

# 第6回：グループで集計する - GROUP BY

---

「〇〇ごと」の合計や件数を計算できるようになる

# ● 本日のゴール

---

- ✓ 「〇〇ごと」の合計や件数を計算できるようになる

# ● 座学パート

---

## 1. GROUP BYとは？

データをグループに分けて、各グループごとに集計する機能。

- 「顧客ごとの購入回数」
- 「商品ごとの売上数量」
- 「日付ごとの売上件数」

## 2. 基本構文

**SELECT**

グループ化する列,  
集計関数

**FROM** テーブル

**GROUP BY** グループ化する列;

## 3. 重要なルール

- SELECTに書く列は、GROUP BYに含まれるか、集計関数である必要がある
- GROUP BYと集計関数（COUNT、SUM、AVGなど）はセットで使う
- 複数の列でグループ化も可能

## 4. イメージで理解

元データ：

customer_id	quantity
C001	5
C001	10
C002	2
C002	1

GROUP BY customer\_id の結果：

customer_id	SUM(quantity)
C001	15
C002	3

# ● 演習パート

---

## 演習1：顧客IDごとの購入回数を数える

```
SELECT
    customer_id,
    COUNT(*) AS 購入回数
FROM 'data/sales.csv'
GROUP BY customer_id;
```

結果を確認：

- 誰が一番多く購入している？

## 並び替えを追加

```
SELECT
    customer_id,
    COUNT(*) AS 購入回数
FROM 'data/sales.csv'
GROUP BY customer_id
ORDER BY 購入回数 DESC;
```



## 演習2：商品IDごとの売上数量合計

```
SELECT
    product_id,
    SUM(quantity) AS 総売上数量
FROM 'data/sales.csv'
GROUP BY product_id;
```

どの商品が一番売れている？

## 演習3：【応用】 売上数量TOP3の商品

```
SELECT
    product_id AS 商品ID,
    SUM(quantity) AS 総売上数量
FROM 'data/sales.csv'
GROUP BY product_id
ORDER BY 総売上数量 DESC
LIMIT 3;
```

これは複合技！

1. GROUP BYで商品ごとに集計
2. ORDER BYで多い順に並び替え
3. LIMITで上位3つだけ表示

# ● 応用練習

---

## 1. 複数の集計を同時に

```
SELECT
    customer_id AS 顧客ID,
    COUNT(*) AS 購入回数,
    SUM(quantity) AS 総購入数,
    AVG(quantity) AS 平均購入数
FROM 'data/sales.csv'
GROUP BY customer_id;
```

## 2. 条件付き GROUP BY

-- 5個以上購入したことがある顧客の統計

**SELECT**

customer\_id,

COUNT(\*) **AS** 大量購入回数,

SUM(quantity) **AS** 総購入数

**FROM** 'data/sales.csv'

**WHERE** quantity >= 5

**GROUP BY** customer\_id;

## 3. 日付でグループ化

-- 日付ごとの売上統計

**SELECT**

order\_date **AS** 売上日,

**COUNT**(\*) **AS** 取引件数,

**SUM**(quantity) **AS** 販売数量

**FROM** 'data/sales.csv'

**GROUP BY** order\_date

**ORDER BY** order\_date;

# ● 実践的な使い方

---

## ケース1：顧客分析レポート

-- 優良顧客の特定（購入回数3回以上）

**SELECT**

customer\_id **AS** 顧客ID,

COUNT(\*) **AS** 購入回数,

SUM(quantity) **AS** 総購入数

**FROM** 'data/sales.csv'

**GROUP BY** customer\_id

**HAVING** COUNT(\*) **>=** 3

**ORDER BY** 総購入数 **DESC**;

## ケース2：商品カテゴリ分析

```
-- カテゴリごとの商品統計
SELECT
    category AS カテゴリ,
    COUNT(*) AS 商品数,
    AVG(price) AS 平均価格,
    MAX(price) AS 最高価格
FROM 'data/products.csv'
GROUP BY category;
```

# ● GROUP BYを理解する

---

## ステップバイステップで確認

### 1. まず元データを確認

```
SELECT customer_id, quantity  
FROM 'data/sales.csv'  
ORDER BY customer_id;
```




## 2. GROUP BYで集計

```
SELECT
    customer_id,
    SUM(quantity) AS 合計
FROM 'data/sales.csv'
GROUP BY customer_id;
```

→ 元データと集計結果を並べて比較できる！

## よくあるエラーと対処法

 エラーになる例：

```
```sql -- product_idがGROUP BYにない！ SELECT customer_id, product_id, --  
これがエラーの原因 COUNT(*) FROM 'data/sales.csv' GROUP BY customer_id;  
```
```



正しい例：

```
```sql SELECT customer_id, COUNT(*) AS 購入回数 FROM 'data/sales.csv'  
GROUP BY customer_id; ```
```

# ● GROUP BYの活用パターン

---

## 1. ランキング作成

```
-- 売れ筋商品ランキング
SELECT
    product_id,
    SUM(quantity) AS 売上数
FROM 'data/sales.csv'
GROUP BY product_id
ORDER BY 売上数 DESC;
```

## 2. 期間集計

-- 月ごとの集計（仮想的な例）

**SELECT**

SUBSTR(order\_date, 1, 7) **AS** 年月,

COUNT(\*) **AS** 件数

**FROM** 'data/sales.csv'

**GROUP BY** SUBSTR(order\_date, 1, 7);






### 3. 複数列でのグループ化

```
-- 顧客×商品の組み合わせ
SELECT
    customer_id,
    product_id,
    SUM(quantity) AS 購入数
FROM 'data/sales.csv'
GROUP BY customer_id, product_id;
```

# ● 本日のまとめ

---

今日学んだこと：

-  GROUP BY でデータをグループ化
-  各グループごとに集計関数を適用
-  HAVING で集計結果をフィルタリング
-  WHEREとHAVINGの使い分けを理解
-  複数列でのグループ化も可能

# よく使うパターン

-- 基本形

**SELECT**

グループ列,

**COUNT**(\*),

**SUM**(数値列),

**AVG**(数値列)

**FROM** テーブル

**GROUP BY** グループ列;

-- ランキング

**SELECT**

グループ列,

集計関数 **AS** 集計値

**FROM** テーブル

**GROUP BY** グループ列

**ORDER BY** 集計値 **DESC**

**LIMIT** 10;



# ● 次回予告

---

第7回では、INNER JOINを使って複数のテーブルを結合する方法を学びます。  
IDだけでなく、商品名や顧客名を表示できるようになります！

# ● 追加演習問題

---

## 問題1：基本的なGROUP BY

以下のグループ化集計を実行：

- 1. 商品IDごとの販売回数
- あなたの答えをここに書いてください
  
- 2. 日付ごとの売上件数と合計数量
- あなたの答えをここに書いてください
  
- 3. カテゴリごとの商品数と最高価格
- あなたの答えをここに書いてください

## 問題2：複数の集計関数

GROUP BYと様々な集計関数の組み合わせ：

```
-- 顧客ごとの購買行動分析
SELECT
  customer_id,
  ____ AS 購入回数,
  ____ AS 総購入数,
  ____ AS 平均購入数,
  ____ AS 最大購入数,
  ____ AS 最小購入数
FROM 'data/sales.csv'
GROUP BY ____;
```

## 問題3：HAVING句の活用

集計結果に条件を付ける：

-- 1. 2回以上売れた商品のリスト

```
SELECT product_id, COUNT(*) as 販売回数  
FROM 'data/sales.csv'  
GROUP BY product_id  
HAVING ____ >= ____;
```

-- 2. 合計購入数が15個以上の顧客

-- あなたの答えをここに書いてください

-- 3. 平均価格が3万円以上のカテゴリ

-- 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

ビジネスで使えるグループ化集計：

- 1. ABC分析（売れ筋商品の特定）
- 商品ごとの売上数量を集計し、上位20%を特定
- あなたの答えをここに書いてください

## 問題4 (続き)

```
-- 2. 顧客セグメント分析
-- 購入金額帯別の顧客数 (仮に単価1万円として)
SELECT
    CASE
        WHEN SUM(quantity) * 10000 < 50000 THEN 'ライト'
        WHEN SUM(quantity) * 10000 < 200000 THEN 'ミドル'
        ELSE 'ヘビー'
    END AS 顧客層,
    COUNT(*) AS 顧客数
FROM 'data/sales.csv'
GROUP BY 顧客層;
```

## チャレンジ問題

- 複数列でのGROUP BY :
- 顧客×商品の組み合わせごとの購入パターン分析
- (同じ顧客が同じ商品を何回、合計何個購入したか)



## 実践問題：月次レポート作成

- 以下のレポートを作成：
- 1. 日別売上サマリー
  - - 各日の取引件数
  - - 各日のユニーク顧客数
  - - 各日の総販売数量
- 2. 商品別パフォーマンス
  - - 各商品の販売回数
  - - 各商品の総販売数量
  - - 購入した顧客数（DISTINCT使用）



## デバッグ練習

以下のクエリの問題点を修正：

```
-- エラー1：GROUP BYに含まれない列
SELECT customer_id, product_id, COUNT(*)
FROM 'data/sales.csv'
GROUP BY customer_id;
```

```
-- エラー2：集計関数なしのGROUP BY
SELECT customer_id
FROM 'data/sales.csv'
GROUP BY customer_id;
```

```
-- エラー3：HAVINGとWHEREの混同
SELECT customer_id, COUNT(*) as cnt
FROM 'data/sales.csv'
```



## 応用：GROUP BYの組み合わせ

```
-- 1. サブクエリを使った分析（先取り）
-- 平均購入数量を上回る顧客のリスト
WITH avg_purchase AS (
    SELECT AVG(quantity) as avg_qty FROM 'data/sales.csv'
)
SELECT customer_id, AVG(quantity) as 顧客平均
FROM 'data/sales.csv'
GROUP BY customer_id
HAVING AVG(quantity) > (SELECT avg_qty FROM avg_purchase);
```



## 応用（続き）

- 2. 複雑な集計
- 顧客ごとの「最頻値商品」を見つける
- ヒント：顧客ごと、商品ごとにカウントして、最大値を探す

## 総合演習：売上分析ダッシュボード

- 経営者向けのダッシュボードに必要な数値を全て集計：
- 1. 顧客別売上ランキングTOP3
- 2. 商品別売上ランキングTOP3
- 3. 日別売上推移（件数と数量）
- 4. カテゴリ別の売上比率
- それぞれ個別のクエリで作成してください

## 上級問題：クロス集計

```
-- 顧客×日付のクロス集計表を作成  
-- (どの顧客がどの日に何個購入したか)
```

```
SELECT
```

```
  customer_id,
```

```
  SUM(CASE WHEN order_date = '2024-01-15' THEN quantity ELSE 0 END) AS "1月15日",
```

```
  SUM(CASE WHEN order_date = '2024-01-16' THEN quantity ELSE 0 END) AS "1月16日",
```

```
  SUM(CASE WHEN order_date = '2024-01-17' THEN quantity ELSE 0 END) AS "1月17日"
```

```
-- 続きを書いてみましょう
```

```
FROM 'data/sales.csv'
```

```
GROUP BY customer_id;
```

# ● FAQ

---

**Q: WHEREとHAVINGの違いは？**

A: WHERE = GROUP BY前の絞り込み、HAVING = GROUP BY後の絞り込み

**Q: GROUP BYの列は必ずSELECTに書く？**

A: 書かなくても動きますが、結果が分かりにくくなるので書くことを推奨。

**Q: 集計結果が0の場合も表示したい**

A: LEFT JOINなどを使う必要があります（第8回で学習）。