

Hbase – to rozproszona baza danych zorientowana kolumnowo zbudowana na systemie plików Hadoop. Jest to projekt wolnego oprogramowania (ang. open source) rozwijany przez firmę Apache.

Mechнизм przechowywania danych w HBase:

Hbase jest bazą danych zorientowaną kolumnowo, a tabele w niej są sortowane według wierszy. Schemat tabeli definiuje tylko "rodziny kolumn" (ang. column families), które są parami wartości i klucza. Tabela ma wiele "rodzin kolumn", a każda "rodzina kolumn" może mieć dowolną liczbę kolumn. Kolejne wartości kolumn są przechowywane w sposób ciągły na dysku. Każda wartość komórki w tabeli ma znacznik czasu. Krótko mówiąc, w HBase:

- Tabela jest zbiorem wierszy.
- Wiersz to zbiór "rodzin kolumn".
- "Rodzina kolumn" to zbiór kolumn.
- Kolumna to zbiór par klucz-wartość.

Przykładowy schemat tabeli HBase:

RowId	Column Family			Column Family			Column Family		
	Col1	Col2	Col3	Col1	Col2	Col3	Col1	Col2	Col3
1									
2									
3									

Przykładowa tabela HBase:

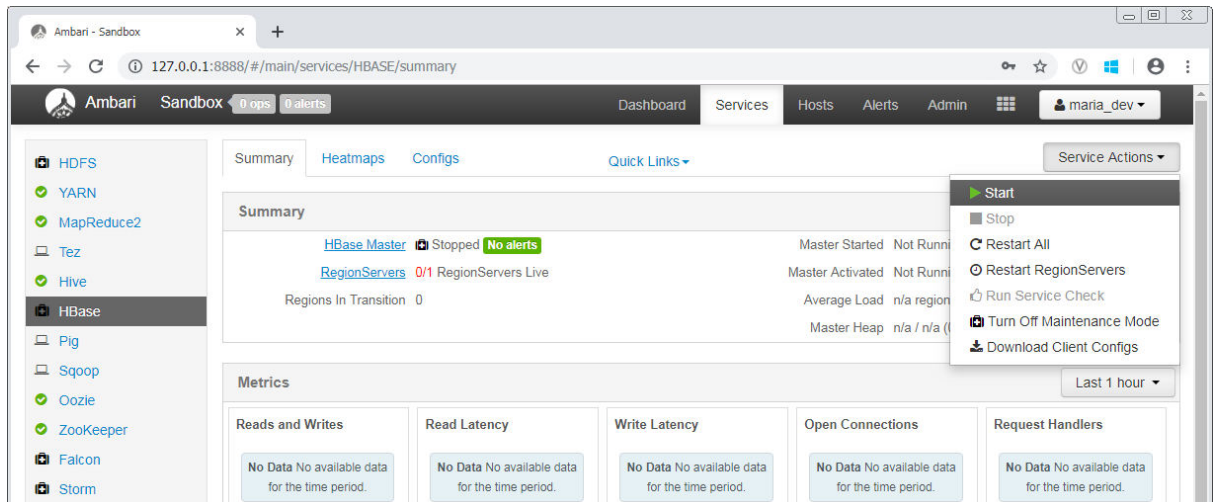
ID	Dane prywatne				Dane służbowe	
	Imię	Nazwisko	Miasto	Telefon	Stanowisko	Zarobki
1	Jan	Nowak	Warszawa	513609915	Programista	200000
2	Piotr	Kowalski	Gdańsk	506876549	Tester	120000
3	Natalia	Kamińska	Łódź	601254144	Menadzer	100000

Uruchamianie HBase w Ambari:

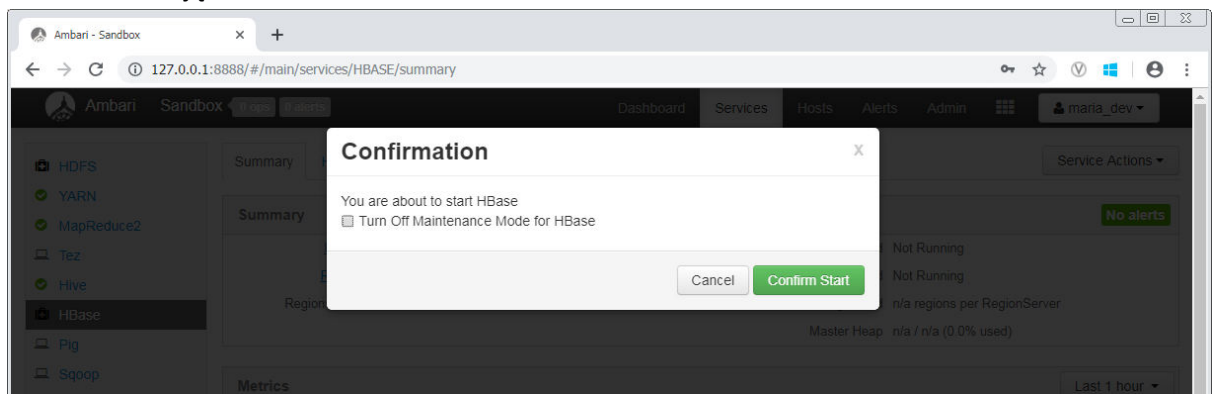
- Po zalogowaniu się do Ambari, wybierz **HBase** z kolumny po lewej stronie:

The screenshot shows the Ambari web interface. On the left, a sidebar lists services: HDFS, YARN, MapReduce2, Tez, Hive, HBase (selected), Pig, Sqoop, Oozie, ZooKeeper, Falcon, and Storm. The main content area is titled 'Summary' and shows the status of the HBase service. The HBase Master is listed as 'Stopped' with a green 'No alerts' badge. Below this, it shows 'RegionServers: 0/1 RegionServers Live' and 'Regions In Transition: 0'. To the right, there are status indicators: 'Master Started: Not Running', 'Master Activated: Not Running', 'Average Load: n/a regions per RegionServer', and 'Master Heap: n/a / n/a (0.0% used)'. At the bottom, there are five metric cards: 'Reads and Writes', 'Read Latency', 'Write Latency', 'Open Connections', and 'Request Handlers', all of which display 'No Data No available data for the time period.' for the 'Last 1 hour' view.

- W prawym górnym rogu z menu **Service Actions** wybierz **Start**:

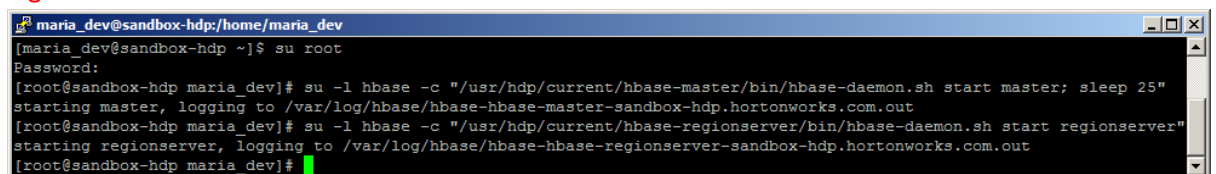


- Potwierdź klikając **Confirm Start**:



Uruchamianie HBase z wiersza poleceń:

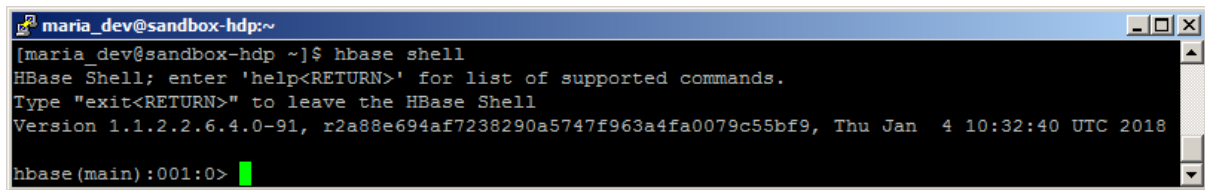
- Zaloguj się na maszynę korzystając w loginu i hasła: **maria_dev**
- Aby uruchomić HBase przełącz się na superuser'a. Wpisz: **su root** i podaj hasło **danilewicz**
- Uruchom HBase na głównym serwerze (Master Server) wpisując:
su -l hbase -c "/usr/hdp/current/hbase-master/bin/hbase-daemon.sh start master; sleep 25"
- Uruchom HBase na serwerach regionalnych (Region Servers) wpisując:
su -l hbase -c "/usr/hdp/current/hbase-regionserver/bin/hbase-daemon.sh start regionserver"



- Na koniec wyloguj się wpisując: **exit**

HBase Shell – jest to powłoka za pomocą której można komunikować się z HBase.

Uruchom HBase Shell wpisując **hbase shell**:



```
maria_dev@sandbox-hdp:~  
[maria_dev@sandbox-hdp ~]$ hbase shell  
HBase Shell; enter 'help<RETURN>' for list of supported commands.  
Type "exit<RETURN>" to leave the HBase Shell  
Version 1.1.2.2.6.4.0-91, r2a88e694af7238290a5747f963a4fa0079c55bf9, Thu Jan 4 10:32:40 UTC 2018  
hbase(main):001:0>
```

Ogólne polecenia:

- status – podaje status HBase’a, np. liczbę serwerów.
- version – podaje numer wersji HBase’a.
- whoami – podaje informacje o użytkowniku.
- table_help – wyświetla listę komend dostępnych dla tabeli

Polecenia do pracy na tabelach:

- create – tworzy tabelę.
- list – wyświetla listę wszystkich tabel w HBase.
- disable – wyłącza tabelę.
- is_disabled – sprawdza czy tabela jest wyłączona.
- enable – włącza tabelę.
- is_enabled – sprawdza czy tabela jest włączona.
- describe – wyświetla opis tabeli.
- alter – zmienia tabelę.
- exists – sprawdza czy tabela istnieje.
- drop – kasuje tabelę z HBase’a.
- drop_all – kasuje table pasujące do schematu regex.

Polecenia do pracy z danymi:

- put – umieszcza wartość komórki w określonej kolumnie w określonym wierszu i w określonej tabeli.
- get – pobiera wartość wiersza lub komórki.
- delete – usuwa wartość komórki w tabeli.
- deleteall – usuwa wszystkie komórki w danym wierszu.
- scan – skanuje i zwraca dane tabeli.
- count – zlicza i zwraca liczbę wierszy w tabeli.
- truncate – wyłącza, kasuje i odtwarza określoną tabelę.

Tworzenie tabeli przy użyciu HBase Shell:

Składnia:

create 'table_name', 'column_family'

gdzie:

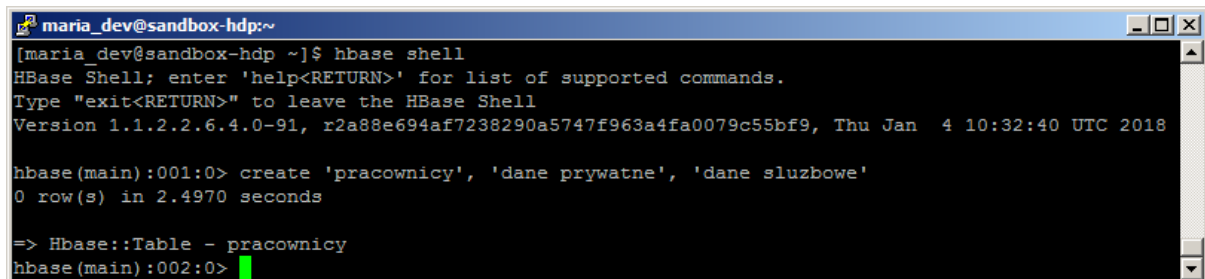
table_name – to nazwa tabeli,

column_family – to nazwa rodziny kolumn.

Przykład:

Utworzymy tabelę pracowników z dwiema rodzinami kolumn: dane prywatne i dane służbowe:

create 'pracownicy', 'dane prywatne', 'dane sluzbowe'



```
maria_dev@sandbox-hdp:~  
[maria_dev@sandbox-hdp ~]$ hbase shell  
HBase Shell; enter 'help<RETURN>' for list of supported commands.  
Type "exit<RETURN>" to leave the HBase Shell  
Version 1.1.2.2.6.4.0-91, r2a88e694af7238290a5747f963a4fa0079c55bf9, Thu Jan 4 10:32:40 UTC 2018  
  
hbase(main):001:0> create 'pracownicy', 'dane prywatne', 'dane sluzbowe'  
0 row(s) in 2.4970 seconds  
  
=> Hbase::Table - pracownicy  
hbase(main):002:0>
```

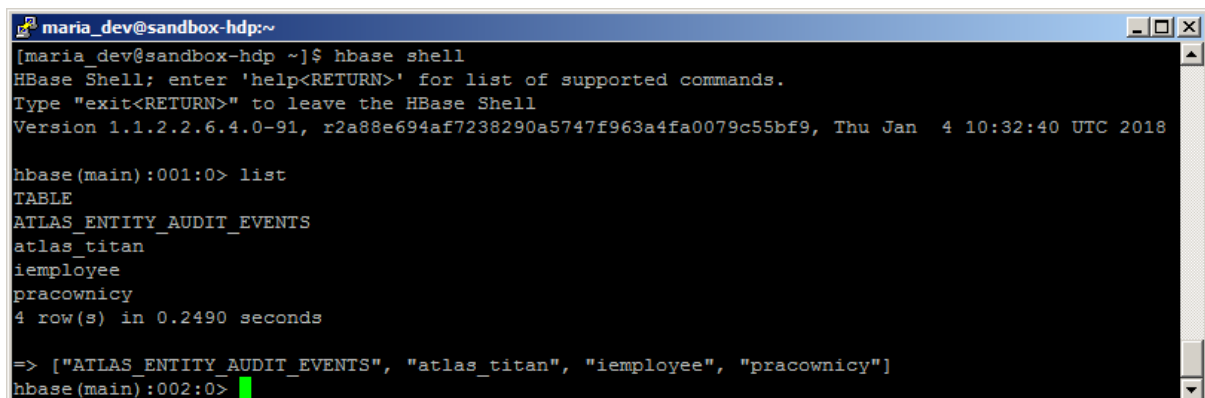
Wyświetlanie listy tabel przy użyciu HBase Shell:

Składnia:

list

Przykład:

Sprawdźmy czy nasza tabela pracowników została utworzona:



```
maria_dev@sandbox-hdp:~  
[maria_dev@sandbox-hdp ~]$ hbase shell  
HBase Shell; enter 'help<RETURN>' for list of supported commands.  
Type "exit<RETURN>" to leave the HBase Shell  
Version 1.1.2.2.6.4.0-91, r2a88e694af7238290a5747f963a4fa0079c55bf9, Thu Jan 4 10:32:40 UTC 2018  
  
hbase(main):001:0> list  
TABLE  
ATLAS_ENTITY_AUDIT_EVENTS  
atlas_titan  
iemployee  
pracownicy  
4 row(s) in 0.2490 seconds  
  
=> ["ATLAS_ENTITY_AUDIT_EVENTS", "atlas_titan", "iemployee", "pracownicy"]  
hbase(main):002:0>
```

Wyłączanie tabeli przy użyciu HBase Shell:

Aby usunąć tabelę lub zmienić jej ustawienia należy najpierw wyłączyć tabelę za pomocą polecenia disable.

Składnia:

disable 'table_name'

gdzie:

table_name – to nazwa tabeli którą chcemy wyłączyć.

Przykład:

Wyłącz tabelę pracownicy wpisując: **disable 'pracownicy'**

```
maria_dev@sandbox-hdp:~  
[maria_dev@sandbox-hdp ~]$ hbase shell  
HBase Shell; enter 'help<RETURN>' for list of supported commands.  
Type "exit<RETURN>" to leave the HBase Shell  
Version 1.1.2.2.6.4.0-91, r2a88e694af7238290a5747f963a4fa0079c55bf9, Thu Jan  4 10:32:40 UTC 2018  
  
hbase(main):001:0> disable 'pracownicy'  
0 row(s) in 2.5240 seconds  
  
hbase(main):002:0>
```

Sprawdź czy tabela jest wyłączona wpisując: **is_disabled 'pracownicy'**

```
maria_dev@sandbox-hdp:~  
[maria_dev@sandbox-hdp ~]$ hbase shell  
HBase Shell; enter 'help<RETURN>' for list of supported commands.  
Type "exit<RETURN>" to leave the HBase Shell  
Version 1.1.2.2.6.4.0-91, r2a88e694af7238290a5747f963a4fa0079c55bf9, Thu Jan  4 10:32:40 UTC 2018  
  
hbase(main):001:0> is_disabled 'pracownicy'  
true  
0 row(s) in 0.2360 seconds  
  
hbase(main):002:0>
```

Włączanie tabeli przy użyciu HBase Shell:

Składnia:

enable 'table_name'

gdzie:

table_name – to nazwa tabeli którą chcemy włączyć.

Przykład:

Włącz tabelę pracownicy wpisując: **enable 'pracownicy'**

```
maria_dev@sandbox-hdp:~  
[maria_dev@sandbox-hdp ~]$ hbase shell  
HBase Shell; enter 'help<RETURN>' for list of supported commands.  
Type "exit<RETURN>" to leave the HBase Shell  
Version 1.1.2.2.6.4.0-91, r2a88e694af7238290a5747f963a4fa0079c55bf9, Thu Jan  4 10:32:40 UTC 2018  
  
hbase(main):001:0> enable 'pracownicy'  
0 row(s) in 1.5440 seconds  
  
hbase(main):002:0>
```

Sprawdź czy tabela jest włączona wpisując: **is_enabled 'pracownicy'**

```
maria_dev@sandbox-hdp:~  
HBase Shell; enter 'help<RETURN>' for list of supported commands.  
Type "exit<RETURN>" to leave the HBase Shell  
Version 1.1.2.2.6.4.0-91, r2a88e694af7238290a5747f963a4fa0079c55bf9, Thu Jan  4 10:32:40 UTC 2018  
  
hbase(main):001:0> is_enabled 'pracownicy'  
true  
0 row(s) in 0.2170 seconds  
  
hbase(main):002:0> █
```

Wyświetlanie opisu tabeli przy użyciu HBase Shell:

Składnia:

describe 'table_name'

gdzie:

table_name – to nazwa tabeli której potrzebujemy opis.

Przykład:

Wyświetl opis tabeli pracownicy wpisując: **describe 'pracownicy'**

```
maria_dev@sandbox-hdp:~  
hbase(main):001:0> describe 'pracownicy'  
Table pracownicy is ENABLED  
pracownicy  
COLUMN FAMILIES DESCRIPTION  
{NAME => 'dane prywatne', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY => 'false', KEEP_DELETE  
D_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VE  
RSIONS => '0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}  
{NAME => 'dane sluzbowe', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY => 'false', KEEP_DELETE  
D_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VE  
RSIONS => '0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}  
2 row(s) in 0.2880 seconds  
  
hbase(main):002:0> █
```

Dodawanie rodziny kolumn przy użyciu HBase Shell:

Składnia:

alter 'table_name', NAME => 'column_family'

gdzie:

table_name – to nazwa tabeli którą bedziemy zmieniać,

column_family – to nazwa rodziny kolumn którą chcemy dodać.

Przykład:

Dodać rodzinę kolumn o nazwie 'dane tajne' to tabeli 'pracownicy' wpisując:

alter 'pracownicy', NAME => 'dane tajne'

```
maria_dev@sandbox-hdp:~  
[maria_dev@sandbox-hdp ~]$ hbase shell  
HBase Shell; enter 'help<RETURN>' for list of supported commands.  
Type "exit<RETURN>" to leave the HBase Shell  
Version 1.1.2.2.6.4.0-91, r2a88e694af7238290a5747f963a4fa0079c55bf9, Thu Jan  4 10:32:40 UTC 2018  
  
hbase(main):001:0> alter 'pracownicy', NAME => 'dane tajne'  
Updating all regions with the new schema...  
1/1 regions updated.  
Done.  
0 row(s) in 2.2250 seconds  
  
hbase(main):002:0>
```

Sprawdź czy rodzina kolumn została dodana wpisując: **describe 'pracownicy'**

```
maria_dev@sandbox-hdp:~  
hbase(main):002:0> describe 'pracownicy'  
Table pracownicy is ENABLED  
pracownicy  
COLUMN FAMILIES DESCRIPTION  
{NAME => 'dane prywatne', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY => 'false', KEEP_DELETE  
D_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VE  
RSIONS => '0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}  
{NAME => 'dane sluzbowe', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY => 'false', KEEP_DELETE  
D_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VE  
RSIONS => '0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}  
{NAME => 'dane tajne', BLOOMFILTER => 'ROW', VERSIONS => '1', IN_MEMORY => 'false', KEEP_DELETE  
D_CELLS => 'FALSE', DATA_BLOCK_ENCODING => 'NONE', TTL => 'FOREVER', COMPRESSION => 'NONE', MIN_VE  
RSIONS => '0', BLOCKCACHE => 'true', BLOCKSIZE => '65536', REPLICATION_SCOPE => '0'}  
3 row(s) in 0.0440 seconds  
  
hbase(main):003:0>
```

Usuwanie rodziny kolumn przy użyciu HBase Shell:

Składnia:

```
alter 'table_name', 'delete' => 'column_family'
```

gdzie:

table_name – to nazwa tabeli z której będziemy usuwać rodzinę kolumn,

column_family – to rodzina kolumn, którą będziemy usuwać.

Przykład:

Usuń rodzinę kolumn 'dane tajne' z tabeli pracownicy wpisując:

```
alter 'pracownicy', 'delete' => 'dane tajne'
```

```
maria_dev@sandbox-hdp:~  
hbase(main):005:0> alter 'pracownicy', 'delete' => 'dane tajne'  
Updating all regions with the new schema...  
1/1 regions updated.  
Done.  
0 row(s) in 2.4200 seconds  
  
hbase(main):006:0>
```

Sprawdzanie czy tabela istnieje przy użyciu HBase Shell:

Składnia:

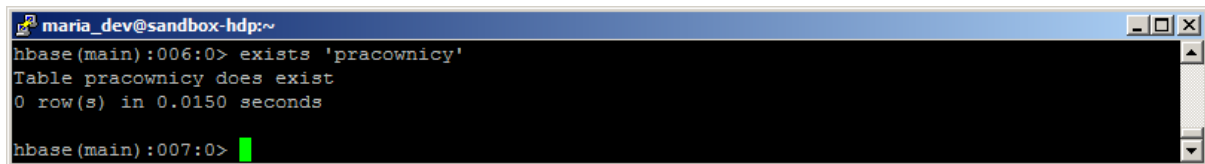
`exists 'table_name'`

gdzie:

`table_name` – to tabela której szukamy.

Przykład:

Sprawdź czy tabela pracownicy istnieje wpisując: `exists 'pracownicy'`



```
maria_dev@sandbox-hdp:~  
hbase(main):006:0> exists 'pracownicy'  
Table pracownicy does exist  
0 row(s) in 0.0150 seconds  
hbase(main):007:0>
```

Usuwanie tabeli przy użyciu HBase Shell:

Składnia:

`drop 'table_name'`

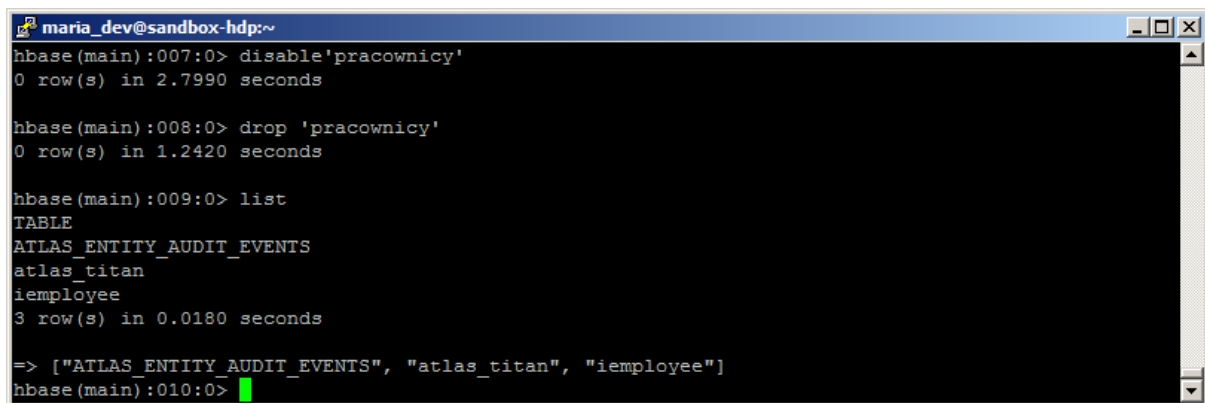
gdzie:

`table_name` – to nazwa tabeli którą chcemy usunąć.

Przykład:

Zanim skasujemy tabelę pracownicy musimy ją wyłączyć wpisując: `disable 'pracownicy'`

Teraz możemy skasować tabelę wpisując: `drop 'pracownicy'`



```
maria_dev@sandbox-hdp:~  
hbase(main):007:0> disable 'pracownicy'  
0 row(s) in 2.7990 seconds  
hbase(main):008:0> drop 'pracownicy'  
0 row(s) in 1.2420 seconds  
hbase(main):009:0> list  
TABLE  
ATLAS_ENTITY_AUDIT_EVENTS  
atlas_titan  
iemployee  
3 row(s) in 0.0180 seconds  
=> ["ATLAS_ENTITY_AUDIT_EVENTS", "atlas_titan", "iemployee"]  
hbase(main):010:0>
```

Wstawianie danych przy użyciu HBase Shell:

Składnia:

`put 'table_name', 'row_id', 'column_family:column_name', 'value'`

gdzie:

`table_name` – to tabela do której wstawiamy dane,

row_id – identyfikator wiersza,

column_family – to rodzina kolumn do której wstawiamy dane,

column_name – to nazwa kolumny do której wstawiamy dane,

value – to dane które wstawiamy.

Przykład:

Dodamy jeden wiersz do tabeli pracownicy:

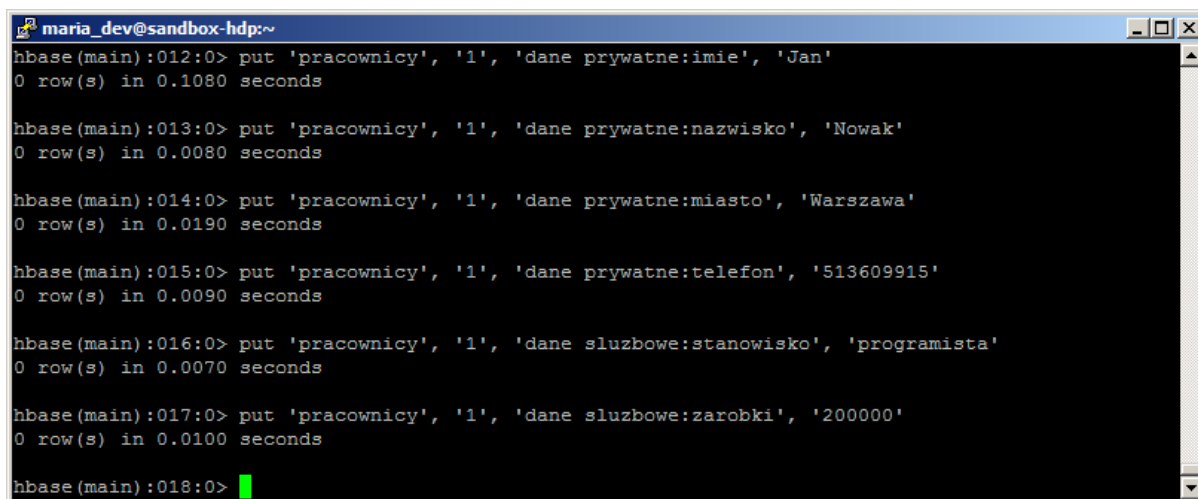
ID	Dane prywatne				Dane służbowe	
	Imię	Nazwisko	Miasto	Telefon	Stanowisko	Zarobki
1	Jan	Nowak	Warszawa	513609915	Programista	200000

Ponieważ wcześniej skasowaliśmy naszą tabelę, więc utwórzmy ją na nowo:

```
create 'pracownicy', 'dane prywatne', 'dane sluzbowe'
```

Teraz możemy wypełnić naszą tabelę wpisując:

```
put 'pracownicy', '1', 'dane prywatne:imie', 'Jan'
put 'pracownicy', '1', 'dane prywatne:nazwisko', 'Nowak'
put 'pracownicy', '1', 'dane prywatne:miasto', 'Warszawa'
put 'pracownicy', '1', 'dane prywatne:telefon', '513609915'
put 'pracownicy', '1', 'dane sluzbowe:stanowisko', 'programista'
put 'pracownicy', '1', 'dane sluzbowe:zarobki', '200000'
```



```
maria_dev@sandbox-hdp:~
hbase(main):012:0> put 'pracownicy', '1', 'dane prywatne:imie', 'Jan'
0 row(s) in 0.1080 seconds

hbase(main):013:0> put 'pracownicy', '1', 'dane prywatne:nazwisko', 'Nowak'
0 row(s) in 0.0080 seconds

hbase(main):014:0> put 'pracownicy', '1', 'dane prywatne:miasto', 'Warszawa'
0 row(s) in 0.0190 seconds

hbase(main):015:0> put 'pracownicy', '1', 'dane prywatne:telefon', '513609915'
0 row(s) in 0.0090 seconds

hbase(main):016:0> put 'pracownicy', '1', 'dane sluzbowe:stanowisko', 'programista'
0 row(s) in 0.0070 seconds

hbase(main):017:0> put 'pracownicy', '1', 'dane sluzbowe:zarobki', '200000'
0 row(s) in 0.0100 seconds

hbase(main):018:0>
```

Wyświetlmy zawartość tabeli wpisując: `scan 'pracownicy'`

```
maria_dev@sandbox-hdp:~  
hbase(main):019:0> scan 'pracownicy'  
ROW COLUMN+CELL  
1 column=dane prywatne:imie, timestamp=1554382430248, value=Jan  
1 column=dane prywatne:miasto, timestamp=1554382449581, value=Warszawa  
1 column=dane prywatne:nazwisko, timestamp=1554382443747, value=Nowak  
1 column=dane prywatne:telefon, timestamp=1554382457086, value=513609915  
1 column=dane sluzbowe:stanowisko, timestamp=1554382466530, value=programista  
1 column=dane sluzbowe:zarobki, timestamp=1554382473698, value=200000  
1 row(s) in 0.0170 seconds  
hbase(main):020:0>
```

Aktualizacja danych przy użyciu HBase Shell:

Składnia:

`put 'table_name', 'row_id', 'column_family:column_name', 'new_value'`

gdzie:

`table_name` – to nazwa tabeli w której aktualizujemy dane,

`row_id` – identyfikator wiersza,

`column_family` – to rodzina kolumn w której aktualizujemy dane,

`column_name` – to nazwa kolumny w której aktualizujemy dane,

`new_value` – to nowa wartość którą chcemy zapisać.

Przykład:

Zmieńmy miasto w naszej tabeli na Katowice wpisując:

`put 'pracownicy', '1', 'dane prywatne:miasto', 'Katowice'`

```
maria_dev@sandbox-hdp:~  
hbase(main):002:0> put 'pracownicy', '1', 'dane prywatne:miasto', 'Katowice'  
0 row(s) in 0.0260 seconds  
hbase(main):003:0>
```

Wyświetlmy zawartość tabeli wpisując: `scan 'pracownicy'`

```
maria_dev@sandbox-hdp:~  
hbase(main):004:0> scan 'pracownicy'  
ROW COLUMN+CELL  
1 column=dane prywatne:imie, timestamp=1554382430248, value=Jan  
1 column=dane prywatne:miasto, timestamp=1554400856765, value=Katowice  
1 column=dane prywatne:nazwisko, timestamp=1554382443747, value=Nowak  
1 column=dane prywatne:telefon, timestamp=1554382457086, value=513609915  
1 column=dane sluzbowe:stanowisko, timestamp=1554382466530, value=programista  
1 column=dane sluzbowe:zarobki, timestamp=1554382473698, value=200000  
1 row(s) in 0.0500 seconds  
hbase(main):005:0>
```

Odczytywanie danych przy użyciu HBase Shell:

Składnia:

`get 'table_name', 'row_id'`

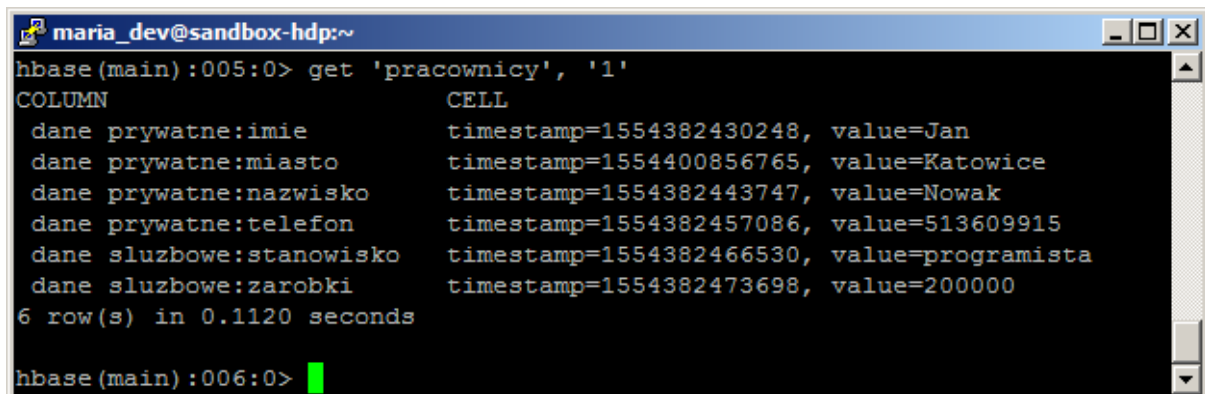
gdzie:

`table_name` – to nazwa tabeli z której będziemy odczytywać dane,

`row_id` – to identyfikator wiersza, który chcemy odczytać.

Przykład:

Wyświetlmy pierwszy wiersz wpisując: `get 'pracownicy', '1'`



```
maria_dev@sandbox-hdp:~  
hbase(main):005:0> get 'pracownicy', '1'  
COLUMN                                CELL  
dane prywatne:imie                    timestamp=1554382430248, value=Jan  
dane prywatne:miasto                  timestamp=1554400856765, value=Katowice  
dane prywatne:nazwisko                timestamp=1554382443747, value=Nowak  
dane prywatne:telefon                 timestamp=1554382457086, value=513609915  
dane sluzbowe:stanowisko               timestamp=1554382466530, value=programista  
dane sluzbowe:zarobki                 timestamp=1554382473698, value=200000  
6 row(s) in 0.1120 seconds  
hbase(main):006:0>
```

Odczyt konkretnej kolumny przy użyciu HBase Shell:

Składnia:

`get 'table_name', 'row_id', {COLUMN => 'column_family:column_name'}`

gdzie:

`table_name` – to nazwa tabeli z której będziemy odczytywać dane,

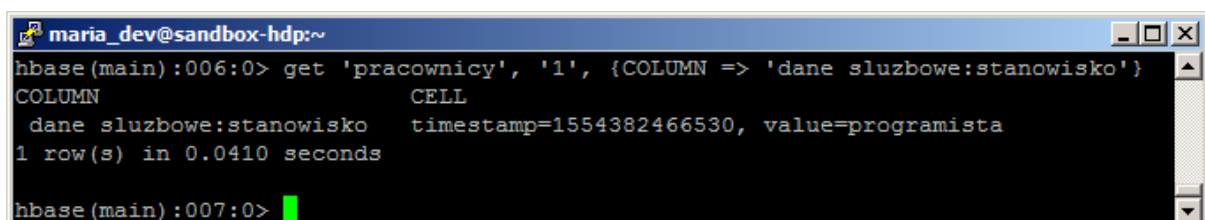
`row_id` – to identyfikator wiersza, który chcemy odczytać,

`column_family` – to rodzina kolumn z której chcemy odczytać dane,

`column_name` – to nazwa kolumny z której chcemy odczytać dane.

Przykład:

Odczytajmy stanowisko wpisując: `get 'pracownicy', '1', {COLUMN => 'dane sluzbowe:stanowisko'}`



```
maria_dev@sandbox-hdp:~  
hbase(main):006:0> get 'pracownicy', '1', {COLUMN => 'dane sluzbowe:stanowisko'}  
COLUMN                                CELL  
dane sluzbowe:stanowisko              timestamp=1554382466530, value=programista  
1 row(s) in 0.0410 seconds  
hbase(main):007:0>
```

Usuwanie określonej komórki w tabeli przy użyciu HBase Shell:

Składnia:

`delete 'table_name', 'row_id', 'column_family:column_name'`

gdzie:

`table_name` – to nazwa tabeli z której będziemy usuwać dane,

`row_id` – to identyfikator wiersza,

`column_family` – to rodzina kolumn z której chcemy usunąć dane,

`column_name` – to nazwa kolumny którą chcemy usunąć.

Przykład:

Usuń miasto wpisując: `delete 'pracownicy', '1', 'dane prywatne:miasto'`

```
maria_dev@sandbox-hdp:~  
hbase(main):008:0> delete 'pracownicy', '1', 'dane prywatne:miasto'  
0 row(s) in 0.0720 seconds
```

Sprawdź czy kolumna została usunięta wpisując: `scan 'pracownicy'`

```
maria_dev@sandbox-hdp:~  
hbase(main):009:0> scan 'pracownicy'  
ROW COLUMN+CELL  
1 column=dane prywatne:imie, timestamp=1554382430248, value=Jan  
1 column=dane prywatne:nazwisko, timestamp=1554382443747, value=Nowak  
1 column=dane prywatne:telefon, timestamp=1554382457086, value=513609915  
1 column=dane sluzbowe:stanowisko, timestamp=1554382466530, value=programista  
1 column=dane sluzbowe:zarobki, timestamp=1554382473698, value=200000  
1 row(s) in 0.0400 seconds  
hbase(main):010:0> █
```

Usuwanie wiersza w tabeli przy użyciu HBase Shell:

Składnia:

`deleteall 'table_name', 'row_id'`

gdzie:

`table_name` – to nazwa tabeli z której chcemy usunąć wiersz,

`row_id` – to identyfikator wiersza, który chcemy usunąć.

Przykład:

Usuń wiersz wpisując: `deleteall 'pracownicy', '1'`

```
maria_dev@sandbox-hdp:~  
hbase(main):011:0> deleteall 'pracownicy', '1'  
0 row(s) in 0.0110 seconds  
hbase(main):012:0> █
```

Zliczanie liczby wierszy przy użyciu HBase Shell:

Składnia:

```
count 'table_name'
```

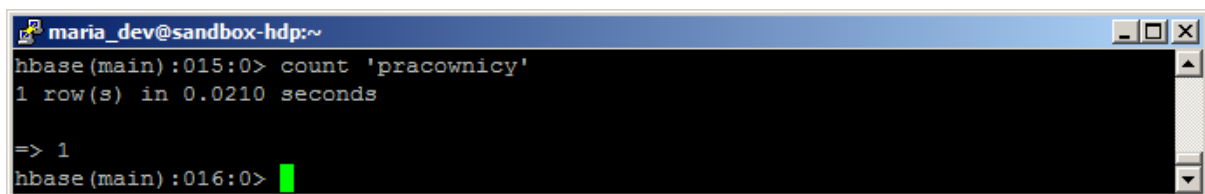
gdzie:

table_name – to nazwa tabeli w której chcemy zliczyć wiersze.

Przykład:

Dodaj dane do tabeli pracownicy wpisując: `put 'pracownicy', '1', 'dane prywatne:imie', 'Jan'`

Zlicz wiersze w tabeli pracownicy wpisując: `count 'pracownicy'`

A screenshot of a terminal window titled 'maria_dev@sandbox-hdp:~'. The terminal shows the HBase Shell prompt 'hbase(main):015:0>'. The user enters the command 'count 'pracownicy''. The output is '1 row(s) in 0.0210 seconds'. The prompt then changes to '=> 1' and back to 'hbase(main):016:0>' with a green cursor.

```
maria_dev@sandbox-hdp:~
hbase(main):015:0> count 'pracownicy'
1 row(s) in 0.0210 seconds
=> 1
hbase(main):016:0> █
```

Wyłączanie, kasowanie i odtwarzanie określone tabeli:

Składnia:

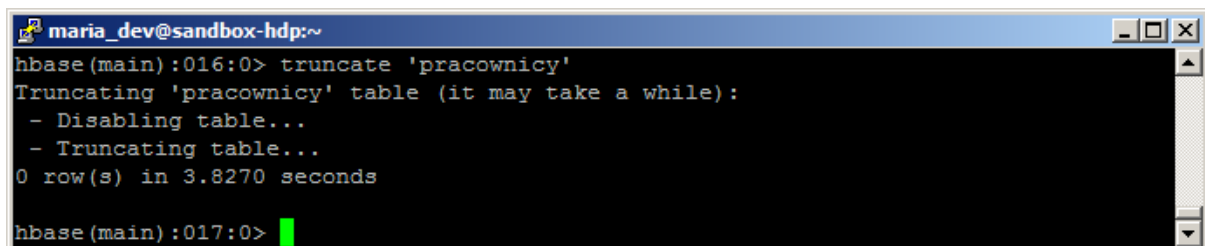
```
truncate 'table_name'
```

gdzie:

table_name – to nazwa tabeli.

Przykład:

Wpisz: `truncate 'pracownicy'`

A screenshot of a terminal window titled 'maria_dev@sandbox-hdp:~'. The terminal shows the HBase Shell prompt 'hbase(main):016:0>'. The user enters the command 'truncate 'pracownicy''. The output is 'Truncating 'pracownicy' table (it may take a while):', followed by two lines: '- Disabling table...' and '- Truncating table...'. The output then shows '0 row(s) in 3.8270 seconds'. The prompt then changes to 'hbase(main):017:0>' with a green cursor.

```
maria_dev@sandbox-hdp:~
hbase(main):016:0> truncate 'pracownicy'
Truncating 'pracownicy' table (it may take a while):
- Disabling table...
- Truncating table...
0 row(s) in 3.8270 seconds
hbase(main):017:0> █
```