Database Migration Wizard to Digital Q.DataBase

User Guide

The Database Migration Wizard to Digital Q.DataBase, developed by Diasoft, is designed to migrate databases from various DBMSs (including MS SQL and Oracle Database) to the Digital Q.DataBase.

As a result of the migration, a database identical to the original one will be created in Digital Q.DataBase. It will include all tables, views, procedures, triggers, and other elements of the source database, adapted to work under the Digital Q.DataBase DBMS.

The Database Migration Wizard to Digital Q.DataBase is available for Ubuntu 22.04, Ubuntu 24.04, and Microsoft Windows 10, 11 in two variants:

- desktop version;
- console version.

Installation on Linux OS

1. On the workstation, enable the standard package repository.
2. Install the package, for example:

```
sudo apt install ./qdatabaseclient64xid-17-17.4.25112401-ubuntu22_x86_64.deb
```

If the package has already been installed, remove it first:

```
sudo apt remove qdatabaseclient64xid-17
```

After the process is complete, all required files will be installed to `/opt/qdb/bin/db_migrator`; in the desktop version of the OS, new shortcuts will appear in the Start menu (Fig. 1):

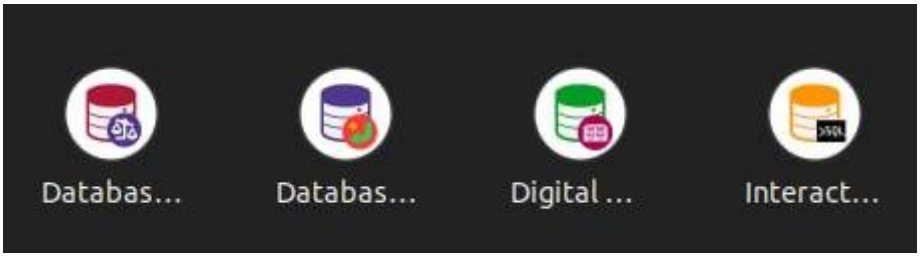

Fig. 1. Startup shortcuts

Installation on Windows

1. Run the .exe file and click “Next” (Fig. 2):

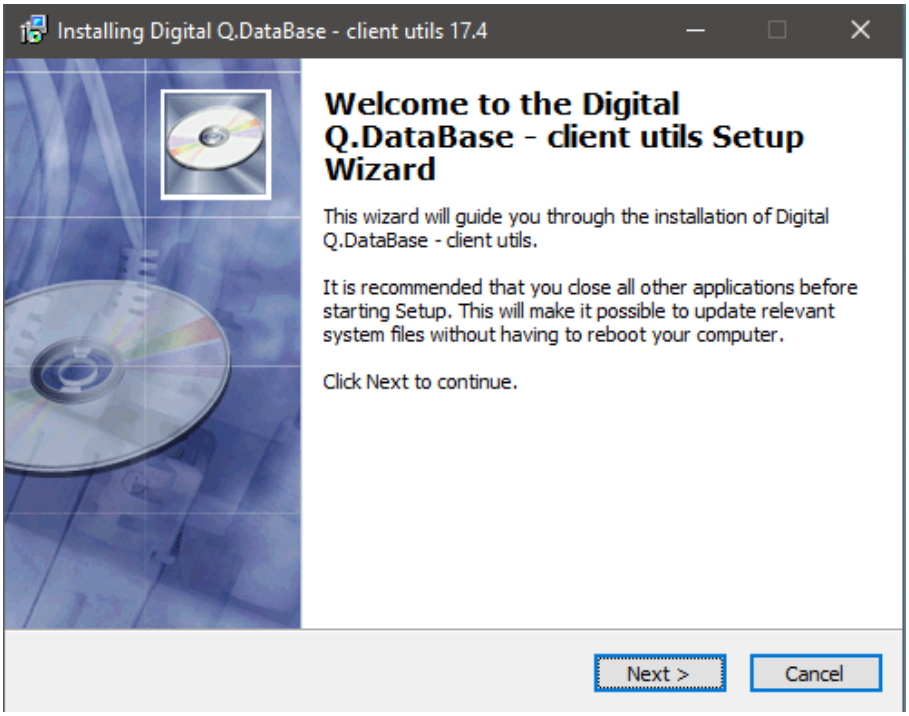

Fig. 2. Setup Wizard: Welcome

2. Select the installation folder and click “Next” (Fig. 3):

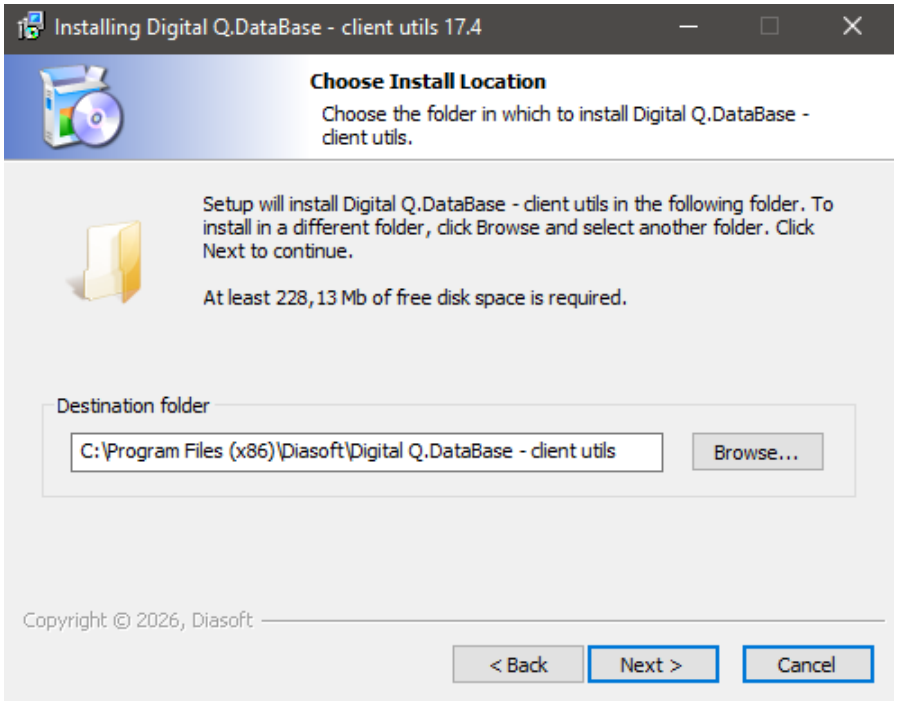

Fig. 3. Setup Wizard: Installation folder

3. Select additional shortcuts and click “Next” (Fig. 4):

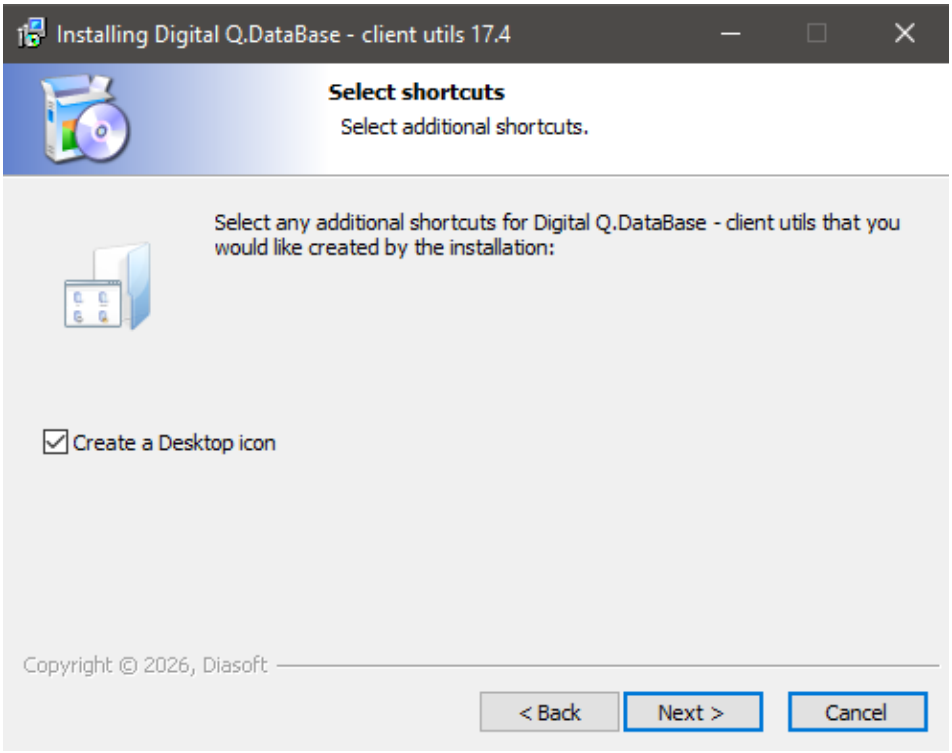

Fig. 4. Setup Wizard: Shortcuts

4. Review the installation settings and click “Install” (Fig. 5):

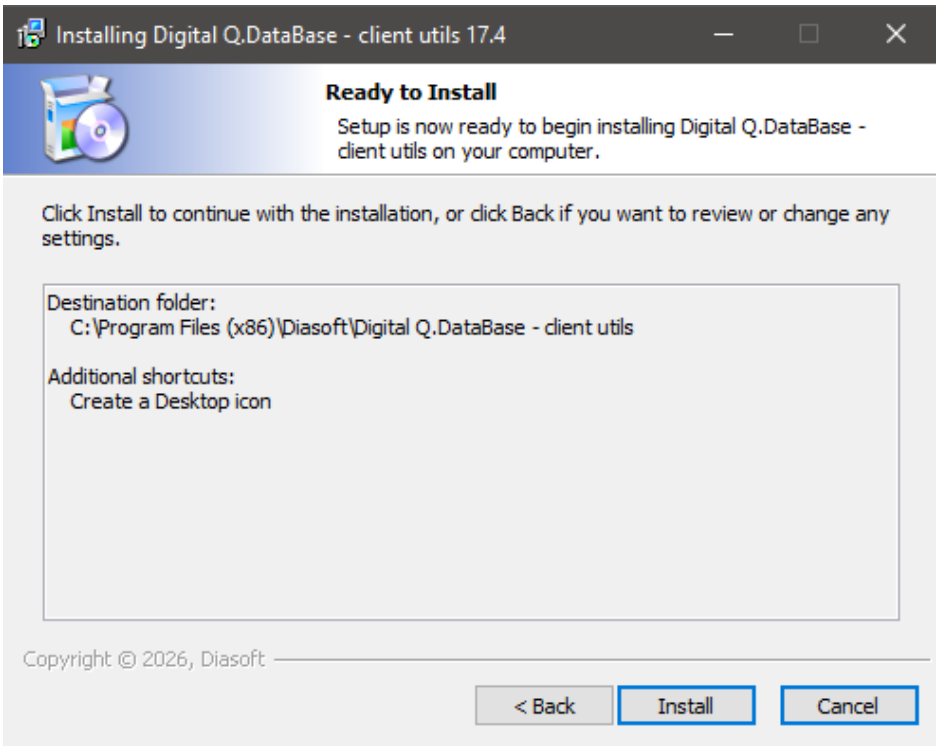

Fig. 5. Setup Wizard: Review settings

5. Wait for the installation to finish and click “Finish” (Fig. 7):

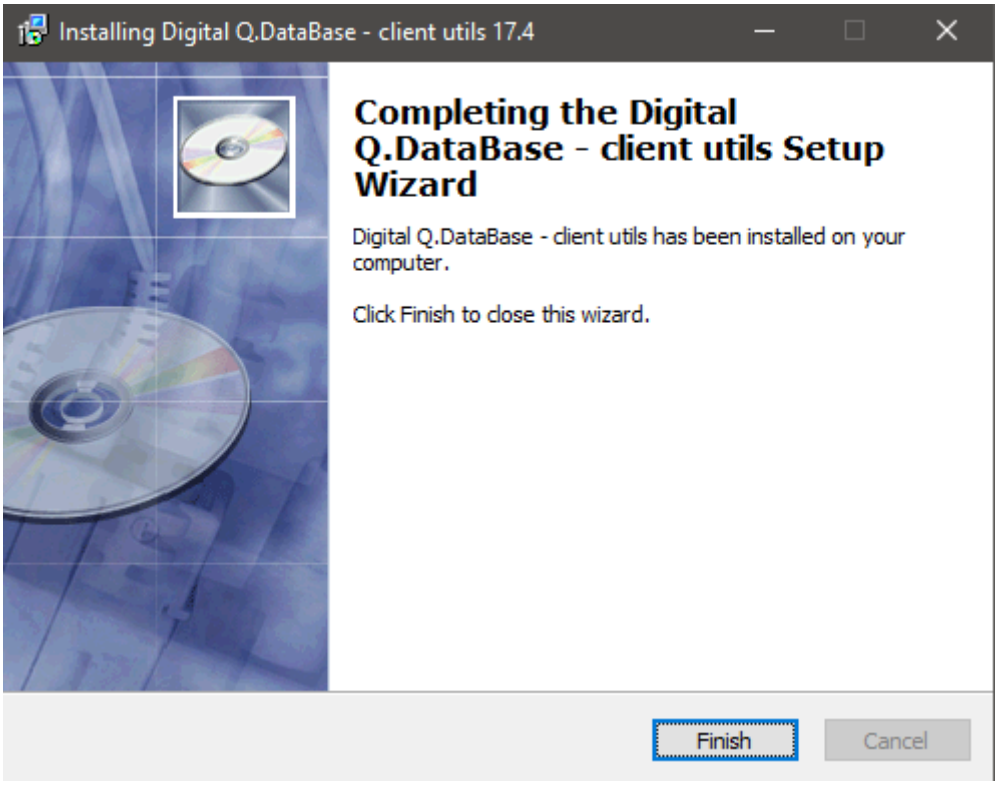

Fig. 6. Setup Wizard: Completion

After completion, the corresponding shortcuts will be created in the Start menu and on the desktop (Fig. 7):

Fig. 7. Installation result

Working with the desktop application

1. Launch the Database Migration Wizard to Digital Q.DataBase using the shortcut and click “Next” (Fig. 8):

Fig. 8. Welcome window

2. Select DBMS type where the source database is located and click “Next”.
   - Microsoft SQL Server;
   - Oracle Database;
   - PostgreSQL.

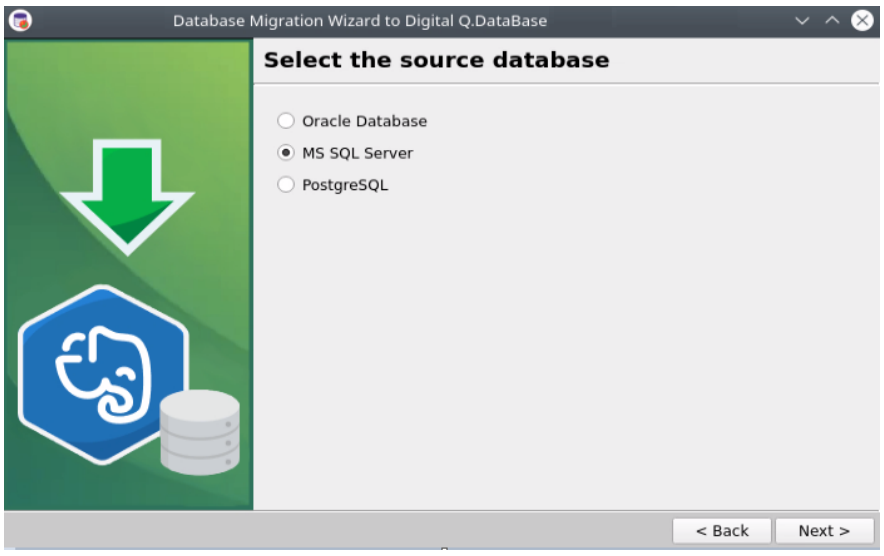

Fig. 9. Selecting the source DBMS

Migrating a database from MS SQL to Digital Q.DataBase

1. Select the data export method (Fig. 10):
   - Full export by a superuser
   - Full export of the dbo schema
   - Custom configuration

   Since the first two migration options are encompassed by the third option, we will consider the custom configuration.

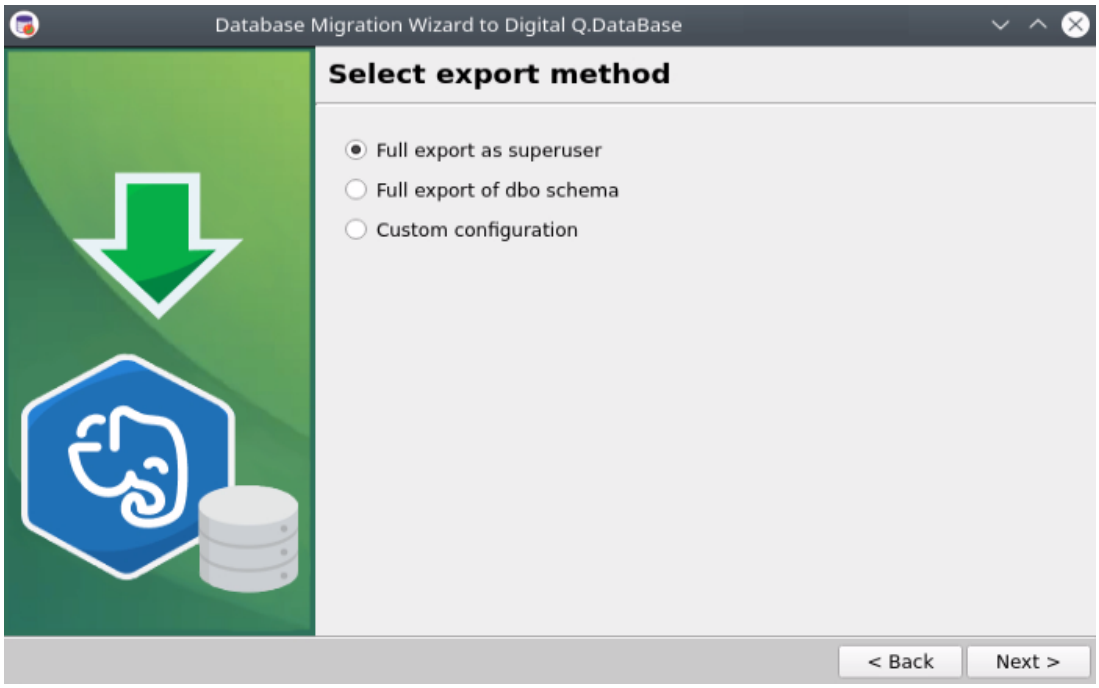

Fig. 10. Data export method

2. Specify the connection parameters for the database to be migrated (Fig. 11):

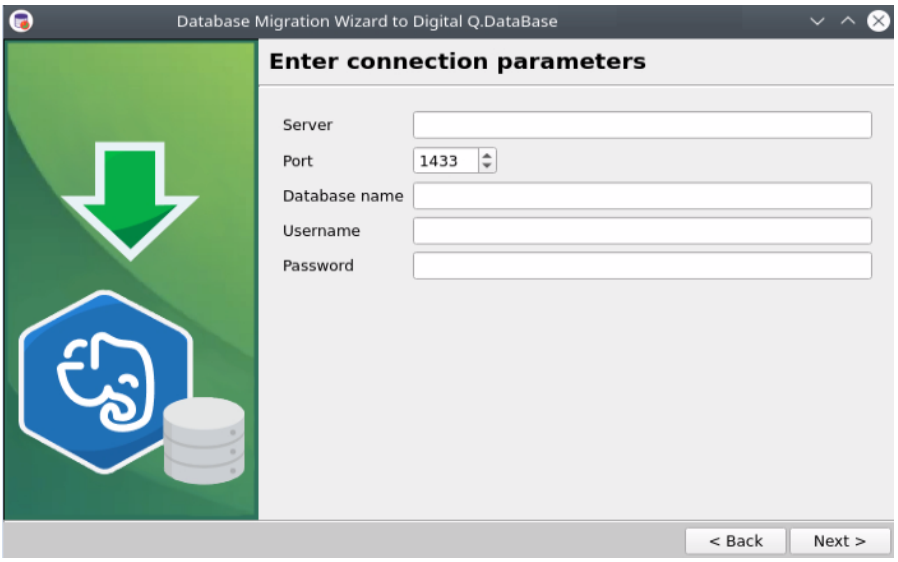

Fig. 11. Connection parameters for the source database

3. Specify where to migrate the data and click “Next” (Fig. 12):
    - To a new database;
    - To an existing database (when part of the database has already been migrated);
    - Create an SQL file for manual loading of the data (specify where to place the file with the exported data).

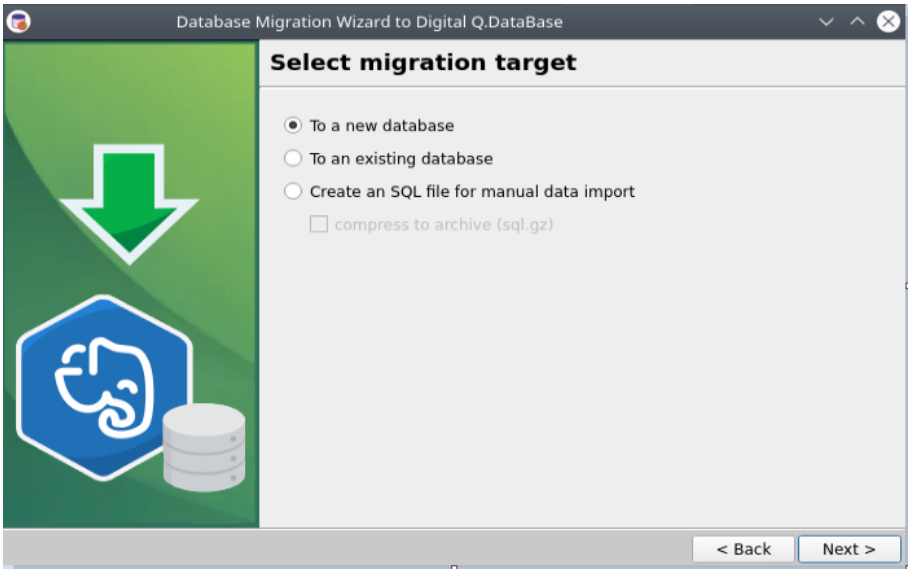

Fig. 12. “Where to migrate the data” window

4. Specify the connection parameters for the database (Digital Q.DataBase) to which the selected objects will be migrated, and click “Next” (Fig. 13):

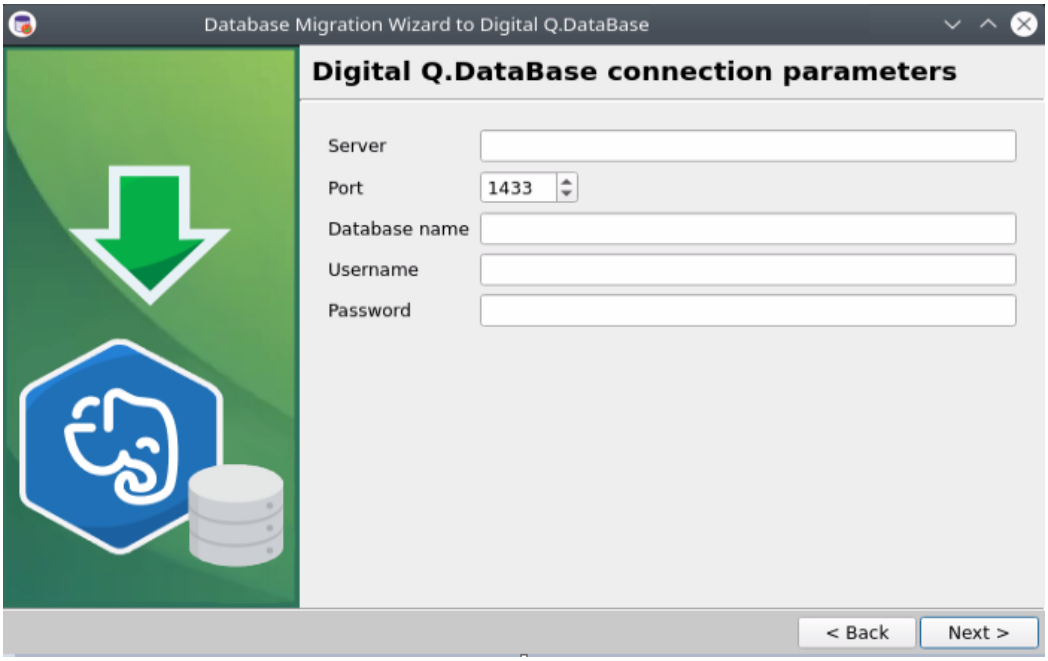

Fig. 13. Connection parameters for the target database

5. Select/deselect the required database objects to be migrated (Fig. 14):

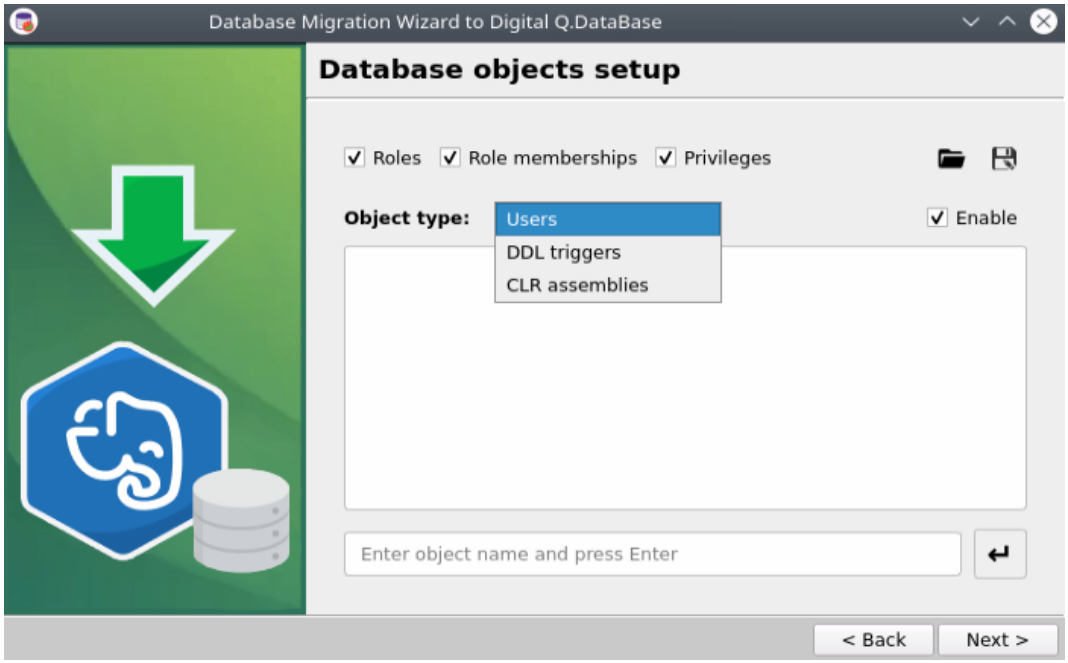

Fig. 14. Database object configuration

6. Specify the schemas to be migrated using the “Add” button. After a schema is added successfully, the following buttons will be available (Fig. 15):
    - Edit schema
    - Configure migration mode, where you can optionally select the database objects to migrate (right pane in Fig. 15)
    - Delete schema

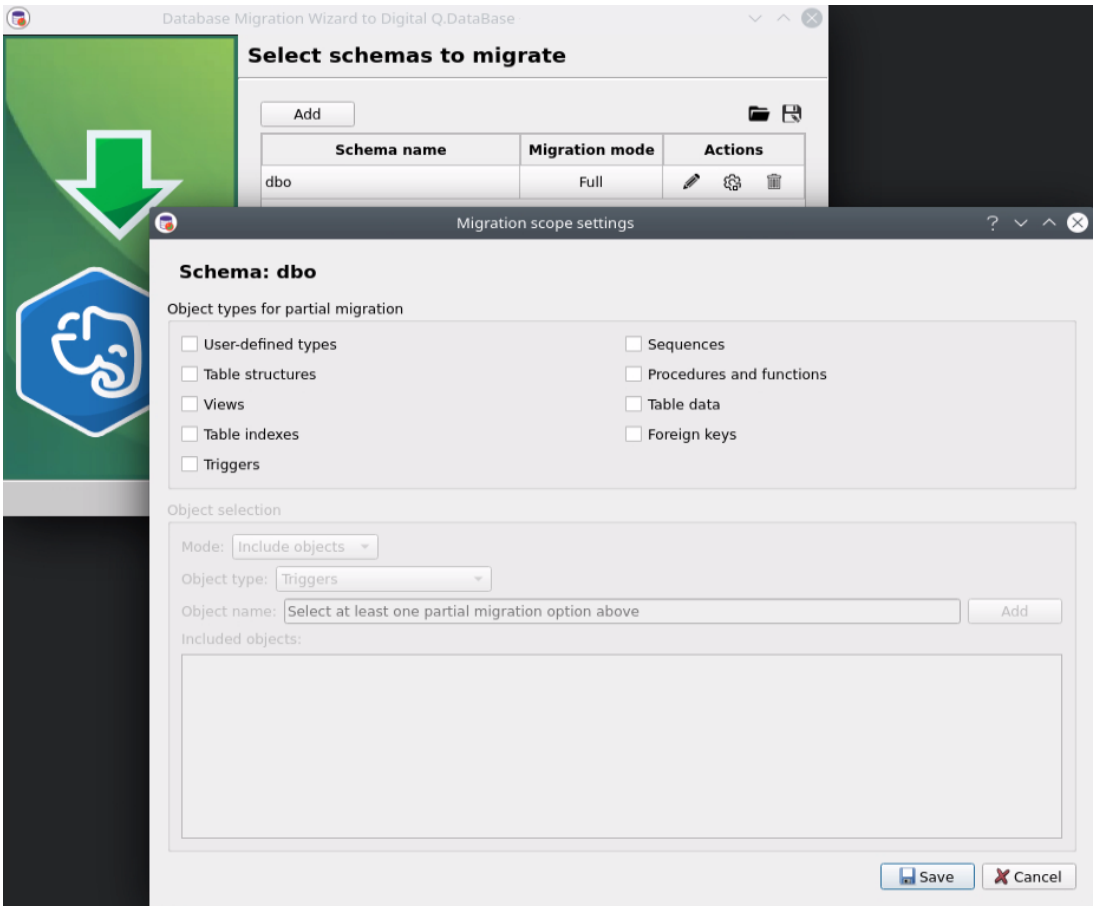

Fig. 15. Specifying schemas for migration and configuring migration mode

7. Review all specified parameters and click “Next” (Fig. 16):

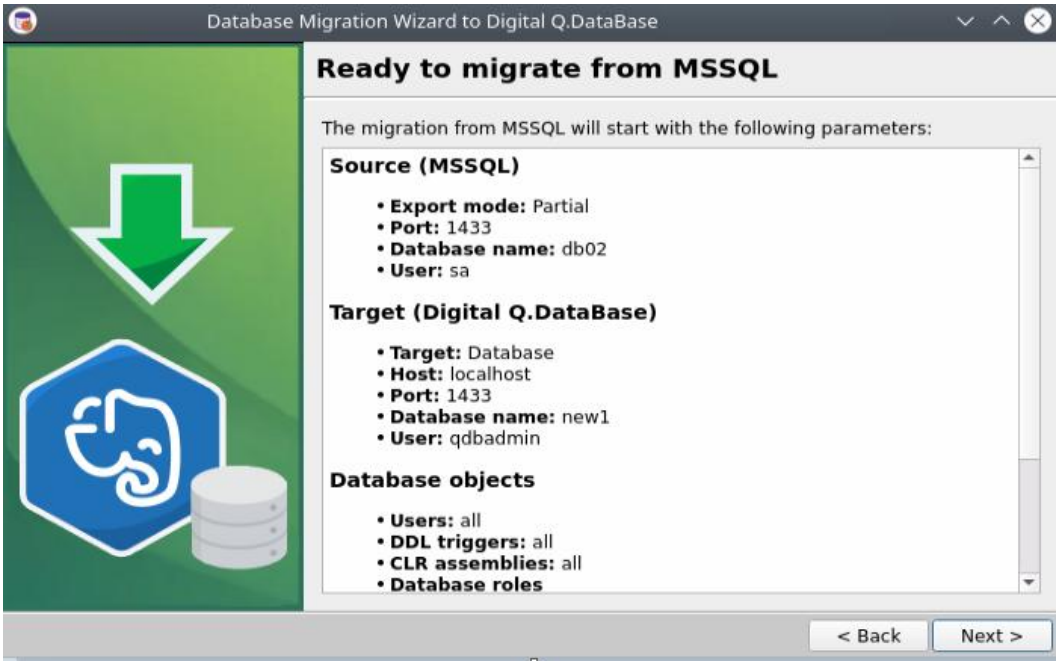

Fig. 16. Reviewing the specified parameters

8. Wait for the data migration to complete successfully and click “Finish” (Fig. 17):

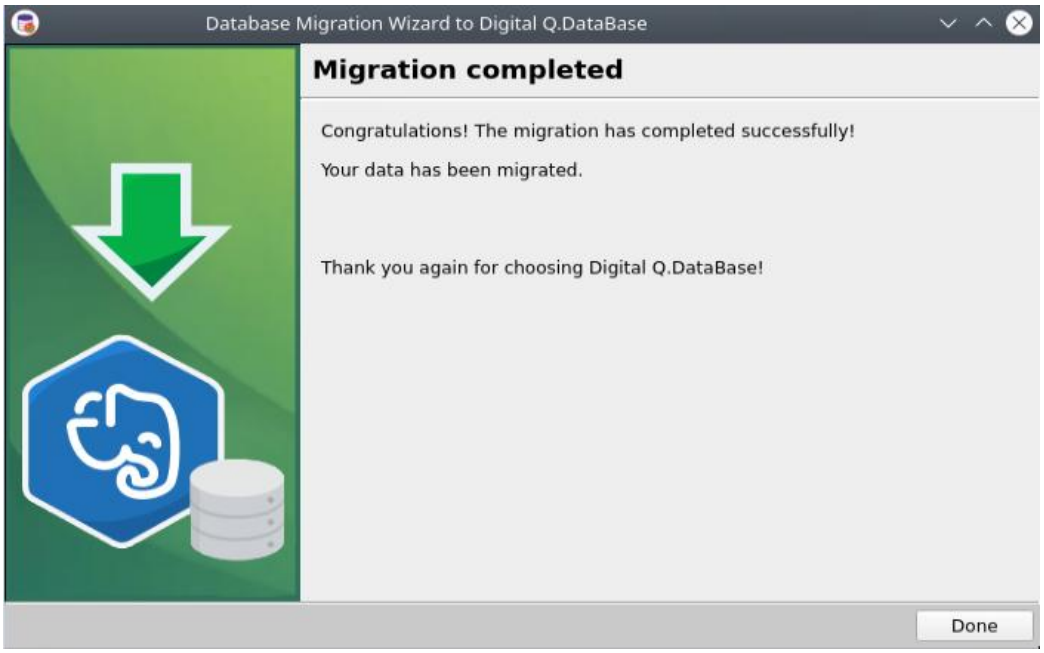

Fig. 17. Successful migration

If migration errors occur, a log file with detailed error information will be generated.

Migrating a database from Oracle to Digital Q.Database

1. Select the data export method (Fig. 18):
    - Full export using a superuser (SYS)
    - Custom configuration
    - Since the first migration option is encompassed by the second option, we will consider the custom configuration.

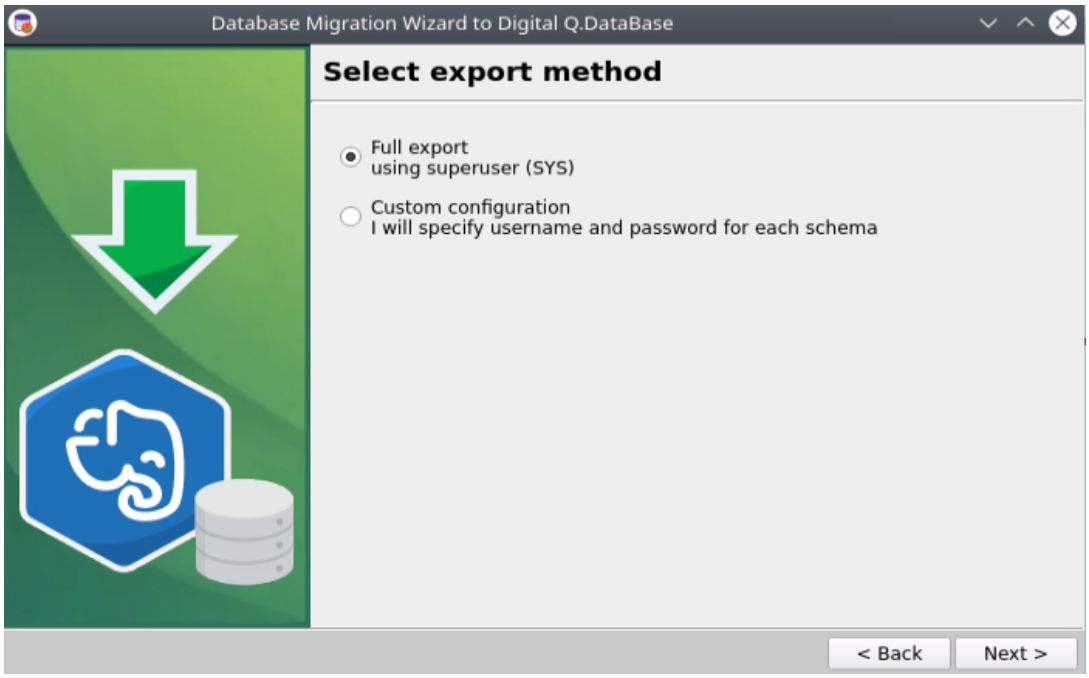

Fig. 18. Data export method

2. Specify the connection parameters for the database to be migrated (Fig. 19):

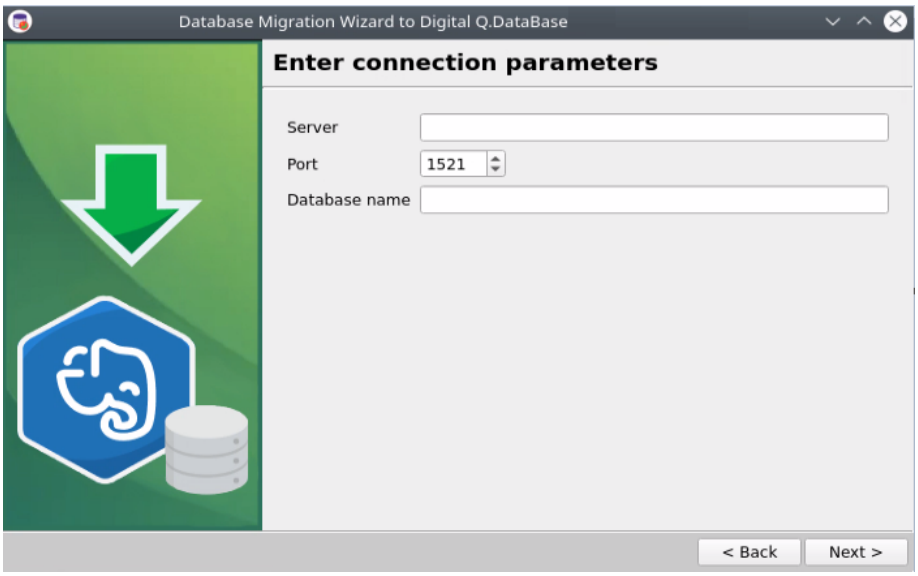

Fig. 19. Connection parameters for the source database

3. Specify where to migrate the data and click “Next” (Fig. 20):
    - To a new database;
    - To an existing database (when part of the database has already been migrated);
    - Create an SQL file for manual loading of the data (specify where to place the file with the exported data).

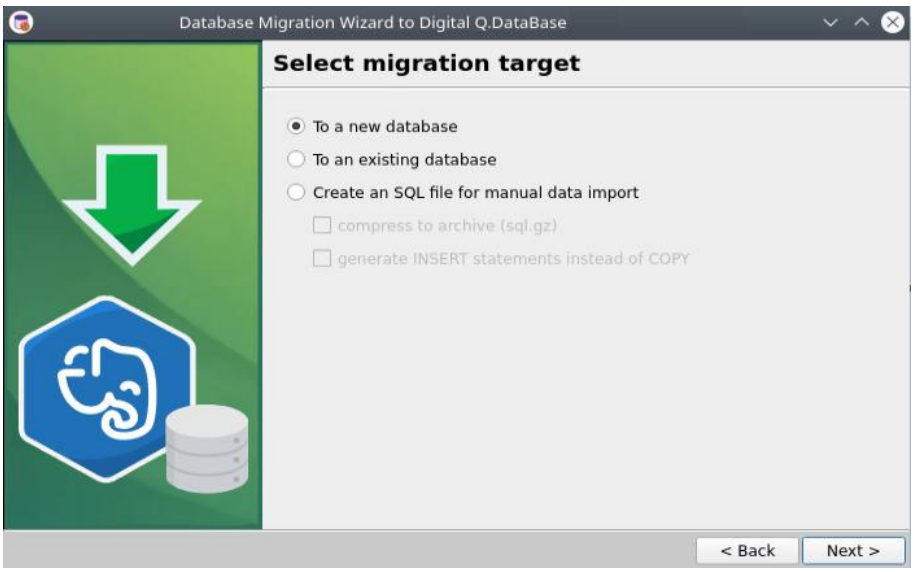

Fig. 20. “Where to migrate the data” window

4. Specify the connection parameters for the database (Digital Q.DataBase) to which the selected objects will be migrated, and click “Next” (Fig. 21):

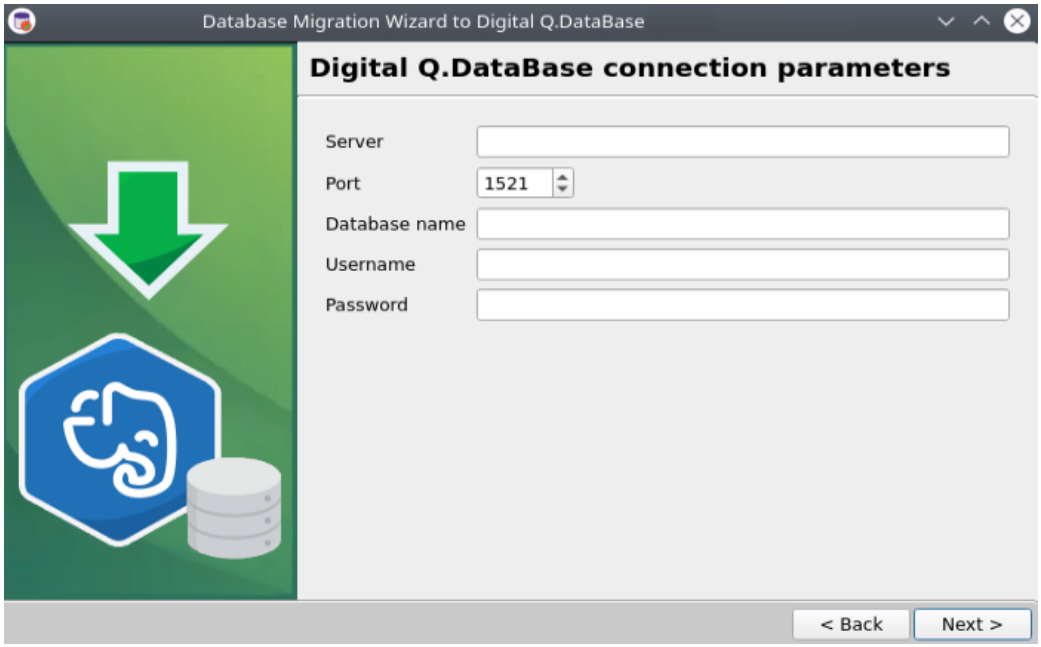

Fig. 21. Connection parameters for the source database

5. Specify the schemas to be migrated using the “Add” button. After a schema is added successfully, the following buttons will be available (Fig. 22):
    - Edit schema
    - Configure migration mode, where you can optionally select the database objects to migrate
    - Delete schema

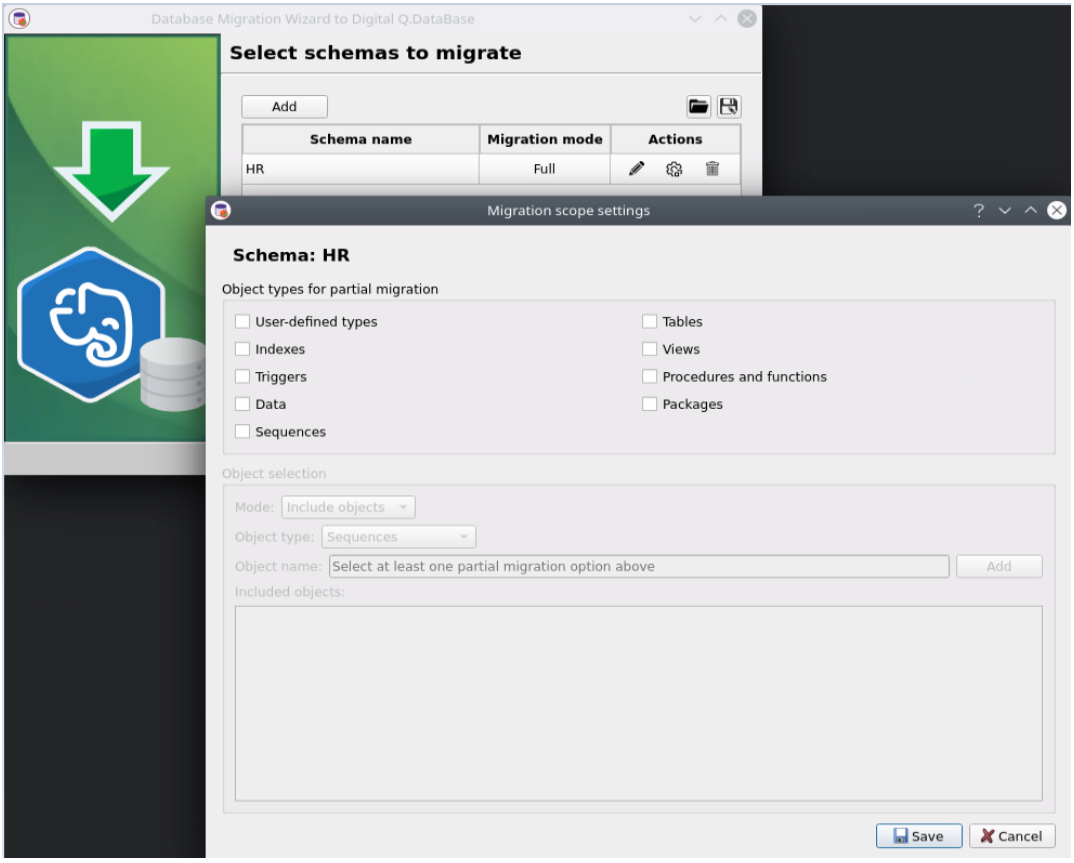

Fig. 22. Specifying schemas for migration and configuring migration mode

6. Review all specified parameters and click “Next”
7. Wait for the data migration to complete successfully and click “Finish”

If migration errors occur, a log file with detailed error information will be generated.

Migrating a database from PostgreSQL to Digital Q.DataBase

1. Specify the connection parameters for the database to be migrated (Fig. 23):

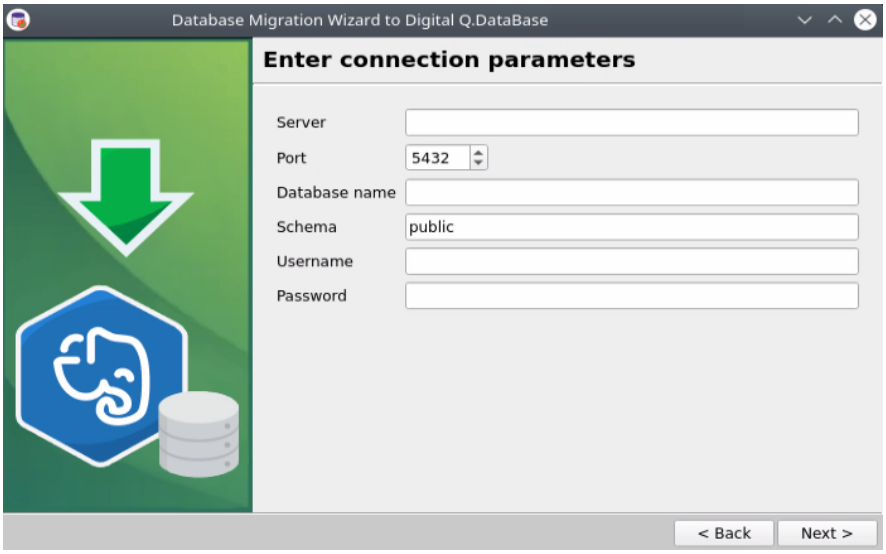

Fig. 23. Connection parameters for the source database

2. Select what to migrate (Fig. 24):
    - All contents of the database;
    - Structure only;
    - Data only (if the structure has already been migrated previously);
    - Selective.

Fig. 24. Selecting object types for migration

3. Select groups of objects to migrate (optional) (Fig. 25):

- Table structure;
- Indexes and constraints (check conditions, selection rules);
- Views;
- Server-side logic (stored procedures, triggers, packages, functions);
- Data;
- Information about users and their privileges.

Additionally, you can select specific objects to migrate (optional).

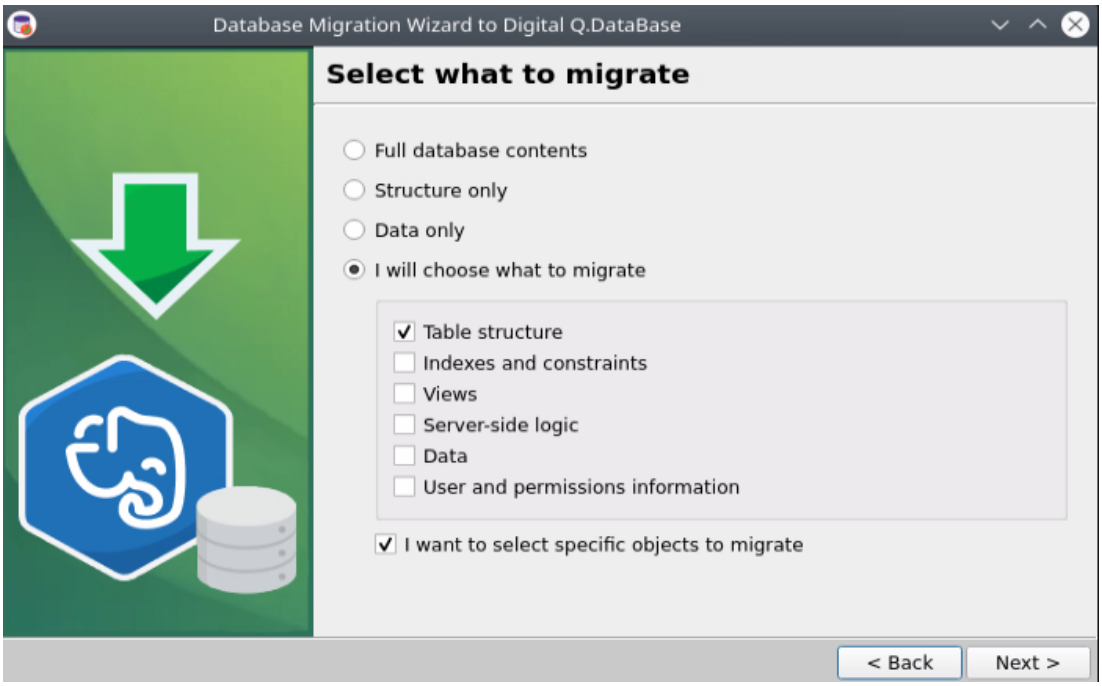

Fig. 25. “Select what to migrate” window

4. To select specific object names, lists are provided with sorting by name and size, as well as options to include or exclude the specified items from the migration process (Fig. 26):

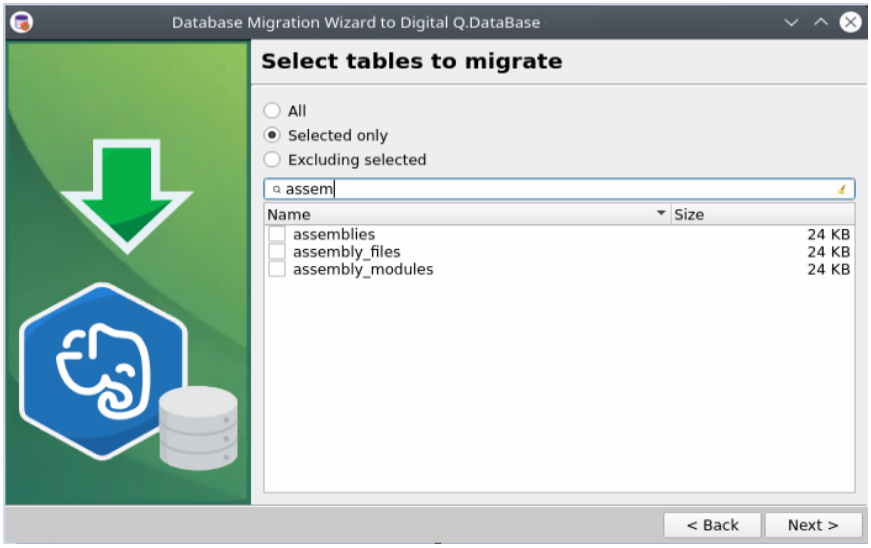

Fig. 26. Selecting object names for migration

5. Specify where to migrate the data and click “Next” (Fig. 27):
    - To a new database;
    - To an existing database (when part of the database has already been migrated);
    - Create an SQL file for manual loading of the data (specify where to place the file with the exported data).

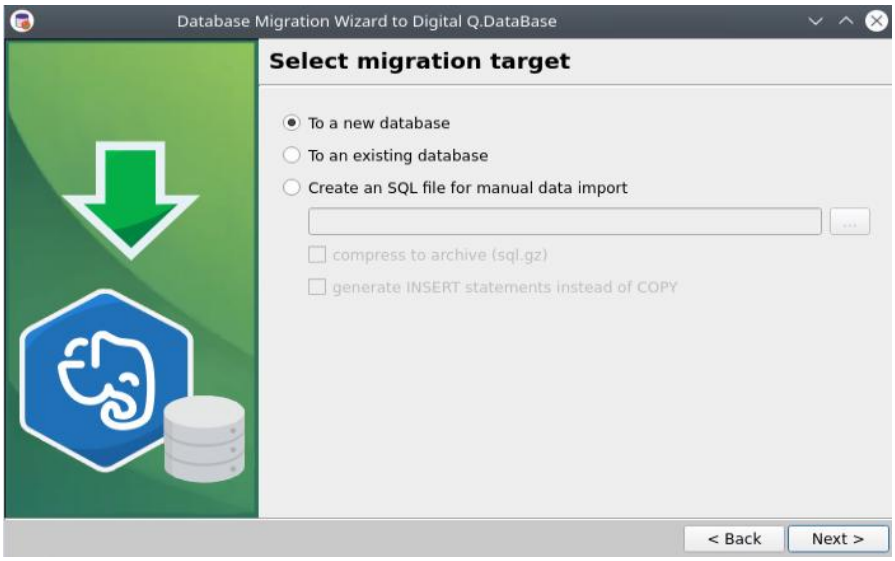

Fig. 27. “Where to migrate the data” window

6. Specify the connection parameters for the database (Digital Q.DataBase) to which the selected objects will be migrated, and click “Next” (Fig. 28):

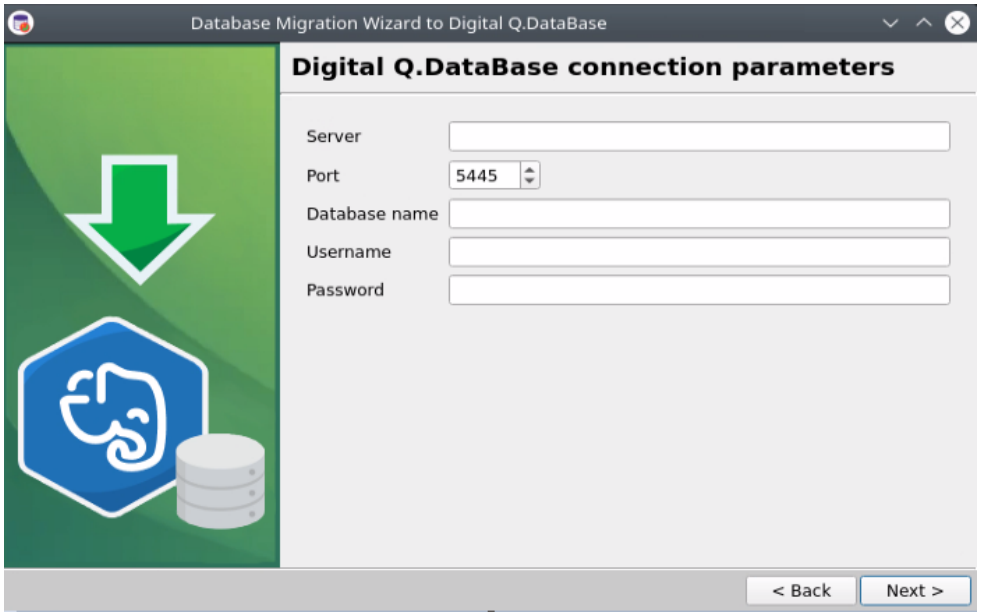

Fig. 28. Connection parameters for the target database

7. Review all specified parameters and click “Next”
8. Wait for the data migration to complete successfully and click “Finish”. If migration errors occur, a log file with detailed error information will be generated.

Working with the console application

Preliminary steps

The following steps can be simplified by using templates from /opt/qdb/config/db_migrator_templates/*.yaml.

1. Go to /opt/qdb/config/db_migrator_templates.
2. Create a configuration file for connecting to the source database.

2.1. To connect to MS SQL, create an mssql.yaml file with the following parameters:

```
db_type: mssql
```

```
server: [hostName_or_IP)]
port: [1433]
user: [userName]
password: [userPassword]
database: [dbName]
schema: [schemaName]
```

2.2. To connect to Oracle, create an oracle.yaml file with the following parameters:

```
db_type: oracle
server: [hostName_or_IP)]
port: [1521]
user: [userName]
password: [userPassword]
database: [dbName]
schema: [schemaName]
sys: [password]
```

2.3. To connect to Postgres, create a pg.yaml file with the following parameters:

```
db_type: postgres
server: [hostName_or_IP]
port: [5432]
user: [userName]
password: [userPassword]
database: [dbName]
schema: [schemaName]
```

3. Create a configuration file for connecting to the target database.

3.1. To connect to Digital Q.DataBase when migrating MS SQL, create the target DB connection configuration file mssql_qdb.yaml with the following parameters (the database to which the migration will be performed):

```
db_type: mssql
server: [hostName_or_IP]
port: [port]
user: [userName]
password: [userPassword]
database: [dbName]
schema: [schemaName]
```

3.2. To connect to Digital Q.DataBase when migrating Oracle Database, create the target DB connection configuration file oracle_qdb.yaml with the following parameters (the database to which the migration will be performed):

```
db_type: postgres
server: [hostName_or_IP]
port: [1521]
user: SYSDBA
password: [userPassword]
database: [dbName]
schema: [schemaName]
```

3.3. To connect to Digital Q.DataBase when migrating PostgreSQL, create the target DB connection configuration file pg_qdb.yaml with the following parameters (the database to which the migration will be performed):

```
db_type: postgres
server: [hostName_or_IP]
port: [5445]
user: qdbadmin
```

```
password: [userPassword]
database: [dbName]
schema: [schemaName]
```

4. To work with the ring migrator (Oracle Database), you need to create the oracle_ctl.yaml file

```
type: oracle
source_host: [hostName_or_IP_source_base]
source_port: [1521]
target_host: [hostName_or_IP_target_base]
target_port: [1521]
target_user: [SYSDBA]
target_password: [userPassword]
mode: [custom / all]
db: [dbName]
processing_dir: ~/.qdbmigr

# Specific Oracle params
[login_sys_user: password_sys_user]
schemas:
  - name: [schemaName1]
    password: [userPassword1]
    custom:
      user_defined_types:
        enabled: [true / false]
```

```
    tables:
      enabled: [true / false]
      list:
        - "Table1"
        - " Table2"
      exclude: [true / false]
    indexes:
      enabled: [true / false]
    views:
      enabled: [true / false]
    triggers:
      enabled: [true / false]
    procedures:
      enabled: [true / false]
      list:
    data:
      enabled: [true / false]
      list:
        - " Table1"
        - " Table2"
      exclude: [true | false]
- name: [schemaName2]
  password: [userPassword2]
```

5. To work with the ring migrator (MSSQL Database), you need to create the mssql_ctl.yaml file

```
type: mssql

source_host: [hostName_or_IP_source_base]
source_port: [1433]
source_database: [DatabaseSourceName]
source_user: [SourceUserName]
source_password: [SourceUserPassword]
target_host: [hostName_or_IP_target_base]
target_port: [1433]
target_database: [DatabaseTargetName]
target_user: [TargetUserName]
target_password: [TargetUserPassword]
mode: custom
processing_dir: ~/.qdbmigr

database_objects:
  users:
    enabled: [true / false]
  database_roles:
    enabled: [true / false]
  role_memberships:
    enabled: [true / false]
  privileges:
    enabled: [true / false]
  ddl_triggers:
    enabled: [true / false]
  clr_assembly:
    enabled: [true / false]



schemas:
  - name: dbo
    custom:
```

```
        user_defined_types:
          enabled: [true / false]
        sequences:
          enabled: [true / false]
        tables:
          enabled: [true / false]
          list:
            - "Table1"
            - "Table2"
          exclude: [true / false]
        procedures:
          enabled: [true / false]
        views:
          enabled: [true / false]
        data:
          enabled: [true / false]
          list:
            - "Table1"
            - "Table2"
          exclude: [true / false]
        indexes:
          enabled: [true / false]
        foreign_keys:
          enabled: [true / false]
        triggers:
          enabled: [true / false]
    - name: schema
```

6. Create the cs.yaml file (settings for partial migration):

```
users:
  enabled: [true/false]
user_defined_types:
  enabled: [true/false]
tables:
```

```
  enabled: [true/false]
  list:
     - "[tablename1]"
     - "[tablename2]"
  exclude: [true/false]
procedures:
  enabled: [true/false]
  list:
     - "[tablename1]"
     - "[tablename2]"
  exclude: [true/false]
views:
  enabled: [true/false]
triggers:
  enabled: [true/false]
ddl_triggers:
  list:
     - "[tablename1]"
     - "[tablename2]"
  exclude: [true/false]
data:
  enabled: [true/false]
  list:
     - "[tablename1]"
     - "[tablename2]"
```

```
    exclude: [true/false]
  indexes:
    enabled: [true/false]
  foreign_keys:
    enabled: [true/false]
```

The enabled: [true/false] value controls migration of the corresponding object group (true - migrate, false - do not migrate).

The list parameter is used to specify specific object names (in this example, a list of tables is provided for which data will or will not be migrated - depending on the exclude parameter).

The exclude: [true/false] value controls inclusion or exclusion of a specific list of objects in the migration process (true - exclude the specified list of objects, false - migrate only the specified objects of the current object type).

The list and exclude parameters can be applied to all object types.

Main parameters

-config-source	Path to the source configuration	File path
-source-check	Check connectivity to the source	Flag
-list-tables	View the list of tables	Flag
-list-procedures	View the list of procedures	Flag
-list-views	View the list of views	Flag
-config-target	Path to the QDB configuration	File path
-scope	Migration scope	all/struct/data/custom
-object-file	Path to the partial migration configuration	File path
-target-action	Target DB action	new/existing/file
-target-file	Path to save the SQL file (required for -target-action file)	File path
-data-format	Data export format	insert/copy
-target-compress	Output file compression format	gzip/bzip2
-processing-dir	User directory for temporary migration files	Directory path

Multithreading parameters

Parameter	Description	Default values

-disable-parallel	Disable multithreading	false

Detailed parameter description:

```
Required parameters:
    -config-source
Path to the source configuration file

    -target-action
Action to perform on the target DB (new|existing|file)
new    - create a new database
existing - use an existing database
file   - save to a file

    -scope
        Migration scope (all|struct|data|custom)
        all    - structure and data
        struct  - structure only
        data   - data only
        custom - user-defined


Required parameter ONLY for the ring migrator (Oracle):
      -config-ctl


Additional parameters:
    -source-check
Check connectivity to the source

    -list-tables
Print the list of available tables

    -list-procedures
Print the list of available procedures
```

```
    -list-views
Print the list of available views


    -config-target
Path to the QDB configuration file (required for -target-action
new/existing)

    -target-file
Path to save the SQL file (required for -target-action file)

    -data-format
Data export format (insert|copy) (default: "copy")
Applies only when -target-action file is used

    -target-compress
Output file compression format (gzip|bzip2)
Applies only when -target-action file is used

    -object-file
Path to the migration object configuration file (used with -
scope=custom)

    -processing-dir
User directory for temporary migration files
```

Running the console migrator

Run the executable from /opt/qdb/bin/db_migrator/:

```
./mssql_migrator [params]

./oracle_migrator [params]
```

```
./pg_migrator [params]

./oracle_ctl [params]

./mssql_ctl [params]
```

Calls to the console migrators oracle_migrator and pg_migrator are similar to the mssql_migrator call; likewise, calling oracle_ctl is similar to mssql_ctl. Therefore, only mssql_migrator and oracle_ctl are described below.

Migration file storage

Temporary migration files are stored in the directory:

```
~/.qdbmigr/migration_sql_queries/YYYY-MM-DD_HH-MM-SS_ИмяМигрируемойБазыДанных/
```

For quick access to the most recent migration, a symbolic link is created:

```
~/.qdbmigr/latest
```

Usage examples

1. Checking database connectivity:

```
./mssql_migrator -config-source ../config/
db_migrator_templates/mssql.yaml -source-check
```

This command checks connectivity to the database specified in the mssql.yaml file (the database from which the migration will be performed).

```
./mssql_migrator -config-source ../config/
db_migrator_templates/mssql_qdb.yaml -source-check
```

This command checks connectivity to the database specified in the qdb.yaml file (the database to which the migration will be performed).

2. Printing the list of available tables:

```
./mssql_migrator -config-source ../config/
db_migrator_templates/mssql.yaml -list-tables
```

This command prints the tables located in the database specified in the mssql.yaml file.

3. Full migration to a new database:

```
./mssql_migrator -config-source ../config/
db_migrator_templates/mssql.yaml -config-target ../config/
db_migrator_templates/mssql_qdb.yaml -scope all -target-
action new
```

This command performs a full migration to a new database, migrating the structure and data from the database specified in mssql.yaml to the database specified by qdb.yaml.

4. Full migration to an existing database:

```
./mssql_migrator -config-source ../config/
db_migrator_templates/mssql.yaml -config-target ../config/
db_migrator_templates/mssql_qdb.yaml -scope all -target-action
existing
```

This command performs a full migration to an existing database, migrating the structure and data from the database.

5. Full migration: export data to a file (insert|copy)

```
./mssql_migrator -config-source ../config/
db_migrator_templates/mssql.yaml -target-action file -target-
file ~/dump_insert.txt  -data-format insert
```

With this command, data will be generated as a group of INSERT statements.

```
./mssql_migrator -config-source ../config/
db_migrator_templates/mssql.yaml -target-action file -target-
file ~/dump_copy.txt -data-format copy
```

With this command, data will be generated as a COPY array.

6. Partial migration to a new database:

```
./mssql_migrator -config-source
../config/db_migrator_templates/mssql.yaml -config-target
```

```
../config/db_migrator_templates/mssql_qdb.yaml -scope custom -
object-file ../config/db_migrator_templates/cs.yaml -target-
action new
```

This command migrates to a new database, transferring the structure and data specified in cs.yaml from the database specified in mssql.yaml to mssql_qdb.yaml.

7. Partial migration to an existing database:

```
./mssql_migrator -config-source
..config/db_migrator_templates/mssql.yaml -config-target
..config/db_migrator_templates/mssql_qdb.yaml -scope custom -
object-file ..config/db_migrator_templates/cs.yaml -target-
action existing
```

This command migrates to an existing database, transferring the structure and data specified in cs.yaml from the database specified in mssql.yaml to qdb.yaml.

8. Partial migration: export data to a file (insert|copy)

```
./mssql_migrator -config-source
..config/db_migrator_templates/mssql.yaml -config-target
..config/db_migrator_templates/mssql_qdb.yaml -scope custom
-object-file ..config/db_migrator_templates/cs.yaml -target-
action file -target-file ~/dump_insert.txt  -data-format
insert
```

With this command, data will be generated as a group of INSERT statements.

```
./mssql_migrator -config-source
..config/db_migrator_templates/mssql.yaml -config-target
..config/db_migrator_templates/mssql_qdb.yaml -scope custom -
object-file ..config/db_migrator_templates/cs.yaml -target-
action file -target-file ~/dump_copy.txt -data-format copy
```

С помощью данной команды данные будут формироваться в формате copy-массива.

1. **Running the ring migrator to a new database (Oracle Database)**

```
./oracle_ctl -config-ctl
../config/db_migrator_templates/oracle_ctl.yaml
```

This command performs a full or partial migration to a new database, migrating the structure and data from the database.

2. **Running the ring migrator to an existing database (Oracle Database)**

```
./oracle_ctl -config-ctl ../config/db_migrator_templates/
oracle_ctl.yaml –target-action existing
```

This command performs a full or partial migration to a new database, migrating the structure and data from the database.

3. **Exporting a database to files with the ring migrator (Oracle Database)**

```
./oracle_ctl -config-ctl ../config/db_migrator_templates/
oracle_ctl.yaml –target-action file
```

With this command, query files will be generated for objects.