SELECT *

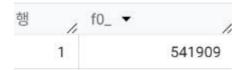
FROM 'modulabs_project.data'

LIMIT 10;

행	InvoiceNo ▼	StockCode ▼	Description ▼	Quantity •	InvoiceDate ▼	// UnitPrice ▼
1	536365	85123A	WHITE HANGING HEART T-LIG	6	2010-12-01 08:26:00 UTC	2.55
2	536365	71053	WHITE METAL LANTERN	6	2010-12-01 08:26:00 UTC	3.39
3	536365	84406B	CREAM CUPID HEARTS COAT H	8	2010-12-01 08:26:00 UTC	2.75
4	536365	84029G	KNITTED UNION FLAG HOT WA	6	2010-12-01 08:26:00 UTC	3.39
5	536365	84029E	RED WOOLLY HOTTIE WHITE H	6	2010-12-01 08:26:00 UTC	3.39
6	536365	22752	SET 7 BABUSHKA NESTING BO	2	2010-12-01 08:26:00 UTC	7.65
7	536365	21730	GLASS STAR FROSTED T-LIGHT	6	2010-12-01 08:26:00 UTC	4.25
8	536366	22633	HAND WARMER UNION JACK	6	2010-12-01 08:28:00 UTC	1.85
9	536366	22632	HAND WARMER RED POLKA DOT	6	2010-12-01 08:28:00 UTC	1.85
10	536367	84879	ASSORTED COLOUR BIRD ORN	32	2010-12-01 08:34:00 UTC	1.69

SELECT COUNT(*)

FROM `modulabs_project.data`;



SELECT

COUNT(InvoiceNo) AS InvoiceNo.

COUNT(StockCode) AS StockCode,

COUNT(Description) AS Description,

COUNT(Quantity) AS Quantity,

COUNT(InvoiceDate) AS InvoiceDate,

COUNT(UnitPrice) AS UnitPrice,

COUNT(CustomerID) AS CustomerID.

COUNT(Country) AS Country

FROM `modulabs_project.data`;



아래는 CASE WHEN 을 사용하여 InvoiceNo의 결측치 비율을 계산하는 쿼리문 예시입니다. # 예시

SELECT

'InvoiceNo' AS column_name,

ROUND(SUM(CASE WHEN InvoiceNo IS NULL THEN 1 ELSE 0 END) / COUNT(*)

* 100, 2) AS missing_percentage

FROM modulabs_project.data;



해당 쿼리문을 다른 컬럼에도 동일하게 반영한 후, UNION ALL로 연결해 보세요.

-- 각 컬럼의 결측치 비율 계산 쿼리

SELECT

'InvoiceNo' AS column_name,

ROUND(SUM(CASE WHEN InvoiceNo IS NULL THEN 1 ELSE 0 END) / COUNT(*)

* 100, 2) AS missing_percentage

FROM `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_percentage

FROM '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_percentage

FROM '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_percentage

FROM '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_percentage FROM `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_percentage

FROM `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_percentage FROM `modulabs_project.data`

UNION ALL

SELECT

'Country' AS column_name,

 ${\tt ROUND(SUM(CASE\ WHEN\ Country\ IS\ NULL\ THEN\ 1\ ELSE\ 0\ END)\ /\ COUNT(*)\ *} \\ 100,\ 2)\ {\tt AS\ missing_percentage}$

FROM `modulabs_project.data`;

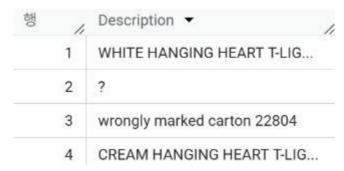
행 //	column_name ▼	missing_percenta
1	CustomerID	24.93
2	Country	0.0
3	InvoiceDate	0.0
4	Quantity	0.0
5	UnitPrice	0.0
6	Description	0.27
7	StockCode	0.0
8	InvoiceNo	0.0

StockCode = '85123A'의 Description을 추출하는 쿼리문을 작성해 보세요.

SELECT DISTINCT Description

FROM `modulabs_project.data`

WHERE StockCode = '85123A';



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

① 이 문으로 data의 행 135,080개가 삭제되었습니다.

SELECT

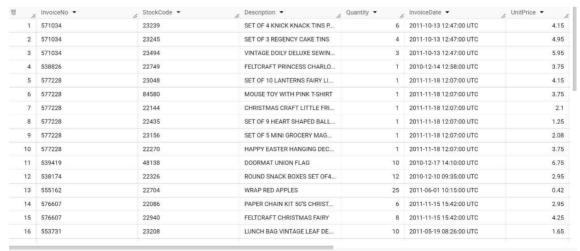
InvoiceNo, StockCode, Description, Quantity, InvoiceDate, UnitPrice, CustomerID, Country,

COUNT(*) AS duplicate_count

FROM 'modulabs_project.data'

GROUP BY InvoiceNo, StockCode, Description, Quantity, InvoiceDate, UnitPrice, CustomerID, Country

HAVING COUNT(*) > 1;



CREATE OR REPLACE TABLE 'modulabs_project.data' AS SELECT DISTINCT \ast

FROM `modulabs_project.data`;



이 문으로 이름이 data인 테이블이 교체되었습니다.

SELECT COUNT(DISTINCT InvoiceNo) AS unique_invoice_count FROM `modulabs_project.data`;



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

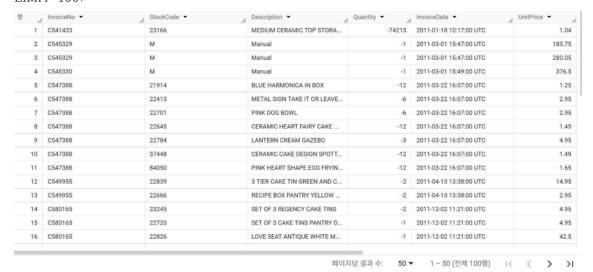


SELECT *

FROM 'modulabs_project.data'

WHERE InvoiceNo LIKE 'C%'

LIMIT 100;



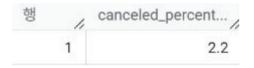
SELECT

ROUND(

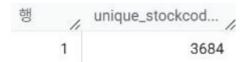
SUM(CASE WHEN InvoiceNo LIKE 'C%' THEN 1 ELSE 0 END) / COUNT(*) * 100,

1) AS canceled_percentage

FROM `modulabs_project.data`;



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



SELECT StockCode, COUNT(*) AS sell_cnt FROM `modulabs_project.data` GROUP BY StockCode
ORDER BY sell_cnt DESC
LIMIT 10;

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

```
WITH UniqueStockCodes AS (

SELECT DISTINCT StockCode

FROM '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;
```

ock_cnt *	number_count ▼ // sto	행 //
3676	5	1
7	0	2
1	1	3

```
SELECT DISTINCT StockCode, number_count
FROM (
    SELECT StockCode,
    LENGTH(StockCode) - LENGTH(REGEXP_REPLACE(StockCode, r'[0-9]', '')) AS
number_count
    FROM `modulabs_project.data`
)
```

WHERE number_count <= 1;</pre>

행 //	StockCode ▼	/ number_count ▼ //
1	POST	0
2	М	0
3	C2	1
4	D	0
5	BANK CHARGES	0
6	PADS	0
7	DOT	0
8	CRUK	0

SELECT

ROUND(SUM(CASE WHEN StockCode IN ('POST', 'D', 'C2', 'M', 'BANK CHARGES', 'PADS', 'DOT', 'CRUK')

THEN 1 ELSE 0 END) / COUNT(*) * 100, 2) AS outlier_percentage FROM `modulabs_project.data`;



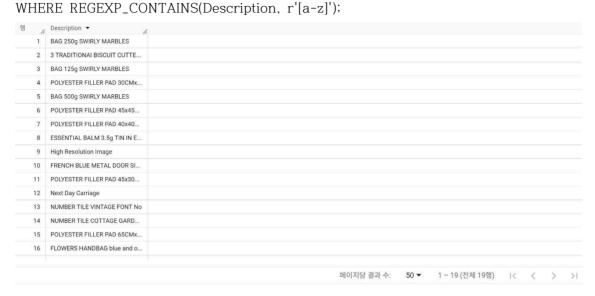
```
DELETE FROM `modulabs_project.data`
WHERE StockCode IN (
    SELECT DISTINCT StockCode
FROM (
    SELECT StockCode,
         LENGTH(StockCode) - LENGTH(REGEXP_REPLACE(StockCode, r'[0-9]', ''))
AS number_count
    FROM `modulabs_project.data`
)
WHERE number_count <= 1
);</pre>
```

이 문으로 data의 행 1,915개가 삭제되었습니다.

SELECT Description, COUNT(*) AS description_cnt FROM `modulabs_project.data` GROUP BY Description
ORDER BY description_cnt DESC
LIMIT 30;

1		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 D	1224
8	LUNCH BAG BLACK SKULL.	1099
9	PACK OF 72 RETROSPOT CAKE	1062
10	SPOTTY BUNTING	1026
11	PAPER CHAIN KIT 50'S CHRIST	1013
12	LUNCH BAG SPACEBOY DESIGN	1006
13	LUNCH BAG CARS BLUE	1000
14	HEART OF WICKER SMALL	990
15	NATURAL SLATE HEART CHAL	989
16	JAM MAKING SET WITH JARS	966

SELECT DISTINCT Description FROM 'modulabs_project.data'

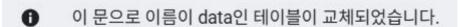


DELETE

FROM `modulabs_project.data`
WHERE Description IN (
 'Next Day Carriage',
 'High Resolution Image',
 'Manual',
 'DOTCOM POSTAGE',
 'Amazon Adjustment',
 'Adjust bad debt',
 'CARRIAGE'
);

CREATE OR REPLACE TABLE `modulabs_project.data` AS SELECT

* EXCEPT (Description), UPPER(Description) AS Description FROM `modulabs_project.data`;

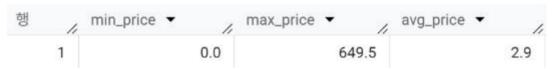


MIN(UnitPrice) AS min_price,

MAX(UnitPrice) AS max_price,

ROUND(AVG(UnitPrice), 2) AS avg_price

FROM `modulabs_project.data`;



SELECT

COUNT(*) AS cnt_quantity,

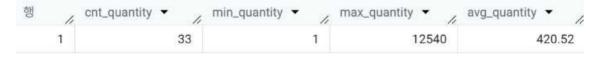
MIN(Quantity) AS min_quantity,

MAX(Quantity) AS max_quantity,

ROUND(AVG(Quantity), 2) AS avg_quantity

FROM 'modulabs_project.data'

WHERE UnitPrice = 0;



CREATE OR REPLACE TABLE 'modulabs_project.data' AS

SELECT *

FROM 'modulabs_project.data'

WHERE UnitPrice > 0;



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

7,	InvoiceDay ▼ //	InvoiceNo ▼ //	StockCode ▼ //	Quantity ▼	InvoiceDate ▼	UnitPrice ▼	Customerli
1	2011-01-18	541431	23166	74215	2011-01-18 10:01:00 UTC	1.04	
2	2011-01-18	C541433	23166	-74215	2011-01-18 10:17:00 UTC	1.04	
3	2010-12-07	537626	84997C	6	2010-12-07 14:57:00 UTC	3.75	
4	2010-12-07	537626	22497	4	2010-12-07 14:57:00 UTC	4.25	
5	2010-12-07	537626	22375	4	2010-12-07 14:57:00 UTC	4.25	
6	2010-12-07	537626	22771	12	2010-12-07 14:57:00 UTC	1.25	
7	2010-12-07	537626	22195	12	2010-12-07 14:57;00 UTC	1.65	
8	2010-12-07	537626	22492	36	2010-12-07 14:57:00 UTC	0.65	
9	2010-12-07	537626	22494	12	2010-12-07 14:57:00 UTC	1.25	
10	2010-12-07	537626	84969	6	2010-12-07 14:57:00 UTC	4.25	
11	2010-12-07	537626	22772	12	2010-12-07 14:57:00 UTC	1.25	
12	2010-12-07	537626	85167B	30	2010-12-07 14:57:00 UTC	1.25	
13	2010-12-07	537626	21064	6	2010-12-07 14:57:00 UTC	5.95	
14	2010-12-07	537626	22212	6	2010-12-07 14:57:00 UTC	2.1	
15	2010-12-07	537626	21171	12	2010-12-07 14:57:00 UTC	1.45	
16	2010-12-07	537626	21731	12	2010-12-07 14:57:00 UTC	1.65	

SELECT

$$\label{eq:max_def} \begin{split} \text{MAX}(\text{DATE}(\text{InvoiceDate})) \ \ \text{OVER()} \ \ \text{AS most_recent_date}, \\ \text{DATE}(\text{InvoiceDate}) \ \ \text{AS InvoiceDay}, \end{split}$$

FROM `modulabs_project.data`;

/	most_recent_date */	36.	InvoiceNo ▼		Quantity ▼ //	InvoiceDate ▼	/ UnitPrice •
1	2011-12-09	2011-10-18	571705	23008	2	2011-10-18 15:22:00 UTC	
2	2011-12-09	2011-09-28	568651	22947	2	2011-09-28 12:04:00 UTC	
3	2011-12-09	2011-08-02	562046	47590B	3	2011-08-02 10:34:00 UTC	
4	2011-12-09	2011-03-03	545475	21984	432	2011-03-03 10:59:00 UTC	
5	2011-12-09	2011-08-18	563614	22749	48	2011-08-18 08:51:00 UTC	
6	2011-12-09	2011-11-30	579692	22693	192	2011-11-30 14:05:00 UTC	
7	2011-12-09	2011-08-22	563950	23375	10	2011-08-22 10:39:00 UTC	
8	2011-12-09	2011-11-07	574885	23084	48	2011-11-07 14:01:00 UTC	
9	2011-12-09	2011-10-17	571312	84251G	12	2011-10-17 10:42:00 UTC	
10	2011-12-09	2011-08-31	564856	35599B	12	2011-08-31 09:11:00 UTC	
11	2011-12-09	2011-08-31	564856	23094	3	2011-08-31 09:11:00 UTC	
12	2011-12-09	2011-05-27	554940	23108	2	2011-05-27 13:49:00 UTC	
13	2011-12-09	2011-11-11	575886	22059	8	2011-11-11 13:57:00 UTC	
14	2011-12-09	2011-03-08	545916	21843	5	2011-03-08 11:46:00 UTC	
15	2011-12-09	2011-08-01	561911	23108	2	2011-08-01 10:26:00 UTC	
16	2011-12-09	2011-01-11	540785	84997D	360	2011-01-11 11:39:00 UTC	

CustomerID,

MAX(DATE(InvoiceDate)) AS InvoiceDay

FROM `modulabs_project.data`

GROUP BY CustomerID;

1	12346	2011-01-18						
2	12347	2011-12-07						
3	12348	2011-09-25						
4	12349	2011-11-21						
5	12350	2011-02-02						
6	12352	2011-11-03						
7	12353	2011-05-19						
8	12354	2011-04-21						
9	12355	2011-05-09						
10	12356	2011-11-17						
11	12357	2011-11-06						
12	12358	2011-12-08						
13	12359	2011-12-02						
14	12360	2011-10-18						
15	12361	2011-02-25						
16	12362	2011-12-06						
			페이지당 결과 수:	50 ▼	1 - 50 (전체 4362행)	1<	-	· >

SELECT

CustomerID,

EXTRACT(DAY FROM MAX(InvoiceDay) OVER () - InvoiceDay) AS recency

FROM (

SELECT

CustomerID,

MAX(DATE(InvoiceDate)) AS InvoiceDay

FROM ecstatic-valve-465201-c2.modulabs_project.data

GROUP BY CustomerID

);

1	12363	109						
2	12397	35						
3	12520	79						
4	12614	277						
5	12851	96						
6	12854	78						
7	12875	143						
8	12952	5						
9	13015	53						
10	13118	19						
11	13235	212						
12	13670	75						
13	13807	367						
14	13837	211						
15	13951	176						
16	14064	29						

```
CREATE OR REPLACE TABLE `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 `modulabs_project.data`

GROUP BY CustomerID
);
```

```
CREATE OR REPLACE TABLE `ecstatic-valve-465201-c2.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 `ecstatic-valve-465201-c2.modulabs_project.data`

GROUP BY CustomerID
);
```

이 문으로 이름이 user_r인 테이블이 교체되었습니다.

CustomerID,

COUNT(DISTINCT InvoiceNo) AS purchase_cnt

FROM 'ecstatic-valve-465201-c2.modulabs_project.data'

GROUP BY CustomerID;

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

SELECT

CustomerID,

SUM(Quantity) AS item_cnt

FROM `ecstatic-valve-465201-c2.modulabs_project.data`

GROUP BY CustomerID;



```
CREATE OR REPLACE TABLE 'ecstatic-valve-465201-c2.modulabs_project.user_rf'
AS
-- (1) 전체 거래 건수 계산
WITH purchase_cnt AS (
 SELECT
   CustomerID,
   COUNT(DISTINCT InvoiceNo) AS purchase_cnt
 FROM 'ecstatic-valve-465201-c2.modulabs_project.data'
 GROUP BY CustomerID
).
-- (2) 구매한 아이템 총 수량 계산
item_cnt AS (
 SELECT
   CustomerID,
   SUM(Quantity) AS item_cnt
 FROM `ecstatic-valve-465201-c2.modulabs_project.data`
 GROUP BY CustomerID
)
-- 기존 user_r 테이블과 결합
SELECT
 pc.CustomerID,
  pc.purchase_cnt,
 ic.item_cnt,
 ur.recency
FROM purchase_cnt AS pc
JOIN item_cnt AS ic
 ON pc.CustomerID = ic.CustomerID
JOIN 'ecstatic-valve-465201-c2.modulabs_project.user_r' AS ur
```

이 문으로 이름이 user_rf인 테이블이 교체되었습니다.

ON pc.CustomerID = ur.CustomerID;

CustomerID,

ROUND(SUM(UnitPrice * Quantity), 1) AS user_total

FROM `ecstatic-valve-465201-c2.modulabs_project.data`

GROUP BY CustomerID;



CREATE OR REPLACE TABLE `ecstatic-valve-465201-c2.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 'ecstatic-valve-465201-c2.modulabs_project.user_rf' rf

LEFT JOIN (

-- 고객 별 총 지출액

SELECT

CustomerID.

ROUND(SUM(UnitPrice * Quantity), 1) AS user_total

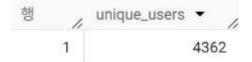
FROM 'ecstatic-valve-465201-c2.modulabs_project.data'

GROUP BY CustomerID

) 11t

ON rf.CustomerID = ut.CustomerID;

SELECT COUNT(DISTINCT CustomerID) AS unique_users FROM `ecstatic-valve-465201-c2.modulabs_project.user_rfm`;



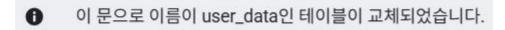
```
    이 문으로 이름이 user_data인 테이블이 교체되었습니다.
```

```
CREATE OR REPLACE TABLE `ecstatic-valve-465201-c2.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(DATE(InvoiceDate),
              LAG(DATE(InvoiceDate)) OVER (PARTITION BY CustomerID ORDER
BY DATE(InvoiceDate)),
              DAY) AS interval_
   FROM
     `ecstatic-valve-465201-c2.modulabs_project.data`
   WHERE CustomerID IS NOT NULL
 GROUP BY CustomerID
)
SELECT
 u.*.
 pi.average_interval
FROM 'ecstatic-valve-465201-c2.modulabs_project.user_data' AS u
LEFT JOIN purchase_intervals AS pi
ON u.CustomerID = pi.CustomerID;
        이 문으로 이름이 user_data인 테이블이 교체되었습니다.
CREATE OR REPLACE TABLE ecstatic-valve-465201-c2.modulabs_project.user_data
AS
WITH TransactionInfo AS (
 SELECT
   CustomerID.
   COUNT(DISTINCT InvoiceNo) AS total_transactions,
   COUNT(DISTINCT CASE WHEN InvoiceNo LIKE 'C%' THEN InvoiceNo END) AS
cancel_frequency
 FROM ecstatic-valve-465201-c2.modulabs_project.data
 WHERE CustomerID IS NOT NULL
 GROUP BY CustomerID
)
SELECT
 u.*.
```

t.cancel_frequency,

ROUND(t.cancel_frequency / t.total_transactions, 2) AS cancel_rate FROM ecstatic-valve-465201-c2.modulabs_project.user_data AS u LEFT JOIN TransactionInfo AS t ON u.CustomerID = t.CustomerID;



SELECT *
FROM `ecstatic-valve-465201-c2.modulabs_project.user_data`;

1	15992	1	17	3	42.0	42.0	3	0.0
2	13194	1	26	135	60.7	60.7	3	0.0
3	14816	1	5	197	271.8	271.8	4	0.0
4	12968	1	73	112	135.9	135.9	5	0.0
5	13095	1	144	70	74.4	74.4	6	0.0
6	18191	1	140	261	207.8	207.8	7	0.0
7	13484	1.	44	241	313.8	313.8	8	0.0
8	12551	1	100	357	168.0	168.0	10	0.0
9	15895	1	94	149	179.2	179.2	11	0.0
10	13898	1	133	325	155.9	155.9	12	0.0
11	14523	1	251	108	241.1	241.1	12	0.0
12	16756	1.	106	214	239.4	239.4	13	0.0
13	13127	10	127	77	259.2	259.2	15	0.0
14	13889	1	146	92	416.9	416.9	15	0.0
15	16183	1	113	67	338.7	338.7	15	0.0
16	12561	1	177	302	238.8	238.8	16	0.0

페이지당 결과 수: 50 ▼ 1 - 50 (전체 4362행) |< **> >**