

SQL クイックリファレンス

SELECT、JOIN、サブクエリ、インデックス、トランザクション

SELECT

```
SELECT * FROM users;
SELECT name, email FROM users;
SELECT DISTINCT city FROM users;
SELECT name AS full_name FROM users;
```

WHERE

比較演算子

= <> (!=) 等しい/等しくない
< > <= >= 比較演算子
AND OR NOT 論理演算子
IS NULL / IS NOT NULL NULL チェック

パターンマッチング

```
SELECT * FROM users WHERE name LIKE 'A%';
-- % = 任意の文字列、_ = 任意の1文字
SELECT * FROM users WHERE age IN (20, 25, 30);
SELECT * FROM users WHERE age BETWEEN 18 AND 30;
```

JOIN

JOINの種類

INNER JOIN 両方のテーブルでマッチする行
LEFT JOIN 左側のすべての行 + 右側のマッチした行
RIGHT JOIN 右側のすべての行 + 左側のマッチした行
FULL OUTER JOIN 両方のテーブルのすべての行
CROSS JOIN 両方のテーブルのデカルト積

JOINの構文

```
SELECT u.name, o.total
FROM users u
INNER JOIN orders o ON u.id = o.user_id;

SELECT u.name, o.total
FROM users u
LEFT JOIN orders o ON u.id = o.user_id;
```

INSERT / UPDATE / DELETE

挿入

```
INSERT INTO users (name, email)
VALUES ('Alice', 'alice@example.com');

INSERT INTO users (name, email) VALUES
('Bob', 'bob@ex.com'),
('Carol', 'carol@ex.com');
```

更新

```
UPDATE users SET email = 'new@ex.com'
WHERE id = 1;
```

削除

```
DELETE FROM users WHERE id = 1;
DELETE FROM users; -- すべての行を削除
```

CREATE TABLE

構文

```
CREATE TABLE users (
  id INTEGER PRIMARY KEY AUTOINCREMENT,
  name TEXT NOT NULL,
  email TEXT UNIQUE,
  age INTEGER DEFAULT 0,
  score REAL
);
```

よく使うデータ型

INTEGER 整数
REAL 浮動小数点数
TEXT 文字列・テキストデータ
BLOB バイナリデータ
BOOLEAN TRUE / FALSE (0/1として格納)
DATE / DATETIME 日付と日時の値

制約

PRIMARY KEY 行の一意識別子
NOT NULL 値が必須
UNIQUE 重複する値を禁止
DEFAULT val 省略時のデフォルト値
CHECK (expz) カスタムバリデーションルール
FOREIGN KEY 他のテーブルへの参照

集計関数

COUNT (*) 行数
COUNT (col) 列の NULL でない値の数
SUM (col) 数値列の合計
AVG (col) 数値列の平均
MIN (col) 最小値
MAX (col) 最大値

使用例

```
SELECT COUNT(*) AS total,
       AVG(age) AS avg_age,
       MAX(score) AS top_score
FROM users;
```

GROUP BY / HAVING

```
SELECT city, COUNT(*) AS num_users
FROM users
GROUP BY city;

SELECT city, AVG(age) AS avg_age
FROM users
GROUP BY city
HAVING AVG(age) > 25;
```

WHERE はグループ化前に行をフィルタリング。HAVING は集計後にグループをフィルタリング

ORDER BY / LIMIT

```
SELECT * FROM users ORDER BY name ASC;
SELECT * FROM users ORDER BY age DESC;
SELECT * FROM users
ORDER BY city, name
LIMIT 10 OFFSET 20; -- 20件スキップして10件取得
```

サブクエリ

WHERE 句内

```
SELECT name FROM users
WHERE id IN (
  SELECT user_id FROM orders
  WHERE total > 100
);
```

派生テーブルとして

```
SELECT city, avg_age FROM (
  SELECT city, AVG(age) AS avg_age
  FROM users GROUP BY city
) WHERE avg_age > 30;
```

インデックス

```
CREATE INDEX idx_name ON users(name);
CREATE UNIQUE INDEX idx_email
ON users(email);
DROP INDEX idx_name;
```

インデックスを付けるタイミング

WHERE 句の列 フィルタリングを高速化
JOIN ON の列 JOINの検索を高速化
ORDER BY の列 ソートを高速化
カーディナリティの高い列 一意の値が多い列が最も効果的