**数据库**：数据库软件：mssqlserver，oracle，db2， ACCESS，SQLITE

**数据库文件**：主数据文件(MDF)和日志文件(log)

班级名称：(四期班训练营)长度可变，有中文----n var char

性别字段：如果考虑“男，女”那么就用bit类型,smallin类型

手机：长度 手机号11位 ，座机咋办，010-88888888 ，0871-6666666

身份证号码：长度：不变18位，有木有中文，没有，Char 会重复，

|  |  |  |  |
| --- | --- | --- | --- |
| **分类** | **备注和说明** | **类型** | **说明** |
| 二进制数据类型 | 存储非子符和文本的数据 | Image | 可用来存储图像 |
| 文本数据类型 | 字符数据包括任意字母、符号或数字字符的组合 | Char,8000 | 固定长度的非 Unicode 字符数据。固定长度的字符串相对于可变长度的字符串来说效率要高一些，在数据长度固定的情况下优先选用固定长度，省去了计算长度的过程，提高效率  Unicode字符会把所有字符都当两位来存储 |
| Varchar,8000 | 可变长度非 Unicode 数据 |
| Nchar,4000 | 固定长度的 Unicode 数据 |
| Nvarchar,4000 | 可变长度 Unicode 数据 |
| Text  varchar(max) | 存储长文本信息(指针,2G)  varchar(max)，大字符串类型可以保存非常多的字符，但是对于这种类型的数据DBMS经常将它们保存到单独的空间中，这就导致了数据的保存和加载速度比较慢，因此除非必要，否则不要使用。 |
| Ntext  nvarchar(max) | Nvarchar(max)代替 |
| 日期和时间 | 日期和时间在单引号内输入 | Datetime  Data  Time | 日期和时间 |
| 数字数据 | 该数据仅包含数字，包括正数、负数以及分数 | Bigint(8字节)  Int(4字节)  Smallint(2字节)  Tinyint(1字节)  Decimal[(p[,s])]  numeric[(p[,s])] | 整型类型  P:精度，默认为18位；  S:小数位数，默认为0 |
| Float[(n)] real | 近似数字类型 (n:尾数位数) |
| 货币数据类型 | 用于十进制货币值,money 和 smallmoney 数据类型精确到它们所代表的货币单位的万分之一。 | Money(C#:double) |  |
| Bit数据类型 | 表示是/否的数据 | Bit | 存储布尔数据类型(1-true 0-false) |

分离数据库：将当前数据库文件和数据库引擎的关系断开，没有任何关系了，这样我就可以随意的拷贝，剪切

在数据库文件上右键🡪任务🡪分离，🡪勾选“删除连接”，🡪确定

附加：把已经分离的数据库文件，重新让数据库引擎管理，重新把管理关系建立上，（分离的逆操作）

脱机：告诉数据库引擎，暂停操作当前数据库，直到你重新“联机”，，也可以随意拷贝，剪切

联机：将脱机的数据库重新加载到数据库列表中

## 主键：

标识一张表里面数据行，到底哪一行是哪一行，

设置主键的时候，不能有重名的，重号的

比如，一家公司，给所有员工编号，a0003 b0002，但是某天人事喝酒了，就弄错了，员工编号就重复了

身份证：有重复的，

银行卡号：升位 19 20

**主键分为**：逻辑主键和业务主键

业务主键就是具有真实意义的，就比如身份证，银行卡，员工编号，一但变化，难以维护

逻辑主键：没有任何实际含义，只为了标识当前列在当前数据表里的唯一标识，由系统自身维护

不可以手动编辑(特殊情况下除外)

主键列：就是不能插入重复数据(默认带索引)

如果数据行数量大于int的最大取值范围，那么说明，这张表的数据已经超过几十亿行了

每一张表都推荐有主键列，主键列，设置标识

最终这个列就会是不会有重复的数据，自动编号的

主键标识列：就算数据行被删除了，增长的数字也是按照原来的增长，

想要重新排号，可以删除主键列，再重新添加上去

## 主外键：

作用：就是减少重复数据，

把一个表中总是重复的数据提取出来，单独放在另外一个表中，这样，在原来的表中只要存储一个指向提取出的数据的表的行，

在数据库里面，在外键上右键，可以添加 主外键约束(约束外键值，只能是主键值有的值)，但是，这已经不推荐使用了

例子：

书：书名，价格，分类ID

分类表：id，分类名字，所在区域

## 创建索引

创建主键索引：右击鼠标设置主键

创建唯一索引：右击鼠标进入🡪索引/键表🡪添加🡪设置(类型为索引、列、是唯一标识)

查看表的索引：展开数据库🡪展开表🡪展开指定数据表🡪展开索引🡪右击选择索引属性

删除表的索引：展开数据库🡪展开表🡪展开指定数据表🡪展开索引🡪右击删除索引

## 维护数据表

在表中插入新列：

改变表中列的排列顺序：

自动标号列和标识符列：列属性🡪展开标识规范🡪是标识（只有在列的数据烈性为int是才可用）

可计算的列：在列属性🡪展开计算列规范🡪添加计算公式

默认值或绑定：列属性🡪默认值或绑定（输入为空时默认写入该值，或绑定的某个值）

## SQL语句

基础语句

selEct \* FROM dbo.Student WHERE Name='张a'----大小写问题

SQL与C#的区别之处：

字符串用单引号表示

关键字大小写不区分

注释用/\* \*/多行,--单行

单等号判断

创建数据库，表

create database MySchool

on

(

--括号一定是圆括号

name='MySchool\_data',--数据库名称

filename='d:\MySchool\_data.mdf',--物理文件名

size=5mb,--初始大小，

maxsize=10mb,--最大大小

filegrowth=15% --主文件增长率

)

log on

(

name='MySchool\_log',--日志文件名

filename='d:\MySchool\_log.ldf',--日志物理文件名

maxsize=4mb,--最大大小

size=2mb,

filegrowth=1mb

)

go

创建数据表

CREATE TABLE Student

(

id int identity(1,1) primary key,列名、数据类型、自增长、主键

NAME NVARCHAR(16) NOT NULL,

age INT NOT NULL

)

## 增删查改

### 增加语句

insert into dbo.Student

(name,gender,Address,Phone,Age,Birthday,CardId,CId) --列名

values

('王五',1,'北京海淀,中关村','13888888888',20,'2012/01/01','111111',2) --各列所插入的值

注意事项：

1. 增加的时候。Bit字段要用 “0,1”表示true和false
2. 时间字段用单引号包括，但是里面还是要遵循基本的时间格式'2012-01-01'
3. INSERT 语句中列的数目要和 VALUES 后面的值的数目一样
4. 不能为标识列插入数据(特殊情况下可以，SET IDENTITY\_INSERT)

小技巧：>>直接拖拽“列”就可以快速的插入列名

**其他形式的解读：**

INSERT INTO dbo.Student

VALUES

('王五',1,'北京海淀,中关村','13888888888',20,'2012/01/01','111111',2)

除了主键列，其他所有字段的值，都要写出来，而且还要按照顺序写出来

INSERT INTO dbo.Student

(Name,Gender,Age,Birthday,CardId)VALUES('王五5',1,20,'2012/01/01','111111')

可以只给指定的列插入数据

INSERT INTO dbo.Student (Name,Gender)VALUES('钱七',0)

不能为phone插入null值

INSERT INTO dbo.Student (Name,Gender,Phone)

VALUES

(N'钱七①②βδΟουÇÄãÏíスヌフ叁叁',0,10000)

在插入特殊字符的时候，字符串前面需要加上大写的N

INSERT INTO dbo.Student

VALUES

('小明',default,N'地址',default,20,'2010-10-10','100000',2)

在插入数据的时候，可以显示的告诉数据库使用该字段的默认值

INSERT INTO dbo.Student

VALUES

('张''四',DEFAULT,N'地址',default,20,'2010-10-10','100000',2)

插入的数据中有单引号的时候，需要输入两个单引号 “''”

另一种形式的insert into语句是和select关键字相结合将查询到的整张表插入到指定的数据表中

### 删除语句

delete FROM Student

删除Student表中所有数据

truncate TABLE Student

清空表，重置表(把表重置会刚刚创建时候的状态，自增列，重新从1开始了)

以上两者效率有很大差别，如果用delete会产生很多的日志，truncate 就只会产生一行日志

语句后面可以使用where关键字来实现条件删除的功能

如：delete from student where sAge > 20

drop TABLE Student

直接删除表，不仅仅是数据，连表都不见了

### 查询语句

**简单的数据检索：**

select \* from student

--查找 所有 从 student表中

//select\*--表示选择要查询表中的哪些列（\*表示全部）

where **条件**--where部分表示要查询表中的哪些行（用来筛选哪些行需要显示）

SELECT sName,sAge FROM Student ----全部检索之后，但只输出指名的列

**检索后只显示指定的列并使用AS关键字为列头赋予新的名字**

SELECT sName AS 姓名,sAge AS 年龄,sBirthday AS 出生日期 FROM Student

**使用where关键字检索符合条件的数据**

SELECT sName FROM Student WHERE sSex=‘女’

Select getdate() --方便快捷地取到服务器的时间

Select Top 的使用

SELECT TOP 10 \* FROM dbo.Student--取出前10条的所有数据

SELECT TOP 10 Name,Gender,Age FROM dbo.Student--取出前10条的部分数据

SELECT TOP 10 PERCENT \* FROM dbo.Student---取出前面10%的数据，所有小数，是进位的，21/10=2.1 3

--19/10 1.9 2

#### Top

top关键字适用于调用者不需要知道全部的数据，只希望查询前面n条的数据。(其常与order by连用)

select top 17 percent \* from TestDB.products

percent-百分之 其显示的结果采用进一法

##### rowcount

rowcount关键字的作用与top关键字的作用一样，只是rowcount是指定具体的行数。

rowcount使用时注意：

* rowcount有极大地灵活性，可以动态地指定返回的行数；
* 使用完毕后一定要记得set rowcount 0不然容易造成系统错误；
* rowcount没有with ties参数，所以返回的结果不全面；
* rowcount全部平台都适用，Top关键字只适用于Transact-SQL语言。

rowcount 4 --设置返回结果的行为4

select \* from [TestDB].[dbo].[Table\_Storage] order by [weight]

go

set rowcount 0 --恢复rowcount

##### with ties

top关键字的另一种用法：（with ties参数）

select top 5 with ties \* from TestDB.Table\_Storage order by weight

--选择质量最轻的5辆车，当最后一辆车还有另外一辆车的质量与其相同时，使用with ties参数来显现全部的信息。

#### Distinct

去除重复行，但是是正对于查询之后的结果，来去除重复行，

它是判断结果数据中，整行数据，只要有一个列不相同，那么就不认为是相同数据

select distinct \* from [TestDB].[dbo].[Table\_Storage]

Top和distinct结合使用的话，distinct是先执行的，先去除重复项，然后再取出想要的条数

#### IN关键字

判断一个列的值是否在后面括号内，

SELECT \* FROM dbo.Student WHERE Gender IN ('男','女')--正确

SELECT \* FROM dbo.Student WHERE Gender IN (0,1,2)--错误，两者数据的类型不同

#### Group by

1、对数据进行分组，分组之后的数据就是“分组信息”，和原来的表里的信息就没有联系了

2、分组之后，可以取到分组数据，就是根据什么字段分组，就能取到字段的名字，

1. 还能使用聚合函数

4、GROUP BY子句必须放到WHERE语句的之后 ,Group By与Order By都是对筛选后的数据进行处理，而Where是用来筛选数据的。

select tName as 姓名,COUNT(tName) as 个数 from teacher group by tName --分组

SELECT \* FROM dbo.Teacher ORDER BY Salary DESC --根据某个列值进行分组

#### ALL

只有在select语句还包括where子句时，ALL关键字才有意义。如果使用ALL关键字，则查询结果包含使用group by子句产生的所有组，即使某些组没有符合搜索条件的行；如果不使用ALL关键字，则查询的结果将不包含不符合条件的行的组。

select product'商品名称',avg(weight) as '商品质量'

from [TestDB].[dbo].[Table\_Storage]

where [weight] > 40

group by all [product]

go

--在使用ALL关键字之后不满足where子句条件的行组将会以null显示

#### Order by

将当前查询出来的数据结果，进行排序，排序按照后面指定的列排序

SELECT \* FROM dbo.Student ORDER BY Gender DESC

DESC—倒序排序(降序)，ASC—正序排序(升序)

9876543210 0123456789

1、汉字，用拼音的首字母排序，升序是从a-z，降序是从z-a

2、Order by 是常放在where后面的，对于其他的语句一般也要放到所有语句的后面，就是先让其他语句进行筛选，全部筛选完成后，最后排序一下

3、系统默认的是升序排序，关键字：ASC，如果想倒过来，关键字：DESC

4、如果是根据多个列排序，那么两个之间用逗号隔开，排序会先按照前面一个排序之后，再按照后面一个排序

5、表中数据是集合，集合是没有顺序的。Order by 返回的数据是有顺序的，故此我们把order by 以后返回的数据集合叫“游标”。

#### Having

对分组信息进行过滤，因为分组之后的信息和原来的表信息没有关系了

Having可以用的之后出现在group子句中的列，还有聚合函数

SELECT Age,COUNT(id) FROM dbo.Student WHERE Age >25 group BY Age HAVING Age>25 --对分组之后的信息进行过滤

SELECT Age,COUNT(id) FROM dbo.Student WHERE Age >25 group BY Age HAVING COUNT(id)>1 --分组之后显示人数>1的组

Where过滤数据表，having过滤分组

* where子句用在from语句后，having子句用在group by语句后
* having子句中可以使用统计函数，而where子句中不能使用；group by子句中也不能使用统计函数，必须是原始列
* 必须在group by子句中列出select查询字段中所有的非集合字段

#### 聚合函数（统计函数）

系统已经定义好的一些函数，方便我们的使用，在使用时我们可以用关键字来直接调用来实现其函数性质

--max() min() sum() avg() count()聚合函数：

SELECT MAX(Age),MIN(Age),SUM(Age),AVG(Age) FROM dbo.Student

Count，取得满足条件的数据行的 行数

----数字，字母，汉字(拼音) 排序规则

--SELECT COUNT(id) FROM dbo.Student WHERE Name='钱七' AND Pwd='888888' 返回一个数字

COUNT(\*) 返回组中的项数。包括 NULL 值和重复项。

COUNT(ALL *expression*) 对组中的每一行都计算 *expression* 并返回非空值的数量。

COUNT(DISTINCT *expression*) 对组中的每一行都计算 *expression* 并返回唯一非空值的数量。

SELECT COUNT(DISTINCT JobTitle) FROM HumanResources.Employee;

对于大于 2^31-1 的返回值，COUNT 生成一个错误。这时应使用 COUNT\_BIG，so推荐使用count(sysId)

注意事项：

1、聚合函数对null值不计算

2、如果一行的数据都是null，count(\*)包含对空值行、重复行的统计。

3、平均成绩select avg(english) from score

#### 模糊查询(Like)

使用系统已经定义好的匹配符，按照定义的规则匹配数据，如果能匹配就查出来

通配符，匹配符：\_ % [] [^]

\_ 代表一个任意字符

% 代表零个或多个任意字符

[] 代表一个字符的取值区间 [0-9] [a-z]

^ 配合[]使用，表示不是这个区间，

数据库不兼容的，sqlserver可以用，其他数据库要用not like

[]里面不区分大小写

效率很低

要通配\_、%、[、^这些字符怎么办？[\_]、[%]、[ [ ]、^(不需要放到中括号里，因为^只有放到中括号中才认为是通配符)

SELECT \* FROM dbo.Student WHERE Name LIKE '王%' --查询表中所有姓王的数据

SELECT \* FROM dbo.Student WHERE Name LIKE '%王%' --查询表中所有姓名中有王的数据

SELECT \* FROM dbo.Student WHERE Name LIKE '%王\_' --查询表中所有姓名中王后面有一个字符的数据

SELECT \* FROM dbo.Student WHERE Name LIKE '王%[0-9]%' --查询中有数字的所有数据

SELECT \* FROM dbo.Student WHERE Name LIKE '王%[a-z]%' --查询表中有字母的所有数据

SELECT \* FROM dbo.Student WHERE Name LIKE '王%[0-9a-z]%' --查询表中有字母或数字的所有数据//或（[0-9,a-z]）

SELECT \* FROM dbo.Student WHERE Name not LIKE '王%[a-z]%' ---查询表中王之后不是字母的所有数据

SELECT \* FROM dbo.Student WHERE Name LIKE '%^%' --查询Name中含有^的数据

SELECT \* FROM dbo.Student WHERE Name LIKE '%[%]%' --查询Name中含有任意个%的数据

#### 空值判断

SELECT \* FROM dbo.Student WHERE Address <> NULL —错误

SELECT \* FROM dbo.Student WHERE Address IS NULL —判断是为null 的

SELECT \* FROM dbo.Student WHERE Address IS NOT NULL —判断不是为null 的

SELECT Name,ISNULL([Address],'地球') FROM dbo.Student

Isnull可以判断后面是否为null，为null就用第二个参数代替

SELECT Name,ISNULL(Age,20) FROM dbo.Student --第二个参数值要与Age的类型一致

SELECT 'a'+NULL --结果为空

#### 多表查询

从多张表中得到一个结果集。

通过student表得到学生姓名

通过score表得到成绩

显示某个学生的英语、数学成绩

select [dbo].[TblStudent].[tSName] as '姓名', Score.[english] as '英语',[dbo].[Score].[math]'数学'

from [dbo].[Score],[dbo].[TblStudent]

where [dbo].[Score].[studentId] = [dbo].[TblStudent].[tSCardId]

--代码说明：

--select子句中展示将要显示的列名（某个表特有的可以省略其前缀）

--from子句中是指定查询数据所在的表

--where子句通过某个数据将两张表连接起来

#### JOIN

##### inner join(内连接)

把两张表或多张表的数据行，按照指定的规则连接在一起，

--1：查询所有学生的姓名、年龄及所在班级

SELECT dbo.Student.Name,Age,dbo.Classes.Name FROM dbo.Student

JOIN dbo.Classes ON dbo.Classes.id=Student.CId

Join 连接多张表的数据时，会筛选要满足所有连接条件的数据才会留下来，其中任何一个条件不满足都不会出现数据

##### LEFT JOIN(左连接)

--4：查询所有学生(参加及未参加考试的都算)及成绩

SELECT \* FROM dbo.Student

LEFT JOIN dbo.Score ON dbo.Student.id =dbo.Score.Sid

在普通的join基础上，优先显示左边的表的数据，如果能满足连接条件，就显示连接之后的右表的数据，如果不满足连接条件，还是会显示左表数据，右表数据部分就是null(不知道)

##### RIGHT JOIN(右连接)

在普通join的基础上，优先显示右表的数据，如连接条件满足，就显示左表的数据，如果不满足就显示为null

--5：请查询出所有没有参加考试

SELECT \* FROM dbo.Student WHERE

id NOT IN (

SELECT dbo.Student.id

FROM dbo.Student

RIGHT JOIN dbo.Score ON dbo.Student.id = dbo.Score.Sid

WHERE dbo.Student.id IS NOT NULL

)

Join配合子查询

INNER JOIN其实就是join，我们在用join的时候，系统就默认识别为innerjoin了

CROSS JOIN会得出两张表的乘积行

#### 联合结果集(Union)

连接两个结果集

1. 两个集合必须具有相同的列数
2. 列具有相同的数据类型（至少能隐式转换的）
3. 最终输出的集合的列名由第一个集合的列名来确定
4. 使用的默认的排序规则对得到的结果进行排序(若要手动进行排序则将order by子句写入第二个select子句中)

SELECT Name,Age,Gender FROM dbo.Student--19

UNION

SELECT Name 姓名,Age 年龄,Gender 性别 FROM dbo.Student—19

Union默认会去除重复行

SELECT Name,Age,Gender FROM dbo.Student--19

UNION All

SELECT Name 姓名,Age 年龄,Gender 性别 FROM dbo.Student—19

Union all 不会去除重复行

一般都是使用union all

#### CASE表达式

在数据库中，对数据进行比对，

两种用法

第一种，不对任何列进行case **条件判断**

SELECT

Sid,English AS en,

CASE

WHEN English >=60 THEN '及格'

WHEN English <60 THEN '不及格'

END AS jige

FROM dbo.Score

从case开始：开始判断，判断什么？不知道，，当when开始的时候才知道判断什么东西

如果在判断一个列的值的时候，满足多个条件(when)，就先满足谁就执行谁

类似于c#中的 if else if else if else if else if else if else if.........else

SELECT

Sid,English AS en,

CASE

WHEN English >60 THEN '英语及格' if else

WHEN Math >60 THEN '数学及格' if else

ELSE '都不及格' else

END AS jige

FROM dbo.Score

第二种 直接case列 **值判断**

SELECT

CASE ptId

WHEN 1 THEN '家人'

WHEN 2 THEN '同事'

WHEN 3 THEN '同学'

END

FROM dbo.PhoneType

从case开始，一开是就知道要判断ptid的值，如果这个值是1就输出”家人”

相当于c#中的switch(age) case

例1：

SELECT id,Name,

(CASE Gender

WHEN 0 THEN '女'

WHEN 1 THEN '男'

END) AS Gender,Salary

FROM dbo.Teacher

在结果集中，可以包含两个列，列名完全相同的，

在Ado中，如果我们习惯了用列名来取数据，这样就会出问题了，

最终，会取出哪个数据呢？

例2：

SELECT

Sid,English AS en,

CASE

WHEN English >60 THEN '及格'

WHEN English <60 THEN English ///报错

END AS jige

FROM dbo.Score

说明：多个then后面的所有数据类型都必须相同

SELECT

Sid,English AS en,

CASE

WHEN English >60 THEN '及格'

WHEN English <60 THEN CAST(English AS NVARCHAR(10))

END AS jige

FROM dbo.Score

如果多个then后面的数据类型不相同可以做数据类型转换

#### 分页查询

AS t 可以吧结果集当成一张表，t就是这张表的表名，我们可以对这张表做任何查询操作

分页倒序显示

SELECT \*

FROM (SELECT TOP 10 \*

FROM [TestDB].[dbo].[products]

WHERE [pid] NOT IN

(SELECT TOP 10 [pid] FROM [TestDB].[dbo].[products])

ORDER BY [TestDB].[dbo].[products].[pid] ASC

) AS t

ORDER BY t.[pid] DESC

分页正序显示

declare @pageSize int,

@currentPage int;

set @pageSize = 10

set @currentPage = 1

select top (@pageSize) \*

from [TestDB].[dbo].[products]

where [pid] not in

(

select top (@pageSize \*(@currentPage - 1)) [pid]

from [TestDB].[dbo].[products]

order by [pid]

)

order by [pid]

--第二种分页方式

SELECT \* FROM

(SELECT \*,ROW\_NUMBER() OVER(ORDER BY ar\_id) AS nrows FROM dbo.Area) AS t

WHERE t.nrows BETWEEN 11 AND 20

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

利用游标存储过程进行分页，效率最差，但最为通用

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

declare @pageSize1 int,

@currentPage1 int;

set @pageSize1 = 10

set @currentPage1 = 3

select top (@pageSize1) \*

from

(

select ROW\_NUMBER() over(order by pid) as RowNumber,\*

from [TestDB].[dbo].[products]

) as t

where RowNumber > (@pageSize1 \* (@currentPage1 - 1))

ROW\_NUMBER()会在结果集的基础上加上了一列，这一列从1开始，中间不会有间断的

然后配合between and使用，效率比较高

### 修改语句

UPDATE dbo.Student SET Age=25

--修改了student表中所有行的age值

UPDATE dbo.Student SET Age=18 WHERE Name='陈珊'

--修改了student表中name是陈珊的age值

UPDATE dbo.Student SET Age=19,gender=0 WHERE id=1

--修改了student表中name是陈珊的age值和gender值

UPDATE dbo.Student SET Phone='10010' WHERE Name='钱七' AND Gender=0

Where后面可以有多个条件判断，多个条件中间用“空格and空格”隔开

UPDATE dbo.Student SET Address='a' WHERE Name='王五' AND Gender=1 OR Name='王五6'

--3

UPDATE dbo.Student SET Address='c' WHERE Name='王五' OR Gender=1 AND Name='王五6'

--4

在数据库执行的时候，and条件是优先于or条件执行的

UPDATE dbo.Student SET age=age+1

UPDATE dbo.Student SET age+=1 --2005 不支持

Set值的时候，可以取出原来的值(在原来的基础上+1)

UPDATE dbo.Student SET Address = '未知' WHERE Address = NULL--这个是不行滴 ---null不能用=判断

UPDATE dbo.Student SET Address = '未知' WHERE Address IS NULL

UPDATE dbo.Student SET Address='北京' WHERE Address IS NOT NULL

NULL 值的特殊判断

UPDATE dbo.Student SET address = '男儿国' WHERE Gender!=0 不等于“!=”

综上所述：

Where之后的语句各条件需要用逻辑关系符号连接

(||)or、(&&)and、(!)not、<、>、>=、<=、 <>（不等）、=（等于）、!=（不等于）

**写法：**update 表名 set 列名=值 ， 列名=值 where 条件 （优先级not>and>or,括号可以改变优先级）

## Where

between and 在数据库内部是做过特殊优化的，执行效率比=> and <= 等这种方式快

SELECT \* FROM dbo.Student WHERE Age BETWEEN 25 AND 30 ---25的有，30也有

相当于Age>=25 AND Age<=30

小分页：

SELECT \* FROM dbo.Student WHERE id BETWEEN （num-1）\*5+1 AND num\*5 (有错)

1:1-5

2:6-10

## 约束

非空约束---就是不能为 null

主键约束(PK) primary key constraint---唯一(不重复的) 且 不为空

唯一约束+非空位数

唯一约束 (UQ)unique constraint 唯一，允许为空(null)，但只能出现一次

默认约束 (DF)default constraint 如果不给值，默认值

检查约束 (CK)check constraint 范围以及格式限制

在设计界面，字段上右键，有“check约束”，点击之后

外键约束 (FK)foreign key constraint 表关系—主外键关系

添加主外键关系，外键的值必须来自主键表

## 执行顺序

查询某张表，如果表里面有1000W数据，我们只查出100条数据

🡪再对着100条数据进行分组，可以筛选分组，

如果指定列，结果集就更小了，这里只有100条指定列的数据

取出前面几条，或者取出重复

最进行排序，排序就只需要对100条数据排序了

## 类型转换

Cast(待转换的值 as 想要转换成的类型)

CAST(1 AS NVARCHAR(10))

## 批量插入

INSERT INTO dbo.Student (Name, Pwd, Gender, Address, Phone, Age, Birthday, CardId, CId)

SELECT Name, Pwd, Gender, Address, Phone, Age, Birthday, CardId, CId FROM dbo.Student

将一个结果集当成值插入数据库，结果集的列数量，类型，都要一样

Select \* into Score2 from Score where 1<>1

可以复制一张表的结构信息，没有主键