ORACLE Rest Data Services API

API Overview

Oracle REST Data Services (ORDS) empowers users to manage and monitor Oracle Database operations through an intuitive REST API. Depending on the database's version and configuration, ORDS offers various services, including managing pluggable databases, data export capabilities, and database performance reviews.

Getting Started

- **Base URL:** https://localhost:8080/ords/hr/_/db-api/stable/database/db_links/SCOTT,MY_LINK
- **Resource Endpoints:** /database/db_links/{owner},{db_link}
- **HTTP Method:** To call the Get Partition API, Use **GET** requests

Authorization / Authentication

Database REST APIs uses a certificate authority (CA) certificate, issued by Verisign, to enable clients to connect securely to the server.

Note: Ensure that you have the appropriate log-in credentials for creating, managing, and deleting Database REST API instances.

You access the Database REST API resources over HTTPS and must provide the following information for authentication:

- An SSL certificate authority (CA) certificate file or bundle to authenticate against the Verisign CA certificate.
- User name and password for your Database REST API account.
- Custom header,X-ID-TENANT-NAME, to identify the identity domain ID.

1. **Authentication for Data Dictionary, Environment, General, Monitoring and Performance**
   For using Data Dictionary, Environment, General, Monitoring and Performance REST APIs, perform the following steps:
   1. REST enable a SCHEMA.
   2. Grant DBA, PDB_DBA roles to the schema user.
   3. Provide the database user name and password to authenticate with your cURL requests.

2. **Authentication for Pluggable Database Lifecycle Management**
   You cannot have an ORDS enabled schema in a container database. To perform the PDB lifecycle management operations, the default CDB administrator credentials, db.cdb.adminUser and db.cdb.adminUser.password must be defined in the connection pool. In this case, it is not required to specify a user schema in the URI.

Pricing

Oracle REST Data Services (ORDS) is included at no additional cost with Oracle Database deployments, including Autonomous Database, where it is available as part of the service without separate pricing. Usage falls under the underlying database licensing or cloud subscription model, such as Autonomous Transaction Processing or Data Warehouse pay-per-use compute and storage.

Prerequisites

Prerequisite	Description
ORDS Installation	ORDS must be installed and configured on a supported Oracle Database (version 19c or later recommended).
Database Account Privileges	Users need a database account with ORDS privileges like ORDS_DB_API_PRIVILEGE or specific roles (e.g., DBA role for db_links endpoint).
Module Enablement	Enable the database API module via the ORDS administration interface.
Security	Ensure HTTPS/TLS is configured for secure communication in production environments.

Status Codes

HTTP Status Code	Description
200 OK	The request was successfully completed. A 200 status is returned for a successful GET or POST method.
201 Created	The request has been fulfilled and resulted in a new resource being created. The response includes a Location header containing the canonical URI for the newly created resource.

	A 201 status is returned from a synchronous resource creation or an asynchronous resource creation that completed before the response was returned.
202 Accepted	The request has been accepted for processing, but the processing has not been completed. The request may or may not eventually be acted upon, as it may be disallowed at the time processing actually takes place. When specifying an asynchronous (__detached=true) resource creation (for example, when deploying an application), or update (for example, when redeploying an application), a 202 is returned if the operation is still in progress. If __detached=false, a 202 may be returned if the underlying operation does not complete in a reasonable amount of time. The response contains a Location header of a job resource that the client should poll to determine when the job has finished. Also, returns an entity that contains the current state of the job.
400 Bad Request	The request could not be processed because it contains missing or invalid information (such as, a validation error on an input field, a missing required value, and so on).
401 Unauthorized	The request is not authorized. The authentication credentials included with this request are missing or invalid.
403 Forbidden	The user cannot be authenticated. The user does not have authorization to perform this request.
404 Not Found	The request includes a resource URI that does not exist.
405 Method Not Allowed	The HTTP verb specified in the request (DELETE, GET, POST, PUT) is not supported for this request URI.
406 Not Acceptable	The resource identified by this request is not capable of generating a representation corresponding to one of the media types in the Accept header of the request. For

	example, the client's Accept header request XML be returned, but the resource can only return JSON.
415 Not Acceptable	The client's ContentType header is not correct (for example, the client attempts to send the request in XML, but the resource can only accept JSON).
500 Internal Server Error	The server encountered an unexpected condition that prevented it from fulfilling the request.
503 Service Unavailable	The server is unable to handle the request due to temporary overloading or maintenance of the server. The <ProductName> REST web application is not currently running.

Throttling Limits / Rate limits

ORDS enforces configurable rate limits via the restEnabledSql service or global pool settings, typically defaulting to 100 requests per minute per user/IP to prevent abuse. Database API endpoints like db_links inherit these, adjustable in ORDS configuration files (e.g., defaults.xml with maxPoolSize and jdbc.MaxLimit). No fixed public rate limits; monitor via ORDS logs and metrics endpoints.

API Reference Document

Resource

/database/db_links/{owner},{db_link}

Introduction

This endpoint retrieves detailed information about a specific database link in an Oracle Database, including its owner, name, and connection attributes, via the ORDS Database API. It supports database management tasks through a RESTful interface.

Headers

Header	Value/Example	Description
Authorization	Basic <base64(user: pass)> or OAuth2 Bearer token	Authentication for ORDS access.
Accept	application/json	Specifies JSON response format.

HTTP Method

GET

Base URL

https://{host}:{port}/ords/{schema-alias}/_/db-api/stable/

Example: https://localhost:8080/ords/hr/_/db-api/stable/database/db_links/SCOTT,MY_LINK

Endpoint

/database/db_links/{owner},{db_link}/

Request URL

https://{host}:{port}/ords/{schema-alias}/_/db-api/stable/database/db_links/{owner},{db_link}

Example: https://localhost:8080/ords/hr/_/db-api/stable/database/db_links/SCOTT,MY_LINK

Response Parameter Table

Parameter	Child Parameter	Data Type	Value	Description
Owner	-	String	“SCOOT”	Owner of the database link.
Db_link	-	String	“MY_LINK	Name of the database link.
host	-	String	"remotehost.example.com"	Host for the remote database connection.
created	-	String	"2025-01-15T10:00:00Z"	Timestamp when the link was created.
links	rel, href	Array/object	{"rel": "self", "href": "..."}	HAL-style navigation links.
username	-	string	"REMOTE_USER"	Username used for the remote connection.

Sample Response

{

"owner": "SCOTT",

"db_link": "MY_LINK",

"host": "remotehost.example.com",

"username": "REMOTE_USER",

"created": "2025-01-15T10:00:00Z",

"links": [

```
   {
     "rel": "self",
     "href": "https://localhost:8080/ords/hr/_/db-
api/stable/database/db_links/SCOTT,MY_LINK"
   },
   {
     "rel": "collection",
     "href": "https://localhost:8080/ords/hr/_/db-api/stable/database/db_links/"
   }
 ]
}
```