Assignment 2 (DIABETES PREDICATION ASSESSMENT PSYLIQ)

SQL Query

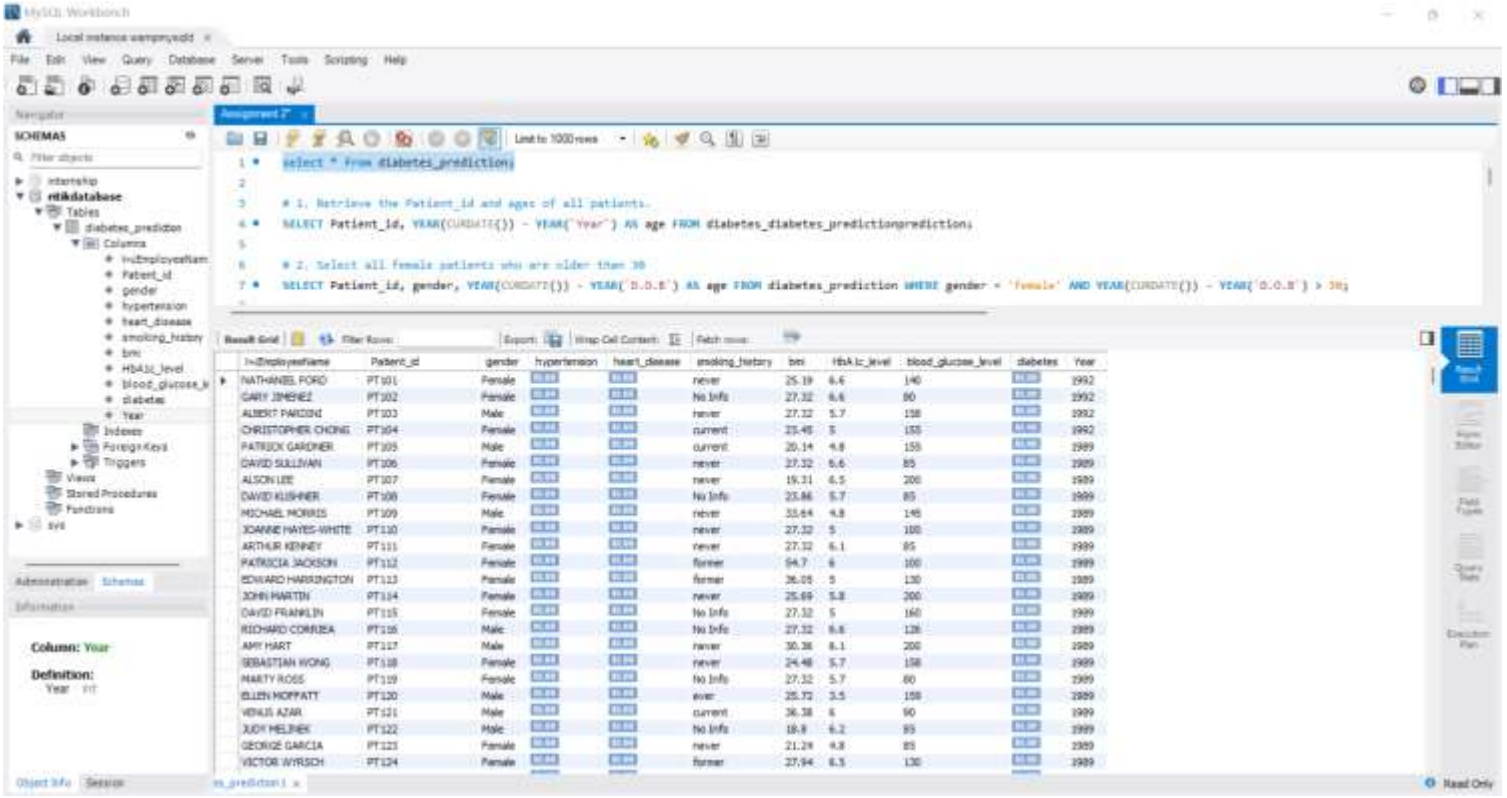
```
select * from diabetes_prediction;


# 1. Retrieve the Patient_id and ages of all patients.
# 1. Add the age column
ALTER TABLE diabetes_prediction
ADD COLUMN age INT;
-- 2. Disable safe update mode
SET SQL_SAFE_UPDATES = 0;

-- 3. Calculate and update the age column
UPDATE diabetes_prediction
SET age = YEAR(CURDATE()) - Year - (DATE_FORMAT(CURDATE(), '%m%d') < DATE_FORMAT(CONCAT(Year, '-12-31'), '%m%d'));

-- 4. Re-enable safe update mode
SET SQL_SAFE_UPDATES = 1;

SELECT Patient_id, Year, age
FROM diabetes_prediction
LIMIT 10;

SELECT Patient_id,
    YEAR(CURDATE()) - Year - (DATE_FORMAT(CURDATE(), '%m%d') < DATE_FORMAT(CONCAT(Year, '-12-31'), '%m%d')) AS age
FROM diabetes_prediction;



# 2. Select all female patients who are older than 30
SELECT Patient_id, gender,
    YEAR(CURDATE()) - Year - (DATE_FORMAT(CURDATE(), '%m%d') < DATE_FORMAT(CONCAT(Year))) AS age
FROM diabetes_prediction
WHERE gender = 'female' AND
   YEAR(CURDATE()) - Year - (DATE_FORMAT(CURDATE(), '%m%d') < DATE_FORMAT(CONCAT(Year))) > 30;
```

```
# 3. Calculate the average BMI of patients
SELECT AVG(bmi) AS average_bmi FROM diabetes_prediction;

# 4. List patients in descending order of blood glucose levels
SELECT Patient_id, blood_glucose_level FROM diabetes_prediction ORDER BY blood_glucose_level DESC;

# 5. Find patients who have hypertension and diabetes
SELECT Patient_id FROM diabetes_prediction WHERE hypertension = 1 AND diabetes = 1;

# 6. Determine the number of patients with heart disease
SELECT COUNT(*) AS heart_disease_count FROM diabetes_prediction WHERE heart_disease = 1;

#7. Group patients by smoking history and count how many smokers and non-smokers there are
SELECT smoking_history, COUNT(*) AS count FROM diabetes_prediction GROUP BY smoking_history;

# 8. Retrieve the Patient_id of patients who have a BMI greater than the average BMI
SELECT Patient_id FROM diabetes_prediction WHERE bmi > (SELECT AVG(bmi) FROM diabetes_prediction);

#9. Find the patient with the highest HbA1c level and the patient with the lowest HbA1c level
-- Highest HbA1c level
SELECT Patient_id, HbA1c_level
FROM diabetes_prediction ORDER BY HbA1c_level DESC LIMIT 1;

-- Lowest HbA1c level
SELECT Patient_id, HbA1c_level FROM diabetes_prediction ORDER BY HbA1c_level ASC LIMIT 1;

#10. Calculate the age of patients in years (assuming the current date as of now)
SELECT Patient_id, (YEAR(CURRENT_DATE) - Year) AS Age FROM diabetes_prediction;

# 11. Rank patients by blood glucose level within each gender group
SELECT Patient_id, gender, blood_glucose_level,
RANK() OVER (PARTITION BY gender ORDER BY blood_glucose_level DESC) AS RankOfPatient
FROM diabetes_prediction;
```

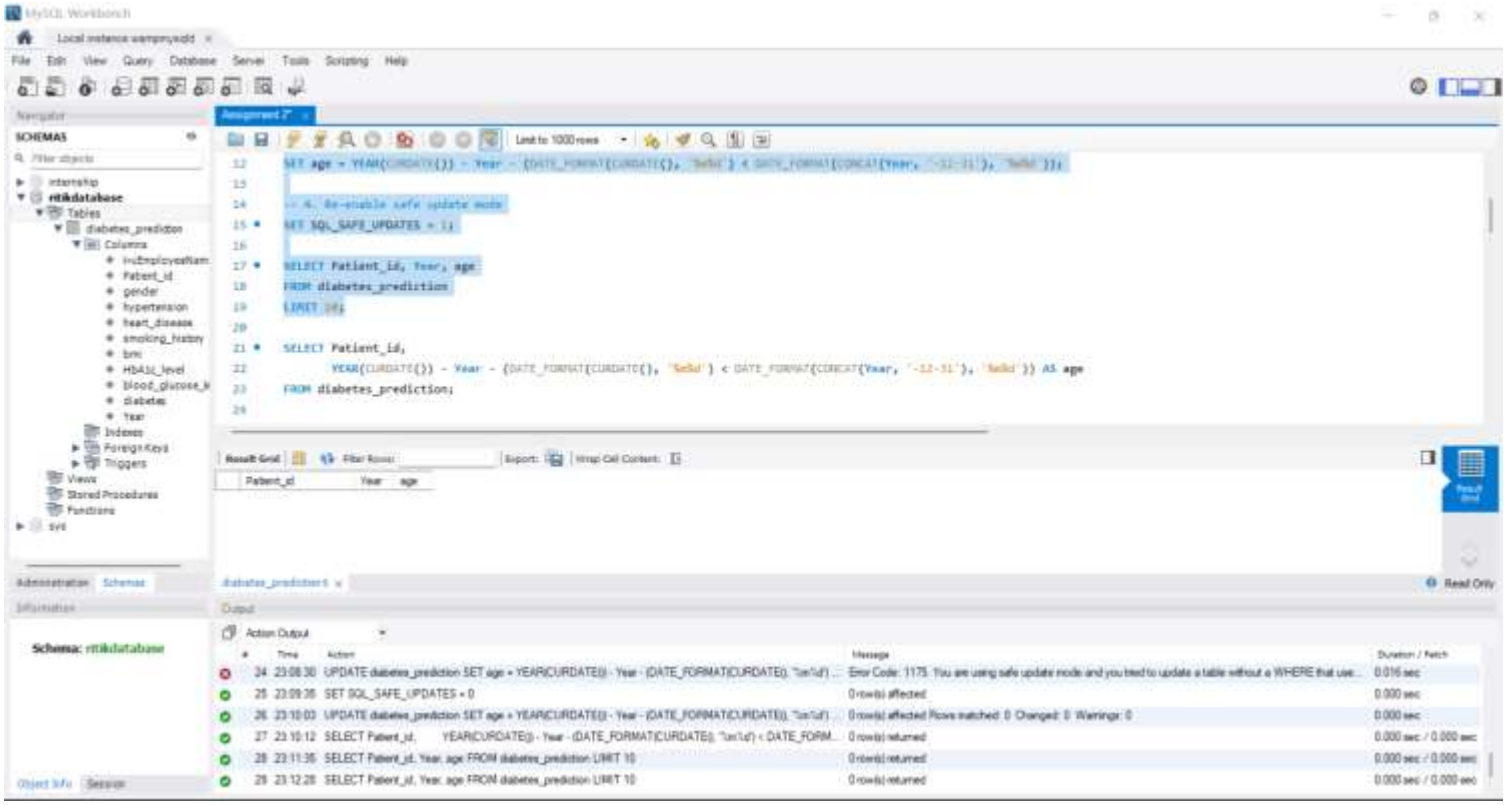
```
# 12. Update the smoking history of patients who are older than 40 to "Ex-smoker"

UPDATE diabetes_prediction

SET smoking_history = 'Ex-smoker'

WHERE (YEAR(CURRENT_DATE) - Year) > 40;


# 13. Insert a new patient into the database with sample data

INSERT INTO diabetes_prediction (Patient_id, gender, Year, hypertension, heart_disease, smoking_history, bmi, HbA1c_level, blood_glucose_level, diabetes)

VALUES ('PT999', 'Male', 1985, 0, 0, 'never', 22.5, 5.5, 90, 0);


# 14. Delete all patients with heart disease from the database

DELETE FROM diabetes_prediction WHERE heart_disease = 1;


#15. Find patients who have hypertension but not diabetes using the EXCEPT operator MySQL does not support the EXCEPT operator directly. Instead, we can use a subquery.

SELECT Patient_id

FROM diabetes_prediction

WHERE hypertension = 1

AND Patient_id NOT IN (

   SELECT Patient_id

   FROM diabetes_prediction

   WHERE diabetes = 1

);

# 16. Define a unique constraint on the Patient_id column to ensure its values are unique

-- Step 1: Check for Duplicate Patient_ids

SELECT Patient_id, COUNT(*)

FROM diabetes_prediction

GROUP BY Patient_id

HAVING COUNT(*) > 1;


-- If duplicates are found, proceed to Step 2

-- Step 2: Remove or Handle Duplicates


-- Option A: Delete duplicates, keeping only the first occurrence
```

```
DELETE dp1
FROM diabetes_prediction dp1
INNER JOIN diabetes_prediction dp2
WHERE dp1.Patient_id = dp2.Patient_id
AND dp1.rowid > dp2.rowid;

-- Option B: Update duplicates to make them unique
-- (Uncomment if you prefer updating instead of deleting)
-- UPDATE diabetes_prediction dp1
-- INNER JOIN (
--     SELECT Patient_id, rowid
--     FROM diabetes_prediction
--     WHERE Patient_id IN (
--         SELECT Patient_id
--         FROM diabetes_prediction
--         GROUP BY Patient_id
--         HAVING COUNT(*) > 1
--     )
-- ) dp2 ON dp1.rowid = dp2.rowid
-- SET dp1.Patient_id = CONCAT(dp1.Patient_id, '_', dp1.rowid);

-- Step 3: Add the Unique Constraint
ALTER TABLE diabetes_prediction
ADD CONSTRAINT unique_patient_id UNIQUE (Patient_id);



# 17. Create a view that displays the Patient_id, ages, and BMI of patients
CREATE VIEW patient_info AS
SELECT Patient_id, (YEAR(CURRENT_DATE) - Year) AS Age, bmi
FROM diabetes_prediction;
SET SQL_SAFE_UPDATES = 1;
```

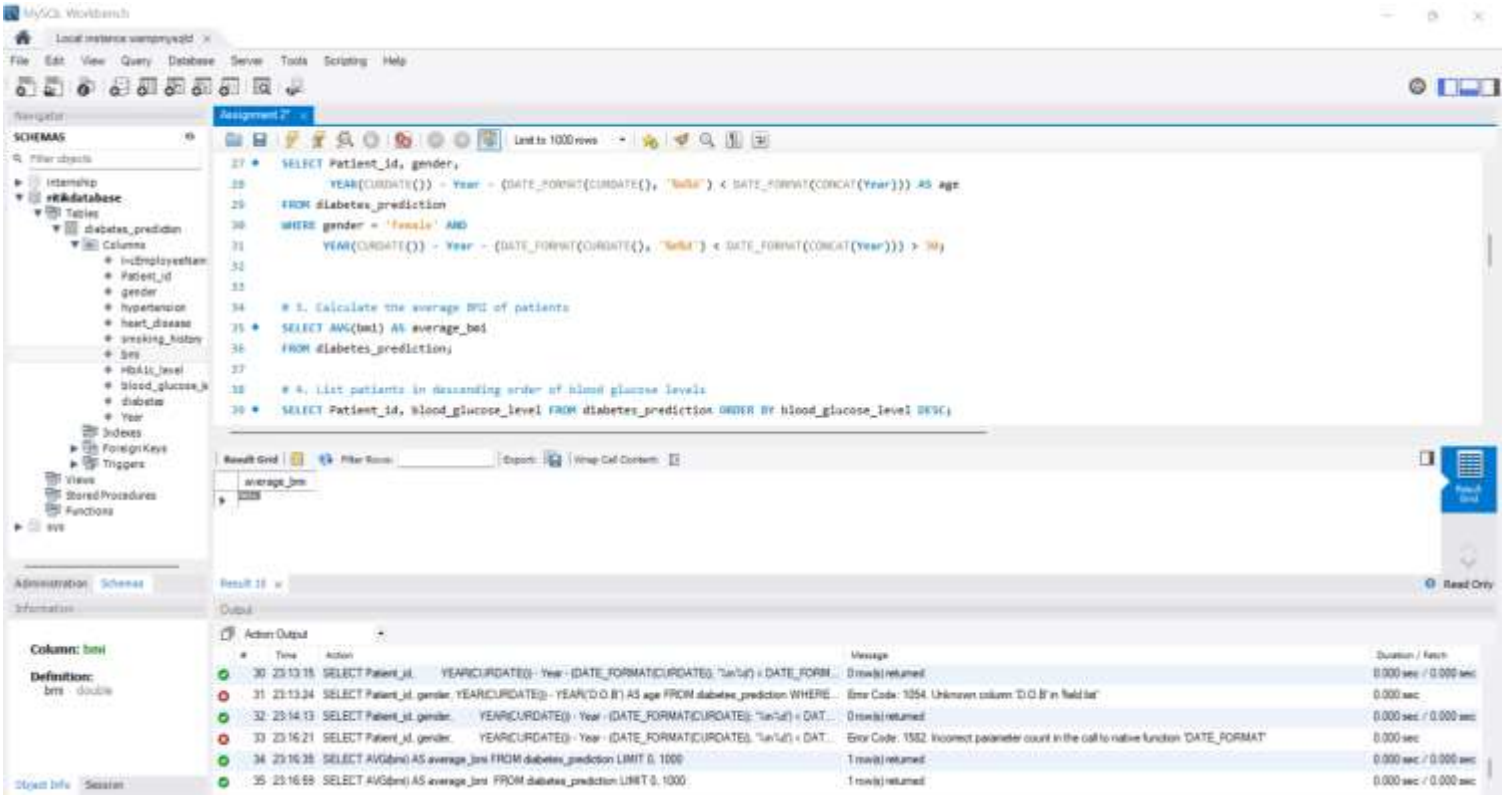
0) Display Data



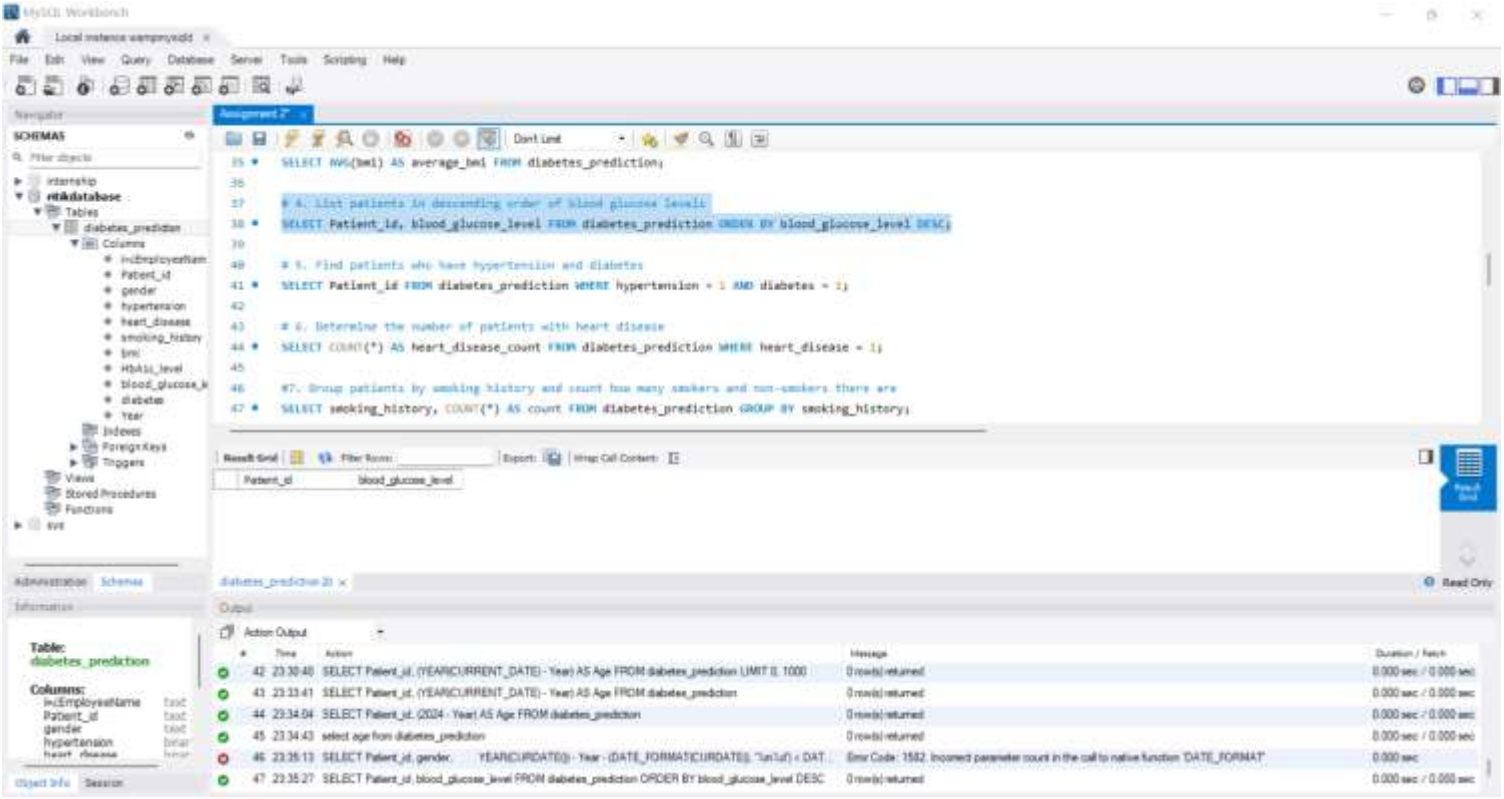
1) Retrieve the Patient_id and ages of all patients.



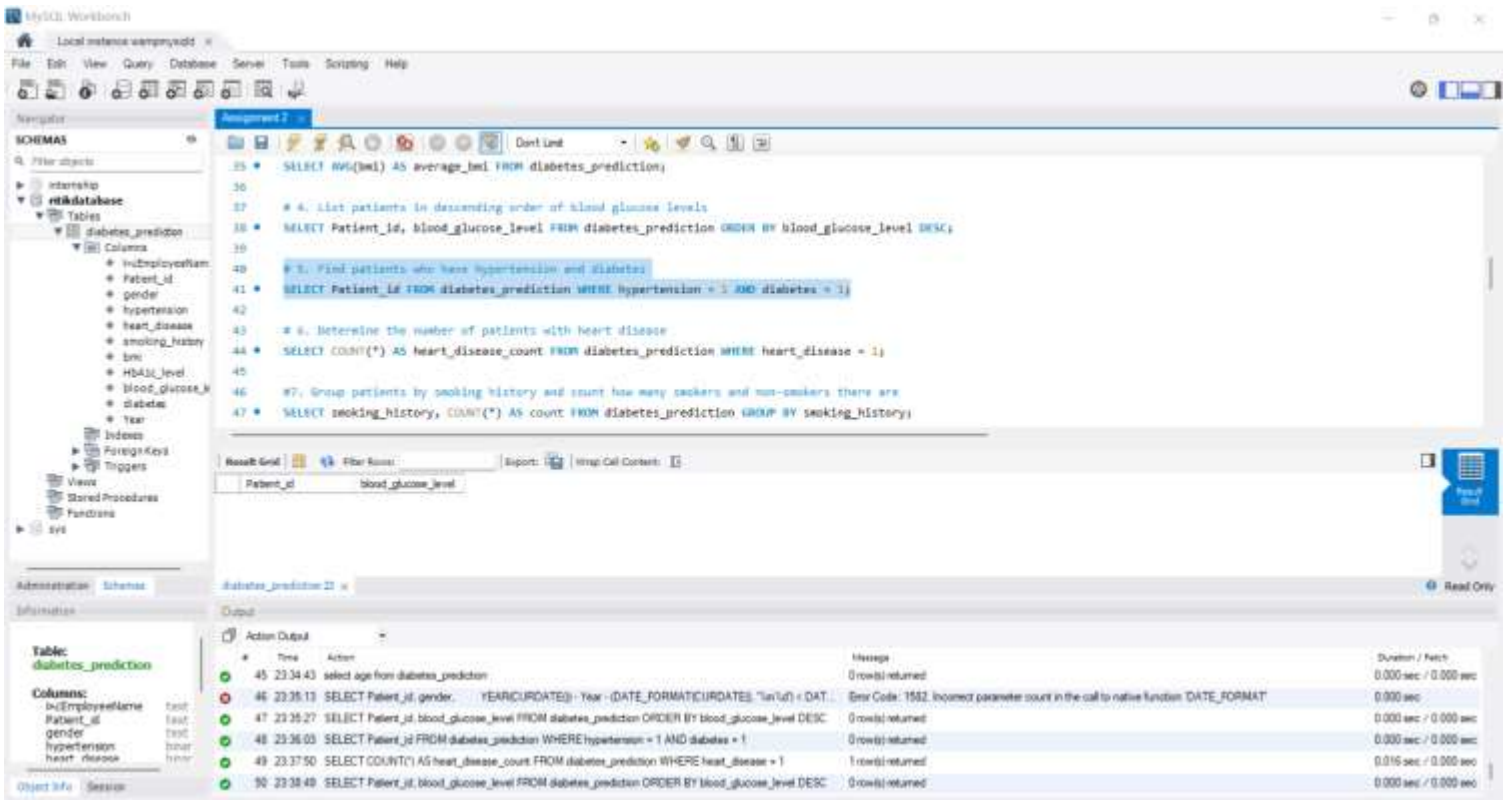
3) Calculate the average BMI of patients.



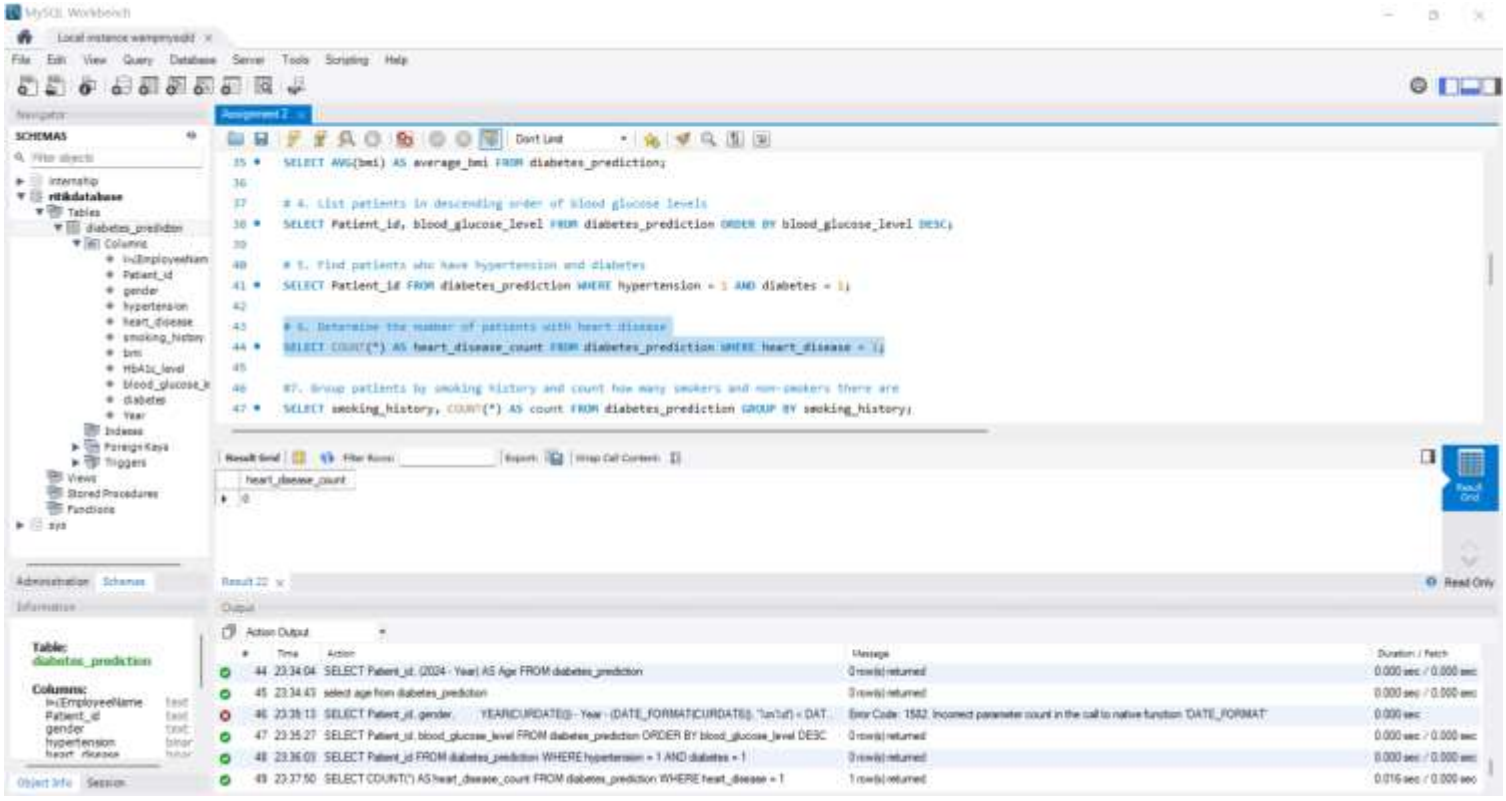
4) List patients in descending order of blood glucose levels.



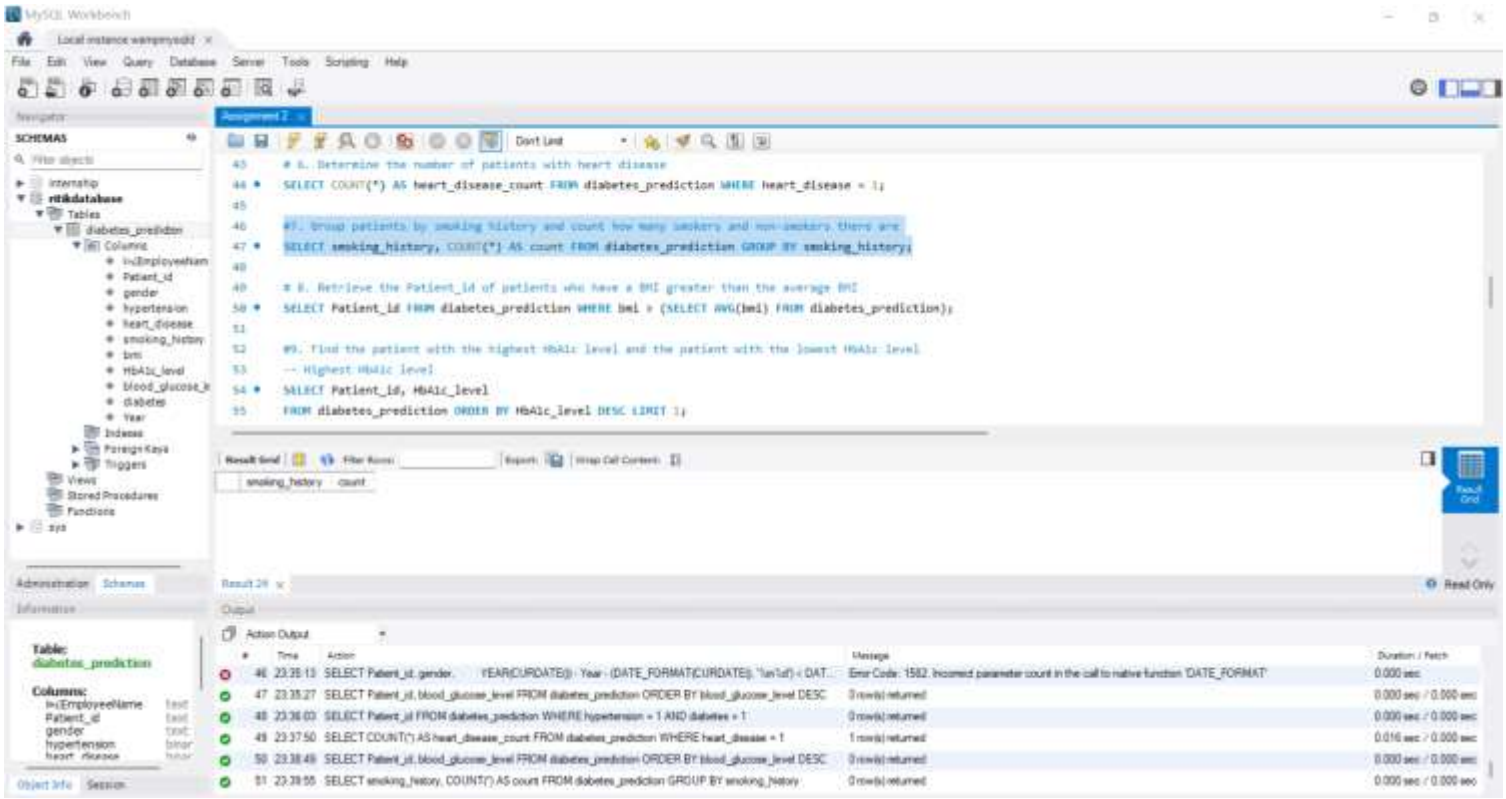
5) Find patients who have hypertension and diabetes.



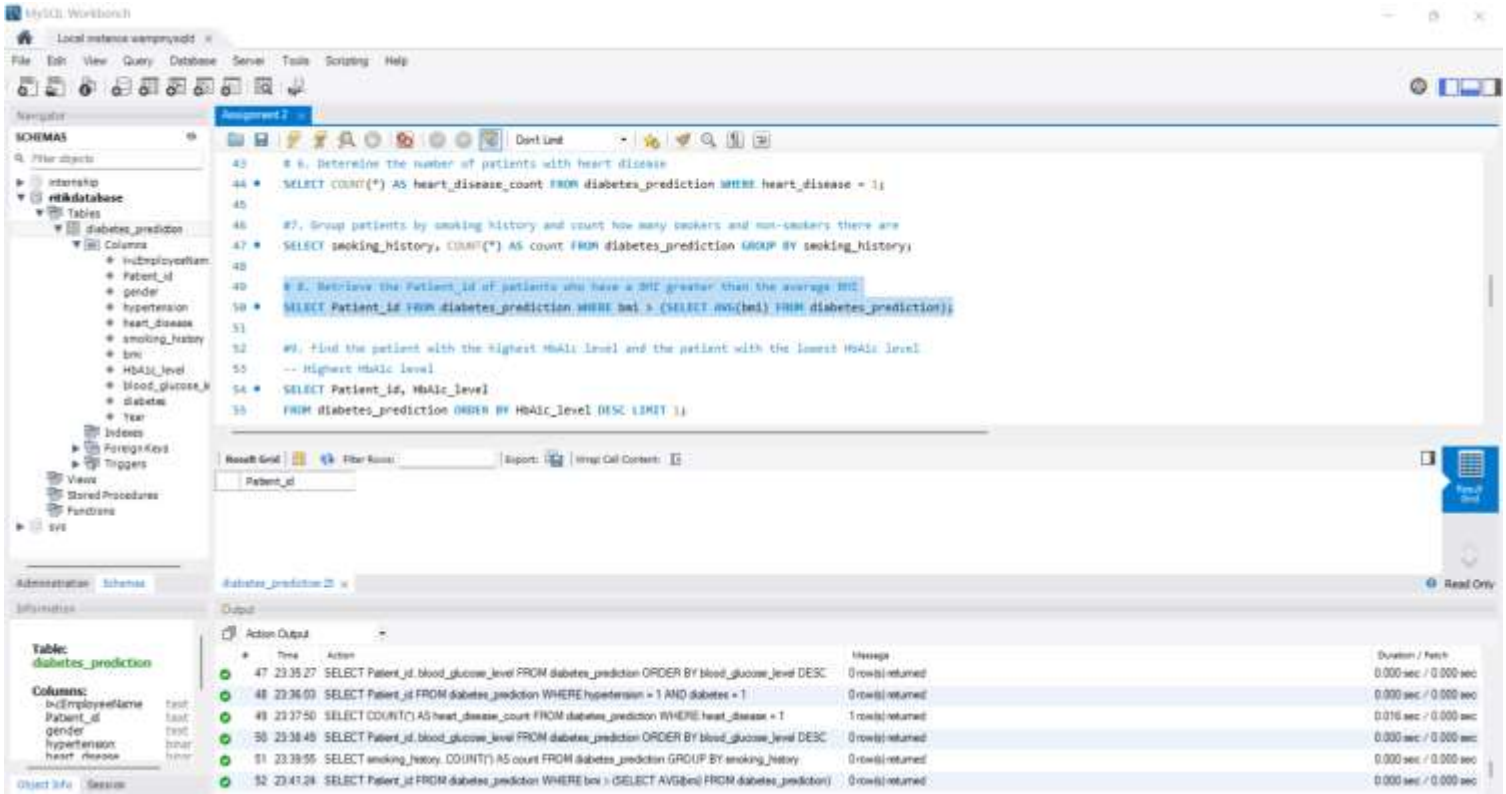
6) Determine the number of patients with heart disease.



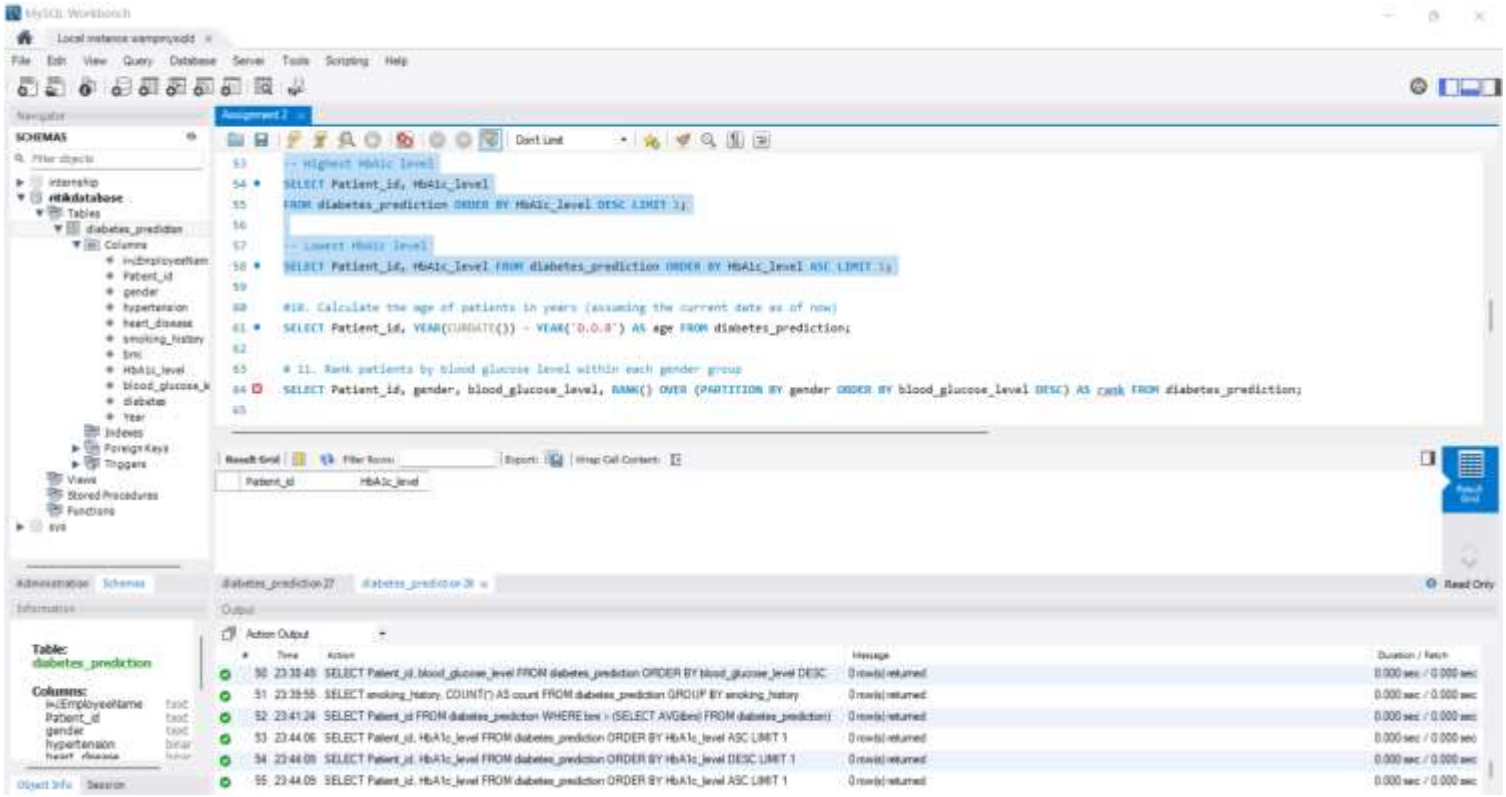
7) Group patients by smoking history and count how many smokers and non-smokers there are.



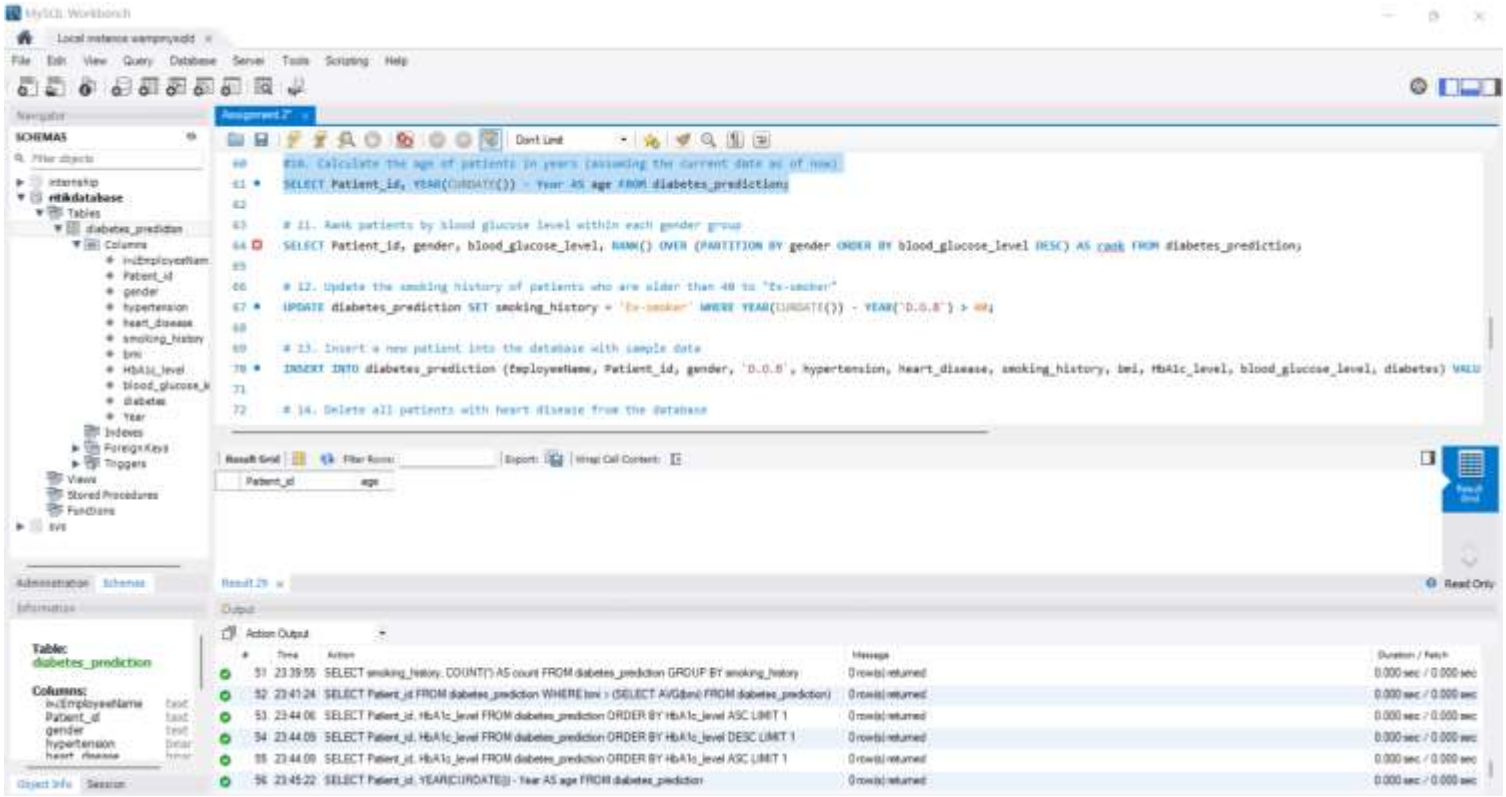
8) Retrieve the Patient_id of patients who have a BMI greater than the average BMI.



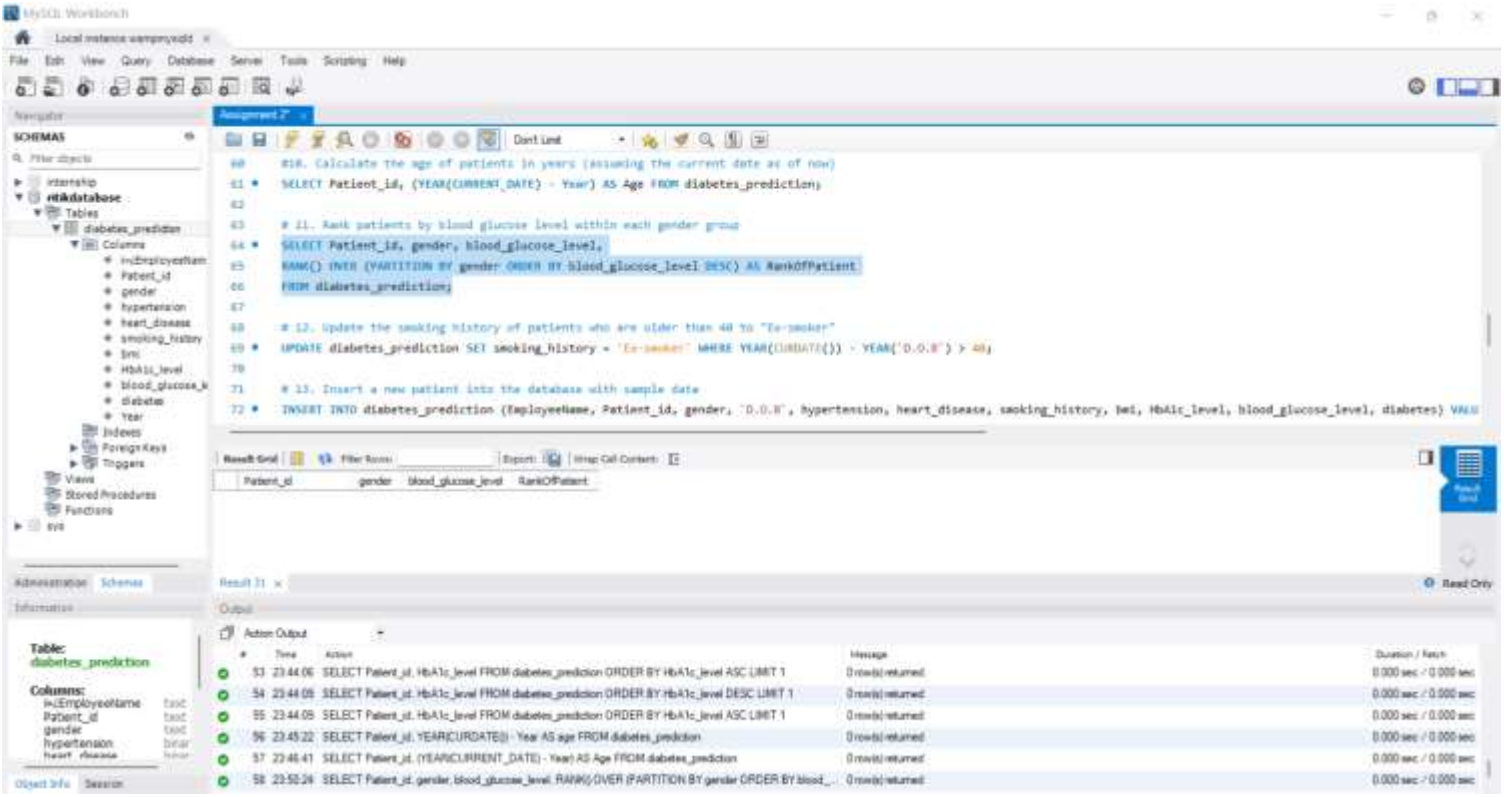
9) Find the patient with the highest HbA1c level and the patient with the lowest HbA1clevel.



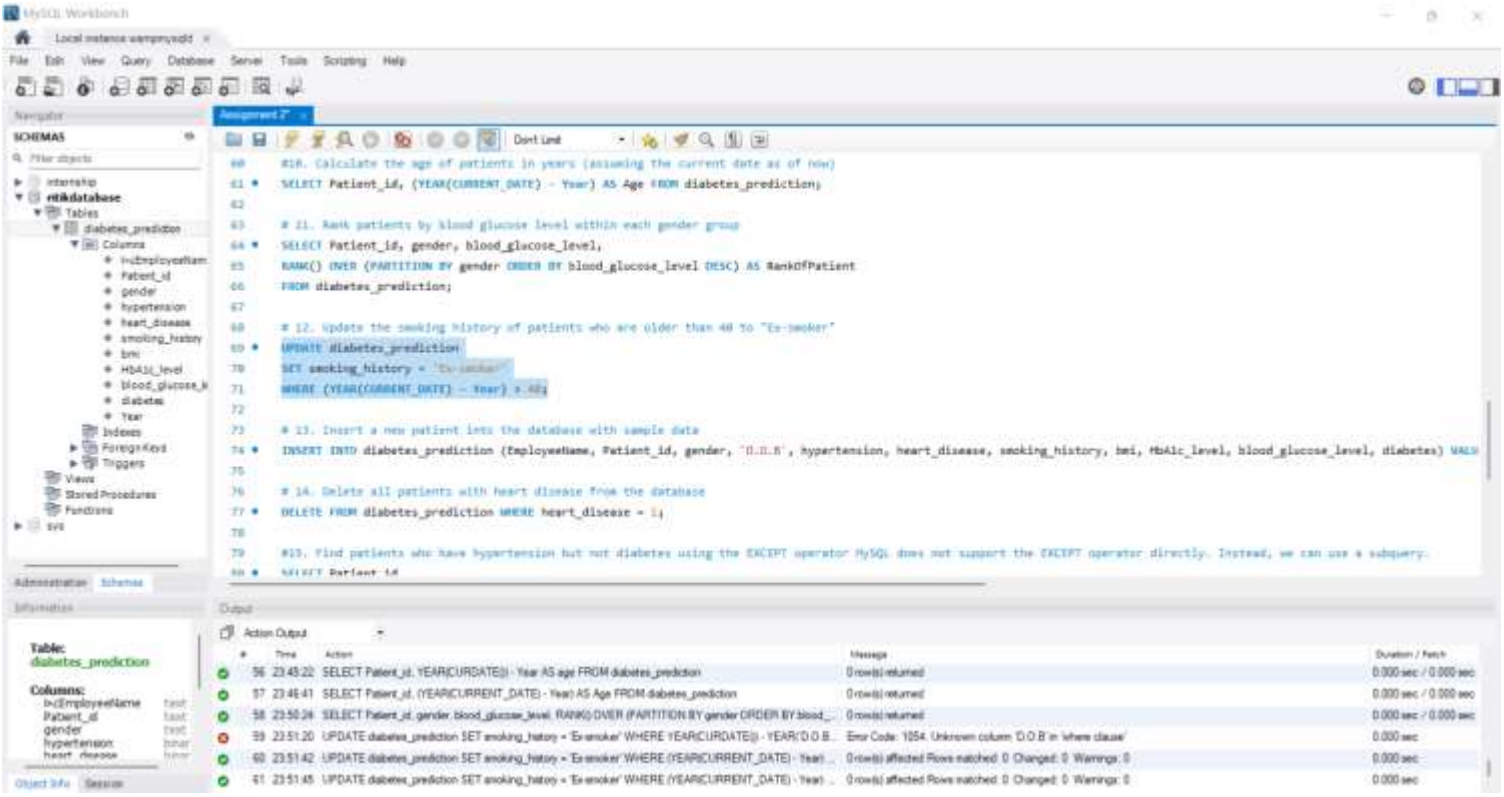
10) Calculate the age of patients in years (assuming the current date as of now).



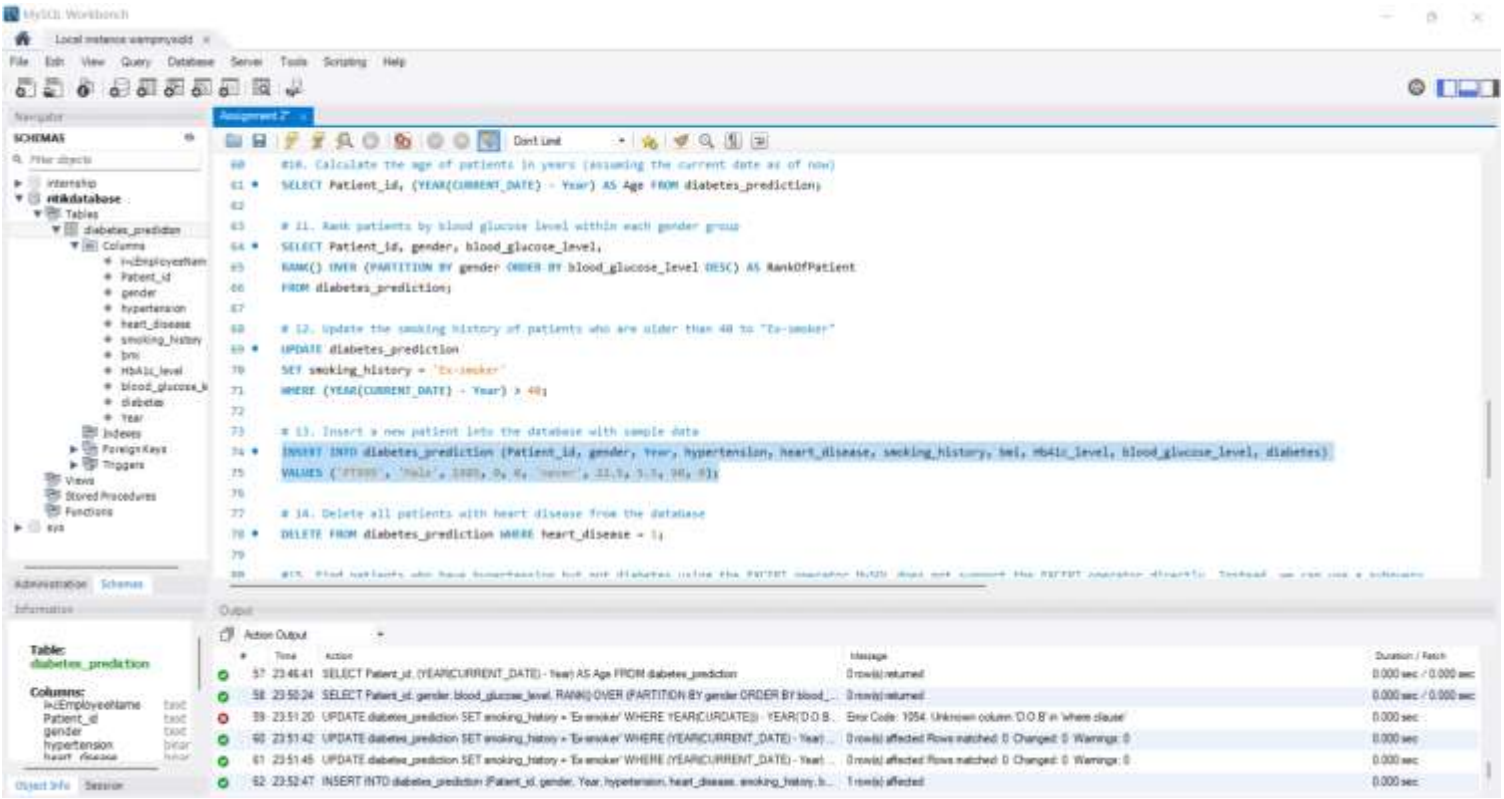
11) Rank patients by blood glucose level within each gender group.



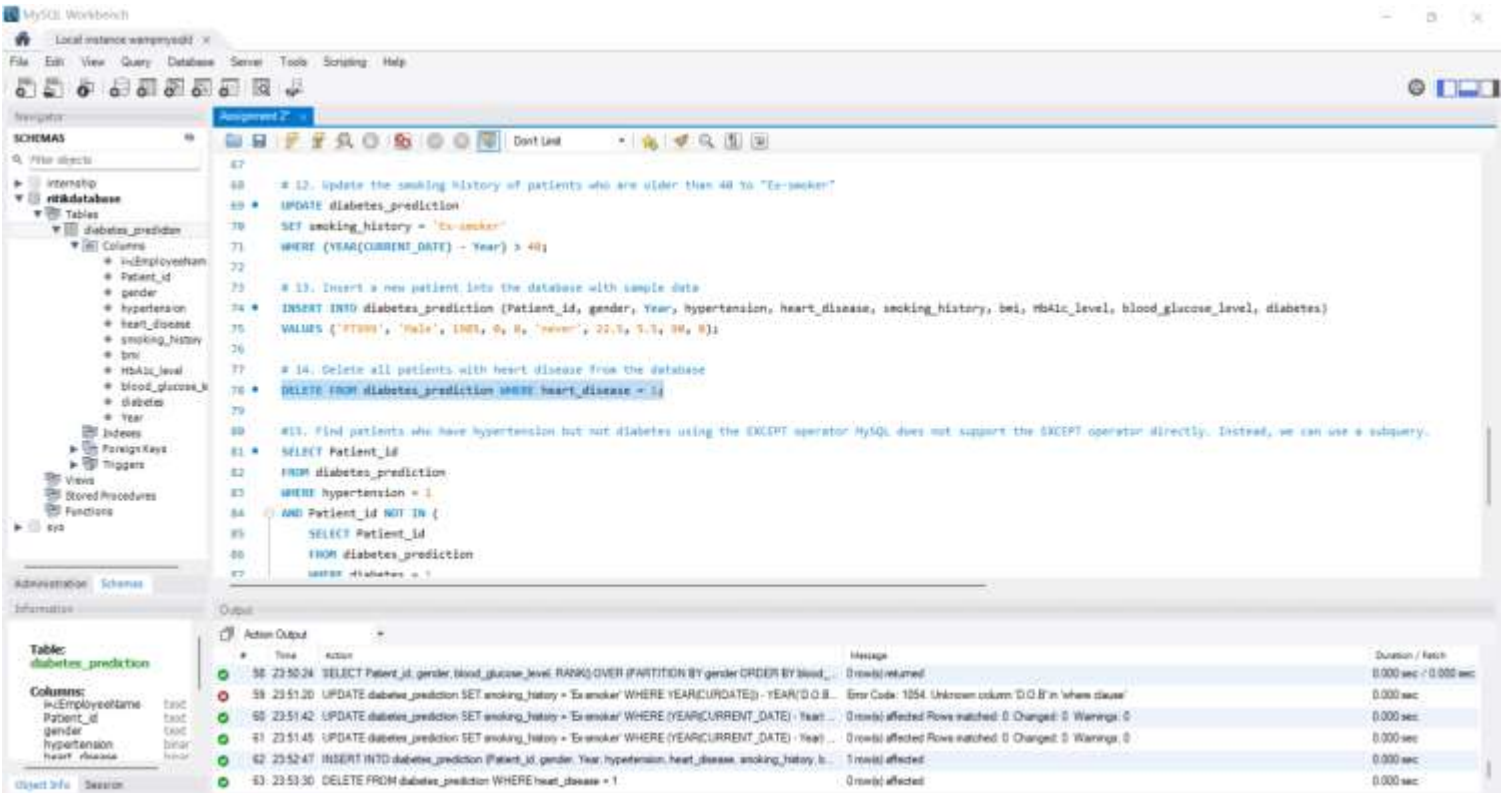
12) Update the smoking history of patients who are olderthan 40 to "Ex-smoker."



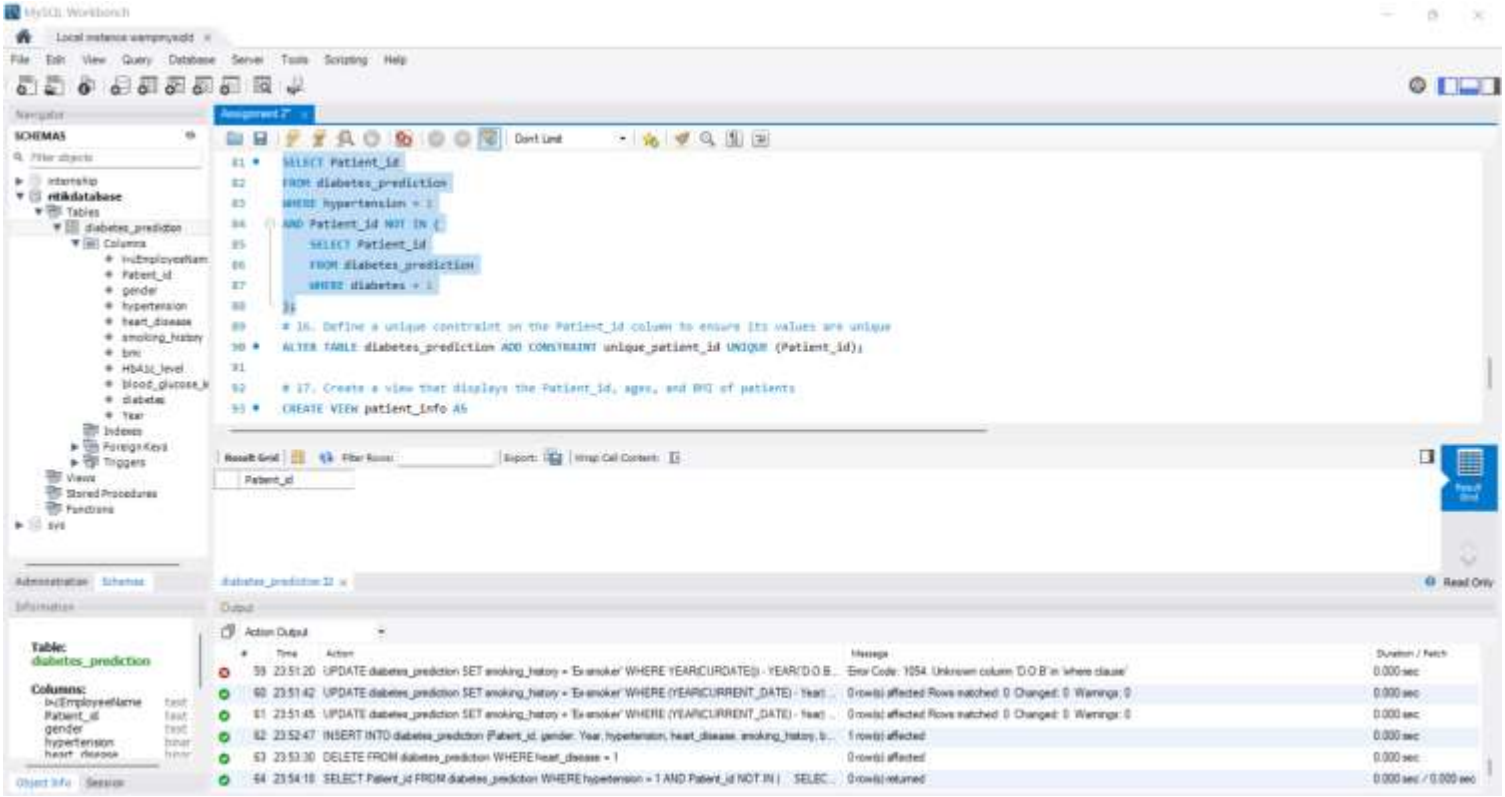
13) Insert a new patient into the database with sample data.



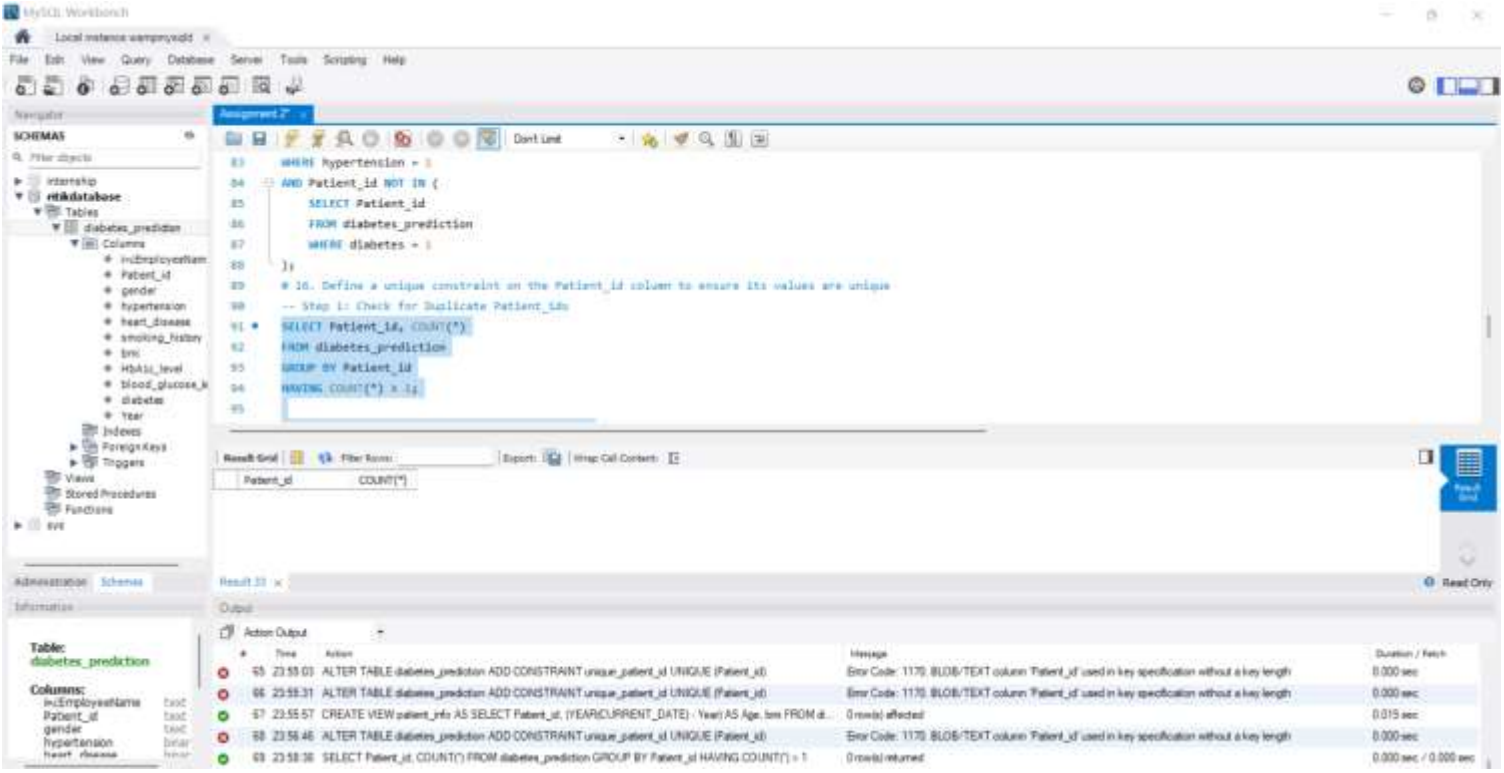
14) Delete all patients with heart disease from the database.



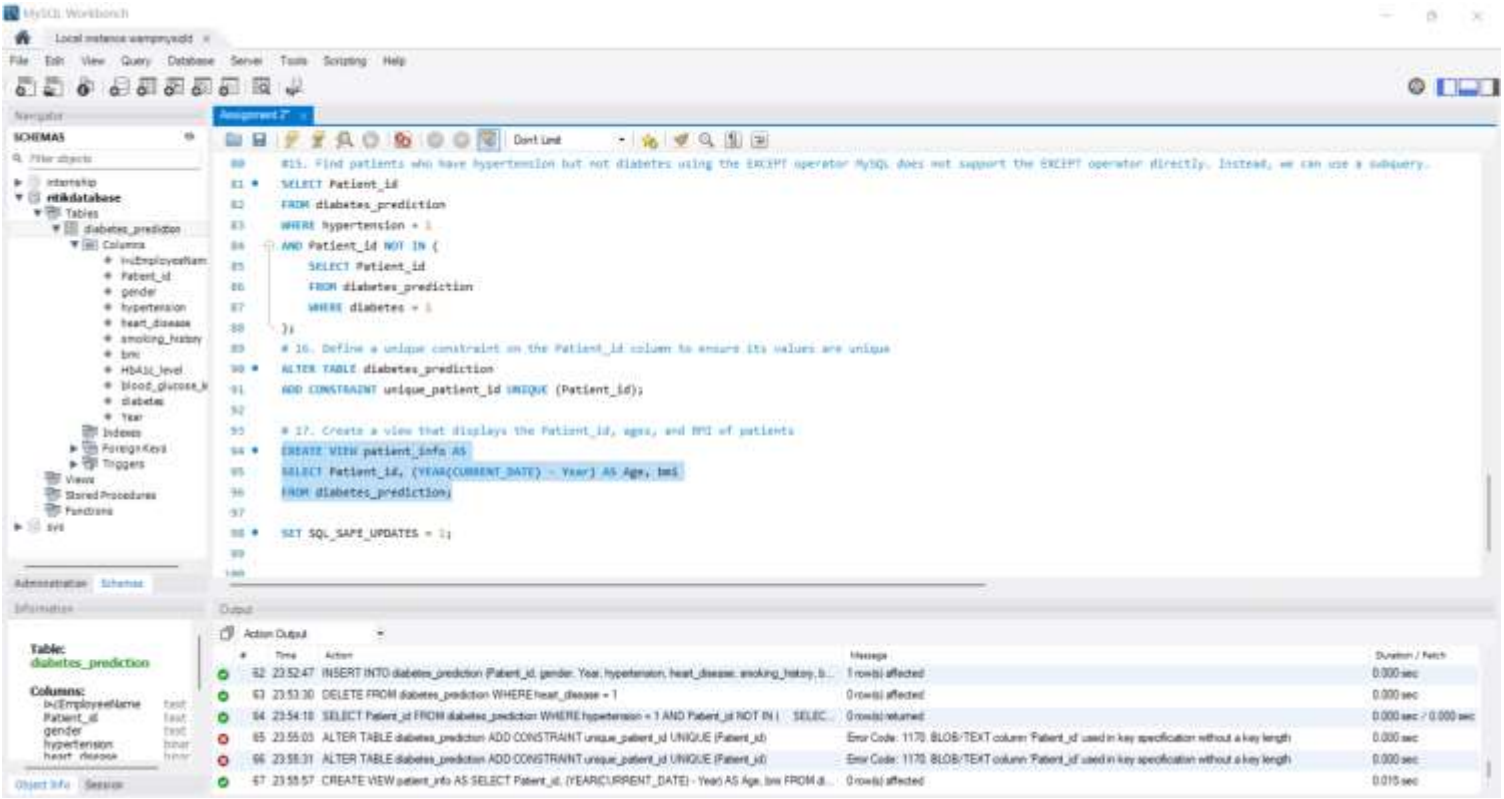
15) Find patients who have hypertension but not diabetes using the EXCEPT operator.



16) Define a unique constraint on the "patient_id" column to ensure its values are unique.



17) Define a unique constraint on the "patient_id" column to ensure its values are unique.



18. Suggest improvements in the database schema to reduce data redundancy and improve data integrity

To reduce data redundancy and improve data integrity, consider the following improvements:

- Normalize the database by splitting it into multiple related tables. For instance:
    - Create a patients table to store personal information (e.g., Patient_id, gender, Year).
    - Create a medical_records table to store medical information (e.g., hypertension, heart_disease, smoking_history, bmi, HbA1c_level, blood_glucose_level, diabetes).
- Use foreign keys to link these tables.
- Ensure that columns such as Patient_id are indexed for faster lookups.
- Use appropriate data types for each column (e.g., DATE for D.O.B).

19. Explain how you can optimize the performance of SQL queries on this dataset

To optimize the performance of SQL queries on this dataset:

- Index columns that are frequently used in WHERE clauses (e.g., Patient_id, Year).
- Use appropriate data types to minimize storage space and improve query performance.
- Avoid using SELECT * in production queries; instead, specify the needed columns.
- Use JOINs efficiently and ensure that tables are properly normalized.
- Regularly update statistics and optimize the database to ensure efficient query plans.