The Usage of Declarative Integrity Constraints in the SQL Databases of Some Existing Software

Erki Eessaar

Department of Software Science,

Tallinn University of Technology,

Estonia

erki.eessaar@taltech.ee

Outline

- Background and goal.
- How we searched the occurrences of database design problems.
- The results.
  - Statistics.
  - Some examples.
- Conclusions and future work.

Declarative integrity constraints in the SQL Standard

- PRIMARY KEY
- UNIQUE
- FOREIGN KEY
- NOT NULL
- CHECK
- Setting the type and maximum field size of a column (is not covered in this research)

A DBMS may lack some of these or offer some extra – for instance, exclusion constraint in PostgreSQL.

Advantages of declaring integrity constraints in the database

- Prevents registration of **certainly incorrect** propositions.
    - Later costly, difficult, or impossible to fix.
- Gives information about the **meaning of data** to human observers and various programs – including the DBMS itself.
    - Query optimization, generation of application code and test data, finding design problems, etc.

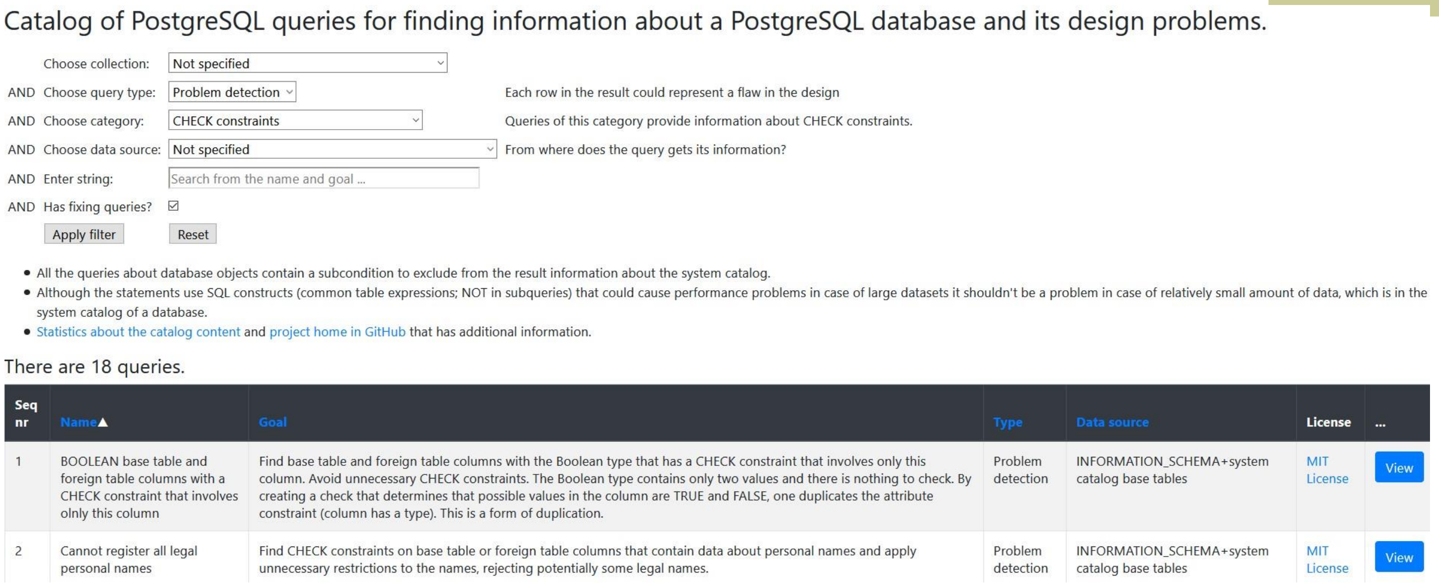
Advantages of declaring integrity constraints in the database (2)

- Simplifies **application development**.
  - Code does not have to be duplicated.
  - Integrity checks in applications could fail in case of *concurrent* use of the same data elements.
- Cannot be **bypassed**.

The goal

- Investigate how well the integrity constraints are implemented in existing SQL databases.
  - Existing case studies are old or report a small number of problems.

The catalog



Catalog of PostgreSQL queries for finding information about a PostgreSQL database and its design problems.

	Choose collection:	Not specified	
AND	Choose query type:	Problem detection	Each row in the result could represent a flaw in the design
AND	Choose category:	CHECK constraints	Queries of this category provide information about CHECK constraints.
AND	Choose data source:	Not specified	From where does the query gets its information?
AND	Enter string:	Search from the name and goal ...	
AND	Has fixing queries?	☑	

- All the queries about database objects contain a subcondition to exclude from the result information about the system catalog.
- Although the statements use SQL constructs (common table expressions; NOT in subqueries) that could cause performance problems in case of large datasets it shouldn't be a problem in case of relatively small amount of data, which is in the system catalog of a database.
- Statistics about the catalog content and project home in GitHub that has additional information.

There are 18 queries.

Seq nr	Name▲	Goal	Type	Data source	License	...
1	BOOLEAN base table and foreign table columns with a CHECK constraint that involves olnly this column	Find base table and foreign table columns with the Boolean type that has a CHECK constraint that involves only this column. Avoid unnecessary CHECK constraints. The Boolean type contains only two values and there is nothing to check. By creating a check that determines that possible values in the column are TRUE and FALSE, one duplicates the attribute constraint (column has a type). This is a form of duplication.	Problem detection	INFORMATION_SCHEMA+system catalog base tables	MIT License	View
2	Cannot register all legal personal names	Find CHECK constraints on base table or foreign table columns that contain data about personal names and apply unnecessary restrictions to the names, rejecting potentially some legal names.	Problem detection	INFORMATION_SCHEMA+system catalog base tables	MIT License	View

https://github.com/erki77/database-design-queries

We have developed a large set of PostgreSQL system catalog-based queries for searching database design problems of PostgreSQL databases.

The catalog (2)

- Many of the queries directly point to problem occurrences.
  - Mistakes.
  - Design smells.
    - Will cause later maintenance problems.
- Each such query documents a design problem.
  - The absolute majority of these could appear in the databases of any SQL DBMS.

- Long development history, still actively used
- Use a PostgreSQL database

The analysis – databases

- FusionForge
  - An open source development management and team collaboration software.
  - **206** base tables (tables from now on) and **1097** columns.
  - Development started in **2001**.
- LedgerSMB
  - An open source enterprise resource planning software.
  - **162** tables and **978** columns.
  - Development started in **2006**.

The analysis – databases (2)

- OTRS Community Edition
  - An open source ticketing software, which can be used to track and manage issues that need resolving.
  - **116** tables and **962** columns.
  - Development started in **2001**.
- Stansoft
  - A Linux financial accounting software.
  - **174** tables and **1931** columns.

Results

- We presented **36** problems with declarative database integrity constraints.
  - Uniqueness constraints (**14** problems).
  - Foreign key (FK) constraints (**13** problems).
    - Compensating actions of FK constraints (**5** problems).
  - CHECK constraints (**5** problems).
  - NOT NULL constraints (**4** problems).

Results (2)

- We **only** listed the problems that had at least one occurrence in one of the databases.
- The listed problems *generalize* smaller related problems.
  - Collection “Find problems about integrity constraints”, which contains all the used queries, has **69** problem detection queries.
  - Each corresponds to a sign of the problems.

Results (3)

- Related works resources (**12 papers**) name *only* **12** of the design problems (**33.3%**).
- The most often reported problems in the resources are:
  - missing foreign key constraints (in **11** resources),
  - specifying a list of values for a column (**6** resources),
  - missing primary keys (**5** resources).

Results (4)

- In the existing resources
  - there is no discussion about **compensating actions**,
  - very little is said about **check constraints**.
- Using system catalog-based queries for problem detection has good *performance*.
  - Execution of all the queries from the collection “Find problems about integrity constraints” took about **25 seconds** per database.

An example – uniqueness constraints

- A table does not have any primary key and unique constraints.
  - Stansoft 100% (174), OTRS 25% (29), FusionForge 24.3% (50), LedgerSMB 1.9% (3).
- A table does not have any means for preventing duplicate rows.
  - Stansoft 32.8% (57), FusionForge 19.9% (41).
    - Stansoft used unique indexes as the only database-level mean for ensuring uniqueness.

An example – foreign key and check constraints

- Missing foreign key constraints.
    - Tables that do not participate in any foreign key constraint – neither as the referencing table nor as the referenced table.
        - Stansoft 100% (174), FusionForge 31.6% (65), OTRS 15.5% (18), LedgerSMB 4.9% (8).
- Only LedgerSMB had any check constraints.
    - Even there many were missing.

Conclusions

- All the analyzed databases had **many problems** with integrity constraints.
  - Lack of constraints.
  - Inconsistent definition of constraints.
- Thus, users of data lack a vital source to understand the **meaning** of data.
- System catalog-based queries are **effective** in finding database design problems.

Future work

- Analysis of existing databases in terms of other classes of problems.
  - Conceptual database schema (base tables).
  - External schemas (derived tables, user-defined routines).
  - Internal schema (for instance, indexes).
  - Naming of database objects.

Thank you for your attention!

Questions?

Erki Eessaar – erki.eessaar@taltech.ee

Reference to the catalog:
https://github.com/erki77/database-design-queries

- Collection “Find problems about integrity constraints”.