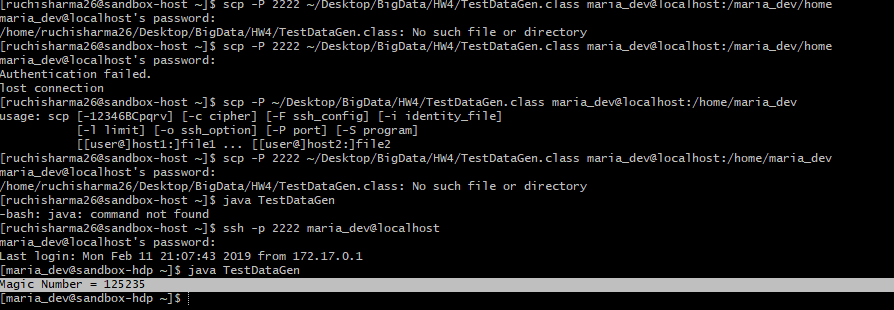
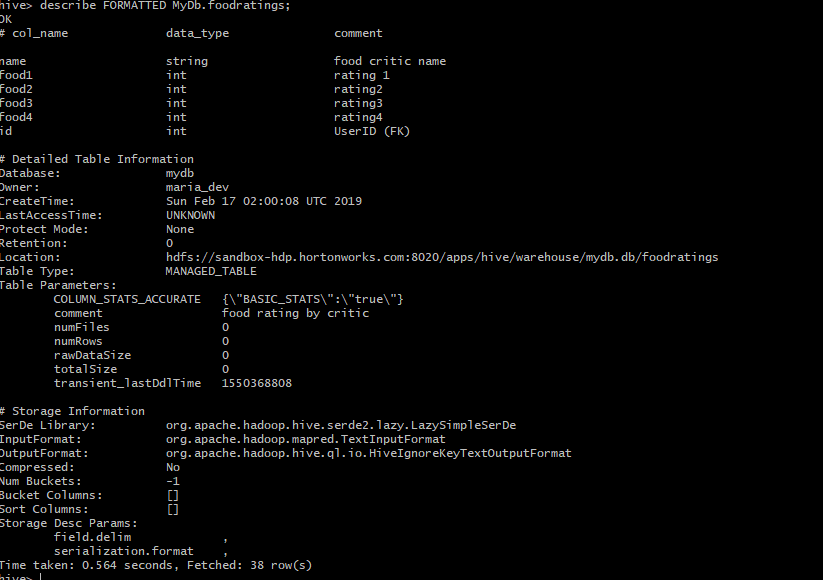
Ruchi Sharma A20429225

**Magic Number: 125235**

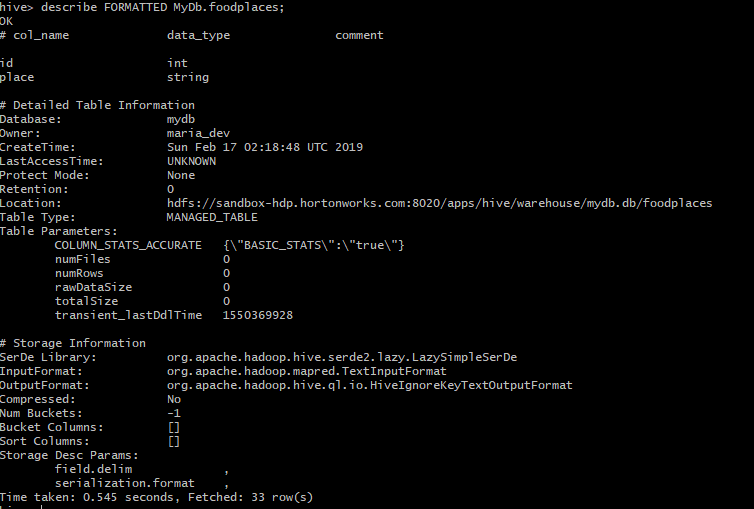


**Exercise 1)**

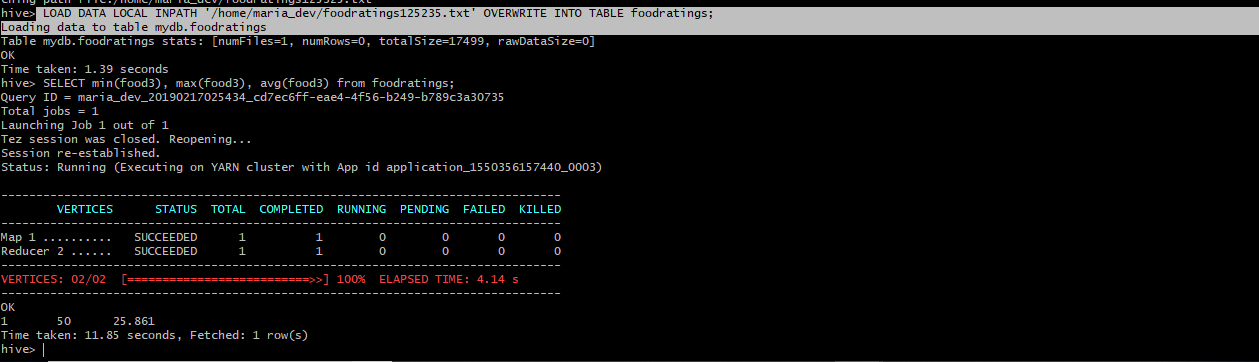
1. **Foodratings**



1. **Foodpalces**



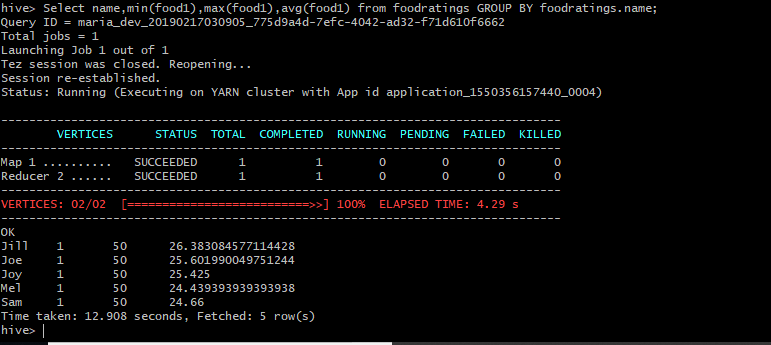
**Exercise 2) (Magic Number: 125235)**



**Command:** LOAD DATA LOCAL INPATH '/home/maria\_dev/foodratings125235.txt' OVERWRITE INTO TABLE foodratings;

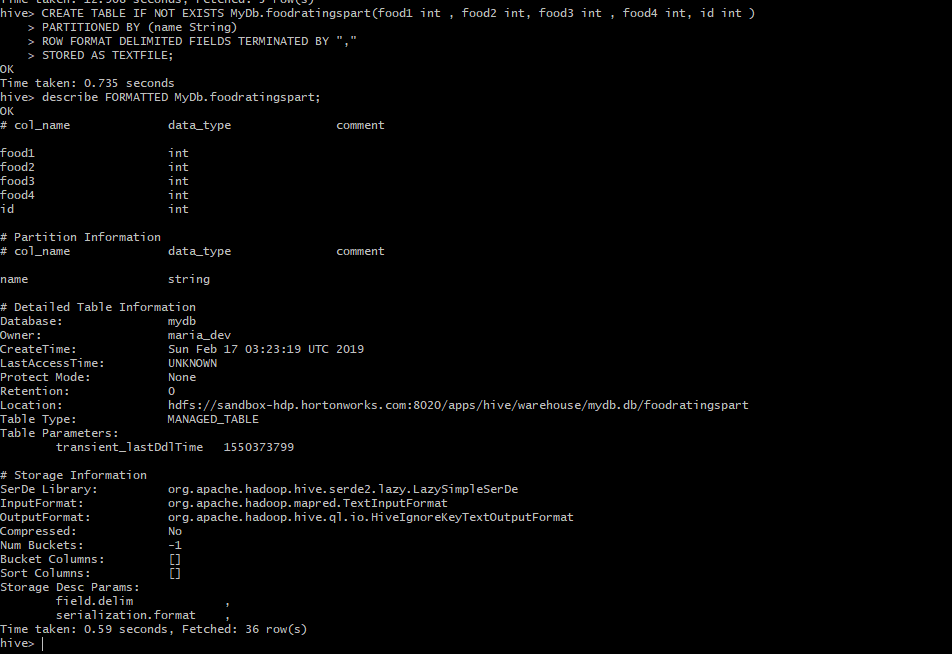
**Hive Command:** SELECT min(food3), max(food3), avg(food3) from foodratings;

**Exercise 3) (Magic Number: 125235)**



**Command:** Select name, min(food1), max(food1), avg(food1) from foodratings GROUP BY foodratings.name;

**Exercise 4) (Magic Number: 125235)**



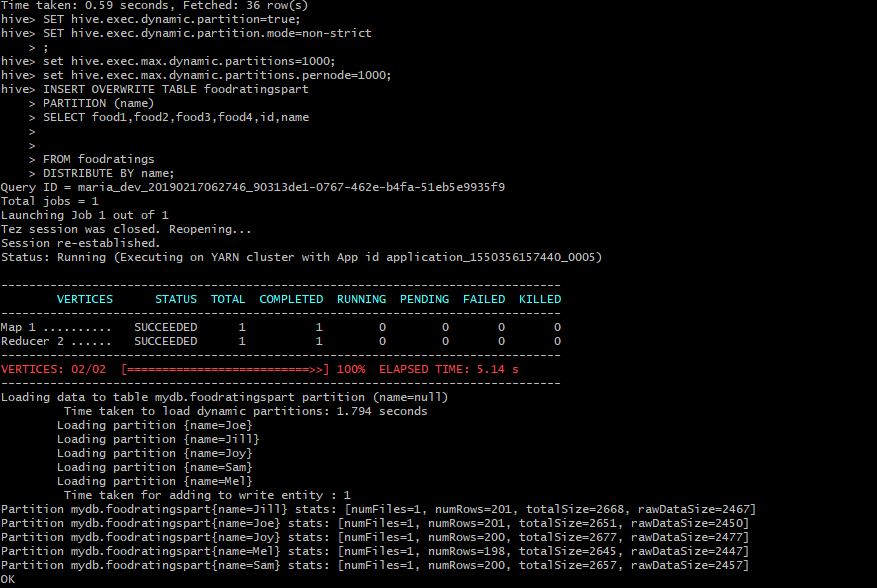
CREATE TABLE IF NOT EXISTS MyDb.foodratingspart (food1 int, food2 int, food3 int, food4 int, id int)

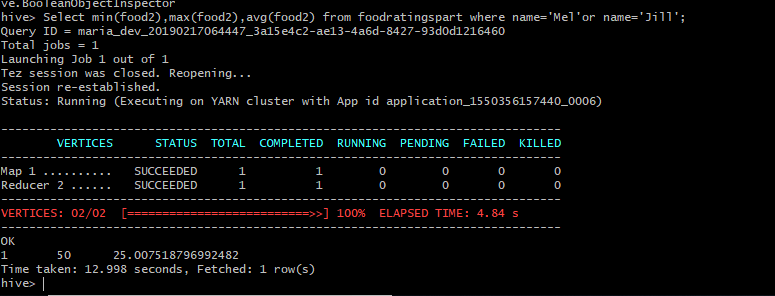
PARTITIONED BY (name String)

ROW FORMAT DELIMITED FIELDSTERMINATED BY “,”

STORED AS TEXTFILE;

**Exercise 5) Configuration:**





**Setting Dynamic Partitioning:**

SET hive.exec.dynamic.partition=true;

SET hive.exec.dynamic.partition.mode=non-strict

set hive.exec.max.dynamic.partitions=1000;

set hive.exec.max.dynamic.partitions.pernode=1000;

**(As provided configuration were not sufficient at my end, I browsed and found these two commands and kept it as configuration command)**

**Load Data from Non-Partitioned to Partitioned:**

INSERT OVERWRITE TABLE foodratingspart

PARTITION (name)

SELECT food1, food2, food3,food4,id,name

FROM foodratings;

Hive Command:

Select min(food2), max(food2), avg(food2) from foodratingspart where name='Mel' or name='Jill';

**Exercise 6)**

**(Magic Number: 125235)**

LOAD DATA LOCAL INPATH '/home/maria\_dev/foodplaces125235.txt' OVERWRITE INTO TABLE

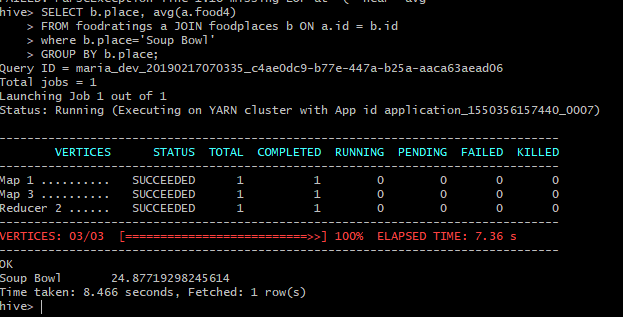
foodplaces;

SELECT b.place avg(a.food4)

FROM foodratings a JOIN foodplaces b ON a.id = b.id

where b.place='Soup Bowl'

GROUP BY b.place;



**Exercise 7)**

The article “Pig latin: a not-so-foreign language for data processing” describes a new language called Pig Latin. It aimed to provide a middle way between the declaration SQL style language (which many developers find unnatural) and the procedural mapping model (very low level and hard). It also offers anew, interactive debug environment called Pig Pen that can lead to even higher productivity gains.

Pig Latin allows optimization, by reordering the code whereas this is not possible in the opaque Map or Reduce function. Provides Quick start as Pig Latin can work directly on data with a proper function to parse over the content into tuples which avoids time consuming data imports. Allows fully ‘Nested Data Model’ and allows complex, non-atomic data types such as set, map, and tuple to occur as fields of a table. Supports custom processing through UDF (User Defined Functions).

The article also describes a novel debugging environment for Pig, called Pig Pen. In conjunction with the step-by-step nature of our Pig Latin language, Pig Pen makes it easy and fast for users to construct and debug their programs in an incremental fashion.

They also offer a way where we could freeze the progression of program after testing and move further without worrying about the previous freeze code. The Pig system compiles Pig Latin expressions into a sequence of map-reduce jobs, and orchestrates the execution of these jobs on Hadoop, an open-source scalable map-reduce implementation.

This section which is based on conciseness, completeness and realism. In the 6th section, the author, on a high level, talks about the use of Pig Latin from group-by-aggregate and rollup queries to more complex tasks like temporal and session analysis. Section 7 compares the Pig Latin with other technologies at Google, Amazon, etc

**Pros:** Pig Latin is more natural looking, and reusable compared to SQL. It can operate over plain input files without schema information and costly data import operations. Its nested data model allows complex, non-atomic data types. It comes with an interactive debugging environment. It can work on multiple data sets and can efficiently generate aggregated results. Pig is not tied to Hadoop only and architected to work with other execution platforms. Pig is open source and available for general use.

**Cons:** Due to its nested data model Pig may generate huge amount of intermediate data. As Pig currently relies on map-reduce jobs, intermediate data need to be generated, transferred and stored in the distributed file system multiple times affecting system throughput and latency.