Redis 集成

Storm/Trident 集成 Redis

Storm-redis 用 Jedis 作为 Redis 客户端。

用法

怎样使用它？

作为 maven 依赖来使用：

```
<dependency>
    <groupId>org.apache.storm</groupId>
    <artifactId>storm-redis</artifactId>
    <version>${storm.version}</version>
    <type>jar</type>
</dependency>
```

对于标准的 Bolt

Storm-redis 提供基本的 Bolt 实现，RedisLookupBolt 和 RedisStoreBolt。

名字也就表明了它的用法，RedisLookupBolt 用 key 从 Redis 中查找值，RedisStoreBolt 存储 key / value 到 Redis。一个 Tuple 匹配 一个 key / value 对，你可以在 TupleMapper 定义匹配模式。

你可以从 RedisDataTypeDescription 选择使用的数据类型。请参照 RedisDataTypeDescription.RedisDataType 查看支持什么数据类型。在一些数据类型中（ hash and sorted set ），将 tuple 转化为元素需要额外的键和转换键。

这些接口与 RedisLookupMapper 和 RedisStoreMapper 相结合，他们分布对应 RedisLookupBolt 和 RedisStoreBolt。

RedisLookupBolt 实例

```
class WordCountRedisLookupMapper implements RedisLookupMapper {
    private RedisDataTypeDescription description;
    private final String hashKey = "wordCount";

    public WordCountRedisLookupMapper() {
        description = new RedisDataTypeDescription(
                RedisDataTypeDescription.RedisDataType.HASH, hashKey);
    }

    @Override
    public List<Values> toTuple(ITuple input, Object value) {
        String member = getKeyFromTuple(input);
        List<Values> values = Lists.newArrayList();
        values.add(new Values(member, value));
        return values;
    }

    @Override
    public void declareOutputFields(OutputFieldsDeclarer declarer) {
        declarer.declare(new Fields("wordName", "count"));
    }

    @Override
    public RedisDataTypeDescription getDataTypeDescription() {
        return description;
    }

    @Override
    public String getKeyFromTuple(ITuple tuple) {
        return tuple.getStringByField("word");
    }

    @Override
    public String getValueFromTuple(ITuple tuple) {
        return null;
    }
}
```

```
JedisPoolConfig poolConfig = new JedisPoolConfig.Builder()
        .setHost(host).setPort(port).build();
RedisLookupMapper lookupMapper = new WordCountRedisLookupMapper();
RedisLookupBolt lookupBolt = new RedisLookupBolt(poolConfig, lookupMapper);
```

RedisStoreBolt 实例

```
class WordCountStoreMapper implements RedisStoreMapper {
    private RedisDataTypeDescription description;
    private final String hashKey = "wordCount";

    public WordCountStoreMapper() {
        description = new RedisDataTypeDescription(
            RedisDataTypeDescription.RedisDataType.HASH, hashKey);
    }

    @Override
    public RedisDataTypeDescription getDataTypeDescription() {
        return description;
    }

    @Override
    public String getKeyFromTuple(ITuple tuple) {
        return tuple.getStringByField("word");
    }

    @Override
    public String getValueFromTuple(ITuple tuple) {
        return tuple.getStringByField("count");
    }
}
```

```
JedisPoolConfig poolConfig = new JedisPoolConfig.Builder()
                .setHost(host).setPort(port).build();
RedisStoreMapper storeMapper = new WordCountStoreMapper();
RedisStoreBolt storeBolt = new RedisStoreBolt(poolConfig, storeMapper);
```

对于不标准的 Bolt

如果 RedisStoreBolt 和 RedisLookupBolt 不适用于你的场景 ，storm-redis 也提供了 AbstractRedisBolt 让你扩展并且适用于你的业务逻辑。

```
public static class LookupWordTotalCountBolt extends AbstractRedisBolt {
        private static final Logger LOG =
LoggerFactory.getLogger(LookupWordTotalCountBolt.class);
        private static final Random RANDOM = new Random();

        public LookupWordTotalCountBolt(JedisPoolConfig config) {
            super(config);
        }

        public LookupWordTotalCountBolt(JedisClusterConfig config) {
            super(config);
        }

        @Override
        public void execute(Tuple input) {
            JedisCommands jedisCommands = null;
            try {
                jedisCommands = getInstance();
                String wordName = input.getStringByField("word");
                String countStr = jedisCommands.get(wordName);
                if (countStr != null) {
                    int count = Integer.parseInt(countStr);
                    this.collector.emit(new Values(wordName, count));

                    // print lookup result with low probability
                    if(RANDOM.nextInt(1000) > 995) {
                        LOG.info("Lookup result - word : " + wordName + " /
count : " + count);
                    }
                } else {
                    // skip
                    LOG.warn("Word not found in Redis - word : " +
wordName);
                }
            } finally {
                if (jedisCommands != null) {
                    returnInstance(jedisCommands);
                }
                this.collector.ack(input);
            }
        }

        @Override
        public void declareOutputFields(OutputFieldsDeclarer declarer) {
            // wordName, count
            declarer.declare(new Fields("wordName", "count"));
        }
    }
```

Trident State 的用法

1.RedisState 和 RedisMapState 为单个 Redis 提供 Jedis 接口。

2.RedisClusterState 和 RedisClusterMapState 为 Redis 集群 提供 JedisCluster 接口。

```
RedisState
java
JedisPoolConfig poolConfig = new JedisPoolConfig.Builder()
.setHost(redisHost).setPort(redisPort) .build();
RedisStoreMapper storeMapper = new WordCountStoreMapper();
RedisLookupMapper lookupMapper = new WordCountLookupMapper();
RedisState.Factory factory = new RedisState.Factory(poolConfig);
```

```
TridentTopology topology = new TridentTopology();
    Stream stream = topology.newStream("spout1", spout);

    stream.partitionPersist(factory,
                            fields,
                            new
RedisStateUpdater(storeMapper).withExpire(86400000),
                            new Fields());

    TridentState state = topology.newStaticState(factory);
    stream = stream.stateQuery(state, new Fields("word"),
                            new RedisStateQuerier(lookupMapper),
                            new Fields("columnName","columnValue"));
```

```
RedisClusterState
java
        Set<InetSocketAddress> nodes = new HashSet<InetSocketAddress>();
        for (String hostPort : redisHostPort.split(",")) {
            String[] host_port = hostPort.split(":");
            nodes.add(new InetSocketAddress(host_port[0],
Integer.valueOf(host_port[1])));
        }
        JedisClusterConfig clusterConfig = new
JedisClusterConfig.Builder().setNodes(nodes)
                                        .build();
        RedisStoreMapper storeMapper = new WordCountStoreMapper();
        RedisLookupMapper lookupMapper = new WordCountLookupMapper();
        RedisClusterState.Factory factory = new
RedisClusterState.Factory(clusterConfig);

        TridentTopology topology = new TridentTopology();
        Stream stream = topology.newStream("spout1", spout);

        stream.partitionPersist(factory,
                                fields,
                                new
RedisClusterStateUpdater(storeMapper).withExpire(86400000),
                                new Fields());

        TridentState state = topology.newStaticState(factory);
        stream = stream.stateQuery(state, new Fields("word"),
                                new RedisClusterStateQuerier(lookupMapper),
                                new Fields("columnName","columnValue"));
```

授权