**Docker的网络模式以及区别，使用场景？**

1. Bridge模式

bridge模式是默认模式，此模式会为每一个容器分配Network Namespace、 设置IP等， 并将一个主机上的Docker容器连接到一个虚拟网桥上。此模式与外界通信使用NAT协议

2. Host模式

基于宿主机的网络，它的网络跟宿主机的网络一模一样，那是因为，在创建这个容器之初、并没有对它的Net网络栈进行隔离，而是直接使用的宿主机的网络，此模式没有网络隔离性， 同时会引起网络资源的竞争与冲突。

使用场景：性能较好，但不便之处是灵活性不高，此模式没有网络隔离性，容器与宿主机出现端口冲突问题。

3. Container模式

这个模式指定新创建的容器和已经存在的一个容器共享一个Network Namespace，而不是和宿主机共享，新创建的容器不会创建自己的网卡，配置自己的IP，而是和一个指定的容器共享IP、端口范围等

使用场景：在于可以将一个应用的多个组件放在不同的容器趾， 这些容器配成container模式的网络， 这样它们可以作为一个整体对外提供服务。 同时， 这种模式也降低了容器间的隔离性。

4. None模式

什么都没有的网络，Docker容器拥有自己的Network Namespace， 但是， 并不为Docker容器进行任何网络配置。 也就是说， 这个Docker容器没有网卡、 IP、 路由等信息。 需要我们自己为Docker容器添加网卡、 配置IP等，给用户很高的自由度来配置网络环境

使用场景：此网络可以运行一些关于安全方面的验证码、效验码等服务

**简要描述AUFS文件系统的特点以及优缺点，如何解决性能问题，如何实现数据持久化（如数据库重启后数据不丢失），并在作业中体现**

AUFS 是一种联合文件系统，它把若干目录按照顺序和权限 mount 为一个目录并呈现出来，默认情况下，只有第一层（第一个目录）是可写的，其余层是只读的。

增加文件：默认情况下，新增的文件都会被放在最上面的可写层中。

删除文件：因为底下各层都是只读的，当需要删除这些层中的文件时，AUFS 使用 whiteout 机制，它的实现是通过在上层的可写的目录下建立对应的whiteout隐藏文件来实现的。

修改文件：AUFS 利用其 CoW （copy-on-write）特性来修改只读层中的文件。AUFS 工作在文件层面，因此，只要有对只读层中的文件做修改，不管修改数据的量的多少，在第一次修改时，文件都会被拷贝到可写层然后再被修改。

**优点：**

1、AUFS可以在多个运行中的容器之间有效地共享映像，从而缩短了容器的启动时间并减少了磁盘空间的使用。

2、AUFS如何在图像层和容器之间共享文件的基本机制非常有效地使用了页面缓存

**缺点：**

1、AUFS存储驱动程序可能会在容器写入时引入巨大的延迟。 这是因为容器第一次写入任何文件时，都需要找到该文件并将其复制到容器的顶层可写层中。 当这些文件存在于许多图像层下面并且文件本身很大时，这些等待时间会增加并且变得更加复杂。

Docker有两种方法实现数据持久化，一种是volumn，一种是bind mount。在这两种方法中，docker推荐使用的是第一种。

**简要描述Docker和虚拟机的对比**

安全性问题。docker目前并不能分辨具体执行指令的用户，只要一个用户拥有执行docker的权限，那么他就可以对docker的容器进行所有操作

资源隔离方面不如虚拟机，docker是利用cgroup实现资源限制的，只能限制资源消耗的最大值，而不能隔绝其他程序占用自己的资源