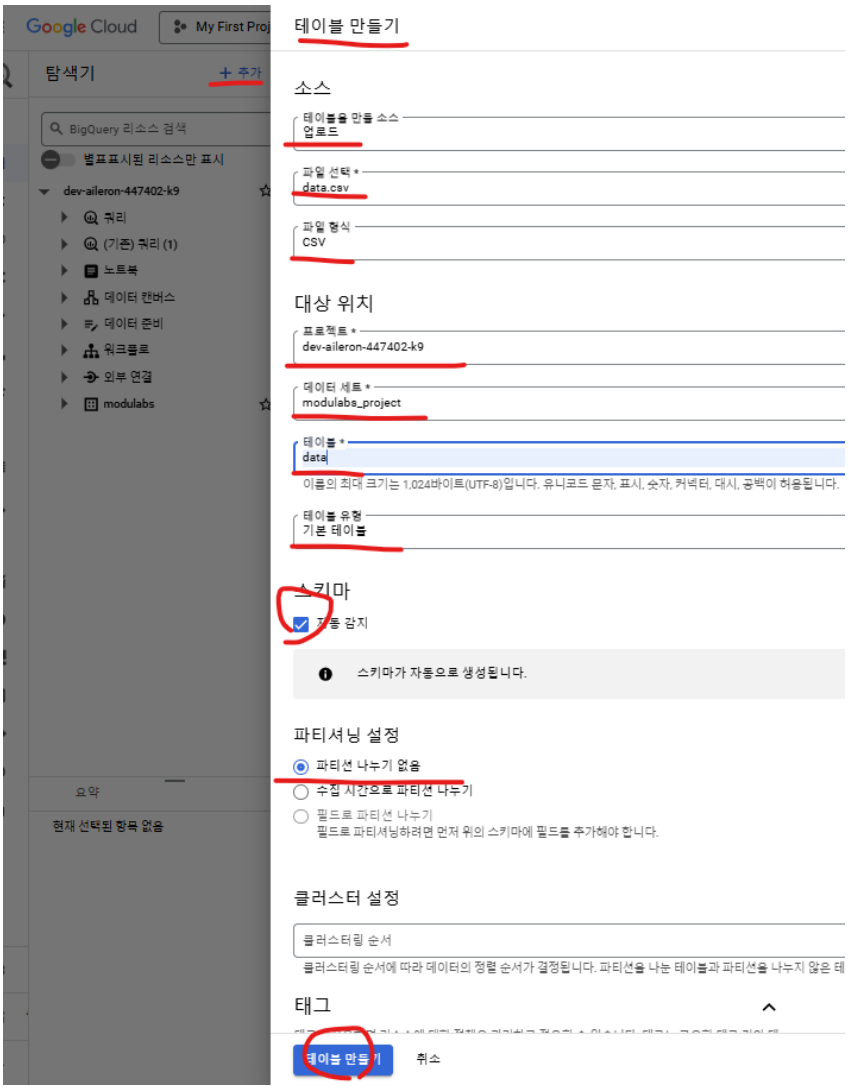
PDA

# 고객을 세그먼테이션하자 [프로젝트]

## 5-2. 데이터 불러오기




### 데이터 살펴보기

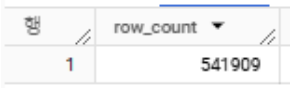
- **테이블에 있는 10개의 행만 출력하기**

```
select *
  from modulabs_project.data
  limit 10;
```

| 행 | InvoiceNo | StockCode | Description | Quantity | InvoiceDate | UnitPrice | CustomerID | Country |
|---|---|---|---|---|---|---|---|---|
| 1 | 536414 | 22139 | null | 56 | 2010-12-01 11:52:00 UTC | 0.0 | null | United Kingdom |
| 2 | 536545 | 21134 | null | 1 | 2010-12-01 14:32:00 UTC | 0.0 | null | United Kingdom |
| 3 | 536546 | 22145 | null | 1 | 2010-12-01 14:33:00 UTC | 0.0 | null | United Kingdom |
| 4 | 536547 | 37509 | null | 1 | 2010-12-01 14:33:00 UTC | 0.0 | null | United Kingdom |
| 5 | 536549 | 85226A | null | 1 | 2010-12-01 14:34:00 UTC | 0.0 | null | United Kingdom |
| 6 | 536550 | 85044 | null | 1 | 2010-12-01 14:34:00 UTC | 0.0 | null | United Kingdom |
| 7 | 536552 | 20950 | null | 1 | 2010-12-01 14:34:00 UTC | 0.0 | null | United Kingdom |
| 8 | 536553 | 37461 | null | 3 | 2010-12-01 14:35:00 UTC | 0.0 | null | United Kingdom |
| 9 | 536554 | 84670 | null | 23 | 2010-12-01 14:35:00 UTC | 0.0 | null | United Kingdom |
| 10 | 536589 | 21777 | null | -10 | 2010-12-01 16:50:00 UTC | 0.0 | null | United Kingdom |

- **전체 데이터는 몇 행으로 구성되어 있는지 확인하기**

```
SELECT count(*) as row_count -- 전체행수를 row_count로표기
  FROM `modulabs_project.data`;
```

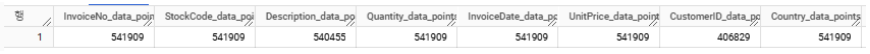


| 행 | row_count |
|---|---|
| 1 | 541909 |

## 데이터 수 세기

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

```
SELECT -- 컬럼의 이름을 알경우
  COUNT(InvoiceNo) AS InvoiceNo_data_points,
  COUNT(StockCode) AS StockCode_data_points,
  COUNT(Description) AS Description_data_points,
  COUNT(Quantity) AS Quantity_data_points,
  COUNT(InvoiceDate) AS InvoiceDate_data_points,
  COUNT(UnitPrice) AS UnitPrice_data_points,
  COUNT(CustomerID) AS CustomerID_data_points,
  COUNT(Country) AS Country_data_points
FROM `modulabs_project.data`;
```



| 행 | InvoiceNo_data_poin | StockCode_data_poi | Description_data_po | Quantity_data_points | InvoiceDate_data_po | UnitPrice_data_point | CustomerID_data_po | Country_data_points |
|---|---|---|---|---|---|---|---|---|
| 1 | 541909 | 541909 | 540455 | 541909 | 541909 | 541909 | 406829 | 541909 |

# 5-4. 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 column_name, ROUND((total - column_value) / total * 100, 2)
  FROM
  (
    SELECT 'InvoiceNo' AS column_name, COUNT(InvoiceNo) AS column_value, COUNT(*)
    AS total FROM modulabs_project.data UNION ALL
    SELECT 'StockCode' AS column_name, COUNT(StockCode) AS column_value, COUNT(*)
    AS total FROM modulabs_project.data UNION ALL
    SELECT 'Description' AS column_name, COUNT(Description) AS column_value, COUNT(*)
    AS total FROM modulabs_project.data UNION ALL
    SELECT 'Quantity' AS column_name, COUNT(Quantity) AS column_value, COUNT(*)
    AS total FROM modulabs_project.data UNION ALL
    SELECT 'InvoiceDate' AS column_name, COUNT(InvoiceDate) AS column_value, COUNT(*)
    AS total FROM modulabs_project.data UNION ALL
    SELECT 'UnitPrice' AS column_name, COUNT(UnitPrice) AS column_value, COUNT(*)
    AS total FROM modulabs_project.data UNION ALL
    SELECT 'CustomerID' AS column_name, COUNT(CustomerID) AS column_value, COUNT(*)
    AS total FROM modulabs_project.data UNION ALL
    SELECT 'Country' AS column_name, COUNT(Country) AS column_value, COUNT(*)
    AS total FROM modulabs_project.data
  ) AS column_data;
```

| 행 | column_name | f0_ |
|---|---|---|
| 1 | CustomerID | 24.93 |
| 2 | UnitPrice | 0.0 |
| 3 | Country | 0.0 |
| 4 | InvoiceDate | 0.0 |
| 5 | Description | 0.27 |
| 6 | StockCode | 0.0 |
| 7 | Quantity | 0.0 |
| 8 | InvoiceNo | 0.0 |

### 결측치 처리 전략

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

```
SELECT distinct Description
    FROM modulabs_project.data
    WHERE StockCode = `85123A`;
```

| 행 | Description |
|---|---|
| 1 | ? |
| 2 | wrongly marked carton 22804 |
| 3 | CREAM HANGING HEART T-LIGHT HOLDER |
| 4 | WHITE HANGING HEART T-LIGHT HOLDER |

### 결측치 처리

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

```
DELETE FROM modulabs_project.data
    WHERE CustomerID IS NULL OR Description IS NULL;
```

작업 정보 | 결과 | 실행 세부정보 | 실행 그래

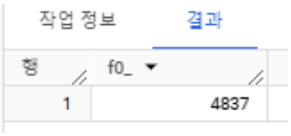
이 문으로 data의 행 135,080개가 삭제되었습니다.

## 5-5. 데이터 전처리(2): 중복값 처리

### 중복값 확인

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

```
SELECT count(*)
FROM (SELECT InvoiceNo, StockCode, Description, Quantity, InvoiceDate, UnitPrice,
            CustomerID, Country, COUNT(*) AS cnt
    FROM modulabs_project.data
    GROUP BY InvoiceNo, StockCode, Description, Quantity, InvoiceDate, UnitPrice,
           CustomerID, Country
    HAVING cnt > 1
    ) AS DuplicateCounts;
```

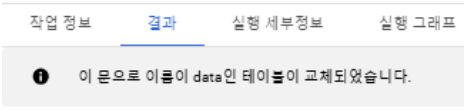



### 중복값 처리

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

```
CREATE OR REPLACE TABLE modulabs_project.data AS
SELECT DISTINCT *
    FROM modulabs_project.data;
```

[결과 이미지를 넣어주세요]



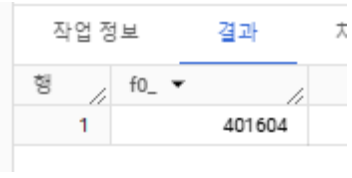

## 5-6. 데이터 전처리(3): 오류값 처리

### `InvoiceNo` 살펴보기

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

```
select count(*)
    from modulabs_project.data;
SELECT COUNT(DISTINCT InvoiceNo) AS UniqueInvoiceCount
    FROM modulabs_project.data;
```

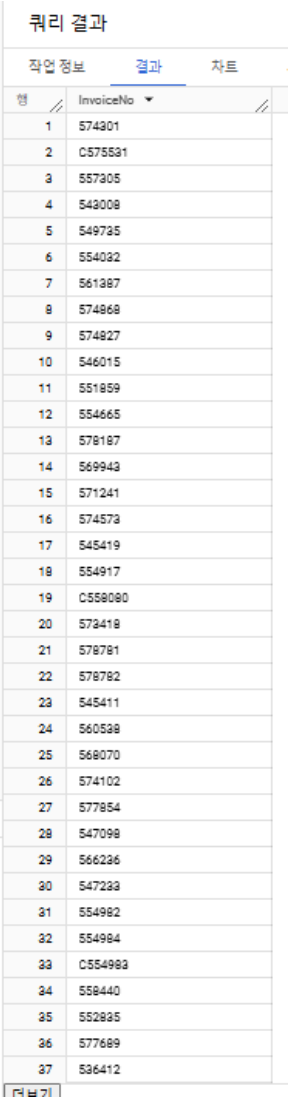
[처음 실행에서 22190 실행을 두번 했더니 결과가 이렇게 됬습니다.]




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

```
SELECT DISTINCT InvoiceNo
    FROM modulabs_project.data
    LIMIT 100;
```

[결과 이미지를 넣어주세요]




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

```
SELECT *
FROM modulabs_project.data
WHERE InvoiceNo LIKE 'C%'
LIMIT 100;
```

[결과 이미지를 넣어주세요]

| 행 | InvoiceNo | StockCode | Description | Quantity | InvoiceDate | UnitPrice | CustomerID | Country |
|---|---|---|---|---|---|---|---|---|
| 1 | C575531 | 22960 | JAM MAKING SET WITH JARS | -4 | 2011-11-10 11:12:00 UTC | 4.25 | 12544 | Spain |
| 2 | C558080 | 22847 | BREAD BIN DINER STYLE IVORY | -1 | 2011-06-26 11:35:00 UTC | 16.95 | 15104 | United Kingdom |
| 3 | C558080 | 22840 | ROUND CAKE TIN VINTAGE RED | -1 | 2011-06-26 11:35:00 UTC | 7.95 | 15104 | United Kingdom |
| 4 | C554983 | 47590B | PINK HAPPY BIRTHDAY BUNTI... | -20 | 2011-05-29 12:18:00 UTC | 4.65 | 17152 | United Kingdom |
| 5 | C554983 | 47590A | BLUE HAPPY BIRTHDAY BUNTI... | -20 | 2011-05-29 12:18:00 UTC | 4.65 | 17152 | United Kingdom |
| 6 | C539709 | 84978 | HANGING HEART JAR T-LIGHT ... | -1 | 2010-12-21 12:33:00 UTC | 1.25 | 18176 | United Kingdom |
| 7 | C539709 | 21485 | RETROSPOT HEART HOT WAT... | -1 | 2010-12-21 12:33:00 UTC | 4.95 | 18176 | United Kingdom |
| 8 | C539709 | 22832 | BROCANTE SHELF WITH HOOKS | -2 | 2010-12-21 12:33:00 UTC | 10.75 | 18176 | United Kingdom |
| 9 | C543620 | 21217 | RED RETROSPOT ROUND CAK... | -1 | 2011-02-10 14:52:00 UTC | 9.95 | 14081 | United Kingdom |
| 10 | C546858 | 22839 | 3 TIER CAKE TIN GREEN AND ... | -1 | 2011-03-17 14:24:00 UTC | 14.95 | 14081 | United Kingdom |
| 11 | C546858 | 21534 | DAIRY MAID LARGE MILK JUG | -1 | 2011-03-17 14:24:00 UTC | 4.95 | 14081 | United Kingdom |
| 12 | C542263 | 22699 | ROSES REGENCY TEACUP AN... | -1 | 2011-01-26 17:16:00 UTC | 2.95 | 14849 | United Kingdom |
| 13 | C553534 | 21467 | CHERRY CROCHET FOOD COV... | -1 | 2011-05-17 15:15:00 UTC | 3.75 | 14849 | United Kingdom |
| 14 | C570996 | 22909 | SET OF 20 VINTAGE CHRISTM... | -12 | 2011-10-13 12:02:00 UTC | 0.85 | 14849 | United Kingdom |
| 15 | C570996 | 23376 | PACK OF 12 VINTAGE CHRIST... | -24 | 2011-10-13 12:02:00 UTC | 0.39 | 14849 | United Kingdom |
| 16 | C543818 | 21038 | SET/4 MODERN VINTAGE COT... | -2 | 2011-02-13 15:45:00 UTC | 2.95 | 16897 | United Kingdom |
| 17 | C543818 | 22622 | BOX OF VINTAGE ALPHABET B... | -2 | 2011-02-13 15:45:00 UTC | 9.95 | 16897 | United Kingdom |
| 18 | C543818 | 21876 | POTTERING MUG | -3 | 2011-02-13 15:45:00 UTC | 1.25 | 16897 | United Kingdom |
| 19 | C543818 | 22138 | BAKING SET 9 PIECE RETROSP... | -2 | 2011-02-13 15:45:00 UTC | 4.95 | 16897 | United Kingdom |
| 20 | C543818 | 22423 | REGENCY CAKESTAND 3 TIER | -2 | 2011-02-13 15:45:00 UTC | 12.75 | 16897 | United Kingdom |
| 21 | C543818 | 85071A | BLUE CHARLIE+LOLA PERSON... | -2 | 2011-02-13 15:45:00 UTC | 2.95 | 16897 | United Kingdom |
| 22 | C543818 | 85071B | RED CHARLIE+LOLA PERSONA... | -2 | 2011-02-13 15:45:00 UTC | 2.95 | 16897 | United Kingdom |
| 23 | C550825 | 20934 | SET/3 POT PLANT CANDLES | -5 | 2011-04-21 10:00:00 UTC | 5.45 | 18177 | United Kingdom |
| 24 | C569422 | 20979 | 36 PENCILS TUBE RED RETRO... | -16 | 2011-10-04 10:38:00 UTC | 1.25 | 12546 | Spain |
| 25 | C556117 | 47503A | ASS FLORAL PRINT MULTI SCR... | -12 | 2011-06-09 09:44:00 UTC | 1.25 | 12802 | Netherlands |
| 26 | C571893 | 21217 | RED RETROSPOT ROUND CAK... | -1 | 2011-10-19 14:13:00 UTC | 9.95 | 13314 | United Kingdom |

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

```
SELECT ROUND(SUM(CASE WHEN InvoiceNo LIKE 'C%' THEN 1 ELSE 0 END) * 100.0 /
    COUNT(*),1) AS CanceledRatio
FROM modulabs_project.data;
```

[결과 이미지를 넣어주세요]

| 행 | CanceledRatio |
|---|---|
| 1 | 2.2 |

## `StockCode` 살펴보기

- **고유한 `StockCode`의 개수를 출력하기**

```
SELECT COUNT(DISTINCT StockCode) AS UniqueStockCodeCount
    FROM modulabs_project.data;
```

[결과 이미지를 넣어주세요]

| 행 | UniqueStockCodeCount |
|---|---|
| 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 modulabs_project.data
GROUP BY StockCode
ORDER BY sell_cnt DESC
LIMIT 10;
```

[결과 이미지를 넣어주세요]

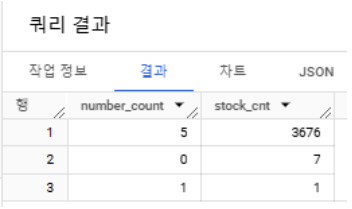
| 행 | StockCode | sell_cnt |
|---|---|---|
| 1 | 85123A | 2065 |
| 2 | 22423 | 1894 |
| 3 | 85099B | 1659 |
| 4 | 47566 | 1409 |
| 5 | 84879 | 1405 |
| 6 | 20725 | 1346 |
| 7 | 22720 | 1224 |
| 8 | POST | 1196 |
| 9 | 22197 | 1110 |
| 10 | 23203 | 1108 |

- **`StockCode`의 문자열 내 숫자의 길이를 구해보기**

```
WITH UniqueStockCodes AS (
  SELECT DISTINCT StockCode
  FROM project_name.modulabs_project.data
)

SELECT
  LENGTH(StockCode) - LENGTH(REGEXP_REPLACE(StockCode, r'[0-9]', ''))
  AS number_count,
  COUNT(*) AS stock_cnt
FROM UniqueStockCodes
GROUP BY number_count
ORDER BY stock_cnt DESC;
```

[결과 이미지를 넣어주세요]




| 행 | number_count | stock_cnt |
|---|---|---|
| 1 | 5 | 3676 |
| 2 | 0 | 7 |
| 3 | 1 | 1 |

- **`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 project_name.modulabs_project.data
)
WHERE number_count <= 1;
```

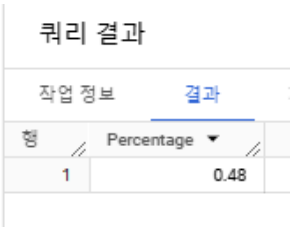
[결과 이미지를 넣어주세요]

| 행 | 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)

```
SELECT
  ROUND(SUM(CASE WHEN StockCode IN ('POST', 'D', 'C2', 'M', 'BANK CHARGES',
  'PADS', 'DOT', 'CRUK')
  THEN 1 ELSE 0 END) * 100.0 / COUNT(*), 2) AS Percentage
FROM modulabs_project.data;
```

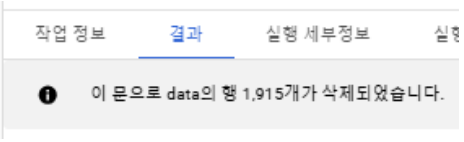
[결과 이미지를 넣어주세요]




- **제품과 관련되지 않은 거래 기록을 제거하기**

```
DELETE FROM modulabs_project.data
WHERE StockCode IN (
  SELECT DISTINCT StockCode
  FROM modulabs_project.data
  WHERE StockCode IN ('BANK CHARGES', 'POST', 'D', 'C2', 'M', 'PADS', 'DOT',
  'CRUK')
);
```

[결과 이미지를 넣어주세요]




## `Description` 살펴보기

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

```
SELECT Description, COUNT(*) AS description_cnt
FROM modulabs_project.data
GROUP BY Description
```

```
ORDER BY description_cnt DESC
LIMIT 30;
```

[결과 이미지를 넣어주세요]

| 행 | Description | description_cnt |
|---|---|---|
| 1 | WHITE HANGING HEART T-LIGHT HOLDER | 2058 |
| 2 | REGENCY CAKESTAND 3 TIER | 1894 |
| 3 | JUMBO BAG RED RETROSPOT | 1659 |
| 4 | PARTY BUNTING | 1409 |
| 5 | ASSORTED COLOUR BIRD ORNAMENT | 1405 |
| 6 | LUNCH BAG RED RETROSPOT | 1345 |
| 7 | SET OF 3 CAKE TINS PANTRY DESIGN | 1224 |
| 8 | LUNCH BAG  BLACK SKULL. | 1099 |
| 9 | PACK OF 72 RETROSPOT CAKE CASES | 1062 |
| 10 | SPOTTY BUNTING | 1026 |
| 11 | PAPER CHAIN KIT 50'S CHRISTMAS | 1013 |
| 12 | LUNCH BAG SPACEBOY DESIGN | 1006 |
| 13 | LUNCH BAG CARS BLUE | 1000 |
| 14 | HEART OF WICKER SMALL | 990 |
| 15 | NATURAL SLATE HEART CHALKBOARD | 989 |
| 16 | JAM MAKING SET WITH JARS | 966 |
| 17 | LUNCH BAG PINK POLKADOT | 961 |
| 18 | LUNCH BAG SUKI DESIGN | 932 |
| 19 | ALARM CLOCK BAKELIKE RED | 917 |
| 20 | REX CASH+CARRY JUMBO SHOPPER | 900 |
| 21 | WOODEN PICTURE FRAME WHITE FINISH | 900 |
| 22 | JUMBO BAG PINK POLKADOT | 897 |
| 23 | SET OF 4 PANTRY JELLY MOULDS | 890 |
| 24 | LUNCH BAG APPLE DESIGN | 890 |
| 25 | BAKING SET 9 PIECE RETROSPOT | 885 |
| 26 | RECIPE BOX PANTRY YELLOW DESIGN | 883 |
| 27 | JAM MAKING SET PRINTED | 883 |
| 28 | LUNCH BAG WOODLAND | 850 |
| 29 | ROSES REGENCY TEACUP AND SAUCER | 844 |
| 30 | VICTORIAN GLASS HANGING T-LIGHT | 843 |

- **대소문자가 혼합된 Description이 있는지 확인하기**

```
SELECT DISTINCT Description
FROM modulabs_project.data
WHERE REGEXP_CONTAINS(Description, r'[a-z]');
```

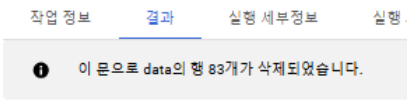
[결과 이미지를 넣어주세요]

| 행 | Description |
|---|---|
| 1 | BAG 125g SWIRLY MARBLES |
| 2 | 3 TRADITIONAl BISCUIT CUTTERS SET |
| 3 | BAG 250g SWIRLY MARBLES |
| 4 | ESSENTIAL BALM 3.5g TIN IN ENVELOPE |
| 5 | FOLK ART GREETING CARD,pack/12 |
| 6 | BAG 500g SWIRLY MARBLES |
| 7 | POLYESTER FILLER PAD 45x45cm |
| 8 | POLYESTER FILLER PAD 40x40cm |
| 9 | Next Day Carriage |
| 10 | FRENCH BLUE METAL DOOR SIGN No |
| 11 | High Resolution Image |
| 12 | POLYESTER FILLER PAD 45x30cm |
| 13 | POLYESTER FILLER PAD 30CMx30CM |
| 14 | NUMBER TILE COTTAGE GARDEN No |
| 15 | NUMBER TILE VINTAGE FONT No |
| 16 | FLOWERS HANDBAG blue and orange |
| 17 | THE KING GIFT BAG 25x24x12cm |
| 18 | POLYESTER FILLER PAD 65CMx65CM |
| 19 | POLYESTER FILLER PAD 60x40cm |

- **서비스 관련 정보를 포함하는 행들을 제거하기**

```
DELETE
FROM modulabs_project.data
WHERE
Description IN ('Next Day Carriage', 'High Resolution Image');
```

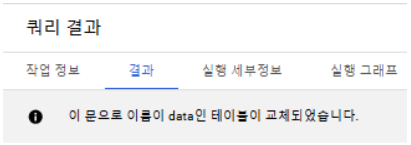
[결과 이미지를 넣어주세요]




- **대소문자를 혼합하고 있는 데이터를 대문자로 표준화 하기**

```
CREATE OR REPLACE TABLE modulabs_project.data AS
SELECT
  * EXCEPT (Description),
   UPPER(Description) AS Description
FROM 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 modulabs_project.data;
```

[결과 이미지를 넣어주세요]

| 행 | min_price | max_price | avg_price |
|---|---|---|---|
| 1 | 0.0 | 649.5 | 2.9049567574060218 |

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

```
SELECT
    COUNT(*) AS cnt_quantity,
    MIN(Quantity) AS min_quantity,
    MAX(Quantity) AS max_quantity,
    AVG(Quantity) AS avg_quantity
FROM modulabs_project.data
WHERE UnitPrice = 0.0;
```

[결과 이미지를 넣어주세요]

| 행 | cnt_quantity | min_quantity | max_quantity | avg_quantity |
|---|---|---|---|---|
| 1 | 33 | 1 | 12540 | 420.5151515151515 |

- `UnitPrice = 0` 를 제거하고 일관된 데이터셋을 유지하기

```
CREATE OR REPLACE TABLE modulabs_project.data AS
SELECT *
FROM modulabs_project.data
WHERE  UnitPrice != 0.0;
```

[결과 이미지를 넣어주세요]

작업 정보 | 결과 | 실행 세부정보 | 실행 그래프

이 문으로 이름이 data인 테이블이 교체되었습니다.

# 5-7. RFM 스코어

## Recency

- `InvoiceDate` 컬럼을 연월일 자료형으로 변경하기

```
SELECT  DATE(InvoiceDate) AS InvoiceDay, *
FROM modulabs_project.data;
```

[결과 이미지를 넣어주세요]

| 행 | InvoiceDay | InvoiceNo | StockCode | Quantity | InvoiceDate | UnitPrice | CustomerID | Country | Description |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 2011-11-03 | 574301 | 20749 | 4 | 2011-11-03 16:15:00 UTC | 7.95 | 12544 | Spain | ASSORTED COLOUR MINI CASES |
| 2 | 2011-11-03 | 574301 | 23514 | 6 | 2011-11-03 16:15:00 UTC | 2.08 | 12544 | Spain | EMBROIDERED RIBBON REEL SALLY |
| 3 | 2011-11-03 | 574301 | 22734 | 6 | 2011-11-03 16:15:00 UTC | 2.89 | 12544 | Spain | SET OF 6 RIBBONS VINTAGE CHRISTMAS |
| 4 | 2011-11-03 | 574301 | 22077 | 12 | 2011-11-03 16:15:00 UTC | 1.95 | 12544 | Spain | 6 RIBBONS RUSTIC CHARM |
| 5 | 2011-11-03 | 574301 | 23511 | 6 | 2011-11-03 16:15:00 UTC | 2.08 | 12544 | Spain | EMBROIDERED RIBBON REEL EMILY |
| 6 | 2011-11-03 | 574301 | 22086 | 6 | 2011-11-03 16:15:00 UTC | 2.95 | 12544 | Spain | PAPER CHAIN KIT 50'S CHRISTMAS |
| 7 | 2011-11-03 | 574301 | 23240 | 6 | 2011-11-03 16:15:00 UTC | 4.15 | 12544 | Spain | SET OF 4 KNICK KNACK TINS DOILY |
| 8 | 2011-11-03 | 574301 | 22621 | 12 | 2011-11-03 16:15:00 UTC | 1.65 | 12544 | Spain | TRADITIONAL KNITTING NANCY |
| 9 | 2011-11-03 | 574301 | 22960 | 6 | 2011-11-03 16:15:00 UTC | 4.25 | 12544 | Spain | JAM MAKING SET WITH JARS |
| 10 | 2011-11-03 | 574301 | 22750 | 4 | 2011-11-03 16:15:00 UTC | 3.75 | 12544 | Spain | FELTCRAFT PRINCESS LOLA DOLL |
| 11 | 2011-11-03 | 574301 | 22144 | 6 | 2011-11-03 16:15:00 UTC | 2.1 | 12544 | Spain | CHRISTMAS CRAFT LITTLE FRIENDS |
| 12 | 2011-11-03 | 574301 | 22910 | 6 | 2011-11-03 16:15:00 UTC | 2.95 | 12544 | Spain | PAPER CHAIN KIT VINTAGE CHRISTMAS |
| 13 | 2011-11-03 | 574301 | 85049A | 12 | 2011-11-03 16:15:00 UTC | 1.25 | 12544 | Spain | TRADITIONAL CHRISTMAS RIBBONS |
| 14 | 2011-11-03 | 574301 | 23512 | 6 | 2011-11-03 16:15:00 UTC | 2.08 | 12544 | Spain | EMBROIDERED RIBBON REEL ROSIE |
| 15 | 2011-11-03 | 574301 | 85049E | 12 | 2011-11-03 16:15:00 UTC | 1.25 | 12544 | Spain | SCANDINAVIAN REDS RIBBONS |
| 16 | 2011-11-03 | 574301 | 20971 | 12 | 2011-11-03 16:15:00 UTC | 1.25 | 12544 | Spain | PINK BLUE FELT CRAFT TRINKET BOX |
| 17 | 2011-11-03 | 574301 | 84879 | 8 | 2011-11-03 16:15:00 UTC | 1.69 | 12544 | Spain | ASSORTED COLOUR BIRD ORNAMENT |
| 18 | 2011-11-03 | 574301 | 22751 | 4 | 2011-11-03 16:15:00 UTC | 3.75 | 12544 | Spain | FELTCRAFT PRINCESS OLIVIA DOLL |
| 19 | 2011-11-10 | C575531 | 22960 | -4 | 2011-11-10 11:12:00 UTC | 4.25 | 12544 | Spain | JAM MAKING SET WITH JARS |
| 20 | 2011-06-19 | 557305 | 23302 | 1 | 2011-06-19 14:42:00 UTC | 1.65 | 13568 | United Kingdom | KNEELING MAT HOUSEWORK DESIGN |
| 21 | 2011-06-19 | 557305 | 22469 | 2 | 2011-06-19 14:42:00 UTC | 1.65 | 13568 | United Kingdom | HEART OF WICKER SMALL |
| 22 | 2011-06-19 | 557305 | 22167 | 1 | 2011-06-19 14:42:00 UTC | 9.95 | 13568 | United Kingdom | OVAL WALL MIRROR DIAMANTE |
| 23 | 2011-06-19 | 557305 | 23067 | 2 | 2011-06-19 14:42:00 UTC | 4.15 | 13568 | United Kingdom | HANGING ENGRAVED METAL HEART |
| 24 | 2011-06-19 | 557305 | 23092 | 1 | 2011-06-19 14:42:00 UTC | 7.9 | 13568 | United Kingdom | LARGE ANTIQUE WHITE PHOTO FRAME |
| 25 | 2011-06-19 | 557305 | 71459 | 2 | 2011-06-19 14:42:00 UTC | 0.85 | 13568 | United Kingdom | HANGING JAM JAR T-LIGHT HOLDER |
| 26 | 2011-06-19 | 557305 | 22561 | 1 | 2011-06-19 14:42:00 UTC | 1.65 | 13568 | United Kingdom | WOODEN SCHOOL COLOURING SET |
| 27 | 2011-06-19 | 557305 | 23298 | 2 | 2011-06-19 14:42:00 UTC | 4.95 | 13568 | United Kingdom | SPOTTY BUNTING |
| 28 | 2011-06-19 | 557305 | 23255 | 1 | 2011-06-19 14:42:00 UTC | 4.15 | 13568 | United Kingdom | CHILDRENS CUTLERY CIRCUS PARADE |
| 29 | 2011-06-19 | 557305 | 47566 | 2 | 2011-06-19 14:42:00 UTC | 4.95 | 13568 | United Kingdom | PARTY BUNTING |
| 30 | 2011-06-19 | 557305 | 20977 | 1 | 2011-06-19 14:42:00 UTC | 1.25 | 13568 | United Kingdom | 36 PENCILS TUBE WOODLAND |

- **가장 최근 구매 일자를 MAX() 함수로 찾아보기**

```
SELECT
    [MAX(DATE(InvoiceDate)) OVER ()] AS most_recent_date,
    [DATE(InvoiceDate)] AS InvoiceDay,
   *
FROM modulabs_project.data;
```

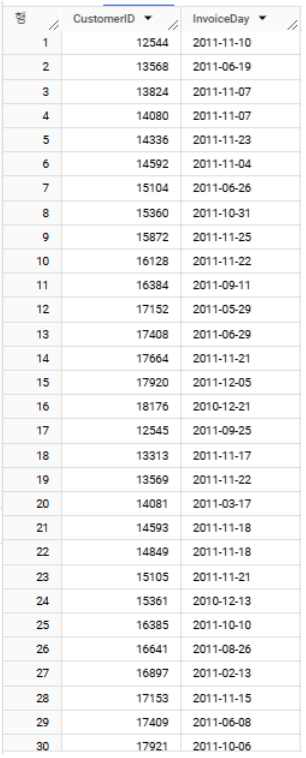
[결과 이미지를 넣어주세요]

| 행 | most_recent_date | InvoiceDay | InvoiceNo | StockCode | Quantity | InvoiceDate | UnitPrice | CustomerID | Country | Description |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2011-12-09 | 2011-09-22 | 567804 | 21034 | 1 | 2011-09-22 12:13:00 UTC | 0.95 | 17920 | United Kingdom | REX CASH+CARRY JUMBO SHOPPER |
| 2 | 2011-12-09 | 2011-01-30 | 542601 | 22839 | 1 | 2011-01-30 11:52:00 UTC | 14.95 | 16897 | United Kingdom | 3 TIER CAKE TIN GREEN AND CREAM |
| 3 | 2011-12-09 | 2010-12-16 | 539082 | 47566 | 5 | 2010-12-16 09:22:00 UTC | 4.65 | 13317 | United Kingdom | PARTY BUNTING |
| 4 | 2011-12-09 | 2011-03-09 | 546121 | 82486 | 1 | 2011-03-09 14:30:00 UTC | 8.95 | 14085 | United Kingdom | WOOD S/3 CABINET ANT WHITE FINISH |
| 5 | 2011-12-09 | 2011-10-25 | 572744 | 21034 | 2 | 2011-10-25 16:14:00 UTC | 0.95 | 14085 | United Kingdom | REX CASH+CARRY JUMBO SHOPPER |
| 6 | 2011-12-09 | 2011-03-02 | 545417 | 22846 | 2 | 2011-03-02 13:42:00 UTC | 16.95 | 15622 | United Kingdom | BREAD BIN DINER STYLE RED |
| 7 | 2011-12-09 | 2010-12-01 | 536595 | 21577 | 1 | 2010-12-01 17:24:00 UTC | 2.25 | 13576 | United Kingdom | SAVE THE PLANET COTTON TOTE BAG |
| 8 | 2011-12-09 | 2011-10-16 | 571281 | 21481 | 28 | 2011-10-16 13:23:00 UTC | 3.39 | 14088 | United Kingdom | FAWN BLUE HOT WATER BOTTLE |
| 9 | 2011-12-09 | 2011-10-16 | 571281 | 23284 | 44 | 2011-10-16 13:23:00 UTC | 7.08 | 14088 | United Kingdom | DOORMAT KEEP CALM AND COME IN |
| 10 | 2011-12-09 | 2011-10-16 | 571281 | 20970 | 20 | 2011-10-16 13:23:00 UTC | 3.39 | 14088 | United Kingdom | PINK FLORAL FELTCRAFT SHOULDER BAG |
| 11 | 2011-12-09 | 2011-11-23 | 578305 | 22660 | 4 | 2011-11-23 15:44:00 UTC | 7.08 | 14088 | United Kingdom | DOORMAT I LOVE LONDON |
| 12 | 2011-12-09 | 2011-11-21 | 577756 | 23108 | 2 | 2011-11-21 14:59:00 UTC | 6.25 | 15115 | United Kingdom | SET OF 10 LED DOLLY LIGHTS |
| 13 | 2011-12-09 | 2011-11-10 | 575718 | 71477 | 24 | 2011-11-10 18:04:00 UTC | 3.29 | 16907 | United Kingdom | COLOUR GLASS. STAR T-LIGHT HOLDER |
| 14 | 2011-12-09 | 2011-10-12 | 570701 | 21917 | 288 | 2011-10-12 09:35:00 UTC | 0.36 | 16652 | United Kingdom | SET 12 KIDS WHITE CHALK STICKS |
| 15 | 2011-12-09 | 2011-10-20 | 572098 | 23486 | 1 | 2011-10-20 15:38:00 UTC | 16.65 | 15117 | United Kingdom | ANTIQUE HEART SHELF UNIT |
| 16 | 2011-12-09 | 2011-09-12 | 566399 | 23318 | 6 | 2011-09-12 13:11:00 UTC | 2.49 | 16141 | United Kingdom | BOX OF 6 MINI VINTAGE CRACKERS |
| 17 | 2011-12-09 | 2011-07-20 | 560718 | 20724 | 100 | 2011-07-20 13:30:00 UTC | 0.72 | 17677 | United Kingdom | RED RETROSPOT CHARLOTTE BAG |
| 18 | 2011-12-09 | 2011-12-08 | 581225 | 23084 | 24 | 2011-12-08 09:55:00 UTC | 1.79 | 17677 | United Kingdom | RABBIT NIGHT LIGHT |
| 19 | 2011-12-09 | 2011-07-28 | C561623 | 23176 | -5 | 2011-07-28 13:43:00 UTC | 2.25 | 17677 | United Kingdom | ABC TREASURE BOOK BOX |
| 20 | 2011-12-09 | 2011-09-23 | 568051 | 23319 | 6 | 2011-09-23 12:40:00 UTC | 2.49 | 18189 | United Kingdom | BOX OF 6 MINI 50'S CRACKERS |
| 21 | 2011-12-09 | 2011-01-04 | 540014 | 21630 | 1 | 2011-01-04 11:34:00 UTC | 8.95 | 14606 | United Kingdom | FLOOR CUSHION ELEPHANT CARNIVAL |
| 22 | 2011-12-09 | 2011-11-30 | 579689 | 21055 | 1 | 2011-11-30 14:00:00 UTC | 8.95 | 14606 | United Kingdom | TOOL BOX SOFT TOY |
| 23 | 2011-12-09 | 2011-09-22 | 567878 | 21843 | 1 | 2011-09-22 14:42:00 UTC | 10.95 | 16910 | United Kingdom | RED RETROSPOT CAKE STAND |

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

```
SELECT
    CustomerID,
     MAX(DATE(InvoiceDate)) AS InvoiceDay
FROM modulabs_project.data
GROUP BY CustomerID;
```

[결과 이미지를 넣어주세요]



| 행 | CustomerID | InvoiceDay |
|---|---|---|
| 1 | 12544 | 2011-11-10 |
| 2 | 13568 | 2011-06-19 |
| 3 | 13824 | 2011-11-07 |
| 4 | 14080 | 2011-11-07 |
| 5 | 14336 | 2011-11-23 |
| 6 | 14592 | 2011-11-04 |
| 7 | 15104 | 2011-06-26 |
| 8 | 15360 | 2011-10-31 |
| 9 | 15872 | 2011-11-25 |
| 10 | 16128 | 2011-11-22 |
| 11 | 16384 | 2011-09-11 |
| 12 | 17152 | 2011-05-29 |
| 13 | 17408 | 2011-06-29 |
| 14 | 17664 | 2011-11-21 |
| 15 | 17920 | 2011-12-05 |
| 16 | 18176 | 2010-12-21 |
| 17 | 12545 | 2011-09-25 |
| 18 | 13313 | 2011-11-17 |
| 19 | 13569 | 2011-11-22 |
| 20 | 14081 | 2011-03-17 |
| 21 | 14593 | 2011-11-18 |
| 22 | 14849 | 2011-11-18 |
| 23 | 15105 | 2011-11-21 |
| 24 | 15361 | 2010-12-13 |
| 25 | 16385 | 2011-10-10 |
| 26 | 16641 | 2011-08-26 |
| 27 | 16897 | 2011-02-13 |
| 28 | 17153 | 2011-11-15 |
| 29 | 17409 | 2011-06-08 |
| 30 | 17921 | 2011-10-06 |

- **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
);
```

[결과 이미지를 넣어주세요]

| 행 | CustomerID | recency |
|---|---|---|
| 1 | 17669 | 42 |
| 2 | 15623 | 141 |
| 3 | 15624 | 116 |
| 4 | 17166 | 38 |
| 5 | 16143 | 3 |
| 6 | 14099 | 16 |
| 7 | 14619 | 239 |
| 8 | 16416 | 26 |
| 9 | 12578 | 21 |
| 10 | 17964 | 31 |
| 11 | 13102 | 1 |
| 12 | 14388 | 8 |
| 13 | 12597 | 19 |
| 14 | 14177 | 84 |
| 15 | 18277 | 58 |
| 16 | 14438 | 306 |
| 17 | 13434 | 74 |
| 18 | 12417 | 3 |
| 19 | 12432 | 42 |
| 20 | 12442 | 3 |
| 21 | 13211 | 10 |
| 22 | 15517 | 233 |
| 23 | 16030 | 309 |
| 24 | 13728 | 36 |
| 25 | 14243 | 8 |
| 26 | 17827 | 5 |

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

```
CREATE OR REPLACE TABLE modulabs_project.user_r AS
SELECT
    CustomerID,
    MAX(DATE(InvoiceDate)) AS InvoiceDay
FROM modulabs_project.data
GROUP BY CustomerID;
```

[결과 이미지를 넣어주세요]

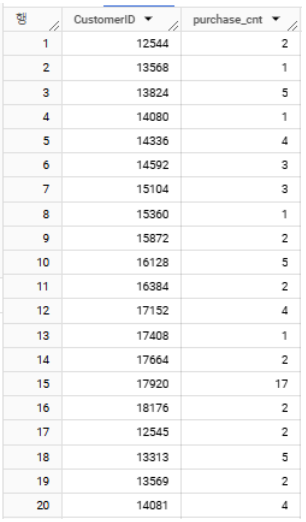
| 행 | CustomerID | InvoiceDay |
|---|---|---|
| 1 | 15381 | 2011-05-10 |
| 2 | 13990 | 2011-05-10 |
| 3 | 16431 | 2011-05-10 |
| 4 | 16306 | 2011-05-10 |
| 5 | 15063 | 2011-05-10 |
| 6 | 17597 | 2011-05-10 |
| 7 | 18133 | 2011-05-11 |
| 8 | 14149 | 2011-05-11 |
| 9 | 14288 | 2011-05-11 |
| 10 | 15732 | 2011-05-11 |
| 11 | 17889 | 2011-05-11 |
| 12 | 14920 | 2011-05-11 |
| 13 | 13235 | 2011-05-11 |
| 14 | 15079 | 2011-05-11 |
| 15 | 13052 | 2011-05-11 |
| 16 | 12976 | 2011-05-12 |
| 17 | 13952 | 2011-05-12 |
| 18 | 17970 | 2011-05-12 |
| 19 | 13837 | 2011-05-12 |
| 20 | 12625 | 2011-05-12 |
| 21 | 14988 | 2011-05-12 |
| 22 | 16244 | 2011-05-12 |
| 23 | 16943 | 2011-05-12 |

## Frequency

- **고객마다 고유한 InvoiceNo의 수를 세어보기**

```
SELECT
  CustomerID,
  COUNT(DISTINCT InvoiceNo) AS purchase_cnt
FROM modulabs_project.data
GROUP BY CustomerID;
```

[결과 이미지를 넣어주세요]



| 행 | CustomerID | purchase_cnt |
|---|---|---|
| 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 |
| 12 | 17152 | 4 |
| 13 | 17408 | 1 |
| 14 | 17664 | 2 |
| 15 | 17920 | 17 |
| 16 | 18176 | 2 |
| 17 | 12545 | 2 |
| 18 | 13313 | 5 |
| 19 | 13569 | 2 |
| 20 | 14081 | 4 |

- **각 고객 별로 구매한 아이템의 총 수량 더하기**

```
SELECT
  CustomerID,
  SUM(Quantity) AS item_cnt
FROM modulabs_project.data
GROUP BY CustomerID;
```

[결과 이미지를 넣어주세요]

| 행 | CustomerID | item_cnt |
|---|---|---|
| 1 | 12544 | 130 |
| 2 | 13568 | 66 |
| 3 | 13824 | 768 |
| 4 | 14080 | 48 |
| 5 | 14336 | 1759 |
| 6 | 14592 | 407 |
| 7 | 15104 | 633 |
| 8 | 15360 | 223 |
| 9 | 15872 | 187 |
| 10 | 16128 | 988 |
| 11 | 16384 | 260 |
| 12 | 17152 | 477 |
| 13 | 17408 | 3 |
| 14 | 17664 | 604 |
| 15 | 17920 | 2471 |
| 16 | 18176 | 279 |
| 17 | 12545 | 517 |
| 18 | 13313 | 851 |
| 19 | 13569 | 155 |
| 20 | 14081 | 589 |
| 21 | 14593 | 330 |
| 22 | 14849 | 5656 |
| 23 | 15105 | 1211 |
| 24 | 15361 | 336 |
| 25 | 16385 | 258 |
| 26 | 16641 | 256 |

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

```
CREATE OR REPLACE TABLE modulabs_project.user_rf AS

-- (1) 전체 거래 건수 계산
WITH purchase_cnt AS (
  SELECT
    CustomerID,
    COUNT(DISTINCT InvoiceNo) AS purchase_cnt
  FROM modulabs_project.data
  GROUP BY CustomerID
),

-- (2) 구매한 아이템 총 수량 계산
item_cnt AS (
  SELECT
    CustomerID,
    SUM(Quantity) AS item_cnt
  FROM modulabs_project.data
  GROUP BY CustomerID
)

-- 기존의 user_r에 (1)과 (2)를 통합
SELECT
  pc.CustomerID AS customer_id, -- 별칭을 사용하여 중복 제거
  pc.purchase_cnt,
  ic.item_cnt,
  ur.*
FROM purchase_cnt AS pc
JOIN item_cnt AS ic
  ON pc.CustomerID = ic.CustomerID
JOIN modulabs_project.user_r AS ur
  ON pc.CustomerID = ur.CustomerID;
```

[결과 이미지를 넣어주세요]

| 행 | customer_id | purchase_cnt | item_cnt | CustomerID | InvoiceDay |
|---|---|---|---|---|---|
| 1 | 15063 | 1 | 56 | 15063 | 2011-05-10 |
| 2 | 13990 | 1 | 190 | 13990 | 2011-05-10 |
| 3 | 17597 | 1 | 1176 | 17597 | 2011-05-10 |
| 4 | 18133 | 1 | 1350 | 18133 | 2011-05-11 |
| 5 | 13235 | 1 | 441 | 13235 | 2011-05-11 |
| 6 | 13052 | 1 | 222 | 13052 | 2011-05-11 |
| 7 | 12976 | 1 | 561 | 12976 | 2011-05-12 |
| 8 | 16006 | 1 | 84 | 16006 | 2011-05-12 |
| 9 | 14988 | 1 | 141 | 14988 | 2011-05-12 |
| 10 | 17970 | 1 | 513 | 17970 | 2011-05-12 |
| 11 | 12770 | 1 | 743 | 12770 | 2011-05-13 |
| 12 | 14873 | 1 | 168 | 14873 | 2011-05-13 |
| 13 | 17263 | 1 | 36 | 17263 | 2011-05-15 |
| 14 | 15333 | 1 | 344 | 15333 | 2011-05-15 |
| 15 | 14689 | 1 | 76 | 14689 | 2011-05-15 |
| 16 | 14888 | 1 | 184 | 14888 | 2011-05-16 |
| 17 | 14489 | 1 | 299 | 14489 | 2011-05-16 |
| 18 | 13976 | 1 | 154 | 13976 | 2011-05-17 |
| 19 | 13572 | 1 | 534 | 13572 | 2011-05-18 |
| 20 | 17556 | 1 | 101 | 17556 | 2011-05-18 |
| 21 | 12690 | 1 | 103 | 12690 | 2011-05-18 |
| 22 | 17245 | 1 | 75 | 17245 | 2011-05-19 |
| 23 | 17871 | 1 | 182 | 17871 | 2011-05-19 |
| 24 | 13887 | 1 | 147 | 13887 | 2011-05-19 |
| 25 | 12353 | 1 | 20 | 12353 | 2011-05-19 |
| 26 | 13391 | 1 | 4 | 13391 | 2011-05-20 |
| 27 | 17455 | 1 | 66 | 17455 | 2011-05-20 |
| 28 | 13243 | 1 | 232 | 13243 | 2011-05-20 |
| 29 | 13227 | 1 | 99 | 13227 | 2011-05-22 |
| 30 | 15292 | 1 | 55 | 15292 | 2011-05-22 |

이 문으로 이름이 user_rf인 새 테이블이 생성되었습니다.

## Monetary

- **고객별 총 지출액 계산 (소수점 첫째 자리에서 반올림)**

```
SELECT
  CustomerID,
    ROUND(SUM(UnitPrice * Quantity), 1) AS user_total
FROM modulabs_project.data
GROUP BY CustomerID;
```

[결과 이미지를 넣어주세요]

| 행 | CustomerID | user_total |
|---|---|---|
| 1 | 12544 | 299.7 |
| 2 | 13568 | 187.0 |
| 3 | 13824 | 1698.9 |
| 4 | 14080 | 45.6 |
| 5 | 14336 | 1614.9 |
| 6 | 14592 | 557.9 |
| 7 | 15104 | 968.6 |
| 8 | 15360 | 427.9 |
| 9 | 15872 | 316.2 |
| 10 | 16128 | 1880.2 |
| 11 | 16384 | 584.5 |
| 12 | 17152 | 1503.5 |
| 13 | 17408 | 32.6 |
| 14 | 17664 | 604.6 |
| 15 | 17920 | 4107.6 |
| 16 | 18176 | 448.6 |
| 17 | 12545 | 832.4 |
| 18 | 13313 | 1555.3 |
| 19 | 13569 | 337.1 |
| 20 | 14081 | 891.8 |
| 21 | 14593 | 617.1 |
| 22 | 14849 | 7904.3 |
| 23 | 15105 | 2298.9 |
| 24 | 15361 | 418.9 |
| 25 | 16385 | 554.9 |
| 26 | 16641 | 231.1 |
| 27 | 16897 | 213.0 |

- **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 modulabs_project.user_rfm AS
SELECT
  rf.CustomerID AS CustomerID,
  rf.purchase_cnt,
  rf.item_cnt,
  -- rf.recency 제거
  ut.user_total,
  ROUND(ut.user_total / NULLIF(rf.purchase_cnt, 0), 1) AS user_average
  -- 평균 거래 금액 계산
FROM modulabs_project.user_rf rf
LEFT JOIN (
  -- 고객 별 총 지출액
  SELECT
    CustomerID,
    SUM(UnitPrice * Quantity) AS user_total -- 총 지출액 계산
  FROM modulabs_project.data
  GROUP BY CustomerID
) ut
ON rf.CustomerID = ut.CustomerID;
```

[결과 이미지를 넣어주세요]

| 행 | CustomerID | purchase_cnt | item_cnt | user_total | user_average |
|---|---|---|---|---|---|
| 1 | 16526 | 1 | 106 | 290.36 | 290.4 |
| 2 | 17939 | 1 | 40 | 99.14 | 99.1 |
| 3 | 15795 | 1 | 581 | 610.01 | 610.0 |
| 4 | 16926 | 1 | 35 | 230.25 | 230.3 |
| 5 | 16658 | 1 | 40 | 123.24 | 123.2 |
| 6 | 17303 | 1 | 144 | 250.039999999... | 250.0 |
| 7 | 13336 | 1 | 559 | 795.12 | 795.1 |
| 8 | 13866 | 1 | 57 | 145.670000000... | 145.7 |
| 9 | 16803 | 1 | 245 | 332.299999999... | 332.3 |
| 10 | 13845 | 1 | 182 | 312.98 | 313.0 |
| 11 | 16776 | 1 | 254 | 371.54 | 371.5 |
| 12 | 14975 | 1 | 380 | 279.94 | 279.9 |
| 13 | 18240 | 1 | 126 | 422.58 | 422.6 |
| 14 | 17684 | 1 | 145 | 239.410000000... | 239.4 |
| 15 | 14968 | 1 | 127 | 147.590000000... | 147.6 |
| 16 | 17901 | 1 | 24 | 110.380000000... | 110.4 |
| 17 | 15733 | 1 | 62 | 162.3 | 162.3 |
| 18 | 15435 | 1 | 54 | 149.290000000... | 149.3 |
| 19 | 15445 | 1 | 58 | 113.5 | 113.5 |
| 20 | 18224 | 1 | 69 | 158.95 | 158.9 |
| 21 | 13841 | 1 | 100 | 85.0 | 85.0 |
| 22 | 16142 | 1 | 422 | 535.339999999... | 535.3 |
| 23 | 13391 | 1 | 4 | 59.8 | 59.8 |

이 문으로 이름이 user_rfm인 새 테이블이 생성되었습니다.

## RFM 통합 테이블 출력하기

- **최종 user_rfm 테이블을 출력하기**

```
SELECT *
    FROM modulabs_project.user_rfm;
```

[결과 이미지를 넣어주세요]


| 행 | CustomerID | purchase_cnt | item_cnt | user_total | user_average |
|---|---|---|---|---|---|
| 1 | 16526 | 1 | 106 | 290.36 | 290.4 |
| 2 | 17939 | 1 | 40 | 99.14 | 99.1 |
| 3 | 15795 | 1 | 581 | 610.01 | 610.0 |
| 4 | 16926 | 1 | 35 | 230.25 | 230.3 |
| 5 | 16658 | 1 | 40 | 123.24 | 123.2 |
| 6 | 17303 | 1 | 144 | 250.039999999... | 250.0 |
| 7 | 13336 | 1 | 559 | 795.12 | 795.1 |
| 8 | 13866 | 1 | 57 | 145.670000000... | 145.7 |
| 9 | 16803 | 1 | 245 | 332.299999999... | 332.3 |
| 10 | 13845 | 1 | 182 | 312.98 | 313.0 |
| 11 | 16776 | 1 | 254 | 371.54 | 371.5 |
| 12 | 14975 | 1 | 380 | 279.94 | 279.9 |
| 13 | 18240 | 1 | 126 | 422.58 | 422.6 |
| 14 | 17684 | 1 | 145 | 239.410000000... | 239.4 |
| 15 | 14968 | 1 | 127 | 147.590000000... | 147.6 |
| 16 | 17901 | 1 | 24 | 110.380000000... | 110.4 |
| 17 | 15733 | 1 | 62 | 162.3 | 162.3 |
| 18 | 15435 | 1 | 54 | 149.290000000... | 149.3 |
| 19 | 15445 | 1 | 58 | 113.5 | 113.5 |
| 20 | 18224 | 1 | 69 | 158.95 | 158.9 |
| 21 | 13841 | 1 | 100 | 85.0 | 85.0 |
| 22 | 16142 | 1 | 422 | 535.339999999... | 535.3 |
| 23 | 13391 | 1 | 4 | 59.8 | 59.8 |

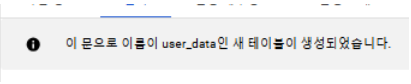

# 5-8. 추가 Feature 추출

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

[결과 이미지를 넣어주세요]



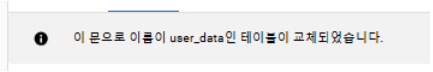

## 2. 평균 구매 주기

- **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 project_name.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_name.modulabs_project.data
    WHERE CustomerID IS NOT NULL
  )
  GROUP BY CustomerID
)

SELECT u.*, pi.* EXCEPT (CustomerID)
FROM project_name.modulabs_project.user_data AS u
LEFT JOIN purchase_intervals AS pi
ON u.CustomerID = pi.CustomerID;
```

[결과 이미지를 넣어주세요]




## 3. 구매 취소 경향성

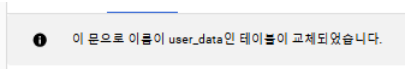
- **고객의 취소 패턴 파악하기**
  **1) 취소 빈도(cancel_frequency) : 고객 별로 취소한 거래의 총 횟수**
  **2) 취소 비율(cancel_rate) : 각 고객이 한 모든 거래 중에서 취소를 한 거래의 비율**
  - **취소 빈도와 취소 비율을 계산하고 그 결과를 `user_data` 에 통합하기**
    **(취소 비율은 소수점 두번째 자리)**

```
CREATE OR REPLACE TABLE modulabs_project.user_data AS

WITH TransactionInfo AS (
  SELECT
    CustomerID,
     COUNT(*) AS total_transactions, -- 전체 거래 건수
    COUNTIF(InvoiceNo LIKE 'C%') AS cancel_frequency -- 취소된 거래 건수
  FROM modulabs_project.data
  GROUP BY CustomerID
)

SELECT u.*, t.* EXCEPT(CustomerID),
ROUND(t.cancel_frequency * 100.0 / NULLIF(t.total_transactions, 0), 2)
AS cancel_rate -- 취소 비율 계산
FROM `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 modulabs_project.user_data ;
```

[결과 이미지를 넣어주세요]

| 행 | CustomerID | purchase_cnt | item_cnt | user_total | user_average | unique_products | average_interval | total_transactions | cancel_frequency | cancel_rate |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 14432 | 6 | 2013 | 2248.4999999999982 | 374.7 | 256 | 0.2 | 377 | 0 | 0.0 |
| 2 | 12428 | 11 | 3477 | 6366.000000000055 | 578.7 | 256 | 0.87 | 292 | 5 | 1.71 |
| 3 | 13268 | 14 | 3525 | 3105.6599999999989 | 221.8 | 256 | 0.56 | 439 | 7 | 1.59 |
| 4 | 16344 | 1 | 18 | 101.10000000000001 | 101.1 | 1 | 0.0 | 2 | 0 | 0.0 |
| 5 | 15753 | 1 | 144 | 79.2 | 79.2 | 1 | 0.0 | 1 | 0 | 0.0 |
| 6 | 17923 | 1 | 50 | 207.50000000000003 | 207.5 | 1 | 0.0 | 1 | 0 | 0.0 |
| 7 | 14679 | 1 | -1 | -2.55 | -2.5 | 1 | 0.0 | 1 | 1 | 100.0 |
| 8 | 16953 | 1 | 10 | 20.8 | 20.8 | 1 | 0.0 | 1 | 0 | 0.0 |
| 9 | 16579 | 1 | -12 | -30.599999999999998 | -30.6 | 1 | 0.0 | 1 | 1 | 100.0 |
| 10 | 14119 | 1 | -2 | -19.9 | -19.9 | 1 | 0.0 | 1 | 1 | 100.0 |
| 11 | 17986 | 1 | 10 | 20.8 | 20.8 | 1 | 0.0 | 1 | 0 | 0.0 |
| 12 | 13366 | 1 | 144 | 56.160000000000004 | 56.2 | 1 | 0.0 | 1 | 0 | 0.0 |
| 13 | 16881 | 1 | 600 | 432.0 | 432.0 | 1 | 0.0 | 1 | 0 | 0.0 |
| 14 | 16144 | 1 | 16 | 175.2 | 175.2 | 1 | 0.0 | 1 | 0 | 0.0 |
| 15 | 13017 | 1 | 48 | 204.0 | 204.0 | 1 | 0.0 | 1 | 0 | 0.0 |
| 16 | 18113 | 1 | 72 | 76.32000000000007 | 76.3 | 1 | 0.0 | 1 | 0 | 0.0 |
| 17 | 14351 | 1 | 12 | 51.0 | 51.0 | 1 | 0.0 | 1 | 0 | 0.0 |
| 18 | 16995 | 1 | -1 | -1.25 | -1.3 | 1 | 0.0 | 1 | 1 | 100.0 |
| 19 | 16061 | 1 | -1 | -29.95 | -29.9 | 1 | 0.0 | 1 | 1 | 100.0 |
| 20 | 13747 | 1 | 8 | 79.6 | 79.6 | 1 | 0.0 | 1 | 0 | 0.0 |
| 21 | 13829 | 1 | -12 | -102.0 | -102.0 | 1 | 0.0 | 1 | 1 | 100.0 |
| 22 | 15313 | 1 | 25 | 52.0 | 52.0 | 1 | 0.0 | 1 | 0 | 0.0 |
| 23 | 15389 | 1 | 400 | 500.0 | 500.0 | 1 | 0.0 | 1 | 0 | 0.0 |
| 24 | 14090 | 1 | 72 | 76.32000000000007 | 76.3 | 1 | 0.0 | 1 | 0 | 0.0 |
| 25 | 17948 | 1 | 144 | 358.56000000000006 | 358.6 | 1 | 0.0 | 1 | 0 | 0.0 |
| 26 | 17443 | 1 | 504 | 534.24 | 534.2 | 1 | 0.0 | 1 | 0 | 0.0 |
| 27 | 15488 | 1 | 72 | 76.32000000000007 | 76.3 | 1 | 0.0 | 1 | 0 | 0.0 |