**SQL Assignment 5**

1. Explain how SQL Query keyword statements are executed in order.

The SQL order of execution defines the order in which the clauses of a query are evaluated. Some of the most common query challenges people run into could be easily avoided with a clearer understanding of the SQL order of execution, sometimes called the SQL order of operations.

1. Explain the advantages of stored procedures and their syntax in relation to recompiling stored procedures.

The advantages of stored procedures:

* Since stored procedures are compiled and stored, whenever you call a procedure the response is quick.
* you can group all the required SQL statements in a procedure and execute them at once.
* 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.
* Using procedures, you can avoid repetition of code moreover with these you can use additional SQL functionalities like calling stored functions.
* Once you compile a stored procedure you can use it in any number of applications. If any changes are needed you can just change the procedures without touching the application code.

Store Procedure Syntax:-

CREATE PROCEDURE

SelectAllCustomers

AS

SELECT \* FROM Customers

GO;

1. Give an example of the derived table.

A derived table is a technique for creating a temporary set of records which can be used within another query in SQL.

USE [SQLTEST]

GO

SELECT \* FROM

(

SELECT [EmpID]

,[FirstName]

,[LastName]

,[Education]

,[YearlyIncome]

,[Sales]

,[DeptID]

FROM [EmployeeDetails]

) AS [Derived Employee Details]

WHERE [Sales] > 500

1. What is the database's trigger? Explain the different forms of triggers that can be found in the database.

A trigger is a stored procedure in database which automatically invokes whenever a special event in the database occurs. For example, a trigger can be invoked when a row is inserted into a specified table or when certain table columns are being updated.

* A **database manipulation (DML)** statement (DELETE, INSERT, or UPDATE)
* A **database definition (DDL)** statement (CREATE, ALTER, or DROP).
* A **database operation** (SERVERERROR, LOGON, LOGOFF, STARTUP, or SHUTDOWN).

Triggers can be defined on the table, view, schema, or database with which the event is associated.

1. What are the benefits and drawbacks of triggers?

the advantages of using MySQL triggers −

* **Integrity of data −** With the help of MySQL trigger we can check the integrity of data in the table. In other words, MySQL triggers are the alternative way to check the integrity of data.
* **Useful for catching errors −** MySQL triggers can catch errors in business logic in the database layer.
* **Alternative way to run scheduled tasks −** Actually by using MySQL triggers we do not have to wait to run the scheduled tasks because the triggers are invoked automatically ‘before’ or ‘after’ a modification is done to the data in the table.
* **Auditing −** Actually MySQL triggers are very much useful for the purpose of auditing of the changes made in the table.
* **Prevention of invalid transactions−**MySQL triggers are very useful in the prevention of invalid transactions.
* **Logging of event−**MySQL triggers can log an event and can also store the information on the access of table.

the disadvantages of using MySQL triggers −

* **Cannot replace all validations −** Actually, MySQL triggers cannot replace all the validations and can only provide an extended validation.
* **Invisible from client applications −**Basically MySQL triggers are invoked and executed invisible from the client applications hence it is very much difficult to figure out what happens in the database layer.
* **Impose load on server −**Triggers can impose a high load on the database server.
* **Not recommended for high velocity of data −**Triggers are not beneficial for use with high-velocity data i.e. the data when a number of events per second are high. It is because in case of high-velocity data the triggers get triggered all the time.

1. Create a stored procedure to call other stored procedures.

create table customer

(

cust\_id int not null,

cust\_name varchar(50),

contact\_name varchar(50),

address varchar(50),

postalcode varchar(50),

country varchar(10),

PRIMARY KEY(cust\_id)

)

select \* from customer

insert into customer values(1,'abhi','abhi','mumbai','00000','india');

insert into customer values(2,'ak','ak','dhili','00001','india');

insert into customer values(3,'thor','thor','nanded','000012','india');

insert into customer values(4,'sank','sank','panjab','00013','india');

insert into customer values(5,'niks','niks','hariyana','00004','india');

insert into customer values(6,'kish','kish','gujrat','00005','india');

insert into customer values(7,'bag','bag','him','00006','india');

delimiter &&

create procedure city(in num int )

begin

select \* from customer order by postalcode asc limit num ;

end &&

delimiter ;

call city(4)

Out operation

delimiter &&

create procedure city(out total\_customer int )

begin

select count(\*)from customer into total\_customer ;

end &&

delimiter ;

call city(@total)

select @total as total\_count

in and out operation :-

create table customer

(

cust\_id int not null,

cust\_name varchar(50),

contact\_name varchar(50),

address varchar(50),

postalcode varchar(50),

country varchar(10),

PRIMARY KEY(cust\_id)

)

select \* from customer

select count(\*)from customer

delimiter &&

create procedure city(in var varchar(25),out total\_customer int )

begin

select count(\*)from customer where country=var into total\_customer ;

end &&

delimiter ;

call city('india',@total)

select @total as total\_count