Fujitsu Software BS2000 ESQL-COBOL
Version 3.0D
December 2013

Readme file

Introduction

Description of modifications/extensions to ESQL-COBOL V3.0A and ESQL-COBOL for SESAM/SQL-Server.

Addendum for User-Guide

ESQL-COBOL V3.0A
ESQL-COBOL for SESAM/SQL-Server
Edition November 2006
Reference number: U22424-J-Z125-3

The following changes are described in this readme file:

This is a compilation of the functional extensions of the ESQL-COBOL PreCompiler for SESAM/SQL-Server as far as free reference format is concerned.

Extension of Chapter 2.6 “SQL comments in an ESQL-COBOL program”.

Keywords:
COBOL2000, SQL, SESAM, Free Reference Format, c-style comments.

This and other current Readme files are available online at https://bs2manuals.ts.fujitsu.com/index.

1 Free Reference Format

COBOL 2000 now has the feature of accepting COBOL source text in free reference format. That means that the old traditional restrictions of line length and different areas on the source line do no longer apply whenever free format is selected. Earlier Versions of the ESQL-PreCompiler could not cope with this format.

The existing functionality of the precompiler is not affected, expecially, COBOL and SQL text in fixed reference format are treated exactly as they were before.

1.1 Scope of the implementation

The ability of the precompiler to accept free reference format is not being switchen on or off by external control but only by the source format directives, as defined by the COBOL Standard. Expecially, whenever starting a new precompilation, the precompiler assumes fixed reference format.

This mimics the behaviour of the BS2000 COBOL compiler.

But there is more to this: Also, the embedded SQL-Text is treated according to the free format rules, although the definition of free format only applies for COBOL source text.
Details as follows.

1.1.1 The linelength

Whenever in free format mode, the precompiler accepts COBOL as well as ESQL text on column positions 1 through and including 248.

The thusly generated host text also contains lines up to this length.

The COBOL Standard, however, mentions a line length of up to 256. Both, the COBOL compiler and the ESQL precompiler, are likely to accept this line length in a later version.

1.1.2 Generated comments

The precompiler generates comments in the following situations:

- De-commentarized SQL-Statements
- Error flags
- General information blocks inside the host text.

1.1.2.1 De-commentarized SQL-Statements

In fixed format, the de-commentarized SQL-Statements look the same way as they did before. In free format, they are turned into free format comments, that is, the two characters "*>" are placed onto column position 1 and 2.

If the SQL-statement actually used column position 1 and 2, those characters are overwritten!

1.1.2.2 Error flags

Error messages are made out of two parts: A line ruler which indicates the column position of an error with "...-----#------...", and a textual error message.

The text of the error message is the same in both formats and is made as a valid comment for both formats. This is done by the character sequence "*>" on column position 7 and 8.

In free format, the line ruler is prefixed with "*>" on column position 1 and 2. That, however, means that a "#" Indicator on column position 1 or 2 is lost! (For this reason, it is most advisable not to make any errors which would be flagged in column position 1 or 2.) The error messages, however, are generated in any case.

1.1.2.3 General information blocks inside the host text

The information block which are inserted into the host-text are comments valid for both formats.

1.1.3 Generated calls in COBOL

For SQL Statements, a COBOL-call-statement is being generated. The parameters of this call may use the full available line length, if necessary. If a line is not sufficient, a newline is inserted.

In the classical fixed format, a newline could be made inside a word with generation of a continuation line. In free format, however, a newline is generated only in place of a whitespace. The reason behind this strategy is that the classical continuation line inside a word is an obsolete feature of COBOL and may be discontinued in the future.

There is no way of enforcing a newline in place of a white space whilst processing fixed format. This is not necessary because you can put any ESQL-statement into a free-fixed format bracketing.

1.1.4 Dependencies on dialect

There are no dependencies on dialect. The free format capability is always available, no matter which SDF-option is applied:

```
... ,ESQL-DIALECT = *ALL-FEATURES(...)

or

... ,ESQL-DIALECT = *ISO(...)

or

... ,ESQL-DIALECT = *OLD(...)
```

1.1.5 SQLCA

The SQLCA include element which is part of the shipment can be used in both formats alike.

1.3 Restrictions of the implementation

A "EXEC SQL" must not follow a "END-EXEC" on the same line.

Another restriction is: the precompiler does not care whether it is used with COBOL2000 or COBOL85. Now, this free format, the precompiler passes a lot of language elements which would be an annoyance to the COBOL85 compiler.

The precompiler cannot possibly know whether it sees free format source text without a preceeding free format source directive. In that case, manipulation of column positions 1 through 7 and 80 and more causes generation of a very confusing host text. In rare cases, the resulting host text may even be a legal one, although not corresponding to the original source text.

1.4 Glossary

* Free Reference Format

german "Freies Referenz Format", often short "Free Format": The feature which this precomiler now accepts. Actually, the word references not only COBOL but SQL as well.

* Fixed Reference Format

german "Festes Referenz Format", short "Fixed Format", or "classica COBOL Format": the age-old way to write COBOL programs.

2 Extension of Chapter 2.6 SQL comments in an ESQL-COBOL program

As of ESQL-COBOL V3.0D C-style comments can be used in SQL statements. There are parenthesized comments which start with /* and end with */ and which can also be nested.

The support of C-style comments /* */ was implemented with some restrictions:

- C-style comments are not allowed in the SQL statements WHENEVER and INCLUDE and in DECLARE SECTION.
- END-EXEC cannot be commented-out by C-style comments within an SQL statement. It is always accepted by the precompiler as an end of SQL statement, even if it is preceded by an opening comment bracket /*.