Transact SQL 常用语句以及函数

Transact SQL 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
     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 stocksex = '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

常用函数(function)

转换函数

convert(数据类型,值,格式)

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() ( AS [ length ])
```

CONVERT() 函数语法如下

```
CONVERT() ([ length ], [, 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(, ,)

-----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(, ,)

-----函数返回两个指定日期在 datepart 方面的不同之处

DATENAME(, ------函数以字符串的形式返回日期的指定部分

DATEPART(, ------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(, ----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([, 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(, --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(),

----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