# **Spring Boot 集成Cassandra(自己写的，完全可用)**



[好想大吃一顿](https://blog.csdn.net/qq_33067315" \t "https://blog.csdn.net/qq_33067315/article/details/_blank) 2020-05-12 18:03:19 IMG_257 1051 IMG_258 收藏 2

版权

SpringBoot中使用cassandra可以通过以下方式：

1.使用spring boot starer 依赖spring的自动注入

2.继承AbstractCassandraConfiguration

3.自己去构建cassandra的session

本文通过继承AbstractCassandraConfiguration的方式去访问cassandra。其他的方式，可以通过spring boot的官方文档进行学习。

基础支撑：

* pom文件里的依赖(cassandra 支持3.0版本)

<dependency>

<groupId>org.springframework.data</groupId>

<artifactId>spring-data-cassandra</artifactId>

*<!--<version>1.5.4.RELEASE</version>-->*

<version>3.0.0.RELEASE</version>

</dependency>

* Cassandra连接的配置(application.yml)

spring:

data:

cassandra:

keyspace-name: *//键空间的名字*

contact-points: *//ip*

username: *//你设置的用户名*

password: *//你设置的密码*

session-name: Test Cluster *//默认的集群名字*

local-datacenter: datacenter1*//默认的数据中心*

* Cassandra Java类的配置

@Configuration

public class CassandraConfig extends AbstractCassandraConfiguration {

*//空间名称*

@Value("${spring.data.cassandra.keyspace-name}")

private String keyspaceName;

*//节点IP（连接的集群节点IP）*

@Value("${spring.data.cassandra.contact-points}")

private String contactPoints;

@Value("${spring.data.cassandra.username}")

private String username;

@Value("${spring.data.cassandra.password}")

private String password;

@Value("${spring.data.cassandra.session-name}")

private String sessionName;

public String getKeyspaceName() {

return keyspaceName;

}

public String getContactPoints() {

return contactPoints;

}

@Override

public String getSessionName() {

return sessionName;

}

@Override

public String getLocalDataCenter() {

return "datacenter1";

}

@Override

public CqlSessionFactoryBean cassandraSession() {

CqlSessionFactoryBean cqlSessionFactoryBean = super.cassandraSession();

cqlSessionFactoryBean.setPassword(password);

cqlSessionFactoryBean.setUsername(username);

return cqlSessionFactoryBean;

}

}

* 实体类的配置 多个分区键的配置

@Setter

@Getter

@Table(value = "")

public class MasterOrderDO {

*/\*\**

*\*数据库的结构 primary key((tenant\_id,sequence\_id),order\_id);*

*\**

*\*\*/*

*//分区键*

@PrimaryKeyColumn(type = PrimaryKeyType.PARTITIONED, ordinal = 0, name = "tenant\_id")

String tenantId;

@PrimaryKeyColumn(type = PrimaryKeyType.PARTITIONED, ordinal = 1, name = "sequence\_id")

String sequenceId;

*//集群键*

@PrimaryKeyColumn(type = PrimaryKeyType.CLUSTERED, ordinal = 2, name = "order\_id")

Long orderId;

@Column(value = "order\_code")

String orderCode;

}

使用CassandraTemplate 进行一些sql的编写。下面是写简单的例子。

@Service

public class TestImpl {

@Resource

private CassandraTemplate cassandraTemplate;

*//自定义sql语句*

public MasterOrderDO findByTenantIdAndSequenceId(String tenantId,String sequenceId) {

String cql=String.format("select \* from master\_order where tenant\_id = '%s' and sequence\_id='%s'",tenantId,sequenceId);

MasterOrderDO masterOrderDO = cassandraTemplate.selectOne(cql, MasterOrderDO.class);

return masterOrderDO;

}

public void test(MasterOrderDO masterOrderDO) {

*//插入*

cassandraTemplate.insert(masterOrderDO);

*//删除*

cassandraTemplate.delete(masterOrderDO);

}

}

注意：

1.在官方文档中 不推荐利用Cassandra进行分页查询。

Cassandra存储库不会扩展PagingAndSortingRepository，因为使用限制/偏移量的经典分页模式不适用于Cassandra。