# DATABASE

## SQL

### 操作

#### 数据库选择

|  |  |
| --- | --- |
| show databases; | 显示所有数据库 |
| create database abc; | 创建数据库abc |
| use abc; | 使用数据库abc |

#### 表操作

|  |  |
| --- | --- |
| create table aa(id int primary key, name varchar(30),age int); | 创建表aa  id，name，age |
| desc aa； | 显示表aa中的字段名 |
| drop table aa; | 删除表aa |
| create table aa(id int primary key auto\_increment,data int);  insert into aa(data) values(33);  insert into aa(data) values(22); | 主键id自动递增 |
| begin  。。。。。。大量操作  commit | 加快操作  commit； 修改数据后提交，否则其他终端无法操作该数据 |

#### 插入

|  |  |
| --- | --- |
| insert into aa values(1,'n1',11); | 向表aa中插入一条数据 |
| insert into aa(id) values(4); | 向表aa中插入一条数据 |

#### 删除

|  |  |
| --- | --- |
| delete from aa where id=4 and name=’n1’; | 删除表aa中所有id为4,name为n1的数据 |
| truncate table aa; | 清空表aa中所有数据，比delete from aa；效率高。  delete不释放空间，truncate会彻底释放空间 |

#### 修改

|  |  |
| --- | --- |
| update aa  set age=55,name=’xx’  where id=1; | 修改表aa中的字段 |

#### 查找

|  |  |
| --- | --- |
| select \* from aa; | 选择aa中所有数据 |
| select name from aa; | 选择aa中所有name数据 |
| select name,age from aa; | 选择aa中所有name和age数据 |
| 条件查询 | |
| select \* from aa where id=1; | 选择aa中所有id为1的数据 |
| select min(id),name,age from aa where id>=5; | 选择aa中id>=5的数据中id最小的数据 |
| select \* from aa  where id between 5 and 30; | 查找id在[5,30]范围内的数据 |
| select \* from aa  where id in(5,10,15); | 查找id为括号内列举值的数据 |
| select distinct name from aa; | name字段相同时排除 |
| select \* from aa  order by id desc; | 查找的数据按照id从大到小排序（不写desc默认从小到大）  排序时，null作为最大值 |
| select \* from aa  where name like ‘+%+’; | 查找所有name以+开头，+结尾的字段【% 任意数量字符，\_ 单个字符】 |
| 子查询 | |
| select name,age from  (select \* from aa where id>15) as temp; | as temp作为()内查询结果的别名，虽然没有被使用但是必须 |
| 多表查询  自连接：一个表中放入了逻辑上两种不同的数据  内连接：符合where条件，数据就被选中，不符合where条件就被过滤  外连接：等于内连接加上匹配不上的记录（全部被选中） | |
| select aa.id,name from aa,cc  where aa.id=bb.id;  sql99写法：  select aa.id name from  aa inner join cc on aa.id=cc.id; | 从aa和bb两张表中进行查询，限制条件为两个表的id相同（显示字段时,如果该字段两个表都有，需要用aa.id这种语法区分） |
| select aa.id name from  aa left outer join cc on aa.id=bb.id; | 在满足aa.id=cc.id情况下，将aa里的数据全部也取出来 |
| select aa.id name from  aa right outer join cc on aa.id=cc.id; | 在满足aa.id=cc.id情况下，将cc里的数据全部也取出来 |
| select id from aa  union  select id from cc; | 将两个表的查询结果合并起来 |

### 功能

生成密码：password("abc")

获取密码字符：select password("abc") password

### sql执行顺序

from->where->group by->having->select->order by

在分组语句中，select后的字段必须是分组标准或经过组函数处理过

### 数据类型

#### 基本类型

number

number(7) 宽度7 不加则不限制宽度

number(7,2) 小数点后占两位，总长为7

char

char(7) 定长字符串 不够则补0，省时间

varchar2(7) 变长，可以数量不足，省空间

#### date类型

1. 默认格式 ‘dd-mon-yy’
2. sysdate

exp: insert into 表名 values(... ‘10-dec-13’);

或者将字符串写成sysdate

取出日期：

to\_char(要处理的日期，’日期格式’)

日期格式：yyyy，mm，dd，hh24，mi，ss

exp：

select to\_char(日期所在字段名，‘yyyy-mm-dd’) from 表名；

格式字符串中间的符号-可任意

day 星期

mon 三个字母的月的缩写

month 全写

pm 表示出pm或者am

放入任意日期：

to\_date(日期字符串，格式字符串)

exp：

to\_date(‘2010-10-10 13;10:25’,’yyyy-mm-dd hh24:mi:ss’)

格式字符串要与日期字符串保持一致

对日期进行调整：

按天进行调整：

select to\_char(日期所在字段名-1，’yyyy-mm-dd’) from 表名；

-1代表减少一天

1/(24\*60\*60) 代表一秒

add\_months(日期所在字段名，-1)

减少一个月

给定一个时间，得到这个时间对应月的最后一天的日期

select to\_char(last\_day(sysdate),’yyyy-mm-dd hh24:mi:ss’) from dual；

只有dd发生变化

next\_day(日期，’星期二’)

跳到下一个星期二

round默认以天为单位进行四舍五入，如果一天的时间过了一半，则变为新的一天（天后面的时间进行四舍五入，保留到天）

或者可以指定单位进行操作：

round(sysdate,’mm’)

trunc 默认以天为单位进行截取

如果以月为单位进行截取，则将小时，分秒清空，日期变为1

得到一个月的最后一天的最后一秒：

trunc(last\_day(sysdate)+1)-1/(24\*60\*60)

变成下个月的开始再减去一秒

### 运算符

is null 判断空

and

or

not

### 函数

单行函数：针对每一行数据都做处理，sql语句影响多行，数据就返回多少个结果

组函数：针对每一组数据进行处理，无论影响多少行，只返回一个结果

|  |  |
| --- | --- |
| upper() |  |
| lower() |  |
| initcap() | 把字符串每个单词的首字母变大写 |
| concat(‘s1’,’s2’) |  |
| length() |  |
| substr(s1,i,n) | s1为要截取的字符串，n代表截取的长度，i代表开始位置（-1代表最后一个字符，-3代表倒数第三个） |
| round | 四舍五入  round(x,0) 代表保留小数点后0位，默认（可不写） |
| trunc | 截取 |
| to\_char(字段，格式字符串) | 格式显示函数： 格式字符串可以省略，代表把数据变成字符串类型  fm 格式字符串开始  9 任意数字  0 强制显示前导0  $  L 本地货币符号  ， 分隔符号  . 小数点  to\_char(salary,”fm$099,999,00”) |
| 分组 | 常见组函数：count，max，min，sum，avg  select count(id),max(salary) from s\_emp;  显示id总量，salary最大值  组函数中可以使用distinct关键字  sum(distinct salary)  组函数对null的处理是忽略的 |
| 过滤 | 对组数据进行过滤（having）  select dept\_id,avg(salary) from s\_emp group by dept\_id having avg(salary)>2000;  只显示大于2000的 |

### 事务操作

update..........;

savepoint a; 设置断点a

update......;

savepoint b; 设置断点b

update......;

rollback to b; 撤销到b点的操作结果

### 数据库约束

对数据库中表的字段的值加一些限制和保护，是对数据保护的最后一道屏障。

主键约束：primary key 字段值不能为null，且不重复

一个表中只能有一个主键

唯一性约束：unique 字段值不能重复

非空约束：not null 字段值不能为null

检查约束：check 字段值必须符合检查条件

外键约束：references foreign key

on delete cascade

on delete set null

列级约束：在定义表的每一列时，只直接对这一列加约束

create table abc(id number primary key);

表级约束：在定义完表的所有列之后，再选择某些列加约束

如果没有给约束起名，系统会自动分配一个约束名

id number constraint xy primary key

自己给约束起名为xy，不满足条件时，报错时会出现xy

检查约束： id number check(id>10)

id加入时必须大于10

表级约束：

not null 无表级约束

1. create table abc(id number, salary number, constraint xy primary key(id, salary));只要有一个满足约束即可
2. create table abc(id number, salary number, primary key(id),not null(salary))

外键约束：外键字段的值引用自主表一个字段的值，受限于主表

|  |  |
| --- | --- |
| 1  2  主键字段 | 主表 |

|  |  |
| --- | --- |
| 1  2  null  1 | 从表 |

只能用主表中已存在的字段1，2和null

先建主表：create table abc(id number primary key);

后建从表：create table def(salary number, sid number

references abc(id));

插入数据时，先插入主表，才能插入从表

删除数据时，先删子表的，才能删除主表的

删除表时，先删除从表，再删除主表，除非使用：

drop table 从表 cascade constraints；

加s代表断开多个关系

级联删除：

在建从表时：

sid number references abc(id) on delete cascade

当删除主表某个字段时：

delete from abc where id=1；

从表中所有sid等于1行的都随之删除

级联置空：

使用 on delete set null 时

从表中所有sid等于1的行都将1变为null

### 数据库其他对象

#### 序列

生成主键的值

1. 创建序列

create sequence sq； sq为序列名

1. 测试序列

create sq.nextval from dual;

1. 删除序列

drop sequence sq；

1. 使用序列

create abc(id number primary key);

insert into abc values(sq.next.nextval);

sq.currval||’a’

可以将两个字符串拼接成一个字符串

没调用一次sq.nextval,sq的值就变化

1. 复杂序列

create sequence sq MAXVALUE 9999;

增加额外限制信息

#### 索引

加速查询

1. 建立索引

主键和唯一键上，系统会自动建立索引

create index 索引名 on 表名(字段名)；

1. 删除索引

drop index 索引名；

#### 视图

本质对应的是一条sql语句，相对于视图对应的真实数据，视图的空间可以忽略不计

可以对同一份物理数据做不同表现

建立视图：

create or replace view 视图名 as select id，first\_name from s\_emp;

为一条sql语句建立一个视图

视图 select \*from 视图名 where id=2；

删除 drop view 视图名；

#### 分页技术

rownum伪列，用来计数

select rownum,id,first\_name from s\_emp where rownum<12;

不能使用 rownum<22 and rownum>12，因为从rownum=1开始运行，一开始不满足就退出

使用：

select \* from(select rownum r, id from s\_emp where rownum<22)where r>12;

### sql三范式

1. 表中的字段不可分

|  |  |  |
| --- | --- | --- |
| 字段1 | 字段2 | |
|  | 字段3 | 字段4 |
|  | |

不能将字段2进行拆分

1. 表中所有非空属性必须依赖主属性
2. 在第二范式的基础上消除了传递依赖

如表中有员工和员工所属部门，如果放在同一张表，则会出现员工不重复，但是所属部门会重复，增加删除不方便

传递依赖：id决定员工号，员工号决定员工名。三个数据或者更多的之间相互传递。

提高范式，就需要折表，减少传递。

## MySql

### 配置

#### 安装

环境变量：

path加入C:\Program Files\mysql-8.0.13-winx64\bin

将mysql-8.0.13-winx64目录放入C:\Program Files

将my.ini放入mysql-8.0.13-winx64目录

=================my.ini===========

[mysql]

default-character-set=utf8

[client]

port=3306

default-character-set=utf8

[mysqld]

port=3306

basedir="C:\Program Files\mysql-8.0.13-winx64"

datadir="C:\Program Files\mysql-8.0.13-winx64\data"

max\_connections=200

character-set-server=utf8

default-storage-engine=INNODB

default\_authentication\_plugin=mysql\_native\_password

运行intall.bat：

cd C:\Program Files\mysql-8.0.13-winx64\bin

mysqld --initialize-insecure

mysqld --install MySQL --defaults-file="C:\Program Files\mysql-8.0.13-winx64\my.ini"

net start mysql

pause

登录：

mysql -u root -p

use mysql;

update user set authentication\_string='' where user='root';

ALTER user 'root'@'localhost' IDENTIFIED BY '新密码'

【非mysql\_native\_password时：

ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql\_native\_password BY '新密码'

FLUSH PRIVILEGES;

】

卸载

net stop mysql

sc delete MySQL

#### 卸载

HKEY\_LOCAL\_MACHINE\SYSTEM\ControlSet001\Services\Eventlog\Application\MySQL文件夹删除

C:\ProgramData\MySQL删除

删除C:\Documents and Settings\All Users\Application Data\MySQL

### 导入

进入db

source d:\rmt.sql

#### 服务器运行

=======启动服务器

管理员模式运行cmd

net start mysql

=======停止

net stop MySQL

#### 登录

mysql -u root -p（第一次登录没有密码，直接按回车过）

mysql -h30.158.59.78 -u用户名 -p\*\* //远程登录 \*\*代表密码

#### 查看状态

mysql> status

#### 修改密码

mysql> set password =password('密码');

mysql> flush privileges;

### 用户管理

以root用户登录

#### 创建用户

insert into mysql.user(Host,User,Password) values("localhost","abc",password("1234"));

insert into mysql.user(Host,User,Password) values("192.168.125.133","abc",password("1234"));

//为192.168.125.133创建用户abc

flush privileges;

#### 删除用户

DELETE FROM mysql.user WHERE User="abc" and Host="localhost";

#### 授权

//为用户创建一个数据库

create database db;

//授权phplamp用户拥有phplamp数据库的所有权限。

grant all privileges on phplampDB.\* to phplamp@localhost identified by '1234';

//刷新系统权限表

flush privileges;

//指定部分权限给一用户

mysql>grant select,update on db.\* to abc@localhost identified by '1234';

//刷新系统权限表

mysql>flush privileges;

#### 修改用户密码

update mysql.user set password=password("new password") where User="abc" and Host="localhost";

mysql>flush privileges;

### 创建/使用数据库

create database \*\*\*;

use \*\*\*;

### 数据类型

int,integer,bigint

float,double

char,varchar

datetime: 2017-12-12 17:00:00

date:2017-12-12

time:17:00:00

### 密码

md5(“password”) 根据字符串创建密码

## mongodb

### 环境配置

环境变量path: C:\Program Files\MongoDB\Server\4.0\bin

安装启动服务：

mongod.exe --install

net start MongoDB

移除服务：

mongod.exe --remove

连接：

mongo.exe

mongod.cfg文件（路径需要手动创建）：

storage:

dbPath: c:\data\db

### 数据库操作

collection ：数据库表/集合（table）

document：数据记录行/文档（row）

field ：数据字段/域（column）

|  |  |
| --- | --- |
| show dbs | 显示所有数据库 |
| db | 显示当前使用的数据库名称 |
|  | |
| use jyl | 创建/选择数据库jyl |
| db.dropDatabase() | 删除当前使用的数据库 |
|  | |
| show collections | 显示当前使用的数据库中的所有集合名 |
| db.createCollection("jylcl") | 创建集合jylcl |
| db.jylcl.drop() | 删除集合jylcl |
|  | |
| db.jylcl.insert({'name':'jyl','age':1}) | 向集合jylcl插入数据，不存在则创建 |
|  | |
| db.jylcl.remove({'name':'jyl'}) | 按照条件删除集合中的文档 |
|  | |
| db.jylcl.update({'name':'jyl'},{$set:{'age':1}}) | 修改集合中的数据 |
|  | |
| db.jylcl.find() | 查找集合中所有数据，find().pretty()以格式化的方式来显示所有文档 |
| db.jylcl.find({'name':'jyl'})  db.jylcl.find({'name':'jyl','age':1}) | 按照条件查找集合中的文档 |