Oracle REST Data Services Architecture, Features, & Configuration

Jeff Smith

Distinguished Product Manager
Jeff.D.Smith@oracle.com
@thatjeffsmith

Marcel Boermann-Pfeifer

Solution Architect
marcel.pfeifer@oracle.com

Agenda

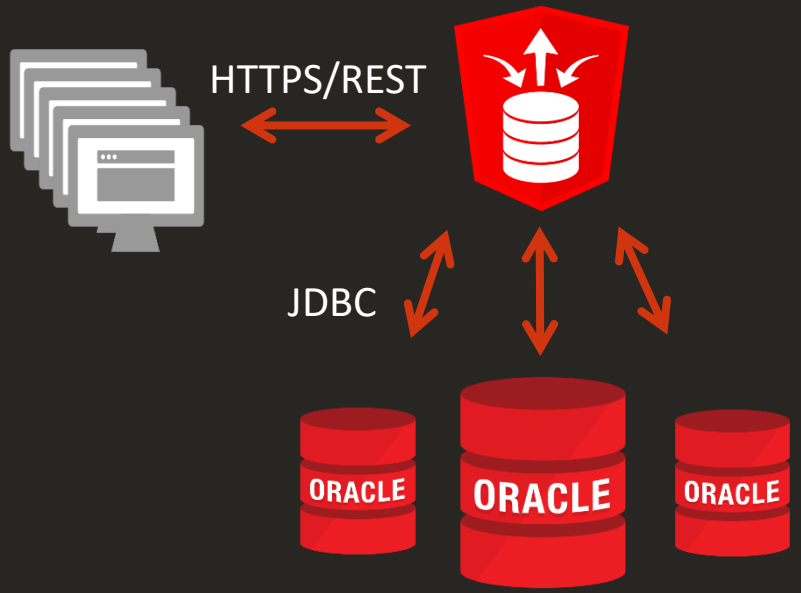
- Introduction to ORDS
- Installation & Configuration
- Features & Demonstration
- Get it running in SVi environment

REST Access to your Oracle Database

FREE

- Oracle REST Data Services (ORDS)
- Auto REST enable your data
  tables, views, SQL, PL/SQL
- Auto paged results
- Auto JSON responses
- Complete read-write functionality
- Full dev support (GUI, CLI, API)
- Oracle Cloud and On-Premises

https://www.oracle.com/REST



Introduction

Use cases, supported environments, and licensing

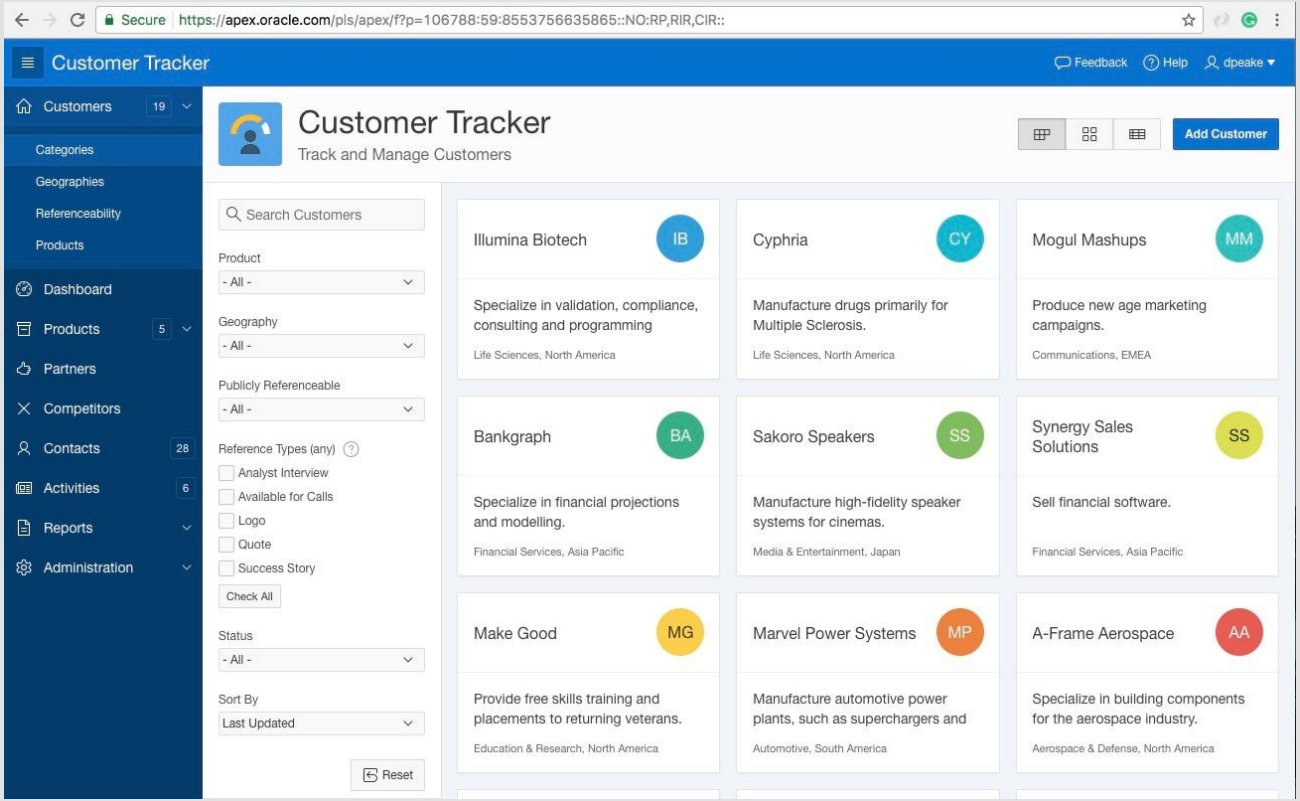
Do you need a…?

- Web listener for your Oracle PL/SQL programs or APEX
- RESTful Services harness for your Oracle Database
- Web client for working with your Oracle Database
- Database management REST API
- Mongo style API for Oracle Database (SODA)

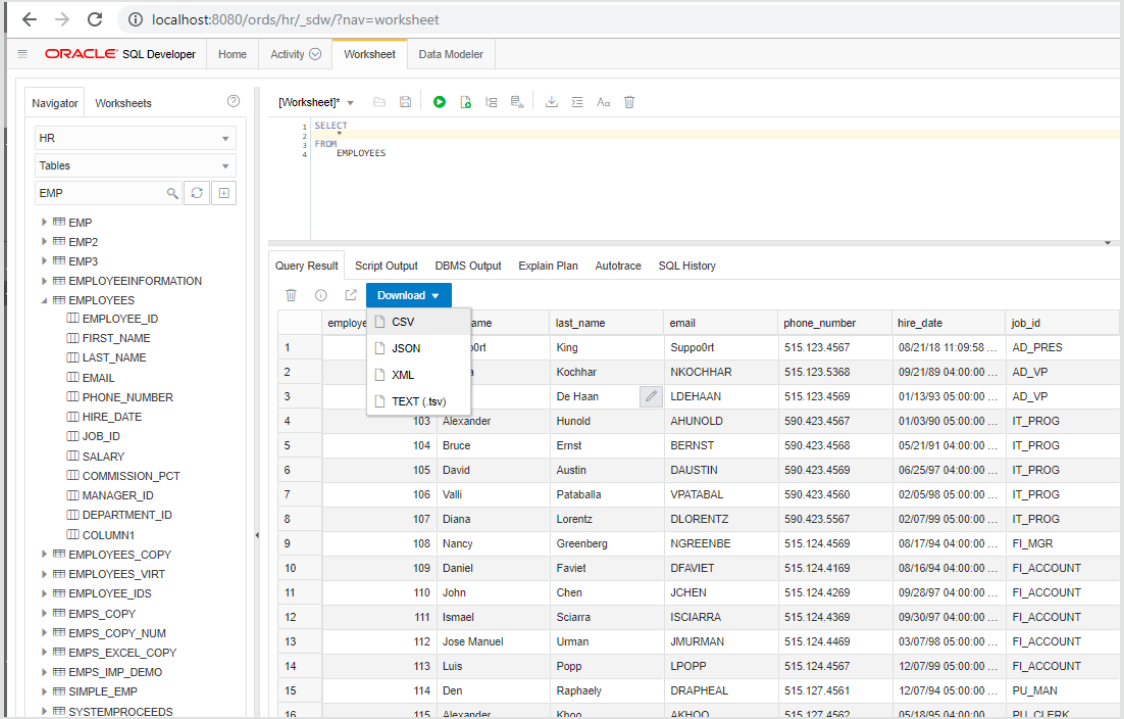
Example: EMPS of DEPT 50

- HTTPS vs SQL
- in {json}
- Page the results
- I need links for each entry
- Handle the data types for me

Example: PL/SQL web listener



Example: Browser based DB client



Example: Database Management REST API

```
POST http://localhost:8080/ords/hr/_/db-api/latest/database/datapump/export
```

POST BODY

```
{
  "datapump_dir": "DATA_PUMP_DIR",
  "filter": "HOCKEY_STATS, UNTAPPD",
  "job_mode": "TABLE",
  "threads": 2
}
```

Response

201 Created | TIME 4.73 s | SIZE 711 B

Preview | Header 5 | Cookie | Timeline


```
{
  "job_name": "DATAPUMP_REST_EXPORT_20190105140710",
  "owner_name": "HR",
  "operation": "EXPORT",
  "job_mode": "TABLE",
  "state": "EXECUTING",
  "degree": 2,
  "attached_sessions": 0,
  "datapump_sessions": 2,
  "job_state": "EXECUTING",
  "links": [
    {
      "rel": "collection",
      "href": "http://localhost:8080/ords/hr/_/db-api/19.1.0/database/datapump/jobs/"
    },
    {
      "rel": "describedby",
      "href": "http://localhost:8080/ords/hr/_/db-api/19.1.0/metadata-catalog/"
    },
    {
      "rel": "related",
      "href": "http://localhost:8080/ords/hr/_/db-api/19.1.0/database/datapump/jobs/HR,DATAPUMP_REST_EXPORT_20190105140710/EXPDAT-2019-05-01-14_07_11.LOG"
    },
    {
      "rel": "self",
      "href": "http://localhost:8080/ords/hr/_/db-api/19.1.0/database/datapump/jobs/HR,DATAPUMP_REST_EXPORT_20190105140710/"
    }
  ]
}
```

All of this is available at no additional cost

- Oracle 11gR2 and higher databases
- Support tied to your Database license (MOS)
- Classic architecture or Multitenant
- On Premise or Oracle Cloud Infrastructure
- Built by the Oracle Database Team FOR the Oracle Database

Oracle REST Data Services

2011

APEX Listener
- APEX based RESTful Services

2012

APEX Listener v2
- ORDS based REST Services
- Standalone Jetty

2015

ORDS v3.0
- APEX no longer required
- SODA
- AUTO REST

2017

ORDS v17
- REST enabled SQL
- Type 3 JDBC Driver
- Open API (Swagger)

2018

ORDS v18
- PL/SQL Gateway enhancements

2019

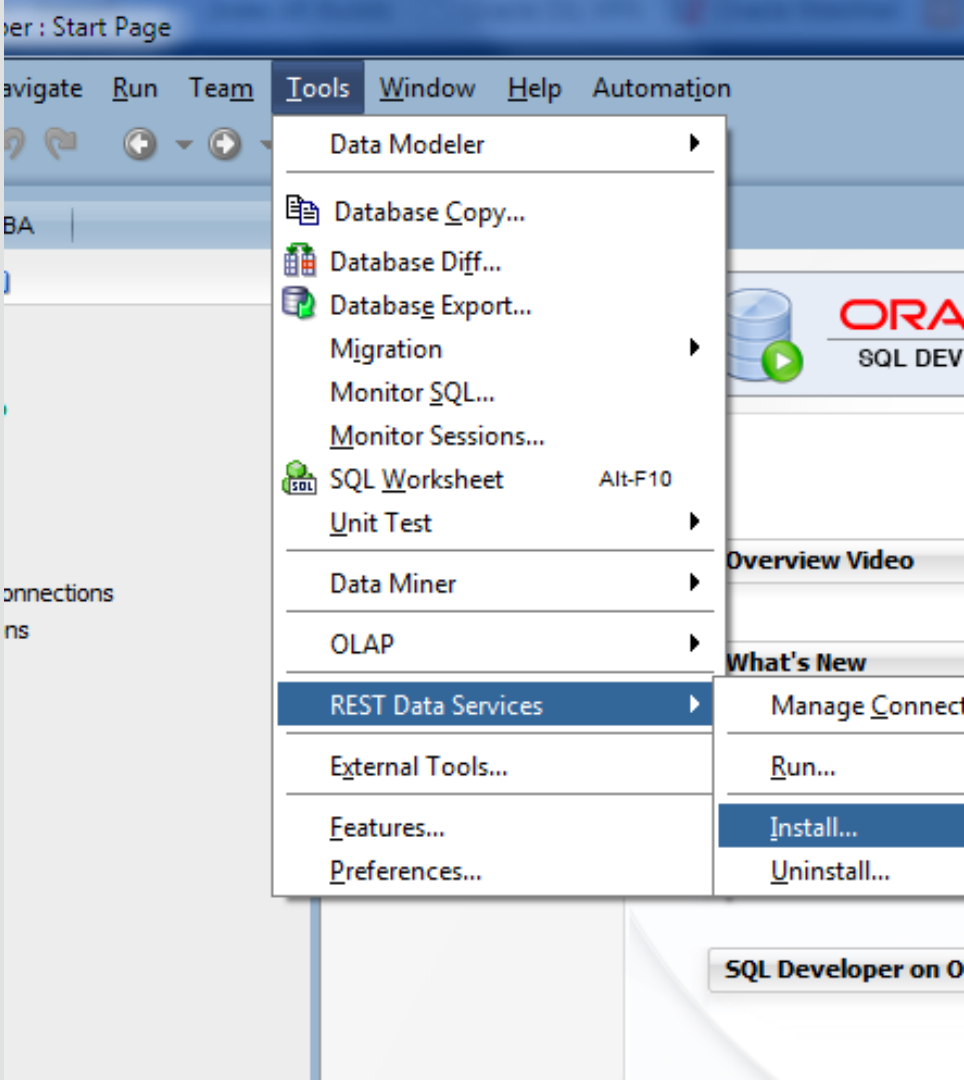
ORDS v19
- DB API
- SQL Developer Web

2020

SQ Developer Web
- REST IDE
- JSON Workshop

Installation & Configuration

How to get started?



Installation & Configuration

- Command line or graphical installers
- Silent install support
- Configure ORDS, then drop into Tomcat or WLS
  - OR run as Standalone process with embedded Jetty webserver

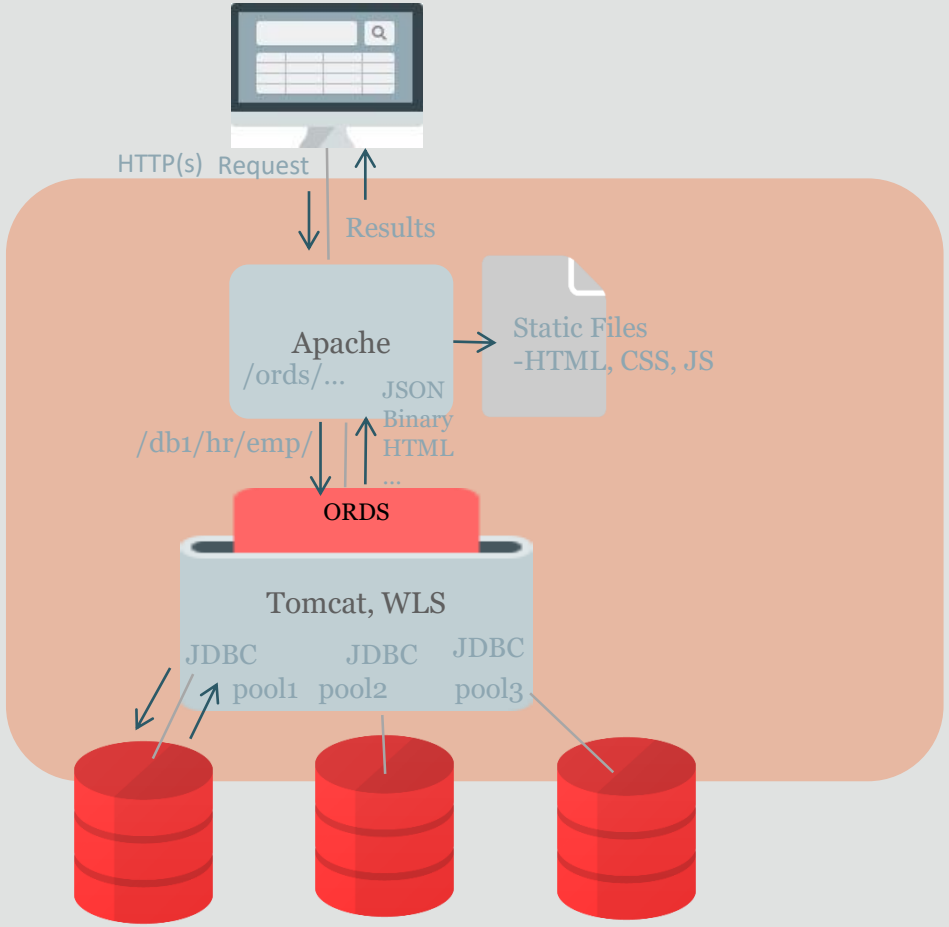
java –jar ords.war install

- Follow the prompts
- Settings written out to %configdir%
- Start when finished

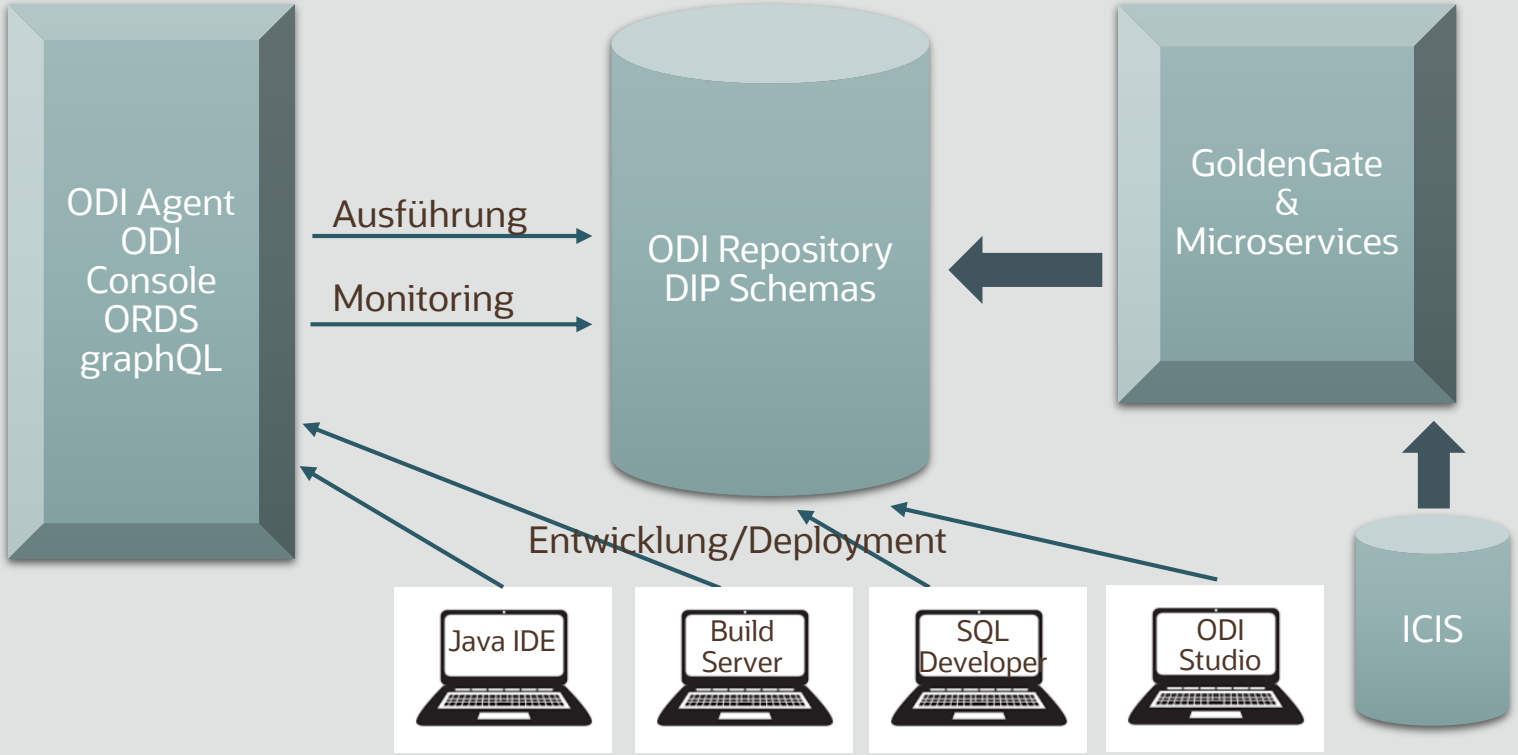
```
c:\ORDS\19.3-Final>java -jar ords.war install advanced
Using existing ORDS configuration files located at c:\ords\config\ords

Verify ORDS schema in Database Configuration apex using connection url jdbc:oracle:thin:@//localhost:1521/orcl

Retrieving information.
2019-12-02T19:20:17.105Z INFO   Oracle REST Data Services schema version 19.3.0.r3161548 is installed.
Enter 1 if you wish to start in standalone mode or 2 to exit [1]:
```

Architecture

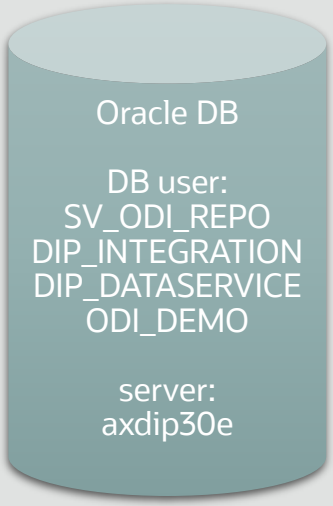
- Webserver layout
- Java Servlet
- Tomcat or WLS
- Standalone mode (Jetty)

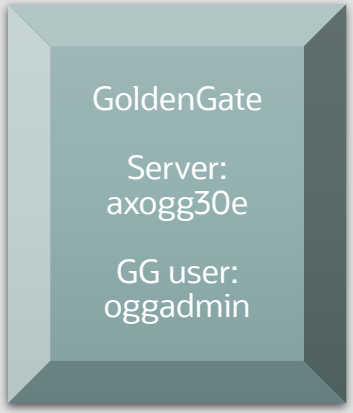


DIP schematisch bei SVi



DIP physisch bei SVi





http://axogg30e:9002

oggadmin / manager

http://lxdip300e:7203/myapp/graphql

http://lxdip300e:7103/ords/mystuff

http://lxdip300e:7201/console

(gqldeploy / gqldeploy1)

http://lxdip300e:7101/console

Logs: /var/log/wls

jdbc:oracle:thin:@axdip30e:1521:DPVIEW10

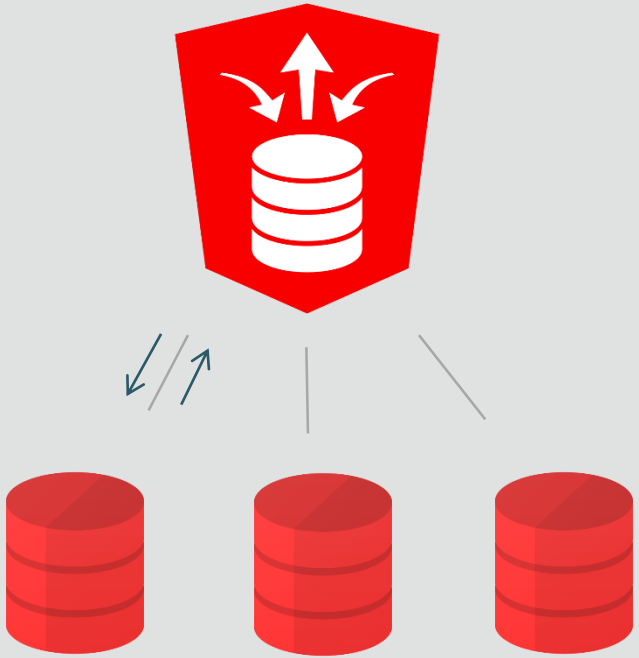
SV_ODI_REPO / start123

DIP_DATASERVICE / wellcome1

DIP_INTEGRATION / wellcome1

ODI_DEMO / odi_demo2021

1:n or 1:1 (ORDS:DB)



- Each database gets a connection pool
- Each database gets a mapping pattern, e.g. /ords/db1/…
- Ideally, 2+ ORDS front-ended with a load balancer for HA

ORDS & Database Communication

- JDBC Conn Pools
- Default Size: 10
- 1st pool => ords/
- Proxy Connects Enabled Schemas

Configuration

- CLI for configuration changes or..
- Edit pool XML and/or Standalone properties files

defaults.xml

C: > ORDS > config > ords > defaults.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE properties SYSTEM "http://java.sun.com/dtd/properties.dtd">
<properties>
<comment>Saved on Tue Nov 12 13:11:52 EST 2019</comment>
<entry key="db.connectionType">basic</entry>
<entry key="db.hostname">localhost</entry>
<entry key="db.port">1521</entry>
<entry key="db.servicename">orcl</entry>
<entry key="feature.sdw">true</entry>
<entry key="restEnabledSql.active">true</entry>
<entry key="database.api.enabled">true</entry>
</properties>
```

Command Prompt - java -jar ords.war setup --database orcl2

```
c:\ORDS\19.3-Final>java -jar ords.war setup --database orcl2
Specify the database connection type to use.
Enter number for [1] Basic  [2] TNS  [3] Custom URL [1]:1
Enter the name of the database server [localhost]:
Enter the database listen port [1521]:
Enter 1 to specify the database service name, or 2 to specify the database SID [1]:
Enter the database service name:orcl2
```

Common Pool Configuration Settings

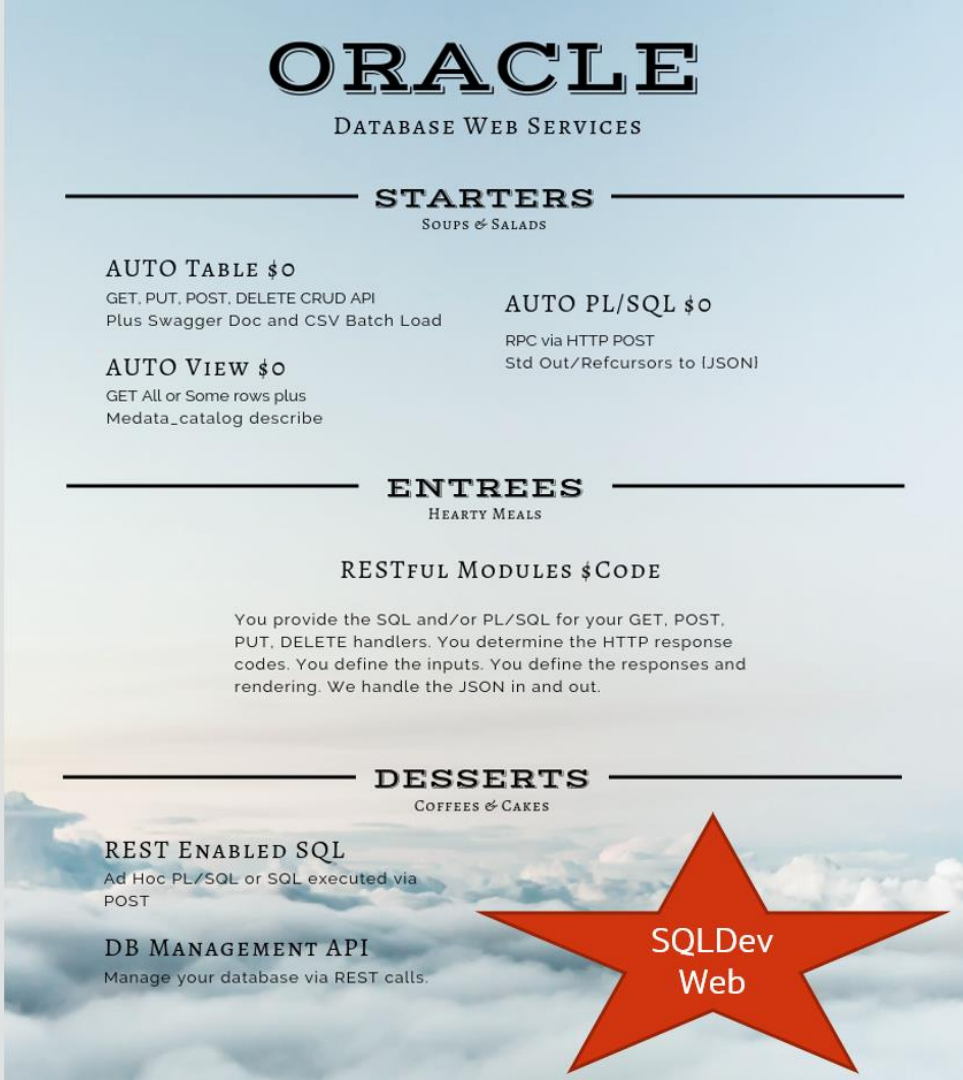
- jdbc.InitialLimit
  # of connections to start with, defaults to 10
- jdbc.MaxLimit
  Maximum # of connections per pool

Bigger Pools <> FASTER!

Help: Real World Perf Sizing Guide

Features

RESTful Services, AutoREST, SQL Developer Web, DB API, SODA, REST Driver



ORDS REST Request & Response Workflow

GET https://host/ords/human/peeps/

```
SQL   SELECT *
        FROM EMP
```

{json}

```
HTTP/1.1 200 OK
{ "employee_id": 100,
"first_name": "Kris"
,"last_name": "Rice",
, “email": “krisrice@oracle.com",
```

Unravelling an ORDS Request URI

https://server:8080/ords/hr/odtug/media/:id

- **ords** – ords.war, Java servlet
- **hr** – schema (alias!), service handler code runs as this user
- **odtug** – module
- **media/:id** – template
- Methods supported
  - GET, PUT, POST, DELETE

Getting Started

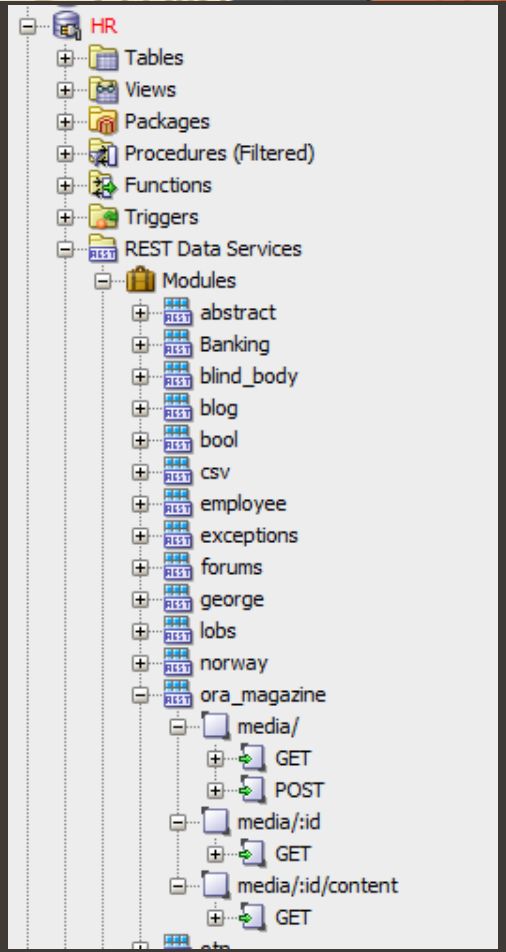
1. Configure ORDS – defaults and DB pool, drop into Tomcat or start standalone
2. REST Enable a SCHEMA
3. Publish a RESTful Service or REST Enable a schema object

```
DECLARE«¶
··PRAGMA·AUTONOMOUS_TRANSACTION;«¶
BEGIN«¶
«¶
····ORDS.ENABLE_SCHEMA(p_enabled·=>·TRUE,«¶
·····················p_schema·=>·'ORDS_DEMO',«¶
·····················p_url_mapping_type·=>·'BASE_PATH',«¶
·····················p_url_mapping_pattern·=>·'autodemo',«¶
·····················p_auto_rest_auth·=>·TRUE);«¶
····«¶
····commit;«¶
«¶
END;
```

REST Enable Schema

- Endpoints defined per schema
- Services executed as that user

Privileges
Resources



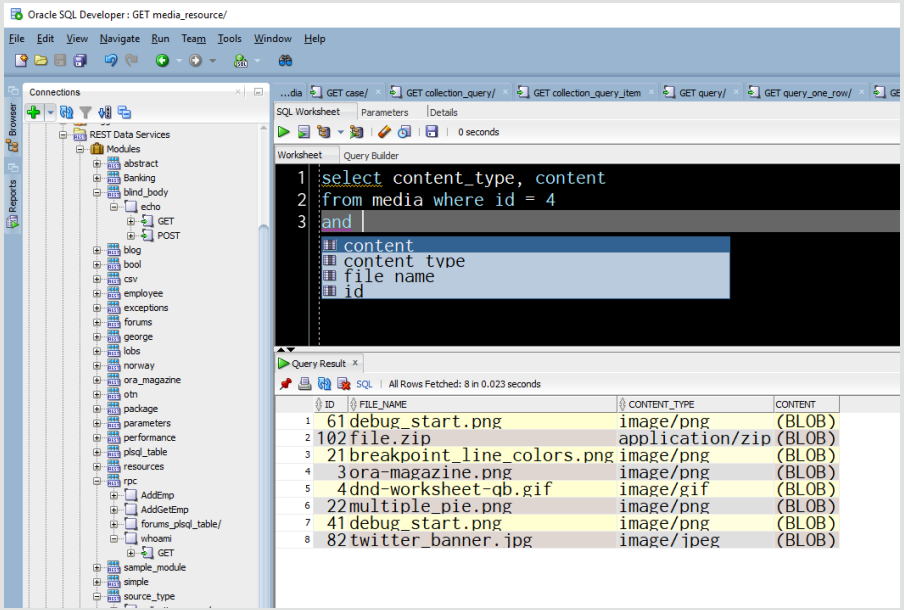
Schema Based Services

- Automatic REST (tables, views, stored procs)
- RESTful Services

HTTP REQ => Database Work

HTTP_CALL	THE_CODE
GET /test/describe	select c.column_name, case
GET /test/version	select banner from v$versio
GET /bool/true/	declare x boolean;begin x :=
GET /test3/handlers	select 1 from dual
GET /employee/employee-no-nulls/	SELECT EMPLOYEE_ID ,FIRST_N
GET /george/ib_test2/	select * from IB_TEST2
GET /test2/sysdate	select sysdate from dual
POST /parameters/headers/:id	begin owa_util.status_lin
GET /test2/media/:id	SELECT content_type, imageF
GET /forums/onecall/	select * employees
GET /test2/case/	select first_name "A", last
POST /blind/echo	begin owa_util.status_line

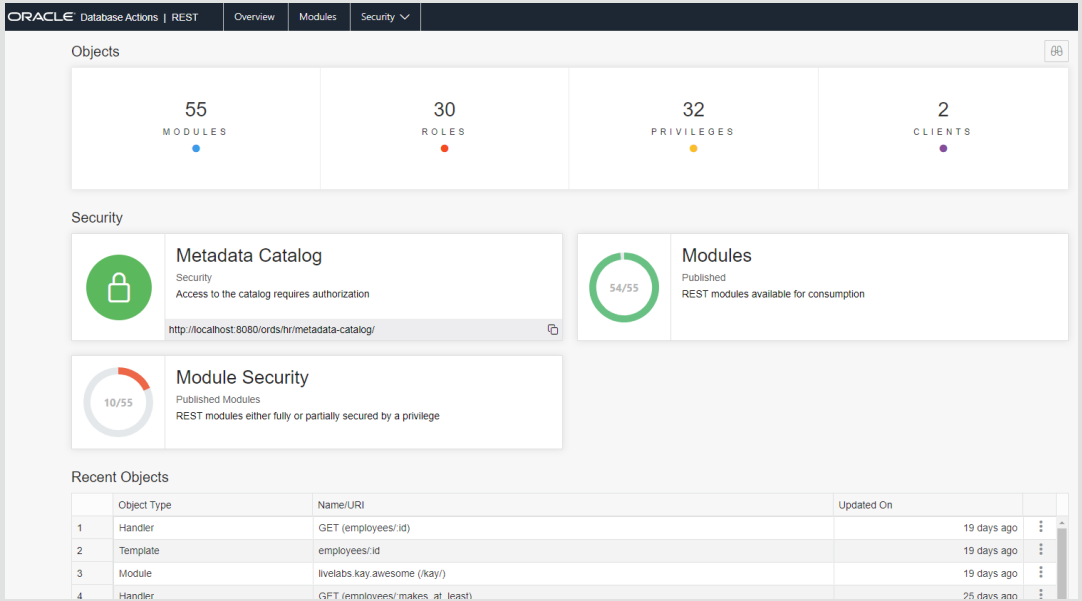
IDE



Oracle SQL Developer

- Full ORDS Integration
- Develop RESTful Services
- REST Enable Objects
- Manage ORDS Roles and Privileges

Browser Based IDE



SQLDev Web

- Develop REST Services
- Manage ORDS Roles, Privileges, OAuth2

Coming Soon: Manage REST Enable Objects (20.4)

CLI & PL/SQL APIs

```
c:\Users\jdsmith\Desktop\ords173>java -jar ords.war help
java -jar ords.war <COMMAND> [Options] [Arguments]

The following commands are available:

               configdir          Set the value of the web.xml
                                  config.dir property

               help               Describe the usage of this
                                  program or its commands

               install            Installs Oracle REST Data
                                  Services

               map-url            Map a URL pattern to the
                                  named database connection

               nosqladd           Add NoSQL store configuration

               nosqldel           Delete NoSQL store
                                  configuration

               oam-config         Configure web.xml to support
                                  Oracle Access Manager
                                  Identity Asserter on Oracle
                                  WebLogic

               plugin             Package one or more plugin
                                  jar files into ords.war

               schema             Install or upgrades ORDS
                                  schema
```

```
DECLARE«¶
··PRAGMA·AUTONOMOUS_TRANSACTION;«¶
BEGIN«¶
«¶
····ORDS.ENABLE_SCHEMA(p_enabled·=>·TRUE,«¶
·····················p_schema·=>·'ORDS_DEMO',«¶
·····················p_url_mapping_type·=>·'BASE_PATH',«¶
·····················p_url_mapping_pattern·=>·'autodemo',«¶
·····················p_auto_rest_auth·=>·TRUE);«¶
····«¶
····commit;«¶
«¶
END;
```

Choose your own adventure!

AUTO REST Advantages

- CRUD APIs, no code
- Single call to create
- Maintained by ORCL
- Lots of features
- Optimized

RESTful Service Advantages

- You're in charge
  - Inputs, outputs, error handling, response codes, formatting
- Full access to SQL/PLSQL
- Easily exported, source controlled
- Transparent

AUTO REST Features

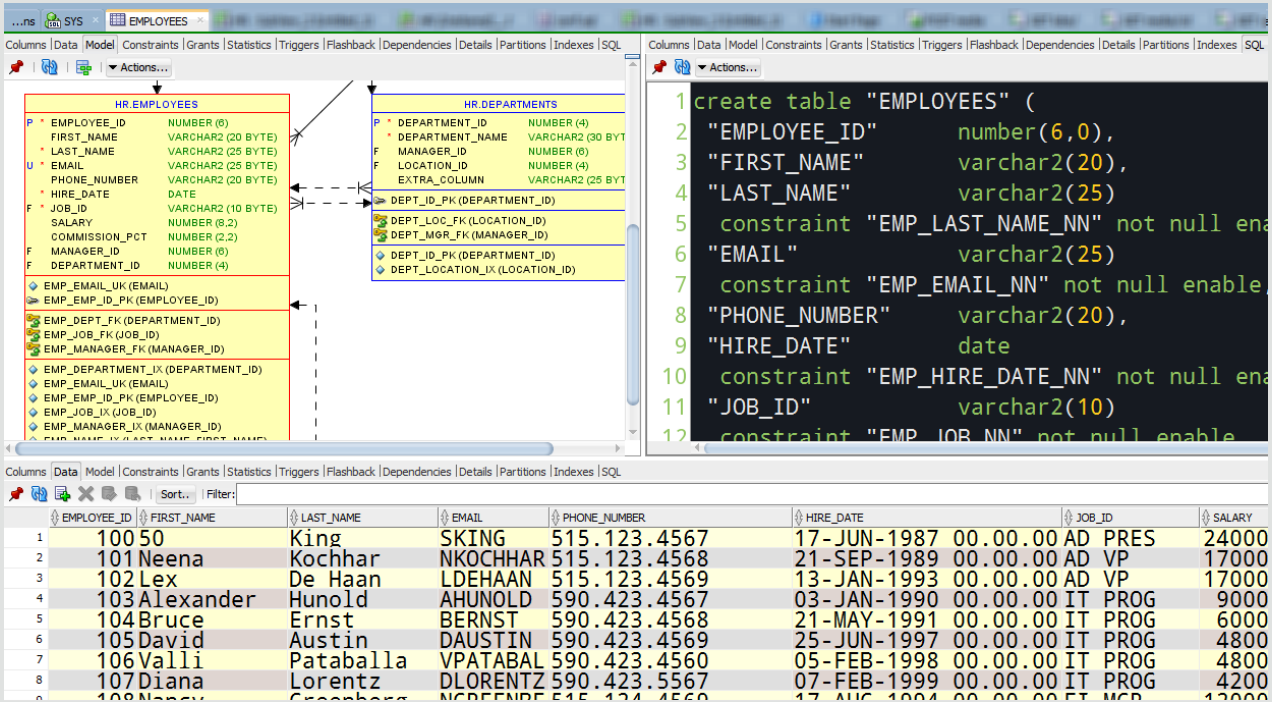
Automatic – ORDS owns the code

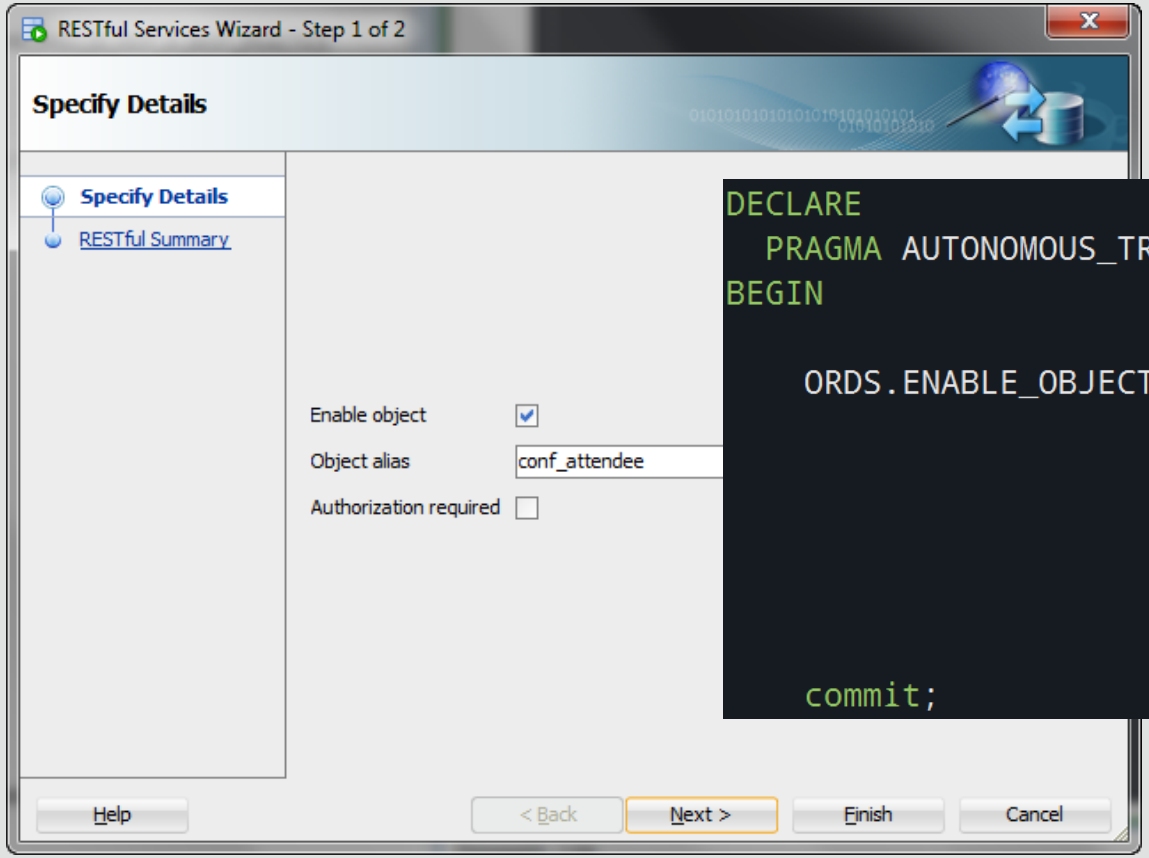
- Auto REST Table
  - Full CRUD API, Data Loading, Metadata (DESC)
- Auto REST View
  - Read interface only (GET)
- Auto PL/SQL (RPC)
  - POST to execute stored PL/SQL
  - We accept {json} in, map to input params, grab output and {json} out

AutoREST Table

/ords/hr/employees/
/ords/hr/employees/:id

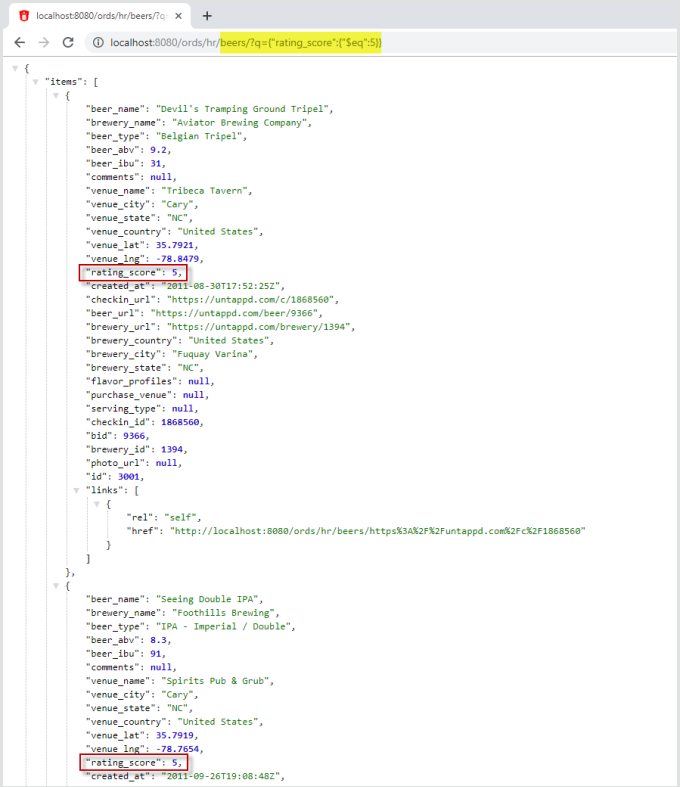
- GET
- PUT
- POST
- DELETE



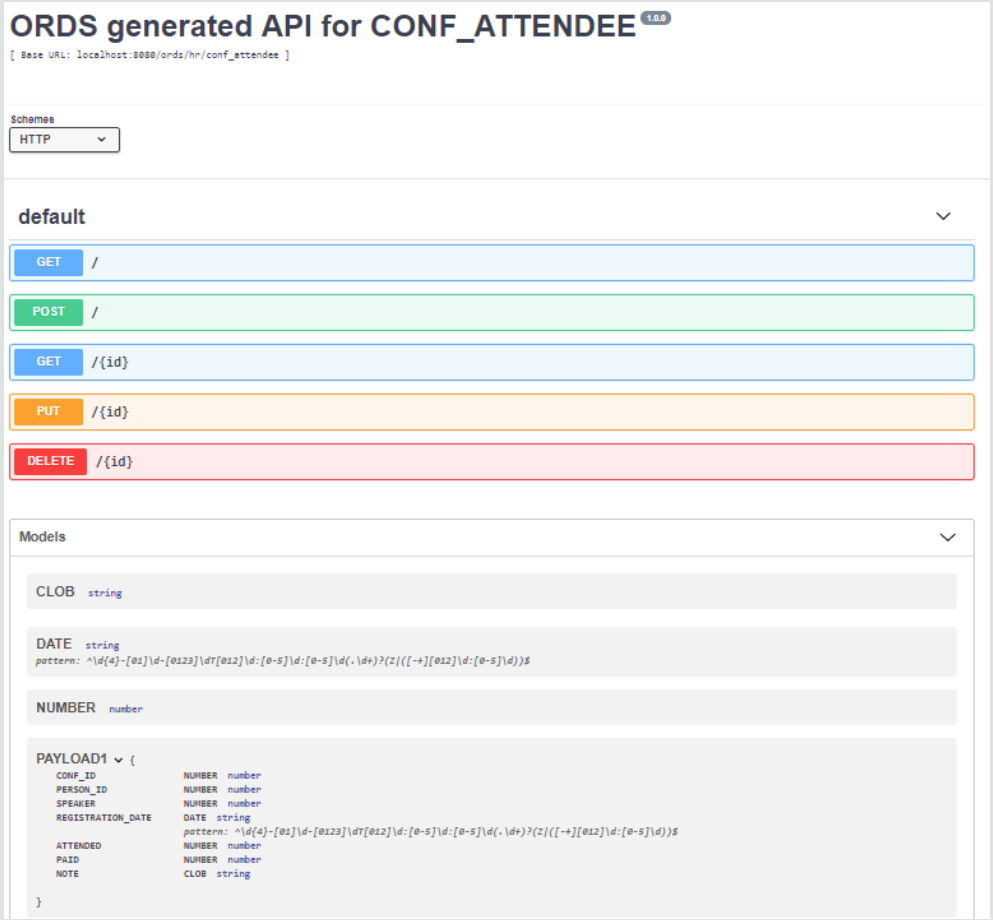



```
DECLARE
  PRAGMA AUTONOMOUS_TRANSACTION;
BEGIN

    ORDS.ENABLE_OBJECT(p_enabled => TRUE,
                       p_schema => 'HR',
                       p_object => 'CONF_ATTENDEE',
                       p_object_type => 'TABLE',
                       p_object_alias => 'conf_attendee',
                       p_auto_rest_auth => FALSE);


    commit;
```

GET on a REST Enabled TABLE



- No code
- SELECT * …
- Query Predicates with ?q={…}
- Paged results
- Links to items



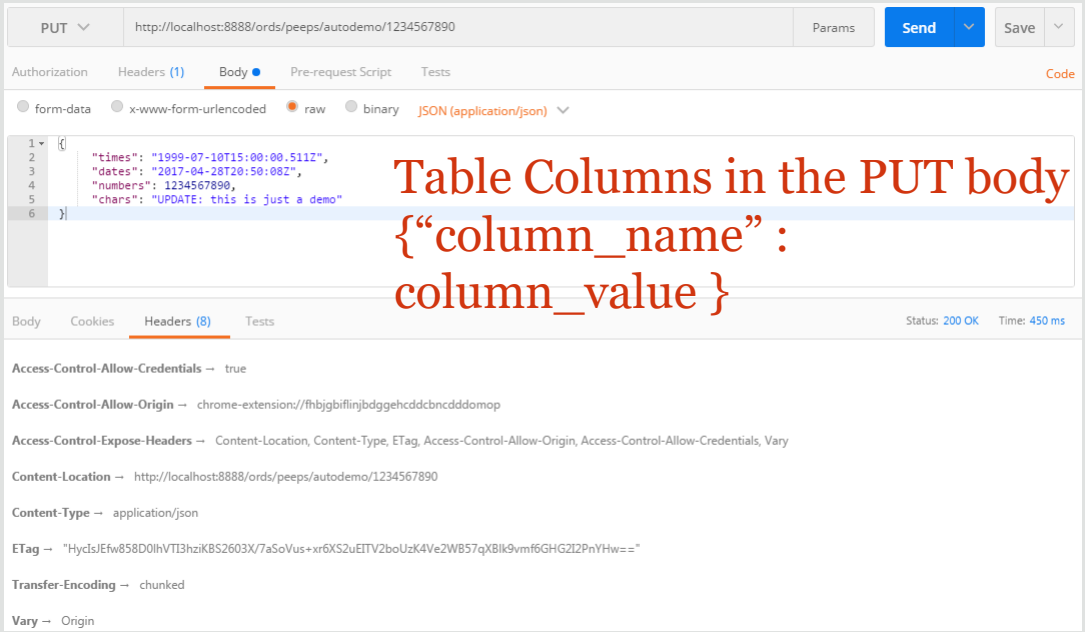
- Schema level Metadata
- Table Metadata
- Get (Select)
  - Query (Filtering/Order/ASOF)
- Insert
- Update
- Delete
- Load CSV

Query the TABLE

- All rows /
- One row /:id – PK Value
  - –No PK, default to ROWID
  - –Multi-column PK /x,y,z
- Some rows /?q={json}

example:
/ords/odidemo/cust360/?q={"$eq": {"custid":102}}

Update a ROW



METHOD : PUT /:PK

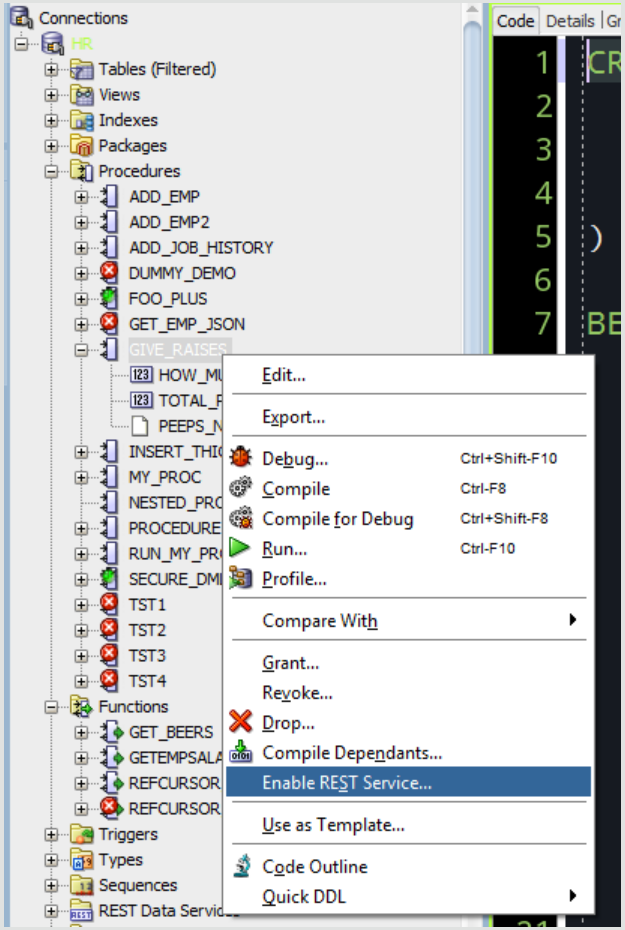
REQUEST BODY : JSON

RESPONSE: 200 OK

- Location (Header)
- JSON (Body)

Existing PL/SQL APIs?

- We auto-magically handle PL/SQL too
- RPC -> POST via HTTPS /ords/hr/procedureA
- Responses & Results (OUTs/RETURNs/REFCURSORs), in {JSON}



RPC over HTTPS (POST)

ORDS takes parameters as JSON, executes PL/SQL, grabs output, sends back down as JSON

OUT INTEGER & SYS_REFCURSOR

```
{
  "total_payroll": 631230,
  "peeps_numbers": [
    {
      "id": 81,
      "name": "Dummy4",
      "salary": 0,
      "hire_date": "2017-06-20T13:29:00Z"
    },
    {
      "id": 65,
      "name": "Bart",
      "salary": 0,
      "hire_date": "2017-06-20T13:29:00Z"
    },
    {
      "id": 79,
      …
    }
```

POST http://localhost:8888/ords/peeps/thickdb/

Authorization Headers Body Pre-request Script Tests

Type No Auth

Body Cookies Headers (6) Tests

Pretty Raw Preview JSON


```
{
  "ret": [
    {
      "log_id": 92996,
      "log_date": "2016-11-07T11:58:02.517Z",
      "owner": "ORDS_METADATA",
      "job_name": "CLEAN_OLD_ORDS_SESSIONS",
      "job_subname": null,
      "status": "SUCCEEDED",
      "error#": 0,
      "req_start_date": "2016-11-05T23:07:45.85Z",
      "actual_start_date": "2016-11-06T02:06:21.506Z",
      "run_duration": {
        "DAYS": 1,
        "HOURS": 9,
        "MINUTES": 51,
        "SECONDS": 36
      },
      "instance_id": 1,
      "session_id": "89,57445",
      "slave_pid": "4907",
      "cpu_used": {
        "NANOS": 180000000
      },
      "credential_owner": null,
      "credential_name": null,
      "destination_owner": null,
      "destination": null,
      "additional_info": null,
      "errors": null,
      "output": null,
      "binary_errors": null,
      "binary_output": null
```


RETURN REFCURSOR

Auto PLSQL - Insert Record

POST http://localhost:8888/ords/peeps/insert_thickdb/ Params Send Save

Authorization Headers (1) Body Pre-request Script Tests Code

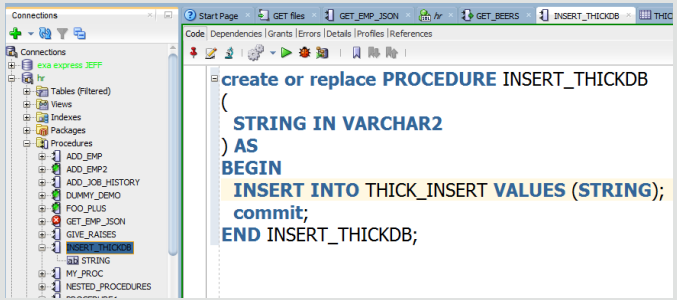
form-data x-www-form-urlencoded raw binary JSON (application/json)

```
{"string":"good morning Montreal"}
```

Body Cookies Headers (6) Tests Status: 200 OK Time: 3398 ms

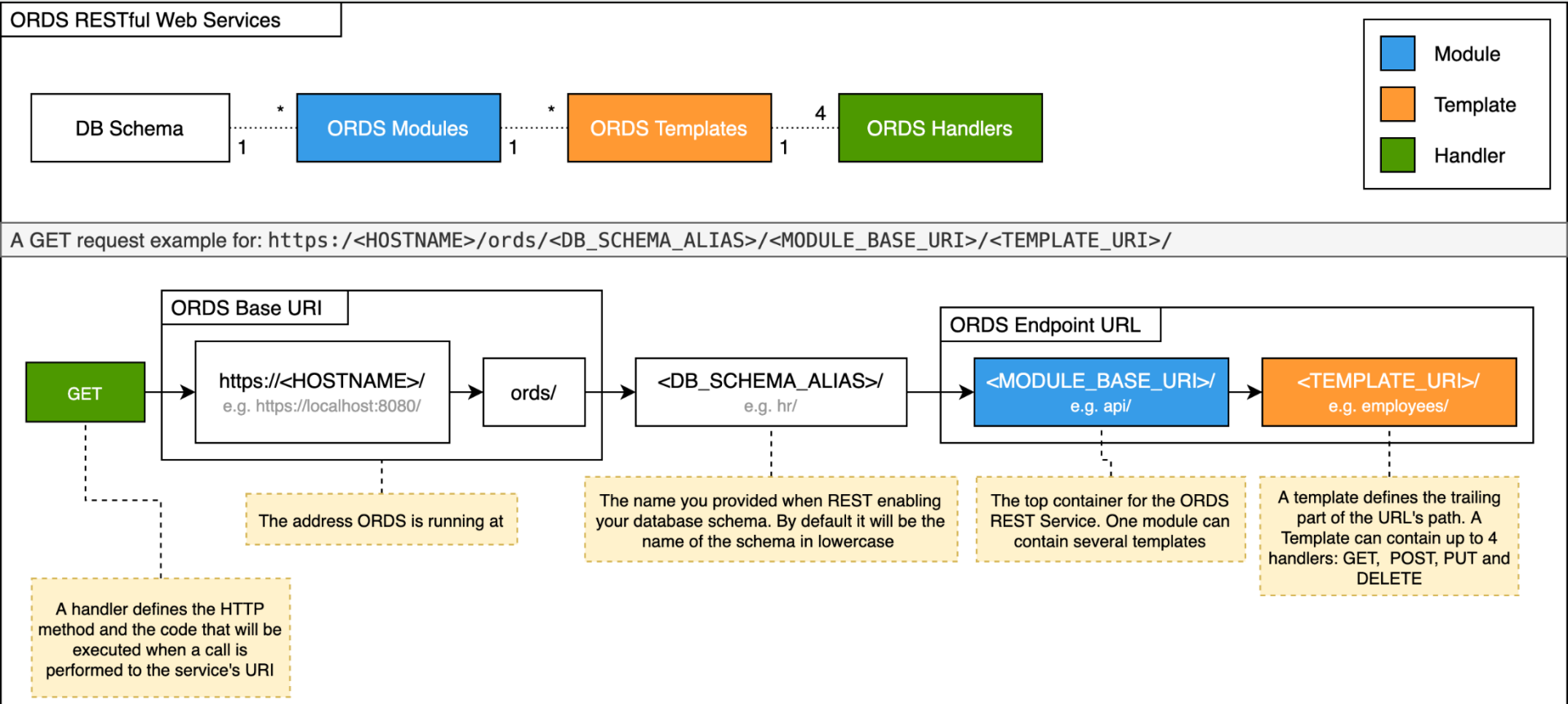
Pretty Raw Preview Save Response

{}



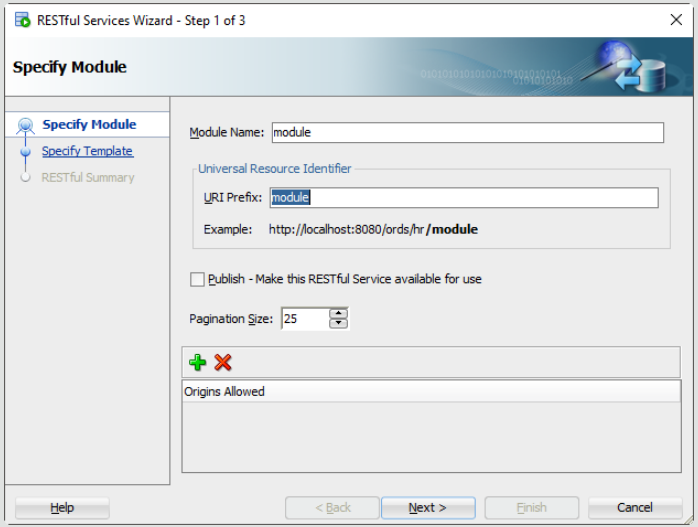
PL/SQL TABLE API

RESTful Services (Modules, Templates, & Handlers)

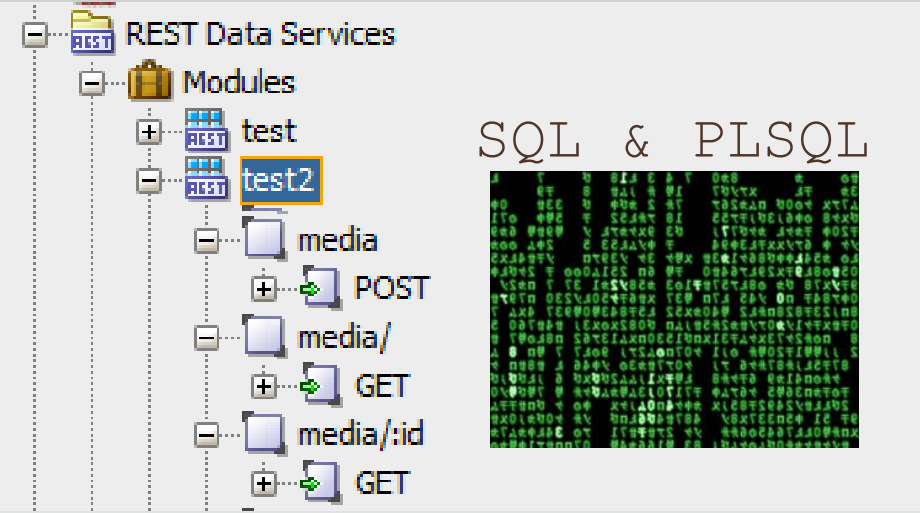


Modules (“packages”)

- A collection of RESTful Services
- Secured as a group with a privilege
- Published or hidden as a group
- Default response pagination size



URI Templates

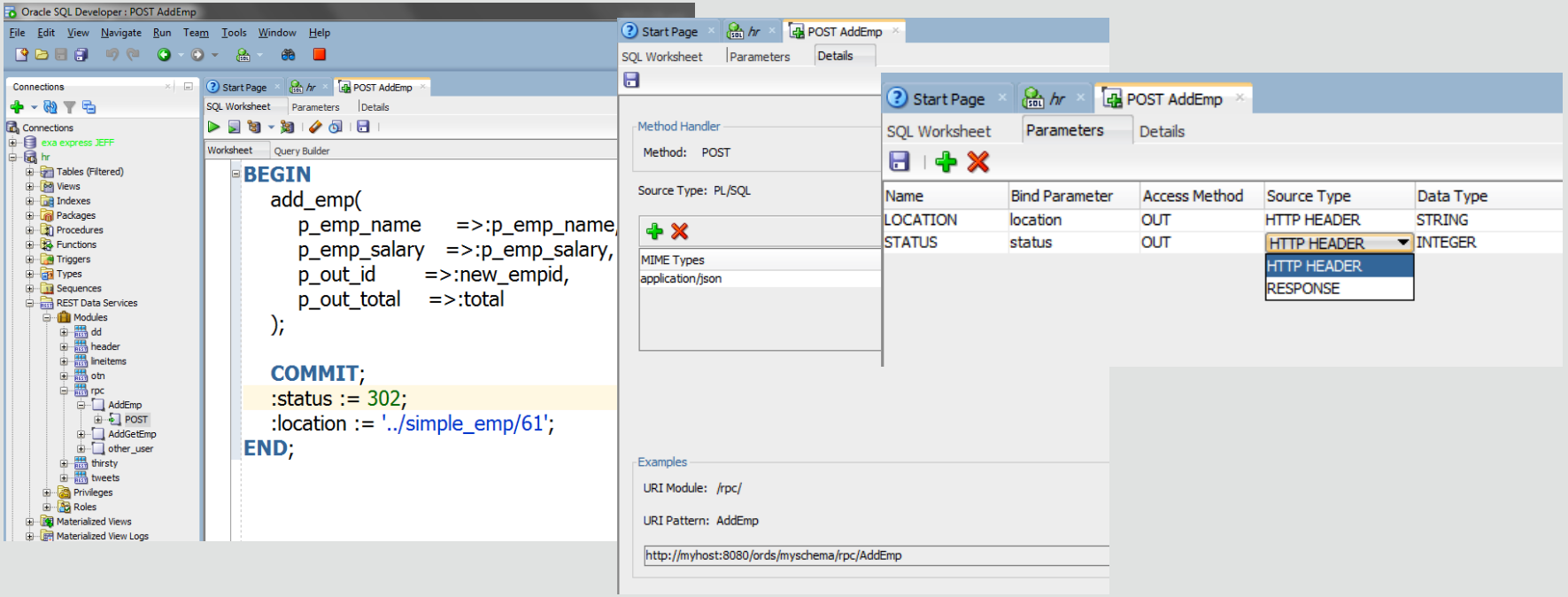


/ords/hr/test2/**employees**

/ords/hr/test2/**employees/**

/ords/hr/test2/**employees/:id**

Handlers – inputs, code, & response



Source Types - Shapes Responses

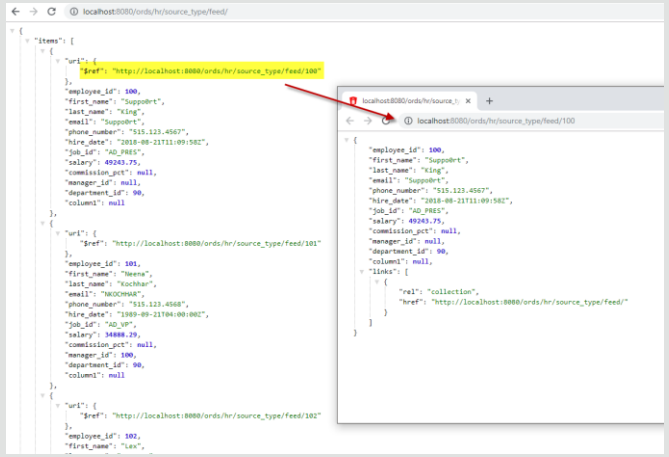
- Collection Query
  - Multiple records/paging
- Collection Query Item
  - Single record only/no paging


```
            "employee_id": 124,
            "first_name": "Kevin",
            "last_name": "Mourgos",
            "email": "KMOURGOS",
            "phone_number": "650.123.5234",
            "hire_date": "1999-11-16T05:00:00Z",
            "job_id": "ST_MAN",
            "salary": 11919.54,
            "commission_pct": null,
            "manager_id": 100,
            "department_id": 50,
            "column1": null
        }
    ],
    "first": {
        "$ref": "http://localhost:8080/ords/hr/source_type/query/"
    },
    "next": {
        "$ref": "http://localhost:8080/ords/hr/source_type/query/?page=1"
    }
}
```


No Metadata

Source Types – Determines Responses

- Feed
  - –Multiple records/paging
  - –First column auto-generates $link
- PL/SQL
  - –Executes PL/SQL Block
- Media Resource
  - –First Column sets Mime Type
  - –Second Column sets raw content (no json-ification)



Feed Source Type

Manual JSONification: JSON DB API

CUSTOMERS

CUST_ID	CUST_LAST_NAME	CUST_CITY
25451	Everett	Heilbronn
25452	Odenwalld	Heilbronn
…	…	…

```
{
  "custid":25451,
  "name":“Everett",
  "City":“Heilbronn“
}
```

Generating JSON – easy with SQL

- JSON_OBJECT creates single JSON objects

```
select json_object('custid' is CUSTID,
                   'name' is LAST_NAME,
                   'City' is CITY) output
from src_customer, src_city
where src_customer.city_id = src_city.city_id;
```

```
AUSGABE
--------------------------------------------------------------------
{"custid":103075,"name":"Haus","City":"Tucumcari"}
{"custid":104203,"name":"Seto","City":"Oran"}
{"custid":102374,"name":"Hornick","City":"Mackville"}
{"custid":102440,"name":"Spivak","City":"Atwood"}
{"custid":102943,"name":"Kelly","City":"Rock Creek"}
{"custid":101789,"name":"Krishnan","City":"Lowndesville"}
{"custid":100115,"name":"Carbery","City":"Frederick"}
```

- JSON_OBJECTAGG aggregates rows into a single column

```
select city,
       json_objectagg('name' is last_name) cust_per_city
from src_customer, src_city
where city like 'Co%'
and src_customer.city_id like src_city.city_id
group by city;
```

Result

```
CUST_CITY
------------------------------
CUST_PER_CITY
--------------------------------------------------------------------------------
Coburg
{"name":"Rice","name":"Tillman","name":"Barnett","name":"Zimmer","name":"Klesser
","name":"Ingersoll","name":"Bane","name":"Vankirk","name":"Lefevre","name":"Tri
mmer","name":"Smeed","name":"Lipp","name":"Klemm","name":"Obrien","name":"Kelley
","name":"Titus","name":"Felt","name":"Joseph","name":"Colven","name":"Snodgrass
","name":"Rosenblum","name":"Burgess","name":"Kraft","name":"Krebs","name":"Zhao
","name":"Carr","name":"Kish","name":"Dwyer","name":"Eastwood","name":"Haole","n
ame":"Lamar","name":"Lengel","name":"Knox","name":"Mannings","name":"Curr","name
":"Jeffreys","name":"Lickey","name":"Ingold","name":"Goode","name":"Kelleher","n
ame":"Ladd","name":"Greenley","name":"Rush","name":"Gilmour","name":"Baley"}

Cochem
{"name":"Eden","name":"Pack","name":"Hale","name":"Newsome","name":"Calahan","na
me":"Lucy","name":"Newkirk","name":"Jewell","name":"Parks","name":"Rust","name":
"Sampler","name":"Stone","name":"Poindexter","name":"Geralt","name":"Wiser","nam
e":"Valentino","name":"Bakker"}
```

Generating JSON – easy with SQL

- JSON_ARRAYAGG is an aggregate function returning a JSON Array

```
select json_object('country_name' is r.country,
                   'ctinfo' is
        (select json_arrayagg(
                json_object('name' is last_name,
                            'phone' is phone,
                            'city' is city,
                            'street' is address absent on null)
                            absent on null)
                from src_customer co , src_city c, src_region r
                where co.city_id = c.city_id
                and c.region_id = r.region_id
                and age < 40)
                absent on null) as ct1
from src_region r;
```

The result – an excerpt

	CT1
1	{"country_name":"USA","ctinfo":[{"name":"McCarthy","phone":"(214) 555 3075","city":"Dallas","street":"27 Pasadena Drive"},{"name":"Travis","phone":"(510) 555 4448","city":"San Francis
2	{"country_name":"USA","ctinfo":[{"name":"McCarthy","phone":"(214) 555 3075","city":"Dallas","street":"27 Pasadena Drive"},{"name":"Travis","phone":"(510) 555 4448","city":"San Francis
3	{"country_name":"USA","ctinfo":[{"name":"McCarthy","phone":"(214) 555 3075","city":"Dallas","street":"27 Pasadena Drive"},{"name":"Travis","phone":"(510) 555 4448","city":"San Francis
4	{"country_name":"USA","ctinfo":[{"name":"McCarthy","phone":"(214) 555 3075","city":"Dallas","street":"27 Pasadena Drive"},{"name":"Travis","phone":"(510) 555 4448","city":"San Francis
5	{"country_name":"France","ctinfo":[{"name":"McCarthy","phone":"(214) 555 3075","city":"Dallas","street":"27 Pasadena Drive"},{"name":"Travis","phone":"(510) 555 4448","city":"San Fran
6	{"country_name":"France","ctinfo":[{"name":"McCarthy","phone":"(214) 555 3075","city":"Dallas","street":"27 Pasadena Drive"},{"name":"Travis","phone":"(510) 555 4448","city":"San Fran
7	{"country_name":"France","ctinfo":[{"name":"McCarthy","phone":"(214) 555 3075","city":"Dallas","street":"27 Pasadena Drive"},{"name":"Travis","phone":"(510) 555 4448","city":"San Fran
8	{"country_name":"France","ctinfo":[{"name":"McCarthy","phone":"(214) 555 3075","city":"Dallas","street":"27 Pasadena Drive"},{"name":"Travis","phone":"(510) 555 4448","city":"San Fran
9	{"country_name":"Great Britain","ctinfo":[{"name":"McCarthy","phone":"(214) 555 3075","city":"Dallas","street":"27 Pasadena Drive"},{"name":"Travis","phone":"(510) 555 4448","city":"S
10	{"country_name":"Great Britain","ctinfo":[{"name":"McCarthy","phone":"(214) 555 3075","city":"Dallas","street":"27 Pasadena Drive"},{"name":"Travis","phone":"(510) 555 4448","city":"S
11	{"country_name":"Great Britain","ctinfo":[{"name":"McCarthy","phone":"(214) 555 3075","city":"Dallas","street":"27 Pasadena Drive"},{"name":"Travis","phone":"(510) 555 4448","city":"S
12	{"country_name":"Great Britain","ctinfo":[{"name":"McCarthy","phone":"(214) 555 3075","city":"Dallas","street":"27 Pasadena Drive"},{"name":"Travis","phone":"(510) 555 4448","city":"S
13	{"country_name":"Nederland","ctinfo":[{"name":"McCarthy","phone":"(214) 555 3075","city":"Dallas","street":"27 Pasadena Drive"},{"name":"Travis","phone":"(510) 555 4448","city":"San F
14	{"country_name":"Nederland","ctinfo":[{"name":"McCarthy","phone":"(214) 555 3075","city":"Dallas","street":"27 Pasadena Drive"},{"name":"Travis","phone":"(510) 555 4448","city":"San F

Generating JSON – easy with SQL

- Special values in JSON

Boolean, e.g. true/false NOT as string

```
select json_object('is_Boolean' is 'true' format json) from dual;
```

	JSON_OBJECT('IS_BOOLEAN'IS'TRUE'FORMATJSON)
1	{"is_Boolean":true}

Null values as “null” or leave them out ?

```
select json_object ('nullField' is null ) from dual;
select json_object ('nullField' is null absent on null) from dual;
```

	JSON_OBJECT('NULLFIELD'ISNULL)
1	{"nullField":null}

VS

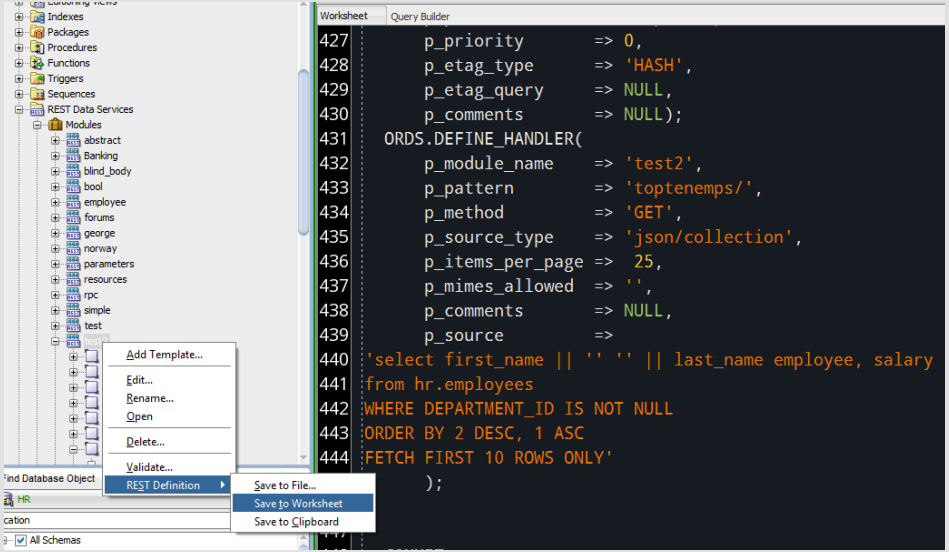
	JSON_OBJECT('NULLFIELD'ISNULLAB
1	{}

Helpers for your Code, Automatic :binds

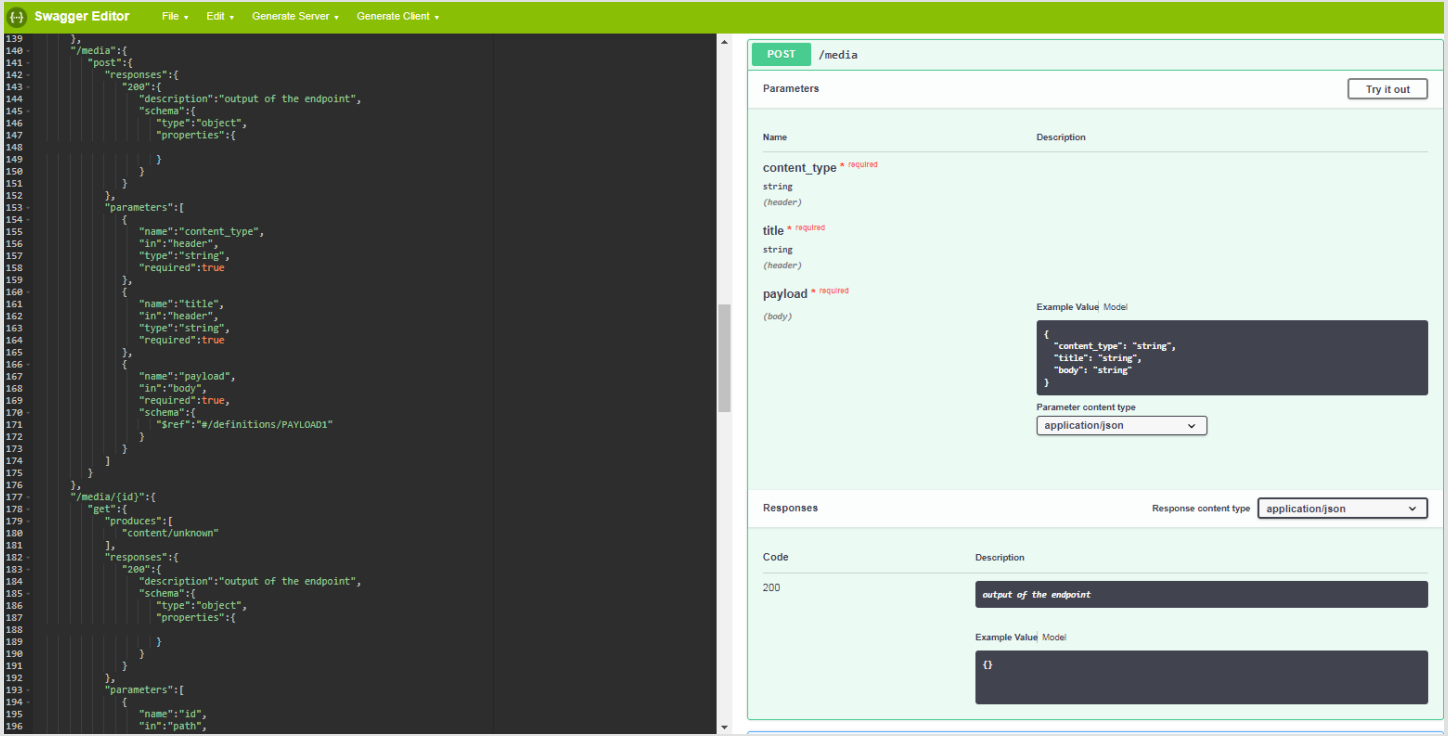
- POST request body
  - **:body** (BLOB) or **:body_text** (CLOB)
- Who is accessing the end point
  - Authenticated user is mapped to :current_user
- HTTP Response Status :status_code & Redirects :forward_location
- :content_type, :fetch_size, :fetch_offset, :row_count, :page_offset

RESTful Service Source

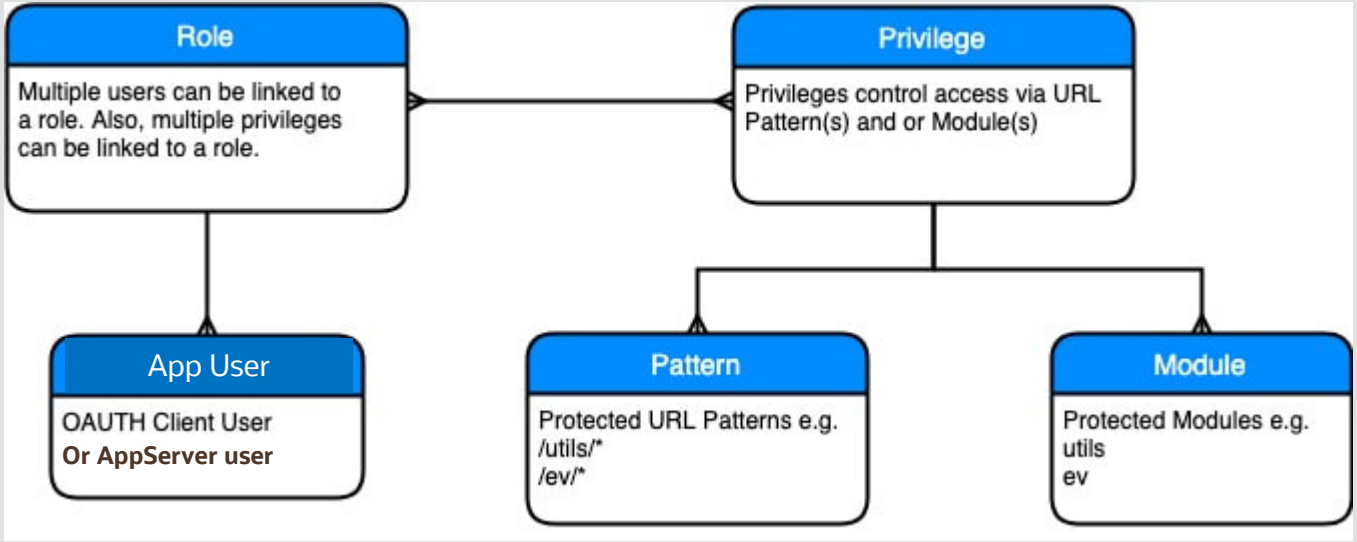
```
I am HR on orcl > rest export test2
-- Generated by SQLcl REST Data Services 18.1.0.0
-- Exported REST Definitions from ORDS Schema Version 17.4.1.353.06.48
-- Schema: HR   Date: Mon Jun 11 13:52:12 EDT 2018
--
BEGIN
  ORDS.ENABLE_SCHEMA(
      p_enabled             => TRUE,
      p_schema              => 'HR',
      p_url_mapping_type    => 'BASE_PATH',
      p_url_mapping_pattern => 'hr',
      p_auto_rest_auth      => FALSE);

  ORDS.DEFINE_MODULE(
      p_module_name    => 'test2',
      p_base_path      => '/test2/',
      p_items_per_page =>  25,
      p_status         => 'PUBLISHED',
      p_comments       => NULL);
  ORDS.DEFINE_TEMPLATE(
      p_module_name    => 'test2',
      p_pattern        => 'case/',
      p_priority       => 0,
      p_etag_type      => 'HASH',
      p_etag_query     => NULL,
      p_comments       => NULL);
  ORDS.DEFINE_HANDLER(
      p_module_name    => 'test2',
      p_pattern        => 'case/',
      p_method         => 'GET',
      p_source_type    => 'json/query',
      p_items_per_page =>  25,
      p_mimes_allowed  => '',
      p_comments       => NULL,
      p_sourc          =>
```



OpenAPI/Swagger



Access Control



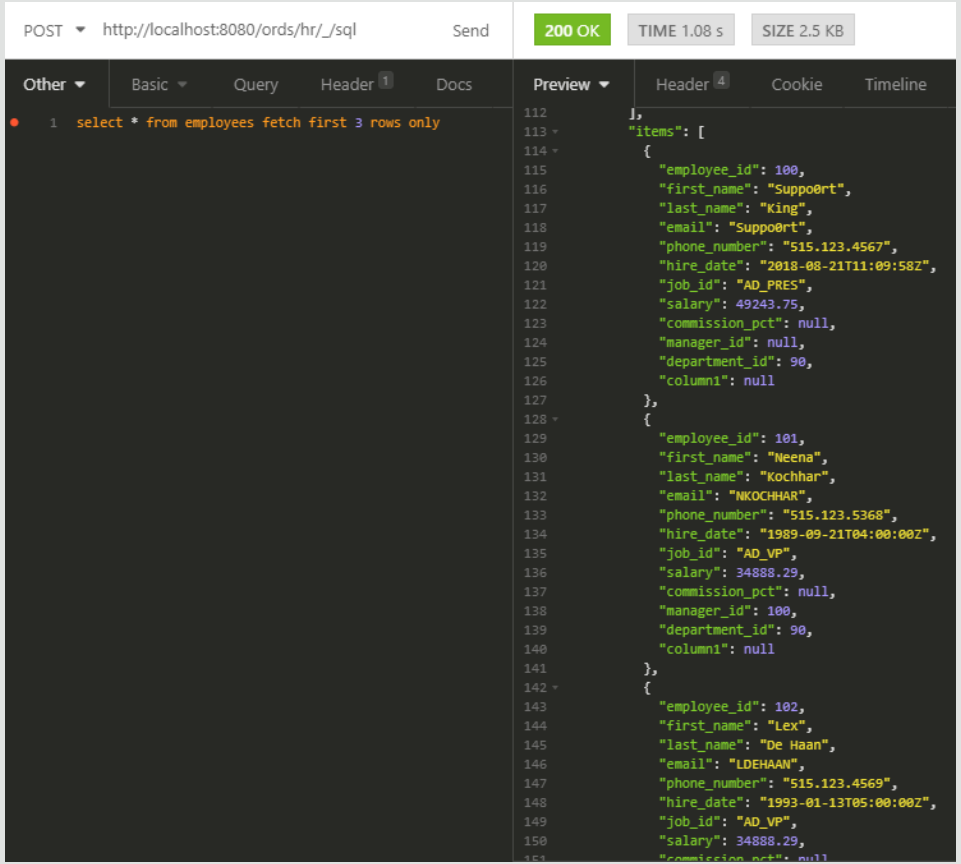
The ‘Extras’

Other things you GET out-of-the-box

- REST Enabled SQL & Type II REST JDBC Driver
- DB API
- SQL Developer Web
- SODA

REST Enabled SQL

- SQL as a Service?
- Application/SQL
- Query on the Post Body
- Single statement or SCRIPT




```
<entry key="restEnabledSql.active">true</entry>
```

No SQLNet Access? Try our REST Driver

```
I am  on  > connect dev/oracle@jdbc:oracle:orest:@http://localhost:8888/ords/ords_demo/
Connected.

Session altered.

DDL Option PRETTY on
DDL Option CONSTRAINTS off
DDL Option REF_CONSTRAINTS off
DDL Option PARTITIONING on
DDL Option TABLESPACE on
DDL Option SEGMENT_ATTRIBUTES off
DDL Option STORAGE off
'HELLOJEFF'
hello jeff


I am ORDS_DEMO on jdbc:oracle:orest:@http://localhost:8888/ords/ords_demo/ > _
```

- jdbc:oracle:orest:
- Single JAR on SQLcl downloads page

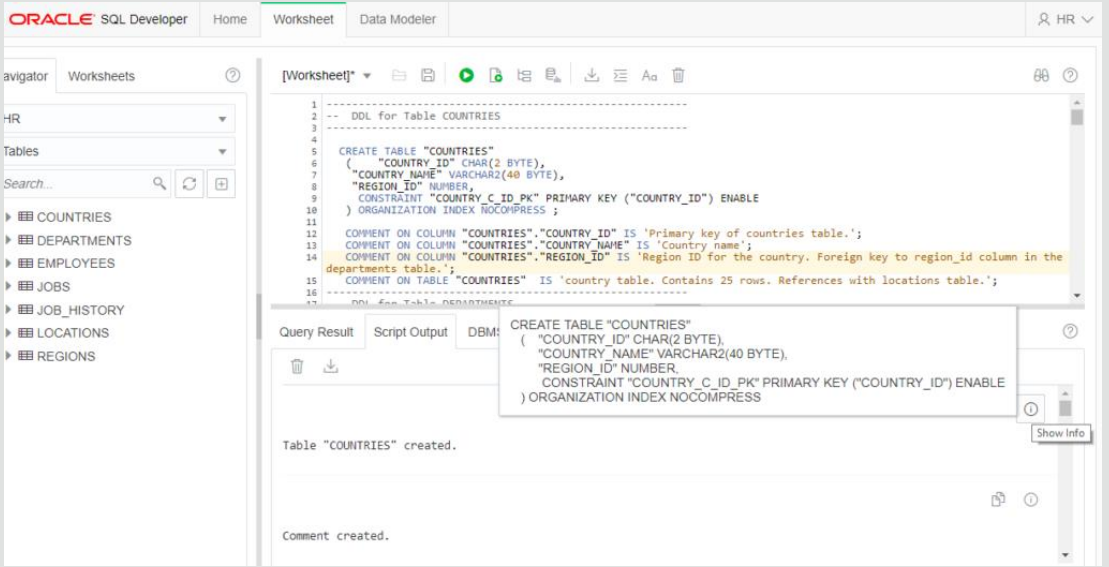
New for ORDS 19.4 – SQL Developer Web

- Requires the DB API & REST Enabled SQL
- Database User Authentication/REST Enabled Schemas Only
- HTML5/js (Oracle JET)
- 11gR2 and higher

Features

- SQL Worksheet
- Create/Edit dialogs
- Excel/CSV Imports
- Schema Diagramming
- REST IDE
- JSON Workshop
- Performance Hub

  ASH Analytics
  Real Time SQL Monitoring

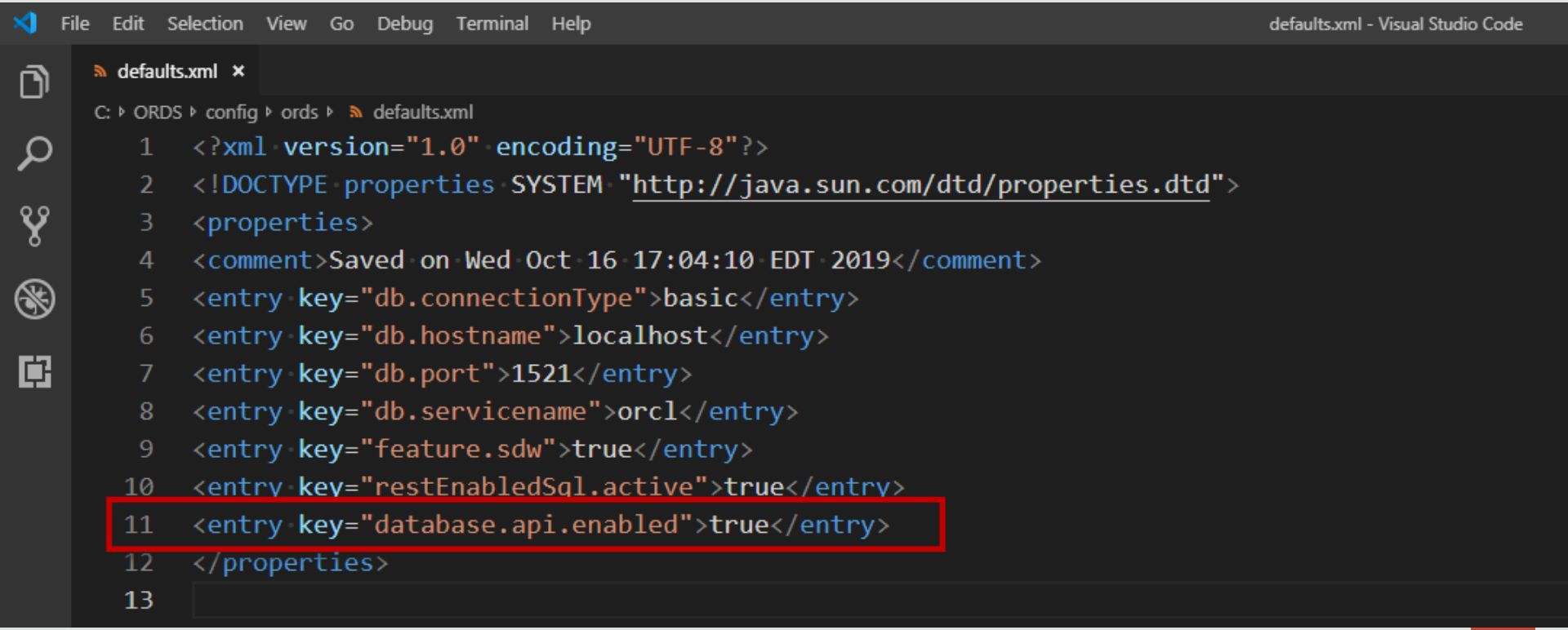


Database Management APIs

Use HTTPS to manage and monitor your Oracle Database

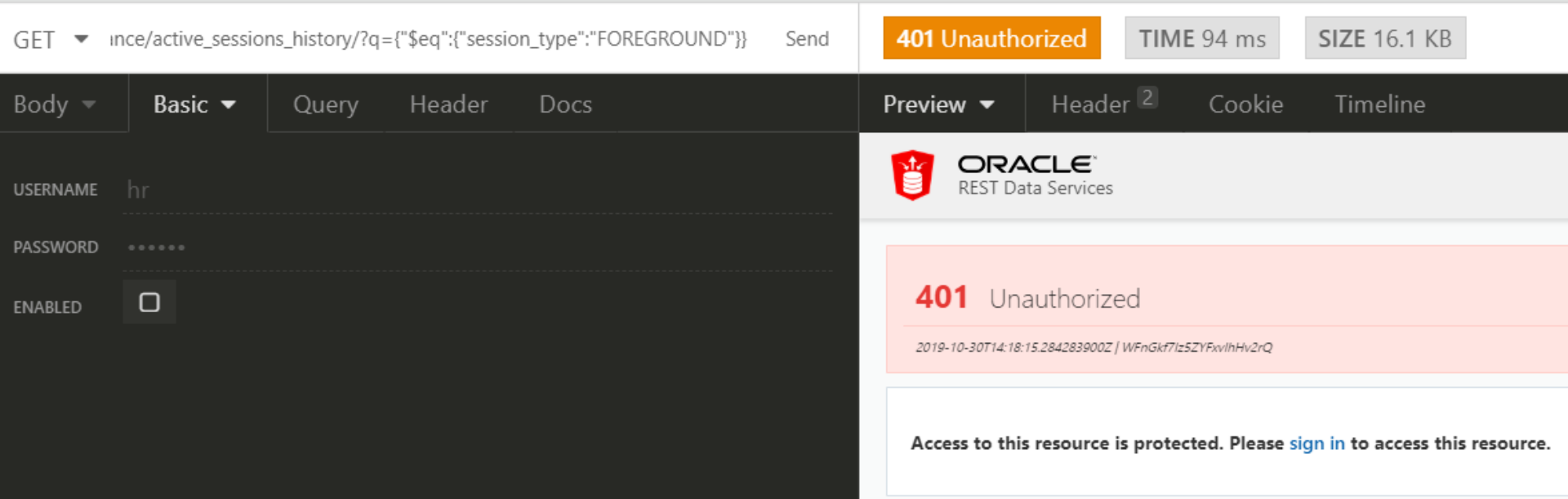
- Supports 11gR2, 12, 18, & 19c
- General, Data Dictionary, Monitoring, Performance, & PDB Lifecycle Management
- Automatic – no code to write, just enable and use it

Enabling it…


File Edit Selection View Go Debug Terminal Help
defaults.xml - Visual Studio Code
defaults.xml
C: ORDS config ords defaults.xml
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE properties SYSTEM "http://java.sun.com/dtd/properties.dtd">
3 <properties>
4 <comment>Saved on Wed Oct 16 17:04:10 EDT 2019</comment>
5 <entry key="db.connectionType">basic</entry>
6 <entry key="db.hostname">localhost</entry>
7 <entry key="db.port">1521</entry>
8 <entry key="db.servicename">orcl</entry>
9 <entry key="feature.sdw">true</entry>
10 <entry key="restEnabledSql.active">true</entry>
11 <entry key="database.api.enabled">true</entry>
12 </properties>
13

Auth

- Authenticate using REST Enabled Oracle USERS



User Setup

Option #1, Mid Tier User:

```
c:\ORDS\19.1-GA>java -jar ords.war user jeff "SQL Administrator"
Enter a password for user jeff:
Confirm password for user jeff:
Apr 23, 2019 9:38:54 AM oracle.dbtools.standalone.ModifyUser execute
INFO: Created user: jeff in file: C:\ORDS\config\ords\credentials
```

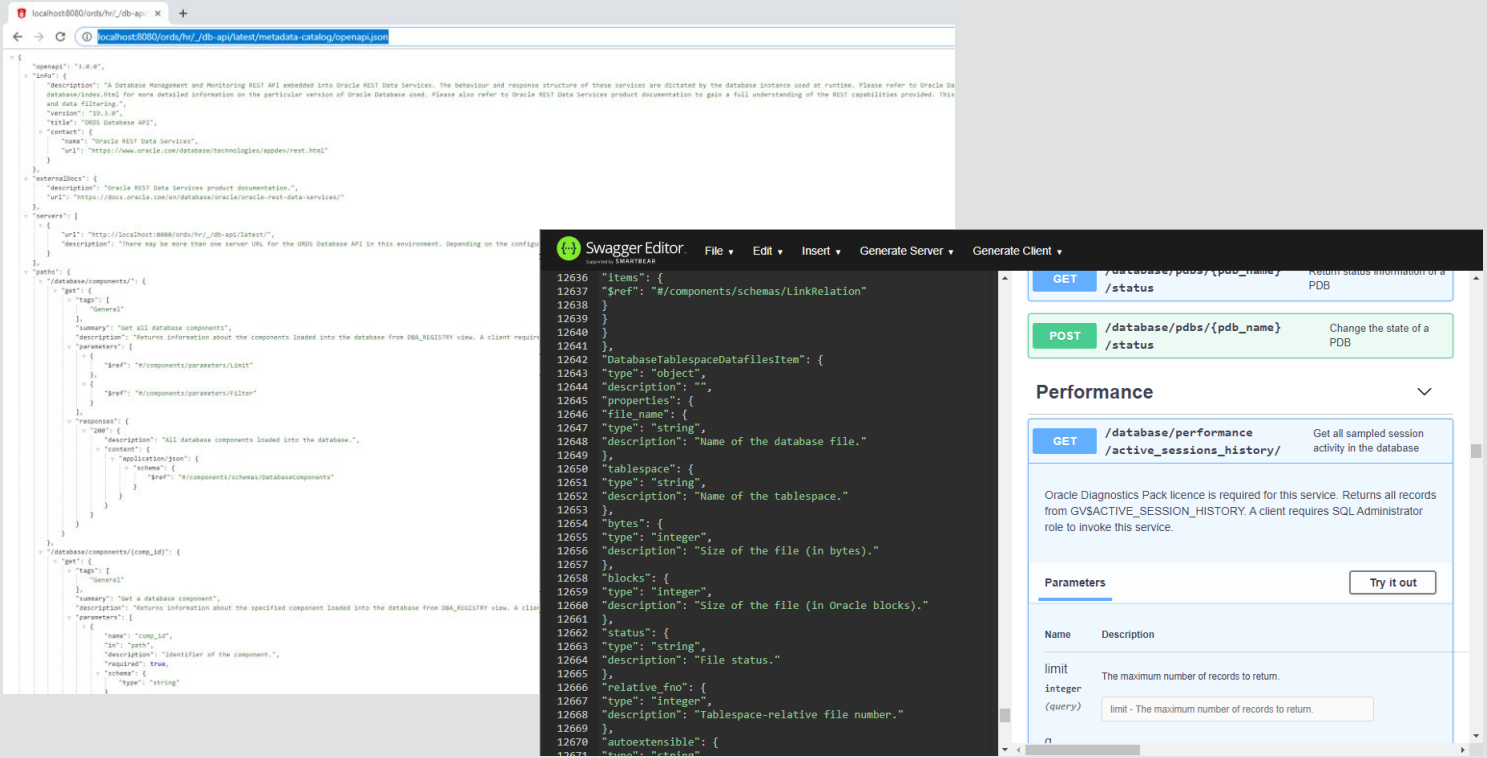
Option #2, DB User:

```
1  BEGIN -- block to be executed as HR
2      ords.enable_schema (
3          p_enabled               => true,
4          p_schema                => 'HR',
5          p_url_mapping_type      => 'BASE_PATH',
6          p_url_mapping_pattern   => 'hr',
7          p_auto_rest_auth        => true
8      );
9
10     COMMIT;
11 END;
12 /
13
14 GRANT dba TO hr; -- or PDB_DBA for 12c+
```

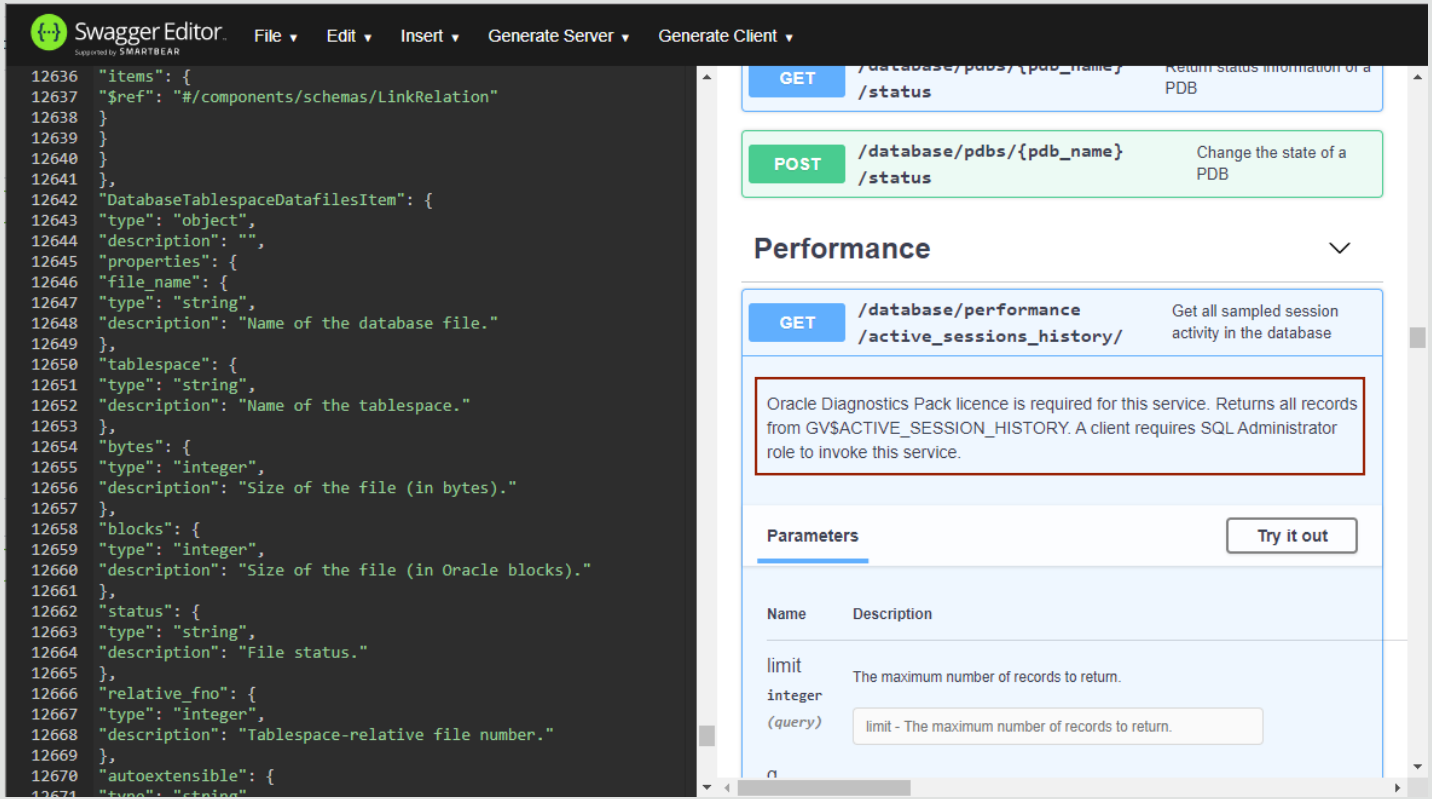
API Docs

- Generate it yourself
- Go to our copy on docs.oracle.com

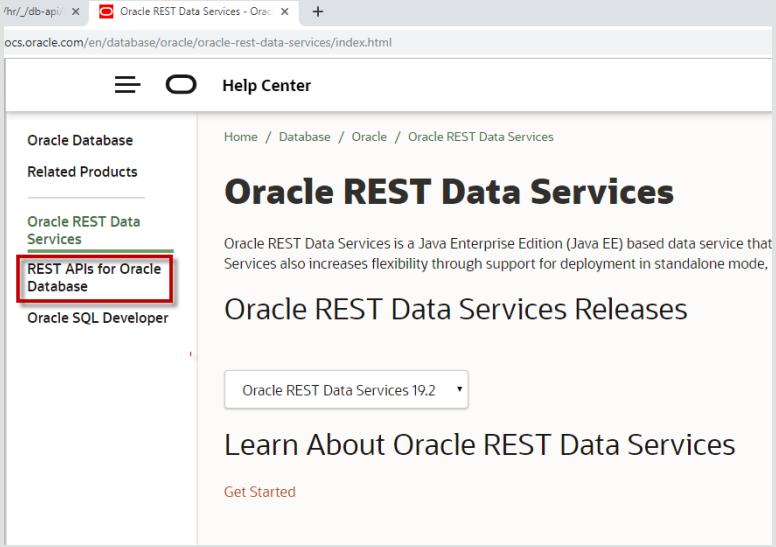
GET the Open API Docs {Swagger 3}



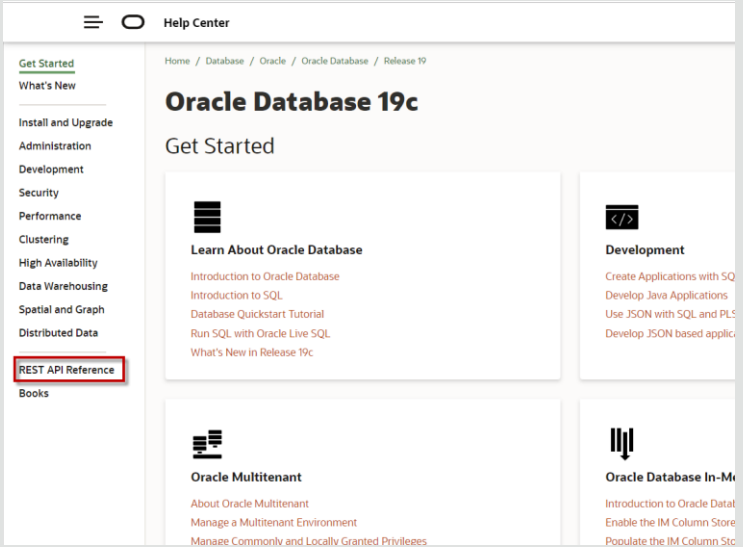
DB User Privs?



Oracle Docs



ORDS Docs



Database Docs

ORDS on YouTube

Other Resources

- SlideShare
- Blogs
- GitHub
- Articles

  UKOUG Scene Why REST, and What's in it or Me?
  Oracle Mag AUTO REST & REST Enabled SQL
- And don't forget Oracle-Base!

ThatJeffSmith