SQL 基礎

1　データベースの基本

1-1　データベースとは

基本となる単語の意味から整理しましょう。

データとは

物事や事象を表したり、構成したりする情報です。文字や数字だけでなく、画像や動画で表されることもあります。

データベースとは

何かの目的や基準で集められたデータの集合体です。データベースであるためには、「集められる目的や基準」が必要です。

ＤＢＭＳとは

ＤＢＭＳ（データベースマネジメントシステム）は、データベースを管理するシステムです。
データを荷物、データベースを倉庫とすれば、ＤＢＭＳは倉庫の管理人に当たります。

具体的なデータベースを考えてゆきます。
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。

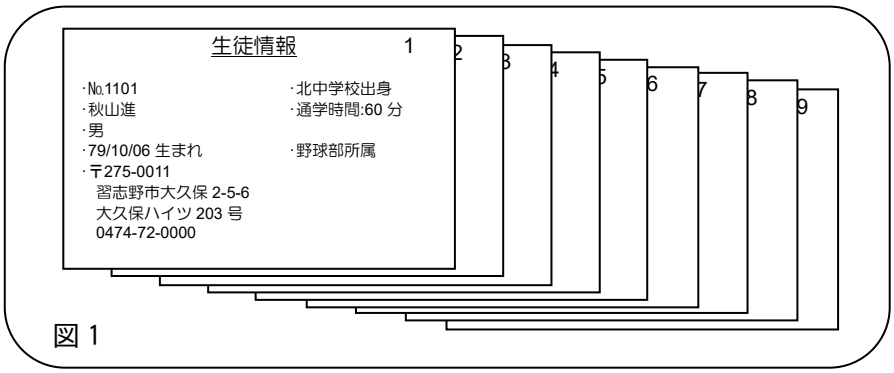


図 1

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
1102	岩瀬 一郎	男	79/12/25	西中	ｻｯｶｰ部	272-0115	市
1103	遠藤 喜一	男	80/03/10	東中	なし	274-0077	船
1104	大西 守	男	79/05/25	第二中	ｻｯｶｰ部	274-0824	船
1105	北川 文彦	男	79/08/15	大山中	野球部	263-0033	千
1106	笹井 秀次	男	79/09/24	北中	ﾃﾆｽ部	289-0346	香
1107	中野 宏志	男	80/02/15	千葉中	ｻｯｶｰ部	283-0811	東
1108	西川 正則	男	79/12/10	南中	美術部	270-0112	流

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マネジメントソフト（ＲＤＢＭＳ）**が行う領域です。

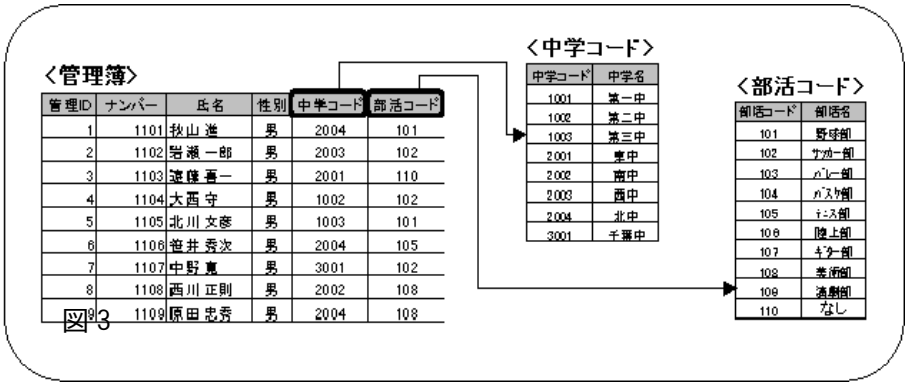


図 3

ＤＢＭＳを利用するのはなぜでしょうか？以下のような理由が考えられます。

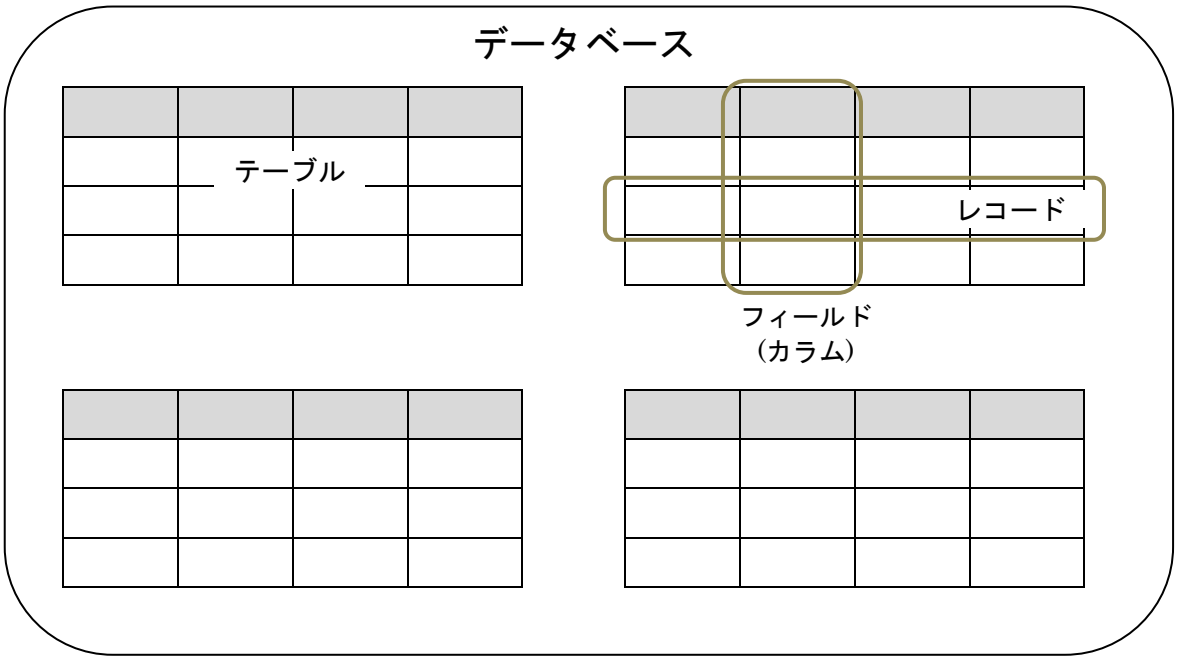
- データ操作の効率化（検索、追加、削除、変更）
- データの無矛盾
- データの維持、保全（セキュリティ、異常時の復旧）
- データ容量の効率化

リレーショナルデータベースマネージメント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。

今回利用するＲＤＢＭＳである MySQL の主なデータ型は以下の通りです。

【整数型】

型名	型に格納できるデータ
TINYINT	1 バイトの整数
SMALLINT	2 バイトの整数
MEDIUMINT	3 バイトの整数
INTEGER	4 バイトの整数
INT	INTEGER と同じ
BIGINT	8 バイトの整数
NUMERIC(n, d)	パック 10 進数（省略値：n=10,d=0）
DECIMAL	NUMERIC に同じ
DEC	NUMERIC に同じ

【浮動小数点型】

型名	型に格納できるデータ
FLOAT(x)	浮動小数点数、x は４か８バイト
DOUBLE(n, d)	浮動小数点数、精度はパラメータによる
DOUBLE PRECISION(n,d)	倍精度の浮動小数点数
REAL	DOUBLE PRECISION と同じ

【文字列型】

型名	型に格納できるデータ
CHAR(n)	固定長文字列、長さは、n による
VARCHAR(n)	可変長文字列、最大長は、n による
TINYTEXT	255 バイトまでの可変長文字列
TEXT	65,535 バイトまでの可変長文字列
MEDIUMTEXT	16,777,215 バイトまでの可変長文字列
LONGTEXT	4,294,967,295 バイトまでの可変長文字列

【日付型】

型名	型に格納できるデータ
DATETIME	日付と時刻
DATE	日付
TIME	時刻
TIMESTAMP(n)	日付と時刻。n は表示上の桁数
YEAR(n)	年。N は２または４

【バイナリー型】

型名	型に格納できるデータ
TINYBLOB	255 バイトまでのバイナリーデータ
BLOB	65,535 バイトまでのバイナリーデータ
MEDIUMBLOB	16,777,215 バイトまでのバイナリーデータ
LONGBLOB	4,294,,967,295 バイトまでのバイナリーデータ

【列挙型】

型名	型に格納できるデータ
ENUM('Value1' (,…))	括弧内に指定された任意の文字列
SET('Value1' (,…))	括弧内に指定された任意の文字列の組み合わせ

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。

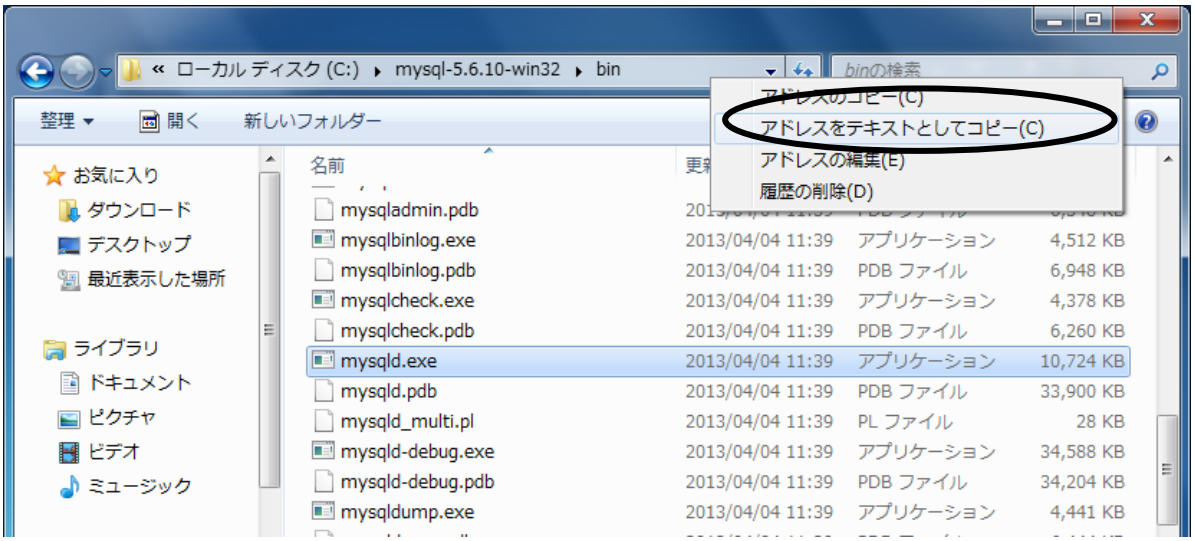
2　MySQL の準備

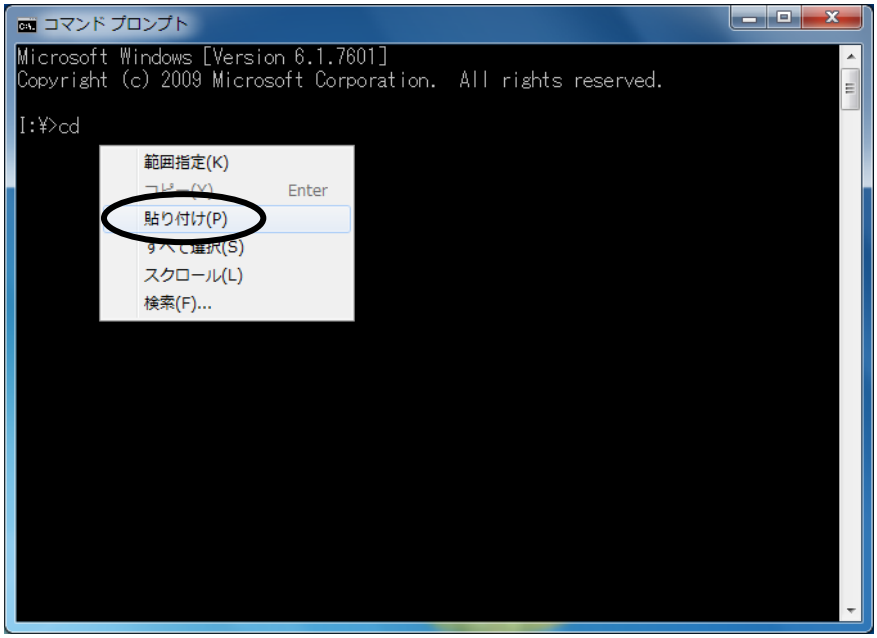
2-1　準備

(1)　MySQL の初期設定

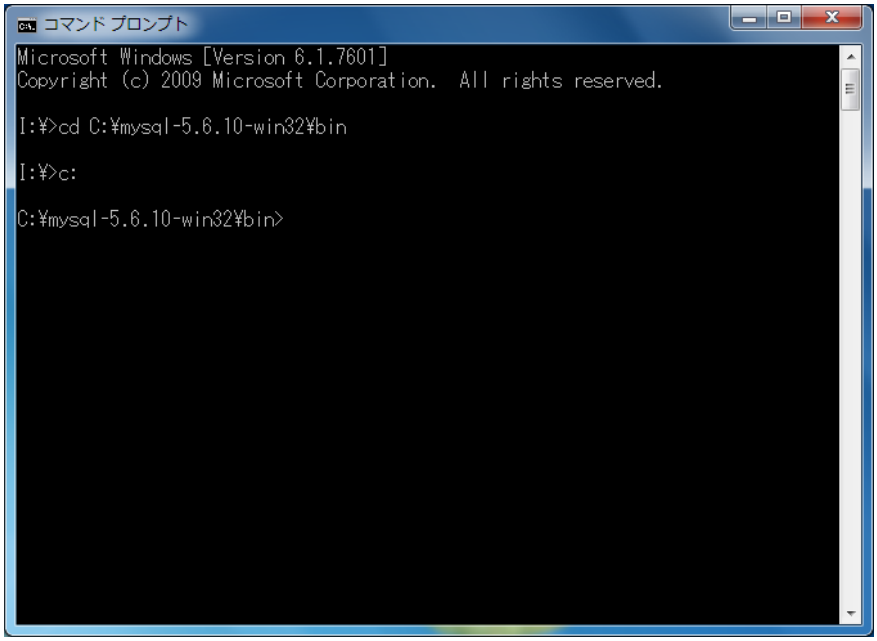
①　コマンドプロンプトを新しく開き、「cd」と入力する

②　MySQL フォルダ下の bin フォルダウィンドウで、アドレスバーからパス名をコピーして①のコマンドプロンプトウィンドウに貼り付けて Enter キーを押す





③　「c:」と入力して Enter キーを押し、ディレクトリを移動する



※　①から②の作業で、MySQL のコマンドが使用できるディレクトリに移動できた

(2)　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 dbwork データベースに展開します。dbwork は今回使用するデータベース名です。

①　「dbwork.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 dbwork
password: *****

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

パスワード(root1)を入力

パスワード(root1)を入力

dbwork という名のデータベースを生成、dbwork.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 dbwork
password: *****
```

パスワード(root1)を入力

使用したデータベースをバックアップを取りたい場合は、以下のコマンドを入力してデータベースをファイルとして保存する

ハイフン(-)が２つ

```
mysqldump -u root -p --databases dbwork > dbwork.sql
password: *****
```

パスワード(root1)を入力

コマンドが起動しているディレクトリに dbwork.sql ファイルが作成される

(3) 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 を起動する

```
mysql -u root -p
password: *****
```

パスワード(root1)を入力

② データベースを指定する

```
USE dbwork;
Database changed
```

dbwork データベースに移動する　以降の SQL 命令はすべて dbwork 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

```
SHOW TABLES;
+-----------------+
| Tables_in_dbwork |
+-----------------+
| author          |
| author_books    |
  ... 中略 ...
| shop            |
| user            |
| worktime        |
+-----------------+
16 rows in set (0.01 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

```
SHOW FIELDS FROM books;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| isbn        | char(13)    | NO   | PRI |         |       |
| title       | varchar(255) | YES  |     | NULL    |       |
| price       | int         | YES  |     | NULL    |       |
| publish     | varchar(30) | YES  |     | NULL    |       |
| publish_date | date        | YES  |     | NULL    |       |
| category_id | char(2)     | YES  |     | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
6 rows in set (0.01 sec)
```

④ MySQL Monitor を終了する

```
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

・　商品テーブル(product)から全てのデータを取得し、表示してみます。

```
SELECT
  *
FROM
  product
;
```

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 ユーザーテーブル(user)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

・ ユーザーテーブル(user)から利用者名と住所(都道府県名)、E-mail アドレスだけを取得し、表示してみます。

```
SELECT
  name,
  prefecture,
  email
FROM
  user
;
```

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。

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 ユーザーテーブル(user)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 社員テーブル(employee)から役職を取り出すための 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 ユーザテーブル(user)からユーザ氏名を、同じ氏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	gender = '男'
<>	等しくない	gender <> '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

・ 社員テーブル(employee)から性別が女(2)で登録されている情報だけを取得します。取り出す列はname、birth、position列とします。

```
SELECT
  name,
  birth,
  position
FROM
  employee
WHERE
  gender = 2
;
```

PRACTICE WHERE 句

Q1 社員テーブル(employee)から最終更新日が「2020-04-01」以降の社員情報を取り出してみましょう。取り出すのは name、position 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
  employee
WHERE
  [    ②    ]
;
```

取得行数＿＿＿＿＿＿

Q2 社員テーブル(employee)から役職が「主任」の情報のみを取り出してみましょう。取り出す列は name、depart_id 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＿＿＿＿＿＿

Q4 書籍情報テーブル(books)から出版社が「海岳社」「WAZEN 社」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　NOT LIKE 演算子を使用してユーザテーブル(user)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 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　BETWEEN 演算子を使用して社員テーブル(employee)から 1999 年生まれの社員を取り出してみましょう。取り出す列は、name、biirth、position 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)から刊行日が 2010 年より前で分類 ID が「LH」(暮らし・健康) 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,
  category_id
FROM
  books
WHERE
    publish_date ＜ '2010-01-01'
  AND
    category_id = 'LH'
;
```

PRACTICE 論理演算子

Q1 書籍情報テーブル(books)から書籍名に「観察」という単語を含み、かつ、価格が 1,000 円未満の情報を取り出してみましょう。取り出す列は isbn、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。

取得行数＿＿＿＿＿＿

Q2 ユーザテーブル(user)から東京都在住で、かつ、E-Mail アドレスのドメインが practice.com のユーザ情報を取り出してみましょう。取り出す列は 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 社員テーブル(employee)から女(2)で役職が課長または部長の情報を取り出してみましょう。取り出す列は name、birth、depart_id 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

・ 書籍情報テーブル(books)から出版社が「WAZEN 社」のデータを抽出してみます。その際、刊行日の新しい順に結果を並べ替えます。取り出す列は title、isbn、publish_date 列とします。

```
SELECT
  title,
  isbn,
  publish_date
FROM
  books
WHERE
  publish='WAZEN 社'
ORDER BY
  publish_date DESC
;
```

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(user)から千葉県、神奈川県に住んでいる人のリストを住所（都道府県名)、利用者名(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 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

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、各社ごとの刊行冊数を求めてみます。取り出す列は publish、publish 列の個数とします。

```
SELECT
  publish,
  COUNT(publish)
FROM
  books
GROUP BY
  publish
;
```

集計関数以外にも文字列関数、日付関数、変換関数、算術関数が用意されています。代表的なものを以下に示します。詳細はマニュアル等を参照ください。

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

分類	関数	概要
日付関数	NOW()	現在の日付と時刻を取得
	CURRENT_DATE()	現在の日付を取得
	CURRENT_TIME()	現在の時刻を取得
	CURRENT_TIMESTAMP()	現在の日時を取得
	DATE_ADD(d,INTERVAL n p)	任意の単位での日付の足し算（注 1） d:日付式、n:数値式、p:日付要素
	DATE_FORMAT(d,f)	日付データを整形して文字列に変換（注 1） d:日付式、f:書式指定文字列
	DATEDIFF(p, d, e)	日付の差を計算する。d<e で正の値を返す p:日付要素、d,e:日付式
	DAY(d)、MONTH(d)、YEAR(d)	日付(d)の日、月、年を取得
	HOUR(d)、MINITS(d)、SECOND(d)	日付(d)の時、分、秒を取得
	LAST_DAY(d)	ある日付(d)の月の最終日を取得
	WEEKDAY(d)	曜日を取得。０が日曜日

注１：MySQL、MariaDB のみの機能です

分類	関数	概要
算術関数	ABS(n)	n の絶対値を取得
	ACOS(n)、ASIN(n)、ATAN(n)	n に対する逆コサイン、逆サイン、逆タンジェント
	COS(n)、SIN(n)、TAN(n)	n に対するコサイン、サイン、タンジェント
	CEIL(n)	n に対しそれ以上で最も小さい整数値を取得
	FLOOR (n)	n に対しそれ以下で最も大きい整数値を取得
	POWER(n,m)	ｎのｍ乗を計算し取得
	GREATEST(a1, a2, ･･･)	最も大きい値を取得
	LEAST(a1, a2, ･･･)	最も小さい値を取得
	MOD(n,m)	n/m の剰余を取得
	SQRT(n)	n の平方根を取得
	RAND()	乱数の発生
	PI()	円周率を取得

PRACTICE GROUP BY 句

Q1 社員テーブル(employee)から所属部署ごとの所属人数を求めましょう。

取得行数＿＿＿＿＿＿

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 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＿＿＿＿＿＿

3-8　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;
```

・ 書籍情報テーブル(books)から出版社ごとの価格で、平均が 2,000 円未満のデータだけを取り出してみます。

```
SELECT
  publish,
  AVG(price)
FROM
  books
GROUP BY
  publish
HAVING
  AVG(price) < 2000
;
```

PRACTICE HAVING 句

Q1 SUBSTRING 関数(P28 参照)を使用して商品テーブル(product)から商品種別(商品コード上 2 桁)ごとの価格で、平均が 450 円未満のデータ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＿＿＿＿＿

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、5 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＿＿＿＿＿

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.name,
  depart.depart_name,
  employee.position
FROM
  employee
INNER JOIN
  depart
ON
  employee.depart_id = depart.depart_id
WHERE
  employee.retired <> 1
ORDER BY
  employee.depart_id ASC,
  employee.e_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.name,
  d.depart_name,
  e.position
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.e_id ASC
;
```

PRACTICE INNER JOIN 句

Q1 社員テーブル(employee)と勤務時間テーブル(worktime)から氏名と 2020 年 10 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 2021 年 12 月の売り上げを店舗名、と売上高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 2022 年の売上高を集計し、店舗名、と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(user)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集計し、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、貸し出し数とします。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(detail)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(d.quantity),
  SUM(p.price * d.quantity)
FROM
  detail AS d
RIGHT JOIN
  product AS p
ON
  p.p_id = d.p_id
GROUP BY
  p.p_id,
  p.p_name
ORDER BY
  SUM(p.price * d.quantity) DESC
;
```

PRACTICE RIGHT JOIN 句、LEFT JOIN 句

Q1 ユーザテーブル(user)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＿＿＿＿＿＿

Q2 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＿＿＿＿＿＿

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、結合する順に左から JOIN 句を指定していきます。

構文〉 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;
```

順番に JOIN 句を実行し、その結果を 1 つのテーブルとみなして次の JOIN 句を使用します。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「房総出版」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。

＊ SQL 文は次ページ

```
SELECT
  b.title,
  a.name,
  b.publish,
  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 = '房総出版'
ORDER BY
  b.publish_date DESC
;
```

PRACTICE 複数の JOIN 句

Q1 社員テーブル(employee)と所属部署テーブル(depart)、勤務時間テーブル(worktime)を結合し、役職が担当で 2020 年 10 月分の平均勤務時間を、平均勤務時間について昇順に出力してみましょう。出力列は depart_name、name、平均勤務時間、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(detail)、ユーザテーブル(user)、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。出力列の別名は発注日、注文番号、利用者氏名、商品名、商品単価、購入数とし、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(user)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	1200017
利用者名	鈴木　徳次郎
利用者名(カナ)	スズキ　トクジロウ
住所(都道府県名)	千葉県
住所(その他)	千葉市北町 1-1-1
電話番号	043-999-9999
E_Mail アドレス	t.suzu99@practice.com

```
INSERT INTO
  user
VALUES(
  '1200017',
  '鈴木　徳次郎',
  'スズキ　トクジロウ',
  '千葉県',
  '千葉市北町 1-1-1',
  '043-999-9999',
  't.suzu99@practice.com')
;
```

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	4-0010-0013-0
書籍名	発酵食品いろいろ
価格	1870
出版社	WAZEN 社
刊行日	2022-12-10
分類 ID	LH

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	A2004
著者名	伊田　研二
著者名(カナ)	イダ　ケンジ
生年月日	1992-02-18

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(user)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	1300027
利用者名	神田　愛
都道府県名	東京都

```
INSERT INTO
  user(
    user_id,
    name,
    prefecture )
VALUES(
  '1300027',
  '神田　愛',
  '東京都')
;
```

複数データを同時に挿入することもできます。

```
INSERT INTO
  user(
    user_id,
    name,
    prefecture )
VALUES
  ('0800013',
   '上藤　倫央',
   '茨城県')
 ,
  ('0900010',
   '横崎　奈智',
   '栃木県')
;
```

PRACTICE 列指定の INSERT 命令

Q1 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	M500004
社員名	藻田　早紀
社員名(カナ)	モダ　サキ
生年月日	2000-05-18
最終更新日	(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	0800011
ISBN コード	4-8833-0024-0
貸出日	(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、消費税を加味した価格に更新してみます。

```
UPDATE
  books
SET
  price = price * 1.1
;
```

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。

・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(user)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。

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 句に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。

・　書籍情報テーブル(books)について、出版社が WAZEN 社で刊行日が 2020-01-01 以降の書籍価格を 100 円値上げしてみます。

```
UPDATE
  books
SET
  price = price + 100
WHERE
  publish = 'WAZEN 社'
 AND
  publish_date >= '2020-01-01'
;
```

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「WAZEN 社」を「和然社」にしてみましょう。

Q2　社員テーブル(employee)で、名前が「山崎　誠」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)で貸出日が 2018 年 3 月 26 日で利用者コードが「1200016」のレコードについて returned 列を 9(紛失)で更新しましょう。

5-5　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 2017 年 6 月以前のデータを削除してみます。

```
DELETE FROM
  sales
WHERE
  s_date <= '2017-06'
;
```

PRACTICE DELETE 命令

Q1 貸し出し記録テーブル(rental)から貸し出し日が 2019 年 12 月 31 日以前で、かつ、東京都の利用者(利用者コード先頭 2 桁：13)について削除してみましょう。

Q2 社員テーブル(employee)から退職済みで、最終更新日が 2020-03-31 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

・　資格情報テーブル(qual)を作成してみます。

資格情報テーブル(qual)

列名	データ型	概要
q_id	CHAR(5)	資格 ID<主キー>
q_name	VARCHAR(50)	資格名
q_class	VARCHAR(50)	級

```
CREATE TABLE
  qual
  (
    q_id CHAR(5) PRIMARY KEY,
    q_name VARCHAR(50),
    q_class VARCHAR(50)
  )
;
```

・　取得資格テーブル(cert)を作成してみます。

取得資格テーブル(cert)

列名	データ型	概要
e_id	CHAR(7)	社員コード
q_id	CHAR(5)	資格 ID
cert_date	DATE	認定日

```
CREATE TABLE
  cert
  (
    e_id CHAR(7) NOT NULL,
    q_id CHAR(5) NOT NULL,
    cert_date DATE
  )
;
```

PRACTICE CREATE 命令

Q1　出版社テーブル(publish)を作成しましょう。

列名	データ型	概要
publish_id	CHAR(5)	出版社コード<主キー>
pub_name	VARCHAR(30)	出版社名
post_no	CHAR(7)	郵便番号(ハイフンなし)
address	VARCHAR(100)	所在地
phone	VARCHAR(15)	電話番号(ハイフンなし)

Q2　店舗担当者テーブル(rep)を新規作成しましょう。

列名	データ型	概要
s_id	CHAR(5)	店舗コード
e_id	CHAR(7)	社員コード

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 ADD 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。

・ 取得資格テーブル(cert)の末尾に、登録日を格納する新規列「reg_date」を DATE 型で追加してみます。

```
ALTER TABLE
  cert
ADD
  reg_date DATE
AFTER
  cert_date
;
```

ALTER TABLE 命令には列の追加以外にもテーブル構造を変更する機能があります。

キーワード	概要
RENAME	テーブル名、カラム名などを変更する
CHANGE	カラム名とカラム定義を変更する
MODIFY	カラム定義を変更する
DROP	カラムを削除する

PRACTICE ALTER TABLE 命令

Q1　書籍情報テーブル(author_books)の author_id 列の後方に flag 列(SMALLINT 型)を追加してみましょう。

Q2　著者情報テーブル(author)の birth 列の後方に以下の２列追加してみましょう。

・gender 列(VARCHAR(5)型、デフォルト値は男)

・reg_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;
```

・　店舗担当者テーブル(rep)を破棄してみます。

```
> DROP TABLE
>   rep
> ;
```