**Spring data Gremlin**

The Spring Data Gremlin Starter provides Spring Data support for the Gremlin query language from Apache, which developers can use with any Gremlin-compatible data store.

Spring Data Gremlin helps to interact with Azure Cosmos DB Graph API.

**Add the dependency**

spring-data-gremlin is present in Maven Central Repository.

<dependency>

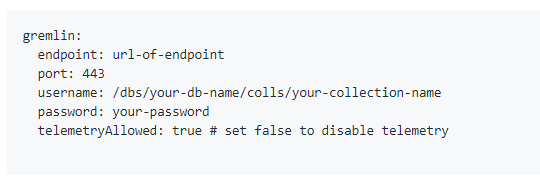
<groupId>com.microsoft.spring.data.gremlin</groupId>

<artifactId>spring-data-gremlin</artifactId>

<version>2.1.7</version>

</dependency>

* **Application.yml**

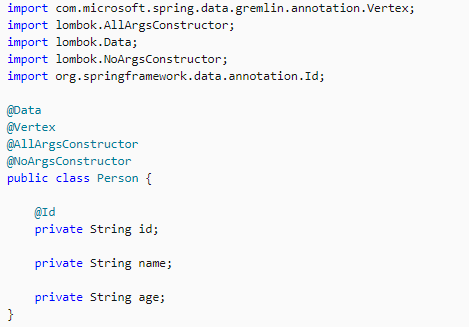


**Features:**

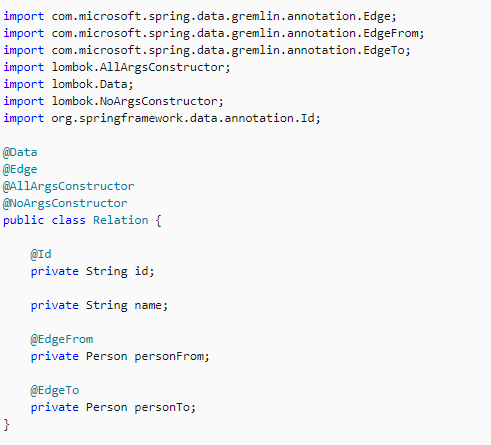
* Spring Data CRUDRepository basic CRUD functionality
  + save
  + findAll
  + findById
  + deleteAll
  + deleteById
* Spring Data [@Id](https://github.com/spring-projects/spring-data-commons/blob/db62390de90c93a78743c97cc2cc9ccd964994a5/src/main/java/org/springframework/data/annotation/Id.java) annotation. There're 2 ways to map a field in domain class to id field of a database entity.
  + annotate a field in domain class with @Id
  + set name of this field to id
* Default annotaion
  + @Vertex maps an Object to a Vertex
  + @VertexSet maps a set of Vertex
  + @Edge maps an Object to an Edge
  + @EdgeSet maps to a set of Edge
  + @EdgeFrom maps to the head Vertex of an Edge
  + @EdgeTo maps to the tail Vertex of an Edge
  + @Graph maps to an Object to a Graph
* Supports advanced operations
  + <T> T findVertexById(Object id, Class<T> domainClass);
  + <T> T findEdgeById(Object id, Class<T> domainClass);
  + <T> boolean isEmptyGraph(T object)
  + long vertexCount()
  + long edgeCount()

### Define an entity

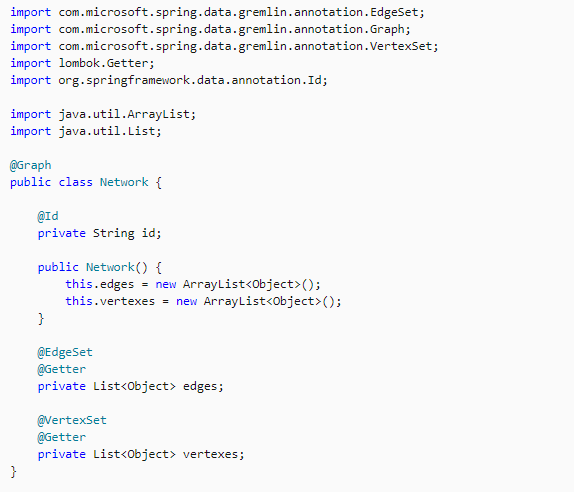
**Define a simple Vertex entity with @Vertex.**



**Define a simple Edge entity with @Edge.**



**Define a simple Graph entity with @Graph.**

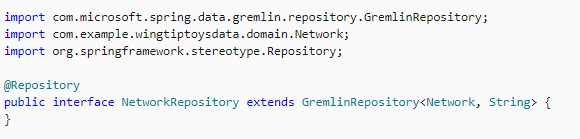


### Create repositories

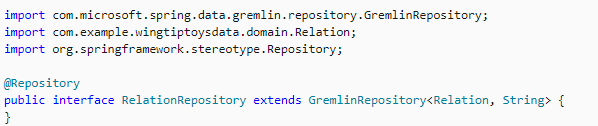
### Need to create three different respositories for three different Entities

### Create respository that extends to GremlinRepository

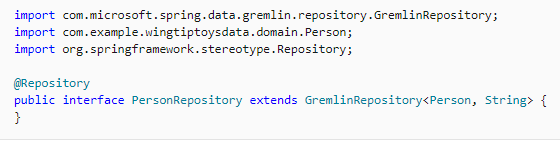
**NetworkRepository.java**



**RelationRepository.java**



**PersonRepository.java**



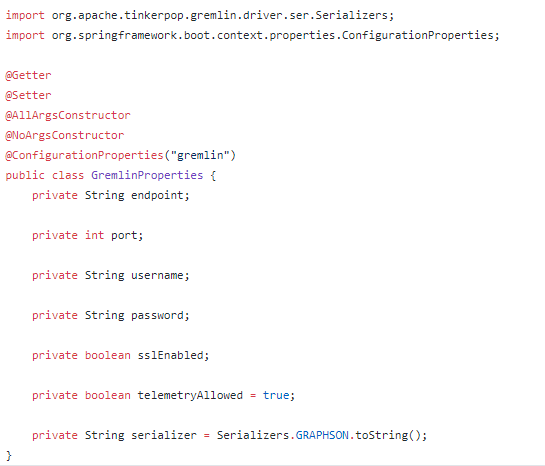
**In the Config Directory:**

Add two files GremlinProperties.java and UserRepositoryConfiguration.java



Use **@EnableGremlinRepositories** to enable sync repository support and all the other are similar to that of CosmosDB.

**GremlinProperties.java**



**endpoint**: Specifies the Gremlin URI for your database, which is derived from the unique **ID** that you specified when you created your Azure Cosmos DB

**port**: Specifies the TCP/IP port, which should be **443** for HTTPS.

**username**: Specifies the unique **Database id** and **Graph id that was given during the graph creation in Cosmos.**

**password:** Specifies either the primary or secondary **Access key**

**telemetryAllowed:** Specify **true** if you want to enable telemetry; otherwise, **false**.