Python
データベース プログラミング 入門

日向俊二◉著

CUTT
カットシステム

■**サンプルファイルのダウンロードについて**

本書掲載のサンプルファイルは、一部を除いてインターネット上のダウンロードサービスからダウンロードすることができます。詳しい手順については、本書の巻末にある袋とじの内容をご覧ください。

なお、ダウンロードサービスのご利用にはユーザー登録と袋とじ内に記されている番号が必要です。そのため、本書を中古書店から購入されたり、他者から貸与、譲渡された場合にはサービスをご利用いただけないことがあります。あらかじめご承知おきください。

・本書の内容についてのご意見、ご質問は、お名前、ご連絡先を明記のうえ、小社出版部宛文書（郵送またはE-mail）でお送りください。
・電話によるお問い合わせはお受けできません。
・本書の解説範囲を越える内容のご質問や、本書の内容と無関係なご質問にはお答えできません。
・匿名のフリーメールアドレスからのお問い合わせには返信しかねます。

はじめに

いつの時代も、データベースはとても重要な技術です。データベースはあらゆる情報を扱うソフトウェアの中心技術の一つといっても過言ではないでしょう。たとえば、ウェブサイトの構築に近ごろよく使われている WordPress にもデータベースが使われています。航空機の運航管理からスーパーマーケットの在庫管理まで、さまざまなシステムの一定の構造を持つ情報は、ほぼすべてがデータベースに保存されています。

本書では、データベースを初めて使う読者を対象に、データベースについての基本的なことから、データベースへの作成や接続、テーブルの作成、データの登録や更新、削除など一連の操作について解説します。また、応用としてデータベースを扱う GUI アプリの作り方を紹介します。

現代の主なデータベースは、多くの場合 SQL という言語を使って操作します。一方、プログラムでデータベースを操作して何らかの機能を実現するためのプログラミング言語には、さまざまな言語が使われます。本書ではプログラミング言語として理解しやすい Python を使って解説しますが、本書で Python を使ったデータベースの概念と操作方法を理解してしまえば、他のプログラミング言語に応用するのは容易です。

本書を活用することで、データベースを完璧に使いこなせるようになる日がまもなく来ることでしょう。

2019 年 1 月 著者記す

■本書の表記

⤵	紙面への印刷の都合で改行していますが、実際には改行しないで 1 行で続けて入力することを表します。
[...]	書式の説明において、[と] で囲んだものは省略可能であることを示します。
［X］	キーボードのキーを押すことを示します。たとえば、［F5］は F5 キーを押すことを意味します。
［S］+［X］	キーボードの S キーを押したまま X キー押すことを示します。［Ctrl］+［F5］は Ctrl キーを押したまま F5 キーを押すことを意味します。
>	OS のコマンドプロンプトを表します。Linux など UNIX 系 OS の場合は $ などで表されますが、本書では > に揃えますので適宜読みかえてください。
>>>	Python のインタープリタ（インタラクティブシェル）のプロンプトを表します。
mysql>	MySQL のコマンドラインツールのプロンプトを表します。

本文を補足するような説明や、知っておくとよい話題です。

特に重要なことです。

COLUMN 特定の項目についてのある程度まとまった補足説明です。

一般的に SQL のコマンドやキーワードなどは大文字／小文字が区別されませんが、本書では、原則として、SQL のコマンドやキーワードは大文字で、Python のキーワードや変数などは小文字で表記します。データベースファイル名やテーブル名などは先頭を大文字にした小文字で表記します。

■注意事項

- 本書の内容は執筆時点の状態で記述しています。Python の最新のバージョンは 3.7.1、SQLite の最新バージョンは 3.25、MySQL の最新バージョンは MySQL 8.0 です。将来、バージョンが変わるなど、何らかの理由で記述と実際とが異なる結果となる可能性があります。
- 本書のプログラムは Python 3 を前提としています。本書の多くの例は Python 2 でも実

行できますが、Python 2 と Python 3 では書式が異なります（Python 2 では、特に引数を囲む括弧が不要です）。

- 本書は Python や SQLite および MySQL のすべてのことについて完全に解説するものではありません。必要に応じて適切なドキュメントなどを参照してください。
- 本書のサンプルは、プログラミングを理解するために掲載するものです。実用的なアプリとして提供するものではありませんので、ユーザーのエラーへの対処やその他の面で省略してあるところがあります。

■本書に関するお問い合わせについて

本書に関するお問い合わせは、sales@cutt.co.jp にメールでご連絡ください。

お問い合わせは本書に記述されている範囲に限らせていただきます。特定の環境や特定の目的に対するお問い合わせ等にはお答えできませんので、あらかじめご了承ください。

また、例として掲載した断片的なコードをやみくもに入力しても動作しません。あるコードを実行するためには、必要なモジュールをインポートしたり、システムの状況に応じてデータベースを準備したり、データベースに接続したりする必要があります。必ずそれまでの説明を良く読んで理解してから、必要な準備を行ったうえでコードを実行してください。「本書の記述通りにコードを打ち込んだがエラーになった」などのご質問にはお答えいたしかねます。

お問い合わせの際には下記事項を明記してくださいますようお願いいたします。

- 氏名
- 連絡先メールアドレス
- 書名
- 記載ページ
- 問い合わせ内容
- 実行環境

目次

第１章

データベースの基礎知識

この章では、データベースの概念や基本的な要素などについて説明します。

1.1 データベースの基礎

データベース（Database）は、多数のデータを一定の構造で保持し、管理するためのものです。

■ データベース

「データベース」という言葉は、「たくさんのデータを集めたもの」と解釈されることがあります。また、「データベース」という言葉がデータを集めた場所のことを指す場合もあります。つまり、単に「データベース」というときには、そこに蓄積されたデータのことだけを考えることがあります。

しかし、プログラミングの世界では、一般にデータベースと呼ぶものの実態は、データを組織的に管理するソフトウェアと一連の一定の構造を持ったデータ全体を指します。ここで、データを組織的に管理するソフトウェアとは、データを登録したり検索したりするための基本的なソフトウェアのことです。また、このときのデータとは、無秩序な情報の集積ではなく、特定の構造を持つ特定の種類の情報を指します。データベースのデータは、無秩序な情報の集まりではなく、必ず一定の構造を持つものとして扱うことができる特定の種類のデータであるという点は重要です。

データベースはたくさんのデータを保存して管理するものであると考えられがちですが、雑多なデータをただ集めただけのものはデータベースとはいいません。データベースのデータは特定の種類のデータで一定の構造を持ちます。

一定の構造を持つように整理できるデータのほとんどは、データベースシステムに保存されます。住所録や不動産の情報のような台帳に記載するようなデータはもちろん、それ以外のデータも整理されてデータベース保存されることがよくあります。たとえば、ブログやいわゆるホームページ（ウェブサイト）の構築に近ごろよく使われている WordPress にも、データベースが使われていて、コンテンツデータ（記事のタイトルや内容など）をデータベースに保管する仕組みになっています。また、航空機の運航管理からスーパーマーケットの在庫管理ま

で、世の中にあるさまざまなシステムの一定の構造を持つデータは、ほぼすべてがデータベースに保存されています。

■ データベース管理システム

データを登録したり検索したりするソフトウェアを、データベース管理システム（DataBase Management System、DBMS）といいます。

DBMS は、データを効率よく操作するためのソフトウェアです。DBMS は Python や C# のようなプログラミング言語を使って活用することができますが、多くの場合、それ自身のユーザーインターフェースを備えていて、それ自身を使ってデータベースを操作することができます。たとえば、DBMS のコマンドを使って、データを登録したり、削除したり、検索したり、集計することができます。この場合、ユーザーはデータベースの構造や DBMS のコマンドの使い方を知っていなければなりません。

なお、データベースというオブジェクト（もの）は、すでに説明したように、ある種のデータが一定の構造で集められているもののことです。また、データベースという言葉が DBMS を指すこともあります。さらに、データベースという言葉が、DBMS とデータを含む全体を指す言葉として使われることもよくあります。

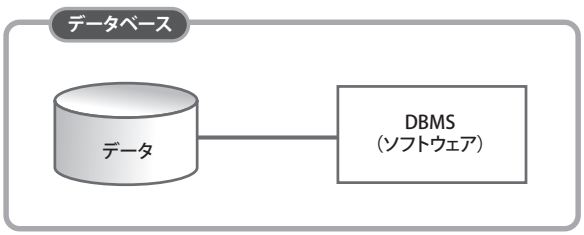

図 1.1 ●データベース（データと DBMS）

■ データベースとデータベースアプリ

データベース管理システム（DBMS）は、アプリ（アプリケーション）やウェブサイトのプログラムから利用して、データベースのデータを利用できるようにすることができます（図 1.2）。アプリからデータベースを利用するようにするときには、一般的には、アプリのプログ

ラミング言語を介して DBMS のコマンドを使ってデータベースを利用します。

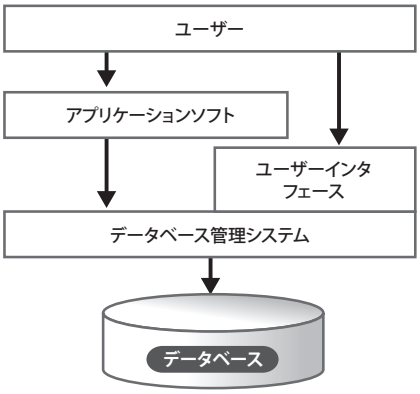

図 1.2 ●ユーザーとデータベース

1.2 データベースの構造

データベースとは、多数のデータを一定の構造で保持するものを指します。多数のデータを無秩序に集めたものはデータベースとはいいません。

■ データベースの構造

データベースは、一般に、テーブル（Table、表）で構成されています。テーブルは、フィールド（Field。カラム、列、項目ともいう）とレコード（Record。行ともいう）で構成されています。つまり、データベースのテーブルは表の形式でデータを保存するものとみなすことができます。

実際には、データベースのデータは、ディスクファイルのような保存媒体に表の形で保存されているわけではありません。しかし、データベースを扱うときにはテーブルをイメージすると理解しやすくなります。

データベースの最小のデータ単位はフィールドです。複数のフィールドで、一つのレコードを構成します。レコードは基本的なアクセス単位です。

複数のレコードをまとめたものがテーブルです（図 1.3）。

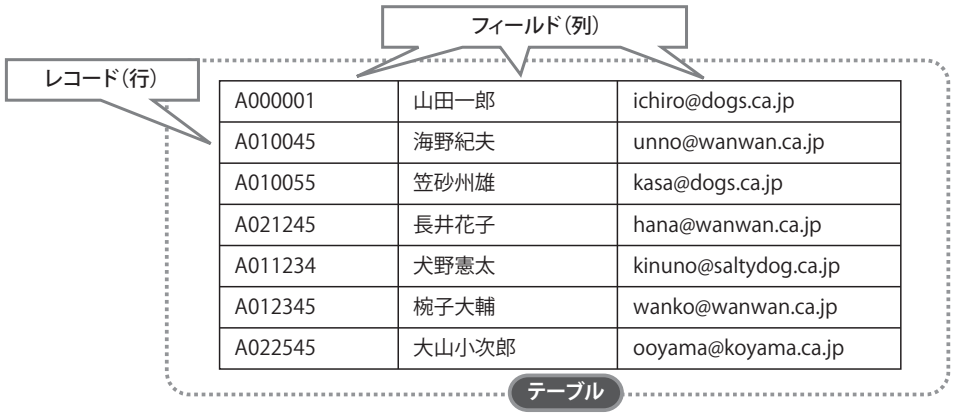

図 1.3 ●テーブル、レコード、フィールド

一般的には、一つのデータベースには、複数のテーブルを保存することができます。

データベース管理システムは、このような構造をベースにして、レコードやフィールドの検索、並べ替え、再結合などの一連の操作を行うことができるようにします。

データを表（テーブル）形式ではなく、オブジェクトとして保存するデータベースを、オブジェクトデータベースと呼びます。いくつかのシステムに分散させたデータベースを分散データベースといいます。

■カレントレコード

データベースで、現在参照しているレコードをカレントレコード（Current Record）といいます。カレントレコードはデータベースプログラミングで重要な概念です。

データベースで、現在参照しているレコードを指すオブジェクトをカーソル（Cursor）といいます。

つまりカーソルが指している位置のレコードが現在操作の対象としているレコードであり、カーソルを移動することによって操作の対象とするレコードを変更することができます。

現在のレコードを指すカーソルは、第 2 章以降の実際のデータベース操作で頻繁に使います。なお、SQL データベースでは、カーソルを「検索結果からデータを 1 件ずつ処理するための仕組み」とみなすことがあります。

■キー

データベースの特定のレコードを識別するフィールドデータをキー（Key）といいます。テーブルの中のある 1 個のレコードを明確に識別するためには、原則として、主キー（プライマリーキー、Primary Key）が必要です。主キーはレコードを明確に区別するために使われるので、主キーの値はレコードごとに異なっていなくてはならず、重複してはなりません。

例外的に、主キーに相当する値を重複できるようにしたデータベースもあります。そのようなデータベースではレコードの番号などで個々のレコードを識別します。

一般的には、主キーのほかにテーブルに複数のキーを定義することができます。主キー以外のキーは、ほとんどの場合、重複できます（詳細はデータベースによって異なります）。

キーとなるデータのフィールドを、キーフィールド（Key Field）といいます。

■フィールドのデータ型

データベースには、数値や文字列などを保存することができますが、一つのフィールドの型は一定でなければなりません。たとえば、ID のフィールドを数値で定義したら、ID のフィールドの値はすべて数値でなければなりません。一つのフィールドに数値と文字列のような異なるデータ型を混在させることはできません。たとえば、ID に「A0123」のようなアルファベット文字を含む表現を使いたい場合には、その ID のフィールドは、文字と数値が混在したフィールドではなく、文字列のフィールドとして定義します。

データをバイナリデータ（2 進数値の並び）として保存することができるデータベースでは、一つのフィールドに任意の型のデータを保存することができます。しかし、初歩のうちはこのようなデータフィールドは特殊なものであると考えてください。

■リレーショナルデータベース

データを一定の構造に整理して複数のテーブル形式で保存し、テーブル相互に関連性を持たせたデータベースを、リレーショナルデータベース（Relational DataBase）といいます。

リレーショナルデータベースの基本的な機能を提供するソフトウェアをリレーショナルデータベース管理システム（Relational DataBase Management System、RDBMS）といいます。

たとえば、売り上げテーブルと顧客テーブルからなる販売管理データベースでは、売り上げテーブルの売り上げ先の顧客 ID と顧客テーブルの顧客 ID との間に関連性を持たせます。このような関連性をリレーションシップ（Relationship）またはリレーション（Relation）といいます。

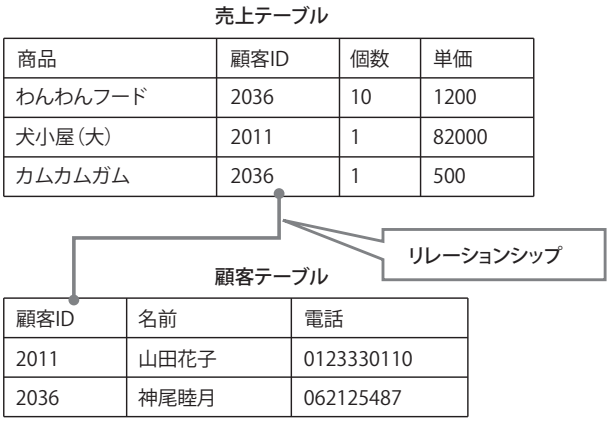

図 1.4 ●リレーションシップ

関連付けるテーブルは 2 個以上、いくつでもかまいません。大規模なデータベースでは多数のテーブルを定義して、相互に関連付けを行います。このテーブルの定義とテーブル間の関連付けを考えることは、データベースの設計の重要な仕事の一つです。

リレーショナルデータベースでは、同じデータを複数のテーブルに持たないようにしたり、さまざまな種類のデータを別のテーブルに分けて管理することで、データアクセスや検索などの効率を良くしたり、プログラムの生産性を高めることができます。図 1.4 の例では、売り上げテーブルに顧客の名前を登録する代わりに顧客 ID を登録することで、具体的な「名前」というデータを複数のテーブルに持たせないようにしています。

データベースで、同じ情報を重複して保存すること（冗長性）を排除することを正規化（Normalization）といいます。

リレーションシップは、データベース内部の実際のテーブル間の特定のフィールドの関連付け情報として定義することも、単なる論理的な関連付けとして定義することもできます。多くのリレーショナルデータベースは、リレーションシップの情報をデータベースに保存することができます。

■ データベースの完全性

データベースのデータは、内容に矛盾がなく、利用する際に問題が発生しないようにしなければなりません。このことは特にリレーショナルデータベースで重要で、関連するテーブル間のデータで過不足や不一致がないようにしなければなりません。たとえば、商品の売り上げ情報を保存する売り上げテーブルと購入者の情報を保存する顧客テーブルがある販売管理データベースで、顧客テーブルの顧客を削除したら、それに関連する売り上げ情報も削除しなければなりません。そうしないと、売った先が不明である売り上げが発生してしまいます。

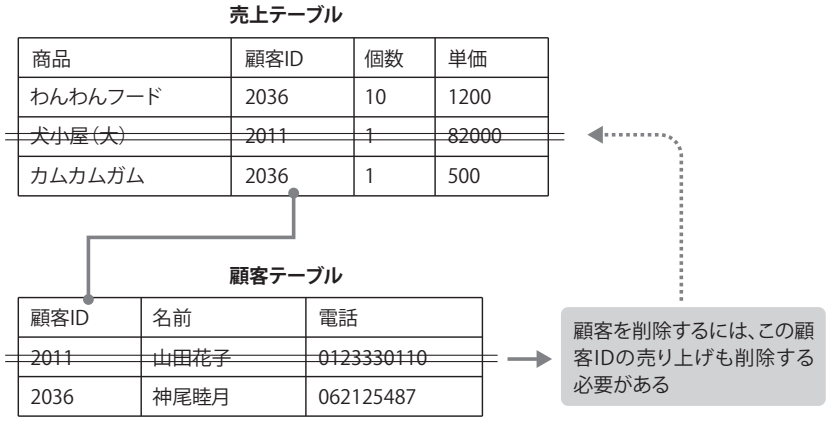

図 1.5 ●データベースの整合

1.3 SQL

現代の主なデータベースは、多くの場合 SQL という言語を使って操作します。

■ SQL

リレーショナルデータベース管理システムで、データの操作や定義を行うための言語を、データベース問い合わせ言語（DataBase Query Language）といいます。最も普及している問い合わせ言語は、SQL（Structured Query Language）です。SQL は ANSI（後に ISO）で言語仕様の標準化が行われており、制定された年ごとに SQL86、SQL89、SQL92、SQL99 などの規格があります。

SQL は標準化が進められていて、現在では多くの SQL が SQL99 に準拠するように作られていますが、どの SQL データベースでもすべて同じであるわけではありません。後の章で説明する SQLite と MySQL でも、同じ目的に対する SQL の命令文はほぼ同じですが、細かい点で異なる部分があります。

SQL では、データの登録や検索などのためのコマンドは、コマンド文字列で指定します。これを SQL 文または SQL クエリまたは SQL クエリーといいます。

SQL コマンドで行うデータベースでの要求を SQL クエリと呼ぶこともあります。

たとえば、次の SQL 文（SQL のコマンド）は、4 文字の顧客のコード（id）と 12 文字の顧客の名前（name）がある customer というテーブルを作成します。

```
CREATE TABLE customer(id CHAR(4) PRIMARY KEY, name CHAR(12));
```

このテーブルの主キー（PRIMARY KEY）は顧客のコード（id）です。

SQL コマンドは長くなる場合があるため、複数行で記述したり入力できます。そのため、論理的な行の最後を明示するために、論理的な行の最後（SQL 文の最後）にセミコロン（;）を付ける場合があります。SQL データベースの種類によってこれは必須である場合と任意である場合があります。本書で説明している SQLite では最後のセミコロンは必須ではありませんが、第 7 章で説明する MySQL では必須です。

次のコマンドは customer というテーブルから、顧客の名前（name）を取り出す SQL 文の例です。

```
SELECT name FROM customer;
```

ここで紹介している SQL コマンドについて、この段階で具体的に理解したり覚える必要はありません。第 2 章以降、本書全体を通じて SQL コマンドについて具体的な例を通して学びます。ここでは SQL コマンドが比較的単純な英単語を使った短い文である点に注目してください。

SQL のコマンドは、以下 3 種類に分類されます。

- データ定義言語（Data Definition Language; DDL）
- データ操作言語（Data Manipulation Language; DML）
- データ制御言語（Data Control Language; DCL）

データ定義の一般的なコマンドは次の通りです。なお、データベースオブジェクトとは、データベースのテーブル、インデックス、制約などを指します。

表 1.1 ●データ定義のコマンド

コマンド	機能
CREATE	データベースオブジェクトを作成（定義）する
DROP	データベースオブジェクトを削除する
ALTER	データベースオブジェクトを変更する

データ操作の一般的なコマンドは次の通りです。

表 1.2 ●データ操作のコマンド

コマンド	機能
INSERT INTO	行データまたはテーブルデータを挿入する
UPDATE ～ SET	テーブルを更新する
DELETE FROM	テーブルからレコードを削除する
SELECT ～ FROM ～ WHERE	テーブルデータを検索する。結果集合を取り出す

データ制御の一般的なコマンドは次の通りです。

表 1.3 ●データ制御のコマンド

コマンド	機能
GRANT	データベース利用者に特定の作業を行う権限を与える
REVOKE	データベース利用者から権限を剥奪する
SET TRANSACTION	トランザクションモードを設定する
BEGIN	トランザクションを開始する
COMMIT	トランザクションを実施（確定）する
ROLLBACK	トランザクションを取り消す
SAVEPOINT	ロールバック地点を設定する
LOCK	テーブルなどの資源を占有する

データベースによっては、その他にもコマンドが用意されている場合があります。またすべての DBMS がこれらのコマンドをすべて装備しているわけではありません。

SQL コマンドは論理的に 1 行の文として記述します（改行があってもかまいません）。SQL コマンドの文の例を表に示します。

表 1.4 ● SQL コマンドの文の例

作業	コマンド
テーブルを作成する	CREATE TABLE table (field type、field type,,,);
テーブルのデータを取得する	SELECT field FROM table WHERE cnd;
テーブルをコピーする	SELECT * INTO toTable FROM fromTable;