```
 TEST-ASCII SECTION.
*Function: Discover if running Ascii or Ebcdic
*00.
     evaluate space
        when x'20'
           set  is-ascii   to true
        when x'40'
           set  is-ebdic   to true
        when other
           set  is-unknown to true
     end-evaluate
*
     exit section.
*-----------------------------------------------------------------
 TEST-64BIT SECTION.
*Function: Discover if running 32/64 bit
*00.
*    Longer pointers in 64-bit architecture
     if function length (addr) <= 4
        set  is-32-bit to true
     else
        set  is-64-bit to true
     end-if
*
     exit section.
*-----------------------------------------------------------------
 TEST-ENDIAN SECTION.
*00.
*    Number-bytes are shuffled in Big-Little endian
     move 128 to byline
     set  address of byte to address of byline
     if function ord(byte) > 0
        set  is-big-endian-yes to true
     else
        set  is-big-endian-no  to true
     end-if
*
     exit section.
*-----------------------------------------------------------------
 end program CBL_OC_DUMP.
```

5.4 Does OpenCOBOL support any SQL databases?

Yes. There is no embedded SQL in OpenCOBOL in terms of `EXEC` but there are *at least* two usable CALL extensions, the EXEC potential of the Firebird `gpre` and the tried and successful use of Oracle's `procob`. There are as of March 12, 2011 quite a few active developments for easing SQL engine access.

- as reported on `http://opencobol.org` the `procob 10.2 Oracle` preprocessor produces code that compiles and executes just fine with OpenCOBOL 1.1 See note about data sizes and the `binary-size:` configuration below.
- A `libdbi` *generic database access* extension is also available. See cobdbi for full details.
- Efforts toward providing a preprocessor for EXEC are underway.
- Jim Currey's team [1] has kindly posted an ease-of-use MySQL preprocessing layer. http://svn.wp0.org/add1/libraries/mysql4Windows4OpenCobol/
- Rumours of a potential Postgres layer have also been heard.
  - Not a rumour anymore. Work on a nicely complete PostgreSQL binding was posted by gchudyk to opencobol.org
- **AND** as a *thing to watch for*, one of the good people of the OpenCOBOL communinity is writing a layer that converts READ and WRITE verbage to SQL calls `at run time`. More on this as it progresses.

5.4.1 SQLite

There are workable prototypes for access to the SQLite3 shell at

- ocshell.c
- with a sample usage program at sqlscreen.cob
- and supporting documentation at sqlscreen.html

The SQLite extension comes in two flavours; a shell mode discussed above and a direct API interface housed at ocsqlite.c

5.4.2 Oracle procob and binary data sizes

Details of the configuration setting for proper Oracle procob processing.

From Angus on `opencobol.org`

```
Hi

I had some trouble with Oracle procob 10.2 and OpenCobol 1.1 with std=mf.
For PIC S9(2) COMP, procob seems to use 2 bytes, and OpenCobol only one.
It doesn't work well. It comes from the parameter binary-size in the
mf.conf, which seems to tell to opencobol the larger of comp type
I modify to binary-size: 2-4-8 and it works (same as the mvs.conf)
Our application works with Microfocus / Oracle, and microfocus use 2 bytes,
like Oracle. Perhaps because we have the mvs toggle

Except for this thing, opencobol and oracle work like a charm,
on a debian 32bit.

Regards,
Angus
```

5.4.3 Sybase ASE

Another post from http://opencobol.org

```
Preliminary work with Sybase ASE 15 indicates that the output of the Sybase
precompiler cobpre64 and cobpre_r64) is compatible with OpenCOBOL.

Some fiddling with COB_LIBRARY_PATH and COB_LIBS was necessary to get ld (on
AIX 6.1) to see and resolve the externals in the generated code. It's also
important to get the "bitness" of the various pieces in agreement, in the UNIX
case, 64-bit.

Works with UNIX (AIX), also Windows XP and 7 in 32-bit mode.

I'll put together a writeup on the process when time permits.

Jim
```

5.4.4 DB2

Another post from http://opencobol.org

```
Re: AN IDEA FOR SQL SUPPORT IN OPENCOBOL
Embedded SQL with DB2 and the DB2 preprocessor (db2 prep)works fine with OpenCobol, too.

Cheers,
Juergen
```

5.4.5 PostgreSQL Sample

Nowhere near as complete as the binding that Gerald posted to opencobol.org the example below was a starting point.

Note that the PostgreSQL runtime library is `libpq`, *ending in q not g*.

Listing 5.7: OpenCOBOL PostgreSQL connection test

```
OCOBOL*> *************************************************************
      *> Author:    Brian Tiffin
      *> Date:      20091129
      *> Purpose:   PostgreSQL connection test
      *> Tectonics: cobc -x -lpq pgcob.cob
      *> *************************************************************
       identification division.
       program-id. pgcob.

       data division.
       working-storage section.
       01 pgconn usage pointer.
       01 pgres  usage pointer.
       01 resptr usage pointer.
       01 resstr pic x(80) based.
       01 result usage binary-long.
       01 answer pic x(80).

      *> *************************************************************
```

```
       procedure division.
       display "Before connect:" pgconn end-display

       call "PQconnectdb" using
           by reference "dbname = postgres" & x"00"
           returning pgconn
       end-call
       display "After connect: " pgconn end-display

       call "PQstatus" using by value pgconn returning result end-call
       display "Status:        " result end-display

       call "PQuser" using by value pgconn returning resptr end-call

       set address of resstr to resptr
       string resstr delimited by x"00" into answer end-string
       display "User:          " function trim(answer) end-display

       display "call PQexec" end-display
       call "PQexec" using
           by value pgconn
           by reference "select version();" & x"00"
           returning pgres
       end-call
       display pgres end-display

      *> Pull out a result. row 0, field 0 <*
       call "PQgetvalue" using
           by value pgres
           by value 0
           by value 0
           returning resptr
       end-call
       set address of resstr to resptr
       string resstr delimited by x"00" into answer end-string
       display "Version:       " answer end-display

       call "PQfinish" using by value pgconn returning null end-call
       display "After finish:  " pgconn end-display

       call "PQstatus" using by value pgconn returning result end-call
       display "Status:        " result end-display

      *> this will now return garbage  <*
       call "PQuser" using by value pgconn returning resptr end-call
       set address of resstr to resptr
       string resstr delimited by x"00" into answer end-string
       display "User after:    " function trim(answer) end-display

       goback.
```

```
       end program pgcob.
```

Run from a user account that has default PostgreSQL credentials:

```
$ cobc -x -lpq pgcob.cob
$ ./pgcob
Before connect:0x00000000
After connect: 0x086713e8
Status:        +0000000000
User:          brian
call PQexec
0x08671a28
Version:       PostgreSQL 8.3.7 on i486-pc-linux-gnu, compiled by GCC gcc-4.3.real (Debian 4.3.
After finish:  0x086713e8
Status:        +0000000001
User after:    PostgreSQL 8.3.7 on i486-pc-linux-gnu, compiled by GCC gcc-4.3.real (Debian 4.3.
```

Note that `User after` is not the valid answer, shown on purpose. The connection had been closed and the status was correctly reported as non-zero, being an error, but this example continued through as a demonstration.

5.5 Does OpenCOBOL support ISAM?

Yes. The official release used Berkeley DB, but there are also experimental configurations of the compiler that use VBISAM, CISAM, DISAM or other external handlers. See What are the configure options available for building OpenCOBOL?3.2 for more details about these options. The rest of this entry assumes the default Berkeley database.

ISAM is an acronymn for Indexed Sequential Access Method.

OpenCOBOL has fairly full support of all standard specified ISAM compile and runtime semantics.

For example:

Listing 5.8: OpenCOBOL ISAM sample

```
OCOBOL >>SOURCE FORMAT IS FIXED
      *> ************************************************************
      *><* ================
      *><* indexing example
      *><* ================
      *><* :Author:    Brian Tiffin
      *><* :Date:      17-Feb-2009
      *><* :Purpose:   Fun with Indexed IO routines
      *><* :Tectonics: cobc -x indexing.cob
      *> ************************************************************
       identification division.
       program-id. indexing.

       environment division.
       configuration section.

       input-output section.
```