Data Definition Language (DDL)

Version 2.5 brings a few significant additions and enhancements to DDL.

Quick links

- CREATE/ALTER/DROP USER
- Syntaxes for altering views
- SSP extension for CREATE VIEW
- ALTER mechanism for computed columns
- GRANTED BY extension for GRANT and REVOKE
- ALTER ROLE
- REVOKE ALL
- Default COLLATION attribute for a database
- ALTER CHARACTER SET
- The DIFFERENCES FILE argument for CREATE DATABASE

Although this release emphasises architectural changes in the movement towards Firebird 3, a number of improvements and extensions have been implemented, in many cases as a response to feature requests in the Tracker.

General alert:

In v.2.5 and beyond, it is possible to alter the data type of a column, even if the column is referenced in a stored procedure or trigger, without an exception being thrown. Because compiled PSQL is stored statically as a binary representation (“BLR”) in a BLOB, the original BLR survives even a backup and restore. Being static, the BLR is not updated by the data type change, either.

This means that BLR that has references to the affected columns from the tables or downstream views, together with any variables declared using the TYPE OF syntax with reference to them is more than likely to be broken without warning. At best, the BLR will be flagged as “needing attention” but tests show that the flag is not set under all conditions.

In short, the engine now no longer stops you from changing the type of a field that has any dependencies in compiled PSQL. It will be a matter for your own change control to identify the affected procedures and triggers and recompile them to accommodate the changes.

Visibility of procedure definition changes on Classic

Dmitry Yemanov

Tracker reference CORE-2052.

One such change addressed the problem of the visibility of altered stored procedures to other connections to the Classic server. Now, such changes are made visible to the entire server as soon as the modifying transaction has completed its commit.

CREATE/ALTER/DROP USER

Alex Peshkov

Tracker reference CORE-696.

In v.2.5, Firebird finally has syntax to enable user accounts on the server to be managed by submitting SQL statements when logged in to a regular database.

The CREATE USER and ALTER USER statements also include the parameters GRANT ADMIN ROLE and REVOKE ADMIN ROLE to enable a user with SYSDBA privileges user to grant the RDB$ADMIN role in the security database to an ordinary user. For the usage description and a full overview of the RDB$ADMIN role, refer to the topic New RDB$ADMIN system role in the Administrative features chapter.

Syntax patterns

The SYSDBA, or a user with SYSDBA privileges in both the current database and the security database, can add a new user:

```
CREATE USER <username> {PASSWORD 'password'}
  [FIRSTNAME 'firstname']
  [MIDDLENAME 'middlename']
  [LASTNAME 'lastname']
  [GRANT ADMIN ROLE];
```

Note: The PASSWORD clause is required when creating a new user. It should be the initial password for that new user. The user can change it later, using ALTER USER.

The SYSDBA, or a user with SYSDBA privileges in both the current database and the security database, can change one or more of the password and proper name attributes of an existing user. Non-privileged users can use this statement to alter only their own attributes.

```
ALTER USER <username>
  [PASSWORD 'password']
  [FIRSTNAME 'firstname']
  [MIDDLENAME 'middlename']
  [LASTNAME 'lastname']
  [{GRANT | REVOKE} ADMIN ROLE];
```

Note: At least one of PASSWORD, FIRSTNAME, MIDDLENAME or LASTNAME must be present.

ALTER USER does not allow the <username> to be changed. If a different <username> is required, the old one should be deleted (dropped) and a new one created.

The SYSDBA, or a user with SYSDBA privileges in both the current database and the security database, can delete a user:

```
DROP USER <username>;
```

Restrictions

CREATE USER and DROP USER statements and GRANT | REVOKE ADMIN ROLE are available only for the SYSDBA, or a user that has acquired the RDB$ADMIN role in both the current database and the security database.

An ordinary user can ALTER his own password and elements of his proper name. An attempt to modify another user will fail.

Examples

SYSDBA or a user with equivalent privileges in both the current database and the security database, can do:

```
CREATE USER alex PASSWORD 'test';

ALTER USER alex FIRSTNAME 'Alex' LASTNAME 'Peshkov';

ALTER USER alex PASSWORD 'IdQfA';
```

See also:

Server security security2.fdb

back to top of page

Syntaxes for altering views

Adriano dos Santos Fernandes

Previously, in order to alter a view definition, it was necessary to save the view definition off-line somewhere and drop the view, before recreating it with its changes. This made things very cumbersome, especially if there were dependencies. Version 2.5 introduces syntaxes for ALTER VIEW and CREATE OR ALTER VIEW.

Tracker references are CORE-770 and CORE-1640.

ALTER VIEW

ALTER VIEW enables a view definition to be altered without the need to recreate (drop and create) the old version of the view and all of its dependencies.

CREATE OR ALTER VIEW

With CREATE OR ALTER VIEW, the view definition will be altered (as with ALTER VIEW) if it exists, or created if it does not exist.

Syntax pattern

```
create [ or alter ] | alter } view <view_name>
```

```
  [ ( <field list> ) ]
as <select statement>
```

Example

```
create table users (
    id integer,
    name varchar(20),
    passwd varchar(20)
);

create view v_users as
    select name from users;

alter view v_users (id, name) as
    select id, name from users;
```

back to top of page

Extensions for CREATE VIEW

The following extensions have been added for CREATE VIEW.

Specify stored procedure in FROM clause

Adriano dos Santos Fernandes

Tracker reference CORE-886.

A selectable stored procedure can now be specified in the FROM clause of a view definition.

Example

```
create view a_view as
select * from a_procedure(current_date);
```

Create UNION view without column list

Dmitry Yemanov

Tracker reference CORE-1402.

The column list can now be omitted from CREATE VIEW when the set is defined by a UNION.

Example

```
recreate view V1 as
```

```
  select d.rdb$relation_id from rdb$database d
  union all
  select d.rdb$relation_id from rdb$database d

recreate view V2 as
  select d.rdb$relation_id as q from rdb$database d
  union all
  select d.rdb$relation_id as w from rdb$database d
```

Inferred column names

Adriano dos Santos Fernandes

Tracker reference CORE-2424.

CREATE VIEW now has the capability to infer column names for views involving a GROUP BY clause or a derived table.

Example

```
create view V as
  select d.rdb$relation_id from rdb$database d
  group by d.rdb$relation_id

create view V as
  select a from (select 1 a from rdb$database);
```

back to top of page

ALTER mechanism for computed columns

Adriano dos Santos Fernandes

Tracker reference CORE-1454.

A column defined as COMPUTED BY <expression> can now be altered using the ALTER TABLE...ALTER COLUMN syntax. This feature can be used only to change the <expression> element of the column definition to a different expression. It cannot convert a computed column to non-computed or vice versa.

Syntax pattern

```
alter table <table-name>
  alter <computed-column-name>
  [type <data-type>]
  COMPUTED BY (<expression>);
```

Examples

```
create table test (
  n integer,
  dn computed by (n * 2)
);
commit;
alter table test
  alter dn computed by (n + n);
```

back to top of page

Extensions for SQL permissions

Alex Peshkov

The following extensions have been implemented in the area of SQL permissions (privileges).

GRANTED BY clause

A GRANTED BY or GRANTED AS clause can now be optionally included in GRANT and REVOKE statements, enabling the grantor to be a user other than the CURRENT_USER (the default).

Syntax pattern

```
grant <right> to <object>
  [ { granted by | as } [ user ] <username> ]
--
revoke <right> from <object>
  [ { granted by | as } [ user ] <username> ]
```

Tip: GRANTED BY and GRANTED AS are equivalent. GRANTED BY is the form recommended by the SQL standard. We support GRANTED AS for compatibility with some other servers (Informix, for example).

Example

Logged in as SYSDBA:

```
create role r1; -- SYSDBA owns the role
  /* next, SYSDBA grants the role to user1
     with the power to grant it to others */
grant r1 to user1 with admin option;
  /* SYSDBA uses GRANTED BY to exercise
     user1's ADMIN OPTION */
grant r1 to public granted by user1;
```

In isql, we look at the effects:

```
SQL>show grant;
```

```
/* Grant permissions for this database */
GRANT R1 TO PUBLIC GRANTED BY USER1
GRANT R1 TO USER1 WITH ADMIN OPTION
SQL>
```

back to top of page

ALTER ROLE

Tracker reference CORE-1660.

The new ALTER ROLE statement has a specialised function to control the assignment of SYSDBA permissions to Windows administrators during trusted authentication. It has no other purpose currently.

Note: For the usage description of ALTER ROLE and a full overview of the RDB$ADMIN role, refer to the topic New RDB$ADMIN system role in the Administrative features chapter.

REVOKE ALL

Tracker reference CORE-2113.

When a user is removed from the security database or another authentication source, such as the operating system ACL, any associated cleanup of SQL privileges in databases has to be performed manually. This extension adds the capability to revoke all privileges in one stroke from a particular user or role.

Syntax pattern

```
REVOKE ALL ON ALL FROM { <user list> | <role list> }
```

Example

Logged in as SYSDBA:

```
# gsec -del guest
# isql employee
fbs bin # ./isql employee
Database: employee
SQL> REVOKE ALL ON ALL FROM USER guest;
SQL>
```

back to top of page

Default COLLATION attribute for a database

Adriano dos Santos Fernandes

Tracker references CORE-1737 and CORE-1803.

An ODS 11.2 or higher database can now have a default COLLATION attribute associated with the default character set, enabling all text column, domain and variable definitions to be created with the same collation order unless a COLLATE clause for a different collation is specified.

The COLLATION clause is optional. If it is omitted, the default collation order for the character set is used.

Tip: Note also that the default collation order for a character set used in a database can now also be changed, thanks to the introduction of syntax for ALTER CHARACTER SET.

Syntax pattern

```
create database <file name>
   [ page_size <page size> ]
   [ length = <length> ]
   [ user <user name> ]
   [ password <user password> ]
   [ set names <charset name> ]
   [ default character set <charset name>
   [ [ collation <collation name> ] ]
   [ difference file <file name> ]
```

Note: The parameter DIFFERENCE FILE is not a new one for CREATE DATABASECREATE DATABASE. It was quietly introduced in association with the nBackup utility in v.2.0 and has been lurking undocumented for three years. For more information, see Evolution of CREATE DATABASE at the end of this chapter.

Example

```
create database 'test.fdb'
    default character set win1252
    collation win_ptbr;
```

back to top of page

ALTER CHARACTER SET command

Adriano dos Santos Fernandes

Tracker reference CORE-1803.

New syntax introduced in this version, enabling the default collation for a character set to be set for a database.

The default collation is used when table columns are created with a given character set (explicitly, through a CHARACTER SET clause in the column or domain definition, or implicitly, through the default character set attribute of the database) and no COLLATION clause is specified.

Note: String constants also use the default collation of the connection character set.

The character set and collation of existing columns are not affected by ALTER CHARACTER SET changes.

Syntax pattern

```
ALTER CHARACTER SET <charset_name>
    SET DEFAULT COLLATION <collation_name>
```

Example

```
create database 'people.fdb'
  default character set win1252;

alter character set win1252
  set default collation win_ptbr;

create table person (
  id integer,
  name varchar(50) /* will use the database default
                      character set and the win1252
                      default collation */
);

insert into person
  values (1, 'adriano');
insert into person
  values (2, 'ADRIANO');

/* will retrieve both records
   because win_ptbr is case insensitive */
select * from person where name like 'A%';
```

Tip: Another improvement allows the current value of RDB$DEFAULT_COLLATE_NAME in the system table RDB$CHARACTER_SETS to survive the backup/restore cycle.

back to top of page

Evolution of CREATE DATABASE

DDL support for the database header attributes introduced to register and change the nBackup states has been evolving since Firebird 2.0. Users of nBackup will be familiar with the ALTER DATABASE statements to begin and end the storage of the “delta” data, in a file apart from the main database, during a full nBackup.

Naming the delta File for nBackup

ALTER DATABASE also has another argument that allows you to set the name that will be used for the file that stores the delta data. To quote from Paul Vinkenoog's excellent manual for nBackup:

By default, the delta file lives in the same directory as the database itself. It also has the same name as the database file, but with .delta appended. There is usually no reason to change this, but it can be done if need be — though not via nBackup itself.

Make a connection to the database with any client that allows you to enter your own SQL statements and give the command:

```
alter database
  add difference file 'path-and-filename'
```

The custom delta file specification is persistent in the database; it is stored in the system table RDB$FILES.

To revert to the default behaviour, issue the following statement:

```
alter database
  drop difference file
```

Those who are still curious may study the details in the v.2.0 Release Notes or in the nBackup manual.

back to top of page

The DIFFERENCES FILE argument for CREATE DATABASE

C. Valderrama

In Firebird 2.0, syntax for prescribing a custom name for the delta file appeared as an extra, optional argument for CREATE DATABASE. You can observe its placement in the syntax pattern given above for the topic Default COLLATION attribute for a database. As with ALTER DATABASE, the keyword for the argument is DIFFERENCE FILE and the argument is a valid file specification. It allows you to specify a custom name for the delta file that will be created whenever ALTER DATABASE BEGIN BACKUP is called, or when the equivalent nBackup shell command is invoked.

Examples of Usage

```
  ]..\bin> isql -user sysdba -pass masterke

Use CONNECT or CREATE DATABASE to specify a database
SQL> create database 'ticks' difference file 'jaguar';
SQL> shell dir jaguar;
Volume in drive F is Firebird
Volume Serial Number is BCD9-4211

Directory of ..\bin
```

```
File Not Found
```

This is correct, we only defined the file name. Now it will be used:

```
SQL> alter database begin backup;
SQL> shell dir jaguar;
Volume in drive F is Firebird
Volume Serial Number is BCD9-4211

Directory of ..\bin

10-11-2009      00:59     8.192 jaguar
                  1 File(s) 8.192 bytes
                  0 Dir(s) 16.617.979.904 bytes free

SQL> alter database end backup;
SQL> shell dir jaguar;
Volume in drive F is Firebird
Volume Serial Number is BCD9-4211

Directory of ..\bin

SQL> drop database;
SQL> ^Z
```

Since the argument is a file name, it goes inside single quotes. Double quotes are not valid: the statement will fail and return a confusing error message.

```
]..\bin> isql -user sysdba -pass masterke

Use CONNECT or CREATE DATABASE to specify a database
SQL> create database 'ticks' difference file 'jaguar';
SQL> alter database add difference file 'leopard';

Statement failed, SQLCODE = -607
unsuccessful metadata update
-Difference file is already defined
```

The message is correct. Even though the delta was deleted by the ALTER DATABASE END BACKUP call, the name of the difference file is stored persistently and only one may exist.

```
SQL> alter database drop difference file;
SQL> alter database begin backup;
```

This does not break anything, because the engine will rescue the situation and create the delta using the default mechanism.

```
SQL> alter database add difference file 'leopard';
SQL> alter database begin backup;
SQL> alter database drop difference file;
```

```
Statement failed, SQLCODE = -607
unsuccessful metadata update
-Cannot change difference file name while database is in backup mode
```

This is correct validation.

```
SQL> alter database end backup;
SQL> drop database;
SQL> ^Z
```