FLIGHT DATA ANALYSIS USING COUCHDB

BDA MINI PROJECT REPORT SUBMITTED BY

Xylene Vinitha Dsouza
4NM17CS211
VI Semester, D Section

Shreema Naik
4NM17CS174
VI Semester, D section

UNDER THE GUIDANCE OF

Mrs. Divya Jennifer D'souza
Assistant professor GD II
Department of Computer Science and Engineering

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE OF

Bachelor of Engineering in Computer Science & Engineering

from

Visvesvaraya Technological University, Belagavi

N.M.A.M. INSTITUTE OF TECHNOLOGY
(An Autonomous Institution under VTU, Belgaum)
AICTE approved, (ISO 9001:2015 Certified), Accredited with 'A' Grade by NAAC
NITTE -574 110, Udupi District, KARNATAKA.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

B.E. CSE Program Accredited by NBA, New Delhi from 1-7-2018 to 30-6-2021

May 2020

NITTE EDUCATION TRUST | N.M.A.M. INSTITUTE OF TECHNOLOGY
(An Autonomous Institution affiliated to Visvesvaraya Technological University, Belagavi)
Nitte – 574 110, Karnataka, India
(ISO 9001:2015 Certified), Accredited with 'A' Grade by NAAC
☎: 08258 - 281039 – 281263, Fax: 08258 – 281265

Department of Computer Science and Engineering

B.E. CSE Program Accredited by NBA, New Delhi from 1-7-2018 to 30-6-2021

CERTIFICATE

"Flight Data Analysis Using Couchdb" is a bonafide work carried out by Xylene Vinitha Dsouza (4NM17CS211) and Shreema Naik (4NM17CS174) in partial fulfilment of the requirements for the award of Bachelor of Engineering Degree in Computer Science and Engineering prescribed by Visvesvaraya Technological University, Belagavi during the year 2019-2020.

It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report. The Mini project report has been approved as it satisfies the academic requirements in respect of the project work prescribed for the Bachelor of Engineering Degree.

Signature of Guide

Signature of HOD

ACKNOWLEDGEMENT

We believe that our project will be complete only after we thank the people who have contributed to make this project successful.

First and foremost, our sincere thanks to our beloved principal, Dr. Niranjan N. Chiplunkar for giving us an opportunity to carry out our project work at our college and providing us with all the needed facilities.

We sincerely thank Dr. K.R. Udaya Kumar Reddy, Head of Department of Computer Science and Engineering, Nitte Mahalinga Adyantaya Memorial Institute of Technology, Nitte.

We express our deep sense of gratitude and indebtedness to our guide Mrs. Divya Jennifer D'souza, Assistant Professor GD II, Department of Computer Science and Engineering, for her inspiring guidance, constant encouragement, support and suggestions for improvement during the course of our project.

We thank all the teaching and non-teaching staff members of the Computer Science and Engineering Department and our parents and friends for their honest opinions and suggestions throughout the course of our project.

Finally, we thank all those who have supported us directly or indirectly throughout the project and making it a grand success.

Xylene Vinitha Dsouza
(4NM17CS211)

Shreema Naik
(4NM17CS174)

ABSTRACT

Flight delays are an important issue in the flight industry, because it will lead to financial crisis in the business. This project identifies the factors influence the occurrence of flight delays. Research survey indicates that every year about 20% of flights are delayed or cancelled. It costs in very big way for both travellers and airlines.

The project is to analyze flight data by gathering data from official web portal. The data that's maintained in web portal is big in size and it is increasing every day. So obviously big data analytics are the best way to analyze the data and extract the useful knowledge from the data set. CouchDB and MapReduce function is used here in this project as a big data concept.

Table of Contents

INTRODUCTION

Flight delay is one of the most common but an unpleasant experience that people dread to have. Every year, a lot of flights get delayed which involves some cost both for the airline and the passenger in different ways. The passenger's time and money get affected and at the same time, the airline's reputation is at stake. Delay is treated as one of the most remembered performance indicators of the airline. There could be some reasons which are inevitable such as weather conditions, air trafficking or any unforeseen event; but there also could be some reasons which can be dealt with by improving the process. Therefore, statistics of the flight delays becomes crucial factor in understanding the flight's performance.

CouchDB stores data as "documents", as one or more field/value pairs expressed as JSON. Field values can be simple things like strings, numbers, or dates; but ordered lists and associative arrays can also be used. Every document in a CouchDB database has a unique id and there is no required document schema.

The stored data is structured using views. In CouchDB, each view is constructed by a JavaScript function that acts as the Map half of a map/reduce operation. The function takes a document and transforms it into a single value that it returns. CouchDB can index views and keep those indexes updated as documents are added, removed, or updated.

SOFTWARE REQUIREMENTS

- VMware Workstation
- Apache CouchDB

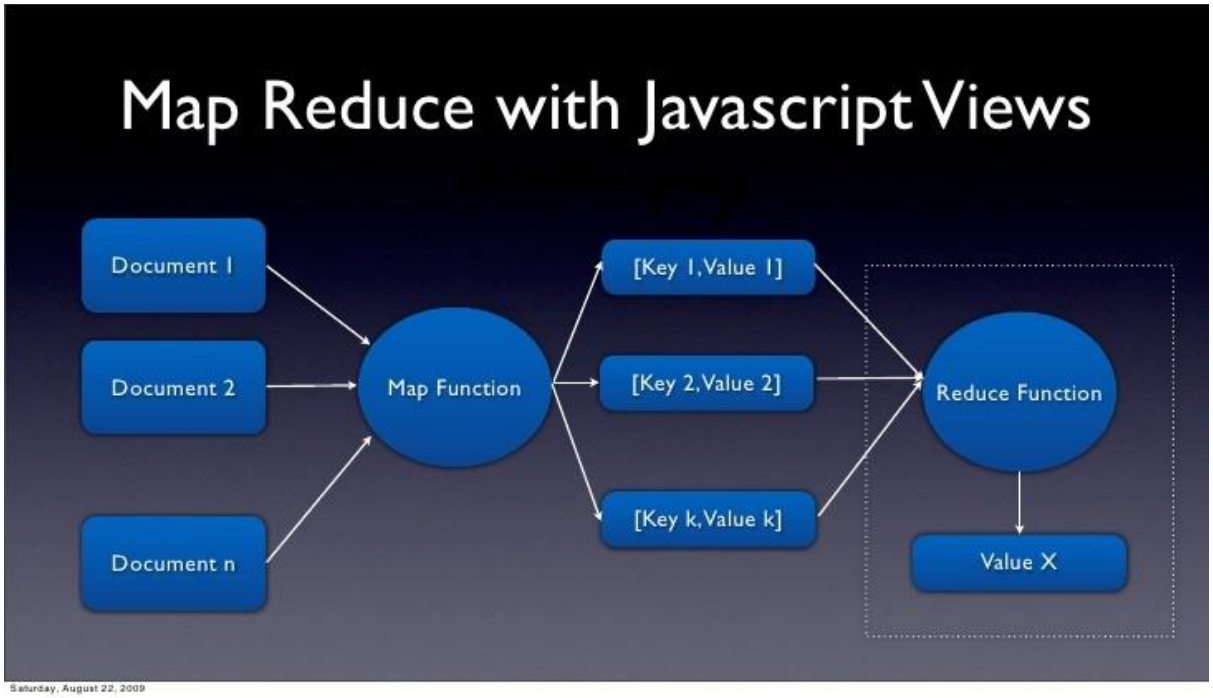
Apache CouchDB lets you access your data where you need it. The Couch Replication Protocol is implemented in a variety of projects and products that span every imaginable computing environment from globally distributed server-clusters, over mobile phones to web browsers. Store your data safely, on your own servers, or with any leading cloud provider. Your web- and native applications love CouchDB, because it speaks JSON natively and supports binary data for all your data storage needs. The Couch Replication Protocol lets your data flow seamlessly between server clusters to mobile phones and web browsers, enabling a compelling offline-first user-experience while maintaining high performance and strong reliability. CouchDB comes with a developer-friendly query language, and optionally MapReduce for simple, efficient, and comprehensive data retrieval.

HARDWARE REQUIREMENTS

- 4GB or 8GB RAM
- Intel i3 or above processor
- 2 GB or above storage
- Dual-core x86 CPU running at 2GHz.
- A block-based storage device (hard disk, SSD, EBS, iSCSI).Network file systems such as CIFS and NFS are not supported.

DESIGN AND ANALYSIS

Map and Reduce function



Introduction to views :

Views are the primary tool used for querying and reporting on CouchDB documents.

Views are useful for many purposes:

- Filtering the documents in your database to find those relevant to a particular process.
- Extracting data from your documents and presenting it in a specific order.
- Building efficient indexes to find documents by any value or structure that resides in them.
- Use these indexes to represent relationships among documents.

The documents are sorted by "_id", which is how they are stored in the database.

Let us consider the example for a view.

```
function (doc) {
   if (doc.date && doc.title) {
      emit (doc.date, doc.title);
   }
}
```

This is a map function, and it is written in JavaScript. It is a simple function definition.

You provide CouchDB with view functions as strings stored inside the views field of a design document. You don't run it yourself. Instead, when you query your view, CouchDB takes the source code and runs it for you on every document in the database your view was defined in. You query your view to retrieve the view result.

All map functions have a single parameter doc. This is a single document in the database. Our map function checks whether our document has a date and a title attribute — luckily, all of our documents have them — and then calls the built-in emit() function with these two attributes as arguments.

The emit() function always takes two arguments: the first is key, and the second is value. The emit(key, value) function creates an entry in our view result. The emit() function can be called multiple times in the map function to create multiple entries in the view results from a single document.

Facts about Map/Reduce:

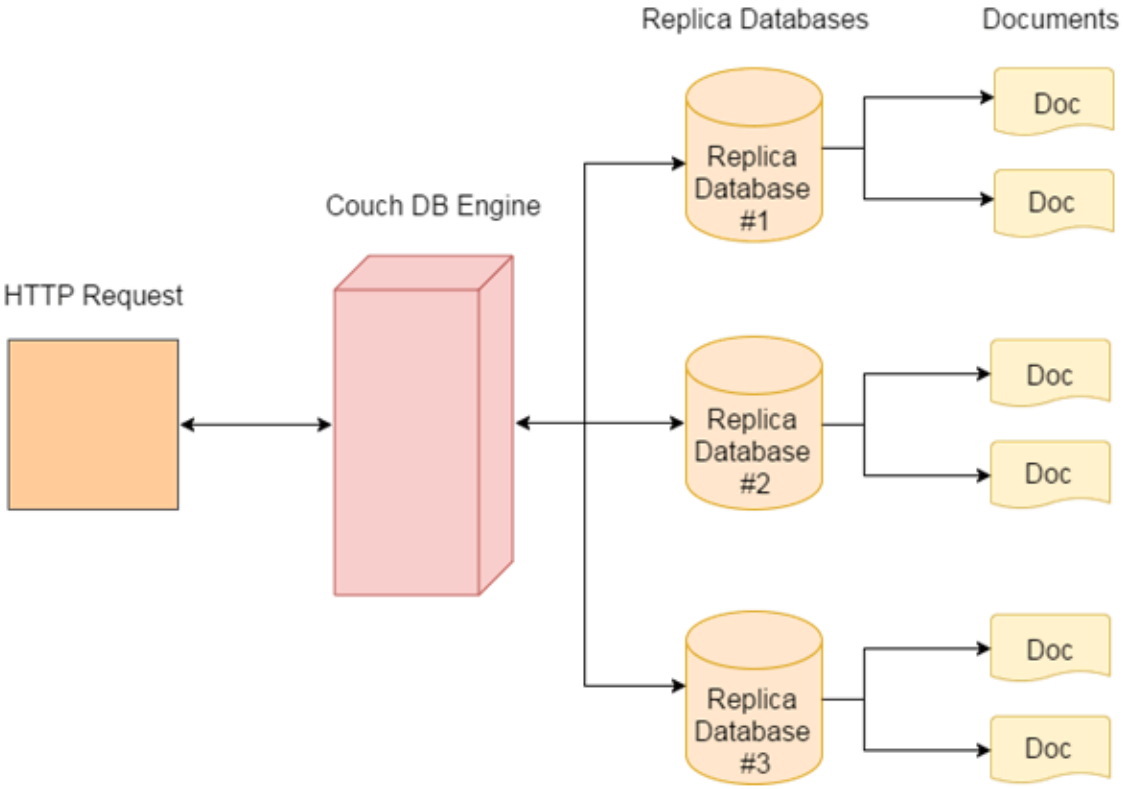
- Programming paradigm popularized and patented by Google.
- Great for parallel jobs.
- No joins between documents.
- Work flow->map builds a list of key/values pair and reduce function reduces the list to a single value.

Reduce Function :

- Has arrays of keys and values as input.
- Should reduce the result of a map to a single value.
- JavaScript is used.
- It is automatically called after the map function is finished.
- It can be ignored with reduce=false.
- It is needed for grouping.

Reduce / Rereduce : It is a rule to use reduce-functions. The input of a reduce function does not only accept the result of a map, but also the result of itself if the map is too large, then it will be split and each part runs through the reduce function ,finally all the results run through the same reduce function again.

IMPLEMENTATION



CouchDB uses multiple formats and protocols to store, transfer, and process its data, it uses JSON to store data, JavaScript as its query language using MapReduce, and HTTP for an API.

CouchDB was first released in 2005 and later became an Apache Software Foundation project in 2008.

Unlike a relational database, a CouchDB database does not store data and relationships in tables. Instead, each database is a collection of independent documents. Each document maintains its own data and self-contained schema. An application may access multiple databases, such as one stored on a user's mobile phone and another on a server. Document metadata contains revision information, making it possible to merge any differences that may have occurred while the databases were disconnected.

CouchDB implements a form of multiversion concurrency control (MVCC) so it does not lock the database file during writes. Conflicts are left to the application

to resolve. Resolving a conflict generally involves first merging data into one of the documents, then deleting the stale one.
Other features include document-level ACID semantics with eventual consistency, (incremental) MapReduce, and (incremental) replication. One of CouchDB's distinguishing features is multi-master replication, which allows it to scale across machines to build high-performance systems. A built-in Web application called Fauxton (formerly Futon) helps with administration.

Main features :

- ACID Semantics
  CouchDB provides ACID semantics. It does this by implementing a form of Multi-Version Concurrency Control, meaning that CouchDB can handle a high volume of concurrent readers and writers without conflict.
- Built for Offline
  CouchDB can replicate to devices (like smartphones) that can go offline and handle data sync for you when the device is back online.
- Distributed Architecture with Replication
  CouchDB was designed with bi-directional replication (or synchronization) and off-line operation in mind. That means multiple replicas can have their own copies of the same data, modify it, and then sync those changes at a later time.
- Document Storage
  CouchDB stores data as "documents", as one or more field/value pairs expressed as JSON. Field values can be simple things like strings, numbers, or dates; but ordered lists and associative arrays can also be used. Every document in a CouchDB database has a unique id and there is no required document schema.
- Eventual Consistency
  CouchDB guarantees eventual consistency to be able to provide both availability and partition tolerance.
- Map/Reduce Views and Indexes
  The stored data is structured using views. In CouchDB, each view is constructed by a JavaScript function that acts as the Map half of a map/reduce operation. The function takes a document and transforms it into a single value that it returns. CouchDB can index views and keep those indexes updated as documents are added, removed, or updated.

- HTTP API
  All items have a unique URI that gets exposed via HTTP. It uses the HTTP methods POST, GET, PUT and DELETE for the four basic CRUD (Create, Read, Update, Delete) operations on all resources

Data manipulation : Documents and Views
CouchDB manages a collection of JSON documents. The documents are organised via views. Views are defined with aggregate functions and filters are computed in parallel, much like MapReduce.
Views are generally stored in the database and their indexes updated continuously. CouchDB supports a view system using external socket servers and a JSON-based protocol. As a consequence, view servers have been developed in a variety of languages.

Steps for installing CouchDB :

- Update the package list:
```
sudo apt update
```
- Add the CouchDB PPA repository:
```
sudo add-apt-repository ppa:couchdb/stable
```
- Update the package list once more:
```
sudo apt update
```
- Then install CouchDB with the command:
```
sudo apt install couchdb
```
- Use curl to verify that CouchDB is installed and running:
```
curl localhost:5984
```
The server will respond with a welcome message:
```
{"couchdb":"Welcome","uuid":"266009d1980fbda93d96cd3bd
95c2e81","version":"1.6.1","vendor":{"version":"16.04"
,"name":"Ubuntu"}}
```

CouchDB uses a basic HTTP interface, and returns JSON objects. You can use the curl utility from the command line to communicate with CouchDB.

- The command to view a list of databases is:

```
curl localhost:5984/_all_dbs
```

The server will return a JSON reply to the command line:

```
student@ubuntu:~$ curl localhost:5984/_all_dbs

["_replicator","_users"]
```

- The curl command uses the GET method by default. To send CouchDB a request with a different method, use the -X flag to override the default method, and specify the method you wish to use instead:

```
curl -X PUT localhost:5984/[etc.]
```

- We must use the PUT method to create a database. The command to create a database named flights is:

```
curl -X PUT http://127.0.0.1:5984/flights
```

The server will confirm the creation:

```
{"ok":true}
```

- We must use the GET method to get the information of the database. The command to create a database named flights is:

```
curl -X GET http://127.0.0.1:5984/flights
```

- The server will respond with the database information.

{"db_name":"flights","doc_count":50,"doc_del_count":2,"update_seq":69,
"purge_seq":0,"compact_running":false,"disk_size":286824,"data_size":31
407,"instance_start_time":"1589356156304603","disk_format_version":6,
"committed_update_seq":69}

- The PUT request is used to create new objects, databases, documents, views and design documents.

```
student@ubuntu:~$ curl -X PUT http://127.0.0.1:5984/flight/"001" -d
'{"YEAR":2015,"MONTH":1,"DAY":1,"DAY_OF_WEEK":4,"AIRLINE
":"AS","FLIGHT_NUMBER":98,"TAIL_NUMBER":"N407AS",
"ORIGIN_AIRPORT":"ANC","DESTINATION_AIRPORT":"SEA",
"SCHEDULED_DEPARTURE":5,"DEPARTURE_TIME":2354,
"DEPARTURE_DELAY":11,"TAXI_OUT":21,"WHEELS_OFF":15,
"SCHEDULED_TIME":205,"ELAPSED_TIME":194,"AIR_TIME":169,
"DISTANCE":1448,"WHEELS_ON":404,"TAXI_IN":4,"SCHEDULED_
ARRIVAL":430,"ARRIVAL_TIME":408,"ARRIVAL_DELAY":22,
"DIVERTED":0,"CANCELLED":0,"CANCELLATION_REASON":"","
AIR_SYSTEM_DELAY":"","SECURITY_DELAY":"","AIRLINE_DEL
AY":"","LATE_AIRCRAFT_DELAY":"","WEATHER_DELAY":""}'
```

- The server will return a JSON document with content “ok”-true indicating the operation was successful along with “id” which stores the id of the document and “rev” indicates revision id.

"ok":true,"id":"001","rev":"1-7b0679872f892890f77bdbba9a41aadc"}

- The GET request is used to get the information of the document from the specified database.

```
curl -X GET http://127.0.0.1:5984/flight/001
```

- The server will return the document from specified database and specified id.

{"_id":"001","_rev":"17b0679872f892890f77bdbba9a41aadc","YEAR":2015,"MONTH":1,"DAY":1,"DAY_OF_WEEK":4,"AIRLINE":"AS","FLIGHT_NUMBER":98,"TAIL_NUMBER":"N407AS","ORIGIN_AIRPORT":"ANC","DESTINATION_AIRPORT":"SEA","SCHEDULED_DEPARTURE":5,"DEPARTURE_TIME":2354,"DEPARTURE_DELAY":-11,"TAXI_OUT":21,"WHEELS_OFF":15,"SCHEDULED_TIME":205,"ELAPSED_TIME":194,"AIR_TIME":169,"DISTANCE":1448,"WHEELS_ON":404,"TAXI_IN":4,"SCHEDULED_ARRIVAL":430,"ARRIVAL_TIME":408,"ARRIVAL_DELAY":22,"DIVERTED":0,"CANCELLED":0,"CANCELLATION_REASON":"","AIR_SYSTEM_DELAY":"","SECURITY_DELAY":"","AIRLINE_DELAY":"","LATE_AIRCRAFT_DELAY":"","WEATHER_DELAY":""}

We can easily use and manage CouchDB from the command line using the Curl utility. However, CouchDB also comes with a built-in web-based administration interface named Futon.

FLIGHT DATASET DESCRIPTION

1: YEAR: Year of the Flight Trip

2: MONTH: Month of the Flight Trip

3: DAY: Day of the Flight Trip

4: DAY_OF_WEEK: Day of week of the Flight Trip

5: AIRLINE: Airline Identifier

6: FLIGHT_NUMBER: Flight Identifier

7: TAIL_NUMBER: Aircraft Identifier

8: ORIGIN_AIRPORT: Starting Airport

9: DESTINATION_AIRPORT: Destination Airport

10: SCHEDULED_DEPARTURE: Planned Departure Time

11: DEPARTURE_TIME: WHEEL_OFF - TAXI_OUT

12: DEPARTURE_DELAY: Total Delay on Departure

13: TAXI_OUT: The time duration elapsed between departure from the origin airport gate and wheels off

14: WHEELS_OFF: The time point that the aircraft's wheels leave the ground

15: SCHEDULED_TIME: Planned time amount needed for the flight trip

16: ELAPSED_TIME: AIR_TIME + TAXI_IN + TAXI_OUT

17: AIR_TIME: The time duration between wheels_off and wheels_on time

18: DISTANCE: Distance between two airports

19: WHEELS_ON: The time point that the aircraft's wheels touch on the ground

20: TAXI_IN: The time duration elapsed between wheels-on and gate arrival at the destination airport

21: SCHEDULED_ARRIVAL: Planned arrival time

22: ARRIVAL_TIME: WHEELS_ON + TAXI_IN

23: ARRIVAL_DELAY: ARRIVAL_TIME - SCHEDULED_ARRIVAL

24: DIVERTED: Aircraft landed on airport that out of schedule

25: CANCELLED: Flight Cancelled (1 = cancelled,0=Not Cancelled)

26: CANCELLATION_REASON: Reason for Cancellation of flight:

27: AIR_SYSTEM_DELAY: Delay caused by air system

28: SECURITY_DELAY: Delay caused by security

29: AIRLINE_DELAY: Delay caused by the airline

30: LATE_AIRCRAFT_DELAY: Delay caused by aircraft

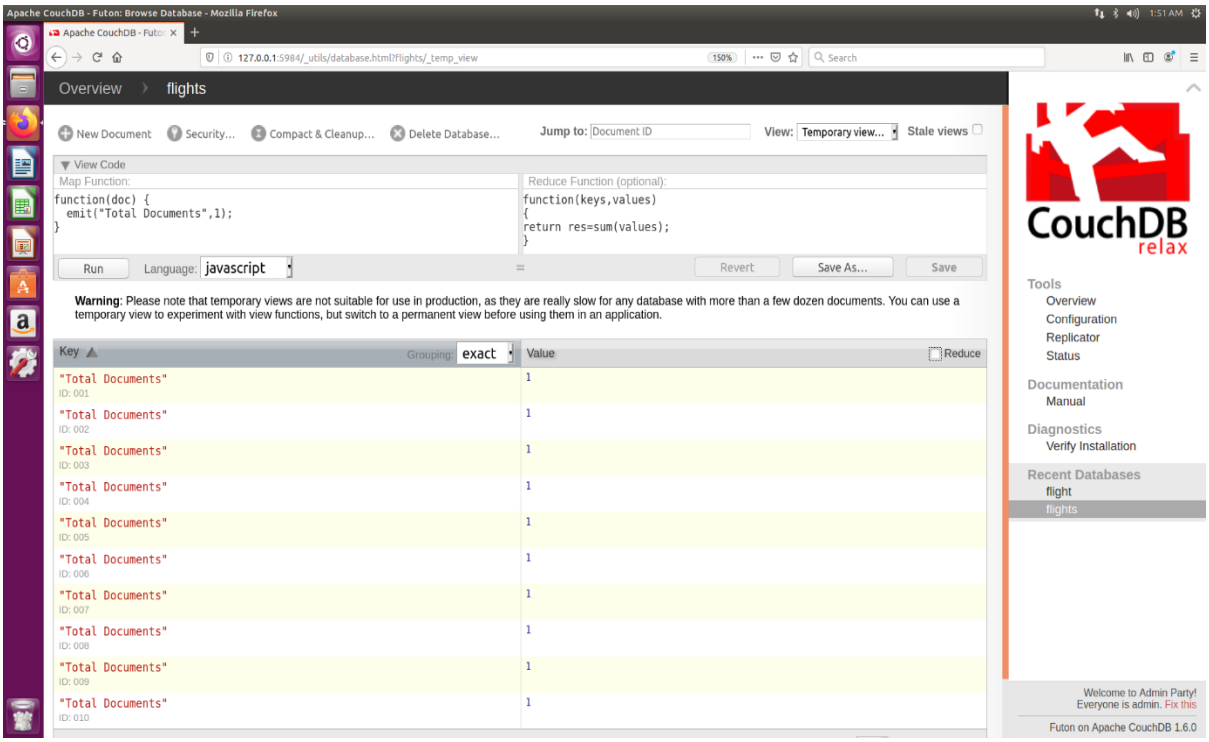
31: WEATHER_DELAY: Delay caused by weather

RESULT

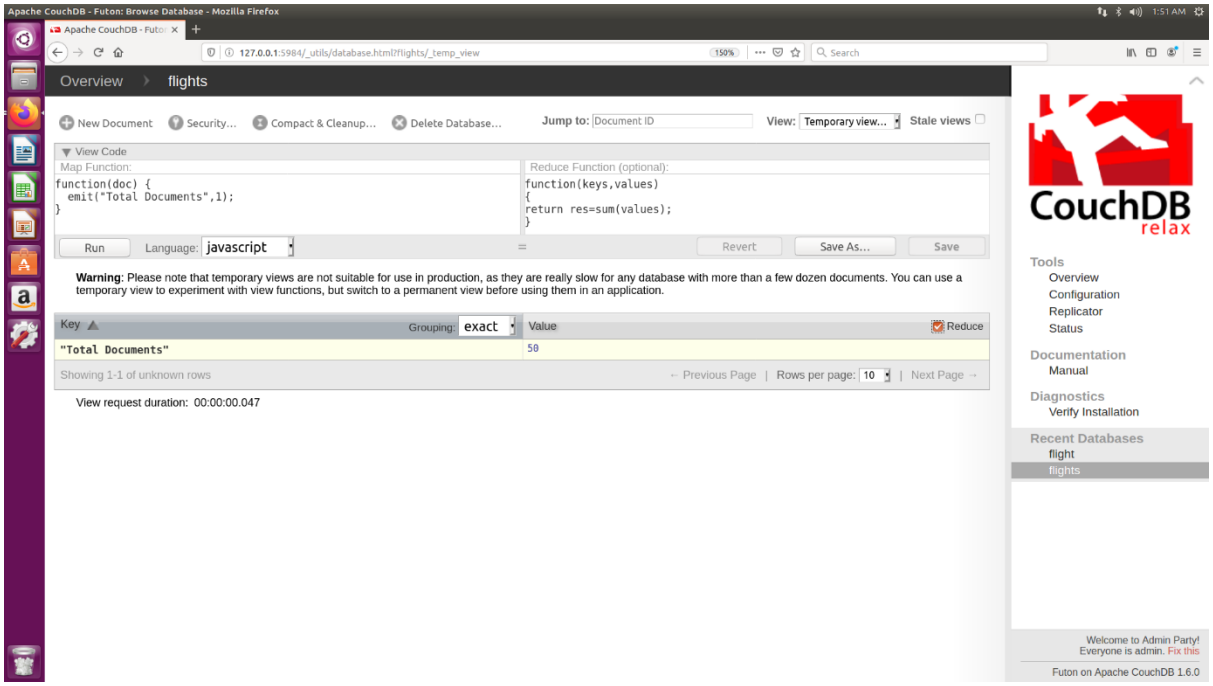
This section contains screenshots of our project queries.

1. Find the total number of documents present in the database.

Map :

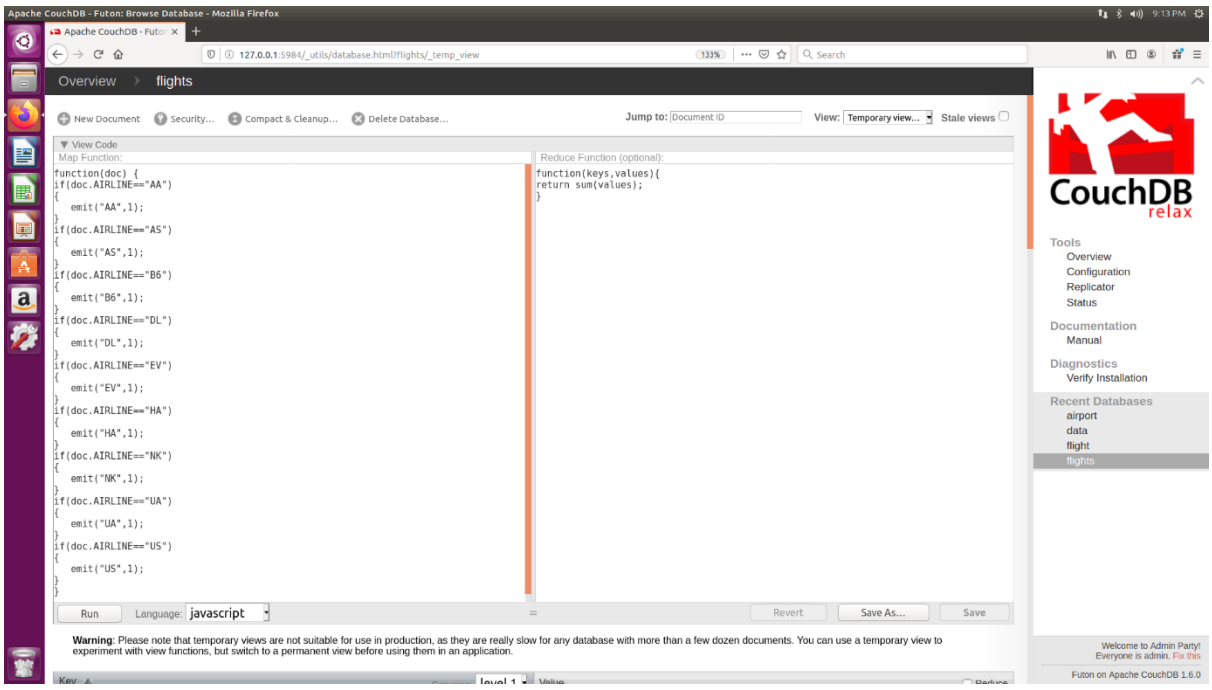


Reduce :

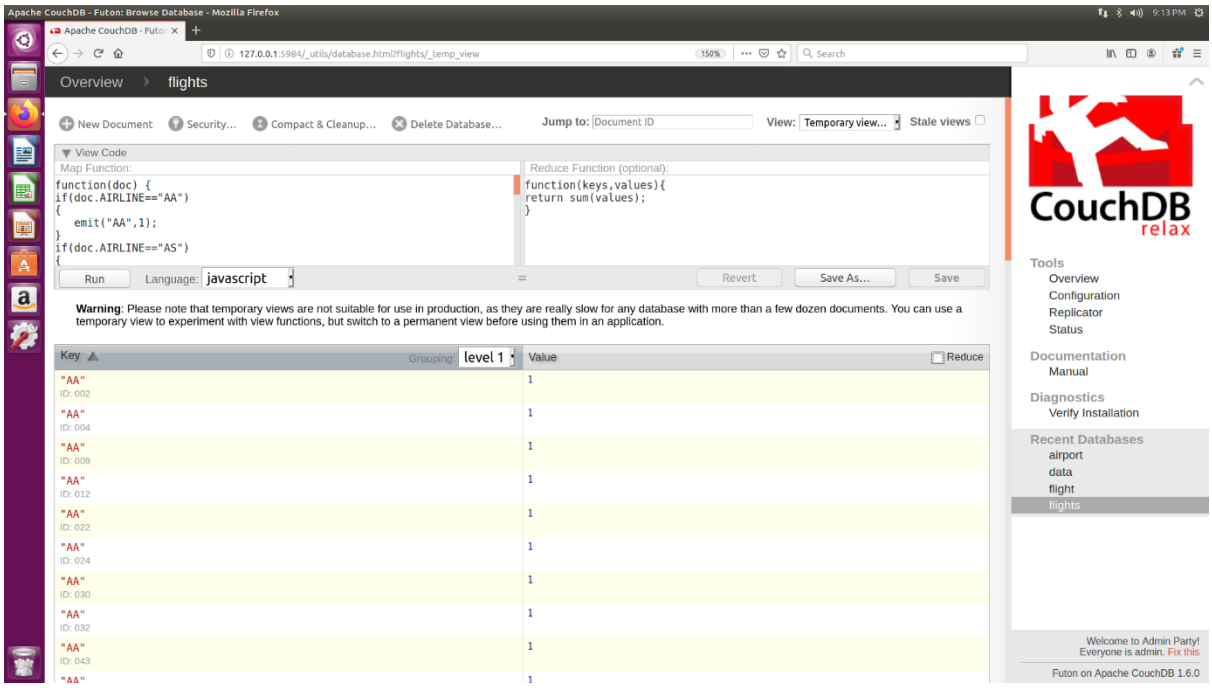


2. List all the airline names present in the database and get the total count of each airline.

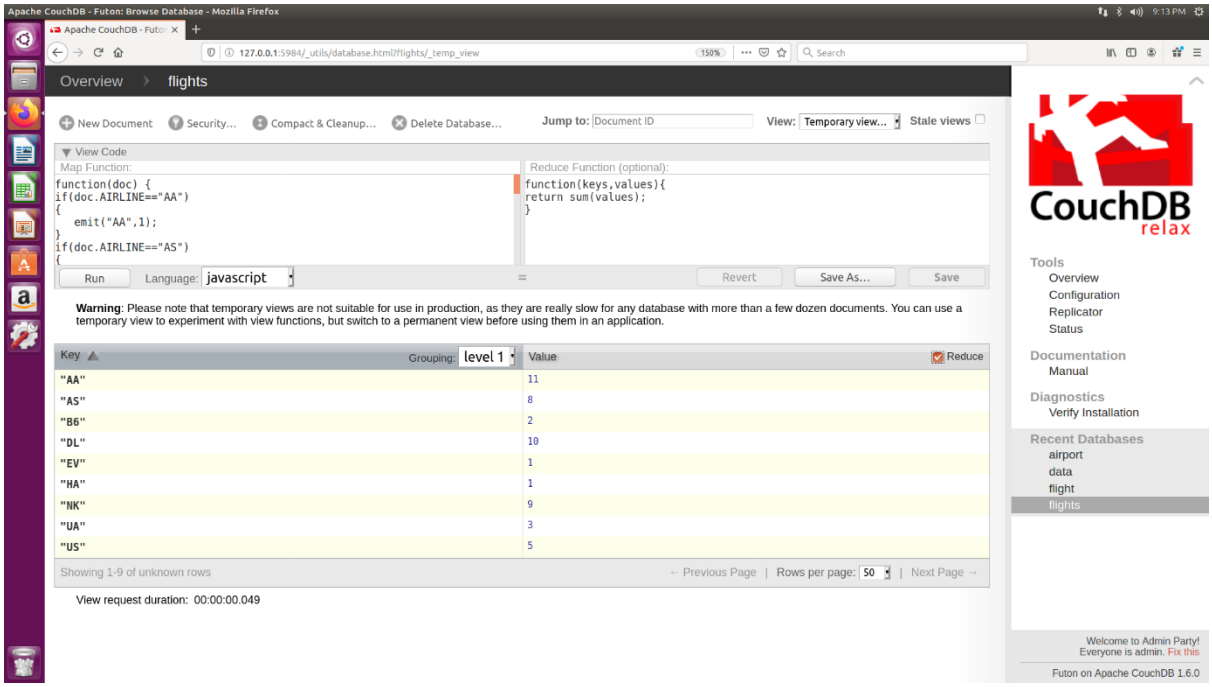
QUERY :



Map :

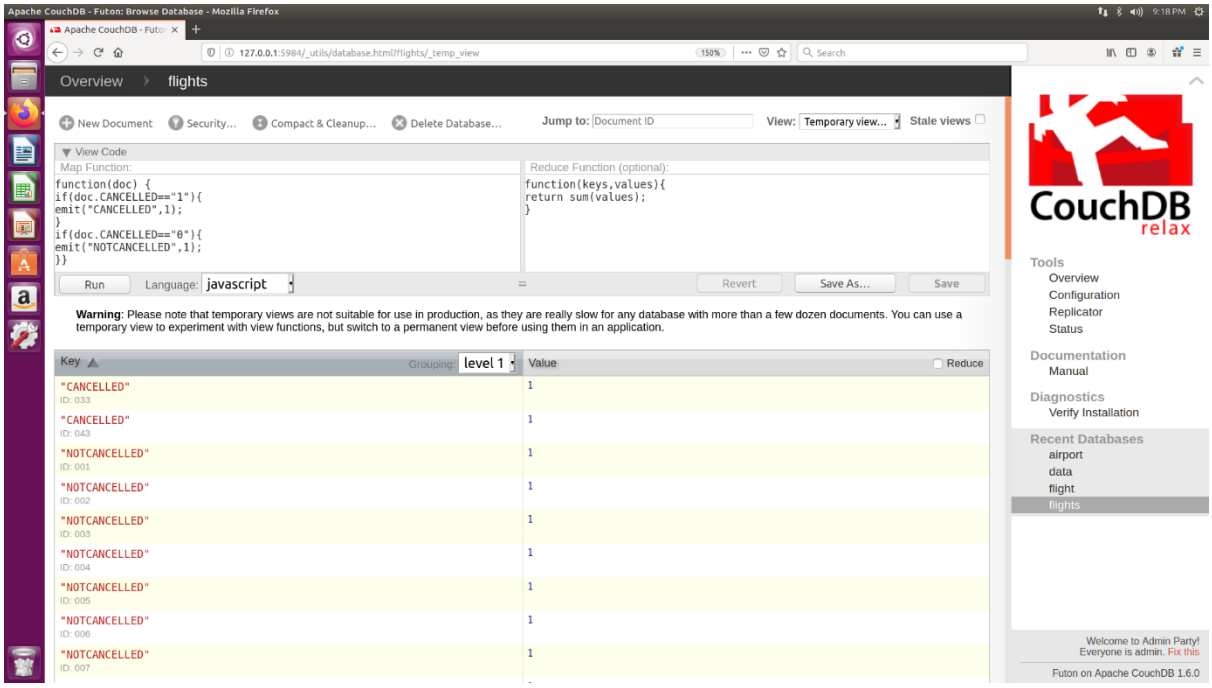

Apache CouchDB - Futon: Browse Database - Mozilla Firefox
Overview
flights
New Document
Security...
Compact & Cleanup...
Delete Database...
Jump to: Document ID
View: Temporary view...
Stale views
View Code
Map Function:
function(doc) {
if(doc.AIRLINE=="AA")
{
emit("AA",1);
}
if(doc.AIRLINE=="AS")
{
Reduce Function (optional):
function(keys,values){
return sum(values);
}
Run
Language: javascript
Revert
Save As...
Save
Warning: Please note that temporary views are not suitable for use in production, as they are really slow for any database with more than a few dozen documents. You can use a temporary view to experiment with view functions, but switch to a permanent view before using them in an application.
Key
Grouping: level 1
Value
Reduce
"AA" ID: 002 1
"AA" ID: 004 1
"AA" ID: 009 1
"AA" ID: 012 1
"AA" ID: 022 1
"AA" ID: 024 1
"AA" ID: 030 1
"AA" ID: 032 1
"AA" ID: 043 1
CouchDB relax
Tools
Overview
Configuration
Replicator
Status
Documentation
Manual
Diagnostics
Verify Installation
Recent Databases
airport
data
flight
flights
Welcome to Admin Party!
Everyone is admin. Fix this
Futon on Apache CouchDB 1.6.0

Reduce :

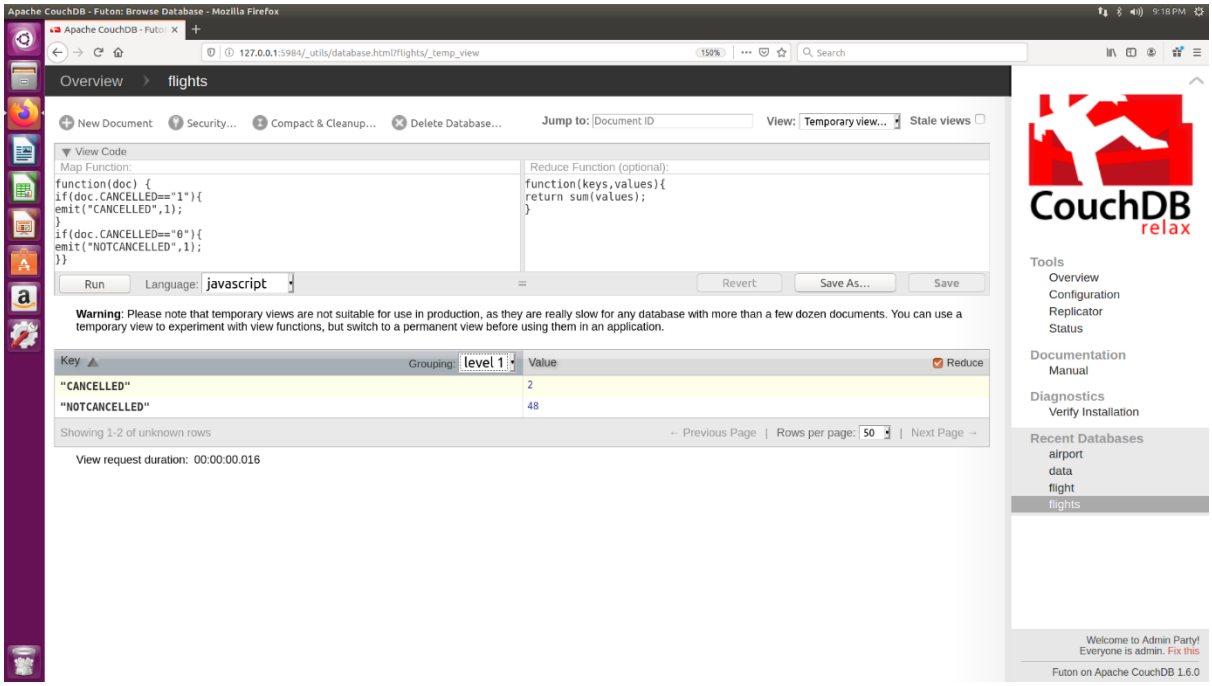

Apache CouchDB - Futon: Browse Database - Mozilla Firefox
Overview
flights
New Document
Security...
Compact & Cleanup...
Delete Database...
Jump to: Document ID
View: Temporary view...
Stale views
View Code
Map Function:
function(doc) {
if(doc.AIRLINE=="AA")
{
emit("AA",1);
}
if(doc.AIRLINE=="AS")
{
Reduce Function (optional):
function(keys,values){
return sum(values);
}
Run
Language: javascript
Revert
Save As...
Save
Warning: Please note that temporary views are not suitable for use in production, as they are really slow for any database with more than a few dozen documents. You can use a temporary view to experiment with view functions, but switch to a permanent view before using them in an application.
Key
Grouping: level 1
Value
Reduce
"AA" 11
"AS" 8
"B6" 2
"DL" 10
"EV" 1
"HA" 1
"NK" 9
"UA" 3
"US" 5
Showing 1-9 of unknown rows
← Previous Page | Rows per page: 50 | Next Page →
View request duration: 00:00:00.049
CouchDB relax
Tools
Overview
Configuration
Replicator
Status
Documentation
Manual
Diagnostics
Verify Installation
Recent Databases
airport
data
flight
flights
Welcome to Admin Party!
Everyone is admin. Fix this
Futon on Apache CouchDB 1.6.0

3. Retrieve the total number of flights from the database which were cancelled and which were not cancelled.(Cancelled=1.Not Cancelled=0)

Map :

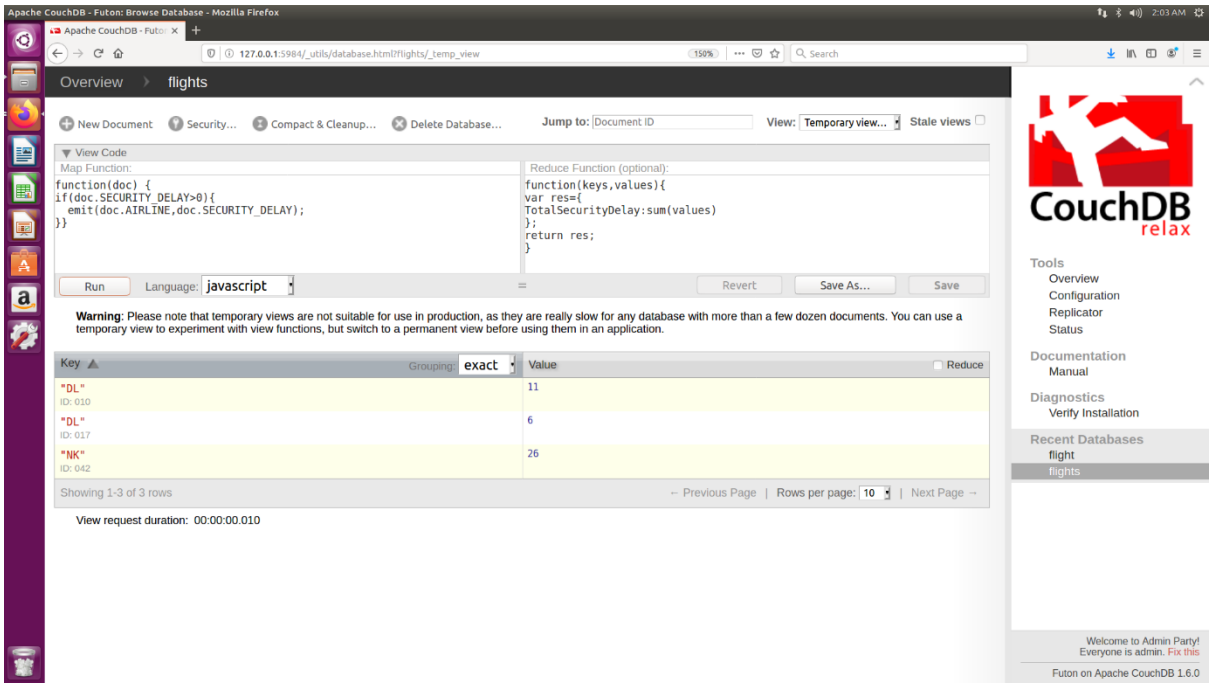


Reduce :

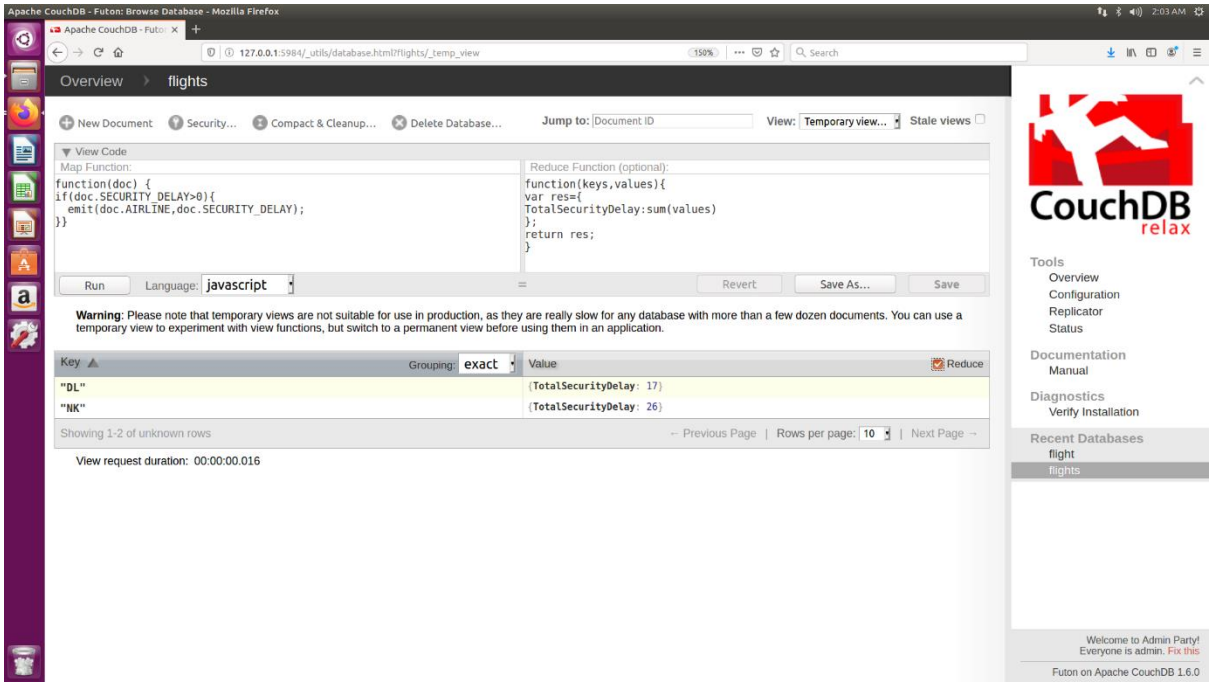


4. Retrieve the airline name which were delayed due to security and get the total no of flights and total delay caused by those flights.

Map :

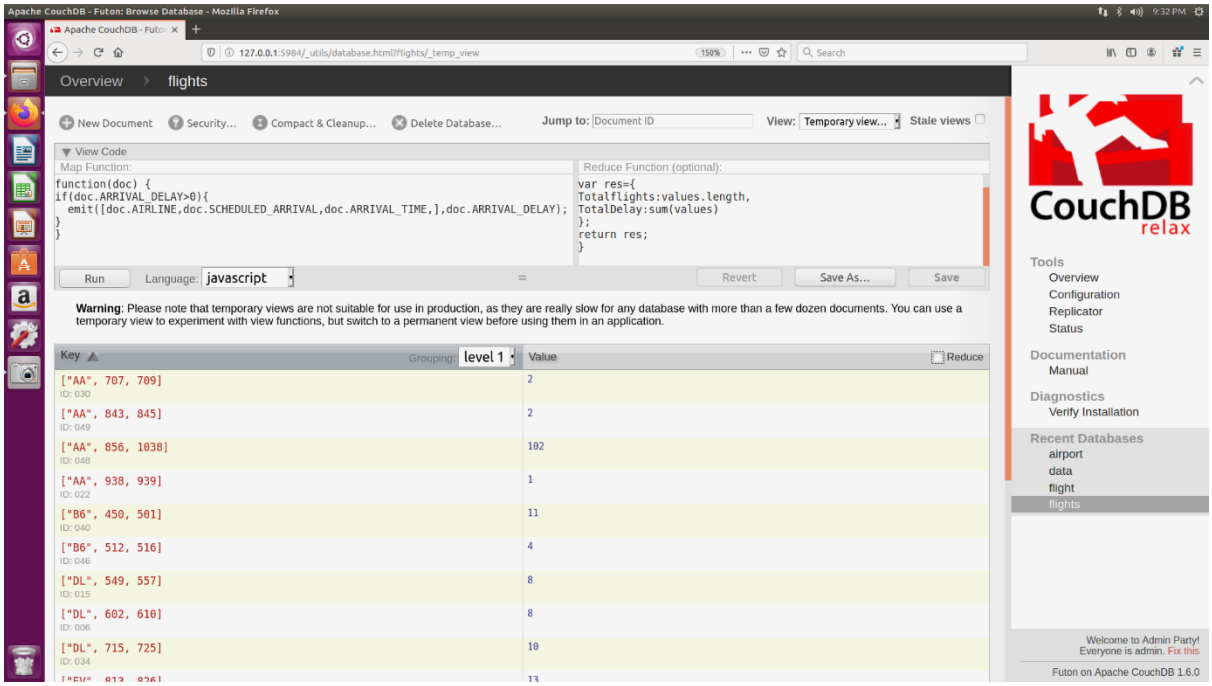


Reduce :

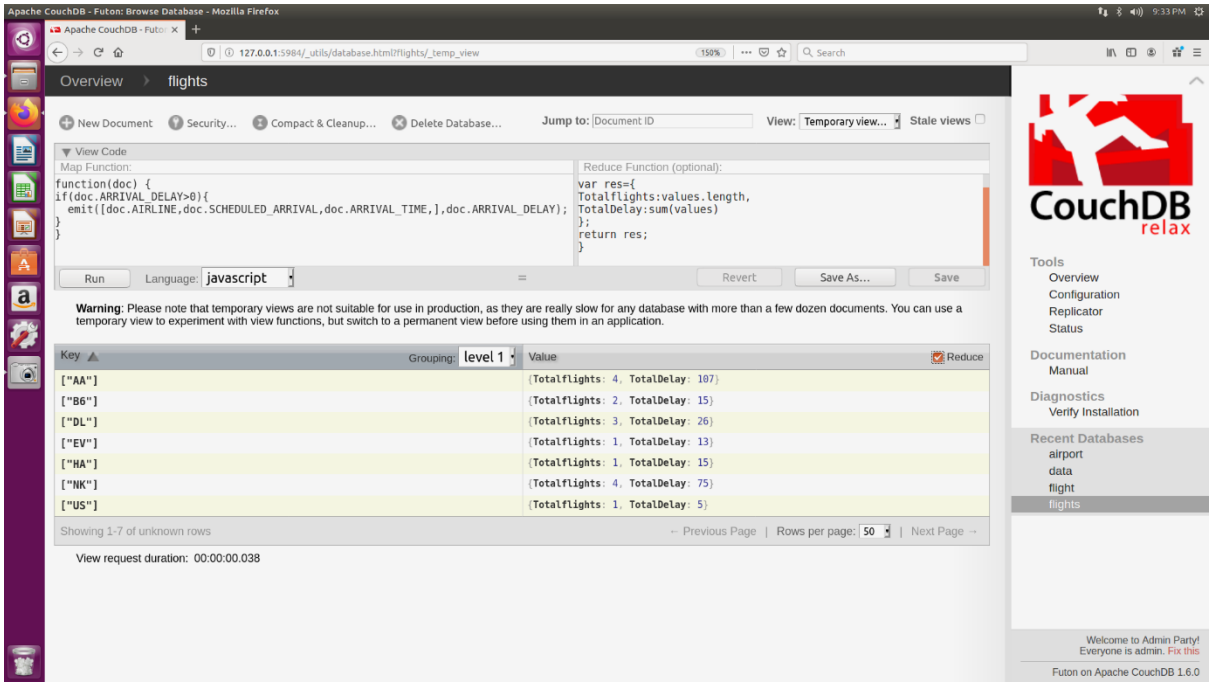


5. Retrieve the airline name, scheduled arrival, arrival time and arrival delay of those flights which arrived late and get the total count of each airline along with total delay caused by those airlines.

Map :

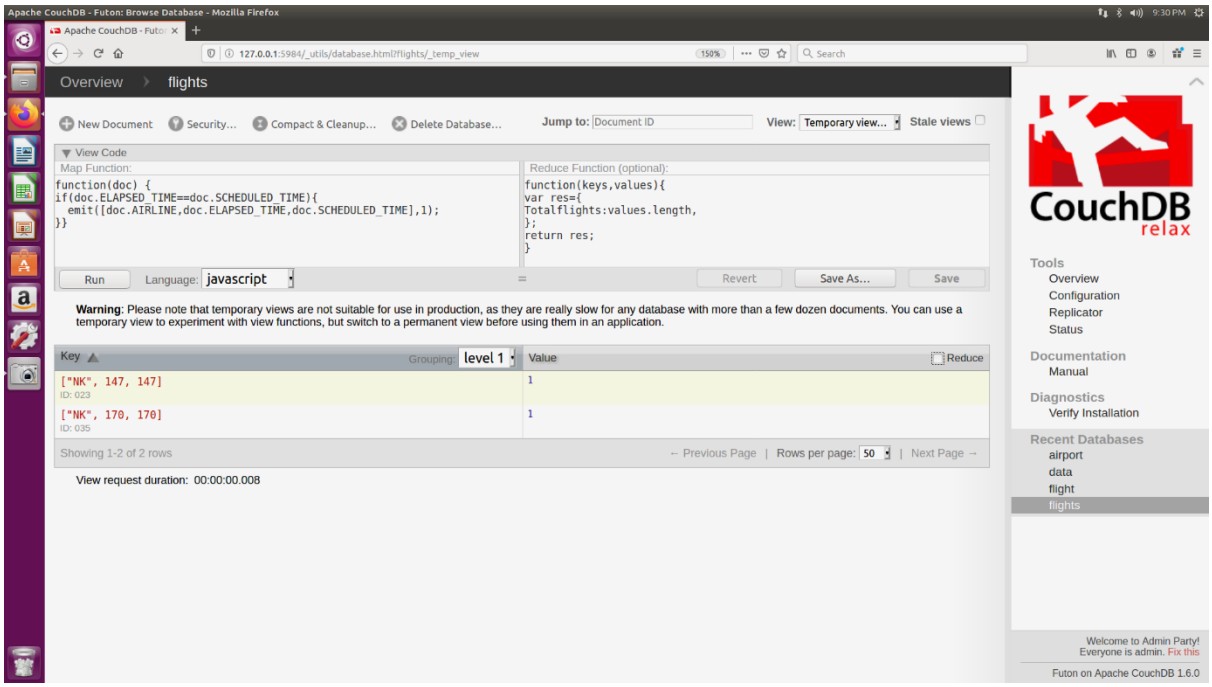


Reduce :

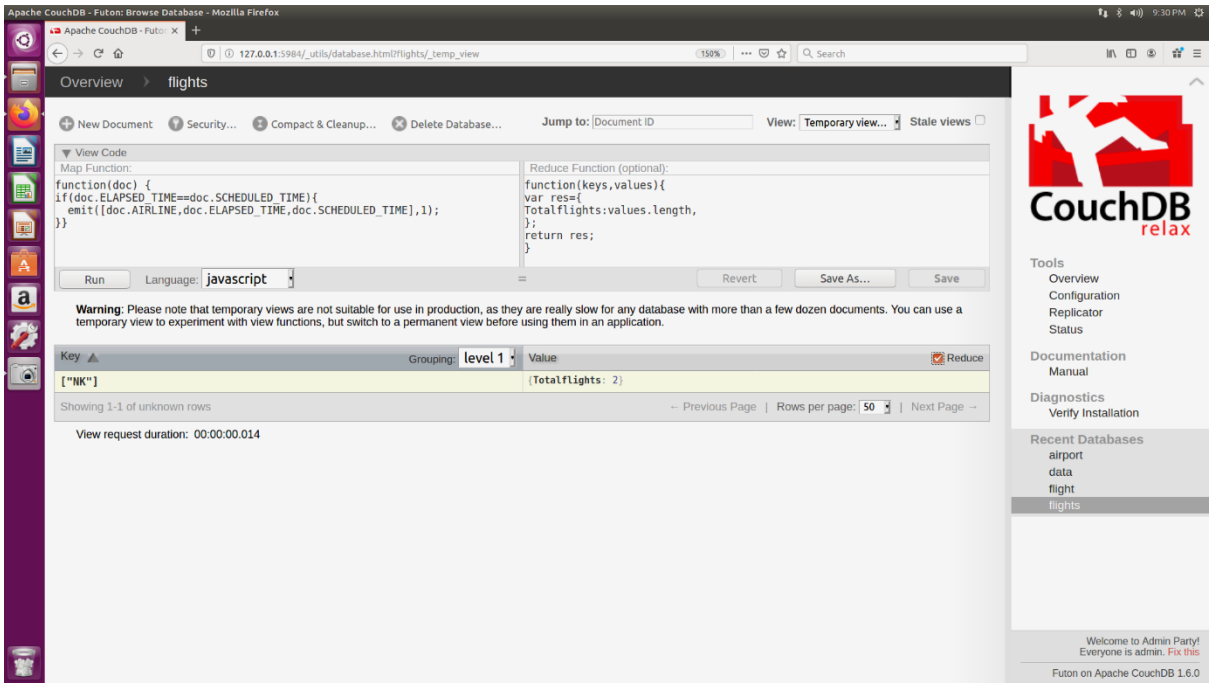


6. Retrieve all the airline names whose elapsed time is equal to scheduled time and get the count of total flights .

Map:



Reduce:



CONCLUSION

Big data is not just an emerging trend but also a necessity. In Flight Data Analysis, it analyses the variety of factors responsible and associated with flight delays for different airlines. Flight delays cause a lot of inconvenience to passengers. It could make them late to their scheduled events or miss a connecting flight, thus leading to anger and frustration. Also, passengers may not always be entitled to compensation when a delay occurs. Big data and the airline industry have a long way to go.

REFERENCES

- https://www.kaggle.com/usdot/flight-delays
- https://www.ionos.com/community/hosting/couchdb/install-and-use-couchdb-on-ubuntu-1604/
- https://en.m.wikipedia.org/wiki/Apache_CouchDB
- https://docs.couchdb.org/en/stable/ddocs/views/intro.html