Data Pipeline com Google Cloud Platform

Stack

Quem é o Gabriel?

- Engenheiro de dados na Avenue Code.
- Carioca, torcedor do Flamengo.
- Formado em engenharia eletrônica no Cefet/RJ.
- Certificações:
  - GCP Professional Data Engineer e Cloud Architect.
  - AWS Solutions Architect Associate e Cloud Practitioner.

Stack

Qual o objetivo deste curso?

Extração	Carregamento	Tratamento e Disponibilização
Extração dos dados a partir da origem com uma API desenvolvida em FastAPI.	Carregamento dos dados extraídos nos recursos de storage da GCP seguindo boas práticas.	Tratamento de dados através de Spark, via criação de cluster Dataproc.

Stack

Como o curso está organizado?

Aula 1

- Conceitos de Cloud.
- Por que Google Cloud Platform (GCP)?
- Apresentação da arquitetura do curso.
- Introdução a Infra as Code e Terraform.
- Introdução a CI/CD e Github Actions.
- Apresentação da fonte de dados.

Stack

Aula 2

- Conceitos de API e FastAPI.
- Conceitos de Data Lake.
- Conceitos de Docker.
- Conceitos de Cloud Run.
- Desenvolvimento da API.
- Testes da API com Postman.
- Deploy da API no Cloud Run.

Stack

Aula 3

- Conceitos de Spark.
- Conceitos de Dataproc.
- Criação de cluster de teste para etapa de tratamento.
- Desenvolvimento das etapas de tratamento.

Stack

Aula 4

- Conceitos de Airflow.
- Conceitos de BigQuery.
- Subida do Airflow com Docker.
- Testar o Airflow com uma DAG simples.
- Desenvolvimento da DAG do pipeline.
- Execução da DAG.
- Desenvolvimento do dashboard com Looker Studio.

Stack

Para o curso, instalar:

- Anaconda.
- Docker.
- Google Cloud SDK.
- Postman.
- Editor de sua preferência (VS Code, Pycharm etc).

Aula 1

Stack

O que é a Google Cloud Platform?

A GCP é uma suite de serviços de computação que roda na mesma infraestrutura que o Google utiliza para os seus principais serviços, como Gmail, Search Engine e YouTube.

Por que GCP?

- O Google tem tudo a ver com dados e eles são extremamente bons em gerenciar e dimensionar big data, fornecendo soluções flexíveis.
- Free tier realmente Free.
- Atualmente a GCP está com a estratégia de ser mais uma cloud ao invés de a cloud que uma empresa usa.
- É minha cloud favorita 😜

Stack

Infraestrutura física

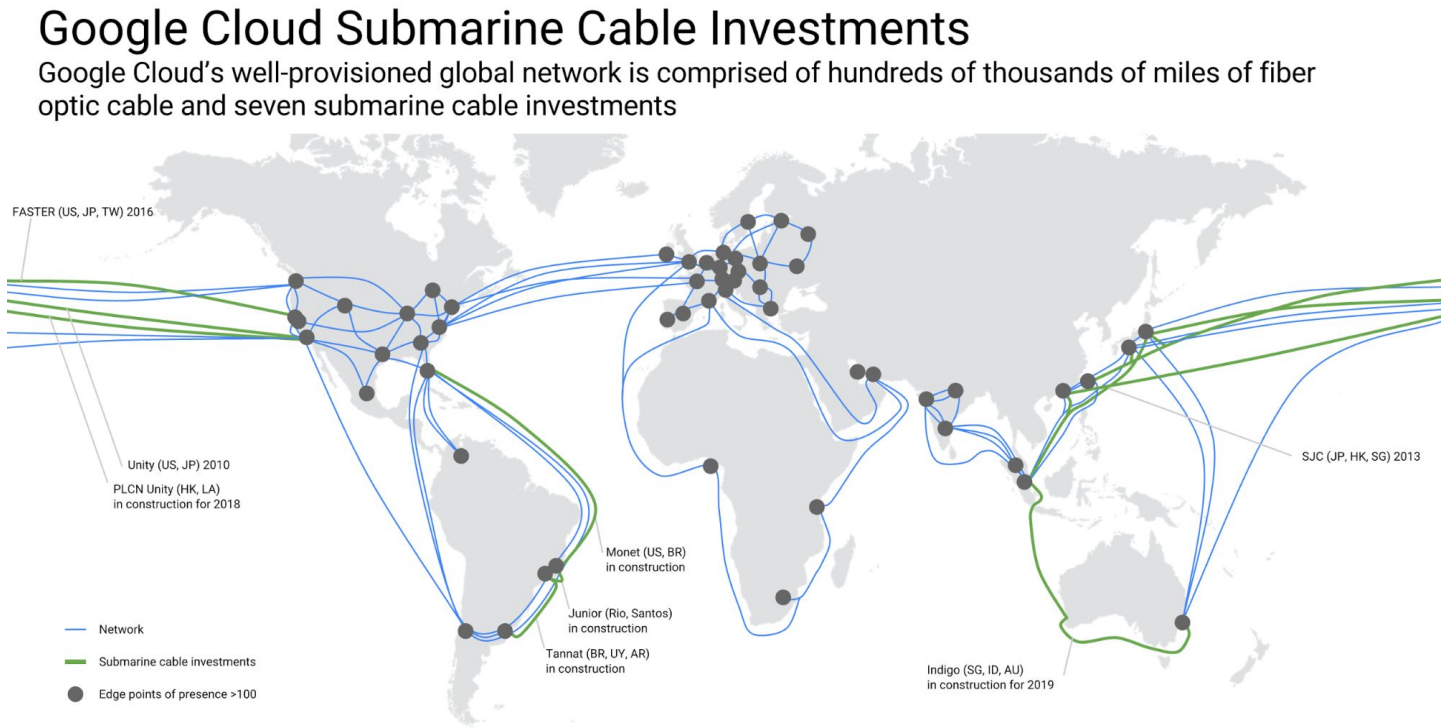
- vCPU.
- Servidor físico.
- Rack.
- Data center.
- Zona.
- Região.
- Multi Região.
- Rede global do Google.
- Pontos de presença (PoP).
- Sistema Global.

Regiões da GCP

Stack

Fonte da imagem: GCP

Rede Global

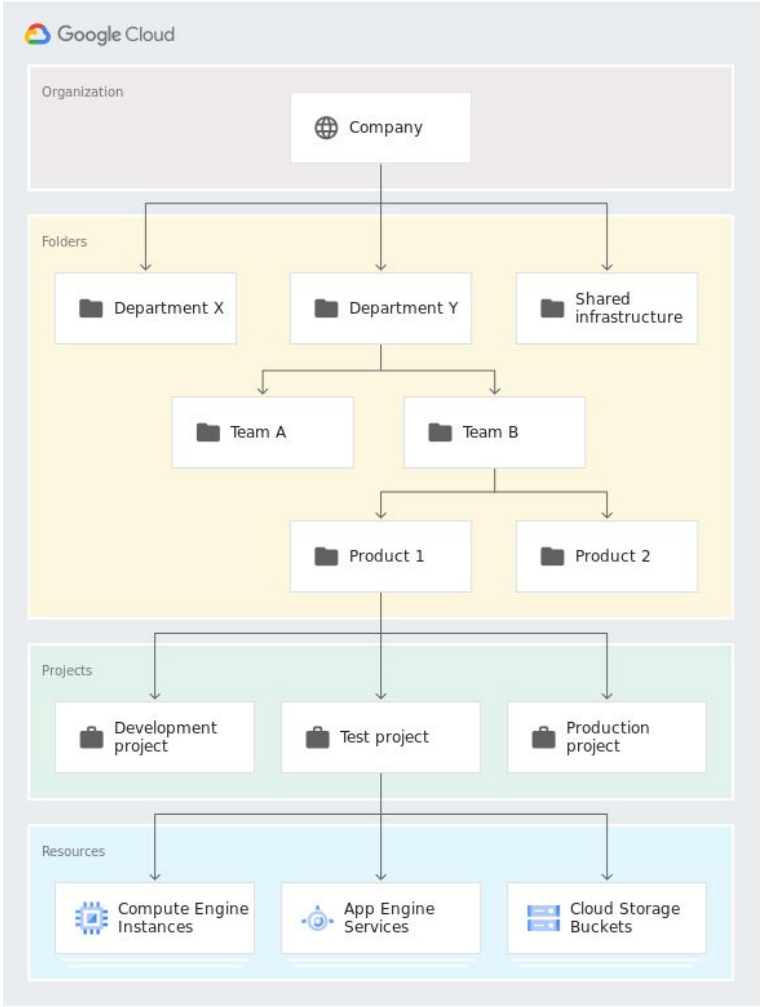


Stack

Fonte da imagem: GCP

Organização na GCP

- Os recursos pertencem a projetos.
- Projetos são semelhantes a contas na AWS.
- Recursos podem ser compartilhados entre projetos.
- Projetos podem ser agrupados e controlados por hierarquia.



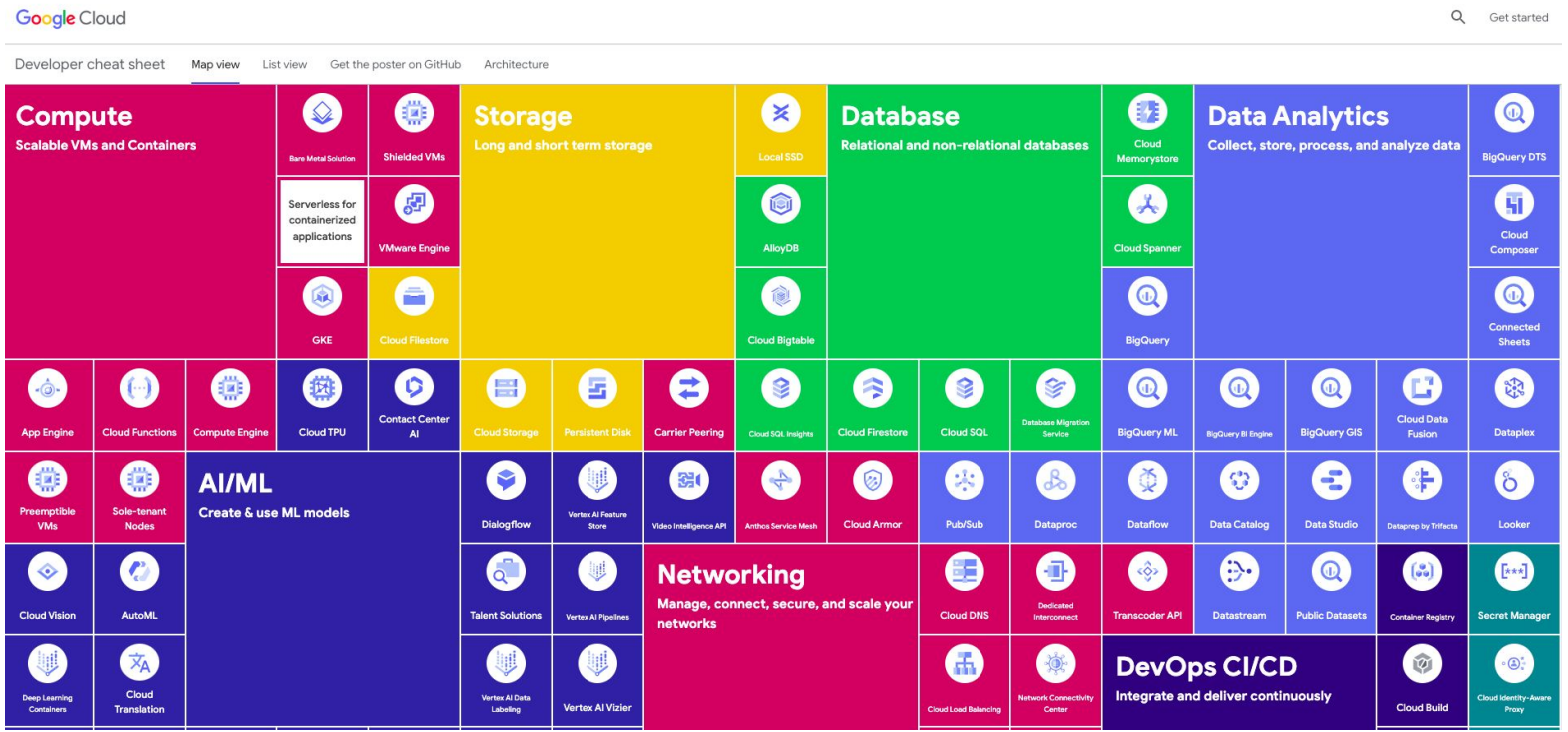
Stack

Como funciona o *Pricing* na GCP?

- Como funciona o pagamento por um serviço?
  - Provisionado: "Esteja pronto para lidar com X de carga". Recursos estão lá, usando ou não.
  - Por uso: "Me cobre apenas o que eu usei".
- Tráfego de rede:
  - De graça na entrada (***ingress***).
  - Cobrado na saída (***egress***), por GBs usado.
  - Egress para outros serviços da GCP frequentemente é de graça, a depender do serviço e da localização.

Stack

The Google Cloud Developer's Cheat Sheet



Stack

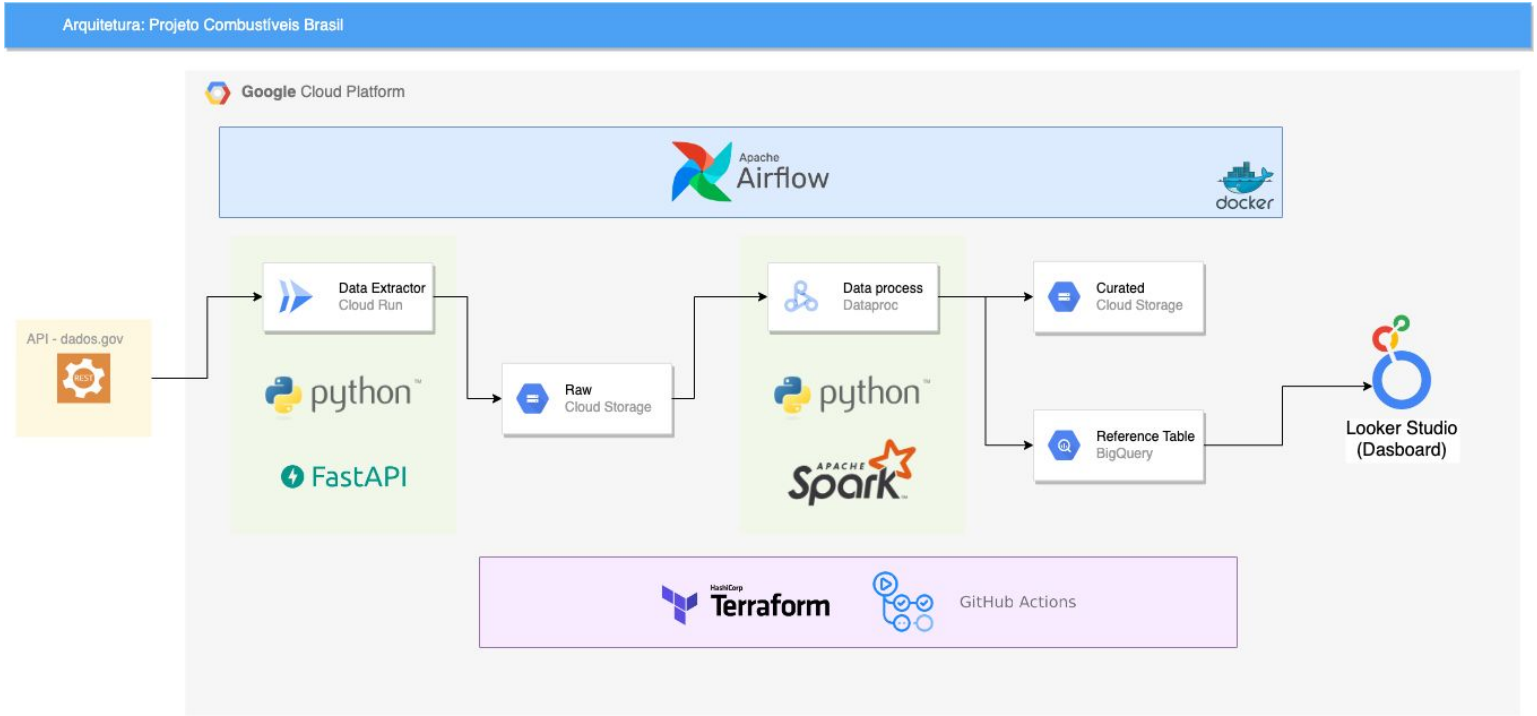
Para outros formatos de cheat sheet: link

Serverless vs Totalmente gerenciado

- **Serverless**: significa "pague conforme usa". Sem tráfego, paga nada, muito tráfego, o escalonamento é automático e paga de acordo com o tráfego.

- **Totalmente gerenciado**: sempre existe um número mínimo de VMs/nós ativos e um custo atrelado a eles, com tráfego ou não. No entanto, não é preciso se preocupar: patches, atualizações, rede, backups, HA, redundância(...) são gerenciados.

O que teremos construído ao final do curso?



Stack

Criação de conta e Free Tier

- Crédito de $300 UDS ou 90 dias.
- Free trial de fato free!
- Para tornar a conta billable, é preciso habilitar manualmente.
  - Ótimo para aprender!
- Ainda precisa de cartão de crédito mas não será efetuada cobrança no Free Tier.
- Contas de email corporativas não são elegíveis para Free Tier.
- Dicas para a conta:
  - Não repetir senha, se possível usar um gerenciador de senhas.
  - Habilitar autenticação multifator (MFA) na conta.
  - Criar alarmes de faturamento.

Stack

Fonte: GCP

Limitações do Free Tier

- No máximo 8 vCPUs simultâneas.
- Não é permitido usar GPUs e TPUs.
- Não é permitido solicitações para aumento de Quotas.
- Não é permitido mineração de cripto moedas.
- Não possui SLAs.
- Não é permitido usar licenças premium de OS (Exemplo: Windows).

Stack

Princípio do privilégio mínimo

- O princípio do privilégio mínimo afirma que um recurso deve ter acesso apenas ao(s) recurso(s) exato(s) de que precisa para funcionar. Por exemplo, se um serviço estiver executando um backup automatizado de banco de dados, o serviço deve ser restrito a permissões somente leitura exatamente no banco de dados em questão.

Stack

Fonte: GCP

Infraestrutura como código

> *"Podemos definir IaC como um método que usa arquivos de definição (em geral arquivos de configuração padronizados, com o json e yaml) para gerenciar e provisionar infraestruturas de TI. Com isso, podemos entender essa prática como uma abordagem de engenharia de software aplicada para operações."*

- Velocidade, simplicidade e replicabilidade.
- Consistência na configuração, menor chance de erro humano.
- Minimização de riscos, controle de versão.
- Aumento na eficiência no desenvolvimento de software.
- Economia financeira, fácil de desativar recursos.
- Fácil de seguir e monitorar políticas e boas práticas.

Stack

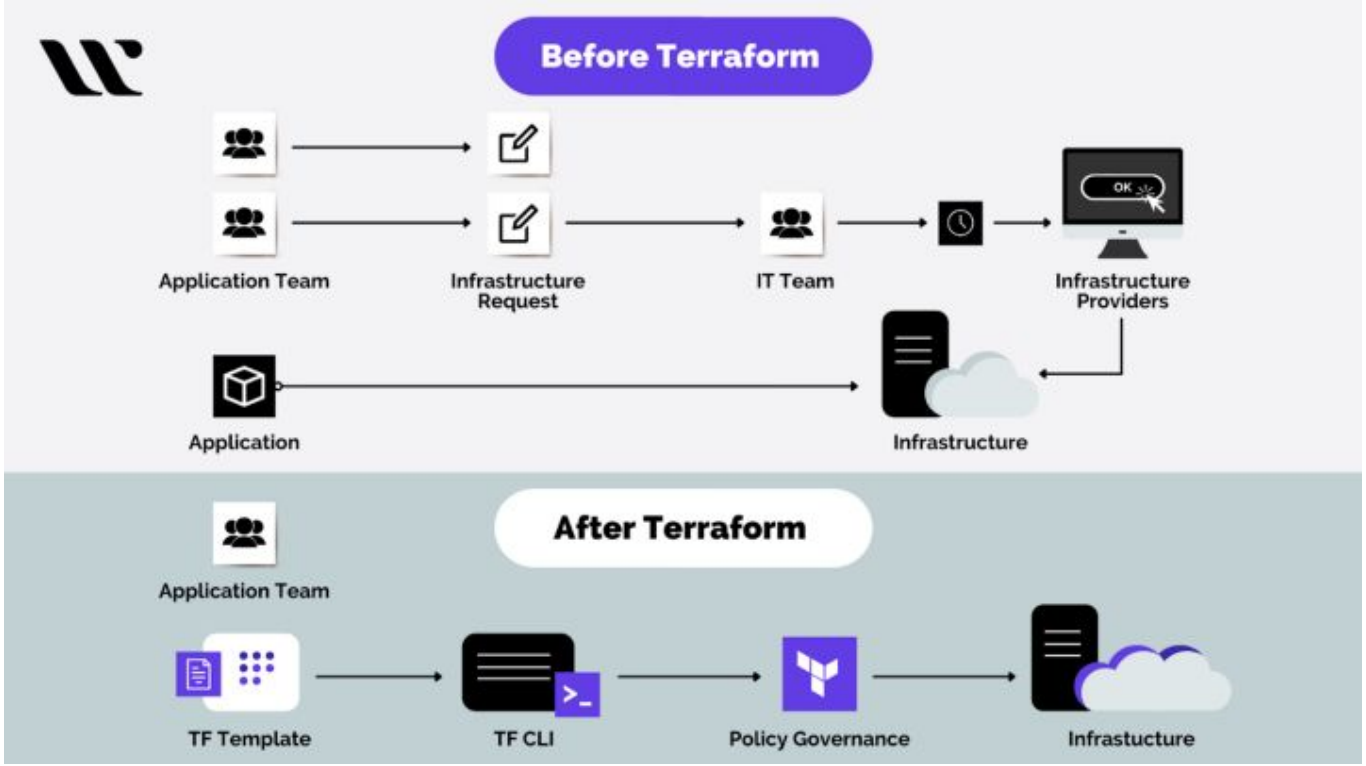
Terraform 101

> *“Terraform é uma ferramenta de infraestrutura como código (IaC) que permite criar e alterar versões de infraestrutura com segurança e eficiência. Isso inclui componentes de baixo nível, como instâncias de computação, armazenamento e rede, bem como componentes de alto nível, como entradas de DNS, recursos de SaaS, et.”*

- Faz orquestração, não apenas gerenciamento de configuração.
- Suporta vários provedores, como AWS, Azure, Oracle, GCP e muitos mais.
- Fornece infraestrutura imutável onde a configuração muda suavemente.
- Usa linguagem de fácil compreensão, HCL (linguagem de configuração HashiCorp).

Stack

Terraform 101



Stack

Terraform - ciclo de vida

- **terraform init**: inicializa o ambiente Terraform (local). Normalmente é executado apenas uma vez por sessão.
- **terraform plan**: compara o estado do Terraform com o estado como está na cloud, constrói e exibe um plano de execução.
- **terraform apply**: executa o plano.
- **terraform destroy**: exclui todos os recursos controlados por este ambiente específico do terraform.

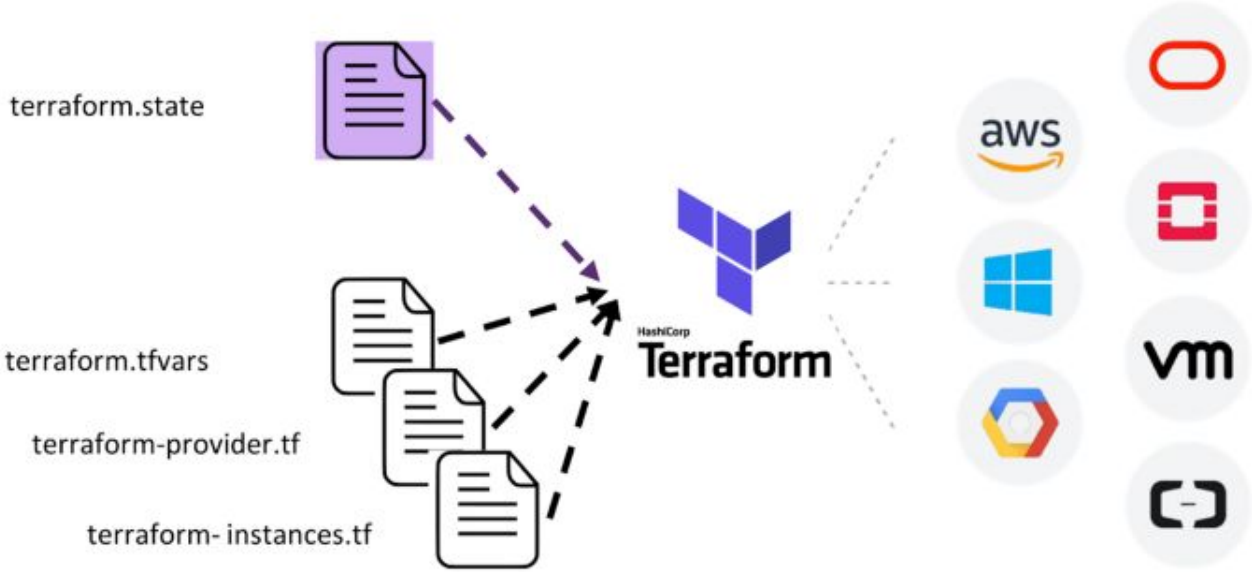
Stack

Terraform - exemplo

```
terraform {
  required_providers {
    google = {
      source = "hashicorp/google"
      version = "3.5.0"
    }
  }
}

provider "google" {
  credentials = file("<NAME>.json")

  project = "<PROJECT_ID>"
  region  = "us-central1"
  zone    = "us-central1-c"
}

resource "google_compute_network" "vpc_network" {
  name = "terraform-network"
}
```

Stack

Terraform - tipos de arquivos



Stack

Fonte da imagem: K21Academy

Conceitos de Controle de Versão

- Em engenharia de software, controle de versão é uma classe de sistemas responsáveis por gerenciar mudanças em programas de computador, documentos, websites ou outras coleções de informações. Controle de versão é um componente essencial do processo de gerenciamento de software.

Stack

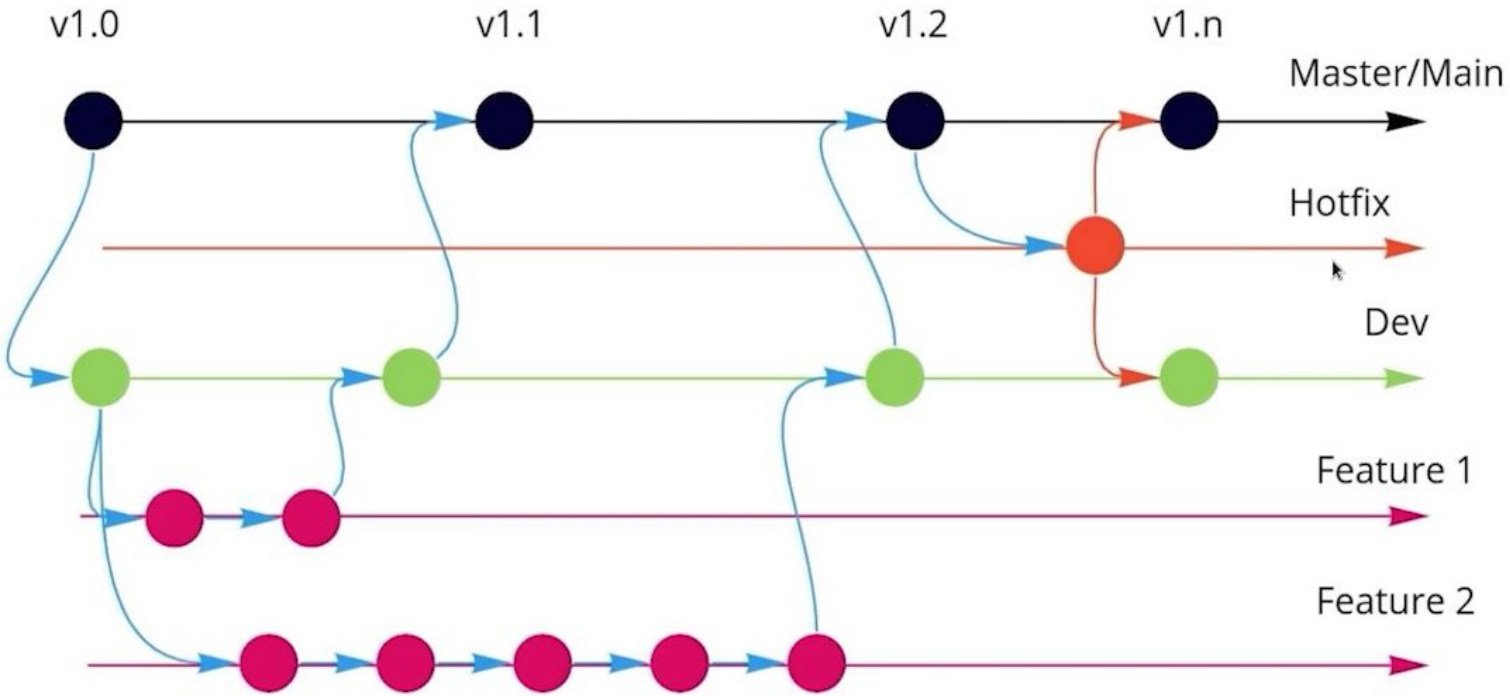
Git

- Git é um sistema de controle de versão distribuído gratuito e de código aberto projetado para lidar com tudo, desde projetos pequenos a muito grandes com velocidade e eficiência.

Stack

Fonte da imagem: K21Academy

Gitflow



Stack

Git: Branches, Commits e Pull Requests

- git init
- git clone
- git add
- git commit
- git diff
- git status
- git tag
- git branch
- git checkout
- git merge
- git push

Stack

Fonte: DZone

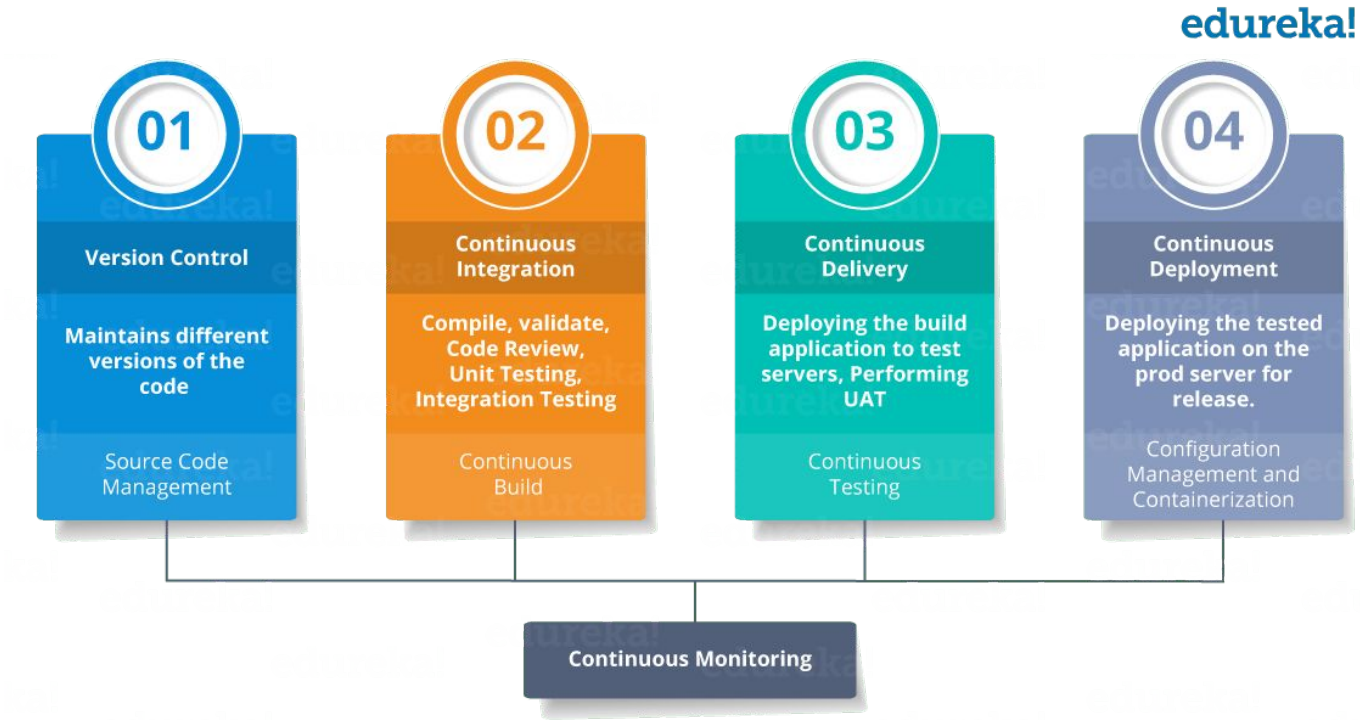
Conceitos de CI/CD

- A integração contínua é uma prática de desenvolvimento de software de DevOps em que os desenvolvedores, com frequência, juntam suas alterações de código em um repositório central. Depois disso, criações e testes são executados.

- Os principais objetivos da integração contínua são encontrar e investigar bugs mais rapidamente, melhorar a qualidade do software e reduzir o tempo que leva para validar e lançar novas atualizações de software.

Stack

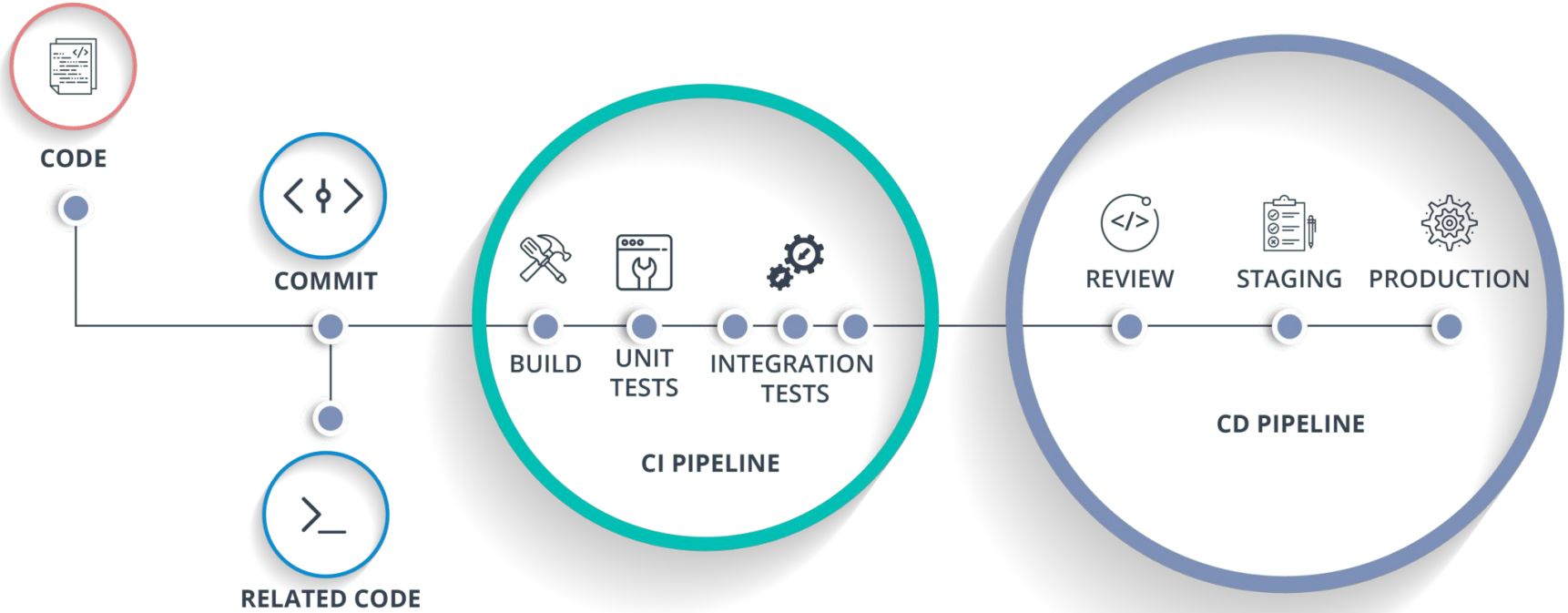
Conceitos de CI/CD



Stack

Fonte da imagem: DZone

Conceitos de CI/CD



Stack

Fonte da imagem: DZone

Ferramentas de CI/CD

- Circle CI
- Github Actions
- Gitlab CI
- Jenkins
- Cloud Build
- AWS CodePipeline
- …

Stack

Fonte da imagem: DZone

Conceitos de Github Actions

- O GitHub Actions permite automatizar, customizar e executar os workflows de desenvolvimento de software diretamente do seu repositório GitHub. Você pode descobrir, criar e compartilhar actions para executar qualquer trabalho que você queira, incluindo CI/CD, e também combinar actions para criar um workflow totalmente customizado

Stack

Conceitos de Github Actions

```yaml
name: learn-github-actions
run-name: ${{ github.actor }} is learning GitHub Actions
on: [push]
jobs:
  check-bats-version:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v3
      - uses: actions/setup-node@v3
        with:
          node-version: '14'
      - run: npm install -g bats
      - run: bats -v
```

Stack

Fonte: Github

Fonte de dados

- Dados da série histórica do preço de combustíveis no brasil.
- Arquivos divididos por semestre desde 2004.
- Formato csv.
- Link: dados.gov.

Stack

Prática:
Criação de conta na GCP e deploy da infra do projeto.

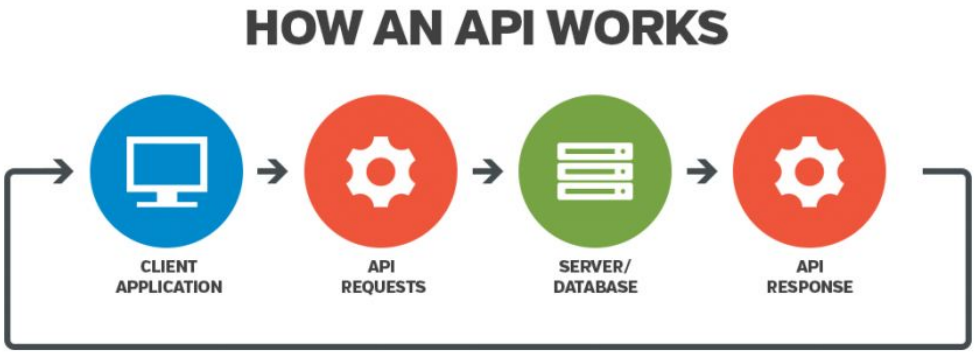
Stack

Aula 2

Stack

Conceitos de API

- A Application Programming Interface (API) é uma interface que permite que duas aplicações interajam entre si. Uma API é uma coleção de funções e procedimentos de software. A API é definida como um código que ajuda dois softwares diferentes a se comunicarem e trocarem dados entre si.

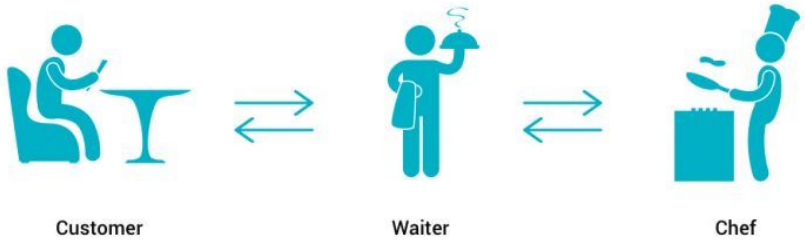


Stack

Fonte da imagem: WellDoneBy

Conceitos de API

- Uma API, assim como um garçom, informa ao sistema o que você deseja e fornece uma resposta para você.



Stack

Fonte da imagem: Mono Solutions

Conceitos de API

- Uma API ajuda dois softwares diferentes a se comunicarem e trocarem dados entre si.
- Ajuda a incorporar conteúdo de qualquer site ou aplicação com mais eficiência.
- As APIs podem acessar os componentes do aplicação. A entrega de serviços e informações é mais flexível.

Stack

Conceitos de FastAPI

- FastAPI é um framework web moderno e rápido (de alto desempenho) para construir APIs com Python 3.7+ baseado em dicas de tipo Python padrão.
- Primeira versão lançada em 2018.
- Concorrentes: Django e Flask.
- Documentação automática.

Stack

Fonte da imagem: FastAPI

Conceitos de Container e Docker

- Um container é uma unidade padrão de software que vai empacotar o código e todas as suas dependências para que a aplicação rode rapidamente e sem problemas em diferentes ambientes.

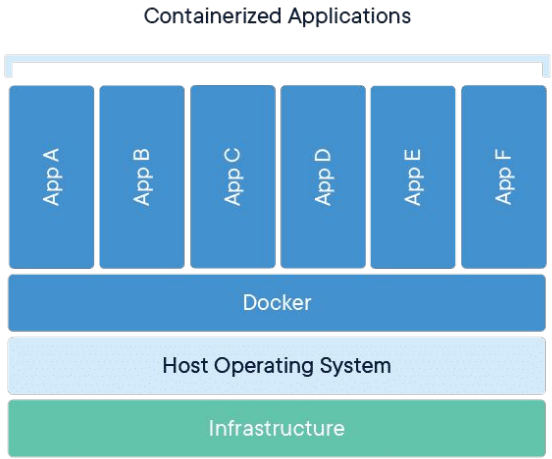
- Uma imagem de container é um pacote executável de código, leve e independente que inclui tudo o que é necessário para a aplicação funcionar: código, runtime, ferramentas e bibliotecas de sistema e configurações.

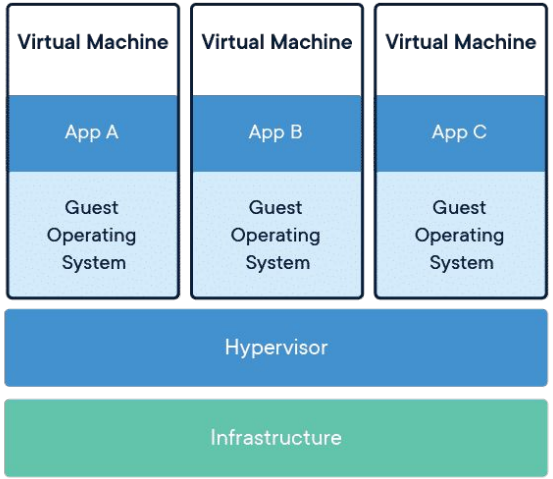
- Resolve o problema clássico: "Na minha máquina funciona!"

Stack

Conceitos de Container e Docker

- Containers vs VM.



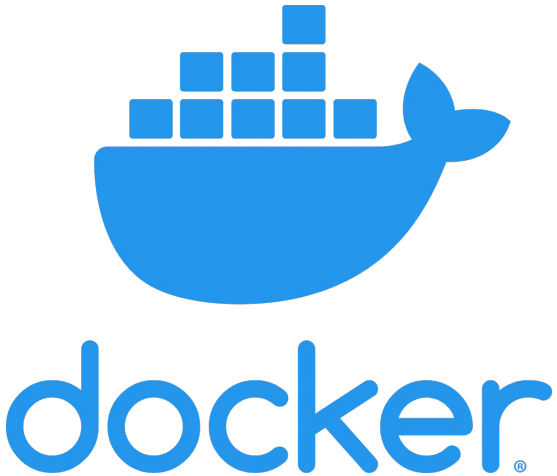


Stack

Fonte da imagem: Docker

Conceitos de Docker

- O Docker é uma plataforma como serviço que usa virtualização no nível do sistema operacional para entregar software em pacotes chamados contêineres. O serviço tem níveis gratuitos e premium. O software que hospeda os containers é chamado de Docker Engine



Stack

Fonte da imagem: Docker

Conceitos de Cloud Run

- O Google Cloud Run é uma plataforma totalmente gerenciada que permite que os usuários executem contêineres *stateless* que podem ser invocados por meio de eventos do Pub/Sub ou solicitações da Web. É um produto ***Serverless***, ou seja, abstrai todas as tarefas de gerenciamento de infraestrutura para que os usuários possam se concentrar na criação de aplicações.

Stack

Fonte da imagem: Whizlabs

Features do Cloud Run

- **Qualquer linguagem**: Os usuários podem usar qualquer linguagem de programação para construir seu aplicativo.
- **Serverless**: não há infraestrutura para gerenciar, portanto, se você implantou alguma aplicação, o Cloud Run gerenciará todos os serviços.
- **Auto-scaling**: aumenta ou diminui automaticamente com base no tráfego.
- **Domínios personalizados**: o Cloud Run mapeia os serviços para os domínios dos usuários.
- **Experiência do desenvolvedor**: Fornece uma linha de comando simples e interface de usuário que ajuda na rápida implantação e gerenciamento do serviço.
- **Redundância**: os serviços do Google Cloud Run são regionais e replicados automaticamente em várias zonas.

Fonte: K21Academy

Pricing do Cloud Run

Pricing for services with CPU only allocated during request processing

Tier	CPU	Memory	Requests[3]
Free	First 180,000 vCPU-seconds free per month	First 360,000 GiB-seconds free per month	2 million requests free per month
1	$0.00002400 / vCPU-second beyond free tier CUD[1]: $0.00001992 If kept idle[2]: $0.00000250	$0.00000250 / GiB-second beyond free tier CUD[1]: $0.000002075 If kept idle[2]: $0.00000250	$0.40 / million requests beyond free tier CUD[1]: $0.332
2	$0.00003360 / vCPU-second beyond free tier CUD[1]: $0.000027888 If kept idle[2]: $0.00000350	$0.00000350 / GiB-second beyond free tier CUD[1]: $0.000002905 If kept idle[2]: $0.00000350	$0.40 / million requests beyond free tier CUD[1]: $0.332

Stack

Fonte: GCP

Conceitos de Cloud Storage

- O Cloud Storage é um serviço para armazenar objetos na Google Cloud. Um objeto é uma parte imutável de dados que consiste em um arquivo de qualquer formato. Você armazena objetos em contêineres chamados **buckets**. Todos os **buckets** estão associados a um projeto e você pode agrupar seus projetos em uma organização. Cada projeto, bucket e objeto no Google Cloud é um recurso no Google Cloud, assim como instâncias do Compute Engine.

Fonte: GCP

Conceitos de Cloud Storage

Storage Class	Name for APIs and CLIs	Minimum storage duration	Typical monthly availability[1]
Standard storage	`STANDARD`	None	• >99.99% in multi-regions and dual-regions • 99.99% in regions
Nearline storage	`NEARLINE`	30 days	• 99.95% in multi-regions and dual-regions • 99.9% in regions
Coldline storage	`COLDLINE`	90 days	• 99.95% in multi-regions and dual-regions • 99.9% in regions
Archive storage	`ARCHIVE`	365 days	• 99.95% in multi-regions and dual-regions • 99.9% in regions

Stack

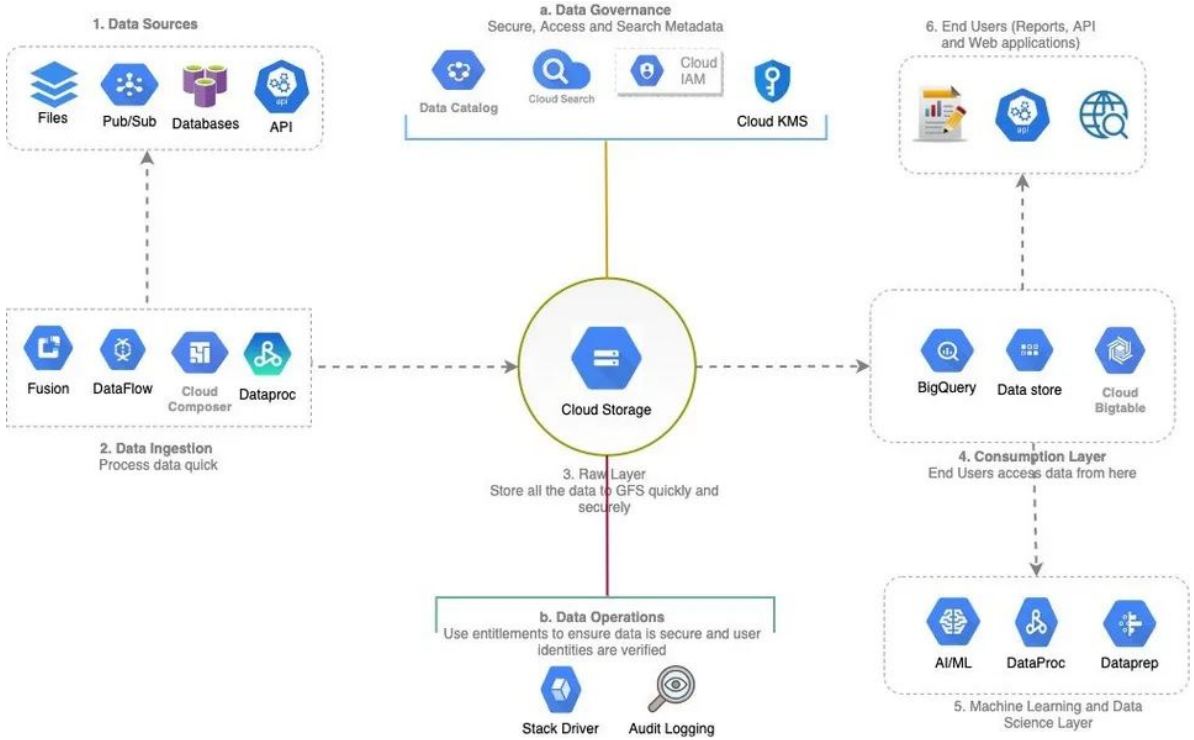
Fonte: GCP

Conceitos de Data Lake

- Um data lake é um repositório centralizado que permite que você armazene todos os seus dados estruturados e não-estruturados em qualquer escala. Você pode armazenar os seus dados sem ter que primeiro estruturá-los, e então executar diferentes tipos de análises: de dashboards e visualizações, a processamento de big data, análise em tempo real e aprendizado de máquina.

Stack

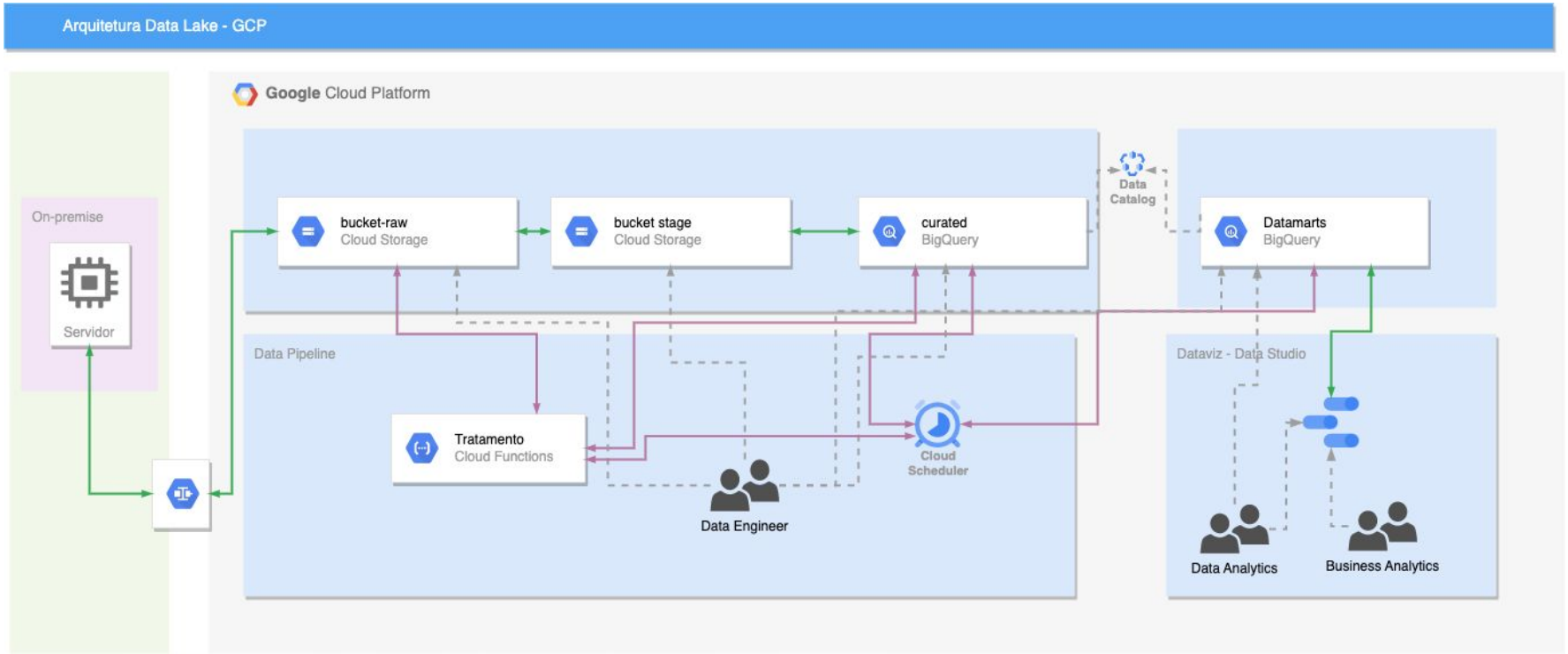
Conceitos de Data Lake



Stack

Fonte da imagem: [Unified Data Science]

Exemplo de Data Lake na GCP



Stack

Camada 1 - raw, bronze

- Esta camada consiste em 1 ou mais buckets que armazenam os dados vindos dos serviços de ingestão. É importante que os dados armazenados nessa camada sejam mantidos e preservados em sua forma original e que nenhuma transformação de dados ocorra.

Stack

Camada 2 - processed, staged, silver

- Esta camada armazena os datasets resultantes da transformação dos dados brutos da camada 1 em arquivos colunares (como Parquet, ORC, ou Avro) usando processamento ETL (por exemplo, Spark). Com os dados organizados em partições e em um formato colunar, os jobs de processamento conseguem ingerir estes dados com maior performance e menores custos.

Stack

Camada 3 - curated, enriched, gold

- Os dados armazenados nesta camada são um subgrupo da camada 2 que foram organizados para usos específicos. Os dados desta camada geralmente são acessados mais frequentemente por diversos stakeholders. Dependendo do uso de caso os dados podem ser servidos com diferentes tecnologias: BigQuery, AWS Redshift, Snowflake etc.

Stack

Particionamento em um Data Lake

- O particionamento de dados ajuda a reduzir os seus custos de processamento e a limitar a quantidade de dados escaneados pelas suas ferramentas de query (exemplo:BigQuery etc) para retornar o resultado da sua busca. É muito comum particionar por um timestamp (ano, mês, dia, hora, etc).

Stack

Prática:
Desenvolvimento da API
e deploy no Cloud Run.

Stack

Aula 3

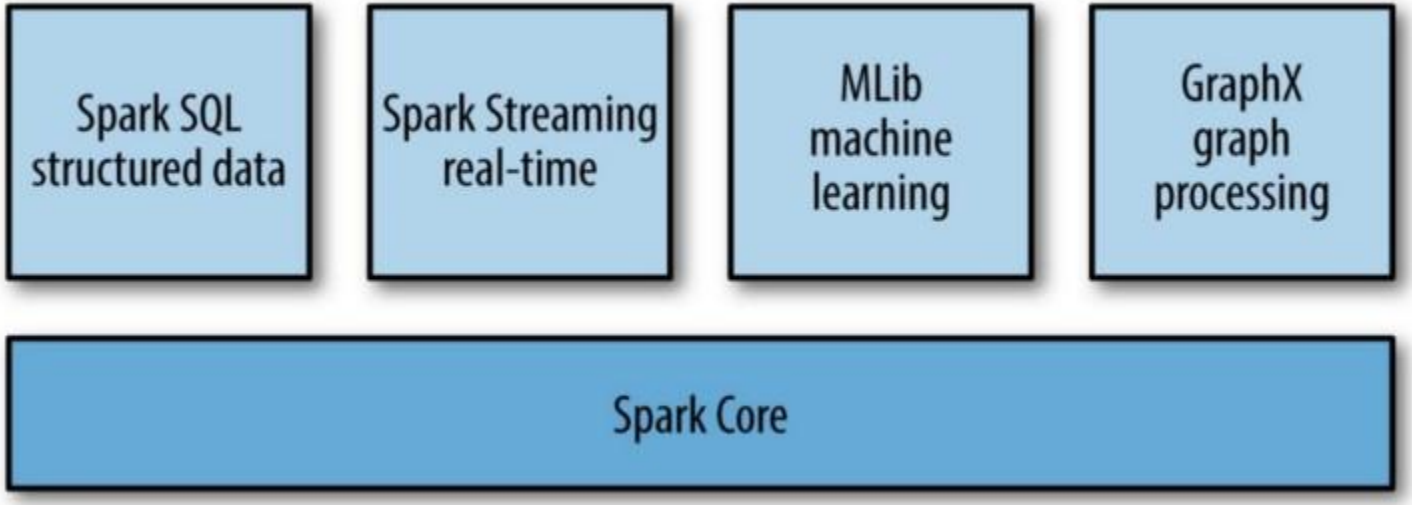
Stack

Conceitos de Spark

- O Apache Spark é uma ferramenta Big Data que tem o objetivo de processar grandes conjuntos de dados de forma paralela e distribuída. Ela estende o modelo de programação MapReduce popularizado pelo Apache Hadoop, facilitando bastante o desenvolvimento de aplicações de processamento de grandes volumes de dados. O Spark também apresenta uma performance muito superior ao Hadoop.

- Outra grande vantagem do Spark, é que todos os componentes funcionam integrados na própria ferramenta, como o Spark Streaming e o Spark SQL. Além disso, ele permite a programação em três linguagens: Java, Scala e Python.

Componentes do Spark



Stack

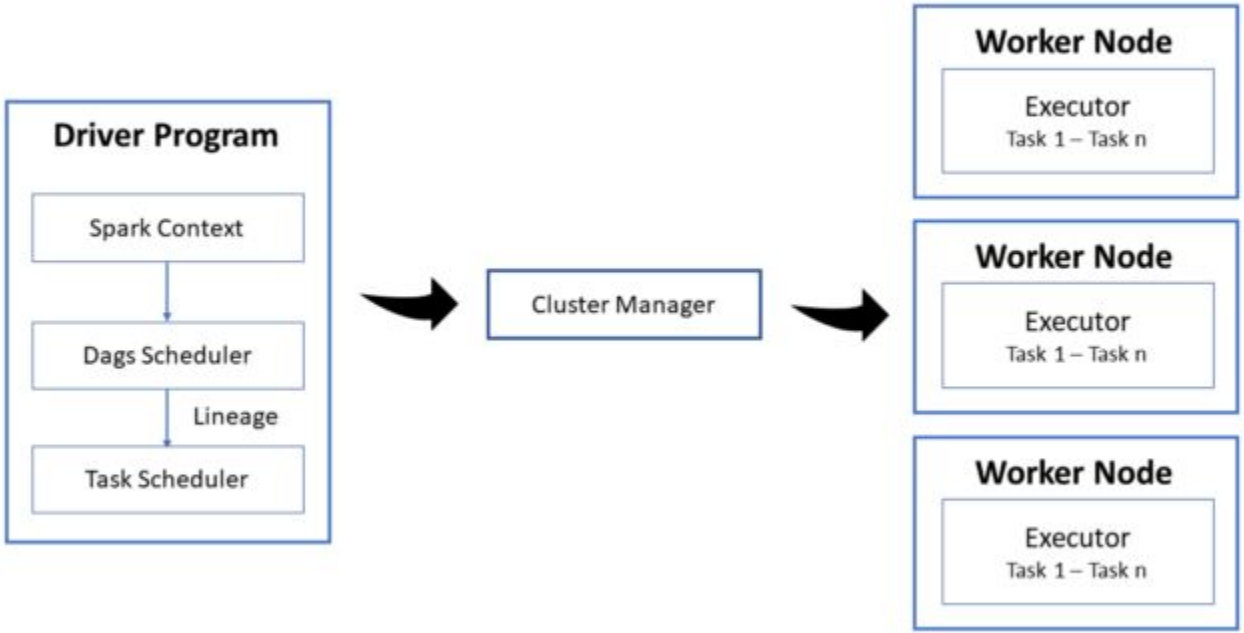
Arquitetura do Spark

- **Driver Program**: é o responsável por orquestrar a execução do processamento de dados.

- **Cluster Manager**: componente responsável por gerenciar as diversas máquinas em um cluster. Só é necessário se o spark for executado de forma distribuída.

- **Workers Nodes**: são as máquinas que executam as tarefas de um programa. Se o spark for executado de forma local na sua máquina, ela irá desempenhar tanto papel de Driver Program como de Workes. Esse modo de execução do spark é chamado de Standalone. Também existe o modo chamado Client mode, onde o Driver Program é executado na mesma máquina onde o programa foi iniciado. E por último existe o modo Cluster, que faz com que o Driver Program rode em qualquer máquina em um cluster.

Arquitetura do Spark



Stack

Spark vs Hadoop

Stack

Spark vs Hadoop

- **Hadoop MapReduce:**
  - Armazena o resultado das operações parciais e finais em disco.
  - Paradigma de desenvolvimento Map Reduce.

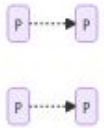
- **Spark:**
  - Armazena o resultado das operações parciais em memória e finais em disco (esse último configurável).
  - Paradigma de desenvolvimento Map Reduce entre outros.

Spark vs Hadoop

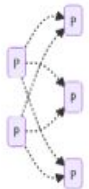
- Com isso, temos que o Spark apresenta vantagem no quesito velocidade de processamento. No entanto, o Spark não vem com um sistema de arquivos distribuídos nativo, ou seja, ele precisa ser integrado a algo que já existe. E é nesse ponto que entra sua dependência com o Hadoop. Em muitos casos o Spark usa o sistema de arquivos Hadoop HDFS (Hadoop Distributed File System). Portanto, o Spark substitui o Hadoop MapReduce, mas é complementar ao HDFS.

Operações no Spark

- **Transformações**:
  - **Narrow**: os dados necessários para serem computados para uma única partição existem nessa mesma partição. Exemplo: select, filter.



  - **Wide**: os dados necessários para serem computados para uma única partição podem existir em várias partições. Exemplo: group by, repartition.

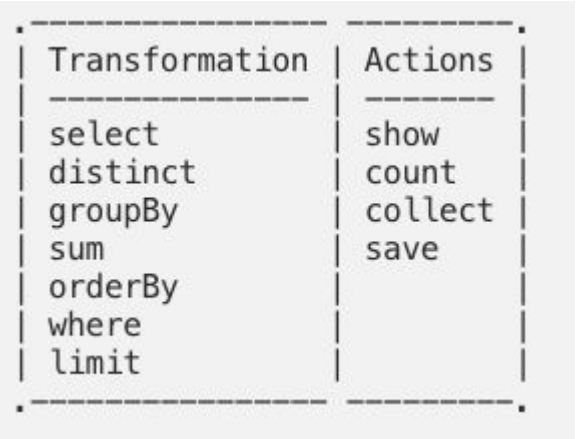


Stack

Fonte da imagem: Towards Data Science

Operações no Spark

- **Ações**:
  - Ações são declarações que solicitarão que um valor seja calculado imediatamente e são declarações ansiosas. Exemplo: show, count, collect.



Transformation	Actions
select	show
distinct	count
groupBy	collect
sum	save
orderBy	
where	
limit	

Stack

Fonte da imagem: Towards Data Science

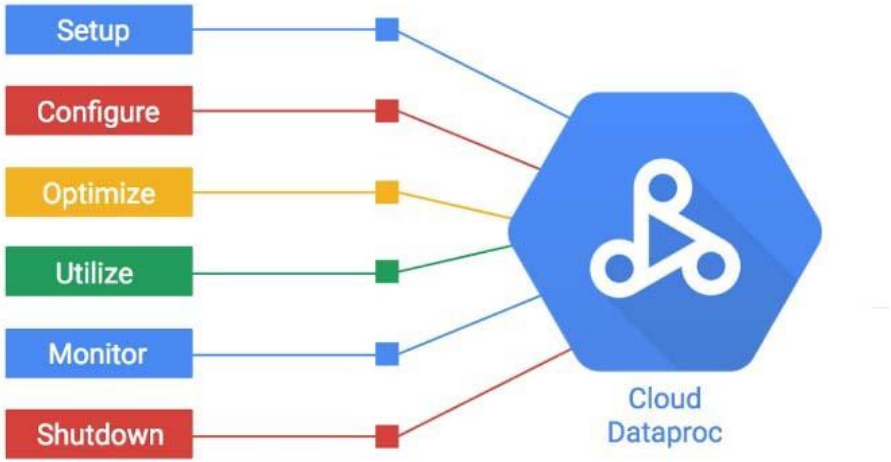
Lazy Evaluation

- Lazy Evaluation é uma estratégia de avaliação que atrasa a avaliação de uma expressão até que seu valor seja necessário.

- No Spark, Lazy Evaluation significa que você pode aplicar quantas **transformações** quiser, mas o Spark não iniciará a execução do processo até que uma **ação** seja chamada.

Conceitos de Dataproc

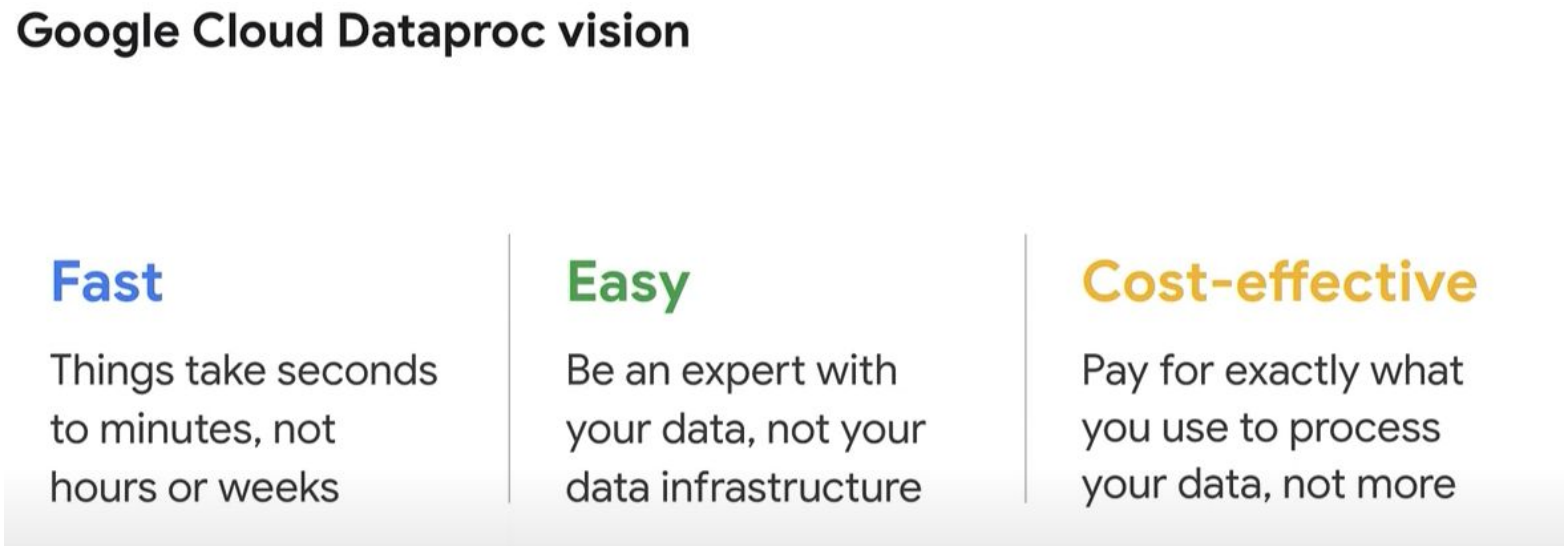
- O Dataproc é um serviço gerenciado do Spark e do Hadoop que permite aproveitar as ferramentas de dados de código aberto para processamento em lote, consultas, streaming e aprendizado de máquina. A automação do Dataproc ajuda a criar clusters rapidamente, gerenciá-los com facilidade e economizar dinheiro desativando os clusters quando não precisar deles.



Stack

Fonte da imagem: Analytics Vidhya

Conceitos de Dataproc



Stack

Fonte da imagem: GCP

Dataproc Serverless

- O Dataproc Serverless permite executar workloads batch em Spark sem exigir que um cluster. O serviço executará o workload em uma infraestrutura de computação gerenciada, dimensionando automaticamente os recursos conforme necessário. As cobranças do Dataproc Serverless se aplicam apenas ao momento em que o workload está em execução.

Dataproc Serverless

Capability	Dataproc Serverless for Spark	Dataproc on Compute Engine
Processing frameworks	Spark 3.2	Spark 3.1 and earlier versions. Other open source frameworks, such as, Hive
Serverless	Yes	No
Startup time	60s	90s
Infrastructure control	No	Yes
Resource management	Spark based	YARN based
GPU support	Planned	Yes
Interactive sessions	Planned (Google managed)	Yes (customer managed)
Custom containers	Yes	No
VM access (for example, SSH)	No	Yes
Java versions	Java 11	Previous versions supported
`OS Login` support *	No	Yes

Stack

Fonte da imagem: GCP

Pricing do Dataproc

As an example, consider a cluster (with master and worker nodes) that has the following configuration:

Item	Machine Type	Virtual CPUs	Attached persistent disk	Number in cluster
Master Node	n1-standard-4	4	500 GB	1
Worker Nodes	n1-standard-4	4	500 GB	5

This Dataproc cluster has 24 virtual CPUs, 4 for the master and 20 spread across the workers. For Dataproc billing purposes, the pricing for this cluster would be based on those 24 virtual CPUs and the length of time the cluster ran (assuming no nodes are scaled down or preempted). If the cluster runs for 2 hours, the Dataproc pricing would use the following formula:

```
Dataproc charge = # of vCPUs * hours * Dataproc price = 24 * 2 * $0.01 = $0.48
```

In this example, the cluster would also incur charges for Compute Engine and Standard Persistent Disk Provisioned Space in addition to the Dataproc charge (see Use of other Google Cloud resources). The billing calculator can be used to determine separate Google Cloud resource costs.

Stack

Fonte da imagem: GCP

Dataflow

- O Dataflow é um serviço de processamento de dados totalmente gerenciado para executar uma ampla variedade de padrões de processamento de dados.
- Suporta processamento Streaming e Batch com o mesmo código.
- Suporta operações de windowing.

Stack

Prática:
Criação do cluster Dataproc e desenvolvimento das etapas de tratamento.

Stack

Aula 4

Stack

Conceito de Airflow

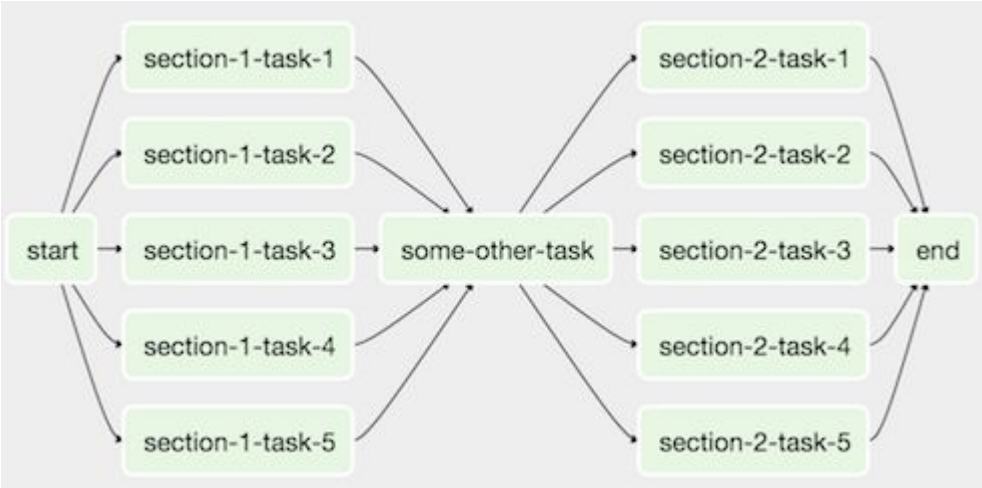
- Apache Airflow é uma plataforma para autoria, agendamento e monitoramento de workflows através de código. Os mesmos benefícios que vimos com IaC se aplicam aqui, quando workflows são definidos como código, eles ficam mais fáceis de manter, testar, colaborar e manter controle de versão.
- Use o Airflow para criar workflows como grafos acíclicos dirigidos (DAGs, directed acyclic graphs) de tarefas. O agendador do Airflow executa as suas tarefas em um conjunto de workers, respeitando as dependências especificadas.
- O Airflow também possui uma interface com o usuário que torna fácil visualizar as pipelines que estão rodando em produção, monitorar o progresso e resolver problemas quando necessário.

Stack

Apache Airflow

Conceito de Airflow - DAGs

- Uma DAG é a coleção de todas as tasks que você deseja rodar, organizadas de uma maneira que refletem os seus relacionamentos e dependências. Uma DAG é definida por um script Python, que representa a estrutura da DAG como código.



Stack

Fonte da imagem: Airflow

Conceito de Airflow - Tasks

- Uma Task define uma unidade de trabalho dentro de uma DAG. Ela é representada por um nó no grafo da DAG, e é também escrita em Python.
- Cada task é uma implementação de um Operator. Por exemplo, um PythonOperator vai executar determinado código Python. Já um BashOperator vai rodar um comando Bash.
- Uma task implementa um operator ao definir valores específicos para aquele operator, como uma função Python no caso do PythonOperator ou um comando Bash, no caso do BashOperator.
- Existem vários Operators diferentes já disponíveis, mas você também pode criar Operators customizados se for necessário.

Conceito de Airflow - Tasks

```
with DAG('my_dag', start_date=datetime(2016, 1, 1)) as dag:
    task_1 = DummyOperator('task_1')
    task_2 = DummyOperator('task_2')
    task_1 >> task_2 # Define dependencies
```

Stack

Fonte da imagem: Airflow

Cloud Composer

- O Cloud Composer é uma versão totalmente gerenciada do Apache Airflow, um serviço de orquestração de fluxo de trabalho. Pode ser usado para criação, agendamento, monitoramento e solução de problemas de fluxos de trabalho distribuídos. A integração com outros serviços do Google Cloud é outro recurso útil.

Stack

Fonte da imagem: Cloud Zone

Conceito de Data Warehouse

- Um data warehouse é um sistema corporativo usado para análise e relatório de dados estruturados e semiestruturados de várias fontes, como transações de ponto de venda, automação de marketing, gerenciamento de relacionamento com o cliente e muito mais. Um data warehouse é adequado para análises ad hoc e relatórios personalizados. Um data warehouse pode armazenar dados atuais e históricos em um só lugar e é projetado para fornecer uma visão de longo alcance dos dados ao longo do tempo, tornando-o um componente primário de inteligência de negócios.

Fonte da imagem: GCP

Conceito de BigQuery

- O BigQuery é um data warehouse totalmente gerenciado que ajuda gerenciar e analisar dados com recursos integrados, como machine learning, análise geoespacial e business intelligence. A arquitetura serverless do BigQuery permite usar consultas SQL para responder às maiores perguntas de negócio de uma organização sem gerenciamento de infraestrutura zero. O mecanismo de análise distribuída e escalável do BigQuery permite consultar terabytes em segundos e petabytes em minutos.

Stack

Fonte da imagem: GCP

Conceito de BigQuery

- Serverless.
- Escala de petabytes.
- Usa SQL mas não é uma base relacional.
- Base analítica.
- Outras features:
  - BQ ML.
  - BQ BI Engine.
  - BQ GIS.

Stack

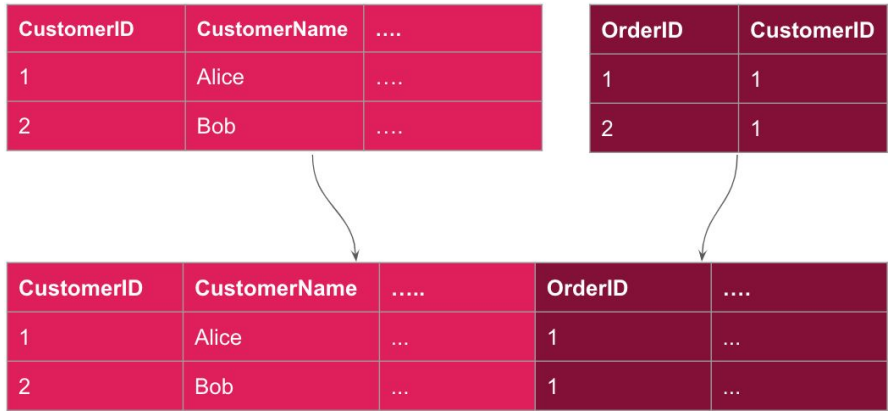
Conceito de BigQuery

- Datasets:
  - Coleção de tabelas e views.
- Tabelas:
  - Suporta estruturas escalares e aninhadas.
  - Armazenamento colunar.
  - Particionadas.
- Formas de realizar queries:
  - SQL UI.
  - commando ***bq***.
  - APIs.

Stack

Conceito de BigQuery - Desnormalização

- O BigQuery tem melhor desempenho quando os dados são desnormalizados. Em vez de manter relações, normalize os dados e aproveite os campos aninhados e repetidos. Campos aninhados e repetidos são suportados nos formatos Avro, Parquet, ORC, JSON (delimitado por nova linha). STRUCT é o tipo que pode ser usado para representar um objeto que pode ser aninhado e ARRAY é o tipo a ser usado para o valor repetido.



Stack

Fonte da imagem:
Towards Data Science

Conceito de BigQuery - Particionamento

- Uma tabela particionada é uma tabela especial dividida em segmentos, chamados de partições, que facilitam o gerenciamento e a consulta de dados. Ao dividir uma tabela grande em partições menores, pode-se melhorar o desempenho da consulta e controlar os custos reduzindo o número de bytes lidos por uma consulta.

- **Time-unit column**: Tables are partitioned based on a `TIMESTAMP`, `DATE`, or `DATETIME` column in the table.
- **Ingestion time**: Tables are partitioned based on the timestamp when BigQuery ingests the data.
- **Integer range**: Tables are partitioned based on an integer column.

Stack

Fonte: GCP

BigQuery - Comparações

	BigQuery	Redshift
Separação de storage e compute	Sim	Só para instâncias RA3
Elasticidade	Serverless. O BQ decide quantos “slots” alocar para cada consulta.	Disponível via “Elastic Resize”. Tempo de inatividade necessário.
Concorrência	Limitado a 100 usuários simultâneos por padrão	15 queries simultâneas por cluster, Escalonamento de até 10 clusters
Tuning	Nenhuma escolha sobre quantos slots são alocados para queries.	Escolha sobre número de nós e seu tipo.

Stack

Fonte: Firebolt

Pricing do BigQuery

- Explorar documentação: GCP

Stack

Prática:

Subida do Airflow com docker e desenvolvimento da DAG do pipeline.

Stack

Obrigado!

Stack