Database Design Document

A quick rundown of why RethinkDB was chosen as well as the thought process behind the design of the database.

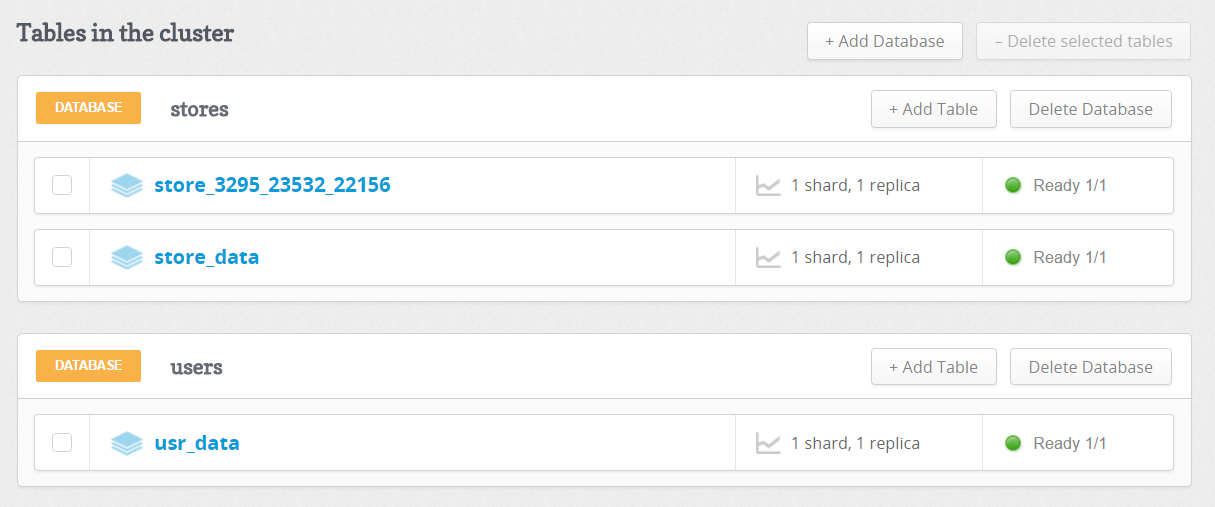
Why Was RethinkDB Chosen?

While considering what features that could be implemented to provide a better experience for customers and more incentive for store managers, RethinkDB presented itself as an option for scalable pushing. As in, whatever updates happen to the database, this information is pushed out into the clients, automatically. This would be very useful for the way the feed system for the customer works, as well as allowing Real-time statistics for the store manager.

However, this database isn’t well suited for Analytical data and processing. So additional trend data will require the database to be converted to another database. A process which can be automated by a weekly backup.

Thought Process Behind Design

The nature of the RethinkDB allows for tables to quickly be made programmatically. Which allows us to quickly section off our store tables within the same store database. This allows us to keep individual store data clean and concise upon lookup. The tables will follow the naming scheme of “store\_????\_????\_????”, where the ? Represent the uniquely generated ids, that the system creates.



Database Mockup Schema

This is a very basic implementation for the database. Which will serve only to get us up and running. The Schema is listed like this because the Database uses Documents.

Users Database => usr\_data table

id

username (could\_be\_empty)

email (could\_be\_empty)

password (could\_be\_empty)

google\_oauth\_token (could\_be\_empty)

Stores database => store\_data

id

store\_id

store\_name

store\_manager

store\_tbl (this will have the tbl name for the store’s additional data)

Stores Database => store\_????\_????\_????

id

broadcast\_id

broadcast\_time

broadcast\_timerange

broadcast\_message

broadcast\_blob