東京ITスクール
TOKYO IT SCHOOL

SQLの概要とテーブル操作

目次

1. SQL の概要

1　標準 SQL

SQL（Structured Query Language）はリレーショナルデータベースを操作するための言語です。SQL を用いることにより、データの検索、データの登録や削除といったデータベース操作が出来ます。

SQL には ISO（国際標準化機構）で定められた標準規格があり、それに準拠した SQL を標準 SQL と呼びます。

2　SQL の文とその種類

SQL は、いくつかのキーワードと、テーブル名や列名などを組み合わせた 1 つの文（SQL 文）として、操作の内容を記述します。キーワードは、最初から意味や使い方が決められている特別な英単語で、「テーブルを検索する」「このテーブルを参照する」などの意味を持つ様々なものがあります。

SQL 文は、RDBMS に与える命令の種類により、次の 3 つに分類されます。

DDL（Data Definition Language）

DDL（データ定義言語）は、データを格納する入れ物であるデータベースやテーブルなどを作成したり削除したりします。DDL に分類される命令は次の通りです。

CREATE：データベースやテーブルなどを作成する
DROP：データベースやテーブルなどを削除する
ALTER：データベースやテーブルなどの構成を変更する

DML（Data Manipulation Language）

DML（データ操作言語）は、テーブルの行を検索したり変更したりします。
DML に分類される命令は次の通りです。

SELECT：テーブルから行を検索する
INSERT：テーブルに新規行を登録する
UPDATE：テーブルの行を更新する
DELETE：テーブルの行を削除する

DCL（Data Control Language）

DCL（データ制御言語）は、データベースに対して行った変更を確定したり取り消したりします。その他、RDBMS のユーザがデータベースにあるもの（テーブルなど）を操作する権限の設定も行います。DCL に分類される命令は次の通りです。

COMMIT：データベースに対して行った変更を確定する
ROLLBACK：データベースに対して行った変更を取り消す
GRANT：ユーザに操作の権限を与える
REVOKE：ユーザから操作の権限を奪う

3　SQL の基本的な記述ルール

SQL 文を書く際には、いくつかの記述ルールを守らなければなりません。

■SQL 文の最後に「;」をつける

1 つのデータベース操作は、1 つの SQL 文で記述します。RDBMS でも、SQL 文を 1 つずつ実行していきます。SQL では文の区切り文字としてセミコロン「;」を使います。

■大文字・小文字は区別されない

SQL では、キーワードの大文字・小文字は区別されません。「SELECT」と書いても「select」と書いても同じように解釈されます。テーブル名や列名などについては、データベースの種類によって異なりますが、Oracle の場合は区別しません。

ただし、テーブルに登録されているデータについては、大文字・小文字が区別されます。例えば、「Yamada」と登録したデータを「YAMADA」や「yamada」と同じに扱ったりはしません。

■定数の書き方には決まりがある

SQL 文では、文字列や日付、数値を直接書く場面が頻繁に出てきます。例えば、テーブルに文字列や日付、数値などのデータを登録する SQL 文を書く時などです。

文字列を SQL 文の中に記述する場合には、「'abc'」というようにシングルクォーテーション「'」で文字列を囲んで、それが文字列であることを示します。

日付を SQL 文の中に記述する場合には、文字列と同じくシングルクォーテーション「'」で囲みます。ただし、日付は様々な表現形式（「'26 Jan 2010'」や「'10/01/26'」など）があります。

一方、数値を SQL 文の中に記述する場合には、何かの記号で囲む必要はありません。「１０００」というように数値だけを書きます。

■単語は半角スペースか改行で区切る

SQL 文では、単語と単語の間を「半角スペース（空白）」または「改行」で区切ります。次のように単語の区切りをなくして繋げて書くとエラーになり、正しく動作しません。

○ CREATE TABLE Emp
× CREATETABLE Emp
× CREATE TABLESEmp

2. テーブルの作成

1　CREATE TABLE 文

SQL の学習を始めるに当たり、まずはテーブルの作成方法から学んでいきましょう。テーブルを作成するのは、CREATE TABLE 文を使用します。CREATE TABLE 文の構文は次の通りです。

（構文）CREATE TABLE 文

```
CREATE TABLE <テーブル名>
(<列名 1><データ型><この列の制約>,
 <列名 2><データ型><この列の制約>,
 <列名 3><データ型><この列の制約>,
 <列名 4><データ型><この列の制約>,
                 :
                 :
 <このテーブルの制約 1>,<このテーブルの制約 2>,……);
```

「列名」とは、表（テーブル）の項目名です。「データ型」とは項目に入る情報の形式、「制約」とはルールの様なものだと思ってください。

2　命名ルール

データベースやテーブル、列といった名前に使える文字は、半角文字のアルファベット、数字、アンダーバー（_）に限られます。例えば、「emp_id」を「emp-id」と書いてはいけません。ハイフンを列名などに使うことは、標準 SQL において認められていないからです。同様に、「$」や「#」や「?」の様な記号も名前に使ってはいけません。

RDBMS の中には、こうした記号や日本語（全角文字）を列の名前などに使うことの出来るものもありますが、それはあくまでその RDBMS が独自に認めているだけで、他の RDBMS でも同様に使えるという保証はありませんので注意しましょう。

また、名前の最初には必ず半角のアルファベットを使わなければなりません。「1_mail」の様な命名は、標準 SQL では禁止されています。「mail_1」の様に表記しましょう。

最後に、1 つのデータベースの中に同じ名前のテーブルを 2 つ以上作る事や、1 つのテーブル内に同じ名前の列を 2 つ以上作る事は出来ません。

3　データ型の指定

列には、必ずデータ型を指定する必要があります。データ型はデータの種類を表わし、数値型や文字列型、日付型などがあります。どの列も、データ型に反するデータを入れることは出来ません。整数型と宣言された列に「'あいうえお'」のような文字列を格納することは出来ませんし、文字列型と宣言された列に「1234」という数値を入力することは出来ません。

データ型には非常に多くの種類があり、また RDBMS 毎に違います。まずは、次に挙げる 4 つの基本的なデータ型を覚えておきましょう。

●INTEGER 型

整数を入れる列に指定するデータ型（整数型）です。少数は入れられません。

●CHAR 型

CHAR は CHARACTER（文字）の略で、文字列を入れる列に指定するデータ型（文字列型）です。CHAR(10)や CHAR(200)のように、列の中に入れることの出来る文字列の長さ（最大長）をカッコ（）で指定します。最大長を超える長さの文字列は入りません。長さの単位は RDBMS により異なり、文字数である場合とバイト長である場合があります。

CHAR 型の列には、固定長文字列という形式で文字列が格納されます。固定長文字列では、列に入れる文字列の長さが最大長に満たない場合、文字数が最大長になるまで空きを半角スペースで埋めます。例えば、CHAR(8)の列に「'abc'」という文字を入れたとしましょう。すると、「'abc□□□□□'」（「abc」の後ろに半角スペースが 5 つ）という形で格納されます。

●VARCHAR 型

CHAR 型と同じく文字列を入れる列に指定するデータ型（文字列型）で、やはり、列の中に入れることの出来る文字列の長さ（最大長）をカッコで指定します。ただし、こちらは可変長文字列という形式で、列の中に文字列が入ります。固定長文字列では、文字数が最大長に満たない場合に半角スペースで埋めていましたが、可変長文字列では文字数が最大長に満たなくても、半角スペースで埋めたりしません。VARCHAR(8)の列に「'abc'」という文字列を入れた場合、データはそのまま「'abc'」です。

また、Oracle は VARCHR というデータ型を持っていますが、使用が推奨されておらず、可変長文字列には、「VARCHAR2」型を使用します。

●DATE 型

日付（年月日）を入れる列に指定するデータ型（日付型）です。また、Oracle の DATA 型は、データに年月日だけでなく時分秒まで含みます。

以上の 4 つが最初に覚えるべき基本的なデータ型です。以下に Oracle を例にその他のデータ型を表します。

データ型	制限	説明
VARCHAR2（n）	最大 4000 バイト	可変長文字列型（n に最大サイズを指定）
NVARCHAR2（n）	最大 4000 バイト	各国語キャラクタセットを使用する UNICODE データ型の可変長文字列型（n に最大サイズを指定）
CHAR（n）	最大 2000 バイト	固定長文字列型（n に最大サイズを指定）
NCHAR（n）	最大 2000 バイト	各国語キャラクタセットを使用する UNICODE データ型の固定長文字列型（n に最大サイズを指定）
NUMBER（n , m）	最大 38 桁	数値型（n に最大桁数、m に少数桁数を指定）
BINARY_FLOAT	正の最大：3.40282E+38F 正の最小：1.17549E-38F	32 ビットの単精度浮動小数点数データ型（5 バイトの領域を使用）
BINARY_DOUBLE	正の最大： 1.79769313486231E+308 正の最小： 2.22507485850720E-308	64 ビットの倍精度浮動小数点数データ型（9 バイトの領域を使用）
DATE	紀元前 4712 年 1 月 1 日～ 紀元 9999 年 12 月 31 日	日付型（年/月/日/時/分/秒を格納し、7 バイトの領域を使用）
TIMESTAMP（n）		日付型（年/月/日/時/分/秒/ミリ秒を格納し、n には小数部の桁数を指定）
RAW	最大 2000 バイト	バイナリデータ
LONG RAW	最大 2 GB	バイナリデータ
BFILE	最大 4 GB	外部ファイル用のポインタを格納（ファイルの実態は、外部ファイルとして OS の管理下になるので DB によるリカバリは出来ない）
BLOB	最大 4 GB	バイナリデータ
LONG	最大 2 GB	可変長文字列型
CLOB	最大 4 GB	シングルバイト、マルチバイトキャラクタ
NCLOB	最大 4 GB	各国語キャラクタセットを使用する UNICODE データ
XMLType		XML 文書を CLOB として格納

4　制約の設定

制約とは、データ型の他に、列に入れるデータに制限や条件を追加する機能です。設定出来る制約は以下の 5 種類です。

制約名	説明
主キー制約（PRIMARY KEY）	一意制約と NOT NULL 制約の複合
一意制約（UNIQUE）	重複するフィールドを禁止する
NOT NULL 制約（NOT NULL）	NULL 値を禁止する ※NULL とは何も値が入っていない状態を指す（空文字列とは違いますので注意が必要）
チェック制約（CHECK）	入力出来る値に一定の制約を設ける
外部参照制約（REFERENCES）	他のテーブルの主キーまたは一意キーを参照する制約（参照先に存在しないデータを入れる事が出来なくなる）
DEFAULT 制約（DEFAULT）	初期値を設定する

また、制約は列に対しての列制約と表に対する表制約の 2 種類があります。列制約で指定する際は、各列の末尾に制約の指定をします。表制約をする場合は、全ての列の定義分の後に制約をカンマ「,」区切りで指定します。また、特徴として定義した制約に名前を付け制約オブジェクトとして操作することや列を複数指定することが出来ます。

以下に列制約の一覧を示します。

制約名	構文
主キー制約	項目名 属性 PRIMARY KEY ※この方法は、主キーが一項目だけの場合のみ使用可能
一意制約	項目名 属性 UNIQUE
NOT NULL 制約	項目名 属性 NOT NULL
チェック制約	項目名 属性 CHECK(正当条件) ※自分の項目に関する条件しか書けない
外部参照制約	項目名 属性 REFERENCES 外部テーブル名（一意キー）
DEFAULT 制約	項目名 属性 DEFAULT 初期値

次に、表制約の一覧を示します。

制約名	構文
主キー制約	CONSTRAINT 制約名 PRIMARY KEY (項目名, 項目名, …) ※複合主キーの設定が可能
一意制約	CONSTRAINT 制約名 UNIQUE (項目名, 項目名, …)
NOT NULL 制約	使用不可
チェック制約	CONSTRAINT 制約名 CHECK(正当条件)
外部参照制約	CONSTRAINT 制約名 FOREIGN KEY(項目名, 項目名, …) REFERENCES 外部テーブル名(一意キー, 一意キー, …)
DEFAULT 制約	使用不可

以下に使用例を示します。

（サンプルコード）列制約

```
CREATE TABLE Emp(
   emp_id        VARCHAR2(10) PRIMARY KEY,
   emp_name      VARCHAR2(10) NOT NULL,
   tel           VARCHAR2(10) UNIQUE,
   age           NUMBER(2) CHECK(age BETWEEN 18 AND 65),
   dept_id       CHAR(2) REFERENCES Dept(dept_id)
);
```

（サンプルコード）表制約

```
CREATE TABLE Emp(
   emp_id_1      VARCHAR2(5),
   emp_id_2      VARCHAR2(10),
   emp_name      VARCHAR2(10),
   tel           VARCHAR2(10),
   age           NUMBER(2),
   dept_id       CHAR(2),
   CONSTRAINT cons_p1 PRIMAEY KEY(emp_id_1, emp_id_2),
   CONSTRAINT cons_u1 UNIQUE(tel),
   CONSTRAINT cons_c1 CHECK(age BETWEEN 18 AND 65),
   CONSTRAINT cons_f1 FOREIGN KEY(dept_id)
                 REFERENCES   Dept(dept_id)
);
```

5　テーブルの作成

それでは実際にテーブルを作成してみましょう。

Emp テーブル

No	論理名称	物理名称	データ型	桁数	制約	備考
1	社員 ID	emp_id	NUMBER	5	PRIMARY KEY	
2	パスワード	emp_pass	VARCHAR2	10	NOT NULL	
3	社員名	emp_name	VARCHAR2	20	NOT NULL	
4	性別	gender	NUMBER	1	NOT NULL	"1"：男 "2"：女
5	住所	address	VARCHAR2	30		
6	誕生日	birthday	DATE			
7	部署 ID	dept_id	NUMBER	2	NOT NULL FOREIGN KEY	

Dept テーブル

No	論理名称	物理名称	データ型	桁数	制約	備考
1	部署 ID	dept_id	NUMBER	2	PRIMARY KEY	
2	部署名	dept_name	VARCHAR2	20	NOT NULL	"1"：総務部 "2"：営業部 "3"：経理部 "4"：資材部

（サンプルコード）CREATE TABLE 文

```
CREATE TABLE Dept(
   dept_id NUMBER(2) PRIMARY KEY,
   dept_name VARCHAR2(20) NOT NULL
);

CREATE TABLE Emp (
   emp_id NUMBER(5) PRIMARY KEY,
   emp_pass VARCHAR2(10) DEFAULT '9999' NOT NULL,
   emp_name VARCHAR2(20) NOT NULL,
   gender NUMBER(1) NOT NULL,
   address VARCHAR2(30),
   birthday DATE,
   dept_id NUMBER(2) NOT NULL,
   CONSTRAINT f1 FOREIGN KEY (dept_id) REFERENCES Dept(dept_id)
);
```

3. テーブルの変更

1　ALTER TABLE 文

テーブルの定義を変更したい場合は、ALTER TABLE 文を使います。ALTER は「変える」という意味です。

2　列の追加

テーブルに列を追加する場合は、以下の様に記述します。

（構文）列を追加する ALTER TABLE 文

```
ALTER TABLE <テーブル名> ADD <列の定義>;
```

なお、Oracle では、以下の様な記述で複数列を一度に変更する事も出来ます。

（構文）複数列を追加する ALTER TABLE 文

```
ALTER TABLE <テーブル名> ADD (<列の定義>,<列の定義>,……);
```

以下に列追加の例を示します。

（サンプルコード）列追加

```
ALTER TABLE Emp ADD mail VARCHAR2(100);
```

3　列の定義の変更

テーブルの列の定義を変更する場合は、以下の様に記述します。

（構文）列の定義を変更する ALTER TABLE 文

```
ALTER TABLE <テーブル名> MODIFY <列の定義>;
```

なお、Oracle では、以下の様な記述で複数列を一度に変更する事も出来ます。

（構文）複数列の定義を変更する ALTER TABLE 文

```
ALTER TABLE <テーブル名> MODIFY (<列の定義>,<列の定義>,……);
```

以下に列定義の変更の例を示します。

（サンプルコード）列定義の変更

```
ALTER TABLE Emp MODIFY mail VARCHAR2(200);
```

4　列名の変更

テーブルの列名を変更する場合には次のように記述します。

（構文）列名を変更する ALTER TABLE 文

```
ALTER TABLE <テーブル名> RENAME COLUMN <列名> TO <新列名>;
```

以下に列名変更の例を示します。

（サンプルコード）列名を変更

```
ALTER TABLE Emp RENAME COLUMN mail TO email;
```

5　テーブル名の変更

テーブルの名前を変更する場合には次のように記述します。

（構文）テーブル名を変更する ALTER TABLE 文

```
ALTER TABLE <テーブル名> RENAME TO <新テーブル名>;
```

以下にテーブル名変更の例を示します。

（サンプルコード）テーブル名を変更

```
ALTER TABLE Emp RENAME TO Employee;
```

6　列の削除

テーブルの列を削除する場合には次のように記述します。

（構文）列を削除する ALTER TABLE 文

```
ALTER TABLE <テーブル名> DROP (<列名>);
```

なお、Oracle では、以下の様な記述で複数列を一度に変更する事も出来ます。

（構文）複数列を削除する ALTER TABLE 文

```
ALTER TABLE <テーブル名> DROP (<列名>,<列名>,……);
```

以下に列削除の例を示します。

（サンプルコード）列削除

```
ALTER TABLE Employee DROP (email);
```

4. テーブルの削除

1　DROP TABLE 文

テーブルを削除する場合は、DROP TABLE 文を使用します。

（構文）DROP TABLE 文

```
DROP TABLE <テーブル名>;
```

以下に列削除の例を示します。

（サンプルコード）DROP TABLE 文

```
DROP TABLE Employee;
```