TBILISI STATE UNIVERSITY

Faculty of Exact and Natural Sciences

Thesis Overview

"*DATABASE NORMALIZER AND TUNER*"

Student: Tigran Budaghyan

Professor: Manana Khachidze

Tbilisi 2014

Relational Database Problem…

Although relational databases provide the best mix of simplicity, stability, flexibility, performance, scalability and compatibility, their performance on each of these points is not enough higher than on similar systems, focused on any single feature. It was not a big problem because universal domination of RDBMS outweighed any shortcomings. However, if ordinary RDBs do not meet the needs, there are always an alternative.

Today, the situation is slightly different. Variety of applications is growing, and with it grows the importance of these features also. And with the growing number of databases, one feature begins to overshadow all others. This is scalability. As more applications running under high load conditions, such as Web services, their scalability requirements can change very fast and grow strong. The first problem can be very difficult to resolve, if we have a relational database located on our own server. Suppose the load on server grows tree times at night. How fast can we upgrade hardware? Solution to the second problem also causes difficulties in case of relational databases.

Relational database scales well only if it is placed on a single server. When the resources of the server run out, we will need to add more machines and distribute the load between them. And in this case the complexity of relational databases starts playing against scalability. If try to increase the number of servers not up to several but hundreds or thousands, complexity increases by an order, and the characteristics that make relational databases so attractive, rapidly reduce to zero chances to use them as a platform for large-scale distributed systems..

To remain competitive, vendors of cloud services have to somehow deal with this limitation, because cloud platform without scalable data storage is useless. Therefore, if vendors want to provide users' scalable data storage, they have only one option. They need to use other types of databases, which have a high ability to scale, even for the cost of other opportunities which are available in relational databases.

These advantages, as well as the existing demand, led to a wave of new database management systems.

Designing relational databases

By designing a database we solve two major problems:

• How to display the domain objects in the abstract data model objects? This is a logical database design.

• How to ensure the efficiency of querying the database? It's a problem of physical database design.

In the case of relational databases is difficult to imagine any general recipes of the physical design. Because it too much depends on the DBMS. Therefore, we only focus to the logical design of relational databases, which are essential when using any relational database.

Consider that the problems of designing a relational database are confined in the following questions:

• What relationships should have a database?

• What attributes should have these relationships?

• Definition of primary and foreign keys and constraints.

Relational database design using normalization

We consider the classical approach in which the entire design process is carried out in terms of the relational data model, by consecutive approximations to a satisfactory set of relational schemes.

Subject domain - is part of the real world, the data about which we want to reflect in the database. For example, as a subject domain, we can select any enterprise accounting, human resources, bank, shop, etc. Subject domain is infinite, and contains both the essential concepts and data as well as unimportant or no meaningful data. So, if as domain account to choose products in stock, then the concept of "waybill" and "invoice" are essential concepts, and that the employee who receiving invoices, has two children - it does not matter to account for the goods. However, from the standpoint of personnel data on the presence of children are essential. Thus, the importance of the data depends on the choice of the subject domain.

The starting point of the design is the representation of subject domain in the form of one or more relationships. The design process is a process of normalization of relations schemes, and each “next” normal form has better properties than the previous one.

RELATIONAL DATABASE NORMALIZER AND TUNER

A few words about….

"DATABASE NORMALIZER AND TUNER" is a Windows Application - complex database analyzer, to facilitate the work of database and system administrators. How does it work?

Short Description: A smart tool for relational databases which connects to given database and analyzes it. It finds anomalies, redundancies in relational databases (normalization part) and analyze system configurations, buffers and caches of database instance, security vulnerabilities, fragmentations, indexes (tuning part). As a result gives recommendations for database structure, sizes configured for buffers and caches etc., and generates SQL scripts for normalization, giving possibility to database administrators to decide which solutions are "useful" and which not for them, also to make modifications on scripts before executing them.

1. DATABASE NORMALIZER

ISSUE

Anomalies and redundancies in relational databases make database slow-working.

SOLUTION

Normalize database at least in 3NF or Boyce-Codd Normal Form to bypass anomalies and redundancies.

CONDITIONS

1) 1NF - all relational databases are in 1NF.

2) 2NF - all non-key columns determined only by the entire primary key.

3) 3NF - 2NF + there is no non-key column determined by another non-key column.

4) BCNF - 3NF + every determinant is a candidate key.

REALIZATION

1) Nothing to do.

2) Steps for normalizing in 2NF are:

1. Select all non-key and PK composite column couples turn by turn and filter by each row of PK composite column, and ensure that in result come out only identical rows, which will mean that those non-key columns are determined by a part of PK.

2. Create a new table with that non-key and key column and delete that non-key column.

3. Set PK on key-column in new table and Foreign Key on appropriate column on old table.

(See "Example 1")

3) Steps for normalizing in 3NF are:

1. Select all non-key column couples turn by turn and filter by each field, and ensure that in result come out only identical rows.

2. Create a new table with those non-key columns.

3. Set PK on one column in new table and FK on appropriate column on old table.

(See "Example 2")

4) Steps for normalizing in BCNF are:

1. Find all possible determinants by selecting all possible column pairs, and check if all rows are unique.

2. Find if there are overlapping candidate keys (there is at least a column that is a part of two or more candidate key).

3. Create a new table with a non-overlapping columns, set PK on one of them and FK on appropriate column on old table.

(See "Example 3")

Example 1

ID	Activity	Fee
1	Football	60
1	Swimming	70
2	Basketball	55
2	Football	60
3	Swimming	70
3	Golf	100
4	Football	60
5	Golf	100
6	Cricket	80
7	Baseball	90
7	Cricket	80

ID	Activity
1	Football
1	Swimming
2	Basketball
2	Football
3	Swimming
3	Golf
4	Football
5	Golf
6	Cricket
7	Baseball
7	Cricket

Activity	Fee
Football	60
Swimming	70
Basketball	55
Golf	100
Cricket	80
Baseball	90

Example 2

StudentID	Hostel	Fee
11	Regie	700
13	Regie	700
14	Obor	400
15	Regie	700
17	Belvedere	1000
20	Obor	400
21	Leu	600
22	Dristor	500
23	Colentina	300
25	Belvedere	1000
28	Ase	1200
29	Leu	600
30	Colentina	300
31	Dristor	500

StudentID	Hostel
11	Regie
13	Regie
14	Obor
15	Regie
17	Belvedere
20	Obor
21	Leu
22	Dristor
23	Colentina
25	Belvedere
28	Ase
29	Leu
30	Colentina
31	Dristor

Hostel	Fee
Regie	700
Obor	400
Belvedere	1000
Leu	600
Dristor	500
Colentina	300
Ase	1200

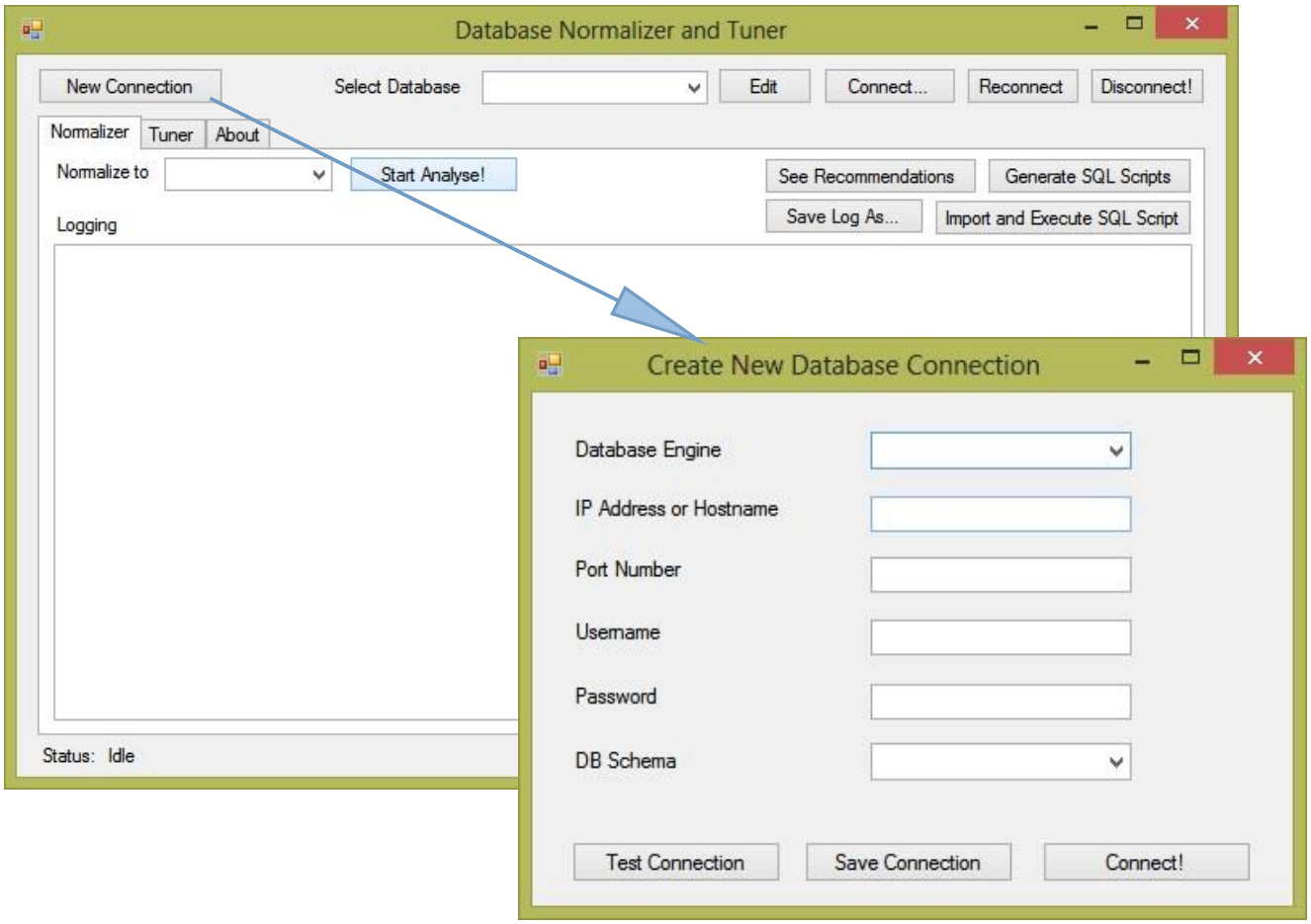
Example 3

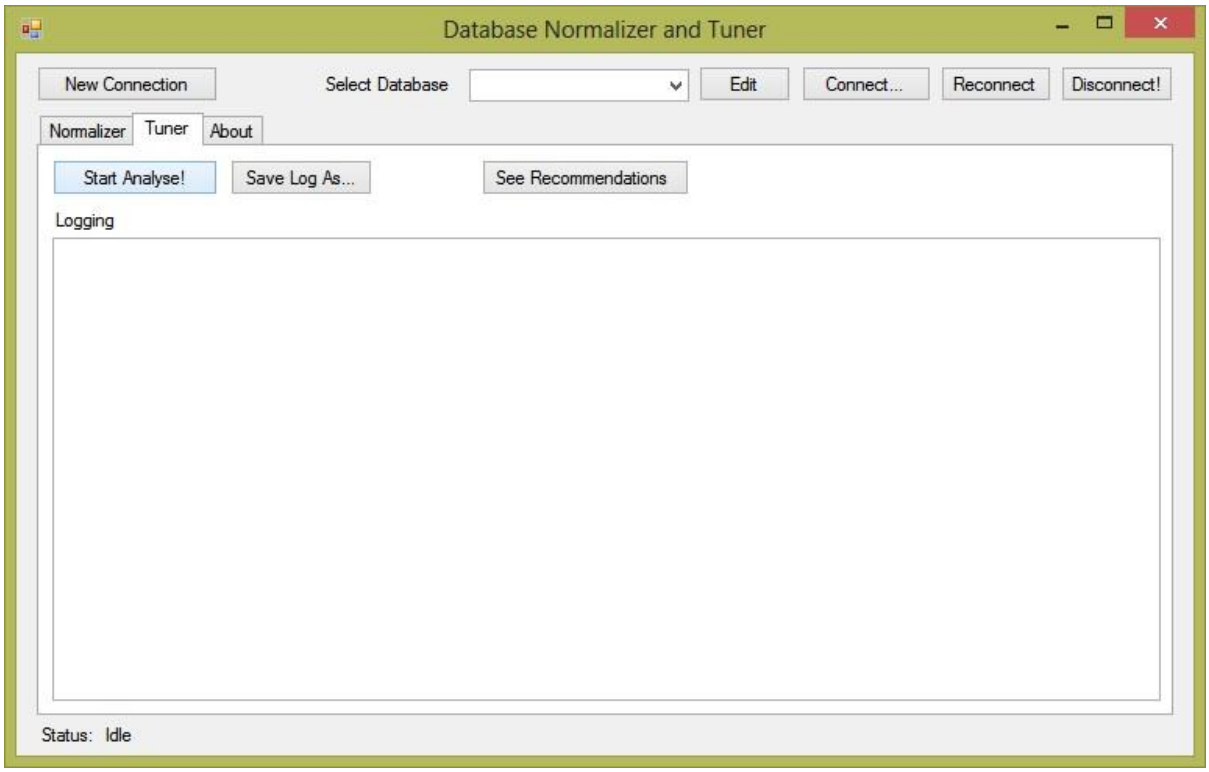
StudentID	Subject	Professor
10	Physics	Ravens
11	Informatics	Bogdan
12	Math	George
12	Physics	Michael
13	English	Alexander
14	Math	Rodger
14	English	Mark
15	Physics	Ravens
15	English	Diana
15	Math	George
16	French	Ekaterina
17	Informatics	Razvan
17	HR	Herisanu

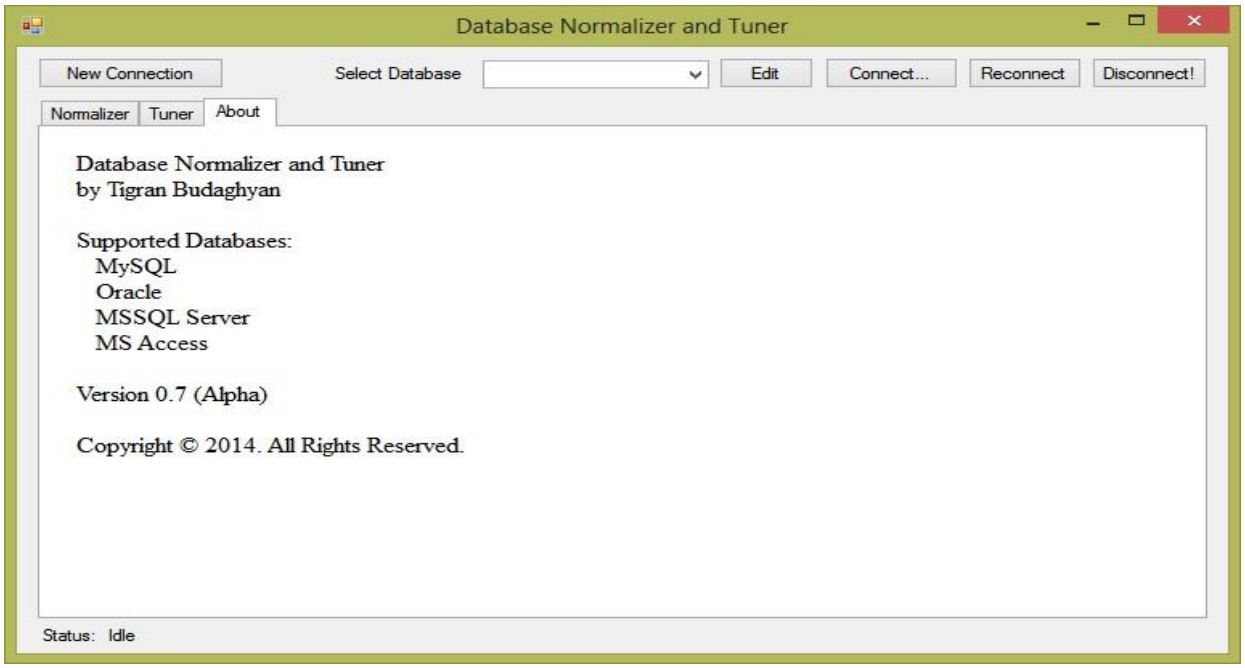
StudentID	Professor
10	Ravens
11	Bogdan
12	George
12	Michael
13	Alexander
14	Rodger
14	Mark
15	Ravens
15	Diana
15	George
16	Ekaterina
17	Razvan
17	Herisanu

Subject	Professor
Physics	Ravens
Informatics	Bogdan
Math	George
Physics	Michael
English	Alexander
Math	Rodger
English	Mark
English	Diana
French	Ekaterina
Informatics	Razvan
HR	Herisanu

APPLICATION INTERFACE







2. DATABASE TUNER

ISSUE

Wrongly configured database engines which can result performance and security issues.

SOLUTION

Find all misconfigurations and drawbacks on database instances and suggest them corrections.

REALIZATION

1. List all users from database system table and see if all users have passwords assigned (give warning if no password assigned) and list all of theirs permissions and roles.
2. List all buffers and caches limits configured and all database usages, then give recommendations which one to increase or decrease.
3. Find which operations are performed without indexes and give recommendations how to avoid index-bypassing.
4. Read hardware and system configurations and recommend what changes can improve performance.
5. Find database fragmentations and give solutions (for instance run OPTIMIZE TABLE).