\*\*\*DOCKER COMPOSE\*\*\*

//запуск нескольких контейнеров в раз с помощью YAM-файла. Запуск подготовленной инфраструктуры в несколько кликов//

Вместо нескольких команд:

docker run --name some-mysql -e MYSQL\_ROOT\_PASSWORD=test123 -d mysql:8.0.31 -запуск бд

docker run --name myphp -d --link some-mysql:db -p 8081:80 phpmyadmin/phpmyadmin -запуск phpMyAdmin в связке с БД

YAML – это язык для сериализации данных. Позволяет хранить сложноорганизованные данные в компактном и читаемом формате. Язык для хранения информации в формате понятном человеку.

YAML-файл – файл, содержащий инструкцию для DOCKER COMPOSE.

vi project.yaml -создаем файл yaml для docker compose

version: ‘3.9’ -версия docker compose

services: -воженные ключи

db: -вложенный ключ

image: marisdb:10.10.2

restart: always -всегда перезапускает контейнер если он был остановлен

environment: -указываем переменные которые необходимы для работы контейнера

MYSQL\_OOT\_PASSWORD: test123

adminer: -вложенный ключ

image: adminer:4.8.1

restart: always

ports:

* 6080:8080

docker-compose up -запуск КОНСОЛЬ БЛОКИРУЕТСЯ!

docker-compose up -d -запуск в режиме демона КОНСОЛЬ НЕ БЛОКИРУЕТСЯ!

docker container inspect root\_adminer\_1 -показ всей информации о нашем контейнере adminer

docker container inspect root\_db \_1 - показ всей информации о нашем контейнере db

docker-compose build -сборка сервисов описанный в конфигурационном файле yaml

docker-compose up -d -деплой нашего проекта и последующий его запуск в фоновом режиме

docker-compose start -запуск ранее остановленных сервисов

docker-compose down -останавливает ранее запущенные сервисы и удаляет все сущности которые были необходимы для запуска проекта

docker-compose stop -останавливает все сервисы которые были необходимы для запуска проекта без их удаления

docker-compose logs -f <service\_name> -собирает логи сервиса который мы запустили

docker-compose exec -позволяет выполнить команду в сервисе не заходя в него

docker-compose images -позволяет вывести все образы которые будут доступны в конфигурационном файле

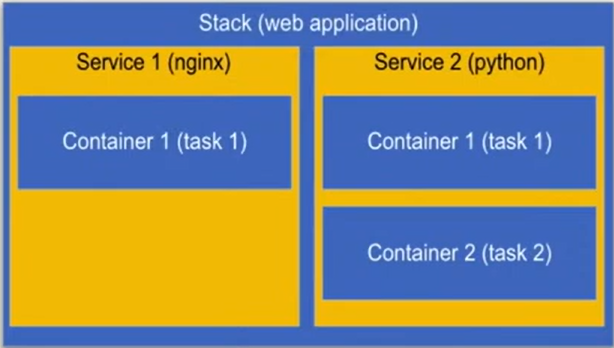
\*\*\*DOCKER SWARM\*\*\*

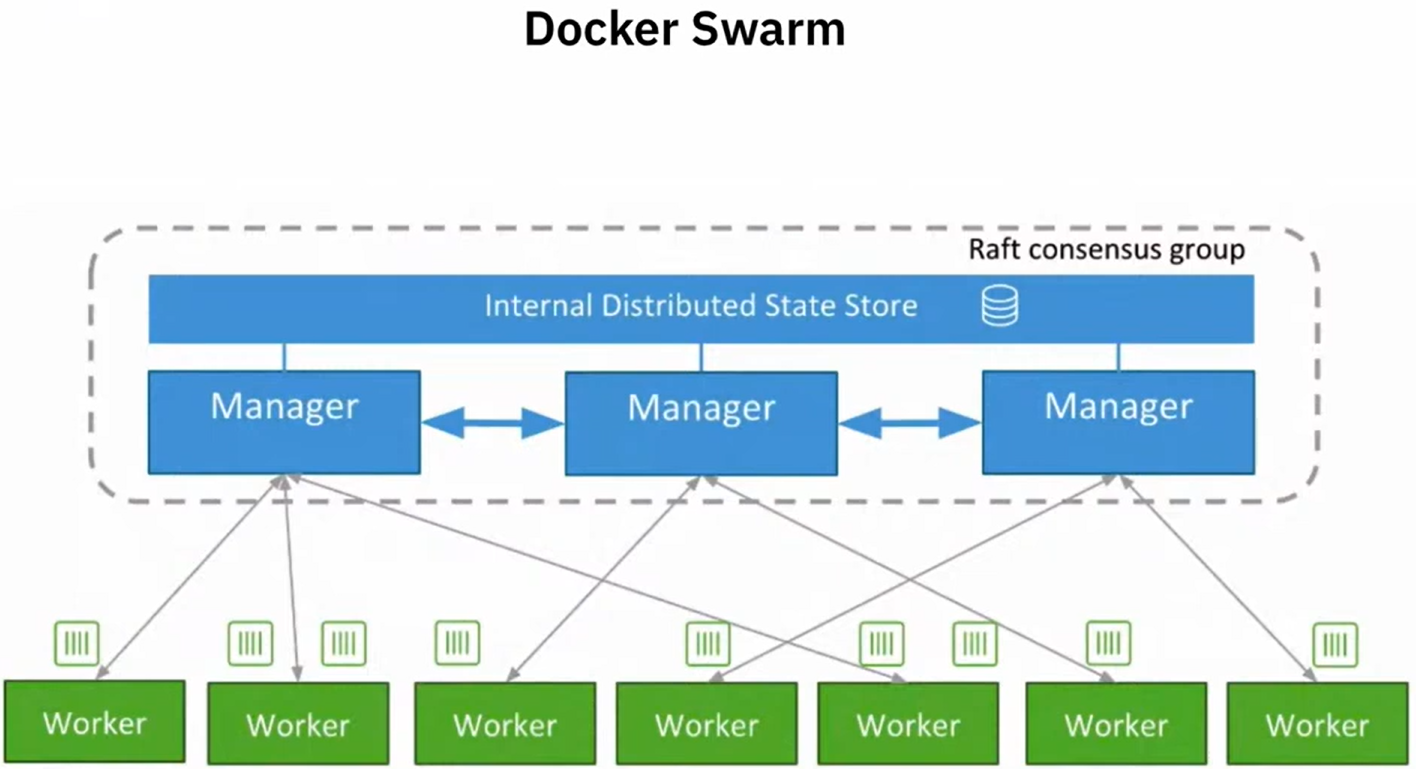
Node -наш сервер с установленным на него Docker. Сервер может быть как физическим так и виртуальной машиной

Stack -это набор сервисов, которые могут быть связаны между собой логически

Сервис -это составляющая стэка

Task (задача) -непосредственно созданный контейнер







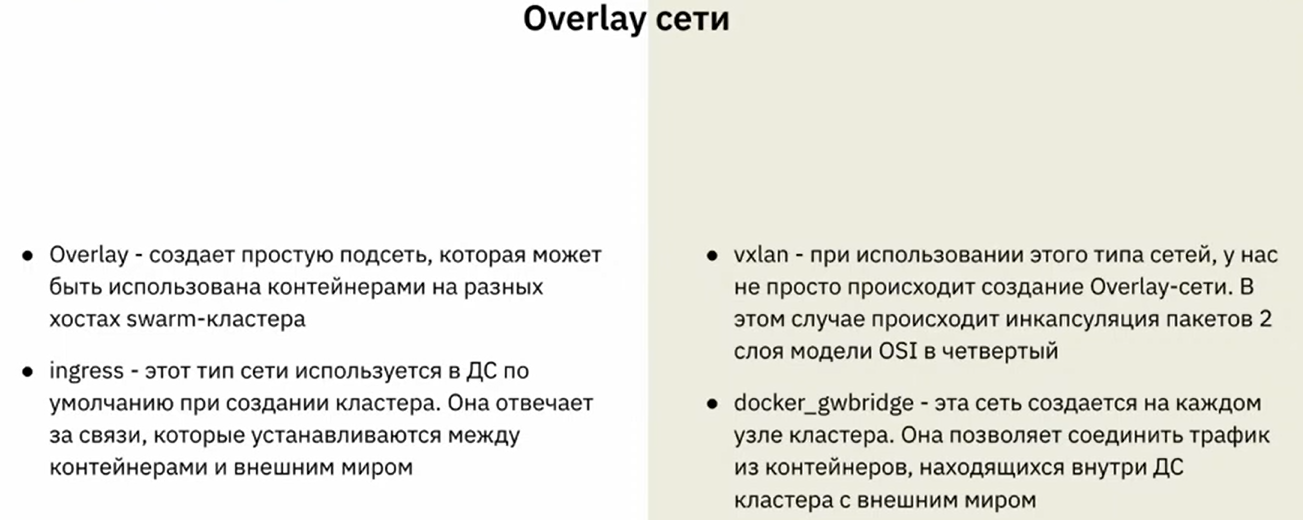
node ls -список нод

docker node rm 4fd5s64dsa5f4 -удаление ноды

docker node promote docker-2 -установка ноды в качестве лидера

docker node –help -инфо

docker node inspect docker-2 -вывод информации о ноде docker-2



docker network ls -вывод сетей в кластере

docker network create --driver overlay --subnet 4.5.6.0/24 test0network --attachable -где test0network имя подсети, --attachable означает что можем присоединять различного рода контейнер

docker run -d --ip 4.5.6.7 --net test-network --name container-1 busybox sleep 3600 -запуск контейнера с установленной подсетью

docker run -d --ip 4.5.6.8 --net test-network --name container-1 busybox sleep 3600 -запуск еще одного контейнера с установленной подсетью

docker exec -it container-2 sh -заходим в контейнер 2, после чего все пингуется

ip a -просмотр сетевой информации (контейнера)

ll /var/run/docker/netns/ -просмотр доступных сетей у докера

nsenter --net=/var/run/ocker/netns/1-vdo80djg5lk -просмотр конкретной сети

СОВЕТ 1 – ИЗОЛИРУЙТЕ СЕТИ! НЕ ИСПОЛЬЛЗУЙТЕ СЕТЬ ХОСТОВОЙ МАШИНЫ! Используйте виртуальные сети для различных проектов