Procedures

MySQL Stored Procedure

A procedure (often called a stored procedure) is a collection of pre-compiled SQL statements stored inside the database. It is a subroutine or a subprogram in the regular computing language. A procedure always contains a name, parameter lists, and SQL statements. We can invoke the procedures by using triggers, other procedures and applications such as Java, Python, PHP, etc. It was first introduced in MySQL version 5. Presently, it can be supported by almost all relational database systems.

If we consider the enterprise application, we always need to perform specific tasks such as database cleanup, processing payroll, and many more on the database regularly. Such tasks involve multiple SQL statements for executing each task. This process might easy if we group these tasks into a single task. We can fulfill this requirement in MySQL by creating a stored procedure in our database.

A procedure is called a recursive stored procedure when it calls itself. Most database systems support recursive stored procedures. But, it is not supported well in MySQL.

Stored Procedure Features

Stored Procedure increases the performance of the applications. Once stored procedures are created, they are compiled and stored in the database.
Stored procedure reduces the traffic between application and database server. Because the application has to send only the stored procedure's name and parameters instead of sending multiple SQL statements.
Stored procedures are reusable and transparent to any applications.
A procedure is always secure. The database administrator can grant permissions to applications that access stored procedures in the database without giving any permissions on the database tables.

MySQL procedure parameter has one of three modes:

IN parameter

It is the default mode. It takes a parameter as input, such as an attribute. When we define it, the calling program has to pass an argument to the stored procedure. This parameter's value is always protected.
OUT parameters

It is used to pass a parameter as output. Its value can be changed inside the stored

procedure, and the changed (new) value is passed back to the calling program. It is noted that a procedure cannot access the OUT parameter's initial value when it starts.
INOUT parameters

It is a combination of IN and OUT parameters. It means the calling program can pass the argument, and the procedure can modify the INOUT parameter, and then passes the new value back to the calling program.

PostgreSQL functions, also known as Stored Procedures, allow you to carry out operations that would normally take several queries and round trips in a single function within the database. Functions allow database reuse as other applications can interact directly with your stored procedures instead of a middle-tier or duplicating code. Functions can be created in a language of your choice like SQL, PL/pgSQL, C, Python, etc.

```
Below is the procedure for insertino of data in table
create or replace procedure
insertempinfo(id integer,empid text, name text, age integer, salary
integer,deptname text)
language plpgsql
AS $$
BEGIN
insert into gbpnihe_schema.employeeinfo
(id,empid,name,age,salary,deptname)
values(id,empid,name,age,salary,deptname);
END
$$
Call insertempinfo(45,'emp45','Ankit','33','60000','HR')
```

```
create or replace function
getempifno (dept1 text)
```

```
returns integer AS $deptempcount$
declare
deptempcount integer;
BEGIN
select count(*) into deptempcount from gbpnihe_schema.employeeinfo
where deptname=dept1;
return deptempcount;
END;$deptempcount$ language plpgsql;
```

Fucntion can be called in where condition
select * from gbpnihe_schema.employeeinfo where id=(SELECT public.getempifno('HR'))