JanusGraph的后端存储配置

JanusGraph支持多种数据存储后端，包括：Apache Cassandra、Apache HBase、Google Cloud Bigtable和Oracle Berkeley DB。本次实践中使用了Apache Cassandra和Orcale Berkeley DB Java Edition分别作为JanusGraph的存储后端进行测试。

1. JanusGraph + Apache Cassandra 集成配置。

Apache Cassandra是一个开源的、分布式、无中心、支持水平拓展、高可用的KEY-VALUE类型的NOSQL数据库[1]。根据JanusGraph的官方文档介绍，Cassandra作为JanusGraph的存储后端时有多种集成模式[2]，这里使用了Remote Server模式，Cassandra以集群方式存在， 运行在其他主机上的JanusGraph基于Socket的读/写来访问Cassandra集群。我们通过控制Cassandra集群的节点数量，运行LDBC的社交网络图Benchmark对JanusGraph的性能进行测试。测试中使用的Cassandra版本是3.11.2。

1.1 单节点的Cassandra连接配置

在本例中JanusGraph与单个节点的Cassandra进行连接，Cassandra运行在实验主机host5上，其主要配置文件conf/cassandra.yaml中的关键配置如下：

|  |
| --- |
| # 集群名称  cluster\_name**:** 'IO Cassandra Cluster'  # 设置种子节点 种子节点为host9 ip地址为192.168.5.34  seed\_provider**:**  -class\_name**:** org.apache.cassandra.locator.SimpleSeedProvider  parameters**:**  -seeds**:** "192.168.5.34"  # 设置监听地址  listen\_address**:** 192.168.5.34  # 启用thrift rpc server，否则JanusGraph无法与Cassandra建立连接  start\_rpc**:** true  rpc\_address**:** 192.168.5.34 |

JanusGraph的相关配置文件（主要是gremlin-server.sh加载的配置文件和运行benchmark的程序加载的配置文件）中添加如下配置：

|  |
| --- |
| storage.backend**=**cassandrathrift  storage.hostname**=**192.168.5.34 |

1.2 两节点的Cassandra连接配置

在本例中我们部署了两个节点的Cassandra集群，使用host9作为集群的seed节点，host4作为普通节点，每个节点的Cassandra配置同上例中的单节点配置类似：seeds设置为host9的ip地址192.168.5.34，listen\_address和rpc\_address设置为各自对应的ip地址，启动时先启动seed节点上的Cassandra进程，之后普通节点启动会自动加入到集群中。集群启动后使用Cassandra提供的nodetool工具查看各节点的状态，如图1.1所示，我们向JanusGraph导入了约100M的数据，这些数据被分配存储在了Cassandra集群的节点中。

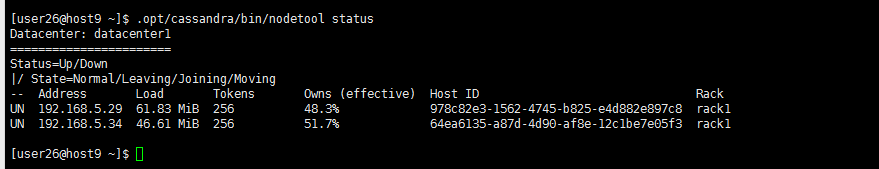


图 1. 1 两节点Cassandra集群状态

1.3 四节点的Cassandra连接配置

为了进一步验证Cassandra集群节点数量对测试性能的影响，在本例中我们使用host5、host7、host8、host10部署在四个结点的Cassandra集群，seed结点为host5和host7，host8和host10作为普通节点加入集群。由于在这些机器上有其他用户运行的Cassandra进程造成了端口占用，除了前面例子中的配置外还需要修改Cassandra的端口，cassandra.yaml中关键配置如下：

|  |
| --- |
| # 修改节点通信端口 默认为7000，ssl默认端口为7001  storage\_port**:** 7200  ssl\_storage\_port**:** 7201  # 修改CQL客户端通信端口 默认为9042  native\_transport\_port**:** 9052  # 修改rpc端口 即JanusGraph与Cassandra的通信端口 默认端口为9160  rpc\_port**:** 9260 |

Cassandra通过JMX监测结点，所以需要修改conf/cassandra-env.sh中相关的端口：

|  |
| --- |
| # 修改 JMX 连接端口 默认为9042  JMX\_PORT**=**"8012" |

除此之外，JanusGraph相关的配置中除了指定存储后端Cassandra的地址，还要指定端口：

|  |
| --- |
| # 修改后端通信端口  storage.port=9260 |

做出如上修改后，就可以分别启动Cassandra集群和JanusGraph的gremlin-server进行连接了。Cassandra集群按照seed节点和普通节点的顺序依次启动，启动完成后使用nodetool查看集群节点状态，如图1.2所示，同上例中一样，导入JanusGraph的数据被分配到了所有的节点进行存储。

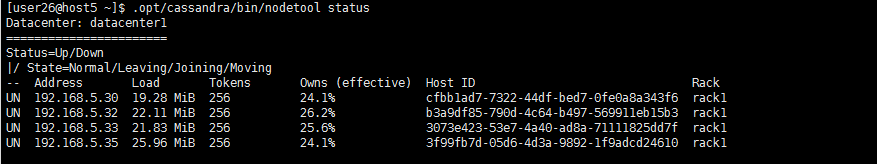


图 1. 2 四节点Cassandra集群状态

2. JanusGraph + Orcale Berkeley DB Java Edition 集成配置。

Berkeley DB是一个是一个开源的文件数据库，介于关系数据库与内存数据库之间，使用方式与内存数据库类似，它提供的是一系列直接访问数据库的函数，而不是像关系数据库那样需要网络通讯、SQL解析等步骤[3]。

JanusGraph使用的相应存储后端为Orcale Berkeley DB Java Edition，它与JanusGraph运行在同一个机器的JVM中，导入JanusGraph的图数据全部都会保存在本地磁盘上。这就将图数据的大小限制在了硬盘空间和内存空间的范围内(大约10-100万个顶点的图)。但是，对于这种规模的图，使用Berkeley DB作为存储后端具有高于其他分布式数据库的性能。

JanusGraph使用Berkeley DB作为存储后端集成的方法比较简单，无需安装Berkeley DB，Orcale Berkeley DB Java Edition 以jar包的形式存在于JanusGraph的lib目录下，所以只需在JanusGraph相关的配置文件（gremlin-server和LDBC Benchmark）中指定后端名称和存储路径即可：

|  |
| --- |
| # 指定后端名称为berkeleyje  storage.backend**=**berkeleyje  # 指定berkeley db 的数据存储路径  storage.hostname**=**../data/graph |

参考文献

[1] Apache Cassandra Homepage [OL] http://cassandra.apache.org/

[2] JanusGraph Documentation> Chapter 16.Apache Cassandra [OL] https://docs.janusgraph.org/latest/cassandra.html

[3] Oracle Berkeley DB Java Edition Homepage [OL] http://http://www.oracle.com/technetwork/database/berkeleydb/overview/index-093405.html