MYSQL クイックリファレンス

データベース、テーブル、CRUD、ジョイン、インデックス、ユーザー

接続

コマンドライン

```
mysql -u root -p
mysql -u user -p -h 127.0.0.1 -P 3306
mysql -u user -p mydb < dump.sql
```

接続文字列

```
mysql://user:password@host:3306/dbname
mysql -u user -p -e "SELECT VERSION();"
```

ステータスコマンド

```
STATUS;
SHOW VARIABLES LIKE 'port';
SHOW PROCESSLIST;
```

データベースとテーブル

データベース操作

```
CREATE DATABASE mydb CHARACTER SET utf8mb4;
SHOW DATABASES;
USE mydb;
DROP DATABASE mydb;
```

テーブル操作

```
CREATE TABLE users (
  id INT AUTO_INCREMENT PRIMARY KEY,
  name VARCHAR(100) NOT NULL,
  email VARCHAR(255) UNIQUE,
  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

テーブル情報

```
SHOW TABLES;
DESCRIBE users;
SHOW CREATE TABLE users;
```

テーブル変更

```
ALTER TABLE users ADD COLUMN age INT;
ALTER TABLE users MODIFY COLUMN name VARCHAR(200);
ALTER TABLE users DROP COLUMN age;
ALTER TABLE users RENAME TO customers;
```

データ型

数値型

`INT`	4 バイト整数（-20 億〜20 億）
`BIGINT`	8 バイト整数
`DECIMAL(p,s)`	固定精度数値（例：DECIMAL(10,2)）
`FLOAT / DOUBLE`	近似浮動小数点数
`BOOLEAN`	TINYINT(1)の別名

文字列型

`VARCHAR(n)`	最大 n 文字の可変長文字列
`TEXT`	最大 65KB のテキスト
`MEDIUMTEXT`	最大 16MB のテキスト
`CHAR(n)`	固定長文字列（パディングあり）
`ENUM('a','b')`	定義済みセットからの 1 つの値

日付と時刻型

`DATE`	YYYY-MM-DD
`DATETIME`	YYYY-MM-DD HH:MM:SS
`TIMESTAMP`	UTC 保存の日時
`TIME`	HH:MM:SS
`JSON`	ネイティブ JSON ドキュメント型

CRUD

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');
```

選択

```
SELECT * FROM users WHERE id = 1;
SELECT name, email FROM users
ORDER BY name LIMIT 10 OFFSET 20;
```

更新

```
UPDATE users SET email = 'new@ex.com'
WHERE id = 1;
UPDATE users SET active = 0
WHERE last_login < '2025-01-01';
```

削除

```
DELETE FROM users WHERE id = 1;
TRUNCATE TABLE users;  -- 高速、AUTO_INCREMENTをリセット
```

アップサート

```
INSERT INTO users (id, name, email)
VALUES (1, 'Alice', 'a@ex.com')
ON DUPLICATE KEY UPDATE
name = VALUES(name), email = VALUES(email);
```

ジョイン

ジョインの種類

`INNER JOIN`	両テーブルで一致する行
`LEFT JOIN`	左テーブルの全行＋右テーブルの一致行
`RIGHT JOIN`	右テーブルの全行＋左テーブルの一致行
`CROSS JOIN`	直積
`SELF JOIN`	テーブル自身とのジョイン

ジョインの例

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

SELECT e.name, m.name AS manager
FROM employees e
LEFT JOIN employees m ON e.manager_id = m.id;
```

インデックス

作成と削除

```
CREATE INDEX idx_name ON users(name);
CREATE UNIQUE INDEX idx_email ON users(email);
CREATE INDEX idx_composite
  ON orders(user_id, created_at);
DROP INDEX idx_name ON users;
```

フルテキストインデックス

```
ALTER TABLE posts ADD FULLTEXT(title, body);
SELECT * FROM posts
WHERE MATCH(title, body) AGAINST('mysql');
```

インデックス情報

```
SHOW INDEX FROM users;
EXPLAIN SELECT * FROM users WHERE name = 'Alice';
```

関数

文字列関数

`CONCAT(a, b)`	文字列の連結
`SUBSTRING(s, pos, len)`	部分文字列の抽出
`UPPER(s) / LOWER(s)`	大文字/小文字変換
`TRIM(s)`	前後の空白を除去
`LENGTH(s)`	バイト単位の文字列長
`REPLACE(s, from, to)`	出現箇所を置換

日付関数

`NOW()`	現在の日時
`CURDATE()`	現在の日付
`DATE_ADD(d, INTERVAL n DAY)`	日付に間隔を加算
`DATEDIFF(d1, d2)`	日数の差
`DATE_FORMAT(d, fmt)`	日付のフォーマット（例：'%Y-%m-%d'）

集計関数

```
SELECT COUNT(*), AVG(price), SUM(qty),
       MIN(price), MAX(price)
FROM products
GROUP BY category HAVING COUNT(*) > 5;
```

ユーザーと権限

ユーザー管理

```
CREATE USER 'app'@'localhost'
IDENTIFIED BY 'secret';
ALTER USER 'app'@'localhost'
IDENTIFIED BY 'newsecret';
DROP USER 'app'@'localhost';
```

権限付与

```
GRANT ALL ON mydb.* TO 'app'@'localhost';
GRANT SELECT, INSERT ON mydb.users
TO 'reader'@'%';
REVOKE INSERT ON mydb.users
FROM 'reader'@'%';
FLUSH PRIVILEGES;
```

権限の確認

```
SHOW GRANTS FOR 'app'@'localhost';
SELECT user, host FROM mysql.user;
```

バックアップとリストア

mysqldump

```
mysqldump -u root -p mydb > backup.sql
mysqldump -u root -p --all-databases > all.sql
mysqldump -u root -p mydb users > users.sql
```

リストア

```
mysql -u root -p mydb < backup.sql
mysql -u root -p -e "SOURCE /path/backup.sql"
```

バイナリログ

```
SHOW BINARY LOGS;
SHOW BINLOG EVENTS IN 'binlog.000001';
mysqlbinlog binlog.000001 | mysql -u root -p
```

よくあるパターン

ページネーション

```
SELECT * FROM users
ORDER BY id LIMIT 20 OFFSET 40;  -- 3ページ目
```

トランザクション

```
START TRANSACTION;
UPDATE accounts SET balance = balance - 100
WHERE id = 1;
UPDATE accounts SET balance = balance + 100
WHERE id = 2;
COMMIT;  -- または ROLLBACK;
```

条件付き挿入

```
INSERT IGNORE INTO users (email, name)
VALUES ('a@ex.com', 'Alice');
-- メールが既に存在する場合はスキップ (UNIQUE)
```

変数とプリペアドステートメント

```
SET @name = 'Alice';
PREPARE stmt FROM
  'SELECT * FROM users WHERE name = ?';
EXECUTE stmt USING @name;
DEALLOCATE PREPARE stmt;
```