

第2回. リレーショナルデータベース入門

- SQL を用いた**テーブル**への**行の挿入**
- **SQL 問い合わせ**の発行と評価結果の確認

作成するテーブル (table)

行 (row) と呼ぶ

id	product_name	type	cost	created_at
1	Fukuoka apple	apple	50	2018-05-07 11:04:53
2	Kumamoto orange L	orange	30	2018-05-07 11:04:54
3	Kumamoto orange M	orange	20	2018-05-07 11:04:55
4	Fukuoka melon	melon	NULL	2018-05-07 11:04:56

列 (column) と呼ぶ

事前準備. SQLite コマンドライン・インタフェースのダウンロード

① SQLite の Web ページを開く.

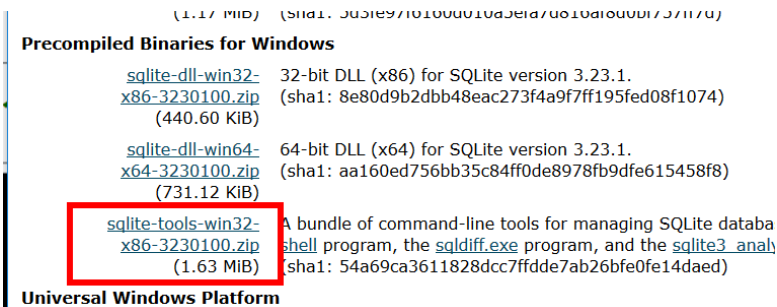
<http://www.sqlite.org/>

事前準備. SQLite コマンドライン・インタフェースのダウンロード

② 「Download」をクリック

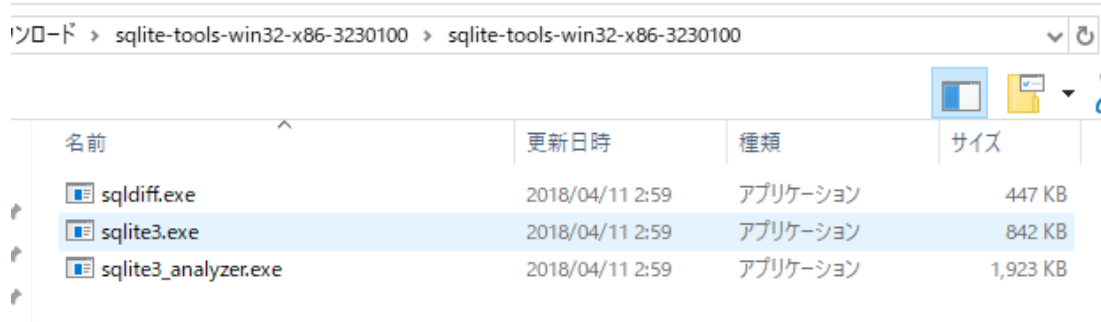


③ Windows 版のダウンロード



事前準備. SQLite コマンドライン・インタフェースのダウンロード

④ ダウンロードした .zip ファイルを展開（解凍）



名前	更新日時	種類	サイズ
sqldiff.exe	2018/04/11 2:59	アプリケーション	447 KB
sqlite3.exe	2018/04/11 2:59	アプリケーション	842 KB
sqlite3_analyzer.exe	2018/04/11 2:59	アプリケーション	1,923 KB

sqlite3.exe を使う

SQLite データベースの新規作成

データベース論理名: C:¥SQLite¥mydb
で, SQLite データベースの新規作成

- ① 前もって Windows で **C:¥SQLite** というディレクトリ（フォルダ）を作成しておく
- ② sqlite3.exe を実行

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

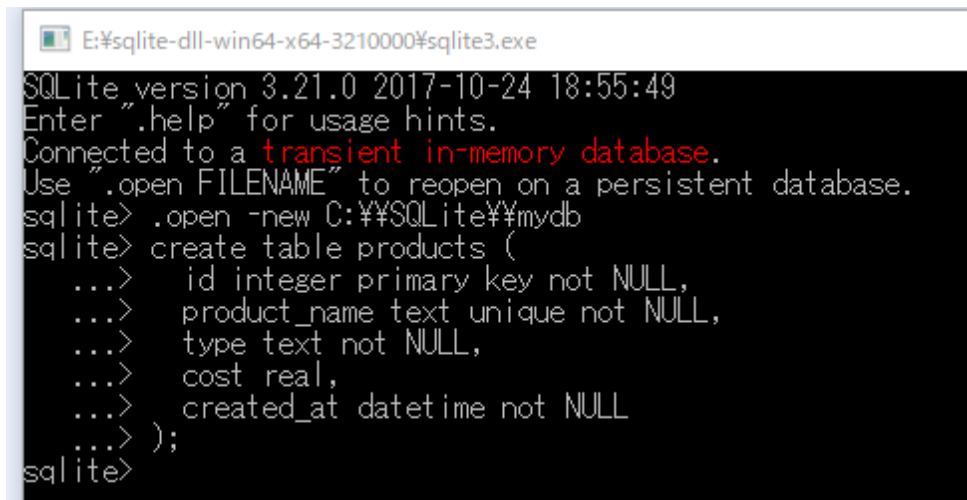
- ③ 新しい画面が開くので確認

```
e:¥Downloads¥sqlite-tools-win32-x86-3230100¥sqlite-tools-win32-x86-3230100¥sqlite3.exe
SQLite version 3.23.1 2018-04-10 17:39:29
Enter ".help" for usage hints.
Connected to a transient in-memory database.
Use ".open FILENAME" to reopen on a persistent database.
sqlite>
```

SQL を用いたテーブル定義と制約の記述

【SQL プログラム】

```
create table products (  
  id integer primary key not NULL,  
  product_name text unique not NULL,  
  type text not NULL,  
  cost real,  
  created_at datetime not NULL  
);
```



```
E:\sqlite-dll-win64-x64-3210000\sqlite3.exe  
SQLite version 3.21.0 2017-10-24 18:55:49  
Enter ".help" for usage hints.  
Connected to a transient in-memory database.  
Use ".open FILENAME" to reopen on a persistent database.  
sqlite> .open -new C:\¥¥SQLite¥¥mydb  
sqlite> create table products (  
...>   id integer primary key not NULL,  
...>   product_name text unique not NULL,  
...>   type text not NULL,  
...>   cost real,  
...>   created_at datetime not NULL  
...> );  
sqlite>
```

SQL を用いたテーブルへの行の挿入

【SQL プログラム】

```
BEGIN TRANSACTION;  
INSERT INTO products VALUES( 1, 'Fukuoka apple', 'apple',  
50, datetime('now') );  
INSERT INTO products VALUES( 2, 'Kumamoto orange L',  
'orange', 30, datetime('now') );  
INSERT INTO products VALUES( 3, 'Kumamoto orange M',  
'orange', 20, datetime('now') );  
INSERT INTO products VALUES( 4, 'Fukuoka melon',  
'melon', NULL, datetime('now') );  
COMMIT;
```

```
E:\sqlite-dll-win64-x64-321000\sqlite3.exe
SQLite version 3.21.0 2017-10-24 18:55:49
Enter ".help" for usage hints.
Connected to a transient in-memory database.
Use ".open FILENAME" to reopen on a persistent database.
sqlite> .open -new C:\¥¥SQLITE¥¥mydb
sqlite> create table products (
...>   id integer primary key not NULL,
...>   product_name text unique not NULL,
...>   type text not NULL,
...>   cost real,
...>   created_at datetime not NULL
...> );
sqlite> BEGIN TRANSACTION;
sqlite> INSERT INTO products VALUES( 1, 'Fukuoka apple', 'apple', 50, datetime('now') );
sqlite> INSERT INTO products VALUES( 2, 'Kumamoto orange L', 'orange', 30, datetime('now') );
sqlite> INSERT INTO products VALUES( 3, 'Kumamoto orange M', 'orange', 20, datetime('now') );
sqlite> INSERT INTO products VALUES( 4, 'Fukuoka melon', 'melon', NULL, datetime('now') );
sqlite> COMMIT;
sqlite>
```


SQL 問い合わせの発行と評価結果の確認

【SQL プログラム】

```
SELECT * FROM products;
```

```
sqlite> SELECT * FROM products;  
1|Fukuoka apple|apple|50.0|2018-05-07 06:07:22  
2|Kumamoto orange L|orange|30.0|2018-05-07 06:07:22  
3|Kumamoto orange M|orange|20.0|2018-05-07 06:07:22  
4|Fukuoka melon|melon||2018-05-07 06:07:22  
sqlite>
```

SQL 問い合わせの発行と評価結果の確認

【SQL プログラム】

```
SELECT * FROM products WHERE type = 'orange';
```

```
sqlite> SELECT * FROM products WHERE type = 'orange';  
2|Kumamoto orange L|orange|30.0|2018-05-07 06:07:22  
3|Kumamoto orange M|orange|20.0|2018-05-07 06:07:22  
sqlite>
```

SQL 問い合わせの発行と評価結果の確認

【SQL プログラム】

```
SELECT * FROM products WHERE cost > 25;
```

```
sqlite> SELECT * FROM products WHERE cost > 25;  
1|Fukuoka apple|apple|50.0|2018-05-07 06:07:22  
2|Kumamoto orange L|orange|30.0|2018-05-07 06:07:22  
sqlite>
```

SQLite コマンドライン・クライアントの終了

次のコマンドを実行

```
.exit
```

```
sqlite> .exit
```

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 ...

- 行の挿入

INSERT INTO ...

テーブル定義とは

リレーショナルデータベースにおいて、

- ・ **テーブル名**
- ・ 各列の**属性名**
- ・ 各列の**データ型**

などを定義すること。 制約の指定も行う

```
create table products (  
  id integer primary key not NULL,  
  product_name text unique not NULL,  
  type text not NULL,  
  cost real,  
  created_at datetime not NULL  
);
```

SQL でのテーブル定義

```
create table <テーブル名> (  
    <属性名> <データ型名> [<列制約>],  
    ...  
    [<テーブル制約>], );
```

※ 列制約とテーブル制約は省略可能，複数可能

※ テーブルの削除は **drop table**

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)

SQL のデータ型

- **NULL** 空値
(a NULL value)
- **integer** 符号付きの整数
(signed integer)
- **real** 浮動小数点値
(floating point value)
- **char, text** 文字列
(text string)
- **datetime** 日付や時刻など
- **bool** ブール値
- **BLOB** バイナリラージオブジェクト
など

列制約

◆ 一貫性制約

- **primary key** 主キー
- **not null** 非空
- **unique** 一意
- **references** <テーブル名> (<属性名>, ...)
参照整合性制約
- **check** (<式>) 更新時にチェック

※ check は SQLite 固有の機能

◆ デフォルト値, 自動インクリメント

- **default**(<式>) デフォルト値
- **autoincrement** 自動インクリメント

テーブル制約

一貫性制約のうち複数の属性に関わるものは、**テーブル制約**の形で記述

- ◆ 一貫性制約
 - **primary key** (<属性名の並び>) 主キー
 - **unique** (<属性名の並び>) 一意
 - **check** (<式>) 更新時にチェック
 - ※ check は SQLite 固有の機能

テーブルへの行の挿入

- **insert into** <テーブル名> **values** (<式>, ...);

指定された値で, 1行挿入

- **insert into** <テーブル名> (<属性名>, ...) **values** (<式>, ...);

指定された値を, 指定された属性名に格納して
1行挿入

BEGIN TRANSACTION, COMMIT, ROLLBACK

トランザクションとはデータベースの処理単位

- **begin transaction**

データベース更新の前に発行

- **commit**

「begin transaction」 以降の全ての
データベース更新操作を確定

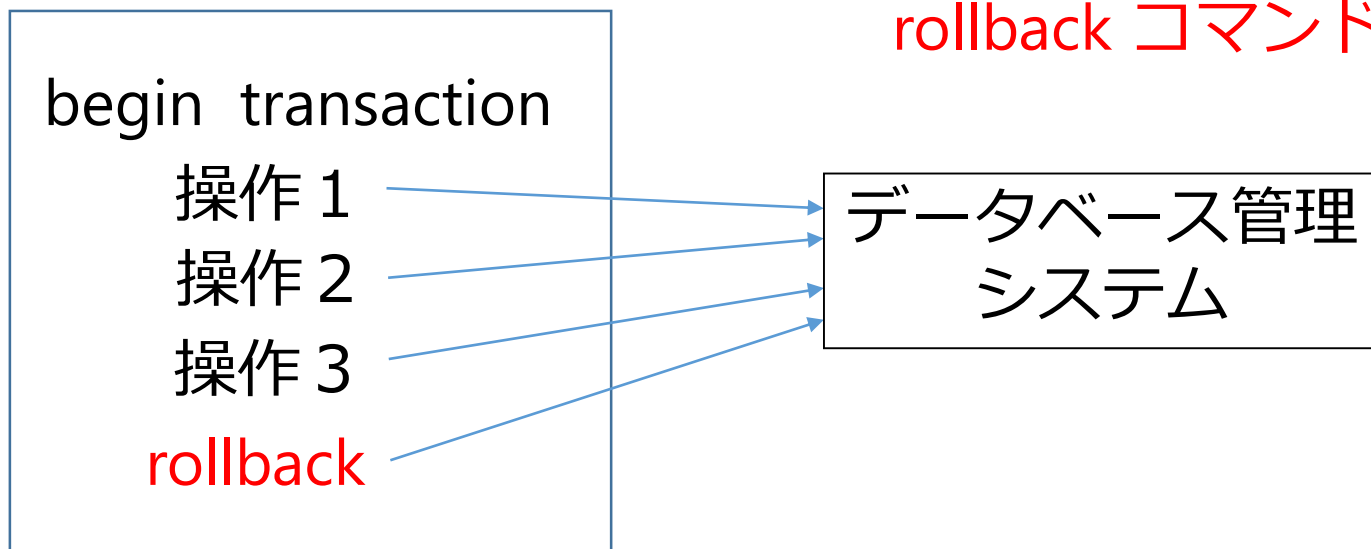
- **rollback**

「begin transaction」 以降の全ての
データベース更新操作を破棄

ロールバック (rollback) のイメージ

操作 1、操作 2、操作 3
と操作していて、
最初に戻したくなったら . . .

rollback コマンド



終わりに

Web ページ: <http://www.kkaneko.jp/dblab/intro/>

その他, さまざまな資料を金子研究室 Web ページで公開しています

<http://www.kkaneko.jp/index-j.html>

謝辞: SQLite の作者に感謝します