International Journal of Computers & Technology
Volume 4 No. 1, Jan - Feb, 2013

THE PLACE OF THE STORED PROCEDURES IN THE CLOUD DATABASES

Dr. Constantin Florin Sîrbu
Al. I. Cuza University of Iaşi, Romania
Department of Business Information Systems

ABSTRACT

The software as a service model ensures not only cloud applications, but also cloud databases. In this paper we analyze the impact on stored procedures of the switch of the existing application to cloud applications and cloud databases. We demonstrate that the stored procedures and cloud applications have common advantages and we emphasize also other advantages of the stored procedures. As a contribution we propose a list of improvements to stored procedures that can be considered in the new versions of the SQL standard and implemented by the database management systems providers. For the moment the SQL standard does not include any feature related to the cloud databases, so we expect that once again the software industry will be one step forward to the SQL standard.

Keywords

Stored procedures, cloud databases, SQL standard, SQL implementations.

1. INTRODUCTION

Classical data processing in client-server architecture involves transfer of the data from the database management system to the client application, and after their processing the transfer of the results back to the database server. From this point of view, the stored procedures have common advantages with the cloud applications. We will name in this article classical systems the systems that do not include cloud technologies.

The processing of data where is stored diminishes considerably the use of the network resources and benefits from the important resources of the servers. By using stored procedures and/or cloud applications, data processing operations are performed on systems much faster than workstations, which may decrease the waiting time of the system.

2. ADVANTAGE OF USING THE STORED PROCEDURES

Performance of the stored procedures comes also from the preparation for their processing at definition time. A query done outside a stored procedure has the disadvantage that each time it is sent by a client application to the database server it undergoes a process of decomposition, analysis and optimization in order to formulate a plan. A stored procedure is decomposed, analyzed, optimized and stored in an executable form when it is created. A stored procedure is not subject to a process of preparing the execution each time it is called and so, it runs faster than an equivalent query.

When a database is used by more applications, the stored procedures ensure avoiding duplication of code and of all the disadvantages arising from this. This is also the case of using more cloud applications with the same database.

One advantage of stored procedures over cloud applications or any other type of application is the possibility of implementing data validation restrictions that are not included in the relational constraints, using triggers. The classical referential restrictions cannot be specified in distributed databases over data from different servers. These restrictions may be implemented in the database environment only with stored procedures.

Stored procedures provide via triggers a possibility to implement access policies, controlling who, what, when, from where and with what tool makes the changes. This role of the stored procedures may still exist in the cloud databases only with the direct contribution of the cloud applications, because these would be the clients of the cloud databases.

The contribution of the stored procedures to the data protection is not only through the access policies, but also in the implementation of the logging mechanism.

The new versions of the major DBMS include or will include the functionality of storing procedures in the database environment, and so, the migration to clouds will be easier if in the old systems were used stored procedures.

Stored functions extend the data manipulation capability of a DBMS because they can be used in SQL queries. By expanding the data processing capability of the database queries the cloud applications will became lighter and less complex.

Stored procedures written in generalized programming languages (Java, C) have the advantage of portability, scalability and rapid development. Such a stored procedure written in a general programming language can be used on any platform (software and hardware) on which that language can be compiled. Furthermore, a stored procedure can run with small changes in a different type of application, making once again the move to clouds more easily. A great portability is achieved between different DBMS's that support the same programming languages for writing stored procedures.

3. THE CHALLENGES OF USING STORED PROCEDURES

Like any other functionality in a system, the stored procedures bring new challenges to data protection and security. For attackers, stored procedures are not only new targets, but also new ways of accessing data from databases.

Stored procedures that are written in traditional programming languages may contain Trojans, worms or viruses, but the stored procedures written in a SQL dialect may not contain malicious programs because the dynamic SQL environments were created in such a way to avoid code multiplication.

The existing implementations of the stored procedures run after the "first come - first served" rule. In classical systems, this may cause problems when the procedures have different importance labels. In the cloud databases the processing resources of the clouds may be huge, triggering from this advantages and disadvantages. An advantage is the fact that an important process will always find the necessary resources to run and a disadvantage may arise if the developers will not consider the optimization issues. In time, the old performance problems will became new financial problems, depending on the contract with the database-service provider.

It is difficult to follow a stored procedure while running (and in some implementations impossible) without accepting, from time to time, the changes made over the data. Although SQL standard does not include such specifications, there are mechanisms that can be used to improve communication between the client that launched a specific stored procedures and the database server (e.g. pipes, autonomous transactions).

The stored procedures cannot be optimized by DBMSs as well as the declarative solutions, and therefore part of the optimize efforts belong to developers. Even when the stored functions are triggered by declarative instructions, their code remains outside optimizer's actions [4]. When is scheduled a move to cloud applications because of the performance problems encountered with the classical systems, the need of code optimization will still be an issue because of the effects over costs. We recall in this regard Date's advice "declarative solutions, when available, are always to be preferred to procedural onces". [3]

4. PROPOSED IMPROVEMENTS FOR STORED PROCEDURES

Despite evident advantages of stored procedures, there are a number of possibilities for improving them. We present a list with proposals for improving the SQL standard and the SQL implementation of the stored procedures.

4.1 The code management

We propose to save in the database dictionary not only the code of the stored procedures, but also code management information, like the different versions of a stored procedure and information regarding the changes brought by a new version of a stored procedure, or the reasons of these changes.

Choosing the version of a stored procedure may be done in at least three different ways:

- By default using the latest version;
- Using dynamically the latest version accessible with the rights of the client at runtime, in case the access control list contains also version data;
- Choosing expressly a specific version to be run.

All this logic should be implemented now in the logical design of each database instead of using the declarative features of SQL.

4.2 The package management

The actual stored procedures may be organized in packages that group procedures and functions that belong to the same system functionality. In large systems, organizing the stored procedures just in packages and simple procedures and functions may not be sufficient for code readability. That is way we propose the creation of a bigger code hierarchy, like subpackages stored in other packages or the possibility to develop groups of packages.

Also related to package management, in the actual SQL standard and SQL dialects one package may be modified just by replacing it entirely, which is an inconvenience in the code management when dealing with large development groups. The creation, modification or deletion of the package's components should be made without affecting the other components of the same package.

4.3 The process management

4.3.1 The relative importance of the procedures

The rule "first come - first served" used in the actual stored procedures lets the database administrator to manage the resource loading. In cloud architecture the developer should be able to specify the relative importance of the stored procedures or the database-service provider should create tools for their clients to administer their own processes. Such tools could be used also for cost optimizations, for example, some procedures that have to run during the night may receive an importance label to run during the night at the cheapest rate (when the cloud is less loaded) independently on the exact hour.

4.3.2 Asynchronous processes

The stored procedures are used also for building ETL tools. One of their disadvantages regarding these tools is that the SQL stored procedures do not support multiple threads and as a consequence they cannot be used to run asynchronously different tasks in big business processes.

Till now this weakness had as a workaround the solution of using two or more database jobs scheduled at exactly the same time. But in cloud databases, the possibility of establishing jobs for a time interval instead of an exact time could lower the costs.

4.4 Improvements on security

The only action that can be specified for a privilege over a SQL stored procedure is now EXECUTE. In practice, more actions are needed for such privileges, like: READ (to be able to read its code), COMPILE (to allow the compilation of a stored procedure), REPLACE (to allow changes over the stored procedures definition and body). These improvement are not directly linked with the cloud features of the databases, but may be implemented for a better security.

4.5 Improvements on triggers

4.5.1 The order of execution

According to the SQL standard "the order of execution of a set of triggers is ascending by value of their timestamp of creation in their descriptors, such that the oldest trigger executes first"[11]. This is one example in which the SQL standard remained one step before the implementations. For example in Oracle 11g the clause FOLLOWS lets the developer to specify the relative firing order of triggers of the same type [9].

4.5.2 Multi base tables

According to the SQL standard "a trigger is an object associated with a single base table or a single viewed table"[11].

In practice, we may need triggers that look exactly the same or with the same basic structure, but defined on different tables. For example, the logging mechanisms implemented in the logical design of the database suppose such triggers. To avoid code duplication, we look for solutions that suppose the definition of the same trigger on different tables and the ability to find dynamically the current base table, at run time, during the execution of a trigger.

The problems encountered in this case by DBMS's producers would be in pre-compiler, when there has to be created a list with the columns accessible at runtime. Maybe a simple and fast solution would be to use in such triggers only the common columns of the group of tables defined for a trigger.

4.5.3 New trigger events and clauses

The SQL standard restricts the events of the triggers just for DML operations on tables, but in the SQL implementations there are more event types (DDL events, schema events, and database events). Besides the trigger events already implemented even without standard specifications, there may be invented new trigger events like "the end of the transaction". In practice, there are businesses rules that cannot be implemented entirely into the database environment without any workarounds because of this lack.

In business information systems, the generation of the accounting journal using the transaction journal may be implemented online or with batch processing. Because all the transaction data is in the database, the output data is saved also in the database, and there is no need for user intervention, the best place to implement this algorithm is the database environment with stored procedures. Both processing strategies (batch and online) may use stored procedures.

The online generation of the accounting notes need to capture the event when the algorithm has to run with triggers declared on the tables used for storing transaction data.

When there are no references between the transactions tables, on these tables can be created INSERT triggers that contain specific logic to generate the correspondent accounting articles. These may be AFTER triggers, to ensure that in the BEFORE triggers were already imposed some business rules.

When there are references between the transactions tables (such as the child table INVOICE_PRODUCTS references the parent table INVOICES) the use of triggers to generate accounting articles is challenging because:

- the rows from the child table may be inserted only after the correspondent insert on the parent table.

- in the body of a trigger the base table is mutating (cannot be queried).

These triggers may not be defined on the INSERT event of the parent table because these accounting articles may need amounts from the child table, but in the child table the correspondent rows were not inserted yet.

Also the child table cannot be the base table of such trigger because is not known what child row is the last that refers to a specific transaction (for example, which is the last line of an invoice) and because these tables cannot be queried from their own triggers.

The actual solutions of such algorithms implemented with stored procedures suppose to capture the moment when the accounting articles may be generated, for example when is updated a status column from the parent table. The risk would be in this case to lose such an event, which does not have a correspondence in the business logic.

In addition to this solution we propose two extensions on the trigger's functionalities, features that do not exist in the actual DBMSs or in the actual SQL standard:

- a new option for triggers to postpone them until the end of the database transaction (just as a check constraint may be delayed with the DEFERRABLE option). In the example above, such a DEFERRABLE trigger should be implemented on the INSERT event of parent table (INVOICES).

- a new trigger event to execute a stored routine after the end of the database transaction (TRIGGER ON COMMIT) and to be able to track whether an individual event took place in the current database transaction. This solution would have the disadvantage that such trigger would be too often executed.

4.6 A standard warning and error management system

The business rules imposed with stored procedures may use errors and warnings. The error stops the existing data processing and does not let the users to break a rule. The warning lets the user to insert some data that break a business rule, but informs him/here about this. In business information systems, when data has to be stored about business transaction that took place, the data has to be saved into the database even if some business rules failed. For these situations the entire warning mechanism affects the application design without any help from the database dictionary and the database built in packages. That is way we propose to extend the error management system with warnings that do not stop the database operations.

4.7 A new DBA role

In the classical systems the DBA roles works at the instance level on all the schemas of a database. In the cloud databases, beside the DBAs of the cloud database, there should be also the role of the tenant's DBAs.

4.8 Stored procedure as a service

Since the stored procedures are also software, the model software as a service may be customized for stored procedures.

The functionality of the stored procedures should be subject of a service, not only between the cloud owner and its tenants, but also between tenants.

The cloud database and its dictionary should ensure the architecture that makes possible stored procedures as a service.

5. CONCLUSIONS

Stored procedures are now widely used in relational and object-relational DBMS market. The main reasons for this spread are their advantages and the need for computational completeness in the database environment.

Thanks to stored procedures, the database servers (in cloud or not) are not only data storage resources, but also data processing and data publishing resources.

Further researches relate to the changes that had to be done to stored procedures for their migration into clouds. The cloud migration of the stored procedures implies certain changes required by the new security challenges that come from sharing the same IT resources. These changes were shortly presented in a recent Oracle article [12].

6. ACKNOWLEDGMENTS

This work was cofinanced from the European Social Fund through Sectoral Operational Programme Human Resources Development 2007-2013, project number POSDRU89/1.5/S/56287 „Postdoctoral programs at the forefront of the excellence research in the technologies of the informational society and innovative product and process development", partner University of Oradea.

7. REFERENCES

[1] Ben-Gan, I., Sarkaand, D., Wolter, R., 2006 Inside Microsoft SQL Server 2005: T-SQL Programming, Microsoft Press

[2] Curino, C., Jones, E. P. C., Popa, R. A., Malviya, N., Wu, E., Madden, S., Balakrishnan, H., Zeldovich, N. 2011 Relational Cloud: A Database-as-a-Service for the Cloud, CIDR, pages 235–241 http://people.csail.mit.edu/nickolai/papers/curino-relcloud-cidr.pdf

[3] Date, C.J. 2004 An Introduction to Database Systems, 8th Edition, Pearson Education.

[4] Faroult, S., Robson, P. 2006 The Art of SQL, O'Reilly, subchapter. 2.15.

[5] Feuerstein, S., Nanda, A., 2005 Oracle PL/SQL for DBAs, O'Reilly

[6] Feuerstein, S., Pribyl, B., 2005 Oracle PL/SQL Programming, O'Reilly

[7] Fotache, D., Hurbean, L., Dospinescu, O., Pavaloaia, V. 2010 Procese organizaţionale şi integrare informaţională. Entreprise Resource Planning, Ed. Universităţii "Al.I.Cuza" Iaşi

[8] Fotache, M. 2005 Proiectarea bazelor de date. Normalizare şi postnormalizare. Implementări SQL şi Oracle, Editura Polirom, Iaşi

[9] Moore, S., Belden, E. 2009 Oracle Database PL/SQL Language Reference 11g Release 1 (11.1), Oracle

[10] Velte, A. T., Velte T. J., Elsenpeter, R. 2010 Cloud Computing: A Practical Approach, McGraw Hill http://neuron.csie.ntust.edu.tw/homework/100/CC/materials/Cloud%20Computing%20-%20A%20practical%20Approach.pdf

[11] *** 2011 ISO/IEC 9075-1:2011 Part 1: Framework (SQL/Framework) Working Draft Documents http://www.wiscorp.com/sql20nn.zip

[12] *** 2012 Oracle Database Cloud Service security lockdown, an Oracle White Paper, http://www.oracle.com/technetwork/database/database-cloud/public/dbcs-security-lockdown-1844133.pdf