Stored Procedures

• Definition
  – A stored procedure is a subroutine available to applications accessing a relational database system. A stored procedure (sometimes called a sproc or an SP) is actually stored in the database.
• Stored procedures can receive input parameters and may return results
• Stored procedures can be called from:
  – Programs written in standard languages, e.g., Java, C#
  – Scripting languages, e.g., JavaScript, VBScript, PHP
  – SQL command prompt, e.g., SQL*Plus, Query Analyzer

Advantages of stored procedures

• Performance
  – Compiled once
  – Procedure calls are quick and efficient
  – Server side computation
  – Decreased network traffic
  – Executable code is cached and shared
  – Lowers memory requirements
  – Grouping SQL statements allows for single call execution
Advantages of stored procedures

**Productivity and Ease of Use**
- Improved Development Efficiency
  - Avoids redundant coding
  - Specialization
  - Common integration/development standards
  - Can be called by practically any app

**Security**
- Limit/restrict users' access to the DB by using SP as middleware or buffer

Stored procedures in Access

- A stored "procedure" in MS Access is simply a stored (typically parameterized) query
  - Access doesn't support multiple statements

  **In Access:** `UpdateProc` (stored as a query):
  ```
  UPDATE Employee SET Emp_LastName = ?
  WHERE Emp_ID = ?
  ```

  **In VB.NET:**
  ```
  cmd.CommandText = "exec UpdateProc 'Williams', 205"
  cmd.ExecuteNonQuery()
  ```

Stored procedures in MySQL

See handout:

“Example of a parameterized query as a stored procedure”
Persistent stored modules (PSM)

• SQL itself does not support control statements such as looping operations
• To support this, the SQL-99 standard defines the use of **persistent stored modules** - blocks of code (SQL and procedures) that are stored and executed at the DBMS
• Different DBMS vendors have thus developed internal programming language extensions to create procedures
  – SQL Server: **Transact-SQL**
  – Oracle: **Procedural SQL (PL/SQL)**

---

PL/SQL Example

DECLARE
W_P1 NUMBER(3) := 0;
W_P2 NUMBER(3) := 0;
W_NUM NUMBER(2) := 0;
BEGIN
WHILE W_P2 < 300 LOOP
  SELECT COUNT(P_Code) AS W_NUM FROM PRODUCT
  WHERE P_PRICE BETWEEN W_P1 AND W_P2;
  DBMS_OUTPUT.PUT_LINE('There are ' || W_NUM || ' Products ');
  W_P1 := W_P2 + 1;
  W_P2 := W_P2 + 50;
END LOOP;
END;

---

Triggers

• A trigger is a procedure that is **automatically** executed by the RDBMS when a given data manipulation event occurs
  – It is invoked either before or after a data row is inserted, updated, or deleted
  – It is associated with a database table
    • Each database table may have one or more triggers
  – It is used to automate critical actions / provide warnings
  – It can be used to enforce constraints
    • Often used to enforce referential integrity

---
**Trigger example (Oracle)**

```sql
CREATE OR REPLACE TRIGGER TRG_LINE_PROD
AFTER INSERT ON LINE
FOR EACH ROW
BEGIN
    UPDATE PRODUCT
    SET P_QOH = P_QOH - NEW.LINE_UNITS
    WHERE PRODUCT.P_CODE = NEW.P_CODE;
END;
```

*:NEW is a reference to the new, changed values
*:OLD would refer to the original values

---

**Trigger example (MySQL)**

```sql
Delimiter //
CREATE TRIGGER update_supplierID
AFTER UPDATE ON Supplier
FOR EACH ROW
BEGIN
    UPDATE item SET item.Supplier_ID = NEW.Supplier_ID
    WHERE item.Supplier_ID = OLD.Supplier_ID;
END
```

---