06: Q51 – Q61 HBase Interview Questions & Answers

Posted on June 8, 2016 by Arulkumaran Kumaraswamipillai

Q51. Is HBase a relational database?
A51. HBase is not a relational database. It is a **NoSQL** database. Hbase is a **column-oriented** (aka columnar) database management system, which runs on top of **HDFS** (Hadoop Distribute File System).

Q52. What does HBase consist of? Why is it called a columnar & key/value pair oriented database?
A52. Set of **tables**, and each table contains **rows** and **columns** like a traditional database, each table must contain a "**row key**" as a primary key, columns are basically name/value pairs grouped as "column families". Each row of a column family can have variable number of name/value pairs. It is like having a map within a "**column family**".

600+ Full Stack Java/JEE Interview Q&As ♥Free ♦FAQs

open all | close all

- Ice Breaker Interview
- Core Java Interview C
- JEE Interview Q&A (3
- Pressed for time? Jav
- SQL, XML, UML, JSC
- Hadoop & BigData Int
  - ♥ 01: Q1 – Q6 Had
  - 02: Q7 – Q15 Hadc
  - 03: Q16 – Q25 Hac
  - 04: Q27 – Q36 Apa
  - 05: Q37 – Q50 Apa
  - 05: Q37-Q41 – Dat
  - 06: Q51 – Q61 HBa
  - 07: Q62 – Q70 HDI
- Java Architecture Inte
- Scala Interview Q&As
- Spring, Hibernate, & M
- Testing & Profiling/Sa
- Other Interview Q&A f
- ▶Free Java Interviev

Start the HBase shell

```
$ hbase shell
```

Create a table '**eai_systems**' with the "column families". You can alter the table to add more column families later on. It is a best practice to keep the column family names as short as possible like "i" for info, "m" for messages, "a" for archival info, etc.

```
hbase(main):006:0>create 'eai_systems', 'i', 'a'
```

Let's insert data into the "**eai_systems**" table. The following example inserts "row key" like 222, 555, etc followed by key/value pairs for a column family.

Row key: 222

i = **coumn family**
target_system = **name**
PayRoll= **value**

```
put 'eai_systems', '222', 'i:target_system', 'Pay
put 'eai_systems', '222', 'i:request_destination'
put 'eai_systems', '222', 'i:request_path', '/mya
put 'eai_systems', '222', 'a:time_to_live_in_days
```

Row key: 555

As you can see, the column family can have variable number of name/value pairs. The row key 555 has additional name/value pairs of invoice_id and customer_id.

```
put 'eai_systems', '555', 'i:target_system', 'Inv
put 'eai_systems', '555', 'i:request_destination'
put 'eai_systems', '555', 'i:request_path', '/mya
```

```
5 put 'eai_systems', '555', 'i:invoice_id', 'INV234
6 put 'eai_systems', '555', 'i:customer_id', 'CUST1
7
```

Specifying only the **row key** can potentially return a ton of data because an individual row can have millions of columns in a columnar database. A key-value pair can look like this

RowKey:**Column Family**:**Column Qualifier**:**Version** => Value

```
1
2 555:i:customer_id:1383859182915 => 'CUST123'
3
```

The versions are implemented using time stamps by default and are sorted in decreasing order so that you automatically get the most recent value if you don't specify a version.

Q53. Why use an HBase database?
A53. HBase is a distributed database to cater for very large tables. The column oriented storage is horizontally salable. The design goal of HBase is to have millions of columns, thousands of versions (i.e timestamped) and billions of rows.

Q54. Are there data types in HBase?
A54. No types in HBase data values. The values are in "**byte array**".

Q55. When should you use HBase?
A55. HBase is great at many things, but it doesn't replace relational databases. HBase is not optimized for classic transactional applications. It doesn't talk SQL and does not support cross record transactions or joins.

1. If your application has a **variable schema** where each row is slightly different, then you should look at HBase.

2. If you have data collections **keyed** on the same key value.

3. If you need key based access to data when storing or retrieving, then you should consider HBase.

Q56. What are the key differences between HDFS & HBase?
A56. HDFS files as **write-once** and **read-many** times files. There is no concept of random reads & writes.

HBase is a database that stores data in the distributed file system (i.e HDFS). HBase data are indexed by "**row key**", and you can access a single row very quickly from billions of rows. You can read data randomly & sequentially.

Q57. Can you create a HBase table without assigning a column family?
A57. No. Column families determine how the data should be physically stored in the HDFS file system. So, you should at least have 1 **column family**.

Q58. What commands do you use to interact with HBase?
A58. Get, Put, Delete, Scan, and Increment.

```
1
2 hbase(main):020:0> get 'eai_systems', '222'
3
```

```
1
2 COLUMN                          CELL
3 a:time_to_live_in_days          timestamp=14653419583
4 i:request_destination           timestamp=14653419417
5 i:request_path                  timestamp=14653419490
6 i:target_system                 timestamp=14653419337
7
8 4 row(s) in 0.0600 seconds
9
```

To get a specific column

```
1
2 hbase(main):037:0> get 'eai_systems', '222', {COL
3
```

```
1
2 COLUMN                          CELL
3
4 i:target_system                 timestamp=14653419337
5
6 1 row(s) in 0.0190 seconds
7
```

Scan the table

```
hbase(main):016:0> scan 'eai_systems'
```

```
ROW                                  COLUMN+CELL
222                                 column=a:time_to_liv
222                                 column=i:request_des
222                                 column=i:request_pat
222                                 column=i:target_syst
222                                 column=m:count, time
555                                 column=i:customer_id
555                                 column=i:invoice_id,
555                                 column=i:request_des
555                                 column=i:request_pat
555                                 column=i:target_syst
555                                 column=m:count, time

2 row(s) in 0.0500 seconds
```

Q59. How does HBase ensure data durability?
A59. HBase writes go to 2 places

1) The write-ahead log (**WAL**), which is also known as the HLog.

2) The **MemStore**, which is a write buffer where HBase accumulates data in memory before a permanent write. The data are flushed to disk to form an **HFile** when the MemStore fills up. HFile is the underlying storage format for HBase. There will be one MemStore per column family.

Q. What happens if the server hosting the MemStore crashes before the contents are flushed to the disk?
A. HBase cluster keeps a **WAL** to record changes as they happen. The WAL is a file on the underlying file system. A write isn't considered successful until a new WAL entry is successfully written. This gives HBase durability.

Q60. Can you explain the terms tombstone records, major compaction, and minor compaction?
A60. The "**delete**" command doesn't immediately physically delete a value. It logically marks the record for deletion by

creating a new "**tombstone**" record. The tombstone record is used to indicate that the deleted value should no longer be included in **Get** or **Scan** results.

During the "major compaction" the tombstone records are reconciled and space is truly recovered from deleted records. In other words, the records are physically deleted.

During the "minor compaction", larger HFiles are created by combining smaller HFiles.

Q61. Where do the HBase master & region servers run?
A61. **HMaster** is the implementation of the Master Server, and is responsible for monitoring all RegionServer instances in the cluster. In a distributed cluster, the Master typically runs on the **NameNode**.

HRegionServer is the implementation of region servers. Region servers are responsible for serving and managing regions. In a distributed cluster, a region server runs on a **DataNode**.

Popular Posts

♦ 11 Spring boot interview questions & answers

828 views

♦ Q11-Q23: Top 50+ Core on Java OOP Interview Questions & Answers

768 views

18 Java scenarios based interview Questions and Answers

401 views

001A: ♦ 7+ Java integration styles & patterns interview questions & answers

389 views

01b: ♦ 13 Spring basics Q8 – Q13 interview questions & answers

296 views

♦ 7 Java debugging interview questions & answers

293 views

01: ♦ 15 Ice breaker questions asked 90% of the time in Java job interviews with hints

286 views

♦ 10 ERD (Entity-Relationship Diagrams) Interview Questions and Answers

280 views

♦ Q24-Q36: Top 50+ Core on Java classes, interfaces and generics interview questions & answers

240 views

001B: ♦ Java architecture & design concepts interview questions & answers

202 views

Bio | Latest Posts

Arulkumaran Kumaraswamipillai

Mechanical Eng to freelance Java developer in 3 yrs. Contracting since 2003, and attended 150+ Java job interviews, and often got 4 - 7 job offers to choose from. It pays to prepare. So, published Java interview Q&A books via Amazon.com in 2005, and sold 35,000+ copies. Books are outdated and replaced with this subscription based site.**945+** paid members. join my LinkedIn Group. **Reviews**

About Arulkumaran Kumaraswamipillai

Mechanical Eng to freelance Java developer in 3 yrs. Contracting since 2003, and attended 150+ Java job interviews, and often got 4 - 7 job offers to choose from. It pays to prepare. So, published Java interview Q&A books via Amazon.com in 2005, and sold 35,000+ copies. Books are outdated and replaced with this subscription based site.**945+** paid members. join my LinkedIn Group. **Reviews**

‹ 03: Spark submit – reading a Sequence File from HDFS

02: Scenarios based Java OO concepts & GoF design patterns – naive Template Method pattern ›

Posted in Hadoop & BigData Interview Q&A, member-paid

Empowers you to open more doors, and fast-track

Technical Know Hows

☀ Java generics in no time ☀ Top 6 tips to transforming your thinking from OOP to FP ☀ How does a HashMap internally work? What is a hashing function?
☀ 10+ Java String class interview Q&As ☀ Java auto un/boxing benefits & caveats ☀ Top 11 slacknesses that can come back and bite you as an experienced Java developer or architect

Non-Technical Know Hows

☀ 6 Aspects that can motivate you to fast-track your career & go places ☀ Are you reinventing yourself as a Java developer? ☀ 8 tips to safeguard your Java career against offshoring ☀ My top 5 career mistakes

Prepare to succeed

☀ Turn readers of your Java CV go from “Blah blah” to “Wow”? ☀ How to prepare for Java job interviews? ☀ 16 Technical Key Areas ☀ How to choose from multiple Java job offers?

Select Category ▼

© Disclaimer