Hadoop

Форматы хранения данных

Типы хранения данных

* txt (обычный текстовый файл)
* SequenceFile (двоичный типа key-value)
* Avro (двоичный от Дуга Каттинга)
* Protobuf (двоичный от Google)
* ORC (колоночный)

Текстовый формат txt

* Плюсы:
  * простой
  * понятный
  * хорошо сжимается
  * легко обрабатывать на любом ЯП

Текстовый формат txt

* Минусы:
  * неэффективное использование диска
  * нет схемы
  * нет сериализации (надо делать)
  * Кодировки

SequenceFile

* Самый первый двоичный формат
* Хранит все в виде ключ - значение
* Поддержка компрессии
  * уровня блока
  * уровня файла

SequenceFile

структура

* Header
* Record
    * Record length
    * Key length
    * Key
    * (Compressed?) Value
    * A sync-marker every few k bytes or so.

SequenceFile

структура header

* version - SEQ4 или SEQ6
* keyClassName - класс для ключа
* valueClassName - класс для значения
* compression - флаг компресии
* blockCompression - флаг блочной компресии
* compressor class - кодак для компрессии
* metadata - метаданные
* sync - маркер

Avro

* Двоичный
* Есть схема
* Есть стандартная сериализация
* Хорошо сжимается
* Высокая производительность
* Поддержка большинства ЯП
* Самоописательный

Avro schema

```
{
  "namespace": "ru.mail.avro",
  "name": "SvdUid",
  "type": "record",
  "fields": [
    {
      "name": "uid_type",
      "type": "string"
    },
    {
      "name": "uid",
      "type": "string"
    }
  ]
}
```

Avro типы данных

* null: пусто
* boolean: двоичное
* int: 32-bit целое
* long: 64-bit целое
* float: 32-bit число с плавающей точкой
* double: 64-bit число с плавающей точкой
* bytes: массив байт
* string: строка в юникоде

Avro типы данных

* records
* enum
* arrays
* maps
* unions
* fixed

Avro enum

```
{
 "doc": "User type",
 "name": "user_type",
 "type": {
  "name": "EXPRESS_USER_TYPES",
  "type": "enum",
  "symbols": [
   "_1POSITIVE",
   "_2NEGATIVE"
  ]
 },
 "order": "ascending"
}
```

Avro array

```
{
  "doc": "Features",
  "name": "features",
  "type": {"type": "array", "items": "double"},
  "order": "ignore"
}
```

Avro map

```
{
  "name": "step",
  "type": {
    "type": "map",
    "values": "string"
  }
}
```

Avro union

```
{
  "name": "photo_big",
  "type": ["string", "null"]
}
```

Avro schema plugin

```
<plugin>
    <groupId>org.apache.avro</groupId>
    <artifactId>avro-maven-plugin</artifactId>
    <version>${org.apache.avro.cdh.version}</version>
    <executions>
      <execution>
        <phase>generate-sources</phase>
        <goals>
          <goal>schema</goal>
        </goals>
```

Avro schema plugin

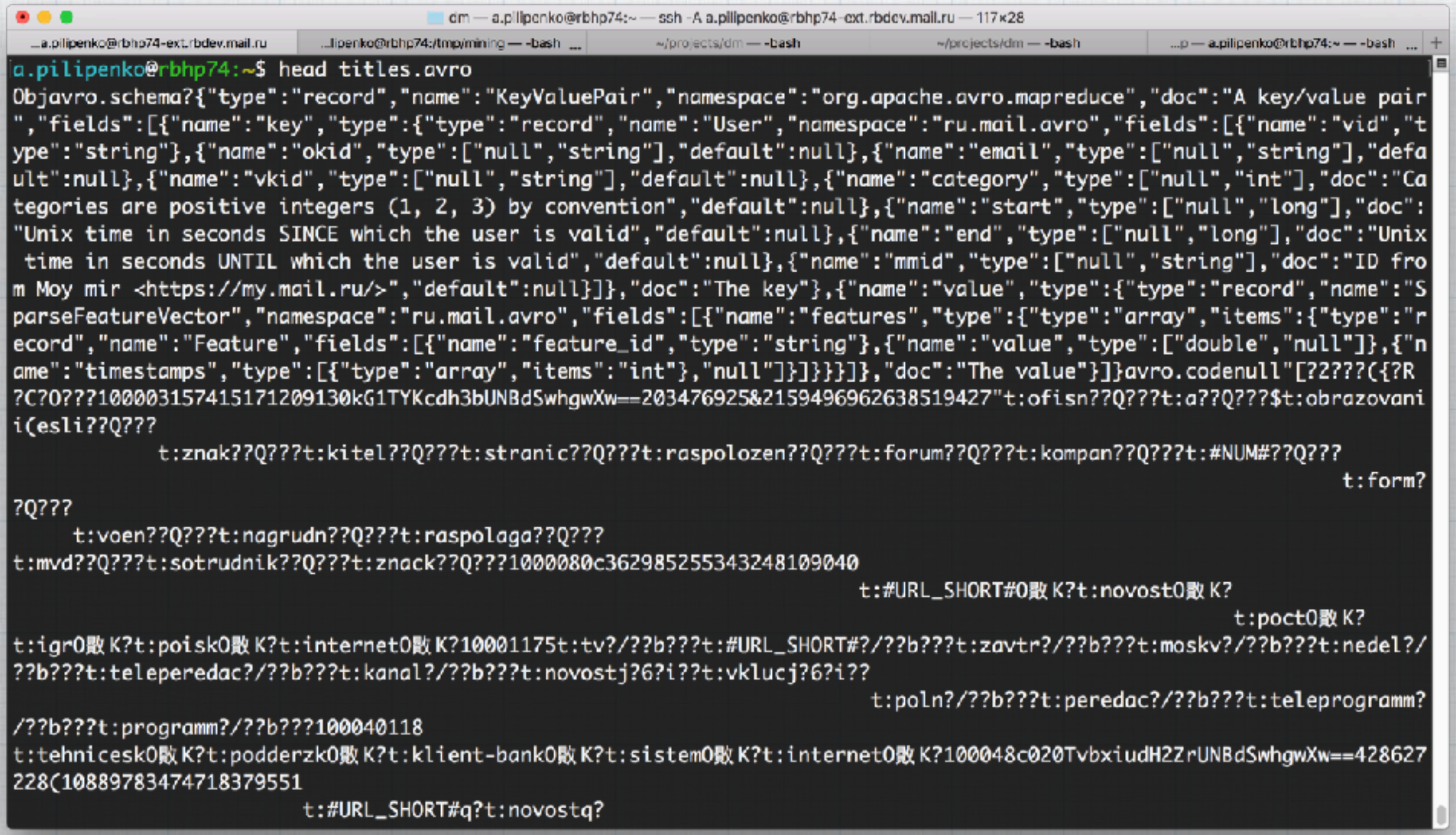
```
<configuration>

        <sourceDirectory>${project.basedir}/src/main/avro/</
sourceDirectory>

        <outputDirectory>${project.basedir}/target/generated-sources/
java/</outputDirectory>

          <imports>
          <import>${project.basedir}/src/main/avro/session.avsc</import>
           /imports>
       </configuration>
    </execution>
  </executions>
</plugin>
```

Avro internals



Avro internals

dm — a.pilipenko@rbhp74:~ — ssh -A a.pilipenko@rbhp74-ext.rbdev.mail.ru — 117×28

a.pilipenko@rbhp74-ext.rbdev.mail.ru | ...lipenko@rbhp74:/tmp/mining — -bash | ~/projects/dm — -bash | ~/projects/dm — -bash | ...p — a.pilipenko@rbhp74:~ — -bash

```
a.pilipenko@rbhp74:~$ hadoop fs -text /data/dm/datasets/2017-12-13/titles/part-r-00044.avro | head
{"key":{"vid":"100003157","okid":{"string":"41517120913"},"email":{"string":"kG1TYKcdh3bUNBdSwhgwXw=="},"vkid":{"stri
ng":"203476925"},"category":{"int":0},"start":null,"end":null,"mmid":{"string":"21594969626385194 27"}},"value":{"feat
ures":[{"feature_id":"t:ofisn","value":{"double":0.165},"timestamps":null},{"feature_id":"t:a","value":{"double":0.16
5},"timestamps":null},{"feature_id":"t:obrazovanii(esli","value":{"double":0.165},"timestamps":null},{"feature_id":"t
:znak","value":{"double":0.165},"timestamps":null},{"feature_id":"t:kitel","value":{"double":0.165},"timestamps":null
},{"feature_id":"t:stranic","value":{"double":0.165},"timestamps":null},{"feature_id":"t:raspolozen","value":{"double
":0.165},"timestamps":null},{"feature_id":"t:forum","value":{"double":0.165},"timestamps":null},{"feature_id":"t:komp
an","value":{"double":0.165},"timestamps":null},{"feature_id":"t:#NUM#","value":{"double":0.165},"timestamps":null},{
"feature_id":"t:form","value":{"double":0.165},"timestamps":null},{"feature_id":"t:voen","value":{"double":0.165},"ti
mestamps":null},{"feature_id":"t:nagrudn","value":{"double":0.165},"timestamps":null},{"feature_id":"t:raspolaga","va
lue":{"double":0.165},"timestamps":null},{"feature_id":"t:mvd","value":{"double":0.165},"timestamps":null},{"feature_
id":"t:sotrudnik","value":{"double":0.165},"timestamps":null},{"feature_id":"t:znack","value":{"double":0.165},"times
tamps":null}]}}
{"key":{"vid":"1000080c","okid":{"string":"362985255343"},"email":null,"vkid":{"string":"248109040"},"category":{"int
":0},"start":null,"end":null,"mmid":null},"value":{"features":[{"feature_id":"t:#URL_SHORT#","value":{"double":8.3333
33333333334E-4},"timestamps":null},{"feature_id":"t:novost","value":{"double":8.333333333333334E-4},"timestamps":null
},{"feature_id":"t:poct","value":{"double":8.333333333333334E-4},"timestamps":null},{"feature_id":"t:igr","value":{"d
ouble":8.333333333333334E-4},"timestamps":null},{"feature_id":"t:poisk","value":{"double":8.333333333333334E-4},"time
stamps":null},{"feature_id":"t:internet","value":{"double":8.333333333333334E-4},"timestamps":null}]}}
{"key":{"vid":"10001175f","okid":null,"email":null,"vkid":null,"category":{"int":0},"start":null,"end":null,"mmid":nu
ll},"value":{"features":[{"feature_id":"t:tv","value":{"double":0.30916666666666665},"timestamps":null},{"feature_id"
:"t:#URL_SHORT#","value":{"double":0.30916666666666665},"timestamps":null},{"feature_id":"t:zavtr","value":{"double":
0.30916666666666665},"timestamps":null},{"feature_id":"t:moskv","value":{"double":0.30916666666666665},"timestamps":n
ull},{"feature_id":"t:nedel","value":{"double":0.30916666666666665},"timestamps":null},{"feature_id":"t:teleperedac",
"value":{"double":0.30916666666666665},"timestamps":null},{"feature_id":"t:kanal","value":{"double":0.309166666666666
65},"timestamps":null},{"feature_id":"t:novost","value":{"double":0.30333333333333334},"timestamps":null},{"feature_i
d":"t:vkluc","value":{"double":0.30333333333333334},"timestamps":null},{"feature_id":"t:poln","value":{"double":0.309
```

Protobuf

(Protocol Buffers)

* Om Google
* Нужен компилятор (надо скачать *)
* Есть схема
* Нет схемы при сериализации *

Protobuf schema

```
syntax="proto2";

package ru.mail.proto;

option java_package = "ru.mail.proto";
option java_outer_classname = "GeoProto";
message LatLonMsg {
    optional double lat = 1;
    optional double lon = 2;
}
```

Protobuf

типы данных

* double
* float
* int32
* int64
* uint32
* uint64
* sint32
* sint64
* fixed32
* fixed64
* sfixed32
* sfixed64
* bool
* string
* bytes

Protobuf

Серилизация

* Magic (несколько байт как маркер)
* Тип сообщения (int - id в репозитории)
* Длина сообщения (int - в байтах)
* Тело сообщения

Protobuf recap

* Хорош для сериализации маленьких сообщений
* Неплохо сжимает
* Неплохая производительность
* Нет стандартного механизма описания
* как следствие - надо писать свой InputFormat

ORC

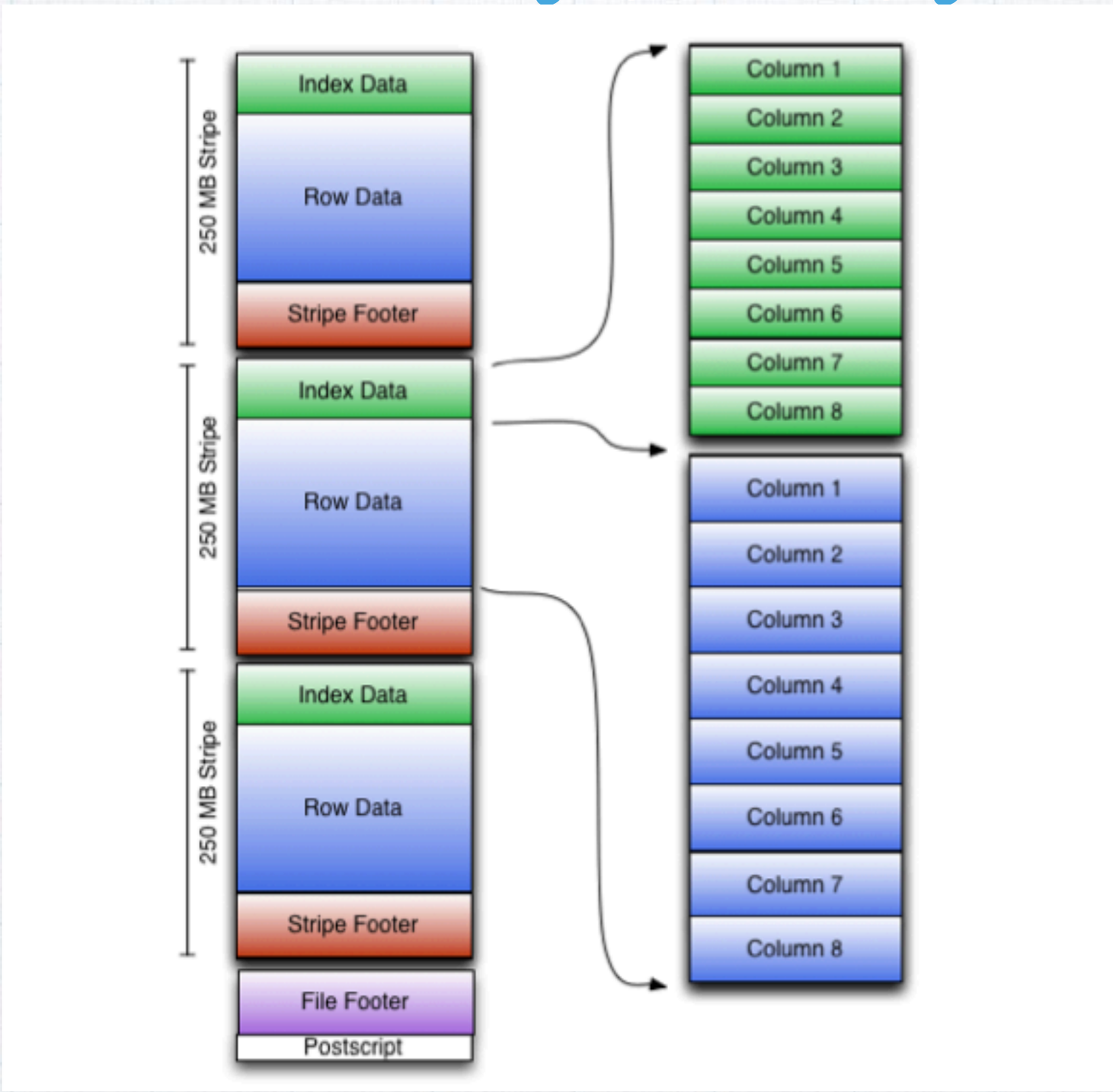
* Колоночный
* Простая интеграция с Hive, Spark
* Оптимизация сериализации комплексных типов
* Можно сплитать без полного сканирования
* Можно эффективно сливать файлы
* Регулируемые параметры потребления памяти для чтения и записи

ORC

типы данных

* Integer
  * boolean (1 bit)
  * tinyint (8 bit)
  * smallint (16 bit)
  * int (32 bit)
  * bigint (64 bit)
* Floating point
  * float
  * double
* String types
  * string
  * char
  * varchar
* Binary blobs
  * binary
* Date/time
  * timestamp
  * date
* Compound types
  * struct
  * list
  * map
  * union

ORC структура



ORC postscript

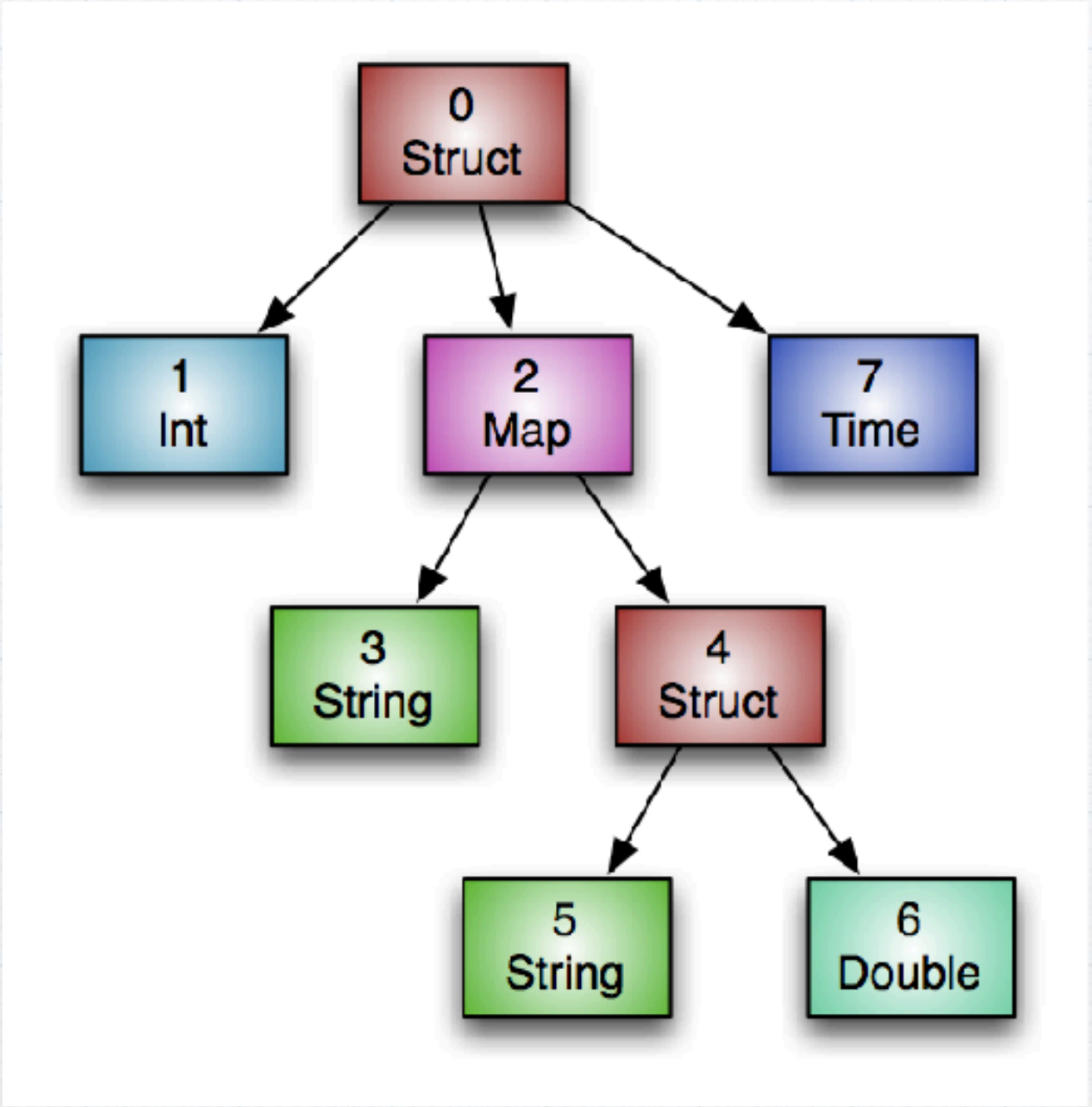
```
message PostScript {
 // the length of the footer section in bytes
 optional uint64 footerLength = 1;
 // the kind of generic compression used
 optional CompressionKind compression = 2;
 // the maximum size of each compression chunk
 optional uint64 compressionBlockSize = 3;
 // the version of the writer
 repeated uint32 version = 4 [packed = true];
 // the length of the metadata section in bytes
 optional uint64 metadataLength = 5;
 // the fixed string "ORC"
 optional string magic = 8000;
}
```

```
enum CompressionKind {
 NONE = 0;
 ZLIB = 1;
 SNAPPY = 2;
 LZO = 3;
 LZ4 = 4;
 ZSTD = 5;
}
```

ORC file footer

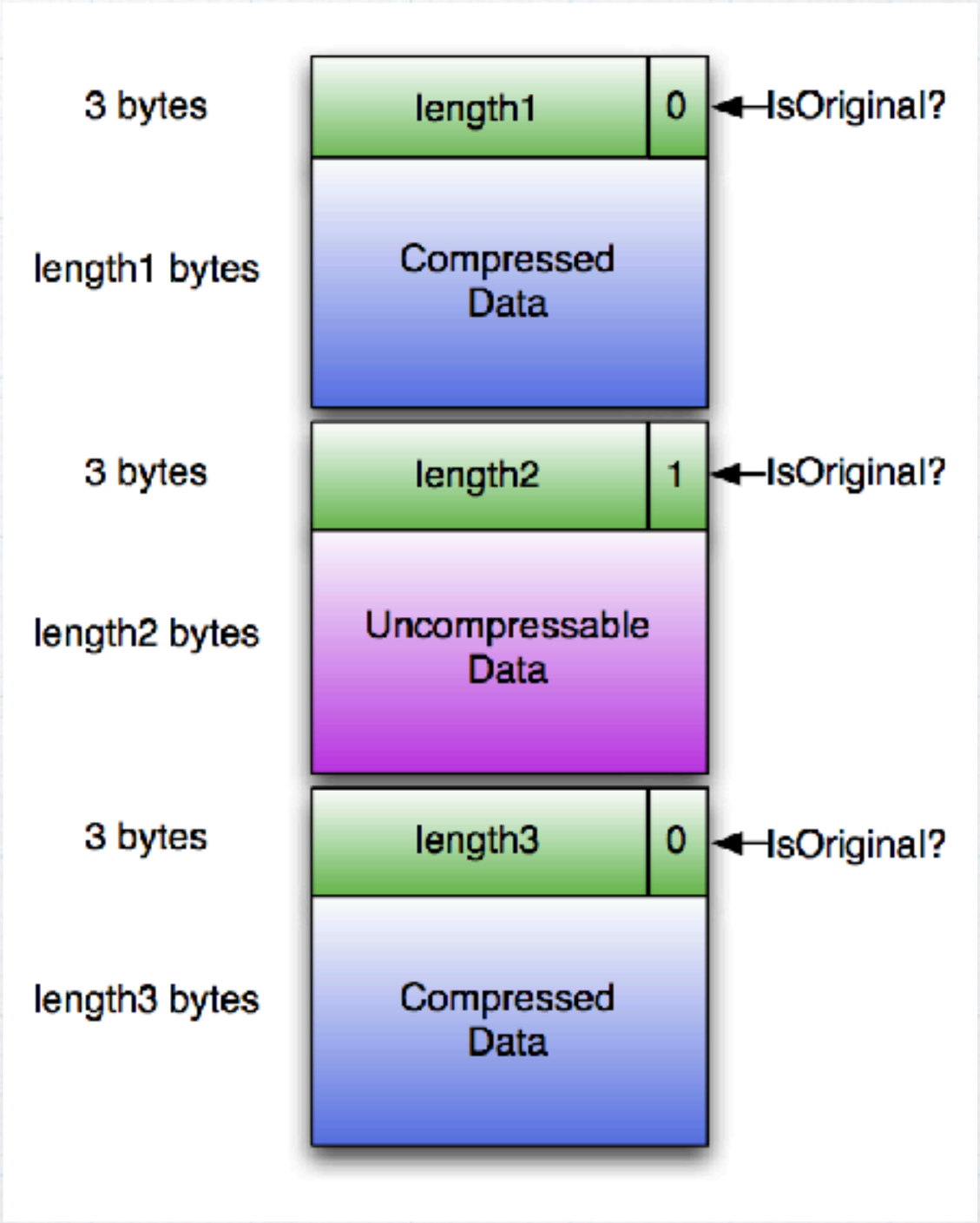
* Список stripe-ов
* Количество строк в каждом стрипе
* Типы данных всех полей
* Индексы по полям (min, max, count, sum)

ORC структура строки



ORC сжатие

* Пишется блоками (256к)
* Проверка размера (жать/не жать)
* Поддержка ZLIB, SNAPPY



ORC

индексы

* File level (общая статистика по всем полям в файле)
* Stripe level (статистика по каждому полю страйпа)
* Row level (статистика по каждому полю по 10K строк в страйпе)

ORC

индексы

* orc.create.index=true
* orc.row.index.stride=10000
* min+max+sum
* orc.bloom.filter.columns="field1,feild2"
* orc.bloom.filter.fpp=0.05

ORC benchmark