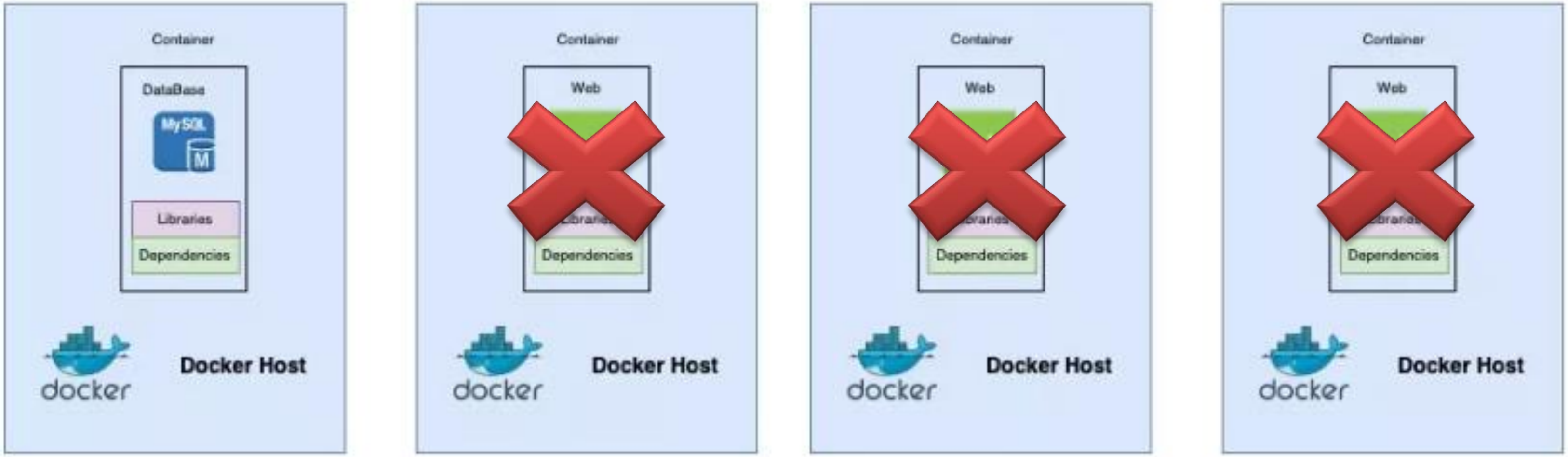
Docker - Orchestration

Orchestration – Getting Started

- Container Life Cycle Management
- Container deployment and start, stop, remove containers
- Scale up and scale down as per the requirement
- Container migration from one host to another in case required
- Load balancing
- Automatic Scheduling
- Monitoring

- E.g If 4 instance of container application is running on 4 nodes
- What if,
  - Container gets removed
  - Node gets down
  - Container application failed to start etc

Problem Statement

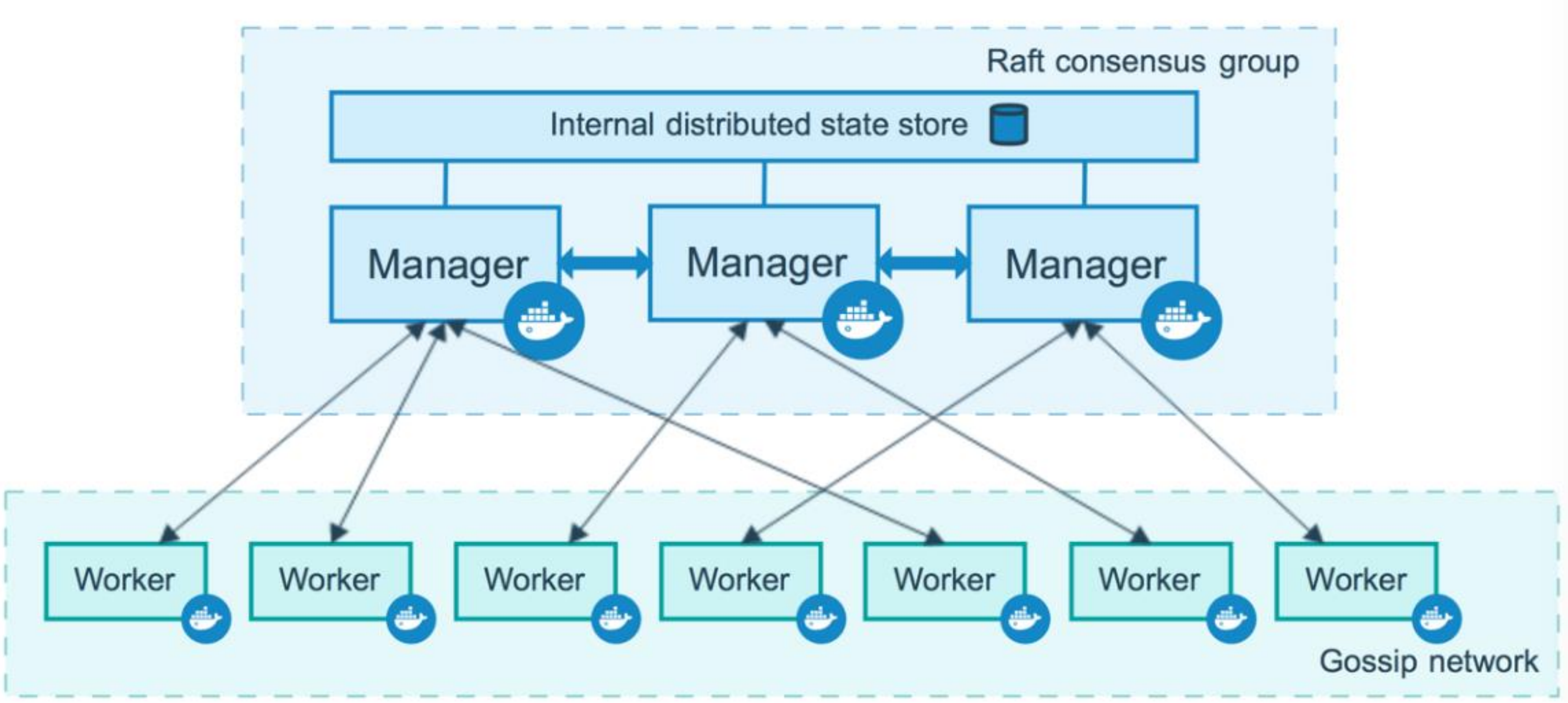


Orchestration Solutions

- Docker Swarm by Docker
- Kubernetes
- Apache Mesos
- Redhat Openshift (OCP)
- AWS ECS/EKS
- Mirantis and so on

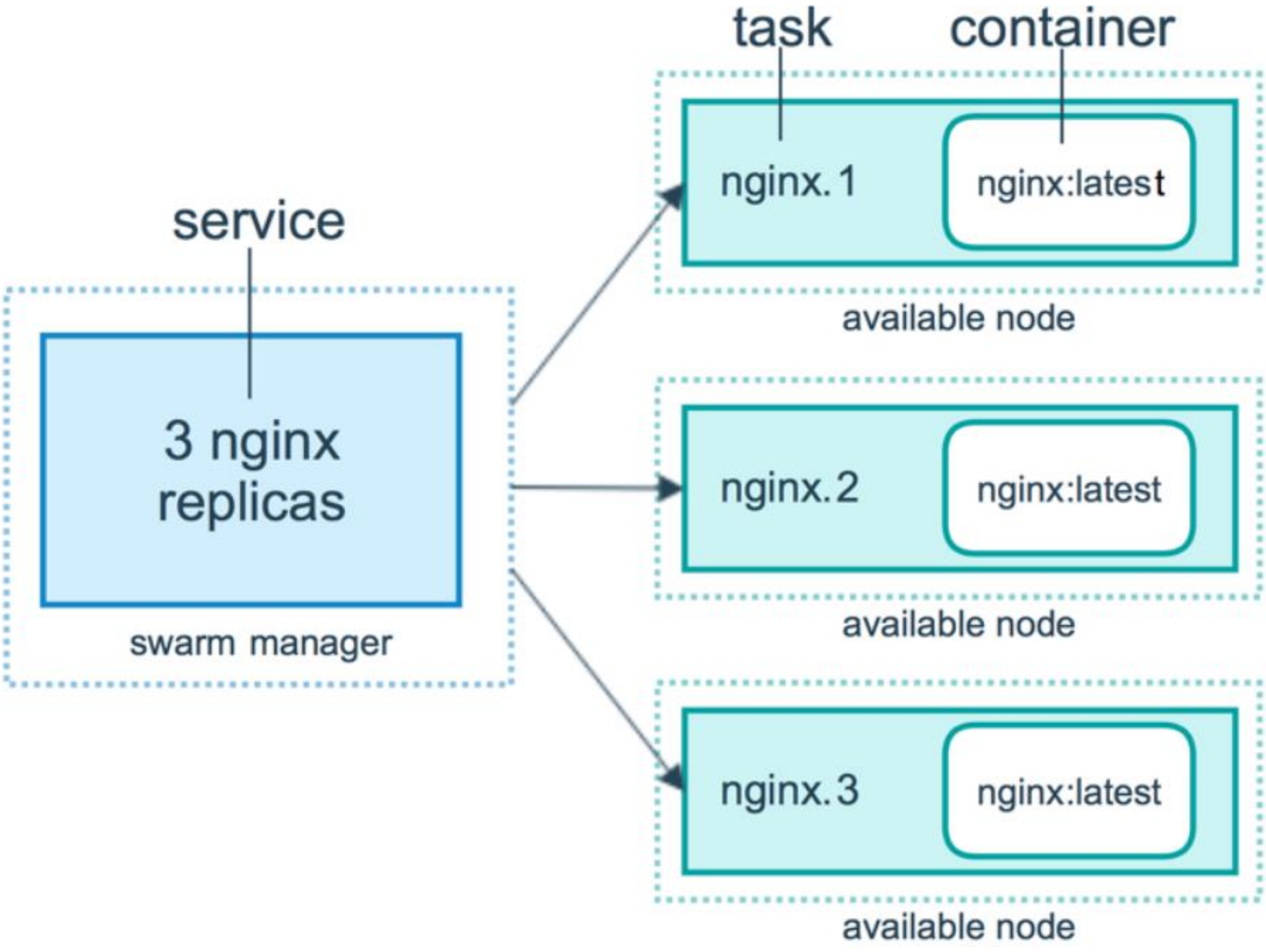
Docker Swarm

- Container orchestration solution supported by Docker
- **Cluster** – Combination of nodes includes Master and worker nodes
- **Master Node** – Manages the Docker swarm
- **Worker Node** – Runs the tasks or containers
- **Task** – Container running on worker node



Docker Service

- Definition of task to execute on Master or worker node
- E.g. docker service create --name webserver --replicas 2 nginx



Scaling

- Scale out or scale in number of containers running under service
- Two ways to scale
  - By using **scale** option
    - E.g docker service scale webapp=1
    - You can scale multiple services in one go
  - By using **update** option
    - E.g docker service update --replicas 2 service02
    - You can scale multiple services in one go

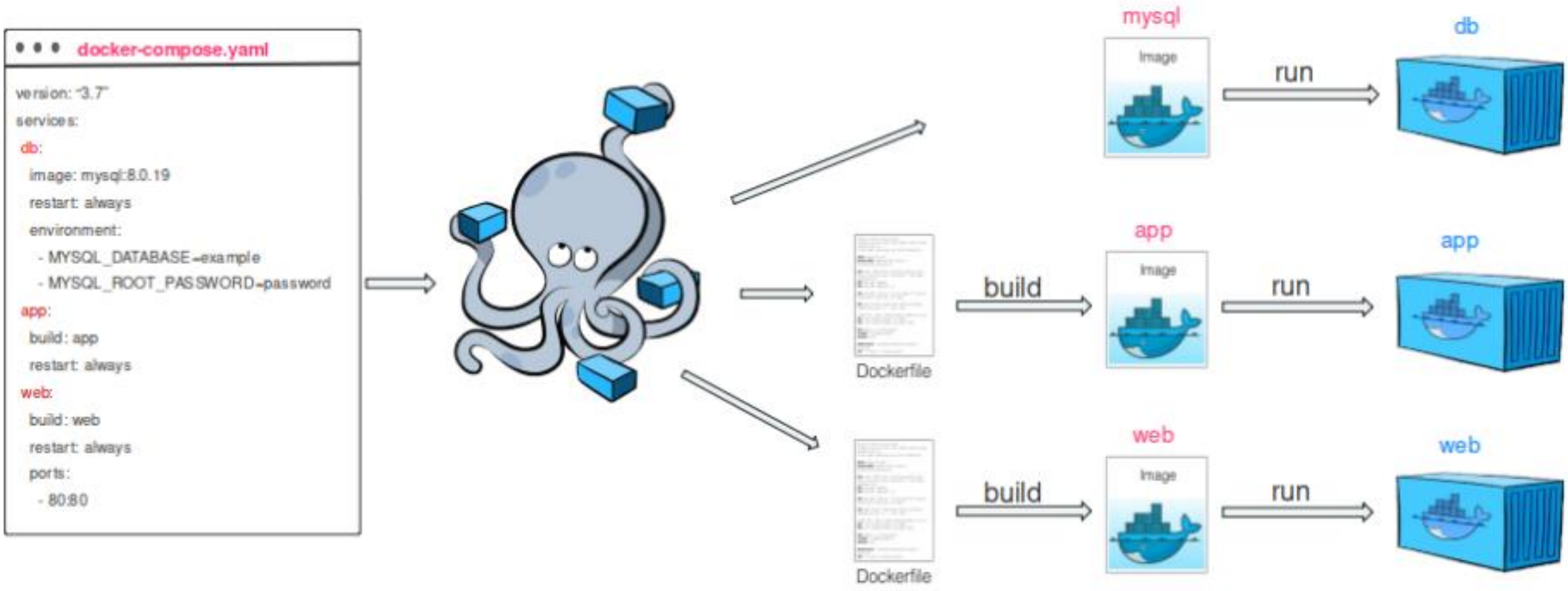
Publishing Port in swarm

- We need to publish the port to access the container application from outside
- Need to publish port during service creation time

Docker Compose

- Tool for deploying multi container application in non swarm environment
- Need to provide yaml configuration file as input
- Can start all application containers by using single command as ‘docker compose up’
- Can stop all application containers by using single command as ‘docker compose down’
- On Win/MAC- Packages are already present, additional package installation needed on Linux
- Not recommended to use in swarm

Docker Compose

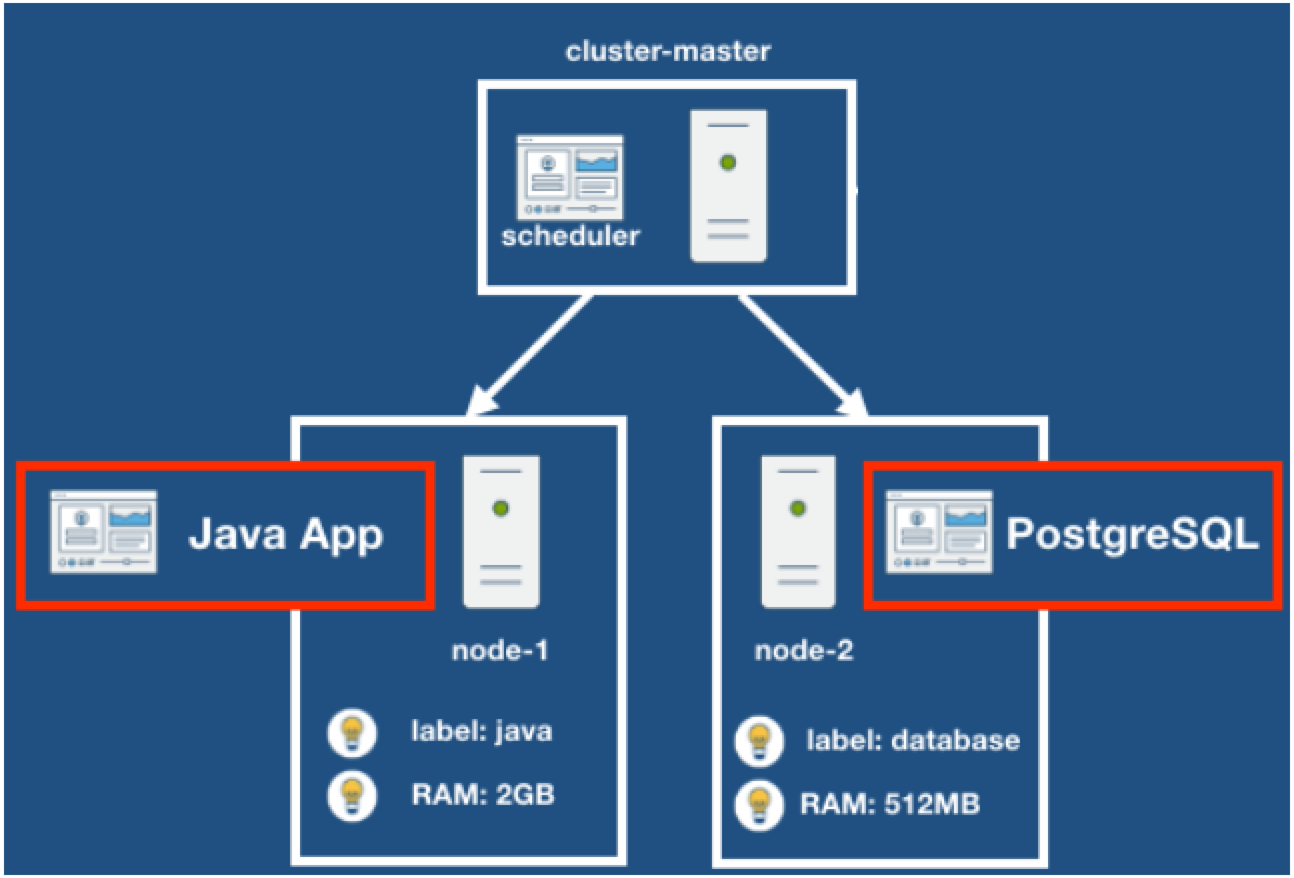


docker-compose.yaml

```
version: "3.7"
services:
 db:
  image: mysql:8.0.19
  restart: always
  environment:
   - MYSQL_DATABASE=example
   - MYSQL_ROOT_PASSWORD=password
 app:
  build: app
  restart: always
 web:
  build: web
  restart: always
  ports:
   - 80:80
```

Container/Task Placement

- Provides control over placement of services
- If you want to deploy specific container on specific node only
- Labels are used for placement


cluster-master
scheduler
Java App
node-1
label: java
RAM: 2GB
node-2
PostgreSQL
label: database
RAM: 512MB