

# IT Assignment Coversheet

**Course**: PROG8080 – Database Management

Program Coordinator: David Allison

Professor/Instructor: Mark Morell

Assignment #: Choose from list

Assignment Type:  Individual  Pair  Team

Date Submitted:

# **Student Information**

|  |  |
| --- | --- |
| Name | Uploaded (for instructor) |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

# 

# IT Standards Marking Sheet

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Programming & SQL Standards - 1% each** | | | |  | |
|  | P1 Meaningful Identifiers | |  | P20 Code Module Size and Focus | |
|  | P2 Prefixes & Hungarian Notation | |  | P21 Single Point of Exit | |
|  | P3 Identifier Case Conventions | |  | P22 Disabled Code & Misleading | |
|  | P4 Header Comments | |  | P23 Each Class in a File Named | |
|  | P5 Method Comments | |  | P24 Class Organization | |
|  | P9 "Magic" Numbers and Strings | |  | P25 Unwise Coding Practice | |
|  | P10 Constant Scope | |  | SQL1 Table Names | |
|  | P11 Indentation | |  | SQL2 Column Names | |
|  | P12 Line Length and Wrapping | |  | SQL3 Keywords & Function Names | |
|  | P13 Blank Lines | |  | SQL4 Header Comments | |
|  | P14 Code Crowding | |  | SQL5 Output Messages | |
|  | P15 Space Around Binary Operators | |  | SQL6 Implementation Comments | |
|  | P16 Space After Delimiters | |  | SQL7 Formatting | |
|  | P17 Curly Brace Alignment | |  | SQL8 Subquery IN and = | |
|  | P19 Global Variables | |  |  | |
|  | |  | | **Late Assignments** | |
| **Days Late** | **Penalty %** |
| **Base Mark:** | |  | | 1 | 5 |
| **Standards Penalties: - %** | | - | | 2 | 10 |
| **Late Penalties: - %** | | - | | 3 | 20 |
| **Final Mark:** | |  | | 4 | 40 |
|  | |  | | 5 | 60 |
| 6 | 80 |
| 7 | 100 |
|  |  |

1. Create a brand-new database (if you didn’t already do this in class – just show

the database in your screenshot along with the syntax you would use) named

“Entertainment”. Create a new schema within this new database named

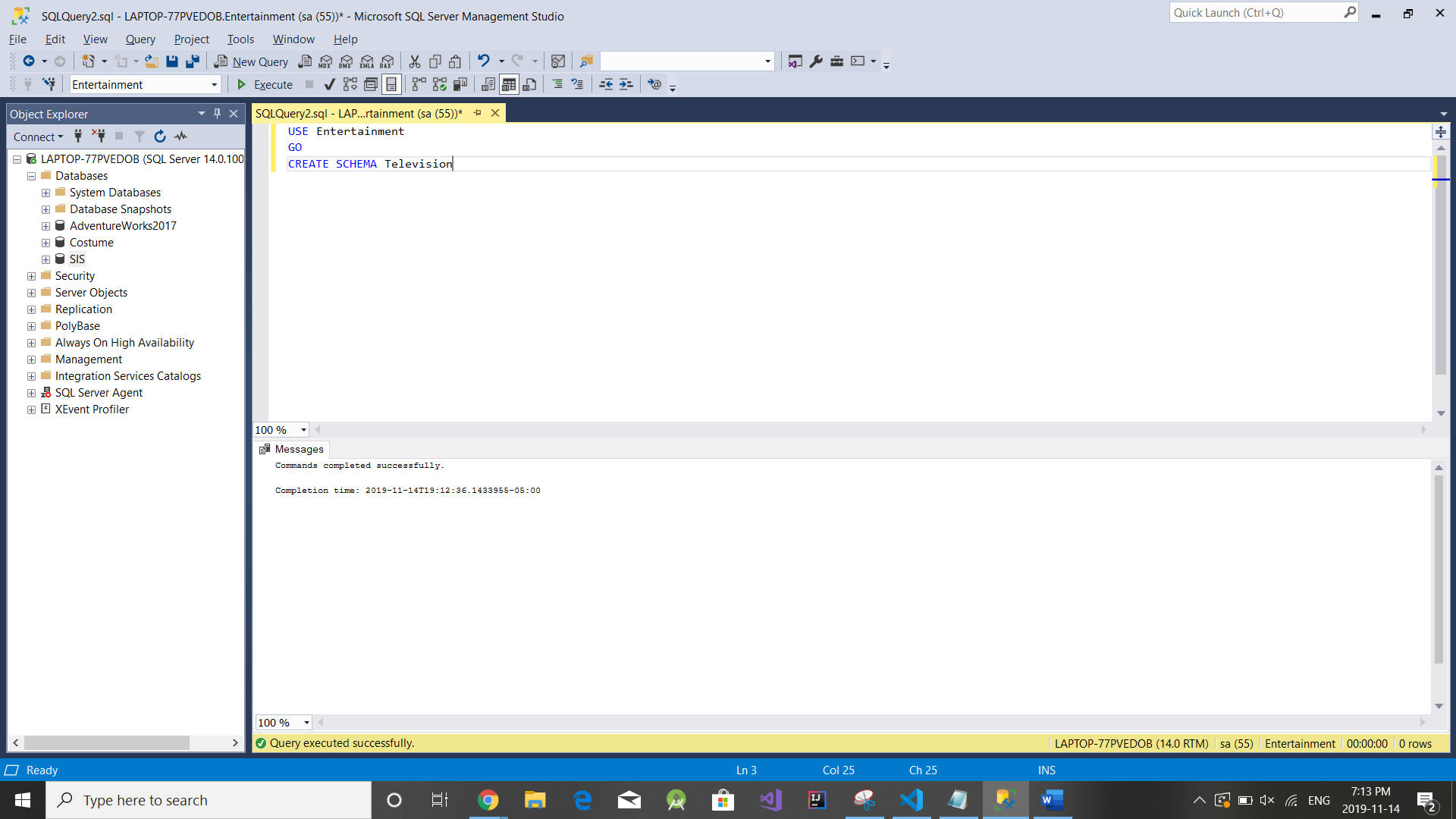
“Television”. Show your scripts to create both.

Create database Entertainment

USE Entertainment

GO

CREATE SCHEMA Television



2)

a) Use Entertainment

Go

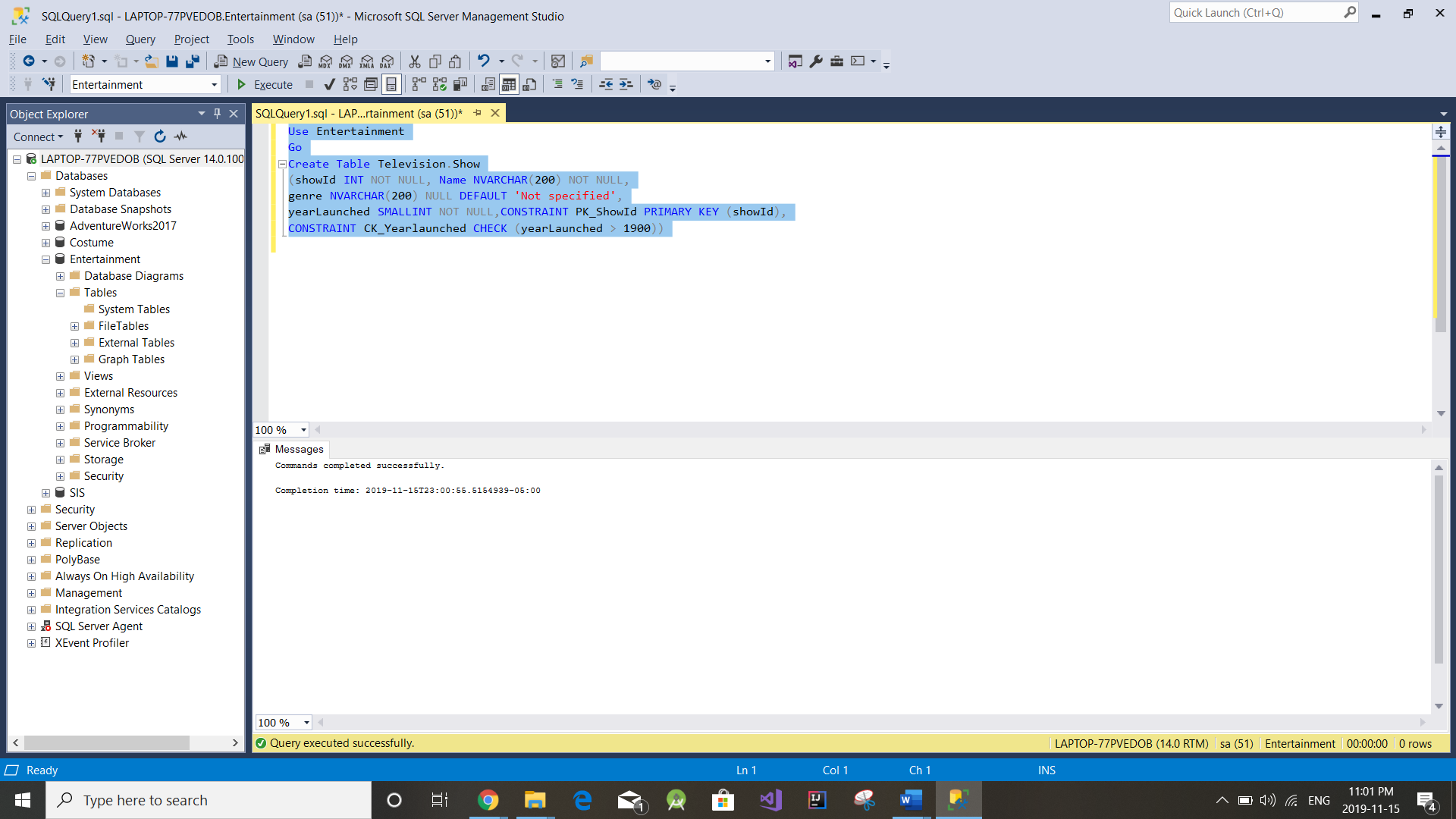
Create Table Television.Show

(showId INT NOT NULL, Name NVARCHAR(200) NOT NULL,

genre NVARCHAR(200) NULL DEFAULT 'Not specified',

yearLaunched SMALLINT NOT NULL,CONSTRAINT PK\_ShowId PRIMARY KEY (showId),

CONSTRAINT CK\_Yearlaunched CHECK (yearLaunched > 1900))



b) Use Entertainment

Go

Create Table Television.Episode

(episodeId int NOT NULL,showId int NOT NULL,

showSeason int NOT NULL,showEpisodeNumber int NULL,

numberViewers int NULL,costToProduce money NULL,

Constraint PK\_episodeId Primary key (episodeId),

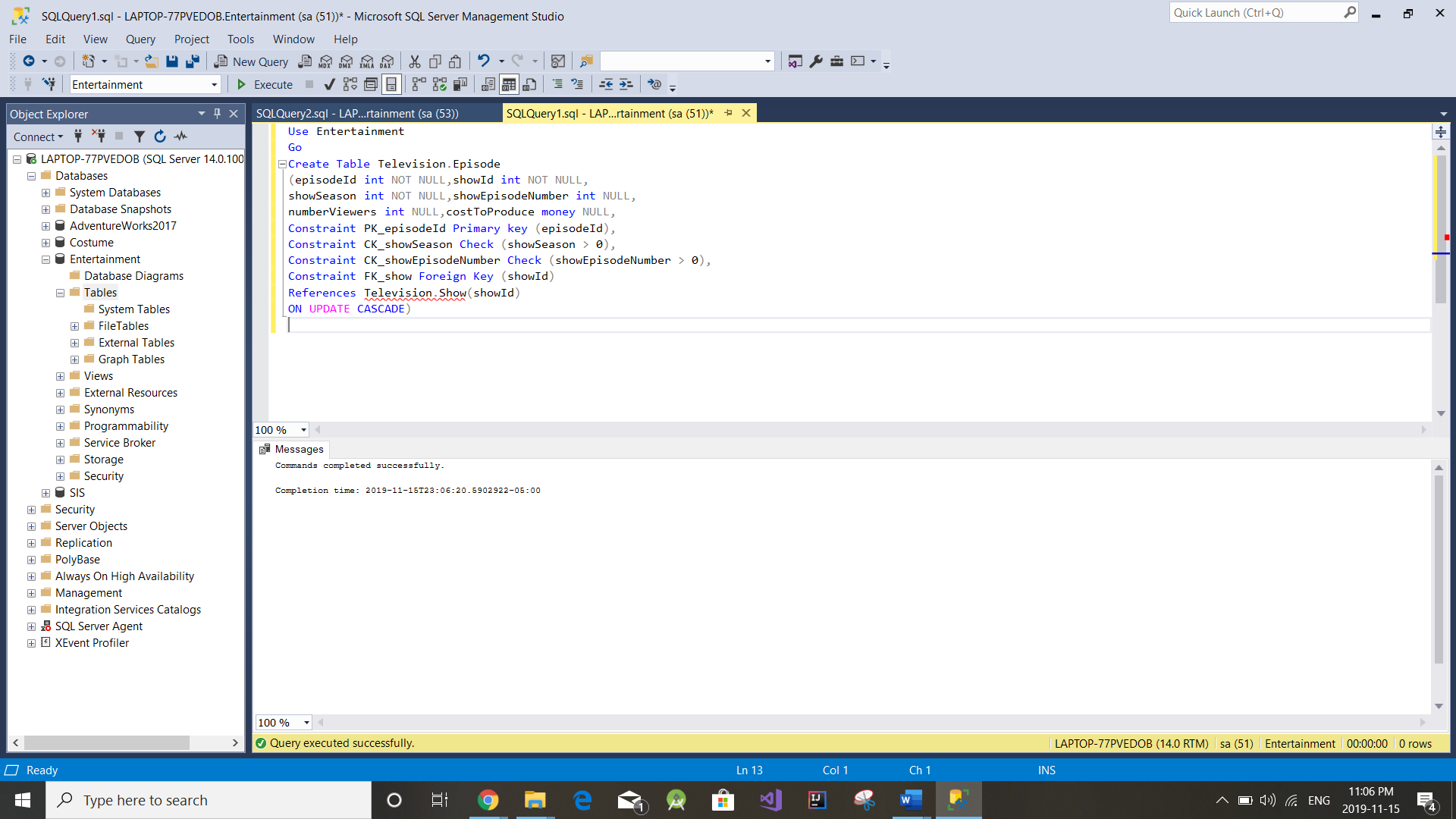
Constraint CK\_showSeason Check (showSeason > 0),

Constraint CK\_showEpisodeNumber Check (showEpisodeNumber > 0),

Constraint FK\_show Foreign Key (showId)

References Television.Show(showId)

ON UPDATE CASCADE)



c) Use Entertainment

Go

Create Table Television.Cast

(Id int NOT NULL,episodeId int NOT NULL,

FirstName nvarchar(256) NOT NULL,LastName nvarchar(256) NOT NULL,

recordingHours int NOT NULL,hourlySalary money NOT NULL,

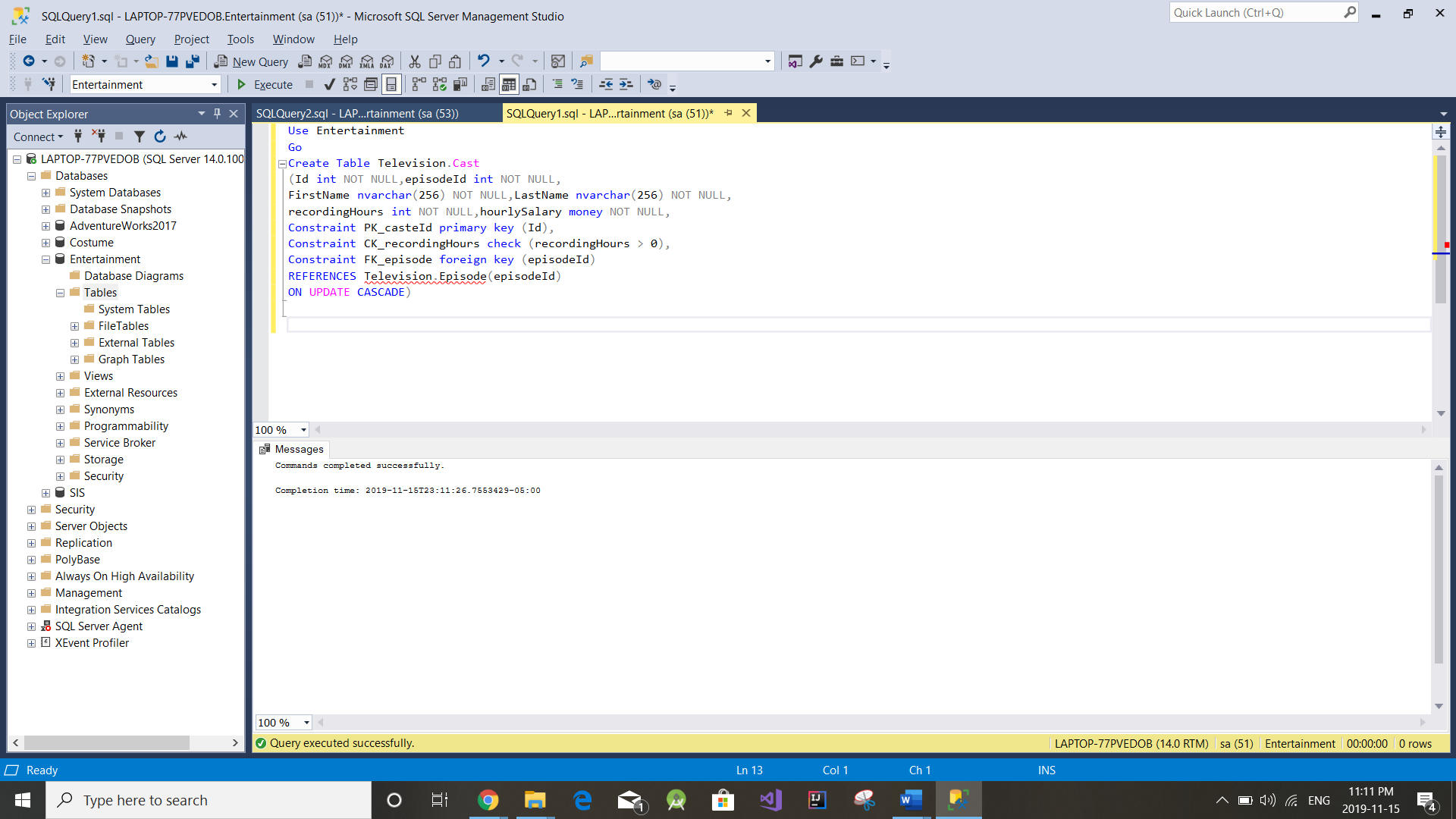
Constraint PK\_casteId primary key (Id),

Constraint CK\_recordingHours check (recordingHours > 0),

Constraint FK\_episode foreign key (episodeId)

REFERENCES Television.Episode(episodeId)

ON UPDATE CASCADE)



3)

3)

Table 1) Use Entertainment

go

INSERT INTO Television.Show(showId,Name, genre, yearLaunched)

VALUES (001, 'Friend','Comedy', 1993)

INSERT INTO Television.Show(showId,Name, genre, yearLaunched)

VALUES (002, 'Game Of Thrones','Drama', 2003)

INSERT INTO Television.Show(showId,Name, genre, yearLaunched)

VALUES (003, 'Picky Blinders','Drama', 2007)

INSERT INTO Television.Show(showId,Name, genre, yearLaunched)

VALUES (004, 'Sherlock','Action', 2010)

INSERT INTO Television.Show(showId,Name, genre, yearLaunched)

VALUES (005, 'picky blinders','Drama', 2005)

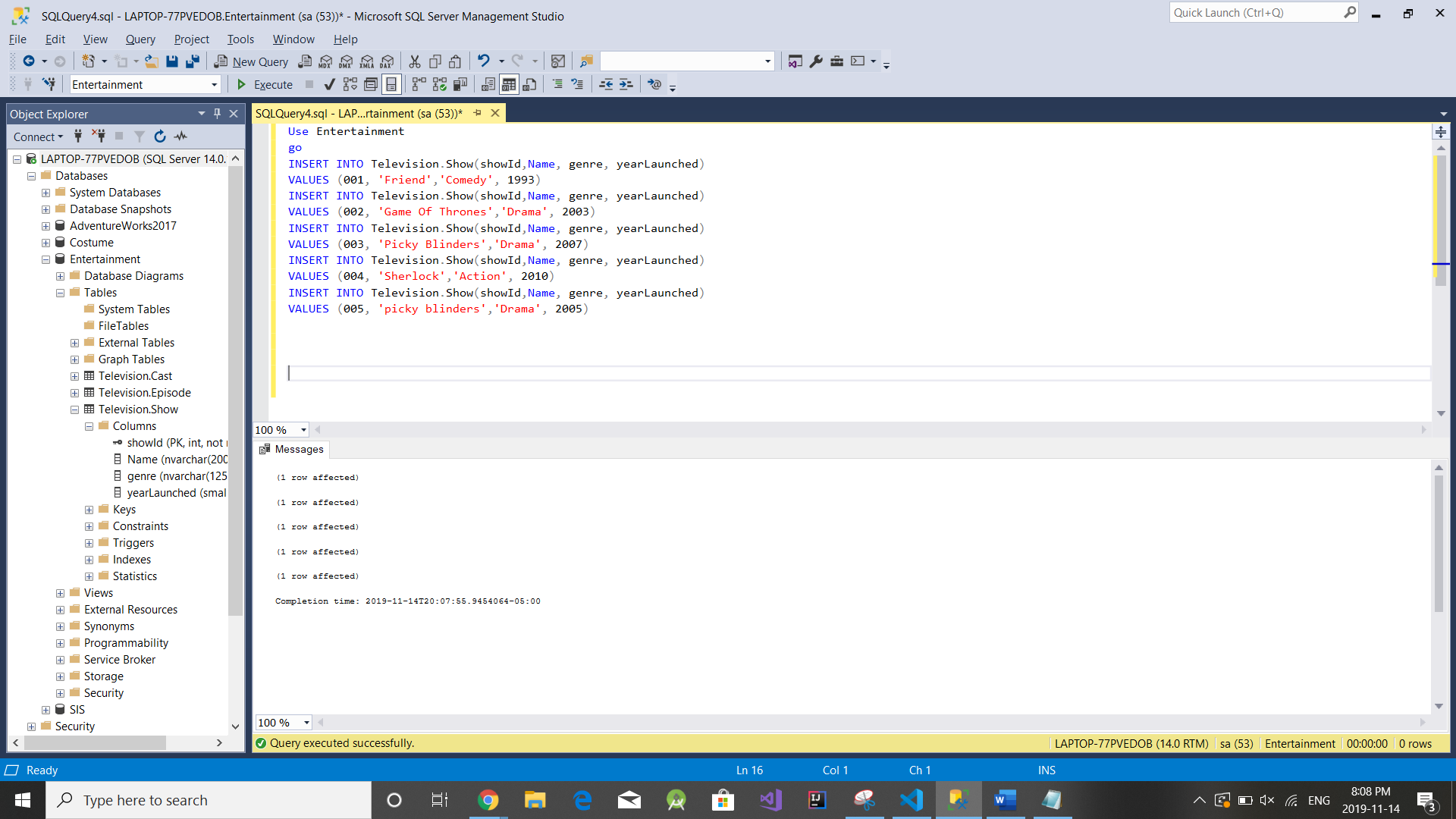


Table 2) use Entertainment

go

insert into Television.Episode (episodeId,showId,showSeason,showEpisodeNumber,numberViewers,costToProduce)

values(0001,001,1,45,5000,50000)

insert into Television.Episode (episodeId,showId,showSeason,showEpisodeNumber,numberViewers,costToProduce)

values(0002,002,3,40,600,60000)

insert into Television.Episode (episodeId,showId,showSeason,showEpisodeNumber,numberViewers,costToProduce)

values(0003,003,5,56,500,56666)

insert into Television.Episode (episodeId,showId,showSeason,showEpisodeNumber,numberViewers,costToProduce)

values(0004,004,3,45,855,8989)

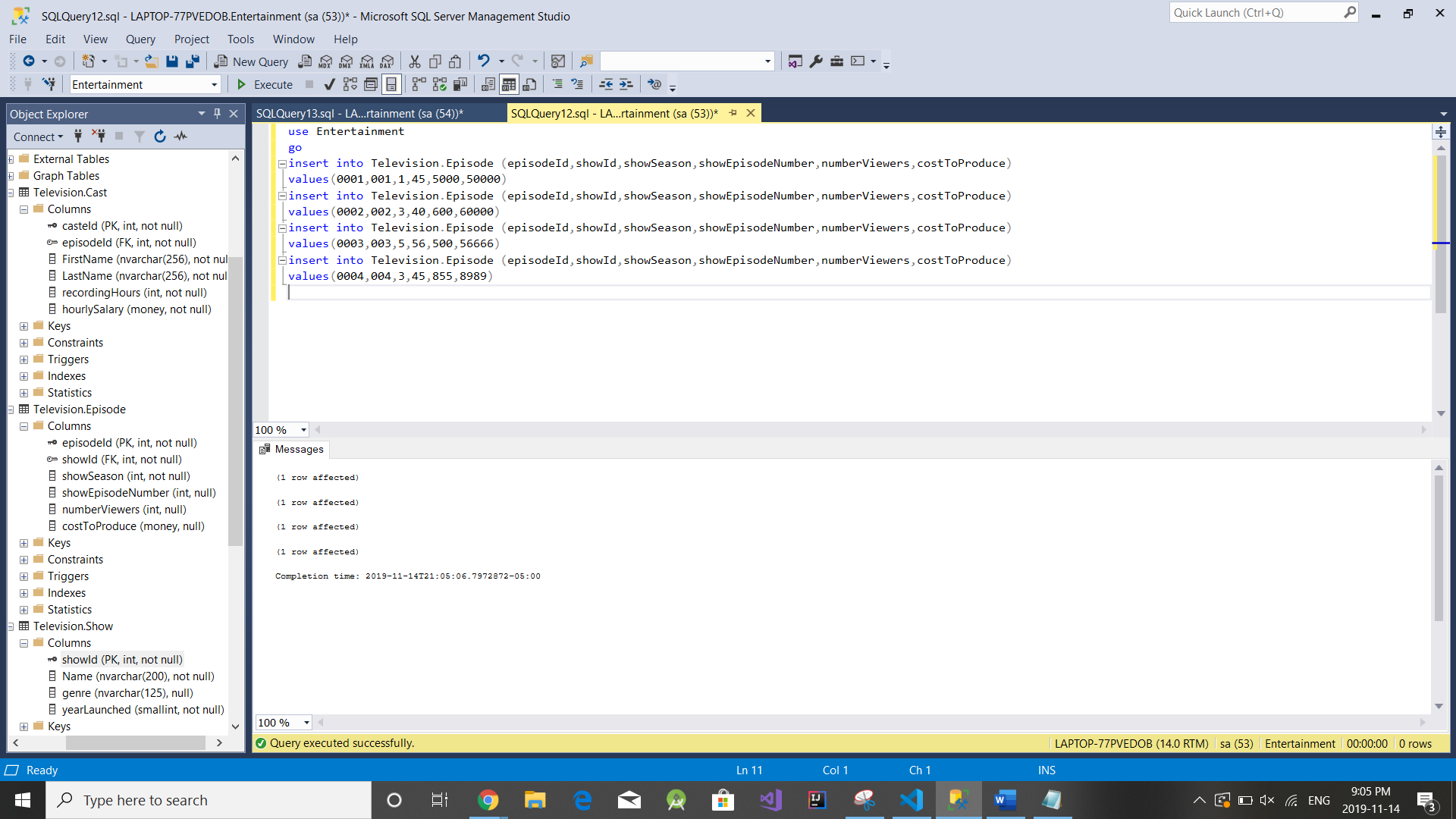


Table 3)

Use Entertainment

go

insert into [Entertainment].[Television].Cast(Id,episodeId,Firstname,LastName,recordingHours,hourlySalary)

values(01,0001,'Jennifer','Anniston', 10,50)

insert into [Entertainment].[Television].Cast(Id,episodeId,Firstname,LastName,recordingHours,hourlySalary)

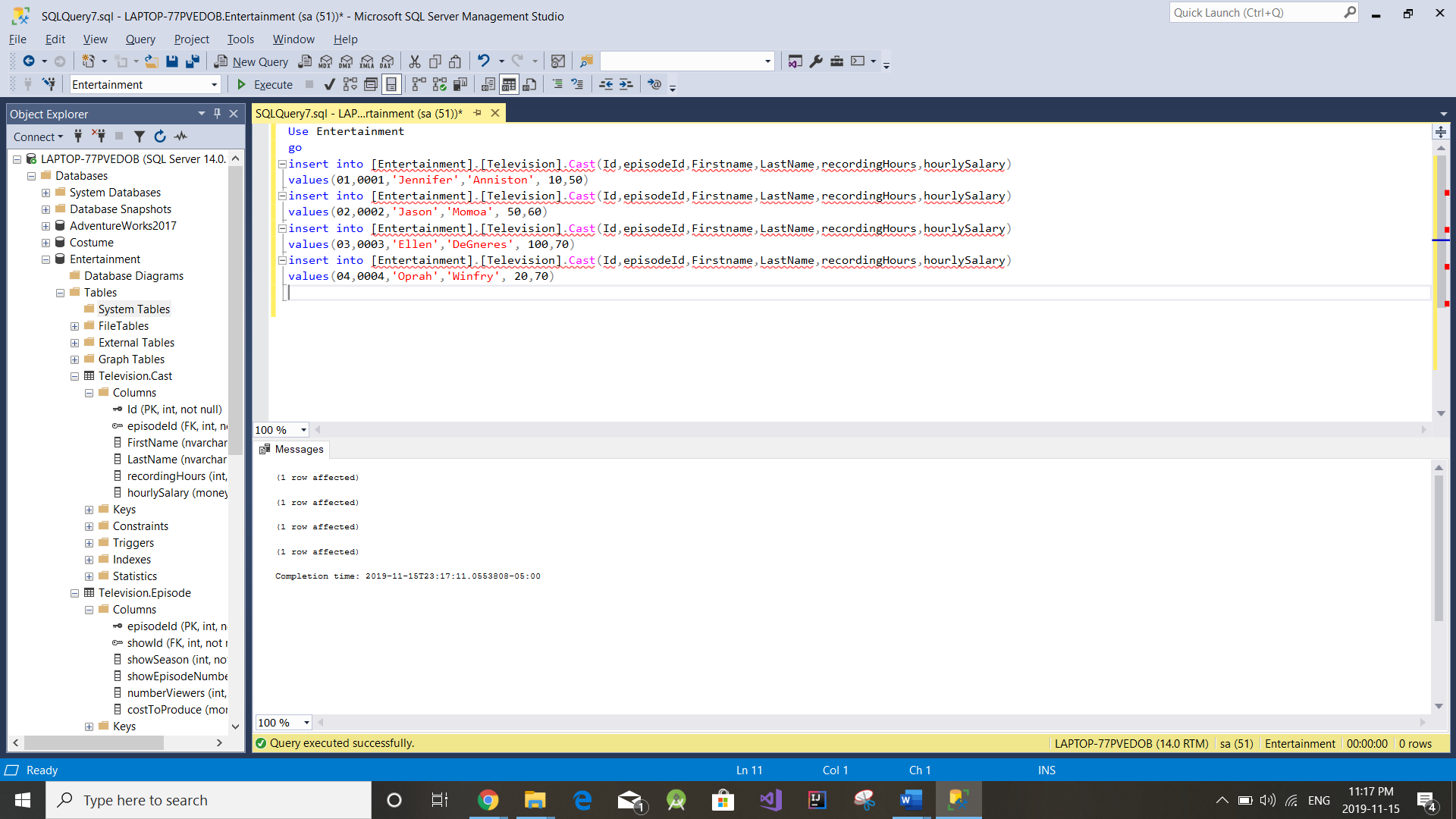
values(02,0002,'Jason','Momoa', 50,60)

insert into [Entertainment].[Television].Cast(Id,episodeId,Firstname,LastName,recordingHours,hourlySalary)

values(03,0003,'Ellen','DeGneres', 100,70)

insert into [Entertainment].[Television].Cast(Id,episodeId,Firstname,LastName,recordingHours,hourlySalary)

values(04,0004,'Oprah','Winfry', 20,70)



4) Use Entertainment

Go

Create function Televison.CalculateTotalSalary (@Id int , @episodeId int)

returns money as

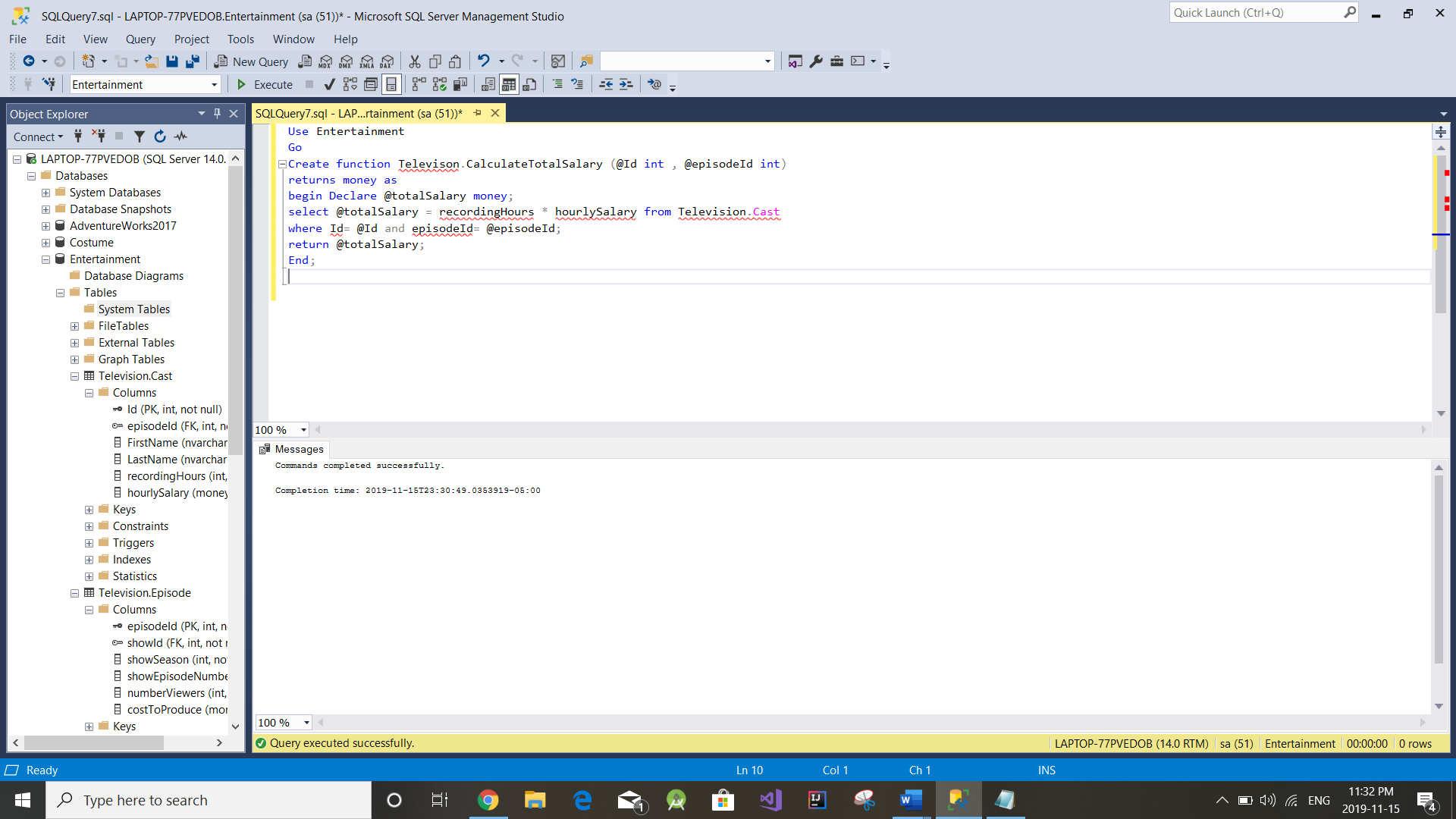
begin Declare @totalSalary money;

select @totalSalary = recordingHours \* hourlySalary from Television.Cast

where Id= @Id and episodeId= @episodeId;

return @totalSalary;

End;



5) use Entertainment

go

create procedure Television.castDetails

@Id as int = null

as

begin

if @Id is Null

begin

select cast.FirstName + ' ' +cast.LastName AS FullName ,

show.Name, episode.showEpisodeNumber,

Television.CalculateTotalSalary(cast.Id, episode.episodeId) As TotalSalary FROM Television.Cast cast

Join Television.Episode episode ON(cast.episodeId = episode.episodeId)

Join Television.Show show ON(episode.showId = show.showId)

end

else

begin

select cast.FirstName + ' ' +cast.LastName AS FullName ,

show.Name, episode.showEpisodeNumber,

Television.CalculateTotalSalary(cast.Id, episode.episodeId) As TotalSalary FROM Television.Cast cast

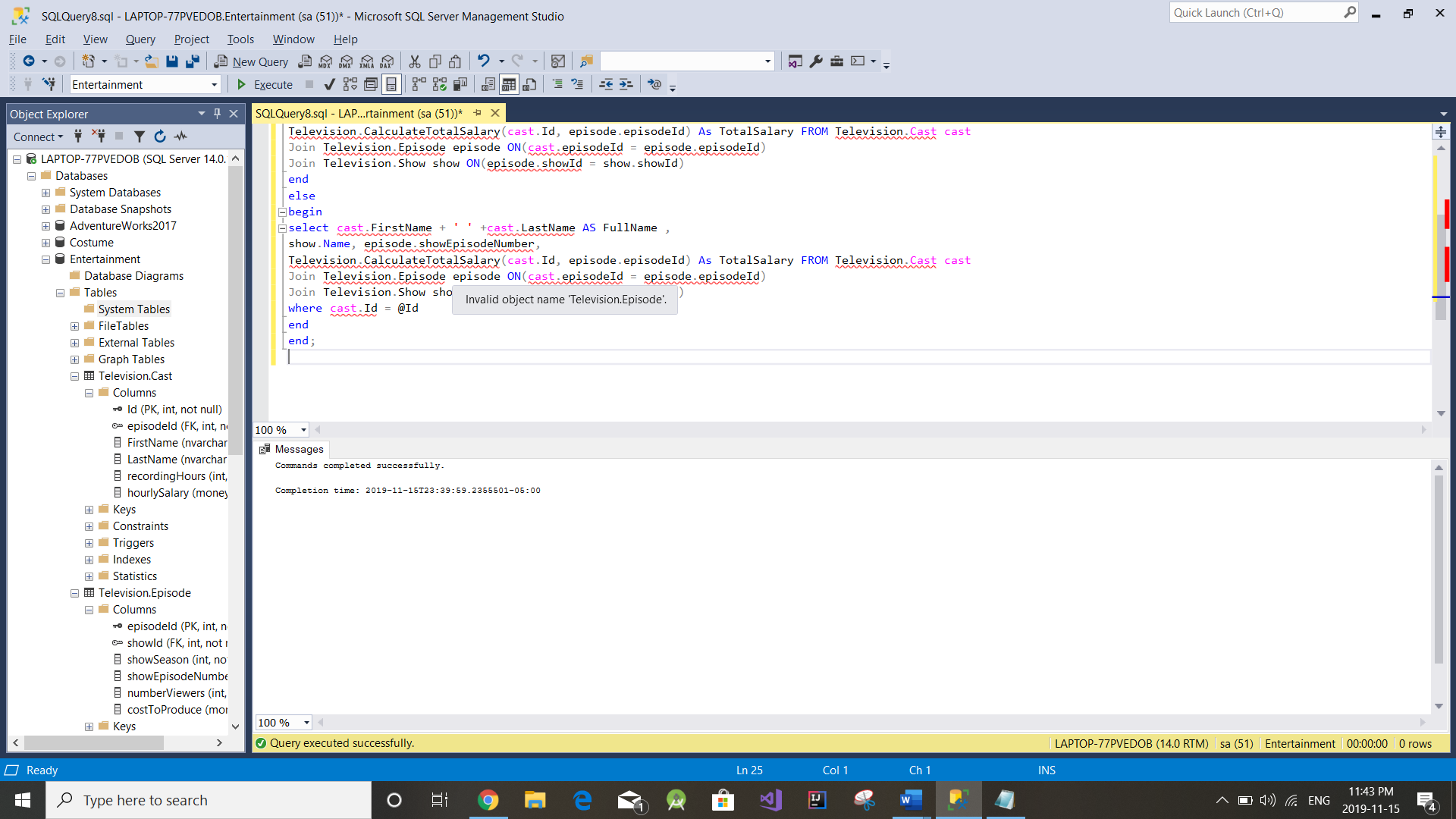
Join Television.Episode episode ON(cast.episodeId = episode.episodeId)

Join Television.Show show ON(episode.showId = show.showId)

where cast.Id = @Id

end

end;



6) use Entertainment

go

Declare

@returnvalue int

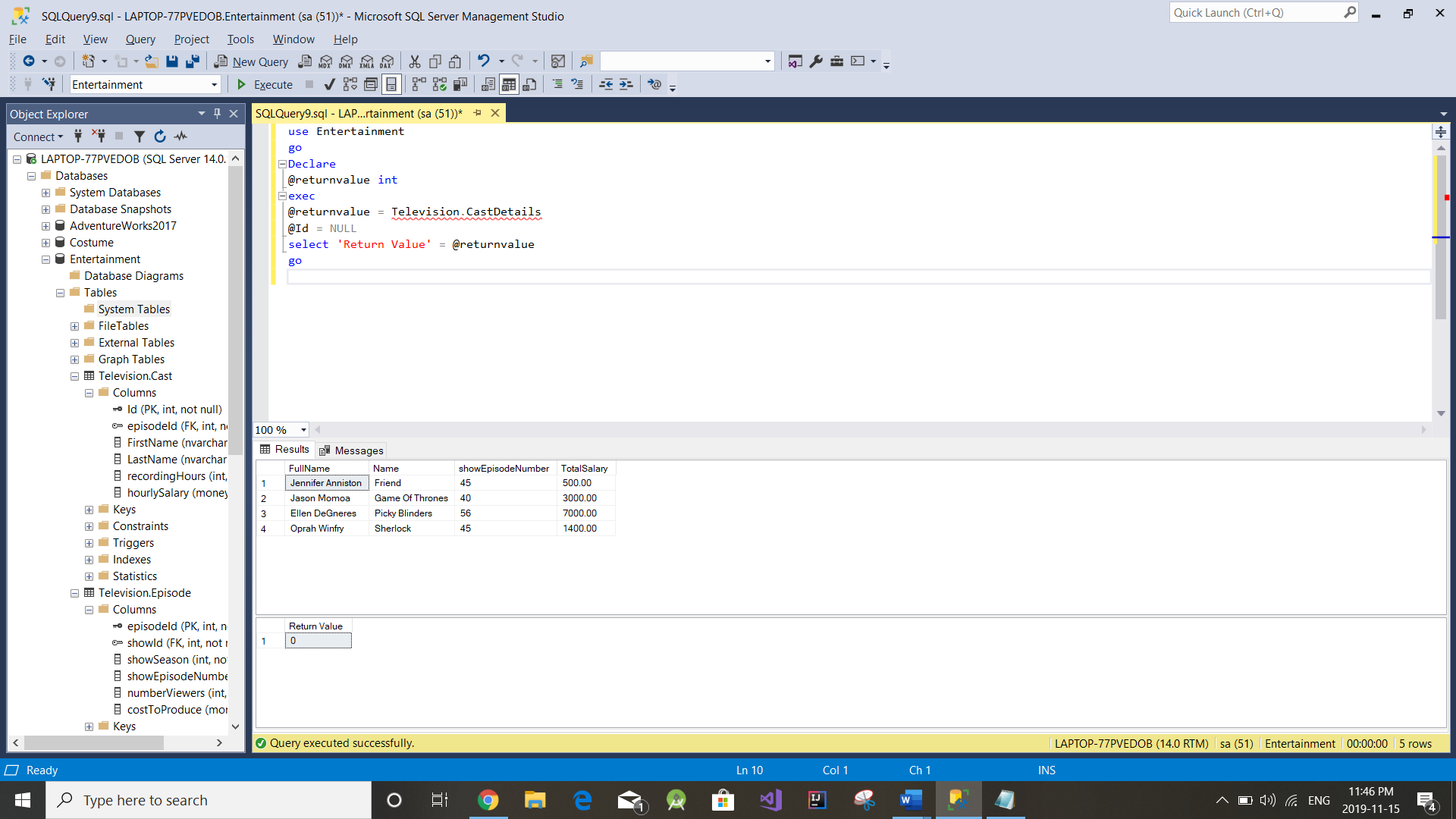
exec

@returnvalue = Television.CastDetails

@Id = NULL

select 'Return Value' = @returnvalue

go



use Entertainment

go

Declare

@returnvalue int

exec

@returnvalue = Television.CastDetails

@Id = 4

select 'Return Value' = @returnvalue

go

