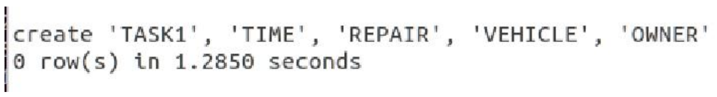
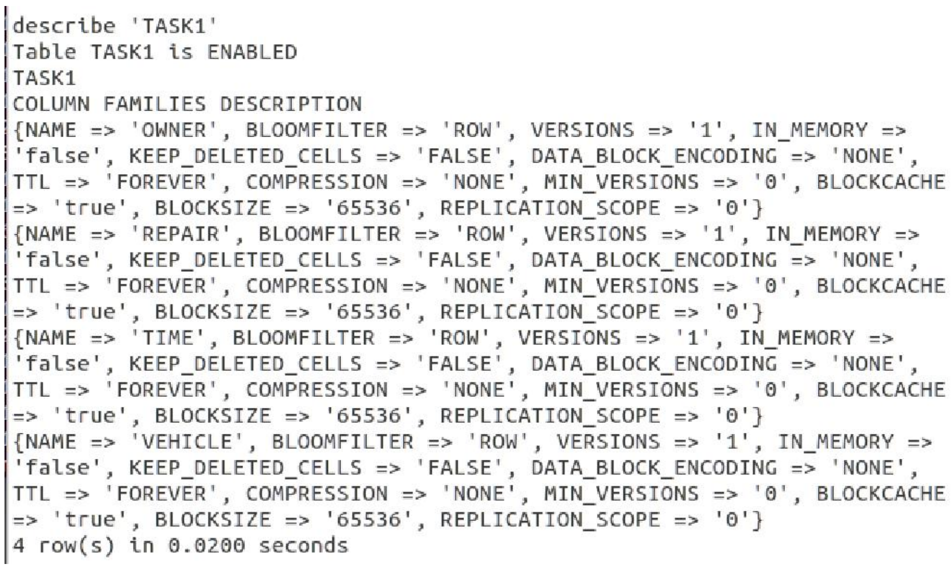
// ISIT312 Assignment3
// Student Name: Huaiying Shao
// UOW ID: 7910356

Task1
Create HBASE table:



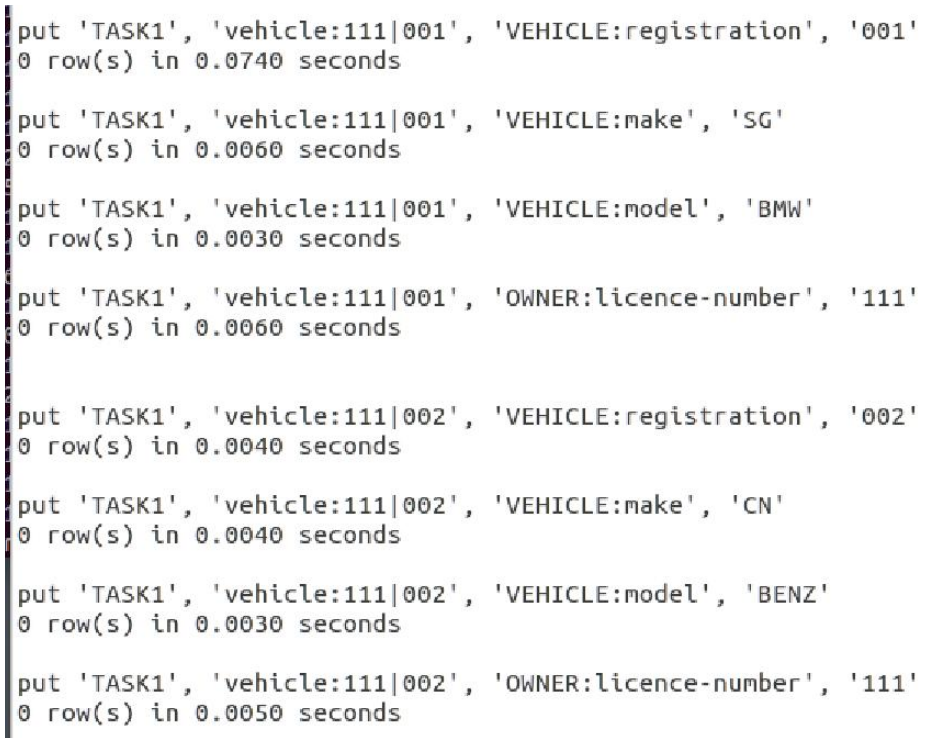
```
create 'TASK1', 'TIME', 'REPAIR', 'VEHICLE', 'OWNER'
0 row(s) in 1.2850 seconds
```

Describe the table schema:



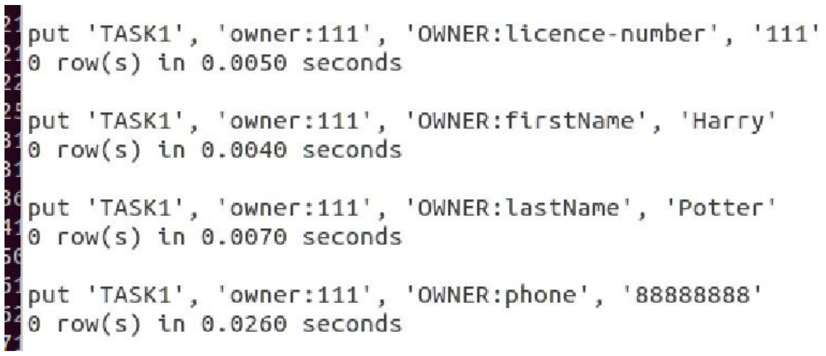
```
describe 'TASK1'
Table TASK1 is ENABLED
TASK1
COLUMN FAMILIES DESCRIPTION
{NAME => 'OWNER', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY =>
'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE',
TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS => '0', BLOCKCACHE
=> 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}
{NAME => 'REPAIR', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY =>
'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE',
TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS => '0', BLOCKCACHE
=> 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}
{NAME => 'TIME', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY =>
'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE',
TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS => '0', BLOCKCACHE
=> 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}
{NAME => 'VEHICLE', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY =>
'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE',
TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS => '0', BLOCKCACHE
=> 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}
4 row(s) in 0.0200 seconds
```

Load sample data into table, put two vehicle into 'TASK1':



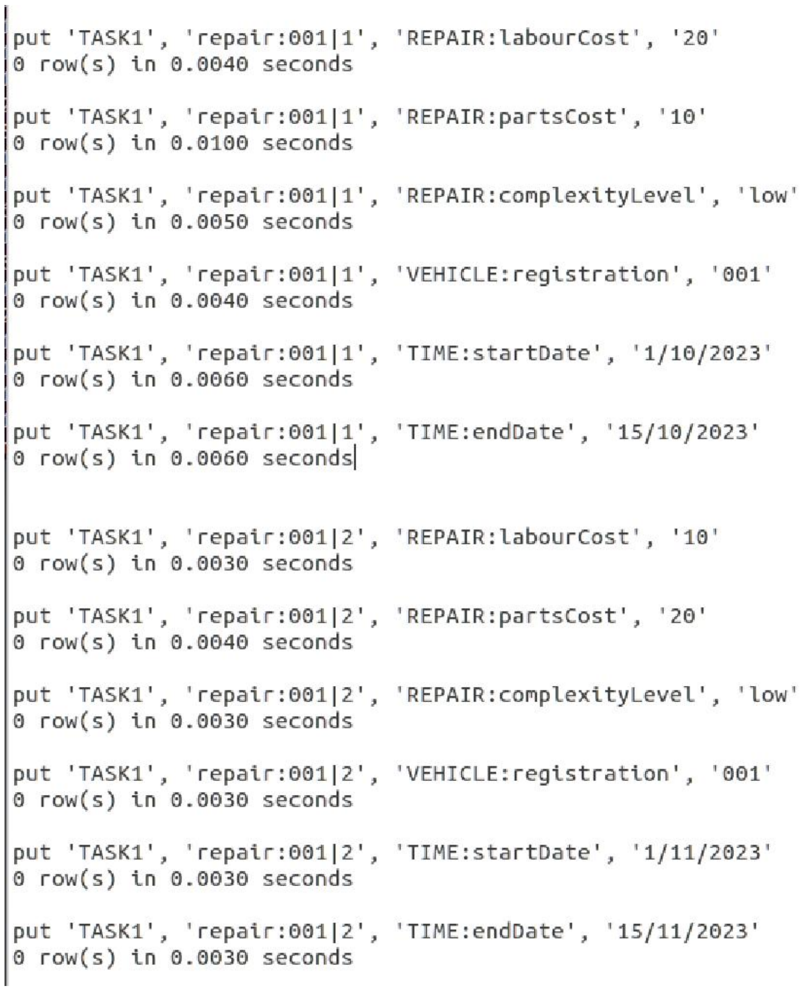
```
put 'TASK1', 'vehicle:111|001', 'VEHICLE:registration', '001'
0 row(s) in 0.0740 seconds

put 'TASK1', 'vehicle:111|001', 'VEHICLE:make', 'SG'
0 row(s) in 0.0060 seconds

put 'TASK1', 'vehicle:111|001', 'VEHICLE:model', 'BMW'
0 row(s) in 0.0030 seconds

put 'TASK1', 'vehicle:111|001', 'OWNER:licence-number', '111'
0 row(s) in 0.0060 seconds


put 'TASK1', 'vehicle:111|002', 'VEHICLE:registration', '002'
0 row(s) in 0.0040 seconds

put 'TASK1', 'vehicle:111|002', 'VEHICLE:make', 'CN'
0 row(s) in 0.0040 seconds

put 'TASK1', 'vehicle:111|002', 'VEHICLE:model', 'BENZ'
0 row(s) in 0.0030 seconds

put 'TASK1', 'vehicle:111|002', 'OWNER:licence-number', '111'
0 row(s) in 0.0050 seconds
```

Put one owner into ‘TASK1’, the owner owns two vehicles:



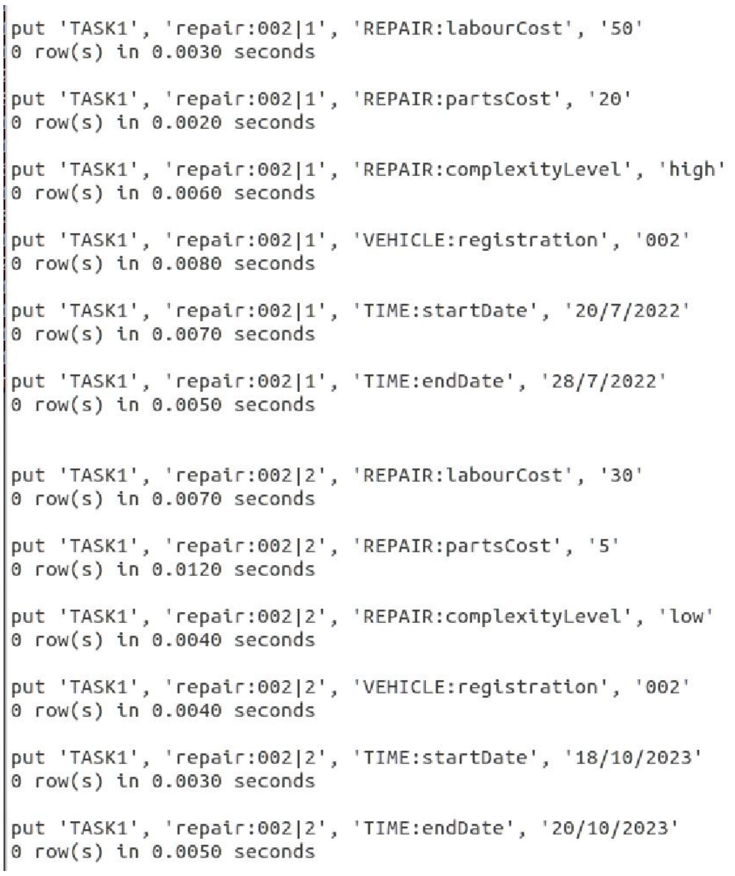
```
put 'TASK1', 'owner:111', 'OWNER:licence-number', '111'
0 row(s) in 0.0050 seconds

put 'TASK1', 'owner:111', 'OWNER:firstName', 'Harry'
0 row(s) in 0.0040 seconds

put 'TASK1', 'owner:111', 'OWNER:lastName', 'Potter'
0 row(s) in 0.0070 seconds

put 'TASK1', 'owner:111', 'OWNER:phone', '88888888'
0 row(s) in 0.0260 seconds
```

For each vehicle has two repair, insert two repair records for vehicle 001:



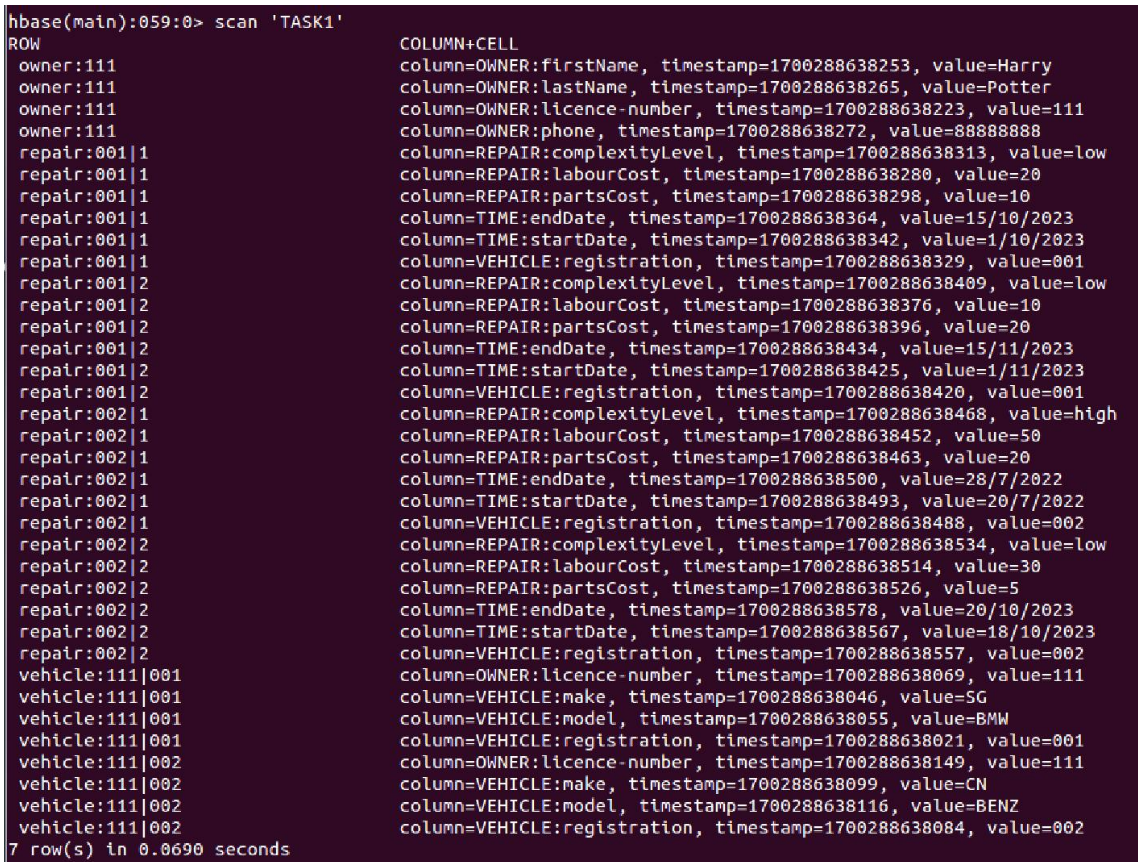
```
put 'TASK1', 'repair:001|1', 'REPAIR:labourCost', '20'
0 row(s) in 0.0040 seconds

put 'TASK1', 'repair:001|1', 'REPAIR:partsCost', '10'
0 row(s) in 0.0100 seconds

put 'TASK1', 'repair:001|1', 'REPAIR:complexityLevel', 'low'
0 row(s) in 0.0050 seconds

put 'TASK1', 'repair:001|1', 'VEHICLE:registration', '001'
0 row(s) in 0.0040 seconds

put 'TASK1', 'repair:001|1', 'TIME:startDate', '1/10/2023'
0 row(s) in 0.0060 seconds

put 'TASK1', 'repair:001|1', 'TIME:endDate', '15/10/2023'
0 row(s) in 0.0060 seconds


put 'TASK1', 'repair:001|2', 'REPAIR:labourCost', '10'
0 row(s) in 0.0030 seconds

put 'TASK1', 'repair:001|2', 'REPAIR:partsCost', '20'
0 row(s) in 0.0040 seconds

put 'TASK1', 'repair:001|2', 'REPAIR:complexityLevel', 'low'
0 row(s) in 0.0030 seconds

put 'TASK1', 'repair:001|2', 'VEHICLE:registration', '001'
0 row(s) in 0.0030 seconds

put 'TASK1', 'repair:001|2', 'TIME:startDate', '1/11/2023'
0 row(s) in 0.0030 seconds

put 'TASK1', 'repair:001|2', 'TIME:endDate', '15/11/2023'
0 row(s) in 0.0030 seconds
```

Insert two repair records for vehicle 002:



```
put 'TASK1', 'repair:002|1', 'REPAIR:labourCost', '50'
0 row(s) in 0.0030 seconds

put 'TASK1', 'repair:002|1', 'REPAIR:partsCost', '20'
0 row(s) in 0.0020 seconds

put 'TASK1', 'repair:002|1', 'REPAIR:complexityLevel', 'high'
0 row(s) in 0.0060 seconds

put 'TASK1', 'repair:002|1', 'VEHICLE:registration', '002'
0 row(s) in 0.0080 seconds

put 'TASK1', 'repair:002|1', 'TIME:startDate', '20/7/2022'
0 row(s) in 0.0070 seconds

put 'TASK1', 'repair:002|1', 'TIME:endDate', '28/7/2022'
0 row(s) in 0.0050 seconds


put 'TASK1', 'repair:002|2', 'REPAIR:labourCost', '30'
0 row(s) in 0.0070 seconds

put 'TASK1', 'repair:002|2', 'REPAIR:partsCost', '5'
0 row(s) in 0.0120 seconds

put 'TASK1', 'repair:002|2', 'REPAIR:complexityLevel', 'low'
0 row(s) in 0.0040 seconds

put 'TASK1', 'repair:002|2', 'VEHICLE:registration', '002'
0 row(s) in 0.0040 seconds

put 'TASK1', 'repair:002|2', 'TIME:startDate', '18/10/2023'
0 row(s) in 0.0030 seconds

put 'TASK1', 'repair:002|2', 'TIME:endDate', '20/10/2023'
0 row(s) in 0.0050 seconds
```

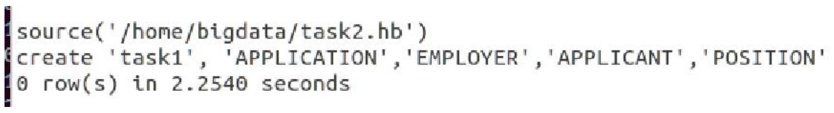
To scan table output:



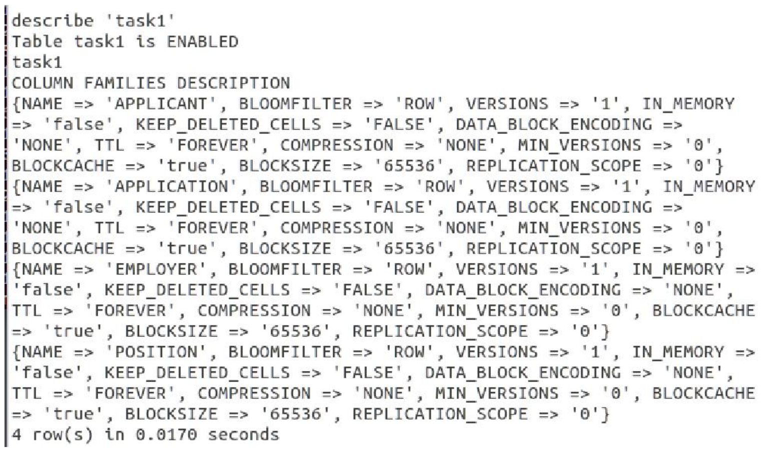
```
hbase(main):059:0> scan 'TASK1'
ROW                                 COLUMN+CELL
 owner:111                          column=OWNER:firstName, timestamp=1700288638253, value=Harry
 owner:111                          column=OWNER:lastName, timestamp=1700288638265, value=Potter
 owner:111                          column=OWNER:licence-number, timestamp=1700288638223, value=111
 owner:111                          column=OWNER:phone, timestamp=1700288638272, value=88888888
 repair:001|1                       column=REPAIR:complexityLevel, timestamp=1700288638313, value=low
 repair:001|1                       column=REPAIR:labourCost, timestamp=1700288638280, value=20
 repair:001|1                       column=REPAIR:partsCost, timestamp=1700288638298, value=10
 repair:001|1                       column=TIME:endDate, timestamp=1700288638364, value=15/10/2023
 repair:001|1                       column=TIME:startDate, timestamp=1700288638342, value=1/10/2023
 repair:001|1                       column=VEHICLE:registration, timestamp=1700288638329, value=001
 repair:001|2                       column=REPAIR:complexityLevel, timestamp=1700288638409, value=low
 repair:001|2                       column=REPAIR:labourCost, timestamp=1700288638376, value=10
 repair:001|2                       column=REPAIR:partsCost, timestamp=1700288638396, value=20
 repair:001|2                       column=TIME:endDate, timestamp=1700288638434, value=15/11/2023
 repair:001|2                       column=TIME:startDate, timestamp=1700288638425, value=1/11/2023
 repair:001|2                       column=VEHICLE:registration, timestamp=1700288638420, value=001
 repair:002|1                       column=REPAIR:complexityLevel, timestamp=1700288638468, value=high
 repair:002|1                       column=REPAIR:labourCost, timestamp=1700288638452, value=50
 repair:002|1                       column=REPAIR:partsCost, timestamp=1700288638463, value=20
 repair:002|1                       column=TIME:endDate, timestamp=1700288638500, value=28/7/2022
 repair:002|1                       column=TIME:startDate, timestamp=1700288638493, value=20/7/2022
 repair:002|1                       column=VEHICLE:registration, timestamp=1700288638488, value=002
 repair:002|2                       column=REPAIR:complexityLevel, timestamp=1700288638534, value=low
 repair:002|2                       column=REPAIR:labourCost, timestamp=1700288638514, value=30
 repair:002|2                       column=REPAIR:partsCost, timestamp=1700288638526, value=5
 repair:002|2                       column=TIME:endDate, timestamp=1700288638578, value=20/10/2023
 repair:002|2                       column=TIME:startDate, timestamp=1700288638567, value=18/10/2023
 repair:002|2                       column=VEHICLE:registration, timestamp=1700288638557, value=002
 vehicle:111|001                    column=OWNER:licence-number, timestamp=1700288638069, value=111
 vehicle:111|001                    column=VEHICLE:make, timestamp=1700288638046, value=SG
 vehicle:111|001                    column=VEHICLE:model, timestamp=1700288638055, value=BMW
 vehicle:111|001                    column=VEHICLE:registration, timestamp=1700288638021, value=001
 vehicle:111|002                    column=OWNER:licence-number, timestamp=1700288638149, value=111
 vehicle:111|002                    column=VEHICLE:make, timestamp=1700288638099, value=CN
 vehicle:111|002                    column=VEHICLE:model, timestamp=1700288638116, value=BENZ
 vehicle:111|002                    column=VEHICLE:registration, timestamp=1700288638084, value=002
7 row(s) in 0.0690 seconds
```

Task2

Processing script ‘task2.hb’ to create table:

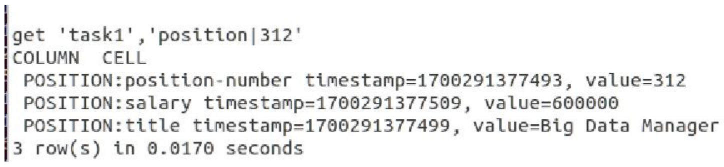


```
source('/home/bigdata/task2.hb')
create 'task1', 'APPLICATION','EMPLOYER','APPLICANT','POSITION'
0 row(s) in 2.2540 seconds
```



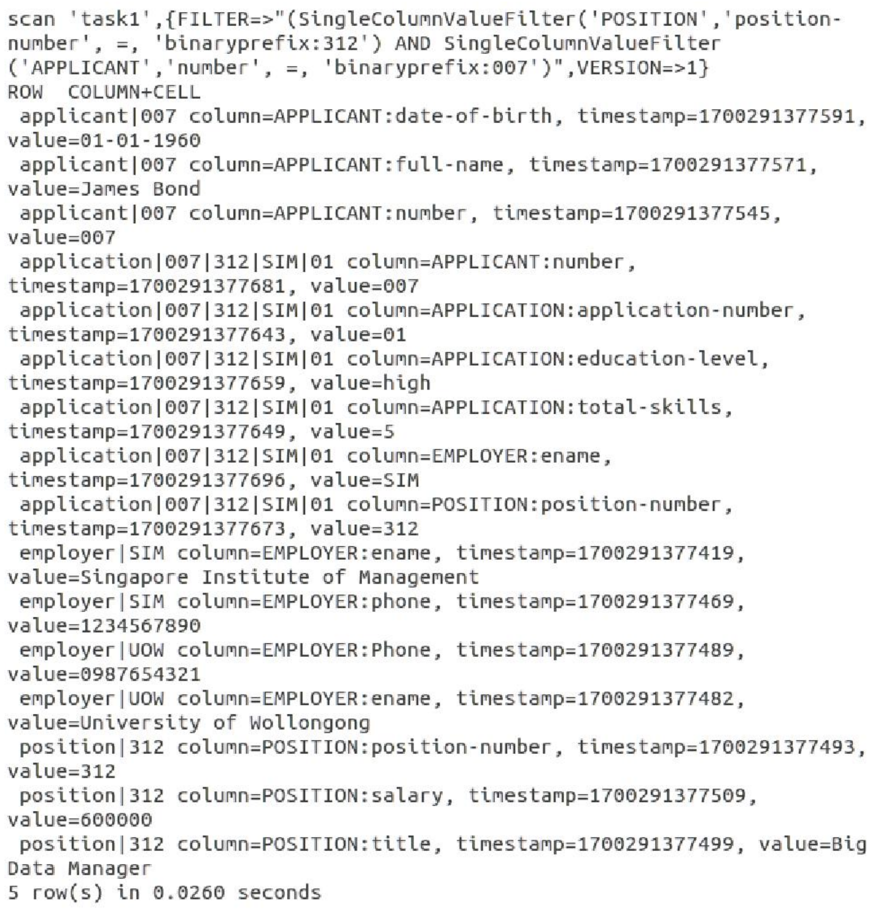
```
describe 'task1'
Table task1 is ENABLED
task1
COLUMN FAMILIES DESCRIPTION
{NAME => 'APPLICANT', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY
=> 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING =>
'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS => '0',
BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}
{NAME => 'APPLICATION', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY
=> 'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING =>
'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS => '0',
BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}
{NAME => 'EMPLOYER', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY =>
'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE',
TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS => '0', BLOCKCACHE
=> 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}
{NAME => 'POSITION', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY =>
'false', KEEP_DELETED_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE',
TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VERSIONS => '0', BLOCKCACHE
=> 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}
4 row(s) in 0.0170 seconds
```

(1)



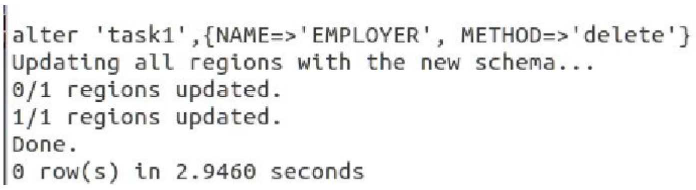
```
get 'task1','position|312'
COLUMN  CELL
 POSITION:position-number timestamp=1700291377493, value=312
 POSITION:salary timestamp=1700291377509, value=600000
 POSITION:title timestamp=1700291377499, value=Big Data Manager
3 row(s) in 0.0170 seconds
```

(2)



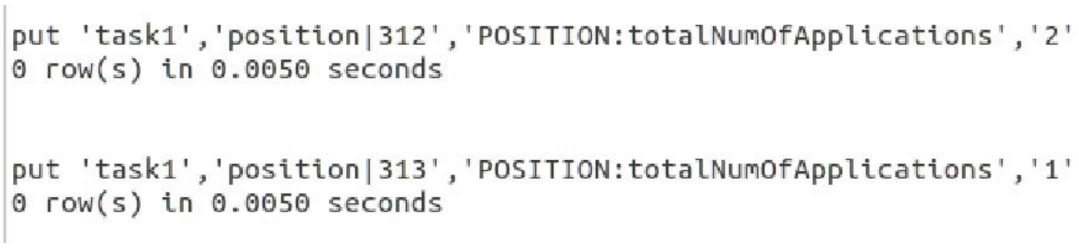
```
scan 'task1',{FILTER=>"(SingleColumnValueFilter('POSITION','position-
number', =, 'binaryprefix:312') AND SingleColumnValueFilter
('APPLICANT','number', =, 'binaryprefix:007')",VERSION=>1}
ROW  COLUMN+CELL
 applicant|007 column=APPLICANT:date-of-birth, timestamp=1700291377591,
value=01-01-1960
 applicant|007 column=APPLICANT:full-name, timestamp=1700291377571,
value=James Bond
 applicant|007 column=APPLICANT:number, timestamp=1700291377545,
value=007
 application|007|312|SIM|01 column=APPLICANT:number,
timestamp=1700291377681, value=007
 application|007|312|SIM|01 column=APPLICATION:application-number,
timestamp=1700291377643, value=01
 application|007|312|SIM|01 column=APPLICATION:education-level,
timestamp=1700291377659, value=high
 application|007|312|SIM|01 column=APPLICATION:total-skills,
timestamp=1700291377649, value=5
 application|007|312|SIM|01 column=EMPLOYER:ename,
timestamp=1700291377696, value=SIM
 application|007|312|SIM|01 column=POSITION:position-number,
timestamp=1700291377673, value=312
 employer|SIM column=EMPLOYER:ename, timestamp=1700291377419,
value=Singapore Institute of Management
 employer|SIM column=EMPLOYER:phone, timestamp=1700291377469,
value=1234567890
 employer|UOW column=EMPLOYER:Phone, timestamp=1700291377489,
value=0987654321
 employer|UOW column=EMPLOYER:ename, timestamp=1700291377482,
value=University of Wollongong
 position|312 column=POSITION:position-number, timestamp=1700291377493,
value=312
 position|312 column=POSITION:salary, timestamp=1700291377509,
value=600000
 position|312 column=POSITION:title, timestamp=1700291377499, value=Big
Data Manager
5 row(s) in 0.0260 seconds
```

(3)



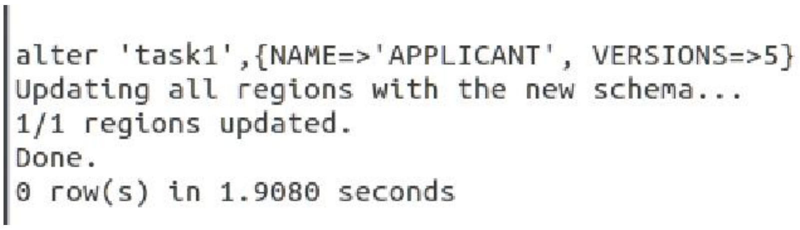
```
alter 'task1',{NAME=>'EMPLOYER', METHOD=>'delete'}
Updating all regions with the new schema...
0/1 regions updated.
1/1 regions updated.
Done.
0 row(s) in 2.9460 seconds
```

(4)



```
put 'task1','position|312','POSITION:totalNumOfApplications','2'
0 row(s) in 0.0050 seconds


put 'task1','position|313','POSITION:totalNumOfApplications','1'
0 row(s) in 0.0050 seconds
```

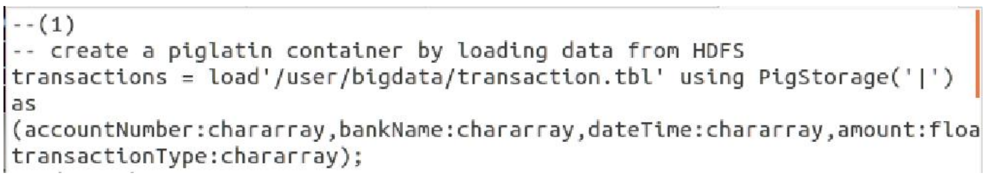
(5)



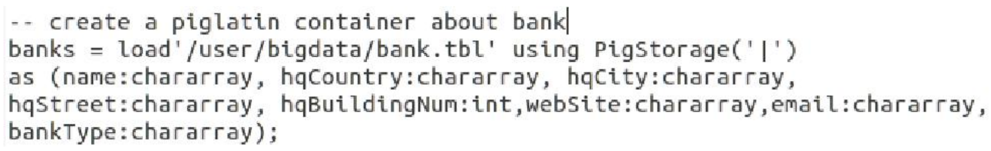
```
alter 'task1',{NAME=>'APPLICANT', VERSIONS=>5}
Updating all regions with the new schema...
1/1 regions updated.
Done.
0 row(s) in 1.9080 seconds
```

Task3

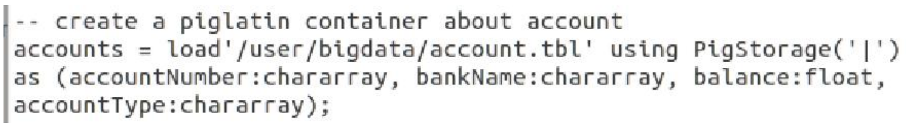
(1) Transfer file into HDFS and create containers:



```
--(1)
-- create a piglatin container by loading data from HDFS
transactions = load'/user/bigdata/transaction.tbl' using PigStorage('|')
as
(accountNumber:chararray,bankName:chararray,dateTime:chararray,amount:floa
transactionType:chararray);
```

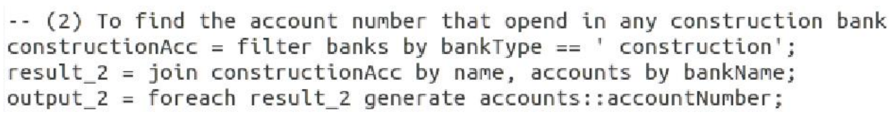


```
-- create a piglatin container about bank
banks = load'/user/bigdata/bank.tbl' using PigStorage('|')
as (name:chararray, hqCountry:chararray, hqCity:chararray,
hqStreet:chararray, hqBuildingNum:int,webSite:chararray,email:chararray,
bankType:chararray);
```

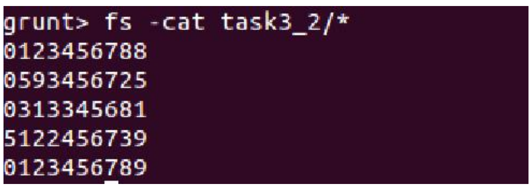


```
-- create a piglatin container about account
accounts = load'/user/bigdata/account.tbl' using PigStorage('|')
as (accountNumber:chararray, bankName:chararray, balance:float,
accountType:chararray);
```

(2) Statement & output:

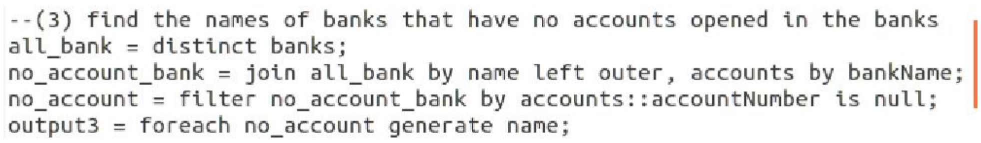


```
-- (2) To find the account number that opend in any construction bank
constructionAcc = filter banks by bankType == ' construction';
result_2 = join constructionAcc by name, accounts by bankName;
output_2 = foreach result_2 generate accounts::accountNumber;
```

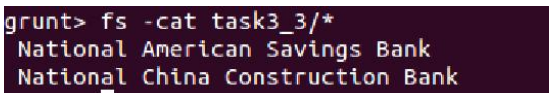


```
grunt> fs -cat task3_2/*
0123456788
0593456725
0313345681
5122456739
0123456789
```

(3) Statement & output:

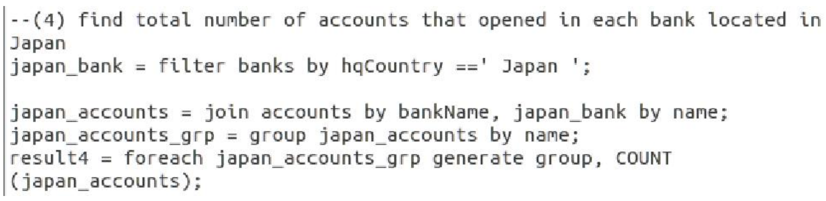


```
--(3) find the names of banks that have no accounts opened in the banks
all_bank = distinct banks;
no_account_bank = join all_bank by name left outer, accounts by bankName;
no_account = filter no_account_bank by accounts::accountNumber is null;
output3 = foreach no_account generate name;
```

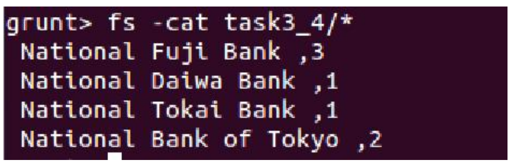


```
grunt> fs -cat task3_3/*
 National American Savings Bank
 National China Construction Bank
```

(4) Statement & output:

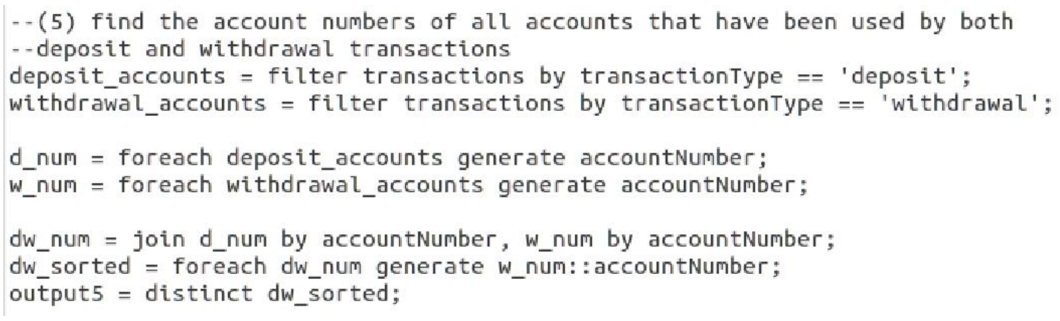


```
--(4) find total number of accounts that opened in each bank located in
Japan
japan_bank = filter banks by hqCountry ==' Japan ';

japan_accounts = join accounts by bankName, japan_bank by name;
japan_accounts_grp = group japan_accounts by name;
result4 = foreach japan_accounts_grp generate group, COUNT
(japan_accounts);
```

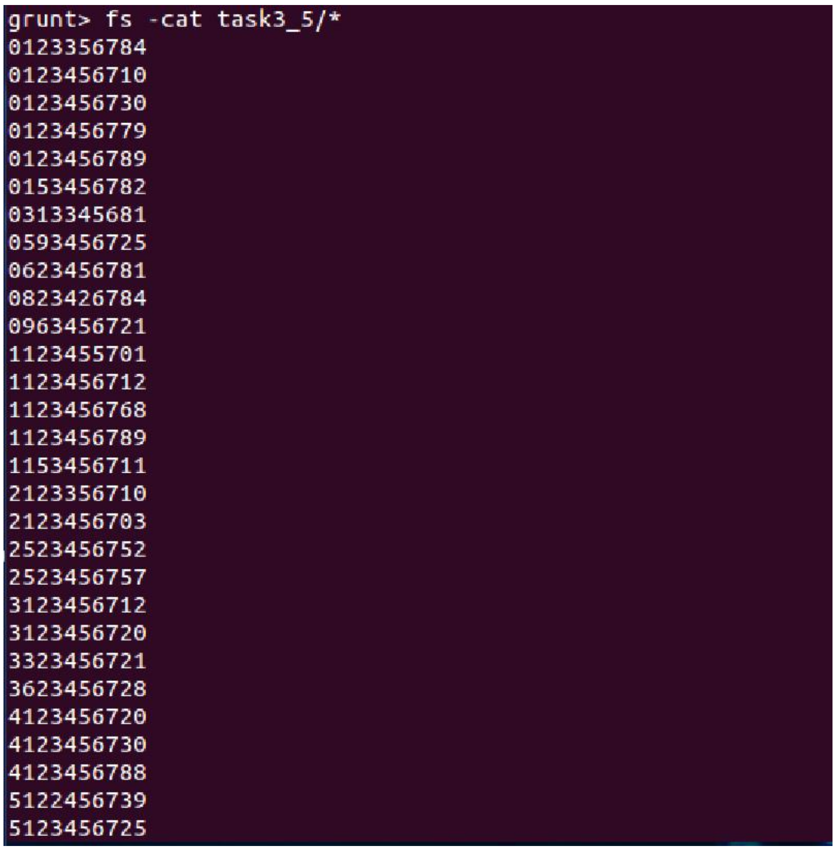


```
grunt> fs -cat task3_4/*
 National Fuji Bank ,3
 National Daiwa Bank ,1
 National Tokai Bank ,1
 National Bank of Tokyo ,2
```

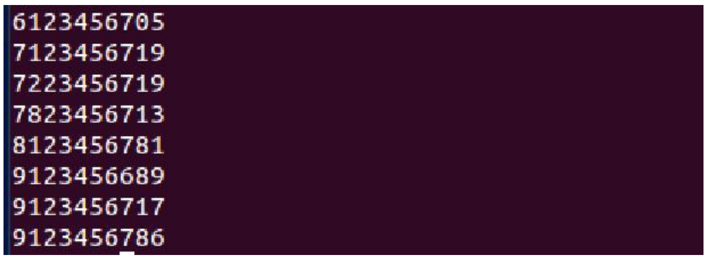
(5) Statement & output:



```
--(5) find the account numbers of all accounts that have been used by both
--deposit and withdrawal transactions
deposit_accounts = filter transactions by transactionType == 'deposit';
withdrawal_accounts = filter transactions by transactionType == 'withdrawal';

d_num = foreach deposit_accounts generate accountNumber;
w_num = foreach withdrawal_accounts generate accountNumber;

dw_num = join d_num by accountNumber, w_num by accountNumber;
dw_sorted = foreach dw_num generate w_num::accountNumber;
output5 = distinct dw_sorted;
```

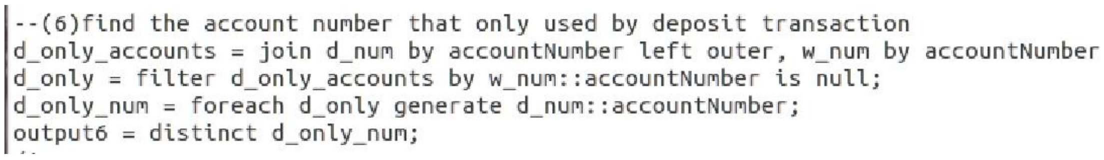


```
grunt> fs -cat task3_5/*
0123356784
0123456710
0123456730
0123456779
0123456789
0153456782
0313345681
0593456725
0623456781
0823426784
0963456721
1123455701
1123456712
1123456768
1123456789
1153456711
2123356710
2123456703
2523456752
2523456757
3123456712
3123456720
3323456721
3623456728
4123456720
4123456730
4123456788
5122456739
5123456725
```

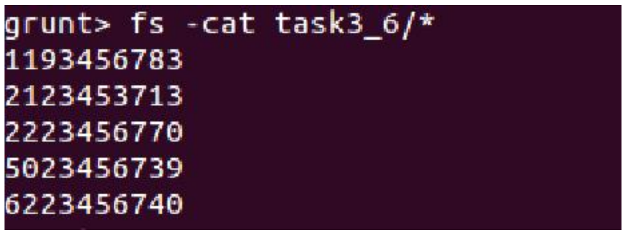


```
6123456705
7123456719
7223456719
7823456713
8123456781
9123456689
9123456717
9123456786
```

(6) Statement & output:



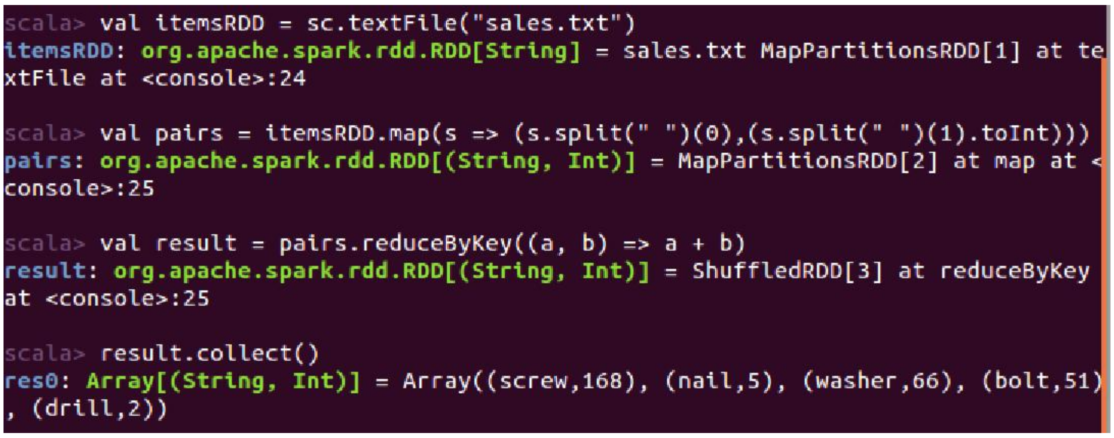
```
--(6)find the account number that only used by deposit transaction
d_only_accounts = join d_num by accountNumber left outer, w_num by accountNumber
d_only = filter d_only_accounts by w_num::accountNumber is null;
d_only_num = foreach d_only generate d_num::accountNumber;
output6 = distinct d_only_num;
```



```
grunt> fs -cat task3_6/*
1193456783
2123453713
2223456770
5023456739
6223456740
```

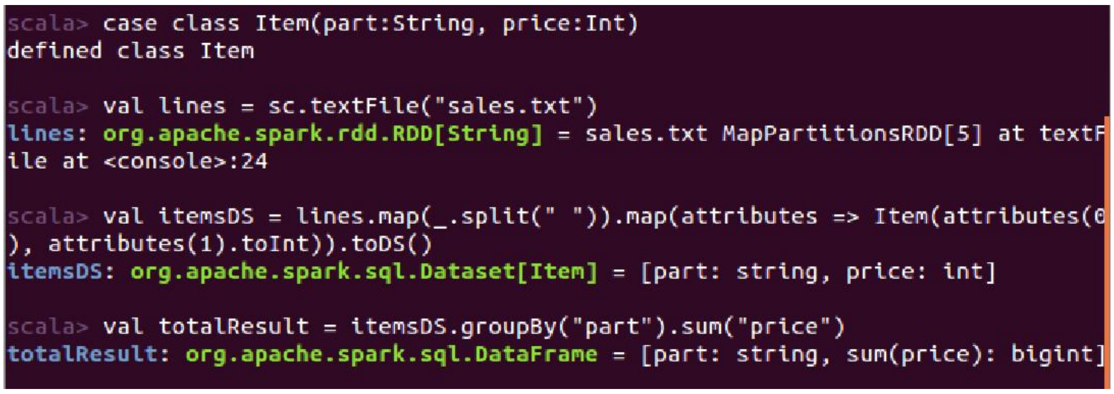
Task4

(1) Load file into RDD:



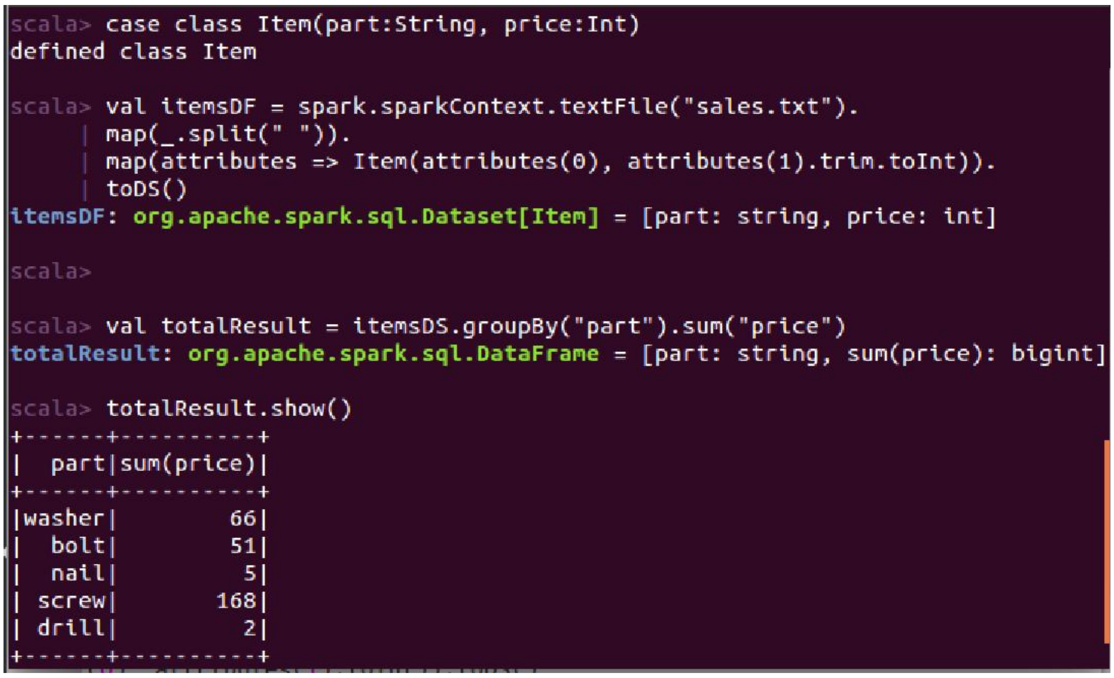
```
scala> val itemsRDD = sc.textFile("sales.txt")
itemsRDD: org.apache.spark.rdd.RDD[String] = sales.txt MapPartitionsRDD[1] at te
xtFile at <console>:24

scala> val pairs = itemsRDD.map(s => (s.split(" ")(0),(s.split(" ")(1).toInt)))
pairs: org.apache.spark.rdd.RDD[(String, Int)] = MapPartitionsRDD[2] at map at <
console>:25

scala> val result = pairs.reduceByKey((a, b) => a + b)
result: org.apache.spark.rdd.RDD[(String, Int)] = ShuffledRDD[3] at reduceByKey
at <console>:25

scala> result.collect()
res0: Array[(String, Int)] = Array((screw,168), (nail,5), (washer,66), (bolt,51)
, (drill,2))
```

(2) Load file into Dataset:



```
scala> case class Item(part:String, price:Int)
defined class Item

scala> val lines = sc.textFile("sales.txt")
lines: org.apache.spark.rdd.RDD[String] = sales.txt MapPartitionsRDD[5] at textF
ile at <console>:24

scala> val itemsDS = lines.map(_.split(" ")).map(attributes => Item(attributes(0
), attributes(1).toInt)).toDS()
itemsDS: org.apache.spark.sql.Dataset[Item] = [part: string, price: int]

scala> val totalResult = itemsDS.groupBy("part").sum("price")
totalResult: org.apache.spark.sql.DataFrame = [part: string, sum(price): bigint]
```

(3) Load file into DataFrame:



```
scala> case class Item(part:String, price:Int)
defined class Item

scala> val itemsDF = spark.sparkContext.textFile("sales.txt").
     | map(_.split(" ")).
     | map(attributes => Item(attributes(0), attributes(1).trim.toInt)).
     | toDS()
itemsDF: org.apache.spark.sql.Dataset[Item] = [part: string, price: int]

scala>

scala> val totalResult = itemsDS.groupBy("part").sum("price")
totalResult: org.apache.spark.sql.DataFrame = [part: string, sum(price): bigint]

scala> totalResult.show()
+------+----------+
|  part|sum(price)|
+------+----------+
|washer|        66|
|  bolt|        51|
|  nail|         5|
| screw|       168|
| drill|         2|
+------+----------+
```