

MySQL

データベースマネージメントシステム(DBMS)

データベース

- データを格納するための構造化された集合
- データを操作するための規則
- データを保護するための規則

データ構造

データ型、インデックス、トリガ、ストアードプロシージャ、ストアードファンクション

データ構造

- 列
- 行
- 表
- 主キー(primary key)

データ型

- 整数

データ型	バイト数(bytes)	範囲	範囲
TINYINT	1	-128	-127
SMALLINT	2	-32768	32767
MEDIUMINT	3	-8388608	8388607
INT	4	-2147483648	2147483647
BIGINT	8		

- 浮動小数点数

- decimal: decimal(5,2) 5桁小数2桁

- 文字列

- char 固定長文字列
- varchar 可変長文字列 + 1 (null '\0')
- text 大量

- enum(enum)

- gender enum('男','女','その他')

- 日付

- date -0-0-0
- datetime -0-0-0-0-0
- timestamp -0-0-0-0-0

1

□ □ □ □ □ □ □ □

NOT NULL	不可以为空
PRIMARY KEY	主键
UNIQUE KEY	唯一键
DEFAULT	默认值
FOREIGN KEY	外键(参照完整性)
AUTO_INCREMENT	自动递增

```
create table student(
    id int unsigned PRIMARY KEY AUTO_INCREMENT COMMENT '学号',
    name varchar(10) NOT NULL COMMENT '姓名',
    age int unsigned COMMENT '年龄',
    class int unsigned COMMENT '班级',
    gender enum("男","女") COMMENT '性别'
    status char(1) DEFAULT '1' COMMENT '状态'
);COMMENT '学生表'

-- 插入数据
insert into student (name,age,class,gender) values('张三',18,3,'女') # status为1
insert into student (name,age,class,gender) values('李四',17,2,'男') # id为10001
```

1

==□□□□->□->□□==

	□□	□□
mysql -uroot -p	□□□□□	
exit/quit/ctrl+d	□□□□□	
select version();	□□□□□□	
select now();	□□□□	

10 of 10

show databases;	显示所有数据库
select database();	显示当前数据库
create database <u>数据库名</u> charset = utf8;	创建数据库

use 云数据库;	云数据库
drop database 云数据库;	云数据库

```
-- 云数据库
show databases;
-- 云数据库
select database();
-- 云数据库
create database itcast charset=utf8;
-- 云数据库
use itcast;
-- 云数据库
drop database itcast
```

云数据库表操作

命令	结果
show tables;	云数据库表信息
desc 命令;	云数据库
show create tabel 命令;	云数据库
create table 命令;	云数据库
comment '命令'	云数据库

```
-- 云数据库
create table xxx(
    id int unsigned primary key auto_increment not null,
    name varchar(20),
    age int unsigned default 0,
    high decimal(5,2),
    gender enum("男","女"),
    cls_id int unsigned
);
```

云数据库表操作

命令	结果
alter table 命令 add 命令 命令;	云数据库
alter table 命令 change 命令 命令 命令;	云数据库
alter table 命令 modify 命令 命令;	云数据库
alter table 命令 drop 命令;	云数据库
drop table 命令;	云数据库

```
-- ①
alter table student modify name varchar(20);
-- ②
alter table student add gender enum("男","女");
-- ③
alter table student drop gender;
-- ④
alter table student modify name COMMENT '姓名'
```

⑤插入语句

⑥	⑦
insert into ⑧ values(...);	⑨
insert into ⑩(⑪,...)values();	⑫
insert into ⑬ values(...),(...);	⑭
insert into ⑮(⑯,...) values(⑯1...)(⑯1...);	⑯

```
-- ⑰
insert into student values(0,"王五",18,166.66,"男",2);
# ⑱
insert into student (name,age,height,gender,class) values("王五",18,166.66,"男",2); # ⑲
⑳
insert into student values(0,"王五",18,166.66,"男",2),(0,"李四",17,154.43,"女",1); # ⑳
⑳
```

⑳更新语句

⑳	⑳
select *from ⑳;	⑳
select ⑳1,⑳2... from ⑳;	⑳
update ⑳ set ⑳1 = ⑳1,⑳2 = ⑳2 ...where ⑳;	⑳
select distinct from ⑳;	⑳

```
-- ㉑
update students set gender = '男'; # ㉒
update students set gender = '女' where id = 2; # ㉓

-- ㉔
select * from students; # ㉔
㉕
select * from students where id = 2; # ㉖
select name,age from students; # ㉗
select name as '姓名',age as '年龄' from students; # ㉘
```

```
select age as '年龄', name as '姓名', from students; # 显示所有学生
select distinct gender from students; # 显示性别
```

删除语句

年龄	姓名
delete from 学生 where 年龄;	删除语句

```
-- 删除语句
delete from students where id = 4;

-- 修改语句
alter table students add is_delete bit default 0; # 添加一个is_delete列
update students set is_delete = 1 where id = 4; # 将id为4的is_delete设为1
```

where语句

- 比较运算符
 - =
 - >
 - <
 - !=
 - <>
- 逻辑运算符
 - and
 - or
 - not

模糊查询

like

- %
- _

```
-- %“%”
select * from students where name like '%';
-- “%”%
select * from students where name like '%_';
-- __%
select * from students where name like '__';
-- __%
select * from students where name like '__%';
```

范围查询

- between and 语句：between A and B [A,B]
- in语句

```
-- 18-30岁
select name from students where age in (18,30);
```

```
-- 18と30以外の年齢
select * from students where age not in (18,30);
-- 18と30の間の年齢
select * from students where age between 18 and 30;
-- 18と30以外の年齢
select * from students where age not between 18 and 30;
```

NULL

- NULLはnull
- NULLはnot null

```
-- NULL
select * from students where height is null;
```

order BY

```
select * from 表 order by カラム名 asc|desc[,カラム名 asc|desc,...]
```

- カラム名1カラム名2
- asc
- desc
- NULL

```
-- 18と34の間の年齢
select * from students where age between 18 and 34 and gender = '男' order by height asc;
-- 18と34の間の年齢,性別
select * from students where age between 18 and 34 and gender = '男' order by height desc,age desc;
```

Aggregate

関数	説明
count(列)	件数
max(列)	最大値
min(列)	最小値
sum(列)	合計
avg(列)	平均

```
-- 件数
select count(*) from students where gender = '男';
-- 最大年齢
select max(age) from students;
-- 最小年齢
select min(height) from students;
```

```
-- 例題
select sum(age) from students;
-- 結果を丸める
select round(avg(age),2) from students;
```

group句

- **group by**: 同じ属性で集計する
- **group concat**

句	説明
group concat(句)	複数の行を1行にまとめる
having 句	条件を追加する
with roll up	階層的集計

```
-- 例題
select gender from students group by gender
-- 計算
select gender,count(*) from students group by gender;
-- 階層的集計
select group concat(name),gender group by gender;
-- 年齢平均30歳以上
select group concat(name),gender group by gender having avg(age) > 30;
-- 総合集計
select gender,count(*) from students group by gender with rollup;
```

limit句

- **limit**: 結果を制限する
sql句
- **limit** 値
- **limit**
 - **offset** 値
 - **offset** 値

```
-- 例題5件
select * from students limit 5;
-- 例題2件目1件目
select * from students limit 0,2;
-- 例題2件目2件目
select * from students limit 2,2;
-- 例題2件目4件目以降
select * from students order by age asc limit 6,2;
```

まとめ

SELECT文の構成要素と各要素の意味

確認

INNER JOIN

SQL: select * from t1 inner join t2 on t1.id = t2.id

- on 条件

```
-- INNER JOIN
select * from students inner join classes;
-- INNER JOIN
select * from students inner join classes on students.cls_id = classes.id;
-- 外连接, CROSS JOIN
select s.name,c.name from students as s inner join classes as c on s.cls_id = c.id;
```

OUTER JOIN

INNER JOIN 和 OUTER JOIN 的区别

- 左连接 left join 表 on 条件
- 右连接 right join 表 on 条件

```
-- LEFT JOIN
select * from students right join classes on students.cls_id = classes.id;
-- RIGHT JOIN
select * from students left join classes on students.cls_id = classes.id;
```

CROSS JOIN

INNER JOIN 和 CROSS JOIN 的区别

EXPLAIN

image-20241007190605047

EXPLAIN 分析

1. EXPLAIN aid
2. EXPLAIN pid t1 at title

==EXPLAIN 分析语句==

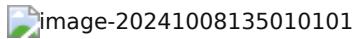
```
-- EXPLAIN
select * from areas as city inner join areas as province on city.pid = province.aid;
```

EXPLAIN

EXPLAIN 分析语句

```
-- EXPLAIN
select avg(height) from students;
-- EXPLAIN 分析语句
select * from students where height > (select avg(height) from students);
```

□□



```
-- □cate_name'□'□name□price
select name,price from goods where cate_name = '□';

-- □
## □
select brand_name from goods group by brand_name;
## □
select distinct brand_name from goods;

-- □avg□
select round(avg(price),2) from goods;

-- □cate_name□
select avg(price),cate_name from goods group by cate_name;

-- □
select cate_name,max(price),min(price),avg(price),count(*) from goods group by
cate_name;

-- □
## □
select avg(price) from goods;
## □
select * from goods where price > (select avg(price) from goods) order by price
desc;

-- □
## □
select cate_name,max(price) as max from goods group by cate_name;
## □
select * from goods inner join (select cate_name,max(price) as max from goods group
by cate_name ) as max_price on goods.cate_name = max_price.cate_name and goods.price
= max_price.max;
```

□□

- □A□B□A□B□
- □

□	□
atler table □ add foreign key(□) references □(□);	□□□
atler table □ drop foreign key □;	□□□
show create table goods;	□□□□□□□

```
-- ①
alter table goods add foreign key(cate_id) references goods_cate(id);
-- ②
## ③
show create table goods;
## ④
atler table ⑤ drop foreign key goods_ibfk_1;

-- ⑥
alter table goods{
    id int primary key auto_increment not null,
    name varchar(20),
    price decimal(5,2),
    cate_id int unsigned,
    brand_id int unsigned

    foreign key (cate_id) references goods_cate(id),
    foreign key (brand_id) references goods_brand(id)
};


```

①

- ②③④⑤⑥sql⑦
- ⑧sql⑨⑩⑪
- ⑫⑬⑭⑮

①	②
create view ③ as select ④;	⑤
show tables;	⑥
select * from v_goods_info;	⑦
drop view ⑧;	⑨

```
-- ⑩id⑪⑫⑬
select s.id,s.name,s.age,s.gender,c.name as cls_name from students as s inner join
classes as c on s.id = c.id;

-- ⑭
create view v_students as select s.id,s.name,s.age,s.gender,c.name as cls_name from
students as s inner join classes as c on s.id = c.id;

-- ⑮
select * from v_students;

-- ⑯
drop view v_students;
```

①

- SQL语句的事务控制语句
- 事务ACID
 - 原子性
 - 一致性
 - 隔离性
 - 持久性

SQL语句	功能
begin; start transaction;	开始事务
commit;	提交事务
rollback;	回滚事务

```
-- 开始
begin;
update students set age = 100 where id = 1;
-- 提交
commit;
```

二

==SQL语句的优化==

SQL语句	功能
show index from 表名;	显示索引信息
alter table 表名 add index 索引名 (列名);	添加索引
drop index 索引名 on 表名;	删除索引

```
-- 打开慢查询日志
set profiling = 1 ;
## 显示慢查询语句
select * from test_index where title = 'ha-99999';
show profiles;
## 优化
alter table test_index add index(title);
## 再次显示慢查询语句
select * from test_index where title = 'ha-99999';
show profiles;
```

三

- MySQL语句执行的优化方法
- MySQL语句的优化
 1. 索引
 - 索引的使用与创建
 2. 语句(语句重写)

- ䷂
- ䷃
- ䷄

3. ䷁(䷁)

- ䷁
- ䷁A䷁B䷁B

SQL

SQL SQL SQL

```
from pymysql import Connection
## ䷁
conn = Connection(host='localhost',port=3306,user='root',password='zxb050818')
print(conn.get_server_info())
## ䷁
cursor = conn.cursor()
## ䷁
conn.select_db('world')
## ䷁sql
find_name = input() # or 1 or
sql = "select * from students where name = '%s'" % find_name
cursor.execute(sql)
## ䷁
content = cursor.fetchall() # ䷁
```

==

- ䷁
- ䷁sql

```
params = [find_name]
sql = 'select * from students where name = %s'
cursor.execute(sql,params)
```

SQL

SQL SQL

SQL

- ䷁DDL
 - ䷁
- ䷁DML
 - ䷁
- ䷁DCL
 - ䷁
- ䷁DQL
 - ䷁

䷁

- 语句
 - 嵌入式SQL语句
 - 简单语句
 - 语句前-语句后(--语句结束语句)
 - 语句#语句
 - 语句/* 语句 */

DDL语句

- 数据库语句
 - show databases ;
 - use 数据库名 ;
 - create database 数据库名 [CHARSET UTF8] ;
 - drop database 数据库名 ;
 - select database() ;
- 表语句
 - show tables ;
 - drop table 表名 ;| drop table if exists 表名 ;
 - create table 表名(字段名 数据类型, 字段名 数据类型) ;
 - int|float|data(精度)|timestamp(精度),varchar(长度):表名

DML语句

- 插入语句 insert into 表[(列1,列2,...,列N)] values(值1,值2,...,值N) [, (值1,值2,...,值N), (值1,值2,...,值N), ..., (值1,值2,...,值N)]
- 删除语句 delete from 表 [where 条件] (通过py语句)或者where条件语句
- 更新语句 update 表 set 列 = 值 [where 条件] 语句where条件语句

DQL语句

- 选择语句 select *语句 from 表 where 条件
- 分组语句 select 语句/聚合语句(聚合函数) from 表 where 条件 group by 表
 - group by语句和select语句
 - 聚合语句
 - sum
 - avg
 - min
 - max
 - count
- 排序语句order by语句 select 语句/聚合语句(聚合函数) from 表 where 条件 group by 表 order by 表ASC|DESC语句
- 限制语句limit语句select语句limit n,m语句n + 1语句limit语句m语句

python与SQL语句

- 通过pymysql模块操作MySQL语句
- 通过pymysql模块语句 commit() 语句autocommit语句true

语句

1. 导入pymysql
2. 建立连接
3. 打开游标
4. pymysql模块语句
5. 语句操作语句

```
from pymysql import Connection
## 本地连接
conn = Connection(host='localhost',port=3306,user='root',password='zxb050818')
print(conn.get_server_info())
## 打开游标
cursor = conn.cursor()
## 选择数据库
conn.select_db('world')
## 执行sql语句
sql = 'select * from students'
cursor.execute(sql)
## 获取一条数据
content = cursor.fetchone()
## 获取所有数据
content = cursor.fetchall()
print(content)
## 关闭游标
cursor.close()
conn.close()
```

本地连接

==本地连接提交==

```
from pymysql import Connection
## 本地连接
conn = Connection(host='localhost',port=3306,user='root',password='zxb050818')
## 打开游标
cursor = conn.cursor()
## 选择数据库
conn.select_db('world')
## 执行插入语句
sql = "insert into students(name) values('张三')"
cursor.execute(sql)
## 获取所有数据
sql = 'select * from students;'
cursor.execute(sql)
content = cursor.fetchall()
for i in content:
    print(i)
## 提交
conn.commit()
## 关闭游标
cursor.close()
conn.close()
```