

DBMS Lab Assignment-7

Name:- M.Sai Chaitra

Roll No. :-19BCS065.

Database:- Travel Agency

1. Write two stored Procedures relevant to your database.

Query :

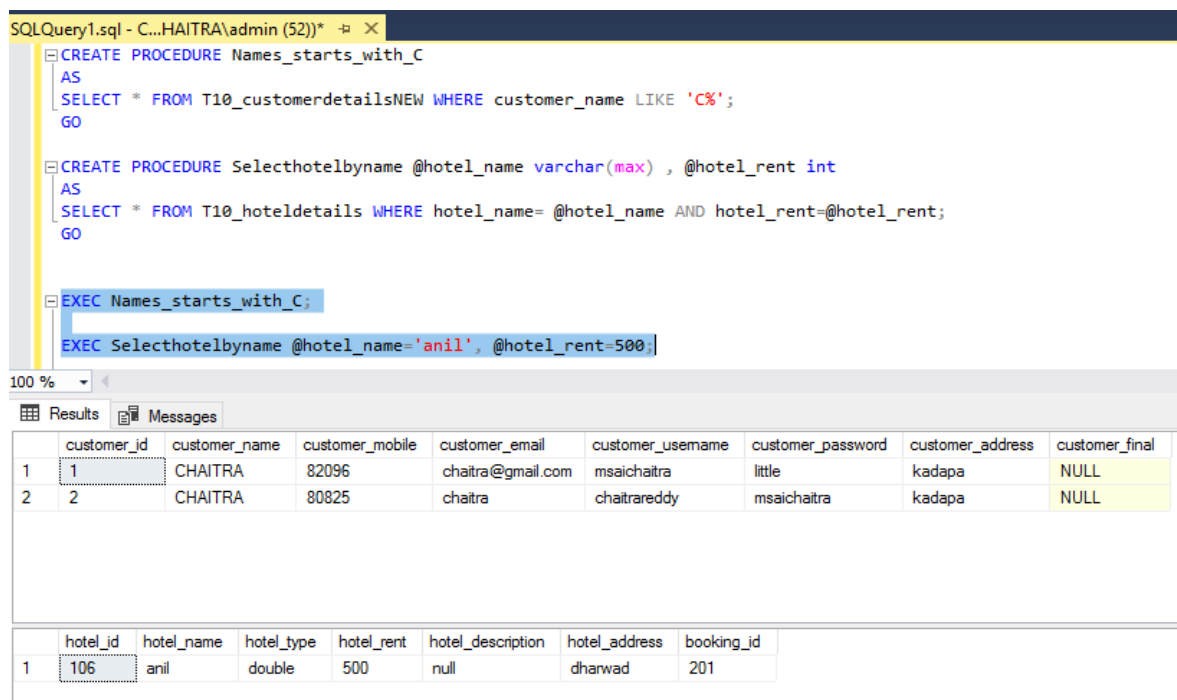
```
CREATE PROCEDURE Names_starts_with_C
AS
SELECT * FROM T10_customerdetailsNEW WHERE customer_name LIKE 'C%';
GO
```

```
CREATE PROCEDURE Selecthotelbyname @hotel_name varchar(max) , @hotel_rent int
AS
SELECT * FROM T10_hoteldetails WHERE hotel_name= @hotel_name AND
hotel_rent=@hotel_rent;
GO
```

```
EXEC Names_starts_with_C;
```

```
EXEC Selecthotelbyname @hotel_name='anil', @hotel_rent=500;
```

Output:



SQLQuery1.sql - C...HAITRA\admin (52))

```
CREATE PROCEDURE Names_starts_with_C
AS
SELECT * FROM T10_customerdetailsNEW WHERE customer_name LIKE 'C%';
GO

CREATE PROCEDURE Selecthotelbyname @hotel_name varchar(max) , @hotel_rent int
AS
SELECT * FROM T10_hoteldetails WHERE hotel_name= @hotel_name AND hotel_rent=@hotel_rent;
GO

EXEC Names_starts_with_C;

EXEC Selecthotelbyname @hotel_name='anil', @hotel_rent=500;
```

100 %

Results Messages

	customer_id	customer_name	customer_mobile	customer_email	customer_username	customer_password	customer_address	customer_final
1	1	CHAITRA	82096	chaitra@gmail.com	msaichaitra	little	kadapa	NULL
2	2	CHAITRA	80825	chaitra	chaitrareddy	msaichaitra	kadapa	NULL

	hotel_id	hotel_name	hotel_type	hotel_rent	hotel_description	hotel_address	booking_id
1	106	anil	double	500	null	dharwad	201

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

T10_packagedetails table before transaction:

	package_id	package_tour_id	package_name	package_amount	package_total	package_type	package_description	package_Final	hotel_id
1	301	401	koorg	500	500		null	500	204
2	302	402	ooty	500	500	vacation	null	500	106
3	303	403	pune	600	600	vacation	null	600	107
4	304	404	mysore	800	800	vacation	null	800	108
5	305	405	kodaikanal	700	700	vacation	NULL	700	109

**AFTER TRANSACTION (when given input is valid):
QUERY:**

```
USE T10_Travel;  
GO
```

```
BEGIN TRANSACTION
```

```
UPDATE T10_packagedetails SET package_name='ooty' WHERE  
package_id=302
```

```
INSERT INTO T10_packagedetails  
VALUES(306,406,'chail',600,600,'vacation',null,600,200)
```

```
COMMIT TRANSACTION
```

```
SELECT * FROM T10_packagedetails
```

OUTPUT:

```
SQLQuery1.sql - C:\HAITRA\admin (52)) *  X
```

```
USE T10_Travel;  
GO  
  
BEGIN TRANSACTION  
  
UPDATE T10_packagedetails SET package_name='ooty' WHERE package_id=302  
  
INSERT INTO T10_packagedetails  
VALUES(306,406,'chail',600,600,'vacation',null,600,200)  
  
COMMIT TRANSACTION  
  
SELECT * FROM T10_packagedetails
```

	package_id	package_tour_id	package_name	package_amount	package_total	package_type	package_description	package_Final	hotel_id
1	301	401	koorg	500	500	vacation	null	500	204
2	302	402	ooty	500	500	vacation	null	500	106
3	303	403	pune	600	600	vacation	null	600	107
4	304	404	mysore	800	800	vacation	null	800	108
5	305	405	kodaikanal	700	700	vacation	NULL	700	109
6	306	406	chail	600	600	vacation	NULL	600	200

AFTER TRANSACTION (when given input is not valid):

QUERY:

USE T10_Travel;

GO

BEGIN TRANSACTION

UPDATE T10_packagedetails SET package_name='ooty' WHERE
package_id=303

INSERT INTO T10_packagedetails
VALUES('truf',400000,504, null,'trip',null,600,200)

COMMIT TRANSACTION

SELECT * FROM T10_packagedetails

OUTPUT:

```
SQLQuery1.sql - C:\HAITRA\admin (52))* X
USE T10_Travel;
GO
BEGIN TRANSACTION
UPDATE T10_packagedetails SET package_name='ooty' WHERE package_id=303
INSERT INTO T10_packagedetails
VALUES('truf',400000,504,null,'trip',null,600,200)
COMMIT TRANSACTION
SELECT * FROM T10_packagedetails
```

100 %

Messages

Msg 213, Level 16, State 1, Line 5
Column name or number of supplied values does not match table definition.

Completion time: 2021-04-30T15:33:25.6249653+05:30

```
SQLQuery1.sql - C:\HAITRA\admin (52))* X
USE T10_Travel;
GO
BEGIN TRANSACTION
UPDATE T10_packagedetails SET package_name='ooty' WHERE package_id=303
INSERT INTO T10_packagedetails
VALUES('truf',400000,504,null,'trip',null,600,200)
COMMIT TRANSACTION
SELECT * FROM T10_packagedetails
```

100 %

Results Messages

	package_id	package_tour_id	package_name	package_amount	package_total	package_type	package_description	package_Final	hotel_id
1	301	401	koorg	500	500	vacation	null	500	204
2	302	402	ooty	500	500	vacation	null	500	106
3	303	403	pune	600	600	vacation	null	600	107
4	304	404	mysore	800	800	vacation	null	800	108
5	305	405	kodaikanal	700	700	vacation	NULL	700	109
6	306	406	chail	600	600	vacation	NULL	600	200

From here we can see that when error occurs in the transaction it gets rolled back. So the table did not get updated and shows previous values. It means all statements inside a transaction should either get succeed or fail as a unit. So this is the atomicity property.

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

Query (Window 1) :-

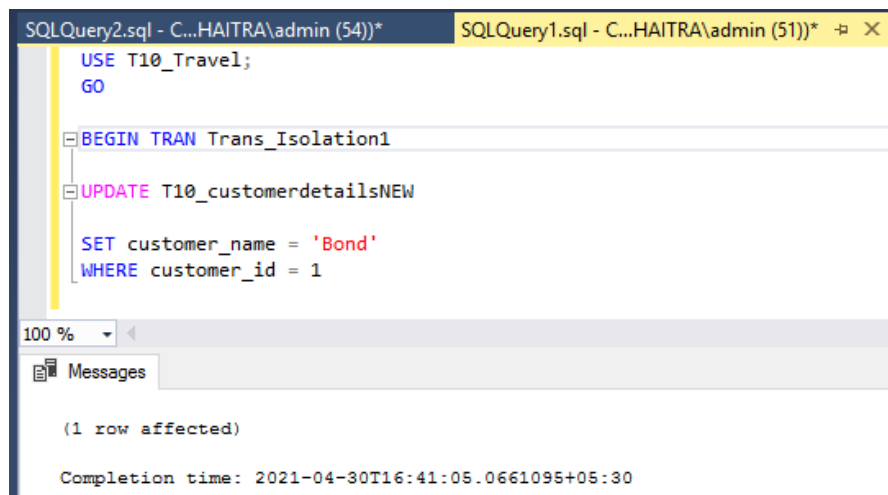
```
USE T10_Travel;  
GO
```

```
BEGIN TRAN Trans_Isolation1
```

```
UPDATE T10_customerdetailsNEW
```

```
SET customer_name = 'Bond'  
WHERE customer_id = 1
```

Output:-



```
SQLQuery2.sql - C:\...HAITRA\admin (54))*  SQLQuery1.sql - C:\...HAITRA\admin (51))* -+ X  
USE T10_Travel;  
GO  
BEGIN TRAN Trans_Isolation1  
UPDATE T10_customerdetailsNEW  
    SET customer_name = 'Bond'  
    WHERE customer_id = 1  
100 %  
Messages  
(1 row affected)  
Completion time: 2021-04-30T16:41:05.0661095+05:30
```

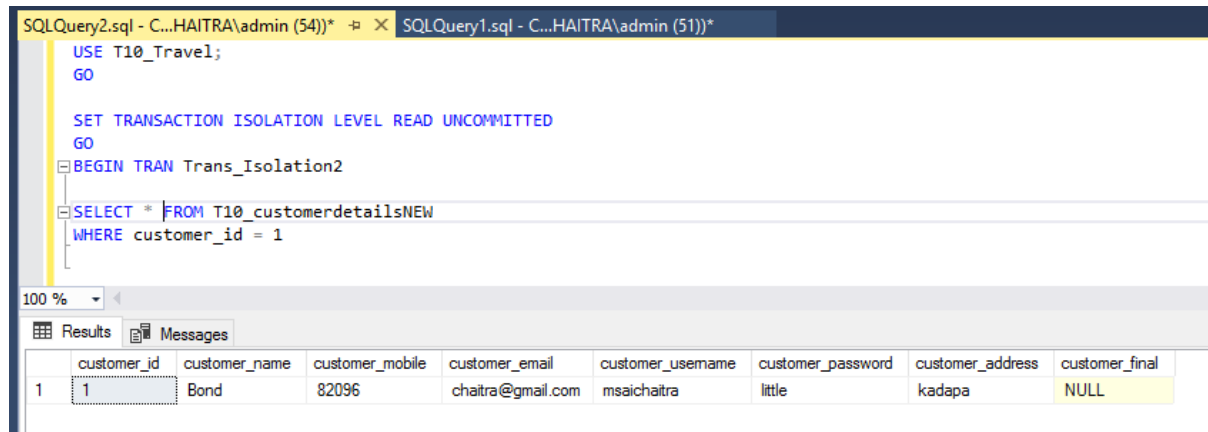
Query (Window 2) :-

```
USE T10_Travel;  
GO
```

```
SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED  
GO  
BEGIN TRAN Trans_Isolation2
```

```
SELECT * FROM T10_customerdetailsNEW
WHERE customer_id = 1
```

Output:



The screenshot shows a SQL Server Enterprise Manager interface. At the top, there are two tabs: 'SQLQuery2.sql - C:\...HAITRA\admin (54))' and 'SQLQuery1.sql - C:\...HAITRA\admin (51))'. The active window displays a SQL script with the following content:

```
USE T10_Travel;
GO

SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED
GO
BEGIN TRAN Trans_Isolation2
SELECT * FROM T10_customerdetailsNEW
WHERE customer_id = 1
```

Below the script, the 'Results' tab is selected, showing a table with 8 columns and 1 row. The columns are: customer_id, customer_name, customer_mobile, customer_email, customer_username, customer_password, customer_address, and customer_final. The row contains the values: 1, Bond, 82096, chaitra@gmail.com, msaichaitra, little, kadapa, and NULL.

customer_id	customer_name	customer_mobile	customer_email	customer_username	customer_password	customer_address	customer_final
1	Bond	82096	chaitra@gmail.com	msaichaitra	little	kadapa	NULL

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