SQL语言及其操作

笔记本:	database_theory		
创建时间:	2021/6/26 18:23	**更新时间:**	2021/6/26 19:01
作者:	134exetj717		

学习完了数据库的基本概述，高级数据模型还有关系数据模型，接下来就是我们的建表过程，也就是需要利用到SQL语言进行操作。

- 在学习语言以前，我们先要了解，SQL语言的功能特点有哪些呢?
  - 功能一体化：SQL语言由三个子语言构成;
  - 语言非过程化：，用户只需定义”做什么“，至于”怎样做“，留给RDBMS系统内部去解决;
  - 交互式与嵌入式使用：即可独立地交互式使用，也可以通过宿主语言（比如java）结合起来，构成操作界面友好的数据库应用系统;
  - 标准化与易移植性：因为几乎所有RDBMS都是用SQL语言。
- 可能我们会好奇，三个子语言分别是做什么的呢?
  - 数据定义子语言：定义数据库、表、视图、索引
    - 日志：数据库故障恢复的重要手段和方法
    - 数据库的定义：

示例：

```
Create Database StuData
On Primary
   ( Name      = StuFile1,
     Filename  = `c: \ production \ data \ StuFile1.mdf',
     Size      = 10MB,
     MaxSize   = 1000MB,
     FileGrowth = 10MB),
   ( Name      = StuFile2,
     Filename  = `c: \ production \ data \ StuFile2.ndf',
     Size      = 10MB,
     MaxSize   = 1000MB,
     FileGrowth = 10%)
Log On
   ( Name      = Stulog,
     Filename  = `c: \ production \ data \ Stulog.ldf',
     Size      = 10MB,
     MaxSize   = 1000MB,
     FileGrowth = 10MB)
```

Designed by Tao Hongcai 2021/5/12

说明：

① 主数据文件扩展名为.mdf;

② 次数据文件扩展名为.ndf;

③ 日志文件扩展名均为.ldf。

    - 表的定义：

```
create 表名(
```

```
stu_id char(8) primary,
...
)
```

- 视图的定义
- 索引的定义
  - 唯一索引 unique
  - 聚簇索引 clustered，聚簇索引是物理位置上的移动
  - 非聚簇索引 nonclustered，类似于链表，只是索引了数据的地址
  - （注意：索引是不是越多越好呢？　答案不是，原因有：1.索引占用磁盘空间；2.系统维护索引结构，需要花费开销。

```
create index 索引名
on 表名 （属性）

drop 【表名索引类型】 index 索引名　　//会将该索引中的全部索引项全部清除
```

- 数据操纵子语言
  - 修改

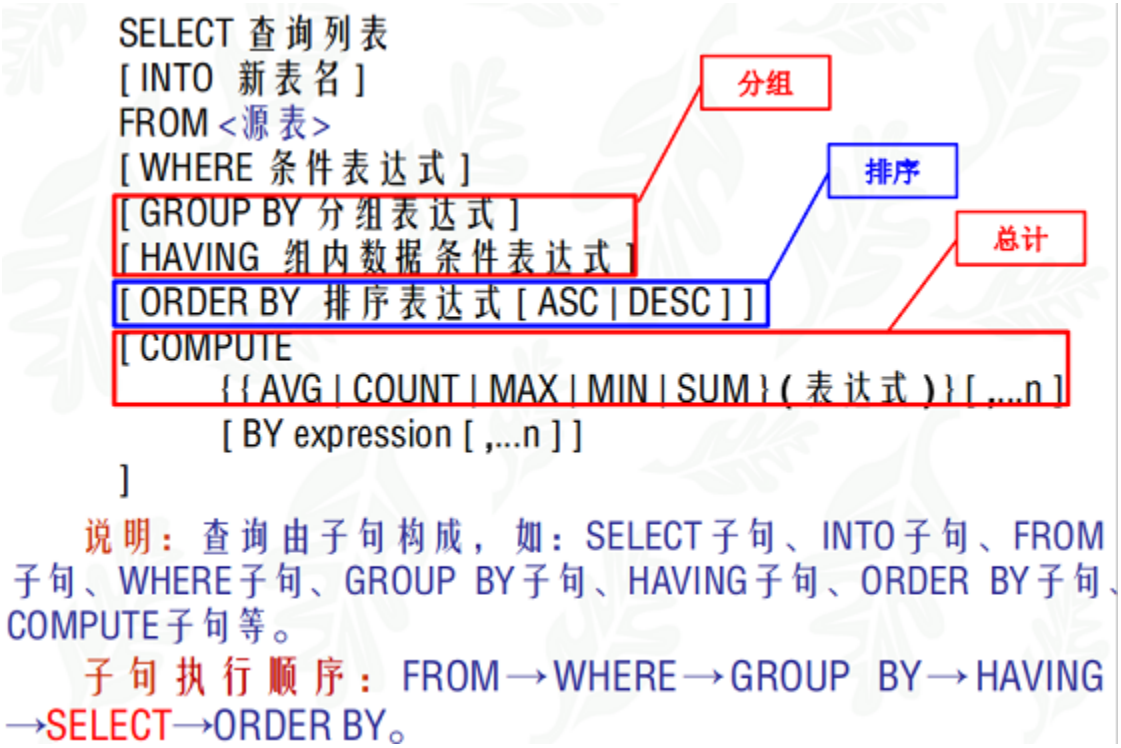
```
UPDATE  publishers
SET  city = 'Atlanta',  state = 'GA'

UPDATE  titles  SET price = price * 2
```

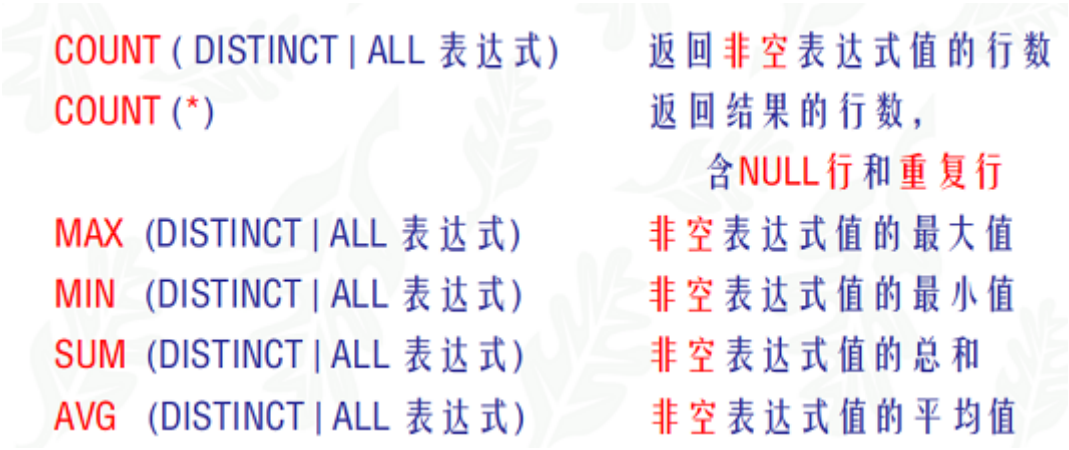
  - 删除

```
DELETE titleauthor
FROM  titleauthor  INNER JOIN  titles
      ON titleauthor.title_id = titles.title_id
WHERE titles.title LIKE '%computers%'
```

  - 查询



（1）LIKE：
1%（匹配任意一串字符）、_（匹配任意一个字符）、[ ]（取其中任意单个字符）、[^]（不能取其中所列字符）。
2 ^应与[ ]联用，如：[^a-f]或[^abcdef]表示a-f这几个字母不能出现。
（2）SOME|ANY：
语法：表达式 { = | < > | ! = | > | > = | ! > | < | < = | ! < }
{ SOME | ANY } (子查询)
示例1：Select * From A
Where A.a = SOME (Select b From B)
示例2：Select * From A
Where 5 <> ANY (Select b From B) 2021/5/12
Designed by Tao Hongcai
（3）ALL：
语法：表达式 { = | < > | ! = | > | > = | ! > | < | < = | ! < }
ALL (子查询)
示例：Select * From A
Where A.a > ALL (Select b From B)
含意：检查子查询是否有结果返回，如有结果则为TRUE；如返回NULL则FALSE。如带NOT则相反。
（4）[NOT] EXISTS：
语法：[NOT] EXISTS (子查询)
示例1：Select * From A
Where EXISTS (Select b From B)
示例2：Select * From A
Where NOT EXISTS (Select b From B) 2021/5/12
Designed by Tao Hongcai
（5）[NOT] IN：
含意：检查测试表达式的值是否在子查询或列表中，是则TRUE，否则FALSE；如带NOT则相反。
语法：测试表达式 [NOT] IN
(
子查询
| 表达式 [,...n]
)
示例1：Select * From A
Where A.a IN (Select b From B)
示例2：Select * From A
Where 8 NOT IN (Select b From B)



- 游标的定义及使用
  - 其中，@@fetch_status全局变量有三种状态，分别是：0（成功）、-1（失败或行超出结果集范围）、-2（数据行丢失）

例子：

```
DECLARE books_csr CURSOR FOR                //定义游标
SELECT title_id, type, price FROM titles
Declare @title_id char(6), @type char(12), @price money    //定义变量
Open books_csr
Fetch books_csr into @title_id, @type, @price        //执行一次fetch操作，指针下移一行
While @@fetch_status != -2      //一般都用 0
Begin
if @@fetch_status = -1
begin
raiserror (‘select failed’, 8, 1)
return
end
if @type= ‘business’
select @title_id, @type, @price, convert(money, @price*1.08)
else if @type = ‘mod_cook’
select @title_id, @type, @price
fetch books_csr into @title_id, @type, @price
End
```