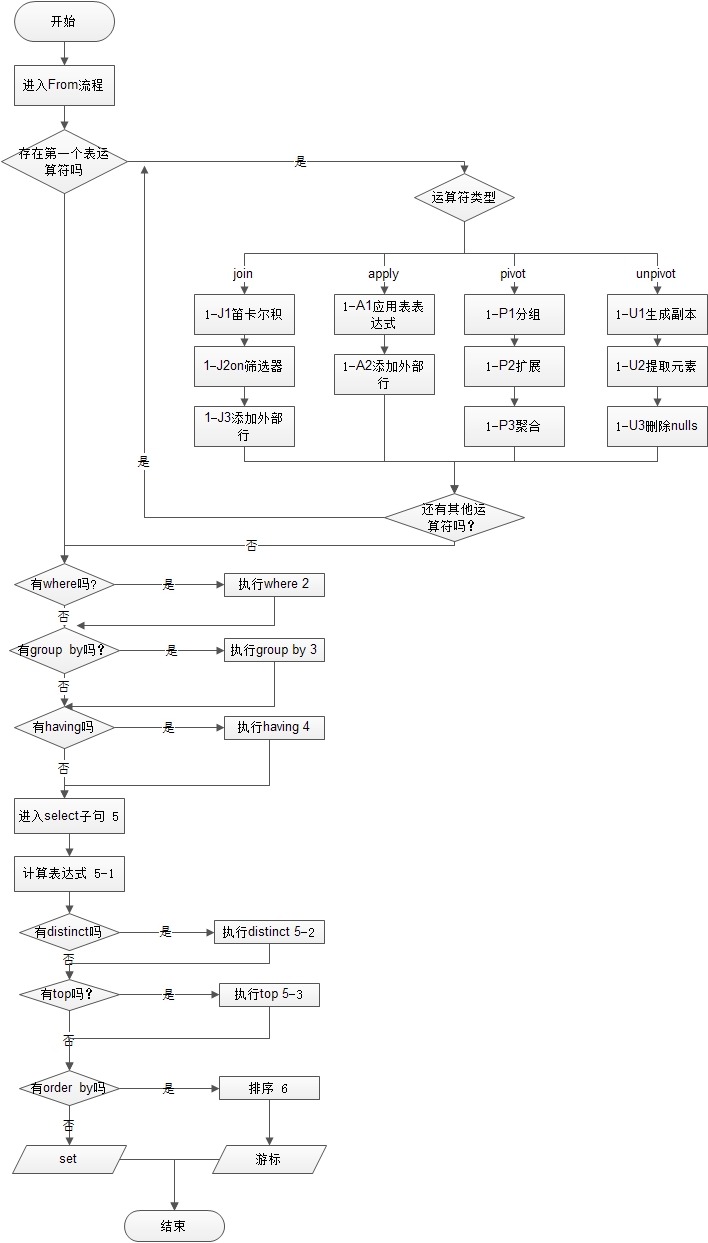
**Select查询详解**

**SQL语句的执行顺序：**

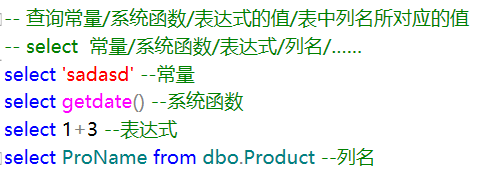


**先From找到一个表 -> 执行表运算符 -> 执行where筛选 -> 执行group by分组 -> 执行 having -> 执行select后面的语句(distinct、聚合函数、列名) -> 执行order by排序 -> top -> 完成**

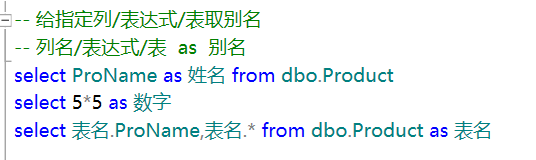
具体请查阅：[http://blog.csdn.net/bitcarmanlee/article/details/51004767](select查询详解.docx)

**语法和实例：**

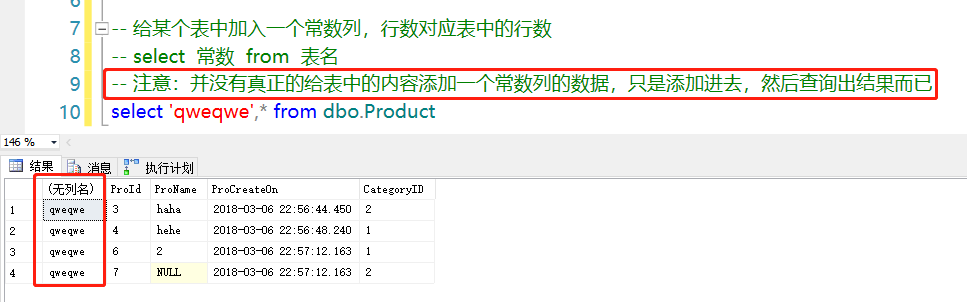
1. select 常量，系统函数，表达式，列名 ....



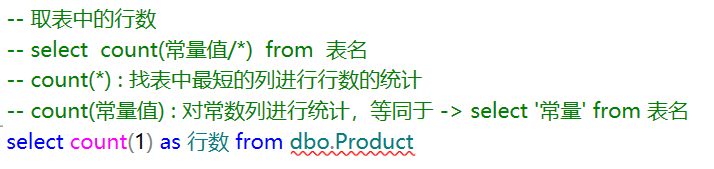
1. 给指定 列/表达式/表 取别名



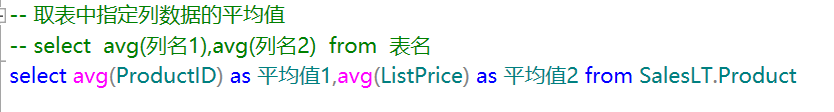
1. 给表中加入一个常数列，输出结果为所查的表所拥有的行数（并不是真正的添加一个列）



1. 聚合函数配合select的使用
2. count()：查找表中的行数



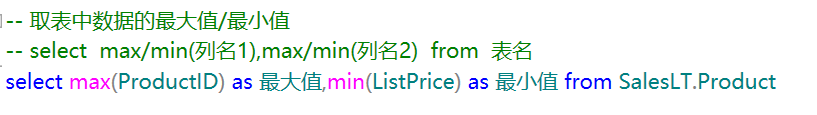
1. avg()：取指定列中的数据的平均值



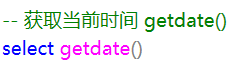
1. sum()：取指定列中的数据的总和



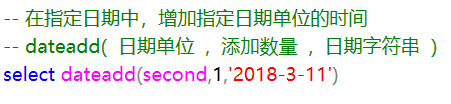
1. max()/min()：取指定列中的数据的最大值/最小值



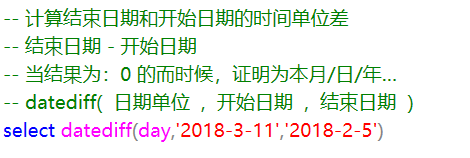
1. 日期函数配合select的使用
2. getdate()：获取当前时间



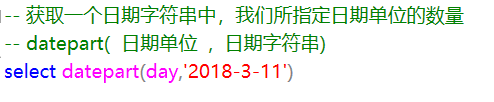
1. dateadd()：在指定日其中，增加指定日期单位的时间



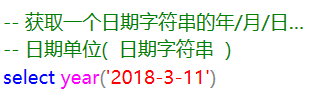
1. datediff()：计算结束日期和开始日期的时间单位差



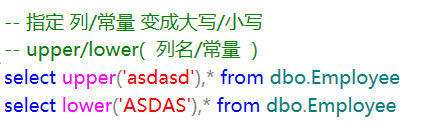
1. datepart()：获取一个日期字符串中，我们所指定的日期单位的数量



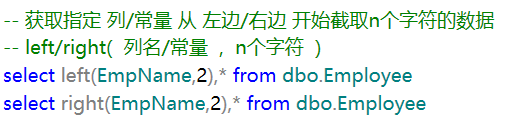
1. year/month/day/…()：取一个日期字符串的年/月/日/…



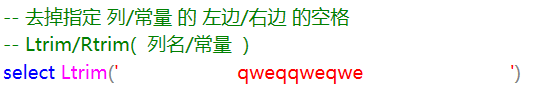
1. 字符串函数与select的配合使用
2. upper/lower()：指定 列/常量 变成大写/小写



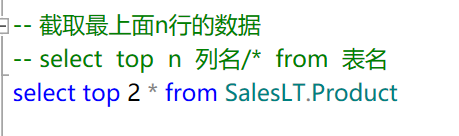
1. left/right()：获取指定 列/常量 从 左边/右边 开始截取n个字符的数据



1. Ltrim/Rtrim()：去掉指定 列/常量 的 做左边/右边 的空格

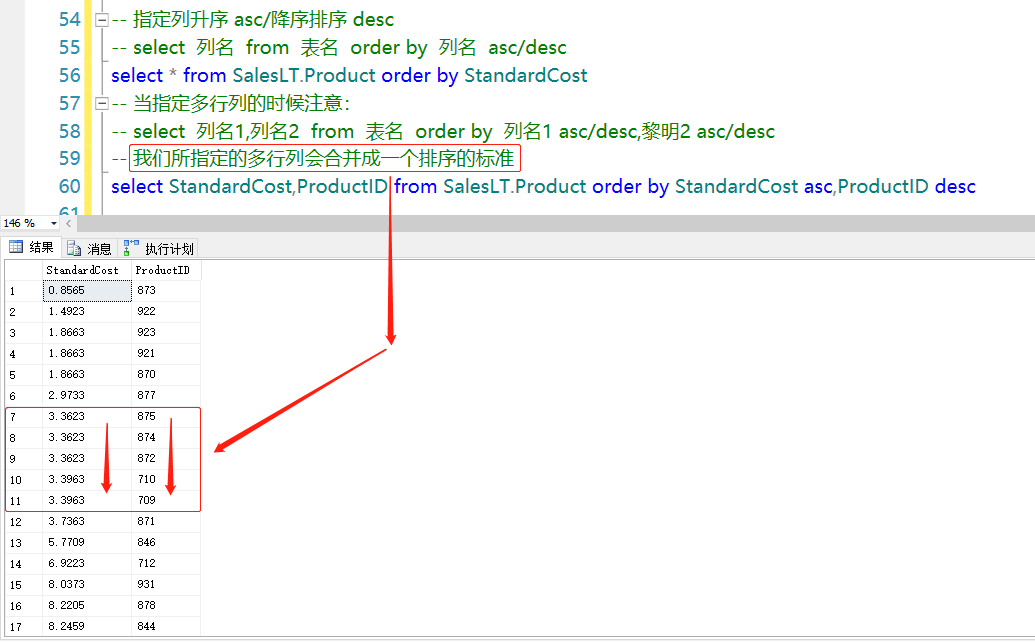


1. top/order by配合select的使用
2. top n：截取由头开始的n行的数据：



1. order by ：指定列的数据 升序排序：asc / 降序排序：desc

**注意：指定多行列时的问题**

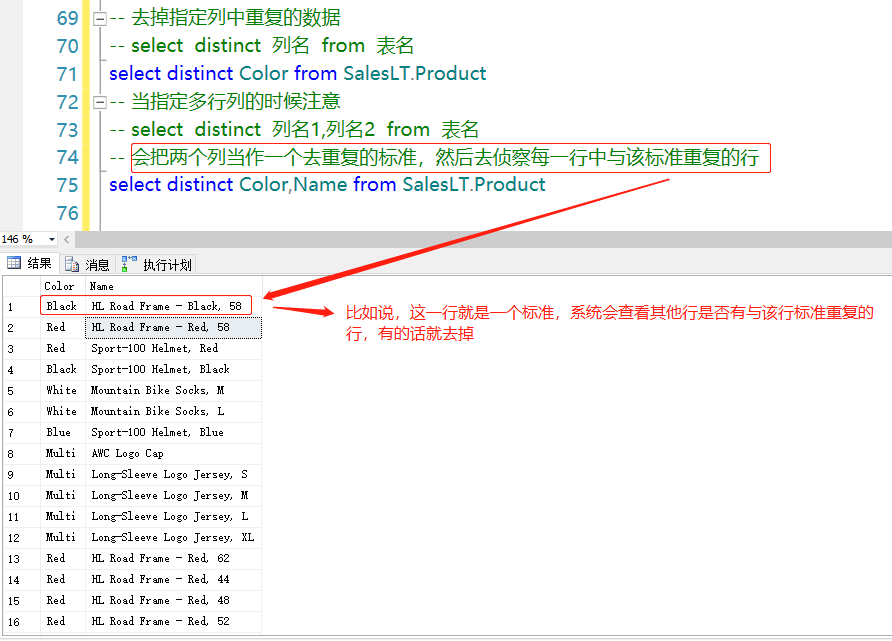


1. top和order by的配合使用

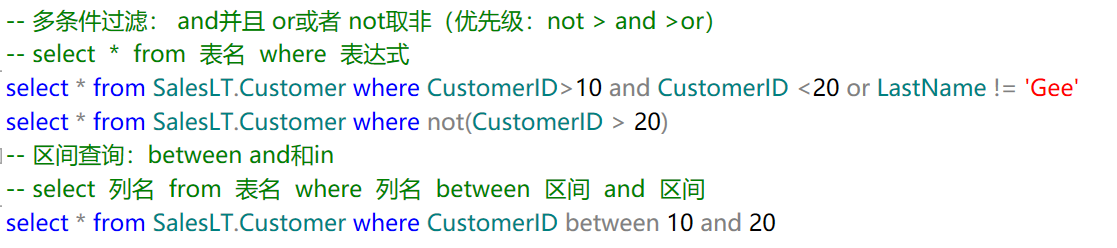


1. 去掉指定列中重复的数据 distinct：

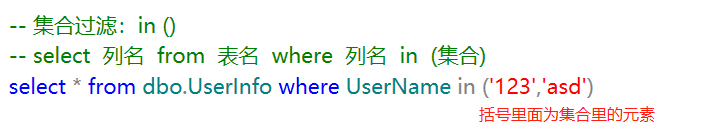
**注意：指定多行列时使用的问题**

****

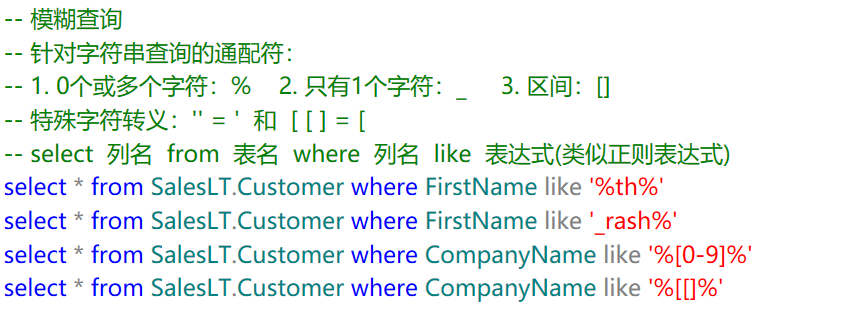
1. where条件过滤查询和select的配合使用
2. 多条件查询 and/or/not 和区间查询 between and



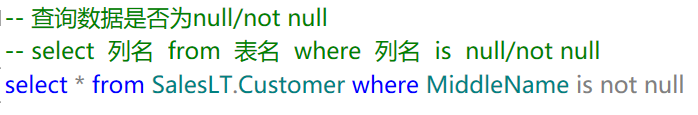
1. 集合过滤 in



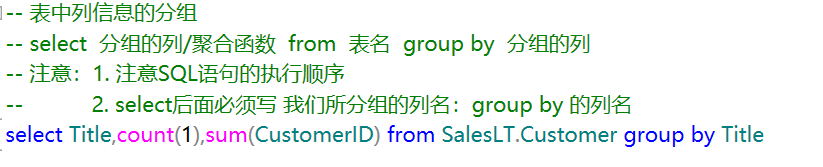
1. 模糊查询 like（类似正则表达式）



1. 查询该列中为 null/not null 的数据

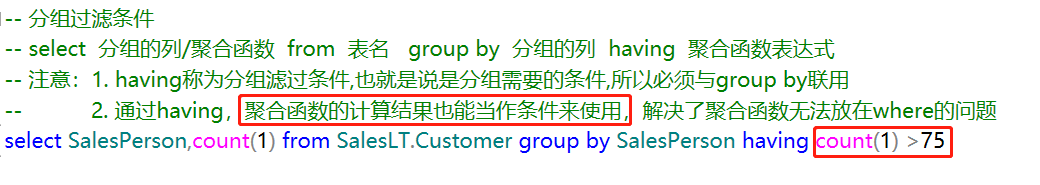


1. group by表中列的数据的分组



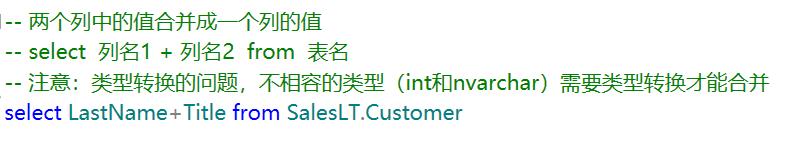
1. 分组条件过滤

**注意：需配合group by使用**

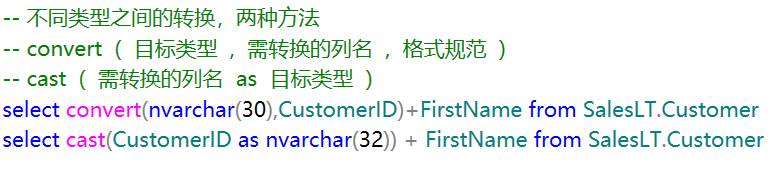


1. 通过 + 号，把某个表中指定两个列的值合并成一个列的值

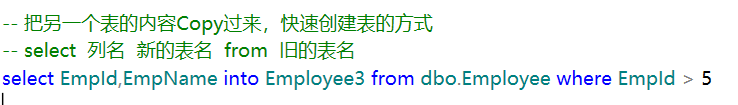
**注意：类型转换的问题**



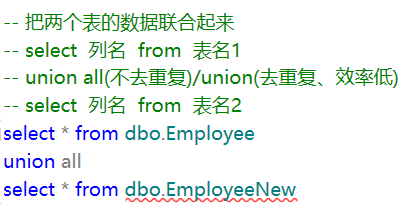
1. 类型转换，12号知识点注意问题的解决办法



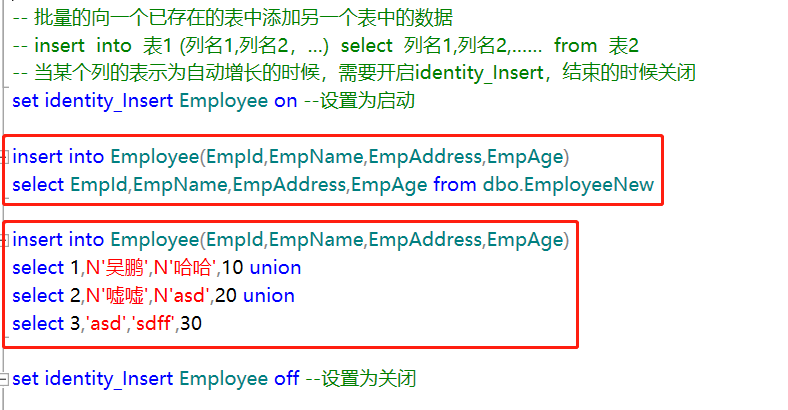
1. 把另一个表的指定内容Copy过来，快速创建一个新的表



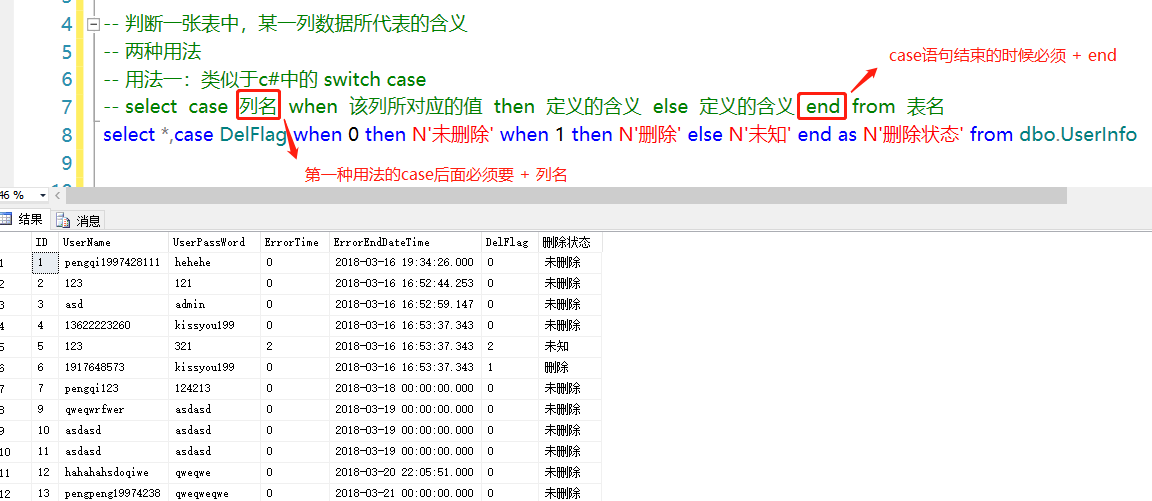
1. 把两个表的数据联合起来，union/union all



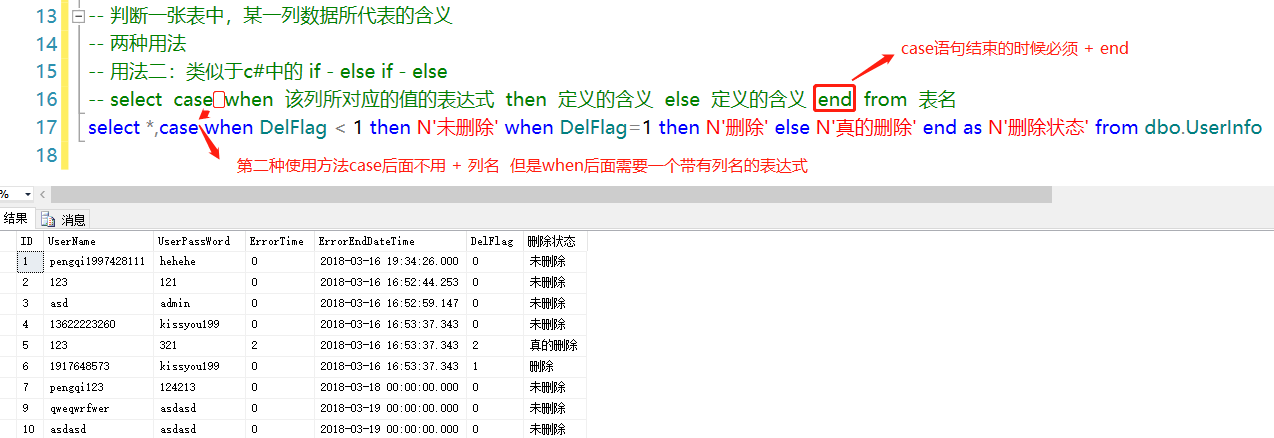
1. 批量的向一个已存在的表加入另一个表中的数据



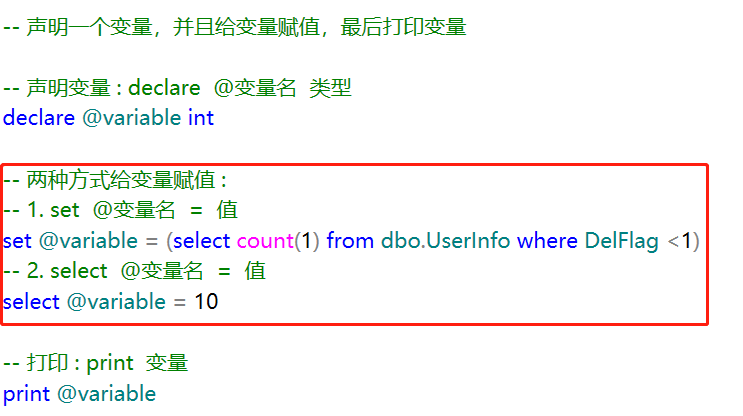
1. 判断某一张表格中，某一列中的数据所代表的含义case-when-else
2. 用法一：类似于c#中的switch case，指定一个列名，并对该列名中所对应的制定的值进行判断



1. 用法二：类似于c#中的if - else if - else，直接通过带有列名的表达式进行判断



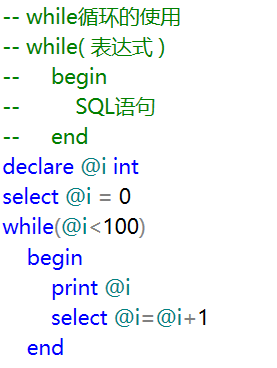
1. 变量的声明、变量的赋值、变量的打印



1. SQL中的if else/if else if else 的使用

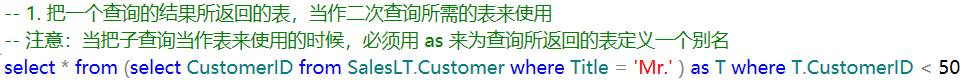


1. SQL中的While()循环的使用



1. 子查询的使用
2. 把一个查询的结果所返回的表，当作二次查询所需的表来使用

注意：当把子查询当作表来使用的时候，必须用 as 来为查询所返回的表定义一个别名



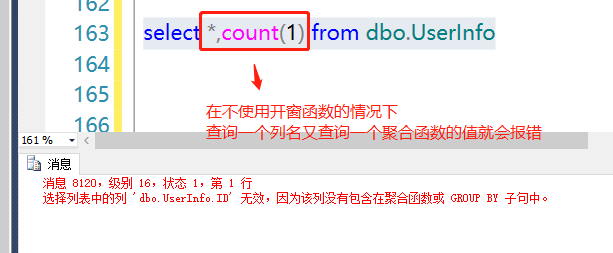
1. 把一个查询的结果所返回的数据，当作是二次查询所需的表达式来使用

注意：子查询所返回的列，必须与二次查询中的 where/haveing… 后面所判断的列相对应

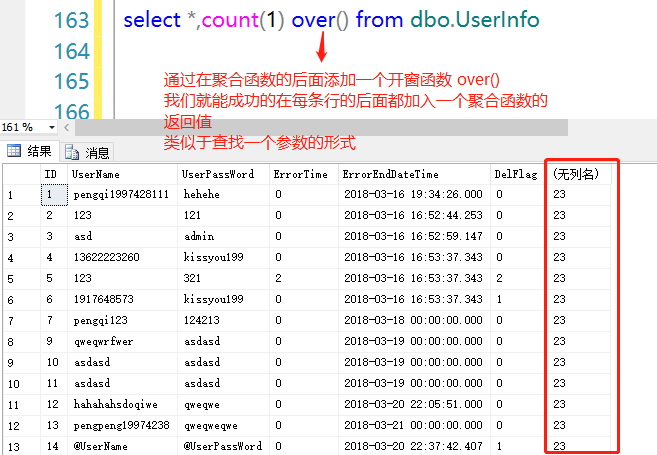


1. over() 开窗函数详解

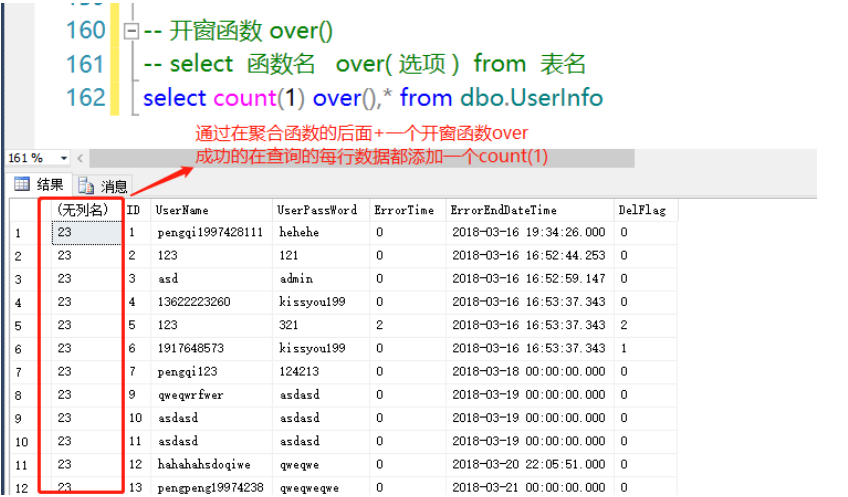
如果单纯的查询一个聚合函数，那么整个的查询结果只能是聚合函数所返回的列，不能在查询聚合函数结果的同时又去查询一个表中的某个列（**除了在使用group by的时候**），否则会报错，如下图：



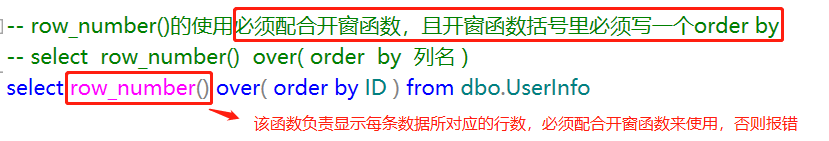
而当在聚合函数的后面写一个开窗函数就很好的解决了这个问题，属于是一个聚合函数的加强版



1. 开窗函数配合普通的聚合函数的使用



1. row\_number()函数的使用必须配合开窗函数



1. 分页查询的处理

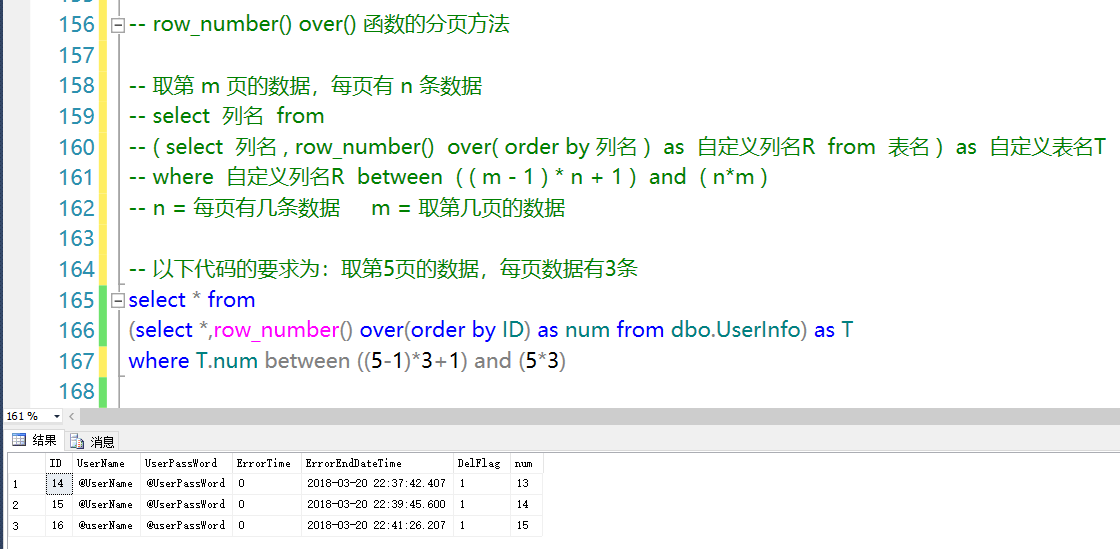
**注意：分页查询并不是真正的把查询结果的表分成一页一页的形式，而是我们把表中的数据排序好，并动态的定义每页有多少条数据，当我们取某一页的数据的时候(如：我需要取第二页的数据，并定义每一页有两条数据，其实就是取3、4行的而数据)就需要用到以下两种方法**

1. 双order by排序分页法：越过多少条，取多少条

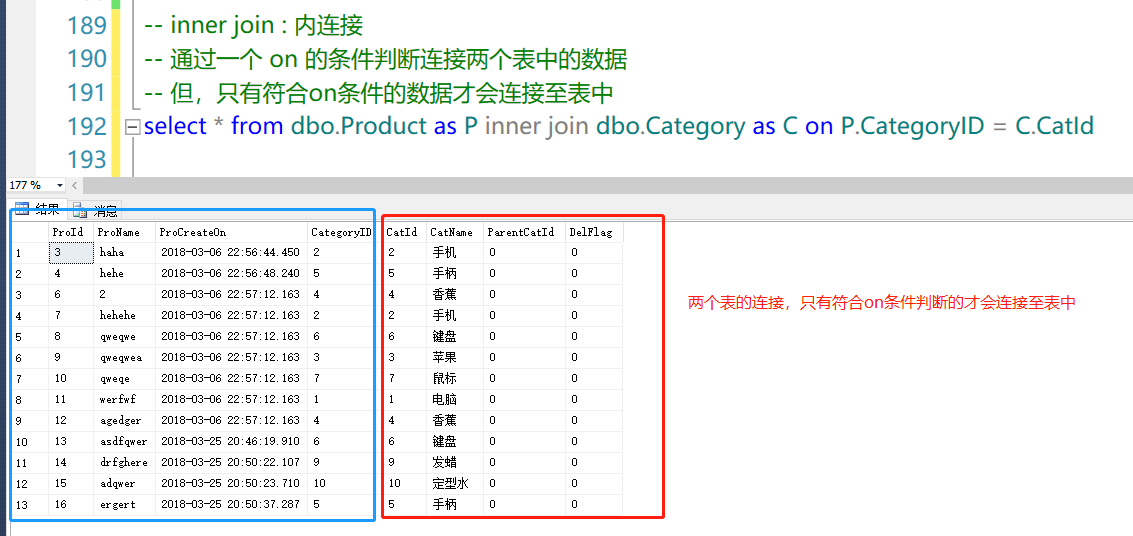


1. row\_number() over() 函数的分页方法

**注意：开窗函数配合row\_number()函数的使用，括号里必须有order by 排序**



1. 表连接 -> inner join ( join ) /left join/right join/full join/cross join
2. inner join ( join ) 内连接：连接两个表的数据，只有符合on的条件判断的数据才会连接至表中



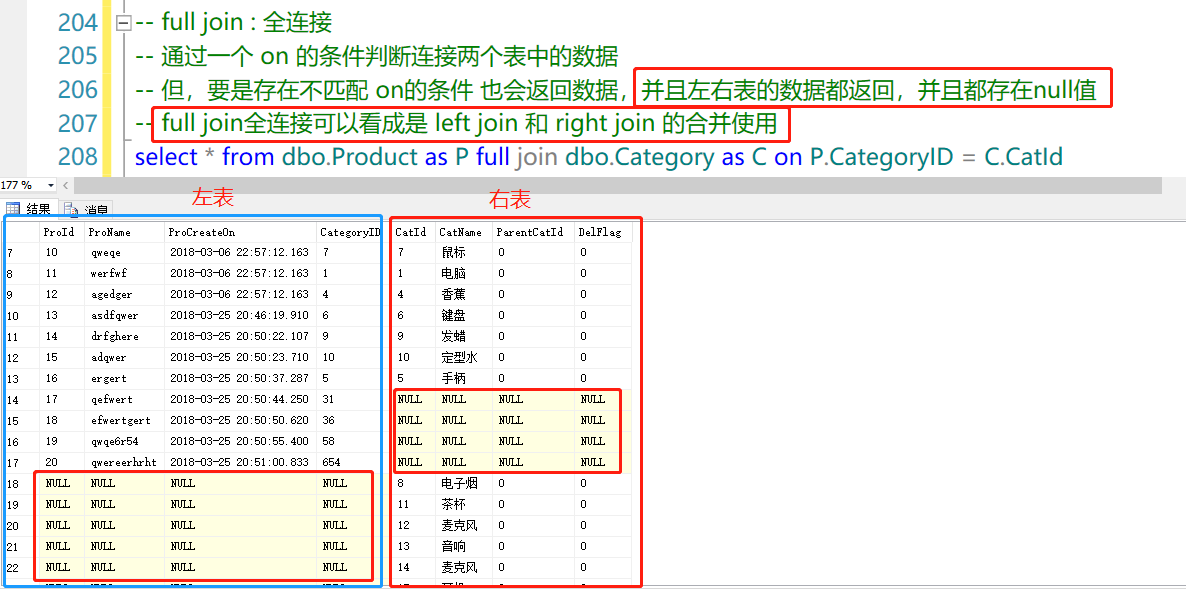
1. left join 左连接：连接两个表的数据，如果存在不符合on的条件判断的数据也会连接至表中，但是只返回左表的真实数据，右表的数据为null



1. right join 右连接：连接两个表的数据，如果存在不符合on的条件判断的数据也会连接至表中，但是只返回右表的真实数据，左边的数据为null



1. full join 全连接：链接两个表的数据，如果存在不符合on的条件判断的数据也会连接至表中，并且左右表都可以返回真实数据和null



1. cross join交叉连接：连接两个表的数据，通过笛卡儿积计算出两个表中所有可以匹配的数据，并查询出来



1. SQL中的全局变量以@@开头，变量都由SQLServer管理和维护，开发人员不能修改
2. @@FETCH\_STATUS：**配合游标的使用**

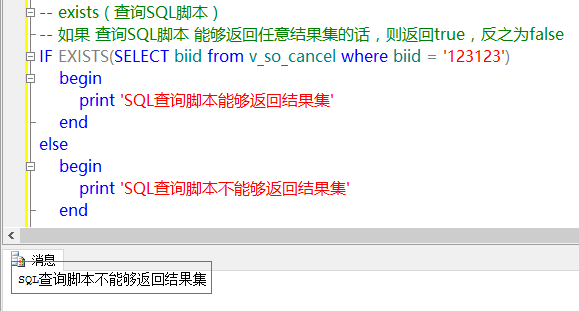
-> @@FETCH\_STATUS = 0：游标执行语句成功

-> @@FETCH\_STATUS = -1：游标执行语句失败或不在结果集当中

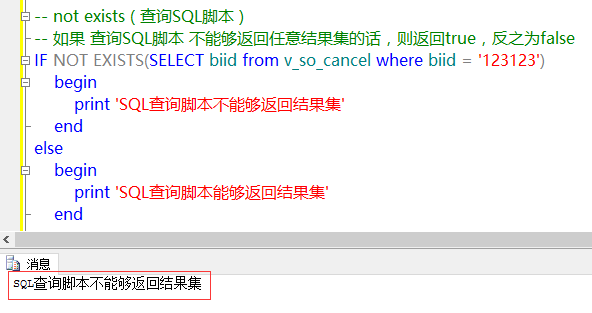
-> @@FETCH\_STATUS = -2：游标所提取的行不存在



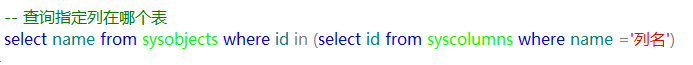
1. exists ( 查询SQL脚本 ) /not exists ( 查询SQL脚本 ) 的使用
2. exists ( 查询SQL脚本 )：如果 查询SQL脚本 能够返回任意结果集的话，则返回ture，反之为false



1. Not exists ( 查询SQL脚本 )：如果 查询SQL脚本 不能够返回任意结果集的话，则返回true，反之为false



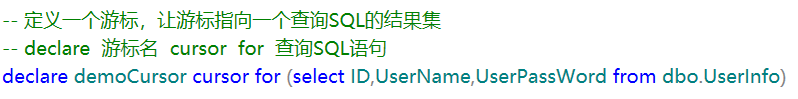
1. 查询指定列在哪个表



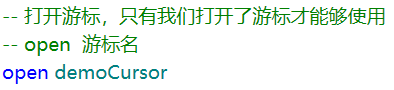
1. SQL中的游标的使用

**注意：游标不同于其他的SQL脚本，是一种面向行（面向过程）的做法**

1. 定义一个游标，让游标指向一个查询SQL的结果集



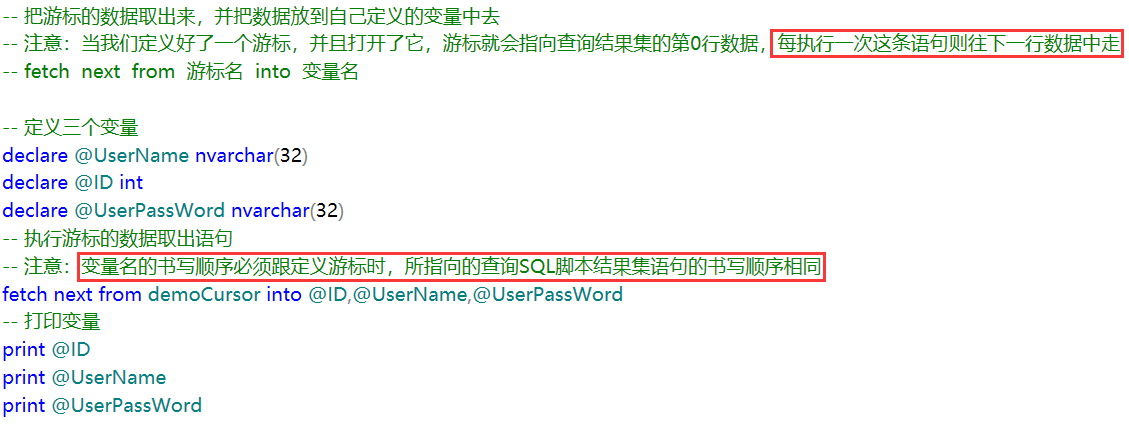
1. 打开游标，只有当我们打开了游标，才能够使用游标



1. 把游标所指向行的数据取出来，并把数据添加进我们定义的变量中

**注意：**

1. **当我们定义好了一个游标，并且打开了他，游标就会指向查询SQL返回的结果集的第0行，每当执行一次指条语句，游标的指针则往下一行走，我们可以配合全局变量 @@FETCH\_STATUS = 0 来使用**
2. **执行游标的数据取出语句的时候，变量名的书写顺序必须跟所指向的查询SQL语句中的列名书的写顺序相同**

****

1. 当我们使用完成游标之后，必须关闭、释放掉游标

