SQLite データベース

IS04 組み込み

SQLite データベースは、ファイルベースで SQL を実行することができる軽量データベースです。データベース１つにつき、1 ファイルで管理し、この中に複数のテーブルを持つことができます。このファイルをアクセスするための実行ファイルをダウンロードするだけという手軽さです

リレーショナルとは

複数のテーブルを関連するフィールドで結合して、大きな表があるように振舞わせるものです。

伝票番号	受注日	販売機設置場所
1001	2006/10/01	駅前販売
1002	2006/10/02	公園販売

商品コード	商品名	単価
A001	オレンジジュース	¥100
A002	りんごジュース	¥100
A003	パインジュース	¥100
B001	缶コーヒー	¥120
C001	紅茶ストレート	¥120
D001	お茶	¥100

伝票番号	商品コード	個数
1001	A001	120
1001	A002	120
1001	A003	120
1002	B001	210
1002	C001	150
1002	D001	100

伝票番号	受注日	販売機設置場所	商品コード	商品名	単価	個数
1001	2006/10/01	駅前販売	A001	オレンジジュース	¥100	120
1001	2006/10/01	駅前販売	A002	りんごジュース	¥100	120
1001	2006/10/01	駅前販売	A003	パインジュース	¥100	120
1002	2006/10/02	公園販売	B001	缶コーヒー	¥120	210
1002	2006/10/02	公園販売	C001	紅茶ストレート	¥120	150
1002	2006/10/02	公園販売	D001	お茶	¥100	100

テーブルを分割して持つことで、データの重複を避け、効率よくデータを保存することができます。データベースの正規化により、テーブルを分割して設計します。

１．SQLite のインストール

sqlite3.exe を適当なフォルダに配置する。

２．SQLite の起動とデータベースの作成

コマンドプロンプトを起動し、sqlite3.exe を配置したフォルダをカレントにする

```
>d:
d:>cd フォルダのパス
d:¥SQLite>
```

データベース名（ファイル名）を指定して実行

```
D:¥SQLite>sqlite3 test1
SQLite version 3.7.9 2011-11-01 00:52:41
Enter ".help" for instructions
Enter SQL statements terminated with a ";"
sqlite>
```

３．SQLite のコマンド

.databases	データベースを一覧表示する
.dump	データベースをダンプする
.exit	コマンドプロンプトを終了する
.import	ファイルからデータをインポートする
.indices	テーブル名を表示する
.nullvalue	NULL を置き換える文字列を設定する
.output	実行結果の出力先を設定する
.quit	コマンドラインプログラムを終了する
.read	コマンドファイルを実行する
.schema	テーブルスキーマを表示する
.show	設定値を一覧表示する
.tables	パターンにマッチするテーブル名を表示する
.timeout	テーブルロック待ちの時間（ミリ秒）を設定する
.width	フィールドの表示幅を設定する

４．SQL の発行

１）テーブルの作成

新しいテーブルを作成します

```
コマンド：create table テーブル名（
            フィールド名１     型　属性、
            フィールド名２     型　属性、
                  ：
            ）；
```

SQLite は型を省略することができます。属性は、指定する必要のあるときだけ指定します。

主な型

分類	データ型	説明
整数	integer	１０桁＋符号の符号付き整数
	bool	0 または 1
実数	float	
	double	
日付/時刻	datetime	日時（年月日と時分秒）
	date	日付のみ
	time	時刻のみ
文字列	varchar(桁数)	文字列
	char(桁数)	文字列
	text	可変長文字列

主な属性

属性	説明
null	NULL 値を許可（デフォルト）
not null	NULL 値を許可しない
primary key	主キーに設定する。主キーを設定すると、重複する入力を許可しない。主キーは、１つのテーブルに１つだけ設定可能
auto_increment	レコードを追加するたびに自動的に１ずつ増えた値を設定

```
sqlite> create table products(
   ...> ID char(10) primary key,
   ...> name text,
   ...> image text,
   ...> price integer
   ...> );
sqlite> .schema
CREATE TABLE products(
ID char(10) primary key,
name text,
image text,
price integer
);
sqlite> .tables
products
sqlite>
```

２）レコードの挿入

指定のテーブルにデータをレコードに追加します。全フィールドのデータをフィールドの順に指定するときは、フィールド名は省略できます。

コマンド：**insert into テーブル名　（フィールド名１、フィールド名２、・・・）**
values(データ１、データ２、・・・)；

```
sqlite> insert into products(id , name) values("A001" , "リンゴ");
sqlite> insert into products values("A002" , "みかん" , "" , 500);
sqlite> insert into products values("A002" , "みかん" , "" , 500);
Error: column id is not unique
```

３）レコードの表示

テーブルに記録されているレコードを表示します。

コマンド：**select 表示するフィールド名 from テーブル名　条件など;**

「表示するフィールド名」に「*」を指定すると、テーブルに含まれるすべてのフィールドが表示されます。複数個のフィールドを指定したいときは、「,」で区切って並べて書きます。

```
sqlite> select * from products;
A1001|リンゴ||
A1002|みかん||500
sqlite>
```

４）レコードの更新

記録されているレコードの内容を変更します。条件を付けて、特定のレコードだけを更新することもできます。

コマンド：**update　テーブル名　　set フィールド名=値　条件;**

指定のフィールドの内容を指定の「値」で変更します。複数のフィールドの値を変更したいときは、「,」で区切って並べて書きます。

※条件の指定は select 文と同じなので、後の節を参照してください。

```
sqlite> update products set price = 350;
sqlite> select * from products;
A1001|リンゴ||350
A1002|みかん||350
sqlite>
```

５）レコードの削除

テーブルからレコードを削除します。条件を付けて、特定のレコードだけを削除することもできます。

コマンド：**delete　from テーブル名　条件;**

※　条件の指定は select 文と同じなので、後の節を参照してください。

```
sqlite> delete from products;
sqlite> select * from products;
sqlite>
```

６）テーブルの削除

記録されているレコードだけでなく、テーブルそのものを削除します。

コマンド：**drop table テーブル名;**

```
sqlite> drop table products;
sqlite> .tables
sqlite>
```

７）データの選択

SQL の中でももっとも重要なのは、適切なレコードと、そこに保存されているデータを読み出すコマンドです。ここでは、いろいろな条件をつけてレコードを選択したり、２つのテーブルを連結したりするコマンドを紹介します。

テーブルに記録されているレコードの中から条件を満たすレコードだけを選択して表示します。

コマンド：**select 表示するフィールド名 from テーブル名　where 条件;**

主な条件

条件	意味	例
＝	一致する	where field1 = 10
<	より小さい	where field1 < 10
>	より大きい	where field1 > 10
<=	以下	where field1 <= 10
>=	以上	where field1 >= 10
<>	一致しない	where field1 <> 10
is NULL	データなし	where field1 is null
between (from) and (to)	範囲指定	where field1 between 10 and 15
like %	文字を含む	where field2 like '%e%'; (e を含むもの)
And	条件の AND	where field1 < 10 and field2 IS null
Or	条件の OR	where field1 < 10 or field2 IS null
Not	条件の否定	where not field1 < 10

例)

```
create table products(
        id      char(10) primary key,
        name    text,
        image   text,
        price   integer
);
```

```
insert into products values("A001" , "りんご" , "apple.jpg" , 200);
insert into products values("A002" , "みかん" , "orange.jpg" , 500);
insert into products values("A003" , "バナナ" , "banana.jpg" , 150);
insert into products values("A004" , "メロン" , "melon.jpg" , 800);
insert into products values("A005" , "メロン" , "melon2.jpg" , 1200);
insert into products values("B001" , "鉛筆" , "pencil.jpg" , 150);
insert into products values("B002" , "消しゴム" , "eraser.jpg" , 50);
insert into products values("B003" , "定規" , "melon.jpg" , 250);
insert into products (id , name , price) values("B004" , "コピー用紙" ,  450);
insert into products values("B005" , "輪ゴム" , "" , 90);
```

```
sqlite> select * from products where id="A001";
A001|りんご|apple.jpg|200
sqlite> select * from products where price < 200;
A003|バナナ|banana.jpg|150
B001|鉛筆|pencil.jpg|150
B002|消しゴム|eraser.jpg|50
B005|輪ゴム||90
sqlite> select * from products where id like "B%";
B001|鉛筆|pencil.jpg|150
B002|消しゴム|eraser.jpg|50
B003|定規|melon.jpg|250
B004|コピー用紙||450
B005|輪ゴム||90
sqlite> select * from products where image is null;
B004|コピー用紙||450
sqlite> select * from products where image ="";
B005|輪ゴム||90
```

８）　　レコードを並べ替える

指定のフィールドの値の昇順、降順に並べ替えてレコードを取り出します。

コマンド（昇順）：**select 表示するフィールド名 from テーブル名　order by フィールド名 ；**
コマンド（降順）：**select 表示するフィールド名 from テーブル名　order by フィールド名　DESC;**

where 句と組み合わせて、特定の条件のレコードのみを並べ替えることもできます。

```
sqlite> select  * from products order by price desc;
A004|メロン|melon.jpg|800
A002|みかん|orange.jpg|500
B004|コピー用紙||450
B003|定規|melon.jpg|250
A001|りんご|apple.jpg|200
A003|バナナ|banana.jpg|150
B001|鉛筆|pencil.jpg|150
B005|輪ゴム||90
B002|消しゴム|eraser.jpg|50
sqlite> select  * from products where id like 'B%' order by price desc;
B004|コピー用紙||450
B003|定規|melon.jpg|250
B001|鉛筆|pencil.jpg|150
B005|輪ゴム||90
B002|消しゴム|eraser.jpg|50
sqlite>
```

９）　　重複データを１つだけ選択

フィールドに同じデータが複数あるとき、１つだけ選択することができます。

コマンド：**select distinct 表示するフィールド名 from テーブル名　；**

```
sqlite> select name from products where price > 500;
メロン
メロン
sqlite> select distinct name from products where price > 500;
メロン
```

フィールド名の別名

フィールド名に別名をつけることができます。

```
mysql> select ID as number , NAME as subject from subject1;
+--------+----------+
| number | subject  |
+--------+----------+
|      1 | 国語     |
|      2 | 数学     |
    :        :
```

フィールド名が長くなるとき別名をつけておくと便利です。

５．データベースの正規化

伝票処理を考えます。

請求書

伝票番号：1

2012 年 6 月 30 日

東京都品川区大崎 x-x-x
品川　太郎様

東京都大田区西蒲田 1-1-1
通販文具店

品番	商品名	単価	個数	金額
N001	A4 標準ノート	245	1	245
P003	鉛筆 HB	1000	2	2000
P005	マジック	150	10	1500
			合計金額	3745

上記のとおり、請求いたします。

請求書 1 枚に対し、購入商品の種類の数は可変です。
このようなときは、1 枚の請求書に 1 回だけ書かれている情報（日付・お客様の情報・伝票番号）と複数書かれている情報（購入商品に関する情報）とを分けて記録します。データを重複して保存することを避けられます。計算すれば求まる情報も記録はしません。
このようにテーブルを分割して設計することをデータベースの正規化といいます。

このような請求書がたくさんあった時、これをそのまま表にすると

伝票番号	日付	顧客名	住所	品番	商品名	単価	個数	金額
1	2012/6/30	品川太郎	東京都・・	N001	A4 標準ノート	245	1	245
				P003	鉛筆 HB	1000	2	2000
				P005	マジック	150	10	1500
2	2012/6/30	鈴木一郎	アメリカ・・・	N001	A4 標準ノート	245	10	2450
3	2010/6/30	野原花子	東京都・・・	P004	シャープペンシル	750	2	1500
				P005	マジック	150	2	300

表にあいているところがあると、データベースでは困りますので、全部埋めます。これが第一正規形です

伝票番号	日付	顧客名	住所	品番	商品名	単価	個数	金額
1	2012/6/30	品川太郎	東京都・・	N001	A4 標準ノート	245	1	245
1	2012/6/30	品川太郎	東京都・・	P003	鉛筆 HB	1000	2	2000
1	2012/6/30	品川太郎	東京都・・	P005	マジック	150	10	1500
2	2012/6/30	鈴木一郎	アメリカ・・・	N001	A4 標準ノート	245	10	2450
3	2010/6/30	野原花子	東京都・・・	P004	シャープペンシル	750	2	1500
3	2010/6/30	野原花子	東京都・・・	P005	マジック	150	2	300

重複を避けるためには、表を分割します。これが第 2 正規形です

伝票番号	日付	顧客名	住所
1	2012/6/30	品川太郎	東京都・・
2	2012/6/30	鈴木一郎	アメリカ・・・
3	2010/6/30	野原花子	東京都・・・

伝票番号	明細番号	品番	商品名	単価	個数	金額
1	1	N001	A4 標準ノート	245	1	245
1	2	P003	鉛筆 HB	1000	2	2000
1	3	P005	マジック	150	10	1500
2	1	N001	A4 標準ノート	245	10	2450
3	1	P004	シャープペンシル	750	2	1500
3	2	P005	マジック	150	2	300

顧客にとっては住所は一つ、商品番号がわかれば商品名や単価は決まります。また、金額は、単価と個数がわかれば計算することができます。さらにテーブルを分割しましょう。これを第 3 正規形といいます

伝票番号	日付	顧客番号
1	2012/6/30	121001
2	2012/6/30	121002
3	2010/6/30	121003

顧客番号	顧客名	住所
121001	品川太郎	東京都・・
121002	鈴木一郎	アメリカ・・・
121003	野原花子	東京都・・・

伝票番号	明細番号	品番	個数
1	1	N001	1
1	2	P003	2
1	3	P005	10
2	1	N001	10
3	1	P004	2
3	2	P005	2

品番	商品名	単価
N001	A4 標準ノート	245
P003	鉛筆 HB	1000
P005	マジック	150
P004	シャープペンシル	750

サンプルデータを作成してください

```
create table sales_book(
        id integer      primary key  AUTOINCREMENT,
        sale_date date,
        customer_id char(10)
);

insert into sales_book (sale_date , customer_id) values ("2012/6/30" , 121001);
insert into sales_book (sale_date , customer_id) values ("2012/6/30" , 121002);
insert into sales_book (sale_date , customer_id) values ("2012/6/30" , 121003);


create table account (
        sale_id integer ,
        account_id ,
        product_id char(10),
        number integer,
        PRIMARY KEY(sale_id , account_id)
) ;


insert into account values(1,1,"N001",1);
insert into account values(1,2,"P003",2);
insert into account values(1,3,"P005",10);
insert into account values(2,1,"N001",10);
insert into account values(3,1,"P004",2);
insert into account values(3,2,"P005",2);

create table customer(
        customer_id char(10) primary key,
        name    text,
        address text
);

insert into customer values("121001","品川太郎","東京都");
insert into customer values("121002","鈴木一郎","アメリカ");
```

```
insert into customer values("121003","野原花子","東京都");



create table product(
        product_id      char(5) primary key,
        name    text,
        price   integer
);

insert into product values("N001","A4 標準ノート" , 245);
insert into product values("P003","鉛筆 HB",1000);
insert into product values("P005","マジック",150);
insert into product values("P004","シャープペンシル",750);
```

テーブルの結合

ほかのテーブルからデータを得るには、テーブルを結合します。

```
select フィールド名 from テーブル名 1,テーブル名 2
                    where テーブル名 1.フィールド名 1 = テーブル名 2.フィールド名 2
```

伝票と明細の結合

```
sqlite> select * from sales_book , account where sales_book.id = account.sale_id;
1|2012/6/30|121001|1|1|N001|1
1|2012/6/30|121001|1|2|P003|2
1|2012/6/30|121001|1|3|P005|10
2|2012/6/30|121002|2|1|N001|10
3|2012/6/30|121003|3|1|P004|2
3|2012/6/30|121003|3|2|P005|2
```

伝票と顧客の結合

```
sqlite> select * from sales_book , customer where sales_book.customer_id = customer.customer_id;
1|2012/6/30|121001|121001|品川太郎|東京都
2|2012/6/30|121002|121002|鈴木一郎|アメリカ
3|2012/6/30|121003|121003|野原花子|東京都
```

明細と商品の結合

```
sqlite> select * from account , product where account.product_id = product.product_id;
1|1|N001|1|N001|A4 標準ノート|245
1|2|P003|2|P003|鉛筆 HB|1000
1|3|P005|10|P005|マジック|150
2|1|N001|10|N001|A4 標準ノート|245
3|2|P005|2|P005|マジック|150
```

伝票と顧客と明細と商品の結合

```
sqlite> select * from sales_book , customer , account , product
   ...>         where sales_book.id = account.sale_id and
   ...>         sales_book.customer_id = customer.customer_id and
   ...>         account.product_id = product.product_id;
1|2012/6/30|121001|121001|品川太郎|東京都|1|1|N001|1|N001|A4 標準ノート|245
1|2012/6/30|121001|121001|品川太郎|東京都|1|2|P003|2|P003|鉛筆 HB|1000
1|2012/6/30|121001|121001|品川太郎|東京都|1|3|P005|10|P005|マジック|150
2|2012/6/30|121002|121002|鈴木一郎|アメリカ|2|1|N001|10|N001|A4 標準ノート|245
3|2012/6/30|121003|121003|野原花子|東京都|3|2|P005|2|P005|マジック|150
```

SQL には、次のように集計を行う構文が用意されています。

集計関数

● フィールドの合計を求める

```
select sum(フィールド名) from テーブル名
```

例）明細テーブルの個数の合計を求める

```
select sum(number) from account;
```

例）明細テーブルの金額の合計を求める

```
select sum(number*price) from account,product where account.product_id = product.product_id;
```

● レコードの件数を数える

```
select count(フィールド名) from テーブル名
```

例）明細テーブルの id が N001 のものの件数を求める

```
select count(account_id ) from account where product_id = "N001";
```

● フィールドの最大値を求める

```
select max(フィールド名) from テーブル名
```

例）商品テーブルの一番高い商品を求める

```
select product_id , max(price) , name from product;
```

● フィールドの最小値を求める

```
select min(フィールド名) from テーブル名
```

例）売れた個数の最小値を求める

```
select min(number) from account;
```

● フィールドの平均値を求める

```
select avg(フィールド名) from テーブル名
```

例）明細テーブルの個数の平均値を求める

```
select avg(number) from account;
```

グループ化

これらの集計関数をグループごとに求めることができます。どのフィールドでグループ化するか、を次のように指定します

```
select 集計関数 from テーブル名 group by フィールド名
```

例）商品別に注文個数を多い順に表示

```
select product_id  , sum(number) as sum_number from account
            group by product_id order by sum_number desc;
```

例）商品別に売り上げ高の多い順に表示

```
select product.product_id , sum(number*price) as total_sales
from account , product where account.product_id = product.product_id
group by product.product_id order by total_sales desc;
```

例）顧客別売上ランキング

```
select customer.customer_id ,customer.name, sum(number*price) as total_sales
        from account , product , sales_book , customer
        where sales_book.id = account.sale_id
        and account.product_id = product.product_id
        and customer.customer_id = sales_book.customer_id
        group by customer.customer_id order by total_sales desc;
```