数据库

一、数据库基本概念

数据库是存放数据的仓库。它的存储空间很大，可以存放百万条、千万条、上亿条数据。但是数据库并不是随意地将数据进行存放，是有一定的规则的，否则查询的效率会很低。

常见的数据库：

- Oracle - 目前世界上使用最为广泛的数据库管理系统，作为一个通用的数据库系统，它具有完整的数据管理功能；作为一个关系数据库，它是一个完备关系的产品；作为分布式数据库，它实现了分布式处理的功能。在Oracle最新的12c版本中，还引入了多承租方架构，使用该架构可轻松部署和管理数据库云。
- DB2 - IBM公司开发的、主要运行于Unix（包括IBM自家的AIX）、Linux、以及Windows服务器版等系统的关系数据库产品。DB2历史悠久且被认为是最早使用SQL的数据库产品，它拥有较为强大的商业智能功能。
- SQL Server - 由Microsoft开发和推广的关系型数据库产品，最初适用于中小企业的数据管理，但是近年来它的应用范围有所扩展，部分大企业甚至是跨国公司也开始基于它来构建自己的数据管理系统。
- MySQL - MySQL是开放源代码的，任何人都可以在GPL（General Public License）的许可下下载并根据个性化的需要对其进行修改。MySQL因为其速度、可靠性和适应性而备受关注。
- PostgreSQL - 在BSD许可证下发行的开放源代码的关系数据库产品。

MySQL简介

MySQL最早是由瑞典的MySQL AB公司开发的一个开放源码的关系数据库管理系统，该公司于2008年被昇阳微系统公司（Sun Microsystems）收购。在2009年，甲骨文公司（Oracle）收购昇阳微系统公司，因此在这之后MySQL成为了Oracle旗下产品。

MySQL在过去由于性能高、成本低、可靠性好，已经成为最流行的开源数据库，因此被广泛地应用于中小型网站开发。随着MySQL的不断成熟，它也逐渐被应用于更多大规模网站和应用，比如维基百科、谷歌（Google）、脸书（Facebook）、淘宝网等网站都使用了MySQL来提供数据持久化服务。

甲骨文公司收购后昇阳微系统公司，大幅调涨MySQL商业版的售价，且甲骨文公司不再支持另一个自由软件项目OpenSolaris的发展，因此导致自由软件社区对于Oracle是否还会持续支持MySQL社区版（MySQL的各个发行版本中唯一免费的版本）有所担忧，MySQL的创始人麦克尔·维德纽斯以MySQL为基础，成立分支计划MariaDB（以他女儿的名字命名的数据库）。有许多原来使用MySQL数据库的公司（例如：维基百科）已经陆续完成了从MySQL数据库到MariaDB数据库的迁移。

- mysql的安装

```
sudo apt install -y mysql-server mysql-client

#首次安装MySQL，如果安装过程中没有提示输入密码，可以使用下面的方法来找到默认的初始密码。
cd /etc/mysql
sudo vim debian.cnf
然后找到：
user     = xxx
password = xxx
登录的时候使用这个用户和密码，进去后再改密码
```

在实际开发中，为了方便用户操作，可以选择图形化的客户端工具来连接MySQL服务器，包括：

- MySQL Workbench（官方提供的工具）
- Navicat for MySQL（界面简单优雅，功能直观强大）
- SQLyog for MySQL（强大的MySQL数据库管理员工具）

二、SQL

基本可分为：

- 数据定义语言DDL (create、drop）
- 数据操作语言DML（insert、delete、update）

- 数据查询语言DQL（select、where、group by、order by 、limit）
- 数据控制语言DCL（grant、revoke）
- 事务处理语言TPL（commit、rollback）

三、操作数据库

- 连接mysql数据库的命令

```
命令:
    mysql -h服务器地址 -u用户名  -p  #不要再p后面直接跟密码
```

  - 忘记密码

```
1.到/etc/mysql/mysqld.conf.d/,编辑mysqld.conf
sudo vim mysqld.conf
2.到[mysqld]下
添加：skip-grant-tables
保存退出
3.重启mysql服务：
   sudo service mysql restart | stop | start
4.重新登录mysql，这个时候不需要密码，登录进去后：
use mysql;
update user set authentication_string=PASSWORD("输入你想设置的密
码") where user='root';
update user set plugin="mysql_native_password";
flush privileges;
5.重启mysql服务
sudo service mysql restart
6.以root和新密码重新登录
```

  - 远程登录

```
如果要远程连接mysql数据库，需要以下步骤：
1.通过mysql -u用户名  -p登录mysql数据库
2.创建一个新用户来远程连接
     create user 'python'@'%' identified by '123'
     GRANT ALL PRIVILEGES ON *.* TO 'python'@'%' IDENTIFIED BY
'123' WITH GRANT OPTION;
3.执行命令：flush privileges;
4.退出mysql，到/etc/mysql/mysql.conf.d下，编辑mysqld.cnf
    找到 “bind-address = 127.0.0.1” ，这一行要注释掉，只需在前面加个
#
5.重启mysql服务
  sudo service mysql restart | start | stop  #重启、启动、停止
  或者：sudo /etc/init.d/mysql restart | start | stop
6.如果开启了防火墙，请添加3306端口
  sudo ufw allow 3306
```

- 数据库操作命令

```
#1.查看库
   show databases;
#2. 创建库
   create database 数据库名 default charset=utf8;# 数据库名不要纯数
字，不要用汉字
#3. 删除库
   drop database 数据库名;
#4. 选中库
   use 数据库名;
#5. 查看表
   show tables;
#6.查看数据库创建语句
show create database 数据库名
#7.查看选中的数据库
select database()
#8 修改数据库字符集
alter database student default charset=utf8;
```

- 注意
  - 每条命令结束必须使用; 或者 \g 结束
  - 退出mysql使用命令quit或exit

四、数据库表

- 创建表

```
create table [if not exists] 表名(
   列名1  类型  [限制],
   列名2  类型  [限制],
   ...
   列名n  类型  [限制]  #最后一列没有逗号
) [engine=myisam | innodb][ default charset=utf8];

primary key 主键 不允许有重复值,不允许为空
auto_increment 自增长，只对int型主键起作用

#复合主键
mysql> create table grade(
    sid int ,
    cid int,
    score float,
    primary key(sid,cid));
```

- 删除表

```
drop table 表名;
```

- 复制表结构

```
create table 表名 like 其他表名
```

- 查看表结构

```
desc 表名;
```

- 查看建表语句

```
show create table 表名;
```

- 修改表

```
#修改字段类型
```

```
   alter table 表名 modify 字段名 类型 [限制]
#增加字段
   alter table 表名  add [column] 字段名 类型 [限制];
#删除字段
   alter table 表名 drop [column] 字段名;
修改字段名和类型
   alter table 表名 change [column] 旧字段名 新字段名 类型 [限制];

#修改表名
  alter table 表名 rename 新表名
  alter table 表名 [engine=myisam] [default charset=utf8];


 # 可以通过first、after指定插入位置
  alter table student add sno varchar(3) not null after sid; //在
sid列后插入
   alter table student add sid int primary key auto_increment
first;//在第一列插入
```

- 字段限制

```
primary key 不允许空值，唯一
not  null 非空
unique 唯一
default  缺省,默认值
```

五、数据类型

- 数值型
  - 整型 能用整型尽量使用整型。包括int、smallint tinyint
  - int(3) 或者 tinyint(2) :3或者2不会去限制你所存储数据的长度， 只有在配合zerofill 零填充的时候 才有意义
  - 浮点数 double 、decimal

类型	大小	范围(有符号)	范围(无符号)	用途
tinyint	1字节	-128-127	0,255	最小整数
int	4字节	-2147483648-2147483647		大整数值
float(m,n)	4-8字节			单精度浮点型(浮点数)
double(m,n)	8字节			双精度浮点型(浮点数)
decimal(m,n)	变长			浮点数(更加精确)

- 字符型

类型	大小	用途
char	0-255字节	存储定长的字符串
varchar	0-65535字节	变长字符串
text	0-65535字节	长文本数据
blob	0-65535字节	二进制的文本(不建议)
enum('w','m')	65535个成员	枚举
set('w','m')	64个成员	集合

注意：

(1) char 和 varchar 的区别:

- char的执行效率高于varchar ， varchar 相对于 char 节省存储空间
- 如果使用char 传入的数据的长度 小于指定的长度的时候 存储的实际长度不够的会拿空格来填充
- 如果使用 varchar 传入的数据的长度 小于指定的长度的时候 存储的实际

长度 为传进来的数据的长度

- 日期时间型

类型	大小	范围	格式	用途
date	3	1000-01-01/9999-12-31	YYYY-MM-DD	日期值
time	3	-838:59:59/838:59:59	HH:MM:SS	时间值
year	4	1901/2155	YYYY	年份值
datetime	8	1000-01-01 00:00:00/9999-12-31 23:59:59	YYYY-MM-DD HH:MM:SS	混合日期和时间
		‘2018-05-4 10:53:00’		

- 枚举enum

```
#是自定义类型，可以多选一,实际上存的值是1，2，3...
alter table user add sex enum('男','女') default '男';
insert into user(name,password,sex)
values('tom','132','男');
values('tom','132',1);
```

- 集合set

```
类似复选框，可以存多个值
alter table student add hobby set('看电影','玩游戏','敲代码','烫
头')
insert into users(uid,hobby) values(22,1+2+4+8)
insert into users(uid,hobby) values(22,1|2|4|8)
insert into users(uid,hobby) values(22,'看电影,玩游戏,敲代码')

0001
0010
-------
0011
```

六、数据操作

1. insert

```
写法一：insert into 表名(字段1，字段2...) values(值1,值2...);

省略了字段列表，则按照建表时的字段顺序进行插入，每一列都要给值
写法二：insert into 表名 values(值1,值2...);
写法三：插入多个记录
       insert into 表名(字段1，字段2...)
             values(值1,值2...),
             (值1,值2...),
             (值1,值2....
写法四：  insert into 表名(name,age,sex)
         select name,age,sex from stars;

         insert into histroy_student select * from student;
```

2.update

```
 update 表名 set 字段1=值1,字段2=值2... where 条件  #不加where修改的是所有的记录
```

3. delete

```
删除表中的数据，自增主键的值不会重新开始
delete from 表名 where 条件；#如果不加条件，会删除表中所有数据,慎重使用
alter table 表名 auto_increment = 5  # 设置自增主键开始值
清空表，自增主键的值重新开始编号
truncate
       truncate table 表名,清空表中所有记录，等价于delete from 表名;
       delete和truncate差别，truncate后，表中自增主键值从1开始
```

七、数据查询

基本结构： select 字段名列表 from 表名

1 基础查询

```
select username,password  from user;
select usernname as 用户名, password as 密码  from user;  #可以给字段起别名
select *  from user; #查询所有字段，慎用，一般不建议使用，会导致无法优化sql语句
select 2018,username,password  from user; #可以有常量，表达式
select sname,2018-year(sbirthday) from student; #year是mysql的内置函数
select distinct username  from  user; #去除重复记录 distinct 针对查询结果去除重复记录，不针对字段
```

2 条件查询（where）

- 关系运算

  关系运算符：> 、 >=、 <、 <=、 =、!=、<>、 between and

  ```
  select username,password from user where uid <10
  select username,password from user where uid != 10
  select username,password from user where uid between 10 and 20
  ```

- 逻辑运算

  逻辑运算符：and 、or、not

  ```
  select username,password from user where uid < 100 and uid > 20;
  select username,password from user where uid > 100 or uid < 20;
  ```

- 集合运算

  集合运算符：in、not in

  ```
  select username,password form user where uid in (2,3,4)
  select username,password form user where uid not in (2,3,4)
  ```

- 判空

  判空运算：is null、is not null

  ```
  select username,password from user where username is null
  ```

- 字符串的模糊查询(like)

  通配符 _代表一个字符，%代表任意长度字符串

```
select * from user where username like '王_';
select * from user where username like '王%';
```

3. 排序（order by）

asc 升序(默认)、desc 降序、

```
select * from user order by age asc;
select * from user order by age desc;
多字段排序
    select name,age from php_user_history  order by age desc,name;#
如果在第一列上有相同的值，在具有相同的age的记录上再按name升序排列
```

4.限制结果集(limit)

limit n #取前n条记录

limit offset,n #从第offset条开始取，取n条

```
select * from php_user_history limit 3;
select * from php_user_history limit 4,2;
注意结果集中记录从0开始数数，offset相对于0开始
实现分页必须的技术点
limit (page-1)*num,num
```

5.集合函数

- count统计结果集中记录数
- max 最大值
- min 最小值
- avg 平均值，只针对数值类型统计
- sum 求和，只针对数值类型统计
- 注意，集合函数不能直接使用在where后面的条件里，但可以在子查询中

```
select count(*) num from user;
select count(distinct age) num from user; //去除重复记录
select * from student where sno = max(sno);//错误
```

6.分组（group by)

将结果集分组统计，规则：

- 出现了groub by的查询语句，select后面的字段只能是集合函数和group by后面有的字段，不要跟其它字段
- 对分组进行过滤，可以使用having

```
select uid, count(*) num from php_forum group by uid;
select uid,title, count(*) num from forum group by uid having
count(*) >=2;
having和where的区别：
    where针对原始表进行过滤
    having 是针对分组进行过滤
```

查询小结

- 整体顺序不能颠倒
- []表示可选，可以有也可以没有

  select 字段 from 表名 [where 条件][group by] [having][order by] [limit]

八 字符集和存储引擎

- 修改字符集

为了能够正常显示中文，必须把数据库的字符集设置为utf8.

```
mysql> show variables like 'character%';  #查看字符集
+--------------------------+----------------------------+
| Variable_name            | Value                      |
+--------------------------+----------------------------+
| character_set_client     | utf8                       |
| character_set_connection | utf8                       |
| character_set_database   | latin1                     |需要修改
| character_set_filesystem | binary                     |
| character_set_results    | utf8                       |
| character_set_server     | latin1                     |需要修改
| character_set_system     | utf8                       |
| character_sets_dir       | /usr/share/mysql/charsets/ |
+--------------------------+----------------------------+
8 rows in set (0.01 sec)
```

```
修改mysql的配置文件
cd /etc/mysql/mysql.conf.d
sudo cp mysql.cnf  mysql.cnf.bak
sudo vim mysql.cnf
在[mysqld]下增加一句：
character_set_server = utf8
保存并重启服务
sudo systemctl restart mysql.service  #重启服务
```

- 数据库引擎

  可以使用show engines命令查看数据库引擎

  ```
  show engines \G
  ```

下面的表格对MySQL几种常用的数据引擎进行了简单的对比。

特性	InnoDB	MRG_MYISAM	MEMORY	MyISAM
存储限制	有	没有	有	有
事务	支持			
锁机制	行锁	表锁	表锁	表锁
B树索引	支持	支持	支持	支持
哈希索引			支持	
全文检索	支持（5.6+）			支持
集群索引	支持			
数据缓存	支持		支持	
索引缓存	支持	支持	支持	支持
数据可压缩				支持
内存使用	高	低	中	低
存储空间使用	高	低		低
批量插入性能	低	高	高	高
是否支持外键	支持			

通过上面的比较我们可以了解到，InnoDB是唯一能够支持外键、事务以及行锁的存储引擎，所以我们之前说它更适合互联网应用，而且它也是较新的MySQL版本中默认使用的存储引擎。

- myisam和innodb的区别
    - myisam查询速度快，不支持事务、不支持外键、支持表锁
    - innodb增删改效率高，支持事务、支持外键，支持行锁