1, view is the result set of a stored query on the data, which the database users can query just as they would in a persistent database collection object. a view can limit the degree of exposure of the underlying tables to the outer world: a given user may have permission to query the view, while denied access to the rest of the base table.

2. View cannot be modify like group by and distinct.

3. A stored procedure is a prepared SQL code that you can save, so the code can be reused over and over again. So if you have an SQL query that you write over and over again, save it as a stored procedure, and then just call it to execute it

4. View is simple showcasing data stored in the database tables whereas a stored procedure is a group of statements that can be executed. A view is faster as it displays data from the tables referenced whereas a store procedure executes sql statements

5. A function has a return type and returns a value. A procedure does not have a return type. But it returns values using the OUT parameters.

6. Stored procedures contain IN and OUT parameters or both. They may return result sets in case you use SELECT statements. Stored procedures can return multiple result sets

7. Stored procedures are typically executed with an EXEC statement. However, you can execute a stored procedure implicitly from within a SELECT statement, provided that the stored procedure returns a result set

8. Trigger is a Special kind of Stored Procedure or an operation that gets executed automatically when an event occurs in the database. DML Triggers, DDL Triggers, LOGON Triggers

9. a trigger can be invoked when a row is inserted into a specified table or when certain table columns are being updated

10. Stored procedures are a pieces of the code in written in PL/SQL to do some specific task. On the other hand, trigger is a stored procedure that runs automatically when various events happen (eg update, insert, delete).

Querys

1.

2. update Territories set TerritoryDescription = 'Arnor' where TerritoryDescription = ‘Gondor’

3. delete from Territories where TerritoryID = 11111 and TerritoryDescription = 'Arnor' and RegionID = 5 delete from Region where RegionDescription = 'Middle Earth' and RegionID = 5

4. create view view\_product\_order\_Yang as select p.ProductName, Count(o.Quantity) QuantityCount from Products p inner join [Order Details] o on o.ProductID = p.ProductID group by p.ProductName;

5. alter proc spProductOrderQuant2 @id int, @total int out as begin select @id = view\_product\_quantity\_order\_Yang.ProductID,@total = view\_product\_quantity\_order\_Yang.QuantityCount from view\_product\_quantity\_order\_Yang where view\_product\_quantity\_order\_Yang.ProductID = @id end declare @id int, @total int exec spProductOrderQuant2 2, @total out print @total

6.

ALTER PROC sp\_Product\_Order\_City\_Gaddam

@ProductName NVARCHAR(50)

AS

BEGIN

SELECT TOP 5 ShipCity,SUM(Quantity) FROM [Order Details] OD JOIN Products P ON P.ProductID = OD.ProductID JOIN Orders O ON O.OrderID = OD.OrderID

WHERE ProductName=@ProductName

GROUP BY ProductName,ShipCity

ORDER BY SUM(Quantity) DESC

END

EXEC sp\_Product\_Order\_City\_Gaddam 'Queso Cabrales'

7.

8. create trigger trg\_ins\_yang on territories

for update as

if exists(select e.employeeid,count(t.TerritoryDescription)from Territories t

join employeeterritories et on t.TerritoryID=et.TerritoryID

join Employees e on et.EmployeeID=e.EmployeeID

where t.TerritoryDescription='stevens point'

group by e.EmployeeID

having count(t.TerritoryDescription)>100)

begin

update Territories

set TerritoryDescription='Tory' where TerritoryDescription='stevens point'

End

drop trigger trg\_ins\_yang

9. create table people\_yang(id int,name char(20),cityid int)

create table city\_yang(cityid int,city char(20))

insert into people\_yang(id,name,cityid)values(1,'Aaron Rodgers',2)

insert into people\_yang(id,name,cityid)values(2,'Russell Wilson',1)

insert into people\_yang(id,name,cityid)values(3,'Jody Nelson',2)

insert into city\_yang(cityid,city)values(1,'Settle')

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

create view Packers\_yi\_yang as

select p.id, p.name from people\_yang p inner join city\_yang c on p.cityid=c.cityid

where c.city='Green bay'

begin tran

rollback

drop table people\_yang

drop table city\_yang

drop view Packers\_yi\_yang

10.

ALTER PROC sp\_birthday\_employee\_gaddam

AS

BEGIN

SELECT \* INTO #EmployeeTemp

FROM Employees WHERE DATEPART(MM,BirthDate) = 02

SELECT \* FROM #EmployeeTemp

END

11. CREATE PROC sp\_gaddam\_1

AS

BEGIN

SELECT City FROM CUSTOMERS

GROUP BY City

HAVING COUNT(\*)>2

INTERSECT

SELECT City FROM Customers C JOIN Orders O ON O.CustomerID=C.CustomerID JOIN [Order Details] OD ON O.OrderID = OD.OrderID

GROUP BY OD.ProductID,C.CustomerID,City

HAVING COUNT(\*) BETWEEN 0 AND 1

END

GO

EXEC sp\_gaddam\_1

GO

CREATE PROC sp\_gaddam\_2

AS

BEGIN

SELECT City FROM CUSTOMERS

WHERE CITY IN (SELECT City FROM Customers C JOIN Orders O ON O.CustomerID=C.CustomerID JOIN [Order Details] OD ON O.OrderID = OD.OrderID

GROUP BY OD.ProductID,C.CustomerID,City

HAVING COUNT(\*) BETWEEN 0 AND 1)

GROUP BY City

HAVING COUNT(\*)>2

END

GO

EXEC sp\_gaddam\_2

GO

12. USE EXCEPT KEYWORD

SELECT \* FROM Customers

EXCEPT

SELECT \* FROM Customers

14. SELECT firstName+' '+lastName from Person where middleName is null UNION SELECT firstName+' '+lastName+' '+middelName+'.' from Person where middleName is not null

15 select top 1 marks from student where sex = 'F' order by marks desc

16. select \* from students order by sex,marks