

第4回：並び替える - ORDER BY

結果を見やすいように昇順・降順で並び替えられるようになる

● 本日のゴール

✓ 結果を見やすいように昇順・降順で並び替えられるようになる

● 座学パート

1. なぜ並び替えが必要？

売上ランキングを作りたい

新しい順に表示したい

価格の安い順に見たい

→ データを「意味のある順番」で見ることで、分析や意思決定がしやすくなる

2. ORDER BY句の基本構文

```
SELECT 列名 FROM 'ファイル名' ORDER BY 並び替えたい列名;
```

-- 降順（大きい順）

```
SELECT * FROM 'data/products.csv' ORDER BY price DESC;
```

-- 昇順（小さい順） - デフォルト

```
SELECT * FROM 'data/products.csv' ORDER BY price ASC;
```

```
SELECT * FROM 'data/products.csv' ORDER BY price; -- ASCは省略可
```

3. ポイント

DESC : 降順 (Descending) - 大→小、新→古、Z→A

ASC : 昇順 (Ascending) - 小→大、古→新、A→Z

何も指定しなければ昇順 (ASC)

複数の列で並び替えも可能

4. 確認ポイント

並び替え結果が視覚的にすぐ確認できる

列ヘッダーをクリックして簡易ソートもできる場合がある

でも、SQLで明示的に書く方が確実！

● 演習パート

演習1：商品を価格が高い順に並び替え

```
SELECT * FROM 'data/products.csv'  
ORDER BY price DESC;
```

結果を確認：

一番高い商品は？（ノートパソコン：89,999円）

一番安い商品は？（ワイヤレスマウス：2,999円）

演習2：売上を日付が新しい順に並び替え

```
SELECT * FROM 'data/sales.csv'  
ORDER BY order_date DESC;
```



確認ポイント：

最新の売上日は？（2024-01-29）

最古の売上日は？（2024-01-15）

演習3：数量5個以上のデータを、数量が多い順に表示

```
SELECT * FROM 'data/sales.csv'  
WHERE quantity >= 5  
ORDER BY quantity DESC;
```

これは複合技！

WHERE句で5個以上に絞り込み

ORDER BY句で数量の多い順に並び替え

● 応用練習

1. 複数列での並び替え

```
-- カテゴリー別、その中で価格の高い順  
SELECT product_name, category, price  
FROM 'data/products.csv'  
ORDER BY category, price DESC;
```

結果の見方：

まずカテゴリーでグループ化される

各カテゴリー内で価格の高い順

2. 列番号を使った並び替え

```
-- 3番目の列 (price) で並び替え  
SELECT product_id, product_name, price  
FROM 'data/products.csv'  
ORDER BY 3 DESC;
```



列番号はSELECTに書いた順番！

3. 計算結果で並び替え

```
-- 売上金額（仮想的な計算）で並び替え
SELECT
    product_id,
    quantity,
    quantity * 10000 AS 売上金額
FROM 'data/sales.csv'
ORDER BY 売上金額 DESC
LIMIT 5;
```

● 実践的な使い方

ケース1：売れ筋商品ランキング

```
-- 販売数量の多い商品TOP3
SELECT
    product_id AS 商品ID,
    SUM(quantity) AS 総販売数
FROM 'data/sales.csv'
GROUP BY product_id
ORDER BY 総販売数 DESC
LIMIT 3;
```

(GROUP BYとSUMは今後学習しますが、先取り！)

ケース2：最近の購入履歴

```
-- 特定顧客の最近の購入履歴
SELECT
    order_date AS 購入日,
    product_id AS 商品,
    quantity AS 数量
FROM 'data/sales.csv'
WHERE customer_id = 'C001'
ORDER BY order_date DESC;
```

● 効率的な確認方法

並び替えの効果を視覚的に確認

1. まず並び替えなしで実行

```
SELECT product_name, price FROM 'data/products.csv';
```

2. 次に価格順で実行

```
SELECT product_name, price FROM 'data/products.csv'  
ORDER BY price DESC;
```

NULL値の扱い

```
-- NULL値は最後に来る（DuckDBの場合）  
SELECT * FROM 'data/customers.csv'  
ORDER BY registration_date DESC;
```


● よく使う並び替えパターン

1. ランキング作成

```
-- TOP 5を取得  
SELECT * FROM テーブル  
ORDER BY 数値列 DESC  
LIMIT 5;
```

2. 時系列分析

```
-- 古い順から新しい順へ
```

3. アルファベット順

```
-- 名前順（あいうえお順）  
SELECT * FROM テーブル  
ORDER BY 名前列;
```

● 本日のまとめ

今日学んだこと：

- ✓ ORDER BY で結果を並び替える
- ✓ DESC で降順（大きい順）
- ✓ ASC で昇順（小さい順）※省略可能
- ✓ 複数列での並び替えも可能
- ✓ WHERE句と組み合わせて使える

よく使うパターン

-- 基本形

ORDER BY 列名 **DESC**

-- 複数列

ORDER BY 列1, 列2 **DESC**

-- WHERE句との組み合わせ

WHERE 条件

ORDER BY 列名 **DESC**

LIMIT 10

● 次回予告

第5回では、COUNT、SUM、AVGを使って、データ全体の集計を行います。

「全部で何件？」「合計はいくら？」「平均は？」といった質問に答えられるようになります！

● 追加演習問題

問題1：基本的な並び替え

以下の要件に従って並び替えてください：

- 1. customers.csvを顧客名のあいうえお順（昇順）で表示
- あなたの答えをここに書いてください

- 2. products.csvを価格の安い順で表示
- あなたの答えをここに書いてください

- 3. sales.csvを数量の多い順で表示（上位5件のみ）
- あなたの答えをここに書いてください

問題2：複数列での並び替え

優先順位を考えて並び替え：

-- 1. sales.csvを顧客ID順、その中で日付の新しい順

SELECT * FROM 'data/sales.csv'

ORDER BY ____, ____ DESC;

-- 2. products.csvをカテゴリ順、その中で価格の高い順

-- あなたの答えをここに書いてください

問題3：条件と組み合わせた並び替え

WHERE句とORDER BYの組み合わせ：

- 1. 電子機器カテゴリの商品を価格の高い順で表示
- あなたの答えをここに書いてください

- 2. C001の購入履歴を日付の古い順で表示
- あなたの答えをここに書いてください

- 3. 5個以上の購入を、購入日の新しい順で上位3件表示
- あなたの答えをここに書いてください

問題4：実務シナリオ

ビジネスで使える並び替え：

- 1. 売れ筋商品ランキング（数量の多い商品TOP3）
 - ヒント：GROUP BYを先取りして使ってみましょう
- ```
SELECT product_id, SUM(quantity) as total
FROM 'data/sales.csv'
GROUP BY product_id
ORDER BY ____ ____
LIMIT ____;
```
- 2. 最近の取引履歴（直近5件）
  - あなたの答えをここに書いてください
- 
- 3. 高額商品カタログ（3万円以上を高い順に）
  - あなたの答えをここに書いてください

## チャレンジ問題

- sales.csvで以下の分析をしてください：
- 「各顧客の最初の購入」を見つける
- ヒント：顧客ごとに日付の一番古いものを探す



## 実践問題：レポート作成

- 月次売上レポート用のクエリを作成：
- 1. 今月の売上を新しい順で表示（全項目）
- 2. 購入数量が多い順に顧客IDを表示（重複あり）
- 3. 商品IDごとの売上を日付順に表示

# デバッグ練習

以下のクエリの問題点を見つけて修正：

```
-- エラー1：ORDER BYの位置
SELECT * FROM 'data/products.csv'
ORDER BY price DESC
WHERE category = '電子機器';
```

```
-- エラー2：存在しない列での並び替え
SELECT product_name, price
FROM 'data/products.csv'
ORDER BY product_id;
```

```
-- エラー3：DESCの位置
SELECT * FROM 'data/sales.csv'
ORDER BY DESC quantity;
```



## 応用：並び替えの活用

```
-- 以下を試してみましょう：
-- 1. 列番号での並び替え（SELECT句の3番目の列で降順）
SELECT customer_id, product_id, quantity
FROM 'data/sales.csv'
ORDER BY 3 DESC;

-- 2. 計算結果での並び替え（仮想的な売上金額）
SELECT *, quantity * 10000 as amount
FROM 'data/sales.csv'
ORDER BY amount DESC;
```

# ● FAQ

---

**Q: DESCとASCを間違えやすい**

A: DESC = Descending（下る） = 大→小、ASC = Ascending（上る） = 小→大

**Q: 日本語（ひらがな、カタカナ）の並び順は？**

A: 文字コード順になります。ひらがな→カタカナ→漢字の順が一般的。

**Q: ORDER BYは必ず最後に書く？**

A: LIMITがある場合は、ORDER BY → LIMITの順番です。