Royal University of Bhutan

འབྲུག་རྒྱལ་འཛིན་གཙུག་ལག་སློབ་སྡེ།།

College of Science and Technology
Rinchending: Bhutan

DBS101
Database Systems Fundamentals
SS(2024)

Practical{3} Report

Submitted By;

Student Name : Tashi Penjor

Enrollment No.: 02230306

Programme : BESWE

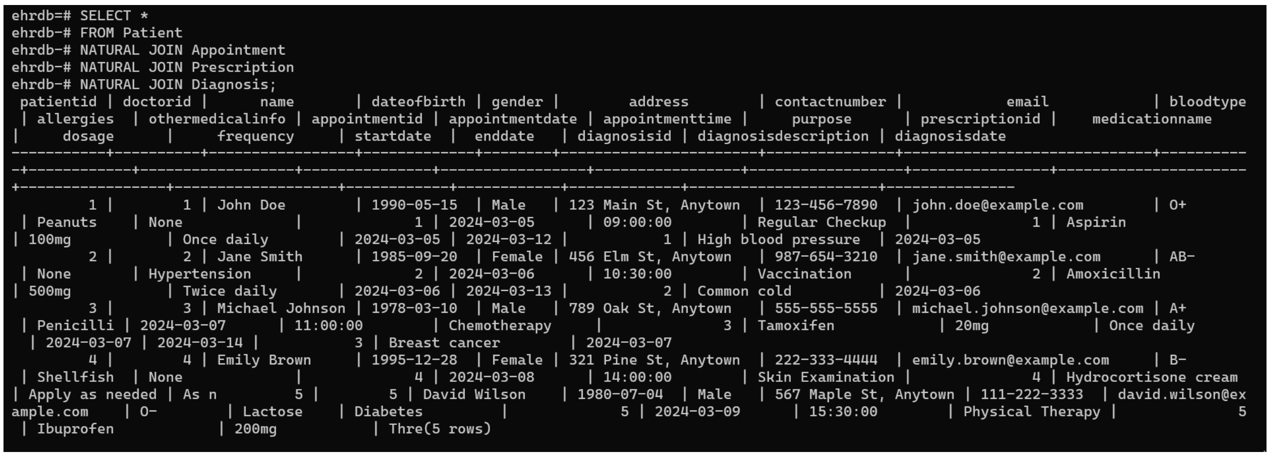
Date : 12/03/2024

Table of content

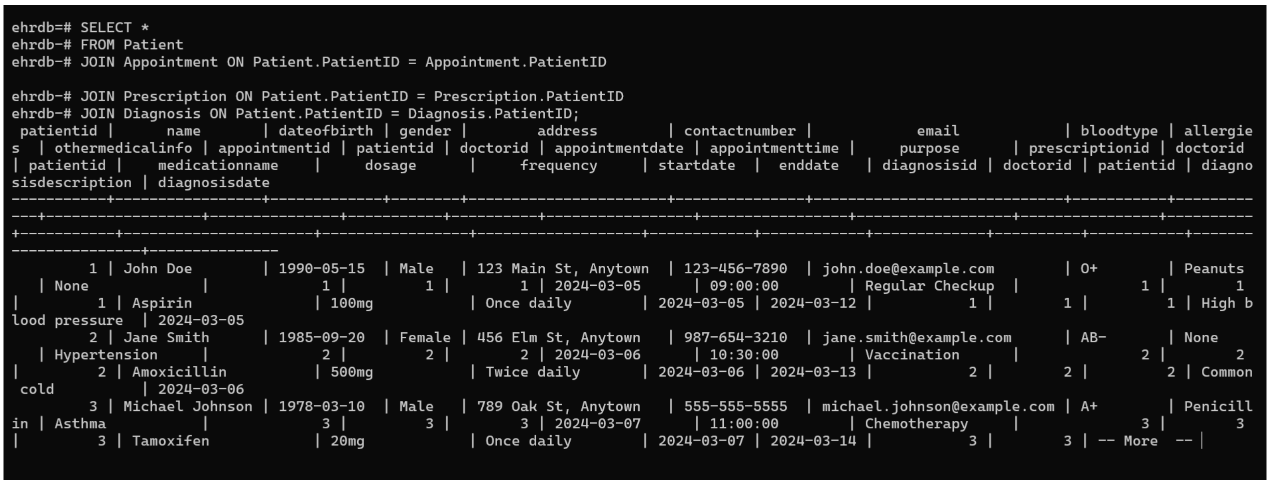
SL.No	Topics	Page number
1	Guided session	2 - 6
2	Conclusion	7 - 8
3	Practical 3	9 - 11
4	Conclusion	12

Topic : Guided Session

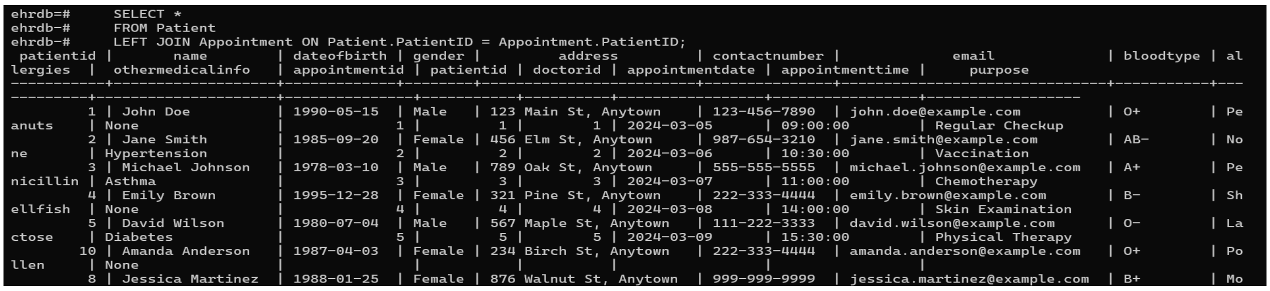
Task 1 : joined the common columns in the table.


```
ehrdb=# SELECT *
ehrdb-# FROM Patient
ehrdb-# NATURAL JOIN Appointment
ehrdb-# NATURAL JOIN Prescription
ehrdb-# NATURAL JOIN Diagnosis;
 patientid | doctorid |       name        | dateofbirth | gender |        address         | contactnumber |            email             | bloodtype
 | allergies  | othermedicalinfo | appointmentid | appointmentdate | appointmenttime |     purpose      | prescriptionid |    medicationname
|     dosage      |      frequency      | startdate  |  enddate   | diagnosisid | diagnosisdescription | diagnosisdate
-----------+----------+-------------------+-------------+--------+------------------------+---------------+------------------------------+----------
-+-----------+------------------+---------------+-----------------+-----------------+------------------+----------------+----------------------
+-----------------+---------------------+------------+------------+-------------+----------------------+---------------
         1 |        1 | John Doe          | 1990-05-15  | Male   | 123 Main St, Anytown   | 123-456-7890  | john.doe@example.com         | O+
 | Peanuts    | None             |             1 | 2024-03-05      | 09:00:00        | Regular Checkup  |              1 | Aspirin
| 100mg           | Once daily          | 2024-03-05 | 2024-03-12 |           1 | High blood pressure  | 2024-03-05
         2 |        2 | Jane Smith        | 1985-09-20  | Female | 456 Elm St, Anytown    | 987-654-3210  | jane.smith@example.com       | AB-
 | None       | Hypertension     |             2 | 2024-03-06      | 10:30:00        | Vaccination      |              2 | Amoxicillin
| 500mg           | Twice daily         | 2024-03-06 | 2024-03-13 |           2 | Common cold          | 2024-03-06
         3 |        3 | Michael Johnson | 1978-03-10  | Male   | 789 Oak St, Anytown    | 555-555-5555  | michael.johnson@example.com | A+
 | Penicilli | 2024-03-07       | 11:00:00        | Chemotherapy     |              3 | Tamoxifen          | 20mg               | Once daily
  | 2024-03-07 | 2024-03-14 |           3 | Breast cancer        | 2024-03-07
         4 |        4 | Emily Brown       | 1995-12-28  | Female | 321 Pine St, Anytown   | 222-333-4444  | emily.brown@example.com      | B-
 | Shellfish  | None             |             4 | 2024-03-08      | 14:00:00        | Skin Examination |              4 | Hydrocortisone cream
| Apply as needed | As n          5 |        5 | David Wilson      | 1980-07-04  | Male   | 567 Maple St, Anytown | 111-222-3333  | david.wilson@ex
ample.com     | O-          | Lactose     | Diabetes            |               5 | 2024-03-09      | 15:30:00        | Physical Therapy |              5
 | Ibuprofen            | 200mg            | Thre(5 rows)
```

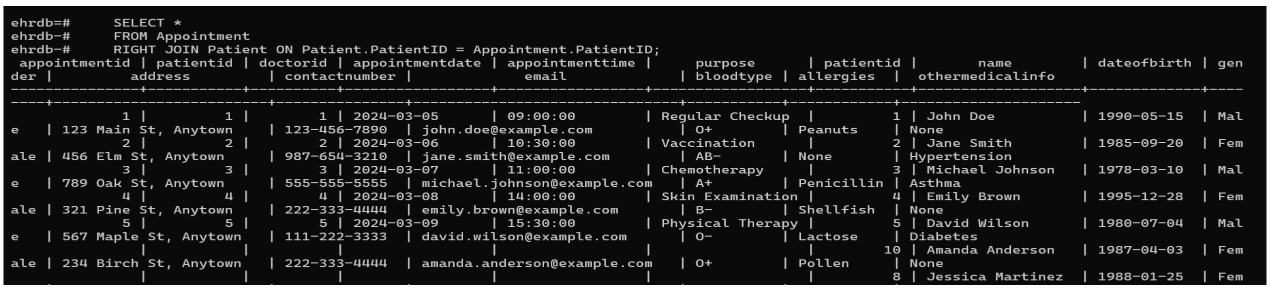
Task 2 : join the column with the same name.


```
ehrdb=# SELECT *
ehrdb-# FROM Patient
ehrdb-# JOIN Appointment ON Patient.PatientID = Appointment.PatientID

ehrdb-# JOIN Prescription ON Patient.PatientID = Prescription.PatientID
ehrdb-# JOIN Diagnosis ON Patient.PatientID = Diagnosis.PatientID;
 patientid |       name        | dateofbirth | gender |        address         | contactnumber |            email             | bloodtype | allergie
s  | othermedicalinfo | appointmentid | patientid | doctorid | appointmentdate | appointmenttime |     purpose      | prescriptionid | doctorid
| patientid |    medicationname    |      dosage       |      frequency      | startdate  |  enddate   | diagnosisid | doctorid | patientid | diagno
sisdescription | diagnosisdate
-----------+-------------------+-------------+--------+------------------------+---------------+------------------------------+-----------+---------
---+------------------+---------------+-----------+----------+-----------------+-----------------+------------------+----------------+----------
+-----------+----------------------+-------------------+---------------------+------------+------------+-------------+----------+-----------+-------
---------------+---------------
         1 | John Doe          | 1990-05-15  | Male   | 123 Main St, Anytown   | 123-456-7890  | john.doe@example.com         | O+        | Peanuts
   | None             |             1 |         1 |        1 | 2024-03-05      | 09:00:00        | Regular Checkup  |              1 |        1
|         1 | Aspirin              | 100mg             | Once daily          | 2024-03-05 | 2024-03-12 |           1 |        1 |         1 | High b
lood pressure  | 2024-03-05
         2 | Jane Smith        | 1985-09-20  | Female | 456 Elm St, Anytown    | 987-654-3210  | jane.smith@example.com       | AB-       | None
   | Hypertension     |             2 |         2 |        2 | 2024-03-06      | 10:30:00        | Vaccination      |              2 |        2
|         2 | Amoxicillin          | 500mg             | Twice daily         | 2024-03-06 | 2024-03-13 |           2 |        2 |         2 | Common
 cold          | 2024-03-06
         3 | Michael Johnson | 1978-03-10  | Male   | 789 Oak St, Anytown    | 555-555-5555  | michael.johnson@example.com | A+        | Penicill
in | Asthma           |             3 |         3 |        3 | 2024-03-07      | 11:00:00        | Chemotherapy     |              3 |        3
|         3 | Tamoxifen            | 20mg              | Once daily          | 2024-03-07 | 2024-03-14 |           3 |        3 | -- More  --
```

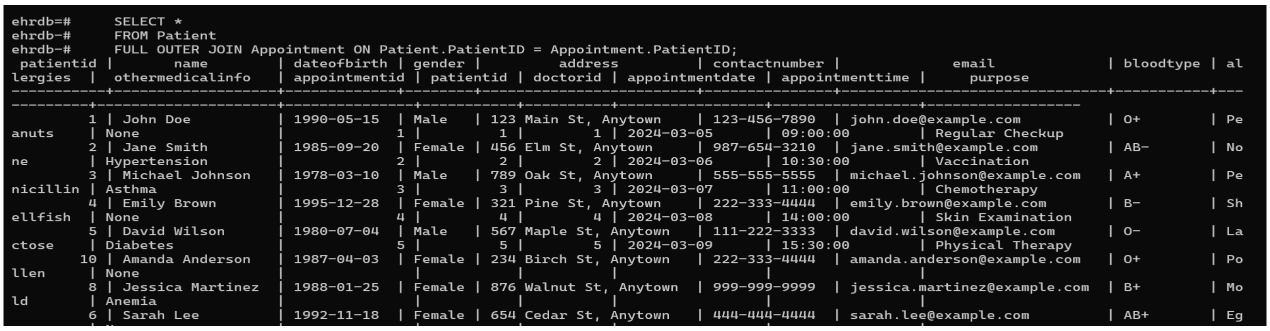
Task 3 : Left outer join returns all records from the left table and the matched records from the right table. Right outer join returns all records from the right table and the matched records from the left table. A full outer join returns all records when there is a match in either the left or right table. If there is no match, NULL values are returned for the columns of the table that lack a match.



```
ehrdb=#     SELECT *
ehrdb-#     FROM Patient
ehrdb-#     LEFT JOIN Appointment ON Patient.PatientID = Appointment.PatientID;
 patientid |        name         | dateofbirth | gender |          address          | contactnumber |             email              | bloodtype | al
lergies  |  othermedicalinfo    | appointmentid | patientid | doctorid | appointmentdate | appointmenttime |     purpose
-----------+---------------------+-------------+--------+---------------------------+---------------+--------------------------------+-----------+---
---------+---------------------+---------------+-----------+----------+-----------------+-----------------+-------------------
         1 | John Doe            | 1990-05-15  | Male   | 123 Main St, Anytown      | 123-456-7890  | john.doe@example.com           | O+        | Pe
anuts    | None                |             1 |         1 |        1 | 2024-03-05      | 09:00:00        | Regular Checkup
         2 | Jane Smith          | 1985-09-20  | Female | 456 Elm St, Anytown       | 987-654-3210  | jane.smith@example.com         | AB-       | No
ne       | Hypertension        |             2 |         2 |        2 | 2024-03-06      | 10:30:00        | Vaccination
         3 | Michael Johnson     | 1978-03-10  | Male   | 789 Oak St, Anytown       | 555-555-5555  | michael.johnson@example.com    | A+        | Pe
nicillin | Asthma              |             3 |         3 |        3 | 2024-03-07      | 11:00:00        | Chemotherapy
         4 | Emily Brown         | 1995-12-28  | Female | 321 Pine St, Anytown      | 222-333-4444  | emily.brown@example.com        | B-        | Sh
ellfish  | None                |             4 |         4 |        4 | 2024-03-08      | 14:00:00        | Skin Examination
         5 | David Wilson        | 1980-07-04  | Male   | 567 Maple St, Anytown     | 111-222-3333  | david.wilson@example.com       | O-        | La
ctose    | Diabetes            |             5 |         5 |        5 | 2024-03-09      | 15:30:00        | Physical Therapy
        10 | Amanda Anderson     | 1987-04-03  | Female | 234 Birch St, Anytown     | 222-333-4444  | amanda.anderson@example.com    | O+        | Po
llen     | None                |               |           |          |                 |                 |
         8 | Jessica Martinez    | 1988-01-25  | Female | 876 Walnut St, Anytown    | 999-999-9999  | jessica.martinez@example.com   | B+        | Mo
```

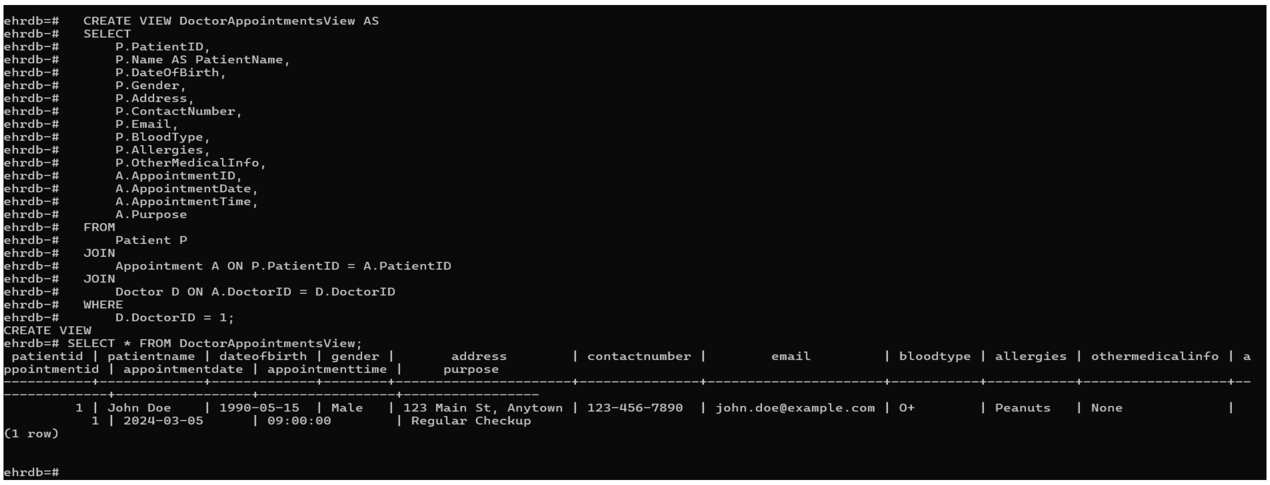


```
ehrdb=#     SELECT *
ehrdb-#     FROM Appointment
ehrdb-#     RIGHT JOIN Patient ON Patient.PatientID = Appointment.PatientID;
 appointmentid | patientid | doctorid | appointmentdate | appointmenttime |     purpose      | patientid |        name         | dateofbirth | gen
der |          address          | contactnumber |             email              | bloodtype | allergies |  othermedicalinfo
---------------+-----------+----------+-----------------+-----------------+------------------+-----------+---------------------+-------------+----
----+---------------------------+---------------+--------------------------------+-----------+-----------+---------------------
             1 |         1 |        1 | 2024-03-05      | 09:00:00        | Regular Checkup  |         1 | John Doe            | 1990-05-15  | Mal
e   | 123 Main St, Anytown      | 123-456-7890  | john.doe@example.com           | O+        | Peanuts   | None
             2 |         2 |        2 | 2024-03-06      | 10:30:00        | Vaccination      |         2 | Jane Smith          | 1985-09-20  | Fem
ale | 456 Elm St, Anytown       | 987-654-3210  | jane.smith@example.com         | AB-       | None      | Hypertension
             3 |         3 |        3 | 2024-03-07      | 11:00:00        | Chemotherapy     |         3 | Michael Johnson     | 1978-03-10  | Mal
e   | 789 Oak St, Anytown       | 555-555-5555  | michael.johnson@example.com    | A+        | Penicillin | Asthma
             4 |         4 |        4 | 2024-03-08      | 14:00:00        | Skin Examination |         4 | Emily Brown         | 1995-12-28  | Fem
ale | 321 Pine St, Anytown      | 222-333-4444  | emily.brown@example.com        | B-        | Shellfish | None
             5 |         5 |        5 | 2024-03-09      | 15:30:00        | Physical Therapy |         5 | David Wilson        | 1980-07-04  | Mal
e   | 567 Maple St, Anytown     | 111-222-3333  | david.wilson@example.com       | O-        | Lactose   | Diabetes
               |           |          |                 |                 |                  |        10 | Amanda Anderson     | 1987-04-03  | Fem
ale | 234 Birch St, Anytown     | 222-333-4444  | amanda.anderson@example.com    | O+        | Pollen    | None
               |           |          |                 |                 |                  |         8 | Jessica Martinez    | 1988-01-25  | Fem
```

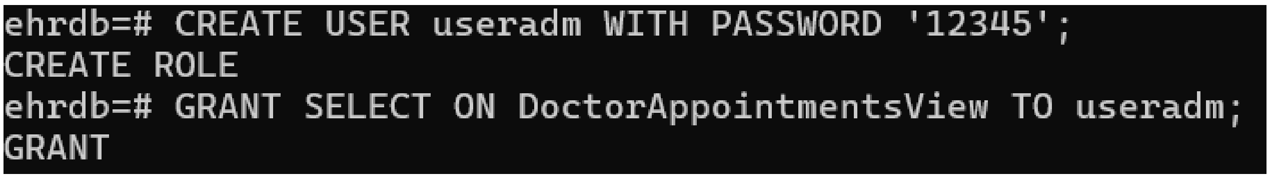


```
ehrdb=#     SELECT *
ehrdb-#     FROM Patient
ehrdb-#     FULL OUTER JOIN Appointment ON Patient.PatientID = Appointment.PatientID;
 patientid |        name         | dateofbirth | gender |          address          | contactnumber |             email              | bloodtype | al
lergies  |  othermedicalinfo    | appointmentid | patientid | doctorid | appointmentdate | appointmenttime |     purpose
-----------+---------------------+-------------+--------+---------------------------+---------------+--------------------------------+-----------+---
---------+---------------------+---------------+-----------+----------+-----------------+-----------------+-------------------
         1 | John Doe            | 1990-05-15  | Male   | 123 Main St, Anytown      | 123-456-7890  | john.doe@example.com           | O+        | Pe
anuts    | None                |             1 |         1 |        1 | 2024-03-05      | 09:00:00        | Regular Checkup
         2 | Jane Smith          | 1985-09-20  | Female | 456 Elm St, Anytown       | 987-654-3210  | jane.smith@example.com         | AB-       | No
ne       | Hypertension        |             2 |         2 |        2 | 2024-03-06      | 10:30:00        | Vaccination
         3 | Michael Johnson     | 1978-03-10  | Male   | 789 Oak St, Anytown       | 555-555-5555  | michael.johnson@example.com    | A+        | Pe
nicillin | Asthma              |             3 |         3 |        3 | 2024-03-07      | 11:00:00        | Chemotherapy
         4 | Emily Brown         | 1995-12-28  | Female | 321 Pine St, Anytown      | 222-333-4444  | emily.brown@example.com        | B-        | Sh
ellfish  | None                |             4 |         4 |        4 | 2024-03-08      | 14:00:00        | Skin Examination
         5 | David Wilson        | 1980-07-04  | Male   | 567 Maple St, Anytown     | 111-222-3333  | david.wilson@example.com       | O-        | La
ctose    | Diabetes            |             5 |         5 |        5 | 2024-03-09      | 15:30:00        | Physical Therapy
        10 | Amanda Anderson     | 1987-04-03  | Female | 234 Birch St, Anytown     | 222-333-4444  | amanda.anderson@example.com    | O+        | Po
llen     | None                |               |           |          |                 |                 |
         8 | Jessica Martinez    | 1988-01-25  | Female | 876 Walnut St, Anytown    | 999-999-9999  | jessica.martinez@example.com   | B+        | Mo
ld       | Anemia              |               |           |          |                 |                 |
         6 | Sarah Lee           | 1992-11-18  | Female | 654 Cedar St, Anytown     | 444-444-4444  | sarah.lee@example.com          | AB+       | Eg
```

TASK 4 : Created a view that can show the patients detail and their upcoming appointment. Found that Jhon Doe has an appointment for regular check up. Created useradm with password ‘12345’ and granted access to the doctor's appointment.

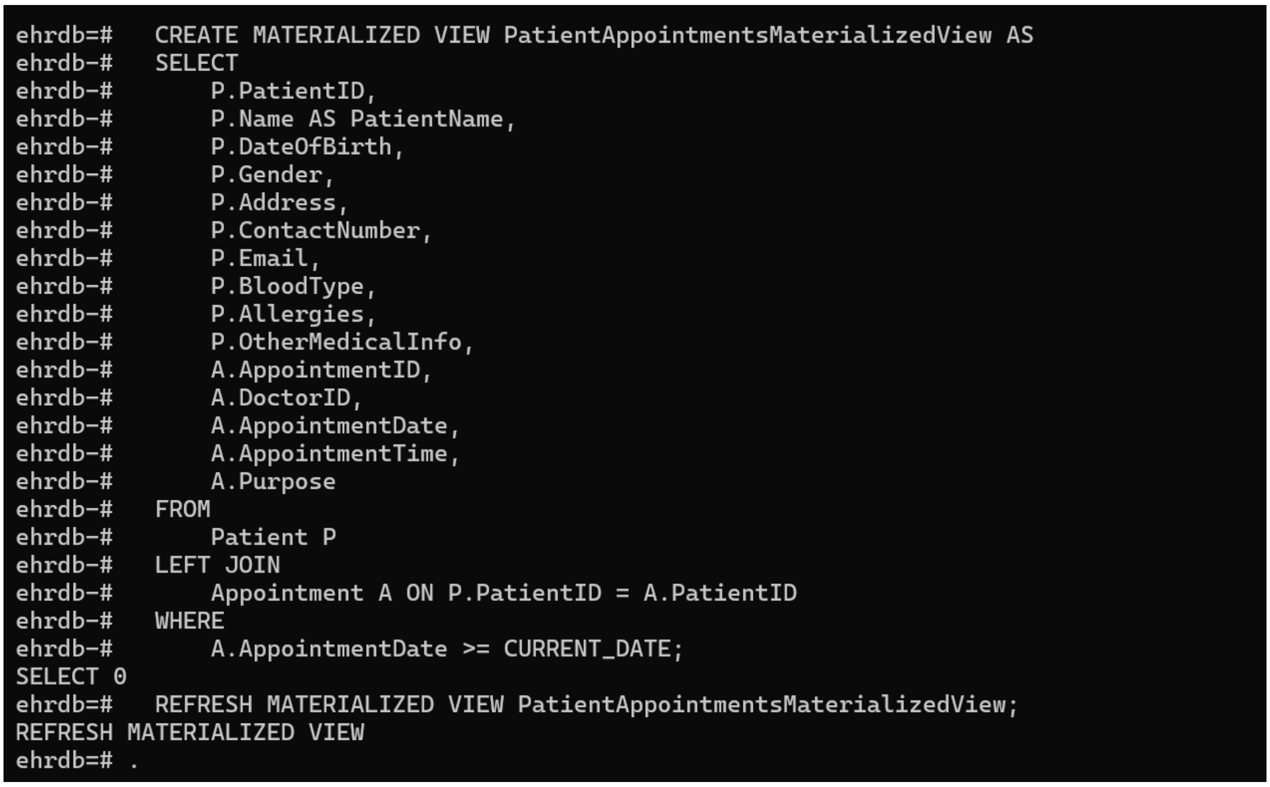


```
ehrdb=#   CREATE VIEW DoctorAppointmentsView AS
ehrdb-#   SELECT
ehrdb-#       P.PatientID,
ehrdb-#       P.Name AS PatientName,
ehrdb-#       P.DateOfBirth,
ehrdb-#       P.Gender,
ehrdb-#       P.Address,
ehrdb-#       P.ContactNumber,
ehrdb-#       P.Email,
ehrdb-#       P.BloodType,
ehrdb-#       P.Allergies,
ehrdb-#       P.OtherMedicalInfo,
ehrdb-#       A.AppointmentID,
ehrdb-#       A.AppointmentDate,
ehrdb-#       A.AppointmentTime,
ehrdb-#       A.Purpose
ehrdb-#   FROM
ehrdb-#       Patient P
ehrdb-#   JOIN
ehrdb-#       Appointment A ON P.PatientID = A.PatientID
ehrdb-#   JOIN
ehrdb-#       Doctor D ON A.DoctorID = D.DoctorID
ehrdb-#   WHERE
ehrdb-#       D.DoctorID = 1;
CREATE VIEW
ehrdb=# SELECT * FROM DoctorAppointmentsView;
 patientid | patientname | dateofbirth | gender |       address        | contactnumber |        email         | bloodtype | allergies | othermedicalinfo | a
ppointmentid | appointmentdate | appointmenttime |     purpose
-----------+-------------+-------------+--------+----------------------+---------------+----------------------+-----------+-----------+------------------+--
-------------+-----------------+-----------------+-----------------
         1 | John Doe    | 1990-05-15  | Male   | 123 Main St, Anytown | 123-456-7890  | john.doe@example.com | O+        | Peanuts   | None             |
             1 | 2024-03-05      | 09:00:00        | Regular Checkup
(1 row)


ehrdb=#
```



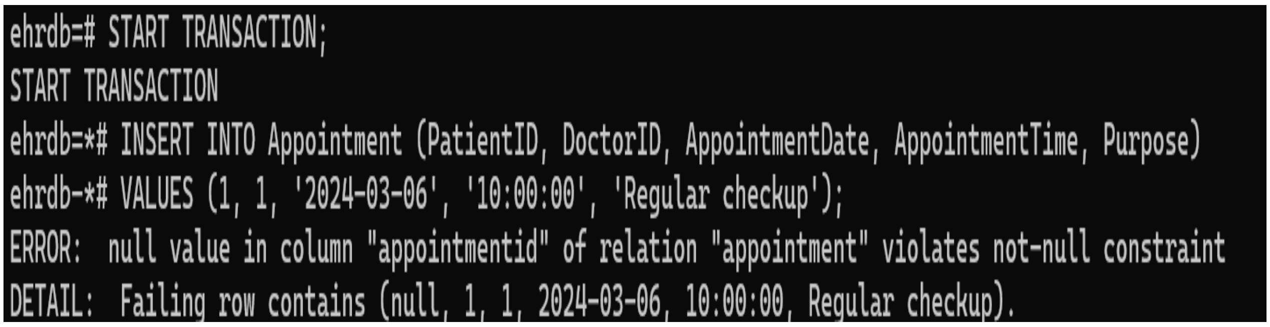
```
ehrdb=# CREATE USER useradm WITH PASSWORD '12345';
CREATE ROLE
ehrdb=# GRANT SELECT ON DoctorAppointmentsView TO useradm;
GRANT
```

Moreover I have created the materialized view to store the result of the query

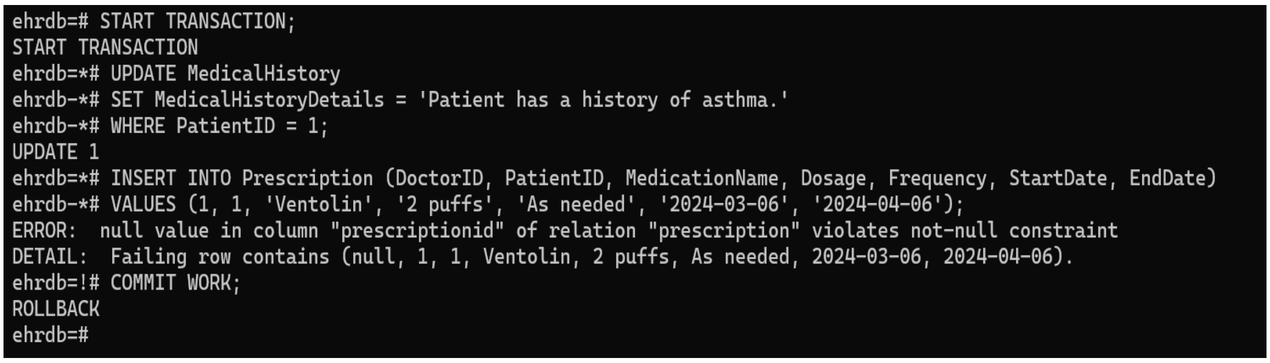


```
ehrdb=#   CREATE MATERIALIZED VIEW PatientAppointmentsMaterializedView AS
ehrdb-#   SELECT
ehrdb-#       P.PatientID,
ehrdb-#       P.Name AS PatientName,
ehrdb-#       P.DateOfBirth,
ehrdb-#       P.Gender,
ehrdb-#       P.Address,
ehrdb-#       P.ContactNumber,
ehrdb-#       P.Email,
ehrdb-#       P.BloodType,
ehrdb-#       P.Allergies,
ehrdb-#       P.OtherMedicalInfo,
ehrdb-#       A.AppointmentID,
ehrdb-#       A.DoctorID,
ehrdb-#       A.AppointmentDate,
ehrdb-#       A.AppointmentTime,
ehrdb-#       A.Purpose
ehrdb-#   FROM
ehrdb-#       Patient P
ehrdb-#   LEFT JOIN
ehrdb-#       Appointment A ON P.PatientID = A.PatientID
ehrdb-#   WHERE
ehrdb-#       A.AppointmentDate >= CURRENT_DATE;
SELECT 0
ehrdb=#   REFRESH MATERIALIZED VIEW PatientAppointmentsMaterializedView;
REFRESH MATERIALIZED VIEW
ehrdb=# .
```

TASK 5 : to see the update query statement, i did a transaction but couldn’t commit the work. So I rolled back the work.

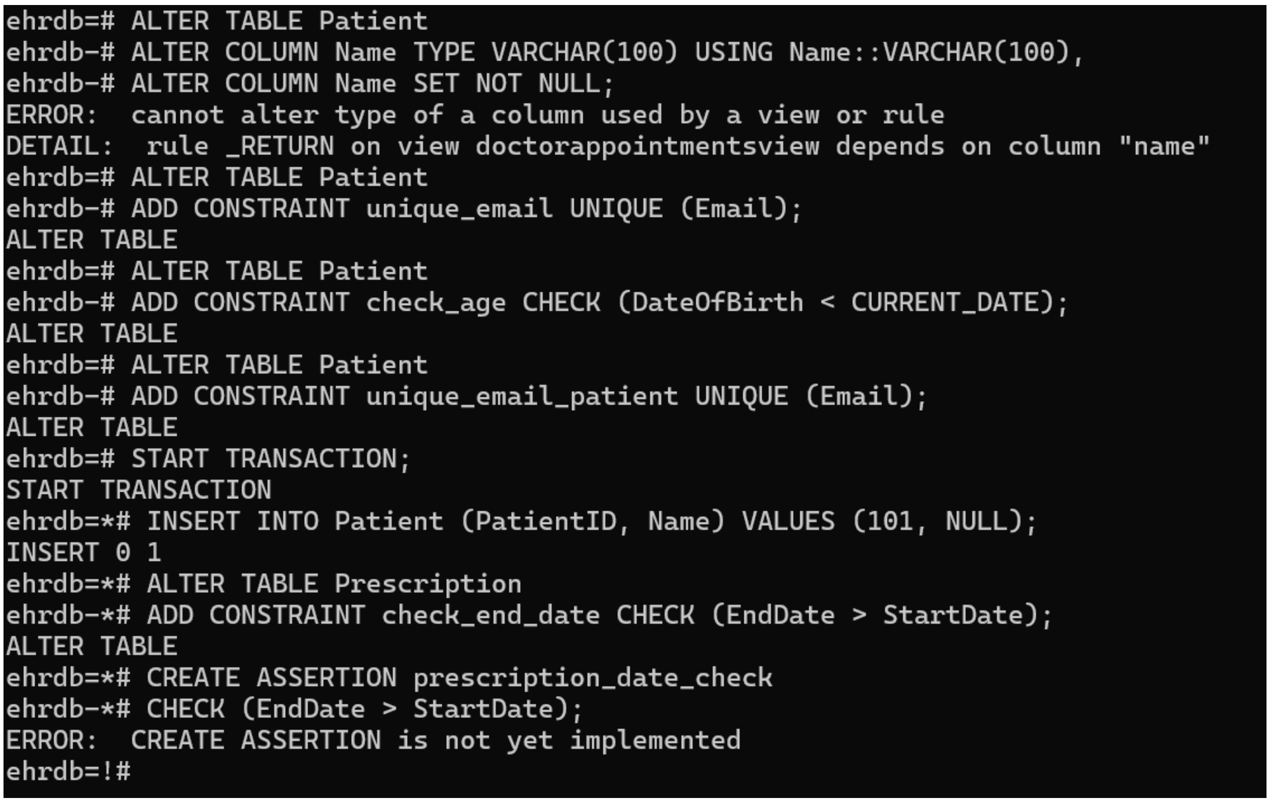


```
ehrdb=# START TRANSACTION;
START TRANSACTION
ehrdb=*# INSERT INTO Appointment (PatientID, DoctorID, AppointmentDate, AppointmentTime, Purpose)
ehrdb-*# VALUES (1, 1, '2024-03-06', '10:00:00', 'Regular checkup');
ERROR:  null value in column "appointmentid" of relation "appointment" violates not-null constraint
DETAIL:  Failing row contains (null, 1, 1, 2024-03-06, 10:00:00, Regular checkup).
```



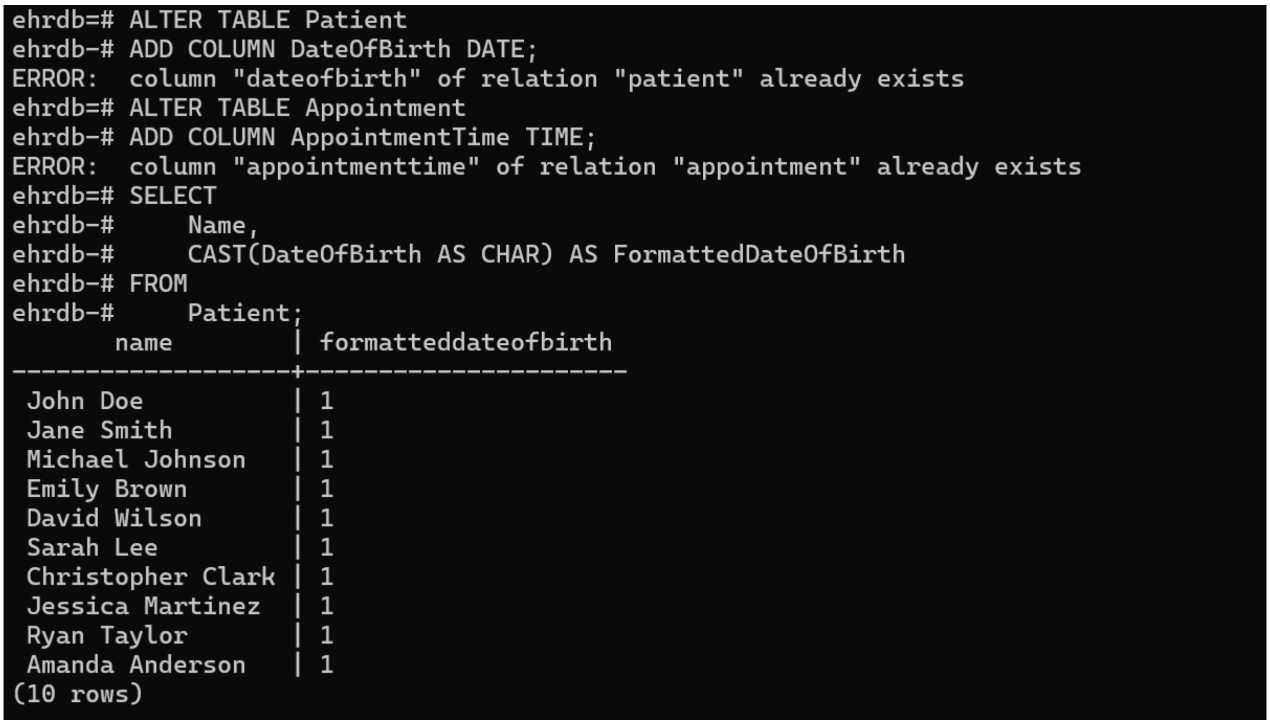
```
ehrdb=# START TRANSACTION;
START TRANSACTION
ehrdb=*# UPDATE MedicalHistory
ehrdb-*# SET MedicalHistoryDetails = 'Patient has a history of asthma.'
ehrdb-*# WHERE PatientID = 1;
UPDATE 1
ehrdb=*# INSERT INTO Prescription (DoctorID, PatientID, MedicationName, Dosage, Frequency, StartDate, EndDate)
ehrdb-*# VALUES (1, 1, 'Ventolin', '2 puffs', 'As needed', '2024-03-06', '2024-04-06');
ERROR:  null value in column "prescriptionid" of relation "prescription" violates not-null constraint
DETAIL:  Failing row contains (null, 1, 1, Ventolin, 2 puffs, As needed, 2024-03-06, 2024-04-06).
ehrdb=!# COMMIT WORK;
ROLLBACK
ehrdb=#
```

Task 6 : to ensure that changes made to the database by authorized users do not result in a loss of data consistency.

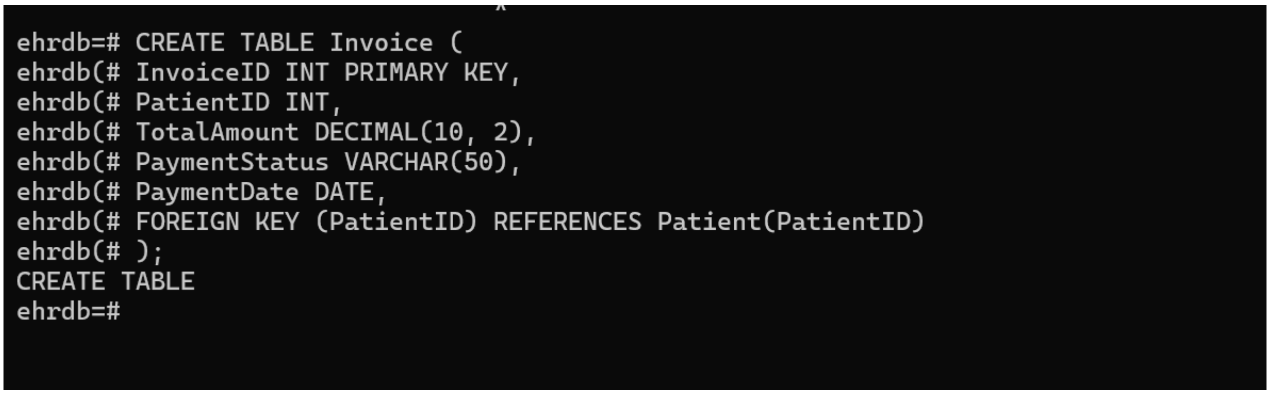


```
ehrdb=# ALTER TABLE Patient
ehrdb-# ALTER COLUMN Name TYPE VARCHAR(100) USING Name::VARCHAR(100),
ehrdb-# ALTER COLUMN Name SET NOT NULL;
ERROR:  cannot alter type of a column used by a view or rule
DETAIL:  rule _RETURN on view doctorappointmentsview depends on column "name"
ehrdb=# ALTER TABLE Patient
ehrdb-# ADD CONSTRAINT unique_email UNIQUE (Email);
ALTER TABLE
ehrdb=# ALTER TABLE Patient
ehrdb-# ADD CONSTRAINT check_age CHECK (DateOfBirth < CURRENT_DATE);
ALTER TABLE
ehrdb=# ALTER TABLE Patient
ehrdb-# ADD CONSTRAINT unique_email_patient UNIQUE (Email);
ALTER TABLE
ehrdb=# START TRANSACTION;
START TRANSACTION
ehrdb=*# INSERT INTO Patient (PatientID, Name) VALUES (101, NULL);
INSERT 0 1
ehrdb=*# ALTER TABLE Prescription
ehrdb-*# ADD CONSTRAINT check_end_date CHECK (EndDate > StartDate);
ALTER TABLE
ehrdb=*# CREATE ASSERTION prescription_date_check
ehrdb-*# CHECK (EndDate > StartDate);
ERROR:  CREATE ASSERTION is not yet implemented
ehrdb=!#
```

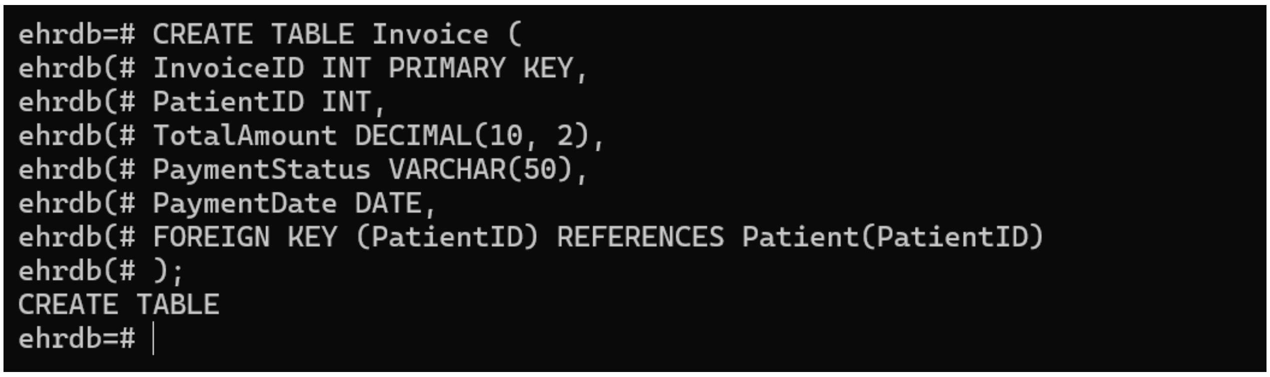
Task 7 : to ensure data integrity and optimize database performance I did SQL data types and schemas.



```
ehrdb=# ALTER TABLE Patient
ehrdb-# ADD COLUMN DateOfBirth DATE;
ERROR:  column "dateofbirth" of relation "patient" already exists
ehrdb=# ALTER TABLE Appointment
ehrdb-# ADD COLUMN AppointmentTime TIME;
ERROR:  column "appointmenttime" of relation "appointment" already exists
ehrdb=# SELECT
ehrdb-#     Name,
ehrdb-#     CAST(DateOfBirth AS CHAR) AS FormattedDateOfBirth
ehrdb-# FROM
ehrdb-#     Patient;
       name         | formatteddateofbirth
-------------------+----------------------
 John Doe           | 1
 Jane Smith         | 1
 Michael Johnson    | 1
 Emily Brown        | 1
 David Wilson       | 1
 Sarah Lee          | 1
 Christopher Clark  | 1
 Jessica Martinez   | 1
 Ryan Taylor        | 1
 Amanda Anderson    | 1
(10 rows)
```



```
ehrdb=# CREATE TABLE Invoice (
ehrdb(# InvoiceID INT PRIMARY KEY,
ehrdb(# PatientID INT,
ehrdb(# TotalAmount DECIMAL(10, 2),
ehrdb(# PaymentStatus VARCHAR(50),
ehrdb(# PaymentDate DATE,
ehrdb(# FOREIGN KEY (PatientID) REFERENCES Patient(PatientID)
ehrdb(# );
CREATE TABLE
ehrdb=#
```



```
ehrdb=# CREATE TABLE Invoice (
ehrdb(# InvoiceID INT PRIMARY KEY,
ehrdb(# PatientID INT,
ehrdb(# TotalAmount DECIMAL(10, 2),
ehrdb(# PaymentStatus VARCHAR(50),
ehrdb(# PaymentDate DATE,
ehrdb(# FOREIGN KEY (PatientID) REFERENCES Patient(PatientID)
ehrdb(# );
CREATE TABLE
ehrdb=#
```

Conclusion:

Task1 : Joining table is a fundamental SQL characteristic. Several types, including FULL OUTER JOIN, LEFT JOIN, RIGHT JOIN, and INNER JOIN, are used to accomplish this. The retrieval of matching facts from both tables, all records from one desk with matching records from the alternative, or all facts from each tables regardless of matches are all made possible through the various be a part of types.

Task 2: Merging Columns Names That Match. It is a traditional exercise to appoint columns whose names coincide with the be a part of condition whilst joining tables. This is because of the reality that these columns typically have values that fit and link facts from one desk to some other. For the proper records to be matched and merged, the join condition is crucial.

Task 3: Understanding the Different Types of Joints. LEFT OUTER JOIN: Returns all facts from the left table plus the matched facts from the proper desk. If there is no healthy, NULL values are presented in the proper desk columns.RIGHT OUTER JOIN: Returns all information from the proper desk plus the matching facts from the left desk. If there may be no match, NULL values are provided inside the left table columns. FULL OUTER JOIN: Returns all statistics if there is a in shape in both the left or proper tables. If there is no suit, NULL values are returned for the table's non-matching columns.

Task 4: Creating a View and Controlling User Access. Creating a view that combines affected person facts with drawing close appointments is a useful SQL utility. This is specifically useful for imparting a complete view of affected person statistics and healthcare regimens. Furthermore, defining a consumer with particular permissions, which include supplying access to medical doctor's appointments, is a crucial factor of database protection and access management.

Task 5: Transactions and Rollbacks. Transactions are a critical element in SQL that ensures statistics integrity. They allow a series of database movements to be run as an unmarried unit, with the option to devote the changes if a success or roll back the entire transaction if any operation fails. This guarantees that the database is constant.

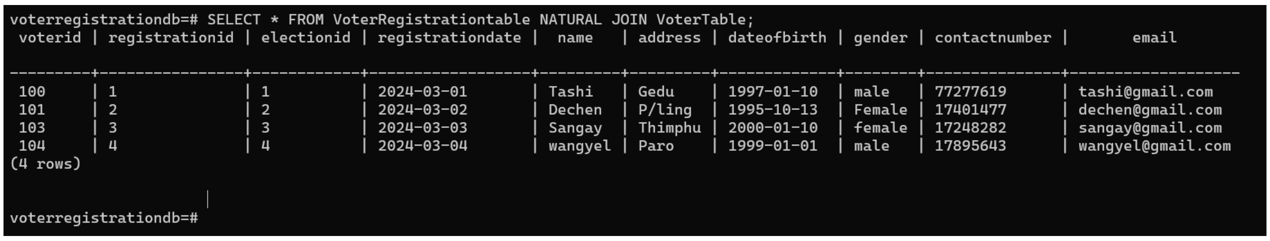
Task 6: Guaranteeing Data Coherence. For database operations to be correct and reliable, information consistency ought to be maintained. This involves making sure that adjustments made to the database by way of authorized users do not cause corruption or lack of records. Data consistency throughout the database can be guaranteed through imposing information integrity rules via using techniques like triggers, constraints, and transactions.

Task 7: Schemas and Data Types in SQL. Creating steady and effective databases requires a radical understanding of SQL records kinds and schemas. Schemas assist in organizing database objects and controlling access permissions, whereas information kinds specify the types of facts that can be stored in a database area. Through meticulous choice of records kinds and schema structure, builders can decorate database overall performance.

In the end, SQL joins, transactions, user access management, and the careful use of information types and schemas are all vital additives of effective database design and management. These responsibilities and concepts highlight the importance of information and making use of SQL concepts to build robust and stable databases.

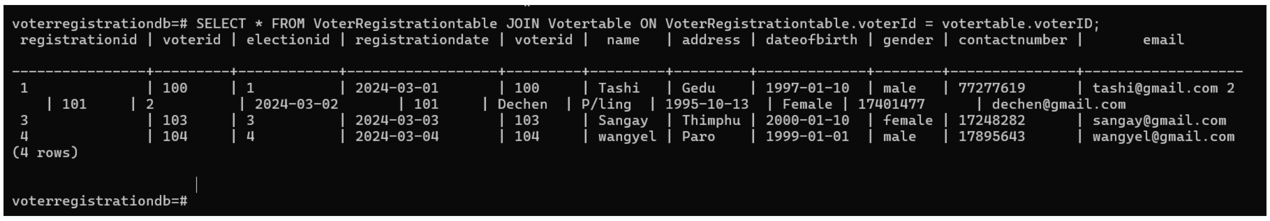
Topics : Practical 3

Task 1 : I joined the table having common columns in the database.



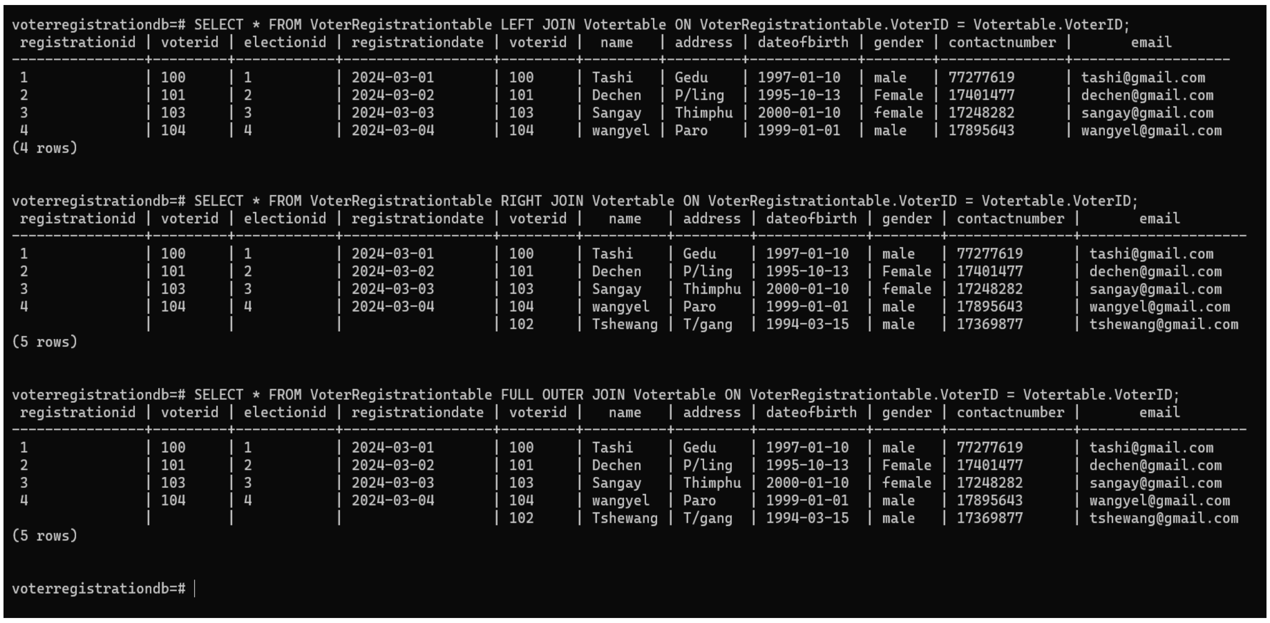
```
voterregistrationdb=# SELECT * FROM VoterRegistrationtable NATURAL JOIN VoterTable;
 voterid | registrationid | electionid | registrationdate |  name   | address | dateofbirth | gender | contactnumber |       email
---------+----------------+------------+------------------+---------+---------+-------------+--------+---------------+-------------------
 100     | 1              | 1          | 2024-03-01       | Tashi   | Gedu    | 1997-01-10  | male   | 77277619      | tashi@gmail.com
 101     | 2              | 2          | 2024-03-02       | Dechen  | P/ling  | 1995-10-13  | Female | 17401477      | dechen@gmail.com
 103     | 3              | 3          | 2024-03-03       | Sangay  | Thimphu | 2000-01-10  | female | 17248282      | sangay@gmail.com
 104     | 4              | 4          | 2024-03-04       | wangyel | Paro    | 1999-01-01  | male   | 17895643      | wangyel@gmail.com
(4 rows)


voterregistrationdb=#
```

Task 2 : I joined the column with the same name using join condition.



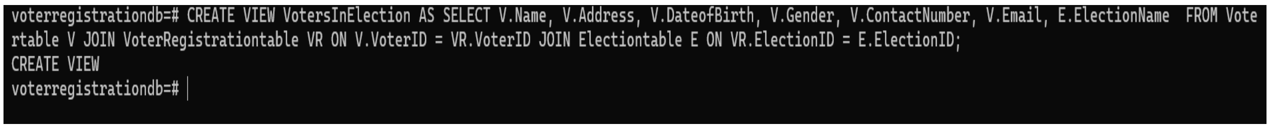
```
voterregistrationdb=# SELECT * FROM VoterRegistrationtable JOIN Votertable ON VoterRegistrationtable.voterId = votertable.voterID;
 registrationid | voterid | electionid | registrationdate | voterid |  name   | address | dateofbirth | gender | contactnumber |       email
----------------+---------+------------+------------------+---------+---------+---------+-------------+--------+---------------+-------------------
 1              | 100     | 1          | 2024-03-01       | 100     | Tashi   | Gedu    | 1997-01-10  | male   | 77277619      | tashi@gmail.com 2
    | 101     | 2          | 2024-03-02       | 101     | Dechen  | P/ling  | 1995-10-13  | Female | 17401477      | dechen@gmail.com
 3              | 103     | 3          | 2024-03-03       | 103     | Sangay  | Thimphu | 2000-01-10  | female | 17248282      | sangay@gmail.com
 4              | 104     | 4          | 2024-03-04       | 104     | wangyel | Paro    | 1999-01-01  | male   | 17895643      | wangyel@gmail.com
(4 rows)


voterregistrationdb=#
```

Task 3 : Left outer join returns all records from the left table and the matched records from the right table. Right outer join returns all records from the right table and the matched records from the left table. A full outer join returns all records when there is a match in either the left or right table. If there is no match, NULL values are returned for the columns of the table that lack a match.

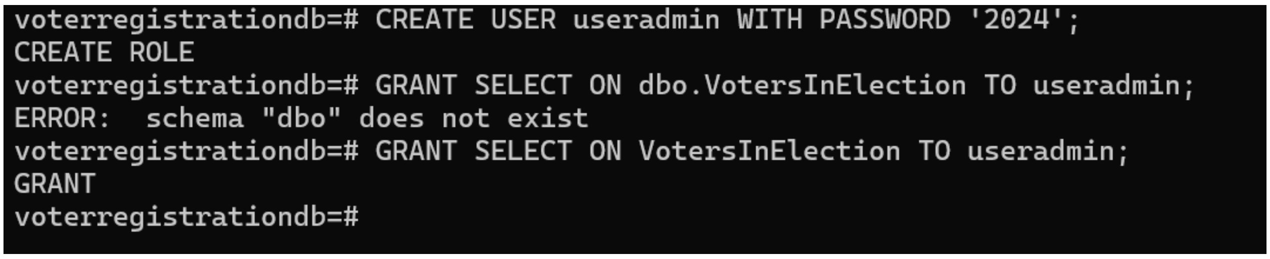


```
voterregistrationdb=# SELECT * FROM VoterRegistrationtable LEFT JOIN Votertable ON VoterRegistrationtable.VoterID = Votertable.VoterID;
 registrationid | voterid | electionid | registrationdate | voterid |  name   | address | dateofbirth | gender | contactnumber |       email
----------------+---------+------------+------------------+---------+---------+---------+-------------+--------+---------------+-------------------
 1              | 100     | 1          | 2024-03-01       | 100     | Tashi   | Gedu    | 1997-01-10  | male   | 77277619      | tashi@gmail.com
 2              | 101     | 2          | 2024-03-02       | 101     | Dechen  | P/ling  | 1995-10-13  | Female | 17401477      | dechen@gmail.com
 3              | 103     | 3          | 2024-03-03       | 103     | Sangay  | Thimphu | 2000-01-10  | female | 17248282      | sangay@gmail.com
 4              | 104     | 4          | 2024-03-04       | 104     | wangyel | Paro    | 1999-01-01  | male   | 17895643      | wangyel@gmail.com
(4 rows)


voterregistrationdb=# SELECT * FROM VoterRegistrationtable RIGHT JOIN Votertable ON VoterRegistrationtable.VoterID = Votertable.VoterID;
 registrationid | voterid | electionid | registrationdate | voterid |   name   | address | dateofbirth | gender | contactnumber |        email
----------------+---------+------------+------------------+---------+----------+---------+-------------+--------+---------------+--------------------
 1              | 100     | 1          | 2024-03-01       | 100     | Tashi    | Gedu    | 1997-01-10  | male   | 77277619      | tashi@gmail.com
 2              | 101     | 2          | 2024-03-02       | 101     | Dechen   | P/ling  | 1995-10-13  | Female | 17401477      | dechen@gmail.com
 3              | 103     | 3          | 2024-03-03       | 103     | Sangay   | Thimphu | 2000-01-10  | female | 17248282      | sangay@gmail.com
 4              | 104     | 4          | 2024-03-04       | 104     | wangyel  | Paro    | 1999-01-01  | male   | 17895643      | wangyel@gmail.com
                |         |            |                  | 102     | Tshewang | T/gang  | 1994-03-15  | male   | 17369877      | tshewang@gmail.com
(5 rows)


voterregistrationdb=# SELECT * FROM VoterRegistrationtable FULL OUTER JOIN Votertable ON VoterRegistrationtable.VoterID = Votertable.VoterID;
 registrationid | voterid | electionid | registrationdate | voterid |   name   | address | dateofbirth | gender | contactnumber |        email
----------------+---------+------------+------------------+---------+----------+---------+-------------+--------+---------------+--------------------
 1              | 100     | 1          | 2024-03-01       | 100     | Tashi    | Gedu    | 1997-01-10  | male   | 77277619      | tashi@gmail.com
 2              | 101     | 2          | 2024-03-02       | 101     | Dechen   | P/ling  | 1995-10-13  | Female | 17401477      | dechen@gmail.com
 3              | 103     | 3          | 2024-03-03       | 103     | Sangay   | Thimphu | 2000-01-10  | female | 17248282      | sangay@gmail.com
 4              | 104     | 4          | 2024-03-04       | 104     | wangyel  | Paro    | 1999-01-01  | male   | 17895643      | wangyel@gmail.com
                |         |            |                  | 102     | Tshewang | T/gang  | 1994-03-15  | male   | 17369877      | tshewang@gmail.com
(5 rows)


voterregistrationdb=#
```

Task 4 : Since it is not always desirable for all users to see the entire set of relations in the database. I created the new user as useradmin with password 12345 using the SQL authorization mechanism to restrict access to relations and granted access to useradmin .

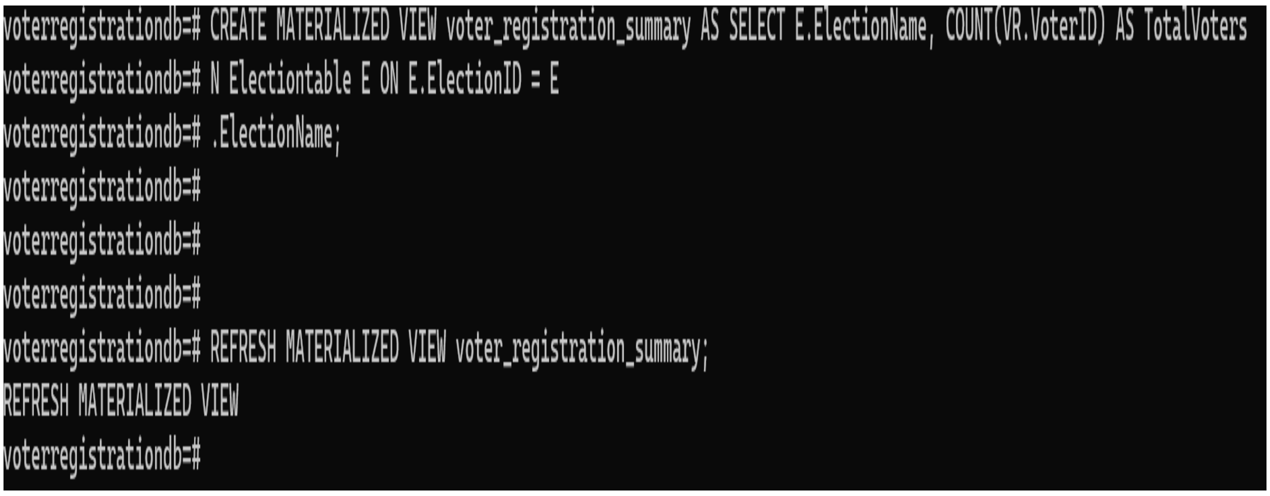


```
voterregistrationdb=# CREATE VIEW VotersInElection AS SELECT V.Name, V.Address, V.DateofBirth, V.Gender, V.ContactNumber, V.Email, E.ElectionName  FROM Vote
rtable V JOIN VoterRegistrationtable VR ON V.VoterID = VR.VoterID JOIN Electiontable E ON VR.ElectionID = E.ElectionID;
CREATE VIEW
voterregistrationdb=# |
```



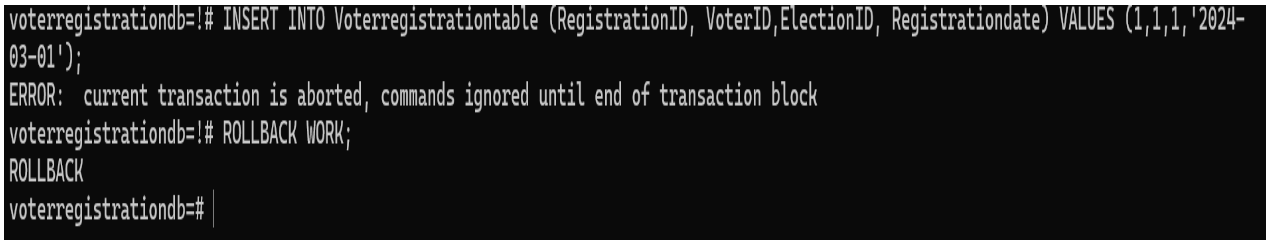
```
voterregistrationdb=# CREATE USER useradmin WITH PASSWORD '2024';
CREATE ROLE
voterregistrationdb=# GRANT SELECT ON dbo.VotersInElection TO useradmin;
ERROR:  schema "dbo" does not exist
voterregistrationdb=# GRANT SELECT ON VotersInElection TO useradmin;
GRANT
voterregistrationdb=#
```

And to check whether the voting relations are stored and kept up-to-date or not I created a materialized view.



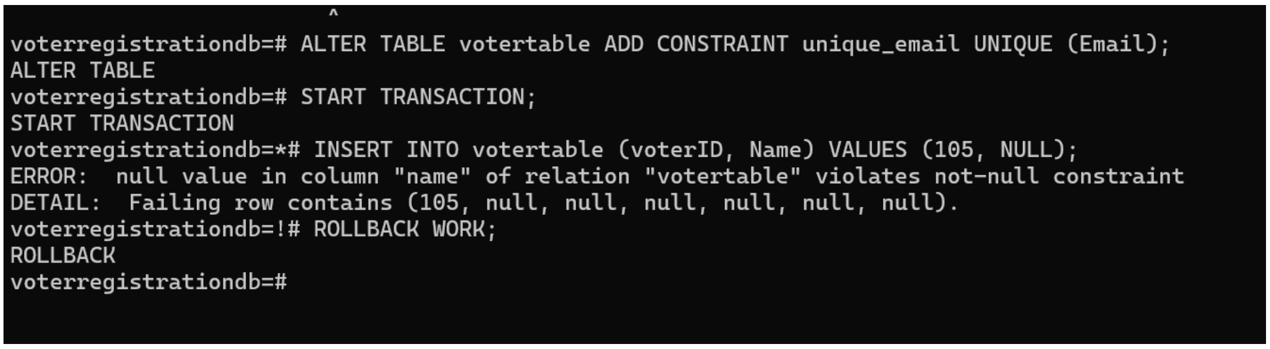
```
voterregistrationdb=# CREATE MATERIALIZED VIEW voter_registration_summary AS SELECT E.ElectionName, COUNT(VR.VoterID) AS TotalVoters
voterregistrationdb=# N Electiontable E ON E.ElectionID = E
voterregistrationdb=# .ElectionName;
voterregistrationdb=#
voterregistrationdb=#
voterregistrationdb=#
voterregistrationdb=# REFRESH MATERIALIZED VIEW voter_registration_summary;
REFRESH MATERIALIZED VIEW
voterregistrationdb=#
```

Task 5 : to see the update query statement, i did a transaction but couldn't commit the work. So I rolled back the work.



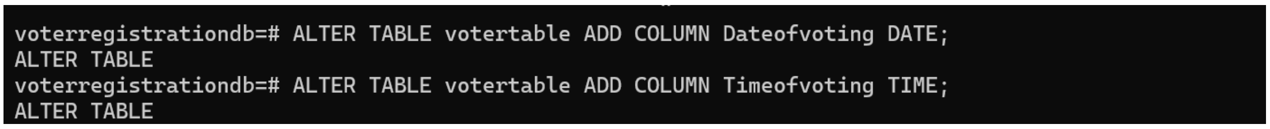
```
voterregistrationdb=!# INSERT INTO Voterregistrationtable (RegistrationID, VoterID,ElectionID, Registrationdate) VALUES (1,1,1,'2024-
03-01');
ERROR:  current transaction is aborted, commands ignored until end of transaction block
voterregistrationdb=!# ROLLBACK WORK;
ROLLBACK
voterregistrationdb=# |
```

Task 6 : in task 6 I tried to check for the null value but there was no null value. So I created a voter email as the unique constraint and started the transaction but it violates the not-null constraint. I have to roll back the work.

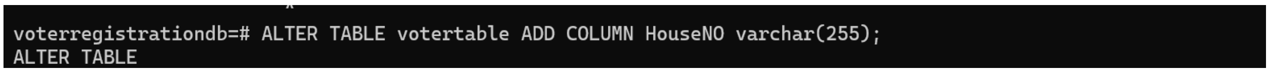


```
                         ^
voterregistrationdb=# ALTER TABLE votertable ADD CONSTRAINT unique_email UNIQUE (Email);
ALTER TABLE
voterregistrationdb=# START TRANSACTION;
START TRANSACTION
voterregistrationdb=*# INSERT INTO votertable (voterID, Name) VALUES (105, NULL);
ERROR:  null value in column "name" of relation "votertable" violates not-null constraint
DETAIL:  Failing row contains (105, null, null, null, null, null, null).
voterregistrationdb=!# ROLLBACK WORK;
ROLLBACK
voterregistrationdb=#
```

Task 7 : To ensure data integrity and optimize database performance I did SQL data types and schemas. I added the column in the voter table (date of voting and time of voting) and to generate the unique key values I added the house number. Since every house has its own unique number.



```
voterregistrationdb=# ALTER TABLE votertable ADD COLUMN Dateofvoting DATE;
ALTER TABLE
voterregistrationdb=# ALTER TABLE votertable ADD COLUMN Timeofvoting TIME;
ALTER TABLE
```



```
voterregistrationdb=# ALTER TABLE votertable ADD COLUMN HouseNO varchar(255);
ALTER TABLE
```

Conclusion:

Those are all critical tasks while working with databases. Joining tables allows us to integrate data from various tables into a single result set, which is useful when searching and evaluating data. Outer joins allow us to include records from one or both tables even if they do not match, which can be important for finding missing data. Moreover, creating a new user with restricted database access is an important security step because it gives us control over who has access to the data and what useradmin can do with it. Materialized views can enhance query performance by storing query results.

On top of that, adding columns to a table and guaranteeing data integrity using unique constraints are critical components of database accuracy and consistency. By including the date and time of voting columns, we can trace when each voter voted. Adding a unique constraint to the house number column ensures that each house is only represented once in the table, reducing data entering errors and ensuring data consistency.