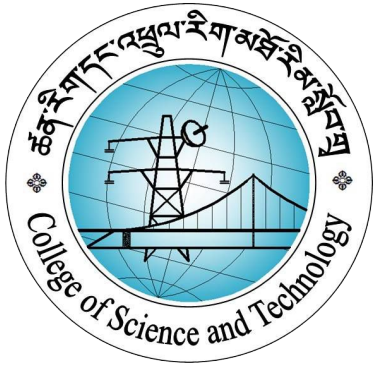

Royal University of Bhutan

DBS101 Database Systems Fundamentals

Lesson 8

Learning Outcomes

1. Implement functions and procedures in SQL
2. Implement triggers in SQL.
3. Write recursive queries in SQL.
4. Use advanced aggregation features in SQL.

Functions and Procedures in SQL

Functions and Procedures: Block of code written to perform a specific task.

Functions and Procedures in SQL: Defines "business logic" to be stored in the database.

Functions and Procedures in SQL

- SQL allows definition of functions and procedures either by the procedural component of SQL or by an external programming language such as Java, C, or C++

Functions Vs Procedure

SL.NO	SQL FUNCTION	SQL PROCEDURE
1	Defined using the CREATE FUNCTION statement	Defined using the CREATE PROCEDURE statement
2	It can be executed from within a SELECT statement or another function	Must be explicitly called using the EXECUTE statement or another procedure
3	Always returns a single value	May or may not return a value
4	Used to perform a single calculation or operation	Used to perform a series of actions

Functions Vs Procedure

SL.NO	SQL FUNCTION	SQL PROCEDURE
5	Can accept input parameters	Can accept input parameters
6	Cannot have output parameters	Can have output parameters
7	SUM, CONCAT, YEAR	Stored Procedures, Triggers, User-Defined Functions

Other Examples

Syntax of Functions

```
create function instructor_of (dept_name varchar(20))
   returns table (
   ID varchar (5),
   name varchar (20),
   dept_name varchar (20),
   salary numeric (8,2))
return table
   (select ID, name, dept name, salary
   from instructor
   where instructor.dept name = instructor of.dept name);
```

Syntax of Procedures

```
create procedure instructor of (dept name varchar(20))
   returns table (
   ID varchar (5),
   name varchar (20),
   dept name varchar (20),
   salary numeric (8,2))
return table
   (select ID, name, dept name, salary
   from instructor
   where instructor.dept name = instructor of.dept name);
```

Language Constructs for Procedures and Functions

SQL supports constructs that give it almost all the power of a general-purpose programming language.

The part of the SQL standard that deals with these constructs is called the **Persistent Storage Module (PSM).**

Language Constructs for Procedures and Functions

```
declare n integer default 0;
for r as
   select budget from department
   where dept_name = 'Music'
   do set n = n- r.budget
end for
```

External Language Routines

Programmers have to learn a new language for each database product.

Although the SQL standard defines the syntax for procedures and functions, most databases do not follow the standard strictly, and there is considerable variation in the syntax supported.

External Language Routines

Functions defined in a programming language and compiled outside the database system may be loaded and executed with the database-system code.

However, doing so carries the risk that a bug in the program can corrupt the internal structures of the database and can bypass the access-control functionality of the database system.

- Executing the code in a sandbox within the database query execution process itself.

External Language Routines

Languages that support external language routines running in a sandbox within the query execution process:

- Oracle and IBM DB2 allow Java
- Microsoft SQL Server allows procedures compiled into the Common Language Runtime (CLR)
- . PostgreSQL allows languages, such as Perl, Python, and Tcl.

Triggers in SQL

A trigger is a statement that the system executes automatically as a side effect of a modification to the database.

Triggers in SQL

- Triggers are essential for enforcing integrity constraints that SQL's constraint mechanism cannot handle.
- Triggers are also useful mechanisms for alerting humans or for starting certain tasks automatically when certain conditions are met.
- Examples include updating related tuples in other relations upon insertion, or automatically placing orders when inventory falls below a minimum level.

Triggers in SQL

- Triggers typically cannot perform updates outside the database, so external actions like placing orders must be handled by separate processes scanning relevant relations.

When not to use Triggers in SQL

- Avoid using triggers when the backup site takes over processing, as it can lead to unintended actions during database replication.
- Triggers should be carefully written to avoid runtime errors that could cause action statement failures or trigger infinite chains of triggering.

When not to use Triggers in SQL

- Some database systems limit trigger chain lengths or flag errors when triggers reference the relation causing their execution.
- Alternatives to triggers, such as stored procedures, are preferable for many applications, offering more control and avoiding potential complications.

Syntax for triggers in SQL

```
create trigger setnull before update of takes
referencing new row as nrow
for each row
when (nrow.grade = ' ')
begin atomic
   set nrow.grade = null;
end;
```

Recursion in SQL

Transitive Closure:

if $(a_1, a_2) \in R$ and $(a_2, a_3) \in R$, then $(a_1, a_3) \in R$.

Recursion in SQL

Transitive Closure:

if $(a_1, a_2) \in R$ and $(a_2, a_3) \in R$, then $(a_1, a_3) \in R$.

- Can be defined in SQL using recursive queries.

Recursion in SQL

- The SQL standard supports a limited form of recursion, using the with recursive clause, where a view (or temporary view) is expressed in terms of itself.
- Recursive queries can be used, for example, to express transitive closure concisely.
- Any recursive view must be defined as the union of two subqueries: a **base query** that is non recursive and a **recursive query** that uses the recursive view

Restrictions on recursive query

- Recursive queries have restrictions; specifically that the query must be monotonic.
- In particular, recursive queries may not use any of the following constructs, since they would make the query nonmonotonic:
  - Aggregation on the recursive view.
  - **not exists** on a subquery that uses the recursive view.
  - Set difference (except) whose right-hand side uses the recursive view.

Flipped Class on Advanced Aggregation Functions

- Join your groups
- Discussion time- 30 mins
- Presentation of demo - 20 mins

Next class:

Unit V : Relational Database Design

5.1 Features of Good Relational Designs
5.2 Decomposition Using Functional Dependencies
5.3 Normal Forms
5.4 Functional-Dependency Theory
5.5 Algorithms for Decomposition Using Functional Dependencies
5.6 Decomposition Using Multivalued Dependencies
5.7 More Normal Forms
5.8 Atomic Domains and First Normal Form
5.9 Database-Design Process
5.10 Modelling Temporal Data