Lecture Notes – VariableStoredProcedures

Section 1: Lecture Summary

The lecture covers using variables in stored procedures in MySQL, including syntax for declaring variables with the **DECLARE** keyword, assigning query results to variables using **SELECT ... INTO**, and executing procedures with **CALL**. Two examples demonstrate this: one counting total orders from the **Orders** table and displaying the result, and another calculating a customer's lifetime spending by summing **TotalAmount** from **Orders** for a given **CustomerID** parameter.

Section 2: Key Concepts and Explanations

Stored procedures follow the syntax **CREATE PROCEDURE procedure_name() BEGIN ... END**, where variables are declared inside the **BEGIN...END** block using **DECLARE variable_name datatype;**. Variables store single values from queries via **SELECT aggregate(*) INTO variable_name FROM table [WHERE condition];**. Procedures without parameters are called as **CALL procedure_name();**, while those with **IN** parameters use **CALL procedure_name(parameter_value);**. Results are output using **SELECT variable_name;** at the end of the procedure. Data types like **INT** for counts or **DECIMAL(10,2)** for monetary sums must match the expected query output.

Section 3: Example Code and Use Cases

Example 1: Procedure to display total number of orders using a variable.

```
DELIMITER //
CREATE PROCEDURE total_orders()
BEGIN
    DECLARE total INT;
    SELECT COUNT(*) INTO total FROM Orders;
    SELECT total;
END //
DELIMITER ;
```

Call the procedure:

```
CALL total_orders();
```

This returns 20 based on the **Orders** table row count.

Example 2: Procedure to fetch a specific customer's lifetime spending using a parameter and variable.

```
DELIMITER //
CREATE PROCEDURE customer_lifetime_value(IN p_cus_id INT)
BEGIN
    DECLARE total_spend DECIMAL(10,2);
    SELECT SUM(TotalAmount) INTO total_spend
    FROM Orders
    WHERE CustomerID = p_cus_id;
    SELECT total_spend;
END //
DELIMITER ;
```

Call the procedure for CustomerID 2:

```
CALL customer_lifetime_value(2);
```

This returns 150500.00, summing **TotalAmount** for Customer 2's orders.

Section 4: Key Takeaways

- Use **DECLARE** inside **BEGIN...END** to define variables with appropriate data types before using them.

- Assign query results to variables with **SELECT ... INTO variable**; variables hold single values.

- Output variable values with a final **SELECT** statement in the procedure.

- Procedures store reusable SQL logic and can accept **IN** parameters for dynamic inputs like **CustomerID**.

- Practice creating and calling procedures with **eCommerceDB** tables like **Orders** to reinforce variable usage.