Semana 7

Objetivo: Apresentação da Arquitetura Lambda que vai ser construída e ferramentas de Ingestão dos dados na Nuvem AWS.

Conteúdo:

- Arquitetura Lambda
  - O que é
  - Etapas de uma arquitetura Lambda
- Data Lake
- Apresentação do Caso de Uso a ser implementado
- Ferramentas de Ingestão
  - Talend

Desafio: Realizar a leitura do material completo e executar os 3 exercícios.

Arquitetura Lambda

Primeiramente, arquitetura Lambda não tem nada a ver com o serviço da AWS. Arquitetura Lambda é um modelo de arquitetura BigData proposto por Nathan Marz [5]. Este modelo independe de soluções tecnológicas específicas para a ingestão, armazenamento e processamento dos dados, ou seja, é um modelo teórico. Marz reforça que não há uma única ferramenta que provem uma solução completa de BigData, é necessário utilizar uma variedade de ferramentas e técnicas para construir um sistema completo de Big Data. Por isso tem-se que reforçar a importância do desenho de arquitetura de uma solução.

Para que os dados sejam processados e entregues atingindo uma expectativa de tempo dos stakeholders, a arquitetura Lambda é dividida em três camadas: **batch layer**, **speed layer** e **serving layer** de acordo com a figura abaixo:

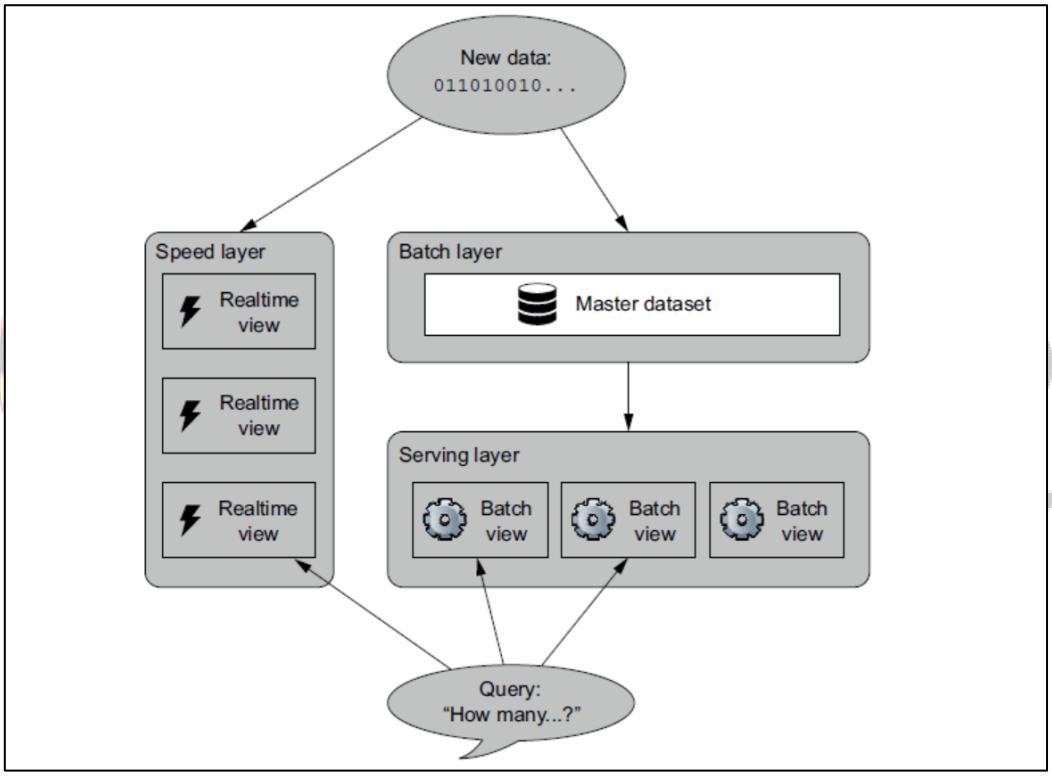


Figura 1 - Arquitetura Lambda proposta por Marz

O dado a ser processado é direcionado para duas camadas (batch e speed). Dentro da **batch layer** esse dado é armazenado de maneira atômica, ou seja, nada é atualizado ou sobrescrito. Caso exista a necessidade de uma mudança, uma nova versão do dado já alterada é armazenada e a anterior não é removida e continua sem mudanças. Isso permite que o dado em seu formato original sempre esteja disponível. Esses dados que estão na **batch layer** são então processados para gerar visualizações pré-calculadas com as informações ajustadas e organizadas de acordo com a necessidade de negócio.

Como a quantidade de dados armazenados a cada dia só cresce, e os dados da **serving layer** só são recebidos ao final do processamento da **batch layer**, o resultado é que cada vez o intervalo para a atualização no **serving layer** fica maior.

Visando compensar esse intervalo a **speed layer** foi criada. Essa camada, que recebe a mesma estrutura atômica de dados, irá processá-los em tempo real e disponibilizá-los para que os sistemas finais disponham dessas informações enquanto esperam pela **batch layer**.

A execução na **speed layer** é bem mais complexa uma vez que os dados precisam ser atualizados e agrupados de acordo com a necessidade do negócio, pois se fossem mantidos em sua forma original inviabilizaria o processamento. Porém essa camada somente precisa se preocupar com os dados que ainda *não foram entregues* pela **batch layer**, o que reduz imensamente a quantidade de dados a ser processada. Assim que o processamento da **batch layer** termina e uma nova versão é disponibilizada, esses dados na **speed layer** podem ser descartados.

Utilizando as três camadas dessa arquitetura é possível processar uma quantidade imensa de dados e mantê-los em sua estrutura original na **batch layer**, disponibilizar esses dados em visualizações pré-computadas (**serving layer**), compensar os intervalos da camada batch e continuar entregando as informações em tempo real (**speed layer**).

Como a arquitetura propõe que os dados sejam armazenados de maneira atômica, o sistema fica protegido até mesmo de erro humano. Também é possível garantir uma visão consistente sem a necessidade de esperar até o final do processamento batch, se beneficiando da **speed layer** e ainda tornando tudo isso transparente aos sistemas finais por meio da **serving layer**.

Mais informações sobre Arquitetura Lambda em: http://lambda-architecture.net/

Data Lake

Um Data Lake é um local central para armazenar todos os seus dados, independentemente de sua origem ou formato[1]. Data Lakes são alimentados com informações em sua forma nativa com pouco ou nenhum processamento realizado para adaptar a estrutura a um esquema corporativo. A estrutura dos dados coletados, portanto, **não é conhecida quando é inserida** no Data Lake, mas é encontrada somente por meio da descoberta, quando lida. Por isso uma grande vantagem de um Data Lake é a **flexibilidade**.

Atributo	Data Lake
Schema	Schema-on-read (Descobre-se a estrutura dos dados em tempo de leitura)
Escala	Escala para grandes volumes a um custo baixo
Métodos de Acesso	Acessado através de sistemas como SQL, programas criados por desenvolvedores e outros métodos
Workload	Suporta processamento batch, além de um recurso aprimorado sobre EDWs para suportar consultas interativas de usuários
Dado	Bruto (Raw), Confiável (Trusted), Refinado(Refined)
Complexidade	Processamento Complexos
Custo/Eficiência	Uso eficiente das capacidades de armazenamento e processamento a um custo muito baixo
Benefícios	• Transforma a economia financeira do armazenamento de grandes quantidades de dados • Suporta HiveQL, Spark e entre outros frameworks de programação de alto nível • Escala para executar em dezenas de milhares de servidores • Permite o uso de qualquer ferramenta • Permite que a análise comece assim que os dados chegam • Permite o uso de conteúdo estruturado e não estruturado em um único armazenamento • Suporta modelagem ágil, permitindo que os usuários alterem modelos, aplicativos e consultas (queries)

Atributos Chaves de um Data Lake

Um repositório para ser considerado um Data Lake deve ter pelo menos as sete (7) características abaixo:

- Deve ser um único repositório compartilhado de dados da organização.
- Aceita todos os tipos de dados estruturados, semi-estruturados e não-estruturados.
- Baixo custo de armazenamento
- Incluir capacidades de orquestração e agendamento de tarefas(jobs)
- Conter um conjunto de aplicativos ou de workflows para consumir, processar ou agir de acordo com os dados
- Suporta regras de segurança e proteção de dados.

- Desacopla o armazenamento do processamento (permitindo alta performance e alta escala).

Funções básicas de um Data Lake

Um Data Lake possui 4 funções básicas: ingestão, armazenamento, processamento e consumo (ingestion, storage/retention, processing, and access). A figura abaixo sintetiza essas funções.

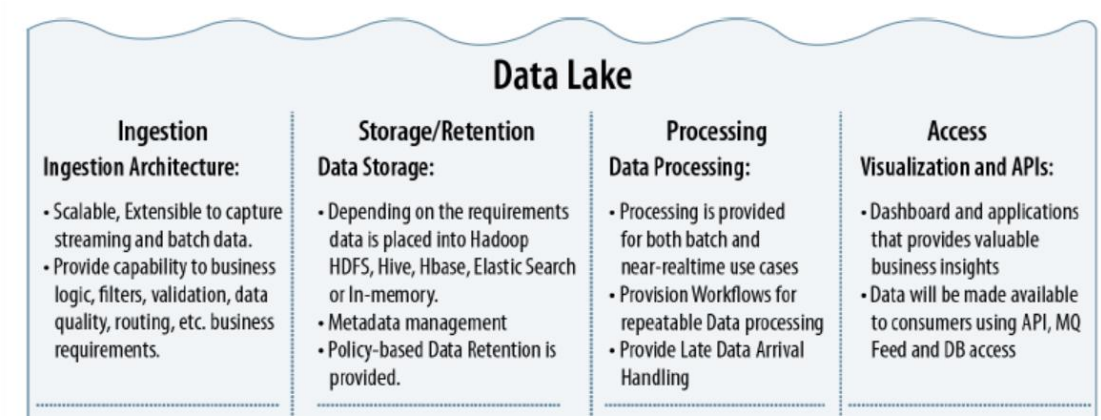


Figura 3 - Funções básicas de um Data Lake

Ingestão

Camada para capturar por streaming e batch os dados das diferentes origens. Importante dessa camada ser gerenciada, pois assim dá controle sobre como o dado foi ingerido, de onde o dado veio, quando o dado chegou e onde o dado está armazenado no Data Lake.

Uma importante parte de uma arquitetura de Data Lake é primeiro colocar o dado em uma área de transição ou stage (transitional area) antes de movê-lo para o repositório de dados brutos (raw).

Processos de governança podem incidir na fase de ingestão como: criptografia, procedência ou linhagem, capturar metadados e limpar os dados.

Armazenamento

Por definição um Data Lake prove melhor eficiência de custos para o armazenamento que um EDW. Como Data Lake utiliza-se de técnicas como schema on-read, cada parte dos dados é armazenada com otimização, não existem linhas ou colunas nulas. Além de utilizar hardware mais barato para tal.

Processamento

Processamento é a fase no qual os dados podem ser transformados em um formato padronizado por usuários de negócios ou cientistas de dados. Essa fase é necessária pois na ingestão não há processos para este fim.

Um dos maiores benefícios dessa metodologia é que diferentes usuários de negócios podem executar diferentes padronizações e transformações, dependendo de suas necessidades. Muito diferente de um EDW.

Com as ferramentas certas, você pode processar dados para casos de uso em batch e em near real-time. O processamento em batch é para cargas de trabalho ETL tradicionais ou grandes

blocos de dados. Streaming é para cenários em que relatórios precisam ser entregue em real-time ou near real-time e não podem esperar por uma atualização diária.

Consumo

Existem várias formas de acessar os dados: queries, extrações baseadas em ferramentas ou extrações que precisam acontecer por meio de uma API. A visualização é uma parte importante desta etapa, onde os dados são transformados em gráficos e gráficos para facilitar a compreensão do dado pelo usuário.

Para garantir o bom funcionamento destas etapas básicas, há uma importante função ainda não listada, o Gerenciamento e Monitoramento.

Gerenciamento e Monitoramento

A governança de dados está se tornando uma parte cada vez mais importante de um Data Lake corporativo. Soluções que apresentem a linhagem do dado, gestão e captura dos metadados, glossário de negócios, entre outras necessidades importantes na organização do Data Lake.

Essas soluções são mais completas em ferramentas de Catálogo de Dados e utilizam diferentes abordagens. Soluções abordam a questão de diferentes ângulos. Um método top-down usa as práticas recomendadas das experiências de EDW das organizações e tenta impor governança e gerenciamento a partir do momento em que os dados são ingeridos no lago de dados. Outras soluções adotam uma abordagem bottom-up que permite aos usuários explorar, descobrir e analisar os dados de maneira muito mais fluida e flexível. E algumas soluções combinam as duas abordagens.

Metadados são extraordinariamente importantes para gerenciar seu Data Lake. Existem três tipos distintos, mas igualmente importantes, de metadados para coletar: dados técnicos, operacionais e de negócios.

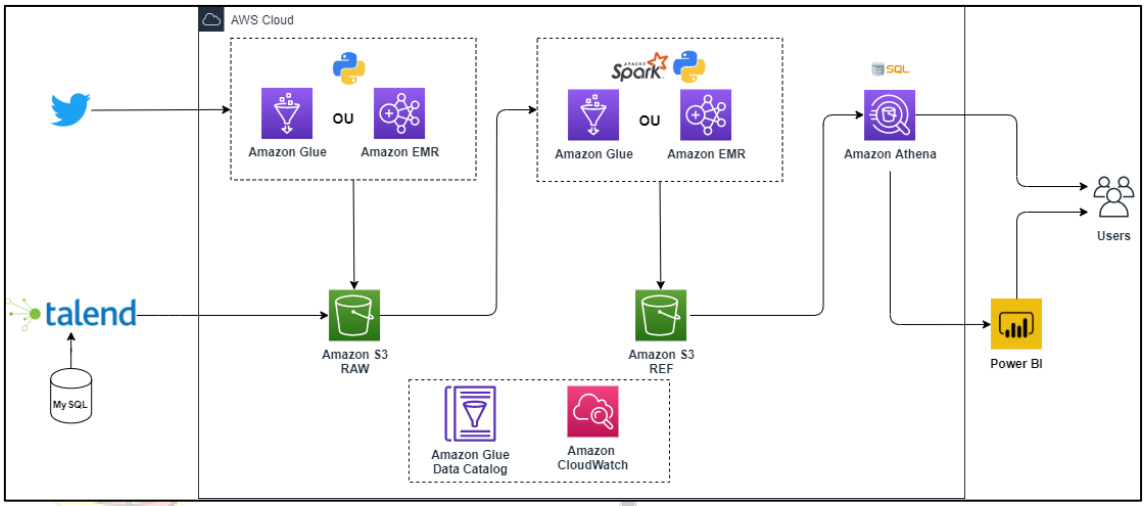
Tipo do Metadado	Descrição	Exemplo
Técnico	Captura a forma e a estrutura de cada conjunto de dados	Tipo de dados (texto, JSON, Avro), estrutura dos dados (os campos e seus tipos)
Operacional	Captura a linhagem (lineage), qualidade, perfil e governança do dado	Locais Localização de origem e destino dos dados, tamanho, número de registros e a linhagem
Negócio	Captura o que isso significa para o usuário	Nomes comerciais (business), descrições, tags, qualidade e regras de mascaramento

Todos esses tipos de metadados devem ser criados e ativamente curados (curated) - caso contrário, o Data Lake é simplesmente uma oportunidade desperdiçada. Esses recursos protegem os dados e reduzem os riscos, pois o data manager sempre saberá de onde os dados vieram, onde estão e como estão sendo usados.

Caso de Uso a ser implementado

Durante o programa aprendemos diversos conceitos e linguagens. Chegou a hora de usá-los para resolver um problema da empresa XPTO. A empresa XPTO é uma organização sem fins lucrativos que tem o objetivo de saber como está o humor dos brasileiros. Para isso ela deseja usar dados do Twitter. Atualmente essa empresa possui uma base de dados com uma massa de dados de 2018 que ela gostaria de analisar em conjunto com dados atuais sobre a popularidade do presidente da República Federativa do Brasil.

Com base neste problema, a XPTO elaborou um desenho de arquitetura de uma solução para resolver o problema deles utilizando a nuvem da AWS. O desenho proposto está abaixo:



A implementação da solução está dividida da seguinte forma:

Atenção: Para os itens 2 e 3, tem-se duas opções de ferramentas. Você deve construir a mesma lógica nas duas ferramentas a efeito de comparação. Depois você escolherá qual das duas ferramentas estará presente na sua solução e o motivo de sua escolha.

1) Ingestão batch: a ingestão do arquivo CSV em Bucket Amazon S3 RAW Zone. Nesta arquitetura experimentaremos a ferramenta Talend. Nesse caso utilizaremos o Talend como parte do processo de ingestão via batch para geração de arquivo (CSV) com base em dados existentes em Banco de Dados MySQL.
2) Ingestão streaming: será através da captura dos tweets em tempo real via AWS Glue e Amazon EMR com Python usando a lib tweepy sendo persistidos em Amazon S3 RAW Zone no mesmo formato original (JSON) e agrupando os tweets em no máximo 100 tweets em cada arquivo JSON.
3) Processamento: ocorrerá através do Apache Spark unificando os processos streaming e batch para gerar uma visão única persistida em Amazon S3 REF Zonre a partir de uma EMR e do AWS Glue. Todos os dados serão persistidos no formato (PARQUET) particionados por data de criação do tweet (dt=<ano mês dia> exemplo: dt=20180331). Esse processamento ocorrerá em 2 momentos. O primeiro chamado de Carga de dados Histórica via Apache Spark Batch carregará os dados já existentes na RAW Zone e o segundo momento chamado de Carga de Dados incremental via Apache Spark Streaming carregará os dados novos que forem sendo gravados na RAW Zone a cada 15 minutos.

Obs.: Deixar o processo 2 (ingestão streaming) executando apenas depois do processo 3 (Processamento) incremental estiver ativo.

Nota: O Arquivo parquet gerado na REF Zone deve conter as seguintes colunas:
ID do tweet, Tweet (mensagem), Data de Criação (Com Ano, mês, dia, hora, minuto e segundo) do Tweet, Origem do Tweet (source), Número de Retweets , Sentimento e Símbolo. Pode ser que alguns campos não existam no processo histórico ou no incremental, nesses casos deixar os campos sem dados nulos.

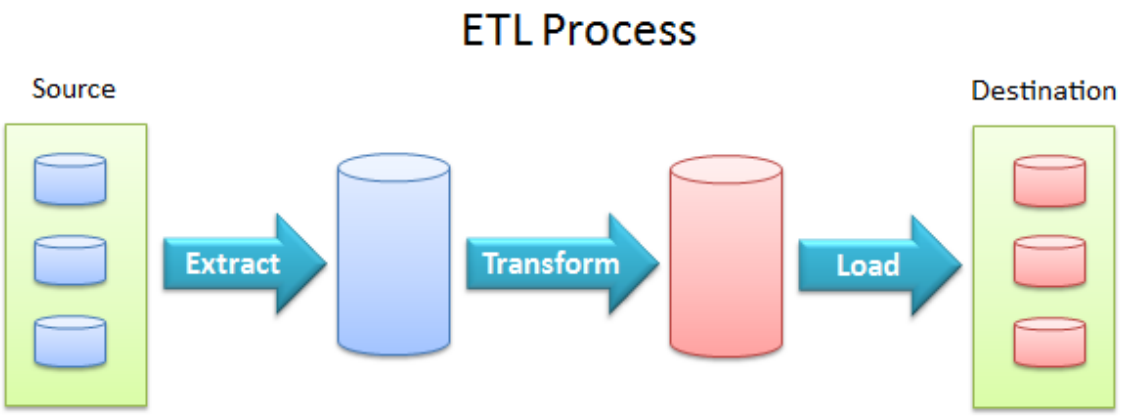
4) Armazenamento: todo o armazenamento das informações geradas em cada etapa do processo ocorrerá em Amazon S3.
5) Consumo: o consumo das informações pelos analistas e demais usuários da XPTO ocorrerá por meio de dashboards implementados na ferramenta PowerBI em conjunto com Amazon Athena e por meio de consultas SQL diretamente no Amazon Athena.

Talend

O **Talend Data Integration** é uma solução aberta e escalável de integração de dados e qualidade de dados para integrar, limpar e perfilar todos e qualquer tipo de dados.

Possui **mais de 900 componentes pré-construídos**. Trabalha com armazenamento em nuvem , integração de dados , gerenciamento de dados , gerenciamento de dados mestre , qualidade de dados, preparação de dados e software e serviços de integração empresariais. Um dos pontos de destaque do Talend é fazer ETL ou ELT.

Extração, Transformação, Carregamento, do inglês Extract Transform Load (ETL) é o processamento de blocos de dados em etapas de Extração, Transformação e Carga. ETLs são comumente utilizados para construir um data warehouse (DW) e Business Intelligence (BI) mas podem ser utilizados para outras finalidades. De acordo com a SAS, nesse processo os dados são retirados (extraídos) de um sistema-fonte, convertidos (transformados) em um formato que possa ser analisado, e armazenados (carregados) em um armazém ou outro sistema. Vale ressaltar que algumas ferramentas não fazem o ETL tradicional e sim uma variação, o ELT.



Na prática, se você precisa importar um arquivo csv (arquivo texto separado por vírgulas) em um banco de dados, convertendo os valores das colunas para tipos de dados aceitos em um banco de dados, você provavelmente usará alguma ferramenta de ETL como o Oracle Data Integrator ou o Pentaho ou Talend.

Para resolver o exemplo acima, os passos de ETL são: um componente extrai os dados de um arquivo texto, componentes de derivação e conversão de colunas transformam os dados e um outro componente carrega os dados no banco de dados[3].

Perfeito! Seus dados são carregados em um banco de dados usando uma tecnologia simples e confiável e todos ficam felizes. Então por que precisamos de uma nova sigla misteriosa para supostamente melhorar esta técnica?

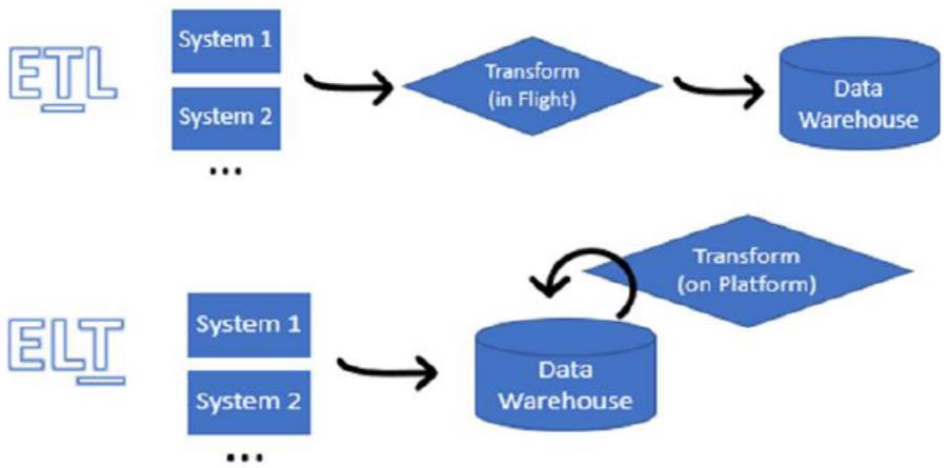
Acredito que um ETL simples como o exemplo mostrado acima, não seria um grande problema em termos de implementação, manutenção e complexidade. Porém, suponha que você precisa fazer isso com milhares de arquivos e cada arquivo tem muitos GB de dados criados diariamente e estes dados devem ser disponibilizados para um grande número de consumidores de dados, Data Warehouses e/ou outras aplicações incluindo aplicações de terceiros? O seu ETL pode se tornar algo bastante complexo, enfrentando problemas de desempenho e de disponibilidade de

dados. Além disso seus usuários só poderão acessar os dados extraídos depois que todo o processo terminou, o que pode levar um tempo considerável dependendo das tecnologias usadas e do contexto da sua organização[3]

Extrair / carregar / transformar (ELT) da mesma forma extrai dados de uma ou várias origens, mas os carrega para o destino sem nenhuma outra formatação. A transformação de dados, em um processo ELT, ocorre no banco de dados de destino. O ELT necessita de menos fontes remotas, exigindo apenas dados brutos e despreparados.

O ELT existe há algum tempo, mas ganhou um interesse maior com as novas ferramentas para movimentação de dados como o Apache Hadoop. Uma grande tarefa como transformar petabytes de dados brutos foi dividida em pequenos trabalhos, processada remotamente e retornada para carregamento no banco de dados[4].

Cada método tem suas vantagens. Ao planejar a arquitetura de dados, os tomadores de decisão de TI devem considerar os recursos internos e o crescente impacto das tecnologias em nuvem ao escolher ETL ou ELT.



Produtos Free Talend:

Open Studio for Data Integration: Inicie seus projetos de ETL e integre os dados

Open Studio for Big Data: Simplifique o ETL para conjuntos de dados grandes e diversos

Data Preparation – Free Desktop: Permitir que os usuários descubram, combinem e limpem os dados

Open Studio for ESB: Acelerar a orquestração de aplicativos e APIs

Open Studio for Data Quality: Avalie a precisão e a integridade dos dados

Stitch Data Loader: Carregue facilmente dados de dezenas de fontes na nuvem em Datawarehouses e Data Lakes em minutos.

Em nosso projeto vamos usar o produto OpenStudio for BigData. Para usarmos esse produto vamos esclarecer alguns conceitos antes:

Repositório: Local de armazenamento que o Talend Studio usa para coletar dados relacionados a todos os itens técnicos usados para descrever modelos de negócios ou para projetar os Jobs.

Projeto: São coleções estruturadas de itens técnicos e seus metadados associados. Todos os trabalhos e modelos de negócios utilizados são organizados em Projetos.

Workspace: É o diretório onde se armazena todas as pastas do seu projeto. É necessário ter uma workspace por conexão do repositório. O Talend Studio permite a conexão a diferentes workspaces ou pode-se usar a workspace padrão.

Job: É o design gráfico, de um ou mais componentes conectados, que permitem executar processos de fluxo de dados. Ele traduz as necessidades de negócios em códigos, rotinas e programas. Os Jobs abordam todas as diferentes origens e destinos necessárias para os processos de integração de dados e todos os outros processos relacionados.

Componente: É um conector pré-configurado usado para executar uma operação de integração de dados específica, independentemente das origens de dados integradas: bancos de dados, aplicativos, arquivos simples, serviços Web, etc. Os componentes são agrupados em categorias de acordo com o uso e exibidos na Paleta de Integração do Talend Studio.

Item: É a unidade técnica fundamental em um projeto. Os itens são agrupados, de acordo com seus tipos como: Job Design, Business model, Context, Code, Metadata, etc. Um item pode incluir outros itens. Por exemplo, os modelos de negócios e os Trabalhos que você projeta são itens, metadados e rotinas que você usa dentro de seus Trabalhos que por sua vez também são itens.

Exercícios Parte 2

1) Executar o Lab: “Instalando e configurando Talend”
2) **Desafio XPTO:** Criar Estruturas de armazenamento S3. Bucket para armazenar dados brutos oriundos da ingestão de dados, tanto batch quanto streaming (RAW Zone) e Bucket para armazenados dados refinados oriundos do processamento (REF Zone)
3) **Desafio XPTO:** Fazer a geração de um arquivo CSV com os dados de 2018 que estão num banco de dados MySQL para o S3 na camada Bruta (RAW Zone) com Talend formatando a as colunas conforme o tipo de dado

Dica: ao utilizar o componente tS3Connection instale os externals jar que o Talend Studio indicar

Dica 2: manual do Talend sobre AWS
https://help.talend.com/reader/3iNwsJWYog7gRV7uP2VB8A/rKrUdHY5Amk__nvoCzfuRQ

Referências

[2] https://www.tutorialspoint.com/sqoop/index.htm

[3]https://www.linkedin.com/pulse/etl-vs-elt-quando-duas-letras-fazem-muita-diferen%C3%A7a-rodrigo-r-g-/

[4] https://www.talend.com/resources/elt-vs-etl/

[5] https://pt.slideshare.net/Celio12/nosql-base-vs-acid-e-teorema-cap