

リレーショナルデータベースには、さまざまな製品が存在します。大きく分類すると、商用製品とオープンソースの 2 種類です。商用製品では、Oracle Database や Microsoft SQL Server、Microsoft Access などがよく知られています。また、オープンソースのリレーショナルデータベースでは、MySQL や PostgreSQL、SQLite が有名です。

...

Oracle Database は、オラクルが発売しているデータベース管理システム (DBMS) で、後述する Microsoft SQL Server や 20 ページで紹介するオープンソース DBMS と市場を 3 分する、高いシェアを持つ製品です。最初のバージョンが登場したのは 1979 年と非常に長い歴史を持ち、DBMS 製品のデファクトスタンダードと言える存在になっています。

...

Microsoft SQL Server は、マイクロソフトから発売されている DBMS 製品で、Oracle Database に次ぐシェアを誇る製品です。Windows、および Windows Server に対応しています。

DBMS の多くはリレーショナル型ですが、リレーショナル型ではない DBMS ももちろんあります。たとえば、Web サーバーで有名な Apache ソフトウェア財団が開発／公開している CouchDB は、ドキュメントデータを Web で公開することに特化した DBMS です。操作には、SQL ではなく JavaScript ベースの JSON (JavaScript Object Notation) という言語を使います。同じようなドキュメント管理のためのデータベースとしては、MongoDB も有名です。CouchDB や MongoDB は、「クラウド時代の DBMS」ともいわれており、将来的にリレーショナルデータベースに代わる存在になるかもしれません。

SQL とは、リレーショナルデータベースを操作するために用いる言語の名称です。SQL は、一見すると何かの略語のようですが、**実際には略語ではなく**、「エスキューエル」と発音します。

...

SQL は IBM が 1974 年に開発した SEQUEL

(Structured English Query Language) という言語を元に開発されました。SEQUEL は、IBM がデータベース管理システム (DBMS) の研究のために作った System R に実装され、SEQUEL/2 にバージョンアップされています。後に、SEQUEL/2 が他社の商標であることがわかったため、名称を SQL に改めました。

現在、数多くの DBMS が存在しますが、SQL の基本的なコマンドは標準化されており、どの DBMS でもある程度同じものが利用可能です。ただし、それぞれの DBMS の機能の違いにより存在しないコマンドがあったり、オプション指定が異なる場合があるため、まったく同じとは言えません。

なお DBMS によっては、SQL 文の最後に必ずデリミタと呼ばれる記号を付ける必要があります。通常、デリミタには「;」（セミコロン）が使われますが、必要に応じて別の記号に変更することも可能です。本書でも、特にことわりのない限り、デリミタとして「;」を使用することにします。なお、Microsoft SQL Server や DB2 ではデリミタは不要です。

====Web====

デリミタ 【delimiter】

複数の要素を列挙する際に、要素の区切りとなる記号や特殊文字(の並び)のこと。例えば、CSV 形式のテキストデータでは、項目をカンマ(「,」)で区切って列挙するが、このカンマのことをデリミタという。デリミタにどんな文字を使うかはデータ形式や処理系によって決まっており、カンマやスペース、タブ、セミコロン、改行などが使われることが多い。

=====

連結演算子とは、文字型のデータに対して、その名の通り連結を行うための演算子です。標準 SQL では「||」が連結演算子として用いられますが、実際には DBMS 製品によって異なる記号が使われています。

使用される連結演算子

| | |
|-----------------------------------|---|
| 標準 SQL、Oracle Database、PostgreSQL | |
| Microsoft SQL Server、Access | + |

====Web====

'DOUBLE VERTICAL LINE' (U+2016)

|| 実線縦バーを 2 個

記号 [Unicode JIS X 0213 文字参照](#) 名称

| | | | | |
|--|--------|--------|----------|----|
| | U+2016 | 1-1-34 | ‖ | 双柱 |
| | | | ‖ | |

=====

約物（やくもの、英: punctuation mark）とは、文字組版など言語の記述に使用する記述記号類の総称で、句読点・疑問符・括弧・アクセントなどのこと。

=====

NULL データが何も入っていない状態

COLUMN

データベース設計に必須の作業 ——正規化

リレーショナルデータベースを構築するときに重要となる作業が「正規化」です。正規化は、DBMS でデータベースを作成したり、SQL コマンドを使ってテーブルを定義したり、といった作業の以前にやっておかなければなりません。

データベースにおける正規化をひと言で表すなら、「**重複する項目をなくす作業**」となるでしょう。正規化には、一般的に第1から第3までの3段階の工程があります。**第1正規化**は、同じ値が何度も出てくる列を、別のテーブルに独立させる作業です。**第2正規化**では、第1正規化で分離したテーブルから、主キー（84 ページ参照）が決まれば他のキーの値も決まる、という関係（部分従属）を探し、独立したテーブルに分離します。**第3正規化**は、ここまでに作成してきたテーブルから、キーにできるものをさらに独立したテーブルへと分離させる作業です。

データベースを構築する際には、設計の段階でこのような正規化の作業を行った後に、はじめてDBMSによって実際にデータベースを作成します。正規化の作業は、SQLとは直接関係がありませんが、リレーショナルデータベースを利用するならぜひ覚えておきたい、重要な作業工程である、といえるでしょう。

正規化を行うことによって、テーブルから重複する項目を別のテーブルに独立させることができるので、データベースから冗長なデータを排除して効率的なデータアクセスを実現したり、データの登録時や変更時に不整合が発生するのを防いだりすることが可能です。ただし、あまりにもテーブルが分割されすぎて、目的のデータを呼び出すために複雑な問い合わせ手順が必要になってしまう場合は、あえて正規化を行わない場合もあります。

ワイルドカード

% 任意の文字列

_ 任意の 1 文字

===Web===

メタ文字

【英】 metacharacter

メタ文字とは、その文字本来の意味とは異なり、プログラムで特別な意味を持たせた文字のことである。

例えば、正規表現では「+」は「足す」ではなく 1 回以上の連続文字を意味するメタ文字である。正規表現による検索では、A+B で検索すれば、AB、AAB、AAAB などのいずれとも一致する。文字として「+」を表現したい場合は、メタ文字としての意味を打ち消すために (エスケープとも言う) 「¥+」と¥を付ける。「¥」自身は「¥¥」と書く。ちなみに、「¥」は Windows の世界では、ファイルの位置を表すためのパス区切りの意味で使用されることがある。

メタ文字は、それぞれの環境や機能の中で任意に意味が決められるものであるため、当の環境において各メタ文字は何を意味するのかを、あらかじめ把握しておく必要がある。

=====

===Web===

別名 (エイリアス)

エイリアス

alias 《コ》 [Mac OS が提供する機能で、別のオブジェクトへのリンク機能だけを持つファイル。Microsoft 社の Windows のショートカット (shortcut) に相当する。]

=====

「正規表現」

より複雑な条件で精密に文字列を検索したい場合にしばしば用いられるのが、正規表現という手法です。正規表現とは、パターンマッチングの方法の 1 つで、高機能なテキストエディタなどに搭載されています。正規表現ではさまざまなメタ文字が存在し、より複雑なパターンマッチング条件の指定が可能です。MySQL と PostgreSQL は、演算子としてこのような正規表現を利用することができます。

膨大な正規表現のすべてをここで紹介することはできないので、以降では基本的な使い方のみ説明しておきます。たとえば「^」は文字列の先頭で、「\$」は文字列の末尾です。

COLUMN

SQL 開発環境を使ってみよう

SQL を勉強する場合、コマンドプロンプトなどで DBMS に接続し、実際に SQL コマンドでテーブルを作成したり、問い合わせを行ったりする方法が一般的です。しかし、コマンド操作に慣れていない人にとっては、ディレクトリ移動などの OS 操作もコマンドで行わなければならない、わずらわしく感じられるかもしれません。

そのような人にお勧めなのが SQL 開発環境です。現在、さまざまな SQL 開発環境が存在しており、中には無料のものもあります。これらは、テーブルの作成や列の定義を GUI で実行できたり、SQL コマンドの入力補完や充実したヘルプが用意されているなど、SQL の学習にうってつけです。ODBC（Microsoft が提唱する、データベースと外部プログラムを接続する方法）によってデータベースに接続し、実際に SQL コマンドを実行できるものもあるので、SQL の勉強だけでなく実際のデータベース開発にも活用できます。SQL 開発環境自体の使い方を覚える必要はありますが、**無料のものなら手軽に試せるので、ぜひ一度使ってみてください。**

SQL 開発環境ほど豊富な機能はありませんが、一部のテキストエディタではプラグインなどで SQL 支援機能を利用可能です。テキストエディタの SQL 支援機能では、SQL コマンドが自動的に色分けされたり、かつこの不整合（不足や重複など）を教えてくれたりします。たとえば、UNIX 系 OS でよく使われる Emacs には SQL モードが用意されており、SQL コマンドを効率よく編集できます。また、Emacs 内で SQL クライアントを起動して SQL コマンドを実行することもできます。

交差結合

3 行 2 列と 4 行 3 列→12 行 5 列

交差結合

```
SELECT * FROM テーブルA, テーブルB;
```

テーブルA

| | |
|----|----|
| A1 | A2 |
| A3 | A4 |
| A5 | A6 |

テーブルB

| | | |
|-----|-----|-----|
| B1 | B2 | B3 |
| B4 | B5 | B6 |
| B7 | B8 | B9 |
| B10 | B11 | B12 |

すべての行の組み合わせが作られる

| | | | | |
|----|----|-----|-----|-----|
| A1 | A2 | B1 | B2 | B3 |
| A1 | A2 | B4 | B5 | B6 |
| A1 | A2 | B7 | B8 | B9 |
| A1 | A2 | B10 | B11 | B12 |
| A3 | A4 | B1 | B2 | B3 |
| A3 | A4 | B4 | B5 | B6 |
| A3 | A4 | B7 | B8 | B9 |
| A3 | A4 | B10 | B11 | B12 |
| A5 | A6 | B1 | B2 | B3 |
| A5 | A6 | B4 | B5 | B6 |
| A5 | A6 | B7 | B8 | B9 |
| A5 | A6 | B10 | B11 | B12 |

テーブルAとテーブルBの
すべての行を組み合わせる



結合後の行数=テーブルAの行数×テーブルBの行数
結合後の列数=テーブルAの列数+テーブルBの列数

COLUMN

SQLインジェクションに注意しよう

本書の中でも何度も述べているように、近年は Web サービスとデータベースとの連携が盛んに行われています。その際に問題となるのが「SQL インジェクション」です。データベースと連携する Web サーバーを構築するなら、SQL インジェクション対策が必須となります。

SQL インジェクションとは、Web サービスの実装の不備を狙って Web サーバーに SQL コマンドを送り込み、不正にデータベースから情報を引き出す、という攻撃手法や脆弱性を意味する言葉です。SQL インジェクションの脆弱性を持つ Web サイトでは、パスワードやクレジットカード情報などを盗み出されてしまう危険があります。

バックエンドでデータベースを利用している Web サービスでは、PHP などのスクリプト言語を使って、SQL コマンドを発行する方法が一般的です。スクリプト言語を使うことによって、ユーザーのリクエストに応じて動的に SQL コマンドを生成できますが、攻撃者はこのしくみを突いて、不正なコマンドを送り込もうとします。

SQL インジェクションによる攻撃を防ぐには、Web サービスを構成する Web アプリケーション側と、データベース側との両面での対策が必要です。Web アプリケーションでの対策としては、不正な SQL コマンドを送り込まれないように、ユーザーが入力したデータから攻撃に利用されそうな特殊文字を取り除く、といった作業が必須です。一方、データベースにおいては、Web サービス経由で利用するユーザーに過度の

権限を与えないように、ユーザー権限の厳密な設定が必要です。IPA（独立行政法人・情報処理推進機構）が公開している、「安全な SQL の呼び出し方」（http://www.ipa.go.jp/Security/vuln/documents/Website_Security_sql.pdf）という文書も参考になるでしょう。

トランザクションとは、データベースに対する一連の処理をまとめた単位のことです。多くの DBMS では、トランザクション単位で操作内容を記録することができ、必要に応じてデータベースをトランザクション単位で以前の状態に戻すことができます。

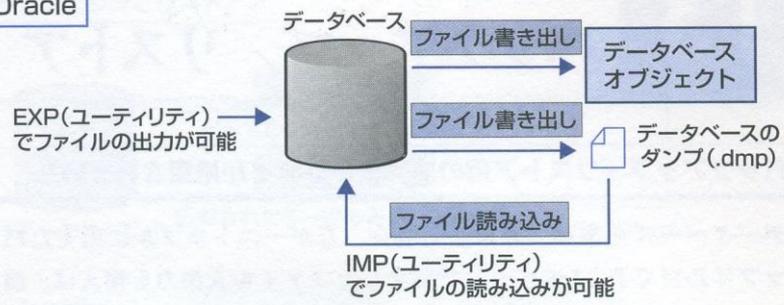
Oracle Database や Microsoft SQL Server は、標準状態でトランザクションが実行されています。MySQL や PostgreSQL、Access では、手動でトランザクションを開始する必要があります（166 ページ参照）。なお MySQL の場合は、InnoDB などトランザクションに対応しているストレージエンジンを使っている場合のみ、トランザクションを利用可能です。

トランザクションが実行されている状態では、操作内容をデータベースに反映するためには、コミットという操作を行わなければなりません（166 ページ参照）。コミット操作を行うことによってトランザクション処理が確定し、**操作内容がデータベースに反映されます**（コミットに関しては、DBMS によって動作内容が異なります。くわしくは 166 ページを参照してください）。

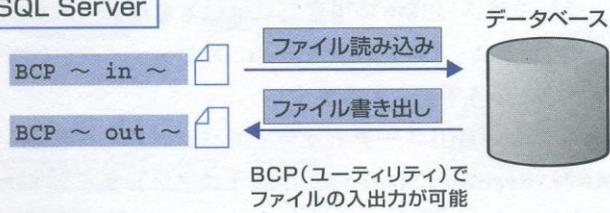
このように、トランザクション単位で操作を行えば、データベース操作中にトラブルが発生し、操作が不正に中断してしまった場合でも、中途半端な操作内容がデータベースに反映されてしまうことはありません。また最初に述べたように、トランザクション単位でデータベースを操作前の状態に戻すことも可能です（ロールバック、166 ページ参照）。

各DBMSのファイル入出力方法

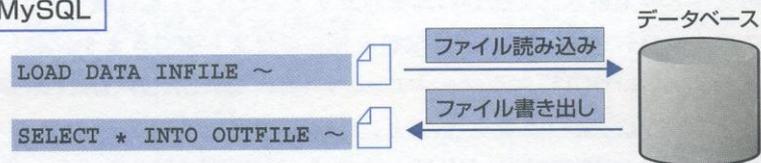
Oracle



SQL Server



MySQL



PostgreSQL

