**Database and Analytics Programming Project:**

Project Journal drafted by Polina Prinii – x21137757

## **Fri, Oct 29th:**

Team meeting held with a duration of 30 minutes to pitch ideas for project topic. Proposed to cover Covid-19 on a country basis with the aim of analysing impact/spread/vaccination rate and so on.

Team agreed on the idea and countries were assigned to each.

Discussed project objectives such as database and programming language requirements. As well as kicked around the idea of potentially placing our combined data analysis on a dashboard hosted on a webservice, dependant on time available to complete project.

**Thu, Nov 4th:**

Research conducted around possible dataset to use for project. Duration of about roughly 2 hours.

Identified an appropriate dataset by Our World Data found here: <https://github.com/owid/covid-19-data>

**Fri, Nov 5th:**

Team meeting held. 30 minutes duration

Discussed the following:

* Journals and the format to submit in.
* Questions around which we required additional clarification from the lecturer.
* Understanding the question our dataset is looking to answer.
* Potential use of AWS
* The challenges behind working locally with our VM's, specifically towards the end of the project where we are looking to connect all 4 DB's for comparative analysis.

**Fri, Nov 12th:**

Team meeting held. 30 minute duration

Reviewed feedback from lecturer. Conducted a pivot from original theme for the project as clarification around the nature of datasets was received. Understanding is that each dataset must be unique therefore, the use of one dataset split on country basis was a no go.

Alun suggested to stick on the topic of Covid-19. Requested I cover the area around vaccinations.

Reviewed deadlines via Gantt Chart and set a soft deadline of the 5th of Dec for the Code part of the project.

**Mon, Nov 15th:**

Git repository created and invite’s extended to the whole team. This took about 30 minutes in total to set up.

**Tue, Nov 16th:**

Held a Quick Git Demo to get the team familiarised with GitHub. Covered areas such as creating folders and files, committing and pulling to Git through Desktop and the importance of the ReadMe file. Call lasted about 30 minutes.

**Thu, Nov 18th:**

Refined my section of the Git repository.

Deleted any test files and or folders. Created a README file which outlines the project objectives along with the aim of analysis for the chosen dataset.

Created a Word file to compile the Journal and updated the journal. All changes are reflected in the History section on Git Desktop or commits on git web.

Overall activity took about 2 hours.

**Fri, Nov 19th:**

Commenced project by setting up appropriate environments for the project via PyCharm the chosen IDE to write code and Oracle VM VirtualBox Manager.

Challenge encountered around the setup of the Python main file which is to be responsible for all project code.

As the chosen IDE is PyCharm, the project creation in PyCharm required for the environment to be specified. For PyCharm to connect to my Virtual Machine it is first necessary to have Python installed.

Having identified the correct steps to for installation Python 3.9 was installed on the VM.

Next challenge presented around installing pip package on the virtual machine as it is a necessary pre-requisite to connecting PyCharm to VM.

Work commenced @ 5:40pm and concluded @ 7:30pm. Work resumed @ 9:15pm.

Continued to attempt connection to VM from PyCharm upon which stumbled across my next challenge. Initially the overall understanding around env setup and connection to the VM was incorrect.

It was understood that both the latest version of Python and pip was to be installed on the VM which would be the environment (to be specified in PyCharm) in order to connect to the IDE located locally on the machine. This was incorrect.

The correct understanding or in other words process for env setup for the main.py file (to hold all python code) was to create the correct environment in PyCharm (in my case “virtualenv” was selected) which was created in the git repository stored locally, ultimately setting up the interpreter. From there within the IDE the main.py file was created. The main.py file will connect with MongoDB and PostgreSQL which are hosted on the VM.

Work concluded @ 10:50pm.

**Sat, Nov 20th:**

Work commenced at 2:20pm.

Today I am working with the main.py file. The aim is to ingest the files which are stored on the Git repository managed by Our World in Data.

For learning purposes, I have chosen to work with .csv files in order to convert it to JSON and then using the JSON files import the datasets into MongoDB.

Additionally, for contingency I have cloned the git repository from which my data is derived in the case the repository is deleted.

Issue encountered around editing the main.py fail, kept receiving the following error: “This file does not belong to the project”, additionally kept getting an error when trying to import pandas package.

Believe it might be an issue with initial set-up with the interpret and the project env.

Decided to uninstall PyCharm and start from scratch.

Identified the root cause was due to the project environment being setup in a folder which was not empty. Solution was to create a “Main Project” folder which would hold the main project drafted within PyCharm.

PyCharm works with empty directories aka folders when creating projects however, since I created my Journal and a README file in my project folder before setting up the PyCharm env, the project creation was experiencing issues.

Project space in PyCharm created and successfully synced with Git.

Work to be continued later, concluded at 4pm.

**Sun, Nov 21st:**

Work commenced at 2:40pm.

Continuing work which was initially intended for yesterday. Though to a few issues with PyCharm it was postponed. Today’s aim is to ingest the chosen datasets through Python. I’ll be using the pandas package to read the csv files directly from GitHub.

Found out the hard way that PyCharm does not automatically install the extensive packages available to Python like Jupyter Notebook.

Additionally, found that PyCharm for some odd reason limits the print output of a data-frame to two columns and 10 rows. To get around this I must use pd options around display to show more than 10 rows and 2 columns at a time. Having applied the pd options to expand the display of columns and rows, it would expand the view of the columns and not the rows. Unsure if there is some issue around the rows but the pd option won’t apply to rows.

For this reason I will just perform a number of print functions to validate if the data came through okay, such as print columns values, to show columns names, a count function to see how many rows and so on.

Included exception handling to cover the following cases:

* No Internet
* Any other possible errors.

Additionally included an if statement allowing the user to pull the data file from local machine in the case the code is being ran outside an internet connection.

Lastly included the additional two dataset around manufacturer of vaccines and age groups as well as cleaned up the code a little.

Tomorrow to commence converting to JSON. Work concluded @ 9:45pm.

**Tue, Nov 23rd:**

Work commenced at 7:40pm.

Today I’m working on taking the declared csv files using Pandas and converting them to a JSON object (file) which will be stored in my Git repo. I’ll be doing this convert in a form of a function additionally, I’ll be including a clause which will be replacing the JSON files on each run of the code as we are working with Git repository at the source dataset and said datasets undergo occasional updates which I want to reflect in the project.

Successfully manged to write a function which conducts the necessary csv to json convert and stores the .json files in a separate folder in Git. Additionally, manage to make the JSON file pretty from the beginning by using the indent clause in the .write() function of pandas.

Next, I move to the clause around the files to be replaced each time the code is ran. This not only helps in testing purposes but also allows me to work with the most up-to-date data in the datasets as I’ve specified the pd.read\_csv function to call form an URL (git repo).

Successfully implemented the replace functionality using the OS package in Python.

Next, I’ll be moving on to importing the three datasets into MongoDB for transforming the data.

Work concluded at 10pm

**Wed, Nov 24th:**

Team meeting after receiving some feedback from our lecturer. For roughly an hour the three of us (Alun, Maria and me) were trying to work with the feedback we received in structuring our big question and how the four of us as a team would tie our work together.

We have agreed to perform a Multiple Linear Regression using our datasets in order to evaluate if the chosen x variables can be stated as good predictors for mortality numbers. The analysis will look at a global view.

The aim of the group work is to prepare a master dataset which is a combination of our chosen datasets after we have performed individual necessary transformations and analysis to better understand out data.

The report and group work will focus on Multiple Regression.

**Thur, Nov 25th:**

Work commenced at 7:40pm.

Today, the aim is to update the README file in my folder outline the group decision for Multiple Regression and how my work aids the preparation for regression analysis. Additionally, I will aim to transfer my raw data types to MongoDB for structuring.

First challenge experienced when attempting to connect to the MongoDB instance on the VM to the Robo3t GUI, had a few challenges in setting up the connection correctly. In the end was a case of unchecking the Authentication section as no authentication is required.

Successfully connected through Python to MongoDB using pymongo package. Imported the three JSON files into MongoDB for storing. One issue encountered in the form of that when the code is executes each time the MongoDB collection get re-populated meaning the data just keeps populating even though it already exists.

Tomorrow’s challenge is to identify a way to either clear or drop the collection before it is populated to ensure there is no duplication.

Work concluded at 10:45pm.

**Sat, Nov 27th:**

Work commenced at 2pm.

Cleaned up the code on both .py files. Introduced for loops to reduce the number of code lines as well as repetition code as I am dealing with 3 dataset files.

Added comments and updated the git repo.

Additionally, added the 3 datasets which are being pulled from local machine to git for easier reference when pulling. Will need to note on project that in the case the pandas package fails to read from URL the path to the hard files needs to be edited to reflect the user currently reviewing the code.

Work put on pause at 4:10pm. To resume by creating a Country name and code table for relationship purposes.

**Sun, Nov 28th:**

Work commenced at 5:15pm, concluded @10pm.

Continued last night’s work by creating a new collection to hold all country names and their respective codes. Which I’m aiming to use as the main relations table in PostgreSQL.

Had a few challenges around dropping the collection as I knew the code would be executed multiple times. Issue was around the == operand. I was stating for the table to drop if == 0, when the correct operand to use was >.

Collection successfully created the collection and populated it using the already present data in the DB, this was achieved through the use of distinct() function and aggregate() function to allow to pass multiple values to return unique records for.

**Tue, Nov 30th:**

Work commenced at 10:30am and continued all the way through to 10pm, with several breaks in between.

Today’s task involved creating the master collection aka table in MongoDB, later this collection will be used to merge the team’s datasets into one dataset to perform Multiple Regression.

Had a bit of a struggle around the maths and the understanding of the datasets at the working hand. I was aiming to calculate the total vaccines administered by country as well as total boosters administered and the total number of people fully vaccinated.

Didn’t realise or more so understand that the data for total vaccinations was an accumulation of previous day and next day and so on. With that I struggled in understanding why I was being returned a value in billions rather than millions when looked at a by country basis. Used Ireland as a reference to understand the maths. In the end it was a simple solution of just pulling a distinct value by location for the specified date of the 24th of November (cut off point decided as a team for the data). Bellow is the code I was typing up for python using Robo3T.

db.df\_1.aggregate(

[

{ $match :

{

date:{

$gte: "2021-01-01",

$lt: "2021-11-24"

}

}

}, {

$group :

{

\_id : "$location",

Total\_Vaccinations\_Administered : { $sum: "$daily\_vaccinations"},

Total\_Boosters\_Administered : {$sum: "$total\_boosters"},

Number\_of\_People\_Fully\_Vaccinated : { $sum: {$divide: ["$people\_fully\_vaccinated", 2]}}

}}

]

)

Next challenge presented in the form of working with obsolete data, in essence I was working with old data and had forgotten to perform a refresh on the csv files stored on my local machine. A blunder on my end. However, once figured out this explained the discrepancies in the data between the csv files and data within Mongo (as it is being directly pulled from the git repo)

Successfully, created and populated the new collection which holds the information outlined above.

Next steps include creating a connection to PostgreSQL, and importing the data from MongoDB for further transformation and analysis. With the goal of multiple regression analysis.

**Wed, Dec 1st:**

Work commenced at 5pm, to 7pm and resumed at 9pm to 10:30pm.

After some data manipulation and an attempt to perform some last data transformations I had concluded to scrap the vaccinations by age group and by manufacturer datasets. This decision was made after some additional analysis of the two datasets and concluding they won’t play a large factor in the multiple linear regression.

After a bit of a battle with MongoDB to write the query logically I was given some advice by our lecturer to focus primarily on the work at large rather than focusing my efforts on painful things.

**Fri, Dec 3rd:**

Work commenced @ 5pm and concluded @7pm.

Initialised the script file which covers to code to export data from MongoDB to PostgreSQl. Managed to create the two tables and several arrays which were to hold the data from MongoDB. Using the find() function from mongo, populated all the required arrays which later will be referenced when inserting into the SQL tables.

Took a stab at ensuring that the table exist error doesn’t appear but left it for a later time.

Team meeting held at 8:30pm, with the aim of aligning actions for Monday.

Call consisted of a Q&A around the programmatic approach for the individual analysis, aka efforts to prepare the dataset for the master file for regression analysis.

Provided the team with a run-through of my code, being a tad ahead of others. Some Q&A followed.

Mutual agreement for the following actions to be complete by for Monday’s 7pm meeting.

Text

Description automatically generated

Meeting concluded @ 10pm.

**Sun, Dec 5th:**

Work commenced @ 3:30pm and concluded @8:30pm.

Today’s tasks included cleaning up and finalising the PostgreSQL import and transform.py file.

I had figured out how to ensure that the table exists error does not show by including an if statement in the try catch to evaluate if the table is not greater than 0 to execute only then the create table function.

As the table is populated once and once only, the if will always return a true on the clause thus not executing the create table line.

Next steps and challenges presented around inserting values into the newly created tables.

Following the steps outlines [here](https://kb.objectrocket.com/postgresql/migrate-mongodb-nosql-to-postgres-with-python-860): on how to migrate data from mongo to postgre I ran into a few tables.

First challenge appeared around inserting the values as per the steps in the link. In the link above the user simply creates arrays to hold specific data from mongo and then calls on the said arrays in the insert syntax. This step was not working for me since some countries in my dataset contained a space, such as Papa New Guinea.

To resolve this, I created additional two new arrays to hold the insert values for both tables using some of the code from the above link, specifically the logic around the INSERT syntax. The aim behind the two new arrays was to compress the earlier arrays created which hold specific data from mongo into one array by enumerating them and assigning an index.

In addition to the above challenge a few error popped up around single quotes and the word null in numeric type columns which postgre wasn’t liking. Error attached below.

Text

Description automatically generated

To get around this I included some regex logic within the for loop which enumerated and created the final arrays to find and replace both the single quote with two single quotes as that is the only way postgre registers them and null with 0.

After that it was just a simple case of calling two for loops to populate both tables in postgre. I couldn’t figure out the duplicate issue which would show each time I’d run the code. Left a TODO to figure that out, only if I have some time at the end.

Helped Maria with git, as she was pushing to git but changes were not reflecting. This was due to the main project repository being stored in the wrong location on the local machine. After moving the git repository to the Git folder, git desktop was able to pick-up the changes and commit them to the repo.

Assisted Alun with the same issue.

**Mon, Dec 6th:**

Work commenced @ 6:30pm and conclude @10pm.

Prior to the team meeting finalised the final export file for the master dataset.

During the team meeting the team identified several issues with my final export, such as:

* Countries missing data for total vaccines administered.
* Countries missing data for number of fully vaccinated people.

Team suggested I identify knowledge in MongoDB which iterates backwards within the rows skipping over null values until it finds a nonnull values. As well as agreement that the data does not need to be specifically for the cut off point of the 24th of November.

Following the suggestion of the team. I started on the execution. Found some logic behind the $ne function within Mongo which essentially means not equal to in Mongo terms.

Text

Description automatically generatedConcluded to the following logic, shown here:

Issue identified upon attempt to perform the final export.

There were no issues within the for loop to pull the date for the aggregate outlined using mongo but more what it was pulling.

For some reason I was only getting the effect of the query for one line rather for all 235 within PyCharm whilst in Robo3T I was getting a perfect execution.

Decided to finish up for the day and return to this issue tomorrow after work.

**Tue, Dec 7th:**

Work commenced @ 6:30pm and conclude @10pm.

Continued the issue from Monday around the discrepancies between the execution of the aggregate function in Robo3T, Multiple attempts were made to receive the same results from Robo3T in Pymongo through PyCharm.

Unfortunately, no solution which I was applying where resolving the issue.

I have tried everything from manipulating the query differently to capture the same data, to editing the for loop which populates the mongodb collection. All resolution attempts reflected in Git. Additionally, this work bled into the team meeting with me trying to finalise the resolution and provide the final dataset for the master merge.

In the end resolved to a workaround where I split the query into two separate queries:

1. Looking at identifying total vaccinations numbers by iterating over null values.
2. Looking at identifying number of people fully vaccinated by iterating over null values.

In short, this issue was a nightmare to resolve, with only a workaround put in place. Evan decided to drop out which increased the workload.

**Wed, Dec 8th:**

Created R project environment for the regression analysis.

Checked in with the group if all changes were coming through okay. No issues form anyone’s end. Took roughly

Clarified with the lecturer on possibilities as to why different results were coming through between MongoDB and PyCharm.

Rashid clarified that there should not be any discrepancies and whilst I was trying to recreate the issue it somehow resolved itself and was executing successfully.

Still unsure as to what may have been causing the issue, but at the same time due to constantly working with the code and exhaustion most likely I may have missed something.

Decided to keep the workaround to not amend the code and potentially have it break again.

**Thur, Dec 9th:**

Team meeting held to kick-off the multiple linear regression analysis.

Maria started us off by creating a .rmd file (R Markdown), as well as explained the difference between a R Script file and a R Markdown, essentially the difference is in the UI of executing the code.

Throughout the meeting Maria was undertaking the scripting with the aim of importing the dataset from Git and running a linear regression to see if any assumptions have been violated.

The lm() function resulted in a good r squared value to begin with, unfortunately though we had a few assumptions violated such as:

* The data needs to show homoscedasticity. (2 violations)
* Our data must not contain significant outliers. (1 violation)

Fortunately, the Normal Q-Q plot was showing a normal distribution, with some slight taildness on each end.

Team agreed to distribute the violations and have them corrected by Friday. I volunteered to resolve the assumption of heteroscedasticity.

Meeting lasted a total of 2 hours.

**Fri, Dec 10th:**

Commenced on action assigned to me around addressing the violation of Homoscedasticity as the graph was showing heteroscedasticity.

Undertook the same steps as in previous project for Statistics in Data Analytics when addressing a violation of Homoscedasticity.

The process began by setting variables for each column within the data frame in work. Once variables have been set continued to plot Histograms for each column aka dependent and independent variables. The Histogram would show if any variables were experiencing skewness and uneven distribution.

A total of 6 variables were identified as intensely skewed and demonstrated good candidacy for the log of nature function, which is commonly used to resolve the violation.

Proceeded to apply the log of nature function to the identified variable which inclusive of the dependent whilst at the same time adding new columns which hold the new log of nature figure to the data frame.

Once complete, moved on to construct the linear model to evaluate if heteroscedasticity has been resolved.

Encountered a few issues with the lm function where it could not comprehend NA/NaN/INF values/figures, which from there proceeded to remove said values/figures from the dataset.

Completed this step the manual way as I was unsure how to conduct this through R.

Once the dataset has been altered, performed the linear model and has concluded that the Residual vs Fitted plot looked good, showing a good linear scatter, indicating that the violation has been dealt with.

Later in the evening conducted a team meeting to evaluate work which I’ve conducted.

Maria proposed to assist with removing the NA/NaN/INF values/figures from the dataset the correct way using code in R.

Continued to discuss the potential effect of the number of missing values for deaths. Alun to address this.

In the end we had an improvement in the regression results with the r squared value improving by 8% to 94% along with the p-value remaining under 0.005.

Work commenced at 5:30pm and concluded at 10pm.

**Sun, Dec 12th:**

Spent a little bit of the Sunday by kicking off the data visualisations in hopes to finalise the ASAP. Plotly has many options for data visualisation but can be tricky to learn and use.

Kicked the work off by running the files as I usually do and in return encountered my first issue.

The deprecation of the .count function of pymongo has finally occurred in full swing. Prior to this day it worked okay with a warning error saying the function is deprecated but use still allowed.

Resolved the .count issue by replacing it with a Boolean to determine if the collection has at least one row in it, if yes it will drop to avoid duplication prior to the collection being populated.

Once resolved moved on to creating the .py file to hold all data visualisation created. Started by getting the logic behind plotting a bar chart using plotly express.

Concluded the day after successfully plotting a bar chart.

**Mon, Dec 13th:**

Team meeting conducted at 8pm to plan for finalisation steps of the project.

Came to an agreement around the Linear Regression is complete. Have decided that by addressing the heteroscedasticity we are happy to conclude the model with an R squared value of .94 for which means that 94% of the variance in the independent variable is explained by the dependent.

Maria proposed to compliment the regression by undertaking the following functions to help evaluate the fitness of the model.

* MAE
* MAPE
* MSE

Next proceeded to planning next steps around the project.

NEXT STEPS:

* Compile functions to evaluate fitness of the model.
* Start on report
* Compile data visualizations.

ACTIONS:

1. Maria to compute MAE, MAPE, MSE using R and complete regression work.
2. Report actions:
   1. Abstract - Group writing.
   2. Introduction - Alun
   3. Related Work - Polina
   4. Methodology - Polina
   5. Result and Evaluation - Maria -report Alun -graphs
   6. Conclusions and Future Work - Group writing
   7. Bibliography - Group writing
3. Alun to compile data visualisations:
   1. Heatmaps
   2. Bar charts
   3. Waterfall charts

DUE Dates:

* Individual Mini Analysis - 16th Dec
* Project Report - 18th Dec
* Video - 21st Dec

Alun went through a quick demo on how to compile a master file aka a file which runs all the individual files in an order but from one location.

Afterwards compiled the master.py file and continued my work on the data visualisation.

**Mon, Dec 14th, 15th and 16th :**

Over the 3 days continued to work with plotly to get the best output of data visualisation to represent the data.

Went through multiple ways to best visualise the data from bar chart, subplots, scatter plots and maps. Git represents all changes applied throughout the three days.

Few things learned:

* Plotly can only display 30 text values on each x or y axis, but all values are represented.
* Plotly graphs are interactive.
* Subplots are a great way to show multiple graphs in one go.

Finalised all data visualisation on the 16th.