How to deploy the tvmaze pipeline

1. Clone the "deb-project2-group2" repository
2. Deploy Snowflake
3. Deploy Airbyte
4. Deploy DBT
5. Deploy Airflow

Note:

- This how-to has been written for windows users.
- Some prerequisites include:
  - Git
  - an AWS account
  - an IAM role with access id and key
  - a Snowflake account
  - Docker Desktop

1. Clone the deb-project2-group2 repository

1) Create a new folder on the C drive called "tvmaze"
2) Clone the repo from your local ~~CMD/~~Git Bash terminal. Run the following:
   - cd "C:\tvmaze"
   - git clone https://github.com/LuckyLukeAtGitHub/deb-project2-group2.git

2. Deploy Snowflake

1) In your browser navigate and login to your Snowflake account
   - Click "+ Worksheet"
   - In the query window run the sql script found at C:\tvmaze\deb-project2-group2\data-integration\snowflake\user_role_grants.sql

3. Deploy Airbyte

1) Create an AWS EC2 (Elastic Compute) instance. Follow the steps below
   - Navigate to EC2 -> Instances
   - Click "Launch Instance"
   - Name and tags: Input "airbyte-ec2-server"
   - Application and OS Images (Amazon Machine Image): Select Amazon Linux
   - Instance type: Select "t2.medium"
   - Key pair (login)
     - Create new key pair
       - Key pair name: Input "airbyte-login"
       - Key pair type: Select "RSA"
       - Private key file format: Select ".pem"
       - Click "Create key pair" and take note where you save the .pem file
   - ~~Network settings: Leave as default~~
   - ~~Configure storage: Leave as default~~
   - ~~Advance details: Leave as default~~
   - Click "Launch instance"

2) Add a firewall rule to your EC2 instance
   - Navigate to your EC2 -> Instances -> *<your_instance_id>*
   - Click "Security"
   - Click on your Security group
   - Click "Edit inbound rules"
   - Click "Add rule"
     - Type: Select "All TCP"
     - Source: Select "Anywhere-IPv4"
   - Click "Save rules"

3) Connect to your instance via your local Git Bash terminal and install Docker and Docker compose
   - 3.1) Set a file path variable to where you downloaded the .perm file. Run the following:
     - SSH_KEY="*<the_path_to_your_pem_file>*"
   - 3.2) Set the instance IP address variable. Run the following:
     - INSTANCE_IP=*<your_ec2_instance_auto-assigned_IP_address>*
   - 3.3) Set the ownership and permission. Run the following:
     - chmod 400 $SSH_KEY
   - 3.4) Connect to the AWS EC2 instance. Run the following:
     - ssh -i $SSH_KEY ec2-user@$INSTANCE_IP
   - 3.5) When prompted, Are you sure you want to continue connecting (yes/no/[fingerprint])? Input "yes"
   - 3.6) Install Docker on the EC2 instance using a SSH session. Run the following:
     - sudo yum update -y
     - sudo yum install -y docker
     - sudo service docker start
     - sudo usermod -a -G docker $USER
   - 3.7) Enable docker to auto start on boot: Run the following:
     - sudo systemctl enable docker
   - 3.8) Install docker-compose using the SSH session. Run the following:
     - sudo wget https://github.com/docker/compose/releases/download/1.26.2/docker-compose-$(uname -s)-$(uname -m) -O /usr/local/bin/docker-compose
     - sudo chmod +x /usr/local/bin/docker-compose
     - docker-compose --version
     - logout

4) Download and start Airbyte using a new SSH session
   - 4.1) Connect to the AWS EC2 instance. Run the following:
     - ssh -i $SSH_KEY ec2-user@$INSTANCE_IP
   - 4.2) Download Airbyte: Run the following:
     - mkdir airbyte
     - cd airbyte
     - wget https://raw.githubusercontent.com/airbytehq/airbyte/master/{.env,docker-compose.yaml}
   - 4.3) Start Airbyte. Run the following:
     - docker-compose up -d
   - 4.4) Enable docker containers to auto start on next boot. Run the following:
     - docker update --restart unless-stopped $(docker ps -q)
     - logout

5) Connect to Airbyte to confirm it is running
   10.1) In your browser navigate to <*your_ec2_instance_auto-assigned_IP_address*>:8000
   10.2) If prompted for login credentials:
      - Username: Input "airbyte"
      - Password: Input "password"
   10.3) On the preferences form
      - Your email: Input <*your_email_address*>

6) Create an AWS ECR (Elastic Container Registry) for the Airbyte custom connector
   - Navigate to ECR > Repositories
   - Click "Create repository"
   - Visibility settings: Select "Private"
   - Repository name: Input "tvmaze-airbyte-connector-ecr"
   - ~~Tag immutability: Leave as Disabled~~
   - ~~Image scan settings->Scan on push: Leave as Disabled~~
   - ~~Encryption settings->KMS encryption: Leave as Disabled~~
   - Click "Create repository"

7) Authenticate to your tvmaze-airbyte-connector-ecr ECR
   7.1) Open your local ~~CMD/~~Git Bash terminal and run the following:
      - aws configure
         - AWS Access Key ID: Input <*your_access_key_id*>
         - AWS Secret Access Key: Input <*your_secret_access_key*>
         - Default region name: Input <*your_aws_region*>
         - Default output format: Hit enter
      - aws ecr get-login-password --region <*your_aws_region*> | docker login --username AWS --password-stdin <*your_tvmaze-airbyte-connector-ecr_uri*>

8) Clone the repo from your local CMD/Git Bash terminal. Run the following:
   - cd "C:\tvmaze"
   - git clone https://github.com/airbytehq/airbyte.git
   - Copy the
   - cp -r "C:\tvmaze\deb-project2-group2\data-integration\airbyte\source-tvmaze" "C:\tvmaze\airbyte\airbyte-integrations\connectors"

9) Build, Tag and Push the tvmaze connector to the ECR
   9.1) Ensure you Docker Desktop is running on your PC
   9.2) Build your connector docker image. Run the following:
      - cd " C:\tvmaze\airbyte\airbyte-integrations\connectors\source-tvmaze"
      - docker build . -t airbyte/source-tvmaze:dev
   9.3) Tag your connector docker image for AWS: Run the following:
      - docker tag <*your_local_image_name*>:dev <*your_tvmaze-airbyte-connector-ecr_uri*>:dev
   9.4) Push your connector docker image. Run the following:
      - docker push <*your_tvmaze-airbyte-connector-ecr_uri*>:dev

10) Connect to your instance via your local Git Bash terminal and pull your connector to the EC2 instance
    - 10.1) Set a file path variable to where you downloaded the .perm file. Run the following:
        - ➢ SSH_KEY="<*the_path_to_your_pem_file*>"
    - 10.2) Set the instance IP address variable. Run the following:
        - ➢ INSTANCE_IP=<your_ec2_instance_auto-assigned_IP_address>
    - 10.3) Set the ownership and permission. Run the following:
        - ➢ chmod 400 $SSH_KEY
    - 10.4) Connect to the AWS EC2 instance. Run the following:
        - ➢ ssh -i $SSH_KEY ec2-user@$INSTANCE_IP
    - 10.5) If prompted, Are you sure you want to continue connecting (yes/no/[fingerprint])? Input "yes"
    - 10.6) Authenticate to your AWS: Run the following:
        - ➢ aws configure
            - ➢ AWS Access Key ID: Input <*your_access_key_id*>
            - ➢ AWS Secret Access Key: Input <*your_secret_access_key*>
            - ➢ Default region name: Input <*your_aws_region*>
            - ➢ Default output format: Hit enter
        - ➢ aws ecr get-login-password --region <*your_aws_region*> | docker login --username AWS --password-stdin <*your_ecr_uri*>
    - 10.7) Pull your connector docker image. Run the following:
        - ➢ docker pull <*your_ecr_uri*>:dev

11) Add the tvmaze custom connector via the Airbyte UI
    - 10.1) In your browser navigate to <*your_ec2_instance_auto-assigned_IP_address*>:8000
    - 10.2) If prompted for login credentials
        - ➢ Username: Input "airbyte"
        - ➢ Password: Input "password"
    - 10.3) Add the custom tvmaze connector
        - ➢ Click "Settings"
        - ➢ Click "Sources"
        - ➢ Click "New connector"
            - ➢ Connector display name: Input "my-custom-tvmaze-source-connector"
            - ➢ Docker repository name: Input <*your_ecr_uri*>
            - ➢ Docker image tag: Input "dev"
            - ➢ Connector Documentation URL: Input "https://www.tvmaze.com/api#schedule"
            - ➢ Click "Add"

12) Add the tvmaze Source in Airbyte
    - ➢ Click "Sources"
    - ➢ Click "Connect your first source"
    - ~~➢ Click "+ New source"~~
        - ➢ Source type: Search and select "tvmaze"
        - ➢ Source Type: Leave as "tvmaze"
        - ➢ Source name: Input "tvmaze_api"
        - ➢ country: Input "US"
        - ➢ start_date: Input "2022-01-01"
        - ➢ Click "Set up source"

13) Add the Snowflake Destination in Airbyte

- Click "Destinations"
- Click "+ New destination"
  - Destination type: Search and select "Snowflake"
  - Destination name: Input "Snowfake"
  - Host: Input *<your_snowflake_account_locator>*
  - Role: Input "AIRBYTE_ROLE"
  - Warehouse: Input "ETL_P2"
  - Database: Input "TVSHOW"
  - Default Schema: Input "SOURCE"
  - Username: Input "airbyte_user"
  - Authorization Method: Select Username and Password
  - Password: Input "project2tvshows"
  - JDBC URL Params: Leave blank
  - Data Staging Method: Select [Recommended] Internal Staging
  - Click "Set up destination"

14) Create a Connection in Airbyte

- Click "Connections"
- Click "+ New connection"
  - Select "tvmaze_api" from an existing source
  - Click "User existing source"
  - Select "Snowflake" from an existing destination
  - Click "User existing destination"
  - Select "Manual" for Transfer – Replication frequency
  - Select "id" for Activate the streams you want to sync - Primary key
  - Click "Set up connection"

4. Deploy DBT

1) Create an environment variable file called "dbt.env" with the following

```
DBT_ENV_SECRET_ACCOUNT=<your_snowflake_account>.<your_snowflake_region>
DBT_ENV_SECRET_USER=DBT_P2
DBT_ENV_SECRET_PASSWORD=project2tvshows
DBT_ENV_SECRET_ROLE=DBT_RW_P2
DBT_ENV_SECRET_WAREHOUSE=ETL_P2
DBT_ENV_SECRET_DATABASE=TVSHOW
DBT_ENV_SECRET_SCHEMA=MODEL
```

2) Create a S3 bucket. Follow the steps below
   - ➢ Navigate to S3 > Buckets
   - ➢ Click "Create bucket"
   - ➢ General configuration
     - ➢ Bucket name: Input "tvmaze-s3-bucket"
     - ➢ AWS Region: Select Asia Pacific (Sydney) ap-southeast-2
   - ➢ Object Ownership
     - ➢ Select ACLs disabled (recommended)
   - ➢ Block Public Access settins for this bucket
     - ➢ Uncheck "Block all public access"
     - ➢ Check "I acknowledge that the current settings might result in this bucket and the objects within becoming public."
   - ➢ Click "Create Bucket"
   - ➢ Click on the newly created "tvmaze-s3-bucket" bucket
   - ➢ Click "Create folder"
   - ➢ Folder
   - ➢ Folder name: Input "env"
   - ➢ Click "Create folder"
   - ➢ Click on the newly created "env" folder
   - ➢ Upload the dbt.env file

3) Create an IAM Role for the dbt ECR to read the env file
   - ➢ Navigate to IAM > Roles
   - ➢ Click "Create role"
   - ➢ Trusted entity type
     - ➢ Select "AWS service"
   - ➢ Use case
     - ➢ Search "Elastic Container Service"
     - ➢ Select "Elastic Container Service Task"
   - ➢ Click "Next
   - ➢ Search for "AmazonECSTaskExecutionRolePolicy" and check the checkbox
   - ➢ Click "Next"
   - ➢ Role Details
     - ➢ Role Name: Input "tvmaze-dbt-ecs-task-s3-env-role"
   - ➢ Click "Create role"
   - ➢ Click on the newly created "tvmaze-dbt-ecs-task-s3-env-role" role
   - ➢ Click "Add permissions"
   - ➢ Select "Create inline policy"
   - ➢ Click the "JSON" tab
   - ➢ Delete any json in the code block

- Insert the following:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:GetBucketLocation"
      ],
      "Resource": [
        "<your_dbt.env_arn>"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetBucketLocation"
      ],
      "Resource": [
        "<your_tvmaze-s3-bucket_arn>"
      ]
    }
  ]
}
```

- Click "Review policy"
- Name: Input "ReadTvmazeBucketEnvPolicy"
- Click "Create policy"

4) Create an AWS ECR (Elastic Container Registry) for the dbt
   - Navigate to ECR > Repositories
   - Click "Create repository"
   - Visibility settings: Select "Private"
   - Repository name: Input "tvmaze-dbt-ecr"
   - Tag immutability: Leave as Disabled
   - Image scan settings->Scan on push: Leave as Disabled
   - Encryption settings->KMS encryption: Leave as Disabled
   - Click "Create repository"

5) Create an ECS Cluster
   - Navigate to ECS > Clusters
   - Click "Create Cluster"
   - Select "EC2 Linux + Networking"
   - Click "Next step"
   - Configure cluster
     - Cluster name: Input "tvmaze-dbt-ecs-cluster"
   - Instance configuration
     - Select "On-Demand Instance"
     - EC2 instance type: Select "t2.micro"
     - Number of instances: Input "1"
   - Networking

        - VPC: Select the existing VPC
        - Subnets: Select any existing subnet
        - Security group: Select the existing default security group
        - Click "Create"

6) Create a ECS Task Definition
    - Navigate to ECS > Task Definitions
    - Click "Create new Task Definition"
    - Select "EC2"
    - Click "Next step"
    - Configure task and container definitions
        - Task definition name: Input "tvmaze-dbt-ecs-task-s3-env-role "
        - Task role: Select "tvmaze-dbt-ecs-task-s3-env-role"
        - Network mode: Leave as default
    - Task execution IAM role
        - Task execution role: Select "tvmaze-dbt-ecs-task-s3-env-role"
    - Task size
        - Task memory: Input "128"
        - Task CPU (unit): Input "1 vCPU"
    - Container definitions
        - Click "Add container"
            - Standard
                - Container name: Input "tvmaze-dbt-ecr-container"
                - Image: *<your_tvmaze-dbt-ecr_uri>*
            - Advanced container configuration
            - Environment Files
                - Click the + sign
                - Location: Input <your_dbt.env_arn>
            - Log configuration
                - Check "Auto-configure CloudWatch Logs"
        - Click "Add"
        - Click "Create"

7) Authenticate to your private Docker registry

    7.1) Open your local CMD terminal and run the following:
    - aws configure
        - AWS Access Key ID: Input <your_access_key_id>
        - AWS Secret Access Key: Input <your_secret_access_key>
        - Default region name: Input <your_aws_region>
        - Default output format: Hit enter
    - aws ecr get-login-password --region <your_aws_region> | docker login --username AWS --password-stdin <*your_tvmaze-dbt-ecr_uri*>

8) Build, Tag and Push the dbt image to the ECR
    8.1) Ensure you Docker Desktop is running on your PC
    8.2) Build your connector docker image. Run the following:
        - cd C:\tvmaze\deb-project2-group2\data-transformation\dbt
        - docker build -t tvmaze-dbt:dev -f docker/Dockerfile .
    8.3) Tag your connector docker image for AWS: Run the following:
        - docker tag <*your_local_image_name*>:dev <*your_tvmaze-dbt-ecr_uri*>:dev
    8.4) Push your connector docker image. Run the following:
        - docker push <*your_tvmaze-dbt-ecr_uri*>:dev

5. Deploy Airflow

9) Create a slack app
    9.1) In your browser navigate to https://api.slack.com/apps
        - Click "Create New App"
        - Click "From scratch"
        - App Name: Input "tvmaze-etl-notification"
        - Pick a workspace to develop your app in: Select your workspace
        - Click Create App
        - Click "Incoming Webhooks" on the left nav bar
        - Click slider to enable "Activate Incoming Webhooks"
        - Click "Add New Webhook to Workspace"
        - Select channel for the app to post in
        - Click "Allow"
        - Take note of your slack password from the Webhook URL https://hooks.slack.com/services/<*your_slack_password*>

10) Build and run Airflow locally
    10.1) Ensure you Docker Desktop is running on your PC
    10.2) Open your local CMD terminal and run the following
        - cd C:\tvmaze\deb-project2-group2\data-ochestration\airflow
        - docker build -t airflow .
        - docker run -p 8080:8080 -v "%cd%:/opt/airflow" airflow standalone

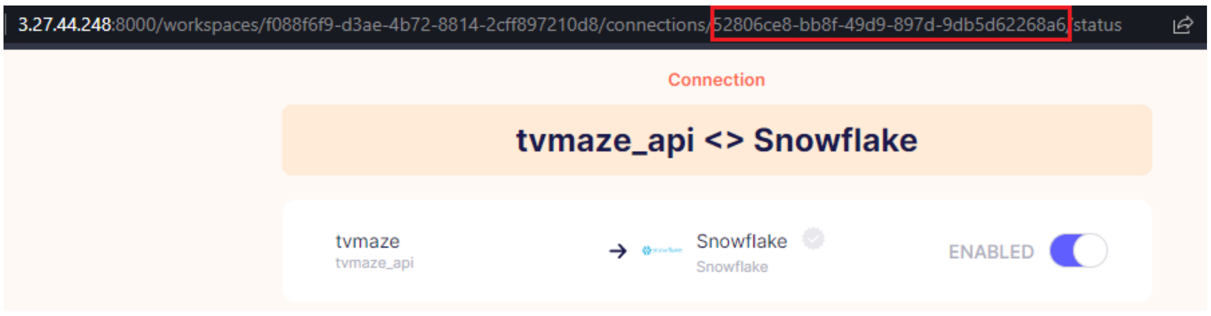
11) Log into the UI and add connections
    11.1) Navigate to "C:\tvmaze\deb-project2-group2\data-ochestration\airflow\standalone_admin_password.txt" to get the password
    11.2) In your browser go to http://localhost:8080/
        - Username: Input "admin"
        - Password: Input <*your_password_found_in_standalone_admin_password.txt*>

12) Add connections in the UI

- Click Admin
- Click connections
- Click the (+) plus sign to Add the tvmave-airbyte-connection
  - Connection Id: Input "tvmaze-airbyte-connection"
  - Connection Type: Select "Airbyte"
  - Host: Input *<your_ec2_instance_auto-assigned_IP_address>*
  - Login: Input "airbyte"
  - Password: Input "password"
  - Port: Input "8001"
- Click the (+) plus sign to Add the aws_login_for_ecs-task
  - Connection Id: Input "aws-login-for-ecs-task"
  - Connection Type: Select Amazon Web Services
  - AWS Access Key ID: Input *<your_access_key_id>*
  - AWS Secret Access Key*:* Input *<your_secret_access_key>*
  - Extra: Input {"region_name": "ap-southeast-2"}
- Click the (+) plus sign to Add the tvmaze-slack-connection
  - Connection Id: Input "tvmaze-slack-connection"
  - Connection Type: Select "HTTP"
  - Host: Input "https://hooks.slack.com/services"
  - Password: Input *<your_slack_password>*

13) Add Variables in the UI

- Click Admin
- Click Variables
- Click the (+) plus sign to Add a new record
  - Key: Input "airbyte_tvmazeapisf_conn_id"
  - Val: Input *<your_airbyte_connection_id>*


3.27.44.248:8000/workspaces/f088f6f9-d3ae-4b72-8814-2cff897210d8/connections/52806ce8-bb8f-49d9-897d-9db5d62268a6/status

Connection

tvmaze_api <> Snowflake

tvmaze
tvmaze_api
Snowflake
Snowflake
ENABLED

14) Run the pipeline

15)