

**si-2. テーブル定義,
データ型, 主キー, SQL
問い合わせ**

**SQL 入門演習 (SQLite3 を
利用) (全 3 回)**

SQL の入門者へ

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

金子邦彦



第2回のアウトライン



- 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

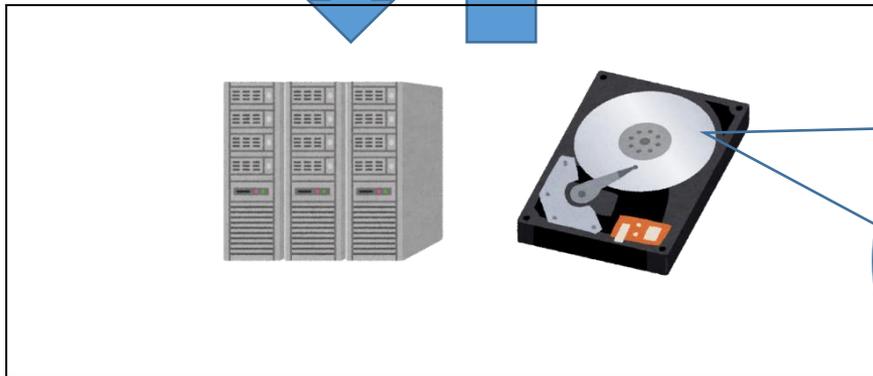
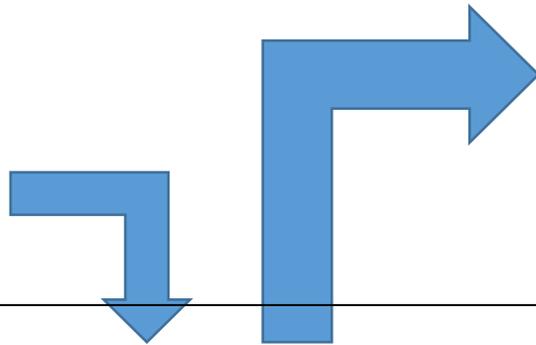
INSERT INTO products **VALUES**(4, 'apple', 150);

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 ...