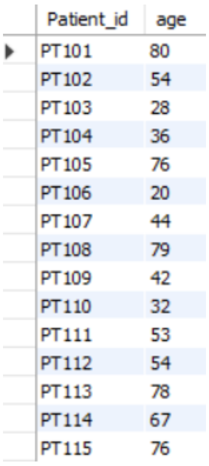
Diabetes Prediction

1) Retrieve the Patient_id and ages of all patients.

Query:

```
SELECT Patient_id, age AS age FROM diabetes_prediction LIMIT 0, 1000
```

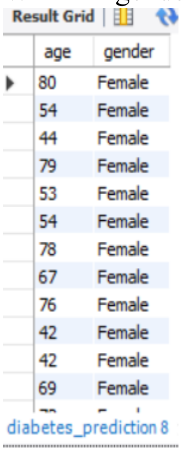


Patient_id	age
PT101	80
PT102	54
PT103	28
PT104	36
PT105	76
PT106	20
PT107	44
PT108	79
PT109	42
PT110	32
PT111	53
PT112	54
PT113	78
PT114	67
PT115	76

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

Query:

```
SELECT age AS age ,gender as gender
FROM diabetes_prediction
WHERE gender = 'Female' AND age > 40;
```

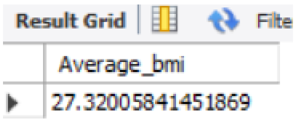


age	gender
80	Female
54	Female
44	Female
79	Female
53	Female
54	Female
78	Female
67	Female
76	Female
42	Female
42	Female
69	Female

3) Calculate the average BMI of patients.

Query:

```
SELECT AVG(BMI) AS Average_bmi
FROM diabetes_prediction;
```

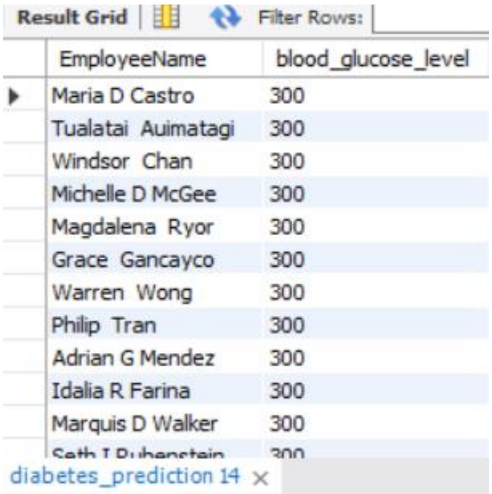


Average_bmi
27.32005841451869

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

Query:

```
SELECT EmployeeName,blood_glucose_level
FROM diabetes_prediction
ORDER BY blood_glucose_level DESC;
```

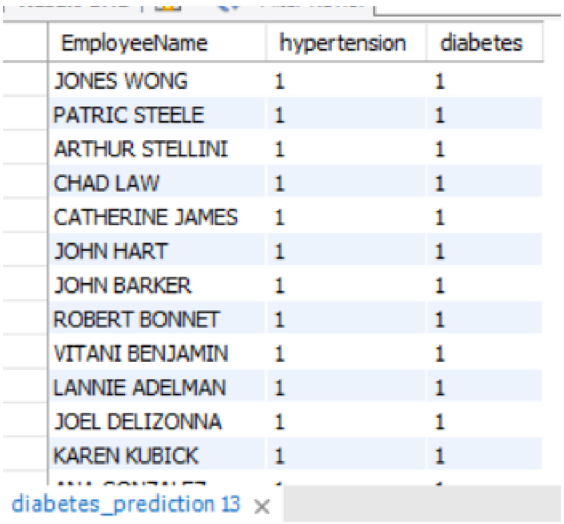


EmployeeName	blood_glucose_level
Maria D Castro	300
Tualatai Auimatagi	300
Windsor Chan	300
Michelle D McGee	300
Magdalena Ryor	300
Grace Gancayco	300
Warren Wong	300
Philip Tran	300
Adrian G Mendez	300
Idalia R Farina	300
Marquis D Walker	300
Seth I Rubenstein	300

5) Find patients who have hypertension and diabetes.

Query:

```
 SELECT EmployeeName, hypertension,diabetes
FROM diabetes_prediction
WHERE hypertension = 1 AND diabetes = 1;
```

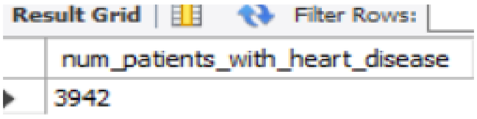


EmployeeName	hypertension	diabetes
JONES WONG	1	1
PATRIC STEELE	1	1
ARTHUR STELLINI	1	1
CHAD LAW	1	1
CATHERINE JAMES	1	1
JOHN HART	1	1
JOHN BARKER	1	1
ROBERT BONNET	1	1
VITANI BENJAMIN	1	1
LANNIE ADELMAN	1	1
JOEL DELIZONNA	1	1
KAREN KUBICK	1	1
ANA GONZALEZ	1	1

6) Determine the number of patients with heart disease.

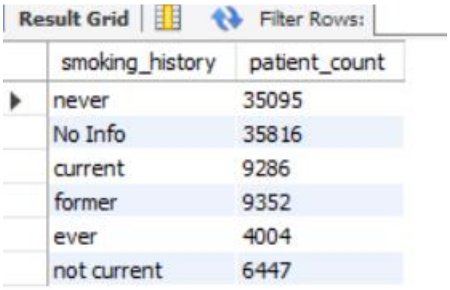
Query:

```
SELECT COUNT(*) AS num_patients_with_heart_disease
FROM diabetes_prediction
WHERE heart_disease = 1;
```



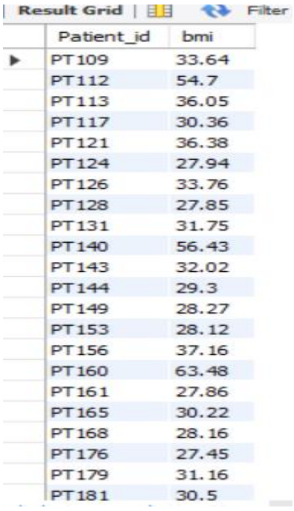
	num_patients_with_heart_disease
▶	3942

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



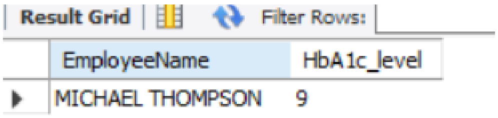
	smoking_history	patient_count
▶	never	35095
	No Info	35816
	current	9286
	former	9352
	ever	4004
	not current	6447

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



	Patient_id	bmi
▶	PT109	33.64
	PT112	54.7
	PT113	36.05
	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
	PT156	37.16
	PT160	63.48
	PT161	27.86
	PT165	30.22
	PT168	28.16
	PT176	27.45
	PT179	31.16
	PT181	30.5

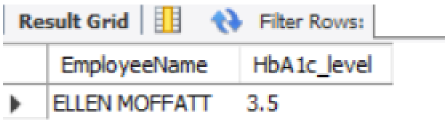
9) Find the patient with the highest HbA1c level and the patient with the lowest HbA1clevel.
Query:(for highest HbA1c level)
SELECT EmployeeName,HbA1c_level
FROM diabetes_prediction
ORDER BY HbA1c_level DESC
LIMIT 1;



EmployeeName	HbA1c_level
MICHAEL THOMPSON	9

Query: (for lowest HbA1c level.)

```
SELECT EmployeeName,HbA1c_level
FROM diabetes_prediction
ORDER BY HbA1c_level
LIMIT 1;
```

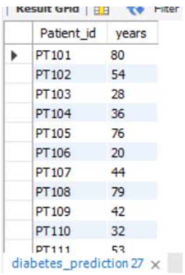


EmployeeName	HbA1c_level
ELLEN MOFFATT	3.5

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

Query:

```
SELECT Patient_id, age AS years
FROM diabetes_prediction;
```

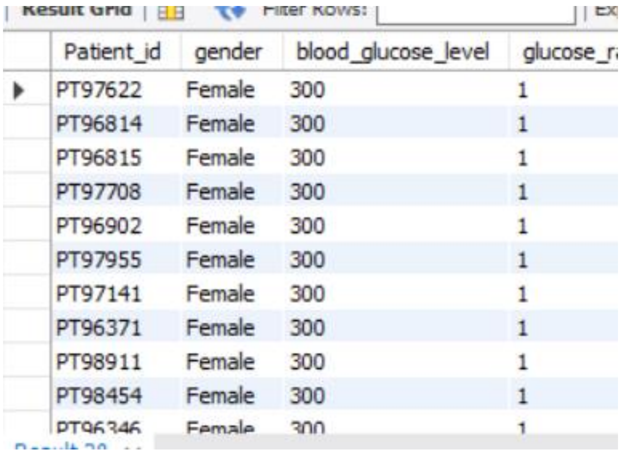


Patient_id	years
PT101	80
PT102	54
PT103	28
PT104	36
PT105	76
PT106	20
PT107	44
PT108	79
PT109	42
PT110	32
PT111	53

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

Query:

```
SELECT
  Patient_id,
  gender,
  blood_glucose_level,
  RANK() OVER (PARTITION BY gender ORDER BY blood_glucose_level DESC) AS glucose_rank
FROM diabetes_prediction;
```



Patient_id	gender	blood_glucose_level	glucose_r
PT97622	Female	300	1
PT96814	Female	300	1
PT96815	Female	300	1
PT97708	Female	300	1
PT96902	Female	300	1
PT97955	Female	300	1
PT97141	Female	300	1
PT96371	Female	300	1
PT98911	Female	300	1
PT98454	Female	300	1
PT96346	Female	300	1

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

Query:

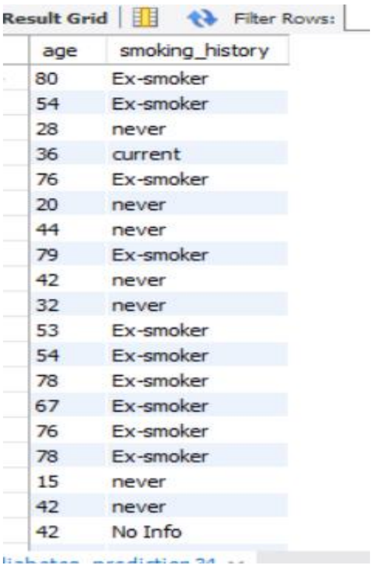
UPDATE diabetes_prediction

SET smoking_history = 'Ex-smoker'

WHERE age > 50

Here is the query to see it:

SELECT age ,smoking_history from diabetes_prediction LIMIT 0, 1000



age	smoking_history
80	Ex-smoker
54	Ex-smoker
28	never
36	current
76	Ex-smoker
20	never
44	never
79	Ex-smoker
42	never
32	never
53	Ex-smoker
54	Ex-smoker
78	Ex-smoker
67	Ex-smoker
76	Ex-smoker
78	Ex-smoker
15	never
42	never
42	No Info

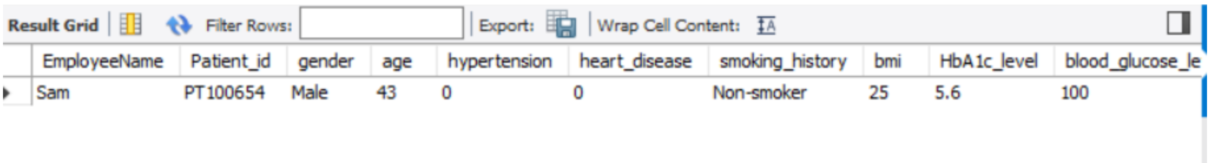
13)Insert a new patient into the database with sample data.

Query:

```
INSERT INTO diabetes_prediction (EmployeeName,Patient_id, gender, Age, smoking_history, BMI, blood_glucose_level, hypertension, diabetes, HbA1c_level, heart_disease)

VALUES

  ('Sam','PT100654', 'Male', '43', 'Non-smoker', 25.0, 100, 0, 0, 5.6, 0);
```

Here is the query to check this

```
SELECT * FROM diabetes_prediction

WHERE Patient_id = `PT100654';
```



EmployeeName	Patient_id	gender	age	hypertension	heart_disease	smoking_history	bmi	HbA1c_level	blood_glucose_le
Sam	PT100654	Male	43	0	0	Non-smoker	25	5.6	100

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

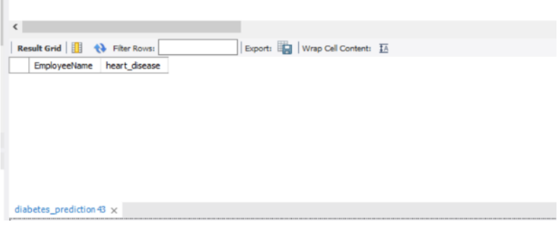
Query:

```
DELETE FROM diabetes_prediction

WHERE heart_disease = '1';
```

Here is the result after running this

Query:

```
select EmployeeName,heart_disease FROM diabetes_prediction

where heart_disease='1'
```



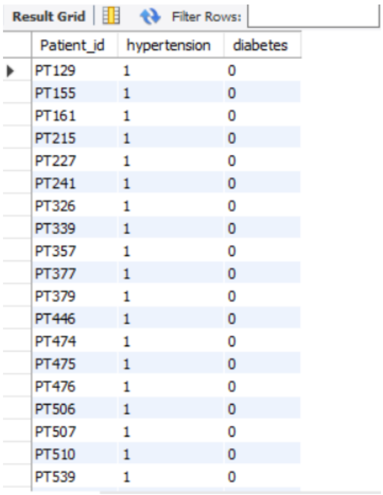
EmployeeName	heart_disease

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

2 of 2

Query:

```
SELECT Patient_id, hypertension, diabetes FROM diabetes_prediction

WHERE hypertension = '1'

EXCEPT

SELECT Patient_id, hypertension, diabetes FROM diabetes_prediction

WHERE diabetes = '1';
```



Patient_id	hypertension	diabetes
PT129	1	0
PT155	1	0
PT161	1	0
PT215	1	0
PT227	1	0
PT241	1	0
PT326	1	0
PT339	1	0
PT357	1	0
PT377	1	0
PT379	1	0
PT446	1	0
PT474	1	0
PT475	1	0
PT476	1	0
PT506	1	0
PT507	1	0
PT510	1	0
PT539	1	0

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

Query:

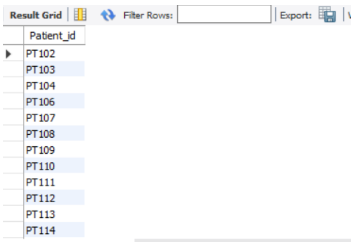
ALTER TABLE diabetes_prediction

MODIFY COLUMN patient_id VARCHAR(255);

Here is the result:

Query:

select Patient_id from diabetes_prediction



Patient_id
PT102
PT103
PT104
PT106
PT107
PT108
PT109
PT110
PT111
PT112
PT113
PT114

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

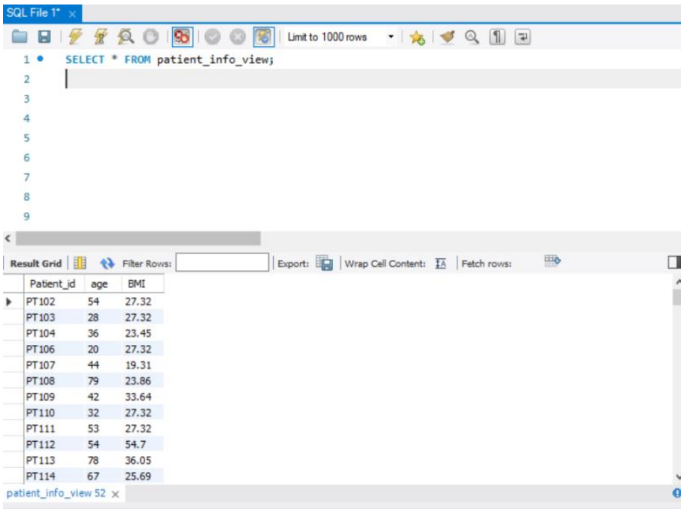
Query:

CREATE VIEW patient_info_view AS

SELECT Patient_id, age, BMI

FROM diabetes_prediction;

Here is the result:



```sql
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
PT112	54	54.7
PT113	78	36.05
PT114	67	25.69

18) Suggest improvements in the database schema to reduce data redundancy and

improve data integrity.

Ans.**1)** Normalization: Ensure that your tables are in at least the Third Normal Form (3NF). This helps eliminate data redundancy by organizing the data into tables and avoiding repeating groups of information. Split large tables into smaller, related tables to reduce redundancy and improve maintainability.

2)Use Primary Keys: Define primary keys for each table. This ensures that each row in the table can be uniquely identified. Consider using surrogate keys (e.g., auto-incremented integers) as primary keys for simplicity and consistency.

3)Foreign Keys: Use foreign keys to establish relationships between tables. This enforces referential integrity, preventing the creation of "orphan" records. Ensure that foreign keys are indexed for better query performance.

4)Data Types: Choose appropriate data types for each column. This not only improves data integrity but also optimizes storage. For example, use DATE or DATETIME for date-related information, and use appropriate numeric types for numerical data.

5)Avoid Storing Derived Data:

Instead of storing data that can be derived from existing data, consider calculating it on-the-fly. For instance, instead of storing the total cost in an order table, calculate it when needed based on the individual item costs.

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

Ans. Indexing: Identify columns used frequently in WHERE clauses and JOIN conditions and create indexes on those columns. This can significantly speed up query performance. Be cautious not to over-index, as this can impact the performance of INSERT, UPDATE, and DELETE operations.

Use EXPLAIN: Use the EXPLAIN statement or equivalent in your database system to analyze the execution plan of a query. This helps identify inefficiencies and suggests areas for optimization.

Avoid SELECT : Instead of using SELECT * to retrieve all columns, explicitly list only the columns needed. This reduces the amount of data transferred and can improve query performance.

Proper Use of Joins: Use the appropriate type of JOIN (INNER JOIN, LEFT JOIN, etc.) based on the relationships between tables. Choose the join type that minimizes the result set size. Ensure that columns used for joining are indexed.

Limit the Result Set: Use the LIMIT clause to restrict the number of rows returned, especially if the query is used for displaying data to users.

Implement pagination to fetch data in smaller chunks.