数据库初级笔记

一、数据库常识

1、什么是数据库

用保存数据的服务器

2、学习数据库的目的

- 测试过程中，定位bug，通过数据去查看业务上的问题
- 测试的时候创建数据，供测试时使用

3、学习的内容

- 数据库的增删和查
- 数据库的表的增删和查改
- 数据库的表结构的修改
- 数据的增删和查改（重点）

4、数据库的版本

- 关系型数据库
  - oracle
  - DB2
  - mysql
  - sqlservice

- 非关系型数据库
  - redis
  - mongoDB
  - mencashe

    非关系型数据库是一种内存数据库，无需事先给数据库建立字段，插入的数据没有字段和字段规则的限制

5、搭建数据库学习环境

- 工作中
  - 数据库是搭建在服务器上的
  - 需要用ip地址和账号密码进行连接
  - 需要使用到数据库连接工具来进行连接
- 学习
  - 使用xampp将数据库集成在自己的电脑上，直接连接自己的电脑来操作和学习
  - 安装xampp
  - 在xampp中运行mysql
  - 安装连接数据库的工具：软件包-->数据库-->Navicat

- 使用Navicat连接本地数据库：连接本地数据库不需要输入账号和密码，直接点击确定即可

6、sql语句分类

- DDL： 数据库/表定义语句（create,alter,drop）
  - 创建，修改，删除
- DML : 数据的操作语句（insert,update,delete）
  - 插入，修改，删除
- DQL : 数据查询语句（select）
  - 查询
- DCL : 数据库控制语句（grant,revoke,commit,rollback）
  - 赋予，移除权限，提交，回滚

二、sql语句

1、数据库语句

新建数据库

- 在连接下新建数据库
- 命令输入方式：
  - 命令界面：选中数据库右键-->选择命令界面
  - 查询页面：点击查询，在点击新建查询
- 新建数据库

```
create database 数据库名  character set utf8 ;

举例
    create database gz2242  character set utf8 ;
```

- 数据库名：
  - 数据名不能为纯数字
  - 不可以用数字开头
  - 不可以用中文和符号（下划线除外）
  - 数据库名不能重复

数据库查询

```
show databases;      #查询当前连接下的所有数据库
```

切换数据库

```
use  数据库名 ;
举例：
    USE gz2242;
```

查看当前数据库

```
查看当前使用的数据库：
    select database();
```

2、建表

表的列是有规则的，这个规则会限定这个字段的所有数据的数据类型和长度

在建表的时候就会先设定好字段的规则

数据类型：

- 数值：
  - int -- 整数型： 0 -1 4555
  - bool -- 布尔值： 0 1
  - float -- 浮点类型： 0.00 -1.2 77.33
- 字符类型：
  - char -- 固定字符： '今天'
    - char(10) ---字段内最长只能输入10个字符，超过10个字符的部分会被截取掉
  - varchar --- 可变字符：
    - varchar(10) -- 字段内最长只能输入10个字符，如果输出的长度没有10位，会自动将没有满的字节释放
- 时间格式
  - 插入日期格式的数据需要用引号如：'2022-04-15'
  - date -- 日期 （2022-04-15）
  - time -- 时间 （11:14:04）
  - datetime -- 日期时间 （2022-04-15 11:14:04）

新建表

- 新建表

```
create table 表名（字段1 数据类型 primary key ，字段2 数据类型）

举例：
    create table g001(id int(5) primary key , name char(3) , room
int(5),in_time datetime )
```

- 查询表

```
show tables;
```

- 查询表结构

```
desc  表名 ;
```

Field	Type	Null	Key	Default	Extra
id	int(5)	NO	PRI	(Null)	
name	char(3)	YES		(Null)	
room	int(5)	YES		(Null)	
in_time	datetime	YES		(Null)	

type : 数据类型

null : 是否为空

key： 键类型

default : 默认值

extra : 额外属性

- 删除表

```
drop table 表名
举例：
    drop table g001
```

- 修改表名

```
alter table 原表名 rename 新表名
举例：
    alter table g001 rename g002
```

3、表结构

增加字段

```
alter table 表名  add  字段名  数据类型
举例：
    alter table g002  add  age  int(2)
```

删除字段

```
alter table 表名  drop  字段名
举例：
    alter table g002  drop age
```

修改字段

```
alter table 表名 change  原字段名  新字段名  数据类型      #替换字段
举例：
    alter table g002  change  name  name_1 int(5)


alter table 表名 modify  字段名  新的数据类型      #修改字段的数据类型，不可以改字段名称
举例：
```

```
    alter table g002 modify  name_1  varchar(5)

修改字段的位置：   只能在修改或增加字段的时候修改字段位置

    after 字段名       #修改字段位置在对应字段的后面
    alter table g002  change  in_time  in_time  datetime  after id

    first   #放到最前面
    alter table g002 change in_time  in_time  datetime  first
```

多项修改

```
alter table 表名  add  字段名  数据类型 , modify  字段名  新的数据类型,change  原字段名  新字段名  数据类型

举例：
    alter table g002 change in_time  in_time  datetime  ,modify  name_1  varchar(10) , add  age  int(2) after id
```

4、表数据操作

insert ， updata , delete

插入数据

```
insert  into  表（字段名1，字段名2） values （值1，值2）
举例：
    insert into g002(id,name_1) VALUES(1,'小明')

注意：
    1、插入的字段必须是在表中真实存在的字段
    2、值的数量和位置一定要和字段一致
    3、注意插入数据是字段的数据类型
    4、如果向表中的左右字段都插入数据，字段可以省略不写
    举例：insert into g002 VALUES('2022-04-15 15:21:33',2,88,'小红',001)
    5、如果需要一次性插入多条数据，可以用逗号隔开数据
    举例：insert into g002 VALUES('2022-04-15 15:21:33',3,88,'小红',001),('2022-04-15 15:21:33',4,88,'小红',001)
```

查询数据

```
select  字段  from  表


select * from g002              #查询整个表的所有字段的数据
select name_1 from g002             #只查询name_1字段的数据
select name_1,age from g002             #只查询name_1和age字段的数据
select * from g002  where  id = 5           #只查询满足id = 5的所有字段数据

练习：
```

```
查询dep表中，来自湖南的人
select * from dep where address='湖南'

查询dep表中工资有10000块的人
select * from  dep where salary = 10000

查询dep表中，来自广东的人，显示姓名，地区，和年龄
select name,address,age from dep  where address='广东'
```

修改数据

```
update 表 set  字段=值  where  条件

update dep set salary = 22000 where name='王玉'

举例：
将dep表中来自湖南的人薪资改为20000
update dep set salary = 20000  where address='湖南'
```

删除数据

```
delete:
delete  from  表名  where 条件     #不加条件会删除整个表的数据，删除时一整行删除

delete from dep where name='王玉'      #删除name为王玉的一整行数据
delete from dep              #将这个表中的所有数据删除

truncate :           #将整个表的数据清空
    truncate  表名

delete和 truncate的区别：
    delete :可以加条件
    delete ：是逐行删除
    truncate:  整个表数据直接删除，速度更快
```

复制数据和表

```
复制数据：
    insert into 表A（表A的字段1，表A的字段2） select 表B的字段1，表B的字段2 from 表B
    举例：
        insert into g002(id,name_1,age) select id,name,age from dep
```

```
复制表：
    create table 复制后的表名  as  select  *  from  原表名  where  条件

    举例：
        create table dep_copy  as  select id,name,age,address from dep
```

5、主键和约束

约束：添加在字段上的规则---了解

主键约束

每个表只有一个主键约束，添加主键约束的字段内的数据是要求非空且唯一的

主键一般是添加在id上

```
primary key

添加主键：
    alter table 表名  modify 字段名 数据类型  primary key

删除主键：
    alter table 表名 drop  primary key


联合主键：
    定义表中多个字段为主键字段
    联合主键要求是多有字段都满足非空且唯一
```

自增长

必须加载主键字段上的约束

```
auto_increment   #自增长约束
增加：
alter table dep_copy  modify id int(5) auto_increment
insert into dep_copy  VALUES (0,'周五',88,'广东')   #添加了自增长约束的字段在插入的时候直接写0即可

删除：
    alter table dep_copy  modify id int(5)
```

非空约束:

```
在插入数据的时候，如果某个字段添加了非空约束，则如果没有插入该字段则报错
添加：
    alter table 表  modify 字段 数据类型 not null
删除：
    alter table 表  modify 字段 数据类型
```

唯一约束

```
unique     #唯一
限制数据可以为空但是数据不能重复

添加：
    alter table 表  modify 字段 数据类型  unique
删除：
    alter table 表  modify 字段 数据类型
```

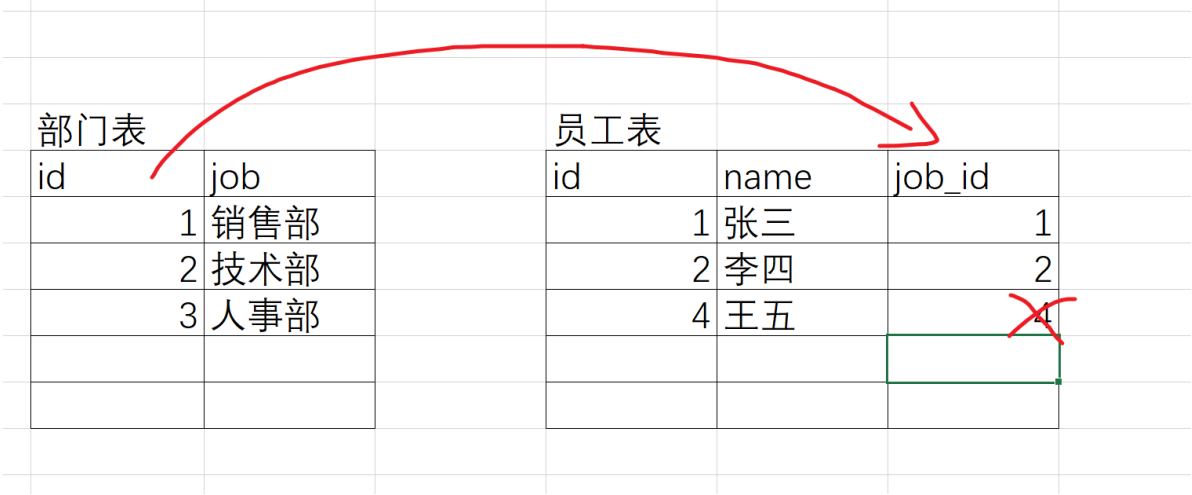
默认约束

添加了默认约束的字段，在插入数据的时候，如果没有插入数据，则使用默认约束

```
default  值
新增：
    alter table 表  modify 字段 数据类型  default  值
删除：
    alter table 表  modify 字段 数据类型  default  null
```

外键约束

一个表的字段受到了另外一个表的主键字段的约束



部门表

id	job
1	销售部
2	技术部
3	人事部

员工表

id	name	job_id
1	张三	1
2	李四	2
4	王五	4

6、查询的条件

运算符

- 赋值运算符

```
=      #在修改变量或者数据的时候使用
```

- 比较运算

```
>  <  >=  <=  =  <>或!=

<>和!= : 不等于

举例：
    select * from dep where salary >= 5000

    select 6<7  #结果为1
    select 6>7  #结果为0
```

- 算数运算符

```
+ - * /  %

% :  取模（求余数）
    举例：select 5%2   # 结果：  1    5-2*2=1
```

- 逻辑运算符

```
and :并且
    select  3>1 and 4>5  结果：0
    select  3>1 and 4<5  结果：1
    select  3<1 and 4>5  结果：0
    select  3<1 and 4<5  结果：0
    举例：
    查找广东的技术人员中，薪资大于8000的人
    select * from dep where address='广东' and job='技术' and salary >8000

or：或者

    举例：
        查找来自广东或湖南的人
        select * from dep where address='广东' or address='湖南'

        查找职位为技术或经理的人
        select * from dep where  job ='技术' or job ='经理'

not：非
    举例：
        查找dep表中，工资不为空的人
        select * from dep  where  salary  is not null


优先级：
```

```
    not -->  and ---> or
    优先级可以通过（）改变


练习：

1、查询年龄小于25的人
select * from dep where  age<25

2、查询年龄小于25的人或年龄大于30的人
select * from dep where  age<25 or age>30

3、查询所有工资是年龄的整倍数的人
select * from dep where  salary % age = 0

4、查询id在5到10的人（包含10）
select * from dep where id >=5 and id <=10

5、查询广东的技术人员或湖南的技术人员
select * from dep where  address='广东' and job ='技术'   or  address='湖南'
and job ='技术'
```

查询条件

- 空值查询

```
空值无法比较，不能用比较符号进行对比（不能用 = null）,只能定性

正确的写法：(如果是赋值，是可以用=的)
    is null
    is not  null

    查找dep表中，工资不为空的人
    select * from dep  where  salary  is not null
```

- 模糊查询： like

```
%  ：  匹配多个个任意字符
_  ：  匹配一个任意字符

举例：
    查找dep表中的老王
    select * from dep  where name like '王%'

    查找dep表中三个字的老王
    select * from dep  where name like '王__'

    查找dep表中，有峰字的人
    select * from dep  where name like '%峰%'
```

区间查询

- between

  格式： 字段名 between 值1 and 值2

  查询结果与：字段>=值1 and 字段 <=值2 一致

```
查询id在5到10的人（包含10）

select * from dep where id  between  5 and 10
```

- in

  格式： 字段 in （值1，值2，值3）

  查询结果与： 字段=值1 or 字段=值2 or 字段=值3 一致

```
查询来自广东，湖南，江西的人

select * from dep where  address='广东' or address='湖南' or address='江西';
select * from dep where address in ('广东','湖南','江西')
```

显示查询

- 取别名

  给展示的字段取一个新的名字，不显示原表的字段。只会当前显示的结果，不会改变原表字段

```
select name '姓名',age '年龄',address  '地址' from dep
```

- 排序

  格式： order by 字段名 asc/desc

  默认是升序

  如果进行多个字段排序，则先进行第一个字段的排序，再进行第二个字段的排序

```
将dep表中按照工资进行降序排序
select * from dep  order by salary desc      #降序
select * from dep  order by salary           #默认是升序


先按照表中的工资降序，在按照年龄升序
select * from dep  order by salary desc , age asc
```

- 去除重复

  当查询结果中包含重复数据的时候，会去除重复数据展示

  当多个字段同时进行去重显示的时候，会将字段进行组合去重

```
distinct 字段名,字段名2

公司中一共有多少种职位
select distinct job  from dep


select distinct job,sex  from dep   #两个字段进行去重
```

- 显示行数

  格式： limit num1，num2

  num1 : 显示行数的起始行，第一行的行号为0，当num1为0 的时候可以省略该参数不写

  num2 : 显示的行的数量，开始的第一行到结束的行数

```
举例：
    显示dep表中第4行数据到10行数据
    select * from dep  limit 3,7

    显示dep表中工资从高到低的前5名
    select * from dep order by salary  desc  limit 5
```

7、函数

- 单行函数
  - 四舍五入：

```
round(num1,num2)
    num1:传入需要四射五入的数字
    num2:
        如果为正数则保留小数点后对应的num位数字
        如果负数则保留小数点前的num位数字

select ROUND(16576.78343,-2)          #结果：16600
select ROUND(16576.78343,2)          #结果：16576.78
```

  - 拼接字符:

    将另个字符进行拼接，返回一个

```
concat(str1,str2)
    str1、str2：分别是两个字符串，这里的参数的数量不受限制

select concat(name,'来自',address) from dep  #王小玉来自湖南
```

  - 拆分字符

```
left(str,num)   #从字符的左边起拆分num个字符
    举例：select left(name,2)from dep
right(str,num)  #从字符的右边起拆分num个字符
```

- 大小写转化

```
upper()          #将括号内的字字母转化成大写
    select upper('ABCdefg')

lower()          #将括号内的字字母转化成小写
    select lower('ABCdefg')
```

- 聚合函数（多行函数）

  普通字段不要与聚合函数同时查询，因为查询结果的数量不同，会导致结果不准确

  例如：

  查询dep中工资最大的人的信息 select max(salary),name from dep

  - max() 返回数据中的最大值
  - min() 返回数据中的最小值
  - avg() 返回数据中的平均值
  - sum() 返回数据中的综合
  - count() 返回数据中的数量
    - 尽量使用 * 或者使用主键字段
    - 空值不参与数量的计算

  举例：

```
查询北京人中，最大薪资的人
select max(salary) from dep where address='北京'
```

  练习：

```
1、查询公司每年需要支出的总工资
select sum(salary)*12  from dep

2、查询技术岗的平均工资
SELECT avg(salary) from dep  where job ='技术'

3、查询公司员工中年龄最大的岁数
select max(age) from dep

4、查询公司员工中20-30岁的平均工资
select avg(salary) from dep where  age between 20 and 30

5、查询工资在5000-8000的人数
select count(*) from dep where salary between 5000 and 8000


1、查询广东的技术人员中，平均薪资
select avg(salary) from dep where address='广东' and job ='技术'

2、查询大于25岁的女性员工的最大薪资
select max(salary) from dep where age>25 and sex ='女'
```

```
3、查询湖南的技术或广东的销售中，最小薪资
select min(salary) from dep where address='湖南' and job='技术'   or
address='广东' and job='销售'
```

7、分组

group by 字段

按照字段内的数据拆分临时表，相同的数据组成一张临时表

当分组之后，查询的字段中只有聚合函数和被分组的字段为有效数据，其他字段没有意义

如果是多字段分子，会按照两个字段的组合进行分组

```
求各个省份的人数dep表
select count(*),address  from dep group by address

求每个省份中年龄最大的数
SELECT address ,max(age) from  dep  group by  address


练习：
    查询不同工种的最大薪资和平均薪资
select  job ,max(salary),avg(salary)  from dep group by job
```

having

- having 条件
- having只能在分组之后使用，过滤的条件可以使用聚合函数，不可以使用单行函数
- where 是在原表的字段经过条件筛选后的结果，where可以使用单行函数，不可以使用聚合函数

```
查询平均薪资大于8000的岗位
select job ,avg(salary) from dep  group by job  having avg(salary)  > 8000


查询平均年龄大于25的岗位
select job,avg(age) from dep group by job having avg(age)>25
```

8、整理

```
select 字段,distinct from  表名  where  条件  group by 字段   having 条件  order by
字段  limit

执行顺序：
    from --> where --> group by  --> having --> select --> distinct --> order by
---> limit
```

练习：
查询江西的各个岗位的平均薪资

```
select job ,avg(salary) from dep where address='江西' group by job
```

查询江西平均薪资大于6500的岗位

```
select job ,avg(salary) from dep where address='江西' group  by job having
 avg(salary) >6500

select  job ,avg(salary), avg(salary) > 6500  from dep where address='江西' group
by job
```