

# Napredni formati

# Apache Avro

Format za serijalizaciju slogova

# O Apache Avro projektu

- Razvija pod okriljem *Apache* fondacije za razvoj softvera
- Nastao u sklopu *Apache Hadoop* alata za obradu velikih masiva podataka
- Pruža na korišćenje servise za serijalizaciju i razmenu podataka
  - Korišćenjem servisa za serijalizaciju, programi mogu efikasno da vrše (de)serijalizaciju poruka
- Podržan u velikom broju programskih jezika (*Java, Python, C#* itd)
- Ključna prednost Avro projekta je dobra podrška za promenu sheme podataka u toku vremena (tzv. “evolucija sheme”)
  - Lako rukovanje nedostajućim, dodatim ili izmenjenim poljima sheme

# Avro format

- Avro format nalaže da se definicija podataka (shema) i sami podaci skladište zajedno
  - U okviru jedne poruke ili fajla
  - **Time je omogućeno čitanje serijalizovane poruke bez obzira na to da li je shema unapred poznata ili ne**
- Shema se skladišti u JSON formatu
  - Zbog toga je definisanje i tumačenje sheme prilično jednostavno
- Sami podaci se skladište u binarizovanom obliku
  - Što omogućava kompaktno pakovanje podataka i njihovu efikasnu obradu
- Definisanje sheme omogućava validaciju podataka prilikom unosa
- *Python* API - avro biblioteka
  - `python3 -m pip install avro`

# Definisanje Avro sheme

- Avro sheme definišu se uz korišćenje JSON dokumenta
- Neophodno je definisati tip za ono što shema opisuje
  - Tip može biti primitivan ili izvedeni (kompleksni)
- Podržani primitivni tipovi:
  - **null, boolean, int, long, float, double, bytes, string**
- Podržani kompleksni tipovi:
  - **record, array, enum, fixed, map, union**
- Pri korišćenju kompleksnih tipova, moguće je koristiti različite attribute u zavisnosti od odabranog tipa
  - Pogledati [specifikaciju](#) avro formata za konkretne detalje

# Primeri

- Korišćenje različitih tipova
  - `avro_playground.py`
  - Obratiti pažnju na to kako se kroz Python API vrši
    - tumačenje sheme,
    - pisanje u fajl,
    - čitanje iz fajla
  - Obratiti pažnju na to kako su definisane sheme
    - za proste tipove,
    - za kompleksne tipove
  - Pomoću `xxd` pročitati generisane fajlove, protumačiti strukturu
- Rad sa eksterno definisanim schemama
  - `.avsc` ekstenzija
  - `avro_students.py`

# Zadatak

1. Definirati shemu (u `.avsc` fajlu) za skladištenje podataka o polovnim automobilima koji su na prodaju. Atributi:
  - Marka
  - Model
  - Godište
  - Kubikaža
  - Tagovi
  - Cena
2. Kreirati `.avro` datoteku i u nju smestiti 5 slogova
3. Pročitati `.avro` datoteku kreiranu u prethodnom koraku, te odrediti najnižu cenu za automobile mlađe od zadatog godišta

Ostali často korišćeni  
formati

# Protobuf format

- Skraćeno od Protocol Buffers
- Predstavlja proširivi mehanizam za serijalizaciju strukturiranih podataka
- Razvijen u Guglu
- Nezavisan od ciljnog programskog jezika
- Često se koristi u kombinaciji sa gRPC komunikacionim protokolom
- Poruke u protobuf formatu je definišu pomoću *.proto* fajlova, na primer:

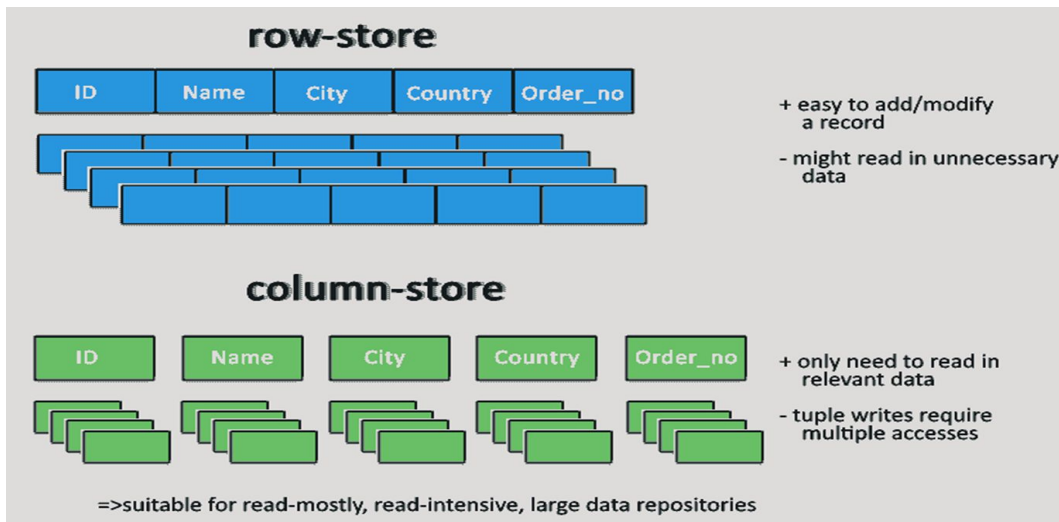
```
message Person {  
    optional string name = 1;  
    optional int32 id = 2;  
    optional string email = 3;  
}
```

# Kolonski orijentisani formati datoteka

- Kod kolonski orijentisanih formata datoteka vrednosti obeležja za isti slog se ne skladište u fizički susednim lokacijama, već se vrednosti jednog obeležja (kolone) za različite slogove skladište zajedno
- Ovakav pristup u skladištenju vrednosti obeležja može imati prednosti u slučaju rada sa velikom količinom podataka (veliki broj slogova - redova) tj. analizom takvih podataka
- Neke od prednosti su
  - Brže upiti kada je potrebno izvršiti operaciju samo nad određenih kolona
  - Efikasnija kompresija podataka u slučaju ponavljajućih vrednosti
- Neki od najpopularnijih kolonski orijentisanih formata su
  - **Parquet**
  - Optimized Row Columnar (**ORC**)

# Orc format

- Orc format je besplatan i open-source kolonski orijentisan format za čuvanje podataka.
- Sličan je ostalim file formatima kao što su RPCFile i Parquet
- Prvi put je objavljen u february 2013. godine
- Kreirala ga je softverska kompanija Hortonworks zajedno sa Facebook-om



# Orc format - prednosti

- Efikasna kompresija - čuva podatke u kolonama i kompresuje što dovodi do manjeg broja čitanja sa diska
- Brzo čitanje - poseduje ugrađene indekse, minimalnu i maksimalnu vrednosti, ali i ostale agregacione funkcije
- Dokazan je u velikim projektima -Facebook koristi Orc format u više od 300 deployment-a

# Orc format - struktura

- Orc fajl sadrži grupe podataka u redovima koji se nazivaju *stripes*
- Sadrži pomoćne podatke u podnožju fajla - *file footer*
- Na kraju fajla se nalazi *postscript* koji sadrži podatke o kompresiji i veličini kompresovanog fajla
- Obično je veličina *strip*-ova je 250 MB
- Detaljnija specifikacija strukture formata može se pronaći sledećem [linku](#)