



データベース入門

目次

1. はじめに	2
1.1. データベースとは	2
1.2. RDB(リレーショナルデータベース)とは	3
1.3. DBMS とは	4
1.4. 各研修とのつながり	4
2. 環境構築	5
2.1. SQLITE のダウンロード	5
2.2. ZIP ファイルの解凍	5
2.3. コマンドプロンプトの起動	6
2.4. SQLITE の起動	7
2.5. (参考)SQLITE の終了	8
3. SQL 基礎	9
3.1. SQL とは	9
3.2. テーブルの作成	10
3.3. CRUD 文とは	11
3.4. INSERT 文	12
3.5. SELECT 文	14
3.6. UPDATE 文	16
3.7. DELETE 文	18
4. SQL 応用	20
4.1. 結合	20
4.2. 集計	23
5. ワークの解答	25

1. はじめに

「データベース」と聞いて、データのかたまりのような、漠然としたイメージが沸くのではないのでしょうか。実はただデータを集めただけでは、データベースとは呼べません。

- ・ データベースとは何か
- ・ 主流のデータ管理方法 RDB とは

について解説します。

1.1. データベースとは

データベースとは、整理された情報の集まりのことです。

「整理された情報の集まり」であることを踏まえ、以下のワークにもトライしてみましょう。

ワーク 1

あなたの身の回りで、データベースが使われているものをできるだけたくさん洗い出してみましょう。

解答例:

()内はデータベースの例

- ・ 家計簿アプリ(入出金データなど)
- ・ お掃除ロボット(火曜の9時に起動、などのスケジュールデータなど)
- ・ スマホゲーム(購入履歴、レベル、友だちデータなど)
- ・ 格安 SIM の管理サーバ(契約内容、データ使用量など)

スマホアプリや SNS などが、すぐに思いついたのではないのでしょうか。データベースが使われているのは、ソフトだけではなくあります。

お掃除ロボットなどの機器に組み込まれていたり、管理サーバなどのインフラにも活用されているなどあらゆる場面で、縁の下の力持ちとして私達の生活を支えているのです。

コラム データベースの由来は米軍基地

データベースの由来は 1950 年頃の米軍基地です。

資料を一箇所に集約し、情報の整理の効率化を図りました。

この基地を Data Base(情報基地)と呼んでいたとのこと。これがデータベースの起源とされています。

1.2. RDB(リレーショナルデータベース)とは

データベースには、管理方法によっていくつかの種類があります。

この講義内では、主流である RDB(リレーショナルデータベース)について説明します。

RDB では、Excel のように、表上でデータを管理します。

この表のことをテーブル、縦軸をカラム(列)、横軸をレコード(行)と呼びます。

リレーショナルデータベースにおいて、データベースとはテーブルの集まりのことであり、テーブルとはカラムとレコードの集まりのことを指します。

テーブル				
カラム				
社員 ID	氏名	年齢	住所	...
1	田中太郎	23	東京都千代田区	...
2	鈴木次郎	25	千葉県千葉市	...
3	渡辺花子	...	東京都台東区	...
4

1.3. DBMS とは

RDB の形式でデータを管理するシステムのことを、RDBMS(リレーショナルデータベースマネジメントシステム)と呼びます。

RDBMS の代表例として、以下のシステムがあります。

【代表的な RDBMS】

名前	特徴
Oracle	商用 RDBMS として、世界で最も多く使われている
Microsoft SQL Server	Microsoft 社の商用 RDBMS
PostgreSQL	オープンソース RDBMS。日本では特に人気がある
MySQL	世界で最も有名なオープンソース RDBMS
SQLite	名前のとおり簡易的に利用できるオープンソース RDBMS

この講義内では、ソフトのインストール等が必要なく、簡単に利用できる SQLite を扱います。

コラム RDBMS 以外=NoSQL

RDBMS 以外のデータベース管理システムを、総称で NoSQL と呼びます。

RDBMS と比べ、大量のデータを処理するのに向いていること・拡張しやすいことが特徴です。

ソーシャルゲームで活用されるなど、近年注目を集めています。

1.4. 各研修とのつながり

1.4.1 Java 研修

Java 研修では、RDB を利用したアプリケーションを作成していきます。

1.4.2 組込み研修

組込み研修では、組み込みシステム特有のデータベースについて学んでいく必要があります。

1.4.3 インフラ研修

インフラ研修では、AWS 上にデータベースサーバーを構築します。

2. 環境構築

まずは、今回使用するソフトである SQLite をダウンロードし、解凍します。

2.1. SQLite のダウンロード

<https://www.sqlite.org/download.html> にアクセスし、「Precompiled Binaries for Windows」内の「sqlite-tools-win32-x86-XXXXXXX.zip」(後半の XXX…部分は、337 などのバージョンの数値が入ります)をクリックし、Zip ファイルをダウンロードします。

SQLite Download Page

Pre-release Snapshots

Source Code

Documentation

Precompiled Binaries for Android

Precompiled Binaries for Linux

Precompiled Binaries for Mac OS X (x86)

Precompiled Binaries for Windows

Universal Windows Platform

2.2. Zip ファイルの解凍

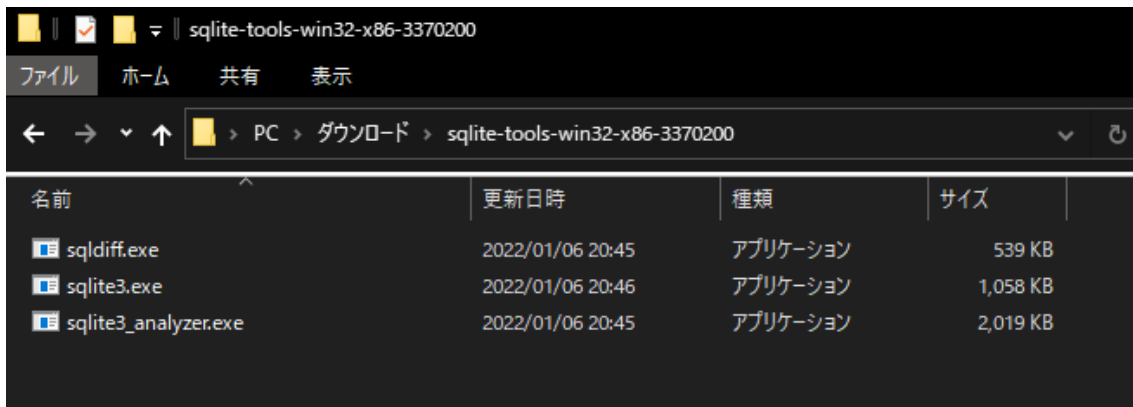
ダウンロードフォルダ内にある「sqlite-tools-win32-x86-XXXXXXX.zip」を解凍します。

解凍した「sqlite-tools-win32-x86-XXXXXXX」フォルダ内のファイルを確認してください。

以下 3 つの exe ファイルがあれば、問題なく解凍できています。

- sqldiff.exe
- sqlite3.exe
- sqlite3_analyzer.exe

※2.4 にてこのフォルダを利用するため、閉じないでください

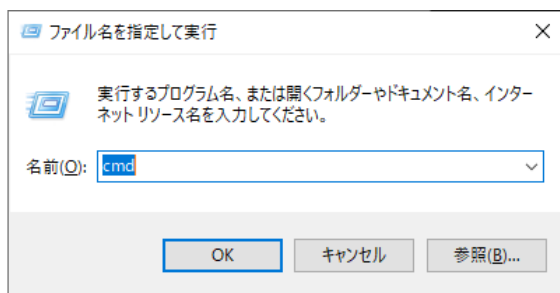


2.3. コマンドプロンプトの起動

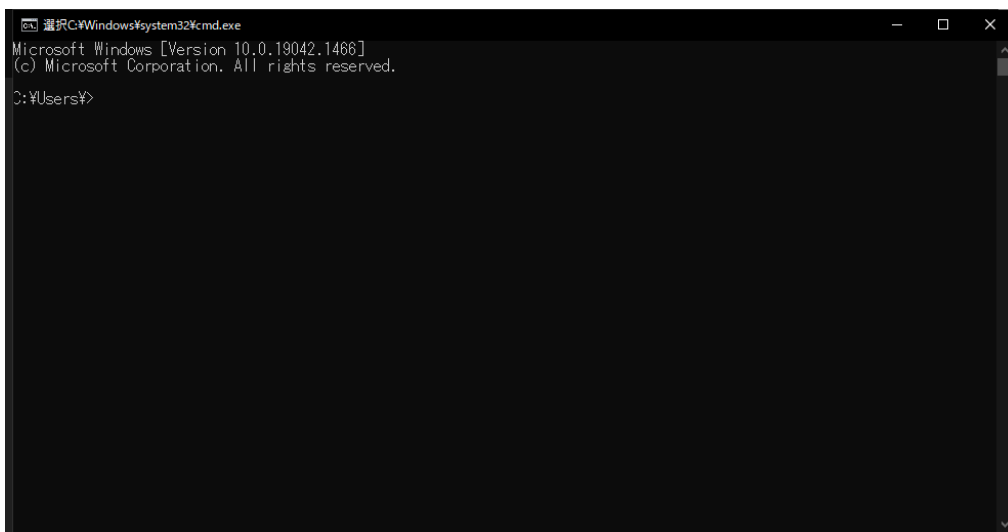
続いて、コマンドプロンプトを起動します。

Windows ボタンと R を同時に押して「ファイル名を検索して実行」を開きます。

cmd と入力し「OK」を押してください。



コマンドプロンプトが起動します。

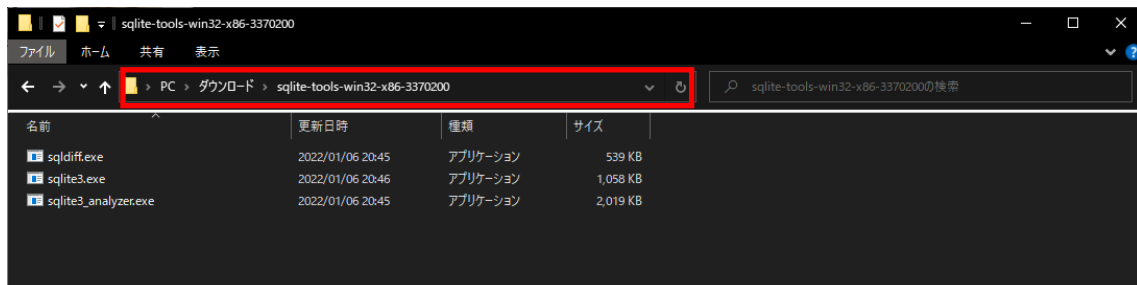


2.4. SQLite の起動

コマンドプロンプトから SQLite を起動します。

2.2 にて解凍したフォルダを開きます。

フォルダパス部分をクリックし、右クリック「コピー」を行ってください。

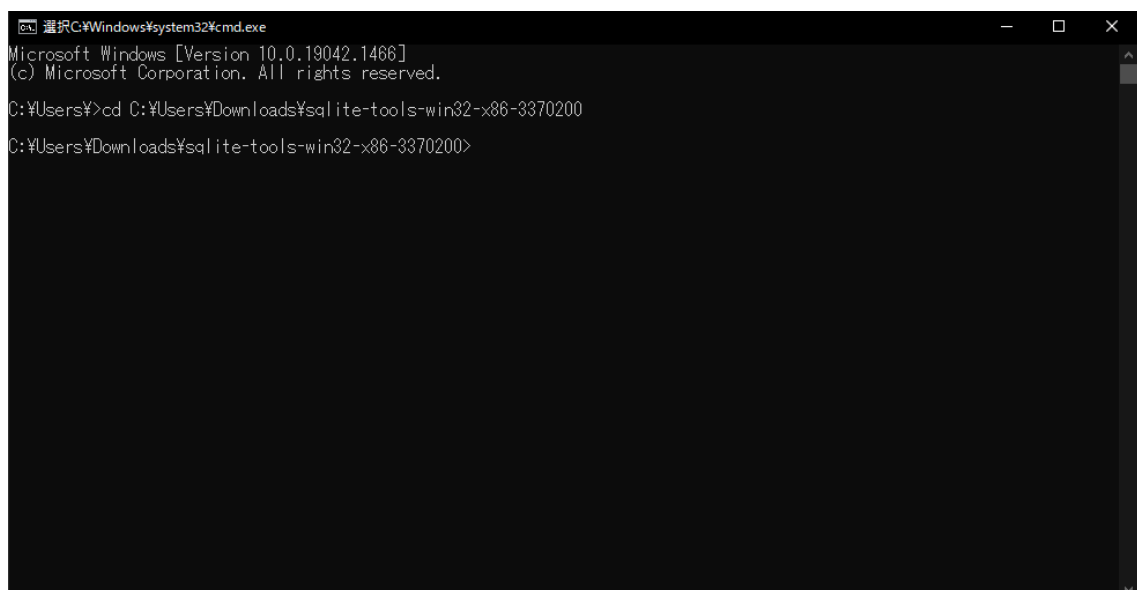


コマンドプロンプトに「cd 」(半角スペース)と入力後、右クリックを押すと、先ほどコピーしたパスが貼り付けられます。

(入力する内容)

cd (コピーしたパス)

そのままエンターキーを押してください。

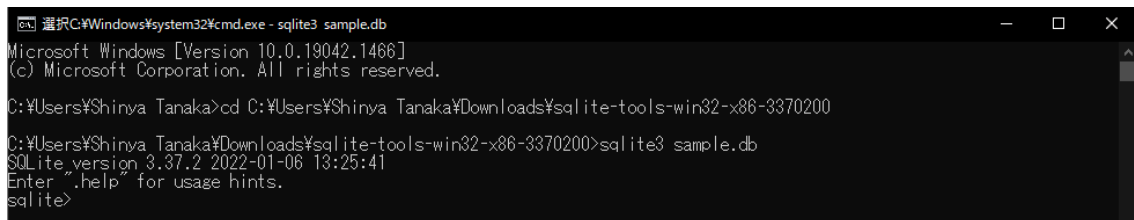


続いて、コマンドプロンプトに以下を入力し、エンターキーを押します。

(入力する内容)

```
sqlite3 sample.db
```

最後の1行が「sqlite>」となっていれば、起動成功です。



```
選択C:\Windows\system32\cmd.exe - sqlite3 sample.db
Microsoft Windows [Version 10.0.19042.1466]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Shinya Tanaka>cd C:\Users\Shinya Tanaka\Downloads\sqlite-tools-win32-x86-3370200

C:\Users\Shinya Tanaka\Downloads\sqlite-tools-win32-x86-3370200>sqlite3 sample.db
SQLite version 3.37.2 2022-01-06 13:25:41
Enter ".help" for usage hints.
sqlite>
```

コマンドプロンプトは、3.2以降の講義で使用します。

開いたままにしておいてください。

2.5. (参考)SQLite の終了

SQLiteを終了するには、exitもしくはquitコマンドを使用します。

(入力する内容)

```
.exit
```

(入力する内容)

```
.quit
```

3. SQL 基礎

RDBMS を操作するには、SQL という言語を使います。

- ・ **SQL とは**
- ・ **SQL の基本操作**

について説明します。

3.1. SQL とは

SQL は、RDBMS を操作する言語です。

どの RDBMS にも共通して使える標準 SQL と、RDBMS 特有の方言的な SQL が存在します。

標準 SQL さえ覚えれば、どの RDBMS についても基本的な操作ができるようになります。

本講義では SQLite を使用しますが、Oracle など他のシステムでも活用可能である標準 SQL を中心に、理解していきましょう。

3.2. テーブルの作成

まずは、テーブルを作成しましょう。

テーブルは、データを格納する入れ物のことです。

テーブル作成には、SQL の CREATE TABLE 文を使用します。

今回は、所属している社員の情報を管理するための「employee テーブル」を作成します。

【社員テーブルのイメージ】

社員番号(id)	社員名(name)	年齢(age)
1	田中	44
2	佐藤	23
3	鈴木	29

1 章で開いたコマンドプロンプトに、以下のコマンドを入力し実行してください。

(実行する SQL)

```
CREATE TABLE employee(id integer,name text,age integer);
```

特にエラーメッセージが表示されなければ、employee テーブルが作成できています。

CREATE TABLE 文で作成したのは「入れ物」であり、まだデータが入っていない状態です。

【空のテーブルのイメージ】

社員番号(id)	社員名(name)	年齢(age)

次の節にて、テーブルへのデータの追加や中身の確認方法について学んでいきましょう。

3.3. CRUD 文とは

テーブル内のデータの検索、追加、修正などを行う SQL 文を CRUD 文と呼びます。

SQL 文は、大きく 3 種類に分けられます。

【SQL 文の種類】

SQL の種類	概要
DDL	データ定義言語(Data Definition Language)。 データを格納するテーブルの作成・変更・削除などを行う。
DML(=CRUD 文)	データ操作言語(Data Manipulation Language) レコードの検索、更新、削除などを行う。
DCL	データ制御言語(Data Control Language) 変更内容の確定・取り消しやユーザー権限の設定を行う。

3.2 でテーブルの作成に使用した CREATE TABLE 文は、DDL(データ定義言語)にあたります。
エンジニアとして使用頻度が高いのは、データの検索や更新を行う DML(データ操作言語)です。

DML は具体的には、以下の 4 種類のデータ操作を行います。

【CRUD 文の一覧】

操作の種類	概要	該当する SQL 文
Create	追加	INSERT 文
Read	読み込み・検索	SELECT 文
Update	既存データの更新	UPDATE 文
Delete	削除	DELETE 文

CRUD 文の由来は、Create、Read、Update、Delete の頭文字です。

それでは CRUD 文を使って、employee テーブルへのデータの追加・読み込みを行ってみましょう。

3.4. INSERT 文

INSERT 文は、データの追加を行う SQL 文です。

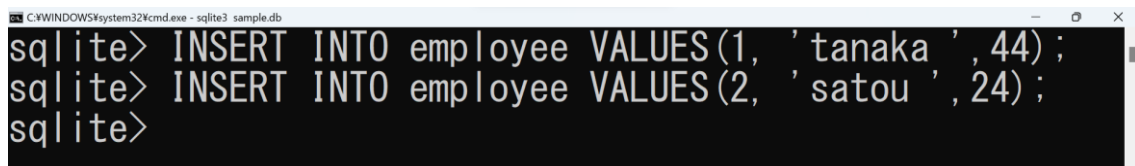
<構文 INSERT 文>

```
INSERT INTO <テーブル名> VALUES (値 1, 値 2, 値 3, ……);
```

構文は確認しましたので、次は employee テーブルにデータを追加してみましょう。

(実行する SQL)

```
INSERT INTO employee VALUES (1, 'tanaka', 44);  
INSERT INTO employee VALUES (2, 'satou', 24);
```



```
C:\WINDOWS\system32\cmd.exe - sqlite3 sample.db  
sqlite> INSERT INTO employee VALUES (1, 'tanaka', 44);  
sqlite> INSERT INTO employee VALUES (2, 'satou', 24);  
sqlite>
```

特にエラーが出なければ、登録ができています。

ワーク 2

次のデータを employee テーブルに追加する INSERT 文を作って、SQLite で実行してみましょう。

【追加したいデータ】

社員番号(id)	社員名(name)	年齢(age)
3	suzuki	29

ヒント:

テキストを追加する場合は、「tanaka」のように「'」（シングルクォーテーション）で囲う必要があります。

3.5. SELECT 文

データの検索を行うには SELECT 文を使います。

<構文 SELECT 文>

```
SELECT <列名 1>, <列名 2>, ……  
FROM <テーブル名>;
```

先ほど登録したデータの id と name 列を検索してみましょう。

(実行する SQL)

```
SELECT id, name FROM employee;
```

【実行結果】

```
1 | tanaka  
2 | satou  
3 | suzuki
```

SELECT 文は、SELECT の後ろに指定した列だけが検索結果に表示されます。

今回は id と name を指定しているため、age は表示されていません。

SELECT の後ろに、列名ではなく「*」(アスタリスク)を指定することで、テーブル内の全ての列を検索できます。

(全ての列を取得する SELECT 文)

```
SELECT * FROM employee;
```

【実行結果】

```
1 | tanaka | 44  
2 | satou | 24  
3 | suzuki | 29
```

ワーク 3

employee テーブルから name と age を表示する SELECT 文を作り、実行してみましょう。

「社員番号が 1 番のデータを検索したい」など、特定の条件を満たすデータだけを検索したい場合、WHERE 句を使います。

<構文 WHERE 句を活用した SELECT 文>

```
SELECT <列名 1>, <列名 2>, ……  
FROM <テーブル名>  
WHERE <条件>;
```

試しに 「id が 1 と一致する(=)データ」を SELECT 文で検索してみましょう。

(実行する SQL)

```
SELECT * FROM employee WHERE id = 1;
```

【実行結果】

```
1 | tanaka | 44
```

WHERE 句は SELECT だけでなく、これから学習する UPDATE 文、DELETE 文にも使えます。「特定のデータだけを対象に SQL を実行する場合、WHERE 句を追加する」と覚えておくとよいでしょう。

3.6. UPDATE 文

既存のレコードを更新するのが、UPDATE 文です。

<構文 UPDATE 文>

```
UPDATE <テーブル名>  
SET <列名> = <値>;
```

UPDATE 文を使って、全社員の年齢を 20 歳に変更してみましょう。

(実行する SQL)

```
UPDATE employee SET age = 20;
```

結果を確認するために、SELECT 文を実行します。

(実行する SQL)

```
SELECT * FROM employee;
```

【実行結果】

```
1 | tanaka | 20  
2 | satou  | 20  
3 | suzuki | 20
```

今回実行したように全てのレコードを更新することは、実際はほとんどありません。WHERE 句で条件を指定し、特定のデータだけを更新するのが UPDATE 文の一般的な使い方です。

<構文 WHERE 句を活用した UPDATE 文>

```
UPDATE <テーブル名> SET <列名> = <値> WHERE <条件>;
```

WHERE 句で絞り込みを行い、id=1 のレコードの age だけを 44 に更新しましょう。

(実行する SQL)

```
UPDATE employee SET age = 44 WHERE id = 1;
```

結果を確認するために、SELECT 文を実行します。

(実行する SQL)

```
SELECT * FROM employee;
```

【実行結果】

```
1 | tanaka | 44  
2 | satou  | 20  
3 | suzuki | 20
```

ワーク 4

以下 2 つの UPDATE 文を作り、それぞれ実行しましょう。

- (1)id が 2 のレコードの age を 24 に更新する
- (2)id が 3 のレコードの age を 29 に更新する

3.7. DELETE 文

レコードを削除するには DELETE 文を使います。

<構文 DELETE 文>

```
DELETE FROM <テーブル名>;
```

DELETE 文も UPDATE 文同様、WHERE 句と組み合わせて、絞り込んで活用することが多いです。

<構文 WHERE 句を活用した DELETE 文>

```
DELETE FROM <テーブル名> WHERE <条件>;
```

age が 44 のレコードを削除する DELETE 文を実行しましょう。

(実行する SQL)

```
DELETE FROM employee WHERE age = 44;
```

SELECT 文で実行結果を確認します。

(実行する SQL)

```
SELECT * FROM employee;
```

【実行結果】

```
2|satou|24
3|suzuki|29
```

ワーク 5

以下の DELETE 文を作り、実行しましょう。

- ・ id が 3 のレコードを削除する

4. SQL 応用

ここまで「1 つのテーブルの特定のデータに対する操作」を行う SQL を説明しました。

最後に、SQL の応用知識として、

- ・ 複数のテーブルを結合してデータを操作する方法
- ・ 複数のレコードを集計する方法

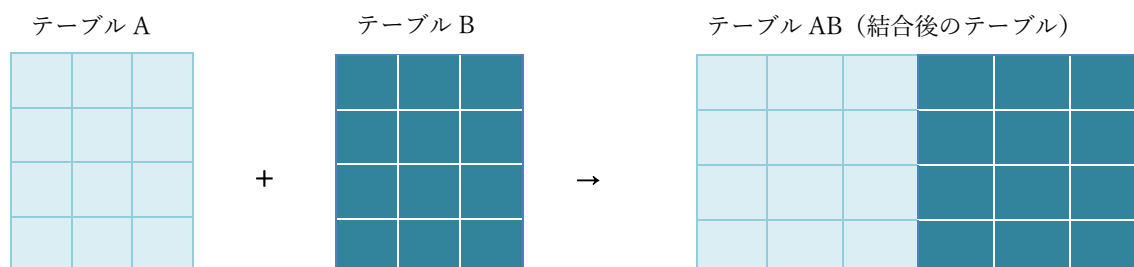
について説明します。

どちらも実務では必須となる知識です。どんなことができるのか、概要をおさえておきましょう。

4.1. 結合

結合とは、複数のテーブルのレコードを組み合わせて、1 つの仮想的なテーブルを一時的に作成する演算のことです。

【結合のイメージ】



実際にやってみましょう。

student テーブルと club テーブルを作成し、レコードを追加します。

【student テーブルのイメージ】

生徒番号(id)	生徒名(name)	部活 ID(club_id)
1	kuwata	1
2	honda	2
3	matsui	1

【club テーブルのイメージ】

部活 ID(club_id)	部活名(club_name)
1	baseball
2	football

以下の SQL を実行してください。

(実行する SQL)

```
CREATE TABLE student(id integer,name text,club_id integer);  
CREATE TABLE club(club_id integer,club_name text);  
  
INSERT INTO student VALUES(1,'kuwata',1);  
INSERT INTO student VALUES(2,'honda',2);  
INSERT INTO student VALUES(3,'matsui',1);  
  
INSERT INTO club VALUES(1,'baseball');  
INSERT INTO club VALUES(2,'football');
```

Student テーブルと club テーブルを結合し、以下の検索結果を表示します。

【実行結果イメージ】

生徒番号(id)	生徒名(name)	部活(club_name)
1	kuwata	baseball
2	honda	football
3	matsui	baseball

(実行する SQL)

```
SELECT id,name,club_name FROM student INNER JOIN club ON student.club_id = club.club_id;
```

【実行結果】

```
1|kuwata|baseball  
2|honda|football  
3|matsui|baseball
```

このように、複数のテーブルを紐付けて情報の検索ができます。

4.2. 集計

SQL では関数を使用して、合計や最大値などの複数のレコードの集計結果の出力も行えます。

代表的な集計関数を紹介します。

【代表的な集計関数】

関数名	概要
COUNT	テーブルのレコード数（行数）を数える
SUM	テーブルの数値列のデータの合計を求める
AVG	テーブルの数値列のデータの平均を求める
MAX	テーブルの任意の列のデータの最大値を求める
MIN	テーブルの任意の列のデータの最小値を求める

関数は以下のように記述します。

<構文 関数>

関数名 (列名)

実際に関数を使った SELECT 文を実行してみましょう。

(employee テーブルの id の最大値を求める SQL)

```
SELECT MAX(id) FROM employee;
```

【実行結果】

2

ワーク 6

以下の SQL 文を作り、実行しましょう。

- ・ employee テーブルの age の平均値を求める SQL

【実行結果】

24.0

5. ワークの解答

ワーク 2

次のデータを employee テーブルに追加する INSERT 文を作って、SQLite で実行してみましょう。

【追加したいデータ】

社員番号(id)	社員名(name)	年齢(age)
3	suzuki	29

ヒント:

テキストを追加する場合は、「tanaka」のように「'」（シングルクォーテーション）で囲う必要があります。

回答:

```
INSERT INTO employee VALUES (3, 'suzuki', 29);
```

ワーク 3

employee テーブルから name と age を表示する SELECT 文を作り、実行してみましょう。

回答:

```
SELECT name, age FROM employee;
```

ワーク 4

以下 2 つの UPDATE 文を作り、それぞれ実行しましょう。

- (1) id が 2 のレコードの age を 24 に更新する
- (2) id が 3 のレコードの age を 29 に更新する

回答:

```
UPDATE employee SET age = 24 WHERE id = 2;  
UPDATE employee SET age = 29 WHERE id = 3;
```

ワーク 5

以下の DELETE 文を作り、実行しましょう。

- ・ id が 3 のレコードを削除する

回答:

```
DELETE FROM employee WHERE id = 3;
```

ワーク 6

以下の SQL 文を作り、実行しましょう。

- ・ employee テーブルの age の平均値を求める SQL

回答:

```
SELECT AVG(age) FROM employee;
```

【実行結果】

```
24.0
```