Florida Institute of Technology

Advanced SQL

- Assertions
- Triggers
- Stored Procedures
- Embedded & Dynamic SQL
- ODBC & JDBC

Assertions

- An *assertion* is a predicate expressing a condition that we wish the database always to satisfy.
- Similar to DDL check constraints, but they can test conditions across multiple tables.
- When an assertion is made, the system tests it for validity, and tests it again on every update that may violate the assertion.

Florida Institute of Technology

Assertion Example

"The sum of all loan amounts for each branch must be no greater than the sum of all account balances at the branch."

```
create assertion sum-constraint check
    (not exists (select * from branch
                 where (select sum(amount) from loan
                        where loan.branch-name = branch.branch-name)
                       > (select sum(balance) from account
                          where account.branch-name = branch.branch-name)))
```

Assertion Example

"Every loan has at least one borrower who maintains an account with a minimum balance of $1000.00"

```
create assertion balance-constraint check
    (not exists (
        select loan-number from loan
        where not exists (
            select borrower.customer-name from borrower, depositor, account
            where loan.loan-number = borrower.loan-number
                and borrower.customer-name = depositor.customer-name
                and depositor.account-number = account.account-number
                and account.balance >= 1000)))
```

Triggers

- A *trigger* is a statement that is executed automatically by the system as a side effect of a modification to the database.

- A trigger has two parts:
  - conditions
  - actions

Trigger Example

- Suppose the bank deals with overdrafts by:
  - Setting the account balance to zero
  - Creating a loan in the amount of the overdraft
- Condition:
  - update to the account relation that results in a negative balance.
- Actions:
  - Create a loan tuple
  - Create a borrower tuple
  - Set the account balance to 0

Trigger Example in SQL:1999

```
create trigger overdraft-trigger after update on account
referencing new row as nrow
for each row
when nrow.balance < 0
begin atomic
        insert into loan values
           (nrow.account-number, nrow.branch-name, – nrow.balance);
        insert into borrower
           (select depositor.customer-name, depositor.account-number
            from depositor
            where nrow.account-number = depositor.account-number);
        update account set balance = 0
           where account.account-number = nrow.account-number
end
```

Florida Institute of Technology

Triggering Events and Actions in SQL

- Triggering event:
  - **insert**, **delete** or **update.**
- Triggers on update can be restricted to specific attributes:
  - **create trigger** *overdraft-trigger* **after update of** *balance* **on** *account*
- Values of attributes before and after an update can be referenced
  - **referencing old row as** (deletes and updates)
  - **referencing new row as** (inserts and updates)

Florida Institute of Technology

When Not To Use Triggers

- Triggers, along with all the other integrity checking mechanisms, provide yet another opportunity to…slow up the database…
- Triggers can be used for many things:
  - Maintaining summary or derived data (e.g. total salary of each department).
  - Replicating databases.
- DBMSs have better, more efficient ways to do many of these things:
  - Materialized views - maintain summary data.
  - Data warehousing - maintaining summary/derived data.
  - Built-in support for replication.

Procedural Extensions and Stored Procedures

- SQL provides a **module** language that permits definition of procedures:
  - Conditional (if-then-else) statements
  - Loops (for and while)
  - Procedure definition with parameters
  - Arbitrary SQL statements

- Stored Procedures:
  - Stored in the DBMS.
  - Executed by calling them by name, on the command-line or from a program.
  - Permit external applications to operate on the database without knowing about internal details about the database or even SQL.
  - A standard that is not uncommon – put all queries in stored procedures; applications are then only allowed to call stored procedures.
  - In the simplest case, a stored procedure simply contains a single query.

Procedural Extensions and Stored Procedures

- Example:

```
CREATE PROCEDURE stpgetauthors
   @surname varchar(30)=null
 AS
BEGIN
   IF @surname = null
      BEGIN
         RAISERROR( 'No selection criteria provided !', 10, 1)
      END
   ELSE
      BEGIN
         SELECT * FROM authors
            WHERE au_lname LIKE @surname
      END
END
```

Florida Institute of Technology

Submitting Queries from Programs

- Programmatic access to a relational database:
  - Embedded SQL
  - Dynamic SQL
- Standards for Dynamic SQL:
  - ODBC
  - JDBC

Example - ODBC

- *Open DataBase Connectivity (ODBC)* is a standard for programs to communicate with database servers.
  - Independent of language, DBMS or operating system.
- ODBC defines an API providing the functionality to:
  - Open a connection to a database
  - Execute queries and updates
  - Get back results

ODBC (Cont.)

- An ODBC program first allocates an “SQL environment,” and then a “database connection handle.”
- An ODBC program then opens the database connection using SQLConnect() with the following parameters:
  - connection handle
  - server to connect to
  - userid
  - password
- Must also specify types of arguments:
  - SQL_NTS denotes previous argument is a null-terminated string.

ODBC Code

```
int ODBCexample()
    {
        HENV            env;     /* environment */
        HDBC            conn;  /* database connection */
        SQLAllocEnv(&env);
        SQLAllocConnect(env, &conn);
        SQLConnect(conn,
              "aura.bell-labs.com", SQL_NTS,
              "avi", SQL_NTS, "avipasswd", SQL_NTS);

       { …. Do actual work … }

        SQLDisconnect(conn);
        SQLFreeConnect(conn);
        SQLFreeEnv(env);
    }
```

ODBC Code (Cont.)

- Main body of program (i.e., “Do actual work”):

```
char        branchname[80];
float       balance;
int         lenOut1, lenOut2;
HSTMT       stmt;
RETCODE     error;   /* query return code */


SQLAllocStmt(conn, &stmt);
char* sqlquery = "select branch_name, sum (balance)
                  from account
                  group by branch_name";
error = SQLExecDirect(stmt, sqlquery, SQL_NTS);
if (error == SQL_SUCCESS) {
    SQLBindCol(stmt, 1, SQL_C_CHAR,   branchname , 80, &lenOut1);
    SQLBindCol(stmt, 2, SQL_C_FLOAT, &balance,       0 , &lenOut2);
    while (SQLFetch(stmt) >= SQL_SUCCESS) {
        printf (" %s  %g\n", branchname, balance);
    }
}
SQLFreeStmt(stmt, SQL_DROP);
```

JDBC

- JDBC is a Java *specific* API for communicating with database systems supporting SQL.
- JDBC supports a variety of features for querying and updating data, and for retrieving query results.
- Similar to ODBC in general structure and operation:
  - Open a connection
  - Create a “statement” object
  - Execute queries using the Statement object to send queries and fetch results
  - Exception mechanism to handle errors