BUS211f1 Analyzing Big Data I

Fall 2018

Project 2: Sam’s Club SQL Queries

Team members: Yudan Ding, Yunqiu(Julie) Li, Kun Qiu, Jiahui Zhong

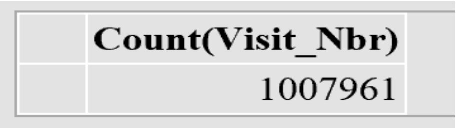
Introduction: An important phase in any data analytics project is to *understand the available data within the business context*. The first several queries here are basically an exploration of the Sam’s Club database.

Query 1 – Store Visits

a. How many store visits occur in our database? Just paste in your code and describe in one sentence.

```
SELECT count(Visit_Nbr)

FROM store_visits;
```

Output:



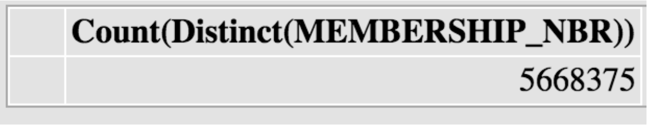
	Count(Visit_Nbr)
	1007961

1007961 store visits occurred in our database.

b. How many members do we have in our database?

```
SELECT count(DISTINCT Membership_Nbr)

FROM  member_index;
```

Output**:**



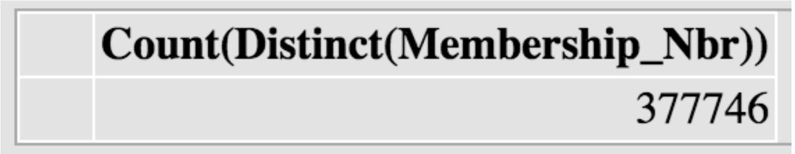
	Count(Distinct(MEMBERSHIP_NBR))
	5668375

There are currently 5668375 members in the database.

c. How many members record a transaction in any store during our sample?

```
SELECT count(DISTINCT Membership_Nbr)

FROM store_visits

WHERE Transaction_Date is not NULL;
```

Output:



	Count(Distinct(Membership_Nbr))
	377746

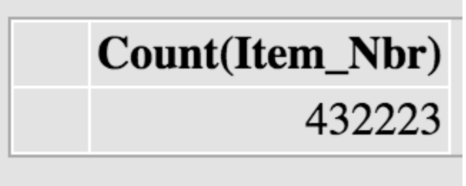
377746 members recorded a transaction in any store in our sample.

Query 2 – Item Scans

a. How many items are recorded in the Sam's Club database?

```
SELECT count(Item_nbr)

FROM item_desc;
```

Output:



	Count(Item_Nbr)
	432223

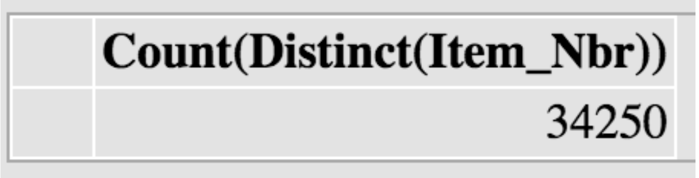
432223 items are recorded in Sam's Club database.

b. How many different items were bought during the available date range in our sample? ***(5 Pts)***

```
SELECT count(DISTINCT Item_nbr)

FROM item_scan
```

```
Where transaction_date is not NULL;
```

Output:



	Count(Distinct(Item_Nbr))
	34250

34250 distinct items were bought during the available date range in our sample. It is a great variety.

c. Given parts (a) and (b) from Queries 1 and 2 above, comment on the sampling of the item_scan and store_visits data?

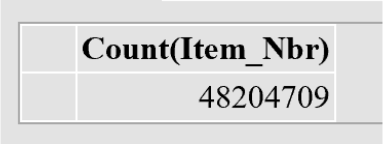
It's a fairly large dataset with many visits, members, and transactions. However, during the available date range, there is only a relatively small proportion (34250/432223 = 7.92%) of items that have been bought by customers. What's more, we can see the proportion of customers with the membership who has been to the stores is also quite low (37746/5668375 = 6.66%), so we can see the size of sample is very small. This may result from less item variety available during the available date range or customers looking for certain items during the available date range. The small size of sample may lead to inconsistency and bias in population. Hence, we suggest Sam's Club enlarge the size of sample.

d. Which variable(s) *should* identify a single row of item scan in the database? Determine the number of unique rows identified by the variable(s)? Report discrepancies, if any.

Step 1:

```
SELECT count(item_nbr)
FROM item_scan
```

Output:

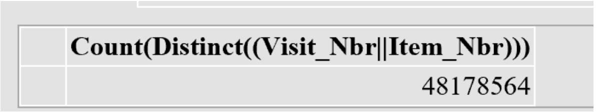


	Count(Item_Nbr)
	48204709

Step 2:

```
SELECT count(DISTINCT((Visit_Nbr|| Item_Nbr)))

FROM item_scan;
```

Output:



Count(Distinct((Visit_Nbr||Item_Nbr)))
48178564

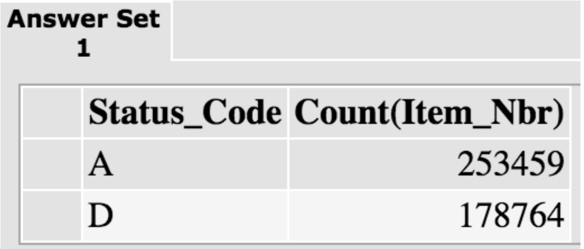
We decided to choose Visit_Nbr and Item_Nbr to identify a single row of item scan. Even though they could potentially serve as primary keys, they don't actually identify unique rows individually. We assumed that one transaction means a scan of an item, which will be reflected by item_nbrs, and there can be many transactions in a certain visit (visit_nbr). Hence, combining these two primary keys would lead to a single row of item scan in the database. According to code from Step 1, there are 48204709 items in the item scan, however, the output of Step 2, in which we use both visit_nbr and item_nbr, indicates there are 48178564 items, so the discrepancies are 26145. This could happen when items with same item_nbr have been scanned for more than once.

Query 3 – Item Scans (Cont.)

a. We know from the documentation that there are multiple status codes for items, which are only indicated by a one-letter code. How many items are for each status code?

```
SELECT status_code, count(item_nbr)
FROM item_desc
GROUP BY status_code;
```

Output:



Answer Set 1

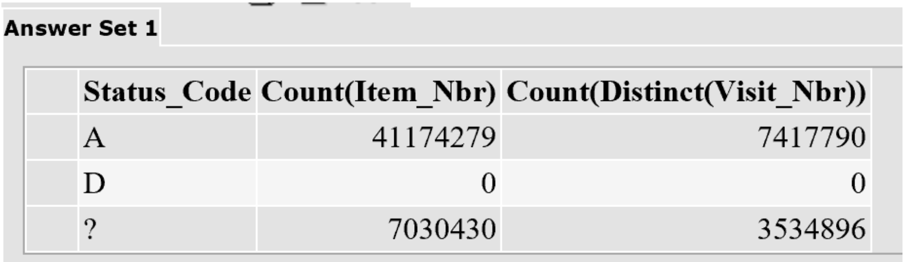
Status_Code	Count(Item_Nbr)
A	253459
D	178764

We can tell from the output that there are 253459 items under the status code "Active"(A), and there are 178764 items under status code "Deactive"(D).

b. Determine the **total number of item scans per status_code** in the database. Your result should be a 3-column table listing the status code, the number of scans for items for that code, total number of visits for that type. Again, what does this tell you about the sampling and the item_desc table?

```
SELECT status_code, count(item_scan.item_nbr), count(DISTINCT item_scan.visit_nbr)
FROM item_scan
FULL JOIN item_desc
ON item_scan.item_nbr = item_desc.item_nbr
GROUP BY status_code;
```

Output:



Answer Set 1

	Status_Code	Count(Item_Nbr)	Count(Distinct(Visit_Nbr))
	A	41174279	7417790
	D	0	0
	?	7030430	3534896

For status code “Active”, there are 41174279 scans for items for that code, and the total number of visits for that type is 7417790. For status code “Deactive”, there is no record in item_scan entity, which means Deactive items will not be sold. The “?” stands for the item_nbrs exist in item_scan but not in item_desc, that means the items which are sold but not recorded.

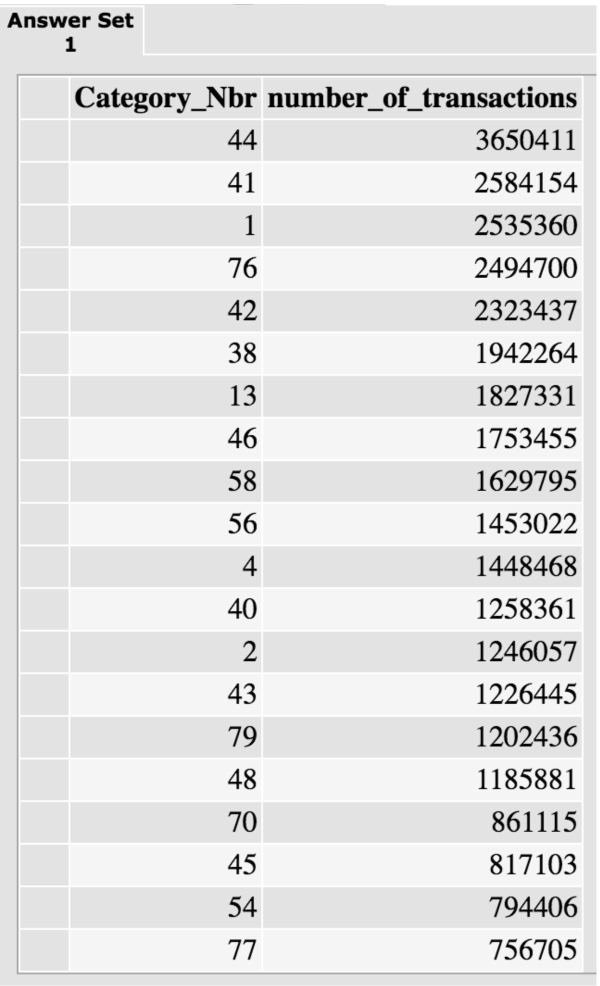
Query 4 – Top 20 Categories

Get the top 20 categories in terms of the number of transactions or total dollar sales. Your result should be a 2-column table listing the category number, and the number of transactions/total dollar sales. Look up the 3 top-earning categories, and describe them in a sentence. Include the code for the category exploration here, and summarize the results of the exploration in the description.

Top 20 categories in terms of the number of transactions

```
SELECT Top 20 category_nbr, count(*) AS number_of_transactions

FROM item_scan

INNER JOIN item_desc

ON item_scan.item_nbr = item_desc.item_nbr

GROUP BY category_nbr

ORDER BY number_of_transactions desc;
```

Output:

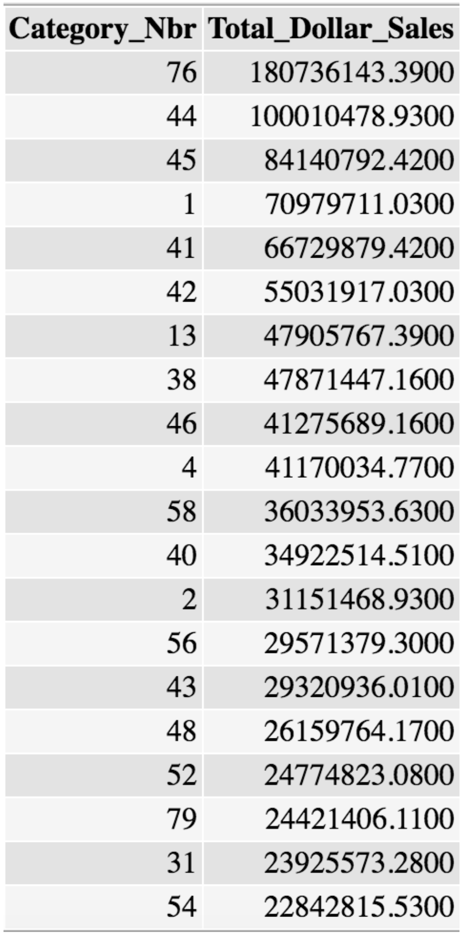


Category_Nbr	number_of_transactions
44	3650411
41	2584154
1	2535360
76	2494700
42	2323437
38	1942264
13	1827331
46	1753455
58	1629795
56	1453022
4	1448468
40	1258361
2	1246057
43	1226445
79	1202436
48	1185881
70	861115
45	817103
54	794406
77	756705

Top 20 categories in terms of total dollar sales:

```
SELECT Top 20 category_nbr, sum(unit_retail_amount*item_quantity) AS Total_Dollar_Sales

FROM item_scan

INNER JOIN item_desc

ON item_scan.item_nbr = item_desc.item_nbr

GROUP BY category_nbr

ORDER BY Total_Dollar_Sales DESC;
```

Output:



Category_Nbr	Total_Dollar_Sales
76	180736143.3900
44	100010478.9300
45	84140792.4200
1	70979711.0300
41	66729879.4200
42	55031917.0300
13	47905767.3900
38	47871447.1600
46	41275689.1600
4	41170034.7700
58	36033953.6300
40	34922514.5100
2	31151468.9300
56	29571379.3000
43	29320936.0100
48	26159764.1700
52	24774823.0800
79	24421406.1100
31	23925573.2800
54	22842815.5300

We can tell from the first query that 3 top-earning categories are category 76, 44 and 45

Queries for category exploration:

To explore what types of products are under each category mentioned above:

(1)

```
SELECT item_nbr, category_nbr, sub_category_nbr, primary_desc

FROM item_desc

WHERE category_nbr = 76
```

(2)

```
SELECT item_nbr, category_nbr, sub_category_nbr, primary_desc

FROM item_desc

WHERE category_nbr = 44
```

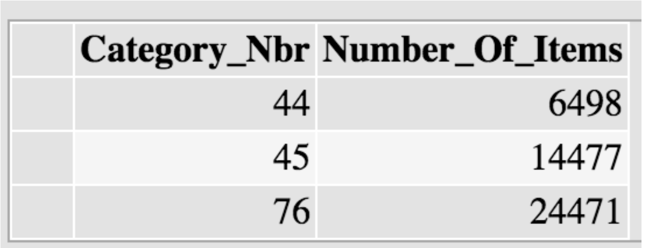
(3)

```
SELECT item_nbr, category_nbr, sub_category_nbr, primary_desc

FROM item_desc

WHERE category_nbr = 45
```

To explore the number of different items under each category mentioned above:

```
SELECT category_nbr, count(DISTINCT item_nbr) AS Number_Of_Items

FROM item_desc

GROUP BY category_nbr

having category_nbr = 76 or category_nbr = 44 or category_nbr = 45

ORDER BY Number_Of_Items;
```

Output:



	Category_Nbr	Number_Of_Items
	44	6498
	45	14477
	76	24471

Result:

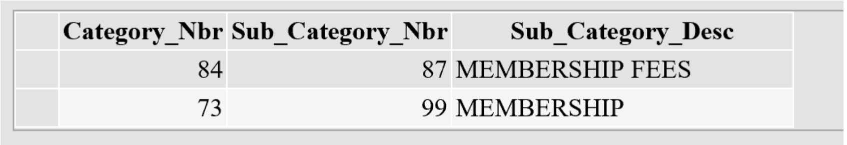
Through the above code, we can find out the subcategories under the three categories and the description of a specific item under each category. Category 44 would probably be meat and staple food as some of the items listed under it are chicken, steak, pizza, pancake etc. Category 76 may be deli meat or bakery and category 45 may be things like tobaccos. Looking at the number of different items under each category mentioned above, category 76 has the largest variety and largest total dollar sales among all categories.

Query 5 – Sam's Club Membership

a. Find the category-subcategory combination(s) for which the sub-category description includes the phrase "Membership". Your result should be a 3-column table with category number, sub-category number and sub-category description.

```
SELECT category_nbr, Sub_Category_Nbr, Sub_Category_Desc

FROM sub_category_desc

WHERE sub_category_desc like '%Membership%';
```

Output:



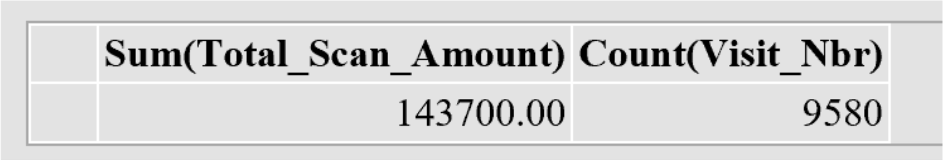
	Category_Nbr	Sub_Category_Nbr	Sub_Category_Desc
	84	87	MEMBERSHIP FEES
	73	99	MEMBERSHIP

There are two kinds of subcategory description that contains the word “Membership”. The first is category 84 and subcategory 87, the description is “Membership fees”; the second is category 73 and subcategory 99, the description is “membership”.

b. Find the total transaction amount and number of transactions for items in category 73, sub-category 99. These are annual fees paid by members. What’s the annual fee per member?

```
SELECT sum(total_scan_amount),count( visit_nbr) FROM item_scan
WHERE item_nbr in (SELECT item_nbr from item_desc WHERE category_nbr = 73 and
sub_category_nbr = 99);
```

Output:



	Sum(Total_Scan_Amount)	Count(Visit_Nbr)
	143700.00	9580

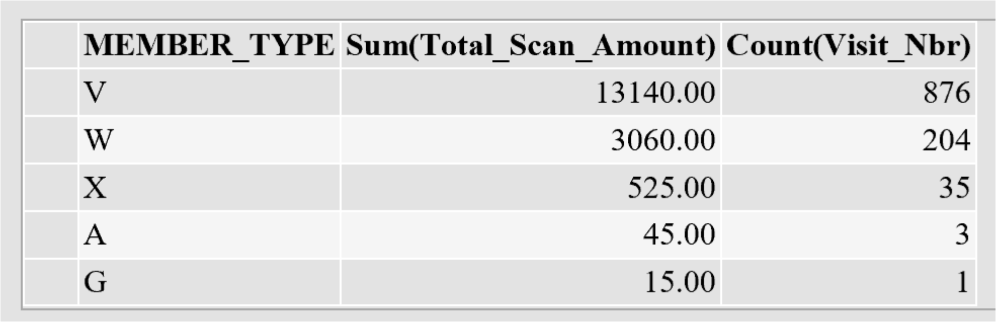
As we can see from the output, the total transaction amount is 143700 and the number of transaction for items in category 73, sub-category 99 is 9580. Hence, the annual fee per member is 143700/9580 = 15 dollars per year per person.

c. Add in membership information to the table from (b), and display total membership paid for all of the membership types. Your result should be a 3-column table with membership type, total transaction amount and number of transactions. Which membership type has the highest revenue?

```
SELECT member_type, sum(item_scan.total_scan_amount),count(item_scan.visit_nbr)
FROM member_index
```

```
LEFT JOIN store_visits
ON member_index.membership_nbr = store_visits.membership_nbr
JOIN item_scan
ON store_visits.visit_nbr = item_scan.visit_nbr
JOIN item_desc
ON item_scan.item_nbr = item_desc.item_nbr
WHERE category_nbr = 73 and sub_category_nbr = 99
GROUP BY member_type
ORDER BY sum(item_scan.total_scan_amount) desc;
```

Output:



MEMBER_TYPE	Sum(Total_Scan_Amount)	Count(Visit_Nbr)
V	13140.00	876
W	3060.00	204
X	525.00	35
A	45.00	3
G	15.00	1

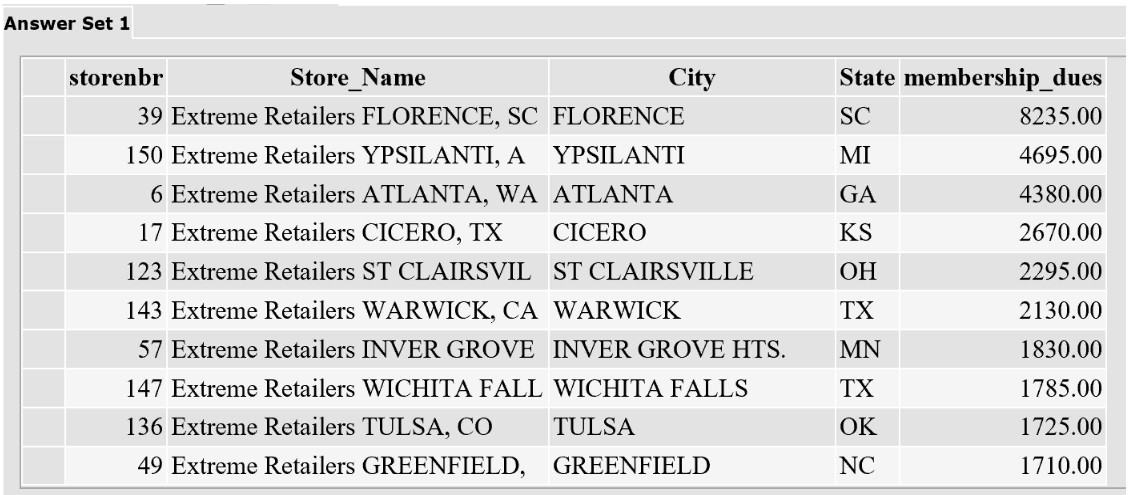
As we can see from the output, the member type v has the highest revenue.

Query 6 – Store Sales

a. Find the top 10 stores that generate the highest membership dues. Your table should contain the store number, store name, city, state and total membership dues collected for the top 10 stores in the descending order. Your final query table will have 5 columns and 10 rows of data.

```
SELECT top 10 item_scan.store_nbr as storenbr, store_name,city , state ,
sum(item_scan.total_scan_amount) as membership_dues
FROM item_scan,store_information,item_desc
WHERE item_scan.store_nbr = store_information.store_nbr
and item_scan.item_nbr = item_desc.item_nbr
and item_desc.category_nbr = 73
and item_desc.sub_category_nbr = 99
GROUP BY storenbr,store_name,city,state
ORDER BYmembership_dues desc;
```

Output:



Answer Set 1

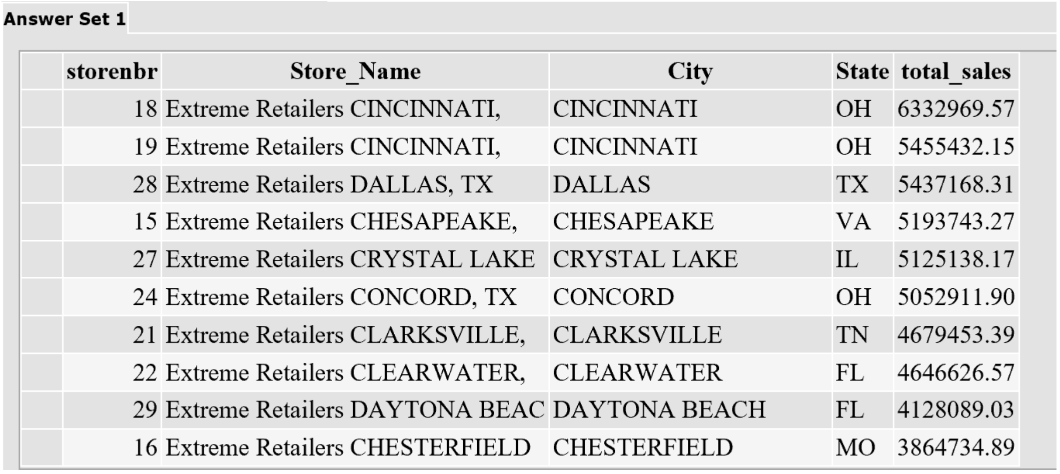
storenbr	Store_Name	City	State	membership_dues
39	Extreme Retailers FLORENCE, SC	FLORENCE	SC	8235.00
150	Extreme Retailers YPSILANTI, A	YPSILANTI	MI	4695.00
6	Extreme Retailers ATLANTA, WA	ATLANTA	GA	4380.00
17	Extreme Retailers CICERO, TX	CICERO	KS	2670.00
123	Extreme Retailers ST CLAIRSVIL	ST CLAIRSVILLE	OH	2295.00
143	Extreme Retailers WARWICK, CA	WARWICK	TX	2130.00
57	Extreme Retailers INVER GROVE	INVER GROVE HTS.	MN	1830.00
147	Extreme Retailers WICHITA FALL	WICHITA FALLS	TX	1785.00
136	Extreme Retailers TULSA, CO	TULSA	OK	1725.00
49	Extreme Retailers GREENFIELD,	GREENFIELD	NC	1710.00

As we can see from the output, the store whose number is 39, located in Florence, SC has the highest membership dues (8235), which is nearly twice as much as the store ranking the second.

b. Generate a similar list of the 10 stores that generate the highest sales. You should sum up the total revenue for each store (excluding sales tax) and list out the Store_Nbr, Store-name, City, State, and Total_Sales for the top 10 stores. Your final query table will have 5 columns and 10 rows of data. Is there any overlap between the stores in part (a) and (b)?

```
SELECT top 10 store_information.store_nbr as storenbr , store_name , city, state ,
sum(total_visit_amt - sales_tax_amt) as total_sales
FROM store_information
INNER JOIN store_visits on store_information.store_nbr = store_visits.store_nbr
GROUP BY storenbr,store_name,city,state
ORDER BY total_sales desc;
```

Output:



Answer Set 1

	storenbr	Store_Name	City	State	total_sales
	18	Extreme Retailers CINCINNATI,	CINCINNATI	OH	6332969.57
	19	Extreme Retailers CINCINNATI,	CINCINNATI	OH	5455432.15
	28	Extreme Retailers DALLAS, TX	DALLAS	TX	5437168.31
	15	Extreme Retailers CHESAPEAKE,	CHESAPEAKE	VA	5193743.27
	27	Extreme Retailers CRYSTAL LAKE	CRYSTAL LAKE	IL	5125138.17
	24	Extreme Retailers CONCORD, TX	CONCORD	OH	5052911.90
	21	Extreme Retailers CLARKSVILLE,	CLARKSVILLE	TN	4679453.39
	22	Extreme Retailers CLEARWATER,	CLEARWATER	FL	4646626.57
	29	Extreme Retailers DAYTONA BEAC	DAYTONA BEACH	FL	4128089.03
	16	Extreme Retailers CHESTERFIELD	CHESTERFIELD	MO	3864734.89

There is no overlap between part (a) and (b).

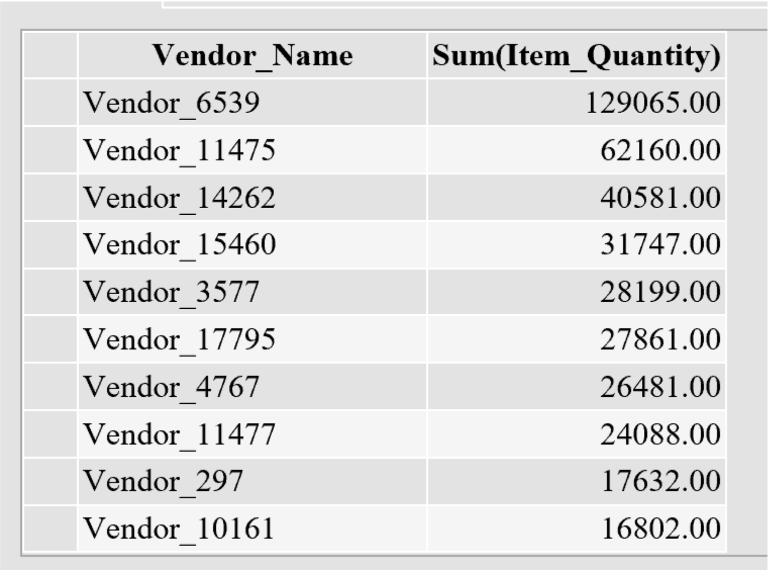
Query 7 – Investigating Vendors

In this query, we focus on the volume of products from different Vendors (suppliers) in two states: Kansas (KS) and Texas (TX). In your query, you'll create a new column called Total Units, which is the sum of item_quantity.

Write a new query that sums the "item_quantity" for all items supplied by vendors to Sam's Clubs in Kansas, and lists the 10 vendors with the highest sales. Note that within our database, Vendor names have been coded as numbers, such as "Vendor_3313". Your result table should have two columns: Vendor Name and Total Units.

```
SELECT top 10 vendor_name, sum(item_quantity)
FROM store_information
INNER JOIN item_scan on store_information.store_nbr = item_scan.store_nbr
INNER JOIN item_desc on item_scan.item_nbr = item_desc.item_nbr
WHERE store_information.state = 'KS' and item_desc.vendor_name is not null
GROUP BY vendor_name
ORDER BY sum(item_quantity) desc;
```

Output:

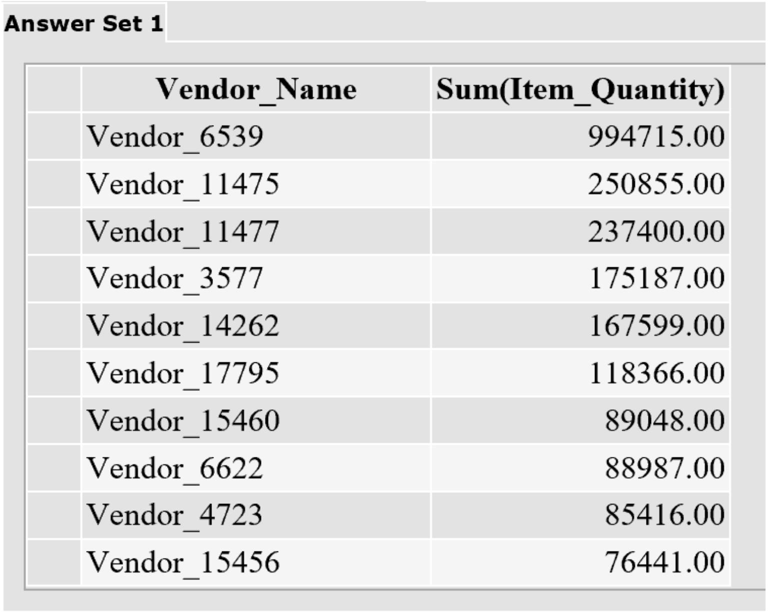


	Vendor_Name	Sum(Item_Quantity)
	Vendor_6539	129065.00
	Vendor_11475	62160.00
	Vendor_14262	40581.00
	Vendor_15460	31747.00
	Vendor_3577	28199.00
	Vendor_17795	27861.00
	Vendor_4767	26481.00
	Vendor_11477	24088.00
	Vendor_297	17632.00
	Vendor_10161	16802.00

Next, run the same query but this time analyze sales in Texas to find the top 10 vendors in that state.

```
SELECT top 10 vendor_name, sum(item_quantity)
FROM store_information
INNER JOIN item_scan on store_information.store_nbr = item_scan.store_nbr
LEFT JOIN item_desc on item_scan.item_nbr = item_desc.item_nbr
WHERE store_information.state = 'TX' and item_desc.vendor_name is not null
GROUP BY vendor_name
ORDER BY sum(item_quantity) desc;
```

Output:



Answer Set 1

	Vendor_Name	Sum(Item_Quantity)
	Vendor_6539	994715.00
	Vendor_11475	250855.00
	Vendor_11477	237400.00
	Vendor_3577	175187.00
	Vendor_14262	167599.00
	Vendor_17795	118366.00
	Vendor_15460	89048.00
	Vendor_6622	88987.00
	Vendor_4723	85416.00
	Vendor_15456	76441.00

Vendor_6539, Vendor_11475, Vendor_14262, Vendor_15460, Vendor_3577, Vendor_17795, Vendor_11477 are in the top 10 list for both states.

Final Reflections

When you have completed the project, please reflect on some of the longer-lasting lessons of this experience. Most teams will make some discoveries or gain key insights either about SQL, SQL shortcuts, or the nature of data structures. Perhaps you noticed something important about this particular business. Write a thoughtful paragraph describing the team's most noteworthy and valuable discovery or insight.

We realize the joins and subqueries would probably be two of the most powerful commands in SQL, which allow us to explore a broader range of attributes within a dataset and generate output based on restricted ranges or values of data. Also, to generate desired output, it's essential to understand the relationship between different entities. It is also really important to gain a deep understanding of the dataset before executing any queries. Because attribute names and entity names themselves are sometime not self-explanatory and users may have different interpretations of what information each entity contains before obtaining a in-depth understanding, and thus may construct queries in a way that could potentially lead to a misleading conclusion. We also found that before arriving at any conclusion, we need to look at data from different perspectives. For example, in query 4 we found that the top 3 categories that are earning most revenue are different from categories that have most of transactions, and categories that have higher total dollar sales may be categories that carry more distinct products. It is too early to jump into any conclusion that Sam's club should consider increasing the availability of products under these categories to increase revenue.