DIABETES PREDICTION ASSESSMENT

1. Retrieve the Patient_id and ages of all patients.

INPUT: SELECT Patient_id, age FROM diabetes_prediction;;

OUTPUT:

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

SCREENSHOT:

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

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

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

diabetes_prediction 2

Output

Action Output

#	Time	Action	Message
1	03:38:14	SELECT Patient_id, age FROM diabetes_prediction LIMIT 0, 1000	1000 row(s) returned

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

INPUT: SELECT * FROM diabetes_prediction
WHERE gender='Female' AND age>40;

OUTPUT:

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

SCREENSHOT:

```sql
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	diabet
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

diabetes_prediction 3

Output

Action Output

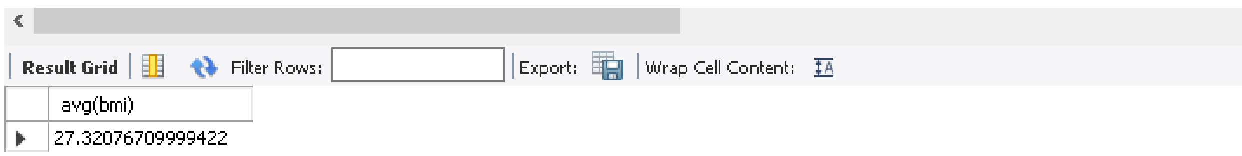
#	Time	Action	Message
1	03:41:38	SELECT * FROM diabetes_prediction WHERE gender='Female' AND age>40 LIMIT 0, 1000	1000 row(s) returned

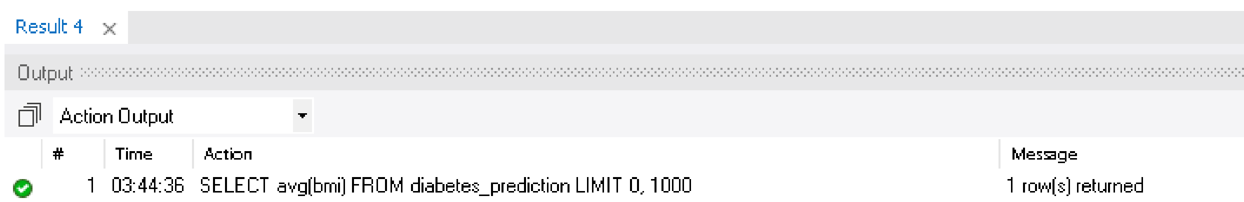
3. Calculate the average BMI of patients.

INPUT: SELECT avg(bmi) FROM diabetes_prediction;

OUTPUT: 27.3207671

SCREENSHOT:





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

INPUT: SELECT * FROM diabetes_prediction
ORDER BY blood_glucose_level DESC;

OUTPUT:

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

SCREENSHOT:

```sql
SELECT * FROM diabetes_prediction
ORDER BY blood_glucose_level DESC;
```

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

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

diabetes_prediction 5

Output

Action Output

#	Time	Action	Message
1	03:46:53	SELECT * FROM diabetes_prediction ORDER BY blood_glucose_level DESC LIMIT 0, 1000	1000 row(s) returned

5. Find patients who have hypertension and diabetes.

INPUT: SELECT * FROM diabetes_prediction
WHERE hypertension=1 AND diabetes=1;

OUTPUT:

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

SCREENSHOT:

```
7 •   SELECT * FROM diabetes_prediction
8     WHERE hypertension=1 AND diabetes=1;
```

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

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

diabetes_prediction 6 ×

Output

Action Output

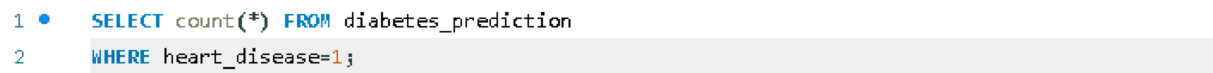
#	Time	Action	Message
1	03:50:39	SELECT * FROM diabetes_prediction WHERE hypertension=1 AND diabetes=1 LIMIT 0, 1000	1000 row(s) returned

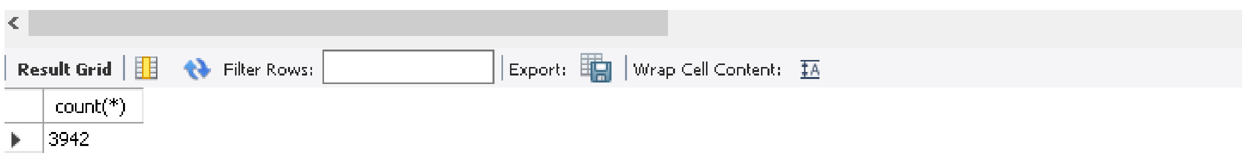
6. Determine the number of patients with heart disease.

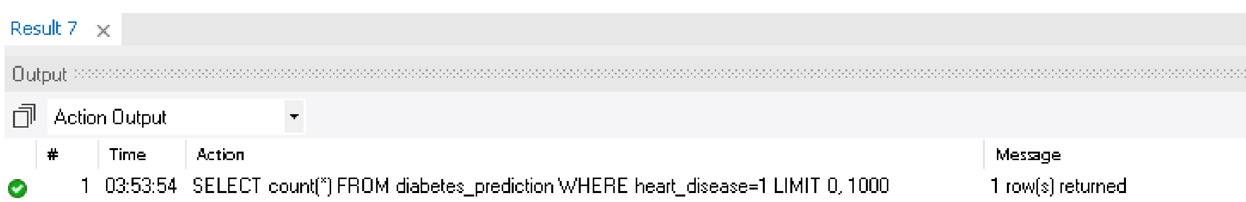
INPUT: SELECT count(*) FROM diabetes_prediction
WHERE heart_disease=1;

OUTPUT: 3942

SCREENSHOT:







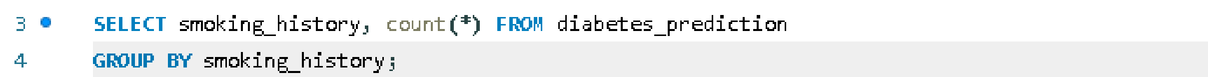
7. Group patients by smoking history and count how many smokers and non-smokers there are.

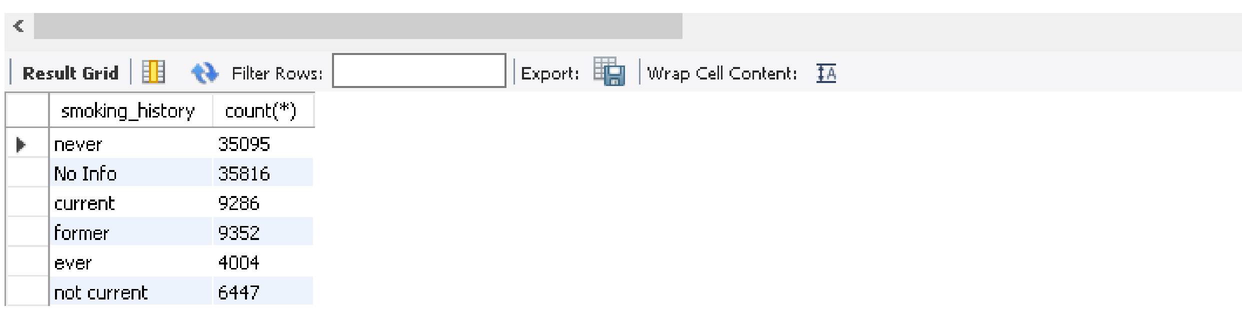
INPUT: SELECT smoking_history, count(*) FROM diabetes_prediction
GROUP BY smoking_history;

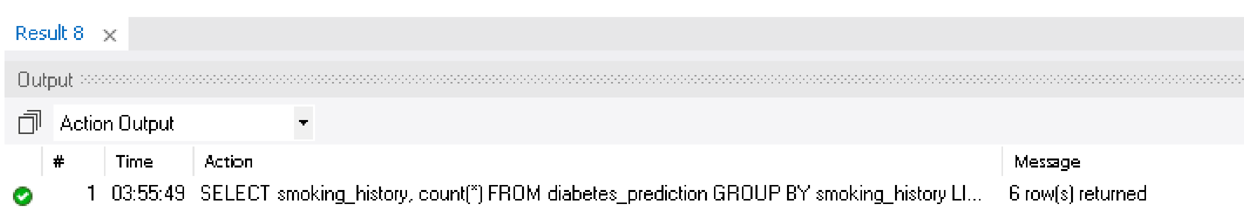
OUTPUT:

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

SCREENSHOT:







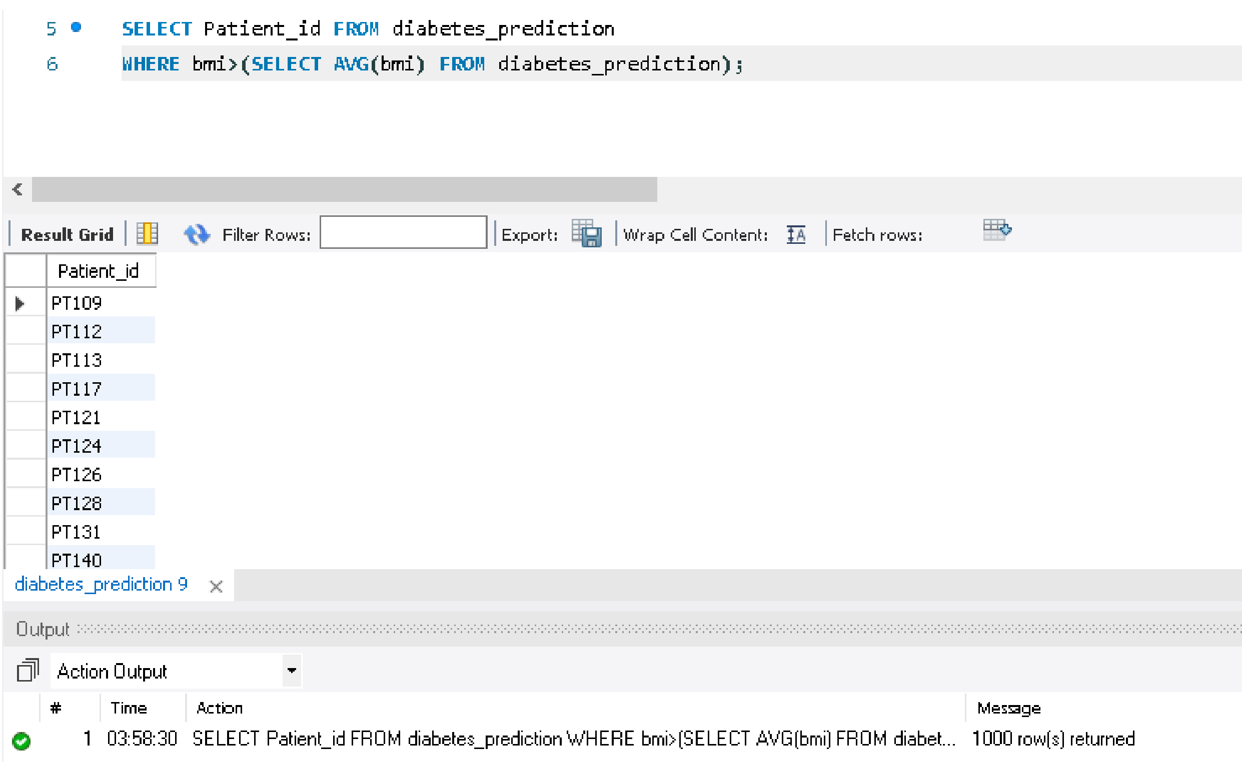
8. Retrieve the Patient_ids of patients who have a BMI greater than the average BMI.

INPUT: SELECT Patient_id FROM diabetes_prediction
WHERE bmi>(SELECT AVG(bmi) FROM diabetes_prediction);

OUTPUT:

PT109
PT112
PT113
PT117
PT121
PT124
PT126
PT128
PT131
PT140

SCREENSHOT:



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

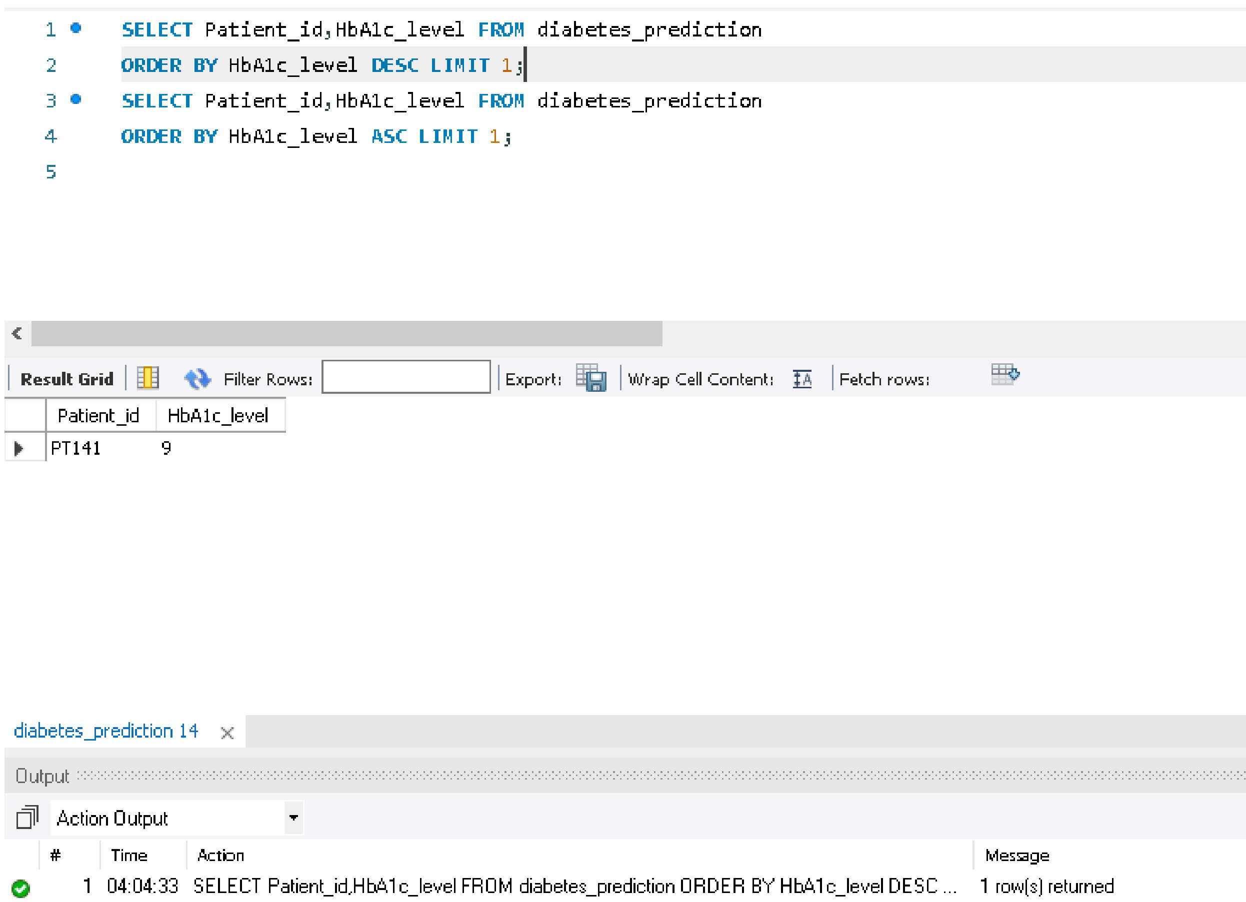
INPUT:

HIGHEST: SELECT Patient_id,HbA1c_level FROM diabetes_prediction
ORDER BY HbA1c_level DESC LIMIT 1;

OUTPUT:

PT141	9

SCREENSHOT:



INPUT:

LOWEST: SELECT Patient_id,HbA1c_level FROM diabetes_prediction
ORDER BY HbA1c_level ASC LIMIT 1;

OUTPUT:

PT120	3.5

SCREENSHOT:

```
SELECT Patient_id,HbA1c_level FROM diabetes_prediction
ORDER BY HbA1c_level DESC LIMIT 1;
SELECT Patient_id,HbA1c_level FROM diabetes_prediction
ORDER BY HbA1c_level ASC LIMIT 1;
```

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

Patient_id	HbA1c_level
PT120	3.5

diabetes_prediction 15

Output

Action Output

#	Time	Action	Message
1	04:04:33	SELECT Patient_id,HbA1c_level FROM diabetes_prediction ORDER BY HbA1c_level DESC ...	1 row(s) returned
2	04:05:35	SELECT Patient_id,HbA1c_level FROM diabetes_prediction ORDER BY HbA1c_level ASC LI...	1 row(s) returned

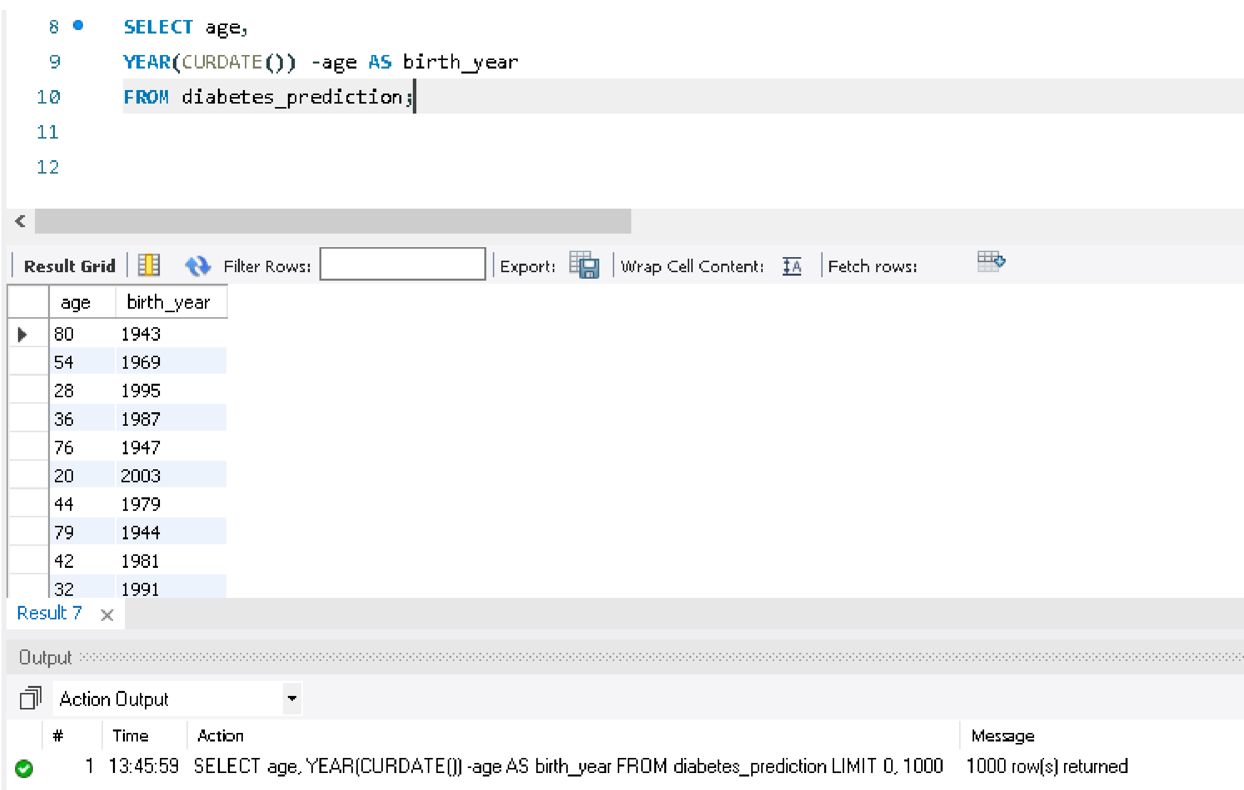
10. Calculate the age of patients in years (assuming the current date as of now).

INPUT: SELECT age,
YEAR(CURDATE()) -age AS birth_year
FROM diabetes_prediction;

OUTPUT:

80	1943
54	1969
28	1995
36	1987
76	1947
20	2003
44	1979
79	1944
42	1981
32	1991

SCREENSHOT:



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

INPUT: SELECT Patient_id, blood_glucose_level, gender,
RANK() OVER (PARTITION BY gender
ORDER BY blood_glucose_level) AS GlucoseLevelRank
FROM diabetes_prediction;

OUTPUT:

PT96324	80	Female	1
PT97215	80	Female	1
PT96610	80	Female	1
PT96379	80	Female	1
PT99580	80	Female	1
PT97219	80	Female	1
PT98849	80	Female	1
PT96778	80	Female	1
PT98364	80	Female	1
PT98209	80	Female	1

SCREENSHOT:

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

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

Patient_id	blood_glucose_level	gender	GlucoseLevelRank
PT82519	300	Male	41102
PT80098	300	Male	41102
PT75403	80	Other	1
PT16803	85	Other	2
PT40438	90	Other	3
PT35030	90	Other	3
PT56339	126	Other	5
PT71240	126	Other	5

Result 1

Output

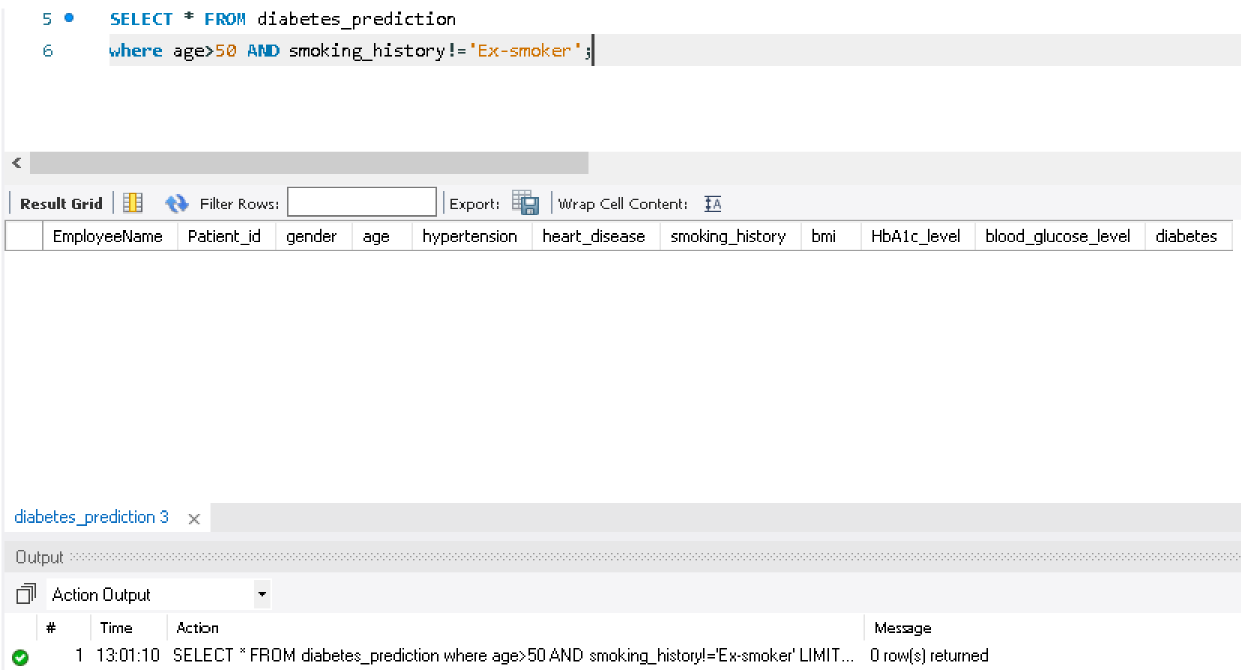
Action Output

#	Time	Action	Message
1	12:45:23	SELECT Patient_id, blood_glucose_level, gender, RANK() OVER (PARTITION BY gender OR...	100000 row(s) returned

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

INPUT: UPDATE diabetes_prediction
SET smoking_history='Ex-smoker'
WHERE AGE>50;

OUTPUT:



SCREENSHOT:

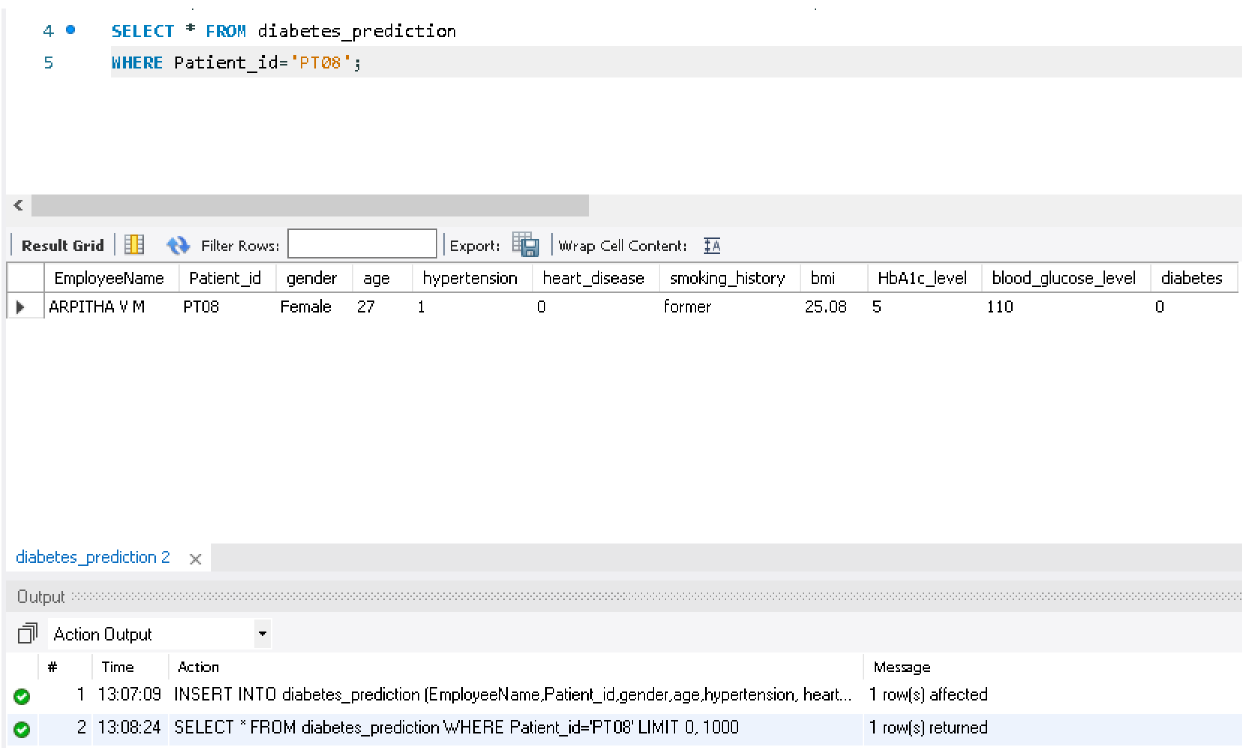

SET SQL_SAFE_UPDATES = 0;
UPDATE diabetes_prediction
SET smoking_history='Ex-smoker'
WHERE AGE>50;
SET SQL_SAFE_UPDATES = 1;
Output
Action Output
Time Action Message
3 12:56:28 SET SQL_SAFE_UPDATES = 1 0 row(s) affected
4 12:57:47 select smoking_history from diabetes_prediction where age>50 LIMIT 0, 1000 1000 row(s) returned

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

INPUT:

INSERT INTO diabetes_prediction (EmployeeName,Patient_id,gender,age,hypertension, heart_disease,smoking_history,bmi,HbA1c_level,blood_glucose_level,diabetes)
values ('ARPITHA V M','PT08','Female',27,1,0,'former',25.08,5,110,0);

OUTPUT:



```
SELECT * FROM diabetes_prediction
WHERE Patient_id='PT08';
```

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

EmployeeName	Patient_id	gender	age	hypertension	heart_disease	smoking_history	bmi	HbA1c_level	blood_glucose_level	diabetes
ARPITHA V M	PT08	Female	27	1	0	former	25.08	5	110	0

diabetes_prediction 2

Output

Action Output

#	Time	Action	Message
1	13:07:09	INSERT INTO diabetes_prediction (EmployeeName,Patient_id,gender,age,hypertension, heart...	1 row(s) affected
2	13:08:24	SELECT * FROM diabetes_prediction WHERE Patient_id='PT08' LIMIT 0, 1000	1 row(s) returned

SCREENSHOT:

```
INSERT INTO diabetes_prediction (EmployeeName,Patient_id,gender,age,hypertension,
heart_disease,smoking_history,bmi,HbA1c_level,blood_glucose_level,diabetes)
values ('ARPITHA V M','PT08','Female',27,1,0,'former',25.08,5,110,0);
```

Output

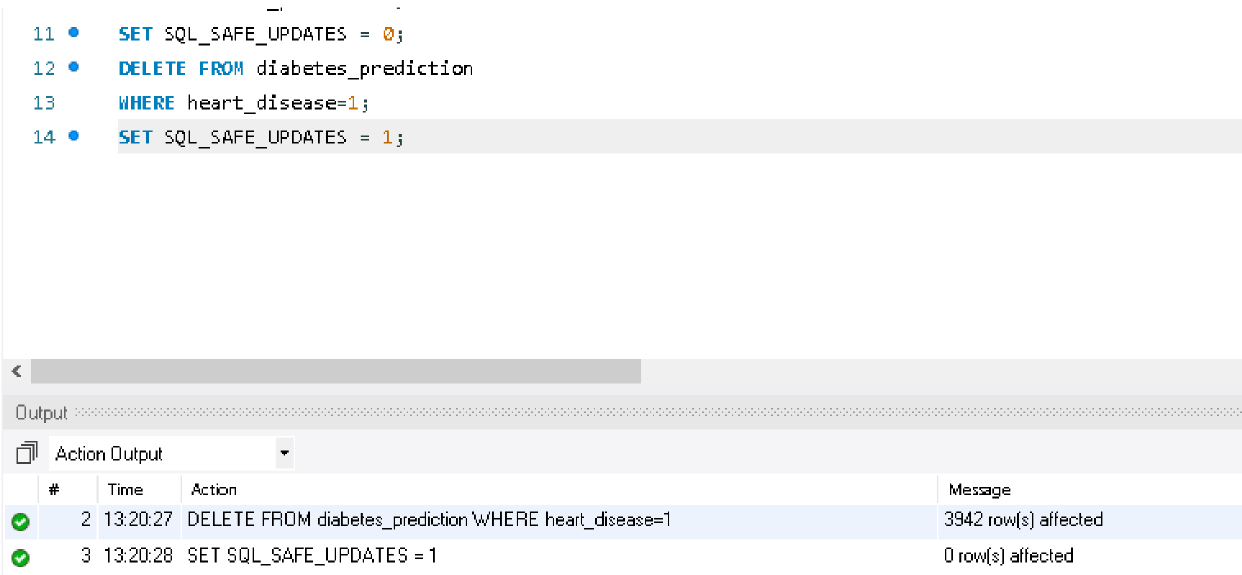
Action Output

	#	Time	Action	Message
	1	13:07:09	INSERT INTO diabetes_prediction (EmployeeName,Patient_id,gender,age,hypertension, heart_...	1 row(s) affected

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

INPUT: DELETE FROM diabetes_prediction
WHERE heart_disease=1;

SCREENSHOT:



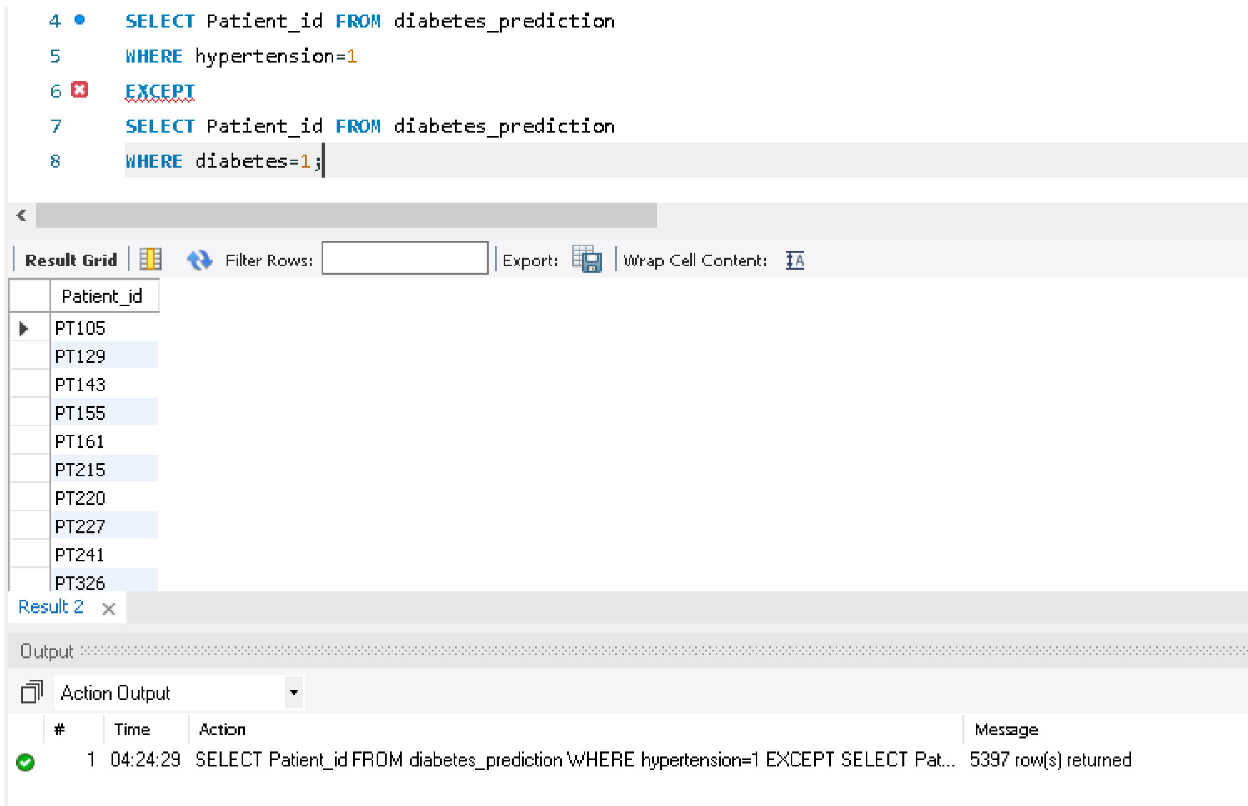
15. Find patients who have hypertension but not diabetes using the EXCEPT operator.

INPUT:
```
SELECT Patient_id FROM diabetes_prediction
WHERE hypertension=1
EXCEPT
SELECT Patient_id FROM diabetes_prediction
WHERE diabetes=1;
```

OUTPUT:

PT105
PT129
PT143
PT155
PT161
PT215
PT220
PT227
PT241
PT326

SCREENSHOT:

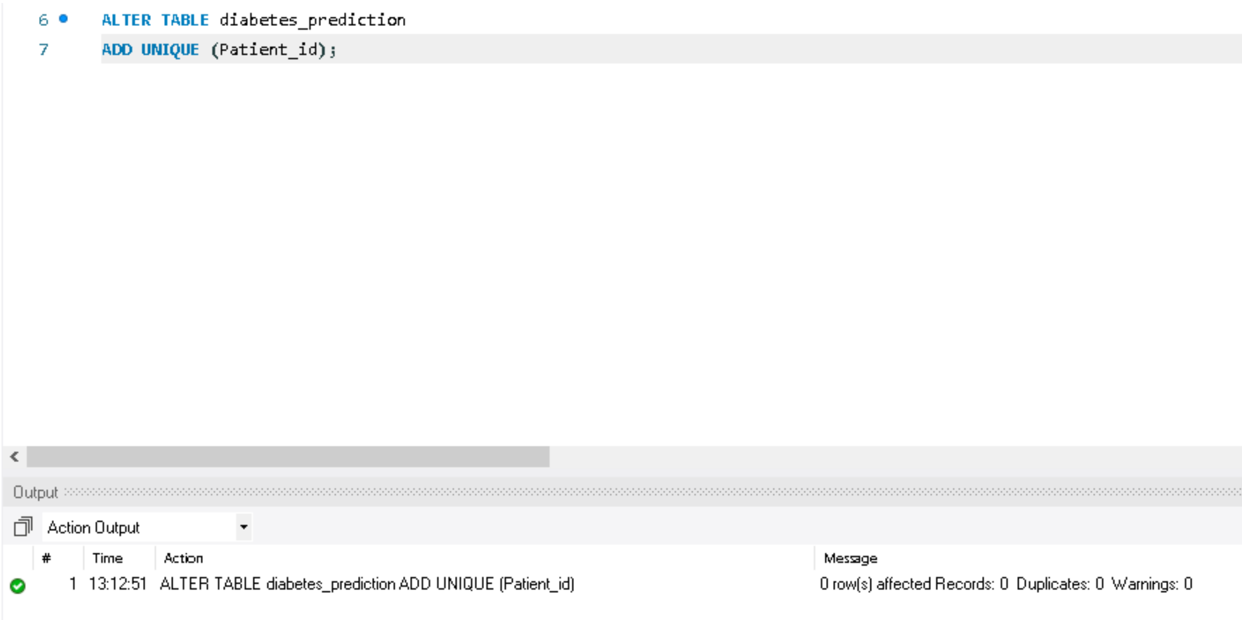


4 SELECT Patient_id FROM diabetes_prediction
5 WHERE hypertension=1
6 EXCEPT
7 SELECT Patient_id FROM diabetes_prediction
8 WHERE diabetes=1;
Result Grid
Filter Rows:
Export:
Wrap Cell Content:
Patient_id
PT105
PT129
PT143
PT155
PT161
PT215
PT220
PT227
PT241
PT326
Result 2
Output
Action Output
#
Time
Action
Message
1
04:24:29
SELECT Patient_id FROM diabetes_prediction WHERE hypertension=1 EXCEPT SELECT Pat...
5397 row(s) returned

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

INPUT: ALTER TABLE diabetes_prediction
ADD UNIQUE (Patient_id);

SCREENSHOT:



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

INPUT: CREATE VIEW PatientInfo AS
SELECT Patient_id,age,bmi
FROM diabetes_prediction;

OUTPUT:

```
1 ● SELECT * FROM internship.patientinfo;
```

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

Patient_id	age	bmi
PT101	80	25.19
PT102	54	27.32
PT103	28	27.32
PT104	36	23.45
PT105	76	20.14
PT106	20	27.32
PT107	44	19.31
PT108	79	23.86
PT109	42	33.64
PT110	32	27.32

patientinfo 1

Output

Action Output

#	Time	Action	Message
1	13:15:48	CREATE VIEW PatientInfo AS SELECT Patient_id,age,bmi FROM diabetes_prediction	0 row(s) affected
2	13:16:07	SELECT * FROM internship.patientinfo LIMIT 0, 1000	1000 row(s) returned

SCREENSHOT:

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

Output

Action Output

	#	Time	Action	Message
	1	13:15:48	CREATE VIEW PatientInfo AS SELECT Patient_id,age,bmi FROM diabetes_prediction	0 row(s) affected
	2	13:16:07	SELECT * FROM internship.patientinfo LIMIT 0, 1000	1000 row(s) returned

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

ANS:
*Normalization: Ensuring that the database schema is normalized to at least the third normal form (3NF). This will help in eliminating unnecessary data and minimizes the update faults/anomalies. Avoiding storing of duplicate information by breaking down the tables into smaller or removing the related tables.
*Use of Primary and Foreign Keys: To uniquely identify each record primary keys are to be defined for each table
and foreign keys to be used to establish relationships between tables. This ensures referential integrity and avoids the need to duplicate data across tables.
*Unique Constraints: Preventing duplicate entries in columns where uniqueness is required by applying unique constraints.
*Data Types: Appropriate data types are to be used for each column to optimize storage and ensure data integrity. For example, use integers for whole numbers, and varchar for variable-length character data.
*Default Values and Constraints: Default values are to be set for columns wherever possible to ensure that all records have a valid value. Use of constraints such as NOT NULL to enforce data integrity rules at the database level also to be noted.
*Indexes: To improve query performance indexes on columns used for searching and joining should be created being cautious not to over-index which might impact the performance.

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

ANS:
*Use Indexing: The columns that are frequently used in WHERE clauses or JOIN conditions are to be properly indexed. Indexing can significantly speed up data retrieval.
*Optimize JOIN Operations: Be mindful of the JOIN operations in the queries, ensure that the join on indexed columns and use appropriate types of JOINs such as INNER JOIN, LEFT JOIN, etc. based on the requirements.
*Limit the Number of Rows Returned: Use the LIMIT or TOP clause to restrict the number of rows returned by the query, especially when dealing with large datasets. This can improve query response time.
*Avoid SELECT : Instead of using SELECT *, explicitly list the columns you need. This reduces the amount of data transferred and can improve query performance.
*Aggregate Functions and GROUP BY: Use aggregate functions such as SUM, AVG, COUNT judiciously and combine them with the GROUP BY clause when necessary. This can help in summarizing data efficiently.
*Subqueries: Subqueries can impact performance and so in some cases rewriting a subquery as a JOIN or using EXISTS can be more efficient.
*Avoid Using DISTINCT Unnecessarily: Using DISTINCT can be resource-intensive. Only use it when necessary, and consider if there are alternative ways to achieve the desired result.
*Optimize WHERE Clause:Write efficient WHERE clauses by avoiding unnecessary conditions. Ensure that the columns involved in conditions are indexed.