**データベース操作・管理**

**データベース操作：**

1．先ずはデータベースに接続する：

* CLI (Command Line Interface)
* GUI（ソフトやブラウザーで直接的な操作）

２．SQL命令を実行する（CRUDを行う）：

* データベース操作：
  + データベース作成：CREATE DATABASE データベース名;
  + データベース表示：SHOW DATABASES;
  + データベース選択：USE データベース名;
  + データベース削除：DROP DATABASE データベース名
* テーブル操作（データベースの中で複数テーブルが存在する）：
  + テーブル作成：CREATE TABLE テーブル名　(

Column名　データ型,

Column名　データ型,

Column名　データ型,

…

PRIMARY KEY (Column名));

* + テーブル削除：DROP TABLE テーブル名;
  + テーブル表示：SHOW TABLES;
  + レコード入力：INSERT INTO テーブル名　（Column名, Column名…） VALUES （フィールド名, フィールド名…）;
  + ※ データ型に合わせた値のみ指定可能
  + レコード取得：SELECT　（Column名, Column名…）　FROM テーブル名　WHERE 条件;
    - SELECT \* FROM テーブル名;
  + データ更新・編集：UPDATE テーブル名 SET Column名 = フィールド名 WHERE 条件
  + データ削除：DELETE FROM テーブル名 WHERE 条件

３．条件の指定：

* 検索する時用いる記号：
  + =, >, <, >=, <=, BETWEEN, IN, LIKE, IS NULL
  + LIKE・NOT LIKE：指定した文字が含まれているデータを検索する
  + SELECT \* FROM `xylog` WHERE (`created\_at` BETWEEN '2021-04-12 20:15:00' AND '2021-04-12 21:00:00') AND `personID`=-1 ORDER BY `uid` DESC LIMIT 100;

４．並べ替えの指定：

* ORDER BY Column名・ORDER BY Column名 DESC

５．データ数の指定

* LIMITはレコードの上限を指定する
* OFFSETは取得開始時間の位置を指定する

６．関数：

* COUNT, MIN, MAX, SUM, AVG
* Group化：SELECT (Column名)　FROM テーブル名　GROUP BY Column名;
* グループ化するデータの条件：SELECT (Column名)　FROM テーブル名　GROUP BY Column名　HAVING 条件;
* WHEREに指定する条件はグループ化する前に実行されて、HAVINGに指定する条件は後に実行される
* JOIN：複数テーブルを結合してデータを取得する
  + 各テーブルのキーとなる列を指定する必要があります
  + INNER JOIN（内部結合）：テーブル間でキーが位置するレコードのみ取得
  + 例文：SELECT \* FROM テーブル名 INNER JOIN IN キーの結合条件;
  + LEFT JOIN, RIGHT JOIN

７．データ型：

* データの形式を指定する
* データを整理するために使われる
* 数値のデータ型：INT（SMALLINT, MEDIUMINT, BIGINT …）, DECIMAL, FLOAT, DOUBLE
  + 値の大きさに応じて適切なデータ型を選ぶのが重要
* 文字列のデータ型：CHAR, VARCHAR, TEXT
* 日付・時間を扱うデータ型：DATETIME, DATE, TIME, YEAR
* 有無を扱うデータ型：BOOLEAN（true,1（真）とfalse,0（偽））

８．データの制約と制限：

* 不正なデータの入力を防ぐ
* 制約：格納出来るデータに制限をかける
* 属性：値にある規則で整える（AUTO\_INCREMENT, NOT NULL, UNIQUE, DEFAULT）
* PRIMARY KEY：レコードを特定する列
* FOREIGN KEY：他のテーブルに関連付けた場合に依存するカラム
  + CREATE TABLE テーブル名 (Column名) FOREIGN KEY　（Column名） REFERENCES 紐づくテーブル名（関連Column名）;

９．Transaction：データベースに対して行われる複数の処理

* Transactionの特性：ACID (Atomic, Consistent, Isolated, Durable)
  + ACIDによってデータの整合性・正確さを確実に保つことが出来ます
  + Atomic（原子性）：処理が全て実行されるか・全て実行されないかに分ける
  + Consistent（一貫性）：設定された条件に満たすことを確認する、データの整合性を保つ
  + Isolated（独立性）：外部から結果だけを見ることが出来ます、処理途中の状態は他の処理に影響しない
  + Durable（永続性）：トランザクションの結果は間違いなく保存される（電源が切れるなどの緊急事態でも失われない）
* COMMIT：トランザクションが成功した場合、データベースに反映されるためにCOMMITを行う。
* ROLLBACK：処理内に問題が発生したら、処理を取り消して前の整合性があった状態に取り戻す
* DEADLOCK：同時に複数トランザクションが行われる場合、最初の処理が終了次第次の処理が実行される
  + 処理が終わるまでロックがかかって、解除されるまで待ってから処理が実行される
  + 対策として、トランザクションからアクセスするデータの順番を統一する