2017

SQL reversing for Vertica

VERSION 1.0

Table of contents

Why do I need this utility? How do I use it?

'SQL reversing for Vertica' allows to perform reserve engineering and generate Vertica DB objects creation script files. It proves to be useful for:

1. Quick viewing of DB objects parameters;
2. Generation of scripts to deploy objects from current DB into a new one;
3. Tracing history of DB objects changes at a level of any file-based version control system.

How we use it:

- For daily automatic reverse engineering of Vertica DB with object generation script saving in SVN or GIT version control systems;
- At any moment in time to open a file for required object in text editor or SQL client to see its structure, parameters, mapping (for tables), access rights assigned to users and roles;
- With version control system we can easily monitor DB objects variation over time and compare versions;
- We use one script to generate all objects transfer from one Vertica DB to another when needed.

How does it differ from EXPORT_OBJECTS function in Vertica?

In addition to metadata of physical database model catalog (schemas, tables, views, counters and functions), 'SQL reversing for Vertica' allows processing other metadata (resource pools, roles, users and object access rights). The utility can save the data in a filesystem hierarchically ('catalog/file'), thus enabling to break DB object script generation down to types and schema affiliation.

Can this utility replace SAP PowerDesigner?

No. PowerDesigner along with 'PowerDesigner Vertica Plug-in' (http://easydata.ru/product/powerdesigner-vertica-plug-in/) makes it possible to complete the full cycle of data warehouses' visual physical simulation and conduct reverse engineering as well as comparison of two database models aimed at changing receiver DB objects to match source objects using ALTER script generation. However, PowerDesigner does not support resource pools and cannot generate a set of files to ensure version control with file-based version control system (only a proprietary version control system, allowable at purchase of respective PowerDesigner license, is supported).

Is it a freeware?

Yes, it is a freeware distributed 'as is' and can be found under 'GNU General Public License v3.0' at: https://github.com/ascrus/verticareverseutilite

Which OS are supported? Launch requirements

The software is written in Java/Groovy using JavaFX, therefore Windows, Linux and MacOS platforms are supported. Proper operation in Linux requires X to be installed in the OS. 30 Mb free disc space and 1 GB RAM are required to launch the software.

Which Vertica versions are supported?

Vertica 7.2.3 and later. We do not guarantee support of earlier Vertica versions.

Is command line support available?

Yes, the software allows to configure installation for launching reverse engineering from the command line without any graphics interface.

Which database objects are supported?

The current version supports:

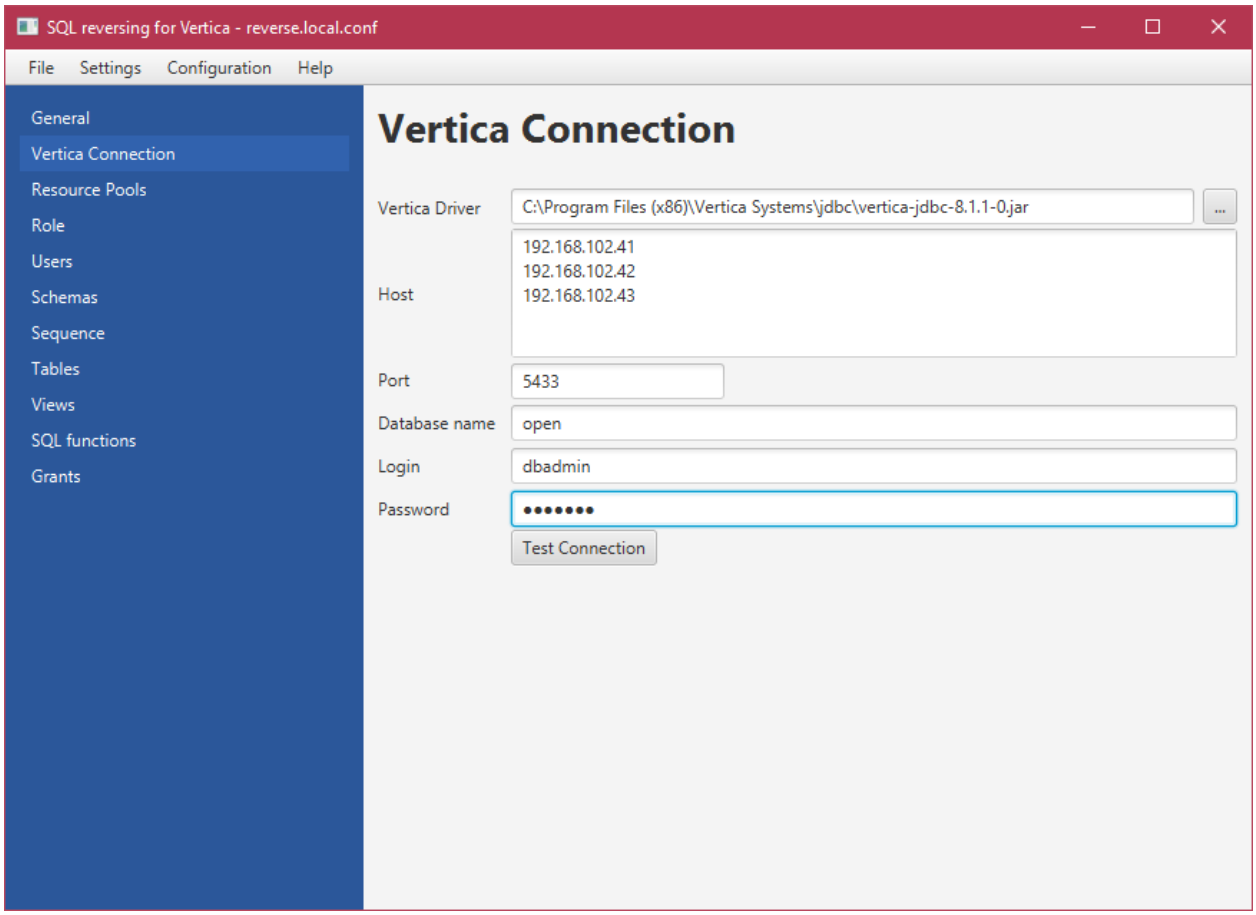
- Generation of scripts for resource pools and roles' and users' access rights;
- Generation of scripts to create roles and assign rights to them;
- Generation of scripts to create internal Vertica users and assign roles to them;
- Generation of scripts to create table schemas and assign owners and access rights;
- Generation of scripts to create permanent and temporary tables with constraints and projections description, as well as owners and access rights assignment;
- Generation of scripts to create views with assigning owners and respective access rights;
- Generation of scripts to create counters with assigning owners and respective access rights;
- Generation of scripts to create UDF functions with assigning owners and respective access rights.

Current version does not support:

- Database parameters;
- Database storage;
- Partitions storage policies;
- Users connected through LDAP;
- Schema objects' access rights inheritance;
- Flex tables;
- External tables;
- Aggregate projections;
- Service tables' data synchronization.

Connection to Vertica

Connection settings for required database shall be entered in 'Vertica Connection' tab:

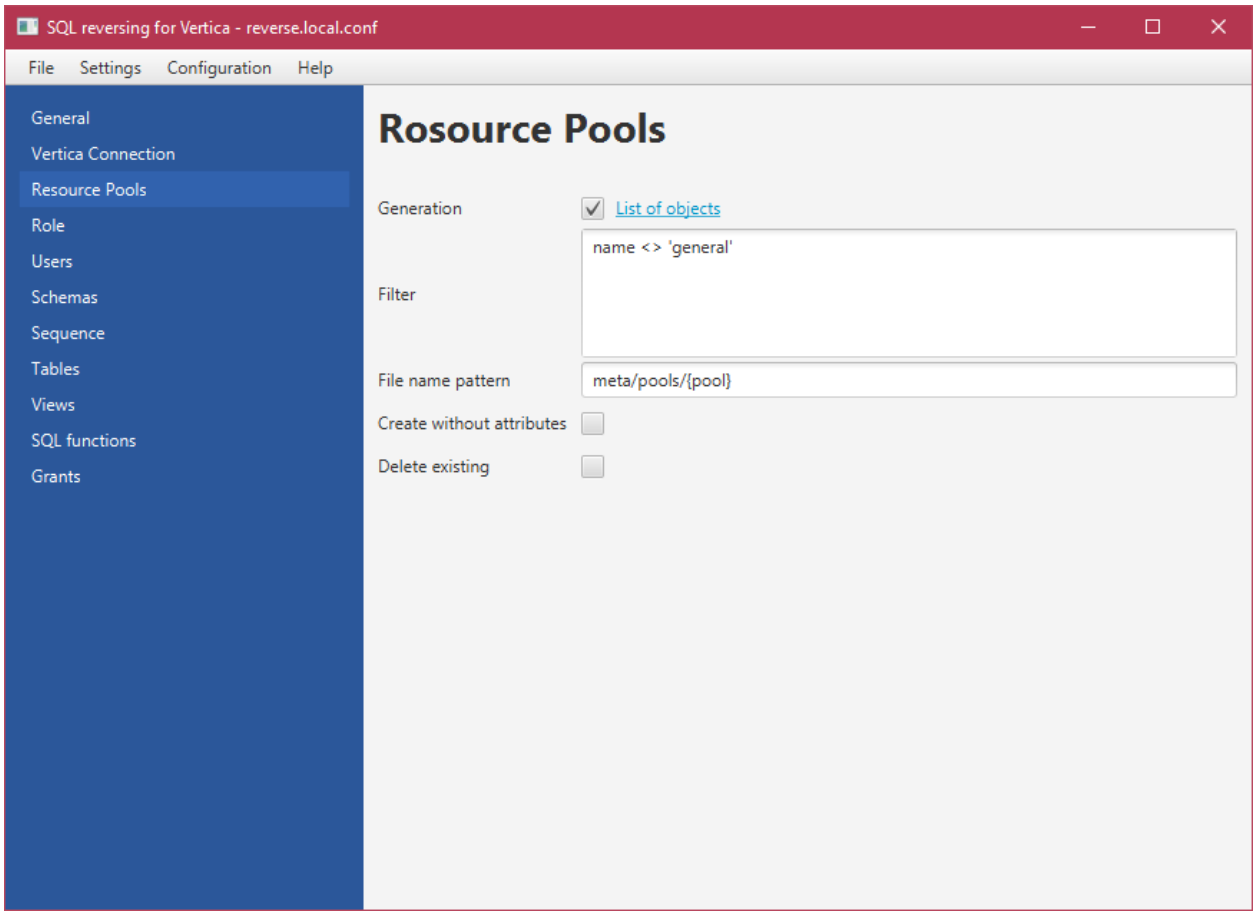


Specify path to jar file of Vertica JDBC driver, using the version compatible with the server's version. Driver's file shall be accessible and valid. Further generation of configuration command-line running installation will include driver's file in distribution package.

If connections balancing is supported by the server, use 'host' field to indicate the list of clusters' nodes allowing the connection. Do not list all cluster nodes; two or three will suffice to enable connection in case one of the hosts is unavailable. When more than one host is specified, connection will be automatically established with any unloaded cluster node.

Resource pools

Use 'Resource pools' tab to set resource pools generation parameters:



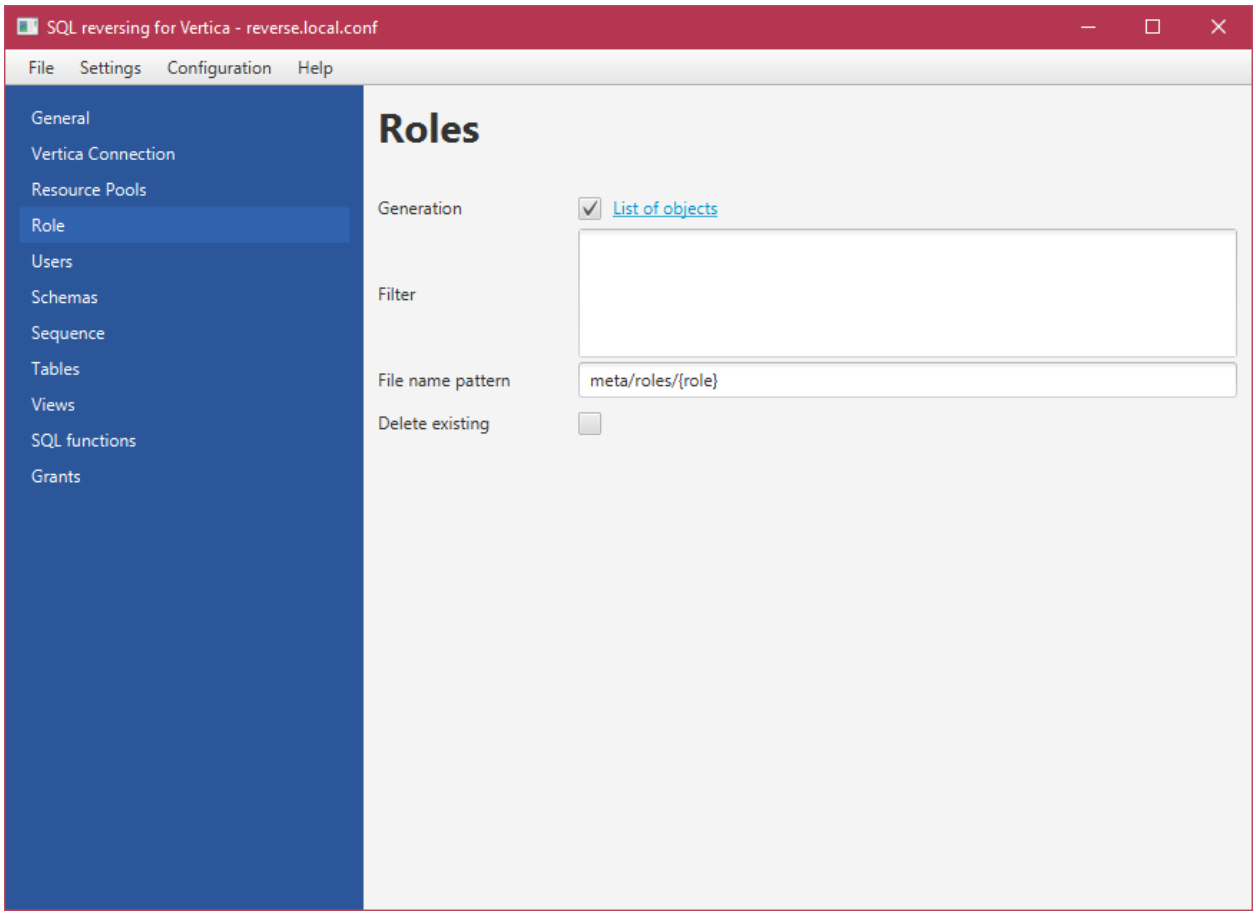
If a specific list of resource pools selected for generation is required, use 'Filter' field to specify selection expression with ANSI SQL expression language. Use filter to access the fields corresponding to Vertica 'v_catalog.resource_pools' table:
name, memorysize, maxmemorysize, executionparallelism, priority, runtimepriority, runtimeprioritythreshold, queuetimeout, plannedconcurrency, maxconcurrency, runtimecap, cpuaffinityset, cpuaffinitymode, cascadeto.

To specify the name of generated script files, both static and dynamic relative path and file name can be set using pool macrovariable. When the static name is specified, all objects, their parameters and access rights will be written into the same file.

If pool creating scripts are required to be generated without pool parameters, deselect 'Create without attributes' checkbox. The script will specify pools with default parameters. To remove existing pools before creating new ones, select 'Delete existing' checkbox.

Users' rights

Roles generation parameters can be set in 'Roles' tab:



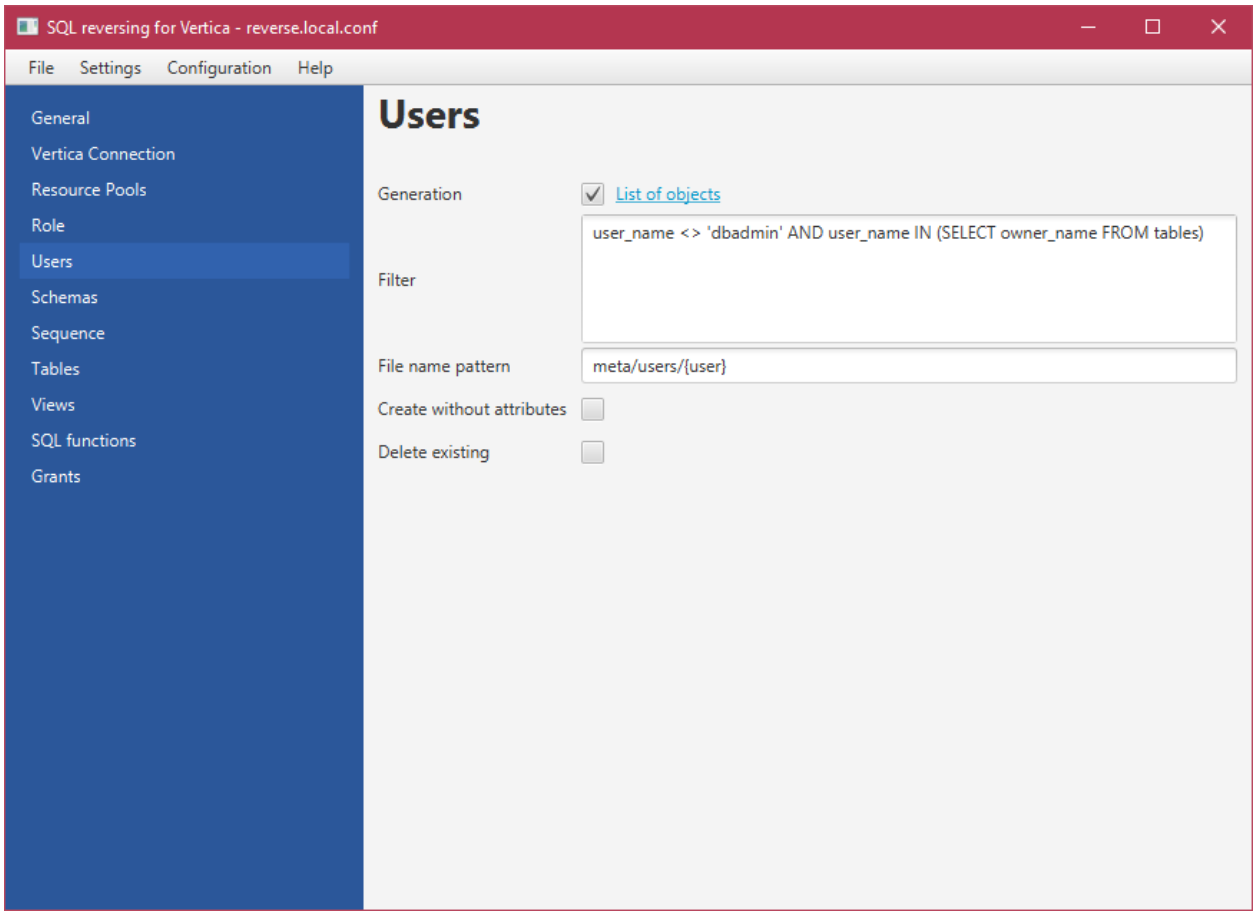
If a specific list of roles for generation is required, use 'Filter' field to specify selection expression with ANSI SQL expression language. Filter allows accessing 'name' field.

To specify the name of generated script files, both static and dynamic relative path and file name can be set using role macrovariable. When the static name is specified, all objects, their parameters and access rights will be written into the same file.

To remove existing roles before creating new ones, select 'Delete existing' checkbox.

Users

Users generation parameters can be set in 'Users' tab:



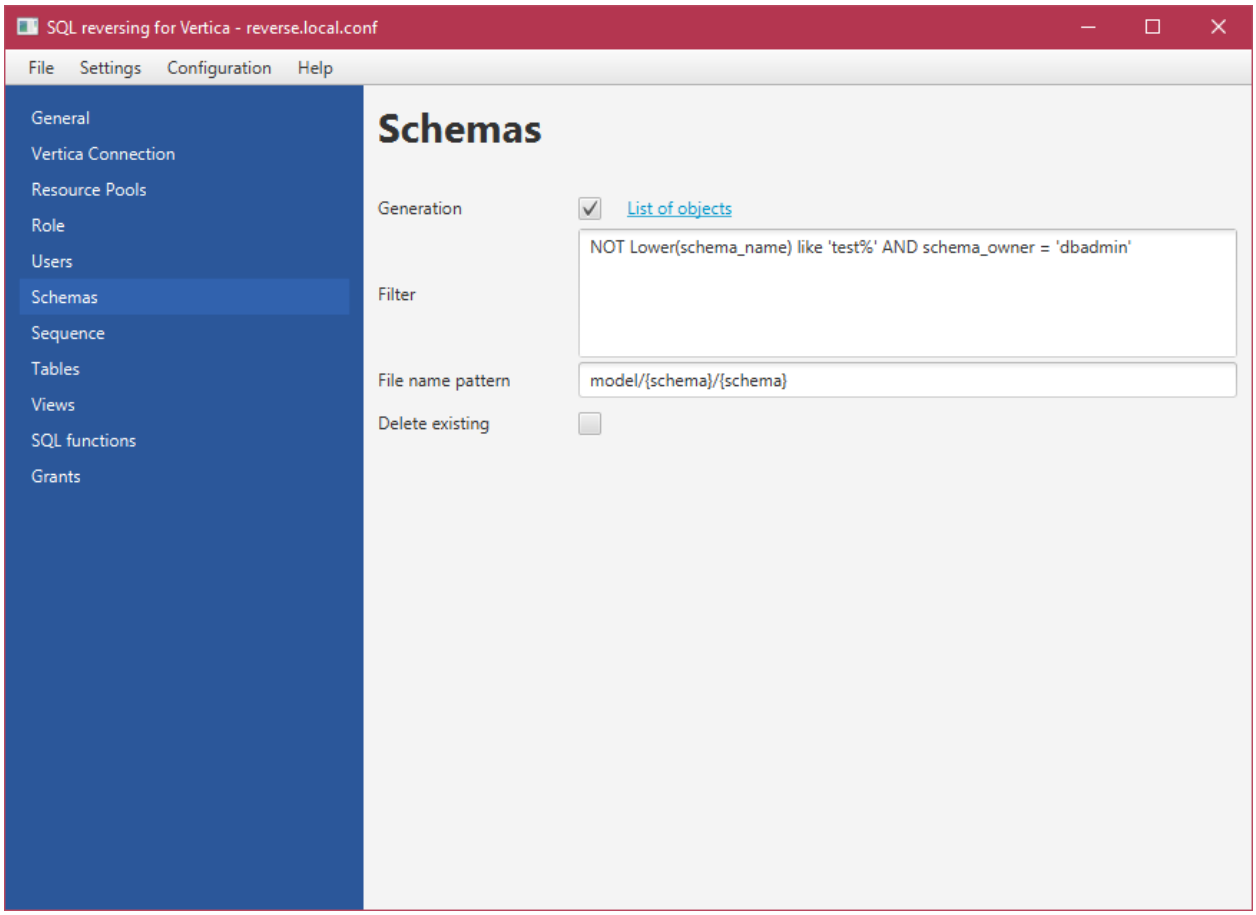
If a specific list of users selected for generation is required, use 'Filter' field to specify selection expression with ANSI SQL expression language. Filter allows accessing fields, corresponding to Vertica 'v_catalog.users' table: user_name, resource_pool, memory_cap_kb, temp_space_cap_kb, run_time_cap, all_roles, default_roles, search_path.

To specify the name of generated script files, both static and dynamic relative path and file name can be set using user macrovariable. When the static name is specified, all objects, their parameters and access rights will be written into the same file.

If users creating scripts are required to be generated without their parameters, deselect 'Create without attributes' checkbox. To remove existing users before creating new ones, select 'Delete existing' checkbox.

Table schemas

Objects' schema generation parameters can be set in 'Schemas' tab:



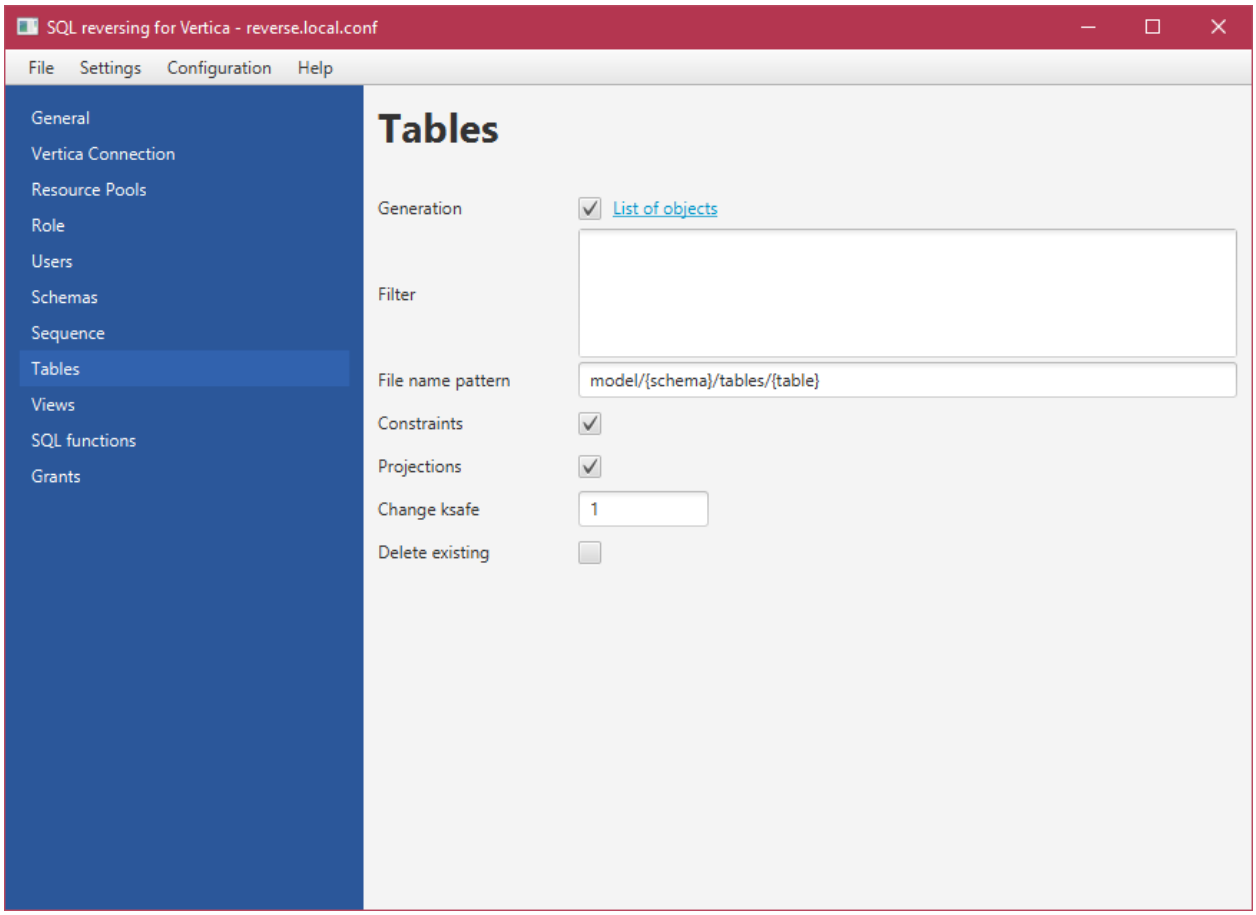
If a specific list of schemas for generation is required, use 'Filter' field to specify selection expression with ANSI SQL expression language. Filter allows accessing fields, corresponding to Vertica 'v_catalog.schemata' table: schema_name, schema_owner.

To show the name of generated script files, both static and dynamic relative path and file name can be set using schema macrovariable. When the static name is specified, all objects, their parameters and access rights will be written into the same file.

To remove existing schemas before creating new ones, select 'Delete existing' checkbox.

Tables

Objects' schema generation parameters can be set in 'Tables' tab:



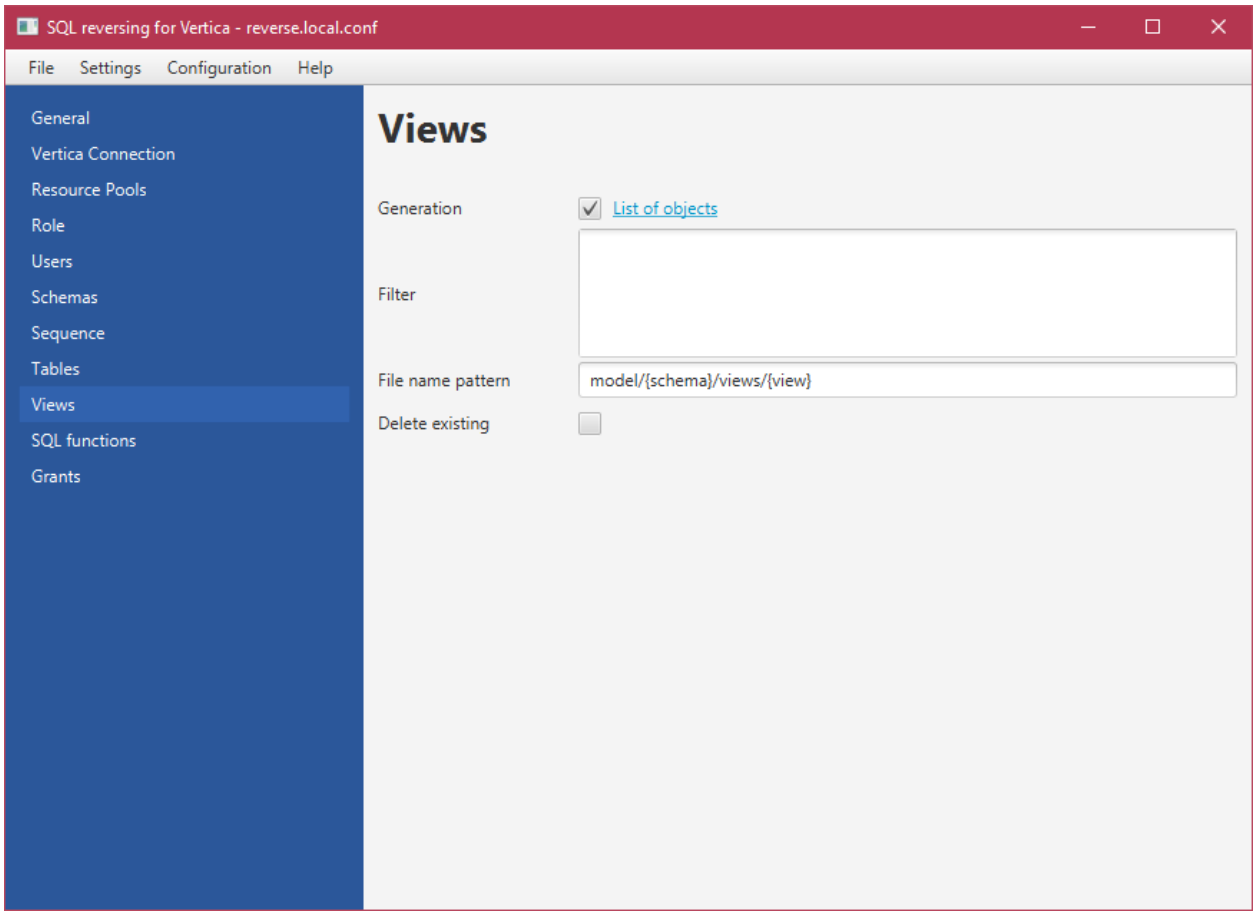
If a specific list of tables for generation is required, use 'Filter' field to specify selection expression with ANSI SQL expression language. Filter allows accessing fields, corresponding to Vertica 'v_catalog.tables' table: table_schema, table_name, owner_name, is_temp_table.

To show the name of generated script files, both static and dynamic relative path and file name can be set using schema and table microvariables. When the static name is specified, all objects, their parameters and access rights will be written into the same file.

Use the tab to include constrains and projections in tables creation. Assign ksafe change for projections with lower threshold, which is useful for database single node installation. Comments generation is supported for tables and columns of projections. To remove existing tables before creating new ones, select 'Delete existing' checkbox.

Views

Objects' schema generation parameters can be set in 'Views' tab:



If a specific list of schemas for generation is required, use 'Filter' field to specify selection expression with ANSI SQL expression language. Filter allows accessing fields, corresponding to Vertica Vertica 'v_catalog.views' table: table_schema, table_name, owner_name, table_id.

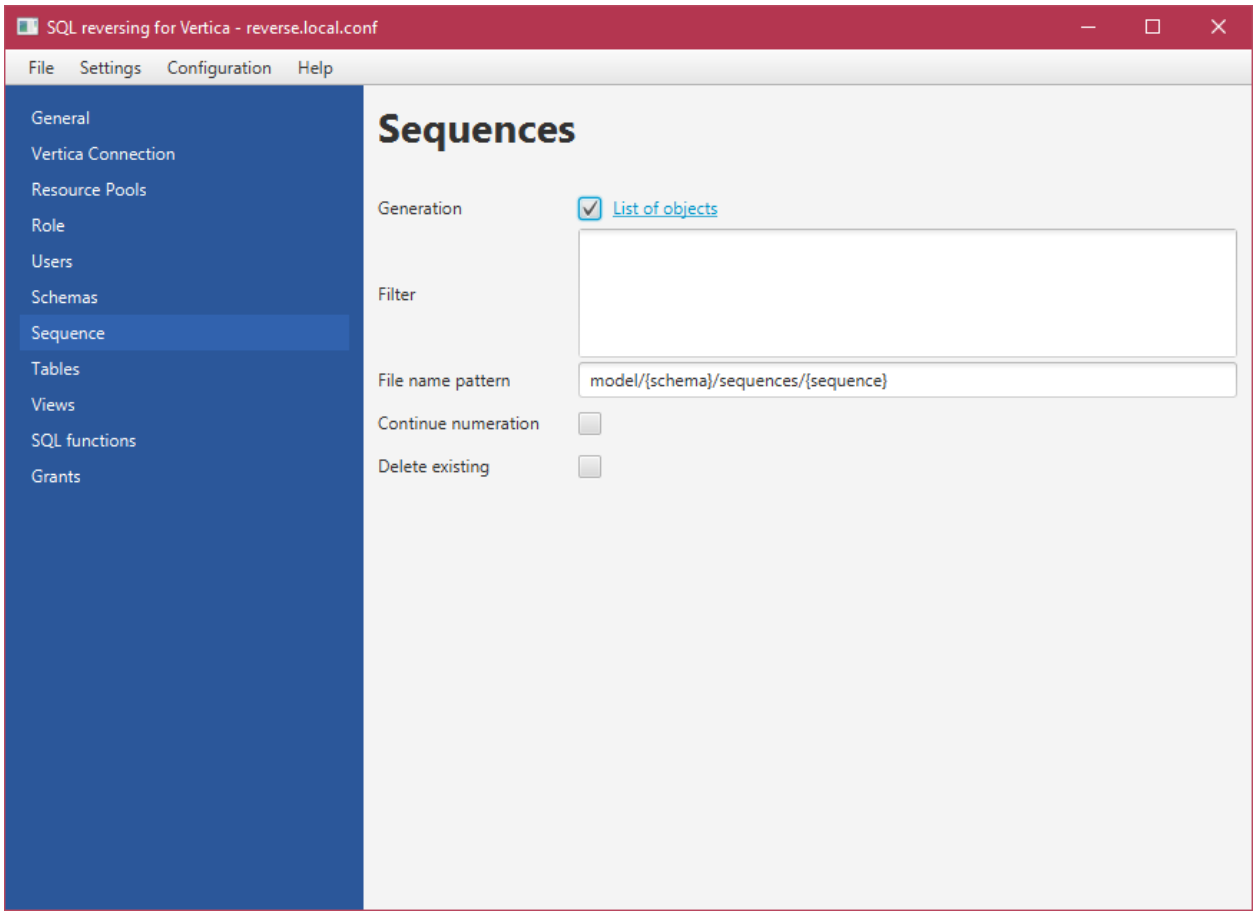
To specify the name of generated script files, both static and dynamic relative path and file name can be set using schema and view microvariables. When the static name is specified, all objects, their parameters and access rights will be written into the same file.

To remove existing views before creating new ones, select 'Delete existing' checkbox.

P.S. as it is impossible to identify interconnections between the views, their generation in the same file is described alphabetically.

Counters

Objects' schema generation parameters can be set in 'Sequence' tab:



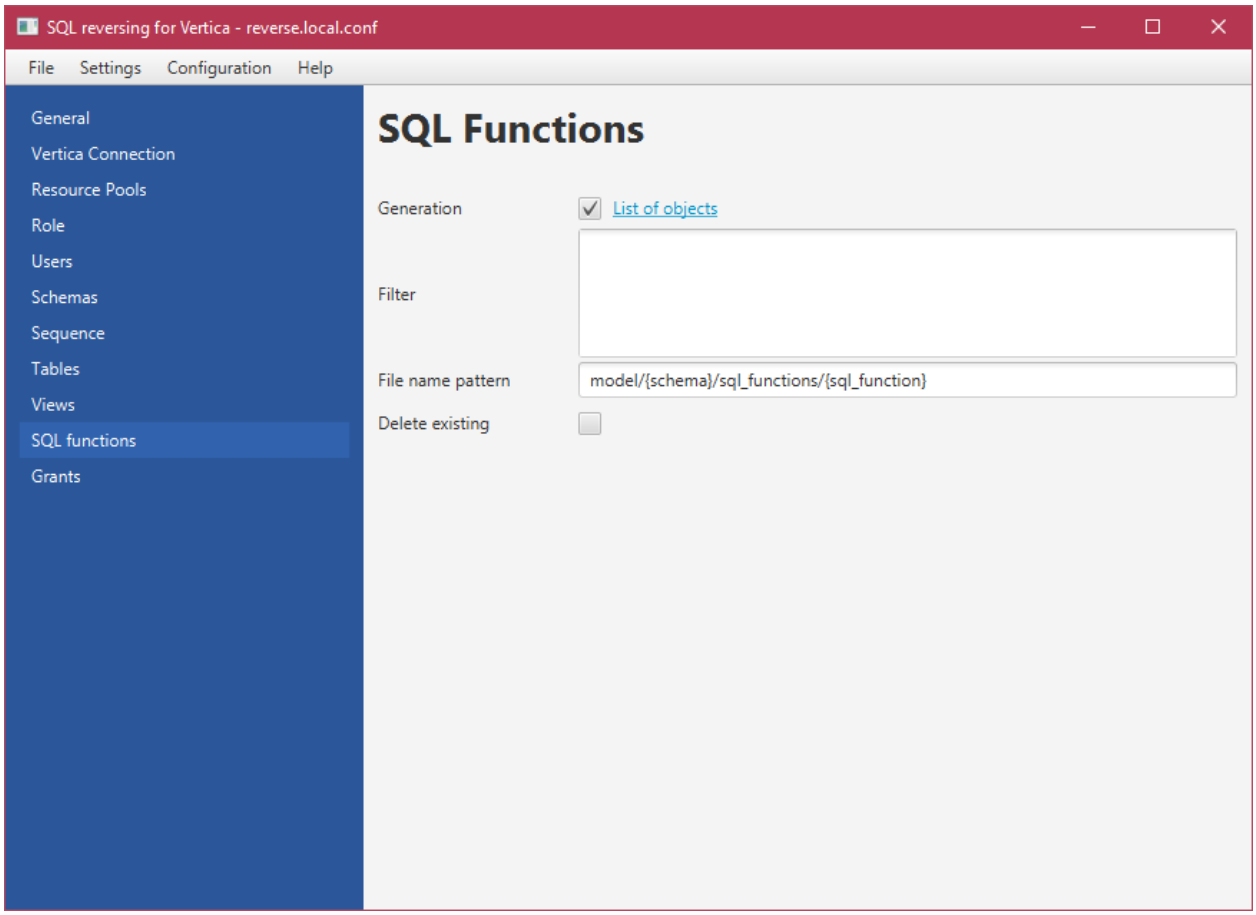
If a specific list of counters selected for generation is required, use 'Filter' field to specify selection expression with ANSI SQL expression language. Filter allows accessing fields, corresponding to Vertica 'v_catalog. sequences' table: sequence_schema, sequence_name, owner_name, identity_table_name, session_cache_count, allow_cycle, output_ordered, increment_by, current_value.

To specify the name of generated script files, both static and dynamic relative path and file name can be set using schema and sequence microvariables. When the static name is specified, all objects, their parameters and access rights will be written into the same file.

If a counter to continue numbering from the current value at reversing is required, select 'Continue numeration' checkbox. To remove existing counters before creating new ones, select 'Delete existing' checkbox.

Functions with SQL expression

Objects' schema generation parameters can be set in 'SQL functions' tab:



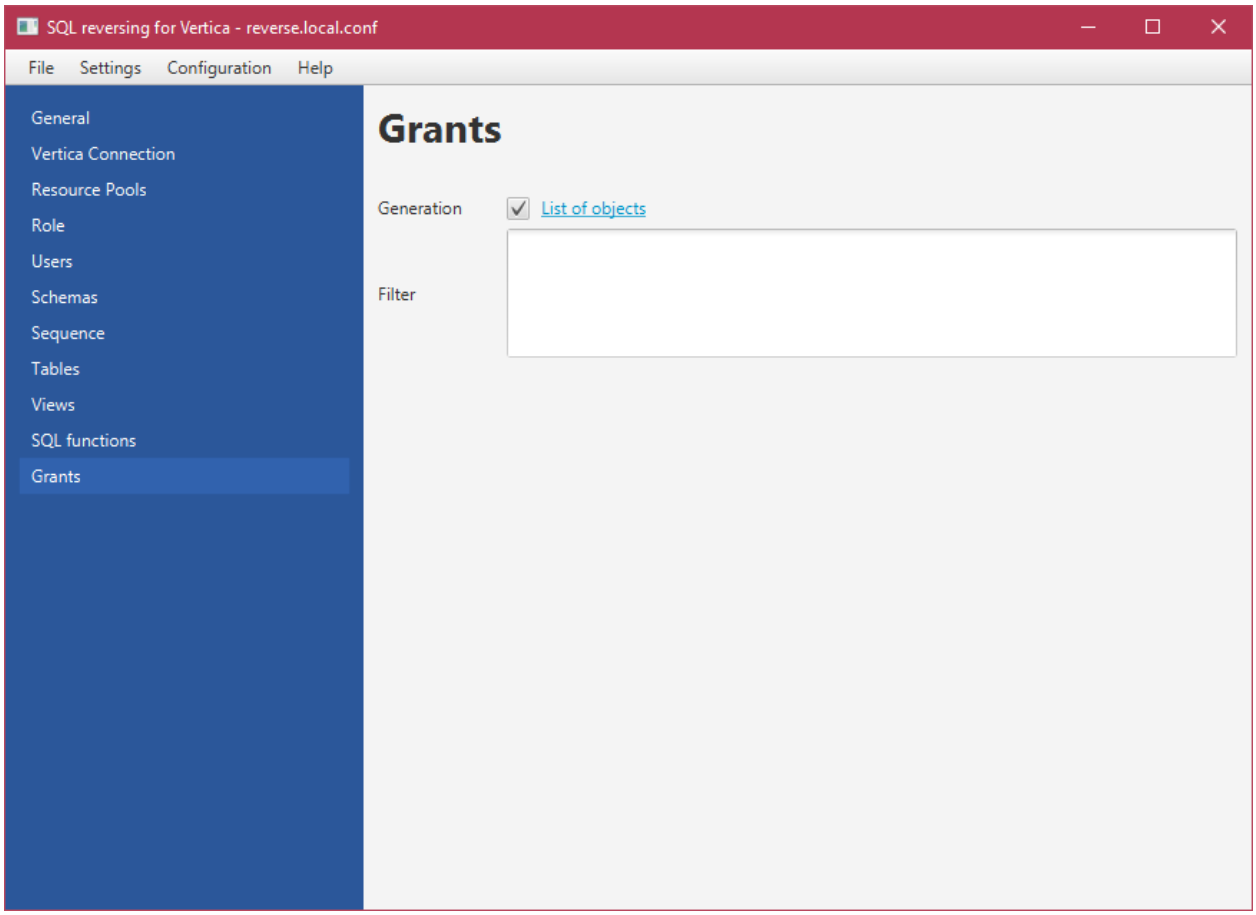
If a specific list of functions for generation is required, use 'Filter' field to specify selection expression with ANSI SQL expression language. Filter allows accessing fields, corresponding to Vertica 'v_catalog.user_functions' table: schema_name, function_name, owner.

To specify the name of generated script files, both static and dynamic relative path and file name can be set using schema and sql_function microvariables. When the static name is specified, all objects, their parameters and access rights will be written into the same file.

To remove existing functions before creating new ones, select 'Delete existing' checkbox.

Objects grants

Objects' schema generation parameters can be set in 'Grants' tab:



If a specific list of grants for generation is required, use 'Filter' field to specify selection expression with ANSI SQL expression language. Filter allows accessing fields, corresponding to Vertica 'v_catalog.grants' table: grantor, privileges_description, object_type, object_schema, object_name, grantee, function_argument_type.

No separate files are generated for grants. They are instead included in files of objects they are intended for. GRANT statement generation is provided for all objects supported by the software.

Working with object filters

Setting a filter in a tab may lead to filtering of subordinate objects. See below for mutual influence of various objects' types:

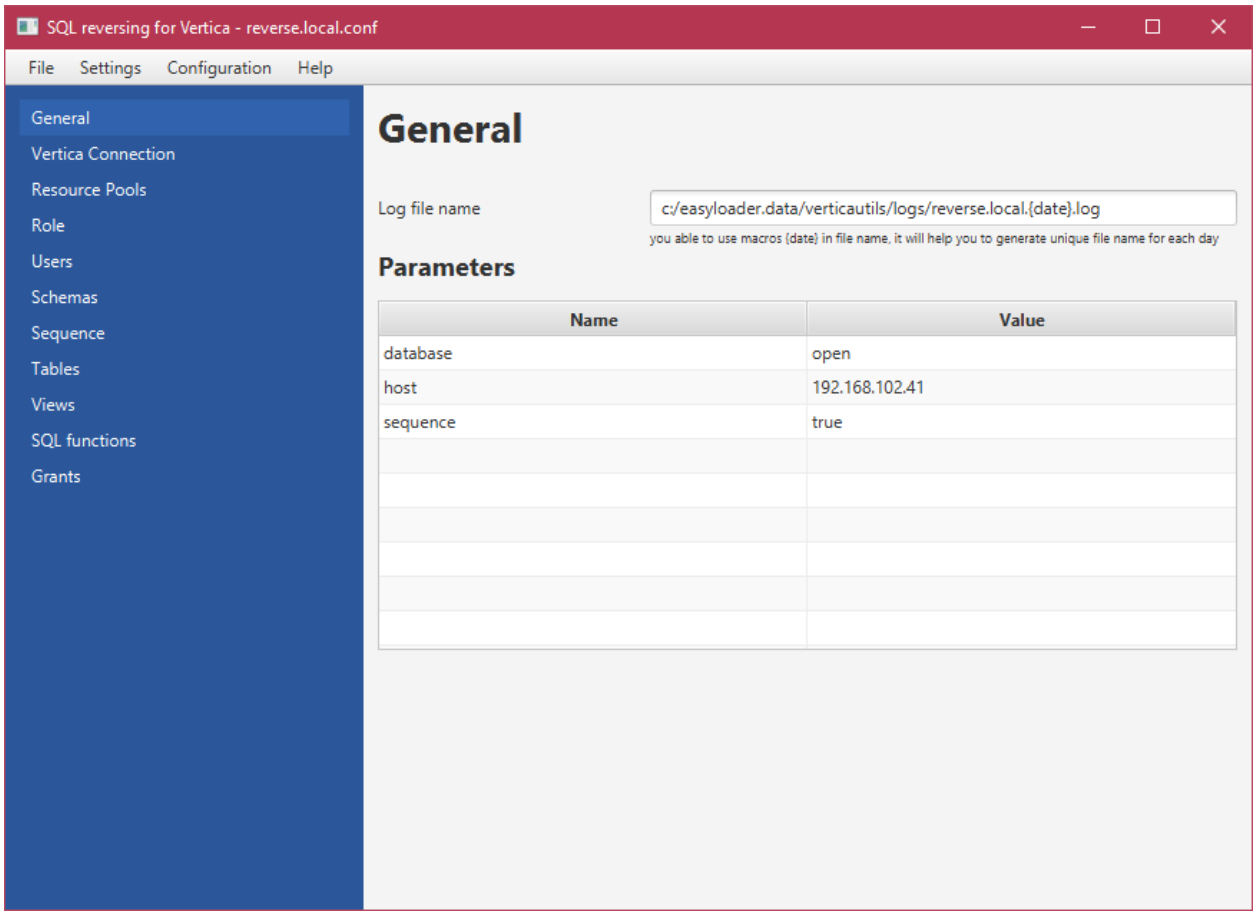
- List of users filter is considered at generation of table schemas, tables, views, counters and functions: excluded are objects with no owners;
- List of table schemas filter is considered at generation of tables, views, counters and functions: excluded are objects with no schemas;
- Role and users filter is considered at generation of DB object access rights: excluded are permission for absent roles and users.

For example, when a table schema condition is set, no filter for schema's objects is required; only objects within schemas filter will be selected.

Parsing of Vertica DB model initially stores all metadata in the in-memory array to apply filters, specified in reversing configuration. It allows using IN (subquery) and EXISTS (subquery) structures in filters to receive dependencies' data. The following metadata tables are available: pools, roles, users, schemas, tables, views, sequences, sql_functions and grants.

Set logging and parameters for values

Logging file name and list of variables are set in 'General' tab:



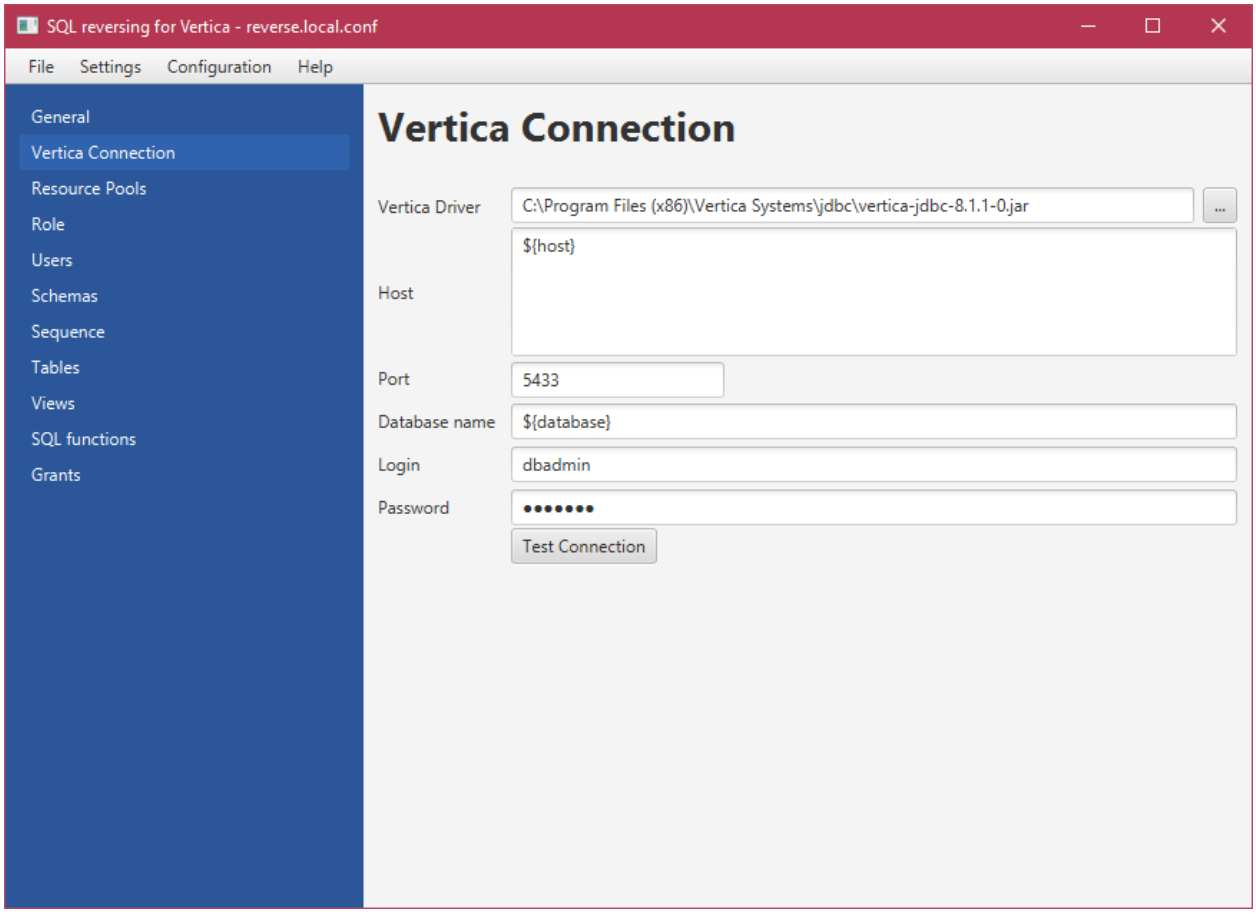
If utility operations shall be logged into a file, its name can be set in 'Log file name' field. {date} microvariable in a log file name is used when daywise records are required.

Names of variables shall contain Latin letters and numerals and Underscore. No spaces or other characters are allowed. Any text expression can be an entry. Names of variables are case sensitive.

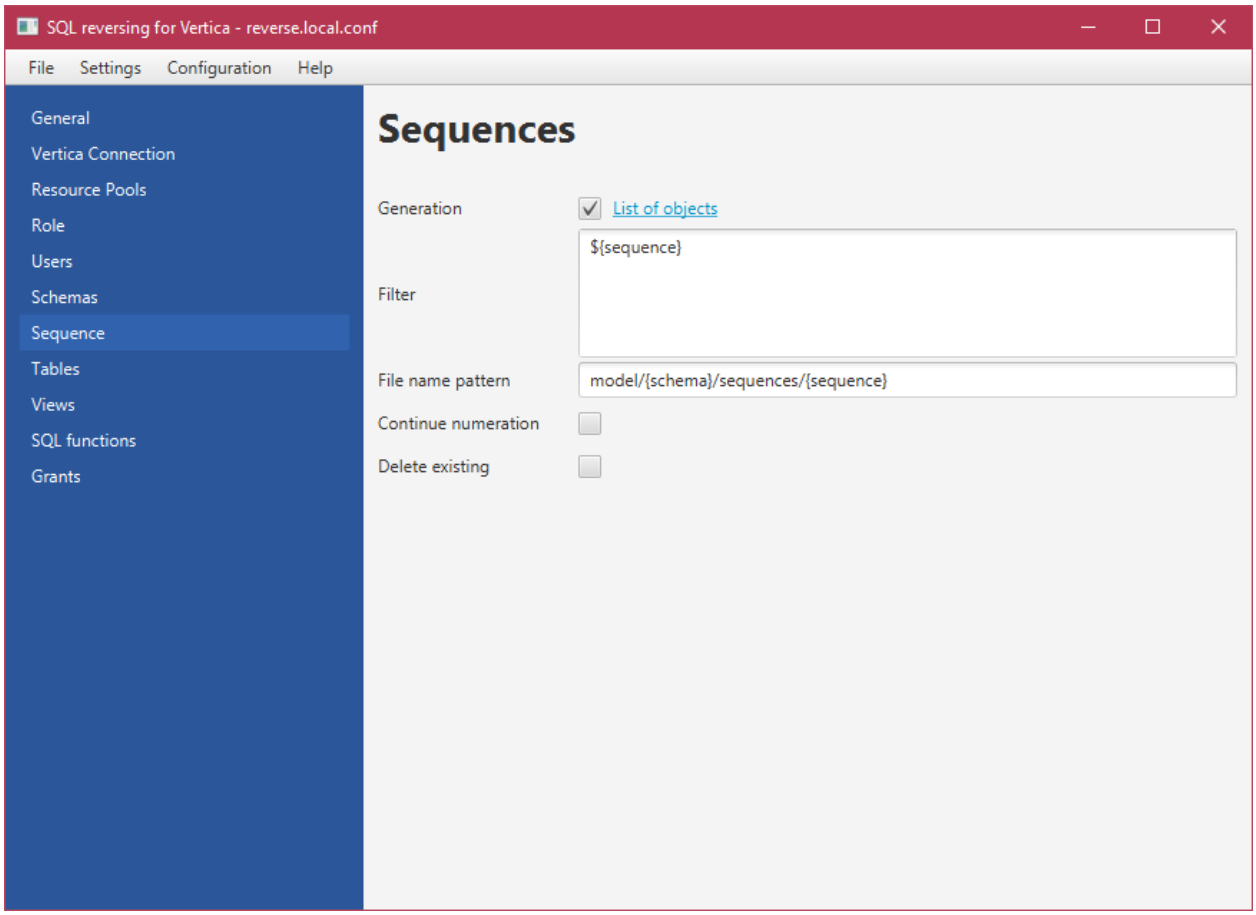
Use of supplied parameters

The software allows taking value of any configuration field as parameters. This is useful for configuring on different Vertica servers or for various runtime environments. Variables shown in 'General' tab, e.g. '${VariableName}' can be used.

Use of variables in Vertical connection parameters (only one address, shown in host variable will be used for the host):

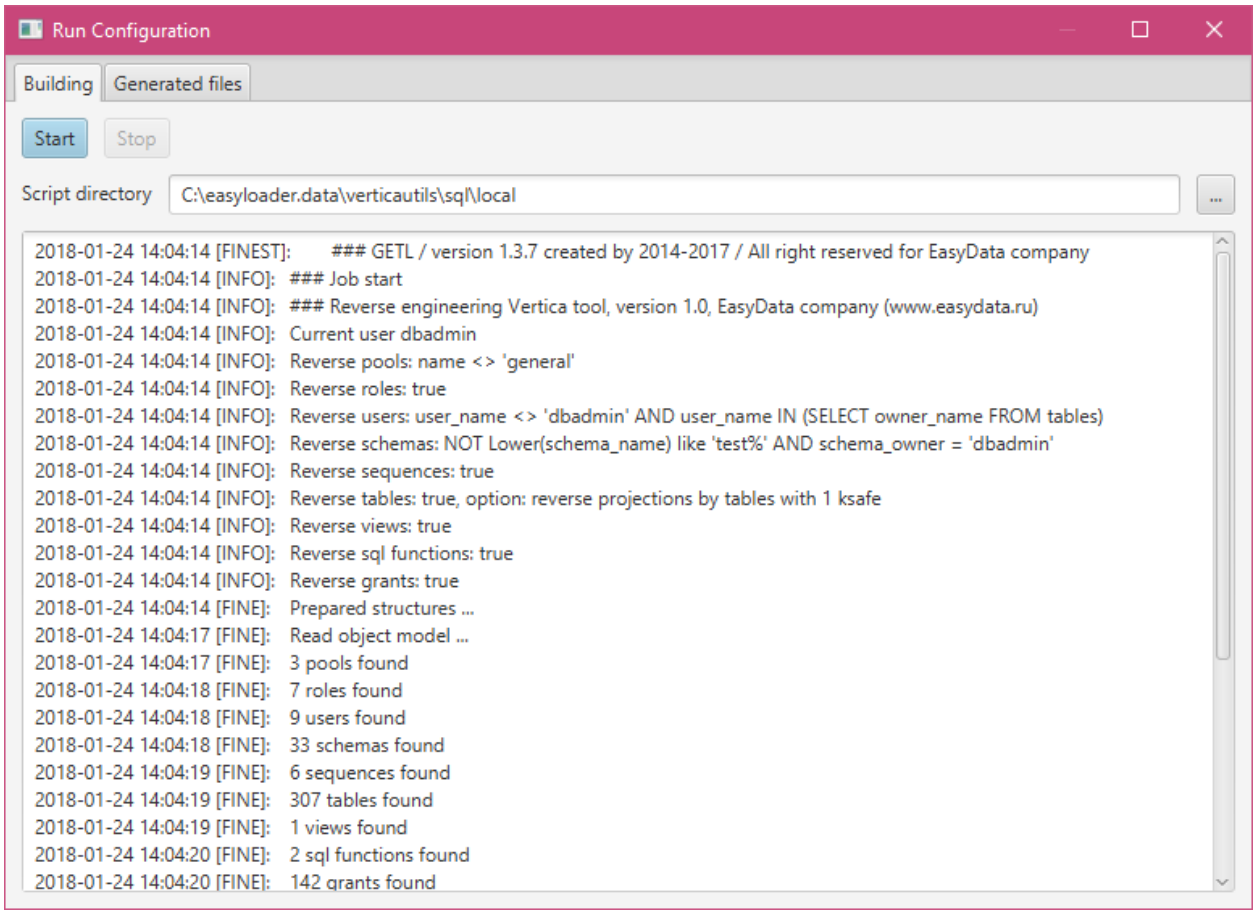


Use of variable to filter list of counters:

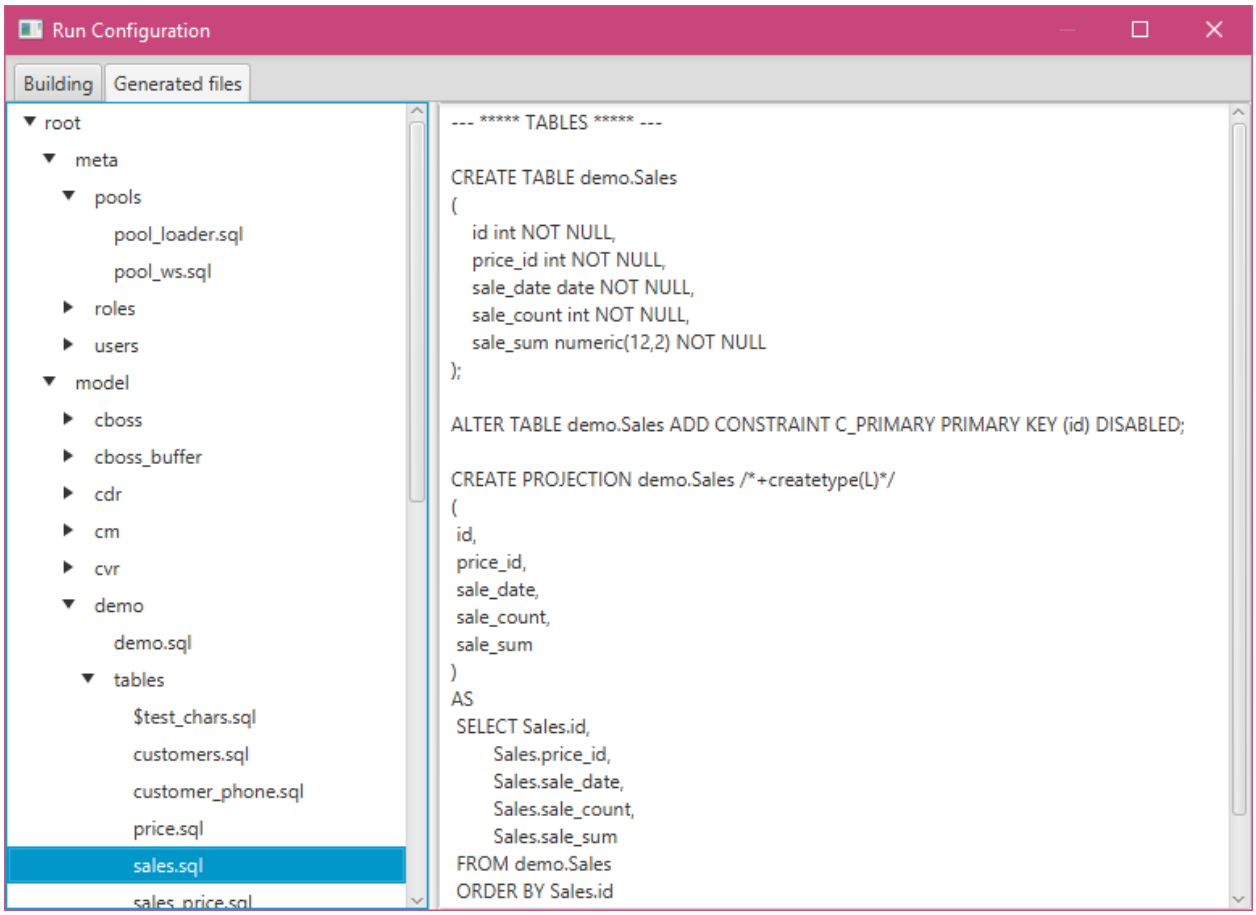


Configuring reverse engineering

To enable reverse engineering, select 'Build' in configuration menu. Specify directory where objects' scripts are to be generated and click 'Start'. You will be asked to delete any files existing in this directory. Work progress log will be displayed while the task is in progress:

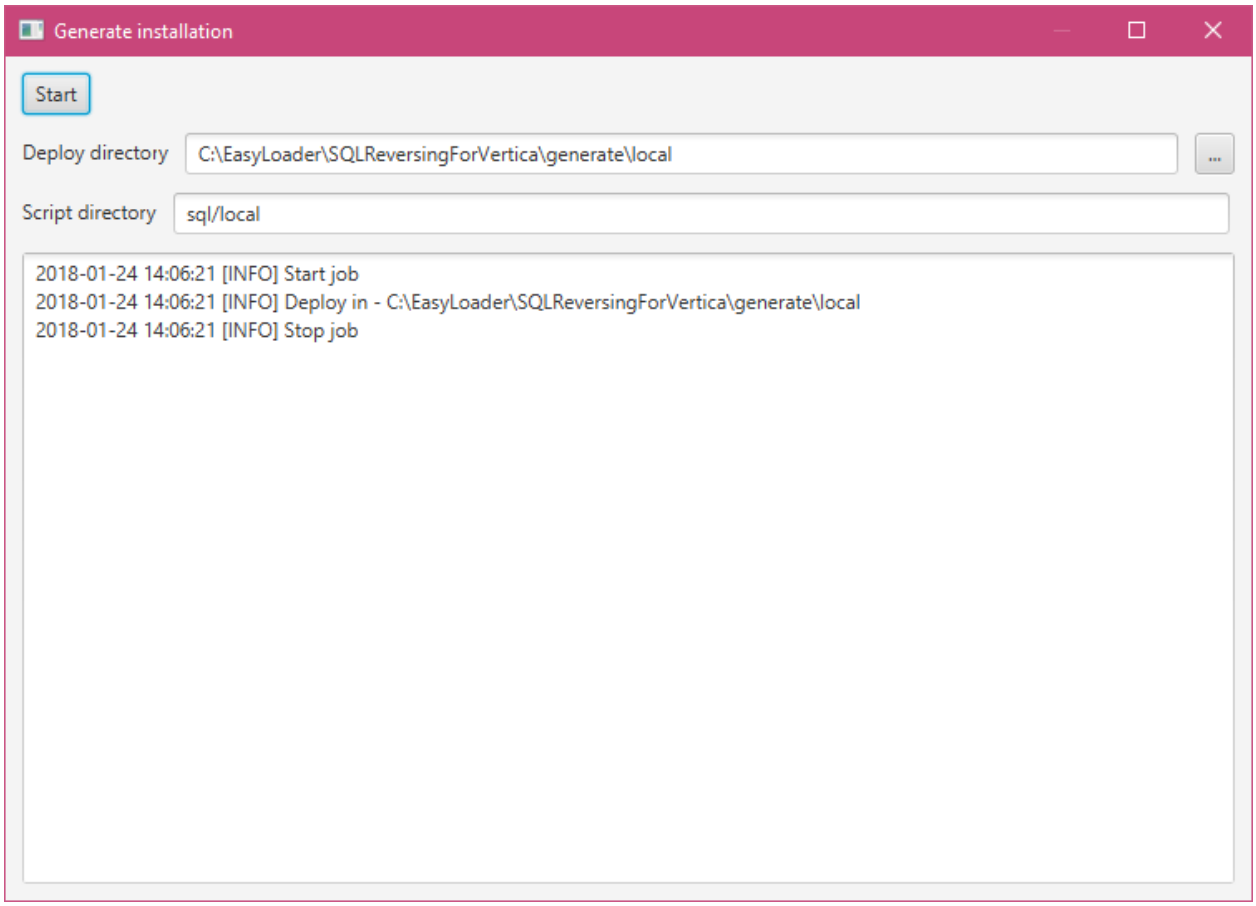


Files can be viewed in 'Generated files' tab or OS file manager once the task is finished:



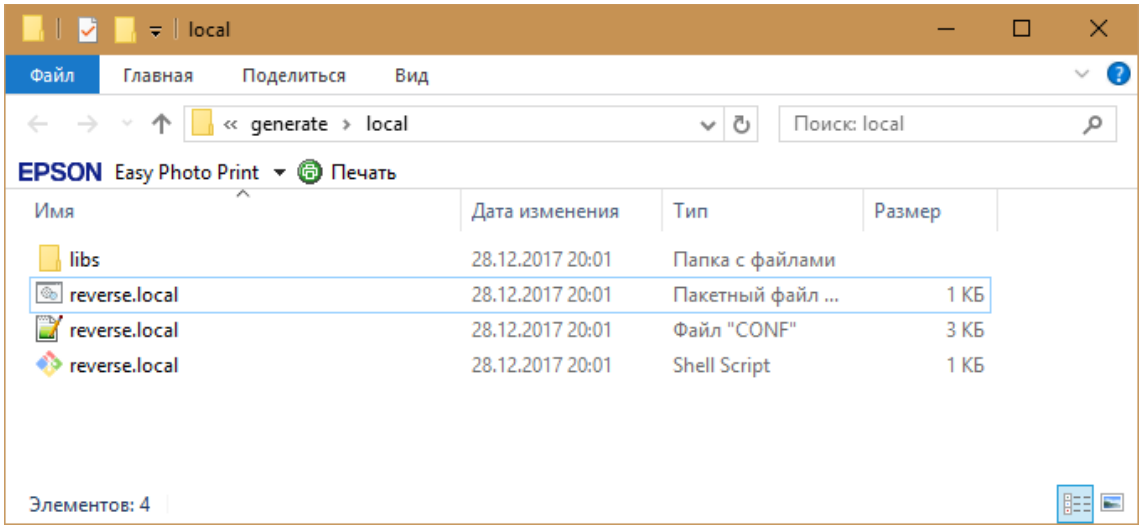
Generating installation for command-line launching

Once customization of configuration required for reverse engineering has been completed, you can automate the process by generating a separate installation to run as scheduled from the command line. To do so select 'Deploy' from configuration menu:



If no script generation path is set, SQL files will be generated in the sql directory that will be created in the current configuration directory.

Once installation has been generated, utility startup file and configuration file will appear in the specified directory:



Should custom value for certain parameters be required during startup, this can be done using command line arguments, e.g.

```
reverse.local.bat vars.host="192.168.102.42" vars.sequence=true
```

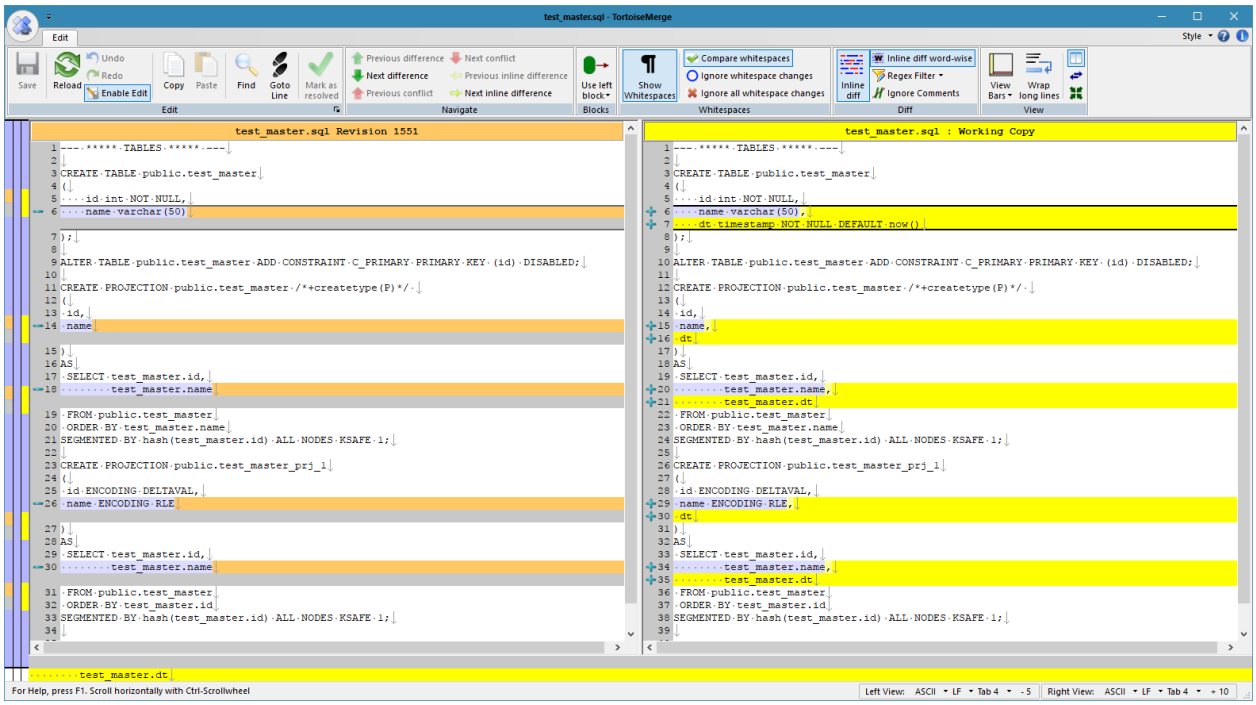
Example of working with SVN

To ensure automatic saving of changed DB objects in SVN, a batch file can be created as below:

```
SET svn="C:\Program Files\TortoiseSVN\bin\svn"
SET scripts="<script path>"
"%svn%" update --username=<user> --password=<psw> "%scripts%"
CALL reverse.local.bat
"%svn%" status -u "%scripts%"
"%svn%" add "%scripts%" -force
"%svn%" commit --username=<user> --password=<pdw> --message "Auto commit" "%scripts%"
```

Now at every DB objects' script generation invoking, SVN will compare and index changes, saving them into repository.

Changes can be viewed with SVN:



In the same manner you can configure script change history in repository of other file-based version control systems.

Copyright and intellectual property rights

All copyrights and intellectual property rights belong to EasyData. The software is distributed for free under GNL 3.0 license.

Download utility: http://easydata.ru/download/reversing/SQLReversingForVertica.zip

Source codes can be found at:
https://github.com/ascrus/verticareverseutilite

Operation of Vertica DB and reverse engineering is ensured by GETL framework by EasyData. You can use this ETL separately or as part of your software under GNL 3.0 license. GETL source codes can be found at:
https://github.com/ascrus/getl

Post your comments and proposals in GITHUB under the respective project.

Autors: Sergey Semykin and Aleksey Konstantinov