Big Data

Apache Spark - SparkSQL

Prof. Jean Paul Barddal

Agenda

Agenda

Spark SQL

- Dados estruturados possuem um schema, que são os atributos e os respectivos tipos
- Spark SQL possui uma abstração de dataset que simplifica nosso trabalho com dados estruturados ou semi-estruturados
- Um dataset é similar a uma tabela: linhas e colunas
- Grande parte dos códigos em Spark estão sendo migrados para SparkSQL dada a facilidade de uso
- Usar datasets é interessante e permite que o Spark processe os dados de acordo com comandos SQL-like

Conceitos Importantes

DataFrame

- SparkSQL possui uma abstração de dados chamada DataFrame desde a versão 1.3
- Permite tratar dados estruturados ou semi-estruturados
- Diferentemente de RDDs, Dataframes mantém um schema
- Usa RDDs, então tem características: em memória, resiliente e distribuído

Dataset

- Disponível desde Spark 1.6
- Programação orientada a objetos
- Segurança de tipagem em tempo de compilação (Java)
- Todos os benefícios de usar schemas

DataFrame e Dataset

- Desde o Spark 2.0, DataFrames e Datasets foram unificados
- Internamente, o Spark ainda diferencia o processamento dos dados de acordo com a estruturação:
- Dataframe é uma versão fracamente tipada de um Dataset
- Dataset é fortemente tipado, significando que precisamos codificar uma classe para representar nossos objetos (relevante quando temos um sistema orientado a objetos)

Otimizador

- Spark SQL é rápido
- Para codificar uma solução usando RDDs, nós teríamos mais linhas de código
- Assim como em RDDs, o Catalyst Optimizer conduz otimizações das transformações a serem realizadas

Agenda

Example

- Vamos carregar os dados do StackOverflow
- Vamos conhecer os comandos básicos do SparkSQL

Atividade

- Voltando ao RealEstate.csv, vamos calcular:
- O maior preço; e
- O valor médio do pé quadrado
- Ambos ordenados, de acordo com o valor médio do pé quadrado

Agenda

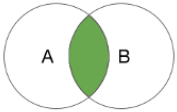
Joins

- Spark SQL suporta os tipos básicos de joins
- Joins são gerenciados pelo Catalyst

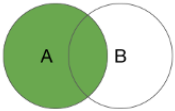
Tipos de Joins

- Nós precisamos definir como um join é feito no parâmetro $\mathrm{joinType}$
- Tipos disponíveis:
- inner
- outer
- left outer
- right outer
- left semi: mesmo que o left outer, mas mantém apenas as colunas da primeira tabela

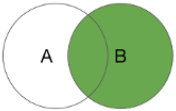
Tipos de Joins



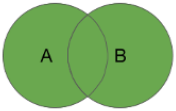
INNER JOIN



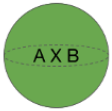
LEFT OUTER JOIN



RIGHT OUTER JOIN



FULL OUTER JOIN



CARTESIAN (CROSS) JOIN

Exemplo

- Vamos realizar um exemplo de join usando os arquivos uk-postcode.csv e uk-makerspaces-identifiable-data.csv
- Importante: o código postal (post code) em cada arquivo possui uma formatação
- No makerspace, temos o código completo (ex.: W1T 3AC)
- No arquivo postcode, temos apenas o prefixo (ex.: W1T)
- Condição do Join: o código do makerspace deve começar com o código no arquivo de postcode

O problema

- W14D T2Y vai casar com W14D e W14!
- Solução:
- Vamos adicionar um espaço em branco para garantir consistência no momento do join
- Assim, W14D T2Y vai casar com W14D mas não com W14