UNIT 2

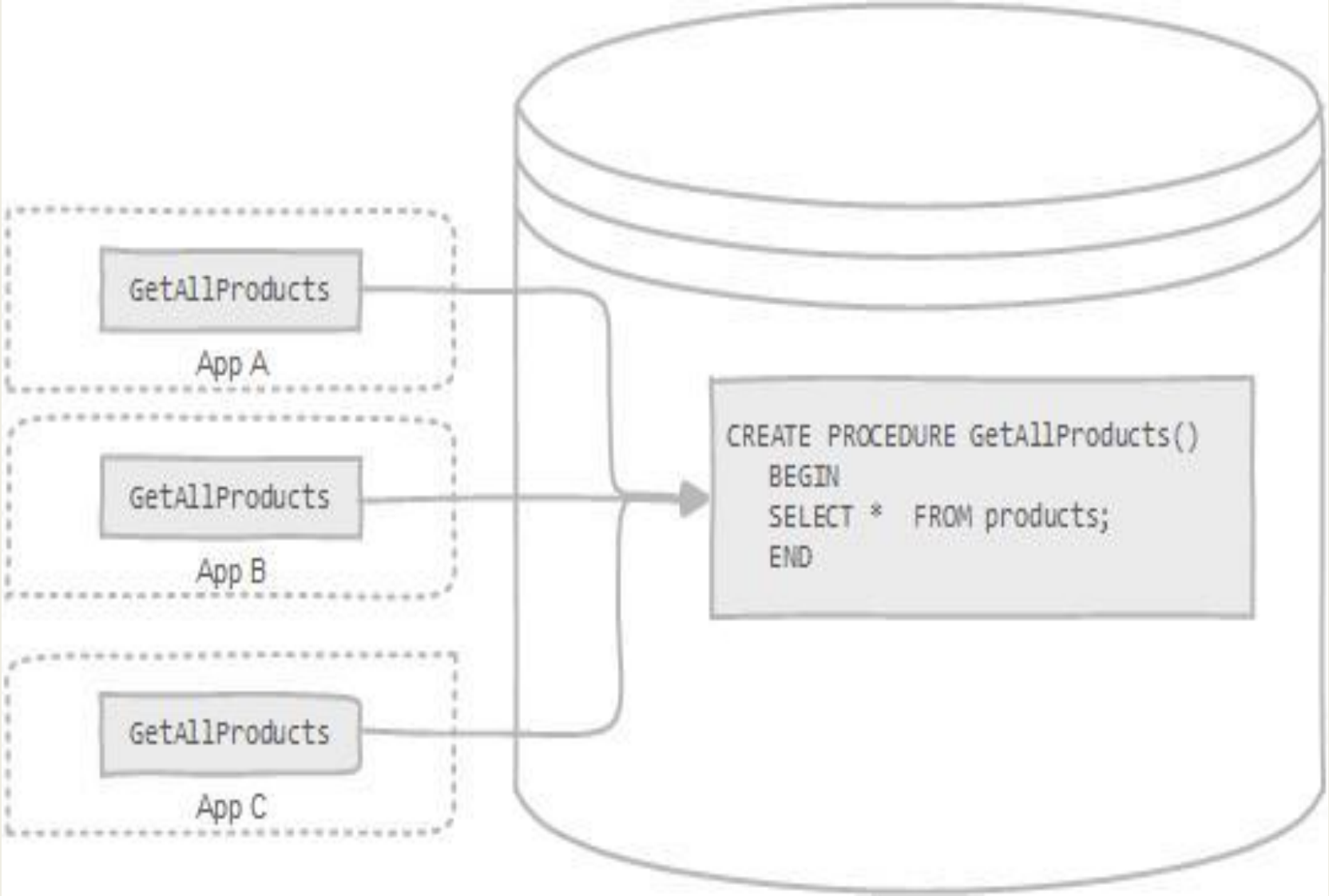
Application Development using Procedural SQL

Topics cover

- Overview of Function and Procedure
- Function and Procedure Usage
- Creation of Stored Procedure
- Calling Stored Programs from Stored Programs
- Creation of User Defined Function
- Calling Function from Stored Programs

Procedure

- Stored Procedure is a logical grouped set of SQL/PL statement that perform a specific task.
- *A stored procedure is a segment of declarative SQL statements stored inside the database catalog.*
- It can help to improve application performance and reduce database access traffic.
- Procedure may or may not return any value.


GetAllProducts
App A
GetAllProducts
App B
GetAllProducts
App C
CREATE PROCEDURE GetAllProducts()
BEGIN
SELECT * FROM products;
END

Structure of stored procedure

```
CREATE PROCEDURE  procedure_name

    (IN   |   OUT   | INOUT   arg1 datatype1,

     IN   |   OUT   | INOUT   arg2 datatype2...)

BEGIN

    ......statement.....

END
```

MODE param_name param_type(param_size)

- The syntax of defining a parameter

Parameter Modes

Parameters in MySQL can be defined as IN, OUT, or INOUT:

- IN
  - *This mode is the default.*
  - *It indicates that the parameter can be passed into the stored program but that any modifications are not returned to the calling program.*
- OUT
  - *The stored program can assign a value to the parameter, and that value will be passed back to the calling program.*
- INOUT
  - *The stored program can read the parameter and that the calling program can see any modifications that the stored program may make to that parameter.*

Advantages of SP

- Stored procedures help increase the performance of the applications.
- Stored procedures help reduce the traffic between application and database server
  - *Because instead of sending multiple lengthy SQL statements, the application has to send only name and parameters of the stored procedure.*
- Stored procedures are reusable and transparent to any applications.
  - *Stored procedures expose the database interface to all applications so that developers don't have to develop functions that are already supported in stored procedures.*
- Stored procedures are secure.
  - *The database administrator can grant appropriate permissions to applications that access stored procedures in the database without giving any permissions on the underlying database tables.*

Disadvantages

- If you use a lot of stored procedures, the memory usage of every connection that is using those stored procedures will increase substantially.
- Constructs of stored procedures make it more difficult to develop stored procedures that have complicated business logic.
- It is difficult to debug stored procedures.
- It is not easy to develop and maintain stored procedures

1. Consider table :

   empBranch (empno, designation, basic_sal, DOB, B_code)

   Branch(B_code, city)

   - Write a procedure that takes city and designation as input parameter. This procedure gives 10% bonus to all employees belonging to that city and having that designation.

1. Consider table :

   bankBranch (custno, cname, Acc_type, Balance, Branch_code)

   Branch(Branch_code, city)

   - Write a procedure that takes city and account type as input parameter. This procedure gives 20% interest to all customers having that account type and belonging to that city.

The IN parameter example

```
CREATE PROCEDURE GetOfficeByCountry(IN countryName VARCHAR(25))
BEGIN
        SELECT * FROM offices WHERE country = countryName;
END //
```

- CALL GetOfficeByCountry('USA')

The OUT parameter example

```
CREATE PROCEDURE GetSalary(IN eid INT, OUT S INT)
BEGIN
        SELECT salary INTO s FROM emp WHERE empid=eid;
        Set s=s+1000;
END$$
```

- CALL GetSalary (3,@s);
- SELECT @s;

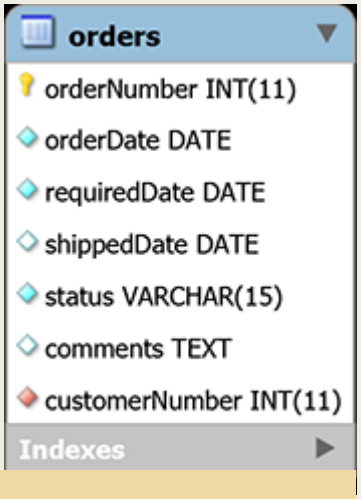
The INOUT parameter example

```
CREATE PROCEDURE set_counter(INOUT count INT(4),IN inc INT(4))
BEGIN
        SET count = count + inc;
END$$
```

```
SET @counter = 1;
CALL set_counter(@counter,1); -- 2
CALL set_counter(@counter,1); -- 3
CALL set_counter(@counter,5); -- 8
SELECT @counter; -- 8
```

Return multiple values example

- MySQL stored function returns only one value.
- To develop stored programs that return multiple values, you need to use stored procedures with INOUT or OUT parameters.




```
CREATE PROCEDURE get_order_by_cust(
IN cust_no INT, OUT shipped INT,OUT canceled INT,OUT resolved INT,OUT disputed INT)
BEGIN
        SELECT count(*) INTO shipped FROM orders WHERE customerNumber = cust_no   AND
        status = 'Shipped';
        SELECT count(*) INTO canceled  FROM orders WHEREcustomerNumber = cust_no AND
        status = 'Canceled';
        SELECT count(*) INTO resolved FROM ordersWHERE   customerNumber = cust_no AND
        status = 'Resolved';
        SELECT count(*) INTO disputed FROM ordersWHERE customerNumber = cust_no AND
        status = 'Disputed';
END
```

FUNCTION

- Function : A function is a logical grouped set of SQL/PL statement that perform a specific task.
- *A stored function is a special kind stored program that returns a single value.*
- PL/SQL functions are created by executing the CREATE FUNCTION statement.
- It always return a value.
- Such functions can be dropped from the database by using the DROP statement.
  - *DROP FUNCTION Function_name;*

Structure of Function

```
CREATE FUNCTION function_name (var1 datatype,var2 datatype) returns
    datatype
BEGIN
    .........SQL PL code...........
    return <value>;
END;
```

Example

```
CREATE FUNCTION simple_function() returns varchar(10)
 BEGIN
                    return 'hello';
  END;
```

Difference between function and procedure

FUNCTION (UDF)	PROCEDURE (SP)
Function always returns a value.	Procedure may or may not return value.
Function is called directly by name of function. E.g.: get_salary(empId)	Procedure is call using CALL statement. E.g.: CALL get_salary(empId)
User-defined functions can not return multiple result sets.	Procedure can return multiple result sets using out variable.
User-defined functions cannot call a stored procedure.	Procedure can call a user define function.