Data Quality Assessment

To ensure that our data is accurate, and that we are providing accurate results needed to derive valuable insights, we must perform a data quality assessment.

Entity Integrity Assessment

We start by performing an Entity Integrity Assessment. This is important because we need to ensure that all primary keys are unique and there are not any null values.

For the Store_Information Table,I used the Store_Nbr as the primary key. I was able to confirm that Store_Nbr meets Referential integrity because there are no null values and only unique values. To learn how this was confirmed, please refer to Table 1 Queries 1 and 2.

For the Member_Index Table ,I used the Membership_Nbr as the primary key. I was able to confirm that Membership_Nbr meets Referential integrity because there are no null values and only unique values. To learn how this was confirmed, please refer to Table 1 Queries 3 and 4.

For the Store_Visits Table, I used the Membership_Nbr as the primary key. I was able to confirm that Membership_Nbr meets Referential integrity because there are no null values and only unique values. To learn how this was confirmed, please refer to Table 1 Queries 5 and 6.

Functional Dependency Assessment

To ensure that the relationship between attributes is consistent and logical, I performed functional dependency assessments.

For the Store_Information table, It is normal to expect that each store has 1 manager, and 1 phone number. I was able to confirm this by using Queries 7 and 8 from Table 1. In the Member_Index table, it is reasonable to assume that each member should have 1 join date. This can be confirmed by Query 9 in Table 1.Lastly, for the Store_Visits table, it is fair to assume that each Visit_Nbr should have only one transaction date. We were able to verify this with Query 10 in Table 2.

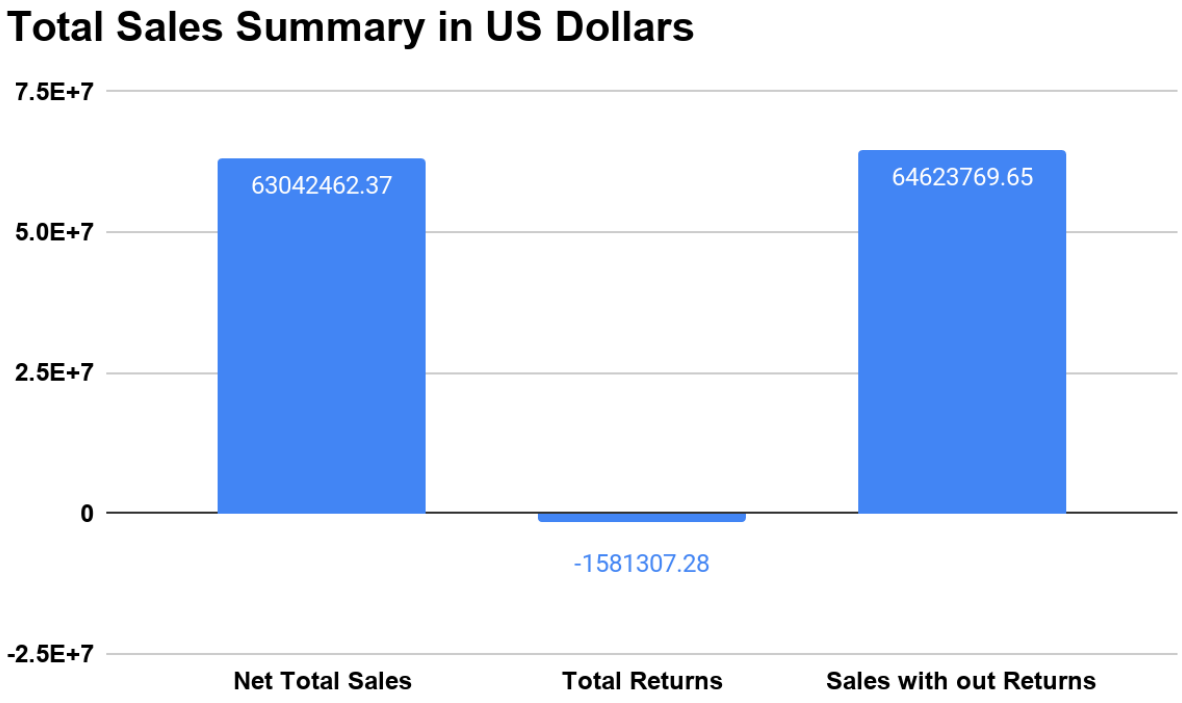
Referential Integrity of Member_Index Table vs STORE_VISITS

Now we will access the referential integrity between the Store_Visits vs the Member_Index Table. To perform this, we needed to ensure that the Membership_Nbr values were consistent across both tables. When running Query 12 in Table 2 we realized there were 193898 members with transactions who aren't in the Member_Index table. With that said, we currently don't have referential integrity. To fix this, I am updating the membership_nbr values to 999999 in the Membership_NBR table so that I can filter them out when performing an analysis. This was completed by using Query 13 in Table 2. To ensure that 999999 shows up in the Member_Index table, I am using Query 14 in Table 1. Having 999999 is important because it will allow me to properly join (inner join on Membership_NBR) the Member_Index Table and the Store_Visits table. Furthermore, to ensure that 999999 shows up in the Member_Index table, I am using the Query 15 in Table 1. Having 999999 is important because it will allow me to properly join (inner join on Membership_NBR) the Member_Index Table and the Store_Visits table. It will also allow me to filter out the members who aren't being evaluated. After completing this, we now have Referential Integrity between the Member_Index table and the Store_Visits table.

Data Analysis and Review Report

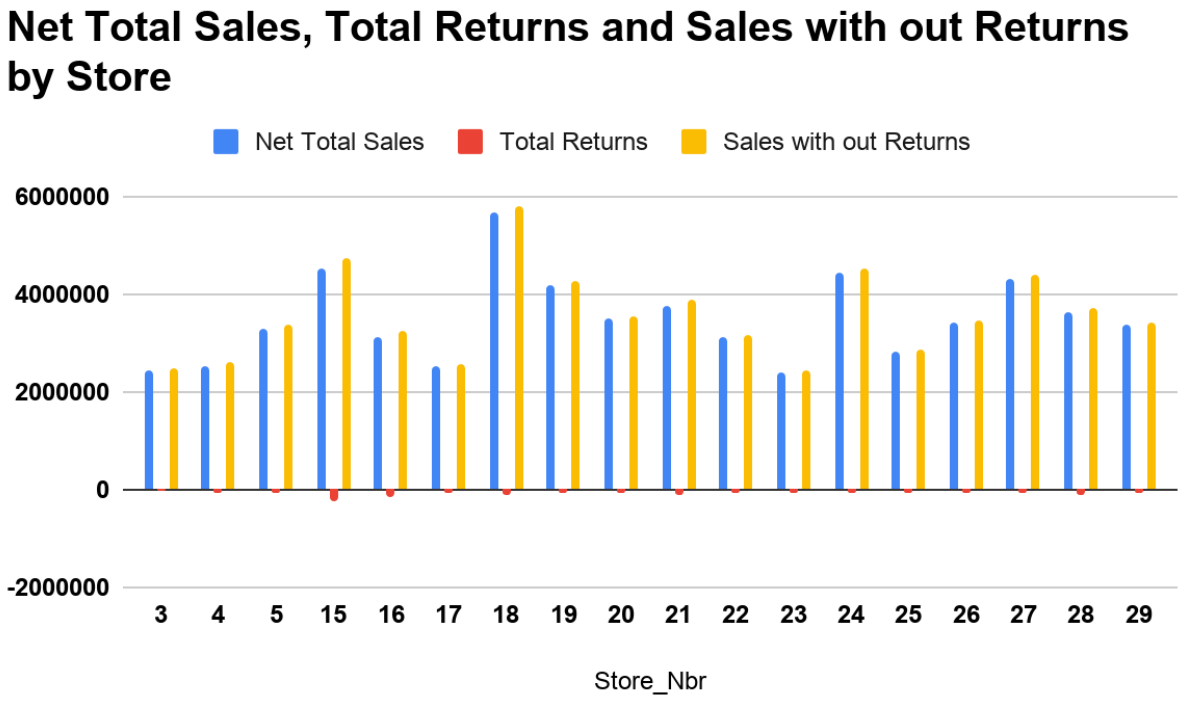
Summary of Total Sales

This is a visualization of total sales. To provide greater insight, I decided to break down the sales by Net Total Sales (including returns), Total Returns, and Sales without Returns to show how much returns are impacting Sales. A small percentage (2.44%) of Sales without Returns resulted in Returns. Given the generous return policy, this is a good performance for Sams Club. This shows that customers are satisfied with their purchases, and that we are doing a good job of marketing and advertising the value of our products. For information on how Net Total Sales, Total Returns, and Sales without Returns are calculated, please refer to Table 2 Queries 1,2, and 3 respectively in the appendix.



Summary of Total Sales by Store

This is a visualization and table of sales by store. To provide greater insight, I decided to break down the sales by Net Total Sales (including returns), Total Returns, and Sales without Returns to show how much returns are impacting Sales. As expected, the stores will have different Sales amounts. This can be attributed to being in different regions. What is alarming, however, is the percentage of Sales without Returns which resulted in Returns for stores 15 (5%) and 16 and (4.22%). With both of these values being significantly larger than the average return rate, it would be nice to know why this is happening. To further understand this, it would be helpful to have a breakdown of the items returned at these stores, and compare that breakdown to other stores.For information on how Net Total Sales, Total Returns, and Sales without Returns are calculated, please refer to Table 2 Queries 4,5, and 6 respectively in the appendix.



Store_Nbr	Net Total Sales	Total Returns	Sales with out Returns	Return Percentage
3	2440226.33	-46187.38	2486413.71	1.857590304
4	2543273.58	-55612.72	2598886.3	2.139867373
5	3302177.61	-79701.13	3381878.74	2.356711642
15	4518633.53	-237992.11	4756625.64	5.003381136

16	3129524.81	-137964.12	3267488.93	4.222328612
17	2523136.09	-50511.62	2573647.71	1.962647017
18	5667744.88	-124245.16	5791990.04	2.145120401
19	4204541.96	-77218.31	4281760.27	1.803424413
20	3486504.83	-71644.74	3558149.57	2.013539302
21	3744885.03	-122209.47	3867094.5	3.160240072
22	3127580.55	-56493.8	3184074.35	1.774261333
23	2398179.99	-52374.86	2450554.85	2.137265363
24	4451885.44	-78230.18	4530115.62	1.726891465
25	2811157.05	-64403.17	2875560.22	2.23967384
26	3405765.8	-74926.78	3480692.58	2.152639978
27	4306308.71	-76564.16	4382872.87	1.746894383
28	3617621.19	-112496.31	3730117.5	3.015891859
29	3363314.99	-62531.26	3425846.25	1.825279228

Summary of Total Sales by Day of Week

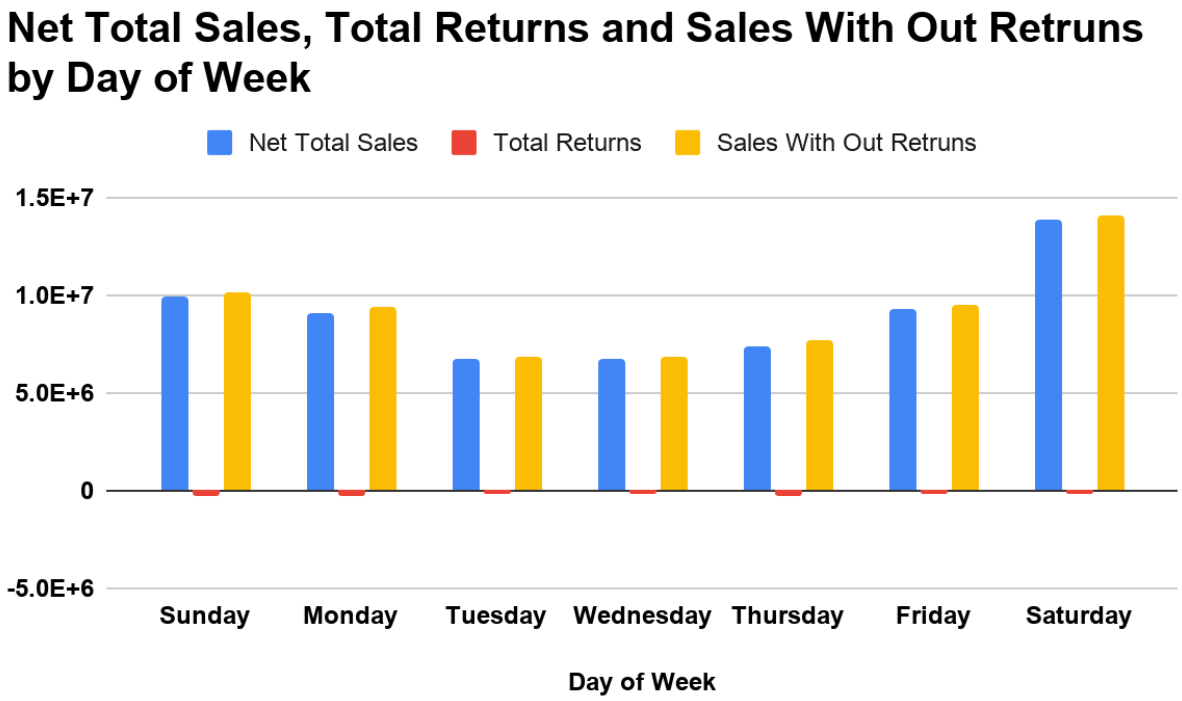
This is a visualization of total sales by day of the week. To provide greater insight, I decided to break down the sales by Net Total Sales (including returns), Total Returns, and Sales without Returns to show how much returns are impacting Sales.

When analyzing this, we see that Saturday has the highest Net Total Sales, and Sales With Out Returns. This is expected behavior since Saturday is when most individuals have the time to go to Same Club. To help our lower performing days, I would recommend offering more promotional deals on those days (Tuesday, Wednesday, and Thursday). This will increase the Sales for those days since consumers will be more incentivised to shop on those days.

While analyzing returns, we notice that from a volume standpoint, Saturday has the highest return amount of $14068346.75. However, Thursday has the highest return percentage of 3.5%. To learn more about why Thursday has a higher return percentage, it would be helpful to know which items are being returned on Thursday, and compare that to the other days of the week.

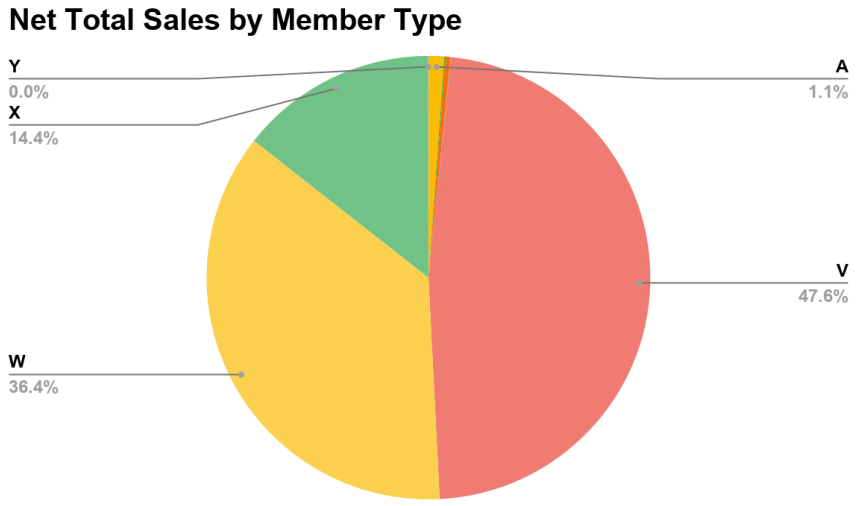
For information on how Net Total Sales, Total Returns, and Sales without Returns are calculated, please refer to Table 2 Queries 7,8, and 9 respectively in the appendix. Please note that the return percentage was calculated in Google Sheets.

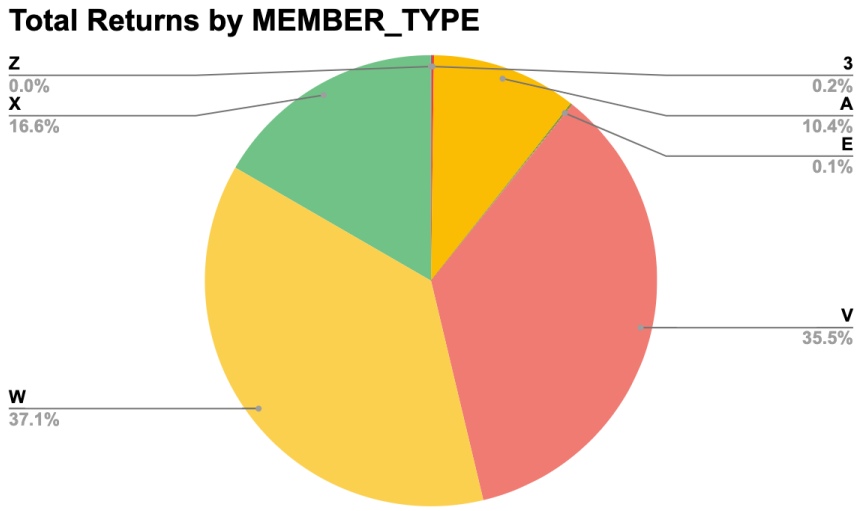
Day of Week	Net Total Sales	Total Returns	Sales With Out Retruns	Return Percentage
Friday	9327127.05	-178346.33	9505473.38	1.876248798
Monday	9128413.18	-321509.41	9449922.59	3.402243848
Saturday	13847146.63	-221200.12	14068346.75	1.572324907
Sunday	9928700.74	-223595.48	10152296.22	2.202412884
Thursday	7394345.12	-271488.54	7665833.66	3.541539669
Tuesday	6704945.05	-186188.5	6891133.55	2.701855923
Wednesday	6711784.6	-178978.9	6890763.5	2.597374007



Summary of Total Sales by Membership Type

This is a visualization of total sales by day of the Membership Type. Based on this, the majority of Sales are from Members_Type V, W, and X with V being the leader. When analyzing the returns, Member Type A consists of a significantly higher percentage of returns (10.4%) despite only accounting for 1.1% of net total sales. To obtain further insight, it would be helpful to know what each Member_Type means.





MEMBER_TYPE	Net Total Sales	Total Returns	Sales With Out Returns
1	11344.31	-453.93	11798.24
3	10517.25	-2828.79	13346.04

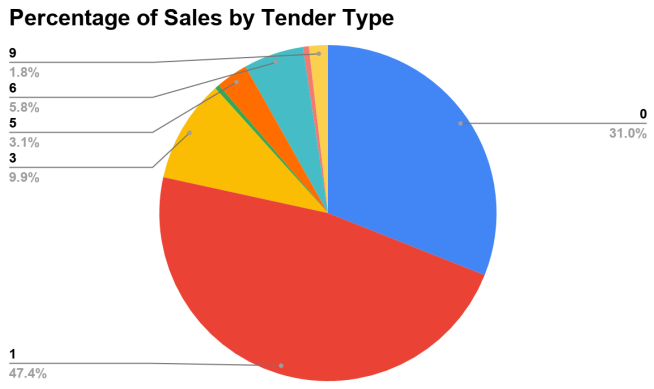
A	702949.68	-164509.82	867459.5
D	56599.15	-946.25	57545.4
E	222714.53	-994.94	223709.47
G	13486.64	-322.64	13809.28
H	1953.13	0	1953.13
V	29990444.56	-561796.56	30552241.12
W	22936850.65	-586319.05	23523169.7
X	9071849.81	-263060.44	9334910.25
Y	14781.15	36.66	14744.49
Z	8971.51	-111.52	9083.03

For information on how Net Total Sales, Total Returns, and Sales without Returns are calculated, please refer to Table 2 Queries 10,11, and 12 respectively in the appendix.

Tender Type Analysis

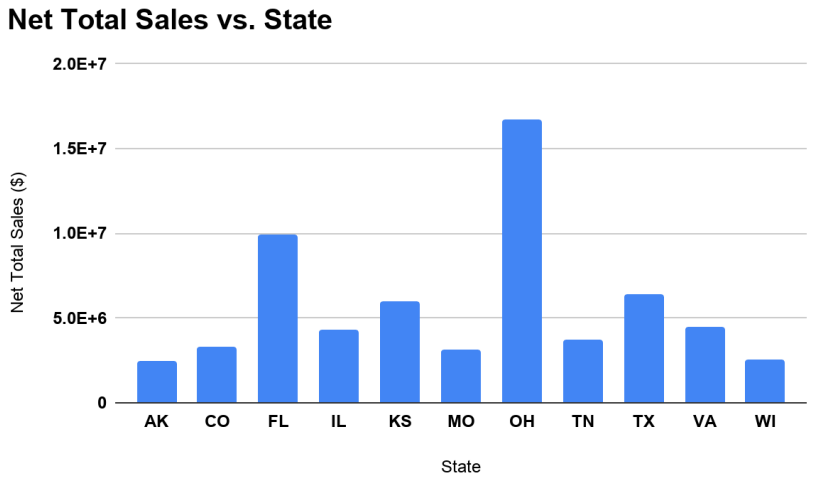
This is a visualization of total sales by day of the Tender Type. Based on this, the top 3 tender types are 0, 1, and 3 which are respectively cash, check, and Discover credit card. Candidly speaking, I am surprised by the percentage of people who pay with check and cash. This is an inefficient method that slows down the checkout process. To improve operational efficiency, Sam's Club should look into promoting/incentivising more digital options. This will discourage customers from paying with cash and check.

For information on how this was calculated, please refer to Table 2 Query 13 in the appendix.

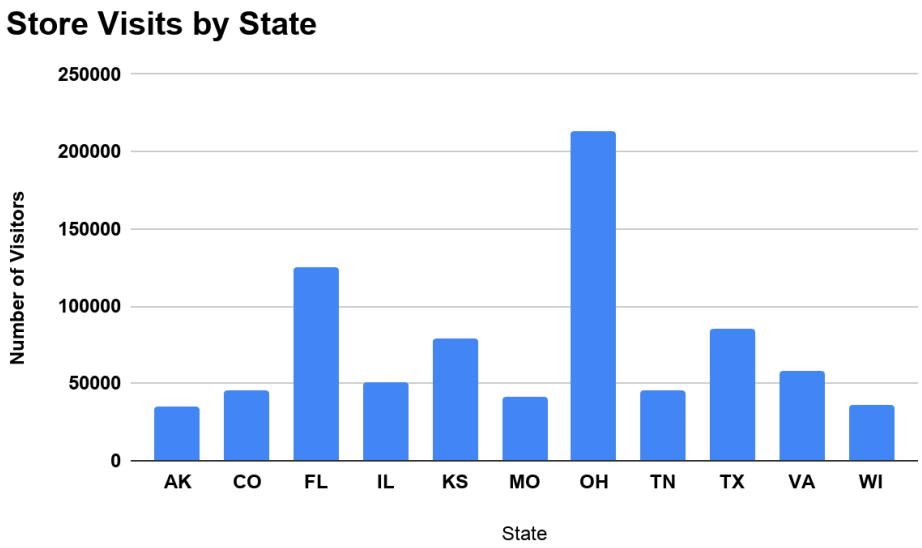


State Analysis

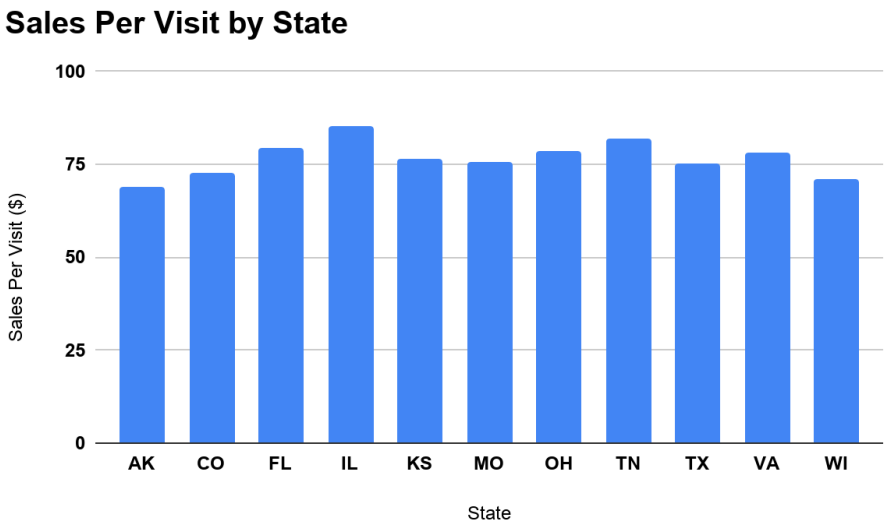
This is a visualization of total sales (including returns) by state. Based on this, Ohio and Florida are the top two performing states.



To dig a little deeper, I looked at the number of store visits for each state. Again, Ohio and Florida were the top two performers.



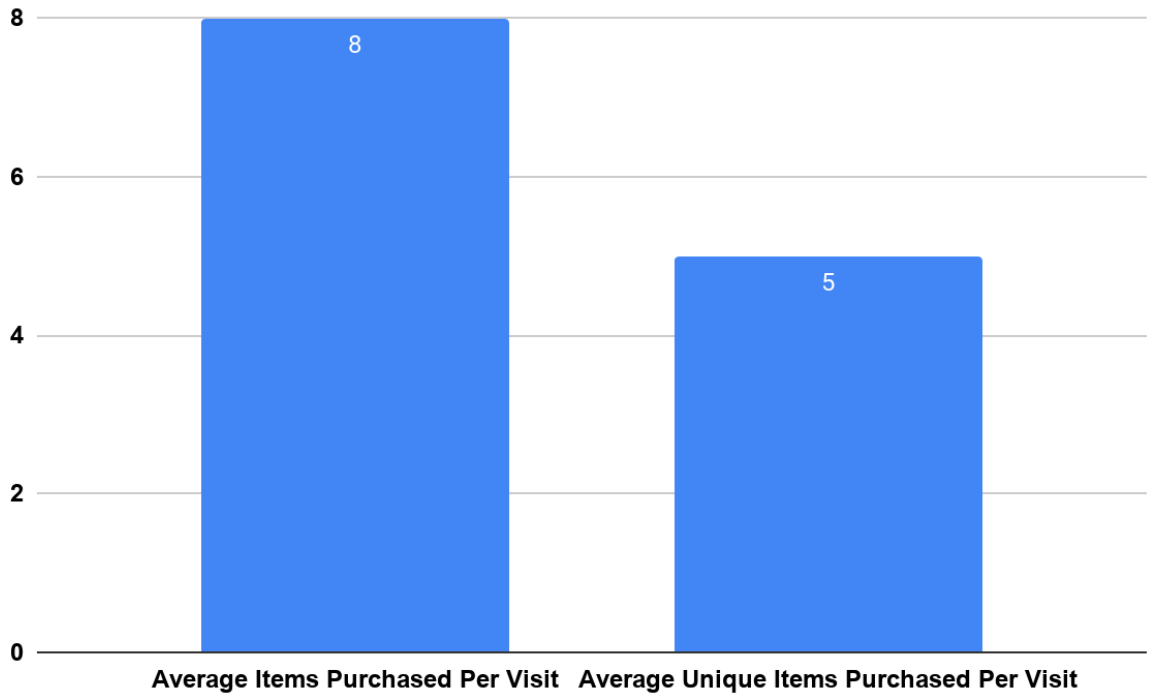
Lastly, I wanted to see how much a consumer was spending per visit by state. When analyzing this, I notice that Illinois has the highest spending per visit. As for why this is happening, this could be due to the city of Chicago where individuals have more money to spend, or this could be due to additional promotional deals/certain items which incline consumers to spend more per visit.



For information on how this is calculated, please refer to Table 2 Query 15 in the appendix. Please note that Sales Per Visit was calculated in Google Sheets.

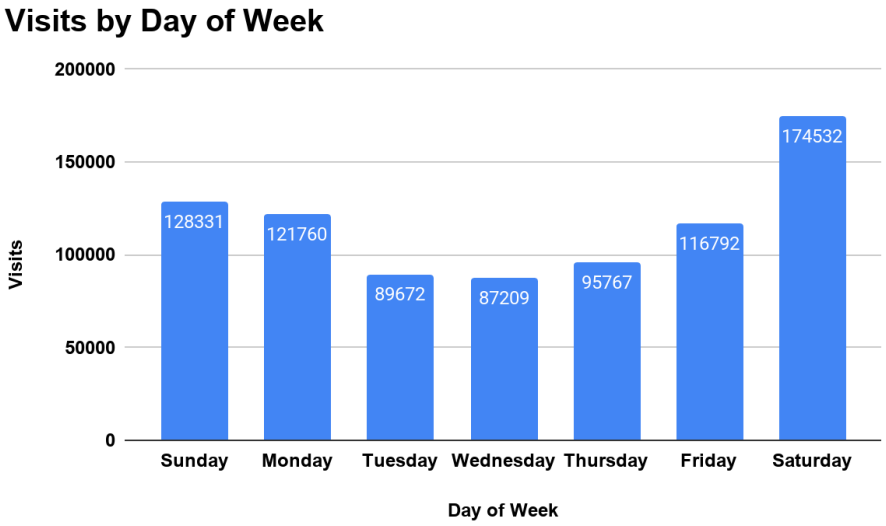
Summary of Purchase Patterns

Based on our data, the average number of items purchased per visit is 8, and the average unique items purchased per visit is 5. As for determining if this is an acceptable result, I would need to know what Sam's Club's goal is, and it would be helpful to know how many items are purchased per visit at other competitors (Walmart, Costco, Target, etc). To learn how this was calculated, please refer to Table 2 Query 16.

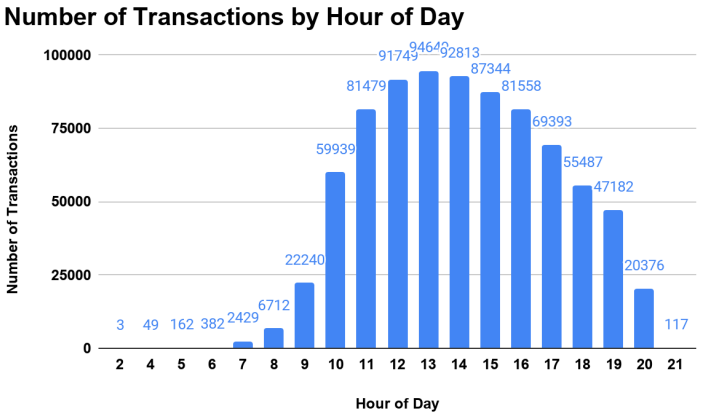


Summary of Member Visit Breakdown

Below is a visualization of the number of visits by the day of the week. As we can see, Saturday attracted the most visitors. Sunday and Monday are a close second and third respectively. With lesser traffic coming on Tuesday - Thursday, Sam's Club should consider offering more promotional deals on those days of the week so that customers are more incentivised to shop on Tuesday-Thursday. For information on how this was calculated, please refer to Table 2 Query 17.



Below is a visualization of the number of transactions broken down by the hour of the day. As we can see, stores are busiest during hours 12-14. The biggest jump in transaction by hours is from the 9th hour to the 10th hour. When viewing this, Sam's Club should use this as a guide for managers to ensure that hours 12-14 are fully staffed because those are peak demand hours. With lesser traction from visitors coming in during the early hours (10th hour and before) Sam's Club should look into offering early morning promotional deals which incentivise shoppers to come in earlier. For a full understanding of how this has been calculated, please refer to Table Query 18.



Characteristics of Member with High Visits

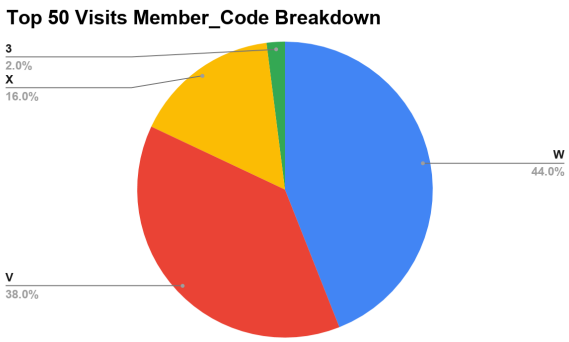
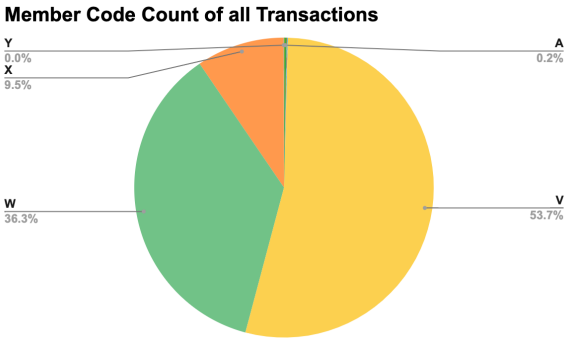
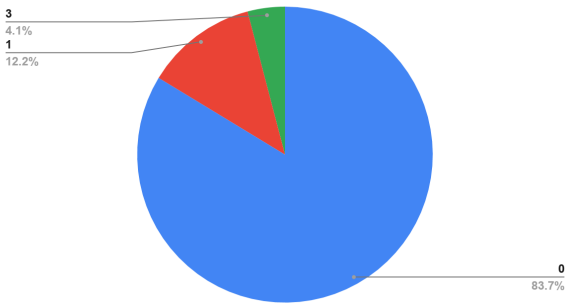
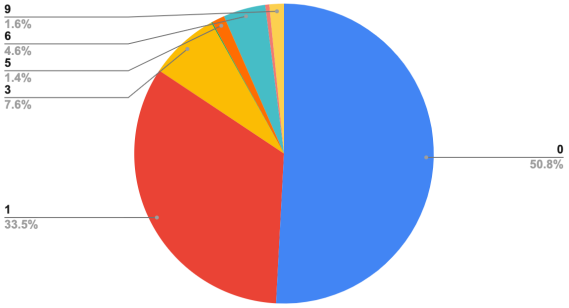
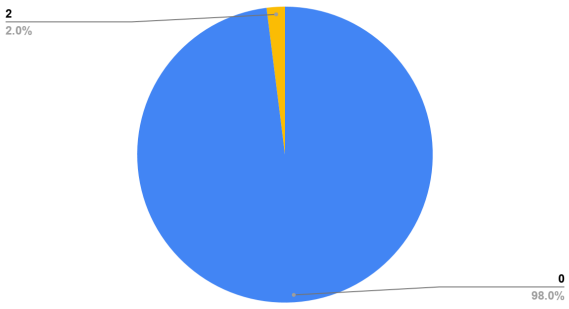
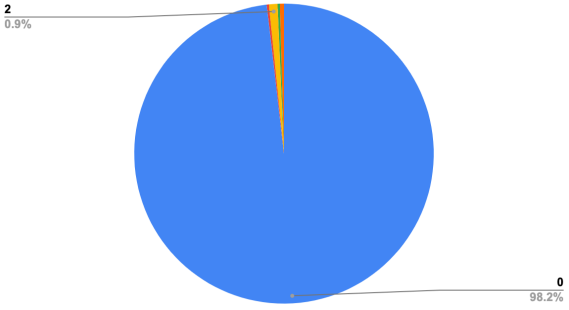
To understand the characteristics of members with high visits, I am comparing Member_Code, Tender_Type, Elite_Stat_Code, and Bus_Cr_Typ_Stat_CD of the members with the top 50 visits versus all members.

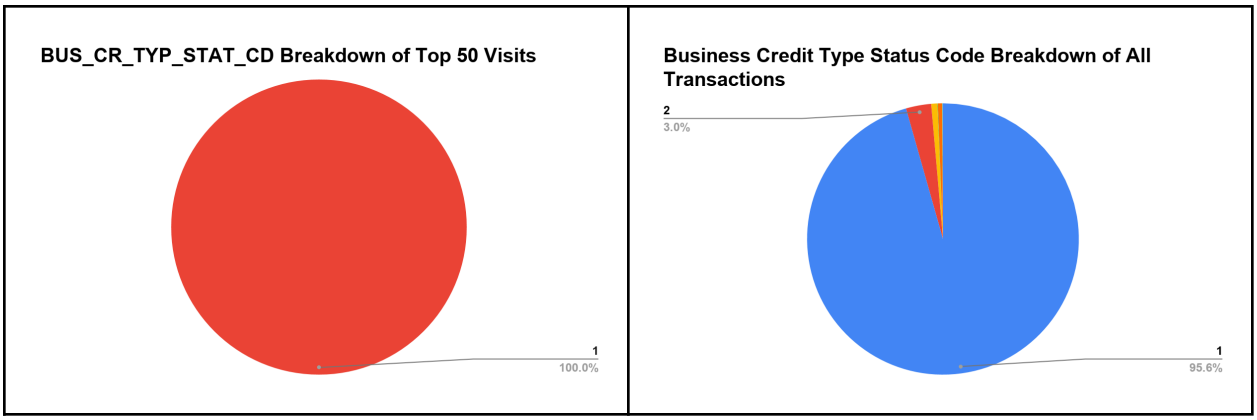
When comparing the Member_Code breakdown, 44% of customers in the Top 50 visits fall under Member_Code W despite making up only 36.3% of all transactions. 16% of customers in the Top 50 visits fall under Member_Code X despite making up only 9.5% of all transactions.Perhaps customers with Member_Code X and W are more likely to be frequent shoppers at Sam's Club.

The Tender_Type distribution of the Top 50 visitors versus all customers is quite different from the Tender_Type distribution of all transactions. Top 50 visitors have a stronger inclination towards using cash (83.7%) versus all members (50.8%). That being said, members who visit Sam’s Club more often are more likely to pay with cash.

Given the almost identical distributions of the top 50 visitors versus all visitors when comparing Elite_Stat_Code and Bus_Cr_Typ_Stat_CD, it is safe to say that Elite_Stat_Code and Bus_Cr_Typ_Stat_CD do not have a significant influence on the frequency of a customer’s visits. Please note that even though 100% of Bus_Cr_Typ_Stat_CD = 1, 95% of all visitors had Bus_Cr_Typ_Stat_CD = 1. The 5% discrepancy is likely due to the much larger pool of customers that are being included.

To learn about how Member_Code, Tender_Type, Elite_Stat_Code, and Bus_Cr_Typ_Stat_CD was calculated for all customers, please refer to Table 2 Queries 22,21, 23, and 24 respectively.

Top 50 Visits	All Customers
	
	
	



Characteristics of Member with High Sales

To understand the characteristics of members with high sales, I am comparing Member_Code, Tender_Type, Elite_Stat_Code, and Bus_Cr_Typ_Stat_CD of the members with the top 50 sales versus the sales of all members.

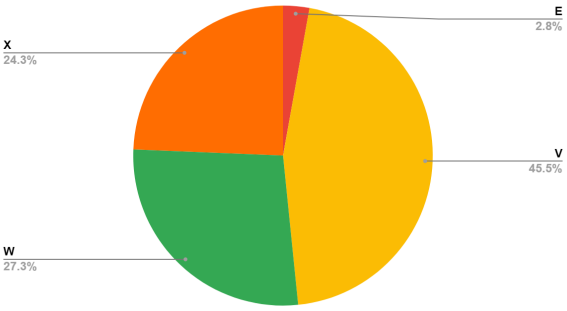
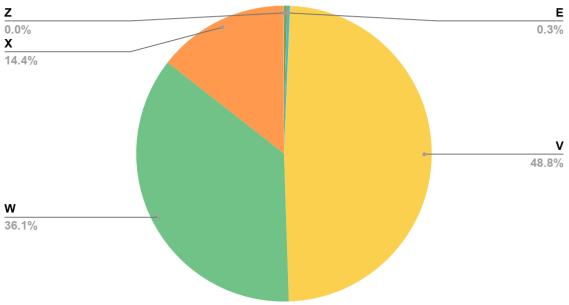
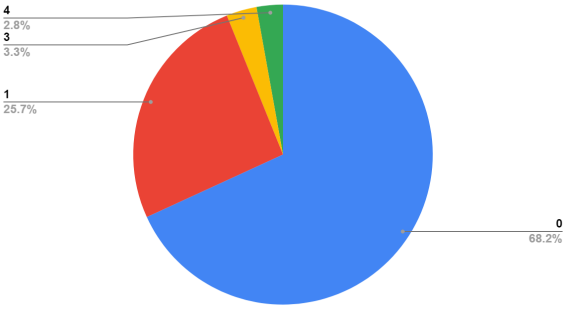
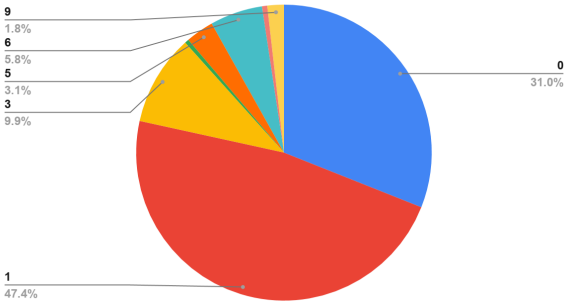
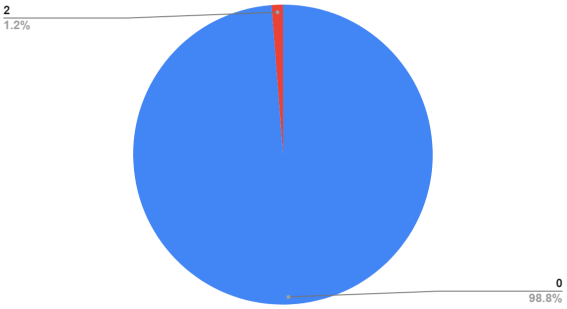
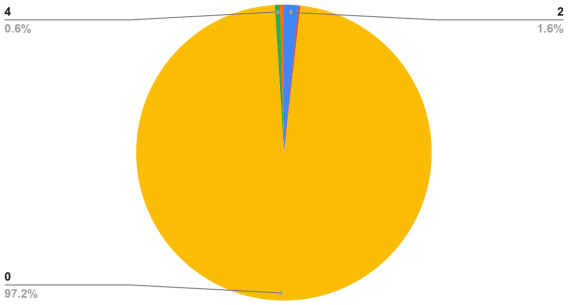
When comparing the Member_Code breakdown, 24.3% of sales from the top 50 sales customers are from Member_ Code X despite making up only 14.4% of all sales. Perhaps customers with Member_Code X are more likely to be frequent purchasers at Sam's Club.

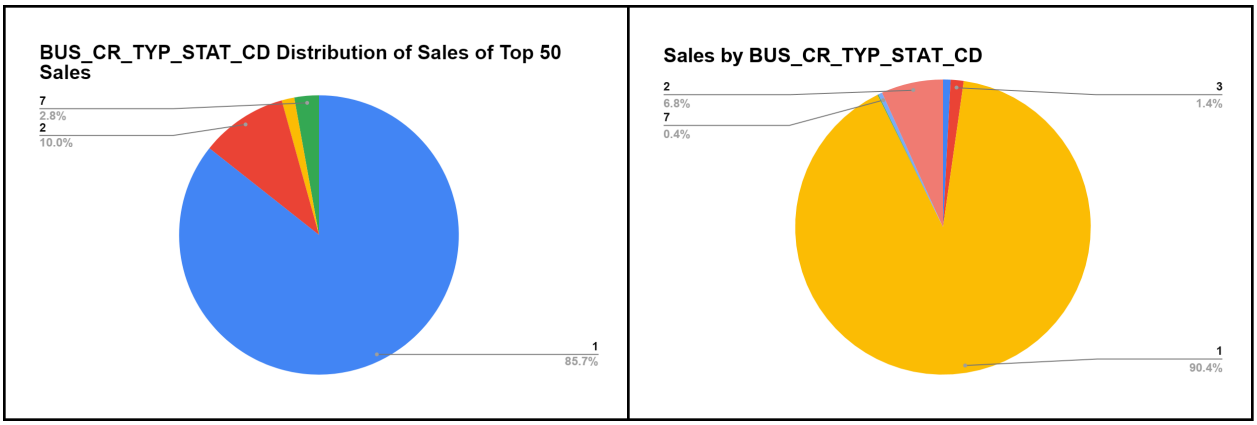
The Tender_Type distribution of the Top 50 purchasers versus all customers is quite different. Tender Type 0 and 1 make up 68.2% and 25.7% of sales in the top 50 sales despite accounting for 31% and 47% of all sales respectively. That being said top 50 purchasers are more likely to favor tender type 0 and less likely to favor tender type 1.

Given the almost identical distributions of the top 50 sales versus all sales when comparing Elite_Stat_Code, it is safe to say that Elite_Stat_Code does not have a significant influence on the sales from a customer.

Lastly, when comparing Bus_Cr_Typ_Stat_CD the distributions are almost identical. In the top 50 sales, however, slightly more sales were generated from Bus_Cr_Typ_Stat_CD (10% versus 6.8%).

To learn more about how Member_Code, Tender_Type, Elite_Stat_Code, and Bus_Cr_Typ_Stat_CD is calculated for the top 50 sales, please refer to Table 2 Query 20. To learn how Member_Code, Tender_Type, Elite_Stat_Code, and Bus_Cr_Typ_Stat_CD is calculated for all sales, please refer to Table 2 Query 25,24,26, and 27 respectively.

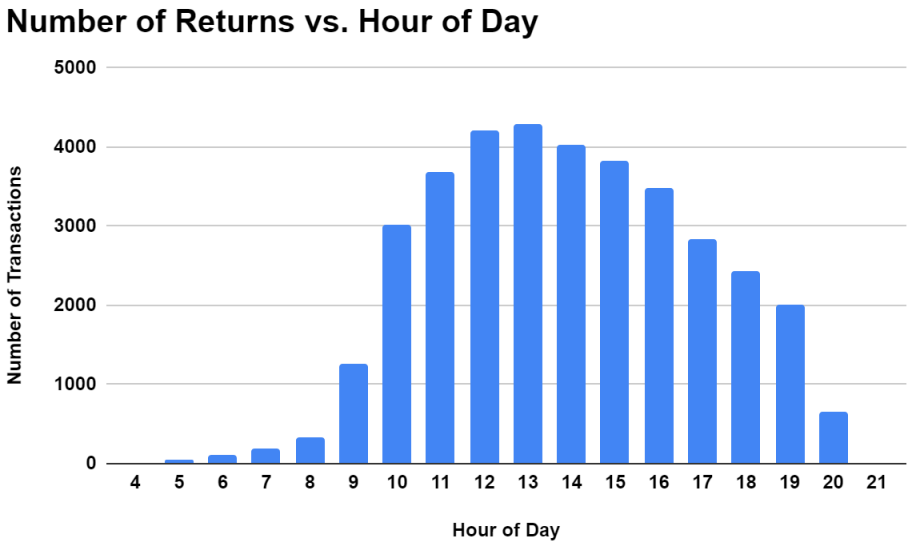
Top 50 Sales	All Customers
Member Code Distribution by Sales of Top 50 Sales 	**Sales by Member Code** 
Tender Type Distribution by Sales of Top 50 Sales 	**Sales by Tender Type** 
Elite_Stat_Code Distribution of Sales of Top 50 Sales 	**Sales by Elite Stat Code** 



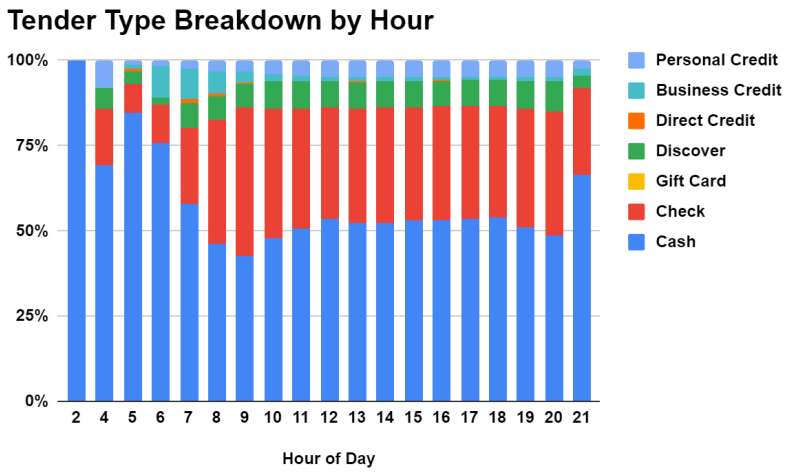
Additional Purchasing Insights

For additional analysis, I am analyzing the returns by the hour of the day as well as the tender type breakdown by the hour of the day.

Below is a chart that shows the number of returns by the hour of the day. During the early hours, the return volume is slow, but there is a major spike from the 9th to the 10th hour. The number of returns increases till the thirteenth hour of the day, and then they slowly decline. This pattern is very similar to the number of transactions by the hour of the day. To learn how this was calculated, please refer to Table 2 Query 30.



Below is a figure that shows the tender type by the hour of the day. When analyzing this visualization, I notice that the usage of checks increases from the 6th hour to the 9th hour, and gradually decreases till the 12th hour where the usage almost plateaus. While this is happening, I noticed that the usage of cash is most affected. The percentage of cash transactions decreases from the 6th hour to the 9th hour, and gradually increases till the 12th hour where the usage almost plateaus. As for why this is happening, my theory is that with checks taking longer to write/process, those who prefer using checks shop during earlier hours when there are less customers. Please refer to Table 2 Query 31 to learn how this was calculated.



Appendix

Table 1	
Query Number	Query
Query 1	select* from Store_Information where Store_Nbr is null
Query 2	select store_nbr, count(*) from Store_Information group by store_nbr having count(*) > 1
Query 3	select* from Member_Index where MEMBERSHIP_NBR is null
Query 4	select Membership_nbr, count(distinct Membership_nbr) from MEMBER_INDEX group by Membership_nbr having count(distinct Membership_nbr) > 1
Query 5	select * from STORE_VISITS where VISIT_NBR is null or MEMBERSHIP_NBR is null
Query 6	select Membership_nbr, count(distinct Membership_nbr) from Store_VISITS group by Membership_nbr having count(distinct Membership_nbr) > 1
Query 7	- Store_Nbr vs Phone_Nbr -- select Store_Nbr, count(distinct Phone_Nbr) from STORE_INFORMATION group by Store_Nbr having count(distinct Phone_Nbr) > 1
Query 8	select Store_Nbr, count(distinct Manager_Name)

	from STORE_INFORMATION group by Store_Nbr having count(distinct Manager_Name) > 1
Query 9	select MEMBERSHIP_NBR, count(distinct JOIN_DATE) from MEMBER_INDEX group by MEMBERSHIP_NBR having count(distinct JOIN_DATE) > 1
Query 10	select VISIT_NBR, count(distinct TRANSACTION_DATE) from STORE_VISITS group by VISIT_NBR having count(distinct TRANSACTION_DATE) > 1
Query 11	select Store_Nbr from Store_Visits where Store_Nbr not in (select Store_Nbr from STORE_INFORMATION)
Query 12	select Membership_NBR from Store_Visits where Membership_NBR not in (select Membership_Nbr from Member_Index)
Query 13	update STORE_VISITS set Membership_Nbr = 999999 where Membership_NBR not in (select Membership_Nbr from Member_Index)
Query 14	update MEMBER_INDEX set MEMBERSHIP_NBR = 999999 where MEMBERSHIP_NBR not in (select MEMBERSHIP_NBR from STORE_VISITS)

Table 2

Query Number	Query
Query 1	select sum(Total_Visit_Amt) as 'Net Total Sales' from STORE_VISITS where MEMBERSHIP_NBR != 999999

Query 2	select sum(Total_Visit_Amt) as 'Total Returns' from STORE_VISITS where Membership_NBR != 999999 and Refund_Code != 0
Query 3	select sum(Total_Visit_Amt) as 'Sales with out Returns' from STORE_VISITS where Membership_NBR != 999999 and Refund_Code != 1
Query 4	select Store_Nbr, sum(Total_Visit_Amt) as 'Net Total Sales' from STORE_VISITS where MEMBERSHIP_NBR != 999999 group by Store_Nbr order by Store_Nbr Asc
Query 5	select Store_Nbr, sum(Total_Visit_Amt) as 'Total Returns' from STORE_VISITS where Membership_NBR != 999999 and Refund_Code != 0 group by Store_Nbr order by Store_Nbr Asc
Query 6	select store_Nbr, sum(Total_Visit_Amt) as 'Sales with out Returns' from STORE_VISITS where Membership_NBR != 999999 and Refund_Code != 1 group by Store_Nbr order by Store_Nbr Asc
Query 7	select DATENAME(weekday, Transaction_Date) as "Day of Week", sum(Total_Visit_Amt) as "Net Total Sales" from store_visits where MEMBERSHIP_NBR != 999999 group by DATENAME(weekday, Transaction_Date) order by 'Day of Week' asc
Query 8	select DATENAME(weekday, Transaction_Date) as "Day of Week", sum(Total_Visit_Amt) as "Total Returns" from store_visits where Membership_NBR != 999999 and Refund_Code != 0 group by DATENAME(weekday, Transaction_Date) order by 'Day of Week' asc
Query 9	select DATENAME(weekday, Transaction_Date) as "Day of Week", sum(Total_Visit_Amt) as "Sales With Out Returns" from store_visits where Membership_NBR != 999999 and Refund_Code != 1 group by DATENAME(weekday, Transaction_Date) order by 'Day of Week' asc
Query 10	select MEMBER_TYPE, sum(Total_Visit_Amt) as "Net Total Sales" from store_visits join MEMBER_INDEX on STORE_VISITS.Membership_Nbr = MEMBER_INDEX.MEMBERSHIP_NBR where Member_Index.MEMBERSHIP_NBR != 999999 group by MEMBER_TYPE

	order by MEMBER_TYPE asc
Query 11	select MEMBER_TYPE, sum(Total_Visit_Amt) as "Total Returns" from store_visits join MEMBER_INDEX on STORE_VISITS.Membership_Nbr = MEMBER_INDEX.MEMBERSHIP_NBR where Member_Index.MEMBERSHIP_NBR != 999999 and Refund_Code != 0 group by MEMBER_TYPE order by MEMBER_TYPE asc
Query 12	from store_visits join MEMBER_INDEX on STORE_VISITS.Membership_Nbr = MEMBER_INDEX.MEMBERSHIP_NBR where MEMBER_INDEX.MEMBERSHIP_NBR != 999999 and Refund_Code != 1 group by MEMBER_TYPE order by MEMBER_TYPE asc
Query 13	select Tender_Type, sum(Total_Visit_Amt) as 'Net Total Sales' from STORE_VISITS where MEMBERSHIP_NBR != 999999 group by Tender_Type order by 'Tender_Type' asc
Query 14	select STORE_VISITS.Store_Nbr, Visit_Nbr, Membership_NBR, Tender_Type, Total_Visit_Amt, Transaction_Date, City, STORE_INFORMATION.State, Zip_Code, Manager_Name from STORE_INFORMATION join Store_Visits on STORE_INFORMATION.Store_Nbr = STORE_VISITS.Store_Nbr where MEMBERSHIP_NBR != 999999
Query 15	select State, sum(Total_Visit_Amt) as ' Net Total Sales' , count(Visit_Nbr) as 'Store Visits' from STORE_INFORMATION join Store_Visits on STORE_INFORMATION.Store_Nbr = STORE_VISITS.Store_Nbr where MEMBERSHIP_NBR != 999999 group by State
Query 16	select avg(Tot_Unique_Itm_Cnt) as "Average Unique Items Purchased Per Visit", avg(Tot_Scan_Cnt) as "Average Items Purchased Per Visit" from STORE_VISITS where Membership_Nbr != 999999
Query 17	select DATENAME(weekday, Transaction_Date) as "Day of Week", count(Visit_Nbr) as "Visits" from store_visits where Membership_NBR != 999999 group by DATENAME(weekday, Transaction_Date)
Query 18	select * from (select count(left(Transaction_Time,1)) as 'Number of Transactions' ,

	cast(left(Transaction_Time,1) as int) as 'Hour of Day', sum(Total_Visit_Amt) as 'Net Total Sales', avg(Tot_Unique_Itm_Cnt) as "Average Unique Items Purchased Per Visit", avg(Tot_Scan_Cnt) as "Average Items Purchased Per Visit" from Store_Visits where LEN(Transaction_Time) = 7 and Membership_NBR != 999999 group by left(Transaction_Time, 1) union select count(left(Transaction_Time,2)) as 'Number of Transactions', cast(left(Transaction_Time,2) as int) as 'Hour of Day', sum(Total_Visit_Amt) as 'Net Total Sales', avg(Tot_Unique_Itm_Cnt) as "Average Unique Items Purchased Per Visit", avg(Tot_Scan_Cnt) as "Average Items Purchased Per Visit" from Store_Visits where LEN(Transaction_Time) = 8 and Membership_NBR != 999999 group by left(Transaction_Time, 2)) as TimeTable order by 'Hour of Day' asc
Query 19	select top 50 store_visits.Membership_Nbr, sum(Total_Visit_Amt) as 'Net Total Sales', Member_Code, Tender_Type, MEMBER_TYPE, ZIP_CODE, ELITE_STAT_CODE, BUS_CR_TYP_STAT_CD, datediff(MONTH, JOIN_DATE, LAST_RENEWAL_DATE) as "Months from join to renewal" from store_visits join MEMBER_INDEX on STORE_VISITS.Membership_Nbr = MEMBER_INDEX.MEMBERSHIP_NBR where store_visits.Membership_NBR != 999999 group by store_visits.Membership_Nbr, Member_Code, Tender_Type,MEMBER_TYPE, ZIP_CODE, ELITE_STAT_CODE, BUS_CR_TYP_STAT_CD, JOIN_DATE, LAST_RENEWAL_DATE order by 'Net Total Sales' desc
Query 20	select top 50 store_visits.Membership_Nbr, count(Visit_Nbr) as 'Total Visits', Member_Code, Tender_Type, MEMBER_TYPE, ZIP_CODE, ELITE_STAT_CODE, BUS_CR_TYP_STAT_CD, datediff(MONTH, JOIN_DATE, LAST_RENEWAL_DATE) as "Months from join to renewal" from store_visits join MEMBER_INDEX on STORE_VISITS.Membership_Nbr = MEMBER_INDEX.MEMBERSHIP_NBR where store_visits.Membership_NBR != 999999 group by store_visits.Membership_Nbr, Member_Code, Tender_Type,MEMBER_TYPE, ZIP_CODE, ELITE_STAT_CODE, BUS_CR_TYP_STAT_CD, JOIN_DATE, LAST_RENEWAL_DATE order by 'Total Visits' desc
Query 21	select Tender_Type, count(Tender_Type) as "Tender Count" from STORE_VISITS where store_visits.Membership_NBR != 999999 group by Tender_Type
Query 22	select Member_Code, count(Member_Code) as "Member Code Count" from STORE_VISITS where store_visits.Membership_NBR != 999999

	group by Member_Code
Query 23	select Elite_Stat_Code, count(Elite_Stat_Code) as "Elite Stat Code Count" from MEMBER_INDEX where MEMBER_INDEX.Membership_NBR != 999999 group by Elite_Stat_Code
Query 24	select BUS_CR_TYP_STAT_CD, count(BUS_CR_TYP_STAT_CD) as "BUS_CR_TYP_STAT_CD Count" from MEMBER_INDEX where MEMBER_INDEX.Membership_NBR != 999999 group by BUS_CR_TYP_STAT_CD
Query 25	select Tender_Type, sum(Total_Visit_Amt) as "Sales by Tender Type" from STORE_VISITS where store_visits.Membership_NBR != 999999 group by Tender_Type
Query 26	select Member_Code, sum(Total_Visit_Amt) as "Sales by Member Code" from STORE_VISITS where store_visits.Membership_NBR != 999999 group by Member_Code
Query 27	select Elite_Stat_Code, sum(Total_Visit_Amt) as "Sales by Elite Stat Code" from MEMBER_INDEX join STORE_VISITS on STORE_VISITS.Membership_Nbr =MEMBER_INDEX.MEMBERSHIP_NBR where MEMBER_INDEX.Membership_NBR != 999999 group by Elite_Stat_Code
Query 28	select BUS_CR_TYP_STAT_CD, sum(Total_Visit_Amt) as "Sales by BUS_CR_TYP_STAT_CD" from MEMBER_INDEX join STORE_VISITS on STORE_VISITS.Membership_Nbr =MEMBER_INDEX.MEMBERSHIP_NBR where MEMBER_INDEX.Membership_NBR != 999999 group by BUS_CR_TYP_STAT_CD
Query 29	select avg(datediff(MONTH, JOIN_DATE, LAST_RENEWAL_DATE)) as " Average Months from join to renewal" from MEMBER_INDEX where MEMBER_INDEX.Membership_NBR != 999999
Query 30	select * from (select count(left(Transaction_Time,1)) as 'Number of Transactions' , cast(left(Transaction_Time,1) as int) as 'Hour of Day', sum(Total_Visit_Amt) as 'Net Total Sales', avg(Tot_Unique_Itm_Cnt) as "Average Unique Items Purchased Per Visit", avg(Tot_Scan_Cnt) as "Average Items Scanned Per Visit" from Store_Visits where LEN(Transaction_Time) = 7 and Membership_NBR != 999999 and Refund_Code =1

	group by left(Transaction_Time, 1) union select count(left(Transaction_Time,2)) as 'Number of Transactions', cast(left(Transaction_Time,2) as int) as 'Hour of Day', sum(Total_Visit_Amt) as 'Returns', avg(Tot_Unique_Itm_Cnt) as "Average Unique Items Returned Per Return-Visit", avg(Tot_Scan_Cnt) as "Average Items Returned Per Return-Visit" from Store_Visits where LEN(Transaction_Time) = 8 and Membership_NBR != 999999 and Refund_Code =1 group by left(Transaction_Time, 2)) as TimeTable order by 'Hour of Day' asc
Query 31	select * from(select cast(left(Transaction_Time,1) as int) as 'Hour of Day' , count(case when tender_type = 0 then 1 end) as 'Cash', count(case when tender_type = 1 then 1 end) as 'Check', count(case when tender_type = 2 then 1 end) as 'Gift Card', count(case when tender_type = 3 then 1 end) as 'Discover', count(case when tender_type = 4 then 1 end) as 'Direct Credit', count(case when tender_type = 5 then 1 end) as 'Business Credit', count(case when tender_type = 6 then 1 end) as 'Personal Credit' from Store_Visits where LEN(Transaction_Time) = 7 and Membership_NBR != 999999 group by left(Transaction_Time, 1) union select cast(left(Transaction_Time,2) as int) as 'Hour of Day' , count(case when tender_type = 0 then 1 end)/count(tender_type) as 'Cash', count(case when tender_type = 1 then 1 end) as 'Check', count(case when tender_type = 2 then 1 end) as 'Gift Card', count(case when tender_type = 3 then 1 end) as 'Discover', count(case when tender_type = 4 then 1 end) as 'Direct Credit', count(case when tender_type = 5 then 1 end) as 'Business Credit', count(case when tender_type = 6 then 1 end) as 'Personal Credit' from Store_Visits where LEN(Transaction_Time) = 8 and Membership_NBR != 999999 group by left(Transaction_Time, 2)) as TimeTable2 order by 'Hour of Day' asc