1. TRY\_CONVERT(), CONVERT(), CAST()
2. CASE WHEN...
3. IIF()
4. TRIGGER
5. .. CATCH
6. SUSER\_NAME() or just SYSTEM\_USER
7. HOST\_NAME()
8. NEWID()
9. @@IDENTITY
10. ERROR\_NUMBER(), ERROR\_SEVERITY(), ERROR\_STATE(), ERROR\_PROCEDURE(), ERROR\_LINE(), ERROR\_MESSAGE()
11. BEGIN TRAN, ROLLBACK TRAN, COMMIT TRAN
12. CURSOR
13. XML / JSON output from query
14. XML parameter into procedure
15. TRUNCATE TABLE DELETE FROM *table*
16. BULK INSERT
17. JOIN, LEFT JOIN, RIGHT JOIN, FULL JOIN, CROSS JOIN
18. Indexing
19. DROP TABLE IF EXISTS *tableName*
20. WHILE()
21. BEGIN ... END
22. IF ... ELSE
23. How to design a dB schema

1. cast/ convert 转换 / get date()

SELECT CAST(PaymentTotal AS INT) / 10,

CAST(PaymentTotal AS INT) % 10

FROM dbo.Invoices

SELECT CAST(PaymentTotal AS INT) / 10,

CAST(PaymentTotal AS INT) % 10,

'$' + CAST(PaymentTotal AS varchar(20)),

CONVERT(CHAR(8), PaymentDate, 1),

CONVERT(CHAR(8), PaymentDate, 2),

CONVERT(VARCHAR(100), PaymentTotal, 1),

CAST(PaymentDate AS VARCHAR(19)),

CAST(PaymentDate AS VARCHAR(11)),

GETDATE(),

CAST(GETDATE() AS VARCHAR(19))

FROM dbo.Invoices

CONVERT(VARCHAR(10),GETDATE(),110)

CONVERT(VARCHAR(11),GETDATE(),106)

CONVERT(VARCHAR(24),GETDATE(),113)

SELECT

CASE WHEN TRY\_CONVERT(float, 'test') IS NULL

THEN 'Cast failed'

ELSE 'Cast succeeded'

END AS Result;

GO

Result

------------

Cast failed

(1 row(s) affected)

SET DATEFORMAT dmy;

SELECT TRY\_CONVERT(datetime2, '12/31/2010') AS Result;

GO

Result

----------------------

NULL

(1 r SELECT TRY\_CONVERT(DECIMAL(5,2), '12.345'); --Returns 12.34

SELECT TRY\_CONVERT(DECIMAL(5,2), '123.45'); --Returns 123.45

SELECT TRY\_CONVERT(DECIMAL(5,2), '123.456'); --Returns 123.46

SELECT TRY\_CONVERT(DECIMAL(5,2), '999.99'); --Returns 999.99

SELECT TRY\_CONVERT(DECIMAL(5,2), '1000.00'); --Returns Null

SELECT TRY\_CONVERT(DECIMAL(5,2), '1234.5'); --Returns Null

SELECT TRY\_CONVERT(DECIMAL(5,2), '12345'); --Returns Null

SELECT TRY\_CONVERT(DECIMAL(5,2), 'Fred'); --Returns Null

ow(s) affected)

2. case when

SELECT DISTINCT

[Status] =

CASE WHEN Balance = 0 THEN 'Paid in Full'

WHEN Balance > 0 THEN 'Outstanding balance'

ELSE   'Refund'

END

,VendorID

,Balance

FROM dbo.vwInvoices

3.

DECLARE @a int = 45, @b int = 40;

SELECT IIF ( @a > @b, 'TRUE', 'FALSE' ) AS Result;

Result

--------

TRUE

(1 row(s) affected)

4. TRIGGER

CREATETRIGGER trigger\_name

ON table|view

FOR|AFTER|INSTEADOF [DELETE][,INSERT][,UPDATE]

AS

Sql\_statement[…n]

SQL语句

GO

Create Trigger truStudent

On Student --在Student表中创建触发器

for Update --为什么事件触发

As --事件触发后所要做的事情

if Update(StudentID)

begin

Update BorrowRecord

Set StudentID=i.StudentID

From BorrowRecord br , Deleted d ,Inserted i --Deleted和Inserted临时表

Where br.StudentID=d.StudentID

end

Create trigger trdStudent

On Student

for Delete

As

Delete BorrowRecord

From BorrowRecord br , Delted d

Where br.StudentID=d.StudentID

5

BEGIN TRY

  SELECT GETDATE()

  SELECT 1/0--Evergreen divide by zero example!

END TRY

BEGIN CATCH

  SELECT 'There was an error! ' + ERROR\_MESSAGE()

  RETURN

END CATCH;

==================================

ALTER PROC usp\_AccountTransaction

  @AccountNum INT,

  @Amount DECIMAL

AS

BEGIN

  BEGIN TRY --Start the Try Block..

    BEGIN TRANSACTION -- Start the transaction..

      UPDATE MyChecking SET Amount = Amount - @Amount

        WHERE AccountNum = @AccountNum

      UPDATE MySavings SET Amount = Amount + @Amount

        WHERE AccountNum = @AccountNum

    COMMIT TRAN -- Transaction Success!

  END TRY

  BEGIN CATCH

    IF @@TRANCOUNT > 0

      ROLLBACK TRAN --RollBack in case of Error

    -- you can Raise ERROR with RAISEERROR() Statement including the details of the exception

    RAISERROR(ERROR\_MESSAGE(), ERROR\_SEVERITY(), 1)

  END CATCH

END

GO

6.7具有参数化筛选器的合并发布使用 SUSER\_SNAME() 和/或 HOST\_NAME() 函数筛选数据。 函数在新建发布向导或 **“发布属性”** 对话框中指定。

默认情况下，HOST\_NAME() 函数返回连接到发布服务器的计算机的名称。 在使用参数化筛选器时，通常在向导的此页上提供值来覆盖此值。 这样，HOST\_NAME() 函数将返回指定的值而非计算机的名称。 有关详细信息，请参阅[参数化行筛选器](https://docs.microsoft.com/zh-cn/sql/relational-databases/replication/merge/parameterized-filters-parameterized-row-filters?view=sql-server-2017)的“覆盖 HOST\_NAME() 值”部分

8.

The following example uses NEWID() to assign a value to a variable declared as the **uniqueidentifier** data type. The value of the **uniqueidentifier** data type variable is printed before the value is tested.

9.

USE AdventureWorks;

GO

--Display the value of LocationID in the last row in the table.

SELECT MAX(LocationID) FROM Production.Location;

GO

INSERT INTO Production.Location (Name, CostRate, Availability, ModifiedDate)

VALUES ('Damaged Goods', **5**, **2.5**, GETDATE());

GO

SELECT **@@IDENTITY** AS 'Identity';

GO

--Display the value of LocationID of the newly inserted row.

SELECT MAX(LocationID) FROM Production.Location;

GO

用select @@identity得到上一次插入记录时自动产生的ID

@@IDENTITY 返回为当前会话的所有作用域中的任何表最后生成的标识值。

10. 4.11

1. USE AdventureWorks;
2. GO
3. BEGIN TRANSACTION;
5. BEGIN TRY
6. *-- Generate a constraint violation error.*
7. DELETE FROM Production.Product
8. WHERE ProductID = 980;
9. END TRY
10. BEGIN CATCH
11. SELECT
12. ERROR\_NUMBER() AS ErrorNumber,
13. ERROR\_SEVERITY() AS ErrorSeverity,
14. ERROR\_STATE() as ErrorState,
15. ERROR\_PROCEDURE() as ErrorProcedure,
16. ERROR\_LINE() as ErrorLine,
17. ERROR\_MESSAGE() as ErrorMessage;
19. IF @@TRANCOUNT > 0
20. ROLLBACK TRANSACTION;
21. END CATCH;
22. IF @@TRANCOUNT > 0
23. COMMIT TRANSACTION;
24. GO
25. CREATE TABLE LogTable
26. (
27. ID int identity(**1**,**1**),--错误序号
28. ErrorNumber int,--错误号
29. ErrorSeverity int,--严重性
30. ErrorState int,--错误状态号
31. ErrorProducure varchar(**200**),--出现错误的存储过程或 触发器的名称
32. ErrorLine int,--导致错误的例程中的行号
33. ErrorMessage varchar(**200**)--错误消息的完整文本
34. )

11.

begin Transaction 可以理解成新建一个还原点。

commit Transaction 提交这个自begin tran开始的修改

rollback Transaction 表示还原到上个还原点。

EXEC TEST\_PROC '文综','包括历史，地理，政治','政治','文综的一门'   
CREATE PROCEDURE [dbo].[TEST\_PROC]   
@A\_Name NVARCHAR(20), -- A表姓名   
@A\_Remark NVARCHAR(4000), -- A表备注   
@B\_Name NVARCHAR(20), -- B表姓名   
@B\_Remark NVARCHAR(4000) -- B表备注   
AS   
BEGIN TRY   
BEGIN TRAN   
-- 在A表中插入数据   
INSERT INTO [dbo].[A]   
( [A\_Name]   
, [A\_Remark] )   
VALUES   
( @A\_Name   
, @A\_Remark )   
-- 在B表中插入数据   
INSERT INTO [dbo].[B]   
( [A\_ID]   
, [B\_Name]   
, [B\_Remark] )   
VALUES   
( @@IDENTITY -- 返回最后插入的标识值   
, @B\_Name   
, @B\_Remark )   
COMMIT TRAN   
END TRY   
BEGIN CATCH   
ROLLBACK TRAN   
INSERT INTO [dbo].[ErrorLog]   
( [EL\_Procedure] -- 异常存储过程名称   
, [EL\_OperateTime] ) -- 报异常时间   
VALUES   
( 'TEST\_PROC'   
, CONVERT(DATETIME,GETDATE(),20) )   
END CATCH

12.  4.11

declare @id int

 declare @name varchar(50)

 declare cursor1 cursor for         --定义游标cursor1

 select \* from table1               --使用游标的对象(跟据需要填入select文)

 open cursor1                       --打开游标

 fetch next from cursor1 into @id,@name  --将游标向下移1行，获取的数据放入之前定义的变量@id,@name中

 while @@fetch\_status=0           --判断是否成功获取数据 进入循环

 begin

 update table1 set name=name+'1'

 where id=@id                           --进行相应处理(跟据需要填入SQL文)

 fetch next from cursor1 into @id,@name  --将游标向下移1行 相当于for(int i;i<n;i++) 中的i+1

 end

 close cursor1                   --关闭游标

 deallocate cursor

**游标一般格式：**  
DECLARE 游标名称 CURSOR FOR

SELECT 字段1,字段2,字段3,...

FROM 表名 WHERE ...  
OPEN 游标名称  
FETCH NEXT FROM 游标名称 INTO 变量名1,变量名2,变量名3,...  
WHILE @@FETCH\_STATUS=0  
        BEGIN  
                  SQL语句执行过程... ...  
                  FETCH NEXT FROM 游标名称 INTO 变量名1,变量名2,变量名3,...  
        END  
CLOSE 游标名称  
DEALLOCATE 游标名称 (删除游标)

USE Northwind  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gifGO  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif--定义游标  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gifDECLARE myCursor CURSOR FOR  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gifSELECT LastName, FirstName FROM Employees  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gifWHERE TitleOfCourtesy ='Mr.'  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif--打开游标  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gifOPEN myCursor  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif--逐行读取  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gifFETCH NEXT FROM myCursor  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif--@@FETCH\_STATUS，FETCH 语句成功  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gifWHILE **@@FETCH\_STATUS** = **0**  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gifBEGIN    
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif   FETCH NEXT FROM myCursor  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gifEND  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif--关闭游标  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gifCLOSE myCursor  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gif--释放游标  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gifDEALLOCATE myCursor  
http://images.csdn.net/syntaxhighlighting/OutliningIndicators/None.gifGO

13,========3.19!!!

14.

RUNCATE TABLE dbo.Guests

INSERT INTO dbo.Guests VALUES (4, 'Tommy')

INSERT INTO dbo.Guests VALUES (4, 'Jack')

DECLARE @xml AS XML = '<Guests>

<add inviteeId = "2" name = "Mike" />

<add inviteeId = "2" name = "Don" />

<add inviteeId = "2" name = "Lucille" />

<delete guestId = "2" />

<update guestId = "1" name = "Tom" />

</Guests>'

--============================================================== INSERT

SELECT \* FROM dbo.Guests

INSERT INTO Guests (inviteeId, guestName)

SELECT ent.value('@inviteeId', 'int'),

ent.value('@name', 'varchar(50)')

FROM @xml.nodes('/Guests/add') foo(ent)

--============================================================== DELETE

SELECT \* FROM dbo.Guests

DELETE FROM Guests WHERE guestId IN (

SELECT ent.value('@guestId','int')

FROM @xml.nodes('/Guests/delete') foo(ent)

)

--============================================================== UPDATE

UPDATE dbo.Guests

SET guestName = tbl.name

FROM (

SELECT [gid] = ent.value('@guestId', 'int'),

[name] = ent.value('@name', 'varchar(50)')

FROM @xml.nodes('/Guests/update') foo(ent)

) AS tbl

WHERE guestId = tbl.gid

SELECT \* FROM dbo.Guests

15

 1、**drop table**表名称                         eg: drop table  dbo.Sys\_Test  
   2、**truncate table**表名称                     eg: truncate  table dbo.Sys\_Test                    
   3、**delete from** 表名称 **where** 列名称 = 值      eg: delete from dbo.Sys\_Test where test='test'

二、drop，truncate，delete区别

    1、drop (删除表)：删除内容和定义，释放空间。简单来说就是**把整个表去掉**.以后要新增数据是不可能的,除非新增一个表。

       drop语句将删除表的结构被依赖的约束（constrain),触发器（trigger)索引（index);依赖于该表的存储过程/函数将被保留，但其状态会变为：invalid。

    2、truncate (清空表中的数据)：删除内容、释放空间但不删除定义(**保留表的数据结构**)。与drop不同的是,只是清空表数据而已。

       注意:truncate 不能删除行数据,要删就要把表清空。

    3、delete (删除表中的数据)：delete 语句用于**删除表中的行**。delete语句执行删除的过程是每次从表中删除一行，并且同时将该行的删除操作作为事务记录在日志中保存

       以便进行进行回滚操作。

       truncate与不带where的delete ：只删除数据，而不删除表的结构（定义）

    4、truncate table 删除表中的所有行，但表结构及其列、约束、索引等保持不变。新行标识所用的计数值重置为该列的种子。如果想保留标识计数值，请改用delete。

       如果要删除表定义及其数据，请使用 drop table 语句。    
    5、对于由foreign key约束引用的表，不能使用truncate table ，而应使用不带where子句的delete语句。由于truncate table 记录在日志中，所以它不能激活触发器。

    6、执行速度，一般来说: drop> truncate > delete。

    7、delete语句是数据库操作语言(dml)，这个操作会放到 rollback segement 中，事务提交之后才生效；如果有相应的 trigger，执行的时候将被触发。

             truncate、drop 是数据库定义语言(ddl)，操作立即生效，原数据不放到 rollback segment 中，不能回滚，操作不触发 trigger。

分类: [MySQL](https://www.cnblogs.com/fjl0418/category/1443468.html)

15 Bulk insert

BULK INSERT Rates

FROM 'c:\temp\US\_Personal\_Savings\_Rates\_1960-2017.txt'

WITH (

FIELDTERMINATOR = '\t',

ROWTERMINATOR = '\n',

FIRSTROW = 2,

KEEPIDENTITY

)

16. join

18 CREATE INDEX index\_name

ON table\_name (column\_name)

CREATE INDEX PersonIndex

ON Person (LastName, FirstName)

CREATE INDEX PersonIndex

ON Person (LastName)

19.

DROP TABLE IF EXISTS myschema.mytable;

DROP TABLE IF EXISTS [table\_name]

20.

GO

WHILE (SELECT AVG(price) FROM titles) < $30

BEGIN

UPDATE titles

SET price = price \* 2

SELECT MAX(price) FROM titles

IF (SELECT MAX(price) FROM titles) > $50

BREAK

ELSE

CONTINUE

END

set @i=1

while @i<30

begin

insert into test (userid) values(@i)

set @i=@i+1

end

21.

if @i > 100 -- 判断

-- 如果条件成立，执行这个语句块

begin

selete \* from DevieInfo where DeviceId = @i

print '筛选完毕！'

end

else

-- 如果条件不成立，执行这个语句块

begin

delete from DevieInfo where DeviceId = @i

print '删除完毕！'

end

=============================