Assignment Day4 –SQL: Comprehensive practice

**Answer following questions**

* What is View? What are the benefits of using views?

Views in SQL are considered as a virtual table. A view also contains rows and columns.

A view can draw data from several different tables and present it as a single table, turning multi-table queries into single-table queries against the view.Views can give a user a "personalized" view of the database structure, presenting the database as a set of virtual tables that make sense for that user.

* Can data be modified through views?

YES,as long as the View is created based on one single table then direct "Update View" statement would work.

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

A stored procedure is a prepared SQL code that you can save, so the code can be reused over and over again.

Since procedures are stored on the database server which is faster than client. You can execute all the complicated quires using it, which will be faster.

* What is the difference between view and stored procedure?

View can not perform modification to any table, but stored procedure can perform modification to one or several tables

View does not accepts parameters, stored procedure accept parameters

* What is the difference between stored procedure and functions?

User defined functions will return values at any cost, Stored procedures may or may not necessarily return values.

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.

You can call a function from a stored procedure.

* Can stored procedure return multiple result sets?

Stored procedures can return multiple result sets.

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

You cannot call a procedure using select statements.Stored procedure may or may not return values, so it can't use select.

* What is Trigger? What types of Triggers are there?

Trigger can automatically execute when DML command statement occurs.Triggers are used to assess/evaluate data before or after data modification using DDL and DML statements.Three type triggers: DDL Trigger,DML Trigger,Logon Trigger.

* What are the scenarios to use Triggers?

Using triggers is quite valid when their use is justified. For example, they have good value in auditing (keeping history of data) without requiring explicit procedural code with every CRUD command on every table.

* What is the difference between Trigger and Stored Procedure?

We can call a stored procedure from inside another stored procedure but we can't directly call another trigger within a trigger. We can only achieve nesting of triggers in which the action (insert, delete, and update) defined within a trigger can initiate execution of another trigger defined on the same table or a different table.

We can execute a stored procedure whenever we want with the help of the exec command, but a trigger can only be executed whenever an event (insert, delete, and update) is fired on the table on which the trigger is defined.

**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.

* 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.
* A new region called 'Middle Earth';

insert into Region values(6,“Middle Earth”)

* A new territory called “Gondor”, belongs to region “Middle Earth”;

insert into Territories values(99999,'Gondor',6)

* A new employee “Aragorn King” who's territory is “Gondor”.

INSERT INTO Employees VALUES(10，'Aragorn','King'，'Sales Representative', 'Ms.' ,'1966-01-01 00:00:00.000','1994-01-01 00:00:00.000', 'yolanda dr.', 'Los Angeles', 'Middle Earth' ,'WG2 7LT', 'USA', '(71) 555-4444' ,452,NULL, 'Aragon has a BA degree in English from St. Lawrence College', 5, '<http://accweb/emmployees/davolio.bmp/>')

INSERT INTO EmployeeTerritories VALUES(10,99999)

* Change territory “Gondor” to “Arnor”.

update territories

SET TerritoryDescription = 'Arnor'

WHERE TerritoryDescription = 'Gondor'

* 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.

DELETE from EmployeeTerritories where TerritoryID = (SELECT TerritoryID FROM Territories WHERE TerritoryDescription = 'Arnor')

DELETE from Territories where TerritoryDescription = 'Arnor'

DELETE from Region where RegionDescription = 'Middel Earth'

* 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\_xie

as

select ProductName,sum(quantity) from [order details] od join products p on od.productID=p.productID group by ProductName

* 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 PROCEDURE sp\_product\_order\_quantity\_xie

@ProductID int,

@TotalOrder int out

AS

begin

SELECT @TotalOrder = SUM(Quantity) from [Order Details] od JOIN Products p ON p.ProductID = od.ProductID

where p.ProductID = @ProductID

group by ProductName

GO

* 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\_xie

@ProductName varchar(50)

as

begin

select top 5 shipCity, sum(quantity) from orders o join [order details] od on o.order\_id=od.order\_id join products p on p.productid=od.productid

group by o.shipcity,p.productName

order by SUM(Quantity) DESC

* 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”.

create proc sp\_move\_employees\_xie

as

begin

IF EXISTS(SELECT EmployeeID FROM EmployeeTerritories WHERE TerritoryID = (SELECT TerritoryID FROM Territories WHERE TerritoryDescription ='Tory'))

BEGIN

DECLARE @TerritotyID INT

SELECT @TerritotyID = MAX(TerritoryID) FROM Territories

BEGIN TRAN

INSERT INTO Territories VALUES(@TerritotyID+1 ,'Stevens Point',3)

UPDATE EmployeeTerritories

SET TerritoryID = @TerritotyID+1

WHERE EmployeeID IN (SELECT EmployeeID FROM EmployeeTerritories WHERE TerritoryID = (SELECT TerritoryID FROM Territories WHERE TerritoryDescription ='Tory'))

* 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.

CREATE TRIGGER move\_emp

ON EmployeeTerritories

AFTER INSERT

AS

DECLARE @EmpCount INT

SELECT @EmpCount = COUNT(\*) FROM EmployeeTerritories WHERE TerritoryID = (SELECT TerritoryID FROM Territories WHERE TerritoryDescription = 'Stevens Point' AND RegionID=3) GROUP BY EmployeeID

IF (@EmpCount>100)

BEGIN

UPDATE EmployeeTerritories

SET TerritoryID = (SELECT TerritoryID FROM Territories WHERE TerritoryDescription ='Tory')

WHERE EmployeeID IN (SELECT EmployeeID FROM EmployeeTerritories WHERE TerritoryID = (SELECT TerritoryID FROM Territories WHERE TerritoryDescription ='Stevens Point' AND RegionID=3))

END

DROP TRIGGER move\_emp

COMMIT

* 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\_your\_last\_name

(id int,city varchar(50))

insert into people\_your\_last\_name values( (1,Seattle), (2, Green Bay))

create table city\_your\_last\_name

(id int,name varchar(50),city int)

insert into city\_your\_last\_name values( (1,Aaron Rodgers, 2), (2, Russell Wilson, 1), (3, Jody Nelson, 2))

insert into people\_your\_last\_name values (3,'Madison')

update people\_your\_last\_name

set city='Madison'

where city='Seattle'

delete from city\_your\_last\_name

where city = 'Seattle'

create view Packers\_your\_name

as

begin

SELECT name FROM WHERE city = 'Green Bay'

select \* from Packers\_your\_name

commit

drop table people\_your\_last\_name

drop table city\_your\_last\_name

drop view Packers\_your\_name

* 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\_employee\_gaddam

AS

BEGIN

SELECT \* INTO birthday\_employees\_your\_last\_name

FROM Employees WHERE DATEPART(MM,BirthDate) = 02

SELECT \* FROM birthday\_employees\_your\_last\_name

* 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\_your\_last\_name\_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

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

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

USE EXCEPT OPERATION.

SELECT \* FROM TB1

EXCEPT

SELECT \* FROM TB1

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.

select CASE WHEN [middle name] is null then [First Name]+' '+[Last Name]

else

[First Name]+' '+[Last Name]+' '+[Middle Name]+'.'

end as [Full Name]

from tb1

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.

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

16.

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

How do you out put this?

select \* from students order by sex,marks

GOOD LUCK.