

ANALYSIS OF CLINICAL TRIAL DATA (MAY 2022)

(1.) DESCRIPTION OF SET UP:

This project involves the analysis of given files containing clinical trial data sets, a list of pharmaceutical industries, and a condition hierarchy of information.

The purpose is to analyze the given data and gain further insight into clinical trial data sets for the year 2021. A clinical trial is a kind of research designed to evaluate and test new interventions such as psychotherapy or medications. The usage of big data tools and techniques will be employed to perform various analysis of the three main data files given.

(1a.)Description of the files to be used include:

(1a.)Clinicaltrial_2021.csv.gz

This file was compressed into .gz file from the source. It means an archive compressed by the standard GNU zip(gzip) compression algorithm. It needs to be decompressed or unzipped to view the contents. This will be decompressed in the data cleaning and preparation process.

(Source: ClinicalTrials.gov)

(1a.) Mesh.csv

This is a plain text file that contains a list of data and allows the data to be stored in a table of structured data. It uses the comma character to separate (or delimit data). This file can be opened in any program hence it is suitable for use in Databricks.

(Source: U.S. National Library of Medicine.)

(1a.)Pharma.csv

This file shares the same features as the mesh.csv file above.

(Source: <https://violationtracker.goodjobsfirst.org/industry/pharmaceuticals>)

(1b). Description of the tool used:

The tool used is called Databricks, a cloud-based data engineering tool that is widely used by companies to process and transform large quantities of data and explore the data. Databricks is developed by the creators of pyspark, an interface for Apache spark in python. It makes available Pyspark shells for a thorough analysis of data in a computing environment where various components are spread across multiple computers.

(1c.) Insight of Analysis:

The analysis bothers on the number of studies in the dataset, all the types of studies along with their frequencies, the top five conditions, and their frequencies, the five most frequent roots, the ten most commons sponsors that are not pharmaceuticals together with the clinical trials sponsored by them, and the number of completed studies each month in the year 2021.

(1d.) Further Analysis :

A comparison between the number of completed and uncompleted studies, the top ten interventions with their frequencies, and the first five Sponsors in the year 2021 with the highest number of sponsored clinical trials were analyzed.

(2.) DATA CLEANING AND PREPARATION

This is the process of ensuring that data is correct, consistent, and usable.

(2.1)Description of Unzipping of Clinicaltrial_2021.csv.gz

The databricks environment was checked to see if the driver is a databricks driver, then the imported file clinicaltrial_2021_csv.gz was imported to Databricks through import from the GUI(Graphic User Interface). This was confirmed present in FileStore/tables with dbutils.fs.ls command as below:

(2.2) A variable called fileroot was declared as clinical_2021

BDDT RDD 2021 DATA (Python)

Declaring a variable

```
Cmd 7
fileroot="clinicaltrial_2021"
dbutils.fs.cp("/FileStore/tables/" + fileroot+"_csv.gz", "file:/tmp/",)
import os
os.environ['fileroot']=fileroot

Command took 0.49 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 25/04/2022, 20:59:43 on Coursework
```

Cmd 8

```
dbutils.fs.ls("file:/tmp/")

FileInfo(path='file:/tmp/chauffeur-daemon-params', name='chauffeur-daemon-params', size=22, modificationTime=1650907016882),
FileInfo(path='file:/tmp/systemd-private-f0eec30e3c094245b1d8b15da62ca719-systemd-logind.service-YzdtZh', name='systemd-private-f0eec30e3c094245b1d8b15da62ca719-systemd-logind.service-YzdtZh', size=4096, modificationTime=1650906978530),
FileInfo(path='file:/tmp/clinicaltrial_2021_csv.gz', name='clinicaltrial_2021_csv.gz', size=11921810, modificationTime=1650916783740),
FileInfo(path='file:/tmp/custom-spark.conf', name='custom-spark.conf', size=216, modificationTime=1650907015294),
FileInfo(path='file:/tmp/chauffeur-daemon.pid', name='chauffeur-daemon.pid', size=4, modificationTime=1650907017038),
FileInfo(path='file:/tmp/.ICE-unix/', name='.ICE-unix', size=4096, modificationTime=1650906978462),
FileInfo(path='file:/tmp/tmp.agduDU$VPP', name='tmp.agduDU$VPP', size=0, modificationTime=1650907017010),
FileInfo(path='file:/tmp/ipykernel-connection-ReplId-2ef08-12dcc-ca0aa-2.json', name='ipykernel-connection-ReplId-2ef08-12dcc-ca0aa-2.json', size=294, modificationTime=1650908142891),
FileInfo(path='file:/tmp/ipykernel-connection-ReplId-1732d-82182-8d6d6-6.json', name='ipykernel-connection-ReplId-1732d-82182-8d6d6-6.json', size=294, modificationTime=1650908132563),
FileInfo(path='file:/tmp/driver-daemon.pid', name='driver-daemon.pid', size=4, modificationTime=1650907026634),
FileInfo(path='file:/tmp/systemd-private-f0eec30e3c094245b1d8b15da62ca719-ntb.service-kvSyci', name='systemd-private-f0eec30e3c094245b1d8b15da62ca719-ntb.service-kvSyci')
```

(2.3) The file Clinicaltrial_2021_csv.gz was unzipped with the command in the screenshot below:

(2.4) The unzipped file which has now become clinical_2021_csv was moved from temp folder to FileStore/tables using the dbutils.fs.mv command.

BDDT DATAFRAME 2021 DATA - X BDDT RDD 2021 DATA - Databri... BDDT HIVE TERMWORK - Databri... Big Data_Edited - Databricks Cor... +

https://community.cloud.databricks.com/?o=233153991320092#notebook/930668662972336/command/930668662972354

BDDT RDD 2021 DATA Python

Coursework

Moving clinicaltrial_2021_csv from temp folder into FileStore/tables

Cmd 17

```
dbutils.fs.mv ("file:/tmp/clinicaltrial_2021_csv", "/FileStore/tables/", True)
```

Out[9]: True

Command took 3.46 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 25/04/2022, 21:04:10 on Coursework

Cmd 18

Checking FileStore/tables to verify clinicaltrial_2021_csv has been properly moved into it.

Cmd 19

```
dbutils.fs.head("FileStore/tables/clinicaltrial_2021_csv")
```

[Truncated to first 65536 bytes]

Out[11]: "Id|Sponsor|Status|Start|Completion|Type|Submission|Conditions|Interventions\r\nNCT02758028|The University of Hong Kong|Recruiting|Aug 2005|Nov 2021|Interventional|Apr 2016||r\nNCT02751957|Duke University|Completed|Jul 2016|Jul 2020|Interventional|Apr 2016|Autistic Disorder,Autism Spec trum Disorder|\r\nNCT02758483|Universidade Federal do Rio de Janeiro|Completed|Mar 2017|Jan 2018|Interventional|Apr 2016|Diabetes Mellitus|\r\nNCT02759848|Istanbul Medeniyet University|Completed|Jan 2012|Dec 2014|Observational|May 2016|Tuberculosis,Lung Diseases,Pulmonary Disease|\r\nNCT02758860|University of Roma La Sapienza|Active, not recruiting|Jun 2016|Sep 2020|Observational [Patient Registry]|Apr 2016|Diverticular Diseases,Diverticulitis,Diverticulosis|\r\nNCT02757209|Consortio Futuro in Ricerca|Completed|Apr 2016|Jan 2018|Interventional|Apr 2016|Asthma|Fluticasone,Xhance,Budesonide,Formoterol Fumarate,Salmeterol Xinafoate|r\nNCT02752438|Ankara University|Unknown status|May 2016|Jul 2017|Observational [Patient Registry]|Apr 2016|Hypoventilation|r\nNCT02753543|Ruijin Hospital|Unknown status|Nov 2015|Nov 2019|Interventional|Apr 2016|Lymphoma|\r\nNCT02757508|Washington University School of Medicine|Completed|Mar 2016|Jul 2017|Interventional|Apr 2016||Vitamins|r\nNCT02753530|Orphazyme|Completed|Aug 2017|Jan 2021|Interve

BDDT RDD TERMW....py clinicaltrial_2021.csv.gz bddt rdd.py Removed Big Data_Edited.py Removed BDDT HIVE TERMW....sql Removed SKYJKQY Internet access Show all

(2.5) The file clinicaltrial_2021_csv was renamed to clinicaltrial_2021.csv by using the dbutils.fs.mv command. This moves clinicaltrial_2021_csv from its present location to FileStore/tables with the new name clinicaltrial_2021.csv

(2.6) The accuracy was checked with the command

`dbutils.fs.ls("FileStore/tables")` as shown in the screenshot below

BDDT RDD 2021 DATA | Python

Coursework

Renaming clinicaltrial_2021_csv to clinicaltrial_2021.csv

```
%python  
dbutils.fs.mv("FileStore/tables/clinicaltrial_2021_csv", "FileStore/tables/clinicaltrial_2021.csv",True)
```

Out[12]: True

Command took 3.29 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 25/04/2022, 21:09:17 on Coursework

Checking FileStore/tables to verify clinicaltrial_2021.csv has been successfully moved into it.

```
dbutils.fs.ls("FileStore/tables")
```

```
725000,  
FileInfo(path='dbfs:/FileStore/tables/clinicaltrial_2019_csv.gz', name='clinicaltrial_2019_csv.gz', size=10060669, modificationTime=1650294482000),  
FileInfo(path='dbfs:/FileStore/tables/clinicaltrial_2020.csv', name='clinicaltrial_2020.csv', size=46318151, modificationTime=1650441272000),  
FileInfo(path='dbfs:/FileStore/tables/clinicaltrial_2020_copy_1.csv', name='clinicaltrial_2020_copy_1.csv', size=46318151, modificationTime=16504042565000),  
FileInfo(path='dbfs:/FileStore/tables/clinicaltrial_2021.csv', name='clinicaltrial_2021.csv', size=50359696, modificationTime=1650917360000),  
FileInfo(path='dbfs:/FileStore/tables/clinicaltrial_2021_csv.gz', name='clinicaltrial_2021_csv.gz', size=11921810, modificationTime=1650912905000),  
FileInfo(path='dbfs:/FileStore/tables/data_scrubbing_using_rdds-1.ipynb', name='data_scrubbing_using_rdds-1.ipynb', size=16994, modificationTime=16451245122937000),  
FileInfo(path='dbfs:/FileStore/tables/data scrubbing using rdds.ipynb', name='data scrubbing using rdds.ipynb', size=16994, modificationTime=1645122937000),
```

BDDT RDD TERMW...py | clinicaltrial_2021.csv.gz | bddt rdd.py | Removed | Big Data_Edited.py | Removed | BDDT HIVE TERMW...sql | Removed | Show all | X

5°C Partly cloudy | 22:50 | 25/04/2022 | ENG UK | 2

(2.7) Opening the files in Databricks:

clinicaltrial_2021.csv file is read by using: sc.textfile in rdd

Consortium|Recruiting|Nov 2017|Oct 2022|Observational|Apr 2016|Chronic Pain, Substance-Related Disorders, Opioid-Relat
ara University|Completed|Jun 2014|Apr 2015|Interventional|Apr 2016|Sleep Apnea Syndromes, Sleep Apnea|\r\nNCT02750709
l|Apr 2016|Tyrosinemias|Nitrosotherapy|\r\nNCT02753907|Yonsei University|Completed|Jun 2015||Interventional|Apr 2016||\r
pr 2016|Hemangioma|\r\nNCT02755298|University of Zurich|Completed|Oct 2016|Nov 2020|Interventional|Mar 2016|Hyperten
rials|Unknown status|Apr 2016|Jun 2019|Interventional|Mar 2016|Carcinoma|Bevacizumab, Erlotinib Hydrochloride|\r\nNCT0
all|Apr 2016|Lymphoma|Prednisone, Cyclophosphamide, Rituximab, Vincristine, Epirubicin|\r\nNCT02757131|The Cleveland Clinic
Command took 0.27 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 03/05/2022, 22:02:55 on Coursework

Cmd 4

Question 1: Checking the number of studies in the dataset ensuring distinct studies

Cmd 5

```
#Reading clinicaltrial_2021.csv
clinicaltrial_2021 = sc.textFile("FileStore/tables/clinicaltrial_2021.csv")
print("Distinct Count in the dataset clinicaltrial_2021 is : ", clinicaltrial_2021.distinct().count())
```

▶ (1) Spark Jobs

Distinct Count in the dataset clinicaltrial_2021 is : 387262

Command took 3.87 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 03/05/2022, 22:02:55 on Coursework

Cmd 6

```
clinicaltrial_2021.first()
```

▶ (1) Command Line

6°C Sunny

Windows taskbar icons: File Explorer, Edge, Mail, OneDrive, Task View, Start, Taskbar settings, Network, Power, Volume, Battery, Language, Date and Time.

Spark.read.options in DataFrames

Inbox (215) - alade... | Create Cluster - De... | BDDT 2021 DF - D... | APA7 citation gene... | "Inequality in educ... | Plagiarism Checker | New Tab | https://community.cloud.databricks.com/?o=233153991320092#notebook/930668662972426/command/930668662972430

BDDT 2021 DF Python

Detached

Reading clinicaltrial_2021.csv file

Cmd 5

```
The delimiter is "|" and the header must be set as True to give a table with headings
#The table name is clinicaltrial_2021
clinicaltrial_2021=
spark.read.options(delimiter="|",header=True).csv("/FileStore/tables/clinicaltrial_2021.csv")
display(clinicaltrial_2021)
```

▶ (1) Spark Jobs

	Id	Sponsor	Status
1	NCT02758028	The University of Hong Kong	Recruiting
2	NCT02751957	Duke University	Completed
3	NCT02758483	Universidade Federal do Rio de Janeiro	Completed
4	NCT02759848	Istanbul Medeniyet University	Completed
5	NCT02758860	University of Roma La Sapienza	Active, recruiting
6	NCT02757209	Consorzio Futuro in Ricerca	Completed
7	NCT02752420	American Health Assistance Foundation	Completed

7°C Mostly cloudy

Windows taskbar icons: File Explorer, Edge, Mail, OneDrive, Task View, Start, Taskbar settings, Network, Power, Volume, Battery, Language, Date and Time.

Spark.read.options declaring infer schema as True and setting delimiters in

Hiveql

```
%python
#Reading clinicaltrial_2021.csv file
clinicalDF=spark.read.format("csv").option("inferSchema",True).option("header",True).option("sep","|").load("dbfs:/FileStore/tables/clinicaltrial_2021.csv")
display(clinicalDF)
permanent_table_name ="clinicaltrial_2021"
clinicalDF.write.saveAsTable(permanent_table_name)
```

ID	Sponsor	Status	Start	Completion	Type
1	The University of Hong Kong	Recruiting	Aug 2005	Nov 2021	Intervention
2	Duke University	Completed	Jul 2016	Jul 2020	Intervention
3	Universidade Federal do Rio de Janeiro	Completed	Mar 2017	Jan 2018	Intervention
4	Istanbul Medeniyet University	Completed	Jan 2012	Dec 2014	Observation
5	University of Roma La Sapienza	Active, not recruiting	Jun 2016	Sep 2020	Observation

Truncated results, showing first 1000 rows.

Command took 16.39 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 03/05/2022, 06:07:28 on COURSEWORK

Reading Mesh.csv file : sc.textfile in rdd

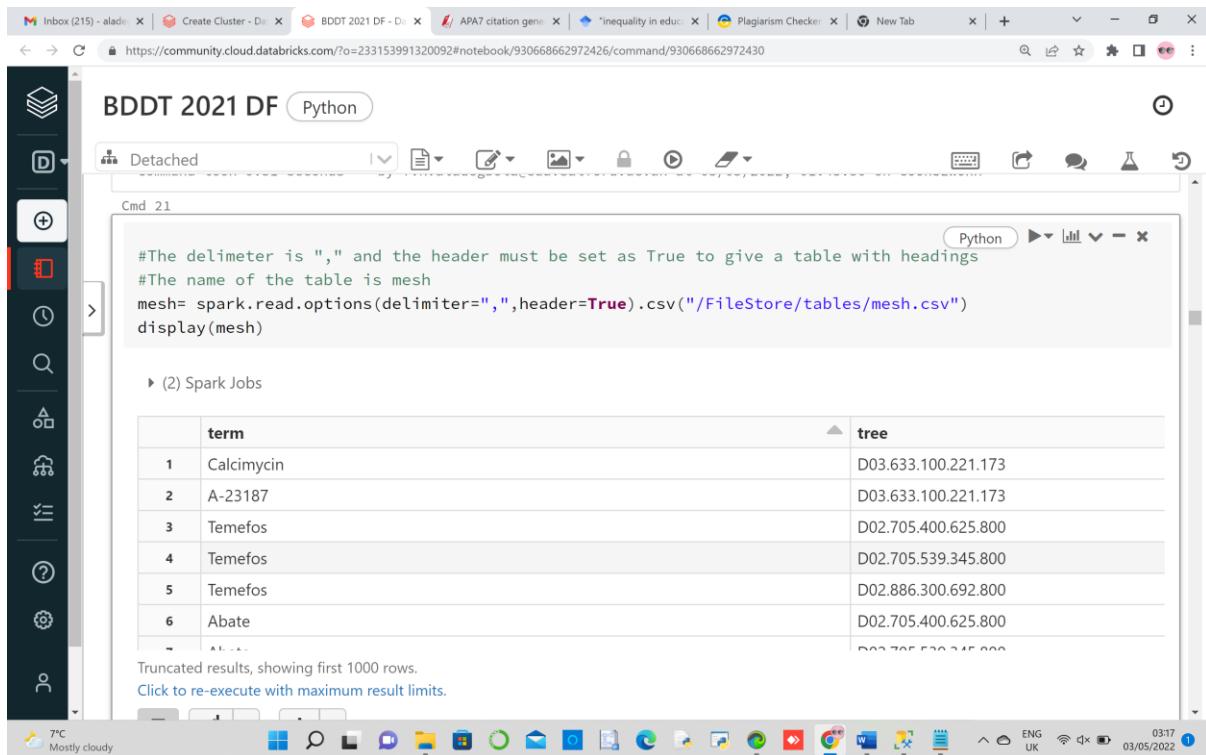
```
mesh=sc.textFile("FileStore/tables/mesh.csv")
mesh.take(5)

▶ (1) Spark Jobs
Out[24]: ['term', 'tree',
 'Calcimycin,D03.633.100.221.173',
 'A-23187,D03.633.100.221.173',
 'Temeffos,D02.705.400.625.800',
 'Temeffos,D02.705.539.345.800']

Command took 0.40 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 27/04/2022, 01:46:21 on COURSEWORK

meshW =mesh.map(lambda x: len(x))
meshW.collect()
```

Spark.read.options in DataFrames



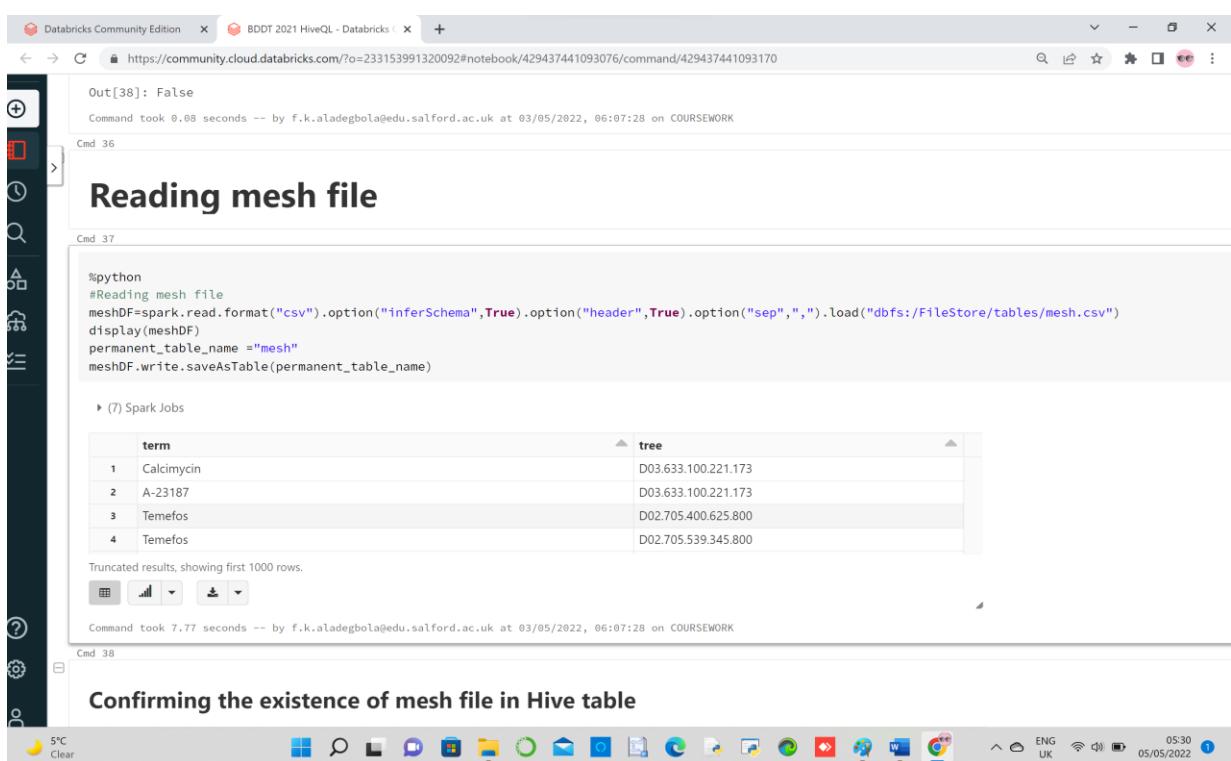
```
#The delimiter is "," and the header must be set as True to give a table with headings
#The name of the table is mesh
mesh= spark.read.options(delimiter=",",header=True).csv("FileStore/tables/mesh.csv")
display(mesh)

▶ (2) Spark Jobs
```

	term	tree
1	Calcimycin	D03.633.100.221.173
2	A-23187	D03.633.100.221.173
3	Temefos	D02.705.400.625.800
4	Temefos	D02.705.539.345.800
5	Temefos	D02.886.300.692.800
6	Abate	D02.705.400.625.800

Truncated results, showing first 1000 rows.
Click to re-execute with maximum result limits.

Spark.read.options declaring infer schema as True and setting delimiters in Hiveql



```
Out[38]: False
Command took 0.08 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 03/05/2022, 06:07:28 on COURSEWORK

Cmd 36

Reading mesh file

Cmd 37

%python
#Reading mesh file
meshDF=spark.read.format("csv").option("inferSchema",True).option("header",True).option("sep",",").load("dbfs:/FileStore/tables/mesh.csv")
display(meshDF)
permanent_table_name ="mesh"
meshDF.write.saveAsTable(permanent_table_name)

▶ (7) Spark Jobs
```

	term	tree
1	Calcimycin	D03.633.100.221.173
2	A-23187	D03.633.100.221.173
3	Temefos	D02.705.400.625.800
4	Temefos	D02.705.539.345.800

Truncated results, showing first 1000 rows.

```
Command took 7.77 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 03/05/2022, 06:07:28 on COURSEWORK
```

```
Cmd 38

Confirming the existence of mesh file
```

Pharma.csv

sc.textfile in rdd

The screenshot shows a Databricks notebook interface. The title bar says 'BDDT 2021 RDD - Python'. The left sidebar has icons for course work, clusters, jobs, and other notebook tabs. The main area shows a command cell labeled 'Cmd 33' containing the following Python code:

```
pharma=sc.textFile("FileStore/tables/pharma.csv")
pharma.take(10)
```

Below the code, the output cell 'Out[28]' displays the first 10 rows of the 'pharma' RDD. The output text is very long and includes many columns such as Company, Parent_Company, Penalty_Amount, Subtraction_From_Penalty, etc. It also contains a detailed description of a legal case involving Abbott Laboratories.

Spark.read.options in DataFrames

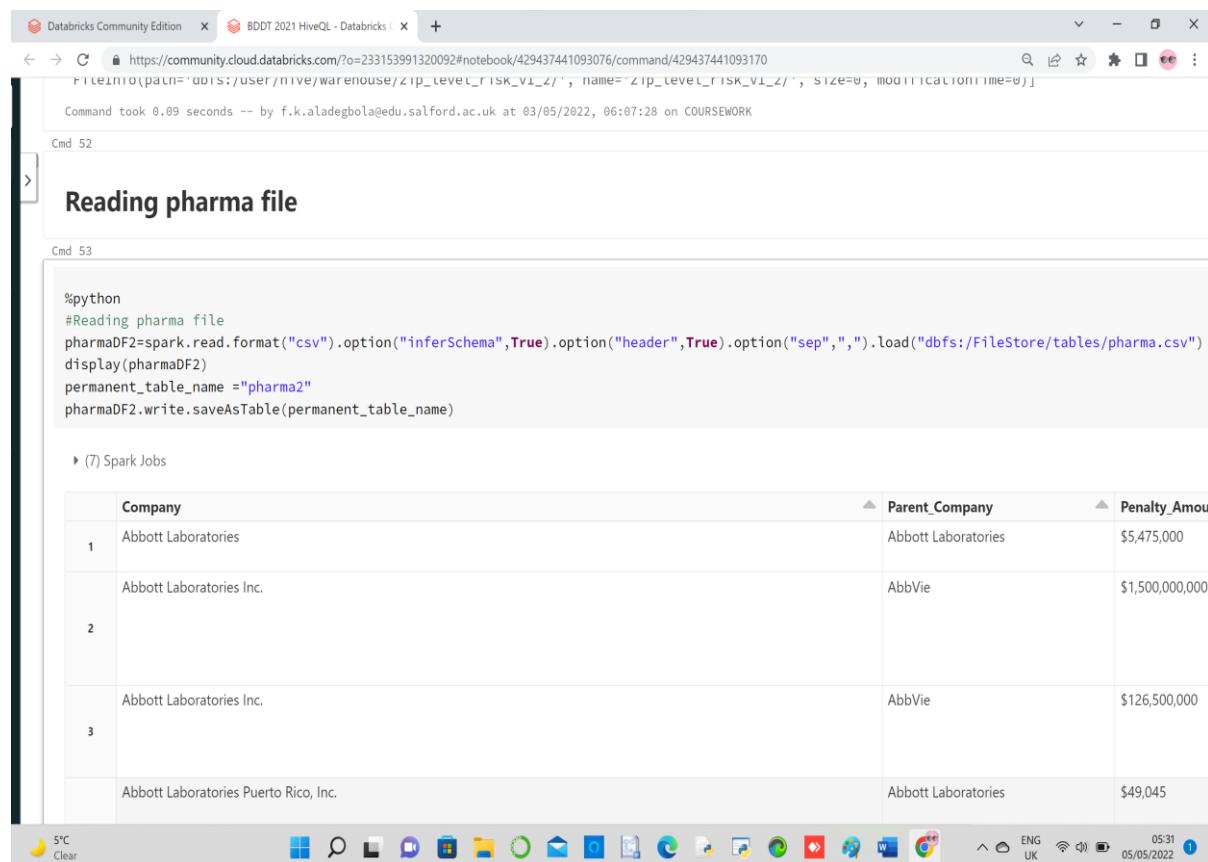
The screenshot shows a Databricks notebook interface. The title bar says 'BDDT 2021 DF - Python'. The left sidebar has icons for course work, clusters, jobs, and other notebook tabs. The main area shows a command cell labeled 'Detached' containing the following Python code:

```
#The delimiter is "," and the header must be set as True to give a table with headings
pharma= spark.read.options(delimiter=",",header=True).csv("/FileStore/tables/pharma.csv")
display(pharma)
```

Below the code, the output cell 'Out[29]' displays a DataFrame with four rows and one column named 'Company'. The data is identical to the RDD output above, showing entries for Abbott Laboratories, Abbott Laboratories Inc., Abbott Laboratories Inc., and Abbott Laboratories Puerto Rico, Inc.

Spark.read.options declaring infer schema as True and setting delimiters in

Hiveql



The screenshot shows a Databricks notebook interface. The top navigation bar has two tabs: "Databricks Community Edition" and "BDDT 2021 HiveQL - Databricks". The URL in the address bar is <https://community.cloud.databricks.com/?o=233153991320092#notebook/429437441093076/command/429437441093170>. The command history shows a command took 0.09 seconds. The current cell, Cmd 53, contains the following Python code:

```
%python  
#Reading pharma file  
pharmaDF2=spark.read.format("csv").option("inferSchema",True).option("header",True).option("sep",",").load("dbfs:/FileStore/tables/pharma.csv")  
display(pharmaDF2)  
permanent_table_name ="pharma2"  
pharmaDF2.write.saveAsTable(permanent_table_name)
```

The code reads a CSV file named "pharma.csv" from the "FileStore/tables" directory and creates a DataFrame named "pharmaDF2". It then displays the DataFrame and saves it as a permanent table named "pharma2".

The resulting DataFrame is shown in a table format:

Company	Parent_Company	Penalty_Amount
Abbott Laboratories	Abbott Laboratories	\$5,475,000
Abbott Laboratories Inc.	AbbVie	\$1,500,000,000
Abbott Laboratories Inc.	AbbVie	\$126,500,000
Abbott Laboratories Puerto Rico, Inc.	Abbott Laboratories	\$49,045

3. PROBLEM ANSWERS

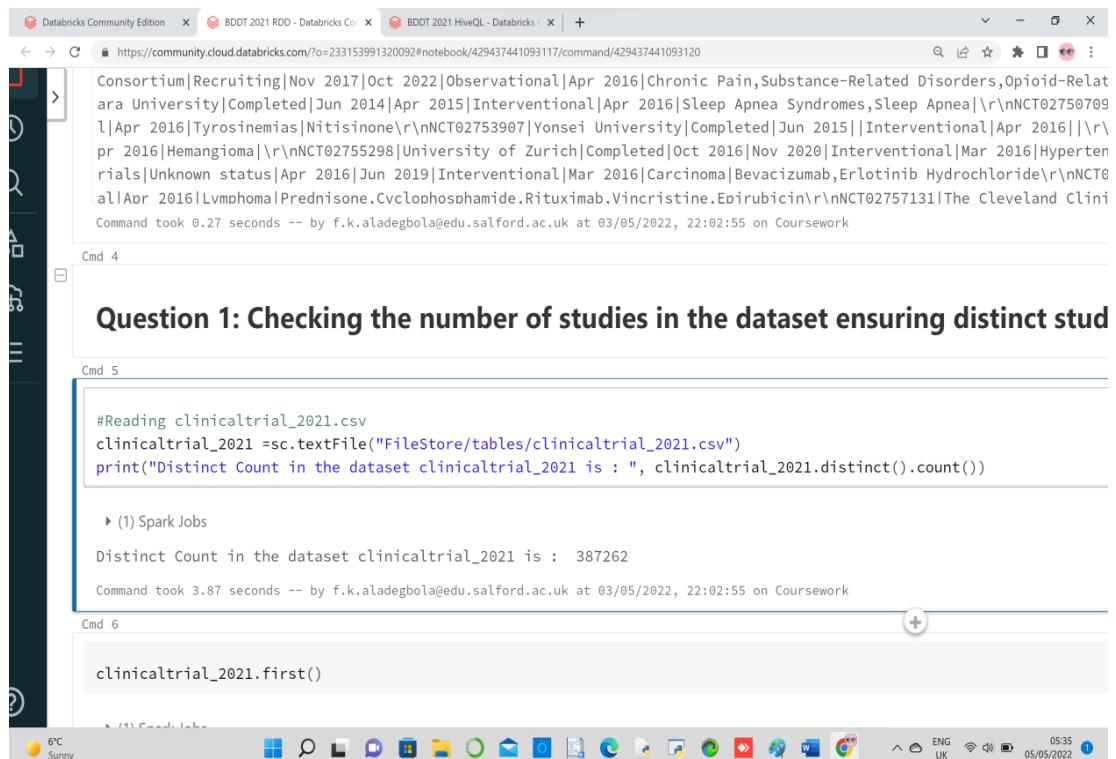
(a) Question 1 : The number of studies in the dataset?

- Assumptions made:

It was assumed that the dataset contained all the clinical trials conducted for the year 2021 and there was no error in recording.

(3a.) Implementation outline RDD

To erase the possibility of counting a clinical study twice, the distinct function was used in counting to remove duplicated values.



The screenshot shows a Databricks notebook interface with three tabs at the top: "Databricks Community Edition", "BDDT 2021 RDD - Databricks Co...", and "BDDT 2021 HiveQL - Databricks". The main area displays a command window with the following content:

```
Consortium|Recruiting|Nov 2017|Oct 2022|Observational|Apr 2016|Chronic Pain,Substance-Related Disorders,Opioid-Relat  
l|Apr 2016|Tyrosinemias|Nitisinone\r\nNCT02753907|Yonsei University|Completed|Jun 2015||Interventional|Apr 2016||\r\n  
pr 2016|Hemangioma|\r\nNCT02755298|University of Zurich|Completed|Oct 2016|Nov 2020|Interventional|Mar 2016|Hyperten  
rials|Unknown status|Apr 2016|Jun 2019|Interventional|Mar 2016|Carcinoma|Bevacizumab,Erlotinib Hydrochloride\r\nNCT0  
all|Apr 2016|Lymphoma|Prednisone,Cyclophosphamide,Rituximab,Vincristine,Epirubicin\r\nNCT02757131|The Cleveland Clin  
Command took 0.27 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 03/05/2022, 22:02:55 on Coursework
```

Below this, a section titled "Question 1: Checking the number of studies in the dataset ensuring distinct stud" is visible. The next command window (Cmd 5) contains the following Scala code:

```
#Reading clinicaltrial_2021.csv  
clinicaltrial_2021 =sc.textFile("FileStore/tables/clinicaltrial_2021.csv")  
print("Distinct Count in the dataset clinicaltrial_2021 is : ", clinicaltrial_2021.distinct().count())
```

The output shows the execution of a Spark job and the resulting count:

```
▶ (1) Spark Jobs  
Distinct Count in the dataset clinicaltrial_2021 is : 387262  
Command took 3.87 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 03/05/2022, 22:02:55 on Coursework
```

The final command window (Cmd 6) shows the execution of the `first()` method on the RDD:

```
clinicaltrial_2021.first()
```

Removal of the data heading called header is essential because it was counted with the clinical trial. This was achieved with the filter function by passing it through the first rdd . The resultant rdd called headerRemvRdd1 was thereafter used in the final count.

```

▶ (1) Spark Jobs
Out[152]: 'Id|Sponsor|Status|Start|Completion|Type|Submission|Conditions|Interventions'
Command took 0.85 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 03/05/2022, 22:02:55 on Coursework

Cmd 7

Remove header

Cmd 8

headerRemvRDD= clinicaltrial_2021.first()
headerRemvRDD1=clinicaltrial_2021.filter(lambda x:x!= headerRemvRDD)
headerRemvRDD1.first()

▶ (2) Spark Jobs
Out[153]: 'NCT02758028|The University of Hong Kong|Recruiting|Aug 2005|Nov 2021|Interventional|A
Command took 1.97 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 03/05/2022, 22:02:55 on Coursework

Cmd 9

headerRemvRDD1.take(4)

  6°C Sunny 05:42 05/05/2022

```

(3a.) HiveQL implementation outline

Opening Clinicaltrial_2019 .csv file transformed it into a table of structured data. To get the count in hiveql, select distinct and count statement is used to return select unduplicated rows with their count respectively.

```

Command took 0.17 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 03/05/2022, 06:07:28 on COURSEWORK

Cmd 12

Question 1: Check the number of studies in the dataset ensuring distinct studies are taken into consideration

Cmd 13

select, count and distinct statements are used to get the total number of unduplicated datasets

Cmd 14

--Getting total number of unduplicated datasets
SELECT COUNT(*) FROM (SELECT DISTINCT * FROM clinicaltrial_2021)

▶ (3) Spark Jobs
count(1)
1 387261

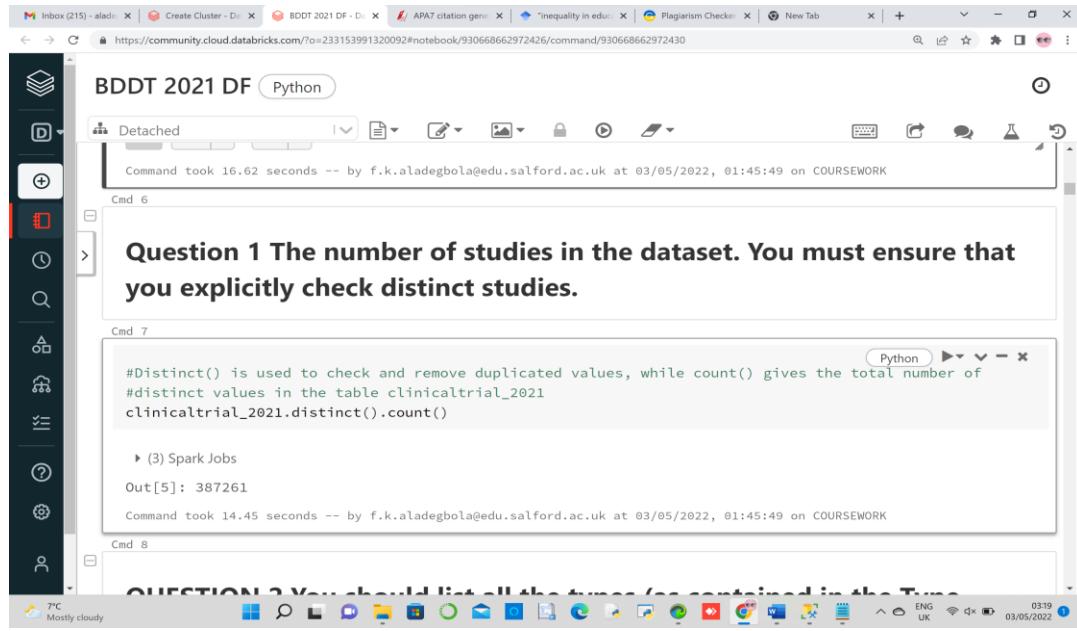
Showing all 1 rows.

  6°C Sunny 05:45 05/05/2022

```

(3a.) PySpark implementation outline DF

Dataframe also makes use of distinct and count statements to return non-duplicated rows. Unlike hiveql, it is applied directly to the named dataframe used to open the file.



The screenshot shows a Databricks notebook interface. The title bar says 'BDDT 2021 DF Python'. The notebook contains the following text:

```
#Distinct() is used to check and remove duplicated values, while count() gives the total number of  
#distinct values in the table clinicaltrial_2021  
clinicaltrial_2021.distinct().count()
```

Below the code, there is a command history:

- Cmd 6: Question 1 The number of studies in the dataset. You must ensure that you explicitly check distinct studies.
- Cmd 7: Out[5]: 387261
- Cmd 8: QUESTION 2 You should list all the steps for calculating the Total

The status bar at the bottom shows the date and time: 03/05/2022, 01:45:49.

(3a.) Result on the submission (final) data –

The number of studies in clinicaltrial_2019.csv resulted in 387,261 using the three implementations in databricks.

(3a.) Discussion of result:

The number of studies in 2019 and 2020 was 326348 and 356466 respectively.

Comparing this number to the number in 2021, a total of 387,261 shows an increase in the number of clinical trials being carried out yearly.

(3b.) PROBLEM ANSWERS

- **Question 2:** You should list all the types (as contained in the Type column) of studies in the dataset along with the frequencies of each type. These should be ordered from most frequent to least frequent.

(3b.) **Assumptions made:**

It was assumed that four types of studies were carried out in the year 2021 and all the studies carried out were properly documented.

(3b.) **PySpark implementation outline RDD**

The rdd used in question one called headerRemvRDd1, whose header was removed was used to map a new rdd called splitRDD. Re was imported as a library in order for the function split to work with the anonymous function lambda. Splitting the file made each row a list. The first column in each row represents x[0], and the second x[1] e.t.c The Type of study is the 6th column represented by x[5]. This was mapped into a new rdd called typelistRdd with the reduceByKey which assigns the number one to each type of clinical trial and then aggregates it and returns the total count for each type of clinical trial.

```

BDDT 2021 HiveQL - Databricks | BDDT 2021 DF - Databricks Com | BDDT 2021 RDD - Databricks Co | + https://community.cloud.databricks.com/?o=233153991320092#notebook/429437441093117/command/429437441093130
BDDT 2021 RDD - Databricks Co
BDDT 2021 RDD - Databricks Co
BDDT 2021 RDD - Databricks Co

BDDT 2021 RDD (Python)

 COURSEWORK
import re
splitRDD = headerRemvRDD1.map(lambda line: re.split('[]]',line))
splitRDD.take(1)

(1) Spark Jobs
Out[35]: [['NCT02758028',
'The University of Hong Kong',
'Recruiting',
'Aug 2005',
'Nov 2021',
'Interventional',
'Apr 2016',
'',
'']]

Command took 0.77 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 27/04/2022, 02:17:36 on COURSEWORK

Cmd 12
from pyspark.sql.functions import *
typeListRDD= splitRDD.map(lambda x: (x[5], 1)).reduceByKey(lambda a,b: a + b)
typeListRDD.take(5)

(2) Spark Jobs
Out[14]: [('Observational', 77540),
('Observational [Patient Registry]', 8180),
('Interventional', 301472),
('Expanded Access', 69)]

```

(3b.) HiveQL implementation outline

Clinicaltrial_2021 is a structured table with defined rows and columns. In order to get the total number of the Types of clinical trials, the ‘select’ statement with ‘count’ was used to specifically return the Type column and enumerate each of the Types. The “groupby” statement was then used to aggregate the total number of counts for each type of clinical trial.

```

SELECT Type, count(*)
FROM clinicaltrial_2021
GROUP BY Type

```

Type	count(1)
1 Observational [Patient Registry]	8180
2 Expanded Access	69
3 Interventional	301472
4 Observational	77540

Showing all 4 rows.

(3b.) PySpark implementation outline DF

The Types of the clinical trial was obtained by applying 'groupBy()' and 'count()' statements to type column. This groups rows that have the same values into summary rows and the result is set by the count column.

QUESTION 2. You should list all the types (as contained in the Type column) of studies in the dataset along with the frequencies of each type. These should be ordered from most frequent to least frequent.

```

#descendingDfx is the new dataframe
#groupBy function splits the objects in Type column and arranges the result while count() gives the number #of entries in clinicaltrial_2021
descendingDfx=clinicaltrial_2021.groupBy('Type').count()
descendingDfx.show()

```

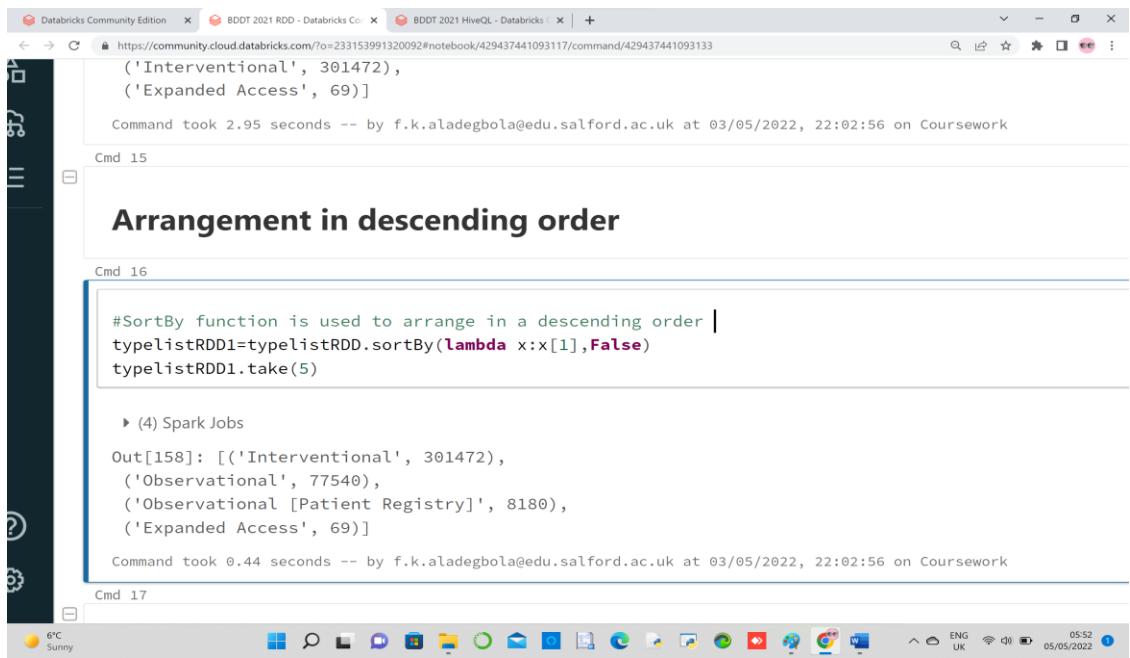
Type	count
Observational [Pa...]	8180
Expanded Access	69
Interventional	301472
Observational	77540

Command took 7.16 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 03/05/2022, 01:45:49 on COURSEWORK

(3b.) Result on the submission (final) data

RDD RESULT

The result obtained was arranged in descending order by making use of the anonymous function lambda and sortBy including the Boolean value False to show the descending order in another rdd called typelistRDD1.



A screenshot of a Databricks notebook interface. The URL in the address bar is https://community.cloud.databricks.com/. The notebook has three tabs: 'BDOT 2021 RDD - Databricks' (active), 'BDOT 2021 HiveQL - Databricks', and 'BDOT 2021 HiveQL - Databricks'. The active tab shows a command window with the following content:

```
('Interventional', 301472),
('Expanded Access', 69)

Command took 2.95 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 03/05/2022, 22:02:56 on Coursework
```

Below this is a section titled 'Arrangement in descending order' with the following code:

```
#SortBy function is used to arrange in a descending order |
typelistRDD1=typelistRDD.sortBy(lambda x:x[1],False)
typelistRDD1.take(5)
```

Output:

```
▶ (4) Spark Jobs
Out[158]: [('Interventional', 301472),
             ('Observational', 77540),
             ('Observational [Patient Registry]', 8180),
             ('Expanded Access', 69)]

Command took 0.44 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 03/05/2022, 22:02:56 on Coursework
```

The status bar at the bottom shows the date as 05/05/2022 and the time as 05:52.

(3b.) HIVEQL RESULT

This implementation used the statements 'orderby' and 'Desc' meaning descending to list the types of clinical trials and arrange their frequencies in descending order.

The screenshot shows a Databricks notebook interface. In the command editor (Cmd 19), a SQL query is run:

```
--descending order arrangement
SELECT Type, count(*)
FROM clinicaltrial_2021
GROUP BY Type
ORDER BY count(1) DESC
```

The results are displayed in a table titled '(2) Spark Jobs':

Type	count(1)
1 Interventional	301472
2 Observational	77540
3 Observational [Patient Registry]	8180
4 Expanded Access	69

Showing all 4 rows.

Command took 1.47 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 03/05/2022, 06:07:28 on COURSEWORK

(3b.) DATAFRAME RESULT

ascending=False in sort was adopted for descending order arrangement.

The screenshot shows a Databricks notebook interface. A Python code snippet is run in the command editor (Cmd 11):

```
#To arrange descendingDfx in a descending order| the sort() function is used, highlighting the count
#column as the coulumn to be used. Ascending is declared as False to get a descending arrangement
descendingDfx=clinicaltrial_2021.groupBy('Type').count().sort('count',ascending=False)
descendingDfx.show()
```

The results are displayed in a table titled '(2) Spark Jobs':

Type	count
Interventional	301472
Observational	77540
Observational [Pa...]	8180
Expanded Access	69

(3b.) Discussion of result:

Using the three implementations on data bricks, interventional emerged as the most widely used type of clinical trial and expanded access the least. Also, this result shows that the types of clinical trials used have remained the same for the years 2019,2020, and 2021 with no addition of other Types.

(3c.) PROBLEM ANSWERS

Question 3: The top 5 conditions (from Conditions) with their frequencies

- Assumptions made:**

The assumption was that the data was properly documented and free of errors during collation.

(3c.) PySpark implementation outline RDD

All the libraries connoted by (*) were imported from pyspark .sql functions.

The 8th column i.e x[7] was passed through the anonymous function lambda to map the conditions column into a list. The flatMap function was adopted thereafter to split /flatten the row because there were more than one disease

on some of the rows. The reduceByKey was used afterward to list and aggregate the frequency of each condition.

```

BDDT 2021 HiveQL - Databricks | BDDT 2021 DF - Databricks Com | BDDT 2021 RDD - Databricks Co | + 
https://community.cloud.databricks.com/?o=233153991320092#notebook/429437441093117/command/429437441093138

BDDT 2021 RDD Python
QUESTION 3: The top 5 conditions (from Conditions) with their frequencies

Cmd 16
from pyspark.sql.functions import *
conditionRDD= splitRDD.map(lambda x: x[7])
conditionRDD.take(3)

▶ (1) Spark Jobs
Out[36]: ['!', 'Autistic Disorder,Autism Spectrum Disorder', 'Diabetes Mellitus']
Command took 0.96 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 27/04/2022, 02:24:15 on COURSEWORK

Cmd 17
Conditions has more than one disease per row as seen above ,so it needs too be splitted.

Cmd 18
conditionRDD1= conditionRDD.flatMap(lambda x: x.split(","))
conditionRDD1.take(2)

▶ (1) Spark Jobs
Out[44]: ['!', 'Autistic Disorder']
Command took 0.79 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 27/04/2022, 02:27:44 on COURSEWORK

Cmd 19
conditionRDD2=conditionRDD1.map(lambda x:(x,1)).reduceByKey(lambda a,b: a+b)
conditionRDD2.take(6)

▶ (1) Spark Jobs
Out[18]: [('!', 65131),
('Autistic Disorder', 867),
('Autism Spectrum Disorder', 880),
('Tuberculosis', 1118),
('Diverticular Diseases', 65),
('Diverticulosis', 18)]

```

(3c.) HiveQL implementation outline

The function (explode(split,x)') was adopted with the delimiter set as a comma in order to split the conditions column. The reason for the split stems from the fact that some of the rows in the conditions column have more than one condition per row.

```

BDDT 2021 HiveQL SQL
Question 3: The top 5 conditions (from Conditions) with their frequencies

Cmd 15
SELECT Conditions
FROM clinicaltrial_2021
(1) Spark Jobs
Conditions
1 null
2 Autistic Disorder,Autism Spectrum Disorder
3 Diabetes Mellitus
4 Tuberculosis,Lung Diseases,Pulmonary Disease
5 Diverticular Diseases,Diverticulum,Diverticulosis
6 Asthma
7 Herniventilation
Truncated results, showing first 1000 rows.
Click to re-execute with maximum result limits.

Cmd 16
The conditions column needs to be splitted because some rows have more than one disease
select Id,Sponsor,explode (split(Conditions,"")) ConditionsSplitted
from clinicaltrial_2021
(1) Spark Jobs
Id Sponsor
1 NCT04463966 Benha University
2 NCT04463966 Benha University
ConditionsSplitted
Postpartum Hemorrhage
Hemorrhage

```

(3c.) PySpark implementation outline DF

Just like Hiveql the same function (`explode(split,x)',')` was used to split the rows with more than one condition per row. A new name with an alias 'conditions' was created.

(3c.) Result on the submission (final) data

The arrangement in descending order for the three implementations follows the same process as the Type column.

- (3c.) RDD: Makes use of `sortBy` and the Boolean expression `False` for a descending order arrangement.

```

from pyspark.sql.functions import *
conditionRDD3=conditionRDD2.sortBy(lambda x:x[1],False)
conditionRDD3.take(6)

▶ (3) Spark Jobs
Out[19]: [(' ', 65131),
('Carcinoma', 13389),
('Diabetes Mellitus', 11080),
('Neoplasms', 9371),
('Breast Neoplasms', 8640),
('Syndrome', 8632)]

Command took 0.37 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 27/04/2022, 01:44:30 on COURSEWORK

Cmd 22

The null value must be removed.

conditionRDD4=conditionRDD3.first()
conditionRDD4remNull=conditionRDD3.filter(lambda x:x!= conditionRDD4)
conditionRDD4remNull.take(5)

▶ (2) Spark Jobs
Out[21]: [('Carcinoma', 13389),
('Diabetes Mellitus', 11080),
('Neoplasms', 9371),
('Breast Neoplasms', 8640),
('Syndrome', 8632)]

```

(3c.) HIVE: Makes use of order by with desc .

```

select explode(split(Conditions,"")) ConditionsSplitted
from clinicaltrial_2021

▶ (1) Spark Jobs
ConditionsSplitted
1 Postpartum Hemorrhage

Truncated results, showing first 1000 rows.
Click to re-execute with maximum result limits.

Command took 0.79 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 27/04/2022, 01:15:28 on COURSEWORK

Cmd 19

create view SplitTheConditions1 as
select explode (split(Conditions,"")) ConditionsSplitted1
from clinicaltrial_2021

OK

Command took 0.49 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 27/04/2022, 01:16:00 on COURSEWORK

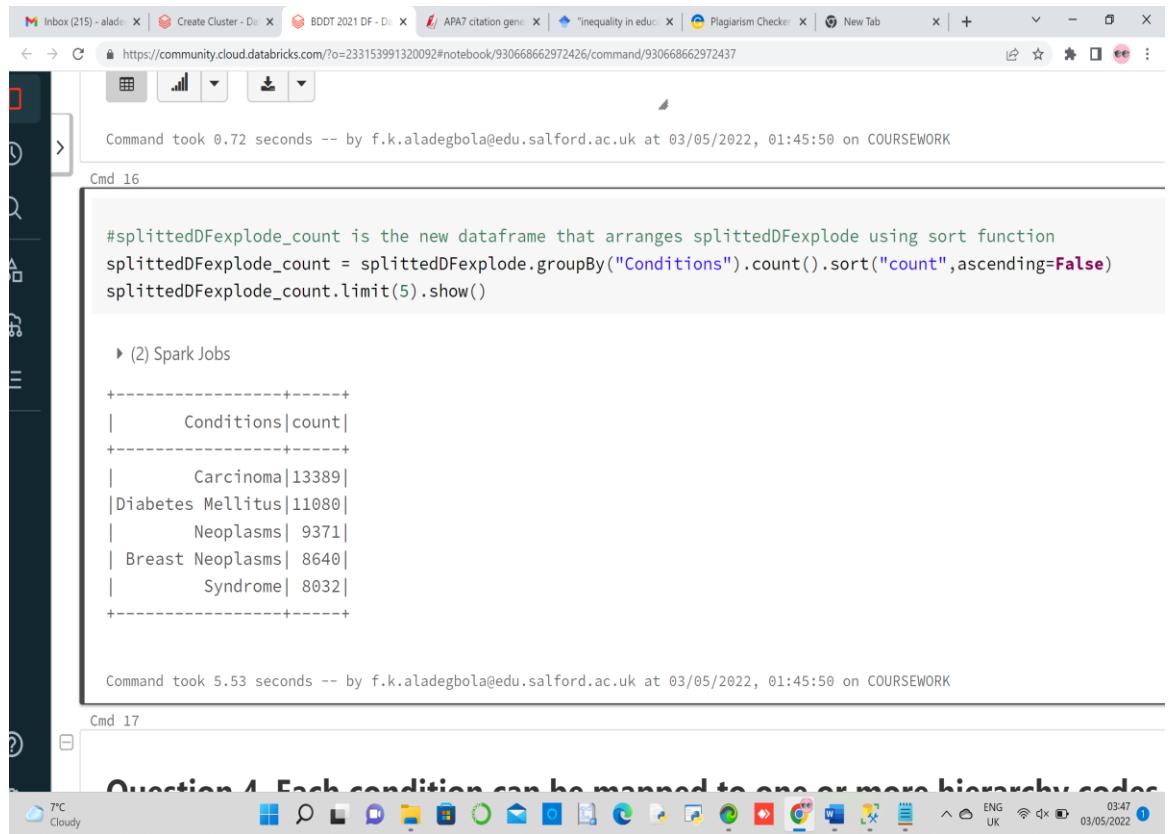
Cmd 20

select ConditionsSplitted1, count(*)
from SplitTheConditions1
group by ConditionsSplitted1
ORDER BY count(*) DESC
limit (5)

▶ (2) Spark Jobs
ConditionsSplitted1 ▲ count(1) ▲
1 Carcinoma 13389
2 Diabetes Mellitus 11080
3 Neoplasms 9371
4 Breast Neoplasms 8640

```

(3c.) **DATAFRAME**: Makes use of sort with Boolean expression False for a descending order arrangement.



The screenshot shows a web browser window with multiple tabs open. The active tab is a Databricks notebook at <https://community.cloud.databricks.com/?o=233153991320092#notebook/930668662972426/command/930668662972437>. The command in the notebook is:

```
#splittedDFexplode_count is the new dataframe that arranges splittedDFexplode using sort function
splittedDFexplode_count = splittedDFexplode.groupBy("Conditions").count().sort("count",ascending=False)
splittedDFexplode_count.limit(5).show()
```

The output of the command is:

```
+-----+----+
|      Conditions|count|
+-----+----+
|      Carcinoma|13389|
|Diabetes Mellitus|11080|
|      Neoplasms| 9371|
| Breast Neoplasms| 8640|
|      Syndrome| 8032|
+-----+----+
```

Below the command, there is a note: "Question 4. Each condition can be mapped to one or more hierarchy codes".

(3c.) **Discussion of result:**

The three implementations gave the same result with carcinoma emerging as the condition with the highest frequency.

(3d.) PROBLEM ANSWERS

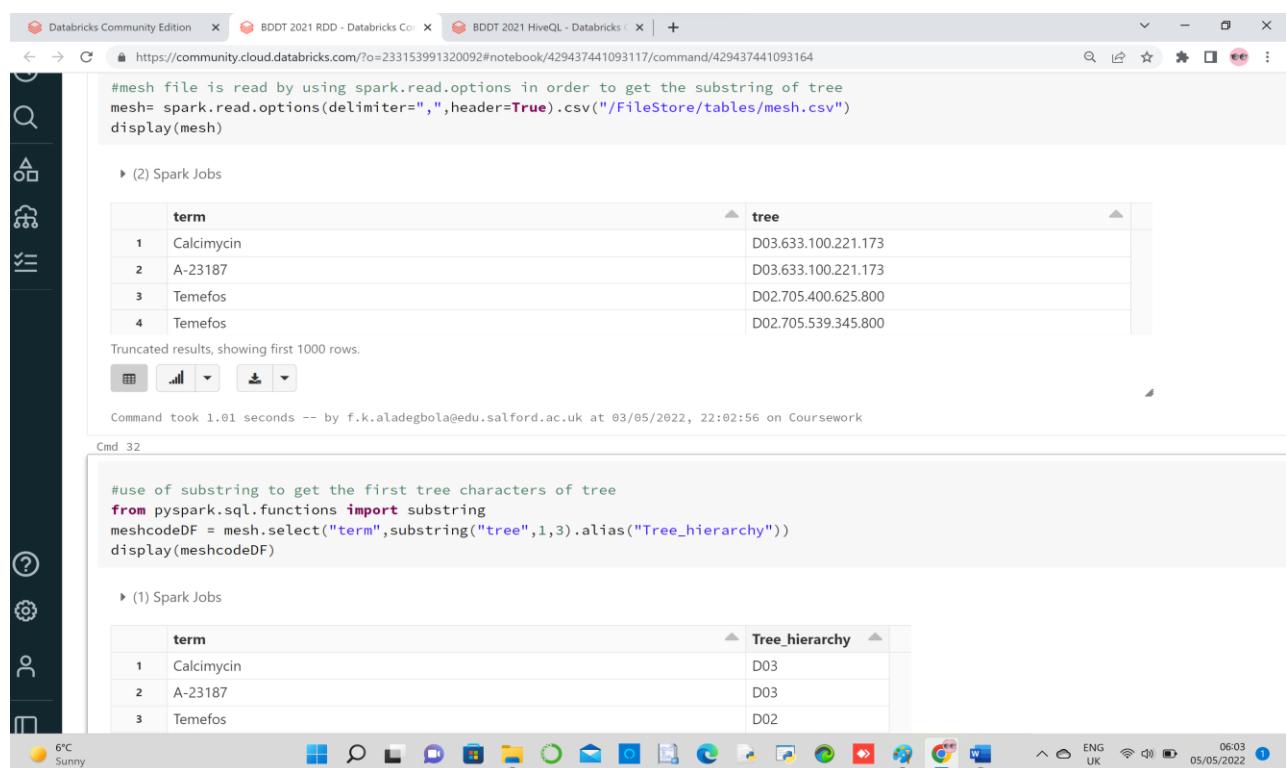
- **Question:** Each condition can be mapped to one or more hierarchy codes.

The client wishes to know the 5 most frequent roots (i.e. the sequence of letters and numbers before the first full stop) after this is one.

3d.) **Assumptions made:** It is assumed that the hierarchy codes are performed to categorize the medical terms reported approximately so that they can be analyzed or reviewed.

(3d.) PySpark implementation outline RDD:

The mesh file is read through spark.read.options. This gives the mesh file in a tabular form. Then, the first three substrings of the column tree are selected using the statement ‘substring’ with an alias of Tree_Hierarchy. The selected term and substring tree are mapped to meshcodeDF



The screenshot shows a Databricks notebook interface with two code cells and their corresponding output tables.

Code Cell 1:

```
#mesh file is read by using spark.read.options in order to get the substring of tree
mesh= spark.read.options(delimiter=",",header=True).csv("/FileStore/tables/mesh.csv")
display(mesh)
```

Output Table 1:

term	tree
1 Calcimycin	D03.633.100.221.173
2 A-23187	D03.633.100.221.173
3 Temefos	D02.705.400.625.800
4 Temefos	D02.705.539.345.800

Truncated results, showing first 1000 rows.

Code Cell 2:

```
#use of substring to get the first tree characters of tree
from pyspark.sql.functions import substring
meshcodeDF = mesh.select("term",substring("tree",1,3).alias("Tree_hierarchy"))
display(meshcodeDF)
```

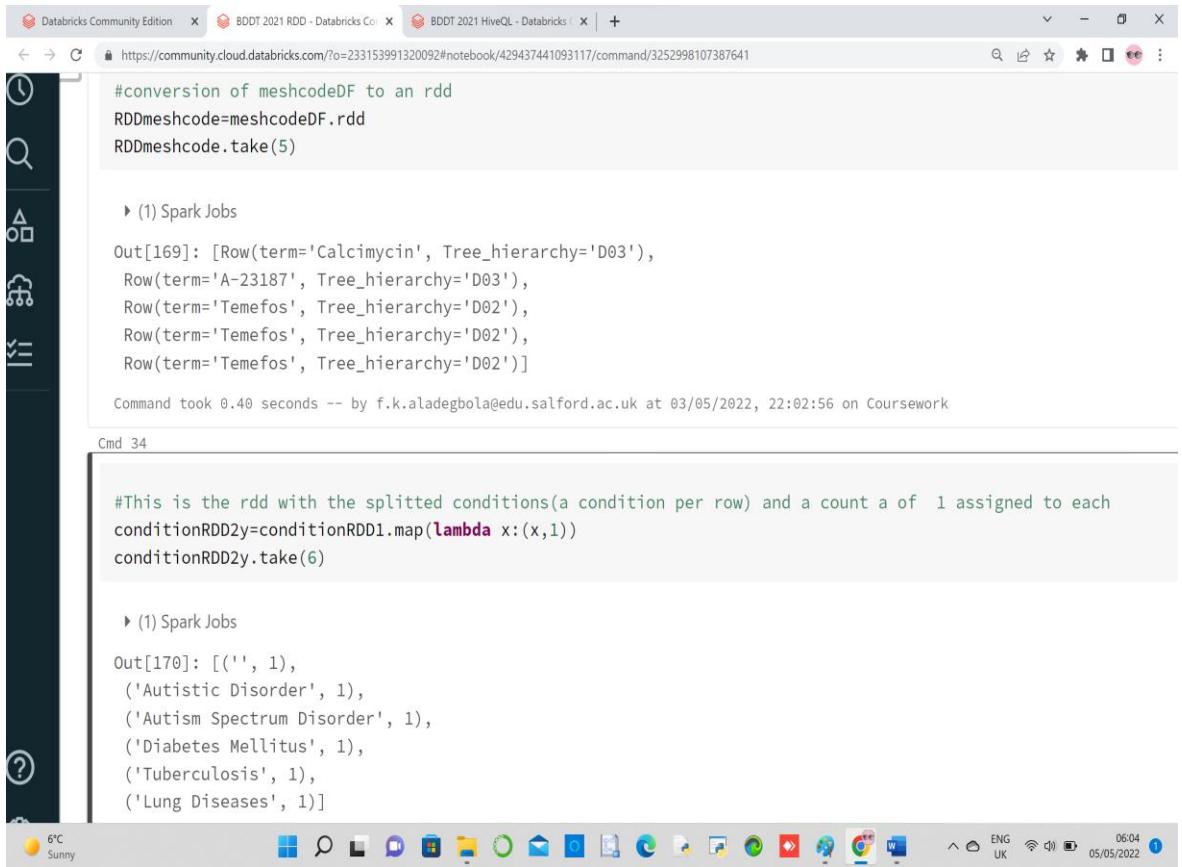
Output Table 2:

term	Tree_hierarchy
1 Calcimycin	D03
2 A-23187	D03
3 Temefos	D02

meshcodeDF is changed to an rdd using the dotrdd i.e (.rdd)

The rdd with the splitted conditions conditionRDD1, earlier in question 3

is mapped to a new rdd called conditionRDD2y by assigning a count of 1 to each condition.



```
#conversion of meshcodeDF to an rdd
RDDmeshcode=meshcodeDF.rdd
RDDmeshcode.take(5)

▶ (1) Spark Jobs

Out[169]: [Row(term='Calcimycin', Tree_hierarchy='D03'),
Row(term='A-23187', Tree_hierarchy='D03'),
Row(term='Temefos', Tree_hierarchy='D02'),
Row(term='Temefos', Tree_hierarchy='D02'),
Row(term='Temefos', Tree_hierarchy='D02')]

Command took 0.40 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 03/05/2022, 22:02:56 on Coursework

Cmd 34

#This is the rdd with the splitted conditions(a condition per row) and a count a of 1 assigned to each
conditionRDD2y=conditionRDD1.map(lambda x:(x,1))
conditionRDD2y.take(6)

▶ (1) Spark Jobs

Out[170]: [(' ', 1),
('Autistic Disorder', 1),
('Autism Spectrum Disorder', 1),
('Diabetes Mellitus', 1),
('Tuberculosis', 1),
('Lung Diseases', 1)]
```

(3d.) Hive implementation outline

Hive makes use of select, inner join, and group by to map the first three substrings of Tree to the conditions in the clinical trial, with a count of it.

The screenshot shows the Databricks SQL interface. The top navigation bar has three tabs: "BDDT 2021 HiveQL - Databricks", "BDDT 2021 DF - Databricks", and "BDDT 2021 RDD - Databricks". The current tab is "BDDT 2021 HiveQL". The URL in the address bar is <https://community.cloud.databricks.com/?o=233153991320092#notebook/429437441093076/command/429437441093103>. The main area displays a SQL query in a code editor:

```

select SUBSTRINGTree, count(*) countvalue
from(select c.ConditionsSplitted, SUBSTRING(m.tree, 1, 3) SUBSTRINGTree
      from (select Id,explode (split(Conditions,",")) ConditionsSplitted
            from clinicaltrial_2021) c
inner join mesh m on c.ConditionsSplitted=m.term)
group by SUBSTRINGTree
    
```

The results of the query are shown in a table:

	SUBSTRINGTree	countvalue
1	E02	137
2	G12	151
3	C22	205
4	I01	758
5	C24	177
6	F01	12686
7	In3	9

Showing all 63 rows.

Below the table, the command took 4.58 seconds and was run by f.k.aladegbola@edu.salford.ac.uk at 27/04/2022, 02:46:11 on COURSEWORK.

(3d.) PySpark implementation outline DF

All libraries are imported from `pyspark.sql.functions` with *, then `substring` is used to select the term column and the first three substrings of the tree column. The resulting rdd `tree_meshDF` is joined to the exploded rdd in question 3, `splittdFExplode`. This creates a table with the columns conditions, term and the first three substrings of tree.

The screenshot shows a Databricks notebook interface. The title bar says "BDDT 2021 DF" and "Python". The main area contains two command cells:

Cmd 23:

```
#The function substring is used to get the first 3 parts of the string in column Tree for datafram mesh. #The new column is named treecode
from pyspark.sql.functions import *
tree_meshDF= mesh.select("term", substring("tree", 1,3).alias("treecode"))
tree_meshDF.take(4)
```

Output:

```
(1) Spark Jobs
Out[14]: [Row(term='Calcimycin', treecode='D03'),
Row(term='A-23187', treecode='D03'),
Row(term='Temefos', treecode='D02'),
Row(term='Temefos', treecode='D02')]
```

Command took 0.85 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 03/05/2022, 01:45:50 on COURSEWORK

Cmd 24:

```
#The exploded conditions column will be joined with the new datafram that has the first 3 substrings of tree where the conditions are equal
Joinedclinical_mesh= splittedDFexplode.join(tree_meshDF, tree_meshDF.term == splittedDFexplode.Conditions, "inner")
display(Joinedclinical_mesh)
```

Output:

```
(1) Spark Jobs
```

Conditions	term	treecode
1 Autistic Disorder	Autistic Disorder	F03
2 Autism Spectrum Disorder	Autism Spectrum Disorder	F03
3 Diabetes Mellitus	Diabetes Mellitus	C19
4 Diabetic Mellitus	Diabetic Mellitus	C19

Truncated results, showing first 1000 rows.
Click to re-execute with maximum result limits.

(3d.) Result on the submission (final) data

(3d.) RDD: The result for question 4 under rdd is mapped to HierarchycodeRDD with the use of map, lambda, reduceBKey, sortBy, and the Boolean expression False to obtain a count, get a frequency and arrange in descending order.

```
#joining the the two needed columns

condition_meshJoin= conditionRDD2y.join(RDDmeshcode)
condition_meshJoin.collect()

▶ (1) Spark Jobs

Out[171]: [('Autistic Disorder', (1, 'F03')),  
 ('Autistic Disorder', (1, 'F03'))]
```

```
# selection of roots with its hierarchy in descending order
HierarchyCodeRDD=condition_meshJoin.map(lambda x: (x[1][1], 1))\
.reduceByKey(lambda c,d: c+d)\
.sortBy(lambda x: x[1],False)
HierarchyCodeRDD.take(5)

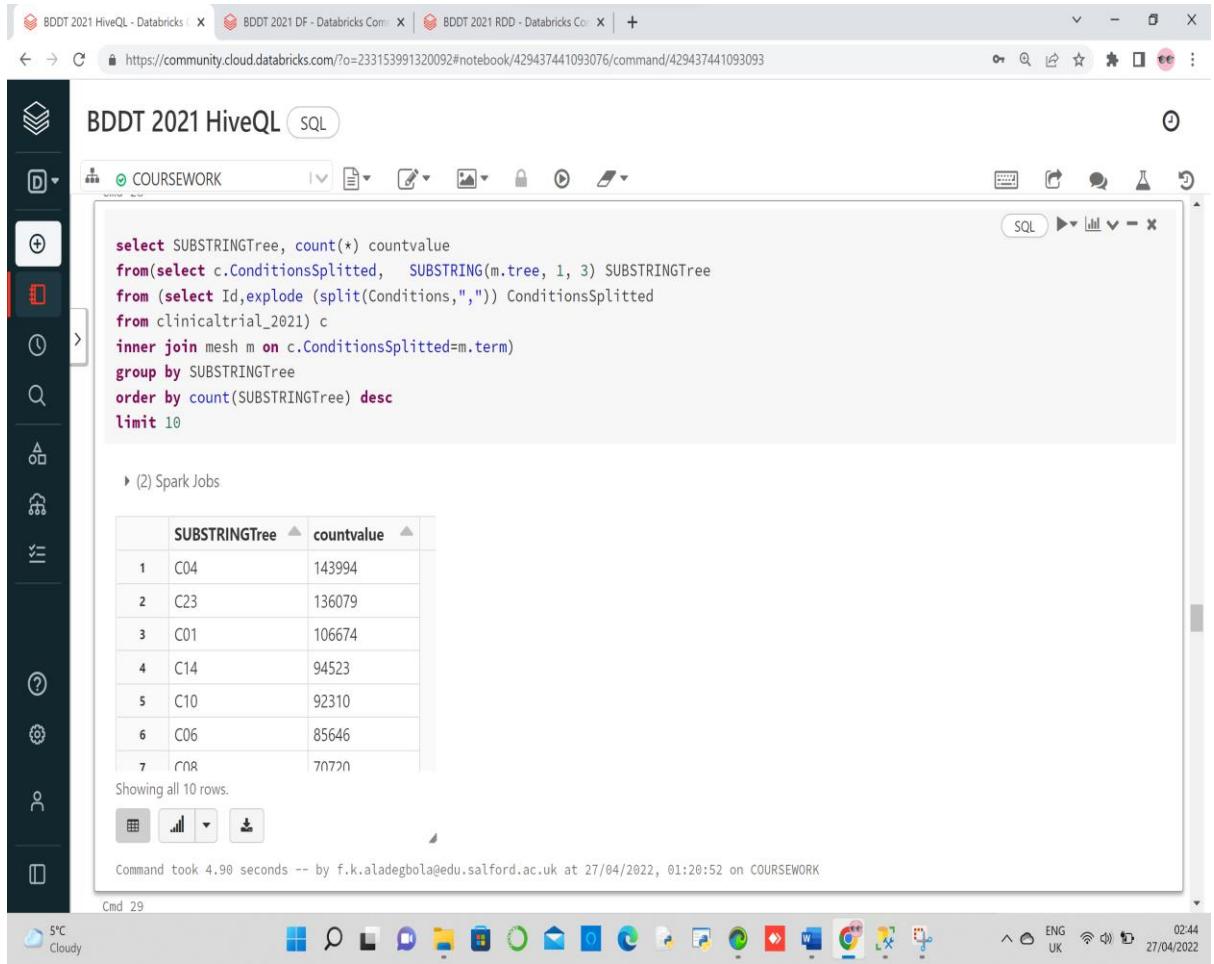
▶ (3) Spark Jobs

Out[172]: [('C04', 143994),  
 ('C23', 136079),  
 ('C01', 106674),  
 ('C14', 94523),  
 ('C10', 92310)]
```

(3d.) HIVE

The select statement is used to pick the relevant columns with an inner join.

The group by, order by and limit are finally used to get a frequency of the first three substring of the column tree arranged in a descending order Frquency.



The screenshot shows a Databricks notebook interface. The left sidebar has a tree view of notebooks. The main area is titled "BDDT 2021 HiveQL". A SQL tab is selected. The code editor contains the following HiveQL query:

```
select SUBSTRINGTree, count(*) countvalue
from(select c.ConditionsSplitted, SUBSTRING(m.tree, 1, 3) SUBSTRINGTree
from (select Id,explode (split(Conditions,"")) ConditionsSplitted
from clinicaltrial_2021) c
inner join mesh m on c.ConditionsSplitted=m.term)
group by SUBSTRINGTree
order by count(SUBSTRINGTree) desc
limit 10
```

Below the code, a table shows the results:

SUBSTRINGTree	countvalue
1 C04	143994
2 C23	136079
3 C01	106674
4 C14	94523
5 C10	92310
6 C06	85646
7 C08	70720

Showing all 10 rows.

Command took 4.90 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 27/04/2022, 01:20:52 on COURSEWORK

(3d.) Data frame

The final result is mapped in a dataframe called tree_meshDF1.it includes the joined tables. The statement ‘grouby’ and ‘sort’ with boolean expression False are used to get a frequency of the hierarchy code in descending order.

The screenshot shows a Databricks notebook interface. At the top, there are several tabs: 'Inbox (215) - aladegbola', 'Create Cluster - Data', 'BDDT 2021 DF - Data', 'APA7 citation gene', 'inequality in education', 'Plagiarism Checker', and 'New Tab'. Below the tabs, the URL is https://community.cloud.databricks.com/?o=233153991320092#notebook/930668662972426/command/930668662972445. The main area displays a table with two rows:

	Autism Spectrum Disorder	Autism Spectrum Disorder	F03
2	Autism Spectrum Disorder	Autism Spectrum Disorder	F03
3	Diabetes Mellitus	Diabetes Mellitus	C19

A message below the table says: 'Truncated results showing first 1000 rows. Click to re-execute with maximum result limits.' Below the table are some visualization icons. A note indicates: 'Command took 3.74 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 03/05/2022, 01:45:50 on COURSEWORK'.

In the 'Cmd 25' section, the following Python code is shown:

```
#The select statement is used to select the 'treecode' column with the first 3 substrings of the column '#tree' and the groupBy, Count and Sort funtions are used to count and arrange the result in a #descending order
tree_meshDF1 =joinedclinical_mesh.select("treecode").groupBy("treecode").count().sort("count",ascending=False)
tree_meshDF1.limit(10).show()
```

The output of this command is a table titled '(2) Spark Jobs' containing 10 rows of data:

treecode	count
C04	143994
C23	136079
C01	106674
C14	94523
C10	92310
C06	85646
C08	70720
C13	42599
C18	41276
C12	40161

A note below the table indicates: 'Command took 7.87 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 03/05/2022, 01:45:50 on COURSEWORK'.

The bottom of the screen shows a Windows taskbar with various icons and system status information: 'Cloudy', temperature '7°C', battery level, signal strength, 'ENG UK', '03:53', and the date '03/05/2022'.

(3d.) **Discussion of result:**

The three implementations produced the same result with the top hierarchies having C as their first substring.

(3e.) **PROBLEM ANSWERS**

Question:5

Find the 10 most common sponsors that are not pharmaceutical companies, along with the number of clinical trials they have sponsored.

(3e.) Assumptions made:

It was assumed that the Parent Company column in the pharma file contains all possible pharmaceutical companies.

(3e.)PySpark implementation outline RDD

The Column Parent_company was selected from pharma file to form pharma_Parentcoy. This was converted to an rdd called pharma_ParentcoyRDD using the command (.rdd).

```
#Selecting Parent_Company column
pharma_Parentcoy=pharma.select("Parent_company")
display(pharma_Parentcoy)

▶ (1) Spark Jobs
```

	Parent_company
1	Abbott Laboratories
2	AbbVie
3	AbbVie

Showing all 968 rows.

```
Command took 0.20 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 03/05/2022, 22:02:56 on Coursework
```

```
#conversion of pharma_Parentcoy into an rdd
pharma_ParentcoyRDD=pharma_Parentcoy.rdd
pharma_ParentcoyRDD.collect()
```

```
▶ (1) Spark Jobs
```

```
Out[178]: [Row(Parent_company='Abbott Laboratories'),
Row(Parent_company='AbbVie'),
Row(Parent_company='AbbVie'),
Row(Parent_company='Abbott Laboratories'),
Row(Parent_company='Johnson & Johnson')]
```

The number 1 was assigned to each row in pharma_ParentcoyRDD using map and lambda to get a count.

A screenshot of a Databricks notebook interface. The top navigation bar shows three tabs: "Databricks Community Edition", "BDDT 2021 RDD - Databricks Co", and "BDDT 2021 HiveQL - Databricks". The current tab is the third one. The URL in the address bar is <https://community.cloud.databricks.com/?o=233153991320092#notebook/429437441093117/command/3252998107387627>. The main area contains a code cell with the following Scala code:

```
#Assigning a count of 1 to all the items in Parent_Company
pharma_ParentcoyRDD1=pharma_ParentcoyRDD.map(lambda x: (x[0],1))
pharma_ParentcoyRDD1.collect()
```

The output section shows the result of the collect() operation:

```
▶ (1) Spark Jobs
Out[179]: [('Abbott Laboratories', 1),
('AbbVie', 1),
('AbbVie', 1),
```

headerRemvRDD1 is clinicaltrial_2021 file with the header removed.

The number 1 was assigned to each row in headerRemvRDD1 using map, lambda and split to another rdd called clinicaltrial_2021DF2. The essence of this is to get an individual count of the rows in the column sponsors.

A screenshot of a Databricks notebook interface. The top navigation bar shows three tabs: "Databricks Community Edition", "BDDT 2021 RDD - Databricks Co", and "BDDT 2021 HiveQL - Databricks". The current tab is the third one. The URL in the address bar is <https://community.cloud.databricks.com/?o=233153991320092#notebook/429437441093117/command/3252998107387627>. The main area contains a code cell with the following Scala code:

```
#Assigning a count of 1 to all the items in Conditions column in clinicaltrial_2021
clinicaltrial_2021DF2=headerRemvRDD1.map(lambda x: (x.split("|")[1], 1))
clinicaltrial_2021DF2.collect()
```

The output section shows the result of the collect() operation:

```
▶ (1) Spark Jobs
Out[180]: [('The University of Hong Kong', 1),
('Duke University', 1),
('Universidade Federal do Rio de Janeiro', 1),
('Istanbul Medeniyet University', 1),
('University of Roma La Sapienza', 1),
```

(3e.) HiveQL implementation outline

In Hive, the statements ‘select’, ‘left join’ and ‘group by’ were used to get a table with the sponsors that are not pharmaceuticals. This was achieved with a left join of the sponsors in clinicaltrial_2021 and Parent_company in Pharma file where the two columns are equal.

BDDT 2021 HiveQL (SQL)

```
select Sponsor, count(Sponsor)
from (select c.Sponsor , p.Parent_Company
      from clinicaltrial_2021 c
      left join pharma2 p on c.Sponsor=p.Parent_Company
      where p.Parent_Company is null) g
group by Sponsor
```

Command took 0.65 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 27/04/2022, 01:23:19 on COURSEWORK

Cmd 36

Sponsor	count(Sponsor)
1 Milton S. Hershey Medical Center	384
2 Goethe University	61
3 Neuroelectrics Corporation	2
4 University Hospital Schleswig-Holstein	128
5 University of Oklahoma	284
6 Oncoscience AG	4
7 Pulsion Medical Systems SE	1

Truncated results, showing first 1000 rows.
Click to re-execute with maximum result limits.

3e.)Dataframe implementation outline

The function left outer join was used to join clinicaltrial_2021 with pharma file where sponsor and parent_Company were equal.

Inbox (215) - alade | Create Cluster - D... | BDDT 2021 DF - D... | APA7 citation gen... | *inequality in educ... | Plagiarism Checke... | New Tab | https://community.cloud.databricks.com/?o=233153991320092#notebook/930668662972426/command/930668662972452

Command took 0.60 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 03/05/2022, 01:45:50 on COURSEWORK

Cmd 32

```
#Joining Clinicaltrial_2021 with pharma file
Joinedclinical_pharma8= clinicaltrial_2021.join(pharma, clinicaltrial_2021.Sponsor == pharma.Parent_Company,"leftouter")
display(Joinedclinical_pharma8)
```

Command took 1.51 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 03/05/2022, 01:45:50 on COURSEWORK

Cmd 33

ID	Sponsor	Status	Start	Complete
1	The University of Hong Kong	Recruiting	Aug 2005	Nov 2020
2	Duke University	Completed	Jul 2016	Jul 2020
3	Universidade Federal do Rio de Janeiro	Completed	Mar 2017	Jan 2018
4	Istanbul Medeniyet University	Completed	Jan 2012	Dec 2014
5	University of Roma La Sapienza	Active, not recruiting	Jun 2016	Sep 2021
6	Consorzio Futuro in Ricerca	Completed	Apr 2016	Jan 2018

Truncated results, showing first 1000 rows.
Click to re-execute with maximum result limits.

(3e.)Result on the submission (final) data –

(3e.)RDD

Clinicaltrial_JoinPharmaF was used as the rdd where clinicaltrial_JoinPharma as filtered for null values. The null values are the sponsors that are not pharmaceuticals.

```
#joining the 2 rdd together
clinicaltrial_JoinPharma=clinicaltrial_2021DF2.leftOuterJoin(pharma_ParentcoRDD1)
clinicaltrial_JoinPharma.take(3)
```

▶ (1) Spark Jobs

```
Out[151]: [('Duke University', (1, None)), ('Duke University', (1, None)), ('Duke University', (1, None))]
```

Command took 3.79 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 09/05/2022, 17:46:42 on Coursework

```
#Use of filter function to get sponsors that are not pharmaceuticals
clinicaltrial_JoinPharmaF=clinicaltrial_JoinPharma.filter(lambda x: (x[1][1] ==None))
clinicaltrial_JoinPharmaF.take(3)
```

▶ (1) Spark Jobs

```
Out[150]: [('Duke University', (1, None)), ('Duke University', (1, None)), ('Duke University', (1, None))]
```

Command took 0.18 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 09/05/2022, 17:46:34 on Coursework

The reduceByKey, sortBy, and Boolean expression False was used to arrange the result in a descending order.

```
#non-pharmaceutical sponsors with their frequency arranged in descending order
clinicaltrial_JoinPharmaFS =clinicaltrial_JoinPharmaF.map(lambda x: (x[0], 1))\
.reduceByKey(lambda c,d:c+d)\
.sortBy(lambda x:x[1],False)
clinicaltrial_JoinPharmaFS.take(10)
```

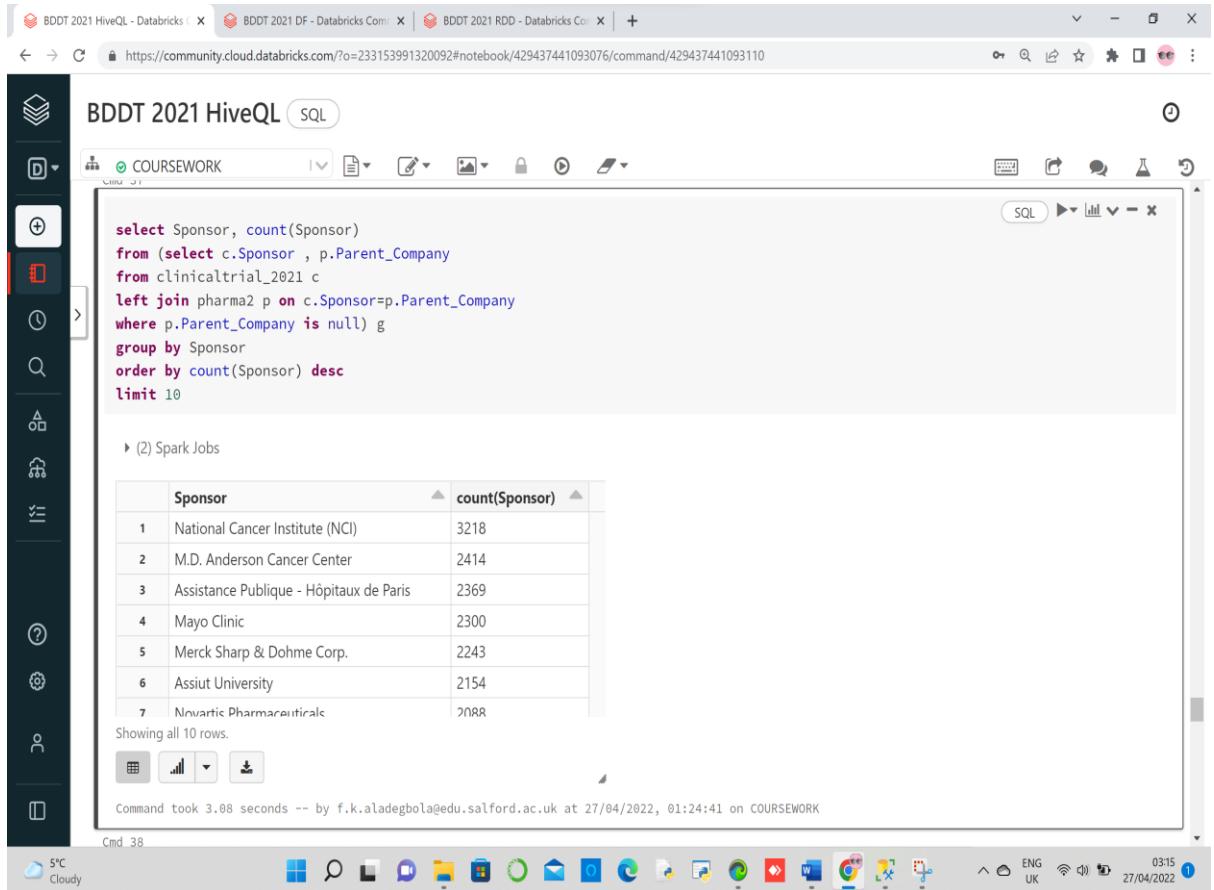
▶ (3) Spark Jobs

```
Out[137]: [('National Cancer Institute (NCI)', 3218), ('M.D. Anderson Cancer Center', 2414), ('Assistance Publique - Hôpitaux de Paris', 2369), ('Mayo Clinic', 2300), ('Merck Sharp & Dohme Corp.', 2243), ('Assiut University', 2154), ('Novartis Pharmaceuticals', 2088), ('Massachusetts General Hospital', 1971), ('Cairo University', 1928), ('Hoffmann-La Roche', 1828)]
```

Command took 1.78 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 09/05/2022, 16:52:35 on Coursework

(3e.)HIVE

group by, order by and limit were used to arrange the result in descending order.



The screenshot shows a Databricks notebook interface titled "BDDT 2021 HiveQL". The notebook contains the following SQL query:

```
select Sponsor, count(Sponsor)
from (select c.Sponsor , p.Parent_Company
from clinicaltrial_2021 c
left join pharma2 p on c.Sponsor=p.Parent_Company
where p.Parent_Company is null) g
group by Sponsor
order by count(Sponsor) desc
limit 10
```

Below the query, there is a section titled "(2) Spark Jobs" which displays a table of sponsor counts:

	Sponsor	count(Sponsor)
1	National Cancer Institute (NCI)	3218
2	M.D. Anderson Cancer Center	2414
3	Assistance Publique - Hôpitaux de Paris	2369
4	Mayo Clinic	2300
5	Merck Sharp & Dohme Corp.	2243
6	Assiut University	2154
7	Novartis Pharmaceuticals	2088

At the bottom of the notebook, it says "Showing all 10 rows." Below the notebook, the status bar shows "Command took 3.08 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 27/04/2022, 01:24:41 on COURSEWORK".

(3e.)DATAFRAME

In Dataframe implementation, the functions filter where Parent_company was null was used, with groupBy and sort to get a frequency in descending order of sponsors that are not pharmaceuticals.

```

BDDT 2021 DF Python
Detached File View: Standard Permissions Run All Clear
https://community.cloud.databricks.com/?o=233153991320092#notebook/930668662972426/command/288380021999343
from pyspark.sql.functions import *
Joinedclinical_pharma8=Joinedclinical_pharma8.select(Joinedclinical_pharma8.Sponsor,Joinedclinical_pharma8.Parent_Company)
display(Joinedclinical_pharma9)

(1) Spark Jobs
+-----+-----+
| Sponsor | Parent_Company |
+-----+-----+
| 1 The University of Hong Kong | null |
| 2 Duke University | null |
| 3 Universidade Federal do Rio de Janeiro | null |
+-----+-----+
Truncated results, showing first 1000 rows.
Click to re-execute with maximum result limits.
+-----+-----+
| Sponsor | count |
+-----+-----+
| National Cancer I... | 3218 |
| M.D. Anderson Can... | 2414 |
| Assistance Publique... | 2369 |
| Mayo Clinic | 2300 |
| Merck Sharp & Doh... | 2243 |
+-----+-----+
Command took 1.88 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 03/05/2022, 01:45:50 on COURSEWORK

```

Cmd 35

```

Python ▶ v - x
selection of Sponsors from the table Joinedclinical_pharma8 where it is null in Parent company,
#Counted and arranged in descending order.This gives sponsors who are not pharmaceuticals.
from pyspark.sql.functions import *
Joinedclinical_pharmaNULL=Joinedclinical_pharma9.filter("Parent_Company is NULL").groupBy("Sponsor").count().sort(col("count").desc())
Joinedclinical_pharmaNULL.show(10)
#Using the function Show(10) gives the first 10 sponsors

(2) Spark Jobs
+-----+-----+
| Sponsor | count |
+-----+-----+
| National Cancer I... | 3218 |
| M.D. Anderson Can... | 2414 |
| Assistance Publique... | 2369 |
| Mayo Clinic | 2300 |
| Merck Sharp & Doh... | 2243 |
+-----+-----+

```

(3e.)Discussion of result:

National Cancer Institute (NCI) emerged as the non-pharmaceutical sponsor with the highest number of clinical trials. This was clearly shown by the result of the 3 implementations. This shows that quite a lot of the clinical trials were cancer-related and the sponsors are keen on sponsoring clinical trials to get results for drugs to be used.

(3f.) PROBLEM ANSWERS

Question:6

Plot the number of completed studies each month in a given year – for the submission dataset, the year is 2021. You need to include your visualization as well as a table of all the values you have plotted for each month.

(3f.) Assumptions made:

It was assumed that the completed studies for each year were properly recorded and there was no error in documentation

(3f.) PySpark implementation outline RDD

Using take() on splitRDD the rdd from clinicaltrial_2021 turned to a list, gives a view of the items in the data according to columns. x[0] represents the Id, x[1]represents sponsors e.t.c

The screenshot shows a Databricks notebook interface with several tabs at the top: BDDT 2021 DF - D, BDDT 2021 HiveQL, BDDT 2021 RDD, Create Cluster - D, Wordtune Editor, Providing data, Modules, and a blank tab. The main area contains the following code and output:

```
#shows the contents of clinicaltrial_2021 has been turned to a list
splitRDD.take(3)
```

```
Out[166]: [[{'NCT02758028', 'The University of Hong Kong', 'Recruiting', 'Aug 2005', 'Nov 2021', 'Interventional', 'Apr 2016', '', ''}, {'NCT02751957', 'Duke University', 'Completed', 'Jul 2016', 'Jul 2020', 'Interventional', 'Apr 2016', 'Autistic Disorder,Autism Spectrum Disorder', ''}, {'NCT02758483', 'Universidade Federal do Rio de Janeiro', 'Completed'}]]
```

Below the code, a note states: "Command took 0.81 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 09/05/2022, 20:07:59 on Coursework". The bottom status bar shows the date as 09/05/2022 and the time as 20:09:09. The system status indicates "ENG UK" and "Light rain".

Filter function was used to extract the needed columns and also to specify the year of completed studies as 2021.

Thereafter, each month was given a count of 1, then reduceByKey and Sort was used to calculate the total frequency and to arrange the months of completed studies in descending order.

```
#filter is used to extract the needed columns
ComStudyRDD1= splitRDD.filter(lambda x: x[2] == "Completed").filter(lambda x:x[4] != "").map(lambda x: x[4].split(" "))
ComStudyRDD1.take(5)

▶ (1) Spark Jobs
Out[140]: [['Jul', '2020'],
['Jan', '2018'],
['Dec', '2014'],
['Jan', '2018'],
['Jul', '2017']]
Command took 0.77 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 09/05/2022, 16:52:35 on Coursework
Cmd 52

#filter was used to specify the year as 2021
ComStudyRDD2=ComStudyRDD1.filter(lambda x: x[1] == "2021").map(lambda x: (x[0],1))
ComStudyRDD2.take(5)

▶ (1) Spark Jobs
Out[156]: [('Jan', 1), ('Jun', 1), ('Mar', 1), ('Jan', 1), ('May', 1)]
Command took 0.93 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 09/05/2022, 19:42:32 on Coursework
Cmd 53

#Arrangement of the frequencies of the months of completed 2021 clinical trials in descending order
ComStudyRDD3=ComStudyRDD2.reduceByKey(lambda x,y:x+y).sortBy(lambda x:[1],False)
ComStudyRDD3.take(5)

▶ (4) Spark Jobs
Out[157]: [('Mar', 1227), ('Jan', 1131), ('Jun', 1094), ('May', 984), ('Apr', 967)]
Command took 3.52 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 09/05/2022, 19:42:47 on Coursework
```

(3f.) HiveQL implementation outline

The select * statement meaning to select all was used to recall the data in clinicaltrial_2021. The function substring was used to get the months and the year and the 'where' clause specified the year as 2021.

Question 6. Plot number of completed studies each month in a given year – for the submission dataset, the year is 2021. You need to include your visualization as well as a table of all the values you have plotted for each month.

```
Cmd 39
select * from clinicaltrial_2021

▶ (1) Spark Jobs
```

ID	Sponsor	Status	Start	Completion	Type	Submission
1	The University of Hong Kong	Recruiting	Aug 2005	Nov 2021	Interventional	Apr 2016

Truncated results, showing first 1000 rows.
Click to re-execute with maximum result limits.

```
Cmd 40
select monthly_completion, count(*) as Y2021
from (select Status
       ,Completion
       ,substring(Completion,1,3) monthly_completion
     from Clinicaltrial_2021
      where substring(Completion,4,8)=2021 and status="Completed")
group by monthly_completion
order by count(*) desc
```

▶ (2) Spark Jobs

monthly_completion	Y2021
1 Mar	1227
2 Jan	1131
3 Jun	1094
4 May	984
5 Apr	967
6 Enh	028

(3f.) Dataframe implementation outline

The select statement and substring were also used in dataframes to get a table for 2021 completion monthly. Substring(Completion 5,8) brings out the year.

```
#Selection of Status column containing 'completed_studies' with Completion column containing the 'dates'
clinicaltrial_completedDF=clinicaltrial_2021.select('Status', 'Completion')
display(clinicaltrial_completedDF)
```

Status	Completion
1	Recruiting
2	Completed

Truncated results, showing first 1000 rows.
Click to re-execute with maximum result limits.

```
#The substring function is used to split the Completion column to separate the year from the month
completedDF= clinicaltrial_completedDF.select("Status",'Completion', substring('Completion', 5,8).alias("Year"))
display(completedDF)
completedDF.count()
```

Status	Completion	Year
1	Nov 2021	2021
2	Jul 2020	2020
3	Jan 2018	2018

Truncated results, showing first 1000 rows.
Click to re-execute with maximum result limits.

completedDF is filtered to specify status as completed only while

completedDF1 is filtered for the specific year to be 2021.

```
#Filter statement is used to specifically bring out studies completed in the year 2021
completedDF1=completedDF.filter(completedDF.Status == "Completed")
completedDF2 =completedDF1.filter(completedDF.Year ==2021)
display(completedDF2)
```

Status	Completion	Year
1	Jan 2021	2021
2	Jun 2021	2021
3	Mar 2021	2021

Truncated results, showing first 1000 rows.
Click to re-execute with maximum result limits.

```
#The substring function applied to Completion column is used to produce a table with the months alone
completedDF3= completedDF2.select('Status','Completion', substring('Completion', 1,3).alias("Month"))
display(completedDF3)
```

Status	Completion	Month
1	Jan 2021	Jan
2	Jun 2021	Jun
3	Mar 2021	Mar

(3f.) Result on the submission (final) data –

(3f.) RDD: The reduceByKey and SortBy with the Boolean value False was used to arrange the months of completed study in 2021.

The screenshot shows a Databricks notebook interface with several open tabs at the top. The main area displays a command cell (Cmd 54) with the following text:

```
Command took 4.14 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 09/05/2022, 14:01:08 on Coursework
```

Below the command cell, a text block states:

The above codes can be combined as shown below

Another command cell (Cmd 55) contains the following Scala code:

```
ComStudyRDD= splitRDD.filter(lambda x: x[2] == "Completed").filter(lambda x:x[4] != "").map(lambda x: x[4].split(" "))\n.filter(lambda x: x[1] == "2021").map(lambda x: (x[0],1)).reduceByKey(lambda x,y:x+y).sortBy(lambda x:x[1],False)\nComStudyRDD.take(12)
```

After running the code, the output (Out[39]) shows a list of tuples representing months and their counts:

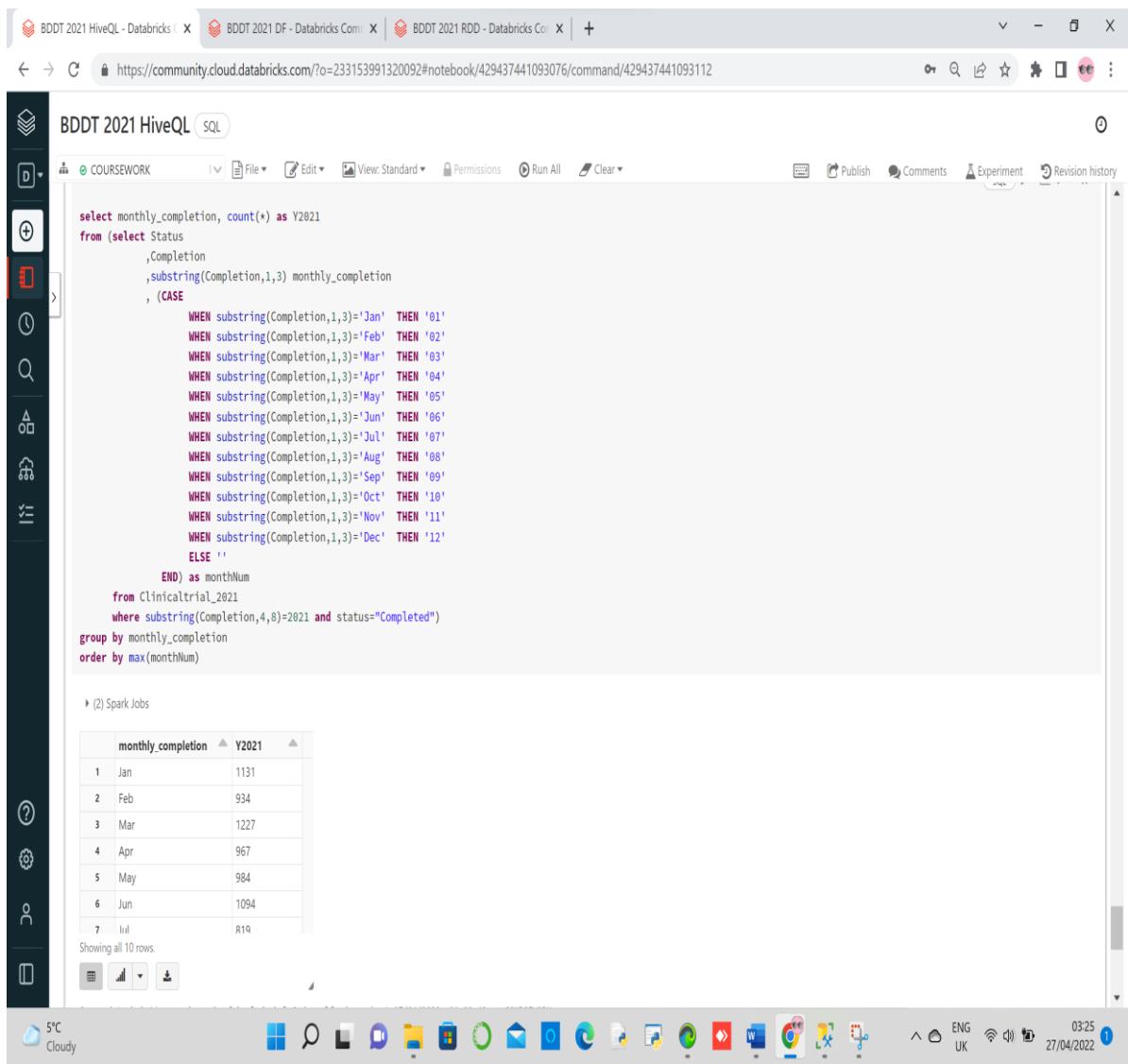
```
Out[39]: [('Mar', 1227),\n('Jan', 1131),\n('Jun', 1094),\n('May', 984),\n('Apr', 967),\n('Feb', 934),\n('Jul', 819),\n('Aug', 700),\n('Sep', 528),\n('Oct', 187)]
```

At the bottom of the notebook, a status bar shows the following information:

- 16°C Rain showers
- Windows taskbar icons
- Network connection: ENG UK
- Date and time: 09/05/2022 14:52

(3f.) **HIVE**

Count, substring group by and order by were used to arrange the months in the calendar arrangement.



The screenshot shows a Databricks notebook interface with three tabs at the top: "BDDT 2021 HiveQL - Databricks", "BDDT 2021 DF - Databricks Com", and "BDDT 2021 RDD - Databricks Co". The current tab is "BDDT 2021 HiveQL". The URL in the browser bar is <https://community.cloud.databricks.com/?o=23315399132009#notebook/429437441093076/command/429437441093112>. The notebook content is a HiveQL query:

```
select monthly_completion, count(*) as Y2021
from (select Status
      ,Completion
      ,substring(Completion,1,3) monthly_completion
      , (CASE
          WHEN substring(Completion,1,3)='Jan' THEN '01'
          WHEN substring(Completion,1,3)='Feb' THEN '02'
          WHEN substring(Completion,1,3)='Mar' THEN '03'
          WHEN substring(Completion,1,3)='Apr' THEN '04'
          WHEN substring(Completion,1,3)='May' THEN '05'
          WHEN substring(Completion,1,3)='Jun' THEN '06'
          WHEN substring(Completion,1,3)='Jul' THEN '07'
          WHEN substring(Completion,1,3)='Aug' THEN '08'
          WHEN substring(Completion,1,3)='Sep' THEN '09'
          WHEN substring(Completion,1,3)='Oct' THEN '10'
          WHEN substring(Completion,1,3)='Nov' THEN '11'
          WHEN substring(Completion,1,3)='Dec' THEN '12'
          ELSE ''
        END) as monthNum
     from Clinicaltrial_2021
      where substring(Completion,4,8)=2021 and status="Completed")
group by monthly_completion
order by max(monthNum)
```

The output of the query is a table:

monthly_completion	Y2021
1 Jan	1131
2 Feb	934
3 Mar	1227
4 Apr	967
5 May	984
6 Jun	1094
7 Jul	819

Showing all 10 rows.

(3f.) **DATA FRAME:** The use of groupBy and count was employed to get the calender months of completed study in 2021.

The screenshot shows a Databricks notebook interface. At the top, there's a header bar with various tabs and a URL: <https://community.cloud.databricks.com/?o=233153991320092#notebook/930668662972426/command/930668662972463>. Below the header is a command history section labeled "Cmd 44" with a timestamp: "Command took 1.10 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 03/05/2022, 02:25:06 on COURSEWORK". The main area displays a Scala code snippet and its output:

```
#The groupBy() and count()function is employed to group the months together and count the total of each #month
completedDF4=completedDF3.groupBy("Month","Status").count()
display(completedDF4)
```

▶ (2) Spark Jobs

	Month	Status	count
1	Sep	Completed	528
2	Apr	Completed	967
3	Oct	Completed	187
4	Jul	Completed	819
5	Aug	Completed	700
6	Jun	Completed	1094
...

Showing all 10 rows.

At the bottom of the screen, there's a taskbar with icons for various applications like File Explorer, Task View, and a search bar. The system tray shows the date and time: 03/05/2022 04:02.

- **Discussion of result:**

	monthly_completion	Y2021
1	Mar	1227
2	Jan	1131
3	Jun	1094
4	May	984
5	Apr	967
6	Feb	934
7	Jul	819
8	Aug	700
9	Sep	528
10	Oct	187

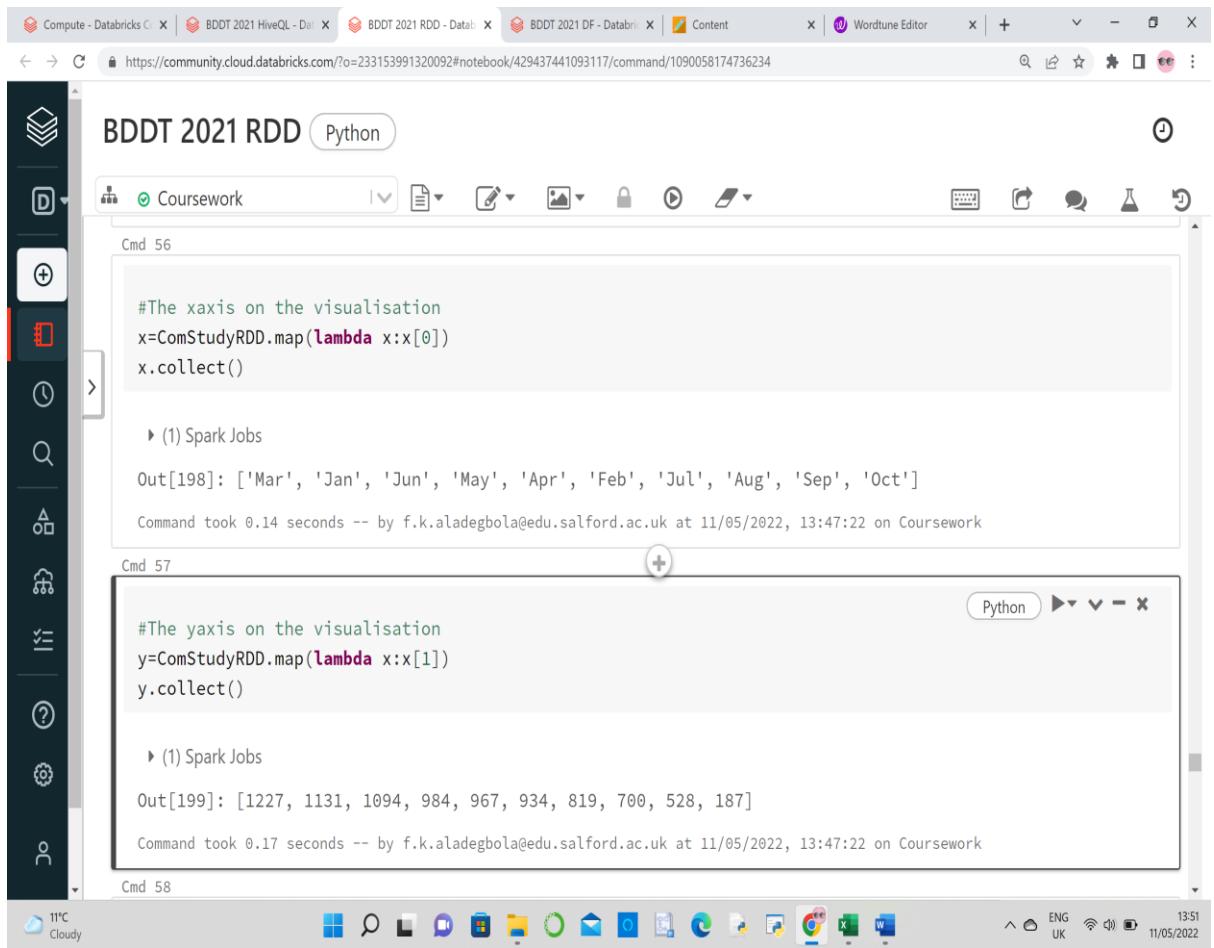
The highest number of completed studies took place in the month of March, followed by January and then June.

(3f.) VISUALISATIONS

RDD

Matplotlib was imported for visualization in RDD.

Matplotlib is a Python package that facilitates creating static, animated, and interactive #visualizationsThe x-axis and y-axis were selected as shown below.



```
#The xaxis on the visualisation
x=ComStudyRDD.map(lambda x:x[0])
x.collect()

▶ (1) Spark Jobs

Out[198]: ['Mar', 'Jan', 'Jun', 'May', 'Apr', 'Feb', 'Jul', 'Aug', 'Sep', 'Oct']

Command took 0.14 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 11/05/2022, 13:47:22 on Coursework

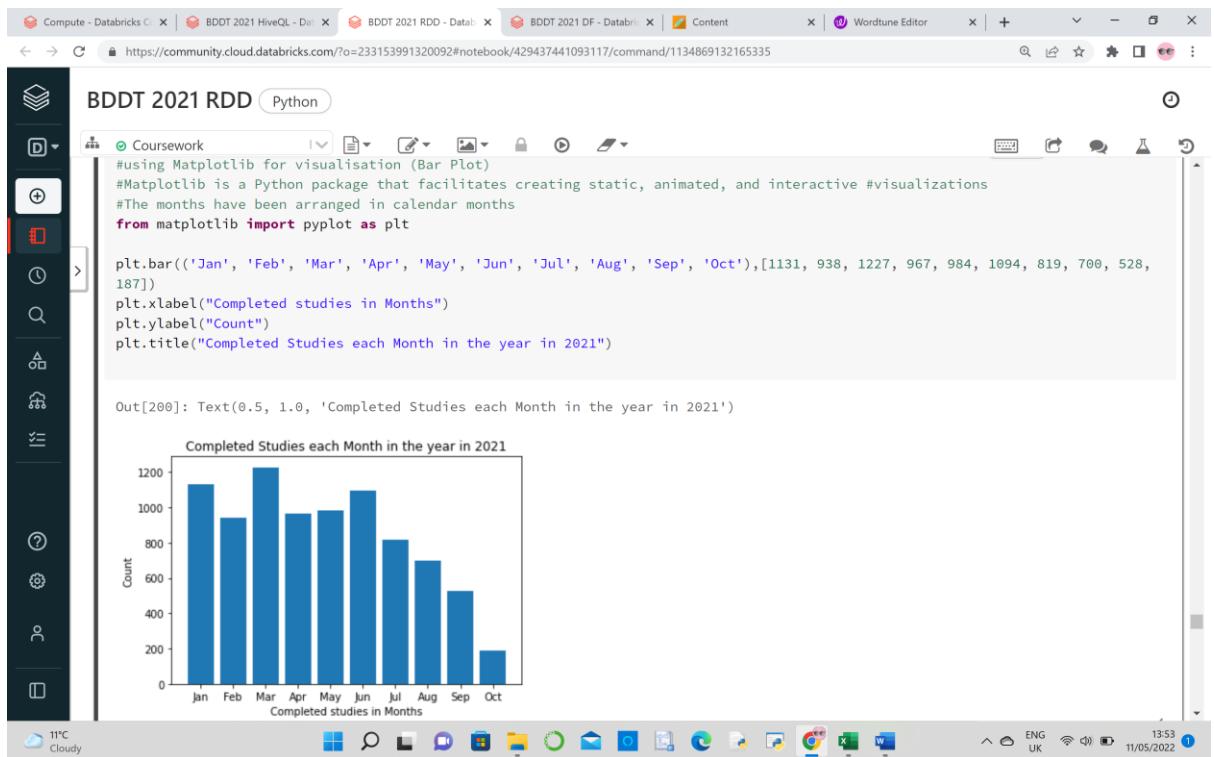
#The yaxis on the visualisation
y=ComStudyRDD.map(lambda x:x[1])
y.collect()

▶ (1) Spark Jobs

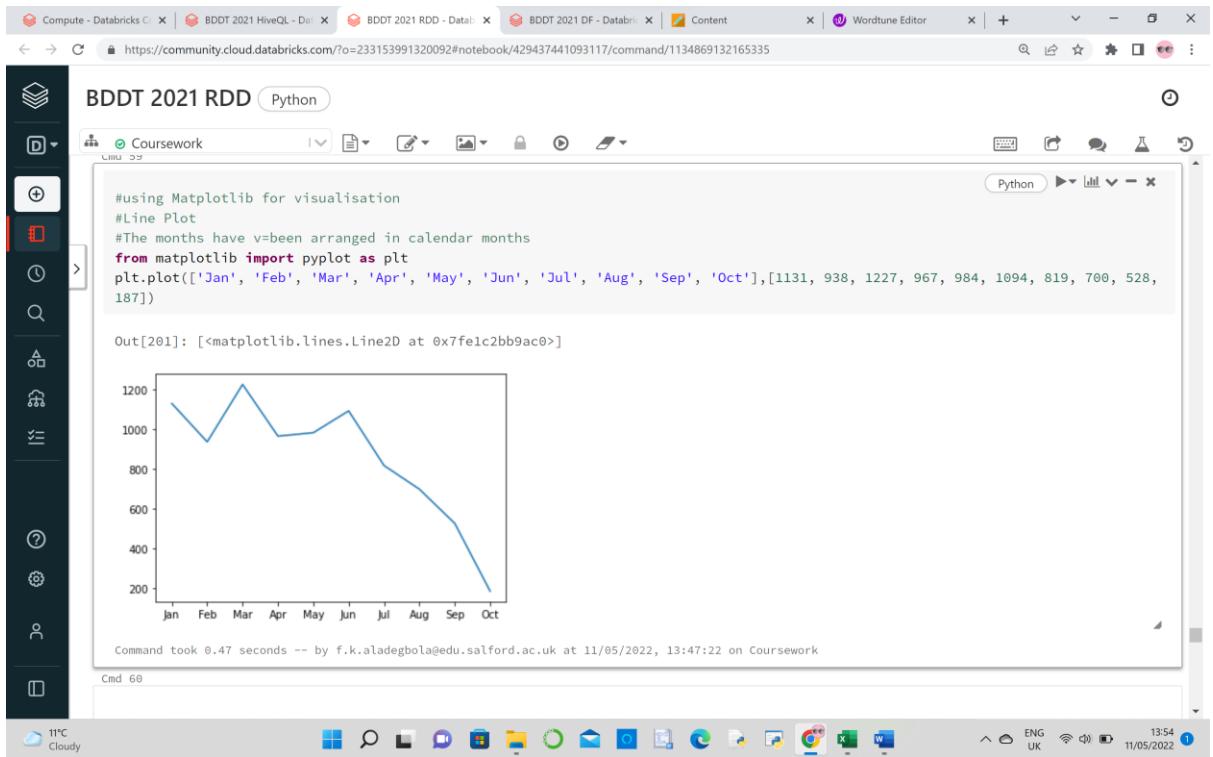
Out[199]: [1227, 1131, 1094, 984, 967, 934, 819, 700, 528, 187]

Command took 0.17 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 11/05/2022, 13:47:22 on Coursework
```

The bar plot in rdd: Matplotlib was used for visualisation (Bar Plot)

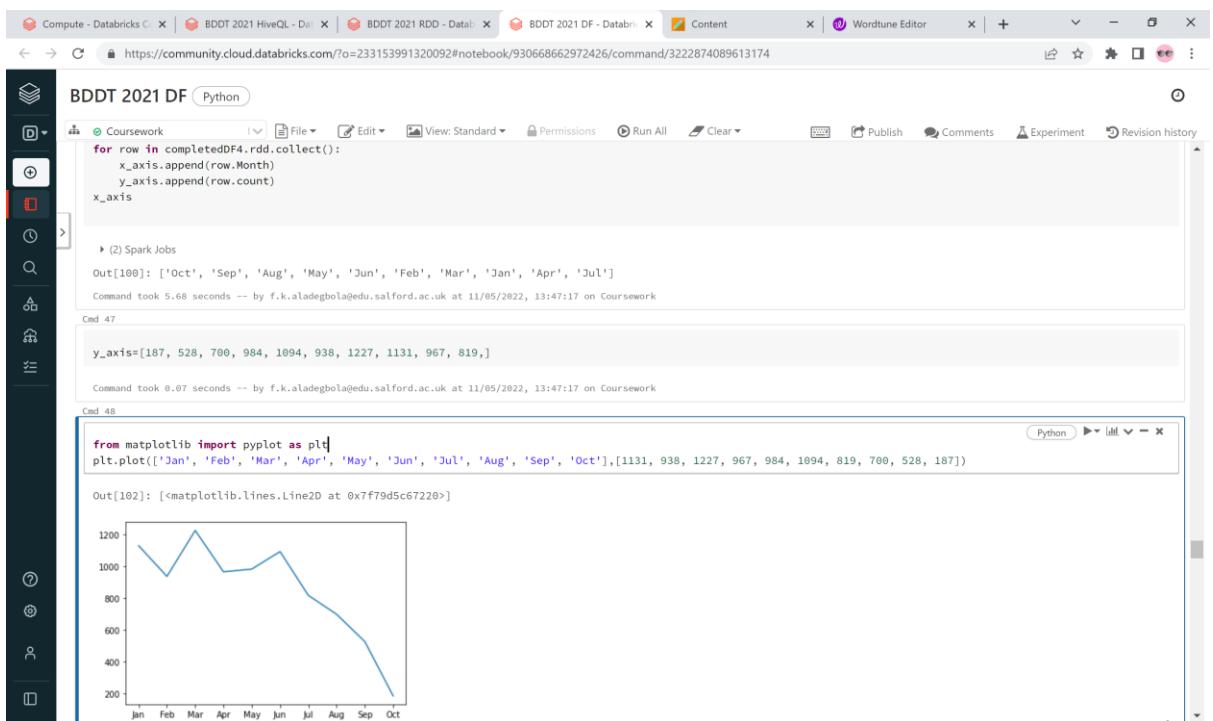


The Line Plot in rdd: Matplotlib

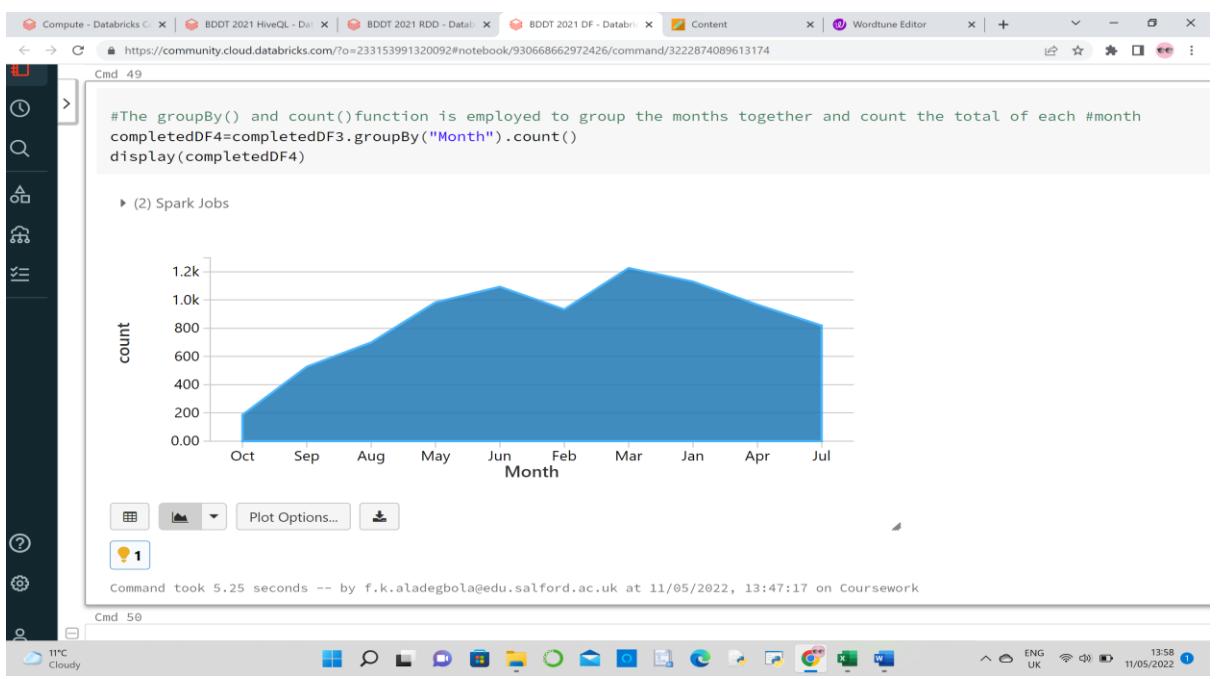


DATAFRAME: Matplotlib was used for visualisation

The Line plot in dataframe showed March as the highest month of completed clinical studies in 2021. It has the highest point on the line plot.



The Area Inbuilt Visualization in Dataframe

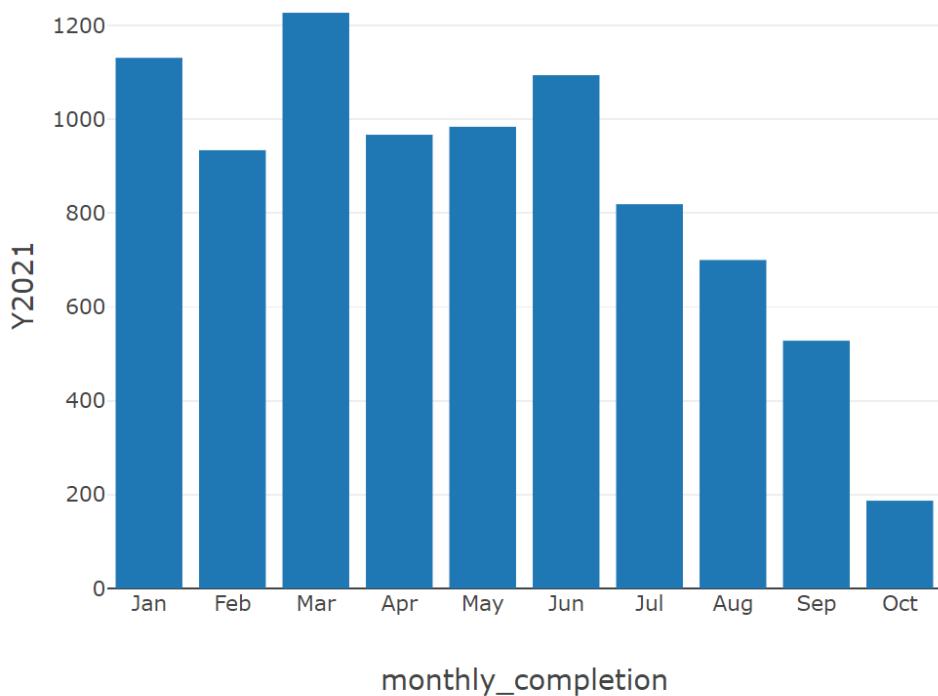


HIVEQL

The Area and Pie chart Inbuilt visualization was used in Hiveql

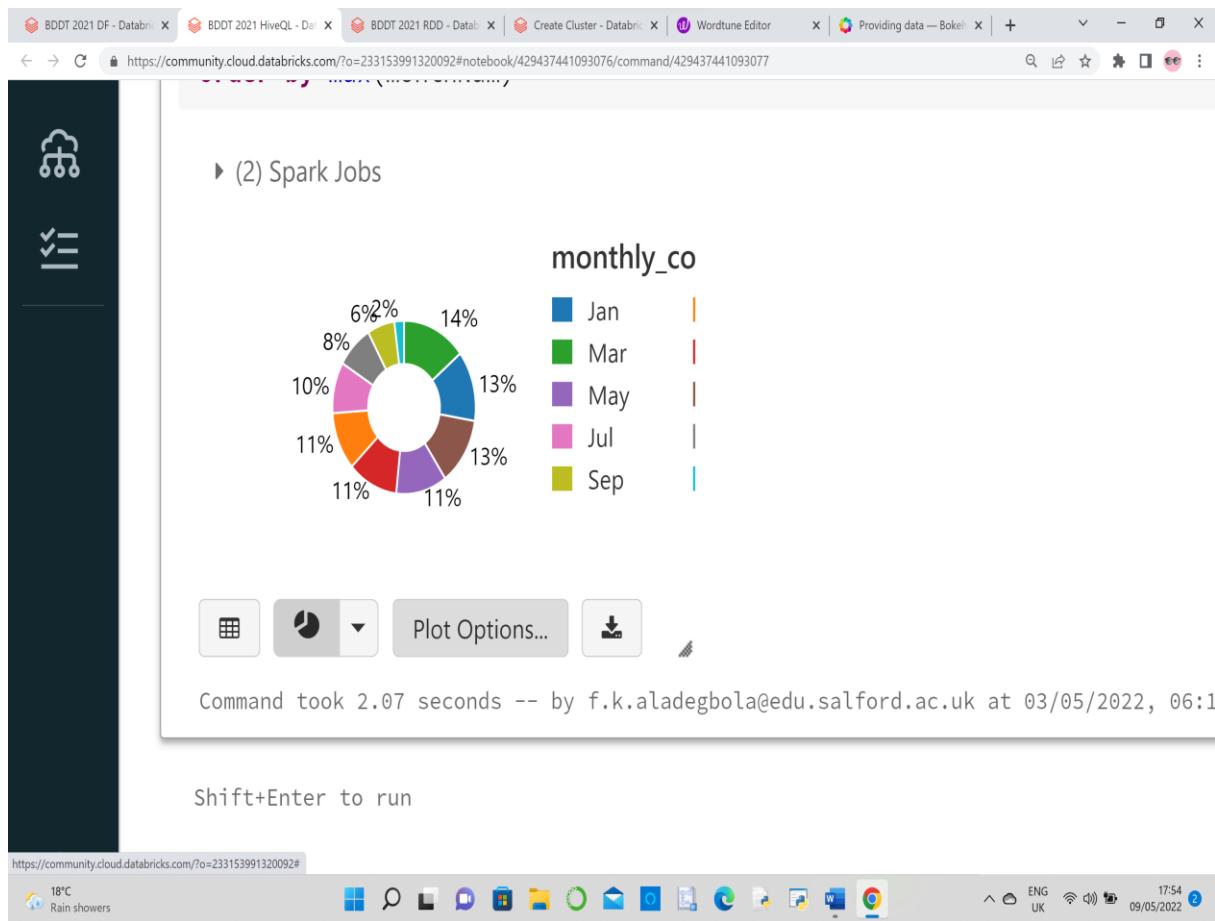
The bar chart showed March as the month with the highest number of completed clinical studies in 2021.

► (2) Spark Jobs



Command took 1.60 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 03/05/2022, 06:10:50 on COURSEWORK

The pie chart showed March as the highest month of a completed clinical study by assigning the highest percentile 14%.



FURTHER ANALYSIS 1

DATAFRAME

Comparison between completed studies and uncompleted studies in terms of count.

A count of completed studies in 2021 was done against the uncompleted studies.

Completed studies showed a count of 8571 while the uncompleted study showed a count of 27,056. This shows that more effort should be geared towards more studies completed. There is a need for research into the reason for a lower number of the completed study.

Inbox (215) - alade... | Create Cluster - D... | BDDT 2021 DF - D... | APA7 citation gene... | "inequality in educ... | Plagiarism Checker... | New Tab

https://community.cloud.databricks.com/?o=233153991320092#notebook/930668662972426/command/930668662972472

Command skipped
Command took 1.78 minutes -- by f.k.aladegbola@edu.salford.ac.uk at 03/05/2022, 01:45:50 on COURSEWORK

Cmd 51

7.Further Investigations: What is the comparison between completed studies and uncompleted studies?

Cmd 52

```
#Filter statement is used to specifically bring out studies completed in the year 2021
completedDF1=completedDF.filter(completedDF.Status == "Completed")
completedDF2 =completedDF1.filter(completedDF.Year ==2021)
display(completedDF2)
completedDF2.count()
```

▶ (4) Spark Jobs

Status	Completion	Year
1	Completed	Jan 2021
2	Completed	Jun 2021
3	Completed	Mar 2021
4	Completed	Jan 2021

Truncated results, showing first 1000 rows.
Click to re-execute with maximum result limits.

Out[50]: 8571

💡

Command took 3.91 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 03/05/2022, 02:30:20 on COURSEWORK

Cmd 53

7°C Mostly cloudy

04:07 03/05/2022

Inbox (215) - alade... | Create Cluster - D... | BDDT 2021 DF - D... | APA7 citation gene... | "inequality in educ... | Plagiarism Checker... | New Tab

https://community.cloud.databricks.com/?o=233153991320092#notebook/930668662972426/command/930668662972472

Command took 3.91 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 03/05/2022, 02:30:20 on COURSEWORK

Cmd 53

```
#Filter statement is used to specifically bring out uncompleted studies in the year 2021
completedDFx=completedDF.filter(completedDF.Status != "Completed")
completedDFy =completedDFx.filter(completedDF.Year ==2021)
display(completedDFy)
completedDFy.count()
```

▶ (3) Spark Jobs

Status	Completion	Year
1	Recruiting	Nov 2021
2	Active, not recruiting	Dec 2021
3	Active, not recruiting	Oct 2021
4	Terminated	Feb 2021
5	Unknown status	Mar 2021

Truncated results, showing first 1000 rows.
Click to re-execute with maximum result limits.

Out[52]: 27056

Command took 2.84 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 03/05/2022, 02:31:16 on COURSEWORK

Cmd 54

Completed studies is 8571 while uncompleted studies is 27056. This shows the rate of uncom much lower than the completed studies

Cmd 55

7°C Mostly cloudy

04:07 03/05/2022

RDD IMPLEMENTATION

Compute - Databricks | BDDT 2021 HiveQL - Data | BDDT 2021 RDD - Data | BDDT 2021 DF - Data | Content | Wordtune Editor | + | - | X

← → C https://community.cloud.databricks.com/?o=233153991320092#notebook/429437441093117/command/3222874089613167

BDDT 2021 RDD Python

Comparison of Monthly Completed and Uncompleted studies in the year 2021

Cmd 62

```
#getting uncompleted studies in the year 2021
UNComStudyRDD=splitRDD.filter(lambda x: x[2] != "Completed").filter(lambda x:x[4] != "").map(lambda x: x[4].split(" "))
.filter(lambda x: x[1] == "2021").map(lambda x: (x[0],1)).reduceByKey(lambda x,y:x+y).sortBy(lambda x:x[1],False)
UNComStudyRDD.take(12)
```

Out[282]: [('Dec', 9968), ('Sep', 2373), ('Oct', 2327), ('Jun', 2688), ('Nov', 1997), ('Aug', 1603), ('Jul', 1589), ('Mar', 1220), ('May', 1128), ('Apr', 1062), ('Jan', 980), ('Feb', 801)]

Command took 4.89 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 11/05/2022, 13:47:22 on Coursework

Cmd 63

```
#The xaxis on the visualisation
UnCompletedx=UNComStudyRDD.map(lambda x:x[0])
UnCompletedx.collect()
```

Out[283]: ['Dec', 'Sep', 'Oct', 'Jun', 'Nov', 'Aug', 'Jul', 'Mar', 'May', 'Apr', 'Jan', 'Feb']

Command took 0.15 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 11/05/2022, 13:47:22 on Coursework

Compute - Databricks | BDDT 2021 HiveQL - Data | BDDT 2021 RDD - Data | BDDT 2021 DF - Data | Content | Wordtune Editor | + | - | X

← → C https://community.cloud.databricks.com/?o=233153991320092#notebook/429437441093117/command/3222874089613167

Cmd 64

```
#The xaxis on the visualisation
UnCompletedy=UNComStudyRDD.map(lambda x:[1])
UnCompletedy.collect()
```

Out[204]: [9968, 2373, 2327, 2008, 1997, 1603, 1589, 1220, 1128, 1062, 980, 801]

Command took 0.17 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 11/05/2022, 13:47:22 on Coursework

Cmd 65

```
#using Matplotlib for visualisation
#Line Plot
#The months have vbeen arranged in calendar months
from matplotlib import pyplot as plt
plt.plot(['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun', 'Jul', 'Aug', 'Sep', 'Oct', 'Nov', 'Dec'],[980, 801, 1220, 1062, 1128, 2008, 1589, 1603, 2373, 2327, 1997, 9968])
```

Out[205]: [`<matplotlib.lines.Line2D at 0x7fe1c2b80c70>`]

13°C Partly sunny

ENG UK 14:10 11/05/2022

There were no completed studies in November and December 2021, hence as shown in the line plot above uncompleted studies reached its peak in November and December 2021. It is advised that a feasibility study should be put in place to check the cause of a high rate of uncompletion at the end of the year and measures should be put in place to mitigate this.

FURTHER ANALYSIS 2

DATAFRAME

What are the top 10 Interventions from Interventions and their frequencies?

```
#The select statement is used to view the contents of the Interventions statement
from pyspark.sql.functions import *
interventionsDFexplode=clinicalcare_2021.select ("Interventions")
display(interventionsDFexplode)
```

Interventions
1 null
2 null
3 null
4 null
5 null
6 Fluticasone,Xhance,Budesonide,Formoterol Fumarate,Salmeterol Xinafoate

Truncated results, showing first 1000 rows.
Click to re-execute with maximum result limits.

Command took 0.28 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 03/05/2022, 02:37:40 on COURSEWORK

Interventions needs to be exploded

Interventions needs to be exploded

```
#Interventions column needs to be splitted because there are more then one in some rows. The use of #explode and split funtion was employed to achieve this
from pyspark.sql.functions import *
interventionsDFExplode1=clinicaltrial_2021.select(explode(split(col("Interventions"),",")).alias("Interventions"))
display(interventionsDFExplode1)
```

▶ (1) Spark Jobs

Interventions
1 Fluticasone
2 Xhance

Truncated results, showing first 1000 rows.
Click to re-execute with maximum result limits.

Command took 0.39 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 03/05/2022, 02:39:56 on COURSEWORK

```
#The groupby() and count() function gives the interventions with their frequencies, while sort() with declaring ascending as False arranges it in a descending order
InterventionsDFExplode_count = interventionsDFExplode1.groupBy("Interventions").count().sort("count",ascending=False)
InterventionsDFExplode_count .limit(10).show()
#limit(10) and show() specifies the first 10 and displays it respectively
```

▶ (2) Spark Jobs

Interventions	count
Paclitaxel	3225
Cyclophosphamide	3012
Dexamethasone	2516
Carboplatin	2392
Antibodies	2335
Vaccines	2214
Gemcitabine	2195
Bevacizumab	2005

7°C Mostly cloudy

FURTHER ANALYSIS 3

DATAFRAME

Who are the first 5 sponsors in the year 2021 with the highest frequency?

Assiut University emerged as the highest sponsor in 2021 with a count of 447 clinical trials sponsored.

Inbox (215) - alade... | Create Cluster - D... | BDDT 2021 DF - D... | APA7 citation gene... | "inequality in educ... | Plagiarism Checke... | New Tab

<https://community.cloud.databricks.com/>

9. Further Investigations :Who are the first five Sponsors in the year 2021?

```
#Selection of the Sponsor and Status columns
#HighestSponsorDF=clinicaltrial_2021.select("clinicaltrial_2021.Sponsor, clinicaltrial_2021.Status, clinicaltrial_2021.Completion")
display(HighestSponsorDF)
```

(1) Spark Jobs

Sponsor	Status	Completion
1 The University of Hong Kong	Recruiting	Nov 2021
2 Duke University	Completed	Jul 2020
3 Universidade Federal do Rio de Janeiro	Completed	Jan 2018
4 Istanbul Medeniyet University	Completed	Dec 2014
5 University of Roma La Sapienza	Active, not recruiting	Sep 2020
6 Consorcio Futuro In Ricerca	Completed	Jan 2018

Truncated results, showing first 1000 rows.
Click to re-execute with maximum result limits.

Command took 0.36 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 03/05/2022, 02:43:08 on COURSEWORK

Cmd 63

```
#Use of substring to create the sponsors in the year 2021
#HighestSponsorDF1= HighestSponsorDF.select("Sponsor", substring("Completion", 5,8).alias("Year_of_Sponsor"))
display(HighestSponsorDF1)
```

(1) Spark Jobs

Sponsor	Year_of_Sponsor
1 The University of Hong Kong	2021
2 Duke University	2020
3 Universidade Federal do Rio de Janeiro	2018
4 Istanbul Medeniyet University	2014
5 University of Roma La Sapienza	2020
6 Consorcio Futuro In Ricerca	2018

Truncated results, showing first 1000 rows.
Click to re-execute with maximum result limits.

Command took 0.42 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 03/05/2022, 02:43:16 on COURSEWORK

Cmd 64

```
#Use of the function Filter to select the sponsors in the year only
#HighestSponsorDF2= HighestSponsorDF1.filter(HighestSponsorDF1.Year_of_Sponsor ==2021)
display(HighestSponsorDF2)
```

(1) Spark Jobs

Sponsor	Year_of_Sponsor
1 The University of Hong Kong	2021
2 Orphazyme	2021

Truncated results, showing first 1000 rows.
Click to re-execute with maximum result limits.

7°C Mostly cloudy

Inbox (215) - alade... | Create Cluster - D... | BDDT 2021 DF - D... | APA7 citation gene... | "inequality in educ... | Plagiarism Checke... | New Tab

<https://community.cloud.databricks.com/>

Cmd 65

```
#Use of groupBy(),Count(),sort() and desc() to get all the sponsorsin 2021
from pyspark.sql.functions import *
HighestSponsorDF3=HighestSponsorDF2.groupBy("Sponsor").count().sort(col("count").desc())
display(HighestSponsorDF3)
```

(2) Spark Jobs

Sponsor	count
1 Assiut University	437
2 Cairo University	369

Truncated results, showing first 1000 rows.
Click to re-execute with maximum result limits.

Command took 3.89 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 03/05/2022, 02:44:47 on COURSEWORK

Cmd 66

```
#Show(5) displays the highest 5 sponsors
HighestSponsorDF3.show(5)
```

(2) Spark Jobs

Sponsor	count
Assiut University	437
Cairo University	369
M.D. Anderson Can...	219
Assistance Publique...	211
National Cancer I...	158

only showing top 5 rows

Command took 3.51 seconds -- by f.k.aladegbola@edu.salford.ac.uk at 03/05/2022, 02:45:03 on COURSEWORK

7°C Mostly cloudy

