COBOL SQL アクセス
言語説明書

はしがき

本書は，JIS の水準 2 に準拠する SQL データ操作言語(SQL-DML)を COBOL プログラム中に埋め込む形式でサポートする SQL プリコンパイラの文法について説明しています。

本書の構成

本書の構成について説明します。

本書は，3 つの章で構成しています。それぞれの章の内容は，次のとおりです。

「第 1 章　SQL 概説（1 ページ）」

SQL 機能の製品概要を説明します。

「第 2 章　SQL 文で使用する基本要素（2 ページ）」

SQL 機能で使用するＳＱＬ文の基本要素を説明します。

「第 3 章　SQL 文の言語規則（32 ページ）」

ＳＱＬ文の言語規則を説明します。

説明書の構成

COBOL SQL アクセスをご使用していただくために各種の説明書を用意しています。

説明書名	記述している内容
COBOL SQL アクセス 言語説明書	COBOL ソースに埋め込む形式でサポートする SQL データ操作言語の言語仕様（書き方や規則）について説明しています。
COBOL SQL アクセス プログラミングの手引	埋込み SQL COBOL ソースを記述して，プログラミングを行うために，SQL 文の具体的な使い方を説明しています。
COBOL SQL アクセス ユーザーズガイド	作成した埋込み SQL COBOL ソースをプリコンパイルして SQL 展開済み COBOL ソースを生成する方法や生成した実行モジュールを実行する際に，埋込み SQL COBOL ソースの CONNECT 文で指定したサーバ名と ODBC ドライバで設定するデータソース名を関連付ける方法について説明しています。

関連製品の説明書

関連製品の説明書として次のものがあります。

説明書名	記述している内容
COBOL 言語説明書	COBOL の言語仕様について説明しています。
COBOL プログラミングの手引	COBOL のプログラミング技法を説明しています。

説明書名	記述している内容
COBOL ユーザーズガイド	COBOL を使ったアプリケーションの開発方法について説明しています。
COBOL 開発環境 操作ガイド	COBOL プログラムプロダクトの機能と操作方法について説明しています。

ご注意

1. 本書の内容の一部または全部を無断転載することは禁止されています。
2. 本書の内容に関しては将来予告なしに変更することがあります。
3. 本書は内容について万全を期して作成いたしましたが，万一ご不審な点や誤り，記載もれなどお気付きのことがありましたら，ご連絡ください。
4. 運用した結果の影響については，(3)項にかかわらず責任を負いかねますのでご了承ください。

商標情報

- Microsoft，Windows は，米国 Microsoft Corporation の米国およびその他の国における登録商標または商標です
- Oracle と Java は，Oracle Corporation およびその子会社，関連会社の米国およびその他の国における登録商標です。文中の社名，商品名等は各社の商標または登録商標である場合があります。
- そのほかの会社名および商標名は各社の商標または登録商標です。なお，本文中では TM や®は明記しておりません。

輸出する際の注意事項

本製品 (ソフトウェア)は日本国内仕様であり，外国の規格等には準拠しておりません。

本製品は日本国外で使用された場合，当社は一切責任を負いかねます。また，当社は本製品に関して海外での保守サービスおよび技術サポート等は行っておりません。

著作権

本書の内容は，日本電気株式会社が開示している情報のすべてが掲載されていない場合，またはほかの方法で開示された情報とは異なった表現の仕方をしている場合があります。また，予告なしに内容が変更または廃止される場合がありますので，あらかじめご承知おきください。

本文中の記号／略称

本書で使用する記号や略称について説明します。

形式で用いている記法

1. 英字の語と日本語の語

   英字の語は予約語を表しています。

   日本語の語は，その項または他の項で記述されている形式を表しています。

2. 角かっこと中かっこ

   a. 角かっこ[]

      角かっこ[]で囲んである部分は書くか省くかを利用者が選択します。

      角かっこ[]内に縦線|で分割した複数の形式がある場合，それらのうちの１個を指定するか，またはすべて省くかを選択できます。

      角かっこ[]内で下線がついている形式は，[]内を省いたときに暗黙的に指定される形式です。

   b. 中かっこ{}

      中かっこ{}に縦線|で分割した複数の形式がある場合，複数の形式のうち，必ず１個の形式を利用者が選択します。

3. 反復記号

   反復記号“…”の意味は以下のとおりです。

   []…は角かっこ[]内における形式の 0 回以上の繰り返しです。

   {}…は中かっこ{}内における形式の１回以上の繰り返しです。

本書の中で使用する略称

略称	意味
SQL 機能	COBOL SQL アクセスを表します

略称	意味
SQL プリコンパイラ	COBOL SQL アクセスのプリコンパイラを表します

目次

第 1 章
SQL 概説

SQL は，1987 年 11 月に日本工業規格(JIS)により制定されたリレーショナルデータベース操作言語(JIS X3005-1987)である。

SQL では，個々のレコードがデータベース上でどのように格納されているかを意識したり，アクセス方法を指定する必要はなく，単に結果として得たい項目とそれが格納されている場所(表)を指定するだけで，容易に検索を行うことが可能である。

SQL を用いると以下のデータ操作が可能である。

- 条件を満足するレコードを選択する(SELECT)
- 表にレコードを追加する(INSERT)
- 表からレコードを削除する(DELETE)
- レコードの列の値を更新する(UPDATE)

1.1　SQL の定義

SQL は，JIS の水準 2 に準拠する SQL データ操作言語(SQL-DML)を COBOL プログラム中に埋め込む形式でサポートする。対象となるホストプログラム言語は COBOL である。

本 SQL 機能でサポートする COBOL 言語は COBOL85 の規格の範囲である。また，接続可能なデータベースは ODBC 3.5 および、COBOL 製品が対象とする OS に対応している ODBC ドライバおよびデータベースを対象とする。

ただし，入れ子プログラム機能には，対応していない。

第 2 章
SQL 文で使用する基本要素

この章では，SQL 文中で使用される基本要素について説明する。

基本要素には，次のものがある。

2.1　文字

SQL では以下に示す文字が使用できる。

1. 数字

   ０ １ ２ ３ ４ ５ ６ ７ ８ ９

2. 英小文字

   ａ ｂ ｃ ｄ ｅ ｆ ｇ ｈ ｉ ｊ ｋ ｌ ｍ ｎ ｏ ｐ ｑ ｒ ｓ ｔ ｕ ｖ ｗ ｘ ｙ ｚ

3. 英大文字

   Ａ Ｂ Ｃ Ｄ Ｅ Ｆ Ｇ Ｈ Ｉ Ｊ Ｋ Ｌ Ｍ Ｎ Ｏ Ｐ Ｑ Ｒ Ｓ Ｔ Ｕ Ｖ Ｗ Ｘ Ｙ Ｚ

4. カナ文字

   ｱ ｲ ｳ ｴ ｵ ｶ ｷ ｸ ｹ ｺ ｻ ｼ ｽ ｾ ｿ ﾀ ﾁ ﾂ ﾃ ﾄ ﾅ ﾆ ﾇ ﾈ ﾉ
   ﾊ ﾋ ﾌ ﾍ ﾎ ﾏ ﾐ ﾑ ﾒ ﾓ ﾔ ﾕ ﾖ ﾗ ﾘ ﾙ ﾚ ﾛ ﾜ ﾝ
   ｧ ｨ ｩ ｪ ｫ ｯ ｬ ｭ ｮ ﾟ（半濁点） ﾞ（濁点） ｰ（長音）

5. 特殊文字

+ - * / ( ) % _ ' < > = : , .空白文字

6. 日本語文字

   日本語文字集合中の文字

2.2　表記法

ここでは，SQL を記述する際に従わなければならない規則について説明する。

2.2.1　識別子

識別子はユーザが定義する名前である。

[形式]

- 書き方 1

```
{ 英字 | カナ文字 }[ 英字 | 数字 | カナ文字 | _（下線） ]…
```

- 書き方 2(日本語識別子)

```
{日本語文字 }…
```

- 書き方 3

```
"[ 英字 | 数字 | カナ文字 | _（下線） | -（ハイフン） ]…"
```

[規則]

1. 書き方１で'_'は２個以上続いてはならない。
2. 書き方２の識別子は，日本語文字の空白を含んではならない。
3. 書き方１の識別子の長さは，それを含む文字の個数であり，18 文字以下でなければならない。
4. 書き方２の識別子の長さは，それを含む日本語文字の個数であり，9 文字以下でなければならない。
5. 書き方３の識別子の長さは，引用符(")で囲まれた文字の個数であり，18 文字以下でなければならない。
6. 書き方１および書き方２で定義された識別子は，予約語と一致してはならない。
7. 識別子中の英大文字，英小文字は同一視されない。すなわち，たとえば ABC，Abc，AbC，abc はいずれも別の識別子とみなされる。

2.2.2　予約語

予約語は，特定の意味を持った文字列である。

SQL92 の予約語の一覧は，以下による。

ABSOLUTE	ACTION	ADD
ALL	ALLOCATE	ALTER
AND	ANY	ARE
AS	ASC	ASSERTION
AT	AUTHORIZATION	AVG
BEGIN	BETWEEN	BIT
BIT_LENGTH	BOTH	BY
CASCADE	CASCADED	CASE
CAST	CATALOG	CHAR
CHAR_LENGTH	CHARACTER	CHARACTER_LENGTH
CHECK	CLOSE	COALESCE
COLLATE	COLLATION	COLUMN
COMMIT	CONNECT	CONNECTION
CONSTRAINT	CONSTRAINTS	CONTINUE
CONVERT	CORRESPONDING	COUNT
CREATE	CROSS	CURRENT
CURRENT_DATE	CURRENT_TIME	CURRENT_TIMESTAMP
CURRENT_USER	CURSOR	DATE
DAY	DEALLOCATE	DEC
DECIMAL	DECLARE	DEFAULT
DEFERRABLE	DEFERRED	DELETE
DESC	DESCRIBE	DESCRIPTOR
DIAGNOSTICS	DICTIONARY	DISCONNECT
DISPLACEMENT	DISTINCT	DOMAIN
DOUBLE	DROP	ELSE
END	END-EXEC	ESCAPE
EXCEPT	EXCEPTION	EXEC
EXECUTE	EXISTS	EXTERNAL
EXTRACT	FALSE	FETCH
FIRST	FLOAT	FOR
FOREIGN	FOUND	FROM
FULL	GET	GLOBAL
GO	GOTO	GRANT
GROUP	HAVING	HOUR
IDENTITY	IGNORE	IMMEDIATE
IN	INCLUDE	INDEX

INDICATOR	INITIALLY	INNER
INPUT	INSENSITIVE	INSERT
INT	INTEGER	INTERSECT
INTERVAL	INTO	IS
ISOLATION	JOIN	KEY
LANGUAGE	LAST	LEFT
LEVEL	LIKE	LOCAL
LOWER	MATCH	MAX
MIN	MINUTE	MODULE
MONTH	NAMES	NATIONAL
NATURAL	NCHAR	NEXT
NO	NONE	NOT
NULL	NULLIF	NUMERIC
OCTET_LENGTH	OF	OFF
ON	ONLY	OPEN
OPTION	OR	ORDER
OUTER	OUTPUT	OVERLAPS
PAD	PARTIAL	POSITION
PRECISION	PREPARE	PRESERVE
PRIMARY	PRIOR	PRIVILEGES
PROCEDURE	PUBLIC	READ
REAL	REFERENCES	RELATIVE
RESTRICT	REVOKE	RIGHT
ROLLBACK	ROWS	SCHEMA
SCROLL	SECOND	SECTION
SELECT	SEQUENCE	SESSION
SESSION_USER	SET	SIZE
SMALLINT	SOME	SPACE
SQL	SQLCA	SQLCODE
SQLERROR	SQLSTATE	SQLWARNING
SUBSTRING	SUM	SYSTEM
SYSTEM_USER	TABLE	TEMPORARY
THEN	TIME	TIMESTAMP
TIMEZONE_HOUR	TIMEZONE_MINUTE	TO
TRAILING	TRANSACTION	TRANSLATE
TRANSLATION	TRUE	UNION
UNIQUE	UNKNOWN	UPDATE
UPPER	USAGE	USER
USING	VALUE	VALUES
VARCHAR	VARYING	VIEW

WHEN	WHENEVER	WHERE
WITH	WORK	WRITE
YEAR		

本 SQL 機能では，以下の文字列も予約語として扱う。

AUTO_INCREMENT	AUXILIARY	BIGINT
BINARY	BOOL	BOOLEAN
CLASS	CLASTER	DATABASE
DATABASES	DATETIME	DIV
ELSEIF	EXPLAIN	FLUSH
IDENTIFIED	IF	LIMIT
MOD	NESTING	OFFSET
PARTITION	PASSWORD	RANGE
REFERENCE	RENAME	START
TEXT	TRIM	TRUNCATE
UNSIGNED	VARBINARY	

2.2.3　注釈

注釈は，SQL 文中の区切りを入れてもかまわないところに任意に挿入することができ，行末まで注釈となる。また，注釈は SQL 文の意味に影響を及ぼさない。 ただし，ホスト変数宣言中で使用するとエラーとなる。

[形式]

```
--[文字]…
```

2.3　名前

ここでは，SQL で用いられる各種の名前について説明する。

各種の名前は，同じ名前があってもよい。たとえば，ABC という表名と ABC という列名があってもよい。

2.3.1　表名

[形式]

```
表識別子
```

表識別子の形式は識別子と同じである。

[説明]

1. 表名は，名前付きの表を指定する。

2.3.2　相関名

[形式]

```
識別子
```

[説明]

1. 相関名は FROM 句中の表を識別するための文字列である。相関名の有効範囲は，SELECT 文，副問合せまたは問合せ指定のいずれかである。有効範囲は入れ子構造になってもよい。異なった有効範囲では，異なった複数の表または同じ表に対して同じ相関名を指定することができる。
2. FROM 句で表名の後ろに相関名を指定すると，その表は相関名によってのみ参照可能となる。

   主に，同じ表名を FROM 句で複数回指定するような場合に使用する。

2.3.3　列名

[形式]

```
識別子
```

[説明]

1. 列名は，名前付きの列を指定する。

2.3.4　カーソル名

[形式]

```
識別子
```

[説明]

1. カーソル名は，カーソルを指定する。

2.3.5　埋込み変数名

[形式]

```
:ホスト変数名－１[.ホスト変数名－２]…
```

[説明]

1. 埋込み変数名は，ホストプログラム中の変数を指定する。
2. ホスト変数名－１，ホスト変数名－２はホスト言語中で定義している変数の名前である。その形式はホスト言語の変数名の書き方に準拠しなければならない。
3. 埋込み変数名のすぐ後に２項演算子の‘－’を書く場合は，空白を入れなければならない。
4. ホスト変数名－２を指定した場合，ホスト変数名－１はホスト変数名－２を直接または間接的に含む集団項目でなければならない。 さらに，ホスト変数－２を複数指定すときは，レベル番号の小さい順に指定しなけれなばらない。このとき，最右端のホスト変数－２は集団項目であってはならない。
5. ホスト変数名－２を省略した場合，ホスト変数名－１は集団項目であってはならない。
6. ホスト変数名－２を指定する場合，ホスト変数名－１に続けて空白を入れずピリオド(.)を記述し，さらに空白を入れずにホスト変数名－２を指定する。

2.3.6　SQL 文変数

[形式]

```
:SQL 文変数
```

[説明]

1. SQL 文変数は，SQL 文を格納する変数である。
2. SQL 文変数はホスト変数である。ただし，データ属性は文字列か可変長文字列でなければならない。

2.3.7　SQL 文識別子

[形式]

```
識別子
```

[説明]

1. 準備した SQL 文を識別する識別子である。

2.4　データ属性

ホスト変数に指定可能なデータ属性について記述する。

[説明]

属性	データベースのデータ型	COBOL 記述[*1]	説明
集団項目		レベル番号(01～47) ホスト変数名 .	
真数	SMALLINT	レベル番号(01～48,77) ホスト変数名 COMP-1.	
	INTEGER	レベル番号(01～48,77) ホスト変数名 COMP-2.	
	NUMERIC	**レベル番号**(01～48,77) **ホスト変数名** PIC {9(n) | 9(n)V9(m) | V9(m) | S9(n) | S9(n)V9(m) | SV9(m) } [[USAGE [IS]] DISPLAY] [[SIGN [IS]] { LEADING | TRAILING} [SEPARATE [CHARACTER]]].	ｎとｍは正の整数で n+m が 18 を越えてはならない。
	DECIMAL	**レベル番号**(01～48,77) **ホスト変数名** PIC {9(n) | 9(n)V9(m) | V9(m) | S9(n) | S9(n)V9(m) | SV9(m) } [USAGE [IS]] {COMP-3 | PACKED-DECIMAL}.	ｎとｍは正の整数で n+m が 18 を越えてはならない
文字列型	CHAR	レベル番号(01～48,77) ホスト変数名 PIC X(n) .	ｎは正の整数でその値は 1 ～ 65535 を越えてはならない。
日本語文字列型	CHAR	レベル番号(01～48,77) ホスト変数名 PIC N(n).	ｎは正の整数でその値は 1 ～ 32767 を越えてはならない。

属性	データベースのデータ型	COBOL 記述[*1]	説明
可変長文字列型	VARCHAR	**レベル番号**(01～48) **ホスト変数名**. 49　**ホスト変数名－１** COMP-2. 49　**ホスト変数名－２** PIC X(n).	n は正の整数でその値は 1 ～ 65535 を越えてはならない。
可変長日本語文字列型	VARCHAR	**レベル番号**(01～48) **ホスト変数名**. 49　**ホスト変数名－１** COMP-2. 49　**ホスト変数名－２** PIC N(n).	n は正の整数でその値は 1 ～ 32767 を越えてはならない。

*1　正確な記述方法は，後述の「2.4.1　データ属性の詳細（10 ページ）」を参照のこと

文字列型／日本語文字列型／可変長文字列型／可変長日本語文字列型の内部コードは COBOL コンパイラのオプションによって決定されます。詳細は『COBOL プログラミングの手引』を参照してください。

データの内部表現については，『COBOL SQL アクセスプログラミングの手引』の「2.1.1 ホスト変数」を参照してください。

COBOL SQL アクセスで使用している USAGE 句については，『COBOL 言語説明書』を参照してください。

2.4.1　データ属性の詳細

SQL 機能で扱うデータ属性の詳細を記載する。

1. 集団項目

   [形式]

   レベル番号(01～47) **ホスト変数名**.

   [説明]

   a. レベル番号は，01～47 が記述できる。

   b. 必ず従属項目が必要である。

      かつ，集団項目および従属項目は以下の規則に従う必要がある。

      i. 1 から 9 までのレベル番号はそれぞれ 01 から 09 と記述してもよい。

      ii. 集団項目に従属する項目のレベル番号は，集団項目のレベル番号よりも大きくなければならない。

      iii. レベル番号は，連続番号である必要はない。

      iv. 集団項目に直接的に従属する項目に用いるレベル番号は，すべて等しくなければならない。

      v. 集団項目に従属する項目は，記述可能はホスト変数のうちいずれの定義を含んでもよい。

vi. 集団項目には，VALUE 句など，いずれの句も指定することはできない。

vii. 集団項目は，SQL 展開済み COBOL ソース内で複数のホスト変数をまとめて扱うために使用することができる。データベースに対応するデータ型がなく，SQL 文中では，従属するホスト変数を修飾するために使用する。

2. 真数

a. 単精度固定 2 進数

[形式]

```
レベル番号(01～48,77) ホスト変数名　[USAGE [IS]] {COMP-1 | COMPUTATIONAL-1}.
```

[説明]

USAGE(用途)句の構文規則は，COBOL 構文規則に従う。

基本項目に指定することができる句のうち，OCCURS 句，REDEFINES 句は記述することはできない。

b. 倍精度固定 2 進数

[形式]

```
レベル番号(01～48,77) ホスト変数名　[USAGE [IS]] {COMP-2 | COMPUTATIONAL-2}.
```

[説明]

USAGE(用途)句の構文規則は，COBOL 構文規則に従う。

基本項目に指定することができる句のうち，OCCURS 句，REDEFINES 句は記述することはできない。

c. 外部１０進数

[形式]

```
レベル番号(01～48,77) ホスト変数名
      {PICTURE | PIC} [IS] {9(n) | 9(n)V9(m) | V9(m) | S9(n) | S9(n)V9(m) | SV9(m)}
      [[USAGE [IS]]DISPLAY]
      [[SIGN [IS]]{ LEADING | TRAILING} [SEPARATE [CHARACTER]]].
```

[説明]

PICTURE(形式)句および USAGE(用途)句，SIGN(符号)句の構文規則は，COBOL 構文規則に従う。

基本項目に指定することができる句のうち，OCCURS 句，REDEFINES 句は記述することはできない。

d. 内部１０進数

[形式]

```
レベル番号(01～48,77) ホスト変数名
       {PICTURE | PIC} [IS] {9(n) | 9(n)V9(m) | V9(m) | S9(n) | S
9(n)V9(m) | SV9(m)}
      [USAGE [IS]]  {COMP-3 | COMPUTATIONAL-3 | PACKED-DECIMAL}.
```

[説明]

PICTURE(形式)句および USAGE(用途)句の構文規則は，COBOL 構文規則に従う。

基本項目に指定することができる句のうち，OCCURS 句，REDEFINES 句は記述することはできない。

3. 文字列型

[形式]

```
レベル番号(01～48,77) ホスト変数名
      {PICTURE | PIC} [IS] X(n)
      [[USAGE [IS]] DISPLAY].
```

[説明]

PICTURE(形式)句および USAGE(用途)句の構文規則は，COBOL 構文規則に従う。

基本項目に指定することができる句のうち，OCCURS 句，REDEFINES 句は記述することはできない。

4. 日本語文字列型

[形式]

```
レベル番号(01～48,77) ホスト変数名
      {PICTURE | PIC} [IS] N(n)
      [[USAGE [IS]] DISPLAY].
```

[説明]

PICTURE(形式)句および USAGE(用途)句の構文規則は，COBOL 構文規則に従う。

基本項目に指定することができる句のうち，OCCURS 句，REDEFINES 句は記述することはできない。

5. 可変長文字列型

[形式]

```
レベル番号(01～48) ホスト変数名－１.
       49 ホスト変数名－２[USAGE [IS]]  {COMP-2 | COMPUTATIONAL-2}.
       49 ホスト変数名－３ {PICTURE | PIC} [IS] X(n) [[USAGE [IS]] DISPLA
Y].
```

[説明]

a. ホスト変数名－１のレベル番号は 01～48，ホスト変数名－２およびホスト変数名－３のレベル番号は 49 のみ記述できる。

b. PICTURE(形式)句および USAGE(用途)句の構文規則は，COBOL 構文規則に従う。

c. ホスト変数名－２は，ホスト変数名－３に設定されている有効な文字数(=バイト数)を示す。

d. 基本項目に指定することができる句のうち，OCCURS 句，REDEFINES 句は記述することはできない。

6. 可変長日本語文字列型

[形式]

```
レベル番号(01～48)　ホスト変数名－１．
        49　ホスト変数名－２[USAGE [IS]]　{COMP-2 | COMPUTATIONAL-2}.
        49　ホスト変数名－３　{PICTURE | PIC} [IS] N(n) [[USAGE [IS]] DISPLA
Y].
```

[説明]

a. ホスト変数名－１のレベル番号は 01～48，ホスト変数名－２およびホスト変数名－３のレベル番号は 49 のみ記述できる。

b. PICTURE(形式)句および USAGE(用途)句の構文規則は，COBOL 構文規則に従う。

c. ホスト変数名－２は，ホスト変数名－３に設定されている有効な日本語文字数(=バイト数ではない)を示す。

d. 基本項目に指定することができる句のうち，OCCURS 句，REDEFINES 句は記述することはできない。

ホスト変数として，上記以外の記述はできない。

2.5　値の指定方法

ここでは，値を指定する値指定すなわち「2.5.1　変数指定（13 ページ）」および「2.5.2　定数（14 ページ）」について示す。

2.5.1　変数指定

[形式]

```
埋込み変数名－１[[INDICATOR]埋込み変数名－２]
```

[説明]

1. 埋込み変数名－２を標識変数という。埋込み変数名－２のデータ型は符号付真数属性でなければならない。また想定小数点を含んではならない。
2. 変数指定が，埋込み変数名－２を含み，それによって指定される変数の値が負ならば，変数指定によって指定される値は，NULL 値である。そうでなければ，変数指定によって指定される値は，埋込み変数名－１によって指定される変数の値である。

2.5.2　定数

ここでは，データの値を表す定数すなわち「2.5.2.1　文字列定数指定（14 ページ）」，「2.5.2.2　日本語文字列定数指定（14 ページ）」および「2.5.2.3　真数定数指定（15 ページ）」について示す

2.5.2.1　文字列定数指定

[形式]

```
'文字[文字]…'
```

[説明]

1. 文字列定数中の２つの連続するアポストロフィ(')は単一のアポストロフィを表し，その長さは１である。
2. 文字列定数のデータ属性は，文字列である。文字列定数の長さは，それを含む文字表現の個数である。
3. 文字列定数の値は，それを含む文字の並びである。
4. 文字列定数の長さは 256 以下でなくてはならない。

2.5.2.2　日本語文字列定数指定

[形式]

```
'日本語文字[日本語文字]…'
```

[説明]

1. 日本語文字列定数のデータ属性は，日本語文字列である。日本語文字列定数の長さはそれを含む日本語文字の個数である。

2. 日本語文字列定数の値は，それを含む日本語文字の並びである。
3. 日本語文字列定数の長さは 128 以下でなければならない。

2.5.2.3　真数定数指定

[形式]

```
[+ | -]{ 整数 | 整数.整数 | 整数. | .整数 }
```

整数

```
{数字}…
```

[説明]

1. 真数定数のデータ属性は，真数である。真数定数の精度は，それを含む数字の数である。真数定数の位取りは，小数点より右側の数字の個数である。小数点を含まない真数定数の位取りは 0 である。
2. 真数定数の値は通常の数学的解釈による値である。
3. 真数定数中の数字の数は 18 以下でなければならない。

2.6　列指定

ここでは，ある表中における列の参照方法について説明する。

[形式]

```
[表名. | 相関名. ]列名
```

[説明]

1. 列指定中の表名または相関名を修飾子という。
2. 列指定は，名前付きの列を参照する。
3. 列指定の修飾子として，次のいずれかを指定する。
    a. 列指定が修飾子を含むならば, 列指定は修飾子に等しい 1 つ以上の表名または相関名の有効範囲内に現れなければならない。複数のそのような表名または相関名があるならば, 最も局所的な有効範囲をもつものを指定する。指定される表名または相関名に関する表は, 列名が列指定の列名と同じである列を含まなければならない。

b. 列指定が修飾子を含まないならば, 列指定は１つ以上の表名または相関名の有効範囲になければならず, 列名が列指定の列名と同じである列を含むような表名および相関名を示すものが最も局所的な有効範囲内に，ただ１つなければならない。その表名または相関名を暗に指定する。

4. 列指定のデータ属性は暗にまたは明に指定する修飾子に対応する表の列名で指定した列名である列のデータ属性である。
5. 列指定は暗にまたは明に指定する修飾子に対応する表の列名を指定した列名である列を参照する。

2.7　集合関数指定

集合関数指定は，引数への関数の適用によって導出される値を指定する。

[形式]

- 書き方 1

```
COUNT (*)
```

- 書き方 2

```
{ AVG | MAX | MIN | SUM | COUNT }(DISTINCT 列指定)
```

- 書き方 3

```
{ AVG | MAX | MIN | SUM }([ALL] 値式)
```

[規則]

1. 集合関数指定の引数は, 副問合せ, 問合せ指定で指定した表またはグループ表のグループである。
2. 書き方２の列指定と書き方３の値式中の各列指定は，明確にその引数の列を参照しなければならない。
3. 書き方３の値式は集合関数指定の引数または，引数の入力の列を参照する列指定を含まなければならず，集合関数指定を含んではならない。

[説明]

1. “SUM”または“AVG を指定した場合，引数のデータ属性は文字列または日本語文字列であってはならない。

2. 書き方２の集合関数指定の引数は，値の集合である。集合は，列指定によって参照している引数の列からすべての NULL 値およびすべての冗長な重複値を取り除くことによって導出する。
3. 書き方３の集合関数指定の引数は，値の集合である。集合は値式を引数の各行に適用した結果からすべての NULL 値を取り除くことによって導出する。“ALL”は指定しても省略しても，集合関数指定の意味に影響を与えない。
4. 集合関数指定の結果は，次のとおりである。
    a. “COUNT”を指定すると，結果は引数の数である。
    b. “AVG”，“MAX”，“MIN”または“SUM”を指定したとき，引数が空ならば，結果は NULL 値である。
    c. “MAX”または“MIN”を指定すると，結果は引数中の値の最大値または最小値となる。このときの大小関係は，比較述語で指定される比較規則を用いて決定する。
    d. “SUM”を指定すると，結果は引数中の値の合計とする。合計は結果のデータ属性の値域内になければならない。
    e. “AVG”を指定すると，結果は引数の値の平均値である。引数中の値の合計は結果のデータ属性の値域内になければならない。

2.8　値式

値式は，四則演算を行う算術式を指定する。

[形式]

```
[ + | - ] 一次子 [ { + | - | * | / } [ + | - ] 一次子 ]…
```

一次子の形式

```
{値指定 | 列指定 | 集合関数指定 | （値式） }
```

[規則]

1. 集合関数指定の書き方２を含む値式は，どの二項演算子も含んではならない。
2. 演算子“＋”，“－”に続く最初の文字は，正または負の符号であってはならない。
3. 一次子のデータ属性が文字列または日本語文字列ならば，値式はどの演算子も含んではならない。

[説明]

1. いずれかの“一次子”の値が NULL 値ならば値式の結果は NULL 値である。
2. 演算子を指定しない場合，値式の結果は指定される一次子の値である。
3. 値式を表の行に適用するとき，その表の列への各参照は，その行のその列の値への参照である。
4. 単項演算子“＋”と“－”は，それぞれ単項プラスと単項マイナスを指定する。単項プラスは，そのオペランドの値を変えない。単項マイナスは，そのオペランドの値の正負を反転する。
5. 二項演算子“＋”，“－”，“＊”，“／”は，それぞれ加算，減算，乗算，除算を指定する。除数は，0 であってはならない。
6. かっこの中の式は，最初に評価し，評価の順序がかっこによって指定しない場合は，単項演算子を乗除算より先に適用し，乗除算を加減算より先に適用し，同順位の演算子を左から右に適用する。

2.9　述語

述語は，“真”または“偽”の真偽値を与える条件を指定する。

[形式]

```
{ 比較述語 | BETWEEN 述語 | IN 述語 | LIKE 述語 | NULL 述語 | 限定述語 | EXISTS 述語 }
```

[規則]

なし

[説明]

1. 述語の結果は，それを表の与えられた行に適用することによって導出する。

2.9.1　比較述語

比較述語は２つの値の比較を指定する。

[形式]

```
値式－１{ = | <> | > | < | >= | <= }{ 値式－２ | 副問合せ}
```

[規則]

1. 値式－１のデータ属性と副問合せの結果または値式－２のデータ属性は比較可能でなければならない。
2. 副問合せを指定するとき，その結果は１つの値に定まらなくてはならない。

[説明]

1. 値式－１の結果が NULL 値ならば，比較述語の結果は不定である。
2. 値式－２を指定するとき，値式－２の結果が NULL 値ならば，比較述語の結果は不定である。
3. 副問合せを指定するとき，副問合せの結果が NULL 値または空ならば，比較述語の結果は不定である。
4. 上記以外の場合，比較述語の結果は真か偽のいずれかである。

   “＝”は左辺の値と右辺の値が等しいことを表す。

   “＜＞”は左辺の値と右辺の値が等しくないことを表す。

   “＜”は左辺の値が右辺の値未満であることを表す。

   “＞”は左辺の値が右辺の値を越えていることを表す。

   “＜＝”は左辺の値が右辺の値以下であることを表す。

   “＞＝”は左辺の値が右辺の値以上であることを表す。
5. 数は，代数値として比較する。
6. ２つの文字列の比較は，同じ順序位置にある文字の比較によって決定する。文字列が同じ長さでないとき，短い方の文字列は長い方の文字列と同じ長さになるように一時的に空白を補って比較を行う。
7. 同じ順序位置にあるすべての文字が等しいならば，２つの文字列は等しい。２つの文字列が等しくないならば，それらの関係は，２つの文字列の左端から最初の等しくない文字の対の比較によって決定する。文字の対の比較は，その文字の内部で表現されている文字コードによって行う。
8. ２つの日本語文字列の比較は，同じ順序位置にある日本語文字の比較によって決定する。日本語文字列が同じ長さでないとき，短い方の日本語文字列は長い方の日本語文字列と同じ長さになるように一時的に空白を補って比較を行う。
9. 同じ順序位置にあるすべての日本語文字が等しいならば，２つの日本語文字列は等しい。２つの日本語文字列が等しくないならば，それらの関係は，２つの日本語文字列の左端から最初の等しくない日本語文字の対の比較によって決定する。文字の対の比較は，その文字が内部で表現されている文字コードによって行う。

2.9.2　BETWEEN 述語

BETWEEN 述語は範囲比較を指定する。

[形式]

```
値式－１ [ NOT ] BETWEEN 値式－２ AND 値式－３
```

[規則]

1. ３つの値式のデータ属性は，比較可能でなければならない。

[説明]

1. “NOT”を指定しないときの BETWEEN 述語の結果は

   ```
   値式－１＞＝値式－２ AND 値式－１＜＝値式－３
   ```

   の結果と同じである。

2. ```
   x NOT BETWEEN y AND z
   ```

   の結果は，

   ```
   NOT( x BETWEEN y AND z )
   ```

   の結果と同じである。

2.9.3　IN 述語

IN 述語は値の集合との比較を指定する。

[形式]

- 書き方１

  ```
  値式 [ NOT ] IN ( 値指定 [ , 値指定 ] … )
  ```

- 書き方２

  ```
  値式 [ NOT ] IN 副問合せ
  ```

[規則]

1. 最初の値式のデータ属性と，副問合せまたはすべての値指定のデータ属性は，比較可能でなければならない。
  ```

[説明]

1. 書き方１で“NOT”を指定しないと，IN 述語の結果は以下のとおりである。
   a. 値式の結果が NULL 値ならば，不定である。
   b. 値式の結果がいずれかの値指定の値と一致するならば，IN 述語の結果は真である。
   c. 上記以外でいずれかの値指定の値が NULL ならば, IN 述語の結果は不定である。
   d. 上記以外で値式の結果がいずれの値指定の値とも一致しないならば，IN 述語の結果は偽である。
2. 書き方２で“NOT”を指定しないと，IN 述語の結果は以下のとおりである。
   a. 値式の結果が NULL 値ならば，不定である。
   b. 値式の結果が副問合せの結果のいずれかと一致するならば，IN 述語の結果は真である。
   c. 上記以外で副問合せの結果のいずれかが NULL 値ならば，IN 述語の結果は不定である。
   d. 上記以外で値式の結果が副問合せの結果のいずれとも一致しないならば，IN 述語の結果は偽である。

2.9.4　LIKE 述語

LIKE 述語はパターン照合比較を指定する。

[形式]

```
列指定 [ NOT ] LIKE 値指定－１ [ ESCAPE 値指定－２ ]
```

[規則]

1. 列指定のデータ属性は，文字列または日本語文字列でなければならない。
2. 列指定のデータ属性が文字列ならば，値指定－１，値指定－２のデータ属性は文字列でなければならない。列指定のデータ属性が日本語文字列ならば，値指定－１のデータ属性は日本語文字列でなければならない。
3. 値指定－２の長さは１でなければならない。

[説明]

1. 列指定によって参照する値または値指定－１の結果が NULL 値ならば，LIKE 述語の結果は不定である。そうでなければ，LIKE 述語の結果は真か偽のいずれかである。

2. “NOT”を指定しないときの LIKE 述語の結果は以下のとおりである。

   列指定によって参照される値と値指定－１の値のパターン比較が行われ，一致すれば，LIKE 述語の結果は真である。パターン比較は以下のように行われる。値指定－１中のパーセント文字は 0 文字以上の任意の文字列と一致し，下線文字は任意の１文字と一致する。それ以外の文字は同じ文字と一致する。ただし，ESCAPE 値指定－２が指定されている場合，その値はそれに続くパーセント文字，下線文字，または値指定―２で指定した文字自身の意味を打ち消すように作用する。すなわち，その値をｘとすると，ｘ％は%，ｘ＿は_，ｘｘはｘとそれぞれ一致する。

3. ｘ　NOT　LIKE　ｙ　[　ESCAPE　ｚ　]の結果は

   NOT　(　ｘ　LIKE　ｙ　[　ESCAPE　ｚ　]　)の結果と同じである。

2.9.5　NULL 述語

NULL 述語は NULL 値のテストをする。

[形式]

```
列指定 IS [ NOT ] NULL
```

[規則]

なし

[説明]

1. “NOT”を指定していないときの NULL 述語の結果は以下のとおりである。
   a. 列指定によって参照される値が NULL 値であるとき，NULL 述語の結果は真である。
   b. そうでなければ，NULL 述語の結果は偽である。
2. ｘ　IS　NOT　NULL　述語の結果は，

   NOT　(　ｘ　IS　NULL　)　の結果と同じである。

2.9.6　限定述語

限定述語は限定された値の集合との比較を指定する。

[形式]

```
値式 { = | <> | > | < | >= | <= }{ ALL | SOME | ANY }副問合せ
```

[規則]

1. 値式と副問合せのデータ属性は，比較可能でなければならない。
2. “SOME”と“ANY”は同義語である。

[説明]

1. “ALL”を指定した場合，限定述語の結果は以下のとおりである。
    a. 副問合せの結果が空かまたは想定されている比較述語が副問合せの結果の値すべてに対して真ならば述語の結果は真である。
    b. 想定されている比較述語が副問合せ結果の値のうちで少なくとも 1 つの値に対して偽ならば，限定述語の結果は偽である。
    c. それ以外は不定である。
2. “SOME”または“ANY”を指定した場合，限定述語の結果は以下のとおりである。
    a. 想定されている比較述語が副問合せ結果の値のうちで少なくとも 1 つの値に対して真ならば，限定述語の結果は真である。
    b. 副問合せの結果が空かまたは想定されている比較述語が副問合せ結果の値すべてに対し偽ならば，限定述語の結果は偽である。
    c. それ以外は不定である。

2.9.7　EXISTS 述語

EXISTS 述語は空集合のテストを指定する。

[形式]

```
EXISTS 副問合せ
```

[規則]

なし

[説明]

1. EXISTS 述語の結果は以下のとおりである。

a. 副問合せの結果が空でないときは真である。

b. 副問合せの結果が空のときは偽である。

2.10　探索条件

探索条件は，指定される条件に論理演算子を適用して“真”または“偽”となる条件を指定する。

[形式]

- 探索条件の形式

```
論理項 [ { AND | OR } 論理項 ]…
```

- 論理項の形式

```
[ NOT ] { 述語 | (探索条件) }
```

[規則]

なし

[説明]

1. 探索条件中で指定した列指定または値式が，その探索条件中に直接含まれているということは，その列指定または値式が集合関数指定内またはその探索条件の副問合せ内に指定していないことである。
2. 探索条件の結果は，指定した各述語を適用することによって得られる結果に，指定した論理演算子を適用することによって得る。論理演算子が指定してないならば，探索条件の結果は，指定した述語の結果である。
3. “NOT(真)”は偽である。“NOT(偽)”は真である。“AND”と“OR”の結果は，次の真理値表による。

AND	真	偽
真	真	偽
偽	偽	偽

OR	真	偽
真	真	真
偽	真	偽

4. かっこの中の式を最初に評価する。評価の順序をかっこによって指定していないとき，“NOT”は，“AND”の前に適用し，“AND”は，“OR”の前に適用し，そして同順序の演算子を左から右の順で適用する。
5. 探索条件を表のある行に適用するとき，探索条件中に直接含む列指定による，その表の列に対する各参照は，その行中の列の値への参照である。

2.11　問合せ指定

問合せ指定は，対象となる表から特定の項目を選択し導出される表を指定する。

[形式]

```
SELECT [ ALL  | DISTINCT ]{ * | 値式[ , 値式]…}
      FROM 表名  [相関名]  [ , 表名 [相関名]]…
      [WHERE 探索条件]
      [GROUP BY 列指定  [ , 列指定] …]
      [HAVING 探索条件]
```

[規則]

1. 問合せ指定中に含まれる各表名に対して SELECT 権限を持っていなければならない。
2. 各値式中の各列指定は，FROM 句中の表の列を一意に参照しなければならない。
3. 予約語“DISTINCT”は，その問合せ指定中の副問合せ以外では，問合せ指定の中で複数回指定してはならない。
4. FROM 句以下で最後に指定した句の結果が対象の表となる。
5. 対象となる表がグループ表ならば，各値式中の列指定はグループ化列を参照するか集合関数指定中で指定しなければならない。対象となる表がグループ表でなく，値式が集合関数指定を含むならば，いずれの値式中のいずれの列指定も集合関数指定中で指定しなければならない。

[説明]

1. 問合せ指定によって指定される表の列の数は，値式の数である。
2. “＊”指定は，値式のならびに等しい。ここで，各値式は FROM 句中で指定される表の各列をそれらの順序位置の昇順で参照する。
3. 問合せ指定の結果である表の各列のデータ属性は，列が導出された値式と同じデータ属性，長さ，精度および位取りである。
4. i 番目の値式が単一の列指定からなるならば，結果の i 番目の列は，その列名が列指定の列名であるような名前付きの列となる。そうでなければ，i 番目の列は，名前なしの列となる。

5. 問合せ指定が次の条件を満たすとき，更新可能となる。
    a. “DISTINCT”指定していない。
    b. いずれの値式も，１つの列指定からなる。
    c. FROM 句がただ１つの表を指定し，その表が更新可能な表を参照する。
    d. WHERE 句が副問合せを含まない。
    e. GROUP BY 句，HAVING 句のいずれも指定しない。
6. GROUP BY 句も HAVING 句も指定しない場合，
    a. 値式が集合関数指定を含むならば，対象となる表は，値式中の各集合関数指定の引数となり，問合せ指定の結果は１つの行からなる表となる。行の i 番目の値は i 番目の値式によって指定する値である。
    b. すべての値式が集合関数指定を含まないならば，各値式は対象となる表の各行に適用し，いくつかの行をもつ表を作成する。その行数は対象となる表のレコードの数である。表の i 番目の列は i 番目の値式を適用することによって導出した値を含む。
7. GROUP BY 句または HAVING 句を指定した場合，
    a. 対象となる表が１つ以上のグループをもつグループ表ならば，各値式は対象となる表の各グループに適用し，いくつかの行をもつ表が作成する。その行数は，対象となる表中のグループの数である。表の i 番目の列は，i 番目の値式を適用することによって導出した値を含む。値式が対象となる表のある与えられたグループに適用するとき，そのグループは，値式中の各集合関数指定の引数となる。
8. “DISTINCT”を指定すると，問合せ指定の結果から冗長な重複行が取り除かれる。行が他の行と重複するとは，同じ順序位置を持つすべての値の対が同一であることである。

2.11.1　FROM 句

FROM 句は，１つ以上の名前付きの表から導出される表を指定する。

[形式]

```
FROM 表名 [ 相関名 ] [ , 表名 [ 相関名 ] ]…
```

[規則]

1. FROM 句中で相関名を指定していない表名は，その FROM 句中で相関名を指定していない他のどの表名とも同じであってはならない。

2. 相関名は，それを含む FROM 句中で指定する他のどの相関名とも同じであってはならず，また，それを含む FROM 句中で相関名を指定していないどの表名の表識別子とも同じであってはならない。
3. FROM 句中で指定した相関名と相関名が指定した表名の有効範囲は，その FROM 句を含む最も内側の副問合せ，問合せ指定または SELECT 文とする。FROM 句中で指定した表名がその FROM 句で定義した有効範囲をもつとは，その表名に相関名を指定していないことである。

[説明]

1. FROM 句の結果の記述は，次のとおりである。
    a. FROM 句が単一の表名を含んでいるならば，FROM 句の結果の記述はその表名によって識別する表の記述と同じである。
    b. FROM 句が複数の表名を含んでいるならば，FROM 句の結果の記述は FROM 句に現れる表名の順で，それらの表名によって識別する表の記述を連結したものである。
2. FROM 句の結果は以下のとおりである。
    a. FROM 句が単一の表名を含むならば，FROM 句の結果は，その表名によって識別する表である。
    b. FROM 句が複数の表名を含むならば，FROM 句の結果は，それらの表名によって識別する表の直積であり，各表の行の可能な組み合わせを含む。

2.11.2　WHERE 句

WHERE 句は，先行する FROM 句の結果に探索条件を適用することによって導出される表を指定する。

[形式]

```
WHERE 探索条件
```

[規則]

1. 探索条件中に直接含まれる各列指定は，一意に FROM 句の結果の列を参照しなければならない。
2. 探索条件中に直接含まれる値式は，集合関数指定を含んではならない。

[説明]

1. 探索条件は，FROM 句の結果の各行に適用する。WHERE 句の結果は，探索条件の結果が真である FROM 句の結果の行からなる表である。

2.11.3　GROUP BY 句

GROUP BY 句は，前に指定した句の結果に GROUP BY 句を適用することによって導出するグループ表を指定する。

[形式]

```
GROUP BY 列指定 [ , 列指定 ]…
```

[規則]

1. GROUP BY 句中の各列指定は，一意に前に指定した句(FROM 句または WHERE 句)の結果の列を参照しなければならない。GROUP　BY 句で参照する列は，グループ化列となる。

[説明]

1. GROUP BY 句の結果は，前に指定した句の結果をグループ化列の値がすべて同一なグループの集まりに分割したものである。

2.11.4　HAVING 句

HAVING 句は，直前で指定した句の結果であるグループ表に対し，探索条件に合うグループだけを選び出す制限を指定する。

[形式]

```
HAVING 探索条件
```

[規則]

1. 探索条件中に直接含まれる各列指定は，グループ化列を一意に参照しなければならない。
2. 前に指定した句の結果の列を参照する探索条件中の副問合せ中に含まれる各列指定はグループ化列を参照するか，集合関数指定中に指定しなければならない。

[説明]

1. 先行する FROM 句，WHERE 句または GROUP BY 句の結果が対象となるグループ表である。HAVING 句の前に GROUP BY 句がなければ, 対象であるグループ表はただ 1 つのグループからなり，グループ化列をもたない。
2. 探索条件は，グループ分けされた各グループに適用する。HAVING 句の結果は，探索条件の結果が真であるようなグループのグループ表である。
3. 探索条件が対象となるグループ表のある与えられたグループに適用するとき，そのグループは，探索条件中に直接含む各集合関数指定の引数となる。

2.12　問合せ式

問合せ式は表を指定する。

[形式]

```
問合せ指定 [ UNION [ ALL ] 問合せ式 ]
```

[規則]

1. “UNION”を指定すると，指定した各問合せ指定の選択リストは，“＊”または列指定からならなければならない。“UNION”の左辺と右辺の結果の表の列数および各列のデータ型の属性，長さ，桁数，位取りは同一でなければならず，結果として導出した表の列は名前なしの列となる。

[説明]

1. “UNION”を指定せず問合せ式が更新可能なら，更新可能な表となる。
2. “UNION”を指定しないと，問合せ結果の表となる。
3. “UNION”を指定すると，“UNION”の左辺と右辺の結果の表中のすべての行を含む表が結果となる。ただし，“ALL”を指定しないと，すべての冗長な重複行は除外される。

2.13　副問合せ

副問合せは，対象となる表から特定の項目を選択し，導出される値の集合を指定する。

[形式]

```
( SELECT [ ALL  | DISTINCT ]{ * | 値式}
     FROM 表名 [相関名] [ , 表名 [相関名]]…
     [WHERE 探索条件] [GROUP BY 列指定 [ , 列指定] …] [HAVING 探索条件] )
```

[規則]

1. 副問合せ中に含まれる各表名に対して SELECT 権限をもっていなければならない。
2. 値式中の各列指定は，FROM 句中の表名で指定する表の列を一意に参照しなければならない。
3. FROM 句以下で最後に指定した句の結果が対象の表となる。
4. 対象となる表がグループの表ならば，値式中の各列指定はグループ化列を参照するか集合関数指定中で指定しなければならない。対象となる表がグループの表でなく，値式が集合関数指定を含むならば，値式の中の列指定は集合関数指定中で指定しなければならない。
5. 予約語“DISTINCT”は，その副問合せ中に含む副問合せを除いて副問合せ中で複数回指定してはならない。
6. 副問合せを比較述語中で指定するならば, GROUP BY 句また HAVING 句を指定してはならない。

[説明]

1. “＊”指定の意味は，次のとおりである。
   a. “＊”を EXISTS 述語以外の任意の述語の副問合せ中で指定するならば，対象となる表の次数は，１でなければならず，また，“＊”は対象となる表の単独の列を参照する列指定からなる値式である。
   b. “＊”を EXISTS 述語の副問合せ中に指定するならば，“＊”は集合関数指定を含まない値式で，かつ副問合せ中で許されている任意の値式である。
2. 副問合せ値のデータ属性は，暗または明の値式のデータ属性である。
3. GROUP BY 句も HAVING 句も指定しない場合，
   a. 値式が集合関数指定を含むならば，対象となる表は，値式中の各集合関数指定の引数となり，副問合せの結果は値式によって指定した値となる。
   b. 値式が集合関数指定を含まないならば，値式を対象となる表の各行に適用し，いくつかの行をもつ表を作る。その行数は対象となる表のレコード数である。
4. GROUP BY 句または HAVING 句を指定した場合, 値式は対象となる表の各グループに適用し，いくつかの行をもつ表を作る。その行数は，対象となる表中のグループの数である。値式が対象となる表のある与えられたグループに適用するとき，そのグループは，値式中の各集合関数指定の引数となる。

5. “DISTINCT”を指定すると，副問合せの結果から冗長な重複行を取り除く。行が他の行と重複するとは，同じ順序位置をもつすべての値の対が同一であることである。

第 3 章
SQL 文の言語規則

この章ではデータ操作の方法について説明する。

データ操作言語は，A 領域，B 領域に関係なく書き始めることができる。

3.1　ホスト変数

[機能]

ホスト変数は，COBOL 言語で宣言され，データベースとの間で共有するデータ項目である。ホスト変数は，データベースとプログラムの間の通信を仲介する。

[形式]

```
EXEC SQL BEGIN DECLARE SECTION END-EXEC.
{ ホスト変数の宣言 | INCLUDE ファイル名文 }…
EXEC SQL END DECLARE SECTION END-EXEC.
```

[規則]

1. ホスト変数の宣言，INCLUDE ファイル名文は，順不同で何回でも記述できる。
2. ホスト変数の宣言の形式は次のとおりである。

   ```
   レベル番号 ホスト変数名 属性
   ```

   宣言できるホスト変数の形式，データ属性については，「2.4　データ属性（9 ページ）」を参照のこと。
3. INCLUDE ファイル名文の形式については，「3.3　INCLUDE ファイル名（34 ページ）」を参照のこと。
4. ホスト変数に無名項目(名標なし/filler)は記述できません。
5. ホスト変数は，作業場所節(WORKING-STORAGE SECTION)でのみ定義することができる。
6. BEGIN DECLARE～END DECLARE は，１翻訳単位のプログラム中，いくつでも記述ができる。

[説明]

1. ホスト変数として定義されたデータ項目のみ埋込み SQL の埋込み変数として参照ができる。

2. ホスト変数として定義したデータ項目は，必ずしも埋込み SQL の埋込み変数として参照されなくてもよい。
3. INCLUDE ファイル名文を指定した場合には，INCLUDE ファイル名文で指定したファイルの中身を展開し，そこに記述されている内容をホスト変数の宣言と解釈する。
4. COBOL のコメント行については，メモとして扱う。

3.2　INCLUDE SQLCA

[機能]

INCLUDE SQLCA は, SQL 文を実行した情報を COBOL プログラムに引き渡すための領域を定義することを指示する。

[形式]

```
EXEC SQL INCLUDE SQLCA END-EXEC.
```

[規則]

1. INCLUDE SQLCA は，作業場所節(WORKING-STORAGE SECTION)でのみ定義することができる。
2. INCLUDE SQLCA は， 1 翻訳単位のプログラム中に最大 1 回記述することができる。

[説明]

1. INCLUDE SQLCA を記述すると，その場所に，下記のデータが展開される。

```
01      SQLCA.
  02    SQLCAID   COMP-2  VALUE 100.
  02    SQLCODE   COMP-2  VALUE 0.
  02    SQLERRM.
    03  SQLERRML  COMP-2  VALUE 0.
    03  SQLERRMC  PIC X(80).
  02    SQLRCNT   COMP-2.
  02    FILLER    COMP-2.
  02    FILLER    PIC X(5).
  02    FILLER    PIC X(1).
  02    FILLER    PIC X(6).
  02    SQLSTATE  PIC X(5).
  02    FILLER    PIC X(1).
  02    SQLMSG    PIC X(256).
  02    FILLER    PIC X(4).
```

各データの意味や設定内容については,『COBOL SQL アクセス プログラミングの手引』の「2.1.3 SQLCA」を参照されたい。

2. INCLUDE SQLCA は必ず 1 回記述がなければならない。記述がなかった場合，複数記述された場合，作業場所節に記述していないときの動作は不定である。

3.3　INCLUDE ファイル名

[機能]

INCLUDE ファイル名文は，指定したファイルの内容を INCLUDE ファイル名文が記述された場所に展開することを指示する。

[形式]

```
EXEC SQL INCLUDE ファイル名 END-EXEC
```

[規則]

1. INCLUDE ファイル名文は，プログラム中のどこの場所に記述してもよい。
2. INCLUDE ファイル名文は， 1 翻訳単位中，何度も記述してよい。
3. ファイル名の規則は次のとおりである。
   a. 文字数は 30 文字以下でなければならない。
   b. 英字から始まり英数字，ハイフン，下線から構成される。ただし，末尾はハイフンであってはならない。

[説明]

1. INCLUDE ファイル名文を記述すると，その場所に，ファイルの内容を展開する。
2. ファイル名の検索規則については，『COBOL SQL アクセスプログラミングの手引』の「1.2 INCLUDE ファイル」を参照されたい。
3. ファイル名には，COBOL の等価規則を適用しない。すなわち，ファイル名が中に英小文字を含んでいても，対応する英大文字には変換しない

   例　EXEC SQL INCLUDE aAA END-EXEC.と記述した場合，SQL プリコンパイラは「aAA」に COBOL の拡張子(.cob または.cbl)を付加したファイル名を検索する。

3.4　静的 SQL 文

静的 SQL 文について説明する。

静的 SQL 文には，次のものがある。

• 「3.4.1　カーソル宣言文（35 ページ）」

3.4.1　カーソル宣言文

[機能]

カーソルを定義する。

[形式]

```
DECLARE カーソル名 CURSOR FOR 問合せ式
[ ORDER BY ソート指定　[ , ソート指定] … ]
```

ソート指定

```
{ 整数 | 列指定 } [ ASC | DESC ]
```

[規則]

1. カーソル名は一意でなければならない。
2. ソート処理の列指定は，カーソル指定で指定される表の列でなければならない。
3. ソート指定には符号なし整数を指定しなければならない。
4. ソート指定に整数 i を指定する場合， i は 1 以上で結果の表の次数以下でなければならない。このときのソート指定は結果の表の i 番目の列を指定したことになる。

[説明]

1. カーソル指定で指定される表は
   a. “ORDER BY”も“UNION”もどちらも指定せず，かつ問合せ指定が更新可能ならば，更新可能な表となる。
   b. そうでなければ，更新可能でない表となる。

2. “ORDER BY”を指定すると，指定した列の値によって結果の表はソートする。
    a. ソート指定を２つ以上指定すると，１番目のソート指定によってソートした後，２番目のソート指定によってソートし，以下順番にソートする。
    b. ソート指定に“DESC”を指定すると，ソート方向は降順となる。“ASC”を指定するかまたは何も指定しなければ，昇順となる。
    c. ソートは比較述語で指定される比較規則によって決定する。NULL 値は，NULL 値以外のいかなる値より小さい値として扱う。

3.4.2　COLSE 文

[機能]

指定したカーソルをクローズ状態にする。

[形式]

```
CLOSE カーソル名
```

[規則]

1. カーソル名で指定したカーソルは，カーソル宣言文によって定義されていなければならない。
2. 指定したカーソルはオープン状態でなければならない。

[説明]

1. カーソル名で指定したカーソルをクローズ状態にする。

3.4.3　単一行の DELETE 文

[機能]

カーソルで位置づけた表の１つの行を削除する。

[形式]

```
DELETE FROM 表名 WHERE CURRENT OF カーソル名
```

[規則]

1. 表名で指定される表に対して DELETE 権限を持たねばならない。

2. カーソル名で指定するカーソルはカーソル宣言文によって定義しなければならない。
3. 指定したカーソルの示す表は更新可能でなければならない。
4. 表名で指定する表は，カーソル名で指定するカーソル指定中の最初の FROM 句中で指定する表でなければならない。
5. 指定したカーソルは，行に位置づけなければならない。

[説明]

1. 指定した表からカーソルで位置づけた行を削除する。

3.4.4　複数行の DELETE 文

[機能]

表から複数の行を削除する。

[形式]

```
DELETE FROM 表名 [WHERE 探索条件]
```

[規則]

1. 表名で指定した表に対して DELETE 権限を持たねばならない。
2. 指定した表は更新可能でなければならない。
3. 指定した表は探索条件中に含まれるどの副問合せ中の FROM 句にも指定してはならない。
4. 表名の有効範囲は，DELETE 文(探索)全体である。

[説明]

1. 探索条件を指定しないと，表のすべての行を削除する。
2. 探索条件を指定すると，その探索条件の結果が真である行をすべて削除する。
3. 行の削除中に誤りが起こると，SQLCODE には負の値が設定される。

3.4.5　FETCH 文

[機能]

カーソルを表の次の行に位置づけ，その行の値を取出す。

[形式]

```
FETCH [ FROM ] カーソル名　INTO 変数指定 [ , 変数指定 ]…
```

[規則]

1. カーソル名で指定するカーソルはカーソル宣言文によって定義しなければならない。
2. 指定したカーソルはオープン状態になければならない。
3. 変数指定の個数は，カーソル名で指定する表の列数と同じでなければならない。
4. カーソル名で指定する表の列のデータ属性と変数指定のデータ属性には以下の規則がある。
    a. i 番目の変数指定のデータ属性が文字列ならば, i 番目の列のデータ属性は文字列でなければならない。
    b. i 番目の変数指定のデータ属性が日本語文字列ならば, i 番目の列のデータ属性は日本語文字列でなければならない。
    c. i 番目の変数指定のデータ属性が真数ならば, i 番目の列のデータ属性は真数でなければならない。

[説明]

1. カーソル名で指定する表が空であるか，またはカーソル名で指定するカーソルの位置が最終行か最終行の後ろにあるならば，そのカーソルは最終行の後ろに位置づけ，SQLCODE に値 100 を代入し，値は変数指定に代入しない。
2. カーソル名で指定するカーソルの位置が行の前ならば，そのカーソルはその行に位置づけ，その行中の値を対応する相手に代入する。
3. カーソル名で指定するカーソルの位置が最終行以外の行ならば，カーソルはその行の直後の行に位置づけ，その行の直後の行中の値をそれらの対応する相手に代入する。
4. 相手に値を代入している間に誤りが起きると，SQLCODE には，負の数を設定する。
5. カーソル名で指定するカーソルの現在行中の対応する値が NULL ならば相手指定には標識変数を指定していなければならず，その標識変数には，-1 を設定する。NULL でない場合，その標識変数には以下の値を設定する。
    a. 変数指定のデータ属性が長さ L の文字列で，カーソルの現在行中の対応する値の長さ M が L を超えるならば，標識変数には M を設定する。
    b. そうでなければ，標識変数には，0 を設定する。
6. i 番目の変数指定で指定する相手は，指定したカーソルが位置づけられている行中の i 番目の値に対応する。
7. 取出し結果は以下のとおりとなる。

a. 変数指定のデータ属性が文字列で, その長さがそれに対応する値の長さと等しいならば，変数指定の値には，それに対応する値を設定する。

b. 変数指定のデータ属性が長さ L の文字列で，それに対応する値の長さが L を超えるならば，変数指定の値には，それに対応する値の先頭の L 文字を設定する。

c. 変数指定のデータ属性が長さ L の文字列で，それに対応する値の長さ M が L 未満ならば，変数指定の先頭の M 文字には，それに対応する値を設定し，変数指定の末尾の L−M 文字には，空白文字を設定する。

d. 変数指定のデータ属性が真数ならば, 変数指定のデータ属性中に上位有効桁数を失わないそれに対する値の表現がなければならず，変数指定の値には，その表現を設定する。

3.4.6　INSERT 文

[機能]

表中に新しい行を追加する。

[形式]

- 書き方 1

```
INSERT INTO  表名 [ ( 列名 [ , 列名 ]… )]
   VALUES ({  値指定 | NULL }[ , 値指定 | , NULL]  …  )
```

- 書き方 2

```
INSERT INTO 表名[ ( 列名 [ , 列名 ] … )] 問合せ指定
```

[規則]

1. 表名で指定する表に対して INSERT 権限を持たねばならない。
2. 指定した表は，更新可能でなければならない。また，問合せ指定または問合せ指定中に含まれる副問合せの FROM 句中で指定する表であってはならない。
3. 列名で指定する列は，指定した表の列でなければならない。また，同じ列を複数回指定してはならない。
4. 書き方 1 では，VALUES 句の後の挿入値の個数は，列名の個数と等しくなければならない。
5. 書き方 2 では，問合せ指定で指定している表の列は，列名の個数と等しくなければならない。
6. INSERT 文の i 番目の項目の挿入値が NULL でないならば

a. i 番目の列名で指定する列のデータ属性が長さ L の文字列ならば，INSERT 文の i 番目の項目のデータ属性は，L 以下の長さの文字列でなければならない。

b. i 番目の列名で指定する列のデータ属性が長さ L の日本語文字列ならば，INSERT 文の i 番目の項目のデータ属性は，L 以下の長さの日本語文字列でなければならない。

c. i 番目の列名で指定する表の列データの属性が真数ならば, INSERT 文の i 番目の項目のデータ属性は，真数でなければならない。

[説明]

1. 表名の後ろの列名の並びを省略すると，表名で指定される表内の順序位置の昇順ですべての列が指定したのと同じ結果になる。
2. INSERT 文の i 番目の項目は，VALUES 句中の i 番目の値指定または NULL を参照する。
3. 表名の後ろに列名の並びを指定するとき，指定していない表の各列には NULL 値を挿入する。
4. VALUES 句を指定すると，次のとおりになる。
   a. VALUES 句中の i 番目の項が値指定ならば，i 番目の列にはその値指定の値を挿入する。
   b. VALUES 句中の i 番目の項が NULL ならば，i 番目の列には NULL 値を挿入する。
5. 問合せ指定を指定すると，挿入の結果は以下のとおりになる。
   a. 問合せ指定の結果が空ならば，SQLRCNT に値 0 を代入し，いかなる行も挿入しない。
   b. 問合せ指定の結果が空でなければ，その結果の各行を指定した表に挿入する。
6. 挿入値が NULL 値でないとき，挿入の結果は以下のとおりになる
   a. 列のデータ属性が文字列で, その長さがそれに対する挿入値の長さと等しいならば，列の値には，それに対応する挿入値を設定する。
   b. 列のデータ属性が長さ L の文字列で，それに対応する挿入値の長さ M が L 未満ならば，列の先頭の M 文字には，それに対応する挿入値を設定し，列の末尾の L－M 文字には，空白文字を設定する。
   c. 列の属性が真数ならば, 列のデータ属性中に上位有効桁数を失わないそれに対応する挿入値の表現がなければならず，列の値には，その表現を設定する。
7. 行の挿入中に誤りが起こると，SQLCODE には負の値を設定する。

3.4.7　OPEN 文

[機能]

指定したカーソルをオープン状態にする。

[形式]

```
OPEN カーソル名
```

[規則]

1. カーソル名で指定するカーソルは，カーソル宣言文によって定義されなければならない。
2. 指定したカーソルはクローズ状態でなければならない。

[説明]

1. カーソル名で指定するカーソルは次の順序でオープンする。
    a. カーソル指定中に現れる変数指定を現在のホスト変数の値と置き換える。
    b. カーソル指定で指定する問合せ式を評価することにより，問合せ式の結果となる表を作成する。
    c. 指定したカーソルは，オープン状態に置き，その位置は表の先頭行の前になる。

3.4.8　単一行 SELECT 文

[機能]

表の指定した行から値を取出す。

[形式]

```
SELECT [ ALL | DISTINCT ] { * | 値式 [ , 値式 ] … }
      INTO 変数指定 [ , 変数指定] …
      FROM 表名 [ 相関名 ] [ , 表名 [ 相関名 ]] … [ WHERE 探索条件 ]
```

[規則]

1. 表名および探索条件中の副問合せの FROM 句に含まれる各表名で指定される表に対して，SELECT 権限を持たねばならない。
2. 値式の個数は，変数指定の個数と同じでなければならない。
3. 値式とそれに対応する変数指定のデータ型には以下の規則がある。

a. 変数指定のデータ属性が文字列ならば, 値式のデータ属性は文字列でなければならない。

b. 変数指定のデータ属性が日本語文字列ならば, 値式のデータ属性は日本語文字列でなければならない。

c. 変数指定のデータ属性が真数ならば, 値式のデータ属性は真数でなければならない。

[説明]

1. INTO 句の変数指定には SELECT 文から INTO 句を除いた問合せ指定の結果が代入される。ただし，その問合せ指定の結果のレコードの数は 1 を越えてはならない。
2. 結果が空ならば，SQLCODE に値 100 が代入され，値は変数指定に代入されない。
3. 変数指定に値を代入している間に誤りが起きると，SQLCODE には，負の値が設定される。
4. ある変数指定に対する値が NULL 値ならば，標識変数が指定されていなければならず，その標識変数には－1 が設定される。その対応する値が NULL 値ではなく，変数指定が標識変数をもつならば，その標識変数は次のとおりとする。
   a. 変数指定のデータ属性が長さ L の文字列で，それに対応する値の長さ M が L を超えるならば，標識変数には M が設定される。
   b. そうでなければ，標識変数には 0 が設定される。
5. 選択の結果は，次のとおりである。
   a. 変数指定のデータ属性が文字列で, その長さがそれに対する値の長さと等しいならば，変数指定の値には，それに対応する値が設定される。
   b. 変数指定のデータ属性が長さ L の文字列で，それに対する値の長さが L を超えるならば，変数指定の値には，それに対する値の先頭の L 文字が設定される。
   c. 変数指定のデータ属性が長さ L の文字列で，それに対応する値の長さ M が L 未満ならば，変数指定の先頭の M 文字には，それに対応する値が設定され，変数指定の末尾の L－M 文字には，空白文字が設定される。
   d. 変数指定のデータ属性が真数ならば, 変数指定のデータ属性中に上位有効桁を失わないそれに対応する値の表現がなければならず，変数指定の値には，その表現が設定される。

3.4.9　単一行の UPDATE 文

[機能]

カーソルが位置づいた表の 1 つの行を更新する。

[形式]

```
UPDATE 表名 SET 列名 = { 値式 | NULL } [ , 列名 = { 値式 | NULL }]…
 WHERE CURRENT OF カーソル名
```

[規則]

1. 表名で指定する表は列名で指定する列に対して UPDATE 権限を持たねばならない。
2. カーソル名で指定するカーソルはカーソル宣言文によって定義していなければならない。
3. 指定したカーソルによって指定する表は更新可能でなければならない。
4. 指定した表は，指定したカーソルのカーソル指定中の最初の FROM 句中で指定する表でなければならない。
5. 値式は，集合関数指定を含んではならない。
6. 列名は，表名で指定する表の列でなければならない。同一の列名を複数回指定してはならない。
7. 表名の有効範囲は，UPDATE 文(位置づけ)全体である。
8. 各代入に対して
    a. 指定した列のデータ属性が長さ L の文字列ならば，値式のデータ属性は，L 以下の長さの文字列でなければならない。
    b. 指定した列のデータ属性が長さ L の日本語文字列ならば，値式のデータ属性は，L 以下の長さの日本語文字列でなければならない。
    c. 指定した列のデータ属性が真数ならば，値式のデータ属性は，真数でなければならない。
9. 指定したカーソルは，行に位置づけなければならない。

[説明]

1. 更新対象となる行は，カーソル名で指定するカーソルが位置づけている行に関連づけてる行である。
2. 更新対象となる行は，各代入文で指定するとおり更新する。
3. 値式が表名で指定する表の列への参照を含む場合，対象となる行のどの値も更新する前の対象となる行中のその列の値に対するものとする。
4. 更新結果は，次のとおりである。
    a. 列のデータ属性が文字列または日本語文字列で，更新値の長さが列の値の長さと等しいならば，列の値には，更新値を設定する。

b. 列のデータ属性が長さ L の文字列または日本語文字列で，更新値の長さ M が L 未満ならば，列の値の先頭の M 文字には，更新値を設定し，列の値の末尾の L－M 文字には，空白文字を設定する。

c. 列のデータ属性が真数ならば, 列のデータ属性中に上位有効桁を失わない更新値の値の表現がなければならず，列の値には，その表現を設定する。

3.4.10　複数行の UPDATE 文

[機能]

表の複数の行を更新する。

[形式]

```
UPDATE 表名 SET 列名 = { 値式 | NULL } [ , 列名 = { 値式 | NULL }]…
 [ WHERE 探索条件]
```

[規則]

1. 表名で指定する表は列名で指定する列に対して UPDATE 権限を持たねばならない。
2. 指定した表は，更新可能でなければならない。
3. 指定した表は探索条件中に含まれるどの副問合せ中の FROM 句にも指定してはならない。
4. 値式は，集合関数指定を含んではならない。
5. 列名で指定する列は，指定した表の列でなければならない。同一の列名が複数回現れてはならない。
6. 表名の有効範囲は，UPDATE 文(探索)全体である。
7. 列名に指定する列に対して，
   a. 指定した列のデータ属性が長さ L の文字列ならば，値式のデータ属性は，L 以下の長さの文字列でなければならない。
   b. 指定した列のデータ属性が長さ L の日本語文字列ならば，値式のデータ属性は，L 以下の長さの日本語文字列でなければならない。
   c. 指定した列のデータ属性が真数ならば，値式のデータ属性は，真数でなければならない。

[説明]

1. 探索条件を指定しないと，表すべての行が更新対象となる。
2. 探索条件を指定すると，その探索条件の結果が真である行がすべて更新対象となる。

3. 更新対象となる行は，各代入文で指定するとおり更新する。
4. 値式が指定した表の列への参照を含む場合，対象となる行のどの値も更新する前の対象となる行中のその列の値に対するものとする。
5. 更新結果は，次のとおりである。
    a. 列のデータ属性が文字列または日本語文字列で, 更新値の長さが列の値の長さと等しいならば，列の値には，更新値を設定する。
    b. 列のデータ属性が長さ L の文字列または日本語文字列で，更新値の長さ M が L 未満ならば，列の値の先頭の M 文字には，更新値を設定し，列の値の末尾の L－M 文字には，空白文字を設定する。
    c. 列のデータ属性が真数ならば, 列のデータ属性中に上位有効桁を失わない更新値の値の表現がなければならず，列の値には，その表現を設定する。
6. 行の更新中に誤りが起こると，SQLCODE には負の値を設定する。

3.5　動的 SQL 文

動的 SQL 文について説明する。

動的 SQL 文には，次のものがある。

3.5.1　動的カーソル宣言文

[機能]

動的 PREPARE 文によって関連づけた SQL 文指示子に従って，動的カーソルを定義する。

[形式]

```
DECLARE カーソル名 CURSOR FOR SQL 文識別子
```

[規則]

1. カーソル名は一意でなければならない。
2. 動的カーソル宣言文の SQL 文識別子と同じ SQL 文識別子を持つ動的 PREPARE 文を指定しなければならない。

[説明]

1. カーソル指定で指定する表は
   a. “ORDER BY” も “UNION” もどちらも指定せず，かつ問合せ指定が更新可能ならば，更新可能な表となる。
   b. そうでなければ，更新可能でない表となる。
2. “ORDER BY” を指定すると，指定した列の値によって結果の表をソートする。
   a. ソート指定を２つ以上指定すると，１番目のソート指定によってソートした後，２番目のソート指定によってソートし，以下順番にソートする。
   b. ソート指定に “DESC” を指定すると，ソート方向は降順となる。“ASC” を指定するかまたは何も指定しなければ，昇順となる。
   c. ソートは比較述語で指定する比較規則によって決定する。NULL 値は，NULL 値以外のいかなる値より小さい値として扱う。

3.5.2　PREPARE 文

[機能]

実行するための文を準備する。

[形式]

```
PREPARE SQL文識別子 FROM SQL文変数
```

[規則]

1. SQL 文変数のデータ型は文字列または可変長文字列でなければならない。

[説明]

1. SQL 文変数に設定する SQL 文は“EXEC SQL”，“END-EXEC”を含んではならない。
2. SQL 文変数に指定できる SQL 文は以下の文である。
   - DELETE 文(位置づけ)
   - DELETE 文(探索)

- INSERT 文
- UPDATE 文(位置づけ)
- UPDATE 文(探索)
- SELECT 文(単一行)
- 問合せ式

3. SQL 文変数に設定する SQL 文にホスト変数を指定してはならない。ホスト変数を指定する場合は動的パラメータによって指定する。
4. PREPARE 文を実行することにより SQL 変数に指定した SQL 文の実行を準備する。準備した SQL 文を実行するには EXECUTE 文を実行しなければならない。一度準備した SQL 文は EXECUTE 文で複数回実行することができる。

3.5.3　EXECUTE 文

[機能]

入力パラメータおよび出力相手に準備した文に関連づけ，その文を実行する。

[形式]

```
EXECUTE SQL 文識別子
        [INTO   変数名 [, 変数名]…]
        [USING 変数名 [, 変数名]…]
```

[規則]

1. 実行する SQL 文が単一行 SELECT 文ならば “INTO” を指定しなければならない。

[説明]

1. EXECUTE 文を行う前に必ず PREPARE 文を実行し SQL 文を準備しなければならない。
2. 準備した SQL 文に動的パラメータがある場合，“USING” を指定しなければならない。

3.5.4　EXECUTE IMMEDIATE 文

[機能]

SQL 文を動的に準備し実行する。

[形式]

```
EXECUTE IMMEDIATE SQL文変数
```

[規則]

1. SQL 文変数のデータ型は文字列または可変長文字列でなければならない。

[説明]

1. SQL 文変数に設定する SQL 文は“EXEC SQL”，“END-EXEC”を含んではならない。
2. SQL 文変数に指定できる SQL 文は以下の文である。
    - DELETE 文(位置づけ)
    - DELETE 文(探索)
    - INSERT 文
    - UPDATE 文(位置づけ)
    - UPDATE 文(探索)

3.5.5　動的 OPEN 文

[機能]

カーソル名に入力パラメータを関連付け，カーソルをオープンする。

[形式]

```
OPEN カーソル名 [USING 変数指定 [, 変数指定]…]
```

[規則]

1. カーソル名で指定するカーソル名は，カーソル宣言文によって宣言しなければならない。
2. 指定したカーソルはクローズ状態でなければならない。

[説明]

1. カーソル名に関連付けた SQL 文は問合せ式でなければならない。
2. カーソル名に関連付けた SQL 文に動的パラメータがあるなら USING 句を指定しなければならない。

3. カーソル名に関連付けた SQL 文を評価することにより，問合せ式の結果となる表を作成する。
4. 指定したカーソルはオープン状態に置かれ，その位置は表の先頭行の前になる。

3.5.6　動的 FETCH 文

[機能]

動的カーソル宣言文で宣言したカーソルに対して行を取出す。

[形式]

```
FETCH [FROM] カーソル名 INTO 変数指定 [, 変数指定]…
```

[規則]

1. カーソル名で指定したカーソルはカーソル宣言文によって定義しなければならない。
2. 指定したカーソルはオープン状態になくてはならない。
3. 変数指定の個数は，カーソル名で指定する表の列数と同じでなければならない。

[説明]

1. カーソル名で指定する表が空であるか，またはカーソル名で指定するカーソルの位置が最終行か最終行の後ろにあるならば，そのカーソルを最終行の後ろに位置づけ，SQLCODE に値 100 を代入し，値は変数指定に代入しない。
2. カーソル名で指定するカーソルの位置が行の前ならば，そのカーソルはその行に位置づけられ，その行中の値を対応する相手に代入する。
3. カーソル名で指定するカーソルの位置が最終行以外の行ならば，カーソルはその行の直後の行に位置づけ，その行の直後の行中の値をそれらの対応する相手に代入する。
4. 相手に値を代入している間に誤りが起きると，SQLCODE には，負の数を設定する。
5. カーソル名で指定するカーソルの現在行中の対応する値が NULL ならば相手指定には標識変数が指定されていなければならず，その標識変数には，-1 を設定する。NULL でない場合，その標識変数には以下の値を設定する。
    a. 変数指定のデータ属性が長さ L の文字列で，カーソルの現在行中の対応する値の長さ M が L を超えるならば，標識変数には M を設定する。
    b. そうでなければ，標識変数には，0 を設定する。
6. i 番目の変数指定で指定する相手は，指定したカーソルが位置づけた行中の i 番目の値に対応する。
7. 取出し結果は以下のとおりとなる。

a. 変数指定のデータ属性が文字列で, その長さがそれに対応する値の長さと等しいならば，変数指定の値には，それに対応する値を設定する。

b. 変数指定のデータ属性が長さ L の文字列で，それに対応する値の長さが L を超えるならば，変数指定の値には，それに対応する値の先頭の L 文字を設定する。

c. 変数指定のデータ属性が長さ L の文字列で，それに対応する値の長さ M が L 未満ならば，変数指定の先頭の M 文字には，それに対応する値を設定し，変数指定の末尾の L−M 文字には，空白文字を設定する。

d. 変数指定のデータ属性が真数ならば, 変数指定のデータ属性中に上位有効桁数を失わないそれに対する値の表現がなければならず，変数指定の値には，その表現を設定する。

3.5.7　単一行の動的 DELETE 文

[機能]

表の行を削除する。

[形式]

```
DELETE FROM 表名 WHERE CURRENT OF カーソル名
```

[規則]

1. 表名で指定される表に対して DELETE 権限を持たねばならない。
2. カーソル名で指定されるカーソルはカーソル宣言文によって定義されていなければならない。
3. 指定したカーソルの示す表は更新可能でなければならない。
4. 指定したカーソルは，行に位置づけなければならない。

[説明]

1. 指定した表からカーソルで位置づけられている行を削除する。

3.5.8　単一行の動的 UPDATE 文

[機能]

表の行を更新する。

[形式]

```
UPDATE 表名 SET 列名 = {値式 | NULL} [, 列名 = {値式 | NULL}]…
       WHERE CURRENT OF カーソル名
```

[規則]

1. 表名で指定する表は列名で指定する列に対して UPDATE 権限を持たねばならない。
2. カーソル名で指定したカーソルはカーソル宣言文によって定義しなければならない。
3. 指定したカーソルによって指定する表は更新可能でなければならない。
4. 値式は，集合関数を含んではならない。
5. 列名は，表名で指定する表の列でなければならない。 同一の列名を複数回指定してはならない。
6. 表名の有効範囲は，UPDATE 文(位置づけ)全体である。
7. 指定したカーソルは，行に位置づけなければならない。

[説明]

1. 更新の対象となる行は，カーソル名で指定するカーソルを位置づけている行に関連付けている行である。
2. 更新の対象となる行は，各代入文で指定したとおり更新する。
3. 値式が表名で指定さる表の列への参照を含む場合，対象となる行のどの値も更新する前の対象となる行中のその列の値に対するものとする。

3.6　トランザクション文

トランザクション文について説明する。

トランザクション文には，次のものがある。

- 「3.6.1　COMMIT 文（51 ページ）」
- 「3.6.2　ROLLBACK 文（52 ページ）」

3.6.1　COMMIT 文

[機能]

現トランザクションを終了してデータベースの更新を行う。

[形式]

```
COMMIT [WORK]
```

[規則]

なし

[説明]

1. 現行トランザクションを終了させる。
2. 現行トランザクションによってオープン状態になったすべてのカーソルをクローズ状態にする。
3. 現行トランザクションによってなされたすべての更新を確定する。

3.6.2　ROLLBACK 文

[機能]

現行トランザクションを無効にする。

[形式]

```
ROLLBACK [WORK]
```

[規則]

なし

[説明]

1. 現行トランザクションによってなされたすべての変更を取り消す。
2. 現行トランザクションによってオープン状態になったすべてのカーソルをクローズ状態にする。
3. 現行トランザクションを終了させ，新たに次のトランザクションを開始させる。

3.7　コネクション文

コネクション文について説明する。

コネクション文には，次のものがある。

3.7.1　CONNECT 文

[機能]

SQL 環境に SQL コネクションを確立する。

[形式]

```
CONNECT TO {SQL サーバ名 [AS コネクション名] [USER 利用者名] | DEFAULT}
```

SQL サーバ名

```
{変数名指定 | 文字列定数}
```

コネクション名

```
{変数名指定 | 文字列定数}
```

利用者名

```
{変数名指定 | 文字列定数}
```

[規則]

1. SQL サーバ名を変数名指定する場合，埋込み変数は可変長文字列で長さと文字列を設定しなければならない。
2. コネクション名を変数名指定する場合，埋込み変数は可変長文字列で長さと文字列を設定しなければならない。
3. 利用者名を変数名指定する場合，埋込み変数は可変長文字列で長さと文字列を設定しなければならない。

[説明]

1. コネクション名を省略した場合，SQL サーバ名とコネクション名は同じになる。

3.7.2　SET CONNECTION 文

[機能]

使用可能なコネクションから 1 つコネクションを選択する。

[形式]

```
SET CONNECTION {DEFAULT | コネクション名}
```

コネクション名

```
{変数名指定 | 文字列定数}
```

[規則]

1. 指定するコネクション名は，CONNECT 文によって確立したコネクション名と一致しなければならない。
2. コネクション名を変数名指定する場合，埋込み変数は可変長文字列で長さと文字列を設定しなければならない。

[説明]

1. 指定するコネクション名は必ず確立していなければならない。

3.7.3　DISCONNECT 文

[機能]

SQL コネクションを終了する。

[形式]

```
DISCONNECT {コネクション名 | ALL | CURRENT | DEFAULT}
```

コネクション名

```
{変数名指定 | 文字列定数}
```

[規則]

1. 指定するコネクション名は，CONNECT 文によって確立されたコネクション名と一致しなければならない。
2. コネクション名を変数名指定する場合，埋込み変数は可変長文字列で長さと文字列を設定しなければならない。

[説明]

1. コネクション名を指定すると指定したコネクションを終了する。
2. コネクション名を指定する場合，必ず接続が確立していなければならない。

3. “ALL” を指定すると接続が確立しているすべてのコネクションを終了する。
4. “CURRENT” を指定すると，現在確立しているコネクションを終了する。
5. 複数データベースへの確立を行っている場合に DISCONNECT 文を実行すると“CURRENT”に接続しているデータベースが不定になる。

3.8　埋込み例外処理

[機能]

SQL 文を実行した結果，例外条件が返ってきた場合の処理を COBOL プログラム中に宣言する。

[形式]

```
WHENEVER {NOT FOUND | SQLERROR}
         {CONTINUE | GOTO :手続き名 | GO TO :手続き名}
```

[説明]

1. “WHENEVER” の後に例外条件として“NOT FOUND”を指定した場合は，FETCH 文等でデータの終りを検出した際(SQLCODE に 100 が返る)に，制御を指定した部分へ渡す。また，例外条件として“SQLERROR”を指定した場合は，入力エラー等が発生した際(SQLCODE に負の値が返る)に制御を指定した部分へ渡す。
2. 例外条件の後に例外動作として’CONTINUE’が指定している場合は，例外が発生した文の次の文に制御が移行し(例外宣言を指定していない場合と同様に動作する)。また，例外動作として“GOTO :手続き名”を指定している場合には，手続き名で示した部分に制御が渡される。
3. “WHENEVER” による例外宣言は，同じ例外宣言を指定している次の“WHENEVER”を指定するまでの間にあるすべての SQL 文に対して有効である。

COBOL SQL アクセス
言語説明書

2018 年 10 月 3 版 発行

日本電気株式会社