# Writing Database Applications using Embedded SQL

There are three different ways of writing your application when you want to use SQL:

* Direct Invocation: directly invoke SQL statements
* Embedded SQL: embed SQL statements directly in your programming language
* Module Language: write SQL statements separately in a module and call them from your host language program.

Therefore, SQL statements and declarations can be embedded, or included directly into another language.

Embedded commands can be inserted wherever a programming command could be specified.

SQL provides support for seven languages; Ada, C, COBOL, FORTRAN, MUMPS, Pascal and PL/I.

Traditional programming languages can consist of embedded SQL statements and declarations where the actions of the programming language cause the SQL statements to be executed. Execution is either static or dynamic.

Static execution:

If you know the exact text of your SQL statements at the time of writing your application, then you can specify the exact wording. This is called static SQL because the source text of the statements does not change while your application is running.

Dynamic execution:

When the precise text of the statements is not known, the programmer cannot include these in the application. This is called dynamic SQL.

Rules for embedding SQL:

* Everything must be preceded by EXEC SQL, the only exceptions to this are BEGIN DECLARE SECTION, END DECLARE SECTION, and the ESCAPE clause.
* You may have to end the statement with some terminator i.e. “;” for C.
* Lower-case letters are allowed for SQL keywords and user-defined names.

Special rules for C:

* The terminator is a semicolon
* SQL constructs must not be within an #include file and must not be modified by a #define directive.
* Statements may be specified anywhere a C statement may be specified within a function block.

Every static embedded SQL application consists of:

* Declaratives
* Statements