wangweichao@tedu.cn

## MySQL概述

1. 什么是数据库

数据库是存储数据的仓库

1. 用数据库的企业

金融机构、游戏网站、购物网站、论坛网站……

1. 提供数据库服务的软件

软件分类：

MySQL、SQL\_Server、Oracle、DB2、MongoDB、Mariadb

1. 在生产环境中，如何选择使用哪个数据库软件
2. 是否开源(开源!=免费)

开源软件：MySQL、MongDB、Mariadb

商业软件：SQL\_Server、Oracle、MongDB、DB2

1. 是否跨平台

不跨平台：SQL\_Server

跨平台：MySQL、MongDB、Mariadb、Oracle、DB2

1. 公司类型

商业软件：政府部门，金融机构

开源软件：游戏网站、购物网站、论坛网站

1. MySQL特点

关系型数据库：核心内容是关系，即二维表

特点

1. 行和列形式存储（表）
2. 表中每一行叫一个记录，每一列叫一个字段
3. 表和表之间的逻辑关联叫做关系

跨平台：可以在Unix、Linux、Windows上运行数据库服务

支持多种编程语言：python、java、php......

1. 数据库软件、数据库、数据库仓库的区别
   1. 数据库软件：

是一种软件，可以看得见，可操作，用来实现数据库逻辑的功能

* 1. 数据库：

是一种逻辑概念，用来存储逻辑的仓库，侧重于存储

* 1. 数据仓库

从数据量来说，数据仓库要比数据库庞大的多，主要用于数据挖掘和数据分析

## MySQL安装

1. Ubuntu安装
2. 安装服务端：sudo apt-get instal mysql-server
3. 安装客户端：sudo apt-get instal mysql-client
4. 启动和连接
5. 服务端启动：

sudo /etc/init.d/mysql start

sudo /etc/init.d/mysql status | stop | restart

1. 客户端连接

命令格式：mysql -h主机地址 -u用户名 -p密码

示例：mysql -hlocalhost -uroot -ptarena

本地登录可省略-h选项

## SQL命令使用规则

1. 使用规则：
   1. 必须以;结尾
   2. 不区分字母大小写
   3. \c来终止当前命令执行
2. 库的管理
   1. 库的基本操作
      1. 查看已有的库：show databases;
   2. 创建库（指定字符集,默认为latin1）
      1. create database 库名 [character set utf8];
   3. 查看创建库的语句
      1. show create database 库名;
   4. 查看当前所在库
      1. select database();
   5. 切换库
      1. use 库名;
   6. 查看库中已有的表
      1. show tables;
   7. 删除库
      1. drop database 库名;
3. 库的命名规则
   1. 可以使用数字、字母、下划线，但是不能为纯数字
   2. 库名区分字母大小写
   3. 库名具有唯一性
   4. 不能使用特殊字符和mysql关键字
4. 表的管理
   1. 表的基本操作
      1. 创建表
         1. create table 表名(字段名1 数据类型,字段名2 数据类型，…)[character set utf8];
         2. 查看创建表的语句（字符集、存储引擎）:

show create table 表名;

* + - 1. 查看表结构：desc 表名;
      2. 删除表：drop table表名;
      3. 修改表名：alter table 表名 rename 新表名；
      4. 注意：
         1. 所有的数据都是以文件的形式存放在数据库目录下
         2. 数据库目录：/var/lib/mysql

1. 表记录的管理
   1. 在表中插入记录（两种方式）
      1. insert into 表名 values(值1),（值2）,……;
      2. insert into 表名（字段名1，字段名2，……）values（值1），（值2），……；
   2. 查询表的记录（两种方式）
      1. select \* from 表名[where 条件];
      2. select 字段名1，字段名2，…… from 表名 [where条件]
2. 如何修改库的默认字符集
   1. 方法：通过更改mysql的配置文件实现
   2. 步骤
      1. 获取root权限：sudo –i
      2. 备份MySQL的配置文件：
         1. cd /etc/mysql/mysql.conf.d/
         2. cp –p mysql.cnf mysql.cnf.bak(-p可保证文件权限不改变)
      3. 修改配置文件：vi mysqld.cnf,在mysqld下添加一句话：character\_set\_server = utf8
      4. 重启MySQL服务/重新加载配置文件（reload）

/etc/init.d/mysql restart/force-reload

* + 1. 创建库验证是否修改成功

1. 客户端把数据存储到数据库服务器上的过程
   1. 连接到数据库服务器mysql –h …-u…-p…
   2. 选择库 use 库名
   3. 创建表/修改表
   4. 断开与数据库服务器的连接 exit/quit/ \q
2. mysql数据类型
   1. 数值类型
      1. 整型
         1. int 大整型（4个字节） 取值范围：0—2\*\*32-1（42亿多）
         2. tinyint 微小整型（1个字节）
            1. 有符号（signed默认）：-128—127
            2. 无符号（unsigned）：0-255 例如：age tinyint unsigned
         3. smallint 小整型（2个字节）
         4. bigint 极大整型（8个字节） 0-2\*\*64-1
      2. 浮点型
         1. float（4个字节，最多显示7个有效位）
            1. 用法：字段名 float（m,n）m：总位数 n：小数位数

如：float（5,2）取值范围：-999.99—999.99

* + - 1. double（8个字节，最多显示15位有效位）
      2. decimal（最多显示28个有效位）
         1. 字段名 decimal（m,n）
         2. 存储空间（整数部分和小数部分分开存储）

规则：将9位数字的倍数包装成4个字节

即：对于每个部分，需要4个字节来存储9位数的每个倍数，剩余数字所需的存储空间如下：

剩余数字 字节

0 0

1-2 1

3-4 2

5-6 3

7-9 4

示例：decimal（19,9）🡪一共9字节

整数部分：10/9 余1 🡪4字节+1=5字节

小数部分：9/9 余0 🡪4字节+0=4字节

* 1. 字符类型
     1. char（定长）
        1. 宽度取值范围：1-255
        2. 不给定宽度，默认为1
     2. varchar（变长）
        1. 取值范围：1—65535
     3. char和varchar的特点
        1. char：浪费存储空间，但是性能高
        2. varchar：节省存储空间，但是性能低
     4. Text/longtext（4G）/blob/longblob（4G）
     5. 字符类型的宽度和数值类型的宽的的区别
        1. 数值类型的宽度为显示宽度，仅仅用于select查询时显示，和占用的存储空间大小无关，可用zerofill查看效果
        2. 字符类型的宽度超过则无法存储
  2. 枚举类型（字段值只能在列举的范围内选择）
     1. 单选：字段名 enum（值1，值2，…，值N） 最多65535个不同值
     2. 多选：字段名 set（值1，值2，…，值N） 最多64个不同值

插入记录时：set（‘reading’，‘singing’，‘music’）

* 1. 日期时间类型
     1. year : 年 yyyy
     2. date : 日期 yyyymmdd
     3. time : 时间 hhmmss
     4. datetime、timestamp : 日期时间 yyyymmddhhmmss

插入记录时，datetime不给值，默认返回NULL，而timestamp字段默认返回系统当前时间

1. 日期时间函数
   1. NOW（）返回服务器当前日期时间yyyy:mm:dd hh:mm:ss
   2. CURDATE （）返回服务器当前日期 yyyy:mm:dd
   3. CURTIME（）返回服务器当前时间hh:mm:ss
   4. year(时间) 返回指定时间的年份 yyyy
   5. date（时间）返回指定时间的日期 yyyy-mm-dd
   6. time（时间）返回指定时间的时间
2. 日期时间运算
   1. 语法格式

select … from 表名 where 字段名 运算符 时间interval 时间间隔单位;

interval是间隔类型关键字

时间间隔单位：1day 、2 hour/minute/year/month

* 1. 示例：select \* from tab1 where meeting > now() – interval 1 day;

1. 表字段的操作
   1. 语法：alter table 表名 执行动作;
   2. 添加字段（ADD）：

alter table 表名 add 字段名 数据类型；

alter table 表名 add 字段名 数据类型 first; 加在第一列

alter table 表名 add 字段名 数据类型 after 字段名；加在指定字段的后面

* 1. 删除字段（drop）

alter table 表名 drop 字段名；

* 1. 修改字段数据类型（modify）

alter table 表名 modify 字段名 新数据类型；

#修改数据类型会受到表中原有数据的限制

1. 表记录的管理（删除、更改）：
   1. 删除记录（delete）
      1. delete from 表名 where 条件；
      2. 注意：delete后如果不加where条件，将删除表中所有记录
   2. 更改表记录（update）：
      1. update 表名 set 字段1=值1，字段2=值2 where 条件；
      2. 注意：如果不加where条件，将会将表中所有记录更新
2. 运算符操作
   1. 数值比较&字符比较
      1. 数值比较运算符：=、!=、>、>=、<、<=
      2. 字符比较运算符：=、!=
   2. 逻辑比较
      1. and(两个或多个条件同时满足）
      2. or（两个或多个条件有一个条件满足即可）
   3. 范围内比较
      1. 运算符

where 字段名 between 值1 and 值 2 #值为数字

where 字段名 in（值1 ，值2…）

where 字段名 not in（值1， 值2…）

* 1. 匹配空、非空
     1. 空：is null
     2. 非空：is not null
     3. 注意：
        1. null：空值，必须用is或is not 去匹配
        2. “”：空字符串，只能用 = 或 != 去匹配
  2. 模糊比较
     1. 语法格式：where 字段名 like 表达式
     2. 表达式
        1. \_：匹配单个字符
        2. %：匹配0到多个字符（null除外）
        3. 示例：

select name from sanguo where name like ‘\_%\_’（匹配至少两个字符的记录）

select name from sanguo where name like ‘赵%’（匹配姓赵的记录）

## SQL查询

* 1. 总结（执行顺序）

3、select ...聚合函数 from 表名

1、where ...

2、group by ...

4、having ...

5、order by ...

6、limit ...

* 1. order by（给查询结果进行排序）
     1. 语法：order by 字段名 排序方式
     2. 排序方式：
        1. ASC（默认）：升序
        2. DESC：降序
  2. limit（【分页】放在语句的最后）
     1. 作用：限制显示查询记录的个数
     2. 用法：
        1. limit n ---->显示n条记录
        2. limit m，n ---->从第m+1条记录开始，显示n条记录

limit 2,3 -->显示第3，4，5条记录

* 1. 聚合函数
     1. 分类
        1. avg（字段名）：求该字段的平均值
        2. sum（字段名）：求该字段的和
        3. max（字段名）：求该字段的最大值
        4. min（字段名）：求该字段的最小值
        5. count（字段名）：统计该字段记录的个数
        6. 示例：select max(gongji) as best from sanguo;（as后加别名）
     2. 空值null不会被统计，‘’会被统计
  2. group by（分组）
     1. 作用：给查询结果进行分组
     2. 示例：select country,avg(gongji) from sanguo group by country;

mysql> select country,count(\*) from sanguo group by country order by count(\*) desc limit 2;

+---------+----------+

| country | count(\*) |

+---------+----------+

| 蜀国 | 4 |

| 魏国 | 3 |

+---------+----------+

* + 1. 注意：
       1. group by 后的字段名必须要为select之后的字段名，如果查询的字段和group by之后的字段不一致，则必须要对该字段进行聚合处理（聚合函数）
  1. having
     1. 作用：对查询的结果进行进一步的筛选
     2. 示例：

mysql> select country,avg(gongji) from sanguo group by country having avg(gongji) > 105 order by avg(gongji) desc limit 2;

+---------+-------------+

| country | avg(gongji) |

+---------+-------------+

| 蜀国 | 578.2500 |

| 魏国 | 328.3333 |

+---------+-------------+

* + 1. 注意：
       1. having语句通常与group by语句联合使用，用来过滤由group by语句返回的结果集
       2. where 语句只能操作表中实际存在的字段，having操作的是由聚合函数生成的显示列
  1. distinct（不显示字段的重复值）
     1. 用法：select distinct 字段名1，字段名2，… from 表名；
     2. 示例：select count(distinct name) as number from sanguo where country='蜀国';
     3. 注意
        1. distinct处理的是distinct和from之间的所有字段，所有字段的值必须相同才可以去重
  2. 查询表记录时做数学运算
     1. 运算符：+ - \* / %
     2. 示例：select id,name,gongji\*2 from sanguo ;

## 约束

* 1. 作用：为了限制无效的数据插入数据表中
  2. 约束分类
     1. 默认约束（default）
        1. 作用：插入记录时，不给该字段赋值，使用默认值
        2. 格式：字段名 数据类型 default 值
     2. 非空约束（not null）
        1. 作用：不允许该字段的值有NULL记录
        2. 格式：字段名 数据类型 not null

## 索引（index）

* 1. 定义：对数据库中表的一列或者多列的值进行排序的一种结构（mysql中索引用BTREE方式）
  2. 优点：可以加快数据的检索速度
  3. 缺点
     1. 当对表中的数据进行增加、删除和修改的时候，索引需要动态维护，降低了数据的维护速度
     2. 索引需要占用物理空间
  4. 索引示例
     1. 开启运行时间检测：
        1. show variables like ‘%pro%’;
        2. set profiling=1;
     2. 执行查询语句：select name from t1 where name=’lucy5555’;
     3. 查看执行时间：show profiles;
     4. 在name字段创建索引：create index index\_name on t1(name);
     5. 执行查询语句：select name from t1 where name=’lucy554612
     6. 查看执行时间：show profiles;
  5. 索引类型
     1. 普通索引（index）
        1. 使用规则：
           1. 一张表中可以有多个index字段
           2. 字段的值可以重复，且可以为null
           3. 经常把做查询条件的字段设置为index字段
           4. index字段的key标志为MUL
        2. 创建index
           1. 创建表时创建普通索引

create table 表名(……， index(字段名1)，index（字段名2）);

示例：create table t1( id int, name varchar(20), age tinyint, index(id),index(name) );

* + - * 1. 在已有表中创建普通索引

create index 索引名 on 表名（字段名）;

* + - 1. 查看索引
         1. desc 表名;
         2. show index from 表名;
      2. 删除inidex：drop index 索引名 on 表名；
         1. 注意：删除index只能一个一个删
    1. 唯一索引（unique）
       1. 使用规则
          1. 一个表中可以有多个unique字段
          2. 对应字段的值不允许重复，但可以为null
          3. key标志：UNT
       2. 创建
          1. 创建表时创建：… ， unique（字段名1），unique（字段名2））；
          2. 在已有表中创建：create unique index 索引名 on 表名（字段名）；
       3. 查看/删除：同index
    2. 主键索引（primary key）&& 自增长属性（auto\_increment）
       1. 使用规则
          1. 一个表中只能有一个字段设置为主键
          2. 对应字段值不允许重复，且不能为null
          3. key标志：PRI
          4. 把表中能够唯一标识一条记录的字段设置为主键，通常把记录编号的字段设置为主键
       2. 创建：
          1. 创建表时创建：….id int primary key auto\_increment,….);或…,primary key(id);
          2. 在已有表中创建：alter table 表名 add primary key（字段名）；
          3. 删除主键：

如有自增长属性，先删除自增长属性：

alter table 表名 modify 字段名 数据类型

删除主键：Alter table 表名 drop primary key;

* + - 1. 在已有表中添加自增长属性并制定起始值
         1. 添加自增长属性：alter table 表名 modify字段名 数据类型 auto\_indrement
         2. 指定起始值：alter table 表名 auto\_increment = 起始值
    1. 外键索引（foreign key）
       1. 定义：让当前表的字段值在另一个表的范围内选择
       2. 语法格式：create…… foreign key（参考字段名）

reference 被参考表名（被参考字段名）

on delete 级联动作

on update 级联动作

* + - 1. 示例：让表二的信息只能从表一中选择

表一：缴费信息表（财务）

学号 姓名 班级 金额

1 linda 2 200

2 tom 2 300

表二：班级信息表（班主任）

学号 姓名 金额

1 linda 200

2 tom 300

create table stu(

-> id int,

-> name varchar(20),

-> money int,

-> foreign key(id) reference money(id) on delete cascade on update cascade

-> );

* + - 1. 级联动作
         1. cascade：数据级联更新：当主表删除记录或更改被参考字段值时，从表级联更新
         2. restrict：默认级联动作，当主表删除记录或更改被参考字段值时，从表中有相关记录则不允许主表操作
         3. set null：当当主表删除记录或更改被参考字段值时，从表中有相关记录的字段值变为null
         4. no action：同restrict，都是立即检查外键限制
      2. 删除外键限制：
         1. alter table 表名 drop foreign key 外建名 ；
         2. 外建名查看方式：show create table 表名；
      3. 在已有表中添加外建限制：alter table 表名 add foreign key(…) references 表名（字段名）on delete 级联动作 on update 级联动作
      4. 使用规则
         1. 两张表参考字段和被参考字段的数据类型要一样
         2. 被参考字段必须为key的一种，通常为primary key

## 数据导入

* 1. 作用：文件系统（U盘等）的内容导入到数据库中
  2. 语法格式：load data infile‘文件名’ into table 表名 fields terminated by ‘分隔符’（表格分隔符默认为逗号）lines terminated by ‘\n’
  3. 操作步骤：
     1. 在数据库中创建对应的表
     2. 把要导入的文件拷贝到数据库的搜索路径中
        1. 查看数据库搜索路径：show variables like ‘secure\_file\_priv’
        2. 执行复制语句：sudo cp 要导入的文件路径 /var/lib/mysql-files
     3. 执行数据导入语句：

load data infile '/var/lib/mysql-files/AID1709.csv' into table stuinfo

-> fields terminated by ','

-> lines terminated by '\n';

## 8. 数据导出

* 1. 作用：把数据库中表记录保存到系统文件里
  2. 语法格式：select … from 表名 into outfile ‘文件名’fields

terminates by ‘分隔符’

lines terminates by ‘\n’;

* 1. 示例：select user,host from user

-> into outfile '/var/lib/mysql-files/user.csv'

-> fields terminated by ','

-> lines terminated by '\n';

* 1. 注意：
     1. 导出的内容由SQL查询语句决定
     2. 执行导出命令时路径必须指定对应的数据库目录下
     3. ### chmod 777 文件名
     4. ### chmod +rw 文件名

## 表的复制

* 1. 复制语法格式：create table 表名 select … from 表名；
  2. 示例：create table tab2 select \* from tab1；
  3. 只复制表结构：create table 表名 select … from 表名 where false；

## 嵌套查询（子查询）

* 1. 定义：把内层的查询结果作为外层查询的条件
  2. 语法：
     1. sql查询语句 where 条件（sql查询语句）
  3. 示例
     1. 把uid的值小于平均值的用户名和uid号显示出来

select name,uid from userinfo where uid <(select avg(uid) from userinfo);

* + 1. 找出每个国家中攻击力最高的英雄的名字和攻击值

select name,gongji from sanguo where gongji in (select max(gongji) from sanguo group by country);

1. 优化
   1. 单表查询
      1. 应尽量避免全表扫描，对于经常用作查询条件的字段做索引
      2. 如果有排序、分组相关的查询，则可以做联合索引
   2. 多表查询
      1. 连接查询时，小表会全表查询，所以索引没用，在大表使用索引，没有where子句，那么就按表的行数来定，如果有where子句，那么就按条件过滤完的行数来定大小表
      2. 用连接查询代替子查询，子查询会产生中间表，效率低
2. 索引失败
   1. like后面的通配符%在前面，索引会失效：like ‘F%’不会失效，like ‘%F’会失效
   2. 没有使用联合索引的第一列，not in，!=，使用MySQL函数，类型转换，or等都无法用到索引
3. 优化总结：
   1. 对查询进行优化，应尽量避免全表扫描，首先应考虑在 where和order by 涉及的列上建立索引
   2. 应尽量避免在 where 子句中使用or 来连接条件，否则将导致MySQL放弃使用索引而进行全表扫描
   3. in和 not in 也要慎用，否则会导致全表扫描，对于连续的数值，能用 between 就不要用 in
   4. 不要在 where 子句中的“=”左边进行函数、算术运算或其他表达式运算，否则系统将可能无法正确使用索引。如：select id from t where num/2=100应改为:select id from t where num=100\*2
   5. 在使用索引字段作为条件时，如果该索引是复合索引，那么必须使用到该索引中的第一个字段作为条件时才能保证系统使用该索引，否则该索引将不会被使用，并且应尽可能的让字段顺序与索引顺序相一致
   6. 尽量使用数字型字段，若只含数值信息的字段尽量不要设计为字符型，这会降低查询和连接的性能，并会增加存储开销
   7. 尽可能的使用 varchar/nvarchar 代替 char/nchar ，因为首先变长字段存储空间小，可以节省存储空间，其次对于查询来说，在一个相对较小的字段内搜索效率显然要高些