شروع کار با SQL Server

درس ششم: توابع توکار

سید کاوه احمدی

توابع کار با رشته‌ها

- `LEFT (character_expression, integer_expression)`
- `RIGHT (character_expression, integer_expression)`
- `LTRIM (character_expression)`
- `RTRIM (character_expression)`
- `LOWER (character_expression)`
- `UPPER (character_expression)`

مثال

- انتخاب ۵۰ کاراکتر اول عناوین خبر

```
SELECT
        LEFT(Title, 50)
FROM
        News
```

توابع کار با رشته‌ها

- `ASCII (character_expression)`
- `CHAR (integer_expression)`

- `UNICODE ('ncharacter_expression')`
- `NCHAR (integer_expression)`

توابع دستکاری رشته‌ها

- **CHARINDEX**(expressionToFind ,expressionToSearch [,start_location])
- **PATINDEX**('%pattern%', expression)
- **CONCAT**(string_value1, string_value2 [,string_valueN])
- **SUBSTRING**(expression ,start, length)
- **REPLACE**(string_expression, string_pattern, string_replacement)
- **LEN**(string_expression)
- **REVERSE**(string_expression)

مثال

- `SELECT CHARINDEX('abc', 'abbabcabbabc')`
  - محل اولین زیر رشته پیدا شده را باز می‌گرداند.
    - اندیس‌ها از ۱ شروع می‌شود.
      - خروجی ۴ است.

- `SELECT CHARINDEX('abc', 'abbabcabbabc', 6)`
  - از موقعیت ۶ رشته به بعد دنبال زیر رشته می‌گردد.
    - خروجی ۱۰ است.

مثال

- `SELECT PATINDEX('%bb__C%', 'abbabcabbabc')`
  - 2

- `SELECT SUBSTRING('abbabcabbabc', 3, 5)`
  - babca

تاریخ در SQL Server

Data type	Format	Range	Accuracy	Storage size (bytes)
time	hh:mm:ss[.nnnnnnn]	00:00:00.0000000 through 23:59:59.9999999	100 nanoseconds	3 to 5
date	YYYY-MM-DD	0001-01-01 through 9999-12-31	1 day	3
smalldatetime	YYYY-MM-DD hh:mm:ss	1900-01-01 through 2079-06-06	1 minute	4
datetime	YYYY-MM-DD hh:mm:ss[.nnn]	1753-01-01 through 9999-12-31	0.00333 second	8
datetime2	YYYY-MM-DD hh:mm:ss[.nnnnnnn]	0001-01-01 00:00:00.0000000 through 9999-12-31 23:59:59.9999999	100 nanoseconds	6 to 8
datetimeoffset	YYYY-MM-DD hh:mm:ss[.nnnnnnn] [+|-]hh:mm	0001-01-01 00:00:00.0000000 through 9999-12-31 23:59:59.9999999 (in UTC)	100 nanoseconds	8 to 10

توابع دریافت تاریخ سرور

- SYSDATETIME()
- SYSDATETIMEOFFSET()
- SYSUTCDATETIME()
- **GETDATE()**
- GETUTCDATE()

توابع دریافت بخش‌های مختلف تاریخ

- `DAY(date)`
- `MONTH(date)`
- `YEAR(date)`
- `DATENAME(datepart, date)`
- `DATEPART(datepart, date)`

توابع دریافت بخش‌های مختلف تاریخ

datepart	Abbreviations
year	yy, yyyy
quarter	qq, q
month	mm, m
day of year	dy, y
day	dd, d
week	wk, ww
weekday	dw, w
hour	hh
minute	mi, n
second	ss, s
millisecond	ms
microsecond	mcs
nanosecond	ns

مثال

```
SELECT
    DATENAME(year, GETDATE()),
    DATENAME(month, GETDATE()),
    DATEPART(day, GETDATE()),
    DATEPART(dayofyear, GETDATE()),
    DATENAME(weekday, GETDATE());
```

توابع دستکاری تاریخ

- `DATEDIFF(datepart, startdate, enddate)`
- `DATEADD(datepart, number, date)`

مثال

```
SELECT DATEDIFF(millisecond, GETDATE(), SYSDATETIME());
SELECT DATEDIFF(day, '2007-05-07 09:53:01', '2007-05-08 09:53:01');
SELECT DATEADD(year, 5, '2006-07-31');
```

مثال

- پرسجویی بنویسید که اخبار درج شده در ماه سوم سال ۲۰۱۴ را بازگرداند.

```
SELECT
        *
FROM
        News
WHERE
        [Date] BETWEEN '2014-03-01' AND '2014-03-31'



SELECT
        *
FROM
        News
WHERE
        YEAR([Date]) = 2014 AND
        MONTH([Date]) = 3
```

مثال

- پرسجویی بنویسید که اخبار درج شده در سی روز گذشته را بازگرداند.

```
SELECT
	*
FROM
	News
WHERE
	[Date] >= DATEADD(dd, -30, GETDATE());
```

مثال

- پرسجویی بنویسید که اخبار درج شده در ماه جاری را بازگرداند.

```
SELECT
	*
FROM
	News
WHERE
	[Date] >= DATEADD(month, DATEDIFF(month, 0, GETDATE()), 0);
```

سایر توابع تاریخ

- `ISDATE (expression)`
- `CAST(expression AS data_type [( length )])`
- `CONVERT (data_type [( length )] , expression [ , style])`

Standard Date Formats

SQL Statement	Sample Output
SELECT CONVERT(VARCHAR(20), GETDATE(), 100)	Jan 1 2005 1:29PM
SELECT CONVERT(VARCHAR(8), GETDATE(), 1) AS [MM/DD/YY]	11/23/98
SELECT CONVERT(VARCHAR(10), GETDATE(), 101) AS [MM/DD/YYYY]	11/23/1998
SELECT CONVERT(VARCHAR(8), GETDATE(), 2) AS [YY.MM.DD]	72.01.01
SELECT CONVERT(VARCHAR(10), GETDATE(), 102) AS [YYYY.MM.DD]	1972.01.01
SELECT CONVERT(VARCHAR(8), GETDATE(), 3) AS [DD/MM/YY]	19/02/72
SELECT CONVERT(VARCHAR(10), GETDATE(), 103) AS [DD/MM/YYYY]	19/02/1972
SELECT CONVERT(VARCHAR(8), GETDATE(), 4) AS [DD.MM.YY]	25.12.05
SELECT CONVERT(VARCHAR(10), GETDATE(), 104) AS [DD.MM.YYYY]	25.12.2005
SELECT CONVERT(VARCHAR(8), GETDATE(), 5) AS [DD-MM-YY]	24-01-98
SELECT CONVERT(VARCHAR(10), GETDATE(), 105) AS [DD-MM-YYYY]	24-01-1998
SELECT CONVERT(VARCHAR(9), GETDATE(), 6) AS [DD MON YY]	04 Jul 06
SELECT CONVERT(VARCHAR(11), GETDATE(), 106) AS [DD MON YYYY]	04 Jul 2006
SELECT CONVERT(VARCHAR(10), GETDATE(), 7) AS [Mon DD, YY]	Jan 24, 98
SELECT CONVERT(VARCHAR(12), GETDATE(), 107) AS [Mon DD, YYYY]	Jan 24, 1998
SELECT CONVERT(VARCHAR(8), GETDATE(), 108)	03:24:53
SELECT CONVERT(VARCHAR(26), GETDATE(), 109)	Apr 28 2006 12:32:29:253PM
SELECT CONVERT(VARCHAR(8), GETDATE(), 10) AS [MM-DD-YY]	01-01-06
SELECT CONVERT(VARCHAR(10), GETDATE(), 110) AS [MM-DD-YYYY]	01-01-2006
SELECT CONVERT(VARCHAR(8), GETDATE(), 11) AS [YY/MM/DD]	98/11/23
SELECT CONVERT(VARCHAR(10), GETDATE(), 111) AS [YYYY/MM/DD]	1998/11/23
SELECT CONVERT(VARCHAR(6), GETDATE(), 12) AS [YYMMDD]	980124
SELECT CONVERT(VARCHAR(8), GETDATE(), 112) AS [YYYYMMDD]	19980124
SELECT CONVERT(VARCHAR(24), GETDATE(), 113)	28 Apr 2006 00:34:55:190
SELECT CONVERT(VARCHAR(12), GETDATE(), 114) AS [HH:MI:SS:MMM(24H)]	11:34:23:013
SELECT CONVERT(VARCHAR(19), GETDATE(), 120)	1972-01-01 13:42:24
SELECT CONVERT(VARCHAR(23), GETDATE(), 121)	1972-02-19 06:35:24.489
SELECT CONVERT(VARCHAR(23), GETDATE(), 126)	1998-11-23T11:25:43:250
SELECT CONVERT(VARCHAR(26), GETDATE(), 130)	28 Apr 2006 12:39:32:429AM
SELECT CONVERT(VARCHAR(25), GETDATE(), 131)	28/04/2006 12:39:32:429AM

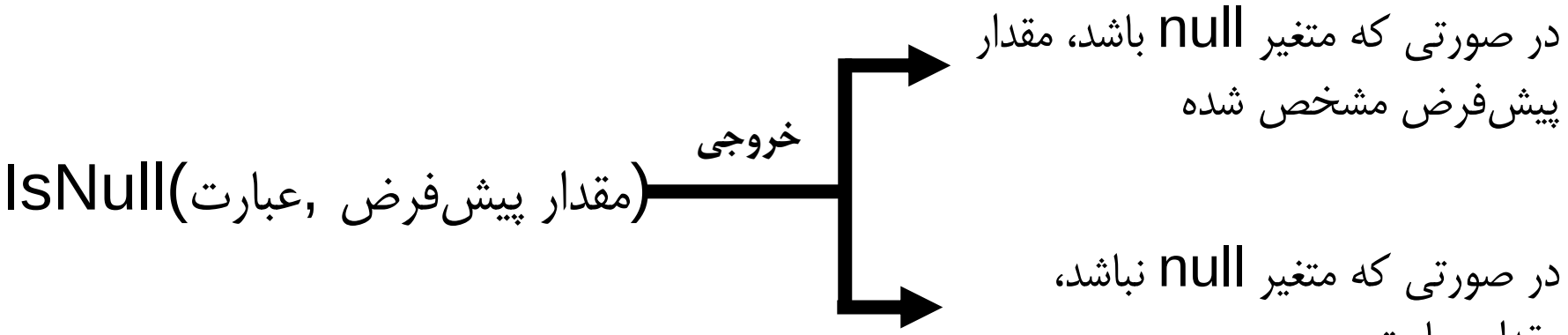
System Functions

$PARTITION	ERROR_SEVERITY
@@ERROR	ERROR_STATE
@@IDENTITY	FORMATMESSAGE
@@PACK_RECEIVED	GETANSINULL
@@ROWCOUNT	GET_FILESTREAM_TRANSACTION_CONTEXT
@@TRANCOUNT	HOST_ID
BINARY_CHECKSUM	HOST_NAME
CHECKSUM	ISNULL
CONNECTIONPROPERTY	ISNUMERIC
CONTEXT_INFO	MIN_ACTIVE_ROWVERSION
CURRENT_REQUEST_ID	NEWID
ERROR_LINE	NEWSEQUENTIALID
ERROR_MESSAGE	ROWCOUNT_BIG
ERROR_NUMBER	XACT_STATE
ERROR_PROCEDURE	

https://msdn.microsoft.com/en-us/library/ms187786%28v=sql.110%29.aspx

تابع IsNull

- بررسی Null بودن یک مقدار



مثال

- در پایگاه داده NewsSystem، عنوان خبر و نام دسته آنرا درج کنید. در صورتی که یک خبر دسته‌بندی نداشته باشد، عنوان بدون دسته درج شود.

```
SELECT
        N.Title,
        ISNULL(C.Name, 'بدون دسته')
FROM
        News N
LEFT OUTER JOIN
        Categories C
ON
        N.CategoryID = C.ID
```

IsNumeric

- اگر پارامتر ورودی از نوع یکی از نوع‌های داده‌ای زیر باشد، مقدار ۱ را باز می‌گرداند:
  - `int, numeric, bigint, money, smallint, smallmoney, tinyint, float, decimal, real`

@@IDENTITY

- مقدار آخرین کلید اصلی درج شده در پایگاه داده توسط کلاینت توسط تابع سیستمی @@IDENTITY قابل دسترس است.

@@IDENTITY

```
INSERT INTO
       Categories(Name)
VALUES
       (‘cat name');

SELECT @@IDENTITY;
```

@@IDENTITY

- @@**IDENTITY** returns the id of the last thing that was inserted by your client's connection to the database.

  Most of the time this works fine, but sometimes a trigger will go and insert a new row that you don't know about, and you'll get the ID from this new row, instead of the one you want
- **SCOPE_IDENTITY()** solves this problem. It returns the id of the last thing that *you inserted* in the SQL code *you sent* to the database. If triggers go and create extra rows, they won't cause the wrong value to get returned. Hooray
- **IDENT_CURRENT** returns the last ID that was inserted by anyone. If some other app happens to insert another row at an unforunate time, you'll get the ID of that row instead of your one.
- If you want to play it safe, always use SCOPE_IDENTITY(). If you stick with @@IDENTITY and someone decides to add a trigger later on, all your code will break.

@@rowcount

- تعداد سطرهایی که دستور DML تحت تاثیر قرار داده است.

```
UPDATE Categories SET Name = 'سیاسی' WHERE id = 100
SELECT @@rowcount
```

Ranking Functions

- ROW_NUMBER() OVER()

https://msdn.microsoft.com/en-us/library/ms189798%28v=sql.110%29.aspx

ROW_NUMBER() OVER()

```
ROW_NUMBER ( )
    OVER ( [ PARTITION BY value_expression , ... [ n ] ] order_by_clause )
```

ROW_NUMBER() OVER()

```
SELECT
        ROW_NUMBER()
                OVER (ORDER BY Id DESC),
        Title
FROM
        News N
```

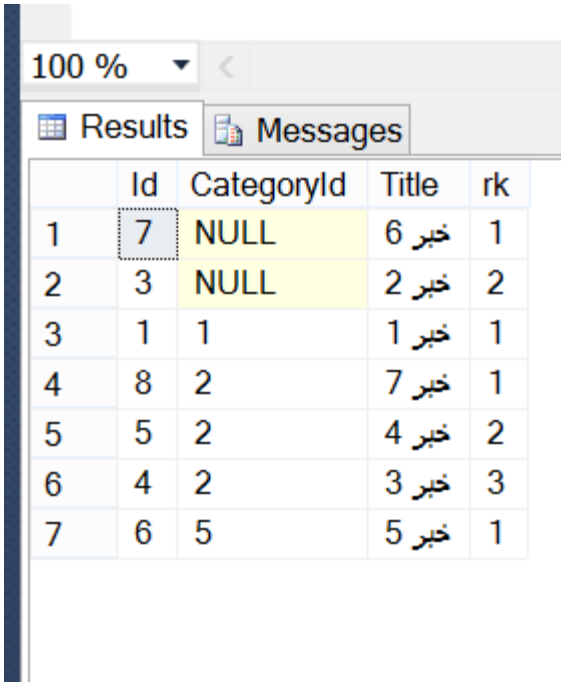

esults Messages

Id	Title	rk
8	خبر 7	1
7	خبر 6	2
6	خبر 5	3
5	خبر 4	4
4	خبر 3	5
3	خبر 2	6
1	خبر 1	7

ROW_NUMBER() OVER()

- رتبه هر خبر (بر اساس تاریخ) در هر دسته می‌خواهیم:

```
SELECT
        n.Id,
        n.CategoryId,
        n.Title,
        ROW_NUMBER()
                OVER(
                        PARTITION BY
                                n.CategoryId
                        ORDER BY
                                n.Date DESC
                ) AS rk
        FROM
News n
```



	Id	CategoryId	Title	rk
1	7	NULL	خبر 6	1
2	3	NULL	خبر 2	2
3	1	1	خبر 1	1
4	8	2	خبر 7	1
5	5	2	خبر 4	2
6	4	2	خبر 3	3
7	6	5	خبر 5	1

تمرین

- آخرین خبر درج شده در هر دسته را می‌خواهیم:

```
SELECT
	*
FROM
	(SELECT
		n.Id,
		n.CategoryId,
		n.Title,
		ROW_NUMBER()
			OVER(
				PARTITION BY
					n.CategoryId
				ORDER BY
					n.Date DESC
			) AS rk
	FROM
		News n
	) AS tbl
WHERE
	rk=1
```

توابع ریاضی

- `FLOOR(numeric_expression)`
- `CEILING(numeric_expression)`
- `ROUND(numeric_expression, length [ ,function ])`

توابع ریاضی

- `ABS(numeric_expression)`
- `POWER(float_expression, y)`
- `RAND([ seed ])`
- `PI()`
- `LOG(float_expression [, base ])`
- `LOG10(float_expression)`
- `SIGN(numeric_expression)`
- `SQRT(float_expression)`
- `SQUARE(float_expression)`

توابع ریاضی

- `SIN(float_expression)`
- `COS(float_expression)`
- `TAN(float_expression)`
- `COT(float_expression)`
- `ASIN(float_expression)`
- `ACOS(float_expression)`
- `ATAN(float_expression)`
- `ATN2(float_expression, float_expression)`

Built-in Functions (Transact-SQL)

- http://technet.microsoft.com/en-us/library/ms174318.aspx

مطالعه بیشتر

- نوع داده‌ای timestamp: موارد استفاده، توابع و کاربردها

```
SELECT
	CAST(0x0000000000000000 AS DATETIME),
	CAST(GETDATE() AS TIMESTAMP)
```

- نوع داده‌ای uniqueidentifier: موارد استفاده، توابع و کاربردها

```
SELECT
	NEWID()
```

- بخش‌های تاریخ شامل Tzoffset و ISO_WEEK

مطالعه بیشتر

```
--database objects
SELECT
	*
FROM
	sys.objects



--indexes in deep
SELECT
	*
FROM
	sys.indexes
WHERE
	object_id = OBJECT_ID('Object_Name')
```