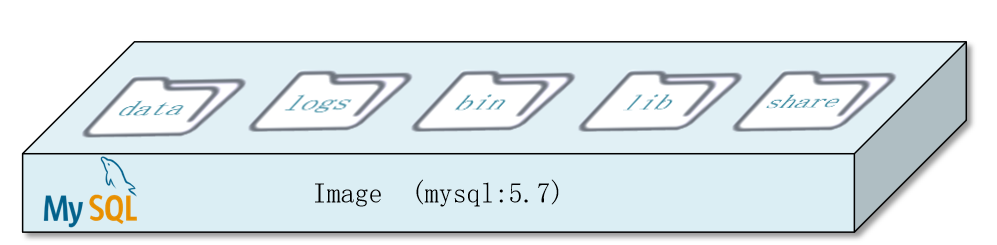
## Docker架构：

已经学习了Docker的原理，为了更好的使用docker，还需要进一步的学习docker的架构：镜像 + 容器。

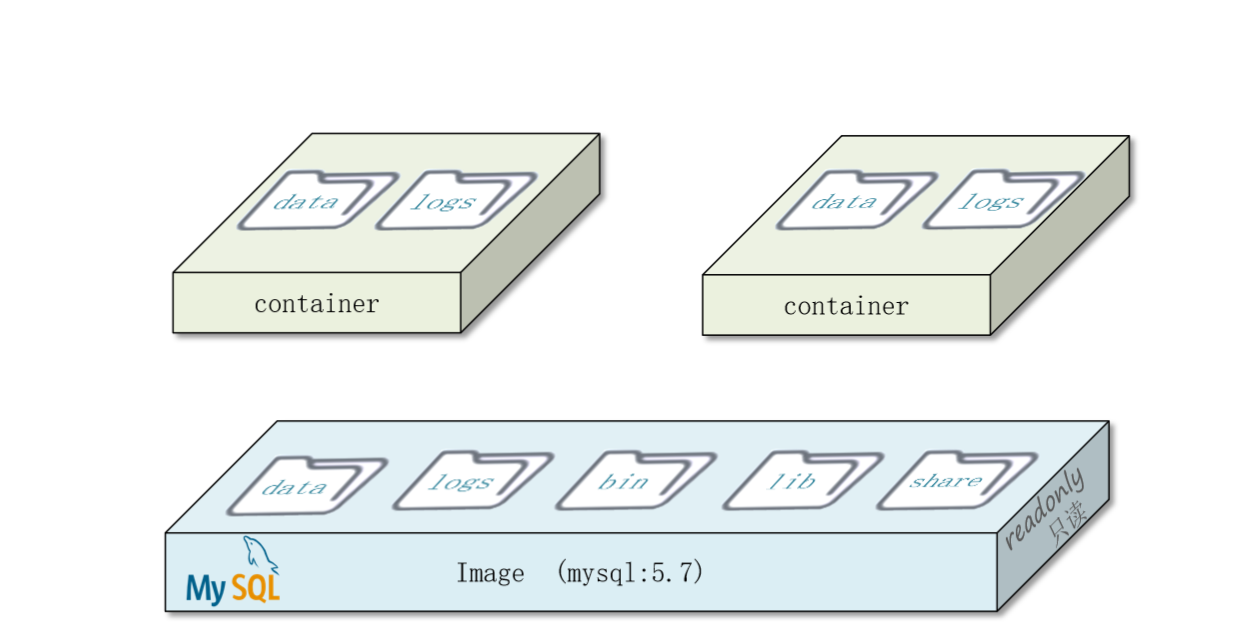
### 容器和镜像

### 镜像（image）：Docker将应用程序及其所需要的依赖、函数库、环境、配置等文件打包在一起，这个东西就称为镜像。

比方说我们之前说的mysql镜像，它里面就会有各种各样所需要的依赖、函数库，这些东西东西最终落地到硬盘，就是一个个的文件，比方说下图中有MySQL运行时需要写数据的data目录文件，还有log日志文件，还有bin可执行文件。。。。这些就组成了mysql应用本身，还有就是Mysql所需要的各种各样的函数库。也就是说，镜像就是硬盘中的文件。



### 容器（Container）：镜像中的应用程序运行后形成的进程就是容器，只是Docker会对容器做隔离，互相对外不可见。



可以简单理解成，把应用（镜像）跑起来，启动的进程，就是容器，只不过在Docker里面，容器还要做隔离，也就是说，容器可以看作是小盒子，这个小盒子里边，会利用linux的手段，给他形成隔离空间，里边会**有自己独立的CPU资源、内存资源、甚至于还有独立的文件系统，那么因此在容器内运行的进程就会以为自己是这台计算机上的唯一进程，从而起到一种隔离的效果**。比如，Mysql镜像，不管是启动成一个容器，还是启动成多个容器，容器之间都是相互隔离的！那么当容器运行的过程中，必然会做数据读写的操作，比方，mysql容器肯定要存数据，存在data目录，但是mysql容器不会把数据写到mysql镜像的data目录，因为如果这样做了，就对mysql镜像产生污染了，不纯洁了，会导致镜像将来不可用。所以说，镜像都是只读的，就是说，**容器在运行过程中，不能往镜像里写东西，只可以基于镜像创建容器，然后容器可以从镜像中读数据，但不能写**。容器要写数据，很简单，从镜像拷贝一份文件到容器自己独立的文件系统中，写数据的时候写到自己容器的空间里，不会对别的容器产生影响，也不会对镜像产生影响，这就是隔离的特性。

这样，将来我们就能基于镜像创建全新的容器，这样，镜像就能放心大胆地交给别人去使用，因为镜像不会被干扰，也不会被影响。

问题来了，我们怎样把镜像共享给别人使用？DockerHub！

### DockerHub

DockerHub是一个Docker镜像的托管平台，类似与GitHub，这样的平台成为Docker Registry（可以称为镜像服务器、镜像仓库），可以有远程Docker Registry、私有Docker Registry。其实就是类似于Maven仓库、GitHub。

我们可以利用docker提供的一些命令，去完成镜像的构建，构建出Redis镜像、MySql镜像。。。。而后把镜像上传到DockerHub。

### Docker架构：

Docker是一个CS（客户端、服务端）架构的程序，由两部分组成（注意：很显然我们安装的docker是完整的S端和C端）；

1：服务端（Server）**:Docker的守护进程，负责接收、处理Docker指令，管理镜像、容器等**。（Server端负责接收用户指令，完成镜像、容器各种各样的操作：构建镜像、拉取镜像、运行容器啊。。。）

2：客户端（client）：**给Server下指令的**，通过命令或者RestAPI向Docker服务端发送指令，可以在本地或者会远程向服务端发送命令。Client端可以向Docker下达命令或者是发送RestAPI的请求，当然，这是看你在哪发，如果你就在Docker所在的机器上，本地来发，命令就可以。如果远程操作docker，就用RestAPI。

比如：**构建镜像的docker build 命令**，这个命令到达DockerServer以后，会被Docker的守护进程：docker daemon守护进程去接收和处理，docker daemon会利用你提供的数据给你构建成一个镜像，

又比如，**拉取镜像的docker pull命令**。这个命令到达DockerServer以后，会被Docker的守护进程：docker daemon守护进程去接收和处理，docker daemon就会去DockerHub拉取你指定的镜像了。

又比如运行镜像，**创建容器的docker run命令**。这个命令到达DockerServer以后，会被Docker的守护进程：docker daemon守护进程去接收和处理，docker daemon帮助你完成容器的创建。

虽然我们可以自己构建镜像，也可以从远程拉取，但是一般情况下，向Redis，nginx，Mysql直接使用官方提供的就行，只有一些我们自己的微服务或者我们需要深度定制，才需要自己构建镜像。

因此，我们一般使用Docker就是首先向DockerServer发命令，获取镜像，然后再把镜像创建成容器，完成部署。

### 总结：

镜像：就是将应用程序及其依赖、环境、配置打包在一起。

容器：镜像运行起来就是容器，容器要形成隔离空间，有自己独立的内存、cpu、文件系统，以避免容器去修改、污染镜像，一个镜像可以运行多个容器，将来形成集群。

Docker结构：

服务端：接收客户端命令或者远程请求，操作镜像或者容器；

客户端：发送命令或者请求到Docekr服务端。

DockerHub：一个镜像托管的服务器，类似的有阿里云镜像服务，统称为DockerRegistry