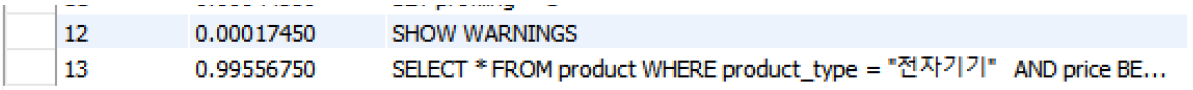
현재 쿼리:

```
SELECT * FROM product
WHERE product_type = "전자기기"
  AND price BETWEEN 100000 AND 500000
```

쿼리 플랜

실행시간 측정

```
-- 실행 시간 측정
SET profiling = 1;
SELECT * FROM product
WHERE product_type = "전자기기"
  AND price BETWEEN 100000 AND 500000
LIMIT 1000000 OFFSET 0;
SHOW PROFILES;
```



성능 분석 결과:

- 실행 시간: 0.995초 (100만 건 기준)

2.2 주문 내역 조회

문제점:

- 사용자별 주문 내역 조회 시 조인 비용 과다
- 날짜 범위 검색 최적화 부족

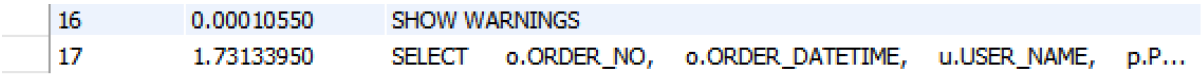
현재 쿼리:

```
SELECT
    o.ORDER_NO,
    o.ORDER_DATETIME,
    u.USER_NAME,
    p.PRODUCT_NAME,
```

```
    o.ORIGINAL_PRICE,
    o.DISCOUNTED_PRICE,
    o.ORDER_STATUS
FROM ORDERS o
JOIN USERS u ON o.USER_ID = u.USER_ID
JOIN PRODUCT p ON o.PRODUCT_ID = p.PRODUCT_ID
WHERE o.ORDER_DATETIME BETWEEN '2025-01-01 00:00:00' AND '2025-07-29 23:59:59'
ORDER BY o.ORDER_DATETIME DESC
```

쿼리 플랜

실행시간 측정



성능 분석 결과:

- 실행 시간: 1.73초(주문100만, 상품100만 기준)
- 조인된 테이블 수: 3개
- 인덱스 활용도: 낮음

3. 최적화 방안

3.1 인덱스 설계

복합 인덱스 생성

```
// 상품조회용 인덱스
CREATE INDEX idx_product_type_price ON product (product_type, price);

// 주문내역조회 인덱스
CREATE INDEX idx_orders_datetime_user_product ON ORDERS (ORDER_DA
TETIME DESC, USER_ID, PRODUCT_ID);
```

인덱스 선택 기준

- **선택도(Selectivity)**: 높은 선택도를 가진 컬럼 우선
- **카디널리티**: 고유값이 많은 컬럼을 인덱스 앞쪽에 배치
- **쿼리 패턴**: WHERE, ORDER BY, JOIN 조건 고려

인덱스 사용 후

실행시간 측정

	Query_ID	Duration	Query
▶	1	0.00014075	SHOW WARNINGS
	2	0.04374050	SELECT * FROM product WHERE product_type = "전자기기" AND price BE...

3.2 쿼리 재설계

주문내역조회

```
SELECT
    o.ORDER_NO,
    o.ORDER_DATETIME,
    u.USER_NAME,
    p.PRODUCT_NAME,
    o.ORIGINAL_PRICE,
    o.DISCOUNTED_PRICE,
    o.ORDER_STATUS
FROM ORDERS o
JOIN USERS u ON o.USER_ID = u.USER_ID
JOIN PRODUCT p ON o.PRODUCT_ID = p.PRODUCT_ID
WHERE o.ORDER_DATETIME BETWEEN '2025-01-01 00:00:00' AND '2025-01-31 23:59:59'
ORDER BY o.ORDER_DATETIME DESC;
```

- 기존 쿼리는 ORDERS 테이블에서 FK, USERS와 PRODUCT 테이블은 PK 인덱스를 사용하여 데이터 조회
- DATETIME을 단일인덱스로, ORDER_ID, USER_ID와 복합인덱스로 사용하여 인덱스를 타게끔 쿼리를 수정하고 힌트를 주어도 기존 쿼리보다 성능이 훨씬 나오지 않음
- 페이징에서도 속도개선을 체감하지 못함
- 결국 DATETIME 조회시간을 줄이는 것으로 타협

4. 결론

이커머스 데이터베이스의 주요 성능 병목점을 식별하고 개선하였습니다. 결과적으로 인덱스 사용과 간단한 쿼리수정정도에서 그쳤지만 DB 성능개선 학습을 하며 파티셔닝, 캐싱처리 등 다양한 방식으로도 성능개선을 이뤄낼 수 있다는 것을 알았습니다. 이번에는 구현하지 못했지만 추후 학습하여 사용해보고 싶었습니다.