PSYLIQ

Diabetes Prediction

Internship Project [PSYLIQ]

Agenda

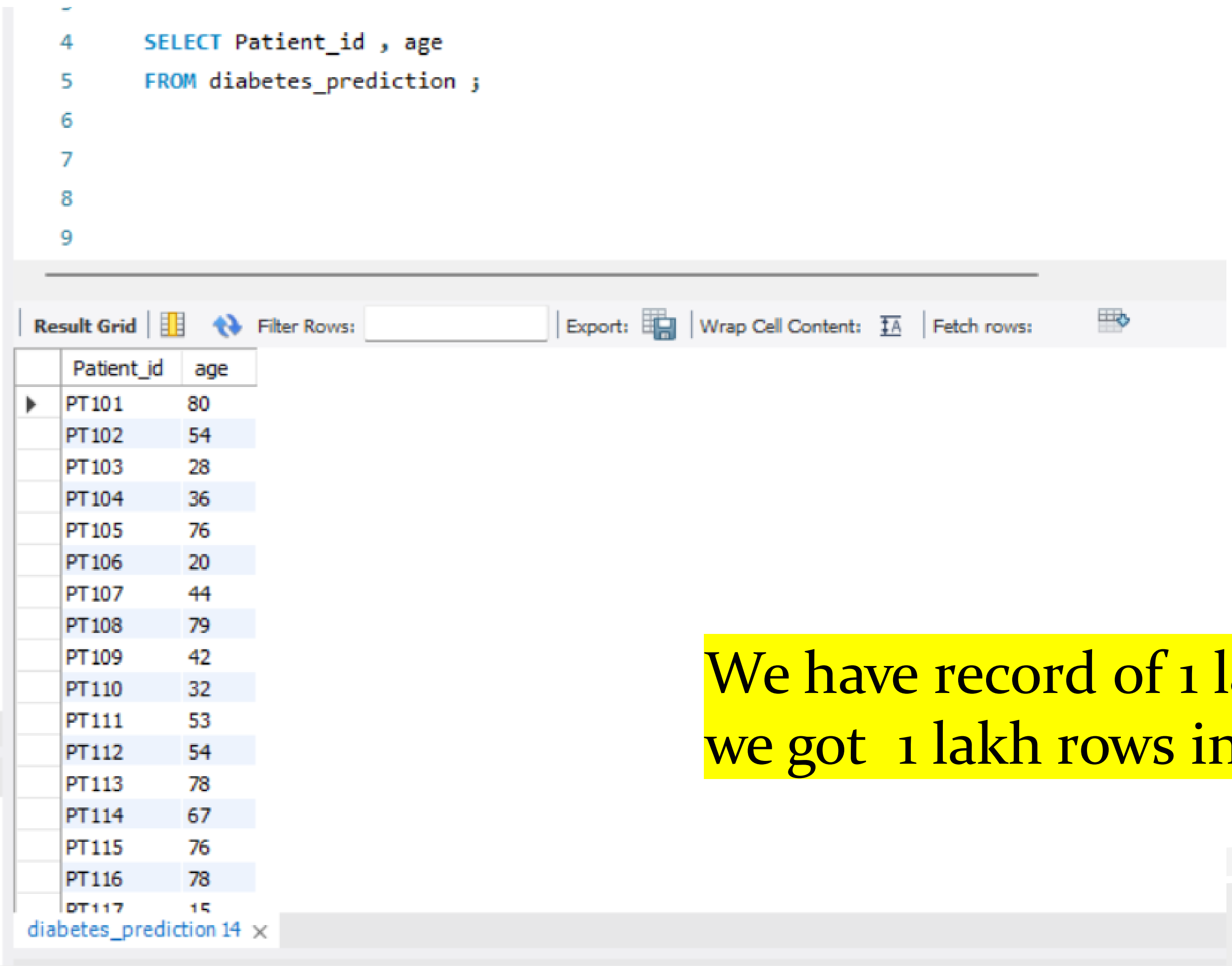
- Introduction
- Analysis
- Summary

Introduction

This dataset is provided by PSYLIQ , In this project, I will use the Diabetes Prediction data set to explore various aspects Diabetes and how they affect patients. The data set contains information about 100000 patients which are diabetes patients and their details such as Patient_id,gender, age, hypertension, heart_disease,smoking_history, bmi, HbA1c_level blood_glucose_level, diabetes .

I will use MSSQL. I will perform data cleaning, data exploration, data visualization, data modelling, and data interpretation. I will also present my findings and insights in a clear and concise report.

1. Retrieve the Patient_id and ages of all patients.

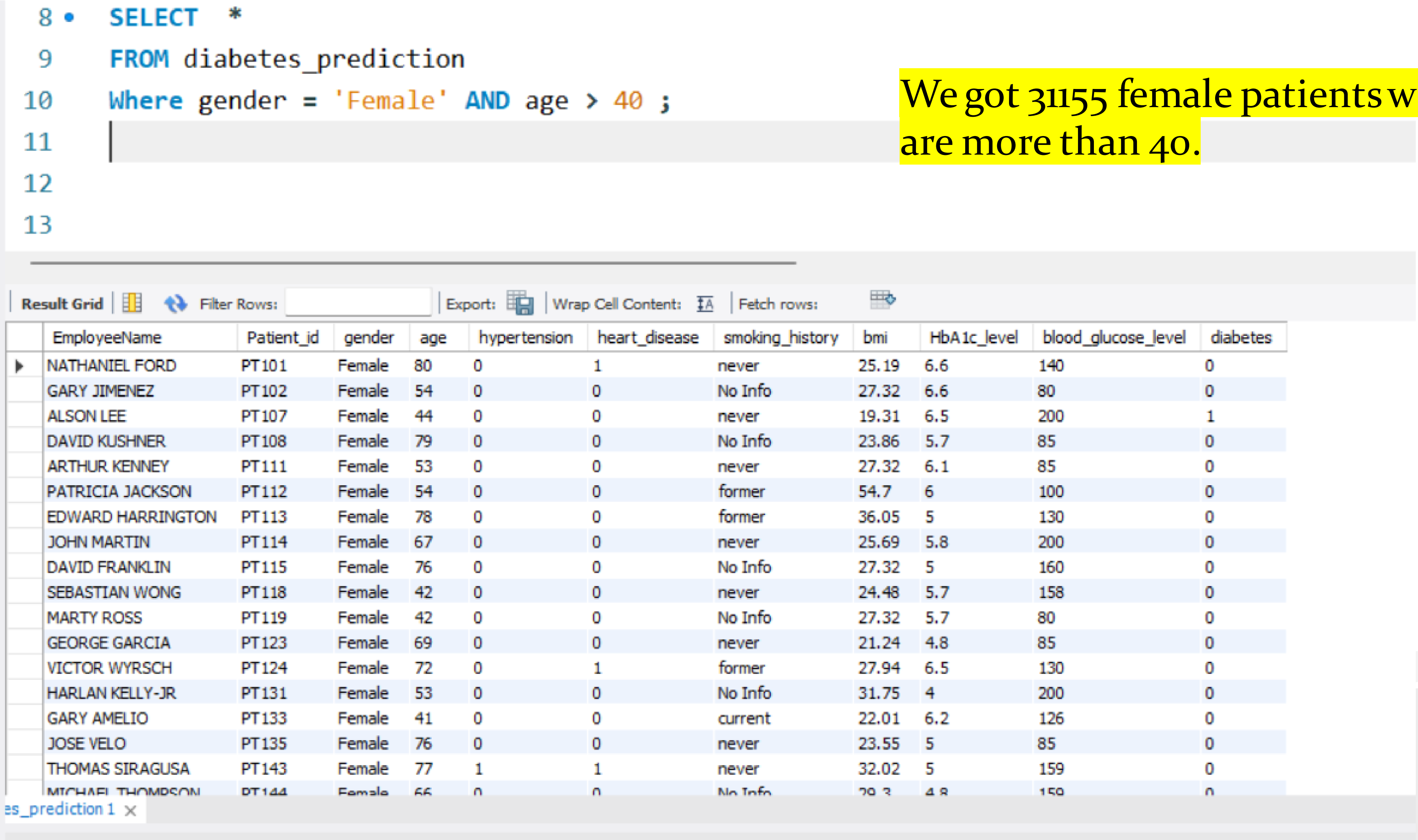


```sql
SELECT Patient_id , age
FROM diabetes_prediction ;
```

Patient_id	age
PT101	80
PT102	54
PT103	28
PT104	36
PT105	76
PT106	20
PT107	44
PT108	79
PT109	42
PT110	32
PT111	53
PT112	54
PT113	78
PT114	67
PT115	76
PT116	78
PT117	15

We have record of 1 lakh patient so we got 1 lakh rows in result.

2. Select all female patients who are older than 40.

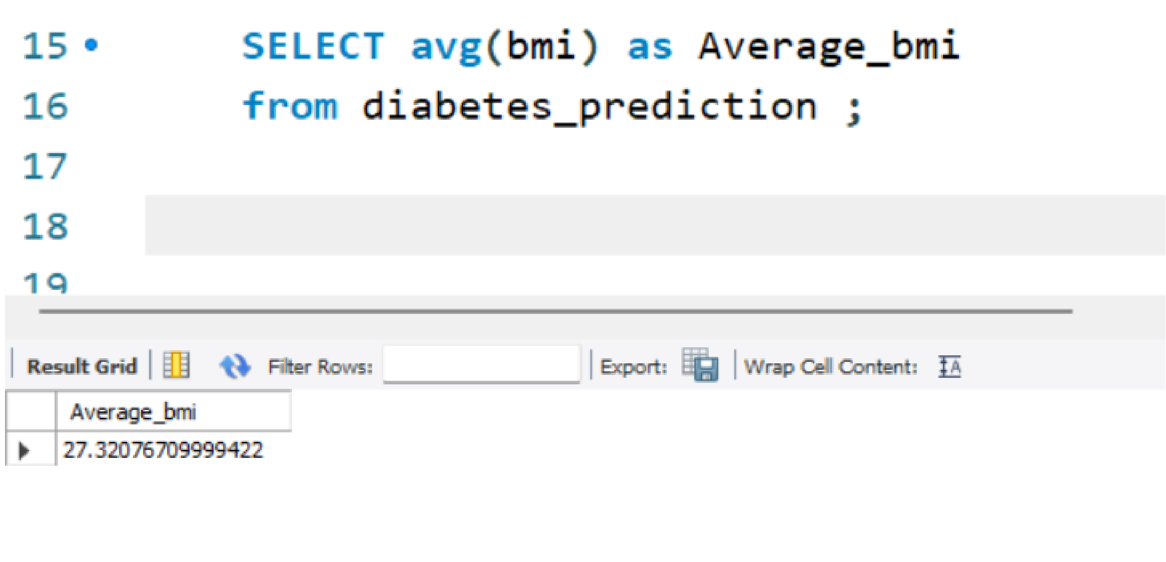


```
SELECT  *
FROM diabetes_prediction
Where gender = 'Female' AND age > 40 ;
```

EmployeeName	Patient_id	gender	age	hypertension	heart_disease	smoking_history	bmi	HbA1c_level	blood_glucose_level	diabetes
NATHANIEL FORD	PT101	Female	80	0	1	never	25.19	6.6	140	0
GARY JIMENEZ	PT102	Female	54	0	0	No Info	27.32	6.6	80	0
ALSON LEE	PT107	Female	44	0	0	never	19.31	6.5	200	1
DAVID KUSHNER	PT108	Female	79	0	0	No Info	23.86	5.7	85	0
ARTHUR KENNEY	PT111	Female	53	0	0	never	27.32	6.1	85	0
PATRICIA JACKSON	PT112	Female	54	0	0	former	54.7	6	100	0
EDWARD HARRINGTON	PT113	Female	78	0	0	former	36.05	5	130	0
JOHN MARTIN	PT114	Female	67	0	0	never	25.69	5.8	200	0
DAVID FRANKLIN	PT115	Female	76	0	0	No Info	27.32	5	160	0
SEBASTIAN WONG	PT118	Female	42	0	0	never	24.48	5.7	158	0
MARTY ROSS	PT119	Female	42	0	0	No Info	27.32	5.7	80	0
GEORGE GARCIA	PT123	Female	69	0	0	never	21.24	4.8	85	0
VICTOR WYRSCH	PT124	Female	72	0	1	former	27.94	6.5	130	0
HARLAN KELLY-JR	PT131	Female	53	0	0	No Info	31.75	4	200	0
GARY AMELIO	PT133	Female	41	0	0	current	22.01	6.2	126	0
JOSE VELO	PT135	Female	76	0	0	never	23.55	5	85	0
THOMAS SIRAGUSA	PT143	Female	77	1	1	never	32.02	5	159	0
MICHAEL THOMPSON	PT144	Female	66	0	0	No Info	29.3	4.8	159	0

We got 31155 female patients whom age are more than 40.

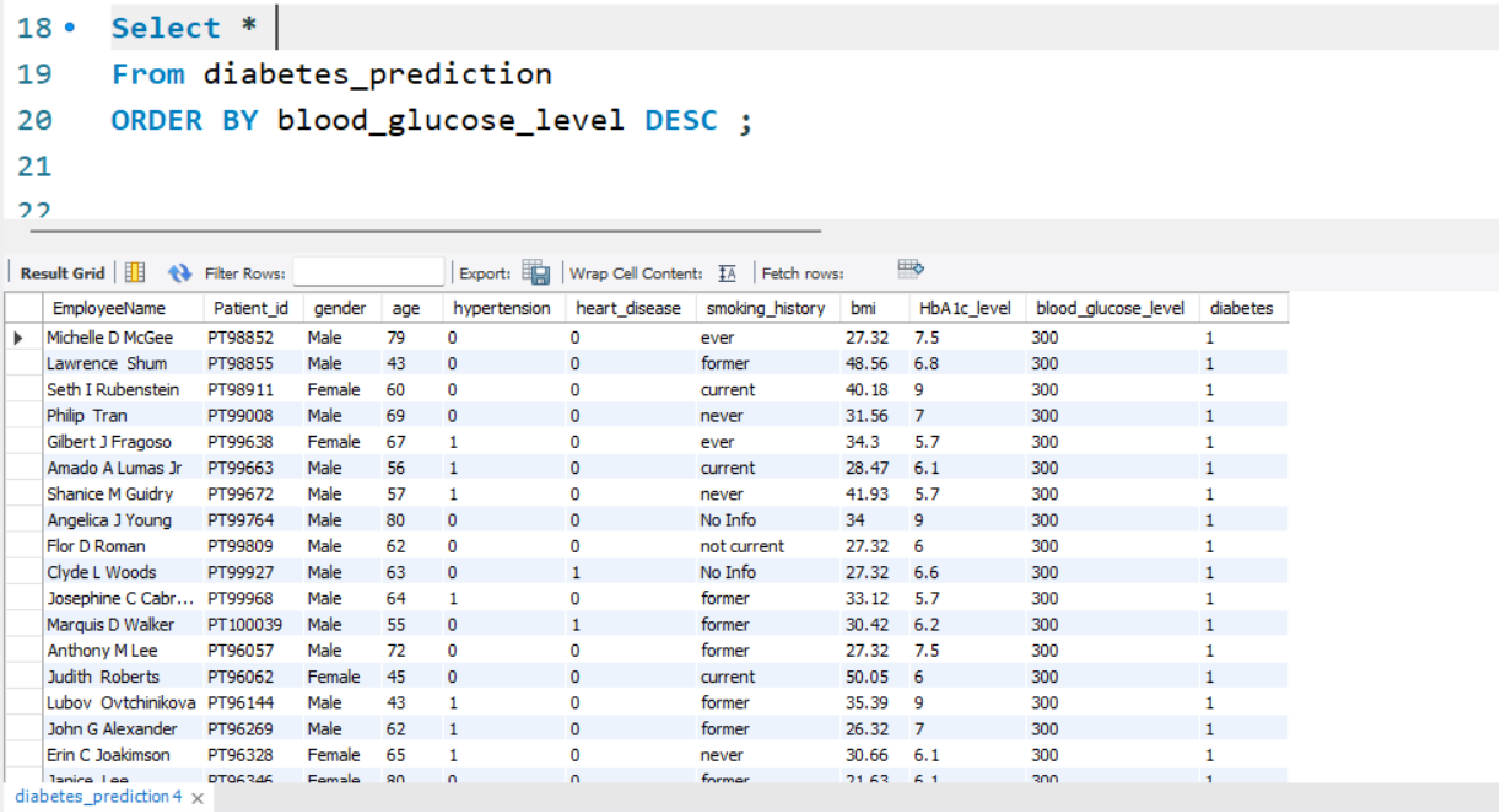
3. Calculate the average BMI of patients.



```sql
SELECT avg(bmi) as Average_bmi
from diabetes_prediction ;
```

```
Average_bmi
27.32076709999422
```

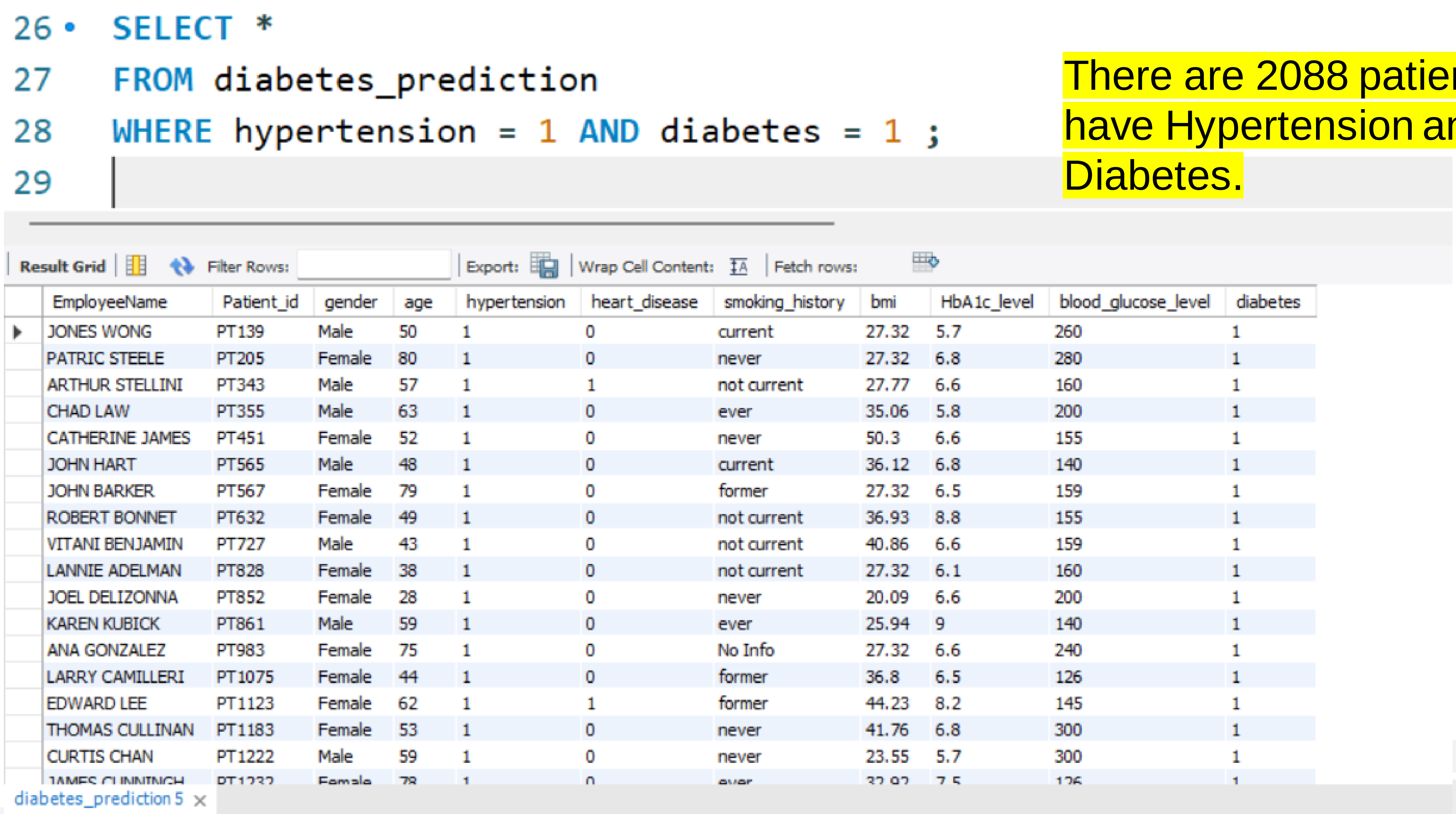
4. List patients in descending order of blood glucose levels.



```sql
Select *
From diabetes_prediction
ORDER BY blood_glucose_level DESC ;
```

EmployeeName	Patient_id	gender	age	hypertension	heart_disease	smoking_history	bmi	HbA1c_level	blood_glucose_level	diabetes
Michelle D McGee	PT98852	Male	79	0	0	ever	27.32	7.5	300	1
Lawrence Shum	PT98855	Male	43	0	0	former	48.56	6.8	300	1
Seth I Rubenstein	PT98911	Female	60	0	0	current	40.18	9	300	1
Philip Tran	PT99008	Male	69	0	0	never	31.56	7	300	1
Gilbert J Fragoso	PT99638	Female	67	1	0	ever	34.3	5.7	300	1
Amado A Lumas Jr	PT99663	Male	56	1	0	current	28.47	6.1	300	1
Shanice M Guidry	PT99672	Male	57	1	0	never	41.93	5.7	300	1
Angelica J Young	PT99764	Male	80	0	0	No Info	34	9	300	1
Flor D Roman	PT99809	Male	62	0	0	not current	27.32	6	300	1
Clyde L Woods	PT99927	Male	63	0	1	No Info	27.32	6.6	300	1
Josephine C Cabr...	PT99968	Male	64	1	0	former	33.12	5.7	300	1
Marquis D Walker	PT100039	Male	55	0	1	former	30.42	6.2	300	1
Anthony M Lee	PT96057	Male	72	0	0	former	27.32	7.5	300	1
Judith Roberts	PT96062	Female	45	0	0	current	50.05	6	300	1
Lubov Ovtchinikova	PT96144	Male	43	1	0	former	35.39	9	300	1
John G Alexander	PT96269	Male	62	1	0	former	26.32	7	300	1
Erin C Joakimson	PT96328	Female	65	1	0	never	30.66	6.1	300	1
Janice Lee	PT96346	Female	80	0	0	former	21.63	6.1	300	1

5. Find patients who have hypertension and diabetes.



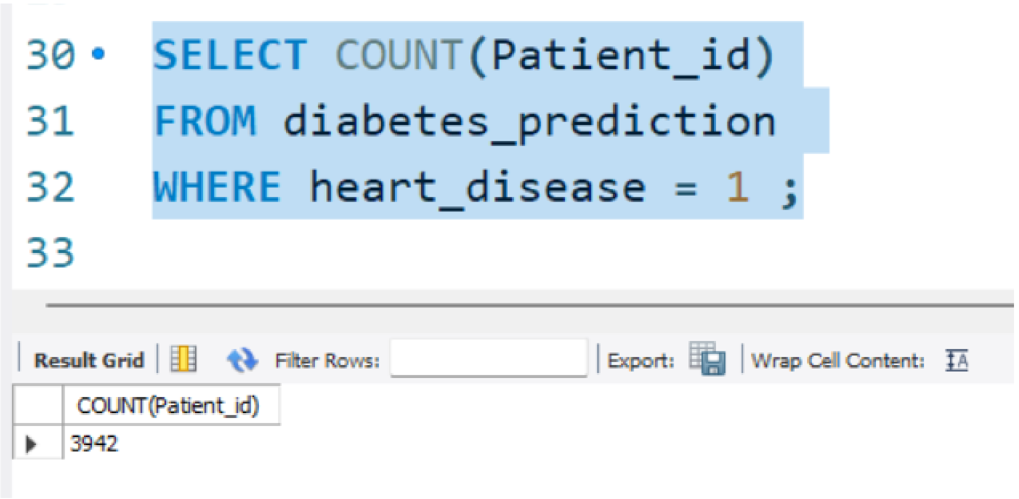
```sql
SELECT *
FROM diabetes_prediction
WHERE hypertension = 1 AND diabetes = 1 ;

```

EmployeeName	Patient_id	gender	age	hypertension	heart_disease	smoking_history	bmi	HbA1c_level	blood_glucose_level	diabetes
JONES WONG	PT139	Male	50	1	0	current	27.32	5.7	260	1
PATRIC STEELE	PT205	Female	80	1	0	never	27.32	6.8	280	1
ARTHUR STELLINI	PT343	Male	57	1	1	not current	27.77	6.6	160	1
CHAD LAW	PT355	Male	63	1	0	ever	35.06	5.8	200	1
CATHERINE JAMES	PT451	Female	52	1	0	never	50.3	6.6	155	1
JOHN HART	PT565	Male	48	1	0	current	36.12	6.8	140	1
JOHN BARKER	PT567	Female	79	1	0	former	27.32	6.5	159	1
ROBERT BONNET	PT632	Female	49	1	0	not current	36.93	8.8	155	1
VITANI BENJAMIN	PT727	Male	43	1	0	not current	40.86	6.6	159	1
LANNIE ADELMAN	PT828	Female	38	1	0	not current	27.32	6.1	160	1
JOEL DELIZONNA	PT852	Female	28	1	0	never	20.09	6.6	200	1
KAREN KUBICK	PT861	Male	59	1	0	ever	25.94	9	140	1
ANA GONZALEZ	PT983	Female	75	1	0	No Info	27.32	6.6	240	1
LARRY CAMILLERI	PT1075	Female	44	1	0	former	36.8	6.5	126	1
EDWARD LEE	PT1123	Female	62	1	1	former	44.23	8.2	145	1
THOMAS CULLINAN	PT1183	Female	53	1	0	never	41.76	6.8	300	1
CURTIS CHAN	PT1222	Male	59	1	0	never	23.55	5.7	300	1
JAMES CUNNINGH	PT1232	Female	78	1	0	ever	32.92	7.5	126	1

There are 2088 patient who have Hypertension and Diabetes.

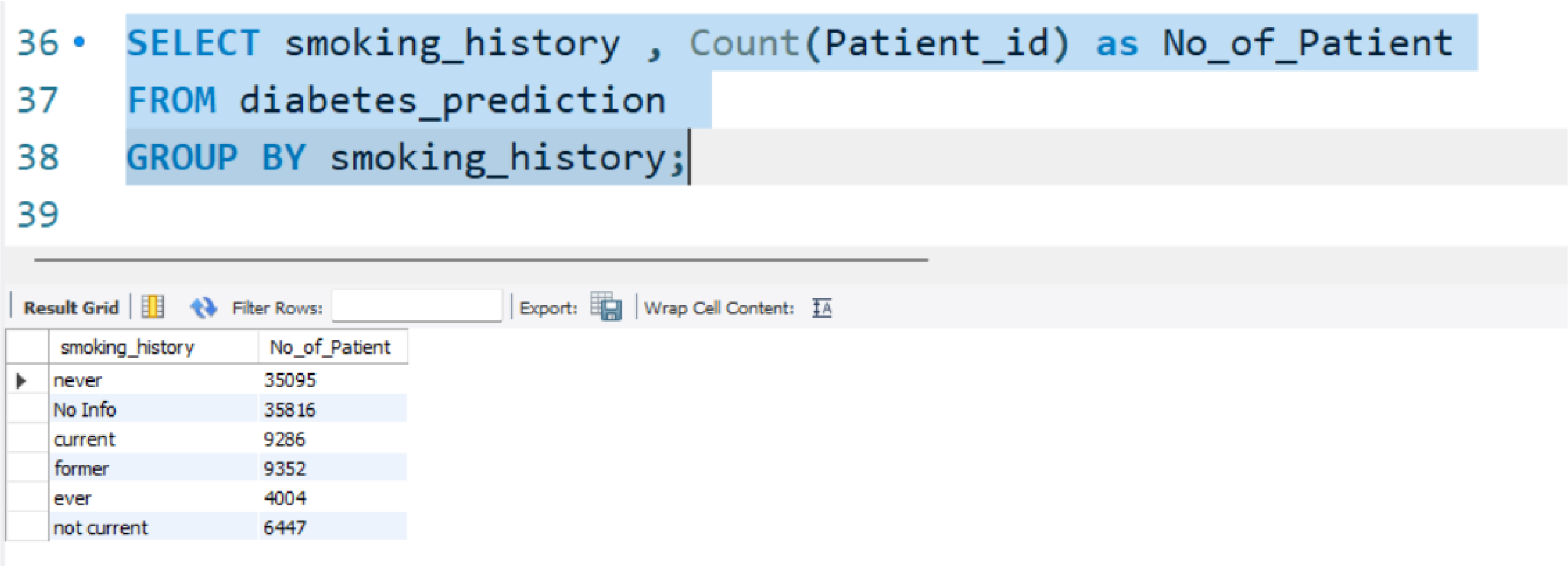
6. Determine the number of patients with heart disease.



```sql
SELECT COUNT(Patient_id)
FROM diabetes_prediction
WHERE heart_disease = 1 ;
```

```
COUNT(Patient_id)
3942
```

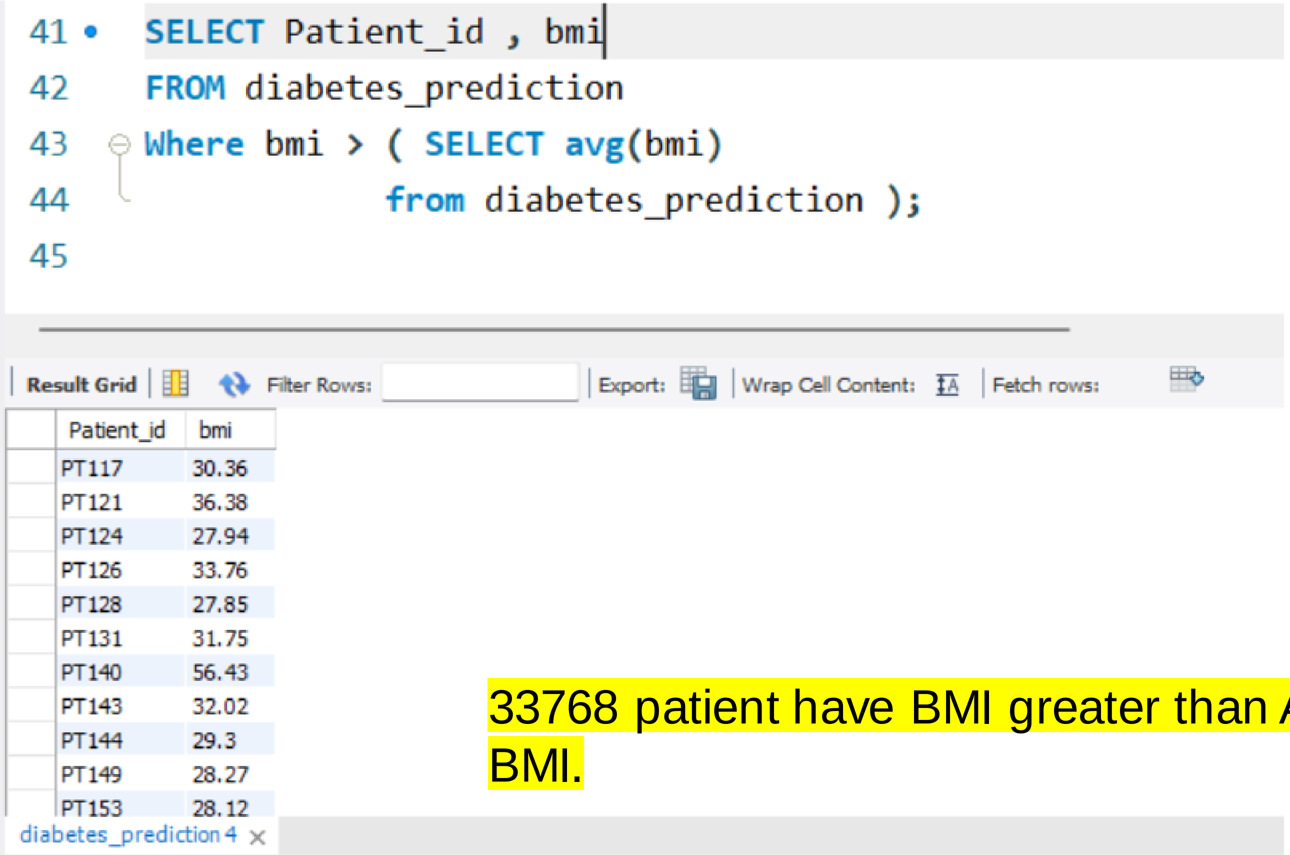
7. Group patients by smoking history and count how many smokers and nonsmokers there are.



```sql
SELECT smoking_history , Count(Patient_id) as No_of_Patient
FROM diabetes_prediction
GROUP BY smoking_history;
```

smoking_history	No_of_Patient
never	35095
No Info	35816
current	9286
former	9352
ever	4004
not current	6447

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

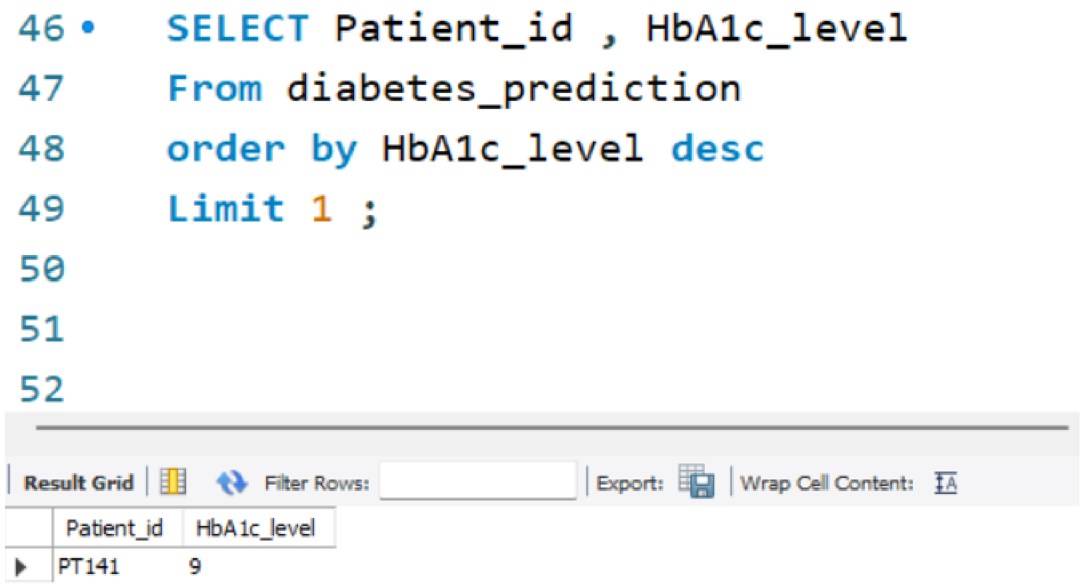


```
SELECT Patient_id , bmi
FROM diabetes_prediction
Where bmi > ( SELECT avg(bmi)
            from diabetes_prediction );
```

Patient_id	bmi
PT117	30.36
PT121	36.38
PT124	27.94
PT126	33.76
PT128	27.85
PT131	31.75
PT140	56.43
PT143	32.02
PT144	29.3
PT149	28.27
PT153	28.12

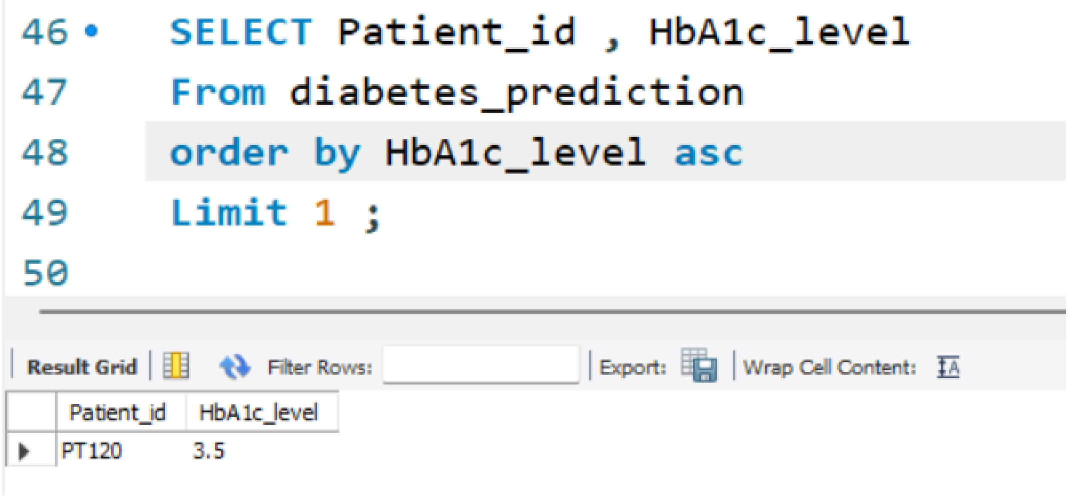
33768 patient have BMI greater than Avg BMI.

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



```
SELECT Patient_id , HbA1c_level
From diabetes_prediction
order by HbA1c_level desc
Limit 1 ;
```

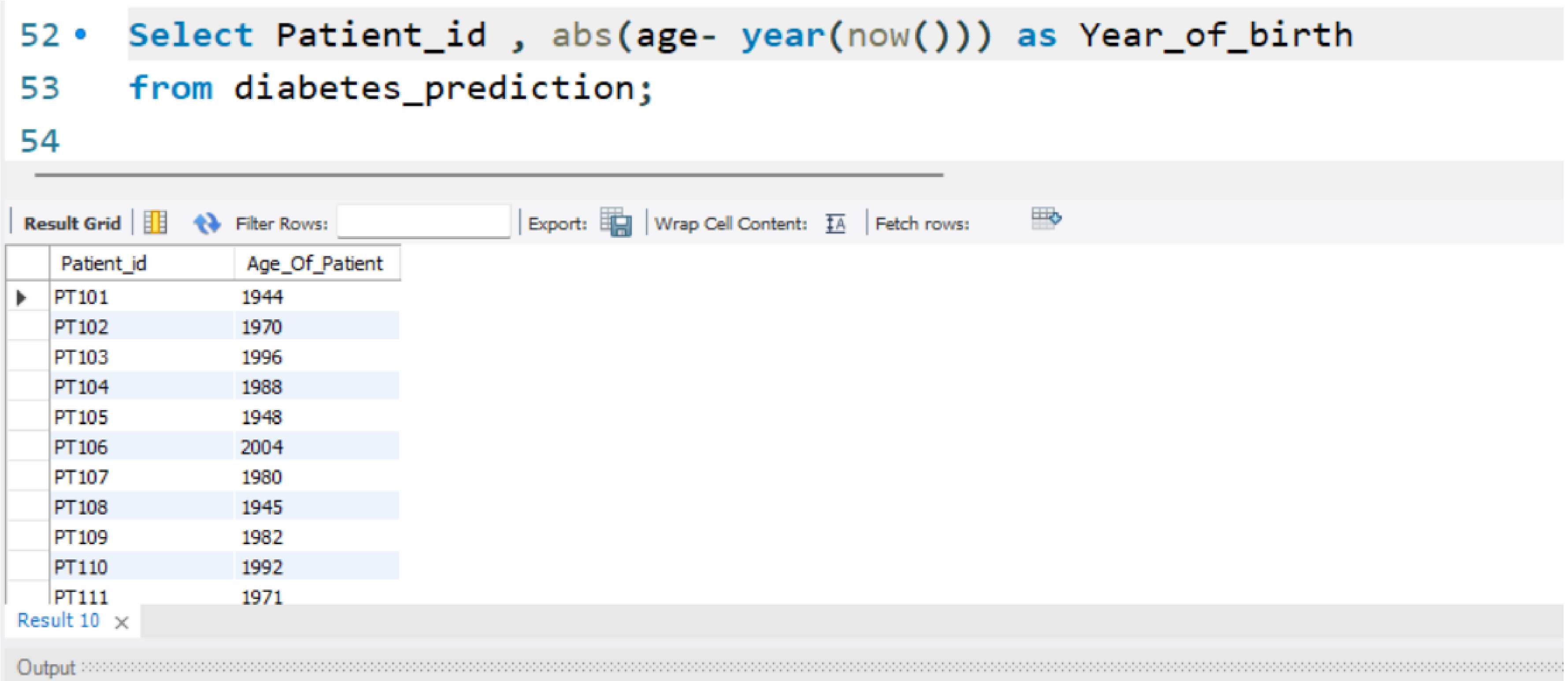
Patient_id	HbA1c_level
PT141	9



```
SELECT Patient_id , HbA1c_level
From diabetes_prediction
order by HbA1c_level asc
Limit 1 ;
```

Patient_id	HbA1c_level
PT120	3.5

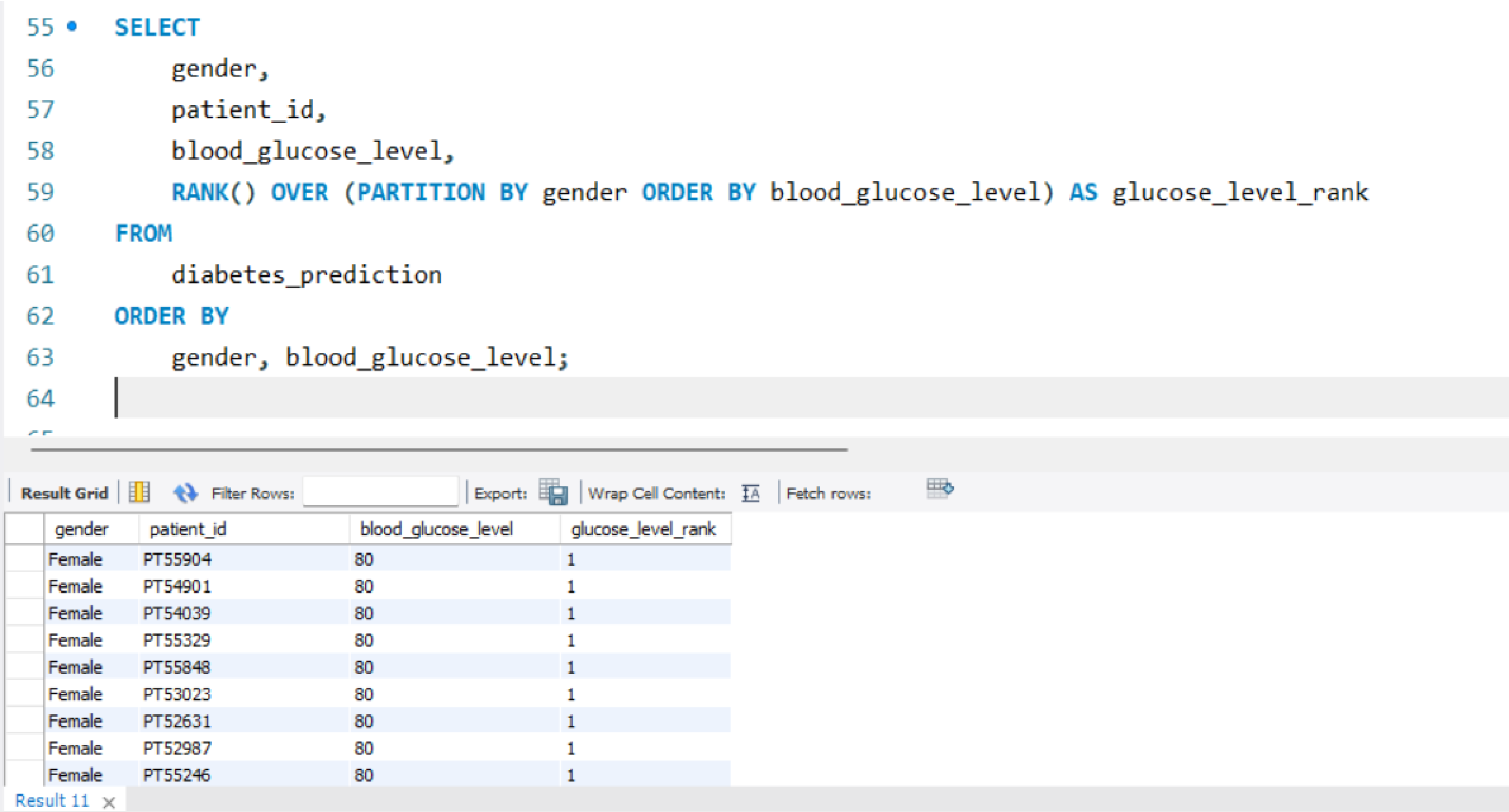
10. Calculate the age of patients in years (assuming the current date as of now).



```sql
Select Patient_id , abs(age- year(now())) as Year_of_birth
from diabetes_prediction;
```

Patient_id	Age_Of_Patient
PT101	1944
PT102	1970
PT103	1996
PT104	1988
PT105	1948
PT106	2004
PT107	1980
PT108	1945
PT109	1982
PT110	1992
PT111	1971

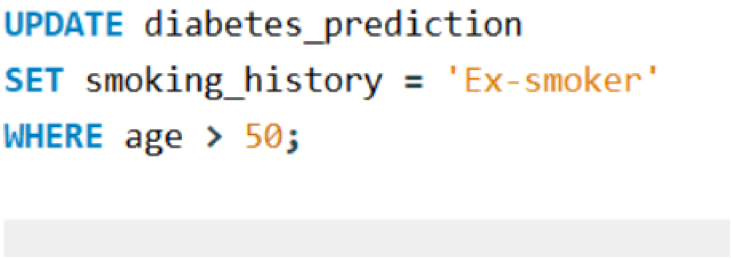
11. Rank patients by blood glucose level within each gender group.



```sql
SELECT
    gender,
    patient_id,
    blood_glucose_level,
    RANK() OVER (PARTITION BY gender ORDER BY blood_glucose_level) AS glucose_level_rank
FROM
    diabetes_prediction
ORDER BY
    gender, blood_glucose_level;
```

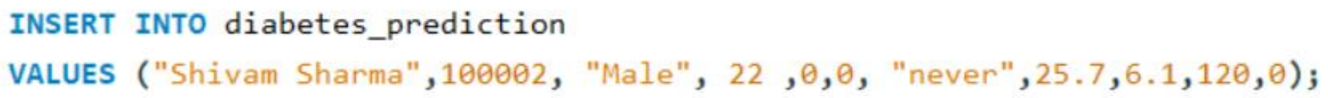
gender	patient_id	blood_glucose_level	glucose_level_rank
Female	PT55904	80	1
Female	PT54901	80	1
Female	PT54039	80	1
Female	PT55329	80	1
Female	PT55848	80	1
Female	PT53023	80	1
Female	PT52631	80	1
Female	PT52987	80	1
Female	PT55246	80	1

12. Update the smoking history of patients who are older than 50 to "Ex-smoker."



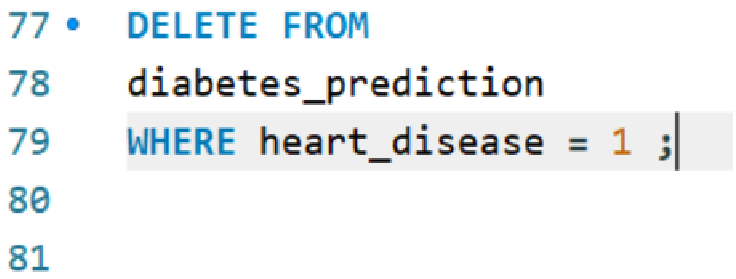
```sql
UPDATE diabetes_prediction
SET smoking_history = 'Ex-smoker'
WHERE age > 50;
```

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



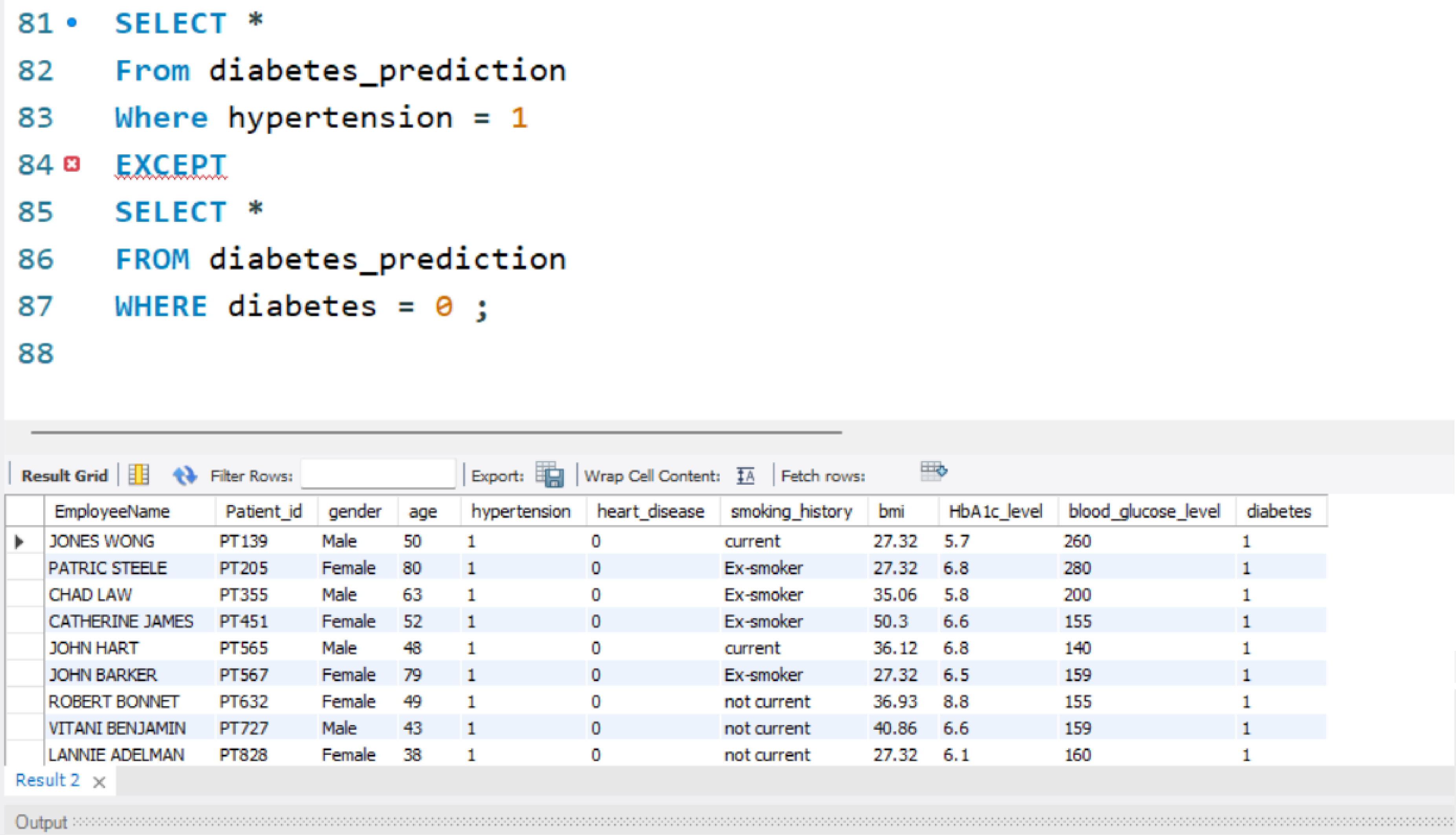
```
INSERT INTO diabetes_prediction
VALUES ("Shivam Sharma",100002, "Male", 22 ,0,0, "never",25.7,6.1,120,0);
```

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



```sql
DELETE FROM
diabetes_prediction
WHERE heart_disease = 1 ;
```

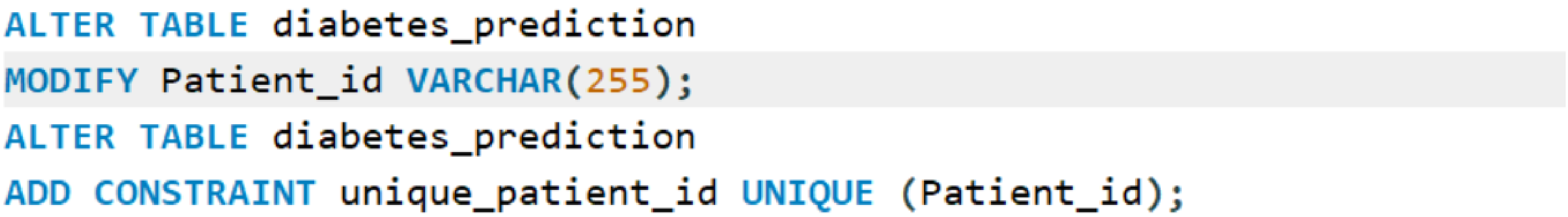
15. Find patients who have hypertension but not diabetes using the EXCEPT operator. 2 of 2



```
SELECT *
From diabetes_prediction
Where hypertension = 1
EXCEPT
SELECT *
FROM diabetes_prediction
WHERE diabetes = 0 ;
```

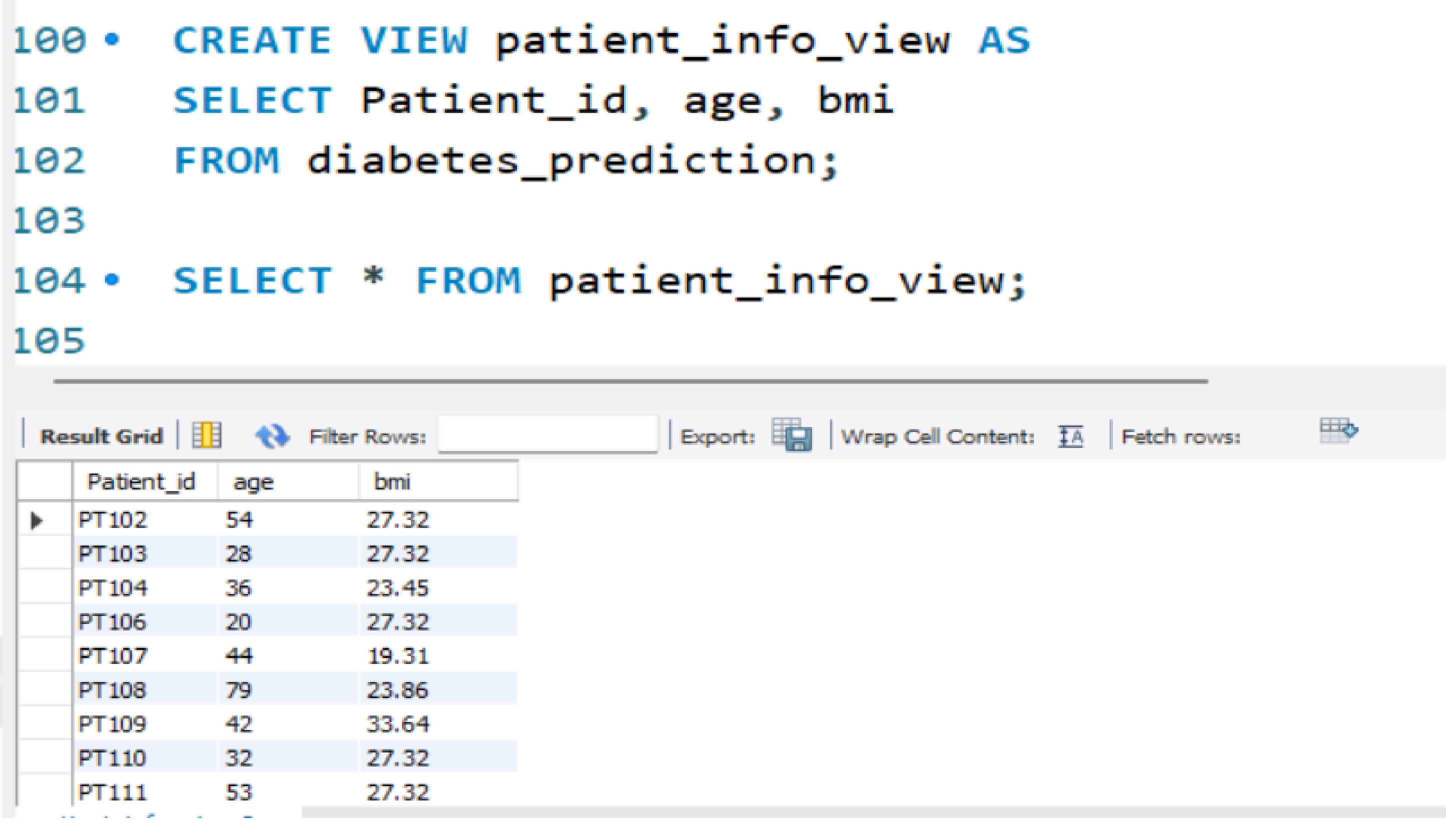
EmployeeName	Patient_id	gender	age	hypertension	heart_disease	smoking_history	bmi	HbA1c_level	blood_glucose_level	diabetes
JONES WONG	PT139	Male	50	1	0	current	27.32	5.7	260	1
PATRIC STEELE	PT205	Female	80	1	0	Ex-smoker	27.32	6.8	280	1
CHAD LAW	PT355	Male	63	1	0	Ex-smoker	35.06	5.8	200	1
CATHERINE JAMES	PT451	Female	52	1	0	Ex-smoker	50.3	6.6	155	1
JOHN HART	PT565	Male	48	1	0	current	36.12	6.8	140	1
JOHN BARKER	PT567	Female	79	1	0	Ex-smoker	27.32	6.5	159	1
ROBERT BONNET	PT632	Female	49	1	0	not current	36.93	8.8	155	1
VITANI BENJAMIN	PT727	Male	43	1	0	not current	40.86	6.6	159	1
LANNIE ADELMAN	PT828	Female	38	1	0	not current	27.32	6.1	160	1

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



```
ALTER TABLE diabetes_prediction
MODIFY Patient_id VARCHAR(255);
ALTER TABLE diabetes_prediction
ADD CONSTRAINT unique_patient_id UNIQUE (Patient_id);
```

17. Create a view that displays the Patient_ids, ages, and BMI of patients.



```sql
CREATE VIEW patient_info_view AS
SELECT Patient_id, age, bmi
FROM diabetes_prediction;

SELECT * FROM patient_info_view;
```

```
  Patient_id  age   bmi
▶ PT102       54    27.32
  PT103       28    27.32
  PT104       36    23.45
  PT106       20    27.32
  PT107       44    19.31
  PT108       79    23.86
  PT109       42    33.64
  PT110       32    27.32
  PT111       53    27.32
```

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

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

1. Normalize the Database:
   - Break down tables into smaller, more manageable entities to reduce redundancy.
   - Use normalization techniques like First Normal Form (1NF), Second Normal Form (2NF), and Third Normal Form (3NF) to eliminate data duplication.
   - Create separate tables for related data to avoid storing the same information in multiple places.

2. Use Foreign Keys:
   - Implement foreign keys to establish relationships between tables and ensure referential integrity.
   - Define foreign keys to link primary keys in one table to corresponding columns in another table.
   - This helps maintain consistency and prevents orphan records.

3. Define Constraints:
   - Utilize constraints like NOT NULL, UNIQUE, and CHECK constraints to enforce data integrity rules.
   - Specify constraints at the column level to restrict the type of data that can be stored.
   - Constraints help prevent invalid data from being inserted into the database.

4. Avoid Redundant Columns:
   - Identify and remove redundant columns that store the same information in multiple tables.
   - Store data in a single location and reference it using foreign keys instead of duplicating it across tables.

5. Use Views:
   - Create views to present data from multiple tables in a consolidated format without duplicating the underlying data.
   - Views can help simplify queries and provide a consistent view of the data to users.

6. Implement Indexes:
   - Create indexes on columns frequently used in queries to improve query performance.
   - Indexes help speed up data retrieval by allowing the database engine to quickly locate relevant rows.

7. Review Data Types:
   - Choose appropriate data types for columns to ensure efficient storage and accurate representation of data.
   - Avoid using generic data types that can lead to data inconsistencies or unnecessary storage space.

By implementing these improvements in the database schema, you can enhance data integrity, reduce redundancy, and optimize the overall structure of the database

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

To optimize the performance of SQL queries on a dataset, consider the following strategies:

1. Use Indexes:
- Create indexes on columns frequently used in WHERE clauses, JOIN conditions, and ORDER BY clauses.
- Indexes help the database engine quickly locate relevant rows, improving query performance.
- However, be cautious not to over-index as it can impact insert and update operations.

2. Optimize Query Structure:
- Write efficient queries by avoiding unnecessary JOINs, subqueries, and complex logic.
- Use EXPLAIN or query execution plans to analyze query performance and identify areas for optimization.
- Consider breaking down complex queries into smaller, more manageable parts.

3. Limit Result Sets:
- Use the LIMIT keyword to restrict the number of rows returned by a query.
- Fetch only the necessary columns instead of retrieving all columns in the SELECT statement.
- Avoid using SELECT * as it can retrieve more data than needed.

4. Use Stored Procedures:
- Implement stored procedures to encapsulate frequently executed queries or business logic.
- Stored procedures can reduce network traffic and improve performance by executing multiple SQL statements in a single call.

5. Avoid SELECT DISTINCT:
- Minimize the use of SELECT DISTINCT as it can be resource-intensive, especially on large datasets.
- Consider alternative approaches like using GROUP BY or refining the query logic to eliminate duplicates.

6. Update Statistics:
- Regularly update table statistics to provide the query optimizer with accurate information about data distribution.
- Outdated statistics can lead to suboptimal query plans and performance degradation.

7. Consider Partitioning:
- Implement table partitioning to divide large tables into smaller, more manageable partitions based on specific criteria (e.g., date ranges).
- Partitioning can improve query performance by reducing the amount of data scanned for each query.

8. Use Caching:
- Utilize caching mechanisms like query caching or application-level caching to store frequently accessed data in memory.
- Caching can reduce the need to repeatedly query the database for the same data, improving overall performance.

By implementing these optimization techniques, you can enhance the performance of SQL queries on your dataset and ensure efficient data retrieval and processing.

Thank You
40
50