Demo 4: Docker Swarm Advanced Features

Prerequisites:

- Launch two EC2 instances, one of which will be used to initialise the swarm cluster and will be assigned as manager. Another will be called a worker. **Keep port 22, 80, 8080 and 443** open for all IPs to access the instances from your local machine over the internet.

- Docker swarm requires **TCP port 2376, TCP port 2377, TCP & UDP port 7946 and UDP port 4789** to communicate among nodes. Keep all ports open for the same security group.

- **Node draining**
  - docker node ls
  - docker service create --name webserver --replicas 5 nginx
  - docker service ps webserver

```
ubuntu@ip-172-31-5-46:~$ docker service ps webserver
ID             NAME          IMAGE          NODE              DESIRED STATE   CURRENT STATE            ERROR     PORTS
drzgpqngahas   webserver.1   nginx:latest   ip-172-31-5-46    Running         Running 29 seconds ago
vlt10u00zknn   webserver.2   nginx:latest   ip-172-31-1-160   Running         Running 28 seconds ago
lbd4iiovszio   webserver.3   nginx:latest   ip-172-31-5-46    Running         Running 29 seconds ago
p4ei5mohmkq2   webserver.4   nginx:latest   ip-172-31-1-160   Running         Running 28 seconds ago
c2c2n8kkppkb   webserver.5   nginx:latest   ip-172-31-1-160   Running         Running 28 seconds ago
ubuntu@ip-172-31-5-46:~$
```

  - docker node update --availability drain ip-172-31-1-160
  - docker service ps webserver

```
ubuntu@ip-172-31-5-46:~$
ubuntu@ip-172-31-5-46:~$ docker service ps webserver
ID             NAME             IMAGE          NODE              DESIRED STATE   CURRENT STATE                ERROR     PORTS
drzgpqngahas   webserver.1      nginx:latest   ip-172-31-5-46    Running         Running 5 minutes ago
qu0ym1eajlnw   webserver.2      nginx:latest   ip-172-31-5-46    Running         Running about a minute ago
vlt10u00zknn    \_ webserver.2   nginx:latest   ip-172-31-1-160   Shutdown        Shutdown about a minute ago
lbd4iiovszio   webserver.3      nginx:latest   ip-172-31-5-46    Running         Running 5 minutes ago
guheqtq25j6i   webserver.4      nginx:latest   ip-172-31-5-46    Running         Running about a minute ago
p4ei5mohmkq2    \_ webserver.4   nginx:latest   ip-172-31-1-160   Shutdown        Shutdown about a minute ago
unmh49pcu2lx   webserver.5      nginx:latest   ip-172-31-5-46    Running         Running about a minute ago
c2c2n8kkppkb    \_ webserver.5   nginx:latest   ip-172-31-1-160   Shutdown        Shutdown about a minute ago
ubuntu@ip-172-31-5-46:~$
```

  - docker node update --availability active ip-172-31-1-160

```
ubuntu@ip-172-31-5-46:~$
ubuntu@ip-172-31-5-46:~$ docker node ls
ID                            HOSTNAME          STATUS    AVAILABILITY   MANAGER STATUS   ENGINE VERSION
wtf9dy2ivefbldna30oiq9qvo     ip-172-31-1-160   Ready     Drain                           20.10.4
xae2pz1d1o8ho4boxpifwt7ut *   ip-172-31-5-46    Ready     Active         Leader           20.10.4
ubuntu@ip-172-31-5-46:~$ docker node update --availability active ip-172-31-1-160
ip-172-31-1-160
ubuntu@ip-172-31-5-46:~$ docker node ls
ID                            HOSTNAME          STATUS    AVAILABILITY   MANAGER STATUS   ENGINE VERSION
wtf9dy2ivefbldna30oiq9qvo     ip-172-31-1-160   Ready     Active                          20.10.4
xae2pz1d1o8ho4boxpifwt7ut *   ip-172-31-5-46    Ready     Active         Leader           20.10.4
ubuntu@ip-172-31-5-46:~$
```

- **Inspecting**
  - docker service inspect webserver

- docker service inspect webserver --pretty

```
ubuntu@ip-172-31-5-46:~$ docker service inspect webserver --pretty

ID:             ku025t37dwqdjh6wpatmjy66q
Name:           webserver
Service Mode:   Replicated
 Replicas:      5
Placement:
UpdateConfig:
 Parallelism:   1
 On failure:    pause
 Monitoring Period: 5s
 Max failure ratio: 0
 Update order:      stop-first
RollbackConfig:
 Parallelism:   1
 On failure:    pause
 Monitoring Period: 5s
 Max failure ratio: 0
 Rollback order:    stop-first
ContainerSpec:
 Image:         nginx:latest@sha256:f3693fe50d5b1df1ecd315d54813a77afd56b0245a404055a946574deb6b34fc
 Init:          false
Resources:
Endpoint Mode:  vip

ubuntu@ip-172-31-5-46:~$
```

- docker node inspect ip-172-31-1-160 --pretty

```
ubuntu@ip-172-31-5-46:~$ docker node inspect ip-172-31-1-160 --pretty
ID:                     wtf9dy2ivefbldna30oiq9qvo
Hostname:               ip-172-31-1-160
Joined at:              2021-02-28 12:36:26.020829527 +0000 utc
Status:
 State:                 Ready
 Availability:          Active
 Address:               172.31.1.160
Platform:
 Operating System:      linux
 Architecture:          x86_64
Resources:
 CPUs:                  1
 Memory:                1.939GiB
Plugins:
 Log:           awslogs, fluentd, gcplogs, gelf, journald, json-file, local, logentries
 Network:               bridge, host, ipvlan, macvlan, null, overlay
 Volume:                local
Engine Version:         20.10.4
TLS Info:
 TrustRoot:
-----BEGIN CERTIFICATE-----
```

- **Node constraints**
  - docker service create --name example --constraint node.labels.region==india --replicas 3 nginx
  - docker ps webserver

```
ubuntu@ip-172-31-5-46:~$
ubuntu@ip-172-31-5-46:~$ docker service ps webserver
ID             NAME          IMAGE          NODE      DESIRED STATE   CURRENT STATE           ERROR                                PORTS
vfu5kb1frqro   webserver.1   nginx:latest             Running         Pending 3 minutes ago   "no suitable node (scheduling …"
v4bxd3i3lsdr   webserver.2   nginx:latest             Running         Pending 3 minutes ago   "no suitable node (scheduling …"
8joutz9m9i0q   webserver.3   nginx:latest             Running         Pending 3 minutes ago   "no suitable node (scheduling …"
ubuntu@ip-172-31-5-46:~$
ubuntu@ip-172-31-5-46:~$
```

- docker node update --label-add region=india ip-172-31-1-160
- docker service ps webserver

```
ubuntu@ip-172-31-5-46:~$
ubuntu@ip-172-31-5-46:~$ docker node update --label-add region=india ip-172-31-1-160
ip-172-31-1-160
ubuntu@ip-172-31-5-46:~$ docker service ps webserver
ID             NAME          IMAGE          NODE              DESIRED STATE   CURRENT STATE          ERROR     PORTS
vfu5kb1frqro   webserver.1   nginx:latest   ip-172-31-1-160   Running         Running 1 second ago
v4bxd3i3lsdr   webserver.2   nginx:latest   ip-172-31-1-160   Running         Running 1 second ago
8joutz9m9i0q   webserver.3   nginx:latest   ip-172-31-1-160   Running         Running 1 second ago
ubuntu@ip-172-31-5-46:~$
ubuntu@ip-172-31-5-46:~$
ubuntu@ip-172-31-5-46:~$
```

Demo 5: Voting Application (Placement Constraints)

The following stack file can be used to deploy docker voting applications with node constraints.

We will deploy a voting application that contains five microservices using the docker swarm stack file. We will create two overlay networks and attach the respective services to those networks, exposing only the frontend services to the host port. We will set environment variables, restart and update policies in the stack file itself. We will put some node constraints for certain services in the stack file. After deploying the application, check where each container is running and understand the use of each keyword in the stack file.

docker-stack-simple.yml

```
version: "3"
services:

  redis:
    image: redis:alpine
    ports:
      - "6379"
    networks:
      - frontend
    deploy:
      replicas: 1
      update_config:
        parallelism: 2
        delay: 10s
      restart_policy:
        condition: on-failure
  db:
    image: postgres:9.4
    environment:
      POSTGRES_USER: "postgres"
      POSTGRES_PASSWORD: "postgres"
    volumes:
      - db-data:/var/lib/postgresql/data
    networks:
```

```
    - backend
  deploy:
    placement:
      constraints: [node.role == manager]
vote:
  image: dockersamples/examplevotingapp_vote:before
  ports:
    - 5000:80
  networks:
    - frontend
  depends_on:
    - redis
  deploy:
    replicas: 1
    update_config:
      parallelism: 2
    restart_policy:
      condition: on-failure
result:
  image: dockersamples/examplevotingapp_result:before
  ports:
    - 5001:80
  networks:
    - backend
  depends_on:
    - db
  deploy:
    replicas: 1
    update_config:
      parallelism: 2
      delay: 10s
    restart_policy:
      condition: on-failure

worker:
  image: dockersamples/examplevotingapp_worker
  networks:
    - frontend
    - backend
  depends_on:
    - db
    - redis
  deploy:
```

```
    mode: replicated
    replicas: 1
    labels: [APP=VOTING]
    restart_policy:
      condition: on-failure
      delay: 10s
      max_attempts: 3
      window: 120s
    placement:
      constraints: [node.role == manager]

networks:
  frontend:
  backend:

volumes:
  db-data:
```

Run the following command to deploy the application.

```
docker stack deploy -c docker-stack-simple.yml
```