
# MySQL

深 圳 市 门 道 信 息 咨 询 有 限 公 司
Shenzhen MT Information Consulting Co . , LTD

# 目录

# 1.前言

该文档仅针对初从事软件测试工程师应该掌握的一些数据库知识做讲解（数据库本身强大且方向众多）

## 1.1 数据库概念

- 什么是数据库

  数据库（Database）是按照数据结构来组织、存储和管理数据的仓库。

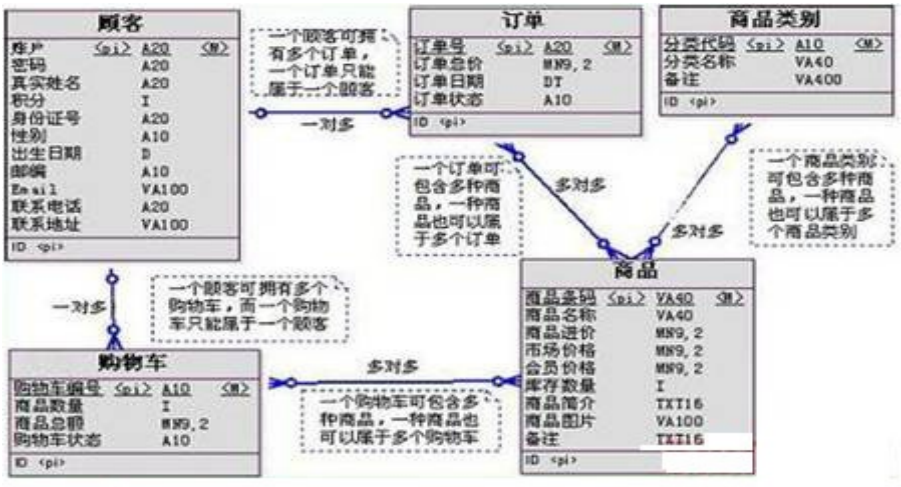
仓库(货架)

数据库(表)

- 数据库分类
  - 小型数据库：Access
  - 中型数据库：MySQL、SQLServer
  - 大型数据库：Oracle、DB2、Sysbase




- 关系型数据库特点： 实体、属性、关系



# 2.MySQL 简介

## MySQL 起源

- 起源不是很明确。一个比较有影响的说法是，基本指南和大量的库和工具带有前缀“my” MySQL AB 创始人之一的 Monty Widenius 的女儿也叫 My。这两个到底是哪一个给出了 MySQL 这个名字至今依然是个密，包括开发者在内也不知道。
- MySQL 的海豚标志的名字叫“sakila”，它是由 MySQL AB 的创始人从用户在“海豚命名”的竞赛中建议的大量的名字表中选出的。
- MySQL 的历史最早可以追溯到 1979 年，Monty Widenius 用 BASIC 设计了一个报表工具，过了不久，又将此工具，使用 C 语言重写，移植到 Unix 平台，当时，它只是一个很底层的面向报表的存储引擎。这个工具叫做 Unireg
- 1996 年，MySQL 1.0 发布,只面向一小拨人，相当于内部发布，10 月，MySQL 3.11.1 发布
- 2000 年 4 月，MySQL 对旧的存储引擎进行了整理，命名为 MyISAM
- 2001 年，集成了存储引擎 InnoDB，这个引擎同样支持事务处理，还支持行级锁。
- 2008 年 1 月 16 号 MySQL 被 Sun 公司收购

## 2.2 MySQL 简介和特点

简介：

- 最流行的关系型数据库管理系统，在 WEB 应用方面 MySQL 是最好的应用软件之一
- 采用了双授权政策，分为社区版和商业版
- 体积小、速度快、成本低，开源码
- 黄金搭档：LAMP、LNMP

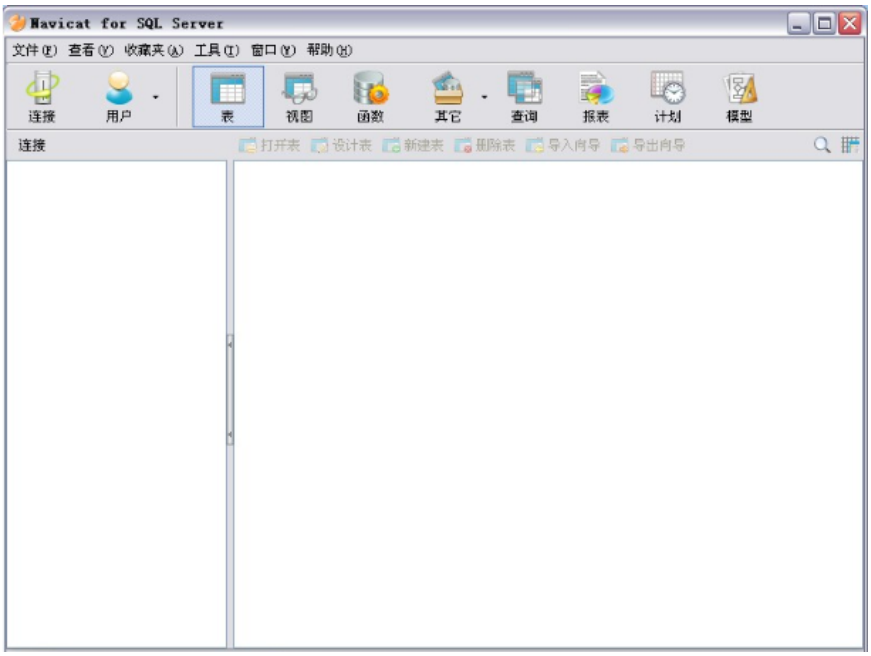
特点：

- 可移植性
- 支持多线程，充分利用 CPU 资源
- 优化的 SQL 查询算法，有效地提高查询速度
- 多语言支持
- 可以处理拥有上千万条记录的大型数据库
- 支持多种存储引擎
- 使用标准的 SQL 数据语言

# 3.图形连接工具

正确安装上 MySQL 后，我们可以通过字符界面操作 MySQL，但字符学习曲线大且实际工作中往往把整个服务器放在机房，我们不可能每次要去操作 MySQL 时就跑机房，因此，需要我们所有人员(包括测试工程师)通过图形连接工具连接到数据库服务器进行操作

一个人通过工具可以访问 N 个不同服务器（相互之间网络畅通情况下）
常见远程工具：navicat、sqlyog、workbench




# 4.测试学习 MySQL 的意义

我们在现实中使用的系统，应用程序需要数据做依托，因此就需要数据库提供这样的服务。
作为测试工程师，我们最主要的任务是**查数据**

# 5.安装

## 安装

在 Linux 中已实现安装或查看 Linux 课件

## 5.2 启动、停止、配置文件

service mysqld start
service mysqld stop
vi /etc/my.cnf

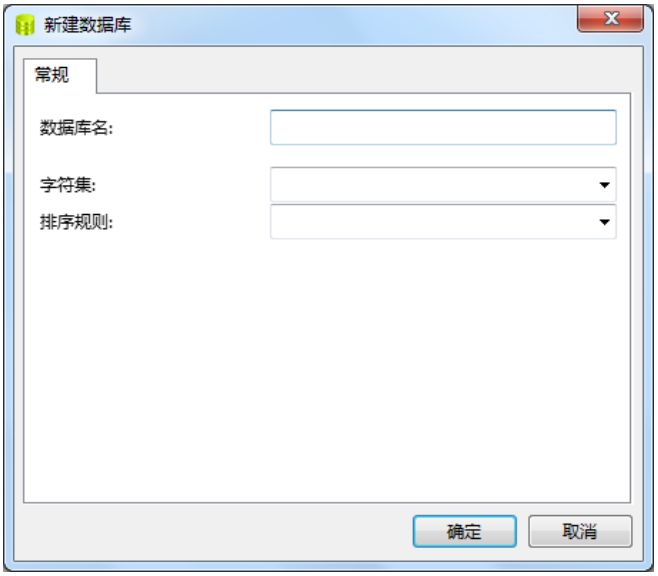
# 6.Navicat 简单使用习惯

## 为什么要连接数据库

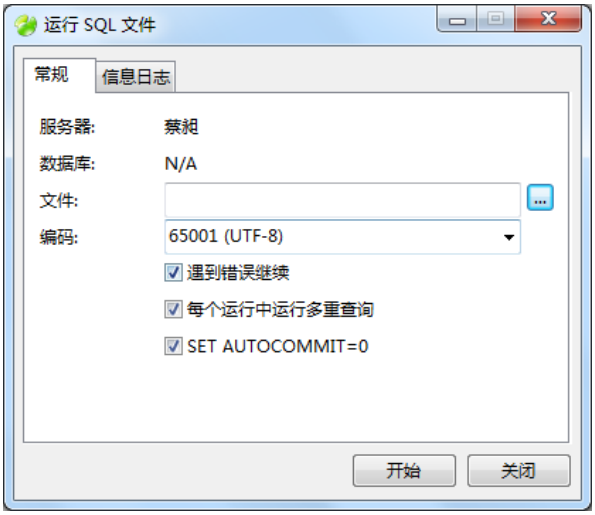
上班后，你测试的系统的所有数据都存储在数据库中，你会遇到这样的场景：
你通过界面新注册了一个用户，但这个用户真的添加上了吗（真的入库了吗）。其实你并不知道，这时你需要连上这个系统对应的数据库进行查看

## 6.2 连接服务器


## 6.3 新建数据库



## 6.4 导入数据文件




# 7.起码的术语

数据库中，行被称为“记录”，列称为“字段”

你上班后，熟悉一个系统，要熟悉他的数据库结构，因此你上班会得到一个"数据库文档.doc"的文件。

对于你来说，要熟悉这个文档，要熟悉表，熟悉表中有哪些字段，熟悉字段中都有哪些数据，这是一个测试工程师最基本的要求

# 8.熟悉我们要用的表

1. 字段名都是英文名（大小写均可）
2. 表内容为大写引用则大写，为小写则小写，为中文则中文，要保持一致

emp：员工编号、员工姓名、职位、上级领导、入职日期、薪金、佣金、部门编号

| EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
|---|---|---|---|---|---|---|---|
| 7369 | SMITH | CLERK | 7902 | 1980-12-17 | 800 | (Null) | 10 |
| 7499 | ALLEN | SALESMAN | 7698 | 1981-02-20 | 1600 | 300 | 30 |
| 7521 | WARD | SALESMAN | 7698 | 1981-02-22 | 1250 | 500 | 30 |
| 7566 | JONES | MANAGER | 7839 | 1981-04-02 | 2975 | (Null) | 20 |
| 7654 | MARTIN | SALESMAN | 7698 | 1981-09-28 | 1250 | 1400 | 30 |
| 7698 | BLAKE | MANAGER | 7839 | 1981-05-01 | 2850 | (Null) | 30 |
| 7782 | CLARK | MANAGER | 7839 | 1981-06-09 | 2450 | (Null) | 10 |
| 7788 | SCOTT | ANALYST | 7566 | 1987-04-19 | 3000 | (Null) | 20 |
| 7839 | KING | PRESIDENT | (Null) | 1981-11-17 | 5000 | (Null) | 10 |
| 7844 | TURNER | SALESMAN | 7698 | 1981-09-08 | 1500 | 0 | 30 |
| 7876 | ADAMS | CLERK | 7788 | 1987-05-23 | 1100 | (Null) | 20 |
| 7900 | JAMES | CLERK | 7698 | 1981-12-03 | 950 | (Null) | 30 |
| 7902 | FORD | ANALYST | 7566 | 1981-12-03 | 3000 | (Null) | 20 |
| 7934 | MILLER | CLERK | 7782 | 1982-01-23 | 1300 | (Null) | 10 |

dept：部门编号、部门名称、部门位置

| DEPTNO | DNAME | LOC |
|---|---|---|
| 10 | ACCOUNTING | NEW YORK |
| 20 | RESEARCH | DALLAS |
| 30 | SALES | CHICAGO |
| 40 | OPERATIONS | BOSTON |

# 9.注释

```
#DELETE FROM EMP
/*DELETE FROM EMP */
-- DELETE FROM EMP 这种注释后面要加一个空格
```

# 10. 简单查询

## 工作场景

1、上班后，有一个功能，展示了所有员工的信息（8 条数据），但问题是：真的只有这些员工吗？其实你并不知道，所以需要通过查询员工表的对应数据，来判断员工列表显示的正确性

2、上班后，有一个功能，展示了所有员工的信息（8 条数据），但数据有的为 0，有的为空，但问题是：这些信息真的是 0 和空吗？其实你并不知道，所以需要通过查询员工表的对应数据，来判断员工列表显示的正确性

# 10.2select

语法：

```
select 字段 from 表;
```

## 查询一个字段

```
select ename from emp;#养成分号结束的习惯
```

## 10.2.2 查询多个字段

```
select ename,sal,comm,deptno from emp; #多个字段逗号分隔
```

## 10.2.3 查询全部字段

```
select * from emp;
#*表示模糊匹配这个表的所有字段。工作中非常常用
```

## 10.2.4 运算和拼接和去除重复

运算：字段类型建议为数字；当运算列为字符串时，运算结果为 0

```
select sal*12 from emp;
select ename*12 from emp;
```

#注意如果字段上有空值，会造成运算结果是空值
#解决办法：ifnull(字段，默认值)

```
select ename,sal,comm,sal+comm,(sal+comm)*12 from emp;

select ename,sal,comm,sal+ifnull(comm,0),(sal+ifnull(comm,0))*12 from
emp;
```

拼接：CONCAT函数

```
select CONCAT(empno,ename) from emp;
select CONCAT(empno,'_',ename) from emp;
```

去除重复

```
select distinct deptno from emp
```

## 10.2.5 别名

1、字段别名
目的：显示时让用户一目了然字段含义

注意：别名要见名知意

```
SELECT
    ename,
    sal,
    comm,
    sal + ifnull(comm, 0) AS 月薪,
    (sal + ifnull(comm, 0)) * 12 AS 年薪
FROM
    emp; #别名不理想，虽然不报错，但不符合实际工作
```

```
SELECT
    ename,
    sal,
    comm,
    sal + ifnull(comm, 0) month_sal,
    (sal + ifnull(comm, 0)) * 12 year_sal
FROM
    emp;
```

2、表别名

目的：记不住字段内容（太长），写长SQL速度慢等

注意：别名.

```
select e.hiredate,e.sal,e.comm from emp e;
```

# 10.3工作场景

你在界面上操作了按照员工姓名查询后，数据展示为 5 条，但真的只有这 5 条员工吗？其实你并不确定，因此连上数据库，查对应的员工信息表并过滤出指定的信息，用查出来的数据和界面数据进行对比，如一致，则说明开发写的确实没问题

# 10.4where

语法：

```
select 字段
from 表
[where 条件]
```

## 数字

1、>,<,=,<=,>=,<>

```
select * from emp where sal>1500; #他隐含了一个朴素的翻译
```

2、between...and...

```
select * from emp where sal between 1500 and 2500;
#包含1500和2500，是个闭区间 [1500,2500]

select * from emp where sal >=1500 and sal<= 2500;
#包含1500和2500，是个闭区间 [1500,2500]
```

## 10.4.2 字符串

1、= #全匹配

```
select ename from emp where job='CLERK';
```

2、like

模糊匹配

_:匹配1个字符

%:匹配0个或多个字符

```
select * from emp where job like '%R%';
```

还有如下不常用写法：R%、__R____、__R____%、%__R____

## 10.4.3 空值

is null

```
select * from emp where mgr is null;
```

## 10.4.4 一串值

in

```
select * from emp where ename in ('SCOTT','smith','菜10','范bb','凤姐
');
```

## 10.4.5 取反

not

```
in                      not in
is null                 is not null
like                    not like
between...and...        not between...and...
```

## 10.4.6 and &or

```
#工作中，基本上全是and
```

```
#and取交集，可以随便写多个，数据会越来越少
#or 取并集，可以随便写多个，数据会越来越多


#工作中这句话最多
#作为测试的你，工作的主要内容是查数据，因此你会先查这个表，然后where数据，发现还是有很多，于是and ,and后发现还是有多余的，于是再and，还有多余，继续and，直到and出我要的数据

select *
  from emp
 where sal > 1500
   and deptno = 30
   and ename = 'BLAKE';

select *
  from emp
 where sal > 1500
   or deptno = 30
   or ename = 'BLAKE';


select *
  from emp
 where sal > 1500
   or deptno = 30
   or deptno=20
   and ename = 'BLAKE';


#相当于下面这句SQL:
select *
  from emp
 where sal > 1500
   or deptno = 30
   or (deptno=20
   and ename = 'BLAKE');
```

## 10.4.7 基础篇作业

1. 选择部门 30 中的雇员
2. 列出所有办事员的姓名、编号和部门
3. 找出佣金高于薪金的雇员

4. 找出佣金高于薪金 60%的雇员
5. 找出部门 10 中所有经理和部门 20 中的所有办事员的详细资料
6. 找出部门 10 中所有经理、部门 20 中所有办事员，既不是经理又不是办事员但其薪金>=2000 的所有雇员的详细资料
7. 找出收取佣金的雇员的不同工作
8. 找出不收取佣金或收取的佣金低于 100 的雇员

# 10.5工作场景

1、界面上分别展示了热销商品，精选商品，促销商品等，但其实在数据存储时，他们都存在商品表中，我们通过分组，就可以实现想要的效果

2、界面上展示了各个部门的信息及部门人员个数统计和该部门的盈利总和，那他又是怎么做出来的呢？这就是分组的魅力

# 10.6分组

## 分组函数

**常见分组函数：avg、sum、max、min、count**

```
#avg、sum 类型只能是number,max、min、count不限制类型,count不统计null值，工作中常用count(*)来查询整个表有多少条记录
select avg(sal) from emp;
select avg(ename) from emp;
select sum(sal) from emp;

select max(ename) from emp;
select min(hiredate) from emp;

select count(comm) from emp;--***
select count(*) from emp;---******
```

```
#可以做运算，但不能带表达式
select sum(sal+nvl(comm,0)) from emp;
select avg(sal>1500) from emp;--错误
select avg(sal) from emp where sal>1500;--正确

#他们都不能作用在where子句中--******************

select * from emp where sal>avg(sal);--错误

select * from emp where sal>(select avg(sal) from emp);--正确
```

## 10.6.2 混合

当我们写完 select 后，马上检查是否有组函数，如果有，马上 group by
分组规则为：除开组函数外的其他字段全部参与分组

```
select job,min(sal) from emp;--错误
select job,min(sal) from emp group by job;

select job,min(sal),ename from emp group by job;--错误
select job,min(sal),ename from emp group by job,ename;

select job,min(sal) from emp group by job,ename;
select job,min(sal),sal from emp group by job,ename;--错误
select job,min(sal),sal from emp group by job,ename,sal;

select job,min(sal),count(*),max(hiredate) from emp group by job;
```

## 10.6.3 having

对分组后的数据过滤

面试时，喜欢问：where和having的区别？ --********
--其他区别请你百度，然后找出你可以理解的说辞来告诉面试官，但是下面这2点是你必须首先要告诉的
where:先过滤再分组，不能使用组函数
having：先分组再过滤，可以随便使用组函数

```
select job,min(sal) from emp group by job having min(sal) >2500;

#having中不能使用字段别名,也不能用除select外的其他字段，但不限制组函数
select job,min(sal) new_sal from emp group by job having
new_sal>2500;
select job,min(sal) new_sal from emp group by job having sal>2500;

select job,min(sal) new_sal from emp group by job having
count(*)>=3;# select上隐藏了所有的组函数

select job,min(sal) new_sal,count(*),max(hiredate) from emp group by
job having count(*)>=3;

select job,min(sal) new_sal,count(*),max(hiredate) from emp group by
job having avg(sal)>=2000;
```

# 10.7工作场景

天猫、淘宝等电商平台，都有按照价格排序、按好评排序；美团外卖还有按距离优先或价格优先等排序，他们就是通过数据库排序后来展示数据的

# 10.8排序

排序永远放在最后做

## 10.8.1升序（默认） asc

```
select job,min(sal) from emp group by job order by job asc;
select job,min(sal) from emp group by job order by job;
```

## 10.8.2降序 desc

```
select job,min(sal) from emp group by job order by job desc;
```

## 10.8.3别名排序

```
select sal,comm,sal+nvl(comm,0) month_sal from emp order by
month_sal;
```

## 10.8.4按照字段位置排

```
select sal,comm,sal+nvl(comm,0) month_sal from emp order by 3;
```

## 10.8.5综合排序

先按照第一个规则排，如果有相同，才按照第二个规则排，如果还有相同，才按照第三个规则排，依次类推

```
select * from emp order by sal,comm desc,job,ename desc,empno;
select * from emp order by sal,comm desc,job,ename desc;

select * from emp order by empno;
select * from emp order by empno,sal,comm desc,job,ename desc;

select * from emp deptno desc,job,sal desc,comm,empno;
```

# 11. 工作场景

1. 以上学习的查询都是针对一个表的，但实际工作中往往表和表之间有巨大的各种关联关系，而这样的关联关系就决定了我们需要用多表查询（子查询）来解决。

2. 请你非常认真的掌握好子查询，他将是你工作和面试和笔试的重中之重
3. 一个好的测试工程师，子查询一定是过关的

# 12. 子查询

## 做题技巧

1. 读一下这个题目，找“的”，右边就是我们要的字段，select 出来
2. 这些字段来自什么表，找出来， from
3. 如果是多表，找出他们的关系，这个关系一般是a.id=b.id
   即2表中有相同字段马上相等
4. 再读一次题目，用拍大腿的精神想问题，自然翻译

```
1. 列出至少有1个雇员的所有部门名
2. 列出薪金比"SMITH"多的所有雇员
3. 列出所有雇员的姓名及其直接上级的姓名
4. 列出入职日期早于其直接上级的所有雇员
5. 列出部门名称和这些部门的雇员,同时列出那些没有雇员的部门
6. 列出所有“CLERK”（办事员）的姓名及其部门名称
7. 列出各种工作类别的最低薪金，显示最低薪金大于1500的记录
8. 列出从事“SALES”（销售）工作的雇员的姓名，假定不知道销售部的部门编号
9. 列出薪金高于公司平均水平的所有雇员
10. 列出与“SCOTT”从事相同工作的所有雇员
11. 列出某些雇员的姓名和薪金，条件是他们的薪金等于部门30中任何一个雇员的薪
12. 列出某些雇员的姓名和薪金，条件是他们的薪金高于部门30中所有雇员的薪金
13. 列出每个部门的信息以及该部门中雇员的数量
14. 列出所有雇员的雇员名称、部门名称和薪
15. 列出各种类别工作的最低工资
16. 列出各个部门的MANAGER（经理）的最低薪金
17. 列出按年薪排序的所有雇员的年薪
18. 列出按年薪排名的所有雇员的年薪
19. 列出按年薪排序的第四名雇员的年薪
20. 列出薪金水平处于第四位的雇员
21. 列出按月薪排序的所有雇员的年薪取第三条记录和第五记录
```

## 12.2全、内、左、右连接

inner join、left join 、right join

```
1. select * from a,b;
2. select * from a,b where a.id=b.id;
   select * from a inner join b on a.id=b.id;
3. select * from a right join b on a.id=b.id;
```

```
id      id
1        1
4        4
         10
         9
3        3

4.  select * from a left join b on a.id=b.id;

id      id
1       1
2
3       3
4       4
5
6
7
```

# 13. limit

- limit是用来限制查询结果的记录数的
- limit x,y。指筛选出结果的第x行后的y行。如果x不填也是可以的，默认为0

语法：

```
SELECT * FROM table LIMIT [offset,] rows | rows OFFSET offset
```

案例：

```
select * from emp limit 0,5;
select * from emp limit 10,5;
select * from emp limit 5;
```

# 14. 日期处理

日期是分为：年月日和时间戳 2 种方式存储的，所以 2 种你都要掌握

```
now(), current_timestamp();     -- 当前日期时间
current_date();                  -- 当前日期
current_time();                  -- 当前时间
date('Y-m-d H:i:s');    -- 获取日期部分
time('Y-m-d H:i:s');    -- 获取时间部分
date_format('yyyy-mm-dd hh:ii:ss', '%d %y %a %d %m %b %j');    -- 格式化时间
unix_timestamp();               -- 获得unix时间戳
```

```
from_unixtime();                    -- 从时间戳获得时间
```

# 15. 工作场景

在实际工作中，是项目经理或开发去建表，我们没有资格，但是作为数据库的体系，我们要学习表的各种知识

# 16. 建表

## 建表

dept：

```
CREATE TABLE `dept` (
  `DEPTNO` int(2) NOT NULL,
  `DNAME` varchar(14) DEFAULT NULL,
  `LOC` varchar(13) DEFAULT NULL,
  PRIMARY KEY (`DEPTNO`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

emp：

```
CREATE TABLE `emp` (
  `EMPNO` int(4) NOT NULL AUTO_INCREMENT,
  `ENAME` varchar(10) DEFAULT NULL,
  `JOB` varchar(9) DEFAULT NULL,
  `MGR` int(4) DEFAULT NULL,
  `HIREDATE` date DEFAULT NULL,
  `SAL` int(7) DEFAULT NULL,
  `COMM` int(7) DEFAULT NULL,
  `DEPTNO` int(2) DEFAULT NULL,
  PRIMARY KEY (`EMPNO`),
  KEY `FK_DEPTNO` (`DEPTNO`),
  CONSTRAINT `fk_deptno` FOREIGN KEY (`DEPTNO`) REFERENCES `dept`
(`DEPTNO`)
) ENGINE=InnoDB AUTO_INCREMENT=7935 DEFAULT CHARSET=utf8;
```

## 16.2四大约束

什么是约束？

约束英文：constraint

约束实际上就是表中数据的限制条件

约束作用

表在设计的时候加入约束的目的就是为了保证表中的记录完整和有效

约束种类

- 非空约束（not null）
- 唯一性约束（unique）
- 主键约束(primary key) PK
- 外键约束(foreign key) FK

非空约束（not null）：用not null约束的字段不能为null值，必须给定具体的数据

```
CREATE TABLE `NewTable` (
    `id` INT NULL,
    `ename` VARCHAR (20) NOT NULL
);
# ERROR 1364 (HY000): Field 'ename' doesn't have a default value
```

#唯一性约束：unique约束的字段，具有唯一性，不可重复，但可以为null

1、列级约束

```
CREATE TABLE `NewTable` (
    `id` INT NULL,
    `ename` VARCHAR (255) NOT NULL unique,
);
# ERROR 1062 (23000): Duplicate entry 'mtesting' for key 'ename'
```

2、表级约束

```
CREATE TABLE `NewTable` (
    `id` INT NULL,
    `ename` VARCHAR (255) NOT NULL,
    CONSTRAINT uk_ename UNIQUE (ename)
);
# ERROR 1062 (23000): Duplicate entry 'mtesting' for key 'ename'
```

3、新建表后追加唯一约束

```
ALTER TABLE ` NewTable` ADD unique(`ename`);
```

#主键

#实际工作中，我们要唯一的保证一条记录是非空且唯一的，这样可以防止“脏”数据，主键能帮我们实现

#工作中，一般第一个字段都叫id或者xxid，都设置为主键

#有了主键就会在主键上新建一个索引，这样通过索引找数据会快

1、列级约束

```
CREATE TABLE `NewTable` (
    `id` INT NULL primary key,
    `ename` VARCHAR (255) NOT NULL unique,
);
```

2、表级约束

```
CREATE TABLE `NewTable` (
    `id` INT NULL,
    `ename` VARCHAR (255) NOT NULL,
    CONSTRAINT pk_id primary key(id)
);
```

3、新建表后追加主键约束

```
alter table `NewTable` add constraint `pk_id` primary key(`id`);
```

4、在MySQL数据库提供了一个自增的数字，专门用来自动生成主键值，主键值不用用户维护，自动生成，自增数从1开始，以1递增

```
CREATE TABLE `NewTable` (
    `id` INT NULL primary key auto_increment,
    `ename` VARCHAR (255) NOT NULL unique,
);
```

外键：表和表之间的关系靠外键维护

1. 外键值可以为 null
2. 外键字段去引用一张表的某个字段的时候，被引用的字段必须具有 unique 约束
3. 有了外键引用之后，表分为父表和子表
4. 创建先创建父表
5. 删除先删除子表数据
6. 插入先插入父表数据

```
CREATE TABLE `NewTable` (
    `id` INT NULL,
    `ename` VARCHAR (255) NOT NULL,
    `cno` INT NULL,
    CONSTRAINT `fk_cno` FOREIGN KEY (`cno`) REFERENCES `dept`
(`DEPTNO`),
) ENGINE = INNODB;
```

新建表后追加外键约束：

```
ALTER TABLE NewTable ADD CONSTRAINT fk_dno FOREIGN KEY (cno)
REFERENCES dept (deptno);
```

# 17. 事务

事务：事务由单独单元的一个或多个 SQL 语句组成，在这个单元中，每个 MySQL 语句是相互依赖的。而整个单独单元作为一个不可分割的整体，如果单元中某条 SQL 语句一旦执行失败或产生错误，整个单元将会回滚。所有受到影响的数据将返回到事物开始以前的状态；如果单元中的所有 SQL 语句均执行成功，则事务被顺利执行。

引擎方案：

- MyISAM，MySQL 5.0 之前的默认数据库引擎，最为常用。拥有较高的插入，查询速度，但不支持事务
- InnoDB 事务型数据库的首选引擎，支持 ACID 事务，支持行级锁定，MySQL 5.5 起成为默认数据库引擎，支持事务

# 18. 工作场景

实际工作，除了常规的查询，还必须要掌握增删改，因为你无法避免你要去“动”数据库的数据

# 19. 增删改

DQL：数据查询语言 select

DDL：数据定义语言 create、alter、drop

DML：数据操纵语言 insert、update、delete

## insert

语法：

```
insert into 表(字段) values (值);
```

案例：

```
insert into shop_user(id,name,pwd) values (4,'菜10','cai10');
```

注意：

```
#1、如果你要对整个表的所有字段都插入，可以不写 “(字段)”,非常不推荐

insert into shop_user values(5,'陈00','chen00');

#2、用其他表的数据作为本表，或以前运维时填补数据的方案
#全部字段
insert into shop_user
select empno,ename,dname
from emp,dept
```

```
where emp.deptno=dept.deptno
and dept.deptno in (10,20);

#部分字段
insert into shop_user(id,pwd)
select empno,dname
from emp,dept
where emp.deptno=dept.deptno
and dept.deptno =30;
```

# 18.2update

语法：

```
update 表
set 字段=新值[,字段=新值][,....]
where 条件
```

案例：

```
UPDATE shop_user
SET pwd = '123abc'
WHERE
    id IN (2, 3);
```

注意：

```
#1、面试时，非常喜欢问 update，所以你一定要牢记
#2、工作中，我的同事因为 where 后少写了条件，导致被扣钱被辞退，所以要学会规避
SELECT * FROM shop_user WHERE pwd <> 'SALES';

#先写成 select，保证 where 条件的正确性，执行后同时可以确定数据
UPDATE shop_user WHERE pwd <> 'SALES';

#改写前面部分为 update
UPDATE shop_user SET NAME = '包包', pwd = '123' WHERE pwd <> 'SALES';

#根据查询的结果知道了字段和字段名，然后写 set 部分
```

# 18.3delete

语法：

```
delete from 表 where 条件
```

案例：

```
delete from shop_user where name is null;
```

注意：

```
#1、删除数据除了 delete 还可以用 truncate（截断，清空）表

truncate table shop_user;


#2、面试时，喜欢问：delete 和 truncate 的区别?
delete：可以带条件，可以删除部分数据，必须要提交 commit 后才能生效，删大数据慢
truncate：不能带条件，只能全部删除数据，不需要 commit，删大数据快

#3、如果又想快，有删部分，用存储过程，每删除 1000 条就提交一次(折中方案)

#4、删除时和 update 的规则一样(先 select 后改写成 delete)

#5、删除时有依赖关系，要先删除引用表，再删主表
```

# 20. 工作场景

作为测试，搭建/维护测试环境中，必不可少的会接触的数据的导入和导出，所以这部分知识也是你必须要完全掌握的

# 21. 导入导出

## 命令模式

## 导出

语法：

```
mysqldump -u 用户名 -p 数据库名 > 导出的文件名
```

案例：

```
mysqldump -u root -p mybatis>d:\person.sql
#输入后会让你输入进入 MySQL 的密码
#mybatis 为数据库名
```

### 21.1.2 导入

```
1、进入 MySQL：
        mysql -u 用户名 -p
2、 输入：
        use 目标数据库名
3、导入文件：source 导入的文件名;
        source d:/person.sql;
```

## 21.2 图形模式

### 导出

选择某数据库----右键----运行 sql 文件----选择需要导出的目录和输入名称----确定

### 21.2.2 导入

选择某数据库----右键----转储 sql 文件----选择需要导出的目录和输入名称----确定

# 23. 数据库灵魂问题

## 23.1 你都了解哪些数据库

面试官，我最熟悉的是 mysql。因为项目就用了 mysql，其次 oracle 和 mssql 也知道一些
非常建议你：下来查询 3 大数据库的一些区别和不同

## 23.2 Oracle（或是问 mysql）你都了解那些？

面试官，工作中主要是：
1、 连上服务器数据库
2、 导入/导出数据
3、 查询（单表、多表），其中左右连必说
4、 分页（limit）
5、 增删改（insert、update、delete）
建议：除门道教授的掌握外，你可以再了解下索引、视图

## 23.3 什么情况下会用到 Oracle（mysql）数据库

1、 环境搭建会用到 mysql 数据库
2、 请认真看该文档中我描述的每个工作场景
其他（如你能说清楚，能加分）：
3、 环境迁移时，把 A 服务器数据库移到 B 服务器
4、 数据库升级时，我们要关注一下前端数据和升级前的一致

5、 接口测试时，比如要用到用户 id，但是不知道 id 多少就要去数据库查一下