1　データベースの基本

1-1　データベースとは

データベースとは、特定のテーマや目的に合わせて集められたデータを、後から抽出したり、分析したりできるように一定の規則にしたがって整理し、保存されたもので、重複や矛盾のないデータの集まりをいいます。データ管理には、次のような３つの方法があります。

図１の「生徒情報」では、用紙１枚ごとにナンバー、氏名、住所、出身中学、所属クラブ等を記入していきます。これは**カード型**データであり、ワープロソフトで行う領域です。

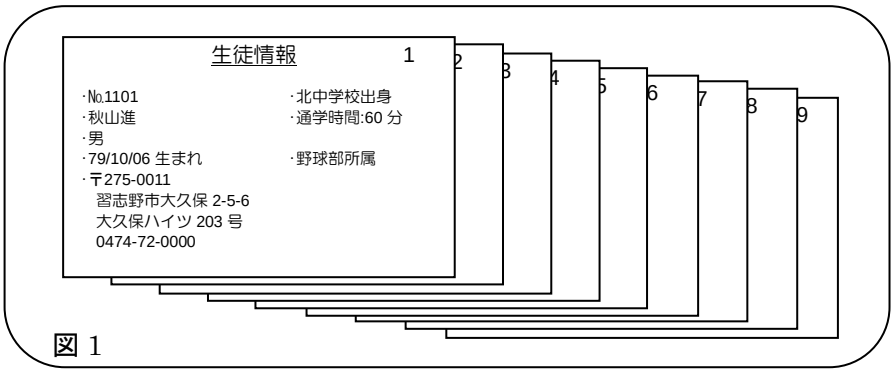


図 1

図２は、**集計表形式**でデータを一元管理する方法で、表計算ソフトで行う領域です。

生徒番号	氏名	性別	生年月日	出身中学	所属クラブ	郵便番号	
1101	秋山 進	男	79/10/05	北中	野球部	275-0011	習志野
1102	岩瀬 一郎	男	79/12/25	西中	ｻｯｶｰ部	272-0115	市川市
1103	遠藤 喜一	男	80/03/10	東中	なし	274-0077	船橋市
1104	大西 守	男	79/05/25	第二中	ｻｯｶｰ部	274-0824	船橋市
1105	北川 文彦	男	79/08/15	大山中	野球部	263-0033	千葉市
1106	笹井 秀次	男	79/09/24	北中	ﾃﾆｽ部	289-0346	香取郡
1107	中野 宏志	男	80/02/15	千葉中	ｻｯｶｰ部	283-0811	東金市
1108	西川 正則	男	79/12/10	南中	美術部	270-0112	流山市

図 2

図３は、複数の表(テーブル)を関連づけ(リレーショナル)、１つのデータベースとして管理する方法で、**リレーショナルデータベースソフト**が行う領域です。

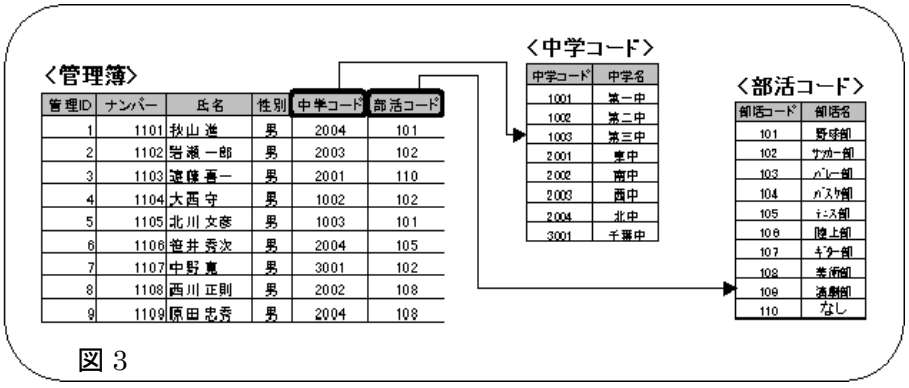


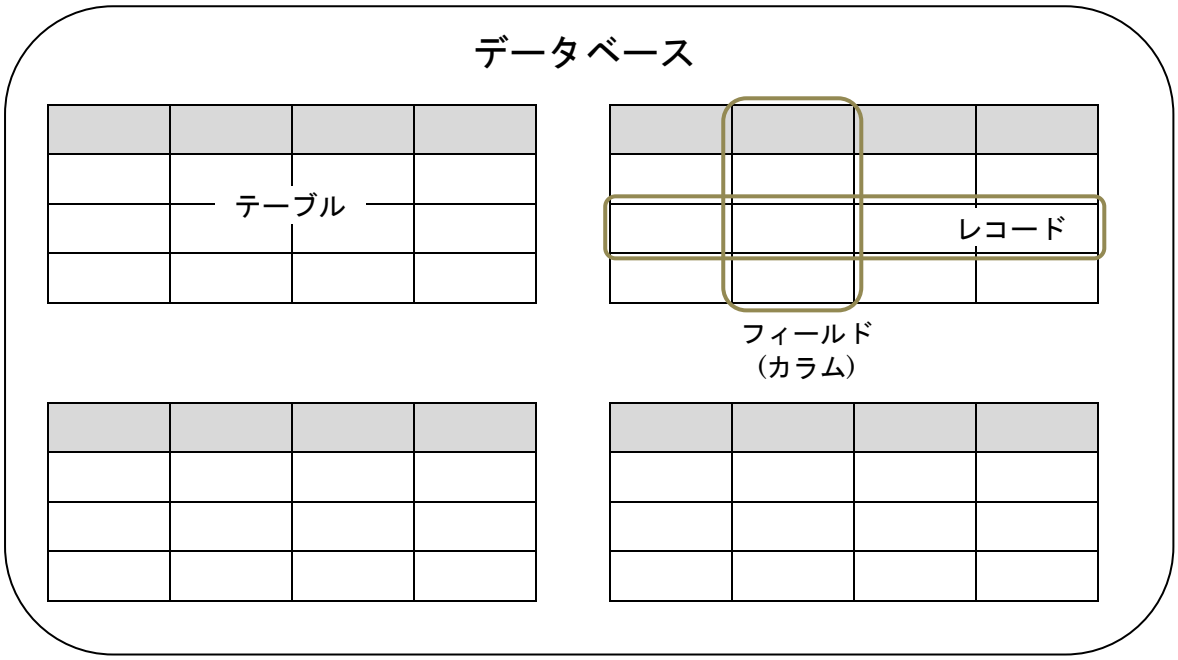
図 3

リレーショナルデータベースソフトの本質的な機能は「**データの管理と抽出**」であり、必要なときに必要なデータを取り出して、利用できるところにあります。

1-2　データベースの構造

リレーショナルデータベースにおいては関連するデータをすべて**テーブル**と呼ばれる 2 次元の表に格納します。

テーブルの横一列の行のことをレコードといい、レコードに含まれる個々の項目(縦の列)のことをカラムまたはフィールドといいます。



テーブルを作成するとき、それぞれのカラムにどのようなデータを格納するかあらかじめ決めておく必要があります。データに格納される値の形式が不揃いだった場合、検索時に意図したデータの抽出ができなくなるからです。テーブルを作成するときにはそれぞれのカラムに**データ型**をセットしておく必要があります。

【主なデータ型】

データ型	データ型名
数値型	TINYINT
	SMALLINT
	MEDIUMINT
	INT
	BIGINT
	FLOAT
	DOUBLE(n, d)
	NUMERIC(n, d)

データ型	データ型名
文字列型	CHAR(n)
	VARCHAR(n)
	TINYTEXT
	TEXT
	MEDIUMTEXT
	LONGTEXT

データ型	データ型名
日付型	DATETIME
	DATE
	TIME
	TIMESTAMP
	YEAR

データ型	データ型名
バイナリ型	TINYBLOB
	BLOB
	MEDIUMBLOB
	LONGBLOB
その他	ENUM
	SET

カラムの性質をきめるのはデータ型だけではなく、データの内容を規定するためのさまざまな条件もあります。例えば、あるテーブルに含まれるレコードを一意に特定する番号をもつカラムを**主キー**(**PRIMARY KEY**)といいます。

主キーを参照するキーのことを**外部キー**(**Foreign Key**)といいます。RDB ではこのようにテーブル間の主キーと外部キーとで対応関係を持たせることで、お互いのテーブルを関連付けることができます。この関連付けのことを**リレーションシップ**といい、主キーと外部キーとの対応関係に対する制約条件のことを**参照制約**といいます。

その他にも、非 NULL 制約(値が空であることを許さない)や UNIQUE 制約(重複した値を許さない)などもあります。これらについては後から紹介していきます。

2　MySQL の準備

1-1　準備

(1)　講座用データベースを展開する

講座用データベースを MySQL サーバの workbook データベースに展開します。workbook は今回使用するデータベース名です。

①　「workbook.sql」があることを確認

②　コマンドプロンプトを起動して以下のように入力

```
> mysqladmin -u root -p ping
Enter password: *****
mysqld is alive
```

パスワード(root1)を入力

MySQL が正常動作していることを確認できる

③　データベースを展開するため、以下のように入力

```
> mysqladmin -u root -p CREATE workbook
password: *****

> mysql -u root -p workbook < 絶対パス名¥workbook.sql
password: *****
```

パスワード(root1)を入力

workbook という名のデータベースを生成、workbook.sql を実行する

※　データベースを破棄したい、初期状態に戻したい場合は、以下のコマンドを入力してデータベースを破棄する

```
> mysqladmin -u root -p DROP workbook
password: *****
```

パスワード(root1)を入力

(2) MySQL Monitor の起動

MySQL には標準で MySQL クライアント(MySQL Monitor)というデータベースを操作するツールが添付されています。

MySQL Monitor はコマンドプロンプト上で動作する CUI(Character User Interface)ベースのツールで、ユーザはプロンプト経由で SQL 命令を発行したり、その結果を確認したりすることができます。

① MySQL Monitor を起動して、日本語指定も行う

```
> mysql -u root -p --default-character-set=sjis
password: *****
```

パスワード(root1)を入力

② データベースを指定する

```
mysql> USE workbook;
Database changed
```

workbook データベースに移動する　以降の SQL 命令はすべて workbook データベース上で実行される

③ データベースの内容を確認する

```
mysql> SHOW TABLES;
+--------------------+
| Tables_in_workbook |
+--------------------+
| access_log         |
| author             |
  ... 中略 ...
| time_card          |
| usr                |
+--------------------+
18 rows in set (0.00 sec)
```

SHOW TABLES 命令は現在選択中のデータベース上に存在するテーブル名を表示する

個別のテーブルに属するカラムの一覧を表示する

```
mysql> SHOW FIELDS FROM books;
+--------------+-------------+------+-----+---------+-------+
| Field        | Type        | Null | Key | Default | Extra |
+--------------+-------------+------+-----+---------+-------+
| isbn         | char(17)    | NO   | PRI |         |       |
| title        | varchar(255) | YES  |     | NULL    |       |
| price        | int(11)     | YES  |     | NULL    |       |
| publish      | varchar(30) | YES  |     | NULL    |       |
| publish_date | date        | YES  |     | NULL    |       |
| category_id  | char(5)     | YES  |     | NULL    |       |
+--------------+-------------+------+-----+---------+-------+
6 rows in set (0.02 sec)
```

④ MySQL Monitor を終了する

```
mysql> exit;
Bye
>
```

Monitor が終了する

3　基本的な検索

SQL はリレーショナルデータベースを操作するためのさまざまな機能を提供します。その中でも頻繁に利用し、活用できる機能も多岐にわたるのが SELECT 命令です。

SELECT 命令を確認していきます。

3-1　テーブルから全ての列を取り出す

手始めにもっとも基本的なテーブルの検索を行ってみましょう。

構文〉 SELECT 命令①

```
SELECT 列名 FROM テーブル名;
```

列名	テーブルに属するカラム名を指定 複数指定する場合は「,」で羅列する 全てのカラムを指定する場合は「*」を記述する
テーブル名	検索するテーブル名を指定

・　アンケートテーブル(quest)から全てのデータを取得し、表示してみます。

```
> SELECT
>   *
> FROM
>   quest
> ;
```

SQL 文は改行を意識しませんが、句の区切り目で改行を入れて構造がわかりやすいようにしてあります。また、予約語の大文字小文字も無視されますが、テーブル名、列名と区別するために予約語は大文字で示します。

PRACTICE SELECT 命令①

Q1 書籍情報テーブル(books)から全ての列を取り出してみましょう。以下の空欄を埋めて SQL 命令を完成させてください。また取得できた行数も記述してください。

```
[    ①    ]
  *
FROM
  [    ②    ]
;
```

取得行数＿＿＿＿＿

Q2 月間売り上げテーブル(sales)から全ての列を取り出すための SQL 命令を記述してみましょう。また取得できた行数も記述してください。

取得行数＿＿＿＿＿

Q3 アンケートテーブル(quest)から全ての列を取り出すための SQL 命令を記述してみましょう。また取得できた行数も記述してください。

取得行数＿＿＿＿＿

3-2　テーブルから特定の列を取り出す

前節では全ての列を表す「*」について確認しました。「*」はとりあえず全てのデータを取り出したいとき便利ですが、以下の理由から「*」を多用するのはお勧めできません。

・不要なデータが結果に含まれることで、表が見難い

・データベースサーバやネットワークに余計な負荷がかかる

特定の列を指定して検索し、情報を表示してみましょう。

構文〉 SELECT 命令②

```
SELECT 列名 1, 列名 2,... FROM テーブル名;
```

列名	テーブルに属するカラム名を指定 具体的列名を「,」で羅列する

・　アンケートテーブル(quest)から回答者の名前と性別、年齢だけを取得し、表示してみます。

```
> SELECT
>   name,
>   sex,
>   age
> FROM
>   quest
> ;
```

列名の順番は必ずしもテーブルの定義順に合わせる必要はありません。列名の指定順番を変えると検索結果の表示順序が変わります。

```
> SELECT
>   sex,
>   name,
>   age
> FROM
>   quest
> ;
+-----+------------+------+
| sex | name       | age  |
+-----+------------+------+
| 男  | 山田太郎   |   30 |
| ... 後略 ...            |
+-----+------------+------+
16 rows in set (0.03 sec)
```

PRACTICE SELECT 命令②

Q1 商品テーブル(product)から商品名と単価を取り出してみましょう。以下の空欄を埋めて SQL 命令を完成させてください。また取得できた行数も記述してください。

```
SELECT
  [    ①    ]
  [    ②    ]
 [    ③    ]
  product
;
```

取得行数＿＿＿＿＿＿

Q2 社員テーブル(employee)から社員の氏、名、役職の列を取り出すための SQL 命令を記述してみましょう。また取得できた行数も記述してください。

取得行数＿＿＿＿＿＿

Q3 書籍情報テーブル(books)から書籍名、刊行日、価格の列を取り出すための SQL 命令を記述してみましょう。また取得できた行数も記述してください。

取得行数＿＿＿＿＿＿

3-3　重複した行を取り除く

SELECT した情報から重複を取り除くにはキーワードを使用します。DISTINCT キーワードを列名の前に指定することで重複のない情報を検索できます。

DISTINCT キーワードを指定しないと、取得した全ての行を取得することになります。ALL キーワードを使うと全ての行を取得することを明示できます。

構文〉 DISTINCT キーワード

```
SELECT DISTINCT 列名 1, 列名 2,... FROM テーブル名;
```

DISTINCT	重複抜きで取得
ALL 又は指定なし	全ての取得

・ 書籍情報テーブル(books)から出版社列を取得します。重複した行は取り除いたものを出力します。

```
> SELECT DISTINCT
>   publish
> FROM
>   books
> ;
```

PRACTICE DISTINCT キーワード

Q1 アンケートテーブル(quest)から都道府県名を重複のない形式で取り出してみましょう。以下の空欄を埋めてSQL命令を完成させてください。また取得できた行数も記述してください。

```
SELECT [    ①    ]
  prefecture
FROM
  [    ②    ]
;
```

取得行数＿＿＿＿＿

Q2 アクセス記録テーブル(access_log)からリンク元URLを重複のない形式で取り出すためのSQL命令を記述してみましょう。また取得できた行数も記述してください。

取得行数＿＿＿＿＿

Q3 ユーザテーブル(usr)からユーザ氏名を、同姓同名のユーザは取り除いて取り出すためのSQL命令を記述してみましょう。また取得できた行数も記述してください。

取得行数＿＿＿＿＿

3-4　条件に合致した行を取り出す

特定の条件に絞って、合致したデータを取り出すことができます。

データの取得条件を指定するには WHERE キーワードを使用します。WHERE キーワードと後続の条件式とを合わせて、「WHERE 句」と言います。

構文〉 WHERE 句

```
SELECT 列名,... FROM テーブル名 WHERE 条件式;
```

条件式	列名 演算子 検索値 ↑ 比較演算子

条件式で使用する演算子は比較演算子です。演算子にはその他にも目的に応じて算術演算子や論理演算子等がありますが、これらについては後述します。

演算子	概要	例
=	等しい	sex = '男'
<>	等しくない	sex <> '男'
>	より大きい	age > 20
<	未満	age < 20
>=	以上	age >= 20
<=	以下	age <= 20
[NOT] LIKE	指定パターンを含む[含まない]	name LIKE '山%'
IS [NOT] NULL	NULL である[ではない]	name IS NULL
[NOT] IN	候補値のいずれかである[ではない]	name IN('山田', '山口')
[NOT] BETWEEN	X～Y の間である[間にない]	age BETWEEN 10 AND 20

比較演算子では指定されたカラムの値と条件値とを比較して、条件に合致した場合に「正しい(真：True)」という結果を、合致しない場合に「正しくない(偽：false)」という結果を返します。WHERE 句では条件式が True とみなされたレコードだけを返します。

！POINT！

NULL：値が未定義であることを示す特別な値
空文字列「''(シングルクォート 2 つ)」や 0 値とは異なる

・ アンケートテーブル(quest)から「女」が回答した結果だけを取得します。取り出す列は name、answer1、answer2 列とします。

```
> SELECT
>   name,
>   answer1,
>   answer2
> FROM
>   quest
> WHERE
>   sex = '女'
> ;
```

PRACTICE WHERE 句 (1/2)

Q1 アンケートテーブル(quest)から 20 歳以上の人間による回答だけを取り出してみましょう。取り出すのは answer1、answer2 列だけとします。また取得できた行数も記述してください。

```
SELECT
  [    ①    ]
FROM
  quest
WHERE
  [    ②    ]
;
```

取得行数＿＿＿＿＿

Q2 アンケートテーブル(quest)から回答日時が「2012/01/01」以降の情報のみを取り出してみましょう。取り出す列は name、answer1、answer2 列とします。また取得できた行数も記述してください。

取得行数＿＿＿＿＿

Q3 貸し出し記録テーブル(rental)から貸し出し中の情報だけを取り出してみましょう。取り出す列は、user_id、isbn 列とします。また取得できた行数も記述してください。

取得行数＿＿＿＿＿

PRACTICE WHERE 句(2/2)

Q4 書籍情報テーブル(books)から出版社が「海原出版」「常和システム」である書籍情報だけを取り出してみましょう。取り出すのは isbn、title、publish 列とします。また取得できた行数も記述してください。

```
SELECT
  [    ①    ]
FROM
  books
WHERE
  [    ②    ]
;
```

取得行数＿＿＿＿＿

Q5 ユーザテーブル(usr)から「東京」に住んでいない人の情報だけを取り出してみましょう。取り出す列は l_name、f_name、email 列とします。また取得できた行数も記述してください。

取得行数＿＿＿＿＿

Q6 アンケートテーブル(quest)から年齢が 30 歳以上 40 歳未満の人の回答のみを取り出してみましょう。取り出す列は、name、sex、prefecture 列とします。また取得できた行数も記述してください。

取得行数＿＿＿＿＿

3-5　複数条件に合致したレコードだけを取り出す

WHERE 句に複数の条件を指定して、より複雑な条件でデータを検索することもできます。

WHERE 句の下では、条件式を論理演算子によって接続することができます。論理演算子には AND(論理積・かつ)、OR(論理和・または)があります。

構文〉 論理演算子を用いた条件式

```
SELECT 列名,... FROM テーブル名
        WHERE 条件式 論理演算子 条件式 ･･･;
```

論理演算子とは前後の条件式が True／False であるかを判定して、複数の条件式全体として True／False いずれであるかを返すものです。WHERE 句は条件式全体が True となる行のみを抽出します。

- 書籍情報テーブル(books)から出版社が「ソソム」または「飛翔社」で、かつ、価格が 3,000 円以上のものだけを取り出してみます。取り出す列は isbn、title、price 列とします。

```
> SELECT
>   isbn,
>   title,
>   price
> FROM
>   books
> WHERE
>     publish IN ('ソソム', '飛翔社')
>   AND
>     price >= 3000
> ;
```

PRACTICE 論理演算子

Q1 アンケートテーブル(quest)から性別が女であり、かつ、年齢が 20 歳代である回答者の情報のみを取り出してみましょう。取り出す列は全列とします。また取得できた行数も記述してください。

取得行数＿＿＿＿＿

Q2 ユーザテーブル(usr)から東京都在住で、かつ、E-Mail アドレスのドメインが examples.com のユーザ情報を取り出してみましょう。取り出す列は l_name、f_name、email 列とします。また取得できた行数も記述してください。

取得行数＿＿＿＿＿

Q3 アンケートテーブル(quest)から感想欄が空でないものを取り出してみましょう。取り出す列は answer2 列のみとします。また取得できた行数も記述してください。

取得行数＿＿＿＿＿

3-6　特定の列を使って行を並べ替える

特定の列をキーにレコードの並べ替えを行うことができます。

データの並べ替えを指定するのはORDER BYキーワードを使用します。ORDER BYキーワードと後続の並べ替えキーとを合わせて、「ORDER BY句」と言います。

構文〉 ORDER BY **句**

```
SELECT 列名,... FROM テーブル名 ORDER BY ソート条件;
```

ソート条件	並べ替えのキーを指定 並べ替え順として、ASC(昇順)かDESC(降順)を指定 並べ替え順を省略すると昇順とみなされる

・　アンケートテーブル(quest)から感想欄が空欄でないデータを抽出してみます。その際、本書の評価が低い順に結果を並べ替えます。取り出す列はanswer1、answer2列とします。

```
> SELECT
>   answer1,
>   answer2
> FROM
>   quest
> WHERE
>     answer2 IS NOT NULL
>   OR
>     answer2 <> ''
> ORDER BY
>   answer1 ASC
> ;
```

「OR」はWHERE句で複数条件を指定する際の論理演算子になります。OR演算子は論理和「または」、ほかにAND演算子で論理積「かつ」があります。

PRACTICE ORDER BY 句

Q1 書籍情報テーブル(books)から価格が2500～3500円の範囲の書籍を価格が安い順に取り出してみましょう。取り出す列はtitle、price列とします。また取得できた行数も記述してください。

```
SELECT
  title,
  price
FROM
  books
WHERE
  [    ①    ]
ORDER BY
  [    ②    ]
;
```

取得行数＿＿＿＿＿＿

Q2 ユーザテーブル(usr)から東京都、千葉県、神奈川県に住んでいる人のリストを姓(カナ)、名(カナ)について昇順で取り出してみましょう。取り出す列はl_name、f_name、prefecture列とします。また取得できた行数も記述してください。

取得行数＿＿＿＿＿＿

3-7　特定の列を使ってデータをグルーピングする

ある特定のカテゴリ単位でデータを集計することができます。

データを集計するにはGROUP BYキーワードを使用します。GROUP BYキーワードと後続のグルーピングキーとを合わせて、「GROUP BY句」と言います。

構文〉 GROUP BY句

```
SELECT グループ化列、集計列
        FROM テーブル名 GROUP BY グループ化キー;
```

グループ化列	グループ化キー列
集計列	集計関数の集計対象となる列
グループ化キー	どの列でグルーピングするかを指定 並べ替え順として、ASC(昇順)かDESC(降順)を指定 並べ替え順を省略すると昇順とみなされる

関数とは、指定された列(値)を予め決められたルールに従って処理し、その結果を返すものです。SQLでは標準で利用可能なさまざまな関数が用意されており、その中でもGROUP BY句と共に使用して集計処理を行う役割を持つ関数の事を、集計関数と呼んでいます。

関数	概要
AVG(列名)	平均値
COUNT(列名)	対象列の件数
MAX(列名)	最大値
MIN(列名)	最小値
SUM(列名)	合計値

GROUP BYキーワードでグループ化を行う場合、取得列にはグループ化キーと集計関数の集計対象となる列しか指定できない点に気をつけましょう。

・ 書籍情報テーブル(books)を出版社ごとにグルーピングし、各社ごとの価格の平均を求めてみます。取り出す列はpublish、price列の平均とします。

```
> SELECT
>   publish,
>   AVG(price)
> FROM
>   books
> GROUP BY
>   publish
> ;
```

集計関数以外にも文字列関数、日付関数、算術関数が用意されています。

分類	関数	概要
文字列関数	ASCII(x)	指定された文字xに対応する文字コードの取得
	CHAR(n)	文字コードを文字に変換
	CHAR_LENGTH(x)	文字列の文字数を取得
	POSITION(sx IN x)	指定文字列の出現位置を取得
	CONCAT(x, x, ･･･)	文字列の結合
	REPLACE(x, o, n)	文字列の置換
	LTRIM(x)	文字列の左側から半角スペースを削除
	RTRIM(x)	文字列の右側から半角スペースを削除
	SUBSTRING(x, n, l)	文字列から指定位置にある文字列を取得
	LOWER(x)	英大文字を小文字に変換
	UPPER(x)	英小文字を大文字に変換

PRACTICE GROUP BY 句

Q1 アンケートテーブル(quest)から性別ごとにそれぞれ年齢の最大／最小値を取り出してみましょう。また取得できた性別ごとの年齢の最大／最小値も記述してください。

男性：最高年齢　　、最少年齢

女性：最高年齢　　、最少年齢

Q2 月間売り上げテーブル(sales)で店舗別の累計売上高を取り出してみましょう。また取得できた行数も記述してください。

取得行数

Q3 アクセス記録テーブル(access_log)からメニューコード別のアクセス数を取り出してみましょう。また取得できた行数も記述してください。

取得行数

3-8　グルーピング結果に条件を付与する

グルーピングした結果値に対して絞り込み条件を付与することができます。
グルーピングした結果に対して絞り込み条件を設定するには HAVING 句を使用します。

構文〉 HAVING 句

```
SELECT グループ化列、集計列
        FROM テーブル名 GROUP BY グループ化キー
                HAVING 条件式;
```

・ アンケートテーブル(quest)から都道府県ごとの評価で、平均が 2 未満のデータだけを取り出してみます。

```
> SELECT
>   prefecture,
>   AVG(answer1)
> FROM
>   quest
> GROUP BY
>   prefecture
> HAVING
>   AVG(answer1) < 2
> ;
```

PRACTICE HAVING 句

Q1 アンケートテーブル(quest)から都道府県ごとに年齢の平均を求め、35 歳以上 50 歳未満のデータのみを取り出してみましょう。また取得できた行数も記述してください。

取得行数＿＿＿＿＿

Q2 書籍情報テーブル(books)から出版社、分類 ID ごとに登録数を求め、登録数が 3 冊未満の情報だけを取り出してみましょう。また取得できた行数も記述してください。

取得行数＿＿＿＿＿

Q3 社員テーブル(employee)から所属部署ごとの女性の人数を求め、3 人以上の部署だけを取り出してみましょう。また取得できた行数も記述してください。

取得行数＿＿＿＿＿

Q4 アクセス記録テーブル(access_log)からアクセス日時が 2011/01/01 以降のものについて、リンク元 URL ごとにアクセス数を算出し、アクセス数が 5 件未満のもののみをアクセス数降順で取り出す以下の SQL 命令を完成させましょう。また、実行して取得できた行数も記述してください。

```
SELECT
  referer,
  [    ①    ]
FROM
  access_log
WHERE
  [    ②    ]
GROUP BY
  referer
[    ③    ]
  [    ①    ] < 5
ORDER BY
  [    ①    ] [    ④    ]
;
```

取得行数＿＿＿＿＿＿

4　RDB 的な検索

2 次元のテーブルをリレーションシップによってお互いに関連付け、複雑な構造データを表現できるのもリレーショナルデータベースの特長です。

ここでは、お互いに関連付けられたテーブルを結合したり、他のテーブルで得られた結果に基づいて異なるテーブルを更新するような検索を確認していきます。

4-1　内部結合で 2 つのテーブルのデータを結合する

SQL では分割したテーブルをひとつに結合するための機能を提供しています。結合を利用することで、2 つのテーブルに対して 1 つの SQL 命令でアクセスすることが可能になります。

2 つのテーブルを結合するには INNER JOIN...ON 句を使用します。

構文〉 INNER JOIN...ON 句

```
SELECT 列名,... FROM テーブル名 1 INNER JOIN テーブル名 2
    ON テーブル名 1.列名 1 = テーブル名 2.列名 2  WHERE 句など;
```

テーブル名	結合する 2 つのテーブル名を 1 と 2 に記述
結合するキー名	結合する(関連付けする)キーを記述

取得する列名にはどのテーブルに属しているかを「テーブル名.列名」の形式で示します。テーブル名を記述するのは冗長的であるという場合には FROM 句で記述するテーブル名でテーブルの別名を用意することもできます。テーブルに別名を用意するには「テーブル名 AS 別名」と記述します。

・ 社員テーブル(employee)と所属部署テーブル(depart)から氏名と所属部署名、役職を取り出してみます。退職済みの社員は除き、所属部署コード、社員コードについて昇順で出力します。

```
> SELECT
>   employee.l_name,
>   employee.f_name,
>   depart.depart_name,
>   employee.class
> FROM
>   employee
> INNER JOIN
>   depart
> ON
>   employee.depart_id = depart.depart_id
> WHERE
>   employee.retired <> 1
> ORDER BY
>   employee.depart_id ASC,
>   employee.s_id ASC
> ;
```

テーブルの別名を用いる場合は以下の命令になります。

```
> SELECT
>   e.l_name,
>   e.f_name,
>   d.depart_name,
>   e.class
> FROM
>   employee AS e
> INNER JOIN
>   depart AS d
> ON
>   e.depart_id = d.depart_id
> WHERE
>   e.retired <> 1
> ORDER BY
>   e.depart_id ASC,
>   e.s_id ASC
> ;
```

PRACTICE INNER JOIN 句

Q1 社員テーブル(employee)とタイムカードテーブル(time_card)から氏名と 2010 年 12 月の勤務時間を、社員コード、日付について昇順で取り出してみましょう。また取得できた行数も記述してください。

取得行数＿＿＿＿＿＿

Q2 店舗テーブル(shop)と月間売り上げテーブル(sales)から 2010 年 12 月の売り上げを売上高の高い順に取り出してみましょう。また取得できた行数も記述してください。

取得行数＿＿＿＿＿＿

Q3 店舗テーブル(shop)と月間売り上げテーブル(sales)から2010年の全売上高を売上高の低い順に出力してみましょう。また取得できた行数も記述してください。

取得行数＿＿＿＿＿

Q4 貸し出し記録テーブル(rental)とユーザテーブル(usr)からそれぞれのユーザについて、現在何冊の貸し出しを行っているかを、貸し出し数の多い順に取得してみましょう。取得列はユーザ氏名、貸し出し数とし、貸し出し数0のユーザは出力する必要はありません。また取得できた行数も記述してください。

取得行数＿＿＿＿＿

4-2　外部結合で 2 つのテーブルのテータを結合する

内部結合では 2 つのテーブルをキーで結合し、一致するものだけを取得しました。結合する一方のテーブルにしかない情報も取得するには外部結合を使用します。

2 つのテーブルを外部結合するには RIGHT JOIN...ON 句(LEFT JOIN...ON 句)を使用します。

構文〉 RIGHT JOIN...ON 句

```
SELECT 列名,... FROM テーブル名 1 RIGHT JOIN テーブル名 2
    ON テーブル名 1.列名 1 = テーブル名 2.列名 2  WHERE 句など;
```

テーブル名 1	結合するテーブル名を記述　キーに一致する情報のみ出力する
テーブル名 2	結合するテーブル名を記述　全ての情報を出力する

RIGHT JOIN...ON 句を使用すると、JOIN 句の右側で宣言したテーブルの情報を全て出力します。逆に JOIN 句の左側で宣言したテーブルの情報を全て出力したい場合は LEFT JOIN...ON 句を使用します。

- 注文明細テーブル(order_desc)と商品テーブル(product)から商品ごとの累計購入数と累計購入額を購入額が多い順に取り出してみます。まったく売れていない商品についても取り出すものとします。
  注文明細テーブルに存在しないデータも含めて商品テーブルから取り出します。

```
> SELECT
>   p.p_name,
>   SUM(o.quantity),
>   SUM(p.price * o.quantity)
> FROM
>   order_desc AS o
> RIGHT JOIN
>   product AS p
> ON
>   p.p_id = o.p_id
> GROUP BY
>   p.p_id,
>   p.p_name
> ORDER BY
>   SUM(p.price * o.quantity) DESC
> ;
```

PRACTICE RIGHT JOIN 句、LEFT JOIN 句

Q1 社員テーブル(employee)と所属部署テーブル(depart)を結合し、所属社員が一人もいない幽霊部署を取得してみましょう。また取得できた行数も記述してください。

取得行数＿＿＿＿＿

Q2 ユーザテーブル(usr)と貸し出し記録テーブル(rental)からユーザごとの貸し出し数を、貸し出し数が多い順に取得してみましょう。取得列はユーザの氏名と貸し出し件数とし、貸し出し件数 0 のユーザも出力します。また取得できた行数も記述してください。

取得行数＿＿＿＿＿

Q3 書籍情報テーブル(books)と貸し出し記録テーブル(rental)とを結合し、書籍ごとの貸し出し数累計を、累計数が多い順に出力しましょう。貸し出し数が 0 の書籍についても出力します。また取得できた行数も記述してください。

取得行数＿＿＿＿＿

Q4 社員テーブル(employee)とタイムカードテーブル(time_card)を結合し、社員ごとの勤務時間平均を、平均時間の多い順に出力しましょう。タイムカードへの記録がない社員の情報も出力します。出力する列は社員氏名、勤務時間平均です。また取得できた行数も記述してください。

取得行数＿＿＿＿＿

4-3　3 つ以上のテーブルを結合する

結合できるのは 2 つのテーブルだけでなく、3 つ以上のテーブルを結合することができます。3 つ以上のテーブルを結合するには、JOIN 句を入れ子にします。

構文〉　入れ子の JOIN 句

```
SELECT 列名,...
    FROM (テーブル名 1 INNER JOIN テーブル名 2
        ON テーブル名 1.列名 1 = テーブル名 2.列名 2_1)
    INNER JOIN テーブル名 3
    ON テーブル名 2.列名 2_2 = テーブル名 3.列名 3 WHERE 句など;
```

()で囲まれた命令を実行し、その結果を 1 つのテーブルとみなして JOIN 句を使用します。()を使い JOIN 句を連ねていくことで、3 つのテーブルはもちろん、それ以上のテーブルを結合することができます。

・ 書籍情報テーブル(books)、著者－書籍情報テーブル(author_books)、著者情報テーブル(author)を結合して、出版社が「海原出版」である書籍情報を刊行日の新しい順に取り出してみます。取得列は書名、著者名、刊行日です。

```
> SELECT
>   b.title,
>   a.name,
>   b.publish_date
> FROM
>   (   books AS b
>    INNER JOIN
>       author_books AS ab
>    ON
>       b.isbn = ab.isbn)
> INNER JOIN
>   author AS a
> ON
>   ab.author_id = a.author_id
> WHERE
>   b.publish = '海原出版'
> ORDER BY
>   b.publish_date DESC
> ;
```

PRACTICE JOIN 句の入れ子

Q1 社員テーブル(employee)と所属部署テーブル(depart)、タイムカードテーブル(time_card)を結合し、社員コード'DA00001'における 2010 年 12 月分の勤務時間を日付について昇順に出力してみましょう。出力列は depart_name、l_name と f_name の結合列、また取得できた行数も記述してください。

取得行数＿＿＿＿＿

Q2 注文書(order_main)と注文明細テーブル(order_desc)、ユーザテーブル(usr)、商品テーブル(product)を結合し、未納の注文について、発注日、注文番号、商品コード昇順に注文情報を出力してみましょう。また取得できた行数も記述してください。

取得行数＿＿＿＿＿

5　データの登録／更新／削除

データベースは既存データの検索だけでなく、新規のデータを挿入したり、変更されたデータを更新したり、不要になったデータを削除したりすることができます。

5-1　新規のデータを挿入する

既存のテーブルに対してデータを挿入するには INSERT 命令を使用します。

構文〉 INSERT 命令

```
INSERT INTO テーブル名 VALUES (値, ... );
```

テーブル名	データを挿入するテーブル名を記述
値	列の定義順にカンマ区切りで記述 文字列／日付型データはシングルクォート(')で括る

・ ユーザテーブル(usr)に新規ユーザ情報を挿入してみます。追加する内容は以下の通りです。

利用者コード	A201007
利用者名(氏)	鈴木
利用者名(名)	徳次郎
利用者名(氏カナ)	スズキ
利用者名(名カナ)	トクジロウ
住所(都道府県名)	群馬県
住所(市町村名)	群馬市北町
住所(その他)	1-1-1
電話番号	040-999-9999
E_Mail アドレス	NULL

```
> INSERT INTO
>   usr
> VALUES(
>   'A201007',
>   '鈴木',
>   '徳次郎',
>   'スズキ',
>   'トクジロウ',
>   '群馬県',
>   '群馬市北町',
>   '1-1-1',
>   '040-999-9999',
>   NULL)
> ;
```

PRACTICE INSERT 命令

Q1 書籍情報テーブル(books)に対して、以下の新規データを挿入しましょう。

項目	値
ISBN コード	4-8833-0000-4
書名	SQL ミニ
価格	1000
出版社	海原出版
刊行日	2010-03-22
分類 ID	S2222

Q2 著者情報テーブル(author)に対して、以下の新規データを挿入しましょう。

項目	値
著者 ID	I0001
著者名	伊田研二
著者名(カナ)	イダケンジ
生年月日	NULL

Q3　所属部署テーブル(depart)に対して、以下の新規データを挿入しましょう。

項目	値
所属部署コード	E03
所属部署名	第三営業部

Q4　商品テーブル(product) に対して、以下の新規データを挿入しましょう。

項目	値
商品コード	SB00000001
商品名	黒スタンプ
単価	1250

5-2　列指定で新規のデータを挿入する

前項の INSERT 命令では全ての列に対して値を指定しました。もし値が不定な列があるデータの場合は NULL 値を指定する必要があります。しかし列数が多いテーブルで全ての不定値に NULL 値を指定するのは冗長的です。また間違いのもとになります。

INSERT 命令では特定列を指定してデータを挿入することができます。

構文〉 INSERT 命令(2)

```
INSERT INTO テーブル名 (列名, ... ) VALUES (値, ... );
```

列名	確定値を入れる列名
値	列名に対応する確定値

指定されなかった列には自動的に NULL 値又は列にあらかじめ定義されたデフォルト値がセットされます。省略された列が NULL を許さず、デフォルト値も設定されていない場合は INSERT 命令は失敗します。

・　ユーザテーブル(usr)に新規ユーザ情報を挿入してみます。追加する内容は以下の通りです。

利用者コード	B201006
利用者名(氏)	神田
利用者名(名)	愛
利用者名(氏カナ)	カンダ
利用者名(名カナ)	アイ
電話番号	040-888-8888

```
> INSERT INTO
>   usr(
>     user_id,
>     l_name,
>     f_name,
>     l_name_kana,
>     f_name_kana,
>     tel )
> VALUES(
>   'B201006',
>   '神田',
>   '愛',
>   'カンダ',
>   'アイ',
>   '040-888-8888')
> ;
```

複数データを同時に挿入することもできます。

```
> INSERT INTO
>   usr(
>     user_id,
>     l_name,
>     f_name )
> VALUES(
>   'D201001',
>   '山田',
>   'リン' )
>   ,
>   ('D201002',
>   '鈴木',
>   '朝子' )
> ;
```

PRACTICE 列指定の INSERT 命令

Q1 アンケートテーブル(quest)に対して、以下の新規データを挿入しましょう。

項目	値
回答者名	佐々木三郎
回答者名(カナ)	ササキサブロウ
性別	男
都道府県	東京都
本書の評価	3
本書の感想	興味深い内容です。
回答日時	(今日の日付)

Q2 貸し出し記録テーブル(rental)に対して、以下の新規データを挿入しましょう。

項目	値
利用者コード	B201002
ISBN コード	4-0010-0000-0
貸出日	(今日の日付)

Q3　著者情報テーブル(author)に対して、以下の新規データを挿入しましょう。

項目	値
著者 ID	U0001
著者名	内田幸子
著者名(カナ)	ウチダサチコ
生年月日	1973-04-10

Q4　社員テーブル(employee) に対して、以下の新規データを挿入しましょう。

項目	値
社員コード	WA00001
社員名(氏)	和田
所属部署コード	J01
最終更新日	(今日の日付)

5-3　既存データを更新する

テーブル上に既に存在するデータを適切な値に変更することをデータの更新と言います。データの更新を行うにはUPDATE命令を使用します。

構文〉 UPDATE命令

```
UPDATE テーブル名 SET 列名 = 値, ... ;
```

列を複数指定する場合には、「列名 = 値」を必要な数だけカンマ区切りで羅列します。また、値には元の列の値を演算するような式を記述することも可能です。

・ 書籍情報テーブル(books)に登録されている全ての書籍に対して、消費税5%を加味した価格に更新してみます。

```
> UPDATE
>   books
> SET
>   price = price * 1.05
> ;
```

PRACTICE UPDATE 命令

Q1 書籍情報テーブル(books)を以下の要領で一括更新しましょう。

・ISBN コードの先頭に、固定値で「ISBN」を追加

・消費税込みの価格を税抜き価格に変更

Q2 ユーザテーブル(usr)のメールアドレスを全て NULL 値でクリアしましょう。

Q3 商品テーブル(product)に登録されている全商品を 10%値上げするため、更新しましょう。

Q4 Q1 で更新した書籍情報テーブル(books)の ISBN コードの先頭に付加した固定文字列「ISBN」を一律取り除きましょう。

5-4　特定の条件に合致するデータを更新する

前項ではテーブル上に既に存在する全てのデータを更新しましたが、一般的にはある特定のデータに対して更新します。ある特定のデータを更新するには WHERE 句を使用します。

構文〉 WHERE 句を使用した UPDATE 命令

```
UPDATE テーブル名 SET 列名 = 値, ... WHERE 条件式;
```

SELECT 命令の時と同じように、WHERE 国は論理演算子を含めた複合的な条件式を記述することも可能です。

・ アンケートテーブル(quest)から回答コードが「3」のデータについて、本書の評価を 3、本書の感想をブランク、回答日時は現在時刻に更新してみます。

```
> UPDATE
>   quest
> SET
>   answer1 = 3,
>   answer2 = '',
>   answered = now()
> WHERE
>   id = 3
> ;
```

PRACTICE WHERE 句を使用した UPDATE 命令

Q1 書籍情報テーブル(books)で、出版社が「海原出版」を「TAILE 出版」にしてみましょう。

Q2 社員テーブル(employee)で、名前が「山田 奈美」さんの役職を「主任」に、最終更新日を今日の日付にしてみましょう。

Q3 貸し出しテーブル(rental)で、2010 年 3 月 31 日より前でかつ、現在貸し出し中であるレコードについて renturned 列を 9(紛失)で更新しましょう。

5-4　既存データを削除する

検索、追加、更新とデータ操作を確認してきましたが、削除も確認しておく必要があります。データを削除するにはDELETE命令を使用します。

構文〉 DELETE命令

```
DELETE FROM テーブル名 WHERE 条件式;
```

WHERE句で条件式を指定することで、条件に合うデータだけが削除されます。WHERE句を省略できますが、その場合、該当するテーブル内全てのデータが削除されます。

・　月間売り上げテーブル(sales)から2010年11月以前のデータを削除してみます。

```
> DELETE FROM
>   sales
> WHERE
>   s_date <= '2010-11'
> ;
```

PRACTICE DELETE 命令

Q1 貸し出し記録テーブル(rental)から貸し出し日が 2010 年 12 月 31 日以前で、かつ、返却済みである情報について削除してみましょう。

Q2 アクセス記録テーブル(access_log)からアクセス日時が 2010 年 06 月 01 日より前か、リンク元の URL が空である情報を削除しましょう。

Q3 社員テーブル(employee)から退職済みで、最終更新日が 2004-03-31 以前の情報を削除しましょう。

6　データベース構造の操作

これまでは既にあるテーブルを使い、検索、追加、更新、削除を行ってきましたが、ここではテーブルを新規作成したり、削除する機能について確認していきます。

6-1　テーブルを作成する

リレーショナルデータベースにおいて、データを管理する基本的な単位は「テーブル」です。効率的なデータの分析、整理にはよく練りこまれたテーブルを作成することが重要です。

テーブルを作成するには CREATE 命令を使用します。

構文〉 CREATE 命令

```
CREATE テーブル名 (列名 データ型 列フラグ, ..., オプション);
```

データ型	列のデータ型を記述　データ型は P2,3 を参照
列フラグ	各列の特性を表すための属性情報　複数のフラグを指定することができる
オプション	列のキーやインデックスなど制約条件を指定

列フラグで指定する PRIMARY KEY(主キー)はテーブル内に複数指定することはできません。ただし、複数の列からなる複合主キーは可能です。その際は列フラグではなく、オプションとして指定します。

主キーはテーブル内のデータを一意に特定するために用いられるもので、必ずしも必須ではありませんが、リレーショナルデータベースの世界では原則として主キーを設定することをお勧めします。

列フラグ	概要
AUTO_INCREMENT	自動連番(データ型は整数型)
DEFAULT 'デフォルト値'	値が未指定の場合のデフォルト値
[NOT] NULL	NULL を許容する[しない]
PRIMARY KEY	重複する値を許さない(NULL 値は不可)
UNIQUE	重複する値を許さない(NULL 値を許可)
REFERENCES テーブル名 (列名,･･･)	外部参照制約

オプションはテーブルに含まれる列のキーやインデックス等の制約条件を指定するものです。列定義の後にカンマ区切りで記述します。

オプション	内容
PRIMARY KEY(列名,･･･)	主キーを作成
INDEX インデックス名(列名,･･･)	インデックスを作成(NULL 値は不可)
UNIQUE インデックス名(列名,･･･)	重複を許さない(NULL 値は不可)
FOREIGN KEY(列名,･･･) REFERENCES テーブル名(列名,･･･)	外部参照制約

・ 書籍分類テーブル(category2)を作成してみます。

書籍情報テーブル(category2)

列名	データ型	概要
category_id	CHAR(5)	分類 ID<主キー>
category_name	VARCHAR(50)	分類名

```
> CREATE TABLE
>   category2
>   (
>     category_id CHAR(5) PRIMARY KEY,
>     category_name VARCHAR(50)
>   )
> ;
```

・ 書籍情報テーブル(books2)を作成してみます。CATEGORY2 を参照できるようにリレーションします。

書籍情報テーブル(books2)

列名	データ型	概要
isbn	CHAR(17)	ISBN コード<主キー>
title	VARCHAR(255)	書籍名
price	INT	価格
publish	VARCHAR(30)	出版社
publish_date	DATE	刊行日
category_id	CHAR(5)	分類 ID

```
> CREATE TABLE
>   books2
>   (
>     isbn CHAR(17) NOT NULL,
>     title VARCHAR(255),
>     price INT,
>     publish VARCHAR(30),
>     publish_date DATE,
>     category_id CHAR(5),
>     PRIMARY KEY (isbn),
>     FOREIGN KEY(category_id) REFERENCES CATEGORY(category_id)
>   )
> ;
```

PRACTICE CREATE 命令

Q1 著者テーブル(author2)を作成しましょう。

列名	データ型	概要
author_id	CHAR(5)	著者 ID<主キー>
name	VARCHAR(30)	著者名
name_kana	VARCHAR(100)	著者名カナ
birth	DATE	生年月日

Q2 著者-書籍情報テーブル(author_books2)を新規作成しましょう。テーブル books2、author2 を参照できるよう、リレーションしましょう。

列名	データ型	概要
isbn	CHAR(13)	ISBN コード<主キー>
author_id	CHAR(5)	著者 ID<主キー>

6-2　既存テーブルに列を追加する

既に運用しているテーブルに、新たに列を追加する必要が出てきたり、主キーなどの制約条件を追加する必要が出てくることがあります。ALTER TABLE 命令を使うと、列や制約条件を後から追加することができます。

構文〉 ALTER TABLE 命令

```
ALTER TABLE テーブル名 ADD 追加内容;
```

追加内容では AFTER 句を使い、列をテーブル内のどこに追加するかを指定できます。FIRST 句を使うと先頭に新規列が追加されます。

・ アンケートテーブル(quest)の末尾に、新規列「last_update」を DATATIME 型で追加してみます。

```
> ALTER TABLE
>   quest
> ADD
>   last_update DATETIME
> AFTER
>   answered
> ;
```

PRACTICE ALTER TABLE 命令

Q1 書籍情報テーブル(books2)の category_id 列の後方に sales 列(INT 型)を追加してみましょう。

Q2 ユーザテーブル(author2)の birth 列の後方に以下の２列追加してみましょう。

・sex 列(VARCHAR(5)型、デフォルト値は男)

・regist_date 列(DATE 型)

6-3　既存テーブルを破棄する

既存のテーブルを破棄するには DROP TABLE 命令を使用します。ただし、データやテーブルごと削除するような命令は、そのデータやテーブルが本当に不要であるのか十分に検討してから行う必要があります。いったん削除したテーブルは元に戻すことができません。

構文〉 DROP TABLE 命令

```
DROP TABLE テーブル名;
```

・　著者情報テーブル(author2)破棄してみます。

```
> DROP TABLE
>   author2
> ;
```