



고객을 세그먼테이션하자 [프로젝트] - 김수경

11-2. 데이터 불러오기

데이터 살펴보기

- 테이블에 있는 10개의 행만 출력하기

```
select *  
from `project-dac3b643-036f-483e-a72.modulabs_project.data`  
limit 10;
```

project_mainquest_03

실행

쿼리 저장

다운로드

공유

일일

다음에서 열기

더보기

```
1 select
2 from 'project-dac386d3-891f-482e-a72_modules-project_data'
3 limit 10;
```

실행 시 이 쿼리가 46.02MB를 처리합니다.

쿼리 결과

결과 저장

다음에서 열기

작업 정보

결과

시각화

JSON

실행 세부정보

실행 그래프

행	Inventory	StockCode	Description	Quantity	InvoicedDate	UnitPrice	CustomerID	Country
1	536365	85123A	WHITE HANGING HEART T.L.G.	4	2010-12-01 08:26:00 UTC	2.55	17850	United Kingdom
2	536365	71053	WHITE METAL LANTERN	6	2010-12-01 08:26:00 UTC	3.39	17850	United Kingdom
3	536365	844068	CREAM CUPID HEARTS COAT H...	8	2010-12-01 08:26:00 UTC	2.75	17850	United Kingdom
4	536365	840290	KNITTED UNION FLAG HOT WA...	6	2010-12-01 08:26:00 UTC	3.39	17850	United Kingdom
5	536365	840296	RED WOOLLY HOTTIE WHITE H...	6	2010-12-01 08:26:00 UTC	3.39	17850	United Kingdom
6	536365	22752	SET 7 BABUSHKA NESTING BO...	2	2010-12-01 08:26:00 UTC	7.65	17850	United Kingdom
7	536365	21730	GLASS STAR FROSTED T-LIGHT...	6	2010-12-01 08:26:00 UTC	4.25	17850	United Kingdom
8	536366	22633	HAND WARMER UNION JACK	6	2010-12-01 08:28:00 UTC	1.85	17850	United Kingdom
9	536366	22632	HAND WARMER RED POLKA DOT	6	2010-12-01 08:28:00 UTC	1.85	17850	United Kingdom
10	536367	84879	ASSORTED COLOUR BIRD ORN...	32	2010-12-01 08:34:00 UTC	1.69	13047	United Kingdom

페이지당 결과 수: 50

1 - 10 (전체 10행)

<

>

- 전체 데이터는 몇 행으로 구성되어 있는지 확인하기

```
select count(*)  
from `project-dac3b643-036f-483e-a72.modulabs_project.data`
```

project_mainquest_03 실행 쿼리 저장 다운로드

```
1 select count(*)  
2 from `project-dac3b643-036f-483e-a72.modulabs_project.data`
```

쿼리 결과

작업 정보	결과	시각화	JSON	실행 세부정보	실행 그래프
행	f0_				
1	541909				

- 541,909행

데이터 수 세기

- COUNT 함수를 사용해서, 각 컬럼별 데이터 포인트의 수를 세어 보기

```
select count(invoiceno) as Invoicenos,
       count(stockcode) as Stockcodes,
       count(description) as Descriptions,
       count(quantity) as Quantities,
       count(invoicedate) as Invoicedates,
       count(unitprice) as Unitprices,
       count(customerid) as Customerids,
       count(country) as Countries
from `project-dac3b643-036f-483e-a72.modulabs_project.data`
```

project_mainquest_03 실행 쿼리 저장 다운로드 공유 일정 다음에서 열기 더보기

```
1 select count(invoiceno) as Invoicenos,
2       count(stockcode) as Stockcodes,
3       count(description) as Descriptions,
4       count(quantity) as Quantities,
5       count(invoicedate) as Invoicedates,
6       count(unitprice) as Unitprices,
7       count(customerid) as Customerids,
8       count(country) as Countries
9 from `project-dac3b643-036f-483e-a72.modulabs_project.data`
```

쿼리 완료됨
주문형 처리 할당량 사용 중

쿼리 결과

작업 정보 결과 시각화 JSON 실행 세부정보 실행 그래프

행	Invoicenos	Stockcodes	Descriptions	Quantities	Invoicedates	Unitprices	Customerids	Countries
1	541909	541909	540455	541909	541909	541909	406829	541909

11-4. 데이터 전처리 방법(1): 결측치 제거

컬럼 별 누락된 값의 비율 계산

- 각 컬럼 별 누락된 값의 비율을 계산
 - 각 컬럼에 대해서 누락 값을 계산한 후, 계산된 누락 값을 UNION ALL을 통해 합치기

```
SELECT 'InvoiceNo' AS column_name,
       ROUND(COUNTIF(InvoiceNo IS NULL) / COUNT(*) * 100, 2) AS missing_percentage
FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`
UNION ALL
SELECT 'StockCode',
       ROUND(COUNTIF(StockCode IS NULL) / COUNT(*) * 100, 2)
FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`
UNION ALL
SELECT 'Description',
       ROUND(COUNTIF(Description IS NULL) / COUNT(*) * 100, 2)
FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`
UNION ALL
SELECT 'Quantity',
       ROUND(COUNTIF(Quantity IS NULL) / COUNT(*) * 100, 2)
FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`
UNION ALL
SELECT 'InvoiceDate',
       ROUND(COUNTIF(InvoiceDate IS NULL) / COUNT(*) * 100, 2)
FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`
UNION ALL
SELECT 'UnitPrice',
       ROUND(COUNTIF(UnitPrice IS NULL) / COUNT(*) * 100, 2)
FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`
UNION ALL
SELECT 'CustomerID',
       ROUND(COUNTIF(CustomerID IS NULL) / COUNT(*) * 100, 2)
FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`
UNION ALL
SELECT 'Country',
```

```
ROUND(COUNTIF(Country IS NULL) / COUNT(*) * 100, 2)
FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`;
```

project_mainquest_03 실행 쿼리 저장 다운로드 공유 일정

```

1 SELECT 'InvoiceNo' AS column_name,
2     ROUND(COUNTIF(InvoiceNo IS NULL) / COUNT(*) * 100, 2) AS missing_percentage
3 FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`
4 UNION ALL
5 SELECT 'StockCode',
6     ROUND(COUNTIF(StockCode IS NULL) / COUNT(*) * 100, 2)
7 FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`
8 UNION ALL
9 SELECT 'Description',
10     ROUND(COUNTIF(Description IS NULL) / COUNT(*) * 100, 2)
11 FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`
12 UNION ALL
13 SELECT 'Quantity',

```

실행 시 이 쿼리가 46.02MB를 처리합니다.

쿼리 결과

작업 정보	결과	시각화	JSON	실행 세부정보	실행 그래프
행	column_name		missing_percenta...		
1	InvoiceNo		0.0		
2	StockCode		0.0		
3	Description		0.27		
4	Quantity		0.0		
5	InvoiceDate		0.0		
6	UnitPrice		0.0		
7	CustomerID		24.93		
8	Country		0.0		

결측치 처리 전략

- StockCode = '85123A' 의 Description 을 추출하는 쿼리문을 작성하기

```

SELECT Description
FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`
WHERE StockCode = '85123A'
GROUP BY Description

```

project_mainquest_03 실행 쿼리 저장 다운로드 공유 일정

```

1 SELECT Description
2 FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`
3 WHERE StockCode = '85123A'
4 GROUP BY Description

```

주문형 처리 할당량 사용 중

쿼리 결과

작업 정보	결과	시각화	JSON	실행 세부정보	실행 그래프
행	Description				
1	WHITE HANGING HEART T-LIG...				
2	?				
3	wrongly marked carton 22804				
4	CREAM HANGING HEART T-LIG...				

결측치 처리 (★ delete 후 이전 원본 데이터를 재활용하기)

- DELETE 구문을 사용하며, WHERE 절을 통해 데이터를 제거할 조건을 제시

```
DELETE FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`  
WHERE Customerid is null  
or Description is null
```

project_mainquest_03 실행 쿼리 저장 다운로드 공유

```
1 DELETE FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`  
2 WHERE Customerid is null  
3 or Description is null
```

주문형 처리 할당량 사용 중

쿼리 결과

작업 정보 결과 실행 세부정보 실행 그래프

이 문으로 data의 행 135,080개가 삭제되었습니다.

11-5. 데이터 전처리(2): 중복값 처리

중복값 확인

- 중복된 행의 수를 세어보기
 - 8개의 컬럼에 그룹 함수를 적용한 후, COUNT가 1보다 큰 데이터를 세어보기

```
SELECT *, count(*) as cnt  
FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`  
group by InvoiceNo, StockCode, Description, Quantity, InvoiceDate, UnitPrice, CustomerID, Country  
having count(*) >= 2
```

project_mainquest_03 실행 쿼리 저장 다운로드 공유 일장 다음에서 열기 더보기

```
1 SELECT *, count(*) as cnt  
2 FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`  
3 group by InvoiceNo, StockCode, Description, Quantity, InvoiceDate, UnitPrice, CustomerID, Country  
4 having count(*) >= 2
```

실행 시 이 쿼리가 35.29MB를 처리합니다.
주문형 처리 할당량 사용 중

쿼리 결과 결과 시각화 JSON 실행 세부정보 실행 그래프

행	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country	cnt
1	571034	23299	SET OF 4 KNOCK KNAACK TINS P...	6	2011-10-13 12:47:00 UTC	4.15	12359	Cyprus	2
2	571034	23494	VINTAGE DOOLY DELINE SEVEN...	3	2011-10-13 12:47:00 UTC	5.95	12359	Cyprus	2
3	571034	22145	SET OF 8 REGENCY CASE TINS	4	2011-10-13 12:47:00 UTC	4.95	12359	Cyprus	2
4	538026	22149	FELTCRAFT PRINCESS CHARLO...	1	2010-12-14 12:56:00 UTC	3.75	12370	Cyprus	2
5	577228	23156	SET OF 5 MINI GROCERY MAG...	1	2011-11-18 12:07:00 UTC	2.08	12391	Cyprus	2
6	577228	84580	MOUSE TOY WITH PINK T-SHIRT	1	2011-11-18 12:07:00 UTC	3.75	12391	Cyprus	2
7	577228	22435	SET OF 9 HEART SHAPED BALL...	1	2011-11-18 12:07:00 UTC	1.25	12391	Cyprus	2
8	577228	22144	CHRISTMAS CRAFT LITTLE FRIL...	1	2011-11-18 12:07:00 UTC	2.1	12391	Cyprus	2
9	577228	23048	SET OF 10 LANTERNS FAIRY LL...	1	2011-11-18 12:07:00 UTC	4.15	12391	Cyprus	3
10	577228	22270	HAPPY EASTER HANGING DEC...	1	2011-11-18 12:07:00 UTC	3.75	12391	Cyprus	2
11	539419	48138	DOORMAT UNION FLAG	10	2010-12-17 14:10:00 UTC	6.75	12431	Australia	2
12	538174	22326	ROUND SNACK BOXES SET OF...	12	2010-12-10 09:35:00 UTC	2.95	12471	Germany	2

페이지당 결과 수: 50 1 - 50 (전체 4837행) < > >

중복값 처리

- 중복값을 제거하는 쿼리문 작성하기
 - CREATE OR REPLACE TABLE 구문을 활용하여 모든 컬럼(*)을 DISTINCT 한 데이터로 업데이트

```
CREATE OR REPLACE TABLE `project-dac3b643-036f-483e-a72.modulabs_project.data` as
select distinct *
from `project-dac3b643-036f-483e-a72.modulabs_project.data`
```

project_mainquest_03 실행 쿼리 저장 다운로드 공유

```
1 CREATE OR REPLACE TABLE `project-dac3b643-036f-483e-a72.modulabs_project.data` as
2 select distinct *
3 from `project-dac3b643-036f-483e-a72.modulabs_project.data`
```

쿼리 완료됨

주문형 처리 할당량 사용 중

쿼리 결과

작업 정보 결과 실행 세부정보 실행 그래프

이 문으로 이름이 data인 테이블이 교체되었습니다.

11-6. 데이터 전처리(3): 오류값 처리

InvoiceNo 살펴보기

- 고유(unique)한 InvoiceNo 의 개수를 출력하기

```
select count(invoiceno)
from `project-dac3b643-036f-483e-a72.modulabs_project.data`
group by invoiceno
```

project_mainquest_03 실행 쿼리 저장 다운로드 공유

```
1 select count(invoiceno)
2 from `project-dac3b643-036f-483e-a72.modulabs_project.data`
3 group by invoiceno
```

쿼리 완료됨

쿼리 결과

작업 정보	결과	시각화	JSON	실행 세부정보	실행 그래프
1	541431				
2	541431				
3	541431				
4	541431				
5	541431				
6	541431				
7	541431				
8	541431				
9	541431				
10	541431				
11	541431				
12	541431				
13	541431				
14	541431				
15	541431				

페이지당 결과 수: 50 1 - 50 (전체 22190명) < >

- 고유한 InvoiceNo 를 앞에서부터 100개를 출력하기

```
select invoiceno
from `project-dac3b643-036f-483e-a72.modulabs_project.data`
```

```
group by invoiceno
order by invoiceno
limit 100
```

project_mainquest_03 실행 쿼리 저장 다운로드 공유 설정 다음에서 열기 더보기

```
1 select invoiceno
2 from `project-dac3b643-036f-483e-a72.modulabs_project.data`
3 group by invoiceno
4 order by invoiceno
5 limit 100
```

실행 시 이 쿼리가 3.07MB를 처리합니다.
주문형 처리 할당량 사용 중

쿼리 결과

작업 정보 결과 시각화 JSON 실행 세부정보 실행 그래프

행	invoiceno
1	S36365
2	S36366
3	S36367
4	S36368
5	S36369
6	S36370
7	S36371
8	S36372
9	S36373
10	S36374
11	S36375
12	S36376
13	S36377
14	S36378
15	S36380

페이지당 결과 수: 50 1 ~ 50 (전체 100행) |< > >|

- InvoiceNo 가 'C'로 시작하는 행을 필터링 할 수 있는 쿼리문을 작성하기 (100행까지만 출력)

```
select invoiceno
from `project-dac3b643-036f-483e-a72.modulabs_project.data`
where invoiceno like 'C%'
group by invoiceno
order by invoiceno
limit 100
```

project_mainquest_03 실행 쿼리 저장 다운로드 공유 설정 다음에서 열기 더보기

```
1 select invoiceno
2 from `project-dac3b643-036f-483e-a72.modulabs_project.data`
3 where invoiceno like 'C%'
4 group by invoiceno
5 order by invoiceno
6 limit 100
```

실행 시 이 쿼리가 3.07MB를 처리합니다.
주문형 처리 할당량 사용 중

쿼리 결과

작업 정보 결과 시각화 JSON 실행 세부정보 실행 그래프

행	invoiceno
1	C336379
2	C336383
3	C336391
4	C336506
5	C336543
6	C336548
7	C336606
8	C336622
9	C336625
10	C336642
11	C336734
12	C336727
13	C336757
14	C336758
15	C336760

페이지당 결과 수: 50 1 ~ 50 (전체 100행) |< > >|

- 구매 건 상태가 Canceled 인 데이터의 비율(%) - 소수점 첫번째 자리까지

```
SELECT ROUND(SUM(CASE WHEN InvoiceNo like 'C%' THEN 1 ELSE 0 END)/count(*) * 100, 1) as InvoiceNo
FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`
```

project_mainquest_03 실행 쿼리 저장 다운로드 공유 일정 다음에서 열기

```

1 SELECT ROUND(SUM(CASE WHEN InvoiceNo like 'C%' THEN 1 ELSE 0 END)/count(*) * 100, 1) as InvoiceNo
2 FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`

```

주문형 처리 할당량 사용 중

쿼리 결과

작업 정보 결과 시각화 JSON 실행 세부정보 실행 그래프

행	InvoiceNo
1	2.2

StockCode 살펴보기

- 고유한 StockCode 의 개수를 출력하기

```

SELECT count(stockcode)
FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`
GROUP BY stockcode

```

project_mainquest_03 실행 쿼리 저장 다운로드 공유 일정 다음에서 열기 더보기

```

1 SELECT count(stockcode)
2 FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`
3 GROUP BY stockcode

```

실행 시 이 쿼리가 2.71MB를 처리합니다.

쿼리 결과 쿼리 저장 다음에서 열기

작업 정보 결과 시각화 JSON 실행 세부정보 실행 그래프

행	count
1	208
2	339
3	18
4	226
5	539
6	149
7	17
8	61
9	60
10	124
11	96
12	75
13	310
...	...

페이지당 결과 수: 50 1 ~ 50 (전체 3664명) |< > |

- 어떤 제품이 가장 많이 판매되었는지 보기 위하여 StockCode 별 등장 빈도를 출력하기
 - 상위 10개의 제품들을 출력하기

```

SELECT StockCode, COUNT(*) AS sell_cnt
FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`
GROUP BY StockCode
ORDER BY sell_cnt DESC
limit 10;

```

project_mainquest_03 실행 쿼리 저장 다운로드 공유

```

1 SELECT StockCode, COUNT(*) AS sell_cnt
2 FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`
3 GROUP BY StockCode
4 ORDER BY sell_cnt DESC
5 limit 10;

```

주문형 처리 할당량 사용 중

쿼리 결과

작업 정보	결과	시각화	JSON	실행 세부정보	실행 그래프
행	StockCode	sell_cnt			
1	85123A	2065			
2	22423	1894			
3	85099B	1659			
4	47566	1409			
5	84879	1405			
6	20725	1346			
7	22720	1224			
8	POST	1196			
9	22197	1110			
10	23203	1108			

- **StockCode**의 컬럼에 있던 값 중에서 숫자를 제외한 문자만 남기고 문자가 몇 자리 수 인지 세고
 ◦ 숫자가 0~1개인 값들에는 어떤 코드들이 들어가 있는지 출력하기

```

SELECT DISTINCT StockCode, number_count
FROM (SELECT StockCode,
      LENGTH(StockCode) - LENGTH(REGEXP_REPLACE(StockCode, r'[0-9]', '')) AS number_count
      FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`)
WHERE number_count <= 1;

```


project_mainquest_03 실행 쿼리 저장 다운로드 공유

```

1 SELECT DISTINCT StockCode, number_count
2 FROM (SELECT StockCode,
3             LENGTH(StockCode) - LENGTH(REGEXP_REPLACE(StockCode, r'[0-9]', '')) AS number_count
4        FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`)
5 WHERE number_count <= 1;

```

주문형 처리 할당량 사용 중

쿼리 결과

작업 정보	결과	시각화	JSON	실행 세부정보	실행 그래프
행	StockCode	number_count			
1	POST	0			
2	M	0			
3	C2	1			
4	D	0			
5	BANK CHARGES	0			
6	PADS	0			
7	DOT	0			
8	CRUK	0			

- **StockCode**의 컬럼에 있던 값 중에서 숫자를 제외한 문자만 남기고 문자가 몇 자리 수 인지 세고
 - 숫자가 0~1개인 값들을 가지고 있는 데이터 수는 전체 데이터 수 대비 몇 퍼센트인지 구하기 (소수점 두 번째 자리까지)

```

WITH stockcode_digits as (SELECT StockCode,
                                LENGTH(StockCode) - LENGTH(REGEXP_REPLACE(StockCode, r'[0-9]', '')) AS number_count
                           FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`)
SELECT ROUND(SUM(CASE WHEN number_count <= 1 THEN 1 ELSE 0 END)/count(*) * 100, 2) as StockCode_per
FROM stockcode_digits

```

[결과 이미지를 넣어주세요]

- (이미 DELETE 함수를 사용해 이미지 첨부 불가..)
 - 답 : 0.48%

- 제품과 관련되지 않은 거래 기록을 제거하기

```

DELETE FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`
WHERE StockCode IN (
  SELECT DISTINCT StockCode
  FROM (SELECT StockCode,
              LENGTH(StockCode) - LENGTH(REGEXP_REPLACE(StockCode, r'[0-9]', '')) AS number_count
        FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`) AS A
  WHERE A.number_count <= 1);

```

project_mainquest_03 실행 쿼리 저장 다운로드 공유 일정

```

1 DELETE FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`
2 WHERE StockCode IN (
3   SELECT DISTINCT StockCode
4   FROM (SELECT StockCode,
5         LENGTH(StockCode) - LENGTH(REGEXP_REPLACE(StockCode, r'[0-9]', '')) AS number_count
6        FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`) AS A
7  WHERE A.number_count <= 1);

```

쿼리 완료됨

주문형 처리 할당량 사용 중

쿼리 결과

작업 정보 결과 실행 세부정보 실행 그래프

이 문으로 data의 행 1,915개가 삭제되었습니다.

Description 살펴보기

- 고유한 Description 별 출현 빈도를 계산하고 상위 30개를 출력하기

```

SELECT Description, COUNT(*) AS description_cnt
FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`
GROUP BY Description
ORDER BY description_cnt desc
limit 30

```

project_mainquest_03 실행 쿼리 저장 다운로드 공유 일정 다음에서 열기 더보기

```

1 SELECT Description, COUNT(*) AS description_cnt
2 FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`
3 GROUP BY Description
4 ORDER BY description_cnt desc
5 limit 30

```

쿼리 완료됨

쿼리 결과 쿼리 저장 다음에서 열기

작업 정보	결과	시각화	JSON	실행 세부정보	실행 그래프
1	WHITE HANDING HEART TIG...		description_cnt	2058	
2	REGENCY CAKESTAND 3 TIER			1894	
3	JUMBO BAG RED RETROSPOT			1699	
4	PARTY BUNTING			1409	
5	ASSORTED COLOUR BIRD ORN...			1405	
6	LUNCH BAG RED RETROSPOT			1345	
7	SET OF 3 CAKE TINS PANTRY D...			1224	
8	LUNCH BAG BLACK SKULL			1099	
9	PACK OF 72 RETROSPOT CAKE...			1062	
10	SPOTTY BUNTING			1026	
11	PAPER CHAIN KIT 50'S CHRIST...			1013	
12	LUNCH BAG SPACEBOY DESIGN			1006	
13	LUNCH BAG CARD BLUE			1000	
14	LAC KITE DECORATION (BLACK)			1000	

페이지당 결과 수: 50 1 - 30 (전체 30행) < >

- 서비스 관련 정보를 포함하는 행들을 제거하기

```

DELETE FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`
WHERE REGEXP_CONTAINS(UPPER(Description),
r'(POSTAGE|CARRIAGE|AMAZON|BANK CHARGES|SAMPLES|TEST|MISSING|DAMAGED|ADJUST|DOTCOMPOSTAGE)');

```



- 대소문자를 혼합하고 있는 데이터를 대문자로 표준화 하기

```
CREATE OR REPLACE TABLE `project-dac3b643-036f-483e-a72.modulabs_project.data` AS
SELECT
  * EXCEPT (Description),
  CASE
    WHEN REGEXP_CONTAINS(Description, r'([A-Z].*[a-z])|([a-z].*[A-Z])')
    THEN UPPER(Description)
    ELSE Description
  END AS Description
FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`;
```



UnitPrice 살펴보기

- UnitPrice 의 최솟값, 최댓값, 평균을 구하기

```
SELECT min(unitprice) AS min_price, max(unitprice) AS max_price, avg(unitprice) AS avg_price
FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`;
```

project_mainquest_03 실행 쿼리 저장 다운로드 공유 일정 다음에

```

1 SELECT min(unitprice) AS min_price, max(unitprice) AS max_price, avg(unitprice) AS avg_price
2 FROM `project-dac3b643-836f-483e-a72.modulabs_project.data`;

```

쿼리 완료됨

쿼리 결과

작업 정보 결과 시각화 JSON 실행 세부정보 실행 그래프

행	min_price	max_price	avg_price
1	0.0	649.5	2.910256885340...

- 단가가 0원인 거래의 개수, 구매 수량(Quantity)의 최솟값, 최댓값, 평균 구하기

```

SELECT count(quantity) AS cnt_quantity, min(quantity) AS min_quantity, max(quantity) AS max_quantity, avg(quantity) AS avg_quantity
FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`
WHERE unitprice = 0

```

project_mainquest_03 실행 쿼리 저장 다운로드 공유 일정 다음에서 열기 더보기

```

1 SELECT count(quantity) AS cnt_quantity, min(quantity) AS min_quantity, max(quantity) AS max_quantity, avg(quantity) AS avg_quantity
2 FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`
3 WHERE unitprice = 0;

```

주문형 처리 할당량 사용 중

쿼리 결과

작업 정보 결과 시각화 JSON 실행 세부정보 실행 그래프

행	cnt_quantity	min_quantity	max_quantity	avg_quantity
1	33	1	12540	420.5151515151...

- UnitPrice = 0 를 제거하고 일관된 데이터셋을 유지하기

```

CREATE OR REPLACE TABLE `project-dac3b643-036f-483e-a72.modulabs_project.data` AS
SELECT *
FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`
WHERE unitprice != 0

```


▶ 실행

📄 쿼리 저장

⬇ 다운로드

+ 공유

```

1 WITH invo as (select *, cast(invoicedate as date) as invoiceday
2                from `project-dac3b643-036f-483e-a72.modulabs_project.data`)
3 select max(invoiceday) as most_recent_date
4 from invo
  
```

✔ 쿼리 완료됨

쿼리 결과

작업 정보

결과

시각화

JSON

실행 세부정보

실행 그래프

행	most_recent_date
1	2011-12-09

- **유저 별로 가장 큰 InvoiceDay를 찾아서 가장 최근 구매일로 저장하기**

```
select CustomerID,
       cast(invoicedate as date) as invoiceday,
       LAST_VALUE(invoicedate) over (partition by CustomerID order by invoicedate rows between current row and unbounded fol
lowing) as last_invoiceday
from 'project-dac3b643-036f-483e-a72.modulabs_project.data'
```

project_mainquest_03

[🔍 검색](#)
[📄 자료문서](#)
[👤 공유](#)
[🕒 일정](#)
[💬 다음에서 열기](#)
[⚙️ 도구박스 열기](#)

```

1 select CustomerID,
2       cast(Invoicedate as date) as Invoiceday,
3       last_Value(Invoicedate over (partition by CustomerID order by Invoicedate rows between current row and unbounded following)) as last_Invoiceday
4 from   _project-dwh-had-wid-f48e-972-modbus_project_data
    
```

✔️ 확인 완료됨
주분할자의 할당할 사용 중

쿼리 결과

[📄 쿼리 저장](#)
[📌 다음에서 열기](#)

작업 정보	결과	시각화	JSON	실행 세부정보	실행 그래프
명	CustomerID ▶ Invoiceday ▶ last_Invoiceday ▶				
1	12346 2010-01-18 2011-01-18 10:17:00 UTC				
2	12346 2010-01-18 2011-01-18 10:17:00 UTC				
3	12347 2010-12-07 2011-12-07 15:52:00 UTC				
4	12347 2010-12-07 2011-12-07 15:52:00 UTC				
5	12347 2010-12-07 2011-12-07 15:52:00 UTC				
6	12347 2010-12-07 2011-12-07 15:52:00 UTC				
7	12347 2010-12-07 2011-12-07 15:52:00 UTC				
8	12347 2010-12-07 2011-12-07 15:52:00 UTC				
9	12347 2010-12-07 2011-12-07 15:52:00 UTC				
10	12347 2010-12-07 2011-12-07 15:52:00 UTC				
11	12347 2010-12-07 2011-12-07 15:52:00 UTC				
12	12347 2010-12-07 2011-12-07 15:52:00 UTC				
13	12347 2010-12-07 2011-12-07 15:52:00 UTC				
xx	12347 2010-12-07 2011-12-07 15:52:00 UTC				

페이지당 결과 수: 50 | 1 ~ 50 (전체 98227행) | < >

- 가장 최근 일자(`most_recent_date`)와 유저별 마지막 구매일(`invoiceDay`)간의 차이를 계산하기

```
SELECT
  CustomerID,
  EXTRACT(DAY FROM MAX(InvoiceDay) OVER () - InvoiceDay) AS recency
FROM (
  SELECT
    CustomerID,
    MAX(DATE(InvoiceDate)) AS InvoiceDay
  FROM project_name.modulabs_project.data
  GROUP BY CustomerID
);
```

제목 없는 쿼리 실행 저장 다운로드 공유

```

1 SELECT
2   CustomerID,
3   EXTRACT(DAY FROM MAX(InvoiceDay) OVER () - InvoiceDay) AS recency
4 FROM (
5   SELECT
6     CustomerID,
7     MAX(InvoiceDate) AS InvoiceDay
8   FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`
9   GROUP BY CustomerID
10 );

```

실행 시 이 쿼리가 6.08MB를 처리합니다.

주문형 처리 할당량 사용 중

쿼리 결과

작업 정보	결과	시각화	JSON	실행 세부정보	실행 그래프
행	CustomerID	recency			
1	12354	232			
2	12388	15			
3	12452	16			
4	12633	58			
5	12688	113			
6	12830	37			
7	12849	31			
8	13426	0			
9	13492	136			
10	13507	3			
11	13727	28			
12	13953	7			
13	14218	42			

- 최종 데이터 셋에 필요한 데이터들을 각각 정제해서 이어붙이고 지금까지의 결과를 `user_r` 이라는 이름의 테이블로 저장하기

```

CREATE OR REPLACE TABLE `project-dac3b643-036f-483e-a72.modulabs_project.user_r` AS
WITH user_r as (SELECT CustomerID,
  EXTRACT(DAY FROM MAX(InvoiceDay) OVER () - InvoiceDay) AS recency
FROM (SELECT CustomerID,
  MAX(InvoiceDate) AS InvoiceDay
FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`
GROUP BY CustomerID))
select *
from user_r;

```

제목 없는 쿼리 실행 저장 다운로드 공유

```

1 CREATE OR REPLACE TABLE `project-dac3b643-036f-483e-a72.modulabs_project.user_r` AS
2 WITH user_r as (SELECT CustomerID,
3   EXTRACT(DAY FROM MAX(InvoiceDay) OVER () - InvoiceDay) AS recency
4 FROM (SELECT CustomerID,
5   MAX(InvoiceDate) AS InvoiceDay
6 FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`
7 GROUP BY CustomerID))
8 select *
9 from user_r;

```

주문형 처리 할당량 사용 중

쿼리 결과

작업 정보	결과	시각화	JSON	실행 세부정보	실행 그래프
행	CustomerID	recency			
1	12609	78			
2	12716	3			
3	12744	56			
4	12906	11			
5	12956	306			
6	13130	94			
7	13261	268			
8	13270	366			
9	13999	1			

페이지당 결과 수: 50 1 ~ 50 (전체 4361 행) < >

제목 없는 쿼리

실행 저장 다운로드 공유 일정 다음에서 열기 더보기

```

1 CREATE OR REPLACE TABLE `project-dac3b643-036f-483e-a72.modulabs_project.user_r` AS
2 WITH user_r AS (SELECT CustomerID,
3                   EXTRACT(DAY FROM MAX(InvoiceDay) OVER () - InvoiceDay) AS recency
4                   FROM (SELECT CustomerID,
5                               MAX(InvoiceDay) AS InvoiceDay
6                               FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`
7                               GROUP BY CustomerID))
8 select *
9 from user_r;

```

실행 시 이 쿼리가 6.08MB를 처리합니다.

주문형 처리 할당량 사용 중

쿼리 결과

작업 정보 결과 실행 세부정보 실행 그래프

이 문으로 이름이 user_r인 테이블이 교체되었습니다.

Frequency

- 고객마다 고유한 InvoiceNo의 수를 세어보기

```

select customerID, count(distinct invoiceno) as purchase_cnt
from `project-dac3b643-036f-483e-a72.modulabs_project.data`
group by customerID

```

제목 없는 쿼리

실행 저장 다운로드 공유 일정 다음에서 열기 더보기

```

1 select customerID, count(distinct invoiceno) as purchase_cnt
2 from `project-dac3b643-036f-483e-a72.modulabs_project.data`
3 group by customerID

```

쿼리 완료됨

쿼리 결과

작업 정보 결과 시각화 JSON 실행 세부정보 실행 그래프

row	customerID	purchase_cnt
1	12346	2
2	12347	7
3	12348	4
4	12349	1
5	12350	1
6	12352	8
7	12353	1
8	12354	1
9	12355	1
10	12356	3
11	12357	1
12	12358	2
13	12359	6
14	12360	3
15	12361	1
16	12362	13
17	12363	2

페이지당 결과 수: 50 1 - 50 (전체 4361행) < >

- 각 고객 별로 구매한 아이템의 총 수량 더하기

```

select customerID, sum(Quantity) as item_cnt
from `project-dac3b643-036f-483e-a72.modulabs_project.data`
group by customerID

```


제목 없는 쿼리

실행

저장

다운로드

공유

```
1 select customerID, sum(Quantity) as item_cnt
2 from `project-dac3b643-036f-483e-a72.modulabs_project.data`
3 group by customerID
```

쿼리 완료됨

쿼리 결과

작업 정보

결과

시각화

JSON

실행 세부정보

실행 그래프

행	customerID	item_cnt
1	12346	0
2	12347	2458
3	12348	2332
4	12349	630
5	12350	196
6	12352	463
7	12353	20
8	12354	530
9	12355	240
10	12356	1573
11	12357	2708
12	12358	242
13	12359	1599
14	12360	1156
15	12361	90
16	12362	2168
17	12363	408

- 전체 거래 건수 계산과 구매한 아이템의 총 수량 계산의 결과를 합쳐서 `user_rf` 라는 이름의 테이블에 저장하기

```
CREATE OR REPLACE TABLE `project-dac3b643-036f-483e-a72.modulabs_project.user_rf` AS
```

```
-- (1) 전체 거래 건수 계산
WITH purchase_cnt AS (
  select customerID, count(distinct invoiceno) as purchase_cnt
  from `project-dac3b643-036f-483e-a72.modulabs_project.data`
  group by customerID
),
```

```
-- (2) 구매한 아이템 총 수량 계산
item_cnt AS (
  select customerID, sum(Quantity) as item_cnt
  from `project-dac3b643-036f-483e-a72.modulabs_project.data`
  group by customerID
)
```

```
-- 기존의 user_r에 (1)과 (2)를 통합
SELECT
  pc.CustomerID,
  pc.purchase_cnt,
  ic.item_cnt,
  ur.recovery
FROM purchase_cnt AS pc
JOIN item_cnt AS ic
```

```
ON pc.CustomerID = ic.CustomerID
JOIN `project-dac3b643-036f-483e-a72.modulabs_project.user_r` AS ur
ON pc.CustomerID = ur.CustomerID;
```

project_mainquest_03 실행 쿼리 저장 다운로드 공유 일정

```
1 CREATE OR REPLACE TABLE `project-dac3b643-036f-483e-a72.modulabs_project.user_rf` AS
2
3 -- (1) 전체 거래 건수 계산
4 WITH purchase_cnt AS (
5   select customerID, count(distinct invoiceno) as purchase_cnt
6   from `project-dac3b643-036f-483e-a72.modulabs_project.data`
7   group by customerID
8 )
9
```

주문형 처리 할당량 사용 중

쿼리 결과

작업 정보 결과 실행 세부정보 실행 그래프

이 문으로 이름이 user_rf인 테이블이 교체되었습니다.

Monetary

- 고객별 총 지출액 계산 (소수점 첫째 자리에서 반올림)

```
select CustomerID, round(sum(Quantity * UnitPrice), 0) as user_total
from `project-dac3b643-036f-483e-a72.modulabs_project.data`
group by CustomerID
having user_total != 0
```

제목 없는 쿼리 실행 쿼리 저장 다운로드 공유 일정 다음에서 열기 더보기

```
1 select CustomerID, round(sum(Quantity * UnitPrice), 0) as user_total
2 from `project-dac3b643-036f-483e-a72.modulabs_project.data`
3 group by CustomerID
4 having user_total != 0
```

쿼리 완료됨

주문형 처리 할당량 사용 중

쿼리 결과

작업 정보 결과 시각화 JSON 실행 세부정보 실행 그래프

행	CustomerID	user_total
1	12347	4310.0
2	12348	1437.0
3	12349	1458.0
4	12350	294.0
5	12352	1265.0
6	12353	89.0
7	12354	1079.0
8	12355	459.0
9	12356	2487.0
10	12357	6208.0
11	12358	928.0
12	12359	6183.0
13	12360	2302.0
14	12361	175.0
15	12362	4655.0

페이지당 결과 수: 50 1 ~ 50 (전체 4349행)

- 고객별 평균 거래 금액 계산

- 고객별 평균 거래 금액을 구하기 위해 1) data 테이블을 user_rf 테이블과 조인(LEFT JOIN) 한 후, 2) purchase_cnt로 나누어서 3) user_rfm 테이블로 저장하기

```
CREATE OR REPLACE TABLE `project-dac3b643-036f-483e-a72.modulabs_project.user_rfm` AS
SELECT
  rf.CustomerID AS CustomerID,
  rf.purchase_cnt,
  rf.item_cnt,
```

```

rf.recenty,
ut.user_total,
round((ut.user_total/rf.purchase_cnt), 0) as user_average
FROM `project-dac3b643-036f-483e-a72.modulabs_project.user_rf` as rf
LEFT JOIN (
  select CustomerID, round(sum(Quantity * UnitPrice), 0) as user_total
  from `project-dac3b643-036f-483e-a72.modulabs_project.data`
  group by CustomerID
  having user_total != 0) as ut
ON rf.CustomerID = ut.CustomerID;

```

제목 없는 쿼리

실행 저장 다운로드 공유 일정 다음에서 열기

```

1 CREATE OR REPLACE TABLE `project-dac3b643-036f-483e-a72.modulabs_project.user_rfm` AS
2 SELECT
3   rf.CustomerID AS CustomerID,
4   rf.purchase_cnt,
5   rf.item_cnt,
6   rf.recenty,
7   ut.user_total,
8   round((ut.user_total/rf.purchase_cnt), 0) as user_average
9 FROM `project-dac3b643-036f-483e-a72.modulabs_project.user_rf` as rf
10 LEFT JOIN (

```

주문형 처리 할당량 사용 중

쿼리 결과

작업 정보 결과 실행 세부정보 실행 그래프

이 문으로 이름이 user_rfm인 테이블이 교체되었습니다.

RFM 통합 테이블 출력하기

- 최종 user_rfm 테이블을 출력하기

```

select *
from `project-dac3b643-036f-483e-a72.modulabs_project.user_rfm`

```

project_mainquest_03

실행 일정 다음에서 열기

실행 세부정보 실행 그래프

쿼리 결과

작업 정보 결과 실행 세부정보 실행 그래프

일	CustomerID	purchase_cnt	item_cnt	recenty	user_total	user_average
1	12713	1	505	0	795.0	795.0
2	14569	1	79	1	227.0	227.0
3	15520	1	314	1	343.0	343.0
4	13436	1	76	1	197.0	197.0
5	13298	1	96	1	360.0	360.0
6	15195	1	1404	2	3861.0	3861.0
7	14204	1	72	2	151.0	151.0
8	15471	1	255	2	448.0	448.0
9	15992	1	17	3	42.0	42.0
10	14578	1	240	3	169.0	169.0
11	12478	1	233	3	546.0	546.0
12	17914	1	457	3	529.0	529.0
13	12650	1	260	3	242.0	242.0
14	14514	1	145	3	315.0	315.0

메시지당 결과 수: 50 1 - 50 (전체 4362명) < >

11-8. 추가 Feature 추출

1. 구매하는 제품의 다양성

- 1) 고객 별로 구매한 상품들의 고유한 수를 계산하기
- 2) `user_rfm` 테이블과 결과를 합치기
- 3) `user_data` 라는 이름의 테이블에 저장하기

```
CREATE OR REPLACE TABLE `project-dac3b643-036f-483e-a72.modulabs_project.user_data` AS
WITH unique_products AS (
  SELECT
    CustomerID,
    COUNT(DISTINCT StockCode) AS unique_products
  FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`
  GROUP BY CustomerID
)
SELECT ur.*, up.* EXCEPT (CustomerID)
FROM `project-dac3b643-036f-483e-a72.modulabs_project.user_rfm` AS ur
JOIN unique_products AS up
ON ur.CustomerID = up.CustomerID;
```

제목 없는 쿼리 실행 저장 다운로드 공유 일정 다음에서 열기

```
1 CREATE OR REPLACE TABLE `project-dac3b643-036f-483e-a72.modulabs_project.user_data` AS
2 WITH unique_products AS (
3   SELECT
4     CustomerID,
5     COUNT(DISTINCT StockCode) AS unique_products
6   FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`
7   GROUP BY CustomerID
8 )
9 SELECT ur.*, up.* EXCEPT (CustomerID)
10 FROM `project-dac3b643-036f-483e-a72.modulabs_project.user_rfm` AS ur
11 JOIN unique_products AS up
12 ON ur.CustomerID = up.CustomerID;
```

쿼리 완료됨

주문형 처리 할당량 사용 중

쿼리 결과

작업 정보 결과 실행 세부정보 실행 그래프

이 문으로 이름이 user_data인 새 테이블이 생성되었습니다.

2. 평균 구매 주기

- 고객들의 쇼핑 패턴을 이해하는 것을 목표 (고객 별 재방문 주기 살펴보기)
 - 균 구매 소요 일수를 계산하고, 그 결과를 `user_data` 에 통합

```
CREATE OR REPLACE TABLE `project-dac3b643-036f-483e-a72.modulabs_project.user_data` AS
WITH purchase_intervals AS (
  -- (2) 고객 별 구매와 구매 사이의 평균 소요 일수
  SELECT
    CustomerID,
    CASE WHEN ROUND(AVG(interval_), 2) IS NULL THEN 0 ELSE ROUND(AVG(interval_), 2) END AS average_interval
  FROM (
    -- (1) 구매와 구매 사이에 소요된 일수
    SELECT
      CustomerID,
      DATE_DIFF(InvoiceDate, LAG(InvoiceDate) OVER (PARTITION BY CustomerID ORDER BY InvoiceDate), DAY) AS interval_
    FROM
      `project-dac3b643-036f-483e-a72.modulabs_project.data`
    WHERE CustomerID IS NOT NULL
  )
)
```

```
GROUP BY CustomerID
)
```

```
SELECT u.*, pi.* EXCEPT (CustomerID)
FROM `project-dac3b643-036f-483e-a72.modulabs_project.user_data` AS u
LEFT JOIN purchase_intervals AS pi
ON u.CustomerID = pi.CustomerID;
```



3. 구매 취소 경향성

- 고객의 취소 패턴 파악하기
 - 취소 빈도(cancel_frequency) : 고객 별로 취소한 거래의 총 횟수
 - 취소 비율(cancel_rate) : 각 고객이 한 모든 거래 중에서 취소를 한 거래의 비율
 - 취소 빈도와 취소 비율을 계산하고 그 결과를 `user_data`에 통합하기 (취소 비율은 소수점 두번째 자리)

```
CREATE OR REPLACE TABLE `project-dac3b643-036f-483e-a72.modulabs_project.user_data` AS
```

```
WITH TransactionInfo AS (
```

```
SELECT
```

```
  CustomerID,
```

```
  COUNT(DISTINCT invoiceno) AS total_transactions,
```

```
  COUNT(DISTINCT CASE WHEN invoiceno LIKE 'C%' THEN invoiceno END) AS cancel_frequency
```

```
FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`
```

```
WHERE customerID IS NOT NULL
```

```
GROUP BY customerID
```

```
)
```

```
SELECT u.*, t.* EXCEPT(CustomerID), ROUND(t.cancel_frequency * 1.0 / t.total_transactions, 2) as cancel_rate
```

```
FROM `project-dac3b643-036f-483e-a72.modulabs_project.user_data` AS u
```

```
LEFT JOIN TransactionInfo AS t
```

```
ON u.CustomerID = t.CustomerID;
```

project_mainquest_03 실행 일정 다음에서 열기 더보기 쿼리 저장

```

5 CustomerID,
6 COUNT(DISTINCT invoiceno) AS total_transactions,
7 COUNT(DISTINCT CASE WHEN invoiceno LIKE 'C%' THEN invoiceno END) AS cancel_frequency
8 FROM `project-dac3b643-036f-483e-a72.modulabs_project.data`
9 WHERE customerID IS NOT NULL
10 GROUP BY customerID
11
12
13 SELECT u.* t.* EXCEPT(customerID) ROUND(t.cancel_frequency * 1.0 / t.total_transactions, 2) AS cancel_rate

```

주문형 처리 할당량 사용 중

쿼리 결과

작업 정보 결과 실행 세부정보 실행 그래프

이 문으로 이름이 user_data인 테이블이 교체되었습니다.

- 다양한 컬럼들을 활용하여 고객의 구매 패턴과 선호도를 보다 심층적으로 이해할 수 있도록 최종적으로 **user_data** 를 출력하기

```

select *
from `project-dac3b643-036f-483e-a72.modulabs_project.user_data`

```

제목 없는 쿼리 실행 저장 다운로드 공유 일정 다음에서 열기 더보기

```

1 select *
2 from `project-dac3b643-036f-483e-a72.modulabs_project.user_data`

```

실행 시 이 쿼리가 374.66KB를 처리합니다.
주문형 처리 할당량 사용 중

쿼리 결과 결과 저장 다음에서 열기

작업 정보 결과 시각화 JSON 실행 세부정보 실행 그래프

row	CustomerID	purchase_cnt	item_cnt	frequency	user_total	user_average	unique_products	average_interval	total_transactions	cancel_frequency	cancel_rate
1	15944	1	50	276	325.0	325.0	6	0.0	1	0	0.0
2	17647	1	30	65	133.0	133.0	7	0.0	1	0	0.0
3	14699	1	77	214	109.0	109.0	8	0.0	1	0	0.0
4	16849	1	33	186	125.0	125.0	8	0.0	1	0	0.0
5	15447	1	85	330	155.0	155.0	9	0.0	1	0	0.0
6	15663	1	106	106	138.0	138.0	9	0.0	1	0	0.0
7	13581	1	30	309	118.0	118.0	10	0.0	1	0	0.0
8	17205	1	884	53	384.0	384.0	13	0.0	1	0	0.0
9	15143	1	150	225	259.0	259.0	14	0.0	1	0	0.0
10	12552	1	151	30	314.0	314.0	15	0.0	1	0	0.0
11	13127	1	127	77	259.0	259.0	15	0.0	1	0	0.0
12	12614	1	76	277	316.0	316.0	20	0.0	1	0	0.0
13	17985	1	171	22	631.0	631.0	20	0.0	1	0	0.0
14	17194	1	112	360	227.0	227.0	25	0.0	1	0	0.0
15	12729	1	231	113	417.0	417.0	25	0.0	1	0	0.0
16	12920	1	128	17	163.0	163.0	27	0.0	1	0	0.0

페이지당 결과 수: 50 1 - 50 (전체 4362명) |< > >>

회고

Keep : 최대한 직접 코드를 작성해보며 화면과 다른 값이 나왔을 때 직접 오류를 수정해보는 과정을 겪어보며 어떤 코드를 작성해야하는지 이해한 것 같다.

Problem : 익숙하지 않은 코드를 마주했을 때 집중력이 급저하되어서 시간을 소요한 부분이 있었던 것 같다.

Try : 최대한 기본기를 탄탄하게 쌓기 위해 더 많은 연습 문제를 풀어봐야 할 것 같다. 그리고 문제나 이후에 분석할 때 최대한 어떤 방향을 원하는지 오해 없이 이해하기 위해 문제 및 제안을 어떻게 해결해나갈지 생각하는 시간도 많이 가져야 할 것 같았다.

▼ 아카이브

X

X