Терминология

Термин	**Описание**
Контейнеризация	способ виртуализации ОС, при котором код приложения, среда запуска, библиотеки и зависимости упаковываются в единую «капсулу» — контейнер.
Контейнер	сущность на основе образа, приложение, которое разворачивается с помощью Docker. Состоит из слоёв, которые можно только читать. При создании контейнера слой, в который можно вносить изменения, добавляется поверх всех остальных слоёв.
Docker	это платформа, которая позволяет упаковать в контейнер приложение со всем окружением и зависимостями, а затем доставить и запустить его в целевой системе, не задумываясь о том, в какой среде она [система] будет функционировать
Docker daemon	Фоновая служба на хосте, которая отвечает за создание, запуск и уничтожение контейнеров.
Dockerfile	специальный файл с инструкциями для Docker. Представляет из себя текстовый документ, содержащий все команды для сборки образа. создаётся по принципу «одна строка — одна команда».
Docker image	это неизменяемый образ, из которого разворачивается контейнер

Образ	Базовый образ — главный элемент контейнеризации в Docker. В нем содержатся процессы и зависимости, необходимые для нормальной работы приложения. Чаще всего скачивают готовые. Можно создать через консоль *docker run image_name* либо внутри Dockerfile прописать команды + в консоли *прописать docker image build*
Пакет	архив, который содержит все необходимые приложению бинарные и конфигурационные файлы, информацию о том, как их следует разместить в файловой системе
Docker Desktop	удобный клиент, который отображает все сущности Docker
Docker Hub	популярный публичный репозиторий, используемый по умолчанию в Docker. Обеспечивает интеграцию с GitHub и BitBucket.
Docker Compose (в работе не используется)	инструмент для управления несколькими контейнерами. Он позволяет создавать контейнеры и задавать их конфигурацию

Интересно:

1. Использование контейнеров позволяет перейти с монолита на микросервисную архитектуру. За счет этого ускоряется разработка новой функциональности, поскольку нет опасений, что изменения в одной компоненте затронут всю остальную систему.

2. Важнейшая особенность контейнеров — их сравнительно короткий жизненный цикл. Любой контейнер можно остановить, перезапустить или уничтожить, если это необходимо. Данные, которые содержатся в контейнере, при этом тоже пропадут. Так выработалось правило проектирования приложений: не хранить важные данные в контейнере. Такой подход называют Stateless.

Описание команд

В инструкции

Инструкция/символ/команда	Описание	Пример (при необходимости)
FROM	Инструкция FROM задает базовый образ для последующих инструкций FROM должен быть первой инструкцией в Dockerfile (не считая комментариев или инструкции ARG)	
\ (в конце строки)	Применяется, чтобы собрать несколько строк в одну инструкцию, иначе – перенос строки	RUN apt update && apt install -y \ postgresql-contrib \ по правилам: одна строка – одна команда, но чтобы код был читабельным, можно разделить. А чтобы при билдинге всё засчиталось за одну строку кода, то можно поставить \ и разделять код на абзацы
RUN	выполняет любые команды и делает коммит результата + позволяет создать слой во время сборки образа	
&&	конкатенация строк и слоёв означает что вместо трех отдельных слоев будет создан лишь один, что заметно уменьшает и размер	

	образа, и скорость его разворачивания	
install (команда apt)	установить пакет	
ENV	задает переменные окружения с именем <key> и значением <value>. Это значение будет находиться в окружении всех команд потомков Dockerfile и могут быть использованы как обычные переменные окружения	
mkdir	создать директорию (каталог, папку)	
chown	Команда chown позволяет изменить пользователя и / или группу, владеющую данным файлом, каталогом или символической ссылкой.	chown [OPTIONS] USER[:GROUP] FILE(s) USER — это имя пользователя или идентификатор пользователя (UID) нового владельца. GROUP — это имя новой группы или идентификатор группы (GID). FILE(s) — это имя одного или нескольких файлов, каталогов или ссылок.
VOLUME	указывает место, которое контейнер будет использовать для постоянной работы и хранения файлов	
EXPOSE	Указывает доступный порт к БД вне контейнера	

CMD	Инструкция CMD указывает, какая команда будет выполнена когда создается новый контейнер на основании уже созданного образа. Инструкция CMD может быть использована только один раз в Dockerfile. Иначе учтётся последняя инструкция	
Дополнительно		
LABEL	позволяет добавлять в образ метаданные (метки), т.е. полезную информацию об образе Docker Например, может включать в себя контактные сведения создателя образа	LABEL maintainer="jeffmshale@gmail.com"
COPY	сообщает Docker о том, что нужно взять файлы и папки из локального контекста сборки и добавить их в текущую рабочую директорию образа (при отсутствии сначала создаст)	COPY . ./app

ADD	Что и COPY + можно добавлять в контейнер файлы, загруженные из удалённых источников, а также распаковывать локальные .tar-файлы	ADD https://raw.githubusercontent.com/discdiver/pachy-vid/master/sample_vids/vid1.mp4 \ /my_app_directory
ARG	позволяет задать переменную, значение которой можно передать из командной строки в образ во время его сборки	ARG my_var=my_default_value.

Интересно:

1. Вместо CMD можно использовать ENTRYPOINT. Основное отличие в доступе к переопределению.

ENTRYPOINT - пользователь образа НЕ может переопределять поведение приложения в контейнере.

CMD - записанную команду по умолчанию пользователь с лёгкостью может переопределить на этапе запуска контейнера.

какую инструкцию, CMD или ENTRYPOINT, стоит выбрать в качестве инструмента для выполнения команд при запуске контейнера:

- Если при каждом запуске контейнера нужно выполнять одну и ту же команду — используйте ENTRYPOINT.
- Если контейнер будет использоваться в роли приложения — используйте ENTRYPOINT.
- Если вы знаете, что при запуске контейнера вам понадобится передавать ему аргументы, которые могут перезаписывать аргументы, указанные в Dockerfile, используйте CMD.

2. Если указанный образ отсутствует в системе, где выполняется процесс сборки Docker (инструкция FROM), подсистема Docker попытается скачать его из общедоступного или частного реестра образов.

3. В отличие от ENV-переменных, ARG-переменные недоступны во время выполнения контейнера. Однако ARG-переменные можно использовать для задания значений по умолчанию для ENV-переменных из командной строки в процессе сборки образа. А ENV-переменные уже будут доступны в контейнере во время его выполнения

4. EXPOSE НЕ открывает порт, только указывает. Для открытия после билдинга необходимо запустить

docker RUN -p 5432:5432

В консоли

Команда	**Описание**	**Пример (при необходимости)**
docker ps -a	Вывести все имеющиеся контейнеры (независимо от состояния)	
docker ps	Проверить на наличие активных контейнеров	**docker ps название_контейнера**
docker stop	Остановить действующий контейнер	
docker build	Создание образа на основе инструкций внутри Dockerfile	docker build -t названиеобраза . -t - тэг (имя, которое последует далее) docker build -t korotkovatest-postgres .
docker image	Отображение всех действующих образов	
docker run		docker run --name название_контейнера -e POSTGRES_PASSWORD=пароль -d -p

		номер:порта название-образа - name позволяет указать пользовательский идентификатор для контейнера, иначе будет сгенерирован самостоятельно -e передаёт переменные окружения -d указывает на запуск контейнера в фоновом режиме. Это значит, что терминал, можно сказать,не захвачен контейнером. И можно выполнять другие команды либо свернуть терминал, в то время, как запущенный контейнер находится всё также активным -p используется для активации порта
docker stop	Остановка контейнера или образа	docker stop название /id контейнера
docker rm	Удаление контейнера	docker rm название/id контейнера
docker rmi	Удаление образа	docker rm название/id образа

Установка IDE Visual Studio Code

1. Установить приложение Docker с оф.сайта
2. Открыть IDE, в моём случае Visual SC

3. Внутри IDE загрузить 2 расширения: Docker + PostgreSQL
4. На панели слева появятся 2 иконки расширений
5. Создать папку, внутри которой docker файл
6. Открыть папку в IDE
7. В Explorer отобразится открытая папка внутри неё необходимо создать файл, нажав на иконку документа с +. Файл должен быть под названием dockerfile
8. Файл сохранён, иконка у файла с эмблемой докера

Важно: одна папка = один докер файл

Инструкции в dockerfile

Инструкции ниже необходимо ввести в область редактора.

Инструкции

```
# официальный образ PostgreSQL

FROM postgres:17


# пакеты

RUN apt update && apt install -y \

    postgresql-contrib \

    vim\

    && rm -rf /var/lib/apt/lists/*


# устанавить переменные окружения

ENV POSTGRES_DB=Korotkova

ENV POSTGRES_USER=KorotkovaUser

ENV POSTGRES_PASSWORD=12345

ENV PGDATA=/var/lib/postgresql/data/pgdata


# создать директорию для хранения данных

RUN mkdir -p $PGDATA && chown -R postgres:postgres $PGDATA


# указать порт

EXPOSE 5432


# установить место, которое контейнер будет использовать для постоянной работы и хранения файлов

VOLUME [«$PGDATA»]
```

запуск PostgreSQL

CMD ["postgres"]

Описание

1. задать базовый образ для последующих инструкций

FROM postgres:17

postgres – название образа из репы https://www.postgresql.org/docs/

#:17 – тег, уточняющий то, какой именно базовый образ нужен. Если тег в инструкцию не включён, тогда Docker исходит из предположения о том, что требуется самый свежий образ из репозитория.

#FROM - задает родительский (главный) образ из репозитория

https://hub.docker.com/_/postgres - ссылка на официальный Docker-репозиторий с образами postgres.

тег 17 указывает Docker'у, какую именно версию образа нужно использовать. Если тег не указан, по умолчанию берётся последняя версия образа.

Образ включает в себя Linux и postgres.

Также есть Alpine-образы.Они маленькие, быстрые и безопасные.

Для Alpine Docker-образа используется apk. apk для типичной Linux-сборки — apt-get. Например, пакеты для базового Ubuntu-образа могут быть установлены и обновлены так: RUN apt-get update && apt-get install my_package.

2. установка последних версий пакетов, которые перечислены после «apt install -y»

RUN apt update && apt install -y \

apt (apt-get) - набор утилит для установки, удаления, обновления, поиска пакетов из базового образа (указан в FROM)

update - APT проходит по списку репозиториев и из каждого репозитория в списке получает информацию о последних пакетах, находящихся в репозитории

&& - после обновление происходит их установка

install - установить пакет

-y --yes - автоматически отвечать "да" на все возникающие вопросы (с сайта)

Но у меня отсутствие -y блокировало создание контейнера без явного указания пароля в команде RUN в консоли

postgresql-contrib \

дополнительный набор расширений и утилит для PostgreSQL

&& **rm** -rf /var/lib/apt/lists/*

удаление ненужных файлов и данных, например, информация о доступных пакетах в репозиториях, временные каталоги и файлы, созданные APT во время обновления списка пакетов; очистка кэша

rm - удаление

rf /var/lib/apt/lists/* - путь к каталогу, который создаётся на ноутбуке

3. установить переменные

#название своей бд

POSTGRES_DB=Korotkova

#имя юзера бд – понадобится для подключения к БД

ENV POSTGRES_USER=KorotkovaUser

пароль для подключения к бд – понадобится для подключения к БД

ENV POSTGRES_PASSWORD=12345

#каталог данных кластера (универсальный), где будут храниться данные PostgreSQL.

ENV PGDATA=/var/lib/postgresql/data/pgdata

#Файлы конфигурации и файлы данных, используемые кластером базы данных, традиционно хранятся вместе в каталоге данных, который обычно называют PGDATA (по имени переменной среды, которую можно использовать для его определения). Обычно PGDATA находится в /var/lib/pgpro/std-10/data.

4. создание директории для хранения данных. Путь к директории хранится в PGDATA

RUN mkdir -p $PGDATA && chown -R postgres:postgres $PGDATA

mkdir -p $PGDATA – загружается каталог из переменной PGDATA

chown -R postgres:postgres $PGDATA – определение владельца каталога данных (и внутренних подкаталогов)

Создание директории при создании контейнера Docker необходимо для **обеспечения правильной работы приложения**.

Директория контекста сборки — это каталог на хост-компьютере, откуда Docker получит файлы для сборки образа. Все файлы из этого каталога будут видны при сборке Docker.

5. указать на порт к бд внутри контейнера из вне. EXPOSE НЕ открывает порт

EXPOSE 5432

6. установить точку монтирования для данных (место, которое контейнер будет использовать для постоянной работы и хранения файлов)

VOLUME [«$PGDATA»]

7. запуск PostgreSQL

CMD ["postgres"]

Инструкция CMD указывает, какая команда будет выполнена когда создается новый контейнер на основании уже созданного образа

Build образа

Открыть Docker Desktop. Внутри либо через консоль проверить на наличие работающих (активных) контейнеров. При необходимости остановить их работу. Поскольку создание ведётся не через docker compose, то можно работать только с 1 активным контейнером

1. Ввести команду docker build -t название- postgres .

Например,docker build -t korotkovatest-postgres .

2. Проверим, что образ создался, появится в списке

docker image

Ради интереса можно зайти в docker desk -> images. Там будет созданный образ

3. создать контейнер на основе образа

docker run --name название_контейнера -e POSTGRES_PASSWORD=пароль -d -p номер:порта название-образа

например, docker run --name Korotkova_DB -e POSTGRES_PASSWORD=12345 -d -p 5432:5432 korotkovatest-postgres

ВАЖНО: необходимо здесь явным образом задать порт, иначе подключиться к БД далее не получится

Доп.

Чтобы удалить:

1. Сначала остановить

docker stop название /id контейнера

2. Удалить

docker rm название/id контейнера – удаление

3. Проверить список всех активных котейнеров

docker ps

docker rmi название/id образа – удаление

проверить docker images

Удаление образа без удаления контейнера, который его использует – невозможно

Подключиться к БД в VSC

1. Открыть плагин Postgres в VSC
2. Нажать на самый первый «+» верху, появится окно add connection
3. Ввести любое название в name
4. Ввести usename, password в соответствии с данными, который были установлены в переменных

ENV POSTGRES_USER=KorotkovaUser

ENV POSTGRES_PASSWORD=12345

5. Нажать connect

При успехе слева появится название созданной БД

6. Навести курсор правее от названия БД, появятся значки. Нажать на тот, что открывает терминал
7. Развернуть до table и нажать на +
8. Появится поле ввода справа, ввести код

Скрипты

```
-- Active: 1733769899649@@127.0.0.1@5432@Korotkova@public

-- members - дочерняя таблица для cards

create table members(

        tab_participant int,

        first_name VARCHAR,

        last_name varchar

);

select*from members


ALTER TABLE members

ADD PRIMARY KEY (tab_participant); --колонка становится первичным ключём


select*from members



INSERT INTO members VALUES (1111, 'Иванов', 'Александр');

insert into members values (2222, 'Сергеев', 'Сергей');

insert into members values (3333, 'Полякова', 'Анна');

select*from members




# cards - дочерняя таблица для table

create table cards(

        id_card int primary key,

        creator_id int,
```

```
	create_date DATE,
	title VARCHAR (100),
	descriptions VARCHAR (1000),
	done BOOLEAN,
	participants_id INT,
	FOREIGN KEY (participants_id) REFERENCES members (tab_participant)
);
select*from cards

insert into cards values (1, 2222, '2023-10-01', 'Планы', 'Описание', true, 2222);
select*from cards

insert into cards values (2, 2222, '2023-10-11', 'Название', 'Описание', true, NULL);
select*from cards

insert into cards values (3, 1111, '2023-10-03', 'Задача на день', 'Работать', false, 3333);
select*from cards

insert into cards values (4, 3333, '2023-11-23', 'Написать аналитику', 'Добавить ссылку', false,
1111);
select*from cards


#главная таблица
create table board(
	id_table int primary key,
	tab_author int,
	title VARCHAR (100)
```

```
);
select*from board

insert into board values (0001, 1111, 'Задачи команды');
select*from board

insert into board values (0002, 3333, 'Доска Поляковой');
select*from board




# дочерняя таблица для table
create table list(
	id_list int primary key,
	creator_id int,
	title VARCHAR (100)
);
select*from list

ALTER TABLE list
ADD COLUMN id_board INT,
ADD CONSTRAINT fk_id_table
FOREIGN KEY (id_board) REFERENCES board (id_table);
select*from list



insert into list values (11, 1111, 'to do', 0001);
```

```
select*from list

insert into list values (22, 1111, 'in progress', 0001);
select*from list

insert into list values (33, 1111, 'done', 0001);
select*from list


DELETE FROM list
WHERE id_list = 11;
select*from list

DELETE FROM list
WHERE id_list = 22;
select*from list

DELETE FROM list
WHERE id_list = 33;
select*from list


insert into list values (11, 1111, 'to do', 0001);
select*from list

insert into list values (22, 1111, 'in progress', 0001);
select*from list
```

```
insert into list values (33, 1111, 'done', 0001);
select*from list
```

```
-- 1) Отобразить все колонки, которые были созданы Ивановым Александром id 1111
#результат - 3 колонки с id 11, 22, 33
select*from list
where creator_id = 1111;

# 2) Отобразить id доски с названием Доска Поляковой
#результат - доска с id 2

select id_table
from board
where title = 'Доска Поляковой';

-- 3) Отобразить карточку, в которой creator_id = 2222, а participants_id = null
-- Результат: id card = 2
select*from cards
where (creator_id = 2222) AND (participants_id is NULL);
```

-- Работа с 2-мя таблицами cards + members

-- 4) Отобразить фамилию и имя создателя карточки с id 1

-- Результат: Сергеев Сергей

```
SELECT cards.id_card, members.first_name, members.last_name FROM cards JOIN members
ON participants_id = tab_participant

where id_card = 1;
```

-- 5) Показать названия всех колонок доски, где id не равно 0002 (=2)

-- Результат: to do, in progress, done

```
select board.id_table, list.title from list LEFT JOIN board ON id_board = id_table

where id_board != 2;
```

Источники

https://docs.docker.com/reference/dockerfile/#run

https://yandex.cloud/ru/blog/posts/2022/03/docker-containers?utm_referrer=https%3A%2F%2Fyandex.ru%2F

https://skillbox.ru/media/code/kak-rabotaet-docker-podrobnyy-gayd-ot-tekhlida/

https://devops.org.ru/dockerfile-summary

https://hmarketing.ru/blog/docker/fayl-dockerfile/

https://learn.microsoft.com/ru-ru/virtualization/windowscontainers/manage-docker/manage-windows-dockerfile

https://habr.com/ru/companies/ruvds/articles/439980/

https://losst.pro/kak-polzovatsya-apt

https://postgrespro.ru/docs/postgresql/17/storage-file-layout

https://timeweb.cloud/tutorials/linux/kak-dat-prava-polzovatelyu-linux