5. SQL基礎：SELECT文による効率的なデータ検索と操作の基本

URL: https://www.kkaneko.jp/de/ds/index.html

金子邦彦

アドバイス

- アクティブな学習を実践しよう

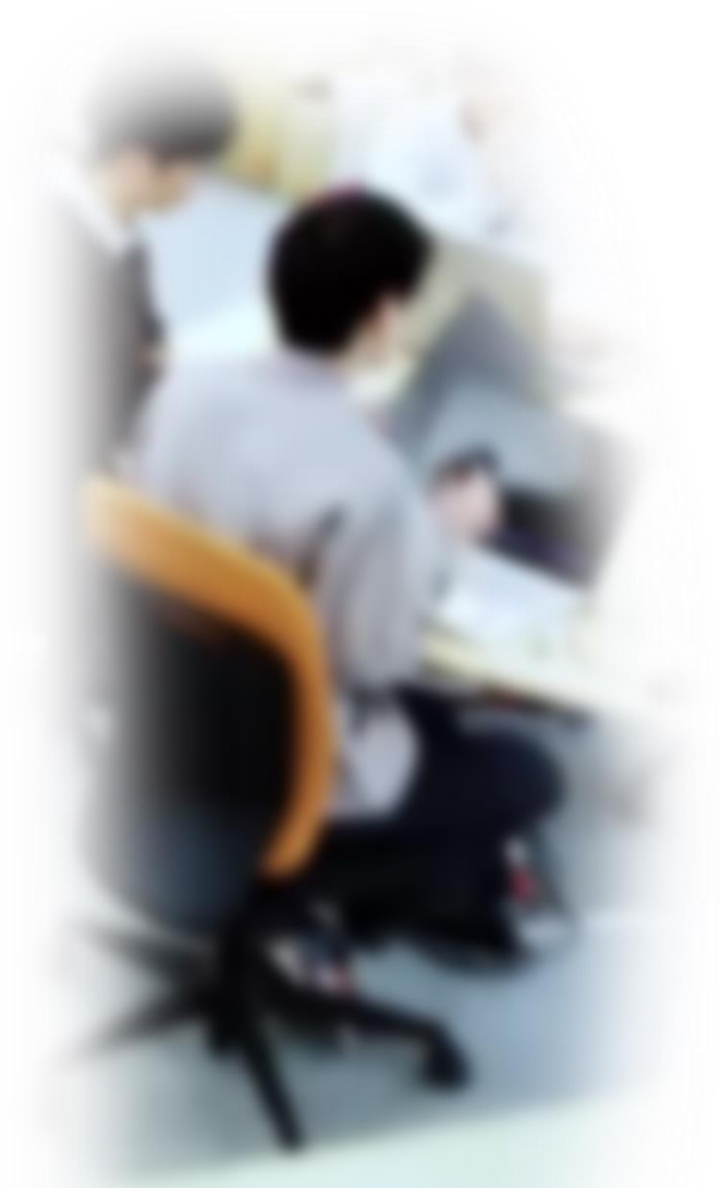
SQLを学ぶ際に、**プログラムを変更した結果を実際に見る**ことも心がけましょう。実際のデータベース操作を通じて学習を深めます。

- 簡単なスタートから

初めはシンプルなものからスタートしましょう。反復練習しましょう。

- ステップ・バイ・ステップで応用に進む

SQLスキルを向上させるために、**少しずつ難易度を上げ**、今まで自分ができなかったことにも**チャレンジ**しましょう。



アウトライン

1. イントロダクション
2. SELECT, FROM, WHERE の役割
3. WHERE による選択、IN
4. 重複の排除、集約
5. 演習

SQLFiddle のサイトにアクセス

Webブラウザを使用

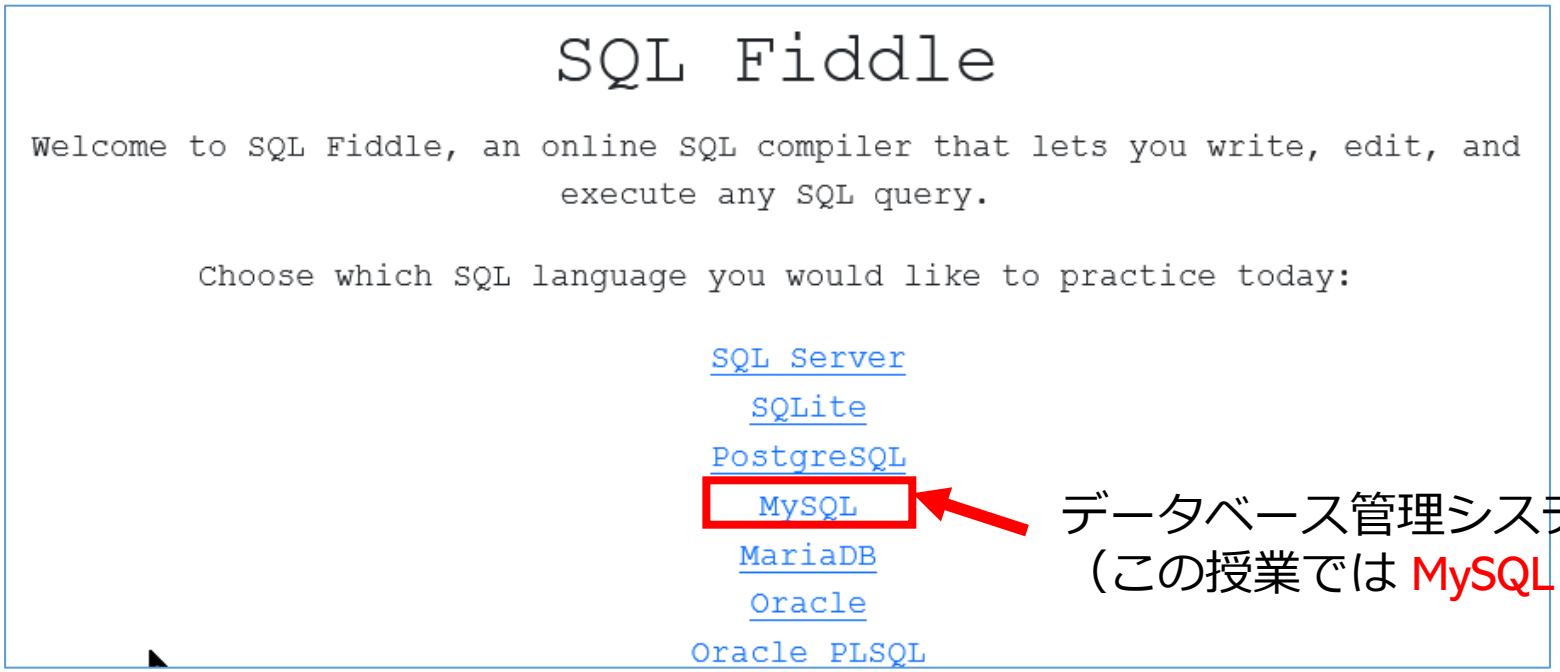
1. ウェブブラウザを開く
2. アドレスバーにSQLFiddleのURLを入力

http://sqlfiddle.com/

3. **MySQL を選ぶ**

URLが分からないときは、Googleなどの**検索エンジン**を利用。
「**SQLFiddle**」と**検索**し、表示された結果からSQLFiddleの
ウェブサイトをクリック。

SQLFiddle でのデータベース管理システムの選択



データベース管理システムの選択
（この授業では MySQL を使用）

SQLFiddle の画面

上のパネル: SQLの入力（複数可能）

- **テーブル定義** CREATE TABLE
- **データの追加** INSERT INTO
- **SQL問い合わせ**。SELECT, FROM, WHERE など

5-1. イントロダクション

リレーショナルデータベースの仕組み

- データを**テーブル**と呼ばれる**表形式で保存**
- **テーブル間**は**関連**で結ばれる
- 複雑な構造を持ったデータを効率的に管理することを可能

商品

ID	商品名	単価
1	みかん	50
2	りんご	100
3	メロン	500

関連

購入

購入者	商品番号
X	1
X	3
Y	2

リレーショナルデータベースの重要性

1. **データの整合性**：リレーショナルデータベースは，**データの整合性を保持するための機能**を有する．これにより，誤ったデータや矛盾したデータが保存されるのを防ぐことができる．
2. **柔軟な問い合わせ（クエリ）能力**：リレーショナルデータベースのSQL（Structured Query Language）（データベース操作言語）の使用により，**複雑な検索やデータの抽出**が可能になる．
3. **トランザクション機能**：一連の操作全体を一つの単位として取り扱うことができる機能．これにより，**データの一貫性と信頼性が向上**する．
4. **セキュリティ**：**アクセス権限の設定**などにより，セキュリティを確保する．

データの安全な保管，効率的なデータ検索・操作，ビジネスや研究の意思決定をサポート．

SQL 理解のための前提知識

○ テーブル

データを**テーブル**と呼ばれる**表形式で保存**

ID	商品名	単価
1	みかん	50
2	りんご	100
3	メロン	500

購入者	商品番号
X	1
X	3
Y	2

○ 問い合わせ（クエリ）

- **問い合わせ（クエリ）**は、**データベース**から**必要なデータを検索、加工するための指令**
- SELECT, FROM, WHERE など、**多様**なコマンドが存在。
- **結合、集計、ソート、副問い合わせ**など、高度な操作も可能

SQL によるテーブル定義

- **テーブル名**：**記録**
- **属性名**：**名前、得点、居室**
- **属性のデータ型**：**テキスト、数値、テキスト**
- データの整合性を保つための**制約**：**なし**

```
CREATE TABLE 記録 (
    名前 TEXT,
    得点 INTEGER,
    居室 TEXT);
```

データ追加のSQL

記録

名前	得点	居室
徳川家康	85	1階
源義経	78	2階
西郷隆盛	90	3階
豊臣秀吉	82	1階
織田信長	75	2階

```
INSERT INTO 記録 VALUES('徳川家康', 85, '1階');
INSERT INTO 記録 VALUES('源義経', 78, '2階');
INSERT INTO 記録 VALUES('西郷隆盛', 90, '3階');
INSERT INTO 記録 VALUES('豊臣秀吉', 82, '1階');
INSERT INTO 記録 VALUES('織田信長', 75, '2階');
```

演習１．テーブル定義とデータの追加

【トピックス】

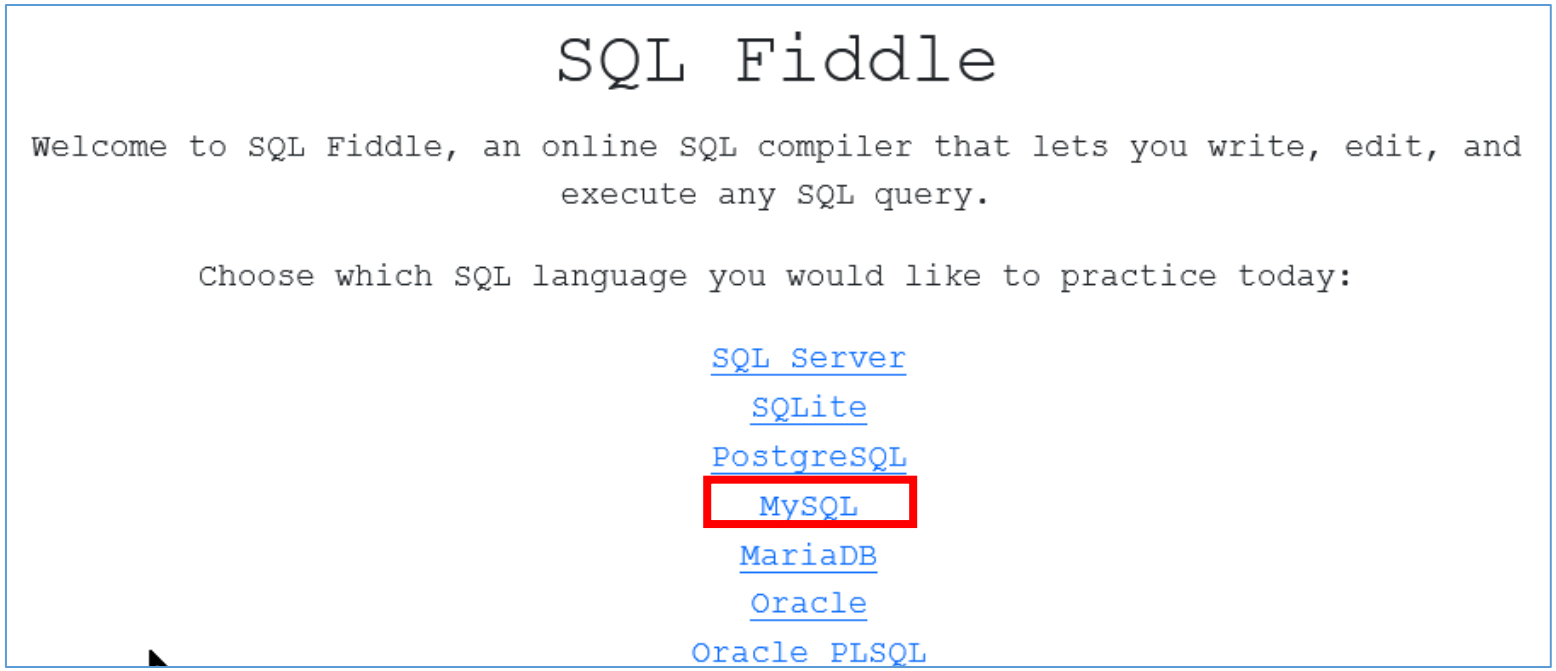
1. **SQL によるテーブル定義**
2. **SQL によるデータの追加**
3. **問い合わせ（クエリ）による確認**

Webブラウザを使用

① アドレスバーにSQLFiddleのURLを入力

http://sqlfiddle.com/

② 「**MySQL**」を選択



③ **上のパネル**に、**テーブル定義**と**データの追加**と**問い合わせ**を行う SQL を入れる。（SQLFiddleで、最初に出てくるSQLは不要なので消す）。

```
create table 記録 (
  名前 text,
  得点 integer,
  居室 text
);
insert into 記録 values ('徳川家康', 85, '1階');
insert into 記録 values ('源義経', 79, '2階');
insert into 記録 values ('西郷隆盛', 90, '3階');
insert into 記録 values ('豊臣秀吉', 82, '1階');
insert into 記録 values ('織田信長', 75, '2階');
select * from 記録;
```

④ 「**Execute**」をクリック

SQL 文が**実行**され、結果が表示される。

⑤ 下のパネルで、**結果を確認。**

名前	得点	居室
徳川家康	85	1階
源義経	79	2階
西郷隆盛	90	3階
豊臣秀吉	82	1階
織田信長	75	2階

あとで使用するのでブラウザを閉じないこと

次の SQL を試してみる。

select 名前 from 記録;

select 得点 from 記録;

結果

```
+--------------+
| 名前 |
+--------------+
| 徳川家康 |
| 源義経 |
| 西郷隆盛 |
| 豊臣秀吉 |
| 織田信長 |
+--------------+
+--------+
| 得点 |
+--------+
| 85 |
| 79 |
| 90 |
| 82 |
| 75 |
+--------+
```

あとで使用するのでブラウザを閉じないこと

5-2. SELECT、FROM、WHEREの役割

最初に全体像を把握する学習のメリット

- **広範な理解**：SQLの**多様な機能と用途**を**早期に理解**できます。
- **学習意欲の向上**：多くの機能を初めて体験することで、何ができるのかを知り、**興味が湧きます**。
- **効率的な学習**：全体像を先に知ることで、**後の学習がスムーズに進みます**。

注意点

- **自分のペースで**：最初の授業ですべてをマスターする必要はありません。
- **基礎の確立**：**次回以降**で基本的な部分をしっかり学び、その後で応用に進みます。

SQL 問い合わせの全体像①

- **select**: **データの検索・加工や射影**

  例：**SELECT * FROM employees**
- **from**: 問い合わせ**対象テーブル**の指定

  例：**FROM employees**
- **where**: 条件に一致する行を**選択**

  例：**WHERE age > 30**
- **join, on**: 結合、結合条件

  例：**JOIN B ON A.b_id = B.id**
- **insert into**: 新しい行の**追加**（挿入）
- **update, set:** 条件に一致する行を**更新**
- **delete from**: 条件に一致する行を**削除**

SQL 問い合わせの全体像②

- **distinct:** 重複行の除去

  例：**SELECT DISTINCT age FROM employees**

- **count:** 行数のカウント

  例：**SELECT COUNT(*) FROM employees**

- **avg, max, min, sum**: 平均、最大、最小、合計の計算

  例：**AVG(salary), MAX(salary)**

- **group by**: 属性でグループ化

  例：**GROUP BY department_id**

- **order by:** 並べ替え（ソート）

  例：**ORDER BY age**

- 副問い合わせ: SQL文の中に別のSQL文を埋め込む。

  例：**WHERE salary > (SELECT AVG(salary) FROM employees)**

SELECT, FROM, WHERE を学ぶことのメリット

- **テーブルによるデータ管理**
  リレーショナルデータベースのテーブルによるデータ管理についての理解が深まる。
- **SQLの柔軟性**
  条件を指定してテーブルからデータを取得する多彩な方法を学ぶ。
- **SQL によるデータアクセス**
  テーブルから必要なデータを取得するスキルを習得。

5-3. WHERE による選択、IN の役割

WHERE による選択

問い合わせ（クエリ）では、**条件を指定**することにより、
データの行単位での選択を行う

WHERE 得点 > 80

「**得点が80より大きい**」行を選択

WHERE 得点 **BETWEEN** 80 **AND** 85

「**得点が80以上かつ85以下**」の範囲にある行を選択

WHERE 居室 **IN** ('1階', '2階');

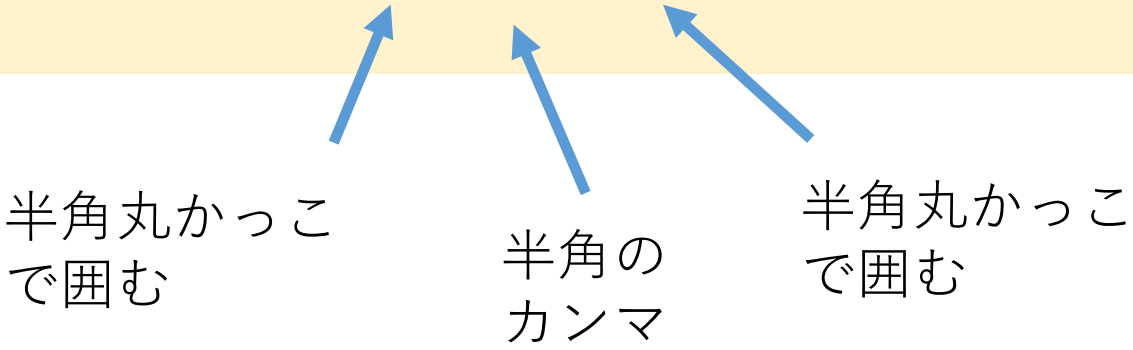
「**居室が'1階'または'2階'**」のいずれかに一致する行を選択

IN の役割

複数の値と比較.
そのうち１つの値でも一致するものを結果とする

「**居室が'1階'または'2階'**」のいずれかに一致する行を選択

```
SELECT *
FROM 記録
WHERE 居室 IN ('1階', '2階');
```



5-4. 重複行の除去、集約

DISTINCT は重複行の除去

SELECT 居室 **FROM** 記録;

結果

居室
1階
2階
3階
1階
2階

SELECT DISTINCT 居室 **FROM** 記録

結果

居室
1階
2階
3階

集約

AVG, MAX, MIN, SUM: 平均、最大、最小、合計

COUNT: 行数

記録

名前	得点	居室
徳川家康	85	1階
源義経	78	2階
西郷隆盛	90	3階
豊臣秀吉	82	1階
織田信長	75	2階

SELECT AVG(得点) **FROM** 記録;

82

SELECT MAX(得点) **FROM** 記録;

90

SELECT MIN(得点) **FROM** 記録;

75

SELECT SUM(得点) **FROM** 記録;

410

SELECT * **FROM** 記録;	**テーブルのすべて**
SELECT 居室 **FROM** 記録;	「**居室**」**の列**をすべて
SELECT DISTINCT 居室 **FROM** 記録;	**重複行**（同じ値の行）を**除去**
SELECT 名前, 得点 **FROM** 記録 **WHERE** 得点 > 80;	「名前」と「得点」の列で、「**得点が80より大きい**」行を選択
SELECT 名前, 得点 **FROM** 記録 **WHERE** 得点 **BETWEEN** 80 **AND** 85;	「名前」と「得点」の列で、「**得点が80以上かつ85以下**」の範囲にある行を選択
SELECT AVG(得点) **FROM** 記録;	すべての「**得点**」**の値**の**平均**
SELECT * **FROM** 記録 **WHERE** 居室 **IN** ('1階', '2階')	「**居室が'1階'または'2階'**」のいずれかに一致する行を選択

5-5. 演習

演習２．SQL による問い合わせ（クエリ）

【トピックス】

1. **選択**
2. **DISTINCT による重複行除去**
3. **パターンマッチング**
4. **IN を用いた条件指定**

① **上のパネル**に、**テーブル定義**と**データの追加**と**問い合わせ**を行う SQL を入れ実行。（**以前の SQL は不要なので消す**）

```
create table 記録 (
   名前 text,
   得点 integer,
   居室 text
);
insert into 記録 values ('徳川家康', 85, '1階');
insert into 記録 values ('源義経', 79, '2階');
insert into 記録 values ('西郷隆盛', 90, '3階');
insert into 記録 values ('豊臣秀吉', 82, '1階');
insert into 記録 values ('織田信長', 75, '2階');
select 居室 from 記録;
select distinct 居室 from 記録;
```

②「**Execute**」をクリック

③ 下のパネルで、**結果を確認**。

```
+--------+
| 居室 |
+--------+
| 1階 |
| 2階 |
| 3階 |
| 1階 |
| 2階 |
+--------+
+--------+
| 居室 |
+--------+
| 1階 |
| 2階 |
| 3階 |
+--------+
```

④ **上のパネル**に、**問い合わせ（クエリ）**を行う SQL を入れる。（**下に追加**）

```
select 名前, 得点 from 記録 where 得点 > 80;
select 名前, 得点 from 記録 where 得点 between 80 and 85;
```

⑤「**Execute**」をクリック

⑥ 下のパネルで、**結果を確認。**

```
+--------------+--------+
| 名前 | 得点 |
+--------------+--------+
| 徳川家康 | 85 |
| 西郷隆盛 | 90 |
| 豊臣秀吉 | 82 |
+--------------+--------+
+--------------+--------+
| 名前 | 得点 |
+--------------+--------+
| 徳川家康 | 85 |
| 豊臣秀吉 | 82 |
+--------------+--------+
```

⑦ **上のパネル**に、**問い合わせ（クエリ）**を行う SQL を入れる。（**下に追加**）

```
select avg(得点) from 記録;
```

⑧「**Execute**」をクリック

⑨ 下のパネルで、**結果を確認。**

```
+---------------+
| avg(得点)  |
+---------------+
| 82.2000 |
+---------------+
```

⑩ **上のパネル**に、**問い合わせ（クエリ）**を行う SQL を入れる。（**下に追加**）

```
select * from 記録 where 居室 in ('1階', '2階');
```

⑪「**Execute**」をクリック

⑫ 下のパネルで、**結果を確認。**

```
+--------------+--------+--------+
| 名前 | 得点 | 居室 |
+--------------+--------+--------+
| 徳川家康 | 85 | 1階 |
| 源義経 | 79 | 2階 |
| 豊臣秀吉 | 82 | 1階 |
| 織田信長 | 75 | 2階 |
+--------------+--------+--------+
```

余裕がある人向けの演習

- テーブル「**記録**」を使用してください
- 後ろのページに解答例を載せているので活用してください
- SQLFiddle などを利用し、**実際に実行してみると、理解が進み効果的**です。

追加演習１．**得点が80以上の人物を選択する**

目的: WHERE句を使用して特定の条件を満たす行を選択する方法を学ぶ

得点が80以上のすべての人物を選択してください。

ヒント: WHERE句を使用して得点の条件を指定

追加演習２．**1階に住む人物の名前を重複なく選択する**

目的: DISTINCTを使用して重複する値を排除する方法を学ぶ。

1階に住む人物の名前を選択してください。そのとき、重複なく表示するようにしてください。

ヒント: DISTINCTとWHERE句を組み合わせて使用

追加演習３．**得点が70から90の範囲内の人物の名前と居室を選択**

目的: BETWEENを使用して、特定の範囲内のデータを選択する方法を学ぶ。

BETWEEN を用いて、得点が70から90の範囲内のすべての人物の名前と居室を選択してください。得点が 70, 90 の人も含めてください。

ヒント: BETWEEN を使用して得点の範囲を指定

追加演習４．**'織田'で始まる名前の人を選択**

目的: LIKEを使用して特定のパターンに一致する行を選択する方法を学ぶ。

'織田'で始まる名前の人をすべて選択してください。

ヒント: LIKEとワイルドカード%を使用してパターンマッチングを行います。

追加演習５．**得点の最大値を得る**

目的: MAX を使用して「得点」の列の最大値を得る方法を学ぶ

得点の最大値を得てください

ヒント: SELECT MAX(得点) のように書くことで、得点の最大値を得ることができます。

追加演習１　**解答例**: SELECT * FROM 記録 WHERE 得点 >= 80;

追加演習２　**解答例**: SELECT DISTINCT 名前 FROM 記録 WHERE 居室 = '1階';

追加演習３　**解答例:** SELECT 名前, 居室 FROM 記録 WHERE 得点 BETWEEN 70 AND 90;

追加演習４　**解答例**：SELECT * FROM 記録 WHERE 名前 LIKE '織田%';

追加演習５　**解答例**：SELECT MAX(*) FROM;

全体まとめ ①

SQL理解の前提知識

- **テーブル**: データを表形式で保存する構造。
- **問い合わせ（クエリ）**: データベースからデータを検索・加工するための指令。SELECT、FROM、WHEREなどのコマンドが存在し、高度な操作も可能。

SQLの select, from, where

select

- 問い合わせ（クエリ）のための基本的な命令。
- 取得したいデータの指定

from

- データ取得の対象となるテーブルを指定

where

- 特定の条件を満たす行の選択

全体まとめ ②

WHERE による選択

問い合わせ（クエリ）では、条件を指定してデータの行単位で選択。

- WHERE 得点 > 80　**得点が80より大きい行**を選択。
- WHERE 得点 BETWEEN 80 AND 85　**得点が80以上かつ85以下の行**を選択。
- WHERE 居室 LIKE '%階';　**居室が'階'で終わる行を選択**。ワイルドカード（%）は任意の文字列を表す。
- WHERE 居室 IN ('1階', '2階')　**居室が'1階'または'2階'**に一致する行を選択。

全体まとめ ③

居室
1階
2階
3階
1階
2階

居室
1階
2階
3階

DISTINCTは重複行の除去

- SELECT 居室 FROM 記録
- SELECT DISTINCT 居室 FROM 記録

集約

AVG, MAX, MIN, SUM: 平均、最大、最小、合計。

- SELECT AVG(得点) FROM 記録;
- SELECT MAX(得点) FROM 記録;
- SELECT MIN(得点) FROM 記録;
- SELECT SUM(得点) FROM 記録;

① テーブルによるデータ管理の理解

SELECT、FROM、WHEREを学ぶことで、リレーショナルデータベース内のテーブルにデータをどのように管理するかについて深い理解が得られます。

② SQLの柔軟性の理解

SQLは柔軟な言語です。SQL の柔軟性を学ぶことで、必要なデータを効率的に取得できるスキルを習得できます。

③ SQLによるデータアクセスのスキル向上

SELECT、FROM、WHEREを理解し、正確に使用できるようになることで、データベースから必要なデータを取得するスキルが向上します。これはデータ分析、レポート作成、ビジネスの意思決定において有用です。