

(個人ワーク2)

SQLで凝ったことをやってみましょう

---



2025年 IT総合研修  
担当講師：花原

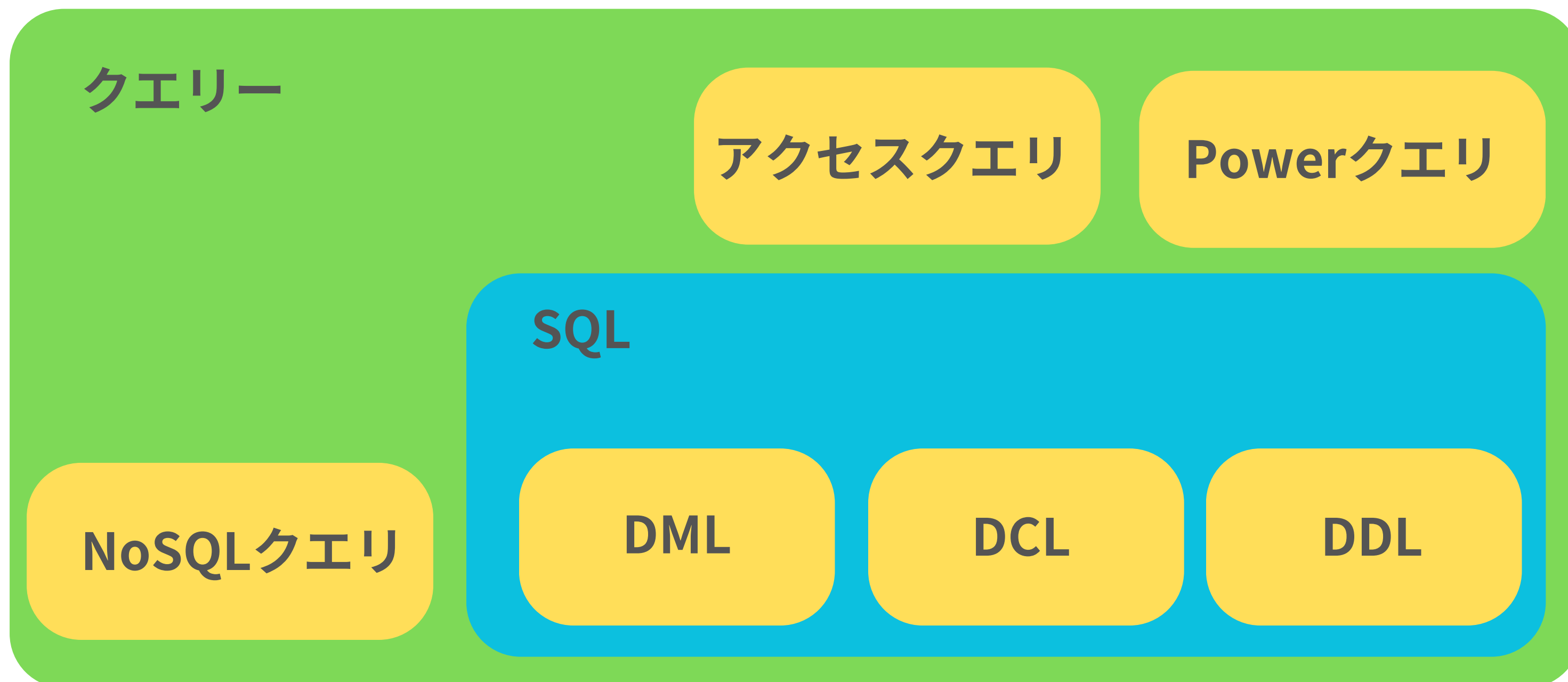
# その前に昨日の補足

---

## クエリーって何？ SQLとは違うの？

Q.アクセスでクエリーがあるけど、クエリーとSQLは違うの？

A.クエリーとは「データを要求する処理」。RDBでのクエリーがSQLみたいな感じ。



Q.演算子と関数って何違いは？

- A. 演算子は2つの値から1つの値を求めるもの。 . . . . + - / × など  
関数はXの値が決まればYの値が決まるもの。 . . . . AVG() MAX() など

ちなみに、SQLでも演算子は使えます。ちょっとやってみましょう。

## データの一貫性、整合性を保持するためのルール

形	種類
INTEGER	整数型
TEXT	文字列
BLOB	データ
REAL	小数点
NUMERIC	固定小数点



# データ型って何なんだよ！

$$1 + 1 = ???$$

1は数字として扱う



$$1 + 1 = 2$$

1は文字として扱う



$$1 + 1 = 11$$

# データ型が一貫性に何に影響するの？

ここはデータ型が無い世界・・・

あい + うえお = あいうえお

20250820 + 12 = ? ? ? ?

12日後？ 12を加算する？ 12を結合する？

0901234567890

これは数字？ 文字列？

10,11,12,1,3

昇順はどうなる？

メモリの最適化、エラーハンドリングなど理由もあるが、  
人が暗黙知で解釈していることを明示的に指定するためという理由がある、

## じゃあSELECTだけでどうしろっていの？



### 「本番からSELECTで抜く」

本番環境からSELECTでデータを抽出してSQLiteにインポート。  
そこでSELECTやUPDATE,INSERTをして分析するなど  
工夫次第でデータ活用はできる。

大きなSELECTのSQLでのレスポンス問題も発生し辛い。

## うちの図書館システムSQL使えないよ



### 知識の有無は大事なポイント

どんなデータを取り出せるかを知ってるだけでも  
保守チームへの依頼精度が変わる。

また、図書館システム以外でも入退館システムなど  
「ログ」を出力できれば、それをインポートして  
SQLiteで解析することができる。

# 別に覚えなくてもいい話。

## SQLiteでの便利な書法

The screenshot shows a SQLite IDE window with a menu bar (データベース構造(D), データ閲覧(B), プラグマ編集(R), SQL実行(X)) and a toolbar. The main area contains a SQL query:

```
1 select a.氏名, a.部署コード, b.部署名
2 from 従業員 a, 部署 b
3 where 1=1
4 and a.部署コード=b.部署コード
5 and a.氏名 like '山口%'
6 -- and a.氏名 lke 'マルコ%'
```

Below the query, a table displays the results:

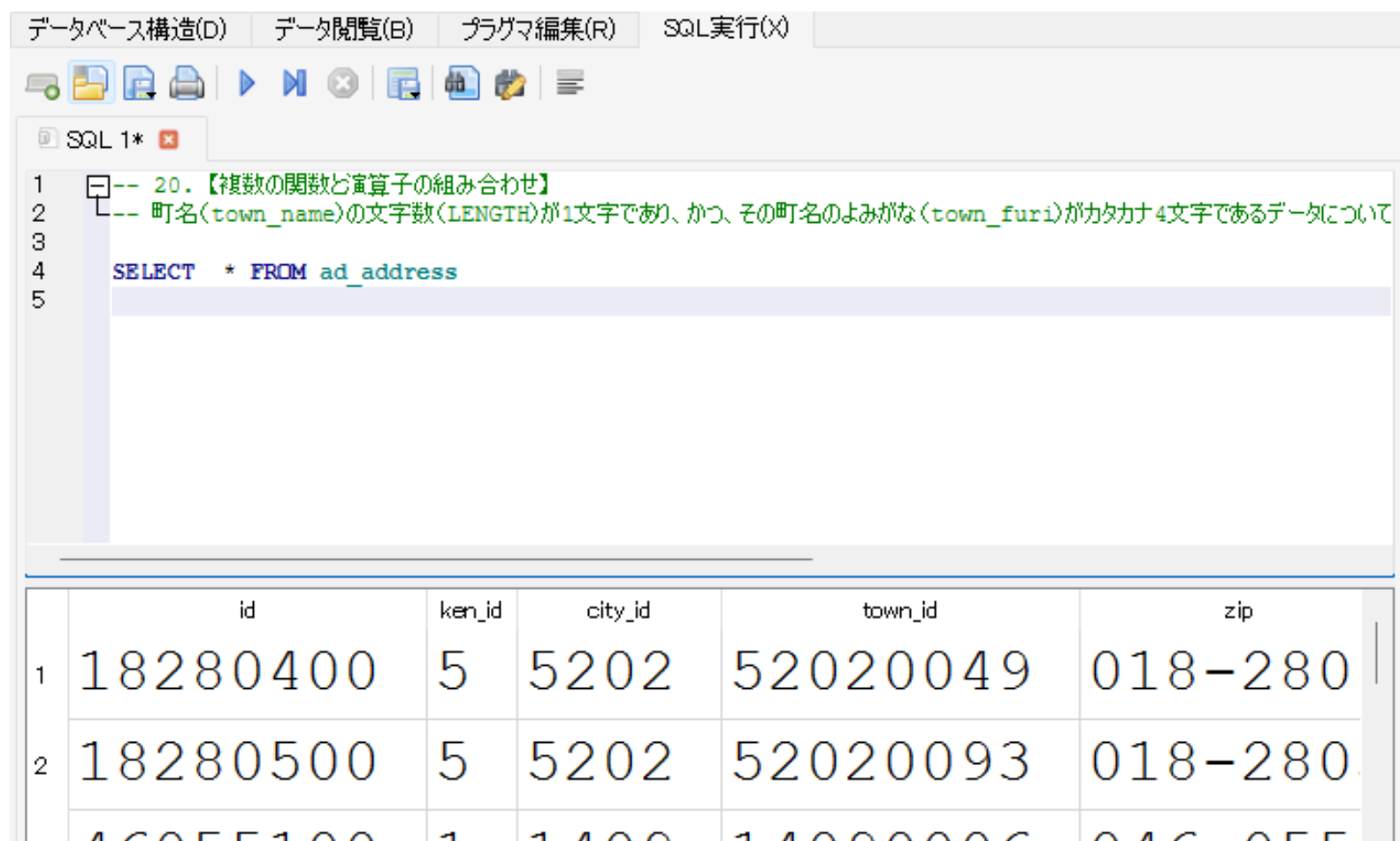
氏名	部署コード	部署名
1 山口重雄	A	本社

The screenshot shows a SQLite IDE window with a menu bar (ファイル(E), 編集(E), ビュー(V), ツール(L), ヘルプ(H)) and a toolbar. The main area contains the following SQL commands:

```
1 COMMIT;
2
3 ROLLBACK;
```

Red and blue boxes highlight the `COMMIT;` and `ROLLBACK;` commands respectively. The toolbar includes buttons for "新しいデータベース(N)", "データベースを開く(O)", "変更を書き込み(W)", and "変更を取り消し(R)".

# SQLを作る手順（人それぞれだけど）

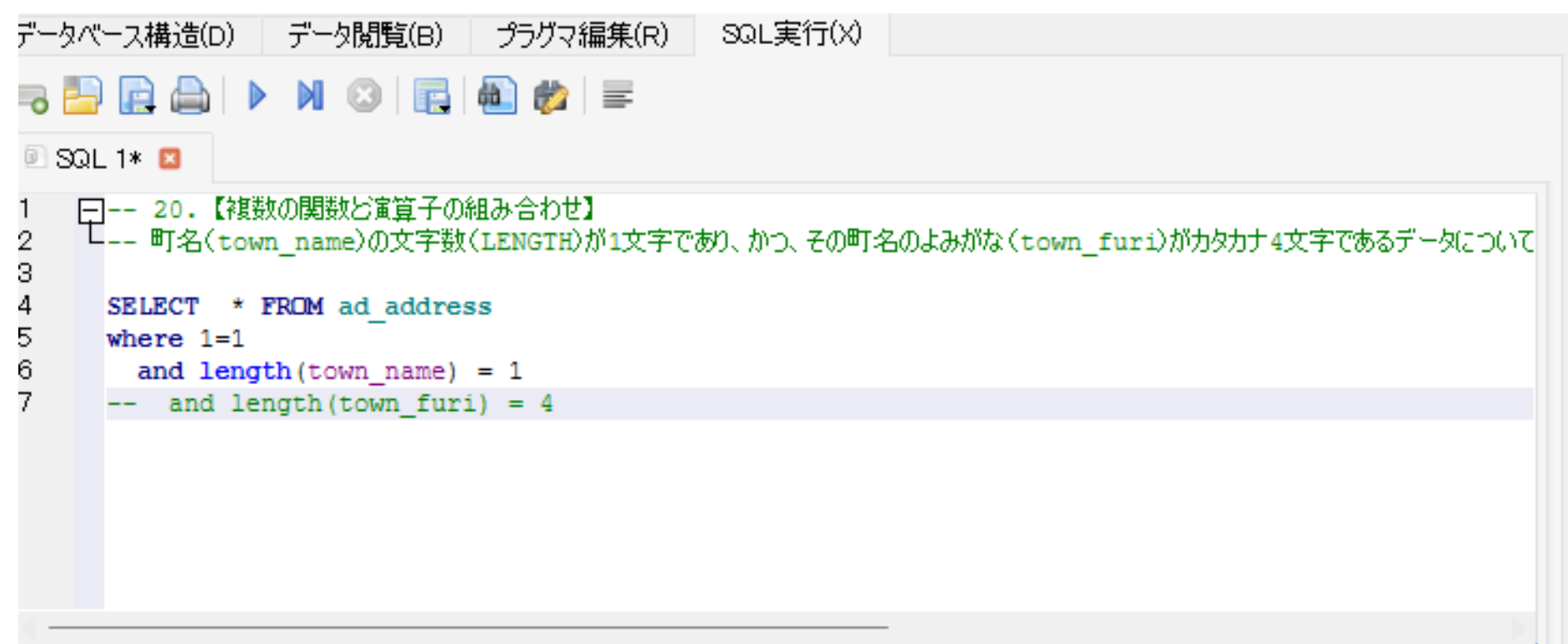


データベース構造(D) データ閲覧(B) プログラム編集(R) SQL実行(X)

SQL 1\*

```
1 -- 20. 【複数の関数と演算子の組み合わせ】  
2 -- 町名(town_name)の文字数(LENGTH)が1文字であり、かつ、その町名のよみがな(town_furi)がカタカナ4文字であるデータについて  
3  
4 SELECT * FROM ad_address  
5
```

	id	ken_id	city_id	town_id	zip
1	18280400	5	5202	52020049	018-280
2	18280500	5	5202	52020093	018-280
	40055100	1	1400	14000000	040-055



データベース構造(D) データ閲覧(B) プログラム編集(R) SQL実行(X)

SQL 1\*

```
1 -- 20. 【複数の関数と演算子の組み合わせ】  
2 -- 町名(town_name)の文字数(LENGTH)が1文字であり、かつ、その町名のよみがな(town_furi)がカタカナ4文字であるデータについて  
3  
4 SELECT * FROM ad_address  
5 where 1=1  
6 and length(town_name) = 1  
7 -- and length(town_furi) = 4
```

(個人ワーク2)

SQLで凝ったことをやってみましょう

---



2025年 IT総合研修  
担当講師：花原

**「書かれた通りに実行する」ではない**



**「最終形をイメージするして実行する」**

「資料にSQLがありますので、それ通りに実行する」  
それも一つですが、実行した後の形をイメージしてください。  
「こういうデータを取り出したい」をイメージしてそのSQLを見る。  
それが理解する近道です。

SQLはすべての行に意味がある。意味の無い行はないです。

(意図して書かない限り)

## 「後で自分で調べよう」の後は来ない



そんな暇ないですよね？

今から少し複雑なことをします。止まったら挙手してください。

「花原進めるの早くてできなかった」からまた後で。

そんな日は来ません。次のStepに進められなかったら挙手してください。

**手を挙げることもより分らないまま次に行く方が恥ずかしいです。**

そしてそういう講義をする私をもっと恥ずかしいのです。

困ったら挙手 or Slackへ！



最高の講師陣が駆け付けます

楽しもう！



# SELECTの応用形（結合）

従業員TBL

従業員番号	氏名	部署コード
000001	山口重雄	A
000002	マルコロドリゲス	C
000003	丸本美幸	B
000004	ニコライボルコフ	D

部署TBL

部署コード	部署名	住所
A	本社	東京都中央区
B	日本首都支店	京都府京都市
C	営業所	東部ごみ処理センター
E	左遷支店	シベリア市中央区

---

二つのテーブルの情報を一回で取得したいときは結合することで

例えば、従業員テーブルの部署コードから、部署の住所を結合させて部署住所の表を作ってみましょう。

# SELECTの応用形（等価結合）

従業員TBL

従業員番号	氏名	部署コード
000001	山口重雄	A
000002	マルコドリゲス	C
000003	丸本美幸	B
000004	ニコライボルコフ	D

部署TBL

部署コード	部署名	住所
A	本社	東京都中央区
B	日本首都支店	京都府京都市
C	営業所	東部ごみ処理センター
E	左遷支店	シベリア市中央区

```
SELECT a.氏名,a.部署コード,b.部署名  
FROM 従業員TBL a, 部署TBL b  
WHERE a.部署コード = b.部署コード
```

結果

氏名	部署コード	部署名
山口重雄	A	本社
マルコドリゲス	C	営業所
丸本美幸	B	日本首都支店

---

2つのテーブルがある場合、どちらの表の項目か分からないので、別名（aやb）を付けて表記します。

等価結合はどちらにもデータがあるものだけでてくるので、部署コードDの人は、部署TBLにデータがないので、取得されません。

# SELECTの応用形（左結合）

従業員TBL

従業員番号	氏名	部署コード
000001	山口重雄	A
000002	マルコドリゲス	C
000003	丸本美幸	B
000004	ニコライボルコフ	D

部署TBL

部署コード	部署名	住所
A	本社	東京都中央区
B	日本首都支店	京都府京都市
C	営業所	東部ごみ処理センター
E	左遷支店	シベリア市中央区

```
SELECT a.氏名,a.部署コード,b.部署名
FROM
  従業員TBL a
LEFT JOIN
  部署TBL b
ON
  a.部署コード = b.部署コード;
```

結果

氏名	部署コード	部署名
山口重雄	A	本社
マルコドリゲス	C	営業所
丸本美幸	B	日本首都支店
ニコライボルコフ	D	(NULL)

---

左結合をすると、左の表の情報をすべて結果に出します。

その際、右側に指定したテーブルに値が無ければNULLで返却されます。

絶対にメインテーブルを全件出したい場合は左結合を使います。

# SELECTの応用形（左結合）

従業員TBL

従業員番号	氏名	部署コード
000001	山口重雄	A
000002	マルコドリゲス	C
000003	丸本美幸	B
000004	ニコライボルコフ	D

部署TBL

部署コード	部署名	住所
A	本社	東京都中央区
B	日本首都支店	京都府京都市
C	営業所	東部ごみ処理センター
E	左遷支店	シベリア市中央区
A	新本社 2	東京都千代田区

```
SELECT a.氏名,a.部署コード,b.部署名
FROM
  従業員TBL a
LEFT JOIN
  部署TBL b
ON
  a.部署コード = b.部署コード;
```

結果

氏名	部署コード	部署名
山口重雄	A	本社
山口重雄	A	新本社 2
マルコドリゲス	C	営業所
丸本美幸	B	日本首都支店
ニコライボルコフ	D	(NULL)

ただし、左側を全部出すというのは、左側をメインテーブルとして、全パターン出すということです。

なので、部署TBLに間違っって部署コードAを2行作ると、山口さんが分身します。ヤバイ。結合条件や部署テーブルの条件を見直しましょう。

# SELECTの応用形（右結合）

従業員TBL

従業員番号	氏名	部署コード
000001	山口重雄	A
000002	マルコドリゲス	C
000003	丸本美幸	B
000004	ニコライボルコフ	D

部署TBL

部署コード	部署名	住所
A	本社	東京都中央区
B	日本首都支店	京都府京都市
C	営業所	東部ごみ処理センター
E	左遷支店	シベリア市中央区

```
SELECT a.氏名,a.部署コード,b.部署名
FROM
  従業員TBL a
RIGHT JOIN
  部署TBL b
ON
  a.部署コード = b.部署コード;
```

結果

氏名	部署コード	部署名
山口重雄	A	本社
マルコドリゲス	C	営業所
丸本美幸	B	日本首都支店
(NULL)	E	左遷支店

---

左があれば右もあります。

右結合をすると、右のテーブルの組み合わせを全件表示します。

## SELECTの応用形（おまけ：等結合）

```
SELECT a.氏名,a.部署コード,b.部署名  
FROM 従業員TBL a, 部署TBL b  
WHERE a.部署コード = b.部署コード
```



```
SELECT a.氏名,a.部署コード,b.部署名  
FROM  
    従業員TBL a  
INNER JOIN  
    部署TBL b  
ON  
    a.部署コード = b.部署コード;
```

---

等結合だけ書き方が変わって思いましたよね。

SQL99以降の標準書法では 等結合はinner joinで書くべきと言われて  
います。左はその当時に戦ってきた古い人間の書き方です。