

# Documentum REST Extensibility Tutorial (6): Make Resource Query-able

---

**Previous:** [Documentum REST Extensibility Tutorial \(5\): Update and Delete Object](#)

## Preliminary

Before diving into this tutorial, you must at least complete the tutorial [Documentum REST Extensibility Tutorial \(2\): Create A Sample Project](#).

## Overview

Documentum REST Services has a **DQL Query Resource** that allows users to execute read-only DQL. When you implement some new resources (e.g. **AliasSet resource**) using Documentum REST Extensibility, you may want the DQL query result on the specific data type (e.g. **dm\_alias\_set**) to give links to your new resources. This tutorial talks about how to make your new resources query-able from DQL Query resource.

If you have read [Documentum REST Extensibility Tutorial \(3\): Explore The Sample Project](#), there is a Documentum REST annotation `@com.emc.documentum.rest.view.annotation.ResourceViewBinding` introduced to bind resource views. This annotation will help us to link the new resource to DQL Query Resource.

This feature works for new services that are designed for specific object types.

## Implementation

In the sample project location `<SDK_ROOT>/maven-kit/generated/acme-rest/acme-rest-resource/src/main/java/com/acme/rest/controller/`, there is a controller `AliasSetController.java` designed for CRUD operations on the `dm_alias_set` data type. This controller meets the requirement that the resource is designed for a specific data type.

When annotating `@ResourceViewBinding` on the class level, we can add a parameter `queryTypes` in the annotation attributes.

```
@Controller("acme#alias-set") @RequestMapping("/repositories/{repositoryName}/alias-sets/{aliasSetId}") @ResourceViewBinding
```

```
queryTypes = "dm_alias_set"
```

This setting means, when `dm_alias_set` is queried in *DQL Query Resource*, the query result item will present a link relation pointing to this annotated resource `acme#alias-set`.

## Deploy and Test

Run Maven goal `mvn clean install` to rebuild the project and redeploy it to the web container again.

Open a REST client debugging tool like PostMan or DHC to test the result.

*Request - DQL query for dm\_alias\_set:*

Execute DQL query `select * from dm_alias_set where owner_name=USER`

GET `http://localhost:8080/acme-rest/repositories/REPO?dql=select%20*%20from%20dm_alias_set%20where%20owner_name=USER`

*Response - DQL query for dm\_alias\_set:*

HTTP/1.1 200 OK Content-Type: application/vnd.emc.documentum+json;charset=UTF-8 { "id":"http://localhost:8080/acme-rest/

*Request - Get alias set resource for 660000058000515:*

Follow link relation *edit* or *self* on the specific query result entry, and make a GET method on the href.

GET http://localhost:8080/acme-rest/repositories/REPO/alias-sets/660000058000515 HTTP/1.1 Authorization: Basic ZG1hZG1pbj

*Response - Get alias set resource for 660000058000515:*

HTTP/1.1 200 OK Content-Type: application/vnd.emc.documentum+json;charset=UTF-8 Content-Length: 725 { "type":"dm\_ali

**Next:** [Documentum REST Extensibility Tutorial \(7\): Make Resource Batch-able](#)

[Learn more about Documentum REST Services >>](#)