Application Tool for Experiments on SQL Server 2005 Transactions

ŞERBAN GHENEA
Omnis Group SRL
6 Barbu Văcărescu, Bl. 2, Apt. 1, Sector 2, Bucharest 020281
ROMANIA

Abstract: - The paper presents the work performed on a SQL database that holds the physical commercial operations carried on in an oil refinery. The data was selected, preprocessed and transformed, in order to obtain a consistent data context. The resulting database was processed using Microsoft Analysing Services Server in SQL Server 2005. Views were defined in order to obtain and process an OLAP cube. Data mining structures were defined and applied on the views and an application was built to generate test transactions, to inspect the locks' status, to visualize data mining results and to assess performance.

Key-Words: - SQL Server 2005, SQL Server Analysis Services, transactions, data mining, experiments

1 Introduction

The application tool we developed allows the user to concurrently execute a number of predefined transactions, to monitor the dynamics of the active transactions and the situation of the locks generated on the Analysis Services Server, to view the logs and to investigate the results obtained by processing various data mining models. In our example, the set of transactions include several selection transactions specific to the TPC-H benchmark, the processing of an OLAP cube, the execution of data mining algorithms provided by the SQL Server 2005 data mining platform, as well as data update transactions – insert and delete.

Selecting various transactions for concurrent execution may lead to deadlocks. Microsoft Windows System Monitor is used to inspect the values of specific performance indicators on SQL Server 2005 Database Engine and on SQL Server 2005 Analysis Services (SSAS) during transactions execution.

2 Data Context

2.1 Data Source

The application was tested using data extracted from the database that holds the physical commercial operations performed in an oil refinery. Customers buy, term or spot, oil products loaded on trucks, railcars and sea-going vessels. A commercial transaction refers to one or more products lifted by the same customer and is represented in the database by one or more primary records, one record for each product. 58,252 such primary records, registered in a period of 3 years, were processed for these experiments.

The main database tables that hold most of the relevant information of physical oil operations are DELIVERY, CONTRACT, PRODUCT, CUSTOMER, ADDRESS.

2.2 Data Preparation

The data extraction process was performed in conjunction with and followed by appropriate data cleansing and modification processes:

- eliminating records with zero quantities, introduced by the interface with the product loading equipment;
- solving data inconsistencies due to the text representation of *City* and *Street* fields (duplicates and misspelling);
- decomposing the *Date* field in 3 separate fields – year, month, day;
- creating new database tables, such as TIME_BY_DAY, based on the above mentioned *Date* field partitioning.

A new Analysis Services project was then created using Business Intelligence Development Studio and a data source was added in order to connect to the SAMPLEDB database. Several data source views were created with the purpose of modifying the structure of the data to make it more relevant to the project.

The Delivery_Facts view includes one record for each customer and for each product delivered to that customer in one day. The view provides 30,111 result

lines, obtained by aggregating the 58,252 primary records in the DELIVERY table. A line in the Delivery_Facts view provides information for the delivered product, the delivery time, the customer, the contract sale type (*Term* or *Spot*) and the delivered quantity (in liters, kilos and liters at 15ºC). There are distinct dimension tables for each information category – product, customer and time.

An OLAP cube was defined using the measures product, customer, contract type and time (TIME_BY_DAY).

Three mining structures were added: multidimensional association, time series and decision trees, as well as one mining model based on each structure.

Figure 1 shows the schema of the obtained SQL Server 2005 data structures, based on the SAMPLEDB database and further refined using the Business Intelligence Development Studio and SSAS.

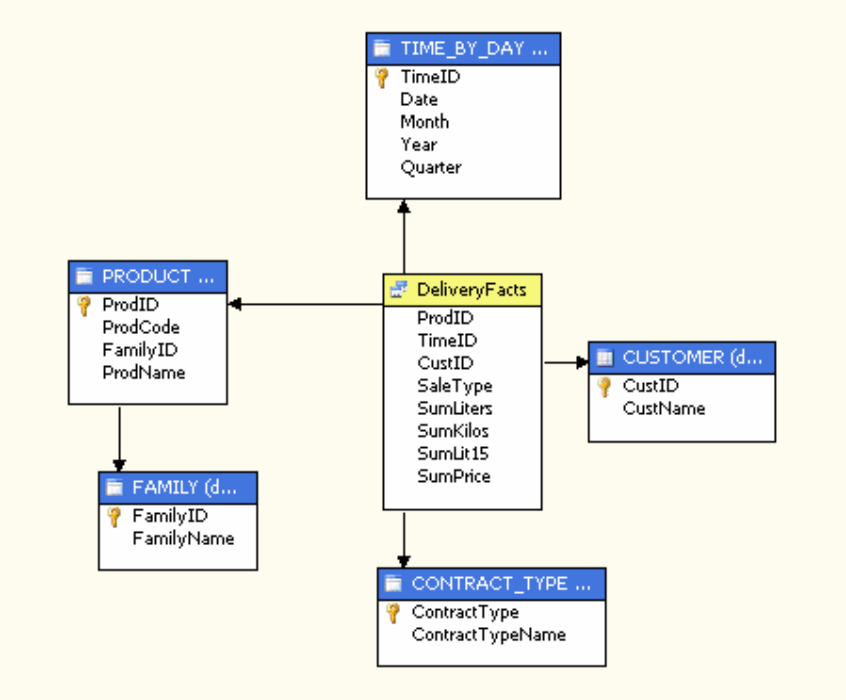

Fig. 1

3 The Application

The equipment used in experiments consisted of:

- client computer Intel Pentium III at 1.13 GHz with 512 MB RAM, running Windows XP Professional with SP2;
- server running SQL Server 2005 Standard Edition (32 bits) on 2 X Xeon 3GHz, with 4GB RAM and 2 X 160 GB mirrored hard disks, under Windows 2003 Server.

Nine types of transactions were defined for the experiments, in order to process:

- 3 selection queries used in TPC-H benchmark (**S1_Pricing Summary Report**, **S2_Forecasting Revenue Change** and **S3_MarketShare)**;
- a 4-D OLAP cube (**S4_Process Deliveries Cube**);
- 3 data mining models (**S5_Process Association Model**, **S6_Process Decision Tree Model**, **S7_Process Time Series Model**);
- 2 data update transactions (**U1_Insert Deliveries** and **U2_Delete Deliveries**).

The TPC-H transactions S1, S2 and S3 were modified according to the structure of SAMPLEDB. S4, which provides the processing of the OLAP cube, scans the facts table and fills each cube cell according to the cube's dimensions and measures. The data mining models, described by transactions S5, S6 and S7 are based on the algorithms embedded in SQL Server 2005 Analysis Services.

The execution of complex transactions S4 – S7 entails the execution of simpler transactions, both at

the SQL Server database engine and at the SSAS level. Transaction U1 ensures the insertion of records in the DELIVERY tables, whereas transaction U2 deletes the records inserted by a previous U1 transaction, specified by its execution ID. These data updates directly affect the Delivery_Facts view and impose new processing of the cube and of the data mining models.

The 9 transaction types were serially and concurrently launched in execution, using the interface provided by the application. The operations performed by SSAS, as well as the queries sent to the SQL Server Database Engine were monitored using Microsoft Windows System Monitor (Performance Monitor), SQL Server Management Studio (Query Analizer) and SQL Server Profiler, which is able to correlate the peformance indicators under analysis.

3.1 Functionalities

The *Execution* tab in the application interface, shown in Figure 2, allows the user to select from a listbox one or more transactions and to launch them into execution by clicking the *Execute* button. The parameters of the update transactions U1 and U2 can be specified in the edit controls below the selection list box. Another listbox, on the lower left side of the screen, displays the transactions currently under execution. A text control displays the content of the first query (transaction) selected for execution.

3.2 Results

The Analysis Server tab of the application interface displays the status of the active SSAS transactions, as well as the lock status of each transaction. Figure 3 shows the locks generated on SSAS by two concurrently active transactions.

If the transactions S4, S5, S6 and S7 were launched at the same time, the Performance Monitor on SQL Server Database Engine indicates that they were serially executed, in the order S4, S6, S5, S7, as shown in Figure 4.

If all transactions are concurrently launched, U1 (insert delivery records) will cause a deadlock due to the interraction with the cube processing carried on in S4. The resulting deadlock is reported by Performance Monitor on SQL Server Database Engine and on SSAS.

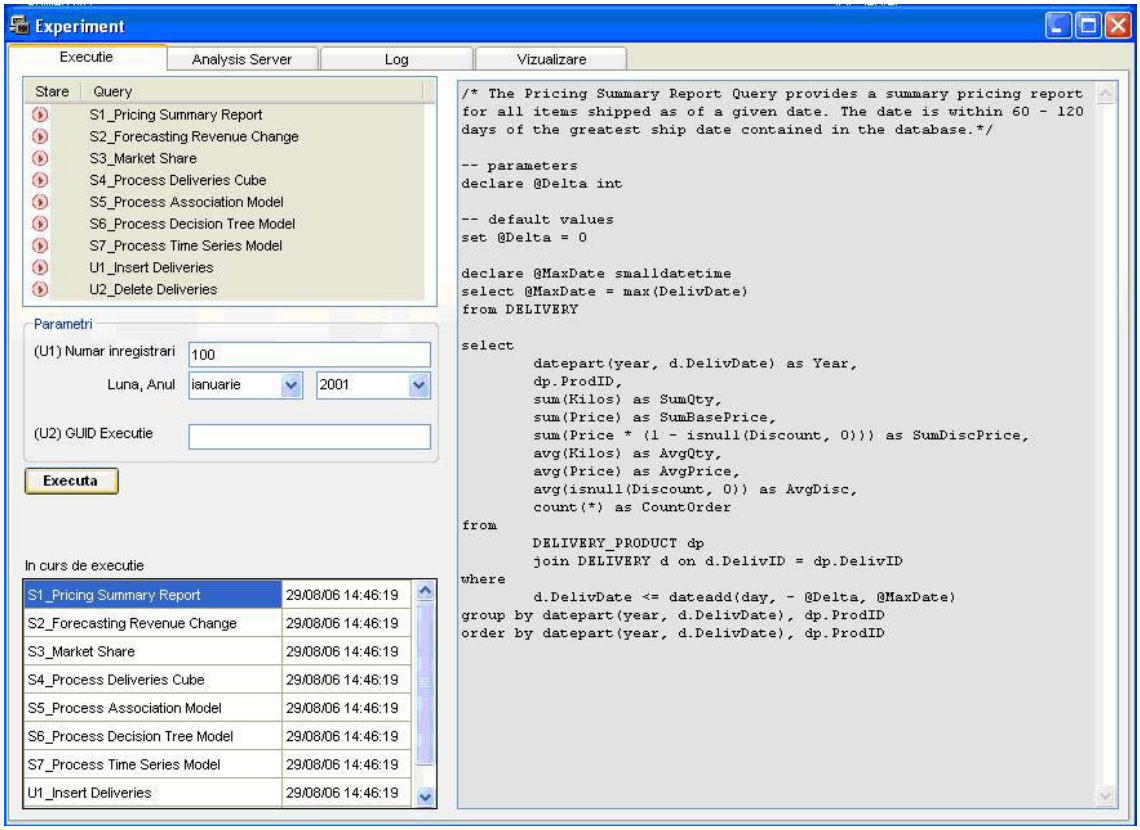

Fig. 2

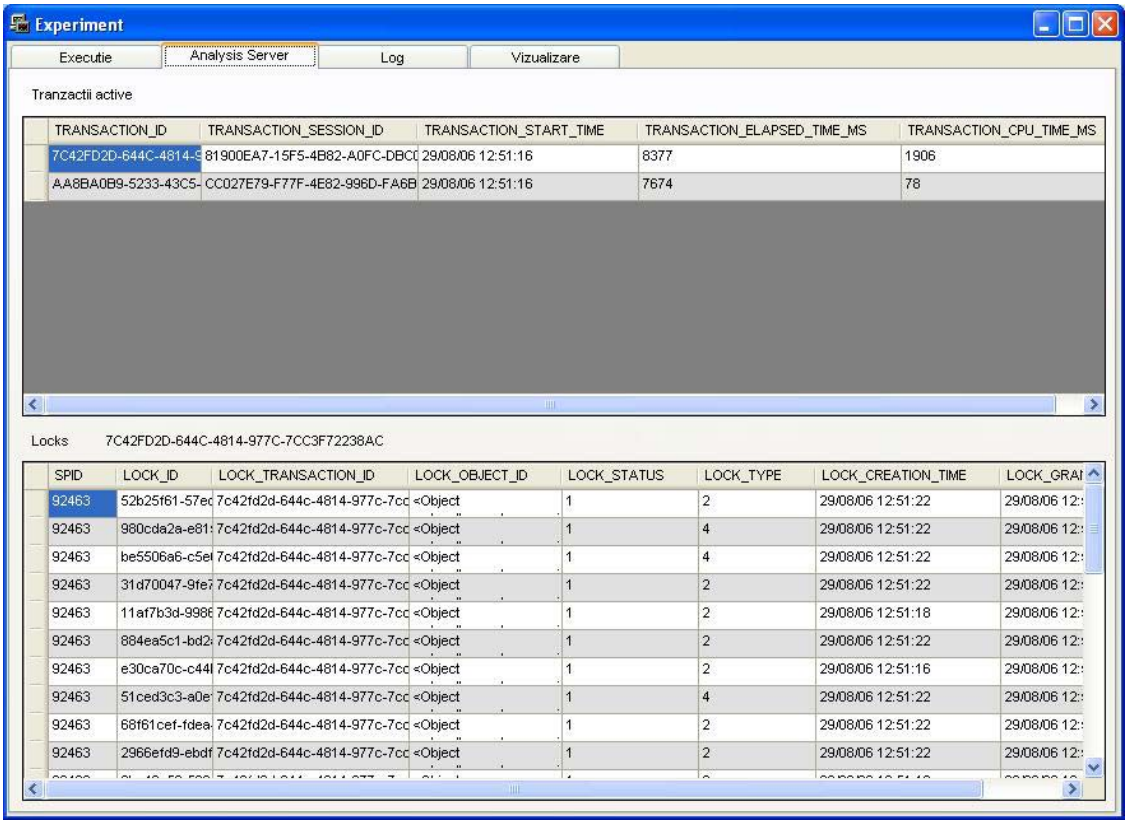
Experiment
Executie
Analysis Server
Log
Vizualizare
Tranzactii active
TRANSACTION_ID
TRANSACTION_SESSION_ID
TRANSACTION_START_TIME
TRANSACTION_ELAPSED_TIME_MS
TRANSACTION_CPU_TIME_MS
7C42FD2D-644C-4814-9
81900EA7-15F5-4B82-A0FC-DBC0
29/08/06 12:51:16
8377
1906
AA8BA0B9-5233-43C5-
CC027E79-F77F-4E82-996D-FA6B
29/08/06 12:51:16
7674
78
Locks
7C42FD2D-644C-4814-977C-7CC3F72238AC
SPID
LOCK_ID
LOCK_TRANSACTION_ID
LOCK_OBJECT_ID
LOCK_STATUS
LOCK_TYPE
LOCK_CREATION_TIME
92463
29/08/06 12:51:22
29/08/06 12:51:18
29/08/06 12:51:16

Fig. 3

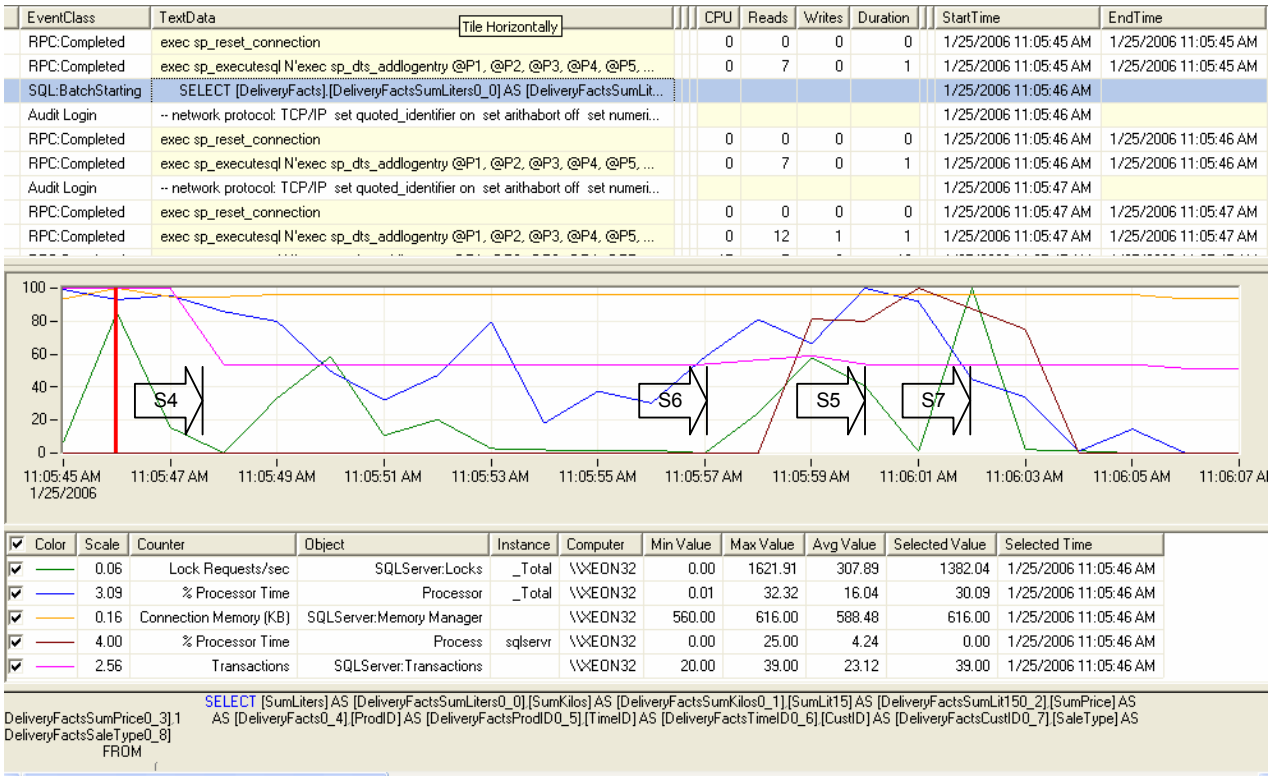
EventClass
TextData
Tile Horizontally
CPU
Reads
Writes
Duration
StartTime
EndTime
RPC:Completed
exec sp_reset_connection
exec sp_executesql N'exec sp_dts_addlogentry @P1, @P2, @P3, @P4, @P5, ...
SQL:BatchStarting
SELECT [DeliveryFacts].[DeliveryFactsSumLiters0_0] AS [DeliveryFactsSumLit...
Audit Login
-- network protocol: TCP/IP set quoted_identifier on set arithabort off set numeri...
1/25/2006 11:05:45 AM
1/25/2006 11:05:46 AM
1/25/2006 11:05:47 AM
S4
S6
S5
S7
11:05:45 AM
1/25/2006
11:05:47 AM
11:05:49 AM
11:05:51 AM
11:05:53 AM
11:05:55 AM
11:05:57 AM
11:05:59 AM
11:06:01 AM
11:06:03 AM
11:06:05 AM
Color
Scale
Counter
Object
Instance
Computer
Min Value
Max Value
Avg Value
Selected Value
Selected Time
Lock Requests/sec
SQLServer:Locks
% Processor Time
Processor
Connection Memory (KB)
SQLServer:Memory Manager
Process
Transactions
SQLServer:Transactions
\\XEON32
sqlservr
1/25/2006 11:05:46 AM
SELECT [SumLiters] AS [DeliveryFactsSumLiters0_0],[SumKilos] AS [DeliveryFactsSumKilos0_1],[SumLit15] AS [DeliveryFactsSumLit150_2],[SumPrice] AS
AS [DeliveryFacts0_4].[ProdID] AS [DeliveryFactsProdID0_5],[TimeID] AS [DeliveryFactsTimeID0_6],[CustID] AS [DeliveryFactsCustID0_7],[SaleType] AS
DeliveryFactsSumPrice0_3],1
DeliveryFactsSaleType0_8]
FROM

Fig. 4

The Microsoft Association algorithm, run by S5, is an association algorithm provided by SSAS, useful for recommendation engines. A recommendation engine recommends products to customers based on items they have already bought, or in which they have indicated an interest. The *Rules* tab displays the rules that the association algorithm discovered. The *Rules* tab includes a grid that contains the following columns: *Probability*, *Importance*, and *Rule*. The probability describes how likely the result of a rule is to occur. The importance is designed to measure the usefulness of a rule. Although the probability that a rule will occur may be high, the usefulness of the rule may in itself be unimportant. For example, the experiment identified a rule stating that if the customer with ID 184 buys products with IDs 03 and 04, he will also buy product with ID 02 (probability 1). However, the rule is considered trivial, as its importance is only 0.264. The greater the importance, the more important the rule is.

The results in Figure 5 were displayed using SQL Server 2005 Data Mining viewer controls.

The Microsoft Time Series algorithm, run by S7, is a regression algorithm provided by SSAS for use in creating data mining models to predict continuous columns, such as product sales, in a forecasting scenario. While other Microsoft algorithms create models, such as a decision tree model, that rely on being given input columns to predict the predictable column, prediction for a time series model is based only on the trends that the algorithm derives from the original dataset while it is creating the model. The results in Figure 6 show a typical model for forecasting prices and sales of a product over time. There are two parts in the diagram: historical information and predicted information (represented with dotted line). The historical information is used by the algorithm to create the model and to generate the forecast. The line that is formed by the combination of historical and predicted information is called a *series*. Relative curves were displayed, because the chart shows both the price and the quantity models and their data scales are considerably different.

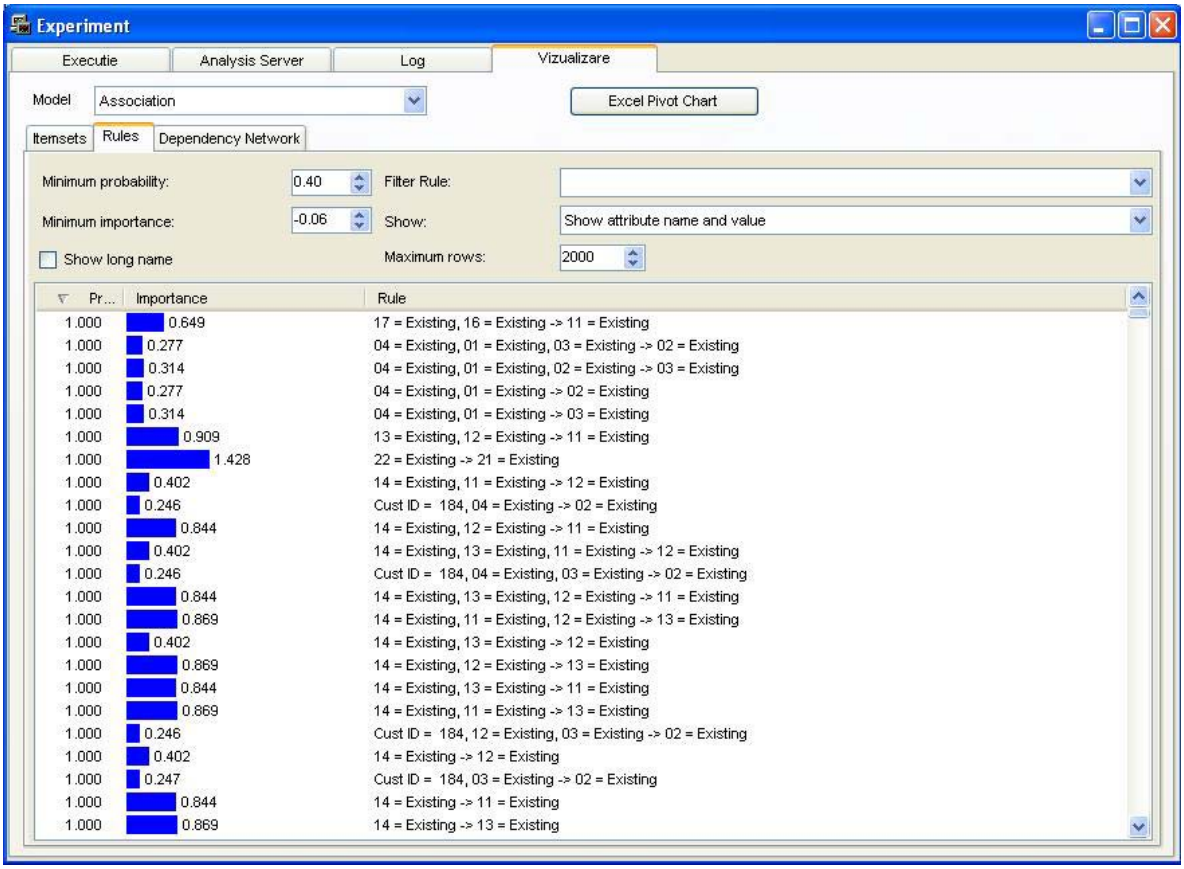

Fig. 5

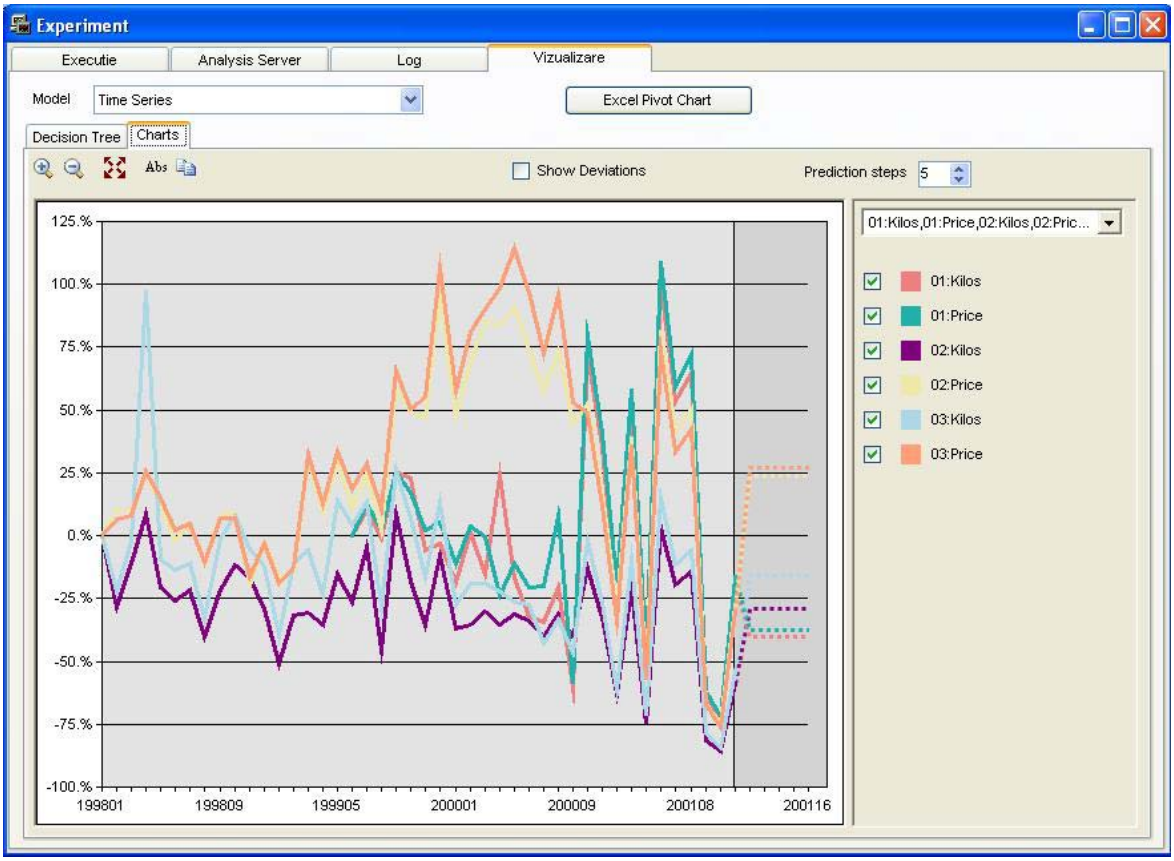

Fig. 6

4 Conclusions and New Developments

The application tool can be used to assess the system performance in running transactions on SQL Server 2005 databases. The processed data volumes can be adjusted using data update transactions – *insert* and *delete*. The SSAS log content and the transactions locks status can be monitored and analysed.

The application will further be developed to allow connection to multiple data sources using Analysis Services OLE DB Provider. It will also integrate all the data mining models provided by SSAS. In fact, the application will be able provide most of the functionalities included in Visual Studio 2005 Business Intelligence Development Studio – creation of data views, OLAP cubes and data mining structures. SQL Server 2005 database administrators will be in the position to perform many business intelligence tasks without using Visual Studio, working with a simpler interface.

A web interface will be integrated in the application, allowing Internet access to the data under analysis.

References:

[1] Jim Gray, Andreas Reuter. *Transaction Processing: Concepts and Techniques*. Morgan Kaufmann Publishers Inc., 1993

[2] Jiawei Han, Micheline Kamber. *Data Mining – Concepts and Techniques*. Morgan Kaufmann Publishers, 2001

[3] Margaret H. Dunham. *Data Mining – Introductory and Advanced Topics*. Prentice Hall, 2003

[4] ZhaoHui Tang, Jamie MacLennan. *Data Mining with SQL Server 2005*. Wiley, 2005

[5] Microsoft SQL Server 2005 Books Online, Microsoft Corporation

[6] Transaction Processing Performance Council - TPC BENCHMARK H Standard Specification Revision 2.3.0

[7] J. Han, J. Pei, Y. Yin. Mining Frequent Patterns without Candidate Generation. *Proc. of ACM-SIGMOD*, 2000.

[8] R. Agrawal, R. Srikant. Fast Algorithms for Mining Association Rules în Large Databases. *Proc. of 20th Int'l conf. on VLDB*, 1994.