MAI172: Advance Database Technologies

Register Number: 2448513

Name: Deshmukh Pratik Bhushanrao

Experiment Number and Name: 5: Demonstration of View database object using SQL Queries.

Date: 01/08/2024 Time: 9.45 to 11.45

Creating Tables:

Creating Patients table:

```
5  • CREATE TABLE Patients (
6        patient_id INT PRIMARY KEY AUTO_INCREMENT,
7        full_name VARCHAR(100) NOT NULL,
8        dob DATE NOT NULL,
9        gender VARCHAR(10),
10       address TEXT,
11       phone VARCHAR(15),
12       insurance_info VARCHAR(100)
13   );
14 • desc Patients;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
patient_id	int	NO	PRI	NULL	auto_increment
full_name	varchar(100)	NO		NULL	
dob	date	NO		NULL	
gender	varchar(10)	YES		NULL	
address	text	YES		NULL	
phone	varchar(15)	YES		NULL	
insurance_info	varchar(100)	YES		NULL	

Inference:

The Patients table is created to store information about patients, including their ID, full name, date of birth, gender, address, phone number, and insurance information. This table serves as the primary repository for patient-related data, enabling efficient management and retrieval of patient details.

Creating Doctors table:

```
16  CREATE TABLE Doctors (
17      doctor_id INT PRIMARY KEY AUTO_INCREMENT,
18      full_name VARCHAR(100) NOT NULL,
19      specialty VARCHAR(50),
20      phone VARCHAR(15),
21      email VARCHAR(100)
22  );
23  desc Doctors;
24
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
doctor_id	int	NO	PRI	NULL	auto_increment
full_name	varchar(100)	NO		NULL	
specialty	varchar(50)	YES		NULL	
phone	varchar(15)	YES		NULL	
email	varchar(100)	YES		NULL	

Inference:

The Doctors table is created to store information about doctors, including their ID, full name, specialty, phone number, and email. This table helps manage doctor-related data, allowing for easy access to information about the medical professionals working in the hospital.

Creating Appointments table:

```
25  CREATE TABLE Appointments (
26      appointment_id INT PRIMARY KEY AUTO_INCREMENT,
27      patient_id INT,
28      doctor_id INT,
29      appointment_date DATE,
30      status VARCHAR(20),
31      FOREIGN KEY (patient_id) REFERENCES Patients(patient_id),
32      FOREIGN KEY (doctor_id) REFERENCES Doctors(doctor_id)
33  );
34  desc Appointments;
35
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
appointment_id	int	NO	PRI	NULL	auto_increment
patient_id	int	YES	MUL	NULL	
doctor_id	int	YES	MUL	NULL	
appointment_date	date	YES		NULL	
status	varchar(20)	YES		NULL	

Inference:

The Appointments table is created to store information about appointments, including the appointment ID, patient ID, doctor ID, appointment date, and status. This table links patients and doctors through appointments, facilitating the scheduling and tracking of patient visits.

Creating Medications table:

```
36 • CREATE TABLE Medications (
37       medication_id INT PRIMARY KEY AUTO_INCREMENT,
38       patient_id INT,
39       doctor_id INT,
40       medication_name VARCHAR(100),
41       dosage VARCHAR(50),
42       start_date DATE,
43       end_date DATE,
44       FOREIGN KEY (patient_id) REFERENCES Patients(patient_id),
45       FOREIGN KEY (doctor_id) REFERENCES Doctors(doctor_id)
46   );
47 •  desc Medications;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
doctor_id	int	NO	PRI	NULL	auto_increment
full_name	varchar(100)	NO		NULL	
specialty	varchar(50)	YES		NULL	
phone	varchar(15)	YES		NULL	
email	varchar(100)	YES		NULL	

Inference:

The Medications table is created to store information about medications prescribed to patients, including the medication ID, patient ID, doctor ID, medication name, dosage, start date, and end date. This table helps track the medications prescribed to patients and the doctors responsible for the prescriptions.

Inserting Records into the created tables:

1. Inserting records in the Patients table:

```
50 •  INSERT INTO Patients (full_name, dob, gender, address, phone, insurance_info) VALUES
51    ('Amit Sharma', '1985-05-15', 'Male', '123 MG Road, Bengaluru', '123-456-7890', 'Insurance A'),
52    ('Priya Singh', '1990-07-20', 'Female', '456 Brigade Road, Bengaluru', '987-654-3210', 'Insurance B'),
53    ('Rahul Verma', '1978-11-30', 'Male', '789 Residency Road, Bengaluru', '555-123-4567', 'Insurance C'),
54    ('Anjali Nair', '1992-03-25', 'Female', '321 Indiranagar, Bengaluru', '555-987-6543', 'Insurance D'),
55    ('Vikram Patel', '1980-08-10', 'Male', '654 Koramangala, Bengaluru', '555-456-7890', 'Insurance E'),
56    ('Sneha Rao', '1987-12-05', 'Female', '987 Whitefield, Bengaluru', '555-654-3210', 'Insurance F'),
57    ('Arjun Desai', '1995-01-15', 'Male', '159 Jayanagar, Bengaluru', '555-789-0123', 'Insurance G'),
58    ('Meera Iyer', '1983-06-20', 'Female', '753 HSR Layout, Bengaluru', '555-321-0987', 'Insurance H'),
59    ('Rohan Gupta', '1975-09-25', 'Male', '852 BTM Layout, Bengaluru', '555-012-3456', 'Insurance I'),
60    ('Kavya Menon', '1998-04-10', 'Female', '951 Marathahalli, Bengaluru', '555-678-9012', 'Insurance J');
61 •  SELECT * FROM Patients;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

patient_id	full_name	dob	gender	address	phone	insurance_info
1	Amit Sharma	1985-05-15	Male	123 MG Road, Bengaluru	123-456-7890	Insurance A
2	Priya Singh	1990-07-20	Female	456 Brigade Road, Bengaluru	987-654-3210	Insurance B
3	Rahul Verma	1978-11-30	Male	789 Residency Road, Bengaluru	555-123-4567	Insurance C
4	Anjali Nair	1992-03-25	Female	321 Indiranagar, Bengaluru	555-987-6543	Insurance D
5	Vikram Patel	1980-08-10	Male	654 Koramangala, Bengaluru	555-456-7890	Insurance E
6	Sneha Rao	1987-12-05	Female	987 Whitefield, Bengaluru	555-654-3210	Insurance F
7	Arjun Desai	1995-01-15	Male	159 Jayanagar, Bengaluru	555-789-0123	Insurance G
8	Meera Iyer	1983-06-20	Female	753 HSR Layout, Bengaluru	555-321-0987	Insurance H
9	Rohan Gupta	1975-09-25	Male	852 BTM Layout, Bengaluru	555-012-3456	Insurance I
10	Kavya Menon	1998-04-10	Female	951 Marathahalli, Bengaluru	555-678-9012	Insurance J

Inference:

Sample records are inserted into the Patients table to populate it with patient information. This data includes various patients with different names, dates of birth, genders, addresses, phone numbers, and insurance information. These records provide a realistic dataset for testing and querying patient-related information

2. Inserting records in the Doctors table:

```
63      -- Insert records into Doctors table
64 •    INSERT INTO Doctors (full_name, specialty, phone, email) VALUES
65      ('Dr. Suresh Kumar', 'Cardiology', '555-123-4567', 'suresh.kumar@hospital.com'),
66      ('Dr. Anjali Mehta', 'Neurology', '555-987-6543', 'anjali.mehta@hospital.com'),
67      ('Dr. Rajesh Reddy', 'Orthopedics', '555-234-5678', 'rajesh.reddy@hospital.com'),
68      ('Dr. Pooja Sharma', 'Pediatrics', '555-876-5432', 'pooja.sharma@hospital.com'),
69      ('Dr. Vikram Singh', 'Dermatology', '555-345-6789', 'vikram.singh@hospital.com');
70 •    SELECT * FROM Doctors;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

doctor_id	full_name	specialty	phone	email
1	Dr. Suresh Kumar	Cardiology	555-123-4567	suresh.kumar@hospital.com
2	Dr. Anjali Mehta	Neurology	555-987-6543	anjali.mehta@hospital.com
3	Dr. Rajesh Reddy	Orthopedics	555-234-5678	rajesh.reddy@hospital.com
4	Dr. Pooja Sharma	Pediatrics	555-876-5432	pooja.sharma@hospital.com
5	Dr. Vikram Singh	Dermatology	555-345-6789	vikram.singh@hospital.com

Inference:

Sample records are inserted into the Doctors table to populate it with doctor information. This data includes various doctors with different names, specialties, phone numbers, and email addresses. These records provide a realistic dataset for testing and querying doctor-related information.

3. Inserting records in the Appointments table:

```
74 •  INSERT INTO Appointments (patient_id, doctor_id, appointment_date, status) VALUES
75    (1, 1, '2024-08-01', 'Scheduled'),
76    (2, 2, '2024-08-02', 'Completed'),
77    (3, 3, '2024-08-03', 'Scheduled'),
78    (4, 4, '2024-08-04', 'Cancelled'),
79    (5, 5, '2024-08-05', 'Scheduled'),
80    (6, 1, '2024-08-06', 'Completed'),
81    (7, 2, '2024-08-07', 'Scheduled'),
82    (8, 3, '2024-08-08', 'Cancelled'),
83    (9, 2, '2024-08-09', 'Scheduled'),
84    (10, 5, '2024-08-10', 'Completed');
85 •  SELECT * FROM Appointments;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

appointment_id	patient_id	doctor_id	appointment_date	status
21	1	1	2024-08-01	Scheduled
22	2	2	2024-08-02	Completed
23	3	3	2024-08-03	Scheduled
24	4	4	2024-08-04	Cancelled
25	5	5	2024-08-05	Scheduled
26	6	1	2024-08-06	Completed
27	7	2	2024-08-07	Scheduled
28	8	3	2024-08-08	Cancelled
29	9	2	2024-08-09	Scheduled
30	10	5	2024-08-10	Completed

Inference:

Sample records are inserted into the Appointments table to populate it with appointment information. This data includes various appointments linking patients and doctors, with different appointment dates and statuses. These records provide a realistic dataset for testing and querying appointment-related information.

4. Inserting records in the Medications table:

```
88 •  INSERT INTO Medications (patient_id, doctor_id, medication_name, dosage, start_date, end_date) VALUES
89    (1, 1, 'Aspirin', '100mg', '2024-08-01', '2024-08-10'),
90    (2, 2, 'Ibuprofen', '200mg', '2024-08-02', '2024-08-12'),
91    (3, 3, 'Paracetamol', '500mg', '2024-08-03', '2024-08-13'),
92    (4, 4, 'Amoxicillin', '250mg', '2024-08-04', '2024-08-14'),
93    (5, 5, 'Metformin', '500mg', '2024-08-05', '2024-08-15'),
94    (6, 1, 'Lisinopril', '10mg', '2024-08-06', '2024-08-16'),
95    (7, 2, 'Atorvastatin', '20mg', '2024-08-07', '2024-08-17'),
96    (8, 3, 'Omeprazole', '40mg', '2024-08-08', '2024-08-18'),
97    (9, 2, 'Simvastatin', '20mg', '2024-08-09', '2024-08-19'),
98    (10, 5, 'Levothyroxine', '50mcg', '2024-08-10', '2024-08-20');
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

medication_id	patient_id	doctor_id	medication_name	dosage	start_date	end_date
31	1	1	Aspirin	100mg	2024-08-01	2024-08-10
32	2	2	Ibuprofen	200mg	2024-08-02	2024-08-12
33	3	3	Paracetamol	500mg	2024-08-03	2024-08-13
34	4	4	Amoxicillin	250mg	2024-08-04	2024-08-14
35	5	5	Metformin	500mg	2024-08-05	2024-08-15
36	6	1	Lisinopril	10mg	2024-08-06	2024-08-16
37	7	2	Atorvastatin	20mg	2024-08-07	2024-08-17
38	8	3	Omeprazole	40mg	2024-08-08	2024-08-18
39	9	2	Simvastatin	20mg	2024-08-09	2024-08-19
40	10	5	Levothyroxine	50mcg	2024-08-10	2024-08-20
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Inference:

Sample records are inserted into the Medications table to populate it with medication information. This data includes various medications prescribed to patients by doctors, with different medication names, dosages, start dates, and end dates. These records provide a realistic dataset for testing and querying medication-related information.

Task 1:

At least 2 views should be created which holds the records from single table and At least 3 views should be created which holds the records from multiple tables.

A. Views from a Single Table:

1. View for Doctors' Information:

   Query:

   CREATE VIEW DoctorsInfo AS SELECT doctor_id, full_name, specialty, phone, email FROM doctors;

   Output:

doctor_id	full_name	specialty	phone	email
1	Dr. Suresh Kumar	Cardiology	555-123-4567	suresh.kumar@hospital.com
2	Dr. Anjali Mehta	Neurology	555-987-6543	anjali.mehta@hospital.com
3	Dr. Rajesh Reddy	Orthopedics	555-234-5678	rajesh.reddy@hospital.com
4	Dr. Pooja Sharma	Pediatrics	555-876-5432	pooja.sharma@hospital.com
5	Dr. Vikram Singh	Dermatology	555-345-6789	vikram.singh@hospital.com
6	Dr. Suresh Kumar	Cardiology	555-123-4567	suresh.kumar@hospital.com
7	Dr. Anjali Mehta	Neurology	555-987-6543	anjali.mehta@hospital.com
8	Dr. Rajesh Reddy	Orthopedics	555-234-5678	rajesh.reddy@hospital.com
9	Dr. Pooja Sharma	Pediatrics	555-876-5432	pooja.sharma@hospital.com
10	Dr. Vikram Singh	Dermatology	555-345-6789	vikram.singh@hospital.com

   Inference:

   This view simplifies access to the essential information about doctors. It can be used to quickly retrieve and display doctor details without exposing all columns of the doctors table.

2. View for Patients' Information:

   Query:

   CREATE VIEW PatientsInfo AS SELECT patient_id, full_name, dob, gender, address, phone, insurance_info FROM patients;

   Output:

patient_id	full_name	dob	gender	address	phone	insurance_info
1	Amit Sharma	1985-05-15	Male	123 MG Road, Bengaluru	123-456-7890	Insurance A
2	Priya Singh	1990-07-20	Female	456 Brigade Road, Bengaluru	987-654-3210	Insurance B
3	Rahul Verma	1978-11-30	Male	789 Residency Road, Bengaluru	555-123-4567	Insurance C
4	Anjali Nair	1992-03-25	Female	321 Indiranagar, Bengaluru	555-987-6543	Insurance D
5	Vikram Patel	1980-08-10	Male	654 Koramangala, Bengaluru	555-456-7890	Insurance E
6	Sneha Rao	1987-12-05	Female	987 Whitefield, Bengaluru	555-654-3210	Insurance F
7	Arjun Desai	1995-01-15	Male	159 Jayanagar, Bengaluru	555-789-0123	Insurance G
8	Meera Iyer	1983-06-20	Female	753 HSR Layout, Bengaluru	555-321-0987	Insurance H
9	Rohan Gupta	1975-09-25	Male	852 BTM Layout, Bengaluru	555-012-3456	Insurance I
10	Kavya Menon	1998-04-10	Female	951 Marathahalli, Bengaluru	555-678-9012	Insurance J

   Inference:

   This view provides a streamlined way to access patient information. It is useful for applications that need to display patient details without requiring access to the entire patients table.

B. Views from Multiple Tables:

1. View for Appointments with Doctor and Patient Details:

Query;

```
CREATE VIEW AppointmentsDetails AS
SELECT a.appointment_id, a.appointment_date, d.full_name AS Doctor_Name,
p.full_name AS Patient_Name
FROM appointments a
JOIN doctors d ON a.doctor_id = d.doctor_id
JOIN patients p ON a.patient_id = p.patient_id;
```

Output:

appointment_id	appointment_date	Doctor_Name	Patient_Name
21	2024-08-01	Dr. Suresh Kumar	Amit Sharma
22	2024-08-02	Dr. Anjali Mehta	Priya Singh
23	2024-08-03	Dr. Rajesh Reddy	Rahul Verma
24	2024-08-04	Dr. Pooja Sharma	Anjali Nair
25	2024-08-05	Dr. Vikram Singh	Vikram Patel
26	2024-08-06	Dr. Suresh Kumar	Sneha Rao
27	2024-08-07	Dr. Anjali Mehta	Arjun Desai
28	2024-08-08	Dr. Rajesh Reddy	Meera Iyer
29	2024-08-09	Dr. Anjali Mehta	Rohan Gupta
30	2024-08-10	Dr. Vikram Singh	Kavya Menon

Inference:

1. This view combines data from the appointments, doctors, and patients tables to provide a comprehensive view of appointments, including the names of the doctors and patients. It is useful for generating reports and displaying appointment details in a user-friendly format.

2. View for Medications Prescribed to Patients

Query:

```
CREATE VIEW MedicationsPrescribed AS
SELECT m.medication_id, m.medication_name, p.full_name AS patient_name,
d.full_name AS Doctor_Name
FROM medications m
JOIN patients p ON m.patient_id = p.patient_id
JOIN doctors d ON m.doctor_id = d.doctor_id;
```

Output:

medication_id	medication_name	patient_name	Doctor_Name
31	Aspirin	Amit Sharma	Dr. Suresh Kumar
32	Ibuprofen	Priya Singh	Dr. Anjali Mehta
33	Paracetamol	Rahul Verma	Dr. Rajesh Reddy
34	Amoxicillin	Anjali Nair	Dr. Pooja Sharma
35	Metformin	Vikram Patel	Dr. Vikram Singh
36	Lisinopril	Sneha Rao	Dr. Suresh Kumar
37	Atorvastatin	Arjun Desai	Dr. Anjali Mehta
38	Omeprazole	Meera Iyer	Dr. Rajesh Reddy
39	Simvastatin	Rohan Gupta	Dr. Anjali Mehta
40	Levothyroxine	Kavya Menon	Dr. Vikram Singh

Inference:

This view provides a detailed look at the medications prescribed to patients, including the names of the prescribing doctors and the patients. It is useful for tracking prescriptions and ensuring that patients receive the correct medications.

3. View for Doctors and Their Specialties with Patient Count

   Query:

```
CREATE VIEW DoctorsSpecialtiesPatientCount AS
SELECT d.doctor_id, d.full_name, d.specialty, COUNT(a.patient_id) AS patient_count
FROM doctors d
LEFT JOIN appointments a ON d.doctor_id = a.doctor_id
GROUP BY d.doctor_id, d.full_name, d.specialty;
```

   Output:

doctor_id	full_name	specialty	patient_count
1	Dr. Suresh Kumar	Cardiology	2
2	Dr. Anjali Mehta	Neurology	3
3	Dr. Rajesh Reddy	Orthopedics	2
4	Dr. Pooja Sharma	Pediatrics	1
5	Dr. Vikram Singh	Dermatology	2
6	Dr. Suresh Kumar	Cardiology	0
7	Dr. Anjali Mehta	Neurology	0
8	Dr. Rajesh Reddy	Orthopedics	0
9	Dr. Pooja Sharma	Pediatrics	0
10	Dr. Vikram Singh	Dermatology	0

   Inference:

This view provides insights into the number of patients each doctor has seen, categorized by their specialty. It is useful for analyzing the workload of doctors and identifying trends in patient visits.

Task 2:

Perform the insert, delete and update operations on the Views which created for performing the different operations and state the inferences.

1. Inserting a new doctor into the DoctorsInfo view:
   Query:
   INSERT INTO DoctorsInfo (doctor_id, full_name, specialty, phone) VALUES (101, 'Dr. Smith', 'Cardiology', '123-456-7890');
   Output:

doctor_id	full_name	specialty	phone	email
1	Dr. Suresh Kumar	Cardiology	555-123-4567	suresh.kumar@hospital.com
2	Dr. Anjali Mehta	Neurology	555-987-6543	anjali.mehta@hospital.com
3	Dr. Rajesh Reddy	Orthopedics	555-234-5678	rajesh.reddy@hospital.com
4	Dr. Pooja Sharma	Pediatrics	555-876-5432	pooja.sharma@hospital.com
5	Dr. Vikram Singh	Dermatology	555-345-6789	vikram.singh@hospital.com
6	Dr. Suresh Kumar	Cardiology	555-123-4567	suresh.kumar@hospital.com
7	Dr. Anjali Mehta	Neurology	555-987-6543	anjali.mehta@hospital.com
8	Dr. Rajesh Reddy	Orthopedics	555-234-5678	rajesh.reddy@hospital.com
9	Dr. Pooja Sharma	Pediatrics	555-876-5432	pooja.sharma@hospital.com
10	Dr. Vikram Singh	Dermatology	555-345-6789	vikram.singh@hospital.com
101	Dr. Smith	Cardiology	123-456-7890	NULL

   Inference:
   If the view is updatable, this will insert a new record into the doctors table. If not, the database will throw an error indicating that the view is not updatable.

2. Deleting a doctor from the DoctorsInfo view:
   Output:
   DELETE FROM DoctorsInfo WHERE doctor_id = 101;
   Output:

doctor_id	full_name	specialty	phone	email
1	Dr. Suresh Kumar	Cardiology	555-123-4567	suresh.kumar@hospital.com
2	Dr. Anjali Mehta	Neurology	555-987-6543	anjali.mehta@hospital.com
3	Dr. Rajesh Reddy	Orthopedics	555-234-5678	rajesh.reddy@hospital.com
4	Dr. Pooja Sharma	Pediatrics	555-876-5432	pooja.sharma@hospital.com
5	Dr. Vikram Singh	Dermatology	555-345-6789	vikram.singh@hospital.com
6	Dr. Suresh Kumar	Cardiology	555-123-4567	suresh.kumar@hospital.com
7	Dr. Anjali Mehta	Neurology	555-987-6543	anjali.mehta@hospital.com
8	Dr. Rajesh Reddy	Orthopedics	555-234-5678	rajesh.reddy@hospital.com
9	Dr. Pooja Sharma	Pediatrics	555-876-5432	pooja.sharma@hospital.com
10	Dr. Vikram Singh	Dermatology	555-345-6789	vikram.singh@hospital.com

   Inference:
   If the view is updatable, this will delete the corresponding record from the doctors table. If not, an error will be thrown.

3. Updating a doctor's phone number in the DoctorsInfo view:
   Query:
   UPDATE DoctorsInfo SET phone= '9876543210' WHERE doctor_id = 10;
   Output:

doctor_id	full_name	specialty	phone	email
1	Dr. Suresh Kumar	Cardiology	555-123-4567	suresh.kumar@hospital.com
2	Dr. Anjali Mehta	Neurology	555-987-6543	anjali.mehta@hospital.com
3	Dr. Rajesh Reddy	Orthopedics	555-234-5678	rajesh.reddy@hospital.com
4	Dr. Pooja Sharma	Pediatrics	555-876-5432	pooja.sharma@hospital.com
5	Dr. Vikram Singh	Dermatology	555-345-6789	vikram.singh@hospital.com
6	Dr. Suresh Kumar	Cardiology	555-123-4567	suresh.kumar@hospital.com
7	Dr. Anjali Mehta	Neurology	555-987-6543	anjali.mehta@hospital.com
8	Dr. Rajesh Reddy	Orthopedics	555-234-5678	rajesh.reddy@hospital.com
9	Dr. Pooja Sharma	Pediatrics	555-876-5432	pooja.sharma@hospital.com
10	Dr. Vikram Singh	Dermatology	9876543210	vikram.singh@hospital.com

Inference:

If the view is updatable, this will update the phone number in the doctors table. If not, an error will be thrown.