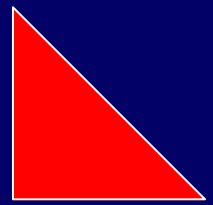
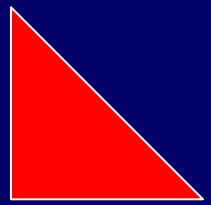


Embedded SQL

Unit 5.1

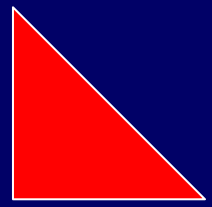
Interactive SQL



So far in the module we have considered only the SQL queries which you can type in at the SQL prompt. We refer to this as 'interactive' SQL.

- SQL is a 'non-procedural' language
- SQL specifies WHAT is required, not HOW this requirement is to be met.
- INTERACTIVE SQL is good for:
  - defining database structure
  - generating low-volume, ad hoc queries
  - prototyping
- INTERACTIVE SQL is not good for the more sophisticated applications for which a programming language with links to SQL might be better.

Embedded SQL

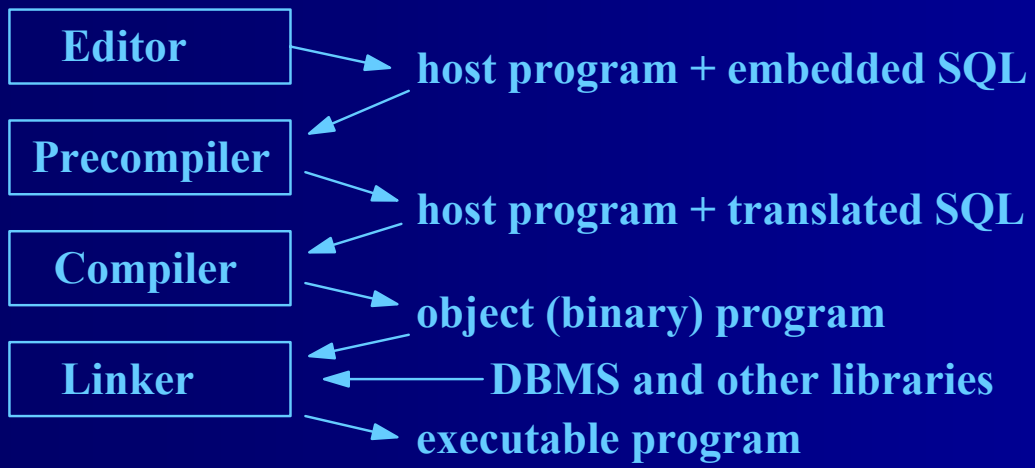


SQL can be embedded within procedural programming languages. These language (sometimes referred to as 3GLs) include C/C++, Cobol, Fortran, and Ada. Thus the embedded SQL provides the 3GL with a way to manipulate a database, supporting:

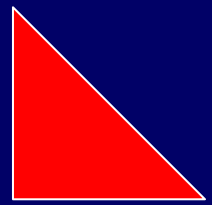
- highly customized applications
- background applications running without user intervention
- database manipulation which exceeds the abilities of simple SQL
- applications linking to Oracle packages, e.g. forms and reports
- applications which need customized window interfaces

SQL Precompiler

A precompiler is used to translate SQL statements embedded in a host language into DBMS library calls which can be implemented in the host language.



Sharing Variables



Variables to be shared between the embedded SQL code and the 3GL have to be specified in the program.

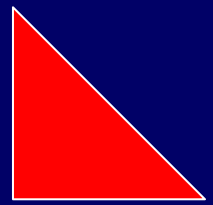
```
EXEC SQL begin declare section;
    varchar userid[10],password[10],cname[15];
    int cno;
EXEC SQL end declare section;
```

We also should declare a link to the DBMS so that database status information can be accessed.

```
EXEC SQL include sqlca;
```

This allows access to a structure sqlca, of which the most common element sqlca.sqlcode has the value 0 (operation OK), >0 (no data found), and <0 (an error).

Connecting to the DBMS



Before operations can be performed on the database, a valid connection has to be established.

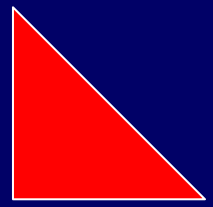
```
EXEC SQL connect :userid identified by :password;
```

- In all SQL statements, variables with the ':' prefix refer to shared host variables, as opposed to database variables (e.g. row or column identifiers).
- This assumes that userid and password have been properly declared and initialised.

When the program is finished using the DBMS, it should disconnect using:

```
EXEC SQL commit release;
```

Queries producing a single row



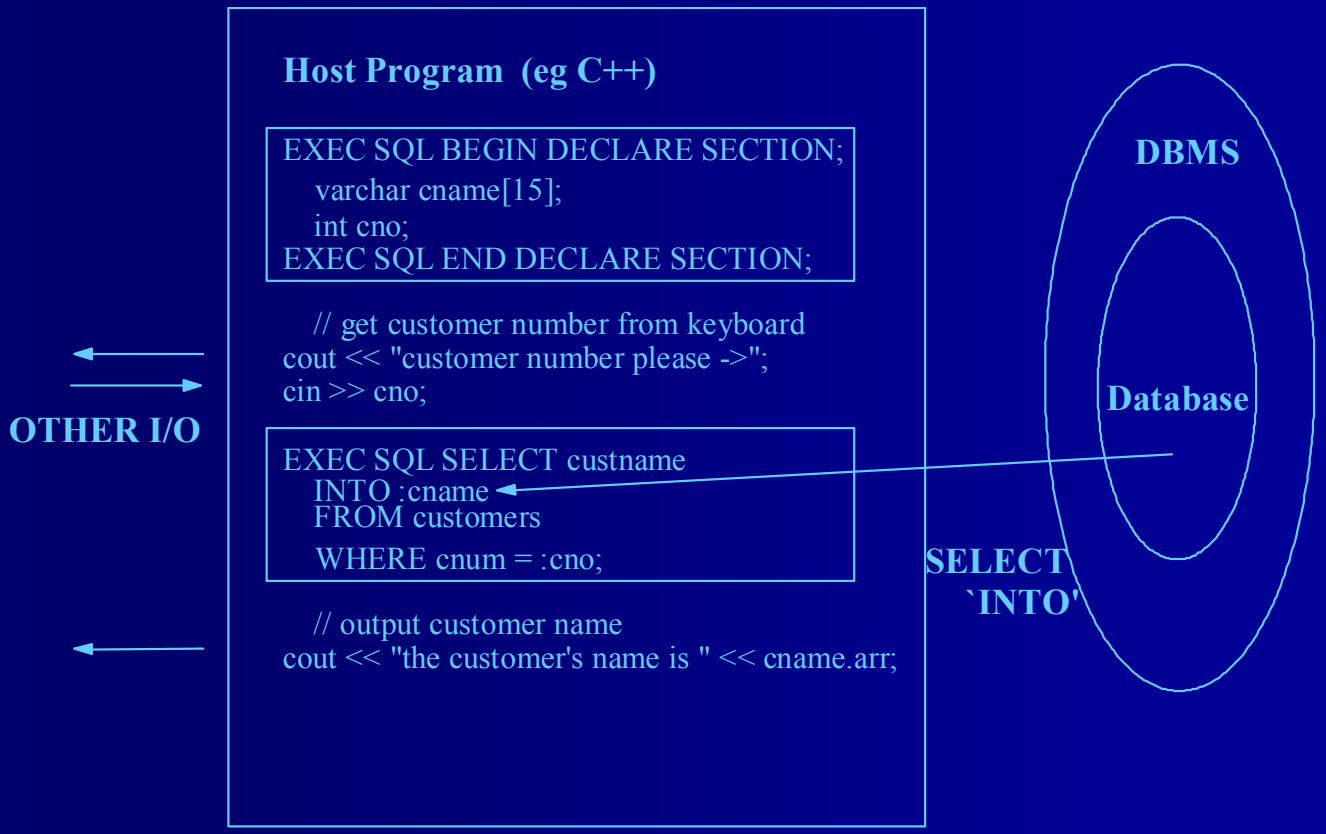
A single piece of data (or row) can be queried from the database so that the result is accessible from the host program.

```
EXEC SQL    SELECT    custname
            INTO      :cname
            FROM      customers
            WHERE     cno = :cno;
```

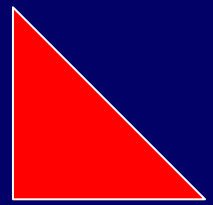
Thus the custname with the unique identifier :cno is stored in :cname.

However, a selection query may generate many rows, and a way is needed for the host program to access results one row at a time.

SELECT with a single result

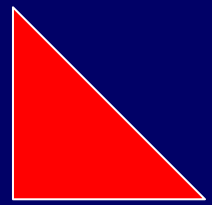


Cursors - SELECT many rows



- A cursor provides a pointer to a single row in the result of a selection query (which may return may rows)
- Cursors are declared by statements similar to view definitions
- One row at a time is accessed through the cursor, which is moved to the next row before each data transfer
- The columns of that one row are 'fetched' into the program variables which can then be manipulated in the normal way by the host program.
- A cursor can also be used to update values in tables.

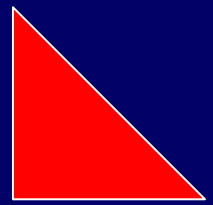
Fetching values



EXEC SQL fetch <cursor-name> into <target-list>

- this moves the cursor to the next row of the select query result and transfers the values from the result row into the program variables specified in target-list.
- a special value is returned when an attempt is made to fetch a non-existent row after the last row available to the cursor, (sqlca.sqlcode > 0)

Declaring and Opening a Cursor



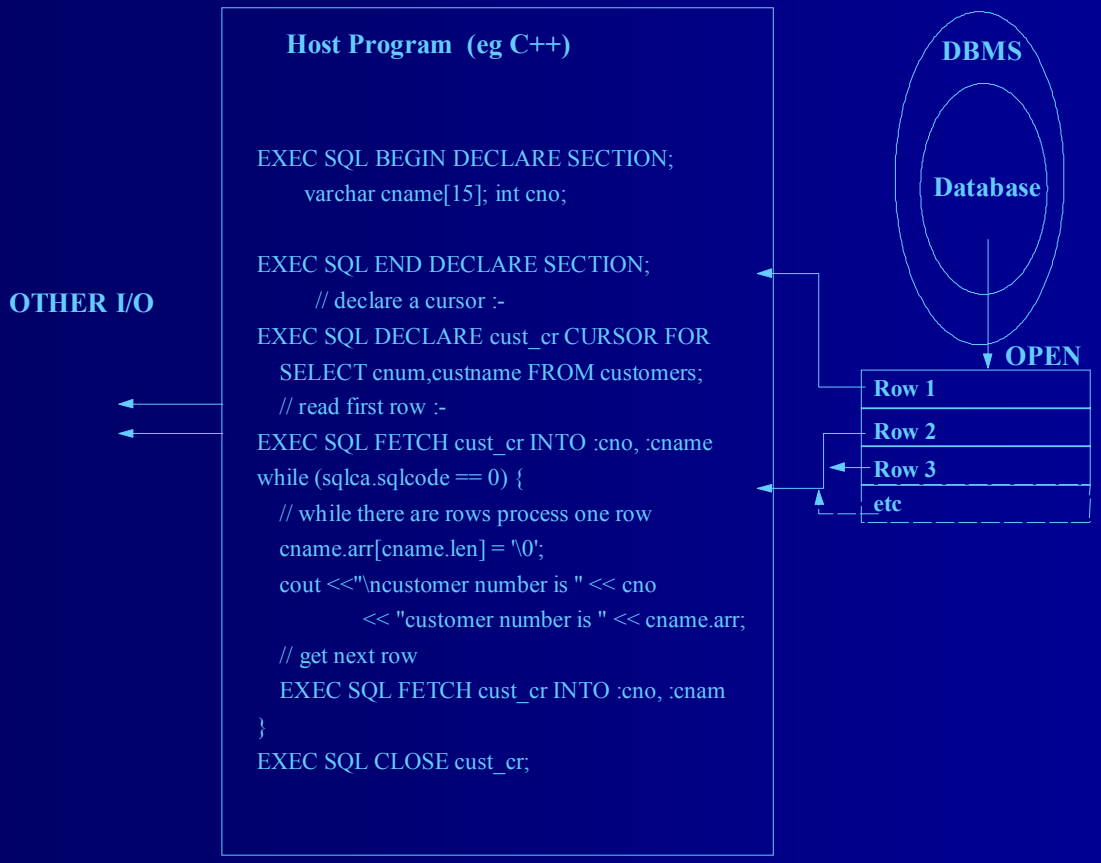
EXEC SQL declare <cursor name> cursor for
<select statement>>
[for update [<of column-list>]]

- the last element of the definition is required only if the cursor is to be used for updates or deletes on the parts of the table involved in the cursor select statement.
- the column-list part is omitted if the rows are to be deleted.

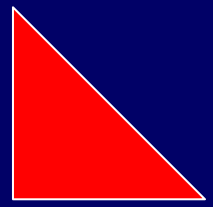
EXEC SQL open <cursor name>

- the view described in the cursor declaration is created
- the cursor is positioned before the first row of the select query result.

Program Example



Summary



- Cursors provide a means of integrating traditional 3GL procedural languages and databases by enabling row-at-a-time access.
- Languages such as Visual Basic and Visual C++ also have build-in statements that enable this type of processing with a dedicated database, like that provided by MS-Access.
- Java has a well-defined interface to enable easy access to databases through a Database Connectivity library - JDBC.
- Unfortunately, there are many 'standards'.