Introduction

The aim of the project is to help enhance the accuracy and efficiency of a pharmaceutical store. A database of available medicines is created for both the customers and Pharmacists to access. Customers can look up for information on the required medicines like the name, diseases it can cure and prices. This system will help Pharmacists to effectively manage the pharmaceutical store and will also have access to the prescriptions given to the customers by the doctors.

Requirements

We started the project by researching on the requirements of the project based on the Pharmacy Management System flow:

1. Customer:
   When a customer arrives at a pharmacy, we identify them based on the customer Id provided to them, if they are new customer they are asked for their full name, date of birth, phone number, gender, and email.
2. Pharmacist:
   Each pharmacist has been granted a unique Emp Id. Here we have a pharmacist working for us. Pharmacists information like their first name, last name and phone numbers are considered here.
3. Prescription:
   Only prescriptions are accepted for most of the medications in the pharmacy. A prescription includes the customer_id , the prescribing doctor's ID and the date the prescription was written. A number of prescribed medicines are listed on each prescription, along with their name, dosage, and number of refills allowed. A pharmacy is prohibited by law from selling more medication than what is prescribed or anything that is not on the prescription.
4. Medicine:
   Medication information before purchasing or ordering it is required. We need information about the medication, including medicine name, manufacturing company, price, and expiration date. Stock amount also needs to be taken into consideration when ordering medications.
5. Doctor:
   For any prescription, basic details of the doctor such as Doctor Id and doctor name should be included.
6. Order:
   The prescription serves as the basis for an order. Because the consumer might purchase less medication than the prescription calls for. Each order has a distinct Order ID that the system automatically assigns. Multiple medications, each with its own amount, order date and price, may be included in a single order.

Non-Trivial Questions

We created a set of non-trivial questions for our database keeping certain use cases in mind. These questions helped us understand our requirement and flow of the project. Non-trivial question should be focused, specific, relevant, and feasible Below are the set of non-trivial questions.

1. Will the pharmacists know how much stock is available for a particular medicine in the store and be notified when they run out of a particular type of medicine?
2. Can customers check if a drug is in stock at the specific pharmacy?
3. What type of medicines does not require prescription?
4. Does the customer, doctor, and the pharmacists have access to the history of every prescription drug that was supplied to the patient?
5. Is the doctors' prescription available to the pharmacists and how?
6. Will the dosage and time of medical administration be included in the medication's details?
7. Do the details of medicine contain the manufacturing date and expiration date?
8. Will the manufacturer name be included in the medication details
9. Are vitamins categorized into prescribed medicines?
10. Does a patient require a follow-up prescription for additional sets of medications?
11. What contents of the database, that is details of the medicines, is visible to the customer?
12. What are the age restrictions for the medicines?
13. Can pills be identified by the color, shape and pill identification number?
14. How can syrups be identified?
15. Will the prescription have a detail of the doctor and hospital name?
16. Will this medicine work safely with my other medications, including other prescription medicines, over-the-counter medicines, and other supplements?
17. Will the price of the medication available?
18. What are the side effects of the medication?
19. Are there any alternatives for this medication?
20. What medical condition does this medicine treat?

ER Diagram

An Entity Relationship (ER) model is a type of flowchart that explains how entities relate to each other and what are the relationships between them. Some use cases of ER Diagram are:

- Database design: An ER diagram is frequently the first stage in determining and documenting the needs for an information systems project. It further aids to model a specific database or databases in terms of certain technologies required to be implemented as well as the business and logic rules.

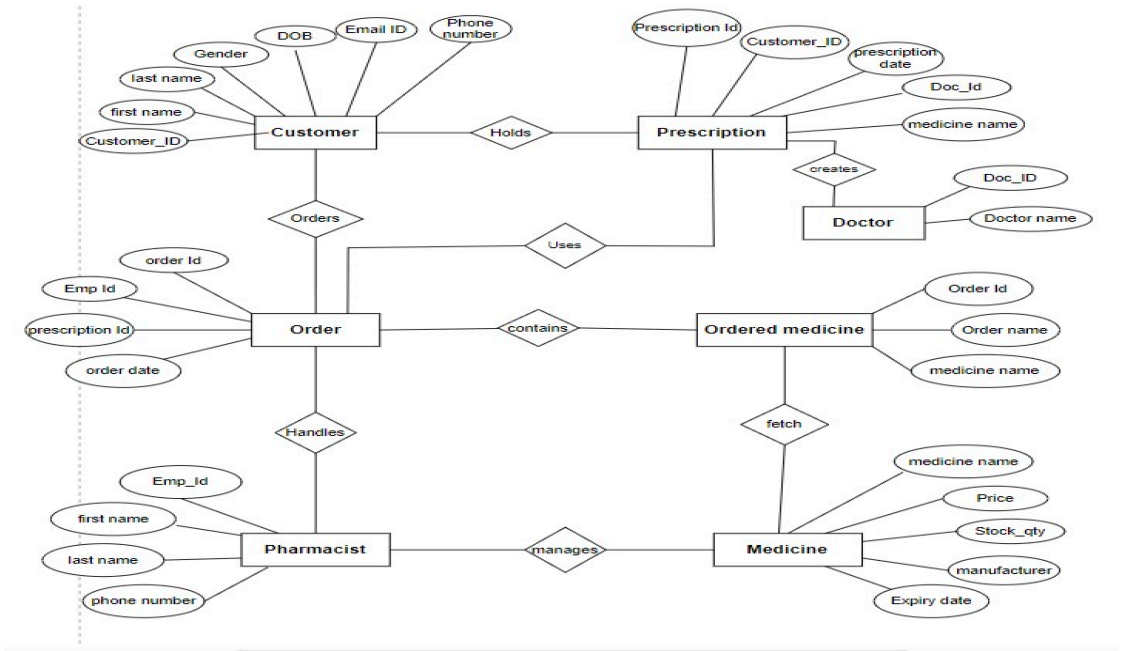
- Database troubleshooting: An ER diagram helps in analyzing and identifying discrepancies in the logic and deployment issues from the existing databases.

- Business Information systems: Relational databases used in business processes can be designed or analyzed using the diagrams. Systems can be streamlined, information can be found more readily, and outcomes can be improved.

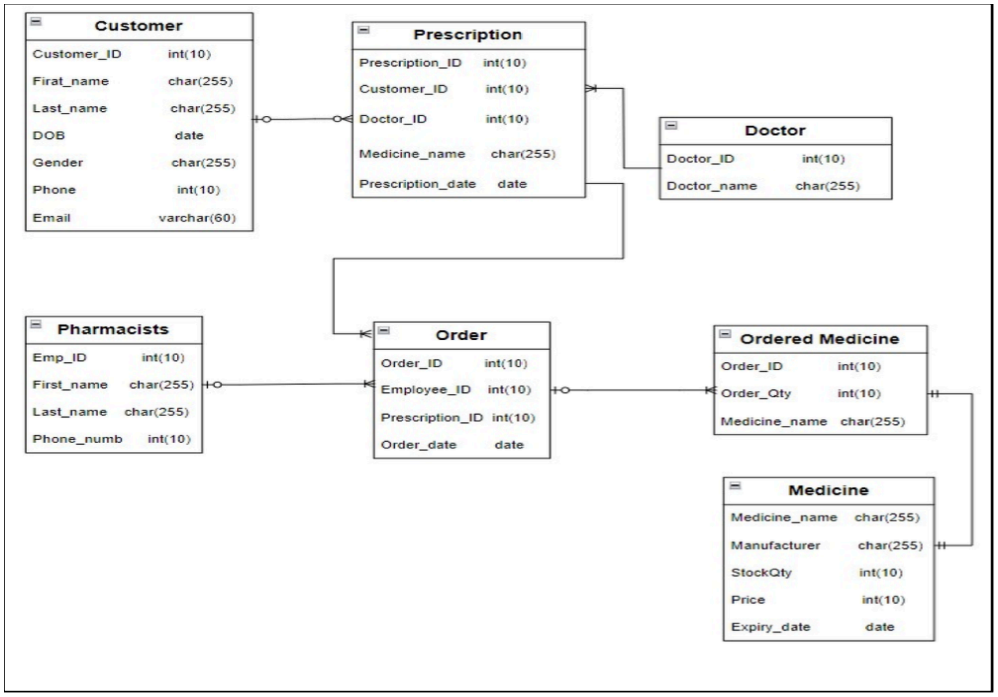
- Business process re-engineering (BPR)**:** ER diagrams are useful for both assessing databases used in new database modeling and business process re-engineering.

- Education: Nowadays, relational data are stored in databases for later retrieval and instructional purposes, therefore ER Diagrams might be helpful in designing those data structures.

ER Diagram:



UML Diagram:



Explanation:

1. A customer can have many prescriptions. Hence this relation is one to many relations.

2. A prescription can have multiple medicines, so the relation is one to many.

3. A prescription can also be a refill; therefore, it can have multiple orders. Hence this relation is also one to many.

4. A customer can have multiple order, but every order is related to just one customer. Hence, this relation is one to many relations.

5. In the medicine table, medicine table can uniquely identify every drug we have.

6. One pharmacist can oversee many orders however one order can only be prepared by one pharmacist. Hence this is a one-to-many relation.

Twitter Scrapping

We used python code to scrape some data for our project from twitter. Below are the steps :

1. Imported all packages required.

```
import tweepy
import csv
import pandas as pd
import os
import wget
import logging
import csv,re
import logging
```

```
import pip
package='tweepy'
pip.main(['install',package])
```

2. Authentication process:

```
consumer_key = "4BqfjRExYD6QjZmSTSOndj0jj"
consumer_secret = "K8AVmfTsOAbNhNuB6gFxDInvHIamxpv5EAlyYsGfDnA9ddLrfb"
access_key = "1591153736260489216-yqUDQLaIPK2d4qH4AGnLi3MEwrMpNt"
access_secret = "wISJSX3CdKRC99Q76vpJ1rxLloVExlg0MORynMCnqs0Dy"
```

```
tweets = pd.DataFrame(columns=["id","created_at","text","media_url","location"])
```

```
tweets.columns
```

```
Index(['id', 'created_at', 'text', 'media_url', 'location'], dtype='object')
```

```
import datetime, time
last_week = datetime.date.today() - datetime.timedelta(9)
since_tweets = datetime.datetime.strptime(time.strftime("%Y-%m-%d"), "%Y-%m-%d")
print (since_tweets)
```

```
auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_key, access_secret)
api = tweepy.API(auth,wait_on_rate_limit=True)

try:
    api.verify_credentials()
    print("Authentication OK")
except:
    print("Error during authentication")
```

```
Authentication OK
```

3. Creating DataFrame:

```
In [20]: import pandas as pd
         tweets_df = pd.DataFrame(tweets_data,columns = ["id","created_at","text","location"])
         tweets_df
```

Out[20]:

	id	created_at	text	location

```
In [ ]:
```

```
In [21]: outfile=re.sub(r"\s+", '_', new_search)
         outfile=outfile+'.csv'
         print(outfile)
         tweets_df.to_csv(outfile, sep=',', encoding='utf-8')
```

---

4. Scrapping twitter using key:

```
import datetime, time
now = datetime.date.today()
date_since = datetime.datetime.strptime(time.strftime("%Y-%m-%d"), "%Y-%m-%d")
print (date_since)
keywords=['Pharmacy', 'medicine']
print (' '.join(keywords))
num_tweets=50
print ("num_tweets ", num_tweets)
```

```
2022-11-28 00:00:00
Pharmacy medicine
num_tweets  50
```

```
new_search = "medicines -filter:retweets"
import tweepy
```

```
tweets = tweepy.Cursor(api.search_tweets,
              q=new_search,
              lang="en",
              until=date_since).items(num_tweets)
```

```
cnt=0
tweets_data = [] #initialize master list to hold our ready tweets
for tweet in tweets:
    print(cnt)
    print(tweet)
    cnt=cnt+1
```

```
0
Status(_api=<tweepy.api.API object at 0x000002C9F9C191C0>, _json={'created_at': 'Sun Nov 27 14:57:17 +0000 2022', 'id': 15968
80849139826688, 'id_str': '1596880849139826688', 'text': '@upasanakonidela @AlwaysRamCharan Madam recently I saw your apol in
my  life. I observed that customer contacted @… https://t.co/6m21SGZrNl', 'truncated': True, 'entities': {'hashtags': [], 'sy
mbols': [], 'user_mentions': [{'screen_name': 'upasanakonidela', 'name': 'Upasana Konidela', 'id': 747433748128473088, 'id_st
r': '747433748128473088', 'indices': [0, 16]}, {'screen_name': 'AlwaysRamCharan', 'name': 'Ram Charan', 'id': 124126166627835
9040, 'id_str': '1241261666278359040', 'indices': [17, 33]}], 'urls': [{'url': 'https://t.co/6m21SGZrNl', 'expanded_url': 'ht
tps://twitter.com/i/web/status/1596880849139826688', 'display_url': 'twitter.com/i/web/status/1…', 'indices': [116, 139]}]},
'metadata': {'iso_language_code': 'en', 'result_type': 'recent'}, 'source': '<a href="http://twitter.com/download/android" re
l="nofollow">Twitter for Android</a>', 'in_reply_to_status_id': 1588157561974403072, 'in_reply_to_status_id_str': '1588157561
974403072', 'in_reply_to_user_id': 747433748128473088, 'in_reply_to_user_id_str': '747433748128473088', 'in_reply_to_screen_n
ame': 'upasanakonidela', 'user': {'id': 888221462456180736, 'id_str': '888221462456180736', 'name': 'Sai PrasannBhattaram',
'screen_name': 'Bhattaramsai', 'location': 'Hyderabad, India', 'description': 'I am a tax consultant in Hyderabad', 'url': No
ne, 'entities': {'description': {'urls': []}}, 'protected': False, 'followers_count': 4, 'friends_count': 16, 'listed_count':
0, 'created_at': 'Fri Jul 21 02:17:53 +0000 2017', 'favourites_count': 3, 'utc_offset': None, 'time_zone': None, 'geo_enable
d': False, 'verified': False, 'statuses_count': 63, 'lang': None, 'contributors_enabled': False, 'is_translator': False, 'is_
translation_enabled': False, 'profile_background_color': 'F5F8FA', 'profile_background_image_url': None, 'profile_background_
image_url_https': None, 'profile_background_tile': False, 'profile_image_url': 'http://pbs.twimg.com/profile_images/158726940
4404101120/dddpRaJG_normal.jpg', 'profile_image_url_https': 'https://pbs.twimg.com/profile_images/1587269404404101120/dddpRaJ
```

```
tweets = tweepy.Cursor(api.search_tweets,
              q=new_search,
              lang="en",
              until=date_since).items(num_tweets)
```

In [27]:
```
tweets
```

Out[27]:
```
<tweepy.cursor.ItemIterator at 0x2c9fc5aca60>
```

In [28]:
```
cnt=0
tweets_data = [] #initialize master list to hold our ready tweets
for tweet in tweets:
    print(cnt)
    tweets_data.append([tweet.id_str,tweet.created_at,tweet.text.encode("utf-8"),tweet.user.location])
    cnt=cnt+1
```

```
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
```

17

In [29]:
```
tweets_data
```

```
  b"I've just been hit by reality, on how funny and pitiful of me to desperately hold on to medicines now, when I used\xe2\x80\xa6 https://t.co/I2G5KycyqO",
   'San Jose del Monte, Bulacan'],
 ['1596343834279088128',
  datetime.datetime(2022, 11, 26, 3, 23, 23, tzinfo=datetime.timezone.utc),
  b'The hospital in Tigray, Mekele, is\xc2\xa0empty of necessary medicines. The medicine shortage may even be worse now than b\xe2\x80\xa6 https://t.co/t5JWUY5k92',
   ''],
 ['1596292368550051840',
  datetime.datetime(2022, 11, 25, 23, 58, 52, tzinfo=datetime.timezone.utc),
  b'The hospital inTigray,Mekele, is\xc2\xa0empty of necessary medicines. The medicine shortage may even be worse nowthan beforo\xe2\x80\xa6 https://t.co/yPCe9cUHuL',
   ''],
 ['1596272674367578114',
  datetime.datetime(2022, 11, 25, 22, 40, 37, tzinfo=datetime.timezone.utc),
  b'Today\xe2\x80\x99s Greatest Alternative Medicines- Health Sciences Institute\n2010 127pg. Softcover\nRegular Price: $3.00\n60% Of\xe2\x80\xa6 https://t.co/YIxtylzfM7',
   'Canada'],
 ['1596271927928446977',
```

In [30]:
```
tweets_df = pd.DataFrame(tweets_data,columns = ["ID","Created_at","Text","Location"])
tweets_df

tweets_df.to_csv(r'Videos\medicines.csv', index = False)
```

In [32]:
```
tweets_df.to_csv(r'Videos\twitter_database.csv', index = False)
```

In [1]:
```
James    John    Robert   Michael  William
David    Richard  Charles  Joseph   Thomas
Christopher  Daniel   Paul     Mark     Donald
George   Kenneth  Steven   Edward   Brian
Ronald   Anthony  Kevin    Jason    Matthew
Gary     Timothy  Jose     Larry    Jeffrey
```

In [34]:
```
outfile=re.sub(r"\s+", '_', new_search)
outfile=outfile+'.csv'
print(outfile)
tweets_df.to_csv(outfile, sep=',', encoding='utf-8')
```

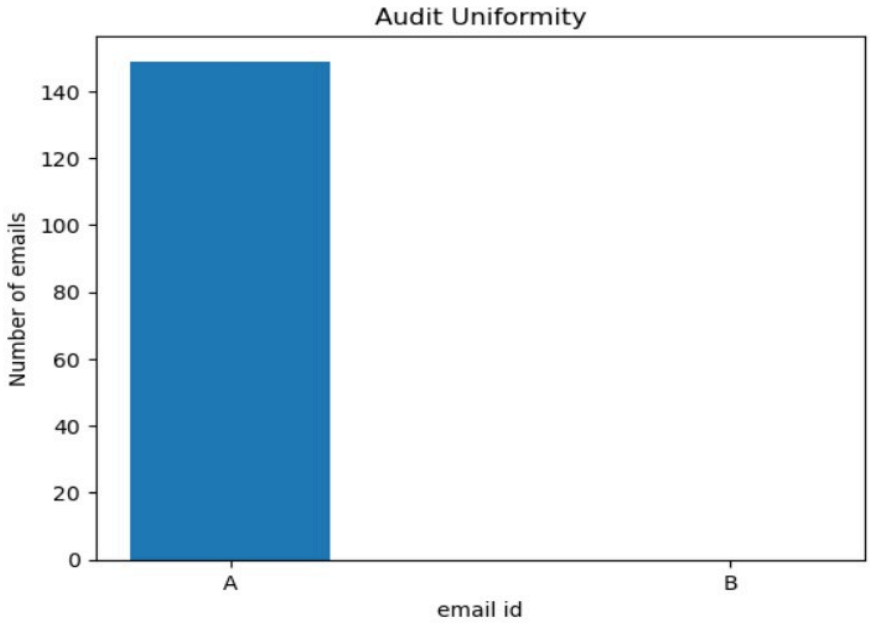
Data Audit:

Audit consistency/uniformity

Here in this we have validated the uniformity of the data using two test cases for customer database.

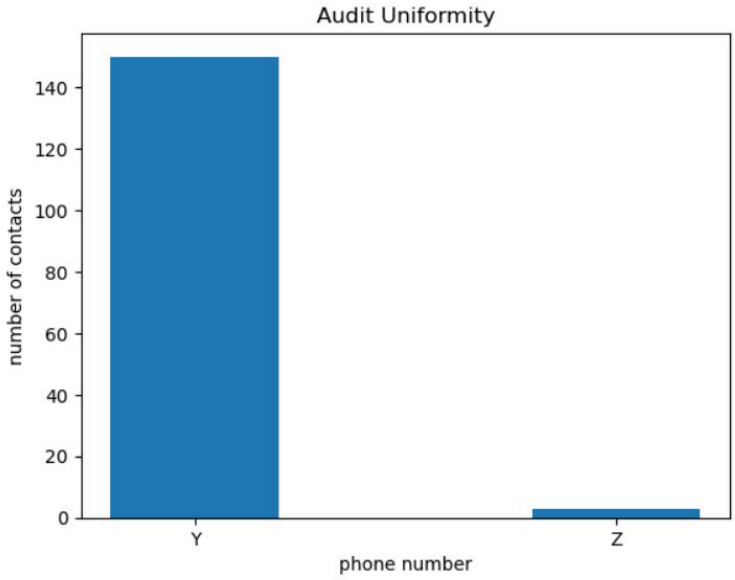
1) Where we validated the email id of the customers with '@' symbol and without '@' symbol and represented them in a bar graph.
   We have a python code for the same

```
In [56]: ##main email audit uniformity check
import pandas as pd
import matplotlib.pyplot as mp

# read csv
df = pd.read_csv("Desktop/dmdd/150-contacts.csv")
df.email



A= x[(x.str.contains("@") == True)]
A
B= y[(y.str.contains("@") == False)]
B

c = ["A","B"]
h= [149,0]
plt.bar(c,h,0.4)
plt.xlabel("email id")
plt.ylabel("Number of emails")
plt.title("Audit Uniformity")
plt.show()
```



We can see that there are no email id without @ symbol in the customer data.

2) Where we validated the phone number of the customers with less than 10 digits and the numbers which is equal to 10 digits and represented them in a bar graph.
We have a python code for the same



Here we can see that out of 149 customer data we have 3 phone numbers that are less that 10 digits ,which does not satisfy the criteria.

In [18]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as mp

df = pd.read_csv('Desktop/dmdd/150-contacts.csv')
df.phone
x= df.phone
x

Y=x.astype(str).apply(lambda x: np.where((len(x)<10)))
Y

Z=x.astype(str).apply(lambda x: np.where((len(x)==10)))
Z

l=["Y","Z"]
m=[150,3]
mp.bar(l,m,0.4)
mp.xlabel("phone number")
mp.ylabel("number of contacts")
mp.title("Audit Uniformity")
mp.show()
```

Audit Accuracy

We have the code and the result for audit accuracy here.

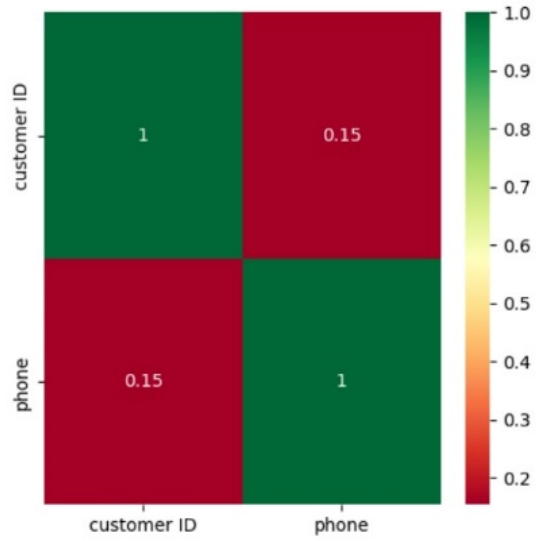
In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
```

In [10]:
```
df= pd.read_csv('Desktop/dmdd/150-contacts.csv')
df.head()
df.info()
x=df.drop('customer ID',axis='columns')
y=df['customer ID']
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 7 columns):
 #   Column       Non-Null Count  Dtype 
---  ------       --------------  ----- 
 0   customer ID  150 non-null    int64 
 1   first_name   150 non-null    object
 2   last_name    150 non-null    object
 3   DOB          150 non-null    object
 4   Gender       150 non-null    object
 5   phone        150 non-null    int64 
 6   email        150 non-null    object
dtypes: int64(2), object(5)
memory usage: 8.3+ KB
```

In [11]:
```
import seaborn as sns
#get correlations of each features in dataset
corrmat = df.corr()
top_corr_features = corrmat.index
plt.figure(figsize=(20,20))
#plot heat map
g=sns.heatmap(df[top_corr_features].corr(),annot=True,cmap="RdYlGn")
```

In [13]:
```
import seaborn as sns
#get correlations of each features in dataset
corrmat = df.corr()
top_corr_features = corrmat.index
plt.figure(figsize=(5,5))
#plot heat map
g=sns.heatmap(df[top_corr_features].corr(),annot=True,cmap="RdYlGn")
```



We also have a heatmap for the same.

Creation of Database:

A. Create a Database:

1. We connect to MySQL Workbench and MySQL Server using terminal using the following Statement :

   mysql.server.start

```
meghanabs@Meghanas-MacBook-Pro ~ % mysql.server start
Starting MySQL
. SUCCESS!
meghanabs@Meghanas-MacBook-Pro ~ %
```

2. Next, we have to create a schema that we will work on.
   CREATE SCHEMA schemaname.

```
1   CREATE SCHEMA `Pharmacy_Management` ;
2
```

B. Create Tables for Database :

To create a database we use SQL Create Statements. We have created 7 tables for this projects : Customer, Doctor, ordered_medicines, pharmacists, prescriptions, Orders, medicines.

Syntax :

```
CREATE TABLE [table_name] (
    column1 datatype,
    column2 datatype,
    column3 datatype, ..
);
```

Now, creating tables for Pharmacy Management System.:

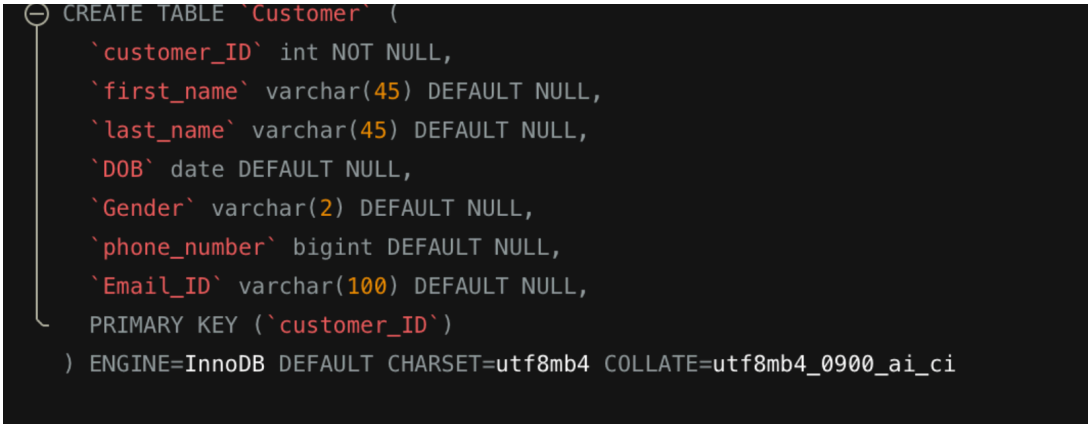
1. Customer Table:

   Columns → customer_ID, first_name, last_name, DOB, Gender, phone_number,email_ID

   Primary Key→ customer_ID

SQL Statement:

```
CREATE TABLE `Customer` (
  `customer_ID` int NOT NULL,
  `first_name` varchar(45) DEFAULT NULL,
  `last_name` varchar(45) DEFAULT NULL,
  `DOB` date DEFAULT NULL,
  `Gender` varchar(2) DEFAULT NULL,
  `phone_number` bigint DEFAULT NULL,
  `Email_ID` varchar(100) DEFAULT NULL,
  PRIMARY KEY (`customer_ID`)
```



2. Doctor Table:

   Columns → Doctor_ID, Doc_Name

   Primary Key→Doctor_ID

   SQL Statement:

```
CREATE TABLE `Doctor` (
  `Doctor_ID` int NOT NULL,
  `Doc_Name` varchar(60) DEFAULT NULL,
  PRIMARY KEY (`Doctor_ID`)
```

```
CREATE TABLE `Doctor` (
  `Doctor_ID` int NOT NULL,
  `Doc_Name` varchar(60) DEFAULT NULL,
  PRIMARY KEY (`Doctor_ID`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE
```

3. Medicine:

   Columns → Medicine_Name, Manufactureres, Price, Stock_Qty,Expiry_Date

   Primary Key→Medicine_Name

   SQL Statement:

   CREATE TABLE `Medicine` (
   `Medicine_Name` varchar(100) NOT NULL,
   `Manufactureres` varchar(100) DEFAULT NULL,
   `Price` double DEFAULT NULL,
   `Stock_Qty` int DEFAULT NULL,
   `Expiry_Date` date DEFAULT NULL,
   PRIMARY KEY (`Medicine_Name`)

```
CREATE TABLE `Medicine` (
  `Medicine_Name` varchar(100) NOT NULL,
  `Manufactureres` varchar(100) DEFAULT NULL,
  `Price` double DEFAULT NULL,
  `Stock_Qty` int DEFAULT NULL,
  `Expiry_Date` date DEFAULT NULL,
  PRIMARY KEY (`Medicine_Name`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

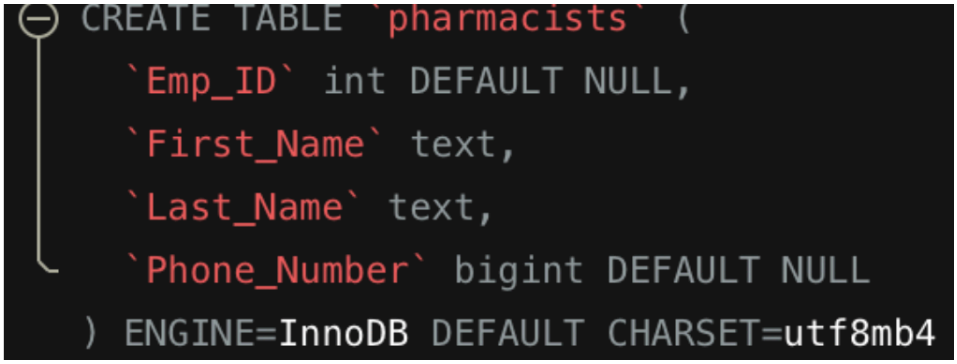
4. Pharmacists:

   Columns → Emp_ID, First_name, Last_Name, Phone_Number

   Primary Key→Emp_ID

   SQL Statement:

CREATE TABLE `pharmacists` (
`Emp_ID` int DEFAULT NULL,
`First_Name` text,
`Last_Name` text,
`Phone_Number` bigint DEFAULT NULL,
PRIMARY KEY ('Emp_ID')



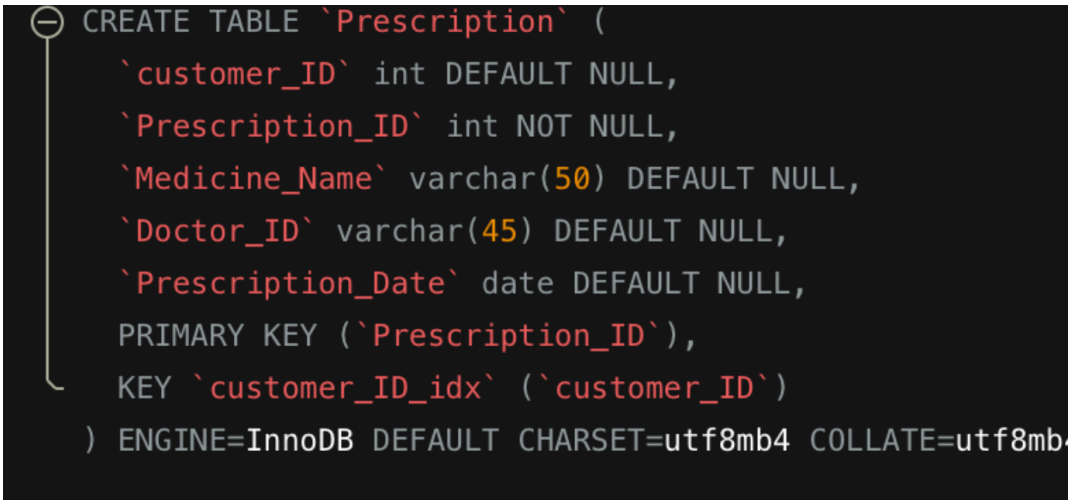
5. Prescription:

Columns → customer_ID, Prescription_ID, Medicine_Name, Doctor_ID, Prescription_Date

Primary Key→Prescripton_ID

SQL Statement:

CREATE TABLE `Prescription` (
`customer_ID` int DEFAULT NULL,
`Prescription_ID` int NOT NULL,
`Medicine_Name` varchar(50) DEFAULT NULL,
`Doctor_ID` varchar(45) DEFAULT NULL,
`Prescription_Date` date DEFAULT NULL,
PRIMARY KEY (`Prescription_ID`)



```
CREATE TABLE `Prescription` (
  `customer_ID` int DEFAULT NULL,
  `Prescription_ID` int NOT NULL,
  `Medicine_Name` varchar(50) DEFAULT NULL,
  `Doctor_ID` varchar(45) DEFAULT NULL,
  `Prescription_Date` date DEFAULT NULL,
  PRIMARY KEY (`Prescription_ID`),
  KEY `customer_ID_idx` (`customer_ID`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb
```

6. Order:

   Columns → Order_ID, Emp_ID, Prescription_ID, Order_Date, Order_qty, Medicine_Name

   Primary Key→Order_ID

   SQL Statement:

```
CREATE TABLE `Orders` (
 `Order_ID` int NOT NULL,
 `Emp_ID` int DEFAULT NULL,
 `Prescription_ID` int DEFAULT NULL,
 `Order_Date` date DEFAULT NULL,
 `Medicine_Name` text,
 `Order_qty` int DEFAULT NULL,
 PRIMARY KEY (`Order_ID`)
)
```

```
CREATE TABLE `Orders` (
  `Order_ID` int NOT NULL,
  `Emp_ID` int DEFAULT NULL,
  `Prescription_ID` int DEFAULT NULL,
  `Order_Date` date DEFAULT NULL,
  PRIMARY KEY (`Order_ID`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 C
```

C. Inserting Data Into Tables

There are two ways to add data into tables:

1- Through SQL Statements:
When the data is not large or you are manually adding a row of data then we can use SQL Statements.

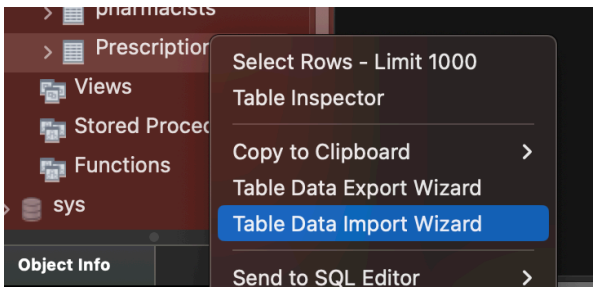
Below is the SQL Syntax:

INSERT INTO *table_name*
VALUES (*value1*, *value2*, *value3*, ...);

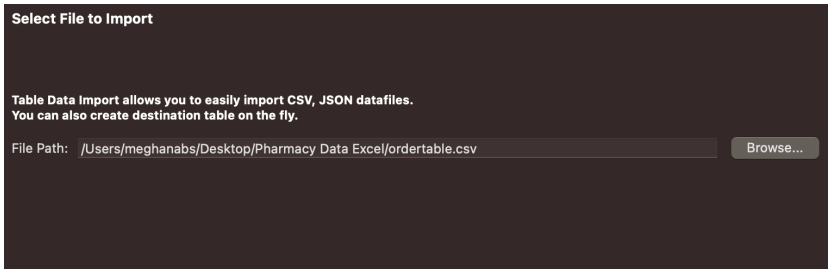
Example:
INSERT INTO Orders
VALUES(23456,6899,76582, 2022-09-24);

2- Uploading CSV files :

-Right click on the table to import data into the table through “Table Date Import Wizard”.

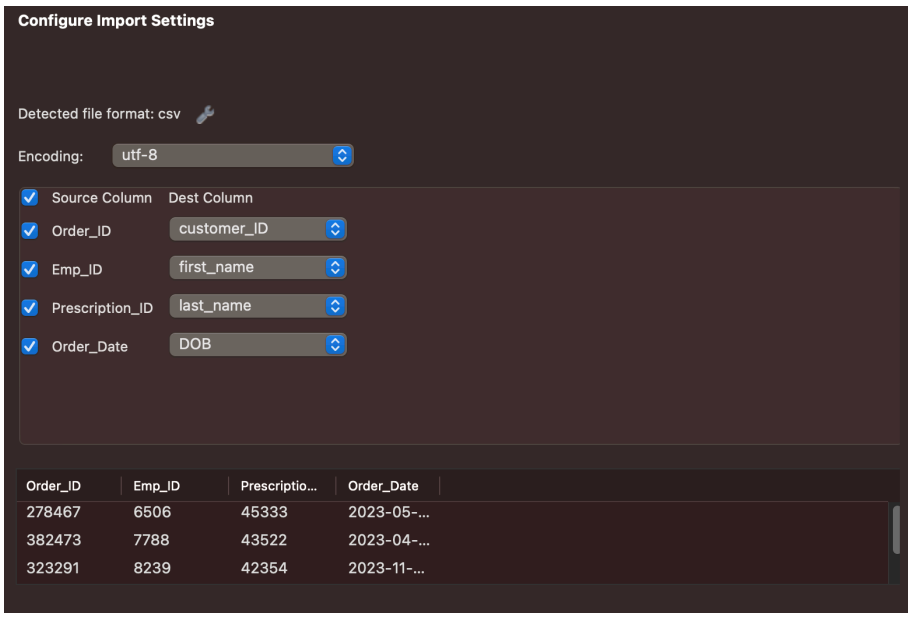


-Select your csv file to import the data



-Select Use existing table and select the table you want to import the data into.

- Set source_column and Destination_columns correctly and import data



For our project we have,
- We have gathered data from multiple websites and cleaned the data to fit our database. -We removed any duplicity in data through Microsoft excel functions as well as SQL.
- We performed data audit and validation on our data to fit our database tables.

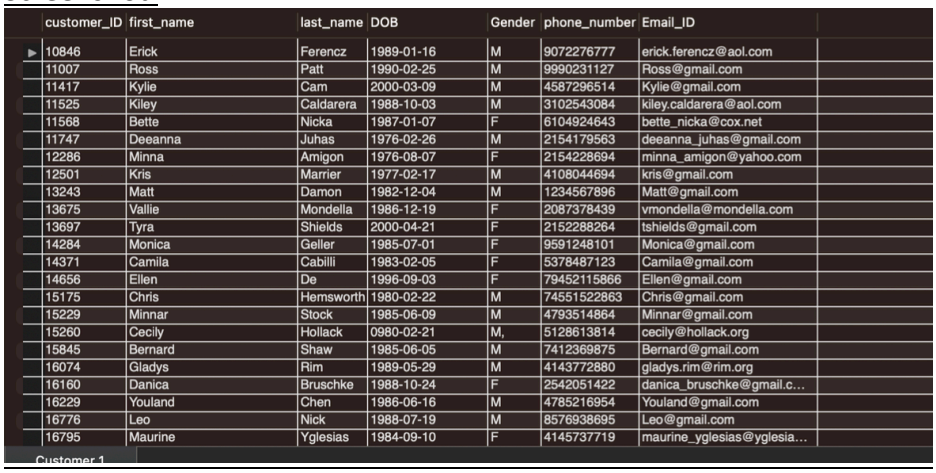
Below are SQL Statements examples to insert Data into our database as well as screenshots of our data fit into our Database Tables:

1. Customer Table:

   SQL Statement

   INSERT INTO `Pharmacy_mangement_system`.`Customer` (`customer_ID`, `first_name`, `last_name`, `DOB`, `Gender`, `phone_number`, `Email_ID`) VALUES('23063','Brock','Bolognia','1997-01-06','M','2126175063','bbolognia@yahoo.com');

   Screenshot:



customer_ID	first_name	last_name	DOB	Gender	phone_number	Email_ID
10846	Erick	Ferencz	1989-01-16	M	9072276777	erick.ferencz@aol.com
11007	Ross	Patt	1990-02-25	M	9990231127	Ross@gmail.com
11417	Kylie	Cam	2000-03-09	M	4587296514	Kylie@gmail.com
11525	Kiley	Caldarera	1988-10-03	M	3102543084	kiley.caldarera@aol.com
11568	Bette	Nicka	1987-01-07	F	6104924643	bette_nicka@cox.net
11747	Deeanna	Juhas	1976-02-26	M	2154179563	deeanna_juhas@gmail.com
12286	Minna	Amigon	1976-08-07	F	2154228694	minna_amigon@yahoo.com
12501	Kris	Marrier	1977-02-17	M	4108044694	kris@gmail.com
13243	Matt	Damon	1982-12-04	M	1234567896	Matt@gmail.com
13675	Vallie	Mondella	1986-12-19	F	2087378439	vmondella@mondella.com
13697	Tyra	Shields	2000-04-21	F	2152288264	tshields@gmail.com
14284	Monica	Geller	1985-07-01	F	9591248101	Monica@gmail.com
14371	Camila	Cabilli	1983-02-05	F	5378487123	Camila@gmail.com
14656	Ellen	De	1996-09-03	F	79452115866	Ellen@gmail.com
15175	Chris	Hemsworth	1980-02-22	M	74551522863	Chris@gmail.com
15229	Minnar	Stock	1985-06-09	M	4793514864	Minnar@gmail.com
15260	Cecily	Hollack	0980-02-21	M,	5128613814	cecily@hollack.org
15845	Bernard	Shaw	1985-06-05	M	7412369875	Bernard@gmail.com
16074	Gladys	Rim	1989-05-29	M	4143772880	gladys.rim@rim.org
16160	Danica	Bruschke	1988-10-24	F	2542051422	danica_bruschke@gmail.c…
16229	Youland	Chen	1986-06-16	M	4785216954	Youland@gmail.com
16776	Leo	Nick	1988-07-19	M	8576938695	Leo@gmail.com
16795	Maurine	Yglesias	1984-09-10	F	4145737719	maurine_yglesias@yglesia…

Customer 1

2. Medicine:

   SQL Statement:

   INSERT INTO `Pharmacy_mangement_system`.`Medicines` (`Medicinesname`, `Manufacturer`, `StockQty`, `Expiry_date`, `Price') VALUES ('Pemazyre','Incyte Biosciences Distribution B.V.','4','2023-01-02','29.00

   Screenshot:

Medicine_Name	Manufactureres	Price	Stock_Qty	Expiry_Date
Adtralza	Sandoz GmbH	55.22	50	2025-02-23
Aubagio	Argenx	10	36	2024-02-23
Biktarvy	Vifor Fresenius Medical Care Renal Pharma Fra…	45.12	12	2025-12-23
Blitzima	H. Lundbeck A/S	40	60	2023-01-23
Byooviz	SUN Pharmaceutical Industries (Europe) B.V.	18	2	2027-02-23
Cevenfacta	Gilead Sciences Ireland UC	33	24	2028-04-23
Comirnaty	Alexion Europe SAS	27	37	2026-02-23
COVID-19 Vaccine (inactivated adjuvanted) Val…	Clovis Oncology Ireland Limited	50	49	2023-06-13
Darunavir Krka	Zentiva, k.s.	38.27	69	2025-02-23
Dexmedetomidine Accord	Accord Healthcare S.L.U.	55.32	5	2023-06-23
Fasturtec	Valneva Austria GmbH	37	10	2027-12-23
Flixabi	Viatris Limited	55	22	2024-08-23
Fuzeon	Dipharma B.V.	53	46	2022-02-24
Genvoya	Eli Lilly Nederland B.V.	29	19	2025-12-23
Ibandronic Acid Teva	Baxalta Innovations GmbH	45.99	8	2022-02-23
Imlygic	Teva B.V.	23.99	2	2023-06-24
Intelence	Moderna Biotech Spain, S.L.	54	29	2025-12-23
Javlor	Substipharm	40	5	2026-02-23
Kaletra	Merck Sharp & Dohme B.V.	13.28	47	2026-12-23
Kevzara	Horizon Therapeutics Ireland DAC	22	1	2025-06-23
Keytruda	Sandoz Pharmaceuticals d.d.	30.12	23	2024-02-23
Kinpeygo	Daiichi Sankyo Europe GmbH	36	21	2022-02-24
Lacosamide Accord	Bristol-Myers Squibb Pharma EEIG	29	2	2024-01-23

3. Pharmacists:

   SQL Statement:

   INSERT INTO `Pharmacy_mangement_system`.`Pharmacists` (`Emp_ID`, `first_name`, `last_name`, `phone_number`) VALUES ('6506','Donald','OConnell','650.507.9833');

   Screenshot:

Emp_ID	First_Name	Last_Name	Phone_Number
6506	Donald	OConnell	6505079833
7788	Douglas	Grant	6505079844
8239	Jennifer	Whalen	5151234444
7853	Michael	Hartstein	5151235555
6899	Pat	Fay	6031236666
9102	Susan	Mavris	5151237777
7570	Hermann	Baer	5151238888
7388	Shelley	Higgins	5151238080
9729	William	Gietz	5151238181
5382	Steven	King	5151234567
9778	Neena	Kochhar	5151234568
7670	Lex	De Haan	5151234569
8035	Alexander	Hunold	5904234567
5082	Bruce	Ernst	5904234568
6150	David	Austin	5904234569
5847	Valli	Pataballa	5904234560
5800	Diana	Lorentz	5904235567
9401	Nancy	Greenberg	5151244569
6032	Daniel	Faviet	5151244169
7494	John	Chen	5151244269
9941	Ismael	Sciarra	5151244369

4. Orders :

   SQL Statement:

   INSERT INTO `Pharmacy_mangement_system`.`Order` (`Order_ID`, `Emp_ID`, `Prescription_ID`, `Order_Date`) VALUES (''234724','6899','76465','2022-09-24');

   Screenshot:

Order_ID	Emp_ID	Prescription_ID	Order_Date
234724	6899	76465	2022-09-24
237417	7853	56744	2022-04-24
278467	6506	45333	2022-12-24
323291	8239	42354	2022-09-24
347643	6798	22987	2022-04-24
348574	7670	13345	2022-04-29
348578	5621	23456	2022-07-24
382473	7788	43522	2022-12-24
384233	7528	98764	2022-05-23
384759	7388	56321	2022-04-04
433853	9102	67574	2022-01-24
437853	5382	66352	2022-11-24
476347	6949	54622	2022-05-24
483756	7570	12356	2022-04-02
584769	9729	14678	2022-04-05
634572	5169	22113	2022-11-24
638475	8272	12342	2022-04-24
645734	5948	54654	2022-04-24
654676	9054	34356	2022-01-24
657464	5228	10675	2022-11-08
734673	7386	24324	2022-03-24
756933	9778	34556	2022-03-02
765347	8346	66533	2022-11-24

5. Ordered_medicines:

   SQL Statement :

   INSERT INTO `Pharmacy_mangement_system`.`Ordered Medicine` (`Order_ID`, `OrderQty`, `DrugName`) VALUES ('205216','Pemazyre','19');

   Screenshot :

Order_ID	Medicine_Name	Order_qty
34537	Pemazyre	8
46873	Pemazyre	6
34536	Tecartus	4
56757	Dexmedetomidine Accord	2
65767	Trumenba	6
56755	Xalkori	8
98767	Macugen	5
78544	Livtencity	6
46887	Flixabi	7
67868	Mysildecard	8
68658	Ocaliva	10
35753	Kevzara	2
35655	Uplizna	3
56212	VidPrevtyn Beta	5
12345	Zynteglo	4
56721	Tyverb	10
21345	Vaxzevria (previously C...	6
32543	Tecfidera	8
90874	Lacosamide Accord	10
46224	Blitzima	6
22435	Truxima	4
23546	Zeposia	10

6. Prescriptions :

   SQL Statement :

   INSERT INTO `Pharmacy_mangement_system`.`Prescription` (`customer_ID`, `Doctor_ID`, `medicine_name`, `Prescription_ID`, `Prescription_Date`) VALUES ('28995','4589','Xalkori','39570','09-11-18');

   Screenshot :

customer_ID	Prescription_ID	Medicine_Name	Doctor_ID	Prescription_Da...
32980	10675	Diclofenac	434879	2022-03-20
29165	12342	Cevenfacta	237423	2018-09-03
47457	12356	Macugen	786321	2018-02-17
28936	13236	Hydroxocobalamin	345347	2021-12-03
22128	13345	Kevzara	544380	2020-05-31
32452	14678	Flixabi	674522	2018-12-02
39884	21358	Suboxone	237423	2020-08-26
40175	22113	Ximluci	274222	2022-08-23
26682	22987	Diazepam	142734	2018-02-28
17725	23456	Truxima	564587	2018-02-17
15175	23522	Paracetamol	342778	2019-04-07
13675	24324	Rozlytrek	436785	2020-09-26
21860	24467	Zynteglo	674522	2022-08-23
14656	25351	Risedronate	587654	2021-03-19
41790	26753	Tazorac	576222	2021-09-21
11747	32154	VidPrevtyn Beta	237423	2018-09-03
42327	32444	Norco	687457	2022-10-02
14284	34113	Hydroxyzine	544380	2021-05-21
36509	34356	Nasonex	344777	2022-11-20
47389	34467	Benztropine	348523	2021-08-17
16074	34556	Ocaliva	457227	2019-08-29
28335	34867	Tyverb	436785	2022-05-21
12501	42354	Tecartus	564587	2022-05-14

7. Doctor:

SQL Statement:

```
INSERT INTO `Pharmacy_mangement_system`.`Doctor` (`Doctor_ID`,`DoctorName`)
VALUES ('1991','ORCHARDJOHN');
```

Screenshot:

Doctor_ID	Doc_Name
137462	SATHRE HOWARD
142734	KORMENDI ROBERT
234723	WILSON JOEL R
237423	AREM , RIDHA
274222	GLASS ROBERT
342778	HAMMOND ISAAC
342874	WOLFF DONALD
344777	JOHN ORCHARD
345347	SHAFFER JAMES
348523	BLOCK MARGARET
364571	GUPTA RAKESH K
434879	LOUISSAINT EDDY
436785	BOURNE GERALD...
454549	ABO-AUDA WAEL
457227	RICE WILLIAM J
458332	MURDOCK KIRK
544380	MCGUIRE KERRY
546577	COLEMAN CONST...
546898	FRELEMIL

Relations and Normalization:

RELATIONS FROM TABLE:

1. **Customer Table :**

Customer_ID	first_name	last_name	DOB	Gender	Phone_number	Email_ID

Primary Key : customer_ID

2. **Pharmacists Table :**

Emp_ID	first_name	last_name	Phone_number

Primary Key: Emp_ID

3. **Doctor Table :**

Doctor_ID	Doc_name

Primary Key : Doctor_ID

4. **Medicine Table :**

Medicine_Name	Manufactureres	price	Stock_qty	Expiry_date

Primary Key : Medicine_Name

5. **Orders Table :**

Order_ID	EMP_ID	Prescription_ID	Order_date

Primary Key : Order_ID
Foreign Key : EMP_ID, Prescription_ID

6. **Ordered_medicines Table :**

Order_ID	Medicine_name	order_qty

Foreign Key : Order_ID, Medicine_name

7. **Prescription Table :**

Prescription_ID	customer_ID	Medicine_Name	Doctor_ID	prescription_date

Primary Key : Prescription_ID
Foreign Key : customer_ID, Medicine_Name, Doctor_ID

NORMALIZATION:

- 1NF : A relation is in first normal from if the following two constrains both apply: -

  - There are no repeating groups in the relation (thus, there is a single fact at the intersection of each row and column of the table).
  - A primary key has been defined, which uniquely identifies each row in the relation.

- 2NF : A database is in second normal form if it satisfies the following conditions:
  - It is in the first normal form.
  - All non-key attributes are fully functional dependent on the primary key or no partial functionally dependencies exists.
- 3NF : A relation is in third normal form if it satisfies the following conditions:
  - It is in the Second Normal form.
  - It has no Transitive Functional Dependency

First Normal Form (1NF) :

1. **Customer Table :**

Customer_ID	first_name	last_name	DOB	Gender	Phone_number	Email_ID

The table is said to be already in 1NF form as it satisfies the conditions of the 1NF Form:

- There are no repeating groups in relation and a primary key defined.

Customer(Customer_ID,first_name,last_name,DOB,Gender,Phone_number,Email_ID)

Customer_ID → first_name, last_name,DOB,Gender,Phone_number,Email_ID

2. **Pharmacists Table**

Emp_ID	first_name	last_name	Phone_number

The table is said to be already in 1NF form as it satisfies the conditions of the 1NF Form:

- There are no repeating groups in relation and a primary key defined.

Pharmacists(Emp_ID,first_name, last_name, phone_number)

Emp_ID → first_name, last_name, phone_number

3. **Doctor Table :**

Doctor_ID	Doc_name

The table is said to be already in 1NF form as it satisfies the conditions of the 1NF Form:

- There are no repeating groups in relation and a primary key defined.

Doctor(Doctor_ID,Doc_name)

Doctor_ID→ Doc_name

4. **Medicine Table:**

Medicine_Name	Manufactureres	price	Stock_qty	Expiry_date

The table is said to be already in 1NF form as it satisfies the conditions of the 1NF Form:

- There are no repeating groups in relation and a primary key defined.

Medicine(Medicine_Name,Manufactureres,price,Stock_qty,Expiry_date)
Medicine_Name → Manufactureres,price,Stock_qty,Expiry_date

5. **Orders Table:**

Order_ID	EMP_ID	Prescription_ID	Medicine_name	order_qty	Order_date

The table is said to be already in 1NF form as it satisfies the conditions of the 1NF Form:

- There are no repeating groups in relation and a primary key defined.

Orders(Order_ID,EMP_ID,Prescription_ID,Medicine_name,order_qty,Order_date)
Order_ID → EMP_ID, Prescription_ID, Medicine_name, order_qty,Order_date

6. **Prescription Table :**

Prescription_ID	customer_ID	Medicine_Name	Doctor_ID	prescription_date

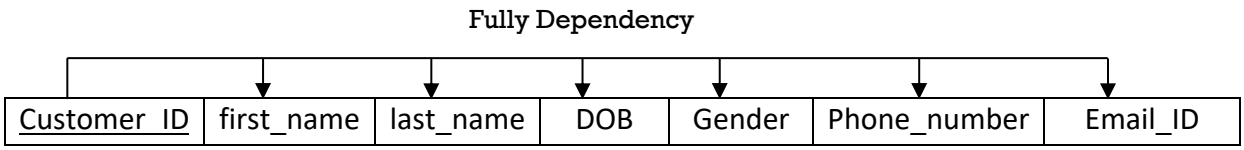
The table is said to be already in 1NF form as it satisfies the conditions of the 1NF Form:

- There are no repeating groups in relation and a primary key defined.

Prescription(Prescription_ID, customer_ID, Medicine_name, Doctor_ID, prescription_date)
Prescription_ID → customer_ID, Medicine_name, Doctor_ID, prescription_date

Second Normal Form (1NF) :

1. **Customer Table :**

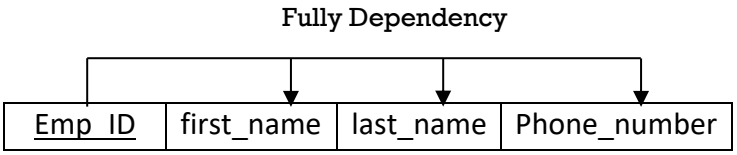


The table is said to be already in 2NF Form as it satisfies the conditions of 2NF form :

-It is in 1NF Form and all non-key attributes are fully dependent on primary key(Customer_ID).

Customer(Customer_ID,first_name,last_name,DOB,Gender,Phone_number,Email_ID)
Customer_ID → first_name, last_name,DOB,Gender,Phone_number,Email_ID

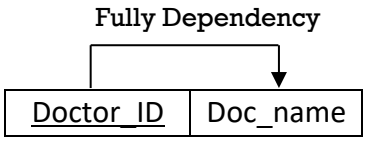
2. Pharmacists Table



The table is said to be already in 2NF Form as it satisfies the conditions of 2NF form :
-It is in 1NF Form and all non-key attributes are fully dependent on primary key(Emp_ID).

Pharmacists(Emp_ID,first_name, last_name, phone_number)
Emp_ID → first_name, last_name, phone_number

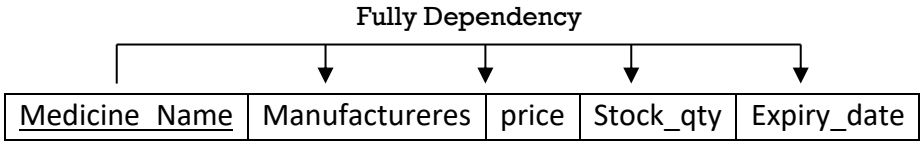
3. Doctor Table :



The table is said to be already in 2NF Form as it satisfies the conditions of 2NF form :
-It is in 1NF Form and all non-key attributes are fully dependent on primary key(Doctor_ID).

Doctor(Doctor_ID,Doc_name)
Doctor_ID→ Doc_name

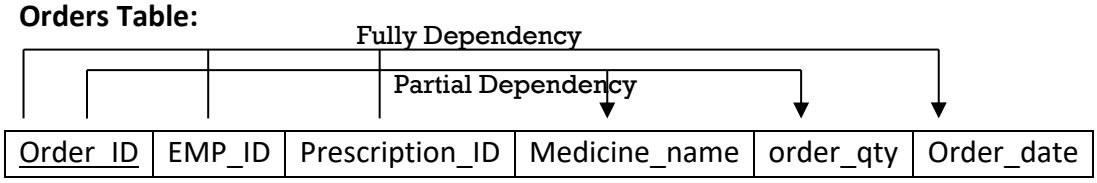
4. **Medicine Table:**



The table is said to be already in 2NF Form as it satisfies the conditions of 2NF form :
-It is in 1NF Form and all non-key attributes are fully dependent on primary key(Medicine_Name).

Medicine(Medicine_Name,Manufactureres,price,Stock_qty,Expiry_date)
Medicine_Name → Manufactureres,price,Stock_qty,Expiry_date

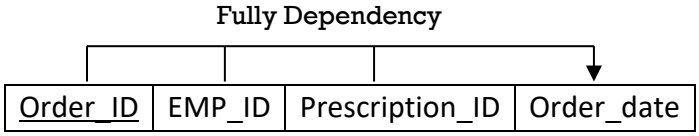
5. **Orders Table:**



Conversion to 2NF :

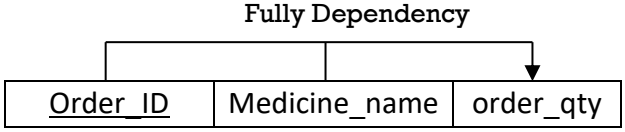
1. To remove Partial Dependency and violation on 2NF, we decompose the above table.
2. Hence there will be a table called Orders and another called ordered_medicines.

Orders Table:



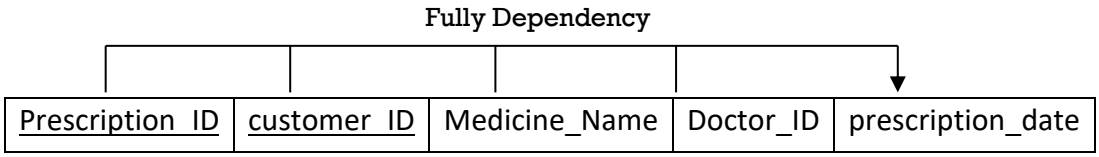
Orders(Order_ID,EMP_ID,Prescription_ID,Order_date)
Order_ID,EMP_ID,Prescription_ID → Order_date

Ordered_medicine :



ordered_medicine(Order_ID,Medicine_name,order_qty)
Order_ID,Medicine_name → order_qty

6. **Prescription :**

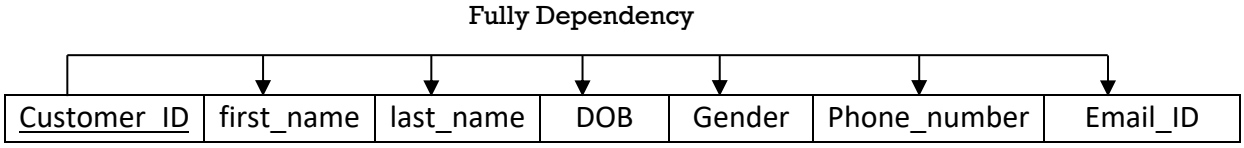


The table is said to be already in 2NF Form as it satisfies the conditions of 2NF form :
-It is in 1NF Form and all non-key attributes are fully dependent on primary key(Prescription_ID).

Presciption(prescription_ID,customer_ID,Medicine_Name, Doctor_ID, prescription_date)
Prescription_ID, customer_ID, Medicine_Name,Doctor_ID → prescription_date

Third Normal Form (1NF) :

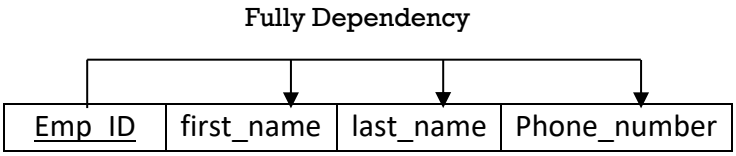
1. **Customer Table :**



The table is said to be already in 3NF Form as it satisfies the conditions of 3NF form :
-It is in 3NF Form and there are no transitive dependencies.

Customer(Customer_ID,first_name,last_name,DOB,Gender,Phone_number,Email_ID)
Customer_ID → first_name, last_name,DOB,Gender,Phone_number,Email_ID

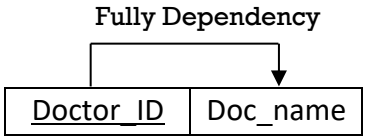
2. **Pharmacists Table**



The table is said to be already in 3NF Form as it satisfies the conditions of 3NF form :
-It is in 3NF Form and there are no transitive dependencies.

Pharmacists(Emp_ID,first_name, last_name, phone_number)
Emp_ID → first_name, last_name, phone_number

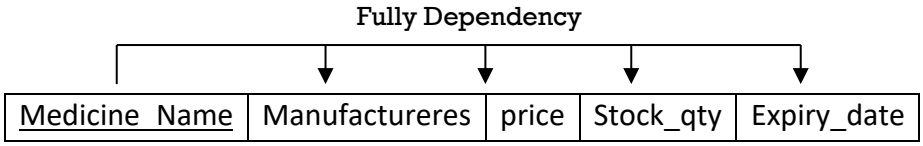
3. Doctor Table :



The table is said to be already in 3NF Form as it satisfies the conditions of 3NF form :
-It is in 3NF Form and there are no transitive dependencies.

Doctor(Doctor_ID,Doc_name)
Doctor_ID→ Doc_name

4. Medicine Table:

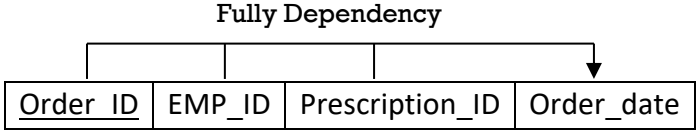


The table is said to be already in 3NF Form as it satisfies the conditions of 3NF form :
-It is in 3NF Form and there are no transitive dependencies.

Medicine(Medicine_Name,Manufactureres,price,Stock_qty,Expiry_date)
Medicine_Name → Manufactureres,price,Stock_qty,Expiry_date

---

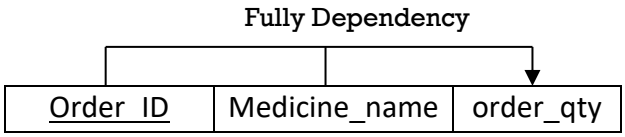
5. Orders Table:



The table is said to be already in 3NF Form as it satisfies the conditions of 3NF form :
-It is in 3NF Form and there are no transitive dependencies.

Orders(Order_ID,EMP_ID,Prescription_ID,Order_date)
Order_ID,EMP_ID,Prescription_ID → Order_date

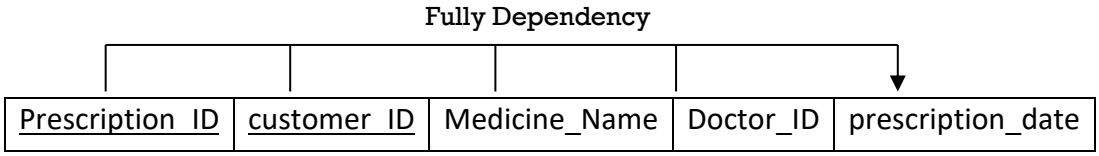
6. Ordered_medicine :



The table is said to be already in 3NF Form as it satisfies the conditions of 3NF form :
-It is in 3NF Form and there are no transitive dependencies.

ordered_medicine(Order_ID,Medicine_name,order_qty)
Order_ID,Medicine_name → order_qty

7. Prescription :

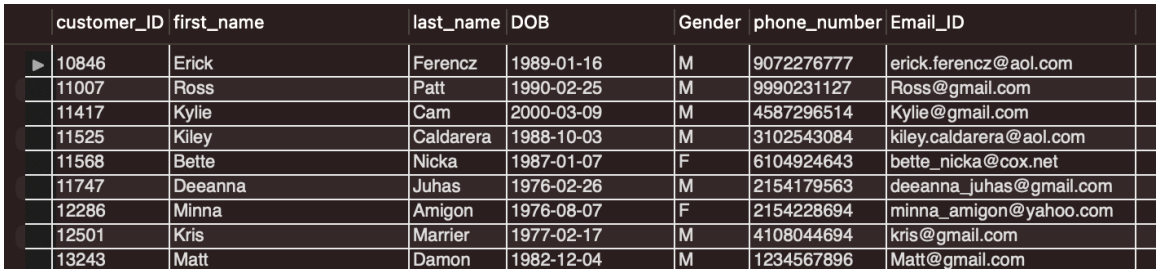


The table is said to be already in 3NF Form as it satisfies the conditions of 3NF form:
-It is in 3NF Form and there are no transitive dependencies.

Presciption(prescription_ID,customer_ID,Medicine_Name, Doctor_ID,
prescription_date)
Prescription_ID, customer_ID, Medicine_Name,Doctor_ID → prescription_date

Normalized Tables and their screenshots:

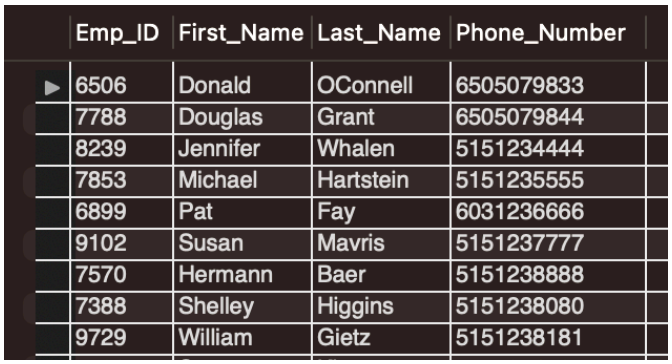
1. **Customer Table :**



customer_ID	first_name	last_name	DOB	Gender	phone_number	Email_ID
10846	Erick	Ferencz	1989-01-16	M	9072276777	erick.ferencz@aol.com
11007	Ross	Patt	1990-02-25	M	9990231127	Ross@gmail.com
11417	Kylie	Cam	2000-03-09	M	4587296514	Kylie@gmail.com
11525	Kiley	Caldarera	1988-10-03	M	3102543084	kiley.caldarera@aol.com
11568	Bette	Nicka	1987-01-07	F	6104924643	bette_nicka@cox.net
11747	Deeanna	Juhas	1976-02-26	M	2154179563	deeanna_juhas@gmail.com
12286	Minna	Amigon	1976-08-07	F	2154228694	minna_amigon@yahoo.com
12501	Kris	Marrier	1977-02-17	M	4108044694	kris@gmail.com
13243	Matt	Damon	1982-12-04	M	1234567896	Matt@gmail.com

Customer(Customer_ID,first_name,last_name,DOB,Gender,Phone_number,Email_ID
)
Customer_ID → first_name, last_name,DOB,Gender,Phone_number,Email_ID

2. **Pharmacists Table :**



Emp_ID	First_Name	Last_Name	Phone_Number
6506	Donald	OConnell	6505079833
7788	Douglas	Grant	6505079844
8239	Jennifer	Whalen	5151234444
7853	Michael	Hartstein	5151235555
6899	Pat	Fay	6031236666
9102	Susan	Mavris	5151237777
7570	Hermann	Baer	5151238888
7388	Shelley	Higgins	5151238080
9729	William	Gietz	5151238181

Pharmacists(Emp_ID,first_name, last_name, phone_number)
Emp_ID → first_name, last_name, phone_number

3. **Doctor Table:**

Doctor_ID	Doc_Name
137462	SATHRE HOWARD
142734	KORMENDI ROBERT
234723	WILSON JOEL R
237423	AREM , RIDHA
274222	GLASS ROBERT
342778	HAMMOND ISAAC
342874	WOLFF DONALD
344777	JOHN ORCHARD
345347	SHAFFER JAMES

Doctor(Doctor_ID,Doc_name)
Doctor_ID→ Doc_name

4. **Medicine Table:**

Medicine_Name	Manufactureres	Price	Stock_Qty	Expiry_Date
Adtralza	Sandoz GmbH	55.22	50	2025-02-23
Aubagio	Argenx	10	36	2024-02-23
Biktarvy	Vifor Fresenius Medical Care Renal Pharma Fra…	45.12	12	2025-12-23
Blitzima	H. Lundbeck A/S	40	60	2023-01-23
Byooviz	SUN Pharmaceutical Industries (Europe) B.V.	18	2	2027-02-23
Cevenfacta	Gilead Sciences Ireland UC	33	24	2028-04-23
Comirnaty	Alexion Europe SAS	27	37	2026-02-23
COVID-19 Vaccine (inactivated adjuvanted) Val…	Clovis Oncology Ireland Limited	50	49	2023-06-13
Darunavir Krka	Zentiva, k.s.	38.27	69	2025-02-23

Medicine(Medicine_Name,Manufactureres,price,Stock_qty,Expiry_date)
Medicine_Name → Manufactureres,price,Stock_qty,Expiry_date

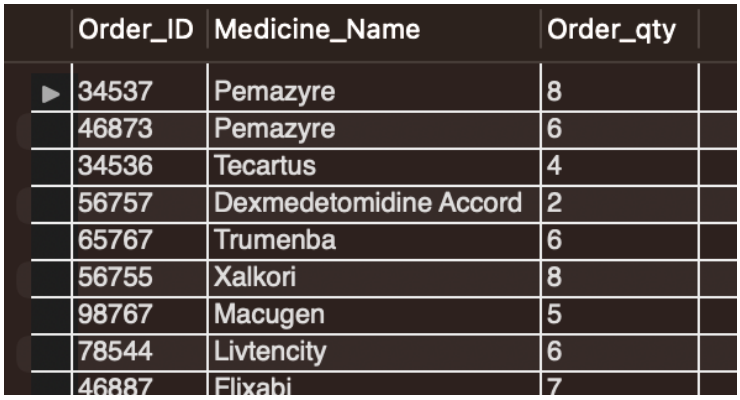
5. **Orders Table :**

Order_ID	Emp_ID	Prescription_ID	Order_Date
234724	6899	76465	2022-09-24
237417	7853	56744	2022-04-24
278467	6506	45333	2022-12-24
323291	8239	42354	2022-09-24
347643	6798	22987	2022-04-24
348574	7670	13345	2022-04-29
348578	5621	23456	2022-07-24
382473	7788	43522	2022-12-24
384233	7528	98764	2022-05-23

Orders(Order_ID,EMP_ID,Prescription_ID,Order_date)

Order_ID,EMP_ID,Prescription_ID → Order_date

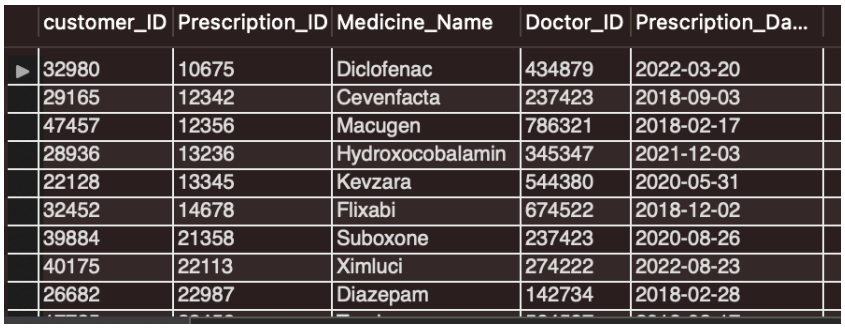
6. **Ordered_medicines Table :**



Order_ID	Medicine_Name	Order_qty
34537	Pemazyre	8
46873	Pemazyre	6
34536	Tecartus	4
56757	Dexmedetomidine Accord	2
65767	Trumenba	6
56755	Xalkori	8
98767	Macugen	5
78544	Livtencity	6
46887	Flixabi	7

ordered_medicine(Order_ID,Medicine_name,order_qty)
Order_ID,Medicine_name → order_qty

7. **Prescriptions Table:**



customer_ID	Prescription_ID	Medicine_Name	Doctor_ID	Prescription_Da...
32980	10675	Diclofenac	434879	2022-03-20
29165	12342	Cevenfacta	237423	2018-09-03
47457	12356	Macugen	786321	2018-02-17
28936	13236	Hydroxocobalamin	345347	2021-12-03
22128	13345	Kevzara	544380	2020-05-31
32452	14678	Flixabi	674522	2018-12-02
39884	21358	Suboxone	237423	2020-08-26
40175	22113	Ximluci	274222	2022-08-23
26682	22987	Diazepam	142734	2018-02-28

Presciption(prescription_ID,customer_ID,Medicine_Name, Doctor_ID, prescription_date)
Prescription_ID, customer_ID, Medicine_Name,Doctor_ID → prescription_date

Usecases:

Teammate 1: Meghana Bangalore Srikantha

Creating Usecases Using Joins

Use Case 1: Extract customers details and their prescriptions between a range of dates.

Select Prescription.Prescription_ID, Customer.customer_ID, Customer.first_name, Prescription.Prescription_Date
FROM Prescription
INNER JOIN Customer ON Prescription.customer_ID=Customer.customer_ID WHERE Prescription_Date Between '2022-05-21' AND '2022-11-22';

```
Select Prescription.Prescription_ID, Customer.customer_ID, Customer.first_name, Prescription.Prescription_Date
FROM Prescription
INNER JOIN Customer ON Prescription.customer_ID=Customer.customer_ID
WHERE Prescription_Date Between '2022-05-21' AND '2022-11-22';
```

Output:

Prescription_ID	customer_ID	first_name	Prescription_Date
22113	40175	Valentine	2022-08-23
24467	21860	Blondell	2022-08-23
32444	42327	Ina	2022-10-02
34356	36509	Roxane	2022-11-20
34867	28335	Jamal	2022-05-21
45243	31565	Lemerson	2022-08-21
45333	37663	Leota	2022-05-23
58774	40439	Simona	2022-08-13

Result 2

Use Case 2: Extract Medicine Names that are less than 10 number in stock from the Ordered Medicines Table along with the Order Quantity.

SELECT Medicine.Medicine_Name, Medicine.Stock_Qty,ordered_medicines.Order_qty From Medicine
RIGHT JOIN ordered_medicines ON Medicine.Medicine_Name = ordered_medicines.Medicine_Name WHERE Stock_Qty < '10';

```
SELECT Medicine.Medicine_Name, Medicine.Stock_Qty,ordered_medicines.Order_qty From Medicine
RIGHT JOIN ordered_medicines ON Medicine.Medicine_Name = ordered_medicines.Medicine_Name
WHERE Stock_Qty < '10';
```

Output:

Medicine_Name	Stock_Qty	Order_qty
Pemazyre	4	8
Pemazyre	4	6
Tecartus	3	4
Dexmedetomidine Accord	5	2
Mysildecard	7	8
Ocaliva	9	10
Kevzara	1	2
Lacosamide Accord	2	10

Use Case 3: To extract the data of pharmacists who worked on orders for particular dates

SELECT pharmacists.Emp_ID, Orders.Order_ID,Orders.Order_Date FROM pharmacists
LEFT JOIN Orders
ON pharmacists.Emp_ID = Orders.Emp_ID
Order By Order_Date DESC;

```
SELECT pharmacists.Emp_ID, Orders.Order_ID,Orders.Order_Date FROM pharmacists
LEFT JOIN Orders
ON pharmacists.Emp_ID = Orders.Emp_ID
Order By Order_Date DESC;
```

Output:

Emp_ID	Order_ID	Order_Date
6506	278467	2022-12-24
7788	382473	2022-12-24
5169	634572	2022-11-24
8346	765347	2022-11-24
5465	987677	2022-11-24
5382	437853	2022-11-24
5228	657464	2022-11-08
8239	323291	2022-09-24

Use Case 4: To extract the data of the doctor who prescribed the medicines.

SELECT Doctor.Doctor_ID,Doctor.Doc_Name, Prescription.Prescription_ID From Doctor
RIGHT JOIN Prescription
ON Doctor.Doctor_ID = Prescription.Doctor_ID
WHERE Doc_Name LIKE '%l%';

```
SELECT Doctor.Doctor_ID,Doctor.Doc_Name, Prescription.Prescription_ID From Doctor
RIGHT JOIN Prescription
ON Doctor.Doctor_ID = Prescription.Doctor_ID
WHERE Doc_Name LIKE '%l%';
```

Output:

Doctor_ID	Doc_Name	Prescription_ID
434879	LOUISSAINT EDDY	10675
786321	FITZSIMMONS DANIEL	12356
674522	COLOMBO DAVID	14678
274222	GLASS ROBERT	22113
436785	BOURNE GERALD WESLEY	24324
674522	COLOMBO DAVID	24467
687457	ULLOA RAUL	32444
348523	BLOCK MARGARET	34467

Use Case 5: Extracts the customer_ID, Prescription_ID and Order_ID

SELECT Customer.customer_ID, Prescription.Prescription_ID, Orders.Order_ID From Customer
JOIN Prescription
ON Customer.customer_ID = Prescription.customer_ID
JOIN Orders
ON Prescription.Prescription_ID = Orders.Prescription_ID;

```
SELECT Customer.customer_ID, Prescription.Prescription_ID, Orders.Order_ID From Customer
JOIN Prescription
ON Customer.customer_ID = Prescription.customer_ID
JOIN Orders
ON Prescription.Prescription_ID = Orders.Prescription_ID;
```

Output:

customer_ID	Prescription_ID	Order_ID
35584	76465	234724
12286	56744	237417
37663	45333	278467
12501	42354	323291
26682	22987	347643
22128	13345	348574
17725	23456	348578
47389	43522	382473

Result 6

Creating Views For Usecases:

1.Extract customers details and their prescriptions between a range of dates.

```
Create View customer_prescription AS
Select Prescription.Prescription_ID,Customer.customer_ID, Customer.first_name,
Prescription.Prescription_Date
FROM Prescription
INNER JOIN Customer ON Prescription.customer_ID=Customer.customer_ID WHERE
Prescription_Date Between '2022-05-21' AND '2022-11-22';
Select * from customer_prescription;
```

Screenshot :

Prescription_ID	customer_ID	first_name	Prescription_Da...
22113	40175	Valentine	2022-08-23
24467	21860	Blondell	2022-08-23
32444	42327	Ina	2022-10-02
34356	36509	Roxane	2022-11-20
34867	28335	Jamal	2022-05-21
45243	31565	Lemerson	2022-08-21
45333	37663	Leota	2022-05-23
58774	40439	Simona	2022-08-13

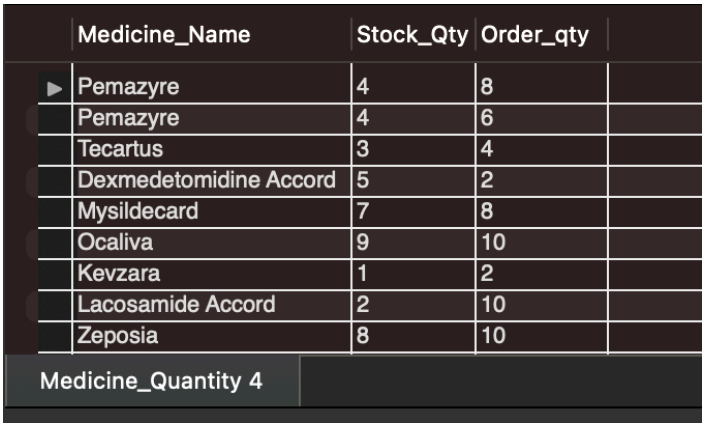
customer_prescription 2

ction Output

2. Extract Medicine Names that are less than 10 number in stock from the Ordered Medicines Table along with the Order Quantity.

```
CREATE VIEW Medicine_Quantity AS
SELECT Medicine.Medicine_Name,Medicine.Stock_Qty,ordered_medicines.Order_qty From
Medicine
RIGHT JOIN ordered_medicines ON Medicine.Medicine_Name =
ordered_medicines.Medicine_Name
WHERE Stock_Qty < '10';
Select * from Medicine_Quantity;
```

Screenshot :



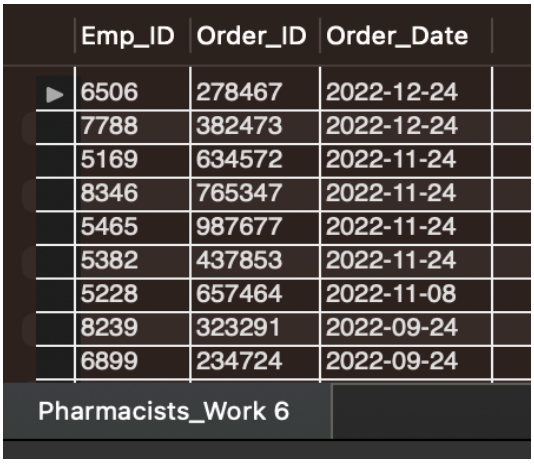
Medicine_Name	Stock_Qty	Order_qty
Pemazyre	4	8
Pemazyre	4	6
Tecartus	3	4
Dexmedetomidine Accord	5	2
Mysildecard	7	8
Ocaliva	9	10
Kevzara	1	2
Lacosamide Accord	2	10
Zeposia	8	10

Medicine_Quantity 4

3. **To extract the data of pharmacists who worked on orders for particular dates**

```
CREATE VIEW Pharmacists_Work AS
SELECT pharmacists.Emp_ID, Orders.Order_ID,Orders.Order_Date FROM pharmacists
LEFT JOIN Orders
ON pharmacists.Emp_ID = Orders.Emp_ID
Order By Order_Date DESC;
SELECT * FROM Pharmacists_Work LIMIT 0, 1000
```

Screenshot :



Emp_ID	Order_ID	Order_Date
6506	278467	2022-12-24
7788	382473	2022-12-24
5169	634572	2022-11-24
8346	765347	2022-11-24
5465	987677	2022-11-24
5382	437853	2022-11-24
5228	657464	2022-11-08
8239	323291	2022-09-24
6899	234724	2022-09-24

Pharmacists_Work 6

4. **To extract the data of the doctor who prescribed the medicines.**

```
Create view Doctor_prescriptions AS
SELECT Doctor.Doctor_ID,Doctor.Doc_Name, Prescription.Prescription_ID
From Doctor
RIGHT JOIN Prescription
ON Doctor.Doctor_ID = Prescription.Doctor_ID
WHERE Doc_Name LIKE '%l%';
SELECT * FROM Doctor_Prescriptions;
```

Screenshot :

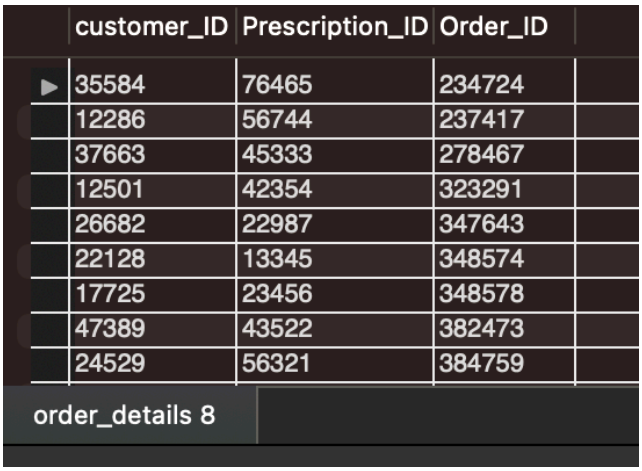
Doctor_ID	Doc_Name	Prescription_ID
434879	LOUISSAINT EDDY	10675
786321	FITZSIMMONS DANIEL	12356
674522	COLOMBO DAVID	14678
274222	GLASS ROBERT	22113
436785	BOURNE GERALD WESLEY	24324
674522	COLOMBO DAVID	24467
687457	ULLOA RAUL	32444
348523	BLOCK MARGARET	34467
457227	RICE WILLIAM J	34556

Doctor_Prescriptions 7

5. **Extracts the customer_ID, Prescription_ID and Order_ID**

```
Create view order_details AS
SELECT Customer.customer_ID, Prescription.Prescription_ID, Orders.Order_ID From Customer
JOIN Prescription
ON Customer.customer_ID = Prescription.customer_ID
JOIN Orders
ON Prescription.Prescription_ID = Orders.Prescription_ID;
SELECT * FROM order_details;
```

Screenshot :



customer_ID	Prescription_ID	Order_ID
35584	76465	234724
12286	56744	237417
37663	45333	278467
12501	42354	323291
26682	22987	347643
22128	13345	348574
17725	23456	348578
47389	43522	382473
24529	56321	384759

order_details 8

Teammate 2: Pooja Kuberaiah

Creating Usecases Using Joins

Use Case 1: To view which customer has ordered medicine on which date

SELECT Orders.Order_ID, Orders.Prescription_ID, Orders.OrderDate
FROM Orders
INNER JOIN Customer ON Orders.Prescription_ID=Customer.Customer_ID;

```
SELECT Orders.Order_ID, Orders.Prescription_ID, Orders.OrderDate
FROM Orders
INNER JOIN Customer ON Orders.Prescription_ID=Customer.Customer_ID;
```

UseCase2: To extract the customers data who ordered medicine with its name and quantity

SELECT Ordered Medicine.OrderQty, Ordered Medicine.Drugname
FROM Ordered Medicine
LEFT JOIN Customer ON Customers.CustomerID = Orders.CustomerID
ORDER BY Customers.First_Name, Customers.Last_Name;

```
SELECT ordered_medicine.Order_qty, ordered_Medicine.Medicine_Name
FROM ordered_medicine
LEFT JOIN Customer ON Customers.Customer_ID = Orders.Customer_ID
ORDER BY Customers.First_Name, Customers.Last_Name;
```

UseCase3: To view the details of pharmacist handling the customer order

SELECT Pharmacists.First_Name, Pharmacists.Last_Name
FROM Pharmacists
RIGHT JOIN Order ON Pharmacists.Emp_ID =Order.Order_ID;

```
SELECT Pharmacists.First_Name, Pharmacists.Last_Name
FROM Pharmacists
RIGHT JOIN Orders ON Pharmacists.Emp_ID =Orders.Order_ID;
```

UseCase4: To view the medicine prescribed by the doctor

SELECT Prescription.Doc_ID, Prescription. Prescription_ID, Prescription_MedicineName
FROM Prescriptions
FULL OUTER JOIN Doctor ON Prescription.Doc_ID= Doctor.Doc_name;

```
SELECT Prescription.Doc_ID, Prescription. Prescription_ID, Prescription_MedicineName
FROM Prescriptions
LEFT JOIN Doctor ON Prescription.Doc_ID= Doctor.Doc_name;
```

UseCase5: To view the customer data with medicine name, doctor_id , prescription

SELECT Prescription.Prescription_ID, Prescription.Customer_ID, Prescription.Doc_ID, Prescription.MedicineName
FROM Prescription
INNER JOIN Customers ON Prescription.Prescription_ID=Customers.CustomerID
ORDER BY Customers.CustomerID, Prescription.MedicineName, Prescription.Doc_ID;

```
SELECT Prescription.Prescription_ID, Prescription.Customer_ID, Prescription.Doc_ID, Prescription.MedicineName
FROM Prescription
INNER JOIN Customers ON Prescription.Prescription_ID=Customers.CustomerID
ORDER BY Customers.CustomerID, Prescription.MedicineName, Prescription.Doc_ID;
```

Creating Views For Usecases:

UsedCase1: To view which customer has ordered medicine on which date

CREATE VIEW customer_order AS
SELECT Order.Order_ID, Order.Prescription_ID, Order.OrderDate
FROM Order
INNER JOIN Customers ON Orders.Prescription_ID=Customers.CustomerID;

UseCase2: To extract the customers data who ordered medicine with its name and quantity

CREATE VIEW customer_Dets AS
SELECT Ordered Medicine.OrderQty, Ordered Medicine.Drugname
FROM Ordered Medicine
LEFT JOIN Customer ON Customers.CustomerID = Orders.CustomerID
ORDER BY Customers.First_Name, Customers.Last_Name;

UseCase3: To view the details of pharmacist handling the customer order

```
CREATE VIEW pharmacists_order AS
SELECT Pharmacists.First_Name, Pharmacists.Last_Name
FROM Pharmacists
RIGHT JOIN Order ON Pharmacists.Emp_ID =Order.Order_ID;
```

UseCase4: To view the medicine prescribed by the doctor

```
CREATE VIEW medicine_prescribed AS
SELECT Prescription.Doc_ID, Prescription. Prescription_ID, Prescription_MedicineName
FROM Prescriptions
FULL OUTER JOIN Doctor ON Prescription.Doc_ID= Doctor.Doc_name;
```

UseCase5: To view the customer data with medicine name, doctor_id , prescription

```
CREATE VIEW customer_data AS
SELECT Prescription.Prescription_ID, Prescription.Customer_ID, Prescription.Doc_ID,
Prescription.MedicineName
FROM Prescription
INNER JOIN Customers ON Prescription.Prescription_ID=Customers.CustomerID
ORDER BY Customers.CustomerID, Prescription.MedicineName, Prescription.Doc_ID;
```