International Journal of Computing and Business Research (IJCBR)
ISSN (Online) : 2229-6166
International Manuscript ID : 22296166V8I1201804
Volume 8 Issue 1 January - February 2018

RethinkDB as a High Performance NoSQL Engine for Real-Time Applications

Inderpreet Boparai
Assistant Professor
Department of Computer Applications
Chandigarh Group of Colleges (CGC)
Landran, Punjab, India

Abstract

With the increasing traffic of heterogeneous as well as unstructured data on the World Wide Web (WWW), the traditional database engines are having enormous issues related to schema management, concurrency, database integrity, security, parallel read-write operations, resource optimization and many others. Now days, the real-time applications are having huge traffic from different channels including Social Media, Mail Groups, Satellites, Internet of Things (IoT) and many others. These types of traffic are generally unstructured and huge which cannot be handled by classical Relational Database Management Systems (RDBMS). To cope up with such issues of Big Data which include Velocity, Volume and Variety (3Vs of Big Data), the advent of Not Only SQL (NoSQL) came to existence which is meant for handling the unstructured and heterogeneous data with higher performance. This manuscript underlines the assorted perspectives with the working environment of RethinkDB as a high performance NoSQL database engine in the cloud as well as other working environment.

Keywords : Cloud Database, NoSQL Database, Real-Time Database, RethinkDB

Introduction

There are different types of NoSQL databases [1] in diversified categories for specific applications so that the higher degree of performance can be accuracy can be achieved [2].

Taxonomy	NoSQL Database
Document Store	ArangoDB, Couchbase, BaseX, Clusterpoint, CouchDB, DocumentDB, MarkLogic, IBM Domino, MongoDB, , RethinkDB Qizx

Key-Value Store	ArangoDB, Keyspace, Flare, Aerospike, RAMCloud, SchemaFree, quasardb
Key-Value Store	Actord, Lightcloud, FoundationDB, InfinityDB, LMDB, Luxio, NMDB, MemcacheDB, TokyoTyrant
Key-Value Cache	Memcached, Apache Ignite, eXtreme Scale, Hazelcast, Infinispan, JBoss Cache, Coherence, Repcached, Velocity
Tuple Store	Coord, Apache River, GigaSpaces
Object Database	Perst, Objectivity/DB, DB4O, Shoal, ZopeDB
Data-Structures	Redis
Wide Column Store	Bigtable, KAI, Amazon DynamoDB, HBase, Hypertable, Cassandra, Druid, OpenNeptune, KDI, Qbase
Key-Value Store	Oracle NoSQL Database, DovetailDB, Riak, Dynomite, Dynamo, Voldemort, SubRecord

RethinkDB as a Document Oriented NoSQL Database

URL : https://www.rethinkdb.com

RethinkDB is a high performance NoSQL database for real-time applications with effective web based administrative panel and user interface [3]. The database system provides excellent features for document storage based applications termed as Document-Oriented Database. RethinkDB provides the features for storage, retrieval and management of semi-structured data to achieve the minimum delay in the database operations. The key concept and paradigm behind the document-oriented database is the storage and handling of documents in unstructured or semi-structured formats. All document formats can be encapsulated with the encoding of data in document-oriented NoSQL databases. The encodings of data include YAML, BSON, XML, JSON alongwith the other binary forms including PDF, MS-Word, Spreadsheets and many others [4][5].

Real-Time Push Architecture of RethinkDB

The real-time web applications consume resources in parallel from different sources including database engine, live update scripts, bandwidth, concurrent connections, security modules and many others. To access and update the real-time data in web applications, there is need to integrate high performance technologies so that the live streaming data can be inserted or updated in minimum delay with minimum resource consumptions. For example, live polling or live chat based applications may be delayed because of enormous parallel users on the same channel and it can increase the waiting time. Using RethinkDB, this delay can be decreased as RethinkDB is developed specifically for the real-time web applications with the scalability and ease of use [6].

The real-time push architecture of RethinkDB is highly advantageous in the following types of applications and cases of implementations

- Real-time Web Applications
- High Traffic E-Commerce Applications
- Real-time Mobile Apps
- Connected Devices and Sharing Information
- Internet of Things (IoT)
- Cloud of Things (CoT)
- Multiplayer Real-time Games
- Satellite Channels
- Wireless Signals Analytics

RethinkDB is having a flexible and high performance query language ReQL to provide the real-time push based updates to multiple users in parallel. In addition, it provides the intuitive operations with the APIs for monitoring the real-time changes which are easy to setup and understand for the programmers. RethinkDB is in implementation by assorted corporate giants, Fortune 500 companies alongwith the startups to manage live applications.

Following are some of the key users and case of RethinkDB

- Jive Software
- Narrative Clip
- Mediafly
- Pristine.io
- CMUNE
- NodeCraft
- Platzi
- Workshape.io

The official packages for Ubuntu Linux, OS X, CentOS, Debian and Windows are available. In addition, the community supported packages are available on rethinkdb.com for Arch Linux, OpenSUSE, Fedora, Linux Mint, Raspbian, Gentoo.

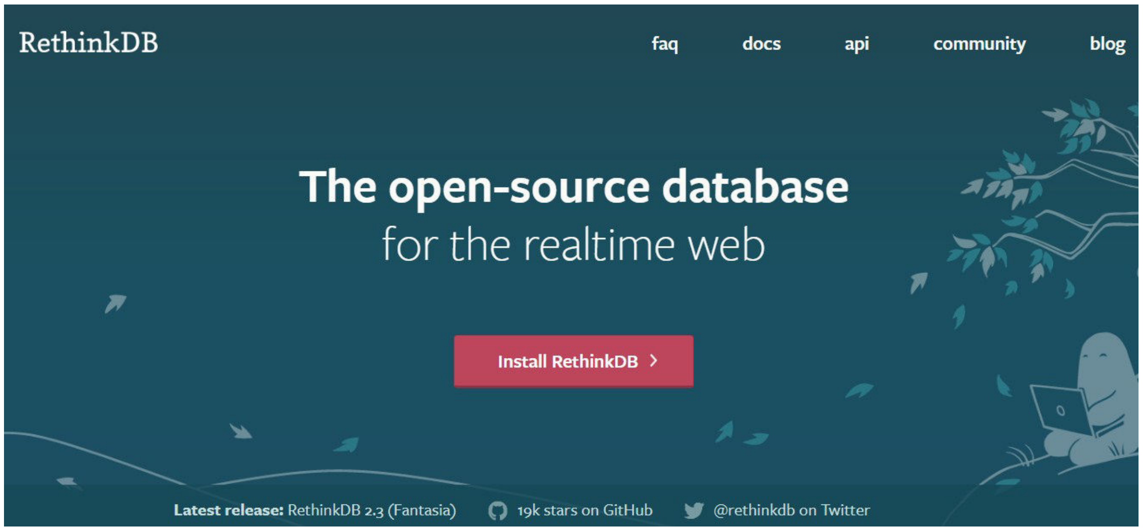


Figure 1 : Official Portal of RethinkDB

Official RethinkDB Client Drivers available for the following

- JavaScript
- Ruby
- Python
- Java

Community-Support Drivers

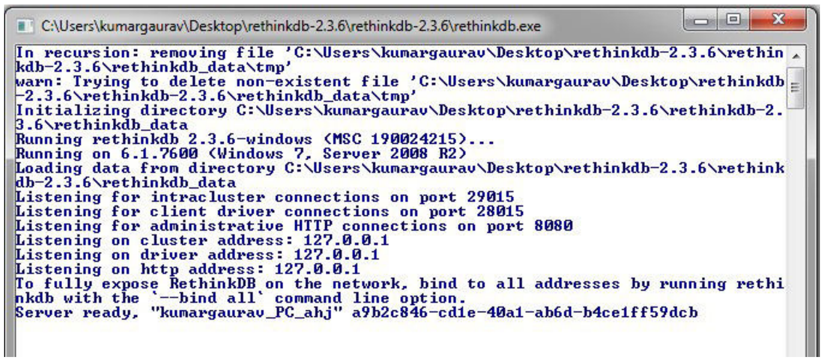
- C#
- Clojure
- Delphi
- Go
- Lua
- PHP
- Swift
- R
- Common Lisp
- Elixir
- Haskell
- Nim
- R
- C++
- Dart
- Erlang
- JS Neumino
- Perl
- Rust

The Limited Features based Drivers of RethinkDB are also available for Objective-C and Scala.

The 64-Bit Binaries for Windows O.S. and above are available on the official URL of RethinkDB *https://www.rethinkdb.com/docs/install/windows/.* To install RethinkDB on Windows, 64-Bit flavor is required.

A ZIP Archive is downloaded and unpacked in any directory. In the directory of RethinkDB, there is a file *rethinkdb.exe*. On double click of *rethinkdb.exe*, the server gets initiated and then it is available to further database operations.

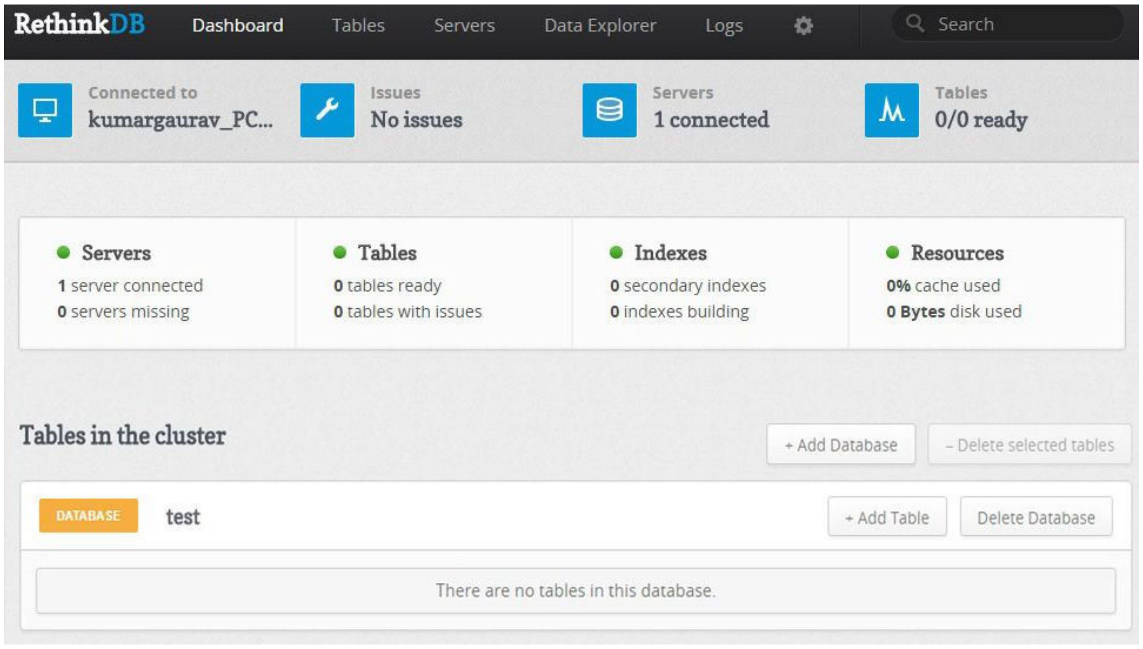
After starting RethinkDB server, the Administration Console is required to be opened on Web Browser with URL *http:// 127.0.0.1:8080* so that different options can be viewed for database administration.


C:\Users\kumargaurav\Desktop\rethinkdb-2.3.6\rethinkdb-2.3.6\rethinkdb.exe

```
In recursion: removing file 'C:\Users\kumargaurav\Desktop\rethinkdb-2.3.6\rethin
kdb-2.3.6\rethinkdb_data\tmp'
warn: Trying to delete non-existent file 'C:\Users\kumargaurav\Desktop\rethinkdb
-2.3.6\rethinkdb-2.3.6\rethinkdb_data\tmp'
Initializing directory C:\Users\kumargaurav\Desktop\rethinkdb-2.3.6\rethinkdb-2.
3.6\rethinkdb_data
Running rethinkdb 2.3.6-windows (MSC 190024215)...
Running on 6.1.7600 (Windows 7, Server 2008 R2)
Loading data from directory C:\Users\kumargaurav\Desktop\rethinkdb-2.3.6\rethink
db-2.3.6\rethinkdb_data
Listening for intracluster connections on port 29015
Listening for client driver connections on port 28015
Listening for administrative HTTP connections on port 8080
Listening on cluster address: 127.0.0.1
Listening on driver address: 127.0.0.1
Listening on http address: 127.0.0.1
To fully expose RethinkDB on the network, bind to all addresses by running rethi
nkdb with the `--bind all` command line option.
Server ready, "kumargaurav_PC_ahj" a9b2c846-cd1e-40a1-ab6d-b4ce1ff59dcb
```

Figure 2 : Starting the RethinkDB Server

In the administrative panel of RethinkDB Server, the options to create database, tables with different privileges are available. The connected servers, tables, indexes and resources can be viewed in this panel.



Connected to kumargaurav_PC...
Issues No issues
Servers 1 connected
Tables 0/0 ready

Servers
1 server connected
0 servers missing

Tables
0 tables ready
0 tables with issues

Indexes
0 secondary indexes
0 indexes building

Resources
0% cache used
0 Bytes disk used

Tables in the cluster

+ Add Database – Delete selected tables

DATABASE test + Add Table Delete Database

There are no tables in this database.

Figure 3 : Dashboard and Control Panel of RethinkDB

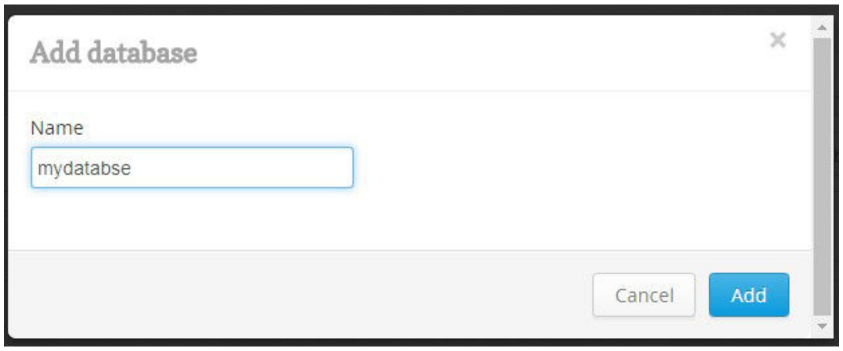
To use a specific directory for the storage and logging of data, following instruction is used

WindowsDirectory:\>rethinkdb.exe -d c:\RethinkDB\data\

To specify a particular server name and cluster

WindowsDirectory:\>rethinkdb.exe -n MyDomain -j mycluster.server.com

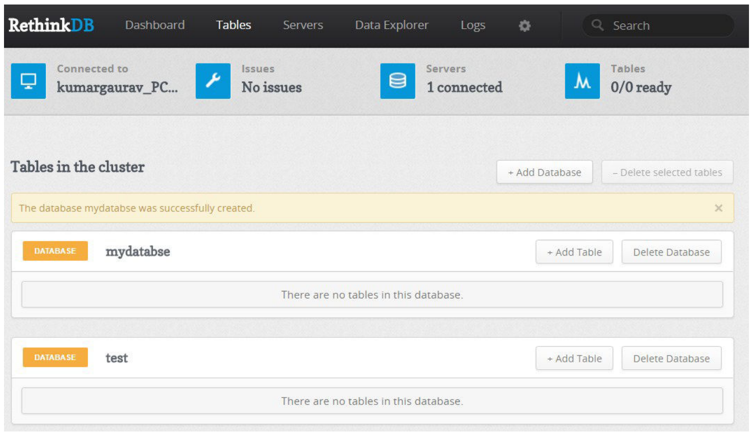
In the administrative console, there is option to create new database with any name. This database name can be called in the front-end applications for real-time applications.


Add database

Name: mydatabse

Figure 4 : Creating New Database

Once the database is created, it is visible in the administration console. In similar way, the table in a particular database can be created with the specification of primary key alongwith the acknowledgement.



Tables in the cluster

The database mydatabse was successfully created.

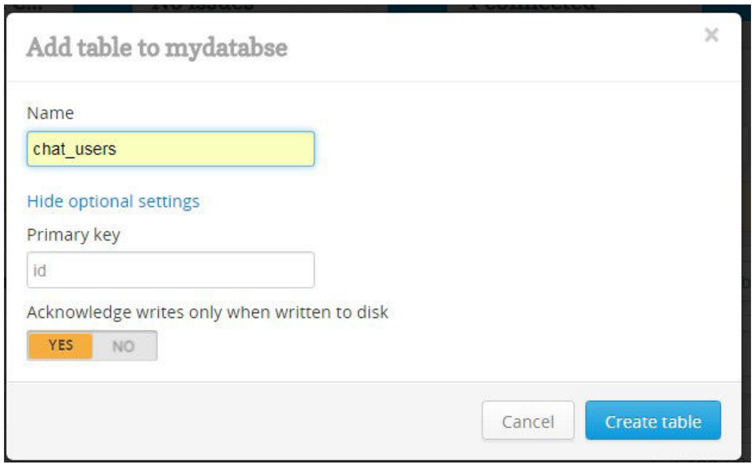
mydatabse

There are no tables in this database.

test

There are no tables in this database.

Figure 5 : View Database after creation



Add table to mydatabse

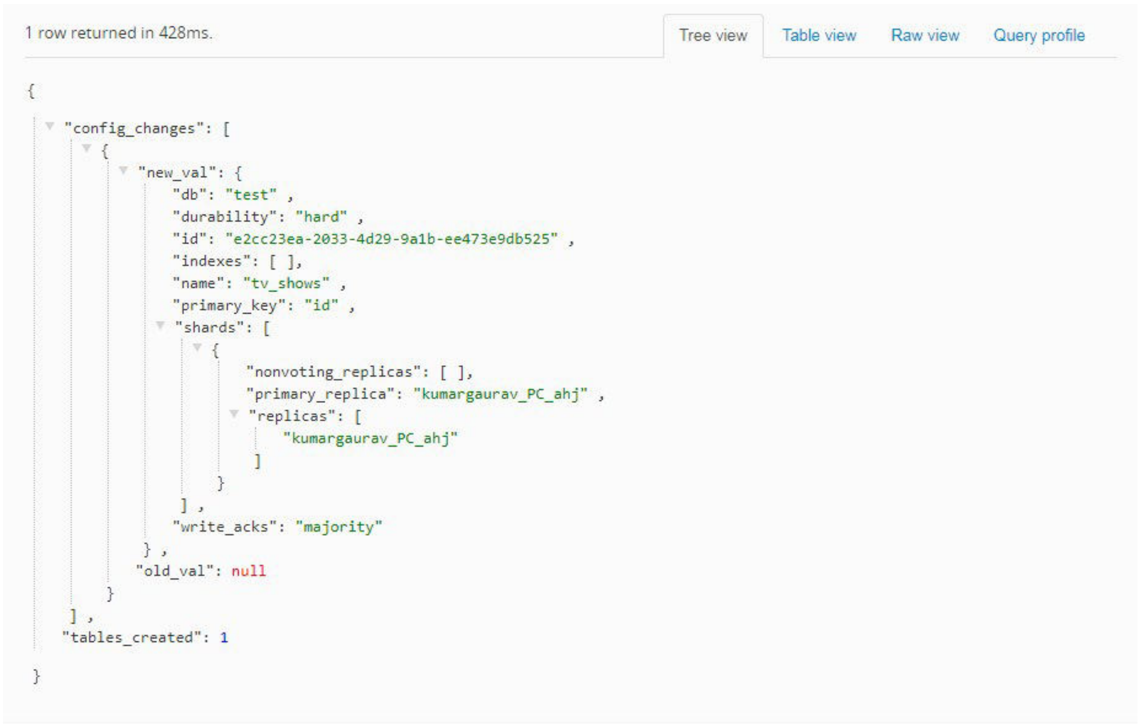
Name: chat_users

Primary key: id

Acknowledge writes only when written to disk: [✓ YES] NO

Figure 6 : Adding Table in specific database

After creation of table in specific database, the RethinkDB engine returns the message in JSON format so that the confirmation of write operation can be displayed having different values inserted in the database table.



```
{
  "config_changes": [
    {
      "new_val": {
        "db": "test" ,
        "durability": "hard" ,
        "id": "e2cc23ea-2033-4d29-9a1b-ee473e9db525" ,
        "indexes": [ ],
        "name": "tv_shows" ,
        "primary_key": "id" ,
        "shards": [
          {
            "nonvoting_replicas": [ ],
            "primary_replica": "kumargaurav_PC_ahj" ,
            "replicas": [
              "kumargaurav_PC_ahj"
            ]
          }
        ] ,
        "write_acks": "majority"
      } ,
      "old_val": null
    }
  ] ,
  "tables_created": 1
}
```

Figure 7 : Returned message after creating table on dashboard by RethinkDB

Installation of RethinkDB on Ubuntu

For installation of RethinkDB on Ubuntu Linux, both 32-Bit and 64-Bit flavors are available.

```
$ source /etc/lsb-release && echo "deb
http://download.rethinkdb.com/apt
$DISTRIB_CODENAME main" | sudo tee
/etc/apt/sources.list.d/rethinkdb.list
$ wget -qO-
https://download.rethinkdb.com/apt/pubkey.gpg |
sudo apt-key add -
$ sudo apt-get update
$ sudo apt-get install rethinkdb
```

Compilation and Installation from source

```
$ sudo apt-get install build-essential protobuf-
compiler python \
          libprotobuf-dev libcurl4-openssl-dev \
          libboost-all-dev libncurses5-dev \
          libjemalloc-dev wget m4
$ wget
https://download.rethinkdb.com/dist/rethinkdb-
version.tgz
$ tar xf rethinkdb- version.tgz
$ cd rethinkdb- version
$./configure --allow-fetch
$ make
$ sudo make install
```

To start the server on Ubuntu Linux, the following instruction is executed from terminal window

```
$ rethinkdb
```

In RethinkDB Administrative Console, there is a *Data Explorer* Tab which provides the panel to execute the commands to work on the databases. As in the following example, a new table titled *chat_users* is created in the database titled *mydatabase*.

r.db('mydatabase').tableCreate('chat_users ')

On click of Run button at the bottom-right of administration console, the query is executed and operation is implemented. In addition, the key combination *Shift+Enter* can be used to run the query entered.

Inserting Records in form of JSON in the table *chat_users*

r.table('chat_users).insert([{ name: 'Username-1', age: 17 },
{ name: 'Username-2', age: 35 }])

Counting the Number of Records in Table

r.table('chat_users).count()

Fetching Records on Specific Conditions

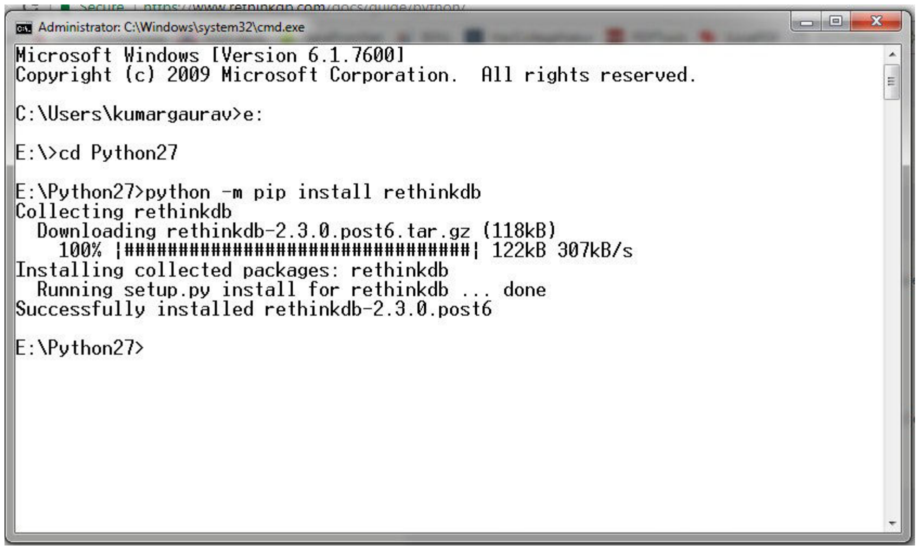
To display the records from table having age more than 20

r.table('chat_users).filter(r.row('age).gt(20))

Interfacing RethinkDB and Python

Python is one of the prominent free and open source programming languages that is having interfaces for almost all NoSQL databases. Python is widely used for cloud applications and assorted real-time high performance web applications. In python, RethinkDB can be easily installed and mapped using pip with the following instruction

WindowsDirectory:\> python -m pip install rethinkdb



```
Microsoft Windows [Version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\kumargaurav>e:

E:\>cd Python27

E:\Python27>python -m pip install rethinkdb
Collecting rethinkdb
  Downloading rethinkdb-2.3.0.post6.tar.gz (118kB)
    100% |################################| 122kB 307kB/s
Installing collected packages: rethinkdb
  Running setup.py install for rethinkdb ... done
Successfully installed rethinkdb-2.3.0.post6

E:\Python27>
```

Figure 8 : Installation of RethinkDB with Python

The package installer of Python automatically fetches the libraries of RethinkDB and map with the existing installation of Python.

After mapping of RethinkDB with Python, the IDLE Shell can use the methods and APIs of RethinkDB.

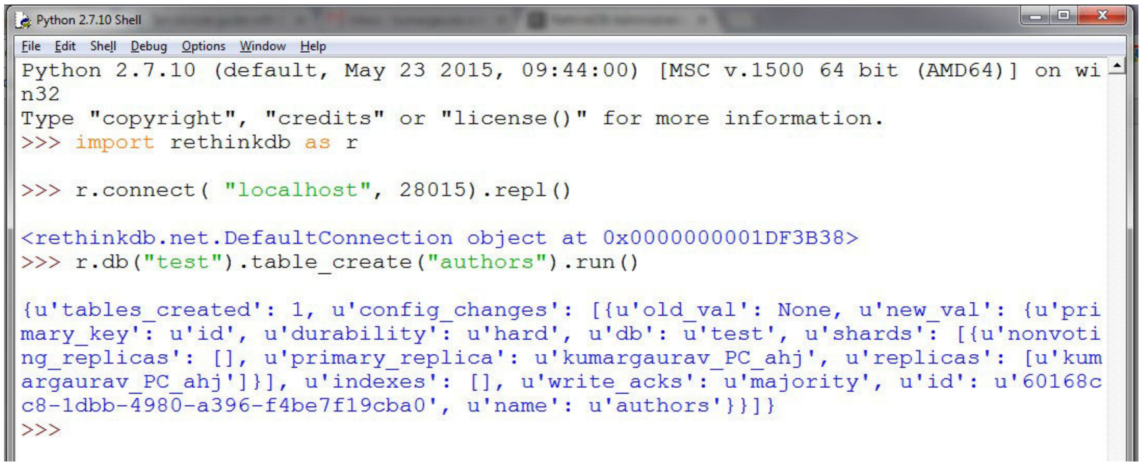
>> import rethinkdb as <Handle>

Using Python IDLE Shell or within the script, the connection can be opened with the following instruction having client driver with port number of 28015

>> r.connect("localhost", 28015).repl()

The repl command set and prepare the default connection with Python Shell

The function table_create() is used to add a new table in the particular database as follows



```
Python 2.7.10 (default, May 23 2015, 09:44:00) [MSC v.1500 64 bit (AMD64)] on wi
n32
Type "copyright", "credits" or "license()" for more information.
>>> import rethinkdb as r

>>> r.connect( "localhost", 28015).repl()

<rethinkdb.net.DefaultConnection object at 0x0000000001DF3B38>
>>> r.db("test").table_create("authors").run()

{u'tables_created': 1, u'config_changes': [{u'old_val': None, u'new_val': {u'pri
mary_key': u'id', u'durability': u'hard', u'db': u'test', u'shards': [{u'nonvoti
ng_replicas': [], u'primary_replica': u'kumargaurav_PC_ahj', u'replicas': [u'kum
argaurav_PC_ahj']}], u'indexes': [], u'write_acks': u'majority', u'id': u'60168c
c8-1dbb-4980-a396-f4be7f19cba0', u'name': u'authors'}}]}
>>>
```

Figure 9 : Execution of RethinkDB instructions in Python IDLE Shell

Conclusion

RethinkDB is the first database that is scalable as well as open source build with the base of JSON for real-time web applications. Using RethinkDB, the developer can direct the database engine to push the updated query results in real-time to the running application so that maximum updates can be done without delay. The real-time push architecture of RethinkDB drastically reduces the efforts, time and resources for the real-time applications. RethinkDB is having a vibrant as well as titanic community of more than 1,00,000 software developers and troubleshooting experts throughout the world. These developers and contributors keep on customizing the code and sharing on the online portals to enrich the overall community of RethinkDB programmers.

References

[1] Han J, Haihong E, Le G, Du J. Survey on NoSQL database. InPervasive computing and applications (ICPCA), 2011 6th international conference on 2011 Oct 26 (pp. 363-366). IEEE.

[2] Li Y, Manoharan S. A performance comparison of SQL and NoSQL

databases. In Communications, computers and signal processing (PACRIM), 2013 IEEE pacific rim conference on 2013 Aug 27 (pp. 15-19). IEEE.

[3] Bekas E, Magoutis K. Cross-layer management of a containerized NoSQL data store. InIntegrated Network and Service Management (IM), 2017 IFIP/IEEE Symposium on 2017 May 8 (pp. 1213-1221). IEEE.

[4] George S. NoSQL–NOT ONLY SQL. International Journal of Enterprise Computing and Business Systems. 2013;2(2).

[5] Vaish G. Getting started with NoSQL. Packt Publishing Ltd; 2013 Mar 26.

[6] Tiepolo G. Getting started with rethinkdb. Packt Publishing Ltd; 2016 Mar 17.