고객을 세그먼테이션하자 [프로젝트]

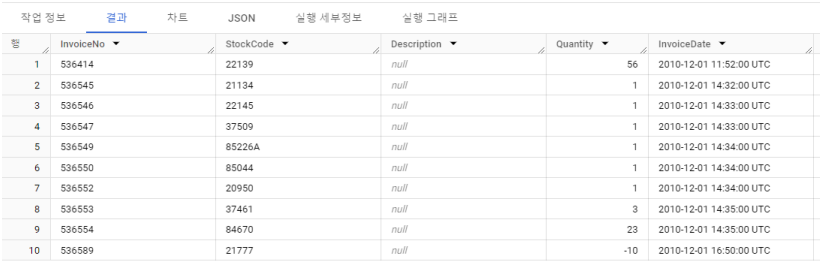
11-2. 데이터 불러오기

데이터 살펴보기

- **테이블에 있는 10개의 행만 출력하기**

```
select *
from trim-mix-439401-g3.modulabs_project.data
limit 10;
```

[결과 이미지를 넣어주세요]



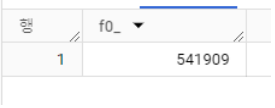
작업 정보 | 결과 | 차트 | JSON | 실행 세부정보 | 실행 그래프

행	InvoiceNo	StockCode	Description	Quantity	InvoiceDate
1	536414	22139	null	56	2010-12-01 11:52:00 UTC
2	536545	21134	null	1	2010-12-01 14:32:00 UTC
3	536546	22145	null	1	2010-12-01 14:33:00 UTC
4	536547	37509	null	1	2010-12-01 14:33:00 UTC
5	536549	85226A	null	1	2010-12-01 14:34:00 UTC
6	536550	85044	null	1	2010-12-01 14:34:00 UTC
7	536552	20950	null	1	2010-12-01 14:34:00 UTC
8	536553	37461	null	3	2010-12-01 14:35:00 UTC
9	536554	84670	null	23	2010-12-01 14:35:00 UTC
10	536589	21777	null	-10	2010-12-01 16:50:00 UTC

- **전체 데이터는 몇 행으로 구성되어 있는지 확인하기**

```
select count(*)
from trim-mix-439401-g3.modulabs_project.data
;
```

[결과 이미지를 넣어주세요]



행	f0_
1	541909

데이터 수 세기

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

```
select
  count(distinct InvoiceNo) as InvoiceNo
  , count(distinct StockCode) as StockCode
  , count(distinct Description) as Description
  , count(distinct Quantity) as Quantity
  , count(distinct InvoiceDate) as InvoiceDate
  , count(distinct UnitPrice) as UnitPrice
  , count(distinct CustomerID) as CustomerID
  , count(distinct Country) as Country
from trim-mix-439401-g3.modulabs_project.data
;
```

[결과 이미지를 넣어주세요]

1	25900	4070	4223	722	23260	1630	4372	38

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(SUM(CASE WHEN InvoiceNo IS NULL THEN 1 ELSE 0 END) / COUNT(*) * 100, 2) AS missing_percent
FROM trim-mix-439401-g3.modulabs_project.data
union all
SELECT
    'StockCode' AS column_name,
    ROUND(SUM(CASE WHEN StockCode IS NULL THEN 1 ELSE 0 END) / COUNT(*) * 100, 2) AS missing_percent
FROM trim-mix-439401-g3.modulabs_project.data
union all
SELECT
    'Description' AS column_name,
    ROUND(SUM(CASE WHEN Description IS NULL THEN 1 ELSE 0 END) / COUNT(*) * 100, 2) AS missing_perce
FROM trim-mix-439401-g3.modulabs_project.data
union all
SELECT
    'Quantity' AS column_name,
    ROUND(SUM(CASE WHEN Quantity IS NULL THEN 1 ELSE 0 END) / COUNT(*) * 100, 2) AS missing_percenta
FROM trim-mix-439401-g3.modulabs_project.data
union all
SELECT
    'InvoiceDate' AS column_name,
    ROUND(SUM(CASE WHEN InvoiceDate IS NULL THEN 1 ELSE 0 END) / COUNT(*) * 100, 2) AS missing_perce
FROM trim-mix-439401-g3.modulabs_project.data
union all
SELECT
    'UnitPrice' AS column_name,
    ROUND(SUM(CASE WHEN UnitPrice IS NULL THEN 1 ELSE 0 END) / COUNT(*) * 100, 2) AS missing_percent
FROM trim-mix-439401-g3.modulabs_project.data
union all
SELECT
    'CustomerID' AS column_name,
    ROUND(SUM(CASE WHEN CustomerID IS NULL THEN 1 ELSE 0 END) / COUNT(*) * 100, 2) AS missing_percen
FROM trim-mix-439401-g3.modulabs_project.data
union all
SELECT
    'Country' AS column_name,
    ROUND(SUM(CASE WHEN Country IS NULL THEN 1 ELSE 0 END) / COUNT(*) * 100, 2) AS missing_percentag
FROM trim-mix-439401-g3.modulabs_project.data
```

[결과 이미지를 넣어주세요]

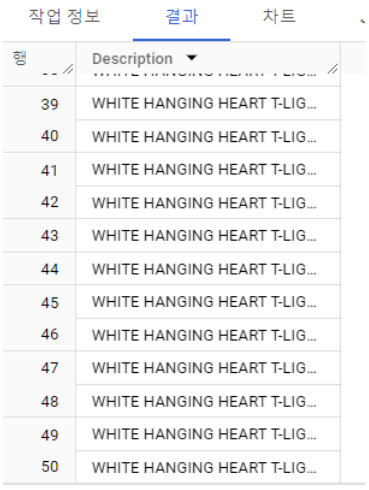
행	column_name	missing_percentage
1	CustomerID	24.93
2	Country	0.0
3	InvoiceNo	0.0
4	Description	0.27
5	Quantity	0.0
6	StockCode	0.0
7	UnitPrice	0.0
8	InvoiceDate	0.0

결측치 처리 전략

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

```
SELECT
    Description
FROM trim-mix-439401-g3.modulabs_project.data
where StockCode = '85123A'
;
```

[결과 이미지를 넣어주세요]

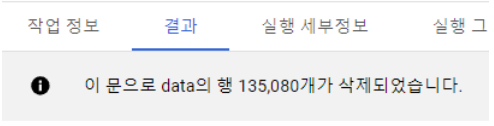


결측치 처리

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

```
delete
from trim-mix-439401-g3.modulabs_project.data
where CustomerID is null
or Description is null;
```

[결과 이미지를 넣어주세요]



11-5. 데이터 전처리(2): 중복값 처리

컬럼 별 누락된 값의 비율 계산 - 위 DELETE문 실행 후

- **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(SUM(CASE WHEN InvoiceNo IS NULL THEN 1 ELSE 0 END) / COUNT(*) * 100, 2) AS missing_percent
FROM trim-mix-439401-g3.modulabs_project.data
union all
SELECT
    'StockCode' AS column_name,
    ROUND(SUM(CASE WHEN StockCode IS NULL THEN 1 ELSE 0 END) / COUNT(*) * 100, 2) AS missing_percent
FROM trim-mix-439401-g3.modulabs_project.data
union all
SELECT
    'Description' AS column_name,
```

```
    ROUND(SUM(CASE WHEN Description IS NULL THEN 1 ELSE 0 END) / COUNT(*) * 100, 2) AS missing_perce
FROM trim-mix-439401-g3.modulabs_project.data
union all
SELECT
    'Quantity' AS column_name,
    ROUND(SUM(CASE WHEN Quantity IS NULL THEN 1 ELSE 0 END) / COUNT(*) * 100, 2) AS missing_percenta
FROM trim-mix-439401-g3.modulabs_project.data
union all
SELECT
    'InvoiceDate' AS column_name,
    ROUND(SUM(CASE WHEN InvoiceDate IS NULL THEN 1 ELSE 0 END) / COUNT(*) * 100, 2) AS missing_perce
FROM trim-mix-439401-g3.modulabs_project.data
union all
SELECT
    'UnitPrice' AS column_name,
    ROUND(SUM(CASE WHEN UnitPrice IS NULL THEN 1 ELSE 0 END) / COUNT(*) * 100, 2) AS missing_percent
FROM trim-mix-439401-g3.modulabs_project.data
union all
SELECT
    'CustomerID' AS column_name,
    ROUND(SUM(CASE WHEN CustomerID IS NULL THEN 1 ELSE 0 END) / COUNT(*) * 100, 2) AS missing_percen
FROM trim-mix-439401-g3.modulabs_project.data
union all
SELECT
    'Country' AS column_name,
    ROUND(SUM(CASE WHEN Country IS NULL THEN 1 ELSE 0 END) / COUNT(*) * 100, 2) AS missing_percentag
FROM trim-mix-439401-g3.modulabs_project.data
```

[결과 이미지를 넣어주세요]

	column_name	missing_percentage
1	Country	0.0
2	CustomerID	0.0
3	Description	0.0
4	InvoiceNo	0.0
5	UnitPrice	0.0
6	InvoiceDate	0.0
7	StockCode	0.0
8	Quantity	0.0

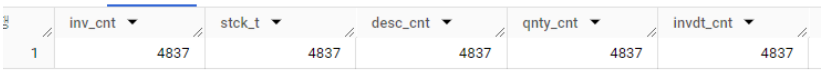
중복값 확인

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

```
select
  sum( case when InvoiceNoCnt > 1 then 1 else 0 end ) as inv_cnt
  , sum( case when StockCodeCnt > 1 then 1 else 0 end ) as stck_t
  , sum( case when DescriptionCnt > 1 then 1 else 0 end ) as desc_cnt
  , sum( case when QuantityCnt > 1 then 1 else 0 end ) as qnty_cnt
  , sum( case when InvoiceDateCnt > 1 then 1 else 0 end ) as invdt_cnt
  , sum( case when UnitPriceCnt > 1 then 1 else 0 end ) as untprc_cnt
  , sum( case when CustomerIDCnt > 1 then 1 else 0 end ) as cstm_cnt
  , sum( case when CountryCnt > 1 then 1 else 0 end ) as cntry_cnt
from
(
  select
    count( InvoiceNo) as InvoiceNoCnt
    , count( StockCode) as StockCodeCnt
   , count( Description) as DescriptionCnt
   , count( Quantity) as QuantityCnt
   , count( InvoiceDate) as InvoiceDateCnt
```

```
    , count( UnitPrice) as UnitPriceCnt
    , count( CustomerID) as CustomerIDCnt
    , count( Country) as CountryCnt
  from trim-mix-439401-g3.modulabs_project.data
  --group by grouping sets(InvoiceNo, StockCode, Description,Quantity,InvoiceDate,UnitPrice,Customer
  group by InvoiceNo, StockCode, Description,Quantity,InvoiceDate,UnitPrice,CustomerID,Country
)
;
```

[결과 이미지를 넣어주세요]



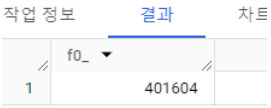
	inv_cnt	stck_t	desc_cnt	qnty_cnt	invdt_cnt
1	4837	4837	4837	4837	4837

중복값 처리

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

```
CREATE OR REPLACE TABLE trim-mix-439401-g3.modulabs_project.data
as
 select
   distinct InvoiceNo
   ,StockCode
   , Description
   , Quantity
  , InvoiceDate
  , UnitPrice
  , CustomerID
  , Country
  from trim-mix-439401-g3.modulabs_project.data
  group by InvoiceNo, StockCode, Description,Quantity,InvoiceDate,UnitPrice,CustomerID,Country
;
```

[결과 이미지를 넣어주세요]



	f0_
1	401604

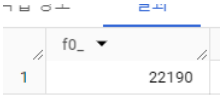
11-6. 데이터 전처리(3): 오류값 처리

`InvoiceNo` 살펴보기

- **고유(unique)한 `InvoiceNo` 의 개수를 출력하기**

```
select  count(distinct InvoiceNo) from trim-mix-439401-g3.modulabs_project.data;
```

[결과 이미지를 넣어주세요]

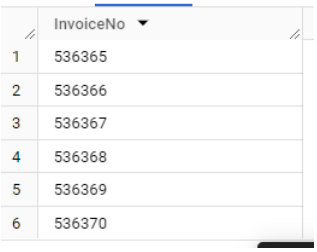


	f0_
1	22190

- **고유한 `InvoiceNo` 를 앞에서부터 100개를 출력하기**

```
select  distinct InvoiceNo
from trim-mix-439401-g3.modulabs_project.data
order by 1
limit 100;
```

[결과 이미지를 넣어주세요]

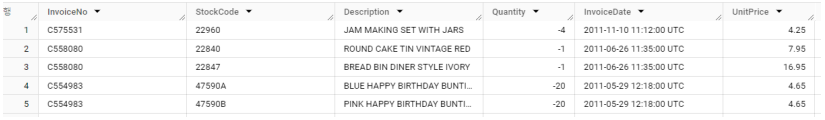


	InvoiceNo
1	536365
2	536366
3	536367
4	536368
5	536369
6	536370

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

```
select *
from trim-mix-439401-g3.modulabs_project.data
where InvoiceNo like 'C%'
limit 100;
```

[결과 이미지를 넣어주세요]

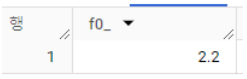


행	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice
1	C575531	22960	JAM MAKING SET WITH JARS	-4	2011-11-10 11:12:00 UTC	4.25
2	C558080	22840	ROUND CAKE TIN VINTAGE RED	-1	2011-06-26 11:35:00 UTC	7.95
3	C558080	22847	BREAD BIN DINER STYLE IVORY	-1	2011-06-26 11:35:00 UTC	16.95
4	C554983	47590A	BLUE HAPPY BIRTHDAY BUNTI...	-20	2011-05-29 12:18:00 UTC	4.65
5	C554983	47590B	PINK HAPPY BIRTHDAY BUNTI...	-20	2011-05-29 12:18:00 UTC	4.65

- **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)
from trim-mix-439401-g3.modulabs_project.data
;
```

[결과 이미지를 넣어주세요]



행	f0_
1	2.2

`StockCode` 살펴보기

- **고유한 `StockCode`의 개수를 출력하기**

```
select  COUNT(distinct StockCode)
from trim-mix-439401-g3.modulabs_project.data
;
```

[결과 이미지를 넣어주세요]

행	f0_
1	3684

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

```
SELECT StockCode, COUNT(*) AS sell_cnt
FROM trim-mix-439401-g3.modulabs_project.data
GROUP BY StockCode
ORDER BY sell_cnt DESC
;
```

[결과 이미지를 넣어주세요]

	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
11	20727	1099
12	22383	1083

- `StockCode`**의 컬럼에 있던 값 중에서 숫자를 제외한 문자만 남기고 문자가 몇 자리 수 인지 세고**
  - **숫자가 0~1개인 값**들에는 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 trim-mix-439401-g3.modulabs_project.data
)
WHERE number_count IN (0, 1)
```

[결과 이미지를 넣어주세요]

	StockCode	number_count
1	POST	0
2	M	0
3	PADS	0
4	D	0
5	BANK CHARGES	0
6	DOT	0
7	CRUK	0
8	C2	1

- `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 퍼센트**인지 구하기 (소수점 두 번째 자리까지)

```
WITH UniqueStockCodes AS (
  SELECT DISTINCT StockCode
  FROM trim-mix-439401-g3.modulabs_project.data
)
SELECT
  LENGTH(StockCode) - LENGTH(REGEXP_REPLACE(StockCode, r'[A-Za-z]', '')) AS literal_count,
  COUNT(*) AS stock_cnt
```

```
FROM UniqueStockCodes
GROUP BY literal_count
ORDER BY stock_cnt DESC;



SELECT
  round(SUM(number_count) / (SELECT COUNT(*) AS cnt FROM trim-mix-439401-g3.modulabs_project.data) *
FROM (
    SELECT StockCode,
      CASE WHEN ( LENGTH(StockCode) - LENGTH(REGEXP_REPLACE(StockCode, r'[0-9]', '')) ) IN (0, 1) TH
    FROM trim-mix-439401-g3.modulabs_project.data
  ) a
  where 1=1
 ;
```

[결과 이미지를 넣어주세요]

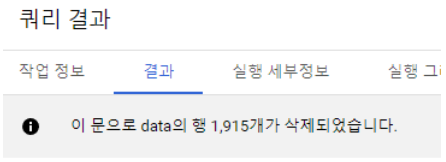
행	literal_count	stock_cnt
1	0	2798
2	1	880
3	4	3
4	2	1
5	11	1
6	3	1

행	f0_
1	0.48

- **제품과 관련되지 않은 거래 기록을 제거하기**

```
DELETE
FROM trim-mix-439401-g3.modulabs_project.data
WHERE   StockCode IN
(
  SELECT StockCode
  FROM trim-mix-439401-g3.modulabs_project.data
  WHERE LENGTH(StockCode) - LENGTH(REGEXP_REPLACE(StockCode, r'[0-9]', '')) IN (0,1)
);
```

[결과 이미지를 넣어주세요]



`Description` 살펴보기

- **고유한 Description 별 출현 빈도를 계산하고 상위 30개를 출력하기**

```
SELECT Description, COUNT(*) AS description_cnt
FROM trim-mix-439401-g3.modulabs_project.data
GROUP BY Description
ORDER BY COUNT(*) DESC
LIMIT 30
```

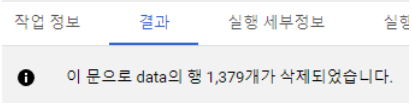
[결과 이미지를 넣어주세요]

	Description	description_cnt
1	WHITE HANGING HEART T-LIG...	2058
2	REGENCY CAKESTAND 3 TIER	1894
3	JUMBO BAG RED RETROSPOT	1659
4	PARTY BUNTING	1409
5	ASSORTED COLOUR BIRD ORN...	1405
6	LUNCH BAG RED RETROSPOT	1345
7	SET OF 3 CAKE TINS PANTRY ...	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

- **서비스 관련 정보를 포함하는 행들을 제거하기**

```
DELETE FROM trim-mix-439401-g3.modulabs_project.data
WHERE REGEXP_CONTAINS(Description, r'[a-z]');
```

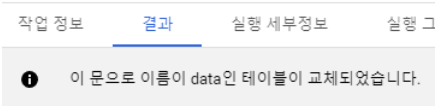
[결과 이미지를 넣어주세요]



- **대소문자를 혼합하고 있는 데이터를 대문자로 표준화 하기**

```
CREATE OR REPLACE TABLE trim-mix-439401-g3.modulabs_project.data AS
SELECT
  * EXCEPT (Description)
  ,UPPER(Description) AS Description
FROM trim-mix-439401-g3.modulabs_project.data
```

[결과 이미지를 넣어주세요]



`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 trim-mix-439401-g3.modulabs_project.data
;
```

[결과 이미지를 넣어주세요]

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(InvoiceNo) AS cnt_quantity
   ,MIN(Quantity) min_quantity
   ,MAX(Quantity) AS max_quantity
   , AVG(Quantity) AS avg_quantity
FROM trim-mix-439401-g3.modulabs_project.data
WHERE UnitPrice = 0
;
```

[결과 이미지를 넣어주세요]

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

- **`UnitPrice = 0`를 제거하고 일관된 데이터셋을 유지하기**

```
delete FROM trim-mix-439401-g3.modulabs_project.data
WHERE UnitPrice = 0;
```

[결과 이미지를 넣어주세요]

작업 정보 | 결과 | 실행 세부정보

이 문으로 data의 행 33개가 삭제되었습니다.

11-7. RFM 스코어

Recency

- **`InvoiceDate` 컬럼을 연월일 자료형으로 변경하기**

```
SELECT DATE(InvoiceDate) AS InvoiceDate_only_date
FROM trim-mix-439401-g3.modulabs_project.data
;
```

[결과 이미지를 넣어주세요]

	InvoiceDate_only_da
1	2010-12-21
2	2011-08-11
3	2011-07-28
4	2011-11-18
5	2010-12-05
6	2011-11-04
7	2011-05-20
8	2011-11-03
9	2011-08-26

- **가장 최근 구매 일자를 MAX() 함수로 찾아보기**

```
SELECT MAX(DATE(InvoiceDate)) AS most_recent_date
FROM trim-mix-439401-g3.modulabs_project.data
;
```

[결과 이미지를 넣어주세요]

행	most_recent_date
1	2011-12-09

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

```
SELECT CustomerID, MAX(DATE(InvoiceDate)) AS most_recent_date
FROM trim-mix-439401-g3.modulabs_project.data
GROUP BY CustomerID
;
```

[결과 이미지를 넣어주세요]

	CustomerID	most_recent_date
1	14911	2011-08-11
2	12507	2011-07-28
3	12444	2011-11-18
4	12647	2010-12-05
5	12431	2011-11-04
6	12415	2011-11-03
7	14646	2011-08-26
8	12457	2011-04-14
9	15107	2011-01-13
10	13985	2011-11-07
11	16818	2011-07-26

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

```
SELECT
  b.CustomerID, DATE_DIFF( RCDT.most_recent_date, date(b.InvoiceDate) , DAY) as diff
FROM (
  SELECT CustomerID, MAX(DATE(InvoiceDate)) AS most_recent_date
  FROM trim-mix-439401-g3.modulabs_project.data
  GROUP BY CustomerID
) RCDT , trim-mix-439401-g3.modulabs_project.data b
WHERE RCDT.CustomerID = b.CustomerID
```

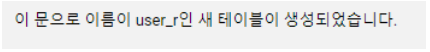
[결과 이미지를 넣어주세요]

행	CustomerID	diff
1	12544	7
2	12544	7
3	12544	7
4	12544	7
5	12544	7
6	12544	7
7	12544	7
8	12544	7
9	12544	7
10	12544	7

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

```
CREATE OR REPLACE TABLE trim-mix-439401-g3.modulabs_project.user_r AS
SELECT
  b.CustomerID, DATE_DIFF( RCDT.most_recent_date, date(b.InvoiceDate) , DAY) as diff
FROM (
  SELECT CustomerID, MAX(DATE(InvoiceDate)) AS most_recent_date
  FROM trim-mix-439401-g3.modulabs_project.data
  GROUP BY CustomerID
) RCDT , trim-mix-439401-g3.modulabs_project.data b
WHERE RCDT.CustomerID = b.CustomerID
;
```

[결과 이미지를 넣어주세요]



Frequency

- **고객마다 고유한 InvoiceNo의 수를 세어보기**

```
SELECT CustomerID , COUNT(DISTINCT InvoiceNo)
FROM trim-mix-439401-g3.modulabs_project.data
GROUP BY CustomerID
;
```

[결과 이미지를 넣어주세요]

행	CustomerID	f0_
1	12544	2
2	13568	1
3	13824	5
4	14080	1
5	14336	4
6	14592	3
7	15104	3
8	15360	1
9	15872	2
10	16128	5
11	16384	2

- **각 고객 별로 구매한 아이템의 총 수량 더하기**

```
SELECT CustomerID , SUM(Quantity)
FROM trim-mix-439401-g3.modulabs_project.data
GROUP BY CustomerID
;
```

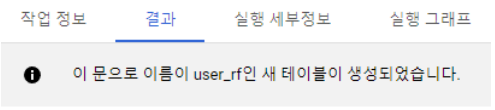
[결과 이미지를 넣어주세요]

행	CustomerID	f0_
1	12544	130
2	13568	66
3	13824	768
4	14080	48
5	14336	1759
6	14592	407
7	15104	633
8	15360	199
9	15872	187
10	16128	976
11	16384	260

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

```
CREATE OR REPLACE TABLE trim-mix-439401-g3.modulabs_project.user_rf AS
WITH purchase_cnt AS (
  SELECT CustomerID , COUNT(DISTINCT InvoiceNo) AS purchase_cnt
  FROM trim-mix-439401-g3.modulabs_project.data
  GROUP BY CustomerID
),
item_cnt AS (
  SELECT CustomerID , SUM(Quantity) AS item_cnt
  FROM trim-mix-439401-g3.modulabs_project.data
  GROUP BY CustomerID
)
SELECT
  pc.CustomerID,
  pc.purchase_cnt,
  ic.item_cnt,
  ur.diff AS recency
FROM purchase_cnt AS pc
JOIN item_cnt AS ic
  ON pc.CustomerID = ic.CustomerID
JOIN trim-mix-439401-g3.modulabs_project.user_r AS ur
  ON pc.CustomerID = ur.CustomerID
  ;
```

[결과 이미지를 넣어주세요]



Monetary

- **고객별 총 지출액 계산 (소수점 첫째 자리에서 반올림)**

```
SELECT
  CustomerID
  , ROUND( SUM(Quantity * UnitPrice) , 1) PRCHS
FROM trim-mix-439401-g3.modulabs_project.data
GROUP BY CustomerID
;
```

[결과 이미지를 넣어주세요]

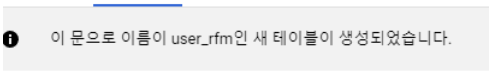
	CustomerID	PRCHS
1	12544	299.7
2	13568	187.1
3	13824	1698.9
4	14080	45.6
5	14336	1614.9
6	14592	557.9
7	15104	968.6
8	15360	417.9
9	15872	316.2
10	16128	1855.0
11	16384	584.5

- **고객별 평균 거래 금액 계산**

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

```
CREATE OR REPLACE TABLE trim-mix-439401-g3.modulabs_project.user_rfm AS
SELECT
  rf.CustomerID AS CustomerID,
  rf.purchase_cnt,
  rf.item_cnt,
  rf.recency,
  ut.user_total,
  ( ut.user_total / rf.purchase_cnt ) as user_average
FROM trim-mix-439401-g3.modulabs_project.user_rf rf
LEFT JOIN (
  -- 고객 별 총 지출액
  SELECT
    CustomerID
    , ROUND( SUM(Quantity * UnitPrice) , 1) user_total
  FROM trim-mix-439401-g3.modulabs_project.data
  GROUP BY CustomerID
) ut
ON rf.CustomerID = ut.CustomerID;
```

[결과 이미지를 넣어주세요]



RFM 통합 테이블 출력하기

- **최종 user_rfm 테이블을 출력하기**

```
select * from  trim-mix-439401-g3.modulabs_project.user_rfm;
```

[결과 이미지를 넣어주세요]

	CustomerID	purchase_cnt	item_cnt	recency	user_total	user_average
1	12544	2	130	0	299.7	149.85
2	13568	1	66	0	187.1	187.1
3	13568	1	66	0	187.1	187.1
4	13568	1	66	0	187.1	187.1
5	13568	1	66	0	187.1	187.1
6	13568	1	66	0	187.1	187.1
7	13568	1	66	0	187.1	187.1
8	13568	1	66	0	187.1	187.1
9	13568	1	66	0	187.1	187.1
10	13568	1	66	0	187.1	187.1
11	13568	1	66	0	187.1	187.1
12	13568	1	66	0	187.1	187.1

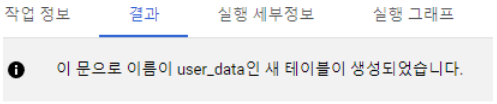
11-8. 추가 Feature 추출

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 trim-mix-439401-g3.modulabs_project.user_data AS
WITH unique_products AS (
  SELECT
    CustomerID,
    COUNT(DISTINCT StockCode) AS unique_products
  FROM trim-mix-439401-g3.modulabs_project.data
  GROUP BY CustomerID
)
SELECT ur.*, up.* EXCEPT (CustomerID)
FROM trim-mix-439401-g3.modulabs_project.user_rfm AS ur
JOIN unique_products AS up
ON ur.CustomerID = up.CustomerID;
```

[결과 이미지를 넣어주세요]

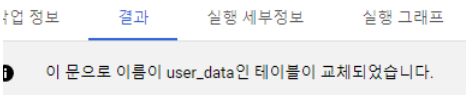


2. 평균 구매 주기

- **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 trim-mix-439401-g3.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_inte
  FROM (
    -- (1) 구매와 구매 사이에 소요된 일수
    SELECT
      CustomerID,
      DATE_DIFF(InvoiceDate, LAG(InvoiceDate) OVER (PARTITION BY CustomerID ORDER BY InvoiceDate), DAY)
    FROM
      trim-mix-439401-g3.modulabs_project.data
    WHERE CustomerID IS NOT NULL
  )
  GROUP BY CustomerID
)
SELECT u.*, pi.* EXCEPT (CustomerID)
FROM trim-mix-439401-g3.modulabs_project.user_data AS u
LEFT JOIN purchase_intervals AS pi
ON u.CustomerID = pi.CustomerID;
```

[결과 이미지를 넣어주세요]



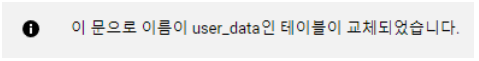
3. 구매 취소 경향성

- **고객의 취소 패턴 파악하기**
  1) 취소 빈도(cancel_frequency) : 고객 별로 취소한 거래의 총 횟수
  2) 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_name.modulabs_project.user_data AS
WITH TransactionInfo AS (
  select
    CustomerID
    , COUNT(InvoiceNo) AS total_transactions
    , sum( cl_frq ) AS cancel_frequency
  from (
      SELECT
        CustomerID
        , InvoiceNo
        , CASE WHEN InvoiceNo LIKE 'C%' THEN 1 ELSE 0 END  AS cl_frq
      FROM trim-mix-439401-g3.modulabs_project.data
  ) group by   CustomerID
)
SELECT u.*, t.* EXCEPT(CustomerID), round(( cancel_frequency / total_transactions ) * 100 , 2) as cance
FROM trim-mix-439401-g3.modulabs_project.user_data AS u
LEFT JOIN TransactionInfo AS t
ON u.CustomerID = t.CustomerID;
```

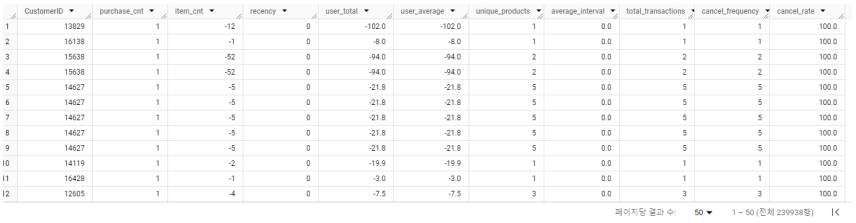
[결과 이미지를 넣어주세요]



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

```
select * from  trim-mix-439401-g3.modulabs_project.user_data
where cancel_frequency > 0;
```

[결과 이미지를 넣어주세요]



	CustomerID	purchase_cnt	item_cnt	recency	user_total	user_average	unique_products	average_interval	total_transactions	cancel_frequency	cancel_rate
1	13829	1	-12	0	-102.0	-102.0	1	0.0	1	1	100.0
2	16138	1	-1	0	-8.0	-8.0	1	0.0	1	1	100.0
3	15638	1	-52	0	-94.0	-94.0	2	0.0	2	2	100.0
4	15638	1	-52	0	-94.0	-94.0	2	0.0	2	2	100.0
5	14627	1	-5	0	-21.8	-21.8	5	0.0	5	5	100.0
6	14627	1	-5	0	-21.8	-21.8	5	0.0	5	5	100.0
7	14627	1	-5	0	-21.8	-21.8	5	0.0	5	5	100.0
8	14627	1	-5	0	-21.8	-21.8	5	0.0	5	5	100.0
9	14627	1	-5	0	-21.8	-21.8	5	0.0	5	5	100.0
10	14119	1	-2	0	-19.9	-19.9	1	0.0	1	1	100.0
11	16428	1	-1	0	-3.0	-3.0	1	0.0	1	1	100.0
12	12605	1	-4	0	-7.5	-7.5	3	0.0	3	3	100.0

페이지당 결과 수: 50 1 – 50 (전체 239938행)

회고

[회고 내용을 작성해