# Firebird 4: новые арифметические типы данных

Александр Пешков
Firebird Foundation

FIREBIRD CONF 2023 | РЕДСОФТ

# Новые типы данных

- DECFLOAT [ варианты точности DECFLOAT(16) / DECFLOAT(34) ]
- INT128 [ базовый тип для Numeric/Decimal(38, x) ]

# Повышение точности вычислений

**IEEE 754r (2008)**

- двоичное представление (мантисса представлена как двоичное число, основание экспоненты - 2)
- двоично-десятичное представление (мантисса представлена как двоично-десятичное число, основание экспоненты - 10)

# Новые типы данных

```
create or alter procedure rnd (osn int)
returns (f type of fval, s type of fval, d double precision)
as
declare variable n int;
begin
    f = 1.0 / cast(osn as type of fval);
    s = 0.0;
    n = 0;
    while (n < 20*3600*osn) do begin
        s = s + f;
        n = n + 1;
    end
    d = cast(n as double precision) / cast(osn as double precision);
    d = abs(d - cast(s as double precision)) * 100.0 / d;
    suspend;
end^
```

# Повышение точности вычислений

FLOAT

| Шаг приращения времени (доли секунды) | Время в результате суммирования (секунды) | Погрешность |
|---|---|---|
| 0.1 | 72520.844 | 0.72% |
| 0.125 | 72000.000 | 0 |

# Повышение точности вычислений

**DOUBLE PRECISION**

| Шаг приращения времени (доли секунды) | Время в результате суммирования (секунды) | Погрешность |
|---|---|---|
| 0.1 | 71999.9999997 | 4.12e-12 |
| 0.125 | 72000.0000000 | 0 |

# Повышение точности вычислений

**DECFLOAT(16)**

| Шаг приращения времени (доли секунды) | Время в результате суммирования (секунды) | Погрешность |
|---|---|---|
| 0.1 | 72000.0000000 | 0 |
| 0.125 | 72000.0000000 | 0 |

# Повышение точности вычислений

**DECFLOAT(16)**

| Шаг приращения времени (доли секунды) | Время в результате суммирования (секунды) | Погрешность |
|---|---|---|
| 0.1 | 72000.0000000 | 0 |
| 0.125 | 72000.0000000 | 0 |
| 1/7 (0.142857142857...) | 72000.0000012 | 1.71e-11 |

# Функции работающие с DECFLOAT

- **Арифметика**
  ( + - * / )
- **Агрегатные и статистические**
  (например, SUM, AVG, STDDEV, CORR)
- **Стандартные**
  (например ABS, SIGN, CEILING, FLOOR, EXP, LOG, POWER, SQRT)
  - без тригонометрии
- **Специальные**
  (COMPARE_DECFLOAT, TOTALORDER, QUANTIZE, NORMALIZE_DECFLOAT)

# Специальные функции

- **COMPARE_DECFLOAT – сравнение с учётом точности и отбрасыванием специальных (NAN и тп) значений**

0 – равно, 1 – меньше, 2 – больше,
3 – одно или оба специальные

```
COMPARE_DECFLOAT(2.17, 2.170) => 2
```

- **TOTALORDER – второй вариант сравнения с учётом точности и специальных значений**

```
NORMALIZE_DECFLOAT(12.00) => 12
NORMALIZE_DECFLOAT(120) => 1.2E+2
```

# Специальные функции

- **QUANTIZE(VAL, PICTURE) – второй аргумент является шаблоном для представления первого**

```
QUANTIZE(1234, 9.999) => 1234.000
```

- **NORMALIZE_DECFLOAT(VAL) – нормальное представление числа**

```
NORMALIZE_DECFLOAT(12.00) => 12
NORMALIZE_DECFLOAT(120) => 1.2E+2
```

# Использование 128-битных целых

- Везде где можно использовать 64-битные
- В промежуточных результатах (AVG)
- Увеличение разрядности при умножении / делении для борьбы с переполнением (SUM)
- Использование DECFLOAT(34) вместо DOUBLE PRECISION

```
LN(BIGINT) => DOUBLE PRECISION
LN(INT128) => DECFLOAT(34)
```

# Программный доступ к данным

- **IUtil – интерфейс служебных вызовов**
- **Доступ к DECFLOAT:**

```
IDecFloat16* getDecFloat16(IStatus* status)
IDecFloat34* getDecFloat34(IStatus* status)
```

- **Доступ к INT128:**

```
IInt128* getInt128(IStatus* status)
```

# Программный доступ к данным

- **IDecFloat16 – интерфейсы для доступа к значениям формата DECFLOAT:**

```
void toBcd(const DEC* from,
        int* sign, unsigned char* bcd, int* exp);
    void toString(IStatus* status, const DEC* from,
        unsigned bufferLength, char* buffer);
    void fromBcd(int sign, const unsigned char* bcd, int exp,
        DEC* to);
    void fromString(IStatus* status, const char* from, DEC* to);

    const unsigned int BCD_SIZE = 16 / 34;

DEC:  FB_DEC16 | FB_DEC34
```

# Программный доступ к данным

- **IDecFloat34 – интерфейсы для доступа к значениям формата DECFLOAT:**

```
void toBcd(const FB_DEC34* from,
        int* sign, unsigned char* bcd, int* exp);
    void toString(IStatus* status, const FB_DEC34* from,
        unsigned bufferLength, char* buffer);
    void fromBcd(int sign, const unsigned char* bcd, int exp,
        FB_DEC34* to);
    void fromString(IStatus* status, const char* from,
        FB_DEC34* to);

    const unsigned int BCD_SIZE = 34;
    const unsigned int STRING_SIZE = 43;
```

# Программный доступ к данным

- IInt128 – интерфейс для доступа к значениям формата INT128:

```
void toString(IStatus* status, const FB_I128* from, int scale,
      unsigned bufferLength, char* buffer);
   void fromString(IStatus* status, int scale, const char* from,
      FB_I128* to);

   const unsigned int STRING_SIZE = 46;
```

# Обратная совместимость

**Старое клиентское ПО не способно работать с новыми типами данных.**

```
SQL: SET BIND OF { type-from }
  TO { type-to | LEGACY | NATIVE };

DPB: isc_dpb_set_bind

Config: DataTypeCompatibility
```

# Обратная совместимость: SQL

```
SET BIND OF { type-from }
  TO { type-to | LEGACY | NATIVE };
```

type-from / type-to – допустимы неполные типы данных

LEGACY – автоматический подбор наиболее подходящего типа данных

NATIVE – отменить преобразование

# Обратная совместимость: DPB

```
isc_dpb_set_bind, <length-byte>,
‘decfloat to varchar;int128 to varchar’

DECFLOAT(16) => VARCHAR(23)
DECFLOAT(34) => VARCHAR(42)
INT128 => VARCHAR(47)
```

# Обратная совместимость: config

```
DataTypeCompatibility = 3.0
```

- возможно задать в firebird.conf и databases.conf
- обеспечивает преобразование в legacy

```
DECFLOAT(16) => DOUBLE PRECISION
DECFLOAT(34) => DOUBLE PRECISION
INT128 => BIGINT
```

# Обратная совместимость

## Приоритет: низший - config

```
DataTypeCompatibility = 3.0

DECFLOAT(16) => DOUBLE PRECISION
DECFLOAT(34) => DOUBLE PRECISION
INT128 => BIGINT
```

# Обратная совместимость

## Приоритет: выше - DPB

```
DataTypeCompatibility = 3.0
isc_dpb_set_bind, ‘decfloat to varchar’

DECFLOAT(16) => VARCHAR(23)
DECFLOAT(34) => VARCHAR(42)
INT128 => BIGINT
```

# Обратная совместимость

## Приоритет: наивысший - SQL

```
DataTypeCompatibility = 3.0
isc_dpb_set_bind, ‘decfloat to varchar’
SET BIND OF decfloat(16) TO native;

DECFLOAT(16) => DECFLOAT(16)
DECFLOAT(34) => VARCHAR(42)
INT128 => BIGINT
```

# Задай свой вопрос спикеру в телеграм-чате


Александр Пешков

Ведущий разработчик
Firebird Foundation