

# Oracle APEX 18.1 and the REST of the World

Marc Sewtz

Senior Software Development Manager  
Oracle America, Inc. – New York, NY







# Marc Sewtz

## Senior Software Development Manager Oracle Application Express / Database Tools

- Joined Oracle Consulting, Hamburg, Germany 1998
- Oracle Corp., New York, NY 1999 - today
- Built my first “APEX” Application in 2001
- Joined the APEX Development Team / Database Tools in 2002
  
- APEX on Twitter: #ORCLAPEX
- Twitter: [@msewtz](https://twitter.com/msewtz)
- LinkedIn: <http://www.linkedin.com/in/msewtz>
- Blog: <http://marcsewtz.blogspot.com>

## Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

- 1 Oracle APEX & Oracle REST Data Services
- 2 REST Enabled SQL in APEX 18.1
- 3 REST Service Consumption in APEX 18.1

- 1 Oracle APEX & Oracle REST Data Services
- 2 REST Enabled SQL in APEX 18.1
- 3 REST Service Consumption in APEX 18.1

# Oracle APEX

## Database-centric web application development framework



Rapid application development of database centric applications from your web browser



Visualize and maintain database data for desktop and mobile



Extending enterprise application solutions



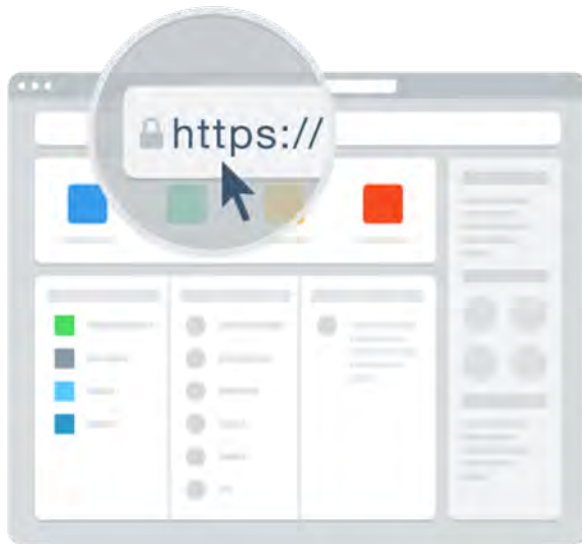
Use SQL and Oracle database features to develop apps



Modernizing legacy applications

# Oracle APEX

Database-centric web application development framework



App Development IDE is a web browser.

**No client software needed**



App definitions are stored in the database as meta data.

**Declarative – No code generation**

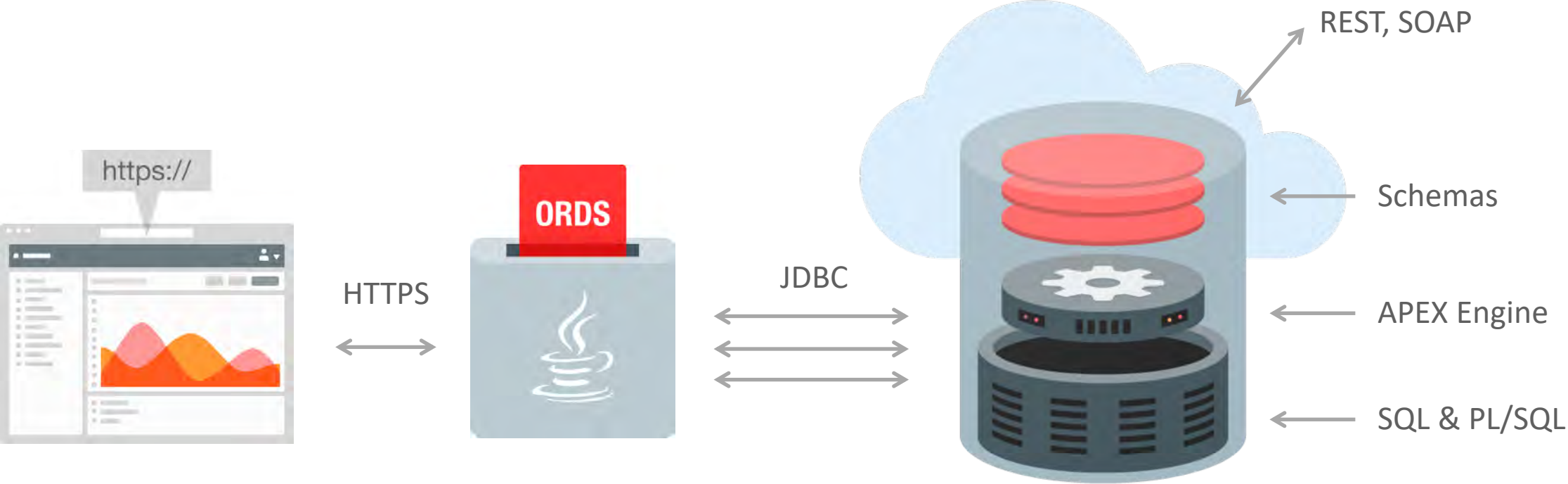


Page generation is efficient with only one request and one response.

**Data processing done in the Database**

# Oracle APEX & Oracle REST Data Services

## Database Centric Architecture



### Oracle REST Data Services

(Weblogic, Jetty, Tomcat)

*No Application Logic  
Converts HTTP to database API calls*

### Oracle Database

(Pluggable or Dedicated, 11g, 12c, 18c)  
XE, SE, EE, Exadata, RAC, DG, GG, IM...

*Zero latency database data access  
Dynamically driven by APEX metadata*



# Oracle REST Data Services (ORDS)

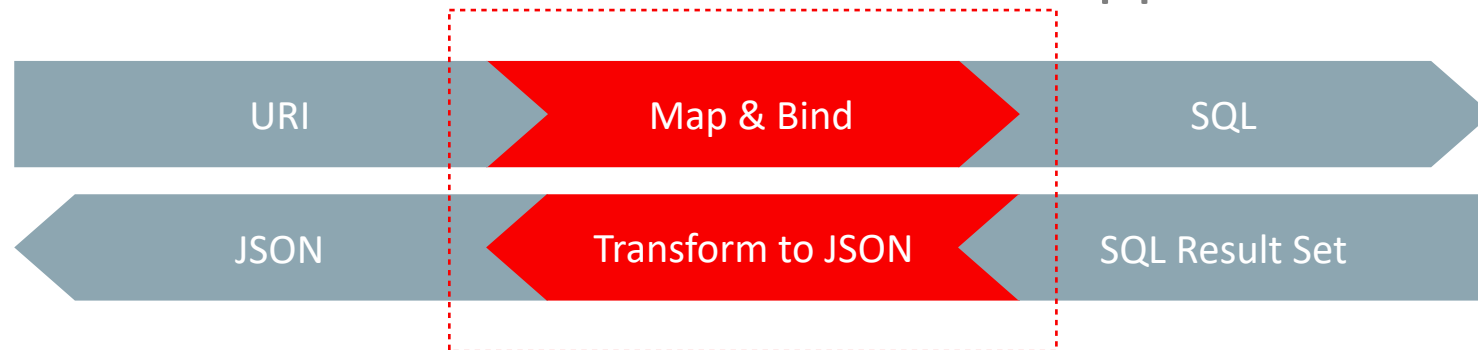
## Overview



- ORDS makes it easy to develop modern REST interfaces for relational data in the Oracle Database, JSON Document Store and Oracle NoSQL DB.
- ORDS maps HTTP(S) verbs (GET, POST, PUT, DELETE, etc.) to database transactions and returns any results formatted using JSON.
- ORDS is used as the Web listener for Oracle Application Express (APEX).



HTTP(S) client



Oracle REST Data Services



Oracle Database

# Oracle REST Data Services (ORDS)

## Overview



- Provides HTTP Access to Oracle Databases (and other databases)
  - Mid tier application
  - Maps http(s) RESTful Gets and Posts to SQL and PL/SQL
  - Declaratively returns results in JSON format
  - JavaScript friendly
  - Allows virtually every App Dev platform to access an Oracle Database
- Supported feature of the Oracle Database since 2010
- Ships with Oracle Database 12.1.0.2+
- Oracle APEX mid-tier, web toolkit applications, mod\_plsql replacement

# Oracle REST Data Services (ORDS)

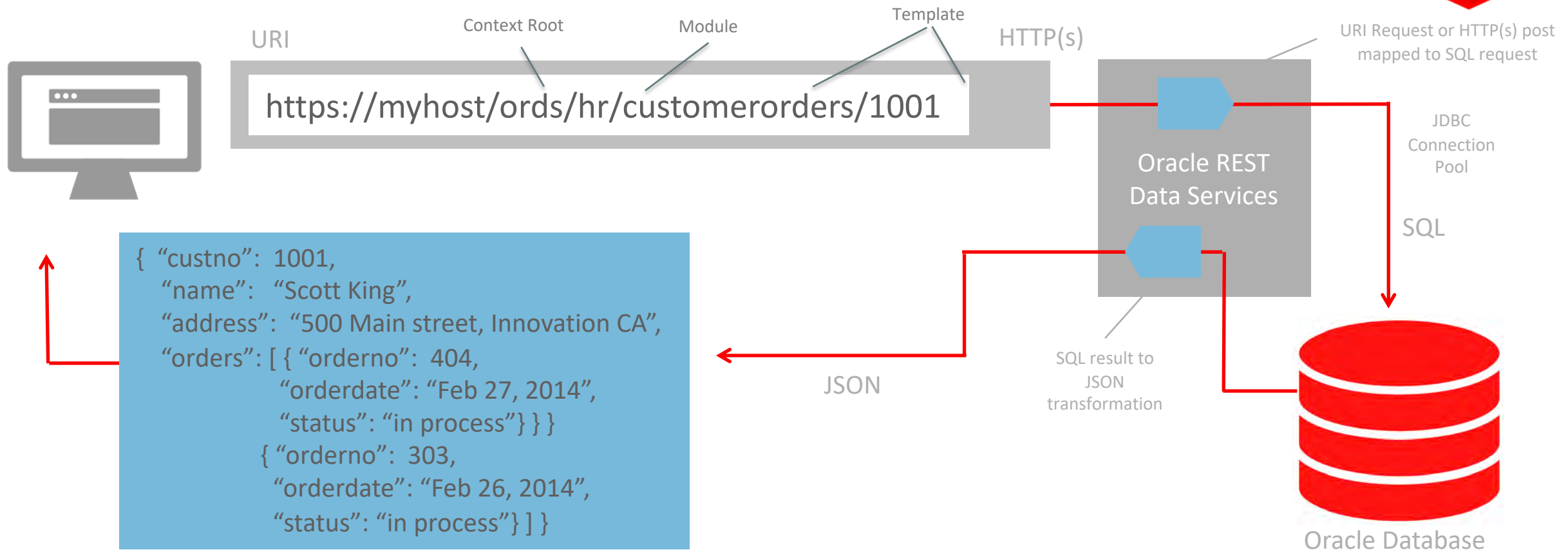
Supports all HTTP Verbs for interacting with all resources



Method	Purpose	Classification	Database Operation
GET	Retrieve resource	Safe, Idempotent	Select
HEAD	Retrieve metadata	Safe, Idempotent	Select
OPTIONS	Methods supported by resource	Safe, Idempotent	N/A
PUT	Create or replace resource	Idempotent	Merge, Update
DELETE	Delete resource	Idempotent	Delete
POST	Anything. Normally create	Unsafe	Insert

# Oracle REST Data Services (ORDS)

HTTP(s) API App-Dev with Relational Tables in Oracle Database



ORDS maps standard URI requests to corresponding relational SQL (not schemaless): e.g. SQL SELECT from customers and orders table.  
ORDS also transforms the SQL results into the highly popular JavaScript Object Notation (JSON), other formats include HTML, binary and CSV.  
Fully committed to supporting any and all standards required by Fusion / SaaS / FMW; we are actively engaged in the ongoing dialog.

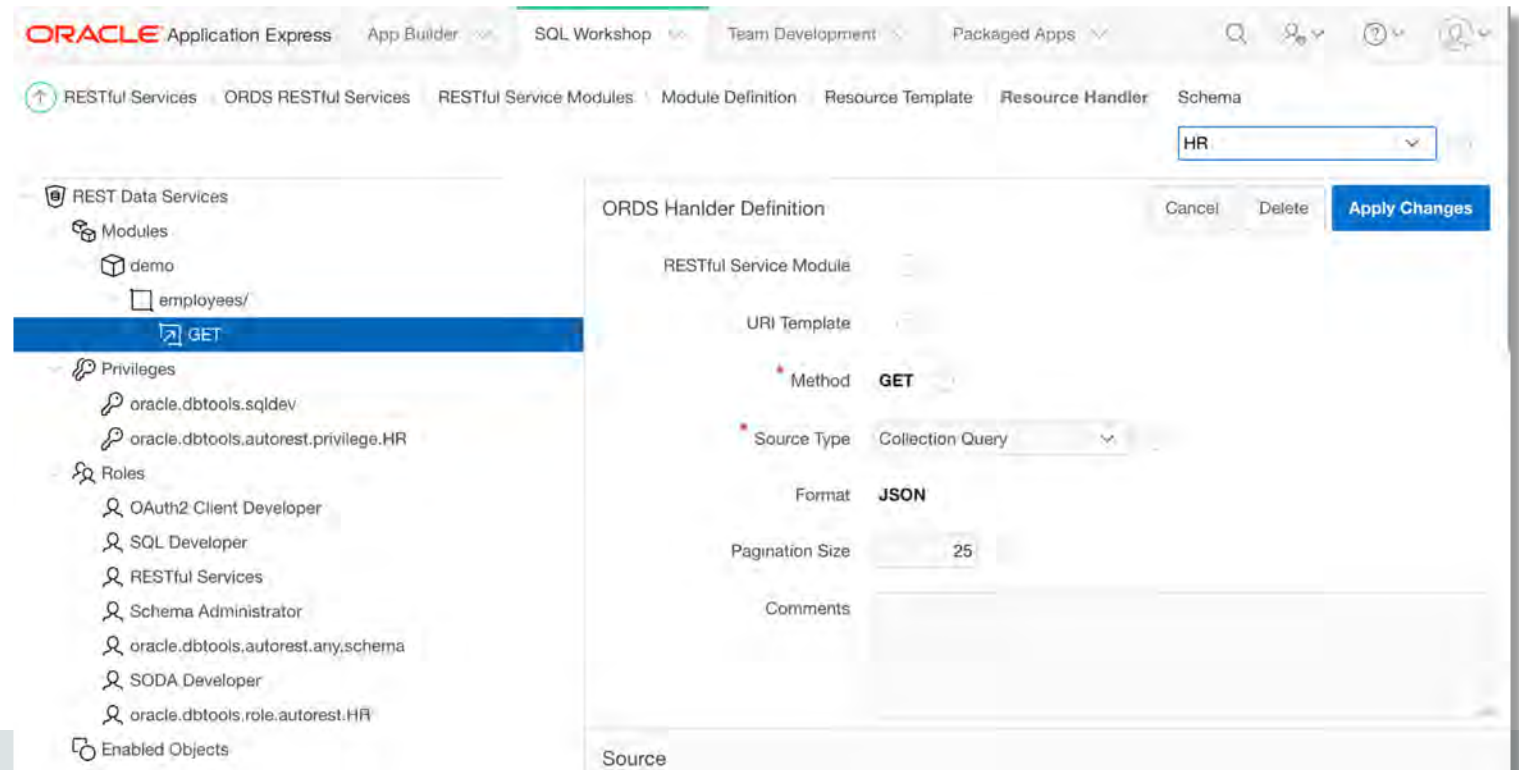


# New REST Workshop in Oracle APEX 18.1

Based on the ORDS Repository



- Declarative REST service support for ORDS-enabled REST services.
- New REST workshop will support REST web services developed using the ORDS repository.
- Provides super-set of functionality available with current repository.
- Existing APEX-based REST services will be migrated.



# Demo

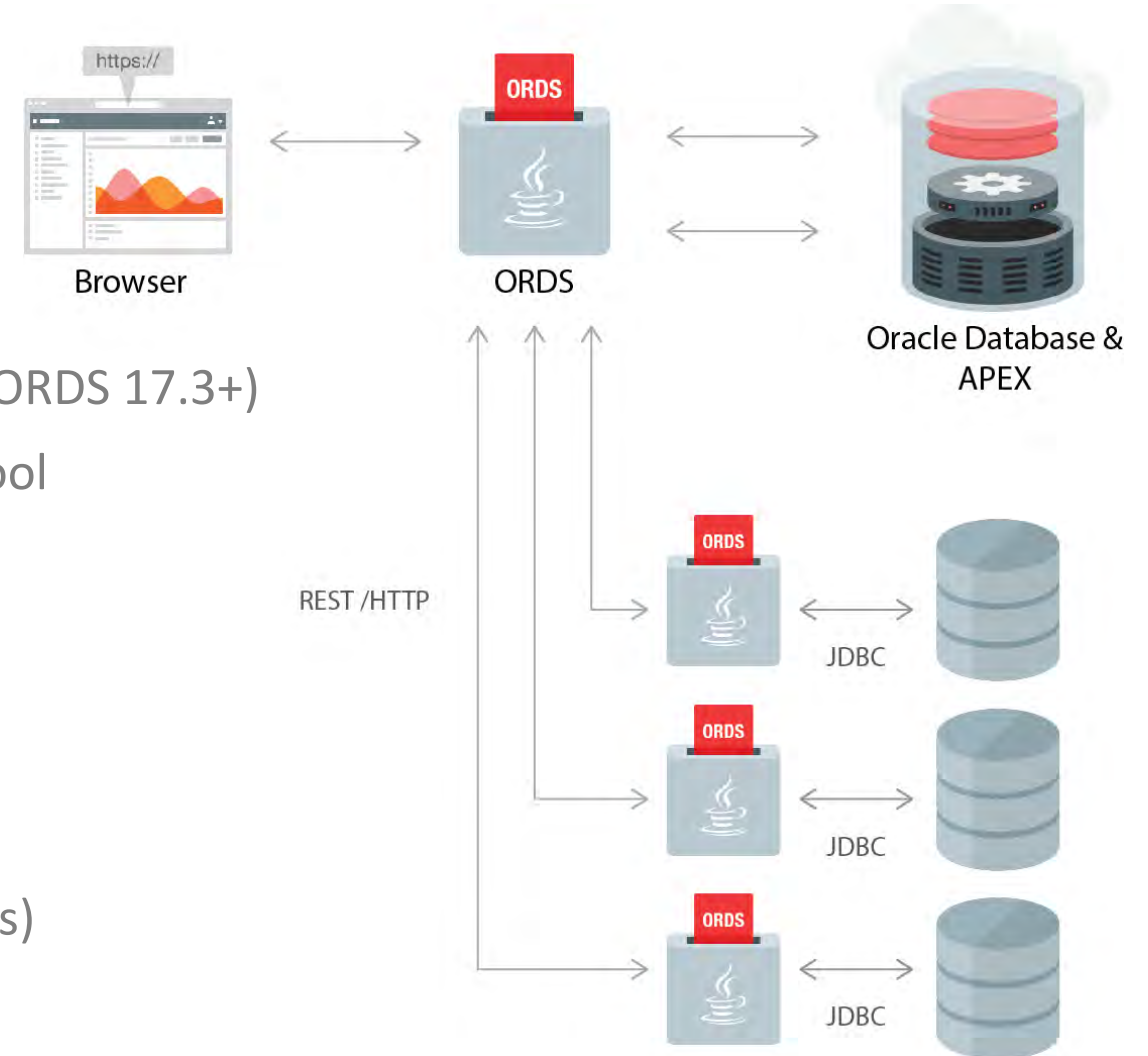
## Oracle REST Data Services

- 1 Oracle APEX & Oracle REST Data Services
- 2 REST Enabled SQL in APEX 18.1
- 3 REST Service Consumption in APEX 18.1

# REST Enabled SQL

## Remote Database Connections

- Executing dynamic SQL or PL/SQL on remote databases using ORDS and REST
  - Relies on the ORDS REST enabled SQL feature (ORDS 17.3+)
  - Requires ORDS instance having a connection pool configured for target database.
- APEX passes SQL query to ORDS over REST.
- Self-describing JSON response.
- Response contains a JSON object with:
  - Result set meta data (column names, data types)
  - The result data
  - Information about pagination





# REST Enabled SQL

## Remote Database Connections

- Based on SQL Developer Core SQL engine
- Needs to be enabled in defaults.xml configuration file:  

```
<entry key="restEnabledSql.active">true</entry>
```
- Once enabled, REST endpoint available for every REST enabled schema:

`http://localhost:8080/ords/hr/_/sql`

```
$ curl -X "POST" "http://localhost:9090/ords/hr/_/sql" \
      -H "Content-Type: application/sql" \
      -u HR:oracle \
      -d '$select * from emp; '
```

# REST Enabled SQL

## Parsing JSON and generating row / column result set

ORDS REST Enabled SQL returns a self-describing result set as follows:

```
{
  "items": [
    {
      "statement-id": 1,
      "statement-type": "query",
      "statement-pos": {
        "start-line": 1,
        "end-line": 1
      },
      "statement-text": "select 'hello world' as colname from dual",
      "metadata": [
        {
          "column-name": "COLNAME",
          "column-type-name": "CHAR",
          "precision": 11,
          "scale": 0,
          "is-nullable": 1
        }
      ]
    },
    ...
  ]
}
```

# REST Enabled SQL

## Parsing JSON and generating row / column result set

ORDS REST Enabled SQL returns a self-describing result set as follows:

```
...
    "result-set": {
      "items": [
        {
          "colname": "hello world"
        }
      ],
      "hasMore": false,
      "limit": 25,
      "offset": 0,
      "count": 1,
      "$asof": {"$scn": "1273919"}
    }
  ]
}
```

# Demo

Using ORDS REST Enabled SQL



# REST Enabled SQL & REST Service Consumption

## New Architecture in APEX 18.1

- Data source for APEX components:
  - Local Database
  - REST Enabled SQL
  - Web Source (REST Service)
- Database
  - Table / View (stores table and column information in meta data)
  - SQL Query
  - PL/SQL Function returning SQL Query

# REST Enabled SQL

## Parsing JSON and generating row / column result set

Building a "Data Parsing Profile":

```
select rownum col_sequence,  
       column_name,  
       col_selector,  
       precision,  
       scale,  
       data_type  
from json_table(  
    dynquery_rest.get_data,  
    '$.items[*].resultSet.metadata[*]'  
    columns (  
        column_name    varchar2(128)    path '$.columnName',  
        col_selector    varchar2(255)    path '$.jsonColumnName',  
        data_type       varchar2(30)     path '$.columnName',  
        precision       number          path '$.precision',  
        scale           number          path '$.scale'  
    )  
)
```

# REST Enabled SQL

## Parsing JSON and generating row / column result set

Building a "Data Parsing Profile":

```
select rownum col_sequence,  
       column_name,  
       col_selector,  
       precision,  
       scale,  
       data_type  
from json_table(  
    dynquery_rest.get_data,  
    '$.items[*].resultSet.metadata[*]'  
    columns (  
        column_name varchar2(128) pat  
        col_selector varchar2(255) pat  
        data_type    varchar2(30)  pat  
        precision    number        pat  
        scale        number        pat  
    )  
)
```

Column name	Col selector	Precision	Scale	Data type
SAL	\$.sal	7	2	NUMBER
MGR	\$.mgr	4	0	NUMBER
JOB	\$.job	9	0	VARCHAR2
HIREDATE	\$.hiredate	0	0	DATE
ENAME	\$.ename	10	0	VARCHAR2
EMPNO	\$.empno	4	0	NUMBER
DEPTNO	\$.deptno	2	0	NUMBER
COMM	\$.comm	7	2	NUMBER

# REST Enabled SQL

## Parsing JSON and generating row / column result set

Constructing a SQL query, based on data parsing profile, to return the actual data:

```
select
  j."EMPNO",
  j."ENAME",
  j."JOB",
  j."MGR",
  to_date( j."HIREDATE", 'YYYY-MM-DD"T"HH24:MI:SS"Z"' ) as "HIREDATE",
  j."SAL",
  j."COMM",
  j."DEPTNO"
from apex_collections c, json_table(
  c.clob001,
  '$.items[*].resultSet.items[*]'
  columns (
    "EMPNO"      NUMBER          path '$.empno',
    "ENAME"      VARCHAR2(10)    path '$.ename',
    "JOB"        VARCHAR2(9)     path '$.job',
    "MGR"        NUMBER          path '$.mgr',
    "HIREDATE"   VARCHAR2(4000)  path '$.hiredate',
    "SAL"        NUMBER          path '$.sal',
    "COMM"       NUMBER          path '$.comm',
    "DEPTNO"     NUMBER          path '$.deptno'
  )
) j
where c.collection_name = 'REST_COLLECTION'
```



# REST Enabled SQL

## Using Remote Database Connections in APEX

- Components like Reports, JET Charts, or Calendars will be able to use these extended Remote Databases as a data source.
- A Remote Server consists of:
  - A name
  - An endpoint URL
  - Authorization credential store
- Remote Databases configured via Shared Components.
- Remote Server and Credentials stored at Workspace level.

# REST Enabled SQL

## Using Remote Database Connections in APEX

- Select “REST Enabled SQL Service” as Data Source
- Choose Table or SQL Query as Source Type
- Initially available for Interactive and Classic Reports, Calendars and JET Charts

Create Interactive Report

Data Source: REST Enabled SQL Service

REST Enabled SQL Service: HR

Source Type:  Table  SQL Query

\* Table / View Owner: Current Schema

\* Table / View Name: EMPLOYEES

\* Select Columns

- EMPLOYEE\_ID (Number)
- FIRST\_NAME (Varchar2)
- LAST\_NAME (Varchar2)
- EMAIL (Varchar2)
- PHONE\_NUMBER (Varchar2)
- HIRE\_DATE (Date)
- JOB\_ID (Varchar2)
- SALARY (Number)
- COMMISSION\_PCT (Number)

# REST Enabled SQL

## Using Remote Database Connections in APEX

- View REST Requests in Debug output:  
*make\_rest\_request*

The screenshot displays the Oracle APEX interface. At the top, there is a search bar with a magnifying glass icon, a 'Go' button, and an 'Actions' dropdown menu. Below this is a table with the following columns: Employee Id, First Name, Last Name, Email, Phone Number, Hire Date, Job Id, and Salary. The table contains two rows of data:

Employee Id	First Name	Last Name	Email	Phone Number	Hire Date	Job Id	Salary
100	Steven	King	SKING	515.123.4567	17-JUN-1987	AD_PRES	24000
101	Neena	Kochhar	NKOCHHAR	515.123.4568	21-SEP-1997	AD_VP	17000

Below the table is a debug window titled 'Message'. It shows the execution details for a REST request. The 'Elapsed' time is 0.03898 and the 'Execution' time is 0.00026. The message content is as follows:

```
get_iv
make_rest_request p_url=>http://192.168.56.101:8080/ords/hr/_/sql,p_http_method=>POST,p_body=>
{"statementText":"begin begin\nexecute immediate q\u0027#alter session set
NLS_TERRITORY=\u0027AMERICAN\u0027#\u0027;\nexecute immediate q\u0027#alter session set
NLS_LANGUAGE=\u0027AMERICAN\u0027#\u0027;\nexecute immediate q\u0027#alter session set
NLS_DATE_FORMAT=\u0027DD-MON-YYYY\u0027#\u0027;\nexecute immediate q\u0027#alter session set
NLS_TIMESTAMP_FORMAT=\u0027DD-MON-YYYY\u0027#\u0027;\nexecute immediate q\u0027#alter session set
NLS_TIMESTAMP_TZ_FORMAT=\u0027DD-MON-YYYY\u0027#\u0027;\nexecute immediate q\u0027#alter session set
NLS_NUMERIC_CHARACTERS=\u0027.,\u0027#\u0027;\nend;\nbegin\nexecute immediate \u0027begin
sys.dbms_application_info.set_client_info(\u0027\u0027APEX using REST Enabled
SQL\u0027\u0027);\nsys.dbms_application_info.set_module(\u0027\u0027APEX:APP
121:3\u0027\u0027,\u0027\u0027PAGE 3\u0027\u0027);end;\u0027;exception when others then
null;end;\nend;\n\nselect i.*\n from(select * from (select
\"EMPLOYEE_ID\", \"FIRST_NAME\", \"LAST_NAME\", \"EMAIL\", \"PHONE_NUM~
```

# Demo

## REST Enabled SQL in APEX

- 1 Oracle APEX & Oracle REST Data Services
- 2 REST Enabled SQL in APEX 18.1
- 3 REST Service Consumption in APEX 18.1**

# REST Service Consumption

## Web Sources

- New data source type called **“Web Source”**
- Declarative method to define references to external REST APIs and generic JSON data feeds.
- Stores metadata about how to parse response data and map it as a virtual table with rows and columns.
- Web Sources will be stored on Workspace level and Shared Components.
- Usable as data sources for APEX components such as Interactive Grids, Reports, Charts, Forms, etc.





# REST Service Consumption

## Use Cases

- Access data from other Oracle Databases using REST
  - Similar use case to Remote SQL, however Remote SQL requires privileges to directly access tables on target database via SQL.
  - Many databases do not allow this type of access but do provide standardized REST services.
- Access internal systems (non Oracle DB) within an APEX application
  - Writing extensions to third-party in-house systems that provide REST APIs.
- Access Oracle SaaS functionality from within APEX applications.
  - Customers using Oracle Cloud SaaS applications use APEX to extend these systems.

# REST Service Consumption

## Use Cases

- Access external APIs (non Oracle Databases)
  - Augmenting APEX applications, while the processing happens on local tables.
  - Usage is likely to increase due to increasing public REST API availability and usage in business applications.
  - Similar functionality is provided today with the REST Client Assistant Packaged App, which allows for generating the SQL to fetch JSON data and to display it within a classic report.

# REST Service Consumption

## New Architecture in APEX 18.1

- Create Web Source in Shared Components
- Web Source Module references to one or more external Web Services
- Modules can contain one or many Operations, which are the references to a concrete external Web Service

The screenshot displays the 'Web Source Discovery' window. At the top, there are three green checkmarks and a 'Preview' button. Below is a table with the following columns: Agencyid, Color, Description, Id, Longname, Shortname, Textcolor, Type, and Url.

Agencyid	Color	Description	Id	Longname	Shortname	Textcolor	Type	Url
MTA NYCT	FAA61A	via Randall's Island / RFK Bridge	MTA NYCT_M35	Ward's Island - East Harlem	M35	FFFFFF	3	http://web.mta.info/ny
MTA NYCT	6CBE45	via Country Club Loop	MTA NYCT_BX24	Country Club - Hutchinson Metro Center	Bx24	FFFFFF	3	http://web.mta.info/ny
MTA NYCT	B933AD	via 57th St / York Av	MTA NYCT_M31	Yorkville - Clinton	M31	FFFFFF	3	http://web.mta.info/ny

Below the table, a preview window shows the JSON structure for a selected web source:

```
{  
  "agencyId": "MTA NYCT",  
  "color": "B933AD",  
  "description": "via Broadway / 42nd St",  
  "id": "MTA NYCT_M104",  
  "longName": "West Harlem - Times Square",  
  "shortName": "M104",  
  "textColor": "FFFFFF",  
  "type": 3,  
  "url": "http://web.mta.info/nyct/bus/schedule/manh/m104cur.pdf"  
},
```

# Demo

## REST Service Consumption

# Integrated Cloud

## Applications & Platform Services

ORACLE®