DATASTAX ACADEMY

Exercise 5.1 – Object Mapping and Annotations

This exercise provides you with hands on experience using object mapping annotations.

In this exercise, you will:

- Retrieve `Video` instances from the database using the `Mapper` class.

Step 1:

1. Navigate to the following file in the IDE: *~/session3/src/test/java/com/datastax/training/killrvideo/dao/cassandra/MapperVideoDAOTest.java*

2. This is a simple class that adds a video and then uses the soon-to-be implemented `getVideo()` method.

Step 2:

1. Navigate to the following file in the IDE: *~/session3/src/main/java/com/datastax/training/killrvideo/model/Video.java*

2. Add the appropriate annotations to the `Video` class to make it compatible with the `Mapper`.

```
package com.datastax.training.killrvideo.model;

import com.datastax.driver.mapping.annotations.*;

import java.nio.ByteBuffer;
import java.util.*;

@Table(name = "videos", readConsistency = "LOCAL_ONE", writeConsistency =
"LOCAL_QUORUM")
public class Video {

    @PartitionKey
    @Column(name = "video_id")
    private UUID videoId;

    @Column(name = "description")
    private String description;
    @Column(name = "title")
    private String title;
    @Column(name = "type")
    private String type;
    @Column(name = "url")
    private String url;
    @Column(name = "release_date")
    private Date releaseDate;
    @Column(name = "release_year")
    private int releaseYear;
    @Column(name = "avg_rating")
    private float avgRating;
    @Column(name = "mpaa_rating")
    private String mpaaRating;
    @Column(name = "tags")
    private Set<String> tags;
    @Column(name = "preview_thumbnail")
    private ByteBuffer previewThumbnail;
    @Column(name = "genres")
    private Set<String> genres;
    @Column(name = "user_id")
    private UUID userId;

    // gets/sets etc.
}
```

Step 3:

1. Navigate to the following file within the IDE: *~/session3/src/main/java/com/datastax/training/killrvideo/model/dao/cassandra/CassandraVideoDAO.java*

2. Scroll down to the `getVideo()` function. Notice it currently only returns *null*.

3. Change this function to use the `Mapper` instance named `mapper` found in the base class `AbstractMapperDAO`. `AbstractMapperDAO` has already created the `mapper` instance for you.

```
@Override
public Video getVideo(UUID video_id) {
    return mapper.get(video_id);
}
```

Step 4:

1. Note the simplicity of the `getVideo()` function.

2. Go back to *MapperVideoDAOTest.java*. Put a breakpoint on the *assertEquals* line and debug `testMapperGetVideo()`. Then analyze the *retreivedVideo* object, which proves the mapper functioned properly.