SQL 语句大全

--语 句 功 能

--数据操作
SELECT --从数据库表中检索数据行和列
INSERT --向数据库表添加新数据行
DELETE --从数据库表中删除数据行
UPDATE --更新数据库表中的数据

-数据定义
CREATE TABLE --创建一个数据库表
DROP TABLE --从数据库中删除表
ALTER TABLE --修改数据库表结构
CREATE VIEW --创建一个视图
DROP VIEW --从数据库中删除视图
CREATE INDEX --为数据库表创建一个索引
DROP INDEX --从数据库中删除索引
CREATE PROCEDURE --创建一个存储过程
DROP PROCEDURE --从数据库中删除存储过程
CREATE TRIGGER --创建一个触发器
DROP TRIGGER --从数据库中删除触发器
CREATE SCHEMA --向数据库添加一个新模式
DROP SCHEMA --从数据库中删除一个模式
CREATE DOMAIN --创建一个数据值域
ALTER DOMAIN --改变域定义
DROP DOMAIN --从数据库中删除一个域

--数据控制
GRANT --授予用户访问权限
DENY --拒绝用户访问
REVOKE --解除用户访问权限

--事务控制
COMMIT --结束当前事务
ROLLBACK --中止当前事务
SET TRANSACTION --定义当前事务数据访问特征

--程序化 SQL
DECLARE --为查询设定游标
EXPLAN --为查询描述数据访问计划
OPEN --检索查询结果打开一个游标

FETCH --检索一行查询结果
CLOSE --关闭游标
PREPARE --为动态执行准备 SQL 语句
EXECUTE --动态地执行 SQL 语句
DESCRIBE --描述准备好的查询

---局部变量
declare @id char(10)
--set @id = '10010001'
select @id = '10010001'

---全局变量
---必须以@@开头

--IF ELSE
declare @x int @y int @z int
select @x = 1 @y = 2 @z=3
if @x > @y
print 'x > y' --打印字符串'x > y'
else if @y > @z
print 'y > z'
else print 'z > y'

--CASE
use pangu
update employee
set e_wage =
case
when job_level = '1' then e_wage*1.08
when job_level = '2' then e_wage*1.07
when job_level = '3' then e_wage*1.06
else e_wage*1.05
end

--WHILE CONTINUE BREAK
declare @x int @y int @c int
select @x = 1 @y=1

```
while @x < 3
begin
print @x --打印变量 x 的值
while @y < 3
begin
select @c = 100*@x + @y
print @c --打印变量 c 的值
select @y = @y + 1
end
select @x = @x + 1
select @y = 1
end
```

```
--WAITFOR
--例 等待 1 小时 2 分零 3 秒后才执行 SELECT 语句
waitfor delay '01:02:03'
select * from employee
--例 等到晚上 11 点零 8 分后才执行 SELECT 语句
waitfor time '23:08:00'
select * from employee
```

SELECT

```
select *(列名) from table_name(表名) where column_name operator value
ex:(宿主)
select * from stock_information where stockid = str(nid)
stockname = 'str_name'
stockname like '% find this %'
stockname like '[a-zA-Z]%' --------- ([]指定值的范围)
stockname like '[^F-M]%' --------- (^排除指定范围)
--------- 只能在使用 like 关键字的 where 子句中使用通配符)
or stockpath = 'stock_path'
or stocknumber < 1000
and stockindex = 24
not stock*** = 'man'
stocknumber between 20 and 100
stocknumber in(10,20,30)
```

order by stockid desc(asc) --------- 排序，desc-降序，asc-升序
order by 1,2 --------- by 列号
stockname = (select stockname from stock_information where stockid = 4)
--------- 子查询
--------- 除非能确保内层 select 只返回一个行的值，
--------- 否则应在外层 where 子句中用一个 in 限定符
select distinct column_name form table_name --------- distinct 指定检索独有的列值，不重复
select stocknumber ,"stocknumber + 10" = stocknumber + 10 from table_name
select stockname , "stocknumber" = count(*) from table_name group by stockname
--------- group by 将表按行分组,指定列中有相同的值
having count(*) = 2 --------- having 选定指定的组

select *
from table1, table2
where table1.id *= table2.id -------- 左外部连接，table1 中有的而 table2 中没有得以 null 表示
table1.id =* table2.id -------- 右外部连接

select stockname from table1
union [all] ----- union 合并查询结果集，all-保留重复行
select stockname from table2

insert

insert into table_name (Stock_name,Stock_number) value ("xxx","xxxx")
value (select Stockname , Stocknumber from Stock_table2)---value 为 select 语句

update

update table_name set Stockname = "xxx" [where Stockid = 3]
Stockname = default

Stockname = null
Stocknumber = Stockname + 4

delete

delete from table_name where Stockid = 3
truncate table_name ---------- 删除表中所有行，仍保持表的完整性
drop table table_name ---------- 完全删除表

alter table --- 修改数据库表结构

alter table database.owner.table_name add column_name char(2) null .....
sp_help table_name ---- 显示表已有特征
create table table_name (name char(20), age smallint, lname varchar(30))
insert into table_name select ..... ----- 实现删除列的方法（创建新表）
alter table table_name drop constraint Stockname_default ---- 删除 Stockname 的 default 约束

function(/*常用函数*/)

----统计函数----
AVG --求平均值
COUNT --统计数目
MAX --求最大值
MIN --求最小值
SUM --求和

--AVG
use pangu
select avg(e_wage) as dept_avgWage

```
from employee
group by dept_id
--MAX
--求工资最高的员工姓名
use pangu
select e_name
from employee
where e_wage =
(select max(e_wage)
from employee)
```

--STDEV()
--STDEV()函数返回表达式中所有数据的标准差

--STDEVP()
--STDEVP()函数返回总体标准差

--VAR()
--VAR()函数返回表达式中所有值的统计变异数

--VARP()
--VARP()函数返回总体变异数

----算术函数----

```
/***三角函数***/
SIN(float_expression) --返回以弧度表示的角的正弦
COS(float_expression) --返回以弧度表示的角的余弦
TAN(float_expression) --返回以弧度表示的角的正切
COT(float_expression) --返回以弧度表示的角的余切
/***反三角函数***/
ASIN(float_expression) --返回正弦是 FLOAT 值的以弧度表示的角
```

ACOS(float_expression) --返回余弦是 FLOAT 值的以弧度表示的角
ATAN(float_expression) --返回正切是 FLOAT 值的以弧度表示的角
ATAN2(float_expression1,float_expression2)
--返回正切是 float_expression1 /float_expres-sion2 的以弧度表示的角
DEGREES(numeric_expression)
--把弧度转换为角度返回与表达式相同的数据类型可为
--INTEGER/MONEY/REAL/FLOAT 类型
RADIANS(numeric_expression) --把角度转换为弧度返回与表达式相同的数据类型可为
--INTEGER/MONEY/REAL/FLOAT 类型
EXP(float_expression) --返回表达式的指数值
LOG(float_expression) --返回表达式的自然对数值
LOG10(float_expression)--返回表达式的以 10 为底的对数值
SQRT(float_expression) --返回表达式的平方根
/***取近似值函数***/
CEILING(numeric_expression) --返回>=表达式的最小整数返回的数据类型与表达式相同可为
--INTEGER/MONEY/REAL/FLOAT 类型
FLOOR(numeric_expression) --返回<=表达式的最小整数返回的数据类型与表达式相同可为
--INTEGER/MONEY/REAL/FLOAT 类型
ROUND(numeric_expression) --返回以 integer_expression 为精度的四舍五入值返回的数据
--类型与表达式相同可为 INTEGER/MONEY/REAL/FLOAT 类型
ABS(numeric_expression) --返回表达式的绝对值返回的数据类型与表达式相同可为
--INTEGER/MONEY/REAL/FLOAT 类型
SIGN(numeric_expression) --测试参数的正负号返回0 零值1 正数或-1 负数返回的数据类型
--与表达式相同可为 INTEGER/MONEY/REAL/FLOAT 类型
PI() --返回值为π 即 3.1415926535897936
RAND([integer_expression]) --用任选的[integer_expression]做种子值得出 0-1 间的随机浮点数

----字符串函数----
ASCII() --函数返回字符表达式最左端字符的 ASCII 码值
CHAR() --函数用于将 ASCII 码转换为字符
--如果没有输入 0 ~ 255 之间的 ASCII 码值 CHAR 函数会返回一个 NULL 值
LOWER() --函数把字符串全部转换为小写
UPPER() --函数把字符串全部转换为大写
STR() --函数把数值型数据转换为字符型数据

LTRIM() --函数把字符串头部的空格去掉
RTRIM() --函数把字符串尾部的空格去掉
LEFT(),RIGHT(),SUBSTRING() --函数返回部分字符串
CHARINDEX(),PATINDEX() --函数返回字符串中某个指定的子串出现的开始位置
SOUNDEX() --函数返回一个四位字符码
--SOUNDEX 函数可用来查找声音相似的字符串但 SOUNDEX 函数对数字和汉字均只返回 0 值
DIFFERENCE() --函数返回由 SOUNDEX 函数返回的两个字符表达式的值的差异
--0 两个 SOUNDEX 函数返回值的第一个字符不同
--1 两个 SOUNDEX 函数返回值的第一个字符相同
--2 两个 SOUNDEX 函数返回值的第一二个字符相同
--3 两个 SOUNDEX 函数返回值的第一二三个字符相同
--4 两个 SOUNDEX 函数返回值完全相同

QUOTENAME() --函数返回被特定字符括起来的字符串
/*select quotename('abc', '{') quotename('abc')
运行结果如下
-------------------------------{
{abc} [abc]*/

REPLICATE() --函数返回一个重复 character_expression 指定次数的字符串
/*select replicate('abc', 3) replicate('abc', -2)
运行结果如下
----------- -----------
abcabcabc NULL*/

REVERSE() --函数将指定的字符串的字符排列顺序颠倒
REPLACE() --函数返回被替换了指定子串的字符串
/*select replace('abc123g', '123', 'def')
运行结果如下
----------- -----------
abcdefg*/

SPACE() --函数返回一个有指定长度的空白字符串
STUFF() --函数用另一子串替换字符串指定位置长度的子串

----数据类型转换函数----
CAST() 函数语法如下
CAST() (<expression> AS <data_ type>[length])
CONVERT() 函数语法如下
CONVERT() (<data_ type>[length], <expression> [, style])

select cast(100+99 as char) convert(varchar(12), getdate())
运行结果如下
---------- -----------
199 Jan 15 2000

----日期函数----
DAY() --函数返回 date_expression 中的日期值
MONTH() --函数返回 date_expression 中的月份值
YEAR() --函数返回 date_expression 中的年份值
DATEADD(<datepart> ,<number> ,<date>)
--函数返回指定日期 date 加上指定的额外日期间隔 number 产生的新日期
DATEDIFF(<datepart> ,<number> ,<date>)
--函数返回两个指定日期在 datepart 方面的不同之处
DATENAME(<datepart> , <date>) --函数以字符串的形式返回日期的指定部分
DATEPART(<datepart> , <date>) --函数以整数值的形式返回日期的指定部分
GETDATE() --函数以 DATETIME 的缺省格式返回系统当前的日期和时间

----系统函数----
APP_NAME() --函数返回当前执行的应用程序的名称
COALESCE() --函数返回众多表达式中第一个非 NULL 表达式的值
COL_LENGTH(<'table_name'>, <'column_name'>) --函数返回表中指定字段的长度值
COL_NAME(<table_id>, <column_id>) --函数返回表中指定字段的名称即列名
DATALENGTH() --函数返回数据表达式的数据的实际长度
DB_ID(['database_name']) --函数返回数据库的编号
DB_NAME(database_id) --函数返回数据库的名称

HOST_ID() --函数返回服务器端计算机的名称
HOST_NAME() --函数返回服务器端计算机的名称
IDENTITY(<data_type>[, seed increment]) [AS column_name])
--IDENTITY() 函数只在 SELECT INTO 语句中使用用于插入一个 identity column 列到新表中
/*select identity(int, 1, 1) as column_name
into newtable
from oldtable*/
ISDATE() --函数判断所给定的表达式是否为合理日期
ISNULL(<check_expression>, <replacement_value>) --函数将表达式中的 NULL 值用指定值替换
ISNUMERIC() --函数判断所给定的表达式是否为合理的数值
NEWID() --函数返回一个 UNIQUEIDENTIFIER 类型的数值
NULLIF(<expression1>, <expression2>)
--NULLIF 函数在 expression1 与 expression2 相等时返回 NULL 值若不相等时则返回 expression1 的值

sql 中的保留字

action add aggregate all
alter after and as
asc avg avg_row_length auto_increment
between bigint bit binary
blob bool both by
cascade case char character
change check checksum column
columns comment constraint create
cross current_date current_time current_timestamp
data database databases date
datetime day day_hour day_minute
day_second dayofmonth dayofweek dayofyear
dec decimal default delayed
delay_key_write delete desc describe
distinct distinctrow double drop
end else escape escaped
enclosed enum explain exists

fields file first float
float4 float8 flush foreign
from for full function
global grant grants group
having heap high_priority hour
hour_minute hour_second hosts identified
ignore in index infile
inner insert insert_id int
integer interval int1 int2
int3 int4 int8 into
if is isam join
key keys kill last_insert_id
leading left length like
lines limit load local
lock logs long longblob
longtext low_priority max max_rows
match mediumblob mediumtext mediumint
middleint min_rows minute minute_second
modify month monthname myisam
natural numeric no not
null on optimize option
optionally or order outer
outfile pack_keys partial password
precision primary procedure process
processlist privileges read real
references reload regexp rename
replace restrict returns revoke
rlike row rows second
select set show shutdown
smallint soname sql_big_tables sql_big_selects
sql_low_priority_updates sql_log_off sql_log_update sql_select_limit
sql_small_result sql_big_result sql_warnings straight_join
starting status string table
tables temporary terminated text
then time timestamp tinyblob
tinytext tinyint trailing to
type use using unique
unlock unsigned update usage
values varchar variables varying
varbinary with write when
where year year_month zerofill

查看全文

常用 SQL 命令和 ASP 编程

在进行数据库操作时，无非就是添加、删除、修改，这得设计到一些常用的 SQL 语句，如下：

SQL 常用命令使用方法：

(1) 数据记录筛选：

sql="select * from 数据表 where 字段名=字段值 order by 字段名 [desc]"

sql="select * from 数据表 where 字段名 like %字段值% order by 字段名 [desc]"

sql="select top 10 * from 数据表 where 字段名 order by 字段名 [desc]"

sql="select * from 数据表 where 字段名 in (值 1,值 2,值 3)"

sql="select * from 数据表 where 字段名 between 值 1 and 值 2"

(2) 更新数据记录：

sql="update 数据表 set 字段名=字段值 where 条件表达式"

sql="update 数据表 set 字段 1=值 1,字段 2=值 2 …… 字段 n=值 n where 条件表达式"

(3) 删除数据记录：

sql="delete from 数据表 where 条件表达式"

sql="delete from 数据表" (将数据表所有记录删除)

(4) 添加数据记录：

sql="insert into 数据表 (字段 1,字段 2,字段 3 …) valuess (值 1,值 2,值 3 …)"

sql="insert into 目标数据表 select * from 源数据表" (把源数据表的记录添加到目标数据表)

(5) 数据记录统计函数：

AVG(字段名) 得出一个表格栏平均值

COUNT(*|字段名) 对数据行数的统计或对某一栏有值的数据行数统计
MAX(字段名) 取得一个表格栏最大的值
MIN(字段名) 取得一个表格栏最小的值
SUM(字段名) 把数据栏的值相加

引用以上函数的方法：

sql="select sum(字段名) as 别名 from 数据表 where 条件表达式"
set rs=conn.excute(sql)

用 rs("别名") 获取统的计值，其它函数运用同上。

(5) 数据表的建立和删除：

CREATE TABLE 数据表名称(字段 1 类型 1(长度),字段 2 类型 2(长度) ……)

例：CREATE TABLE tab01(name varchar(50),datetime default now())

DROP TABLE 数据表名称 (永久性删除一个数据表)

在 ASP 编程时，下面这些语句是必须知道的：

1.连接数据库
a. ASP 与 Access 数据库连接：

```
<%@ language=VBs cript%>
<%
dim conn,mdbfile
mdbfile=server.mappath("数据库名称.mdb")
set conn=server.createobject("adodb.connection")
conn.open "driver={microsoft access driver (*.mdb)};uid=admin;pwd=数据库密码;dbq="&mdbfile
%>
```

b. ASP 与 SQL 数据库连接：

```
<%@ language=VBs cript%>
<%
dim conn
set conn=server.createobject("ADODB.connection")
con.open "PROVIDER=SQLOLEDB;DATA SOURCE=SQL 服务器名称或 IP 地址;UID=sa;PWD=数据库密码;DATABASE=数据库名称
%>
```

建立记录集对象：

set rs=server.createobject("adodb.recordset")
rs.open SQL 语句,conn,3,2

2. 记录集对象的方法：

rs.movenext 将记录指针从当前的位置向下移一行
rs.moveprevious 将记录指针从当前的位置向上移一行
rs.movefirst 将记录指针移到数据表第一行
rs.movelast 将记录指针移到数据表最后一行
rs.absoluteposition=N 将记录指针移到数据表第 N 行
rs.absolutepage=N 将记录指针移到第 N 页的第一行
rs.pagesize=N 设置每页为 N 条记录
rs.pagecount 根据 pagesize 的设置返回总页数
rs.recordcount 返回记录总数
rs.bof 返回记录指针是否超出数据表首端，true 表示是， false 为否
rs.eof 返回记录指针是否超出数据表末端，true 表示是， false 为否
rs.delete 删除当前记录，但记录指针不会向下移动
rs.addnew 添加记录到数据表末端
rs.update 更新数据表记录

附：

常数 常数值 说明
——————————————————————————————————
adLockReadOnly 1 缺省值，Recordset 对象以只读方式启动，无法运行 AddNew、Update 及 Delete 等方法
adLockPrssimistic 2 当数据源正在更新时，系统会暂时锁住其他用户的动作，以保持数据一致性。
adLockOptimistic 3 当数据源正在更新时，系统并不会锁住其他用户的动作，其他用户可以对数据进行增、删、改的操作。
adLockBatchOptimistic 4 当数据源正在更新时，其他用户必须将 CursorLocation 属性改为 adUdeClientBatch 才能对数据进行增、删、改的操作。

分类: (一般分类) :: 评论 (0) :: 静态链接网址 :: 引用 (0)
<!-- -->

SQLSERVER2000 功略(4)---自动化管理

发表人:kendy517 | 发表时间: 2007 年二月 09 日, 11:56

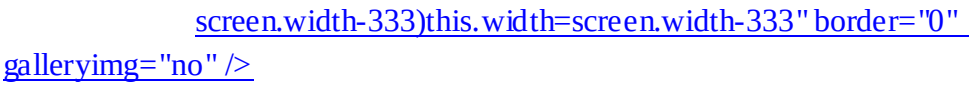
面对大量反复的工作，即使一个优秀的管理员也会感到很大的负担。但为了保持服务器维持在最佳运行状态，这些又时不得不作的工作。这时，SQLSERVER 想你所想，提供了自动化的管理措施。在我们工作之前，我们需要了解自动化背后的好助手—SQLSERVER Agent,这个就是 SQLSERVER 的代理程序，他是运行在后台的服务，可谓是幕后英雄！那他有什么功能呢，如下：
① 作业调度。
② 执行作业。
③ 产生报警。
④ 在指定的事件发生时，通知指定的管理员或操作员。

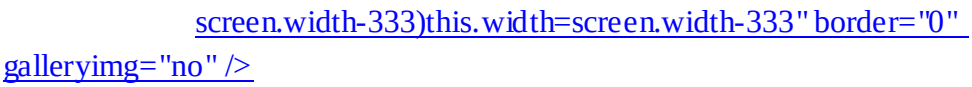
说到这里，我们就谈谈自动化中的角色：操作员，作业，警报。
操作员：所谓的操作员只是人名与其通信方式（可有电子邮件地址，呼叫器或 NET SEND 的目的地三种）的设置而已。只要将服务器管理人员的联络方式都以操作员设置妥当时，当 SQLSERVER 遇到各种需要处理的状况，SQLSERVER Agent 就会通过指定的方式通知管理人员前来完成，以达到及时的排除各种问题。
作业：作业是 SQLSERVER Agent 诸多功能中最常用的。所谓作业就是可让 SQLSERVERAgent 自动执行的一组操作，它既可以自动执行（这个就涉及到调度），也可以手动执行。在定义作业必须注意以下内容：作业的名称，作业的种类，作业的所有者，作业的说明文字。
警报：就是指针对特殊的事件，要发信给操作员的设置。在作业的设置中的“通知“页面中指定了要将作业执行状态通知操作员时，其实就是在设置警报（如下图）

screen.width-333)this.width=screen.width-333" border="0" galleryimg="no" />
在定义警报时，可有两种设置事件的方式，一种是指明了针对某特定事件，也就是只要在服务器发生该事件。另一种是指定事件的严重度（Serverity）,只要是达到指定严重程度的所有事件，SQLSERVER Agent 就会发信通知操作员。
了解了概念，就是动手的时候了，这些我们就以图片给与说明，第一个我们新建“操作员“，过程如下：

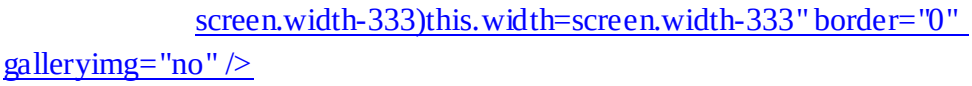
screen.width-333)this.width=screen.width-333" border="0" galleryimg="no" />

screen.width-333)this.width=screen.width-333" border="0" galleryimg="no" />
接着新建“作业“过程如下：





再来看看“警报“的创建过程：



(注意在建立警报的“响应”栏目里，对应着相应的作业和操作员)

谈过了这些功能，是不是觉得真是很方便呀，当然了良好的运用不仅仅就这简单的几步，这需要大家在应用中适时的总结以适应我们的需要。现在，我们再谈论一个 SQLSERVER
的另一项不错的功能—脚本。

所谓的利用脚本，就是可以将数据库对象或服务器中的其他对象以脚本（Script）方式存入文件，也就是存成 SQL 程序以供我们继续利用。

同样脚本也可以用于这里的自动化管理。例如：我们可以将操作员，作业和警报等对象存成 Script 文件后，再拿到别的服务器上加以利用。这是不是在不同的服务器上创建相同的操作员等重复的工作了。这就是脚本的效率所在。

如果需要存为脚本文件，可以在企业管理器中右击相应的对象在“所有任务“/“生成 SQL 脚本”命令，设置确定即可。

谈了上面的自动化管理的方式是很实用的，但似乎是不是缺乏全面性，要为自己的服务器构建一套完善的自动化维护机制，可能需要花费不少心力，但是将大部分的日常维护工作适度的自动化，多少可以减轻管理上的负担，为此
SQLSERVER 还提供了一项可以将整套数据库维护工作自动化的功能，这就是数据库的维护计划。

我们可以在企业管理器中执行“工具“/数据库维护计划器，这样就见到了如下界

面，依次设置即可。
这次的讨论就说到这里，关于自动化管理，我不用说大家都明白，合理的设置会给你的工作带来什么样的影响，好好体会，让工作更加轻松！

分类: (一般分类) :: 评论 (0) :: 静态链接网址 :: 引用 (0)
<!-- -->

SQLSERVER2000 功略(3)－数据库的登陆及其用户权限

发表人:kendy517 | 发表时间: 2007 年二月 09 日, 11:54

SQLSERVER2000 功略(3)－数据库的登陆及其用户权限
SQLSERVER2000 功略－数据库的登陆及其用户权限
说过了两章纯粹的理论，是不是有乏味的感觉，好，我们从这一讲就进入实质的管理阶段。首先，我们讨论的是 SQLSERVER 安全管理中的登陆和权限问题。SQLSERVER 的登陆是安全管理的第一关。这个决定了那些人可以使用 SQLSERVER（就像进屋时的房门钥匙）。关于登陆控制，SQLSERVER 则采用了两种不同的检查方式：windows 验证和 windows，sqlserver 混合验证。前者就是我们登陆 windows 时的用户帐号，而后者则是在保留前者的前提下也允许 SQLSERVER 中建立的登陆帐号。这也是为用户提供方便的选择。
我们拥有了登陆的帐号，就可以进入 SQLSERVER 的控制中心，那我们可以做些什么哪？这个肯定是有控制的进行的。SQLSERVER 中的访问权限分为两方面，分别是：服务器本身的操作权限和数据库的访问权限。我们分别加以简介：服务器的操作权限是从真个服务器来说的，是面向所有数据库的定义。例如：是否有权创建或者删除登陆帐号，是否可以建立数据库等等。这些也就是通过设置服务器角色（server role）来实现的。SQLSERVER 内建了七个服务器角色，分别代表七种服务器工作的操作权。
数据库的访问权限就是针对某一个数据库而言的。他可以分为两层，第一层是限制何人访问数据库，第二层限制可访问数据库中的那些表，视图和存储过程，以及是否可以用 select,insert,update 等等。这个权限往往是通过数据库角色得以实现的。
（服务器角色不可以自定义，数据库角色可以自定义）
仅仅知道上面这些概念是不行的，因为 SQLSERVER 中的权限是比较复杂的，因为彼此之间可能存在某些权链。所以仅仅概念是不容易说清的，我们讨论的就是基础的东西，懂得了这些在真正工作中才可以更好的掌握管理技能。我们简单的谈谈权链的问题吧！
由于每个对象都有他自己的拥有权信息，这个对象可能和其他对象存在一定的依存关系，比如：一个视图是从另外一个视图建立过来的。这些相互依存的对象关系可能拉的很长，因此相依存的对象的拥有权就形成一个由上而下的链结，这在 SQLSERVER 中就称作拥有权链（Ownership Chain）。正是由于这样，可能所产生的访问权限变得非常复杂。所以，我们在这里提出以视图和存储过程来限制访问范围。通过视图确定用户可以访问的内容，然后把视图的相关权限赋予具体的用户，这样可以有效的减轻权限设置的工作量（特别是用户访问范围跨度较大的情况下）。除了视图，我们也可以自定义存储过程来对数据的访问加以限制，即

把一组可查询某特定数据的语句存成存储过程，然后将其 execute 权限设置给用户，用户就可以执行他来取得数据了。这样省去了很多复杂的授权过程，是不是很方便呀！
简单的说了说这些，我们再来谈谈 SQLSERVER 中特殊的东西：sa 帐号，public 数据库角色，guest 数据库用户。
Sa 是 SQLSERVER 默认的管理员帐号，主要是为了向下兼容而设置的。这个也就是安装好 SQLSERVER 时就有的帐号，而且权限极大，所以在安装过程中最好把他的密码进行设置，不要为空。这个帐号相当于服务器角色中的 system administrator，而且不可以删除。
Public 数据库角色是一个公共的数据库角色，也就是说当你新建一个数据库角色的时候，默认的就属于这个角色。这个也是不可删除的。所以为了安全期间，我们对 public 数据库对象不要赋予过多的权限（除非有这个需要）。
Guest 是一个特殊的数据库用户，类似于 windows NT/2000 下的 guest 帐号，都是给未经授权的用户提供某种程度的访问权限。如果不需要这个用户可以将其删除。
说了这些概念，其实现都是通过“企业管理器”实现的，我们就这些谈谈用相应的系统存储过程：
登陆管理：sp_addlogin 新增帐号
sp_droplogin 删除帐号
sp_grantlogins 允许某 windows 帐号访问
sp_denylogins 禁止某 windows 帐号访问
sp_revokelogins 删除某 windows 帐号访问
sp_helplogin 查看帐号信息
sp_addsrvrolemember 将某用户添加到服务器角色
sp_dropsrvrolemember 将某用户从数据库角色中删除
sp_helpsrvrole 查看服务器角色的信息

数据库用户管理：sp_grantaccess 建立数据库用户
sp_revokedbaccess 删除数据库用户
sp_helpuser 查看用户信息

数据库角色管理：sp_addrole 建立数据库角色
sp_addrolemember 将用户加入数据库角色
sp_helprole 查看数据库角色信息
sp_helprolemember 查看某数据库角色的所有成员
sp_droprole 删除角色
sp_droprolemember 删除角色中的某一成员
好了，看了上面的不要感觉麻烦呀，这可是让你高效工作的宝贝呀，熟能生巧，我们一起联系多了，其实掌握就觉得很轻松了！继续努力！：）

分类: (一般分类) :: 评论 (0) :: 静态链接网址 :: 引用 (0)
<!-- -->

SQLSERVER2000 攻略(1)—工作流程篇

发表人:kendy517 | 发表时间: 2007 年二月 09 日, 11:51

SQLSERVER2000 是 windows 平台上一个重要的数据管理工具。了解数据库的朋友对他一定很熟悉。我们就此在这里作一系列的讨论，希望通过这次的交流我们对 SQLSERVER2000 有个深刻的理解。首先，我们探讨一下他的工作流程，这可是工作的基础呀，好，我们开始了！

简单的说，SQLSERVER2000 是个采用主从结构的关系数据库系统。哈，是不是太简单，毕竟现在市面上的数据库产品几乎都是这个结构。我们来深入一点，所谓 SQLSERVER 的主从结构（Client/Server）,就是由 SQLSERVER 扮演存放数据和提供数据给客户端的角色，当用户要取用数据时，则可调用各种不同的客户端应用程序，通过 SQLSERVER 所支持的接口，向 SQLSERVER 提供请求，然后取得数据库中的数据并返回给用户。这个过程应该好理解吧。

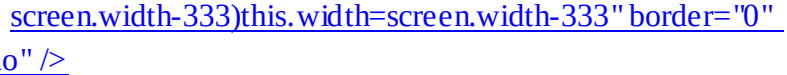
看了这个是不是觉得很 easy 呀！其实，这只是比较传统的双层式 Client/Server 结构，近年来三层式或者多层式主从结构的应用正在流行。例如：配合微软的 BackOffice 中的其他服务器产品，即可建立多层主从运算模式。
了解了基本的数据流程，那是不是要问是什么实现客户端与服务器之间的沟通哪？这个是个比较有意义的问题。我们知道了作什么，就应改知道他是怎么做的。所谓，打破沙锅问到底！我们说一下工作原理：SQLSERVER 提供了多种不同的接口让客户端（应用程序）顺利与服务器进行交流，在此接口的基础上，由 NET-Library(网络函数库)建立两边 IPC（Interprocess Commication）通讯管道来进行工作。具体的说就是：当客户端程序使用某种接口向服务器提出查询请求时，数据库接口驱动程序调用客户端的 Net-library,而 Net-library 则调用 IPC 应用程序接口产生 IPC,通过 IPC 送达服务端，服务端接受请求后经 SQLSERVER 处理，以同样的方式将结果送回客户端。（这个过程有两个情况，Client 和 Server 在同一台计算机上，Net-library 会使用 local IPC,如果是网络存取，则会有双方所使用的网络协议产生 Remote IPC）图示如下：
了解了外部工作机制，咱们看看服务端是怎么分工的，毕竟多数的数据工作是由服务器完成的。整体上说，整个 SQLSERVER 的数据管理系统分为如下几个组件：
▲Open Data Service(ODS):负责处理由 Net-library 送来的查询请求，ODS 会调用 SQLSERVER 的内部功能来完成客户端的需求。
▲SQL Server Service:这个 SQLSERVER 的“心脏”负责管理我们的数据库，以及所有建立，查询和修改的数据库的操作。此数据库管理系统的核心又分为几个单元，其中最主要的两个单元就是负责解 SQL 语法和安排查询执行计划等工作的关系引擎（Relational Engine）,以及负责管理数据库文件和文件读写等工作的存储引擎（Storage Engine）。这两个组件之间则是通过 OLE DB 接口来通信的。
▲SQL SERVER Agent(SQL SERVER 代言程序)：可以说此组件是重要性仅次于 SQL SERVER Service 的组件，这个服务提供了有关工作编排，以及服务器有问题时通知管理员等相关功能的支持。
▲MS DTC Service:如果将 SQL SERVER 安装在多台服务器上，要让客户端能够在单个事务中对不同的服务器（数据源）进行处理，就须借助 MS DTC(Microsoft Distributed Transaction Coordinator)Service 来负责这类分布式事务的协调工作，

以确保在各个数据库上的更改操作能正确完成。
了解了上面的过程，是不是有点印象了，来我们理清一下思路，问一下自己：客户端的工作请求怎么传递给服务器的？服务器又是怎么处理相应的请求的？认真想想这两个问题，其实你会觉得原来“不过如此”!^_^

分类: (一般分类) :: 评论 (0) :: 静态链接网址 :: 引用 (0)
<!-- -->

SQLSERVER 中使用存储过程(Stored Procedure)

发表人:kendy517 | 发表时间: 2007 年二月 09 日, 11:50
随着 SQLSERVER 不断的学习，认识的深入，我们会发现越来越多的功能呈现在我们面前，这些功能都是十分强大的，在我们应用中发挥着十分有意义的作用，也因此感知作为一个大型的数据库管理系统的魅力所在。从这一篇开始，我们着手讨论“存储过程“，”触发器“，”游标“，”事务“和”锁“的应用，如果熟练的掌握了这些，我想驾驭 SQLSERVER 的能力将是一个新的层次。学习是为了应用，掌握就是为了解决问题。希望通过我们一起的学习，我们都可以很好的利用这个工具为我们服务。这是我们学习的目的也是我们的奋斗目标。好，我们继续吧！
说起存储过程，我们先了解他的概念，这是我们讨论必走的第一步：存储过程就是将所需要的工作，预先以 SQL 程序写好，命名后进行保存，以后需要作这些工作时可以使用 EXECUTE 指令来调用，即可自动完成相应任务。这里的存储过程也是自动化的一个方面，总之就是为了方便管理的一种措施。让我来说说他的优点吧：
1．执行效率高（这点不容置疑）
2．统一的操作流程：也就是通过存储过程的操作避免了一些操作过程中可能无意中认为的错误，只要确定了制作存储过程时是正确地，以后在调用过程中就不用担心了。大家使用时流程是一样的。
3．重复使用
4．安全性：这一点我们在数据库的安全策略里讨论过，可以参考前边的文章。也就是说：我们可以利用存储过程作为数据存储的管道。可以让客户在一定的范围内对数据进行操作。另外，存储过程是可以加密的，这样别人就看不到他的内容了。
存储过程分为三类：
系统存储过程（System stored Procedure）sp_开头，为 SQLSERVER 内置存储过程：



扩展存储过程（Extended stored Procedure），也就是外挂程序，用于扩展 SQLSERVER 的功能，以 sp_或者 xp_开头，以 DLL 的形式单独存在。

screen.width-333)this.width=screen.width-333" border="0" galleryimg="no" />
Δ（观察上面的你会发现系统存储过程和扩展存储过程都是在 master 数据库中。sp_开头的可是全局的，任何一个数据库都可以直接调用的。）
用户定义的存储过程（User-defined stored Procedure），这个就是用户在具体的数据库中自己定义的，名字最好不要以 sp_和 xp_开头，防止混乱。
了解了基本概念，就到应用的阶段了。
首先创建一个存储过程(在 pubs 数据库中)，我们命名为 MyProce 示例代码如下(功能为向 stores 表中插入 stor_id,stor_name 两个字段值)：

```
create procedure MyProce
@param1 char(4), @param2 varchar(40) --定义参数，作为存储过程的接口
with encryption --存储过程加密
as insert stores (stor_id,stor_name) values(@param1,@param2)
go
```

我们如此调用：

```
exec MyProce '111111','Leijun'book' --参数赋值，调用存储过程
```

用企业管理器创建如图所示：

screen.width-333)this.width=screen.width-333" border="0" galleryimg="no" />
如果我们要修改，可以查看相应的存储过程的"属性"，如上图在"文本"窗体中修改。
但是注意我们这个因为用了 with encryption 语句，所以，打开时将有下面的提示，不允许查看，这也就是加密。

screen.width-333)this.width=screen.width-333" border="0" galleryimg="no" />
看了上面的是不是有所了解了，其实创建时还有其他的参数可以使用，我们这里只是一个简单的例子，更多的应用需要我们在实践中不断的总结，这样才能更加

灵活的应用。 下面我们再来看一个创建的例子(这个的作用是在 authors 表中查找一个人名，表中把一个名字分为两字段存储了，如果查到了，打印“查有此人 ID:”及其 au_id 字段值)：

```
CREATE procedure SearchMe
@param1 varchar(10),@param2 varchar(30)
as
select @param2=au_id
from authors
where au_fname+au_lname=@param1
if @@rowcount>0 --全局变量，记录影响到的行
print'查有此人 ID:'+@param2
GO
```

我们这样执行：Exec SearchMe ‘leijun’,null
由于查找到了所以显示如下：

screen.width-333)this.width=screen.width-333" border="0" galleryimg="no" />

例子就简单列举这些，因为他的灵活性很大，所以我们只简单的说明一下，如果要更好的利用，这就需要我们不断的摸索了。
下面我们谈谈使用过程中我了解到的注意事项：
1．在存储过程中，有些建立对象的语句是不可使用的：create default,create trigger,create procedure,create view,create rule.
2．在同一数据库中，不同的所有者可以建立相同名称的对象名。例如：a.sample,b.sample,c.sample 三个数据表可以同时存在。如果存储过程中未指明对象的所有者（例如存储过程中的语句 select * from sample，这句中的 sample 没有指明所有者），在执行的过程中默认的所有者查找顺序是：相应的存储过程的建立者－>相应数据库的所有者。如果这个查找过程中没有把所有者确定下来，系统就要报错。
（这里我额外插一句：如果需要严密的数据操作，在任何操作中尽量加上所有者，例如 leijun.sample）
3．在存储过程名称前边添加＃或者＃＃，所建立的存储过程则是“临时存储过程“（＃是局部临时存储过程，＃＃是全局临时存储过程）。
上面的都是一些容易忽略的，特别是第二条，我们一定的认真思考，也许这些有意无意的忽略是我们造成错误的根源！！
转载自：leijun

分类: (一般分类) :: 评论 (0) :: 静态链接网址 :: 引用 (0)
<!-- -->

[Php+APACHE+MySQL][或在 windows 已装 IIS 下的]配置

发表人:kendy517 | 发表时间: 2007 年二月 09 日, 09:30
前段时间的文章:
http://www.blueidea.com/bbs/NewsDetail.asp?id=2372560

可以一看,但是其对本篇价值是 1%.希望本篇能补充之缺陷.

一：下载安装程序

Apache 可以从 http://httpd.apache.org/download.cgi 下载
PHP 可以从 http://www.php.net 下载.
mysql 在 www.mysql.com/下载.

二 安装程序(以下以 2k 为例说明,其他 windows 系统的类似)

首先如果您安装了 IIS 的话,Internet 服务管理器--->默认 web 站点-->右击--->停止.

1: 先是 Apache,有 4.2M 左右,没啥说的,点击安装到 D 盘,为了方西系统重装,D 盘方便.一路会很容易的安装的成功.

2: php 5.1.X 的安装版是 2.44m 左右,可以点击安装,选择服务器的时候选择 apache,不过,没有组建扩展,还有下载扩展的 dll 文件包(有的 19 m 左右,不一定),复制到 C:WINNT 目录下面的.可以直接复制到 d 盘文件下面也可以,笔者不知道为什么选择了第一种.

3: MySQL 我下载的是 4.1.7 版本, 36.7M 左右,也是一路点击,然后启动之.

三：必要设置

如果我们要使用 CGI 二进制文件,那么我们要将如下指令插入到 Apache 的 httpd.conf 配置文件中，以设置 CGI 二进制文件：

PHP 以 CGI 方式安装到 Apache:

```
ScriptAlias /php/ "D:/php/"
AddType application/x-httpd-php .php
Action application/x-httpd-php "/php/php.exe"
```

如果我们想把 PHP 作为 Apache 2.0 的模块，那么就一定要移动 php4ts.dll 到 winnt/system32（Windows NT/2000）或 windows/system32（Windows XP），覆盖原有文件（如果有的话），对于 PHP 5，这个文件是 php5ts.dll。然后我们要

插入如下两行到 httpd.conf 中，以使我们的 PHP 作为 Apache 的 PHP-Module 安装：

PHP 以模块方式安装到 Apache:

; For PHP 5 do something like this:
LoadModule php5_module "c:/php/php5apache2.dll"
AddType application/x-httpd-php .php

添加这 2 行代码到 httpd.conf,httpd.conf 文件可通过计算机搜索找出.哈哈....

再在里面找出:

DocumentRoot "D:/web/web/php"
相关行,可以修改成自己的 php 网站根目录.

如果你把那些扩展都复制到 C:winnt 下面的话.

测试:<html>
<body>
<? phpinfo(); ?>
</body>
<html>
应该:正常可以看到的.

关于连接 MySQL 的,请看:
http://www.blueidea.com/bbs/NewsDetail.asp?id=2408054

为了您的 IIS 能继续工作,请 在 internet 信息服务 ---> 默认 web 站点-->右击属性---> IP 地址:192.168.1.105 , 端口: 81 ,否则会提示端口以已使用之类的信息.

完毕,有问题请马上提出,早安.
天气:大雨,ccdot 发表于 2006-1-21 2:48:43,阅读了 460 次,共有个 1 回复.

花了大半天的时间，终于把全新的 php 平台建了起来，历经了太多错误和反复，看了网上无数篇教材，可仍错漏百出，唉，真不知道那些写教程的人在写什么。不过苍天不负有心人，摸索了好久，最终还是成功了.：) ..写下来，以后也有个参照。

调试环境如下：
OS:win2000 sp4
apache_2052-win32 (msi 格式)
php-5.0.2-Win32 (zip 包)
mysql-5.0.1-alpha-snapshot-win (zip 包)

所有程序(apache,php,mysql)安装在 F:/server 目录下。

●解压 php 到 f:/serverphp 文件夹中。
PHP.ini 的一些配置
. doc_root = F:/phpf //默认 php 文档位置，设为 apache 中一样，偶没设，却也行!!
. extension_dir = F:/server/php/ext //扩展文件位置,用作支持 mysql，一定要设正确
. session.save_path = c:/temp //session 存储位置
. cgi.force_redirect=0 //如果出现 Security Alert! The PHP CGI cannot be accessed directly 的警告，设置此值
. register_global，把它设置为 on. //避免 post 变量不能传递
. display_errors = on //显示错误消息

●装好 apache 到 f:/server/apache2，首先让 Apache2 能 支持 php5

1、打开 apache 的配置文件，添加如下两行(php 安装在 F:/server/php 下)
. AddType application/x-httpd-php .php
. LoadModule php5_module F:/server/php/php5apache2.dll
这样就已经可以支持了 php 了，不知网上的教程里为什么还要把那些 dll 文件复制得满天飞（版本是相同的），郁闷。

2、httpd.conf 的一些其他必要设置：
. 索引页名称 DirectoryIndex index.php
. 默认文档位置，设置 DocumentRoot 路径

3、网上流传的配置，我没有 copy，但也成功了。
. 把 php5ts.dll 复制到 winnt 目录或 system 目录中

●安装 mysql 到 f:/servermysql 中。运行 bin 目录中 winmysqladmin.exe。设置你的用户名和密码，端口不必更改。

●接着让 PHP 支持 Mysql 数据库
. 将安装目录中的 php.ini-recommended 复制到 winnt 目录中，改名为 php.ini

查找 [MySQL] 字段内的内容.修改如下.

mysql.default_port = 3306
// 这里是 MYSQL 的端口.

mysql.default_host = localhost
// 这里是本地主机.

mysql.default_user = root
// 这里是用户

mysql.default_password = abcdefg
// 这里是密码

修改到这里就可以让 Php 与 MYSQL 关联了.
最后启动 PHP 对 MYSQL 模块支持.

查找 extension=php_mysql.dll 字串.删除掉前面的 ; 号

extension=php_mysql.dll

请注意.如果不启动 PHP 对 MYSQL 模块支持.将无法使 PHPMYADMIN 正常工作.将提示:

cannot load MySQL extension, please check PHP Configuration
无法装入 MySQL 扩展，请检查 PHP 的配置。

可是我设置完这样，重启 apache 却出现找不到 libmySQL.dll 动态链接库的提示，几经折磨才发现，必须把 php 中的 libmysql.dll 复制到 apache 中的 bin 目录中才 ok（网上教材可没有）。

●让 apache 也支持 mysql
把 php 目录下的 libmySQL.dll 复制到 apache 的 bin 目录下。

ok，调试 phpinfo()，一切正常。可是我写了句 echo "我爱你";，就出现了中文乱码。老问题，从网上搜得一段话，顺利解决了问题。

●原因很简单，由于 Apache 它默认的编码是西欧编码，你只要在“httpd.conf”中找到“AddDefaultCharset ISO-8859-1”将它修改为“AddDefaultCharset GB2312 ”，保存然后重新启动 Apache 再试，可能有的朋友的已经解决了，但还是有很多的朋友的网站依然是乱码，包括我的，这是什么原因呢，网上也有介绍把它改为“AddDefaultCharset off ”，可能在你的网站也不一定行得通，实质问题在于它是由网页制作软件引起的，如果你使用的是 Macromedia 公司的 Dreamweaver 系列软件制作的，由于它的编码为“gb2312”，所以就会出现乱码，因此你只要在你网页的代码中找到“”，把“gb2312”改为“GB2312”然后保存网页，再试试看，行了吧！

是不是太简单了，但是我为了把这乱码问题解决，也足足花了二天的时间才找到真正的原因，如果你做好了，别忘记请我喝酒呀！

分类: (一般分类) :: 评论 (0) :: 静态链接网址 :: 引用 (0)
<!-- -->

MSSQL 经典语句

发表人:kendy517 | 发表时间: 2007 年二月 09 日, 09:24

1.按姓氏笔画排序: Select * From TableName Order By CustomerName Collate Chinese_PRC_Stroke_ci_as

2.数据库加密:select encrypt('原始密码')
select pwdencrypt('原始密码')
select pwdcompare('原始密码','加密后密码') = 1--相同；否则不相同 encrypt('原始密码')
select pwdencrypt('原始密码')
select pwdcompare('原始密码','加密后密码') = 1--相同；否则不相同

3.取回表中字段:declare @list varchar(1000),@sql nvarchar(1000)
select @list=@list+','+b.name from sysobjects a,syscolumns b where a.id=b.id and a.name='表 A'
set @sql='select '+right(@list,len(@list)-1)+' from 表 A'
exec (@sql)

4.查看硬盘分区:EXEC master..xp_fixeddrives

5.比较 A,B 表是否相等:if (select checksum_agg(binary_checksum(*)) from A)
=
(select checksum_agg(binary_checksum(*)) from B)
print '相等'
else
print '不相等'

6.杀掉所有的事件探察器进程:DECLARE hcforeach CURSOR GLOBAL FOR
SELECT 'kill '+RTRIM(spid) FROM master.dbo.sysprocesses
WHERE program_name IN('SQL profiler',N'SQL 事件探查器')
EXEC sp_msforeach_worker '?'

7.记录搜索:开头到 N 条记录
Select Top N * From 表

N 到 M 条记录(要有主索引 ID) Select Top M-N * From 表 Where ID in (Select Top M ID From 表) Order by ID Desc

---

N 到结尾记录

Select Top N * From 表 Order by ID Desc

8.如何修改数据库的名称:sp_renamedb 'old_name', 'new_name'

9：获取当前数据库中的所有用户表 select Name from sysobjects where xtype='u' and status>=0

10：获取某一个表的所有字段 select name from syscolumns where id=object_id('表名')

11：查看与某一个表相关的视图、存储过程、函数 select a.* from sysobjects a, syscomments b where a.id = b.id and b.text like '%表名%'

12：查看当前数据库中所有存储过程 select name as 存储过程名称 from sysobjects where xtype='P'

13：查询用户创建的所有数据库 select * from master..sysdatabases D where sid not in(select sid from master..syslogins where name='sa')
或者
select dbid, name AS DB_NAME from master..sysdatabases where sid <> 0x01

14：查询某一个表的字段和数据类型 select column_name,data_type from information_schema.columns
where table_name = '表名'

[n].[标题]: Select * From TableName Order By CustomerName

[n].[标题]: Select * From TableName Order By CustomerName

from：http://tb.blog.csdn.net/TrackBack.aspx?PostId=585515
分类: (一般分类) :: 评论 (0) :: 静态链接网址 :: 引用 (0)
<!-- -->

触发器-MSSQL 常用操作

发表人:kendy517 | 发表时间: 2007 年二月 09 日, 09:18
不再新开文章.

这里只打算讲解四部分了，也就最简单、最常用的四部分。

1、触发器。

定义： 何为触发器？在 SQL Server 里面也就是对某一个表的一定的操作，触发某种条件，从而执行的一段程序。触发器是一个特殊的存储过程。
常见的触发器有三种：分别应用于 Insert , Update , Delete 事件。(SQL Server 2000 定义了新的触发器，这里不提）

我为什么要使用触发器？比如，这么两个表：

```
Create Table Student( --学生表
StudentID int primary key, --学号
....
)

Create Table BorrowRecord( --学生借书记录表
BorrowRecord int identity(1,1), --流水号
StudentID int , --学号
BorrowDate datetime, --借出时间
ReturnDAte Datetime, --归还时间
...
)
```

用到的功能有:
1.如果我更改了学生的学号,我希望他的借书记录仍然与这个学生相关(也就是同时更改借书记录表的学号);
2.如果该学生已经毕业，我希望删除他的学号的同时，也删除它的借书记录。
等等。

这时候可以用到触发器。对于 1，创建一个 Update 触发器：

```
Create Trigger truStudent
On Student
for Update
As
if Update(StudentID)
begin

Update BorrowRecord
Set StudentID=i.StudentID
From BorrowRecord br , Deleted d ,Inserted i
Where br.StudentID=d.StudentID

end
```

理解触发器里面的两个临时的表：Deleted , Inserted 。注意 Deleted 与 Inserted 分别表示触发事件的表“旧的一条记录”和“新的一条记录”。
一个 Update 的过程可以看作为：生成新的记录到 Inserted 表，复制旧的记录到 Deleted 表，然后删除 Student 记录并写入新纪录。

对于 2，创建一个 Delete 触发器
Create trigger trdStudent
On Student
for Delete
As
Delete BorrowRecord
From BorrowRecord br , Delted d
Where br.StudentID=d.StudentID

从这两个例子我们可以看到了触发器的关键：A.2 个临时的表；B.触发机制。
这里我们只讲解最简单的触发器。复杂的容后说明。
事实上，我不鼓励使用触发器。触发器的初始设计思想，已经被“级联”所替代.

来自:http://www.innerv.com/blogview.asp?logID=512

分类: (一般分类) :: 评论 (0) :: 静态链接网址 :: 引用 (0)
<!-- -->

MSSQL 数据库导出和导入

发表人:kendy517 | 发表时间: 2007 年二月 09 日, 09:17

SQL Server 导出导入数据方法

2004-02 余枫 一、导出导入 SQL Server 里某个数据库 1.在 SQL Server 企业管理器里选中要转移的数据库，按鼠标右键，选所有任务->备份数据库。 2.备份 选数据库-完全，目的 备份到 按添加按钮 文件名 在 SQL Server 服务器硬盘下输入一个自定义的备份数据库文件名(后缀一般是 bak) 重写 选重写现有媒体 最后按确定按钮。 如果生成的备份数据库文件大于 1M，要用压缩工具压缩后再到 Internet 上传输。 3.通过 FTP 或者 remote desktop 或者 pcanywhere 等方法 把第二步生成的备份数据库文件或者其压缩后的文件传到目的 SQL Server 数据库，如果有压缩要解压。 4.目的 SQL Server 数据库如果还没有此数据库，先创建一个新的数据库； 然后选中这个新创建的数据库，按鼠标右键，选所有任务->还原数据库 还原->从设备->选择设备->磁盘->添加(找到要导入的备份数据库文件名）->确定 还原备份集->数据库-完全 最后按确定按钮。完全的数据库导入成功了。(如果在已经存在的 SQL Server 数据库上还原数据库可能遇到有还有其它人正在使用它而恢复操做失败， 可以去看 ->管理->当前活动->锁/对象->找到数据库下锁的进程号->到查询分析器里用 kill 进程号杀掉这些锁， 然后再做还原)
注意：如果在原有的目的 SQL Server 数据库上从备份文件(*.bak)还原数据库 会

把已经存在的表、存储过程等数据库对象全部替换成最近这次导入的备份数据库里的内容。 如果一定要还原备份文件(*.bak)里部分数据,需要另外建一个新数据库, 其逻辑名称和数量同备份文件(*.bak)里数据库的逻辑名称和数量一致； 新数据库的物理文件名称取得一定要和备份文件(*.bak)里数据库的物理文件不一样才行。 二、导出导入 SQL Server 里某个表 1.没有防火墙，同一个局域网里或不在同一个局域网里,但通过 Internet 可以互相访问 在 SQL Server 企业管理器里选中目的数据库 ，按鼠标右键，选所有任务->导入数据-> 弹出数据转换服务导入/导出向导窗口->下一步-> 选数据源-> 数据源（用于 SQL Server 的 Microfost OLE DB 提供程序）-> 服务器(可选择局域网内能访问到的所有 SQL Server 服务器,或者直接输入 IP 地址)-> 选择使用 windows 身份验证还是使用 SQL Serve 身份验证(输入数据库的用户名和密码)-> 数据库(可选择上面选中 SQL Server 服务器上所有权限范围内的数据库)->下一步-> 选择目的->目的（用于 SQL Server 的 Microfost OLE DB 提供程序）-> 服务器(默认为上一步里选中的导出服务器，也可以选其它局域网内能访问到的所有 SQL Server 服务器,或者直接输入 IP 地址)-> 目的数据库(可选择上面选中 SQL Server 服务器上所有权限范围内的数据库)->下一步-> 制定表复制或查询->选从源数据库复制表和视图(也可

———————————————————————————————————————