김진형

11-2. 데이터 불러오기

데이터 살펴보기

- **테이블에 있는 10개의 행만 출력하기**

```
SELECT *
FROM avid-involution-439402-i8.modulabs_project.data
LIMIT 10;
```

쿼리 결과

결과 저장 데이터 탐

작업 정보 결과 차트 JSON 실행 세부정보 실행 그래프

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(*)
FROM avid-involution-439402-i8.modulabs_project.data
```

쿼리 결과

작업 정보 결과 차트 JSON 실행 세부정보 실행 그래프

행	f0_
1	541909

데이터 수 세기

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

```
 SELECT
    COUNT(InvoiceNo) AS column1,
    COUNT(StockCode) AS column2,
    COUNT(Description) AS column3,
    COUNT(Quantity) AS column4,
    COUNT(InvoiceDate) AS column5,
    COUNT(UnitPrice) AS column6,
    COUNT(CustomerID) AS column7,
    COUNT(Country) AS column8

FROM avid-involution-439402-i8.modulabs_project.data
```

작업 정보 결과 차트 JSON 실행 세부정보 실행 그래프

행	column1	column2	column3	column4	column5	column6	column7	column8
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(SUM(CASE WHEN InvoiceNo IS NULL THEN 1 ELSE 0 END) / COUNT(*) * 100, 2) AS missing_percent
FROM
```

[결과 이미지를 넣어주세요]

결측치 처리 전략

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

```
SELECT Description
FROM avid-involution-439402-i8.modulabs_project.data
WHERE StockCode = '85123A';
```

작업 정보 | 결과 | 차트 | JSON | 실행 세부정보

행	Description
1	?
2	wrongly marked carton 22804
3	CREAM HANGING HEART T-LIG...
4	CREAM HANGING HEART T-LIG...
5	CREAM HANGING HEART T-LIG...
6	CREAM HANGING HEART T-LIG...
7	CREAM HANGING HEART T-LIG...
8	CREAM HANGING HEART T-LIG...
9	CREAM HANGING HEART T-LIG...
10	CREAM HANGING HEART T-LIG...
11	CREAM HANGING HEART T-LIG...
12	WHITE HANGING HEART T-LIG...
13	WHITE HANGING HEART T-LIG...

결측치 처리

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

```
DELETE FROM avid-involution-439402-i8.modulabs_project.data
WHERE Description IS NULL
OR CustomerID IS NULL
```

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 duplicates
FROM (
    SELECT COUNT(*) AS Count
    FROM  avid-involution-439402-i8.modulabs_project.data
    GROUP BY InvoiceNo, StockCode, Description, Quantity, InvoiceDate, UnitPrice, CustomerID, Countr
HAVING COUNT(*) > 1
)
```

작업 정보 | 결과 | 차트 | JSON

행	duplicates
1	4837

중복값 처리

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

```
CREATE OR REPLACE TABLE avid-involution-439402-i8.modulabs_project.data
AS SELECT DISTINCT * FROM avid-involution-439402-i8.modulabs_project.data
```

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(DISTINCT InvoiceNo) AS CountInvoice
FROM avid-involution-439402-i8.modulabs_project.data
```

쿼리 결과

작업 정보 | 결과 | 차트 | JSON | 실행 세부정보

행	CountInvoice
1	22190

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

```
SELECT DISTINCT InvoiceNo
FROM avid-involution-439402-i8.modulabs_project.data
LIMIT 100;
```

작업 정보 결과

행	InvoiceNo
1	574301
2	C575531
3	557305
4	543008
5	549735
6	554032
7	561387
8	574868
9	574827
10	546015
11	551859
12	554665
13	578187
14	569943
15	571241
16	574573
17	545419

더보기

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

```
SELECT *
FROM project_name.modulabs_project.data
WHERE # [[YOUR QUERY]]
LIMIT 100;
```

작업 정보 결과 차트 JSON 실행 세부정보 실행 그래프

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	22840	ROUND CAKE TIN VINTAGE RED	-1	2011-06-26 11:35:00 UTC	7.95	15104	United Kingdom
3	C558080	22847	BREAD BIN DINER STYLE IVORY	-1	2011-06-26 11:35:00 UTC	16.95	15104	United Kingdom
4	C554983	47590A	BLUE HAPPY BIRTHDAY BUNTI...	-20	2011-05-29 12:18:00 UTC	4.65	17152	United Kingdom
5	C554983	47590B	PINK HAPPY BIRTHDAY BUNTI...	-20	2011-05-29 12:18:00 UTC	4.65	17152	United Kingdom
6	C539709	21485	RETROSPOT HEART HOT WAT...	-1	2010-12-21 12:33:00 UTC	4.95	18176	United Kingdom
7	C539709	84978	HANGING HEART JAR T-LIGHT ...	-1	2010-12-21 12:33:00 UTC	1.25	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	21534	DAIRY MAID LARGE MILK JUG	-1	2011-03-17 14:24:00 UTC	4.95	14081	United Kingdom
11	C546858	22839	3 TIER CAKE TIN GREEN AND ...	-1	2011-03-17 14:24:00 UTC	14.95	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

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

```
SELECT SELECT
    ROUND((COUNT(CASE WHEN InvoiceNo LIKE 'C%' THEN 1 END) / COUNT(*)) * 100, 1) AS rate
FROM
    avid-involution-439402-i8.modulabs_project.data;
```

작업 정보 결과 차트 JSON

행	rate
1	2.2

`StockCode` 살펴보기

- 고유한 `StockCode` 의 개수를 출력하기

```
SELECT
    COUNT(DISTINCT StockCode) AS uniq_stockcode_ct
```

```
FROM
    avid-involution-439402-i8.modulabs_project.data;
```

작업 정보 | 결과 | 차트 | JSON

행	unique_stockcode_c
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
    avid-involution-439402-i8.modulabs_project.data
GROUP BY
    StockCode
ORDER BY
    sell_cnt DESC
LIMIT 10;
```

작업 정보 | 결과 | 차트 | 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 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 avid-involution-439402-i8.modulabs_project.data
)
WHERE number_count <= 1;
```

[결과 이미지를 넣어주세요]

- **`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 DISTINCT StockCode, number_count
FROM (
  SELECT StockCode,
```

```
    LENGTH(StockCode) - LENGTH(REGEXP_REPLACE(StockCode, r'[0-9]', '')) AS number_count
  FROM avid-involution-439402-i8.modulabs_project.data
)
WHERE # [[YOUR QUERY]];
```

[결과 이미지를 넣어주세요]

- **제품과 관련되지 않은 거래 기록을 제거하기**

```
DELETE FROM avid-involution-439402-i8.modulabs_project.data
 WHERE StockCode IN (
SELECT DISTINCT StockCode
FROM (
  SELECT StockCode
  FROM avid-involution-439402-i8.modulabs_project.data
  WHERE StockCode IN ('POST', 'D','C2', 'M', 'BANK CHARGES', 'PADS', 'DOT', 'CRUK')
) AS non_product_codes
 );
```

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_name.modulabs_project.data
FROM avid-involution-439402-i8.modulabs_project.data
GROUP BY Description
ORDER BY description_cnt DESC
LIMIT 30;
```

작업 정보 | 결과 | 차트 | JSON | 실행 세부정보

행	Description	description_cnt
18	LUNCH BAG SUKI DESIGN	932
19	ALARM CLOCK BAKELIKE RED	917
20	WOODEN PICTURE FRAME WH...	900
21	REX CASH+CARRY JUMBO SH...	900
22	JUMBO BAG PINK POLKADOT	897
23	LUNCH BAG APPLE DESIGN	890
24	SET OF 4 PANTRY JELLY MOU...	890
25	BAKING SET 9 PIECE RETROSP...	885
26	RECIPE BOX PANTRY YELLOW ...	883
27	JAM MAKING SET PRINTED	883
28	LUNCH BAG WOODLAND	850
29	ROSES REGENCY TEACUP AN...	844
30	VICTORIAN GLASS HANGING T...	843

- **서비스 관련 정보를 포함하는 행들을 제거하기**

```
DELETE
FROM avid-involution-439402-i8.modulabs_project.data
WHERE
  Description LIKE 'Next Day Carriage' OR
  Description LIKE 'High Resolution Image'
```

작업 정보 | 결과 | 실행 세부정보 | 실행 그래프

이 문으로 data의 행 0개가 삭제되었습니다.

순서대로 진행하지 않아서 삭제 결과가 이렇게 나왔습니다.

- **대소문자를 혼합하고 있는 데이터를 대문자로 표준화 하기**

```
CREATE OR REPLACE TABLE avid-involution-439402-i8.modulabs_project.data AS
SELECT
  * EXCEPT (Description),
  UPPER(Description) AS Description
FROM avid-involution-439402-i8.modulabs_project.data
```

작업 정보 | 결과 | 실행 세부정보 | 실행 그래프

이 문으로 이름이 data인 테이블이 교체되었습니다.

`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 avid-involution-439402-i8.modulabs_project.data;
```

작업 정보 | 결과 | 차트 | JSON | 실행 세부정보

행	min_price	max_price	avg_price
1	0.0	649.5	2.907457172951...

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

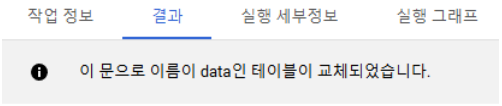
```
SELECT
  COUNT(*) AS cnt_quantity,
  MIN(quantity) AS min_quantity,
  MAX(quantity) AS max_quantity,
  AVG(quantity) AS avg_quantity
FROM avid-involution-439402-i8.modulabs_project.data
WHERE UnitPrice = 0;
```

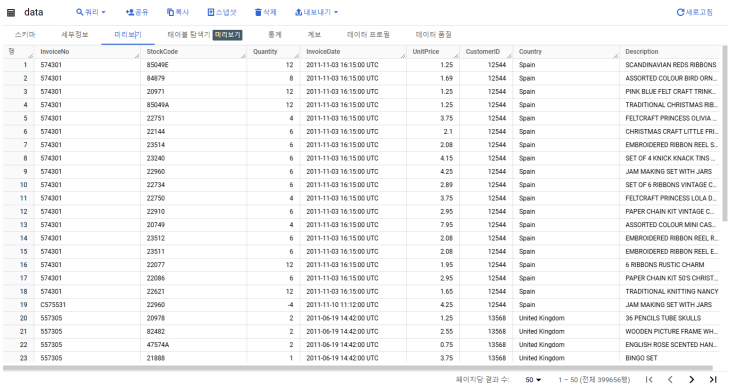
작업 정보 | 결과 | 차트 | JSON | 실행 세부정보 | 실행 그래프

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 avid-involution-439402-i8.modulabs_project.data AS
SELECT *
FROM avid-involution-439402-i8.modulabs_project.data
WHERE UnitPrice != 0;
```





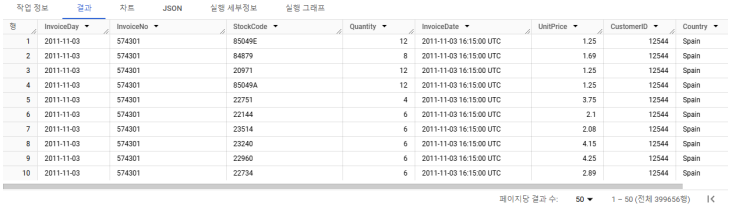
행	InvoiceNo	StockCode	Quantity	InvoiceDate	UnitPrice	CustomerID	Country	Description
1	574301	85049E	12	2011-11-03 16:15:00 UTC	1.25	12544	Spain	SCANDINAVIAN REDS RIBBONS
2	574301	84879	8	2011-11-03 16:15:00 UTC	1.69	12544	Spain	ASSORTED COLOUR BIRD ORN...
3	574301	20971	12	2011-11-03 16:15:00 UTC	1.25	12544	Spain	PINK BLUE FELT CRAFT TRINK...
4	574301	85049A	12	2011-11-03 16:15:00 UTC	1.25	12544	Spain	TRADITIONAL CHRISTMAS RIB...
5	574301	22751	4	2011-11-03 16:15:00 UTC	3.75	12544	Spain	FELTCRAFT PRINCESS OLIVIA ...
6	574301	22144	6	2011-11-03 16:15:00 UTC	2.1	12544	Spain	CHRISTMAS CRAFT LITTLE FRI...
7	574301	23514	6	2011-11-03 16:15:00 UTC	2.08	12544	Spain	EMBROIDERED RIBBON REEL S...
8	574301	23240	6	2011-11-03 16:15:00 UTC	4.15	12544	Spain	SET OF 4 KNICK KNACK TINS ...
9	574301	22960	6	2011-11-03 16:15:00 UTC	4.25	12544	Spain	JAM MAKING SET WITH JARS
10	574301	22734	6	2011-11-03 16:15:00 UTC	2.89	12544	Spain	SET OF 6 RIBBONS VINTAGE C...
11	574301	22750	4	2011-11-03 16:15:00 UTC	3.75	12544	Spain	FELTCRAFT PRINCESS LOLA D...
12	574301	22910	6	2011-11-03 16:15:00 UTC	2.95	12544	Spain	PAPER CHAIN KIT VINTAGE C...
13	574301	20749	4	2011-11-03 16:15:00 UTC	7.95	12544	Spain	ASSORTED COLOUR MINI CAS...
14	574301	23512	6	2011-11-03 16:15:00 UTC	2.08	12544	Spain	EMBROIDERED RIBBON REEL R...
15	574301	23511	6	2011-11-03 16:15:00 UTC	2.08	12544	Spain	EMBROIDERED RIBBON REEL E...
16	574301	22077	12	2011-11-03 16:15:00 UTC	1.95	12544	Spain	6 RIBBONS RUSTIC CHARM
17	574301	22086	6	2011-11-03 16:15:00 UTC	2.95	12544	Spain	PAPER CHAIN KIT 50'S CHRIST...
18	574301	22621	12	2011-11-03 16:15:00 UTC	1.65	12544	Spain	TRADITIONAL KNITTING NANCY
19	C575531	22960	-4	2011-11-10 11:12:00 UTC	4.25	12544	Spain	JAM MAKING SET WITH JARS
20	557305	20978	2	2011-06-19 14:42:00 UTC	1.25	13568	United Kingdom	36 PENCILS TUBE SKULLS
21	557305	82482	2	2011-06-19 14:42:00 UTC	2.55	13568	United Kingdom	WOODEN PICTURE FRAME WH...
22	557305	47574A	2	2011-06-19 14:42:00 UTC	0.75	13568	United Kingdom	ENGLISH ROSE SCENTED HAN...
23	557305	21888	1	2011-06-19 14:42:00 UTC	3.75	13568	United Kingdom	BINGO SET

11-7. RFM 스코어

Recency

- `InvoiceDate` 컬럼을 연월일 자료형으로 변경하기

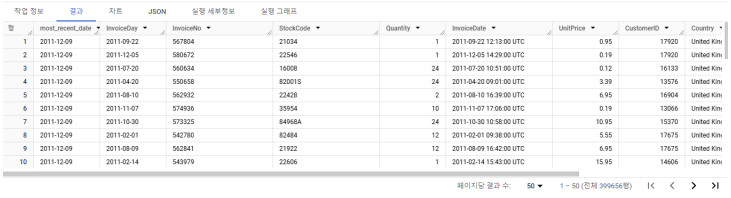
```
SELECT DATE(InvoiceDate) AS InvoiceDay, *
FROM avid-involution-439402-i8.modulabs_project.data
```



행	InvoiceDay	InvoiceNo	StockCode	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
1	2011-11-03	574301	85049E	12	2011-11-03 16:15:00 UTC	1.25	12544	Spain
2	2011-11-03	574301	84879	8	2011-11-03 16:15:00 UTC	1.69	12544	Spain
3	2011-11-03	574301	20971	12	2011-11-03 16:15:00 UTC	1.25	12544	Spain
4	2011-11-03	574301	85049A	12	2011-11-03 16:15:00 UTC	1.25	12544	Spain
5	2011-11-03	574301	22751	4	2011-11-03 16:15:00 UTC	3.75	12544	Spain
6	2011-11-03	574301	22144	6	2011-11-03 16:15:00 UTC	2.1	12544	Spain
7	2011-11-03	574301	23514	6	2011-11-03 16:15:00 UTC	2.08	12544	Spain
8	2011-11-03	574301	23240	6	2011-11-03 16:15:00 UTC	4.15	12544	Spain
9	2011-11-03	574301	22960	6	2011-11-03 16:15:00 UTC	4.25	12544	Spain
10	2011-11-03	574301	22734	6	2011-11-03 16:15:00 UTC	2.89	12544	Spain

- 가장 최근 구매 일자를 MAX() 함수로 찾아보기

```
SELECT
  MAX(DATE(InvoiceDate)) OVER() AS most_recent_date,
  DATE(InvoiceDate) AS InvoiceDay,
   *
FROM avid-involution-439402-i8.modulabs_project.data
```



행	most_recent_date	InvoiceDay	InvoiceNo	StockCode	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
1	2011-12-09	2011-09-22	567804	21034	1	2011-09-22 12:13:00 UTC	0.95	17920	United Kin
2	2011-12-09	2011-12-05	580672	22546	1	2011-12-05 14:29:00 UTC	0.19	17920	United Kin
3	2011-12-09	2011-07-20	560634	16008	24	2011-07-20 10:51:00 UTC	0.12	16133	United Kin
4	2011-12-09	2011-04-20	539658	82001S	24	2011-04-20 09:01:00 UTC	3.39	13576	United Kin
5	2011-12-09	2011-08-10	562932	22428	2	2011-08-10 16:39:00 UTC	6.95	16904	United Kin
6	2011-12-09	2011-11-07	574936	35954	10	2011-11-07 17:06:00 UTC	0.19	13066	United Kin
7	2011-12-09	2011-10-30	573325	84968A	24	2011-10-30 10:58:00 UTC	10.95	15370	United Kin
8	2011-12-09	2011-02-01	542780	82484	12	2011-02-01 09:38:00 UTC	5.55	17675	United Kin
9	2011-12-09	2011-08-09	562841	21922	12	2011-08-09 16:42:00 UTC	6.95	17675	United Kin
10	2011-12-09	2011-02-14	543979	22606	1	2011-02-14 15:43:00 UTC	15.95	14606	United Kin

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

```
SELECT
    CustomerID,
    MAX(DATE(InvoiceDate)) AS most_recent_date
FROM avid-involution-439402-i8.modulabs_project.data
GROUP BY CustomerID
```

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

행	CustomerID	most_recent_date
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

- **가장 최근 일자(`most_recent_date`)와 유저 별 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 avid-involution-439402-i8.modulabs_project.data
  GROUP BY CustomerID
);
```

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

행	CustomerID	recency
1	17408	163
2	16385	60
3	16131	51
4	13831	16
5	17941	130
6	15638	301
7	13606	29
8	17968	373
9	14652	77
10	14908	74

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

```
CREATE OR REPLACE TABLE avid-involution-439402-i8.modulabs_project.user_r AS
SELECT
  CustomerID,
  EXTRACT(DAY FROM MAX(InvoiceDay) OVER () - InvoiceDay) AS recency
```

```
FROM (
  SELECT
    CustomerID,
    MAX(DATE(InvoiceDate)) AS InvoiceDay
  FROM avid-involution-439402-i8.modulabs_project.data
  GROUP BY CustomerID
);
```

작업 정보 | 결과 | 실행 세부정보 | 실행 그래프

이 문으로 이름이 user_r인 새 테이블이 생성되었습니다.

user_r 쿼리 공유

스키마 | 세부정보 | 미리보기 | 테이블

행	CustomerID	recency
1	17754	0
2	15804	0
3	12748	0
4	17581	0
5	13426	0
6	15311	0
7	16626	0
8	18102	0
9	17001	0
10	16558	0
11	16705	0
12	13777	0
13	14397	0

Frequency

- **고객마다 고유한 InvoiceNo의 수를 세어보기**

```
SELECT
  CustomerID,
  COUNT(DISTINCT InvoiceNo) AS purchase_cnt
FROM avid-involution-439402-i8.modulabs_project.data
GROUP BY CustomerID;
```

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

행	CustomerID	purchase_cnt
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

- **각 고객 별로 구매한 아이템의 총 수량 더하기**

```
SELECT
  CustomerID,
  SUM(Quantity) AS item_cnt
FROM avid-involution-439402-i8.modulabs_project.data
GROUP BY CustomerID;
```

작업 정보 | 결과 | 차트 | JSON | 실행 세부정보

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

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

```
CREATE OR REPLACE TABLE avid-involution-439402-i8.modulabs_project.user_rf AS

----- 전체 거래 건수 계산
WITH purchase_cnt AS (
  SELECT
    CustomerID,
    COUNT(DISTINCT InvoiceNo) AS purchase_cnt
  FROM avid-involution-439402-i8.modulabs_project.data
  GROUP BY CustomerID
),

------ 구매한 아이템 총 수량 계산
item_cnt AS (
  SELECT
    CustomerID,
    SUM(Quantity) AS item_cnt
  FROM avid-involution-439402-i8.modulabs_project.data
  GROUP BY CustomerID
),
recency_data AS (          <--  오류가 반복되어 챗지피티 문의 결과임
  SELECT
    CustomerID,
    MAX(DATE(InvoiceDate)) AS recency
  FROM avid-involution-439402-i8.modulabs_project.data
  GROUP BY CustomerID
)
SELECT
  pc.CustomerID,
  pc.purchase_cnt,
  ic.item_cnt,
  rd.recency
FROM purchase_cnt AS pc
JOIN item_cnt AS ic
  ON pc.CustomerID = ic.CustomerID
JOIN recency_data AS rd
  ON pc.CustomerID = rd.CustomerID;
```

작업 정보 | 결과 | 실행 세부정보 | 실행 그래프

이 문으로 이름이 user_rf인 새 테이블이 생성되었습니다.

Monetary

- **고객별 총 지출액 계산 (소수점 첫째 자리에서 반올림)**

```
SELECT CustomerID,ROUND(SUM(UnitPrice * Quantity)) AS user_total
FROM avid-involution-439402-i8.modulabs_project.data
GROUP BY CustomerID;
```

[결과 이미지를 넣어주세요]

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

행	CustomerID	user_total
1	12544	300.0
2	13568	187.0
3	13824	1699.0
4	14080	46.0
5	14336	1615.0
6	14592	558.0
7	15104	969.0
8	15360	428.0
9	15872	316.0
10	16128	1880.0
11	16384	584.0
12	17152	1504.0

- **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 avid-involution-439402-i8.modulabs_project.user_rfm AS
SELECT
  rf.CustomerID AS CustomerID,
  rf.purchase_cnt,
  rf.item_cnt,
  rf.recency,
  ut.user_total,
  ROUND(ut.user_total / rf.purchase_cnt, 1) AS user_average
FROM avid-involution-439402-i8.modulabs_project.user_rf rf
LEFT JOIN (
  SELECT
    CustomerID,
    SUM(UnitPrice * Quantity) AS user_total
  FROM avid-involution-439402-i8.modulabs_project.data
  GROUP BY CustomerID
) ut
ON rf.CustomerID = ut.CustomerID;
```

작업 정보 | 결과 | 실행 세부정보 | 실행 그래프

이 문으로 이름이 user_rfm인 테이블이 교체되었습니다.

RFM 통합 테이블 출력하기

- **최종 user_rfm 테이블을 출력하기**

user_rfm 쿼리 공유 복사 스냅샷 삭제 내보내기

스키마 세부정보 미리보기 테이블 탐색기 미리보기 통계 계보 데이터

행	CustomerID	purchase_cnt	item_cnt	recency	user_total	user_average
1	15063	1	56	2011-05-10	370.799999...	370.8
2	13990	1	190	2011-05-10	311.3	311.3
3	17597	1	1176	2011-05-10	2044.37000...	2044.4
4	13052	1	222	2011-05-11	348.15	348.1
5	18133	1	1350	2011-05-11	931.499999...	931.5
6	13235	1	441	2011-05-11	1031.06999...	1031.1
7	12976	1	561	2011-05-12	738.6	738.6
8	16006	1	84	2011-05-12	101.4	101.4
9	17970	1	513	2011-05-12	562.79	562.8
10	14988	1	141	2011-05-12	334.539999...	334.5
11	14873	1	168	2011-05-13	519.68	519.7
12	12770	1	743	2011-05-13	1351.44999...	1351.4
13	14689	1	76	2011-05-15	112.800000...	112.8
14	15333	1	344	2011-05-15	1028.56	1028.6
15	17263	1	36	2011-05-15	63.4400000...	63.4
16	14489	1	299	2011-05-16	463.380000...	463.4
17	14888	1	184	2011-05-16	369.2	369.2
18	13976	1	154	2011-05-17	357.980000...	358.0
19	12690	1	103	2011-05-18	335.01	335.0
20	13572	1	534	2011-05-18	1384.25	1384.3
21	17556	1	101	2011-05-18	157.900000...	157.9
22	17871	1	182	2011-05-19	155.899999...	155.9
23	17245	1	75	2011-05-19	171.45	171.4

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 avid-involution-439402-i8.modulabs_project.user_data AS
WITH unique_products AS (
  SELECT
    CustomerID,
    COUNT(DISTINCT StockCode) AS unique_products
  FROM avid-involution-439402-i8.modulabs_project.data
  GROUP BY CustomerID
)
SELECT ur.*, up.* EXCEPT (CustomerID)
FROM avid-involution-439402-i8.modulabs_project.user_rfm AS ur
```

```
JOIN unique_products AS up
ON ur.CustomerID = up.CustomerID;
```

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 avid-involution-439402-i8.modulabs_project.user_data AS
WITH purchase_intervals AS (
   SELECT
    CustomerID,
    CASE WHEN ROUND(AVG(interval_), 2) IS NULL THEN 0 ELSE ROUND(AVG(interval_), 2) END AS average_inte
  FROM (
    SELECT
      CustomerID,
      DATE_DIFF(InvoiceDate, LAG(InvoiceDate) OVER (PARTITION BY CustomerID ORDER BY InvoiceDate), DAY)
    FROM
      avid-involution-439402-i8.modulabs_project.data
    WHERE CustomerID IS NOT NULL
  )
  GROUP BY CustomerID
)

SELECT u.*, pi.* EXCEPT (CustomerID)
FROM avid-involution-439402-i8.modulabs_project.user_data AS u
LEFT JOIN purchase_intervals AS pi
ON u.CustomerID = pi.CustomerID;
```

작업 정보 | 결과 | 실행 세부정보 | 실행 그래프

이 문으로 이름이 user_data인 테이블이 교체되었습니다.

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 avid-involution-439402-i8.modulabs_project.user_data AS
WITH TransactionInfo AS (
  SELECT
    CustomerID,
    COUNT(*) AS total_transactions,
    SUM(CASE WHEN InVoiceNo LIKE 'C%' THEN 1 ELSE 0 END) AS cancel_frequency
  FROM avid-involution-439402-i8.modulabs_project.data
  WHERE CustomerID IS NOT NULL
  GROUP BY CustomerID
)
SELECT
  u.*,
```

```
  t.* EXCEPT(CustomerID),
ROUND(t.cancel_frequency / t.total_transactions, 2) AS cancel_rate
FROM avid-involution-439402-i8.modulabs_project.user_data AS u
LEFT JOIN TransactionInfo AS t
ON u.CustomerID = t.CustomerID;
```

user_data

쿼리 공유 복사 스냅샷 삭제 내보내기

스키마 세부정보 미리보기 테이블 탐색기 미리보기 통계 계보 데이터 프로필 데

행	CustomerID	purchase_cnt	item_cnt	recency	user_total	user_average	unique_products
1	17331	1	16	2011-08-08	175.2	175.2	1
2	17443	1	504	2011-05-04	534.24	534.2	1
3	17752	1	192	2010-12-15	80.64	80.6	1
4	12814	1	48	2011-08-30	85.92	85.9	1
5	16323	1	50	2011-05-27	207.500000...	207.5	1
6	16344	1	18	2011-07-04	101.100000...	101.1	1
7	18133	1	1350	2011-05-11	931.499999...	931.5	1
8	14119	1	-2	2010-12-20	-19.9	-19.9	1
9	16765	1	4	2011-02-18	34.0	34.0	1
10	15668	1	72	2011-05-06	76.3200000...	76.3	1
11	16138	1	-1	2010-12-06	-7.95	-8.0	1
12	16093	1	20	2011-08-25	17.0	17.0	1
13	15753	1	144	2011-02-08	79.2	79.2	1
14	18068	1	6	2011-02-23	101.699999...	101.7	1
15	16953	1	10	2011-11-09	20.8	20.8	1
16	17102	1	2	2011-03-23	25.5	25.5	1
17	17986	1	10	2011-10-14	20.8	20.8	1
18	15940	1	4	2011-02-01	35.8	35.8	1
19	17291	1	72	2011-02-04	550.800000...	550.8	1
20	18233	1	4	2011-01-18	440.0	440.0	1
21	16078	1	16	2011-03-01	79.2	79.2	1
22	17923	1	50	2011-03-02	207.500000...	207.5	1
23	13302	1	5	2011-07-07	63.75	63.8	1

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

```
SELECT *
FROM avid-involution-439402-i8.modulabs_project.user_data
LIMIT 50
```

쿼리 결과

결과 저장 데이터 탐색

작업 정보 결과 자트 JSON 실행 세부정보 실행 그래프

행	stomerID	purchase_cnt	item_cnt	recency	user_total	user_average	unique_products	average_interval	total_transactions	cancel_frequency	cancel_rate
34	16323	1	50	2011-05-27	207.5000000000...	207.5	1	0.0	1	0	
35	16738	1	3	2011-02-15	3.75	3.8	1	0.0	1	0	
36	18184	1	60	2011-11-24	49.8	49.8	1	0.0	1	0	
37	12814	1	48	2011-08-30	85.92	85.9	1	0.0	1	0	
38	17331	1	16	2011-08-08	175.2	175.2	1	0.0	1	0	
39	16138	1	-1	2010-12-06	-7.95	-8.0	1	0.0	1	1	
40	13829	1	-12	2010-12-15	-102.0	-102.0	1	0.0	1	1	
41	13099	1	288	2011-09-01	207.3599999999...	207.4	1	0.0	1	0	
42	16061	1	-1	2011-03-15	-29.95	-29.9	1	0.0	1	1	
43	15524	1	4	2011-11-15	440.0	440.0	1	0.0	1	0	
44	15940	1	4	2011-02-01	35.8	35.8	1	0.0	1	0	
45	14576	1	12	2010-12-02	35.4000000000...	35.4	1	0.0	1	0	
46	14679	1	-1	2010-12-03	-2.55	-2.5	1	0.0	1	1	
47	16765	1	4	2011-02-18	34.0	34.0	1	0.0	1	0	
48	13747	1	8	2010-12-01	79.6	79.6	1	0.0	1	0	
49	16148	1	72	2011-02-16	76.3200000000...	76.3	1	0.0	1	0	
50	16093	1	20	2011-08-25	17.0	17.0	1	0.0	1	0	

페이지당 결과 수: 50 1 – 50 (전체 50행)

회고

순차적으로 진행하다가 해결점을 찾지 못하고 건너 뛴 상태에서 진행했다. 다시 처음부터 진행하느라 시간이 많이 소요되었다. 구문 작성을 진행하다가 결정적으로 막히면 챗지피티에 문의해 해결하곤 했다. 최종 결과물이 나왔지만 완전하게 결과가 산출되었는지는 확신하기 어렵다.