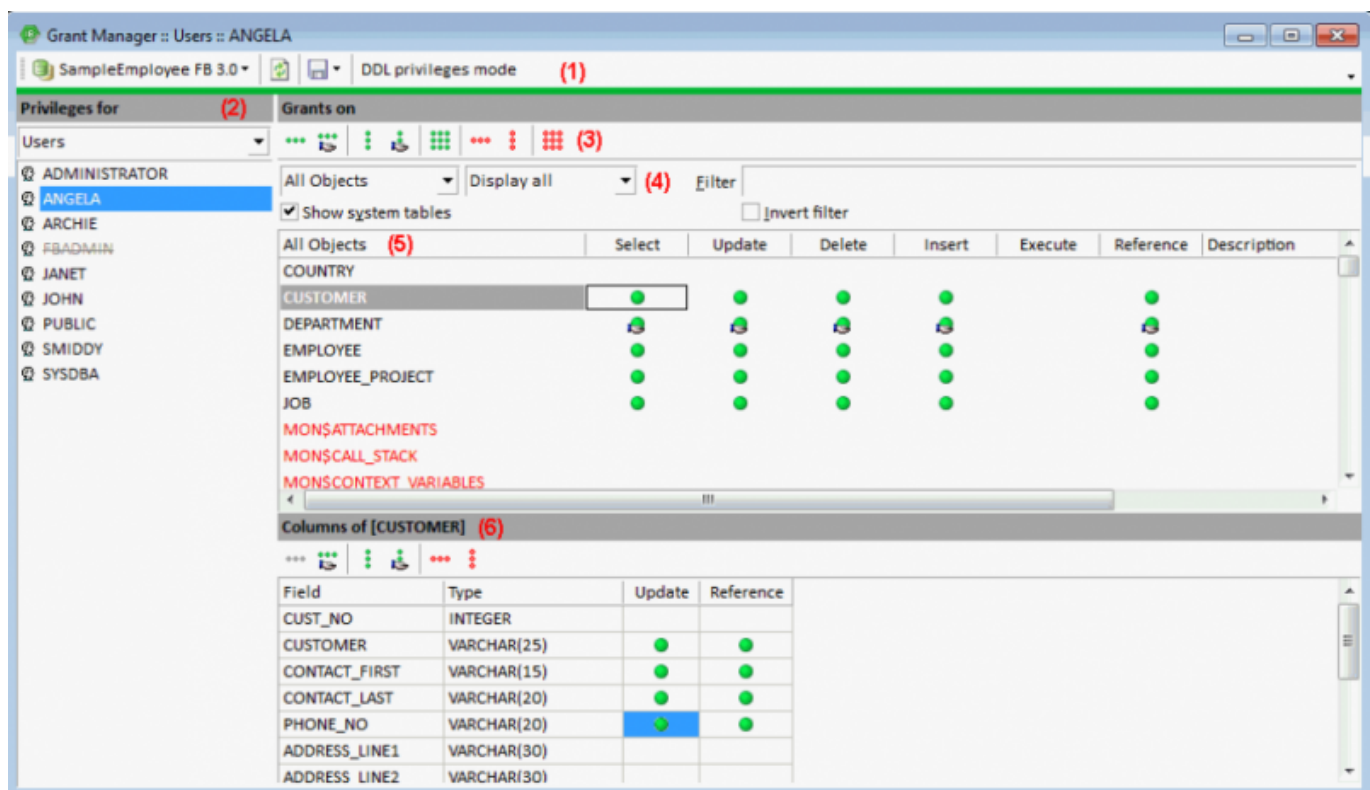


Grant Manager

The Grant Manager is used to administrate database security by controlling user permissions for a specific [database](#). It allows you to specify the access rights for users, [roles](#) and [database objects](#). It is possible to grant rights for database objects on the [Grants page](#) in the object editors. (This feature is unfortunately not included in the [free IBExpert Personal Edition](#).)

To start the Grant Manager select the IBExpert menu item, [Tools / Grant Manager](#), use the respective [icon](#) in the [Tools toolbar](#), or double-click on a role in the [DB Explorer](#). Alternatively use the [DB Explorer](#)'s right mouse-click menu item *Edit Role* or key combination [Ctrl + O].

The *Grant Manager* Editor appears:



(1) Toolbar: The toolbar displays the [alias](#) name for the current selected [connected database](#). Another database on this server can be selected from the drop-down list at the top of the window. To the right of the selected database, there are two icon options to enable *Refresh* and *Save privileges to script*. When the *DDL privileges mode* is enabled, this allows you to assign rights and privileges to metadata objects. Please refer [Firebird 3.0 DDL privileges](#) to for further information.

(2) Privileges for: The drop-down list (default = *Users*) allows a group for the processing of privileges to be selected. The options include:

- users
- [roles](#)
- [views](#)
- [triggers](#)
- [procedures](#)

Once a database object has been selected, a full list of such users/objects in this database is

displayed in the panel directly below.

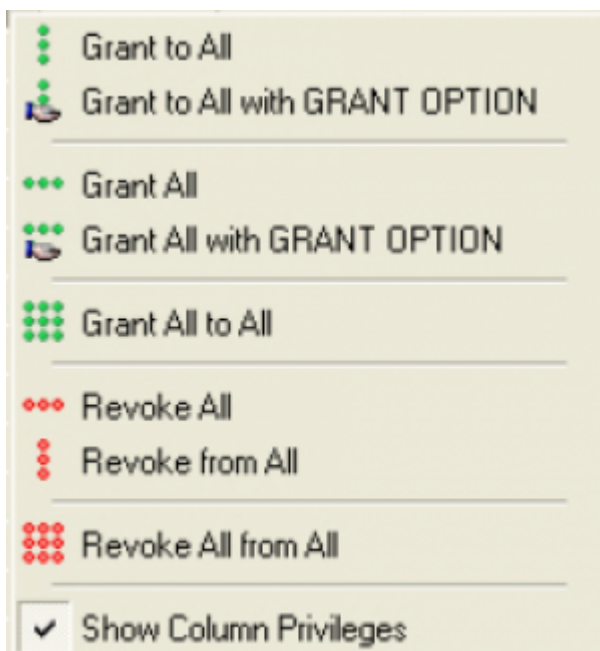
Firebird 2.5 introduced a new role, **RDB\$ADMIN**, for databases ODS version 11.2 and higher. This role allows regular users to be granted **SYSDBA-similar** rights at database level.

The Grant Manager reads the users list from **SEC\$USERS** table if one exists in the database.

(3) Grants toolbar: The **Grants toolbar** enables the user to quickly assign or revoke rights to one or more objects, or for one or more operations. These can also be found in the right-click pop-up menu (see below).

(4) Filters: It is possible, using the drop-down lists, to specify exactly which grants should be displayed, i.e. for all database objects (default), just the tables, just the views or just the procedures. Furthermore the user can determine whether all of the selected objects should be displayed, or only those with grants, or only those not granted. To the right of these drop-down lists is an empty filter field for user-defined filters. It is also possible to specify whether system tables should be included or the user-defined filter inverted, using the check boxes provided.

(5) The main window displays the object grants in a grid, displaying the granted operations Select, Update, Delete, Insert, Execute and Reference for the listed objects. A green circle indicates that access for this operation on this database object has been granted; a green circle held by a hand indicates that the **GRANT WITH GRANT AUTHORITY** option has been granted. **UPDATE/REFERENCE** privileges on certain columns only for tables/views is also indicated visually. A gray ball means that there is at least one column with a granted privilege. A gray ball in the hand means that there is at least one column with a privilege granted with the grant option. An empty field indicates logically that either no rights have been granted, or they have been revoked.



A further menu option here is *Show Column Privileges* (checkbox). This blends the lower window in and out **(6)**, which displays the individual **columns** for **tables** and **views**, allowing *Update* and *Reference* rights to be granted and revoked for individual **fields** in the selected object. *Reference* rights are important when working with primary and foreign keys. For example, **TABLE_A** creates a foreign key on **TABLE_B**. If a data set is inserted into **TABLE_A**, **TABLE_B** needs to be referenced, whether the data entry is permitted. Therefore, **TABLE_A** will require **REFERENCE** rights (also known

as read permission) on `TABLE_B`.

Rights can be simply granted and revoked by double-clicking (or using the space bar) on the grid fields (in both the upper (object) and lower (column) windows). Alternatively, to assign several rights (i.e. select, update, delete and insert) to a single object or to assign one operative right to all objects displayed, use either the Grant Manager toolbar or the right-click menu.

Please note that *Reference* rights only allow the user to read [data sets](#) if there is a [foreign key](#) relationship to other data. And the *Grant All to All* command may only be performed by the database owner or the `SYSDBA`.

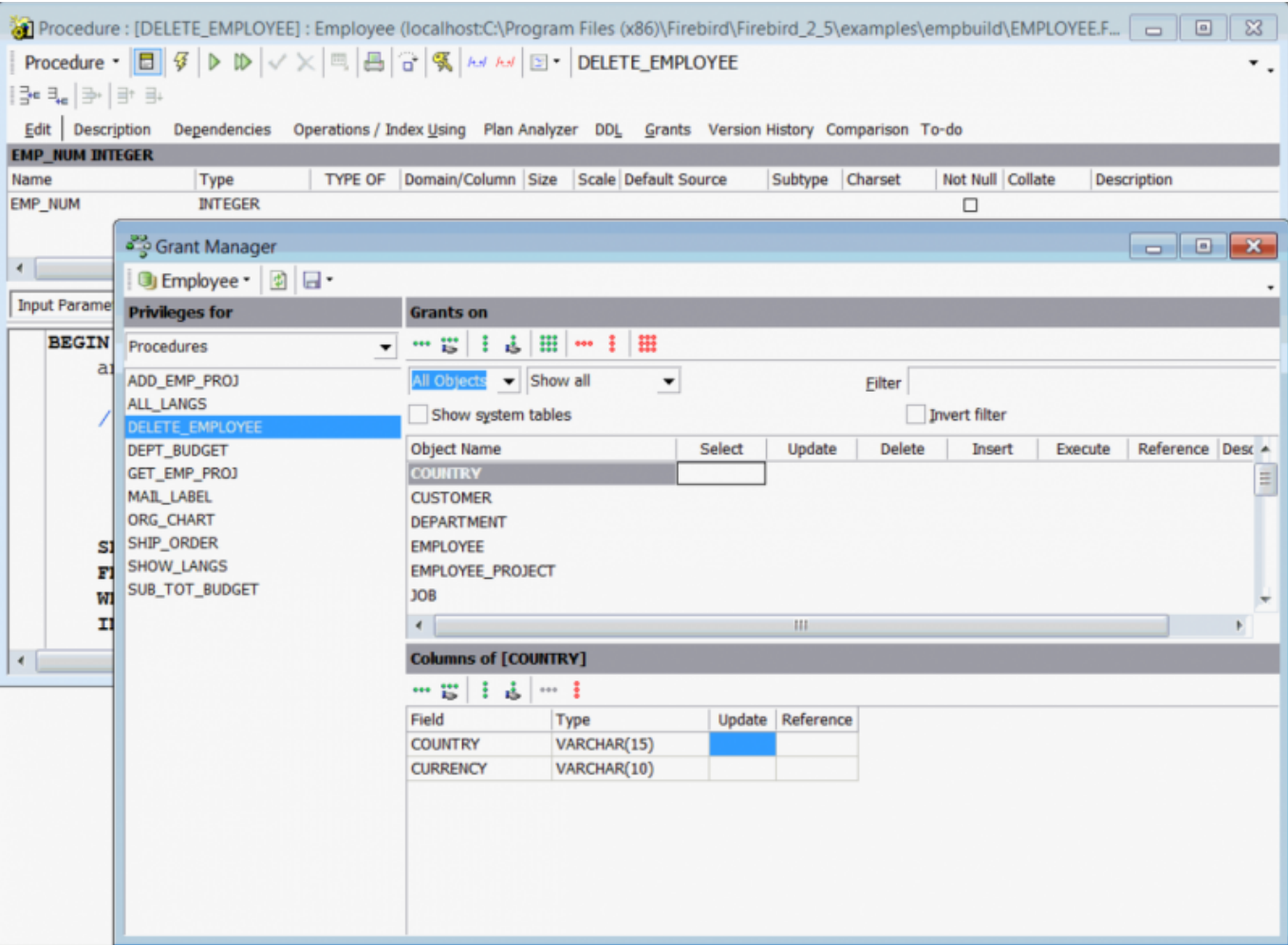
The majority of these operations can also be performed in the [Grants](#) pages, found in the individual database object editors. These were introduced to remind the developer not to forget the assignment of rights, when creating or altering a database object! They allow the developer to check existing permissions for the object concerned and, if necessary, subsequently assign rights for a new or existing object.

Rights are however in practice usually administered at the front end. There is, as a rule, only one system user, with which the program can log into the database. For those preferring direct SQL input, please refer to [GRANT](#) and [REVOKE](#).

[back to top of page](#)

Opening the Grant Manager for an active object

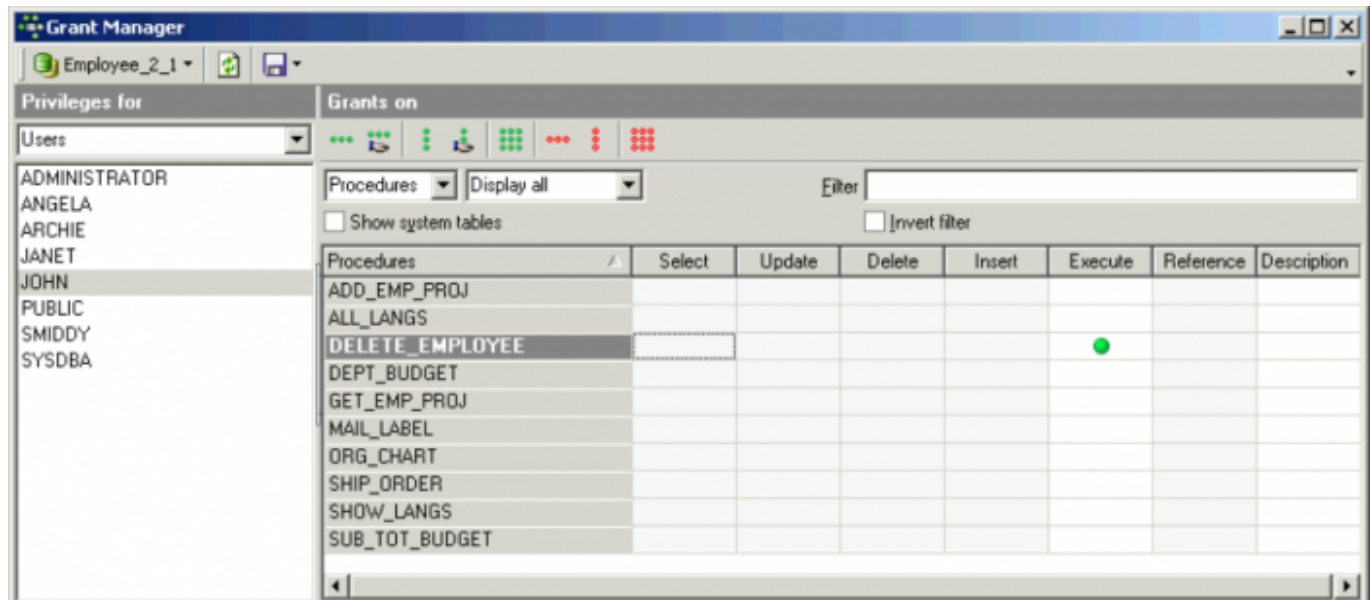
If there is an active [stored procedure](#), [view](#) or [trigger editor](#) opened on the screen, the corresponding object will be automatically selected when the Grant Manager is started:



[back to top of page](#)

Granting access to stored procedures

To grant a user the right to execute [stored procedures](#), use the Grant Manager *Execute* column:



or the SQL EXECUTE statement. For example, to grant Janet and John the right to execute the stored procedure SP_Delete_Employee, use the following:

```
GRANT EXECUTE
ON PROCEDURE SP_Delete_Employee
TO Janet, John;
```

Firebird/InterBase® considers stored procedures as virtual users of the [database](#). If a stored procedure modifies a [table](#), the procedure needs the relevant privileges on that table. So the user only needs *Execute* privileges on the procedure and not any separate rights for the table. In this situation, the stored procedure performs the changes on behalf of the user.

If a stored procedure needs the ability to execute another stored procedure, simply select *Procedures* from the *Privileges For* list and *Procedures* from the *Grants On* list, to grant the *Execute* privilege on the desired procedure. Using SQL the [GRANT statement](#) is necessary, naming the procedure instead of one or more users (<user_list>).

[back to top of page](#)

Using the GRANT AUTHORITY option

A user that has been granted certain privileges, may also be assigned the authority to grant those privileges in turn to other users. This is known as assigning grant authority. Firebird/InterBase® allows by [default](#) only the creator of a [table](#) and the SYSDBA to grant additional privileges onto other users.

Grant authority can be assigned in the IBExpert or the [Grants](#) pages in the relevant object editors, using the *Grant All with GRANT OPTION* or the *Grant to All with GRANT OPTION* icons or right-click menu items:

Table : [PROJ_DEPT_BUDGET] : Employee_2_1 (localhost/3021\FB3021\EMPLOYEE.FDB)

Table

Get record count

PROJ_DEPT_BUDGET

Fields

Constraints

Indices

Dependencies

Triggers

Data

Master/Detail View

Description

DDL

Grants

Logging

Comparison

To-do

Users

Show all

Filter

☐ Invert filter

Users	Select	Update	Delete	Insert	Execute	Reference	Description
ADMINISTRATOR							
ANGELA							
ARCHIE							
JANET							
JOHN							
PUBLIC							
SMIDDY							
SYSDBA							

Columns of [PROJ_DEPT_BUDGET]

Field	Type	Update	Reference
FISCAL_YEAR	INTEGER		
PROJ_ID	CHAR(5)		
DEPT_NO	CHAR(3)		
QUART_HEAD_CNT	INTEGER(1:4)		
PROJECTED_BUDGET	DECIMAL(12,2)		

It is also simple to see which grant authorities have already been assigned to which users and roles.

In SQL the `WITH GRANT OPTION` clause may be used in conjunction with a grant of privileges, to assign users the authority to grant their privileges in turn to other users (refer to [GRANT statement](#) for the full syntax and examples).

Firebird 3.0 USAGE privilege

The `USAGE` privilege for [generators](#) and [exceptions](#) was introduced in Firebird 3.0: Excerpt from [The Firebird 3.0 Release Notes \(27 January 2014 - Document v.0300-08 - for Firebird 3.0 Alpha 2\)](#):

Privileges to protect other metadata objects

New SQL-2008 compliant `USAGE` permission is introduced to protect metadata objects other than tables, views, procedures and functions.

Syntax pattern

```
GRANT USAGE ON <object type> <name> TO <grantee list>
[<grant option> <granted by clause>]
--
REVOKE USAGE ON <object type> <name> FROM <grantee list>
[<granted by clause>]
--
<object type> ::= {DOMAIN | EXCEPTION | GENERATOR | SEQUENCE | CHARACTER
```

```
SET | COLLATION}
```

Notes

The initial `USAGE` permission is granted to the object owner (user who created the object). In Firebird 3.0 Alpha 1, only `USAGE` permissions for exceptions ([CORE-2884](#)) and generators/sequences (`gen_id`, next value for: [CORE-2553](#)) are enforced. Permissions for other object types will be validated in subsequent releases.

[back to top of page](#)

Firebird 3.0 DDL privileges

The IBExpert Grant Manager offers the Firebird 3.0 *DDL privileges* mode. DDL Privileges are a new security feature in Firebird 3.0.

The following is an excerpt from the [The Firebird 3.0 Release Notes \(8 December 2014 - Document v.0300-18 - for Firebird 3.0 Beta 1\)](#) chapter, Security:

User Privileges for Metadata Changes

Dmitry Yemanov
with Roman Simakov

In Firebird 3, the system tables are read-only. This SQL syntax provides the means to assign metadata write privileges to specified users or roles for specified objects. See Tracker item [CORE-735](#).

Note: Some people have been applying the nickname *DDL privileges* to this feature. Don't confuse it with [DDL triggers](#)! A more useful nickname would be *Metadata privileges*.

Syntax Patterns

Granting metadata privileges:

```
GRANT CREATE <object-type>  
  TO [USER | ROLE] <user-name> | <role-name> [WITH GRANT OPTION];
```

GRANT ALTER ANY <object-type>

```
  TO [USER | ROLE] <user-name> | <role-name> [WITH GRANT OPTION];  
GRANT DROP ANY <object-type>  
  TO [USER | ROLE] <user-name> | <role-name> [WITH GRANT OPTION];
```

Revoking metadata privileges:

```
REVOKE [GRANT OPTION FOR] CREATE <object-type>  
  FROM [USER | ROLE] <user-name> | <role-name>;  
REVOKE [GRANT OPTION FOR] ALTER ANY <object-type>
```

```
FROM [USER | ROLE] <user-name> | <role-name>;
REVOKE [GRANT OPTION FOR] DROP ANY <object-type>
FROM [USER | ROLE] <user-name> | <role-name>;
```

Special form for database access:

```
GRANT CREATE DATABASE TO [USER | ROLE] <user-name> | <role-name>;
GRANT ALTER DATABASE
  TO [USER | ROLE] <user-name> | <role-name> [WITH GRANT OPTION];
GRANT DROP DATABASE
  TO [USER | ROLE] <user-name> | <role-name> [WITH GRANT OPTION];

REVOKE CREATE DATABASE FROM [USER | ROLE] <user-name> | <role-name>;
REVOKE [GRANT OPTION FOR] ALTER DATABASE
  FROM [USER | ROLE] <user-name> | <role-name>;
REVOKE [GRANT OPTION FOR] DROP DATABASE
  FROM [USER | ROLE] <user-name> | <role-name>;
```

Notes on Usage

- <object-type> can be any of the following:

CHARACTER SET	COLLATION	DOMAIN	EXCEPTION
FILTER	FUNCTION	GENERATOR	PACKAGE
PROCEDURE	ROLE	SEQUENCE	TABLE
VIEW			

Note: The metadata for triggers and indices are accessed through the privileges for the table that owns them.

- If the **ANY** option is used, the user will be able to perform any operation on any object
- If the **ANY** option is absent, the user will be able to perform operations on the object only if he owns it
- If the **ANY** option was acquired via a **GRANT** operation then, to revoke it, the **REVOKE** operation must accord with that **GRANT** operation.

Example

```
GRANT CREATE TABLE TO Joe;
GRANT ALTER ANY TABLE TO Joe;
REVOKE CREATE TABLE FROM Joe;
```

Source: *The Firebird 3.0 Release Notes* by Helen Borrie (Collator/Editor): 29 November 2014 - Document v.0300-16 - for Firebird 3.0 Beta 1.

From:
<https://www.ibexpert.com/docu/> - **IBExpert**

Permanent link:
<https://www.ibexpert.com/docu/doku.php?id=02-ibexpert:02-08-ibexpert-tools-menu:grant-manager>

Last update: **2023/10/07 16:17**

