MySQL 快速参考

数据库、表、CRUD、连接、索引、用户管理

连接

命令行

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

连接字符串

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

状态命令

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

数据库与表

数据库操作

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

表操作

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

表信息

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

修改表

```
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;
```

数据类型

数值类型

INT	4 字节整数（-20 亿到 20 亿）
BIGINT	8 字节整数
DECIMAL(p,s)	精确数值（如 DECIMAL(10,2)）
FLOAT / DOUBLE	近似浮点数
BOOLEAN	TINYINT(1) 的别名

字符串类型

VARCHAR(n)	最长 n 字符的可变长字符串
TEXT	最多 65 KB 文本
MEDIUMTEXT	最多 16 MB 文本
CHAR(n)	固定长度字符串，不足则补空格
ENUM('a','b')	从定义集合中取一个值

日期与时间

DATE	YYYY-MM-DD
DATETIME	YYYY-MM-DD HH:MM:SS
TIMESTAMP	以 UTC 存储的日期时间
TIME	HH:MM:SS
JSON	原生 JSON 文档类型

CRUD

插入

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

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

查询

```
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';
```

删除

```
DELETE FROM users WHERE id = 1;
TRUNCATE TABLE users;  -- fast, resets AUTO_INCREMENT
```

Upsert

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

连接（JOIN）

JOIN 类型

INNER JOIN	两表中均匹配的行
LEFT JOIN	左表所有行 + 右表匹配行
RIGHT JOIN	右表所有行 + 左表匹配行
CROSS JOIN	笛卡尔积
SELF JOIN	表与自身连接

JOIN 示例

```
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;
```

索引

创建与删除

```
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;
```

全文索引

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

索引信息

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

函数

字符串函数

CONCAT(a, b)	拼接字符串
SUBSTRING(s, pos, len)	提取子字符串
UPPER(s) / LOWER(s)	大小写转换
TRIM(s)	去除首尾空格
LENGTH(s)	字节长度
REPLACE(s, from, to)	替换出现的字符串

日期函数

NOW()	当前日期时间
CURDATE()	当前日期
DATE_ADD(d, INTERVAL n DAY)	日期加上间隔
DATEDIFF(d1, d2)	相差天数
DATE_FORMAT(d, fmt)	格式化日期（如 '%Y-%m-%d'）

聚合函数

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

用户与权限

用户管理

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

授权

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

查看权限

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

备份与恢复

mysqldump

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

恢复

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

MySQL 快速参考

二进制日志

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

常见模式

分页

```
SELECT * FROM users
ORDER BY id LIMIT 20 OFFSET 40;  -- page 3
```

事务

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

条件插入

```
INSERT IGNORE INTO users (email, name)
VALUES ('a@ex.com', 'Alice');
-- skips if email already exists (UNIQUE)
```

变量与预处理语句

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