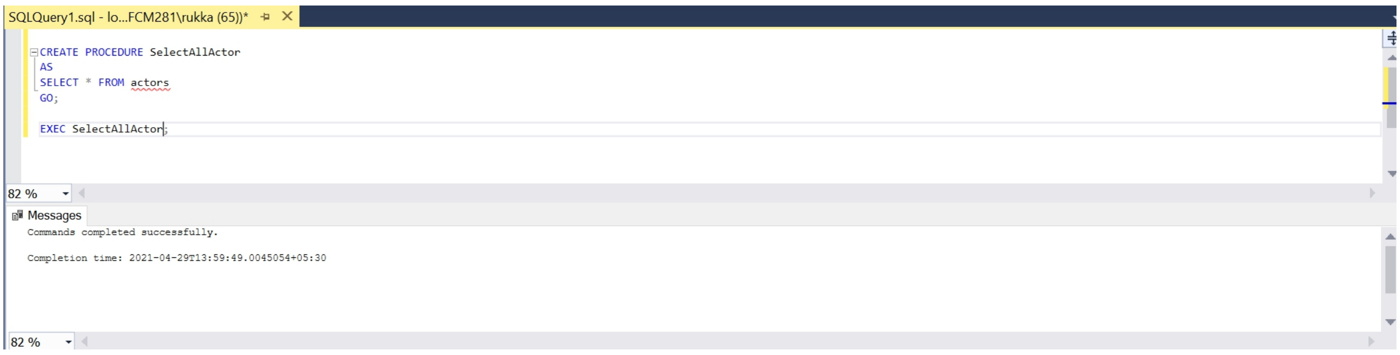
DBMS LAB ASSIGNMENT - 7

Name : K.V.S BHARADWAJ
Roll No : 19BCS047

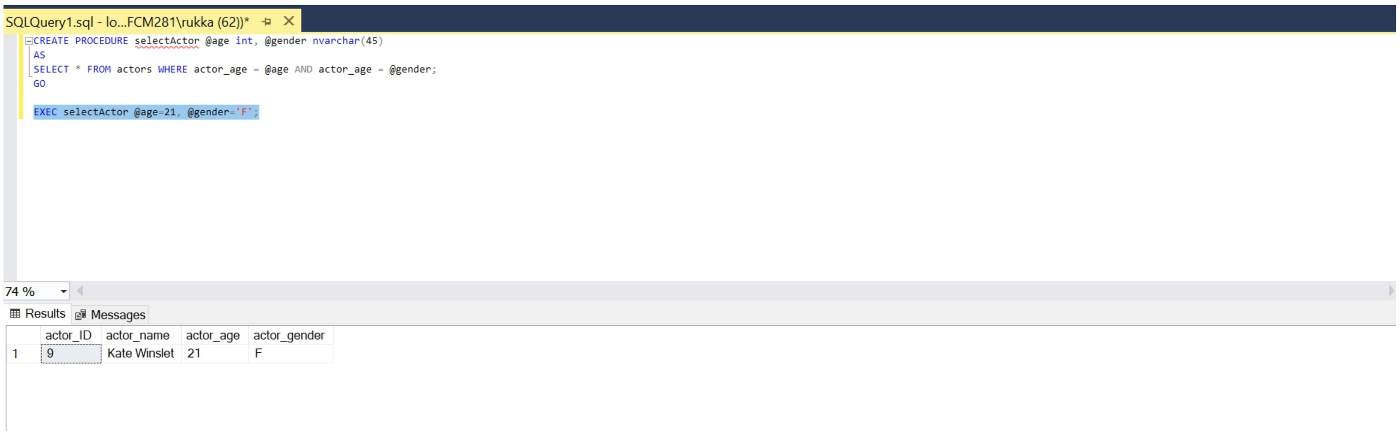
Q1) Write two stored Procedures relevant to your database.

OUTPUT :


SQLQuery1.sql - lo...FCM281\rukka (65))*

```
CREATE PROCEDURE SelectAllActor
AS
SELECT * FROM actors
GO;

EXEC SelectAllActor;
```

```
Commands completed successfully.

Completion time: 2021-04-29T13:59:49.0045054+05:30
```

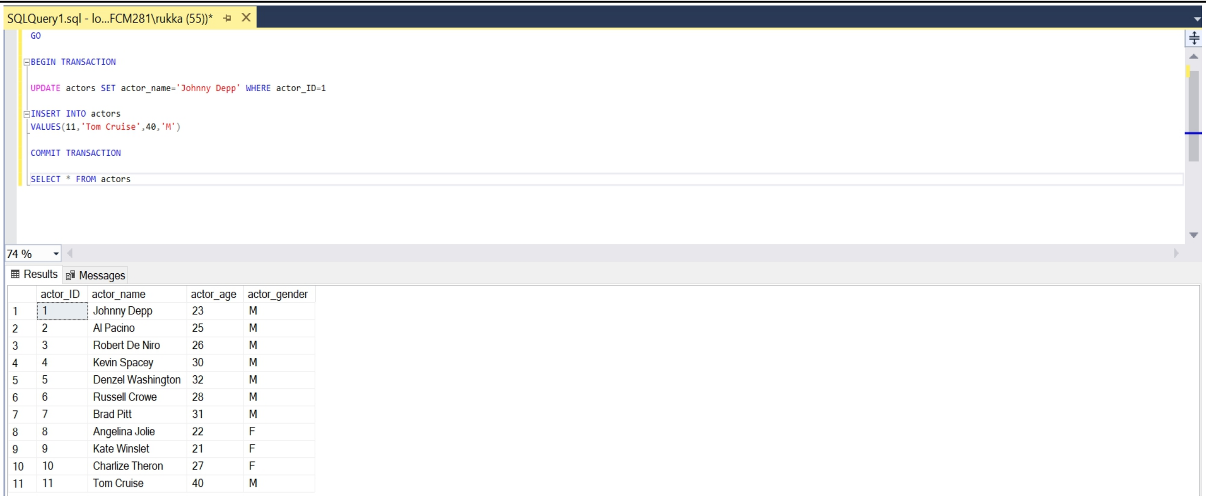

SQLQuery1.sql - lo...FCM281\rukka (62))*

```
CREATE PROCEDURE selectActor @age int, @gender nvarchar(45)
AS
SELECT * FROM actors WHERE actor_age = @age AND actor_age = @gender;
GO

EXEC selectActor @age=21, @gender='F';
```

	actor_ID	actor_name	actor_age	actor_gender
1	9	Kate Winslet	21	F

Q2) Write a transaction to illustrate atomicity (related to your database)

OUTPUT after transaction is successful


SQLQuery1.sql - lo...FCM281\rukka (55))*

```sql
GO

BEGIN TRANSACTION

UPDATE actors SET actor_name='Johnny Depp' WHERE actor_ID=1

INSERT INTO actors
VALUES(11,'Tom Cruise',40,'M')

COMMIT TRANSACTION

SELECT * FROM actors
```

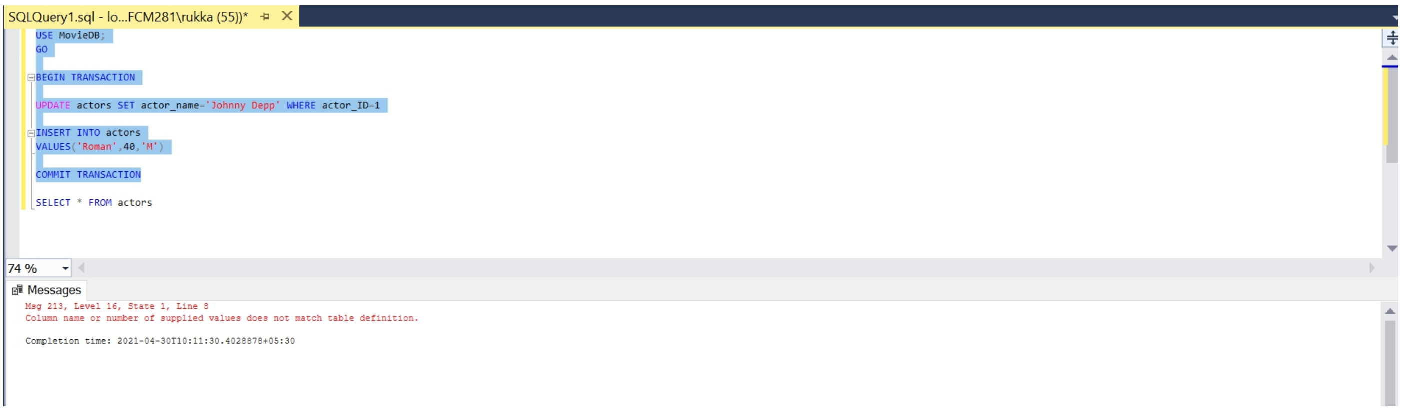
74 %

Results | Messages

	actor_ID	actor_name	actor_age	actor_gender
1	1	Johnny Depp	23	M
2	2	Al Pacino	25	M
3	3	Robert De Niro	26	M
4	4	Kevin Spacey	30	M
5	5	Denzel Washington	32	M
6	6	Russell Crowe	28	M
7	7	Brad Pitt	31	M
8	8	Angelina Jolie	22	F
9	9	Kate Winslet	21	F
10	10	Charlize Theron	27	F
11	11	Tom Cruise	40	M

This time we will insert wrong information in the Customers table to fail the insertion deliberately.

OUTPUT after transaction:

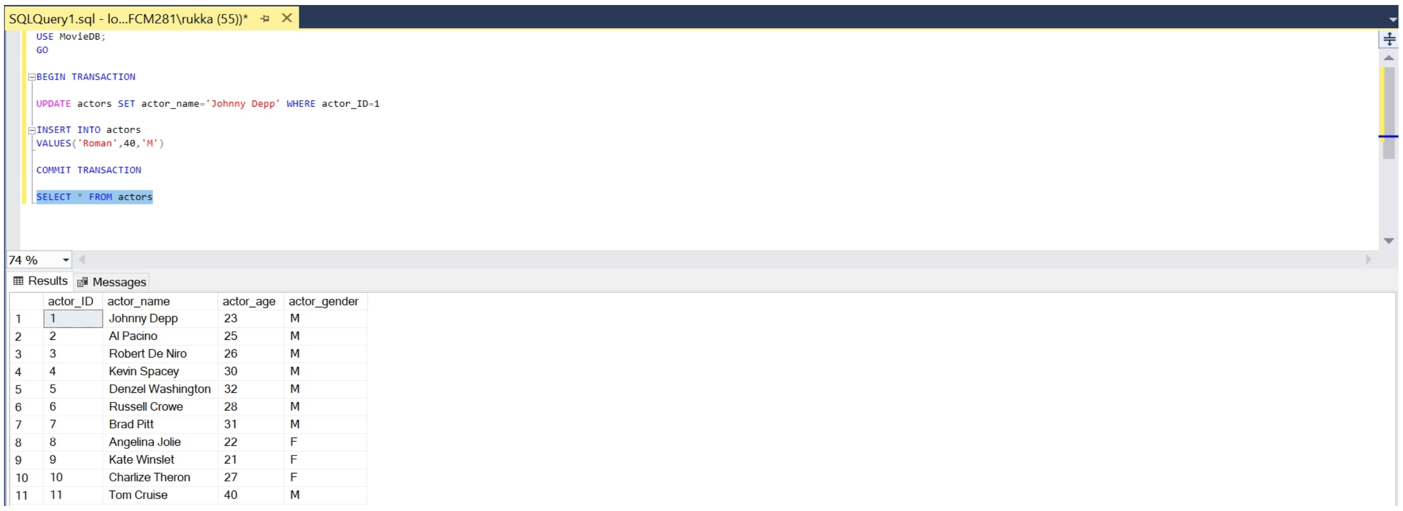

SQLQuery1.sql - lo...FCM281\rukka (55))*

```sql
USE MovieDB;
GO

BEGIN TRANSACTION

UPDATE actors SET actor_name='Johnny Depp' WHERE actor_ID=1

INSERT INTO actors
VALUES('Roman',40,'M')

COMMIT TRANSACTION

SELECT * FROM actors
```

74 %

Messages

```
Msg 213, Level 16, State 1, Line 8
Column name or number of supplied values does not match table definition.

Completion time: 2021-04-30T10:11:30.4028878+05:30
```


SQLQuery1.sql - lo...FCM281\rukka (55))*

```sql
USE MovieDB;
GO

BEGIN TRANSACTION

UPDATE actors SET actor_name='Johnny Depp' WHERE actor_ID=1

INSERT INTO actors
VALUES('Roman',40,'M')

COMMIT TRANSACTION

SELECT * FROM actors
```

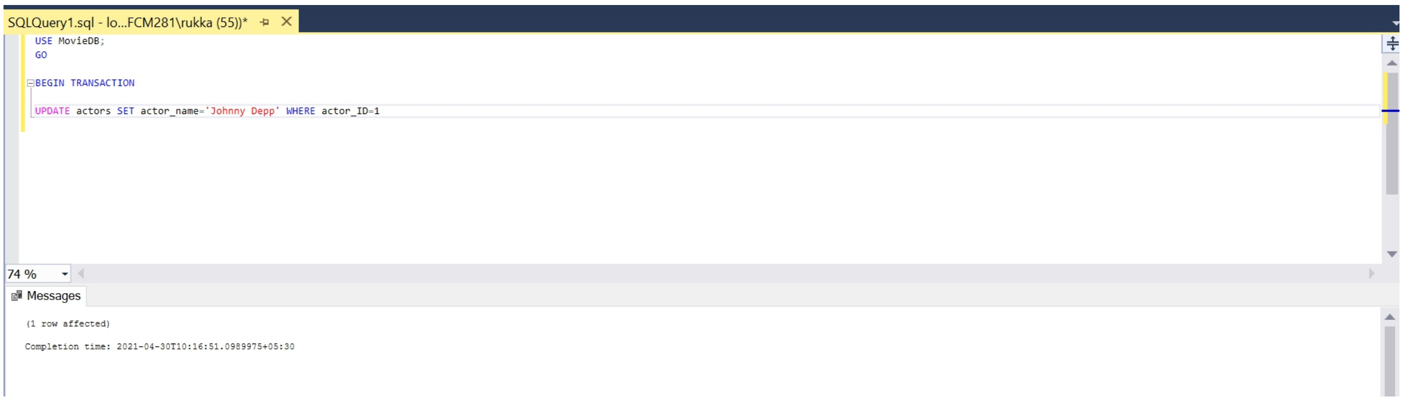
74 %

Results | Messages

	actor_ID	actor_name	actor_age	actor_gender
1	1	Johnny Depp	23	M
2	2	Al Pacino	25	M
3	3	Robert De Niro	26	M
4	4	Kevin Spacey	30	M
5	5	Denzel Washington	32	M
6	6	Russell Crowe	28	M
7	7	Brad Pitt	31	M
8	8	Angelina Jolie	22	F
9	9	Kate Winslet	21	F
10	10	Charlize Theron	27	F
11	11	Tom Cruise	40	M

As we could see here when error occurs in the transaction it got rolled back so the table has its previous values and didn't get updated.
It means all the statements inside a transaction should either succeed or fail as a unit. So this is the atomicity property.

Q3) Write a transaction to illustrate isolation level. It can be on commit or uncommitted (related to your database) ?

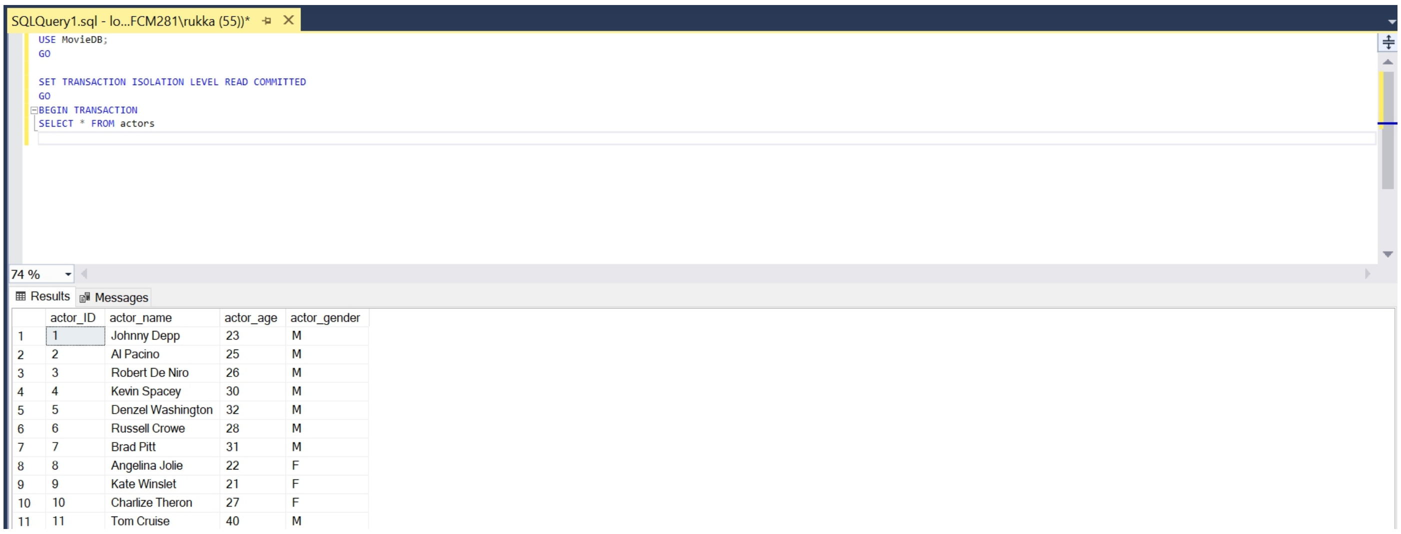

SQLQuery1.sql - lo...FCM281\rukka (55))*

```
USE MovieDB;
GO

BEGIN TRANSACTION

UPDATE actors SET actor_name='Johnny Depp' WHERE actor_ID=1
```

74 %

Messages

```
(1 row affected)

Completion time: 2021-04-30T10:16:51.0989975+05:30
```


SQLQuery1.sql - lo...FCM281\rukka (55))*

```
USE MovieDB;
GO

SET TRANSACTION ISOLATION LEVEL READ COMMITTED
GO
BEGIN TRANSACTION
SELECT * FROM actors
```

74 %

Results | Messages

	actor_ID	actor_name	actor_age	actor_gender
1	1	Johnny Depp	23	M
2	2	Al Pacino	25	M
3	3	Robert De Niro	26	M
4	4	Kevin Spacey	30	M
5	5	Denzel Washington	32	M
6	6	Russell Crowe	28	M
7	7	Brad Pitt	31	M
8	8	Angelina Jolie	22	F
9	9	Kate Winslet	21	F
10	10	Charlize Theron	27	F
11	11	Tom Cruise	40	M