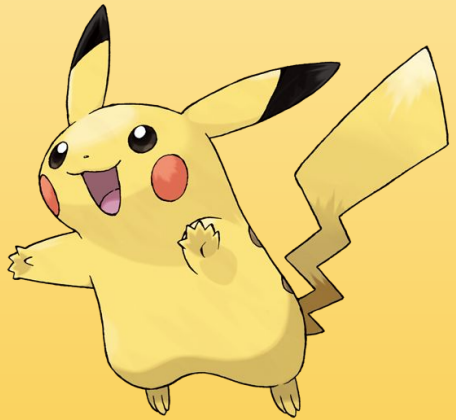
SQL基礎

～ポケモンでSQLを理解しよう～



SQLを発表する理由

・案件先で触れる機会があったため。
・なんとなく触ったことはあるけど深く勉強したことは
ないよという人にSQLを学習する機会を提供するため。

ポケモンをデータとして扱う理由

・私自身ポケモンが好きだから
・データベース、SQLを伝える上でデータも複数あり、
エンタメとの掛け算がしやすいと考えたため。

ポケモンとは？

・1996年に任天堂から発売されてた
ポケモンと呼ばれるキャラクターを育てたり、
戦わせたりするロールプレイングゲーム

・ポケモン（キャラクター）は全151種類
（※第一世代のみ）

・ポケモンには
名前, イラスト, No. , 分類, タイプ, 高さ,
重さ, 特性, ずかんの記述
などさまざまなデータがあります。

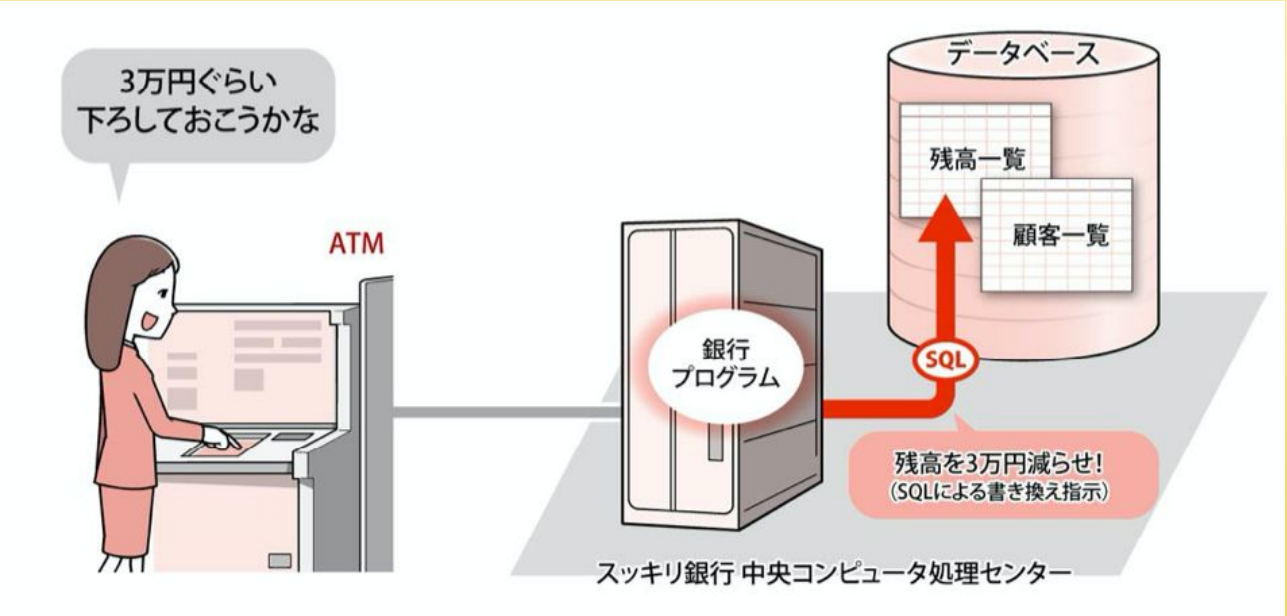
今回はそれら複数のデータをデータベースで管理し、
SQL言語を使って操作していきます。

データベースとは？

- 決まった形式（データモデル）で整理されたデータの集まりのこと
- 例）在庫，顧客情報，アカウント情報，購入情報など
- 一般的には表の形式でデータを管理する（RDB)が使用される

SQLとは？

・データベースを操作するための言語のこと
・データを新たに保存したり、書き換えたり、消したり、取得したりといった操作を行うことができる。

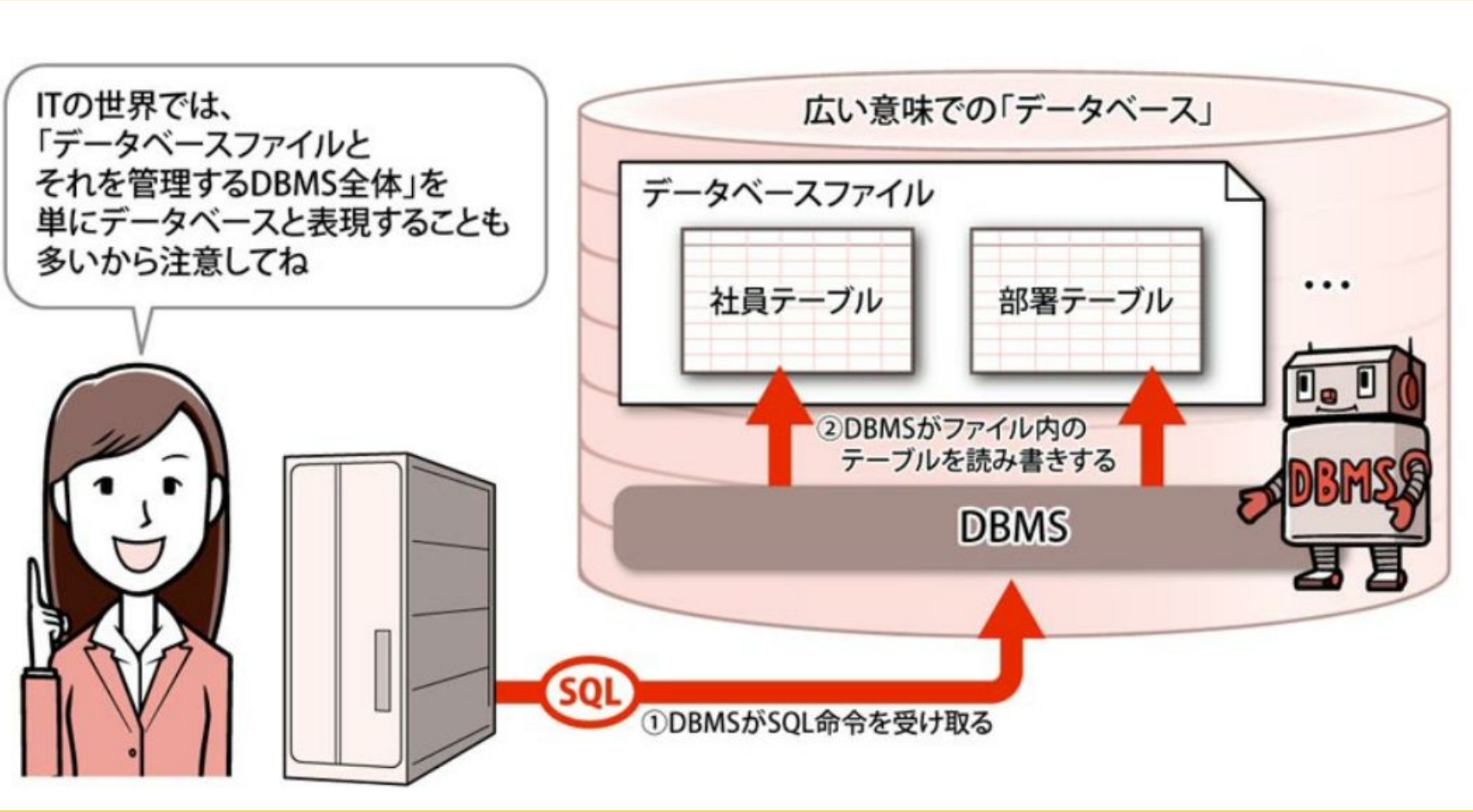


データベース管理システム (DBMS)

SQLファイルはデータベース管理システムと呼ばれるプログラムに送ることでデータベースを操作できる。

データベース管理システムの種類

OracleDatabase, Db2, SQL Server, MySQL, MariaDB, PostgreSQL
SQLite, H2 Database ...


ITの世界では、
「データベースファイルと
それを管理するDBMS全体」を
単にデータベースと表現することも
多いから注意してね
広い意味での「データベース」
データベースファイル
社員テーブル
部署テーブル
・・・
②DBMSがファイル内の
テーブルを読み書きする
DBMS
DBMS
SQL
①DBMSがSQL命令を受け取る

SQL操作を体験する

ポケモンのデータを入れるテーブルを作成しよう

■テーブルの作成を行うSQL文

CREATE TABLE テーブル名(カラム名 データ型)

pokemon_1 ← テーブル名
auto_increment ← データ追加時に現在格納されている最大値に 1 を追加した数値を自動で格納
char, varchar ← 文字列
float ← 実数（小数まで扱える）

【TRY1:pokemon_1テーブル作成】

```
CREATE TABLE pokemon_1(
  id int auto_increment,
  PRIMARY KEY (id),
  number char(3),
  name varchar(50),
  classification varchar(255),
  type_1 varchar(10),
  type_2 varchar(10),
  height NUMERIC(3,1),
  weight NUMERIC(3,1),
  Characteristic varchar(50),
  explanation varchar(255)
);
```

作成したテーブルにデータを入れてみよう

■テーブルにデータを登録するSQL文

INSERT INTO テーブル名 (カラム名1, カラム名2,...) VALUES (値1, 値2,...);

カラム名は順番に ずかんNo., ポケモンの名前, タイプ1, タイプ2, 高さ, 重さ
,特性, ずかんの記述　これらを表す。

【TRY2:ピカチュウのデータをテーブルに入れる】

```
-- ピカチュウ
INSERT INTO pokemon_1 (number, name, classification, type_1, type_2, height, weight,
Characteristic, explanation)
        VALUES ('025', 'ピカチュウ', 'ねずみポケモン', 'でんき', null,  0.4, 6.0, 'せいでんき', 'つくる　でんきが　きょうりょくな　ピカチュウほど　ほっぺの　ふくろは　やわらかく　よく　のびるぞ');
```

ポケモンのデータの入ったテーブルを検索してみよう

■テーブルの中のデータを検索（取得）する

```
SELECT カラム名1, カラム名2, カラム名3...
  FROM テーブル名;
```

【TRY3:全てのポケモンの名前をpokemon_1テーブルから取得する。】

```
SELECT name
   FROM pokemon_1;
```

【TRY4:全てのポケモンの名前, タイプ1, タイプ2, 図鑑説明を pokemon_1テーブルから取得する。】

```
SELECT name, type_1, type_2, explanation
   FROM pokemon_1;
```

【TRY5:全てのポケモンのカラムをpokemon_1テーブルから取得する。】

```
SELECT *
   FROM pokemon_1;
```

または

```
SELECT id,number, name, classification, type_1, type_2, height, weight,
Characteristic, explanation
   FROM pokemon_1;
```

※　*で全てのカラムを取得する

特定のポケモンを検索してみよう

■WHEREで検索する条件を指定する

```
SELECT カラム名1, カラム名2, カラム名3...
  FROM テーブル名
 WHERE カラム名 = 値       （<, >, <=, >=）
       AND カラム名 = 値     （OR）
```

【TRY6:重さ100kgよりも重いポケモンを検索する】

```
SELECT *
FROM pokemon_1
WHERE weight > 100;
```

【TRY7:高さ1m以下のポケモンを検索する】

```
SELECT *
FROM pokemon_1
WHERE height <= 1;
```

【TRY8:高さ1.7mかつ重さ90.5kgのポケモンを検索する】

```
SELECT name
FROM pokemon_1
WHERE height = 1.7
AND weight = 90.5;
```

？？？？

リザードン

ポケットモンスター

POCKET MONSTERS

ポケモンのデータを更新・削除しよう

■UPDATE, SETでデータを更新（修正）する

```
UPDATE テーブル名
      SET カラム名 = 値   --セットする値
 WHERE カラム名 = 値     --レコードの指定（条件式）;
```

※UPDATE, SETだけでWHEREの指定がないと
全てのレコードの値が更新されるので注意。

■DELETE, FROMでデータを更新（修正）する

```
DELETE FROM テーブル名
          WHERE カラム名 = 値     --レコードの指定（条件式）;
```

※DELETE FROM テーブル名;
だけでWHEREの指定がないと全てのレコードが消えるので注意。

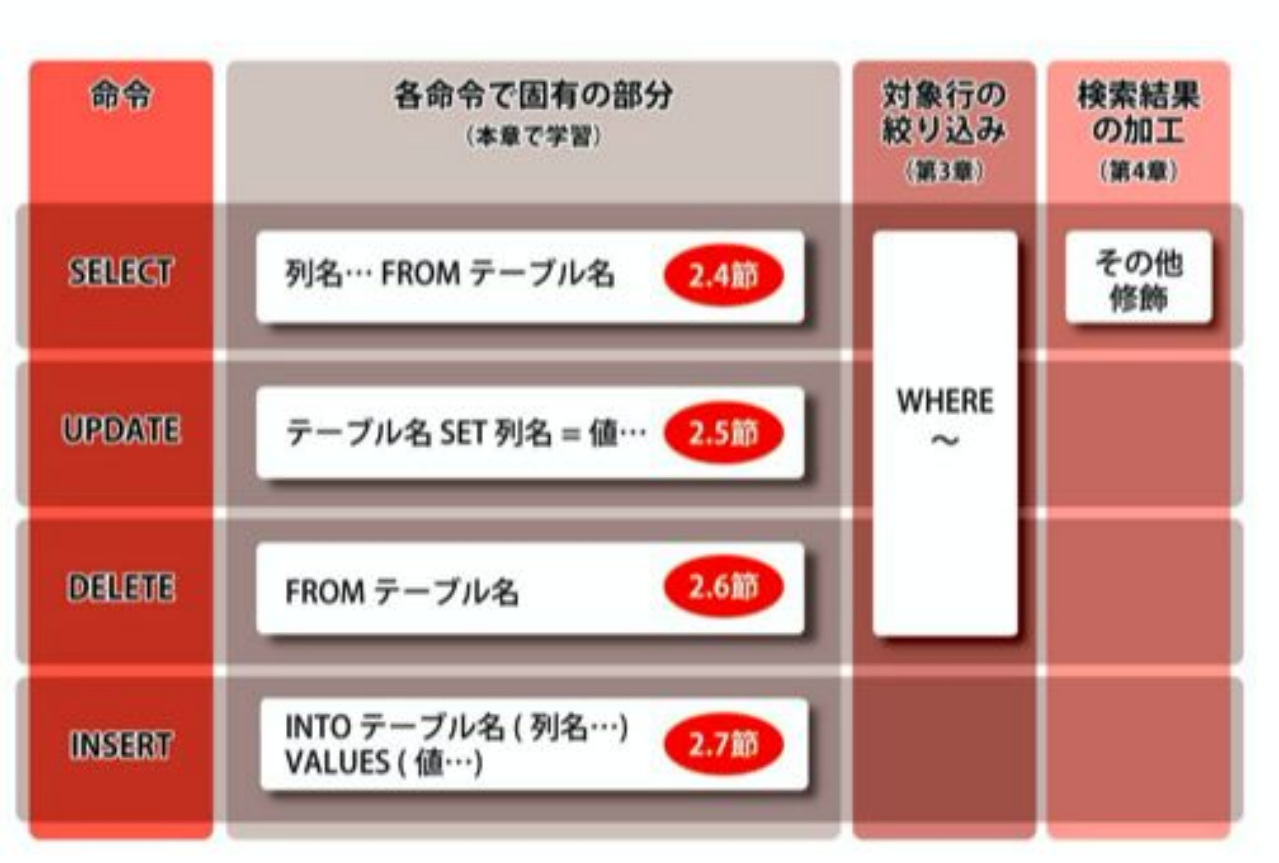
【TRY9:nameカラムのけつばんをホウオウに変更する。】

```
UPDATE pokemon_1
      SET classification = 'にじいろポケモン',
                    name = 'ホウオウ',
                  type_1 = 'ほのお',
                  type_2 = 'ひこう',
                  height = 3.8,
                  weight = 199.0,
          characteristic = 'プレッシャー',
             explanation = 'からだは なないろに かがやきとんだあとに
                           にじが できるとしんわに のこされているポケモン。'
WHERE name = 'けつばん';
```

【TRY10:ホウオウのデータを削除する】

```
DELETE FROM pokemon_1
WHERE name = 'ホウオウ';
```

操作の種類は大きく4種類



すでに出た命令文

```
SELECT *
FROM pokemon_1
WHERE weight > 100;

INSERT INTO pokemon_1 (..., ..., ..., ..., ..., ..., ..., ..., ...,)
VALUES ('025', '...', '...', '...', ...,  0.0, 0.0, '...', '...');

UPDATE pokemon_1
       SET …, …, …
 WHERE name = 'けつばん';

DELETE FROM pokemon_1
WHERE name = 'ホウオウ';
```

ポケモンのデータに別名を定義しよう

■取得した際のnameカラムに別名を定義する。

```
SELECT カラム名 AS 別名
    FROM テーブル名 AS 別名
 WHERE カラム名 = 値
```

・結果表における列のタイトルを任意のものに変更できる。
・わかりにくい、長い列名も、分かりやすく短い別名を付けて利用できる。

【TRY11:カラム名を好きなポケモンに変えて取得する】

```
SELECT name AS 好きなポケモン
FROM pokemon_1 AS 第一世代
WHERE name = 'ピカチュウ'
OR name = 'ロコン';
```

ポケモンのnullデータを判定しよう

■NULLであることを判定する。

```
SELECT カラム名
    FROM テーブル名
 WHERE カラム名 IS NULL;
```

■NULLでないことを判定する。

```
SELECT カラム名
    FROM テーブル名
 WHERE カラム名 IS NOT NULL;
```

※NULLは＝では判定できません。

```
SELECT カラム名
    FROM テーブル名
 WHERE カラム名 = NULL;     ※エラーが発生する。
```

【TRY12：タイプが単体のポケモンの名前とタイプを取得する。】

```
SELECT name, type_1, type_2
    FROM pokemon_1
 WHERE type_2 IS NULL;
```

【TRY13：タイプが複数のポケモンの名前とタイプを取得する。】

```
SELECT name, type_1, type_2
    FROM pokemon_1
 WHERE type_2 IS NOT NULL;
```

※NULLは＝では判定できない例

```
SELECT name, type_1, type_2
    FROM pokemon_1
 WHERE type_2 = NULL;
```

※取得不可

ポケモンの名前があるパターンを含んでいるかを調べよう

■パターンマッチング…文字列があるパターンを含んでいるかをチェックすること

```
SELECT カラム名
    FROM テーブル名
 WHERE カラム名 LIKE '%文字列%';
// 指定カラムのデータに文字列が含むデータを取得

SELECT カラム名
    FROM テーブル名
 WHERE カラム名 LIKE '文字列%';
// 指定カラムのデータに文字列から始まるデータを取得

SELECT カラム名
    FROM テーブル名
 WHERE カラム名 LIKE '%文字列';
// 指定カラムのデータに文字列で終わるデータを取得
```

【TRY14:ポケモンの名前に「サイ」が含むポケモンのデータを取得】

```
SELECT *
    FROM pokemon_1
 WHERE name LIKE '%サイ%';
```

【TRY15:ポケモンの名前が「サン」で始まるポケモンのデータを取得】

```
SELECT *
    FROM pokemon_1
 WHERE name LIKE 'サン%';
```

【TRY16:ポケモンの名前が「ン」で終わるポケモンのデータを取得】

```
SELECT *
    FROM pokemon_1
 WHERE name LIKE '%ン';
```

ポケモンの高さ・重さを範囲指定して取得しよう

■**カラムが 数値1以上 数値2以下の範囲にあるレコードのみを検索する**

```
SELECT カラム名
    FROM テーブル名
 WHERE カラム名 BETWEEN 数値1 AND 数値2';
```

※同義表現　※BETWEENの方が処理性能が悪いことがあるので注意

```
SELECT カラム名
    FROM テーブル名
 WHERE カラム名 >= 数値1 AND カラム名 <= 数値2';
```

【TRY17:高さが0cm以上1cm以下のポケモンを取得する。】

```
SELECT *
    FROM pokemon_1
 WHERE height BETWEEN 0 AND 1;

SELECT *
    FROM pokemon_1
 WHERE height >= 0 AND height <= 1;
```

【TRY18:重さが10kg以上20kg以下のポケモンを取得する。】

```
SELECT *
    FROM pokemon_1
 WHERE weight BETWEEN 10 AND 20;

SELECT *
    FROM pokemon_1
 WHERE weight >= 10 AND weight <= 20;
```

ポケモンの値リストのを含むデータを取得しよう

■**カラムが値1,値2, 値3...を含むレコードのみを検索する**

```
SELECT カラム名
    FROM テーブル名
 WHERE カラム名 IN (値1, 値2, 値3...)';
```

■**カラムが値1,値2, 値3...を含まないレコードのみを検索する**

```
SELECT カラム名
    FROM テーブル名
 WHERE カラム名 NOT IN (値1, 値2, 値3...)';
```

【TRY19:type_1がみず、ほのお、くさのポケモンのデータを取得する。】

```
SELECT *
    FROM pokemon_1
 WHERE type_1 IN ('みず', 'ほのお', 'くさ');

SELECT *
    FROM pokemon_1
 WHERE type_1 = 'みず' OR type_1 = 'ほのお' OR type_1 = 'くさ';
```

【TRY20:type_1がみず、ほのお、くさのポケモン以外のデータを取得する。】

```
SELECT *
    FROM pokemon_1
 WHERE type_1 NOT IN ('みず', 'ほのお', 'くさ');

SELECT *
   FROM pokemon_1
WHERE NOT type_1 = 'みず' XOR type_1 = 'くさ' XOR type_1 = 'ほのお';
```

ポケモンを複数の値から比較して取得しよう

■**値リストのそれぞれと比較して、いずれかが真なら真**

```
SELECT カラム名
    FROM テーブル名
 WHERE カラム名 ANY (値1, 値2, 値3...)';
```

■**値リストのそれぞれと比較して、すべて真なら真**

```
SELECT カラム名
    FROM テーブル名
 WHERE カラム名 ALL (値1, 値2, 値3...)';
```

【TRY21:高さが1m, 2m, 3mのいずれかよりも低いポケモンのデータを取得する】

```
SELECT *
    FROM pokemon_1
 WHERE height ANY (1, 2, 3);
```

【TRY22:重さが10kg, 100kg, 200kg全てよりも軽いポケモンのデータを取得する】

```
SELECT *
    FROM pokemon_1
 WHERE weight ALL (10, 100, 200);
```

ポケモンの値リストのを含むデータを取得しよう

■**カラムが値1,値2, 値3...を含むレコードのみを検索する**

```
SELECT カラム名
    FROM テーブル名
 WHERE カラム名 IN (値1, 値2, 値3...)';
```

■**カラムが値1,値2, 値3...を含まないレコードのみを検索する**

```
SELECT カラム名
    FROM テーブル名
 WHERE カラム名 NOT IN (値1, 値2, 値3...)';
```

【TRY19:type_1がみず、ほのお、くさのポケモンのデータを取得する。】

```
SELECT *
    FROM pokemon_1
 WHERE type_1 IN ('みず', 'ほのお', 'くさ');

SELECT *
    FROM pokemon_1
 WHERE type_1 = 'みず' OR type_1 = 'ほのお' OR type_1 = 'くさ';
```

【TRY20:type_1がみず、ほのお、くさのポケモン以外のデータを取得する。】

```
SELECT *
    FROM pokemon_1
 WHERE type_1 NOT IN ('みず', 'ほのお', 'くさ');

SELECT *
   FROM pokemon_1
WHERE NOT type_1 = 'みず' XOR type_1 = 'くさ' XOR type_1 = 'ほのお';
```

ポケモンのタイプ一覧を取得して結果を並べ替えよう

■**値の重複を除外する**

SELECT (DISTINCT) カラム名
　FROM テーブル名';

■**検索結果を並べ替える**

SELECT カラム名
　FROM テーブル名
 ORDER BY カラム名 並び順;

※並び順の種類
ASC：昇順　DESC：降順

【TRY21:ポケモンのtype_1から一覧を重複なく取得する。】

```
SELECT DISTINCT type_1
    FROM pokemon_1;
```

【TRY22:TRY21の取得データを昇順（アイウエオ順）に並び替える。】

```
SELECT DISTINCT type_1
    FROM pokemon_1
ORDER BY type_1 ASC;
```

```
/*＜コラム＞
カラム順番を表した番号（列番号）で指定することも可能
カラムの順番が変わると取得されるカラムも変わるので非推奨かつ使いづらい
|1 |    2    |   3  |        4        |   5   |    6   |   7   |   8   |         9         |       10       |
|id|number|name|classification|type_1|type_2|height|weight|Characteristic|explanation|

SELECT DISTINCT type_1
    FROM pokemon_1
ORDER BY 5 ASC;     ※MySQL対応してなかった、、、*/
```

ポケモンのバトルチームをランダムで作ろう

■**取得する件数を設定した数値の件数に制限して取得する。**

```
SELECT カラム名
    FROM テーブル名'
    LIMIT 数値;
```

■**読み飛ばすレコード数を指定して取得する。**

```
SELECT カラム名
    FROM テーブル名
 LIMIT 数値 OFFSET 数値 ;  //必ずLIMITが先OFFSETが後
```

■**ランダムに設定した数値の件数分のレコード数を取得する**

```
SELECT カラム名
    FROM テーブル名
 ORDER BY RAND() LIMIT 数値;
```

【TRY23:ポケモンの名前を先頭から20件取得する。】

```
SELECT name
    FROM pokemon_1
  LIMIT 20;
```

【TRY24:ポケモンの名前を24件読み飛ばして20件取得する。】

```
SELECT name
    FROM pokemon_1
LIMIT 20 OFFSET 24;
```

【TRY25:ポケモンの名前をランダムに6件取得してバトルチームを作る。】

```
SELECT name AS チーム○○
   FROM pokemon_1
ORDER BY RAND()  LIMIT 6;
```

【TRY26:でんきタイプのポケモンの名前をランダムに6件取得してﾊﾞﾄﾙﾁｰﾑを作る。】

```
SELECT name AS チーム○○
   FROM pokemon_1
WHERE type_1 = 'でんき' OR type_2 = 'でんき'
ORDER BY RAND()  LIMIT 6;
```

ポケモンSQL問題集 TRY1-14

-- 【TRY1:pokemon_1テーブル作成】
-- 【TRY2:ピカチュウのデータをテーブルに入れる】
-- 【TRY3:全てのポケモンの名前をpokemon_1テーブルから取得する。】
-- 【TRY4:全てのポケモンの名前，タイプ1，タイプ2，図鑑説明をpokemon_1テーブルから取得する。】
-- 【TRY5:全てのポケモンのカラムをpokemon_1テーブルから取得する。】
-- 【TRY6:重さ100kgよりも重いポケモンを検索する】
-- 【TRY7:高さ1m以下のポケモンを検索する】
-- 【TRY8:高さ1.7mかつ重さ90.5kgのポケモンを検索する】
-- 【TRY9:nameカラムのけつばんをホウオウに変更する。】
-- 【TRY10:ホウオウのデータを削除する】
-- 【TRY11:カラム名をすきなポケモンに変えて取得する】
-- 【TRY12：タイプが単体のポケモンの名前とタイプを取得する。】
-- 【TRY13：タイプが複数のポケモンの名前とタイプを取得する。】
-- 【TRY14:ポケモンの名前に「サイ」が含むポケモンのデータを取得】

ポケモンSQL問題集 TRY15-26

-- 【TRY15:ポケモンの名前が「サン」で始まるポケモンのデータを取得】

-- 【TRY16:ポケモンの名前が「ン」で終わるポケモンのデータを取得】

-- 【TRY17:高さが0cm以上1cm以下のポケモンを取得する。】

-- 【TRY18:重さが10kg以上20kg以下のポケモンを取得する。】

-- 【TRY19:type_1がみず、ほのお、くさのポケモンのデータを取得する。】

-- 【TRY20:type_1がみず、ほのお、くさのポケモン以外のデータを取得する。】

-- 【TRY21:ポケモンのtype_1から一覧を重複なく取得する。】

-- 【TRY22:TRY21の取得データを昇順（アイウエオ順）に並び替える。】

-- 【TRY23:ポケモンの名前を先頭から20件取得する。】

-- 【TRY24:ポケモンの名前を２４件読み飛ばして20件取得する。】

-- 【TRY25:ポケモンの名前をランダムに6件取得してバトルチームを作る。】

-- 【TRY26:でんきタイプのポケモンの名前をランダムに6件取得してﾊﾞﾄﾙﾁｰﾑを作る。】

参考にした書籍

スッキリわかるSQL入門 第2版 ドリル222問付き！

SQL・・・パソコンやスマホのブラウザだけで
SQLの実行や練習ができるクラウドサービス

https://dokoql.com/

KIRI=2800YEN

も入力しない）

ご清聴
ありがとうございました！