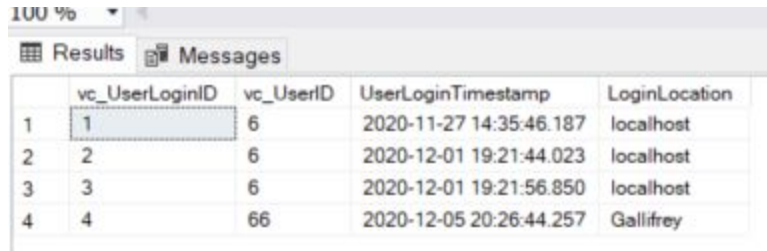


--Creating a guestuser database user
CREATE USER guestuser FOR LOGIN guestuser



The screenshot shows a SQL Server interface with the 'Results' tab active. It displays a table with four columns: 'vc_UserLoginID', 'vc_UserID', 'UserLoginTimestamp', and 'LoginLocation'. There are four rows of data. The first row is highlighted. The interface also shows a 'Messages' tab and a zoom level of 100%.

	vc_UserLoginID	vc_UserID	UserLoginTimestamp	LoginLocation
1	1	6	2020-11-27 14:35:46.187	localhost
2	2	6	2020-12-01 19:21:44.023	localhost
3	3	6	2020-12-01 19:21:56.850	localhost
4	4	66	2020-12-05 20:26:44.257	Gallifrey

SELECT * FROM vc_User

SELECT * FROM vc_MostProlificUsers

DECLARE @addedValue int
EXEC @addedValue = vc_AddUserLogin 'TheDoctor', 'Gallifrey'

EXEC vc_FinishVidCast @newVC

--Creating a guestuser database user
CREATE USER guestlogin FOR LOGIN guestlogin

--Grant read permission on the user table
GRANT SELECT ON vc_User to guestlogin

--Revoke the select permission
REVOKE SELECT ON vc_User to guestlogin

--Give them the view instead
GRANT SELECT ON vc_MostProlificUsers to guestlogin

--Allow guest user to run some stored procedures
GRANT EXECUTE ON vc_AddUserLogin TO guestlogin

GRANT EXECUTE ON vc_FinishVidCast TO guestlogin

DECLARE @newVC int
INSERT INTO vc_Vidcast
 (VidCastTitle, StartDateTime, ScheduleDurationMinutes, vc_UserID, vc_StatusID)
VALUES (
 'Rock Your Way To Success'
 , DATEADD(n, -45, GETDATE())

```
, 45
, (SELECT vc_UserID FROM vc_User WHERE UserName = 'tardy')
, (SELECT vc_StatusID FROM vc_Status WHERE StatusText = 'Started')
)
```

```
SET @newVC = @@identity
```

```
SELECT * FROM vc_UserLogin
```

Results Messages	
(No column name)	
1	Yay! It worked!

	lab_LogID	lab_Joint
1	1	1
2	3	2
3	2	3
4	4	5

	lab_TestID	lab_testText
1	1	One
2	5	Picon
3	3	Three
4	2	Two

The first failed because when we tried to insert the value 'One' in row 34, it did not add it because it already existed. Once the transaction code ran, there was no change to the row count, but once i added my last name on row 34, that was a new value and the row count actually changed.

```
-- Creating a new table
```

```
CREATE TABLE lab_Test(
    lab_TestID int identity primary key,
    lab_testText varchar(20) unique not null,
)
```

```
/*
```

This will be a table to keep a log of
created lab_Test records.

We don't want to add a row to this if the insert into
lab_test fails.

```
*/
```

```
CREATE TABLE lab_log (  
    lab_LogID int identity primary key,  
    lab_loInt int unique not null  
)
```

```
INSERT INTO lab_Test (lab_testText) VALUES ('One'), ('Two'), ('Three')  
INSERT INTO lab_Log (lab_loInt) SELECT lab_TestID FROM lab_Test
```

--Step 1: Begin the transaction

```
BEGIN TRANSACTION
```

--Step 2: Access the state of things

```
DECLARE @rc int
```

```
SET @rc = @@ROWCOUNT --Initially 0
```

--Step 3: Make the change

--On success, @@ROWCOUNT is incremental by 1

--On failure, @@ROWCOUNT does not change

```
INSERT INTO lab_test (lab_testText) VALUES ('Picon')
```

--Step 4: Check the new state of things

```
IF(@rc = @@ROWCOUNT)--If @@ROWCOUNT was not changed, fail  
BEGIN
```

--Step 5, if failed

```
SELECT 'Bail out! It Failed!'
```

```
ROLLBACK
```

```
END
```

```
ELSE -- Success! Continue.
```

```
BEGIN
```

-- Step 5 if succeeded

```
SELECT 'Yay! It worked!'
```

```
INSERT INTO lab_Log (lab_loInt) VALUES (@@identity)
```

```
COMMIT
```

```
END
```

```
SELECT * FROM lab_Log
```

```
SELECT * FROM lab_Test
```