上一篇分享了数据库的基础知识，以及如何安装数据库，今天这篇分享数据库操作和 SQL。

SQL 全称是 Structured Query Language，翻译后就是结构化查询语言，是一种数据库查询和程序设计语言，用于存取数据以及查询、更新和管理关系数据库系统。

至于 ORACLE、DB2、Sybase、SQLServer、MySQL、MS Access 等都是数据库，虽然语法上有差异，但是基本上都是大同小异。作为一名数据从业者，虽然市面上有很多很智能很自助的数据工具，甚至有的拖拖拽拽就能实现，但作为一名报表工程师、数据分析师，不懂一点 SQL 是绝逼不行的。

之前很多文章对于 SQL 都一笔带过，轻描带写，略忽视这项基本技能的重要性，所以今天就来讲讲如何学习 SQL。

市面上都大量的书籍和教学视频，这里就帮大家提炼核心要点，给个学习方向。

本次的文章主要从以下几个方面进行说明，大家可以参考着学习。

- SQL 快速学习路线
- 数据库表基础操作
- SQL 基础语句
- SQL 高级语句

SQL 快速学习路线

零基础自学 SQL 时通常会遇到 2 个坑：

一坑：学习之前先安装数据库软件，可以是 SQL Sever，也可以是 MySQL。对于新人而言，安装数据库软件挺费时间，坑太多。我刚上大学那会要求 2 天内自学 SQL，结果我花了 1 天时间安装 SQL

Sever。前一篇文章分享了 MySQL 的安装教程，刚开始学用不上太庞杂的功能，所以推荐小而美的 MySQL，SQL Sever 也是可以的。

二坑：一上来就背 SQL 语法。我看过太多新人在那边吭哧吭哧做笔记背 SQL，个人不赞成这种学法，不实践记不住。就和 Excel 函数一样，理解含义和如何使用，关键时候去 w3school 查询一下就行了，以后用得多了自然就掌握了。**（后台回复关键词“SQL”获得）**

SQL 学习路线

1、下载安装 MySQL，或者安装软件 phpstudy（这个软件自带了 mysql 数据库，而且安装启动方便）。

2、我这里用的是 phpstudy，打开 phpstudy 后，点击下图中的启动，点击后数据库服务就会启动了。

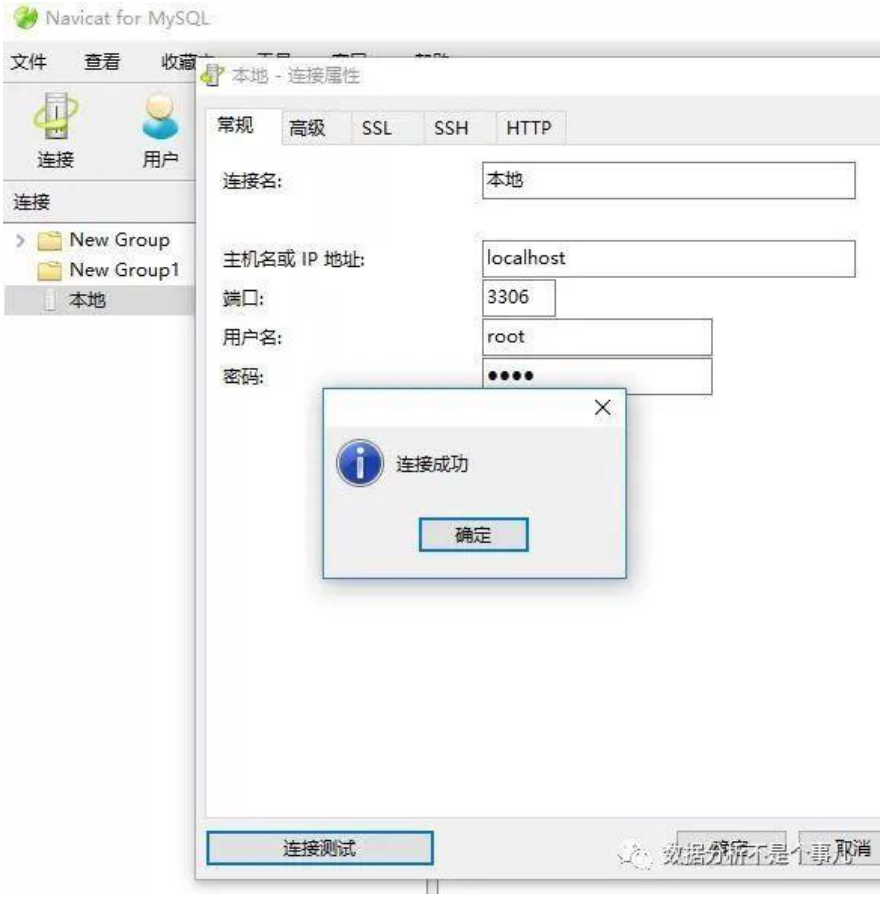
3、下载安装 Navicat 软件

这个软件可以轻松连接数据库，我们可以在这个软件中练习学习 SQL 语法

4、使用 Navicat 建立数据库连接

点击文件——新建连接，连接名随便填写，比如我写的是“本地”。主机名、端口都不需要修改，用户名和密码都是 root（因为我们安装的 phpstudy 里的 mysql 默认用户名密码就是 root），点击连接测试显示为成功后点确定保存。

以后再次访问时，双击即可。

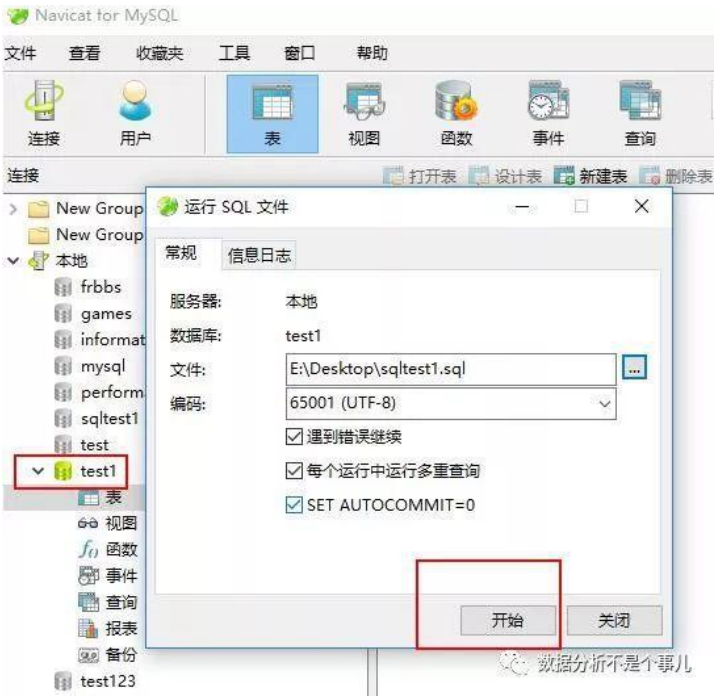


5、下载习题进行学习，内含习题需要用到的数据表。

6、导入第 5 步中下载的数据表。**（后台回复关键词“SQL”获得）**

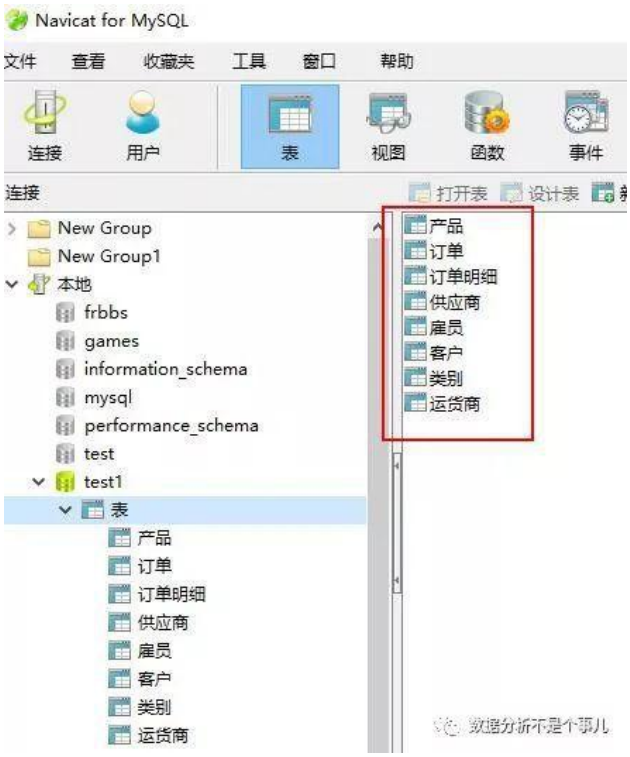
右击第 4 步中的“本地”，新建一个数据库 test1，双击打开新建的数据库（打开后颜色是绿色）。

拖拽刚才下载的数据表文件（sqltest1.sql）到 test1 上面。



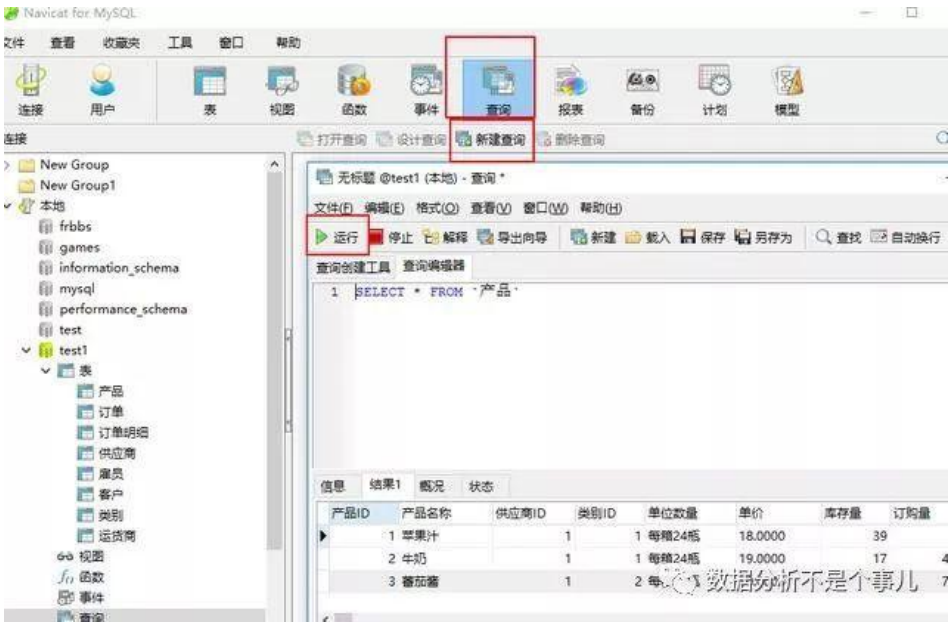
这时会出现上述弹窗，点击开始，等待导入完成后，点击关闭

在左侧空白处右击，选择“刷新”，即可看到刚才导入的数据表了。



7、打开 sql 语法编辑器

如下图点击查询——新建查询，在空白处输入 sql 语法，点击运行，如果输入错误会有报错，输入正确则会返回查询内容。



8、边做边学

打开第一部分习题，边做边学，通过查询 **w3c 的 sql 语法手册**完成

9、巩固练习

尝试不看 sql 语法手册，独立完成第二部分习题。

以上是 sql 的学习路径，接下来是 sql 语句的学习，掌握使用场景和含义，具体语法就不做解释了，大家参考 **w3c 的 sql 语法手册**学习。这里我归了下类，挑重点讲。

数据库表基础操作

首先数据库表必掌握的基础操作，建表、删表、往表里增数据、往表里删数据以及最重要的取数等等。

1、CREATE TABLE（建表）

怎么着也得先建立表~

```
CREATE TABLE emp
(
id int NOT NULL PRIMARY KEY,//添加主键
name varchar(20),
sex varchar(2),
age int,
chengji int,
money double
)

CREATE TABLE orders
(
o_id int NOT NULLPRIMARY KEY,
orderNo int,
e_id int,
FOREIGN KEY (e_id) REFERENCES emp(id)//添加一个外键
)
```

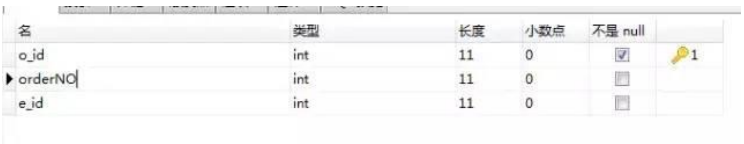
当你建表成功后，发现忘记添加主键，或者忘记添加外键，莫着急。使用以下咒语即可：

```
添加主键：ALTER TABLE emp ADD PRIMARY KEY(id);
添加外键：ALTER TABLE orders ADD FOREIGN KEY (e_id) REFERENCES emp(id);
```

建好的表如下：

名	类型	长度	小数点	不是 null	
id	int	11	0	☑	🔑1
name	varchar	20	0	☐	
sex	varchar	5	0	☐	
age	int	11	0	☐	
chengji	int	11	0	☐	
money	double	0	0	☐	

emp 表



名	类型	长度	小数点	不是 null	
o_id	int	11	0	☑	🔑1
orderNO	int	11	0	☐	
e_id	int	11	0	☐	

orders 表

2、INSERT(插入)

向表中插入数据

```
//向emp表中插入数据，插一条为例
//插入字符时使用 ' ',插入数值时不需要。
INSERT INTO emp VALUES(1,'ace','nan',14,12,12.456);
...
//向orders表中插入数据，插一条为例
INSERT INTO orders VALUES(1,2323,11);
```

id	name	sex	age	chengji	money
1	ace	nan	14	12	12.456
2	ace	nan	67	2	2.3
3	belli	nan	26	4	2.63
4	cao	nan	35	4	234.343
5	demo	nan	56	12	7.832
6	demo	nan	78	24	5.123
7	ac	nv	35	23	40.345
8	ac	nan	34	3	39.567
9	fk	nv	50	3	234.54
10	fku	nan	89	(Null)	35.673

插入数据后的 emp 表

o_id	orderNO	e_id
1	123	3
2	224	4
3	654	1
4	342	2
5	654	5
6	3423423	5
7	2332	6
8	234	7
9	2334	8
10	22323	(Null)

插入数据后的 orders 表

```
//向表中的某列插入数据
insert into table_name(列名1, 列名2, ...) values('值1', '值2', ...);
```

3、UPDATE(更新)

```
//更新某一行的某一列
update emp set age=12 where name='ace';
//更新某一行的若干列
update emp set age=13,chengji=34 where name='ace';
```

4、DELETE(删除)

```
//删除表中的某一行
delete from emp where id=8;
//删除表中的所有数据
delete * from emp;
```

在使用 delete 删除 emp 表中数据时，要注意该表与其他表是否存在关联关系，比如：外键。emp 表 id 是表 orders 的外键，如果要删除 emp 表中的 id，得先删除 orders 表中的外键。

5、DISTINCT（去重）

在表中，可能会包含重复值，这并不成问题。不过，有时你也许希望仅仅返回唯一不同的值。使用 distinct 关键字进行处理，用于返回唯一不同的值。

注意：distinct 关键字是去重！去重！去重！*把列中的重复值去掉！

曾经我在笔试的时候，有道 SQL 考题：请写出表中所有重复的 name 的所有数据。我当时没反应过来，直接就用了 distinct 关键字，后来就....

```
//来来来，正确的写法
select * from emp where
 name in
 (select name from emp group by name
 having count(1)>=2//count(1)，其实就是计算一共有多少符合条件的行,count(*)会全表扫描，速
 );
```

```
//查询emp表中的name，返回唯一的名字
select distinct name from emp;
```

6、Select（取数）

SELECT 语句用于从表中选取数据，结果被存储在一个结果表中（称为结果集）。这是以后最常用的操作，占据你 90%。

SQL SELECT 语法

SELECT 列名称 FROM 表名称

以及：

SELECT * FROM 表名称

比如需获取名为 "LastName" 和 "FirstName" 的列的内容（从名为 "Persons" 的数据库表），请使用类似这样的 SELECT 语句：

SELECT LastName,FirstName FROM Persons

SELECT 通常结合其他函数和语法使用。

SQL 基础语句

在实际的 SQL 使用中，肯定会涉及到有关函数的使用，这里简单介绍几种初学时必学的函数类型。

1、AVG ( )

AVG 函数返回数值列的平均值。NULL 值不包括在计算中。

```
//当求平均值的列包含null值时，null是不包括在计算中的（逻辑很正常~）
select AVG（chengji） as avg_chengji from emp;
//找出成绩高于平均成绩的name
select name from emp where
  chengji >(select avg(chengji) as avg_chengji from emp);
```

2、COUNT()

COUNT(）函数用于返回匹配指定条件的行数。

```
//count（count_name）
//返回指定列的值的数目，NULL值不算
select count(chengji) from emp;
//count(*)
//返回表中的记录数
select count(*) from emp;
```

3、MAX()

MAX 函数返回指定列的最大值，NULL 值不包括在计算中

```
//成绩最大
SELECT MAX(chengji) FROM emp;
```

4、MIN()

MIN 函数返回的指定列的最小值，NULL 值不包括在计算中

```
//年龄最小
SELECT MIN(age) FROM emp;
```

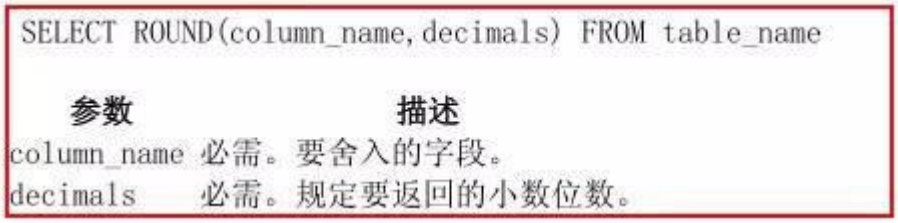
5、SUM()

SUM 函数返回指定列的总数

```
select sum(chengji) as sum from emp;
```

6、ROUND()

ROUND 函数用于把数值字段舍入为指定的小数位数



SELECT ROUND(column_name,decimals) FROM table_name

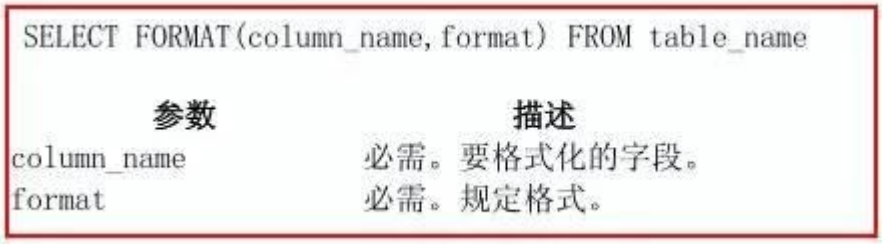
参数	描述
column_name	必需。要舍入的字段。
decimals	必需。规定要返回的小数位数。

round 函数需要的参数

```
//将money小数点保留1位
SELECT name,ROUND(money,1) as money FROM emp;
```

7、FORMAT()

FORMAT 函数用于对字段的显示进行格式化



SELECT FORMAT(column_name,format) FROM table_name

参数	描述
column_name	必需。要格式化的字段。
format	必需。规定格式。

format 函数需要的参数

SQL 高级语句

这一部分的内容是通常用到的，属于最开始学习 SQL 知识时必须要熟练的，我这里大致列出几项。

1、LIMIT

```
select * from emp limit 4;//只显示4行的数据
//限制显示的数据条数
```

2、LIKE

一般配合 where 使用，搜索条件中的指定模式

```
//% 可以理解为定义通配符
select * from emp where name like 'a%';
select * from emp where name like '%a';
select * from emp where name like '%a%';
//查询表中不包含的数据，使用"NOT"
select * from emp where name not like 'a%';
```

在上面我们可以看到，通配符“%”的使用方法，所以通配符必须要配合 like 运算符一块使用。

通配符还有以下几种：

```
//使用 "_"通配符
select * from emp where name like 'a_';
//使用[charlist]通配符
select * from where name like '[ac]%';
//使用[!charlist]通配符
select * from where name like '[!ac]%';
```

3、IN

从字面意思就可以知道它的作用是什么了

```
//找出名字为'demo'和'ace'的所有数据
select * from emp where name in ('demo','ace');
```

4、JOIN

联表运算符 JOIN，该运算符是用于将两个或者两个以上的表进行关联，并从这些表中查询数据。

对于联表来说，通过使用主键（primary key）和外键（foreign key）也可以建立连接。

```
//使用主键、外键将表连接
select e.name,e.age,o.orderNo
  from
  emp as e,orders as o//as 用法就当成为表起了一个别名
  where e.id=o.e_id;
```

除了上述直接使用条件关联，下面我们可以用可读性更高的 INNER JOIN 来写

```
//INNER JOIN的使用和JOIN的效果一样，以INNER JOIN为例
select e.name,e.age,o.ordersNo
  from
  emp e INNER JOIN
  orders o ON
  e.id=o.e_id ORDER BY name;
```

还有其他几种方连接方式（外连接）：

- LEFT JOIN:就算右表中没有匹配，也从左表返回所有的行
- RIGHT JOIN：即使左表没有匹配，也行右表返回所有的行
- FULL JOIN:只要有一个表存在着匹配，就返回行

5、ALTER

穿插介绍一下 alter，前面的例子中已经包含了几种 alter 使用方法。

```
//修改表中某列值
ALTER TABLE emp
ALTER COLUMN age int;
//给表中增加列
alter table emp add COLUMN money double;
```

6、UNION

UNION 操作符用于合并两个或多个 SELECT 语句的结果集。

> 请注意，UNION 内部的 SELECT 语句必须拥有相同数量的列。列也必须拥有相似的数据类型。同时，每条 SELECT 语句中的列的顺序必须相同。

上面引用的意思就是：道不同，不相为谋!

UNION 和 UNION ALL 命令几乎是等效的，不过加了“ALL”，就会列出所有的值。

```
//使用UNION
SELECT column_name(s) FROM table_name1
UNION
SELECT column_name(s) FROM table_name2

//使用UNION ALL
SELECT column_name(s) FROM table_name1
UNION ALL
SELECT column_name(s) FROM table_name2
```

注意：因为其也具有“唯一性”，容易和 PRIMARY KEY 混淆。面试或笔试常考两者的不同，在这里说明一下：

与 PRIMARY KEY 不同的是，每个表可以有多个 UNIQUE 约束，但是每个表只能有一个 PRIMARY KEY 约束。

为表添加 UNION，这里给出使用的 SQL 语法。

```
//为已创建的表添加UNION
ALTER TABLE table_name ADD UNIQUE (column_name);
//定义多个UNION约束
ALTER TABLE Persons
ADD CONSTRAINT u_name（约束名） UNIQUE (column_name1,column_name2,...);

//通过约束名来撤销
ALTER TABLE Persons
DROP INDEX u_name;
```

7、AUTO-INCREMENT(自增)

在运用中，我们希望在每添加一条数据后，自动的为我们的主键创建值。

```
create table emp
(
id int not null auto_increment,//默认自增起始值为1，每条数据记录递增加1
...
primary key(id)
);
/**
想改变自增的起始值，使用下列 SQL ：
ALTER TABLE emp AUTO_INCREMENT=100;
**/
```

8、ORDER BY

在前面中已经使用到了有关 order by 的 SQL 语句，order by 该语句用于对结果集进行排序，默认是进行 ASC 正序排序（从小到大）。

排序的两种方式：

- ASC:升序（从小到大）
- DESC:降序（从大到小）

举栗：

```
//对emp表中的name进行ASC排序查询
select * from emp order by name;
```

```
14  SELECT name,age FROM emp ORDER BY NAME;
```

信息 | 结果1 | 概况 | 状态

name	age
ace	67
belli	26
cao	35
demo	56
demo	78
tom	14

ASC 排序

对于 DESC 排序，这里就不进行举例了，大家可以自己写 SQL 试一下。

9、GROUP BY

通常配合合计函数使用，根据一个或多个列对结果集进行分组。

```
//文章前面的一个sql语句，查询表中重复的数据
select * from emp where
 name in
 (select name from emp group by name
 having count(1)>=2
) ;
```

具体的用法在介绍函数时会涉及到。

10、HAVING

在上面的例子中，我们使用 where 关键字来增加查询条件，这里增加 having 字句是因为，where 关键字无法与合计函数一起使用

同样引用上面的 SQL 语句。

具体的用法在介绍函数时会涉及到。

11、DEFAULT

DEFAULT 约束用于向列中插入默认值。

```
//在建表时可以设置默认值
create table hello
  (
    id int not null primary key,
    ...
    name varchar(255) default 'zhangsan'
  );
```

本次文章中写的相关知识点是我以前在学习中随手记录的，对一些 SQL 大牛来说，这些已经是耳熟能详了。会不会让你们产生一种灌水的错觉？？？？哈哈~