



# **EMBEDDED SQL**

# SQL is very high-level, but

- Too low level for end-users
- Requires detailed knowledge of DB schema
- Unsatisfactory as a user interface
- Solution: *End user sees only an interface written by a DB programmer*
  - ▣ "host language" such as COBOL, C, VB, etc.
  - ▣ program contains the SQL, hard-coded or dynamically generated

# "Impedance" Mismatch

- Host language doesn't understand SQL
- Host language data types vs SQL data types
- Conventional imperative languages don't match declarative outlook of SQL
  - ▣ Whole table operations
  - ▣ Say "what" not "how"
- Must somehow connect to the DBMS

# Solutions

- "Cursor" concept
  - ▣ expose one row at a time
  - ▣ process a result table like a sequential file
- Embedded SQL
  - ▣ Standard SQL statements in host source, translated by preprocessor
- DBMS via an API (Application Programming interface)
  - ▣ Calls use host language conventions

# CLI: Call-level Interface

- DB APIs have been around for decades
  - ▣ relational and non-relational
- "The" CLI is a recently standardized API
  - ▣ Calls to SQL in host language syntax
  - ▣ Added to the SQL Standard in 1995
  - ▣ Based on Microsoft ODBC (Open Database Connectivity)
  - ▣ Program declares a struct (class) for each row type (i.e. for each table)
    - "data exchange" supports data type conversion between DB format and host (e.g. C++)

# Embedded SQL Programming Guide

## Alternatives for Coding DB2 Applications:

There are three main alternatives for coding a DB2 application:

- You can use embedded SQL, which enables your application to perform any task or series of tasks that you can perform using standard SQL statements. By embedding SQL statements, your applications can work with the database, and access and manipulate data using one of the supported host programming languages. Embedded SQL is discussed in "Embedding SQL Statements in a Host Language".

- You can use the REXX interpretive language to code your DB2 applications and immediately execute them without having to pre-compile, compile, or link, as you do when you use host languages. REXX is discussed in "Interactive Programming Using REXX".
- You can use the DB2 Call Level Interface (DB2 CLI) to increase the portability of your applications by enabling independence from any one database vendor's programming interface. DB2 CLI is discussed in "Programming With the DB2 Call Level Interface (CLI)".


# Embedding SQL Statements in a Host Language

- Applications can be written with SQL statements embedded within a host language. The SQL statements provide the database interface, while the host language provides the remaining support needed for the application to execute.
- Embedding SQL Statements  
C/C++

```
strcpy( statement, "UPDATE staff SET job = 'Clerk' WHERE  
job = 'Mgr'"); EXEC SQL EXECUTE IMMEDIATE :statement;  
if ( SQLCODE < 0 )  
printf( "Update Error: SQLCODE = %ld \n", SQLCODE );
```



- In this example, the SQLCODE field of the SQLCA structure is checked to determine whether the update was successful.
- SQL statements placed in an application are not specific to the host language. The database manager provides a way to convert the SQL syntax to something the host language can process.
- For the C, C++, COBOL or FORTRAN language, this conversion is handled by the DB2 pre-compiler. The pre-compiler converts embedded SQL statements directly into DB2 run-time services API calls. Note that compilers with integrated preprocessor support are available from non-IBM vendors. With these compilers, the conversion and compilation can be handled in one step without invoking the PREP command. The IBM pre-compiler is invoked using the PREP command.

- 
- When the pre-compiler processes a source file, it specifically looks for SQL statements and avoids the non-SQL host language. It can find SQL statements because they are surrounded by special delimiters.

## □ Language: Example Source Code

### C/C++

EXEC SQL

-- SQL, or C (/\* \*/) or C++ (//) comments allowed here

DECLARE C1 CURSOR FOR sname; EXEC SQL ROLLBACK  
WORK

-- SQL comments or

/\* C comments or \*/

// C++ comments allowed here ;

/\* Only C or C++ comments allowed here \*/

EXEC SQL ROLLBACK WORK;

/\* Only C or C++ comments allowed here \*/