Assignment Day4 –SQL: Comprehensive practice

# Answer following questions

1. What is View? What are the benefits of using views?

View is virtual table that contains from one or more tables. We can control security of accessing of the data. Also, it doesn’t hold any data and exist physically in the database, so we can save storage.

1. Can data be modified through views?

I can’t directly modify data in views

1. What is stored procedure and what are the benefits of using it?

Stored procedure is collection of sql statmenet and sql command logic that is compiled and stored on database. The main purpose is to minimize data traffic and improve performance of data operations such as select, update, delete data

1. What is the difference between view and stored procedure?

View is simple showing data stored in the database tables whereas a Stored procedure is a group of statements that can be executed.

1. What is the difference between stored procedure and functions?

Functions must return a value whereas it is optional in Stored procedure. Also, Functions can have only input parameters for it whereas Procedures can have input or output parameters.

Procedure can be called Functions, but Functions cannot be called Procedures.

1. Can stored procedure return multiple result sets?

Yes, most stored procedures return multiple result sets like including one or more select statements.

1. Can stored procedure be executed as part of SELECT Statement? Why?

Stored procedures are typically executed with an EXEC statement, but we can execute a stored procedures implicitly from within a select statement, provided that the stored procedure returns a result set.

1. What is Trigger? What types of Triggers are there?

Trigger is a set of actions that are performed in response to an insert, update, or delete operation on specified table. When sql operation is executed, the trigger is activated. It is optional and are defined using CREATE trigger statement.

1. What are the scenarios to use Triggers?

Such as log table modifications. Some tables have sensitive data such as customer email, employee salary, etc., that you want to log all the changes. In this case, you can create the UPDATE trigger to insert the changes into a separate table.

1. What is the difference between Trigger and Stored Procedure?

A stored procedure is a user defined piece of code written in the local version of SQL, which may return a value as of function that is invoked by calling it explicitly. A trigger is a stored procedure that runs automatically when various events happen (eg update, insert, delete)

# Write queries for following scenarios

Use Northwind database. All questions are based on assumptions described by the Database Diagram sent to you yesterday. When inserting, make up info if necessary. Write query for each step. Do not use IDE. BE CAREFUL WHEN DELETING DATA OR DROPPING TABLE.

1. Lock tables Region, Territories, EmployeeTerritories and Employees. Insert following information into the database. In case of an error, no changes should be made to DB.
   1. A new region called “Middle Earth”;
   2. A new territory called “Gondor”, belongs to region “Middle Earth”;
   3. A new employee “Aragorn King” who's territory is “Gondor”.
2. Change territory “Gondor” to “Arnor”.
3. Delete Region “Middle Earth”. (tip: remove referenced data first) (Caution: do not forget WHERE or you will delete everything.) In case of an error, no changes should be made to DB. Unlock the tables mentioned in question 1.
4. Create a view named “view\_product\_order\_[your\_last\_name]”, list all products and total ordered quantity for that product.

create view view\_product\_order\_kim

as select p.productid, p.productname, sum(od.Quantity) quantitycount

from products p inner join [order details] od on od.productid = p.productid

group by p.productid, p.productname

1. Create a stored procedure “sp\_product\_order\_quantity\_[your\_last\_name]” that accept product id as an input and total quantities of order as output parameter.

create proc sp\_product\_order\_quantity\_kim

(@product\_id int,

@total\_quantity float output)as

begin

select @product\_id = p.productid

from products p

join [order details] od

on p.productid = od.productid

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

group by p.productid

end

1. Create a stored procedure “sp\_product\_order\_city\_[your\_last\_name]” that accept product name as an input and top 5 cities that ordered most that product combined with the total quantity of that product ordered from that city as output.

create proc sp\_product\_order\_city\_kim

(@product\_name varchar(50),

@order\_city varchar(50) output)

as

begin

select @product\_name =dt.productname from (select top 5 d.productid, d.productname

from (select p.productid, p.productname, sum(od.quantity) t from products p

inner join [order details] od

on p.productid = od.productid

group by p.ProductID, p.productname) as d order by d.t desc) dt

left join(

select \* from (select dd.productid, dd.city, rank() over(parition by productid order by q desc) [rk]

from (select p.productid, c.city, sum(od.quantity) q 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) dd)cc where cc.rk =1) x

on dt.productid = x.productid

where x.city = @order\_city

where x.city = @order\_city

end

1. Lock tables Region, Territories, EmployeeTerritories and Employees. Create a stored procedure “sp\_move\_employees\_[your\_last\_name]” that automatically find all employees in territory “Tory”; if more than 0 found, insert a new territory “Stevens Point” of region “North” to the database, and then move those employees to “Stevens Point”.
2. Create a trigger that when there are more than 100 employees in territory “Stevens Point”, move them back to Troy. (After test your code,) remove the trigger. Move those employees back to “Troy”, if any. Unlock the tables.
3. Create 2 new tables “people\_your\_last\_name” “city\_your\_last\_name”. City table has two records: {Id:1, City: Seattle}, {Id:2, City: Green Bay}. People has three records: {id:1, Name: Aaron Rodgers, City: 2}, {id:2, Name: Russell Wilson, City:1}, {Id: 3, Name: Jody Nelson, City:2}. Remove city of Seattle. If there was anyone from Seattle, put them into a new city “Madison”. Create a view “Packers\_your\_name” lists all people from Green Bay. If any error occurred, no changes should be made to DB. (after test) Drop both tables and view.

create table people\_kim (id int, name char(20), cityid int)

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

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

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

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

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

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

create view Packers\_kim as

select \*

from people\_kim k join city\_kim c on k.cityid=c.cityid

where c.city = 'Green bay'

begin tran

rollback

drop table people\_kim

drop table city\_kim

drop view Packers\_kim

1. Create a stored procedure “sp\_birthday\_employees\_[you\_last\_name]” that creates a new table “birthday\_employees\_your\_last\_name” and fill it with all employees that have a birthday on Feb. (Make a screen shot) drop the table. Employee table should not be affected.

create proc sp\_birthday\_employees\_kim as

begin

select EmployeeID, lastname, firstname, title, titleofcourtesy, birthdate, hiredate, photo

into birthday\_employees\_kim

from Employees

end

drop table birthday\_employees\_kim

1. Create a stored procedure named “sp\_your\_last\_name\_1” that returns all cites that have at least 2 customers who have bought no or only one kind of product. Create a stored procedure named “sp\_your\_last\_name\_2” that returns the same but using a different approach. (sub-query and no-sub-query).

create proc sp\_kim\_1 as

select c.city, count(c.customerid)

from customers c inner join (select x.customerid, count(x.customerid) xx

from(select distinct c.customerid, p.productid from products p

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

join orders o on od.orderid=o.orderid

join customers c on o.customerid = c.customerid) x

group by x.customerid

having count(x.customerid)<2) s

on c.customerid = s.customerid

group by city

having count(c.customerid) >1

1. How do you make sure two tables have the same data?

I can store the total data with union and check, if the number of data for two tables is not same data, then it’s not same data.

14.

|  |  |  |
| --- | --- | --- |
| First Name | Last Name | Middle Name |
| John | Green |  |
| Mike | White | M |

Output should be

|  |
| --- |
| Full Name |
| John Green |
| Mike White M. |

Note: There is a dot after M when you output.

declare @fullname varchar(20)

select @fullname = firstname + lastname + middlename +'.'

from people

print @fullname

15.

|  |  |  |
| --- | --- | --- |
| Student | Marks | Sex |
| Ci | 70 | F |
| Bob | 80 | M |
| Li | 90 | F |
| Mi | 95 | M |

Find the top marks of Female students.

If there are to students have the max score, only output one.

declare @student varchar(20)

declare @marks int

set @student

set @marks = (select max(marks)from student where

sex = 'F')

print @student

16.

|  |  |  |
| --- | --- | --- |
| Student | Marks | Sex |
| Li | 90 | F |
| Ci | 70 | F |
| Mi | 95 | M |
| Bob | 80 | M |

How do you out put this?

declare @student varchar(20)

declare @marks int

set @student

set @marks = (select max(marks)from student where

sex = 'F')

print @student

GOOD LUCK.