|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ Информатика, искусственный интеллект и системы управления

КАФЕДРА Системы обработки информации и управления\_\_\_\_\_\_\_\_\_\_\_

**ОТЧЕТ**

**К ЛАБОРАТОРНОЙ РАБОТЕ № 2**

**По дисциплине «Автоматизация развертывания и эксплуатации программного обеспечения»**

**Контейнеризация Docker**

Студент ИУ5Ц-91Б  А.В. Рябкин

(Группа) (Подпись, дата) (И.О.Фамилия)

Преподаватель  А.Р. Якубов

(Подпись, дата) (И.О.Фамилия)

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

План выполнения лабораторной работы:

Часть 1. Базовые команды Docker

1.1. Подготовка рабочего окружения

1.2. Образа - docker pull, docker images

1.3. Метки и удаление образа - docker tag

1.4. Запускаем контейнер - docker run, docker logs

1.5. Списки контейнеров - docker ps

1.6. Подключаемся к работающему контейнеру - docker exec

1.7. Список изменений - docker diff

1.8. Завершаем контейнер - docker stop, docker kill, docker rm

1.9. Не теряем данные - docker volume

1.10. Контейнер Adminer

1.10. Сети - docker network

Часть 2. Продвинутая работа с Docker

2.1. Настройка базы данных

2.2. Запускаем Adminer

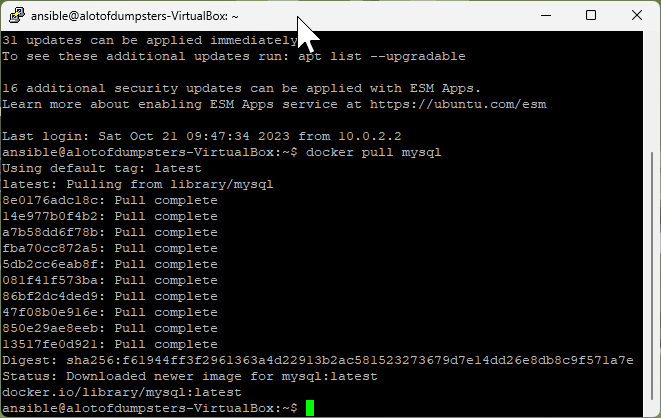
2.3. Запускаем свой сервис

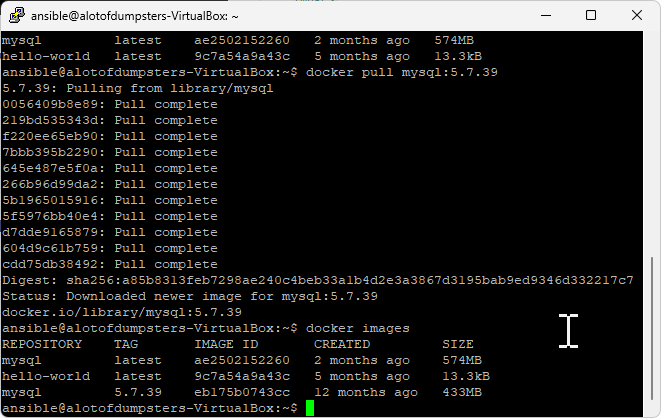
2.4. Подробнее про сборку образа

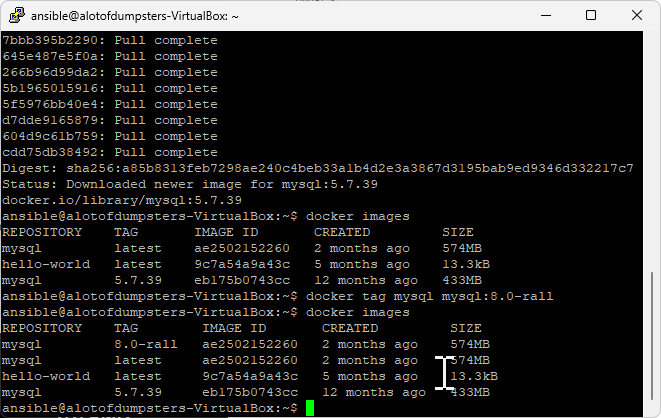
2.5. Оптимизируем сборку

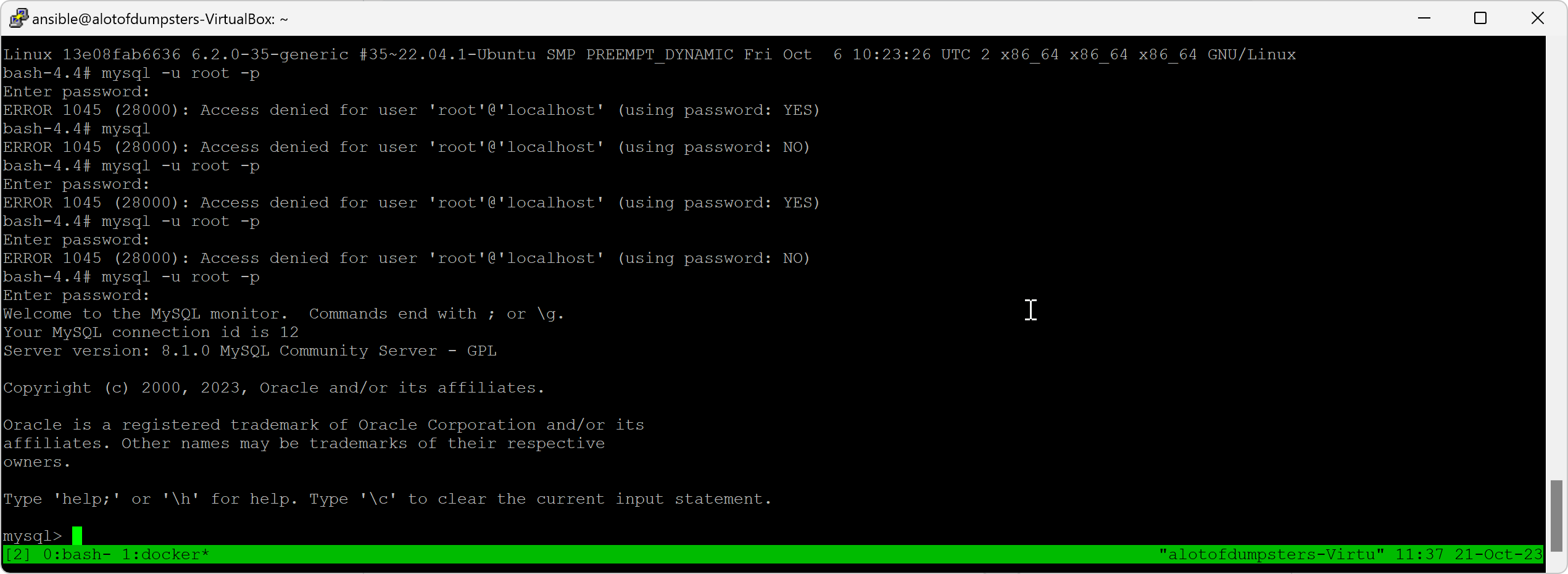
2.6. Многоэтапная сборка

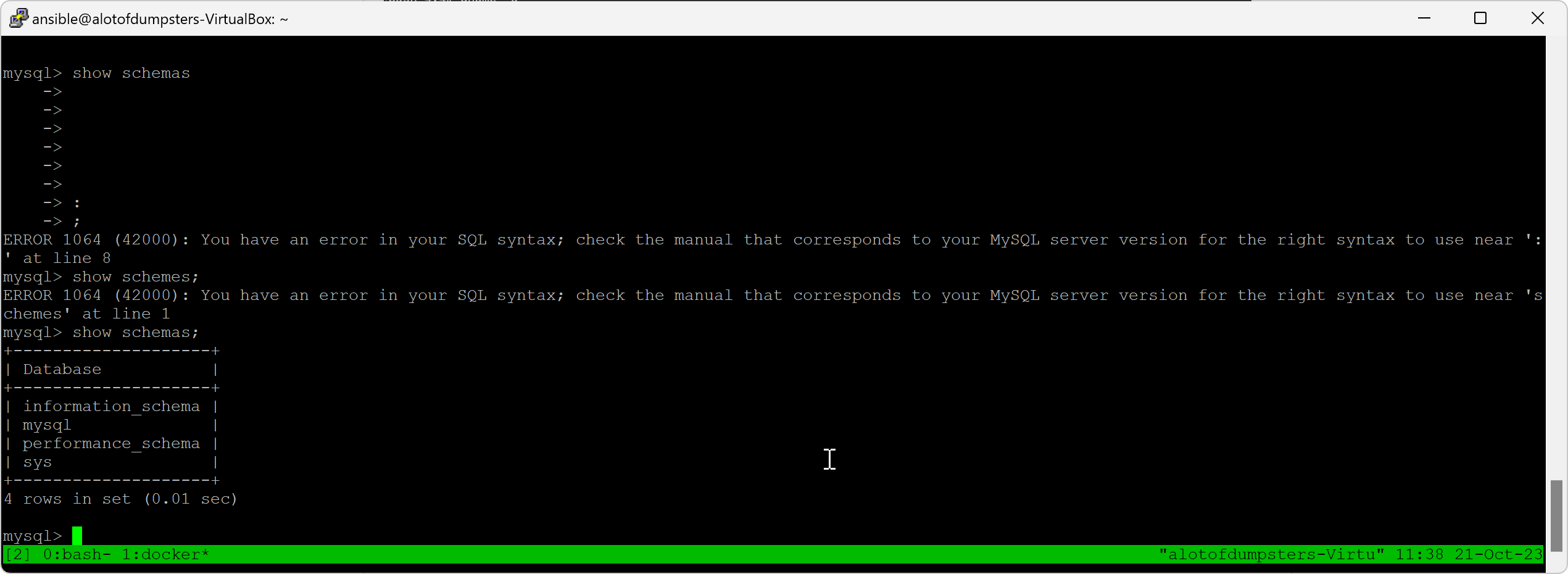
2.7. Делимся образом - docker push

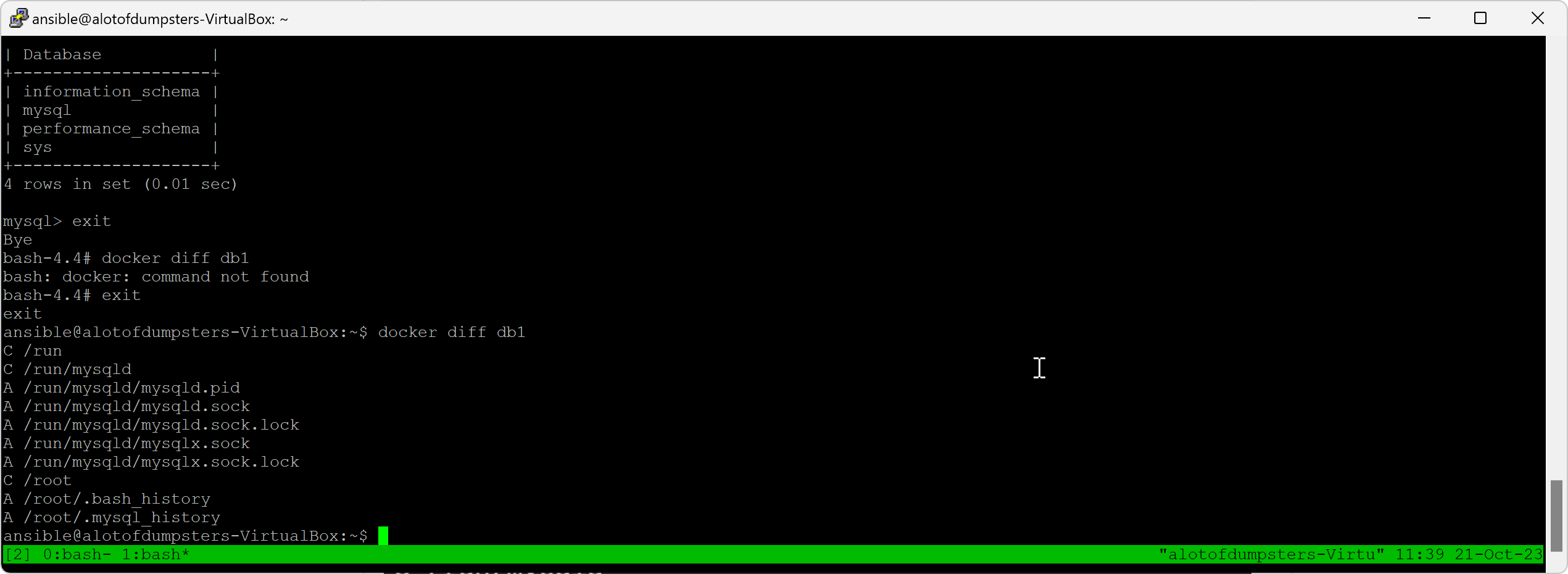


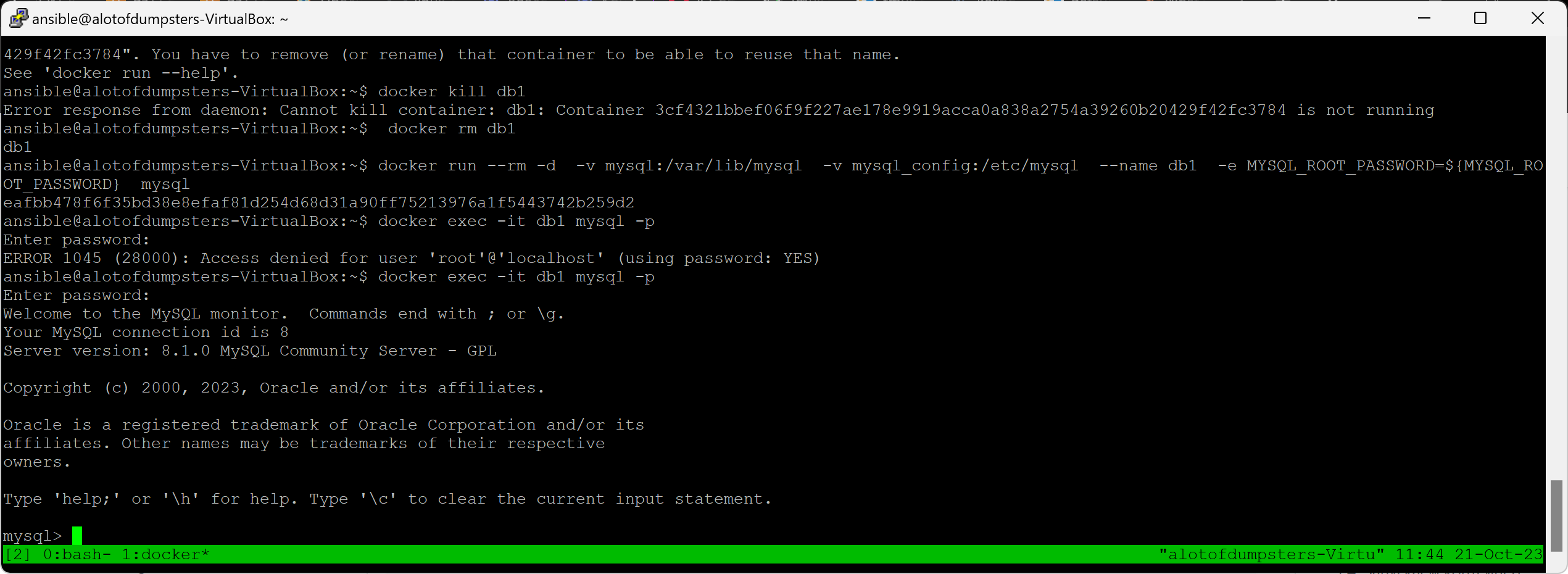


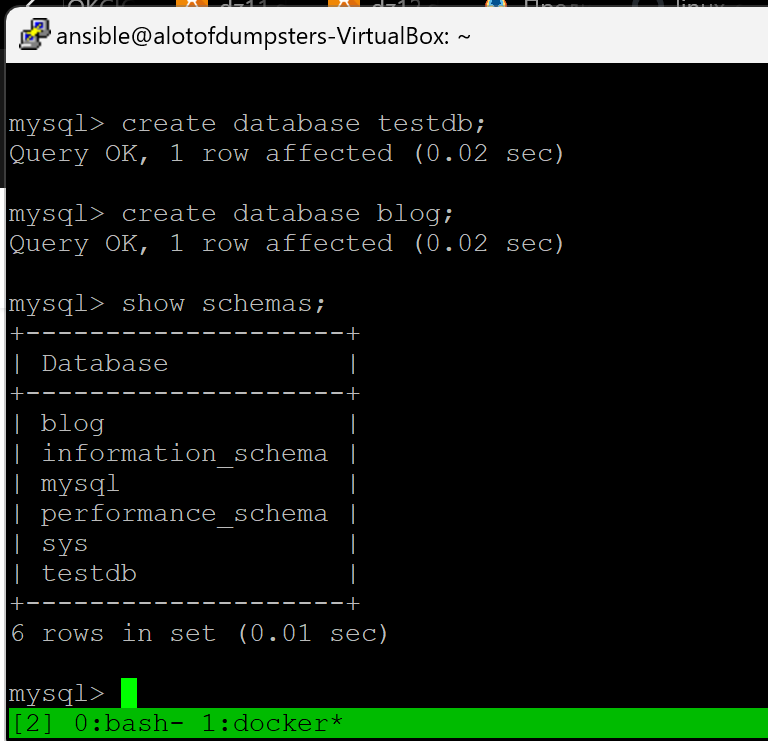


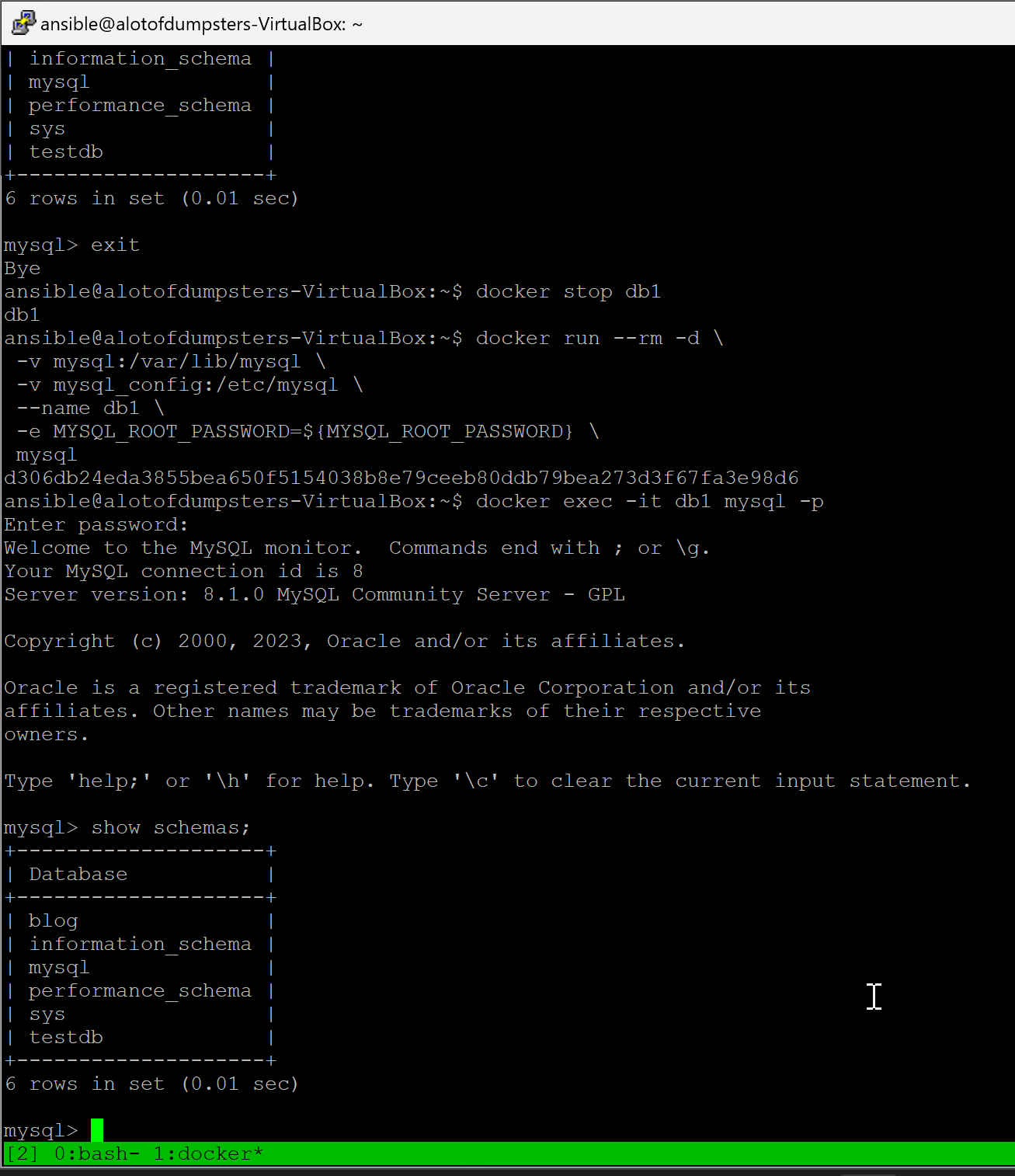


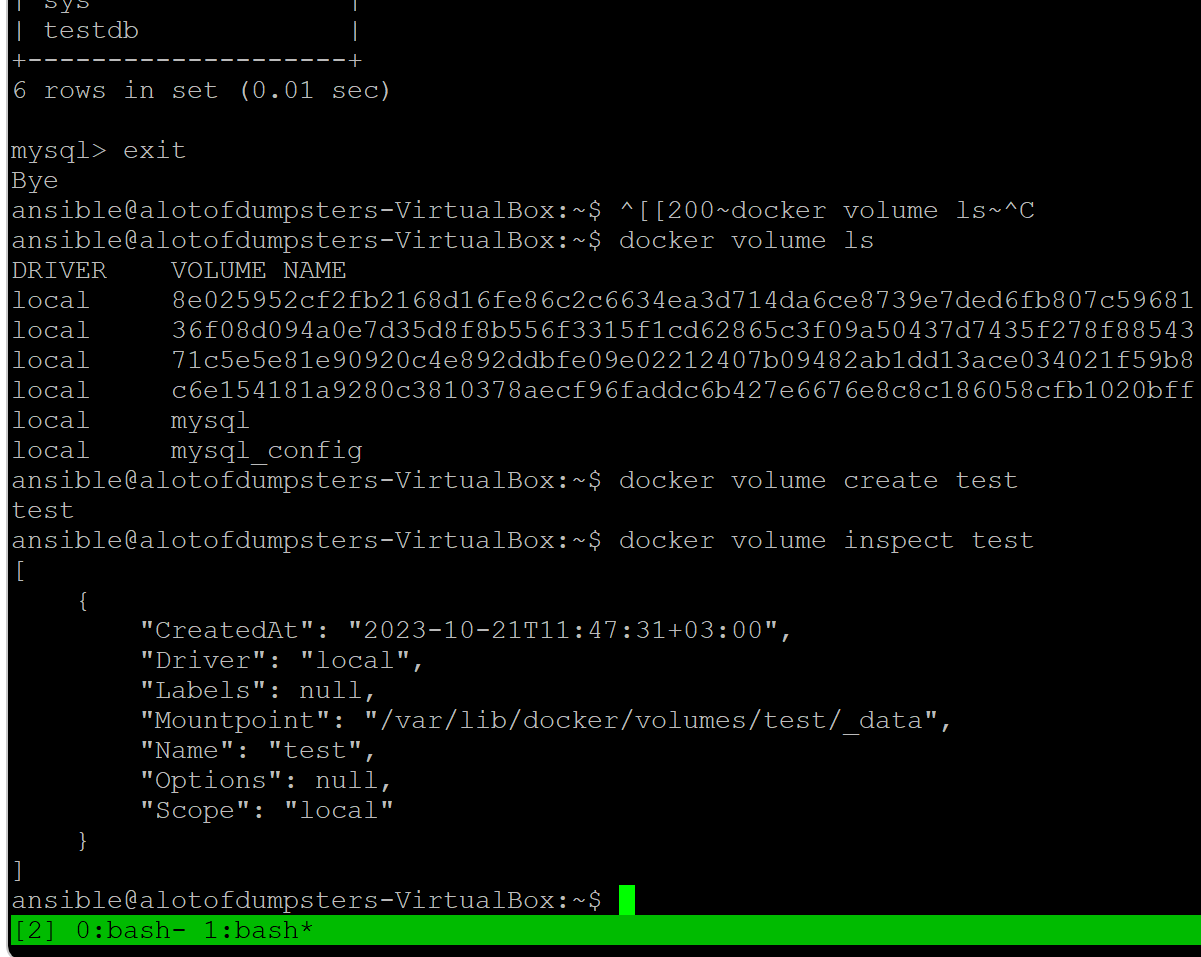


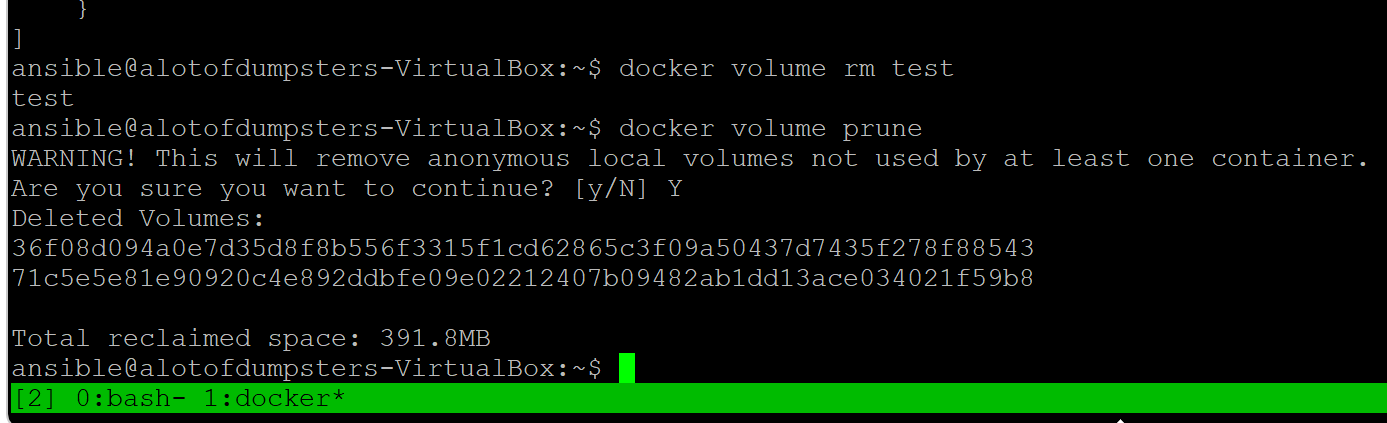


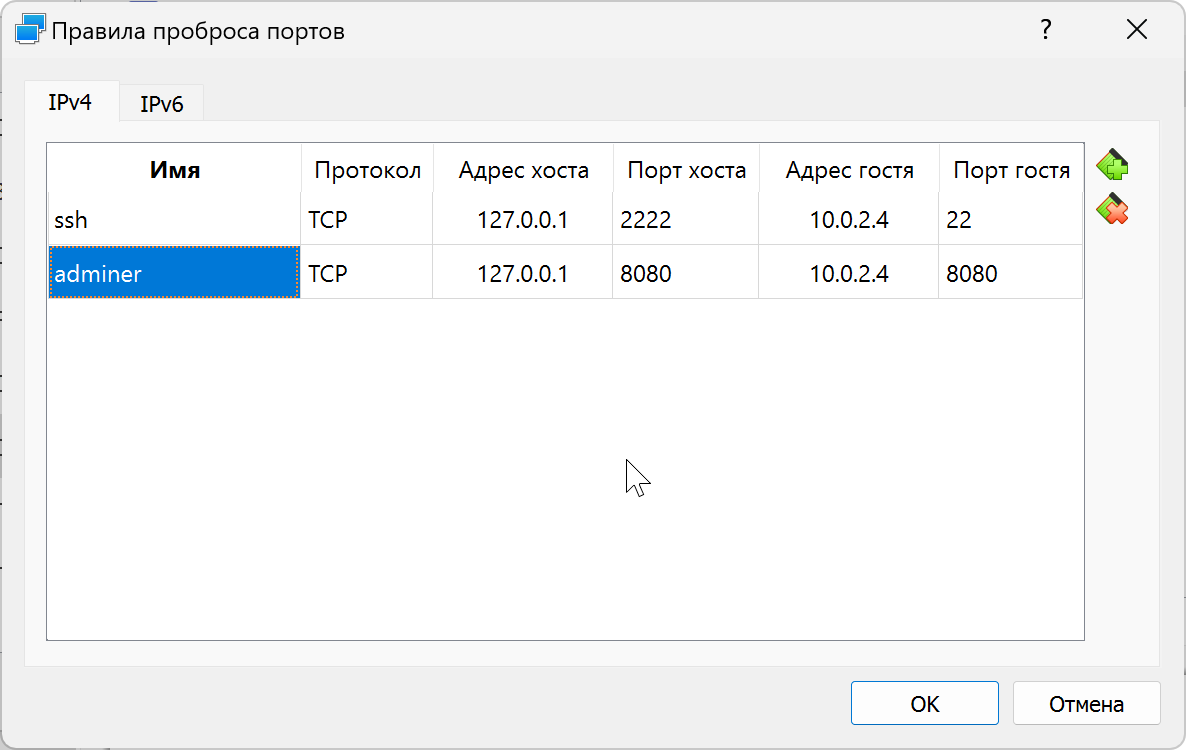


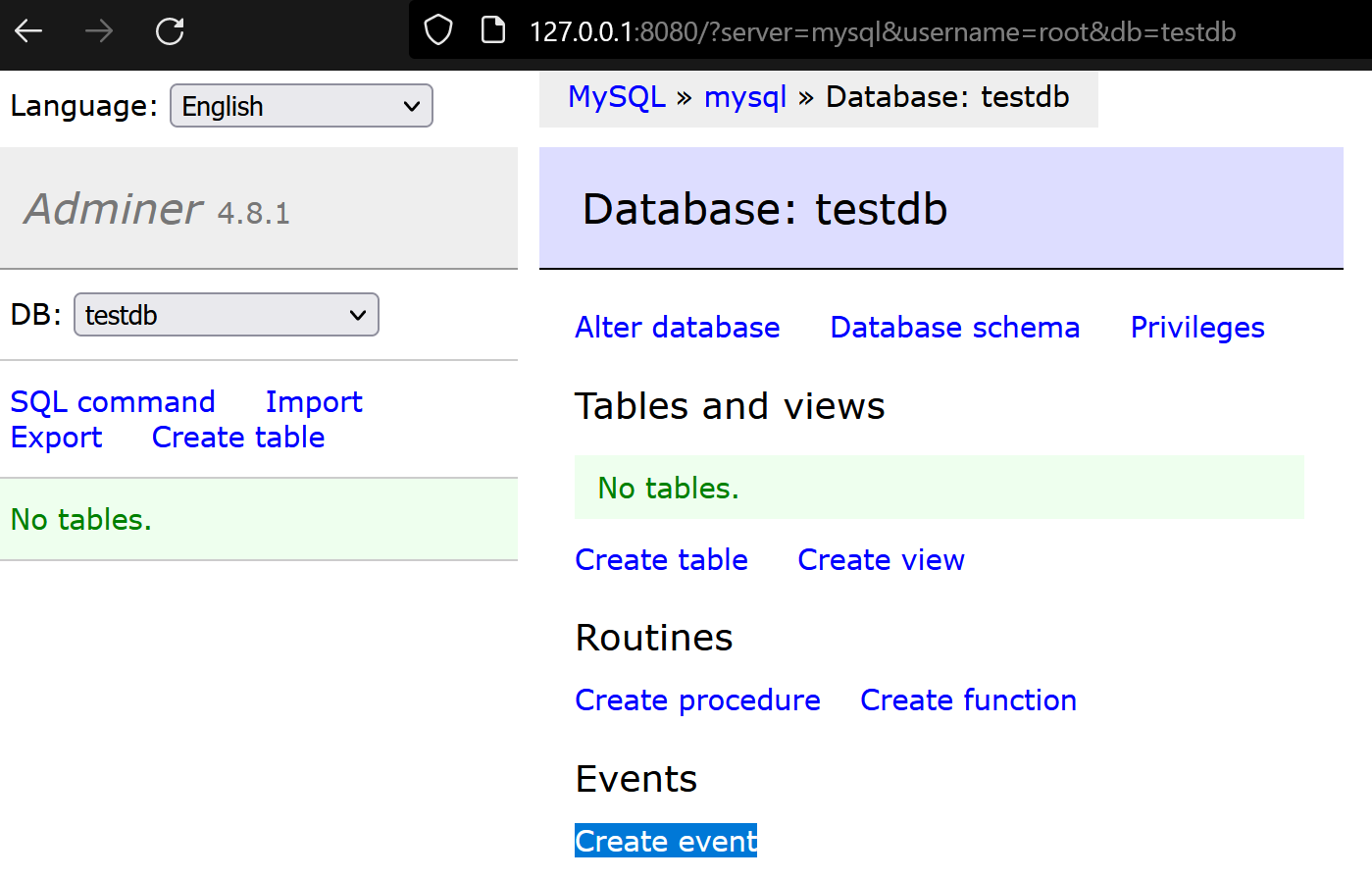


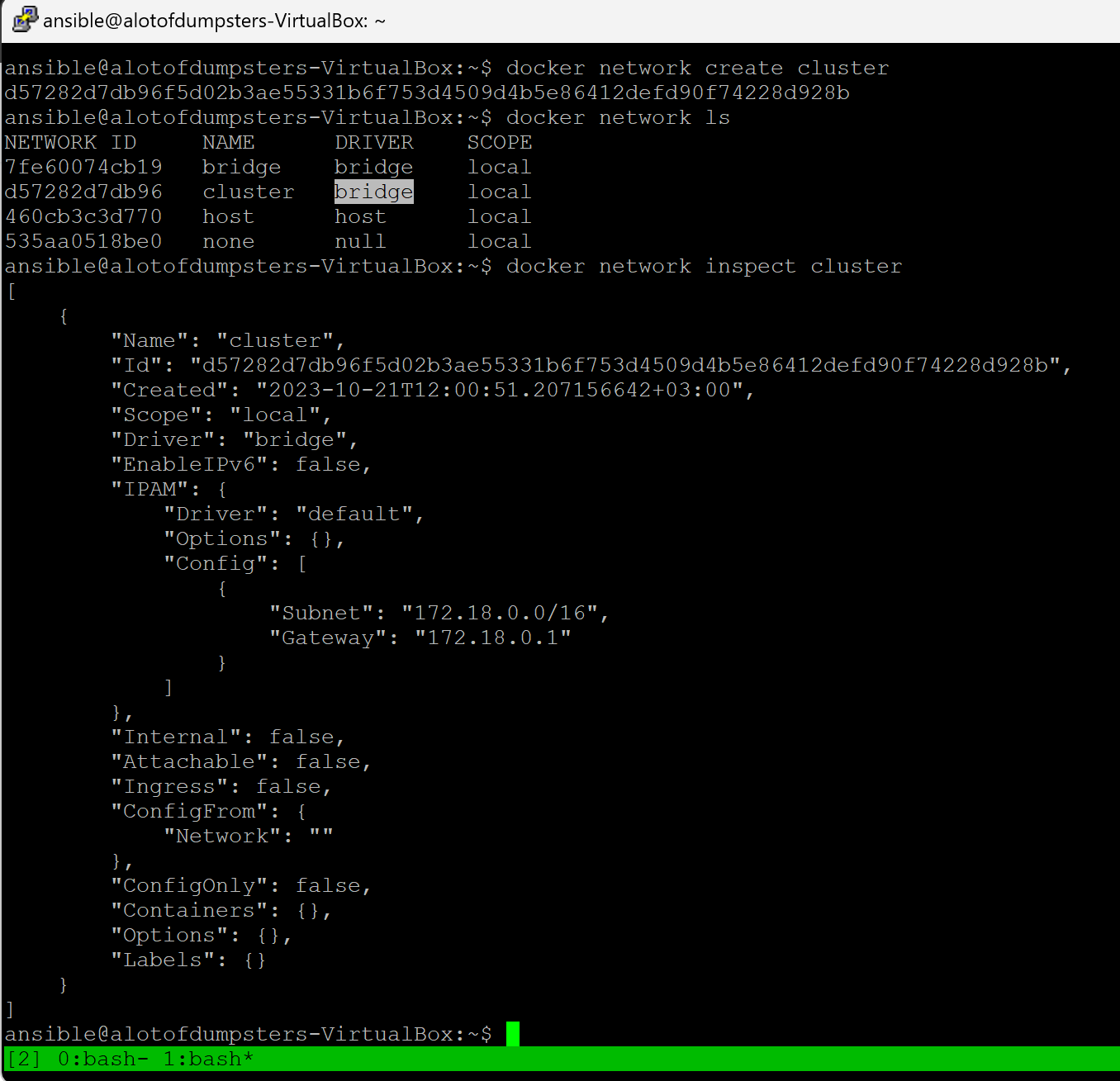


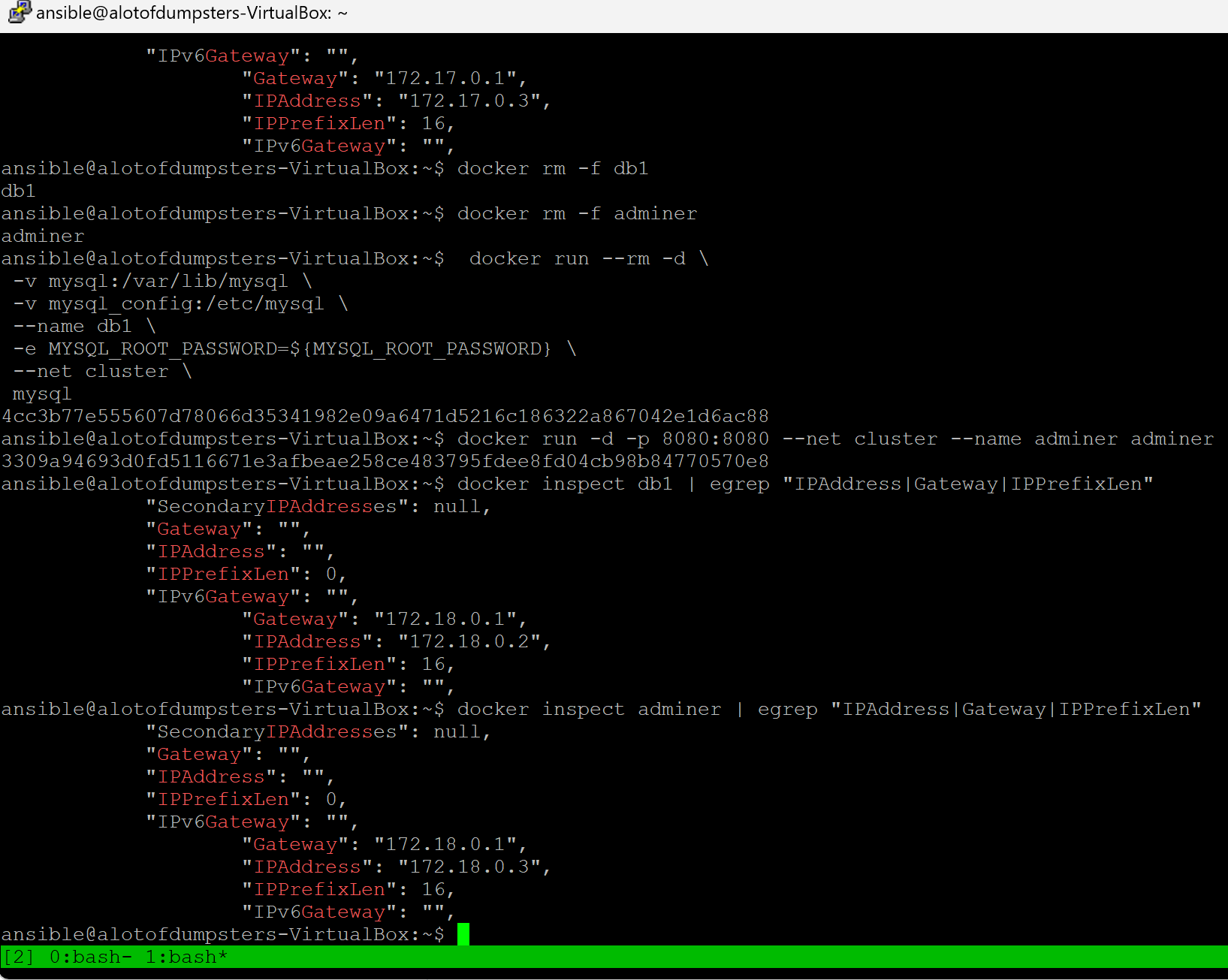


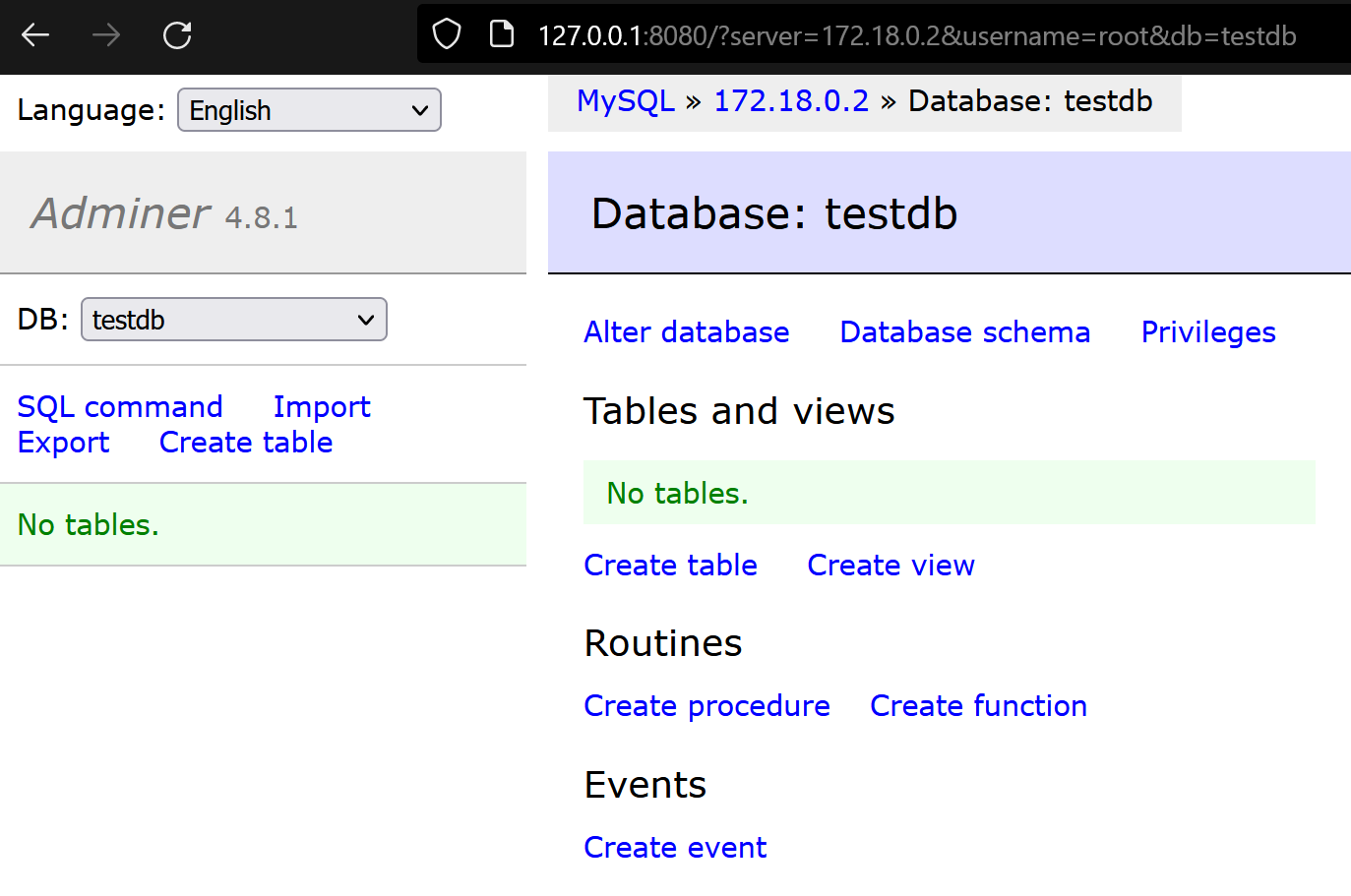


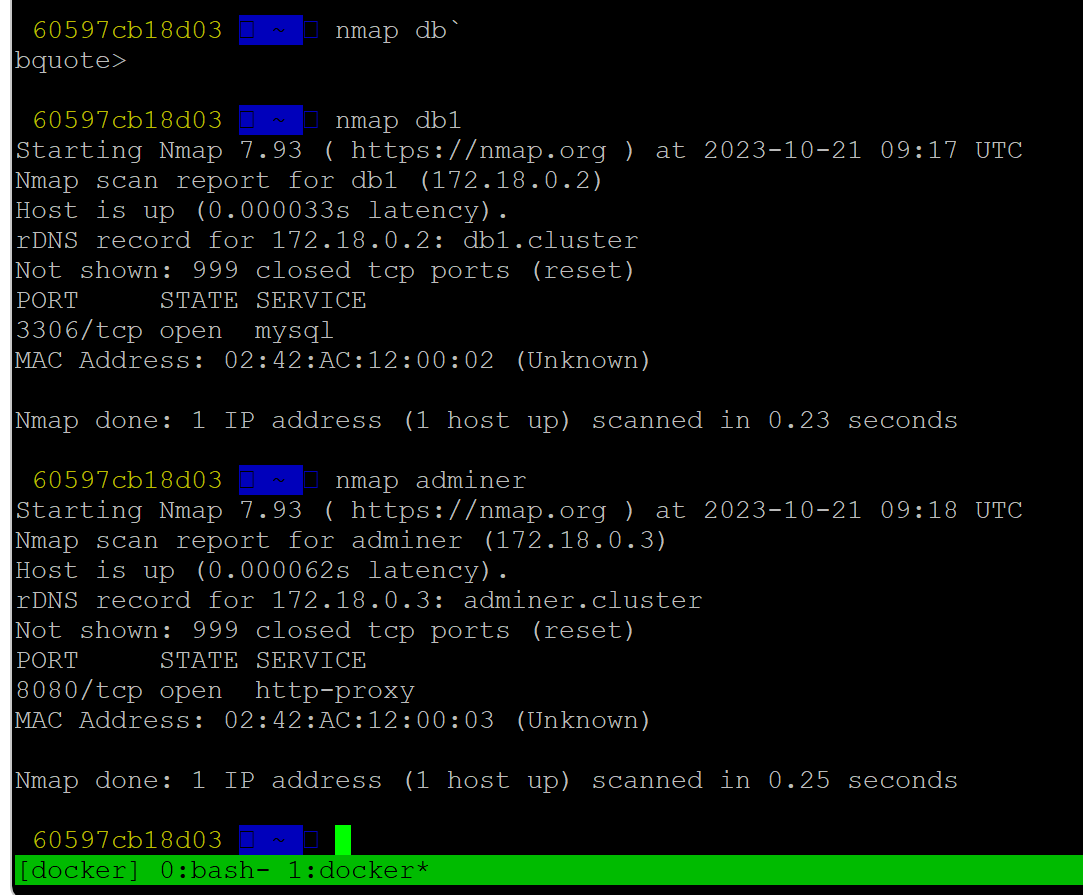


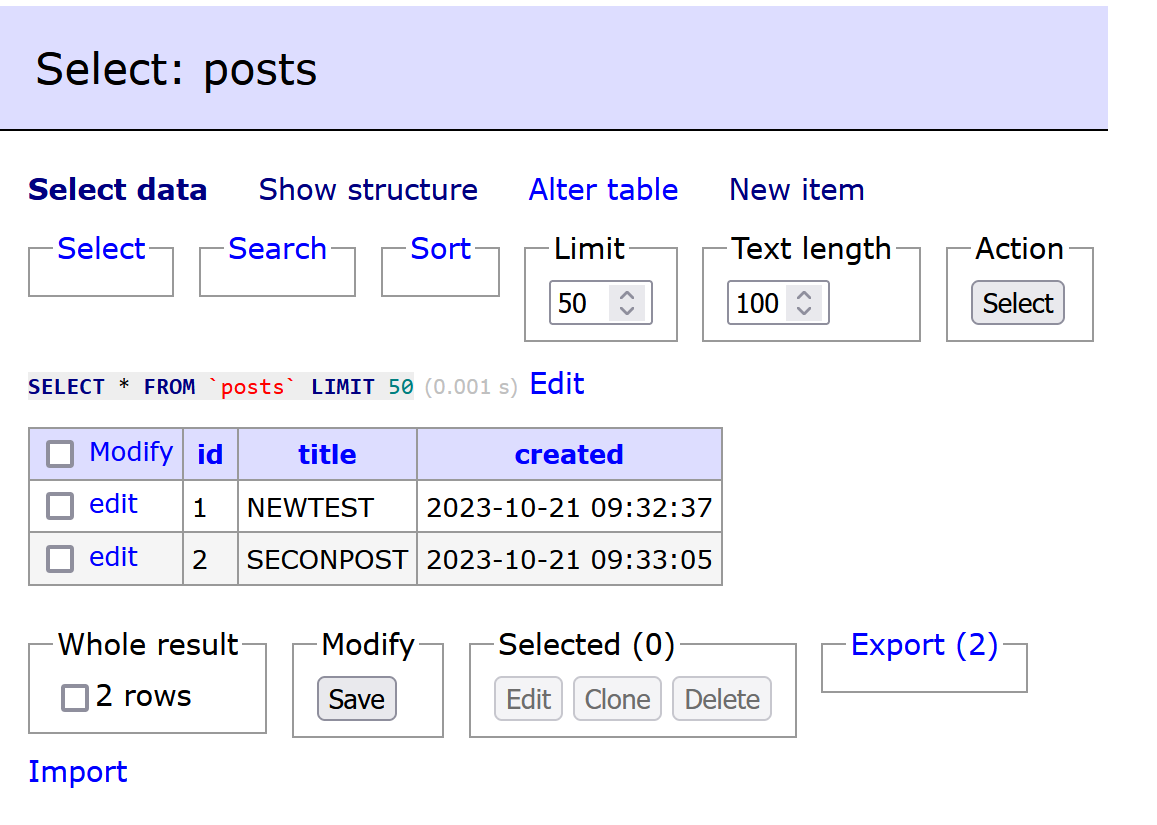


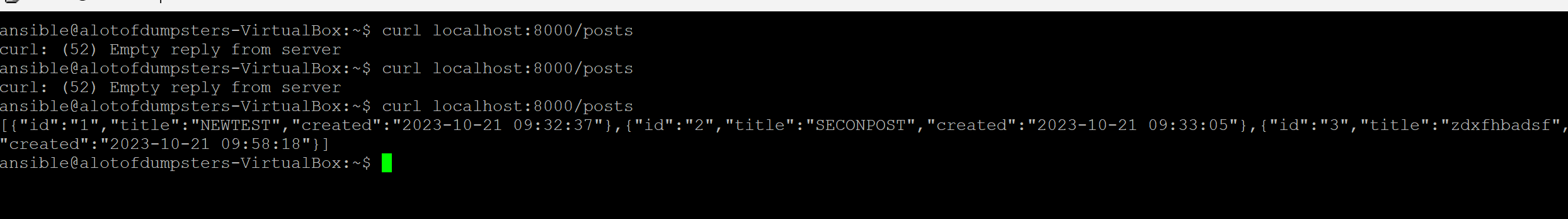


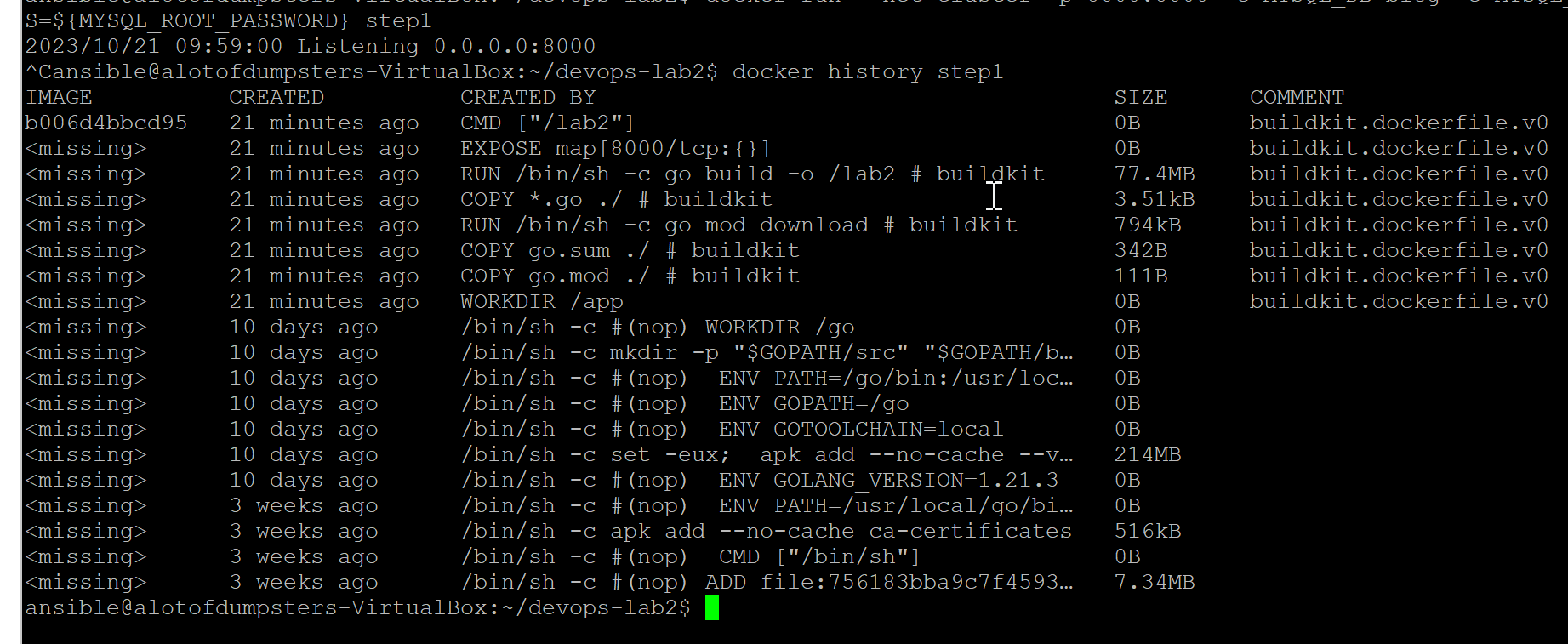


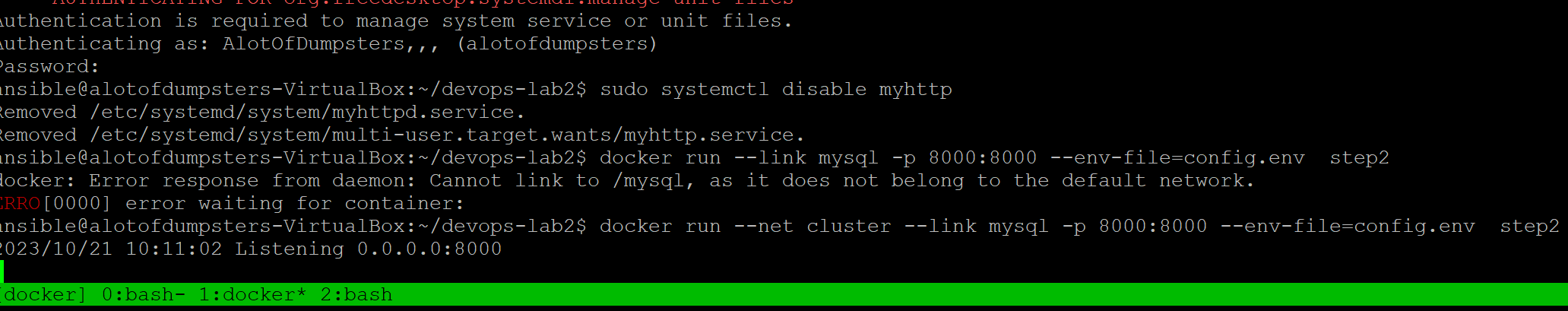


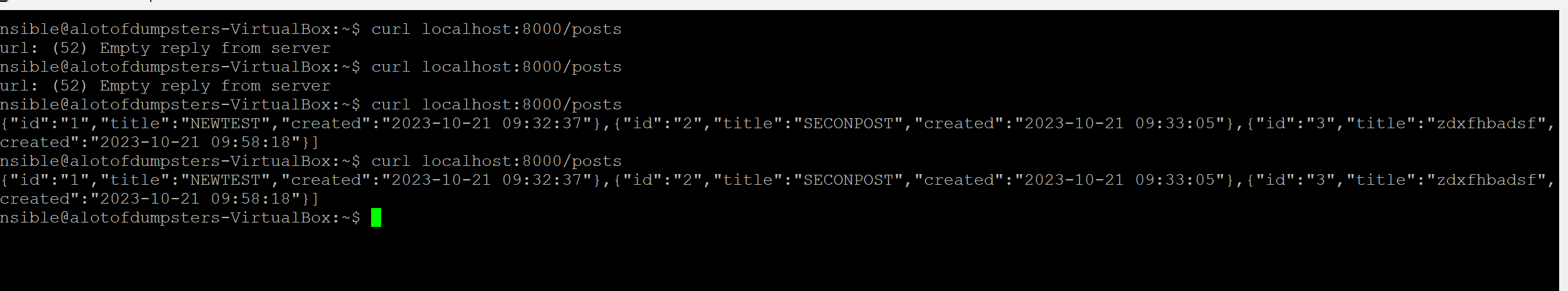


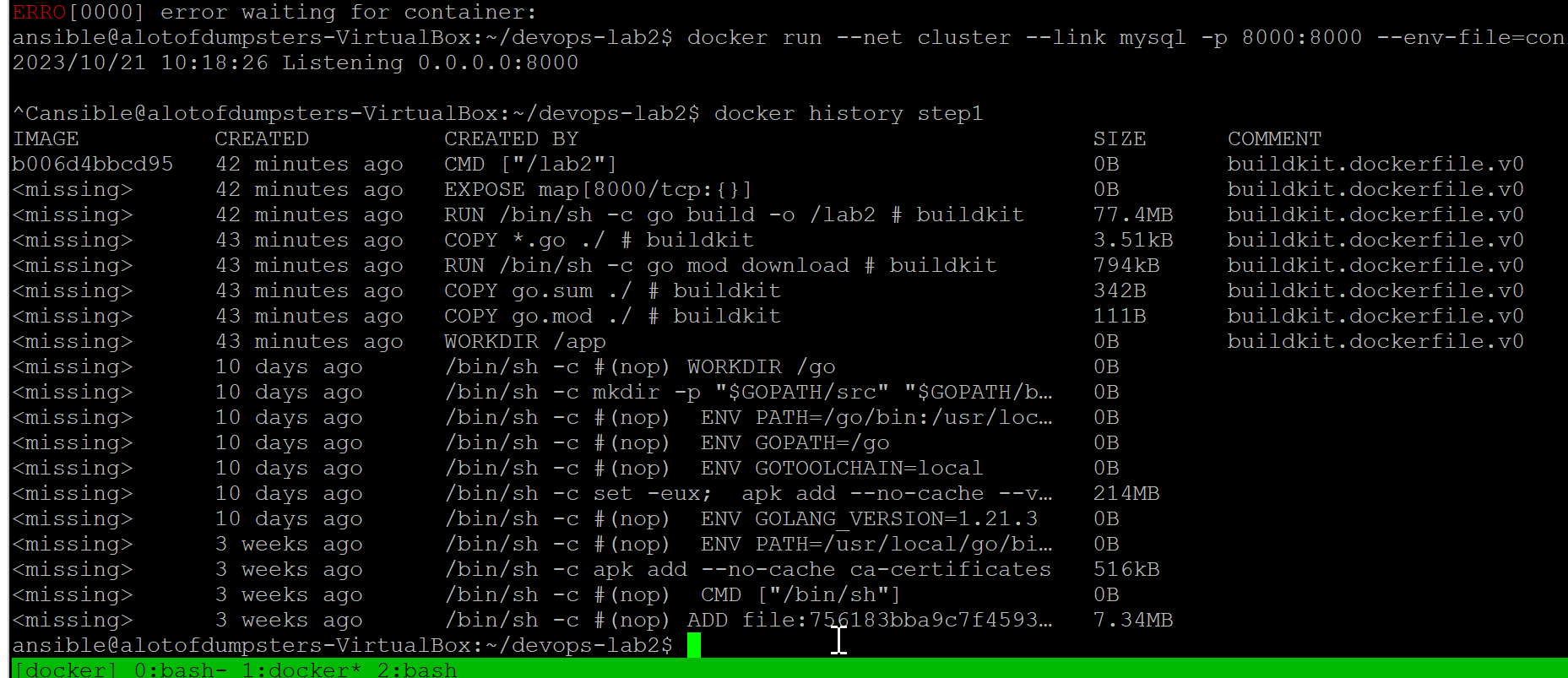


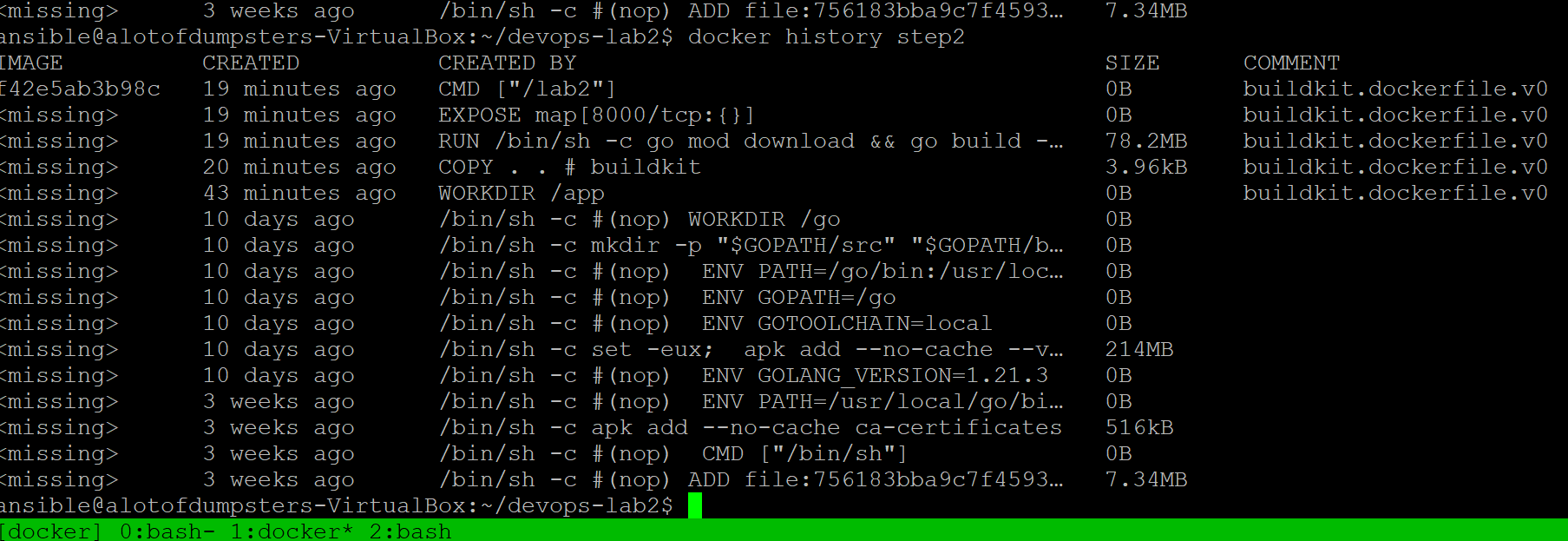


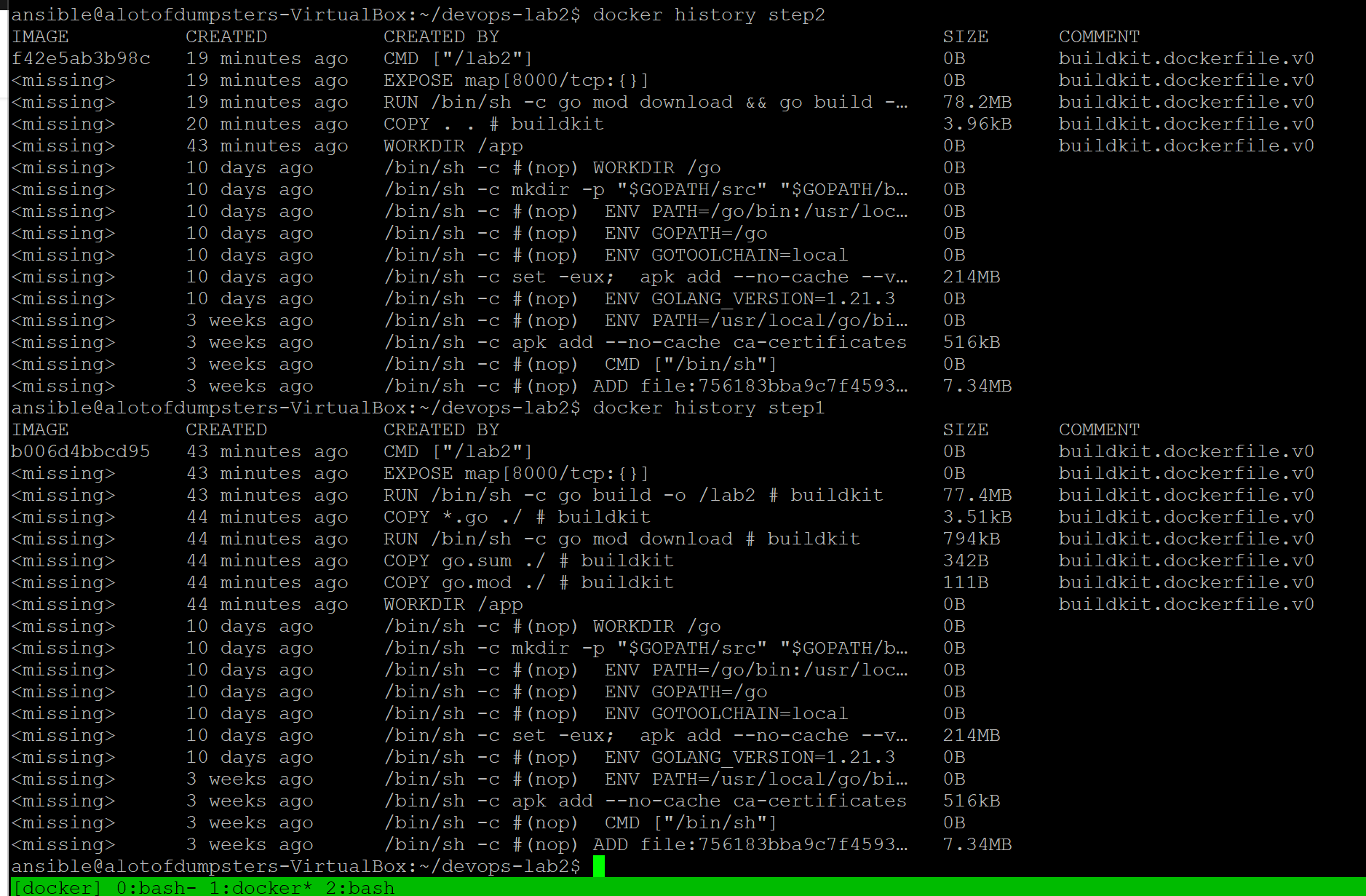


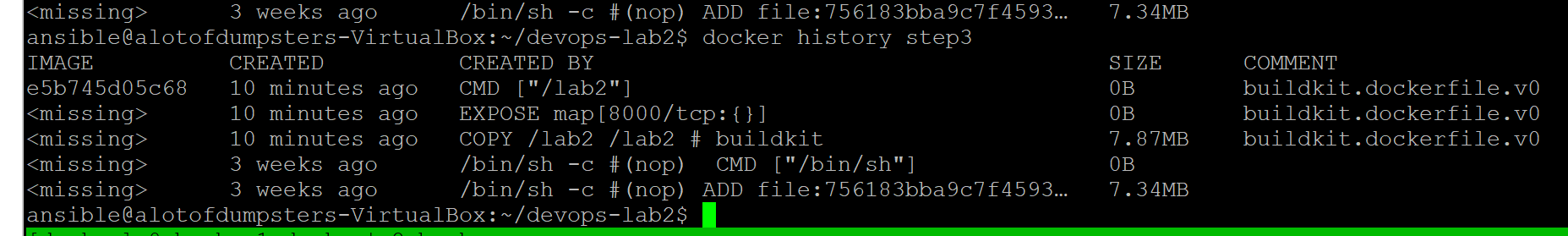












Разница между степ1 и степ2: степ2 оптимизирован, в нем меньше строк, операция копирования копирует сразу все в рабочую директории, а не в две команды для двух отдельных файлов или диркеторий как в степ1. Правда я не знаю про необходимость всего, а не только нужной части. Еще в степ2 две команды в одну переделали при помощи &, но это не должно же влиять на скорость, тут просто строчку сокартили. Мне кажется что в степ1 неоптимизирован именно процесс копирования и загрузки, их порядок.

В степ3 уже используется builder - это принципиальное отличие. Посмотрим на https://docs.docker.com/engine/reference/builder/ , тут не нашел, посмотрим тут https://blog.alexellis.io/mutli-stage-docker-builds/ , тут хорошо обьясняют мультистажную сборку. В итоге у нас два этапа, на первом собирается все, а на втором доставляется.

Сравнив все три истории можно заметить, тчо они последовательно уменьшаются и в степ3 все что есть это, в общем-то, операция копирования и открытия порта.

Мне так же потребовалось добавить --net cluster чтобы обе команды для степ2 и степ3 запускались нормально, иначе говорило что майскул не на дефолтной подсети