A drawing of a face

Description automatically generated

**BACKEND SYSTEM INTEGRATOR**

**BASI DI DATI - SQL**

**Docente: Alberto Venditti**

**Titolo argomento: SQL Server datatypes**

SQL Server Datatypes **in breve...**

*Alberto Venditti*

# Datatypes numerici

SQL Server fornisce datatypes numerici di quattro categorie: interi (bigint, int, smallint, tinyint), floating-point (float, real), esatti (numeric, decimal) e valuta (money, smallmoney).

## 1.1 Datatypes interi

|  |  |  |
| --- | --- | --- |
| ***Datatype*** | ***Range*** | ***Dimensione*** |
| bigint | ±9.223.372.036.854.775.807 ±263 | 8 bytes |
| int | ±2.147.483.647 ±231 | 4 bytes |
| smallint | ±32.767 ±215 | 2 bytes |
| tinyint | 0÷255 0÷28 | 1 byte |

## 1.2 Datatypes floating-point

|  |  |  |  |
| --- | --- | --- | --- |
| ***Datatype*** | ***Range*** | ***Precisione*** | ***Dimensione*** |
| real | ±1,18E-38  ±3,40E+38 | fino a 7 cifre | 4 bytes |
| float | ±2,23E-308  ±1,79E+308 | fino a 15 cifre | 8 bytes |
| float(*n*) |  |  |  |

Per l'immissione dei valori, il separatore della parte decimale è sempre il punto, a prescindere dalla lingua installata.

Per compatibilità con l'ANSI SQL, è stato mantenuto il datatype float(*n*) (con *n* precisione, opzionale), che viene memorizzato come un real con 1 ≤ *n*≤ 7, come un float con 8 ≤ *n*≤ 15.

## 1.3 Datatypes esatti

|  |  |  |  |
| --- | --- | --- | --- |
| ***Datatype*** | ***Range*** | ***Precisione*** | ***Dimensione*** |
| numeric(*p*,*s*)  decimal(*p*,*s*) | ±1E-38  ±1E+38 | fino a 28 cifre | 2÷17 bytes |

I due datatypes sono identici. I parametri *p* ed *s* specificano, rispettivamente, *precision* (numero di cifre significative, *p*≤ 28) e *scale* (numero di cifre decimali, *s*≤ *p*).

Esempio: decimal(7,2) ⇒ ± 99.999,99

## 1.4 Datatypes valuta

|  |  |  |
| --- | --- | --- |
| ***Datatype*** | ***Range*** | ***Dimensione*** |
| money | ±922.337.203.685.477,5807 | 8 bytes |
| smallmoney | ±214.749,3647 | 4 bytes |

Per l'inserimento, un valore numerico di tipo valuta deve essere preceduto dal simbolo di dollaro (a prescindere dalla lingua installata).

Esempio: $1234.50

# Datatypes stringa

|  |  |  |
| --- | --- | --- |
| ***Datatype*** | ***Dimensione*** | ***Note*** |
| char(n) | max 8000 caratteri | lunghezza fissa,  completato con blanks |
| varchar(n) | max 8000 caratteri | lunghezza variabile,  non completato con blanks |
| text  varchar(max) | max 2 miliardi di caratteri  (2 Gb) |  |

I valori di tipo stringa devono essere inseriti fra apici singoli o doppi. Valori di lunghezza superiore alle dimensioni del campo vengono troncati senza fornire alcun messaggio d'errore.

Colonne definite come char(n) NULL vengono trattate come varchar(n).

# Datatypes stringa Unicode

|  |  |  |
| --- | --- | --- |
| ***Datatype*** | ***Dimensione*** | ***Note*** |
| nchar(n) | max 4000 caratteri | lunghezza fissa |
| nvarchar(n) | max 4000 caratteri | lunghezza variabile |
| ntext  nvarchar(max) | max 1 miliardo di caratteri |  |

Permettomo l'uso di set di caratteri di più lingue. Ogni carattere Unicode è memorizzato su due bytes.

I valori di tipo stringa Unicode devono essere inseriti fra apici singoli o doppi. Valori di lunghezza superiore alle dimensioni del campo vengono troncati senza fornire alcun messaggio d'errore.

# Datatypes data/ora

|  |  |  |  |
| --- | --- | --- | --- |
| ***Datatype*** | ***Range*** | ***Dimensione*** | ***Precisione*** |
| datetime | 01/01/1753  31/12/9999 | 8 bytes | 3 millisecondi |
| datetime2 | 01/01/0001  31/12/9999 | 6-8 bytes | 100 nanosecondi |
| smalldatetime | 01/01/1900  06/06/2079 | 4 bytes | 1 minuto |
| time | 00:00:00.999  23:59:59.999 | 3-5 bytes | 100 nanosecondi |
| date | 01/01/0001  31/12/9999 | 3 bytes | 1 giorno |

Il datatype datetime parte dal 1753 poiché su date precedenti non potrebbe essere garantita l'accuratezza, visto il cambiamento di calendario avvenuto nel settembre 1752.

L'inserimento di date richiede, come per le stringhe, l'immissione di apici semplici o doppi.

Sono riconosciuti parecchi formati di data differenti, elencati nella seguente tabella. Si noti: la differente accuratezza dei vari datatype; l'immissione mm/gg/aa (formato *mdy*); l'interpretazione dell'anno come appartenente al XXI secolo quando fornito su due cifre ed inferiore a 50 (windowing configurabile).

|  |  |  |
| --- | --- | --- |
| **Valore inserito** | **datetime** | **smalldatetime** |
| Sep 3, 1996  13:17:35.332 | Sep 3 1996 1:17:35:333PM | Sep 3 1996 1:18PM |
| 9/3/96 1pm | Sep 3 1996 1:00:00:000PM | Sep 3 1996 1:00PM |
| 9.3.96 13:00 | Sep 3 1996 1:00:00:000PM | Sep 3 1996 1:00PM |
| 3 sep 96 13:00 | Sep 3 1996 1:00:00:000PM | Sep 3 1996 1:00PM |
| 13:25:19 | **Jan 1 1900** 1:25:19:000PM | **Jan 1 1900** 1:25PM |
| 3 september 96 | Sep 3 1996 **12:00:00:000AM** | Sep 3 1996 **12:00AM** |
| 4/5/76 | Apr 5 **19**76 12:00:00:000AM | Apr 5 **19**76 12:00AM |
| 4/15/46 | Apr 15 **20**46 12:00:00:000AM | Apr 15 **20**46 12:00AM |

L'mmissione di date è possibile nei formati: *mdy*, *dmy*, *ymd*, *ydm*, *myd*, *dym*. Per impostare un formato di immissione differente da quello predefinito (il quale dipende dalla lingua dell’utente connesso), si utilizza l'opzione DATEFORMAT. L'impostazione di DATEFORMAT è *session-level*.

Esempio: set dateformat dmy

go

insert TabProva values('24/01/70')

go

# Datatypes binari

|  |  |  |
| --- | --- | --- |
| ***Datatype*** | ***Dimensione*** | ***Note*** |
| binary(n) | max 8000 bytes | lunghezza fissa, completato con zeri |
| varbinary(n) | max 8000 bytes | lunghezza variabile,  non completato con zeri |
| image  varbinary(max) | max 2 Gb |  |

Le colonne con datatype binario sono gestite in modo analogo ai datatype stringa (anche dal punto di vista del NULL). L'inserimento di un campo binario avviene specificando i valori dei bytes in esadecimale.

Esempio: binfield binary(4) not null

0xa134c2ff

Valori inseriti: A1, 34, C2, FF

Ovvero, in decimale: 161, 52, 194, 255

# Altri datatypes

### Timestamp

Datatype binario che permette di identificare la "versione" di ogni record. Una colonna timestamp viene aggiornata automaticamente da SQL Server (utilizzando l'identificatore univoco della transazione tratto dal transaction log). Permette di gestire l'*optimistic locking*. Una tabella può contenere al massimo una colonna timestamp.

### Cursor

Datatype che consente di mantenere un reference ad un cursore via codice.

### Uniqueidentifier

Datatype che permette di memorizzare un "globally unique identifier" (GUID). Vedi anche funzione NEWID().

### Bit

Datatype utilizzabile come *flag* logico. Può contenere esclusivamente il valore 0 oppure il valore 1; non è indicizzabile.

### Sysname

Corrisponde ad un datatype nvarchar(128) avente tutte le restrizioni tipiche dei nomi per gli oggetti, le colonne e i database in SQL Server.

### sql\_variant

Datatype utilizzabile per memorizzare dati di tipo diverso (una colonna di tipo sql\_variant può contenere, in record diversi, dati di tipo diverso.

### table

Datatype utilizzabile in programmazione per memorizzare temporaneamente result set da sottoporre ad ulteriori elaborazioni (questa tecnica ha alcuni elementi in comune con l'utilizzo di tabelle temporanee in tempdb).