# DBMS-LAB ASSIGNMENT 7
# NAME: KASHA SINGH
# REG. NO.: 19BCS051

1. Write two stored Procedures relevant to your database.

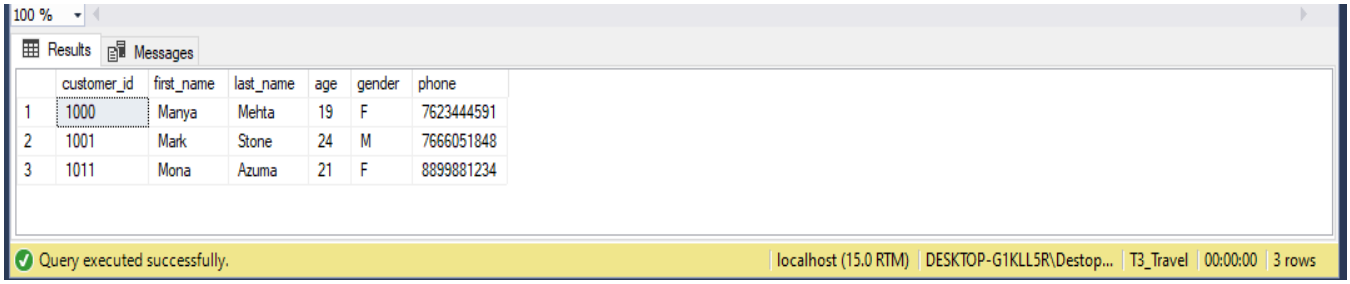
QUERY:

```
USE T3_Travel;

CREATE PROCEDURE Names_starting_with_M
AS
SELECT * FROM T3_CustomerDetails WHERE first_name LIKE 'M%'
GO

EXEC Names_starting_with_M;
```

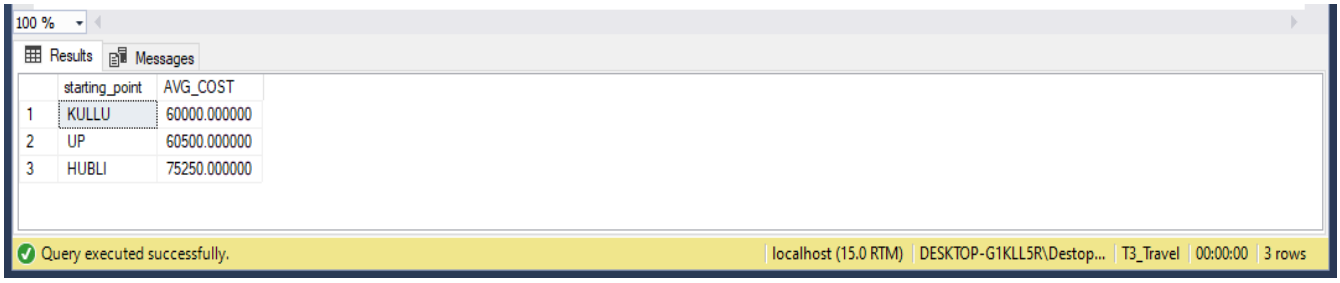
DATABASE OUTPUT:




```
CREATE PROCEDURE PackageAvgCost
AS
SELECT starting_point, AVG(cost) AS AVG_COST FROM T3_PackageDetails
GROUP BY starting_point ORDER BY AVG_COST
GO

EXEC PackageAvgCost;
```

DATABASE OUTPUT:

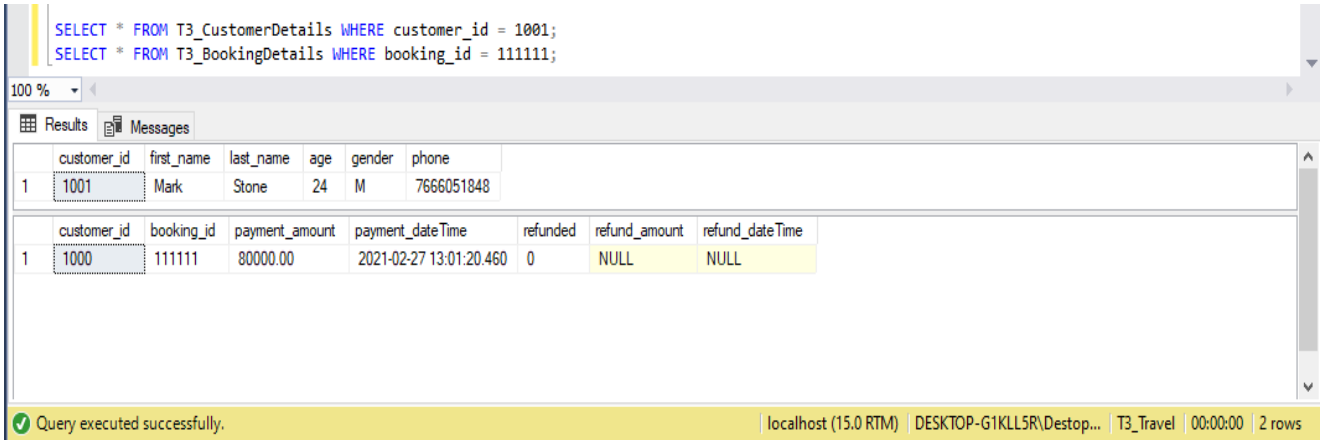


2. Write a transaction to illustrate atomicity (related to your database)

QUERY:

```
--Atomicity--
BEGIN TRAN Transaction_Update
UPDATE T3_CustomerDetails SET age = 24 WHERE customer_id = 1001
UPDATE T3_BookingDetails SET payment_amount = 80000 where booking_id = 111111
COMMIT
```

DATABASE OUTPUT:




As the transaction is atomic, both of the updates on the two separate tables will commit together, or they will rollback together.

3. Write a transaction to illustrate isolation level. It can be on commit or uncommit read (related to your database)

Window 1:

```
--Isolation Property--
USE T3_Travel;
GO
BEGIN TRAN Trans_Isolation

UPDATE T3_CustomerDetails
SET last_name = 'Bond'
WHERE customer_id = 1105
```

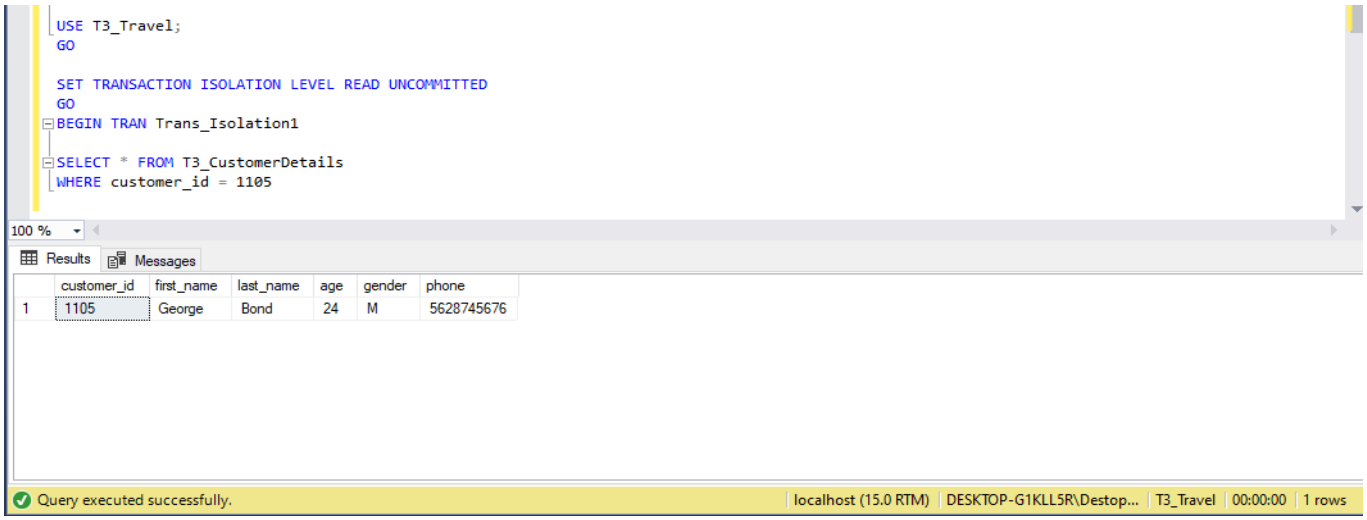
Window 2:

```
USE T3_Travel;
GO
```

```
SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED
GO
BEGIN TRAN Trans_Isolation1

SELECT * FROM T3_CustomerDetails
WHERE customer_id = 1105
```

DATABASE OUTPUT:




When we set the isolation level to read uncommitted, we will be able to see the last_name set to 'Bond', called Dirty Read.