Software Engineering Issues in RDBMS, a Preliminary Survey

Julien Delplanque

Software Engineering Issues in RDBMS, a Preliminary Survey

Julien Delplanque
Université de Lille, CRIStAL, CNRS, UMR 9189, Lille, France
RMoD Team, Inria Lille Nord Europe
julien.delplanque@inria.fr

***Abstract*—Relational database management systems (RDBMS) come with many features. Tables and columns but also views, triggers or stored procedures. Some database architects (DBAs) feel a lack of tools to help them during the implementation and the evolution of the database. In this paper, we conducted a preliminary survey in the PostgreSQL community to get an idea of the usage they make of some of the PostgreSQL features and to gather feedbacks of the issues they face using these features. The results of this survey suggest that these features are fairly well used by DBAs and that DBAs using them encounter implementation issues.**

I. Introduction

Relational databases [1] have existed for several decades, but the management of their evolution seems to remain complex. A database architect contacted our team to report issues he had while managing his relational databases. These issues were initially related to the implementation of views (*i.e.*, named and stored select query). When a view has to be modified, it has to be deleted and then recreated. The problem is that if this view is used by another view, the dependant view has to be deleted and recreated as well. When you have a lot of views depending on each others, it becomes hard to modify them. Indeed, the problem can be solved by hand but depending on the number of dependent views it can be very complex and error prone. By further questioning the DBA, we realised that he was facing other issues related to stored procedures (*i.e.*, set of SQL instructions pre-compiled and available to the database entities and potentially to applications accessing the RDBMS) or triggers (*i.e.*, a set of SQL instructions to execute when a specific event occurs in the database).

A study of commercial solutions for RDBMS management suggested us that a "database approach" is adopted by those solutions. That is to say, they provide utilities to create/modify/destroy entities in a database and to do a dependency analysis based on the meta-data provided by the RDBMS. There are three main problems with this approach. First, the dependencies concerning stored procedures are unknown by these tools because their source code are stored as strings in the RDBMS meta-data. Second, these tools warn the user that a dependency makes a change impossible but do not propose a solution to allow the change to be integrated. Finally, no visualisation of these dependencies is provided to database architects which leads them to build the dependency graph manually in order to evaluate the impact of the change to apply.

These issues in RDBMS tools make the maintenance of databases (DBs) containing a lot of entities difficult for DBAs. The lack of support for automated or semi-automated change integration in the database makes the maintenance even more difficult. Before addressing these problems, we want to explore if these issues are faced on the field by other DBAs and if other issues are met. In this paper, the results of a twofold preliminary survey are presented. This survey aims to: 1) provide a coarse-grained estimation of these usages by questioning DBAs directly and 2) get feedback of implementation issues encountered by DBAs in order to be able to analyse their causes and try to find automatic or semi-automatic solutions.

II. Pre-Survey in PostgreSQL Lille Community

In order to have a first idea on the usage of views, stored procedures and triggers in companies and a first idea of problems DBAs are facing while using these entities, we conducted a preliminary survey in a meeting of the PostgreSQL Lille community. This section describes the protocol used to conduct the survey and presents our findings.

A. Context and Participants

PostgreSQL Lille community's meeting was organized by two companies in a bar, which provided a casual atmosphere. The meeting's schedule consisted in two 45 minutes presentations with a pause of 15 minutes between. About 50 people participated to the meeting from which 13 were asked to answer the survey during the pause and after the conference. No compensation was provided in exchange to the survey answer. 1 person refused to answer the survey claiming that he had not enough experience in database administration/architecture. Thus, 12 participants answered the survey during this meeting.

B. Survey's Description

The survey was written in French (French being the native language of the meeting's participants). It consisted of 16 questions. 12 questions concerning the usage of views, stored procedures and triggers in databases and 4 concerning personal information of participants.

The 12 questions can be presented as 3 similar groups of 4 questions. With **X** being one of the words "views","stored procedures" or "triggers", the four questions were:

1) What is the proportion of your databases that contain **X**?
2) How many **X** do they hold compared to the number of tables?
3) Are your new databases using **X**? If not, why?
4) Do you encounter problems while implementing **X**? If so, what problems?

Questions 1 and 2 were multiple choice questions with, as propositions, *0-20%*, *20-40%*, *40-60%*, *60-80%* and *80-100%*. Question 3 was a *Yes/No* question with the possibility to explain the reason why new databases are not using **X**. Question 4 was a *Yes/No* question with the possibility to explain the problems encountered during the development of **X**.

The 4 personal questions were the following:

1) In which company are you working? Or in which open-source project?
2) What is your position?
3) How long have you been in this position?
4) How many years of experience do you have with PostgreSQL?

For each question the participant could answer freely what resulted in various different answers.

C. Results and Interpretation

Figure 1 illustrates the results of usage question 1: "What is the proportion of your databases that contain **X**?". It seems that, for each type of entity, most of the interviewees perceive that only 0-20% of their database(s) use them. 6 participants use views in 0-20% of their DBs while the 6 others use them in intervals included in 20-100%. A similar observation can be done for stored procedures with 7 participants using them in 0-20% of their DBs and 5 in intervals included in 20-100%. For triggers, 6 people said that 0-20% of their DBs use triggers, 5 said 20-40% and 1 said 80-100%. These results suggest that views, stored procedure and triggers are not always used in database systems. This observation seems normal, some applications do not require other features than tables and columns from the DB .

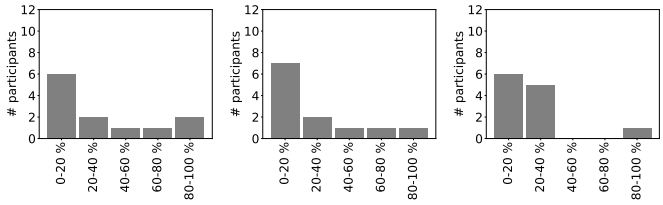
The results of the second question, about the proportion of views/stored procedures/triggers compared to the number of tables in the database, are illustrated in Figure 2. The interviewees answers suggest that, generally, the proportion of views, stored procedures, triggers against the number of tables is in the interval 0-20%. For triggers, there are more answers (4/12) for the interval of 20-40%. The data collected suggest that views, stored procedures and triggers usually represent a proportion of 0-20% relatively to the number of tables.

Figure 3 summarizes the results of questions 3 and 4. As a reminder, question 3 asked participants if they use views/stored procedures/triggers in their new databases and question 4 asked participants if they have problems while implementing those entities. In the plot, people using one of these entities and having problems are illustrated as the black part of the "yes" bar. Figure 3 suggests that all these entities are used by a significant part of the survey's participants. Some of those users encounter problems during the implementation. Stored procedure is the type of entity that causes the most problems to its users (3/7 have problems during implementation).

Questions 3 and 4 let the possibility to the participants to explain their answer (depending if it was "yes" or "no"). Some people gave a justification to their answer even if it was not required and others did not provide a explanations when it was required. Table I presents the main ideas of participant's explanations. These explanations have been summarized to be able to group those expressing the same idea.

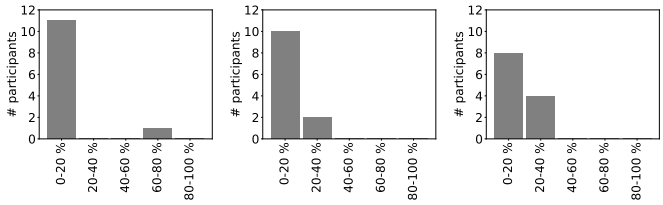
For questions about the usage of views/stored procedures/triggers, the most common answer is that the usage of these kinds of entities is not required by the project(s). The other reasons got the same scores for each kind of entity excepts for triggers where 2 people reported performance issues to justify their answer. Explanations summaries of answers about implementation issues mostly have a single occurrence except for the "programming language complexity" reported twice for stored procedures.

[1]https://www.postgresql.org/docs/current/static/sql-notify.html



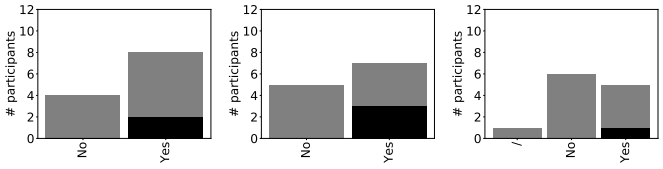
(a) Views (b) Stored procedures (c) Triggers

Fig. 1. Interviewees' perception of the proportion of their databases using views/stored procedures/triggers.



(a) Views (b) Stored procedures (c) Triggers

Fig. 2. Interviewees' perception of the proportion of views/stored procedures/triggers in their databases relatively to the number of tables.



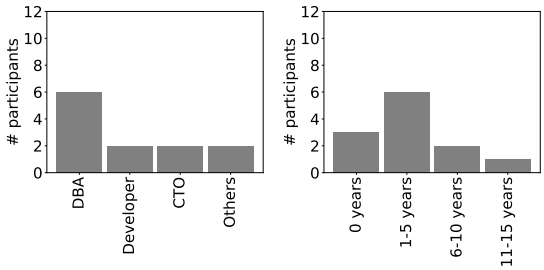
(a) Views (b) Stored procedures (c) Triggers

Fig. 3. Interviewees' perception of the usage or not of views/stored procedures/triggers in their databases. In the interviewees that use views/stored procedures/triggers, the black part of the bar represents those having problems.

TABLE I
SUMMARIES OF PARTICIPANTS' EXPLANATIONS TO QUESTIONS 3 AND 4 FOR VIEWS, STORED PROCEDURES AND TRIGGERS.

Answer	Justification summary	#
Views usage		
No	Not required by the project(s).	2
No	No advantage compared to a simple request.	1
No	Not well managed by ORMs.	1
No	Work made in application side	1
Yes	Developers' lack of awareness.	1
Yes	To fool the ORM.	1
Yes	For materialized views.	1
Views implementation issues		
Yes	Problem when altering a table used by a view.	1
Yes	Developers' lack of knowledge.	1
Yes	Views hide the schema complexity but not model's problems.	1
Stored procedures usage		
No	Not required by the project(s).	3
No	No advantage compared to simple requests.	1
No	Functional aspect not wanted.	1
Yes	Allows automation of tasks.	1
Yes	Permissions management.	1
Stored procedures implementation issues		
Yes	Programming language complexity	2
Yes	Maintenance complicated.	1
Triggers usage		
No	Not required by the project(s).	3
No	Performance issues.	2
No	Maintainability problems.	1
No	Brings a logic aspect in DB side.	1
Yes	For partitioning the database.	1
Yes	With pg_notify[1] extension.	1
Triggers implementation issues		
Yes	Performance issues.	1

Figure 4 illustates profiles of participants. Most of them are claiming to be DBAs (6/12), there are also 2 developers, 2 CTOs and 2 who could not be grouped (1 geomatics specialist and 1 claiming to be leader of PostgreSQL community). 6 participants have 1-5 years of experience with PostgreSQL. 2 have 6-10 years of experience and 1 has 11-15 years. There are 3 participants who have 0 years of experience. One of them just started to use PostgreSQL 1 month ago and the 2 others are Oracle users. These Oracle users were able to answer questions on views, stored procedures and triggers since Oracle RDBMS offers these features too.



(a) Participants' jobs (b) Experience with RDBMS

Fig. 4. Profiles of the participants.

D. Threats to Validity

The results of the survey are based on the participants' perception of databases they maintain. Because of that, there are factors that limit the validity of our observations.

First of all, the perception of DBAs on the number of entities in their databases may be wrong. Indeed, we do not know what is the difference between their perception of database sizes and the reality. Nevertheless, answers to questions concerning the proportion of databases using a particular entity type and the proportion of a particular entity type in their database schemas provide the informations that views, stored procedures and triggers are used in companies' databases and that this usage can be of different degrees.

Second, the proposed percentage intervals provide a poor precision on the real proportion of entity types. Indeed, we realised that it is particularly awkward for the "0-20%" interval. A lot of people gave this answers and it is not possible to know if they mean 0% or between 1% and 20%. This is an issue of the survey that was not detected during its conception because of our expectations on the results (we thought the intervals selected would be higher). However, answers to question 3 tend to reduce this threat since the number of people who are not using views/stored procedures/triggers in their DBs is lower than those providing the 0-20% interval as answer to question 1.

Third, concerning the justifications to questions 3 and 4, we realised that a survey may not be the best way to ask DBAs about issues they encounter while implementing views, stored procedures and triggers. Indeed, only a few participants provided implementation issues but these issues should, if they actually occur, be experienced by the other DBAs. We believe that maybe the other DBAs experience these issues but did not think about it while answering the question. In hindsight, we think that the question can be quite difficult to answer on paper and maybe using a focus group method [4] would be more appropriate to get information from DBAs experience.

Finally, the survey was conducted on a small sample of database users (12 participants). A larger scale survey is required to confirm our results.

III. Related Work

Lu et al. [2] did a survey covering various aspects of the usage of SQL in industrial organisations. In particular, the authors studied the complexity of SQL queries, the usage of various SQL features, the difficulties encountered and the actions taken to solve those difficulties. Statistics about each aspect are provided.

Pönighaus [3] did an empirical study to assess the "coding frequency" of SQL statements. They analysed the use of the four "data manipulation statements" (namely `SELECT`, `INSERT`, `UPDATE` and `DELETE`) from database request modules in 4 proprietary databases using the RDBMS *IBM-DB2 V2R2* . To retrieve these statements, they queried the DB's catalog and processed the result in order to do a statistical analysis of statements usages.

These two articles provide a work similar to ours but at the "statement level" rather than at the "entity level".

IV. Conclusion

In this paper, we conducted a survey on the PostgreSQL community in order to 1) get an idea of the extent of their usage of views, stored procedures and triggers and 2) get a feedback of implementation issues they encounter.

The results of this survey provide that,

- Views, stored procedures and triggers are usually used in 0-20% of their databases;
- The number views, stored procedures and triggers usually represent 0-20% of the number of tables in their databases;
- 7/12 people use stored procedures and 3 of them have problems during the implementation;
- "Programming language complexity" and "maintenance complicated" issues are reported for stored procedures; and
- "Performance issues" is reported for triggers.

The answers given by participants need to be further analysed but they suggest that there is a need for tools support to help DBAs during DB implementation.

V. Future Work

Future work is threefold. First, a review of the literature has to be made to determine if the problems reported by DBAs have been solved but are not integrated in commercial solutions for RDBMS management or if these problems are still unsolved.

Second, it could be interesting to see if the usage results are generalisable on a greater sample of participants. To test this, a larger scale study could be realised (taking into consideration the issues of the pre-survey). Also, as highlighted previously, asking participants about issues encountered during implementation in a survey may not be suitable. Another possibility would be to use the focus group method. By asking these questions to a group of DBAs able to interact with each others, it may be possible to collect richer answers.

Finally, analysing DB schemas directly to extract the number of view(s), stored procedure(s), trigger(s), etc... would provide more solid results by avoiding DBA's possible misconceptions on DBs. Furthermore, such analysis would allow us to compare open-source and proprietary DB schemas concerning the usage of such entities.

References

[1] Edgar F Codd. A relational model of data for large shared data banks. *Communications of the ACM*, 13(6):377–387, 1970.

[2] Hongjun Lu, Hock Chuan Chan, and Kwok Kee Wei. A survey on usage of sql. *ACM SIGMOD Record*, 22(4):60–65, 1993.

[3] Richard Pönighaus. Favourite sql-statements - an empirical analysis of sql-usage in commercial applications. *Information Systems and Data Management*, pages 75–91, 1995.

[4] Forrest Shull, Janice Singer, and Dag IK Sjøberg. Guide to advanced empirical software engineering. chapter 4, pages 93–116. Springer, 2008.