USECASE STUDY PROJECT

Group no: Group 22

Student Names: Kirthana Shri C, Sabeen Mustafa

EXECUTIVE SUMMARY

In the past decade, technological advancements have led to the development of online platforms for all services. COVID-19's crisis has triggered significant changes in the way people, businesses, and governments use digital technologies. A wide range of digital technologies are being used today, and this has led to an increase in electronic access to health care and consultancy services. Therefore, we aim to create a website that can be used to diagnose health problems, develop treatment plans and provide services to patients at home.

The purpose of our project is to provide an interface for doctors, patients, and other staff (from the laboratory and pharmacy) to conduct their respective activities. Paraworld can be used to diagnose and treat patients in remote area. By using paraworld for scheduled follow-up visits, doctors and patients will be more connected, and the likelihood of follow-up will be higher, reducing missed appointments and enhancing patient outcomes.

I. INTRODUCTION

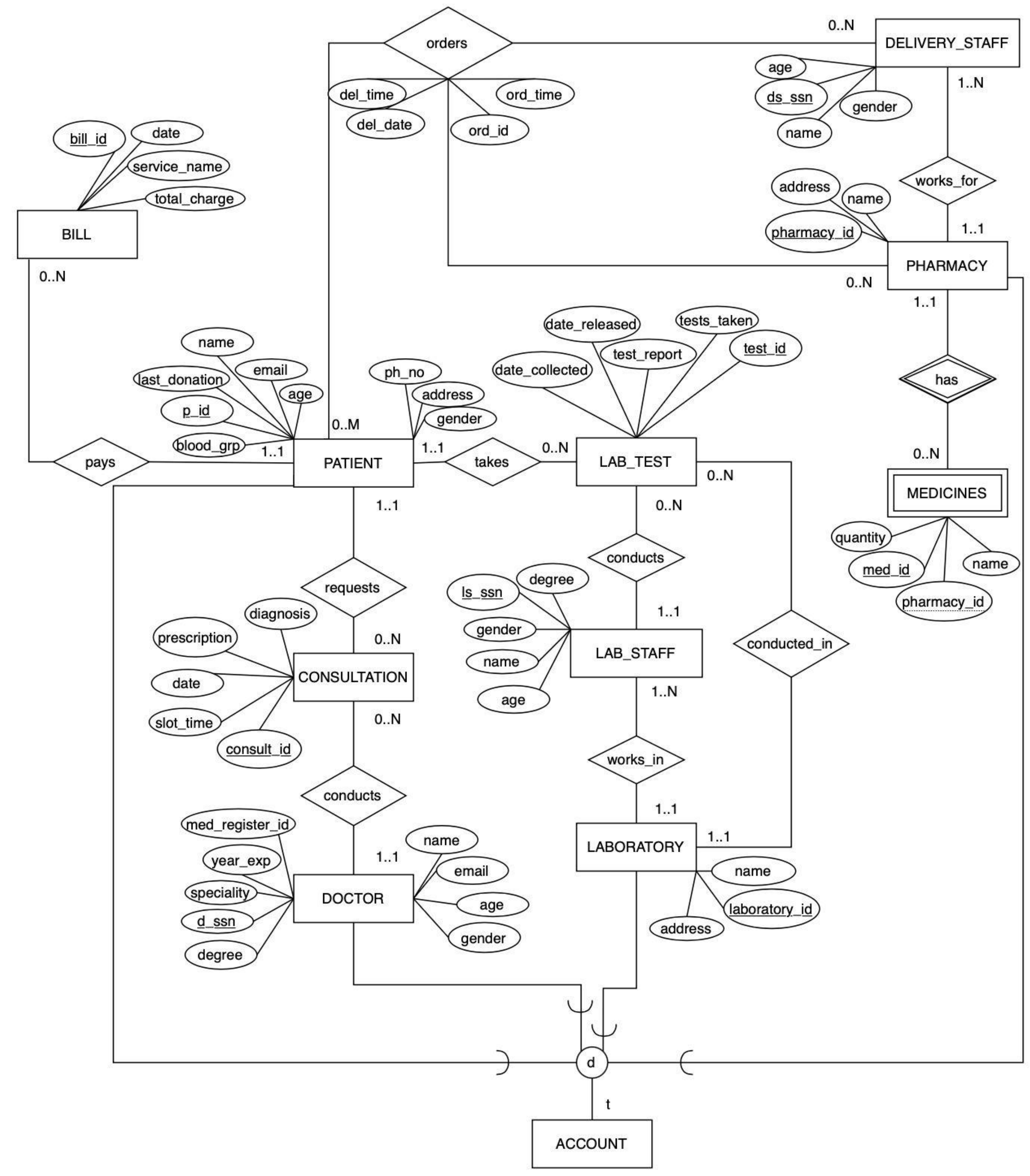
To utilize the features offered, doctors, patients, members of the working team, and organizations such as pharmacies and testing laboratories must register with paraworld. Doctors from various departments provide consultation services to patients through paraworld. They can provide the patients with diagnoses and prescribe them appropriate medication. With the patient's consent, they can also have access to the patient's medical history and records that are stored in the patient's data. Once the patients have reviewed the doctor's schedule, they can make appointments for consultation. If the patient chooses to use the paraworld laboratory home services located near the patient's area then the patient can book an appointment from the available time slots and a staff person from the laboratory center will go to collect the samples at the scheduled time. The patient can also see their billing details in their profile and make online payments for consultation, testing, and medication. Additionally, patients can give feedback on the medical professionals and the paraworld services.

Additionally, paraworld offers pharmacy services, whereby the business finds the closest pharmacy that is registered with paraworld and accepts patient delivery orders. The pharmacy sends the order to the patient at the specified address. Paraworld also provides an additional feature for blood donations that will encourage the users to donate blood after verifying their medical history of the user and if they donate blood then they will be given a free consultation coupon. It is only possible to donate blood once every eight weeks.

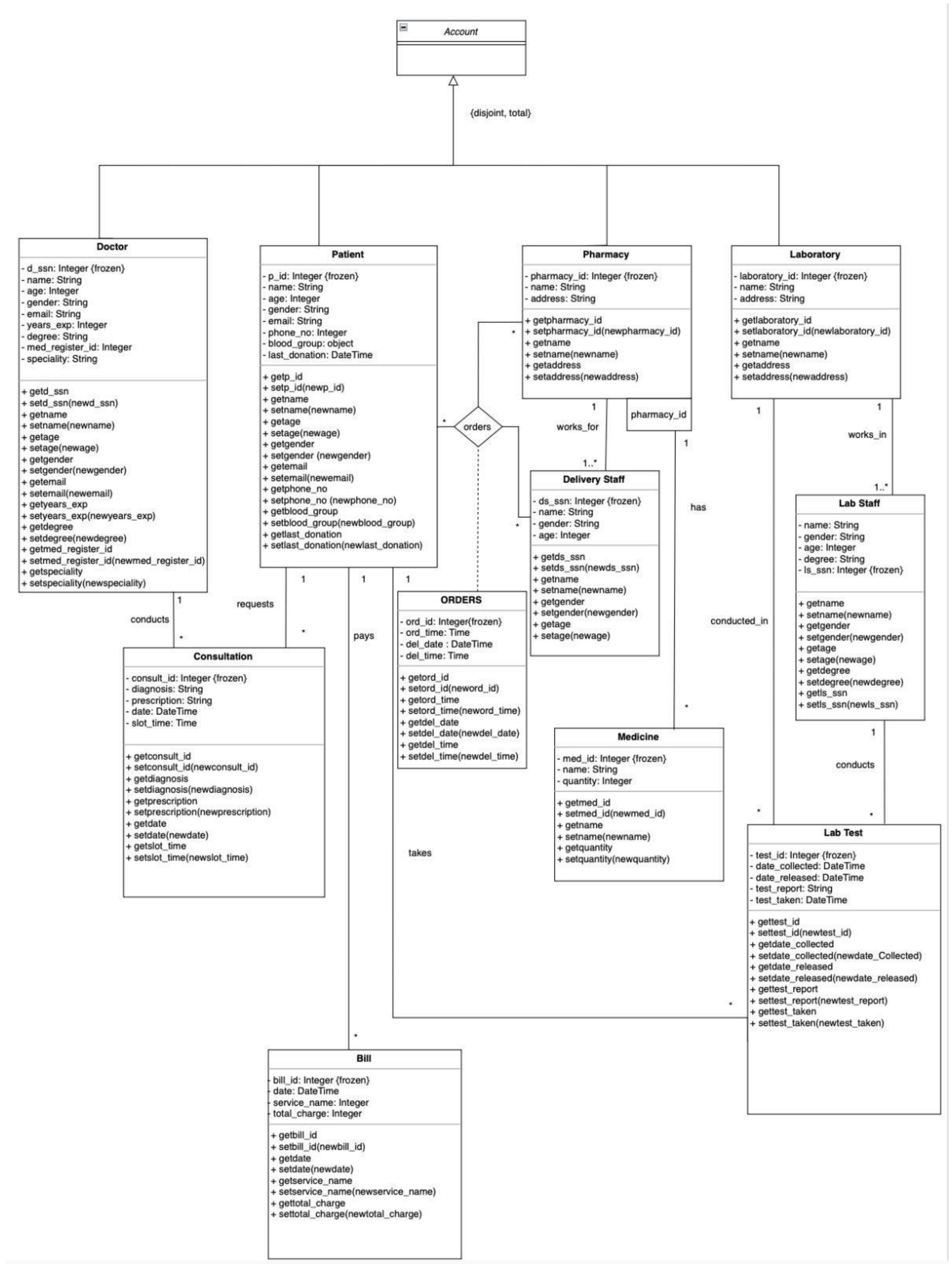
The company will store the data of the users registered in various categories. The company has three divisions of services consultation, laboratory, and pharmacy. The company needs to store the basic information of patients, doctors, and other staff. For each patient, the company needs to store the name, address, insurance details, age, gender, blood group, height, weight, contact number and email address. For doctors, their register ID, name, SSN, address, age, gender, department, degree and consulting hours. For Lab staff, the company stores their name, address, age, gender, and Laboratory name. For medicines delivery agents, the company stores their name, address, age, gender, and order ID.

The process of consultation appointment needs to be recorded. For this process, the company needs to store the doctor's name, patient name, appointment time, appointment date, patient medical history, diagnosis, prescription, and medical tests to be taken. For the process of laboratory services, the company needs to store the laboratory branch name, test name, patient name, date, and appointment time, staff ID, and lab technician ID. For the process of delivery by pharmacy, the company needs to store the branch ID, order ID, patient address, order details, and order date.

II. CONCEPTUAL MODEL



EER DIAGRAM



UML CLASS DIAGRAM

III. RELATIONAL MODELS

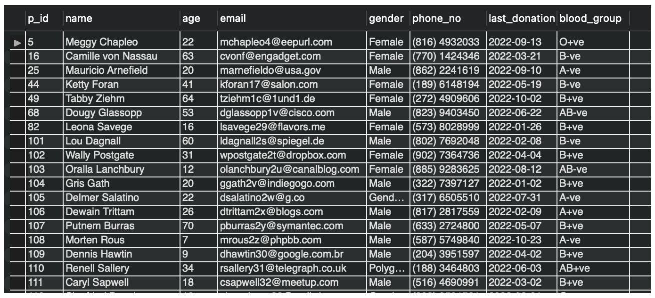
PATIENT(p_id, name, age, gender, email, phone_no, blood_group, last_donation)
p_id – Primary Key
DOCTOR(d_ssn, name, age, gender, email, years_exp, degree, med_register_id, speciality)
d_ssn – Primary Key
CONSULTATION(consult_id, p_id, d_ssn, diagnosis, prescription, date, slot_time)
consult_id – Primary Key
p_id foreign key refers to *p_id* in PATIENT , NOT NULL
d_ssn foreign key refers to *d_ssn* in DOCTOR, NOT NULL
LABORATORY(laboratory_id, name, address)
laboratory_id – Primary Key
LAB_STAFF(ls_ssn, laboratory_id, name, gender, age, degree)
ls_ssn – Primary Key
laboratory_id foreign key refers to *laboratory_id* in LABORATORY , NOT NULL
LAB_TEST(test_id, p_id, ls_ssn, laboratory_id, date_collected, date_released, test_report, test_taken)
test_id – Primary Key
p_id foreign key refers to *p_id* in PATIENT , NOT NULL
laboratory_id foreign key refers to *laboratory_id* in LABORATORY , NOT NULL
ls_ssn foreign key refers to *ls_ssn* in LAB_STAFF , NOT NULL
PHARMACY(pharmacy_id, name, address)
pharmacy_id – Primary Key
MEDICINES(med_id, name, quantity, pharmacy_id)
med_id – Primary Key
pharmacy_id foreign key refers to *pharmacy_id* in PHARMACY , NOT NULL
DELIVERY_STAFF(ds_ssn, name, gender, age, pharmacy_id)
ds_ssn – Primary Key
pharmacy_id foreign key refers to *pharmacy_id* in PHARMACY , NOT NULL
ORDER (p_id, pharmacy_id, ds_ssn, ord_id, date, ord_time, del_time)
ord_id – Primary Key
p_id foreign key refers to *p_id* in PATIENT , NOT NULL
pharmacy_id foreign key refers to *pharmacy_id* in PHARMACY , NOT NULL
ds_ssn foreign key refers to *ds_ssn* in DELIVERY_STAFF , NOT NULL
BILL(bill_id, p_id, date, service_name, total_charge)
bill_id – Primary Key
p_id foreign key refers to *p_id* in PATIENT , NOT NULL

IV. IMPLEMENTATION OF RELATIONAL MODEL VIA MYSQL AND NOSQL

MYSQL IMPLEMENTATION:

-- Q1 Selecting patients who have not used any service

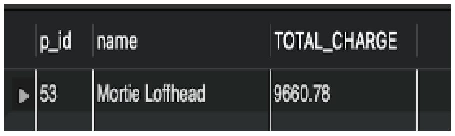
```
SELECT * FROM PATIENT
WHERE P_ID NOT IN (SELECT P.P_ID FROM PATIENT P
                            JOIN BILL B
                            ON P.P_ID = B.P_ID);
```



p_id	name	age	email	gender	phone_no	last_donation	blood_group
5	Meggy Chapleo	22	mchapleo4@eepurl.com	Female	(816) 4932033	2022-09-13	O+ve
16	Camille von Nassau	63	cvonf@engadget.com	Female	(770) 1424346	2022-03-21	B-ve
25	Mauricio Arnefield	20	marnefieldo@usa.gov	Male	(862) 2241619	2022-09-10	A-ve
44	Ketty Foran	41	kforan17@salon.com	Female	(189) 6148194	2022-05-19	B-ve
49	Tabby Ziehm	64	tziehm1c@1und1.de	Female	(272) 4909606	2022-10-02	B+ve
68	Dougy Glassopp	53	dglassopp1v@cisco.com	Male	(823) 9403450	2022-06-22	AB-ve
82	Leona Savege	16	lsavege29@flavors.me	Female	(573) 8028999	2022-01-26	B+ve
101	Lou Dagnall	60	ldagnall2s@spiegel.de	Male	(802) 7692048	2022-02-08	B-ve
102	Wally Postgate	31	wpostgate2t@dropbox.com	Female	(902) 7364736	2022-04-04	B+ve
103	Oralla Lanchbury	12	olanchbury2u@canalblog.com	Female	(885) 9283625	2022-08-12	AB-ve
104	Gris Gath	20	ggath2v@indiegogo.com	Male	(322) 7397127	2022-01-02	B+ve
105	Delmer Salatino	22	dsalatino2w@g.co	Gend...	(317) 6505510	2022-07-31	A-ve
106	Dewain Trittam	26	dtrittam2x@blogs.com	Male	(817) 2817559	2022-02-09	A+ve
107	Putnem Burras	70	pburras2y@symantec.com	Male	(633) 2724800	2022-05-07	B+ve
108	Morten Rous	7	mrous2z@phpbb.com	Male	(587) 5749840	2022-10-23	A-ve
109	Dennis Hawtin	9	dhawtin30@google.com.br	Male	(204) 3951597	2022-04-02	B+ve
110	Renell Sallery	34	rsallery31@telegraph.co.uk	Polyg...	(188) 3464803	2022-06-03	AB+ve
111	Caryl Sapwell	18	csapwell32@meetup.com	Male	(516) 4690991	2022-03-02	B+ve

-- Q2 Retrieve patient detail who paid the highest consultation fee together with the fee amount

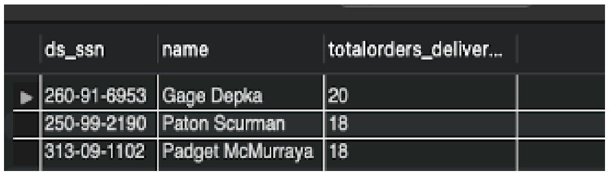
```
SELECT P.p_id, P.name,B.TOTAL_CHARGE
FROM BILL B
JOIN PATIENT P ON
          B.p_id = P.p_id
WHERE B.SERVICE_NAME = 'CONSULTATION' AND
          B.TOTAL_CHARGE >=ALL(SELECT TOTAL_CHARGE
                                FROM BILL
                                WHERE SERVICE_NAME = 'CONSULTATION');
```



p_id	name	TOTAL_CHARGE
53	Mortie Loffhead	9660.78

-- Q3 Retrieving ssn,name,total delivered orders of delivery staff with top 3 most outstanding delivery orders

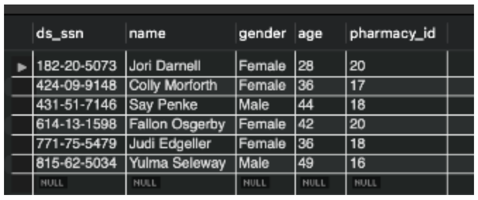
```
SELECT DS.ds_ssn, DS.name, T1.totalorders_delivered
FROM (SELECT ds_ssn, count(*) as totalorders_delivered
          FROM ORDERS
          GROUP BY ds_ssn) T1
JOIN DELIVERY_STAFF DS
          ON DS.ds_ssn = T1.ds_ssn
WHERE 3 > (SELECT COUNT(*)
      FROM (SELECT ds_ssn, count(*) as totalorders_delivered
          FROM ORDERS
          GROUP BY ds_ssn) T2
           WHERE T1.totalorders_delivered < T2.totalorders_delivered)
ORDER BY T1.totalorders_delivered DESC;
```



ds_ssn	name	totalorders_deliver...
260-91-6953	Gage Depka	20
250-99-2190	Paton Scurman	18
313-09-1102	Padget McMurraya	18

-- Q4 Retrieving delivery persons who have not delivered orders to any patients

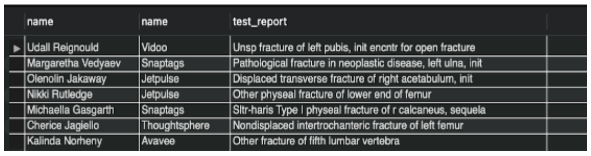
```
SELECT *
FROM DELIVERY_STAFF DS
WHERE NOT EXISTS
          (SELECT *
          FROM PATIENT P
            WHERE EXISTS
                (SELECT *
                 FROM ORDERS O
                  WHERE P.p_id = O.p_id AND
                        DS.ds_ssn = O.ds_ssn));
```



ds_ssn	name	gender	age	pharmacy_id
182-20-5073	Jori Darnell	Female	28	20
424-09-9148	Colly Morforth	Female	36	17
431-51-7146	Say Penke	Male	44	18
614-13-1598	Fallon Osgerby	Female	42	20
771-75-5479	Judi Edgeller	Female	36	18
815-62-5034	Yulma Seleway	Male	49	16
NULL	NULL	NULL	NULL	NULL

-- Q5 Retrieving patient name of patients who has taken test in a laboratory where report contain fracture word, together with laboratory name and test report

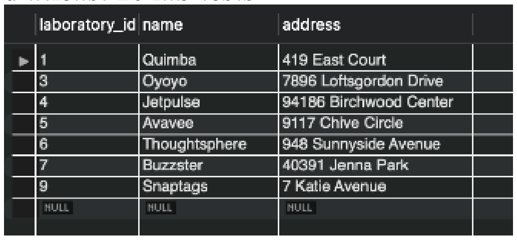
```
SELECT P.name, L.name, LT.test_report
FROM PATIENT P
JOIN LAB_TEST LT
        ON P.p_id = LT.p_id
JOIN LABORATORY L
        ON L.laboratory_id = LT.laboratory_id
WHERE LT.test_report LIKE '%FRACTURE%';
```



name	name	test_report
Udall Reignould	Vidoo	Unsp fracture of left pubis, init encntr for open fracture
Margaretha Vedyaev	Snaptags	Pathological fracture in neoplastic disease, left ulna, init
Olenolin Jakaway	Jetpulse	Displaced transverse fracture of right acetabulum, init
Nikki Rutledge	Jetpulse	Other physeal fracture of lower end of femur
Michaella Gasgarth	Snaptags	Sltr-haris Type I physeal fracture of r calcaneus, sequela
Cherice Jagiello	Thoughtsphere	Nondisplaced intertrochanteric fracture of left femur
Kalinda Norheny	Avavee	Other fracture of fifth lumbar vertebra

-- Q6 Retrieving details of all laboratories that have performed atleast 10 lab tests

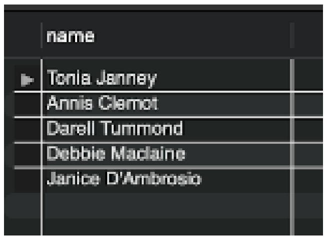
```
SELECT *
FROM LABORATORY L
WHERE 10 < (SELECT count(*)
                FROM LAB_TEST LT
    WHERE L.laboratory_id = LT.laboratory_id);
```



laboratory_id	name	address
1	Quimba	419 East Court
3	Oyoyo	7896 Loftsgordon Drive
4	Jetpulse	94186 Birchwood Center
5	Avavee	9117 Chive Circle
6	Thoughtsphere	948 Sunnyside Avenue
7	Buzzster	40391 Jenna Park
9	Snaptags	7 Katie Avenue
NULL	NULL	NULL

-- Q7 Retrieving name of the doctor who has diagnosed 'M62830' code or has the speciality in Psychiatry using union.

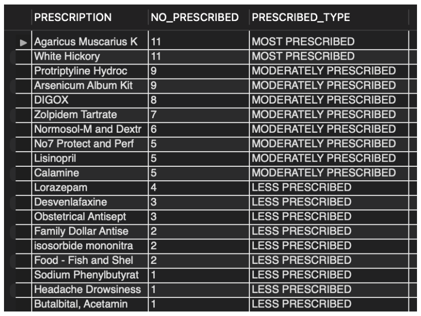
```
SELECT name
FROM DOCTOR
WHERE d_ssn IN (SELECT d_ssn
                                        FROM CONSULTATION
            WHERE diagnosis = 'M62830')

UNION

SELECT name
FROM DOCTOR
WHERE d_ssn IN (SELECT d_ssn
                                        FROM DOCTOR
            WHERE speciality = 'Psychiatry');
```



name
Tonia Janney
Annis Clemot
Darell Tummond
Debbie Maclaine
Janice D'Ambrosio

-- Q8 RETRIEVING EACH TYPE OF PRESCRIPTION AND CATEGORIZE THEM AS MOST PRESCRIBED,MODERATELY PRESCRIBED AND LESS PRESCRIBED

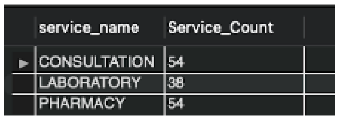
```
SELECT PRESCRIPTION, COUNT(PRESCRIPTION) AS NO_PRESCRIBED,
CASE
          WHEN COUNT(PRESCRIPTION) > 10 THEN 'MOST PRESCRIBED'
          WHEN COUNT(PRESCRIPTION) <=10 AND
COUNT(PRESCRIPTION)>=5 THEN 'MODERATELY PRESCRIBED'
                    ELSE 'LESS PRESCRIBED'
          END AS PRESCRIBED_TYPE
FROM CONSULTATION
GROUP BY PRESCRIPTION
ORDER BY COUNT(PRESCRIPTION) DESC;
```



PRESCRIPTION	NO_PRESCRIBED	PRESCRIBED_TYPE
Agaricus Muscarius K	11	MOST PRESCRIBED
White Hickory	11	MOST PRESCRIBED
Protriptyline Hydroc	9	MODERATELY PRESCRIBED
Arsenicum Album Kit	9	MODERATELY PRESCRIBED
DIGOX	8	MODERATELY PRESCRIBED
Zolpidem Tartrate	7	MODERATELY PRESCRIBED
Normosol-M and Dextr	6	MODERATELY PRESCRIBED
No7 Protect and Perf	5	MODERATELY PRESCRIBED
Lisinopril	5	MODERATELY PRESCRIBED
Calamine	5	MODERATELY PRESCRIBED
Lorazepam	4	LESS PRESCRIBED
Desvenlafaxine	3	LESS PRESCRIBED
Obstetrical Antisept	3	LESS PRESCRIBED
Family Dollar Antise	2	LESS PRESCRIBED
isosorbide mononitra	2	LESS PRESCRIBED
Food - Fish and Shel	2	LESS PRESCRIBED
Sodium Phenylbutyrat	1	LESS PRESCRIBED
Headache Drowsiness	1	LESS PRESCRIBED
Butalbital, Acetamin	1	LESS PRESCRIBED

-- Q9 Retrieving service_name and count of each type that has service charge more than average service charge

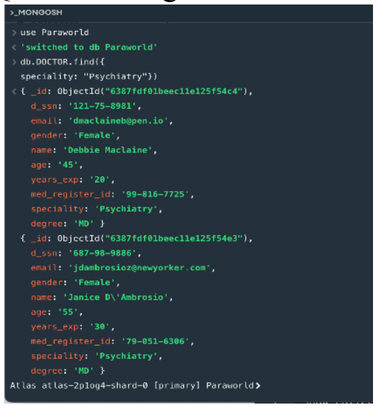
```
SELECT service_name,count(service_name) as Service_Count
FROM BILL
WHERE bill_id = ANY(SELECT bill_id
                                                  FROM BILL
                                                  WHERE total_charge > (SELECT avg(total_charge) FROM
BILL))
GROUP BY service_name;
```



service_name	Service_Count
CONSULTATION	54
LABORATORY	38
PHARMACY	54

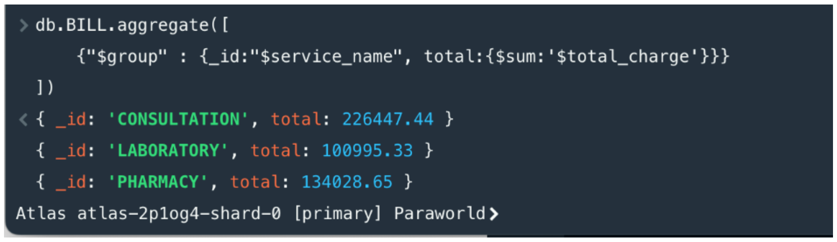
NoSQL IMPLEMENTATION IN MONGODB COMPASS:

Q1. Retrieving doctor data from collection DOCTOR whose speciality is Psychiatry.



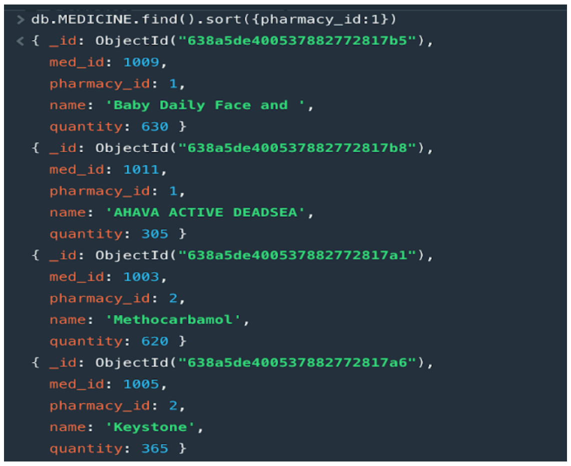
```
>_MONGOSH
> use Paraworld
< 'switched to db Paraworld'
> db.DOCTOR.find({
  speciality: "Psychiatry"})
< { _id: ObjectId("6387fdf01beec11e125f54c4"),
    d_ssn: '121-75-8981',
    email: 'dmaclaineb@pen.io',
    gender: 'Female',
    name: 'Debbie Maclaine',
    age: '45',
    years_exp: '20',
    med_register_id: '99-816-7725',
    speciality: 'Psychiatry',
    degree: 'MD' }
  { _id: ObjectId("6387fdf01beec11e125f54e3"),
    d_ssn: '687-98-9886',
    email: 'jdambrosioz@newyorker.com',
    gender: 'Female',
    name: 'Janice D\'Ambrosio',
    age: '55',
    years_exp: '30',
    med_register_id: '79-051-6306',
    speciality: 'Psychiatry',
    degree: 'MD' }
Atlas atlas-2plog4-shard-0 [primary] Paraworld>
```

Q2. Retrieving data from collection BILL grouping by service name and total service charge.



```
> db.BILL.aggregate([
      {"$group" : {_id:"$service_name", total:{$sum:'$total_charge'}}}
  ])
< { _id: 'CONSULTATION', total: 226447.44 }
  { _id: 'LABORATORY', total: 100995.33 }
  { _id: 'PHARMACY', total: 134028.65 }
Atlas atlas-2p1og4-shard-0 [primary] Paraworld>
```

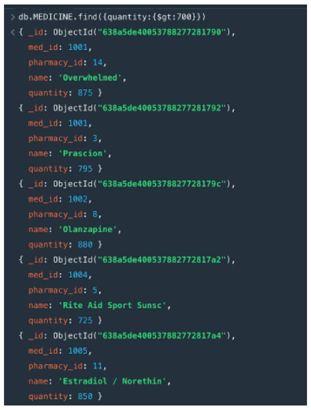
Q3. Retrieve data from collection MEDICINE and sort in ascending order by pharmacy id.



```
> db.MEDICINE.find().sort({pharmacy_id:1})
< { _id: ObjectId("638a5de400537882772817b5"),
    med_id: 1009,
    pharmacy_id: 1,
    name: 'Baby Daily Face and ',
    quantity: 630 }
  { _id: ObjectId("638a5de400537882772817b8"),
    med_id: 1011,
    pharmacy_id: 1,
    name: 'AHAVA ACTIVE DEADSEA',
    quantity: 305 }
  { _id: ObjectId("638a5de400537882772817a1"),
    med_id: 1003,
    pharmacy_id: 2,
    name: 'Methocarbamol',
    quantity: 620 }
  { _id: ObjectId("638a5de400537882772817a6"),
    med_id: 1005,
    pharmacy_id: 2,
    name: 'Keystone',
    quantity: 365 }
```

.

Q4. Retrieving data from collection MEDICINE whose quantity is greater than 700.

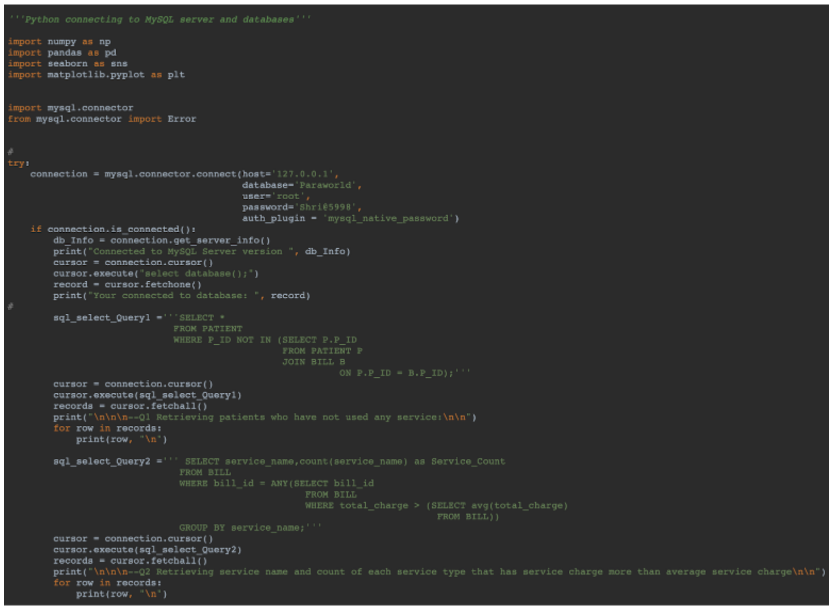


```
> db.MEDICINE.find({quantity:{$gt:700}})
< { _id: ObjectId("638a5de40053788277281790"),
    med_id: 1001,
    pharmacy_id: 14,
    name: 'Overwhelmed',
    quantity: 875 }
  { _id: ObjectId("638a5de40053788277281792"),
    med_id: 1001,
    pharmacy_id: 3,
    name: 'Prascion',
    quantity: 795 }
  { _id: ObjectId("638a5de4005378827728179c"),
    med_id: 1002,
    pharmacy_id: 8,
    name: 'Olanzapine',
    quantity: 880 }
  { _id: ObjectId("638a5de400537882772817a2"),
    med_id: 1004,
    pharmacy_id: 5,
    name: 'Rite Aid Sport Sunsc',
    quantity: 725 }
  { _id: ObjectId("638a5de400537882772817a4"),
    med_id: 1005,
    pharmacy_id: 11,
    name: 'Estradiol / Norethin',
    quantity: 850 }
```

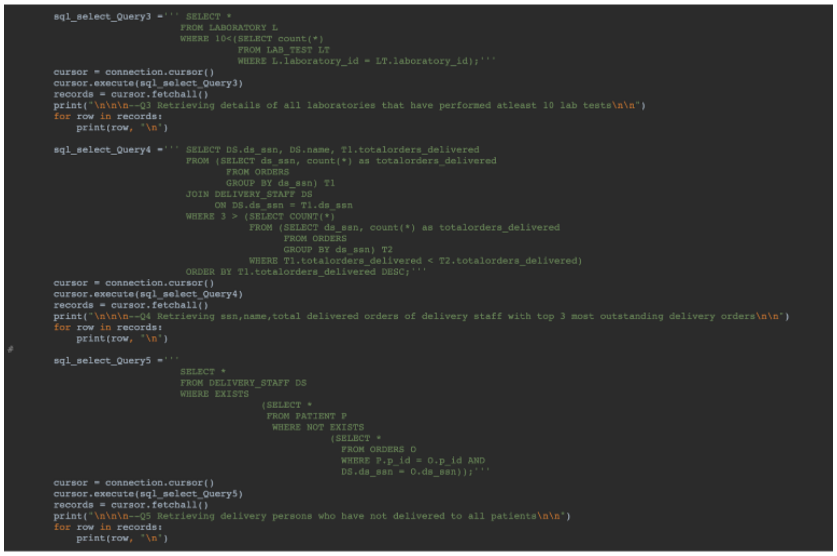
V. DATABASE ACCESS VIA PYTHON

STEP 1: The connection of MySQL to Python is done using mysql.connector, followed by cursor.excecute to run and cursor.fetchall to fetch all tuples from the query.

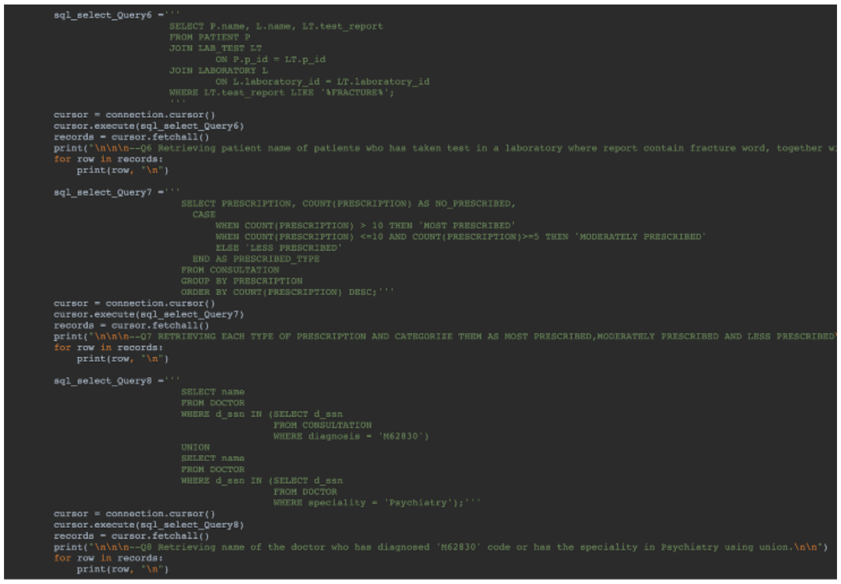
STEP 2: Fetched queries(list of tuples) converted into a dataframe using pandas library and plotted graphs using seaborn and matplotlib for analytics of Paraworld.



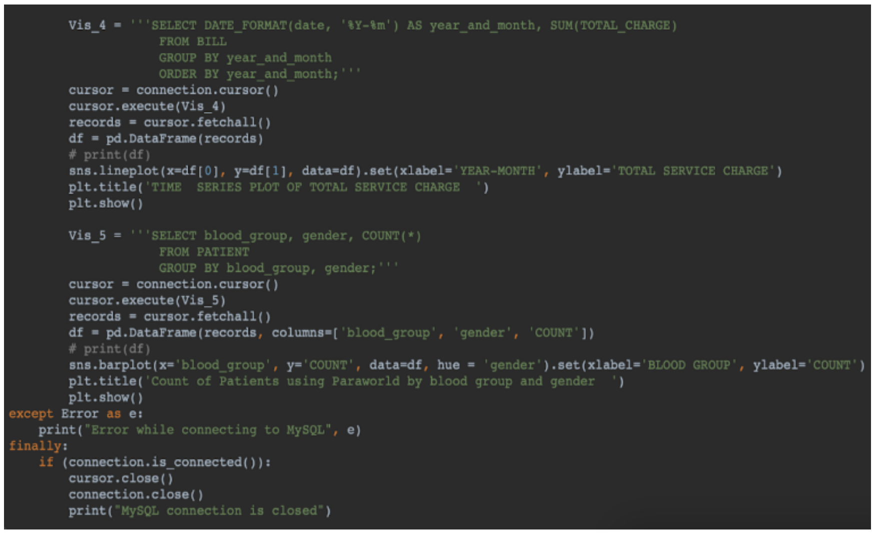
```python
'''Python connecting to MySQL server and databases'''

import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt


import mysql.connector
from mysql.connector import Error


#
try:
    connection = mysql.connector.connect(host='127.0.0.1',
                                         database='Paraworld',
                                         user='root',
                                         password='Shri@5998',
                                         auth_plugin = 'mysql_native_password')
    if connection.is_connected():
        db_Info = connection.get_server_info()
        print("Connected to MySQL Server version ", db_Info)
        cursor = connection.cursor()
        cursor.execute("select database();")
        record = cursor.fetchone()
        print("Your connected to database: ", record)
#
        sql_select_Query1 ='''SELECT *
                         FROM PATIENT
                         WHERE P_ID NOT IN (SELECT P.P_ID
                                            FROM PATIENT P
                                            JOIN BILL B
                                                      ON P.P_ID = B.P_ID);'''
        cursor = connection.cursor()
        cursor.execute(sql_select_Query1)
        records = cursor.fetchall()
        print("\n\n\n--Q1 Retrieving patients who have not used any service:\n\n")
        for row in records:
            print(row, "\n")

        sql_select_Query2 =''' SELECT service_name,count(service_name) as Service_Count
                          FROM BILL
                          WHERE bill_id = ANY(SELECT bill_id
                                              FROM BILL
                                              WHERE total_charge > (SELECT avg(total_charge)
                                                                    FROM BILL))
                          GROUP BY service_name;'''
        cursor = connection.cursor()
        cursor.execute(sql_select_Query2)
        records = cursor.fetchall()
        print("\n\n\n--Q2 Retrieving service name and count of each service type that has service charge more than average service charge\n\n")
        for row in records:
            print(row, "\n")
```



```python
        sql_select_Query3 =''' SELECT *
                          FROM LABORATORY L
                          WHERE 10<(SELECT count(*)
                                    FROM LAB_TEST LT
                                    WHERE L.laboratory_id = LT.laboratory_id);'''
        cursor = connection.cursor()
        cursor.execute(sql_select_Query3)
        records = cursor.fetchall()
        print("\n\n\n--Q3 Retrieving details of all laboratories that have performed atleast 10 lab tests\n\n")
        for row in records:
            print(row, "\n")

        sql_select_Query4 =''' SELECT DS.ds_ssn, DS.name, T1.totalorders_delivered
                           FROM (SELECT ds_ssn, count(*) as totalorders_delivered
                                 FROM ORDERS
                                 GROUP BY ds_ssn) T1
                           JOIN DELIVERY_STAFF DS
                                ON DS.ds_ssn = T1.ds_ssn
                           WHERE 3 > (SELECT COUNT(*)
                                      FROM (SELECT ds_ssn, count(*) as totalorders_delivered
                                            FROM ORDERS
                                            GROUP BY ds_ssn) T2
                                      WHERE T1.totalorders_delivered < T2.totalorders_delivered)
                           ORDER BY T1.totalorders_delivered DESC;'''
        cursor = connection.cursor()
        cursor.execute(sql_select_Query4)
        records = cursor.fetchall()
        print("\n\n\n--Q4 Retrieving ssn,name,total delivered orders of delivery staff with top 3 most outstanding delivery orders\n\n")
        for row in records:
            print(row, "\n")
#
        sql_select_Query5 ='''
                          SELECT *
                          FROM DELIVERY_STAFF DS
                          WHERE EXISTS
                                       (SELECT *
                                        FROM PATIENT P
                                         WHERE NOT EXISTS
                                                    (SELECT *
                                                     FROM ORDERS O
                                                     WHERE P.p_id = O.p_id AND
                                                     DS.ds_ssn = O.ds_ssn));'''
        cursor = connection.cursor()
        cursor.execute(sql_select_Query5)
        records = cursor.fetchall()
        print("\n\n\n--Q5 Retrieving delivery persons who have not delivered to all patients\n\n")
        for row in records:
            print(row, "\n")
```



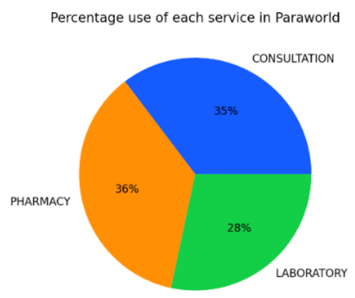
```python
        sql_select_Query6 ='''
                        SELECT P.name, L.name, LT.test_report
                        FROM PATIENT P
                        JOIN LAB_TEST LT
                                ON P.p_id = LT.p_id
                        JOIN LABORATORY L
                                ON L.laboratory_id = LT.laboratory_id
                        WHERE LT.test_report LIKE '%FRACTURE%';
                        '''
        cursor = connection.cursor()
        cursor.execute(sql_select_Query6)
        records = cursor.fetchall()
        print("\n\n\n--Q6 Retrieving patient name of patients who has taken test in a laboratory where report contain fracture word, together w
        for row in records:
            print(row, "\n")

        sql_select_Query7 ='''
                          SELECT PRESCRIPTION, COUNT(PRESCRIPTION) AS NO_PRESCRIBED,
                            CASE
                                WHEN COUNT(PRESCRIPTION) > 10 THEN 'MOST PRESCRIBED'
                                WHEN COUNT(PRESCRIPTION) <=10 AND COUNT(PRESCRIPTION)>=5 THEN 'MODERATELY PRESCRIBED'
                                ELSE 'LESS PRESCRIBED'
                            END AS PRESCRIBED_TYPE
                          FROM CONSULTATION
                          GROUP BY PRESCRIPTION
                          ORDER BY COUNT(PRESCRIPTION) DESC;'''
        cursor = connection.cursor()
        cursor.execute(sql_select_Query7)
        records = cursor.fetchall()
        print("\n\n\n--Q7 RETRIEVING EACH TYPE OF PRESCRIPTION AND CATEGORIZE THEM AS MOST PRESCRIBED,MODERATELY PRESCRIBED AND LESS PRESCRIBED
        for row in records:
            print(row, "\n")

        sql_select_Query8 ='''
                          SELECT name
                          FROM DOCTOR
                          WHERE d_ssn IN (SELECT d_ssn
                                          FROM CONSULTATION
                                          WHERE diagnosis = 'M62830')
                          UNION
                          SELECT name
                          FROM DOCTOR
                          WHERE d_ssn IN (SELECT d_ssn
                                          FROM DOCTOR
                                          WHERE speciality = 'Psychiatry');'''
        cursor = connection.cursor()
        cursor.execute(sql_select_Query8)
        records = cursor.fetchall()
        print("\n\n\n--Q8 Retrieving name of the doctor who has diagnosed 'M62830' code or has the speciality in Psychiatry using union.\n\n")
        for row in records:
            print(row, "\n")
```



```
        Vis_4 = '''SELECT DATE_FORMAT(date, '%Y-%m') AS year_and_month, SUM(TOTAL_CHARGE)
                    FROM BILL
                    GROUP BY year_and_month
                    ORDER BY year_and_month;'''
        cursor = connection.cursor()
        cursor.execute(Vis_4)
        records = cursor.fetchall()
        df = pd.DataFrame(records)
        # print(df)
        sns.lineplot(x=df[0], y=df[1], data=df).set(xlabel='YEAR-MONTH', ylabel='TOTAL SERVICE CHARGE')
        plt.title('TIME  SERIES PLOT OF TOTAL SERVICE CHARGE  ')
        plt.show()

        Vis_5 = '''SELECT blood_group, gender, COUNT(*)
                    FROM PATIENT
                    GROUP BY blood_group, gender;'''
        cursor = connection.cursor()
        cursor.execute(Vis_5)
        records = cursor.fetchall()
        df = pd.DataFrame(records, columns=['blood_group', 'gender', 'COUNT'])
        # print(df)
        sns.barplot(x='blood_group', y='COUNT', data=df, hue = 'gender').set(xlabel='BLOOD GROUP', ylabel='COUNT')
        plt.title('Count of Patients using Paraworld by blood group and gender  ')
        plt.show()
except Error as e:
    print("Error while connecting to MySQL", e)
finally:
    if (connection.is_connected()):
        cursor.close()
        connection.close()
        print("MySQL connection is closed")
```

VISUALIZATIONS OF PARAWORLD DATA

the

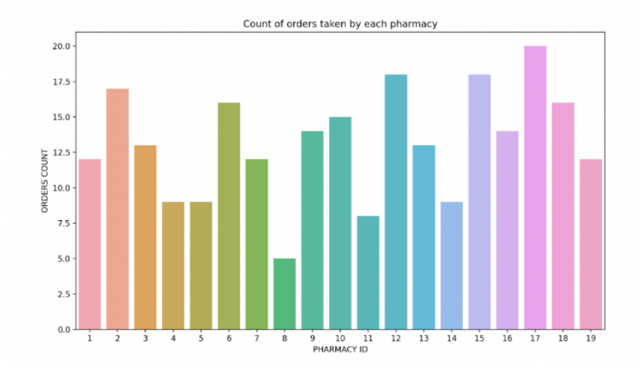


a

PLOT 1

The above pie chart helps Paraworld to determine proportion of each service used.

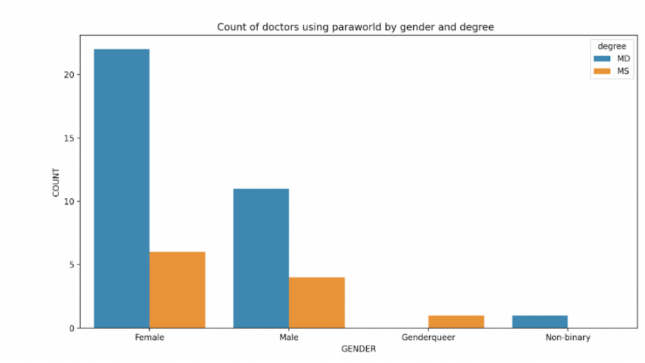
(INFERENCE: Pharmacy and Consultation services are largely used when compared to laboratory service with difference of 7-8%)



PLOT 2

The above bar plot assists Paraworld in understanding the active participation of pharmacies in delivering orders.

(INFERENCE: Pharmacy ID 17 has taken the highest orders with 20 orders, whereas Pharmacy ID 8 has taken the lowest orders with 5 orders.)

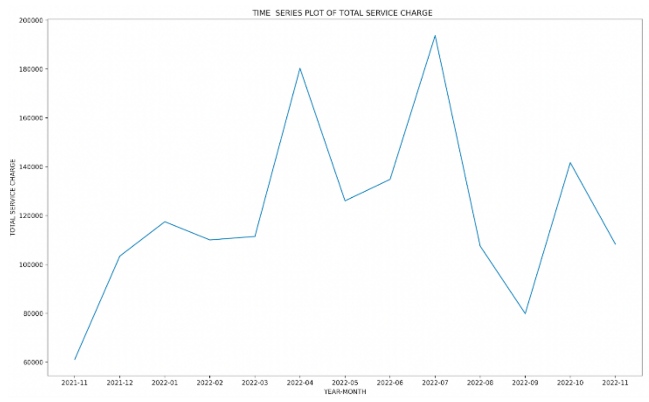


PLOT 3

The above bar graph helps Paraworld understand the involvement of doctors of different degrees across genders.

(INFERENCES: More than 20 female doctors with MD degree actively practice medicine in Paraworld and about 7 with MS degree actively practice   medicine in Paraworld. However, only 11 male doctors with MD degree actively practice medicine in Paraworld and 4 with MS degree actively practice medicine in Paraworld .

Only 2 doctors from other genders practice medicine in Paraworld.)

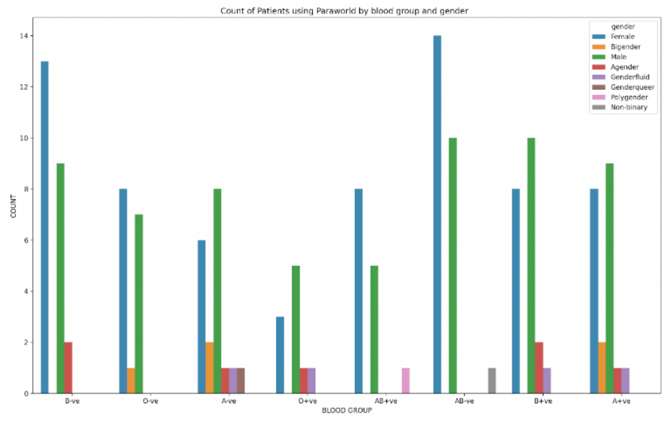


PLOT 4

The above line graph shows the trend of total service charge over time.

(INFERENCE: The peak of Paraworld's total service charges occurred in April and July of the year 2022.

The nadir of total service charge occurred in the month of September of the year 2022.)



PLOT 5

The above bar plot helps to compare the number of patients from different genders across 8 blood groups.

(INFERENCE: Patients with AB-ve use Paraworld more than others, however A+ve and A-ve blood groups have greater gender participation. Fewer patients are from the O+ve blood group.

VII. SUMMARY AND RECOMMENDATION

Paraworld is designed using a MySQL database. It can be used by patients, doctors and various other staff to access services by sitting at their homes. It will result in easier access, for patients who don't have access to transportation.

Improvements can be made in the schema by connecting doctors to the orders directly so as to make it easier for verification and directly providing prescriptions.

The shortcoming of the project is for NoSQL databases. A few more tables should be added for running Neo4J for getting better results.