무료 전자 책

# 배우기

# Microsoft SQL Server

Free unaffiliated eBook created from
**Stack Overflow contributors.**

#sql-server

1
1: Microsoft SQL Server 2
2
2
Examples 2
INSERT / SELECT / UPDATE / DELETE : 2
4
4
5
5
5
5
6
6
7
7
7
8
9
TRUNCATE TABLE 9
9
9
2: Azure SQL 11
Examples 11
Azure SQL 11
Azure SQL 11
Azure SQL 11
Elastic Azure SQL 12
3: bcp ( ) 13
13
Examples 13
( ) 13

4: CASE ..... 14
..... 14
Examples ..... 14
CASE ..... 14
CASE ..... 14
5: CLUSTERED COLUMNSTORE ..... 15
Examples ..... 15
CLUSTERED COLUMNSTORE ..... 15
..... 15
CLUSTERED COLUMNSTORE ..... 15
6: CREATE VIEW ..... 16
Examples ..... 16
CREATE VIEW ..... 16
..... 16
INNER JOIN ..... 16
CREATE INDEXED VIEW ..... 17
..... 17
..... 18
7: DBCC ..... 19
Examples ..... 19
DBCC ..... 19
DBCC ..... 19
DBCC ..... 20
DBCC ..... 20
DBCC ..... 21
8: DBMAIL ..... 22
..... 22
Examples ..... 22
..... 22
..... 22
HTML ..... 22
9: FOR XML PATH ..... 24

24

Examples 24

Hello World XML 24

24

XPath 24

FOR XML PATH 25

10: GROUP BY 27

Examples 27

27

GROUP 28

, 28

30

ROLLUP CUBE GROUP BY 31

11: JSON 33

Examples 33

JSON 33

JSON 33

JSON 33

JSON 33

JSON 34

JSON JOIN 34

JSON 34

12: JSON 35

Examples 35

JSON 35

JSON PATH 35

JSON ( ) 35

INCLUDE_NULL_VALUES 36

ROOT 36

JSON AUTO 36

JSON 37

13: JSON 38

Examples 38
JSON 38
JSON 38
JSON JSON 38
FOR JSON JSON 39
FOR JSON JSON 39
14: JSON SQL 40
Examples 40
JSON 40
JSON ISJSON 40
JSON 40
JSON 40
JSON 41
15: Microsoft SQL Server Studio 42
Examples 42
42
42
42
16: MS SQL Server DDL 45
Examples 45
45
45
45
45
46
17: NULLs 47
47
47
Examples 47
NULL 47
ANSI NULLS 48
ISNULL () 48

null null ..... 49
COALESCE () ..... 49
NULL ..... 49
18: OPENJSON ..... 51
Examples ..... 51
: JSON ..... 51
JSON ..... 51
JSON ..... 51
JSON ..... 52
JSON ..... 52
19: OVER ..... 54
..... 54
..... 54
Examples ..... 54
OVER ..... 54
..... 55
..... 55
NTILE ..... 56
20: SCOPE_IDENTITY () ..... 57
..... 57
Examples ..... 57
..... 57
21: SELECT ..... 58
..... 58
Examples ..... 58
SELECT ..... 58
WHERE ..... 58
ORDER BY ..... 58
GROUP BY ..... 58
HAVING ..... 59
N ..... 59
OFFSET FETCH ..... 59

FROM ( ) 59
22: SQL Server Management Studio (SSMS) 61
61
Examples 61
IntelliSense 61
23: SQL Server 62
Examples 62
SQL Server 2016 62
XML SQL Server 2008/2012/2014 63
T-SQL XML 63
24: SQL Server JSON 64
64
64
64
Examples 64
FOR JSON JSON 64
JSON 64
CROSS APPLY OPENJSON JSON 65
JSON 66
FOR JSON JSON 67
OPENJSON JSON 67
25: SQL Server 68
Examples 68
STUFF 68
String_Agg 68
26: SQL Server 69
69
Examples 69
AS 69
= 69
69
AS 69

27: SQLCMD ..... 71
..... 71
Examples ..... 71
SQLCMD.exe ..... 71
28: SQLCMD txt ..... 72
..... 72
Examples ..... 72
SQLCMD ..... 72
29: STUFF ..... 73
..... 73
Examples ..... 73
STUFF () ..... 73
FOR XML ..... 73
( ) ..... 74
SQL ..... 74
STUFF () ..... 74
30: TEMP ..... 76
..... 76
Examples ..... 76
..... 76
..... 76
..... 77
31: TRY / CATCH ..... 78
..... 78
Examples ..... 78
TRY / CATCH ..... 78
try-catch ..... 78
try catch ..... 79
RAISERROR ..... 79
TRY / CATCH ..... 79
32: WHILE ..... 81
..... 81

Examples 81
While 81
min while 81
33: Windows SQL Server 82
Examples 82
82
34: 83
Examples 83
83
SQL / 83
83
83
35: 85
85
Examples 85
85
36: 86
Examples 86
86
86
37: 87
Examples 87
87
87
87
87
cmd 87
87
87
38: 88
88
Examples 88

COALESCE ..... 88
..... 88
null ..... 88
**39:** ..... **90**
..... 90
Examples ..... 90
..... 90
**40:** ..... **91**
Examples ..... 91
CLR ..... 91
SQL CLR .dll ..... 91
SQL Server CLR ..... 91
SQL Server CLR ..... 92
SQL Server CLR ..... 92
**41:** ..... **93**
..... 93
..... 93
Examples ..... 93
..... 93
..... **93**
..... **93**
**:** ..... **94**
CTE n ..... 94
CTE ..... 94
CTE ..... 95
CTE ..... 95
AS CTE ..... 96
**42:** ..... **98**
Examples ..... 98
..... 98
JSON ..... 98
.....

98

43: ..... 99

Examples ..... 99

..... 99

44: ..... 100

..... 100

Examples ..... 100

/ ..... 100

GUID ..... 100

..... 100

..... 100

..... 100

..... 101

45: (Hekaton) ..... 102

Examples ..... 102

..... 102

..... 102

..... 103

46: ..... 104

Examples ..... 104

Invoices ..... 104

47: ..... 105

..... 105

..... 105

Examples ..... 105

CONVERT ..... 105

FORMAT ..... 107

DateTime ..... 109

DATEADD ..... 110

..... 110

DATEDIFF ..... 111

..... 111

112
DateTime 112
112
CROSS 113
113
48: 120
120
120
Examples 120
CTE 120
120
49: 121
Examples 121
121
50: 123
Examples 123
123
51: 125
Examples 125
125
IIF 125
52: 126
Examples 126
IF 126
IF 126
IF..ELSE 126
IF ... ELSE IF ... 127
IF ... ELSE 127
53: SQL Server (2000 - 2016) 128
128
Examples 128
SQL Server 2000 - 2016 128

54: ..... 131
Examples ..... 131
BULK INSERT ..... 131
..... 131
OPENROWSET (BULK) ..... 131
OPENROWSET (BULK) ..... 131
OPENROWSET (BULK) json ..... 132
55: ..... 133
..... 133
Examples ..... 133
..... 133
..... 134
..... 134
..... 134
..... 135
..... 135
..... 135
56: ..... 136
Examples ..... 136
TRY PARSE ..... 136
..... 136
..... 137
..... 137
..... 137
57: ..... 139
..... 139
..... 139
Examples ..... 139
..... 139
..... 139
REPLACE ..... 139
58: ..... 141
.....

141

Examples....141

....141

CREATE USER....141

CREATE ROLE....141

....141

**59: ....142**

....142

Examples....142

....142

....142

....142

**60: ....143**

....143

Examples....143

....143

....143

....144

....144

....145

....145

Windows ....145

....145

....145

....146

, , ....147

SQL ....147

....150

....150

**61: SQL....152**

Examples....152

SQL ....152

SQL ..... 152
SQL SQL ..... 152
SQL ..... 153
62: SQL ..... 154
..... 154
Examples ..... 154
SQL ..... 154
63: ..... 155
Examples ..... 155
..... 155
..... 155
random () ..... 155
..... 155
..... 155
64: ..... 157
..... 157
Examples ..... 157
..... 157
..... 157
65: ID ..... 159
Examples ..... 159
SCOPE_IDENTITY () ..... 159
@@ ..... 159
IDENT_CURRENT ( 'tablename') ..... 159
@@ IDENTITY MAX (ID) ..... 160
66: OLTP (Hekaton) ..... 161
Examples ..... 161
..... 161
.dll ..... 162
..... 162
..... 163
..... 164

67: ..... 165
..... 165
Examples ..... 166
..... 166
..... 166
..... 166
ASCII ..... 167
CharIndex ..... 167
..... 167
..... 167
..... 168
..... 169
..... 169
LTrim ..... 169
RTrim ..... 169
..... 170
NChar ..... 170
..... 170
PatIndex ..... 170
..... 170
..... 171
..... 171
String_Split ..... 171
..... 172
..... 173
Soundex ..... 173
..... 173
..... 174
String_escape ..... 175
68: ..... 177
..... 177
Examples ..... 177
..... 177

DML ..... 177
69: ..... 179
..... 179
Examples ..... 179
..... 179
SET ..... 179
SELECT ..... 180
..... 180
..... 181
..... 181
70: ..... 182
..... 182
..... 182
..... 182
Examples ..... 182
/ / MERGE ..... 182
CTE ..... 183
MERGE ..... 184
- ..... 184
EXCEPT ..... 185
71: ..... 187
Examples ..... 187
..... 187
72: ..... 188
..... 188
..... 188
Examples ..... 188
int UDT ..... 188
UDT ..... 188
UDT : ..... 188
UDT : ..... 188
73: ..... 190

Examples....190
....190
....190
....190
....190
....190
....191
....191
....191
....191
....191
74: ....193
Examples....193
1. ....193
2. ....193
3. ( )....193
4. ....194
5. TargetQueue ....194
75: ....196
....196
....196
Examples....196
....196
76: ....197
....197
....197
....197
Examples....197
()....197
DENSE_RANK ()....197
77: ....199
Examples....199

199
199
199
199
78: - TempDb 200
Examples 200
TempDb 200
TempDB 200
79: 201
Examples 201
201
201
201
201
80: 202
202
202
Examples 202
202
202
203
203
81: 204
204
Examples 204
204
205
206
207
208
208
209

209

82: ..... 211

..... 211

Examples ..... 211

Hello World INTO ..... 211

INSERT ..... 211

..... 211

..... 211

ID OUTPUT ..... 212

SELECT INSERT ..... 212

83: ..... 213

..... 213

Examples ..... 213

..... 213

84: ..... 215

Examples ..... 215

/ ..... 215

/ ..... 215

..... 215

..... 216

..... 216

85: ..... 217

Examples ..... 217

..... 217

86: ..... 219

Examples ..... 219

..... 219

..... 219

..... 220

..... 220

SQL Server ..... 220

, , ..... 220

87: ..... 222
..... 222
Examples ..... 222
..... 222
? ..... 222
(FOR SYSTEM_TIME AS OF ) ..... 222
SYSTEM_TIME ..... 222
FOR SYSTEM_TIME FROM ..... 223
FOR SYSTEM_TIME IN ( , ) ..... 223
FOR SYSTEM_TIME ALL ..... 223
SQL Server ..... 223
88: ..... 225
..... 225
..... 225
Examples ..... 225
..... 225
OUT ..... 226
..... 226
..... 226
out ..... 227
..... 227
If ... Else Insert into Operation ..... 227
SQL ..... 228
..... 229
..... 230
89: ..... 231
Examples ..... 231
A. , ..... 231
..... 231
..... 231
..... 232

90: 233
233
Examples 233
233
233
233
91: 234
234
Examples 234
ORDER BY 234
234
ORDER BY 234
235
92: 236
236
236
Examples 236
() 236
AVG () 236
MAX () 237
MIN () 237
() 237
GROUP BY Column_Name COUNT (Column_Name) 238
93: 240
240
240
Examples 240
ROW_NUMBER 240
OFFSET FETCH 240
241
SQL Server 241
SQL Server 2012/2014 241

SQL Server 2005 / 2008 / R2 ..... 241
SQL Server 2000 ..... 241
ORDER BY OFFSET FETCH NEXT SQL Server 2012/2014 ..... 242
94: ..... 243
Examples ..... 243
..... 243
..... 243
30 ..... 243
95: ..... 244
..... 244
..... 244
Examples ..... 244
..... 244
..... 244
96: ..... 246
Examples ..... 246
..... 246
97: ..... 247
Examples ..... 247
JOIN Hints ..... 247
..... 247
..... 248
UNION ..... 248
MAXDOP ..... 248
..... 248
98: ..... 249
..... 249
..... 249
Examples ..... 249
..... 249
..... 249
..... 250

99: ..... 251
..... 251
Examples ..... 251
..... 251
100: ..... 252
Examples ..... 252
..... 252
..... 252
( ) ..... 252
101: ..... 254
..... 254
..... 254
Examples ..... 254
..... 254
..... 254
" " ? ..... 254
..... 255
..... 255
..... 255
102: ..... 257
..... 257
Examples ..... 257
..... 257
103: ..... 258
..... 258
Examples ..... 258
..... 258
104: ..... 259
Examples ..... 259
..... 259
105: ..... 261
.....

261

Examples....261

....261

106: ....262

Examples....262

....262

....262

, , , , ....262

107: ....264

....264

....264

Examples....264

....264

108: ....265

Examples....265

Row_Number ()....265

109: / ....266

....266

....266

Examples....266

- ....266

PIVOT UNPIVOT (T-SQL)....266

PIVOT....268

110: ....270

Examples....270

....270

111: ....272

....272

....272

....272

Examples....272

TOP ..... 272
PERCENT ..... 272
FETCH ..... 272
112: ..... 273
Examples ..... 273
RLS ..... 273
RLS ..... 273
RLS ..... 274
113: / ..... 275
Examples ..... 275
..... 275
..... 277
..... 279

# 1: Microsoft SQL Server

SQL Server .

| | |
|---|---|
| SQL Server 2016 | 2016-06-01 |
| SQL Server 2014 | 2014-03-18 |
| SQL Server 2012 | 2011-10-11 |
| SQL Server 2008 R2 | 2010-04-01 |
| SQL Server 2008 | 2008-08-06 |
| SQL Server 2005 | 2005-11-01 |
| SQL Server 2000 | 2000-11-01 |

## Examples

### INSERT / SELECT / UPDATE / DELETE :

**D** ATA **M** anipulation anguage ( DML) `INSERT` , `UPDATE` `DELETE` :

```
-- Create a table HelloWorld

CREATE TABLE HelloWorld (
    Id INT IDENTITY,
    Description VARCHAR(1000)
)


-- DML Operation INSERT, inserting a row into the table
INSERT INTO HelloWorld (Description) VALUES ('Hello World')


-- DML Operation SELECT, displaying the table
SELECT * FROM HelloWorld


-- Select a specific column from table
SELECT Description FROM HelloWorld


-- Display number of records in the table
SELECT Count(*) FROM HelloWorld


-- DML Operation UPDATE, updating a specific row in the table
UPDATE HelloWorld SET Description = 'Hello, World!' WHERE Id = 1
```

```
-- Selecting rows from the table (see how the Description has changed after the update?)
SELECT * FROM HelloWorld


-- DML Operation - DELETE, deleting a row from the table
DELETE FROM HelloWorld WHERE Id = 1


-- Selecting the table. See table content after DELETE operation
SELECT * FROM HelloWorld
```

.

.

```
USE Northwind;
GO
SELECT TOP 10 * FROM Customers
ORDER BY CompanyName
```

`Northwind` `CompanyName` `Customer` 10 (Microsoft ).

| | CustomerID | CompanyName | ContactName | ContactTitle | Address | City |
|---|---|---|---|---|---|---|
| ▸ | ALFKI | Alfreds Futterkiste | Maria Anders | Sales Representative | Obere Str. 57 | Berlin |
| | ANATR | Ana Trujillo Emparedados y helados | Ana Trujillo | Owner | Avda. de la Constitución 2222 | México D.F. |
| | ANTON | Antonio Moreno Taquería | Antonio Moreno | Owner | Mataderos 2312 | México D.F. |
| | AROUT | Around the Horn | Thomas Hardy | Sales Representative | 120 Hanover Sq. | London |
| | BERGS | Berglunds snabbköp | Christina Berglund | Order Administrator | Berguvsvägen 8 | Luleå |
| | BLAUS | Blauer See Delikatessen | Hanna Moos | Sales Representative | Forsterstr. 57 | Mannheim |
| | BLONP | Blondesddsl père et fils | Frédérique Citeaux | Marketing Manager | 24, place Kléber | Strasbourg |
| | BOLID | Bólido Comidas preparadas | Martín Sommer | Owner | C/ Araquil, 67 | Madrid |
| | BONAP | Bon app' | Laurence Lebihan | Owner | 12, rue des Bouchers | Marseille |
| | BOTTM | Bottom-Dollar Markets | Elizabeth Lincoln | Accounting Manager | 23 Tsawassen Blvd. | Tsawassen |

`Use Northwind;` . [Database]. [Schema]. [Table] .

```
SELECT TOP 10 * FROM Northwind.dbo.Customers
ORDER BY CompanyName

SELECT TOP 10 * FROM Pubs.dbo.Authors
ORDER BY City
```

. "" `dbo` . . `dbo` . . .

( : `Date` ) .

```
-- descending order
SELECT TOP 10 [Date] FROM dbo.MyLogTable
ORDER BY [Date] DESC
```

( ). ( :

```
-- descending order
SELECT top 10 "Date" from dbo.MyLogTable
order by "Date" desc
```

. . .

```
-- descending order
SELECT top 10 "Date" from dbo.MyLogTable
where UserId='johndoe'
order by "Date" desc
```

. T-SQL NChar NVarchar N (:

```
SELECT TOP 10 * FROM Northwind.dbo.Customers
WHERE CompanyName LIKE N'AL%'
ORDER BY CompanyName
```

AL ( % . DOS ( DIR AL* ) ( : DIR AL* ). LIKE . .

. : Northwind Region . Region RegionDescription RegionDescription . RgionID RgionID ( Top 5 5 RgionID ).

```
SELECT TOP 5 Territories.*,
    Regions.RegionDescription
FROM Territories
INNER JOIN Region
    ON Territories.RegionID=Region.RegionID
ORDER BY TerritoryDescription
```

Territories RegionDescription Region . TerritoryDescription .

. . .

```
<TableName> [as] <alias>
```

as . .

```
SELECT TOP 5 t.*,
    r.RegionDescription
FROM Territories t
INNER JOIN Region r
    ON t.RegionID = r.RegionID
ORDER BY TerritoryDescription
```

x u . , Employee SupervisorId .

```
SELECT e.*,
    s.Name as SupervisorName -- Rename the field for output
FROM Employee e
INNER JOIN Employee s
    ON e.SupervisorId = s.EmployeeId
WHERE e.EmployeeId = 111
```

Join . ? UNION . . .

```
SELECT FirstName+' '+LastName as ContactName, Address, City FROM Employees
UNION
SELECT ContactName, Address, City FROM Customers
```

, . ( ) ( UNION ALL ). , , SELECT . SELECT FirstName LastName Employee ContactName
.

( ) . "" .

:

```
DECLARE @Region TABLE
(
  RegionID int,
  RegionDescription NChar(50)
)
```

. @ . DML , .

```
INSERT INTO @Region values(3,'Northern')
INSERT INTO @Region values(4,'Southern')
```

.

```
INSERT INTO @Region
SELECT * FROM dbo.Region WHERE RegionID>2;
```

dbo.Region @Region . . , .

```
SELECT * FROM Territories t
JOIN @Region r on t.RegionID=r.RegionID
```

Northern Southern . . .

: , 100 , Microsoft .

. SQL Server Management Studio .

```
PRINT 'Hello World!';
```

:

```
SELECT *
FROM table_name
```

* . SELECT WHERE .

( : `e` .

```
SELECT *
FROM Employees AS e
```

+ ". *"  .

```
SELECT e.*, d.DepartmentName
FROM Employees AS e
    INNER JOIN Department AS d
        ON e.DepartmentID = d.DepartmentID
```

.

```
SELECT * FROM [server_name].[database_name].[schema_name].[table_name]
```

/  .

,  `table_name`  .  .

:  `SELECT *`  .  ,  . SELECT  .

```
SELECT col1, col2, col3
FROM table_name
```

.

```
SELECT <column names>
FROM <table name>
WHERE <condition>
```

:

```
SELECT FirstName, Age
FROM Users
WHERE LastName = 'Smith'
```

.

```
SELECT FirstName, Age
FROM Users
WHERE LastName = 'Smith' AND (City = 'New York' OR City = 'Los Angeles')
```

```
UPDATE HelloWorlds
SET HelloWorld = 'HELLO WORLD!!!'
WHERE Id = 5
```

"HelloWorld"  "HELLO WORLD !!!" . HelloWorlds  "Id = 5" .

:  "where"  .

.

. .

`Score` 1 ( ).

```
UPDATE Scores
SET score = score + 1
```

UPDATE UPDATE .

Transact-SQL . .

-- .

```
-- This is a comment
SELECT *
FROM MyTable -- This is another comment
WHERE Id = 1;
```

/* */. .

```
/* This is
a multi-line
comment block. */
SELECT Id = 1, [Message] = 'First row'
UNION ALL
SELECT 2, 'Second row'
/* This is a one liner */
SELECT 'More';
```

SQL . SQL .

/* */ . .

```
/*
SELECT *
FROM CommentTable
WHERE Comment = '/*'
*/
```

. .

```
/*
SELECT *
FROM CommentTable
WHERE Comment = '/*'
*/  */
```

```
SELECT @@VERSION
```

MS SQL Server .

```
SELECT @@SERVERNAME
```

MS SQL Server .

```
SELECT @@SERVICENAME
```

MS SQL Server Windows .

```
SELECT serverproperty('ComputerNamePhysicalNetBIOS');
```

SQL Server . .

```
SELECT * FROM fn_virtualservernodes();
```

SQL Server . .

DML (Data Manipulation Language) . DML `UPDATE` , `TRUNCATE` , `INSERT` `DELETE` . .

DML . . DML . `ROLLBACK` `COMMIT` . `ROLLBACK` . `COMMIT` DML .

**:**

```
--Create a test table

USE [your database]
GO
CREATE TABLE test_transaction (column_1 varchar(10))
GO

INSERT INTO
 dbo.test_transaction
        ( column_1 )
VALUES
        ( 'a' )

BEGIN TRANSACTION --This is the beginning of your transaction

UPDATE dbo.test_transaction
SET column_1 = 'B'
OUTPUT INSERTED.*
WHERE column_1 = 'A'


ROLLBACK TRANSACTION  --Rollback will undo your changes
           --Alternatively, use COMMIT to save your results

SELECT * FROM dbo.test_transaction   --View the table after your changes have been run

DROP TABLE dbo.test_transaction
```

**:**

-

. . . . .

- . . .

```
DELETE
FROM Helloworlds
```

. . DROP TABLE , .

.

```
TRUNCATE TABLE HelloWords
```

DELETE .

1. .
2. IDENTITY .
3. TRUNCATE ( DELETE WHERE )

**TRUNCATE**

1. FOREIGN KEY TRUNCATE .
2. INDEXED VIEW
3. TRANSACTIONAL REPLICATION MERGE REPLICATION
4. .

[sic]

## TRUNCATE TABLE

```
TRUNCATE TABLE Helloworlds
```

Helloworlds . Delete from Table Delete from Table . Truncate where . Truncate .

ID ( : ID 1 ). ID C g , .

```
SELECT * INTO NewTable FROM OldTable
```

.

1. .
2. SELECT ... INTO . SELECT ... INTO . . INSERT INTO ... SELECT FROM .
3. , SELECT ... INTO . SELECT ... INTO .
4. ORDER BY . . SELECT ... INTO .
5. . SELECT ... INTO .

[ *sic* ]

table_name .

```
SELECT COUNT(*) AS [TotalRowCount] FROM table_name;
```

HEAP (index_id = 0) (index_id = 1) .

```
SELECT  [Tables].name                AS [TableName],
        SUM( [Partitions].[rows] )    AS [TotalRowCount]
FROM    sys.tables AS [Tables]
JOIN    sys.partitions AS [Partitions]
    ON  [Tables].[object_id]    =    [Partitions].[object_id]
    AND [Partitions].index_id IN ( 0, 1 )
--WHERE    [Tables].name = N'table name' /* uncomment to look for a specific table */
GROUP BY    [Tables].name;
```

. / .

Microsoft SQL Server : https://riptutorial.com/ko/sql-server/topic/236/microsoft-sql-server-

# 2: Azure SQL

## Examples

### Azure SQL

Azure SQL .

Azure SQL , (, ) (S0, S1, P4, P11 ) .

```
select @@version
SELECT DATABASEPROPERTYEX('Wwi', 'EDITION')
SELECT DATABASEPROPERTYEX('Wwi', 'ServiceObjective')
```

### Azure SQL

ALTER DATABASE Azure SQL .

```
ALTER DATABASE WWI
MODIFY (SERVICE_OBJECTIVE = 'P6')
-- or
ALTER DATABASE CURRENT
MODIFY (SERVICE_OBJECTIVE = 'P2')
```

.

40802, 16, 1, 1 '.....' '.....' . ' ' .

ALTER DATABASE .

### Azure SQL

Azure SQL Server .

```
ALTER DATABASE <<mydb>>
ADD SECONDARY ON SERVER <<secondaryserver>>
WITH ( ALLOW_CONNECTIONS = ALL )
```

( ). . master . ALLOW_CONNECTIONS ALL ( NO ) .

1 .

```
ALTER DATABASE mydb FAILOVER
```

.

```
ALTER DATABASE <<mydb>>
REMOVE SECONDARY ON SERVER <<testsecondaryserver>>
```

## Elastic  Azure SQL

SQL  SQL  .

```
CREATE DATABASE wwi
( SERVICE_OBJECTIVE = ELASTIC_POOL ( name = mypool1 ) )
```

.

```
CREATE DATABASE wwi
AS COPY OF myserver.WideWorldImporters
( SERVICE_OBJECTIVE = ELASTIC_POOL ( name = mypool1 ) )
```

Azure SQL  : https://riptutorial.com/ko/sql-server/topic/7113/azure-sql--

# 3: bcp ( )

(bcp) Microsoft SQL Server . bcp SQL Server .

## Examples

### ( )

```
REM Truncate table (for testing)
SQLCMD -Q "TRUNCATE TABLE TestDatabase.dbo.myNative;"

REM Import data
bcp TestDatabase.dbo.myNative IN D:\BCP\myNative.bcp -T -n

REM Review results
SQLCMD -Q "SELECT * FROM TestDatabase.dbo.myNative;"
```

bcp ( ) : https://riptutorial.com/ko/sql-server/topic/10942/bcp------

# 4: CASE

SQL Server case . "SELECT DATENAME (WEEKDAY, GETDATE ())" .

## Examples

### CASE

. case .

```
SELECT CASE DATEPART(WEEKDAY, GETDATE())
    WHEN 1 THEN 'Sunday'
    WHEN 2 THEN 'Monday'
    WHEN 3 THEN 'Tuesday'
    WHEN 4 THEN 'Wednesday'
    WHEN 5 THEN 'Thursday'
    WHEN 6 THEN 'Friday'
    WHEN 7 THEN 'Saturday'
END
```

### CASE

Searched Case . case .

```
DECLARE @FirstName varchar(30) = 'John'
DECLARE @LastName varchar(30) = 'Smith'

SELECT CASE
    WHEN LEFT(@FirstName, 1) IN ('a','e','i','o','u')
        THEN 'First name starts with a vowel'
    WHEN LEFT(@LastName, 1) IN ('a','e','i','o','u')
        THEN 'Last name starts with a vowel'
    ELSE
        'Neither name starts with a vowel'
END
```

CASE : https://riptutorial.com/ko/sql-server/topic/7238/case-

# 5: CLUSTERED COLUMNSTORE

## Examples

### CLUSTERED COLUMNSTORE

INDEX cci CLUSTERED COLUMNSTORE .

```
DROP TABLE IF EXISTS Product
GO
CREATE TABLE Product (
    ProductID int,
    Name nvarchar(50) NOT NULL,
    Color nvarchar(15),
    Size nvarchar(5) NULL,
    Price money NOT NULL,
    Quantity int,
    INDEX cci CLUSTERED COLUMNSTORE
)
```

COLUMSTORE .

CREATE CLUSTERED COLUMNSTORE INDEX .

```
DROP TABLE IF EXISTS Product
GO
CREATE TABLE Product (
    Name nvarchar(50) NOT NULL,
    Color nvarchar(15),
    Size nvarchar(5) NULL,
    Price money NOT NULL,
    Quantity int
)
GO
CREATE CLUSTERED COLUMNSTORE INDEX cci ON Product
```

### CLUSTERED COLUMNSTORE

.

```
ALTER INDEX cci ON Products
REBUILD PARTITION = ALL
```

CLUSTERED COLUMNSTORE "" .

.

CLUSTERED COLUMNSTORE : https://riptutorial.com/ko/sql-server/topic/5774/clustered-columnstore

# 6: CREATE VIEW

## Examples

### CREATE VIEW

```
CREATE VIEW view_EmployeeInfo
AS
    SELECT EmployeeID,
        FirstName,
        LastName,
        HireDate
    FROM Employee
GO
```

.

```
SELECT FirstName
FROM view_EmployeeInfo
```

. .

```
CREATE VIEW view_EmployeeReport
AS
    SELECT EmployeeID,
        FirstName,
        LastName,
        Coalesce(FirstName,'') + ' ' + Coalesce(LastName,'') as FullName,
        HireDate
    FROM Employee
GO
```

SELECT . Employee FirstName LastName .

```
CREATE VIEW view_EmployeeInfo
WITH ENCRYPTION
AS
SELECT EmployeeID, FirstName, LastName, HireDate
FROM Employee
GO
```

### INNER JOIN

```
CREATE VIEW view_PersonEmployee
AS
    SELECT P.LastName,
        P.FirstName,
        E.JobTitle
    FROM Employee AS E
    INNER JOIN Person AS P
        ON P.BusinessEntityID = E.BusinessEntityID
```

```
GO
```

, . Person FirstName LastName Employee JobTitle .

.

```
SELECT *
FROM view_PersonEmployee
WHERE JobTitle LIKE '%Manager%'
```

## CREATE INDEXED VIEW

WITH SCHEMABINDING .

```
CREATE VIEW view_EmployeeInfo
WITH SCHEMABINDING
AS
    SELECT EmployeeID,
        FirstName,
        LastName,
        HireDate
    FROM [dbo].Employee
GO
```

.

```
CREATE UNIQUE CLUSTERED INDEX IX_view_EmployeeInfo
ON view_EmployeeInfo
(
     EmployeeID ASC
)
```

.

- .
- .
- , , .
- . WITH SCHEMABINDING .
- COUNT, MIN, MAX, TOP, .

MSDN .

VIEW GROUP BY . . VIEW . SQL . VIEW . .

https://www.simple-talk.com/sql/t-sql-programming/sql-view-beyond-the-basics/

```
CREATE VIEW BigSales (state_code, sales_amt_total)
AS SELECT state_code, MAX(sales_amt)
```

```
     FROM Sales
    GROUP BY state_code;
```

UNION UNION ALL VIEW . UNION . UNION UNION ALL . UNION [ALL] VIEW a . , UNION .

https://www.simple-talk.com/sql/t-sql-programming/sql-view-beyond-the-basics/

```
CREATE VIEW DepTally2 (emp_nbr, dependent_cnt)
AS (SELECT emp_nbr, COUNT(*)
      FROM Dependents
     GROUP BY emp_nbr)
   UNION
   (SELECT emp_nbr, 0
      FROM Personnel AS P2
     WHERE NOT EXISTS
          (SELECT *
             FROM Dependents AS D2
            WHERE D2.emp_nbr = P2.emp_nbr));
```

CREATE VIEW : https://riptutorial.com/ko/sql-server/topic/3815/create-view

# 7: DBCC

## Examples

### DBCC

DBCC .

:

```
DBCC DROPCLEANBUFFERS
```

columnstore columnstore .

```
DBCC FREEPROCCACHE
-- or
DBCC FREEPROCCACHE (0x060006001ECA270EC0215D05000000000000000000000000);
```

SQL . . SQL .

```
DBCC FREESYSTEMCACHE ('ALL', myresourcepool);
-- or
DBCC FREESYSTEMCACHE;
```

. ( **myresourcepool)** o = .

```
DBCC FLUSHAUTHCACHE
```

.

```
DBCC SHRINKDATABASE (MyDB [, 10]);
```

MyDB 10 % . . ID .

```
DBCC SHRINKFILE (DataFile1, 7);
```

DataFile1 . 7MB ( ).

```
DBCC CLEANTABLE (AdventureWorks2012,'Production.Document', 0)
```

.

### DBCC

DBCC .

```
ALTER TABLE Table1 WITH NOCHECK ADD CONSTRAINT chkTab1 CHECK (Col1 > 100);
GO
DBCC CHECKCONSTRAINTS(Table1);
--OR
DBCC CHECKCONSTRAINTS ('Table1.chkTable1');
```

nocheck . DBCC .

DBCC , .

```
DBCC CHECKTABLE tablename1 | tableid
DBCC CHECKDB databasename1 | dbid
DBCC CHECKFILEGROUP filegroup_name | filegroup_id | 0
DBCC CHECKCATALOG databasename1 | database_id1 | 0
```

## DBCC

DBCC .

```
DBCC PROCCACHE
```

.

```
DBCC OUTPUTBUFFER ( session_id [ , request_id ])
```

session_id ( request_id) 16 ASCII .

```
DBCC INPUTBUFFER ( session_id [ , request_id ])
```

Microsoft SQL Server .

```
DBCC SHOW_STATISTICS ( table_or_indexed_view_name , column_statistic_or_index_name)
```

## DBCC

SQL Server SQL Server / . DBCC .

3205 3206 .

```
DBCC TRACEON (3205, -1);
DBCC TRACEON (3206);
```

3205 , 3206 .

```
DBCC TRACEON (3205, -1);
DBCC TRACEON (3206);
```

2528 3205 .

```
DBCC TRACESTATUS (2528, 3205);
```

## DBCC

DBCC  SQL Server  . DBCC  DBCC HELP (...) .

DBCC .

```
DBCC HELP ('?');
```

DBCC CHECKDB .

```
DBCC HELP ('CHECKDB');
```

DBCC : https://riptutorial.com/ko/sql-server/topic/7316/dbcc

# 8: DBMAIL

- sp_send_dbmail [[@ =] ' '] [, [@ =] ' [; ... n] '] [, [@copy_recipients =]'copy_recipient [; ... n] '] [, [@blind_copy_recipients =]'blind_copy_recipient [; ', [@ =]' '] [, [@ =]' ' [, [@ body_format =] 'body_format'] [, [@importance =] ''] [, @ =] ''] [, [@file_attachments =] ' [; ...] [, [@ query_attachment_filename =] query_attachment_filename] [, [@ query_attachment_filename]], [@ query_attachment_filename =], [@ query_attachment_filename] @, [@ query_result_header =] query_result_header], [@query_result_width =] query_result_width] [, [@ query_result_separator]], [query_result_separator]], [@exclude_query_output =] exclude_query_output] [, [@append_query_error =] append_query_error] [, [@query_no_truncate = ] [query_no_truncate] [, [@ query_result_no_padding =] @query_result_no_padding] [, [@mailitem_id =] mailitem_id] [OUTPUT]

## Examples

recipient@someaddress.com .

```
EXEC msdb.dbo.sp_send_dbmail
    @profile_name = 'The Profile Name',
    @recipients = 'recipient@someaddress.com',
    @body = 'This is a simple email sent from SQL Server.',
    @subject = 'Simple email'
```

`SELECT * FROM Users  recipient@someaddress.com .`

```
EXEC msdb.dbo.sp_send_dbmail
    @profile_name = 'The Profile Name',
    @recipients = 'recipient@someaddress.com',
    @query = 'SELECT * FROM Users',
    @subject = 'List of users',
    @attach_query_result_as_file = 1;
```

### HTML

HTML `sp_send_dbmail` .

SQL Server 2012

```
DECLARE @html VARCHAR(MAX);
SET @html = CONCAT
(
    '<html><body>',
    '<h1>Some Header Text</h1>',
    '<p>Some paragraph text</p>',
    '</body></html>'
)
```

SQL Server 2012

```
DECLARE @html VARCHAR(MAX);
SET @html =
    '<html><body>' +
    '<h1>Some Header Text</h1>' +
    '<p>Some paragraph text</p>' +
    '</body></html>';
```

`@body argument` `@html` . HTML `@body` .

```
EXEC msdb.dbo.sp_send_dbmail
    @recipients='recipient@someaddress.com',
    @subject = 'Some HTML content',
    @body = @html,
    @body_format = 'HTML';
```

DBMAIL : https://riptutorial.com/ko/sql-server/topic/4908/dbmail

# 9: FOR XML PATH

FOR XML .

- FOR XML RAW - <row> <row> .
- FOR XML AUTO -  .
- FOR XML EXPLICIT - FOR XML EXPLICIT  FOR XML PATH  .

## Examples

### Hello World XML

```
SELECT 'Hello World' FOR XML PATH('example')
```

```
<example>Hello World</example>
```

SQL Server 2008

```
WITH XMLNAMESPACES (
    DEFAULT 'http://www.w3.org/2000/svg',
    'http://www.w3.org/1999/xlink' AS xlink
)
SELECT
    'example.jpg' AS 'image/@xlink:href',
    '50px' AS 'image/@width',
    '50px' AS 'image/@height'
FOR XML PATH('svg')
```

```
<svg xmlns:xlink="http://www.w3.org/1999/xlink" xmlns="http://www.w3.org/2000/svg">
    <image xlink:href="firefox.jpg" width="50px" height="50px"/>
</svg>
```

### XPath

```
SELECT
    'XPath example' AS 'head/title',
    'This example demonstrates ' AS 'body/p',
    'https://www.w3.org/TR/xpath/' AS 'body/p/a/@href',
    'XPath expressions' AS 'body/p/a'
FOR XML PATH('html')
```

```
<html>
    <head>
        <title>XPath example</title>
    </head>
    <body>
        <p>This example demonstrates <a href="https://www.w3.org/TR/xpath/">XPath
expressions</a></p>
    </body>
</html>
```

FOR XML PATH . NULL '' . : AS * .

```
DECLARE @tempTable TABLE (Ref INT, Des NVARCHAR(100), Qty INT)
INSERT INTO @tempTable VALUES (100001, 'Normal', 1), (100002, 'Foobar', 1), (100003, 'Hello
World', 2)

SELECT ROW_NUMBER() OVER (ORDER BY Ref) AS '@NUM',
     'REF' AS 'FLD/@NAME', REF AS 'FLD', '',
     'DES' AS 'FLD/@NAME', DES AS 'FLD', '',
     'QTY' AS 'FLD/@NAME', QTY AS 'FLD'
FROM @tempTable
FOR XML PATH('LIN'), ROOT('row')
```

```
<row>
  <LIN NUM="1">
    <FLD NAME="REF">100001</FLD>
    <FLD NAME="DES">Normal</FLD>
    <FLD NAME="QTY">1</FLD>
  </LIN>
  <LIN NUM="2">
    <FLD NAME="REF">100002</FLD>
    <FLD NAME="DES">Foobar</FLD>
    <FLD NAME="QTY">1</FLD>
  </LIN>
  <LIN NUM="3">
    <FLD NAME="REF">100003</FLD>
    <FLD NAME="DES">Hello World</FLD>
    <FLD NAME="QTY">2</FLD>
  </LIN>
</row>
```

() SQL Server . "" .

SELECT SQL Server .

'FLD / @ NAME' FOR XML PATH XML .

ROOT('row') row XML .

## FOR XML PATH

FOR XML PATH . CSV .

```
DECLARE @DataSource TABLE
(
    [rowID] TINYINT
   ,[FirstName] NVARCHAR(32)
);

INSERT INTO @DataSource ([rowID], [FirstName])
VALUES (1, 'Alex')
      ,(2, 'Peter')
      ,(3, 'Alexsandyr')
      ,(4, 'George');

SELECT STUFF
```

```
(
    (
        SELECT ',' + [FirstName]
        FROM @DataSource
        ORDER BY [rowID] DESC
        FOR XML PATH(''), TYPE
    ).value('.', 'NVARCHAR(MAX)')
    ,1
    ,1
    ,''
);
```

:

- ORDER BY .
- STUFF .

```
SELECT STUFF
(
    (
        SELECT '---' + [FirstName]
        FROM @DataSource
        ORDER BY [rowID] DESC
        FOR XML PATH(''), TYPE
    ).value('.', 'NVARCHAR(MAX)')
    ,1
    ,3 -- the "3" could also be represented as: LEN('---') for clarity
    ,''
);
```

- TYPE .value NVARCHAR(MAX) .

FOR XML PATH : https://riptutorial.com/ko/sql-server/topic/727/for-xml-path

# 10: GROUP BY

## Examples

| ID | ID | | |
|---|---|---|---|
| 1 | 2 | 5 | 100 |
| 1 | | 2 | 200 |
| 1 | 4 | 1 | 500 |
| 2 | 1 | 4 | 50 |
| | 5 | 6 | 700 |

.

```
SELECT customerId
FROM orders
GROUP BY customerId;
```

:

| ID |
|---|
| 1 |
| 2 |
| |

`count()` .

```
SELECT customerId,
       COUNT(productId) as numberOfProducts,
       sum(price) as totalPrice
FROM orders
GROUP BY customerId;
```

:

| ID | numberOfProducts | |
|---|---|---|
| 1 | | 800 |
| 2 | 1 | 50 |
| | 1 | 700 |

## GROUP

GROUP BY .

```
declare @temp table(age int, name varchar(15))

insert into @temp
select 18, 'matt' union all
select 21, 'matt' union all
select 21, 'matt' union all
select 18, 'luke' union all
select 18, 'luke' union all
select 21, 'luke' union all
select 18, 'luke' union all
select 21, 'luke'

SELECT Age, Name, count(1) count
FROM @temp
GROUP BY Age, Name
```

.

| | | |
|---|---|---|
| 18 | | |
| 21 | | 2 |
| 18 | | 1 |
| 21 | | 2 |

,

Group by join . . .

| | | |
|---|---|---|
| 1 | | 20 |
| 2 | | 21 |
| | | 18 |

.

| Subject_Id | |
|---|---|
| 1 | |
| 2 | |
| | |

( N : N ) "" . Student_subjects :

| Subject_Id | |
|---|---|
| 1 | 1 |
| 2 | 2 |
| 2 | 1 |
| | 2 |
| 1 | |
| 1 | 1 |

. GROUP BY . JOIN GROUP BY .

```
Select Students.FullName, COUNT(Subject Id) as SubjectNumber FROM Students_Subjects
LEFT JOIN Students
ON Students_Subjects.Student_id = Students.Id
GROUP BY Students.FullName
```

.

| FullName | SubjectNumber |
|---|---|
| | |
| | 2 |
| | 1 |

GROUP BY ( : students_Subjects ) . GROUPING :

```
SELECT Students.FullName, Subjects.Subject,
COUNT(Students_subjects.Subject_id) AS NumberOfOrders
FROM ((Students_Subjects
INNER JOIN Students
ON Students_Subjcets.Student_id=Students.Id)
INNER JOIN Subjects
ON Students_Subjects.Subject_id=Subjects.Subject_id)
GROUP BY Fullname,Subject
```

.

| FullName | | SubjectNumber |
|---|---|---|
| | | 2 |
| | | 1 |

| FullName | | SubjectNumber |
|---|---|---|
| | | 1 |
| | | 1 |
| | | 1 |

WHERE GROUP BY WHERE ( COUNT(*) ). HAVING .

.

```
DECLARE @orders TABLE(OrderID INT, Name NVARCHAR(100))

INSERT INTO @orders VALUES
( 1, 'Matt' ),
( 2, 'John' ),
( 3, 'Matt' ),
( 4, 'Luke' ),
( 5, 'John' ),
( 6, 'Luke' ),
( 7, 'John' ),
( 8, 'John' ),
( 9, 'Luke' ),
( 10, 'John' ),
( 11, 'Luke' )
```

```
SELECT Name, COUNT(*) AS 'Orders'
FROM @orders
GROUP BY Name
```

| | |
|---|---|
| | 2 |
| | 5 |
| | 4 |

HAVING .

```
SELECT Name, COUNT(*) AS 'Orders'
FROM @orders
GROUP BY Name
HAVING COUNT(*) > 2
```

| | |
|---|---|
| | 5 |
| | 4 |

GROUP BY HAVING SELECT .

```
SELECT Name, COUNT(DISTINCT OrderID)
```

HAVING

```
HAVING COUNT(DISTINCT OrderID) > 2
```

## ROLLUP CUBE GROUP BY

ROLLUP .

- CUBE .
- ROLLUP .

| | | |
|---|---|---|
| | | 124 |
| | | 223 |
| | | 101 |
| | | 210 |

```
SELECT CASE WHEN (GROUPING(Item) = 1) THEN 'ALL'
            ELSE ISNULL(Item, 'UNKNOWN')
       END AS Item,
       CASE WHEN (GROUPING(Color) = 1) THEN 'ALL'
            ELSE ISNULL(Color, 'UNKNOWN')
       END AS Color,
       SUM(Quantity) AS QtySum
FROM Inventory
GROUP BY Item, Color WITH ROLLUP

Item                 Color                QtySum
-------------------- -------------------- ---------------------------
Chair                Blue                 101.00
Chair                Red                  210.00
Chair                ALL                  311.00
Table                Blue                 124.00
Table                Red                  223.00
Table                ALL                  347.00
ALL                  ALL                  658.00
```

(7 )

ROLLUP CUBE CUBE .

```
ALL                  Blue                 225.00
ALL                  Red                  433.00
```

https://technet.microsoft.com/en-us/library/ms189305(v=sql.90).aspx

GROUP BY : https://riptutorial.com/ko/sql-server/topic/3231/group-by

# 11: JSON

## Examples

### JSON

JSON_VALUE JSON select .

```
select ProductID, Name, Color, Size, Price, JSON_VALUE(Data, '$.Type') as Type
from Product
where JSON_VALUE(Data, '$.Type') = 'part'
```

### JSON

JSON JSON JSON . JSON .

```
select JSON_VALUE(Data, '$.Type') as type,
        AVG( cast(JSON_VALUE(Data, '$.ManufacturingCost') as float) ) as cost
from Product
group by JSON_VALUE(Data, '$.Type')
having JSON_VALUE(Data, '$.Type') is not null
```

### JSON

JSON ISJSON .

```
select ProductID, Name, Color, Size, Price, JSON_VALUE(Data, '$.Type') as Type
from Product
where JSON_VALUE(Data, '$.Type') = 'part'
and ISJSON(Data) > 0
```

### JSON

JSON_MODIFY . UPDATE JSON .

```
update Product
set Data = JSON_MODIFY(Data, '$.Price', 24.99)
where ProductID = 17;
```

JSON_MODIFY Price ( ). NULL . JSON_MODIFY ( JSON ). JSON JSON_QUERY .

```
update Product
set Data = JSON_MODIFY(Data, '$.tags', JSON_QUERY('["promo","new"]'))
where ProductID = 17;
```

JSON_QUERY "JSON " . JSON_QUERY JSON ( ) JSON_MODIFY JSON .

## JSON

JSON_MODIFY JSON .

```
update Product
set Data = JSON_MODIFY(Data, 'append $.tags', "sales")
where ProductID = 17;
```

[ "sales"] . JSON_MODIFY ( JSON ). JSON JSON_QUERY .

```
update Product
set Data = JSON_MODIFY(Data, 'append $.tags', JSON_QUERY('{"type":"new"}'))
where ProductID = 17;
```

JSON_QUERY "JSON " . JSON_QUERY JSON ( ) JSON_MODIFY JSON .

## JSON JOIN

JSON " " JSON . JOIN CROSS APPLY . JSON .

```
select ProductID, Name, Size, Price, Quantity, PartName, Code
from Product
    CROSS APPLY OPENJSON(Data, '$.Parts') WITH (PartName varchar(20), Code varchar(5))
```

Product Part .

## JSON

Tags [ "promo", "sales"] .

```
select ProductID, Name, Color, Size, Price, Quantity
from Product
    CROSS APPLY OPENJSON(Data, '$.Tags')
where value = 'sales'
```

OPENJSON . .

JSON : https://riptutorial.com/ko/sql-server/topic/5028/json--

# 12: JSON

## Examples

### JSON

SELECT JSON . FOR JSON PATH .

```
SELECT top 3 object_id, name, type, principal_id FROM sys.objects
FOR JSON PATH
```

JSON JSON . JSON .

```
[
  {"object_id":3,"name":"sysrscols","type":"S "},
  {"object_id":5,"name":"sysrowsets","type":"S "},
  {"object_id":6,"name":"sysclones","type":"S "}
]
```

principal_id NULL ( ).

### JSON PATH

FOR JSON PATH JSON .

```
SELECT top 3 object_id as id, name as [data.name], type as [data.type]
FROM sys.objects
FOR JSON PATH
```

. (data.name data.type) . , ( ).

```
[
  {"id":3,"data":{"name":"sysrscols","type":"S "}},
  {"id":5,"data":{"name":"sysrowsets","type":"S "}},
  {"id":6,"data":{"name":"sysclones","type":"S "}}
]
```

### JSON ( )

WITHOUT_ARRAY_WRAPPER . / .

```
SELECT top 3 object_id, name, type, principal_id
FROM sys.objects
WHERE object_id = 3
FOR JSON PATH, WITHOUT_ARRAY_WRAPPER
```

.

```
{"object_id":3,"name":"sysrscols","type":"S "}
```

## INCLUDE_NULL_VALUES

FOR JSON NULL . "key": NULL INCLUDE_NULL_VALUES .

```
SELECT top 3 object_id, name, type, principal_id
FROM sys.objects
FOR JSON PATH, INCLUDE_NULL_VALUES
```

principal_id NULL .

```
[
  {"object_id":3,"name":"sysrscols","type":"S ","principal_id":null},
  {"object_id":5,"name":"sysrowsets","type":"S ","principal_id":null},
  {"object_id":6,"name":"sysclones","type":"S ","principal_id":null}
]
```

## ROOT

JSON .

```
SELECT top 3 object_id, name, type FROM sys.objects
FOR JSON PATH, ROOT('data')
```

JSON .

```
{
  "data":[
            {"object_id":3,"name":"sysrscols","type":"S "},
            {"object_id":5,"name":"sysrowsets","type":"S "},
            {"object_id":6,"name":"sysclones","type":"S "}
          ]
}
```

## JSON AUTO

JSON .

```
SELECT top 5 o.object_id, o.name, c.column_id, c.name
FROM sys.objects o
    JOIN sys.columns c ON o.object_id = c.object_id
FOR JSON AUTO
```

JSON .

```
[
  {
   "object_id":3,
   "name":"sysrscols",
   "c":[
```

```
        {"column_id":12,"name":"bitpos"},
        {"column_id":6,"name":"cid"}
       ]
  },
  {
    "object_id":5,
    "name":"sysrowsets",
    "c":[
         {"column_id":13,"name":"colguid"},
         {"column_id":3,"name":"hbcolid"},
         {"column_id":8,"name":"maxinrowlen"}
     ]
  }
]
```

## JSON

FOR JSON PATH  FOR JSON AUTO  JSON  FOR JSON  JSON  .

```
SELECT top 5 o.object_id, o.name,
        (SELECT column_id, c.name
            FROM sys.columns c WHERE o.object_id = c.object_id
            FOR JSON PATH) as columns,
        (SELECT parameter_id, name
            FROM sys.parameters p WHERE o.object_id = p.object_id
            FOR JSON PATH) as parameters
FROM sys.objects o
FOR JSON PATH
```

JSON  JSON .

JSON  : https://riptutorial.com/ko/sql-server/topic/4661/json-

# 13: JSON

## Examples

### JSON

JSON_MODIFY JSON .

```
declare @json nvarchar(4000) = N'{"Id":1,"Name":"Toy Car","Price":34.99}'
set @json = JSON_MODIFY(@json, '$.Price', 39.99)
print @json -- Output: {"Id":1,"Name":"Toy Car","Price":39.99}
```

JSON "Price": 39.99 . JSON_MODIFY key : value .

key : value NULL .

```
declare @json nvarchar(4000) = N'{"Id":1,"Name":"Toy Car","Price":34.99}'
set @json = JSON_MODIFY(@json, '$.Price', NULL)
print @json -- Output: {"Id":1,"Name":"Toy Car"}
```

JSON_MODIFY .

### JSON

JSON_MODIFY " .

```
declare @json nvarchar(4000) = N'{"Id":1,"Name":"Toy Car","Tags":["toy","game"]}'
set @json = JSON_MODIFY(@json, 'append $.Tags', 'sales')
print @json -- Output: {"Id":1,"Name":"Toy Car","Tags":["toy","game","sales"]}
```

JSON_MODIFY (append) .

```
declare @json nvarchar(4000) = N'{"Id":1,"Name":"Toy Car","Price":34.99}'
set @json = JSON_MODIFY(@json, 'append $.Tags', 'sales')
print @json -- Output {"Id":1,"Name":"Toy Car","Tags":["sales"]}
```

### JSON JSON

JSON_MODIFY JSON JSON .

```
declare @json nvarchar(4000) = N'{"Id":1,"Name":"Toy Car"}'
set @json = JSON_MODIFY(@json, '$.Price',
                        JSON_QUERY('{"Min":34.99,"Recommended":45.49}'))
print @json
-- Output: {"Id":1,"Name":"Toy Car","Price":{"Min":34.99,"Recommended":45.49}}
```

JSON_QUERY JSON "" . "" JSON_MODIFY . JSON_QUERY .

```
{"Id":1,"Name":"Toy Car","Price":'{\"Min\":34.99,\"Recommended\":45.49}'}
```

JSON_MODIFY NULL .

## FOR JSON JSON

FOR JSON SELECT JSON JSON .

```
declare @json nvarchar(4000) = N'{"Id":17,"Name":"WWI"}'
set @json = JSON_MODIFY(@json, '$.tables',
                        (select name from sys.tables FOR JSON PATH) )
print @json

(1 row(s) affected)
{"Id":1,"Name":"master","tables":[{"name":"Colors"},{"name":"Colors_Archive"},{"name":"OrderLines"},{"n
```

JSON_MODIFY FOR JSON JSON JSON .

**WITHOUT_ARRAY_WRAPPER** SELECT FOR JSON . JSON . JSON .

## FOR JSON JSON

FOR JSON WITHOUT_ARRAY_WRAPPER SELECT JSON JSON .

```
declare @json nvarchar(4000) = N'{"Id":17,"Name":"WWI"}'
set @json = JSON_MODIFY(@json, '$.table',
                        JSON_QUERY(
                         (select name, create_date, schema_id
                           from sys.tables
                           where name = 'Colors'
                           FOR JSON PATH, WITHOUT_ARRAY_WRAPPER)))
print @json

(1 row(s) affected)
{"Id":17,"Name":"WWI","table":{"name":"Colors","create_date":"2016-06-
02T10:04:03.280","schema_id":13}}
```

SELECT ( ) WITHOUT_ARRAY_WRAPPER JSON JSON . JSON_MODIFY JSON_QUERY .

JSON **JSON_QUERY FOROUT JSON, WITHOUT_ARRAY_WRAPPER** .

JSON : https://riptutorial.com/ko/sql-server/topic/6883/json--

# 14: JSON SQL

## Examples

### JSON

JSON NVARCHAR . NoSQL .

```
CREATE TABLE ProductCollection (
  Id int identity primary key,
  Data nvarchar(max)
)
```

JSON `nvarchar(max)` . `nvarchar(4000)` `varchar(8000)` 8KB.

### JSON ISJSON .

JSON . JSON CHECK JSON :

```
CREATE TABLE ProductCollection (
  Id int identity primary key,
  Data nvarchar(max)
       CONSTRAINT [Data should be formatted as JSON]
       CHECK (ISJSON(Data) > 0)
)
```

ALTER TABLE CHECK .

```
ALTER TABLE ProductCollection
    ADD CONSTRAINT [Data should be formatted as JSON]
        CHECK (ISJSON(Data) > 0)
```

### JSON

JSON .

```
CREATE TABLE ProductCollection (
  Id int identity primary key,
  Data nvarchar(max),
  Price AS JSON_VALUE(Data, '$.Price'),
  Color JSON_VALUE(Data, '$.Color') PERSISTED
)
```

PERSISTED JSON . JSON . JSON .

### JSON

JSON .

```
SELECT * FROM ProductCollection
WHERE JSON_VALUE(Data, '$.Color') = 'Black'
```

filter sort ( JSON_VALUE (Data, '$ .Color')) JSON .

```
ALTER TABLE ProductCollection
ADD vColor as JSON_VALUE(Data, '$.Color')

CREATE INDEX idx_JsonColor
ON ProductCollection(vColor)
```

.

## JSON

JSON .

```
CREATE TABLE ProductCollection (
  Id int identity primary key nonclustered,
  Data nvarchar(max)
) WITH (MEMORY_OPTIMIZED=ON)
```

JSON :

- JSON .
- JSON .

JSON SQL : https://riptutorial.com/ko/sql-server/topic/5029/json-sql--

# 15: Microsoft SQL Server  Studio

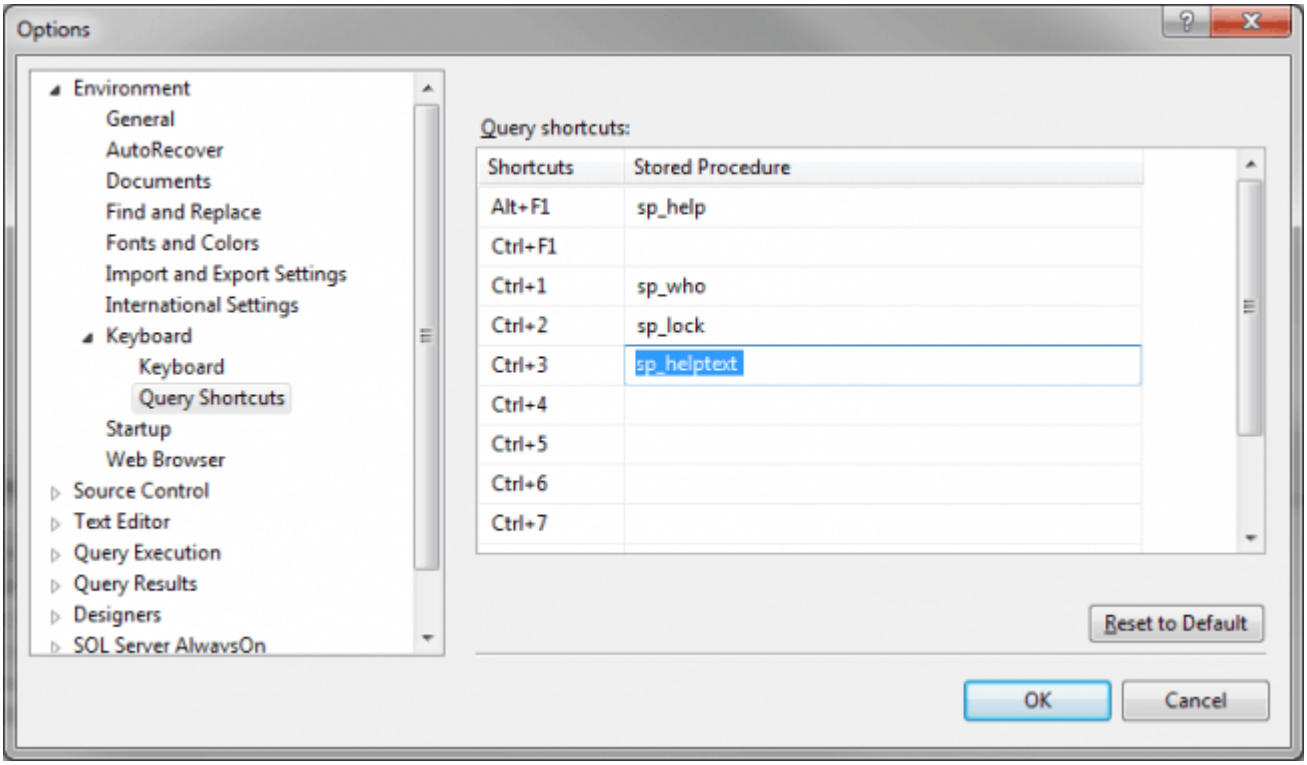
## Examples

1. ( `Ctrl` + `N` )
2. ( `Ctrl` + `Tab` )
3. /  ( `Ctrl` + `R` )
4. ( `Ctrl` + `E` )
5. ( `Ctrl` + `Shift` + `U` , `Ctrl` + `Shift` + `L` ).
6. Intellisense  ( `Ctrl` + `Space` , `Tab` )
7. ( `Ctrl` + `G` )
8. SQL Server Managament Studio  ( `Ctrl` + `F4` )

1. SQL Server Management Studio  ( `ALT` )
2. ( `Alt` + `HYPHEN` )
3. ( `Shift` + `F` )
4. ( `Ctrl` + `N` ).
5. ( `Ctrl` + `Shift` + `0` )
6. ( `Ctrl` + `Shift` + `A` ).
7. ( `Ctrl` + `Shift` + `A` ).
8. ( `Ctrl` + `Shift` + `Q` )
9. ( `ESC` ).

-> . -> ->  .

SSMS  .  .



. Ctrl + 3 .

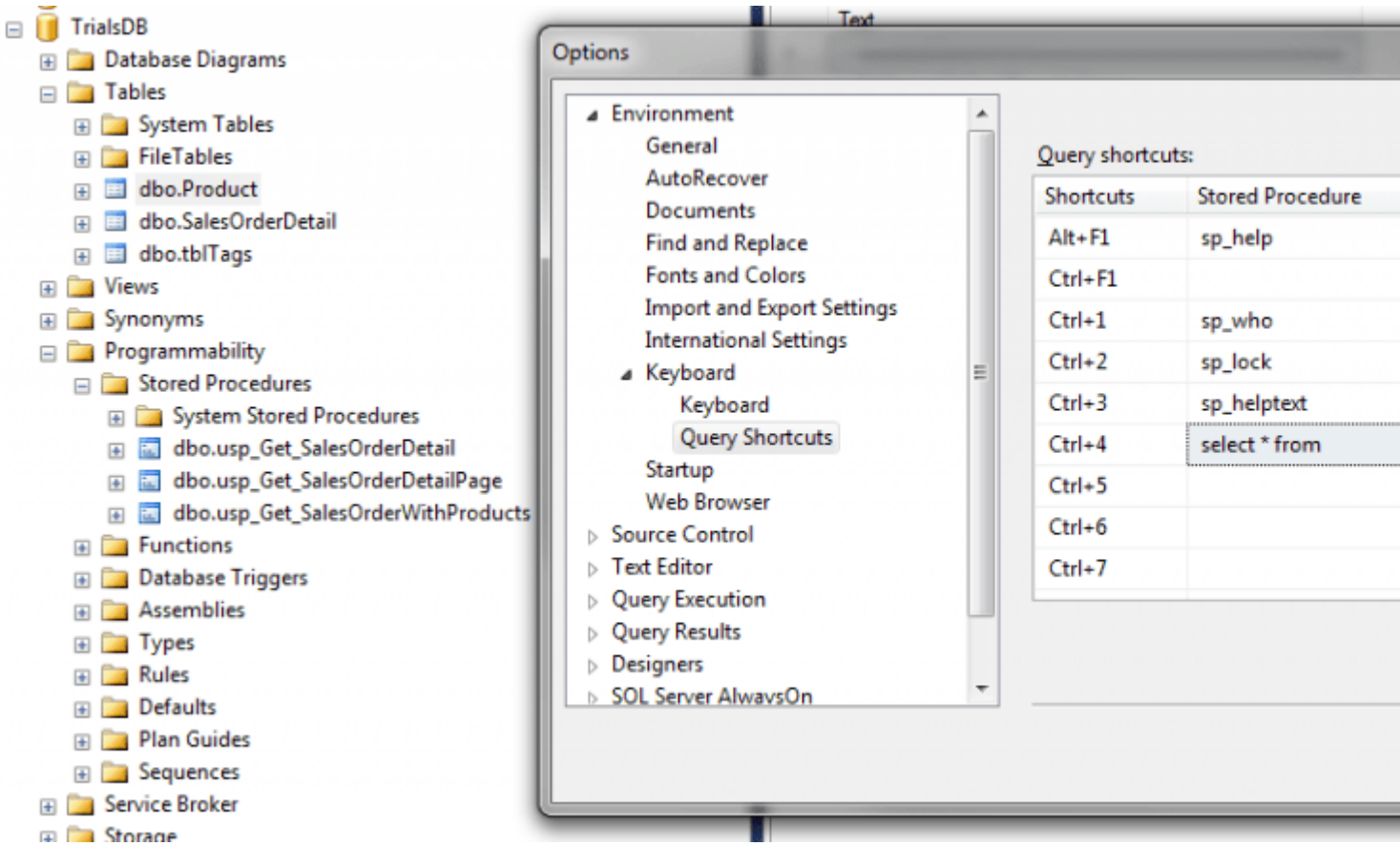
```
usp_Get_SalesOrderDetail
```

100 %

Results | Messages

| | Text |
|---|---|
| 1 | -- ============================================ |
| 2 | -- Author: <Author,Sibeesh Venu> |
| 3 | -- Create date: <Create Date, 18-Feb-2016> |
| 4 | -- Description: <Description,To fetch SalesOrderDetail> |
| 5 | -- ============================================ |
| 6 | CREATE PROCEDURE [dbo].[usp_Get_SalesOrderDetail] |
| 7 | AS |
| 8 | BEGIN |
| 9 | -- SET NOCOUNT ON added to prevent extra result sets ... |
| 10 | -- interfering with SELECT statements. |
| 11 | SET NOCOUNT ON; |
| 12 | |
| 13 | -- Select statements for procedure here |
| 14 | SELECT top(100) SalesOrderID,SalesOrderDetailID,Carri... |
| 15 | END |

CTRL + 5 ( ) . .



CTRL + 4 ( )  .

Microsoft SQL Server  Studio  : https://riptutorial.com/ko/sql-server/topic/7749/microsoft-sql-server--studio---

# 16: MS SQL Server  DDL

## Examples

**DDL** (= **"D** ATA **D** efinition  anguage"),  .

---

SQL `C:\Program Files\Microsoft SQL Server\MSSQL11.INSTSQL2012\MSSQL\DATA\  Northwind .`

```
USE [master]
GO

CREATE DATABASE [Northwind]
 CONTAINMENT = NONE
 ON  PRIMARY
 (
  NAME = N'Northwind',
  FILENAME = N'C:\Program Files\Microsoft SQL
Server\MSSQL11.INSTSQL2012\MSSQL\DATA\Northwind.mdf' , SIZE = 5120KB , MAXSIZE = UNLIMITED,
FILEGROWTH = 1024KB
 )
 LOG ON
 (
  NAME = N'Northwind_log',
  FILENAME = N'C:\Program Files\Microsoft SQL
Server\MSSQL11.INSTSQL2012\MSSQL\DATA\Northwind_log.ldf' , SIZE = 1536KB , MAXSIZE = 2048GB ,
FILEGROWTH = 10%
 )
GO

ALTER DATABASE [Northwind] SET COMPATIBILITY_LEVEL = 110
GO
```

**:** T-SQL `*.mdf` `*.ldf` . .

---

SQL `dbo` `Categories` ( `Use <DatabaseName>` ).

```
CREATE TABLE dbo.Categories(
    CategoryID int IDENTITY NOT NULL,
    CategoryName nvarchar(15) NOT NULL,
    Description ntext NULL,
    Picture image NULL,
      CONSTRAINT PK_Categories PRIMARY KEY CLUSTERED
      (
         CategoryID ASC
      )
      WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF,
            ALLOW_ROW_LOCKS = ON, ALLOW_PAGE_LOCKS = ON) ON PRIMARY
) ON PRIMARY TEXTIMAGE_ON PRIMARY
```

---

SQL `dbo` `Summary_of_Sales_by_Year` ( `Use <DatabaseName>` ).

```
CREATE VIEW dbo.Summary_of_Sales_by_Year AS
  SELECT ord.ShippedDate, ord.OrderID, ordSub.Subtotal
  FROM Orders ord
  INNER JOIN [Order Subtotals] ordSub ON ord.OrderID = ordSub.OrderID
```

`Orders` `[Order Subtotals]` `ShippedDate` , `OrderID` `Subtotal` . `[Order Subtotals]` Northwind .

---

SQL `dbo` `CustOrdersDetail` ( `Use <DatabaseName>` ).

```
CREATE PROCEDURE dbo.MyCustOrdersDetail @OrderID int, @MinQuantity int=0
AS BEGIN
  SELECT ProductName,
    UnitPrice=ROUND(Od.UnitPrice, 2),
    Quantity,
    Discount=CONVERT(int, Discount * 100),
    ExtendedPrice=ROUND(CONVERT(money, Quantity * (1 - Discount) * Od.UnitPrice), 2)
  FROM Products P, [Order Details] Od
  WHERE Od.ProductID = P.ProductID and Od.OrderID = @OrderID
  and Od.Quantity>=@MinQuantity
END
```

.

```
exec dbo.MyCustOrdersDetail 10248
```

@ OrderId = 10248 ( > = 0 ) . .

```
exec dbo.MyCustOrdersDetail 10248, 10
```

10 ( ) .

MS SQL Server DDL : https://riptutorial.com/ko/sql-server/topic/5463/ms-sql-server--ddl-

# 17: NULLs

SQL Server `NULL` . , `NULL` . . `NULL` , `NULL` .

SQL Server `IS NULL` , `IS NOT NULL` , `ISNULL()` , `COALESCE()` null .

## Examples

### NULL

`NULL` .

.

```
id someVal
 ----
 0 NULL
 1 1
 2 2
```

:

```
 SELECT id
 FROM table
 WHERE someVal = 1
```

id `1` .

```
 SELECT id
 FROM table
 WHERE someVal <> 1
```

id `2` .

```
 SELECT id
 FROM table
 WHERE someVal IS NULL
```

id `0` .

```
 SELECT id
 FROM table
 WHERE someVal IS NOT NULL
```

ids `1` `2` .

NULL `=` , `<>` "" .

```
SELECT id
FROM table
WHERE ISNULL(someVal, -1) <> 1
```

```
SELECT id
FROM table
WHERE someVal IS NULL OR someVal <> 1
```

0 2 .

ANSI Null .

## ANSI NULLS

MSDN

SQL Server ANSI_NULLS ON OFF . .

ANSI NULLS off null = / <> .

:

```
id someVal
 ----
 0 NULL
 1 1
 2 2
```

ANSI NULLS on .

```
SELECT id
FROM table
WHERE someVal = NULL
```

. ANSI NULLS .

```
set ansi_nulls off

SELECT id
FROM table
WHERE someVal = NULL
```

id 0 .

## ISNULL ()

IsNull() null .

:

1. . .
2.

. null . .

`IsNull()` check .

```
DECLARE @MyInt int -- All variables are null until they are set with values.

SELECT ISNULL(@MyInt, 3) -- Returns 3.
```

`COALESCE` .

## null null .

null null .
null `is null` .

```
DECLARE @Date date = '2016-08-03'
```

. (NULL) `6` .

```
SELECT CASE WHEN @Date = NULL THEN 1
            WHEN @Date <> NULL THEN 2
            WHEN @Date > NULL THEN 3
            WHEN @Date < NULL THEN 4
            WHEN @Date IS NULL THEN 5
            WHEN @Date IS NOT NULL THEN 6
```

@Date `null` `5` .

```
SET @Date = NULL -- Note that the '=' here is an assignment operator!

SELECT CASE WHEN @Date = NULL THEN 1
            WHEN @Date <> NULL THEN 2
            WHEN @Date > NULL THEN 3
            WHEN @Date < NULL THEN 4
            WHEN @Date IS NULL THEN 5
            WHEN @Date IS NOT NULL THEN 6
```

## COALESCE ()

`COALESCE ()` NULL NULL .

```
DECLARE @MyInt int -- variable is null until it is set with value.
DECLARE @MyInt2 int -- variable is null until it is set with value.
DECLARE @MyInt3 int -- variable is null until it is set with value.

SET @MyInt3  = 3

SELECT COALESCE (@MyInt, @MyInt2 ,@MyInt3 ,5) -- Returns 3 : value of @MyInt3.
```

ISNULL () COALESCE () ISNULL () ( ) NULL . `ISNULL` .

## NULL

NULLS .

```
create table #outertable (i int)
create table #innertable (i int)

insert into #outertable (i) values (1), (2),(3),(4), (5)
insert into #innertable (i) values (2), (3), (null)

select * from #outertable where i in (select i from #innertable)
--2
--3
--So far so good

select * from #outertable where i not in (select i from #innertable)
--Expectation here is to get 1,4,5 but it is not. It will get empty results because of the
NULL it executes as {select * from #outertable where i not in (null)}

--To fix this
select * from #outertable where i not in (select i from #innertable where i is not null)
--you will get expected results
--1
--4
--5
```

null .

NULLs : https://riptutorial.com/ko/sql-server/topic/5044/nulls

# 18: OPENJSON

## Examples

### : JSON

OPENJSON JSON JSON : .

```
declare @json NVARCHAR(4000) = N'{"Name":"Joe","age":27,"skills":["C#","SQL"]}';
SELECT * FROM OPENJSON(@json);
```

| | | |
|---|---|---|
| | | 1 |
| | 27 | 2 |
| | [ "C #", "SQL"] | 4 |

(0), (1), (2), (3), (4) (5) .

### JSON

OPENJSON JSON JSON .

```
declare @json nvarchar(4000) = N'[
  {"Number":"SO43659","Date":"2011-05-31T00:00:00","Customer":
"MSFT","Price":59.99,"Quantity":1},
  {"Number":"SO43661","Date":"2011-06-
01T00:00:00","Customer":"Nokia","Price":24.99,"Quantity":3}
]'

SELECT    *
FROM OPENJSON (@json)
    WITH (
          Number   varchar(200),
          Date     datetime,
          Customer varchar(200),
          Quantity int
  )
```

WITH OPENJSON . JSON . JSON WITH ( : Price) . .

| | | | |
|---|---|---|---|
| SO43659 | 2011-05-31T00 : 00 : 00 | MSFT | 1 |
| SO43661 | 2011-06-01T00 : 00 : 00 | | |

### JSON

OPENJSON JSON JSON . WITH .

```
declare @json nvarchar(4000) = N'[
  {"data":{"num":"SO43659","date":"2011-05-
31T00:00:00"},"info":{"customer":"MSFT","Price":59.99,"qty":1}},
  {"data":{"number":"SO43661","date":"2011-06-
01T00:00:00"},"info":{"customer":"Nokia","Price":24.99,"qty":3}}
]'

SELECT    *
FROM OPENJSON (@json)
    WITH (
          Number   varchar(200) '$.data.num',
          Date     datetime '$.data.date',
          Customer varchar(200) '$.info.customer',
          Quantity int '$.info.qty',
  )
```

WITH OPENJSON . type JSON . JSON . .

| | | | |
|---|---|---|---|
| SO43659 | 2011-05-31T00 : 00 : 00 | MSFT | 1 |
| SO43661 | 2011-06-01T00 : 00 : 00 | | |

## JSON

OPENJSON JSON JSON . JSON nvarchar (max) AS JSON .

```
declare @json nvarchar(4000) = N'[
  {"Number":"SO43659","Date":"2011-05-
31T00:00:00","info":{"customer":"MSFT","Price":59.99,"qty":1}},
  {"Number":"SO43661","Date":"2011-06-
01T00:00:00","info":{"customer":"Nokia","Price":24.99,"qty":3}}
]'

SELECT    *
FROM OPENJSON (@json)
    WITH (
          Number   varchar(200),
          Date     datetime,
          Info nvarchar(max) '$.info' AS JSON
  )
```

"" . .

| | | |
|---|---|---|
| SO43659 | 2011-05-31T00 : 00 : 00 | { "": "MSFT", "": 59.99, "": 1} |
| SO43661 | 2011-06-01T00 : 00 : 00 | { "": "Nokia", "Price": 24.99, "qty": 3} |

## JSON

JSON . OrderItems .

```
declare @json nvarchar(4000) = N'[
  {"Number":"SO43659","Date":"2011-05-31T00:00:00",
    "Items":[{"Price":11.99,"Quantity":1},{"Price":12.99,"Quantity":5}]},
  {"Number":"SO43661","Date":"2011-06-01T00:00:00",

"Items":[{"Price":21.99,"Quantity":3},{"Price":22.99,"Quantity":2},{"Price":23.99,"Quantity":2}]}

]'
```

OPENJSON . AS JSON . Items OPENJSON JSON . JOIN "".

```
SELECT    *
FROM
    OPENJSON (@json)
    WITH (  Number varchar(200), Date datetime,
            Items nvarchar(max) AS JSON )
        CROSS APPLY
            OPENJSON (Items)
            WITH ( Price float, Quantity int)
```

:

| | | | | |
|---|---|---|---|---|
| SO43659 | 2011-05-31 00 : 00 : 00.000 | [{ "": 11.99, "": 1}, { "": 12.99, "": 5}] | 11.99 | 1 |
| SO43659 | 2011-05-31 00 : 00 : 00.000 | [{ "": 11.99, "": 1}, { "": 12.99, "": 5}] | 12.99 | 5 |
| SO43661 | 2011-06-01 00 : 00 : 00.000 | [{ "Price : 21.99,"Quantity ": 3}, {"Price ": 22.99,"Quantity ": 2}, {"Price ": 23.99,"Quantity ": 2}] | 21.99 | |
| SO43661 | 2011-06-01 00 : 00 : 00.000 | [{ "Price : 21.99,"Quantity ": 3}, {"Price ": 22.99,"Quantity ": 2}, {"Price ": 23.99,"Quantity ": 2}] | 22.99 | 2 |
| SO43661 | 2011-06-01 00 : 00 : 00.000 | [{ "Price : 21.99,"Quantity ": 3}, {"Price ": 22.99,"Quantity ": 2}, {"Price ": 23.99,"Quantity ": 2}] | 23.99 | 2 |

OPENJSON : https://riptutorial.com/ko/sql-server/topic/5030/openjson

# 19: OVER

| |
|---|
| PARTITION BY " . |

OVER . . OVER .

- 
- 

, , N .

OVER GROUP BY . OVER / GROUP BY .

# **1 :** SQL Server 2008 (R2) ORDER BY ( ) .

## Examples

### OVER

Cars costumer , , (COUNT) .

Id CustomerId MechanicId

```
SELECT CustomerId,
       SUM(TotalCost) OVER(PARTITION BY CustomerId) AS Total,
       AVG(TotalCost) OVER(PARTITION BY CustomerId) AS Avg,
       COUNT(TotalCost) OVER(PARTITION BY CustomerId) AS Count,
       MIN(TotalCost) OVER(PARTITION BY CustomerId) AS Min,
       MAX(TotalCost) OVER(PARTITION BY CustomerId) AS Max
  FROM CarsTable
 WHERE Status = 'READY'
```

OVER . .

| ID | | | | | |
|---|---|---|---|---|---|
| 1 | 430 | 215 | 2 | 200 | 230 |
| 1 | 430 | 215 | 2 | 200 | 230 |

.

Eg GROUP BY .

```
SELECT CustomerId,
       SUM(TotalCost) AS Total,
       AVG(TotalCost) AS Avg,
```

```
       COUNT(TotalCost) AS Count,
       MIN(TotalCost) AS Min,
       MAX(TotalCost)  AS Max
  FROM CarsTable
 WHERE Status = 'READY'
GROUP BY CustomerId
```

. .

```
SELECT item_id, sale_Date
       SUM(quantity * price) OVER(PARTITION BY item_id ORDER BY sale_Date ROWS BETWEEN
UNBOUNDED PRECEDING) AS SalesTotal
  FROM SalesTable
```

. Id .

```
SELECT MostRecentBook.Name, MostRecentBook.Title
FROM ( SELECT Authors.Name,
              Books.Title,
              RANK() OVER (PARTITION BY Authors.Id ORDER BY Books.Id DESC) AS NewestRank
       FROM Authors
       JOIN Books ON Books.AuthorId = Authors.Id
     ) MostRecentBook
WHERE MostRecentBook.NewestRank = 1
```

RANK . .

- `RANK()` : , .
- `DENSE_RANK()` : , .
- `ROW_NUMBER()` : " ''.

, CreationDate .

```
SELECT Authors.Name,
       Books.Title,
       Books.CreationDate,
       RANK() OVER (PARTITION BY Authors.Id ORDER BY Books.CreationDate DESC) AS RANK,
       DENSE_RANK() OVER (PARTITION BY Authors.Id ORDER BY Books.CreationDate DESC) AS
DENSE_RANK,
       ROW_NUMBER() OVER (PARTITION BY Authors.Id ORDER BY Books.CreationDate DESC) AS
ROW_NUMBER,
FROM Authors
JOIN Books ON Books.AuthorId = Authors.Id
```

:

| | | | | DENSE_RANK | ROW_NUMBER |
|---|---|---|---|---|---|
| 1 | 1 | 22/07/2016 | 1 | 1 | 1 |
| 1 | 2 | 22/07/2016 | 1 | 1 | 2 |
| 1 | 3 | 21/07/2016 | | 2 | |

| | | | | DENSE_RANK | ROW_NUMBER |
|---|---|---|---|---|---|
| 1 | 4 | 21/07/2016 | | 2 | 4 |
| 1 | 5 | 21/07/2016 | | 2 | 5 |
| 1 | 6 | 2011 4 7 | 6 | | 6 |
| 2 | 7 | 2011 4 7 | 1 | 1 | 1 |

## NTILE

4 .

```
-- Setup data:
declare @values table(Id int identity(1,1) primary key, [Value] float, ExamId int)
insert into @values ([Value], ExamId) values
(65, 1), (40, 1), (99, 1), (100, 1), (90, 1), -- Exam 1 Scores
(91, 2), (88, 2), (83, 2), (91, 2), (78, 2), (67, 2), (77, 2) -- Exam 2 Scores

-- Separate into four buckets per exam:
select ExamId,
       ntile(4) over (partition by ExamId order by [Value] desc) as Quartile,
       Value, Id
from @values
order by ExamId, Quartile
```

| | ExamId | Quartile | Value | Id |
|---|---|---|---|---|
| 1 | 1 | 1 | 100 | 4 |
| 2 | 1 | 1 | 99 | 3 |
| 3 | 1 | 2 | 90 | 5 |
| 4 | 1 | 3 | 65 | 1 |
| 5 | 1 | 4 | 40 | 2 |
| 6 | 2 | 1 | 91 | 9 |
| 7 | 2 | 1 | 91 | 6 |
| 8 | 2 | 2 | 88 | 7 |
| 9 | 2 | 2 | 83 | 8 |
| 10 | 2 | 3 | 78 | 10 |
| 11 | 2 | 3 | 77 | 12 |
| 12 | 2 | 4 | 67 | 11 |

ntile . ntile(100) ntile(100) .

OVER : https://riptutorial.com/ko/sql-server/topic/353/over-

# 20: SCOPE_IDENTITY ()

- SELECT SCOPE_IDENTITY ();
- SELECT SCOPE_IDENTITY () AS [SCOPE_IDENTITY];
- SCOPE_IDENTITY ()

## Examples

SCOPE_IDENTITY ()  ID  ID . ,  , ,  .  ,  .

INSERT INTO ([column1], [column2]) VALUES (8,9);

SELECT SCOPE_IDENTITY () AS [SCOPE_IDENTITY];

SCOPE_IDENTITY ()  : https://riptutorial.com/ko/sql-server/topic/5326/scope-identity---

# 21: SELECT

SQL `SELECT` . `SELECT` `WHERE` , `GROUP BY` `ORDER BY` .

## Examples

### SELECT

( ) .

```
SELECT *
FROM sys.objects
```

.

```
SELECT object_id, name, type, create_date
FROM sys.objects
```

### WHERE

WHERE .

```
SELECT *
FROM sys.objects
WHERE type = 'IT'
```

### ORDER BY

ORDER BY .

```
SELECT *
FROM sys.objects
ORDER BY create_date
```

### GROUP BY

GROUP BY .

```
SELECT type, count(*) as c
FROM sys.objects
GROUP BY type
```

( ) .

| |
|---|
| SQ |

| | |
|---|---|
| | 72 |
| | 16 |
| PK | 1 |
| | 5 |

## HAVING

HAVING .

```
SELECT type, count(*) as c
FROM sys.objects
GROUP BY type
HAVING count(*) < 10
```

| | |
|---|---|
| SQ | |
| PK | 1 |
| | 5 |

## N

TOP N .

```
SELECT TOP 10 *
FROM sys.objects
```

## OFFSET FETCH

OFFSET FETCH TOP . N1 N2 .

```
SELECT *
FROM sys.objects
ORDER BY object_id
OFFSET 50 ROWS FETCH NEXT 10 ROWS ONLY
```

OFFSET 50 .

```
SELECT *
FROM sys.objects
ORDER BY object_id
OFFSET 50 ROWS
```

## FROM ( )

SELECT FROM .

```
declare @var int = 17;

SELECT @var as c1, @var + 2 as c2, 'third' as c3
```

/ .

SELECT : https://riptutorial.com/ko/sql-server/topic/4662/select-

# 22: SQL Server Management Studio (SSMS)

SQL Server Management Studio (SSMS) SQL Server SQL .

SSMS Microsoft .

SSMS .

## Examples

### IntelliSense

IntelliSense . IntelliSense .

```
Ctrl + Shift + R Edit | IntelliSense | Refresh Local Cache Edit | IntelliSense | Refresh Local Cache Edit | IntelliSense | Refresh Local Cache.
```

IntelliSense .

SQL Server Management Studio (SSMS) : https://riptutorial.com/ko/sql-server/topic/10642/sql-server-management-studio--ssms-

# 23: SQL Server

## Examples

### SQL Server 2016

**SQL Server 2016** Split . `STRING_SPLIT`

**:** .

:

(, nvarchar, varchar, nchar char) .

:

( : nvarchar (1), varchar (1), nchar (1) char (1)) .

**:** .

**:**

```
Select Value
From STRING_SPLIT('a|b|c','|')
```

```
String    : 'a|b|c'
separator : '|'
```

**:**

```
+-----+
|Value|
+-----+
|a    |
+-----+
|b    |
+-----+
|c    |
+-----+
```

**:**

```
SELECT value
FROM STRING_SPLIT('',',')
```

**:**

```
  +-----+
  |Value|
```

```
  +-----+
1 |     |
  +-----+
```

WHERE .

```
SELECT value
FROM STRING_SPLIT('',',')
WHERE LTRIM(RTRIM(value))<>''
```

## XML SQL Server 2008/2012/2014

STRING_SPLIT XML .

**:**

```
SELECT split.a.value('.', 'VARCHAR(100)') AS Value
FROM   (SELECT Cast ('<M>' + Replace('A|B|C', '|', '</M><M>')+ '</M>' AS XML) AS Data) AS A
       CROSS apply data.nodes ('/M') AS Split(a);
```

**:**

```
+-----+
|Value|
+-----+
|A    |
+-----+
|B    |
+-----+
|C    |
+-----+
```

## T-SQL XML

```
Declare @userList Table(UserKey VARCHAR(60))
Insert into @userList values ('bill'),('jcom'),('others')
--Declared a table variable and insert 3 records

Declare @text XML
Select  @text = (
        select UserKey  from @userList for XML Path('user'), root('group')
)
--Set the XML value from Table

Select @text

--View the variable value
XML:
\<group>\<user>\<UserKey>bill\</UserKey>\</user>\<user>\<UserKey>jcom\</UserKey>\</user>\<user>\<UserKe
```

SQL Server : https://riptutorial.com/ko/sql-server/topic/3713/sql-server---

# 24: SQL Server JSON

- **JSON_VALUE** (expression, path) - JSON .
- **JSON_QUERY** (expression [, path]) - JSON .
- **OPENJSON** (jsonExpression [, path]) - JSON JSON .
- **ISJSON** (expression) - JSON .
- **JSON_MODIFY** (expression, path, newValue) - JSON JSON .

| | |
|---|---|
| | JSON . |
| | JSON . . [append] [lax \| ] $. <json > |
| jsonExpression | JSON . |

OPENJSON 130 . 130 SQL Server OPENJSON . Azure SQL 120 . .

```
ALTER DATABASE <Database-Name-Here> SET COMPATIBILITY_LEVEL = 130
```

## Examples

### FOR JSON JSON

( )

| | | |
|---|---|---|
| 1 | | 23 |
| 2 | | 31 |

```
SELECT Id, Name, Age
FROM People
FOR JSON PATH
```

```
[
    {"Id":1,"Name":"John","Age":23},
    {"Id":2,"Name":"Jane","Age":31}
]
```

### JSON

**JSON_VALUE** **JSON_QUERY** JSON JSON / .

```
DECLARE @json NVARCHAR(100) = '{"id": 1, "user":{"name":"John"}, "skills":["C#","SQL"]}'
```

```
SELECT
    JSON_VALUE(@json, '$.id') AS Id,
    JSON_VALUE(@json, '$.user.name') AS Name,
    JSON_QUERY(@json, '$.user') AS UserObject,
    JSON_QUERY(@json, '$.skills') AS Skills,
    JSON_VALUE(@json, '$.skills[0]') AS Skill0
```

| | | UserObject | | 0 |
|---|---|---|---|---|
| 1 | | { "name": "John"} | [ "C #", "SQL"] | # |

## CROSS APPLY OPENJSON JSON

. , .

```
DECLARE @json nvarchar(1000) =
N'[
    {
        "id":1,
        "user":{"name":"John"},
        "hobbies":[
            {"name": "Reading"},
            {"name": "Surfing"}
        ]
    },
    {
        "id":2,
        "user":{"name":"Jane"},
        "hobbies":[
            {"name": "Programming"},
            {"name": "Running"}
        ]
    }
 ]'
```

```
SELECT
    JSON_VALUE(person.value, '$.id') as Id,
    JSON_VALUE(person.value, '$.user.name') as PersonName,
    JSON_VALUE(hobbies.value, '$.name') as Hobby
FROM OPENJSON (@json) as person
    CROSS APPLY OPENJSON(person.value, '$.hobbies') as hobbies
```

WITH .

```
SELECT
    Id, person.PersonName, Hobby
FROM OPENJSON (@json)
WITH(
    Id int '$.id',
    PersonName nvarchar(100) '$.user.name',
    Hobbies nvarchar(max) '$.hobbies' AS JSON
) as person
CROSS APPLY OPENJSON(Hobbies)
WITH(
    Hobby nvarchar(100) '$.name'
```

```
)
```

| | PersonName | |
|---|---|---|
| 1 | | |
| 1 | | |
| 2 | | |
| 2 | | |

## JSON

JSON SQL Server JSON .

```
CREATE TABLE JsonTable
(
    id int identity primary key,
    jsonInfo nvarchar(max),
    CONSTRAINT [Content should be formatted as JSON]
    CHECK (ISJSON(jsonInfo)>0)
)
```

```
INSERT INTO JsonTable
VALUES(N'{"Name":"John","Age":23}'),
(N'{"Name":"Jane","Age":31}'),
(N'{"Name":"Bob","Age":37}'),
(N'{"Name":"Adam","Age":65}')
GO
```

'Adam' .

```
SELECT *
FROM JsonTable Where
JSON_VALUE(jsonInfo, '$.Name') = 'Adam'
```

SQL Server .

JSON . JSON `$.Name` .

```
ALTER TABLE JsonTable
ADD vName as JSON_VALUE(jsonInfo, '$.Name')

CREATE INDEX idx_name
ON JsonTable(vName)
```

SQL Server .

: SQL `JSON_VALUE(jsonInfo, '$.Name')` `JSON_VALUE(jsonInfo, '$.Name')` , `vName`

## FOR JSON JSON

*FOR JSON* **WITHOUT_ARRAY_WRAPPER** JSON . .

: JSON .

( )

| | | |
|---|---|---|
| 1 | | 23 |
| 2 | | 31 |

```
SELECT Id, Name, Age
FROM People
WHERE Id = 1
FOR JSON PATH, WITHOUT_ARRAY_WRAPPER
```

```
{"Id":1,"Name":"John","Age":23}
```

## OPENJSON JSON

**OPENJSON** JSON . WITH . . WITH SQL . / JSON .

```
DECLARE @json NVARCHAR(100) = '{"id": 1, "user":{"name":"John"}, "skills":["C#","SQL"]}'

SELECT *
FROM OPENJSON (@json)
    WITH(Id int '$.id',
        Name nvarchar(100) '$.user.name',
        UserObject nvarchar(max) '$.user' AS JSON,
        Skills nvarchar(max) '$.skills' AS JSON,
        Skill0 nvarchar(20) '$.skills[0]')
```

| | | UserObject | | 0 |
|---|---|---|---|---|
| 1 | | { "name": "John"} | [ "C #", "SQL"] | # |

SQL Server JSON : https://riptutorial.com/ko/sql-server/topic/2568/sql-server-json

# 25: SQL Server

## Examples

### STUFF

SubjectId Student . subjectId .

SQL Server

```
create table #yourstudent (subjectid int, studentname varchar(10))

insert into #yourstudent (subjectid, studentname) values
 ( 1       ,'Mary'    )
,( 1       ,'John'    )
,( 1       ,'Sam'    )
,( 2       ,'Alaina')
,( 2       ,'Edward')

select subjectid,  stuff(( select concat( ',', studentname) from #yourstudent y where
y.subjectid = u.subjectid for xml path('')),1,1, '')
    from #yourstudent u
    group by subjectid
```

### String_Agg

SQL Server 2017 vnext STRING_AGG . ,

```
create table #yourstudent (subjectid int, studentname varchar(10))

insert into #yourstudent (subjectid, studentname) values
 ( 1       ,'Mary'    )
,( 1       ,'John'    )
,( 1       ,'Sam'    )
,( 2       ,'Alaina')
,( 2       ,'Edward')

select subjectid, string_agg(studentname, ',') from #yourstudent
    group by subjectid
```

SQL Server : https://riptutorial.com/ko/sql-server/topic/9892/sql-server---

# 26: SQL Server

SQL Server

## Examples

### AS

ANSI SQL  RDBMS .  .

```
CREATE TABLE AliasNameDemo (id INT,firstname VARCHAR(20),lastname VARCHAR(20))

INSERT INTO AliasNameDemo
VALUES      (1,'MyFirstName','MyLastName')

SELECT FirstName +' '+ LastName As FullName
FROM   AliasNameDemo
```

### =

.  .  .  .

```
CREATE TABLE AliasNameDemo (id INT,firstname VARCHAR(20),lastname VARCHAR(20))

INSERT INTO AliasNameDemo
VALUES      (1,'MyFirstName','MyLastName')

SELECT FullName = FirstName +' '+ LastName
FROM   AliasNameDemo
```

.

```
CREATE TABLE AliasNameDemo(id INT,firstname VARCHAR(20),lastname VARCHAR(20))

INSERT INTO AliasNameDemo
VALUES      (1,'MyFirstName','MyLastName')

SELECT *
FROM   (SELECT firstname + ' ' + lastname
        FROM   AliasNameDemo) a (fullname)
```

- 

### AS

AS  . AS  .

```
CREATE TABLE AliasNameDemo (id INT,firstname VARCHAR(20),lastname VARCHAR(20))

INSERT INTO AliasNameDemo
```

```
VALUES      (1,'MyFirstName','MyLastName')

SELECT FirstName +' '+ LastName FullName
FROM   AliasNameDemo
```

SQL Server : https://riptutorial.com/ko/sql-server/topic/10784/sql-server--

# 27: SQLCMD

SQLCMD.exe PATH .

## Examples

### SQLCMD.exe

```
echo off

cls

sqlcmd.exe -S "your server name" -U "sql user name" -P "sql password" -d "name of databse" -Q
"here you may write your query/stored procedure"
```

. SQL Server Express ( ) .

SQLCMD : https://riptutorial.com/ko/sql-server/topic/5396/sqlcmd

# 28: SQLCMD  txt

- sqlcmd -S SHERAZM-E7450 \ SQL2008R2 -d Baseline_DB_Aug_2016 -o c : \ employee.txt -Q "select * from employee"

## Examples

### SQLCMD

sqlcmd -S yourservername \ instancename -d database_name -o outputfilename_withpath -Q " "

.

-S

-d

-o  ( )

-Q

SQLCMD  txt  : https://riptutorial.com/ko/sql-server/topic/7076/sqlcmd--txt---

# 29: STUFF

| | |
|---|---|
| character_expression | |
| start_position | character_expression length replacement_string . |
| | character_expression character_expression |
| replacement_string | character_expression |

## Examples

### STUFF ()

STUFF() . "Svr" "Server" . start_position length .

```
SELECT STUFF('SQL Svr Documentation', 5, 3, 'Server')
```

SQL Svr Documentation. SQL Server Documentation SQL Svr Documentation.

### FOR XML

FOR XML .

Customers .

```
SELECT
    STUFF( (SELECT ';' + Email
        FROM Customers
        where (Email is not null and Email <> '')
        ORDER BY Email ASC
        FOR XML PATH('')),
    1, 1, '')
```

FOR XML PATH('')) ; . STUFF ; . STUFF XML varchar .

: XML . , < < FOR XML PATH, TYPE).value('.[1]','varchar(MAX)') FOR XML PATH('')) FOR XML PATH, TYPE).value('.[1]','varchar(MAX)')

```
SELECT
    STUFF( (SELECT ';' + Email
        FROM Customers
        where (Email is not null and Email <> '')
        ORDER BY Email ASC
        FOR XML PATH, TYPE).value('.[1]','varchar(900)'),
    1, 1, '')
```

MySQL `GROUP_CONCAT` PostgreSQL 9.0+ `string_agg` GROUP BY . ( , `GROUP_CONCAT` ).

## ( )

```
/*
The result can be use for fast way to use columns on Insertion/Updates.
Works with tables and views.

Example: eTableColumns  'Customers'
ColumnNames
------------------------------------------------------
Id, FName, LName, Email, PhoneNumber, PreferredContact

INSERT INTO Customers (Id, FName, LName, Email, PhoneNumber, PreferredContact)
    VALUES (5, 'Ringo', 'Star', 'two@beatles.now', NULL, 'EMAIL')
*/
CREATE PROCEDURE eTableColumns (@Table VARCHAR(100))
AS
SELECT ColumnNames =
   STUFF( (SELECT ', ' +  c.name
FROM
    sys.columns c
INNER JOIN
    sys.types t ON c.user_type_id = t.user_type_id
WHERE
    c.object_id = OBJECT_ID( @Table)
        FOR XML PATH, TYPE).value('.[1]','varchar(2000)'),
    1, 1, '')
GO
```

## SQL

`FOR XML PATH STUFF` .

```
  select distinct t1.id,
      STUFF(
             (SELECT ', ' + convert(varchar(10), t2.date, 120)
              FROM yourtable t2
              where t1.id = t2.id
              FOR XML PATH (''))
              , 1, 1, '')  AS date
    from yourtable t1;
```

## STUFF () .

STUFF (Original_Expression, Start, Length, Replacement_expression)

STUFF () Replacement_expression Length .

```
 Select FirstName, LastName,Email, STUFF(Email, 2, 3, '*****') as StuffedEmail From Employee
```

.

| | | | StuffedEmail |
|---|---|---|---|
| | | James@hotmail.com | J*****s@hotmail.com |
| | | Shyam@hotmail.com | S*****m@hotmail.com |
| | | Ram@hotmail.com | R ***** hotmail.com |

STUFF : https://riptutorial.com/ko/sql-server/topic/703/stuff-

# 30: TEMP

.

.

tempdb .

?

1. .
2. .
3. .

.

## Examples

- .

  .

  # (#temp) .

  ```
  CREATE TABLE #LocalTempTable(
                 StudentID      int,
                 StudentName    varchar(50),
                 StudentAddress varchar(150))
  ```

```
insert into #LocalTempTable values ( 1, 'Ram','India');

select * from #LocalTempTable
```

.

```
"Invalid object name #LocalTempTable"
```

- ## (## temp) .

  .

  .

  ```
  CREATE TABLE ##NewGlobalTempTable(
                 StudentID      int,
                 StudentName    varchar(50),
                 StudentAddress varchar(150))
  ```

```
Insert Into ##NewGlobalTempTable values ( 1,'Ram','India');
Select * from ##NewGlobalTempTable
```

: .

ID ( , ). .

```
There is already an object named '#tempTable' in the database.
```

. .

```
drop table #tempTable
```

( : ) .

```
Cannot drop the table '#tempTable', because it does not exist or you do not have permission.
```

. .

```
IF OBJECT_ID ('tempdb..#tempTable', 'U') is not null DROP TABLE #tempTable
```

TEMP : https://riptutorial.com/ko/sql-server/topic/5328/temp--

# 31: TRY / CATCH

TRY / CATCH MS SQL Server T-SQL .

.NET T-SQL .

## Examples

### TRY / CATCH

datetime .

```
BEGIN TRANSACTION
BEGIN TRY
    INSERT INTO dbo.Sale(Price, SaleDate, Quantity)
    VALUES (5.2, GETDATE(), 1)
    INSERT INTO dbo.Sale(Price, SaleDate, Quantity)
    VALUES (5.2, 'not a date', 1)
    COMMIT TRANSACTION
END TRY
BEGIN CATCH
    ROLLBACK TRANSACTION -- First Rollback and then throw.
    THROW
END CATCH
```

.

```
BEGIN TRANSACTION
BEGIN TRY
    INSERT INTO dbo.Sale(Price, SaleDate, Quantity)
    VALUES (5.2, GETDATE(), 1)
    INSERT INTO dbo.Sale(Price, SaleDate, Quantity)
    VALUES (5.2, GETDATE(), 1)
    COMMIT TRANSACTION
END TRY
BEGIN CATCH
    THROW
    ROLLBACK TRANSACTION
END CATCH
```

### try-catch

RAISERROR TRY CATCH .

```
DECLARE @msg nvarchar(50) = 'Here is a problem!'
BEGIN TRY
    print 'First statement';
    RAISERROR(@msg, 11, 1);
    print 'Second statement';
END TRY
BEGIN CATCH
    print 'Error: ' + ERROR_MESSAGE();
```

```
END CATCH
```

10 RAISERROR ( 11) TRY BLOCK CATCH . ERROR_MESSAGE () . .

```
First statement
Error: Here is a problem!
```

## try catch .

( ) 10 RAISERROR throw .

```
BEGIN TRY
    print 'First statement';
    RAISERROR( 'Here is a problem!', 10, 15);
    print 'Second statement';
END TRY
BEGIN CATCH
    print 'Error: ' + ERROR_MESSAGE();
END CATCH
```

RAISERROR CATCH . :

```
First statement
Here is a problem!
Second statement
```

## RAISERROR

TRHOW CATCH catch throw .

```
DECLARE @msg nvarchar(50) = 'Here is a problem! Area: ''%s'' Line:''%i'''
BEGIN TRY
    print 'First statement';
    RAISERROR(@msg, 11, 1, 'TRY BLOCK', 2);
    print 'Second statement';
END TRY
BEGIN CATCH
    print 'Error: ' + ERROR_MESSAGE();
    THROW;
END CATCH
```

( ) . . :

```
First statement
Error: Here is a problem! Area: 'TRY BLOCK' Line:'2'
Msg 50000, Level 11, State 1, Line 26
Here is a problem! Area: 'TRY BLOCK' Line:'2'
```

## TRY / CATCH

try catch throw .

```
DECLARE @msg nvarchar(50) = 'Here is a problem!'
BEGIN TRY
    print 'First statement';
    THROW 51000, @msg, 15;
    print 'Second statement';
END TRY
BEGIN CATCH
    print 'Error: ' + ERROR_MESSAGE();
    THROW;
END CATCH
```

CATCH  THROW  throw.

```
First statement
Error: Here is a problem!
Msg 51000, Level 16, State 15, Line 39
Here is a problem!
```

RAISERROR  .

- RASIERROR  THROW  .
- THROW  ( )  RAISERROR sys.messages  ID  .
- ( 16) ( )
- THROW RAISERROR  .  RAISERROR  FORMATMESSAGE  .

TRY / CATCH  : https://riptutorial.com/ko/sql-server/topic/5189/try---catch

# 32: WHILE

`WHILE` SQL Server .

.

## Examples

### While

`WHILE CURSORS` . 0 99 .

```
 DECLARE @i int = 0;
 WHILE(@i < 100)
 BEGIN
    PRINT @i;
    SET @i = @i+1
 END
```

### min while

```
DECLARE @ID AS INT;

SET @ID = (SELECT MIN(ID) from TABLE);

WHILE @ID IS NOT NULL
BEGIN
    PRINT @ID;
    SET @ID = (SELECT MIN(ID) FROM TABLE WHERE ID > @ID);
END
```

WHILE : https://riptutorial.com/ko/sql-server/topic/4249/while-

# 33: Windows SQL Server

## Examples

Editions Matrix SQL Server .

- : . -RDBMS . 10G . .
- : . .
- Enterprise Edition : SQL Server . .
- Developer Edition : Enterprise Edition .

SQL Server / GUI SQLSetup.exe .

, . GUI SQLSetup.exe .

Windows SQL Server : https://riptutorial.com/ko/sql-server/topic/5801/windows-sql-server-

# 34:

## Examples

.

```
ALTER DATABASE tpch SET QUERY_STORE = ON
```

SQL Server / Azure SQL sys.query_store .

- sys.query_store_query
- sys.query_store_query_text
- sys.query_store_plan
- sys.query_store_runtime_stats
- sys.query_store_runtime_stats_interval
- sys.database_query_store_options
- sys.query_context_settings

### SQL /

qeries, , CPU , .

```
SELECT Txt.query_text_id, Txt.query_sql_text, Pl.plan_id,
        avg_duration, avg_cpu_time,
        avg_physical_io_reads, avg_logical_io_reads
FROM sys.query_store_plan AS Pl
JOIN sys.query_store_query AS Qry
    ON Pl.query_id = Qry.query_id
JOIN sys.query_store_query_text AS Txt
    ON Qry.query_text_id = Txt.query_text_id
JOIN sys.query_store_runtime_stats Stats
    ON Pl.plan_id = Stats.plan_id
```

.

```
EXEC sp_query_store_remove_query 4;
EXEC sp_query_store_remove_plan 3;
```

/ ID.

.

```
EXEC sp_query_store_reset_exec_stats 3;
```

ID.

SQL . QO .

```
EXEC sp_query_store_unforce_plan @query_id, @plan_id
```

QO .

.

```
EXEC sp_query_store_force_plan @query_id, @plan_id
```

QO .

: https://riptutorial.com/ko/sql-server/topic/7349/-

# 35:

. ." "

## Examples

:

```
SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED;
SELECT * FROM Products WHERE ProductId=1;
SET TRANSACTION ISOLATION LEVEL REPEATABLE READ; --return to the default one
```

1. READ UNCOMMITTED - . . .
2. REPEATABLE READ - . NONREPEATABLE . NEW ROWS ( where ).
3. SNAPSHOT - . . , . - DB :

```
ALTER DATABASE DBTestName SET ALLOW_SNAPSHOT_ISOLATION ON;GO;
SET TRANSACTION ISOLATION LEVEL SNAPSHOT;
```

4. READ COMMITTED - SQL . . . DB . READ_COMMITTED_SNAPSHOT - . .

```
ALTER DATABASE DBTestName SET ALLOW_SNAPSHOT_ISOLATION ON;GO;
SET TRANSACTION ISOLATION LEVEL READ COMMITTED; --return to the default one
```

5. SERIALIZABLE - , , . DataBase .

```
SET TRANSACTION ISOLATION LEVEL SERIALIZABLE ;
```

: https://riptutorial.com/ko/sql-server/topic/5331/---

# 36:

## Examples

.

. , ,  .

```
Create table NetProfit
(
    SalaryToEmployee            int,
    BonusDistributed            int,
    BusinessRunningCost         int,
    BusinessMaintenanceCost     int,
    BusinessEarnings            int,
    BusinessNetIncome
                As BusinessEarnings - (SalaryToEmployee          +
                                       BonusDistributed          +
                                       BusinessRunningCost       +
                                       BusinessMaintenanceCost    )

)
```

.

```
Insert Into NetProfit
    (SalaryToEmployee,
     BonusDistributed,
     BusinessRunningCost,
     BusinessMaintenanceCost,
     BusinessEarnings)
Values
    (1000000,
     10000,
     1000000,
     50000,
     2500000)
```

```
CREATE TABLE [dbo].[ProcessLog](
[LogId] [int] IDENTITY(1,1) NOT NULL,
[LogType] [varchar](20) NULL,
[StartTime] [datetime] NULL,
[EndTime] [datetime] NULL,
[RunMinutes]  AS
(datediff(minute,coalesce([StartTime],getdate()),coalesce([EndTime],getdate())))
```

..

: https://riptutorial.com/ko/sql-server/topic/5561/--

# 37:

## Examples

```
Exec sp_configure 'show advanced options' ,1
RECONFIGURE
GO
-- Show all configure
sp_configure
```

```
Exec sp_configure 'backup compression default',1
GO
RECONFIGURE;
```

```
sp_configure 'fill factor', 100;
GO
RECONFIGURE;
```

.

```
USE master;
GO
-- Set recovery every 3 min
EXEC sp_configure 'recovery interval', '3';
RECONFIGURE WITH OVERRIDE;
```

### cmd

```
EXEC sp_configure 'xp_cmdshell', 1
GO
RECONFIGURE
```

```
USE master
EXEC sp_configure 'max server memory (MB)', 64
RECONFIGURE WITH OVERRIDE
```

```
EXEC sp_configure "number of checkpoint tasks", 4
```

: https://riptutorial.com/ko/sql-server/topic/5185/-

# 38:

- COALESCE ([Column1], [Column2] .... [ColumnN]

## Examples

### COALESCE

.

. BEGIN END .

```
BEGIN

    --Table variable declaration to store sample records
    DECLARE @Table TABLE (FirstName varchar(256), LastName varchar(256))

    --Inserting sample records into table variable @Table
    INSERT INTO @Table (FirstName, LastName)
    VALUES
    ('John','Smith'),
    ('Jane','Doe')

    --Creating variable to store result
    DECLARE @Names varchar(4000)

    --Used COLESCE function, so it will concatenate comma seperated FirstName into @Names
varible
    SELECT @Names = COALESCE(@Names + ',', '') + FirstName
    FROM @Table

    --Now selecting actual result
    SELECT @Names
    END
```

COALESCE() NON NULL . . NON NULL .

```
DECLARE @Table TABLE (UserID int, PhoneNumber varchar(12), CellNumber varchar(12))
INSERT INTO @Table (UserID, PhoneNumber, CellNumber)
VALUES
(1,'555-869-1123',NULL),
(2,'555-123-7415','555-846-7786'),
(3,NULL,'555-456-8521')


SELECT
    UserID,
    COALESCE(PhoneNumber, CellNumber)
FROM
    @Table
```

### null

```
SELECT COALESCE(NULL, NULL, 'TechOnTheNet.com', NULL, 'CheckYourMath.com');
Result: 'TechOnTheNet.com'

SELECT COALESCE(NULL, 'TechOnTheNet.com', 'CheckYourMath.com');
Result: 'TechOnTheNet.com'

SELECT COALESCE(NULL, NULL, 1, 2, 3, NULL, 4);
Result: 1
```

: https://riptutorial.com/ko/sql-server/topic/3234/

# 39:

2 .

X / Y

() / . SQL Server . SRID (Spatial Reference ID) 4326 . SRID.

## Examples

Point . .

| | |
|---|---|
| X | Point x float . |
| Y | Point y float . |
| | / Well Known Text (WKB) |
| | / Well Known Binary (WKB) |
| SRID | geometry / geography ID (SRID) int . |

```
--Explicit constructor
DECLARE @gm1 GEOMETRY = GEOMETRY::Point(10,5,0)

DECLARE @gg1 GEOGRAPHY = GEOGRAPHY::Point(51.511601,-0.096600,4326)

--Implicit constructor (using WKT - Well Known Text)
DECLARE @gm1 GEOMETRY = GEOMETRY::STGeomFromText('POINT(5 10)', 0)

DECLARE @gg1 GEOGRAPHY= GEOGRAPHY::STGeomFromText('POINT(-0.096600 51.511601)', 4326)

--Implicit constructor (using WKB - Well Known Binary)
DECLARE @gm1 GEOMETRY = GEOMETRY::STGeomFromWKB(0x0101000000000000000000144000000000000002440,
0)

DECLARE @gg1 GEOGRAPHY= GEOGRAPHY::STGeomFromWKB(0x01010000005F29CB10C7BAB8BFEACC3D247CC14940,
4326)
```

: https://riptutorial.com/ko/sql-server/topic/6816/-

# 40:

## Examples

### CLR

CLR . CLR .

```
sp_configure 'show advanced options', 1;
GO
RECONFIGURE;
GO
sp_configure 'clr enabled', 1;
GO
RECONFIGURE;
GO
```

CLR TRUSTWORTHY ON .

```
ALTER DATABASE MyDbWithClr SET TRUSTWORTHY ON
```

### SQL CLR .dll

.Net , , .dll . CLR .dll SQL Server .

```
CREATE ASSEMBLY MyLibrary
FROM 'C:\lib\MyStoredProcedures.dll'
    WITH PERMISSION_SET = EXTERNAL_ACCESS
```

PERMISSION_SET .dll ( : , , ) Safe.

PERMISSION_SET = EXTERNAL_ACCESS .

sys.assemblies CLR .

```
SELECT *
FROM sys.assemblies asms
WHERE is_user_defined = 1
```

### SQL Server CLR

.Net .dll SQL .

```
CREATE FUNCTION dbo.TextCompress(@input nvarchar(max))
RETURNS varbinary(max)
AS EXTERNAL NAME MyLibrary.[Name.Space.ClassName].TextCompress
```

.Net . AS EXTERNAL NAME , / .

CLR .

```
SELECT * FROM dbo.sysobjects WHERE TYPE ='FS'
```

## SQL Server CLR

.Net .dll SQL Server .

```
CREATE TYPE dbo.Point
EXTERNAL NAME MyLibrary.[Name.Space.Point]
```

T-SQL . EXTERNAL NAME , .

## SQL Server CLR

.Net .dll SQL .

```
CREATE PROCEDURE dbo.DoSomethng(@input nvarchar(max))
AS EXTERNAL NAME MyLibrary.[Name.Space.ClassName].DoSomething
```

.Net . AS EXTERNAL NAME , / .

: https://riptutorial.com/ko/sql-server/topic/7116/---

# 41:

- WITH *cte_name* [( *column_name_1* , *column_name_2* , ...)] AS ( *cte_expression* )

( ; ) CTE .

```
;WITH CommonTableName (...) SELECT ... FROM CommonTableName ...
```

CTE . CTE , CTE CTE CTE CTE .

## Examples

```
CREATE TABLE dbo.Employees
(
    EmployeeID INT NOT NULL PRIMARY KEY,
    FirstName NVARCHAR(50) NOT NULL,
    LastName NVARCHAR(50) NOT NULL,
    ManagerID INT NULL
)

GO

INSERT INTO Employees VALUES (101, 'Ken', 'Sánchez', NULL)
INSERT INTO Employees VALUES (102, 'Keith', 'Hall', 101)
INSERT INTO Employees VALUES (103, 'Fred', 'Bloggs', 101)
INSERT INTO Employees VALUES (104, 'Joseph', 'Walker', 102)
INSERT INTO Employees VALUES (105, 'Žydrė', 'Klybė', 101)
INSERT INTO Employees VALUES (106, 'Sam', 'Jackson', 105)
INSERT INTO Employees VALUES (107, 'Peter', 'Miller', 103)
INSERT INTO Employees VALUES (108, 'Chloe', 'Samuels', 105)
INSERT INTO Employees VALUES (109, 'George', 'Weasley', 105)
INSERT INTO Employees VALUES (110, 'Michael', 'Kensington', 106)
```

```
;WITH cteReports (EmpID, FirstName, LastName, SupervisorID, EmpLevel) AS
(
    SELECT EmployeeID, FirstName, LastName, ManagerID, 1
    FROM Employees
    WHERE ManagerID IS NULL

    UNION ALL

    SELECT e.EmployeeID, e.FirstName, e.LastName, e.ManagerID, r.EmpLevel + 1
    FROM Employees        AS e
    INNER JOIN cteReports AS r ON e.ManagerID = r.EmpID
)

SELECT
    FirstName + ' ' + LastName AS FullName,
    EmpLevel,
    (SELECT FirstName + ' ' + LastName FROM Employees WHERE EmployeeID =
cteReports.SupervisorID) AS ManagerName
FROM cteReports
```

```
ORDER BY EmpLevel, SupervisorID
```

:

| FullName | EmpLevel | ManagerName |
|---|---|---|
| | 1 | |
| | 2 | |
| | 2 | |
| | 2 | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | 4 | |

## CTE n

:

```
|  ID  | FirstName | LastName | Gender | Salary |
+------+-----------+----------+--------+--------+
| 1    | Jahangir  | Alam     | Male   | 70000  |
| 2    | Arifur    | Rahman   | Male   | 60000  |
| 3    | Oli       | Ahammed  | Male   | 45000  |
| 4    | Sima      | Sultana  | Female | 70000  |
| 5    | Sudeepta  | Roy      | Male   | 80000  |
+------+-----------+----------+--------+--------+
```

CTE ( ) :

```
 WITH RESULT AS
(
    SELECT SALARY,
           DENSE_RANK() OVER (ORDER BY SALARY DESC) AS DENSERANK
    FROM EMPLOYEES
)
SELECT TOP 1 SALARY
FROM RESULT
WHERE DENSERANK = 1
```

N 2 . 3  N 3 .

## CTE

:

```
|  ID  | FirstName | LastName | Gender | Salary |
+------+-----------+----------+--------+--------+
|  1   | Mark      | Hastings | Male   | 60000  |
|  1   | Mark      | Hastings | Male   | 60000  |
|  2   | Mary      | Lambeth  | Female | 30000  |
|  2   | Mary      | Lambeth  | Female | 30000  |
|  3   | Ben       | Hoskins  | Male   | 70000  |
|  3   | Ben       | Hoskins  | Male   | 70000  |
|  3   | Ben       | Hoskins  | Male   | 70000  |
+------+-----------+----------+--------+--------+
```

CTE ( ) :

```
WITH EmployeesCTE AS
(
   SELECT *, ROW_NUMBER()OVER(PARTITION BY ID ORDER BY ID) AS RowNumber
   FROM Employees
)
DELETE FROM EmployeesCTE WHERE RowNumber > 1
```

:

```
|  ID  | FirstName | LastName | Gender | Salary |
+------+-----------+----------+--------+--------+
|  1   | Mark      | Hastings | Male   | 60000  |
|  2   | Mary      | Lambeth  | Female | 30000  |
|  3   | Ben       | Hoskins  | Male   | 70000  |
+------+-----------+----------+--------+--------+
```

## CTE

```
DECLARE @startdate CHAR(8), @numberDays TINYINT

SET @startdate = '20160101'
SET @numberDays = 10;

WITH CTE_DatesTable
AS
(
  SELECT CAST(@startdate as date) AS [date]
  UNION ALL
  SELECT DATEADD(dd, 1, [date])
  FROM CTE_DatesTable
  WHERE DATEADD(dd, 1, [date]) <= DateAdd(DAY, @numberDays-1, @startdate)
)

SELECT [date] FROM CTE_DatesTable

OPTION (MAXRECURSION 0)
```

@startdate @numberDays .

## CTE

2011 (2012 - 1) .

```
WITH yearsAgo
(
    myYear
)
AS
(
     -- Base Case: This is where the recursion starts
     SELECT DATEPART(year, GETDATE()) AS myYear

     UNION ALL  -- This MUST be UNION ALL (cannot be UNION)

     -- Recursive Section: This is what we're doing with the recursive call
     SELECT yearsAgo.myYear - 1
     FROM yearsAgo
     WHERE yearsAgo.myYear >= 2012
)
     SELECT myYear FROM yearsAgo;  -- A single SELECT, INSERT, UPDATE, or DELETE
```

| myYear |
|---|
| 2016 |
| 2015 |
| 2014 |
| 2013 |
| 2012 |
| 2011 |

MAXRECURSION ( ).

```
WITH yearsAgo
(
    myYear
)
AS
(
     -- Base Case
     SELECT DATEPART(year , GETDATE()) AS myYear
     UNION ALL
     -- Recursive Section
     SELECT yearsAgo.myYear - 1
     FROM yearsAgo
     WHERE yearsAgo.myYear >= 2002
)
     SELECT * FROM yearsAgo
     OPTION (MAXRECURSION 10);
```

530, 16, 1, 2 . 10 .

## AS CTE

```
;WITH cte_query_1
AS
(
    SELECT *
    FROM database.table1
),
cte_query_2
AS
(
    SELECT *
    FROM database.table2
)
SELECT *
FROM cte_query_1
WHERE cte_query_one.fk IN
(
    SELECT PK
    FROM cte_query_2
)
```

AS u . .

: https://riptutorial.com/ko/sql-server/topic/1343/--

# 42:

## Examples

CROSS APPLY - "" .

(ProductList ) Company . ProductList .

```
SELECT *
FROM Companies c
     CROSS APPLY dbo.GetProductList( c.ProductList ) p
```

*ProductList* .

### JSON

CROSS APPLY JSON "" .

JSON (ProductList ) Company . OPENJSON . OPENJSON JSON "" .

```
SELECT *
FROM Companies c
     CROSS APPLY OPENJSON( c.ProductList )
                 WITH ( Id int, Title nvarchar(30), Price money)
```

*ProductList* JSON WITH OPENJSON .

*STRING_SPLIT* . **CROSS APPLY** *STRING_SPLIT* "" .

( : promo, sales, new) Product . STRING_SPLIT CROSS APPLY .

```
SELECT *
FROM Products p
     CROSS APPLY STRING_SPLIT( p.Tags, ',' ) tags
WHERE tags.value = 'promo'
```

STRING_SPLIT . .

**: SQL Server 2016** *STRING_SPLIT* .

: https://riptutorial.com/ko/sql-server/topic/5462/-

# 43:

## Examples

```
USE AdventureWorks;
GRANT CREATE TABLE TO MelanieK;
GO
```

### SHOWPLAN

```
USE AdventureWorks2012;
GRANT SHOWPLAN TO AuditMonitor;
GO
```

### GRANT OPTION CREATE VIEW

```
USE AdventureWorks2012;
GRANT CREATE VIEW TO CarmineEs WITH GRANT OPTION;
GO
```

```
use YourDatabase
go
exec sp_addrolemember 'db_owner', 'UserName'
go
```

: https://riptutorial.com/ko/sql-server/topic/5333/---

# 44:

. .

( .)

## Examples

### /

```
 -- Identity primary key - unique arbitrary increment number
 create table person (
 id int identity(1,1) primary key not null,
 firstName varchar(100) not null,
 lastName varchar(100) not null,
 dob DateTime not null,
 ssn varchar(9) not null
 )
```

### GUID

```
 -- GUID primary key - arbitrary unique value for table
 create table person (
 id uniqueIdentifier default (newId()) primary key,
 firstName varchar(100) not null,
 lastName varchar(100) not null,
 dob DateTime not null,
 ssn varchar(9) not null
 )
```

```
 -- natural primary key - using an existing piece of data within the table that uniquely
identifies the record
 create table person (
 firstName varchar(100) not null,
 lastName varchar(100) not null,
 dob DateTime not null,
 ssn varchar(9) primary key not null
 )
```

```
 -- composite key - using two or more existing columns within a table to create a primary key
 create table person (
 firstName varchar(100) not null,
 lastName varchar(100) not null,
 dob DateTime not null,
 ssn varchar(9) not null,
 primary key (firstName, lastName, dob)
 )
```

```
ALTER TABLE person
 ADD CONSTRAINT pk_PersonSSN PRIMARY KEY (ssn)
```

( ssn ) .

```
ALTER TABLE Person
 DROP CONSTRAINT pk_PersonSSN
```

: https://riptutorial.com/ko/sql-server/topic/4543/-

# 45: (Hekaton)

## Examples

T-SQL dll C . .

- CREATE PROCEDURE
- NATIVE_COMPILATION
- SCHEMABINDING
- EXECUTE AS OWNER

BEGIN END BEGIN ATOMIC .

```
BEGIN ATOMIC
   WITH (TRANSACTION ISOLATION LEVEL=SNAPSHOT, LANGUAGE='us_english')
   -- T-Sql code goes here
END
```

:

```
CREATE PROCEDURE usp_LoadMemOptTable (@maxRows INT, @FullName NVARCHAR(200))
WITH
    NATIVE_COMPILATION,
    SCHEMABINDING,
    EXECUTE AS OWNER
AS
BEGIN ATOMIC
WITH (TRANSACTION ISOLATION LEVEL=SNAPSHOT, LANGUAGE='us_english')
    DECLARE @i INT = 1
    WHILE @i <= @maxRows
    BEGIN
        INSERT INTO dbo.MemOptTable3 VALUES(@i, @FullName, GETDATE())
        SET @i = @i+1
    END
END
GO
```

C dll . . .

- CREATE FUNCTION
- NATIVE_COMPILATION
- SCHEMABINDING

BEGIN END BEGIN ATOMIC .

```
BEGIN ATOMIC
   WITH (TRANSACTION ISOLATION LEVEL=SNAPSHOT, LANGUAGE='us_english')
   -- T-Sql code goes here
END
```

:

```
CREATE FUNCTION [dbo].[udfMultiply]( @v1 int, @v2 int )
RETURNS bigint
WITH NATIVE_COMPILATION, SCHEMABINDING
AS
BEGIN ATOMIC WITH (TRANSACTION ISOLATION LEVEL = SNAPSHOT, LANGUAGE = N'English')

    DECLARE @ReturnValue bigint;
    SET @ReturnValue = @v1 * @v2;

    RETURN (@ReturnValue);
END

-- usage sample:
SELECT dbo.udfMultiply(10, 12)
```

. C dll . 2016 . .

- CREATE FUNCTION
- NATIVE_COMPILATION
- SCHEMABINDING

BEGIN END BEGIN ATOMIC .

```
BEGIN ATOMIC
   WITH (TRANSACTION ISOLATION LEVEL=SNAPSHOT, LANGUAGE='us_english')
   -- T-Sql code goes here
END
```

:

```
CREATE FUNCTION [dbo].[udft_NativeGetBusinessDoc]
(
   @RunDate VARCHAR(25)
)
RETURNS TABLE
WITH SCHEMABINDING,
     NATIVE_COMPILATION
AS
     RETURN
(
    SELECT BusinessDocNo,
           ProductCode,
           UnitID,
           ReasonID,
           PriceID,
           RunDate,
           ReturnPercent,
           Qty,
           RewardAmount,
           ModifyDate,
           UserID
    FROM dbo.[BusinessDocDetail_11]
    WHERE RunDate >= @RunDate
);
```

(Hekaton) : https://riptutorial.com/ko/sql-server/topic/6089/-----hekaton-

# 46:

## Examples

### Invoices

```
INSERT INTO Invoices [ /* column names may go here */ ]
VALUES (123, '1234abc', '2016-08-05 20:18:25.770', 321, 5, '2016-08-04');
```

- IDENTITY .

```
INSERT INTO Invoices ([ID], [Num], [DateTime], [Total], [Term], [DueDate])
VALUES (123, '1234abc', '2016-08-05 20:18:25.770', 321, 5, '2016-08-25');
```

: https://riptutorial.com/ko/sql-server/topic/5323/-

# 47:

- EOMONTH (  [, month_to_add])

https://msdn.microsoft.com/en-us/library/ms187819.aspx `DateTime` 3ms .

datetime  datetime  .000, .003  .007  .

| | |
|---|---|
| 01/01/98 23 : 59 : 59.999 | 1998-01-02 00 : 00 : 00.000 |
| ------ | ------ |
| 01/01/98 23 : 59 : 59.995 | 1998-01-01 23 : 59 : 59.997 |
| 01/01/98 23 : 59 : 59.996 | |
| 01/01/98 23 : 59 : 59.997 | |
| 01/01/98 23 : 59 : 59.998 | |
| ------ | ------ |
| 01/01/98 23 : 59 : 59.992 | 1998-01-01 23 : 59 : 59.993 |
| 01/01/98 23 : 59 : 59.993 | |
| 01/01/98 23 : 59 : 59.994 | |
| ------ | ------ |
| 01/01/98 23 : 59 : 59.990 | 1998-01-01 23 : 59 : 59.990 |
| 01/01/98 23 : 59 : 59.991 | |
| ------ | ------ |

`time` , `datetime2`  `datetimeoffset` .

## Examples

### CONVERT

CONVERT  datetime  .

```
SELECT GETDATE() AS [Result] -- 2016-07-21 07:56:10.927
```

. SQL Server .

```
DECLARE @convert_code INT = 100 -- See Table Below
SELECT CONVERT(VARCHAR(30), GETDATE(), @convert_code) AS [Result]
```

| @convert_code | |
|---|---|
| **100** | "Jul 21 2016 7:56 AM" |
| **101** | '07/21/2016' |
| **102** | "2016.07.21" |
| **103** | "21/07/2016" |
| **104** | "21.07.2016" |
| **105** | "21-07-2016" |
| **106** | '2016 7 21 ' |
| **107** | "2016 7 21 " |
| **108** | "07:57:05" |
| **109** | "2021 7 21 7 : 57 : 45 : 707AM" |
| **110** | "07-21-2016" |
| **111** | "2016/07/21" |
| **112** | "20160721" |
| **113** | 2116 7 21 07 : 57 : 59 : 553 " |
| **114** | "07 : 57 : 59 : 553" |
| **120** | "2016-07-21 07:57:59" |
| **121** | "2016-07-21 07 : 57 : 59.553" |
| **126** | "2016-07-21T07 : 58 : 34.340" |
| **127** | "2016-07-21T07 : 58 : 34.340" |
| **130** | "16 ???? 1437 7 : 58 : 34 : 340AM" |
| **131** | "16/10/1437 7 : 58 : 34 : 340AM" |

```
SELECT GETDATE() AS [Result]                                    -- 2016-07-21 07:56:10.927
```

```
UNION SELECT CONVERT(VARCHAR(30),GETDATE(),100) AS [Result] -- Jul 21 2016  7:56AM
UNION SELECT CONVERT(VARCHAR(30),GETDATE(),101) AS [Result] -- 07/21/2016
UNION SELECT CONVERT(VARCHAR(30),GETDATE(),102) AS [Result] -- 2016.07.21
UNION SELECT CONVERT(VARCHAR(30),GETDATE(),103) AS [Result] -- 21/07/2016
UNION SELECT CONVERT(VARCHAR(30),GETDATE(),104) AS [Result] -- 21.07.2016
UNION SELECT CONVERT(VARCHAR(30),GETDATE(),105) AS [Result] -- 21-07-2016
UNION SELECT CONVERT(VARCHAR(30),GETDATE(),106) AS [Result] -- 21 Jul 2016
UNION SELECT CONVERT(VARCHAR(30),GETDATE(),107) AS [Result] -- Jul 21, 2016
UNION SELECT CONVERT(VARCHAR(30),GETDATE(),108) AS [Result] -- 07:57:05
UNION SELECT CONVERT(VARCHAR(30),GETDATE(),109) AS [Result] -- Jul 21 2016  7:57:45:707AM
UNION SELECT CONVERT(VARCHAR(30),GETDATE(),110) AS [Result] -- 07-21-2016
UNION SELECT CONVERT(VARCHAR(30),GETDATE(),111) AS [Result] -- 2016/07/21
UNION SELECT CONVERT(VARCHAR(30),GETDATE(),112) AS [Result] -- 20160721
UNION SELECT CONVERT(VARCHAR(30),GETDATE(),113) AS [Result] -- 21 Jul 2016 07:57:59:553
UNION SELECT CONVERT(VARCHAR(30),GETDATE(),114) AS [Result] -- 07:57:59:553
UNION SELECT CONVERT(VARCHAR(30),GETDATE(),120) AS [Result] -- 2016-07-21 07:57:59
UNION SELECT CONVERT(VARCHAR(30),GETDATE(),121) AS [Result] -- 2016-07-21 07:57:59.553
UNION SELECT CONVERT(VARCHAR(30),GETDATE(),126) AS [Result] -- 2016-07-21T07:58:34.340
UNION SELECT CONVERT(VARCHAR(30),GETDATE(),127) AS [Result] -- 2016-07-21T07:58:34.340
UNION SELECT CONVERT(VARCHAR(30),GETDATE(),130) AS [Result] -- 16 ???? 1437  7:58:34:340AM
UNION SELECT CONVERT(VARCHAR(30),GETDATE(),131) AS [Result] -- 16/10/1437  7:58:34:340AM
```

## FORMAT

### SQL Server 2012

`FORMAT()` .

`DATETIME` `VARCHAR` .

```
DECLARE @Date DATETIME = '2016-09-05 00:01:02.333'

SELECT FORMAT(@Date, N'dddd, MMMM dd, yyyy hh:mm:ss tt')
```

2016 9 5 12:01:02 AM

`DATETIME` `2016-09-05 00:01:02.333` , .

| | |
|---|---|
| yyyy | 2016 |
| | 16 |
| MMMM | |
| MM | 09 |
| | 9 |
| dddd | |
| ddd | |
| DD | 05 |

| | |
|---|---|
| | 5 |
| HH | 00 |
| H | 0 |
| hh | 12 |
| h | 12 |
| mm | 01 |
| | 1 |
| SS | 02 |
| | 2 |
| tt | |
| | |
| fff | 333 |
| ff | 33 |
| | |

FORMAT() .

```
DECLARE @Date DATETIME = '2016-09-05 00:01:02.333'

SELECT FORMAT(@Date, N'U')
```

2016 9 5 4:01:02

| | |
|---|---|
| | 2016 9 5 |
| | 2014 9 5 |
| | 2016 9 5 12:01:02 AM |
| | 2016 9 5 12:01 |
| | 9/5/2016 12:01:02 AM |
| | 9/5/2016 12:01 AM |
| | 9 5 |
| | 2016-09-05T00 : 01 : 02.3330000 |

| |
|---|
| 2016 9 5 , 00:01:02 GMT |
| 2016-09-05T00 : 01 : 02 |
| 12:01:02 AM |
| 12:01 |
| 2016 9 5 4:01:02 |
| 2016-09-05 00 : 01 : 02Z |
| 2016 9 |

: *en-US* . *FORMAT()* .

```
DECLARE @Date DATETIME = '2016-09-05 00:01:02.333'

SELECT FORMAT(@Date, N'U', 'zh-cn')
```

2016 9 5 4:01:02

## DateTime

GETDATE GETUTCDATE .

SQL Server .

GETDATE . GETUTCDATE UTC .

WHERE SELECT .

:

```
-- example query that selects the current time in both the server time zone and UTC
SELECT GETDATE() as SystemDateTime, GETUTCDATE() as UTCDateTime

-- example query records with EventDate in the past.
SELECT * FROM MyEvents WHERE EventDate < GETDATE()
```

- .

```
SELECT
    GETDATE(),          --2016-07-21 14:27:37.447
    GETUTCDATE(),       --2016-07-21 18:27:37.447
    CURRENT_TIMESTAMP,  --2016-07-21 14:27:37.447
    SYSDATETIME(),      --2016-07-21 14:27:37.4485768
    SYSDATETIMEOFFSET(),--2016-07-21 14:27:37.4485768 -04:00
    SYSUTCDATETIME()    --2016-07-21 18:27:37.4485768
```

## DATEADD

:

```
DATEADD (datepart , number , datetime_expr)
```

number . number .

```
DECLARE @now DATETIME2 = GETDATE();
SELECT @now;                          --2016-07-21 14:39:46.4170000
SELECT DATEADD(YEAR, 1, @now)         --2017-07-21 14:39:46.4170000
SELECT DATEADD(QUARTER, 1, @now)      --2016-10-21 14:39:46.4170000
SELECT DATEADD(WEEK, 1, @now)         --2016-07-28 14:39:46.4170000
SELECT DATEADD(DAY, 1, @now)          --2016-07-22 14:39:46.4170000
SELECT DATEADD(HOUR, 1, @now)         --2016-07-21 15:39:46.4170000
SELECT DATEADD(MINUTE, 1, @now)       --2016-07-21 14:40:46.4170000
SELECT DATEADD(SECOND, 1, @now)       --2016-07-21 14:39:47.4170000
SELECT DATEADD(MILLISECOND, 1, @now)--2016-07-21 14:39:46.4180000
```

: DATEADD datepart datepart . ( m vs mi , ww vs w ).

datepart .

| datepart | |
|---|---|
| | yy, yyyy |
| | qq, q |
| | mm, m |
| | dy, y |
| | dd, d |
| | wk, ww |
| | dw, w |
| | hh |
| | mi, n |
| | ss, s |
| | ms |
| | |
| | ns |

: ( m vs mi , ww vs w ). datepart datepart ( month , minute , week , weekday ).

## DATEDIFF

:

```
DATEDIFF (datepart, datetime_expr1, datetime_expr2)
```

datetime_expr  datetime_expr2 , .

```
DECLARE @now DATETIME2 = GETDATE();
DECLARE @oneYearAgo DATETIME2 = DATEADD(YEAR, -1, @now);
SELECT @now                                  --2016-07-21 14:49:50.9800000
SELECT @oneYearAgo                           --2015-07-21 14:49:50.9800000
SELECT DATEDIFF(YEAR, @oneYearAgo, @now)       --1
SELECT DATEDIFF(QUARTER, @oneYearAgo, @now)    --4
SELECT DATEDIFF(WEEK, @oneYearAgo, @now)       --52
SELECT DATEDIFF(DAY, @oneYearAgo, @now)        --366
SELECT DATEDIFF(HOUR, @oneYearAgo, @now)       --8784
SELECT DATEDIFF(MINUTE, @oneYearAgo, @now)     --527040
SELECT DATEDIFF(SECOND, @oneYearAgo, @now)     --31622400
```

: DATEDIFF datepart  .  ( m vs mi , ww vs w ).

DATEDIFF UTC SQL Server  . UTC ( )  .

```
select  DATEDIFF(hh, getutcdate(), getdate()) as 'CentralTimeOffset'
```

DATEPART datetime datepart  .

DATENAME  datepart  . DATENAME  DATENAME .

datetime ,  , datepart DATEPART .

:

```
DATEPART ( datepart , datetime_expr )
DATENAME ( datepart , datetime_expr )
DAY ( datetime_expr )
MONTH ( datetime_expr )
YEAR ( datetime_expr )
```

:

```
DECLARE @now DATETIME2 = GETDATE();
SELECT @now                       --2016-07-21 15:05:33.8370000
SELECT DATEPART(YEAR, @now)       --2016
SELECT DATEPART(QUARTER, @now)    --3
SELECT DATEPART(WEEK, @now)       --30
SELECT DATEPART(HOUR, @now)       --15
SELECT DATEPART(MINUTE, @now)     --5
SELECT DATEPART(SECOND, @now)     --33
-- Differences between DATEPART and DATENAME:
SELECT DATEPART(MONTH, @now)      --7
SELECT DATENAME(MONTH, @now)      --July
```

```
SELECT DATEPART(WEEKDAY, @now)    --5
SELECT DATENAME(WEEKDAY, @now)    --Thursday
--shorthand functions
SELECT DAY(@now)    --21
SELECT MONTH(@now)  --7
SELECT YEAR(@now)   --2016
```

: DATEPART DATENAME datepart . ( m VS mi , ww VS w ).

DATEADD DATEDIFF .

```
SELECT DATEADD(d, -1, DATEADD(m, DATEDIFF(m, 0, '2016-09-23') + 1, 0))
-- 2016-09-30 00:00:00.000
```

### SQL Server 2012

EOMONTH .

```
SELECT EOMONTH('2016-07-21')        --2016-07-31
SELECT EOMONTH('2016-07-21', 4)     --2016-11-30
SELECT EOMONTH('2016-07-21', -5)    --2016-02-29
```

## DateTime

DateTime Date .

1. SELECT CONVERT(Date, GETDATE())
2. SELECT DATEADD(dd, 0, DATEDIFF(dd, 0, GETDATE())) 2016-07-21 00 : 00 : 00.000 .
3. SELECT CAST(GETDATE() AS DATE)
4. SELECT CONVERT(CHAR(10),GETDATE(),111)
5. SELECT FORMAT(GETDATE(), 'yyyy-MM-dd')

4 5 .

2 datetime DOB .

```
  CREATE FUNCTION [dbo].[Calc_Age]
    (
    @DOB datetime , @calcDate datetime
    )
    RETURNS int
    AS
   BEGIN
declare @age int

IF (@calcDate < @DOB  )
RETURN -1

-- If a DOB is supplied after the comparison date, then return -1
SELECT @age = YEAR(@calcDate) - YEAR(@DOB) +
  CASE WHEN DATEADD(year,YEAR(@calcDate) - YEAR(@DOB)
  ,@DOB) > @calcDate THEN -1 ELSE 0 END
```

```
RETURN @age


END
```

: 1/1/2000 .

```
SELECT  dbo.Calc_Age('2000-01-01',Getdate())
```

## CROSS

SQL Server 2012

Transact SQL `[DATEFROMPARTS][1]` ( `[DATETIMEFROMPARTS][1]` ) `Date` `DateTime` .

```
 DECLARE @myDate DATE=DATEFROMPARTS(1988,11,28)
 DECLARE @someMoment DATETIME=DATEFROMPARTS(1988,11,28,10,30,50,123)
```

`DATEFROMPARTS` Year, Month, Day `DATETIMEFROMPARTS` , , ,, , .

. ( datetime) , .

| | SQL | |
|---|---|---|
| YY-MM-DD | (CONVERT (VARCHAR (10), SYSDATETIME (), 20), 8) [YY-MM-DD]<br>(CONVERT (VARCHAR (8), SYSDATETIME (), 11), '/', '-') AS [YY-MM-DD] | 11-06-08 |
| YYYY-MM-DD | SELECT CONVERT (VARCHAR (10), SYSDATETIME (), 120) AS [YYYY-MM-DD]<br>SELECT [YYYY-MM-DD]  (CONVERT (VARCHAR (10), SYSDATETIME (), 111), '/', '-' | 2011-06-08 |
| YYYY-MD | SELECT VARCHAR (2)) + '-'+ CAST (DAY (SYSDATETIME ()) AS VARCHAR (4)) + '-'+ CAST ( (SYSDATETIME ()) AS VARCHAR 2)) AS [YYYY-MD] | 2011-6-8 |
| YY-MD | + VARCHAR (2)) + '-'+ CAST (DAY (SYSDATETIME (2) )) AS VARCHAR (2)) AS [YY-MD] | 11-6-8 |
| MD-YYYY | YYAR (SYSDATETIME ()) AS VARCHAR (2)) + '-'+ CAST ( (SYSDATETIME ()) AS VARCHAR (2) + '-'+ CAST 4)) AS [MD-YYYY] | 6-8-2011 |
| MD-YY | SELECT CAST (MONTH (SYSDATETIME ()) AS VARCHAR (2) + '-'+ CAST (DAY (SYSDATETIME ()) AS VARCHAR (2)) + '-'+ RIGHT (CAST (YEAR (SYSDATETIME VARCHAR (4)), 2) AS [MD-YY] | 6-8-11 |

| | SQL | |
|---|---|---|
| DM-YYYY | SELECT CAST (DAY (SYSDATETIME ()) AS VARCHAR (2)) + '-'+ CAST (MONTH (SYSDATETIME ()) AS VARCHAR (2)) + '-'+ CAST (YEAR (SYSDATETIME ()) AS VARCHAR 4)) AS [DM-YYYY] | 8-6-2011 |
| DM-YY | SELECT CAST (DAY (SYSDATETIME ()) AS VARCHAR (2)) + '-'+ CAST (MONTH (SYSDATETIME ()) AS VARCHAR (2)) + '-'+ RIGHT (CAST (YEAR (SYSDATETIME VARCHAR (4)), 2) AS [DM-YY] | 8-6-11 |
| YY-MM | (CONVERT (VARCHAR (7), SYSDATETIME (), 20), 5) [YY-MM]<br>SELECT SUBSTRING (CONVERT (VARCHAR (10), SYSDATETIME (), 120), 3, 5) [YY-MM] | 11-06 |
| YYYY-MM | (VARCHAR (7), SYSDATETIME (), 120) [YYYY-MM] | 2011-06 |
| YY-M | 2) + '-'+ CAST (MONTH (SYSDATETIME ()) AS VARCHAR (2)) AS [YY-M] | 11-6 |
| YYYY-M | [VARCHAR (4)) + '-'+ CAST (MONTH (SYSDATETIME ()) AS VARCHAR (2)) AS [YYYY-M] | 2011-6 |
| MM-YY | SELECT RIGHT (CONVERT (VARCHAR (8), SYSDATETIME (), 5), 5) [MM-YY]<br>SELECT SUBSTRING (CONVERT (VARCHAR (8), SYSDATETIME (), 5), 4, 5) [MM-YY] | 06-11 |
| MM-YYYY | SELECT RIGHT (CONVERT (VARCHAR (10), SYSDATETIME (), 105), 7) [MM-YYYY] | 06-2011 |
| M-YY | 2) AS [M-YY] (2)) + '-'+ RIGHT (CAST (YEAR (SYSDATETIME ()) AS VARCHAR (4) | 6-11 |
| M-YYYY | [M-YYYY] AS VARCHAR (2)) + '-'+ CAST ( (SYSDATETIME ()) AS VARCHAR (4) | 6-2011 |
| MM-DD | SELECT CONVERT (VARCHAR (5), SYSDATETIME (), 10) AS [MM-DD] | 06-08 |
| DD-MM | SELECT  (VARCHAR (5), SYSDATETIME (), 5) [DD-MM] | 08-06 |
| | AS VARCHAR (2)) + '-'+ CAST ( (SYSDATETIME ()) AS VARCHAR (2) AS [MD] | 6-8 |
| DM | AS VARCHAR (2)) + '-'+ CAST (MONTH (SYSDATETIME ()) AS VARCHAR (2) AS [DM] | 8-6 |

| | SQL | |
|---|---|---|
| M / D / YYYY | YYAR (SYSDATETIME ()) AS VARCHAR (2)) + '/'+ CAST (DAY (SYSDATETIME ()) AS VARCHAR (2) + '/'+ CAST (YEAR (SYSDATETIME ()) AS VARCHAR 4)) AS [M / D / YYYY] | 2011/06/8 |
| M / D / YY | SELECT VARCHAR (2)) + '/'+ CAST (DAY (SYSDATETIME ()) AS VARCHAR (2)) + '/'+ RIGHT (CAST (YEAR (SYSDATETIME ()) AS VARCHAR (4)), 2) AS [M / D / YY] | 2011/06/08 |
| D / M / YYYY | SELECT VARCHAR (2)) + '/'+ CAST (YEAR (SYSDATETIME ()) AS VARCHAR (2)) + '/'+ CAST ( (SYSDATETIME ()) AS VARCHAR 4)) AS [D / M / YYYY] | 8/6/2011 |
| D / M / YY | SELECT CAST (DAY (SYSDATETIME ()) AS VARCHAR (2) + '/'+ CAST (MONTH (SYSDATETIME ()) AS VARCHAR (2)) + '/'+ RIGHT (CAST (YEAR (SYSDATETIME VARCHAR (4)), 2) AS [D / M / YY] | 2011/08/06 |
| YYYY / M / D | AS VARCHAR (2)) + '/'+ CAST (DAY (SYSDATETIME ()) AS VARCHAR (4)) + '/'+ CAST ( (SYSDATETIME ()) AS VARCHAR 2)) AS [YYYY / M / D] | 2011/6/8 |
| YY / M / D | + VARCHAR (2)) + '/'+ CAST (DAY (SYSDATETIME (2) )) AS VARCHAR (2)) AS [YY / M / D] | 11/6/8 |
| MM / YY | (CONVERT (VARCHAR (8), SYSDATETIME (), 3), 5) [MM / YY] | 06/11 |
| MM / YYYY | SELECT RIGHT (CONVERT (VARCHAR (10), SYSDATETIME (), 103), 7) [MM / YYYY] | 06/2011 |
| M / YY | 2) AS [M / YY] (2)) + '/'+ RIGHT (CAST (YEAR (SYSDATETIME ()) AS VARCHAR (4) | 6/11 |
| M / YYYY | [M / YYYY] AS VARCHAR (2)) + '/'+ CAST ( (SYSDATETIME ()) AS VARCHAR (4) | 6/2011 |
| YY / MM | SELECT  (VARCHAR (5), SYSDATETIME (), 11) [YY / MM] | 11/06 |
| YYYY / MM | SELECT CONVERT (VARCHAR (7), SYSDATETIME (), 111) AS [YYYY / MM] | 2011/06 |
| YY / M | 2) + '/'+ CAST (MONTH (SYSDATETIME ()) AS VARCHAR (2)) AS [YY / M] | 11/6 |
| YYYY / M | AS VARCHAR (2)) AS [YYYY / M] + '/'+ CAST (MONTH (SYSDATETIME ()) AS VARCHAR (2) | 2011/6 |

| | SQL | |
|---|---|---|
| MM / DD | SELECT CONVERT (VARCHAR (5), SYSDATETIME (), 1) AS [MM / DD] | 06/08 |
| DD / MM | SELECT CONVERT (VARCHAR (5), SYSDATETIME (), 3) AS [DD / MM] | 08/06 |
| M / D | [M / D] AS VARCHAR (2)) + '/'+ CAST ( (SYSDATETIME ()) AS VARCHAR | 6/8 |
| D / M | [D / M] AS VARCHAR (2)) + '/'+ CAST ( (SYSDATETIME ()) AS VARCHAR | 8/6 |
| MM.DD.YYYY | SELECT REPLACE (CONVERT (VARCHAR (10), SYSDATETIME (), 101), '/', '.') AS [MM.DD.YYYY] | 06.08.2011 |
| MM.DD.YY | SELECT REPLACE (CONVERT (VARCHAR (8), SYSDATETIME (), 1), '/', '.') AS [MM.DD.YY] | 06.08.11 |
| MDYYYY | SELECT CAST (MONTH (SYSDATETIME ()) AS VARCHAR (2)) + '.' + CAST (DAY (SYSDATETIME ()) AS VARCHAR (2)) + '.' + CAST (YEAR (SYSDATETIME ()) AS VARCHAR (4)) AS [MDYYYY] | 6.8.2011 |
| MDYY | SELECT CAST (MONTH (SYSDATETIME ()) AS VARCHAR (2)) + '.' + CAST (DAY (SYSDATETIME ()) AS VARCHAR (2)) + '.' + RIGHT (CAST (YEAR (SYSDATETIME ()) AS VARCHAR (4)), 2) AS [MDYY] | 6.8.11 |
| DD.MM.YYYY | SELECT CONVERT (VARCHAR (10), SYSDATETIME (), 104) AS [DD.MM.YYYY] | 08.06.2011 |
| DD.MM.YY | SELECT CONVERT (VARCHAR (10), SYSDATETIME (), 4) AS [DD.MM.YY] | 08.06.11 |
| DMYYYY | SELECT CAST (DAY (SYSDATETIME ()) AS VARCHAR (2)) + '.' + CAST (MONTH (SYSDATETIME ()) AS VARCHAR (2)) + '.' + CAST (YEAR (SYSDATETIME ()) AS VARCHAR (4)) AS [DMYYYY] | 8.6.2011 |
| DMYY | SELECT CAST (DAY (SYSDATETIME ()) AS VARCHAR (2)) + '.' + CAST (MONTH (SYSDATETIME ()) AS VARCHAR (2)) + '.' + RIGHT (CAST (YEAR (SYSDATETIME ()) AS VARCHAR (4)), 2) AS [DMYY] | 8.6.11 |
| YYYY.MD | SELECT CAST (YEAR (SYSDATETIME ()) AS VARCHAR (4)) + '.' + CAST (MONTH (SYSDATETIME ()) AS VARCHAR (2)) + '.' + CAST (DAY (SYSDATETIME ()) AS VARCHAR (2)) AS [YYYY.MD] | 2011.6.8 |

| | SQL | |
|---|---|---|
| YY.MD | SELECT RIGHT (CAST (YEAR (SYSDATETIME ()) AS VARCHAR (4)), 2) + '.' + CAST (MONTH (SYSDATETIME ()) AS VARCHAR (2)) + '.' + CAST (DAY (SYSDATETIME ()) AS VARCHAR (2)) AS [YY.MD] | 11.6.8 |
| MM.YYYY | SELECT RIGHT (CONVERT (VARCHAR (10), SYSDATETIME (), 104), 7) AS [MM.YYYY] | 06.2011 |
| MM.YY | SELECT RIGHT (CONVERT (VARCHAR (8), SYSDATETIME (), 4), 5) AS [MM.YY] | 06.11 |
| M.YYYY | SELECT CAST (MONTH (SYSDATETIME ()) AS VARCHAR (2)) + '.' + CAST (YEAR (SYSDATETIME ()) AS VARCHAR (4)) AS [M.YYYY] | 6.2011 |
| M.YY | SELECT CAST (MONTH (SYSDATETIME ()) AS VARCHAR (2)) + '.' + RIGHT (CAST (YEAR (SYSDATETIME ()) AS VARCHAR (4)), 2) AS [M.YY] | 6.11 |
| YYYY.MM | SELECT CONVERT (VARCHAR (7), SYSDATETIME (), 102) AS [YYYY.MM] | 2011.06 |
| YY.MM | SELECT CONVERT (VARCHAR (5), SYSDATETIME (), 2) AS [YY.MM] | 11.06 |
| YYYY.M | SELECT CAST (YEAR (SYSDATETIME ()) AS VARCHAR (4)) + '.' + CAST ( (SYSDATETIME ()) AS VARCHAR (2)) [YYYY.M] | 2011.6 |
| YY.M | SELECT RIGHT (CAST (YEAR (SYSDATETIME ()) AS VARCHAR (4)), 2) + '.' + CAST ( (SYSDATETIME ()) AS VARCHAR (2)) [YY.M] | 11.6 |
| MM.DD | SELECT RIGHT (CONVERT (VARCHAR (8), SYSDATETIME (), 2), 5) AS [MM.DD] | 06.08 |
| DD.MM | SELECT CONVERT (VARCHAR (5), SYSDATETIME (), 4) AS [DD.MM] | 08.06 |
| MMDDYYYY | SELECT REPLACE (CONVERT (VARCHAR (10), SYSDATETIME (), 101), '/', '') AS [MMDDYYYY] | 06082011 |
| MMDDYY | SELECT REPLACE (CONVERT (VARCHAR (8), SYSDATETIME (), 1), '/', '') [MMDDYY] | 060811 |
| DDMMYYYY | SELECT REPLACE (CONVERT (VARCHAR (10), SYSDATETIME (), 103), '/', '') AS [DDMMYYYY] | 08062011 |

| | SQL | |
|---|---|---|
| DDMMYY | SELECT REPLACE (CONVERT (VARCHAR (8), SYSDATETIME (), 3), '/', '') [DDMMYY] | 080611 |
| MMYYYY | ( ( (VARCHAR (10), SYSDATETIME (), 103), '/', ''), 6) AS [MMYYYY] | 062011 |
| MMYY | ( ( (VARCHAR (8), SYSDATETIME (), 3), '/', ''), 4) AS [MMYY] | 0611 |
| YYYYMM | (VARCHAR (6), SYSDATETIME (), 112) [YYYYMM] | 201106 |
| YYMM | SELECT CONVERT (VARCHAR (4), SYSDATETIME (), 12) [YYMM] | 1106 |
| DD, YYYY | 2) + ','+ DATENAME (YEAR, SYSDATETIME ()) AS [ DD, YYYY]  DATENAME (MONTH, SYSDATETIME ()) + ' +'(DAY, SYSDATETIME | 2011  6  8 |
| YYYY | SELECT [MON YYYY] (MONTH, SYSDATETIME ()), 3) + ''+ DATENAME (YEAR, SYSDATETIME ()) | 2011  6 |
| YYYY | SELECT DATENAME (MONTH, SYSDATETIME ()) + ''+ DATENAME (YEAR, SYSDATETIME ()) AS [ YYYY] | 2011  6 |
| | [DD Month] (DD ), DATENAME (MONTH, SYSDATETIME ()) ( '0'+ DATENAME (DAY, SYSDATETIME ()), 2) + ' | 6  8 |
| DD | DATENAME (MONTH, SYSDATETIME ()) + ''+ RIGHT ( '0'+ DATENAME (DAY, SYSDATETIME ()), 2) [ DD] | 6  8 |
| DD  YY | CAST (YEAR (SYSDATETIME ()) AS VARCHAR (4)), 2) ''+ DATENAME (MM, SYSDATETIME ()) + [DD  YY] | 6  11  08 |
| DD  YYYY | [DD Month YYYY] + DATENAME (YEAR, SYSDATETIME ()) + DATENAME (MONTH, SYSDATETIME ()) + '+ DATENAME (YEAR, SYSDATETIME () | 2011  6  8 |
| Mon-YY | SELECT REPLACE ( (CONVERT (VARCHAR (9), SYSDATETIME (), 6), 6), '', '-') AS [Mon-YY] | Jun-08 |
| -  YYYY | SELECT REPLACE ( (CONVERT (VARCHAR (11), SYSDATETIME (), 106), 8), '', '-') AS [Mon-YYYY] | 2011  6 |
| DD-Mon-YY | [DD-Mon-YY]  SELECT REPLACE (CONVERT (VARCHAR (9), SYSDATETIME (), 6), '', | 08-Jun-11 |
| DD-Mon-YYYY | SELECT [DD-Mon-YYYY]  SELECT REPLACE (CONVERT (VARCHAR (11), SYSDATETIME (), 106), '', | 2011  6  8 |

: https://riptutorial.com/ko/sql-server/topic/1471/

# 48:

| | |
|---|---|
| @FromDate | . |
| @ToDate | . |

Dates . http://support.microsoft.com/default.aspx

## Examples

### CTE

CTE .

```
Declare  @FromDate    Date = '2014-04-21',
         @ToDate      Date = '2014-05-02'

;With DateCte (Date) As
(
    Select  @FromDate Union All
    Select  DateAdd(Day, 1, Date)
    From    DateCte
    Where   Date < @ToDate
)
Select  Date
From    DateCte
Option  (MaxRecursion 0)
```

`MaxRecursion` 100 (100) `Option (MaxRecursion N)` `N` `MaxRecursion` . `0` `MaxRecursion` .

.

```
Declare   @FromDate   Date = '2014-04-21',
          @ToDate     Date = '2014-05-02'

;With
   E1(N) As (Select 1 From (Values (1), (1), (1), (1), (1), (1), (1), (1), (1), (1)) DT(N)),
   E2(N) As (Select 1 From E1 A Cross Join E1 B),
   E4(N) As (Select 1 From E2 A Cross Join E2 B),
   E6(N) As (Select 1 From E4 A Cross Join E2 B),
   Tally(N) As
   (
        Select    Row_Number() Over (Order By (Select Null))
        From    E6
   )
Select   DateAdd(Day, N - 1, @FromDate) Date
From     Tally
Where    N <= DateDiff(Day, @FromDate, @ToDate) + 1
```

: https://riptutorial.com/ko/sql-server/topic/3232/--

# 49:

## Examples

```
CREATE TABLE #TEST
(
Id INT,
Name VARCHAR(10)
)

Insert Into #Test
select 1,'A'
Union All
Select 1,'B'
union all
Select 1,'C'
union all
Select 2,'D'
```

. Id Column 3 .

```
Id   Name
1    A
1    B
1    C
2    D
```

.

```
Select Top (1)  Id,Name From
#test
Order By Id ;
```

***: ( )***

```
Id   Name
1    B
```

Ties Option .

```
Select Top (1) With Ties Id,Name
 From
#test
Order By Id
```

***:***

```
Id   Name
1    A
1    B
1    C
```

SQL Server  Column by Order  .  ..

```
Select Top (1) With Ties Id,Name
 From
#test
Order By Id ,Name
```

:

```
Id   Name
1    A
```

Ties Option  SQL Server  Tied .

: https://riptutorial.com/ko/sql-server/topic/2546/--

# 50:

## Examples

**Union** . . . .

1. .

2. .

:

Marksheet1, Marksheet2 Marksheet3 . Marksheet3 Marksheet2 Marksheet2 .

**1** : **1**

| SubjectCode | SubjectName | MarksObtained |
|---|---|---|
| 101 | Physics | 87 |
| 102 | Chemistry | 75 |
| 103 | Maths | 85 |
| 104 | English | 89 |
| 105 | Computer | 95 |

**2** : **2**

| CourseCode | CourseName | MarksObtained |
|---|---|---|
| 201 | PhysicsII | 82 |
| 202 | ChemistryII | 86 |
| 203 | MathsII | 95 |
| 204 | EnglishII | 70 |
| 205 | ComputerII | 86 |

**3** : **3**

| SubjectCode | SubjectName | MarksObtained |
|---|---|---|
| 201 | PhysicsII | 82 |
| 202 | ChemistryII | 86 |
| 203 | MathsII | 95 |
| 204 | EnglishII | 70 |
| 205 | ComputerII | 86 |

**Marksheet1 Marksheet2**

```
SELECT SubjectCode, SubjectName, MarksObtained
FROM Marksheet1
UNION
SELECT CourseCode, CourseName, MarksObtained
FROM Marksheet2
```

: Union Marksheet1 Marksheet2 .

```
SELECT SubjectCode, SubjectName, MarksObtained
FROM Marksheet1
UNION
SELECT CourseCode, CourseName, MarksObtained
FROM Marksheet2
UNION
SELECT SubjectCode, SubjectName, MarksObtained
FROM Marksheet3
```

| | SubjectCode | SubjectName | MarksObtained |
|---|---|---|---|
| 1 | 101 | Physics | 87 |
| 2 | 102 | Chemistry | 75 |
| 3 | 103 | Maths | 85 |
| 4 | 104 | English | 89 |
| 5 | 105 | Computer | 95 |
| 6 | 201 | PhysicsII | 82 |
| 7 | 202 | ChemistryII | 86 |
| 8 | 203 | MathsII | 95 |
| 9 | 204 | EnglishII | 70 |
| 10 | 205 | ComputerII | 86 |

```
SELECT SubjectCode, SubjectName, MarksObtained
FROM Marksheet1
UNION ALL
SELECT CourseCode, CourseName, MarksObtained
FROM Marksheet2
UNION ALL
SELECT SubjectCode, SubjectName, MarksObtained
FROM Marksheet3
```

| | SubjectCode | SubjectName | MarksObtained |
|---|---|---|---|
| 1 | 101 | Physics | 87 |
| 2 | 102 | Chemistry | 75 |
| 3 | 103 | Maths | 85 |
| 4 | 104 | English | 89 |
| 5 | 105 | Computer | 95 |
| 6 | 201 | PhysicsII | 82 |
| 7 | 202 | ChemistryII | 86 |
| 8 | 203 | MathsII | 95 |
| 9 | 204 | EnglishII | 70 |
| 10 | 205 | ComputerII | 86 |
| 11 | 201 | PhysicsII | 82 |
| 12 | 202 | ChemistryII | 86 |
| 13 | 203 | MathsII | 95 |
| 14 | 204 | EnglishII | 70 |
| 15 | 205 | ComputerII | 86 |

Marksheet3 UNION .

: https://riptutorial.com/ko/sql-server/topic/5590/-

# 51:

## Examples

SQL Server 2012

`. index values  NULL .`

:

1. `index` : , `values` . 1 .
2. `values` : ,

```
SELECT CHOOSE (1, 'apples', 'pears', 'oranges', 'bananas') AS chosen_result

chosen_result
-------------
apples
```

### IIF

SQL Server 2012

true  false  .

:

1. dtermine `boolean_expression`
2. `boolean_expression` true `true_value` .
3. `boolean_expression` false `false_value` .

```
SELECT IIF (42 > 23, 'I knew that!', 'That is not true.') AS iif_result

iif_result
------------
I knew that!
```

SQL Server 2012

`IIF CASE  .  .`

```
SELECT CASE WHEN 42 > 23 THEN 'I knew that!' ELSE 'That is not true.' END AS iif_result

iif_result
------------
I knew that!
```

: https://riptutorial.com/ko/sql-server/topic/10647/-

# 52:

## Examples

### IF

T-SQL IF..ELSE .

, `1 = 1` True `BEGIN..END` Print `'One is equal to One'`

```
IF ( 1 = 1)  --<-- Some Expression
 BEGIN
   PRINT 'One is equal to One'
 END
```

### IF

IF .

`IF` , `BEGIN...END` . First Third true .

```
IF (1 = 1)  --<-- Some Expression      --<-- This is true
BEGIN
    PRINT 'First IF is True'          --<-- this will be executed
END

IF (1 = 2)  --<-- Some Expression
BEGIN
    PRINT 'Second IF is True'
END

IF (3 = 3)  --<-- Some Expression        --<-- This true
BEGIN
    PRINT 'Thrid IF is True'             --<-- this will be executed
END
```

### IF..ELSE

`IF..ELSE IF BEGIN..END` True `BEGIN..END` .

`False ELSE BEGIN..END` `BEGIN..END` .

false Else `'First expression was not true'` .

```
IF ( 1 <> 1)  --<-- Some Expression
 BEGIN
     PRINT 'One is equal to One'
 END
ELSE
 BEGIN
```

```
     PRINT 'First expression was not true'
 END
```

## IF ... ELSE IF ...

IF...ELSE IF True IF ELSE ELSE IF .

IF ELSE IF True ELSE 'No other expression is true'

```
IF ( 1 = 1 + 1 )
    BEGIN
       PRINT 'First If Condition'
    END
ELSE IF (1 = 2)
    BEGIN
        PRINT 'Second If Else Block'
    END
ELSE IF (1 = 3)
    BEGIN
        PRINT 'Third If Else Block'
    END
ELSE
    BEGIN
        PRINT 'No other expression is true'  --<-- Only this statement will be printed
    END
```

## IF ... ELSE

. IF...ELSE IF .

. true . true .

```
IF (1 = 1 + 1)
BEGIN
    PRINT 'First If Condition'
END
ELSE IF (1 = 2)
BEGIN
    PRINT 'Second If Else Block'
END
ELSE IF (1 = 3)
BEGIN
    PRINT 'Third If Else Block'
END
ELSE IF (1 = 1)      --<-- This is True
BEGIN
    PRINT 'Last Else Block'  --<-- Only this statement will be printed
END
```

: https://riptutorial.com/ko/sql-server/topic/5186/-

# 53: SQL Server (2000 - 2016)

2004 SQL Server . 2000 SQL Server 2016 . , , , , SQL . . , . . 2000 20 .

## Examples

### SQL Server 2000 - 2016

**SQL Server 2000 .**

1. (BIGINT, SQL_VARIANT, TABLE).
2. DDL .
3. .
4. XML
5. .
6. ( ).

**2005 .**

1. "WITH TIES" TOP .
2. INSERTED DELETED DML (Data Manipulation Commands) OUTPUT
3. PIVOT UNPIVOT .
4. TRY / CATCH
5. 
6. (CTE)
7. ( , , .NET )
8. Service Broker ( )
9. ( )
10. SMTP
11. HTTP ( T-SQL )
12. (MARS). .
13. SQL Server Integration Services ( ETL (, ) )
14. Analysis Services Reporting Services .
15. . PARTITION SCHEME PARTITION FUNCTION .

**2008 .**

1. DATE TIME
2. - SYSUTCDATETIME () SYSDATETIMEOFFSET ()
3. - .
4. ( 2GB )
5. 
6. INSERT, UPDATE DELETE MERGE
7. HierarchyID
8. - .
9. GROUPING SETS - GROUP BY .

10. FILESTREAM

**2008 R2 .**

1. PowerPivot - .
2. 3.0
3. 
4. StreamInsight
5. 
6. SharePoint
7. DACPAC ( )
8. SQL Server 2008

**2012 .**

1. - I / O .
2. - "OFFSET" "FETCH" .
3. - .
4. AlwaysOn
5. Windows Server
6. 
7. 
8. PowerView
9. SQL Azure
10. (SSAS)
11. DQS
12. - 2008 FILESTREAM .
13. THROW
14. SQL Server Management Studio a. SQL Server 2012 . .
    . IntelliSense - .

**2014 .**

1. OLTP - 20 .
2. AlwaysOn
3. 
4. 
5. (Updatable Column )
6. ( SELECT INTO )
7. Office 365 Power BI
8. 
9. 

**2016 .**

1. - .
2. 
3. SQL Server PolyBase
4. JSON
5. 

6. AlwaysOn
7. OLTP
8. TempDB
9.
10.
11.

SQL Server 2016 T-SQL

1. PARTITION TRUNCATE TABLE
2. DROP IF
3. STRING_SPLIT  STRING_ESCAPE
4. ALTER TABLE  WITH (ONLINE = ON | OFF)  .
5. DBCC CHECKDB, DBCC CHECKTABLE  DBCC CHECKFILEGROUP  MAXDOP
6. ALTER DATABASE SET AUTOGROW_SINGLE_FILE
7. ALTER  AUTOGROW_ALL_FILES
8. COMPRESS  DECOMPRESS
9. FORMATMESSAGE
10. 2016  SERVERPROPERTY 8

. InstanceDefaultDataPath

. InstanceDefaultLogPath

. ProductBuild

. ProductBuildType

. ProductMajorVersion

. ProductMinorVersion

. ProductUpdateLevel

h. ProductUpdateReference

SQL Server  (2000 - 2016)  : https://riptutorial.com/ko/sql-server/topic/10129/--sql-server---2000---2016-

# 54:

## Examples

### BULK INSERT

WITH .

```
BULK INSERT People
FROM 'f:\orders\people.csv'
WITH  (  CODEPAGE = '65001',
         FIELDTERMINATOR =',',
         ROWTERMINATOR ='\n'
      );
```

CODEPAGE UTF-8 TERMINATORS .

BULK INSERT SQL Server .

```
BULK INSERT People
FROM 'f:\orders\people.csv'
```

BULK INSERT C C .

### OPENROWSET (BULK)

OPENROWSET (BULK) .

```
INSERT INTO myTable(content)
   SELECT BulkColumn
          FROM OPENROWSET(BULK N'C:\Text1.txt', SINGLE_BLOB) AS Document;
```

SINGLE_BLOB .

### OPENROWSET (BULK)

Yu FORMATFILE .

```
INSERT INTO mytable
SELECT a.*
FROM OPENROWSET(BULK 'c:\test\values.txt',
   FORMATFILE = 'c:\test\values.fmt') AS a;
```

format_file.fmt values.txt .

```
9.0
2
1  SQLCHAR  0  10 "\t"        1  ID               SQL_Latin1_General_Cp437_BIN
```

```
2  SQLCHAR  0  40 "\r\n"      2  Description       SQL_Latin1_General_Cp437_BIN
```

## OPENROWSET (BULK)  json

OPENROWSET .

OPENROWSET (BULK)  JSON  JSON  OPENJSON  BulkColumn .

```
SELECT book.*
 FROM OPENROWSET (BULK 'C:\JSON\Books\books.json', SINGLE_CLOB) as j
 CROSS APPLY OPENJSON(BulkColumn)
       WITH( id nvarchar(100), name nvarchar(100), price float,
             pages int, author nvarchar(100)) AS book
```

: https://riptutorial.com/ko/sql-server/topic/7330/--

# 55:

, ( ) SQL Server .

## Examples

**Integer** .

- 
- 
- 
- int
- bigint

. .

| | | |
|---|---|---|
| | 0 1 | 1 ** |
| | 0 ~ 255 | 1 |
| | -2 ^ 15 (-32,768) ~ 2 ^ 15-1 (32,767) | 2 |
| int | -2 ^ 31 (-2,147,483,648) ~ 2 ^ 31-1 (2,147,483,647) | 4 |
| bigint | -2 ^ 63 (-9,223,372,036,854,775,808) ~ 2 ^ 63-1 (9,223,372,036,854,775,807) | 8 |

- 
- 
- 
- 

. . .

.

| | | |
|---|---|---|
| [(p [, s])] | [(p [, s])] | -10 ^ 38 + 1 ~ 10 ^ 38 - 1 |

*10* Precision [p] Scale [s] .

. , 1 999 3 ( 100 ). 18.

Scale . 0.00 999.99 5 (5 ) 2 ( 2 ) . . 0.

*10* .

| | |
|---|---|
| 1 - 9 | 5 |
| 10-19 | 9 |
| 20-28 | 13 |
| 29-38 | 17 |

. 4 - . 2 . 2 .

| | | |
|---|---|---|
| | -922,337,203,685,477.5808 ~ 922,337,203,685,477.5807 | 8 |
| | -214,748.3648 214,748.3647 | 4 |

- [( **n** )]
-

. . .

| | | |
|---|---|---|
| | -1.79E + 308 ~ -2.23E-308, 0 2.23E-308 ~ 1.79E + 308 | **n** . |
| | -3.40E + 38 - -1.18E - 38, 0 1.18E - 38 - 3.40E + 38 | 4 |

**n** . float 53 . *float (24)* .

| **n** | | |
|---|---|---|
| 1-24 | 7 | 4 |
| 25-53 | 15 | 8 |

SQL Server .

-
- smalldatetime

SQL Server 2012 SQL Server .

-
- datetimeoffset
- datetime2
-

-
- varchar
-

- nchar
- nvarchar
- ntext

-
- varbinary
-

-
-
- hierarchyid
- uniqueidentifier
- sql_variant
- XML
-
-

: https://riptutorial.com/ko/sql-server/topic/5260/-

# 56:

## Examples

### TRY PARSE

SQL Server 2012

( ) .

. NULL .

: TRY_PARSE (string_value AS data_type [USING culture])

String_value - NVARCHAR (4000) .
Data_type - .
- . culture 'Fr-FR' . PARSE .

```
DECLARE @fakeDate AS varchar(10);
DECLARE @realDate AS VARCHAR(10);
SET @fakeDate = 'iamnotadate';
SET @realDate = '13/09/2015';

SELECT TRY_PARSE(@fakeDate AS DATE); --NULL  as the parsing fails

SELECT TRY_PARSE(@realDate AS DATE); -- NULL due to type mismatch

SELECT TRY_PARSE(@realDate AS DATE USING 'Fr-FR'); -- 2015-09-13
```

.

SQL Server 2012

NULL . / . .

: TRY_CONVERT ( [()], [, ])

TRY_CONVERT () . null .

Data_type - . .
Expression - .
- . "May, 18 2013" 111 .

```
DECLARE @sampletext AS VARCHAR(10);
SET @sampletext = '123456';
DECLARE @ realDate AS VARCHAR(10);
SET @realDate = '13/09/2015';
SELECT TRY_CONVERT(INT, @sampletext); -- 123456
SELECT TRY_CONVERT(DATETIME, @sampletext); -- NULL
```

```
SELECT TRY_CONVERT(DATETIME, @realDate, 111); -- Sep, 13 2015
```

.

SQL Server 2012

NULL . / . .

: TRY_CAST (expression AS data_type [(length)])

TRY_CAST () . null .

Expression - .
Data_type - .
- .

```
DECLARE @sampletext AS VARCHAR(10);
SET @sampletext = '123456';

SELECT TRY_CAST(@sampletext AS INT); -- 123456
SELECT TRY_CAST(@sampletext AS DATE); -- NULL
```

Cast () .

CAST ([Expression] AS )

3 . .

```
DECLARE @A varchar(2)
DECLARE @B varchar(2)

set @A='25a'
set @B='15'

Select CAST(@A as int) + CAST(@B as int)  as Result
--'25a' is casted to 25 (string to int)
--'15' is casted to 15 (string to int)

--Result
 --40

DECLARE @C varchar(2)  = 'a'

select CAST(@C as int) as Result
--Result
 --Conversion failed when converting the varchar value 'a' to data type int.
```

.

datetime varchar . Convert . CONVERT () / .

(data_type (), , )

datetime smalldatetime . 100 (yyyy) .

```
select convert(varchar(20),GETDATE(),108)

13:27:16
```

: https://riptutorial.com/ko/sql-server/topic/5034/--

# 57:

- TO *backup_device [, ... n]* WITH *with_options [, ... o]*
- *backup_device [, ... n] with with_options [, ... o]* .

| | |
|---|---|
| *backup_device* | {DISK TAPE} () . (,). |
| *with_options* | . . |

## Examples

*'Users' 'D : \ DB_Backup'* . .

```
BACKUP DATABASE Users TO DISK = 'D:\DB_Backup'
```

*'Users' 'D : \ DB_Backup'* .

```
RESTORE DATABASE Users FROM DISK = 'D:\DB_Backup'
```

### REPLACE

.

3154 : .

WITH REPLACE .

```
RESTORE DATABASE WWIDW
FROM DISK = 'C:\Backup\WideWorldImportersDW-Full.bak'
WITH REPLACE
```

.

3156, 16, 3, 1 'WWI_Primary' 'D : \ Data \ WideWorldImportersDW.mdf' . WITH MOVE .

. .

```
RESTORE DATABASE WWIDW
FROM DISK = 'C:\Backup\WideWorldImportersDW-Full.bak'
WITH REPLACE,
MOVE 'WWI_Primary' to 'C:\Data\WideWorldImportersDW.mdf',
MOVE 'WWI_UserData' to 'C:\Data\WideWorldImportersDW_UserData.ndf',
MOVE 'WWI_Log' to 'C:\Data\WideWorldImportersDW.ldf',
MOVE 'WWIDW_InMemory_Data_1' to 'C:\Data\WideWorldImportersDW_InMemory_Data_1'
```

.

: https://riptutorial.com/ko/sql-server/topic/5826/---

# 58:

:

```
{GRANT| REVOKE | DENY} {PERMISSION_NAME} [ON {SECURABLE}] TO {PRINCIPAL};
```

- { | REVOKE | DENY} -
  - : " "
  - : " "
  - Deny : " (," DENY SELECT " SELECT )
- PERMISSION_NAME - . . , GRANT SELECT GRANT SELECT .
- SECURABLE - . .GRANT SELECT TO [aUser]; ; " SELECT GRANT " .
- PRINCIPAL - . ( ) ( ) .

## Examples

```
GRANT SELECT ON [dbo].[someTable] TO [aUser];

REVOKE SELECT ON [dbo].[someTable] TO [aUser];
--REVOKE SELECT [dbo].[someTable] FROM [aUser]; is equivalent

DENY SELECT ON [dbo].[someTable] TO [aUser];
```

### CREATE USER

```
--implicitly map this user to a login of the same name as the user
CREATE USER [aUser];

--explicitly mapping what login the user should be associated with
CREATE USER [aUser] FOR LOGIN [aUser];
```

### CREATE ROLE

```
CREATE ROLE [myRole];
```

```
-- SQL 2005+
exec sp_addrolemember @rolename = 'myRole', @membername = 'aUser';
exec sp_droprolemember @rolename = 'myRole', @membername = 'aUser';

-- SQL 2008+
ALTER ROLE [myRole] ADD MEMBER [aUser];
ALTER ROLE [myRole] DROP MEMBER [aUser];
```

: . , . / . , / .

: https://riptutorial.com/ko/sql-server/topic/6788/--

# 59:

SQL Server .

. .

. .

, ( )

Enterprise Developer .

## Examples

SQL Server ( ) . .

```
CREATE DATABASE MyDatabase_morning -- name of the snapshot
ON (
     NAME=MyDatabase_data, -- logical name of the data file of the source database
     FILENAME='C:\SnapShots\MySnapshot_Data.ss' -- snapshot file;
)
AS SNAPSHOT OF MyDatabase; -- name of source database
```

.

```
CREATE DATABASE MyMultiFileDBSnapshot ON
    (NAME=MyMultiFileDb_ft, FILENAME='C:\SnapShots\MyMultiFileDb_ft.ss'),
    (NAME=MyMultiFileDb_sys, FILENAME='C:\SnapShots\MyMultiFileDb_sys.ss'),
    (NAME=MyMultiFileDb_data, FILENAME='C:\SnapShots\MyMultiFileDb_data.ss'),
    (NAME=MyMultiFileDb_indx, FILENAME='C:\SnapShots\MyMultiFileDb_indx.ss')
AS SNAPSHOT OF MultiFileDb;
```

.

```
RESTORE DATABASE MYDATABASE FROM DATABASE_SNAPSHOT='MyDatabase_morning';
```

**:** !

DELETE DATABASE .

```
DROP DATABASE Mydatabase_morning
```

.

: https://riptutorial.com/ko/sql-server/topic/677/--

# 60:

SQL Server .

ISO `INFORMATION_SCHEMA` SQL Server `sys` .

## Examples

.

```
USE YourDatabaseName
SELECT COUNT(*) from INFORMATION_SCHEMA.TABLES
WHERE TABLE_TYPE = 'BASE TABLE'
```

SQL Server 2008 . .

```
SELECT COUNT(*) FROM sys.tables
```

.

SQL Server 2005

```
SELECT *
FROM INFORMATION_SCHEMA.ROUTINES
WHERE ROUTINE_TYPE = 'PROCEDURE'
```

`ROUTINE_NAME` , `ROUTINE_SCHEMA` `ROUTINE_DEFINITION` .

SQL Server 2005

```
SELECT *
FROM sys.objects
WHERE type = 'P'
```

SQL Server 2005

```
SELECT *
FROM sys.procedures
```

`is_auto_executed` , `is_execution_replicated` , `is_repl_serializable` `skips_repl_constraints`
sys.objects .

SQL Server 2005

```
SELECT *
FROM sysobjects
WHERE type = 'P'
```

.

'SearchTerm' .

SQL Server 2005

```
SELECT o.name
FROM syscomments c
INNER JOIN sysobjects o
    ON c.id=o.id
WHERE o.xtype = 'P'
    AND c.TEXT LIKE '%SearchTerm%'
```

SQL Server 2005

```
SELECT p.name
FROM sys.sql_modules AS m
INNER JOIN sys.procedures AS p
    ON m.object_id = p.object_id
WHERE definition LIKE '%SearchTerm%'
```

**1 :** SQL Server 2000 (12 )

```
SELECT * FROM dbo.sysdatabases
```

**2 :** ( : , , ).

: SQL Server 2005 .

```
SELECT * FROM sys.databases
```

**3 :** sp_MSForEachDB .

```
EXEC sp_MSForEachDB 'SELECT ''?'' AS DatabaseName'
```

**4 :** SP , , .

```
EXEC sp_helpdb
```

**5** ,

```
EXEC sp_databases
```

```
SELECT  d.name AS 'Database',
        d.database_id,
        SF.fileid,
        SF.name AS 'LogicalFileName',
        CASE SF.status & 0x100000
            WHEN 1048576 THEN 'Percentage'
            WHEN 0 THEN 'MB'
        END AS 'FileGrowthOption',
        Growth AS GrowthUnit,
        ROUND(((CAST(Size AS FLOAT)*8)/1024)/1024,2) [SizeGB], -- Convert 8k pages to GB
        Maxsize,
```

```
        filename AS PhysicalFileName

FROM    Master.SYS.SYSALTFILES SF
Join    Master.SYS.Databases d on sf.fileid = d.database_id

Order by d.name
```

.

```
select * from sys.databases WHERE name = 'MyDatabaseName';
```

```
SELECT
    s.name + '.' + t.NAME AS TableName,
    SUM(a.used_pages)*8 AS 'TableSizeKB'  --a page in SQL Server is 8kb
FROM sys.tables t
    JOIN sys.schemas s on t.schema_id = s.schema_id
    LEFT JOIN sys.indexes i ON t.OBJECT_ID = i.object_id
    LEFT JOIN sys.partitions p ON i.object_id = p.OBJECT_ID AND i.index_id = p.index_id
    LEFT JOIN sys.allocation_units a ON p.partition_id = a.container_id
GROUP BY
    s.name, t.name
ORDER BY
    --Either sort by name:
    s.name + '.' + t.NAME
    --Or sort largest to smallest:
    --SUM(a.used_pages) desc
```

## Windows

( Windows ) .

```
xp_logininfo 'DOMAIN\user'
```

COLUMNS TABLES . () .

```
SELECT
    c.name AS ColName,
    t.name AS TableName
FROM
    sys.columns c
    JOIN sys.tables t ON c.object_id = t.object_id
WHERE
    c.name LIKE '%MyName%'
```

.

Enterprise .

sys.dm_db_persisted_sku_features .

```
SELECT * FROM sys.dm_db_persisted_sku_features
```

.

.

. . / .

```
DECLARE @SearchStr nvarchar(100)
SET @SearchStr = '## YOUR STRING HERE ##'


    -- Copyright © 2002 Narayana Vyas Kondreddi. All rights reserved.
    -- Purpose: To search all columns of all tables for a given search string
    -- Written by: Narayana Vyas Kondreddi
    -- Site: http://vyaskn.tripod.com
    -- Updated and tested by Tim Gaunt
    -- http://www.thesitedoctor.co.uk
    --
http://blogs.thesitedoctor.co.uk/tim/2010/02/19/Search+Every+Table+And+Field+In+A+SQL+Server+Database+U

    -- Tested on: SQL Server 7.0, SQL Server 2000, SQL Server 2005 and SQL Server 2010
    -- Date modified: 03rd March 2011 19:00 GMT
    CREATE TABLE #Results (ColumnName nvarchar(370), ColumnValue nvarchar(3630))

    SET NOCOUNT ON

    DECLARE @TableName nvarchar(256), @ColumnName nvarchar(128), @SearchStr2 nvarchar(110)
    SET  @TableName = ''
    SET @SearchStr2 = QUOTENAME('%' + @SearchStr + '%','''')

    WHILE @TableName IS NOT NULL

    BEGIN
        SET @ColumnName = ''
        SET @TableName =
        (
            SELECT MIN(QUOTENAME(TABLE_SCHEMA) + '.' + QUOTENAME(TABLE_NAME))
            FROM     INFORMATION_SCHEMA.TABLES
            WHERE         TABLE_TYPE = 'BASE TABLE'
                AND    QUOTENAME(TABLE_SCHEMA) + '.' + QUOTENAME(TABLE_NAME) > @TableName
                AND    OBJECTPROPERTY(
                        OBJECT_ID(
                            QUOTENAME(TABLE_SCHEMA) + '.' + QUOTENAME(TABLE_NAME)
                             ), 'IsMSShipped'
                               ) = 0
        )

        WHILE (@TableName IS NOT NULL) AND (@ColumnName IS NOT NULL)

        BEGIN
            SET @ColumnName =
            (
                SELECT MIN(QUOTENAME(COLUMN_NAME))
                FROM     INFORMATION_SCHEMA.COLUMNS
                WHERE         TABLE_SCHEMA    = PARSENAME(@TableName, 2)
                    AND    TABLE_NAME    = PARSENAME(@TableName, 1)
                    AND    DATA_TYPE IN ('char', 'varchar', 'nchar', 'nvarchar', 'int',
'decimal')
                    AND    QUOTENAME(COLUMN_NAME) > @ColumnName
            )
```

```
            IF @ColumnName IS NOT NULL

            BEGIN
                INSERT INTO #Results
                EXEC
                (
                    'SELECT ''' + @TableName + '.' + @ColumnName + ''', LEFT(' + @ColumnName +
', 3630) FROM ' + @TableName + ' (NOLOCK) ' +
                    ' WHERE ' + @ColumnName + ' LIKE ' + @SearchStr2
                )
            END
        END
    END

    SELECT ColumnName, ColumnValue FROM #Results

DROP TABLE #Results
- See more at: http://thesitedoctor.co.uk/blog/search-every-table-and-field-in-a-sql-server-
database-updated#sthash.bBEqfJVZ.dpuf
```

''

```
SELECT
    s.name AS [schema],
    t.object_id AS [table_object_id],
    t.name AS [table_name],
    c.column_id,
    c.name AS [column_name],
    i.name AS [index_name],
    i.type_desc AS [index_type]
FROM sys.schemas AS s
INNER JOIN sys.tables AS t
    ON s.schema_id = t.schema_id
INNER JOIN sys.columns AS c
    ON t.object_id = c.object_id
LEFT JOIN sys.index_columns AS ic
    ON c.object_id = ic.object_id and c.column_id = ic.column_id
LEFT JOIN sys.indexes AS i
    ON ic.object_id = i.object_id and ic.index_id = i.index_id
ORDER BY [schema], [table_name], c.column_id;
```

## SQL

```
USE msdb
Go


SELECT dbo.sysjobs.Name AS 'Job Name',
      'Job Enabled' = CASE dbo.sysjobs.Enabled
            WHEN 1 THEN 'Yes'
            WHEN 0 THEN 'No'
      END,
      'Frequency' = CASE dbo.sysschedules.freq_type
            WHEN 1 THEN 'Once'
            WHEN 4 THEN 'Daily'
            WHEN 8 THEN 'Weekly'
            WHEN 16 THEN 'Monthly'
```

```
            WHEN 32 THEN 'Monthly relative'
            WHEN 64 THEN 'When SQLServer Agent starts'
      END,
      'Start Date' = CASE active_start_date
            WHEN 0 THEN null
            ELSE
            substring(convert(varchar(15),active_start_date),1,4) + '/' +
            substring(convert(varchar(15),active_start_date),5,2) + '/' +
            substring(convert(varchar(15),active_start_date),7,2)
      END,
      'Start Time' = CASE len(active_start_time)
            WHEN 1 THEN cast('00:00:0' + right(active_start_time,2) as char(8))
            WHEN 2 THEN cast('00:00:' + right(active_start_time,2) as char(8))
            WHEN 3 THEN cast('00:0'
                        + Left(right(active_start_time,3),1)
                        +':' + right(active_start_time,2) as char (8))
            WHEN 4 THEN cast('00:'
                        + Left(right(active_start_time,4),2)
                        +':' + right(active_start_time,2) as char (8))
            WHEN 5 THEN cast('0'
                        + Left(right(active_start_time,5),1)
                        +':' + Left(right(active_start_time,4),2)
                        +':' + right(active_start_time,2) as char (8))
            WHEN 6 THEN cast(Left(right(active_start_time,6),2)
                        +':' + Left(right(active_start_time,4),2)
                        +':' + right(active_start_time,2) as char (8))
      END,

      CASE len(run_duration)
            WHEN 1 THEN cast('00:00:0'
                        + cast(run_duration as char) as char (8))
            WHEN 2 THEN cast('00:00:'
                        + cast(run_duration as char) as char (8))
            WHEN 3 THEN cast('00:0'
                        + Left(right(run_duration,3),1)
                        +':' + right(run_duration,2) as char (8))
            WHEN 4 THEN cast('00:'
                        + Left(right(run_duration,4),2)
                        +':' + right(run_duration,2) as char (8))
            WHEN 5 THEN cast('0'
                        + Left(right(run_duration,5),1)
                        +':' + Left(right(run_duration,4),2)
                        +':' + right(run_duration,2) as char (8))
            WHEN 6 THEN cast(Left(right(run_duration,6),2)
                        +':' + Left(right(run_duration,4),2)
                        +':' + right(run_duration,2) as char (8))
      END as 'Max Duration',
    CASE(dbo.sysschedules.freq_subday_interval)
            WHEN 0 THEN 'Once'
            ELSE cast('Every '
                        + right(dbo.sysschedules.freq_subday_interval,2)
                        + ' '
                        +     CASE(dbo.sysschedules.freq_subday_type)
                                    WHEN 1 THEN 'Once'
                                    WHEN 4 THEN 'Minutes'
                                    WHEN 8 THEN 'Hours'
                              END as char(16))
    END as 'Subday Frequency'
FROM dbo.sysjobs
LEFT OUTER JOIN dbo.sysjobschedules
ON dbo.sysjobs.job_id = dbo.sysjobschedules.job_id
```

```
INNER JOIN dbo.sysschedules ON dbo.sysjobschedules.schedule_id = dbo.sysschedules.schedule_id
LEFT OUTER JOIN (SELECT job_id, max(run_duration) AS run_duration
            FROM dbo.sysjobhistory
            GROUP BY job_id) Q1
ON dbo.sysjobs.job_id = Q1.job_id
WHERE Next_run_time = 0

UNION

SELECT dbo.sysjobs.Name AS 'Job Name',
      'Job Enabled' = CASE dbo.sysjobs.Enabled
            WHEN 1 THEN 'Yes'
            WHEN 0 THEN 'No'
      END,
      'Frequency' = CASE dbo.sysschedules.freq_type
            WHEN 1 THEN 'Once'
            WHEN 4 THEN 'Daily'
            WHEN 8 THEN 'Weekly'
            WHEN 16 THEN 'Monthly'
            WHEN 32 THEN 'Monthly relative'
            WHEN 64 THEN 'When SQLServer Agent starts'
      END,
      'Start Date' = CASE next_run_date
            WHEN 0 THEN null
            ELSE
            substring(convert(varchar(15),next_run_date),1,4) + '/' +
            substring(convert(varchar(15),next_run_date),5,2) + '/' +
            substring(convert(varchar(15),next_run_date),7,2)
      END,
      'Start Time' = CASE len(next_run_time)
            WHEN 1 THEN cast('00:00:0' + right(next_run_time,2) as char(8))
            WHEN 2 THEN cast('00:00:' + right(next_run_time,2) as char(8))
            WHEN 3 THEN cast('00:0'
                        + Left(right(next_run_time,3),1)
                        +':' + right(next_run_time,2) as char (8))
            WHEN 4 THEN cast('00:'
                        + Left(right(next_run_time,4),2)
                        +':' + right(next_run_time,2) as char (8))
            WHEN 5 THEN cast('0' + Left(right(next_run_time,5),1)
                        +':' + Left(right(next_run_time,4),2)
                        +':' + right(next_run_time,2) as char (8))
            WHEN 6 THEN cast(Left(right(next_run_time,6),2)
                        +':' + Left(right(next_run_time,4),2)
                        +':' + right(next_run_time,2) as char (8))
      END,

      CASE len(run_duration)
            WHEN 1 THEN cast('00:00:0'
                        + cast(run_duration as char) as char (8))
            WHEN 2 THEN cast('00:00:'
                        + cast(run_duration as char) as char (8))
            WHEN 3 THEN cast('00:0'
                        + Left(right(run_duration,3),1)
                        +':' + right(run_duration,2) as char (8))
            WHEN 4 THEN cast('00:'
                        + Left(right(run_duration,4),2)
                        +':' + right(run_duration,2) as char (8))
            WHEN 5 THEN cast('0'
                        + Left(right(run_duration,5),1)
                        +':' + Left(right(run_duration,4),2)
                        +':' + right(run_duration,2) as char (8))
```

```
            WHEN 6 THEN cast(Left(right(run_duration,6),2)
                        +':' + Left(right(run_duration,4),2)
                        +':' + right(run_duration,2) as char (8))
      END as 'Max Duration',
    CASE(dbo.sysschedules.freq_subday_interval)
            WHEN 0 THEN 'Once'
            ELSE cast('Every '
                        + right(dbo.sysschedules.freq_subday_interval,2)
                        + ' '
                        +     CASE(dbo.sysschedules.freq_subday_type)
                                    WHEN 1 THEN 'Once'
                                    WHEN 4 THEN 'Minutes'
                                    WHEN 8 THEN 'Hours'
                                END as char(16))
    END as 'Subday Frequency'
FROM dbo.sysjobs
LEFT OUTER JOIN dbo.sysjobschedules ON dbo.sysjobs.job_id = dbo.sysjobschedules.job_id
INNER JOIN dbo.sysschedules ON dbo.sysjobschedules.schedule_id = dbo.sysschedules.schedule_id
LEFT OUTER JOIN (SELECT job_id, max(run_duration) AS run_duration
            FROM dbo.sysjobhistory
            GROUP BY job_id) Q1
ON dbo.sysjobs.job_id = Q1.job_id
WHERE Next_run_time <> 0

ORDER BY [Start Date],[Start Time]
```

.

```
SELECT sdb.Name AS DatabaseName,
    COALESCE(CONVERT(VARCHAR(50), bus.backup_finish_date, 120),'-') AS LastBackUpDateTime
FROM sys.sysdatabases sdb
    LEFT OUTER JOIN msdb.dbo.backupset bus ON bus.database_name = sdb.name
ORDER BY sdb.name, bus.backup_finish_date DESC
```

.

```
SELECT
    [d].[name] AS database_name,
    [r].restore_date AS last_restore_date,
    [r].[user_name],
    [bs].[backup_finish_date] AS backup_creation_date,
    [bmf].[physical_device_name] AS [backup_file_used_for_restore]
FROM master.sys.databases [d]
    LEFT OUTER JOIN msdb.dbo.[restorehistory] r ON r.[destination_database_name] = d.Name
    INNER JOIN msdb.dbo.backupset [bs] ON [r].[backup_set_id] = [bs].[backup_set_id]
    INNER JOIN msdb.dbo.backupmediafamily bmf ON [bs].[media_set_id] = [bmf].[media_set_id]
ORDER BY [d].[name], [r].restore_date DESC
```

```
SELECT DISTINCT
 o.name AS Object_Name,o.type_desc
 FROM sys.sql_modules m
    INNER JOIN sys.objects  o ON m.object_id=o.object_id
 WHERE m.definition Like '%myField%'
 ORDER BY 2,1
```

SProcs, Views `myField` .

: https://riptutorial.com/ko/sql-server/topic/697/---

# 61: SQL

## Examples

### SQL

SQL . EXEC, EXECUTE sp_executesql string SQL .

```
sp_executesql N'SELECT * FROM sys.objects'
-- or
sp_executesql @stmt = N'SELECT * FROM sys.objects'
-- or
EXEC sp_executesql N'SELECT * FROM sys.objects'
-- or
EXEC('SELECT * FROM sys.columns')
-- or
EXECUTE('SELECT * FROM sys.tables')
```

SQL . sp_executesql , / SQL .

```
declare @table nvarchar(40) = N'product items'
EXEC(N'SELECT * FROM ' + @table)
declare @sql nvarchar(40) = N'SELECT * FROM ' + QUOTENAME(@table);
EXEC sp_executesql @sql
```

@table QUOTENAME . @table , .

### SQL

AS = ' ' SQL .

```
EXEC(N'SELECT * FROM product') AS USER = 'dbo'
```

SQL dbo . dbo SQL .

### SQL SQL

```
SET @sql = N'SELECT COUNT(*) FROM AppUsers WHERE Username = ''' + @user + ''' AND Password =
''' + @pass + ''''
EXEC(@sql)
```

user ***myusername "OR 1 = 1 -*** .

```
SELECT COUNT(*)
FROM AppUsers
WHERE Username = 'myusername' OR 1=1 --' AND Password = ''
```

@username 1 = 1 . 0 .

.

## SQL

SQL .

```
SET @sql = N'SELECT COUNT(*) FROM AppUsers WHERE Username = @user AND Password = @pass
EXEC sp_executesql @sql, '@user nvarchar(50), @pass nvarchar(50)', @username, @password
```

x 2 , .

sp_executesql SQL .

SQL : https://riptutorial.com/ko/sql-server/topic/6871/-sql

# 62: SQL

SQL Server .

## Examples

### SQL

```
if object_id('tempdb.dbo.#temp') is not null drop table #temp
create table #temp
(
    dateValue datetime,
    category varchar(3),
    amount decimal(36,2)
)

insert into #temp values ('1/1/2012', 'ABC', 1000.00)
insert into #temp values ('2/1/2012', 'DEF', 500.00)
insert into #temp values ('2/1/2012', 'GHI', 800.00)
insert into #temp values ('2/10/2012', 'DEF', 700.00)
insert into #temp values ('3/1/2012', 'ABC', 1100.00)


DECLARE
    @cols AS NVARCHAR(MAX),
    @query  AS NVARCHAR(MAX);

SET @cols = STUFF((SELECT distinct ',' + QUOTENAME(c.category)
            FROM #temp c
            FOR XML PATH(''), TYPE
            ).value('.', 'NVARCHAR(MAX)')
        ,1,1,'')

set @query = '
            SELECT
                dateValue,
                ' + @cols + '
            from
            (
                select
                     dateValue,
                     amount,
                     category
                from #temp
           ) x
            pivot
            (
                 sum(amount)
                for category in (' + @cols + ')
            ) p '


exec sp_executeSql @query
```

SQL : https://riptutorial.com/ko/sql-server/topic/10751/-sql-

# 63:

## Examples

email ()  .

```
ALTER TABLE  Company
ALTER COLUMN Email ADD MASKED WITH (FUNCTION = 'email()')
```

Company  .

mXXX@XXXX.com

zXXX@XXXX.com

rXXX@XXXX.com

.

```
ALTER TABLE  Company
ALTER COLUMN Phone ADD MASKED WITH (FUNCTION = 'partial(5,"XXXXXXX",2)')
```

,  ,  .

Company  .

(381) XXXXXXX39

(360) XXXXXXX01

(415) XXXXXXX05

### random ()

.

```
ALTER TABLE  Product
ALTER COLUMN Price ADD MASKED WITH (FUNCTION = 'random(100,200)')
```

(  ).

SELECT  .

```
ALTER TABLE  Company
ALTER COLUMN Postcode ADD MASKED WITH (FUNCTION = 'default()')
```

.

```
GRANT UNMASK TO MyUser
```

.

```
REVOKE UNMASK TO MyUser
```

: https://riptutorial.com/ko/sql-server/topic/7052/--

# 64:

SQL Server . CPU . .

Enterprise Edition

## Examples

```
select *
from sys.dm_resource_governor_workload_groups

select *
from sys.dm_resource_governor_resource_pools
```

.

```
CREATE RESOURCE POOL [PoolAdhoc] WITH(min_cpu_percent=0,
        max_cpu_percent=50,
        min_memory_percent=0,
        max_memory_percent=50)
GO
```

worload

```
CREATE WORKLOAD GROUP [AdhocMedium] WITH(importance=Medium) USING [PoolAdhoc]
```

.

```
create function [dbo].[ufn_ResourceGovernorClassifier]()
  returns sysname with schemabinding
as
begin
    return CASE
                WHEN APP_NAME() LIKE 'Microsoft Office%'                        THEN
'AdhocMedium'         -- Excel
                WHEN APP_NAME() LIKE 'Microsoft SQL Server Management Studio%'    THEN
'AdhocMedium'         -- Adhoc SQL
                WHEN SUSER_NAME() LIKE 'DOMAIN\username'                    THEN 'AdhocMedium'
-- Ssis
                ELSE 'default'
            END
end

GO

alter resource governor
with (classifier_function = dbo.ufn_ResourceGovernorClassifier)

GO

alter resource governor reconfigure
```

```
GO
```

: https://riptutorial.com/ko/sql-server/topic/4146/-

# 65: ID

## Examples

### SCOPE_IDENTITY ()

```
CREATE TABLE dbo.logging_table(log_id INT IDENTITY(1,1) PRIMARY KEY,
                               log_message VARCHAR(255))

CREATE TABLE dbo.person(person_id INT IDENTITY(1,1) PRIMARY KEY,
                        person_name VARCHAR(100) NOT NULL)
GO;

CREATE TRIGGER dbo.InsertToADifferentTable ON dbo.person
AFTER INSERT
AS
    INSERT INTO dbo.logging_table(log_message)
    VALUES('Someone added something to the person table')
GO;

INSERT INTO dbo.person(person_name)
VALUES('John Doe')

SELECT SCOPE_IDENTITY();
```

ID . dbo.person 1.

@@

```
CREATE TABLE dbo.logging_table(log_id INT IDENTITY(1,1) PRIMARY KEY,
                               log_message VARCHAR(255))

CREATE TABLE dbo.person(person_id INT IDENTITY(1,1) PRIMARY KEY,
                        person_name VARCHAR(100) NOT NULL)
GO;

CREATE TRIGGER dbo.InsertToADifferentTable ON dbo.person
AFTER INSERT
AS
    INSERT INTO dbo.logging_table(log_message)
    VALUES('Someone added something to the person table')
GO;

INSERT INTO dbo.person(person_name)
VALUES('John Doe')

SELECT @@IDENTITY;
```

ID . logging_table ID SQL Server .

### IDENT_CURRENT ( 'tablename')

```
SELECT IDENT_CURRENT('dbo.person');
```

ID .

## @@ IDENTITY MAX (ID)

```
SELECT MAX(Id) FROM Employees -- Display the value of Id in the last row in Employees table.
GO
INSERT INTO Employees (FName, LName, PhoneNumber) -- Insert a new row
VALUES ('John', 'Smith', '25558696525')
GO
SELECT @@IDENTITY
GO
SELECT MAX(Id) FROM Employees -- Display the value of Id of the newly inserted row.
GO
```

SELECT .

ID : https://riptutorial.com/ko/sql-server/topic/5674/---id

# 66: OLTP (Hekaton)

## Examples

```
-- Create demo database
CREATE DATABASE SQL2016_Demo
 ON  PRIMARY
(
    NAME = N'SQL2016_Demo',
    FILENAME = N'C:\Dump\SQL2016_Demo.mdf',
    SIZE = 5120KB,
    FILEGROWTH = 1024KB
 )
 LOG ON
 (
    NAME = N'SQL2016_Demo_log',
    FILENAME = N'C:\Dump\SQL2016_Demo_log.ldf',
    SIZE = 1024KB,
    FILEGROWTH = 10%
 )
GO

use SQL2016_Demo
go

-- Add Filegroup by MEMORY_OPTIMIZED_DATA type
ALTER DATABASE SQL2016_Demo
    ADD FILEGROUP MemFG CONTAINS MEMORY_OPTIMIZED_DATA
GO


--Add a file to defined filegroup
ALTER DATABASE SQL2016_Demo ADD FILE
    (
        NAME = MemFG_File1,
        FILENAME = N'C:\Dump\MemFG_File1' -- your file path, check directory exist before
executing this code
    )
TO FILEGROUP MemFG
GO

--Object Explorer -- check database created
GO

-- create memory optimized table 1
CREATE TABLE dbo.MemOptTable1
(
    Column1     INT         NOT NULL,
    Column2     NVARCHAR(4000)  NULL,
    SpidFilter  SMALLINT    NOT NULL   DEFAULT (@@spid),

    INDEX ix_SpidFiler NONCLUSTERED (SpidFilter),
    INDEX ix_SpidFilter HASH (SpidFilter) WITH (BUCKET_COUNT = 64),

    CONSTRAINT CHK_soSessionC_SpidFilter
        CHECK ( SpidFilter = @@spid ),
)
```

```
    WITH
        (MEMORY_OPTIMIZED = ON,
         DURABILITY = SCHEMA_AND_DATA);  --or DURABILITY = SCHEMA_ONLY
go

-- create memory optimized table 2
CREATE TABLE MemOptTable2
(
    ID INT NOT NULL PRIMARY KEY NONCLUSTERED HASH WITH (BUCKET_COUNT = 10000),
    FullName NVARCHAR(200) NOT NULL,
    DateAdded DATETIME NOT NULL
) WITH (MEMORY_OPTIMIZED = ON, DURABILITY = SCHEMA_AND_DATA)
GO
```

## .dll

```
SELECT
    OBJECT_ID('MemOptTable1') AS MemOptTable1_ObjectID,
    OBJECT_ID('MemOptTable2') AS MemOptTable2_ObjectID
GO

SELECT
    name,description
FROM sys.dm_os_loaded_modules
WHERE name LIKE '%XTP%'
GO
```

:

```
SELECT
    name,type_desc,durability_desc,Is_memory_Optimized
FROM sys.tables
    WHERE Is_memory_Optimized = 1
GO
```

, tempdb .

```
CREATE TYPE dbo.testTableType AS TABLE
(
   col1 INT NOT NULL,
   col2 CHAR(10)
);
```

memory_optimized=on .

```
CREATE TYPE dbo.testTableType AS TABLE
(
   col1 INT NOT NULL,
   col2 CHAR(10)
)WITH (MEMORY_OPTIMIZED=ON);
```

.

```
CREATE TABLE ##tempGlobalTabel
```

```
(
    Col1   INT   NOT NULL ,
    Col2   NVARCHAR(4000)
);
```

:

```
CREATE TABLE dbo.tempGlobalTabel
(
    Col1   INT   NOT NULL   INDEX ix NONCLUSTERED,
    Col2   NVARCHAR(4000)
)
    WITH
        (MEMORY_OPTIMIZED = ON,
         DURABILITY = SCHEMA_ONLY);
```

(## temp) .

1. `##temp SCHEMA_ONLY`
   - .
2. Transact-SQL `##temp` temp .
3. `DROP TABLE ##temp DELETE FROM temp .`
4. `CREATE TABLE ##temp -` .

. T-SQL .

```
DECLARE @tvp TABLE
(
    col1   INT NOT NULL ,
    Col2   CHAR(10)
);
```

.

```
CREATE TYPE dbo.memTypeTable
AS TABLE
(
    Col1  INT NOT NULL INDEX ix1,
    Col2  CHAR(10)
)
WITH
    (MEMORY_OPTIMIZED = ON);
```

.

```
DECLARE @tvp memTypeTable
insert INTO @tvp
values (1,'1'),(2,'2'),(3,'3'),(4,'4'),(5,'5'),(6,'6')

SELECT * FROM @tvp
```

:

```
Col1    Col2
1       1
2       2
3       3
4       4
5       5
6       6
```

```
CREATE TABLE [dbo].[MemOptimizedTemporalTable]
(
    [BusinessDocNo] [bigint] NOT NULL,
    [ProductCode] [int] NOT NULL,
    [UnitID] [tinyint] NOT NULL,
    [PriceID] [tinyint] NOT NULL,
    [SysStartTime] [datetime2](7) GENERATED ALWAYS AS ROW START NOT NULL,
    [SysEndTime] [datetime2](7) GENERATED ALWAYS AS ROW END NOT NULL,
    PERIOD FOR SYSTEM_TIME ([SysStartTime], [SysEndTime]),

    CONSTRAINT [PK_MemOptimizedTemporalTable]  PRIMARY KEY NONCLUSTERED
    (
        [BusinessDocNo] ASC,
        [ProductCode] ASC
    )
)
WITH (
    MEMORY_OPTIMIZED = ON , DURABILITY = SCHEMA_AND_DATA, -- Memory Optimized Option ON
    SYSTEM_VERSIONING = ON (HISTORY_TABLE = [dbo].[MemOptimizedTemporalTable_History] ,
DATA_CONSISTENCY_CHECK = ON )
)
```

OLTP (Hekaton) : https://riptutorial.com/ko/sql-server/topic/5295/--oltp--hekaton-

# 67:

( ) :

- 
- 
- Charindex
- 
- 
- 
- 
- 
- 
- Ltrim
- Nchar
- Patindex
- 
- 
- 
- 
- 
- Rtrim
- Soundex
- 
- 
- String_escape
- String_split
- 
- 
- 

-

## Examples

.

:

1. . `text ntext varchar nvarchar`
2. . 0 `bigint` (9,223,372,036,854,775,807).
   .

```
SELECT LEFT('This is my string', 4) -- result: 'This'
```

entier .

```
SELECT LEFT('This is my string', 50) -- result: 'This is my string'
```

.

:

1. . `text ntext varchar nvarchar`
2. . 0 `bigint` (9,223,372,036,854,775,807). .

```
SELECT RIGHT('This is my string', 6) -- returns 'string'
```

entier .

```
SELECT RIGHT('This is my string', 50) -- returns 'This is my string'
```

char .

:

1. . `text ntext varchar nvarchar` .
2. . ( `int bigint` ). ( : SQL 1 1 . 1 . , 0 . . x (start index - 1). 0 .
3. . 0 `bigint` (9,223,372,036,854,775,807). .

```
SELECT SUBSTRING('This is my string', 6, 5) -- returns 'is my'
```

+ entier .

```
SELECT SUBSTRING('Hello World',1,100) -- returns 'Hello World'
```

.

```
SELECT SUBSTRING('Hello World',15,10) -- returns ''
```

## ASCII

ASCII int .

```
SELECT ASCII('t') -- Returns 116
SELECT ASCII('T') -- Returns 84
SELECT ASCII('This') -- Returns 84
```

ASCII 127 .

```
SELECT ASCII(N'ï') -- returns 239 when `SERVERPROPERTY('COLLATION') =
'SQL_Latin1_General_CP1_CI_AS'`
```

63 int (ASCII ).

```
SELECT ASCII(N'□') -- returns 63
SELECT ASCII(nchar(2039)) -- returns 63
```

## CharIndex

.

:

1. ( 8000 )
2. ( , )
3. ( ) . int big int . 1 .

`varchar(max)` , `nvarchar(max)` `varbinary(max)` `CHARINDEX` `bigint` . `int` .

```
SELECT CHARINDEX('is', 'this is my string') -- returns 3
SELECT CHARINDEX('is', 'this is my string', 4) -- returns 6
SELECT CHARINDEX(' is', 'this is my string') -- returns 5
```

int ASCII char .

```
SELECT CHAR(116) -- Returns 't'
SELECT CHAR(84)  -- Returns 'T'
```

/ `CHAR(10)` , `CHAR(13)` . AsciiTable.com .

0 255 CHAR `NULL` `NULL` .

`CHAR` `char(1)`

.

: `LEN` .

```
SELECT LEN('My string'), -- returns 9
       LEN('My string   '), -- returns 9
       LEN('   My string') -- returns 12
```

.  LEN  .

```
DECLARE @str varchar(100) = 'My string   '
SELECT LEN(@str + 'x') - 1 -- returns 12
```

.  . ( ) .

```
SELECT LEN(CONVERT(NVARCHAR(MAX), @str) + 'x') - 1
```

DATALENGTH .

```
DECLARE @str varchar(100) = 'My string   '
SELECT DATALENGTH(@str) -- returns 12
```

DATALENGTH () . varchar nvarchar .

```
DECLARE @str nvarchar(100) = 'My string   '
SELECT DATALENGTH(@str) -- returns 24
```

( ) . 0 .

```
DECLARE @str nvarchar(100) = 'My string   '
SELECT DATALENGTH(@str) / DATALENGTH(LEFT(LEFT(@str, 1) + 'x', 1)) -- returns 12
```

## SQL Server 2012 . ( ) .

REPLACE LEN . .

## SQL Server 2012

. CONCAT .

```
SELECT CONCAT('This', ' is', ' my', ' string') -- returns 'This is my string'
```

: ( + ) concat null .

```
SELECT CONCAT('This', NULL, ' is', ' my', ' string'), -- returns 'This is my string'
       'This' + NULL + ' is' + ' my' + ' string' -- returns NULL.
```

.

```
SELECT CONCAT('This', ' is my ', 3, 'rd string') -- returns 'This is my 3rd string'
```

.

```
DECLARE @Age INT=23;
SELECT CONCAT('Ram is ', @Age,' years old');  -- returns 'Ram is 23 years old'
```

SQL Server 2012

CONCAT ( + ) . .

```
SELECT 'This is the number ' + CAST(42 AS VARCHAR(5)) --returns 'This is the number 42'
```

( varchar nvarchar ) .

:

1. . varchar .

```
SELECT LOWER('This IS my STRING') -- Returns 'this is my string'

DECLARE @String nchar(17) = N'This IS my STRING';
SELECT LOWER(@String) -- Returns 'this is my string'
```

( varchar nvarchar ) .

:

1. . varchar .

```
SELECT UPPER('This IS my STRING') -- Returns 'THIS IS MY STRING'

DECLARE @String nchar(17) = N'This IS my STRING';
SELECT UPPER(@String) -- Returns 'THIS IS MY STRING'
```

## LTrim

( varchar nvarchar ) . , .

:

1. . text , ntext image varcher .

```
SELECT LTRIM('    This is my string') -- Returns 'This is my string'
```

## RTrim

( varchar nvarchar ) . , .

:

1. . text , ntext image varcher .

```
SELECT RTRIM('This is my string     ') -- Returns 'This is my string'
```

.

:

1. . `nchar` `nvarchar` .

```
SELECT UNICODE(N'Ɛ') -- Returns 400

DECLARE @Unicode nvarchar(11) = N'Ɛ is a char'
SELECT UNICODE(@Unicode) -- Returns 400
```

## NChar

( `nchar(1)` `nvarchar(2)` ) .

:

1. . 0 65535 (CS) 0 1114111 . `null` . .

```
SELECT NCHAR(257) -- Returns 'ā'
SELECT NCHAR(400) -- Returns 'Ɛ'
```

.

:

1. . `varchar` .

```
Select REVERSE('Sql Server') -- Returns 'revreS lqS'
```

## PatIndex

.

:

1. . . 8000 . ( `%` , `_` ) . . .

2. . .

```
SELECT PATINDEX('%ter%', 'interesting') -- Returns 3.

SELECT PATINDEX('%t_r%t%', 'interesting') -- Returns 3.

SELECT PATINDEX('ter%', 'interesting') -- Returns 0, since 'ter' is not at the start.

SELECT PATINDEX('inter%', 'interesting') -- Returns 1.

SELECT PATINDEX('%ing', 'interesting') -- Returns 9.
```

( `varchar` ) .

:

1. . ( 8000). `null` . 0 . (8000 Replicate .

```
SELECT SPACE(-1) -- Returns NULL
SELECT SPACE(0)  -- Returns an empty string
SELECT SPACE(3)  -- Returns '   ' (a string containing 3 spaces)
```

.

:

1. . 2 .
2. . `bigint` . , `null` . 0 .

```
SELECT REPLICATE('a', -1)  -- Returns NULL

SELECT REPLICATE('a', 0)  -- Returns ''

SELECT REPLICATE('a', 5)  -- Returns 'aaaaa'

SELECT REPLICATE('Abc', 3) -- Returns 'AbcAbcAbc'
```

**:** `varchar(max)` `nvarchar(max)` 8000 . .

```
SELECT LEN(REPLICATE('a b c d e f g h i j k l', 350)) -- Returns 7981

SELECT LEN(REPLICATE(cast('a b c d e f g h i j k l' as varchar(max)), 350)) -- Returns 8050
```

( `varchar` `nvarchar` ) .

:

1. . . 2 .
2. . . 2 . pattern .
3. . . 2 .

```
SELECT REPLACE('This is my string', 'is', 'XX') -- Returns 'ThXX XX my string'.
```

**:**

- `varchar(max)` `nvarchar(max)` `replace` 8,000 .
- - `nvarchar` `nvarchar` , `varchar` .
- `NULL` `NULL` `NULL`

## String_Split

SQL Server 2016

. `STRING_SPLIT()` `FROM` `FROM` .

:

1. . ( `char` , `nchar` , `varchar` `nvarchar` )
2. . ( `char(1)` , `nchar(1)` , `varchar(1)` `nvarchar(1)` ) .

. `value` `nvarchar` `nchar` `nvarchar` , `varchar` .

.

```
SELECT value FROM STRING_SPLIT('Lorem ipsum dolor sit amet.', ' ');
```

:

```
value
-----
Lorem
ipsum
dolor
sit
amet.
```

:

`STRING_SPLIT` **130** . 130 SQL Server `STRING_SPLIT` . .

```
ALTER DATABASE [database_name] SET COMPATIBILITY_LEVEL = 130
```

SQL Server 2016

SQL Server . . Aaron Bertrand Split strings .

( `varchar` ) .

:

1. float . .
2. . . , ( ) . 10.
3. . . . 16 16 .

```
SELECT STR(1.2) -- Returns '         1'

SELECT STR(1.2, 3) -- Returns '  1'

SELECT STR(1.2, 3, 2) -- Returns '1.2'

SELECT STR(1.2, 5, 2) -- Returns ' 1.20'

SELECT STR(1.2, 5, 5) -- Returns '1.200'

SELECT STR(1, 5, 2) -- Returns ' 1.00'

SELECT STR(1) -- Returns '         1'
```

SQL Server .

:

1. . 128 ( `sysname` ) . 128 function `null` .
2. . . . ( ' `` ), ( `{` , `[` , `(` , `<` `>` , `)` , `]` , `}` ) ( `"` ). null .

```
SELECT QUOTENAME('what''s my name?')      -- Returns [what's my name?]

SELECT QUOTENAME('what''s my name?', '[') -- Returns [what's my name?]
SELECT QUOTENAME('what''s my name?', ']') -- Returns [what's my name?]

SELECT QUOTENAME('what''s my name?', '''') -- Returns 'what''s my name?'

SELECT QUOTENAME('what''s my name?', '"') -- Returns "what's my name?"

SELECT QUOTENAME('what''s my name?', ')') -- Returns (what's my name?)
SELECT QUOTENAME('what''s my name?', '(') -- Returns (what's my name?)

SELECT QUOTENAME('what''s my name?', '<') -- Returns <what's my name?>
SELECT QUOTENAME('what''s my name?', '>') -- Returns <what's my name?>

SELECT QUOTENAME('what''s my name?', '{') -- Returns {what's my name?}
SELECT QUOTENAME('what''s my name?', '}') -- Returns {what's my name?}

SELECT QUOTENAME('what''s my name?', '`') -- Returns `what's my name?`
```

## Soundex

4 ( `varchar` ) .

:

1. . .

soundex 4 . , ( a, e, i, o, u, h, w y ) .

```
SELECT SOUNDEX ('Smith') -- Returns 'S530'

SELECT SOUNDEX ('Smythe') -- Returns 'S530'
```

soundex ( `int` ) .

:

1. 1.
2. 2.

.

soundex 4 0 .

```
SELECT  SOUNDEX('Green'),  -- G650
```

```
        SOUNDEX('Greene'),  -- G650
        DIFFERENCE('Green','Greene') -- Returns 4

SELECT  SOUNDEX('Blotchet-Halls'),  -- B432
        SOUNDEX('Greene'),  -- G650
        DIFFERENCE('Blotchet-Halls', 'Greene') -- Returns 0
```

## SQL Server 2012

( ) NVARCHAR . - .

:

1. value . . .
2. format . NVARCHAR . Microsoft .
3. culture . . culture nvarchar culture.

:

```
DECLARE @d DATETIME = '2016-07-31';

SELECT
    FORMAT ( @d, 'd', 'en-US' ) AS 'US English Result' -- Returns '7/31/2016'
   ,FORMAT ( @d, 'd', 'en-gb' ) AS 'Great Britain English Result' -- Returns '31/07/2016'
   ,FORMAT ( @d, 'd', 'de-de' ) AS 'German Result' -- Returns '31.07.2016'
   ,FORMAT ( @d, 'd', 'zh-cn' ) AS 'Simplified Chinese (PRC) Result' -- Returns '2016/7/31'
   ,FORMAT ( @d, 'D', 'en-US' ) AS 'US English Result' -- Returns 'Sunday, July 31, 2016'
   ,FORMAT ( @d, 'D', 'en-gb' ) AS 'Great Britain English Result' -- Returns '31 July 2016'
   ,FORMAT ( @d, 'D', 'de-de' ) AS 'German Result' -- Returns 'Sonntag, 31. Juli 2016'
```

:

```
SELECT FORMAT( @d, 'dd/MM/yyyy', 'en-US' ) AS 'DateTime Result' -- Returns '31/07/2016'
      ,FORMAT(123456789,'###-##-####') AS 'Custom Number Result' -- Returns '123-45-6789',
      ,FORMAT( @d,'dddd, MMMM dd, yyyy hh:mm:ss tt','en-US') AS 'US' -- Returns 'Sunday, July
31, 2016 12:00:00 AM'
      ,FORMAT( @d,'dddd, MMMM dd, yyyy hh:mm:ss tt','hi-IN') AS 'Hindi' -- Returns रविवार, जुलाई 31,
2016 12:00:00 पूर्वाह्न
   ,FORMAT ( @d, 'dddd', 'en-US' )  AS 'US' -- Returns 'Sunday'
      ,FORMAT ( @d, 'dddd', 'hi-IN' )  AS 'Hindi' -- Returns 'रविवार'
```

FORMAT CURRENCY , PERCENTAGE NUMBERS .

```
DECLARE @Price1 INT = 40
SELECT FORMAT(@Price1,'c','en-US') AS 'CURRENCY IN US Culture' -- Returns '$40.00'
       ,FORMAT(@Price1,'c','de-DE') AS 'CURRENCY IN GERMAN Culture' -- Returns '40,00 €'
```

10 .

```
DECLARE @Price DECIMAL(5,3) = 40.356
SELECT FORMAT( @Price, 'C') AS 'Default', -- Returns '$40.36'
       FORMAT( @Price, 'C0') AS 'With 0 Decimal', -- Returns '$40'
       FORMAT( @Price, 'C1') AS 'With 1 Decimal', -- Returns '$40.4'
       FORMAT( @Price, 'C2') AS 'With 2 Decimal', -- Returns '$40.36'
```

```
   DECLARE @Percentage float = 0.35674
   SELECT FORMAT( @Percentage, 'P') AS '% Default', -- Returns '35.67 %'
   FORMAT( @Percentage, 'P0') AS '% With 0 Decimal', -- Returns '36 %'
   FORMAT( @Percentage, 'P1') AS '% with 1 Decimal'  -- Returns '35.7 %'
```

```
DECLARE @Number AS DECIMAL(10,2) = 454545.389
SELECT FORMAT( @Number, 'N','en-US') AS 'Number Format in US', -- Returns '454,545.39'
FORMAT( @Number, 'N','en-IN')  AS 'Number Format in INDIA', -- Returns '4,54,545.39'
FORMAT( @Number, '#.0')     AS 'With 1 Decimal', -- Returns '454545.4'
FORMAT( @Number, '#.00')    AS 'With 2 Decimal', -- Returns '454545.39'
FORMAT( @Number, '#,##.00') AS 'With Comma and 2 Decimal', -- Returns '454,545.39'
FORMAT( @Number, '##.00')   AS 'Without Comma and 2 Decimal', -- Returns '454545.39'
FORMAT( @Number, '000000000') AS 'Left-padded to nine digits' -- Returns '000454545'
```

: ( )

```
Category         Type             .Net type
-----------------------------------------
Numeric          bigint           Int64
Numeric          int              Int32
Numeric          smallint         Int16
Numeric          tinyint          Byte
Numeric          decimal          SqlDecimal
Numeric          numeric          SqlDecimal
Numeric          float            Double
Numeric          real             Single
Numeric          smallmoney       Decimal
Numeric          money            Decimal
Date and Time    date             DateTime
Date and Time    time             TimeSpan
Date and Time    datetime         DateTime
Date and Time    smalldatetime    DateTime
Date and Time    datetime2        DateTime
Date and Time    datetimeoffset   DateTimeOffset
```

**:**

- `FORMAT` culture `NULL` . , format `NULL` .
- `FORMAT` .NET Framework CLR (Common Language Runtime) .
- `FORMAT` CLR . ( ) ( ) .

FORMAT .

## String_escape

SQL Server 2016

( `nvarchar(max)` ) .

:

1. . `nvarchar` .
2. . . `'json'` .

```
SELECT STRING_ESCAPE('\   /
\\    "     ', 'json') -- returns '\\\t\/\n\\\\\t\"\t'
```

:

```
Special character    Encoded sequence
-------------------------------------
Quotation mark (")   \"
Reverse solidus (\)  \\
Solidus (/)          \/
Backspace            \b
Form feed            \f
New line             \n
Carriage return      \r
Horizontal tab       \t


Control character    Encoded sequence
-------------------------------------
CHAR(0)             \u0000
CHAR(1)             \u0001
...                 ...
CHAR(31)            \u001f
```

: https://riptutorial.com/ko/sql-server/topic/4113/-

# 68:

. . .

. .

## Examples

SQL Server DDL DML .

DDL DDL (Data Definition Language) . `CREATE` , `ALTER` `DROP` Transact-SQL .

DML DML (Data Manipulation Language) . `INSERT` , `UPDATE` `DELETE` Transact-SQL .

DML .

1. After ( )
   - AFTER .
   - AFTER UPDATE .
   - AFTER DELETE .
2. - INSTEAD OF INSERT .
   - INSTEAD OF UPDATE .
   - INSTEAD OF DELETE .

### DML

DML dml ( `insert` , `update` `delete` ) .
dml dml . , dml , .

DML , `inserted` `deleted` .

DML , . , .

:

```
CREATE TRIGGER tblSomething_InsertOrUpdate ON tblSomething
FOR INSERT
AS

    INSERT INTO tblAudit (TableName, RecordId, Action)
    SELECT 'tblSomething', Id, 'Inserted'
    FROM Inserted

GO

CREATE TRIGGER tblSomething_InsertOrUpdate ON tblSomething
FOR UPDATE
AS
```

```
    INSERT INTO tblAudit (TableName, RecordId, Action)
    SELECT 'tblSomething', Id, 'Updated'
    FROM Inserted

GO

CREATE TRIGGER tblSomething_InsertOrUpdate ON tblSomething
FOR DELETE
AS

    INSERT INTO tblAudit (TableName, RecordId, Action)
    SELECT 'tblSomething', Id, 'Deleted'
    FROM Deleted

GO
```

tblSomething , tblAudit .

: https://riptutorial.com/ko/sql-server/topic/5032/

# 69:

- DECLARE @VariableName DataType [= Value];
- SET @VariableName = Value;

## Examples

```
DECLARE @Employees TABLE
(
    EmployeeID INT NOT NULL PRIMARY KEY,
    FirstName NVARCHAR(50) NOT NULL,
    LastName NVARCHAR(50) NOT NULL,
    ManagerID INT NULL
)
```

`CREATE TABLE Name (Columns) .  DECLARE @Name TABLE (Columns) .`

`SELECT` SQL Server . .

"@TableVariableName" .

```
DECLARE @Table1 TABLE (Example INT)
DECLARE @Table2 TABLE (Example INT)

/*
-- the following two commented out statements would generate an error:
SELECT *
FROM @Table1
INNER JOIN @Table2 ON @Table1.Example = @Table2.Example

SELECT *
FROM @Table1
WHERE @Table1.Example = 1
*/

-- but these work fine:
SELECT *
FROM @Table1 T1
INNER JOIN @Table2 T2 ON T1.Example = T2.Example

SELECT *
FROM @Table1 Table1
WHERE Table1.Example = 1
```

### SET

```
DECLARE @VariableName INT
SET @VariableName = 1
PRINT @VariableName
```

1

SET .

## SELECT

SELECT .

```
DECLARE @Variable1 INT, @Variable2 VARCHAR(10)
SELECT @Variable1 = 1, @Variable2 = 'Hello'
PRINT @Variable1
PRINT @Variable2
```

1

---

SELECT . ( . .)

```
CREATE TABLE #Test (Example INT)
INSERT INTO #Test VALUES (1), (2)

DECLARE @Variable INT
SELECT @Variable = Example
FROM #Test
ORDER BY Example ASC

PRINT @Variable
```

2

```
SELECT TOP 1 @Variable = Example
FROM #Test
ORDER BY Example ASC

PRINT @Variable
```

1

.

```
SELECT TOP 0 @Variable = Example
FROM #Test
ORDER BY Example ASC

PRINT @Variable
```

1

.

```
DECLARE
  @Var1 INT = 5,
  @Var2 NVARCHAR(50) = N'Hello World',
  @Var3 DATETIME = GETDATE()
```

SQL Server 2008 R2

:

- `+=`
- `-=`
- `*=`
- `/=`
- `%=`
- `&=` AND
- `^=` XOR
- `|=` OR

:

```
DECLARE @test INT = 42;
SET @test += 1;
PRINT @test;    --43
SET @test -= 1;
PRINT @test;    --42
SET @test *= 2
PRINT @test;    --84
SET @test /= 2;
PRINT @test;    --42
```

.

```
DECLARE @CurrentID int = (SELECT TOP 1 ID FROM Table ORDER BY CreateDate desc)


DECLARE @Year int = 2014
DECLARE @CurrentID int = (SELECT ID FROM Table WHERE Year = @Year)
```

.

: https://riptutorial.com/ko/sql-server/topic/2566/

# 70:

SQL Server 2008 MERGE , .

MERGE .

- MSDN - https://msdn.microsoft.com/en-us/library/bb510625.aspx [WITH <common_table_expression> [, ... n]] MERGE [TOP (expression) [PERCENT]] [INTO] < ] [] [] [] [WITH (<merge_history>)] [[AS] _ ) USING <table_source> ON <merge_search_condition> [ [AND <clause_search_condition>] THEN <merge_matched> [... n] [NOT BY MATCHED [BY THERE <merge_matches>] [OPEN (<query_hint> [,.]) [TARGET] [AND <clause_search_condition>] THEN <merge_not_matched> ..n])]; <target_table> :: = {[database_name. schema_name. | schema_name. ], {{...}]}}} <table_source> :: = {table_or_view_name [target_number]] { [AS] table_alias] [<tablesample_clause>] [WITH (table_hint [[,] ... n])] | rowset_function [[AS] table_alias] [(bulk_column_alias [, ... n])] | user_defined_function [[AS] _ ] | OPENXML <openxml_clause> | derived_table [AS] table_alias [(column_alias [, ... n])] | <joined_table> | <pivoted_table> | <unpivoted_table>} <merge_search_condition> :: = <search_condition> <merge_matched> :: = {UPDATE SET <set_clause> | DELETE} <set_clause> :: = SET {column_name = {expression | DEFAULT | NULL} | {udt_column_name. {{property_name = expression | field_name = } | method_name ( [, ... n])}} | column_name {.WRITE (, @Offset, @Length)} | @variable = | @variable = column = expression | column_name {+ = | - = | * = | / = | % = | & = | ^ = | | =} | @ {+ = | - = | * = | / = | % = | & = | ^ = | | =} | @variable = column {+ = | - = | * = | / = | % = | & = | ^ = | | =} } [, ... n] <merge_not_matched> :: = {INSERT [(column_list)] {VALUES (values_list) | }} < > :: = < > :: = {[NOT] | (<search_condition>)} [{AND | } [NOT] {| (<search_condition>)}] [, ... n] :: = { {= | <> | ! = |

  | > = | ! > | <| <= | ! <} | string_expression [NOT] LIKE string_expression [ESCAPE 'escape_character'] | [NOT] BETWEEN | [NOT] NULL | ({column | *}, '<contains_search_condition>') | FREETEXT ({column | *}, 'freetext_string') | expression [NOT] IN ( | [, ... n]) | {= | <> | ! = | | > = | ! > | <| <= | ! <} { | | ANY} ( ) | EXISTS ( )} <output_clause> :: = {[OUTPUT <dml_select_list> INTO {@table_variable | output_table} [(column_list)]] [OUTPUT <dml_select_list>]} <dml_select_list> :: = {<column_name> | scalar_expression} [[AS] column_alias_identifier] [, ... n] <column_name> :: = {DELETED | | from_table_name}. {* | column_name} | $ action

, . , , .

## Examples

### / / MERGE

```
MERGE INTO targetTable
```

```
USING sourceTable
ON (targetTable.PKID = sourceTable.PKID)

WHEN MATCHED AND (targetTable.PKID > 100) THEN
    DELETE

WHEN MATCHED AND (targetTable.PKID <= 100) THEN
    UPDATE SET
        targetTable.ColumnA = sourceTable.ColumnA,
        targetTable.ColumnB = sourceTable.ColumnB

WHEN NOT MATCHED THEN
    INSERT (ColumnA, ColumnB) VALUES (sourceTable.ColumnA, sourceTable.ColumnB);

WHEN NOT MATCHED BY SOURCE THEN
    DELETE
; --< Required
```

:

- MERGE INTO targetTable -
- USING sourceTable - ( )
- ON ... - targetTable sourceTable .
- WHEN MATCHED -
- ◦ AND (targetTable.PKID > 100) -
- THEN DELETE - targetTable
- THEN UPDATE - SET .... SET ....
- WHEN NOT MATCHED - **targetTable** .
- WHEN NOT MATCHED BY SOURCE - **sourceTable**

:

. , . WHEN NOT MATCHED THEN INSERT .

.

:

- WHEN MATCHED INSERT
- UPDATE X . .

## CTE

```
WITH SourceTableCTE AS
(
    SELECT * FROM SourceTable
)
MERGE
 TargetTable AS target
USING SourceTableCTE AS source
ON (target.PKID = source.PKID)
WHEN MATCHED THEN
    UPDATE SET target.ColumnA = source.ColumnA
WHEN NOT MATCHED THEN
```

```
    INSERT (ColumnA) VALUES (Source.ColumnA);
```

## MERGE

```
MERGE INTO TargetTable  AS Target
USING (VALUES (1,'Value1'), (2, 'Value2'), (3,'Value3'))
       AS Source (PKID, ColumnA)
ON Target.PKID = Source.PKID
WHEN MATCHED THEN
    UPDATE SET target.ColumnA= source.ColumnA
WHEN NOT MATCHED THEN
    INSERT (PKID, ColumnA) VALUES (Source.PKID, Source.ColumnA);
```

-

MERGE -

1. **dbo.Product** : .
2. **dbo.ProductNew** : .

T-SQL .

```
IF OBJECT_id(N'dbo.Product',N'U') IS NOT NULL
DROP TABLE dbo.Product
GO

CREATE TABLE dbo.Product (
ProductID INT PRIMARY KEY,
ProductName NVARCHAR(64),
PRICE MONEY
)

IF OBJECT_id(N'dbo.ProductNew',N'U') IS NOT NULL
DROP TABLE dbo.ProductNew
GO

CREATE TABLE dbo.ProductNew (
ProductID INT PRIMARY KEY,
ProductName NVARCHAR(64),
PRICE MONEY
)

INSERT INTO dbo.Product VALUES(1,'IPod',300)
,(2,'IPhone',400)
,(3,'ChromeCast',100)
,(4,'raspberry pi',50)

INSERT INTO dbo.ProductNew VALUES(1,'Asus Notebook',300)
,(2,'Hp Notebook',400)
,(3,'Dell Notebook',100)
,(4,'raspberry pi',50)
```

dbo.ProductNew dbo.Product Target Table . .

1. dbo.ProductNew dbo.Product dbo.Product .

2. dob.ProductNew dob.Product dbo.Product .

3. dbo.ProductNew dbo.Product dbo.Product . MERGE .

```
MERGE dbo.Product AS SourceTbl
USING dbo.ProductNew AS TargetTbl ON (SourceTbl.ProductID = TargetTbl.ProductID)
WHEN MATCHED
            AND SourceTbl.ProductName <> TargetTbl.ProductName
            OR SourceTbl.Price <> TargetTbl.Price
    THEN UPDATE SET SourceTbl.ProductName = TargetTbl.ProductName,
                SourceTbl.Price = TargetTbl.Price
WHEN NOT MATCHED
    THEN INSERT (ProductID, ProductName, Price)
         VALUES (TargetTbl.ProductID, TargetTbl.ProductName, TargetTbl.Price)
WHEN NOT MATCHED BY SOURCE
    THEN DELETE
OUTPUT $action, INSERTED.*, DELETED.*;
```

| | $action | ProductID | ProductName | PRICE | ProductID | ProductName | PRICE |
|---|---|---|---|---|---|---|---|
| 1 | UPDATE | 1 | Asus Notebook | 300.00 | 1 | IPod | 300.00 |
| 2 | UPDATE | 2 | Hp Notebook | 400.00 | 2 | IPhone | 400.00 |
| 3 | UPDATE | 3 | Dell Notebook | 100.00 | 3 | ChromeCast | 100.00 |

: MERGE .

## EXCEPT

EXCEPT .

```
MERGE TargetTable targ
USING SourceTable AS src
    ON src.id = targ.id
WHEN MATCHED
    AND EXISTS (
        SELECT src.field
        EXCEPT
        SELECT targ.field
        )
    THEN
        UPDATE
        SET field = src.field
WHEN NOT MATCHED BY TARGET
    THEN
        INSERT (
            id
            ,field
            )
        VALUES (
            src.id
            ,src.field
            )
WHEN NOT MATCHED BY SOURCE
    THEN
        DELETE;
```

: https://riptutorial.com/ko/sql-server/topic/4550/

# 71:

## Examples

. CRUD . . SQL Server INFORMATION_SCHEMA () .

, .

```
SELECT 'GRANT EXEC ON core.' + r.ROUTINE_NAME + ' TO ' + <MyDatabaseUsername>
FROM INFORMATION_SCHEMA.ROUTINES r
WHERE r.ROUTINE_CATALOG = '<MyDataBaseName>'
```

: https://riptutorial.com/ko/sql-server/topic/7929/---

# 72:

(UDT) . , .

UDT .

- .
- UDT PRIMARY KEY UNIQUE UDT .
- UDT .

## Examples

### int UDT

```
CREATE TYPE dbo.Ids as TABLE
(
    Id int PRIMARY KEY
)
```

### UDT

```
CREATE TYPE MyComplexType as TABLE
(
    Id int,
    Name varchar(10)
)
```

### UDT :

```
CREATE TYPE MyUniqueNamesType as TABLE
(
    FirstName varchar(10),
    LastName varchar(10),
    UNIQUE (FirstName,LastName)
)
```

: .

### UDT :

```
CREATE TYPE MyUniqueNamesType as TABLE
(
    FirstName varchar(10),
    LastName varchar(10),
    CreateDate datetime default GETDATE()
    PRIMARY KEY (FirstName,LastName)
)
```

: https://riptutorial.com/ko/sql-server/topic/5280/---

# 73:

## Examples

. .

```
CREATE TABLE Employees
(
    ID CHAR(900),
    FirstName NVARCHAR(3000),
    LastName NVARCHAR(3000),
    StartYear CHAR(900)
)
GO

CREATE CLUSTERED INDEX IX_Clustered
ON Employees(ID)
GO
```

. . SQL Server 2008 / 2012 999 .

: https://msdn.microsoft.com/en-us/library/ms143432.aspx

```
CREATE TABLE Employees
(
    ID CHAR(900),
    FirstName NVARCHAR(3000),
    LastName NVARCHAR(3000),
    StartYear CHAR(900)
)
GO

CREATE NONCLUSTERED INDEX IX_NonClustered
ON Employees(StartYear)
GO
```

```
SP_HELPINDEX tableName
```

```
CREATE VIEW  View_Index02
WITH SCHEMABINDING
AS
SELECT c.CompanyName, o.OrderDate, o.OrderID, od.ProductID
     FROM dbo.Customers C
        INNER JOIN dbo.orders O ON c.CustomerID=o.CustomerID
            INNER JOIN dbo.[Order Details] od ON o.OrderID=od.OrderID
GO

CREATE UNIQUE CLUSTERED INDEX IX1 ON
    View_Index02(OrderID, ProductID)
```

```
DROP INDEX IX_NonClustered ON Employees
```

.

```
sys.dm_db_index_physical_stats (
    { database_id | NULL | 0 | DEFAULT }
  , { object_id | NULL | 0 | DEFAULT }
  , { index_id | NULL | 0 | -1 | DEFAULT }
  , { partition_number | NULL | 0 | DEFAULT }
  , { mode | NULL | DEFAULT }
)


Sample :

SELECT * FROM sys.dm_db_index_physical_stats
    (DB_ID(N'DBName'), OBJECT_ID(N'IX_NonClustered '), NULL, NULL , 'DETAILED');
```

| avg_fragmentation_in_percent | |
|---|---|
| *> 5 %  <= 30 %* | |
| *> 30 %* | REBUILD |

```
ALTER INDEX IX_NonClustered ON tableName REORGANIZE;


ALTER INDEX ALL ON Production.Product
 REBUILD WITH (FILLFACTOR = 80, SORT_IN_TEMPDB = ON,
          STATISTICS_NORECOMPUTE = ON);
```

.

```
ALTER INDEX All ON tableName REBUILD;
```

.

.

```
ALTER INDEX All ON tableName REORGANIZE;
```

,

```
EXEC sp_MSForEachTable 'ALTER INDEX ALL ON ? REBUILD'
```

"SP_HELPINDEX Table_Name"  Kimberly Tripp  (  ) .  .
:

```
USE Adventureworks
EXEC sp_SQLskills_SQL2012_helpindex 'dbo.Product'
```

Tibor Karaszi  (  ) .  . :

```
USE Adventureworks
EXEC sp_indexinfo 'dbo.Product'
```

: https://riptutorial.com/ko/sql-server/topic/4998/

# 74:

## Examples

### 1.

. ,,,, .

: https://msdn.microsoft.com/en-us/library/bb522893.aspx

### 2.

```
ALTER DATABASE [MyDatabase] SET ENABLE_BROKER WITH ROLLBACK IMMEDIATE;
```

### 3. ( )

```
USE [MyDatabase]

CREATE MESSAGE TYPE [//initiator] VALIDATION = WELL_FORMED_XML;
    GO

CREATE CONTRACT [//call/contract]
(
    [//initiator] SENT BY INITIATOR
)
GO

CREATE QUEUE  InitiatorQueue;
GO

CREATE QUEUE  TargetQueue;
GO

CREATE SERVICE InitiatorService
      ON QUEUE InitiatorQueue
(
     [//call/contract]

)

CREATE SERVICE TargetService
ON QUEUE TargetQueue
(
     [//call/contract]

)

GRANT SEND ON SERVICE::[InitiatorService] TO PUBLIC
GO

GRANT SEND ON SERVICE::[TargetService] TO PUBLIC
GO
```

.

## 4.

. **3** . **( )** .

```
USE [MyDatabase]

DECLARE @ch uniqueidentifier = NEWID()
DECLARE @msg XML

BEGIN DIALOG CONVERSATION @ch
    FROM SERVICE [InitiatorService]
    TO SERVICE 'TargetService'
    ON CONTRACT [//call/contract]
    WITH ENCRYPTION = OFF; -- more possible options

        SET @msg = (
                    SELECT 'HelloThere' "elementNum1"
                    FOR XML PATH(''), ROOT('ExampleRoot'), ELEMENTS XSINIL, TYPE
                    );

SEND ON CONVERSATION @ch MESSAGE TYPE [//initiator] (@msg);
END CONVERSATION @ch;
```

TargetQueue msg .

## 5. TargetQueue

. . **3.** **( )** .

.

```
USE [MyDatabase]
GO

SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO

CREATE PROCEDURE [dbo].[p_RecieveMessageFromTargetQueue]

    AS
    BEGIN

    declare
    @message_body xml,
    @message_type_name nvarchar(256),
    @conversation_handle uniqueidentifier,
    @messagetypename nvarchar(256);



   WHILE 1=1
   BEGIN
```

```
    BEGIN TRANSACTION
        WAITFOR(
        RECEIVE TOP(1)
        @message_body = CAST(message_body as xml),
        @message_type_name = message_type_name,
        @conversation_handle = conversation_handle,
        @messagetypename = message_type_name
        FROM DwhInsertSmsQueue
        ), TIMEOUT 1000;

         IF (@@ROWCOUNT = 0)
            BEGIN
                ROLLBACK TRANSACTION
                BREAK
            END

         IF (@messagetypename = '//initiator')
             BEGIN

                IF OBJECT_ID('MyDatabase..MyExampleTableHelloThere') IS NOT NULL
                    DROP TABLE dbo.MyExampleTableHelloThere

                SELECT @message_body.value('(/ExampleRoot/"elementNum1")[1]', 'VARCHAR(50)')
AS MyExampleMessage
                INTO dbo.MyExampleTableHelloThere

             END



         IF (@messagetypename = 'http://schemas.microsoft.com/SQL/ServiceBroker/EndDialog')
            BEGIN
                END CONVERSATION @conversation_handle;
            END

    COMMIT TRANSACTION
   END


END
```

: TargetQueue .

```
USE [MyDatabase]

ALTER QUEUE [dbo].[TargetQueue] WITH STATUS = ON , RETENTION = OFF ,
ACTIVATION
 (  STATUS = ON , --activation status
    PROCEDURE_NAME = dbo.p_RecieveMessageFromTargetQueue , --procedure name
    MAX_QUEUE_READERS = 1 , --number of readers
    EXECUTE AS SELF  )
```

: https://riptutorial.com/ko/sql-server/topic/7651/-

# 75:

- PARSENAME ( 'object_name', object_piece)

| 'object_name' | |
|---|---|
| . object_name sysname. . , , . | . object_piece int . 1 = 2 = 3 = 4 = |

## Examples

```
Declare @ObjectName nVarChar(1000)
Set @ObjectName = 'HeadOfficeSQL1.Northwind.dbo.Authors'

SELECT
 PARSENAME(@ObjectName, 4) as Server
,PARSENAME(@ObjectName, 3) as DB
,PARSENAME(@ObjectName, 2) as Owner
,PARSENAME(@ObjectName, 1) as Object
```

:

| | DB |
|---|---|
| HeadofficeSQL1 | |

| |
|---|
| dbo |

: https://riptutorial.com/ko/sql-server/topic/5775/

# 76:

- DENSE_RANK () OVER ([<partition_by_clause>] <order_by_clause>)
- RANK () OVER ([partition_by_clause] order_by_clause)

| | |
|---|---|
| <partition_by_clause> | FROM DENSE_RANK . PARTITION BY OVER (Transact-SQL) . |
| <order_by_clause> | DENSE_RANK . |
| OVER ( [ partition_by_clause ] order_by_clause) | partition_by_clause FROM partition_by_clause . . order_by_clause . order_by_clause . OVER <rows or range clause> RANK . OVER (Transact-SQL) . |

. , SalesYTD . SalesYTD 2 . . DENSE_RANK .

. 1 .

DENSE_RANK . .

## Examples

### ()

A RANK () .

:

```
Select Studentid,Name,Subject,Marks,
RANK() over(partition by name order by Marks desc)Rank
From Exam
order by name,subject

   Studentid    Name    Subject    Marks    Rank

    101         Ivan    Maths       70       2
    101         Ivan    Science     80       1
    101         Ivan    Social      60       3
    102         Ryan    Maths       60       2
    102         Ryan    Science     50       3
    102         Ryan    Social      70       1
    103         Tanvi   Maths       90       1
    103         Tanvi   Science     90       1
    103         Tanvi   Social      80       3
```

### DENSE_RANK ()

RANK () . .

```
Select  Studentid, Name,Subject,Marks,
DENSE_RANK() over(partition by name order by Marks desc)Rank
From Exam
order by name

Studentid    Name    Subject    Marks    Rank
101          Ivan    Science    80       1
101          Ivan    Maths      70       2
101          Ivan    Social     60       3
102          Ryan    Social     70       1
102          Ryan    Maths      60       2
102          Ryan    Science    50       3
103          Tanvi   Maths      90       1
103          Tanvi   Science    90       1
103          Tanvi   Social     80       2
```

: https://riptutorial.com/ko/sql-server/topic/5031/-

# 77:

## Examples

```
CREATE SCHEMA dvr AUTHORIZATION Owner
    CREATE TABLE sat_Sales (source int, cost int, partid int)
    GRANT SELECT ON SCHEMA :: dvr TO  User1
    DENY SELECT ON SCHEMA :: dvr to User 2
GO
```

```
ALTER SCHEMA dvr
    TRANSFER dbo.tbl_Staging;
GO
```

tbl_Staging dbo dvr .

```
DROP SCHEMA dvr
```

. . / .

: https://riptutorial.com/ko/sql-server/topic/5806/

# 78: - TempDb

## Examples

### TempDb

TempDb . TempDb .

```
SELECT
 SUM (user_object_reserved_page_count)*8 as usr_obj_kb,
 SUM (internal_object_reserved_page_count)*8 as internal_obj_kb,
 SUM (version_store_reserved_page_count)*8  as version_store_kb,
 SUM (unallocated_extent_page_count)*8 as freespace_kb,
 SUM (mixed_extent_page_count)*8 as mixedextent_kb
FROM sys.dm_db_file_space_usage
```

| Attribute | Meaning |
|---|---|
| Higher number of user objects | More usage of Temp tables , cursors or temp variables |
| Higher number of internal objects | Query plan is using a lot of database. Ex: sorting, Group by etc. |
| Higher number of version stores | Long running transaction or high transaction throughput |

### TempDB

TempDB .

```
USE [MASTER]
SELECT * FROM sys.databases WHERE database_id = 2
```

```
USE [MASTER]
SELECT * FROM sys.master_files WHERE database_id = 2
```

DMV TempDb . TempDb .

```
SELECT * FROM sys.dm_db_session_space_usage WHERE session_id = @@SPID
```

- TempDb : https://riptutorial.com/ko/sql-server/topic/4427/----tempdb

# 79:

## Examples

```
CREATE SEQUENCE [dbo].[CustomersSeq]
AS INT
START WITH 10001
INCREMENT BY 1
MINVALUE -1;
```

```
CREATE TABLE [dbo].[Customers]
(
    CustomerID INT DEFAULT (NEXT VALUE FOR [dbo].[CustomersSeq]) NOT NULL,
    CustomerName VARCHAR(100),
);
```

```
INSERT INTO [dbo].[Customers]
       ([CustomerName])
 VALUES
       ('Jerry'),
       ('Gorge')

SELECT * FROM [dbo].[Customers]
```

| ID |
|---|
| 10001 |
| 10002 |

```
DELETE FROM [dbo].[Customers]
WHERE CustomerName = 'Gorge';

INSERT INTO [dbo].[Customers]
       ([CustomerName])
 VALUES ('George')

SELECT * FROM [dbo].[Customers]
```

| ID |
|---|
| 10001 |
| 10003 |

: https://riptutorial.com/ko/sql-server/topic/5324/

# 80:

| | |
|---|---|
| WITH PRIVATE KEY | CREATE CERTIFICATE . (FILE='D:\Temp\CertTest\private.pvk', DECRYPTION BY PASSWORD = 'password'); |

DER . Base64 SQL Server .

```
Msg 15468, Level 16, State 6, Line 1
An error occurred during the generation of the certificate.
```

Base64 OS DER .

---

OS . ' / TDE ' .

---

https://msdn.microsoft.com/en-us/library/ms187798.aspx .

/ TDE . https://msdn.microsoft.com/en-us/library/bb934049.aspx

. https://msdn.microsoft.com/en-us/library/ms188061.aspx

## Examples

```
CREATE CERTIFICATE My_New_Cert
FROM FILE = 'D:\Temp\CertTest\certificateDER.cer'
GO
```

```
SELECT EncryptByCert(Cert_ID('My_New_Cert'),
'This text will get encrypted') encryption_test
```

( ) .

NULL . MS " 512 RSA 53 1024 117 2048 245 " .

EncryptByAsymKey . 2 (16 ) . 1024 58 .

```
USE TDE
CREATE DATABASE ENCRYPTION KEY
WITH ALGORITHM = AES_256
ENCRYPTION BY SERVER CERTIFICATE My_New_Cert
GO

ALTER DATABASE TDE
SET ENCRYPTION ON
GO
```

' '(TDE)

```
-- Create the key and protect it with the cert
CREATE SYMMETRIC KEY My_Sym_Key
WITH ALGORITHM = AES_256
ENCRYPTION BY CERTIFICATE My_New_Cert;
GO

-- open the key
OPEN SYMMETRIC KEY My_Sym_Key
DECRYPTION BY CERTIFICATE My_New_Cert;

-- Encrypt
SELECT EncryptByKey(Key_GUID('SSN_Key_01'), 'This text will get encrypted');
```

```
SELECT EncryptByPassphrase('MyPassPhrase', 'This text will get encrypted')
```

() .

: https://riptutorial.com/ko/sql-server/topic/7096/

# 81:

SQL (Structured Query Language) JOIN . ANSI SQL JOIN .

## Examples

`Inner join` ( `ON` ) / / . `.inner join` .

```
SELECT *
FROM table_1
INNER JOIN table_2
  ON table_1.column_name = table_2.column_name
```

`JOIN` .

```
SELECT *
FROM table_1
JOIN table_2
  ON table_1.column_name = table_2.column_name
```

```
/* Sample data. */
DECLARE @Animal table (
    AnimalId Int IDENTITY,
    Animal Varchar(20)
);

DECLARE @AnimalSound table (
    AnimalSoundId Int IDENTITY,
    AnimalId Int,
    Sound Varchar(20)
);

INSERT INTO @Animal (Animal) VALUES ('Dog');
INSERT INTO @Animal (Animal) VALUES ('Cat');
INSERT INTO @Animal (Animal) VALUES ('Elephant');

INSERT INTO @AnimalSound (AnimalId, Sound) VALUES (1, 'Barks');
INSERT INTO @AnimalSound (AnimalId, Sound) VALUES (2, 'Meows');
INSERT INTO @AnimalSound (AnimalId, Sound) VALUES (3, 'Trumpets');
/* Sample data prepared. */

SELECT
    *
FROM
    @Animal
    JOIN @AnimalSound
        ON @Animal.AnimalId = @AnimalSound.AnimalId;
```

```
AnimalId    Animal              AnimalSoundId AnimalId    Sound
----------- ------------------- ------------- ----------- --------------------
1           Dog                 1             1           Barks
2           Cat                 2             2           Meows
3           Elephant            3             3           Trumpets
```

## ( )

1 2 Table2 1

```
select *
  from Table1 t1
    inner join Table2 t2 on t1.ID_Column = t2.ID_Column
    left  join Table3 t3 on t1.ID_Column = t3.ID_Column
  where t2.column_name = column_value
    and t3.ID_Column is null
  order by t1.column_name;
```

A cross join . . . :

```
SELECT * FROM table_1
CROSS JOIN table_2
```

:

```
/* Sample data. */
DECLARE @Animal table (
    AnimalId Int IDENTITY,
    Animal Varchar(20)
);

DECLARE @AnimalSound table (
    AnimalSoundId Int IDENTITY,
    AnimalId Int,
    Sound Varchar(20)
);

INSERT INTO @Animal (Animal) VALUES ('Dog');
INSERT INTO @Animal (Animal) VALUES ('Cat');
INSERT INTO @Animal (Animal) VALUES ('Elephant');

INSERT INTO @AnimalSound (AnimalId, Sound) VALUES (1, 'Barks');
INSERT INTO @AnimalSound (AnimalId, Sound) VALUES (2, 'Meows');
INSERT INTO @AnimalSound (AnimalId, Sound) VALUES (3, 'Trumpet');
/* Sample data prepared. */

SELECT
    *
FROM
    @Animal
    CROSS JOIN @AnimalSound;
```

:

```
AnimalId    Animal               AnimalSoundId AnimalId    Sound
----------- -------------------- ------------- ----------- --------------------
1           Dog                  1             1           Barks
2           Cat                  1             1           Barks
3           Elephant             1             1           Barks
1           Dog                  2             2           Meows
2           Cat                  2             2           Meows
3           Elephant             2             2           Meows
```

```
1           Dog                 3           3           Trumpet
2           Cat                 3           3           Trumpet
3           Elephant            3           3           Trumpet
```

CROSS JOIN . "" (ANSI SQL-92 ) / .

```
SELECT *
FROM @Animal, @AnimalSound;
```

" " CROSS JOIN .

```
SELECT *
FROM
    @Animal
    JOIN @AnimalSound
        ON 1=1
```

LEFT JOIN ON . ON NULL . LEFT JOIN .

```
SELECT * FROM table_1 AS t1
LEFT JOIN table_2 AS t2 ON t1.ID_Column = t2.ID_Column
```

RIGHT JOIN ON . ON NULL . RIGHT JOIN .

```
SELECT * FROM table_1 AS t1
RIGHT JOIN table_2 AS t2 ON t1.ID_Column = t2.ID_Column
```

FULL JOIN LEFT JOIN RIGHT JOIN FULL JOIN . ON . ON NULL ., NULL . FULL JOIN .

```
SELECT * FROM table_1 AS t1
FULL JOIN table_2 AS t2 ON t1.ID_Column = t2.ID_Column
```

```
/* Sample test data. */
DECLARE @Animal table (
    AnimalId Int IDENTITY,
    Animal Varchar(20)
);

DECLARE @AnimalSound table (
    AnimalSoundId Int IDENTITY,
    AnimalId Int,
    Sound Varchar(20)
);

INSERT INTO @Animal (Animal) VALUES ('Dog');
INSERT INTO @Animal (Animal) VALUES ('Cat');
INSERT INTO @Animal (Animal) VALUES ('Elephant');
INSERT INTO @Animal (Animal) VALUES ('Frog');

INSERT INTO @AnimalSound (AnimalId, Sound) VALUES (1, 'Barks');
INSERT INTO @AnimalSound (AnimalId, Sound) VALUES (2, 'Meows');
INSERT INTO @AnimalSound (AnimalId, Sound) VALUES (3, 'Trumpet');
INSERT INTO @AnimalSound (AnimalId, Sound) VALUES (5, 'Roars');
/* Sample data prepared. */
```

## LEFT OUTER JOIN

```
SELECT *
FROM @Animal As t1
LEFT JOIN @AnimalSound As t2 ON t1.AnimalId = t2.AnimalId;
```

LEFT JOIN

```
AnimalId    Animal               AnimalSoundId AnimalId    Sound
----------- -------------------- ------------- ----------- --------------------
1           Dog                  1             1           Barks
2           Cat                  2             2           Meows
3           Elephant             3             3           Trumpet
4           Frog                 NULL          NULL        NULL
```

```
SELECT *
FROM @Animal As t1
RIGHT JOIN @AnimalSound As t2 ON t1.AnimalId = t2.AnimalId;
```

RIGHT JOIN

```
AnimalId    Animal               AnimalSoundId AnimalId    Sound
----------- -------------------- ------------- ----------- --------------------
1           Dog                  1             1           Barks
2           Cat                  2             2           Meows
3           Elephant             3             3           Trumpet
NULL        NULL                 4             5           Roars
```

```
SELECT *
FROM @Animal As t1
FULL JOIN @AnimalSound As t2 ON t1.AnimalId = t2.AnimalId;
```

FULL JOIN

```
AnimalId    Animal               AnimalSoundId AnimalId    Sound
----------- -------------------- ------------- ----------- --------------------
1           Dog                  1             1           Barks
2           Cat                  2             2           Meows
3           Elephant             3             3           Trumpet
4           Frog                 NULL          NULL        NULL
NULL        NULL                 4             5           Roars
```

UPDATE   .

```
CREATE TABLE Users (
    UserId int NOT NULL,
    AccountId int NOT NULL,
    RealName nvarchar(200) NOT NULL
)

CREATE TABLE Preferences (
    UserId int NOT NULL,
    SomeSetting bit NOT NULL
```

```
)
```

Users SomeSetting Preferences SomeSetting .

```
UPDATE p
SET p.SomeSetting = 1
FROM Users u
JOIN Preferences p ON u.UserId = p.UserId
WHERE u.AccountId = 1234
```

p . FROM Preferences . Users AccountId .

```
Update t
SET  t.Column1=100
FROM Table1 t LEFT JOIN Table12 t2
ON t2.ID=t.ID
```

```
UPDATE t1
SET t1.field1 = t2.field2Sum
FROM table1 t1
INNER JOIN (select field3, sum(field2) as field2Sum
from table2
group by field3) as t2
on t2.field3 = t1.field3
```

/ ( : Count, Avg, Max Min) / . , ID / .

. Buy Orders PurchaseOrderLineItems ( ) . ID .

```
SELECT po.Id, po.PODate, po.VendorName, po.Status, item.ItemNo,
  item.Description, item.Cost, item.Price
FROM PurchaseOrders po
LEFT JOIN
     (
       SELECT l.PurchaseOrderId, l.ItemNo, l.Description, l.Cost, l.Price, Max(l.id) as Id
       FROM PurchaseOrderLineItems l
       GROUP BY l.PurchaseOrderId, l.ItemNo, l.Description, l.Cost, l.Price
     ) AS item ON item.PurchaseOrderId = po.Id
```

(self join) . .

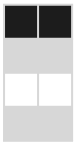
Employees .

| | | _ID |
|---|---|---|
| 1 | | |
| 2 | | 1 |
| | | 2 |

Boss_ID ID . . ( Bosses .

```
SELECT Employees.Name,
    Bosses.Name AS Boss
FROM Employees
INNER JOIN Employees AS Bosses
    ON Employees.Boss_ID = Bosses.ID
```

.



DELETE . .

```
CREATE TABLE Users (
    UserId int NOT NULL,
    AccountId int NOT NULL,
    RealName nvarchar(200) NOT NULL
)

CREATE TABLE Preferences (
    UserId int NOT NULL,
    SomeSetting bit NOT NULL
)
```

Preferences Users .

```
DELETE p
FROM Users u
INNER JOIN Preferences p ON u.UserId = p.UserId
WHERE u.AccountId = 1234
```

p FROM Preferences Users AccountId .

.

```
Table People
PersonID FirstName
       1 Alice
       2 Bob
       3 Eve

Table Scores
PersonID Subject Score
       1 Math    100
       2 Math     54
       2 Science  98
```

:

```
Select * from People a
left join Scores b
on a.PersonID = b.PersonID
```

:

```
PersonID FirstName PersonID Subject Score
       1 Alice            1 Math    100
       2 Bob              2 Math     54
       2 Bob              2 Science  98
       3 Eve           NULL NULL   NULL
```

:

```
Select * from People a
left join Scores b
on a.PersonID = b.PersonID
where Subject = 'Math'
```

`Subject  NULL` Bob Eve .

`People` :

```
Select * from People a
left join Scores b
on a.PersonID = b.PersonID
and b.Subject = 'Math'
```

: https://riptutorial.com/ko/sql-server/topic/1008/

# 82:

INSERT INTO .

## Examples

### Hello World INTO

```
CREATE TABLE MyTableName
(
    Id INT,
    MyColumnName NVARCHAR(1000)
)
GO

INSERT INTO MyTableName (Id, MyColumnName)
VALUES (1, N'Hello World!')
GO
```

### INSERT

( ) .

```
INSERT INTO USERS (FIRST_NAME, LAST_NAME)
VALUES ('Stephen', 'Jiang');
```

nullable, ID, . . Null , . () .

SQL Server 2008 :

```
INSERT INTO USERS VALUES
(2, 'Michael', 'Blythe'),
(3, 'Linda', 'Mitchell'),
(4, 'Jillian', 'Carson'),
(5, 'Garrett', 'Vargas');
```

SQL Server "UNION ALL" .

```
INSERT INTO USERS (FIRST_NAME, LAST_NAME)
SELECT 'James', 'Bond' UNION ALL
SELECT 'Miss', 'Moneypenny' UNION ALL
SELECT 'Raoul', 'Silva'
```

INSERT "INTO" . SQL Server INSERT 1000 .

.

```
INSERT INTO USERS(Id, FirstName, LastName)
VALUES (1, 'Mike', 'Jones');
```

```
INSERT INTO USERS
VALUES (1, 'Mike', 'Jones');
```

insert .

```
INSERT INTO USERS(FirstName, LastName, Id)
VALUES ('Mike', 'Jones', 1);
```

## ID OUTPUT .

INSERT `OUTPUT INSERTED.ColumnName` ( : ID) . `IDENTITY` .

( : ADO.net) `SELECT` .

```
-- CREATE TABLE OutputTest ([Id] INT NOT NULL PRIMARY KEY IDENTITY, [Name] NVARCHAR(50))

INSERT INTO OutputTest ([Name])
OUTPUT INSERTED.[Id]
VALUES ('Testing')
```

ID

```
-- CREATE a table variable having column with the same datatype of the ID

DECLARE @LastId TABLE ( id int);

INSERT INTO OutputTest ([Name])
OUTPUT INSERTED.[Id] INTO @LastId
VALUES ('Testing')

SELECT id FROM @LastId

-- We can set the value in a variable and use later in procedure

DECLARE @LatestId int = (SELECT id FROM @LastId)
```

## SELECT INSERT

SQL ( )

```
INSERT INTO Table_name (FirstName, LastName, Position)
SELECT FirstName, LastName, 'student' FROM Another_table_name
```

SELECT 'student' .

: https://riptutorial.com/ko/sql-server/topic/3814/--

# 83:

SQL Server Agent SQL Server . . , . SQL Server .

## Examples

- sp_add_job .

```
USE msdb ;
GO
EXEC dbo.sp_add_job
@job_name = N'Weekly Job' ;  -- the job name
```

- sp_add_jobStep .

```
EXEC sp_add_jobstep
@job_name = N'Weekly Job',  -- Job name to add a step
@step_name = N'Set database to read only',  -- step name
@subsystem = N'TSQL',  -- Step type
@command = N'ALTER DATABASE SALES SET READ_ONLY',   -- Command
@retry_attempts = 5,  --Number of attempts
@retry_interval = 5 ; -- in minutes
```

- ```
  EXEC dbo.sp_add_jobserver
  @job_name = N'Weekly Sales Data Backup',
  @server_name = 'MyPC\data;   -- Default is LOCAL
  GO
  ```

### SQL

sp_add_schedule .

```
USE msdb
GO

EXEC sp_add_schedule
    @schedule_name = N'NightlyJobs' ,  -- specify the schedule name
    @freq_type = 4,   -- A value indicating when a job is to be executed (4) means Daily
    @freq_interval = 1,  -- The days that a job is executed and depends on the value of
`freq_type`.
    @active_start_time = 010000 ;   -- The time on which execution of a job can begin
GO
```

sp_add_schedule . .

SQL sp_attach_schedule .

```
-- attaches the schedule to the job BackupDatabase
EXEC sp_attach_schedule
   @job_name = N'BackupDatabase',  -- The job name to attach with
   @schedule_name = N'NightlyJobs' ;  -- The schedule name
```
```

```
GO
```

: https://riptutorial.com/ko/sql-server/topic/5329/---

# 84:

## Examples

/

. () . () . FOREIGN KEY REFERENCES .

CompanyId Company ID Employee .

```
create table Company (
   CompanyId int primary key,
   Name nvarchar(200)
)
create table Employee (
    EmployeeId int,
    Name nvarchar(200),
    CompanyId int
        foreign key references Company(companyId)
)
```

Employee.CompanyId Company.CompanyId . ID .

FOREIGN KEY " " .

/

companyId 1 . companyId 1 .

```
insert into Employee values (17, 'John', 1)
```

CompanyId .

```
insert into Employee values (17, 'John', 111111)
```

547, 16, 0, 12 INSERT FOREIGN KEY "FK__Employee__Compan__1EE485AA". "MyDb", "dbo.Company", 'CompanyId' . .

employee s h .

```
delete from company where CompanyId = 1
```

547, 16, 0, 14 DELETE REFERENCE "FK__Employee__Compan__1EE485AA" . "MyDb", "dbo.Employee", 'CompanyId' . .

" " .

**FOREIGN KEY** . Employee table CompanyId Company Employee . ALTER TABLE

.

```
alter table Employee
    add  foreign key (CompanyId) references Company(CompanyId)
```

**FOREIGN KEY** . Employee CompanyId Company Employee . ALTER TABLE .

```
alter table Employee
    add CompanyId int foreign key references Company(CompanyId)
```

sys.foreignkeys .

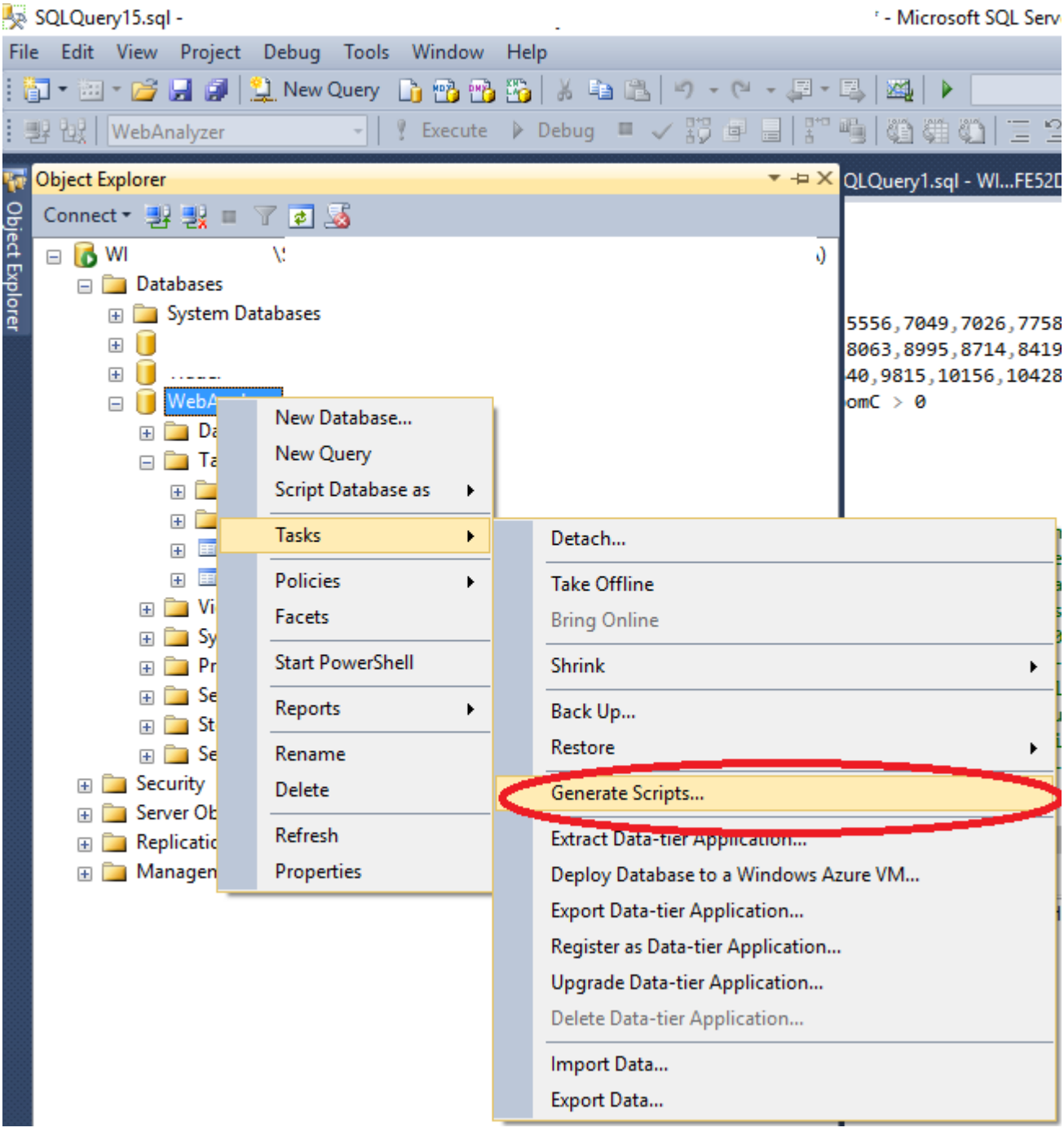
```
select name,
 OBJECT_NAME(referenced_object_id) as [parent table],
 OBJECT_NAME(parent_object_id) as [child table],
 delete_referential_action_desc,
 update_referential_action_desc
from sys.foreign_keys
```

: https://riptutorial.com/ko/sql-server/topic/5355/-

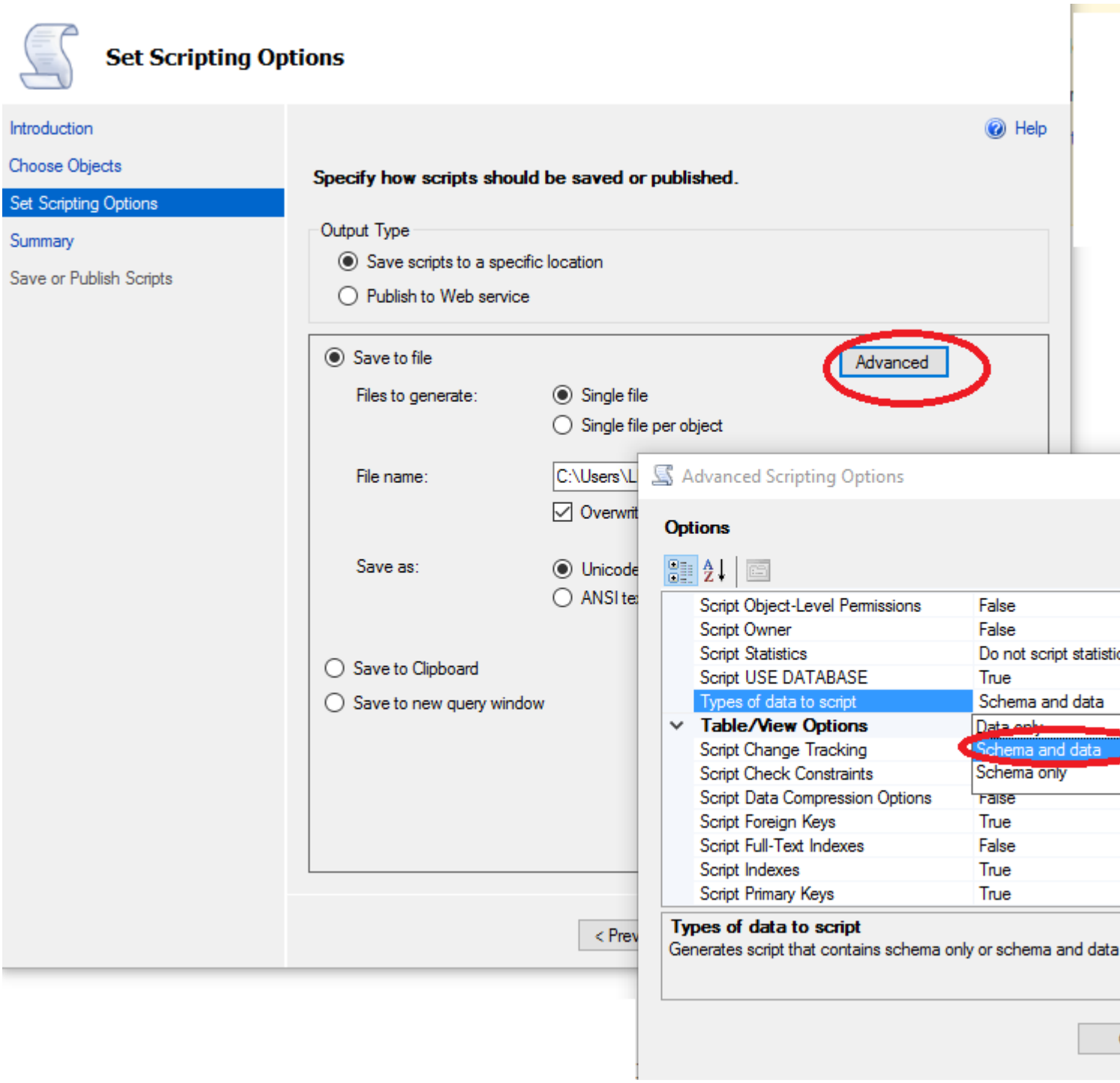
# 85:

## Examples

1. □□□ □□□ -> Tasks -> Generate Scripts...




2. Next Next Advanced Types of data to script Schema and data ( )




3. `Next` `Finish` `.sql` .

4. `.sql` .

: https://riptutorial.com/ko/sql-server/topic/4451/

# 86:

## Examples

.

```
EXEC sp_helpserver;
```

```
sp_who2
```

SQL Server . , .

sp_who2 .

```
-- Create a variable table to hold the results of sp_who2 for querying purposes

DECLARE @who2 TABLE (
      SPID INT NULL,
      Status VARCHAR(1000) NULL,
      Login SYSNAME NULL,
      HostName SYSNAME NULL,
      BlkBy SYSNAME NULL,
      DBName SYSNAME NULL,
      Command VARCHAR(8000) NULL,
      CPUTime INT NULL,
      DiskIO INT NULL,
      LastBatch VARCHAR(250) NULL,
      ProgramName VARCHAR(250) NULL,
      SPID2 INT NULL, -- a second SPID for some reason...?
      REQUESTID INT NULL
)

INSERT INTO @who2
EXEC sp_who2

SELECT    *
FROM    @who2 w
WHERE    1=1
```

:

```
-- Find specific user sessions:
SELECT  *
FROM    @who2 w
WHERE   1=1
    and  login = 'userName'

-- Find longest CPUTime queries:
SELECT  top 5 *
FROM    @who2 w
WHERE   1=1
order   by CPUTime desc
```

```
SELECT    SERVERPROPERTY('ProductVersion') AS ProductVersion,
          SERVERPROPERTY('ProductLevel') AS ProductLevel,
          SERVERPROPERTY('Edition') AS Edition,
          SERVERPROPERTY('EngineEdition') AS EngineEdition;
```

```
SELECT  DATEDIFF(DAY, login_time, getdate()) UpDays
FROM    master..sysprocesses
WHERE   spid = 1
```

## SQL Server

SQL Server , .

```
SELECT    SERVERPROPERTY('MachineName') AS Host,
          SERVERPROPERTY('InstanceName') AS Instance,
          DB_NAME() AS DatabaseContext,
          SERVERPROPERTY('Edition') AS Edition,
          SERVERPROPERTY('ProductLevel') AS ProductLevel,
          CASE SERVERPROPERTY('IsClustered')
            WHEN 1 THEN 'CLUSTERED'
            ELSE 'STANDALONE' END AS ServerType,
          @@VERSION AS VersionNumber;
```

, ,  .

### DB  sp

```
SELECT execquery.last_execution_time AS [Date Time], execsql.text AS [Script]
FROM sys.dm_exec_query_stats AS execquery
CROSS APPLY sys.dm_exec_sql_text(execquery.sql_handle) AS execsql
ORDER BY execquery.last_execution_time DESC
```

```
SELECT o.type_desc AS ROUTINE_TYPE,o.[name] AS ROUTINE_NAME,
m.definition AS ROUTINE_DEFINITION
FROM sys.sql_modules AS m INNER JOIN sys.objects AS o
ON m.object_id = o.object_id WHERE m.definition LIKE '%Keyword%'
order by ROUTINE_NAME
```

```
SELECT t.name AS table_name,
SCHEMA_NAME(schema_id) AS schema_name,
c.name AS column_name
FROM sys.tables AS t
INNER JOIN sys.columns c ON t.OBJECT_ID = c.OBJECT_ID
where c.name like 'Keyword%'
ORDER BY schema_name, table_name;
```

```
WITH LastRestores AS
(
SELECT
    DatabaseName = [d].[name] ,
    [d].[create_date] ,
    [d].[compatibility_level] ,
```

```
    [d].[collation_name] ,
    r.*,
    RowNum = ROW_NUMBER() OVER (PARTITION BY d.Name ORDER BY r.[restore_date] DESC)
FROM master.sys.databases d
LEFT OUTER JOIN msdb.dbo.[restorehistory] r ON r.[destination_database_name] = d.Name
)
SELECT *
FROM [LastRestores]
WHERE [RowNum] = 1
```

```
select top 100 * from databaselog
Order by Posttime desc
```

### SP

```
SELECT name, create_date, modify_date
FROM sys.objects
WHERE type = 'P'
Order by modify_date desc
```

: https://riptutorial.com/ko/sql-server/topic/2029/---

# 87:

SQL Server 2016 .

SQL Server 2016 . ( : ) . C C datetime2 . . .

## Examples

```
CREATE TABLE dbo.Employee
(
  [EmployeeID] int NOT NULL PRIMARY KEY CLUSTERED
  , [Name] nvarchar(100) NOT NULL
  , [Position] varchar(100) NOT NULL
  , [Department] varchar(100) NOT NULL
  , [Address] nvarchar(1024) NOT NULL
  , [AnnualSalary] decimal (10,2) NOT NULL
  , [ValidFrom] datetime2 (2) GENERATED ALWAYS AS ROW START
  , [ValidTo] datetime2 (2) GENERATED ALWAYS AS ROW END
  , PERIOD FOR SYSTEM_TIME (ValidFrom, ValidTo)
 )
 WITH (SYSTEM_VERSIONING = ON (HISTORY_TABLE = dbo.EmployeeHistory));
```

**INSERTS : INSERT,** (UTC ) **ValidFrom** , 9999- **ValidTo** 12-31. .

**UPDATES : UPDATE** **ValidTo** (UTC ) . . **ValidFrom** (UTC ) . **ValidTo** 9999-12-31 .

**DELETE,** , (UTC ) **ValidTo** . . . . u .

**MERGE** : **MERGE** **MERGE** ( **INSERT** , **UPDATE** / **DELETE** ) .

**:** datetime2 . , **SYSTEM_TIME** UTC .

### ?

```
SELECT * FROM Employee
    FOR SYSTEM_TIME
        BETWEEN '2014-01-01 00:00:00.0000000' AND '2015-01-01 00:00:00.0000000'
            WHERE EmployeeID = 1000 ORDER BY ValidFrom;
```

### (FOR SYSTEM_TIME AS OF )

() .

```
SELECT * FROM Employee
    FOR SYSTEM_TIME AS  OF '2016-08-06 08:32:37.91'
```

### SYSTEM_TIME

<end_date_time> FOR SYSTEM_TIME FROM <start_date_time> TO <end_date_time> .

```
SELECT * FROM Employee
    FOR SYSTEM_TIME BETWEEN  '2015-01-01' AND '2015-12-31'
```

## FOR SYSTEM_TIME FROM

FROM <start_date_time> <end_date_time> . TO . . FROM TO .

```
SELECT * FROM Employee
    FOR SYSTEM_TIME FROM '2015-01-01' TO '2015-12-31'
```

## FOR SYSTEM_TIME IN ( , )

CONTAINED IN datetime . .

```
SELECT * FROM Employee
    FOR SYSTEM_TIME CONTAINED IN ('2015-04-01', '2015-09-25')
```

## FOR SYSTEM_TIME ALL

.

```
SELECT * FROM Employee
    FOR SYSTEM_TIME ALL
```

## SQL Server

. .

```
CREATE SCHEMA History
GO
CREATE TABLE dbo.Department
(
    DepartmentNumber char(10) NOT NULL PRIMARY KEY NONCLUSTERED,
    DepartmentName varchar(50) NOT NULL,
    ManagerID int  NULL,
    ParentDepartmentNumber char(10) NULL,
    SysStartTime datetime2 GENERATED ALWAYS AS ROW START HIDDEN NOT NULL,
    SysEndTime datetime2 GENERATED ALWAYS AS ROW END HIDDEN NOT NULL,
    PERIOD FOR SYSTEM_TIME (SysStartTime,SysEndTime)
)
WITH
    (
        MEMORY_OPTIMIZED = ON, DURABILITY = SCHEMA_AND_DATA,
        SYSTEM_VERSIONING = ON ( HISTORY_TABLE = History.DepartmentHistory )
    );
```

**SQL Server** . , .

```
ALTER TABLE dbo.Employee
```

SET (SYSTEM_VERSIONING = OFF);

:

```
    DELETE FROM dbo.EmployeeHistory
```

EndTime <= '2017-01-26 14:00:29';

.

```
ALTER TABLE dbo.Employee
```

SET (SYSTEM_VERSIONING = ON (HISTORY_TABLE = [dbo]. [EmployeeHistory], DATA_CONSISTENCY_CHECK = ON));

Azure SQL Azure SQL . , .

```
ALTER DATABASE CURRENT
```

GO TEMPORAL_HISTORY_RETENTION ON GO

.

```
ALTER TABLE dbo.Employee
```

SET (SYSTEM_VERSIONING = ON (HISTORY_RETENTION_PERIOD = 90 ));

90 . SQL Server 2016 - TEMPORAL_HISTORY_RETENTION HISTORY_RETENTION_PERIOD SQL Server 2016 - .

TEMPORAL_HISTORY_RETENTION .

```
Msg 102, Level 15, State 6, Line 34
```

'TEMPORAL_HISTORY_RETENTION' .

HISTORY_RETENTION_PERIOD :

```
Msg 102, Level 15, State 1, Line 39
```

'HISTORY_RETENTION_PERIOD' .

: https://riptutorial.com/ko/sql-server/topic/5296/-

# 88:

SQL Server . . / .

- CREATE {PROCEDURE | PROC} [schema_name.]
- [@parameter [type_schema_name.]
- [VARYING] [= default] [OUT | | READONLY]
- , @parameter [type_schema_name.]
- [VARYING] [= default] [OUT | | READONLY]]
- [WITH { | RECOMPILE | EXECUTE AS }]
- [ ]
-
-
- [ _ ]
- executable_section
- ;

## Examples

Authors

```
CREATE PROCEDURE GetName
(
    @input_id INT = NULL,      --Input parameter,  id of the person, NULL default
    @name VARCHAR(128) = NULL  --Input parameter, name of the person, NULL default
)
AS
BEGIN
    SELECT Name + ' is from ' + Country
    FROM Authors
    WHERE Id = @input_id OR Name = @name
END
GO
```

. EXECUTE EXEC .

```
EXECUTE GetName @id = 1
EXEC Getname @name = 'Ernest Hemingway'
```

EXEC . .

```
GetName NULL, 'Ernest Hemingway'
```

.

```
 CREATE PROCEDURE dbo.sProcTemp
 (
    @Param1 INT,
    @Param2 INT
```

```
)
AS
BEGIN

    SELECT
        Param1 = @Param1,
        Param2 = @Param2

END
```

@ Param1 @ Param2 . .

```
  EXEC dbo.sProcTemp @Param1 = 0,@Param2=1
```

.

```
  EXEC dbo.sProcTemp @Param2 = 0,@Param1=1
```

, @ param2 @ Param1 . , . .

sp_ . procuedres SQL Server . "sp_" SQL Server master . master . "sp_" .

```
Use Master

CREATE PROCEDURE sp_GetName
(
    @input_id INT = NULL,      --Input parameter,  id of the person, NULL default
    @name VARCHAR(128) = NULL  --Input parameter, name of the person, NULL default
)
AS
BEGIN
    SELECT Name + ' is from ' + Country
    FROM Authors
    WHERE Id = @input_id OR Name = @name
END
GO
```

## OUT

OUTPUT .

---

```
CREATE PROCEDURE SprocWithOutParams
(
    @InParam VARCHAR(30),
    @OutParam VARCHAR(30) OUTPUT
)
AS
BEGIN
    SELECT @OutParam = @InParam + ' must come out'
    RETURN
END
GO
```

```
DECLARE @OutParam VARCHAR(30)
EXECUTE SprocWithOutParams 'what goes in', @OutParam OUTPUT
PRINT @OutParam
```

# out

```
CREATE PROCEDURE SprocWithOutParams2
(
    @InParam VARCHAR(30),
    @OutParam VARCHAR(30) OUTPUT,
    @OutParam2 VARCHAR(30) OUTPUT
)
AS
BEGIN
    SELECT @OutParam = @InParam +' must come out'
    SELECT @OutParam2 = @InParam +' must come out'
    RETURN
END
GO
```

```
DECLARE @OutParam VARCHAR(30)
DECLARE @OutParam2 VARCHAR(30)
EXECUTE SprocWithOutParams2 'what goes in', @OutParam OUTPUT, @OutParam2 OUTPUT
PRINT @OutParam
PRINT @OutParam2
```

## If ... Else  Insert into Operation

`Employee` :

```
CREATE TABLE Employee
(
    Id INT,
    EmpName VARCHAR(25),
    EmpGender VARCHAR(6),
    EmpDeptId INT
)
```

null  Employee  .

```
CREATE PROCEDURE spSetEmployeeDetails
(
    @ID int,
    @Name VARCHAR(25),
    @Gender VARCHAR(6),
    @DeptId INT
)
AS
BEGIN
```

```
    IF (
        (@ID IS NOT NULL AND LEN(@ID) !=0)
        AND (@Name IS NOT NULL AND LEN(@Name) !=0)
        AND (@Gender IS NOT NULL AND LEN(@Gender) !=0)
        AND (@DeptId IS NOT NULL AND LEN(@DeptId) !=0)
    )
    BEGIN
        INSERT INTO Employee
        (
            Id,
            EmpName,
            EmpGender,
            EmpDeptId
        )
        VALUES
        (
            @ID,
            @Name,
            @Gender,
            @DeptId
        )
    END
ELSE
    PRINT 'Incorrect Parameters'
END
GO
```

```
DECLARE @ID INT,
    @Name VARCHAR(25),
    @Gender VARCHAR(6),
    @DeptId INT

EXECUTE spSetEmployeeDetails
    @ID = 1,
    @Name = 'Subin Nepal',
    @Gender = 'Male',
    @DeptId = 182666
```

## SQL

SQL SQL . SQL SQL .

SQL :

```
CREATE PROC sp_dynamicSQL
@table_name      NVARCHAR(20),
@col_name        NVARCHAR(20),
@col_value       NVARCHAR(20)
AS
BEGIN
DECLARE  @Query  NVARCHAR(max)
SET      @Query = 'SELECT * FROM ' + @table_name
SET      @Query = @Query + ' WHERE ' + @col_name + ' = ' + ''''+@col_value+''''
EXEC     (@Query)
END
```

SQL `@table_name, @col_name, and @col_value` `@table_name, @col_name, and @col_value` . .

. SQL . .

```
DECLARE @table_name     NVARCHAR(20) = 'ITCompanyInNepal',
        @col_name       NVARCHAR(20) = 'Headquarter',
        @col_value      NVARCHAR(20) = 'USA'

EXEC    sp_dynamicSQL   @table_name,
                        @col_name,
                        @col_value
```

| ID | CompanyName | CompanyAddress | Headquarter | NumberOfEmployee |
|---|---|---|---|---|
| 1 | CompanyOne | Kathmandu | USA | 300 |
| 2 | CompanyTwo | Kathmandu | USA | 260 |
| 3 | CompanyThree | Kathmandu | Nepal | 300 |
| 4 | CompanyFour | Kathmandu | Nepal | 180 |
| 6 | CompanySix | Janakpur | USA | 50 |
| 7 | CompanySeven | Janakpur | Australia | 100 |
| 8 | CompanyEight | Birganj | Australia | 150 |
| 9 | CompanyNine | Biratnagar | Canada | 200 |
| 10 | CompanyTen | Pokhara | India | 85 |

| ID | CompanyName | CompanyAddress | Headquarter | NumberOfEmployee |
|---|---|---|---|---|
| 1 | CompanyOne | Kathmandu | USA | 300 |
| 2 | CompanyTwo | Kathmandu | USA | 260 |
| 6 | CompanySix | Janakpur | USA | 50 |
| 1 | CompanyA | Banglore | USA | 400 |
| 2 | CompanyB | Banglore | USA | 450 |

`#systables` ad `#systables` .

```
select
    o.name,
    row_number() over (order by o.name) as rn
into
    #systables
from
    sys.objects as o
where
    o.type = 'S'
```

.

```
declare
    @rn int = 1,
    @maxRn int = (
                    select
                        max(rn)
                    from
                        #systables as s
                    )
declare    @tablename sys name
```

. `select @rn @rn` ex `set @rn = @rn + 1` . . `@rn` `#systables` . .

```
while @rn <= @maxRn
    begin

        select
            @tablename = name,
            @rn = @rn + 1
        from
            #systables as s
        where
            s.rn = @rn

        print @tablename
    end
```

```
CREATE PROCEDURE SprocWithSimpleLoop
(
    @SayThis VARCHAR(30),
    @ThisManyTimes INT
)
AS
BEGIN
    WHILE @ThisManyTimes > 0
    BEGIN
        PRINT @SayThis;
        SET @ThisManyTimes = @ThisManyTimes - 1;
    END

    RETURN;
END
GO
```

: https://riptutorial.com/ko/sql-server/topic/3213/--

# 89:

## Examples

### A. ,

AdventureWorks2012 HumanResources.JobCandidate JobCandidateID . ft . ft Resume .

```
USE AdventureWorks2012;
GO
CREATE UNIQUE INDEX ui_ukJobCand ON HumanResources.JobCandidate(JobCandidateID);
CREATE FULLTEXT CATALOG ft AS DEFAULT;
CREATE FULLTEXT INDEX ON HumanResources.JobCandidate(Resume)
   KEY INDEX ui_ukJobCand
   WITH STOPLIST = SYSTEM;
GO
```

https://www.simple-talk.com/sql/learn-sql-server/understanding-full-text-indexing-in-sql-server/

https://msdn.microsoft.com/en-us/library/cc879306.aspx

https://msdn.microsoft.com/en-us/library/ms142571.aspx

```
USE AdventureWorks2012;
GO
CREATE FULLTEXT CATALOG production_catalog;
GO
CREATE FULLTEXT INDEX ON Production.ProductReview
 (
  ReviewerName
     Language 1033,
  EmailAddress
     Language 1033,
  Comments
     Language 1033
 )
  KEY INDEX PK_ProductReview_ProductReviewID
      ON production_catalog;
GO
```

```
USE AdventureWorks2012;
GO
CREATE FULLTEXT INDEX ON Production.Document
  (
  Title
      Language 1033,
  DocumentSummary
      Language 1033,
  Document
      TYPE COLUMN FileExtension
      Language 1033
  )
```

```
  KEY INDEX PK_Document_DocumentID
         WITH STOPLIST = SYSTEM, SEARCH PROPERTY LIST = DocumentPropertyList, CHANGE_TRACKING
OFF, NO POPULATION;
   GO
```

```
ALTER FULLTEXT INDEX ON Production.Document SET CHANGE_TRACKING AUTO;
GO
```

```
SELECT product_id
FROM products
WHERE CONTAINS(product_description, "Snap Happy 100EZ" OR FORMSOF(THESAURUS,'Snap Happy') OR
'100EZ')
AND product_cost < 200 ;


SELECT candidate_name,SSN
FROM candidates
WHERE CONTAINS(candidate_resume,"SQL Server") AND candidate_division =DBA;
```

https://msdn.microsoft.com/en-us/library/ms142571.aspx .

: https://riptutorial.com/ko/sql-server/topic/4557/--

# 90:

. . .

## Examples

```
CREATE VIEW dbo.PersonsView
AS
SELECT
    name,
    address
FROM persons;
```

.

```
IF OBJECT_ID('dbo.PersonsView', 'V') IS NOT NULL
    DROP VIEW dbo.PersonsView
GO

CREATE VIEW dbo.PersonsView
AS
SELECT
    name,
    address
FROM persons;
```

SCHEMABINDING , h . , .

```
CREATE VIEW dbo.PersonsView
WITH SCHEMABINDING
AS
SELECT
    name,
    address
FROM dbo.PERSONS  -- database schema must be specified when WITH SCHEMABINDING is present
```

. . sp_refreshview .

: https://riptutorial.com/ko/sql-server/topic/5327/

# 91:

ORDER BY .

**ORDER BY .**

ORDER BY MSDN . https://msdn.microsoft.com/en-us/library/ms188385.aspx

## Examples

### ORDER BY

Employees Id, FName LName () LName .

```
SELECT Id, FName, LName FROM Employees
ORDER BY LName
```

:

| | FName | LName |
|---|---|---|
| 2 | | |
| 1 | | |
| 4 | | |
| | | |

field DESC . , LName .

```
SELECT Id, FName, LName FROM Employees
ORDER BY LName DESC
```

`ORDER BY` ASCending DESCending .

, http://stackoverflow.com/documentation/sql/280/example-databases/1207/item-sales-table#t=20160721131406643421 1 SaleDate . .

```
SELECT ItemId, SaleDate, Quantity
FROM [Item Sales]
ORDER BY SaleDate ASC, Quantity DESC
```

`ASC` .

### ORDER BY

`CASE ORDER BY` . 1 2 .

| | FName | LName | | ID | DepartmentId | | HireDate |
|---|---|---|---|---|---|---|---|
| 1 | | | 1234567890 | | 1 | 1000 | 01-01-2002 |
| 2 | | | 2468101214 | 1 | 1 | 400 | 23-03-2005 |
| | | | 1357911131 | 1 | 2 | 600 | 12-05-2009 |
| 4 | | | 1212121212 | 2 | 1 | 500 | 24-07-2016 |
| 5 | | | 1372141312 | 2 | 2 | 400 | 25-03-2015 |

```
The following query will provide the required results:
SELECT Id, FName, LName, Salary FROM Employees
ORDER BY Case When DepartmentId = 1 then LName else Salary end
```

/ case .

order by Group :

| | |
|---|---|
| . | 6 |
| | 4 |
| | 10 |

order by case group when 'Total' then 1 when 'Retired' then 2 else 3 end :

| | |
|---|---|
| | 10 |
| | 4 |
| . | 6 |

: https://riptutorial.com/ko/sql-server/topic/4149/

# 92:

SQL Server .

- AVG ([ALL | DISTINCT] )
- COUNT ([ALL | DISTINCT] )
- MAX ([ALL | DISTINCT] )
- MIN ([ALL | DISTINCT] )
- SUM ([ALL | DISTINCT] )

## Examples

### ()

.

. *Marksheet* .

| SubjectCode | SubjectName | MarksObtained |
|---|---|---|
| 101 | Physics | 87 |
| 102 | Chemistry | 75 |
| 103 | Maths | 85 |
| 104 | English | 89 |
| 105 | Computer | 95 |

```
Select SUM(MarksObtained) From Marksheet
```

`sum` NULL .

:

```
106    Italian    NULL
```

.

### AVG ()

.

. *Marksheet* .

| SubjectCode | SubjectName | MarksObtained |
|---|---|---|
| 101 | Physics | 87 |
| 102 | Chemistry | 75 |
| 103 | Maths | 85 |
| 104 | English | 89 |
| 105 | Computer | 95 |

```
Select AVG(MarksObtained) From Marksheet
```

`average` NULL .

:

```
106    Italian    NULL
```

.

## MAX ()

.

. *Marksheet* .

| SubjectCode | SubjectName | MarksObtained |
|---|---|---|
| 101 | Physics | 87 |
| 102 | Chemistry | 75 |
| 103 | Maths | 85 |
| 104 | English | 89 |
| 105 | Computer | 95 |

```
Select MAX(MarksObtained) From Marksheet
```

## MIN ()

.

. *Marksheet* .

| SubjectCode | SubjectName | MarksObtained |
|---|---|---|
| 101 | Physics | 87 |
| 102 | Chemistry | 75 |
| 103 | Maths | 85 |
| 104 | English | 89 |
| 105 | Computer | 95 |

```
Select MIN(MarksObtained) From Marksheet
```

## ()

.

. *Marksheet* .

| SubjectCode | SubjectName | MarksObtained |
|---|---|---|
| 101 | Physics | 87 |
| 102 | Chemistry | 75 |
| 103 | Maths | 85 |
| 104 | English | 89 |
| 105 | Computer | 95 |

```
Select COUNT(MarksObtained) From Marksheet
```

`count` NULL . count * ( ) NULL .

:

```
106    Italian    NULL
```

.

`COUNT(*)` . `COUNT(1)` (Null) .

```
Select COUNT(1) From Marksheet
```

### GROUP BY Column_Name COUNT (Column_Name)

.

:

| ReportName | ReportPrice |
|---|---|
| | 10.00 $ |
| | 10.00 $ |
| | 10.00 $ |
| 2 | 11.00 $ |
| | 10.00 $ |
| 3 | 14.00 $ |
| 3 | 14.00 $ |

| ReportName | ReportPrice |
|---|---|
| 4 | 100.00 $ |

```
SELECT
    ReportName AS REPORT NAME,
    COUNT(ReportName) AS COUNT
FROM
    REPORTS
GROUP BY
    ReportName
```

| | |
|---|---|
| | 4 |
| 2 | 1 |
| 3 | 2 |
| 4 | 1 |

: https://riptutorial.com/ko/sql-server/topic/5802/-

# 93:

SQL Server

- SELECT * FROM TableName ORDER BY id OFFSET 10 ROWS FETCH  10 ;

## Examples

### ROW_NUMBER

SQL Server 2008

`ROW_NUMBER`  `BETWEEN`  " . : 1  1-10,  2  11-20,  3  21-30 .

```
WITH data
AS
(
    SELECT ROW_NUMBER() OVER (ORDER BY name) AS row_id,
        object_id,
        name,
        type,
        create_date
    FROM sys.objects
)
SELECT *
FROM data
WHERE row_id BETWEEN 41 AND 50
```

: `WHERE  ROW_NUMBER`  .

```
SELECT object_id,
    name,
    type,
    create_date
FROM sys.objects
WHERE ROW_NUMBER() OVER (ORDER BY name) BETWEEN 41 AND 50
```

SQL Server  .

4108,  15,  1,  6

SELECT  ORDER BY  .

### OFFSET FETCH

SQL Server 2012

`OFFSET FETCH`  . N1 ( `OFFSET` )  N2 ( `FETCH` )  .

```
SELECT *
```

```
FROM sys.objects
ORDER BY object_id
OFFSET 40 ROWS FETCH NEXT 10 ROWS ONLY
```

`ORDER BY` .

SQL Server `TOP` .

```
 SELECT TOP 10 *
 FROM
 (
    SELECT
    TOP 50 object_id,
        name,
        type,
        create_date
    FROM sys.objects
    ORDER BY name ASC
) AS data
ORDER BY name DESC
```

`name` 50 . 50 10 ( 10 ).

### SQL Server

# SQL Server 2012/2014

```
DECLARE @RowsPerPage INT = 10, @PageNumber INT = 4
SELECT OrderId, ProductId
FROM OrderDetail
ORDER BY OrderId
OFFSET (@PageNumber - 1) * @RowsPerPage ROWS
FETCH NEXT @RowsPerPage ROWS ONLY
```

# SQL Server 2005 / 2008 / R2

```
DECLARE @RowsPerPage INT = 10, @PageNumber INT = 4
SELECT OrderId, ProductId
FROM (
    SELECT OrderId, ProductId, ROW_NUMBER() OVER (ORDER BY OrderId) AS RowNum
    FROM OrderDetail) AS OD
WHERE OD.RowNum BETWEEN ((@PageNumber - 1 ) * @RowsPerPage) + 1
AND @RowsPerPage * @PageNumber
```

# SQL Server 2000

```
DECLARE @RowsPerPage INT = 10, @PageNumber INT = 4
SELECT OrderId, ProductId
FROM (SELECT TOP (@RowsPerPage) OrderId, ProductId
        FROM (SELECT TOP ((@PageNumber)*@RowsPerPage) OrderId, ProductId
                FROM OrderDetail
                ORDER BY OrderId) AS OD
    ORDER BY OrderId DESC) AS OD2
ORDER BY OrderId ASC
```

## ORDER BY OFFSET  FETCH NEXT  SQL Server 2012/2014

10  .

```
SELECT * FROM TableName ORDER BY id OFFSET 10 ROWS FETCH NEXT 10 ROWS ONLY;
```

**:**

- ORDER BY OFFSET FETCH .
- FETCH OFFSET .  ORDER BY ... FETCH .
- TOP  OFFSET FETCH  .

: https://riptutorial.com/ko/sql-server/topic/6874/-

# 94:

## Examples

6 (126 ) :

```
SELECT TradeDate, AVG(Px) OVER (ORDER BY TradeDate ROWS BETWEEN 63 PRECEDING AND 63 FOLLOWING)
AS PxMovingAverage
FROM HistoricalPrices
```

TradeDate 63 . TradeDate 63 . .

.

. . . row_number () (by order by ...) ( my_ranking = 1).

```
select *
from (
    select
        *,
        row_number() over (order by crdate desc) as my_ranking
    from sys.sysobjects
) g
where my_ranking=1
```

.

### 30

30

```
SELECT
    value_column1,
    (   SELECT
            AVG(value_column1) AS moving_average
        FROM Table1 T2
        WHERE ( SELECT
                    COUNT(*)
                FROM Table1 T3
                WHERE date_column1 BETWEEN T2.date_column1 AND T1.date_column1
                ) BETWEEN 1 AND 30
    ) as MovingAvg
FROM Table1 T1
```

: https://riptutorial.com/ko/sql-server/topic/3209/-

# 95:

- DECLARE cursor_name CURSOR [ | ]
  - [FORWARD_ONLY | ]
    [ | | | ]
    [READ_ONLY | SCROLL_LOCKS | [OPTIMISTIC]
    [TYPE_WARNING]
  - FOR select_statement
  - [FOR UPDATE [OF column_name [, ... n]]]

. .

## Examples

. .

```
DECLARE @orderId AS INT

-- here we are creating our cursor, as a local cursor and only allowing
-- forward operations
DECLARE rowCursor CURSOR LOCAL FAST_FORWARD FOR
    -- this is the query that we want to loop through record by record
    SELECT [OrderId]
    FROM [dbo].[Orders]

-- first we need to open the cursor
OPEN rowCursor

-- now we will initialize the cursor by pulling the first row of data, in this example the
[OrderId] column,
-- and storing the value into a variable called @orderId
FETCH NEXT FROM rowCursor INTO @orderId

-- start our loop and keep going until we have no more records to loop through
WHILE @@FETCH_STATUS = 0
BEGIN

    PRINT @orderId

    -- this is important, as it tells SQL Server to get the next record and store the
[OrderId] column value into the @orderId variable
    FETCH NEXT FROM rowCursor INTO @orderId

END

-- this will release any memory used by the cursor
CLOSE rowCursor
DEALLOCATE rowCursor
```

:

```
/* Prepare test data */
DECLARE @test_table TABLE
```

```
(
    Id INT,
    Val VARCHAR(100)
);
INSERT INTO @test_table(Id, Val)
VALUES
    (1, 'Foo'),
    (2, 'Bar'),
    (3, 'Baz');
/* Test data prepared */

/* Iterator variable @myId, for example sake */
DECLARE @myId INT;

/* Cursor to iterate rows and assign values to variables */
DECLARE myCursor CURSOR FOR
    SELECT Id
    FROM @test_table;

/* Start iterating rows */
OPEN myCursor;
FETCH NEXT FROM myCursor INTO @myId;

/* @@FETCH_STATUS global variable will be 1 / true until there are no more rows to fetch */
WHILE @@FETCH_STATUS = 0
BEGIN

    /* Write operations to perform in a loop here. Simple SELECT used for example */
    SELECT Id, Val
    FROM @test_table
    WHERE Id = @myId;

    /* Set variable(s) to the next value returned from iterator; this is needed otherwise the
cursor will loop infinitely. */
    FETCH NEXT FROM myCursor INTO @myId;
END
/* After all is done, clean up */
CLOSE myCursor;
DEALLOCATE myCursor;
```

SSMS . . .

| |
|---|
| 1 |
| (1 ) |
| 2 |
| (1 ) |
| |
| (1 ) |

: https://riptutorial.com/ko/sql-server/topic/870/

# 96:

## Examples

SQL Server Seek  Scan  .

SQL Server  . `where name = 'Foo'`  .

SQL Server  .

.

: https://riptutorial.com/ko/sql-server/topic/7713/-

# 97:

## Examples

### JOIN Hints

SQL Server (QO) .

- 
- LOOP
- 

QO . JOIN . Inner LOOP QO .

```
select top 100 *
from Sales.Orders o
    inner loop join Sales.OrderLines ol
    on o.OrderID = ol.OrderID
```

MERGE .

```
select top 100 *
from Sales.Orders o
    inner merge join Sales.OrderLines ol
    on o.OrderID = ol.OrderID
```

HASH :

```
select top 100 *
from Sales.Orders o
    inner hash join Sales.OrderLines ol
    on o.OrderID = ol.OrderID
```

GROUP BY SQL Server (QO) .

- HASH
- 

QO . OPTION (ORDER GROUP) QO Stream aggregate Stream aggregate .

```
select OrderID, AVG(Quantity)
from Sales.OrderLines
group by OrderID
OPTION (ORDER GROUP)
```

OPTION ( ) QO .

```
select OrderID, AVG(Quantity)
from Sales.OrderLines
```

```
group by OrderID
OPTION (HASH GROUP)
```

number_rows . . number_rows .

```
select OrderID, AVG(Quantity)
from Sales.OrderLines
group by OrderID
OPTION (FAST 20)
```

## UNION

UNION (QO) .

- ()
- ()
- ()

OPTION () .

```
select OrderID, OrderDate, ExpectedDeliveryDate, Comments
from Sales.Orders
where OrderDate > DATEADD(day, -1, getdate())
UNION
select PurchaseOrderID as OrderID, OrderDate, ExpectedDeliveryDate, Comments
from Purchasing.PurchaseOrders
where OrderDate > DATEADD(day, -1, getdate())
OPTION(HASH UNION)
-- or OPTION(CONCAT UNION)
-- or OPTION(MERGE UNION)
```

## MAXDOP

.

```
SELECT OrderID,
    AVG(Quantity)
FROM Sales.OrderLines
GROUP BY OrderID
OPTION (MAXDOP 2);
```

sp_configure Resource Governor MAXDOP . MAXDOP 0 .

SQL Server . . SQL Server .

```
SELECT *
FROM mytable WITH (INDEX (ix_date))
WHERE field1 > 0
    AND CreationDate > '20170101'
```

: https://riptutorial.com/ko/sql-server/topic/6881/-

# 98:

Drop SQL . .

MSDN

- DROP TABLE (Transact-SQL)
- DROP PROCEDURE (Transact-SQL)
- DROP DATABASE (Transact-SQL)

## Examples

**DROP TABLE** , , , .

(, , ) .

FOREIGN KEY . .

.

```
DROP TABLE [ IF EXISTS ] [ database_name . [ schema_name ] . | schema_name . ]
table_name [ ,...n ] [ ; ]
```

- `IF EXISTS` -
- `database_name` -
- `schema_name` -
- `table_name` -

**TABLE_1** .

```
DROP TABLE Table_1;
```

**HR** dbo **TABLE_1**

```
DROP TABLE HR.Table_1;
```

**HR** **TABLE_1**

```
DROP TABLE HR.external.TABLE_1;
```

**DROP DATABASE** SQL Server (, , ) .

.

'sp_detach_db' ( ) .

SQL Server  SQL Server  .

.

```
DROP DATABASE [ IF EXISTS ] { database_name | database_snapshot_name } [ ,...n ] [;]
```

- `IF EXISTS` -
- `database_name` -  .
- `database_snapshot_name` -  .
-

.

```
DROP DATABASE Database1;
```

```
DROP DATABASE Database1, Database2;
```

```
DROP DATABASE Database1_snapshot17;
```

```
DROP DATABASE IF EXISTS Database1;
```

SQL  2  .

1. `##GlobalTempTable`  .
2. `#LocalTempTable` -  ( - ID `SELECT @@SPID`  ).

.

```
DROP TABLE [ database_name . [ schema_name ] . | schema_name . ] table_name
```

SQL Server  2016 :

```
IF(OBJECT_ID('tempdb..#TempTable') is not null)
    DROP TABLE #TempTable;
```

SQL Server 2016 :

```
DROP TABLE IF EXISTS #TempTable
```

: https://riptutorial.com/ko/sql-server/topic/9532/-

# 99:

( TVP) . .

.

## Examples

, .

```
CREATE TYPE names as TABLE
(
    FirstName varchar(10),
    LastName varchar(10)
)
GO
```

.

```
CREATE PROCEDURE prInsertNames
(
    @Names dbo.Names READONLY -- Note: You must specify the READONLY
)
AS

INSERT INTO dbo.TblNames (FirstName, LastName)
SELECT FirstName, LastName
FROM @Names
GO
```

:

```
DECLARE @names dbo.Names
INSERT INTO @Names VALUES
('Zohar', 'Peled'),
('First', 'Last')

EXEC dbo.prInsertNames @Names
```

: https://riptutorial.com/ko/sql-server/topic/5285/---

# 100:

## Examples

( SSMS) .

```
SELECT Column1, Column2, Column3 FROM MySourceTable;
```

.

```
INSERT INTO MyTargetTable (Column1, Column2, Column3)
SELECT Column1, Column2, Column3 FROM MySourceTable;
```

.

```
SELECT Column1, Column2, Column3 FROM MySourceTable;
```

MyNewTable .

```
SELECT Column1, Column2, Column3
INTO MyNewTable
FROM MySourceTable;
```

### ( )

. SQL .

? .

.

```
SELECT Key1, Key2, Column3, Column4 FROM MyTable;
```

:

```
INSERT INTO TargetTable (Key1, Key2, Column3, Column4)
SELECT Key1, Key2, Column3, Column4 FROM MyTable;
```

Key1 , Key2

```
DELETE MyTable
WHERE EXISTS (
    SELECT * FROM TargetTable
    WHERE TargetTable.Key1 = SourceTable.Key1
    AND TargetTable.Key2 = SourceTable.Key2
);
```

Key1 , Key2 .

, . . .

```
BEGIN TRAN;

INSERT INTO TargetTable (Key1, Key2, Column3, Column4)
SELECT Key1, Key2, Column3, Column4 FROM MyTable;


DELETE MyTable
WHERE EXISTS (
    SELECT * FROM TargetTable
    WHERE TargetTable.Key1 = SourceTable.Key1
    AND TargetTable.Key2 = SourceTable.Key2
);

COMMIT TRAN;
```

: https://riptutorial.com/ko/sql-server/topic/1467/-----

# 101:

- { | READ COMMITTED | REPEATABLE READ | SNAPSHOT | } [; ]

MSDN : SET TRANSACTION ISOLATION LEVEL

## Examples

SQL Server 2008 R2

```
SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED
```

. (, ). " " .

SQL Server 2008 R2

```
SET TRANSACTION ISOLATION LEVEL READ COMMITTED
```

. . READ COMMITTED READ_COMMITTED_SNAPSHOT .

- OFF ( ) .
- ON READ COMMITTED READCOMMITTEDLOCK .

: READ COMMITTED SQL Server .

### " " ?

( ) .

2 . .

**1** - .

```
CREATE TABLE dbo.demo (
    col1 INT,
    col2 VARCHAR(255)
);
GO
--This row will get committed normally:
BEGIN TRANSACTION;
    INSERT INTO dbo.demo(col1, col2)
    VALUES (99, 'Normal transaction');
COMMIT TRANSACTION;
--This row will be "stuck" in an open transaction, causing a dirty read
BEGIN TRANSACTION;
    INSERT INTO dbo.demo(col1, col2)
    VALUES (42, 'Dirty read');
--Do not COMMIT TRANSACTION or ROLLBACK TRANSACTION here
```

**2** - .

```
SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED;
SELECT * FROM dbo.demo;
```

:

```
col1        col2
----------- ---------------------------------------
99          Normal transaction
42          Dirty read
```

: :

```
COMMIT TRANSACTION;
DROP TABLE dbo.demo;
GO
```

### SQL Server 2008 R2

```
SET TRANSACTION ISOLATION LEVEL REPEATABLE READ
```

READ COMMITTED . .

: . READ COMMITTED ..

### SQL Server 2008 R2

```
SET TRANSACTION ISOLATION LEVEL SNAPSHOT
```

., .

SNAPSHOT ( ) .

SNAPSHOT . .

---

: SNAPSHOT ALLOW_SNAPSHOT_ISOLATION ON .

### SQL Server 2008 R2

```
SET TRANSACTION ISOLATION LEVEL SERIALIZEABLE
```

. . INSERT .

SELECT HOLDLOCK .

---

: .

: https://riptutorial.com/ko/sql-server/topic/5114/--

# 102:

| | |
|---|---|
| transaction_name | [ *with mark* ] . / (!) |
| [ 'description'] | [ *transaction_name* ] |

## Examples

```
BEGIN TRY -- start error handling
    BEGIN TRANSACTION; -- from here on transactions (modifictions) are not final
        -- start your statement(s)
        select 42/0 as ANSWER  -- simple SQL Query with an error
        -- end your statement(s)
    COMMIT TRANSACTION; -- finalize all transactions (modifications)
END TRY   -- end error handling -- jump to end
BEGIN CATCH -- execute this IF an error occured
        ROLLBACK TRANSACTION; -- undo any transactions (modifications)
-- put together some information as a query
    SELECT
        ERROR_NUMBER() AS ErrorNumber
        ,ERROR_SEVERITY() AS ErrorSeverity
        ,ERROR_STATE() AS ErrorState
        ,ERROR_PROCEDURE() AS ErrorProcedure
        ,ERROR_LINE() AS ErrorLine
        ,ERROR_MESSAGE() AS ErrorMessage;

END CATCH;  -- final line of error handling
GO -- execute previous code
```

: https://riptutorial.com/ko/sql-server/topic/5859/-

# 103:

, , , T-SQL . .

. SQL Server Management Studio .

## Examples

SQL Server ( [ ] ) . . .

```
SELECT [Description]
FROM   dbo.TableName
WHERE  [Name] = 'foo'
```

SQL Server ' . *O'Shea* .

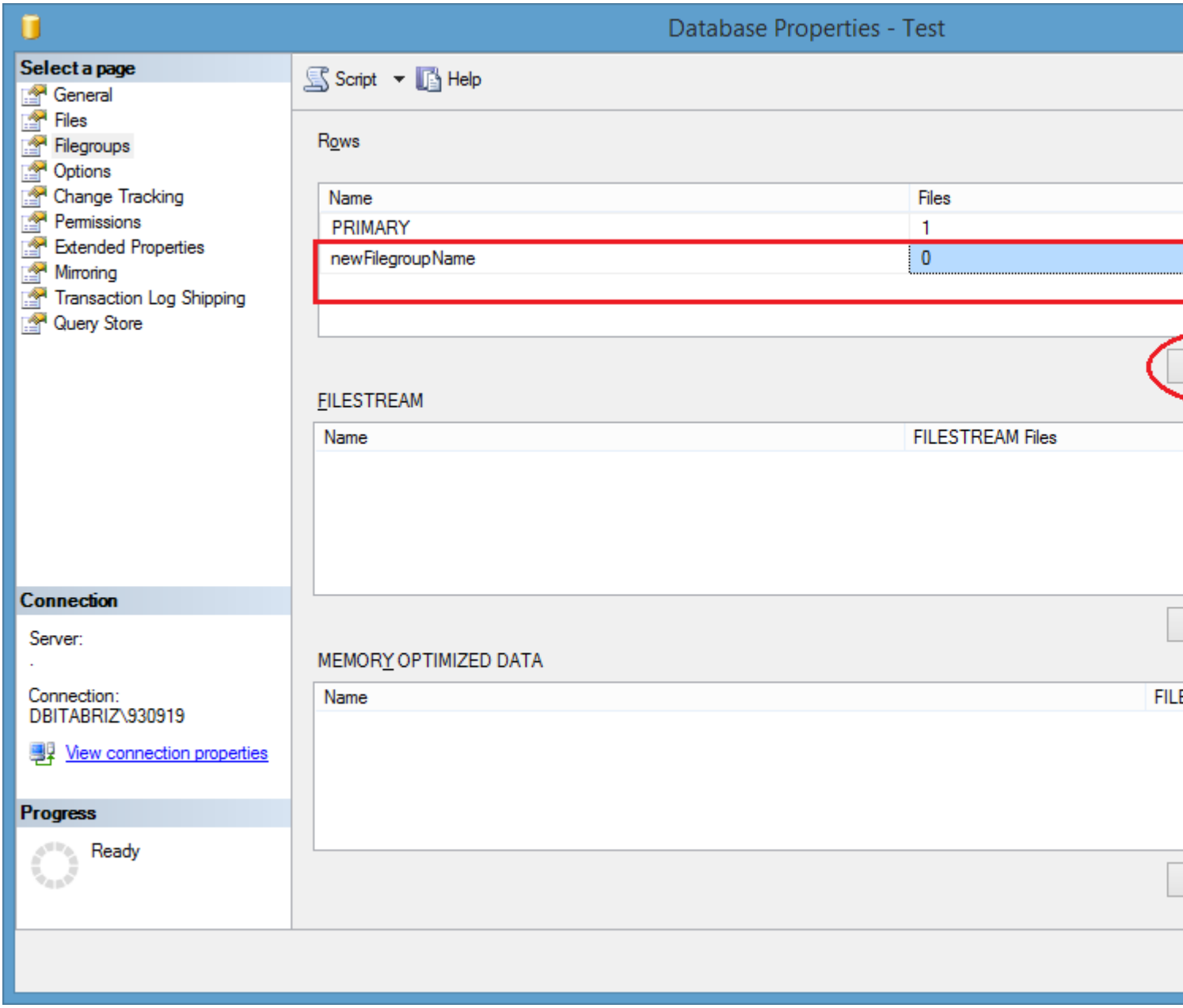
```
SELECT [Description]
FROM   dbo.TableName
WHERE  [Name] = 'O''Shea'
```

: https://riptutorial.com/ko/sql-server/topic/7156/----

# 104:

## Examples

. :




SQL :

```
USE master;
GO
-- Create the database with the default data
-- filegroup and a log file. Specify the
-- growth increment and the max size for the
-- primary data file.

CREATE DATABASE TestDB ON PRIMARY
```

```
(
    NAME = 'TestDB_Primary',
    FILENAME = 'C:\Program Files\Microsoft SQL
Server\MSSQL12.MSSQLSERVER\MSSQL\DATA\TestDB_Prm.mdf',
    SIZE = 1 GB,
    MAXSIZE = 10 GB,
    FILEGROWTH = 1 GB
), FILEGROUP TestDB_FG1
(
    NAME = 'TestDB_FG1_1',
    FILENAME = 'C:\Program Files\Microsoft SQL
Server\MSSQL12.MSSQLSERVER\MSSQL\DATA\TestDB_FG1_1.ndf',
    SIZE = 10 MB,
    MAXSIZE = 10 GB,
    FILEGROWTH = 1 GB
),
(
    NAME = 'TestDB_FG1_2',
    FILENAME = 'C:\Program Files\Microsoft SQL
Server\MSSQL12.MSSQLSERVER\MSSQL\DATA\TestDB_FG1_2.ndf',
    SIZE = 10 MB,
    MAXSIZE = 10 GB,
    FILEGROWTH = 1 GB
) LOG ON
(
    NAME = 'TestDB_log',
    FILENAME = 'C:\Program Files\Microsoft SQL
Server\MSSQL12.MSSQLSERVER\MSSQL\DATA\TestDB.ldf',
    SIZE = 10 MB,
    MAXSIZE = 10 GB,
    FILEGROWTH = 1 GB
);

go
ALTER DATABASE TestDB MODIFY FILEGROUP TestDB_FG1 DEFAULT;
go

-- Create a table in the user-defined filegroup.
USE TestDB;
Go

CREATE TABLE MyTable
(
    col1 INT PRIMARY KEY,
    col2 CHAR(8)
)
ON TestDB_FG1;
GO
```

: https://riptutorial.com/ko/sql-server/topic/5461/-

# 105:

FILESTREAM varbinary (max) BLOB (Binary Large Object)  SQL Server  NTFS  . Transact-SQL  FILESTREAM  , ,  . Win32  .

## Examples

: MSDN https://technet.microsoft.com/en-us/library/bb933993(v=sql.105).aspx

: https://riptutorial.com/ko/sql-server/topic/9509/-

# 106:

## Examples

```
SELECT        ps.name AS PartitionScheme
            , fg.name AS [FileGroup]
            , prv.*
            , LAG(prv.Value) OVER (PARTITION BY ps.name ORDER BY ps.name, boundary_id) AS
PreviousBoundaryValue

FROM        sys.partition_schemes ps
INNER JOIN    sys.destination_data_spaces dds
            ON dds.partition_scheme_id = ps.data_space_id
INNER JOIN    sys.filegroups fg
            ON dds.data_space_id = fg.data_space_id
INNER JOIN    sys.partition_functions f
            ON f.function_id = ps.function_id
INNER JOIN    sys.partition_range_values prv
            ON f.function_id = prv.function_id
            AND dds.destination_id = prv.boundary_id
```

[TechNet Microsoft ] [1] ,

.

. .

```
ALTER TABLE [SourceTable] SWITCH TO [TargetTable]
```

NULL , . .

[1] : https://technet.microsoft.com/en-us/library/ms191160(v=sql.105).aspx `IDENTITY` .

, , , ,

```
SELECT DISTINCT
    object_name(i.object_id) AS [Object Name],
    c.name AS [Partition Column],
    s.name AS [Partition Scheme],
    pf.name AS [Partition Function],
    prv.tot AS [Partition Count],
    prv.miVal AS [Min Boundry Value],
    prv.maVal AS [Max Boundry Value]
FROM sys.objects o
INNER JOIN sys.indexes i ON i.object_id = o.object_id
INNER JOIN sys.columns c ON c.object_id = o.object_id
INNER JOIN sys.index_columns ic ON ic.object_id = o.object_id
    AND ic.column_id = c.column_id
    AND ic.partition_ordinal = 1
INNER JOIN sys.partition_schemes s ON i.data_space_id = s.data_space_id
INNER JOIN sys.partition_functions pf ON pf.function_id = s.function_id
OUTER APPLY(SELECT
                COUNT(*) tot, MIN(value) miVal, MAX(value) maVal
            FROM sys.partition_range_values prv
```

```
            WHERE prv.function_id = pf.function_id) prv
--WHERE object_name(i.object_id) = 'table_name'
ORDER BY OBJECT_NAME(i.object_id)
```

where  where table_name actual table name table_name    .

: https://riptutorial.com/ko/sql-server/topic/3212/

# 107:

/ . .

MSDN ISOLATION LEVEL .

## Examples

```
CREATE TABLE [dbo].[Table_1](
    [Id] [int] IDENTITY(1,1) NOT NULL,
    [title] [varchar](50) NULL,
 CONSTRAINT [PK_Table_1] PRIMARY KEY CLUSTERED
(
    [Id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS =
ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
```

() ( **--rollback** ) DB .

```
begin tran

INSERT INTO Table_1 values('Title 1')

SELECT * FROM [Test].[dbo].[Table_1]

--rollback
```

```
begin tran

set transaction isolation level READ UNCOMMITTED

SELECT * FROM [Test].[dbo].[Table_1]
```

( ) . ( ).

```
-- Rollback the first transaction
rollback
```

( ). .

.

```
set transaction isolation level READ UNCOMMITTED
```

: https://riptutorial.com/ko/sql-server/topic/8235/-

# 108:

## Examples

### Row_Number ()

```
SELECT Row_Number() OVER(ORDER BY UserName) As RowID, UserFirstName, UserLastName
FROM Users
```

RowID .

```
SELECT *
FROM
    ( SELECT Row_Number() OVER(ORDER BY UserName) As RowID, UserFirstName, UserLastName
      FROM Users
    ) As RowResults
WHERE RowID Between 5 AND 10
```

: https://riptutorial.com/ko/sql-server/topic/5803/---

# 109: /

- SELECT <non-pivoted column> ,
  [ ] AS <column name> ,
  [ ] AS <column name> ,
  ...
  [ ] AS <column name>
  ( <SELECT query that produces the data> )
  AS <alias for the source query>
  (
  <aggregation function> ( <column being aggregated> )
  [ <column that contains the values that will become column headers> ]
  IN ([ ], [ ],
  ... [ ])
  ) AS <alias for the pivot table> <optional ORDER BY clause> ;

PIVOT UNPIVOT ( ) . .

## Examples

-

Example Database Item Sales Table .

ID " .

```
SELECT [100], [145]
  FROM (SELECT ItemId , Quantity
          FROM #ItemSalesTable
       ) AS pivotIntermediate
 PIVOT (   SUM(Quantity)
           FOR ItemId IN ([100], [145])
       ) AS pivotTable
```

" [] []

.

| 100 | 145 |
|---|---|
| 45 | 18 |

### PIVOT UNPIVOT (T-SQL)

.

.

```
CREATE TABLE tbl_stock(item NVARCHAR(10), weekday NVARCHAR(10), price INT);

INSERT INTO tbl_stock VALUES
('Item1', 'Mon', 110), ('Item2', 'Mon', 230), ('Item3', 'Mon', 150),
('Item1', 'Tue', 115), ('Item2', 'Tue', 231), ('Item3', 'Tue', 162),
('Item1', 'Wed', 110), ('Item2', 'Wed', 240), ('Item3', 'Wed', 162),
('Item1', 'Thu', 109), ('Item2', 'Thu', 228), ('Item3', 'Thu', 145),
('Item1', 'Fri', 120), ('Item2', 'Fri', 210), ('Item3', 'Fri', 125),
('Item1', 'Mon', 122), ('Item2', 'Mon', 225), ('Item3', 'Mon', 140),
('Item1', 'Tue', 110), ('Item2', 'Tue', 235), ('Item3', 'Tue', 154),
('Item1', 'Wed', 125), ('Item2', 'Wed', 220), ('Item3', 'Wed', 142);
```

.

```
+========+=========+=======+
|   item | weekday | price |
+========+=========+=======+
|  Item1 |    Mon  |   110 |
+--------+---------+-------+
|  Item2 |    Mon  |   230 |
+--------+---------+-------+
|  Item3 |    Mon  |   150 |
+--------+---------+-------+
|  Item1 |    Tue  |   115 |
+--------+---------+-------+
|  Item2 |    Tue  |   231 |
+--------+---------+-------+
|  Item3 |    Tue  |   162 |
+--------+---------+-------+
|          . . .           |
+--------+---------+-------+
|  Item2 |    Wed  |   220 |
+--------+---------+-------+
|  Item3 |    Wed  |   142 |
+--------+---------+-------+
```

PIVOT  weekday  .

```
SELECT * FROM tbl_stock
PIVOT (
    AVG(price) FOR weekday IN ([Mon], [Tue], [Wed], [Thu], [Fri])
) pvt;
```

:

```
+--------+------+------+------+------+------+
|  item  |  Mon |  Tue |  Wed |  Thu |  Fri |
+--------+------+------+------+------+------+
|  Item1 |  116 |  112 |  117 |  109 |  120 |
|  Item2 |  227 |  233 |  230 |  228 |  210 |
|  Item3 |  145 |  158 |  152 |  145 |  125 |
+--------+------+------+------+------+------+
```

,PIVOT  UNPIVOT  .

```
SELECT * FROM tbl_stock
```

```
PIVOT (
    AVG(price) FOR weekday IN ([Mon], [Tue], [Wed], [Thu], [Fri])
) pvt
UNPIVOT (
    price FOR weekday IN ([Mon], [Tue], [Wed], [Thu], [Fri])
) unpvt;
```

:

```
+=======+========+=========+
|  item |  price | weekday |
+=======+========+=========+
| Item1 |    116 |     Mon |
+-------+--------+---------+
| Item1 |    112 |     Tue |
+-------+--------+---------+
| Item1 |    117 |     Wed |
+-------+--------+---------+
| Item1 |    109 |     Thu |
+-------+--------+---------+
| Item1 |    120 |     Fri |
+-------+--------+---------+
| Item2 |    227 |     Mon |
+-------+--------+---------+
| Item2 |    233 |     Tue |
+-------+--------+---------+
| Item2 |    230 |     Wed |
+-------+--------+---------+
| Item2 |    228 |     Thu |
+-------+--------+---------+
| Item2 |    210 |     Fri |
+-------+--------+---------+
| Item3 |    145 |     Mon |
+-------+--------+---------+
| Item3 |    158 |     Tue |
+-------+--------+---------+
| Item3 |    152 |     Wed |
+-------+--------+---------+
| Item3 |    145 |     Thu |
+-------+--------+---------+
| Item3 |    125 |     Fri |
+-------+--------+---------+
```

## PIVOT

`PIVOT` `IN` . `PIVOT` IN .

`Bookstore` `Books` . .

```
Table: Books
----------------------------
BookId (Primary Key Column)
Name
Language
NumberOfPages
EditionNumber
YearOfPrint
```

```
YearBoughtIntoStore
ISBN
AuthorName
Price
NumberOfUnitsSold
```

.

```
CREATE TABLE [dbo].[BookList](
      [BookId] [int] NOT NULL,
      [Name] [nvarchar](100) NULL,
      [Language] [nvarchar](100) NULL,
      [NumberOfPages] [int] NULL,
      [EditionNumber] [nvarchar](10) NULL,
      [YearOfPrint] [int] NULL,
      [YearBoughtIntoStore] [int] NULL,
[NumberOfBooks] [int] NULL,
[ISBN] [nvarchar](30) NULL,
      [AuthorName] [nvarchar](200) NULL,
      [Price] [money] NULL,
      [NumberOfUnitsSold] [int] NULL,
 CONSTRAINT [PK_BookList] PRIMARY KEY CLUSTERED
(
      [BookId] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS =
ON, ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]

GO
```

, , , ,  PIVOT

```
SELECT * FROM
  (
   SELECT YearBoughtIntoStore AS [Year Bought],[Language], NumberOfBooks
   FROM BookList
  ) sourceData
 PIVOT
  (
  SUM(NumberOfBooks)
  FOR [Language] IN (English, Russian, German, Hindi, Latin)
  ) pivotrReport
```

SQL .

```
DECLARE @query VARCHAR(4000)
DECLARE @languages VARCHAR(2000)
SELECT @languages =
        STUFF((SELECT DISTINCT '],['+LTRIM([Language])FROM [dbo].[BookList]
        ORDER BY '],['+LTRIM([Language]) FOR XML PATH('') ),1,2,'') + ']'
SET @query=
'SELECT * FROM
  (SELECT YearBoughtIntoStore AS [Year Bought],[Language],NumberOfBooks
   FROM BookList) sourceData
PIVOT(SUM(NumberOfBooks)FOR [Language] IN ('+ @languages +')) pivotrReport' EXECUTE(@query)
```

/  : https://riptutorial.com/ko/sql-server/topic/591/----

# 110:

## Examples

SQL . .

1. ,
2. ORDER BY .
3. BLOB, , .

IN, SELECT , where , select , select select, insert, update delete .

ITCompanyInNepal :

| ID | CompanyName | CompanyAddress | Headquarter | NumberOfEmployee |
|---|---|---|---|---|
| 1 | CompanyOne | Kathmandu | USA | 350 |
| 2 | CompanyTwo | Kathmandu | USA | 310 |
| 3 | CompanyThree | Kathmandu | Nepal | 300 |
| 4 | CompanyFour | Kathmandu | Nepal | 180 |
| 5 | CompanyFive | Birgunj | Denmark | 150 |
| 6 | CompanySix | Janakpur | USA | 100 |
| 7 | CompanySeven | Janakpur | Australia | 100 |
| 8 | CompanyEight | Birganj | Australia | 150 |
| 9 | CompanyNine | Biratnagar | Canada | 200 |
| 10 | CompanyTen | Pokhara | India | 85 |

: **Select SubQuery**

**In where .**

```
SELECT *
FROM ITCompanyInNepal
WHERE Headquarter IN (SELECT Headquarter
                      FROM ITCompanyInNepal
                      WHERE Headquarter = 'USA');
```

**where**

```
SELECT *
FROM ITCompanyInNepal
WHERE NumberOfEmployee < (SELECT AVG(NumberOfEmployee)
                          FROM ITCompanyInNepal
                      )
```

```
SELECT   CompanyName,
         CompanyAddress,
         Headquarter,
         (Select SUM(NumberOfEmployee)
         FROM ITCompanyInNepal
         Where Headquarter = 'USA') AS TotalEmployeeHiredByUSAInKathmandu
```

```
FROM     ITCompanyInNepal
WHERE    CompanyAddress = 'Kathmandu' AND Headquarter = 'USA'
```

### insert

IndianCompany  ITCompanyInNepal . IndianCompany  .

| ID | CompanyName | CompanyAddress | Headquarter | NumberOfEmployee |
|---|---|---|---|---|
| 1 | CompanyA | Banglore | USA | 450 |
| 2 | CompanyB | Banglore | USA | 500 |
| 3 | CompanyC | Hyderabad | Denmark | 480 |
| 4 | CompanyD | Hyderabad | Australia | 780 |
| 5 | CompanyE | Delhi | Canada | 790 |

```
INSERT INTO ITCompanyInNepal
SELECT *
FROM IndianCompany
```

50  .

```
UPDATE ITCompanyInNepal
SET NumberOfEmployee = NumberOfEmployee - 50
WHERE Headquarter IN (SELECT Headquarter
                      FROM ITCompanyInNepal
                      WHERE Headquarter = 'USA')
```

### Delete

.

```
DELETE FROM ITCompanyInNepal
WHERE Headquarter IN (SELECT Headquarter
                     FROM ITCompanyInNepal
                     WHERE Headquarter = 'Denmark')
```

: https://riptutorial.com/ko/sql-server/topic/5629/-

# 111:

. SQL Server `TOP` `OFFSET FETCH` .

| | |
|---|---|
| `TOP` | . . |
| `PERCENT` | . `TOP` . |

`ORDER BY` .

## Examples

### TOP

`SELECT` 100 .

```
SELECT TOP 100 *
FROM table_name;
```

.

```
DECLARE @CountDesiredRows int = 100;
SELECT TOP (@CountDesiredRows) *
FROM table_name;
```

### PERCENT

`SELECT` 15 % .

```
SELECT TOP 15 PERCENT *
FROM table_name
```

### FETCH

SQL Server 2012

`FETCH` `TOP` .

```
SELECT *
FROM table_name
ORDER BY 1
OFFSET 0 ROWS
FETCH NEXT 50 ROWS ONLY
```

: https://riptutorial.com/ko/sql-server/topic/1555/--

# 112:

## Examples

### RLS

Sql Server 2016+ Azure Sql select . .

, .

```
DROP FUNCTION IF EXISTS dbo.pUserCanAccessCompany
GO
CREATE FUNCTION

dbo.pUserCanAccessCompany(@CompanyID int)

    RETURNS TABLE
    WITH SCHEMABINDING
AS RETURN (
    SELECT 1 as canAccess WHERE

    CAST(SESSION_CONTEXT(N'CompanyID') as int) = @CompanyID

)
```

SESSION_CONTEXT . database_id .

. WHERE . policy .

.

```
CREATE SECURITY POLICY dbo.CompanyAccessPolicy
    ADD FILTER PREDICATE dbo.pUserCanAccessCompany(CompanyID) ON dbo.Company
    WITH (State=ON)
```

. Company CompanyID SELECT .

### RLS

policy . .

policy .

```
ALTER SECURITY POLICY dbo.CompanyAccessPolicy
    ADD FILTER PREDICATE dbo.pUserCanAccessCompany(CompanyID) ON dbo.Company
```

.

```
ALTER SECURITY POLICY dbo.CompanyAccessPolicy
    DROP FILTER PREDICATE ON dbo.Company
```

.

```
ALTER SECURITY POLICY dbo.CompanyAccessPolicy WITH ( STATE = OFF );
```

.

```
ALTER SECURITY POLICY dbo.CompanyAccessPolicy WITH ( STATE = ON );
```

## RLS

. .

CREATE FUNCTION

dbo.pUserCanAccessProduct (@ CompanyID int)

```
RETURNS TABLE
WITH SCHEMABINDING
```

AS RETURN (1 canAccess .

CAST (SESSION_CONTEXT (N'CompanyID ') int ) @ @CompanyID

) SESSION_CONTEXT . database_id .

. WHERE . policy .

CompanyID .

dbo.ProductAccessPolicy ADB (ADD BLOCK PREDICATE) dbo.pUserCanAccessProduct (CompanyID) dbo.Product

. :

dbo.ProductAccessPolicy ADOBE_PROTECATE dbo.pUserCanAccessProduct (CompanyID) ON dbo.Product INSERT

.

[{AFTER {INSERT | }}
| {BEFORE {UPDATE | }}]

: https://riptutorial.com/ko/sql-server/topic/7045/--

# 113: /

## Examples

.

```
-- Create a table as an example
CREATE TABLE SortOrder
(
    ID INT IDENTITY PRIMARY KEY,
    [Text] VARCHAR(256)
)
GO

-- Insert rows into the table
INSERT INTO SortOrder ([Text])
SELECT ('Lorem ipsum dolor sit amet, consectetur adipiscing elit')
UNION ALL SELECT ('Pellentesque eu dapibus libero')
UNION ALL SELECT ('Vestibulum et consequat est, ut hendrerit ligula')
UNION ALL SELECT ('Suspendisse sodales est congue lorem euismod, vel facilisis libero
pulvinar')
UNION ALL SELECT ('Suspendisse lacus est, aliquam at varius a, fermentum nec mi')
UNION ALL SELECT ('Praesent tincidunt tortor est, nec consequat dolor malesuada quis')
UNION ALL SELECT ('Quisque at tempus arcu')
GO
```

(ORDER BY) SQL Server ( ) . , . , .

ORDER BY == . .

```
-- It may seem the rows are sorted by identifiers,
-- but there is really no way of knowing if it will always work.
-- And if you leave it like this in production, Murphy gives you a 100% that it wont.
SELECT * FROM SortOrder
GO
```

.

- ( ), ASC
- ( ), DESC

```
-- Ascending - upwards
SELECT * FROM SortOrder ORDER BY ID ASC
GO

-- Ascending is default
SELECT * FROM SortOrder ORDER BY ID
GO

-- Descending - downwards
SELECT * FROM SortOrder ORDER BY ID DESC
GO
```

((n) char (n) varchar) . .

. . .

. .

```
SELECT * FROM SortOrder ORDER BY CHECKSUM(NEWID())
GO
```

.

```
CREATE PROCEDURE GetSortOrder
AS
    SELECT *
    FROM SortOrder
    ORDER BY ID DESC
GO

EXEC GetSortOrder
GO
```

SQL Server ( ) .

```
/* This may or may not work, and it depends on the way
   your SQL Server and updates are installed */
CREATE VIEW VwSortOrder1
AS
    SELECT TOP 100 PERCENT *
    FROM SortOrder
    ORDER BY ID DESC
GO

SELECT * FROM VwSortOrder1
GO

-- This will work, but hey... should you really use it?
CREATE VIEW VwSortOrder2
AS
    SELECT TOP 99999999 *
    FROM SortOrder
    ORDER BY ID DESC
GO

SELECT * FROM VwSortOrder2
GO
```

ORDER BY , .

```
SELECT *
FROM SortOrder
ORDER BY [Text]

-- New resultset column aliased as 'Msg', feel free to use it for ordering
SELECT ID, [Text] + ' (' + CAST(ID AS nvarchar(10)) + ')' AS Msg
FROM SortOrder
ORDER BY Msg
```

```
-- Can be handy if you know your tables, but really NOT GOOD for production
SELECT *
FROM SortOrder
ORDER BY 2
```

.

`order by [column]  order by [column]` .  case .

```
Group
-----
Total
Young
MiddleAge
Old
Male
Female
```

`order by` :

```
Select * from MyTable
Order by Group
```

.

```
Group
-----
Female
Male
MiddleAge
Old
Total
Young
```

'case'  .

```
Select * from MyTable
Order by case Group
    when 'Total' then 10
    when 'Male' then 20
    when 'Female' then 30
    when 'Young' then 40
    when 'MiddleAge' then 50
    when 'Old' then 60
    end
```

.

```
Group
-----
Total
Male
Female
```

```
Young
MiddleAge
Old
```

/ : https://riptutorial.com/ko/sql-server/topic/5332/----

| S. No | | Contributors |
|---|---|---|
| 1 | Microsoft SQL Server | Abhilash R Vankayala, Abhishek Jain, Ahmad Aghazadeh, Ahmar, Akshay Anand, alalp, Almir Vuk, Arthur D, ATC, Athafoud, BeaglesEnd, Bhanu, Biju jose, Blachshma, bluefeet, ChrisM, Christos, Community, cteski, D M, Darshak, Gidil, Gordon Bell, Greg Bray, Iztoksson, Jared Hooper, JerryOL, Job AJ, Joe Taras, John Odom, John Slegers, JonasCz, K48, kafka , Lamak, Laughing Vergil, Mahesh Dahal, Malt, Martin Smith, Matt, Matt, Max, Mihai-Daniel Virna, Mudassir Hasan, n00b, Nick, Nikolay Kostov, onupdatecascade, OzrenTkalcecKrznaric, Peter Tirrell, Phrancis, Prateek, Sam, Shaneis, Thuta Aung, Tony L., Tot Zam, Uberzen1, Umachandar - Microsoft, user_0, user2314737, VoidDemon, Zsuzsa |
| 2 | Azure SQL | Jovan MSFT |
| 3 | bcp ( ) | MarmiK |
| 4 | CASE | Laughing Vergil, RamenChef, Vikas Vaidya |
| 5 | CLUSTERED COLUMNSTORE | Jovan MSFT |
| 6 | CREATE VIEW | Almir Vuk, cteski, Edathadan Chief aka Arun, Hadi, Josh B, Robert Columbia, Tot Zam |
| 7 | DBCC | Jovan MSFT |
| 8 | DBMAIL | Phrancis |
| 9 | FOR XML PATH | bluefeet, gotqn, Keith Hall, Wolfgang |
| 10 | GROUP BY | Andy, Edathadan Chief aka Arun, Jenism, juergen d, Julien Vavasseur, Kiran Ukande, Matas Vaitkevicius, ShlomiR |
| 11 | JSON | bakedpatato, James, Jovan MSFT |
| 12 | JSON | James, Jovan MSFT |
| 13 | JSON | Jovan MSFT |
| 14 | JSON SQL | Ed Harper, Jovan MSFT, RamenChef |
| 15 | Microsoft SQL Server Studio | Bino Mathew Varghese, cteski, Sibeesh Venu |

| | | |
|---|---|---|
| 16 | MS SQL Server DDL | Matt |
| 17 | NULLs | Amir Pourmand, Hadi, Kannan Kandasamy, Kritner, Laughing Vergil, podiluska, Sean Branchaw, Zohar Peled |
| 18 | OPENJSON | James, Jovan MSFT |
| 19 | OVER | Athafoud, bluefeet, Brandon, DVT, gofr1, Lamak, Paul Bambury , RamenChef, Rowland Shaw, Sam |
| 20 | SCOPE_IDENTITY () | Dheeraj Kumar |
| 21 | SELECT | cteski, Jovan MSFT |
| 22 | SQL Server Management Studio (SSMS) | dd4711 |
| 23 | SQL Server | Jibin Balachandran, Jovan MSFT, MasterBob, Pரதீ ப், RamenChef |
| 24 | SQL Server JSON | Jovan MSFT, Mono |
| 25 | SQL Server | Kannan Kandasamy |
| 26 | SQL Server | Pரதீ ப் |
| 27 | SQLCMD | Eugene Niemand, Techie |
| 28 | SQLCMD txt | sheraz mirza |
| 29 | STUFF | Arthur D, bluefeet, Chetan Sanghani, dacohenii, Kiran Ukande, Luis Bosquez, MrE, user1690166 |
| 30 | TEMP | APH, New |
| 31 | TRY / CATCH | Jovan MSFT, ravindra, Uberzen1 |
| 32 | WHILE | lord5et, Matas Vaitkevicius, podiluska, RamenChef, Wojciech Kazior |
| 33 | Windows SQL Server | Luis Bosquez |
| 34 | | bakedpatato, Jovan MSFT |
| 35 | | RamenChef, sqlandmore.com |
| 36 | | cnayak, Kannan Kandasamy |

| | | |
|---|---|---|
| 37 | | Ahmad Aghazadeh |
| 38 | | Bharat Prasad Satyal, Edathadan Chief aka Arun, Karthikeyan, Matej, scsimon, Tab Alleman |
| 39 | | cteski, Neil Kennedy, RamenChef, Vladimir Oselsky |
| 40 | | Jovan MSFT |
| 41 | | Arif, bbrown, cteski, DForck42, Jeffrey Van Laethem, Jovan MSFT, kafka, Keith Hall, Monty Wild, SQLMason |
| 42 | | Hamza Rabah, Jovan MSFT, Tom V |
| 43 | | Oluwafemi |
| 44 | | Kritner |
| 45 | (Hekaton) | bakedpatato, Jovan MSFT |
| 46 | | Randall |
| 47 | | A_Arnold, Adam Porad, Akshay Anand, Bellash, cteski, Edathadan Chief aka Arun, JamieA, Jared Hooper, Kritner, Lamak, Mert Gülsoy, Nick, Phrancis, SHD, Siyual, Soukai, UnhandledExcepSean, Zohar Peled |
| 48 | | James, Siyual |
| 49 | | TheGameiswar |
| 50 | | cnayak |
| 51 | | dd4711 |
| 52 | | cteski, M.Ali, RamenChef |
| 53 | SQL Server (2000 - 2016) | Dan Guzman, M.Ali |
| 54 | | Jovan MSFT |
| 55 | | Laughing Vergil, Matas Vaitkevicius |
| 56 | | Ben O, Edathadan Chief aka Arun |
| 57 | | Jones Joseph, Jovan MSFT |
| 58 | | Ben Thul |
| 59 | | Akash, Daryl, Jovan MSFT, Wolfgang |

| | | |
|---|---|---|
| 60 | | Andrea, Anuj Tripathi, Baodad, Brent Ozar, dario, feetwet, James Anderson, JamieA, Jasmin Solanki, Jeffrey Van Laethem, jyao, Kritner, Laughing Vergil, LowlyDBA, Mahendra, Moshiour, Phrancis, Rhumborl, scsimon, Shiva, spaghettidba, Tot Zam, TZHX, Umachandar - Microsoft |
| 61 | SQL | Jovan MSFT |
| 62 | SQL | Jesse |
| 63 | | Jovan MSFT |
| 64 | | Ako, RamenChef |
| 65 | ID | Jeffrey Van Laethem, sqluser, Tot Zam |
| 66 | OLTP (Hekaton) | Akshay Anand, Behzad, Brandon, Jovan MSFT, Martijn Pieters |
| 67 | | A_Arnold, anon, cteski, FoxyBOA, Hadi, hatchet, Igor Micev, Jibin Balachandran, Jovan MSFT, mtb, Phrancis, Raidri, Ricardo C, Ross Presser, takrl, Zohar Peled |
| 68 | | Oluwafemi, The_Outsider, Zohar Peled |
| 69 | | APH, Keith Hall, Phrancis |
| 70 | | Abhilash R Vankayala, Abubakar Riaz, Alex, David Kaminski, dd4711, Hari K M, Moshiour, Rogerio Soares, Serg |
| 71 | | 5arx |
| 72 | | Jivan, Zohar Peled |
| 73 | | Ahmad Aghazadeh, Akshay Anand, cteski, Henrik Staun Poulsen, Martin Smith, Tom V |
| 74 | | Ken S., Matej, RamenChef |
| 75 | | Mani |
| 76 | | cteski, kolunar, New |
| 77 | | Merenix |
| 78 | - TempDb | Anuj Tripathi, RamenChef |
| 79 | | Josh Morel |
| 80 | | Rubenisme |
| 81 | | 4444, Akshay Anand, Andy, APH, Bino Mathew Varghese, |

| | | |
|---|---|---|
| | | cteski, Dean Ward, DhruvJoshi, Dileep, Gajendra, HK1, Iztoksson, Jeffrey Van Laethem, Joao Araujo, JonH, L J, Lamak , Laughing Vergil, LowlyDBA, mtb, Nikolay Kostov, OzrenTkalcecKrznaric, Phrancis, Ram Grandhi, SqlZim |
| 82 | | Abubakar Riaz, barcanoj, DVJex, Hari K M, intox, martinshort, Matas Vaitkevicius, Max, Michael Stum, n00b, Piotr Nawrot, Robert Columbia, Tot Zam, woony |
| 83 | | Edathadan Chief aka Arun, Hadi |
| 84 | | Jovan MSFT |
| 85 | | Matas Vaitkevicius |
| 86 | | Bino Mathew Varghese, feetwet, James Anderson, Kritner, LowlyDBA, S.Karras, scsimon |
| 87 | | Ahmad Aghazadeh, Akshay Anand, Ben O, Mspaja |
| 88 | | Bino Mathew Varghese, cnayak, cteski, Erik Oppedijk, Eugene Niemand, Hari K M, Jayasurya Satheesh, Matas Vaitkevicius, Nathan Skerl, Pirate X, scsimon |
| 89 | | Edathadan Chief aka Arun |
| 90 | | Benjamin Hodgson, Daniel Lemke, Max |
| 91 | | APH, beercohol, cteski, Gidil, RamenChef |
| 92 | | Akshay Anand, cnayak, cteski, Jeffrey L Whitledge, Joe Taras, Vexator |
| 93 | | cteski, Jovan MSFT, Sender |
| 94 | | andyabel, feetwet, MarmiK |
| 95 | | Kane, Phrancis |
| 96 | | DForck42 |
| 97 | | cteski, DARKOCEAN, Jovan MSFT, user_0 |
| 98 | | Ignas, Jakub Ojmucianski, Justin Rohr, Max, scsimon |
| 99 | | Zohar Peled |
| 100 | | Nick.McDermaid |
| 101 | | Phrancis |

| | | |
|---|---|---|
| 102 | | Metanormal |
| 103 | | bassrek |
| 104 | | Behzad |
| 105 | | Raghu Ariga |
| 106 | | Dan Guzman, James Anderson, John Odom, Susang |
| 107 | | Max |
| 108 | | Pat |
| 109 | / | Athafoud, bluefeet, DhruvJoshi, kolunar |
| 110 | | cnayak |
| 111 | | alalp, chrisb, cteski, ErikE |
| 112 | | Carsten Hynne, Jovan MSFT |
| 113 | / | APH, OzrenTkalcecKrznaric |