

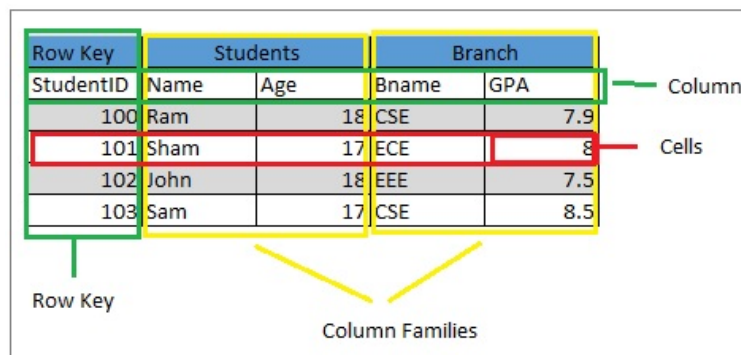
PTS Vaja 4

Za vajo 4 bomo uporabili naslednje tehnologije in orodja:

- importtsv
- Apache HBase 2.3.7

1. Apache HBase

HBase je NoSQL podatkovna baza zgrajena nad HDFS-om. Za iskanje določenih podatkov v HDFS-u, potrebno je pregledati celotno množico podatkov. Za razliko od tega pristopa, HBase omogoča naključno dodajanje in branje podatkov, kar izboljšuje učinkovitost sistema. V HBase-u se podatki interno shranjujejo v indeksirane dokumente shranjene v HDFS-u (angl. StoreFiles). Osnovne operacije za obdelavo podatkov v HBase-u so: dodajanje (angl. put), pridobivanje (angl. get) in snemanje (angl. scan). Podatkom se pristopa na podlagi sortiranih ključev vrstic (angl. rowkey) (slika 1). HBase model vsebuje stolpčne družine (angl. column family), ki vsebujejo več združenih stolpcev v obliki parov ključ-vrednost.



Slika 1: HBase podatkovni model ¹

¹ <https://dwgeek.com/wp-content/uploads/2017/09/Apache-HBase-Data-Model-Explanation.jpg>

Pri namestitvi HBase-a je nujno upoštevati združljivost različic s Hadoopom. V našem primeru je HBase v2.3.7 tista, ki je združljiva s Hadoopom 3.2.1. Podobno kot Hadoop, HBase tudi ponuja tri načina namestitve: samostojen, psevdo-porazdeljen in popolnoma porazdeljen.

Sama namestitvev HBase-a v gruči je enostavna. Privzeto HBase shranjuje podatke na interni lokaciji, vendar je možno vzpostaviti varianto samostojnega načina, pri čemer se podatki trajno shranjujejo v HDFS. V tem primerju je potrebno določiti IP naslov HDFS mape za HBase podatke ter uporabo samostojnega načina v *conf/hbase-site.xml* dokumentu:

```
<property>
  <name>hbase.rootdir</name>
  <value>hdfs://localhost:9000/hbase</value>
```

```

</property>
<property>
  <name>hbase.cluster.distributed</name>
  <value>false</value>
</property>

```

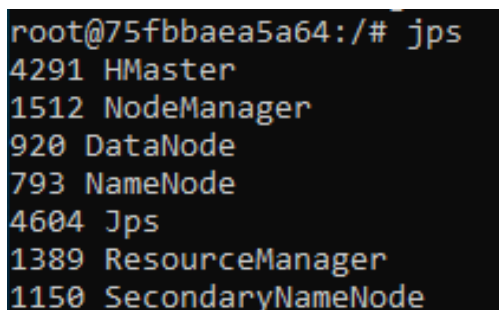
V HDFS-u ni potrebno vnaprej ustvariti mape *hbase*, saj to sistem sam naredi. Opazite, da je pot določena v parametru *hbase.rootdir* podobna poti HDFS-a (isti port).

Seveda, preden zaženemo HBase je treba tudi zagotoviti, da je HDFS zagnan, in da NameNode ni v varnem načinu (izvedete ukaz: *hadoop dfsadmin -safemode leave*).

Najprej vstopimo v korensko mapo HBase namestitve s pomočjo ukaza:

```
cd $HBASE_HOME$
```

Ko smo v korenski mapi HBase-a, lahko zaženemo HBase procese z uporabo skripte *start-hbase.sh*, ki se nahaja v */bin* mapi znotraj HBase korenske mape. Če je vse uspešno zagnano, bi v izpisu ukaza *jps* za izpis aktivnih Java procesov morali videti HMaster proces vezan za HBase:



```

root@75fbbaea5a64:/# jps
4291 HMaster
1512 NodeManager
920 DataNode
793 NameNode
4604 Jps
1389 ResourceManager
1150 SecondaryNameNode

```

Slika 2: Seznam Java procesov po zagnanem HBase-u.

HBase ukaze pišemo znotraj HBase terminala, v katerega vstopimo z ukazom: *hbase shell* (ko smo znotraj */bin* mape).

V HBase-u je imenski prostor (angl. namespace) ekvivalent relacijski podatkovni bazi. Privzeto sta v sistemu dostopna dva imenska prostora *default* in *hbase*. Nov imenski prostor "pts" lahko ustvarimo z ukazom: *create_namespace 'pts'*. Seznamu imenskih prostorov dostopamo z izvedbo ukaza: *list_namespace*. Kot naslednji korak v imenskem prostoru ustvarjamo tabele. Če ne uporabljamo privzeti *default* imenski prostor, potrebno je pri usvarjanju tabele določiti tudi imenski prostor v katerem jo želimo ustvariti (ločeno s dvopičjem) (npr. *create 'pts:tabela'*).

Zdaj bomo ustvarili ciljno tabelo *offers* v HBase-u, ki bo vsebovala eno stolpčno družino (*o* za naročila), medtem ko pa bo vsaka vrstica določena z *offer_id* stolpcem kot ključem vrstice (*row_key*). Omenjeno tabelo ustvarimo z naslednjim ukazom v HBase terminalu:

```
create 'pts:offers', 'rest', 'item', 'o'
```

Z eštudija prevzemite dokument *restaurant-offers.csv*, ki vsebuje skupne rezultate iz pogleda *v_offer*, in sicer stolpce *offer_id*, *rest_id*, *rest_name*, *item_id*, *item_name*, *price*. Ta dokument dodajte v HDFS. *Opomba:* Za vajo lahko tudi uporabite CSV datoteko, ki ste jo uporabili v prejšnji vaji za implementacijo MapReduce posla. Ne pozabite shraniti Hadoop sistemske spremenljivke:

```
export PATH=$PATH:$HADOOP_HOME/bin
```

Ko ste dodali datoteko v HDFS, vrstice iz tega dokumenta uvezete v HBase *offers* tabelo z uporabo orodja *ImportTsv*.

```
hbase org.apache.hadoop.hbase.mapreduce.ImportTsv -Dimporttsv.separator=,  
-Dimporttsv.columns="HBASE_ROW_KEY,rest:rest_id,rest:rest_name,  
item:item_id, item:item_name,o:price" pts:offers hdfs://localhost:9000/restaurant-offers.csv
```

Zdaj lahko preverimo status tabele *offers* z izvedbo ukaza, ki nam bo vrnil vse vnose za posamezne stolpce v tabeli:

```
scan 'pts:offers'
```

Razen ukaza *scan*, za branje podatkov lahko uporabimo tudi ukaz *get*, ki vrne eno vrstico na podlagi ključa. Za filtriranje podatkov so dostopni predoločeni filtri. Za ročno dodajanje vnosov v tabelo uporabljamo ukaz *put*: *put* <'tablename'>,<'rowname'>,<'columnvalue'>,<'value'>. Če želimo izbrisati določeno tabelo jo najprej moremo onemogočiti z ukazom *disable*.

Dodatni viri:

- <https://www.guru99.com/hbase-shell-general-commands.html>
- <https://dwgeek.com/read-hbase-tables-using-scan-shell-command-examples.html/>
- https://docs.cloudera.com/documentation/enterprise/6/6.3/topics/admin_hbase_filtering.html
- <https://dwgeek.com/commonly-used-hbase-table-management-shell-commands.html/>