



Firebird White Paper

Firebird 4 and IBExpert

Fikret Hasovic, July 2021

Following years of development, on June 01, 2021, the Firebird Project announced the general availability of Firebird 4.0 — the latest major release of the Firebird relational database.

Firebird 4.0 introduced new data types and many improvements, but without any radical changes to the architecture or operation. Some of the most important are:

- Built-in logical replication
- Extended length of metadata identifiers (up to 63 characters)
- New INT128 and DECFLOAT data types, longer precision for NUMERIC/DECIMAL data types
- Support for international time zones
- Configurable time-outs for connections and statements
- Pooling of external connections
- Batch operations in the API
- Built-in cryptographic functions
- New ODS (version 13) with new system and monitoring tables
- Maximum page size increased to 32KB

A full list of changes is available in the Release Notes, as well as a complete Language Reference. Download kits are available at: <https://firebirdsql.org/en/firebird-4-0-0/>.

I am not going to describe how to install Firebird 4.0. I would recommend installation from the zip archive, as the fastest and most flexible installation. One of the advantages of the installation from the zip archive is the ability to easily install several Firebird versions simultaneously on the same computer. Just keep in mind that the `SYSDBA` user is not created automatically and should be created using the standard interactive query tool `isql.exe` (recommended), or using the security database management tool `gsec.exe`.

However, feel free to use your usual installation procedure.

By default, Firebird runs in *SuperServer* mode (recommended). To change the architecture/mode, it is necessary to change the parameter `ServerMode` in `firebird.conf`. If you have used Classic or SuperClassic in 3.0, and you have specific reasons for that, then of course continue using it.



Uncomment the *ServerMode* (by removing the symbol #) in the `firebird.conf` file and set one of the following: Super, SuperClassic or Classic.

```
ServerMode = Classic
```

However, from our point of view, only reason for setting server mode to classic is using embedded with multiple applications, but your reason may be different.

Firebird 4.0 introduced new date-time data types with time zones support. Even if you don't need or plan to use data types with time zone support, keep in mind that `CURRENT_TIMESTAMP` and `CURRENT_TIME` now return data types with time zones. To enable transition of the legacy code, it is necessary to enable the compatibility mode, which allows transparent conversion of types with time zones to the types without time zones. However, note that this conversion will work incorrectly if the connection default time zone is not set properly.

Usually, the time zone of the connection is set on the client side. If the time zone is not set on the client side, the time zone of the operating system will be used. However, you can specify the default time zone using the `firebird.conf` parameter `DefaultTimeZone`:

```
DefaultTimeZone = Europe/Berlin
```

Firebird 4.0 introduced new On-Disk Structure (ODS) number 13.0. You may convert your database to the new Firebird 4.0 format using *gbak* – first perform the backup with the previous Firebird version, then restore on Firebird 4.0. But before doing that you should check and fix any incompatibilities between the Firebird versions.

Incompatibilities at SQL level may usually happen with database objects (PSQL procedures and functions) and with DSQL queries.

Here are some common problems at SQL level that you may need to fix before moving to the new ODS:

- New reserved words
- Column names in PSQL cursors
- New data types
- Time and date literals

A complete list of incompatibilities is available in the [Firebird 4.0 Release Notes 4.0, "Compatibility Issues"](#) chapter.

There are also new system tables added in ODS13; consult [the documentation](#) for details.

One very important thing to keep in mind is that support for the external function (UDF) feature is deprecated in Firebird 4. So UDFs cannot be used with the default configuration, since the



parameter `UdfAccess` in the `firebird.conf` is set to `None` and the UDF libraries `ib_udf` and `fbudf` have been removed from the distribution.

Most of the functions in those libraries were already deprecated in previous Firebird versions and replaced with built-in functions. Replacements for a few of the remaining functions are now available, either in a new library of user-defined routines (UDRs), or as a script with conversions to PSQL stored functions.

The Firebird 4 distribution contains a script to migrate those UDF declarations.

However, if you still want to use UDFs, you must change the `firebird.conf` parameter

```
UdfAccess = Restrict UDF
```

After preparation you can try to upgrade your database(s) using the `gbak` tool.

As a first step you should make a backup copy of your database using your current Firebird version

```
gbak -b -g -V -user <username> -pas <password> -se <service>  
<database> <backup_file> -Y <log_file>
```

In the next step you need to restore this backup using Firebird 4

```
gbak -c -v -user <username> -pas <password> -se <service>  
<backup_file> <database_file> -Y <log_file>
```

Important note: the `-v` and `-Y` options should be used so that you can see in the log file what went wrong during the restore process, since migration is not always successful at the first attempt.

Note that this log will not indicate incompatibilities at SQL level, because metadata objects are not recompiled (altered) during the restore. If any procedure, trigger or view contains incompatibilities, then only altering this object will raise a Firebird error. To fix the database and remove these errors you need to extract the database with the previous Firebird version, and try to create an empty database from this script using Firebird 4, fixing SQL errors one by one.

Following the restore you may see the following warnings:

```
gbak: WARNING:function floor is not defined  
gbak: WARNING: module name or entrypoint could not be found
```

This means that you have an UDF function declared in the database, but the library is missing, since those libraries are deprecated and have been removed in Firebird 4.0. If you only used the `ib_udf` and `fbudf` libraries, you may replace them with the built-in functions or their safe UDR counterparts from `udf_compat.dll`. This can be done using the script

```
C:\Program Files\Firebird\Firebird_4_0\misc\upgrade\v4.0\udf_replace.sql
```

You can execute the following command

```
isql -user sysdba -pas masterkey -i udf_replace.sql {your-database}
```



The Firebird 4 client library `fbclient.dll` is compatible with previous versions at API level, but some compatibility problems could exist for some SQL queries.

As a side-effect of implementing the internal 128-bit integer data type, some improvements were made to the way Firebird handles the precision of intermediate results from calculations involving long `NUMERIC` and `DECIMAL` data types. In prior Firebird versions numerics, backed internally by the `BIGINT` data type (i.e. with precision between 10 and 18 decimal digits), were multiplied/divided using the same `BIGINT` data type for the result, which could cause overflow errors due to the limited precision available. In Firebird 4, such calculations are performed using 128-bit integers, thus reducing the possibilities of unexpected overflows.

For example, some expressions can return new data types that your application cannot process without change, and such alterations can be time-consuming. Or you may need to alter the code of the data access components. To simplify migration to new versions, you can use and set the `DatatypeCompatibility` parameter to the compatibility mode with the required version in `firebird.conf` or `databases.conf`:

```
DatatypeCompatibility = 3.0
```

This is the quickest way to achieve compatibility with the new data types. You don't need to add all new data types at once, so it is possible to add one data type first, then another, etc, and you can set the data type binding of the types that your applications still do not support.

For example, you have added time zone support to your application, but still do not support `INT128` and `DECFLOAT`. In this case you create trigger like this one:

```
create or alter trigger tr_ac_set_bind
on connect
as
begin
    set bind of int128 to legacy;
    set bind of decfloat to legacy;
end
```

Very important: Firebird 4 not only introduces Read Consistency for Statements in Read-Committed Transactions, but also makes it a default mode for all `READ COMMITTED` transactions, regardless of their `RECORD VERSION` or `NO RECORD VERSION` properties. This has been done to comply with the SQL specification. Be aware that this new behavior may have some side effects. The most important are so-called "restarts" on update conflicts.

Another important behavior is that active cursors in `READ COMMITTED READ CONSISTENCY` transactions prevent garbage collection even in Read Only mode. It is now recommended to stop using single long-running transactions `READ COMMITTED READ ONLY`, and divide them into several transactions, each of them remaining active as long as necessary.



If you find that `READ CONSISTENCY` mode is undesirable for any reason, set the configuration parameter `ReadConsistency` to the previous behavior.

Also, keep in mind that if an `INSERT` statement contains a `RETURNING` clause which refers to columns of the underlying table, the appropriate `SELECT` privilege must be granted.

Inside IBExpert Firebird 4.0 Support

We at **IBExpert** are working hard to provide full Firebird 4.0 compatibility.

Since IBExpert version 2017.10.01 and Firebird 4 Alpha, IBExpert supports the following:

- `DECFLOAT`, `BINARY`, `VARBINARY` data types.
- `DECFLOAT` types are now bound to `CHAR`. `SET DECFLOAT BIND CHAR` is performed just after the connection to database is established.
- Parameter tips for `COMPARE_DECFLOAT`, `NORMALIZE_DECFLOAT`, `TOTALORDER`, `QUANTIZE` functions in code editors.
- Extensions to the `IDENTITY` type in the Table and Column editors (autoincrement fields).
- `SQL SECURITY` clause is supported in the corresponding object editors (tables, views, triggers, procedures, packages).
- Extract Metadata supports new data types, `SQL SECURITY` clause and extensions to the `IDENTITY` type.

Since then, more features have been added gradually, such as:

Support of `NUMERIC/DECIMAL` data types with a maximum precision of 34 digits (Firebird 4) in the **Database object editors**; support for cumulative roles and default roles in the **User Manager**; initial support of time zones, support of `BINARY/VARBINARY` data types; support of Firebird 4 functions in the parameters tips in **Code editors**; support of Firebird 4 syntax extensions in **PSQL object editors**; support of management statements in the **PSQL debugger**; support of `DECFLOAT` values in **Export data into a script**; support of `TIME/TIMESTAMP WITH TIME ZONE` columns in **Test Data Generator**; support of system privileges in the **Grant Manager**; support of `DECFLOAT` and `NUMERIC(38,x)` variables and parameters in the **PSQL Debugger**; support of `DECFLOAT` and `NUMERIC(38,x)` parameters in the **SQL Editor**; bug fixes for missing `SQL SECURITY` clause in lazy mode in the **Trigger Editor** and for the missing `USAGE` privileges while working with a Firebird 4 database in **Grant Manager**; initial support of `DECFLOAT/NUMERIC(38)` fields in the **Test Data Generator**; support of `DECFLOAT` and `NUMERIC(38)` fields; sorting, filtering and aggregate

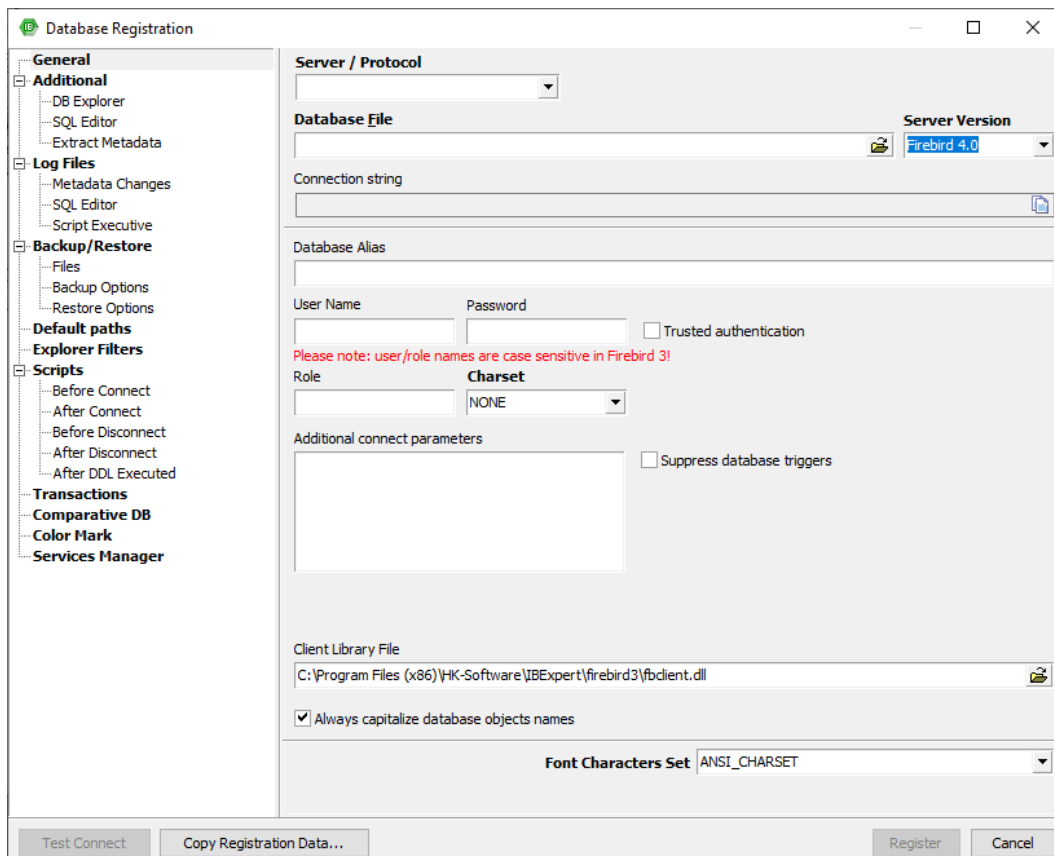


functions implemented in the **Data Grid**; parameter hints for new Firebird 4 functions (`HEX_ENCODE`, `HEX_DECODE`, `MAKE_DBKEY`, `RDB$GET_TRANSACTION_CN`) have been added, as well as highlighting of new Firebird 4 keywords and reserved words.

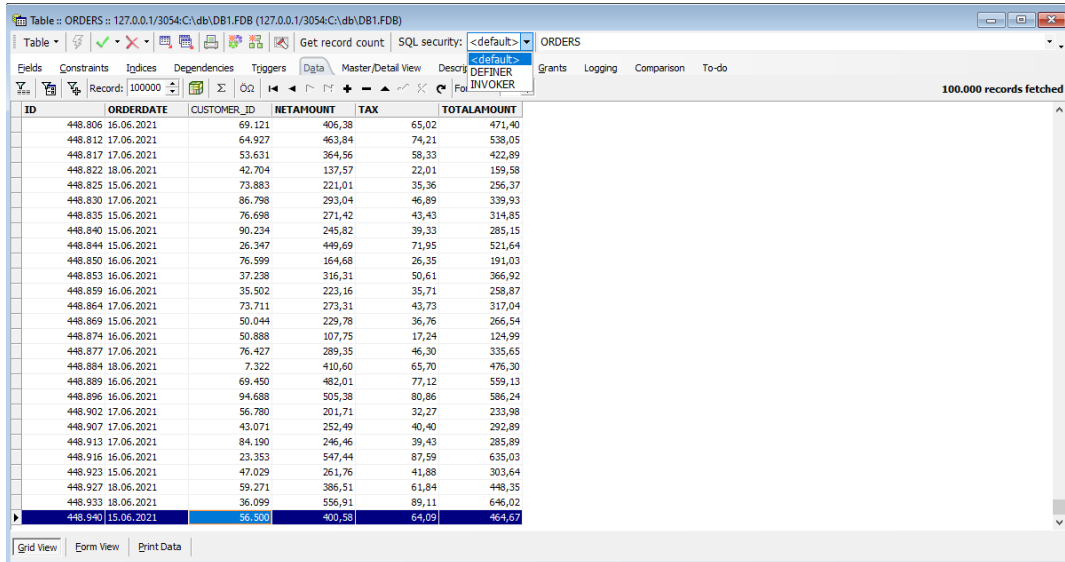
It is **important** to mention that most (if not all) new features of Firebird 4 are also available in IBEBlock, so you might want to update your scripts and gain advantage of these new features.

A brief tour through IBExpert dialogs where Firebird 4 features are clearly visible

If you open the **Database Registration** dialog, you can already see the Firebird 4.0 support that has been added:

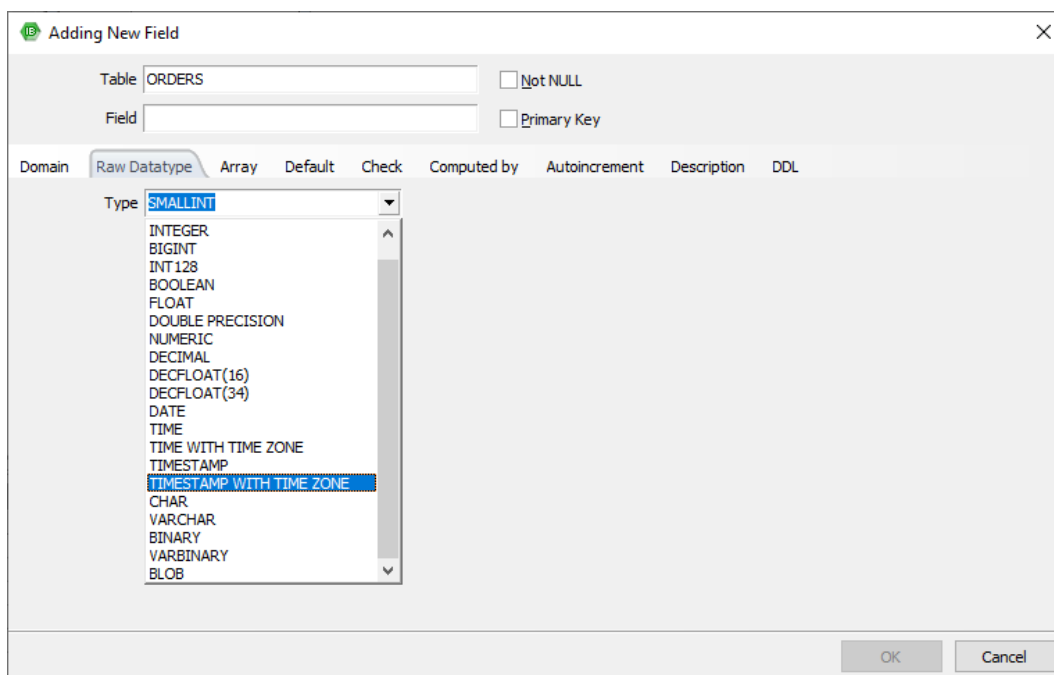


Also, on the **Data** page of your Firebird 4.0 database, you can see the new SQL Security feature:



ID	ORDERDATE	CUSTOMER_ID	NETAMOUNT	TAX	TOTALAMOUNT
448.806	16.06.2021	69.121	406,38	65,02	471,40
448.812	17.06.2021	64.927	463,84	74,21	538,05
448.817	17.06.2021	53.631	364,56	58,33	422,89
448.822	18.06.2021	42.704	137,57	22,01	159,58
448.825	15.06.2021	73.883	221,01	35,36	256,37
448.830	17.06.2021	86.798	293,04	46,89	339,93
448.835	15.06.2021	76.698	271,42	43,43	314,85
448.840	15.06.2021	90.234	245,82	39,33	285,15
448.844	15.06.2021	26.347	449,69	71,95	521,64
448.850	16.06.2021	76.599	164,68	26,35	191,03
448.853	16.06.2021	37.238	316,31	50,61	366,92
448.859	16.06.2021	35.502	223,16	35,71	258,87
448.864	17.06.2021	73.711	273,31	43,73	317,04
448.869	15.06.2021	50.044	229,78	36,76	266,54
448.874	16.06.2021	50.888	107,75	17,24	124,99
448.877	17.06.2021	76.427	289,35	46,30	335,65
448.884	18.06.2021	7.322	410,60	65,70	476,30
448.889	16.06.2021	69.450	482,01	77,12	559,13
448.896	16.06.2021	94.688	505,38	80,86	586,24
448.902	17.06.2021	56.780	201,71	32,27	233,98
448.907	17.06.2021	43.071	252,49	40,40	292,89
448.913	17.06.2021	84.190	246,46	39,43	285,89
448.916	16.06.2021	23.353	547,44	87,59	635,03
448.923	15.06.2021	47.029	261,76	41,88	303,64
448.927	18.06.2021	59.271	386,51	61,84	448,35
448.933	18.06.2021	36.099	556,91	89,11	646,02
448.940	15.06.2021	36.500	400,20	64,09	464,29

In the **Adding New Field** dialog, the new data types are listed:



Adding New Field

Table: Not NULL

Field: Primary Key

Domain: Raw Datatype Array Default Check Computed by Autoincrement Description DDL

Type:

- INTEGER
- BIGINT
- INT 128
- BOOLEAN
- FLOAT
- DOUBLE PRECISION
- NUMERIC
- DECIMAL
- DECFLOAT(16)
- DECFLOAT(34)
- DATE
- TIME
- TIME WITH TIME ZONE
- TIMESTAMP
- TIMESTAMP WITH TIME ZONE**
- CHAR
- VARCHAR
- BINARY
- VARBINARY
- BLOB

OK Cancel

As mentioned above, since version 4.0, Firebird supports time zones.



This support is however incompatible with previous versions' `'CURRENT_TIME'` and `'CURRENT_TIMESTAMP'` expressions.

In version 4, `'CURRENT_TIME'` and `'CURRENT_TIMESTAMP'` will respectively return expressions with the data types `'TIME WITH TIME ZONE'` and `'TIMESTAMP WITH TIME ZONE'`.

To make your queries and database code compatible with future versions (including Firebird 4.0), since Firebird 3.0.4 you can instead use `'LOCALTIME'` and `'LOCALTIMESTAMP'`. In version 3, these `'LOCAL*'` expressions will work identically to their correspondents `'CURRENT_*'` expressions.

In version 4, `'LOCAL*'` expressions will continue to work identically as in version 3, but the `'CURRENT_*'` expressions will change.

You should not start using `'LOCALTIME'` and `'LOCALTIMESTAMP'` if your database may have to be downgraded to version 3.0.3 or an older version, as these old versions will not recognize the new expressions.

Time zone support consists of `TIME WITH TIME ZONE` and `TIMESTAMP WITH TIME ZONE` data types, expressions and statements to work with time zones and conversion between data types without/with time zones.

The first important thing to understand is that the `TIME WITHOUT TIME ZONE`, `TIMESTAMP WITHOUT TIME ZONE` and `DATE` data types are defined to use the session time zone when converting from or to a `TIME WITH TIME ZONE` or `TIMESTAMP WITH TIME ZONE`. `TIME` and `TIMESTAMP` are synonymous to their respective `WITHOUT TIME ZONE` data types.

The session time zone, as the name implies, can be a different one for each database attachment. It can be set with the `isc_dpb_session_time_zone` DPB, and if not, it starts by default defined by the `firebird.conf` parameter `DefaultTimeZone` or the same time zone used by the Firebird OS process if the parameter is not defined. A change in the `DefaultTimeZone` configuration or the OS time zone does not change the default of a running Firebird process.

It can be changed with the `SET TIME ZONE` statement to a given time zone or reset to its original value with `SET TIME ZONE LOCAL`.

A time zone may be a string with a time zone region (for example, `America/Sao_Paulo`) or an hours:minutes displacement (for example, `-03:00`) from GMT.

A time/timestamp with time zone is considered equal to another time/timestamp with time zone if their conversion to UTC are equal, for example, time

```
'10:00 -02' = time '09:00 -03',
```




since both are the same as time '12:00 GMT'. This is also valid in the context of UNIQUE constraints and for sorting purposes.

BINARY, VARBINARY, BINARY VARYING data types are also supported, for example

```
1. DECLARE VARIABLE VAR1 VARBINARY(10);
2. CREATE TABLE TABLE1 (FIELD1 BINARY(16),
                           FIELD2 VARBINARY(100),
                           FIELD3 BINARY VARYING(1000));
```

Important notes here:

- If the length is omitted for type BINARY, it is considered to be 1.
- They can be distinguished from text types by the value 1 in RDB\$FIELD_SUB_TYPE.
- The character set is set to OCTETS for backward compatibility.
- In the API they are similar to the corresponding text types, getSubType() returns 0.

DECFLOAT is a DB2-compliant numeric type. DECFLOAT stores decimal values precisely (unlike FLOAT or DOUBLE PRECISION that provide binary approximation), therefore being an ideal choice for business applications. Firebird, according to the IEEE standard, has both 16- and 34-digit decimal float encodings. All intermediate calculations are performed with 34-digit values.

IBExpert supports Firebird 4 NUMERIC/DECIMAL data types with a maximum accuracy of 34 digits.

The maximum precision of NUMERIC and DECIMAL data types has been increased to 34 digits.

Numerics with a precision of less than 19 digits using SMALLINT, INTEGER, BIGINT or DOUBLE PRECISION are base data types depending upon the number of digits and SQL dialect. When precision is between 19 and 34 digits DECFLOAT(34) is used. The actual precision is always increased to 34 digits. For complex calculations such digits are casted (internally) to DECFLOAT(34) and the result of various mathematical (log, exp, etc.) and aggregate functions using a high precision numeric argument is DECFLOAT(34).

Another notable addition to Firebird 4.0 is the built-in logical replication, but that will be a topic of a new whitepaper.

For more information and usage scenarios, please consult the Firebird 4 Release Notes and Language Reference.