**druid的概念**

1. **Druid主要进程相关**

java `cat conf/druid/historical/jvm.config | xargs` -cp conf/druid/\_common:conf/druid/historical:lib/\* io.druid.cli.Main server historical

java `cat conf/druid/broker/jvm.config | xargs` -cp conf/druid/\_common:conf/druid/broker:lib/\* io.druid.cli.Main server broker

java `cat conf/druid/coordinator/jvm.config | xargs` -cp conf/druid/\_common:conf/druid/coordinator:lib/\* io.druid.cli.Main server coordinator

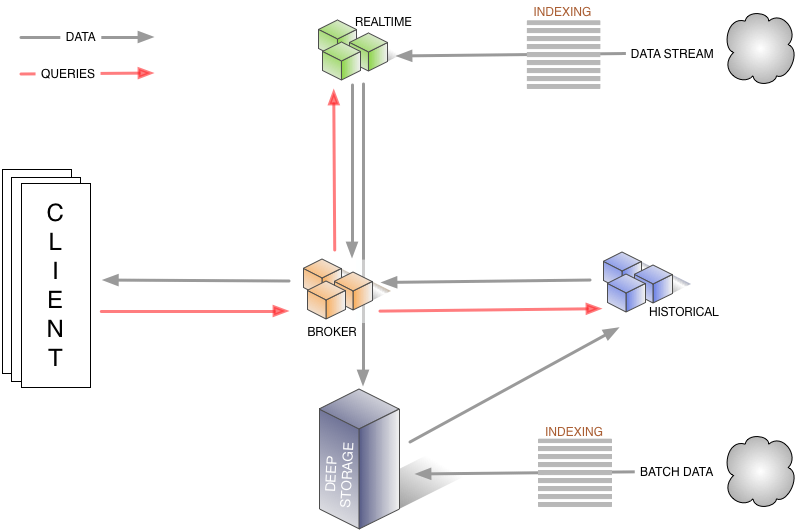
java `cat conf/druid/overlord/jvm.config | xargs` -cp conf/druid/\_common:conf/druid/overlord:lib/\* io.druid.cli.Main server overlord

java `cat conf/druid/middleManager/jvm.config | xargs` -cp conf/druid/\_common:conf/druid/middleManager:lib/\* io.druid.cli.Main server middleManager

1. **Query Service**

* **Historical**：它下载不变的segments到本地,提供segments的查询服务。
* **Broker**：负责分发查询，并收集各Historical的查询结构及对结果进行合并。
* **Coordinator**：管理集群中historical节点的segments，通知historical节点下载新的segments,删除旧的segments,以及迁移segments以达到负载均衡等。

**数据流图**



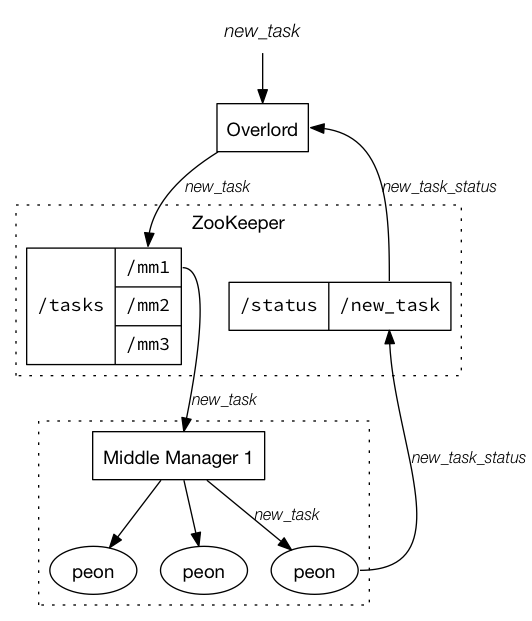
数据以流的形式【目前支持pull/push】进入实时节点，实时节点实时实时进行索引的创建，并保证实时数据可查。到达配置的事件后，将创建的索引上传至Deep Storage【一般为hdfs】，然后协调节点分配Historical节点下载数据，解压后提供历史数据查询服务，实时节点此时停止服务。

由上图可以看出：客户端发起一个查询 请求给Broker节点，Broker节点会根据根据查询条件找到对应数据存放节点【实时节点或历史节点】，然后下发查询给实时节点或者历史节点进行查询，查询返回的结果在Broker做合并。

1. **Indexing Service**

* **Overlord**：负责接收、分布式分发任务，以及围绕任务创建锁，和对任务的状态进行反馈。
* **MiddleManager**：负责提交启动具体的任务进程。
* **Peon**：MiddleManager负责创建的，专用于执行具体的任务经常。
* **Task**：Peon进程中具体的任务，包括index、append、merge、kill、hadoop\_convert\_segment、convert\_segment等。

**工作流程**



1. **Druid常用的接口**