Myka Miller

Final Project Report

Data Quality Assessment

Using the STORE_VISITS, MEMBER_INDEX, and STORE_INFORMATION tables, assessments were run for entity integrity and referential integrity. The layout of this report will be the conclusions of the Entity Integrity Assessment for each table followed by the Referential Integrity Assessment for each relationship (STORE_VISITS to MEMBER_INDEX and STORE_VISITS to STORE_INFORMATION).

The SQL code in Appendix A was used on the STORE_VISITS table to determine entity integrity. The first section of code demonstrates that each visit number (VISIT_NBR) is only associated with one record—every record has a unique visit number. The following two sections of code are to identify any null or empty entries. Since the visit number is a unique identifier and has no null or blank values in the table, the STORE_VISITS table has entity integrity.

Appendix B shows the code used with the STORE_INFORMATION table to determine entity integrity. The first section of code demonstrates that each store number (STORE_NBR) is only associated with one record—every record has a unique store number. The following two sections of code are to identify any null or empty entries. Since the store number is a unique identifier and has no null or blank values in the table, the STORE_INFORMATION table has entity integrity.

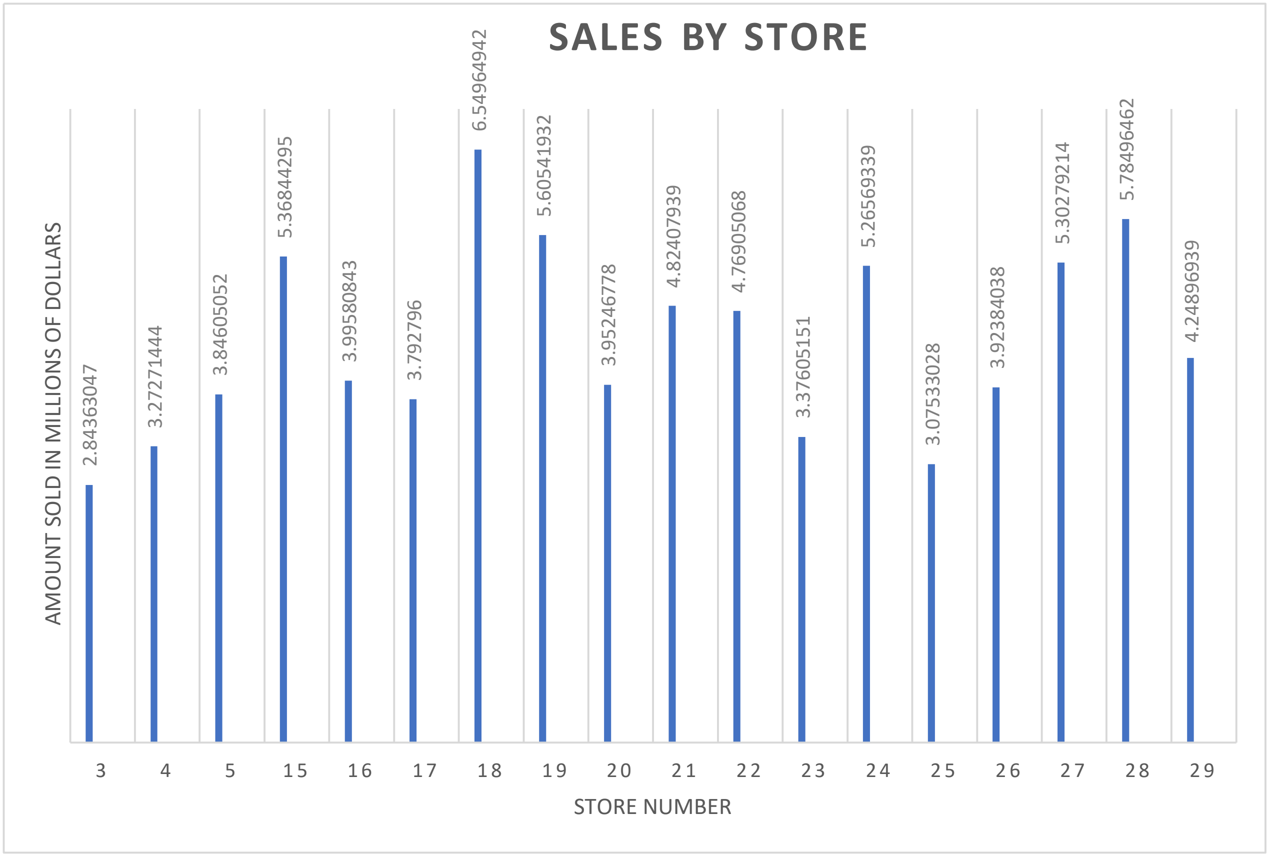
The code in Appendix C was used for the MEMBER_INDEX table to determine entity integrity. The first section of code demonstrates that each Membership Number (MEMBERSHIP_NBR) is only associated with one record—every record has a unique membership number. The following two sections of code are to identify any null or empty entries. Since the membership number is a unique identifier and has no null or blank values in the table, the MEMBER_INDEX table has entity integrity.

The referential integrity between STORE_VISITS and MEMBER_INDEX was performed using the code in Appendix D. The results showed that there were records in STORE_VISITS with a Membership Number (Membership_Nbr) that did not appear in the MEMBER_INDEX table. In order to fix the referential integrity issues, the code in Appendix E shows the addition of a dummy variable that was used to correct the records in STORE_VISITS that did not reference a record in the MEMBER_INDEX table. There were approximately 94,000 records that would need to be altered. Due to the lack of information, these records would not be able to be properly corrected. The dummy variable allows for these records to be easily identified for later use or correction if more information becomes available.

A similar code was used to check for referential integrity of STORE_VISITS and STORE_INFORMATION as seen in Appendix F. This showed that there were no instances of a Store_Nbr in STORE_VISITS not referencing a STORE_NBR in STORE_INFORMATION which shows referential integrity.

Store Sales

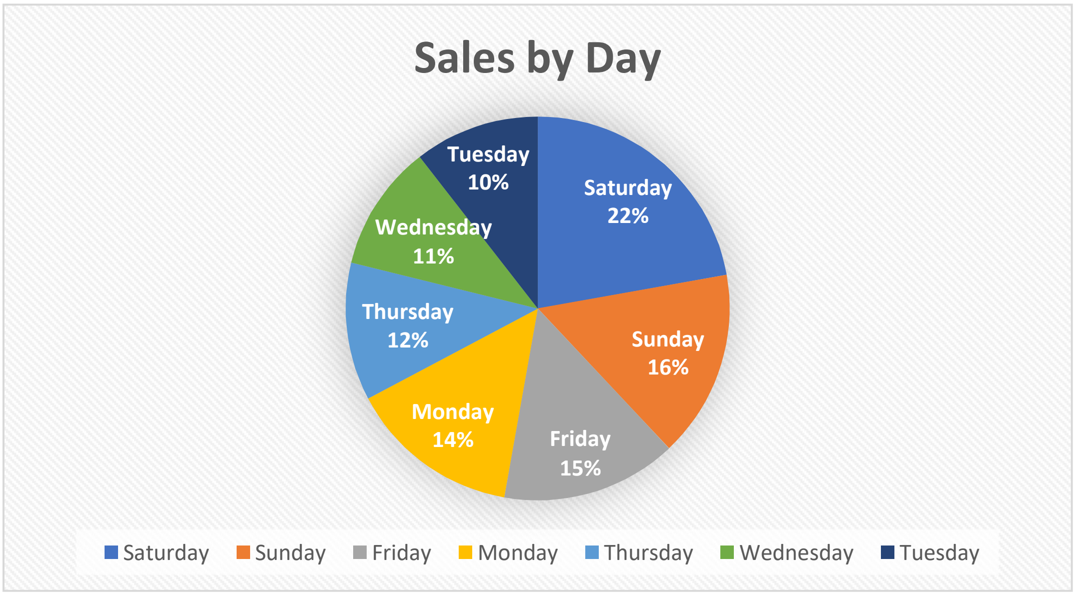
The total sales for Sam's Club during this 30-day period was $79,797,751.11 – this was based under the assumption that if the Total_Visit_Amt was positive it was a purchase and if negative it was a return. When broken down by stores, the top performers are stores 18, 28, and 19. All the stores compared can be seen below by using the code in appendix G.



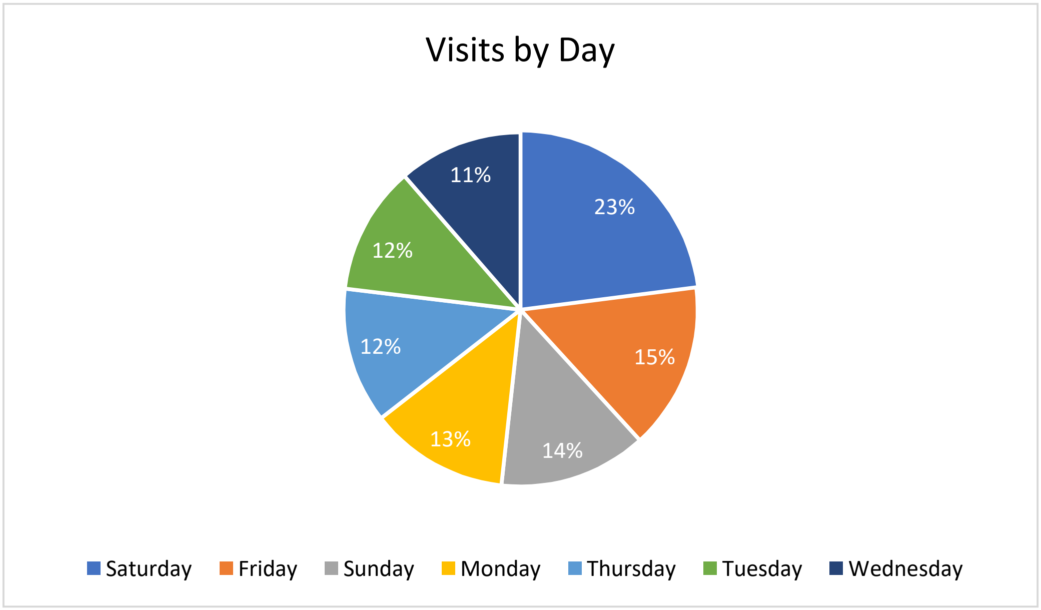
Of the top 5 performing stores, all of them were coded open on Sundays and two were listed under Region 23. It may be beneficial to speak with Region 23's manager to identify any distinguishing characteristics leading to high success. On the opposite end, of the lowest 5, two stores were in Region 38 (Appendix P). Some investigations and comparisons between the two regions would have potential benefits.

By Day Sales and Visits

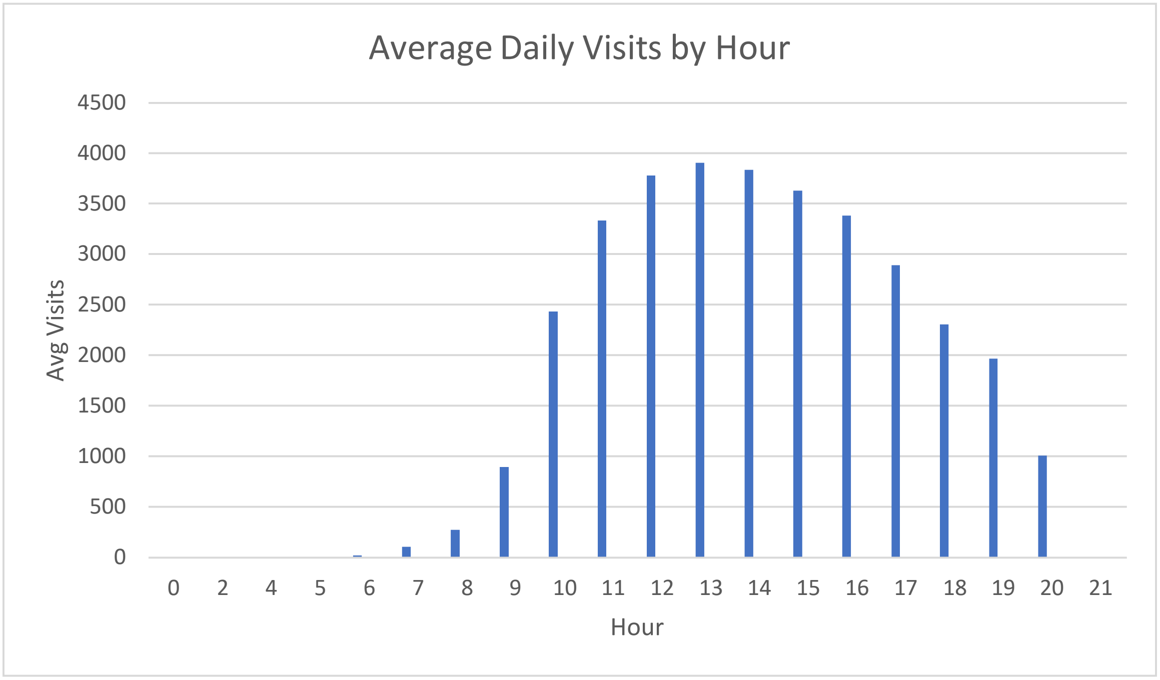
The chart below (using appendix H) shows that weekend sales make up over half of the sales for the sample period with Saturday making up nearly a fourth of the sales.



Similarly, the Visits per Day data (appendix I) also identified the weekends as the highest vist days.



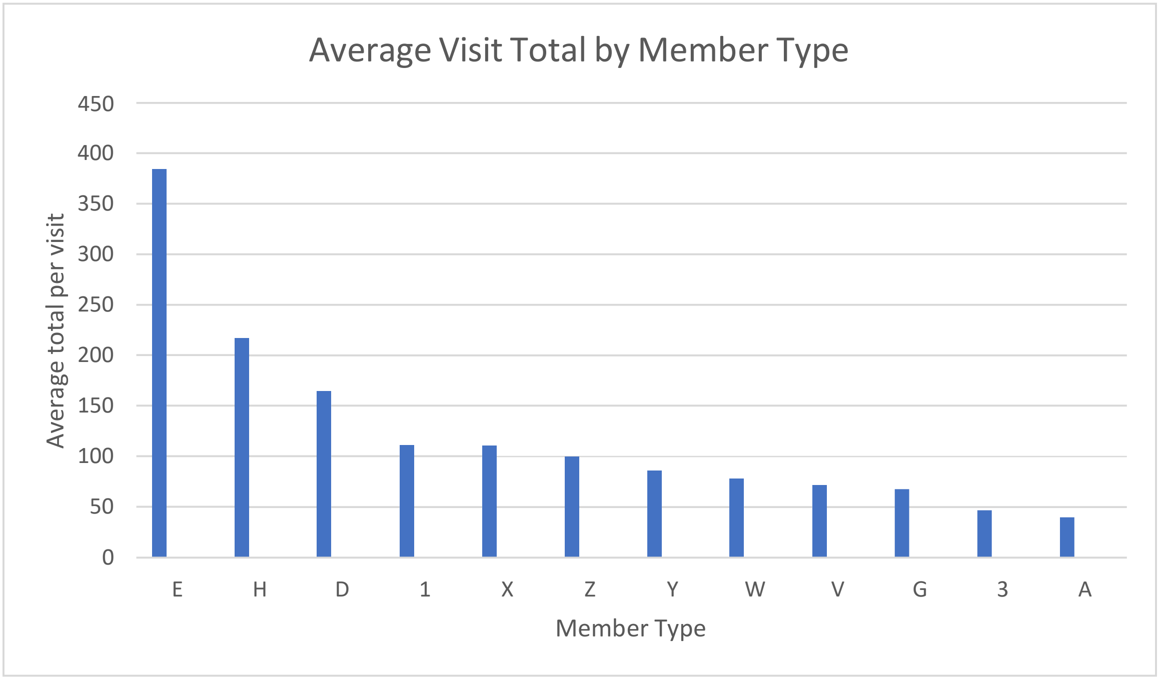
The visits per day data can be broken down further into hours by using the code inappendix J and finding that 11:00 to 16:00 are, on average, the peak times.

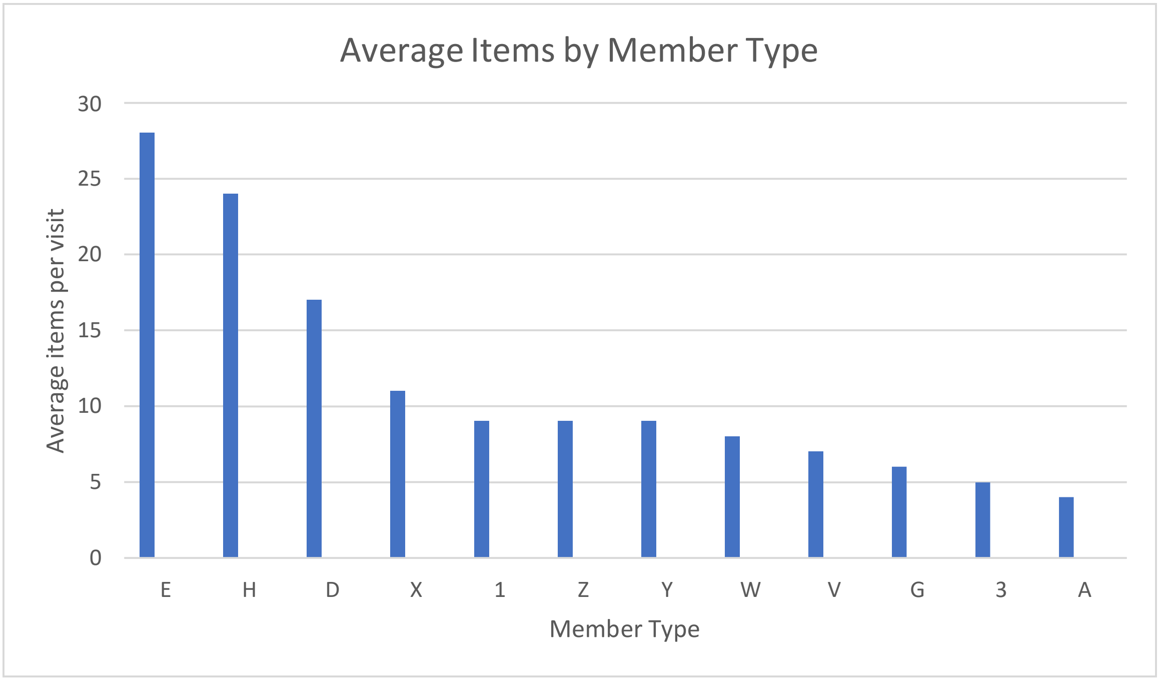


Buying Habits by Member Types

As a reminder, a dummy variable was put in, so when comparing member types and their sales, the member type " – " will be excluded. The member types with the highest total sales were types V and W (Appendix K)—however when looking closer this category has a few large spends that skew the outputs. To address this, I recommend using the average spends and visits to better understand the spending and visiting patterns and potentially doing further cleaning of the data for quality.

Based on the sample data, members typically purchase an average of 8 items with an average total cost of $79.17 (Appendix L). We can break this down further by finding the averages across the member types as in Appendix M and resulting in the following information





Despite member types V and W having the most total spends, types E, H, and D lead in overall behavior as far as spending and items purchased.

When looking further into the high spending types of members we find that there is not a pattern—or more recent data is needed—to spending and the average length of memberships for the member types seen in Appendix N. The main characteristic of the high spenders and visitors is that their member status code is A (active) which is to be expected. However, based on the results from Appendix O, 8% of our members in big spender groups like E, H, and D are not active anymore. This could be as simple of a fix as reaching out and giving these customers an incentive to reactivate their accounts.

Appendix A

Entity Integrity of STORE_VISITS

```
select Visit_Nbr, count(*)
from Store_Visits
group by Visit_Nbr
having count(*) > 1;
       --No records: Visit Number is a unique identifier
select *
from store_visits
where Visit_Nbr is NULL;
       --no NULLS
select *
from store_visits
where len(Visit_Nbr) = 0;
       --No empty Visit Numbers
       --Visit_Nbr = PK for Store_Visits
```

Appendix B

Entity Integrity of STORE_INFORMATION

```
select Store_Nbr, count(*)
from Store_Information
group by Store_Nbr
having count(*) > 1;
       --No records: Store_Nbr is a unique identifier

select *
from Store_Information
where Store_Nbr is NULL;
       --no NULLS
select *
from Store_Information
where len(Store_Nbr) = 0;
       --No empty Store_Nbr
       --Str_Nbr = PK for Store_Information
```

Appendix C

Entity Integrity of MEMBER_INDEX

```
select Membership_Nbr, count(*)
from Member_Index
group by Membership_Nbr
having count(*) > 1;
       --No records: Membership_Nbr is a unique identifier

select *
from Member_Index
where Membership_Nbr is NULL;
       --no NULLS

select *
from Member_Index
where len(Membership_Nbr) = 0;
       --No empty Membership_Nbr
       --Membership_Nbr = PK for Member_Index
```

Appendix D

Referential Integrity Assessment for STORE_VISITS and MEMBER_INDEX

```
select Distinct Membership_Nbr
from Store_Visits
where Membership_Nbr not in (select Membership_Nbr from Member_Index);
--Use dummy variable: Membership_Nbr = 0000
```

Appendix E

Dummy Variable to Fix Referential Integrity Issue

```
insert into Member_Index values (0000, '-', '-', '-', '-', '-', 00-00-0000, 00-00-0000,
'255', '255', '255' , '-', '255', 00-00-0000);

update Store_Visits
set Membership_Nbr = 0000
where Membership_Nbr not in (select Membership_Nbr from Member_Index);

select Distinct Membership_Nbr
from Store_Visits
where Membership_Nbr not in (select Membership_Nbr from Member_Index);
```

Appendix F

Referential Integrity Assessment of STORE_VISITS and STORE_INFORMATION

```
select Distinct Store_Nbr
from Store_Visits
where Store_Nbr not in (select Store_Nbr from Store_Information);
--NO instances: has referential integrity
```

Appendix G

Total Sales and Sales by Store

```
--total sales
select sum(Total_Visit_Amt)
from STORE_VISITS

--Sales by store greatest to least
select Store_Nbr, sum(Total_Visit_Amt) as [Total Sales]
from STORE_VISITS
group by Store_Nbr
order by sum(Total_Visit_Amt) desc;
```

Appendix H

Sales by Day

```
--Sales by day greatest to least
select DateName(Weekday, Transaction_Date) as Day, sum(Total_Visit_Amt) as [Total Sales]
from STORE_VISITS
group by DateName(Weekday, Transaction_Date)
order by sum(Total_Visit_Amt) desc;
```

Appendix I

Visits by Day

```
--Visits by day
Select datename(Weekday, Transaction_Date) as Day, avg(cnt) as Visits
from (
       select count(Visit_Nbr) as cnt,Transaction_Date
       from STORE_VISITS
       group by Transaction_Date
       )T
group by datename(Weekday, Transaction_Date)
order by avg(cnt) desc;
```

Appendix J

Average Visits by Hour

```
--visits by hour
       --add Hour column
alter table store_visits
Add Hour int
       --set Hour column by case to split Transaction_Time
update store_Visits
set Hour = (case
       when Transaction_Time between 0 and 1000000 then 0
       when Transaction_Time between 1000000 and 2000000 then 1
       when Transaction_Time between 2000000 and 3000000 then 2
       when Transaction_Time between 3000000 and 4000000 then 3
       when Transaction_Time between 4000000 and 5000000 then 4
       when Transaction_Time between 5000000 and 6000000 then 5
       when Transaction_Time between 6000000 and 7000000 then 6
       when Transaction_Time between 7000000 and 8000000 then 7
       when Transaction_Time between 8000000 and 9000000 then 8
       when Transaction_Time between 9000000 and 10000000 then 9
       when Transaction_Time between 10000000  and 11000000 then 10
       when Transaction_Time between 11000000  and 12000000 then 11
       when Transaction_Time between 12000000  and 13000000 then 12
       when Transaction_Time between 13000000  and 14000000 then 13
       when Transaction_Time between 14000000  and 15000000 then 14
       when Transaction_Time between 15000000  and 16000000 then 15
       when Transaction_Time between 16000000  and 17000000 then 16
       when Transaction_Time between 17000000  and 18000000 then 17
       when Transaction_Time between 18000000  and 19000000 then 18
       when Transaction_Time between 19000000  and 20000000 then 19
       when Transaction_Time between 20000000  and 21000000 then 20
       when Transaction_Time between 21000000  and 22000000 then 21
       when Transaction_Time between 22000000  and 23000000 then 22
       when Transaction_Time between 23000000  and 24000000 then 23
              end);
       --find avg. visits by Hour
 select C.Hour, avg(cnt) as avg_visits
 from STORE_VISITS V, (
       select count_big(Visit_Nbr) as cnt, Transaction_Date, Hour
       from STORE_VISITS
       group by Transaction_Date, Hour
       ) C
```

Appendix K

Top Total Spends by Member Type

```
--top total spends
select Member_Type, sum(Total_Visit_Amt) as [Total Sales]
from STORE_VISITS v
       left join MEMBER_INDEX m
       on m.MEMBERSHIP_NBR = v.Membership_Nbr
group by Member_Type
order by sum(Total_Visit_Amt) desc;
```

Appendix L

Average Spending and Items Purchased

```
--AVG SPEND
select avg(total_visit_amt)
from STORE_VISITS;

--AVG ITEMS
 select avg(Tot_Scan_Cnt)
 from STORE_VISITS;
```

Appendix M

Average Spending and Items Purchased by Member Type

```
--AVG SPEND BY MEMBER TYPES

select Member_Type, avg(Total_Visit_Amt) as [Avg Sales]
from STORE_VISITS v
left join MEMBER_INDEX m
on m.MEMBERSHIP_NBR = v.Membership_Nbr
where Member_Type != '-'
group by Member_Type
order by avg(Total_Visit_Amt) desc


--AVG ITEMS BY MEMBER TYPES

select Member_Type, avg(Tot_Scan_Cnt) as [Avg Items]
from STORE_VISITS v
left join MEMBER_INDEX m
on m.MEMBERSHIP_NBR = v.Membership_Nbr
where Member_Type != '-'
group by Member_Type
order by avg(Tot_Scan_Cnt) desc
```

Appendix N

Membership Lengths by Member Types

```
--Avg membership length
select Member_Type,
       avg((datediff(day, Join_Date, Getdate())/365.25)) as [Average Membership]
from MEMBER_INDEX
group by Member_Type
order by avg((datediff(day, Join_Date, Getdate())/365.25));
```

Appendix O

Member Status of Top 50 Visitors

```
 select Member_Status_CD, count(Membership_Nbr) as [Number of Members]
 from MEMBER_INDEX
 where  Membership_Nbr in (
	select top (50) Membership_Nbr as Top_Mbr
	from STORE_VISITS
	group by Membership_Nbr
	having count(Visit_Nbr) < 70
	order by count(Visit_Nbr) desc
	)
group by Member_Status_CD
order by count(Membership_Nbr) desc;
```

Appendix P

Qualities of top and low stores

```
--Qualities of top stores
select Store_Nbr, Region_Nbr, District_Nbr, Open_Sunday_Flag
from STORE_INFORMATION
where Store_Nbr in (
	select top 5 Store_Nbr
	from STORE_VISITS
	group by Store_Nbr
	order by sum(Total_Visit_Amt) desc
	)
order by Store_Nbr;


--Qualities of bottom stores
select Store_Nbr, Region_Nbr, District_Nbr, Open_Sunday_Flag
from STORE_INFORMATION
where Store_Nbr in (
	select top 5 Store_Nbr
	from STORE_VISITS
	group by Store_Nbr
	order by sum(Total_Visit_Amt)
	)
order by Store_Nbr
```