Data Import and Export Methods

Martina Durneková, Michal Kvet
University of Žilina
Žilina, Slovakia
{Martina.Durnekova, Michal.Kvet}@fri.uniza.sk

***Abstract*—Nowadays, working with data, data recovery and data transfer between systems is essential part of many processes. Therefore, the ability to know the different types of methods, that allow us to export data from database system and import that data into another or the same system, is important. It is also important to know, when the specific method can be used, and which one is the fastest in time. This paper focuses on import and export techniques and compares the speed of their execution.**

I. Introduction

Data transfer between systems is a common part of working practices nowadays. Especially when a person works with a huge amount of data. In addition to data transfer, it is necessary to think about backups and restoring data in the event of a system error. Each database system offers several methods by which data can be exported and also imported into the system. These methods are applied in many organizations, where huge amount of data is generated and processed. Exported data can be used further in analysis or statistics.

These methods have many advantages, why it is advisable to use them. We do not have to process data manually one by one or in a small group into the database. Carrying a big portion of data in one package is one of the advantages of using export and import methods. Another advantage is that the particular object or table can be backed up and restored later. It is also possible to use the methods to copy objects from one schema to another.

On the other hand, the disadvantage of these methods might be the processing speed. Each of these methods is different. There are methods that use both, the client side, and the server side. Other methods are performed only on the server side, so they do not need additional costs to connect the client side and to establish network communication between the client and server.

The client-server architecture is shown in Fig. 1. The client and server are connected via network. The client requests and receives information from the server. The server processes and executes the requirements from client and sends the result to the client device.

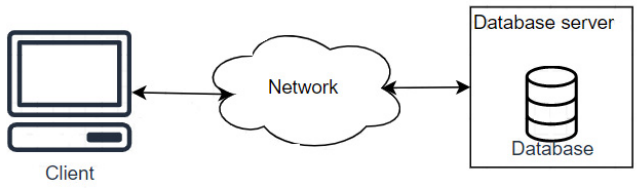

Fig. 1. Client - server architecture

The first and second section of the paper will be focused on actual state and existing methods for importing and exporting data. In these sections, the import and export methods will be described in more detail.

The third part of the paper will be focused on description of the experiments which were performed on different data in terms of size. Evaluation of the results of the experiments will also be part of this section.

In this paper, we focused on transaction processing.

II. Import Methods

Databases offer us various kind of data imports. This chapter describes some methods of importing data, which we can use to import data into the database system.

*A. SQL*Loader*

SQL*Loader is the utility of Oracle database system. This utility is one of the methods of the importing data from source file into database table. The file can be in various formats, which were created as an output from other systems. SQL*Loader can read only files, in which data is in the delimited format. Therefore, before loading the data, it is necessary to convert it into correct format. Data is usually in CSV format or some variants of it [9].

SQL*Loader distinguishes two types of methods:

- CONVENTIONAL - is the default method of the system. INSERT statements are being generated from processed data. These statements are executed within transactions and generates REDO/UNDO.
- DIRECT - is method, which writes data to the datafiles directly. It uses High Water Mark. It is a pointer to the last block of the table.

Import procedure of SQL*Loader is described below. It contains steps, how it is possible to use SQL*Loader utility.

At first, it is necessary to have data in the correct format stored in a file, for example CSV file format is appropriate. The next step is to create the database table with appropriate columns and datatypes. Then the control file must be created. Control file is file with suffix .ctl and it describes how the data is interpreted in the file and other options to load the data. The last step is to execute SQL*Loader utility with defined file. You can use command line with defined path where your

control files are saved [7], [9]. The statement below shows example of loading data using SQL*Loader tool.

```
$ sqlldr login/password@connect_string
control=control_file_name.ctl
```

The structure of the control file is displayed in the Fig. 2.

```
OPTIONS
  (
    DIRECT = {TRUE | FALSE}
    PARALLEL = {TRUE | FALSE},
    SKIP = n
    ....
  )
LOAD DATA
INFILE 'PATH_TO_FILE' "str '\n'"
BADFILE 'PATH_TO_BADFILE'
DISCARDFILE 'PATH_TO_DISCARDFILE'
APPEND
INTO TABLE table_name
FIELDS TERMINATED BY ','
OPTIONALLY ENCLOSED BY '"' AND '"'
TRAILING nullcols
  (
     column1    ,
     column2    ,
     column3    ,
     ....
  )
```

Fig. 2. Control file structure

The Fig. 3 refers to the concept of the SQL*Loader in Oracle database. As Fig. 3 shown, control file and data file enter to the SQL*Loader. SQL*Loader loads the data into database and, in some cases, create three types of files. Discard file, Log file and Bad file.

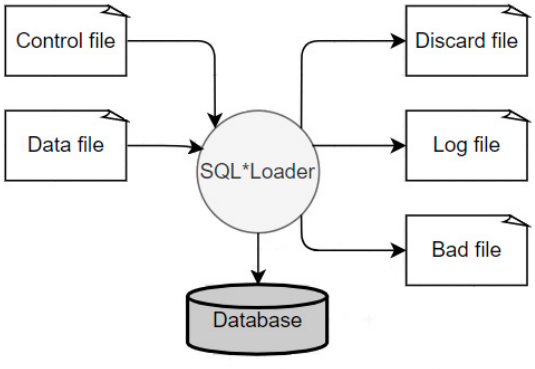

Fig. 3. Concept of SQL*Loader in Oracle database

The log file is always created and contains detailed information of the load, it also includes description of errors which occurred during the load. Records which were filtered out of the load due to not matching any selection criteria defined in the control file are included in discard file. The refused records are located in bad file [6].

B. *External tables*

External tables are an additional component of the SQL*Loader. External tables allow us to access data in external sources as if the data were stored in a database table. In the past, external tables were read-only, however later versions of Oracle database systems enable tables to be written to [7]. Syntax of the creating an external table is displayed in the Fig. 4.

As can be seen on the Fig. 4, external tables are created by SQL statement CREATE TABLE ....ORGANIZATION EXTERNAL and it is necessary to specify some attributes. Differecne between ORACLE_LOADER and ORACLE_DATAPUMP is, that ORACLE_LOADER access driver is the default and loads data which comes from text datafiles. On the other hand, ORACLE_DATAPUMP access driver can load data which comes from binary dump file [8].

As other import methods, external tables have same restrictions too.

- External table cannot load data into a column defined by the data type Long.
- External table cannot describe data which are stored in the database.
- External table cannot be changed by statement UPDATE.

Import procedure of external table is described below. It contains steps, how it is possible to use external tables.

At first, we create directory object which represents physical directory of operation system, in which data file is stored. This file contains data to load. The next step is external table creating. Then the content of external table can be displayed using the SELECT statement [7], [8].

```
CREATE TABLE table_name
  (
    column1   data_type,
    column2   data_type,
    ....
  )
ORGANIZATION EXTERNAL
  (
    TYPE{ ORACLE_LOADER | ORACLE_DATAPUMP }
    DEFAULT DIRECTORY directory_name
    ACCESS PARAMETERS
      (RECORDS DELIMITED BY '\n'
       SKIP 1
       BADFILE 'BADFILE_NAME.bad'
       DISCARDFILE 'DISCARDFILE_NAME.dsc'
       LOGFILE 'LOGFILE_NAME.log'
       FIELDS TERMINATED BY ','
       OPTIONALLY ENCLOSED BY '"'
       ....
       MISSING FIELD VALUES ARE NULL
       (
        column1          ,
        column2          ,
        ....
       )
    )
 LOCATION ('FILE_NAME')) REJECT LIMIT UNLIMITED;
```

Fig. 4. Syntax of the external table

*1) Difference between External tables and SQL*Loader*

In this section some of the differences between loading data with external table using ORACLE_LOADER access driver and SQL*Loader will be described.

The first difference is when there are multiple primary input datafiles. When SQL*Loader is used, it creates bad file and discard file for each input datafile, while external tables are used, it creates only one bad file and one discard file for all input datafiles. Another difference is that using of COCATENATE to combine multiple records into one or loading datatypes as GRAPHIC, VARGRAPHIC or using columns with types Long, Nested table, Varray and other are not supported with external table loads.

C. IMP method

This type of import is also known as original import utility. As other types of imports, IMP method is used to restore the database or to retrieve the transferred data from one database to another database. Functionality IMP is a client-server side tool. It means that the dump file is read from a local directory on the device where the tool is run [2], [3].

Import utility reads the data blocks from the file generated by export utility, creates INSERT statements and other DDL statements and loads data into the database. Statements are executed on the server side.

If a communication flow error is interrupted, the import will fail. Another limitation of this method is that the import speed depends on the speed of network communication. Therefore IMP methos is not so effective.

Below, there is the command to start the data import.

```
$ imp login/password@connect_string
file=export_file_name.exp
```

Export file name is name of the dump file, which was created by the export utility. This file is in a binary form and contains objects on a defined order. The type definitions are located in the file in the first place. Then there are table definitions, table data, table indexes, followed by constraints, views, procedures and the last part of the file is formed by bitmap and other types of indexes.

The import supports many other parameters for example buffer, filesize, log, statistics, tables, etc.

D. IMPDP method

This kind of method is also known as data pump import. It is also utility which is used for loading a dump file generated by export utility into target database. The file is in the binary format. This kind of method was introduced in Oracle Database 10g and still is improved [2].

Difference between this method and IMP method is that IMPDP method is performed without the client involvement. It works only on the server side. Therefore, any data transfer between the client and the server is eliminated [3].

When the impdp method is used, it is necessary to create a mapping object for accessing data storage on the server. User usually does not have access to data storage, it is mainly for safety.

Import procedure of impdp method is described below. It contains steps, how it is possible to use import utility.

At first it is necessary to create directory on the server. Then the Oracle directory can be created. During the creation of Oracle directory, it is necessary to map it to the created directory on the server. Oracle directory is a global database object, its owner is SYS. Oracle directory and its mapping is possible to make by command:

```
CREATE DIRECTORY directory_name AS
server_directory_path
```

When directories are created, there is a need to set access rights for writing and execution to the directory. After these steps it is possible to execute import. Below, there is the command to start the data import.

```
$ impdp login/password@connect_string
directory=Oracle_directory
dumpfile=export_file_name.dmp
```

The import supports many other parameters as tables, logfile, help, status, schemas, etc. These parameter files enable to specify command-line parameters in the file for ease to reuse. They are also useful because they help us to avoid some errors from typing long data pump commands on the command line [1].

III. EXPORT METHODS

A. EXP method

This type of export is also known as original export utility. It is a utility, that is used to transfer data between database systems. During the export, the SELECT statement is generated, and exported file is created on the client device in binary format. Exported file can be read by import utility into another database.

Functionality EXP is a client-server side tool. It means, that the dump file is written to a local directory on the device where the tool is run [2]. If there is a communication flow error, export will fail. Another disadvantage of this method is that the speed of export depends on the communication network speed [3].

Below, there is the command to start the data export.

```
$ exp login/password@connect_string
file=export_file_name.exp
tables=list_of_exported_tables
```

Export file name represents file, which is created by export. Parameter table contains list of tables, which are exported. One of the parameter specifies the use of direct parh. Its name is DIRECT. This parameter is important because the export will extract data by reading the data directly. It does not process SQL statements. Compared to the conventional path,

direct path is much faster. There are many other parameters which are the same as in IMP method [4], [5].

B. EXPDP method

This kind of method is also known as data pump export. This type of export is considered more efficient and faster than original export because it is performed only on the server side. So, the costs of connecting the client and server part, are eliminated [3].

As with IMPDP method it is newer version of EXP method, and it is necessary to have a mapping object and the Oracle dictionary created. If access rights are not set, they have to be set by command $chmod. Below, there is the command to start the data export.

```
$ expdp login/password@connect_string
tables=list_of_tables
directory=Oracle_directory
dumpfile=export_file_name.dmp
```

Many parameters are available in EXPDP utility. Export file name represents name of dump file which will be created. Then there is the logfile parameter which define name of logfile which will be created. This file contains information about executed activities and errors which occurred during the export. Parameter with name tables defines a list of tables which will be exported. Then there are other types of parameters as schemas, full, content, query etc.

IV. EXPERIMENTS

The experiments, that were performed, are aimed at comparing import and export methods over time. This type of experiment is important in order to determine the duration of each method. Based on that it is possible to determine which method had the best processing times and decide which method can be used is specific problem.

This paper contains three parts of the comparison. The first part is focused on import methods, the second part describes export methods and the last one is aimed on comparing the speed of the select statement in external and internal tables.

Performance characteristics have been obtained by using the Oracle 19c database system. Parameters of used computer are:

- ***operation memory*** – 16 GB
- ***HDD*** – 1000 GB
- ***processor*** – Intel Xeon E5620; 2,4GHz (8 cores)

Network characteristics are:

- **upload speed** – 5 Mbps
- **download speed** – 50 Mbps
- **latency** – 11.5 ms

All experiments were tested on three types of datasets. It is possible to divide them into:

- ***small dataset*** - contains 19532 records.
- ***medium dataset*** - contains about 6 500 000 records.
- ***large dataset*** - contains about 32 400 000 records.

A. Import methods

The experiment was tested on three types of datasets which were imported into the database. The Table I shows the import duration for each dataset. The time in the Table I is given in seconds.

TABLE I. DATA IMPORT DURATION (SECONDS)

	sqlldr	imp	impdp
Small dataset	3.188	3.875	6.0
Medium dataset	906.112	379.290	22.50
Large dataset	4592.933	2116.032	92.50

The graphic representation is shown in the Fig. 5. Values in graph represents data import duration in seconds for each dataset.

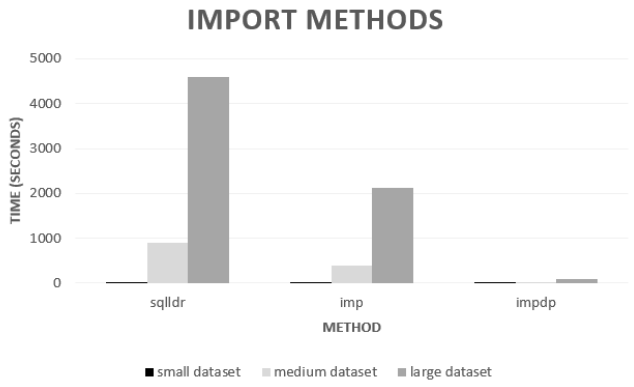

Fig. 5. Graphic representation of data import method

As can be seen in the graph, with a small dataset, the results are comparably the same for all import methods. If there are some differences, it might be due to the current server load. With the medium and large dataset, the imp and sqlldr functions are performed on the client side, and therefore there are additional costs for connecting the client to the server. The biggest difference in the execution time of import methods was manifested in the large dataset, which in the case of the sqlldr function took one hour and sixteen minutes, in the case of the imp function took about 35 minutes and the fastest function was impdp, that is executed on the server side without client involvement.

The conclusion of this experiment is that the fastest function is impdp because it is performed on the server side and data transfer between the server and the client side is eliminated. With a small dataset, the result is not so radical, but with larger datasets, the processing time of the functions imp and sqlldr is much slower than with the impdp function.

B. Export methods

As with import methods, experiments with export methods were tested on the same datasets. Two methods were compared. The first one was exp and the second one was expdp. The Table II shows the export duration for each dataset. The time in the Table II is given in seconds.

TABLE II. DATA EXPORT DURATION (SECONDS)

	exp	expdp
Small dataset	5.266	7.889
Medium dataset	354.072	24
Large dataset	2940.488	84

The graphic representation is shown in the Fig. 6. Values in graph represents data export duration in seconds for each dataset.

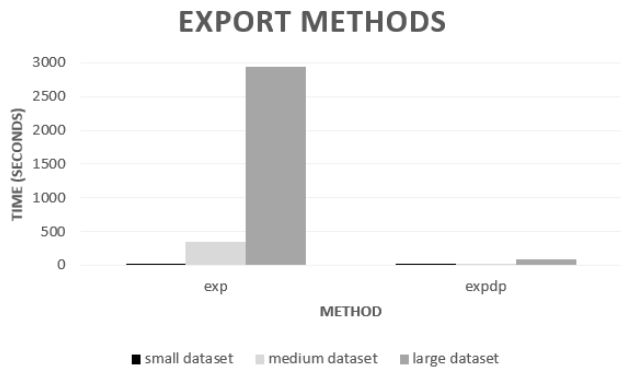

Fig. 6. Graphic representation of data export method

As can be seen in the graph, the execution time of export in both cases is almost the same. Time differences become apparent for larger datasets. With a medium dataset, exp function took about 6 minutes while expdp function took only 24 seconds. The last dataset shows the largest time difference between functions. Using exp function, execution time is about 50 minutes while using expdp function, execution time is only 84 seconds.

The conclusion of this experiment is that the fastest function is expdp because, like impdp, it is performed only on the server side, without client involvement. There is no need to connect to the client side and therefore expdp function is much faster.

C. Internal vs. external table

The following experiment compares the execution speed of the select statement in external and internal tables. Difference between these two types of tables is that external table accesses data that is in an external source. The data is not stored directly in the database table.

For the purpose of the experiment, three external tables and three internal tables were created. Internal tables contain the same data as experiments above and external tables were mapped to the same datasets as experiments described above. Experiment was aimed at comparing the performance speed of a simple select statement, which returns count of all records from the table. The SELECT statement looked like this:

```
SELECT COUNT(*) FROM tab_name;
```

SELECT statement was performed 10 times for each table. Times given in Table III show the average of these times.

TABLE III. EXTERNAL VS. INTERNAL TABLE (SECONDS)

	Internal table	External table
Small dataset	0.023	0.108
Medium dataset	0.401	26.522
Large dataset	4.853	107.260

The Table III displays duration of select statement for three datasets of different tables. The time in the Table III is given in seconds.

The graphic representation is shown in the Fig. 7. Values in graph represents duration of the select statement tested on internal and external tables. Duration is expressed in seconds for each dataset.

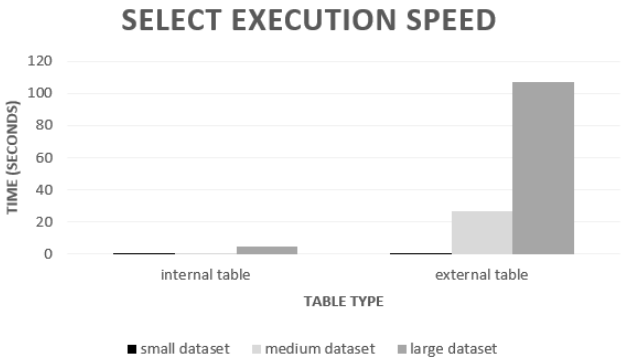

Fig. 7. Graphic representation of the SELECT execution time

As can be seen in the graph, with a small dataset, the duration of the select statement for internal table is 0.023 seconds on average while for external table is 0.108 seconds. Larger differences occur with the medium dataset. Internal table is able to execute select statement in 0.401 seconds, while the duration of the select statement took 26.522 seconds in external table. The biggest difference in time was in the last part of experiment. The execution time of the select statement in the internal table was 4.853 second on average, while in the external table it was 1 minute and 47 seconds on average.

The conclusion of this experiment is that internal tables execute statements faster than external tables. It is because the data is stored directly in the database table. Mapping of external tables is extremely fast, but they must be connected to an external data source, that is stored on the server side, and therefore the execution time of the statements is longer. Another difference between tables is that it is not possible to insert or delete data from the external table, while in the internal table it is possible.

V. Conclusion

Import and export methods are very important for transferring data between databases, or for restoring data in the database. When working with databases, it is good to know what possibilities the database system offers for data transfer and loading, what methods for import and export are available and which of them are more efficient.

This paper is focused on an overview of import and export methods, which are described in the second and the third chapter. The fourth chapter of the article is focused on the comparison of import and export methods in terms of time. Three types of experiments were performed. Each of them was tested on the same dataset.

The first experiment was focused on comparison of the import methods. As shown, the impdp method was the fastest in time. This was mainly because it is running at the server level and therefore the additional costs of connecting to the client are eliminated.

The second experiment was focused on comparison of the export methods. As shown, the expdp method was the fastest in time. It is due to the same reason as impdp method.

The last experiment was focused on comparing the execution time of the select statement in the internal and external table. Select statement, executed in internal table, was faster than in external table, because the data are directly stored in the database table. So, there is no need to connect to an external data source.

The main aim of this paper was to introduce methods of data transfer, compare execution time of import methods, export methods and external table in transaction processing. Server-side tools as impdp and expdp were much faster than client-server side tools. This is because there is no need to transfer data between client and server.

Acknowledgment

This article was created in the framework of the National project IT Academy – Education for the 21st Century, which is supported by the European Social Fund and the European Regional Development Fund in the framework of the Operational Programme Human Resources.

The work is also supported by the project VEGA *1/0089/19 Data analysis methods and decisions support tools for service systems supporting electric vehicles* and the *Grant system* of the University of Žilina.

References

[1] Oracle, *Oracle Database Utilities Data Pump – Best Practices for Export and Import*. 2019.

[2] B. Thomas, "Do More with Data Pump: Tips and Techniques", unpublished.

[3] Oracle, *Oracle Database Utilities: Full Transportable Export/Import, 2018*

[4] M. Kvet, K. Matiaško, M. Kvet, "Complex time management in databases", *Central European Journal of Computer Science* vol.4, 2014, pp. 269-284, doi: 10.2479/s13537-014-0207-4

[5] M. Kvet, K. Matiaško, "Transaction Management in Temporal System", *Iberian Conference on Information Systems and Technologies,* CISTI, 2014, doi: 10.1109/CISTI.2014.6876998

[6] M. Kvet, K. Matiaško, "Time as the Important Factor of the Data Retrieval – Table type Classification", *Advances in Intelligent systems and Computing,* vol. 569, 2017, pp. 492-502, doi:10.1007/978-3-319-56535-4_50

[7] T. Laszewski, P. Nauduri, *Migrating to the cloud – Chapter 5 Database Schema and Data Migration,* 2012, pp. 93-130

[8] J. Clarke, *SQL Injection Attacks and Defense – Chaper 6 Exploiting the Operation System,* 2009, pp 271-315

[9] Oracle, *Express Mode Loading with SQL*Loader in Oracle Database 12c,* 2013.