Java-Success.com

Prepare to fast-track, choose & go places with 800+ Java & Big Data Q&As with lots of code & diagrams.

06: Avro Schema evolution tutorial

Posted on December 18, 2016

Q1. What do you understand by the term “AVRO **schema evolution**“?
A1. Schema evolution is the term used for how the store behaves when Avro schema is changed after data has been written to the store using an older version of that schema.

Q2. When does the schema evolution take place?
A2. During **deserialization**. In other words, when reading an avro file that was written with an older schema, and you can read it with a newer (i.e. evolved) schema provided you have defined the **default values** in your schema.

300+ Java Interview FAQs

300+ Java FAQs 🔥
16+ Java Key Areas Q&As
150+ Java Architect FAQs
80+ Java Code Quality Q&As
150+ Java Coding Q&As

300+ Big Data Interview FAQs

300+ Big Data FAQs 🔥
Tutorials - Big Data
TUT - ⏯ Starting Big Data
TUT - Starting Spark & Scala

employee-ver1.avsc

```
{
    "type": "record",
    "namespace": "com.myapp",
    "name": "organization",
    "fields": [
        {
            "name": "name",
            "type": "string",
            "doc": "The employee name",
            "default": null
        }
        ,
        {
            "name": "title",
            "type": "string",
            "doc": "employee title",
            "default": null
        }

     ]
}

```

employee-ver2.avsc

The schema has evolved with an additional field named “salary”. It defines a “default” as “null”.

```

{
    "type": "record",
    "namespace": "com.myapp",
    "name": "organization",
    "fields": [
        {
            "name": "name",
            "type": "string",
            "doc": "The employee name",
            "default": null
        }

        ,
        {
            "name": "title",
            "type": "string",
            "doc": "employee title",
```

TUT - Starting with Python
TUT - Kafka
TUT - Pig
TUT - Apache Storm
TUT - Spark Scala on Zeppelin
TUT - Cloudera
TUT - Cloudera on Docker
TUT - File Formats
TUT - Spark on Docker
TUT - Flume
TUT - Hadoop (HDFS)
TUT - HBase (NoSQL)
TUT - Hive (SQL)
TUT - Hadoop & Spark
TUT - MapReduce
TUT - Spark and Scala
TUT - Spark & Java
TUT - PySpark on Databricks
TUT - Zookeeper

```
            "default": null
        }
        ,
        {
           "name": "salary",
            "type": ["null", "double" ],
            "doc": "employee salary",
            "default": null
        }
    ]
}

```

Java code to write/read to/from sequence file

It initially rites and reads with “employee-**ver1**.avsc”, and then with evolved schema “employee-**ver2**.avsc”.

```

package com.mytutorial;

import java.io.IOException;
import java.net.URI;
import java.net.URL;

import org.apache.avro.Schema;
import org.apache.avro.file.DataFileReader;
import org.apache.avro.file.DataFileWriter;
import org.apache.avro.file.FileReader;
import org.apache.avro.file.SeekableInput;
import org.apache.avro.generic.GenericData;
import org.apache.avro.generic.GenericDatumReade
import org.apache.avro.generic.GenericDatumWrite
import org.apache.avro.generic.GenericRecord;
import org.apache.avro.io.DatumReader;
import org.apache.avro.mapred.FsInput;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.FSDataOutputStream;
import org.apache.hadoop.fs.FileSystem;
import org.apache.hadoop.fs.Path;

public class CreateOrAppendToAvroOnHdfs {

    public static void main(String[] args) {

        Configuration hdfsConf = new Configurati

        try {
            String uri = "hdfs://localhost:8020/
```

```
32              FileSystem hdfs = FileSystem.get(URI
33
34
35              Path filePath = new Path(uri);
36              hdfs.setReplication(filePath, (short
37
38              URL avroUrlSchemaVersion1 = CreateOr
39              Schema avroSchemaVersion1 = new Sche
40
41              writeToSequenceFile(hdfs, filePath, 
42              readFromSequenceFile(filePath, hdfsC
43
44
45              URL avroUrlSchemaVersion2 = CreateOr
46              Schema avroSchemaVersion2 = new Sche
47
48              readFromSequenceFile(filePath, hdfsC
49
50
51
52          } catch (IOException ex) {
53              ex.printStackTrace();
54          } finally {
55          }
56      }
57
58      private static void writeToSequenceFile(File
59
60          DataFileWriter<GenericRecord> fileWriter
61          DataFileWriter<GenericRecord> appendFile
62          FSDataOutputStream out = null;
63
64          GenericDatumWriter<GenericRecord> datumW
65          fileWriter = new DataFileWriter<GenericR
66
67
68          if (!hdfs.exists(filePath)) {
69              // If the path doesn't exist, create
70              out = hdfs.create(filePath);
71              appendFileWriter = fileWriter.create
72
73          } else {
74              // Otherwise just append to the exis
75              out = hdfs.append(filePath);
76              appendFileWriter = fileWriter.append
77          }
78
79           GenericRecord myRecord = new GenericDat
80           myRecord.put("name", "John");
81           myRecord.put("title", "VP");
82
83
84          appendFileWriter.append(myRecord);
85          appendFileWriter.close();
86          fileWriter.close();
```

```
87      }
88
89      private static void readFromSequenceFile(Patl
90          SeekableInput input = new FsInput(filePa
91          DatumReader<GenericRecord> reader = new (
92          FileReader<GenericRecord> fileReader = Da
93
94          for (GenericRecord datum : fileReader) {
95              System.out.println("value = " + datu
96          }
97
98          fileReader.close();
99      }
100 }
101
```

Output

```
1
2 value = {"name": "John", "title": "VP"}
3 value = {"name": "John", "title": "VP", "salary":
4
```

Q3. What schema modifications can you safely perform?

A3.

1) A field with a default value is added, and a field that was previously defined with a default value is removed.

2) A field's doc or order attribute is changed, added or removed.

3) A field's default value is added, or changed.

4) Field or type aliases are added, or removed.

5) A non-union type may be changed to a union that contains only the original type, or vice-versa.

‹ 01b: Convert XML file To Sequence File – writing & reading – Hadoop File System (i.e HDFS)

10: Q80 – Q87 HBase Schema Design Interview Q&As ›

Disclaimer