DB EXPLORER

DATABASE TOOL

Complex organizations have lots of databases, with different roles and environments, different providers, and this may create difficulties when managing all these databases.

DBExplorer is a WEB tool offering a single application to perform database tasks on any number of databases.

Application key features:

- strong security
- multi environment
- database changes logging with script exporting and undo of changes.
- unintended data update prevention
- easy export (sql and excel) and import of data (huge sql scripts)
- database compare of structure and data among environments
- reporting, tabular data and charts.

Currently it can handle databases like MSSQL, DB2 (both windows and mainframe), Oracle, MySql.

You can have any number of databases, grouped together by role (business type) and environment. You can easily choose to handle a certain environment and its databases or even more environments at once.

Environments

- Grouping databases into environments may give you a better way to work with environment data.
- You can define application properties and configurations specifics to a given environment (user acces, user rights, limits and all other application configuration that may be possible)
- When starting a work session, you just connect to the environment you need and all the databases of the environment will be available for you to inquiry, modify, export data etc.
- Different color schemes for every environment, for easier recognition of environment while working
- Easy compare of database structure and data between two environments. Comparing may be done in two ways:
    - every time you open a row in a table you can see corresponding row in the "other" environment, also you can easily copy the row from one environment to another simply by using "copy" and "paste" buttons.
    - compare process of all database tables (or a subset of tables) for both structure and data.

Business Types

Grouping databases under the same purpose/logic offers easy configuration for a database logic. You can define rules and configurations that will apply to all databases of the same environment.

Note: you can also have exceptions, you may define rules and configurations that will apply only to the specific environment(s), if needed.

Security

Having a web application helps to control who can access the databases without giving database user and password to the users. Revoking access to a user is very easy, simply disable/delete the user login in the application, no database user and password needs to be changed, nobody actually knows the database credentials but the web/database admin.

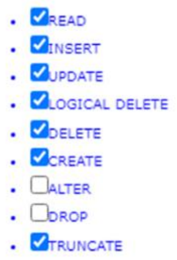
Access to application is also secured, you can link an IP (or a list of IP) to a specific application login and that user will not be allowed from any other IP. More, in case of repeated (five) bad user/password combinations application is locking both the IP and the login. There is no expiration of the lock, only application administrators can remove the lock, so nobody can brute force an user/password for more than 5 times. This policy, combined with solid password requirements and IP binding offers the login security you can trust. Also, the login used is not shown anywhere, the user name, displayed in the application reports is not the same with the login name used when login in the application.

Access rights are given to User Groups, so new users will easily inherit the same rights as any other user of the group he belongs to.

The rights on tables are very flexible, application can discriminate, for each table and user group the access to below actions:

- View Table Data
- Edit Table Rows
- Delete Table Rows
- Logically Delete Table Rows
- Insert new rows
- Specific rights to each DDL instructions (CREATE, ALTER, DROP, TRUNCATE)

More, you can also discriminate at column level, you can choose to hide a column to a user or to make it read only, no modification allowed on that column, even if the user may modify the table rows.



- ☑ READ
- ☑ INSERT
- ☑ UPDATE
- ☑ LOGICAL DELETE
- ☑ DELETE
- ☑ CREATE
- ☐ ALTER
- ☐ DROP
- ☑ TRUNCATE

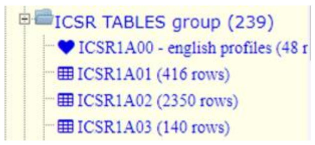
Also, you may differentiate between groups of users on other aspects, like how many rows a recordset will return (limit row) and what application utilities (query tool, database compare etc) they may use.

Working with Data

DBExplore is reading databases structures (and cache it) to offer fast connection time and all the information needed when working with a database.

Tables are loaded in a tree structure, you may group together tables by role, give tables logical descriptions (also it reads the description given in the database, if present), give distinctive icons to each table and other customizations like table extended explanation text, that will be shown to users when viewing or editing the table, one message for view and a different for edit, if needed).

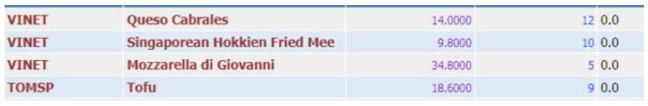
You can have any number of tree folders, as suited by your business model.



```
⊟ ICSR TABLES group (239)
   ├ ♥ ICSR1A00 - english profiles (48 r
   ├ ICSR1A01 (416 rows)
   ├ ICSR1A02 (2350 rows)
   └ ICSR1A03 (140 rows)
```

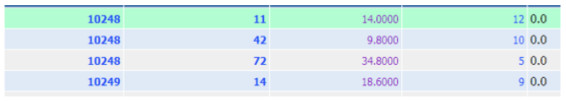
Searching for a table name or even by tables containing a field is also possible.

When displaying a table data the application is using the defined foreign keys to show a more human readable information based on the defined foreign keys. Users may readable information, taken from parent tables defined in foreign keys, instead of codes. Also, plain data view is available, if needed.



VINET	Queso Cabrales	14.0000	12	0.0
VINET	Singaporean Hokkien Fried Mee	9.8000	10	0.0
VINET	Mozzarella di Giovanni	34.8000	5	0.0
TOMSP	Tofu	18.6000	9	0.0

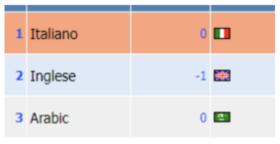
or



10248	11	14.0000	12	0.0
10248	42	9.8000	10	0.0
10248	72	34.8000	5	0.0
10249	14	18.6000	9	0.0

Different color schemes are used to show different data types, FK data or PK fields.

Also binary data is handled, you may see images, download or upload binary content to tables.

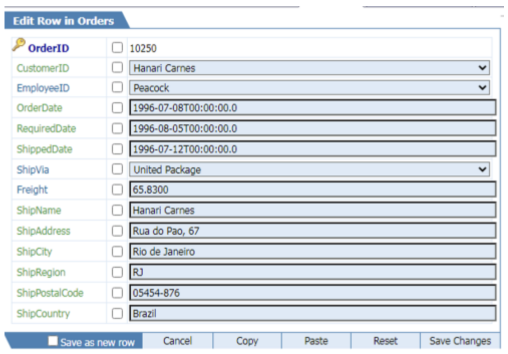


1	Italiano	0	🇮🇹
2	Inglese	-1	🇬🇧
3	Arabic	0	🇸🇦

Data is loaded very fast due to the way applications sends data to the page, with the **minimal overhead**, all the data is sent as an array and the presentation of the data is done in the client page, allowing fast data loading (tables with 10k rows being loaded in less than a second).

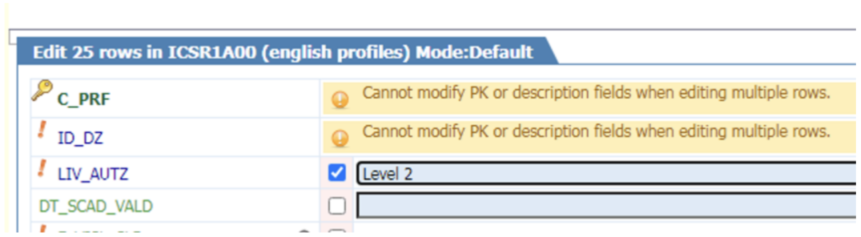
Once you got the data from server you can **sort** data by columns or **filter** it, all being done in the page (without involving the server again, which means less workload for your database)

Data edit offers also a human readable data, offering both row data and FK information when editing/inserting a table row. Features like copy and paste, duplicate row are also available for a better user experience.


Edit Row in Orders

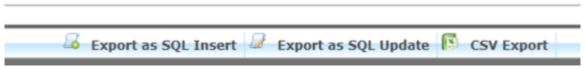
OrderID	10250
CustomerID	Hanari Carnes
EmployeeID	Peacock
OrderDate	1996-07-08T00:00:00.0
RequiredDate	1996-08-05T00:00:00.0
ShippedDate	1996-07-12T00:00:00.0
ShipVia	United Package
Freight	65.8300
ShipName	Hanari Carnes
ShipAddress	Rua do Pao, 67
ShipCity	Rio de Janeiro
ShipRegion	RJ
ShipPostalCode	05454-876
ShipCountry	Brazil

Multiple row editing is also possible (in case you want to change values for one (or more) columns in a table for many rows you don't need to repeat the change for each row, you can select the rows and enter in edit mode, you will change the desired columns values only for all the selected rows.


Edit 25 rows in ICSR1A00 (english profiles) Mode:Default

C_PRF		Cannot modify PK or description fields when editing multiple rows.
ID_DZ		Cannot modify PK or description fields when editing multiple rows.
LIV_AUTZ	✓	Level 2
DT_SCAD_VALD	☐	

If there are tables without primary keys the application will consider as PK (a 'fake' primary key) all the not nullable fields or the Autoincrement field, if any. Even with a bad table design you will be able to edit data in the tables using an edit dialog.



Every table may be exported as insert or update sql scripts or as CSV data. Advanced modes for data export are available from the **advanced query tool**.

Querying Data

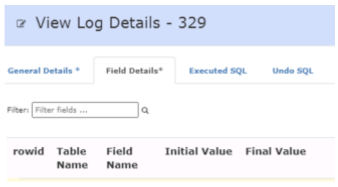
Querying data is possible also for non-technical people, they can choose the data filters in a dialog like the edit dialog and the SQL query is built for them. The query dialog will be displayed before viewing any table if the row count in that table is too big. Application will not get infinite data from server, all the database inquiries are **limited** to a number of rows (changeable by environment/business type, if needed).

Query parameters are saved in the session so every time you query tables having the same field name like the ones your queried for before, the query dialog is prefilled with the values you previously queried, helping users to save time and offering a **fast** way to work with data.

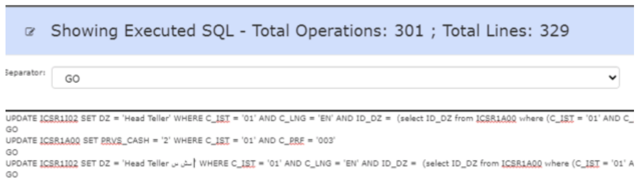
For technical users we offer an [advanced mode] where they can execute any SQL and DDL queries.

Database changes history

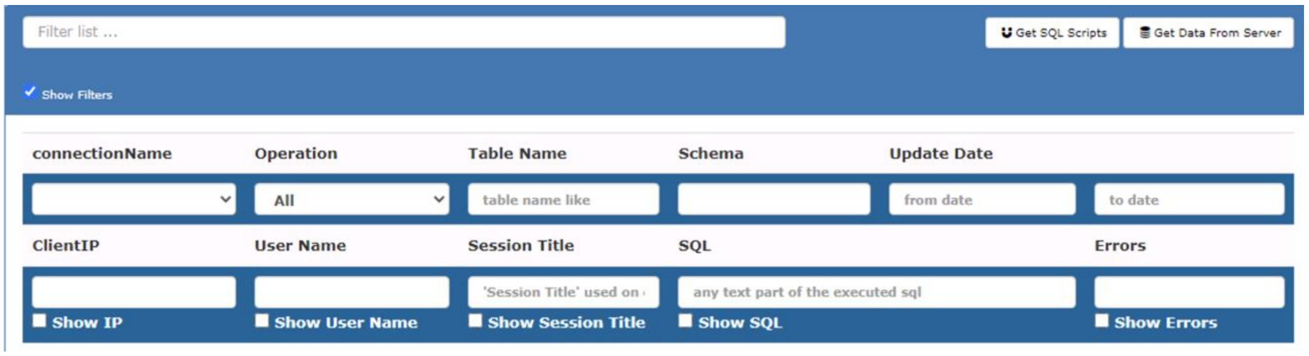
All the modifications are **logged** (together with date, user, IP information and a change reason). The log also contains information about table data before the modification, an **undo** SQL script (in case you will need later to undo the changes) plus the executed modification script in case you want to **propagate the changes** to another environment. The reason attached to every change in the data will help to propagate/undo only a specific activity related changes.



rowid	Table Name	Field Name	Initial Value	Final Value


Showing Executed SQL - Total Operations: 301 ; Total Lines: 329
Separator: GO

```
UPDATE ICSR1I02 SET DZ = 'Head Teller' WHERE C_IST = '01' AND C_LNG = 'EN' AND ID_DZ =  (select ID_DZ from ICSR1A00 where (C_IST = '01' AND C_
GO
UPDATE ICSR1A00 SET PRVS_CASH = '2' WHERE C_IST = '01' AND C_PRF = '003'
GO
UPDATE ICSR1I02 SET DZ = 'Head Teller ش س' WHERE C_IST = '01' AND C_LNG = 'EN' AND ID_DZ =  (select ID_DZ from ICSR1A00 where (C_IST = '01' A
GO
```



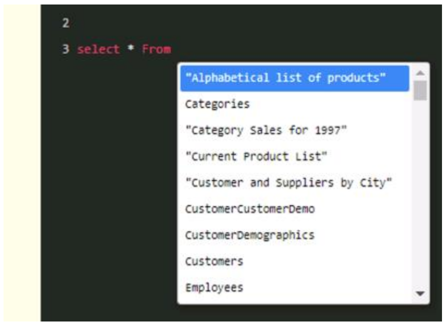
These features offer you a solid grip on your data, allowing to control changes, propagate changes to other environments or undo the changes.

Advanced Query Tool

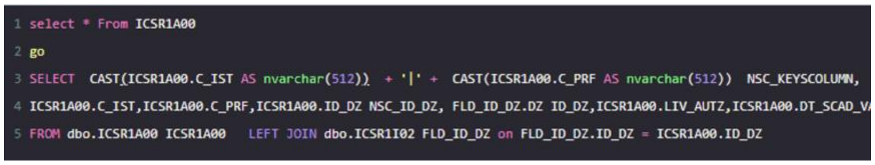
For advanced user there is a page where they can do anything with the database, execute any queries, updates, DDL etc.

The tool offers **intellisense autocompletion** for SQL keywords plus tables and table columns and other adavanced editing features (undo, redo, undo selection, delete line, brackets highlighting, searched text highlighting, 56 (fifty-six) color schemes and more)

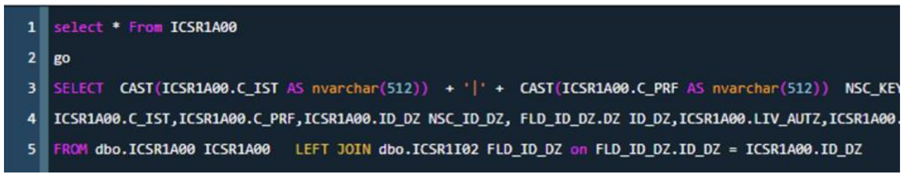
You may execute large amounts of SQL instructions, like from any other database tool.



```
2
3 select * From
```

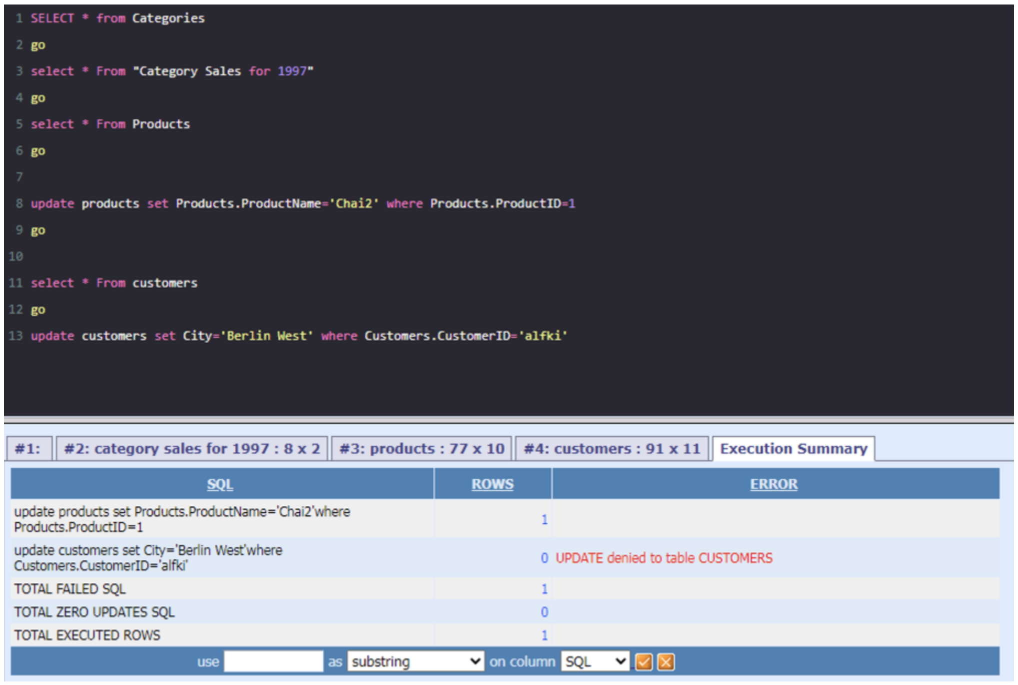


```
select * From ICSR1A00
go
SELECT  CAST(ICSR1A00.C_IST AS nvarchar(512))  + '|' +  CAST(ICSR1A00.C_PRF AS nvarchar(512))  NSC_KEYSCOLUMN,
ICSR1A00.C_IST,ICSR1A00.C_PRF,ICSR1A00.ID_DZ NSC_ID_DZ, FLD_ID_DZ.DZ ID_DZ,ICSR1A00.LIV_AUTZ,ICSR1A00.DT_SCAD_V
FROM dbo.ICSR1A00 ICSR1A00   LEFT JOIN dbo.ICSR1I02 FLD_ID_DZ on FLD_ID_DZ.ID_DZ = ICSR1A00.ID_DZ
```



```
select * From ICSR1A00
go
SELECT  CAST(ICSR1A00.C_IST AS nvarchar(512))  + '|' +  CAST(ICSR1A00.C_PRF AS nvarchar(512))  NSC_KEY
ICSR1A00.C_IST,ICSR1A00.C_PRF,ICSR1A00.ID_DZ NSC_ID_DZ, FLD_ID_DZ.DZ ID_DZ,ICSR1A00.LIV_AUTZ,ICSR1A00.
FROM dbo.ICSR1A00 ICSR1A00   LEFT JOIN dbo.ICSR1I02 FLD_ID_DZ on FLD_ID_DZ.ID_DZ = ICSR1A00.ID_DZ
```

Multiple resultsets and execution summary is displayed after every action, as you can see in the below example.

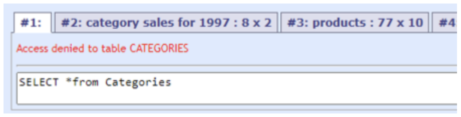


```
SELECT * from Categories
go
select * From "Category Sales for 1997"
go
select * From Products
go

update products set Products.ProductName='Chai2' where Products.ProductID=1
go

select * From customers
go
update customers set City='Berlin West' where Customers.CustomerID='alfki'
```

SQL	ROWS	ERROR
update products set Products.ProductName='Chai2'where Products.ProductID=1	1	
update customers set City='Berlin West'where Customers.CustomerID='alfki'	0	UPDATE denied to table CUSTOMERS
TOTAL FAILED SQL	1	
TOTAL ZERO UPDATES SQL	0	
TOTAL EXECUTED ROWS	1	



Access denied to table CATEGORIES

```
SELECT *from Categories
```

Please notice that the table rights are in place even from the advanced query tool, some tables having access denied and some update denied.

Also, after every action, the execution summary offers information about total updated rows, about failed sqls and even for zero updates sql (instructions that executed without errors but no data was updated), because is usefull to know if an expected update did not update any data.

Also there is available the option to **upload SQL files** even for **huge** scripting files (gigabytes) that will allow users to load huge amount of data by using SQL scripting (unlike aqua studio where the imported file size is limited).

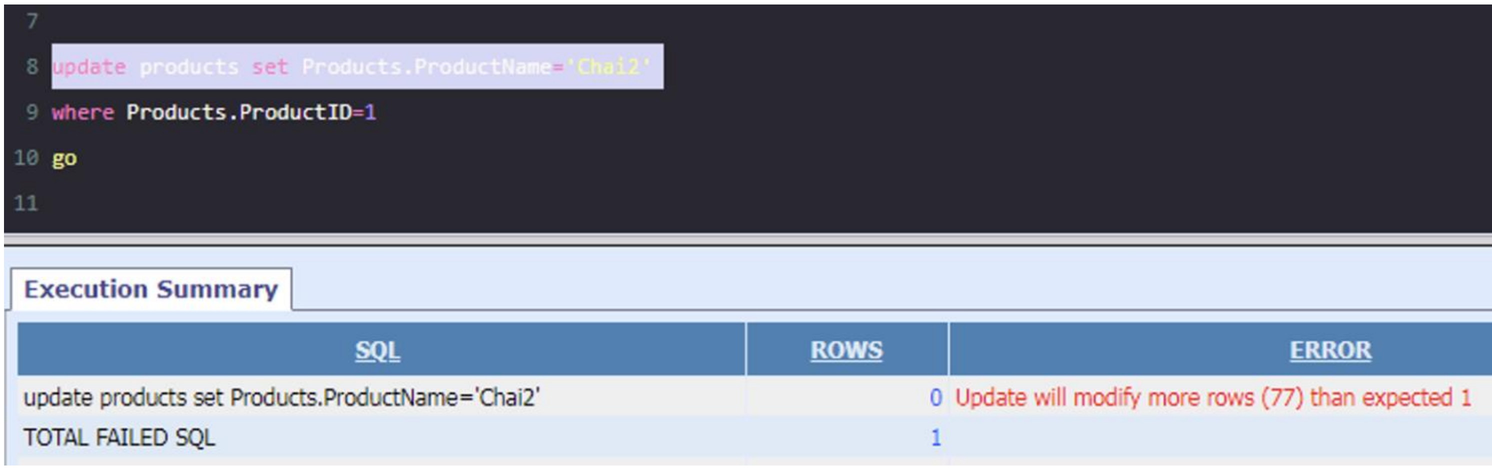
Unintended updates prevention

Application will prevent users from making unintended updates or deletes.

One key feature is the **impossibility to affect more rows than the intended ones**, by mistake.

If the sql command will affect more than one row the update will fail unless you also specify in advance how many rows you believe will be affected.

Default is always one, so if you send a query by mistake that will affect more than one the query will not execute. Below is an example, if you miss the where when sending the command the application will help by not executing it, unless you specified as expected row count a number equal or higher than 77. More, the maximum number of affected rows allowed can be set in application setting for each group of users and environment/connection.



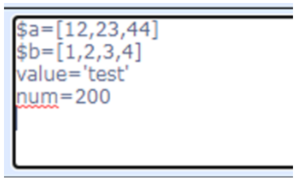
```
update products set Products.ProductName='Chai2'
where Products.ProductID=1
go
```

SQL	ROWS	ERROR
update products set Products.ProductName='Chai2'	0	Update will modify more rows (77) than expected 1
TOTAL FAILED SQL	1	

Variables

While in Advanced Query Tool, if you click on the **Variables** button, a textarea will be revealed, allowing you to define "variables". They will be used to REPLACE TEXT in your SQL instructions BEFORE sending them to the server.

Also, you can use **CTRL+R** to just replace the variables in the selected text in the editor (you can use it for all purposes).



```
$a=[12,23,44]
$b=[1,2,3,4]
value='test'
num=200
```

A variable is defined simply like: Variablename=variablevalue.

However, you can define **array variables**, by enclosing the values between [and].

This will make the replace process transform to a text generator process, the selected text will be duplicated for each result of the cartesian product in the arrays.

You can, for example, insert lots of data simply by defining the variables you want to be inserted and let the replace engine generate the cartesian product of text for you.

Note: if you want to use the character= in the variable name just escape the = like \=. Same, if you want to use a comma in the variable array element, escape it with \,

Data Export

You can easily export data from the query tool using SQL like instructions.

For any valid SELECT instruction you can replace the sql keyword select to

- EXPORT INSERT will generate insert scripts for the selected data
- EXPORT UPDATE will generate update scripts for the selected data
- EXPORT CREATE will generate create ddl scripts for the table
- EXPORT DROP will generate drop ddl scripts for all the objects of the table
- EXPORT EXCEL will generate an excel file with the selected data
- EXPORT COMPARE will generate scripts based on comparison between two databases

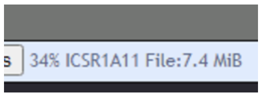
You can execute batch export instructions, a single file will be generated, containing the zipped data for all.

Export Excel will generate a single excel file one sheet per export instruction.

Insert, update, create and drop export instructions will generate one sql file with all the scripts written sequentially to the file, export excel will create a single excel file with multiple sheets, both in the same zipped archive that will be available for download to the user that executed the export commands.

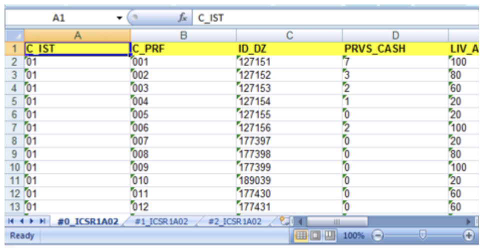
Important: once an export batch was started the user does not need to be logged in, especially since an export batch may take lot of time to complete. The export job is asynchronous and will execute regardless of the user interaction, once started.

Whenever the user will enter the advanced query tool page he can see the status of the export job, percentage done, exported file size so far and the current table of the job. Once the job is completed a link for download will be available.

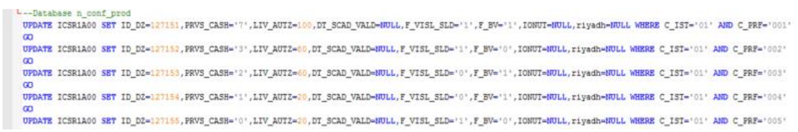


34% ICSR1A11 File:7.4 MiB

Note: however, the process can pe stopped if the user who launched the export process will start a new process. The running export process of the user (each user may have separate export processes) will be cancelled and a new export will start.



	A	B	C	D	
1	C_IST	C_PRF	ID_DZ	PRVS_CASH	LIV_A
2	01	001	127151	7	100
3	01	002	127152	3	80
4	01	003	127153	2	60
5	01	004	127154	1	20
6	01	005	127155	0	20
7	01	006	127156	2	100
8	01	007	177397	0	20
9	01	008	177398	0	80
10	01	009	177399	0	100
11	01	010	189039	0	20
12	01	011	177430	0	60
13	01	012	177431	0	60



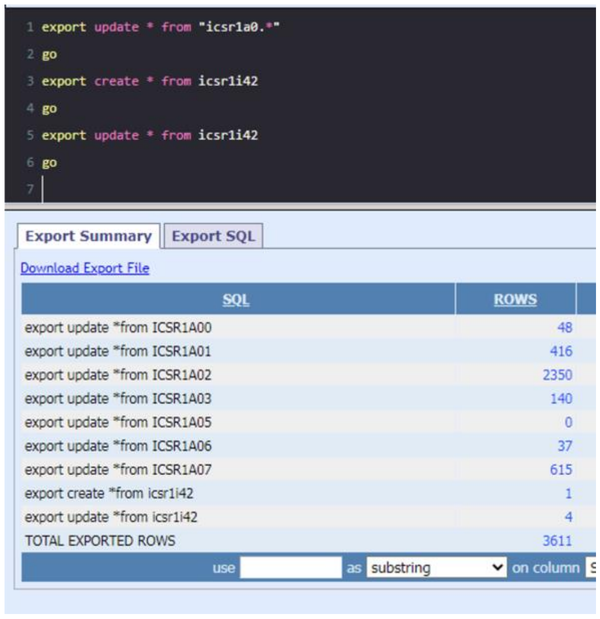
```
--Database n_conf_prod
UPDATE ICSR1A00 SET ID_DZ=127151,PRVS_CASH='7',LIV_AUTZ=100,DT_SCAD_VALD=NULL,F_VISL_SLD='1',F_BV='1',IONUT=NULL,riyadh=NULL WHERE C_IST='01' AND C_PRF='001'
GO
UPDATE ICSR1A00 SET ID_DZ=127152,PRVS_CASH='3',LIV_AUTZ=80,DT_SCAD_VALD=NULL,F_VISL_SLD='1',F_BV='0',IONUT=NULL,riyadh=NULL WHERE C_IST='01' AND C_PRF='002'
GO
UPDATE ICSR1A00 SET ID_DZ=127153,PRVS_CASH='2',LIV_AUTZ=60,DT_SCAD_VALD=NULL,F_VISL_SLD='0',F_BV='1',IONUT=NULL,riyadh=NULL WHERE C_IST='01' AND C_PRF='003'
GO
UPDATE ICSR1A00 SET ID_DZ=127154,PRVS_CASH='1',LIV_AUTZ=20,DT_SCAD_VALD=NULL,F_VISL_SLD='0',F_BV='1',IONUT=NULL,riyadh=NULL WHERE C_IST='01' AND C_PRF='004'
GO
UPDATE ICSR1A00 SET ID_DZ=127155,PRVS_CASH='0',LIV_AUTZ=20,DT_SCAD_VALD=NULL,F_VISL_SLD='1',F_BV='0',IONUT=NULL,riyadh=NULL WHERE C_IST='01' AND C_PRF='005'
```

Wildchar Table Naming

You may execute instructions (select, update, delete, export etc) on more tables based on table naming wildchar. Instead of writing hundreds of lines of scripting, one per each table, you can use .* in the table name and the application will execute the instruction for all the tables matching the wildchar rule.

Note: when using wildchars you have to enclose the table name between " and ".

Example: using "icsr1a0.*" for the table name will make the application to execute the instruction for all tables starting with icsr1a0. Below an example:



```
export update * from "icsr1a0.*"
go
export create * from icsr1i42
go
export update * from icsr1i42
go
```

Export Summary | Export SQL

Download Export File

SQL	ROWS
export update *from ICSR1A00	48
export update *from ICSR1A01	416
export update *from ICSR1A02	2350
export update *from ICSR1A03	140
export update *from ICSR1A05	0
export update *from ICSR1A06	37
export update *from ICSR1A07	615
export create *from icsr1i42	1
export update *from icsr1i42	4
TOTAL EXPORTED ROWS	3611

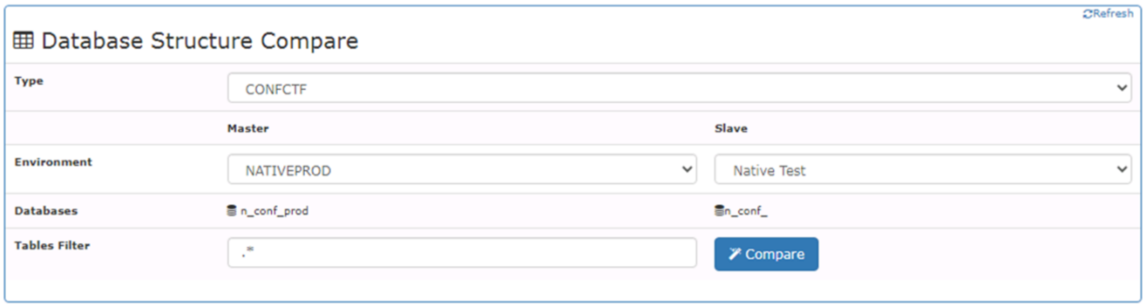
use as substring on column

Visual Structure Compare

It is available based on configuration and allows you to compare two databases (prod and test, for example) at structure level.

You simply choose the business type for which you want to compare and the databases, on the left the source, on the right the destination of comparison.

Note: if will see all the defined databases but only those for which you have the rights to connect will be available to compare.



Database Structure Compare

Type	CONFCTF	
	Master	Slave
Environment	NATIVEPROD	Native Test
Databases	n_conf_prod	n_conf_
Tables Filter	.*	Compare

If you are not already connected to the chosen databases, it may take several seconds (depending on how large the DB is) to load the database objects before comparison starts.

Results will offer information about missing tables, missing fields or different field attributes.

Note: only tables having differences will be displayed in the results list, only fields having differences also will be shown.

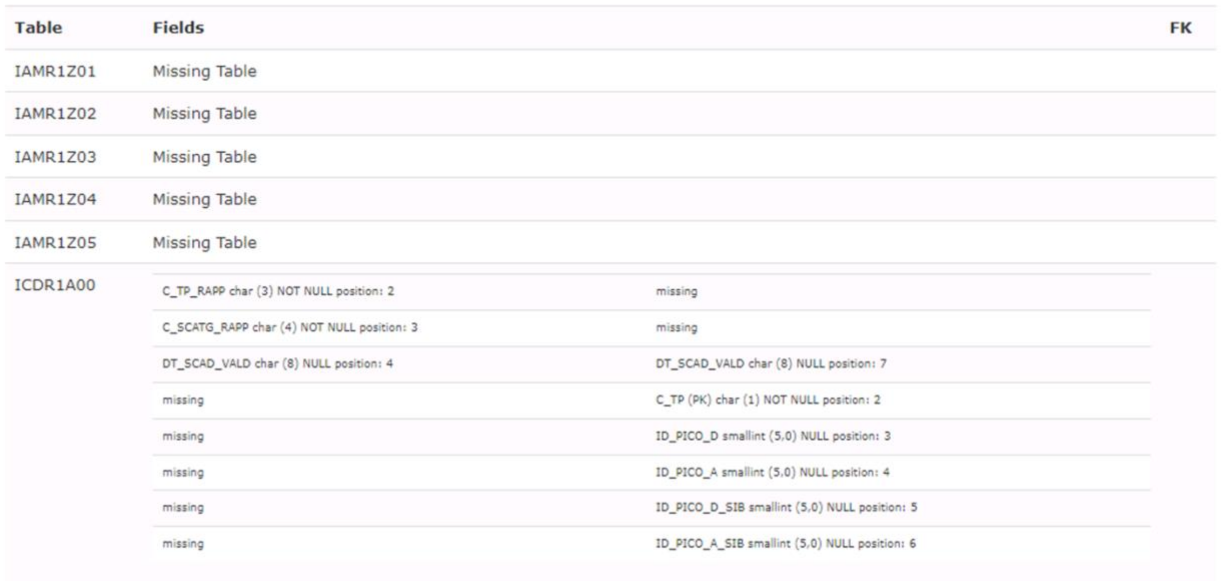


Table	Fields		FK
IAMR1Z01	Missing Table		
IAMR1Z02	Missing Table		
IAMR1Z03	Missing Table		
IAMR1Z04	Missing Table		
IAMR1Z05	Missing Table		
ICDR1A00	C_TP_RAPP char (3) NOT NULL position: 2	missing	
	C_SCATG_RAPP char (4) NOT NULL position: 3	missing	
	DT_SCAD_VALD char (8) NULL position: 4	DT_SCAD_VALD char (8) NULL position: 7	
	missing	C_TP (PK) char (1) NOT NULL position: 2	
	missing	ID_PICO_D smallint (5,0) NULL position: 3	
	missing	ID_PICO_A smallint (5,0) NULL position: 4	
	missing	ID_PICO_D_SIB smallint (5,0) NULL position: 5	
	missing	ID_PICO_A_SIB smallint (5,0) NULL position: 6	

Database Compare Scripting

When you need to compare two databases (compare same business type database between prod and test, for example) we offer you a powerful tool to help you with it.

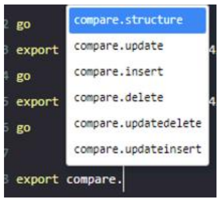
Using the EXPORT COMPARE command will generate scripts that will align the "destination" database with the "source" or "master" database.

Basically, the compare command will compare the tables specified in the command from both databases. It will generate any DDL needed to align the structure and also the SQL needed to align the data.

Available mods for compare are

- Compare structure will generate DDL scripts
- Compare update :scripts to update the existing rows from destination with the source db
- Compare insert : scripts to insert missing rows (rows that are missing in the destination)
- Compare delete: scripts to delete extra rows (from the source, if you want to align also the source)
- Compare updateinsert: same as insert and update
- Compare updatedelete: same as insert and delete

The available commands are also offered by the intellisense autocompletion.



```
go        compare.structure
export    compare.update
go        compare.insert
export    compare.delete
go        compare.updatedelete
          compare.updateinsert
export compare.
```

Note: no data is actually modified, scripts are generated only.

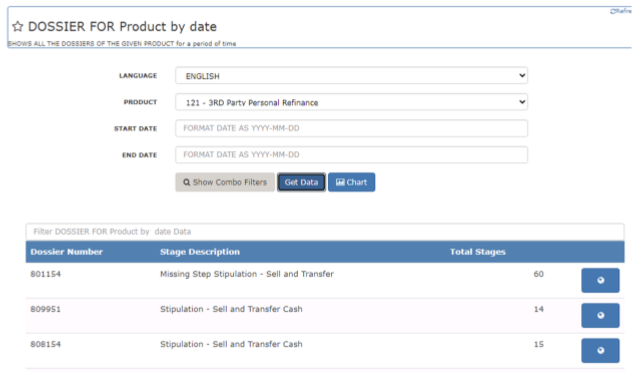
Reporting

You can create reports easily, reports will offer tabular data results and charts are also available. Reports are created environment transparent, every report created can be run in all environments.

Reports may be run also cross databases, you may take data from one database and use it as input for another set of data from a different database, within the same environment.

With a solid SQL knowledge you can create any kind of report. More, reports can be called in cascade, meaning for each result row it is possible to use it as input for another report to get detailed information and so on.

Report results can be exported as excel also.

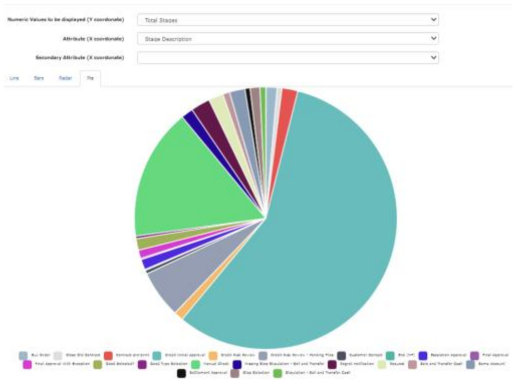


☆ DOSSIER FOR Product by date

SHOWS ALL THE DOSSIERS OF THE GIVEN PRODUCT for a period of time

LANGUAGE	ENGLISH
PRODUCT	121 - 3RD Party Personal Refinance
START DATE	FORMAT DATE AS YYYY-MM-DD
END DATE	FORMAT DATE AS YYYY-MM-DD

Dossier Number	Stage Description	Total Stages
801154	Missing Step Stipulation - Sell and Transfer	60
809951	Stipulation - Sell and Transfer Cash	14
808154	Stipulation - Sell and Transfer Cash	15



Numeric Values to be displayed (Y coordonate): Total Stages
Attribute (X coordonate): Stage Description
Secondary Attribute (X coordonate):