Insurance Data Management and Quality Assessment

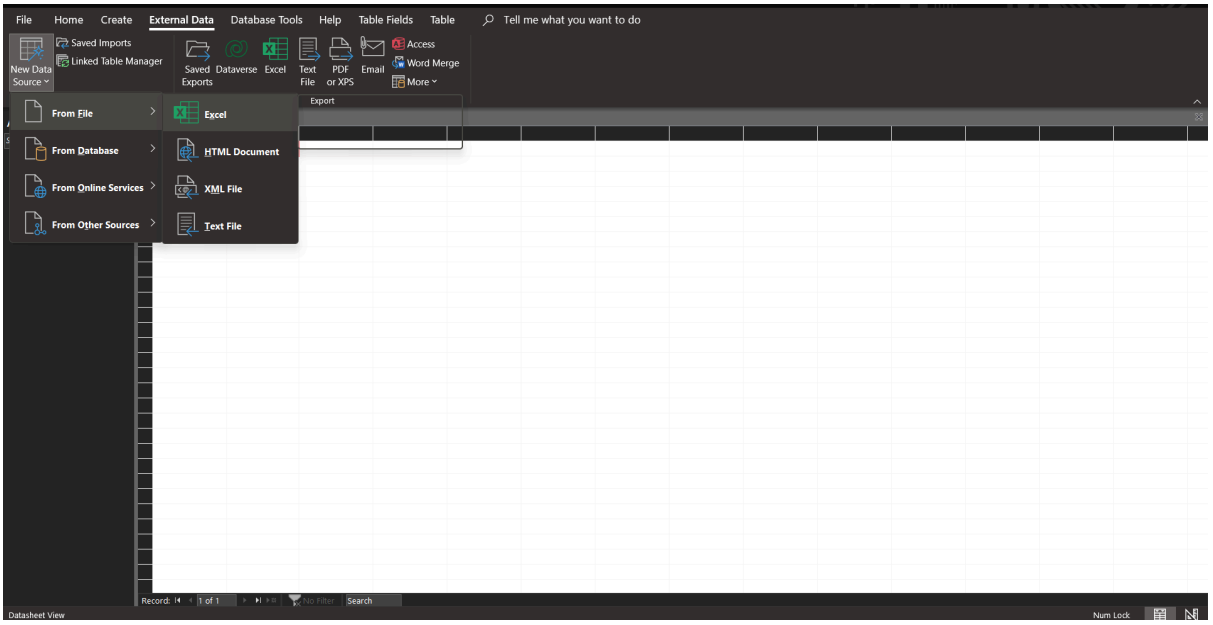
Introduction

This study explains how data collected from clients is used by insurance companies to develop new business strategies and to enhance the quality of services offered to customers. To boost the company's profits and assist it in expanding its customer base, the corporation needs to implement new marketing tactics within a specified amount of time and under a predetermined budget. It is imperative that problems with data inequality be rectified before any adjustments can be made in the marketing department. Analysing the current customers' data using SQL and R data analysis tools to adjust marketing tactics according to the requirements of the consumers.

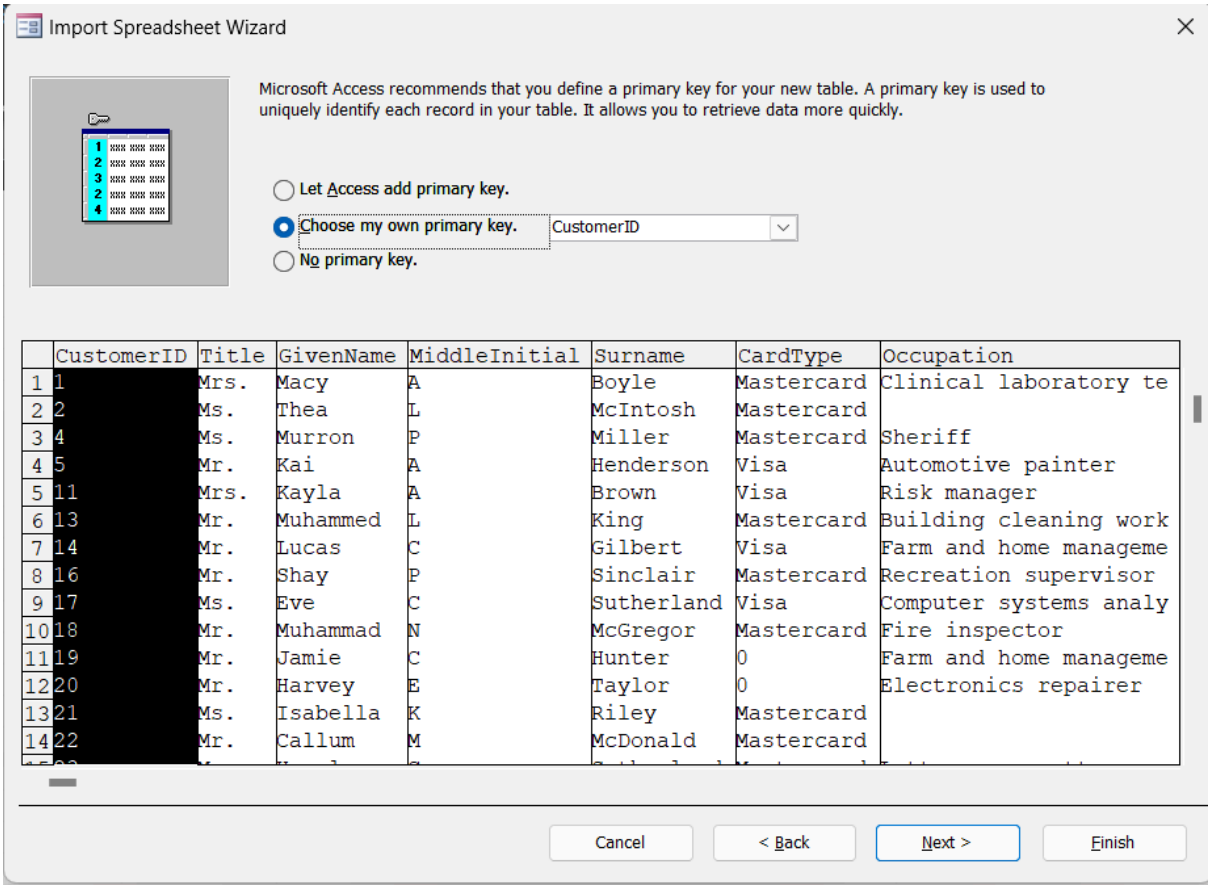
Database Documentation

- Database development in Microsoft Access

The four datasets provided to us; Customer, Motor Policies, Health Policies and Travel Policies in .xlsx format are imported in MS Access by clicking on ‘External Data’ > ‘New Data Source’ > ‘From File’ > ‘Excel’. Refer to the screenshot for further reference:



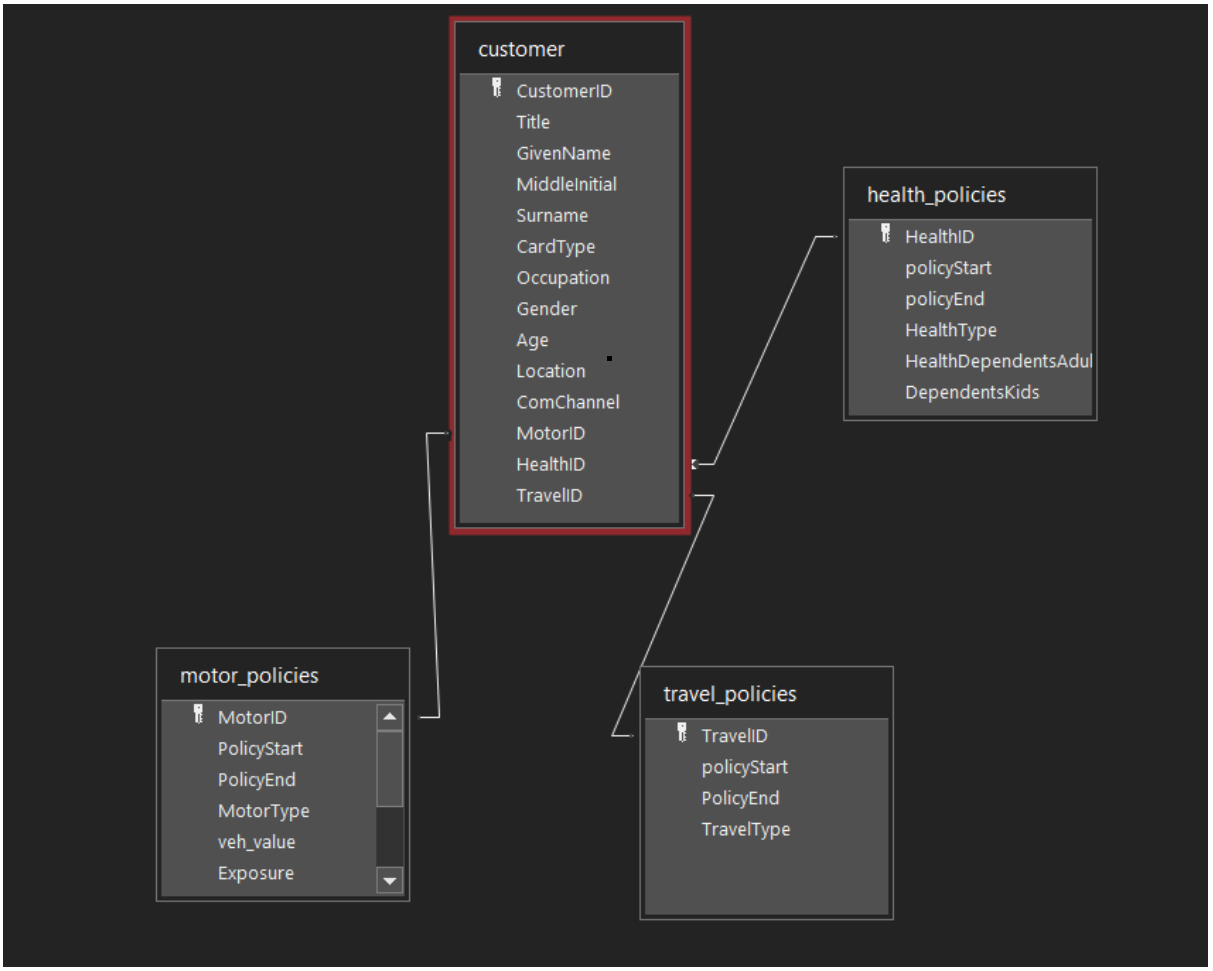
Customer dataset is the primary table, and the other three datasets holds the information related to the specific policies and are required to be related to the main table. Setting up ‘Primary Key’ should be the first step before forming associations with the other tables as it helps uniquely identify each record in the table and allows data to be retrieved quickly. The ‘Primary key’ in the variables or columns of each individual datasets are; Customer ID in the Customer dataset, Motor ID in the Motor Policy dataset, Health ID in the Health Policy dataset and Travel ID in the Travel Policy dataset. This is done while uploading the data in the MS Access, follow the screenshot for reference:



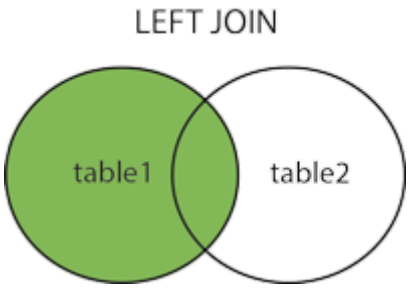
- Linking the table to form Analytics Base Table

Merge the data from the other tables; Health, Motor and Travel policy, we need to identify the foreign keys in tables mentioned above in order to form relationships with the primary tables, i.e., Customer table. Foreign keys are another table's primary key. The information about the same has been explained above.

The relation is formed between the primary key of the main dataset and the foreign keys of the other datasets, below is the pictorial representation of the same.



Using the Left Join function, the analytics base table is created. The left join function is used out of all other join fucntion for the given situation, because it helps process the records from the left (customer) table and the matched records from the other datasets (Health, Motor and Travel policy). Also, it helps retain all the unique observations specific to each datasets and doesn't negate then in the final resulting tables. If the data is not available in the right tables, the observations are populated as NULL values. This can be explained with a help of a following venn diagram:



- Limitations of the current approach

The limitations of using the MS Access tool for the given task are paramount and affect the proficiency expected from the tools in this fast-growing age of data.

1. Size limitation - The data in the MS Access is stored in a single file and the maximum size is only limited 2GB. It will prevent the more data to be saved on the same file. Thus, it makes it unideal for organization churning data in large amounts. (Taneja, 2022)

2. Lags and Slow processing - If the data consisting of multimedia files is used the processing becomes significantly slow and impacts the overall experience. This can be also the result of a greater number of users working on the same data concurrently. (Taneja, 2022)

3. Limitation to the number of users – The official statement around the number of users working the same database in the MS Access is 255, however it is not entirely true in terms of its impact on the processing speed. Only 10 users can work on the same database at the same time and even that starts to impact the speed of the tool. However, modern databases like Hyperbase and Airtable follow the client-server design enabling multiple users to access the database without affecting the speed. (Taneja, 2022)

Data Quality Report

The goals are to identify any quality issues with data like inconsistency, outliers, or missing values from the data. All the files were uploaded to the R Studio for addressing the quality issues using the 'read_excel' command falling under the (readxl) library.

```
#Reading the datasets
customerdata <- read_excel("Data 1_Customer.xlsx")
motorpolicies <- read_excel("Data 2_Motor Policies.xlsx")
healthpolicies <- read_excel("Data 3_Health Policies.xlsx")
travelpolicies <- read_excel("Data 4_Travel Policies.xlsx")
```

Once all the datasets are upload, using the left join function falling under the 'dplyr' package, an analytics base table is formed "ABT_Final". The resultant tables 4085 observation and thirty-two variables.

```
#Joining all the datasets in to Analysis Base Table
ABT1 <- left_join(customerdata,motorpolicies,by='MotorID')
ABT2 <- left_join(ABT1,healthpolicies,by="HealthID")
ABT_Final <- left_join(ABT2,travelpolicies,by="TravelID")
```

```
> str(ABT_Final)
tibble [4,085 × 32] (S3: tbl_df/tbl/data.frame)
 $ CustomerID            : num [1:4085] 1 2 4 5 11 13 14 16 17 18 ...
 $ Title                 : chr [1:4085] "Mrs." "Ms." "Ms." "Mr." ...
 $ GivenName             : chr [1:4085] "Macy" "Thea" "Murron" "Kai" ...
 $ MiddleInitial         : chr [1:4085] "A" "L" "P" "A" ...
 $ Surname               : chr [1:4085] "Boyle" "McIntosh" "Miller" "Henderson" ...
 $ CardType              : chr [1:4085] "Mastercard" "Mastercard" "Mastercard" "Visa" ...
 $ Occupation            : chr [1:4085] "Clinical laboratory technologist" NA "Sheriff" "Automotive painter" ...
 $ Gender                : chr [1:4085] "female" "female" "female" "male" ...
 $ Age                   : num [1:4085] 23 44 19 47 54 49 19 68 20 49 ...
 $ Location              : chr [1:4085] "Urban" "Urban" "Urban" "Rural" ...
 $ ComChannel            : chr [1:4085] "SMS" "Phone" "SMS" "Phone" ...
 $ MotorID               : num [1:4085] NA NA 6391 8876 1850 ...
 $ HealthID              : num [1:4085] NA 7154 NA 4975 3888 ...
 $ TravelID              : num [1:4085] 6255 NA NA 2178 NA ...
 $ PolicyStart           : POSIXct[1:4085], format: NA NA "2019-05-01" ...
 $ PolicyEnd.x           : POSIXct[1:4085], format: NA NA "2020-05-01" ...
 $ MotorType             : chr [1:4085] NA NA "Bundle" "Single" ...
 $ veh_value             : num [1:4085] NA NA 1.83 1.55 0.45 ...
 $ Exposure              : num [1:4085] NA NA 0.591 0.504 0.602 ...
 $ clm                   : num [1:4085] NA NA 0 0 0 0 0 0 0 0 ...
 $ Numclaims             : num [1:4085] NA NA 0 0 0 0 0 0 0 0 ...
 $ v_body                : chr [1:4085] NA NA "HBACK" "HBACK" ...
 $ v_age                 : num [1:4085] NA NA 1 1 4 3 1 4 1 4 ...
 $ LastClaimDate         : POSIXct[1:4085], format: NA NA NA ...
 $ policyStart.x         : POSIXct[1:4085], format: NA "2019-02-14" NA ...
 $ policyEnd             : POSIXct[1:4085], format: NA "2020-02-14" NA ...
 $ HealthType            : chr [1:4085] NA "Level1" NA "Level1" ...
 $ HealthDependentsAdults: num [1:4085] NA 2 NA 1 2 1 NA 0 0 2 ...
 $ DependentsKids        : num [1:4085] NA 3 NA 2 3 2 NA 0 0 2 ...
 $ policyStart.y         : POSIXct[1:4085], format: "2020-07-20" NA NA ...
 $ PolicyEnd.y           : POSIXct[1:4085], format: "2020-07-27" NA NA ...
 $ TravelType            : chr [1:4085] "Premium" NA NA "Business" ...
```

After observing the data in the resultant analytics base table, we can address the data quality issues as follow:

1. Removing Inconsistencies with variables – These types of quality issue included typos, misplaces punctuations or similar character data which could affect the data analysis and addressing them is paramount. The examples of such an issue can be found in the Gender, Title, ComChannel etc. variables/columns.

- Gender – Using the command 'table(ABT_Final$Gender)' we assess the unique values in the gender column.

```
> table(ABT_Final$Gender)

     f female      m   male
    14   2061      9   2001
```

As we can see, the 'f' and 'm' are clearly incorrect and needs to be addressed to streamline the values under the said column. Using the mutate function we will replace the 'f' and 'm' values to 'female' and 'male'.

```
ABT_Final <- ABT_Final %>%
  mutate(Gender = replace(Gender, Gender == 'f', 'female'))
ABT_Final <- ABT_Final %>%
  mutate(Gender = replace(Gender, Gender == 'm', 'male'))
```

The resultant table shows that the values have been amended.

```
> table(ABT_Final$Gender)

female   male
  2075   2010
```

Similarly, with 'Title' and 'ComChannel' column, using the same function the values will be amended as follow:

From

```
> table(ABT_Final$ComChannel)

    E Email     P Phone     S   SMS
    5  1759     5  1559     2   755
> table(ABT_Final$Title)

 Dr.   Mr  Mr. Mrs.  Ms.
 127    6 1938  948 1066
```

To

```
> table(ABT_Final$ComChannel)

Email Phone   SMS
 1764  1564   757
```

```
> table(ABT_Final$Title)

 Dr.  Mr. Mrs.  Ms.
 127 1944  948 1066
```

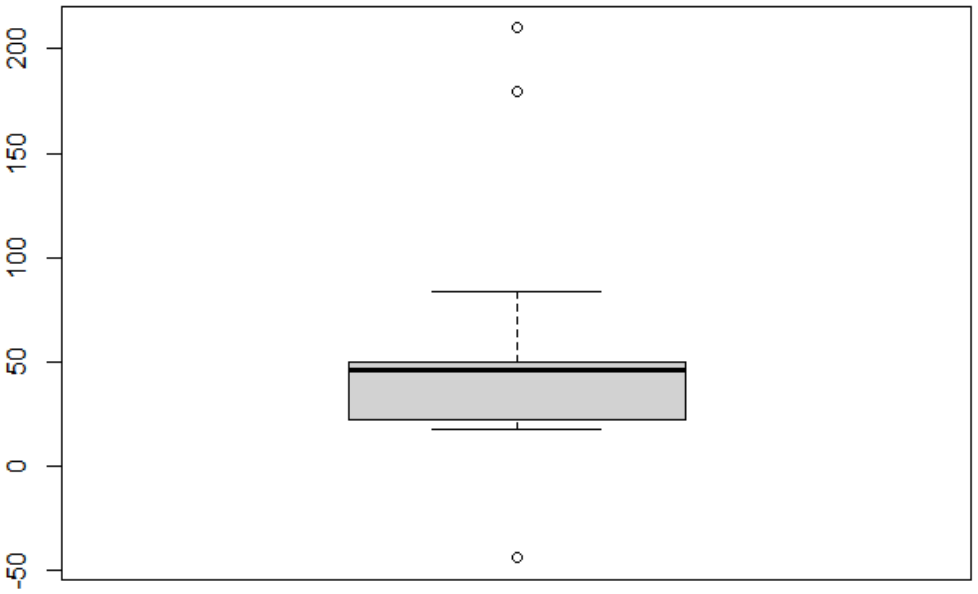
2. NA Values – Using the left join when the analytics base table is formed, it creates a lot of missing values. This is because of the unique values from the right set of tables do not present in the primary table. This can be found using the command colsum(is.na()). Following, can be observed from performing the said function:

```
> colSums(is.na(ABT_Final))
          CustomerID                Title            GivenName       MiddleInitial              Surname              CardType
                   0                    0                    0                   0                    0                     0
          Occupation               Gender                  Age             Location          ComChannel             MotorID
                1554                    0                    0                   0                    0                   728
             HealthID             TravelID          PolicyStart         PolicyEnd.x           MotorType          veh_value
                1547                 1980                  728                 728                  728                   728
            Exposure                  clm           Numclaims              v_body              v_age          LastClaimDate
                 728                  728                  728                 728                  728                  3871
      policyStart.x           policyEnd          HealthType HealthDependentsAdults       DependentsKids       policyStart.y
                1547                 1547                1547                1547                 1547                  1980
        PolicyEnd.y           TravelType
                1980                 1980
```

In a dataset of 4085 observation, there are a lot of NA values in different important columns like 1554 missing values in 'Occupation'. This is an incredible amount given the size of the analytical base table. Thus, those values shall remain the dataset as it would impact the quality of analytical assessment done the analytical base table.

3. Invalid Values – This includes the values that don't make any logical sense. For the given dataset, after observing the summary and box plot graph of the 'Age' column, it is evident that age values falling outside the range 0 to 84 are outliers.

```
> summary(ABT_Final$Age)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 -44.00   22.00   46.00   41.38   50.00  210.00
```



Thus, using the filter function and pipe operators we would eliminate these outliers to maintain a higher degree of quality of our analysis. This can be observed below:

```
> summary(ABT_Final$Age)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  18.00   22.00   46.00   41.33   50.00   84.00
```

4. Incorrect Data Type – Using the str() command or more direct typeof() command, we can clearly observe that a few of the variables/columns are in the incorrect data type.

```
> str(ABT_Final)
tibble [4,085 × 32] (S3: tbl_df/tbl/data.frame)
 $ CustomerID            : num [1:4085] 1 2 4 5 11 13 14 16 17 18 ...
 $ Title                 : chr [1:4085] "Mrs." "Ms." "Ms." "Mr." ...
 $ GivenName             : chr [1:4085] "Macy" "Thea" "Murron" "Kai" ...
 $ MiddleInitial         : chr [1:4085] "A" "L" "P" "A" ...
 $ Surname               : chr [1:4085] "Boyle" "McIntosh" "Miller" "Henderson" ...
 $ CardType              : chr [1:4085] "Mastercard" "Mastercard" "Mastercard" "Visa" ...
 $ Occupation            : chr [1:4085] "Clinical laboratory technologist" NA "Sheriff" "Automotive painter" ...
 $ Gender                : chr [1:4085] "female" "female" "female" "male" ...
 $ Age                   : num [1:4085] 23 44 19 47 54 49 19 68 20 49 ...
 $ Location              : chr [1:4085] "Urban" "Urban" "Urban" "Rural" ...
 $ ComChannel            : chr [1:4085] "SMS" "Phone" "SMS" "Phone" ...
 $ MotorID               : num [1:4085] NA NA 6391 8876 1850 ...
 $ HealthID              : num [1:4085] NA 7154 NA 4975 3888 ...
 $ TravelID              : num [1:4085] 6255 NA NA 2178 NA ...
 $ PolicyStart           : POSIXct[1:4085], format: NA NA "2019-05-01" ...
 $ PolicyEnd.x           : POSIXct[1:4085], format: NA NA "2020-05-01" ...
 $ MotorType             : chr [1:4085] NA NA "Bundle" "Single" ...
 $ veh_value             : num [1:4085] NA NA 1.83 1.55 0.45 ...
 $ Exposure              : num [1:4085] NA NA 0.591 0.504 0.602 ...
 $ clm                   : num [1:4085] NA NA 0 0 0 0 0 0 0 0 ...
 $ Numclaims             : num [1:4085] NA NA 0 0 0 0 0 0 0 0 ...
 $ v_body                : chr [1:4085] NA NA "HBACK" "HBACK" ...
 $ v_age                 : num [1:4085] NA NA 1 1 4 3 1 4 1 4 ...
 $ LastClaimDate         : POSIXct[1:4085], format: NA NA NA ...
 $ policyStart.x         : POSIXct[1:4085], format: NA "2019-02-14" NA ...
 $ policyEnd             : POSIXct[1:4085], format: NA "2020-02-14" NA ...
 $ HealthType            : chr [1:4085] NA "Level1" NA "Level1" ...
 $ HealthDependentsAdults: num [1:4085] NA 2 NA 1 2 1 NA 0 0 2 ...
 $ DependentsKids        : num [1:4085] NA 3 NA 2 3 2 NA 0 0 2 ...
 $ policyStart.y         : POSIXct[1:4085], format: "2020-07-20" NA NA ...
 $ PolicyEnd.y           : POSIXct[1:4085], format: "2020-07-27" NA NA ...
 $ TravelType            : chr [1:4085] "Premium" NA NA "Business" ...
```

For instance, variables like ‘Title’, ‘CardType’, ‘Gender’, ‘Location’, ‘ComChannel’ etc. have set values. As an example, ‘Title’ column is of ‘Chr’ data types whereas it should be in ‘Factor’ with "Dr." "Mr." "Mrs." "Ms.". This will be converted using the as.factor() command. The same can be observed below:

```
> str(ABT_Final)
tibble [4,082 × 32] (S3: tbl_df/tbl/data.frame)
 $ CustomerID            : num [1:4082] 1 2 4 5 11 13 14 16 17 18 ...
 $ Title                 : Factor w/ 4 levels "Dr.","Mr.","Mrs.",..: 3 4 4 2 3 2 2 2 4 2 ...
 $ GivenName             : chr [1:4082] "Macy" "Thea" "Murron" "Kai" ...
 $ MiddleInitial         : chr [1:4082] "A" "L" "P" "A" ...
 $ Surname               : chr [1:4082] "Boyle" "McIntosh" "Miller" "Henderson" ...
 $ CardType              : Factor w/ 3 levels "0","Mastercard",..: 2 2 2 3 3 2 3 2 3 2 ...
 $ Occupation            : chr [1:4082] "Clinical laboratory technologist" NA "Sheriff" "Automotive painter" ...
 $ Gender                : Factor w/ 4 levels "f","female","m",..: 2 2 2 4 2 4 4 4 1 4 ...
 $ Age                   : num [1:4082] 23 44 19 47 54 49 19 68 20 49 ...
 $ Location              : Factor w/ 2 levels "Rural","Urban": 2 2 2 1 1 1 1 1 2 2 ...
 $ ComChannel            : Factor w/ 3 levels "Email","Phone",..: 3 2 3 2 1 1 1 2 1 1 ...
 $ MotorID               : num [1:4082] NA NA 6391 8876 1850 ...
 $ HealthID              : num [1:4082] NA 7154 NA 4975 3888 ...
 $ TravelID              : num [1:4082] 6255 NA NA 2178 NA ...
 $ PolicyStart           : POSIXct[1:4082], format: NA NA "2019-05-01" "2019-10-07" ...
 $ PolicyEnd.x           : POSIXct[1:4082], format: NA NA "2020-05-01" "2020-10-07" ...
 $ MotorType             : chr [1:4082] NA NA "Bundle" "Single" ...
 $ veh_value             : num [1:4082] NA NA 1.83 1.55 0.45 ...
 $ Exposure              : num [1:4082] NA NA 0.591 0.504 0.602 ...
 $ clm                   : num [1:4082] NA NA 0 0 0 0 0 0 0 0 ...
 $ Numclaims             : num [1:4082] NA NA 0 0 0 0 0 0 0 0 ...
 $ v_body                : chr [1:4082] NA NA "HBACK" "HBACK" ...
 $ v_age                 : num [1:4082] NA NA 1 1 4 3 1 4 1 4 ...
 $ LastClaimDate         : POSIXct[1:4082], format: NA NA NA NA ...
 $ policyStart.x         : POSIXct[1:4082], format: NA "2019-02-14" NA "2019-05-17" ...
 $ policyEnd             : POSIXct[1:4082], format: NA "2020-02-14" NA "2020-05-17" ...
 $ HealthType            : chr [1:4082] NA "Level1" NA "Level1" ...
 $ HealthDependentsAdults: num [1:4082] NA 2 NA 1 2 1 NA 0 0 2 ...
 $ DependentsKids        : num [1:4082] NA 3 NA 2 3 2 NA 0 0 2 ...
 $ policyStart.y         : POSIXct[1:4082], format: "2020-07-20" NA NA "2020-11-28" ...
 $ PolicyEnd.y           : POSIXct[1:4082], format: "2020-07-27" NA NA "2020-11-29" ...
 $ TravelType            : chr [1:4082] "Premium" NA NA "Business" ...
```

As a recommendation, the organisations can address the data quality issues right at the sources by employing the services of high-quality data professionals. Most of the time, problems with the quality

of the data can be resolved by cleaning up the original source. In this situation, the proverb "garbage in, garbage out" is applicable because if the source data are wrong or incomplete, the database will become corrupted, and the results will be of inferior quality. (Redpoint Global, 2016)

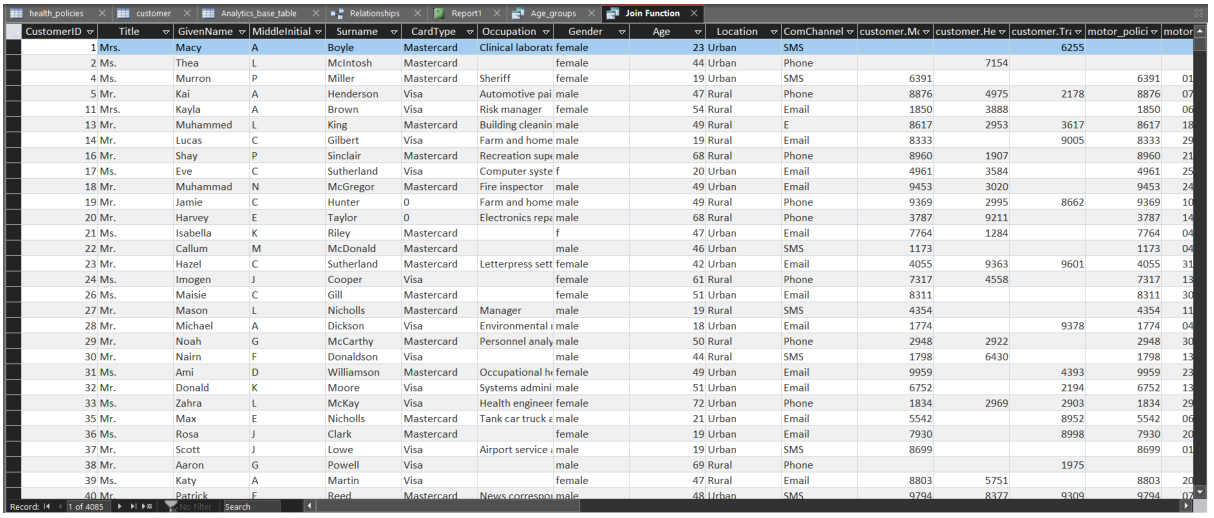
Insights Report

SQL queries and insights

- Forming the Analytics Base Table

  Using the query below, the Analytics Base Table is created:

  SELECT * INTO Analytics_base_table
  FROM ((customer LEFT JOIN motor_policies ON customer.MotorID = motor_policies.MotorID) LEFT JOIN health_policies ON customer.HealthID = health_policies.HealthID) LEFT JOIN travel_policies ON customer.TravelID = travel_policies.TravelID;



CustomerID	Title	GivenName	MiddleInitial	Surname	CardType	Occupation	Gender	Age	Location	ComChannel	customer.Mc	customer.He	customer.Tra	motor_polici	motor
1	Mrs.	Macy	A	Boyle	Mastercard	Clinical laborat	female	23	Urban	SMS			6255		
2	Ms.	Thea	L	McIntosh	Mastercard		female	44	Urban	Phone		7154			
4	Ms.	Murron	P	Miller	Mastercard	Sheriff	female	19	Urban	SMS	6391			6391	01
5	Mr.	Kai	A	Henderson	Visa	Automotive pai	male	47	Rural	Phone	8876	4975	2178	8876	07
11	Mrs.	Kayla	A	Brown	Visa	Risk manager	female	54	Rural	Email	1850	3888		1850	06
13	Mr.	Muhammed	L	King	Mastercard	Building cleanin	male	49	Rural	E	8617	2953	3617	8617	18
14	Mr.	Lucas	C	Gilbert	Visa	Farm and home	male	19	Rural	Email	8333		9005	8333	29
16	Mr.	Shay	P	Sinclair	Mastercard	Recreation supe	male	68	Rural	Phone	8960	1907		8960	21
17	Ms.	Eve	C	Sutherland	Visa	Computer syste	f	20	Urban	Email	4961	3584		4961	25
18	Mr.	Muhammad	N	McGregor	Mastercard	Fire inspector	male	49	Urban	Email	9453	3020		9453	24
19	Mr.	Jamie	C	Hunter	0	Farm and home	male	49	Rural	Phone	9369	2995	8662	9369	10
20	Mr.	Harvey	E	Taylor	0	Electronics repa	male	68	Rural	Phone	3787	9211		3787	14
21	Ms.	Isabella	K	Riley	Mastercard		f	47	Urban	Email	7764	1284		7764	04
22	Mr.	Callum	M	McDonald	Mastercard		male	46	Urban	SMS	1173			1173	04
23	Mr.	Hazel	C	Sutherland	Mastercard	Letterpress sett	female	42	Urban	Email	4055	9363	9601	4055	31
24	Ms.	Imogen	J	Cooper	Visa		female	61	Rural	Phone	7317	4558		7317	13
26	Ms.	Maisie	C	Gill	Mastercard		female	51	Urban	Email	8311			8311	30
27	Mr.	Mason	L	Nicholls	Mastercard	Manager	male	19	Rural	SMS	4354			4354	11
28	Mr.	Michael	A	Dickson	Visa	Environmental i	male	18	Urban	Email	1774		9378	1774	04
29	Mr.	Noah	G	McCarthy	Mastercard	Personnel analy	male	50	Rural	Phone	2948	2922		2948	30
30	Mr.	Nairn	F	Donaldson	Visa		male	44	Rural	SMS	1798	6430		1798	13
31	Ms.	Ami	D	Williamson	Mastercard	Occupational h	female	49	Urban	Email	9959		4393	9959	23
32	Mr.	Donald	K	Moore	Visa	Systems admini	male	51	Urban	Email	6752		2194	6752	13
33	Ms.	Zahra	L	McKay	Visa	Health engineer	female	72	Urban	Phone	1834	2969	2903	1834	29
35	Mr.	Max	E	Nicholls	Mastercard	Tank car truck a	male	21	Urban	Email	5542		8952	5542	06
36	Ms.	Rosa	J	Clark	Mastercard		female	19	Urban	Email	7930		8998	7930	20
37	Mr.	Scott	J	Lowe	Visa	Airport service a	male	19	Urban	SMS	8699			8699	01
38	Mr.	Aaron	G	Powell	Visa		male	69	Rural	Phone			1975		
39	Ms.	Katy	A	Martin	Visa		female	47	Rural	Email	8803	5751		8803	20
40	Mr.	Patrick	F	Reed	Mastercard	News correspo	male	48	Urban	SMS	9794	8377	9309	9794	07

  The said query uses the functioning of the LEFT JOIN in SQL to record and match the column from the previously mentioned tables. Elements like primary keys and primary keys are crucial in the operation derived from the said query.

- Relationship between Number of Claims and Gender columns

  The said relation is derived using the following query:

  SELECT t1.[Gender], count(t1.[Numclaims]) AS Number_of_Claims
  FROM Analytics_base_table AS t1
  WHERE t1.[Gender] <> ''
  GROUP BY t1.[Gender]
  HAVING count(t1.[Numclaims]) >1;

Gender	Number_of_Claims
female	1695
male	1659

By getting the count of the number of claims by individual values of the ‘Gender’ column, we can understand the relation of said variables. Since the column ‘Numclaims’ is unique to the Motor Policy dataset, it can be concluded that the customer’s belonging the gender value ‘Females’ has more count of claims, thus they are more prone to paying higher premiums on renewing of the policies and thus should be carefully targeted just before the renewal deadline. It can be assumed that they understand the importance and utility of the given policy and the marketing team thus can be more creative toward targeting more ‘Female’ customers.

- Relation between Age Group and Health Policy

  This relation is derived in two steps using the following SQL queries:

  STEP 1. Creating Age Groups

  ```
  SELECT CustomerID, Age, Switch (Age<=30, "Young Adults",
  Age >30 and Age <=50 ,"Middle Aged",
  Age >50, "Old") AS Age_Group INTO NewTable
  FROM Analytics_base_table;
  ```

  First, using the functionality of ‘SWITCH’ statement, the following age groups are created:

  Customer falling in the age bracket of 0 – 30, are categorised as ‘Young Adults’,
  Customer in the age bracket of 31-50 are categorised as ‘Middle Aged’,
  And customers above the age of 50 are categorised as ‘Old’

  Then, using the functionality of ‘INTO’ statement, the outcome is then saved into a new table. “NEW TABLE”

CustomerID	Age	Age_Group
1	23	Young Adults
2	44	Middle Aged
4	19	Young Adults
5	47	Middle Aged
11	54	Old
13	49	Middle Aged
14	19	Young Adults
16	68	Old
17	20	Young Adults
18	49	Middle Aged
19	49	Middle Aged
20	68	Old
21	47	Middle Aged
22	46	Middle Aged
23	42	Middle Aged
24	61	Old
26	51	Old
27	19	Young Adults
28	18	Young Adults
29	50	Middle Aged
30	44	Middle Aged
31	49	Middle Aged
32	51	Old
33	72	Old
35	21	Young Adults
36	19	Young Adults
37	19	Young Adults
38	69	Old
39	47	Middle Aged
40	48	Middle Aged

STEP 2. Relating the data from the query with the columns of the Analytics Base Table

```
SELECT COUNT(Analytics_base_table.customer_HealthID) AS ['Number of Health Policies'], NewTable.Age_Group
FROM Analytics_base_table LEFT JOIN NewTable ON Analytics_base_table.CustomerID = NewTable.CustomerID
WHERE Analytics_base_table.customer_HealthID IS NOT NULL
GROUP BY NewTable.Age_Group;
```

Using the functionality of the LEFT JOIN statement, the columns from the left table (Analytics Base Table) and right table (NewTable) are matched and recorded to form relation between the age groups and health policy.

'Number of Health Policies'	Age_Group
1539	Middle Aged
696	Old
303	Young Adults

From the resultant table, it can be easily identified that customers falling the ‘Middle Aged’ age group understand the importance of health policy and are proactive in having one. While the lowest count is from the customers falling in ‘Young Adults’ category, this is can enable the insurance company and their marketing team to target these customers as it is the right

time such an age group and they can make the offer more lucrative by offering the special premium rates to the younger age groups.

- Relation between Gender, Location and 3 Policies

  The SQL query used is as follow:

  SELECT Location, Gender, COUNT(*) AS ["Number of Policies"]
  FROM Analytics_base_table
  WHERE customer_HealthID IS NOT NULL AND customer_TravelID AND customer_MotorID
  GROUP BY Location, Gender;

  The outcome of the said query can be observed below:

Location	Gender	"Number of Policies"
Rural	female	194
Rural	male	190
Urban	female	310
Urban	male	281

  The table provides crucial information involving the location and gender and all 3 policies. The first hypothesis is that customer from the 'Urban' location have a higher count of people who have bought all the 3 policies, thus marketing team should come up with innovative ways to target people falling the 'Rural' location. Second hypothesis is that from both the location, 'Female' customer has a higher count in terms of people who bought all 3 policies. Thus, the focus should shift more towards targeting people belonging to the 'Male' gender by restructuring marketing strategies that should attract males more.

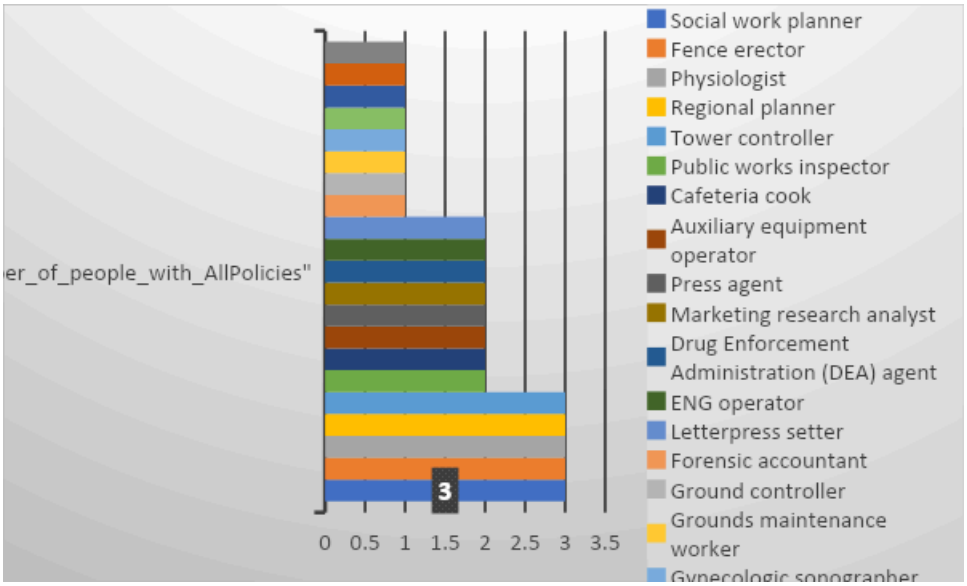
- Relation between Occupation and all three policies

  The mentioned relation is achieved through following query:

  SELECT COUNT(*) AS ["Number_of_people_with_AllPolicies"], Occupation
  FROM Analytics_base_table
  WHERE customer_HealthID and customer_TravelID and customer_MotorID and Occupation IS NOT NULL
  GROUP BY Occupation;

  The outcome from the same can be explained below as follow:

"Number_of_people_with_AllPolicies"	Occupation
3	Social work planner
3	Fence erector
3	Physiologist
3	Regional planner
3	Tower controller
2	Public works inspector
2	Cafeteria cook
2	Auxiliary equipment operator
2	Press agent
2	Marketing research analyst
2	Drug Enforcement Administration (DEA
2	ENG operator
2	Letterpress setter
2	Fitness worker
2	Emergency medical technician
2	Content editor
2	Systems developer
2	PBX installer
2	Forest conservation and logging worke
2	National account executive
2	Electronics repairer
2	Manifold binding worker
2	Public health dietitian
2	Production planning and expediting cle
2	Electrical and electronics engineering t
2	Fire fighter
2	Bellhop
2	Social science research assistant
2	Farm and home management advisor
2	Marine electronics technician



The resultant data would enable the strategic team to understand their customers more in terms of the occupational category they fall into and thus formulate new strategies to target specifically those occupational categories where the people has shown interest to only one type of policy in order to increase the sales. This would enable them to target their marketing strategies for the available set of values in the occupation column.

Limitation to the current approach and Recommendations

In the addition to the limitation explained previously around using the MS Access, we will now discuss the limitation involving the R language for analysis. It is not a simple language to master, its learning curve is quite steep. It is an effective programming language for programmers with prior experience. (Simplilearn, 2022)

The same level of safety is not present as R lacks even the most elementary security precautions. As a result, it is not a recommended solution for the creation of web-safe applications. In addition, R cannot currently be incorporated within web browsers as this is a lengthy procedure. R is a slower programming language compared to Python and MATLAB. It consumes a considerable amount of memory as memory management is not seen as one of R's strong points. (Simplilearn, 2022)

The data for R must be stored in the physical memory of the computer. In contrast, the increasing frequency of cloud-based storage may render this disadvantage moot soon. The documentation and bundle are of variable quality throughout. Documentation and packages may be inadequate, inconsistent, or non-existent. This is the expense of utilising a language that does not receive official or dedicated support but is maintained and developed by the community. (Simplilearn, 2022)

Appendix 1

SQL queries

- SELECT * INTO Analytics_base_table
  FROM ((customer LEFT JOIN motor_policies ON customer.MotorID = motor_policies.MotorID) LEFT JOIN health_policies ON customer.HealthID = health_policies.HealthID) LEFT JOIN travel_policies ON customer.TravelID = travel_policies.TravelID;

- SELECT t1.[Gender], count(t1.[Numclaims]) AS Number_of_Claims
  FROM Analytics_base_table AS t1
  WHERE t1.[Gender] <> ''
  GROUP BY t1.[Gender]
  HAVING count(t1.[Numclaims]) >1;

- SELECT COUNT(Analytics_base_table.customer_HealthID) AS ['Number of Health Policies'], NewTable.Age_Group

FROM Analytics_base_table LEFT JOIN NewTable ON Analytics_base_table.CustomerID = NewTable.CustomerID
WHERE Analytics_base_table.customer_HealthID IS NOT NULL
GROUP BY NewTable.Age_Group;

- SELECT Location, Gender, COUNT(*) AS ["Number of Policies"]
  FROM Analytics_base_table
  WHERE customer_HealthID IS NOT NULL AND customer_TravelID AND customer_MotorID
  GROUP BY Location, Gender;

- SELECT COUNT(*) AS ["Number_of_people_with_AllPolicies"], Occupation
  FROM Analytics_base_table
  WHERE customer_HealthID and customer_TravelID and customer_MotorID and Occupation IS NOT NULL
  GROUP BY Occupation;

Appendix 2

R codes

```
#Installing Packages

install.packages('tidyverse')


#Loading Libraries

library(dplyr)

library(tidyverse)

library(readxl)

library(ggplot2)


#Setting Working Directories

getwd()

setwd("C:/Users/Prayas Sachdeva/Downloads")

getwd()


#Reading the datasets

customerdata <- read_excel("Data 1_Customer.xlsx")

motorpolicies <- read_excel("Data 2_Motor Policies.xlsx")

healthpolicies <- read_excel("Data 3_Health Policies.xlsx")

travelpolicies <- read_excel("Data 4_Travel Policies.xlsx")
```

```
#Joining all the datasets in to Analysis Base Table
ABT1 <- left_join(customerdata,motorpolicies,by='MotorID')
ABT2 <- left_join(ABT1,healthpolicies,by="HealthID")
ABT_Final <- left_join(ABT2,travelpolicies,by="TravelID")
str(ABT_Final)
boxplot(ABT_Final$Age)

#Summary of the ABT_Final
summary(ABT_Final)

#Data Quality

#1 Replacing Inconsistencies
#1.1 GENDER
ABT_Final <- ABT_Final %>%
  mutate(Gender = replace(Gender, Gender == 'f', 'female'))
ABT_Final <- ABT_Final %>%
  mutate(Gender = replace(Gender, Gender == 'm', 'male'))
table(ABT_Final$Gender)
#1.2 COMCHANNEL
ABT_Final <- ABT_Final %>%
  mutate(ComChannel = replace(ComChannel, ComChannel == 'E', 'Email'))
ABT_Final <- ABT_Final %>%
  mutate(ComChannel = replace(ComChannel, ComChannel == 'P', 'Phone'))
ABT_Final <- ABT_Final %>%
  mutate(ComChannel = replace(ComChannel, ComChannel == 'S', 'SMS'))
table(ABT_Final$ComChannel)
#1.3 TITLE
ABT_Final <- ABT_Final %>%
  mutate(Title = replace(Title, Title == 'Mr', 'Mr.'))
table(ABT_Final$Title)
```

```
#2 SUM of NA values
colSums(is.na(ABT_Final))


#3 AGE Invalid Values
summary(ABT_Final$Age)
ABT_Final <- ABT_Final %>%
  filter(Age > 0)  %>%
  filter(Age <= 85)


#4 Converting strings to factors
ABT_Final$Title <- as.factor(ABT_Final$Title)
ABT_Final$CardType <- as.factor(ABT_Final$CardType)
ABT_Final$Gender <- as.factor(ABT_Final$Gender)
ABT_Final$Location <- as.factor(ABT_Final$Location)
ABT_Final$ComChannel <- as.factor(ABT_Final$ComChannel)
```

References

Global, R. (2022) *4 ways to solve data quality issues*, *Redpoint Global*. Available at: https://www.redpointglobal.com/blog/4-ways-solve-data-quality-issues/#:~:text=Fix%20data%20in%20the%20source,and%20produce%20low%20quality%20results. (Accessed: November 21, 2022).

Simplilearn, S. (2022) *What is R: Overview, its applications and what is R used for: Simplilearn*, *Simplilearn.com*. Simplilearn. Available at: https://www.simplilearn.com/what-is-r-article#does_r_have_any_drawbacks (Accessed: November 21, 2022).

Taneja, P. (2022) *5 limitations of Microsoft Access*, *HyperOffice Blog*. Available at: https://www.hyperoffice.com/blog/2019/09/25/5-limitations-of-microsoft-access/ (Accessed: November 21, 2022).

Venn Diagram of a Left Join function (2022) *www.w3schools.com*. Available at: https://www.w3schools.com/sql/img_leftjoin.gif (Accessed: November 21, 2022).