Lecture Notes – IntroductionToStoredProcedures

Section 1: Lecture Summary

Stored procedures are SQL programs stored in the database that are pre-compiled for reuse. They contain SQL statements and can be called from application programs without requiring direct SQL knowledge from programmers, enhancing security and reducing development effort. Key aspects include syntax using **CREATE PROCEDURE**, **DELIMITER //** for handling semicolons, **BEGIN...END** block, and execution with **CALL**. Example creates a simple stored procedure to retrieve all customers from the **Customers** table in eCommerceDB.

Section 2: Key Concepts and Explanations

Stored procedures are reusable database objects stored in the data dictionary, similar to tables defined by DDL. They function as database APIs for application development, allowing access without exposing raw SQL queries. **Key features** include support for parameters, variables, conditional statements, and loops, mixing SQL with programming logic. **Types** categorized as simple (basic SQL statements), parameterized (with parameters and variables), and those with control flow (conditionals or loops). **Syntax** requires **DELIMITER //** before **CREATE PROCEDURE procedure_name() BEGIN SQL_statement; END //** and **DELIMITER ;** after, due to semicolons marking statement ends. Execution uses **CALL procedure_name();**.

Section 3: Example Code and Use Cases

```
DELIMITER //
CREATE PROCEDURE GetAllCustomers()
BEGIN
  SELECT * FROM Customers;
END //
DELIMITER ;
```

This creates a simple stored procedure in eCommerceDB to retrieve all data from **Customers** (**CustomerID**, **FirstName**, **LastName**, **Gender**, **Email**, **City**, **JoinDate**).

```
CALL GetAllCustomers();
```

Executes the procedure, displaying all customer records. Stored procedures appear under the database in MySQL Workbench after refresh and can be called repeatedly.

Section 4: Key Takeaways

Stored procedures improve security by limiting direct data access, are pre-compiled for efficiency, and support parameters and control structures. Use **DELIMITER //** to define due to internal semicolons. Simple procedures handle basic SELECTs like retrieving from **Customers**; advanced types with parameters and logic follow in subsequent coverage.