Assignment 4:

Set a:

1. It is a virtual table that takes data from one or more columns used to aggregate information. Views are not stored in the databases. Hence, it does not use extra storage and can be created from tables on same or different databases. The main usage of views is to implement security as it restricts users to specify records and expediate data analysis.
2. Yes
3. It’s lines of code that’s compiled and stored in the database in a single execution plan. It can be saved and reused which increases the speed of execution and code reusability and decrease network traffic
4. A view is the template of a table and does not take parameters but stored procedure consists of compiled statements that takes input parameters and returns and output. Views can be used in stored procedures while stored procedures cant be used in views
5. Sp allows select, insert, update and delete while function only allows select  
   sp can’t be used in select, where or having statements while a function can  
   transactions can be used in sp but not functions  
   sp can have inputs and outputs and can return no values, one or multiple whole while functions only take input parameters and return one value specifically.  
   sp can call function but function can’t call sp  
   in sp, try-catch exception handling can be used but not can’t be used in functions
6. Yes
7. No. stored procedures are not allowed in select, where or having statements
8. It’s a type of stored procedure that gets fired in response to an event and have many types (DDL, DML, Logon and CLR triggers)
9. To enforce business rules and data integrity. As well as maintaining record of all changes
10. Triggers run automatically while sp must be called. Hence, event and action must be identifies for triggers but not sp.

A trigger can call an sp but not the opposite.

Set b:

Q1: a,b,c

begin transaction

select \* from region with(holdlock)

select \* from Territories with(holdlock)

select \* from Employees with(holdlock)

select \* from EmployeeTerritories with(holdlock)

insert into region values (5, 'Middle Earth')

insert into Territories values (99999, 'Gondor', 5)

insert into Employees(LastName, FirstName, region) values

('King', 'Aragorn', 5)

insert into EmployeeTerritories values (10, 99999)

rollback

Q2:

begin transaction

update Territories set TerritoryDescription = 'Arnor'

where TerritoryDescription = 'Gondor'

Q3:

begin transaction

delete from territories where territoryid = 99999 and

TerritoryDescription = 'Arnor' and regionid = 5

delete from region where RegionDescription = 'Middle Earth'

rollback

begin transaction

select \* from region with(holdlock)

select \* from Territories with(holdlock)

select \* from empolyees with(holdlock)

commit

Q4:

create view view\_product\_order\_barq as

select p.productname, sum(od.quantity) 'Total Ordered Quantity'

from products p

inner join [Order Details] od on p.ProductID = od.ProductID

group by p.ProductName

Q5:

create proc sp\_product\_order\_quantity\_barq(@product\_id int,

@total\_quantity int output) as

begin

select @product\_id = p.productid from products p

inner join [Order Details] od

on p.ProductID = od.ProductID

where sum(od.quantity) = @total\_quantity

group by p.ProductID

end

Q6:

create proc sp\_product\_order\_city\_barq(@product\_name varchar(20),

@order\_city varchar(20) output) as

begin

select @product\_name=t1.productname from (select top 5 t2.ProductID,

t2.ProductName from (select p.ProductID,p.ProductName,sum(od.quantity) t3 from Products p

inner join [Order Details] od on p.ProductID = od.ProductID

group by p.ProductID, p.ProductName) as t2 Order by t2.t3 desc) t1

left join(select \* from (select t4.productid,

t4.city, rank() over(partition by productid

order by quan desc) [rnk] from(select p.ProductID,

c.city, sum(od.quantity) quan from Customers c

join orders o on c.CustomerID= o.CustomerID

left join [Order Details] od on o.OrderID=od.OrderID

left join Products p on od.ProductID=p.ProductID

group by p.ProductID, c.City ) t4 ) c2

where c2.rnk=1) t5

on t1.productid= t5.productid

where t5.city =@order\_city

end

Q7:

create proc sp\_move\_employees\_barq

@terroity\_name char(20) = 'Tory' as

if exists(select e.EmployeeID, count(terr.TerritoryDescription) c

from Territories terr

inner join employeeterritories empt

on terr.TerritoryID=empt.TerritoryID

inner join Employees e

on empt.EmployeeID=e.EmployeeID

where TerritoryDescription = @terroity\_name and

count(terr.TerritoryDescription) > 0)

begin

insert into Territories values

(11111,'Stevens Point',6)

insert into Region values(6,'North')

end

go

Q8:

create trigger trg on territories

for update as

if exists(select e.employeeid, count(terr.TerritoryDescription)

from Territories terr

inner join employeeterritories empt on terr.TerritoryID=empt.TerritoryID

inner join Employees e on empt.EmployeeID=e.EmployeeID

where terr.TerritoryDescription= 'Stevens Point'

group by e.EmployeeID having count(terr.TerritoryDescription)>100)

begin

update Territories set TerritoryDescription= 'Tory' where

TerritoryDescription='Stevens Point'

end

drop trigger trg

Q9:

create table people\_barq (personId int, name char(20), cityid int)

create table city\_barq (cityId int, city char(20))

insert into city\_barq(cityid,city) values(1,'Seattle')

insert into city\_barq(cityid,city) values(2,'Green Bay')

insert into people\_barq(personid,name,cityid) values(1,'Aaron Rodgers',2)

insert into people\_barq(personid,name,cityid) values(2,'Russell Wilson',1)

insert into people\_barq(personid,name,cityid) values(3,'Jody Nelson',2)

update city\_barq set city = 'Madison' where city = 'Seattle'

create view Packers\_shatha as

select p.personid, p.name from people\_barq p

inner join city\_barq c on p.cityid=c.cityid

where c.city='Green bay'

begin transaction

rollback

drop table people\_barq

drop table city\_barq

drop view Packers\_shatha

Q10:

create proc sp\_birthday\_employees\_barq as

begin

select employeeid, LastName,FirstName, birthdate,HireDate

into birthday\_employees\_barq from Employees

where month(Birthdate)=2

end

drop table birthday\_employees\_barq

Q11:

create proc sp\_barq\_1 as

select c.city, count(c.CustomerID) from Customers c

inner join (select dt2.CustomerID, count(dt2.CustomerID)dt1

from (select distinct c.CustomerID,

p.ProductID from Products p

inner join [Order Details] od on p.ProductID=od.ProductID

inner join Orders o on od.OrderID=o.OrderID

inner join Customers c on o.CustomerID=c.CustomerID)dt2

group by dt2.CustomerID

having count(dt2.CustomerID)<2) co2 on c.CustomerID= co2.CustomerID

group by city

having count(c.CustomerID) > 1

Q12:

By using union, to confirm both tables have identical data, Row count returned in below query should be same as number of rows in emp1 or emp2(**row count of below query= row count of emp1= row count of emp2**).

select \* From emp1

union

select \* From emp2

source: <http://www.besttechtools.com/articles/article/sql-query-to-check-two-tables-have-identical-data>

Q13:

create table person([First Name] varchar(20),

[Last Name] varchar(20), [Middle name] varchar(1))

insert into person([first name], [last name]) values('John', 'Green')

insert into person values ('Mike', 'White', 'M')

select [first name] + ' ' + [last name] +

' ' + coalesce([middle name]+'.', '')

as FullName from person

Q14:

create table scores(Student varchar(20),

Marks int, Sex varchar(1))

insert into scores values ('Ci', 70, 'F')

insert into scores values ('Bob', 80, 'M')

insert into scores values ('Li', 90, 'F')

insert into scores values ('Mi', 95, 'M')

select \* from scores

select top 1 with ties \*

from scores order by sex, marks desc

Q15:

select \*

from scores order by sex, marks desc