DATABASDESIGN FÖR INGENJÖRER - 1DL124

Sommar 2005

En introduktionskurs i databassystem

http://user.it.uu.se/~udbl/dbt-sommar05/
alt. http://www.it.uu.se/edu/course/homepage/dbdesign/st05/

Kjell Orsborn
Uppsala Database Laboratory
Department of Information Technology, Uppsala University,
Uppsala, Sweden
Database API:s

(Elmasri/Navathe ch. 9)

Kjell Orsborn

Department of Information Technology
Uppsala University, Uppsala, Sweden
Database user interfaces

• **Textual interfaces**
  – Such as BSQL for Mimer

• **Graphical interfaces**
  – Most well-known is QBE (Query-By-Example) originally developed by IBM. MS Access uses a QBE variant.

• **SQL application programming interfaces**
  – Requires management of sessions, sql statements and some control of query optimization.
  – Call-level interfaces
  – Embedded SQL
Call-Level Interfaces

- **Vendor-specific call-level interfaces**
  - An SQL API usually for one or several host languages like C, C++, Java, Fortan, COBOL etc.
  - Support to manage sessions, SQL statements and data conversions

- **SQL Call Level Interface (CLI),**
  - The Call Level Interface (CLI) is a standard SQL API created by The Open Group. The API is defined for C and COBOL only. ISBN: 1-85912-081-4, X/Open Document Number: C451, 1995.

- **SQL/CLI**
  - Call-Level Interface (SQL/CLI) is an implementation-independent CLI to access SQL databases. SQL/CLI is an ISO standard ISO/IEC 9075-3:1995 Information technology -- Database languages -- SQL -- Part 3: Call-Level Interface (SQL/CLI). The current SQL/CLI effort is adding support for SQL:1999.

- **ODBC**
  - (Microsoft) Open Database Connectivity is a standard SQL API. ODBC is based on the Call Level Interface (CLI) specifications from SQL, X/Open (now part of The Open Group), and the ISO/IEC. ODBC was created by the SQL Access Group and released Sept, 1992.

- **JDBC - Java Database Connectivity**
  - JDBC is an SQL API for Java (to be strictly correct, JDBC is not an acronym).
The ODBC architecture

- ODBC API is independent of any one programming language, database system or operating system.
The JDBC architecture

- JDBC API is independent of (relational) DBMS and operating system
Alt. JDBC architecture (JDBC-ODBC bridge)

- Makes ODBC accessible from JDBC such that no special JDBC drivers are required.
Programming with SQL CLI interfaces

- Query: we would like to print the name of all employees that earn more than 100000 in a program using JDBC:
- Source table: EMPLOYEE(SSN, NAME, INCOME)

```java
//Load SUN's JDBC-ODBCbridge:
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
//Connect to database server:
Connection con = DriverManager.getConnection("jdbc:odbc:S", "myLogin","myPassword");
//Create JDBC statement holder:
Statement stmt = con.createStatement();
String query = "SELECT NAME FROM EMPLOYEE WHERE INCOME > 10000";
ResultSet rs = stmt.executeQuery(query);
//Iterate over query result:
while(rs.next()) //while tuples in result set do
{
    String s = rs.getString("NAME");
    System.out.println(s);
}
```
Programming with SQL CLI interfaces cont.

• In the following code we have a loop that reads the lower income level from the user and prints out all employees that earn more (the `prepareStatement` method compiles and optimizes (prepares) the parameterized query).

```java
{ 
    // Load SUN's JDBC-ODBC bridge:
    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
    // Connect to database server S:
    Connection con = DriverManager.getConnection("jdbc:odbc:S", "myLogin", "myPassword");
    Statement stmt = con.createStatement();
    // Parameterized query:
    String query = "SELECT NAME FROM EMPLOYEE WHERE INCOME > ?";
    Int incomeLimit;  // Prepare parameterized query:
    PreparedStatement getPersons = con.prepareStatement(query);
    while(....){
        // Code to read lower income limit into incomeLimit
        ....
        // Set 1st parameter to incomeLimit:
        getPersons.setInt(1,incomeLimit);
        ResultSet rs = stmt.executeQuery(getPersons);
        // Iterate over query result:
        while (rs.next()) {
            String s = rs.getString("NAME");
            System.out.println(s);
        }
    }
}
```
Embedded SQL

- Host language include embedded and specially marked SQL statements.
- Embedded statements are extracted by preprocessor, translated and replaced by database calls, precompiled (prepared) and stored on server.
- The preprocessed application is then compiled normally
- Supports dynamic recompilation
- Reduces optimization cost and can be somewhat simpler than CLI programming.