

15-1 SQLite 3 のインストール, データベース新規作成, テーブル 定義, テーブルの削除

1. SQLite 3 のインストールと基本操作
2. テーブル定義, データ型, 主キー, SQL 問い合わせ
3. 結合

SQL の基本を, SQLite 3 を用いて演習する.

URL: <https://www.kkaneko.jp/data/sqlite3/index.html>

第1回のアウトライン



- SQLite 3 のインストール
- SQLite 3 データベースの新規作成
- SQLite 3 コマンドラインシェルの終了

SQLite 3



- SQLite 3は、**リレーショナルデータベース管理システム**.
- SQLite 3 の特徴
 - アカウント（ユーザ名やパスワード）の機能がない
 - 設定なしで動く
 - 並行処理制御，リモートとの通信の機能は SQLite 3 にない
- SQLite 3 の URL: **<https://www.sqlite.org>**
- SQLite 3 のコピーライト:
<https://www.sqlite.org/copyright.html>
- Windows での SQLite 3 のインストール:
<https://www.kkaneko.jp/data/sqlite3/sqlite3.html>

SQLite 3 のインストール



① SQLite の Web ページを開く。

<https://www.sqlite.org/>

SQLite 3 のインストール



② 「Download」 をクリック



[Home](#) [About](#) [Documentation](#) [Download](#) [License](#) [Support](#) [Purchase](#)

What Is SQLite?

SQLite is a C-language library that implements a [small](#), [fast](#), [self-contained](#), [high-reliability](#), [full-featured](#), SQL database engine. SQLite is the [most used](#) database engine in the world. SQLite is built into all mobile phones and most computers and comes bundled inside countless other applications that people use every day. [More Information...](#)

The SQLite [file format](#) is stable, cross-platform, and backwards compatible and the developers pledge to keep it that way [through the year 2050](#). SQLite database files are commonly used as containers to transfer rich content between systems [\[1\]](#) [\[2\]](#) [\[3\]](#) and as a long-term archival format for data [\[4\]](#). There are over 1 trillion (1e12) SQLite databases in active use [\[5\]](#).

SQLite [source code](#) is in the [public-domain](#) and is free to everyone to use for any purpose.

Latest Release

[Version 3.35.5](#) (2021-04-14) [Download](#) [Prior Releases](#)

③ Windows 版のダウンロード

Precompiled Binaries for Windows

[sqlite-dll-win32-x86-3350500.zip](#) 32-bit DLL (x86) for SQLite version 3.35.5.
(sha3: 55b49ce165984865d62918cbd0ad34fa7af82820e4f5d10c899e2191dc)
(538.88 KiB)

[sqlite-dll-win64-x64-3350500.zip](#) 64-bit DLL (x64) for SQLite version 3.35.5.
(sha3: 15ba94e68c6596bb292ba6a4ce303f5beaf731754846d01fd7a55a6edf)
(879.80 KiB)

[sqlite-tools-win32-x86-3350500.zip](#) A bundle of command-line tools for managing SQLite database f
the [sqldiff.exe](#) program, and the [sqlite3_analyzer.exe](#) program.
(sha3: 295214b0c3d6bfef32400632fc5237e1cb57c07ba0b76e94c98fb0c3fe9)
(1.81 MiB)

SQLite 3 のインストール



④ ダウンロードした .zip ファイルを展開 (解凍)

Windows (C:) > sqlite3



名前	更新日時	種類
 sqldiff.exe	2021/05/08 13:01	アプリケーション
 sqlite3.exe	2021/05/08 13:01	アプリケーション
 sqlite3_analyzer.exe	2021/05/08 13:01	アプリケーション

SQLite 3 の起動



- 起動

sqlite3.exe を使う

Windows (C:) > sqlite3

名前	更新日時	種類
sqldiff.exe	2021/05/08 13:01	アプリケーション
sqlite3.exe	2021/05/08 13:01	アプリケーション
sqlite3_analyzer.exe	2021/05/08 13:01	アプリケーション

SQLite 3 データベースの新規作成



データベースファイル名: **C:¥sqlite3¥mydb**
で, **SQLite 3 データベースの新規作成**

- ① 前もって Windows で **C:¥sqlite3** のような名前の **データベースディレクトリ** を作成しておく
- ② **sqlite3.exe** を実行 (SQLite 3 の起動)

名前	更新日時	種類
sqldiff.exe	2018/04/11 2:59	アプリケーション
sqlite3.exe	2018/04/11 2:59	アプリケーション
sqlite3_analyzer.exe	2018/04/11 2:59	アプリケーション

- ③ 新しい画面が開くので確認

選択C:¥sqlite3¥sqlite3.exe

```
SQLite version 3.35.5 2021-04-19 18:32:05
Enter ".help" for usage hints.
Connected to a transient in-memory database.
Use ".open FILENAME" to reopen on a persistent database.
sqlite> _
```

SQLite 3 データベースの新規作成



データベースファイル名: `C:¥sqlite3¥mydb`
で, SQLite 3 データベースの新規作成

- ④ 「`.open --new`」で, データベースの新規作成

```
.open --new C:¥¥sqlite3¥¥mydb
```

```
sqlite> .open --new C:¥¥sqlite3¥¥mydb
sqlite> _
```

- ⑤ 「`.exit`」で, SQLite 3 を終了

```
sqlite> .exit
```

SQLite 3 の起動と終了



• 起動

sqlite3.exe を使う

Windows (C:) > sqlite3

名前	更新日時	種類
sqldiff.exe	2021/05/08 13:01	アプリケーション
sqlite3.exe	2021/05/08 13:01	アプリケーション
sqlite3_analyzer.exe	2021/05/08 13:01	アプリケーション

• 終了

次のコマンドを実行

```
.exit
```

データベースオープン



SQLite 3 で、次のコマンドを実行

```
.open --new <データベースファイル名>
```

新規作成して
オープン

```
.open <データベースファイル名>
```

既存のものを
オープン

※ データベースの新規作成について

- 指定したファイルが**すでに存在するとき**は、**削除**され、**空のデータベースファイル**が新規作成される
- 指定したファイルが**存在しないとき**は、**空のデータベースファイル**が新規作成される

SQLite 3 を実行できるオンラインサイトの例



- GDB online Debugger のサイト

URL: <https://www.onlinegdb.com/>

The screenshot displays the OnlineGDB website interface. The browser address bar shows the URL <https://www.onlinegdb.com/>. The page title is "GDB online Debugger | Compile X". The main content area shows a code editor with the following SQL code:

```
1 create table products (  
2   id integer primary key not NULL,  
3   product_name text unique not NULL,  
4   type text not NULL,  
5   cost real,  
6   created_at datetime not NULL  
7 );  
8 BEGIN TRANSACTION;  
9 INSERT INTO products VALUES ( 1, 'Fukuoka apple', 'apple', 50, datetime('now') );  
10 INSERT INTO products VALUES ( 2, 'Kumamoto orange L', 'orange', 30, datetime('now') );  
11 INSERT INTO products VALUES ( 3, 'Kumamoto orange M', 'orange', 20, datetime('now') );  
12 INSERT INTO products VALUES ( 4, 'Fukuoka melon', 'melon', NULL, datetime('now') );  
13 COMMIT;  
14 SELECT * FROM products;
```

The language dropdown menu is set to "SQLite". The console output shows the execution results of the SQL queries:

```
input  
1|Fukuoka apple|apple|50.0|2021-05-12 06:35:41  
2|Kumamoto orange L|orange|30.0|2021-05-12 06:35:41  
3|Kumamoto orange M|orange|20.0|2021-05-12 06:35:41  
4|Fukuoka melon|melon|||2021-05-12 06:35:41  
  
...Program finished with exit code 0  
Press ENTER to exit console.
```

Language で
SQLite を選ぶ

15-2 SQLite3 でテーブルの作成

アウトライン



- テーブル
- テーブル定義
- データ型
- 主キー
- NULL
- テーブルへのレコードの挿入
- SQL 問い合わせ

テーブルの例



テーブル名: products

id	name	price
1	orange	50
2	apple	100
3	melon	500

テーブル定義



テーブル名: products

テーブル定義では,

- テーブル名
- 属性の属性名
- 属性のデータ型

などを設定して, テーブルを定義する

id	name	price
1	orange	50
2	apple	100
3	melon	500

```
CREATE TABLE products (  
  id INTEGER PRIMARY KEY NOT NULL,  
  name TEXT NOT NULL,  
  price REAL);
```

属性のデータ型



id	name	price
1	orange	50
2	apple	100
3	melon	500

属性名

テーブル
の本体



整数

INTEGER



長いテキスト

TEXT



浮動小数点数

REAL

← SQL のキーワード

それぞれの属性のデータ型

属性のデータ型



Access の主なデータ型	SQL のキーワード	
	NULL	空値
短いテキスト	CHAR	文字列
長いテキスト	TEXT	文字列
数値	INTEGER, REAL	整数や浮動小数 点数
日付／時刻	DATETIME	日付や時刻など
Yes／No	BIT, BOOL	ブール値

※ **整数**は **INTEGER**, **浮動小数点数** (小数付きの数) は **REAL**

※ **短いテキスト**は半角 **255文字分**までが目安
それ以上になる可能性があるときは**長いテキスト**

主キー



通し番号, 学生番号のように, 1つのテーブルの中で
同じ値が2回以上出ないと前もって分かっている**属性**

id	name	price
1	orange	50
2	apple	100
3	melon	500

主キー

リレーショナルデータベースの NULL



- **NULL** は「ヌル」あるいは「ナル」と読む
- **リレーショナルデータベース**で **NULL** は、次の場合に使う
 1. **未定, 未知, 不明** (分からない場合)
 2. **非存在** (もともと存在しない場合)

テーブル定義と一貫性制約



【SQL プログラム】

```
CREATE TABLE products (  
  id INTEGER PRIMARY KEY NOT NULL,  
  name TEXT NOT NULL,  
  price REAL);
```

id:

主キー (PRIMARY KEY),
NULL になることはない (NOT NULL)

name:

NULL になることはない (NOT NULL)


テーブルの制約について記述.

データベースの一貫性を維持するのに役立つ.

テーブル定義 products

【SQL プログラム】

```
CREATE TABLE products (  
  id INTEGER PRIMARY KEY NOT NULL,  
  name TEXT NOT NULL,  
  price REAL);
```

 選択C:\sqlite3\sqlite3.exe

```
SQLite version 3.35.5 2021-04-19 18:32:05  
Enter ".help" for usage hints.  
Connected to a transient in-memory database.  
Use ".open FILENAME" to reopen on a persistent database.  
sqlite> CREATE TABLE products (  
  ...>   id INTEGER PRIMARY KEY NOT NULL,  
  ...>   name TEXT NOT NULL,  
  ...>   price REAL);  
sqlite> _
```

新しいレコードの挿入



テーブル名: products

id	name	price
1	orange	50
2	apple	100
3	melon	500



id	name	price
1	orange	50
2	apple	100
3	melon	500
4	apple	150

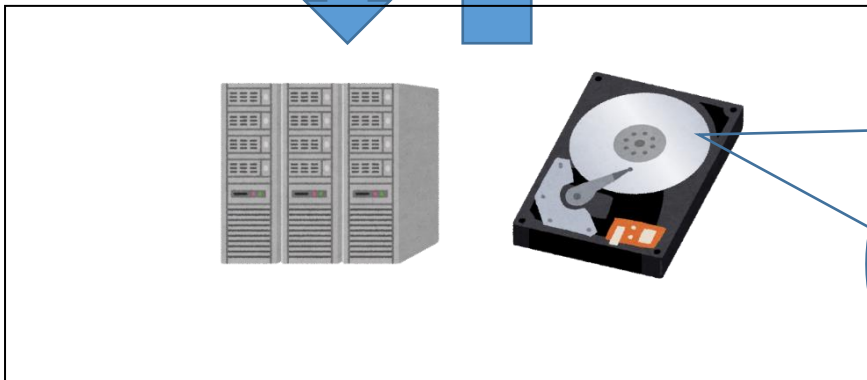
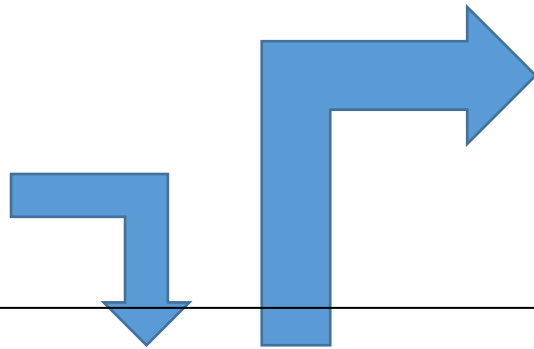
INSERT INTO products **VALUES**(4, 'apple', 150);

テーブル名 値の並び. 半角のカンマ「,」で区切る
※ 文字列は半角の「'」で囲む

問い合わせ（クエリ）の仕組み



問い合わせ
（クエリ）
のコマンド



リレーショナル
データベースシステム

問い合わせ（クエリ）
の結果は、**テーブル形式の**
データ

id	name	price
1	orange	50
2	apple	100
3	melon	500

	what	at	
赤	XX	貸出	2021-05-11 13:30:18
赤	XX	返却	2021-05-11 13:30:18
青	YY	貸出	2021-05-11 13:30:18
緑	ZZ	貸出	2021-05-11 13:30:18

データの種類ごとに分かれた、たくさんの**テーブル**

レコードの挿入, SQL 問い合わせ



【SQL プログラム】

```
INSERT INTO products VALUES( 1, 'orange', 50 );  
INSERT INTO products VALUES( 2, 'apple', 100 );  
INSERT INTO products VALUES( 3, 'melon', 500 );  
SELECT * FROM products;
```

```
sqlite> INSERT INTO products VALUES ( 1, 'orange', 50 );  
sqlite> INSERT INTO products VALUES ( 2, 'apple', 100 );  
sqlite> INSERT INTO products VALUES ( 3, 'melon', 500 );  
sqlite> SELECT * FROM products;  
1 | orange | 50.0  
2 | apple  | 100.0  
3 | melon  | 500.0  
sqlite> _
```

【SQL プログラム】

```
SELECT * FROM products WHERE price > 90;
```

```
sqlite> SELECT * FROM products WHERE price > 90;  
2|apple|100.0  
3|melon|500.0  
sqlite>
```

SQL を用いたテーブルの削除



テーブル products の削除

【SQL プログラム】

```
drop table products;
```

```
sqlite> drop table products;  
sqlite> _
```

ここで使用した SQL



- テーブル定義

CREATE TABLE ...

- 問い合わせ

SELECT ... FROM ...

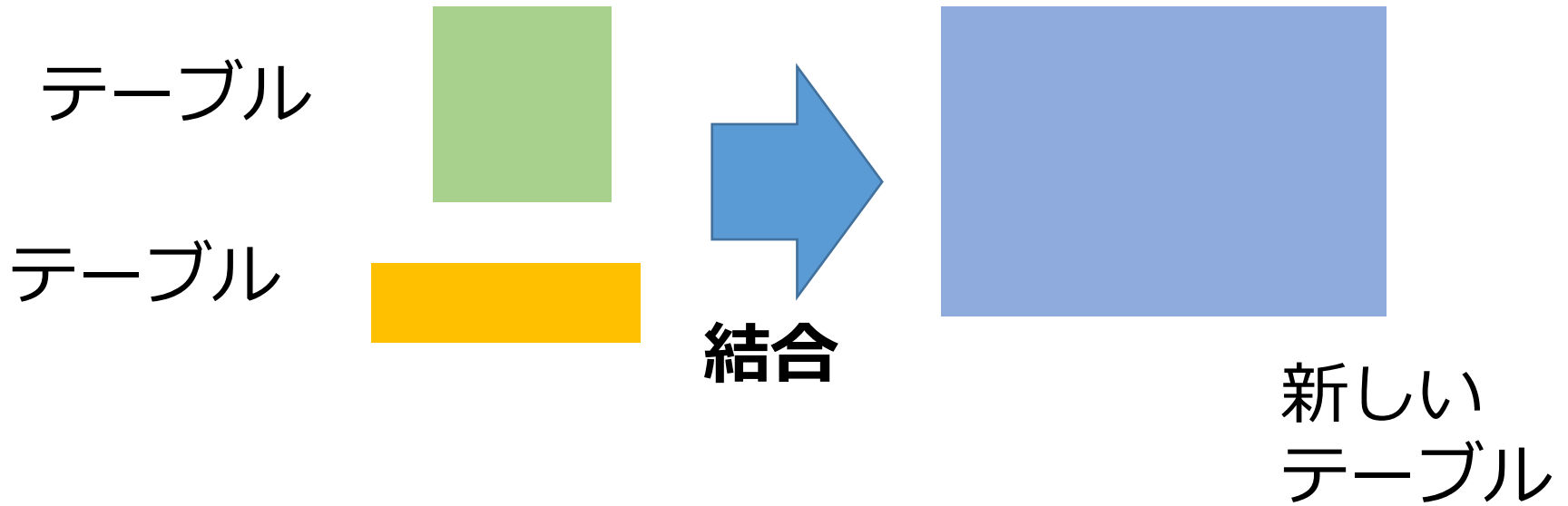
SELECT ... FROM ... WHERE ...

- レコードの挿入

INSERT INTO ...

15-3 SQLによる結合


結合



テーブル定義 teacher

【SQL プログラム】

```
CREATE TABLE teacher (  
  id INTEGER PRIMARY KEY NOT NULL,  
  name TEXT NOT NULL,  
  teacher_name TEXT NOT NULL);
```

 選択C:\sqlite3\sqlite3.exe

```
SQLite version 3.35.5 2021-04-19 18:32:05  
Enter ".help" for usage hints.  
Connected to a transient in-memory database.  
Use ".open FILENAME" to reopen on a persistent database.  
sqlite> CREATE TABLE teacher (  
  ...>   id INTEGER PRIMARY KEY NOT NULL,  
  ...>   name TEXT NOT NULL,  
  ...>   teacher_name TEXT NOT NULL);  
sqlite> _
```


テーブル定義 student

【SQL プログラム】

```
CREATE TABLE student (  
  id INTEGER PRIMARY KEY NOT NULL,  
  student_name TEXT NOT NULL,  
  tid INTEGER NOT NULL,  
  score INTEGER);
```

```
sqlite> CREATE TABLE student (  
  ...>   id INTEGER PRIMARY KEY NOT NULL,  
  ...>   student_name TEXT NOT NULL,  
  ...>   tid INTEGER NOT NULL,  
  ...>   score INTEGER);  
sqlite> _
```

レコードの挿入と確認



【SQLプログラム】

```
INSERT INTO teacher VALUES(1, 'db', 'k');  
INSERT INTO teacher VALUES(2, 'python', 'a');  
SELECT * FROM teacher;
```

```
sqlite> INSERT INTO teacher VALUES(1, 'db', 'k');  
sqlite> INSERT INTO teacher VALUES(2, 'python', 'a');  
sqlite> SELECT * FROM teacher;  
1|db|k  
2|python|a  
sqlite> _
```

レコードの挿入と確認



【SQL プログラム】

```
INSERT INTO student VALUES(1, 'kk', 1, 85);  
INSERT INTO student VALUES(2, 'aa', 1, 75);  
INSERT INTO student VALUES(3, 'nn', 1, 90);  
INSERT INTO student VALUES(4, 'kk', 2, 85);  
INSERT INTO student VALUES(5, 'nn', 2, 75);  
SELECT * FROM student;
```

```
sqlite> INSERT INTO student VALUES (1, 'kk', 1, 85);  
sqlite> INSERT INTO student VALUES (2, 'aa', 1, 75);  
sqlite> INSERT INTO student VALUES (3, 'nn', 1, 90);  
sqlite> INSERT INTO student VALUES (4, 'kk', 2, 85);  
sqlite> INSERT INTO student VALUES (5, 'nn', 2, 75);  
sqlite> SELECT * FROM student;  
1 | kk | 1 | 85  
2 | aa | 1 | 75  
3 | nn | 1 | 90  
4 | kk | 2 | 85  
5 | nn | 2 | 75  
sqlite>
```

【SQL プログラム】

```
SELECT * FROM teacher, student  
WHERE teacher.id = student.tid;
```

```
sqlite> SELECT * FROM teacher, student  
...> WHERE teacher.id = student.tid;  
1 | db | k | 1 | kk | 1 | 85  
1 | db | k | 2 | aa | 1 | 75  
1 | db | k | 3 | nn | 1 | 90  
2 | python | a | 4 | kk | 2 | 85  
2 | python | a | 5 | nn | 2 | 75  
sqlite>
```