

Main Quest 2 추천시스템

평가 항목

추천시스템 1

- 1. 추천시스템의 이름
- 2. 추천시스템의 테마 및 목적
- 3. 구현 로직 (코드, 쿼리 및 구조도)
- 4. 결과
- 5. 참고자료

추천시스템 2

- 1. 추천시스템의 이름
- 2. 추천시스템의 테마 및 목적
- 3. 구현 로직 (코드, 쿼리 및 구조도)
- 4. 결과
- 5. 참고자료

추천시스템 3

- 1. 추천시스템의 이름
- 2. 추천시스템의 테마 및 목적
- 3. 구현 로직 (코드, 쿼리 및 구조도)
- 4. 결과
- 5. 참고자료

추천시스템 4

- 1. 추천시스템의 이름
- 2. 추천시스템의 테마 및 목적
- 3. 구현 로직 (코드, 쿼리 및 구조도)
- 4. 결과
- 5. 참고자료

회고

좋았던 점

아쉬운 점

개선할 점

평가 항목

채점 기준	설명
1. 추천 시스템의 창의성과 복잡성	5가지의 추천 시스템 아이디어가 모두 독창적인가?
2. SQL 구현의 정확성 및 효율성	SQL 쿼리가 데이터의 목적에 맞게 설계되었는가?
3. 보고서 구성 및 설명력	논리적으로 추천 시스템 구현 로직을 설명했는가?

추천시스템 1

1. 추천시스템의 이름

• 이런 상품은 어때요?

2. 추천시스템의 테마 및 목적

• 현재 상품과 완전히 동일한 카테고리 내 인기가 높은 상품을 알려주어 선택지의 폭을 넓히고, 새로운 상품을 탐색하는 과정을 건너뛰어 탐색 중 발생하는 고객 이탈을 줄이기

3. 구현 로직 (코드, 쿼리 및 구조도)

- **인기가 높은 상품**의 기준 설정 → 현재 상품과 동일한 카테고리의 제품을 비교하여 리뷰가 100개 이상이고, 평점 평균이 현재 상품보다 높은 상품 탐 새
- 기준이 될 현재 상품이 필요하므로 임의의 상품을 현재 상품으로 가정하여 비교 후 추출

```
with now_item as(
SELECT *
FROM `projects.amazon`
ORDER BY RAND()
LIMIT 1)
select
```

```
pa.product_name,
  pa.category,
  pa.rating,
  pa.rating_count,
  pa.about_product,
  pa.actual_price,
  pa.discount_percentage,
  pa.discounted_price
from now_item n, `projects.amazon` pa
where
  pa.category = n.category
  and
  pa.rating_count >= 100
 and
  pa.rating > n.rating
order by
  pa.rating desc,
  pa.rating_count desc;
```

4. 결과

• Computers&Accessories|Accessories&Peripherals|HardDiskBags 카테고리 설정 시



5. 참고자료

• <u>rand()</u>

추천시스템 2

1. 추천시스템의 이름

• 보다 저렴한 상품도 있어요

2. 추천시스템의 테마 및 목적

• 동일 제품명을 가진 상품 중 최종 가격이 현재 상품보다 저렴한 상품을 제시, 구매를 유도

3. 구현 로직 (코드, 쿼리 및 구조도)

• 기준이 될 **현재 상품**이 필요하므로 product_name 이 중복된 상품 중 무작위로 하나를 선택, 현재 상품으로 가정 → 보다 저렴한 상품이 있다면 가격이 저렴한 순서대로 정렬하여 출력

```
with duplication as(
 SELECT product_name,
 COUNT(product_name) AS cnt
  FROM `projects.amazon`
  GROUP BY product_name
  HAVING cnt >= 2
 ),
now_item as(
  SELECT a.*
  FROM `projects.amazon` a, duplication d
  where a.product_name = d.product_name
 ORDER BY RAND()
 limit 1
 )
select
    pa.product_id,
    pa.product_name,
    pa.discounted_price,
```

```
pa.actual_price,
  pa.discount_percentage

from
  now_item n, `projects.amazon` pa
where
  pa.product_name = n.product_name
  and
  pa.discounted_price < n.discounted_price
order by
  pa.product_name, pa.discounted_price;</pre>
```

• 위처럼 쿼리를 작성할 경우 now_item 에서 랜덤으로 추출된 상품이 최저가일 수 있으므로, 예시를 위해 최저가가 아니면서 product_name 이 중복된 상 품 중 하나를 선정해 아래처럼 추출

```
select
    a.product_id,
    a.product_name,
    a.category,
    a.discounted_price,
    a.actual_price,
    a.discount_percentage
from
`projects.amazon` a,
(select * from `projects.amazon` where product_id = 'BOBCZCQTJX') as dupe
where
    a.product_name = dupe.product_name
    and
    a.discount_percentage > 0.0
    and
    a.discounted_price < dupe.discounted_price
order by
    a.discounted_price;</pre>
```

4. 결과



5. 참고자료

.

추천시스템 3

1. 추천시스템의 이름

• 이렇게 평가했어요

2. 추천시스템의 테마 및 목적

• 특정 사용자의 리뷰를 편리하게 조회하고 추천하는 상품을 수월하게 살펴볼 수 있게 함

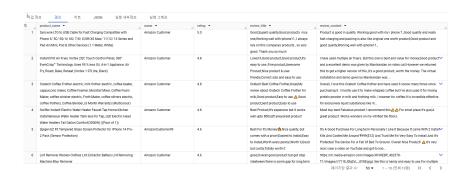
3. 구현 로직 (코드, 쿼리 및 구조도)

- 선택한 사용자가 리뷰를 작성한 상품을 조회하고, 평점이 높은 순으로 정렬 (여기서는 Amazon Customer라는 사용자를 임의 지정)
- rating 이 String 타입이라 숫자가 아닌 값이 들어있는 컬럼도 존재 → safe_cast 함수를 통해 실수만 추출하도록 조건 추가

```
with find_user as(
   SELECT *
   FROM `projects.amazon`,
   unnest (SPLIT(user_name, ',')) AS name
   )
select
   distinct(product_name), name, rating, review_title, review_content
```

```
from
  find_user
where
  safe_cast(rating as float64) is not null
and
  name = 'Amazon Customer'
order by
  safe_cast(rating as float64) desc;
```

4. 결과



5. 참고자료

• safe cast()

추천시스템 4

1. 추천시스템의 이름

• 가장 크게 할인 중이에요!

2. 추천시스템의 테마 및 목적

• 카테고리 별 할인 폭이 큰 상품을 나열하여 사용자 관심 유발

3. 구현 로직 (코드, 쿼리 및 구조도)

• 제시된 카테고리 값을 버티컬 라인을 기준으로 나누고, 이렇게 나눈 카테고리마다 할인 폭이 큰 순서대로 상품을 찾아 최대 3개까지 정렬

```
with category_data as(
  select
   product_id,
   product_name,
   category,
   SPLIT(category, '|') as splited_category,
   discount_percentage
 from `projects.amazon`
),
categories as(
 SELECT
   distinct(product_id),
   product_name,
   category,
   category_value,
   discount_percentage
    category_data,
    UNNEST(splited_category) AS category_value
ranked as (
  select
    product_name,
    category_value,
    discount_percentage,
```

```
row_number() over (
     partition by category_value
     order by discount_percentage desc
 ) as value_rank
 from categories
)
select
 category_value,
 product name,
 concat(round(discount_percentage*100,0),'%') as discount_per
 ranked
where
 value_rank <= 3</pre>
 discount_percentage > 0
 category_value,
 discount_per desc;
```

4. 결과



5. 참고자료

• row_number()

회고

그간 학습 내용을 토대로 다양한 인사이트를 도출하는 것이 목표인 두 번째 메인 퀘스트.

내가 짠 쿼리는 너무 조잡하거나 단순한 결과만 내는 게 아닐까 싶었지만, 결과물을 제출해야 한다는 생각이 컸다.🤗

좋았던 점

- → 강의에서 배우지 않았으나 목적에 맞는 함수를 찾아 적용하고, 이를 통해 어떤 가치 있는 값을 추출할 수 있을지 이리저리 생각하는 데에서 오는 즐거움 이 있었다.
- → 적당한 긴장감을 통해 집중한 채 하루를 보낼 수 있었고, 타인과 비교할 도리가 없어 주눅 들거나 좌절할 일이 없었다.

아쉬운 점

추천 시스템

- 이 시간대 많이 구매한 상품
- 나와 같은 상품을 구매한 고객들이 선택한 상품
- 아직 리뷰가 하나도 작성되지 않은 상품 알림
- 재구매 상품의 구매 횟수 알림 등등 여러 생각을 해봤지만...
- → 어떤 데이터가 있는지 파악하지 않은 채 무턱대고 궁리하다가 아이디어들을 폐기해야 했다.
- → Looker Studio까지는 손대지 못했으나 툴에 대한 이해도 상승이 목적인 만큼, 과제와는 별개로 만져보며 익숙해지도록 하자🦾
- → 쿼리 짜는 방법도, 사용하는 함수도 제각각이라 다른 분들은 무슨 아이디어를 내셨을지 구경하고 싶지만 따로 확인할 방법이 없어 아쉽다.

개선할 점

→ 이번에는 18시 전에 제출할 수 있을 줄 알았는데...! 사용할 수 있는 자원부터 확인해야 하는 걸 일의 순서가 뒤죽박죽이라 또다시 시간 낭비. 인간의 욕심은 끝이 없고 같은 실수를 반복한다☺️