Indian Institute of Information Technology Surat

Lab Report on
Advanced Database Management (CS 604) Practical

Submitted by

[RAHUL KUMAR SINGH] (UI21CS44)

Course Faculty

Mr. Rishi Sharma

Department of Computer Science and Engineering

Indian Institute of Information Technology Surat

Gujarat-394190, India

Jan-2024

Lab No: 6

Aim: To Implement Object Oriented Approach for writing PL/SQL codes (MySQL)

Description:

A) Write a PL/SQL code to create a class for a "Person" with attributes such as name, age, and address.
B) Write a PL/SQL code to Implement methods in the "Person" class to display the details and update the age.
C) Write a PL/SQL code to implement a method to calculate the annual bonus based on the salary in the "Employee" class.
D) Write a PL/SQL code to create a "Manager" subclass inheriting from the "Employee" class, and add an attribute to store the number of employees managed.

Source Code:

A) Implementing a "Person" class:

```
Drop TABLE Person;
CREATE TABLE Person (
  objectId VARCHAR(100) PRIMARY KEY,
  name VARCHAR(100),
  age INT,
  address VARCHAR(200)
);
-- Dropping Procedures
DROP PROCEDURE DisplayDetails;
DROP PROCEDURE UpdateAge;
DROP PROCEDURE AppendPerson;
```

B) Implementing methods to create object, display details and update age:

```
DELIMITER //
CREATE PROCEDURE DisplayDetails(IN object_id VARCHAR(100))
BEGIN
  SELECT name, age, address FROM Person WHERE objectId = object_id;
END //
CREATE PROCEDURE UpdateAge(IN object_id VARCHAR(100), IN new_age INT)
BEGIN
  UPDATE Person SET age = new_age WHERE objectId = object_id;
END //
CREATE PROCEDURE AppendPerson(IN object_id VARCHAR(100), IN person_name VARCHAR(100), IN new_age INT, IN
person_address VARCHAR(100))
BEGIN
  INSERT INTO Person values (object_id, person_name, new_age, person_address);
END //
DELIMITER ;
-- Calling Procedures:
CALL AppendPerson("person1", "Rahul Kumar Singh", 20, "Raigarh");
CALL DisplayDetails("person1");
CALL UpdateAge("person1", 21);
CALL DisplayDetails("person1");
```

C) Implementing methods to create object, display details and calculate the annual bonus based on salary:

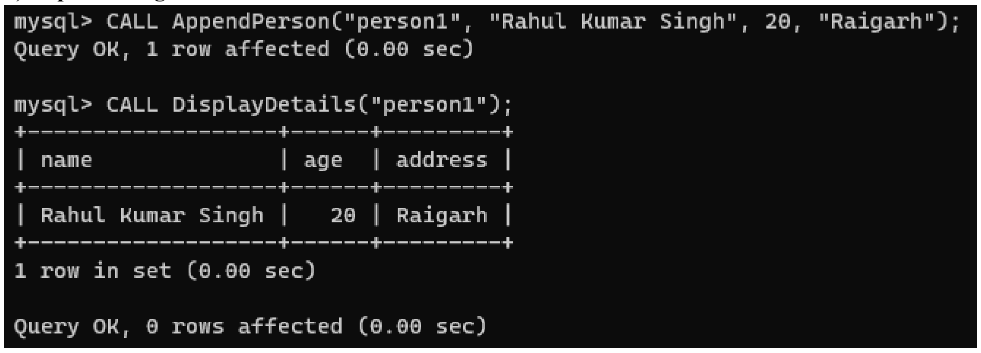
```
-- Dropping Procedures
DROP PROCEDURE DisplayEmpDetails;
DROP PROCEDURE AppendEmployee;
DROP PROCEDURE CalculateAnnualBonus;
DELIMITER //
CREATE PROCEDURE DisplayEmpDetails(IN object_id VARCHAR(100))
BEGIN
  SELECT * FROM Employee WHERE objectId = object_id;
END //
CREATE PROCEDURE AppendEmployee(IN object_id VARCHAR(100), IN person_name VARCHAR(100), IN new_age INT, IN
person_address VARCHAR(100), IN salary DECIMAL(10,2))
BEGIN
  INSERT INTO Employee values (object_id, person_name, new_age, person_address, salary);
END //
CREATE PROCEDURE CalculateAnnualBonus(IN object_id VARCHAR(100))
BEGIN
  SELECT salary * 0.1 FROM Employee WHERE objectId = object_id; -- 10% bonus rate.
END //
DELIMITER ;
-- Calling Procedures
CALL CalculateAnnualBonus(4000);
```

D) Implementing an "Employee" and "Manager" subclass:

```
Drop TABLE Employee;
Drop TABLE Manager;
CREATE TABLE Employee (
  objectId VARCHAR(100) PRIMARY KEY,
  name VARCHAR(100),
  age INT,
  address VARCHAR(200),
  salary DECIMAL(10,2)
);
CREATE TABLE Manager AS
SELECT * FROM Employee;
ALTER TABLE Manager
ADD num_employees_managed INT;
-- Dropping Procedures
DROP PROCEDURE DisplayManDetails;
DROP PROCEDURE CalculateManagerBonus;
DROP PROCEDURE AppendManager;
DELIMITER //
CREATE PROCEDURE DisplayManDetails(IN object_id VARCHAR(100))
BEGIN
  SELECT * FROM Manager WHERE objectId = object_id;
END //
CREATE PROCEDURE AppendManager(IN object_id VARCHAR(100), IN person_name VARCHAR(100), IN new_age INT, IN
person_address VARCHAR(100), IN salary DECIMAL(10,2), IN num_emp INT)
BEGIN
  INSERT INTO Manager values (object_id, person_name, new_age, person_address, salary, num_emp);
END //
CREATE PROCEDURE CalculateManagerBonus(IN object_id VARCHAR(100))
BEGIN
  SELECT salary * 0.15 + num_employees_managed * 1000 FROM Manager WHERE objectId = object_id; -- Bonus
END //
DELIMITER ;
-- Calling Procedures
CALL AppendManager("manager1", "Rahul Kumar Singh", 20, "Raigarh", 10000.00, 10);
CALL DisplayManDetails("manager1");
CALL CalculateManagerBonus("manager1");
```

Output:

A) Implementing a "Person" class:

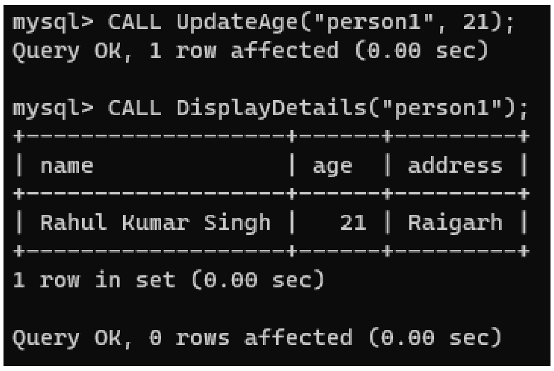


```
mysql> CALL AppendPerson("person1", "Rahul Kumar Singh", 20, "Raigarh");
Query OK, 1 row affected (0.00 sec)

mysql> CALL DisplayDetails("person1");
+-------------------+------+---------+
| name              | age  | address |
+-------------------+------+---------+
| Rahul Kumar Singh |   20 | Raigarh |
+-------------------+------+---------+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)
```

B) Implementing methods to create object, display details and update age:

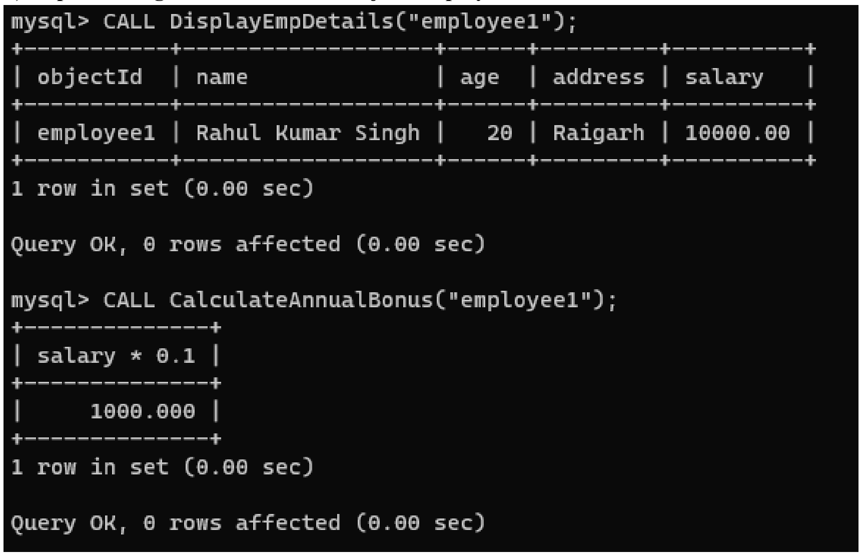


```
mysql> CALL UpdateAge("person1", 21);
Query OK, 1 row affected (0.00 sec)

mysql> CALL DisplayDetails("person1");
+-------------------+------+---------+
| name              | age  | address |
+-------------------+------+---------+
| Rahul Kumar Singh |   21 | Raigarh |
+-------------------+------+---------+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)
```

C) Implementing methods to create object, display details and calculate the annual bonus based on salary:



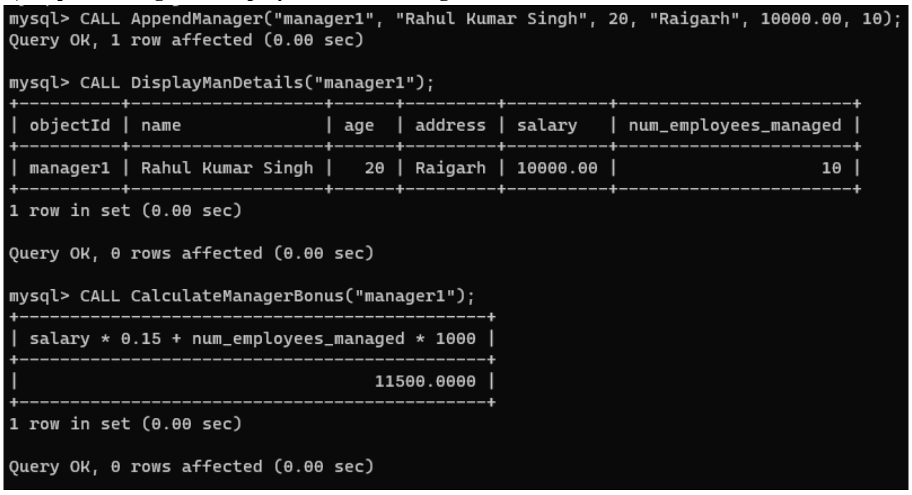
```
mysql> CALL DisplayEmpDetails("employee1");
+-----------+-------------------+------+---------+----------+
| objectId  | name              | age  | address | salary   |
+-----------+-------------------+------+---------+----------+
| employee1 | Rahul Kumar Singh |   20 | Raigarh | 10000.00 |
+-----------+-------------------+------+---------+----------+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

mysql> CALL CalculateAnnualBonus("employee1");
+--------------+
| salary * 0.1 |
+--------------+
|     1000.000 |
+--------------+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)
```

D) Implementing an "Employee" and "Manager" subclass:



```
mysql> CALL AppendManager("manager1", "Rahul Kumar Singh", 20, "Raigarh", 10000.00, 10);
Query OK, 1 row affected (0.00 sec)

mysql> CALL DisplayManDetails("manager1");
+----------+-------------------+------+---------+----------+-----------------------+
| objectId | name              | age  | address | salary   | num_employees_managed |
+----------+-------------------+------+---------+----------+-----------------------+
| manager1 | Rahul Kumar Singh |   20 | Raigarh | 10000.00 |                    10 |
+----------+-------------------+------+---------+----------+-----------------------+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

mysql> CALL CalculateManagerBonus("manager1");
+------------------------------------------+
| salary * 0.15 + num_employees_managed * 1000 |
+------------------------------------------+
|                               11500.0000 |
+------------------------------------------+
1 row in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)
```

Conclusion:

- Help us to understand the importance of object-oriented approach.
- Provide various features of object-oriented approach like Polymorphism, Inheritance and Encapsulation.
- To be able to implement the MySQL code into an Object-oriented programming model.