
# *BIGDATA HADOOP*
# *USING*
# *IBM INFOSPHERE BIGINSIGHTS*

**Submitted to:**

**Mr. Prashant Gangwar**

**Submitted by:**

Tanya Ghumman
Shivani Sinha
Sonali Anand
Ankush Tomar
Ayush Talreja

**BIG DATA**

Big data is a term that describes the large volume of data – both structured and unstructured – that inundates a business on a day-to-day basis. But it's not the amount of data that's important. Its what organizations do with the data that matters. Big data can be analyzed for insights that lead to better decisions and strategic business moves.

# Big Data History and Current Considerations

While the term "big data" is relatively new, the act of gathering and storing large amounts of information for eventual analysis is ages old. The concept gained momentum in the early 2000s when industry analyst Doug Laney articulated the now-mainstream definition of big data as the three Vs:

**Volume:** Organizations collect data from a variety of sources, including business transactions, social media and information from sensor or machine-to-machine data. In the past, storing it would've been a problem – but new technologies (such as Hadoop) have eased the burden.

**Velocity:** Data streams in at an unprecedented speed and must be dealt with in a timely manner. RFID tags, sensors and smart metering are driving the need to deal with torrents of data in near-real time.

**Variety:** Data comes in all types of formats – from structured, numeric data in traditional databases to unstructured text documents, email, video, audio, stock ticker data and financial transactions.

At SAS, we consider two additional dimensions when it comes to big data:

**Variability:** In addition to the increasing velocities and varieties of data, data flows can be highly inconsistent with periodic peaks. Is something trending in social media? Daily, seasonal and event-triggered peak data loads can be challenging to manage. Even more so with unstructured data.

**Complexity:** Today's data comes from multiple sources, which makes it difficult to link, match, cleanse and transform data across systems. However, it's

necessary to connect and correlate relationships, hierarchies and multiple data linkages or your data can quickly spiral out of control.

# Objectives of Big Data

In this section, we will explain the objectives that we strive to achieve while dealing with Big Data:

Cost reduction- MIPS (Million Instructions Per Second) and above terabyte storage for structured data are now cheaply delivered through big data technologies like Hadoop clusters. This is mainly because of the ability of Big Data technologies to utilize commodity scale hardware for processing by employing techniques such as data sharing, distributed computing etc.

Faster processing speeds- Big data technologies have also helped reducing large scale-analytics processing from hours to minutes. These technologies have also been instrumental in real time analytics reducing processing times to seconds.

Big Data based offerings- Big data technologies have enabled organizations to leverage big data in developing new product and service offerings. The best example may be LinkedIn, which has used big data and data scientists to develop a broad array of product offerings and features, including People You May Know, Groups You May Like, Jobs You May Be Interested In, Who has Viewed My Profile, and several others. These offerings have brought millions of new customers to LinkedIn.

Supporting internal business decisions- Just like traditional data analytics, big data analytics can be employed to support business decisions when there are new and less structured data sources.

For example, any data that can shed light on customer satisfaction is helpful, and much data from customer interactions is unstructured such as website clicks, transaction records, and voice recordings from call centres.

# WHY IS BIG DATA IMPORTANT?

The importance of big data doesn't revolve around how much data you have, but what you do with it. You can take data from any source and analyze it to find answers that enable 1) cost reductions, 2) time reductions, 3) new product development and optimized offerings, and 4) smart decision making. When you combine big data with high-powered analytics, you can accomplish business-related tasks such as:

- Determining root causes of failures, issues and defects in near-real time.
- Generating coupons at the point of sale based on the customer's buying habits.
- Recalculating entire risk portfolios in minutes.
- Detecting fraudulent behavior before it affects your organization.

# HOW IT WORKS:

BEFORE DISCOVERING HOW BIG DATA CAN WORK FOR YOUR BUSINESS, YOU SHOULD FIRST UNDERSTAND WHERE IT COMES FROM. THE SOURCES FOR BIG DATA GENERALLY FALL INTO ONE OF THREE CATEGORIES:

## Streaming Data:

This category includes data that reaches your IT systems from a web of connected devices. You can analyze this data as it arrives and make decisions on what data to keep, what not to keep and what requires further analysis.

## Social Media Data:

The data on social interactions is an increasingly attractive set of information, particularly for marketing, sales and support functions. It's often in unstructured or semi structured forms, so it poses a unique challenge when it comes to consumption and analysis.

## Publicly Available Sources:

Massive amounts of data are available through open data sources like the US government's data.gov, the CIA World Facebook or the European Union Open Data Portal.

After identifying all the potential sources for data, consider the decisions you'll need to make once you begin harnessing information.

These include:

### **Storage and Management**:
Whereas storage would have been a problem several years ago, there are now low-cost options for storing data if that's the best strategy for your business.

### **How much of it to analyze**:
Some organizations don't exclude any data from their analyses, which is possible with today's high-performance technologies such as grid computing or in-memory analytics. Another approach is to determine upfront which data is relevant before analyzing it.

### **How to use any insights you uncover**:
The more knowledge you have, the more confident you'll be in making business decisions. It's smart to have a strategy in place once you have an abundance of information at hand.

THE FINAL STEP IN MAKING BIG DATA WORK FOR YOUR BUSINESS IS TO RESEARCH THE TECHNOLOGIES THAT HELP YOU MAKE THE MOST OF BIG DATA AND BIG DATA ANALYTICS. CONSIDER:

- Cheap, abundant storage.
- Faster processors.
- Affordable open source, distributed big data platforms, such as Hadoop.
- Parallel processing, clustering, MPP, virtualization, large grid environments, high connectivity and high throughputs.
- Cloud computing and other flexible resource allocation arrangements.

# REAL FACTS:

- New York Stock Exchange generates 1 TB/day.
- Google processes 700 PB/month.
- Facebook hosts 10 billion photos taking 1 PB of storage.

# BIG DATA CHALLENGES:

The major challenges associated with big data are as follows:

- Capturing data
- Curation
- Storage
- Searching
- Sharing
- Transfer
- Analysis
- Presentation

# BIG DATA SOLUTIONS

## Traditional Enterprise Approach

In this approach, an enterprise will have a computer to store and process big data. For storage purpose, the programmers will take the help of their choice of database vendors such as Oracle, IBM, etc. In this approach, the user interacts with the application, which in turn handles the part of data storage and analysis.

## Limitation

This approach works fine with those applications that process less voluminous data that can be accommodated by standard database servers, or up to the limit of the processor that is processing the data. But when it comes to dealing with huge amounts of scalable data, it is a hectic task to process such data through a single database bottleneck.

## Google's Solution

Google solved this problem using an algorithm called MapReduce. This algorithm divides the task into small parts and assigns them to many computers, and collects the results from them which when integrated, form the result dataset.

## Hadoop

Using the solution provided by Google, Doug Cutting and his team developed an Open Source Project called HADOOP.
Hadoop runs applications using the MapReduce algorithm, where the data is processed in parallel with others. In short, Hadoop is used to develop applications that could perform complete statistical analysis on huge amounts of data.

# Data Lake

- One of the strong use case of Big Data technologies is to analyze the data, and find out the hidden patterns and information out of it. For this to be effective, all the data from sources must be saved without any loss or tailoring. However traditional RDBMS databases and most of NoSQL storage systems require data to be transformed to a specific format in order to be utilized - adding, updating, searching - effectively.
- Data Lake concept is introduced to fill this gap and talks about storing the data in raw state (same state as data exist in source systems) without any data loss and transformation. For the same reason, Data Lake is also referred as Data Landing Area.
- Data Lake is rather a concept and can be implemented using any suitable technology/software that can hold the data in any form along with ensuring that no data loss is occurred using distributed storage providing failover.
- Example of such a technology would be Apache Hadoop where its MapReduce component could be used to load data into its distributed file system known as Hadoop Distributed File System (HDFS).
- As we can see that there are two differences in the process. Firstly, we have ETL component instead of data loader which emphasizes that in case of data warehouse.
- Input data is transformed and tailored to a pre-defined schema in order to be saved to data warehouse.
- This process of ETL, in most cases, results into data loss due to fixed schemas. Second difference is that in case of data warehouse, there are no data processors as data is already in a pre-defined schema ready to be consumed by data analysts.

# Benefits of Data Lake

There are following benefits that companies can reap by implementing Data Lake

- Data Consolidation - Data Lake enales enterprises to consolidate its data available in various forms such as videos, customer care recordings, web logs, documents etc. in one place which was not possible with traditional approach of using data warehouse.

- Schema-less and Format-free Storage - Data Lake talks about the storage of data in its raw form i.e. same format as it is sent from source systems. This eliminates need of source systems having to emit data in a pre-defined schema.

- No Data Loss - Since Data Lake doesn't require source systems to emit the data in a pre-defined schema, source systems do not need to tailor the data. This enables the access of all the data to data analysts and scientists resulting into more accurate analysis.

- Cost Effectiveness - Data Lake talks about distributed storage wherein commodity hardware can be utilized to store the huge volumes of data. procuring and levearing the commodity hardware for storage is much cost effective than using the high configuration hardware.

# Challenges with Data Lake

- Ability to accommodate any format and type of data sometimes can convert Data Lake into a data mess referred as Data Swarm. Hence, it is important that enterprise take caution while implementing Data Lake to ensure that data is property maintained and accessible in Data Lake.

- Another challenge with Data Lake is that since it stores the raw data, each type of analysis requires the data transformation and tailoring from scratch requiring additional processing infrastructure every time you want to do some analysis on data stored in Data Lake.

# Data Warehouse

A data warehouse is a subject-oriented, integrated, time-variant and non-volatile collection of data in support of management's decision making process - Bill Inmon.

William H. Inmon (born 1945) is an American computer scientist, recognized by many as the father of the data warehouse

In computing, a **data warehouse** (**DW** or **DWH**), also known as an **enterprise data warehouse** (**EDW**), is a system used for reporting and data analysis.

- **Subject-Oriented**: A data warehouse can be used to analyze a particular subject area. For example, "sales" can be a particular subject.
- **Integrated**: A data warehouse integrates data from multiple data sources. For example, source A and source B may have different ways of identifying a product, but in a data warehouse, there will be only a single way of identifying a product.
- **Time-Variant**: Historical data is kept in a data warehouse. For example, one can retrieve data from 3 months, 6 months, 12 months, or even older data from a data warehouse. This contrasts with a transactions system, where often only the most recent data is kept. For example, a transaction system may hold the most recent address of a customer, where a data warehouse can hold all addresses associated with a customer.
- **Non-volatile**: Once data is in the data warehouse, it will not change. So, historical data in a data warehouse should never be altered.

# Data Mart

- A data mart is the access layer of the data warehouse environment that is used to get data out to the users.
- A database, or collection of databases, designed to help managers make strategic decisions about their business. Whereas a data warehouse combines databases across an entire enterprise, data marts are usually smaller and focus on a particular subject or department. Some data marts, called dependent data marts, are subsets of larger data warehouses.

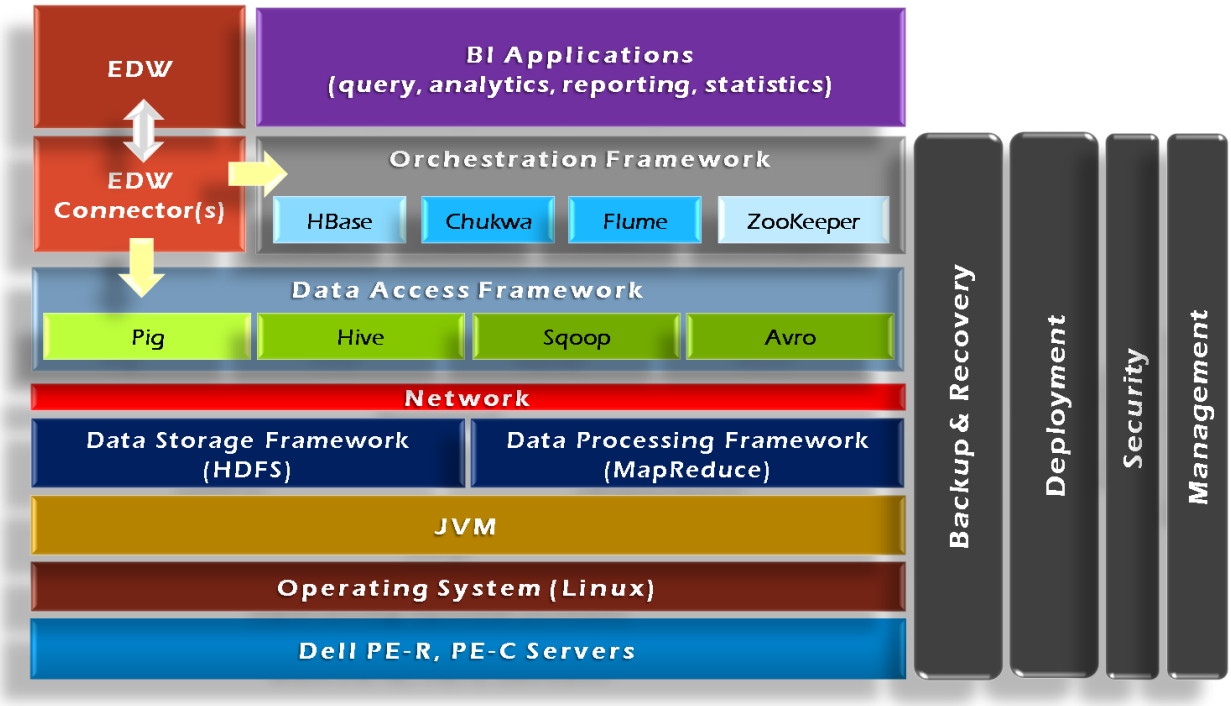
# Introduction to Hadoop

An open-source software framework that supports data-intensive distributed applications, licensed under the Apache v2license.

**Goals / Requirements:**

- Abstract and facilitate the storage and processing of large and/or rapidly growing data sets
- Structured and non-structured data
- Simple programming models
- High scalability and availability
- Use commodity (cheap!) hardware with little redundancy
- Fault-tolerance
- Move computation rather than data

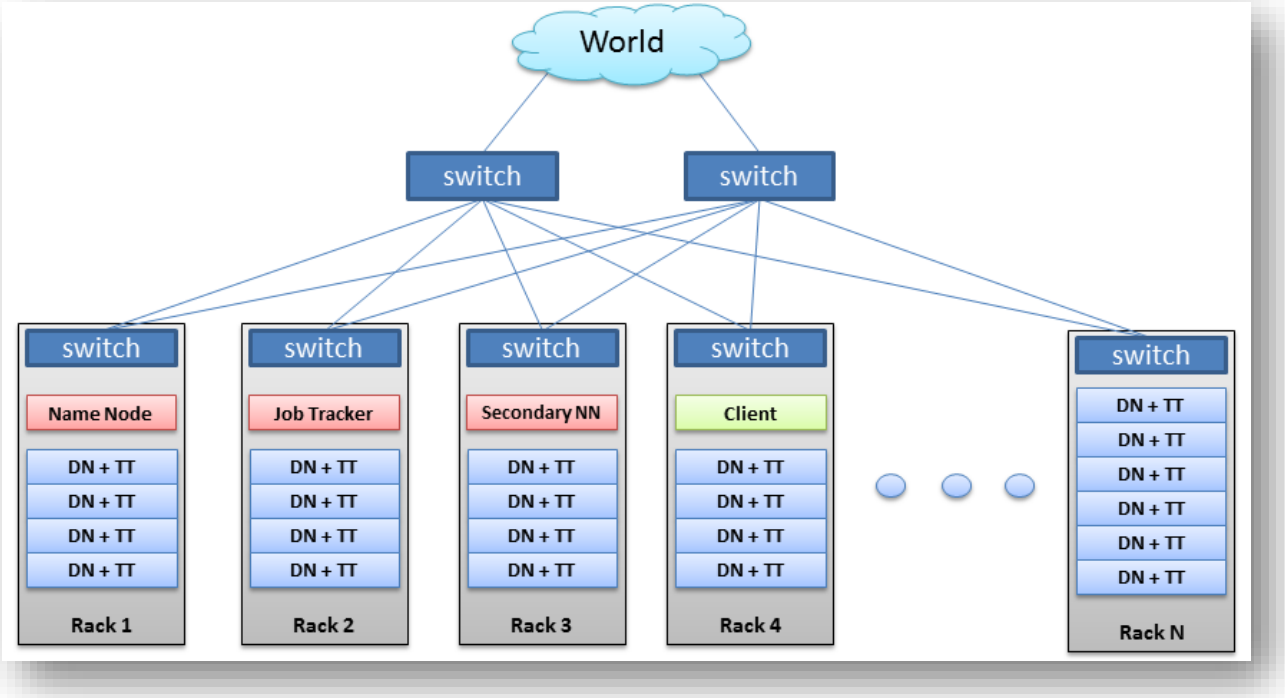
# Hadoop Framework Tools




# Hadoop Architecture (HDFS)

- Distributed, with some centralization

- Main nodes of cluster are where most of the computational power and storage of the system lies
- Main nodes run Task Tracker to accept and reply to MapReduce tasks, and also Data Node to store needed blocks closely as possible
- Central control node runs Name Node to keep track of HDFS directories & files, and Job Tracker to dispatch compute tasks to Task Tracker
- Written in Java, also supports Python and Ruby




- Hadoop Distributed Filesystem
- Tailored to needs of MapReduce
- Targeted towards many reads of filestreams
- Writes are more costly
- High degree of data replication (3x by default)
- No need for RAID on normal nodes

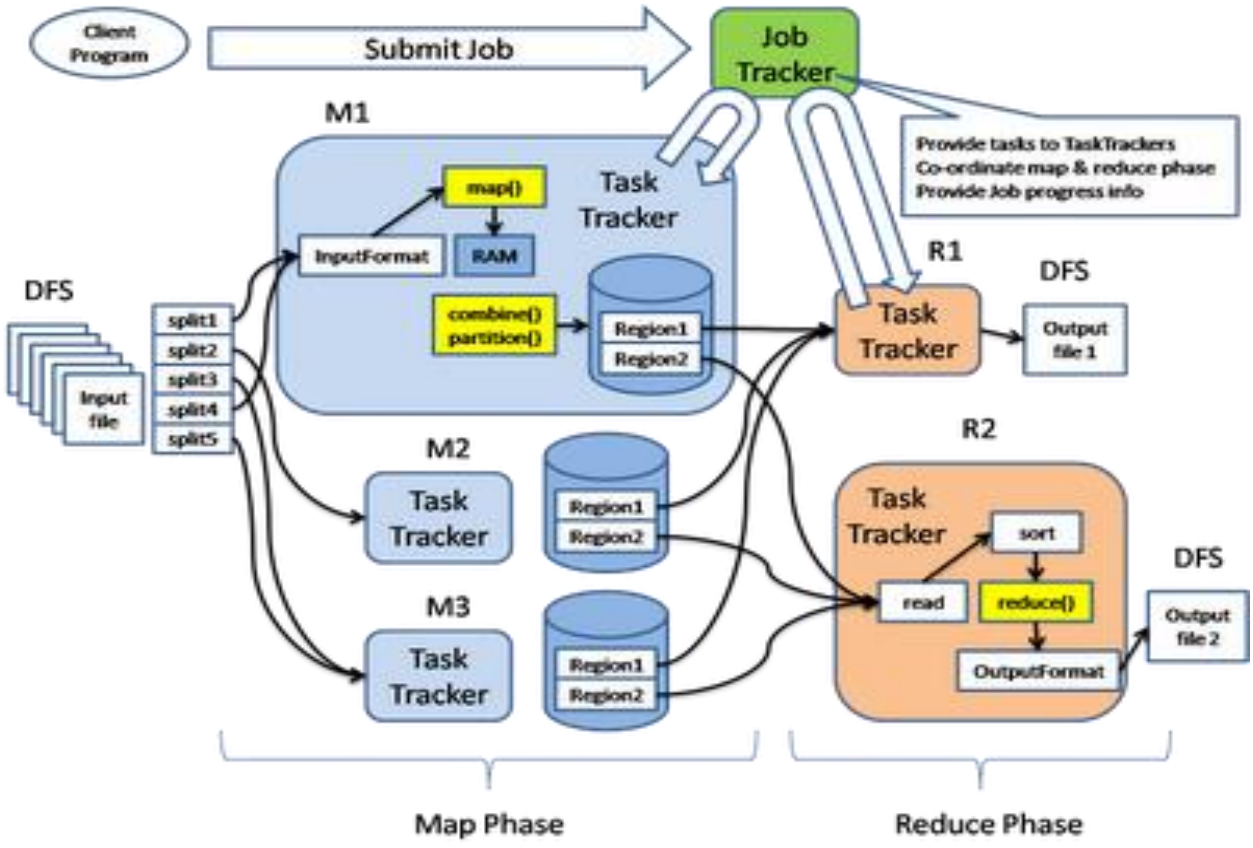
- Large blocksize (64MB)
- Location awareness of DataNodes in network

## NameNode:

- Stores metadata for the files, like the directory structure of a typical FS.
- The server holding the NameNode instance is quite crucial, as there is only one.
- Transaction log for file deletes/adds, etc. Does not use transactions for whole blocks or file-streams, only metadata.
- Handles creation of more replica blocks when necessary after a DataNode failure

## DataNode:

- Stores the actual data in HDFS
- Can run on any underlying filesystem (ext3/4, NTFS, etc)
- Notifies NameNode of what blocks it has
- NameNode replicates blocks 2x in local rack, 1x elsewhere




## MapReduce Engine:

- JobTracker & TaskTracker
- JobTracker splits up data into smaller tasks("Map") and sends it to the TaskTracker process in each node
- TaskTracker reports back to the JobTracker node and reports on job progress, sends data ("Reduce") or requests new jobs
- None of these components are necessarily limited to using HDFS
- Many other distributed file-systems with quite different architectures work
- Many other software packages besides Hadoop's MapReduce platform make use of HDFS

**Hadoop is in use at most organizations that handle big data:**

- Yahoo!
- Facebook

- Amazon
- Netflix
- Etc…
- Some examples of scale:
- Yahoo!'s Search Webmap runs on 10,000 core Linux cluster and powers Yahoo! Web search

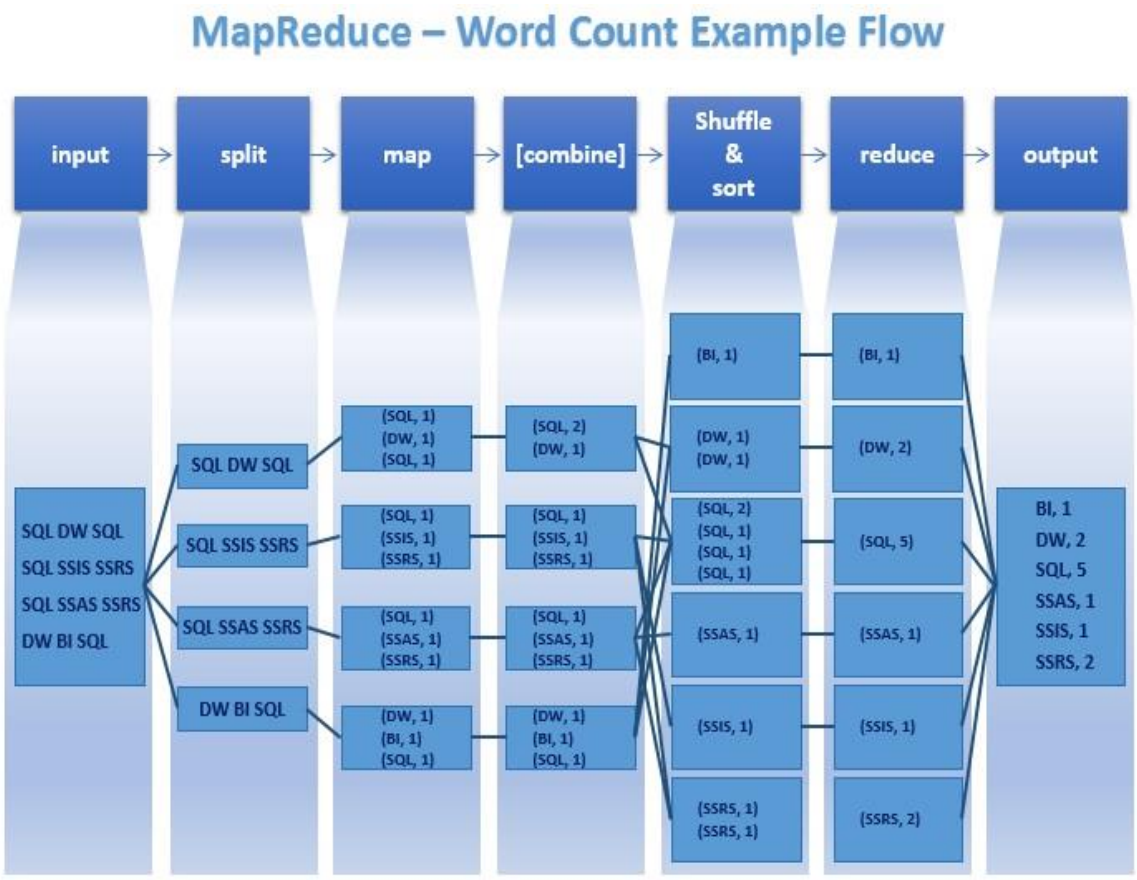
FB's Hadoop cluster hosts 100+ PB of data (July, 2012) & growing at ½ PB/day (Nov, 2012).

# Goals of HDFS

Fault detection and recovery: Since HDFS includes a large number of commodity hardware, failure of components is frequent. Therefore HDFS should have mechanisms for quick and automatic fault detection and recovery.
Huge datasets: HDFS should have hundreds of nodes per cluster to manage the applications having huge datasets.
Hardware at data: A requested task can be done efficiently, when the computation takes place near the data. Especially where huge datasets are involved, it reduces the network traffic and increases the throughput.

# MapReduce

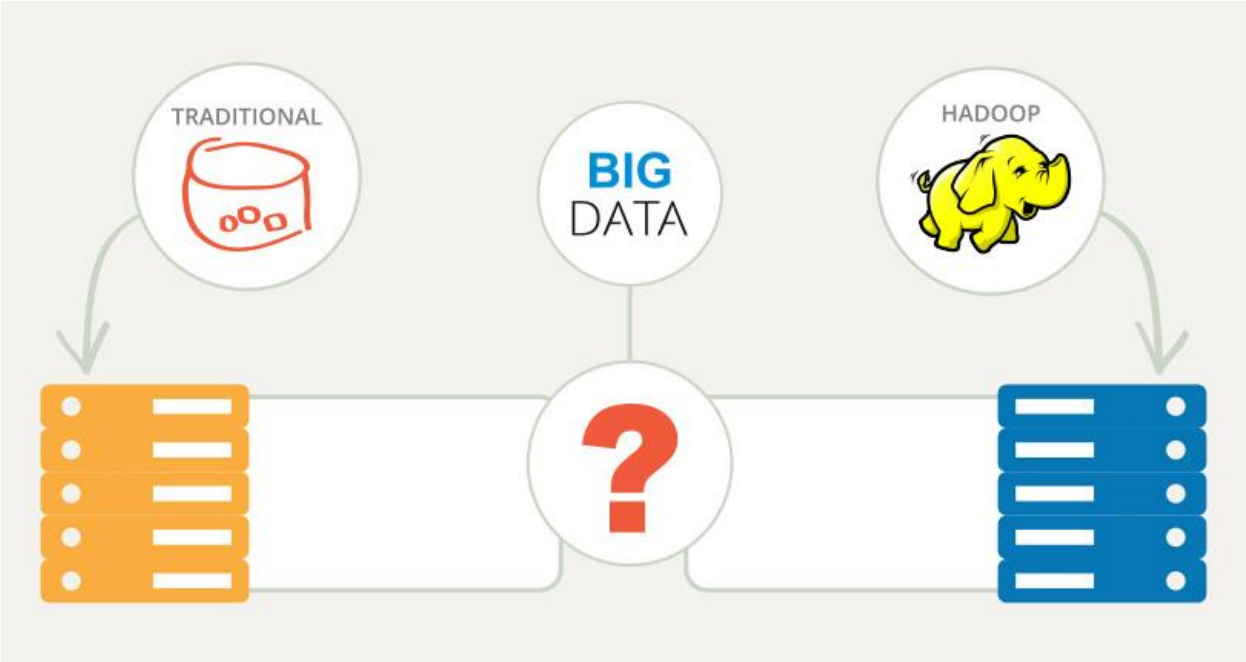
MapReduce is a programming model and an associated implementation for processing and generating large data sets with a parallel, distributed algorithm on a cluster. Conceptually similar approaches have been very well known since 1995 with the Message Passing Interface standard having reduce and scatter operations.




# Advantages of Hadoop

- Hadoop framework allows the user to quickly write and test distributed systems. It is efficient, and it automatic distributes the data and work across the machines and in turn, utilizes the underlying parallelism of the CPU cores.
- Hadoop does not rely on hardware to provide fault-tolerance and high availability (FTHA), rather Hadoop library itself has been designed to detect and handle failures at the application layer.
- Servers can be added or removed from the cluster dynamically and Hadoop continues to operate without interruption.
- Another big advantage of Hadoop is that apart from being open source, it is compatible on all the platforms since it is Java based.

# Hadoop vs. Traditional Database:




Today's ultra-connected world is generating massive volumes of data at ever-accelerating rates. As a result, big data analytics has become a powerful tool for businesses looking to leverage mountains of valuable data for profit and competitive advantage. In the midst of this big data rush, Hadoop, as an on-premise or cloud-based platform has been heavily promoted as the one-size fits all solution for the business world's big data problems. While Hadoop has lived up to much of the hype, there are certain situations where running workloads on a traditional database may be the better solution.

For companies wondering which functionality will better serve their big data use case needs, here are some key questions that need to be asked when choosing between Hadoop databases – including cloud-based Hadoop services such as Qubole – and a traditional database.

## Is Hadoop a Database?

Hadoop is not a database, but rather an open source software framework specifically built to handle large volumes of structure and semi-structured data. Organizations considering Hadoop adoption should evaluate whether their current or future data needs require the type of capabilities Hadoop offers.

## Is the data being analyzed structured or unstructured?

**Structured Data**: Data that resides within the fixed confines of a record or file is known as structured data. Owing to the fact that structured data – even in large volumes – can be entered, stored, queried, and analyzed in a simple and straightforward manner, this type of data is best served by a traditional database.

**Unstructured Data:** Data that comes from a variety of sources, such as emails, text documents, videos, photos, audio files, and social media posts, is referred to as unstructured data. Being both complex and voluminous, unstructured data cannot be handled or efficiently queried by a traditional database. Hadoop's ability to join, aggregate, and analyze vast stores of multi-source data without having to structure it first allows organizations to gain deeper insights quickly. Thus Hadoop is a perfect fit for companies looking to store, manage, and analyze large volumes of unstructured data.

## Is a scalable analytics infrastructure needed?

Companies whose data workloads are constant and predictable will be better served by a traditional database.

Companies challenged by increasing data demands will want to take advantage of Hadoop's scalable infrastructure. Scalability allows servers to be added on demand to accommodate growing workloads. As a cloud-based Hadoop service, Qubole offers more flexible scalability by spinning virtual servers up or down within minutes to better accommodate fluctuating workloads.

Will a Hadoop Database implementation be cost-effective?
Cost-effectiveness is always a concern for companies looking to adopt new technologies. When considering a Hadoop implementation, companies need to do their homework to make sure that the realized benefits of a Hadoop deployment outweigh the costs. Otherwise it would be best to stick with a traditional database to meet data storage and analytics needs.

All things considered, Hadoop has a number of things going for it that make implementation more cost-effective than companies may realize. For one thing, Hadoop saves money by combining open source software with commodity servers. Cloud-based Hadoop platforms such as Qubole reduce costs further by eliminating the expense of physical servers and warehouse space.

Hybrid systems, which integrate Hadoop platforms with traditional relational databases, are gaining popularity as cost-effective ways for companies to leverage the benefits of both platforms.

### Is fast data analysis critical?

Hadoop was designed for large distributed data processing that addresses every file in the database. And that type of processing takes time. For tasks where fast performance isn't critical, such as running end-of-day reports to review daily transactions, scanning historical data, and performing analytics where a slower time-to-insight is acceptable, Hadoop is ideal.

On the other hand, in cases where organizations rely on time-sensitive data analysis, a traditional database is the better fit. That's because shorter time-to - insight isn't about analyzing large unstructured datasets, which Hadoop does so well. It's about analyzing smaller data sets in real or near-real time, which is what traditional databases are well equipped to do.

Hybrid systems are also a good fit to consider, as they allow companies to use traditional databases to run small, highly interactive workloads while using Hadoop to process huge, complex data sets.

### Which approach is best?

That all depends. While the benefits of big data analytics in providing deeper insights that lead to competitive advantage are real, those benefits can only be realized by companies that exercise due diligence in making sure that Hadoop is the analytics tool that best serves their needs.

# IBM INFOSPHERE

- It is a software platform designed to help firms discover and analyze business insights hidden in large volumes of a diverse range of data—data that's often ignored or discarded because it's too impractical or difficult to process using traditional means.
- Examples of such data include log records, click streams, social media data, news feeds, electronic sensor output, and even some transactional data. To help firms derive value from such data in an efficient manner, BigInsights incorporates several open source projects (including Apache™ Hadoop™) and a number of IBM-developed technologies.

- IBM's big data platform, which includes software for processing streaming data and persistent data.
- BigInsights and InfoSphere can be deployed together to support real-time and batch analytics of various forms of raw data, or they can be deployed individually to meet specific application objectives.

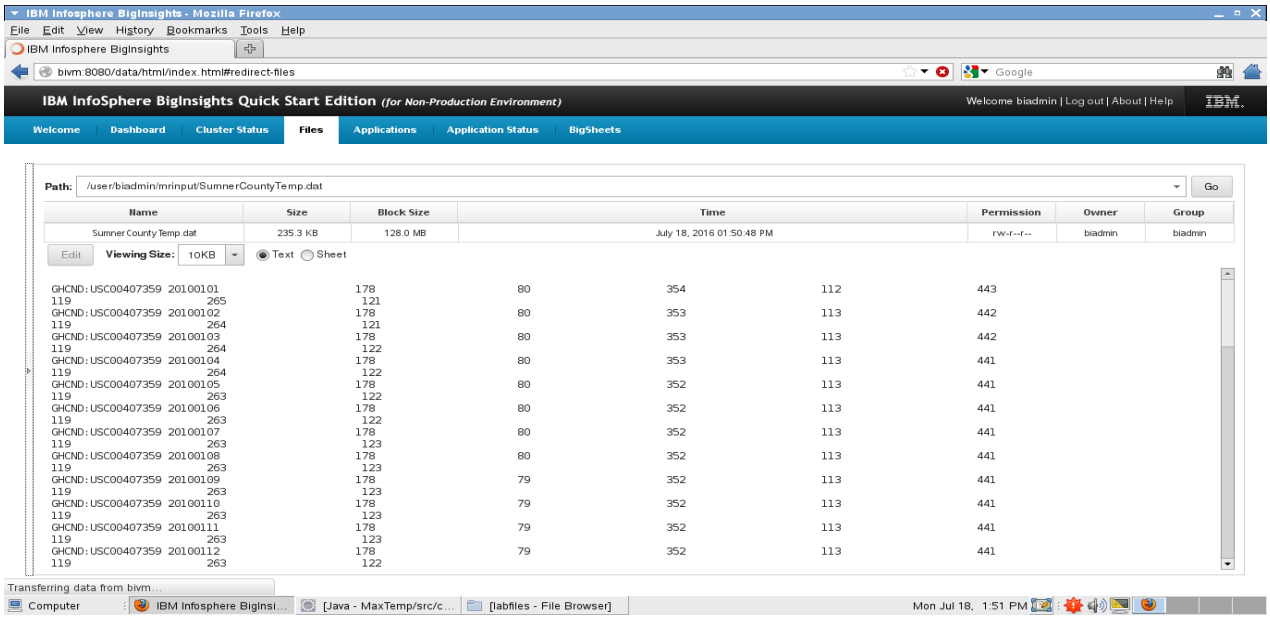
# MAPREDUCE PROGRAM ANALYSIS

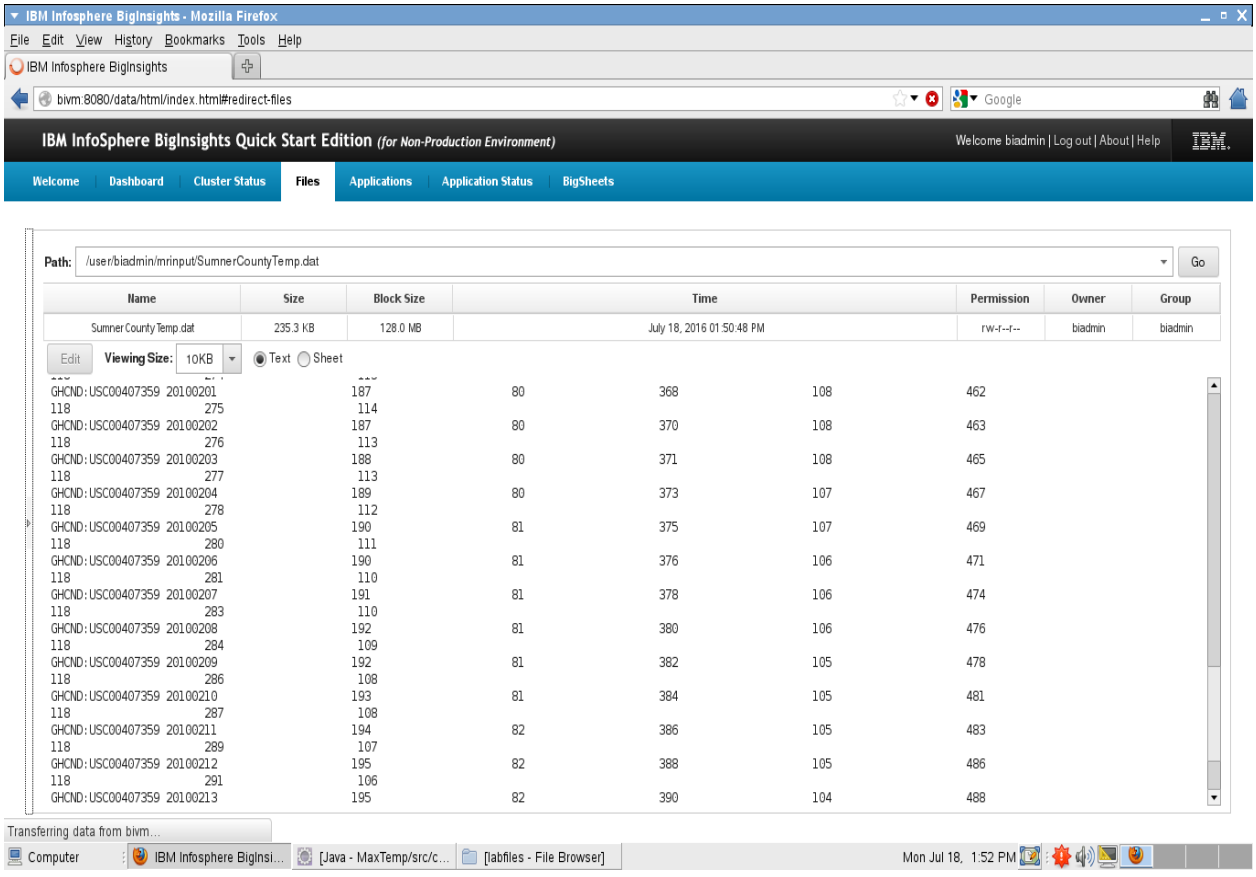
# ANALYSIS-1

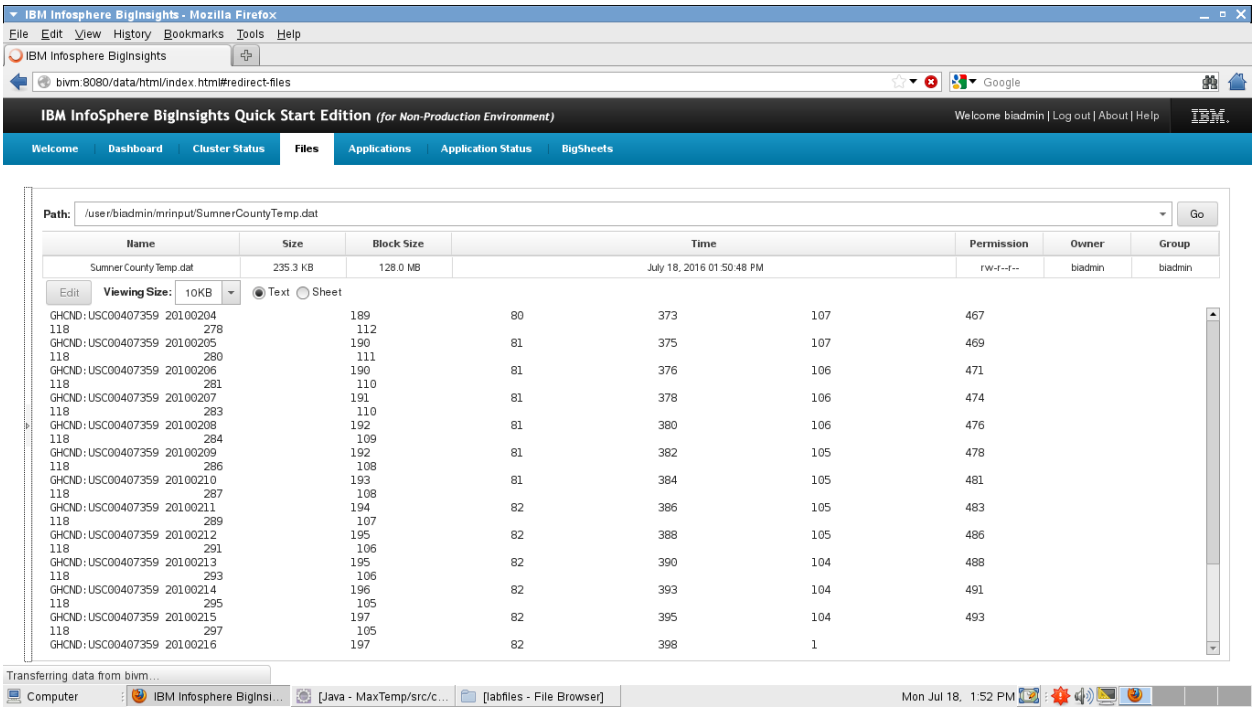
AIM: - Develops a MapReduce application that finds the highest average monthly temperature using the Java perspective in Eclipse.
This analysis perform using the InfoSphere BigInsights 2.1 Quick Start Edition, but has been tested on the 3.0 image.

## 1.1 Start the BigInsights components

__ 1. Log into your BigInsights image with a userid of biadmin and a password of biadmin.
__ 2. Start your BigInsights components. Use the icon on the desktop.
__ 3. From a command line (right-click the desktop and select Open in Terminal) execute:
hadoop fs -mkdir TempData
__ 4. Upload some temperature data from the local file system.
hadoop fs -copyFromLocal /home/labfiles/SumnerCountyTemp.dat /user/biadmin/TempData
__ 5. You can view this data from the *Files* tab in the Web Console or by executing the following command. The values in the 95th column (354, 353, 353,353, 352...) are the average daily temperatures. They are the result of multiplying the actual average temperature value times 10. (That way you don’t have to worry about working with decimal points.)
hadoop fs -cat TempData/SumnerCountyTemp.dat


IBM Infosphere BigInsights - Mozilla Firefox
File Edit View History Bookmarks Tools Help
IBM Infosphere BigInsights
bivm:8080/data/html/index.html#redirect-files
IBM InfoSphere BigInsights Quick Start Edition (for Non-Production Environment)
Welcome biadmin | Log out | About | Help
IBM
Welcome Dashboard Cluster Status Files Applications Application Status BigSheets
Path: /user/biadmin/mrinput/SumnerCountyTemp.dat
Go
Name Size Block Size Time Permission Owner Group
SumnerCountyTemp.dat 235.3 KB 128.0 MB July 18, 2016 01:50:48 PM rw-r--r-- biadmin biadmin
Edit Viewing Size: 10KB Text Sheet
GHCND:USC00407359 20100101 178 80 354 112 443
119 265 121
GHCND:USC00407359 20100102 178 80 353 113 442
119 264 121
GHCND:USC00407359 20100103 178 80 353 113 442
119 264 122
GHCND:USC00407359 20100104 178 80 353 113 441
119 264 122
GHCND:USC00407359 20100105 178 80 352 113 441
119 263 122
GHCND:USC00407359 20100106 178 80 352 113 441
119 263 122
GHCND:USC00407359 20100107 178 80 352 113 441
119 263 123
GHCND:USC00407359 20100108 178 80 352 113 441
119 263 123
GHCND:USC00407359 20100109 178 79 352 113 441
119 263 123
GHCND:USC00407359 20100110 178 79 352 113 441
119 263 123
GHCND:USC00407359 20100111 178 79 352 113 441
119 263 123
GHCND:USC00407359 20100112 178 79 352 113 441
119 263 122
Transferring data from bivm...
Computer
IBM Infosphere BigInsi...
[Java - MaxTemp/src/c...
[labfiles - File Browser]
Mon Jul 18, 1:51 PM


IBM Infosphere BigInsights - Mozilla Firefox
File Edit View History Bookmarks Tools Help
IBM Infosphere BigInsights
bivm:8080/data/html/index.html#redirect-files
IBM InfoSphere BigInsights Quick Start Edition (for Non-Production Environment)
Welcome biadmin | Log out | About | Help
IBM
Welcome Dashboard Cluster Status Files Applications Application Status BigSheets
Path: /user/biadmin/mrinput/SumnerCountyTemp.dat
Go
Name Size Block Size Time Permission Owner Group
SumnerCountyTemp.dat 235.3 KB 128.0 MB July 18, 2016 01:50:48 PM rw-r--r-- biadmin biadmin
Edit Viewing Size: 10KB Text Sheet
GHCND:USC00407359 20100201 187 80 368 108 462
118 275 114
GHCND:USC00407359 20100202 187 80 370 108 463
118 276 113
GHCND:USC00407359 20100203 188 80 371 108 465
118 277 113
GHCND:USC00407359 20100204 189 80 373 107 467
118 278 112
GHCND:USC00407359 20100205 190 81 375 107 469
118 280 111
GHCND:USC00407359 20100206 190 81 376 106 471
118 281 110
GHCND:USC00407359 20100207 191 81 378 106 474
118 283 110
GHCND:USC00407359 20100208 192 81 380 106 476
118 284 109
GHCND:USC00407359 20100209 192 81 382 105 478
118 286 108
GHCND:USC00407359 20100210 193 81 384 105 481
118 287 108
GHCND:USC00407359 20100211 194 82 386 105 483
118 289 107
GHCND:USC00407359 20100212 195 82 388 105 486
118 291 106
GHCND:USC00407359 20100213 195 82 390 104 488
Transferring data from bivm...
Computer
IBM Infosphere BigInsi...
[Java - MaxTemp/src/c...
[labfiles - File Browser]
Mon Jul 18, 1:52 PM




## 1.2 Define a Java project in Eclipse

__ 1. Start Eclipse using the icon on the desktop. Go with the default workspace.
__ 2. Make sure that you are using the Java perspective. Click Window->Open Perspective->Other. Then select Java. Click OK.
__ 3. Create a Java project. Select File->New->Java Project.
__ 4. Specify a *Project name* of MaxTemp. Click Finish.
__ 5. Right-click the *MaxTemp* project, scroll down and select Properties.
__ 6. Select Java Build Path.
__ 7. In the *Properties* dialog, select the Libraries tab.
__ 8. Click the Add Library pushbutton.
__ 9. Select *BigInsights Libraries* and click Next. Then click Finish. Then click OK.

1.3 Create a Java package and the mapper class

__ 1. In the *Package Explorer* expand *MaxTemp* and right-click src. Select New->Package.
__ 2. Type a *Name* of com.some.company. Click Finish.
__ 3. Right-click *com.some.company* and select New->Class.
__ 4. Type in a *Name* of MaxTempMapper. It will be a *public* class. Click Finish.

The data type for the input key to the mapper will be *LongWritable.* The data itself will be of type *Text.* The output key from the mapper will be of type *Text.* And the data from the mapper (the temperature) will be of type *IntWritable.*

__ 5. Your class:

__ a. You will need to import *java.io.IOException.*

__ b. Exend Mapper<LongWritable, Text, Text, IntWritable>

__ c. Define a public class called **map.**

__ d. Your code should look like the following:

```
package com.some.company;
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;
public class MaxTempMapper extends
Mapper<LongWritable, Text, Text, IntWritable> {
@Override
public void map(LongWritable key, Text value, Context context)
throws IOException, InterruptedException {
}
}
```

## 1.4 Complete the mapper

You are reading in a line of data. You will want to convert it to a string so that you can do some string manipulation. You will want to extract the month and average temperature for each record.

The month begins at the 22th character of the record (zero offset) and the average temperature begins at the 95th character. (Remember that the average temperature value is three digits.)

__ 1. In the *map* method, add the following code (or whatever code you think is required):

```
String line = value.toString();
String month = line.substring(22,24);
int avgTemp;
avgTemp = Integer.parseInt(line.substring(95,98));
context.write(new Text(month), new IntWritable(avgTemp));
```

__ 2. Save your work.

## 1.5 Create the reducer class

__ 1. In the *Package Explorer* right-click *com.some.company* and select New->Class.
__ 2. Type in a *Name* of MaxTempReducer. It will be a *public* class. Click Finish. The data type for the input key to the reducer will be *Text.* The data itself will be of type
*IntWritable.* The output key from the reducer will be of type *Text.* And the data from the
reducer will be of type *IntWritable.*
__ 3. Your class:
__ a. You will need to import *java.io.IOException.*
__ b. Extend Reducer<Text, LongWritable, Text, IntWritable>
__ c. Define a public class called reduce.
__ d. Your code should look like the following:

```
package com.some.company;
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;
public class MaxTempReducer extends
Reducer<Text, IntWritable, Text, IntWritable> {
@Override
public void reduce(Text key, Iterable<IntWritable> values, Context
context)
throws IOException, InterruptedException {
}
}
```

## 1.6 Complete the reducer

For the reducer, you want to iterate through all values for a given key. For each value, check
to see if it is higher than any of the other values.
__ 1. Add the following code (or your variation) to the *reduce* method.

```
int maxTemp = Integer.MIN_VALUE;
for (IntWritable value: values) {
maxTemp = Math.max(maxTemp, value.get());
}
context.write(key, new IntWritable(maxTemp));
```

__ 2. Save your work.

## 1.7 Create the mainclass

__ 1. In the Package Explorer right-click com.some.company and select New->Class.
__ 2. Type in a *Name* of MaxMonthTemp. It will be a public class. Click Finish.

The GenericOptionsParser() will extract any input parameters that are not system parameters and place them in an array. In your case, two parameters will be passed to your
application. The first parameter is the input file. The second parameter is the output
directory. (This directory must not exist or your MapReduce application will fail.)
Your code should look like this:

```
package com.some.company;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.util.GenericOptionsParser;
import com.some.company.MaxTempReducer;
import com.some.company.MaxTempMapper;
public class MaxMonthTemp {
public static void main(String[] args) throws Exception {
Configuration conf = new Configuration();
String[] programArgs =
new GenericOptionsParser(conf, args).getRemainingArgs();
if (programArgs.length != 2) {
System.err.println("Usage: MaxTemp <in> <out>");
System.exit(2);
}
Job job = new Job(conf, "Monthly Max Temp");
job.setJarByClass(MaxMonthTemp.class);
job.setMapperClass(MaxTempMapper.class);
job.setCombinerClass(MaxTempReducer.class);
job.setReducerClass(MaxTempReducer.class);
```

```
job.setOutputKeyClass(Text.class);
job.setOutputValueClass(IntWritable.class);
FileInputFormat.addInputPath(job, new Path(programArgs[0]));
FileOutputFormat.setOutputPath(job, new Path(programArgs[1]));
// Submit the job and wait for it to finish.
System.exit(job.waitForCompletion(true) ? 0 : 1);
}
}
```

__ 3. Save your work.

## 1.8 Create a JAR file

__ 1. In the *Package Explorer,* expand the *MaxTemp* project. Right-click *src* and select **Export.**

__ 2. Expand *Java* and select *JAR file.* Click **Next.**

__ 3. Click the *JAR file* **browse** pushbutton. Type in a name of **MyMaxTemp.jar**. Keep *biadmin*
as the folder. Click **OK.**

__ 4. Click **Finish.**

To save time, you are not going to run your application now. You will add a combiner class and
then run the application.

### 1.9 Run your application

__ 1. You need a command line. You may have to open a new one. Change to *biadmin's* home directory.

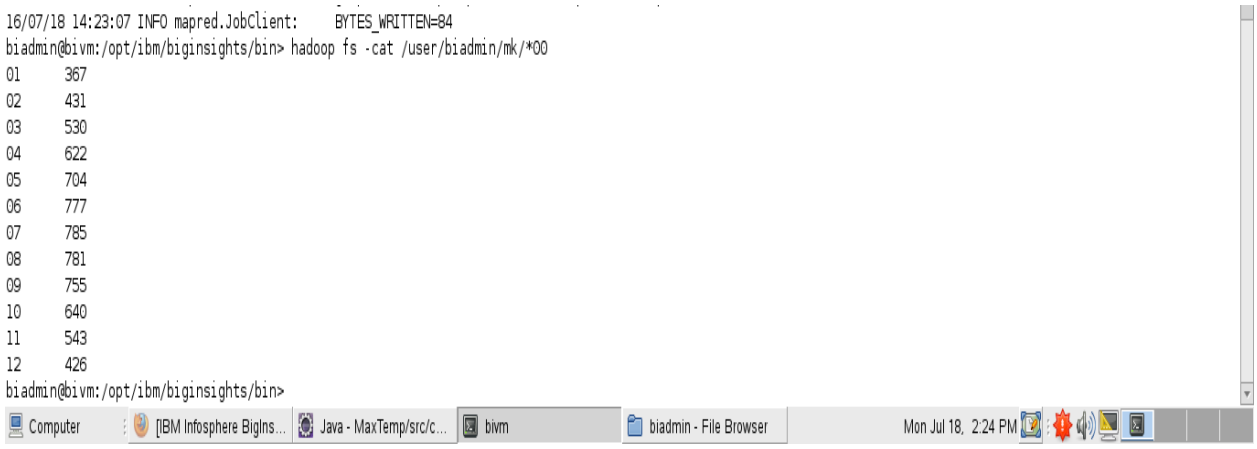
```
cd ~
```

__ 2. Execute your program. At the command line type:

```
hadoop jar MyMaxTemp.jar com.some.company.MaxMonthTemp
/user/biadmin/TempData/SumnerCountyTemp.dat /user/biadmin/TempDataOut
```

__ 3. Close the open edit windows in Eclipse**.**

### 1.10 Output




# ANALYSIS 2

AIM :- Develops a MapReduce application that finds out how many persons make recharge of Rs.400 or more than Rs.400 to their number using the Java perspective in Eclipse.

This analysis perform using the InfoSphere BigInsights 2.1 Quick Start Edition, but has been tested on the 3.0 image.

**2.1 Start the BigInsights components**

__ 1. Log into your BigInsights image with a userid of **biadmin** and a password of **biadmin.**

__ 2. Start your BigInsights components. Use the icon on the desktop.

__ 3. From a command line (right-click the desktop and select **Open in Terminal**) execute:

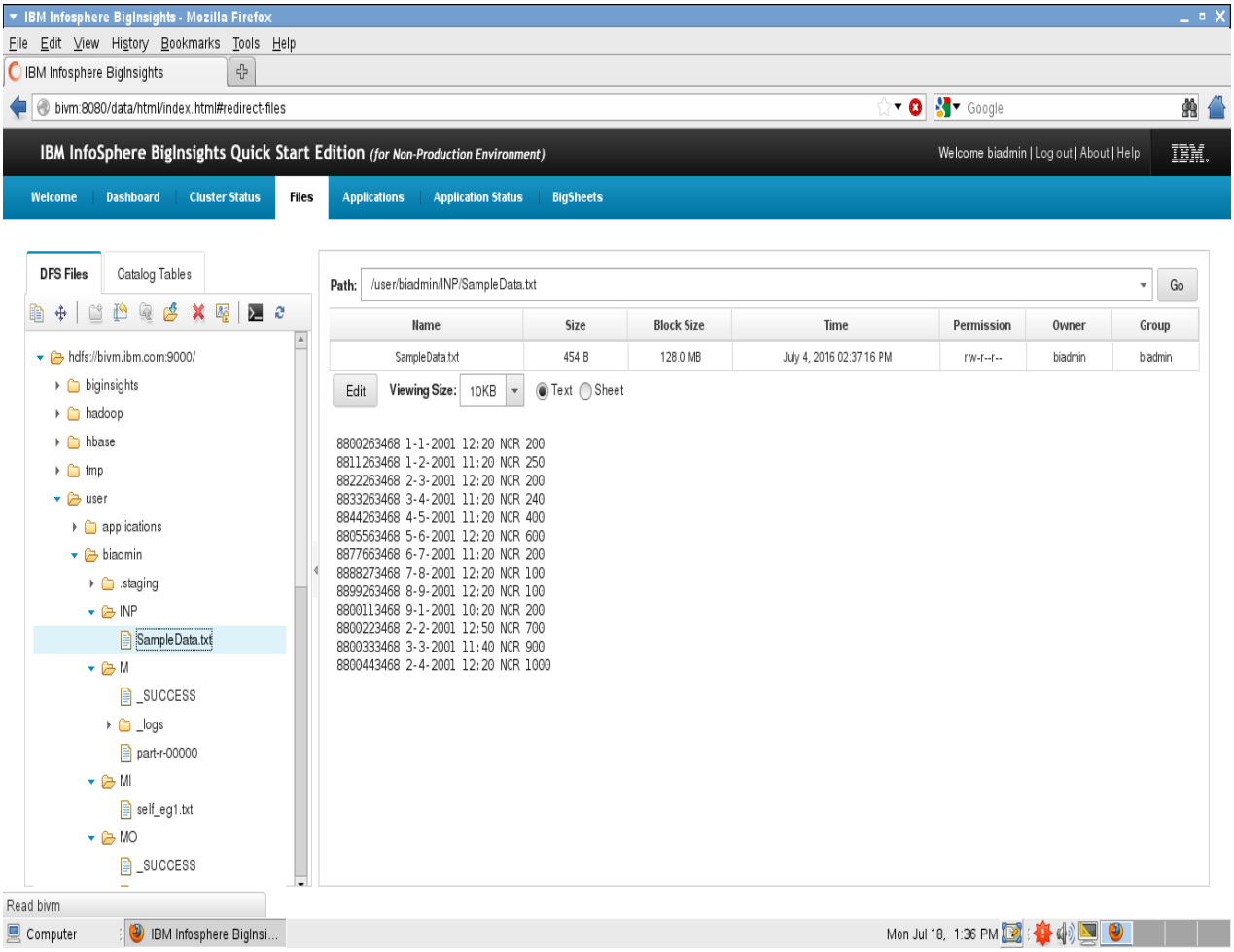
hadoop fs -mkdir mob

__ 4. Upload some temperature data from the local file system.

hadoop fs -copyFromLocal /home/labfiles/SampleData.txt /user/biadmin/mob

__ 5. You can view this data from the *Files* tab in the Web Console or by executing the following command.

hadoop fs -cat mob/ SampleData.txt




**2.2 Define a Java project in Eclipse**

__ 1. Start Eclipse using the icon on the desktop. Go with the default workspace.

__ 2. Make sure that you are using the Java perspective. Click **Window->Open Perspective->Other.** Then select **Java.** Click **OK.**

__ 3. Create a Java project. Select **File->New->Java Project.**

__ 4. Specify a *Project name* of **mobile** Click **Finish.**

__ 5. Right-click the *mobile* project, scroll down and select **Properties.**

__ 6. Select **Java Build Path.**

__ 7. In the *Properties* dialog, select the **Libraries** tab.

__ 8. Click the **Add Library** pushbutton.

__ 9. Select *BigInsights Libraries* and click **Next.** Then click **Finish.** Then click **OK.**

**2.3 Create a Java package and the program**

__ 1. In the *Package Explorer* expand *Mobile* and right-click **src.** Select **New->Package.**

__ 2. Type a *Name* of **hadoop** Click **Finish.**
__ 3. Right-click *hadoop* and select **New->Class.**
__ 4.Now creating MAPREDUCE program using class within class concept of java perspective

```
package hadoop;
import java.util.*;
import java.io.IOException;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapred.*;
public class  Mobile
{
public static class E_EMapper extends MapReduceBase implements
Mapper<LongWritable,Text,Text,IntWritable>
{
  public void map(LongWritable key, Text value,
OutputCollector<Text,IntWritable> output,Reporter reporter) throws IOException

  {
                String line = value.toString();
                String lasttoken = null;
                StringTokenizer s = new StringTokenizer(line," ");
                String mob = s.nextToken();
                while(s.hasMoreTokens())
                {
                        lasttoken=s.nextToken();
                }
                int amout = Integer.parseInt(lasttoken);
                output.collect(new Text(mob),new IntWritable(amout));
   }
}
public static class E_EReduce extends MapReduceBase implements Reducer<
Text, IntWritable, Text, IntWritable >
{
public void reduce( Text key, Iterator <IntWritable> values,
OutputCollector<Text, IntWritable> output,Reporter reporter) throws
IOException
{
int maxamt=400;
int val=Integer.MIN_VALUE;
```

```
while (values.hasNext())
   {
      if((val=values.next().get())>maxamt)
      {
        output.collect(key, new IntWritable(val));
      }
   }
}
}
 //Main function
public static void main(String args[])throws Exception
{
JobConf conf = new JobConf(WordCount.class);
conf.setJobName("max_eletricityunits");
conf.setOutputKeyClass(Text.class);
conf.setOutputValueClass(IntWritable.class);
conf.setMapperClass(E_EMapper.class);
conf.setCombinerClass(E_EReduce.class);
conf.setReducerClass(E_EReduce.class);
conf.setInputFormat(TextInputFormat.class);
conf.setOutputFormat(TextOutputFormat.class);
FileInputFormat.setInputPaths(conf, new Path(args[0]));
FileOutputFormat.setOutputPath(conf, new Path(args[1]));
JobClient.runJob(conf);

}
}
```

__ 3. Save your work.

**2.4 Create a JAR file**

__ 1. In the *Package Explorer,* expand the *Mobile* project. Right-click *src* and select **Export.**

__ 2. Expand *Java* and select *JAR file.* Click **Next.**

__ 3. Click the *JAR file* **browse** pushbutton. Type in a name of **mobile.jar**. Keep *biadmin*
as the folder. Click **OK.**

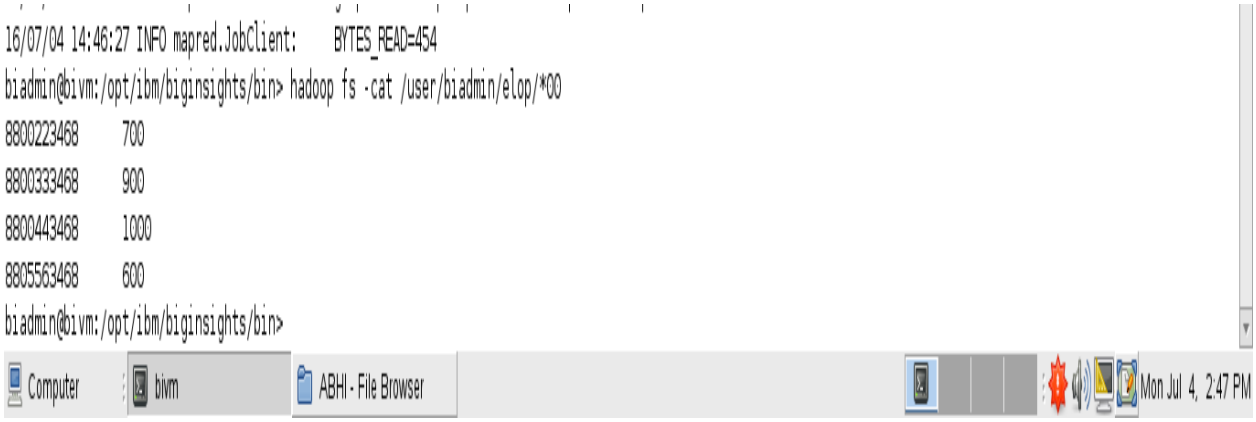
__ 4. Click **Finish.**

Then run your application.

**2.5 Run your application**

__ 1. You need a command line. You may have to open a new one. Change to *biadmin's* home directory.
cd ~
__ 2. Execute your program. At the command line type:
hadoop jar mobile.jar hadoop.Mobile
/user/biadmin/mob/SampleData.txt  /user/biadmin/mobOut
__ 3. Close the open edit windows in Eclipse**.**

### 2.6 Output



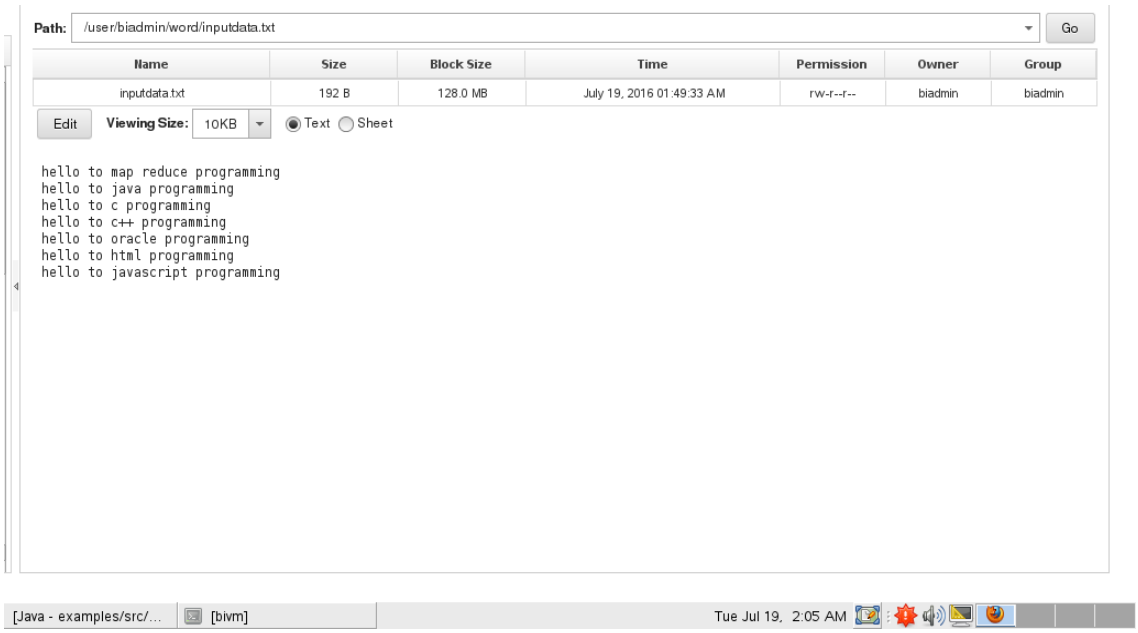

## ANALYSIS 3

AIM :- Develops a MapReduce application that counts the number of words present in a text file using the Java perspective in Eclipse.

This analysis perform using the InfoSphere BigInsights 2.1 Quick Start Edition, but has been tested on the 3.0 image.

### 3.1 Start the BigInsights components

__ 1. Log into your BigInsights image with a userid of **biadmin** and a password of **biadmin.**
__ 2. Start your BigInsights components. Use the icon on the desktop.
__ 3. From a command line (right-click the desktop and select **Open in Terminal**) execute:
hadoop fs -mkdir word
__ 4. Upload some temperature data from the local file system.
hadoop fs -copyFromLocal /home/labfiles/inputData.txt /user/biadmin/word
__ 5. You can view this data from the *Files* tab in the Web Console or by executing the following command.
hadoop fs -cat word/ inputData.txt




**3.2 Define a Java project in Eclipse**

__ 1. Start Eclipse using the icon on the desktop. Go with the default workspace.

__ 2. Make sure that you are using the Java perspective. Click **Window->Open Perspective->Other.** Then select **Java.** Click **OK.**

__ 3. Create a Java project. Select **File->New->Java Project.**

__ 4. Specify a *Project name* of **mobile** Click **Finish.**

__ 5. Right-click the *wordcount* project, scroll down and select **Properties.**

__ 6. Select **Java Build Path.**

__ 7. In the *Properties* dialog, select the **Libraries** tab.

__ 8. Click the **Add Library** pushbutton.

__ 9. Select *BigInsights Libraries* and click **Next.** Then click **Finish.** Then click **OK.**

**3.3 Create a Java package and the program**

__ 1. In the *Package Explorer* expand *wordcount* and right-click **src.** Select **New->Package.**

__ 2. Type a *Name* of **org.apache.hadoop.examples** Click **Finish.**

__ 3. Right-click and select **New->Class.**

__ 4.Now creating MAPREDUCE program using class within class concept of java perspective

```
package org.apache.hadoop.examples;
import java.io.IOException;
import java.util.StringTokenizer;
```

```
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.util.GenericOptionsParser;
public class WordCount {
  public static class TokenizerMapper  extends Mapper<Object, Text, Text,
IntWritable>{
    private final static IntWritable one = new IntWritable(1);
    private Text word = new Text();
    public void map(Object key, Text value, Context context ) throws IOException,
InterruptedException {
      StringTokenizer itr = new StringTokenizer(value.toString());
      while (itr.hasMoreTokens()) {
        word.set(itr.nextToken());
        context.write(word, one);
      }
    }
  }
  public static class IntSumReducer
       extends Reducer<Text,IntWritable,Text,IntWritable> {
    private IntWritable result = new IntWritable();
    public void reduce(Text key, Iterable<IntWritable> values, Context context)
throws IOException, InterruptedException {
      int sum = 0;
      for (IntWritable val : values) {
        sum += val.get();
      }
      result.set(sum);
      context.write(key, result);
    }
  }
  public static void main(String[] args) throws Exception
  {
    Configuration conf = new Configuration();
```

```
    String[] otherArgs = new GenericOptionsParser(conf,
args).getRemainingArgs();
    if (otherArgs.length != 2) {
      System.err.println("Usage: wordcount <in> <out>");
      System.exit(2);
    }
    Job job = new Job(conf, "word count");
    job.setJarByClass(WordCount.class);
    job.setMapperClass(TokenizerMapper.class);
    job.setCombinerClass(IntSumReducer.class);
    job.setReducerClass(IntSumReducer.class);
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(IntWritable.class);
    FileInputFormat.addInputPath(job, new Path(otherArgs[0]));
    FileOutputFormat.setOutputPath(job, new Path(otherArgs[1]));
    System.exit(job.waitForCompletion(true) ? 0 : 1);
  }
}
```

__ 3. Save your work.

## 3.4 Create a JAR file

__ 1. In the *Package Explorer,* expand the *wordcount*project. Right-click *src* and select Export.

__ 2. Expand *Java* and select *JAR file.* Click Next.

__ 3. Click the *JAR file* browse pushbutton. Type in a name of word.jar. Keep *biadmin* as the folder. Click OK.

__ 4. Click Finish.

Then run your application.

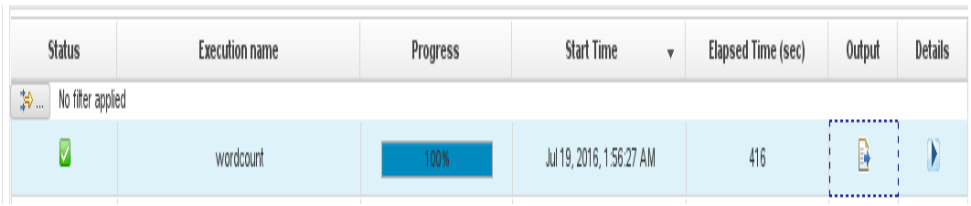
## 3.5 Run your application

__ 1. You need a command line. You may have to open a new one. Change to *biadmin's* home directory.

```
cd ~
```

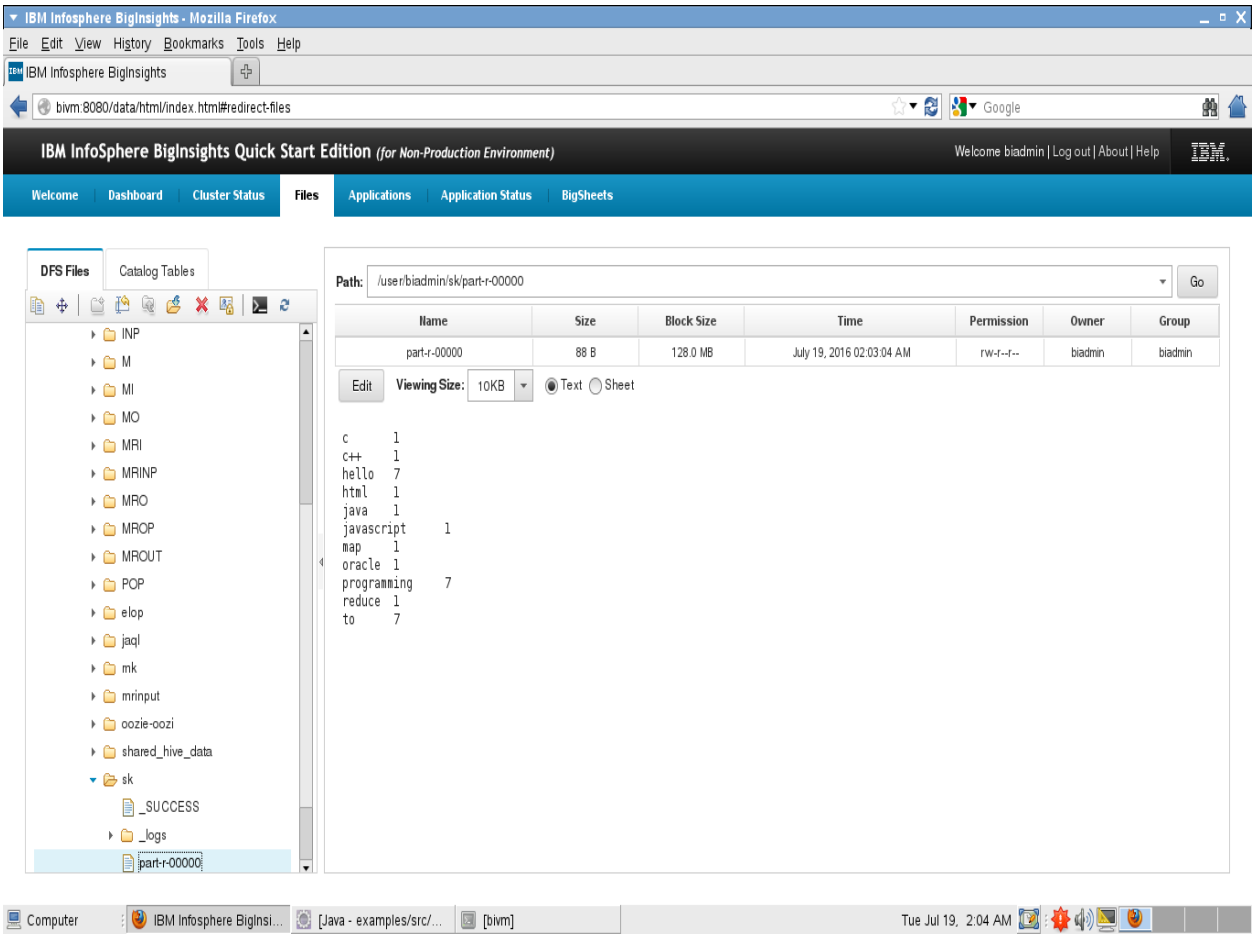
__ 2. Execute your program. At the command line type:

```
hadoop jar mobile.jar hadoop.Mobile
/user/biadmin/word/Data.txt  /user/biadmin/wordOut
```




__ 3. Close the open edit windows in Eclipse**.**

## 3.6 Output




# ANALYSIS 4

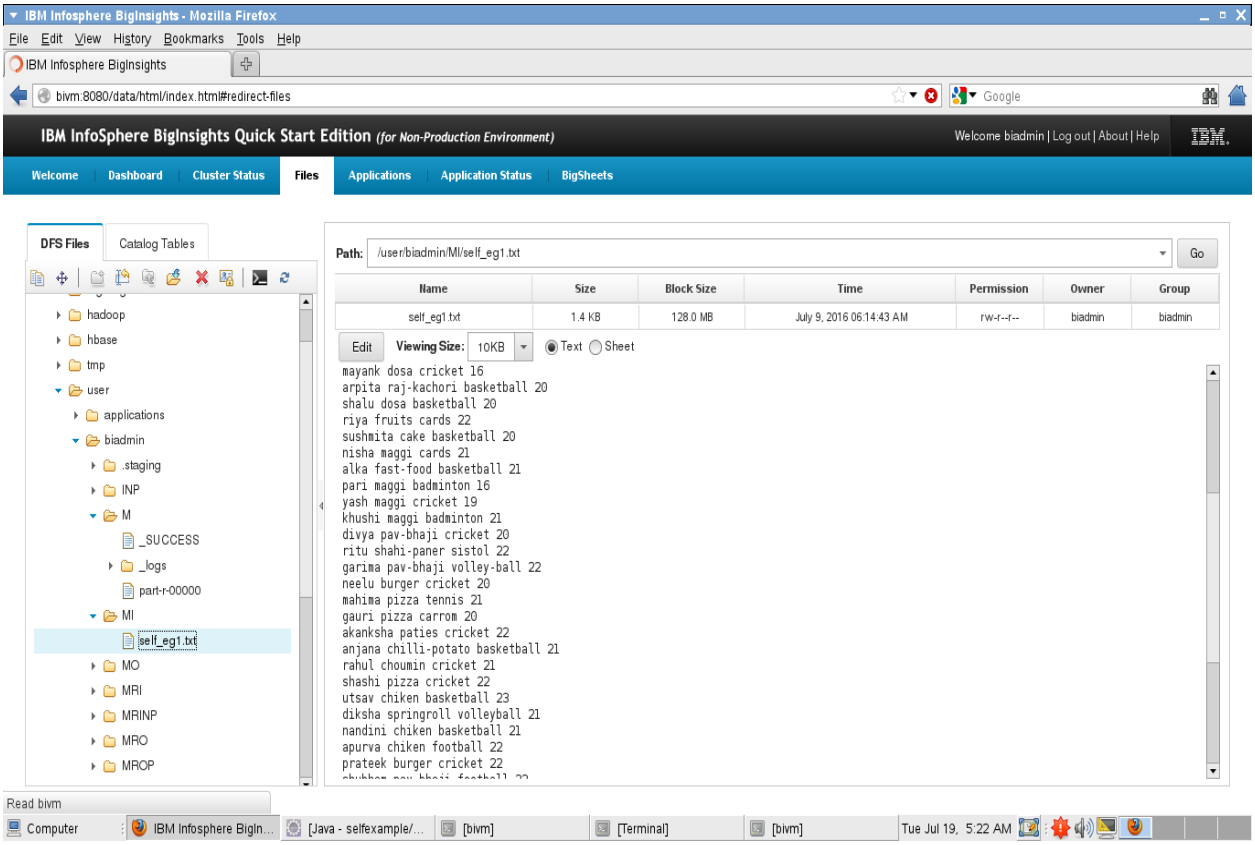
**AIM:-** Develops a MapReduce application that finds out the name of students having age more than 20 years using the Java perspective in Eclipse.

## 4.1 Start the BigInsights components

__ 1. Log into your BigInsights image with a userid of biadmin and a password of biadmin.

__ 2. Start your BigInsights components. Use the icon on the desktop.

__ 3. From a command line (right-click the desktop and select Open in Terminal) execute:
hadoop fs -mkdir TempData
__ 4. Upload some temperature data from the local file system.
hadoop fs -copyFromLocal /home/labfiles/ self_eg1.txt /user/biadmin/stu
__ 5. You can view this data from the *Files* tab in the Web Console or by executing the
following command.
cat TempData/ self_eg1.txt




## 4.2 Define a Java project in Eclipse

__ 1. Start Eclipse using the icon on the desktop. Go with the default workspace.
__ 2. Make sure that you are using the Java perspective. Click **Window->Open Perspective->Other.** Then select **Java.** Click **OK.**
__ 3. Create a Java project. Select **File->New->Java Project.**
__ 4. Specify a *Project name* of **example.** Click **Finish.**
__ 5. Right-click the **example** project, scroll down and select **Properties.**
__ 6. Select **Java Build Path.**
__ 7. In the *Properties* dialog, select the **Libraries** tab.
__ 8. Click the **Add Library** pushbutton.

__ 9. Select *BigInsights Libraries* and click **Next.** Then click **Finish.** Then click **OK.**

## 4.3 Create a Java package and the mapper class

__ 1. In the *Package Explorer* expand *MaxTemp* and right-click **src.** Select **New->Package.**
__ 2. Type a *Name* of **com.sona.example.** Click **Finish.**
__ 3. Right-click *com.sona.example* and select **New->Class.**
__ 4. Type in a *Name* of **mappp.** It will be a *public* class. Click **Finish.**
The data type for the input key to the mapper will be *LongWritable.* The data itself will be of type *Text.* The output key from the mapper will be of type *Text.* And the data from the mapper (the temperature) will be of type *IntWritable.*
__ 5. Your class:
__ a. You will need to import *java.io.IOException.*
__ b. Exend Mapper<LongWritable, Text, Text, IntWritable>
__ c. Define a public class called **map.**
__ d. Your code should look like the following:

```
package com.sona.example;
import java.util.*;
import java.io.IOException;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapred.*;

public class mapppp extends MapReduceBase implements
Mapper<LongWritable, Text, Text, IntWritable> {
        public void map(LongWritable key, Text value, OutputCollector<Text,IntWritable>
output,Reporter reporter) throws IOException  {
                String line = value.toString();
                String lasttoken = null;
                StringTokenizer s = new StringTokenizer(line," ");
                String name = s.nextToken();
                while(s.hasMoreTokens())
                {
                        lasttoken=s.nextToken();
                }
                int age= Integer.parseInt(lasttoken);
                output.collect(new Text(name),new IntWritable(age));
        }
}
```

## 4.4 Create the reducer class

__ 1. In the *Package Explorer* right-click package com.sona.example and select **New->Class.**
__ 2. Type in a *Name* of **rreducce.** It will be a  class. Click **Finish.**
The data type for the input key to the reducer will be *Text.* The data itself will be of type *IntWritable.* The output key from the reducer will be of type *Text.* And the data from the reducer will be of type *IntWritable.*
__ 3. Your class:
__ a. You will need to import *java.io.IOException.*
__ b. Extend Reducer<Text, LongWritable, Text, IntWritable>
__ c. Define a public class called **reduce.**

__ d. Your code should look like the following:

```
package com.sona.example;
import java.util.*;
import java.io.IOException;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapred.*;

public class rrredduce extends MapReduceBase implements Reducer< Text, IntWritable, Text,
IntWritable >
{public void reduce( Text key, Iterator <IntWritable> values, OutputCollector<Text,
IntWritable> output,Reporter reporter) throws IOException {
        int minage=20;
        int val=Integer.MIN_VALUE;
        while (values.hasNext())
           {
              if((val=values.next().get())>minage)
              {
                output.collect(key, new IntWritable(val));
              }
           }
}
}
```

## 4.5 Create the mainclass

__ 1. In the *Package Explorer* right-click *com.sona.example* and select **New->Class.**

__ 2. Type in a *Name* of **mmainn.** It will be a *public* class. Click **Finish.**

The *GenericOptionsParser()* will extract any input parameters that are not system parameters and place them in an array. In your case, two parameters will be passed to your application. The first parameter is the input file. The second parameter is the output directory. (This directory must not exist or your MapReduce application will fail.)

Your code should look like this:

```
package com.sona.example;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapred.*;

public class mmainn {
        public static void main(String[] args) throws Exception{
                JobConf conf = new JobConf(mmainn.class);
                conf.setJobName("max_eletricityunits");
                conf.setOutputKeyClass(Text.class);
                conf.setOutputValueClass(IntWritable.class);
                conf.setMapperClass(mapppp.class);
                conf.setCombinerClass(rrredduce.class);
                conf.setReducerClass(rrredduce.class);
                conf.setInputFormat(TextInputFormat.class);
                conf.setOutputFormat(TextOutputFormat.class);
```

```
            FileInputFormat.setInputPaths(conf, new Path(args[0]));
            FileOutputFormat.setOutputPath(conf, new Path(args[1]));
            JobClient.runJob(conf);
        }
}
```

3. Save your work.

## 4.6 Create a JAR file

__ 1. In the *Package Explorer,* expand the *MaxTemp* project. Right-click *src* and select **Export.**

__ 2. Expand *Java* and select *JAR file.* Click **Next.**

__ 3. Click the *JAR file* **browse** pushbutton. Type in a name **stu.jar.** Keep *biadmin* as the folder. Click **OK.**

__ 4. Click **Finish.**

To save time, you are not going to run your application now. You will add a combiner class and then run the application.

## 4.7 Run your application

__ 1. You need a command line. You may have to open a new one. Change to *biadmin's* home directory.

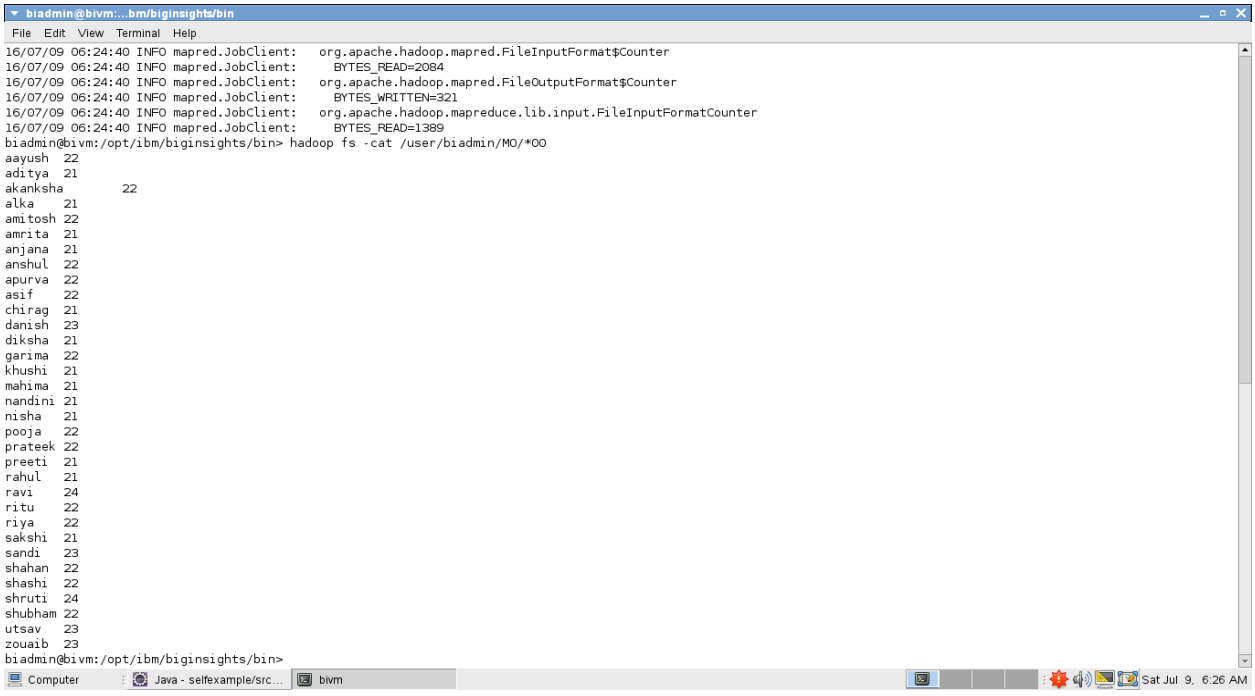
cd ~

__ 2. Execute your program. At the command line type:

hadoop jar MyMaxTemp.jar com.some.company.MaxMonthTemp /user/biadmin/Mrinp/self_eg1.txt /user/biadmin/stuout

__ 3. Close the open edit windows in Eclipse**.**

## 4.8Output




# JAQL

Jaql is primarily a query language for JavaScript Object Notation (JSON), but it supports more than just JSON. It allows you to process both structured and nontraditional data and was donated by IBM to the open source community (just one of many contributions IBM has made to open source).

Specifically, Jaql allows you to select, join, group, and filter data that is stored in HDFS, much like a blend of Pigand Hive. Jaql's query language was inspired by many programming and query languages, including Lisp, SQL, XQuery, and Pig. Jaql is a functional, declarative query language that is designed to process large data sets. For parallelism, Jaql rewrites high-level queries, when appropriate, into "low-level" queries consisting of MapReducejobs.

## JSON

Before we get into the Jaql language, let's first look at the popular data interchange format known as JSON, so that we can build our Jaql examples on top of it. Application developers are moving in large numbers towards JSON as their choice for a data interchange format, because it's easy for humans to read, and because of its structure, it's easy for applications to parse or generate.

JSON is built on top of two types of structures. The first is a collection of name/value pairs (which, as you learned earlier in the "The Basics of MapReduce" section, makes it ideal for data manipulation in Hadoop, which works on key/value pairs). These name/value pairs can represent anything since they are simply text strings (and subsequently fit well into existing models) that could represent a record in a database, an object, an associative array, and more. The second JSON structure is the ability to create an ordered list of values much like an array, list, or sequence you might have in your existing applications. An object in JSON is represented as {string:value}, where an array can be simply represented by [ value, value, … ], where value can be a string, number, another JSON object, or another JSON array.

Both Jaql and JSON are record-oriented models, and thus fit together perfectly. Note that JSON is not the only format that Jaql supports—in fact, Jaql is extremely flexible and can support many semi-structured data sources such as XML, CSV, flat files, and more. However, in consideration of the space we have, we'll use the JSON example above in the following Jaql

queries. As you will see from this section, Jaql looks very similar to Pig but also has some similarity to SQL.

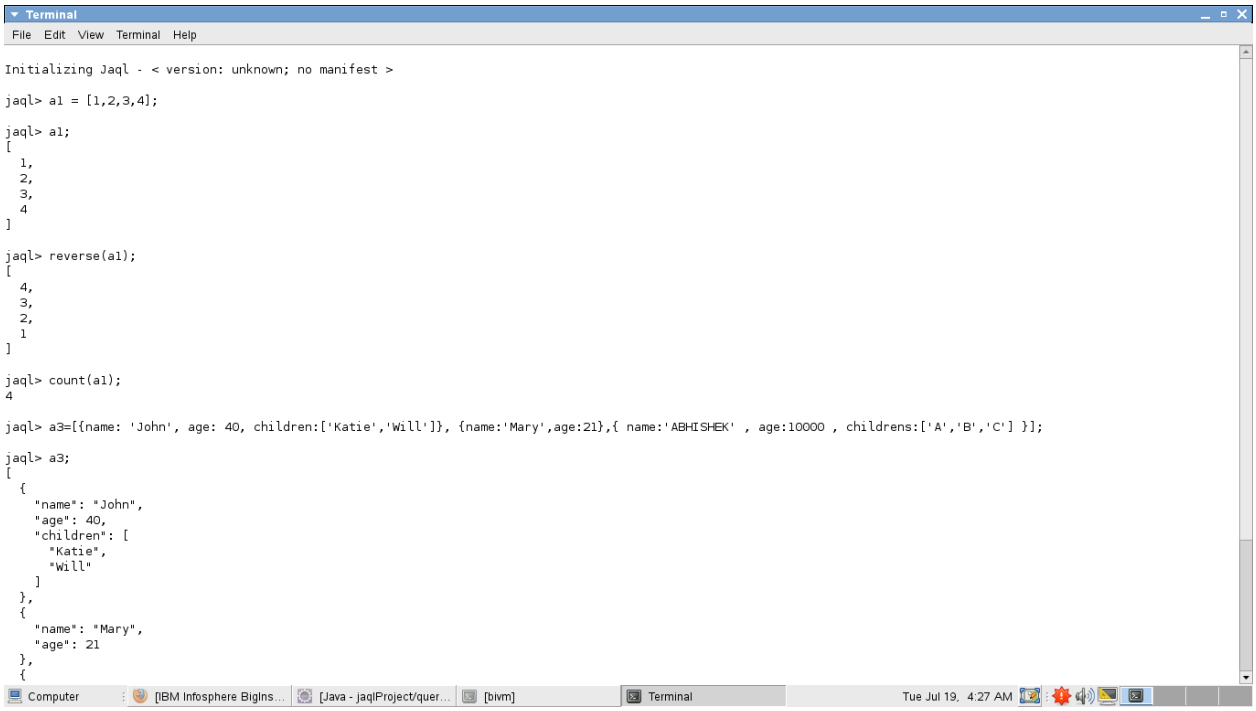
# JAQL Query

Much like a MapReduce job is a flow of data, Jaql can be thought of as a pipeline of data flowing from a source, through a set of various operators, and out into a sink (a destination). The operand used to signify flow from one operand to another is an arrow: ->. Unlike SQL, where the output comes first (for example, the SELECT list), in Jaql, the operations listed are in natural order, where you specify the source, followed by the various operators you want to use to manipulate the data, and finally the sink.

## Basic Commands

```
jaql>a1 = [1,2,3,4];

    jaql>a1[3];

    jaql>a1[0:2];

    jaql>a1[0]==9;

    jaql>a1 = replaceElement(a1,0,9);

    jaql>a1;

    jaql>a1 = range(1,20);   // craete a new array from 1 to 20

    jaql>a1;

    jaql>reverse(a1);

    jaql>count(a1);
```

# Output




# JSON Commands

jaql>a3=[{name: 'John', age: 40, children:['Katie','Will']},
{name:'Mary',age:21},{ name:'ABHISHEK' , age:10000 ,
childrens:['A','B','C'] }];

a3=[

{

name: 'John',

age: 40,

children:['Katie','Will']

},

{

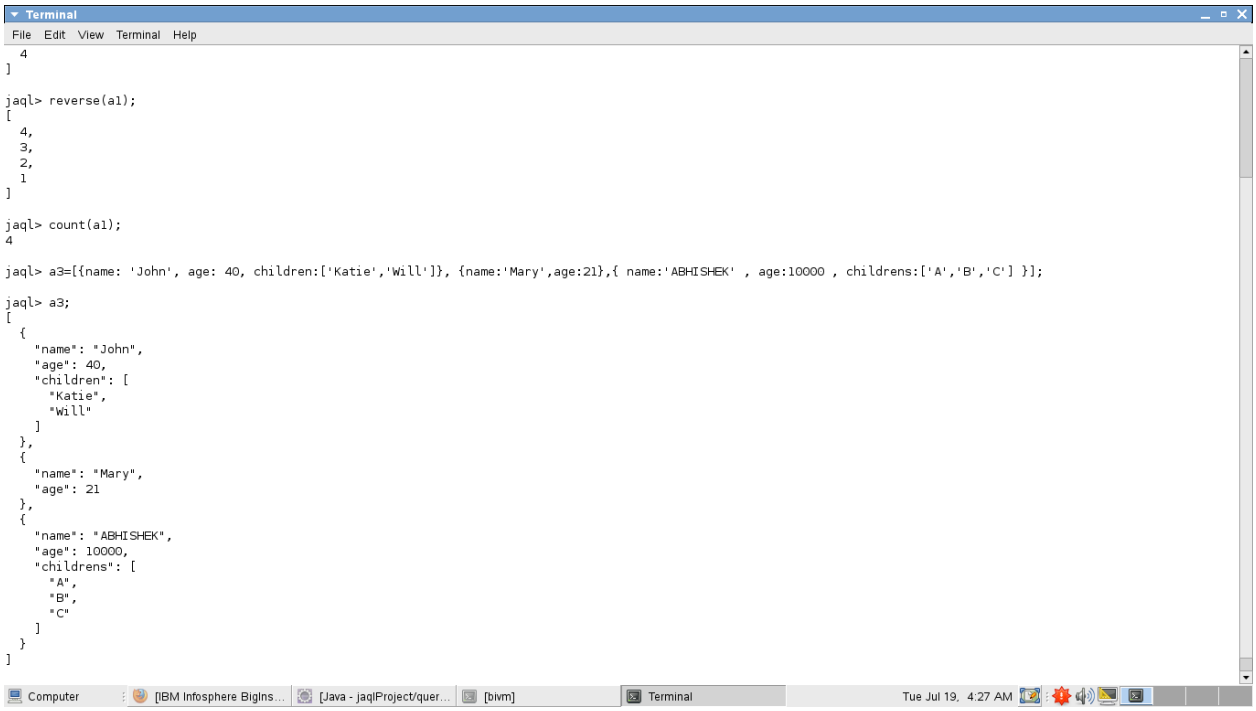
name:'Mary',

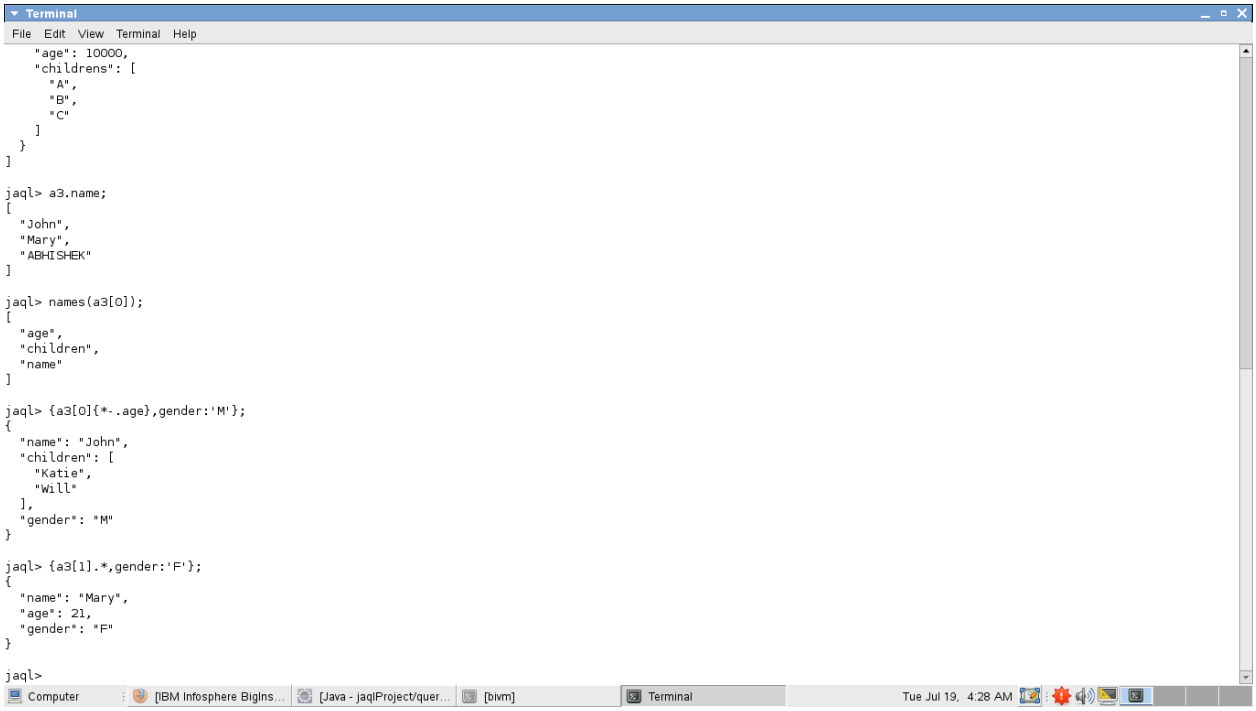
```
                age:21
            }
        ];
```

Access Commands

```
jaql>a3.name;                   //all name
jaql>a3[*].name;            //all name
jaql>a3[0].name             //name field for the first record
jaql>a3[0].children[0];     //record-1 childeren-1
jaql>a3[0]{.name,.age};         // first record name and age
jaql>a3;
jaql>{a3[1].*,gender:'F'};      //temp add a new attribute
jaql>a3[0]{*-.age};             //temp remove an attribute
jaql>{a3[0]{*-.age},gender:'M'}; // add and remove at same time.
jaql>names(a3[0]);              //List the field names in the first record;
```

OUTPUT




# Working with Eclipse

Eclipse is used to perform batch processing that means we can perform multiple queries on a single run.

For Example: Twitter Analysis

jaql>tweets
=read(file("file:///home/biadmin/labfiles/SampleData/Twitter%20Search.json"));

tweets;

Retrieve a single field, from_user, using the transform command.

jaql>tweets -> transform $.from_user;

jaql>tweets->transform {$.created_at, $.from_user, $.iso_language_code, $.text};

Create a new record, tweetsrecs and at the same time change the name of one of the fields

jaql>tweetrecs = tweets->transform {$.created_at, $.from_user, language:$.iso_language_code, $.text};

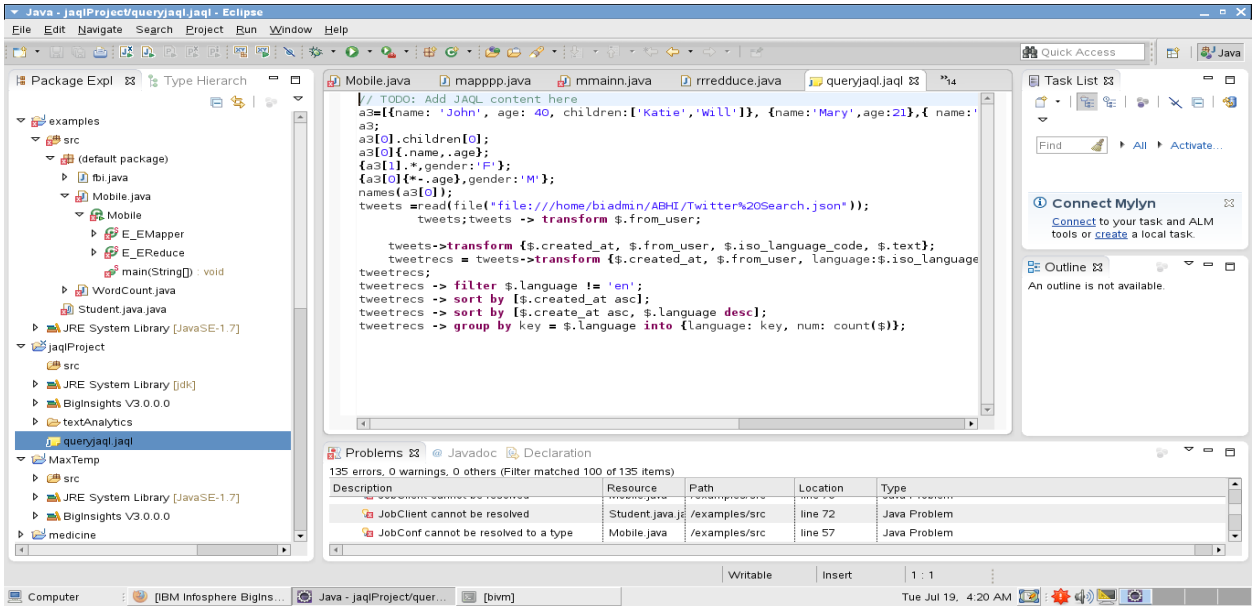
jaql>tweetrecs

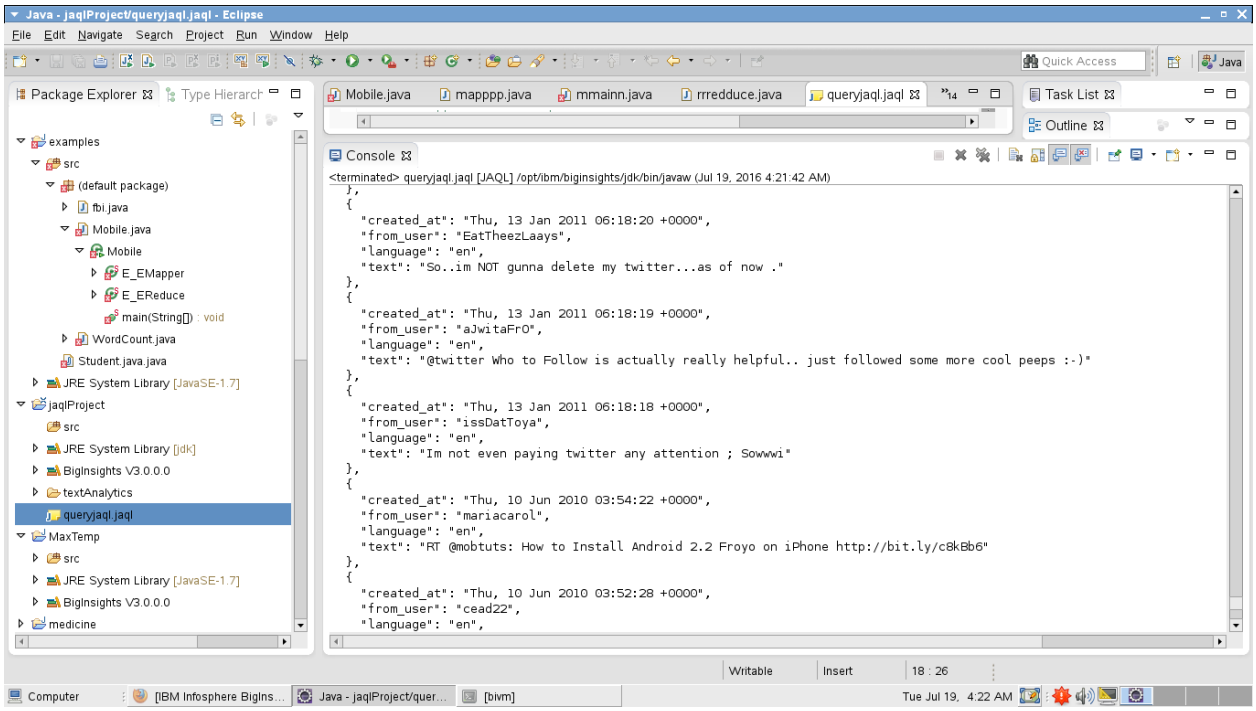
Using filters:

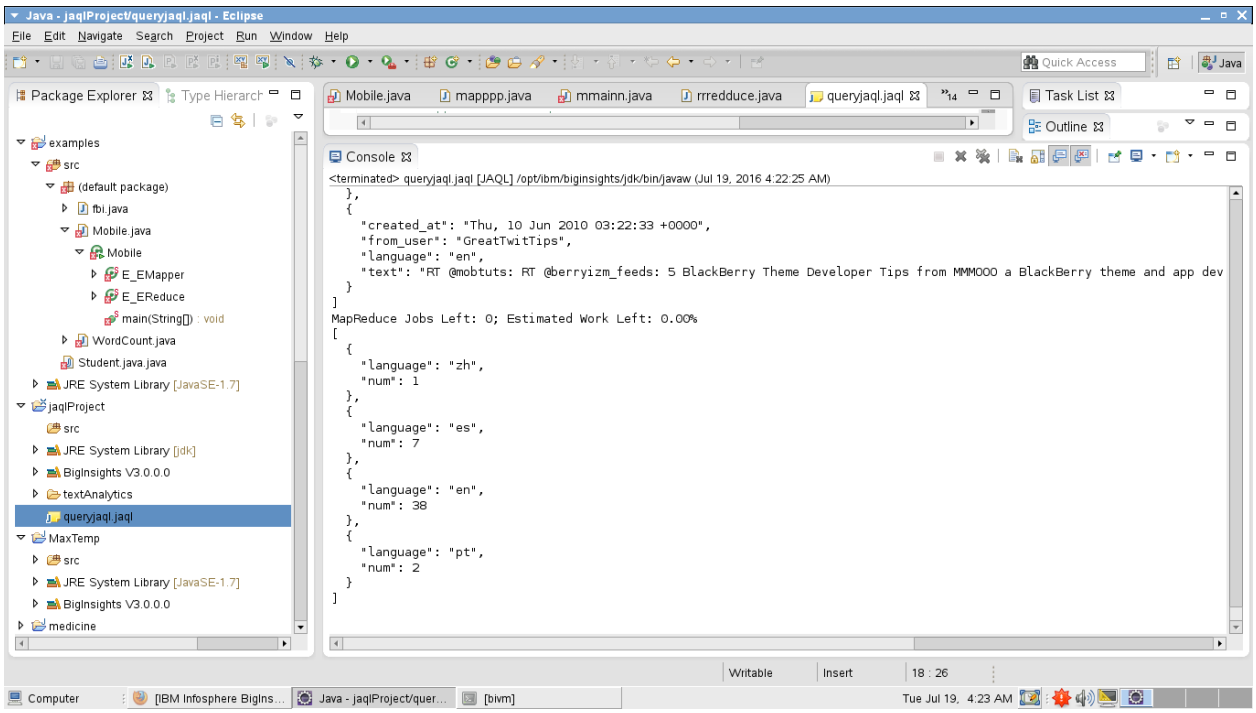
Use the filter operator to see all non-English language records from tweetrecs

jaql>tweetrecs -> filter $.language != 'en';

# Output:




Java - jaqlProject/queryjaql.jaql - Eclipse
File Edit Navigate Search Project Run Window Help
Package Explorer
Type Hierarch
Mobile.java
mapppp.java
mmainn.java
rrredduce.java
queryjaql.jaql
Task List
Outline
Console
<terminated> queryjaql.jaql [JAQL] /opt/ibm/biginsights/jdk/bin/javaw (Jul 19, 2016 4:21:42 AM)
examples
src
(default package)
fbi.java
Mobile.java
Mobile
E_EMapper
E_EReduce
main(String[]) : void
WordCount.java
Student.java.java
JRE System Library [JavaSE-1.7]
jaqlProject
src
JRE System Library [jdk]
BigInsights V3.0.0.0
textAnalytics
queryjaql.jaql
MaxTemp
src
JRE System Library [JavaSE-1.7]
BigInsights V3.0.0.0
medicine
{
"created_at": "Thu, 13 Jan 2011 06:18:20 +0000",
"from_user": "EatTheezLaays",
"language": "en",
"text": "So..im NOT gunna delete my twitter...as of now ."
},
{
"created_at": "Thu, 13 Jan 2011 06:18:19 +0000",
"from_user": "aJwitaFrO",
"language": "en",
"text": "@twitter Who to Follow is actually really helpful.. just followed some more cool peeps :-)"
},
{
"created_at": "Thu, 13 Jan 2011 06:18:18 +0000",
"from_user": "issDatToya",
"language": "en",
"text": "Im not even paying twitter any attention ; Sowwwi"
},
{
"created_at": "Thu, 10 Jun 2010 03:54:22 +0000",
"from_user": "mariacarol",
"language": "en",
"text": "RT @mobtuts: How to Install Android 2.2 Froyo on iPhone http://bit.ly/c8kBb6"
},
{
"created_at": "Thu, 10 Jun 2010 03:52:28 +0000",
"from_user": "cead22",
"language": "en",
Writable
Insert
18 : 26
Computer
[IBM Infosphere BigIns...
Java - jaqlProject/quer...
[bivm]
Tue Jul 19, 4:22 AM


Java - jaqlProject/queryjaql.jaql - Eclipse
File Edit Navigate Search Project Run Window Help
Package Explorer
Type Hierarch
Mobile.java
mapppp.java
mmainn.java
rrredduce.java
queryjaql.jaql
Task List
Outline
Console
<terminated> queryjaql.jaql [JAQL] /opt/ibm/biginsights/jdk/bin/javaw (Jul 19, 2016 4:22:25 AM)
examples
src
(default package)
fbi.java
Mobile.java
Mobile
E_EMapper
E_EReduce
main(String[]) : void
WordCount.java
Student.java.java
JRE System Library [JavaSE-1.7]
jaqlProject
src
JRE System Library [jdk]
BigInsights V3.0.0.0
textAnalytics
queryjaql.jaql
MaxTemp
src
JRE System Library [JavaSE-1.7]
BigInsights V3.0.0.0
medicine
},
{
"created_at": "Thu, 10 Jun 2010 03:22:33 +0000",
"from_user": "GreatTwitTips",
"language": "en",
"text": "RT @mobtuts: RT @berryizm_feeds: 5 BlackBerry Theme Developer Tips from MMMOOO a BlackBerry theme and app dev
}
]
MapReduce Jobs Left: 0; Estimated Work Left: 0.00%
[
{
"language": "zh",
"num": 1
},
{
"language": "es",
"num": 7
},
{
"language": "en",
"num": 38
},
{
"language": "pt",
"num": 2
}
]
Writable
Insert
18 : 26
Computer
[IBM Infosphere BigIns...
Java - jaqlProject/quer...
[bivm]
Tue Jul 19, 4:23 AM

# HIVE

Hive allows SQL developers to write Hive Query Language (HQL) statements that are similar to standard SQL statements; now you should be aware that HQL is limited in the commands it understands, but it is still pretty useful. HQL statements are broken down by the Hive service into MapReducejobs and executed across a Hadoop cluster.

For anyone with a SQL or relational database background, this section will look very familiar to you. As with any database management system (DBMS), you can run your Hive queries in many ways. You can run them from a command line interface (known as the Hive shell), from a Java Database Connectivity (JDBC) or Open Database Connectivity (ODBC) application leveraging the Hive JDBC/ODBC drivers, or from what is called a Hive Thrift Client. The Hive Thrift Client is much like any database client that gets installed on a user's client machine (or in a middle tier of a three-tier architecture): it communicates with the Hive services running on the server. You can use the Hive Thrift Client within applications written in C++, Java, PHP, Python, or Ruby (much like you can use these client-side languages with embedded SQL to access a database such as DB2 or Informix).

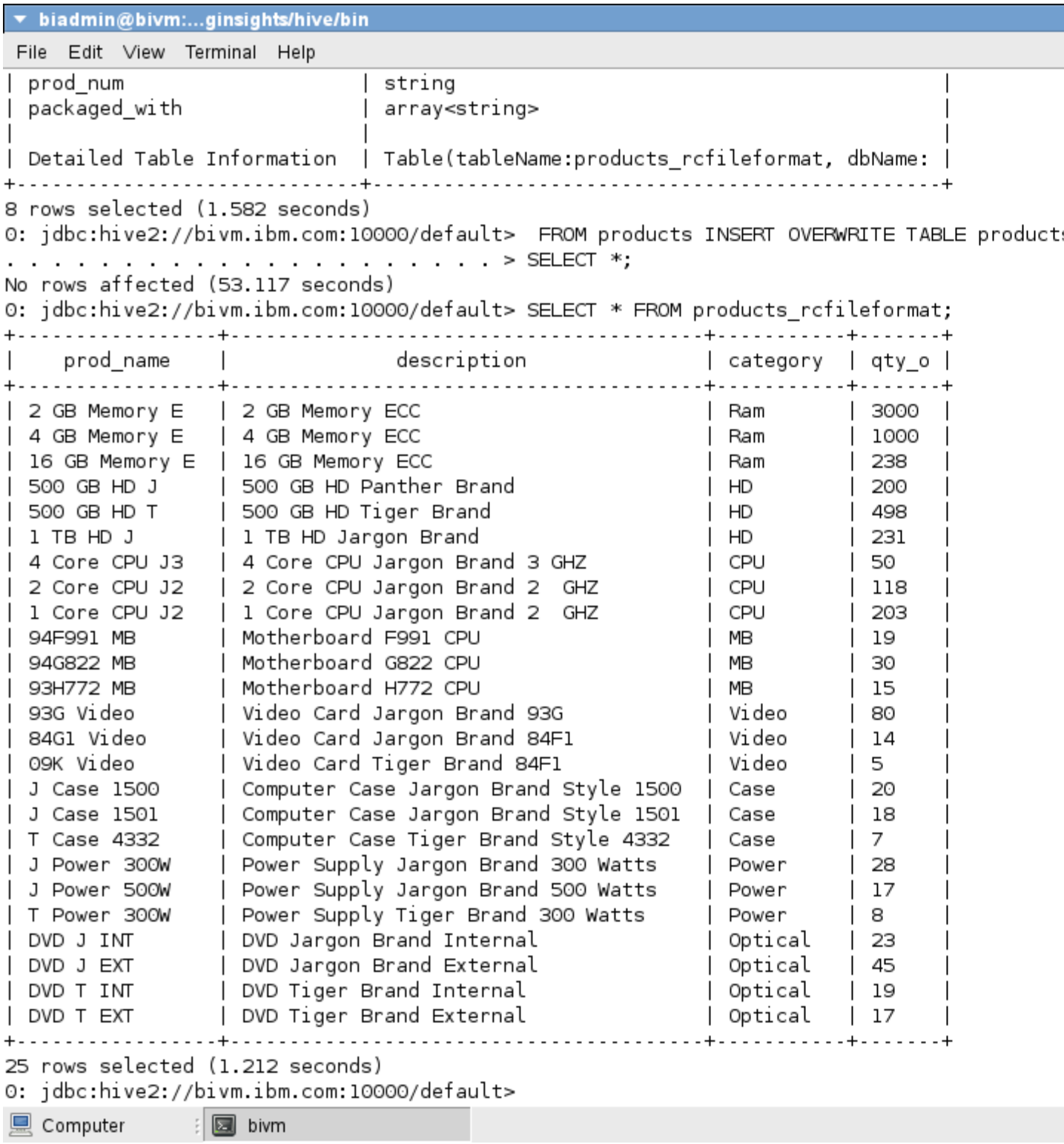
Hive looks very much like traditional database code with SQL access. However, because Hive is based on Hadoop and MapReduce operations, there are several key differences. The first is that Hadoop is intended for long sequential scans, and because Hive is based on Hadoop, you can expect queries to have a very high latency (many minutes). This means that Hive would not be appropriate for applications that need very fast response times, as you would expect with a database such as DB2. Finally, Hive is read-based and therefore not appropriate for transaction processing that typically involves a high percentage of write operations.
If you're interested in SQL on Hadoop, in addition to Hive, IBM offers Big SQL which makes accessing Hive datasets faster and more secure. Checkout our videos, below, for a quick overview of Hive and Big SQL.

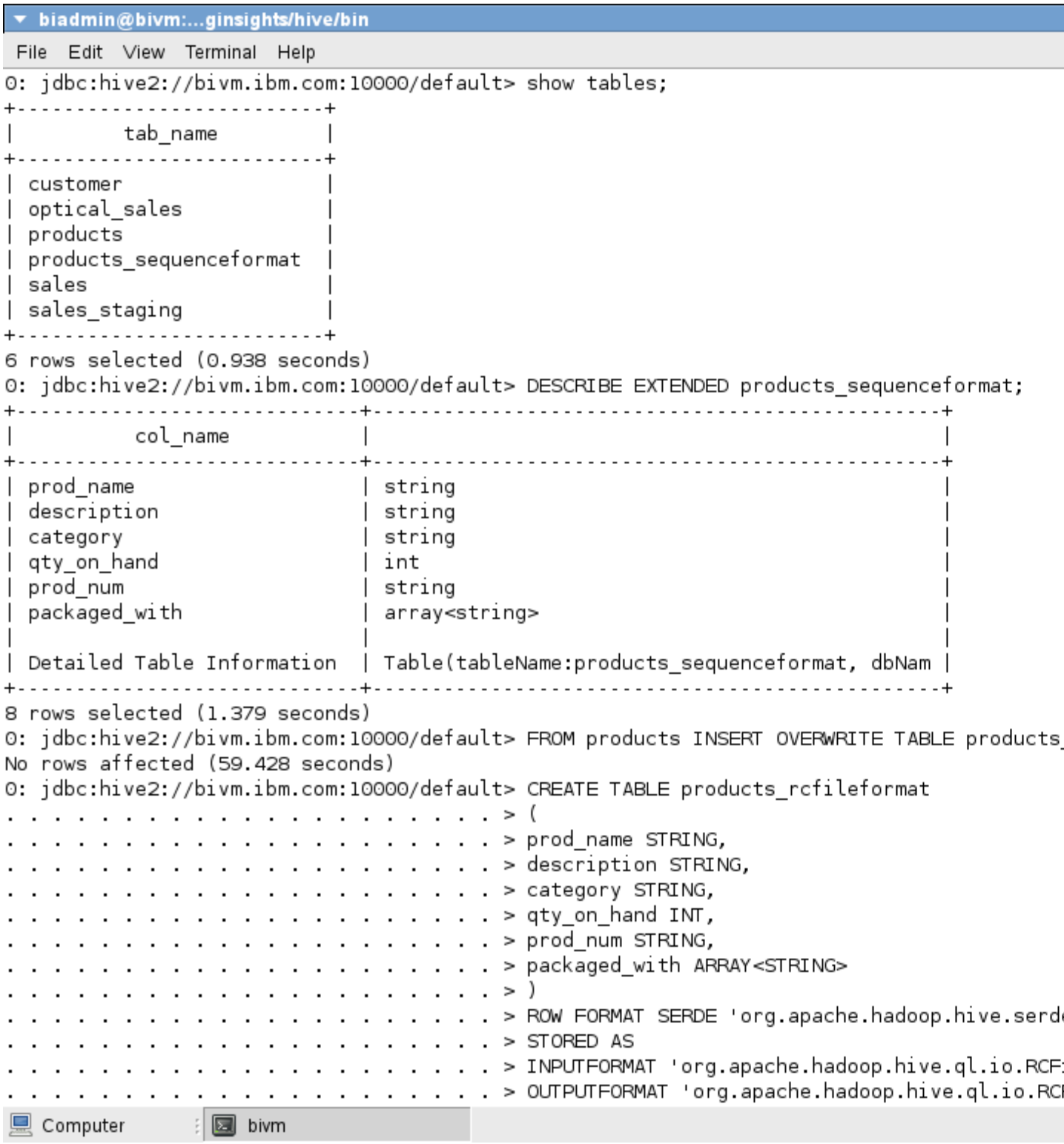
biadmin@bivm:...ginsights/hive/bin

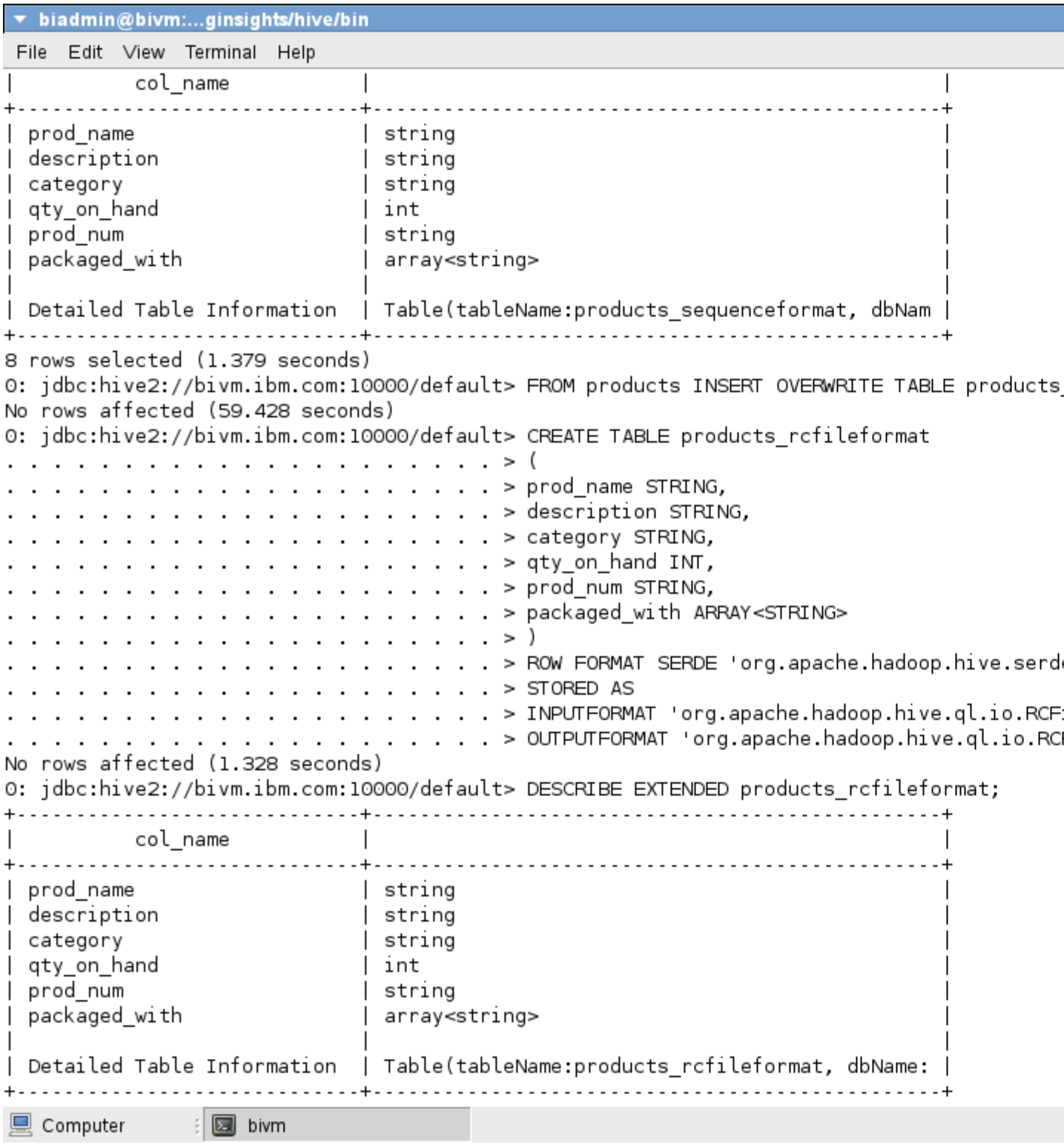
File Edit View Terminal Help

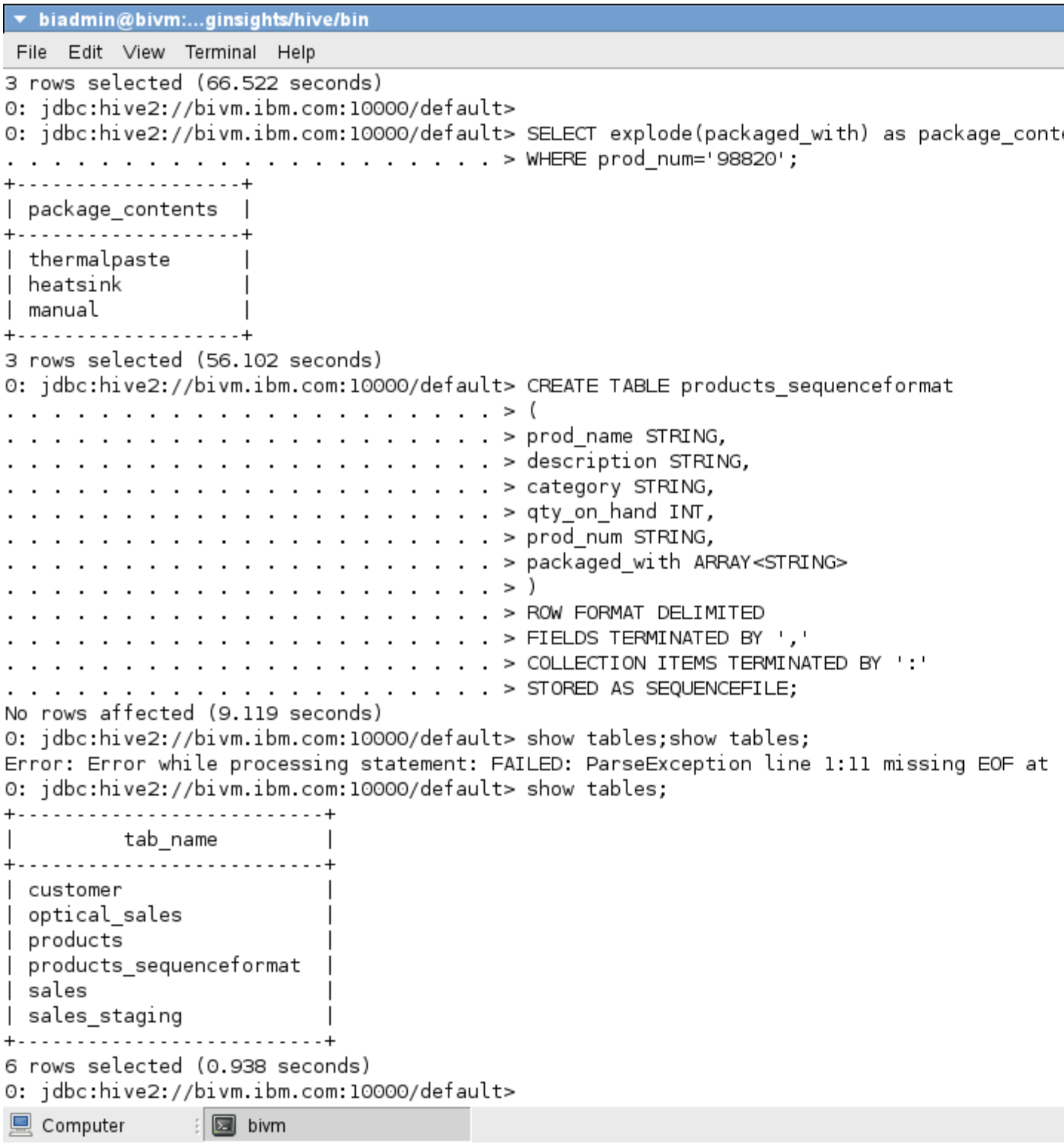
```
. . . . . . . . . . . . . . . . . . . . . . . > qty_on_hand INT,
. . . . . . . . . . . . . . . . . . . . . . . > prod_num STRING,
. . . . . . . . . . . . . . . . . . . . . . . > packaged_with ARRAY<STRING>
. . . . . . . . . . . . . . . . . . . . . . . > )
. . . . . . . . . . . . . . . . . . . . . . . > ROW FORMAT DELIMITED
. . . . . . . . . . . . . . . . . . . . . . . > FIELDS TERMINATED BY ','
. . . . . . . . . . . . . . . . . . . . . . . > COLLECTION ITEMS TERMINATED BY ':'
. . . . . . . . . . . . . . . . . . . . . . . > STORED AS ORC;
No rows affected (1.224 seconds)
0: jdbc:hive2://bivm.ibm.com:10000/default> FROM products INSERT OVERWRITE TABLE products_
No rows affected (57.28 seconds)
0: jdbc:hive2://bivm.ibm.com:10000/default> SELECT * FROM products_orcformat;
+-----------------+-----------------------------------------+-----------+-------+
|    prod_name    |               description               | category  | qty_o |
+-----------------+-----------------------------------------+-----------+-------+
| 2 GB Memory E   | 2 GB Memory ECC                         | Ram       | 3000  |
| 4 GB Memory E   | 4 GB Memory ECC                         | Ram       | 1000  |
| 16 GB Memory E  | 16 GB Memory ECC                        | Ram       | 238   |
| 500 GB HD J     | 500 GB HD Panther Brand                 | HD        | 200   |
| 500 GB HD T     | 500 GB HD Tiger Brand                   | HD        | 498   |
| 1 TB HD J       | 1 TB HD Jargon Brand                    | HD        | 231   |
| 4 Core CPU J3   | 4 Core CPU Jargon Brand 3 GHZ           | CPU       | 50    |
| 2 Core CPU J2   | 2 Core CPU Jargon Brand 2  GHZ          | CPU       | 118   |
| 1 Core CPU J2   | 1 Core CPU Jargon Brand 2  GHZ          | CPU       | 203   |
| 94F991 MB       | Motherboard F991 CPU                    | MB        | 19    |
| 94G822 MB       | Motherboard G822 CPU                    | MB        | 30    |
| 93H772 MB       | Motherboard H772 CPU                    | MB        | 15    |
| 93G Video       | Video Card Jargon Brand 93G             | Video     | 80    |
| 84G1 Video      | Video Card Jargon Brand 84F1            | Video     | 14    |
| 09K Video       | Video Card Tiger Brand 84F1             | Video     | 5     |
| J Case 1500     | Computer Case Jargon Brand Style 1500   | Case      | 20    |
| J Case 1501     | Computer Case Jargon Brand Style 1501   | Case      | 18    |
| T Case 4332     | Computer Case Tiger Brand Style 4332    | Case      | 7     |
| J Power 300W    | Power Supply Jargon Brand 300 Watts     | Power     | 28    |
| J Power 500W    | Power Supply Jargon Brand 500 Watts     | Power     | 17    |
| T Power 300W    | Power Supply Tiger Brand 300 Watts      | Power     | 8     |
| DVD J INT       | DVD Jargon Brand Internal               | Optical   | 23    |
| DVD J EXT       | DVD Jargon Brand External               | Optical   | 45    |
| DVD T INT       | DVD Tiger Brand Internal                | Optical   | 19    |
| DVD T EXT       | DVD Tiger Brand External                | Optical   | 17    |
+-----------------+-----------------------------------------+-----------+-------+
```

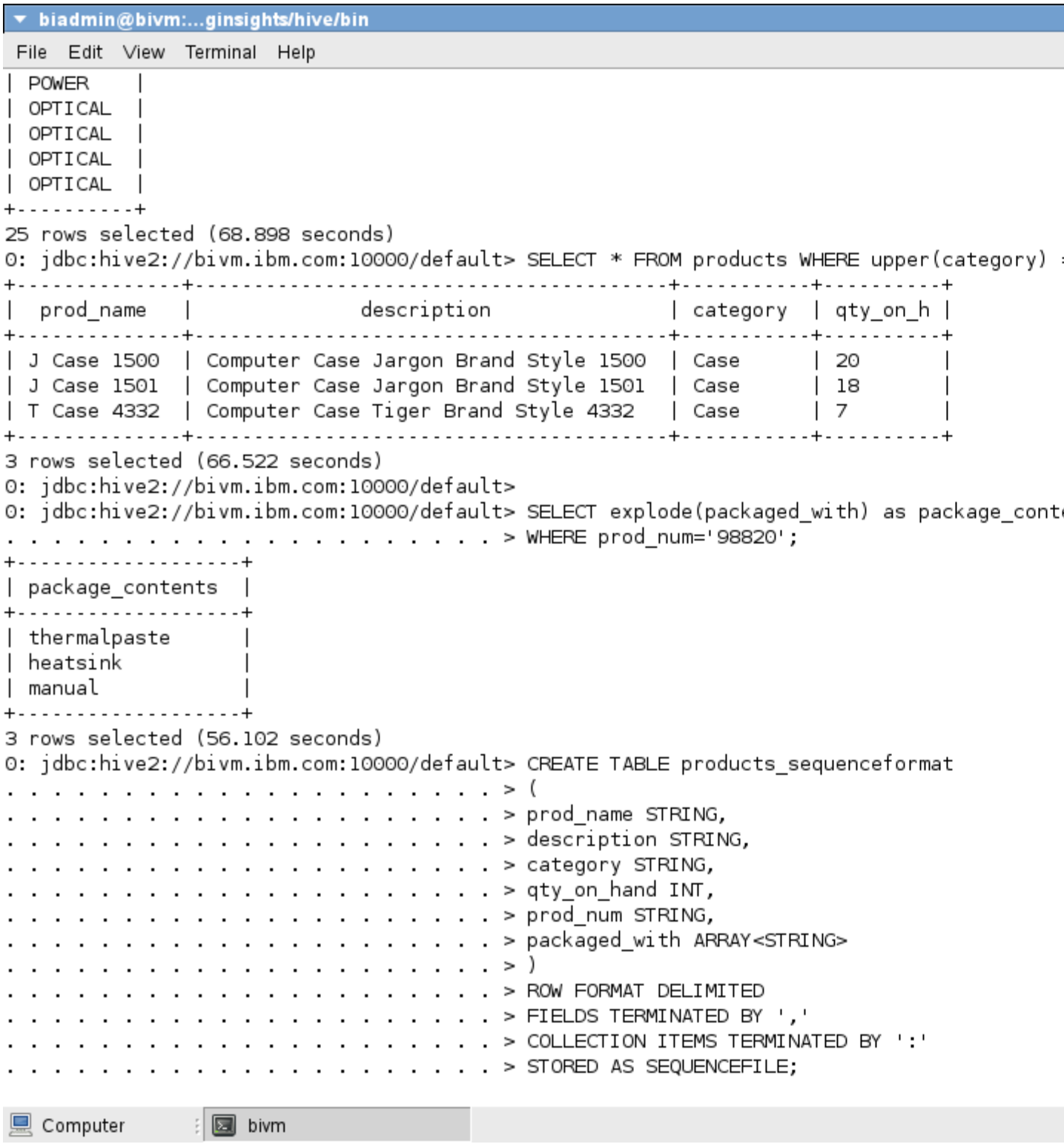
Computer bivm

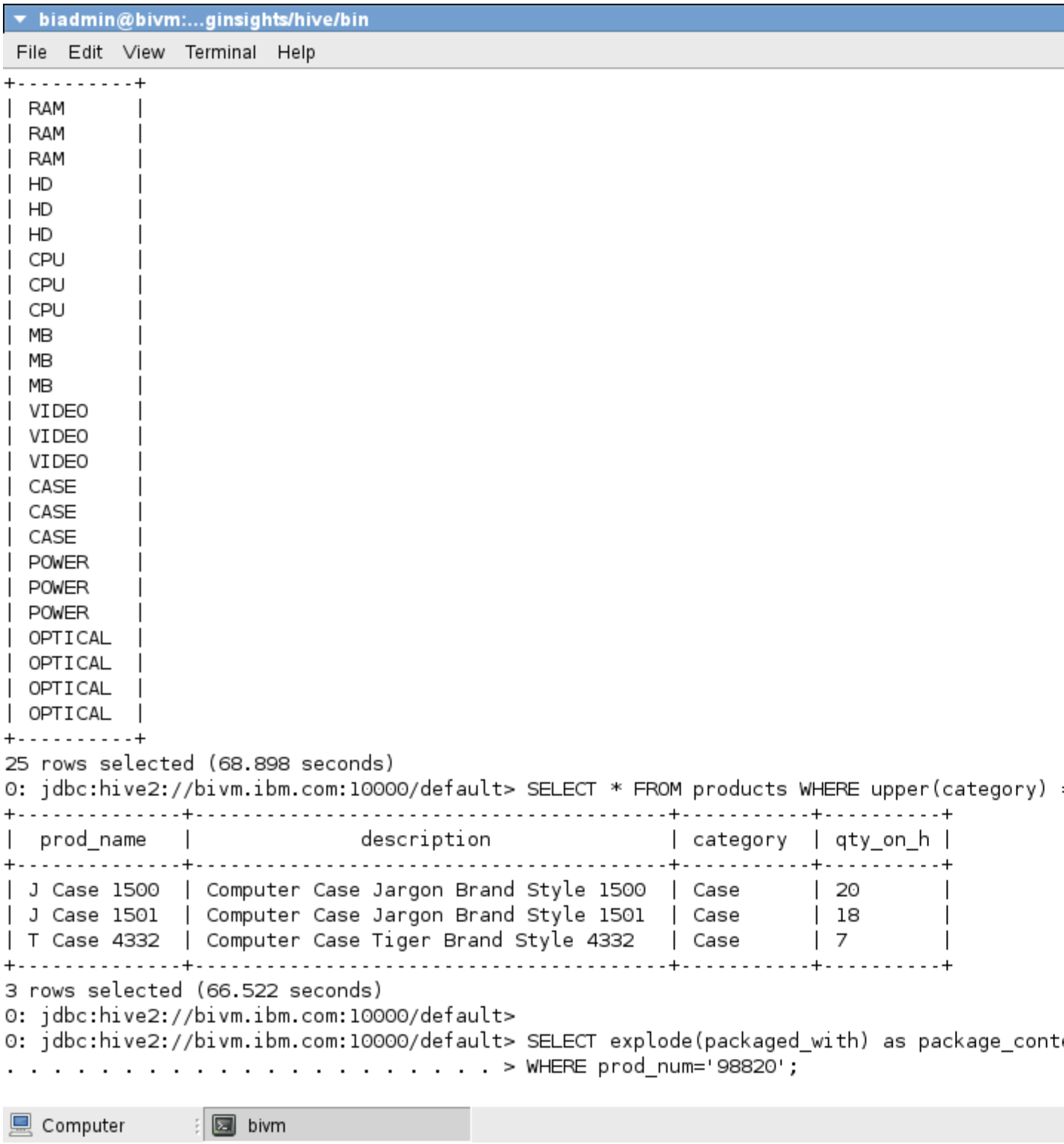

biadmin@bivm:...ginsights/hive/bin
File Edit View Terminal Help
| prod_num | string |
| packaged_with | array<string> |
| | |
| Detailed Table Information | Table(tableName:products_rcfileformat, dbName: |
8 rows selected (1.582 seconds)
0: jdbc:hive2://bivm.ibm.com:10000/default> FROM products INSERT OVERWRITE TABLE products
. . . . . > SELECT *;
No rows affected (53.117 seconds)
0: jdbc:hive2://bivm.ibm.com:10000/default> SELECT * FROM products_rcfileformat;
| prod_name | description | category | qty_o |
| 2 GB Memory E | 2 GB Memory ECC | Ram | 3000 |
| 4 GB Memory E | 4 GB Memory ECC | Ram | 1000 |
| 16 GB Memory E | 16 GB Memory ECC | Ram | 238 |
| 500 GB HD J | 500 GB HD Panther Brand | HD | 200 |
| 500 GB HD T | 500 GB HD Tiger Brand | HD | 498 |
| 1 TB HD J | 1 TB HD Jargon Brand | HD | 231 |
| 4 Core CPU J3 | 4 Core CPU Jargon Brand 3 GHZ | CPU | 50 |
| 2 Core CPU J2 | 2 Core CPU Jargon Brand 2 GHZ | CPU | 118 |
| 1 Core CPU J2 | 1 Core CPU Jargon Brand 2 GHZ | CPU | 203 |
| 94F991 MB | Motherboard F991 CPU | MB | 19 |
| 94G822 MB | Motherboard G822 CPU | MB | 30 |
| 93H772 MB | Motherboard H772 CPU | MB | 15 |
| 93G Video | Video Card Jargon Brand 93G | Video | 80 |
| 84G1 Video | Video Card Jargon Brand 84F1 | Video | 14 |
| 09K Video | Video Card Tiger Brand 84F1 | Video | 5 |
| J Case 1500 | Computer Case Jargon Brand Style 1500 | Case | 20 |
| J Case 1501 | Computer Case Jargon Brand Style 1501 | Case | 18 |
| T Case 4332 | Computer Case Tiger Brand Style 4332 | Case | 7 |
| J Power 300W | Power Supply Jargon Brand 300 Watts | Power | 28 |
| J Power 500W | Power Supply Jargon Brand 500 Watts | Power | 17 |
| T Power 300W | Power Supply Tiger Brand 300 Watts | Power | 8 |
| DVD J INT | DVD Jargon Brand Internal | Optical | 23 |
| DVD J EXT | DVD Jargon Brand External | Optical | 45 |
| DVD T INT | DVD Tiger Brand Internal | Optical | 19 |
| DVD T EXT | DVD Tiger Brand External | Optical | 17 |
25 rows selected (1.212 seconds)
0: jdbc:hive2://bivm.ibm.com:10000/default>
Computer
bivm

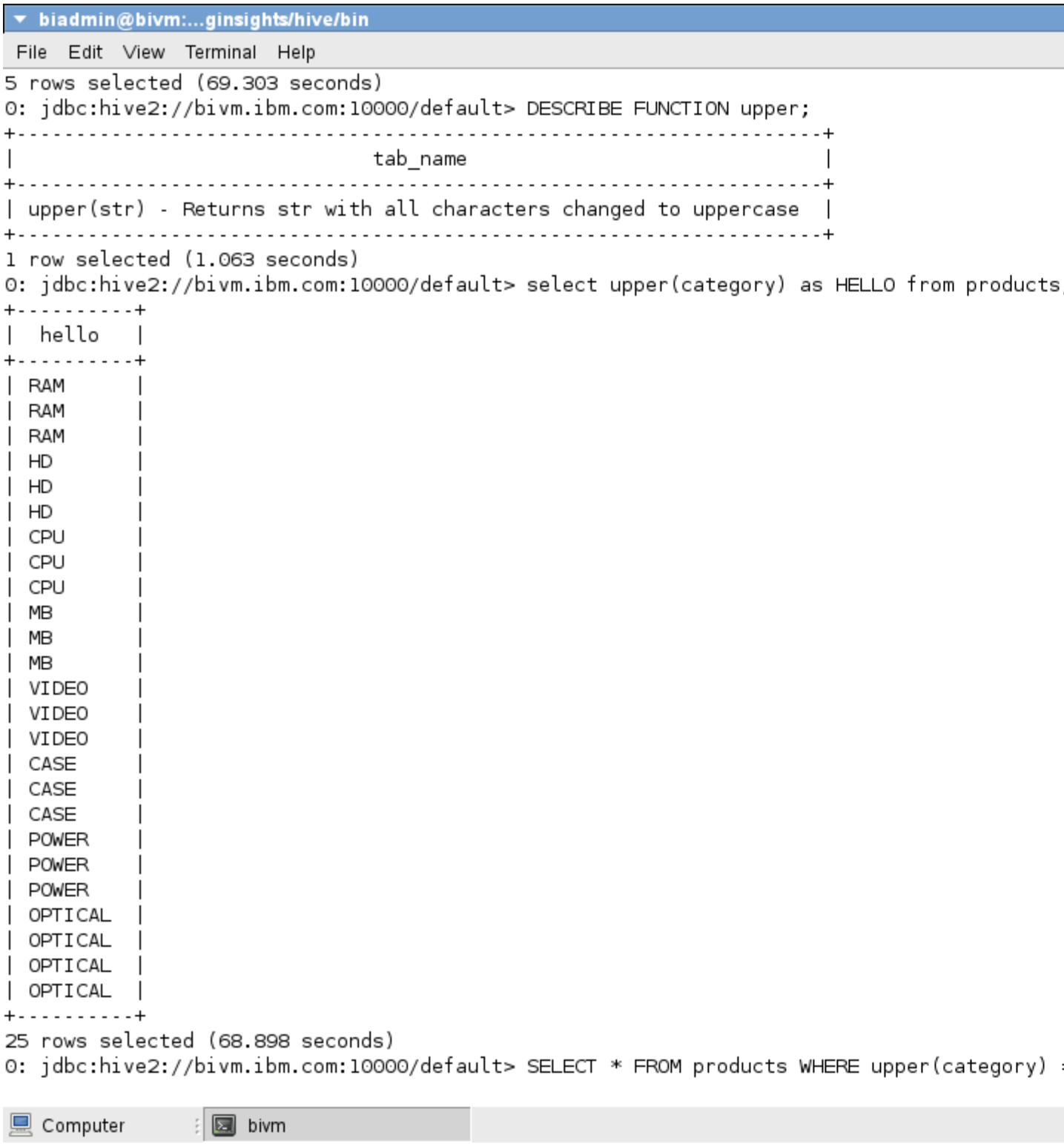

biadmin@bivm:...ginsights/hive/bin
File Edit View Terminal Help
0: jdbc:hive2://bivm.ibm.com:10000/default> show tables;
| tab_name |
| customer |
| optical_sales |
| products |
| products_sequenceformat |
| sales |
| sales_staging |
6 rows selected (0.938 seconds)
0: jdbc:hive2://bivm.ibm.com:10000/default> DESCRIBE EXTENDED products_sequenceformat;
| col_name | |
| prod_name | string |
| description | string |
| category | string |
| qty_on_hand | int |
| prod_num | string |
| packaged_with | array<string> |
| Detailed Table Information | Table(tableName:products_sequenceformat, dbNam |
8 rows selected (1.379 seconds)
0: jdbc:hive2://bivm.ibm.com:10000/default> FROM products INSERT OVERWRITE TABLE products
No rows affected (59.428 seconds)
0: jdbc:hive2://bivm.ibm.com:10000/default> CREATE TABLE products_rcfileformat
. . . . . > (
. . . . . > prod_name STRING,
. . . . . > description STRING,
. . . . . > category STRING,
. . . . . > qty_on_hand INT,
. . . . . > prod_num STRING,
. . . . . > packaged_with ARRAY<STRING>
. . . . . > )
. . . . . > ROW FORMAT SERDE 'org.apache.hadoop.hive.serd
. . . . . > STORED AS
. . . . . > INPUTFORMAT 'org.apache.hadoop.hive.ql.io.RCF
. . . . . > OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.RC
Computer
bivm


biadmin@bivm:...ginsights/hive/bin
File Edit View Terminal Help
| col_name | |
| prod_name | string |
| description | string |
| category | string |
| qty_on_hand | int |
| prod_num | string |
| packaged_with | array<string> |
| Detailed Table Information | Table(tableName:products_sequenceformat, dbNam |
8 rows selected (1.379 seconds)
0: jdbc:hive2://bivm.ibm.com:10000/default> FROM products INSERT OVERWRITE TABLE products_
No rows affected (59.428 seconds)
0: jdbc:hive2://bivm.ibm.com:10000/default> CREATE TABLE products_rcfileformat
. . . . . > (
. . . . . > prod_name STRING,
. . . . . > description STRING,
. . . . . > category STRING,
. . . . . > qty_on_hand INT,
. . . . . > prod_num STRING,
. . . . . > packaged_with ARRAY<STRING>
. . . . . > )
. . . . . > ROW FORMAT SERDE 'org.apache.hadoop.hive.serde
. . . . . > STORED AS
. . . . . > INPUTFORMAT 'org.apache.hadoop.hive.ql.io.RCF
. . . . . > OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.RCF
No rows affected (1.328 seconds)
0: jdbc:hive2://bivm.ibm.com:10000/default> DESCRIBE EXTENDED products_rcfileformat;
| col_name | |
| prod_name | string |
| description | string |
| category | string |
| qty_on_hand | int |
| prod_num | string |
| packaged_with | array<string> |
| Detailed Table Information | Table(tableName:products_rcfileformat, dbName: |
Computer
bivm

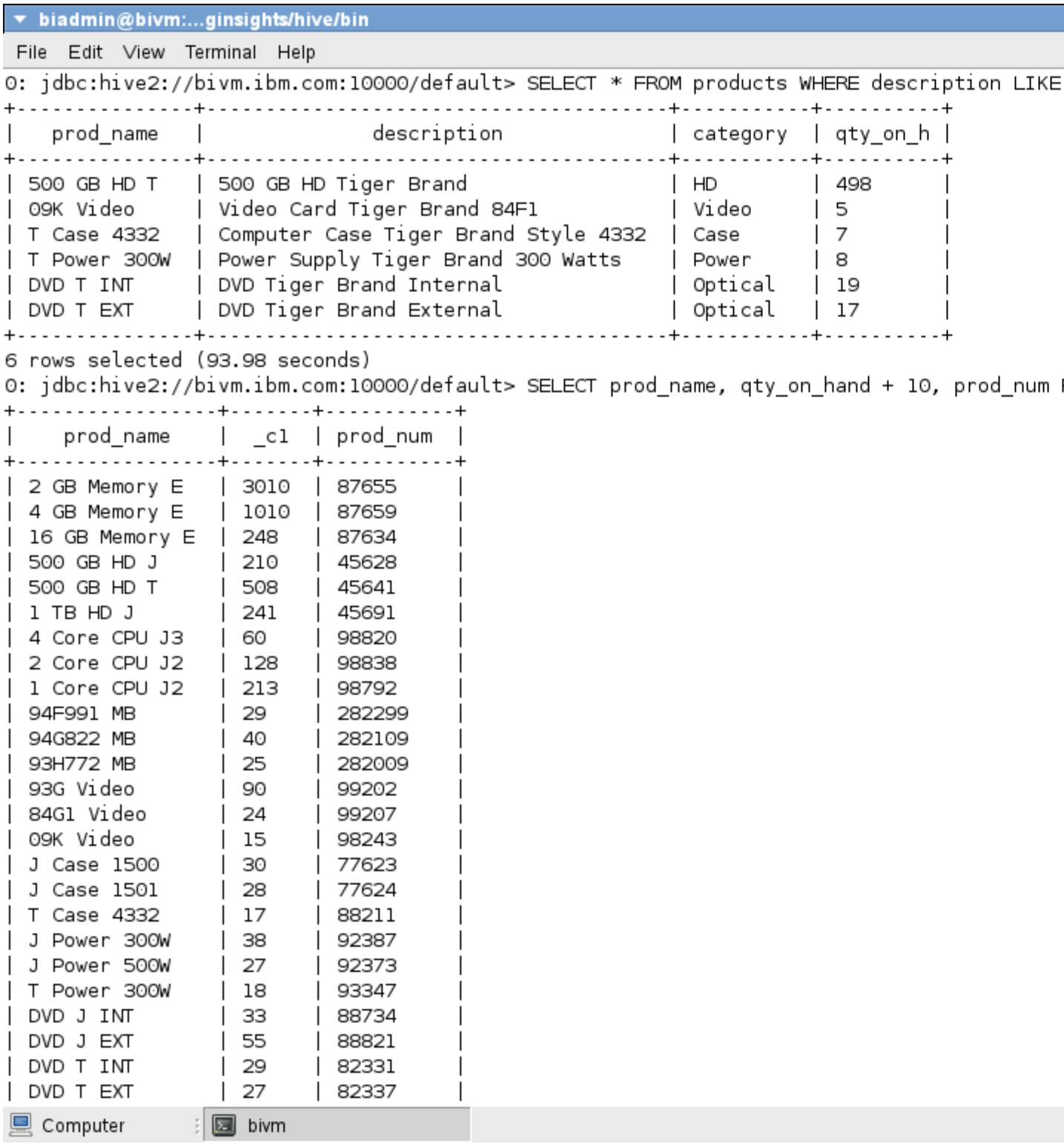

biadmin@bivm:...ginsights/hive/bin
File Edit View Terminal Help
3 rows selected (66.522 seconds)
0: jdbc:hive2://bivm.ibm.com:10000/default>
0: jdbc:hive2://bivm.ibm.com:10000/default> SELECT explode(packaged_with) as package_conte
. . . . . > WHERE prod_num='98820';
+------------------+
| package_contents |
+------------------+
| thermalpaste |
| heatsink |
| manual |
+------------------+
3 rows selected (56.102 seconds)
0: jdbc:hive2://bivm.ibm.com:10000/default> CREATE TABLE products_sequenceformat
. . . . . > (
. . . . . > prod_name STRING,
. . . . . > description STRING,
. . . . . > category STRING,
. . . . . > qty_on_hand INT,
. . . . . > prod_num STRING,
. . . . . > packaged_with ARRAY<STRING>
. . . . . > )
. . . . . > ROW FORMAT DELIMITED
. . . . . > FIELDS TERMINATED BY ','
. . . . . > COLLECTION ITEMS TERMINATED BY ':'
. . . . . > STORED AS SEQUENCEFILE;
No rows affected (9.119 seconds)
0: jdbc:hive2://bivm.ibm.com:10000/default> show tables;show tables;
Error: Error while processing statement: FAILED: ParseException line 1:11 missing EOF at
0: jdbc:hive2://bivm.ibm.com:10000/default> show tables;
+--------------------------+
| tab_name |
+--------------------------+
| customer |
| optical_sales |
| products |
| products_sequenceformat |
| sales |
| sales_staging |
+--------------------------+
6 rows selected (0.938 seconds)
0: jdbc:hive2://bivm.ibm.com:10000/default>
Computer
bivm

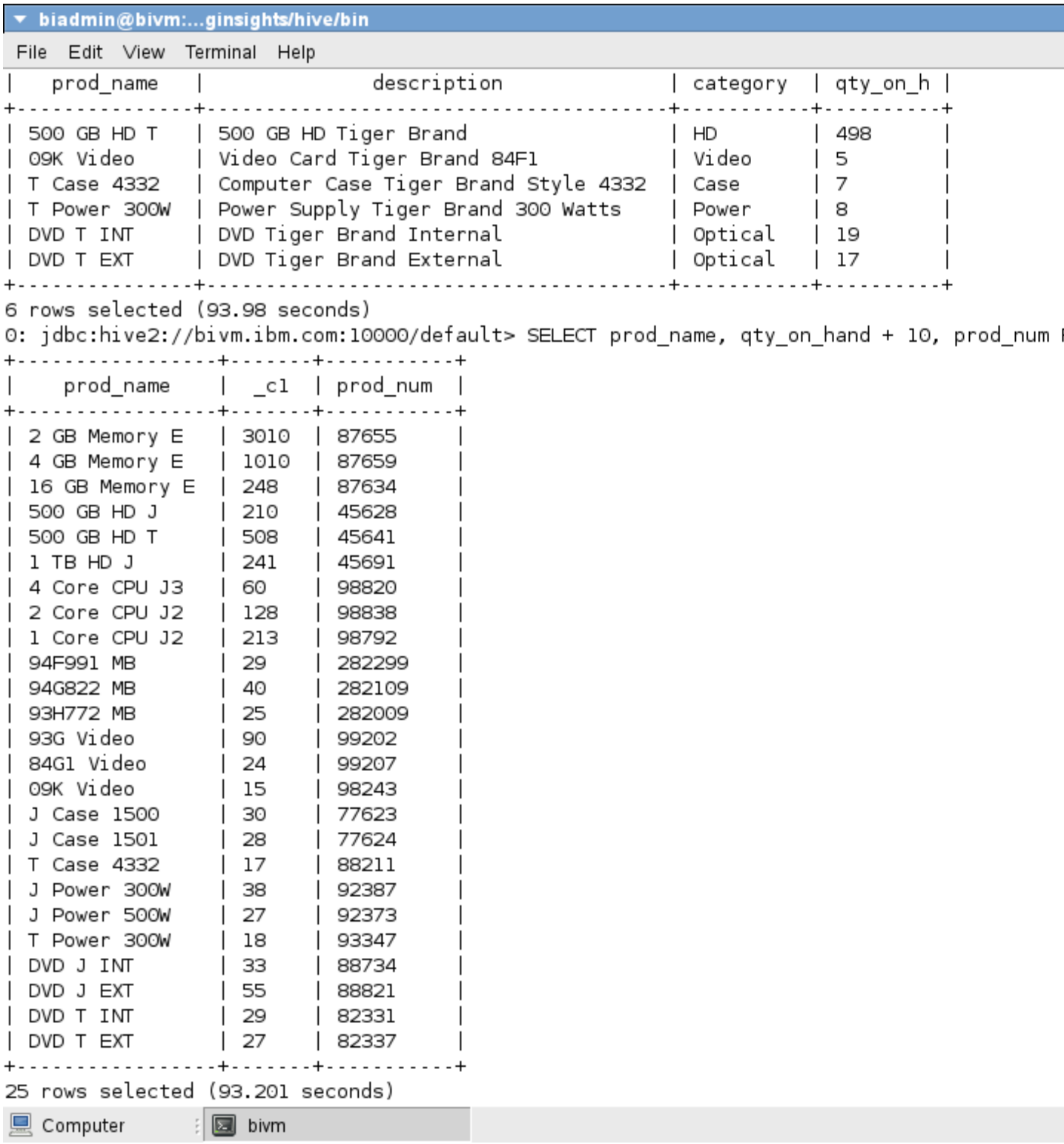

biadmin@bivm:...ginsights/hive/bin
File Edit View Terminal Help
| POWER |
| OPTICAL |
| OPTICAL |
| OPTICAL |
| OPTICAL |
+----------+
25 rows selected (68.898 seconds)
0: jdbc:hive2://bivm.ibm.com:10000/default> SELECT * FROM products WHERE upper(category) =
+--------------+----------+----------+
| prod_name | description | category | qty_on_h |
+--------------+----------+----------+
| J Case 1500 | Computer Case Jargon Brand Style 1500 | Case | 20 |
| J Case 1501 | Computer Case Jargon Brand Style 1501 | Case | 18 |
| T Case 4332 | Computer Case Tiger Brand Style 4332 | Case | 7 |
+--------------+----------+----------+
3 rows selected (66.522 seconds)
0: jdbc:hive2://bivm.ibm.com:10000/default>
0: jdbc:hive2://bivm.ibm.com:10000/default> SELECT explode(packaged_with) as package_conte
. . . . . > WHERE prod_num='98820';
+------------------+
| package_contents |
+------------------+
| thermalpaste |
| heatsink |
| manual |
+------------------+
3 rows selected (56.102 seconds)
0: jdbc:hive2://bivm.ibm.com:10000/default> CREATE TABLE products_sequenceformat
. . . . . > (
. . . . . > prod_name STRING,
. . . . . > description STRING,
. . . . . > category STRING,
. . . . . > qty_on_hand INT,
. . . . . > prod_num STRING,
. . . . . > packaged_with ARRAY<STRING>
. . . . . > )
. . . . . > ROW FORMAT DELIMITED
. . . . . > FIELDS TERMINATED BY ','
. . . . . > COLLECTION ITEMS TERMINATED BY ':'
. . . . . > STORED AS SEQUENCEFILE;
Computer
bivm


biadmin@bivm:...ginsights/hive/bin
File Edit View Terminal Help
+----------+
| RAM |
| RAM |
| RAM |
| HD |
| HD |
| HD |
| CPU |
| CPU |
| CPU |
| MB |
| MB |
| MB |
| VIDEO |
| VIDEO |
| VIDEO |
| CASE |
| CASE |
| CASE |
| POWER |
| POWER |
| POWER |
| OPTICAL |
| OPTICAL |
| OPTICAL |
| OPTICAL |
+----------+
25 rows selected (68.898 seconds)
0: jdbc:hive2://bivm.ibm.com:10000/default> SELECT * FROM products WHERE upper(category) =
+--------------+----------------------------------------+----------+----------+
| prod_name | description | category | qty_on_h |
+--------------+----------------------------------------+----------+----------+
| J Case 1500 | Computer Case Jargon Brand Style 1500 | Case | 20 |
| J Case 1501 | Computer Case Jargon Brand Style 1501 | Case | 18 |
| T Case 4332 | Computer Case Tiger Brand Style 4332 | Case | 7 |
+--------------+----------------------------------------+----------+----------+
3 rows selected (66.522 seconds)
0: jdbc:hive2://bivm.ibm.com:10000/default>
0: jdbc:hive2://bivm.ibm.com:10000/default> SELECT explode(packaged_with) as package_conte
. . . . . > WHERE prod_num='98820';
Computer
bivm

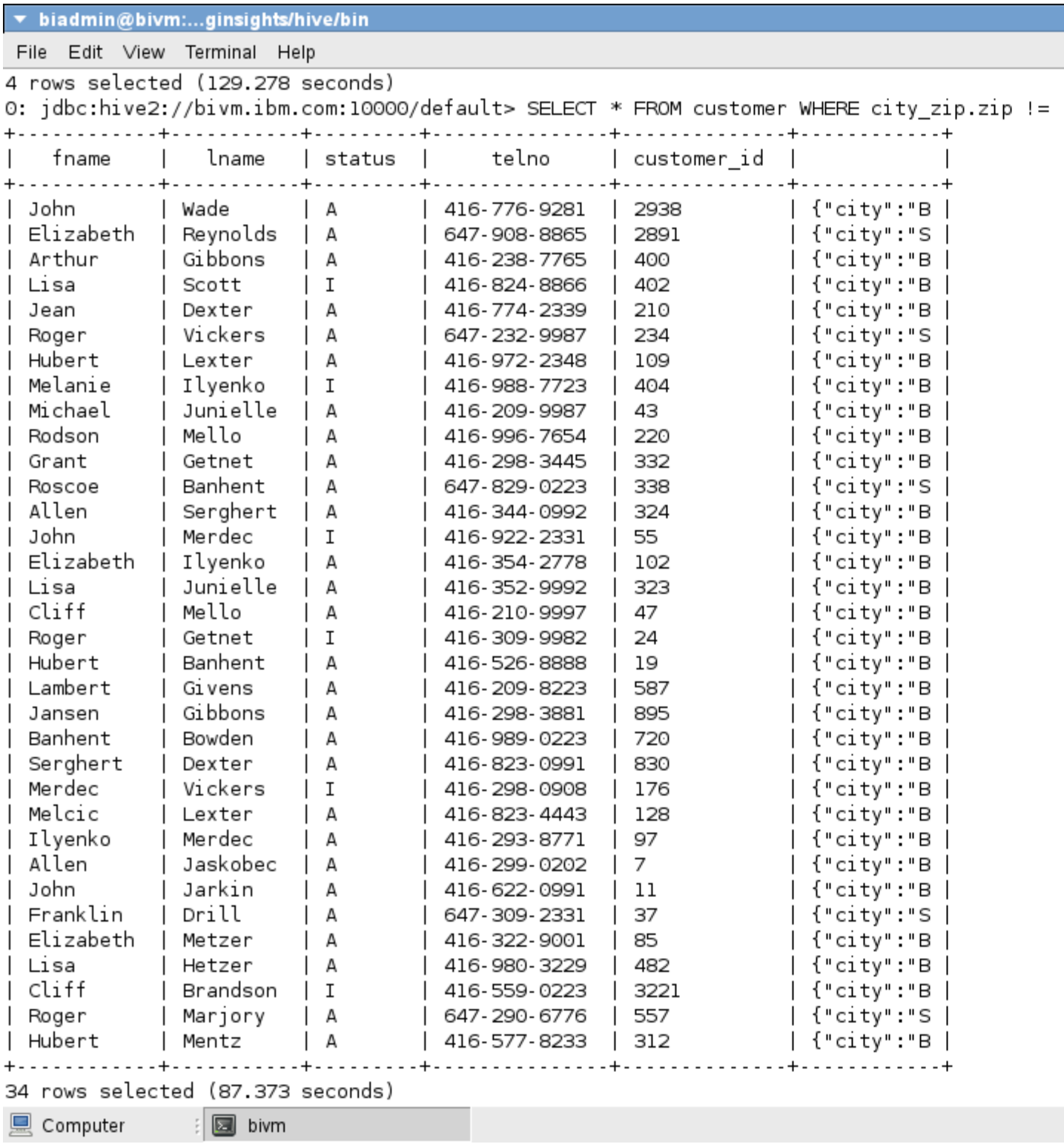

biadmin@bivm:...ginsights/hive/bin
File Edit View Terminal Help
5 rows selected (69.303 seconds)
0: jdbc:hive2://bivm.ibm.com:10000/default> DESCRIBE FUNCTION upper;
tab_name
upper(str) - Returns str with all characters changed to uppercase
1 row selected (1.063 seconds)
0: jdbc:hive2://bivm.ibm.com:10000/default> select upper(category) as HELLO from products
hello
RAM
RAM
RAM
HD
HD
HD
CPU
CPU
CPU
MB
MB
MB
VIDEO
VIDEO
VIDEO
CASE
CASE
CASE
POWER
POWER
POWER
OPTICAL
OPTICAL
OPTICAL
OPTICAL
25 rows selected (68.898 seconds)
0: jdbc:hive2://bivm.ibm.com:10000/default> SELECT * FROM products WHERE upper(category)
Computer
bivm

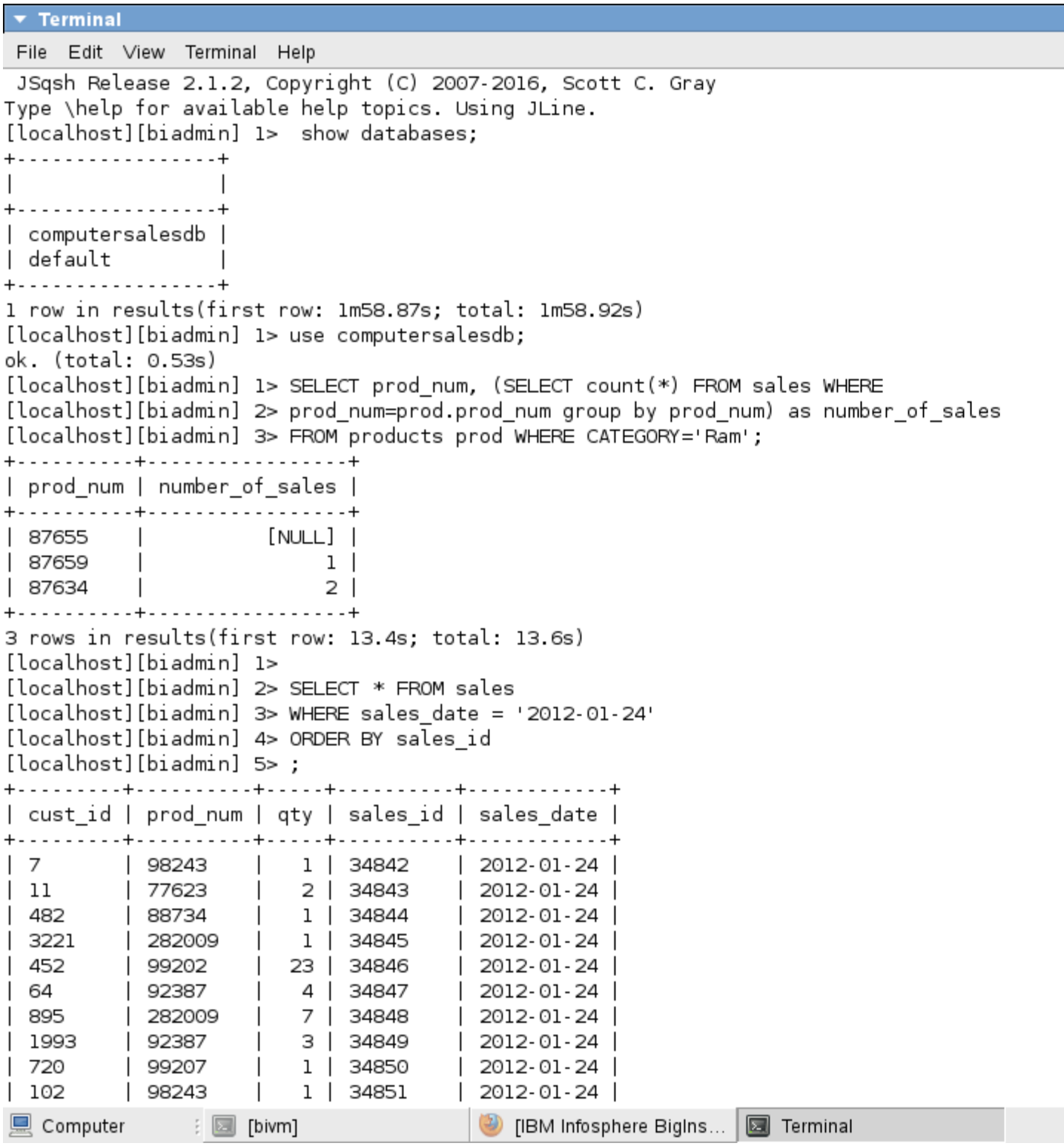
```
biadmin@bivm:...ginsights/hive/bin
File  Edit  View  Terminal  Help
| 94G822 MB        | 40    | 282109    |
| 93H772 MB        | 25    | 282009    |
| 93G Video        | 90    | 99202     |
| 84G1 Video       | 24    | 99207     |
| 09K Video        | 15    | 98243     |
| J Case 1500      | 30    | 77623     |
| J Case 1501      | 28    | 77624     |
| T Case 4332      | 17    | 88211     |
| J Power 300W     | 38    | 92387     |
| J Power 500W     | 27    | 92373     |
| T Power 300W     | 18    | 93347     |
| DVD J INT        | 33    | 88734     |
| DVD J EXT        | 55    | 88821     |
| DVD T INT        | 29    | 82331     |
| DVD T EXT        | 27    | 82337     |
+------------------+-------+-----------+
25 rows selected (93.201 seconds)
0: jdbc:hive2://bivm.ibm.com:10000/default> SELECT * FROM customer WHERE lname='Merdec' OR
Error: Error while processing statement: FAILED: ParseException line 1:35 character '' no
line 1:42 character '' not supported here
line 1:53 character '' not supported here
line 1:60 character '' not supported here (state=42000,code=40000)
0: jdbc:hive2://bivm.ibm.com:10000/default> SELECT * FROM customer WHERE lname='Merdec' OR
+-----------+---------+---------+---------------+--------------+---------------+
|   fname   |  lname  | status  |     telno     | customer_id  |               |
+-----------+---------+---------+---------------+--------------+---------------+
| Jack      | Merdec  | A       | 905-216-0989  | 54           | {"city":"Medi |
| Jovan     | Melcic  | A       | 905-324-4456  | 92           | {"city":"Medi |
| John      | Merdec  | I       | 416-922-2331  | 55           | {"city":"Bigc |
| Franklin  | Melcic  | A       | 905-561-2330  | 647          | {"city":"Medi |
| Ilyenko   | Merdec  | A       | 416-293-8771  | 97           | {"city":"Bigc |
+-----------+---------+---------+---------------+--------------+---------------+
5 rows selected (69.303 seconds)
0: jdbc:hive2://bivm.ibm.com:10000/default> DESCRIBE FUNCTION upper;
+-----------------------------------------------------------------+
|                            tab_name                             |
+-----------------------------------------------------------------+
| upper(str) - Returns str with all characters changed to uppercase  |
+-----------------------------------------------------------------+
1 row selected (1.063 seconds)
0: jdbc:hive2://bivm.ibm.com:10000/default>
Computer   bivm
```

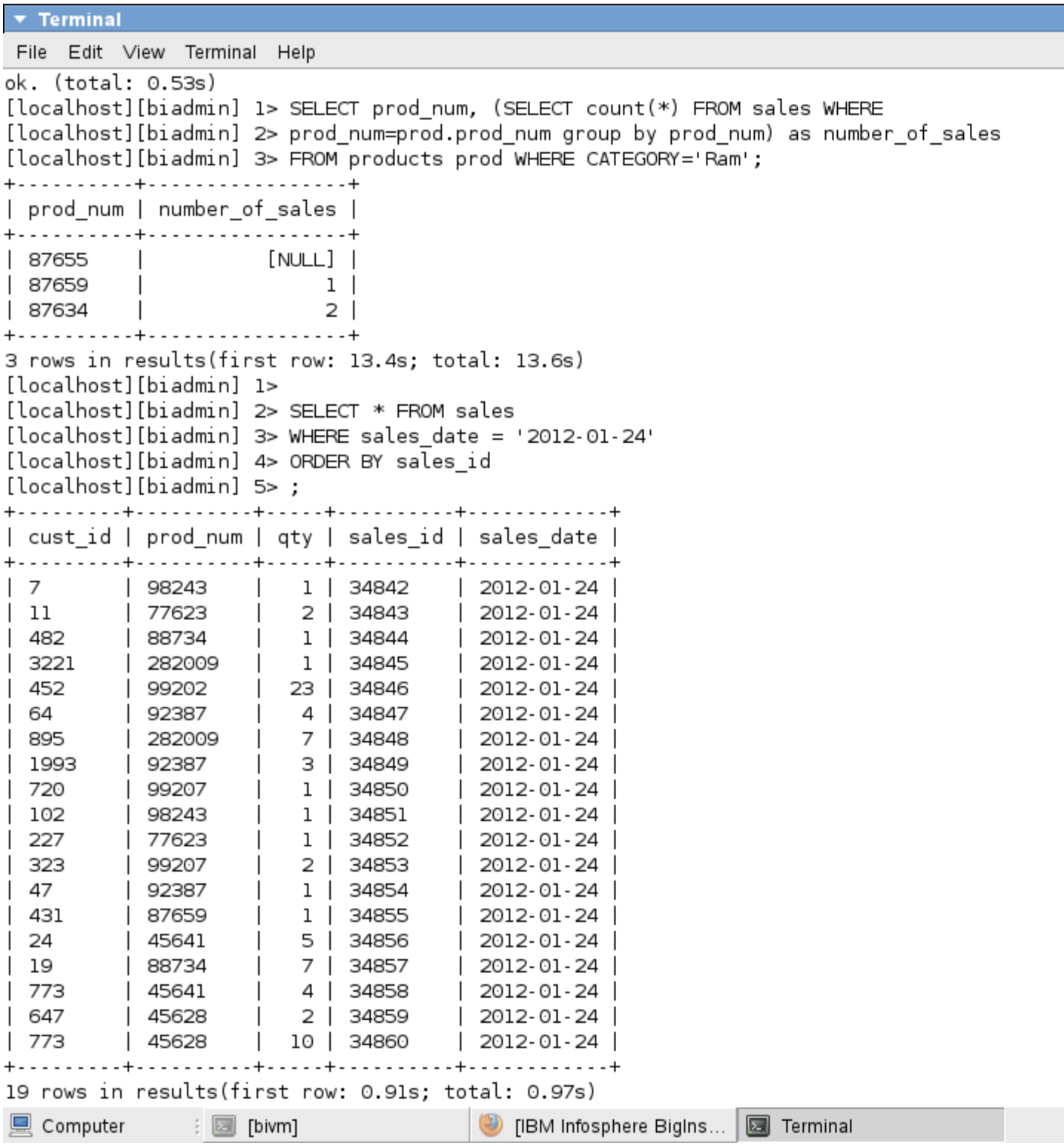

biadmin@bivm:...ginsights/hive/bin
File Edit View Terminal Help
0: jdbc:hive2://bivm.ibm.com:10000/default> SELECT * FROM products WHERE description LIKE
| prod_name | description | category | qty_on_h |
| 500 GB HD T | 500 GB HD Tiger Brand | HD | 498 |
| 09K Video | Video Card Tiger Brand 84F1 | Video | 5 |
| T Case 4332 | Computer Case Tiger Brand Style 4332 | Case | 7 |
| T Power 300W | Power Supply Tiger Brand 300 Watts | Power | 8 |
| DVD T INT | DVD Tiger Brand Internal | Optical | 19 |
| DVD T EXT | DVD Tiger Brand External | Optical | 17 |
6 rows selected (93.98 seconds)
0: jdbc:hive2://bivm.ibm.com:10000/default> SELECT prod_name, qty_on_hand + 10, prod_num
| prod_name | _c1 | prod_num |
| 2 GB Memory E | 3010 | 87655 |
| 4 GB Memory E | 1010 | 87659 |
| 16 GB Memory E | 248 | 87634 |
| 500 GB HD J | 210 | 45628 |
| 500 GB HD T | 508 | 45641 |
| 1 TB HD J | 241 | 45691 |
| 4 Core CPU J3 | 60 | 98820 |
| 2 Core CPU J2 | 128 | 98838 |
| 1 Core CPU J2 | 213 | 98792 |
| 94F991 MB | 29 | 282299 |
| 94G822 MB | 40 | 282109 |
| 93H772 MB | 25 | 282009 |
| 93G Video | 90 | 99202 |
| 84G1 Video | 24 | 99207 |
| 09K Video | 15 | 98243 |
| J Case 1500 | 30 | 77623 |
| J Case 1501 | 28 | 77624 |
| T Case 4332 | 17 | 88211 |
| J Power 300W | 38 | 92387 |
| J Power 500W | 27 | 92373 |
| T Power 300W | 18 | 93347 |
| DVD J INT | 33 | 88734 |
| DVD J EXT | 55 | 88821 |
| DVD T INT | 29 | 82331 |
| DVD T EXT | 27 | 82337 |
Computer
bivm

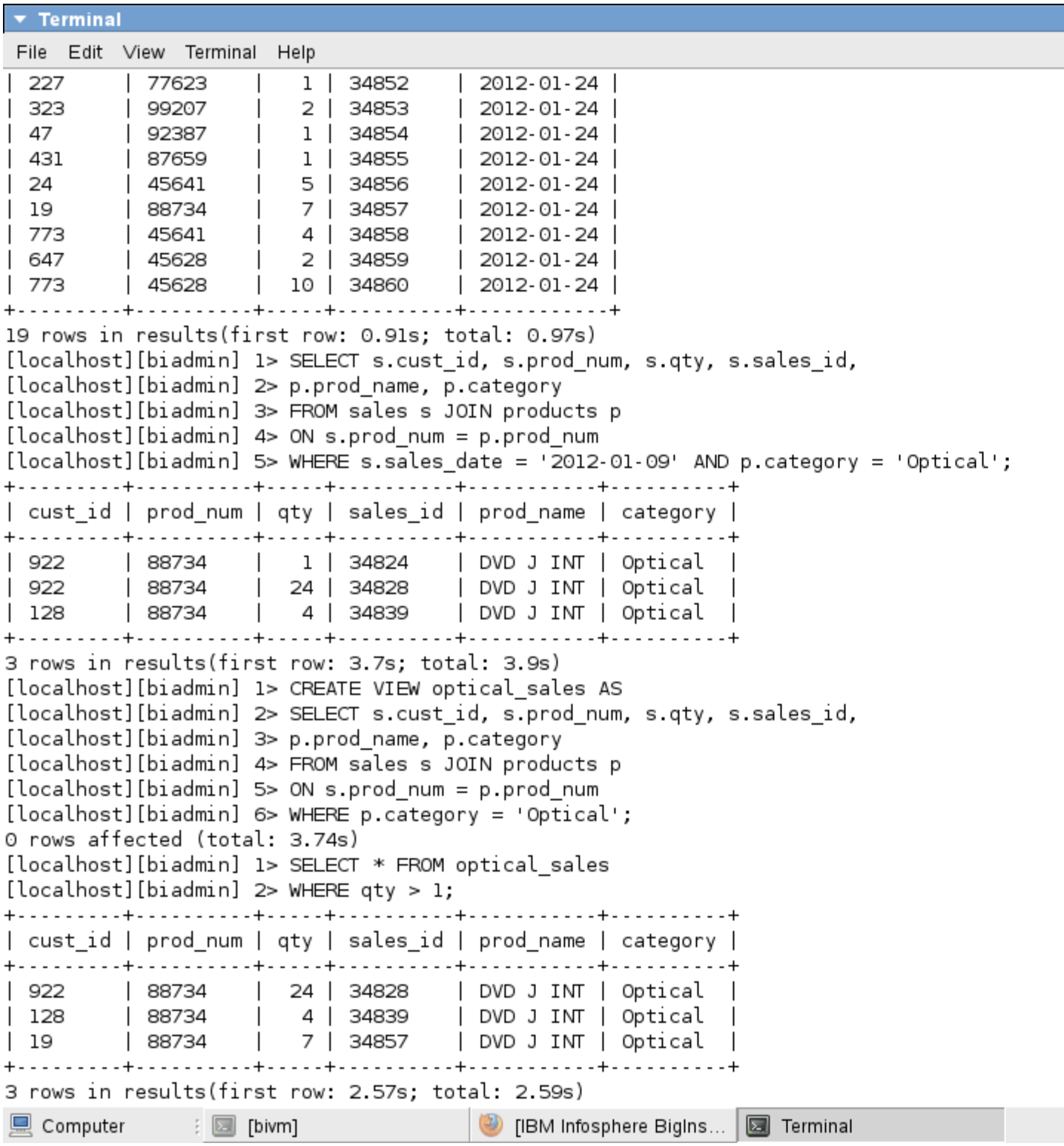

biadmin@bivm:...ginsights/hive/bin
File Edit View Terminal Help
| prod_name | description | category | qty_on_h |
| 500 GB HD T | 500 GB HD Tiger Brand | HD | 498 |
| 09K Video | Video Card Tiger Brand 84F1 | Video | 5 |
| T Case 4332 | Computer Case Tiger Brand Style 4332 | Case | 7 |
| T Power 300W | Power Supply Tiger Brand 300 Watts | Power | 8 |
| DVD T INT | DVD Tiger Brand Internal | Optical | 19 |
| DVD T EXT | DVD Tiger Brand External | Optical | 17 |
6 rows selected (93.98 seconds)
0: jdbc:hive2://bivm.ibm.com:10000/default> SELECT prod_name, qty_on_hand + 10, prod_num
| prod_name | _c1 | prod_num |
| 2 GB Memory E | 3010 | 87655 |
| 4 GB Memory E | 1010 | 87659 |
| 16 GB Memory E | 248 | 87634 |
| 500 GB HD J | 210 | 45628 |
| 500 GB HD T | 508 | 45641 |
| 1 TB HD J | 241 | 45691 |
| 4 Core CPU J3 | 60 | 98820 |
| 2 Core CPU J2 | 128 | 98838 |
| 1 Core CPU J2 | 213 | 98792 |
| 94F991 MB | 29 | 282299 |
| 94G822 MB | 40 | 282109 |
| 93H772 MB | 25 | 282009 |
| 93G Video | 90 | 99202 |
| 84G1 Video | 24 | 99207 |
| 09K Video | 15 | 98243 |
| J Case 1500 | 30 | 77623 |
| J Case 1501 | 28 | 77624 |
| T Case 4332 | 17 | 88211 |
| J Power 300W | 38 | 92387 |
| J Power 500W | 27 | 92373 |
| T Power 300W | 18 | 93347 |
| DVD J INT | 33 | 88734 |
| DVD J EXT | 55 | 88821 |
| DVD T INT | 29 | 82331 |
| DVD T EXT | 27 | 82337 |
25 rows selected (93.201 seconds)
Computer
bivm

```
biadmin@bivm:...ginsights/hive/bin
File  Edit  View  Terminal  Help
0: jdbc:hive2://bivm.ibm.com:10000/default> use computersalesdb;
No rows affected (16.152 seconds)
0: jdbc:hive2://bivm.ibm.com:10000/default> INSERT OVERWRITE DIRECTORY '/user/biadmin/MRI
. . . . . . . . . . . . . . . . . . . . . . > select * from optical_sales where qty > 1;
No rows affected (173.404 seconds)
0: jdbc:hive2://bivm.ibm.com:10000/default>
0: jdbc:hive2://bivm.ibm.com:10000/default>
0: jdbc:hive2://bivm.ibm.com:10000/default>
0: jdbc:hive2://bivm.ibm.com:10000/default> select * from optical_sales where qty > 1;
+----------+-----------+------+-----------+------------+-----------+
| cust_id  | prod_num  | qty  | sales_id  | prod_name  | category  |
+----------+-----------+------+-----------+------------+-----------+
| 19       | 88734     | 7    | 34857     | DVD J INT  | Optical   |
| 922      | 88734     | 24   | 34828     | DVD J INT  | Optical   |
| 128      | 88734     | 4    | 34839     | DVD J INT  | Optical   |
+----------+-----------+------+-----------+------------+-----------+
3 rows selected (101.127 seconds)
0: jdbc:hive2://bivm.ibm.com:10000/default>  SELECT * FROM products WHERE packaged_with[0
+------------+----------------------------+-----------+--------------+---------+
| prod_name  |        description         | category  | qty_on_hand  | prod_nu |
+------------+----------------------------+-----------+--------------+---------+
| 1 TB HD J  | 1 TB HD Jargon Brand       | HD        | 231          | 45691   |
| 94F991 MB  | Motherboard F991 CPU       | MB        | 19           | 282299  |
| 94G822 MB  | Motherboard G822 CPU       | MB        | 30           | 282109  |
| DVD J EXT  | DVD Jargon Brand External  | Optical   | 45           | 88821   |
+------------+----------------------------+-----------+--------------+---------+
4 rows selected (129.278 seconds)
0: jdbc:hive2://bivm.ibm.com:10000/default> SELECT * FROM customer WHERE city_zip.zip !=
+------------+-----------+---------+---------------+--------------+------------+
|   fname    |   lname   | status  |     telno     | customer_id  |            |
+------------+-----------+---------+---------------+--------------+------------+
| John       | Wade      | A       | 416-776-9281  | 2938         | {"city":"B |
| Elizabeth  | Reynolds  | A       | 647-908-8865  | 2891         | {"city":"S |
| Arthur     | Gibbons   | A       | 416-238-7765  | 400          | {"city":"B |
| Lisa       | Scott     | I       | 416-824-8866  | 402          | {"city":"B |
| Jean       | Dexter    | A       | 416-774-2339  | 210          | {"city":"B |
| Roger      | Vickers   | A       | 647-232-9987  | 234          | {"city":"S |
| Hubert     | Lexter    | A       | 416-972-2348  | 109          | {"city":"B |
| Melanie    | Ilyenko   | I       | 416-988-7723  | 404          | {"city":"B |
| Michael    | Junielle  | A       | 416-209-9987  | 43           | {"city":"B |
| Rodson     | Mello     | A       | 416-996-7654  | 220          | {"city":"B |
Computer  bivm
```

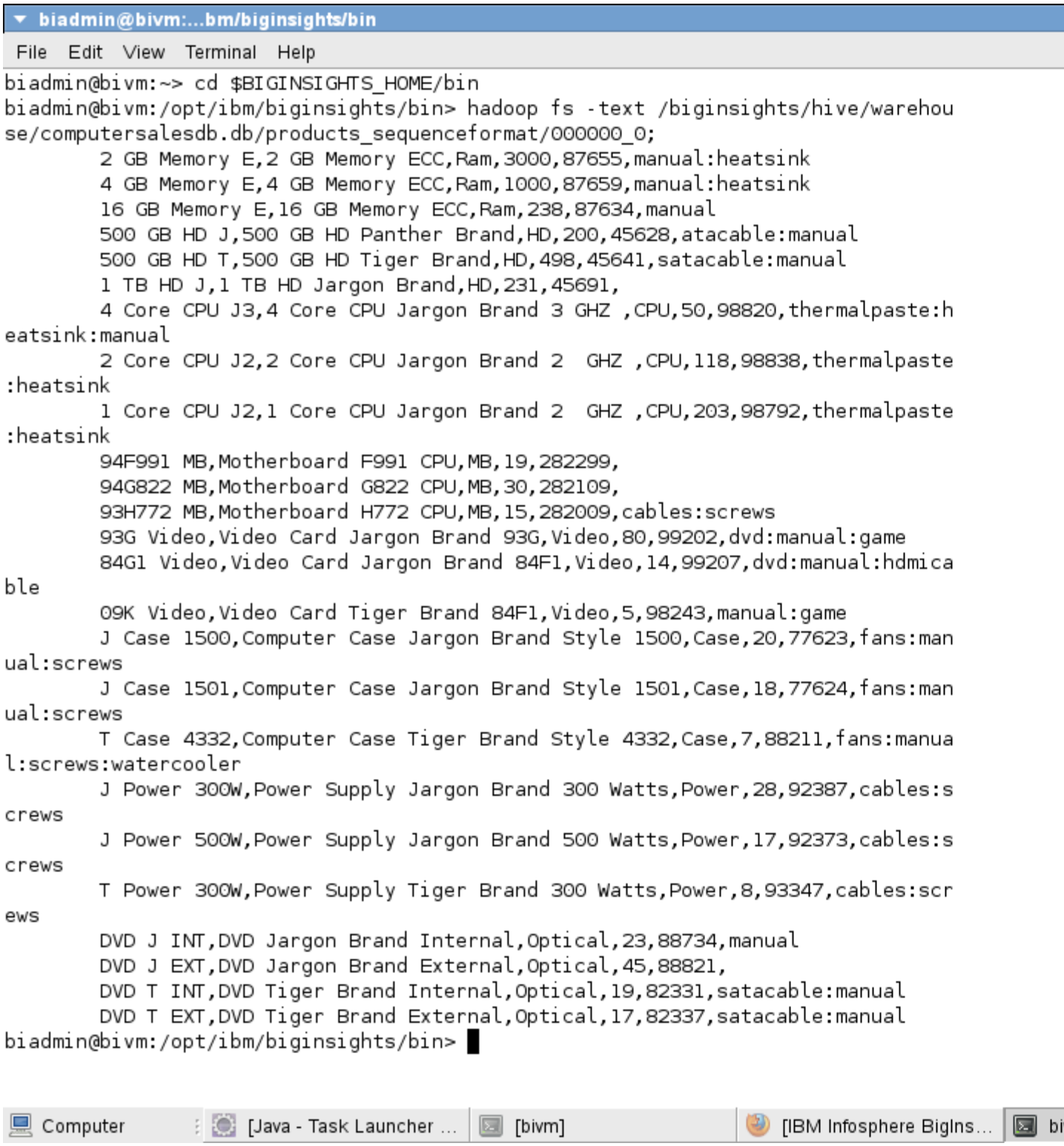

biadmin@bivm:...ginsights/hive/bin
File Edit View Terminal Help
4 rows selected (129.278 seconds)
0: jdbc:hive2://bivm.ibm.com:10000/default> SELECT * FROM customer WHERE city_zip.zip !=
| fname | lname | status | telno | customer_id | |
| John | Wade | A | 416-776-9281 | 2938 | {"city":"B |
| Elizabeth | Reynolds | A | 647-908-8865 | 2891 | {"city":"S |
| Arthur | Gibbons | A | 416-238-7765 | 400 | {"city":"B |
| Lisa | Scott | I | 416-824-8866 | 402 | {"city":"B |
| Jean | Dexter | A | 416-774-2339 | 210 | {"city":"B |
| Roger | Vickers | A | 647-232-9987 | 234 | {"city":"S |
| Hubert | Lexter | A | 416-972-2348 | 109 | {"city":"B |
| Melanie | Ilyenko | I | 416-988-7723 | 404 | {"city":"B |
| Michael | Junielle | A | 416-209-9987 | 43 | {"city":"B |
| Rodson | Mello | A | 416-996-7654 | 220 | {"city":"B |
| Grant | Getnet | A | 416-298-3445 | 332 | {"city":"B |
| Roscoe | Banhent | A | 647-829-0223 | 338 | {"city":"S |
| Allen | Serghert | A | 416-344-0992 | 324 | {"city":"B |
| John | Merdec | I | 416-922-2331 | 55 | {"city":"B |
| Elizabeth | Ilyenko | A | 416-354-2778 | 102 | {"city":"B |
| Lisa | Junielle | A | 416-352-9992 | 323 | {"city":"B |
| Cliff | Mello | A | 416-210-9997 | 47 | {"city":"B |
| Roger | Getnet | I | 416-309-9982 | 24 | {"city":"B |
| Hubert | Banhent | A | 416-526-8888 | 19 | {"city":"B |
| Lambert | Givens | A | 416-209-8223 | 587 | {"city":"B |
| Jansen | Gibbons | A | 416-298-3881 | 895 | {"city":"B |
| Banhent | Bowden | A | 416-989-0223 | 720 | {"city":"B |
| Serghert | Dexter | A | 416-823-0991 | 830 | {"city":"B |
| Merdec | Vickers | I | 416-298-0908 | 176 | {"city":"B |
| Melcic | Lexter | A | 416-823-4443 | 128 | {"city":"B |
| Ilyenko | Merdec | A | 416-293-8771 | 97 | {"city":"B |
| Allen | Jaskobec | A | 416-299-0202 | 7 | {"city":"B |
| John | Jarkin | A | 416-622-0991 | 11 | {"city":"B |
| Franklin | Drill | A | 647-309-2331 | 37 | {"city":"S |
| Elizabeth | Metzer | A | 416-322-9001 | 85 | {"city":"B |
| Lisa | Hetzer | A | 416-980-3229 | 482 | {"city":"B |
| Cliff | Brandson | I | 416-559-0223 | 3221 | {"city":"B |
| Roger | Marjory | A | 647-290-6776 | 557 | {"city":"S |
| Hubert | Mentz | A | 416-577-8233 | 312 | {"city":"B |
34 rows selected (87.373 seconds)
Computer
bivm

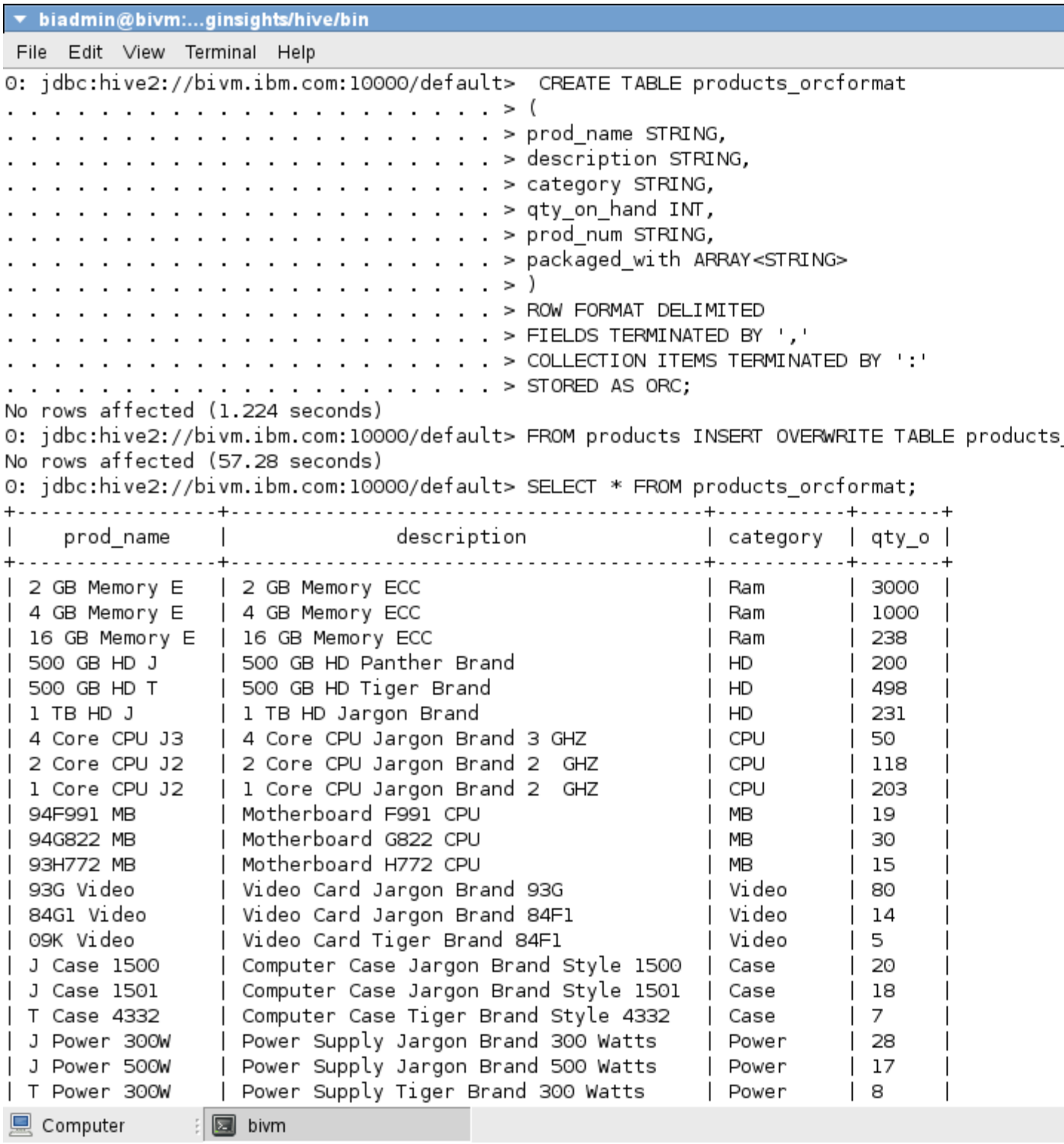

Terminal
File Edit View Terminal Help
JSqsh Release 2.1.2, Copyright (C) 2007-2016, Scott C. Gray
Type \help for available help topics. Using JLine.
[localhost][biadmin] 1> show databases;
+-----------------+
|                 |
+-----------------+
| computersalesdb |
| default         |
+-----------------+
1 row in results(first row: 1m58.87s; total: 1m58.92s)
[localhost][biadmin] 1> use computersalesdb;
ok. (total: 0.53s)
[localhost][biadmin] 1> SELECT prod_num, (SELECT count(*) FROM sales WHERE
[localhost][biadmin] 2> prod_num=prod.prod_num group by prod_num) as number_of_sales
[localhost][biadmin] 3> FROM products prod WHERE CATEGORY='Ram';
+----------+-----------------+
| prod_num | number_of_sales |
+----------+-----------------+
| 87655    |          [NULL] |
| 87659    |               1 |
| 87634    |               2 |
+----------+-----------------+
3 rows in results(first row: 13.4s; total: 13.6s)
[localhost][biadmin] 1>
[localhost][biadmin] 2> SELECT * FROM sales
[localhost][biadmin] 3> WHERE sales_date = '2012-01-24'
[localhost][biadmin] 4> ORDER BY sales_id
[localhost][biadmin] 5> ;
+---------+----------+-----+----------+------------+
| cust_id | prod_num | qty | sales_id | sales_date |
+---------+----------+-----+----------+------------+
| 7       | 98243    |   1 | 34842    | 2012-01-24 |
| 11      | 77623    |   2 | 34843    | 2012-01-24 |
| 482     | 88734    |   1 | 34844    | 2012-01-24 |
| 3221    | 282009   |   1 | 34845    | 2012-01-24 |
| 452     | 99202    |  23 | 34846    | 2012-01-24 |
| 64      | 92387    |   4 | 34847    | 2012-01-24 |
| 895     | 282009   |   7 | 34848    | 2012-01-24 |
| 1993    | 92387    |   3 | 34849    | 2012-01-24 |
| 720     | 99207    |   1 | 34850    | 2012-01-24 |
| 102     | 98243    |   1 | 34851    | 2012-01-24 |
Computer
[bivm]
[IBM Infosphere BigIns...
Terminal


Terminal
File Edit View Terminal Help
ok. (total: 0.53s)
[localhost][biadmin] 1> SELECT prod_num, (SELECT count(*) FROM sales WHERE
[localhost][biadmin] 2> prod_num=prod.prod_num group by prod_num) as number_of_sales
[localhost][biadmin] 3> FROM products prod WHERE CATEGORY='Ram';
+----------+-----------------+
| prod_num | number_of_sales |
+----------+-----------------+
| 87655 | [NULL] |
| 87659 | 1 |
| 87634 | 2 |
+----------+-----------------+
3 rows in results(first row: 13.4s; total: 13.6s)
[localhost][biadmin] 1>
[localhost][biadmin] 2> SELECT * FROM sales
[localhost][biadmin] 3> WHERE sales_date = '2012-01-24'
[localhost][biadmin] 4> ORDER BY sales_id
[localhost][biadmin] 5> ;
+---------+----------+-----+----------+------------+
| cust_id | prod_num | qty | sales_id | sales_date |
+---------+----------+-----+----------+------------+
| 7 | 98243 | 1 | 34842 | 2012-01-24 |
| 11 | 77623 | 2 | 34843 | 2012-01-24 |
| 482 | 88734 | 1 | 34844 | 2012-01-24 |
| 3221 | 282009 | 1 | 34845 | 2012-01-24 |
| 452 | 99202 | 23 | 34846 | 2012-01-24 |
| 64 | 92387 | 4 | 34847 | 2012-01-24 |
| 895 | 282009 | 7 | 34848 | 2012-01-24 |
| 1993 | 92387 | 3 | 34849 | 2012-01-24 |
| 720 | 99207 | 1 | 34850 | 2012-01-24 |
| 102 | 98243 | 1 | 34851 | 2012-01-24 |
| 227 | 77623 | 1 | 34852 | 2012-01-24 |
| 323 | 99207 | 2 | 34853 | 2012-01-24 |
| 47 | 92387 | 1 | 34854 | 2012-01-24 |
| 431 | 87659 | 1 | 34855 | 2012-01-24 |
| 24 | 45641 | 5 | 34856 | 2012-01-24 |
| 19 | 88734 | 7 | 34857 | 2012-01-24 |
| 773 | 45641 | 4 | 34858 | 2012-01-24 |
| 647 | 45628 | 2 | 34859 | 2012-01-24 |
| 773 | 45628 | 10 | 34860 | 2012-01-24 |
+---------+----------+-----+----------+------------+
19 rows in results(first row: 0.91s; total: 0.97s)
Computer
[bivm]
[IBM Infosphere BigIns...
Terminal


Terminal
File Edit View Terminal Help
| 227 | 77623 | 1 | 34852 | 2012-01-24 |
| 323 | 99207 | 2 | 34853 | 2012-01-24 |
| 47 | 92387 | 1 | 34854 | 2012-01-24 |
| 431 | 87659 | 1 | 34855 | 2012-01-24 |
| 24 | 45641 | 5 | 34856 | 2012-01-24 |
| 19 | 88734 | 7 | 34857 | 2012-01-24 |
| 773 | 45641 | 4 | 34858 | 2012-01-24 |
| 647 | 45628 | 2 | 34859 | 2012-01-24 |
| 773 | 45628 | 10 | 34860 | 2012-01-24 |
19 rows in results(first row: 0.91s; total: 0.97s)
[localhost][biadmin] 1> SELECT s.cust_id, s.prod_num, s.qty, s.sales_id,
[localhost][biadmin] 2> p.prod_name, p.category
[localhost][biadmin] 3> FROM sales s JOIN products p
[localhost][biadmin] 4> ON s.prod_num = p.prod_num
[localhost][biadmin] 5> WHERE s.sales_date = '2012-01-09' AND p.category = 'Optical';
| cust_id | prod_num | qty | sales_id | prod_name | category |
| 922 | 88734 | 1 | 34824 | DVD J INT | Optical |
| 922 | 88734 | 24 | 34828 | DVD J INT | Optical |
| 128 | 88734 | 4 | 34839 | DVD J INT | Optical |
3 rows in results(first row: 3.7s; total: 3.9s)
[localhost][biadmin] 1> CREATE VIEW optical_sales AS
[localhost][biadmin] 2> SELECT s.cust_id, s.prod_num, s.qty, s.sales_id,
[localhost][biadmin] 3> p.prod_name, p.category
[localhost][biadmin] 4> FROM sales s JOIN products p
[localhost][biadmin] 5> ON s.prod_num = p.prod_num
[localhost][biadmin] 6> WHERE p.category = 'Optical';
0 rows affected (total: 3.74s)
[localhost][biadmin] 1> SELECT * FROM optical_sales
[localhost][biadmin] 2> WHERE qty > 1;
| cust_id | prod_num | qty | sales_id | prod_name | category |
| 922 | 88734 | 24 | 34828 | DVD J INT | Optical |
| 128 | 88734 | 4 | 34839 | DVD J INT | Optical |
| 19 | 88734 | 7 | 34857 | DVD J INT | Optical |
3 rows in results(first row: 2.57s; total: 2.59s)
Computer
[bivm]
[IBM Infosphere BigIns...
Terminal


biadmin@bivm:...bm/biginsights/bin
File Edit View Terminal Help
:heatsink
1 Core CPU J2,1 Core CPU Jargon Brand 2 GHZ ,CPU,203,98792,thermalpaste
:heatsink
94F991 MB,Motherboard F991 CPU,MB,19,282299,
94G822 MB,Motherboard G822 CPU,MB,30,282109,
93H772 MB,Motherboard H772 CPU,MB,15,282009,cables:screws
93G Video,Video Card Jargon Brand 93G,Video,80,99202,dvd:manual:game
84G1 Video,Video Card Jargon Brand 84F1,Video,14,99207,dvd:manual:hdmica
ble
09K Video,Video Card Tiger Brand 84F1,Video,5,98243,manual:game
J Case 1500,Computer Case Jargon Brand Style 1500,Case,20,77623,fans:man
ual:screws
J Case 1501,Computer Case Jargon Brand Style 1501,Case,18,77624,fans:man
ual:screws
T Case 4332,Computer Case Tiger Brand Style 4332,Case,7,88211,fans:manua
l:screws:watercooler
J Power 300W,Power Supply Jargon Brand 300 Watts,Power,28,92387,cables:s
crews
J Power 500W,Power Supply Jargon Brand 500 Watts,Power,17,92373,cables:s
crews
T Power 300W,Power Supply Tiger Brand 300 Watts,Power,8,93347,cables:scr
ews
DVD J INT,DVD Jargon Brand Internal,Optical,23,88734,manual
DVD J EXT,DVD Jargon Brand External,Optical,45,88821,
DVD T INT,DVD Tiger Brand Internal,Optical,19,82331,satacable:manual
DVD T EXT,DVD Tiger Brand External,Optical,17,82337,satacable:manual
biadmin@bivm:/opt/ibm/biginsights/bin> hadoop fs -cat /biginsights/hive/warehouse/computer
RCF org.apache.hadoop.io.compress.DefaultCodec ive.io.rcfile.column.number F Y
Computer
[Java - Task Launcher ...
[bivm]
[IBM Infosphere BigIns...


biadmin@bivm:...bm/biginsights/bin
File Edit View Terminal Help
biadmin@bivm:~> cd $BIGINSIGHTS_HOME/bin
biadmin@bivm:/opt/ibm/biginsights/bin> hadoop fs -text /biginsights/hive/warehou
se/computersalesdb.db/products_sequenceformat/000000_0;
2 GB Memory E,2 GB Memory ECC,Ram,3000,87655,manual:heatsink
4 GB Memory E,4 GB Memory ECC,Ram,1000,87659,manual:heatsink
16 GB Memory E,16 GB Memory ECC,Ram,238,87634,manual
500 GB HD J,500 GB HD Panther Brand,HD,200,45628,atacable:manual
500 GB HD T,500 GB HD Tiger Brand,HD,498,45641,satacable:manual
1 TB HD J,1 TB HD Jargon Brand,HD,231,45691,
4 Core CPU J3,4 Core CPU Jargon Brand 3 GHZ ,CPU,50,98820,thermalpaste:h
eatsink:manual
2 Core CPU J2,2 Core CPU Jargon Brand 2 GHZ ,CPU,118,98838,thermalpaste
:heatsink
1 Core CPU J2,1 Core CPU Jargon Brand 2 GHZ ,CPU,203,98792,thermalpaste
:heatsink
94F991 MB,Motherboard F991 CPU,MB,19,282299,
94G822 MB,Motherboard G822 CPU,MB,30,282109,
93H772 MB,Motherboard H772 CPU,MB,15,282009,cables:screws
93G Video,Video Card Jargon Brand 93G,Video,80,99202,dvd:manual:game
84G1 Video,Video Card Jargon Brand 84F1,Video,14,99207,dvd:manual:hdmica
ble
09K Video,Video Card Tiger Brand 84F1,Video,5,98243,manual:game
J Case 1500,Computer Case Jargon Brand Style 1500,Case,20,77623,fans:man
ual:screws
J Case 1501,Computer Case Jargon Brand Style 1501,Case,18,77624,fans:man
ual:screws
T Case 4332,Computer Case Tiger Brand Style 4332,Case,7,88211,fans:manua
l:screws:watercooler
J Power 300W,Power Supply Jargon Brand 300 Watts,Power,28,92387,cables:s
crews
J Power 500W,Power Supply Jargon Brand 500 Watts,Power,17,92373,cables:s
crews
T Power 300W,Power Supply Tiger Brand 300 Watts,Power,8,93347,cables:scr
ews
DVD J INT,DVD Jargon Brand Internal,Optical,23,88734,manual
DVD J EXT,DVD Jargon Brand External,Optical,45,88821,
DVD T INT,DVD Tiger Brand Internal,Optical,19,82331,satacable:manual
DVD T EXT,DVD Tiger Brand External,Optical,17,82337,satacable:manual
biadmin@bivm:/opt/ibm/biginsights/bin>
Computer
[Java - Task Launcher ...
[bivm]
[IBM Infosphere BigIns...


biadmin@bivm:...ginsights/hive/bin
File Edit View Terminal Help
0: jdbc:hive2://bivm.ibm.com:10000/default> CREATE TABLE products_orcformat
. . . . . > (
. . . . . > prod_name STRING,
. . . . . > description STRING,
. . . . . > category STRING,
. . . . . > qty_on_hand INT,
. . . . . > prod_num STRING,
. . . . . > packaged_with ARRAY<STRING>
. . . . . > )
. . . . . > ROW FORMAT DELIMITED
. . . . . > FIELDS TERMINATED BY ','
. . . . . > COLLECTION ITEMS TERMINATED BY ':'
. . . . . > STORED AS ORC;
No rows affected (1.224 seconds)
0: jdbc:hive2://bivm.ibm.com:10000/default> FROM products INSERT OVERWRITE TABLE products_
No rows affected (57.28 seconds)
0: jdbc:hive2://bivm.ibm.com:10000/default> SELECT * FROM products_orcformat;
| prod_name | description | category | qty_o |
| 2 GB Memory E | 2 GB Memory ECC | Ram | 3000 |
| 4 GB Memory E | 4 GB Memory ECC | Ram | 1000 |
| 16 GB Memory E | 16 GB Memory ECC | Ram | 238 |
| 500 GB HD J | 500 GB HD Panther Brand | HD | 200 |
| 500 GB HD T | 500 GB HD Tiger Brand | HD | 498 |
| 1 TB HD J | 1 TB HD Jargon Brand | HD | 231 |
| 4 Core CPU J3 | 4 Core CPU Jargon Brand 3 GHZ | CPU | 50 |
| 2 Core CPU J2 | 2 Core CPU Jargon Brand 2 GHZ | CPU | 118 |
| 1 Core CPU J2 | 1 Core CPU Jargon Brand 2 GHZ | CPU | 203 |
| 94F991 MB | Motherboard F991 CPU | MB | 19 |
| 94G822 MB | Motherboard G822 CPU | MB | 30 |
| 93H772 MB | Motherboard H772 CPU | MB | 15 |
| 93G Video | Video Card Jargon Brand 93G | Video | 80 |
| 84G1 Video | Video Card Jargon Brand 84F1 | Video | 14 |
| 09K Video | Video Card Tiger Brand 84F1 | Video | 5 |
| J Case 1500 | Computer Case Jargon Brand Style 1500 | Case | 20 |
| J Case 1501 | Computer Case Jargon Brand Style 1501 | Case | 18 |
| T Case 4332 | Computer Case Tiger Brand Style 4332 | Case | 7 |
| J Power 300W | Power Supply Jargon Brand 300 Watts | Power | 28 |
| J Power 500W | Power Supply Jargon Brand 500 Watts | Power | 17 |
| T Power 300W | Power Supply Tiger Brand 300 Watts | Power | 8 |
Computer
bivm

# PIG

Pig was initially developed at Yahoo! to allow people using Apache Hadoop® to focus more on analyzing large data sets and spend less time having to write mapper and reducer programs. Like actual pigs, who eat almost anything, the Pig programming language is designed to handle any kind of data—hence the name!

Pig is made up of two components: the first is the language itself, which is called PigLatin (yes, people naming various Hadoop projects do tend to have a sense of humor associated with their naming conventions), and the second is a runtime environment where PigLatin programs are executed. Think of the relationship between a Java Virtual Machine (JVM) and a Java application. In this section, we'll just refer to the whole entity as Pig.

## The programming language

Let's first look at the programming language itself so you can see how it's significantly easier than having to write mapper and reducer programs.

1. The first step in a Pig program is to LOAD the data you want to manipulate from HDFS.

2. Then you run the data through a set of transformations (which, under the covers, are translated into a set of mapper and reducer tasks).

3. Finally, you DUMP the data to the screen or you STORE the results in a file somewhere.

### LOAD

As is the case with all the Hadoop features, the objects that are being worked on by Hadoop are stored in HDFS. In order for a Pig program to access this data, the program must first tell Pig what file (or files) it will use, and that's done through the LOAD 'data_file' command (where 'data_file' specifies either an HDFS file or directory). If a directory is specified, all the files in that directory will be loaded into the program. If the data is stored in a file format that is not natively accessible to Pig, you can optionally add the USING function to the LOAD statement to specify a user-defined function that can read in and interpret the data.

### TRANSFORM

The transformation logic is where all the data manipulation happens. Here you can FILTER out rows that are not of interest, JOIN two sets of data files, GROUP data to build aggregations, ORDER results, and much more.

### DUMP and STORE

If you don’t specify the DUMP or STORE command, the results of a Pig pro¬gram are not generated. You would typically use the DUMP command, which sends the output to the screen, when you are debugging your Pig programs. When you go into production, you simply change the DUMP call to a STORE call so that any results from running your programs are stored in a file for further processing or analysis. Note that you can use the DUMP command anywhere in your program to dump intermediate result sets to the screen, which is very useful for debugging purposes.

## Advantages Of PIG

- Decrease in development time. This is the biggest advantage especially considering vanilla map-reduce jobs' complexity, time-spent and maintenance of the programs.
- Learning curve is not steep, anyone who does not know how to write vanilla map-reduce or SQL for that matter could pick up and can write map-reduce jobs; not easy to master, though.
- Procedural, not declarative unlike SQL, so easier to follow the commands and provides better expressiveness in the transformation of data every step. Comparing to vanilla map-reduce, it is much more like an english language. It is concise and unlike Java but more like Python.
- I really liked the idea of dataflow where everything is about data even though we sacrifice control structures like for loop or if structures. This enforces the developer to think about the data but nothing else. In Python or Java, you create the control structures(for loop and ifs) and get the data transformation as a side effect. In here, data and because of data, data transformation is a first class citizen. Without data, you

cannot create for loops, you need to always transform and manipulate data. But if you are not transforming data, what are you doing in the very first place?

- Since it is procedural, you could control of the execution of every step. If you want to write your own UDF(User Defined Function) and inject in one specific part in the pipeline, it is straightforward.

## Disadvantages Of PIG

- Especially the errors that Pig produces due to UDFS(Python) are not helpful at all. When something goes wrong, it just gives exec error in udf even if problem is related to syntax or type error, let alone a logical one. This is a big one. At least, as a user, I should get different error messages when I have a syntax error, type error or a runtime error.
- Not mature. Even if it has been around for quite some time, it is still in the development. (only recently they introduced a native datetime structure which is quite fundamental for a language like Pig especially considering how an important component of datetime for time-series data.
- Support: Stackoverflow and Google generally does not lead good solutions for the problems.
- Data Schema is not enforced explicitly but implicitly. I think this is big one, too. The debugging of pig scripts in my experience is %90 of time schema and since it does not enforce an explicit schema, sometimes one data structure goes bytearray, which is a “raw” data type and unless you coerce the fields even the strings, they turn bytearray without notice. This may propagate for other steps of the data processing.
- Minor one: There is not a good ide or plugin for Vim which provides more functionality than syntax completion to write the pig scripts.
- The commands are not executed unless either you dump or store an intermediate or final result. This increases the iteration between debug and resolving the issue.

- Hive and Pig are not the same thing and the things that Pig does quite well Hive may not and vice versa. However, someone who knows SQL could write Hive queries(most of SQL queries do already work in Hive) where she cannot do that in Pig. She needs to learn Pig syntax.

Pig basically has two execution modes

1] Local Mode

2] Map-Reduce Mode

Mapreduce mode is the default mode;

Local Mode - When you run Pig in local mode, you need access to a single machine; all files are installed and run using your local host and local file system.

Here *all files* means all the files which you are going to process and all the jars or anything which you are referring/using in pig Script.

Mapreduce Mode - When you run Pig in mapreduce mode, you are dealing with Hadoop cluster and HDFS(Hadoop Distributed File System).

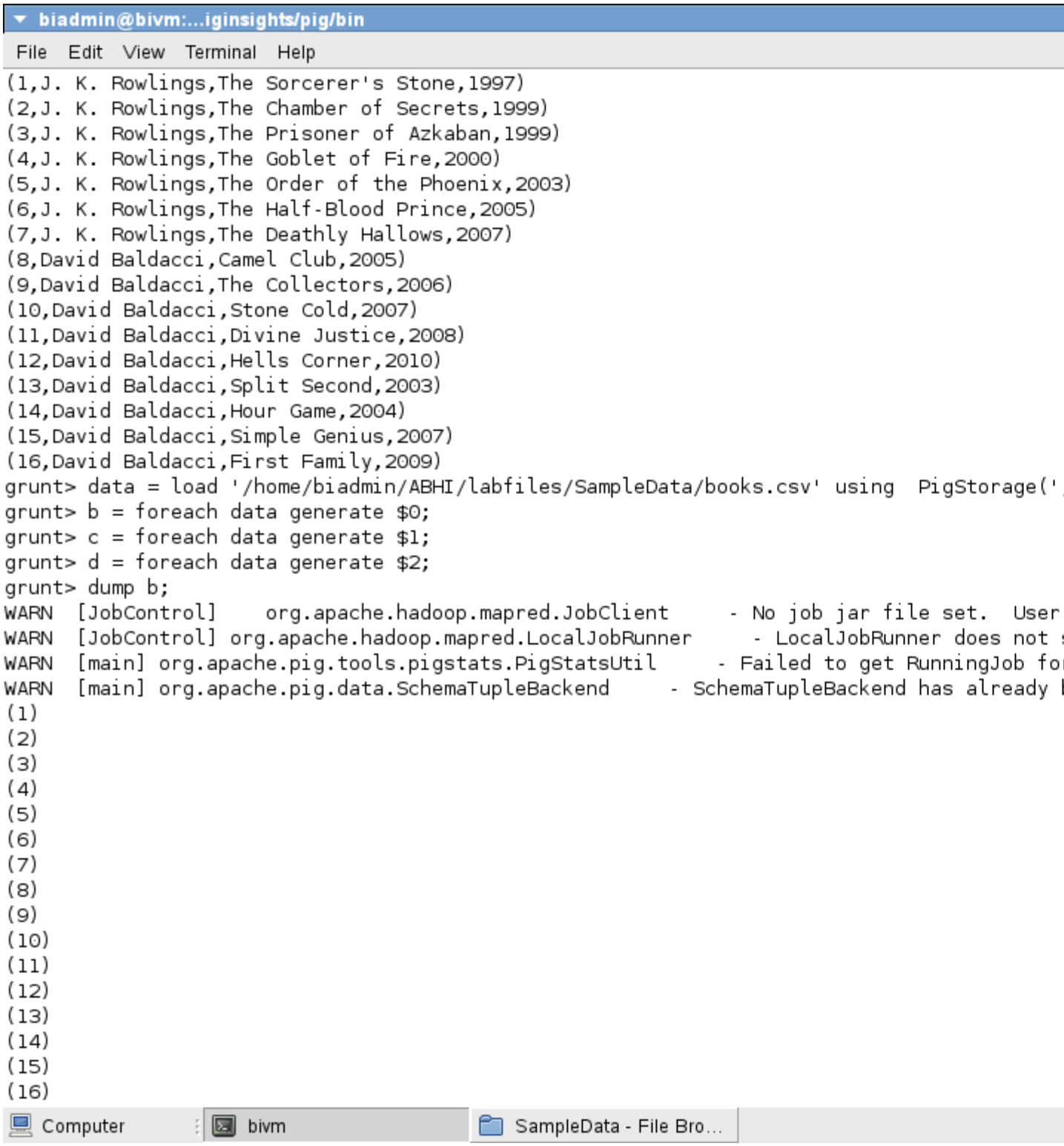
pig -x local '/YOUR_PATH_TO_PIG_SCRIPT/script.pig'

# PIG ANALYSIS

## Commands to start pig in local mode:

### 1-cd $PIG_HOME/bin

### 2-./pig -x local


biadmin@bivm:...iginsights/pig/bin
File Edit View Terminal Help
(1,J. K. Rowlings,The Sorcerer's Stone,1997)
(2,J. K. Rowlings,The Chamber of Secrets,1999)
(3,J. K. Rowlings,The Prisoner of Azkaban,1999)
(4,J. K. Rowlings,The Goblet of Fire,2000)
(5,J. K. Rowlings,The Order of the Phoenix,2003)
(6,J. K. Rowlings,The Half-Blood Prince,2005)
(7,J. K. Rowlings,The Deathly Hallows,2007)
(8,David Baldacci,Camel Club,2005)
(9,David Baldacci,The Collectors,2006)
(10,David Baldacci,Stone Cold,2007)
(11,David Baldacci,Divine Justice,2008)
(12,David Baldacci,Hells Corner,2010)
(13,David Baldacci,Split Second,2003)
(14,David Baldacci,Hour Game,2004)
(15,David Baldacci,Simple Genius,2007)
(16,David Baldacci,First Family,2009)
grunt> data = load '/home/biadmin/ABHI/labfiles/SampleData/books.csv' using PigStorage('
grunt> b = foreach data generate $0;
grunt> c = foreach data generate $1;
grunt> d = foreach data generate $2;
grunt> dump b;
WARN [JobControl] org.apache.hadoop.mapred.JobClient - No job jar file set. User
WARN [JobControl] org.apache.hadoop.mapred.LocalJobRunner - LocalJobRunner does not
WARN [main] org.apache.pig.tools.pigstats.PigStatsUtil - Failed to get RunningJob fo
WARN [main] org.apache.pig.data.SchemaTupleBackend - SchemaTupleBackend has already
(1)
(2)
(3)
(4)
(5)
(6)
(7)
(8)
(9)
(10)
(11)
(12)
(13)
(14)
(15)
(16)
Computer
bivm
SampleData - File Bro...


biadmin@bivm:...iginsights/pig/bin
File Edit View Terminal Help
WARN [JobControl] org.apache.hadoop.mapred.JobClient - No job jar file set. User
WARN [JobControl] org.apache.hadoop.mapred.LocalJobRunner - LocalJobRunner does not
WARN [main] org.apache.pig.tools.pigstats.PigStatsUtil - Failed to get RunningJob fo
WARN [main] org.apache.pig.data.SchemaTupleBackend - SchemaTupleBackend has already
(J. K. Rowlings)
(J. K. Rowlings)
(J. K. Rowlings)
(J. K. Rowlings)
(J. K. Rowlings)
(J. K. Rowlings)
(J. K. Rowlings)
(David Baldacci)
(David Baldacci)
(David Baldacci)
(David Baldacci)
(David Baldacci)
(David Baldacci)
(David Baldacci)
(David Baldacci)
(David Baldacci)
(David Baldacci)
grunt> dump d;
WARN [JobControl] org.apache.hadoop.mapred.JobClient - No job jar file set. User
WARN [JobControl] org.apache.hadoop.mapred.LocalJobRunner - LocalJobRunner does not
WARN [main] org.apache.pig.tools.pigstats.PigStatsUtil - Failed to get RunningJob fo
WARN [main] org.apache.pig.data.SchemaTupleBackend - SchemaTupleBackend has already
(The Sorcerer's Stone)
(The Chamber of Secrets)
(The Prisoner of Azkaban)
(The Goblet of Fire)
(The Order of the Phoenix)
(The Half-Blood Prince)
(The Deathly Hallows)
(Camel Club)
(The Collectors)
(Stone Cold)
(Divine Justice)
(Hells Corner)
(Split Second)
(Hour Game)
(Simple Genius)
(First Family)
Computer
bivm
SampleData - File Bro...


biadmin@bivm:...iginsights/pig/bin
File Edit View Terminal Help
at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:94)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.ja
at java.lang.reflect.Method.invoke(Method.java:619)
at org.apache.pig.backend.hadoop23.PigJobControl.submit(PigJobControl.java:128)
at org.apache.pig.backend.hadoop23.PigJobControl.run(PigJobControl.java:191)
at java.lang.Thread.run(Thread.java:853)
at org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher
Caused by: org.apache.hadoop.mapreduce.lib.input.InvalidInputException: Input path does no
at org.apache.hadoop.mapreduce.lib.input.FileInputFormat.listStatus(FileInputForma
at org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.PigTextInputFormat
at org.apache.hadoop.mapreduce.lib.input.FileInputFormat.getSplits(FileInputFormat
at org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.PigInputFormat.get
... 19 more
ERROR [main] org.apache.pig.tools.pigstats.PigStatsUtil - 1 map reduce job(s) failed!
ERROR [main] org.apache.pig.tools.grunt.Grunt - ERROR 1066: Unable to open iterat
Details at logfile: /opt/ibm/biginsights/pig/bin/pig_1468832230434.log
grunt> a = load '/home/biadmin/ABHI/labfiles/SampleData/books.csv' using PigStorage(',') a
grunt> b = foreach a generate author,title;
grunt> dump b;
WARN [JobControl] org.apache.hadoop.mapred.JobClient - No job jar file set. User
WARN [JobControl] org.apache.hadoop.mapred.LocalJobRunner - LocalJobRunner does not s
WARN [main] org.apache.pig.tools.pigstats.PigStatsUtil - Failed to get RunningJob for
WARN [main] org.apache.pig.data.SchemaTupleBackend - SchemaTupleBackend has already b
(J. K. Rowlings,The Sorcerer's Stone)
(J. K. Rowlings,The Chamber of Secrets)
(J. K. Rowlings,The Prisoner of Azkaban)
(J. K. Rowlings,The Goblet of Fire)
(J. K. Rowlings,The Order of the Phoenix)
(J. K. Rowlings,The Half-Blood Prince)
(J. K. Rowlings,The Deathly Hallows)
(David Baldacci,Camel Club)
(David Baldacci,The Collectors)
(David Baldacci,Stone Cold)
(David Baldacci,Divine Justice)
(David Baldacci,Hells Corner)
(David Baldacci,Split Second)
(David Baldacci,Hour Game)
(David Baldacci,Simple Genius)
(David Baldacci,First Family)
grunt>
Computer
bivm
SampleData - File Bro...