- Erweitertes Format: `1985-04-12T23:20:50+01:00`

Am SQL Server ist hier das 'T' in der Formatierung anzugeben.

Aber auch die Datentypen der Datenbankhersteller können unterschiedliche Wertebereiche aufweisen, die SQL Dialekte können also vom Standard abweichen. Hier lauert bereits die erste Fehlerquelle, wenn Daten etwa über Schnittstellen ausgetauscht werden und am jeweiligen System andere Datenbanken im Einsatz sind.

Hier etwa die Unterschiede von Datum und Zeit von MySQL 5.x

Datentyp	Wertebereich	Genauigkeit
datetime	01.01.1000 00:00:00 bis 31.12.9999 23:59:59	1 Sekunde
date	01.01.1000 bis 31.12.9999	1 Tag
time	–838:59:59 bis 838:59:59	1 Sekunde
year	1901 bis 2055	1 Jahr

SQL Server Numerische Datentypen

Datentyp	Wertebereich	Wertebereich
bit	0	1
tinyint	0	255
smallint	-32.768	32.767
int	-2.147.483.648	2.147.483.647
bigint	-9.223.372.036.854.775.808	9.223.372.036.854.775.808
decimal, numeric	$-10^{\pm 38}+1$	$10^{\pm 38}-1$
smallmoney	-214.748,3648	214.748,3647
money	-922.337.203.685.477,5808	922.337.203.685.477,580
float	$-3.40 * 10^{\pm 308}$	$3.40 * 10^{\pm 308}$
real	$-1.79 * 10^{\pm 38}$	$1.79 * 10^{\pm 38}$

Bei decimal und money gibt es unterschiedliche Ergebnisse durch Runden. Die Entscheidung des passenden Datentypes ist also bei einer Finanzanwendung eine Entscheidende.

```
DECLARE @dOne decimal(19,4) = 1,
@dThree decimal(19,4) = 3,
@mOne money = 1,
@mThree money = 3;

SELECT @dOne / @dThree * @dThree AS DecimalResult,
@dOne * @dThree / @dThree AS ReorderedDecimalResult,
@mOne / @mThree * @mThree AS MoneyResult,
@mOne * @mThree / @mThree AS ReorderedMoneyResults;
GO
```

DecimalResult	ReorderedDecimalResult	MoneyResult	ReorderedMoneyResults
1.000000	1.000000	0,9999	1,00