

목차

1. ERD 고려점

- a. 통합 및 확장성 관리
- b. 분류 및 코드 관리
- c. 구현 및 설명

2. SQL 고려점

- a. 데이터 생성 코드
- b. 상품 추천조회 고려
 - i. 추천조회 순위 고려
 - ii. 데이터 변화량(델타) 고려
 - iii. 델타의 변화량 고려
- c. 데이터 추천조회 코드

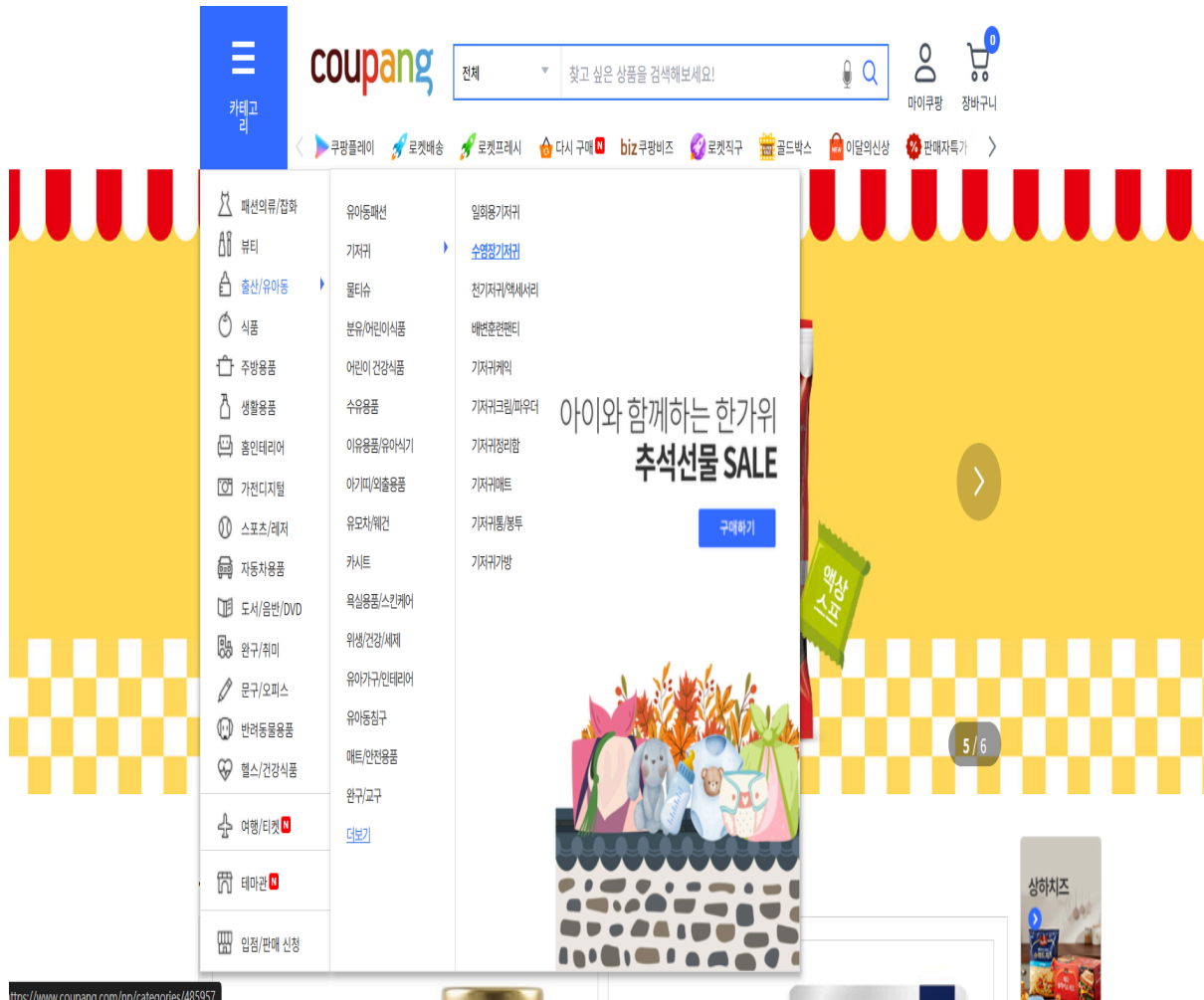
1-a. 통합 및 확장성 관리

설명 : 본인이 한 프로젝트는 책 관련 데이터베이스입니다. 당연히 주로 책에 관한 테이블이 있습니다.

근데 사업이나 회사를 운영하다 보면 사업이나 서비스를 확대해야 하는 경우도 있습니다. 이런 확장같은 경우에는 데이터베이스를 따로 구축해서 운영하면 되지 않느냐라고 생각할 수 있습니다.

하지만, 따로 구축해서 운영하는 것도 비용이고 확장 서비스가 잘 되리라는 보장이 없습니다. 코에 걸면 코걸이 귀에 걸면 귀걸이라는 말이 있듯이 책관련 데이터베이스를 설계하는게 아닌 포괄적인 제품 데이터베이스를 설계하는 관점으로 한번 만들어 보았습니다. 우선 통합적으로 관리하되 분리할 필요가 발생시 따로 분리해서 운영하는게 좋지 않을까 생각이 들었습니다. 이러한 상황을 가정하여 한번 만들어 보았습니다.

1-b. 분류 및 코드 관리



설명 : 분류는 위와 같이 쿠팡을 참조하였습니다. 분류 및 분리는 기본적으로 완벽하게 할 수는 없습니다. 계속해서 다른 형태의 신제품같은게 나올 여지가 있기 때문입니다.

1-c. ERD 구현 및 설명

ERD 설명 : 주로 주문, 상품, 서비스등에 대한 **ERD**로 구성하였습니다.

ERD 주소 : <https://www.erdcloud.com/d/LCNq3MhjRTXCgBW9R>

- **상태 및 요청등** : 코드성 테이블입니다. 특정 여러 테이블에 대한 요청정보나 상태정보를 저장하기 위해 만들었습니다.
1개의 테이블로도 할 수 있으나 1개 테이블로 관리하면 이러한 코드들 또한 여러 개로 분리 될 수 있으므로 2개 정도로 분리하였습니다.

예를 들어, 주문 실패와 같은 경우를 생각하면 주문이 실패했을 상태는 다시 주문실패/연결실패, 주문실패/잔액부족, 주문실패/재고부족 등과 같은 상황이 발생할 수 있기에 이렇게 하였습니다.

- **상품 환불및반품** : 주문상품의 상태코드를 통해 판별하도록 하였습니다. 환불 및 반품과 같은 테이블을 추가하지 않고도 상태코드등을 이용해 이러한 상태를 추적하도록 만들어 보았습니다.
- **상품카테고리** : 대분류, 중분류, 소분류 테이블로 분리하였습니다.
- **서비스** : A/S나 협력업체 서비스등을 고려하여 만들어 보았습니다.
- **주문상품배송** : 본인이 알기에는(잘못 알고 있을수도 있습니다.) 복합인덱스를 설정하면 첫번째, 두번째, ... 이런식으로 필터링이 된다고 알고 있기에 **PK**인덱스를 보면 주문상품배송지를 **first index**로 했는데 같은 혹은 비슷한 배송지에 대해 데이터를 먼저 필터링 해서 보여주는게 좋다고 생각을 했습니다.

2-a. 데이터 생성코드

설명 : **init market.sql** 참고하시면 됩니다. 처음부터 끝까지 실행하면 됩니다.

2-b-i. 추천조회 순위 고려

설명 : 상품카테고리 클릭 했을 때 상품이 추천조회되는 경우를 한번 생각해보았습니다. 우선 어떻게 상품을 보여줘야 하는지에 대해서 설명을 하겠습니다.

방법1) 최신순 : 상품을 최신순으로 보여주게 되면 잘 팔리는 상품을 보여주는게 곤란합니다.

방법2) 평점 및 추천순 : 이 방법이 일반적인 방법이지만, 사용자의 추천수에 좌지우지 되는 경향이 있고 정말 좋아서 추천을 했는지 아니면 싫어서 추천을 했는지등에 대한 여부를 판단할 수가 없습니다. 게다가 블랙 컨슈머와 같이 점수에 일부러 조작을 할 수도 있기 때문입니다. 즉 상대적인 지표로 생각이 듭니다.

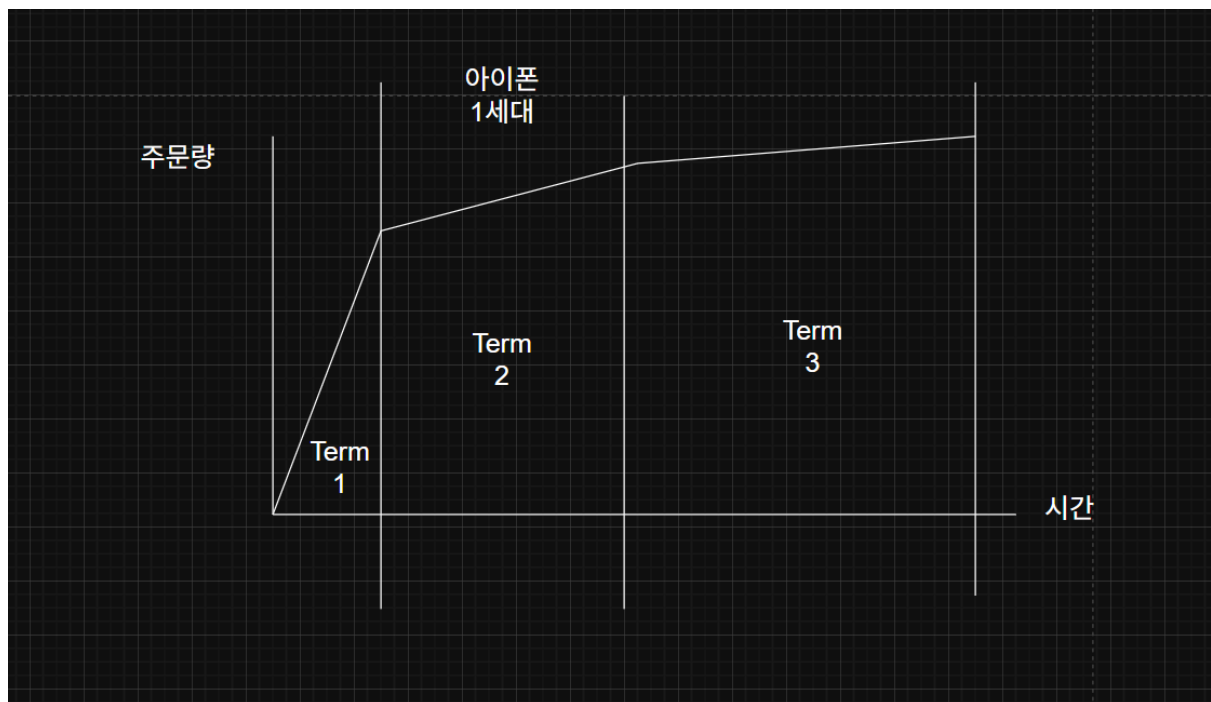
방법3) 판매량 및 주문량순 : 방법2의 상대적인 문제점을 보완하기 위해 절대적인 지표인 판매량 및 주문량순으로 표시를 하는 방법입니다. 하지만 이 방법에 대해 조금 더 자세히 알아볼 필요가 있다 생각이 듭니다.

***그래프 모양 설명** : 어디까지나 본인이 생각한 대략적인 그래프 모양이므로 양해바랍니다.

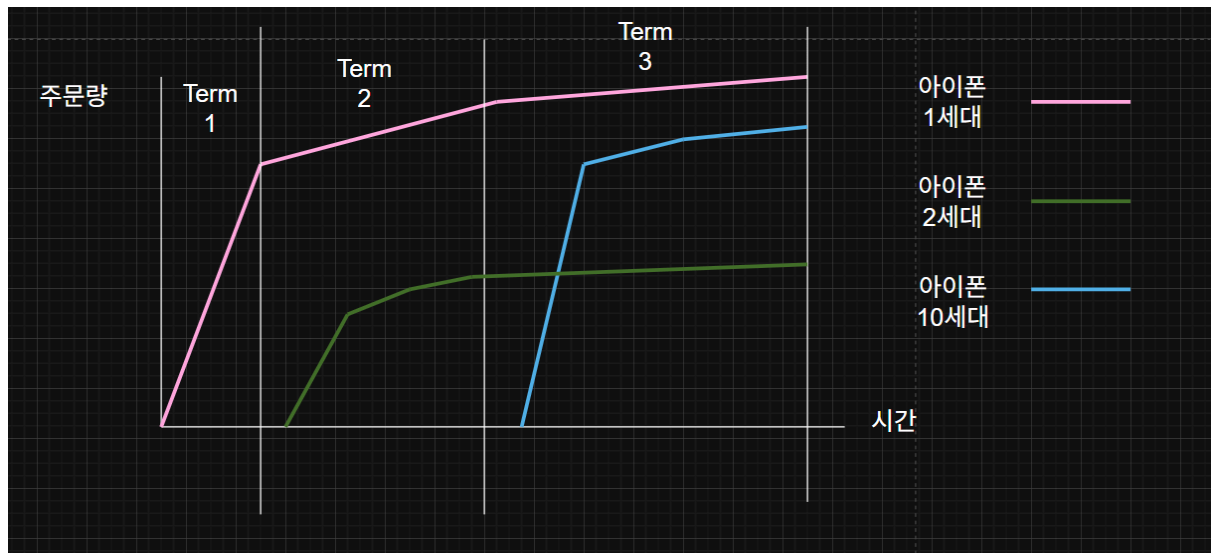
***주문량 설명** : 주문테이블에 있는 주문량을 의미합니다. 주문데이터가 삭제되지 않는다는 가정하에 설명을 하였습니다.

2-b-ii. 데이터 변화량 고려

아이폰 1세대의 주문량 및 판매량 : 아이폰 1세대는 최초의 상용 스마트폰이라 할 수 있습니다. 이 제품의 주문량이 어떨까?라고 생각하였을 때, 최초적이고 그리고 혁신적인 제품이라 갤럭시와 같은 다른 대체재 또한 없다 생각이 듭니다. 당연히 엄청 팔렸을 것이라 생각이 듭니다. 이를 그래프로 보면 다음과 같습니다.



설명 : 위의 그래프의 특징은 주문량 자체는 크지만 주문량의 변화량(이하 델타라 하겠습니다.)이 시간이 지남에 따라 감소하게 됩니다. 그럼 이제 위의 그래프에 본인이 생각하는 아이폰 **10**세대, **2**세대를 포함한 그래프는 다음과 같습니다.



설명 : 위의 그래프를 통해 알 수 있는 점이 몇가지가 있는데, 만약 단순 주문량에 대해 해당 제품을 추천조회에 올리게 되면 아이폰 **1**세대가 계속 노출이 될것 입니다. 근데 아이폰 **1**세대를 사용하는 사람은 거의 없습니다. 이것 방지하기 위해 특정 유효기간에서 특정 델타값 미만으로 떨어지면 해당 제품은 추천대상에서 제외가 되도록 하는게 좋다 판단이 됩니다. 이유는 다음과 같습니다.

특정 델타값 설정 이유 : 델타값 = 주문 변화량값 = 해당제품 기호도 = 해당 제품 살 확률 이므로 아이폰 **1**세대와 같은 구시대적인 제품을 걸러내기 위해 필요하다 생각합니다.

특정 유효기간 설정 이유 : 특정 유효기간 설정이 없어도 델타값은 감소하게 되어있습니다. 그런데 이 델타값이 너무 느리게 감소하게 됩니다.

왜냐하면 위의 그래프를 보면 알 수 있듯이, **TERM 3**의 아이폰 **1**세대 주문 변화량 값과 영점에서부터 최신 주문량 좌표점의 변화량(기울기)는 너무나 상이하기 때문입니다. 아이폰 **1**세대의 **TERM 3** 기울기는 작지만, 처음부터 최신까지의 기울기는 여전히 어느정도 크기 때문입니다.

그래서 유효기간 설정이 없으면 아이폰 **1**세대와 같은 제품들은 장기간동안 델타값을 충족하게 되므로 계속해서 보여지게 될거라 생각이 듭니다.

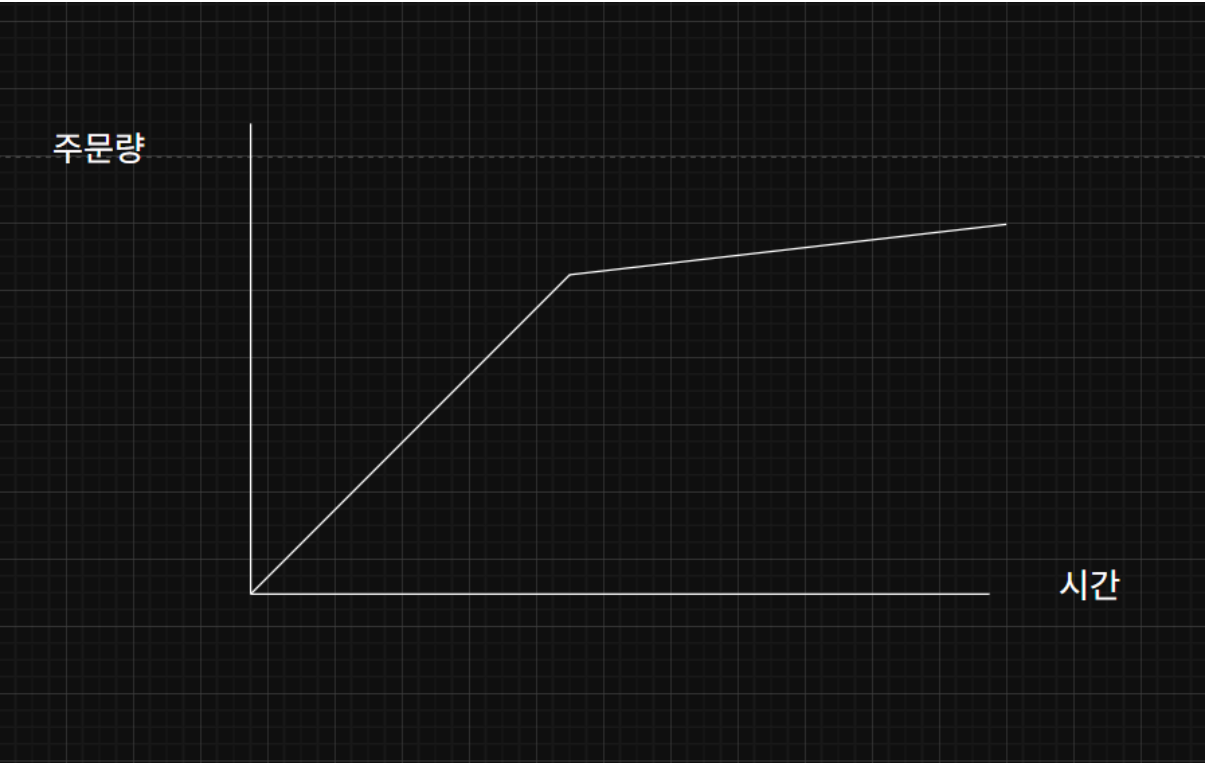
2-b-iii. 델타의 변화량 고려

설명 : 델타의 변화량에 대해 다음의 **2**가지 상황에 대해 설명을 하겠습니다.

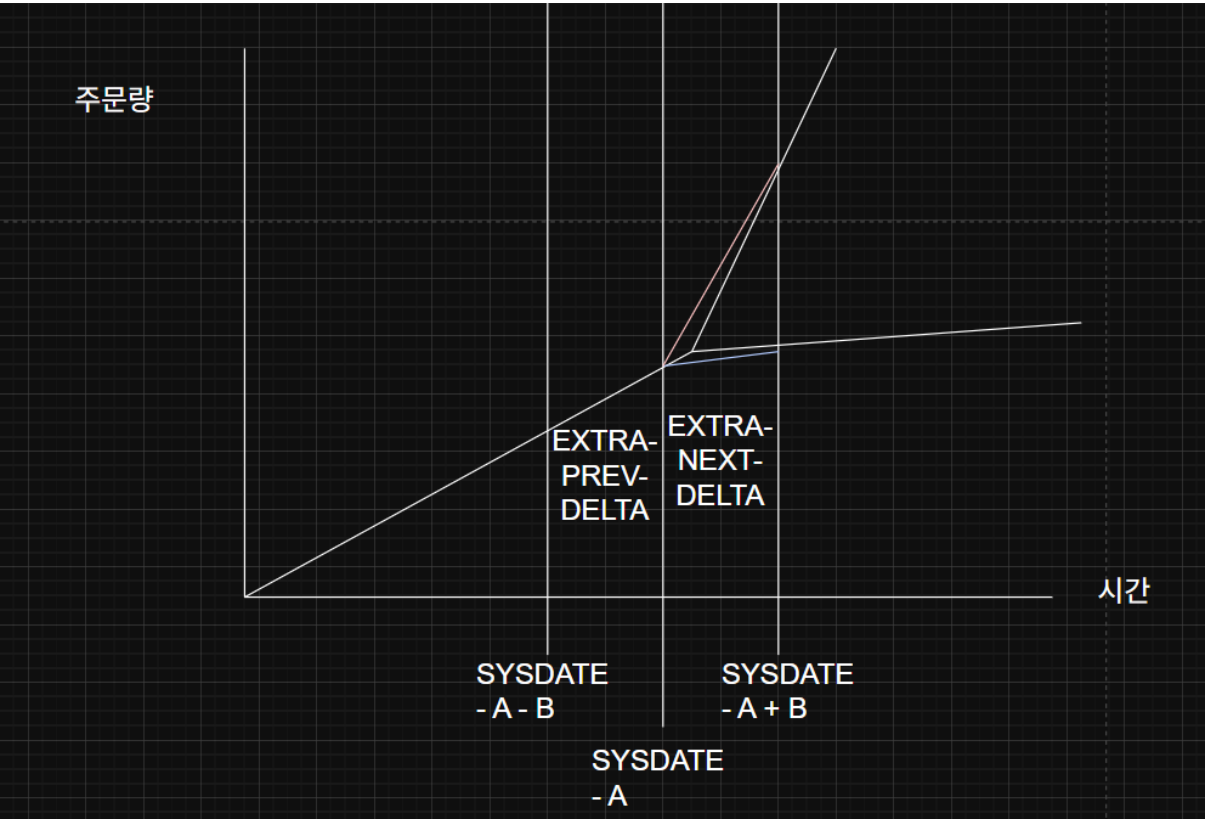
상황 1 : 어떤 상품이 갑자기 홍보 및 마케팅 **SNS**등을 통해 주문량이 급증하는 상황입니다.



상황 2 : 어떤 상품에 대해 갑자기 문제가 생기는 경우입니다. 제품 하자이슈등이 발견되어 주문량이 급감하는 상황입니다.



급증, 급감 상태체크 구현 방법 :



설명 : **SYSDATE - A**를 기준으로 **EXTRA-NEXT-DELTA**와 **EXTRA-PREV-DELTA**를 구한 뒤, 이 차이값을 구해 양수면 급증, 음수면 급감이라 생각하면 될거라 생각이 듭니다. 급등 및 급강을 확인해야 하므로 A와 B의 값은 작게 설정할 필요가 있다 생각이 듭니다. 크게 설정하면 그만큼 유효기간이 늘어나기 때문에 급등 및 급강의 크기가 지속적으로 감소하고 이러한 지속성은 이러한 순간적 상황에는 어울리지 않기 때문입니다.

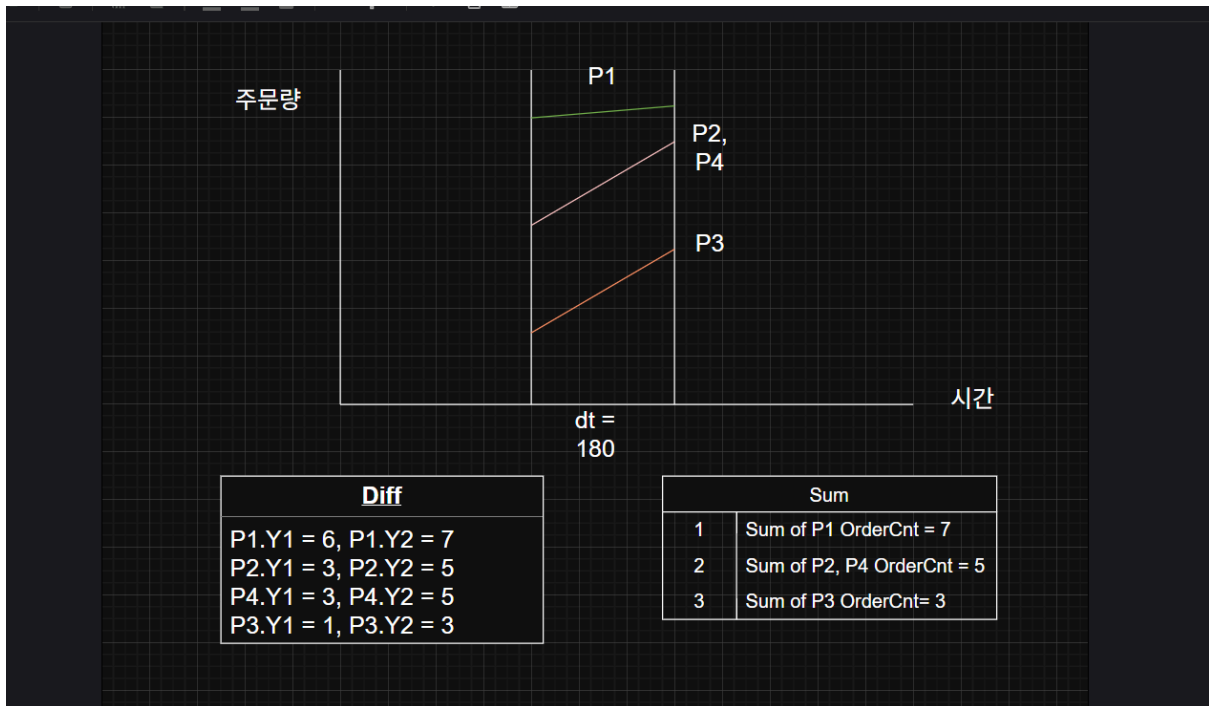
2-c. 데이터 추천조회 코드

설명 : **sql for market.sql** 참고하시면 됩니다. 카테고리 테이블에 있는 기준값들은 서버에서 패러미터로 받아와서 처리하는 걸로 가정하였습니다. 버전**1** 코드는 처음본인이 생각해낸 코드이고 버전**2**코드는 버전**1**코드를 약간 개량하여 코드를 구성해보았습니다. 결과는 버전**2** 코드의 결과입니다.

추가 고려점 : 데이터 조회는 평점순, 판매량순, 최신순도 포함하는게 좋다고 생각합니다. 왜냐하면 소비자는 제품에 대한 여러가지 순서를 볼 필요가 있으며 추천조회가 항상 소비자가 원하는게 아닐 수도 있기 때문입니다.

- **추천조회순위 :**
 - **1순위 :** 델타값보다 큰가
 - **2순위 :** 판매량 순
 - **3순위 :** 델타값 순
 - **4순위 :** 평점 순
 - **5순위 :** 평점 횟수 순
 - **6순위 :** 상품 열람수 순

참고 그래프 :



참고그래프 설명 : 기준 주문량값은 **0.01**로 유효기간은 **180**일로 정해보았습니다. **SUM**표는 각각의 상품(**P1, P2, P3, P4**)의 현재 주문량 값입니다. 이값이 **DIFF**표의 **Y2**값이며 **Y1**값은 **180**일 이전의 주문량 값입니다.

결과 : **P1**은 기준 주문량 충족 안돼서 제외되었습니다. **P3**는 주문량 저조해서 뒤로 밀려나게 되었습니다. **P2, P4** 중 위의 순위 고려하여 표시를 하면 다음과 같습니다.

49

50

51

질의 결과 x

SQL | 인출된 모든 행: 3(0.006초)

	PRODUCT_ID	PRODUCT_NAME	PRODUCT_REG_DATE	PRODUCT_RATING	PRODUCT_RATING_CNT	PRODUCT_VIEW_CNT
1	4	productName4	25/10/16	3	4	4
2	2	productName2	25/10/16	2	2	2
3	3	productName3	25/10/16	3	3	3