**注：所有相关安装包在搭建的Master节点的/opt/software目录下**

### 任务一：Hadoop 完全分布式安装配置

本环节需要使用root用户完成相关配置，安装Hadoop需要配置前置环境。命令中要求使用绝对路径，具体要求如下:

1. 将master节点JDK安装包解压到/opt/module路径中(若路径不存在，则需新建)，将JDK解压命令复制并粘贴至对应报告中;
2. 修改容器中/etc/profile文件，设置JDK环境变量并使其生效，配置完毕后在master节点分别执行“java -version”和“javac”命令，将命令行执行结果分别截图并粘贴至对应报告中;
3. 请完成host相关配置，将三个节点分别命名为master、slave1、slave2，并做免密登录，用scp命令并使用绝对路径从master复制JDK解压后的安装文件到slave1、slave2节点（若路径不存在，则需新建），并配置slave1，slave2相关环境变量，将全部与JDK的scp复制命令复制并粘贴至对应报告中;
4. 在master将Hadoop解压到/opt/module(若路径不存在，则需新建)目录下，并将解压包分发至slave1、slave2中，其中master，slave1，slave2节点均作为datanode,配置好相关环境，初始化Hadoop环境namenode，将初始化命令及初始化结果截图（截取初始化结果日志最后20行即可）粘贴至对应报告中；
5. 启动hadoop集群（包括hdfs和yarn），使用jps命令查看master节点与slave1节点的java进程，将jps命令与结果截图粘贴至对应报告中。

### 任务二：Hadoop 伪分布式安装配置

本环节需要使用root用户完成相关配置，安装Hadoop需要配置前置环境。命令中要求使用绝对路径，具体要求如下:

1. 将容器master节点JDK安装包解压到/opt/module路径中(若路径不存在，则需新建)，将JDK解压命令复制并粘贴至对应报告中;
2. 修改容器/etc/profile文件，设置JDK环境变量并使其生效，配置完毕后在Master节点分别执行“java -version”和“javac”命令，将命令行执行结果分别截图并粘贴至对应报告中;
3. 请完成host相关配置，将三个节点分别命名为master、slave1、slave2，并做免密登录，使用ssh命令，从master免密登录slave1中，将ssh免密登录命令与执行结果截图并粘贴至对应报告中;
4. 在Master将Hadoop解压到/opt/module(若路径不存在，则需新建)目录下，完成相关配置，初始化hdfs，将初始化命令文字及初始化结果截图（截取初始化结果结束倒数20行即可）粘贴至对应报告中；
5. 启动Hadoop集群（包括hdfs和yarn），使用jps命令查看master节点java进程，将查看结果截图粘贴至对应报告中。

### 任务三：Hadoop HA安装部署

本环节需要使用root用户完成相关配置，安装Hadoop需要配置前置环境。命令中要求使用绝对路径，具体要求如下:

1. 分别将Master节点Hadoop，Zookeeper,JDK安装包解压到/opt/module路径中(若路径不存在，则需新建)，其中将JDK，Hadoop解压命令复制并粘贴至对应报告中;
2. 请完成host相关配置，将三个节点分别命名为master、slave1、slave2并做免密登录，修改容器中/etc/profile文件，设置JDK环境变量并使其生效，分发jdk至slave1，slave2中，均配置完毕后在Master节点分别执行“java -version”和“javac”命令，将命令行执行结果分别截图并粘贴至对应报告中;
3. 配置好zookeeper，其中zookeeper使用集群模式，分别在master，slave1，slave2作为其集群的节点，使用zkServer.sh status获取zookeeper服务端状态，将命令和结果截图粘贴至对应报告中（注：只截取三个节点中zookeeper server角色模式为leader的节点）；
4. 配置好Hadoop HA，请将dfs.ha.namenodes.hadoopcluster设置为nn1,nn2，同时yarn.resourcemanager.ha.rm-ids设置为rm1,rm2，并在master启动nn1与rm1，在salve1启动nn2与rm2，将master，slave1，slave2均作为datanode，分发hadoop至slave1，slave2中。启动yarn与hdfs的HA集群（Hadoop HA集群），并在mster节点上使用命令分别查看服务nn2与rm2进程状态，并将查看命令及结果截图粘贴至对应报告中；
5. Hadoop HA配置并启动完毕后，使用jps在slave1节点查看服务进程，将查看命令及结果截图粘贴至对应报告中。

### 任务四：Flume安装配置

本环节需要使用root用户完成相关配置，已安装Hadoop及需要配置前置环境，具体要求如下：

1. 将Master节点Flume安装包解压到/opt/module目录下，将解压命令复制并粘贴至对应报告中；
2. 完善相关配置设置，Hive环境变量，并使环境变量生效，执行命令flume-ng version并将命令与结果截图粘贴至对应报告中；
3. 启动Flume传输Hadoop日志（namenode或datanode日志），查看HDFS中/tmp/flume目录下生成的内容，将查看命令及结果（至少5条结果）截图并粘贴至对应报告中。

### 任务五：Kafka安装配置

本环节需要使用root用户完成相关配置，已安装Hadoop及需要配置前置环境，具体要求如下：

1. 将Master节点Zookeeper，kafka安装包解压到/opt/module目录下，将kafka解压命令复制并粘贴至对应报告中；
2. 配置好zookeeper，其中zookeeper使用集群模式，分别在master，slave1，slave2作为其节点（若zookpeer已安装配置好，则无需再次配置），配置好kafka的环境变量，使用kafka-server-start.sh --version查看kafka的版本内容，并将命令和结果截图粘贴至对应报告中；
3. 完善其他配置并分发kafka文件到slave1,slave2中，并在每个节点启动Kafka，创建topic，其中toipic名称为installtopic，分区数为2，副本数为2，将创建命令和创建成果截图粘贴到对应报告中；

### 任务六：Hive安装配置

本环节需要使用root用户完成相关配置，已安装Hadoop及需要配置前置环境，具体要求如下：

1. 将Master节点Hive安装包解压到/opt/module目录下，将命令复制并粘贴至对应报告中；
2. 设置Hive环境变量，并使环境变量生效，执行命令hive --version并将命令与结果截图容复制并粘贴至对应报告中；
3. 完成相关配置并添加所依赖包，将MySQL数据库作为Hive元数据库。初始化Hive元数据，并通过schematool相关命令执行初始化，将初始化结果截图（范围为命令执行结束的最后10行）复制粘贴至对应报告中。

### 任务七：Spark on Yarn安装配置

本环节需要使用root用户完成相关配置，已安装Hadoop及需要配置前置环境，具体要求如下：

1. 将Spark包解压到路径/opt/module路径中(若路径不存在，则需新建)，将完整解压命令复制粘贴至对应报告中;；
2. 修改容器中/etc/profile文件，设置Spark环境变量并使环境变量生效，在/opt目录下运行命令spark-submit --version，将命令与结果截图粘贴至对应报告中;
3. 完成on yarn相关配置，使用spark on yarn 的模式提交$SPARK\_HOME/examples/jars/spark-examples\_2.12-3.1.1.jar 运行的主类为org.apache.spark.examples.SparkPi，将运行结果截图粘贴至对应报告中（截取Pi结果的前后各5行）。

（运行命令为：spark-submit --master yarn --class org.apache.spark.examples.SparkPi $SPARK\_HOME/examples/jars/spark-examples\_2.12-3.1.1.jar）

### 任务八：Flink on Yarn安装配置

本环节需要使用root用户完成相关配置，已安装Hadoop及需要配置前置环境，具体要求如下：

1. 将Flink包解压到路径/opt/module路径中(若路径不存在，则需新建)，将完整解压命令复制粘贴至对应报告中;
2. 修改容器中/etc/profile文件，设置Flink环境变量并使环境变量生效。在容器中/opt目录下运行命令flink --version，将命令与结果至对应报告中;
3. 开启Hadoop集群，在yarn上以per job模式（即Job分离模式，不采用Session模式）运行 $FLINK\_HOME/examples/batch/WordCount.jar，将运行结果最后10行截图粘贴至对应报告中。

示例 ：

flink run -m yarn-cluster -p 2 -yjm 2G -ytm 2G $FLINK\_HOME/examples/batch/WordCount.jar

### 任务九：HBase分布式安装配置

本环节需要使用root用户完成相关配置，安装HBase需要配置前置环境。命令中要求使用绝对路径，具体要求如下:

1. 确认是否完成Hadoop和Zookeeper的分布式（Hadoop伪分布式不可以）安装部署，没有的话请进行安装部署并启动。完成部署后在三个节点使用jps命令，并将结果截图粘贴至对应报告中；
2. 将Master节点Hbase安装包解压到/opt/module目录下，将命令复制并粘贴至对应报告中；
3. 将解压包分发至slave1、slave2中，并修改相关配置和环境。配置好环境变量，在容器中/opt目录下运行命令hbase version，将命令与结果至对应报告中;
4. 启动HBase后在各节点使用jps命令查看，并将结果截图粘贴至对应报告中；正常启动后在hbase shell中查看命名空间，将命令和结果截图粘贴至对应报告中；

### 任务十：ClickHouse单机安装配置

本环节需要使用root用户完成相关配置，具体要求如下：

1. 将Master节点ClickHouse相关安装包解压到/opt/module/clickhouse目录下（若路径不存在，则需新建），将命令复制并粘贴至对应报告中；
2. 执行启动各个相关脚本，将命令复制并粘贴至对应报告中；
3. 设置远程访问并移除默认监听文件（listen.xml），同时由于9000端口被hadoop占用，需要将clickhouse的端口更改为9001，将上述要求的配置或命令复制并粘贴至对应报告中；
4. 启动clickhouse，启动后查看clickhouse运行状态，并将命令和结果截图粘贴至对应报告中。

### 任务十一：Hudi部署

本环节需要使用root用户完成相关配置，具体要求如下：

1. 安装Hadoop分布式集群（Hadoop伪分布式不可以），在master节点安装spark on yarn。安装完成后启动hadoop集群（包括hdfs和yarn），使用jps命令查看master节点与slave1节点的java进程，将jps命令与结果截图粘贴至对应报告中；使用spark on yarn 的模式提交$SPARK\_HOME/examples/jars/spark-examples\_2.11-2.1.1.jar 运行的主类为org.apache.spark.examples.SparkPi，将运行结果截图粘贴至对应报告中（截取Pi结果的前后各5行）

（运行命令为：spark-submit --master yarn --class org.apache.spark.examples.SparkPi $SPARK\_HOME/examples/jars/spark-examples\_2.11-2.1.1.jar）；

1. 将Master节点maven相关安装包解压到/opt/module/目录下（若路径不存在，则需新建），将命令复制并粘贴至对应报告中；
2. 配置maven本地库为/opt/software/RepMaven/，将命令和配置复制并粘贴至对应报告中；配置maven的环境变量，并在/opt/下执行mvn -v，将运行结果截图粘贴至对应报告中；
3. 将Master节点hudi相关安装包解压到/opt/module/目录下（若路径不存在，则需新建），将命令复制并粘贴至对应报告中；
4. 完成解压安装及配置后使用maven对hudi进行构建（spark3.1，scala-2.12），使用spark-shell操作hudi（启动时若需要hudi-spark3.1-bundle\_2.12-0.12.0.jar，该包已放置在/opt/software下，若不需要请忽略），将spark-shell启动命令复制并粘贴至对应报告中；使用spark-shell运行下面给到的案例，并将最终查询结果截图粘贴至对应报告中。

import org.apache.hudi.QuickstartUtils.\_

import scala.collection.JavaConversions.\_

import org.apache.spark.sql.SaveMode.\_

import org.apache.hudi.DataSourceReadOptions.\_

import org.apache.hudi.DataSourceWriteOptions.\_

import org.apache.hudi.config.HoodieWriteConfig.\_

import org.apache.hudi.common.model.HoodieRecord

val tableName = "hudi\_trips\_cow"

val basePath = "file:///tmp/hudi\_trips\_cow"

val dataGen = new DataGenerator

val inserts = convertToStringList(dataGen.generateInserts(10))

val df = spark.read.json(spark.sparkContext.parallelize(inserts, 2))

df.write.format("hudi").

options(getQuickstartWriteConfigs).

option(PRECOMBINE\_FIELD\_OPT\_KEY, "ts").

option(RECORDKEY\_FIELD\_OPT\_KEY, "uuid").

option(PARTITIONPATH\_FIELD\_OPT\_KEY, "partitionpath").

option(TABLE\_NAME, tableName).

mode(Overwrite).

save(basePath)

val tripsSnapshotDF = spark.read.format("hudi").load(basePath + "/\*/\*/\*/\*")

tripsSnapshotDF.createOrReplaceTempView("hudi\_trips\_snapshot")

spark.sql("select fare, begin\_lon, begin\_lat, ts from hudi\_trips\_snapshot where fare > 20.0").show()