目录

[Mysql day03 1](#_Toc500863295)

[1 SQL 1](#_Toc500863296)

[2 DML 2](#_Toc500863297)

[2.1 insert 2](#_Toc500863298)

[2.2 update 3](#_Toc500863299)

[2.3 delete 4](#_Toc500863300)

[3 DQL - select 4](#_Toc500863301)

[4 where 子句 5](#_Toc500863302)

[5 distinct 9](#_Toc500863303)

[6 order by 子句 10](#_Toc500863304)

[7 字段别名 11](#_Toc500863305)

[8 目前,select 结构 12](#_Toc500863306)

[9 作业 12](#_Toc500863307)

Mysql day03

# SQL

结构化的查询语言,所有数据库都通过SQL来访问数据

SQL有标准语法规范,个数据库厂商都有自己的SQL扩展语法

SQL分类:

* DDL 数据定义语句,库、表管理
* DML 数据操作语句,增删改查
* DQL 数据查询语句,select

# DML

## insert

* insert into tb2 values(1,2,3);  
  向所有字段插入值
* insert into tb2(a,c) values(1,3)  
  向指定的字段插入值
* 将查询的结果数据,全部插入表中

insert into tb2 select \* from tb3;

* 将查询的结果创建成一张表  
  create table tb4   
  as  
  select \* from tb3;
* 插入特殊字符
  + 单引号用两个单引号转义  
    '' --> '
  + sql注入攻击

密码: 1' or '1'='1

查询用户:

select \* from users

where name='swfwe' and

pwd='1' or '1'='1'

* + 避免sql注入攻击,单引号替换成两个单引号

select \* from users

where name='swfwe' and

pwd='1'' or ''1''=''1'

* + 特殊字符\

用转义运算符\对斜杠进行转义,转义成一个斜杠字符

\\

批量数据插入

set names gbk;

use test;

-- 新建学生备份表

create table xuesheng\_bak(

id int primary key,

name varchar(20)

)engine=innodb charset=utf8;

-- 从xuesheng查询数据,插入bak表

insert into xuesheng\_bak

select id,name from xuesheng;

select \* from xuesheng\_bak;

-- 查询学生数据,把结果直接创建成表

create table xuesheng\_bak2

as -- 可以省略

select id,name from xuesheng;

select \* from xuesheng\_bak2;

插入特殊字符

insert into xuesheng(name)

values('I''m 张三'),('李\\四');

select \* from xuesheng;

## update

* update tb2 set a=1,b=2,c=3
* update tb2 set a=1,b=2,c=3  
  where ...
* 特殊更新方式:  
  排序后,只更新前几条数据

例如: 销量最低的5件商品,降价

update items set price=price\*0.5

order by sales\_num limit 5;

* + - order by 以指定字段排序
    - limit 指定前几行数据

update ... order by ... limit ...

-- 新添加的3个学生去2班

update xuesheng

set banji\_id=2

order by id desc -- desc 降序

limit 3;

select \* from xuesheng;

## delete

* delete from tb2
* delete from tb2 where ...
* 特殊删除方式:

排序后,删除前几条数据

例如: 销量最低的5件商品删除

delete from items

order by sales\_num

limit 5;

delete ... order by ... limit ...

-- 删除最后添加的3个学生

delete from xuesheng

order by id desc

limit 3;

select \* from xuesheng;

# DQL - select

select 字段列表 from 表 where ...

# where 子句

过滤条件

|  |  |
| --- | --- |
| = | 等值过滤 |
| <> | 不等过滤 |
| > >= < <= | 比较大小 |
| between ... and ... | >=小 and <=大 |
| in(20,15,38,30) | 取列表中指定的值 |
| is null  is not null | null值过滤 |
| like  \_ 通配单个字符  % 通配任意多个字符  \\_ 转义成普通下划线字符  \% 转义成普通百分号字符 | 字符串的模糊查询 |
| not | not between ... and ...  not in(1,2,3,4)  is not null  not like |
| and | 并且 |
| or | 或者 |

准备测试库hr

-- 下载hr\_mysql.sql脚本文件

-- 存放在d盘根目录(写路径方便)

source d:\hr\_mysql.sql

show tables;

desc employees;

select \* from employees;

select employee\_id,first\_name,manager\_id from employees;

比较过滤

-- 1. 员工id employee\_id 是120的员工

select employee\_id,first\_name,salary

from employees

where employee\_id=120;

-- 2. 主管id, manager\_id 是120的员工

select employee\_id,first\_name,

salary,manager\_id

from employees

where manager\_id=120;

-- 3. 部门id, department\_id 是90的员工

select employee\_id,first\_name,

salary,department\_id

from employees

where department\_id=90;

-- 4. 岗位id, job\_id 是 'IT\_PROG'

select employee\_id,first\_name,

salary,job\_id

from employees

where job\_id='IT\_PROG';

-- 5. 排除 50部门

select employee\_id,first\_name,

salary,department\_id

from employees

where department\_id<>50;

-- 6. 工资 salary 小于3000

select employee\_id,first\_name,salary

from employees

where salary<3000;

-- 7. 工资范围 [8000, 10000]

select employee\_id,first\_name,salary

from employees

where salary between 8000 and 10000;

-- where salary>=8000 and salary<=10000;

in过滤

-- 8. 员工id是 100,120,121 三个人

select employee\_id,first\_name,salary

from employees

where employee\_id in(100,120,121);

-- 9. 岗位代码是'IT\_PROG','SA\_MAN','HR\_REP'

select employee\_id,first\_name,

salary,job\_id

from employees

where job\_id in('IT\_PROG','SA\_MAN','HR\_REP');

null值过滤

-- 10. 没有主管的人

select employee\_id,first\_name,

salary,manager\_id

from employees

where manager\_id is null;

-- 11. 没有部门的人

select employee\_id,first\_name,

salary,department\_id

from employees

where department\_id is null;

-- 12. 所有有提成的人 commission\_pct

select employee\_id,first\_name,

salary,commission\_pct,job\_id

from employees

where commission\_pct is not null;

like

-- 13. first\_name 中包含en

select employee\_id,first\_name,salary

from employees

where first\_name like '%en%';

-- 14. 第3、4个字符是en

select employee\_id,first\_name,salary

from employees

where first\_name like '\_\_en%';

-- 15. job\_id 包含'\_A'

select employee\_id,first\_name,salary,job\_id

from employees

where job\_id like '%\\_A%'; -- \\_匹配下划线字符

not

-- 16. 工资低于3000或工资大于15000

select employee\_id,first\_name,salary

from employees

where salary not between 3000 and 15000;

-- where salary<3000 or salary>15000;

-- 17. 排除50和80部门

select employee\_id,first\_name,

salary,department\_id

from employees

where department\_id not in(50,80);

-- 18. 电话 phone\_number 不包含4

select employee\_id,first\_name,

salary,phone\_number

from employees

where phone\_number not like '%4%';

练习

-- 19. 995-1-1年之前入职(hire\_date)

select employee\_id,first\_name,

salary,hire\_date

from employees

where hire\_date<'1995-1-1';

-- 20. 种（job\_id）以 SA 开头的员工

select employee\_id,first\_name,

salary,job\_id

from employees

where job\_id like 'SA%';

-- 21.  en ，且在80 部门中的员工

select employee\_id,first\_name,last\_name

salary,department\_id

from employees

where (first\_name like '%en%' or

last\_name like '%en%') and

department\_id=80;

-- 22. 中所有员工 ，和工种后缀是 CLERK 的员工

select employee\_id,first\_name,job\_id

salary,department\_id

from employees

where department\_id=90 or job\_id like '%CLERK';

# distinct

去除重复

* select distinct a from tb1;  
  去除 a 字段重复值
* select distinct a,b from tb1;  
  a,b字段组合不重复

distinct

-- 23. 所有部门id,去除null值

select distinct department\_id

from employees

where department\_id is not null;

-- 24. 所有主管id, manager\_id,去除null值

select distinct manager\_id

from employees

where manager\_id is not null;

-- 25. 所有工作岗位id, job\_id

select distinct job\_id

from employees;

-- 26. 部门和主管组合不重复

select distinct department\_id,manager\_id

from employees;

-- 27. 哪些部门有文员 CLERK 后缀

select distinct department\_id

from employees

where job\_id like '%CLERK';

# order by 子句

排序,需要指定排序的字段和升降序

* select...from...where...order by a  
  以a字段升序排序
* select...from...where...order by a,b  
  先以a字段升序排序,a的值相同时,再按b升序排序
* asc - ascend 升序(默认)
* desc - descend 降序
* order by a desc
* order by a desc, b desc
* order by 子句必须出现在where之后  
  先过滤数据,过滤剩下的数据再排序

order by

-- 28. 所有员工按工资从高到低排序

select employee\_id,first\_name,salary

from employees

order by salary desc;

-- 29. 所有员工按部门升序排序,系统同部门按工资降序排

select employee\_id,first\_name,

salary,department\_id

from employees

order by department\_id, salary desc;

-- 30. 薪水大于等于 10000  的员工，按姓名排序

select employee\_id,

first\_name,last\_name,salary

from employees

where salary >= 10000

order by first\_name,last\_name;

-- 31. 查询所有员工，按部门编号升序排列,相同部门，按 first\_name升序排列

select employee\_id,first\_name,

salary, department\_id

from employees

order by department\_id,first\_name;

-- 33. 查询 50 和 80 部门的员工，按工资降序排列，工资相同按工种代码排列

select employee\_id,first\_name,

salary, department\_id,job\_id

from employees

where department\_id in(50,80)

order by salary desc, job\_id;

# 字段别名

select commission\_pct as **p**

from ...

where commission\_pct is not null

order by **p**

* as 别名  
  as 可以省略,一般不写
* where 不能使用字段别名  
  mysql处理sql查询顺序:

1. from找表
2. where过滤
3. 过滤后的数据,选取字段
4. order 排序

* order by 能用字段别名
* 什么时候使用字段别名?  
  用表达式计算生成的列,起一个别名

字段别名

-- 34. 计算年薪,按年薪从高到低排序

select employee\_id,first\_name,

salary\*12 **sal**

from employees

order by **sal** desc;

-- 35. 姓名连接显示,其中包含en的员工,按name排序

select employee\_id,

concat(first\_name,' ',last\_name) **name**

from employees

where concat(first\_name,' ',last\_name) like '%en%'

order by **name**;

# 目前,select 结构

select

distinct

from

where

order by asc|desc

# 作业

分类表

create table tb\_item\_cat(

id bigint(20) primary key auto\_increment,

parent\_id bigint(20),

name varchar(50),

status tinyint(1) default 1,

sort\_order int(4),

is\_parent tinyint(1) default 1,

created datetime,

updated datetime,

foreign key(parent\_id) references tb\_item\_cat(id)

)engine=innodb charset=utf8;

商品表

create table tb\_item(

id bigint(20) primary key auto\_increment,

cid bigint(10) not null,

brand varchar(50),

model varchar(50),

title varchar(100),

sell\_point varchar(500),

price bigint,

num int(10),

barcode varchar(30),

image varchar(500),

status tinyint(4) default 2,

created datetime,

updated datetime,

foreign key(cid) references tb\_item\_cat(id)

)engine=innodb charset=utf8;

商品详情

create table tb\_item\_desc(

item\_id bigint(20) primary key,

item\_desc text,

created datetime,

updated datetime,

foreign key(item\_id) references tb\_item(id)

)engine=innodb charset=utf8;

用户表

create table tb\_user(

id bigint(20) primary key auto\_increment,

username varchar(50) unique not null,

password varchar(32) not null,

phone varchar(20) unique,

email varchar(50) unique,

created datetime,

updated datetime

)engine=innodb charset=utf8;

收藏夹

create table tb\_collect(

id bigint(20) primary key auto\_increment,

user\_id bigint(20) not null,

item\_id bigint(20) not null,

item\_title varchar(100),

item\_price bigint(20),

item\_image varchar(200),

item\_param\_data varchar(200),

status tinyint(4) default 1,

created datetime,

updated datetime,

foreign key(user\_id) references tb\_user(id),

foreign key(item\_id) references tb\_item(id)

)engine=innodb charset=utf8;

购物车

create table tb\_cart( -- 购物车表

id bigint(20) primary key auto\_increment comment '自增主键',

user\_id bigint(20) not null comment '用户id',

item\_id bigint(20) not null comment '商品id',

num int(10) default 1 comment '商品数量',

item\_title varchar(100) comment '商品名称',

item\_image varchar(200) comment '商品图片',

item\_price bigint(20) comment '商品价格',

created datetime comment '创建时间',

updated datetime comment '更新时间',

foreign key(user\_id) references tb\_user(id),

foreign key(item\_id) references tb\_item(id)

)engine=innodb charset=utf8;

* + 1. 订单表(tb\_order)  
       订单的主键有生成规则  
       例如:用户id+时间毫秒值+随机数字
    2. 订单详情表(tb\_order\_item)