Hospital Database Management System

Ankita Yadav

Background Information:

Hospitals are the primary source of healthcare for millions of people. With the increasing demand for medical services and the need for better quality of care, it is imperative that hospitals have a robust and efficient system for managing patient data. A well-designed Hospital Database Management system can provide real-time access to patient information, improve the quality of care, and increase operational efficiency.

Business Problem:

The manual system of storing and managing patient data in hospitals is no longer sufficient to meet the demands of modern healthcare. The manual system is time-consuming, prone to errors, and not scalable. This leads to slow access to patient information, longer wait times, and a lower quality of care. Additionally, the manual system does not provide any reporting or analytics capabilities, which makes it difficult for hospital management to make informed decisions based on patient data. Moreover, this system is not equipped to handle the growing demand for medical services and the need for a better quality of care.

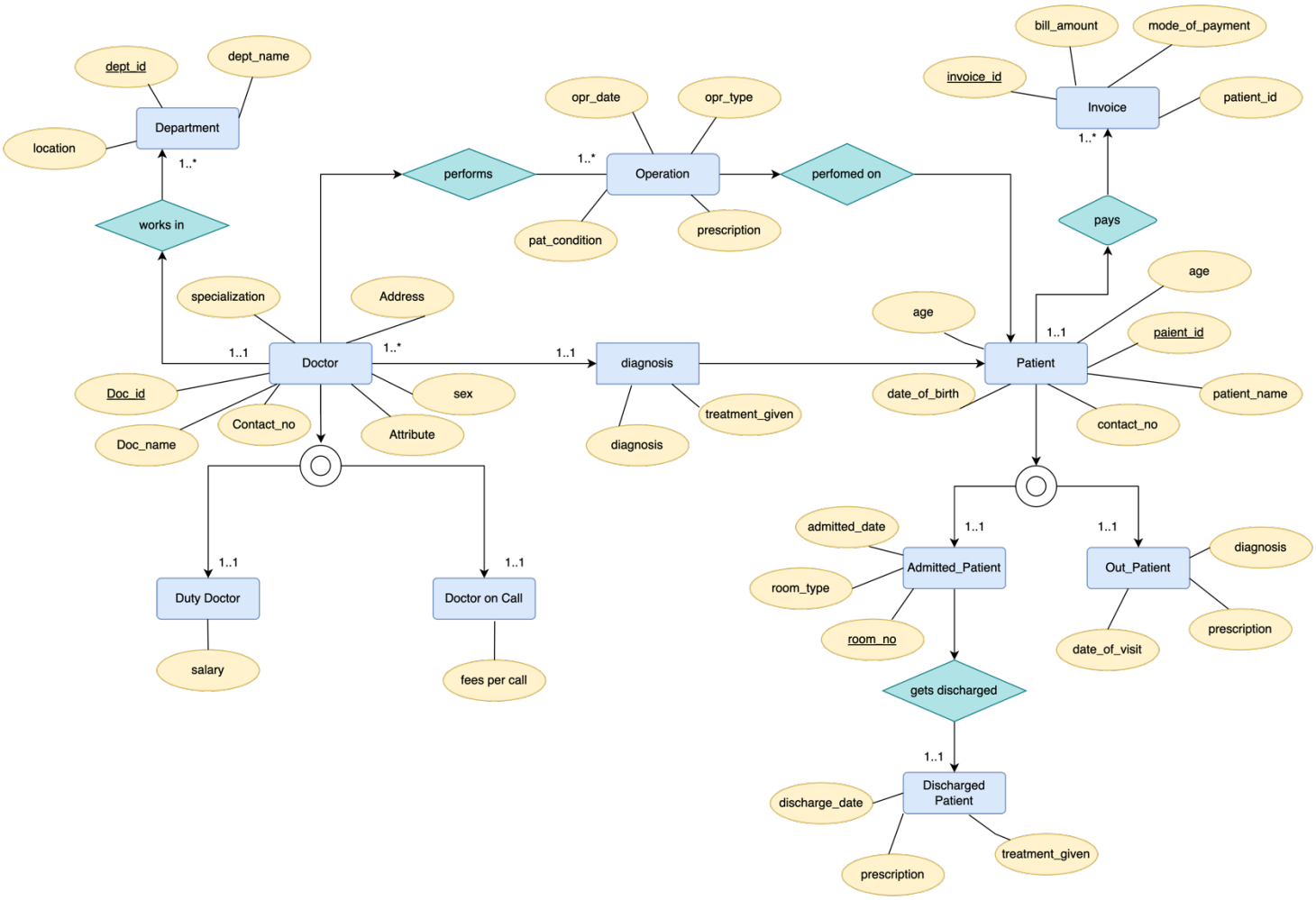
The lack of a proper database management system also poses a risk to patient privacy and confidentiality. The manual system lacks security measures to protect patient information from unauthorized access. The system is also vulnerable to human errors, such as misfiling or misplacing of patient records, which can result in data breaches and loss of patient trust.

Description:

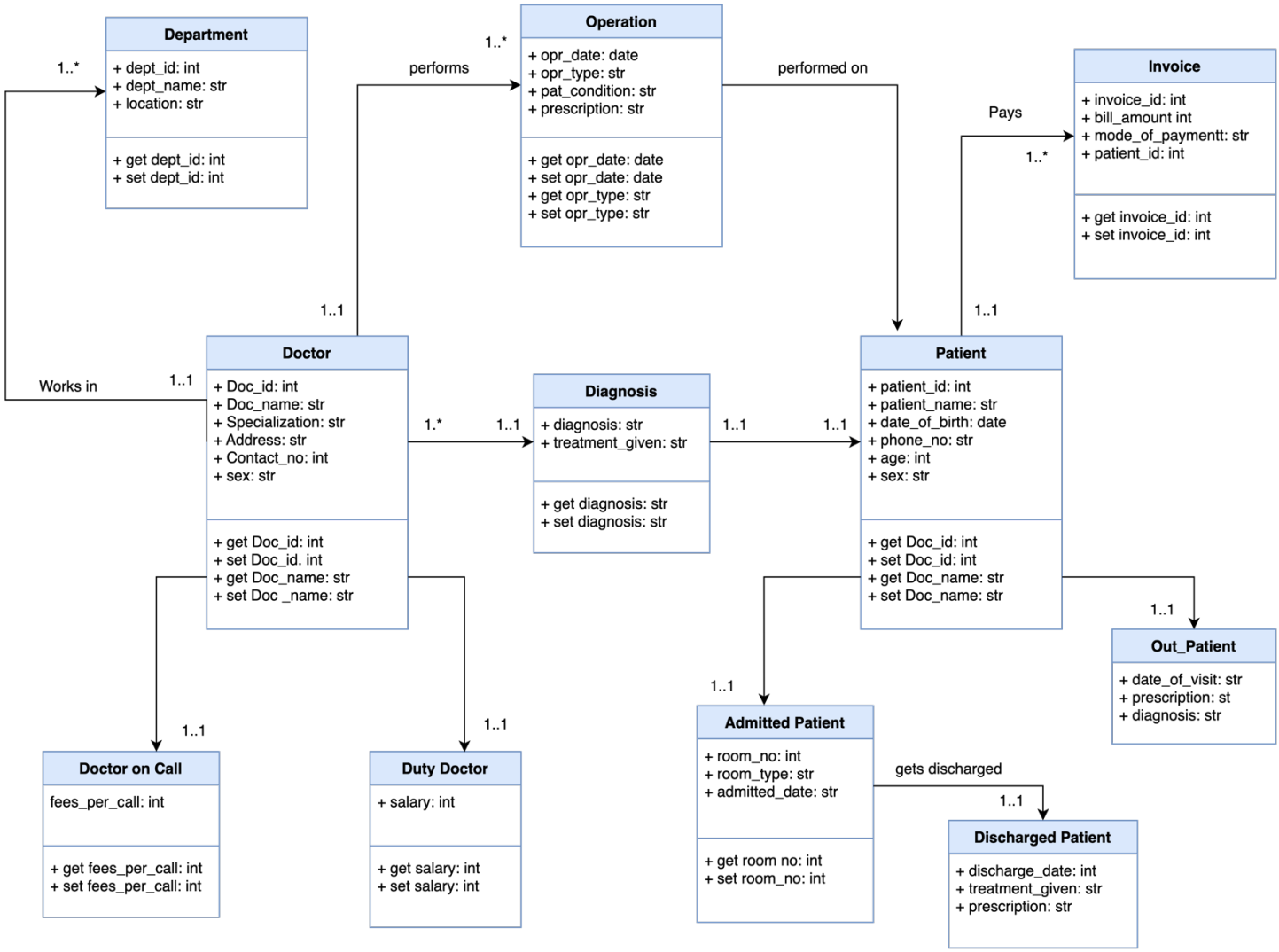
At the hospital reception, patients are given an entry card for check-ups from the relevant doctors. After being recorded, patients visit the assigned doctor for their check-ups and any necessary treatment. Depending on their situation, they may be admitted to the hospital and have the option of choosing a private or general room. Before admission, they must complete a basic contact information form and pay any pre-treatment fees. The same process must be followed upon discharge. The final amount owed is determined based on the treatment received, tests conducted, operations performed, and insurance coverage. The hospital has multiple departments, such as cancer, neurology, pathology, and gynecology, due to its multiple specialties. There are two types of doctors: duty doctors who make daily visits to the hospital, and call-on doctors who are called in for special circumstances if the assigned doctor is not available.

Conceptual Diagrams

1. ERR Diagram:



2. UML Diagram:



Mapping Conceptual Model to Relational Model

Primary Key: Bold Underlined
Foreign Key: Bold Italicized

1. DEPARTMENTS (**DEPT_ID**, DEPT_LOCATION, DEPT_NAME)
2. DOCTORS (**DOC_ID**, DOC_NAME, SPECIALIZATION, ADDRESS, PHONE_NO, DEPT_NO)
3. DUTY_DOC (**DOC_ID**, DOC_NAME, SALARY)
4. DOCTOR_ON_CALL (**DOC_ID**, DOC_NAME, FEES_PER_CL)
5. PATIENT_ENTRY (**PATIENT_ID**, PATIENT_NAME, AGE, SEX, PHONE_NO, CHECKUP_DATE, **DEPT_ID**)
6. PATIENT_DIAGNOSE (**PATIENT_ID**, DIAGNOSIS, TREATMENT_GIVEN, **DOC_ID**, **DEPT_ID**)
7. PATIENT_ADMIT (**PATIENT_ID**, ADVANCE_PYMT, PAYMENT_MODE, **DEPT_ID**, ADMITTED_DATE, DIAGNOSIS, **ROOM_NO**)
8. PATIENT_DISCHARGE (**PATIENT_ID**, DISCHARGE_DATE, TREATMENT_GIVEN, MEDICINE, PAYMENT_MODE)
9. PATIENT_REGULAR (**PATIENT_ID**, DATE_OF_VISIT, DIAGNOSIS, TREATMENT_GIVEN, MEDICINE, MODE_OF_PAYMENT, **DOC_ID**)
10. PATIENT_OPERATED (**PATIENT_ID**, DATE_OF_OPERATION, PATIENT_CONDITION, OPERATION_DURATION, MEDICINES, DIAGNOSIS, **DOC_ID**)
11. ROOM_DETAILS (**ROOM_NO**, ROOM_TYPE, STATUS)
12. INVOICE (**INVOICE_ID**, FEES_AMOUNT, MODE_OF_PMT, **PATIENT_ID**)

Implementation of NoSQL

Implementation Model:

```
CREATE DATABASE `Hospital Management System`;
USE `Hospital Management System`;
```

Creating the table for the schema Hospital Management System:

Table 1: DEPARTMENTS

This table stores information about hospital departments, including the department ID, department name, and department location.

```
CREATE TABLE DEPARTMENTS (
    DEPT_ID INT PRIMARY KEY,
    DEPT_LOCATION VARCHAR(50),
    DEPT_NAME VARCHAR(50)
);
```

Table 2: DOCTORS

This table stores information about the doctors in the hospital, including their ID, name, specialization, address, phone number, and department ID. The department ID is a foreign key referencing the DEPARTMENTS table.

```
CREATE TABLE DOCTORS (
    DOC_ID VARCHAR(50) PRIMARY KEY,
    DOC_NAME VARCHAR(50),
    SPECIALIZATION VARCHAR(50),
    ADDRESS VARCHAR(100),
    PHONE_NO VARCHAR(20),
    DEPT_NO INT,
    FOREIGN KEY (DEPT_NO) REFERENCES DEPARTMENTS(DEPT_ID)
);
```

Table 3: DUTY_DOC

This table stores information about the doctors on duty, including their ID, name, and salary.

```sql
CREATE TABLE DUTY_DOC (
    DOC_ID VARCHAR(50) PRIMARY KEY,
    DOC_NAME VARCHAR(50),
    SALARY FLOAT
);
```

Table 4: DOCTOR_ON_CALL

This table stores information about the doctors on call, including their ID, name, and fees per consultation.

```sql
CREATE TABLE DOCTOR_ON_CALL (
    DOC_ID VARCHAR(50) PRIMARY KEY,
    DOC_NAME VARCHAR(50),
    FEES_PER_CL FLOAT
);
```

Table 5: PATIENT_ENTRY

This table stores information about the patients who visit the hospital, including their ID, name, age, sex, phone number, checkup date, and department ID. The department ID is a foreign key referencing the DEPARTMENTS table.

```sql
CREATE TABLE PATIENT_ENTRY (
    PATIENT_ID VARCHAR(10) PRIMARY KEY,
    PATIENT_NAME VARCHAR(50),
    AGE INT,
    SEX VARCHAR(10),
    PHONE_NO VARCHAR(20),
    CHECKUP_DATE DATE,
    DEPT_ID INT,
    FOREIGN KEY (DEPT_ID) REFERENCES DEPARTMENTS(DEPT_ID)
);
```

Table 6: PATIENT_DIAGNOSE

This table stores information about the diagnosis and treatment given to patients, including the patient ID, diagnosis, treatment given, doctor ID, and department ID. The patient ID, doctor ID, and department ID are foreign keys referencing the PATIENT_ENTRY, DOCTORS, and DEPARTMENTS tables, respectively.

```
CREATE TABLE PATIENT_DIAGNOSE (
  PATIENT_ID VARCHAR(10),
  DIAGNOSIS VARCHAR(100),
  TREATMENT_GIVEN VARCHAR(100),
  DOC_ID VARCHAR(50),
  DEPT_ID INT,
  FOREIGN KEY (PATIENT_ID) REFERENCES PATIENT_ENTRY(PATIENT_ID),
  FOREIGN KEY (DOC_ID) REFERENCES DOCTORS(DOC_ID),
  FOREIGN KEY (DEPT_ID) REFERENCES DEPARTMENTS(DEPT_ID)
);
```

Table 7: PATIENT_ADMIT

This table stores information about the patients who are admitted to the hospital, including their ID, advance payment, payment mode, department ID, admitted date, diagnosis, and room number. The patient ID and department ID are foreign keys referencing the PATIENT_ENTRY and DEPARTMENTS tables, respectively.

```
CREATE TABLE PATIENT_ADMIT (
  PATIENT_ID VARCHAR(10) PRIMARY KEY,
  ADVANCE_PYMT FLOAT,
  PAYMENT_MODE VARCHAR(20),
  DEPT_ID INT,
  ADMITTED_DATE DATE,
  DIAGNOSIS VARCHAR(100),
  ROOM_NO INT,
  FOREIGN KEY (PATIENT_ID) REFERENCES PATIENT_ENTRY(PATIENT_ID),
  FOREIGN KEY (DEPT_ID) REFERENCES DEPARTMENTS(DEPT_ID)
);
```

Table 8: PATIENT_DISCHARGE

This table stores information about the patients who are discharged from the hospital, including their ID, discharge date, treatment given, medicine, and payment mode. The patient ID is a foreign key referencing the PATIENT_ENTRY table.

```
CREATE TABLE PATIENT_DISCHARGE (
    PATIENT_ID VARCHAR(10),
    DISCHARGE_DATE DATE,
    TREATMENT_GIVEN VARCHAR(100),
    MEDICINE VARCHAR(100),
    PAYMENT_MODE VARCHAR(20),
    FOREIGN KEY (PATIENT_ID) REFERENCES PATIENT_ENTRY(PATIENT_ID)
);
```

Table 9: PATIENT_REGULAR

This table stores information about the patients who visit the hospital regularly, including their ID, date of visit, diagnosis, treatment given, medicine, payment mode, and doctor ID. The patient ID and doctor ID are foreign keys referencing the PATIENT_ENTRY and DOCTORS tables, respectively.

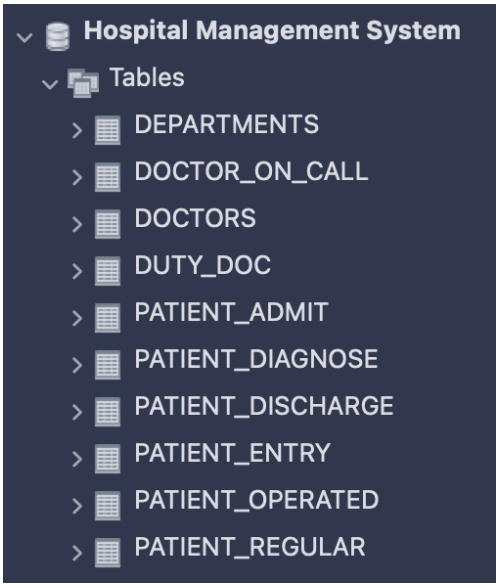
```
CREATE TABLE PATIENT_DISCHARGE (
    PATIENT_ID VARCHAR(10),
    DISCHARGE_DATE DATE,
    TREATMENT_GIVEN VARCHAR(100),
    MEDICINE VARCHAR(100),
    PAYMENT_MODE VARCHAR(20),
    FOREIGN KEY (PATIENT_ID) REFERENCES PATIENT_ENTRY(PATIENT_ID)
);
```

Table 10: PATIENT_OPERATED

This table stores information about the patients who undergo surgery, including their ID, date of operation, patient condition, operation duration, medicine, diagnosis, and doctor ID. The patient ID is a foreign key referencing the PATIENT_ENTRY table.

```
CREATE TABLE PATIENT_OPERATED (
  PATIENT_ID VARCHAR(10),
  DATE_OF_OPERATION DATE,
  PATIENT_CONDITION VARCHAR(100),
  OPERATION_DURATION INT,
  MEDICINES VARCHAR(100),
  DIAGNOSIS VARCHAR(100),
  DOC_ID VARCHAR(50),
  FOREIGN KEY (PATIENT_ID) REFERENCES PATIENT_ENTRY(PATIENT_ID)
  );
```

After performing these queries, the Hospital Management System will have these tables and will look like this:



This is how all he data in above-mentioned Tables look like:

```
1  SELECT * FROM DEPARTMENTS;
2
3
4
5
```

100% 15:1

Result Grid | Filter Rows: Search | Edit: | Export/Import:

DEPT_ID	DEPT_LOCATION	DEPT_NAME
1	Floor2	Cardiology
2	Floor2	Oncology
3	Floor1	Neurology
4	Floor3	Orthopedics
5	Floor3	Dermatology
6	Floor2	Gastroenterology
7	Floor3	Psychiatry
8	Floor3	Pediatrics
9	Floor1	Radiology
10	Floor3	Urology
11	Floor1	Hematology
12	Floor3	Nephrology
13	Floor2	Pulmonology
14	Floor3	Endocrinology
15	Floor3	Ophthalmology

```
1 •  SELECT * FROM DOCTOR_ON_CALL;
```

100% 30:1

Result Grid Filter Rows: Search Edit: Export/Import:

DOC_ID	DOC_NAME	FEES_PER_CL
DOC001	John Smith	67786.6
DOC002	Jane Doe	875981
DOC003	Robert Johnson	957560
DOC004	Emily Chen	876982
DOC010	Avery Adams	800000
DOC016	Nicholas Anderson	925000
DOC017	Madison Hernandez	725000
DOC018	Gabriel Green	825000
DOC019	Evelyn Rivera	737250
DOC020	Alexander Scott	850000
DOC024	David Lee	897500
DOC025	Maria Hernandez	852500
DOC028	Christian Bale	850000

```
1 •  SELECT * FROM DOCTORS;
```

100% 15:1

Result Grid Filter Rows: Search Edit: Export/Import:

DOC_ID	DOC_NAME	SPECIALIZATION	ADDRESS	PHONE_NO	DEPT_NO
DOC001	Dr. John Smith	Cardiology	123 Main St., Anytown, USA	555-1234	1
DOC002	Jane Doe	Pediatrics	456 Elm St., Anytown, USA	555-5678	8
DOC003	Robert Johnson	Oncology	789 Oak St., Anytown, USA	555-9012	2
DOC004	Emily Chen	Dermatology	321 Maple St., Anytown, USA	555-3456	5
DOC005	David Kim	Neurology	654 Pine St., Anytown, USA	555-7890	3
DOC006	Maria Rodriguez	Urology	987 Cedar St., Anytown, USA	555-2345	10
DOC007	Michael Lee	Gastroenterology	654 Birch St., Anytown, USA	555-6789	6
DOC008	Megan Taylor	Endocrinology	321 Oak St., Anytown, USA	555-0123	14
DOC009	William Davis	Cardiology	789 Maple St., Anytown, USA	555-4567	1
DOC010	Avery Adams	Pediatrics	123 Pine St., Anytown, USA	555-8901	8
DOC011	Sophia Wilson	Oncology	456 Cedar St., Anytown, USA	555-2345	2
DOC012	Ethan Garcia	Dermatology	789 Birch St., Anytown, USA	555-6789	5
DOC013	Isabella Herna...	Neurology	321 Main St., Anytown, USA	555-0123	3
DOC014	Ryan Martin	Psychiatry	654 Elm St., Anytown, USA	555-4567	7
DOC015	Nora Perez	Gastroenterology	987 Oak St., Anytown, USA	555-8901	6
DOC016	Nicholas Ander...	Endocrinology	123 Cedar St., Anytown, USA	555-2345	14

```
1 •  SELECT * FROM DUTY_DOC;
```

100% 15:1

Result Grid Filter Rows: Search Edit: Export/Import:

DOC_ID	DOC_NAME	SALARY
DOC005	David Kim	820000
DOC006	Maria Rodriguez	760001
DOC007	Michael Lee	820555
DOC008	Megan Taylor	779871
DOC009	William Davis	950700
DOC011	Sophia Wilson	650550
DOC012	Ethan Garcia	855689
DOC013	Isabella Hernandez	950000
DOC014	Ryan Martin	667200
DOC015	Nora Perez	895000
DOC021	Hazel Torres	87000.5
DOC022	Christian Flores	89000.7
DOC023	Sarah Smith	779300
DOC024	David Lee	89400.6
DOC025	Maria Hernandez	92000.9
DOC026	Sarah Flores	792215
DOC027	Alexander Malik	895000

```
1 •  SELECT * FROM PATIENT_ENTRY;
```

100% 15:1

Result Grid Filter Rows: Search Edit: Export/Import:

PATIENT_ID	PATIENT_NAME	AGE	SEX	PHONE_NO	CHECKUP_DATE	DEPT_ID
P0001	John Doe	35	Male	555-1234	2023-03-17	1
P0002	Jane Smith	42	Female	555-5678	2023-03-17	2
P0004	Alice Brown	55	Female	555-4321	2023-03-18	3
P0005	Sam Lee	45	Male	555-3456	2023-03-19	2
P0006	Emily Davis	31	Female	555-8765	2023-03-19	7
P0007	Tom Wilson	50	Male	555-1111	2023-03-20	8
P0008	Sara Kim	23	Female	555-2222	2023-03-20	9
P0009	Alex Chen	38	Male	555-3333	2023-03-21	10
P0010	Grace Lee	27	Female	555-4444	2023-03-21	11
P0011	Mike Smith	47	Male	555-5555	2023-03-22	12
P0012	Jenny Kim	33	Female	555-6666	2023-03-22	13
P0013	David Brown	60	Male	555-7777	2023-03-23	14

```
1 •  SELECT * FROM PATIENT_ADMIT;
```

100% 29:1

Result Grid Filter Rows: Search Edit: Export/Import:

PATIENT_ID	ADVANCE_PYMT	PAYMENT_MODE	DEPT_ID	ADMITTED_DATE	DIAGNOSIS	ROOM_NO
P0002	2000	Cash	2	2023-03-18	High blood pressure	202
P0003	1000	Debit Card	3	2023-03-18	Migraine	303
P0004	2500	Cash	3	2023-03-18	Asthma	304
P0005	3000	Credit Card	2	2023-03-18	Sprained ankle	205
P0006	4000	Cash	7	2023-03-18	Depression	706
P0007	5000	Debit Card	8	2023-03-18	Heart disease	807
P0008	2500	Credit Card	9	2023-03-18	Migraine	908
P0009	1500	Cash	10	2023-03-18	Kidney stones	9
P0010	3000	Debit Card	11	2023-03-18	Depression	110
P0011	2500	Credit Card	12	2023-03-18	High cholesterol	211
P0012	2000	Cash	1	2023-03-18	Insomnia	102
P0013	4500	Debit Card	14	2023-03-18	Diabetes	314
P0014	1500	Cash	15	2023-03-18	Common cold	415
P0015	7500	Credit Card	15	2023-03-18	Broken arm	516
P0016	3000	Cash	3	2023-03-18	Sinus infection	17
P0017	2500	Credit Card	4	2023-03-18	Asthma	18

```
1 •  SELECT * FROM PATIENT_ADMIT;
```

100% 15:1

Result Grid Filter Rows: Search Edit: Export/Import:

PATIENT_ID	ADVANCE_PYMT	PAYMENT_MODE	DEPT_ID	ADMITTED_DATE	DIAGNOSIS	ROOM_NO
P0002	2000	Cash	2	2023-03-18	High blood pressure	202
P0003	1000	Debit Card	3	2023-03-18	Migraine	303
P0004	2500	Cash	3	2023-03-18	Asthma	304
P0005	3000	Credit Card	2	2023-03-18	Sprained ankle	205
P0006	4000	Cash	7	2023-03-18	Depression	706
P0007	5000	Debit Card	8	2023-03-18	Heart disease	807
P0008	2500	Credit Card	9	2023-03-18	Migraine	908
P0009	1500	Cash	10	2023-03-18	Kidney stones	9
P0010	3000	Debit Card	11	2023-03-18	Depression	110
P0011	2500	Credit Card	12	2023-03-18	High cholesterol	211
P0012	2000	Cash	1	2023-03-18	Insomnia	102
P0013	4500	Debit Card	14	2023-03-18	Diabetes	314
P0014	1500	Cash	15	2023-03-18	Common cold	415
P0015	7500	Credit Card	15	2023-03-18	Broken arm	516
P0016	3000	Cash	3	2023-03-18	Sinus infection	17
P0017	2500	Credit Card	4	2023-03-18	Asthma	18
P0018	4000	Debit Card	5	2023-03-18	Anxiety	119
P0020	2000	Credit Card	7	2023-03-18	Food poisoning	121
P0030	1500	Cash	4	2023-03-18	Broken leg	120
P0033	1500	Credit Card	5	2023-03-18	Heart disease	101

```
1 •  SELECT * FROM PATIENT_REGULAR;
```

100% 31:1

Result Grid Filter Rows: Search Export:

PATIENT_ID	DATE_OF_VISIT	DIAGNOSIS	TREATMENT_GIVEN	MEDICINE	MODE_OF_PAYMENT	DOC_ID
P0003	2022-01-03	Migraine	Painkillers	Ibuprofen	Cash	D0003
P0004	2022-01-04	Asthma	Inhaler	Albuterol	Insurance	D0004
P0005	2022-01-05	Sprained ankle	Rest and ice	Ibuprofen	Cash	D0003
P0006	2022-01-06	Depression	Therapy sessions	Fluoxetine	Credit card	D0006
P0007	2022-01-07	Heart disease	Surgery	Aspirin	Insurance	D0004
P0008	2022-01-08	Appendicitis	Surgery	Morphine	Cash	D0005
P0009	2022-01-09	Kidney stones	Painkillers and fluids	Oxycodone	Credit card	D0009
P0010	2022-01-10	Depression	Counseling and medication	Citalopram	Cash	D0006
P0011	2022-01-11	High cholesterol	Prescription medication	Atorvastatin	Insurance	D0007
P0012	2022-01-12	Insomnia	Prescription for sleep aid	Zolpidem	Cash	D0001
P0013	2022-01-13	Diabetes	Insulin injections	Insulin	Credit card	D0008
P0014	2022-01-14	Common cold	Rest and fluids	Paracetamol	Cash	D0009
P0015	2022-01-15	Broken arm	Surgery and rehabilitation	Codeine	Insurance	D0010
P0016	2022-01-16	Sinus infection	Antibiotics	Amoxicillin	Credit card	D0011

Performing SQL Queries:

Query 1: Retrieve all the doctors from the DOCTORS table along with their department information:

```
SELECT D.*, DEPT_NAME, DEPT_LOCATION FROM DOCTORS D
JOIN DEPARTMENTS DEPT ON D.DEPT_NO = DEPT.DEPT_ID;
```

100% 1:5

Result Grid Filter Rows: Search Export:

DOC_ID	DOC_NAME	SPECIALIZATION	ADDRESS	PHONE_NO	DEPT_NO	DEPT_NAME	DEPT_LOCATION
DOC001	Dr. John Smith	Cardiology	123 Main St., Anytown, USA	555-1234	1	Cardiology	Floor2
DOC009	William Davis	Cardiology	789 Maple St., Anytown, USA	555-4567	1	Cardiology	Floor2
DOC003	Robert Johnson	Oncology	789 Oak St., Anytown, USA	555-9012	2	Oncology	Floor2
DOC011	Sophia Wilson	Oncology	456 Cedar St., Anytown, USA	555-2345	2	Oncology	Floor2
DOC019	Evelyn Rivera	Oncology	321 Pine St., Anytown, USA	555-4567	2	Oncology	Floor2
DOC005	David Kim	Neurology	654 Pine St., Anytown, USA	555-7890	3	Neurology	Floor1
DOC013	Isabella Hernandez	Neurology	321 Main St., Anytown, USA	555-0123	3	Neurology	Floor1

Result Grid

Form Editor

Query 2: Retrieve all the patients who had a checkup on a specific date:

```
SELECT * FROM PATIENT_ENTRY WHERE CHECKUP_DATE = '2023-03-20';
```

100% 1:7

Result Grid Filter Rows: Search Edit: Export/Import:

PATIENT_ID	PATIENT_NAME	AGE	SEX	PHONE_NO	CHECKUP_DATE	DEPT_ID
P0007	Tom Wilson	50	Male	555-1111	2023-03-20	8
P0008	Sara Kim	23	Female	555-2222	2023-03-20	9
P0024	Samantha Winters	25	Female	555-0456	2023-03-20	9

Query 3: Retrieve the name and id of doctors who have a salary greater than 90000:

```
SELECT DOC_ID, DOC_NAME FROM DUTY_DOC WHERE SALARY > 90000;
```

Result Grid | Filter Rows: Search | Edit: | Export/Import:

DOC_ID	DOC_NAME
DOC005	David Kim
DOC006	Maria Rodriguez
DOC007	Michael Lee
DOC008	Megan Taylor
DOC009	William Davis
DOC011	Sophia Wilson
DOC012	Ethan Garcia
DOC013	Isabella Hernandez
DOC014	Ryan Martin
DOC015	Nora Perez
DOC023	Sarah Smith
DOC025	Maria Hernandez
DOC026	Sarah Flores
DOC027	Alexander Malik

Query 4: Retrieve the names of patients who have been diagnosed with a specific condition and treated by a specific doctor:

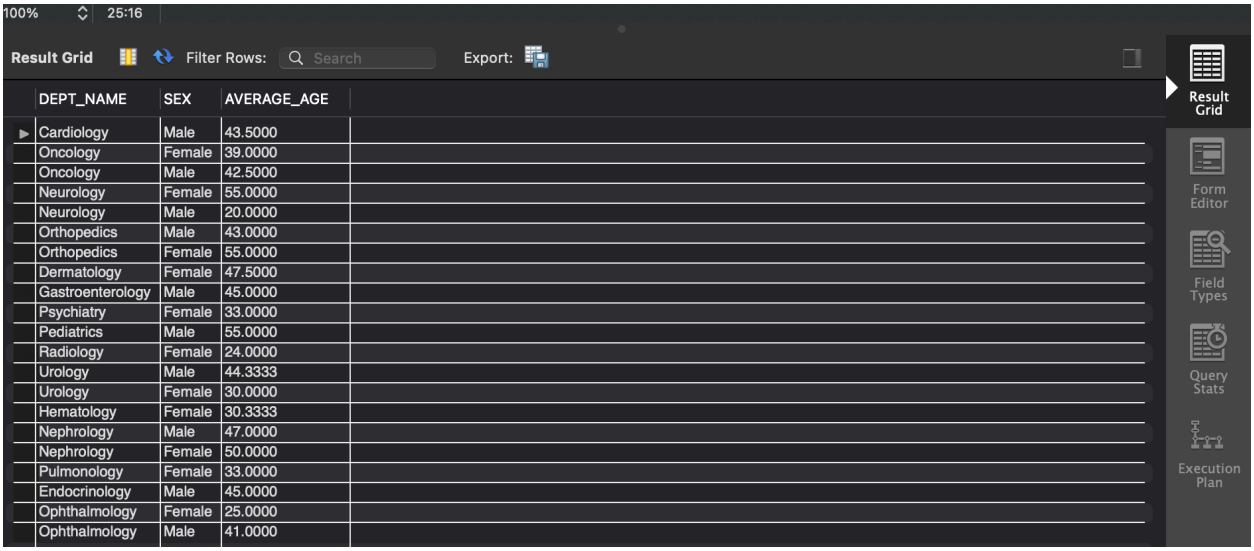
```
SELECT PATIENT_NAME FROM PATIENT_ENTRY PE
JOIN PATIENT_DIAGNOSE PD ON PE.PATIENT_ID = PD.PATIENT_ID
WHERE PD.DIAGNOSIS = 'Heart Disease';
```

Result Grid | Filter Rows: Search | Export:

PATIENT_NAME	DOC_ID	TREATMENT_GIVEN
Tom Wilson	D0004	Surgery
Daniel Wilson	D0005	Medication and lifestyle changes

Query 5: Retrieve the average age of male and female patients for each department:

```
SELECT DEPT_NAME, SEX, AVG(AGE) AS AVERAGE_AGE FROM PATIENT_ENTRY
JOIN DEPARTMENTS ON PATIENT_ENTRY.DEPT_ID = DEPARTMENTS.DEPT_ID
GROUP BY DEPT_NAME, SEX;
```

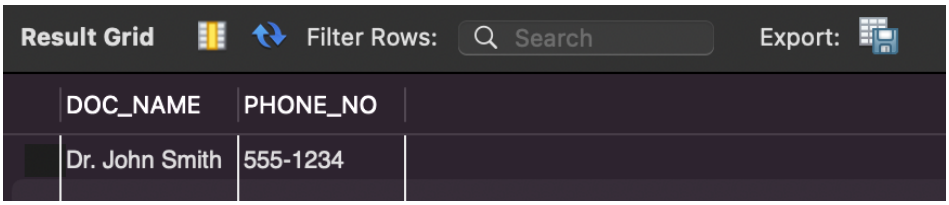


DEPT_NAME	SEX	AVERAGE_AGE
Cardiology	Male	43.5000
Oncology	Female	39.0000
Oncology	Male	42.5000
Neurology	Female	55.0000
Neurology	Male	20.0000
Orthopedics	Male	43.0000
Orthopedics	Female	55.0000
Dermatology	Female	47.5000
Gastroenterology	Male	45.0000
Psychiatry	Female	33.0000
Pediatrics	Male	55.0000
Radiology	Female	24.0000
Urology	Male	44.3333
Urology	Female	30.0000
Hematology	Female	30.3333
Nephrology	Male	47.0000
Nephrology	Female	50.0000
Pulmonology	Female	33.0000
Endocrinology	Male	45.0000
Ophthalmology	Female	25.0000
Ophthalmology	Male	41.0000

`

Query 6: Retrieve the Doctor names and their Phone Number and of all doctors in the "Cardiology" department:

```
SELECT DOC_NAME, PHONE_NO
FROM DOCTORS
WHERE DEPT_NO = (SELECT DEPT_ID
                 FROM DEPARTMENTS
                 WHERE DEPT_NAME = 'Cardiology')
  AND DOC_ID IN (SELECT DOC_ID
                 FROM PATIENT_DIAGNOSE
                 WHERE DEPT_ID = (SELECT DEPT_ID
                                  FROM DEPARTMENTS
                                  WHERE DEPT_NAME = 'Cardiology'));
```



Query 7: Retrieve the names and phone numbers of all doctors who have treated patients with the same diagnosis as patient with ID 'P0011'

```
SELECT DOC_ID, DOC_NAME, PHONE_NO
FROM DOCTORS
WHERE DOC_ID IN (SELECT DOC_ID
                 FROM PATIENT_DIAGNOSE
                 WHERE DIAGNOSIS = (SELECT DIAGNOSIS
                                    FROM PATIENT_DIAGNOSE
                                    WHERE PATIENT_ID = 'P0011'));
```

Result Grid | Filter Rows: Search | Edit: | Export/Import:

PATIENT_ID	PATIENT_NAME	AGE	SEX	PHONE_NO	CHECKUP_DATE	DEPT_ID
P0007	Tom Wilson	50	Male	555-1111	2023-03-20	8
P0008	Sara Kim	23	Female	555-2222	2023-03-20	9
P0024	Samantha Winters	25	Female	555-0456	2023-03-20	9
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Query 8: Retrieve the list of all doctors on duty and doctors on call:

```
SELECT *
FROM DUTY_DOC
UNION
SELECT DOC_ID, DOC_NAME, NULL
FROM DOCTOR_ON_CALL;
```

DOC_ID	DOC_NAME	SALARY
DOC005	David Kim	820000
DOC006	Maria Rodriguez	760001
DOC007	Michael Lee	820555
DOC008	Megan Taylor	779871
DOC009	William Davis	950700
DOC011	Sophia Wilson	650550
DOC012	Ethan Garcia	855689
DOC013	Isabella Hernandez	950000
DOC014	Ryan Martin	667200
DOC015	Nora Perez	895000
DOC021	Hazel Torres	87000.5
DOC022	Christian Flores	89000.7
DOC023	Sarah Smith	779300
DOC024	David Lee	89400.6
DOC025	Maria Hernandez	92000.9
DOC026	Sarah Flores	792215
DOC027	Alexander Malik	895000

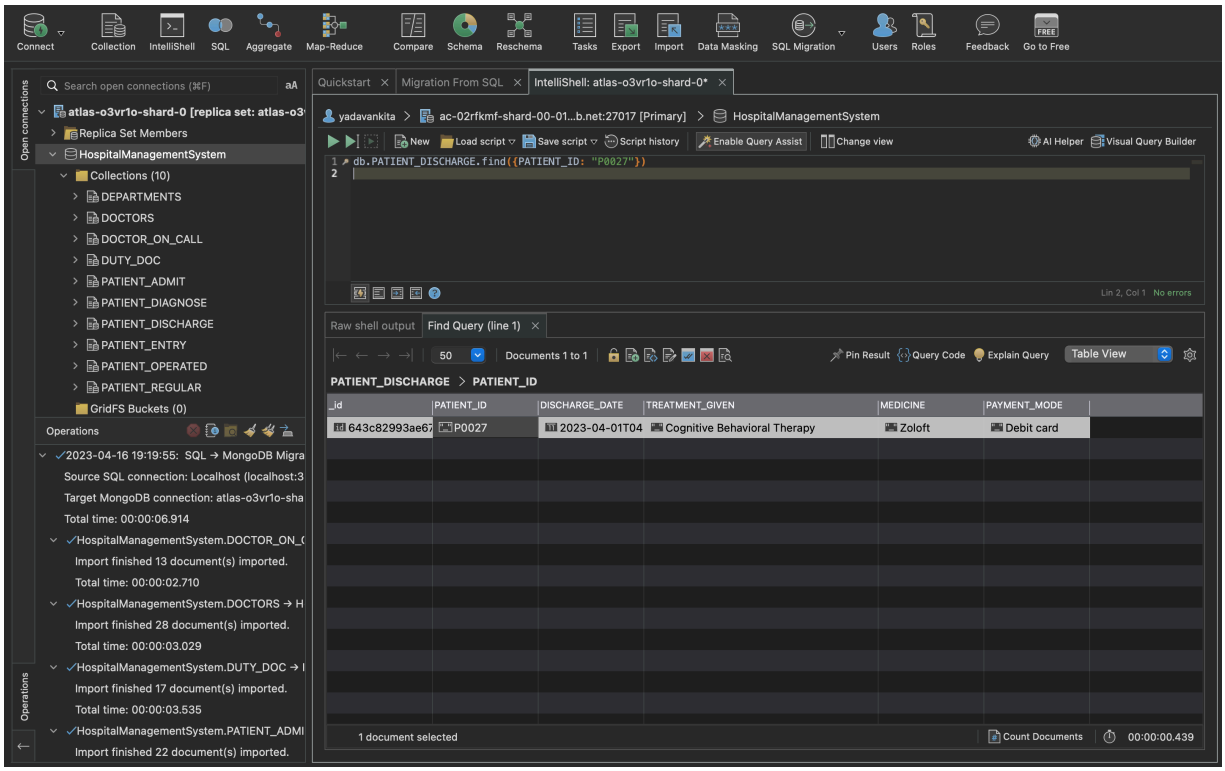
Implementation of NoSQL

Importing database in MongoDB:

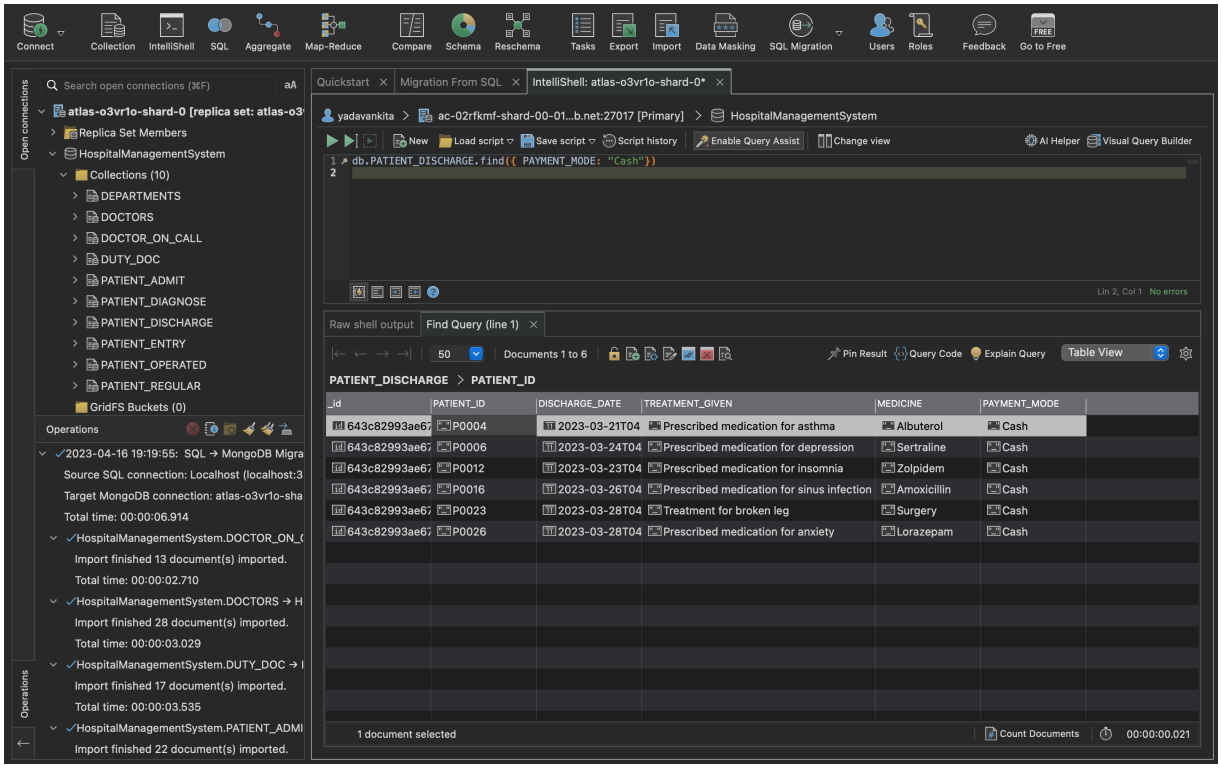
1. Download Studio3T for MongoDB. Load dataset on Cloud0 on MongoDB atlas website.
2. Open Studio3T and add following connections:
   - Open SQLMigration Tab and add your database from SQL Server using host IP, username, password used for creating SQL database. This is Source Connection.
   - Add new connection using URL from mongodb atlas website. This is Target Connection.
3. Click on Add tables to add tables (visible on left pane of the screen) from the database.
4. For executing queries, click on IntelliShell to write queries.

NoSQL Queries:

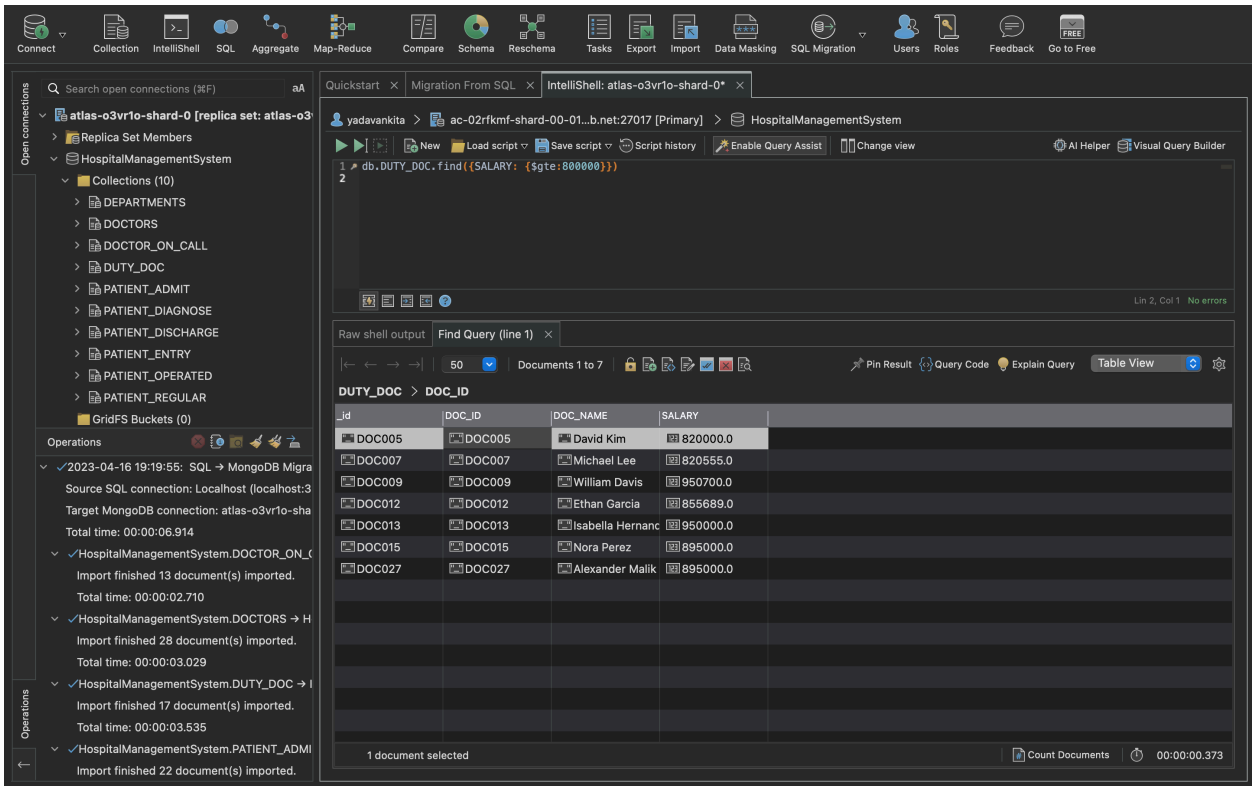
1. Find all diagnoses for a patient:



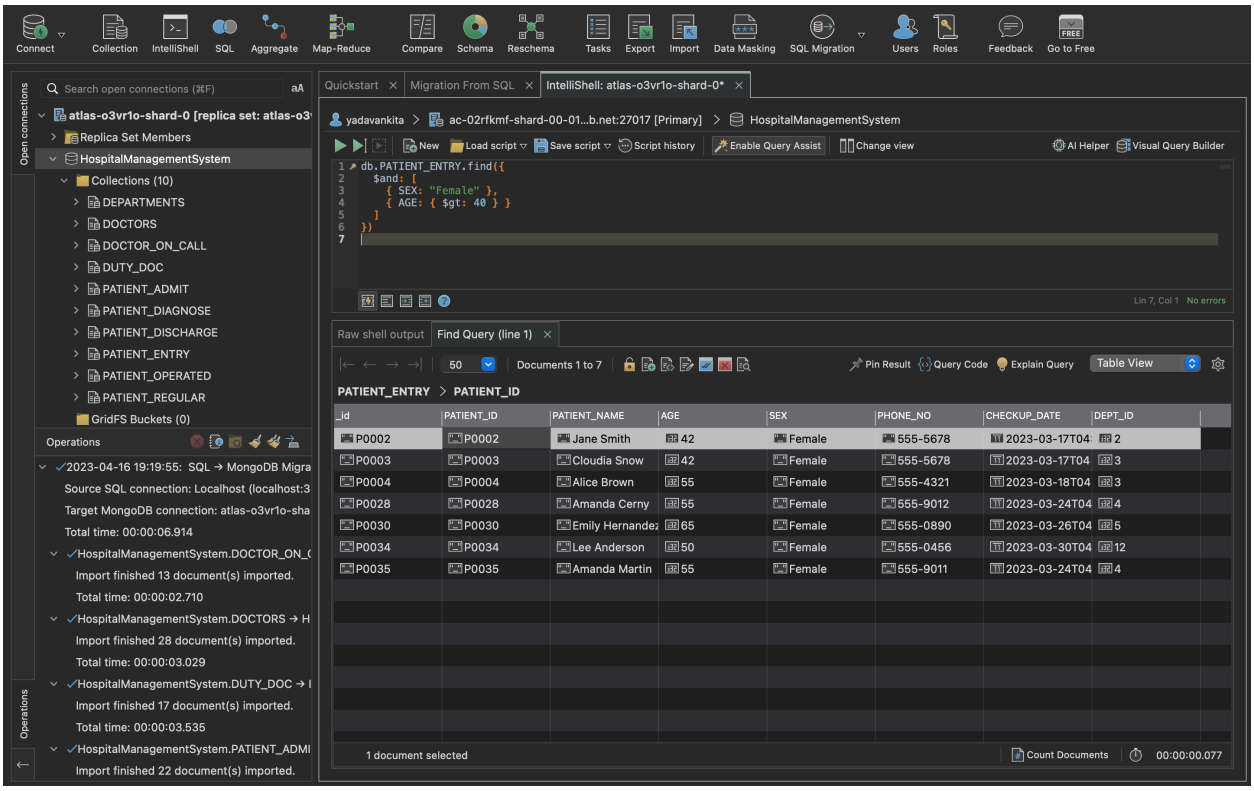
2. Find all patients who have been discharged with the payment mode as Cash:



3. Find the doctors who had salary greater than equal to 800000.



4. Find all the Female records with age greater than 50.



Database Access with Python

The following code was used to access the Hospital Management System Database. By using Python module of mysql.connecter, the database was connected with MySQL.

```
# pip install mysql-connector-python
# pip install gapminder
```

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import gapminder
from gapminder import gapminder
import sqlite3
import mysql.connector
```

```
mydb = mysql.connector.connect(host='localhost',
                               port='3306',
                               user='root',
                               passwd='Abcd1234@',
                               db="HospitalManagementSystem",
                               auth_plugin='mysql_native_password',
                               buffered=True)
```

```
mycursor = mydb.cursor()
```

```
mycursor.execute("SHOW DATABASES")
for x in mycursor:
    print(x)
```

```
('HospitalManagementSystem',)
('information_schema',)
('mysql',)
('performance_schema',)
('sys',)
```

```
mycursor.execute("SHOW TABLES")
```

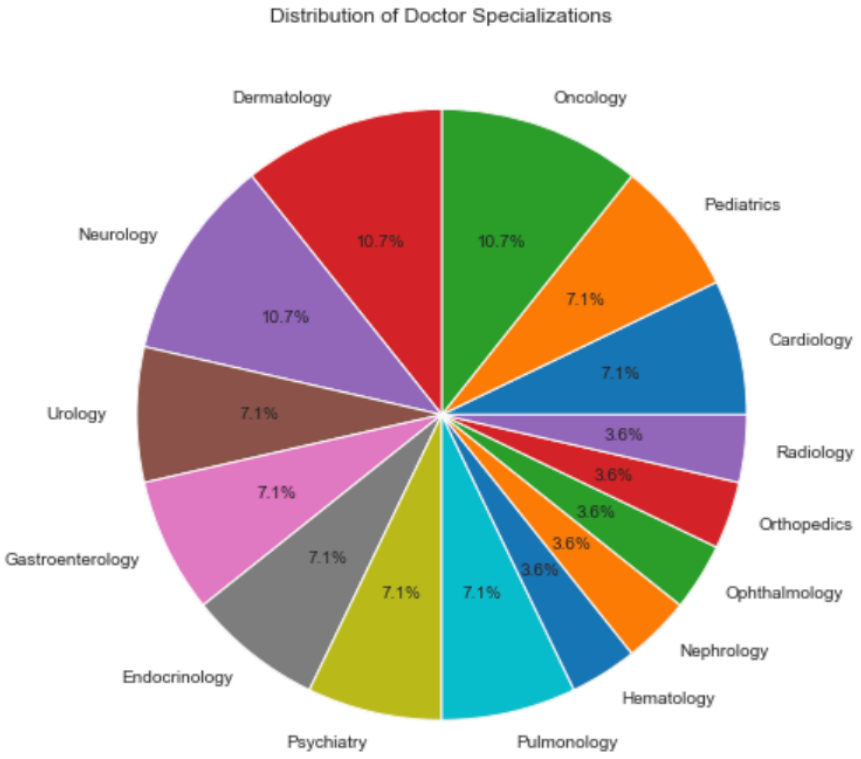
```
for x in mycursor:
    print(x)
```

```
('DEPARTMENTS',)
('DOCTOR_ON_CALL',)
('DOCTORS',)
('DUTY_DOC',)
('PATIENT_ADMIT',)
('PATIENT_DIAGNOSE',)
('PATIENT_DISCHARGE',)
('PATIENT_ENTRY',)
('PATIENT_OPERATED',)
('PATIENT_REGULAR',)
```

Using the MySQL queries, the results were fetched and visualized using python. Following were the Business Questions.

What is the distribution of doctor specializations?

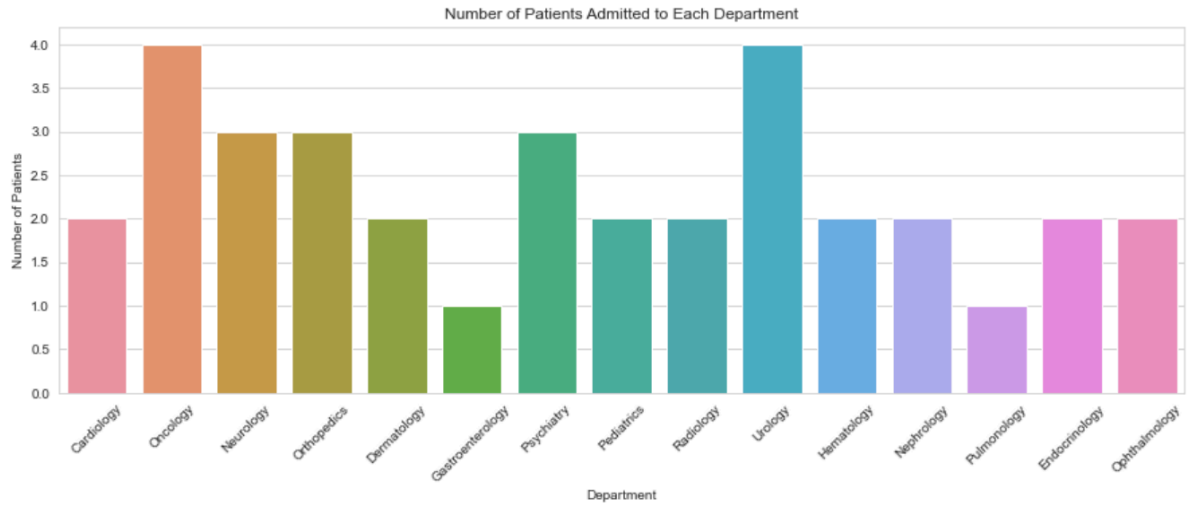
```
mycursor = mydb.cursor()
query1 = "SELECT SPECIALIZATION, COUNT(*) as COUNT FROM DOCTORS GROUP BY SPECIALIZATION"
mycursor.execute(query1)

# Fetch results
results1 = mycursor.fetchall()

# Create a pandas dataframe from the results
df1 = pd.DataFrame(results1, columns=["Specialization", "Count"])

# Create the pie chart using matplotlib
plt.figure(figsize=(10,8))
plt.title("Distribution of Doctor Specializations")
plt.pie(x=df1['Count'], labels=df1['Specialization'], autopct='%1.1f%%')
plt.show()
```



The pie chart above shows the distribution of doctor specializations in the dataset. We can see that the largest percentage of doctors have a specialization in Dermatology, Oncology, Neurology, followed by Urology. Other specializations such as Orthopedics, Radiology, Ophthalmology, and Nephrology have a smaller percentage of doctors.

What is the distribution of patient counts by department?

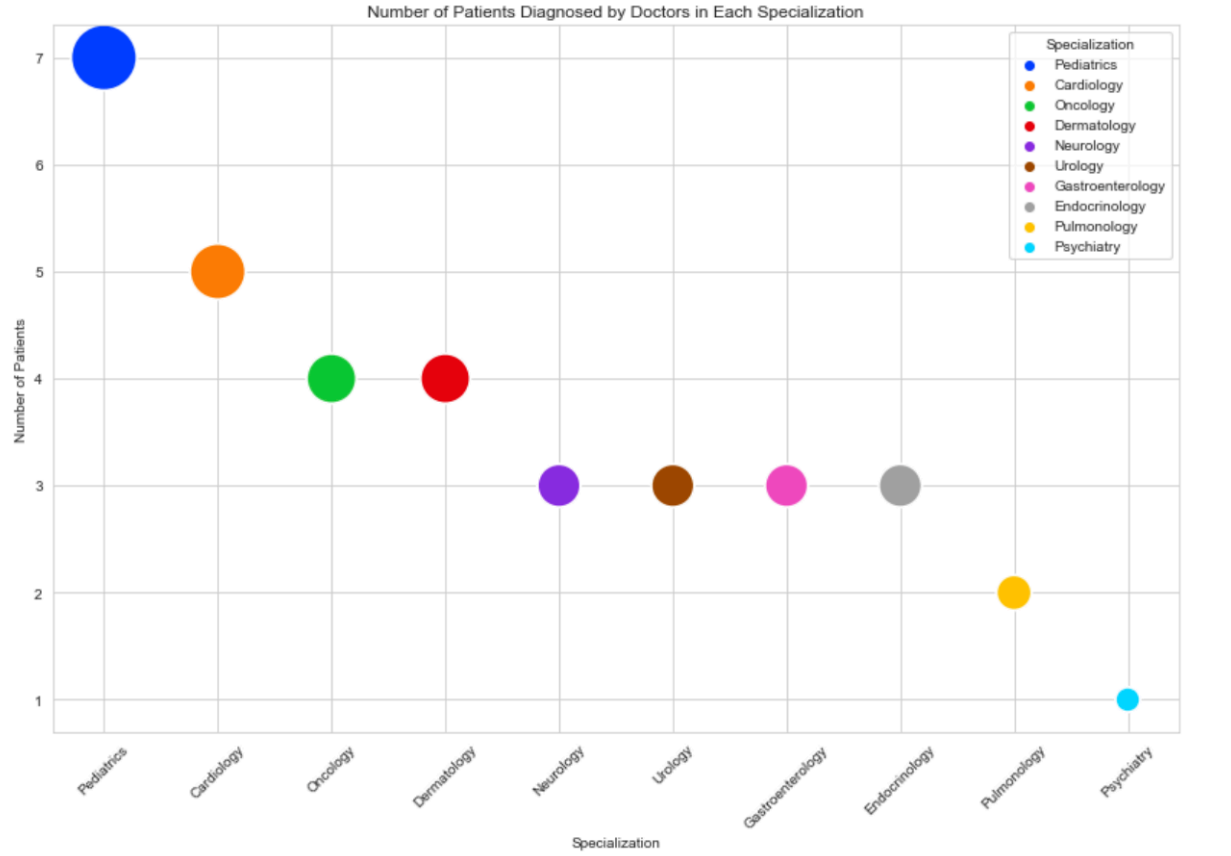
```
## Bar chart of patient count by department:

query2 = """
SELECT DEPT_NAME,
COUNT(*) FROM PATIENT_ENTRY JOIN DEPARTMENTS
ON PATIENT_ENTRY.DEPT_ID = DEPARTMENTS.DEPT_ID
GROUP BY DEPT_NAME;
"""

# Creating dataframe from query result
df2 = pd.read_sql(query2, mydb)

# Creating bar chart using seaborn
sns.set_style('whitegrid')
plt.figure(figsize=(15, 5))
plt.title("Number of Patients Admitted to Each Department")
sns.barplot(x='DEPT_NAME', y='COUNT(*)', data=df2)
plt.xlabel("Department")
plt.ylabel("Number of Patients")
plt.xticks(rotation=45)
plt.show()
```



The bar chart above shows the number of patients admitted to each department in the dataset. We can see that the Oncology and Urology departments has the highest number of patients, The Gastroenterology department has the lowest number of patients in the dataset.

What is the number of patients diagnosed by doctors in each specialization?

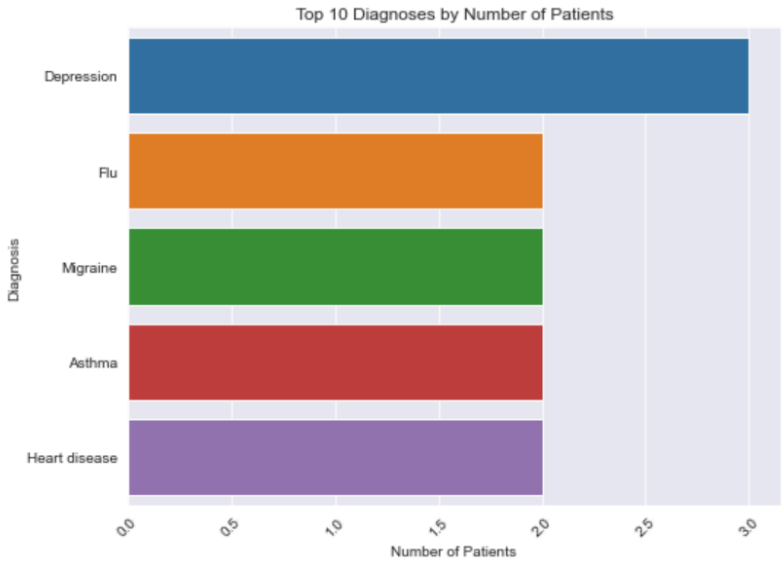
```
# Query to get number of patients diagnosed by doctors in each specialization
query3 = """
SELECT DOCTORS.SPECIALIZATION,
COUNT(*) AS NUM_PATIENTS
FROM PATIENT_DIAGNOSE JOIN DOCTORS
ON PATIENT_DIAGNOSE.DOC_ID = DOCTORS.DOC_ID
GROUP BY DOCTORS.SPECIALIZATION
ORDER BY NUM_PATIENTS DESC
"""

# Executing query
mycursor.execute(query3)

# Fetching data
data3 = mycursor.fetchall()

# Converting data to pandas dataframe
df3 = pd.DataFrame(data3, columns=['Specialization', 'Num_Patients'])

# Creating scatter plot using seaborn
plt.figure(figsize=(14, 9))
sns.scatterplot(data=df3,
                x="Specialization",
                y="Num_Patients",
                s=df3['Num_Patients']*300,
                hue="Specialization",
                palette="bright")
plt.title("Number of Patients Diagnosed by Doctors in Each Specialization")
plt.xlabel("Specialization")
plt.ylabel("Number of Patients")
plt.xticks(rotation=45)

# Displaying plot
plt.show()
```



The number of Pediatrics has the highest number of patients followed by Cardiology and Oncology.

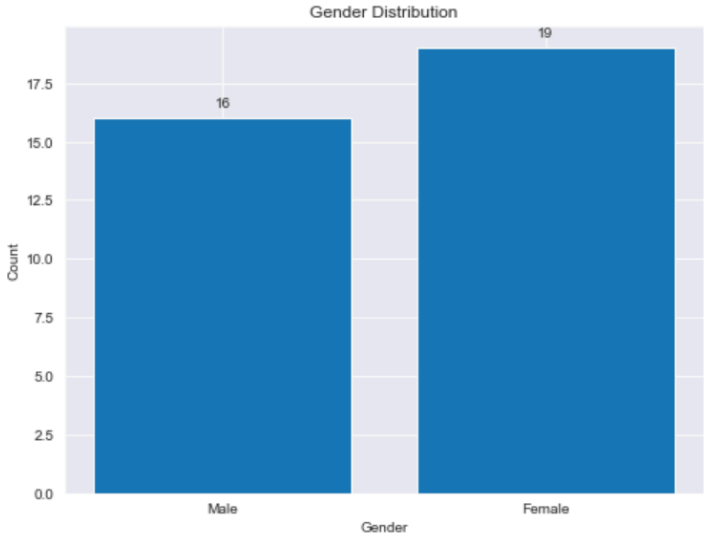
What is the Most Common Diagonosis for the patients?

```
query4 = """
SELECT DIAGNOSIS,
COUNT(*) AS NUM_PATIENTS
FROM PATIENT_DIAGNOSE
GROUP BY DIAGNOSIS
ORDER BY NUM_PATIENTS DESC LIMIT 5
"""
mycursor.execute(query4)

# Fetching data
data4 = mycursor.fetchall()

# Creating a Pandas DataFrame from the data
df4 = pd.DataFrame(data4, columns=['Diagnosis', 'Number of Patients'])

# Creating a barplot using Seaborn
plt.figure(figsize=(8, 6))
sns.set_style('darkgrid')
sns.barplot(x='Number of Patients', y='Diagnosis', data=df4, orient = 'h')

# Adding labels and title
plt.xlabel('Number of Patients')
plt.ylabel('Diagnosis')
plt.title('Top 10 Diagnoses by Number of Patients')
plt.xticks(rotation=45)


# Displaying plot
plt.show()
```



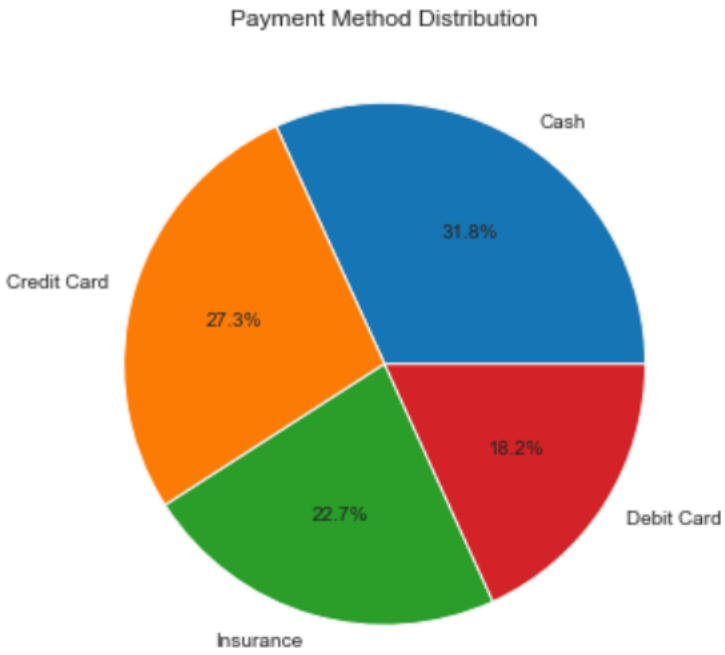
What is the Male to Female Ratio for the patients?

```
a = mycursor.execute('''SELECT sex, COUNT(*) as count FROM
    `HospitalManagementSystem`.PATIENT_ENTRY GROUP BY sex;''')

mycursor.execute(a)
myresult = mycursor.fetchall()
df5 = pd.DataFrame(myresult, columns=['gender', 'count'])

plt.figure(figsize=(8, 6))
plt.bar(df5['gender'], df5['count'])
plt.title('Gender Distribution')
plt.xlabel('Gender')
plt.ylabel('Count')

# Add counts above bars
for i, v in enumerate(df5['count']):
    plt.text(i, v + 1/2, str(v), ha='center')

plt.show()
```



What was the Payment method used by Patients?

```
b = mycursor.execute('''SELECT PATIENT_ADMIT.PAYMENT_MODE, COUNT(*) as payment_count
                    FROM `HospitalManagementSystem`.PATIENT_ADMIT
                    GROUP BY PAYMENT_MODE;''')
mycursor.execute(b)
myresult = mycursor.fetchall()
payment = pd.DataFrame(myresult, columns=['payment_mode', 'payment_count'])
payment
# Create a pie chart
plt.figure(figsize=(8, 6))
plt.pie(payment['payment_count'], labels=payment['payment_mode'], autopct='%1.1f%%')
plt.title('Payment Method Distribution')
plt.show()
```



Summary and Recommendation

This is a Hospital Management system prototype database created in MySQL that can be easily implemented in hospitals to improve their efficiency and usefulness. It comprises of 11 tables that contain detailed information regarding patients, doctors, diagnoses, medications, treatments, and payments. The system provides an organized approach to data management, which can lead to cost savings and enhanced hospital efficiency.

The SQL, NoSQL, and Python Implementations in the project demonstrates system controls paperwork for patients, physicians, and employees while also protecting the safety of the facility. Hospital staff can react to emergencies and get in touch with specialists for their specific specialties with ease because to convenient access to patient information, which includes their current condition, diagnosis, treatment, and medication.

The Hospital Management System Database's multi-implementation approach using MySQL, NoSQL, and Python is an effective way to cater to different use cases and preferences. It is recommended to continue exploring other database management systems to determine which system is best suited for the specific needs of the hospital.

Additionally, implementing more advanced features such as machine learning algorithms can further enhance the system's capabilities and provide more personalized diagnosis and treatment recommendations. The system's security and privacy should be a top priority, and appropriate data protection measures such as access controls and encryption should be implemented to prevent unauthorized access and data breaches.

Furthermore, regular updates and maintenance of the database are necessary to ensure the accuracy and up-to-date information. Lastly, training and support for hospital staff on how to use the system effectively can significantly improve the system's adoption rate and overall effectiveness.

Overall, the Hospital Management System Database's multi-implementation approach and continued improvement can lead to better patient care, streamlined operations, cost savings, and improved hospital efficiency.