SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

GREAT LEARNING

20PITE51J – SQL FOR DATA SCIENCE

MINI PROJECT – QUESTIONS2

Created Database for Assignment.

```
 CREATE DATABASE CT3;
  USE CT3;
```

1) **From the following tables write a SQL query to find those orders where the order amount exists between 500 and 2000. Return ord_no, purch_amt, cust_name, city. (8 marks)**

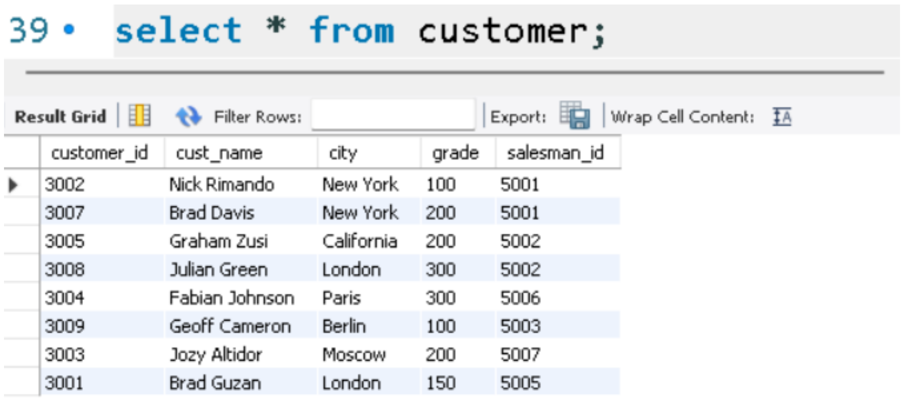
Step 1: Creating the tables:

```
CREATE TABLE customer (
   customer_id INT,
   cust_name VARCHAR(50),
   city VARCHAR(50),
   grade INT,
   salesman_id INT
);

CREATE TABLE orders (
   ord_no INT,
   purch_amt DECIMAL(10,2),
   ord_date DATE,
   customer_id INT,
   salesman_id INT
);
```

Step 2: Inserting the records and displaying it:

INSERT INTO customer (customer_id, cust_name, city, grade, salesman_id)
VALUES
(3002, 'Nick Rimando', 'New York', 100, 5001),
(3007, 'Brad Davis', 'New York', 200, 5001),
(3005, 'Graham Zusi', 'California', 200, 5002),
(3008, 'Julian Green', 'London', 300, 5002),
(3004, 'Fabian Johnson', 'Paris', 300, 5006),
(3009, 'Geoff Cameron', 'Berlin', 100, 5003),
(3003, 'Jozy Altidor', 'Moscow', 200, 5007),
(3001, 'Brad Guzan', 'London', 150, 5005);

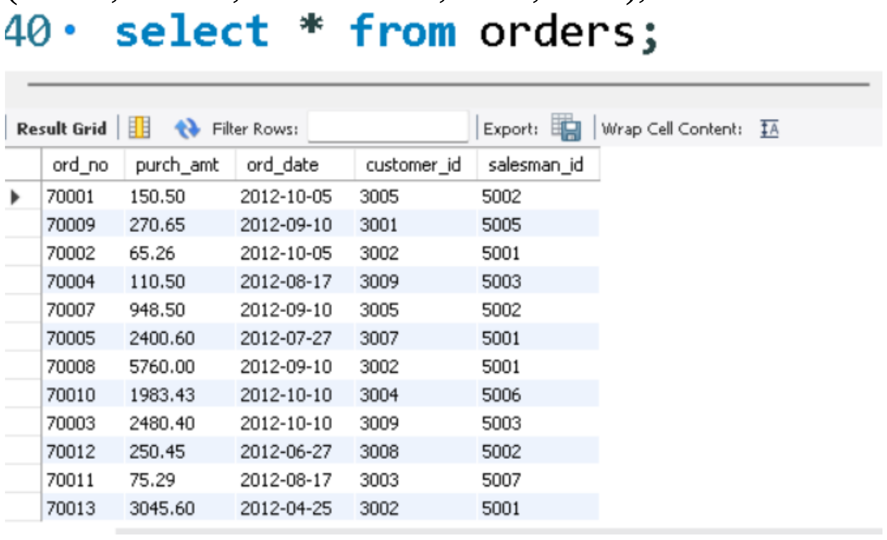


```
select * from customer;
```

customer_id	cust_name	city	grade	salesman_id
3002	Nick Rimando	New York	100	5001
3007	Brad Davis	New York	200	5001
3005	Graham Zusi	California	200	5002
3008	Julian Green	London	300	5002
3004	Fabian Johnson	Paris	300	5006
3009	Geoff Cameron	Berlin	100	5003
3003	Jozy Altidor	Moscow	200	5007
3001	Brad Guzan	London	150	5005

INSERT INTO orders (ord_no, purch_amt, ord_date, customer_id,
salesman_id) VALUES
(70001, 150.5, '2012-10-05', 3005, 5002),
(70009, 270.65, '2012-09-10', 3001, 5005),
(70002, 65.26, '2012-10-05', 3002, 5001),
(70004, 110.5, '2012-08-17', 3009, 5003),
(70007, 948.5, '2012-09-10', 3005, 5002),
(70005, 2400.6, '2012-07-27', 3007, 5001),
(70008, 5760, '2012-09-10', 3002, 5001),
(70010, 1983.43, '2012-10-10', 3004, 5006),
(70003, 2480.4, '2012-10-10', 3009, 5003),
(70012, 250.45, '2012-06-27', 3008, 5002),

```
(70011, 75.29, '2012-08-17', 3003, 5007),
(70013, 3045.6, '2012-04-25', 3002, 5001);
```



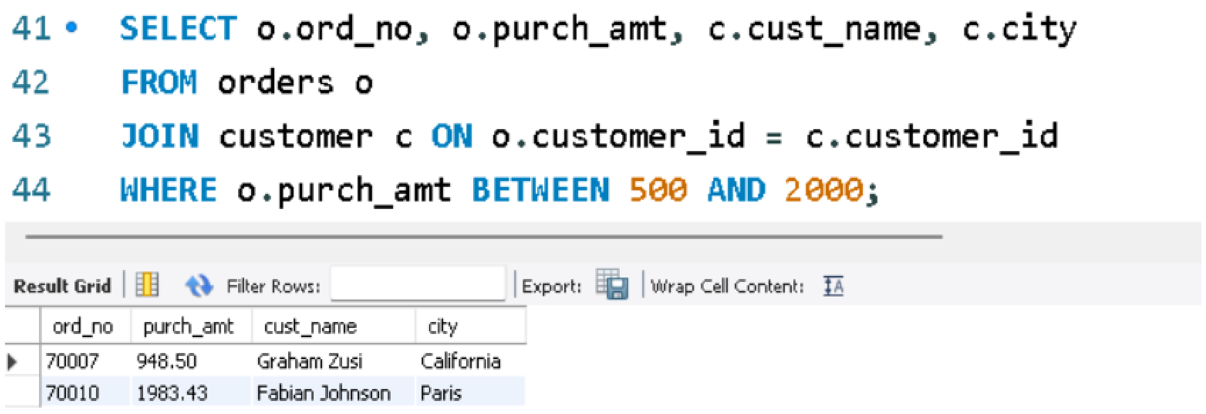
```
select * from orders;
```

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

ord_no	purch_amt	ord_date	customer_id	salesman_id
70001	150.50	2012-10-05	3005	5002
70009	270.65	2012-09-10	3001	5005
70002	65.26	2012-10-05	3002	5001
70004	110.50	2012-08-17	3009	5003
70007	948.50	2012-09-10	3005	5002
70005	2400.60	2012-07-27	3007	5001
70008	5760.00	2012-09-10	3002	5001
70010	1983.43	2012-10-10	3004	5006
70003	2480.40	2012-10-10	3009	5003
70012	250.45	2012-06-27	3008	5002
70011	75.29	2012-08-17	3003	5007
70013	3045.60	2012-04-25	3002	5001

Step 3: write a SQL query to find those orders where the order amount exists between 500 and 2000. Return ord_no, purch_amt, cust_name, city.

```
SELECT o.ord_no, o.purch_amt, c.cust_name, c.city
FROM orders o
JOIN customer c ON o.customer_id = c.customer_id
WHERE o.purch_amt BETWEEN 500 AND 2000;
```



```
SELECT o.ord_no, o.purch_amt, c.cust_name, c.city
FROM orders o
JOIN customer c ON o.customer_id = c.customer_id
WHERE o.purch_amt BETWEEN 500 AND 2000;
```

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

ord_no	purch_amt	cust_name	city
70007	948.50	Graham Zusi	California
70010	1983.43	Fabian Johnson	Paris

Inference:

The above query performs joining operation between customer and orders table. The result set contains details such as ord_no, purch_amt, cust_name, city where purch_amt is between 500 and 2000.

2) **From the following tables write a SQL query to find the salesperson and customer who reside in the same city. Return Salesman, cust_name and city. (8 marks)**

Step 1: Creating the tables:

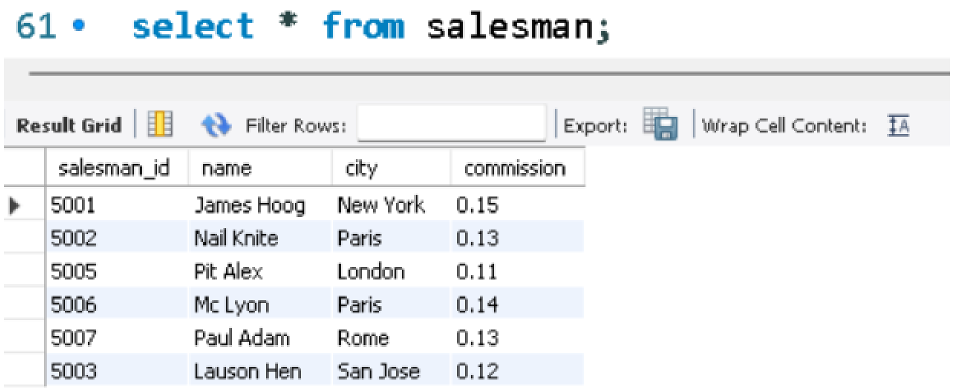
```
CREATE TABLE salesman (
   salesman_id INT,
   name VARCHAR(50),
   city VARCHAR(50),
   commission DECIMAL(4,2)
);

CREATE TABLE customer (
   customer_id INT,
   cust_name VARCHAR(50),
   city VARCHAR(50),
   grade INT,
   salesman_id INT
);
```

Step 2: Inserting the records and displaying it:

```
INSERT INTO salesman (salesman_id, name, city, commission) VALUES
(5001, 'James Hoog', 'New York', 0.15),
(5002, 'Nail Knite', 'Paris', 0.13),
(5005, 'Pit Alex', 'London', 0.11),
```

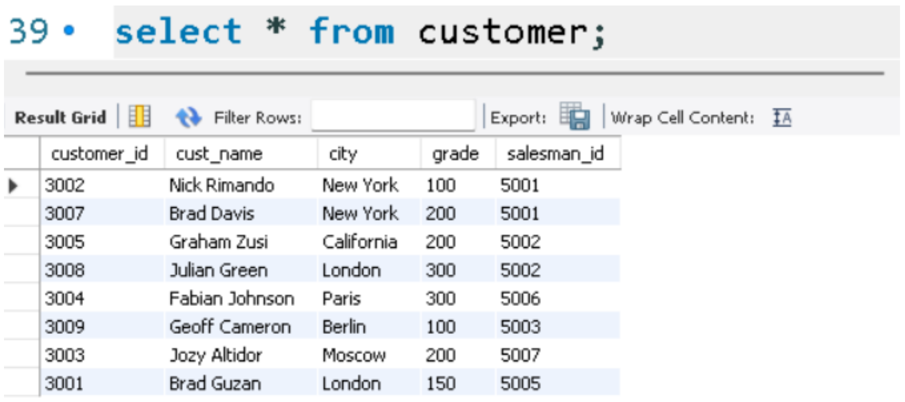
(5006, 'Mc Lyon', 'Paris', 0.14),
(5007, 'Paul Adam', 'Rome', 0.13),
(5003, 'Lauson Hen', 'San Jose', 0.12);



```
select * from salesman;
```

salesman_id	name	city	commission
5001	James Hoog	New York	0.15
5002	Nail Knite	Paris	0.13
5005	Pit Alex	London	0.11
5006	Mc Lyon	Paris	0.14
5007	Paul Adam	Rome	0.13
5003	Lauson Hen	San Jose	0.12

INSERT INTO customer (customer_id, cust_name, city, grade, salesman_id)
VALUES
(3002, 'Nick Rimando', 'New York', 100, 5001),
(3007, 'Brad Davis', 'New York', 200, 5001),
(3005, 'Graham Zusi', 'California', 200, 5002),
(3008, 'Julian Green', 'London', 300, 5002),
(3004, 'Fabian Johnson', 'Paris', 300, 5006),
(3009, 'Geoff Cameron', 'Berlin', 100, 5003),
(3003, 'Jozy Altidor', 'Moscow', 200, 5007),
(3001, 'Brad Guzan', 'London', 150, 5005);

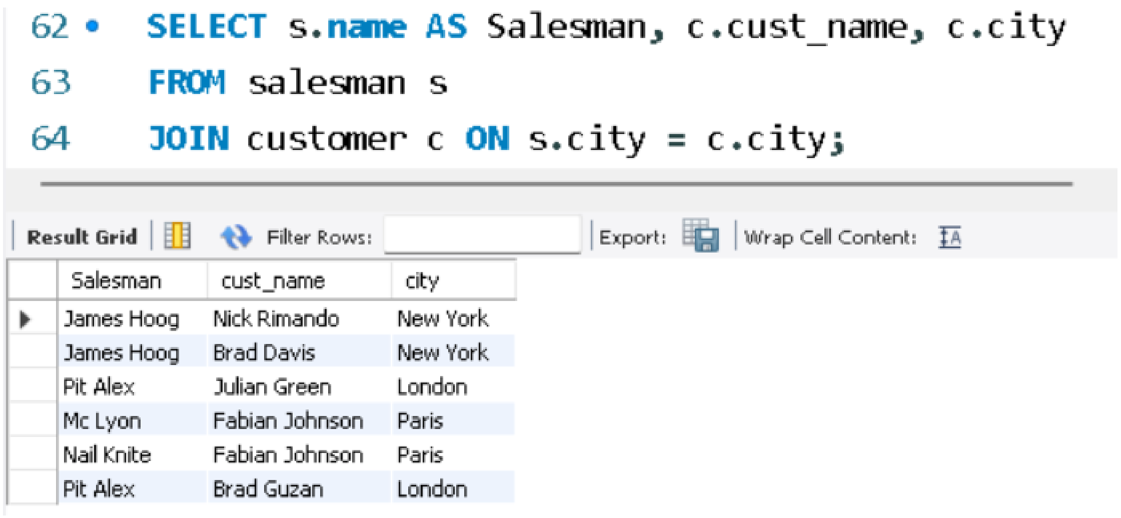


```
select * from customer;
```

customer_id	cust_name	city	grade	salesman_id
3002	Nick Rimando	New York	100	5001
3007	Brad Davis	New York	200	5001
3005	Graham Zusi	California	200	5002
3008	Julian Green	London	300	5002
3004	Fabian Johnson	Paris	300	5006
3009	Geoff Cameron	Berlin	100	5003
3003	Jozy Altidor	Moscow	200	5007
3001	Brad Guzan	London	150	5005

Step 3: write a SQL query to find the salesperson and customer who reside in the same city. Return Salesman, cust_name and city.

select * from salesman;
SELECT s.name AS Salesman, c.cust_name, c.city
FROM salesman s
JOIN customer c ON s.city = c.city;



```
SELECT s.name AS Salesman, c.cust_name, c.city
FROM salesman s
JOIN customer c ON s.city = c.city;
```

Salesman	cust_name	city
James Hoog	Nick Rimando	New York
James Hoog	Brad Davis	New York
Pit Alex	Julian Green	London
Mc Lyon	Fabian Johnson	Paris
Nail Knite	Fabian Johnson	Paris
Pit Alex	Brad Guzan	London

Inference:

The above query performs the joining operation between Salesman and customer table. The result set contains Salesman and customer who reside in same city.

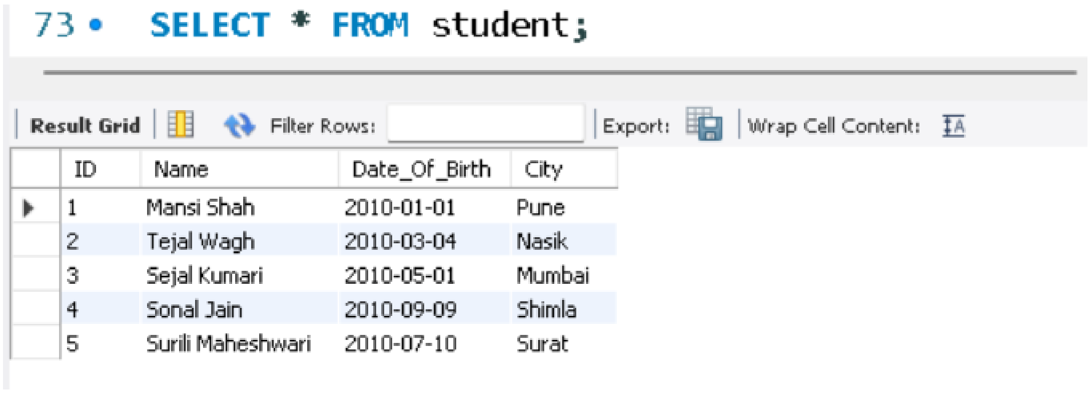
3) Implementation of Date functions and Conditional statements. (7 marks)

Step 1: Creating the table:

```
CREATE TABLE student(
ID int,
Name varchar(20),
Date_Of_Birth date,
City varchar(20) );
```

Step 2: Inserting the records and displaying it:

```
INSERT INTO student VALUES(
1, 'Mansi Shah', '2010-01-01', 'Pune'),
(2,'Tejal Wagh', '2010-03-04', 'Nasik'),
(3,'Sejal Kumari', '2010-05-01','Mumbai'),
(4, 'Sonal Jain', '2010-09-09', 'Shimla'),
(5, 'Surili Maheshwari', '2010-07-10', 'Surat');
```

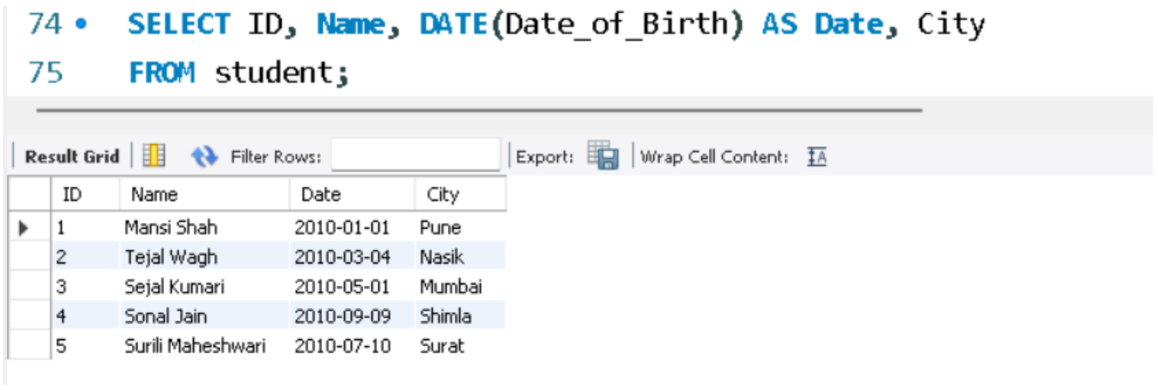


```
SELECT * FROM student;
```

ID	Name	Date_Of_Birth	City
1	Mansi Shah	2010-01-01	Pune
2	Tejal Wagh	2010-03-04	Nasik
3	Sejal Kumari	2010-05-01	Mumbai
4	Sonal Jain	2010-09-09	Shimla
5	Surili Maheshwari	2010-07-10	Surat

Step 3: Write a query to display all the details from the student table with the date from the DateTime_Birth column of the student table.

SELECT ID, Name, DATE(Date_of_Birth) AS Date, City
FROM student;



```
SELECT ID, Name, DATE(Date_of_Birth) AS Date, City
FROM student;
```

ID	Name	Date	City
1	Mansi Shah	2010-01-01	Pune
2	Tejal Wagh	2010-03-04	Nasik
3	Sejal Kumari	2010-05-01	Mumbai
4	Sonal Jain	2010-09-09	Shimla
5	Surili Maheshwari	2010-07-10	Surat

Inference:

The above query operates on the student table, dealing with the extraction of date information from the Date_of_Birth column

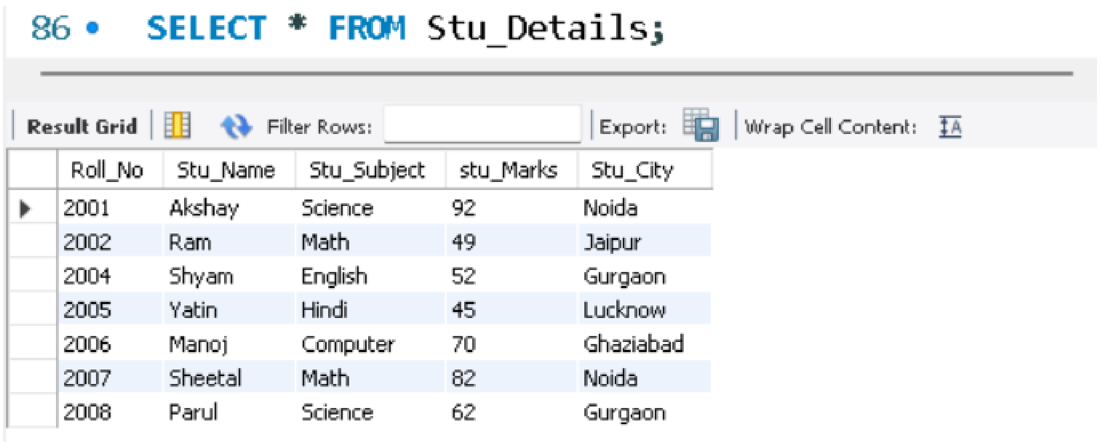
4) **Consider the following table: (7 marks)**

Step 1: Creating the table:

CREATE TABLE Stu_Details(
Roll_No int,
Stu_Name varchar(20),
Stu_Subject varchar(20),
stu_Marks int,
Stu_City varchar(20));

Step 2: Inserting the records and displaying it:

```
INSERT INTO Stu_Details VALUES(
2001, 'Akshay', 'Science', 92, 'Noida'),
(2002, 'Ram', 'Math', 49, 'Jaipur'),
(2004, 'Shyam', 'English', 52, 'Gurgaon'),
(2005, 'Yatin', 'Hindi', 45, 'Lucknow'),
(2006, 'Manoj', 'Computer', 70, 'Ghaziabad'),
(2007, 'Sheetal', 'Math', 82, 'Noida'),
(2008, 'Parul', 'Science', 62, 'Gurgaon');
```



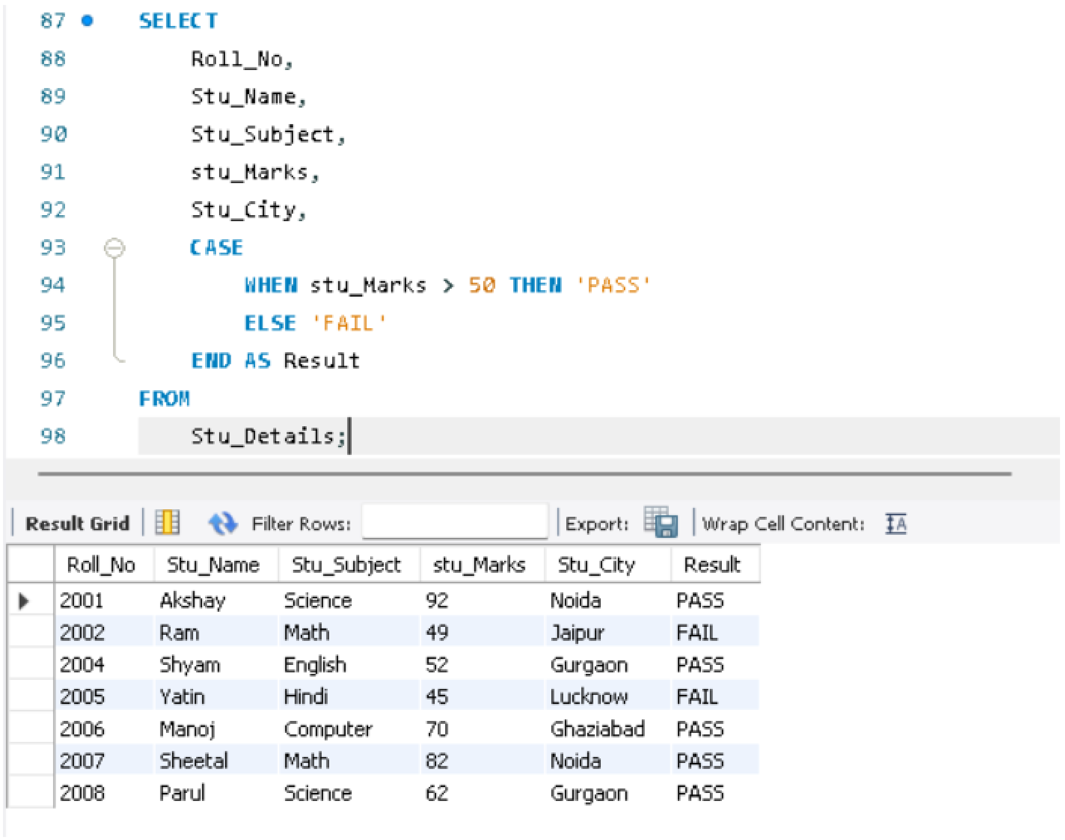
```
SELECT * FROM Stu_Details;
```

Roll_No	Stu_Name	Stu_Subject	stu_Marks	Stu_City
2001	Akshay	Science	92	Noida
2002	Ram	Math	49	Jaipur
2004	Shyam	English	52	Gurgaon
2005	Yatin	Hindi	45	Lucknow
2006	Manoj	Computer	70	Ghaziabad
2007	Sheetal	Math	82	Noida
2008	Parul	Science	62	Gurgaon

Step 3: From the above table write a query to display the result as PASS or FAIL where marks>50 using conditional expression

```
SELECT
   Roll_No,
   Stu_Name,
   Stu_Subject,
   stu_Marks,
   Stu_City,
   CASE
```

```
        WHEN stu_Marks > 50 THEN 'PASS'
        ELSE 'FAIL'
    END AS Result
FROM
    Stu_Details;
```



```
SELECT
    Roll_No,
    Stu_Name,
    Stu_Subject,
    stu_Marks,
    Stu_City,
    CASE
        WHEN stu_Marks > 50 THEN 'PASS'
        ELSE 'FAIL'
    END AS Result
FROM
    Stu_Details;
```

Roll_No	Stu_Name	Stu_Subject	stu_Marks	Stu_City	Result
2001	Akshay	Science	92	Noida	PASS
2002	Ram	Math	49	Jaipur	FAIL
2004	Shyam	English	52	Gurgaon	PASS
2005	Yatin	Hindi	45	Lucknow	FAIL
2006	Manoj	Computer	70	Ghaziabad	PASS
2007	Sheetal	Math	82	Noida	PASS
2008	Parul	Science	62	Gurgaon	PASS

Inference:

The above query operates on the Stu_Details table, determining whether a student has passed or failed based on their marks. Conditional expressions in SQL, such as CASE statements, are powerful for manipulating and transforming data based on specified conditions.