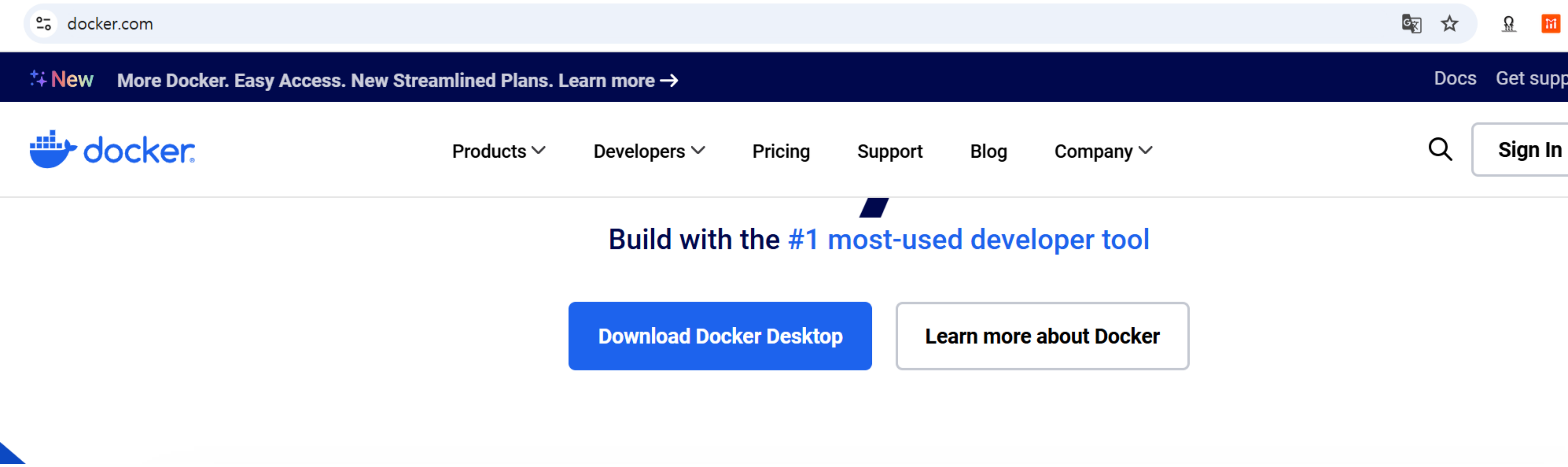
Docker

docker

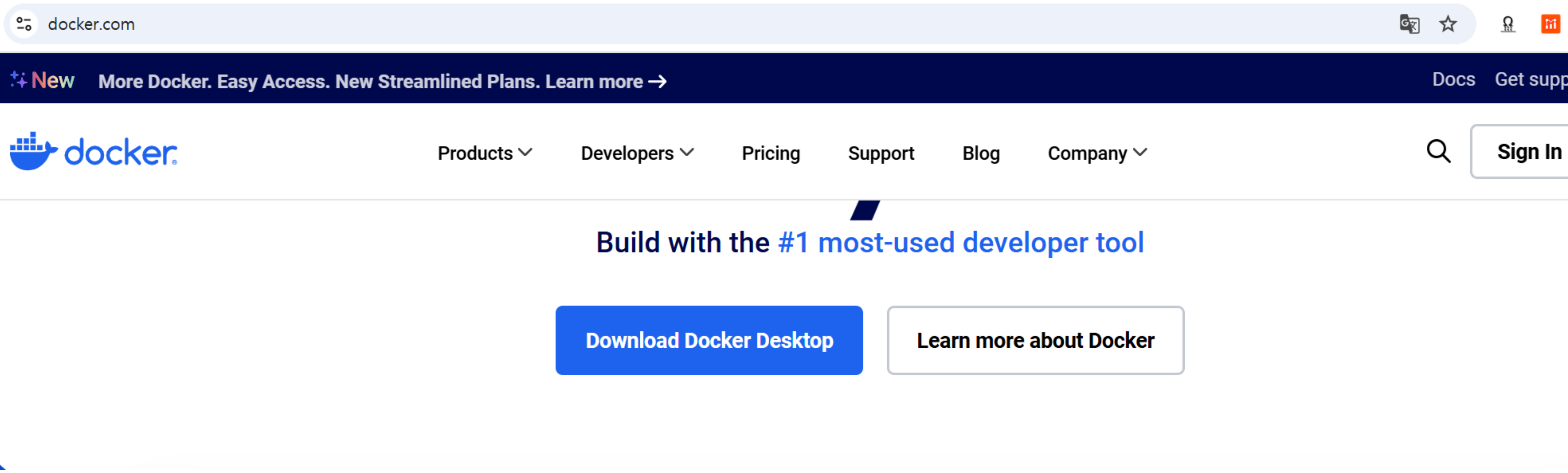
- docker



Build with the #1 most-used developer tool

docker

- docker

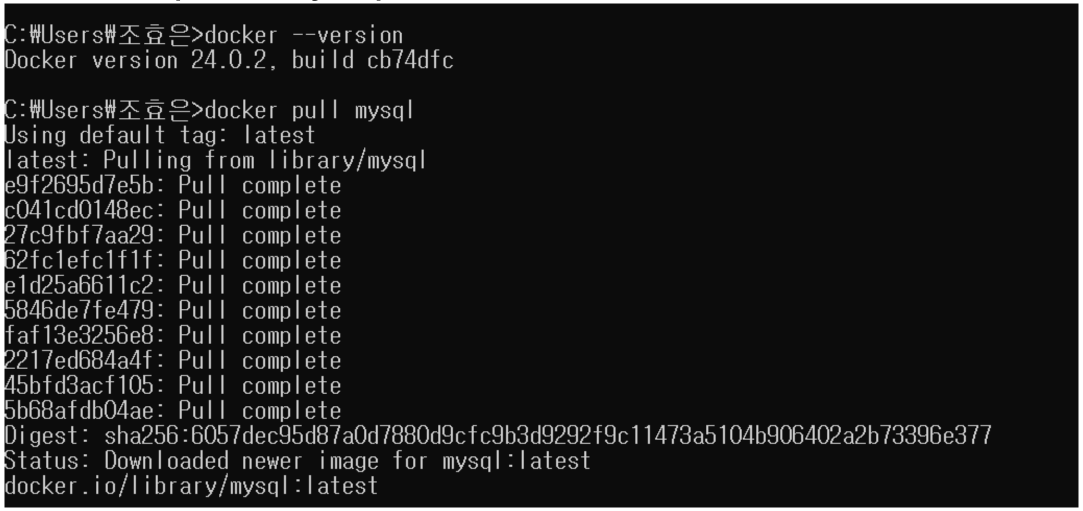


dokcer

install

docker –version

docker pull mysql
or
docker pull mysql:latest
or
docker pull mysql:8



```
C:₩Users₩조효은>docker --version
Docker version 24.0.2, build cb74dfc

C:₩Users₩조효은>docker pull mysql
Using default tag: latest
latest: Pulling from library/mysql
e9f2695d7e5b: Pull complete
c041cd0148ec: Pull complete
27c9fbf7aa29: Pull complete
62fc1efc1f1f: Pull complete
e1d25a6611c2: Pull complete
5846de7fe479: Pull complete
faf13e3256e8: Pull complete
2217ed684a4f: Pull complete
45bfd3acf105: Pull complete
5b68afdb04ae: Pull complete
Digest: sha256:6057dec95d87a0d7880d9cfc9b3d9292f9c11473a5104b906402a2b73396e377
Status: Downloaded newer image for mysql:latest
docker.io/library/mysql:latest
```

docker images

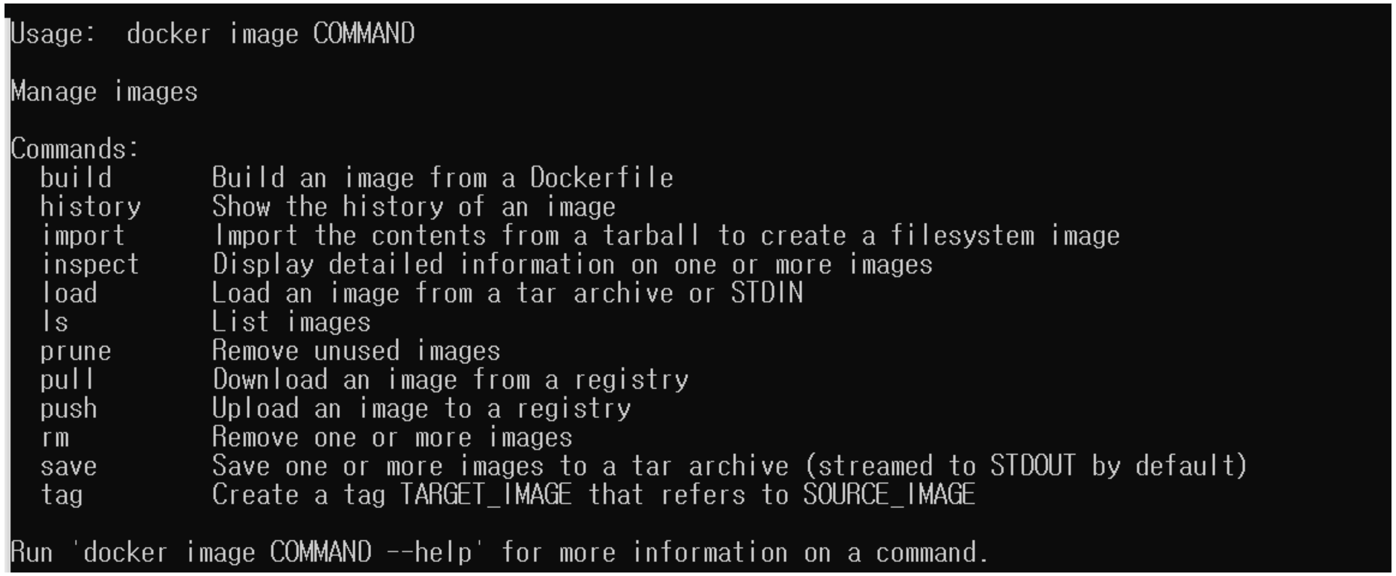
-- 최신버전 mysql 이미지 설치
docker pull mysql / $ docker pull mysql:8.0.22

-- 다운로드한 docker 이미지를 확인한다
docker images

docker

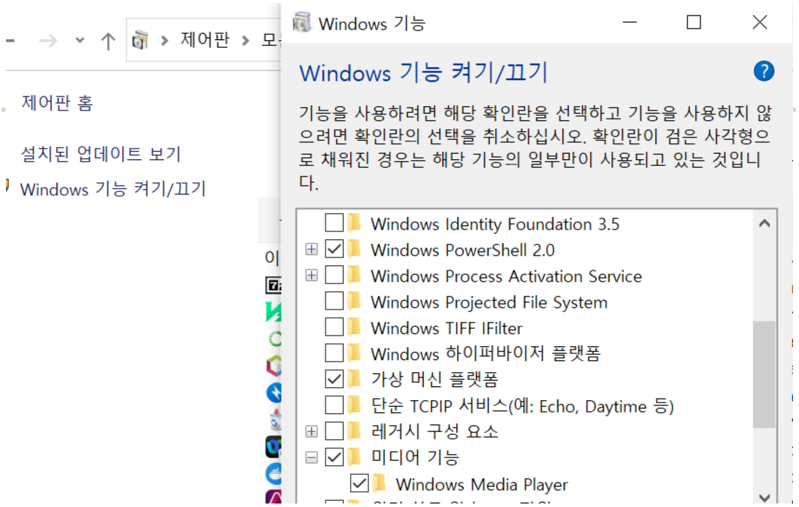
- docker

명령어



```
Usage:  docker image COMMAND

Manage images

Commands:
  build       Build an image from a Dockerfile
  history     Show the history of an image
  import      Import the contents from a tarball to create a filesystem image
  inspect     Display detailed information on one or more images
  load        Load an image from a tar archive or STDIN
  ls          List images
  prune       Remove unused images
  pull        Download an image from a registry
  push        Upload an image to a registry
  rm          Remove one or more images
  save        Save one or more images to a tar archive (streamed to STDOUT by default)
  tag         Create a tag TARGET_IMAGE that refers to SOURCE_IMAGE

Run 'docker image COMMAND --help' for more information on a command.
```

docker


Windows 기능

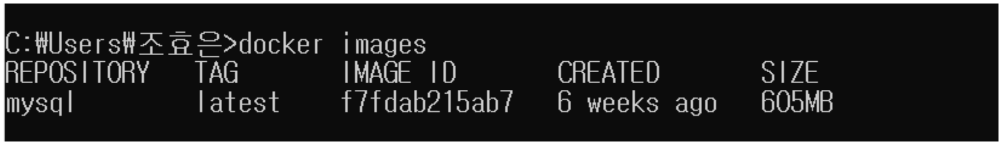
Windows 기능 켜기/끄기

기능을 사용하려면 해당 확인란을 선택하고 기능을 사용하지 않으려면 확인란의 선택을 취소하십시오. 확인란이 검은 사각형으로 채워진 경우는 해당 기능의 일부만이 사용되고 있는 것입니다.

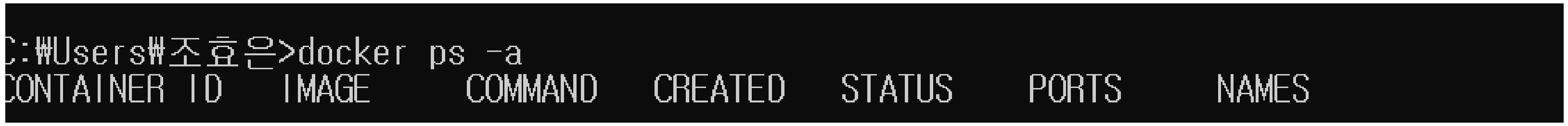
- [ ] Windows Identity Foundation 3.5
- [✓] Windows PowerShell 2.0
- [ ] Windows Process Activation Service
- [ ] Windows Projected File System
- [ ] Windows TIFF IFilter
- [ ] Windows 하이퍼바이저 플랫폼
- [✓] 가상 머신 플랫폼
- [ ] 단순 TCPIP 서비스(예: Echo, Daytime 등)
- [ ] 레거시 구성 요소
- [✓] 미디어 기능
  - [✓] Windows Media Player

- docker

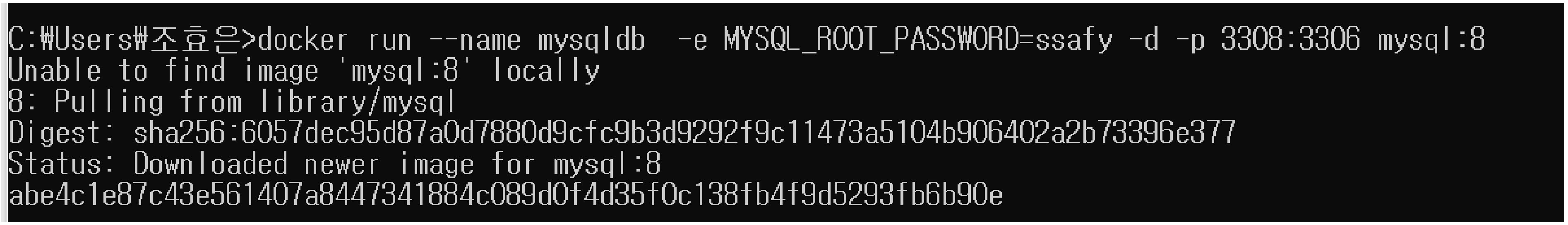
docker images


```
C:₩Users₩조효은>docker images
REPOSITORY    TAG       IMAGE ID       CREATED       SIZE
mysql         latest    f7fdab215ab7   6 weeks ago   605MB
```

docker ps -a


```
C:₩Users₩조효은>docker ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
```

docker run --name mysqldb -e MYSQL_ROOT_PASSWORD=kossa -d -p 3308:3306 mysql:latest
docker run --name mysqldb -e "MYSQL_ROOT_PASSWORD=kossa" -e "MYSQL_DATABASE=mydb" -e "MYSQL_USER=root" -e "MYSQL_PASSWORD=kossa" -d -p 3308:3306 mysql:latest

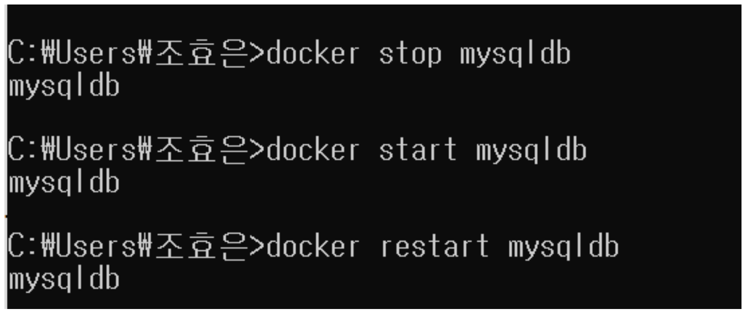

```
C:₩Users₩조효은>docker run --name mysqldb  -e MYSQL_ROOT_PASSWORD=ssafy -d -p 3308:3306 mysql:8
Unable to find image 'mysql:8' locally
8: Pulling from library/mysql
Digest: sha256:6057dec95d87a0d7880d9cfc9b3d9292f9c11473a5104b906402a2b73396e377
Status: Downloaded newer image for mysql:8
abe4c1e87c43e561407a8447341884c089d0f4d35f0c138fb4f9d5293fb6b90e
```

docker

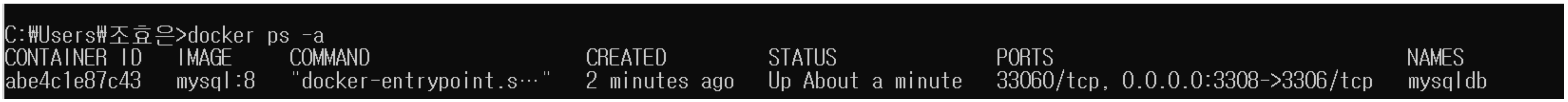
- docker

docker ps -a

```
version: "3" # 파일 규격 버전
services: # 이 항목 밑에 실행하려는 컨테이너 들을 정의
  db: # 서비스 명
    image: mysql:latest # 사용할 이미지
    container_name: mysqldb # 컨테이너 이름 설정
    ports:
      - "3308:3306" # 접근 포트 설정 (컨테이너 외부:컨테이너 내부)
    environment: # -e 옵션
      MYSQL_ROOT_PASSWORD: "ssafy"  # MYSQL 패스워드 설정 옵션
    command: # 명령어 실행
      - --character-set-server=utf8mb4
      - --collation-server=utf8mb4_unicode_ci
    volumes:
      - D:/java/docker:/var/lib/mysql # -v 옵션 (다렉토리 마운트 설정)
```



```
C:₩Users₩조효은>docker stop mysqldb
mysqldb

C:₩Users₩조효은>docker start mysqldb
mysqldb

C:₩Users₩조효은>docker restart mysqldb
mysqldb
```

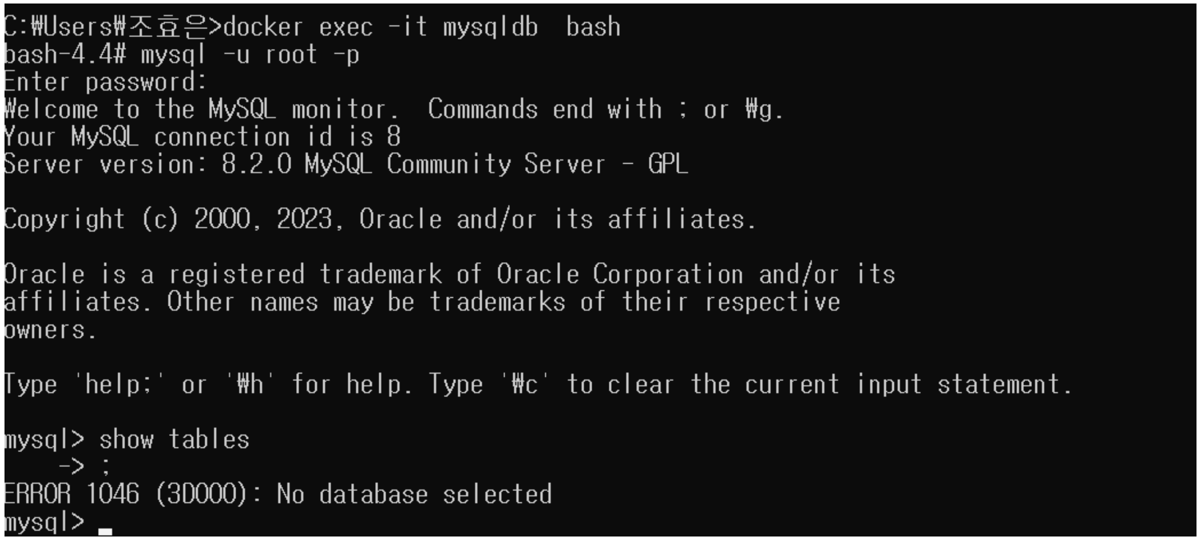


```
C:₩Users₩조효은>docker ps -a
CONTAINER ID   IMAGE     COMMAND                  CREATED         STATUS              PORTS                               NAMES
abe4c1e87c43   mysql:8   "docker-entrypoint.s…"   2 minutes ago   Up About a minute   33060/tcp, 0.0.0.0:3308->3306/tcp   mysqldb
```

docker

- docker

```
docker exec -it mysqldb  bash
mysql -u root -p
ssafy
```



```
C:₩Users₩조효은>docker exec -it mysqldb  bash
bash-4.4# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or ₩g.
Your MySQL connection id is 8
Server version: 8.2.0 MySQL Community Server - GPL

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '₩h' for help. Type '₩c' to clear the current input statement.

mysql> show tables
    -> ;
ERROR 1046 (3D000): No database selected
mysql> _
```

docker

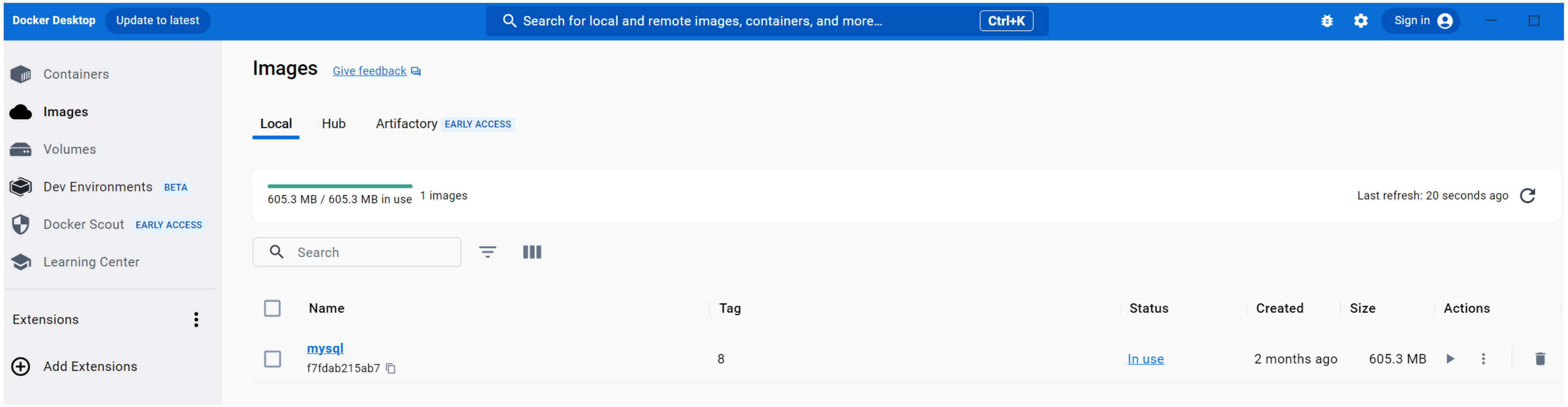
- docker

```
docker exec -it mysqldb  bash
mysql -u root -p
kossa
show tables;
```

docker

- docker

```
docker exec -it mysqldb  bash
mysql -u root -p
ssafy
```



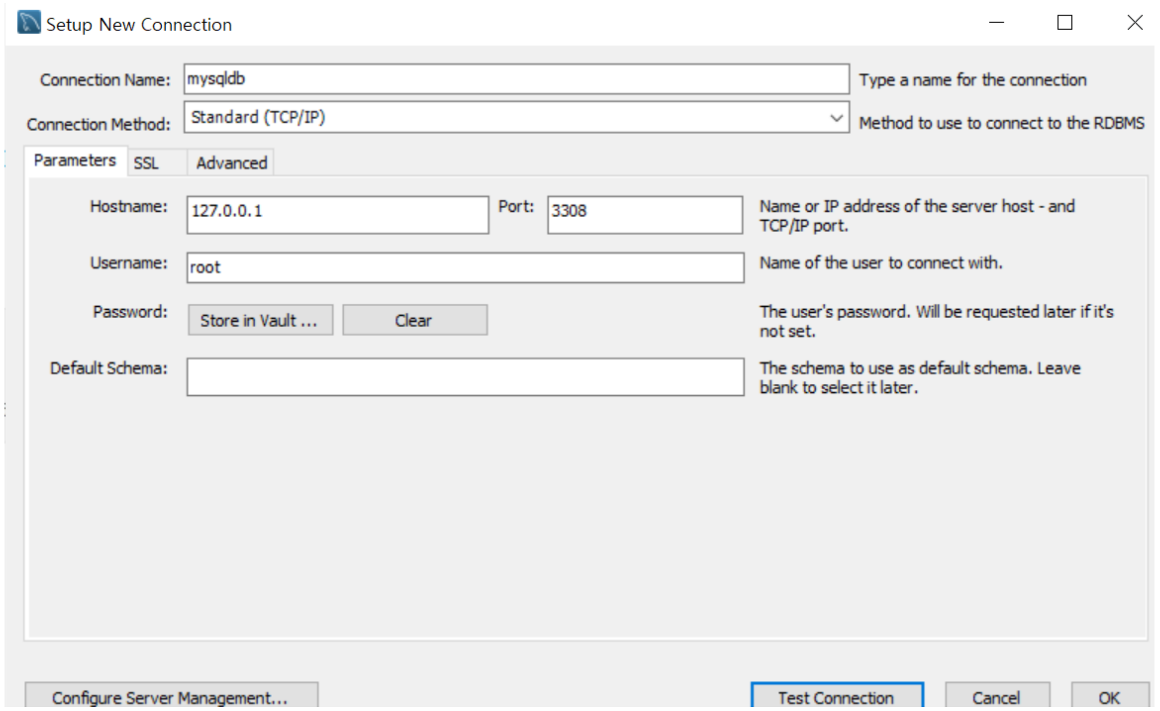
605.3 MB / 605.3 MB in use 1 images

Last refresh: 20 seconds ago

Name	Tag	Status	Created	Size	Actions
mysql f7fdab215ab7	8	In use	2 months ago	605.3 MB	

docker

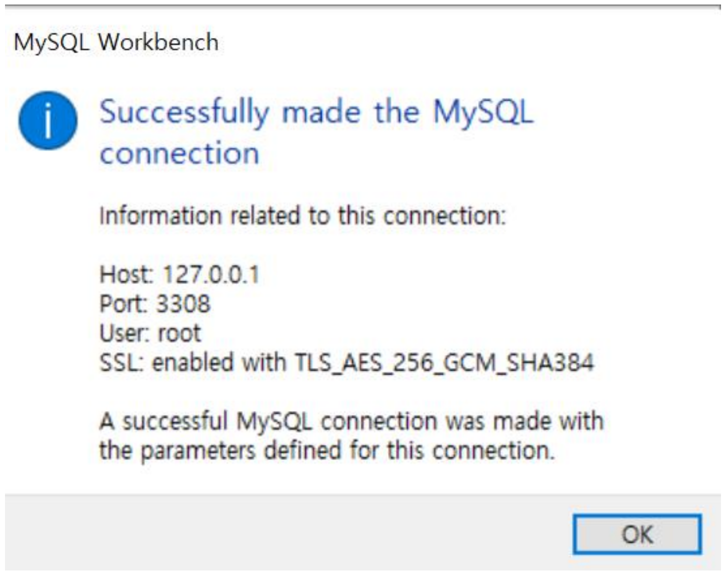
- docker



Setup New Connection

Connection Name:	mysqldb	Type a name for the connection
Connection Method:	Standard (TCP/IP)	Method to use to connect to the RDBMS

Hostname:	127.0.0.1 Port: 3308	Name or IP address of the server host - and TCP/IP port.
Username:	root	Name of the user to connect with.
Password:	Store in Vault ... Clear	The user's password. Will be requested later if it's not set.
Default Schema:		The schema to use as default schema. Leave blank to select it later.



MySQL Workbench

Successfully made the MySQL connection

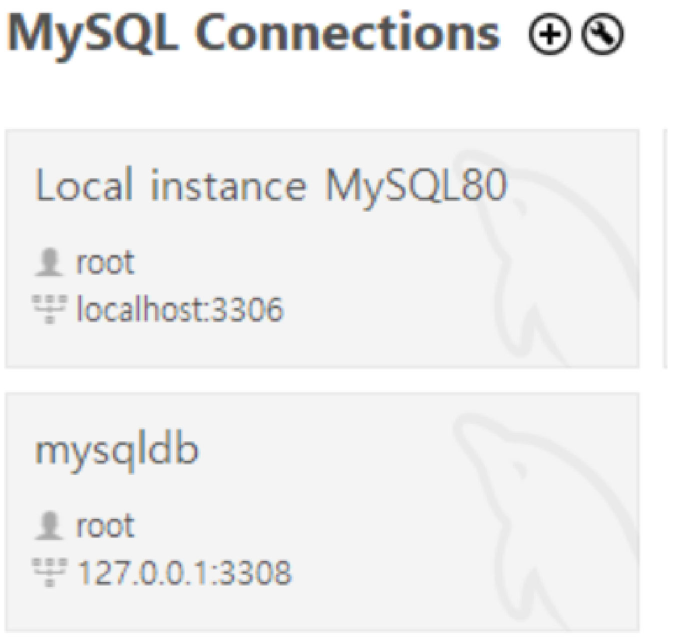
Information related to this connection:

Host: 127.0.0.1
Port: 3308
User: root
SSL: enabled with TLS_AES_256_GCM_SHA384

A successful MySQL connection was made with the parameters defined for this connection.

docker

- docker



MySQL Connections

Local instance MySQL80

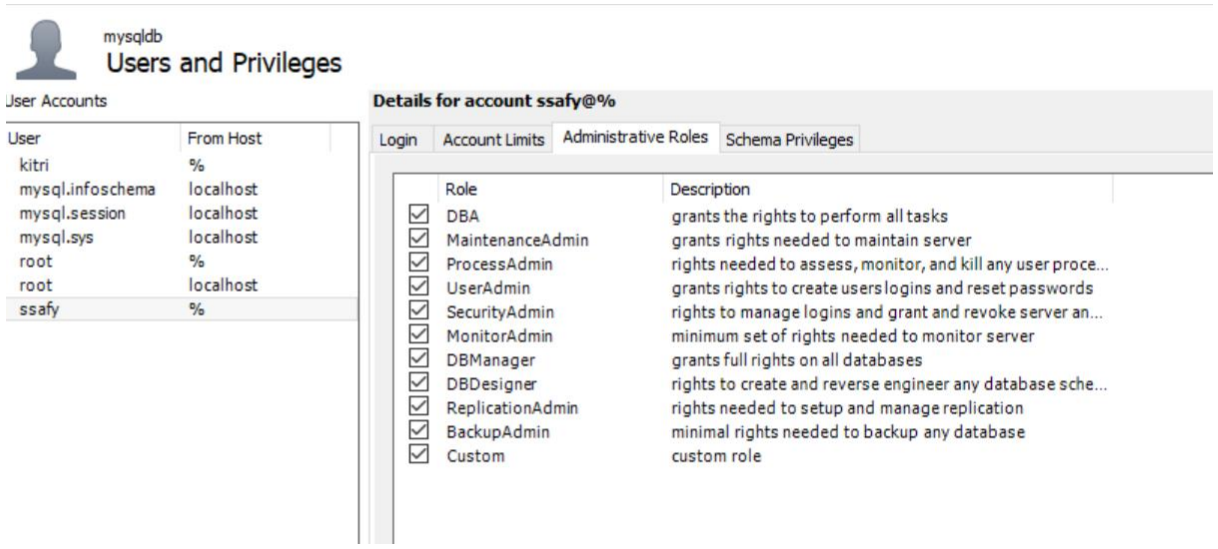
root

localhost:3306

mysqldb

root

127.0.0.1:3308



mysqldb

Users and Privileges

User Accounts

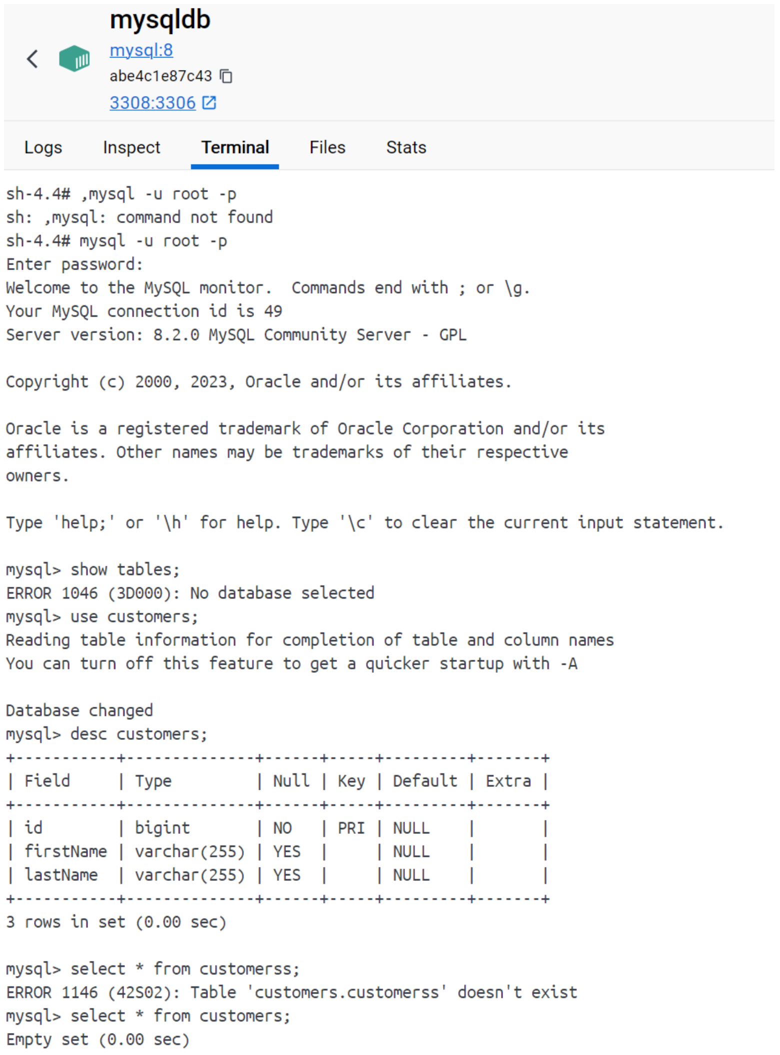
User	From Host
kitri	%
mysql.infoschema	localhost
mysql.session	localhost
mysql.sys	localhost
root	%
root	localhost
ssafy	%

Details for account ssafy@%

	Role	Description
✓	DBA	grants the rights to perform all tasks
✓	MaintenanceAdmin	grants rights needed to maintain server
✓	ProcessAdmin	rights needed to assess, monitor, and kill any user proce...
✓	UserAdmin	grants rights to create users logins and reset passwords
✓	SecurityAdmin	rights to manage logins and grant and revoke server an...
✓	MonitorAdmin	minimum set of rights needed to monitor server
✓	DBManager	grants full rights on all databases
✓	DBDesigner	rights to create and reverse engineer any database sche...
✓	ReplicationAdmin	rights needed to setup and manage replication
✓	BackupAdmin	minimal rights needed to backup any database
✓	Custom	custom role

docker

- docker

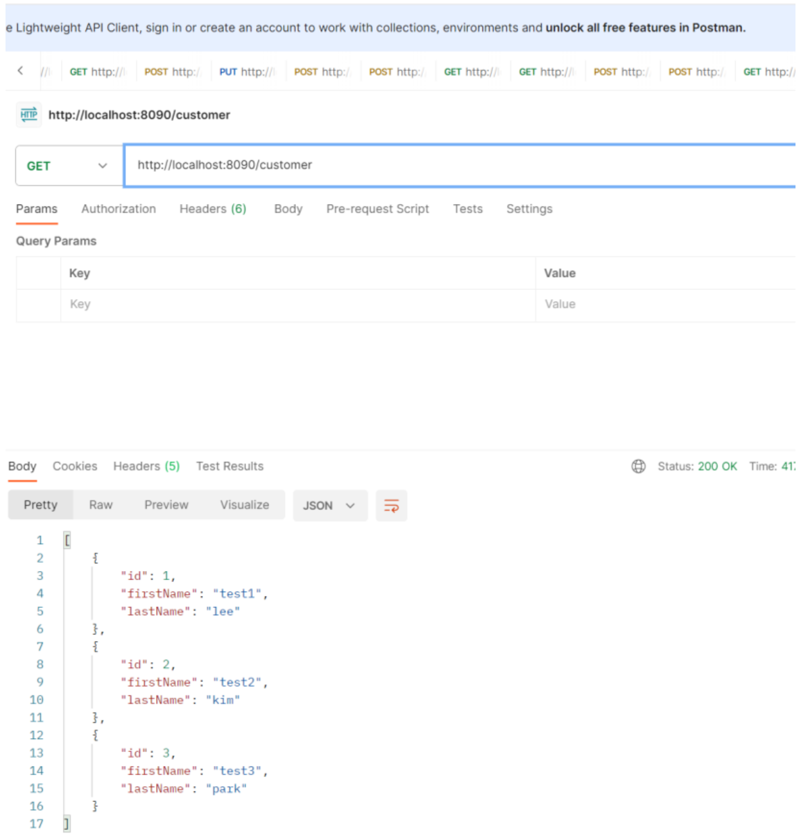


```
sh-4.4# ,mysql -u root -p
sh: ,mysql: command not found
sh-4.4# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 49
Server version: 8.2.0 MySQL Community Server - GPL

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show tables;
ERROR 1046 (3D000): No database selected
mysql> use customers;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> desc customers;
+-----------+--------------+------+-----+---------+-------+
| Field     | Type         | Null | Key | Default | Extra |
+-----------+--------------+------+-----+---------+-------+
| id        | bigint       | NO   | PRI | NULL    |       |
| firstName | varchar(255) | YES  |     | NULL    |       |
| lastName  | varchar(255) | YES  |     | NULL    |       |
+-----------+--------------+------+-----+---------+-------+
3 rows in set (0.00 sec)

mysql> select * from customerss;
ERROR 1146 (42S02): Table 'customers.customerss' doesn't exist
mysql> select * from customers;
Empty set (0.00 sec)
```

docker

docker

```
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.datasource.url=jdbc:mysql://localhost:3308/customers?useUnicode=true&serverTimezone=Asia/Seoul&characte
rEncoding=utf8
spring.datasource.username=ssafy
spring.datasource.password=ssafy
spring.sql.init.mode=always
```


e Lightweight API Client, sign in or create an account to work with collections, environments and unlock all free features in Postman.
GET http:// POST http:/ PUT http:// POST http:/ POST http:/ GET http:// GET http:// POST http:/ POST http:/ GET http://
Params Authorization Headers (6) Body Pre-request Script Tests Settings
Query Params
Key Value
Body Cookies Headers (5) Test Results
Pretty Raw Preview Visualize JSON

http://localhost:8090/customer

GET http://localhost:8090/customer

Status: 200 OK Time: 41

```
[
    {
        "id": 1,
        "firstName": "test1",
        "lastName": "lee"
    },
    {
        "id": 2,
        "firstName": "test2",
        "lastName": "kim"
    },
    {
        "id": 3,
        "firstName": "test3",
        "lastName": "park"
    }
]
```

docker

- docker

```
spring.config.activate.on-profile= customers
spring.datasource.driver-class-name=org.testcontainers.jdbc.ContainerDatabaseDriver
spring.datasource.url=jdbc:tc:mysql:8:///localhost:3308/customers?useUnicode=true&serverTimezone=Asia/Seoul
&characterEncoding=utf8
spring.datasource.username=ssafy
spring.datasource.password=ssafy
spring.sql.init.mode=always
```

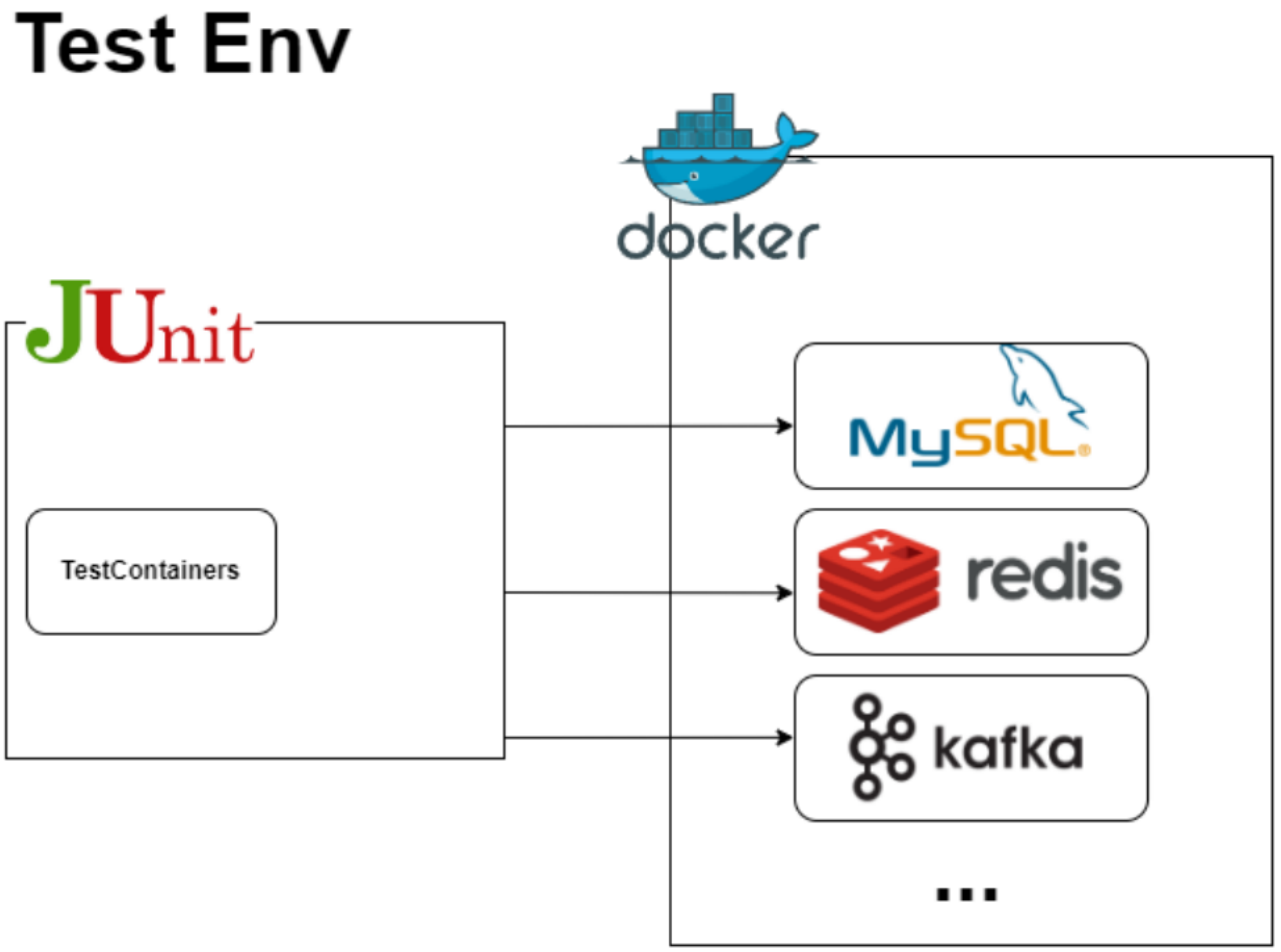
docker

- docker

```
@ActiveProfiles("customers")
@Transactional
@SpringBootTest
@Testcontainers
class CustomerIntegrationTest {

        @Test
        void create_customer(){
                Customer customer = new Customer(100L, "123", "lee");
                Customer save = customerRepository.save(customer);
                System.out.println("save = " + save.getId());
                assertEquals(save.getId(), 100L);
        }
```

docker

- docker



docker

- docker

```
spring.config.activate.on-profile= employees
spring.datasource.driver-class-
name=org.testcontainers.jdbc.ContainerDatabaseDriver
spring.datasource.url=jdbc:tc:mysql:8:///localhost:3308/customers?useUnicode=tr
ue&serverTimezone=Asia/Seoul&characterEncoding=utf8
spring.datasource.username=ssafy
spring.datasource.password=ssafy
spring.sql.init.mode=always
```

docker

- docker

```
@ActiveProfiles("customers")
@Transactional
@SpringBootTest
@Testcontainers
class DatabaseTest2 {

@Autowired
private CustomerRepository customerRepository;

@Test
void schema_script_data_should_be_three() {
customerRepository.save(new Customer(4L, "123", "lee"));
customerRepository.save(new Customer(5L, "124", "ggg"));
customerRepository.save(new Customer(6L, "125", "jjj"));
// when
//entityManager.flush();
//entityManager.clear();
System.out.println("1 ----------------" +customerRepository.findAll().size());
```

docker

- docker

```java
@Retention(RetentionPolicy.RUNTIME)
@Target(ElementType.TYPE)
@ActiveProfiles("employees")
@DataJpaTest
@AutoConfigureTestDatabase(replace =
AutoConfigureTestDatabase.Replace.NONE)
public @interface TCDataJpaTest {
}
```

docker

- docker

```
@TCDataJpaTest
class CustomerRepositoryTest extends AbstractContainerBaseTest {


@Autowired
private CustomerRepository customerRepository;


@Test
void schema_script_data_should_be_three() {
customerRepository.save(new Customer(4L, "123", "lee"));
customerRepository.save(new Customer(5L, "123", "lee"));
customerRepository.save(new Customer(6L, "123", "lee"));
List<Customer> customers = customerRepository.findAll();
customers.forEach(System.out::println);
assertEquals(customers.size(), 3);
}
```

docker

- application-test.properties

```
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.datasource.url=jdbc:mysql://localhost:3308/customers?useUnicode=t
rue&serverTimezone=Asia/Seoul&characterEncoding=utf8
spring.datasource.username=ssafy
spring.datasource.password=ssafy
spring.jpa.database=mysql
```

docker

- StudentRepositoryTest 직접 docker mysql

```
@SpringBootTest(
properties = {
"spring.config.location=classpath:application-test.properties"
}
)
class StudentRepositoryTest {


@Autowired
private EmployeeRepository employeeRepository;


@Test
public void givenStudentObject_whenSave_thenReturnSavedStudent(){
Employee employee = new Employee(0L,"kangzt", "010-1234-9876", "kangt@naver.com", "kafka",
"kafka stream");
Employee savedemployee= employeeRepository.save(employee);
Assertions.assertNotNull(savedemployee);
Assertions.assertNotNull(savedemployee.getId());
```

docker

MyBatis 패키지 동일하게

```
package com.honey.edu.employee.model.mapper;

import com.honey.edu.common.PageRequest;
import com.honey.edu.employee.model.Employee;
import com.honey.edu.employee.model.EmployeeExample;
import com.honey.edu.employee.model.EmployeeExample.Criteria;
import com.honey.edu.employee.model.service.EmployeeService;

import lombok.extern.slf4j.Slf4j;

import static org.assertj.core.api.Assertions.assertThat;
import static org.junit.jupiter.api.Assertions.*;

import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;

@Slf4j
@MybatisTest
@AutoConfigureTestDatabase(replace = AutoConfigureTestDatabase.Replace.NONE)
class EmployeeMapperTest {// extends AbstractContainerBaseTest {

                @Autowired
    private EmployeeMapper employeeMapper;
```

docker

- docker

```
            @Autowired
private EmployeeMapper employeeMapper;

// JUnit for save student operation - BDD style
@Test
@DisplayName("Order Mapper Test")
public void testselectByEmployee(){
            EmployeeExample empcriteria=new EmployeeExample();
                            empcriteria.setOrderByClause("employee_id desc");
                            assertEquals(107, employeeMapper.selectByExample(empcriteria).size());

                            log.debug( "=====================================" );
                            //assertThat(employeeMapper.selectByExample(empcriteria).size()).isEqualTo(107);
}
```

docker

- docker-compose -f docker-compose.yml up -d

```
version: '3.1'

services:
mysql:
image: mysql:8.0
container_name: mysqldb
environment:
- MYSQL_DATABASE=customers
- MYSQL_ROOT_PASSWORD=ssafy
ports:
- 3308:3306
volumes:
- D:\java:/data/mysql
```

docker

- docker

```
D:₩java₩mysqlyml>docker run --name mysqldb  -e MYSQL_ROOT_PASSWORD=ssafy -d -p
3308:3306 mysql -v D:/java/docker:/var/lib/mysql
Unable to find image 'mysql:latest' locally
latest: Pulling from library/mysql
e9f2695d7e5b: Pull complete
c041cd0148ec: Pull complete
27c9fbf7aa29: Pull complete
62fc1efc1f1f: Pull complete
e1d25a6611c2: Pull complete
5846de7fe479: Pull complete
faf13e3256e8: Pull complete
2217ed684a4f: Pull complete
45bfd3acf105: Pull complete
5b68afdb04ae: Pull complete
Digest: sha256:6057dec95d87a0d7880d9cfc9b3d9292f9c11473a5104b906402a2b73396e377
Status: Downloaded newer image for mysql:latest
3092e418fc1ce1880a7d86ad764f9594c8f7df3d36853a6cb1618ba99e00b2c5
```

docker

- docker