

# HBase

Igor Yakushin  
ivy2@uchicago.edu

December 9, 2019

- **H**adoop data**B**ase
- Consists of tables
- Each table is sparse, distributed, persistent, multidimensional sorted map, indexed by rowkey, column family, column, timestamp
- Can store structured, semistructured, unstructured data
- Does not care about types
- Not a relational database, does not speak SQL natively, does not enforce relationship in data
- Designed to run on a cluster of computers, scale horizontally as you add more machines to the cluster
- The main operations are: create (table), put (value into cell), get (value from cell), scan (values from cells)
- Various auxiliary operations: alter, list, describe, ...

```
Row Key      Column Family: {Column Qualifier:Version:Value}
-----
00001      CustomerName: {'FN': 1383859182496:'John',
               'LN': 1383859182858:'Smith',
               'MN': 1383859183001:'Timothy',
               'MN': 1383859182915:'T'}
           ContactInfo: {'EA': 1383859183030:'John.Smith@xyz.com',
                          'SA': 1383859183073:'1 Hadoop Lane, NY 11111'}
00002      CustomerName: {'FN': 1383859183103:'Jane',
               'LN': 1383859183163:'Doe',
           ContactInfo: {'SA': 1383859185577:'7 HBase Ave, CA 22222'}
```

- Internally HBase table is stored in **HFiles** - different set of files for different column families
- HFiles for the same column family are periodically merged together or split and distributed among the nodes to maintain high performance and fault-tolerance
- On top of HDFS
- One can specify how many latest versions of data to keep in HBase table or to query versions in a particular date-time range

# HBase: shell, create, put, list, describe

```
$ hbase shell
```

```
> create 'CustomerContactInfo', 'CustomerName', 'ContactInfo'
> put 'CustomerContactInfo', '00001', 'CustomerName:FN', 'John'
> put 'CustomerContactInfo', '00001', 'CustomerName:LN', 'Smith'
> put 'CustomerContactInfo', '00001', 'CustomerName:MN', 'T'
> put 'CustomerContactInfo', '00001', 'ContactInfo:EA', 'John.Smith@xyz.com'
> put 'CustomerContactInfo', '00001', 'ContactInfo:SA', '1 Hadoop Lane, NY 11111'
> put 'CustomerContactInfo', '00002', 'CustomerName:FN', 'Jane'
> put 'CustomerContactInfo', '00002', 'CustomerName:LN', 'Doe'
> put 'CustomerContactInfo', '00002', 'ContactInfo:SA', '7 HBase Ave, CA 22222'
> list
=> ["CustomerContactInfo"]
> describe 'CustomerContactInfo'
Table CustomerContactInfo is ENABLED
CustomerContactInfo
COLUMN FAMILIES DESCRIPTION
{NAME => 'ContactInfo', DATA_BLOCK_ENCODING => 'NONE', BLOOMFILTER => 'ROW',
REPLICATION_SCOPE => '0', VERSIONS => '1', COMPRESSION => 'NONE',
MIN_VERSIONS => '0', TTL => 'FOREVER', KEEP_DELETED_CELLS => 'FALSE',
BLOCKSIZE => '65536', IN_MEMORY => 'false', BLOCKCACHE => 'true'}
{NAME => 'CustomerName', DATA_BLOCK_ENCODING => 'NONE', BLOOMFILTER => 'ROW',
REPLICATION_SCOPE => '0', VERSIONS => '1', COMPRESSION => 'NONE', MIN_VERSIONS => '0',
TTL => 'FOREVER', KEEP_DELETED_CELLS => 'FALSE', BLOCKSIZE => '65536',
IN_MEMORY => 'false', BLOCKCACHE => 'true'}
```

# HBase: alter, scan, VERSIONS

```
> alter 'CustomerContactInfo', NAME => 'CustomerName', VERSIONS => 5
> describe 'CustomerContactInfo'
Table CustomerContactInfo is ENABLED
CustomerContactInfo
COLUMN FAMILIES DESCRIPTION
{NAME => 'ContactInfo', DATA_BLOCK_ENCODING => 'NONE', BLOOMFILTER => 'ROW',
REPLICATION_SCOPE => '0', VERSIONS => '1', COMPRESSION => 'NONE',
MIN_VERSIONS => '0', TTL => 'FOREVER', KEEP_DELETED_CELLS => 'FALSE',
BLOCKSIZE => '65536', IN_MEMORY => 'false', BLOCKCACHE => 'true'}
{NAME => 'CustomerName', DATA_BLOCK_ENCODING => 'NONE', BLOOMFILTER => 'ROW',
REPLICATION_SCOPE => '0', VERSIONS => '5', COMPRESSION => 'NONE',
MIN_VERSIONS => '0', TTL => 'FOREVER', KEEP_DELETED_CELLS => 'FALSE',
BLOCKSIZE => '65536', IN_MEMORY => 'false', BLOCKCACHE => 'true'}
> put 'CustomerContactInfo', '00001', 'CustomerName:MN', 'Timothy'
> scan 'CustomerContactInfo', {VERSIONS => 2}
ROW      COLUMN+CELL
00001    column=ContactInfo:EA, timestamp=1471196578957, value=John.Smith@xyz.com
00001    column=ContactInfo:SA, timestamp=1471196578988, value=1 Hadoop Lane, NY 11111
00001    column=CustomerName:FN, timestamp=1471196578805, value=John
00001    column=CustomerName:LN, timestamp=1471196578859, value=Smith
00001    column=CustomerName:MN, timestamp=1471197270641, value=Timothy
00001    column=CustomerName:MN, timestamp=1471196578901, value=T
00002    column=ContactInfo:SA, timestamp=1471196579070, value=7 HBase Ave, CA 22222
00002    column=CustomerName:FN, timestamp=1471196579016, value=Jane
00002    column=CustomerName:LN, timestamp=1471196579042, value=Doe
```

# HBase: get, disable, drop, quit

```
> get 'CustomerContactInfo', '00001'
COLUMN                                CELL
ContactInfo:EA                        timestamp=1471196578957, value=John.Smith@xyz.com
ContactInfo:SA                        timestamp=1471196578988, value=1 Hadoop Lane, NY 11111
CustomerName:FN                       timestamp=1471196578805, value=John
CustomerName:LN                       timestamp=1471196578859, value=Smith
CustomerName:MN                       timestamp=1471197270641, value=Timothy
> get 'CustomerContactInfo', '00001', {COLUMN => 'CustomerName:MN'}
COLUMN                                CELL
CustomerName:MN                       timestamp=1471197270641, value=Timothy
> disable 'CustomerContactInfo'
> drop 'CustomerContactInfo'
> quit
```

Besides hbase shell, one can use HBase with

- MapReduce
- Hive
- Pig
- Spark
- Impala