Course: IST-659
Name: Sharat Sripada
Homework #8
Date Submitted: 6/4/2020
Topic: Lab08 – Database Programming

Part 1 – Introducing Functions, Views, and Stored Procedures

Functions

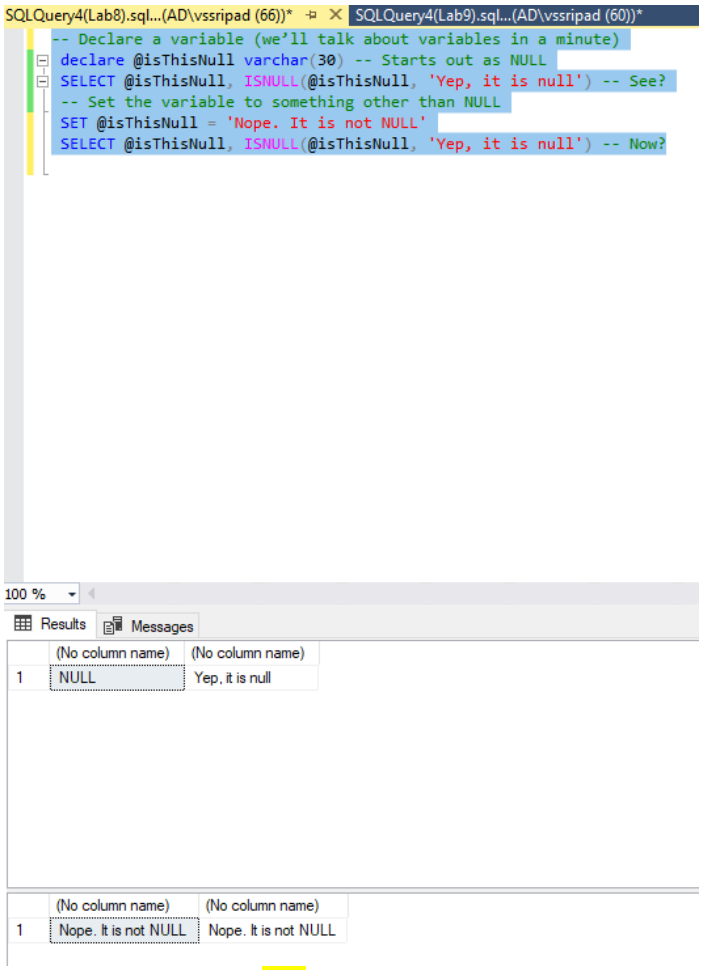


Fig: Is null usage

```sql
-- My first user-defined function in SQL
CREATE FUNCTION dbo.AddTwoInts(@firstnumber int, @secondnumber int)
RETURNS int AS
BEGIN
    DECLARE @returnvalue int
    SET @returnvalue = @firstnumber + @secondnumber
    RETURN @returnvalue
END
GO

SELECT dbo.AddTwoInts(5,10)
```

100 %

Results | Messages

	(No column name)
1	15

Fig: My first-function

Functions That Are More... well... Functional

```sql
38 SELECT TOP 10
39     *
40     , dbo.VidCastCount(vc_UserID) as VidCastCount
41 FROM vc_User
42 ORDER BY VidCastCount DESC
43
44 SELECT
45     *
46     , dbo.VidCastCount(vc_UserID) as VidCastCount
47 FROM vc_User
48 ORDER BY VidCastCount DESC
49
```

00 %

Results | Messages

	vc_UserID	UserName	EmailAddress	UserDescription	WebSiteURL	UserRegisteredDate	VidCastCount
1	20	ecstatic	blandit.enim.consequat@loremutaliquam.co.uk	Bandwidth series A financing niche market.	NULL	2017-11-16 00:00:00.000	22
2	40	principle	ac.uma@miac.com	Business-to-business ecosystem ramen social media...	http://principle.vidcast659.site	2017-11-01 12:14:24.000	19
3	24	metacarpal	et.magna.Praesent@placerataugueSed.org	Research & development startup long tail strategy g...	http://metacarpal.vidcast659.site	2017-05-30 11:31:12.000	18
4	43	canadian	Curabitur.dictum.Phasellus@eleifendnec.com	Agile development ownership business-to-consumer...	http://canadian.vidcast659.site	2017-06-27 05:16:48.000	18
5	26	przewalski	amet@Maurismolestie.org	NULL	http://przewalski.vidcast659.site	2017-02-11 08:52:48.000	17
6	36	silly	accumsan@gravidasagittisDuis.net	Stock founders early adopters low hanging fruit A/B...	http://silly.vidcast659.site	2017-05-25 14:38:24.000	17
7	7	wood	turpis.egestas.Fusce@massanonante.net	Technology investor marketing alpha.	http://wood.vidcast659.site	2017-06-21 15:36:00.000	16
8	11	doughnut	ipsum.primis@Cumsociis.com	Assets sales incubator user experience ecosystem ...	http://doughnut.vidcast659.site	2017-01-31 01:12:00.000	16
9	14	groggy	ornare.In.faucibus@egestas.ca	Sales niche market user experience investor social ...	http://groggy.vidcast659.site	2017-04-20 09:50:24.000	16
10	47	these	parturient.montes@ipsum.ca	Entrepreneur virality freemium crowdsource long tail ...	http://these.vidcast659.site	2017-12-07 03:50:24.000	16

Fig: VidCastCount function

In your own words, in your answers document, describe what lines 49 through 53 above actually do. Also, how is it that this code knows that the vc_User record with vc_UserID = 20 has 22 vc_VidCast records?

The SELECT statement above essentially calls the function 'VidCastCount' and provides the vc_UserID from table vc_User as an argument/parameter. The vc_UserID is scoped then to the local variable userid, which is used in the SELECT statement within it:

```
SELECT @returnvalue = COUNT(VidCastTitle) from vc_VidCast

        WHERE vc_UserID=@userid
```

Finally, the data is sorted in descending order & then the TOP 10 records are shown.

We can code a function that accepts the tag text as a parameter and looks up the vc_TagID for the vc_Tag record for that TagText.

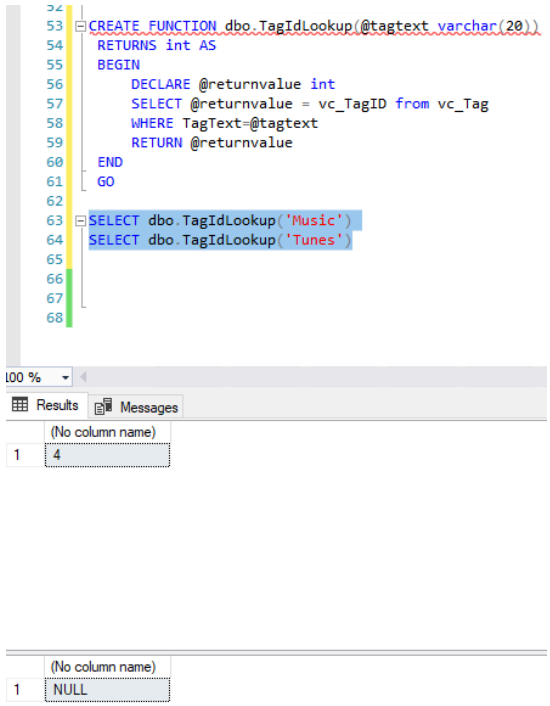


Fig: TagIDlookup function

In your own words, in your answers document, describe what lines 75 and 76 above actually do. Also, when line 76 executed, we received a NULL from SQL Server. How come?

First, the function TagIDlookup is written to take the TagText as input and return the corresponding vc_TagID as seen below:

```
SELECT @returnvalue = vc_TagID from vc_Tag
        WHERE TagText=@tagtext
```

Value vc_TagID is then returned as a result. With respect to the NULL, when examining the vc_Tag table we see TagText = 'Tunes' is NOT PRESENT.

Views

```
69  -- Views
70  CREATE VIEW vc_MostProlificUsers AS
71      SELECT TOP 10
72       *
73       , dbo.VidCastCount(vc_UserID) as VidCastCount
74      FROM vc_User
75      ORDER BY VidCastCount DESC
76  GO
77
78  SELECT * FROM vc_MostProlificUsers
```

Results | Messages

	vc_UserID	UserName	EmailAddress	UserDescription	WebSiteURL	UserRegisteredDate	VidCastCount
1	20	ecstatic	blandit.enim.consequat@loremutaliquam.co.uk	Bandwidth series A financing niche market.	NULL	2017-11-16 00:00:00.000	22
2	40	principle	ac.uma@miac.com	Business-to-business ecosystem ramen social media...	http://principle.vidcast659.site	2017-11-01 12:14:24.000	19
3	24	metacarpal	et.magna.Praesent@placerataugueSed.org	Research & development startup long tail strategy g...	http://metacarpal.vidcast659.site	2017-05-30 11:31:12.000	18
4	43	canadian	Curabitur.dictum.Phasellus@elelifendnec.com	Agile development ownership business-to-consumer...	http://canadian.vidcast659.site	2017-06-27 05:16:48.000	18
5	26	przewalski	amet@Maurismolestie.org	NULL	http://przewalski.vidcast659.site	2017-02-11 08:52:48.000	17
6	36	silly	accumsan@gravidasagittisDuis.net	Stock founders early adopters low hanging fruit A/B...	http://silly.vidcast659.site	2017-05-25 14:38:24.000	17
7	7	wood	turpis.egestas.Fusce@massanonante.net	Technology investor marketing alpha.	http://wood.vidcast659.site	2017-06-21 15:36:00.000	16
8	11	doughnut	ipsum.primis@Cumsociis.com	Assets sales incubator user experience ecosystem ...	http://doughnut.vidcast659.site	2017-01-31 01:12:00.000	16
9	14	groggy	ornare.In.faucibus@egestas.ca	Sales niche market user experience investor social ...	http://groggy.vidcast659.site	2017-04-20 09:50:24.000	16
10	47	these	parturient.montes@ipsum.ca	Entrepreneur virality freemium crowdsource long tail ...	http://these.vidcast659.site	2017-12-07 03:50:24.000	16

Fig: MostProlificUsers View

In your own words, in your answers document, describe what lines 79 through 87 above are doing.

We are creating a View and placing within a SELECT statement that calls a function (here dbo.VidCastCount) which was previously defined.

Stored Procedures

Stored procedures vs functions

Stored procedures are like functions in that they perform operations based on provided parameter values, but they are different in that they can perform more complex database activities. For instance, whereas a user-defined function can make no changes to the database in any way, a stored procedure can be used to encapsulate and abstract statements such as `INSERT`, `UPDATE`, and `DELETE`.

```
80  -- Stored procedures
81  CREATE PROCEDURE vc_ChangeUserEmail(@username varchar(20), @newemail varchar(50))
82  AS
83  BEGIN
84      UPDATE vc_User SET EmailAddress = @newemail
85      WHERE UserName = @username
86  END
87  GO
88
89  EXEC vc_ChangeUserEmail 'tardy', 'kmstudent@syr.edu'
90
91  SELECT * FROM vc_User where UserName = 'tardy'
```

100 %

Results | Messages

	vc_UserID	UserName	EmailAddress	UserDescription	WebSiteURL	UserRegisteredDate
1	6	tardy	kmstudent@syr.edu	Startup leverage growth hacking bootstrapping sc...	http://tardy.vidcast659.site	2017-03-12 15:36:00.000

Fig: Update email using Stored procedure

In your own words, in your answers document, describe what lines 91 through 104 above are doing.

The stored procedure vc_ChangeUserEmail takes two parameters username and newemail & within it implements an update carefully using the WHERE clause.

@@Identity

```
93  -- @@Identity
94  INSERT INTO vc_Tag (TagText) VALUES ('Cat Videos')
95  SELECT * FROM vc_Tag where vc_TagID = @@IDENTITY
96
```

100 %

Results | Messages

	vc_TagID	TagText	TagDescription
1	15	Cat Videos	NULL

Fig: @@Identity example

```
111 DECLARE @addedValue int
112     EXEC @addedValue = vc_AddUserLogin 'tardy', 'localhost'
113 SELECT
114     vc_User.vc_UserID
115     , vc_User.UserName
116     , vc_UserLogin.UserLoginTimestamp
117     , vc_UserLogin.LoginLocation
118 FROM vc_User
119 JOIN vc_UserLogin on vc_User.vc_UserID = vc_USerLogin.vc_UserID
120 WHERE vc_UserLoginID = @addedValue
121
122
123
124
```

100 %

Results | Messages

	vc_UserLoginID	vc_UserID	UserLoginTimestamp	LoginLocation
1	1	6	2020-06-03 06:21:34.193	localhost

Your UserLoginTimestamp value will be different than the one shown. On your answers doc, explain why this is.

The UserLoginTimestamp has data-type datetime. This means, each time a new userlo

On your answers doc, also identify one way we could simplify the code in the stored procedure above. (Hint: Look back at how we did a lookup with the vc_Tag table)

Part 2 – Putting all together

Coding your own Functions

Custom Function-1

The code below is the beginning of a function intended to retrieve a vc_UserID from the vc_User table given a specified @userName. Complete the code to assign the correct vc_UserID to @returnValue

```
-- Coding your own functions
-- Function to get UserID from table given a username
CREATE FUNCTION dbo.vc_UserIDLookup(@userName varchar(20))
RETURNS int AS
BEGIN
    DECLARE @returnValue int
    SELECT @returnValue = vc_UserID FROM vc_User
    WHERE UserName = @userName

    RETURN @returnValue
END
GO

SELECT 'Trying the vc_UserIDLookup function.', dbo.vc_UserIDLookup('tardy')
```

00 %

Results | Messages

	(No column name)	(No column name)
1	Trying the vc_UserIDLookup function.	6

Fig: Function to return vc_UserID given UserName

Custom Function-2

Author a function called dbo.vc_TagVidCastCount that calculates the count of vc_VidCastIDs for a given vc_TagID. Consult the diagram at the end of this document as a reference for the tables involved.

```sql
-- Function to count vc_VidCastIDs for a given vc_TagID
CREATE FUNCTION dbo.vc_TagVidCastCount(@tagID int)
RETURNS int AS
BEGIN
    DECLARE @returnValue int
    SELECT @returnValue = count(vc_VidCastID) FROM vc_VidCastTagList
    WHERE vc_T
    RETURN @returnValue
END
GO

SELECT
    vc_Tag.TagText
    , dbo.vc_TagVidCastCount(vc_Tag.vc_TagID) as VidCasts
    FROM vc_Tag
```

local variable @returnValue int

Results | Messages

	TagText	VidCasts
1	Art	256
2	Audio Recording	266
3	Baseball	242
4	Basketball	236
5	Cat Videos	0
6	Collectibles	258
7	Consoles	260
8	Fashion	240
9	Football	259
10	Games	254
11	Motors	0
12	Music	237
13	Personal	263
14	Professional	264
15	Sports - General	235

Fig: Function to return vc_UserID given UserName

Custom Function-3

Code a function called vc_VidCastDuration that SUMs the total number of minutes of actual duration for VidCasts with a Finished status given a vc_UserID as a parameter. This function should return the SUM as an int.

```sql
CREATE FUNCTION dbo.vc_VidCastDuration(@userid int)
RETURNS INT AS
BEGIN
    DECLARE @returnValue INT
    SELECT @returnValue = SUM(DATEDIFF(n, vc_VidCast.StartDateTime, vc_VidCast.EndDateTime)) FROM vc_VidCast
    JOIN vc_Status ON vc_Status.vc_StatusID = vc_VidCast.vc_StatusID
    WHERE vc_Status.StatusText = 'Finished' AND
      vc_VidCast.vc_UserID = @userid
    RETURN @returnValue
END
GO

SELECT
    *
    , dbo.vc_VidCastDuration(vc_UserID) as TotalMinutes
FROM vc_User
```

Results | Messages

	vc_UserID	UserName	EmailAddress	UserDescription	WebSiteURL	UserRegisteredDate	TotalMinutes
1	1	ethanol	Donec.tempus@penatibusetmagnis.co.uk	Agile development non-disclosure agreement equity ...	http://ethanol.vidcast659.site	2017-12-30 22:19:12.000	1859
2	2	dispatcher	quam@aptenttacitisociosqu.ca	A/B testing handshake disruptive seed money infogr...	http://dispatcher.vidcast659.site	2017-12-08 03:36:00.000	1368
3	3	camel	mauris@massanon.edu	User experience founders branding entrepreneur iter...	http://camel.vidcast659.site	2017-08-14 03:21:36.000	1859
4	4	infatuated	mollis@Nam.org	Lean startup launch party angel investor branding b...	http://infatuated.vidcast659.site	2017-06-07 17:02:24.000	1426
5	5	hygienist	magna.Ut@necumasuscipit.ca	Business model canvas accelerator pivot network ef...	http://hygienist.vidcast659.site	2017-03-17 23:16:48.000	1139
6	6	tardy	kmstudent@syr.edu	Startup leverage growth hacking bootstrapping scru...	http://tardy.vidcast659.site	2017-03-12 15:36:00.000	2303
7	7	wood	turpis.egestas.Fusce@massanonante.net	Technology investor marketing alpha.	http://wood.vidcast659.site	2017-06-21 15:36:00.000	2292
8	8	mallard	vel.lectus.Cum@velliteget.edu	Assets sales success bandwidth business model ca...	http://mallard.vidcast659.site	2017-09-15 19:55:12.000	1828
9	9	lifted	eu@elitsed.net	NULL	http://lifted.vidcast659.site	2017-04-15 20:24:00.000	1828
10	10	gum	ut@pharetraQuisqueac.com	Infographic incubator hypotheses client conversion ...	http://gum.vidcast659.site	2017-02-24 09:07:12.000	1238
11	11	doughnut	ipsum.primis@Cumsociis.com	Assets sales incubator user experience ecosystem a...	http://doughnut.vidcast659.site	2017-01-31 01:12:00.000	1830
12	12	bewildered	Donec.porttitor.tellus@odioAliquamvulputate.edu	Infrastructure research & development venture burn ...	http://bewildered.vidcast659.site	2017-12-29 09:07:12.000	1944
13	13	albite	nisi@vitaemauris.org	Learning curve partnership buzz value proposition re...	http://albite.vidcast659.site	2017-07-25 00:00:00.000	1971

Fig: Function to return SUM of total minutes for Vidcasts given a UserID

Coding your own Views

Custom View-1

To fetch details from Vidcast utilizing function vc_TagVidCastCount

```sql
CREATE VIEW vc_TagReport AS
    SELECT
        vc_Tag.TagText
        , dbo.vc_TagVidCastCount(vc_Tag.vc_TagID) as VidCasts
    FROM vc_Tag
GO
SELECT * FROM vc_TagReport
ORDER BY VidCasts DESC
```

100 %

Results | Messages

	TagText	VidCasts
1	Audio Recording	266
2	Professional	264
3	Personal	263
4	Consoles	260
5	Football	259
6	Collectibles	258
7	Art	256
8	Games	254
9	Baseball	242
10	Fashion	240
11	Music	237
12	Basketball	236
13	Sports - General	235
14	Motors	0
15	Cat Videos	0

Fig: View to call function vc_TagVidCastCount

Custom View-2

Alter the view called vc_MostProlificUsers, adding a column called TotalMinutes that calls the vc_VidCastDuration function we created earlier in part 2.

```sql
-- Alter VIEW vc_MostProlificUsers
SELECT * FROM vc_MostProlificUsers
ALTER VIEW vc_MostProlificUsers AS
    SELECT TOP 10
        *
        , dbo.VidCastCount(vc_UserID) as VidCastCount
        , dbo.vc_VidCastDuration(vc_UserID) as TotalMinutes
    FROM vc_User
    ORDER BY VidCastCount DESC
GO

SELECT UserName, VidCastCount, TotalMinutes FROM vc_MostProlificUsers
```

100 %

Results | Messages

	UserName	VidCastCount	TotalMinutes
1	ecstatic	22	2682
2	principle	19	3413
3	canadian	18	1928
4	metacarpal	18	3053
5	przewalski	17	2664
6	silly	17	2851
7	archives	16	2374
8	doughnut	16	1830
9	groggy	16	2464
10	sines	16	2316

Fig: Alter View vc_MostProlificUsers

Coding Your Own Stored Procedures

Custom Stored Procedure-1

```sql
CREATE PROCEDURE vc_AddTag(@tagText varchar(20), @description varchar(100)=NULL) AS
BEGIN
    INSERT INTO vc_Tag (TagText, TagDescription)
    VALUES (@tagText, @description)
    RETURN @@identity
END
GO

DECLARE @newTagID int
EXEC @newTagID = vc_AddTag 'SQL', 'Finally, a SQL Tag'
SELECT * FROM vc_Tag where vc_TagID = @newTagID
```

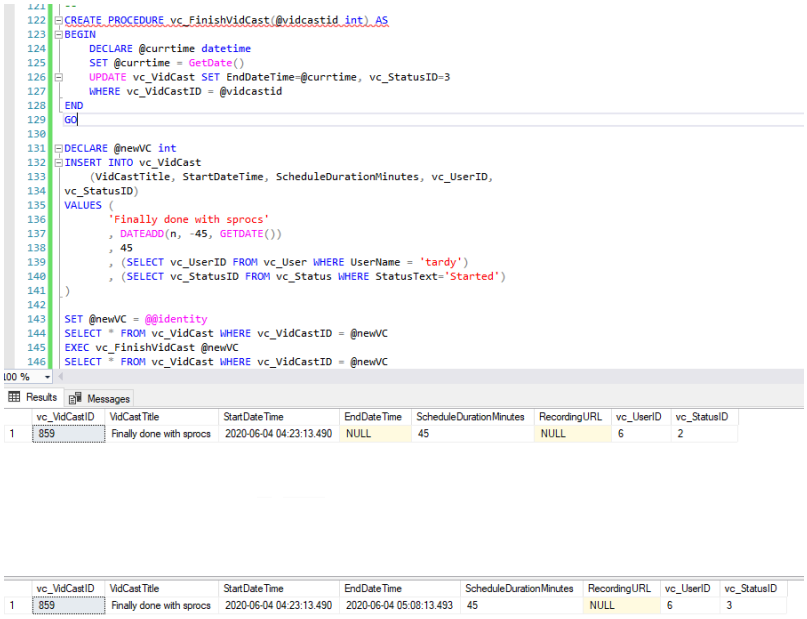
00 %

Results | Messages

	vc_TagID	TagText	TagDescription
1	17	SQL	Finally, a SQL Tag

Fig: Procedure to insert a value into vc_Tag

Custom Stored Procedure-2



```
--
CREATE PROCEDURE vc_FinishVidCast(@vidcastid int) AS
BEGIN
    DECLARE @currtime datetime
    SET @currtime = GetDate()
    UPDATE vc_VidCast SET EndDateTime=@currtime, vc_StatusID=3
    WHERE vc_VidCastID = @vidcastid
END
GO

DECLARE @newVC int
INSERT INTO vc_VidCast
    (VidCastTitle, StartDateTime, ScheduleDurationMinutes, vc_UserID,
vc_StatusID)
VALUES (
        'Finally done with sprocs'
        , 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_VidCast WHERE vc_VidCastID = @newVC
EXEC vc_FinishVidCast @newVC
SELECT * FROM vc_VidCast WHERE vc_VidCastID = @newVC
```

Results | Messages

	vc_VidCastID	VidCastTitle	StartDateTime	EndDateTime	ScheduleDurationMinutes	RecordingURL	vc_UserID	vc_StatusID
1	859	Finally done with sprocs	2020-06-04 04:23:13.490	NULL	45	NULL	6	2

	vc_VidCastID	VidCastTitle	StartDateTime	EndDateTime	ScheduleDurationMinutes	RecordingURL	vc_UserID	vc_StatusID
1	859	Finally done with sprocs	2020-06-04 04:23:13.490	2020-06-04 05:08:13.493	45	NULL	6	3

Fig: Modify vc_Vidcast to mark Status as Finished and update timestamp

Appendix – SQL Queries Lab08

Kept losing connection to the remote-desktop server and hence lost some lines of code.

```
-- Declare a variable (we'll talk about variables in a minute)
 declare @isThisNull varchar(30) -- Starts out as NULL
 SELECT @isThisNull, ISNULL(@isThisNull, 'Yep, it is null') -- See?
 -- Set the variable to something other than NULL
 SET @isThisNull = 'Nope. It is not NULL'
 SELECT @isThisNull, ISNULL(@isThisNull, 'Yep, it is null')

 -- My first user-defined function in SQL
 CREATE FUNCTION dbo.AddTwoInts(@firstnumber int, @secondnumber int)
 RETURNS int AS
 BEGIN
     DECLARE @returnvalue int
        SET @returnvalue = @firstnumber + @secondnumber
        RETURN @returnvalue
 END
 GO

 SELECT dbo.AddTwoInts(5,10)

 --Our VidCast service is interested in the number of VidCasts created by VidCast users

 -- First, let's write the select statement
 SELECT COUNT(vc_VidCast.VidCastTitle) from vc_VidCast

 CREATE FUNCTION dbo.VidCastCount(@userid int)
 RETURNS int AS
 BEGIN
```

```sql
    DECLARE @returnvalue int
    SELECT @returnvalue = COUNT(VidCastTitle) from vc_VidCast
       WHERE vc_UserID=@userid
       RETURN @returnvalue
END
GO

-- Example
SELECT dbo.VidCastCount(34)

SELECT TOP 10
    *
       , dbo.VidCastCount(vc_UserID) as VidCastCount
FROM vc_User
ORDER BY VidCastCount DESC

SELECT
    *
       , dbo.VidCastCount(vc_UserID) as VidCastCount
FROM vc_User
ORDER BY VidCastCount DESC

-- We can code a function that accepts the tag text as a parameter and looks up the
vc_TagID for the vc_Tag record for that TagText.
CREATE FUNCTION dbo.TagIdLookup(@tagtext varchar(20))
RETURNS int AS
BEGIN
    DECLARE @returnvalue int
    SELECT @returnvalue = vc_TagID from vc_Tag
       WHERE TagText=@tagtext
       RETURN @returnvalue
END
GO

SELECT dbo.TagIdLookup('Music')
SELECT dbo.TagIdLookup('Tunes')

-- Part-2: Putting it all together
-- Coding your own functions
-- Function to get UserID from table given a username
CREATE FUNCTION dbo.vc_UserIDLookup(@userName varchar(20))
RETURNS int AS
BEGIN
    DECLARE @returnValue int
       SELECT @returnValue = vc_UserID FROM vc_User
       WHERE UserName = @userName

       RETURN @returnValue
END
GO

SELECT 'Trying the vc_UserIDLookup function.', dbo.vc_UserIDLookup('tardy')

-- LOST SOME WORK HERE...SOB! SOB! --

-- Coding your own Views
-- User defined View-1
CREATE VIEW vc_TagReport AS
```

```sql
     SELECT
         vc_Tag.TagText
         , dbo.vc_TagVidCastCount(vc_Tag.vc_TagID) as VidCasts
     FROM vc_Tag
GO
SELECT * FROM vc_TagReport
ORDER BY VidCasts DESC

-- Alter VIEW vc_MostProlificUsers
SELECT * FROM vc_MostProlificUsers
ALTER VIEW vc_MostProlificUsers AS
    SELECT TOP 10
           *
              , dbo.VidCastCount(vc_UserID) as VidCastCount
              , dbo.vc_VidCastDuration(vc_UserID) as TotalMinutes
       FROM vc_User
       ORDER BY VidCastCount DESC
GO

SELECT UserName, VidCastCount, TotalMinutes FROM vc_MostProlificUsers

-- Coding Your Own Stored Procedures

-- Inserting a value into the vc_Tag table
CREATE PROCEDURE vc_AddTag(@tagText varchar(20), @description varchar(100)=NULL) AS
BEGIN
    INSERT INTO vc_Tag (TagText, TagDescription)
       VALUES (@tagText, @description)
    RETURN @@identity
END
GO

DECLARE @newTagID int
EXEC @newTagID = vc_AddTag 'SQL', 'Finally, a SQL Tag'
SELECT * FROM vc_Tag where vc_TagID = @newTagID

--
CREATE PROCEDURE vc_FinishVidCast(@vidcastid int) AS
BEGIN
    DECLARE @currtime datetime
       SET @currtime = GetDate()
    UPDATE vc_VidCast SET EndDateTime=@currtime, vc_StatusID=3
       WHERE vc_VidCastID = @vidcastid
END
GO

DECLARE @newVC int
INSERT INTO vc_VidCast
    (VidCastTitle, StartDateTime, ScheduleDurationMinutes, vc_UserID,
vc_StatusID)
VALUES (
       'Finally done with sprocs'
          , DATEADD(n, -45, GETDATE())
          , 45
          , (SELECT vc_UserID FROM vc_User WHERE UserName = 'tardy')
          , (SELECT vc_StatusID FROM vc_Status WHERE StatusText='Started')
)
```

```sql
SET @newVC = @@identity
SELECT * FROM vc_VidCast WHERE vc_VidCastID = @newVC
EXEC vc_FinishVidCast @newVC
SELECT * FROM vc_VidCast WHERE vc_VidCastID = @newVC
```