**Fake Names**

There is a dataset called FakeNames. Our aim in this analysis is to predict the column named Amount Paid. The dataset contains 50007 rows and 47 columns Data cleaning. Below is the description of the columns:

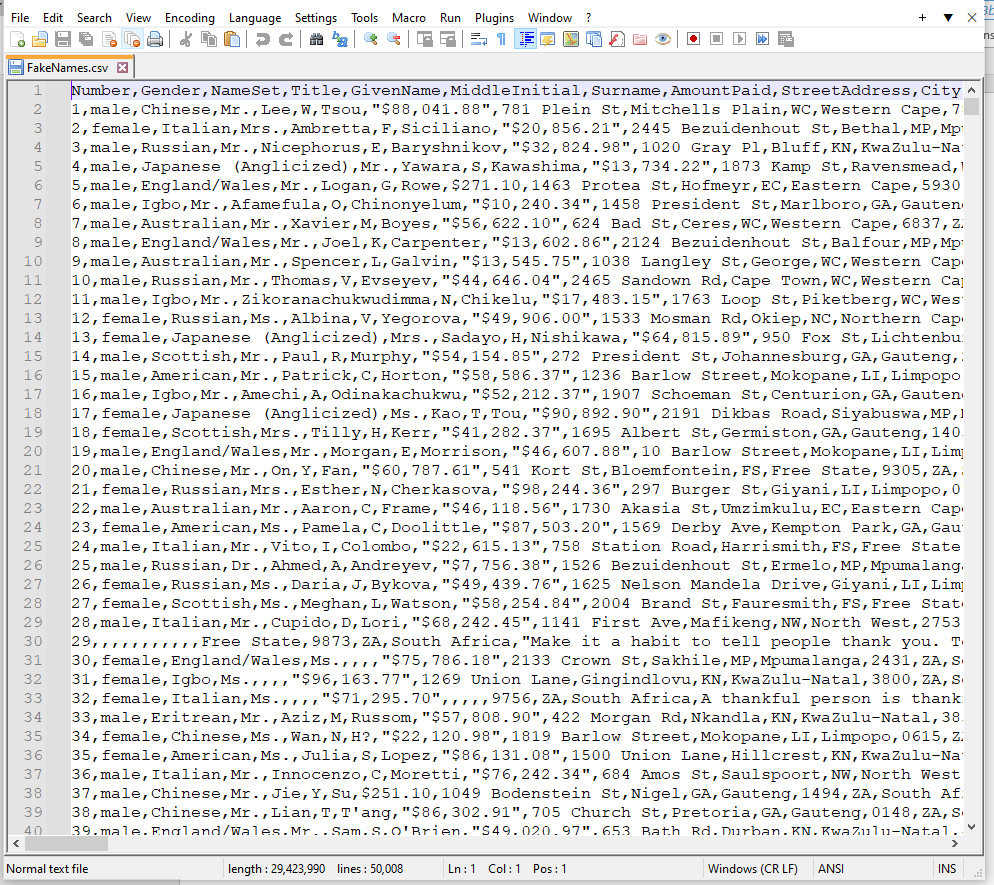
RowNumber-Index, Number-Identification Number, Gender-Male, Female gender group, NameSet-Nationality, Title-Mr., Mrs.,Ms., GivenName-Client's name, MiddleInitial-Middle name initial, Surname-Client's surname, AmountPaid-Bill paid, StreetAddress-Home address, City-City of location of cafe, State-State of location of cafe: Initials, StateFull- State of location of cafe: full name of the state, ZipCode- Zipcode of location of cafe, Country-Country name: Initials, CountryFull- Country name: full, Feedback-Feedback left be clients for service provided, EmailAddress-Client's email adddress, Username-username of a client if order was done on the webpage, Password-password for log in, BrowserUserAgent-Browser used by client, TelephoneNumber-Cell phone number, TelephoneCountryCode-Telephone country code, MothersMaiden-Mother's maiden name, Birthday-The date of birthday of a client, TropicalZodiac-Zodiac sign of a client, CCType-Credit card type, CCNumber-Credit card number, CVV2-credit card cvv code, CCExpires-credit card expiration date, NationalID-National identification number, UPS-uninterruptible power supply, WesternUnionMTCN-Western Union Money Transfer Control Number, MoneyGramMTCN-MoneyGram Money Transfer Control Number, Color-color of credit card, Occupation-Client's occupation, Company-Client's workplace, Vehicle-Client's vehicle, Domain-Domain used by company, BloodType-Client's blood type, Pounds-Client's weight in pounds, Kilograms-Client's weight in kilograms, FeetInches-Client's height in inches, Centimeters-Client's height in centimeters, GUID-globally unique identifier, Latitude-Latitude of location, Longitude-Longitude of location

We can divide our analysis into the following steps:

* Data Uploading Integration (Note Text, Excel, Visual Studio)
* Data Cleaning and Preparation (SQL)
* Data Visualization (Tableau)
* Data Manipulation (Python)
* Data Prediction (Python)

**Data Uploading Integration**

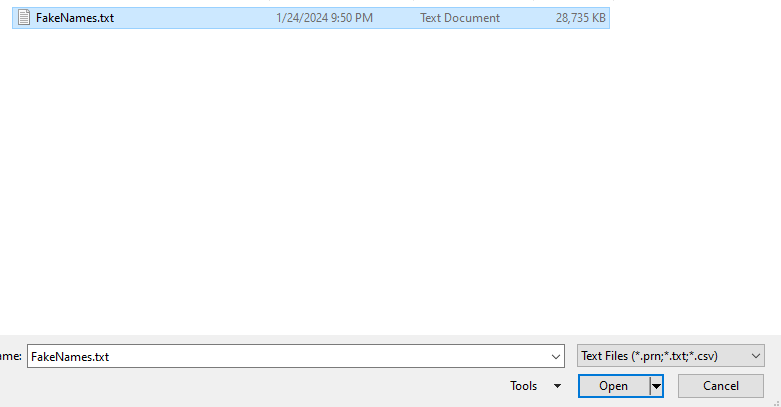
1. Importing the dataset from https://www.superdatascience.com/pages/training , the file name is ‘Fake Names’. Opening with Notepad++ before moving to Excel. This will help us to avoid losing essential data as date format changes (dd-mm-yyyy to mm-dd-yyyy) and general format changes. The dataset contains 50007 rows and 47 columns and looks as below:



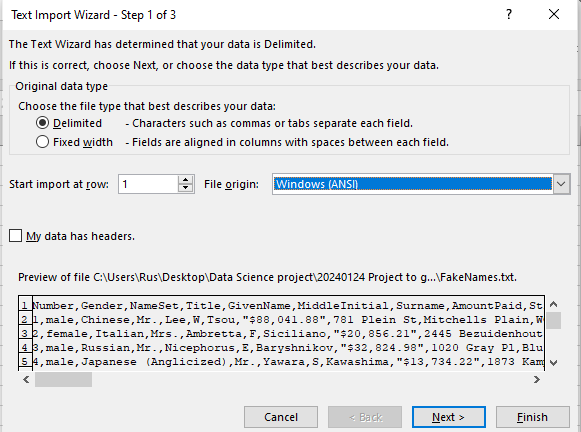
1. To load and import all the data properly we have to check all null values available in the dataset and set some changes when necessary. We should save the file in text format and open it with Excel. It will help us to keep all data unchanged by Excel as it usually does to CSV files. All data stored in text format will remain values untied to Excel formatting.



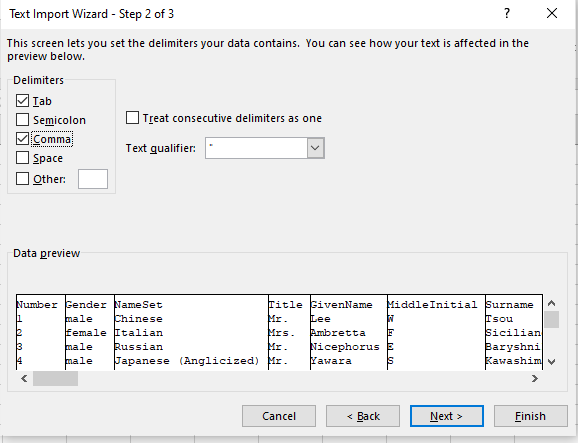
Opening with Excel:



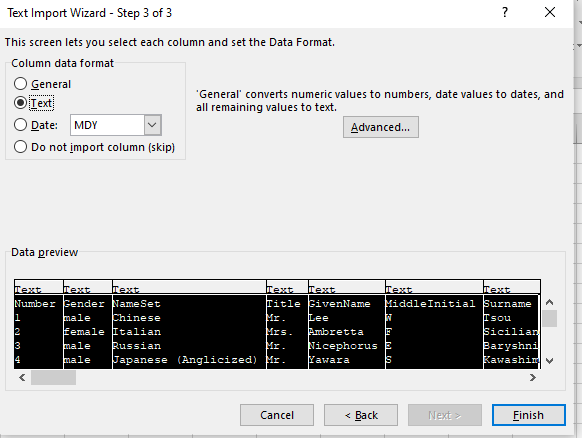
Delimited format is selected:



In the next window we will select comma as a separator, as initially the file with dataset was comma separated file and text qualifier should be identified as quotation mark:

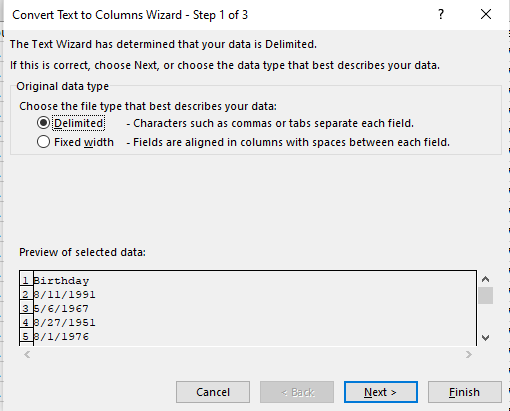


In the next window of third step all data values type should be identified as text:

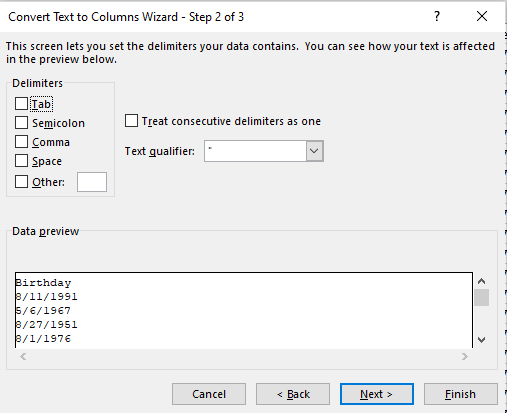


1. Fixing the dollar sign in the amount column and changing the date to the relevant format in Excel:

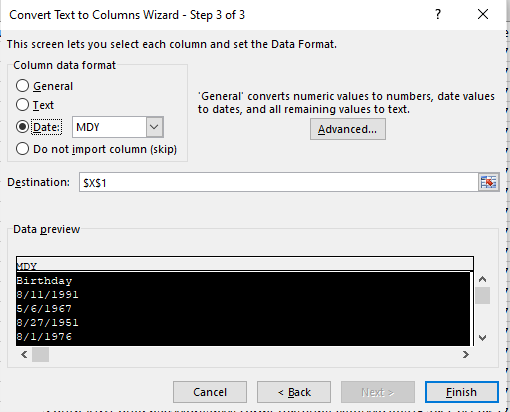
We will start with the date format. As the date was stored in text format, first we need to change the format from text to date format using Data-Text to Columns, then change the format from the current to customized one, in our case yyyy-mm-dd format.



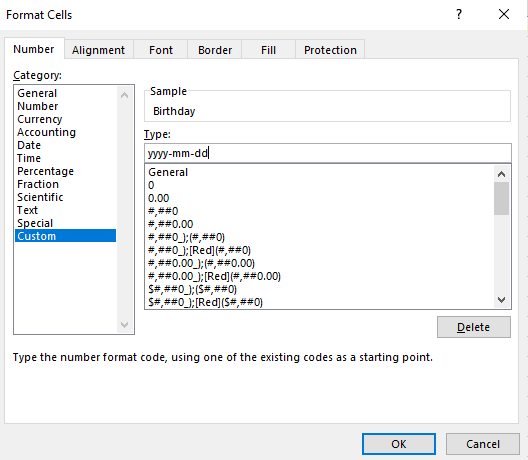
Since we do not need a delimiter for one column the options will remain unselected:



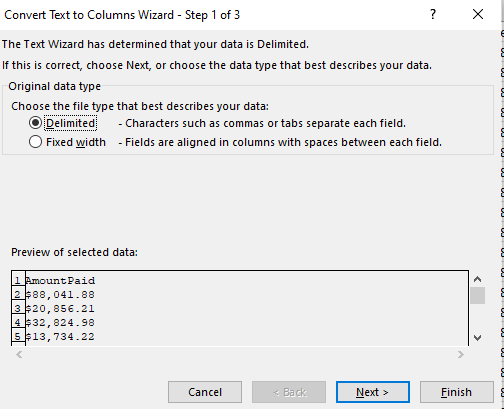
In the third window we will select date format as mdy:



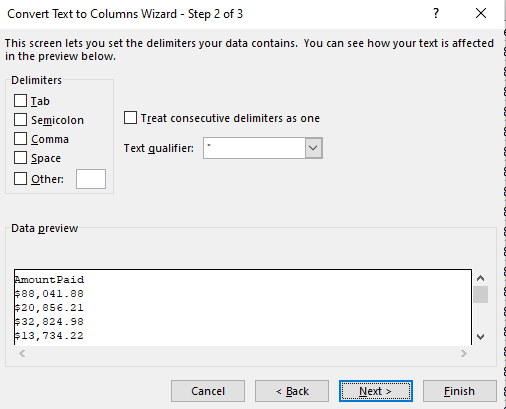
The final yyyy-mm-dd date format can be achieved as follows:

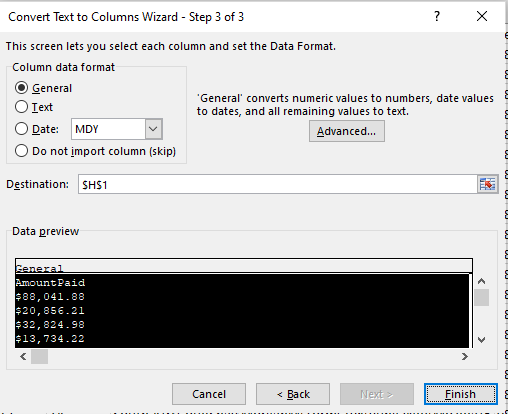


Since there is only one column with a date, there is only one column left, namely, the amount column to be format changed:

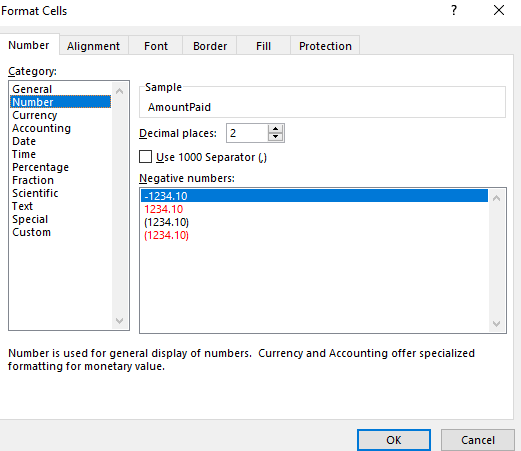


Again delimiter options are left empty.

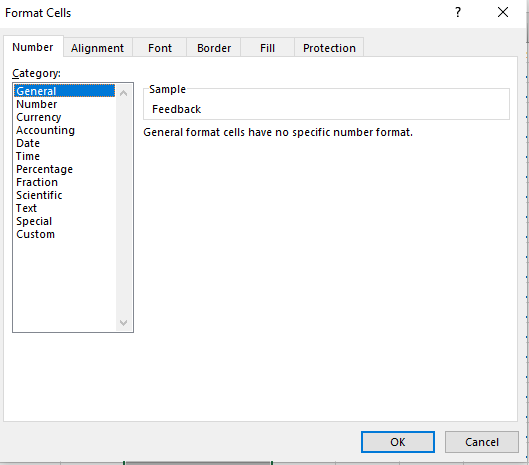




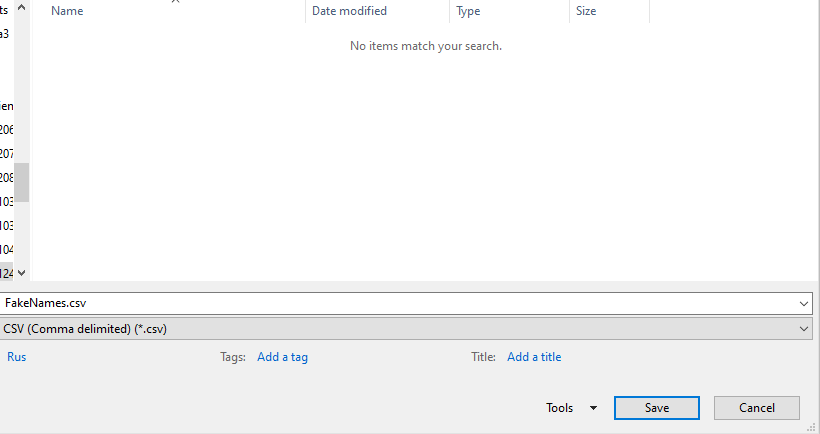
Changing the format from general to number, removing the dollar sign:



It is good to check also the Feedback column, as the feedback is stored as text, in an Excel file there might be a limit to text symbols, and it might lead to losing some feedback information. To avoid this, changing the format from text to general is needed.

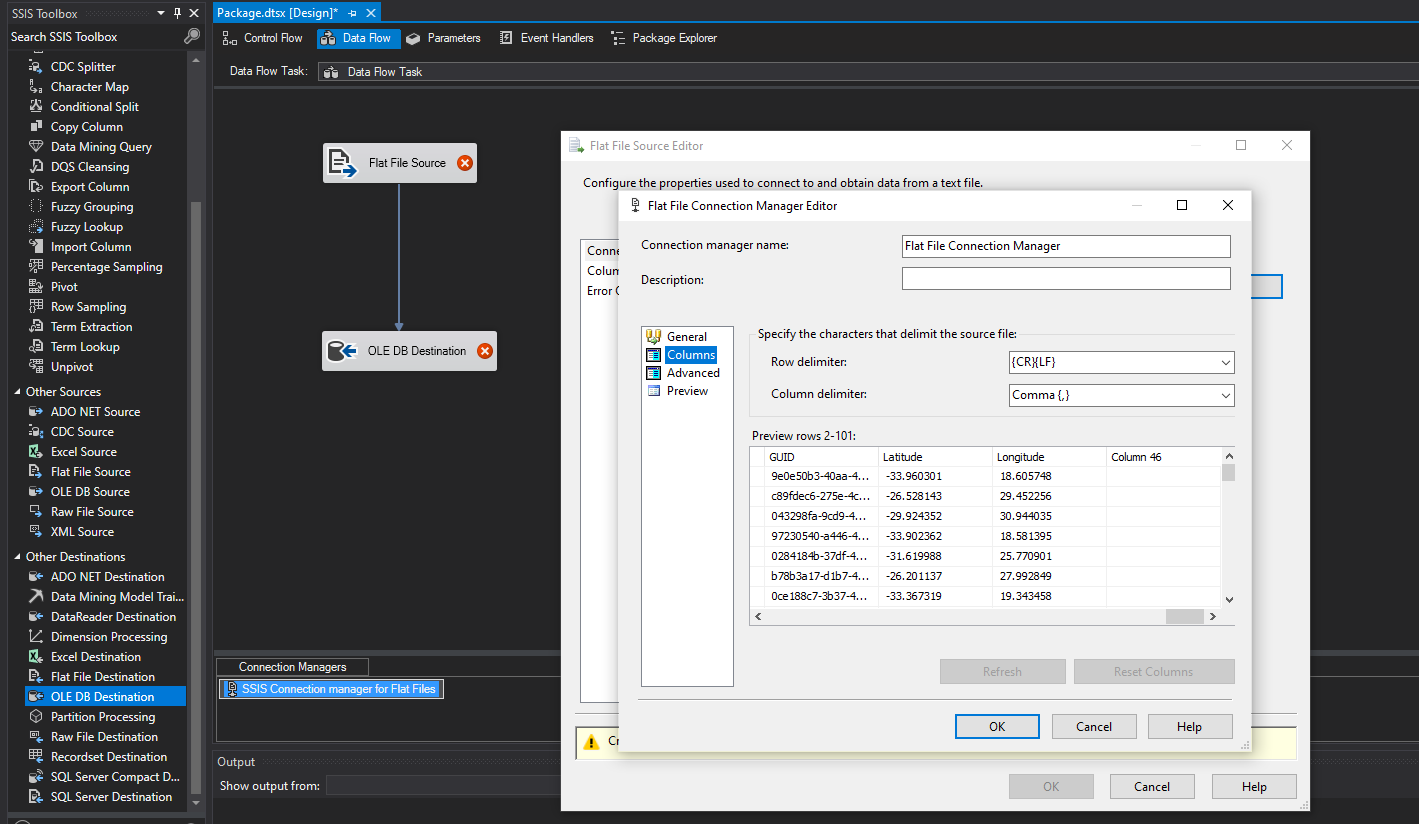


As the format changing is done we can proceed with the saving the file in csv format type.



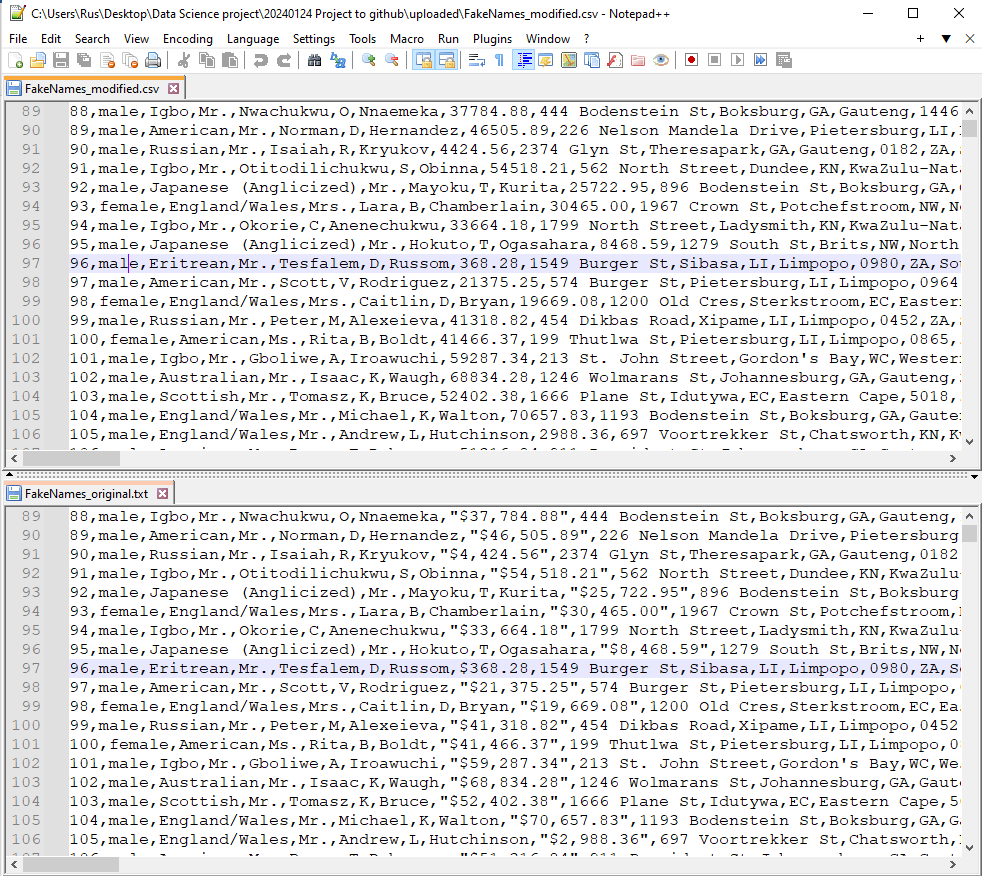
1. ETL (Extract-Transform-Load) process using SSIS platform.

We will upload the file using the Virtual Studio and create an Integration Service Project. As the file might contain some errors due to missed text qualifiers, we need to make changes if there are any.

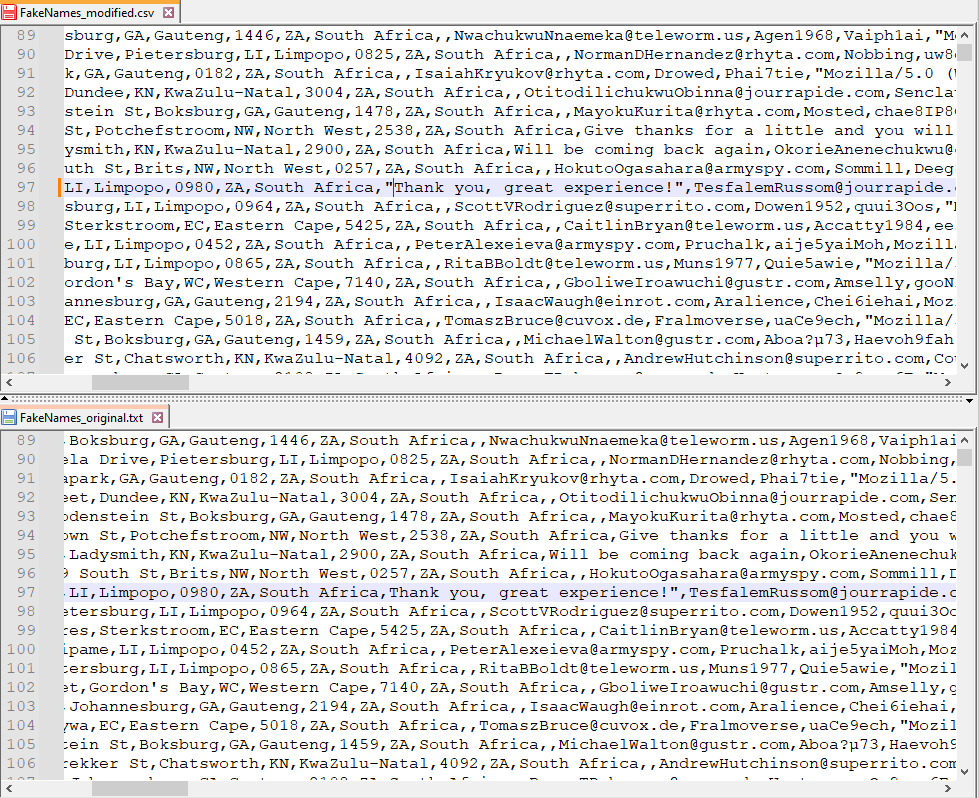


Before uploading the modified file, it is good to check columns, if there are no mismatches in the values as above. The last column might be added because of the shifting data to the right. To solve this problem, we need to check the row that causes this problem. Most of this kind of problem is driven by inappropriate separation of the values, meaning text qualifiers incorrectly identify text columns. Below we can see these rows in the original file.

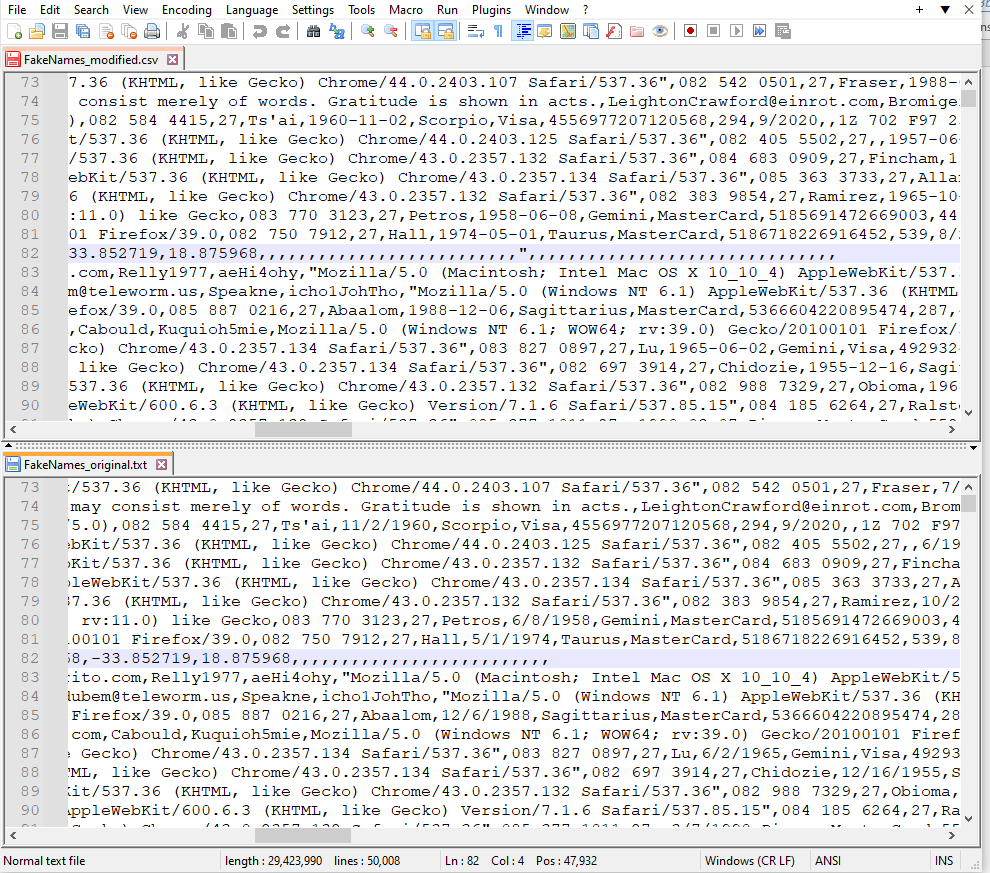
The column 96 in the file below was shifted to the right. To reveal the problem that, it caused, we will use Notepad++ to compare the modified file and the original one:



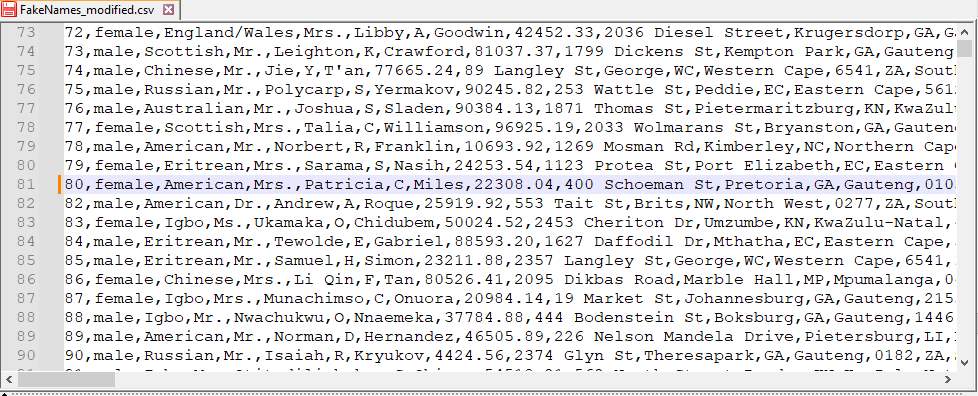
In this row, the quotation mark was missed in the feedback column, so the last part of the feedback was shifted to the next column, causing all remaining data to get skewed to the right. To solve this issue, we can modify this row by adding quotation marks, thereby coming to the right order.



The next column to be checked is row 81. In this case, column values shifted to the left.

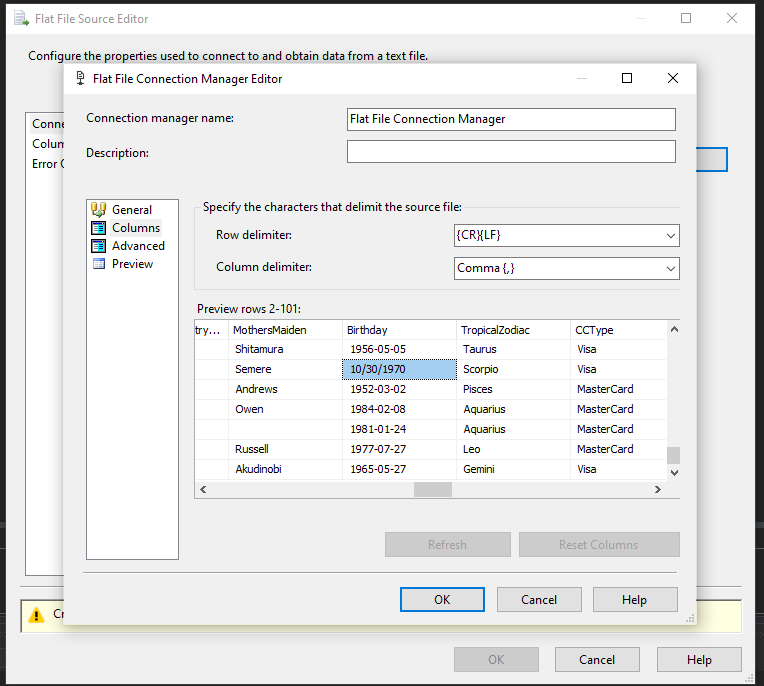


Again the problem is caused by the text qualifier, however adding a quotation mark will not change the issue completely, as the data in this row is not full. There are values missed as in the above. In this case to get rid of the anomaly better to exclude this particular row from the dataset. To be updated and to have proof it is essential to track all the rows excluded and modifications done in a separate report file. Row 82 was excluded:



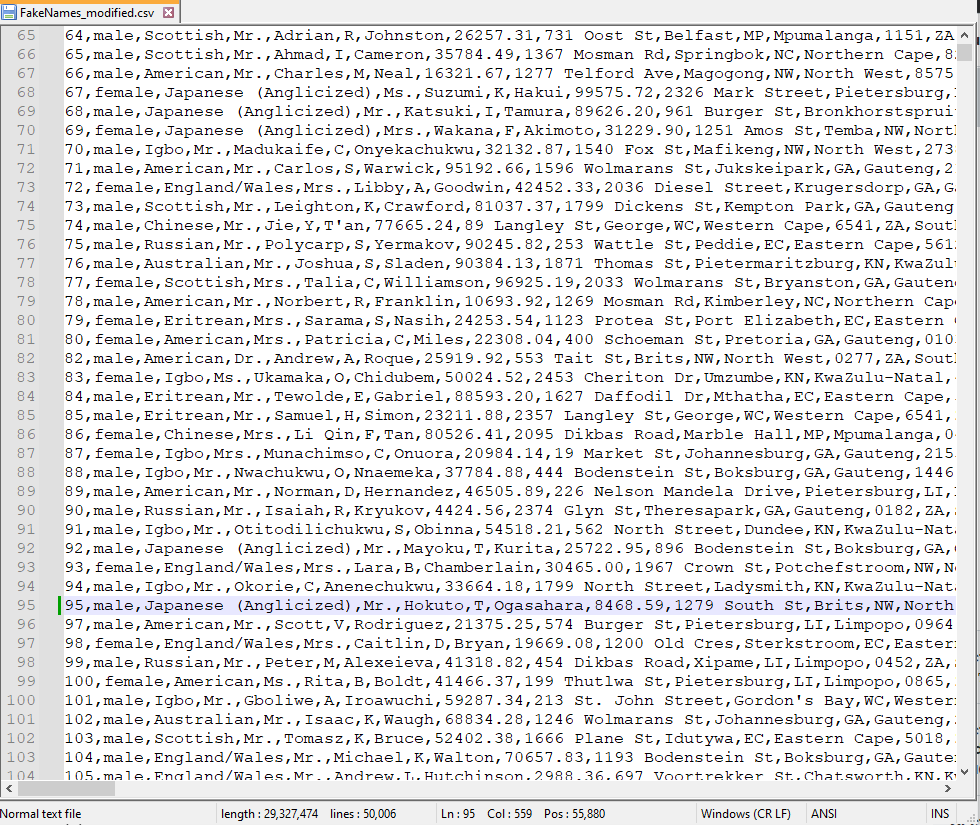
As many more errors can occur in data upload, it is better to automatize the process and let Visual Studio deal with the rest of the errors. That will save us time, and during the uploading, there will be a chance to see and identify what kind of errors we have in the dataset. However, if the dataset contains fewer data, it is quite possible to handle the anomalies manually as above.

Another issue that can be seen in this file is the data in row 96 as below:

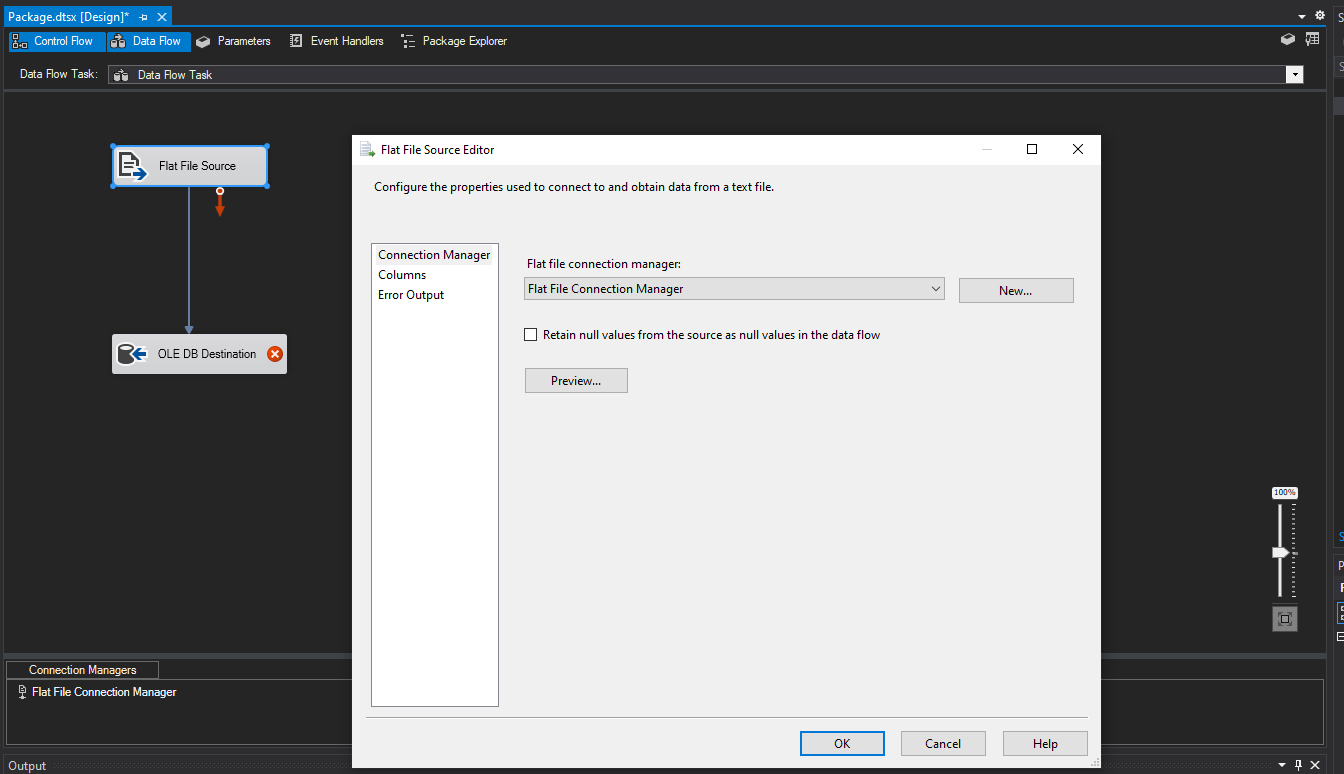


This problem occurs when the data is skewed to the right. When the file was opened with Excel and the date format was properly changed we didn’t make any changes to skewed data. The changes were done after. So basically, when we put the data back in their order by columns, the unformatted date shifted to the left to the initial column ‘Birthday’. This kind of data might lead to the wrong calculations, which is why we need to either change to the right format or delete the row. In this case, we will remove the row from the csv file to be uploaded.

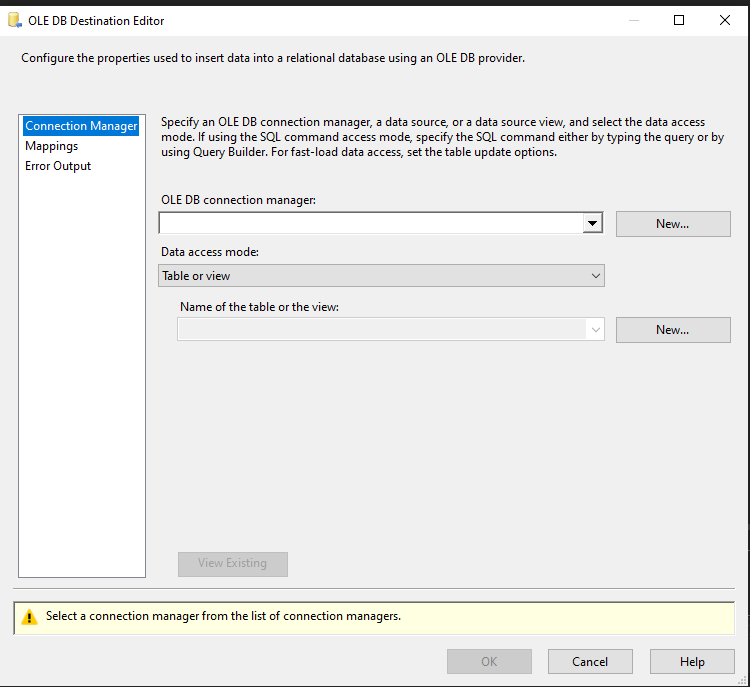




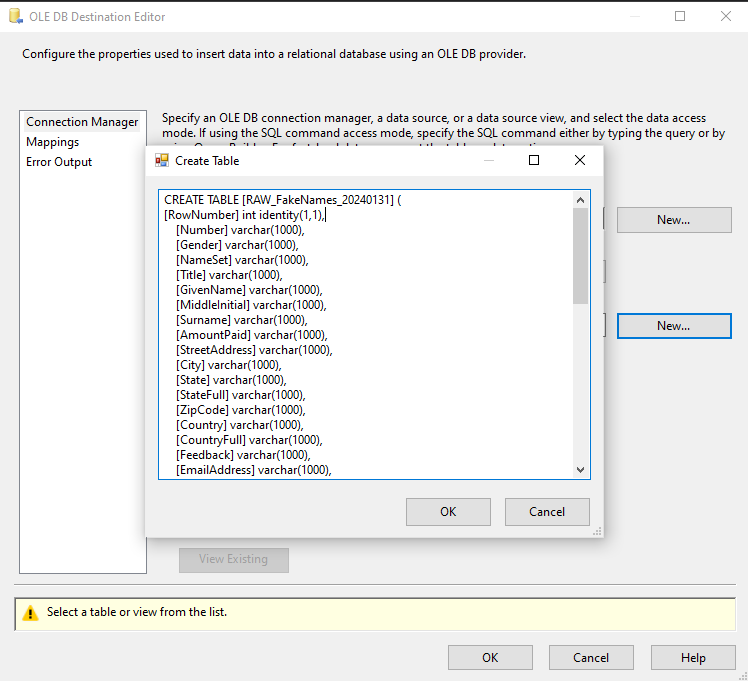
1. Uploading the data as Flat File Source via Visual Studio:



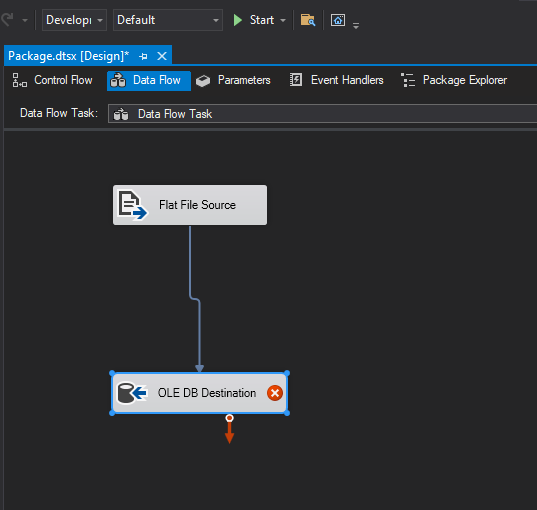
1. OLE DB Destination



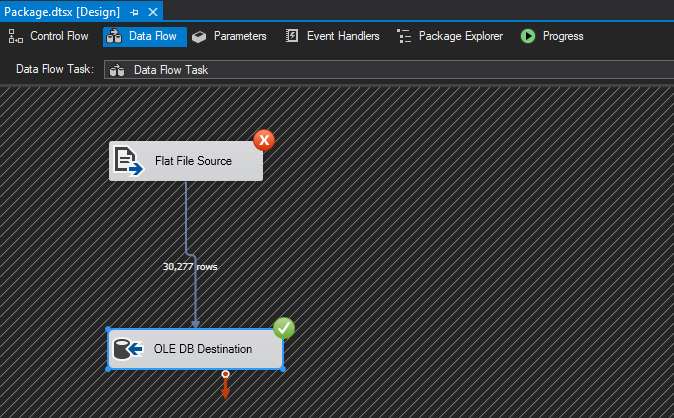
For OLE DB Destination we choose a new table, and select connection manager, connection manager name should be SQL Server name as we are going to integrate the database with SQL Server via Visual Studio. We create a new table as below, giving the name for the table and adding the Row Number.



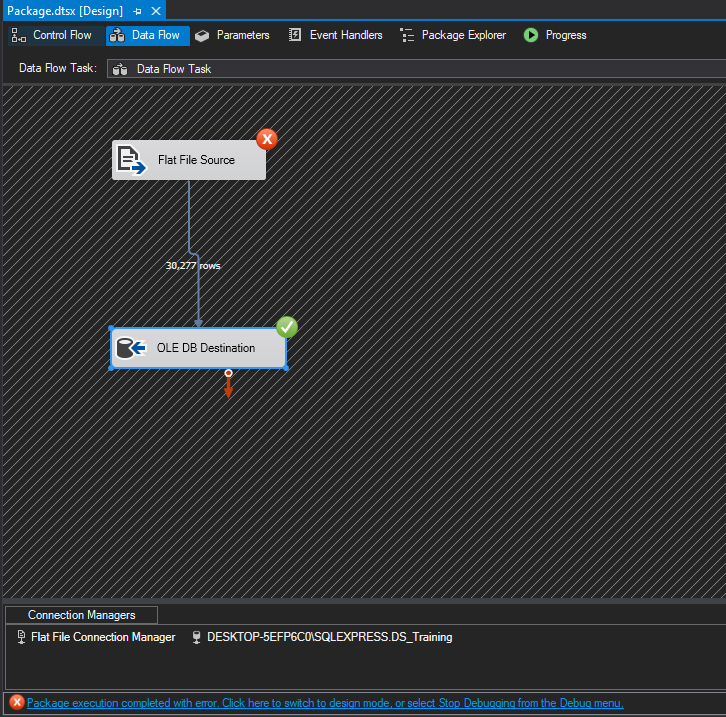
We can start proceed integration.



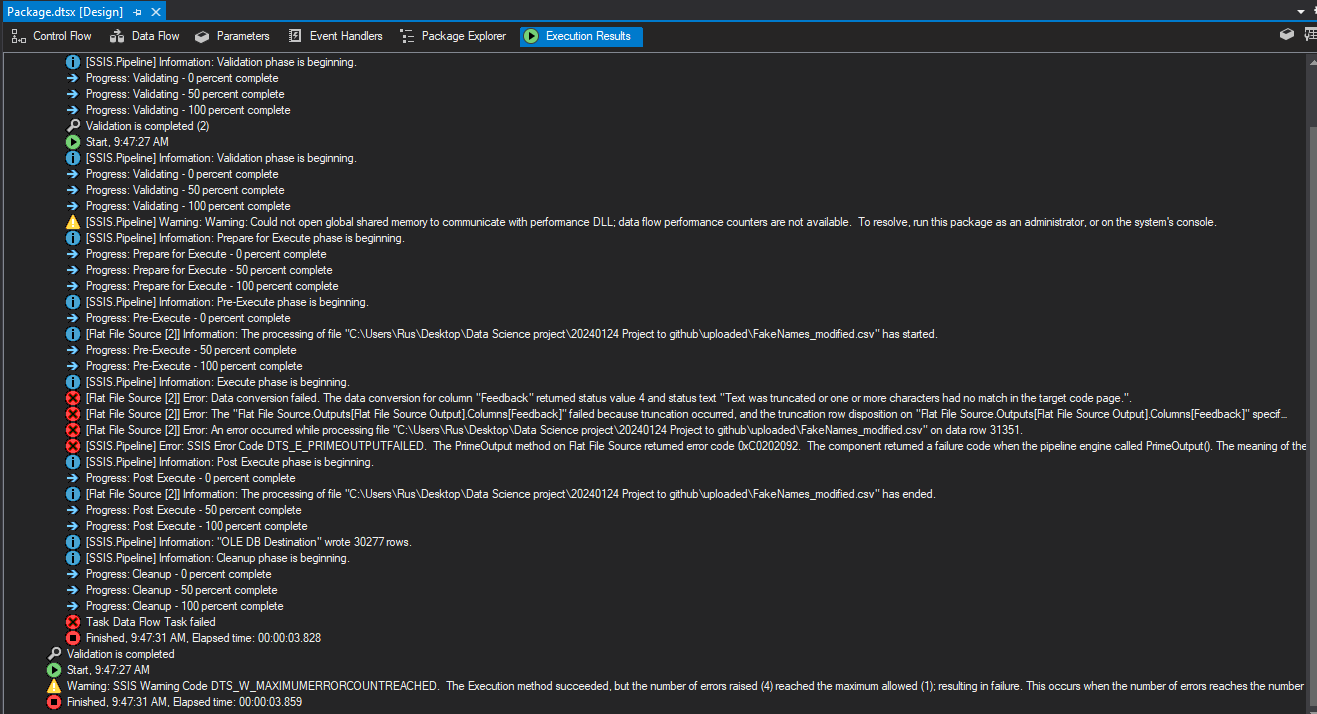
After the uploading process stops we can see details of the error if there are any.



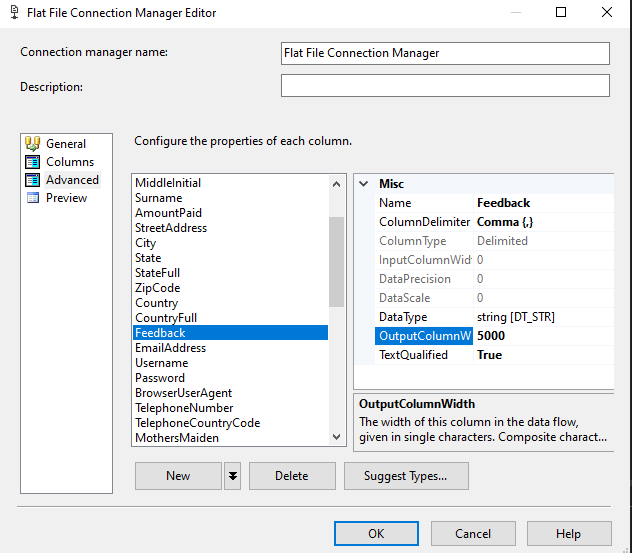
There were only 30277 rows uploaded, however, we have more rows than that. Therefore, we assume there is some error that happened while uploading.



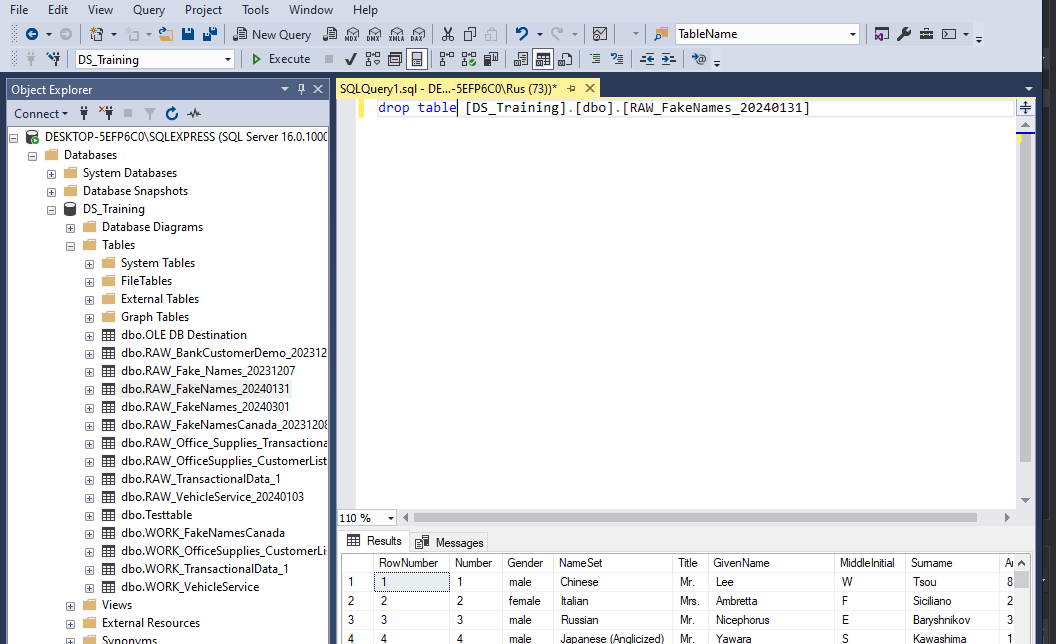
So here are the execution results:



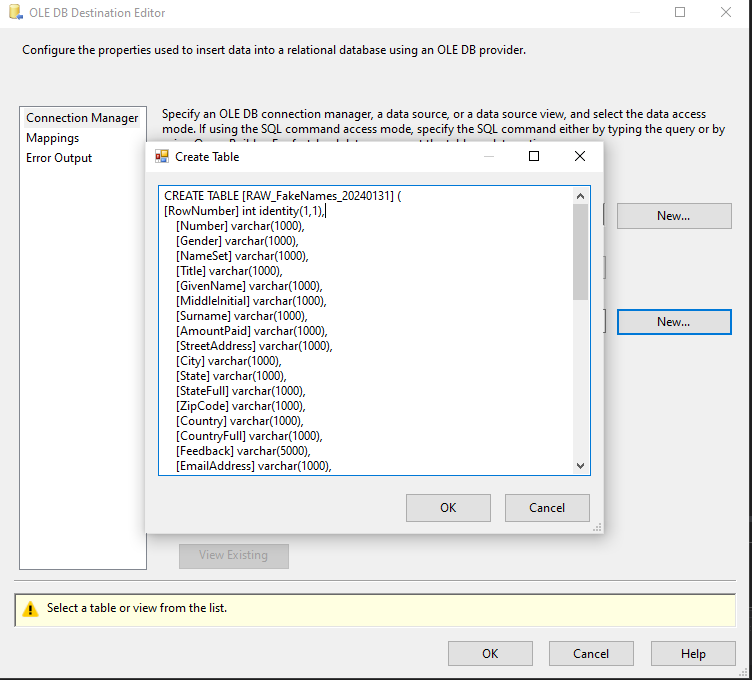
As above we can see there is an error on row 31351 due to the text column ‘Feedback’ and it should be corrected. After checking the CSV file, it is obvious that the text column ‘Feedback’ contains more symbols than we identified. Initially symbol restriction was 50, we changed it to 1000, however, row 31351 contains longer text than we expected. So we can choose an upper limit of 5000.



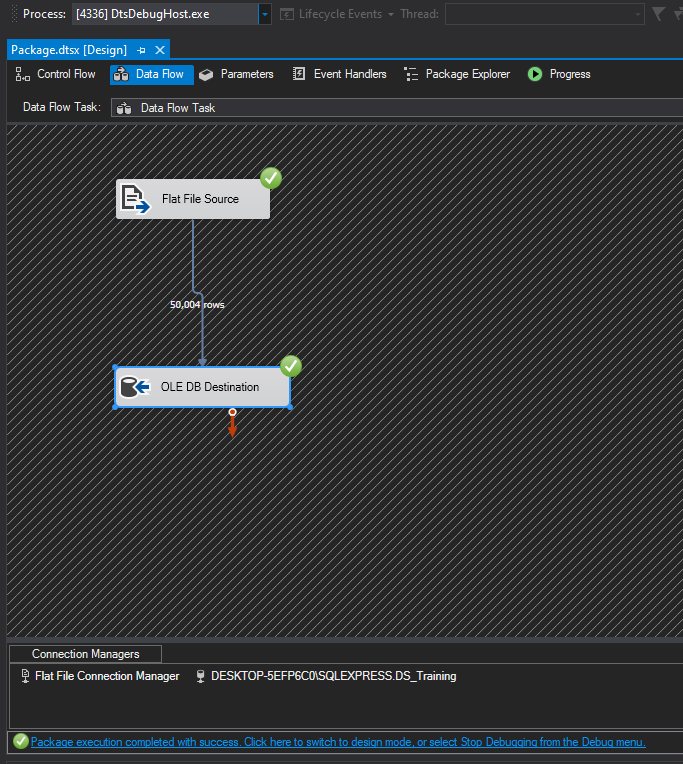
As the Destination Database was already uploaded halfway stopped on row 31351, the table that we have in SQL is not correct, so we should remove the table, otherwise the data might be doubled in SQL.



After the table was dropped we need to repeat the OLE DB Destination process.

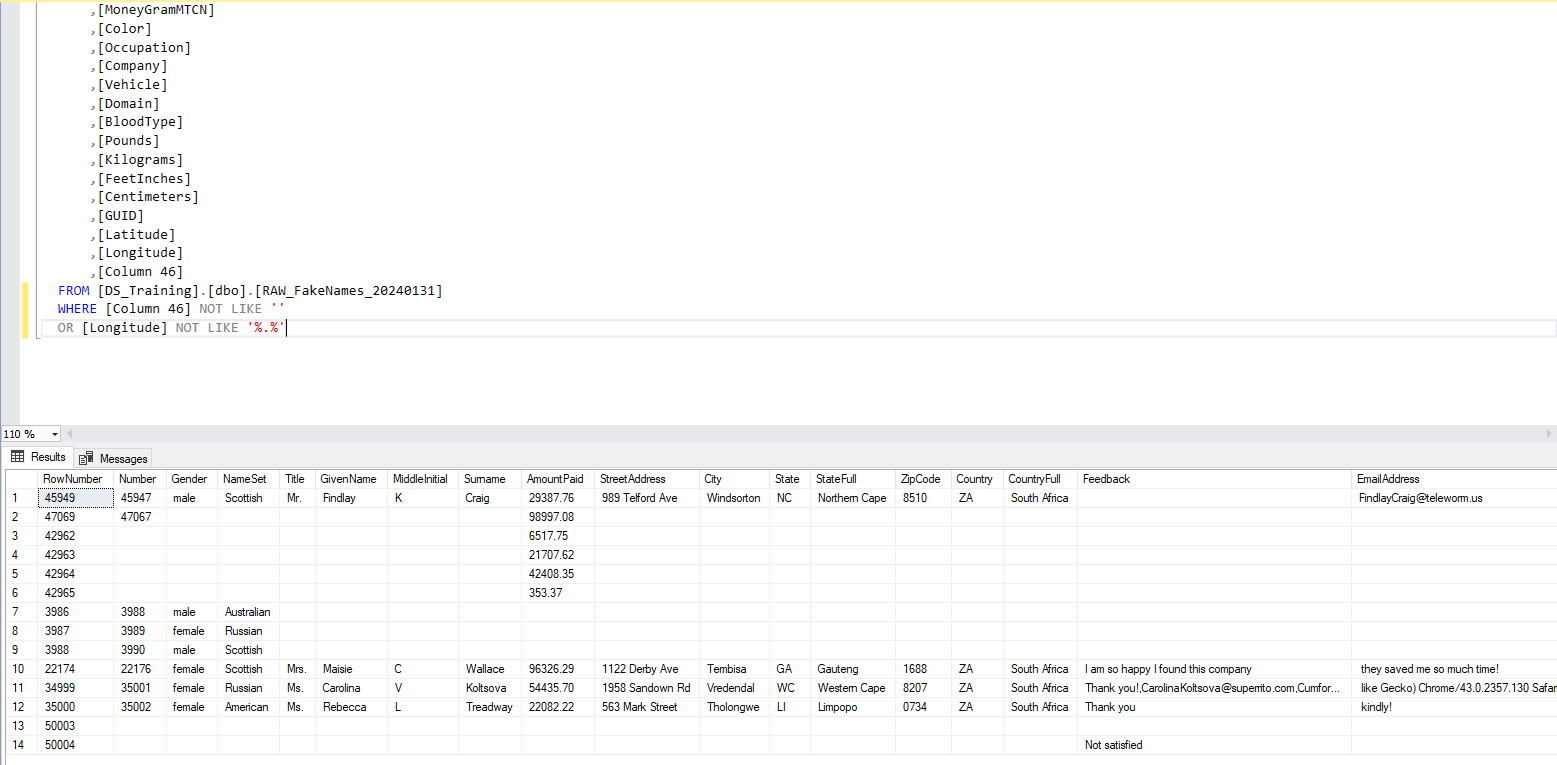


All rows are successfully uploaded as below:

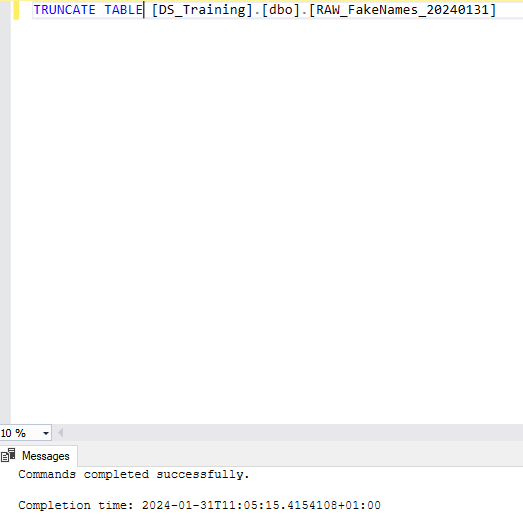


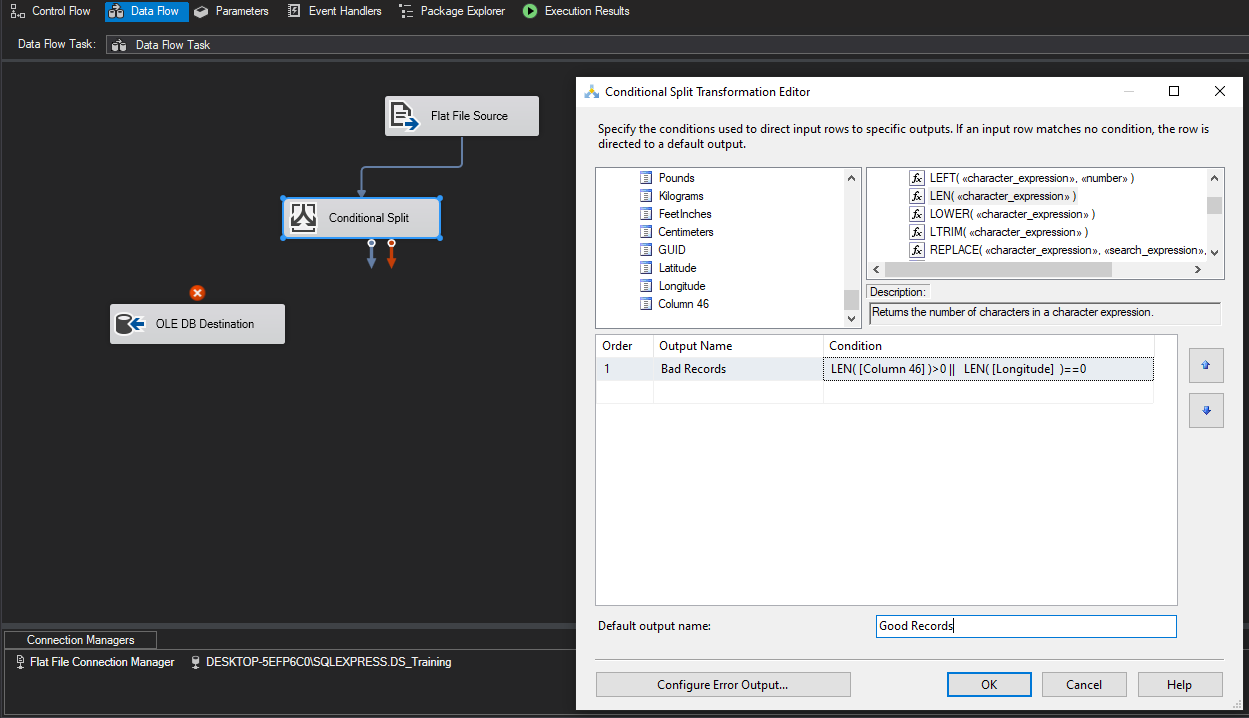
1. Finding anomalies in SQL and how to deal with them on the upload stage in SSIS

From here we can switch to SQL, as data was uploaded there already. However, we also need to find errors in SQL and correct them in the SQL table. In SQL we see that the number of rows is 50004 instead of 50000. We might expect the data to get skewed and it affected the number of rows. To check if there are any mismatches or shifted data we will apply filters in SQL to see them as below:

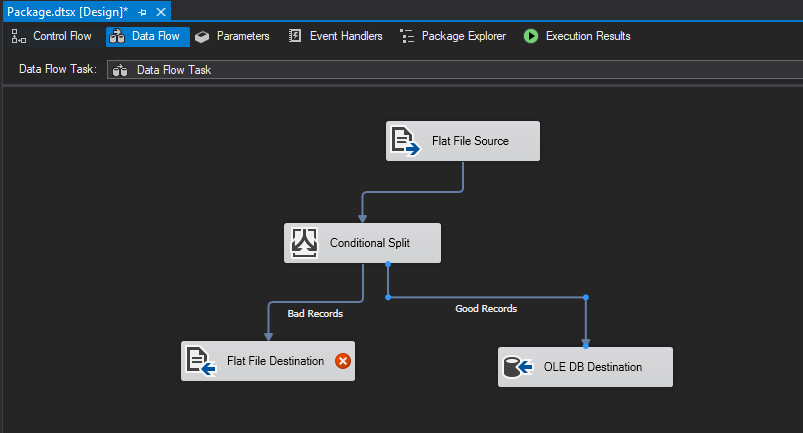


Thus we have 14 rows with anomalies. These rows can be removed or corrected in SQL or Visual Studio while uploading the file. It is more personal preference. There is an advantage in removing them while uploading in Visual Studio, as we can apply conditional separation of rows and automatically create a report with excluded rows. Before we will switch to Visual Studio for conditional split we need to truncate the table in SQL to avoid duplication of data.

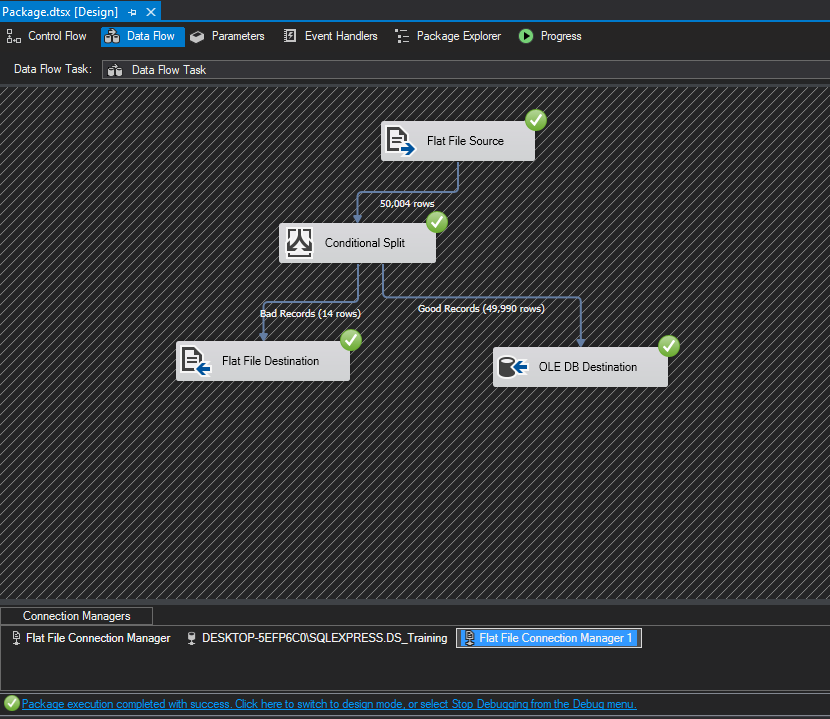




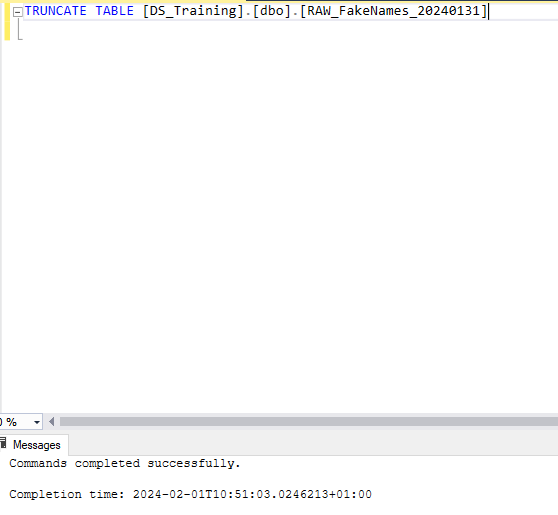
The conditional split is added as above. Records that are fit to the applied condition will be generated in Bad Records, while other data with no anomalies will be stored in Good Records.



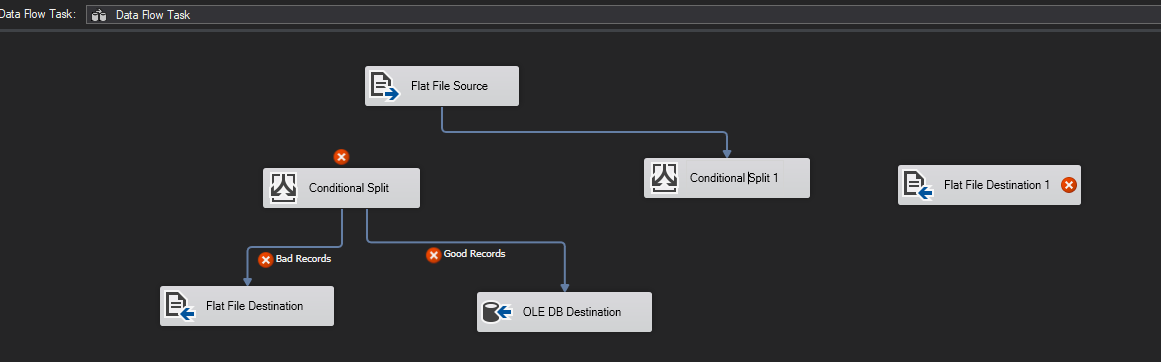
For flat file destination, we should choose the folder to be saved, thus records generated will be directly stored in that folder. After this step is done we can start the uploading and below its result:

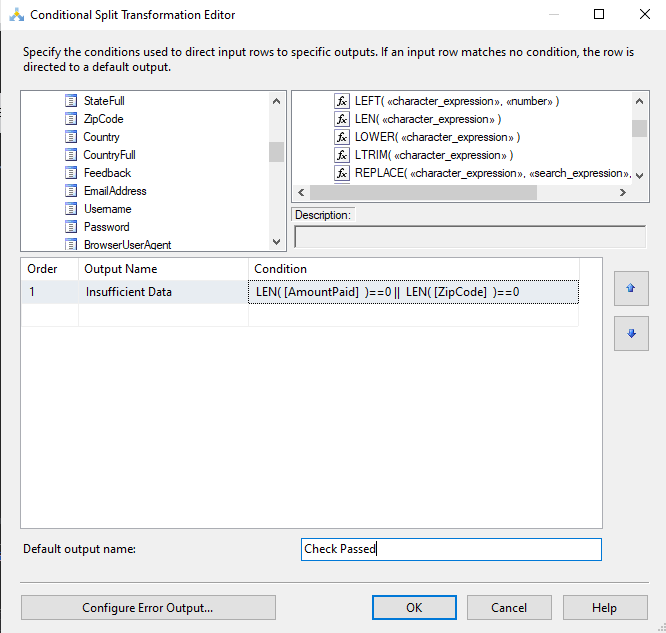


Data was successfully uploaded with the conditional split. If we want to add another conditional split to check if there are more anomalies to be excluded, we should first truncate the table in SQL before we start uploading the dataset to SSIS to avoid duplication in SQL.

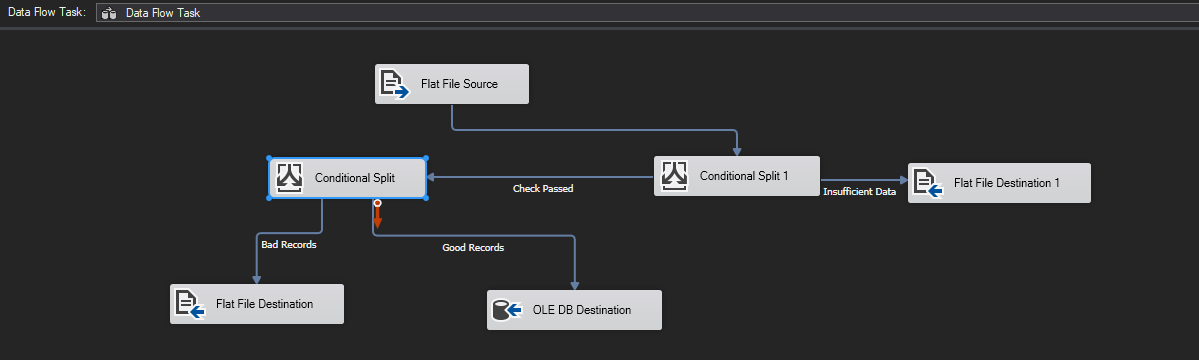


The following step is adding conditional split in SSIS as below:

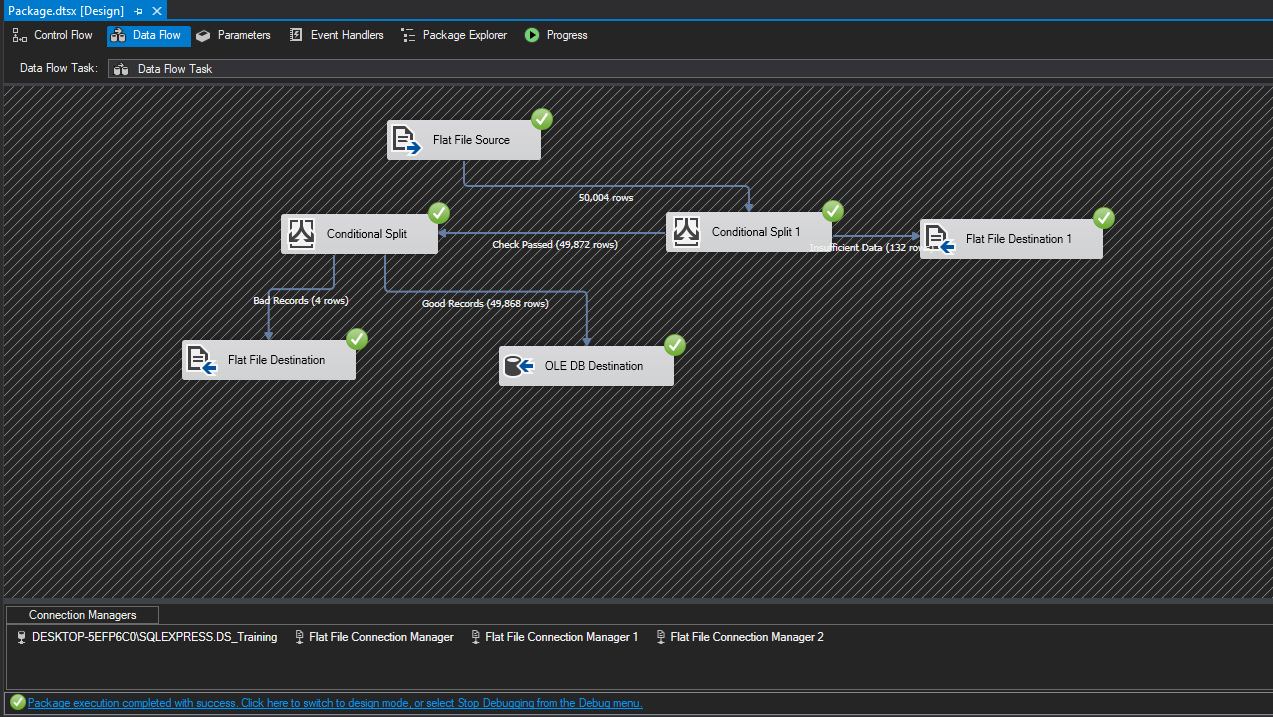




We added this split to exclude all the rows with empty cells in columns ‘Amount Paid’ and ‘ZipCode’, as they are important columns in making analysis, it does not make sense to keep these rows if they don’t have proper values to analyze. We will end up having one flat file Insufficient data, a record containing excluded rows and Check Passed output which will go to the second conditional split, generating Bad Records, and the final output will be uploaded to the SQL Server, where we can make further corrections.

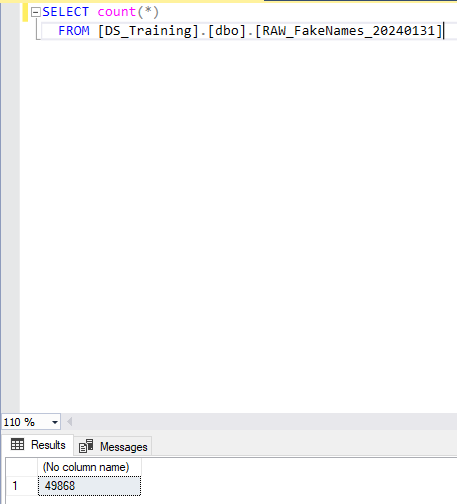


After execution, we can see there are 132 rows excluded as insufficient data and 4 rows selected as Bad Records.



**Data Cleaning and Preparation (SQL)**

If we check the Bad Records, we can see these rows include collapsed data which is related to the incorrect used quotation mark. They can be corrected in a preloaded flat file and to be uploaded again (The table should be truncated in SQL first). In case of Insufficient Data there is nothing to do with missed data, but only to contact the data provider and request the full data. To check if we truncated the table in SQL earlier to avoid duplication we need to switch to SQL and count the number of rows as below:

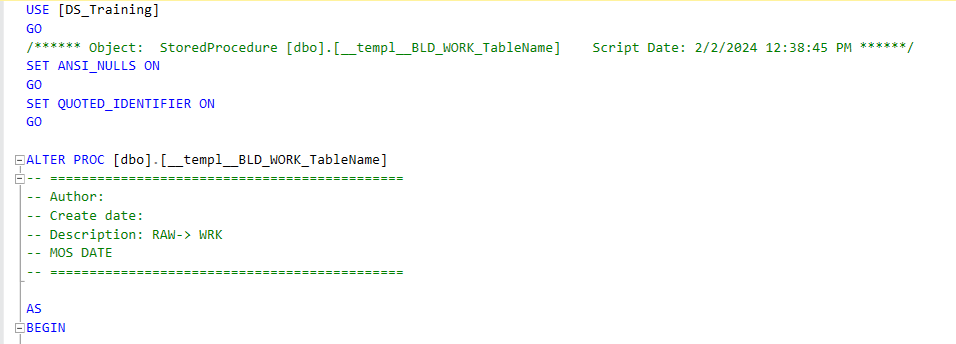


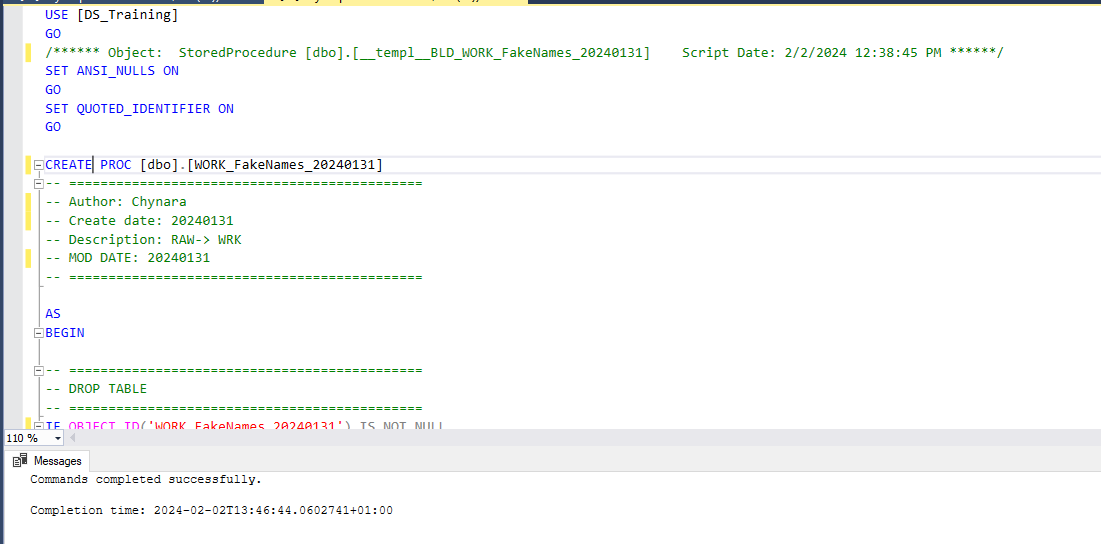
The number of rows in SQL and in SSIS are matching, meaning there is no duplication which will corrupt our analysis further.

1. Data Wrangling after the load.

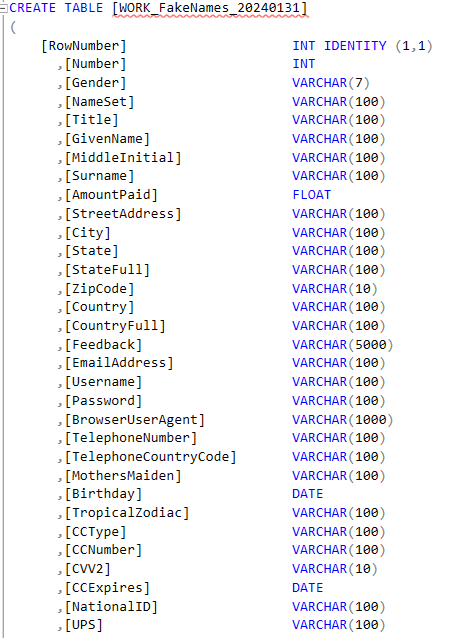
After data was loaded successfully we can switch to SQL to complete data manipulation. When all the corrections are done in SQL and the file becomes a WRK table, we can further use this data for Data Visualization in Tableau or Power BI, and for Machine Learning in Python.

To track the changes in data that we are going to implement it is recommended to create stored procedures, namely procs for saving the codes and then being able to implement them whenever we need the data, it is also easier to track what kind of changes were done, in case of teamwork, who made the changes what the purpose was. In our case procs was already created and saved and we can use the template of it as below:

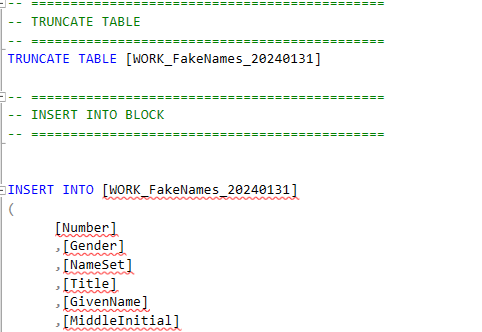


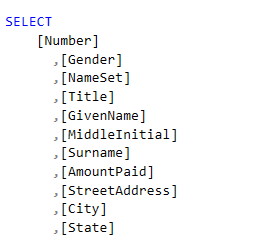


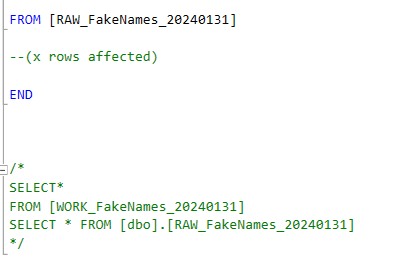
We created the proc and table to make the RAW dataset to WRK dataset as above using the following steps:



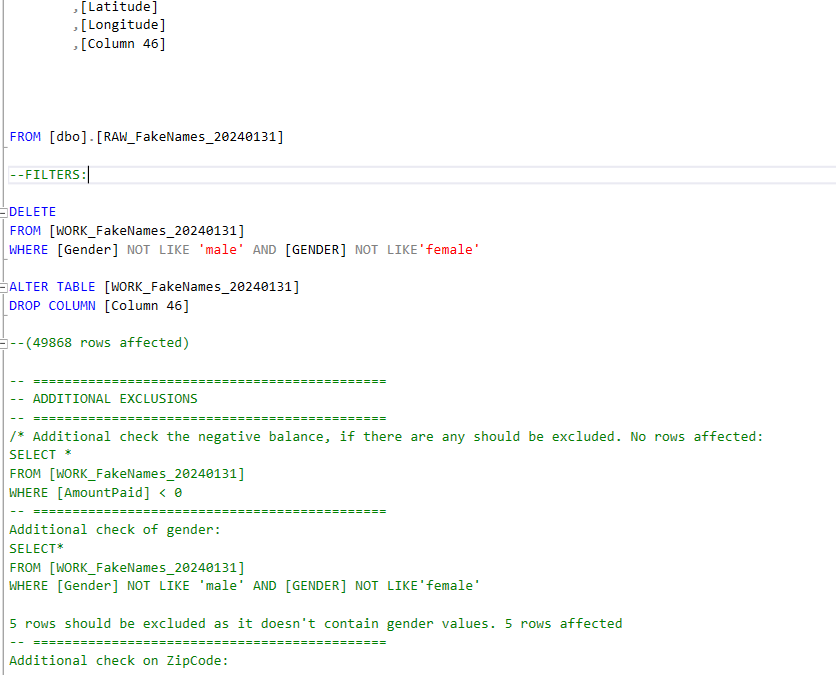
The table was created using the column from the uploaded file, then we truncated the table just in case to avoid duplications and inserted the dataset from the RAW database to WRK:

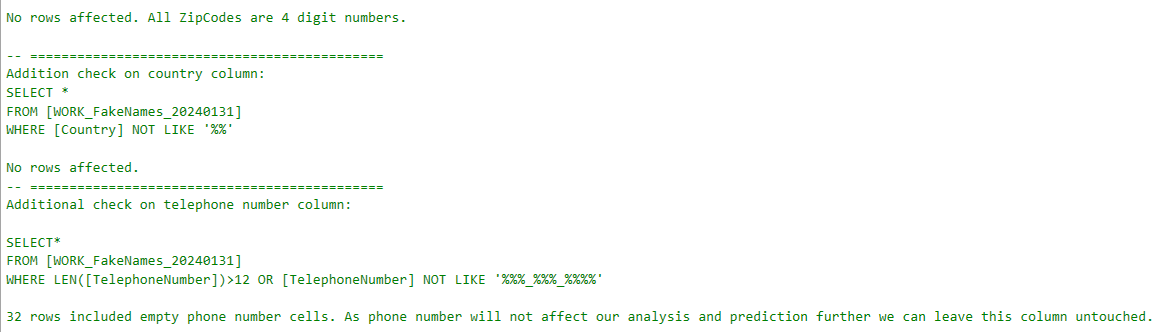


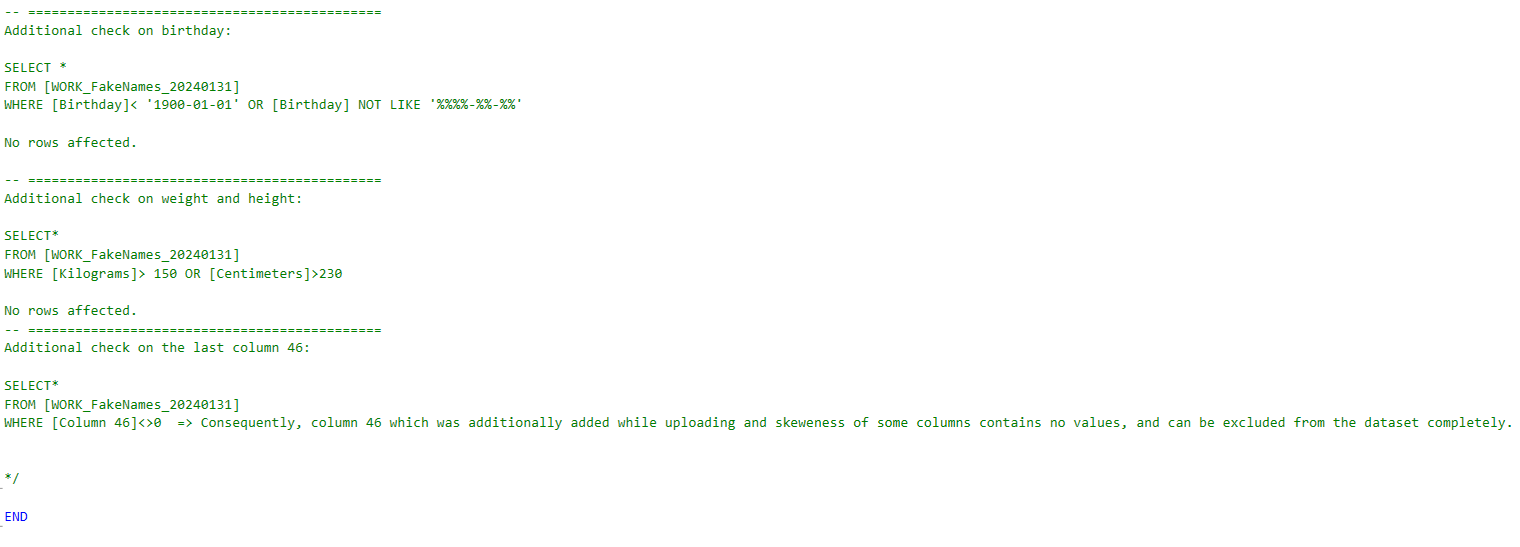




Thus proc was used to create the work table. After the Work Table is prepared we can add some of our filters and make an additional check on particular columns before we will analyze them in Tableau and Python. Here is the checklist:



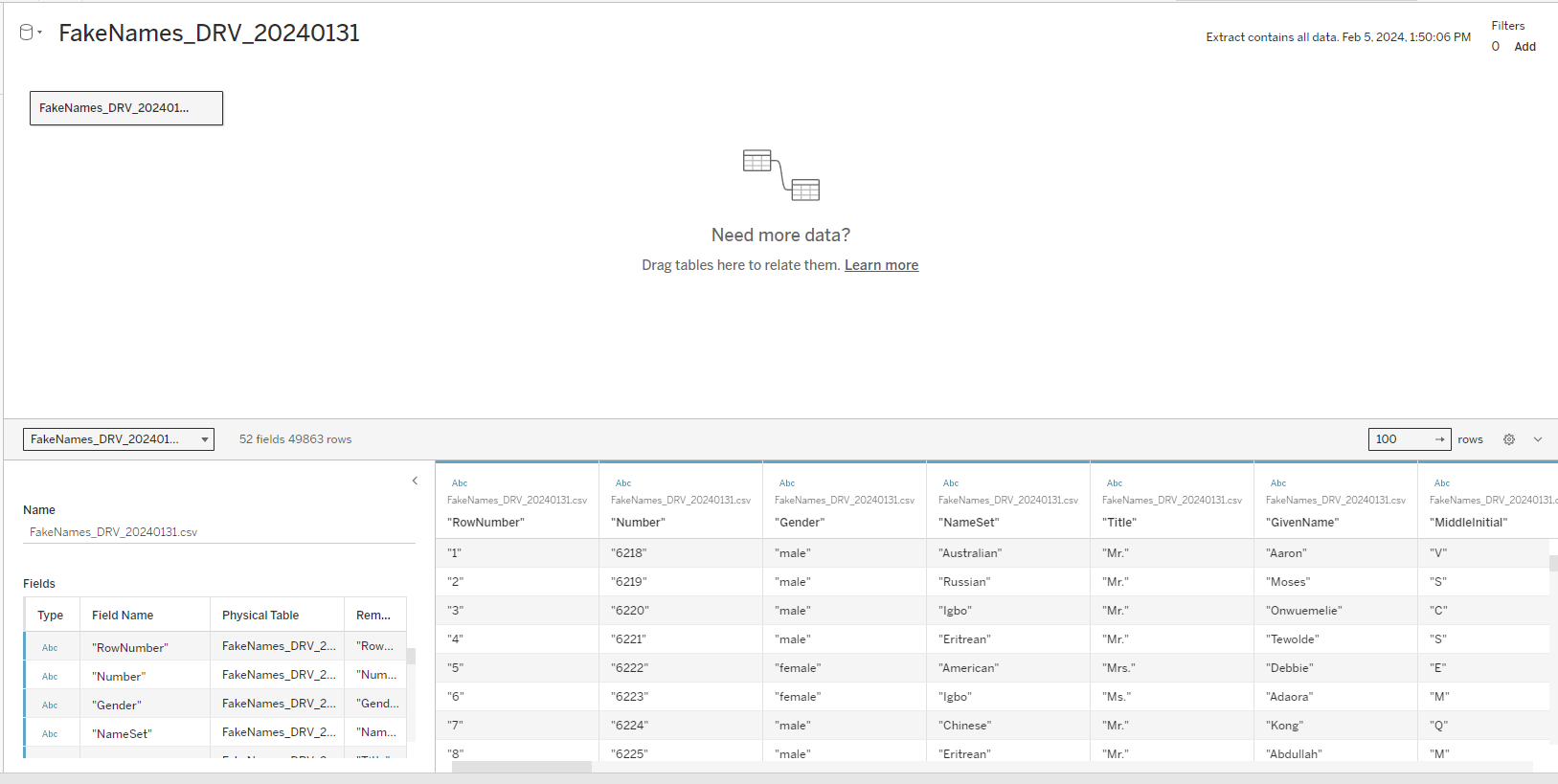




Now we can save the WRK file for our future analysis. After filters are applied we can save the table as a csv file.

**Data Visualization (Tableau)**

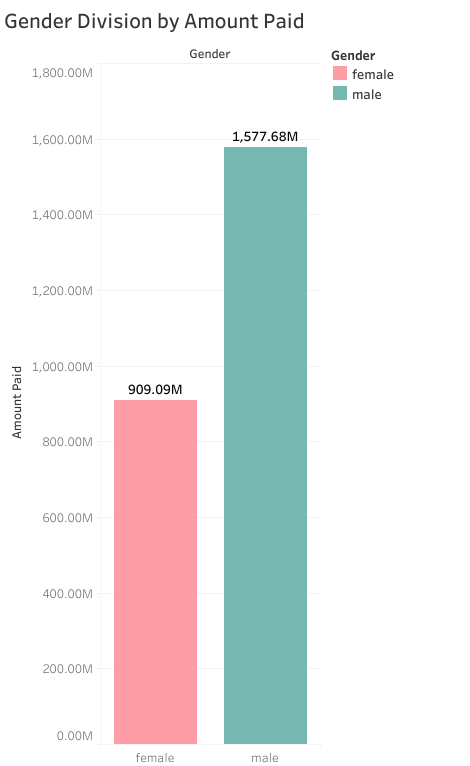
The ready table can be used for Tableau visualization for data mining.



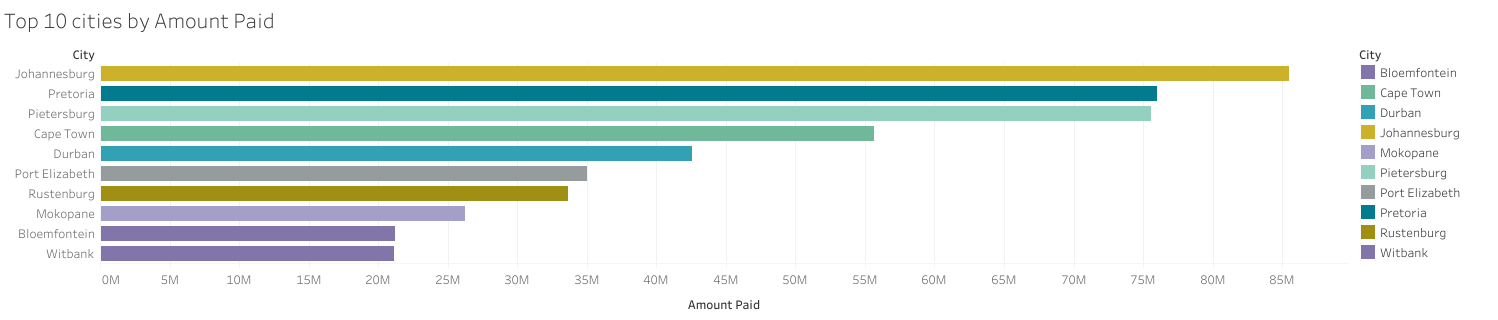
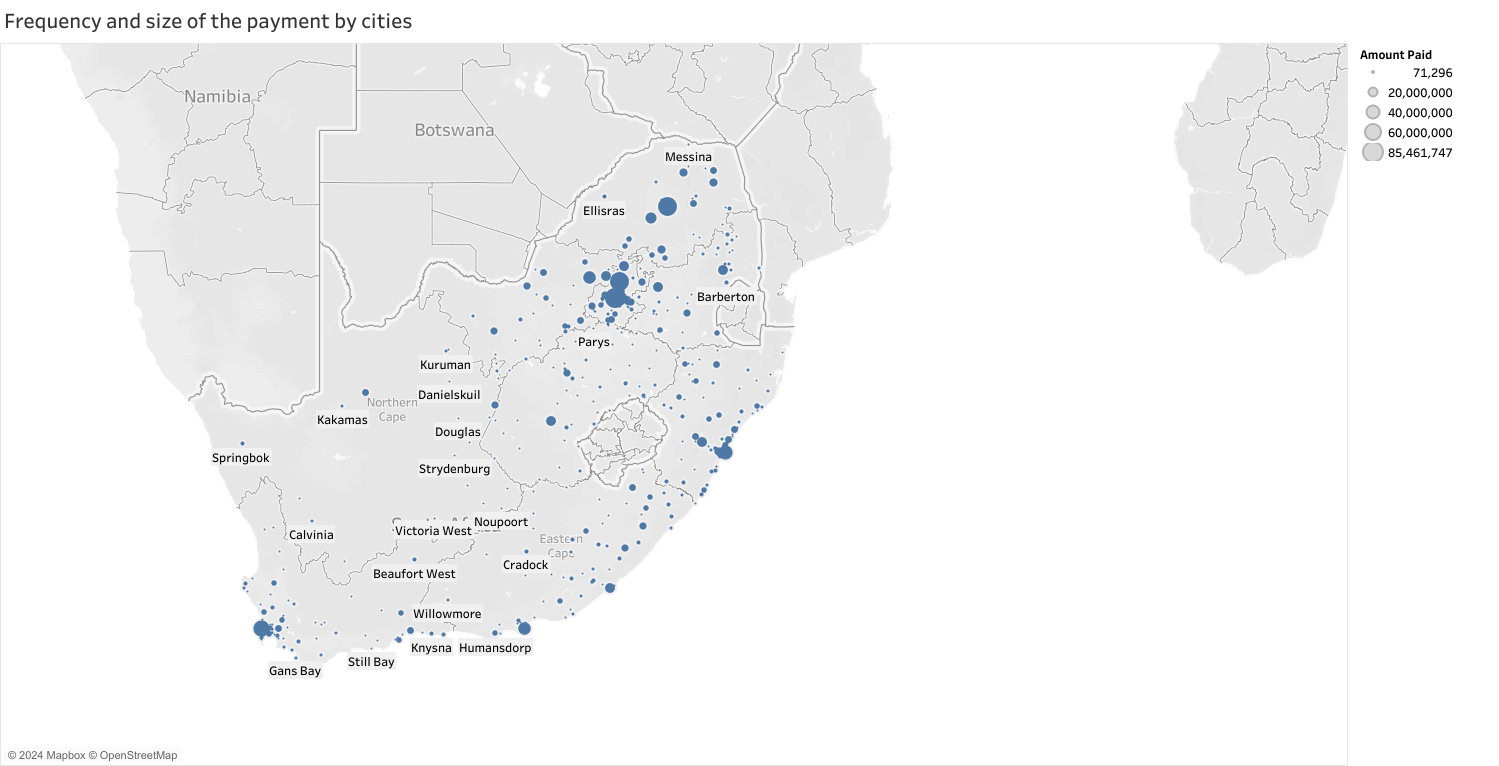
After the file is uploaded to the Tableau, we can display the plots of values we needed as the following:

Our first visualization will be about the Gender division by Amount Paid as below:

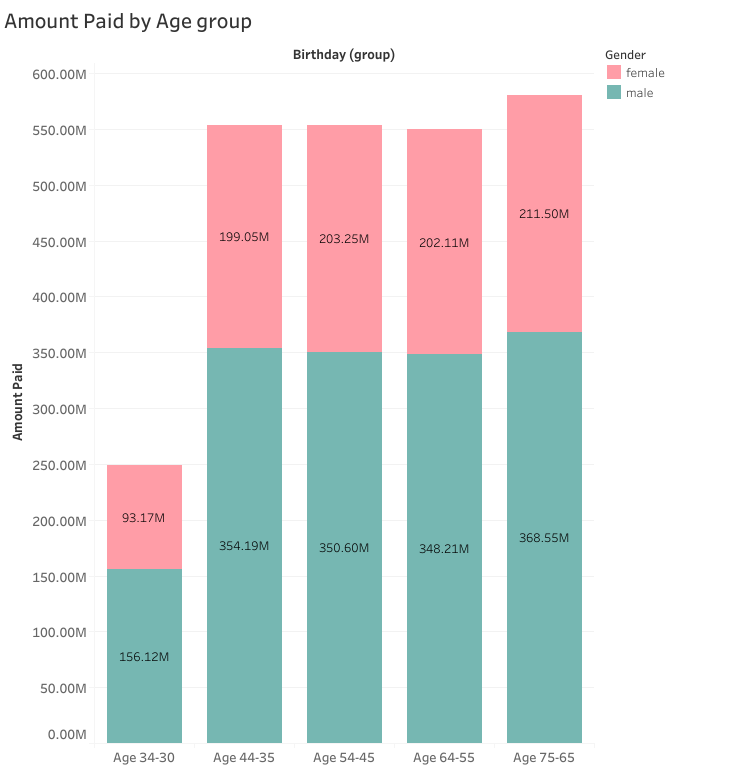
Here we can notice that the sum of the amount paid is higher for the male group. While the sum of the amount is 909.09 million for the female group.



From the following map plot, we can see the frequency and size of the payment by cities. The bigger the size of the circle the higher the amount paid. Taking into account the next histogram for the top 10 cities by Amount paid we can conclude that Johannesburg, Pretoria, Pietersburg, Cape Town, Durban, Port Elizabeth, and others are the cities where the amount paid was the highest. For Johannesburg, this amount reaches over 85M.

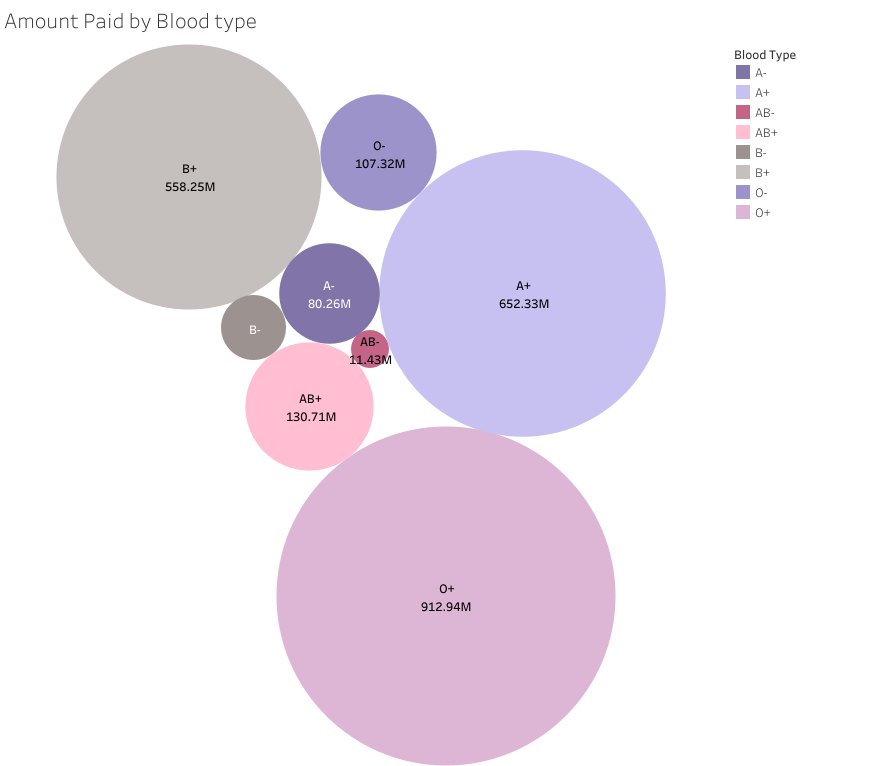


The next histogram shows us the amount paid by age group and gender, which was manually split between 34-30, 44-35, 54-45, 64-55, and 75-65 age groups as below. The mentioned histogram helps us define that the age group 65-75 is the age group with the higher sum of the amount paid among other groups, and the age group 30-34 is a group with the lower sum of the amount. If there will be a policy to be changed to increase the amount paid by clients, the aim group should be 35-75 age slice and male gender.



Following the next bubble chart, we can split the amount paid by blood type, the bigger the bubble is the higher the amount paid. As in the chart, the blood types O+, A+, and B+ have the highest amount paid.

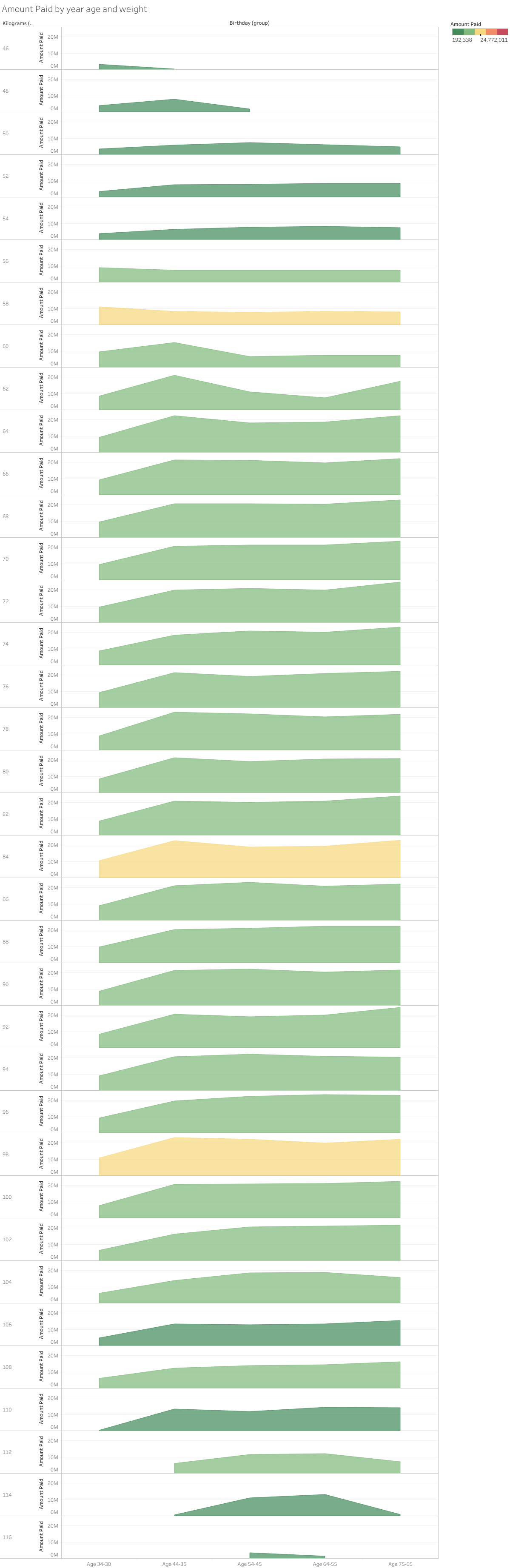
It is hard to conclude if there is a real correlation between amount the of blood type or if is it just a coincidence. If there is a correlation further investigation can be done toward this approach. In the service industry, it might be difficult to obtain information about the blood type and often it is not included in mane research. However, it might be very useful in the medical field. Promoting medicines or medical services considering the blood type is essential in this field.



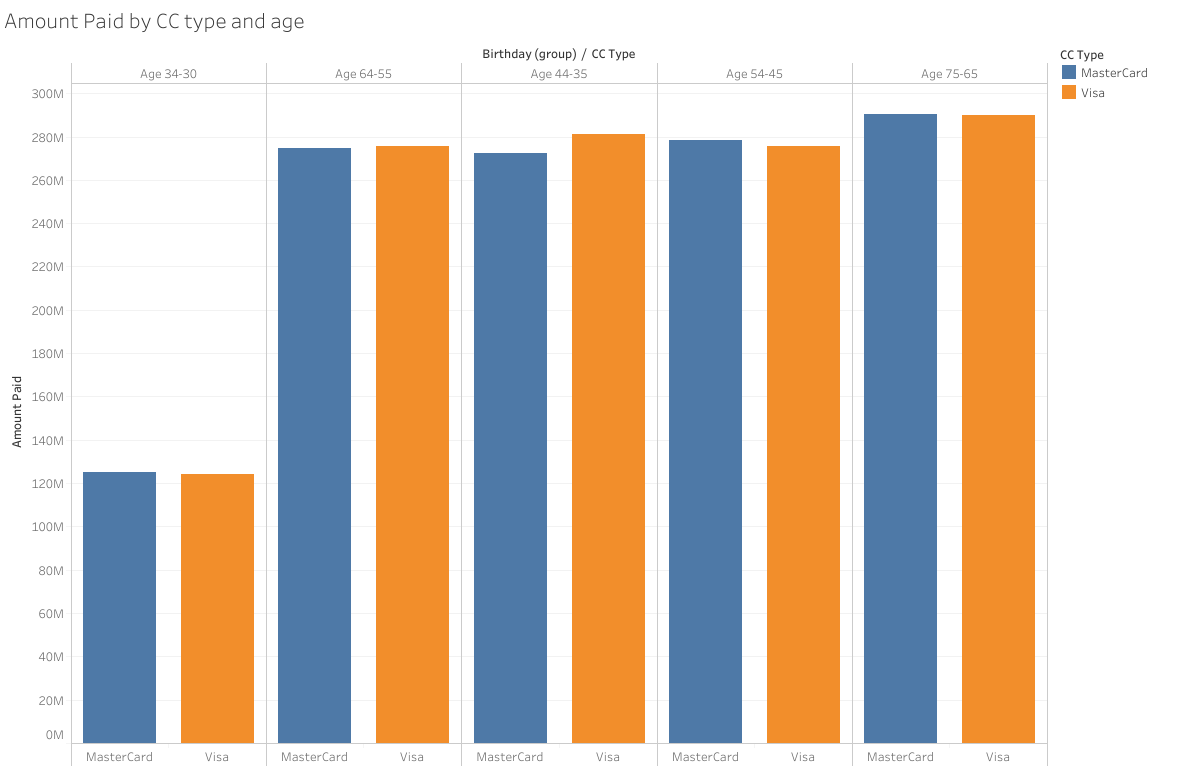
The following observation helps us to display the range of clients by their heights and to see what height group is more tend to visit café. As we can see height groups 165, 170, and 175 have the highest amount paid which is over the 350M, while other groups have lower than 350M. Also, we can see a gender split where the males tend to visit café often as by amounts paid. Another point here is the height difference between genders. Females tend to have lower height than males.



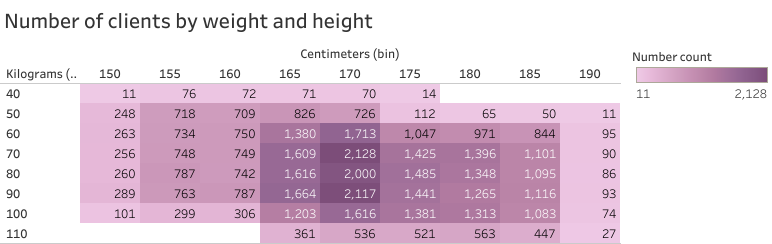
Below we can see a chart with weight, age group, and the sum of the amount paid. On the left side, there is weight in kilograms, on the bottom age group split, and the area inside shows the amount paid painted with a green-red color range, where green means the lower sum of the amount paid and the red is one-way opposite. By the chart below we can conclude that the 58, 84, and 98 kilograms groups ranked with the highest amount paid and mostly prevailed with the 44-35 age group.



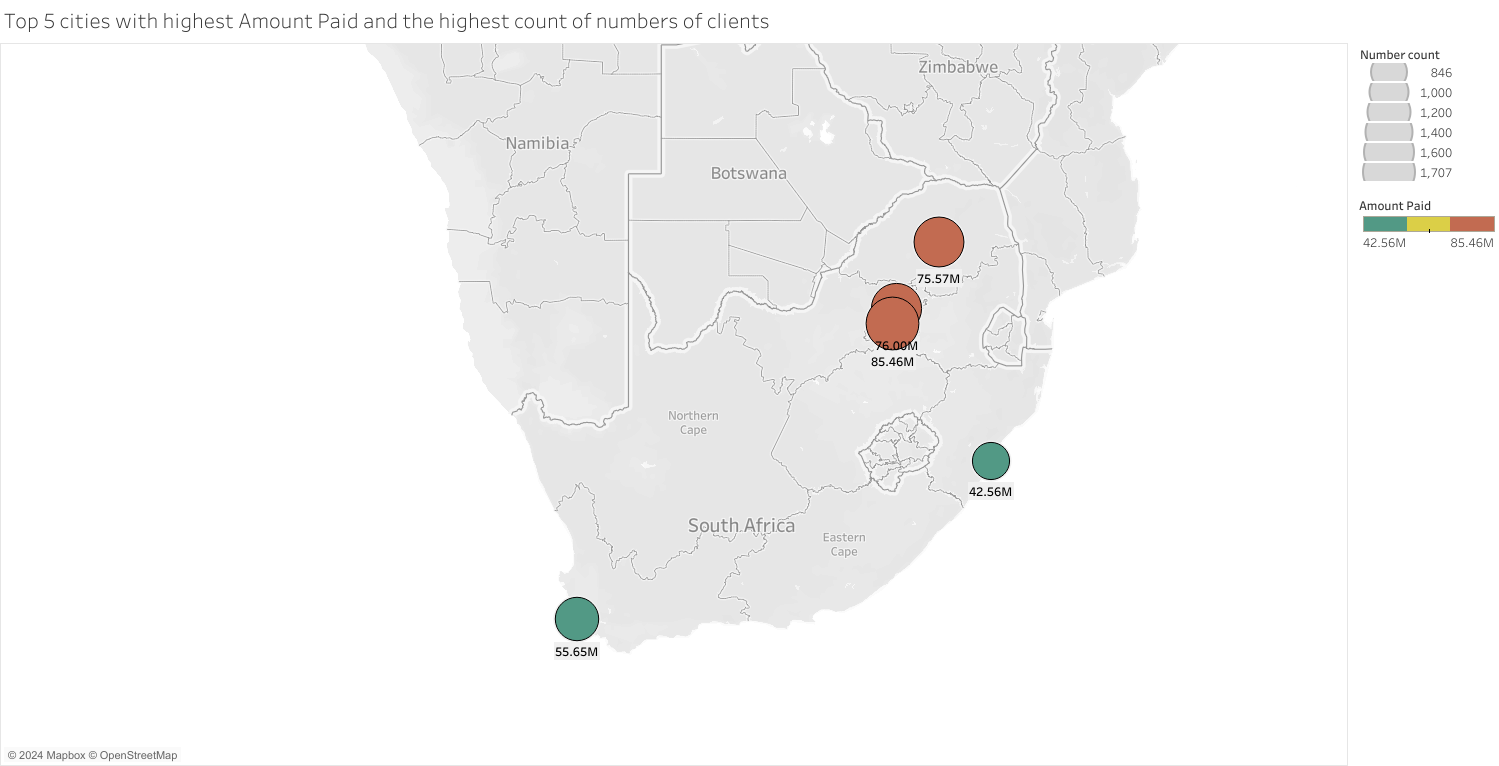
Amount Paid by CC type and age. Below bar chart below monitors two types of cards that were used while paying the bills. The differences between those two cards are not so big, almost no difference, but what is noticeable is the age group again, as was already shown before. There is no difference among the age groups, showing which group prefers which card while paying.



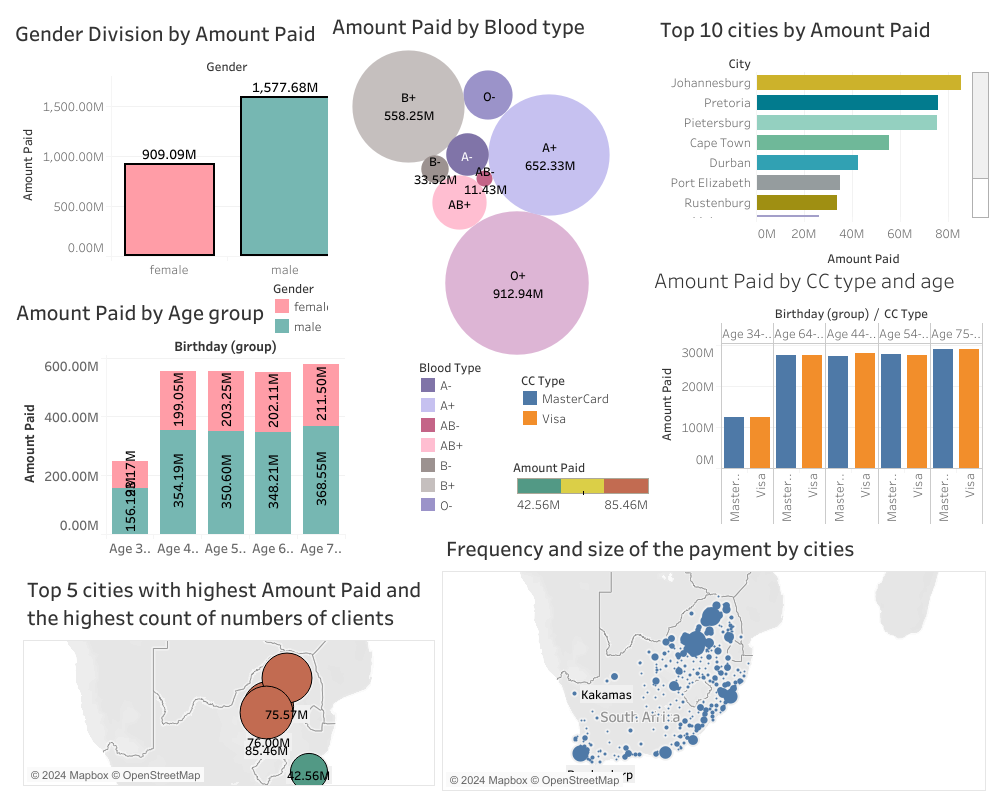
The next mapping is about the number of clients by weight and height. It demonstrates that clients usually have a weight range of 60-90 kilograms and a height of 170cm, and the sum of the amount paid in this group ranges between 1713M-2128M.



Below is shown geographical mapping of the top 5 cities with the highest count of numbers of clients. As the frequency of visiting the café in these cities is highest at the same time we can see the sum of the amount paid ranked by colors where brown is the highest among those cities and green is the lowest. We can conclude that despite the fact that there are many clients in some cities, the total bill and amount paid consequently might be lower, while there are other cities which have the amount paid, and total bill respectively higher no matter of count of clients.



And as the conclusion to all the tables shown before there is a good option to create the dashboard using Tableau as below. Which will show us all the tables on one page, moreover dashboard is very useful in filtering. Filtering can be done on the same page, and the dashboard will recalculate automatically and show the respective values taking into account filters.



<https://public.tableau.com/authoring/APFULL/Weightandheight/FakeNames%20Dashboard#1>

Dashboard with worksheets saved using the link above.

**Prediction with ML using Python**

In this section of the project, we will consider the prediction of the amount paid as the target value using Python libraries.