Title: docker-compose with multiple databases

Post Body:

I'm trying to figure out how to implement docker using docker-compose.yml with 2 databases imported from sql dumps.

```
httpd:     container_name: webserver     build: ./webserver/     ports:         - 80:80     links:         - mysql         - m
```

The above returns the following:

```
Kronos:mybuild avanche$ ./run.sh  Creating sqlserver Creating webdata Creating sqlserver2  ERROR: for mysql2  driver failed pr
```

Basically, I'm trying to get my whole stack setup in a single docker compose file, create 2 databases and import the respective sql dumps. Anyone have any suggestions?

Accepted Answer:

Just as an update to anyone else who may look into this.

I solved this by removing:

```
MYSQL_DATABASE: dbname
```

from **docker-compose.yml** and adding the relevant create database statements directly to the sql file being passed to **docker-entrypoint-initdb.d**.

At that stage, sql commands are performed under root, so you'll also need to add a statement to grant relevant permissions to the database user you want to use.

Highest Rated Answer:

Multiple databases in a single Docker container

The answers elsewhere on this page set up a dedicated container for each database, but a single MySQL server is capable of hosting multiple databases. Whether you should is a different question, but if you want multiple databases in a single container, here's an example.

docker-compose.yml:

```
version: '3'  volumes:   db:     driver: local  services:   db:     image: mysql:5.7     command: mysqld --character-set-serve
```

docker/provision/mysql/init/01-databases.sql:

```
# create databases CREATE DATABASE IF NOT EXISTS `primary`; CREATE DATABASE IF NOT EXISTS `secondary`;  # create root user and
```

How does this work?

This works because the MySQL Docker project has an entrypoint script that will run through all files in the `/docker-entrypoint-initdb.d` folder, if it exists. This is useful for setting up databases and initializing their schema and data. In docker-compose, we're using `volumes` to map that virtual folder to a folder on the host system.