FLOAT and DOUBLE PRECISION

FLOAT data types are used to store values with significant decimals. The following FLOAT types are supported:

Type	Size	Value range
Float	4 bytes	7 significant decimals; -3.4 x 10^-38 to 3.4 x 10^38
Double Precision	8 bytes	15 significant decimals; -1.7 x 10^-308 to 1.7 x 10^308

A column with the defined data type FLOAT can store a single-precision figure with up to 7 significant decimals. The decimal point can float between all seven of these digits. If a number with more than 7 decimal places needs to be saved, decimals beyond the seventh position are truncated. FLOAT columns require 4 bytes of storage.

A column with the defined data type DOUBLE PRECISION can store numbers with 15 significant decimals. This uses 8 bytes of storage. As with the FLOAT column, the decimal point can float within the column. The DOUBLE PRECISION data type is implemented in the majority of InterBase® platforms as a 64 bit number.

FLOAT types can be implemented for any calculative operations. They offer an optimal performance and sufficient range of values. It is possible to specify the display format of a FLOAT field under Environment Options / Grid / Display Formats.

The DOUBLE PRECISION data type can be written as follows:

```
DOUBLE PRECISION
DOUBLE
```

The main advantage of a DOUBLE PRECISION data type is the large number of decimal places e.g. 1/3 in DOUBLE PRECISION would be 0,33333333333333 in NUMERIC(18,4) it would be 0,3333. Please note: up until dialect 1 NUMERIC and DOUBLE PRECISION were identical i.e. an SQL with the data type NUMERIC(15,2) results in the following:

Result with dialect 1:

```
CREATE TABLE TEST(WERT NUMERIC(15,2));
INSERT INTO TEST(WERT) VALUES(100);
SELECT * FROM TEST;  result 100
UPDATE TEST SET WERT=WERT/3;
SELECT * FROM TEST;  result 33,33
UPDATE TEST SET WERT=WERT*3;
SELECT * FROM TEST;  result 100
```

Result with dialect 3:

```
CREATE TABLE TEST(WERT NUMERIC(15,2));
INSERT INTO TEST(WERT) VALUES(100);
SELECT * FROM TEST;  result 100
UPDATE TEST SET WERT=WERT/3;
```

```
SELECT * FROM TEST;  result 33,33
UPDATE TEST SET WERT=WERT*3;
SELECT * FROM TEST;  result 99,99
```

Since dialect 3 NUMERIC data is rounded according to commercial rounding rules; up to dialect 1 NUMERIC data is rounded according to technical rounding rules.