

HOL5 - ATP tooling

INDICE

HOL5 - ATP TOOLING	1
EJERCICIO 1: CREAR USUARIO DE MACHINE LEARNING	3
EJERCICIO 2: UTILIZAR SQL*DEVELOPER WEB	5
EJERCICIO 3: CREAR UN NOTEBOOK EN ORACLE MACHINE LEARNING.....	16
EJERCICIO 4: UTILIZACIÓN DE APEX.....	20
EJERCICIO 5 (OPCIONAL): CONFIGURACIÓN DE SEGURIDAD DE ACCESO OAUTH2.....	26



Ejercicio 1: Crear usuario de Machine Learning

En este ejercicio se explica crear un usuario de Oracle Machine Learning. Este usuario lo utilizaremos en ejercicios posteriores para crear un Notebook y consultar datos del esquema HR.
En la pantalla principal del ATP, elegir la pestaña “Tools”, y luego cliquar el botón “Open Oracle ML User Administration”

The screenshot shows the 'Autonomous Database Details' page for the database 'atplabpub'. At the top, there are several tabs: 'DB Connection', 'Performance Hub', 'Service Console', 'Scale Up/Down', and 'More Actions'. Below these, the 'Tools' tab is selected. Under the 'Tools' tab, there are four sections: 'SQL Developer Web', 'Oracle Application Express', 'Oracle ML User Administration', and 'SODA Drivers'. The 'Oracle ML User Administration' section is highlighted with a red box around its 'Open Oracle ML User Administration' button.

En la pantalla de login, entrar las **credenciales de ADMIN**:

Usuario: ADMIN

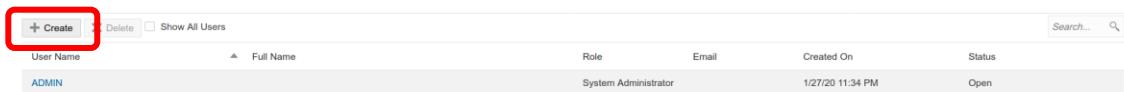
Contraseña: Autonomous#2020

The screenshot shows the 'SIGN IN' page for Oracle Machine Learning. It features a large blue header with a white cloud icon on the left and the word 'SIGN IN' in the center. Below the header, it says 'Database name: ATPLABPUB'. The main form area is titled 'Sign in with your Oracle Machine Learning Database Administrator credentials'. It has two input fields: 'USERNAME *' containing 'ADMIN' and 'PASSWORD *' containing a series of dots. At the bottom is a blue 'Sign In' button, which is highlighted with a red box.

En la pantalla siguiente, vemos un listado de **usuarios de OML**, solo ADMIN de momento.



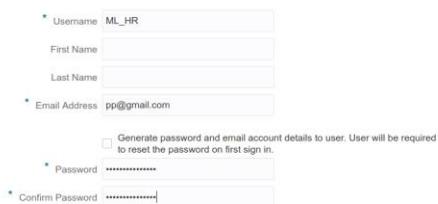
Users



User Name	Full Name	Role	Email	Created On	Status
ADMIN		System Administrator		1/27/20 11:34 PM	Open

Cliquear el botón “Create”.

Create User



Username: ML_HR
First Name:
Last Name:
Email Address: pp@gmail.com
 Generate password and email account details to user. User will be required to reset the password on first sign in.
Password:
Confirm Password:

En la pantalla de creación de usuario, llenar la información y pulsar el botón “Create”.

Usuario: ML_HR

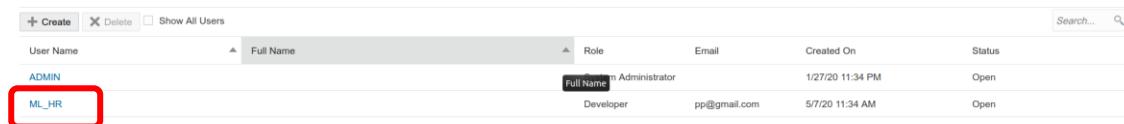
Password: Autonomous#2020

Email: cualquier valor con formato email.

Importante: de-chequear “Generate password and email account details to user”, para poder teclear la contraseña.

Una vez creado, el nuevo usuario aparece en la lista de usuarios de OML:

Users



User Name	Full Name	Role	Email	Created On	Status
ADMIN		Administrator		1/27/20 11:34 PM	Open
ML_HR		Developer	pp@gmail.com	5/7/20 11:34 AM	Open

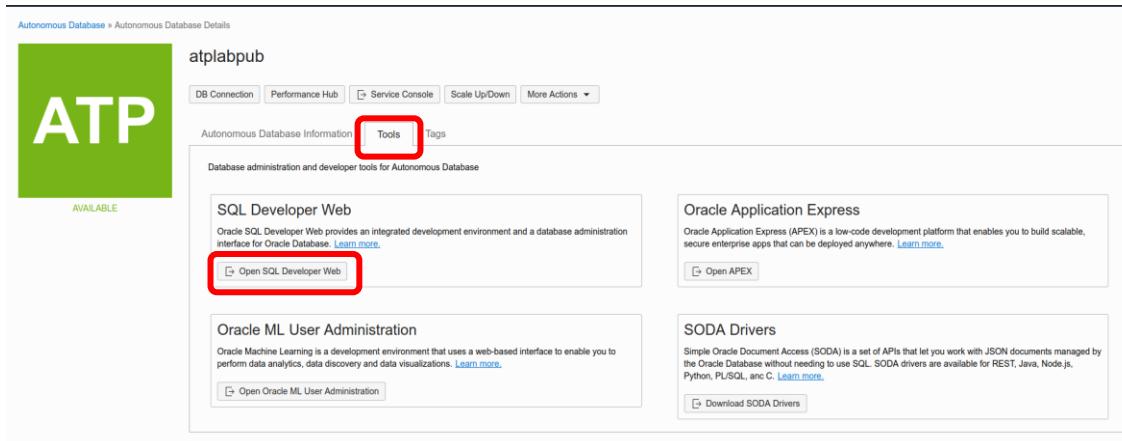


Ejercicio 2: Utilizar Sql*Developer Web

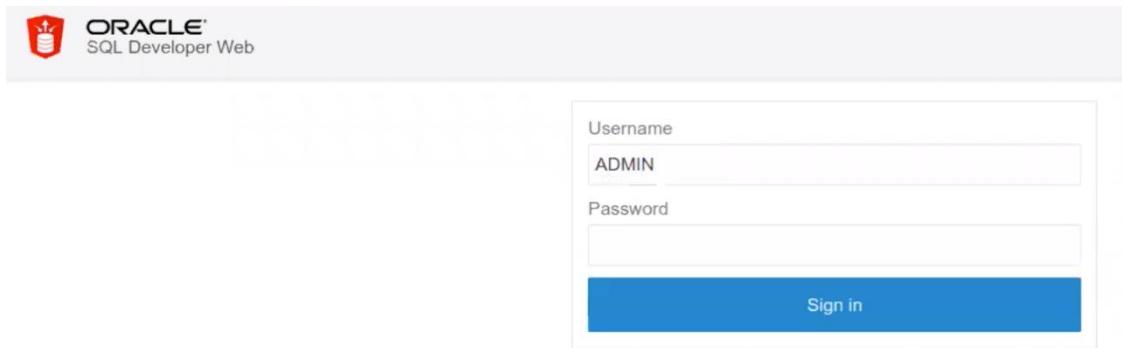
En este ejercicio, vamos a utilizar Sql*Developer Web para:

- Habilitar ORDS para el esquema HR
- Como HR, ejecutar algunas consultas SQL
- Habilitar una política de Data Redaction sobre un campo de la tabla “employees”
- Dar privilegios de consulta al usuario ML_HR sobre la tabla “employees”
- Habilitar ORDS sobre la tabla “employees”

Desde la pantalla principal del ATP, en la pestaña “Tools”, cliquar el botón “Open Sql Developer Web”.

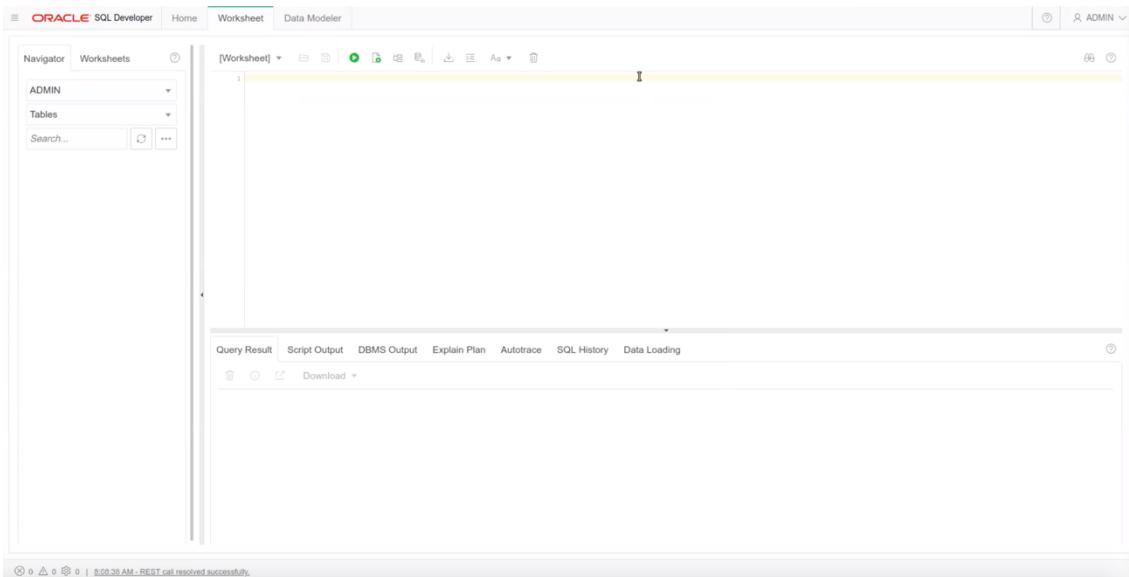


Esto nos lleva a la pantalla de login: conectarse como usuario ADMIN, password “Autonomous#2020”.



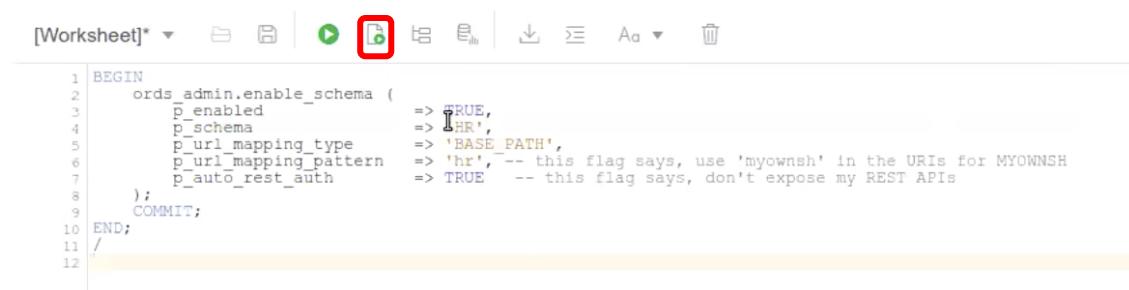
Una vez hecho esto, accedemos a la pantalla de **SQL Developer Web**, Que nos dará acceso para interactuar con la base de datos mediante SQL.





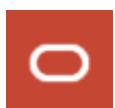
A continuación, ejecutamos el siguiente código, que habilita el usuario HR en ORDS y lo habilita para poder acceder con SQL Developer Web:

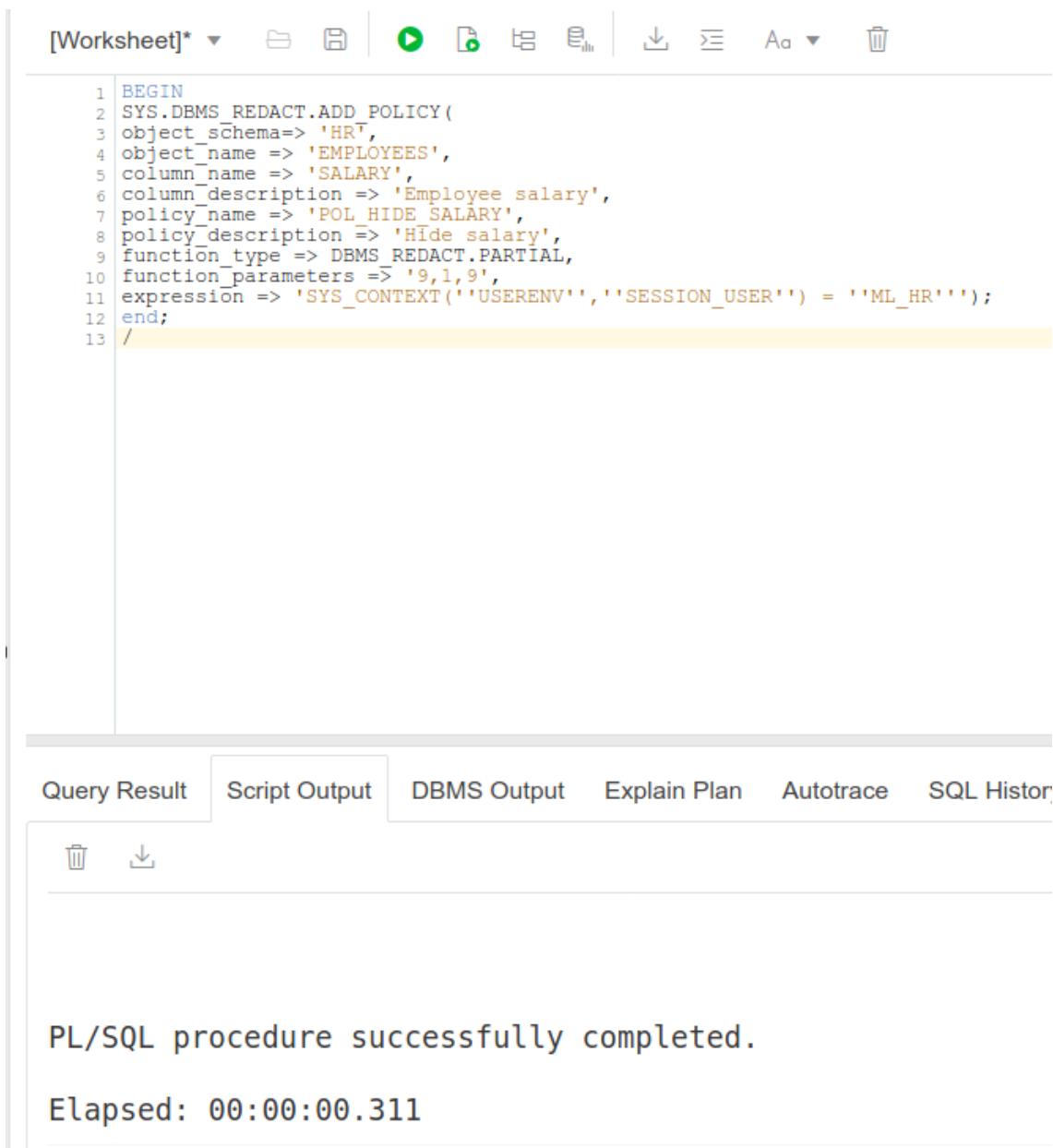
```
BEGIN
    ord.admin.enable_schema (
        p_enabled          => TRUE,
        p_schema            => 'HR',
        p_url_mapping_type => 'BASE_PATH',
        p_url_mapping_pattern => 'hr',
        p_auto_rest_auth   => TRUE );
    COMMIT;
END;
/
```



Luego, para preparar un ejercicio posterior, vamos a habilitar **Data Redaction** sobre la tabla “HR.employees”, para impedir que el usuario ML_HR pueda ver el contenido de la columna “salary”. Ejecutamos el código siguiente:

```
BEGIN
    SYS.DBMS_REDACT.ADD_POLICY(
        object_schema=> 'HR',
        object_name => 'EMPLOYEES',
        column_name => 'SALARY',
        column_description => 'Employee salary',
        policy_name => 'POL_HIDE_SALARY',
        policy_description => 'Hide salary',
        function_type => DBMS_REDACT.PARTIAL,
        function_parameters => '9,1,9',
        expression => 'SYS_CONTEXT(''USERENV'', ''SESSION_USER'') = ''ML_HR''');
    end;
/
```





The screenshot shows a SQL developer interface with a script editor and execution results.

Script Editor Content:

```
1 BEGIN
2   SYS.DBMS_REDACT.ADD_POLICY(
3     object_schema=> 'HR',
4     object_name => 'EMPLOYEES',
5     column_name => 'SALARY',
6     column_description => 'Employee salary',
7     policy_name => 'POL_HIDE_SALARY',
8     policy_description => 'Hide salary',
9     function_type => DBMS_REDACT.PARTIAL,
10    function_parameters => '9,1,9',
11    expression => 'SYS_CONTEXT(''USERENV'', ''SESSION_USER'') = ''ML_HR''');
12  end;
13 /
```

Execution Results:

PL/SQL procedure successfully completed.

Elapsed: 00:00:00.311

Esta política impedirá que el usuario ML_HR pueda ver el contenido del campo “salary” en la tabla “HR.employees”.

Una vez hecho esto, podemos acceder a [SQL Developer Web](#) mediante la URL anterior, pero cambiando el usuario admin por HR.

Primero nos desconectamos del Sql*Developer Web, y cerramos la pestaña del navegador.



```

1 BEGIN
2   ORDS_ADMIN.grant_privileges(
3     p_grantee          => 'TRUE',
4     p_grantor          => 'HR',
5     p_grant_type        => 'CREATE_TABLE',
6     p_grant_on_schema   => 'TRUE',
7     p_grant_on_table    => 'TRUE',
8     p_grant_on_column   => 'TRUE' );
9  END;
10 /

```

PL/SQL procedure successfully completed.
Elapsed: 00:00:00.409

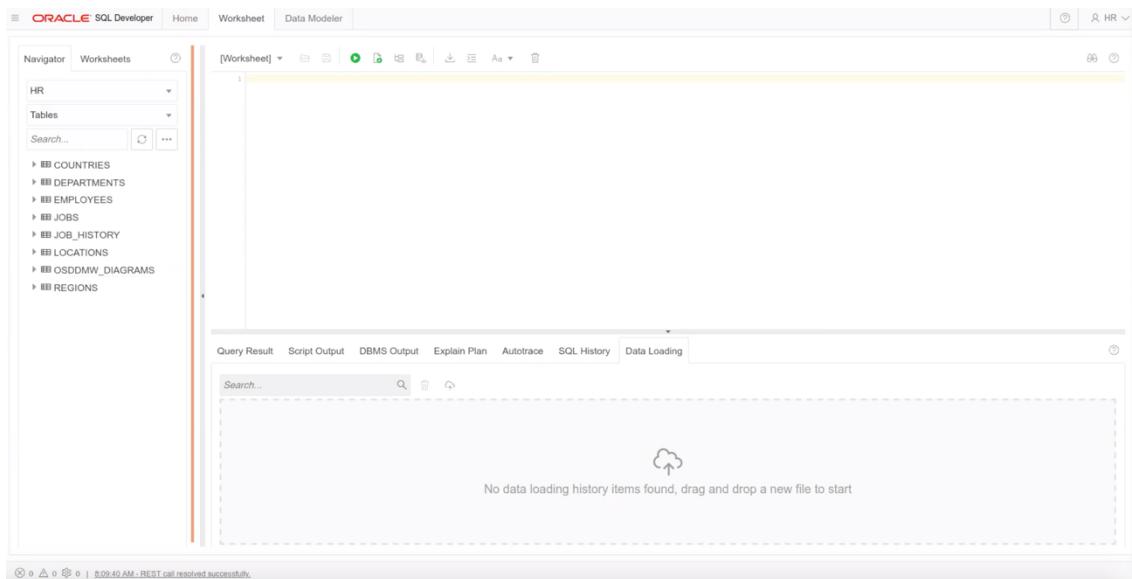
Luego desde la pagina principal del ATP, pestaña “**Tools**”, volvemos a cliquar “**Open Sql Developer Web**”, para volver a la pantalla de login. En la URL, **cambiamos “admin” por “hr”**:



Volverá a aparecer la consola de login, volvemos a introducir el nombre de usuario y contraseña. En este caso el usuario HR/hr o hr/hr (**contraseña siempre en minúsculas**).

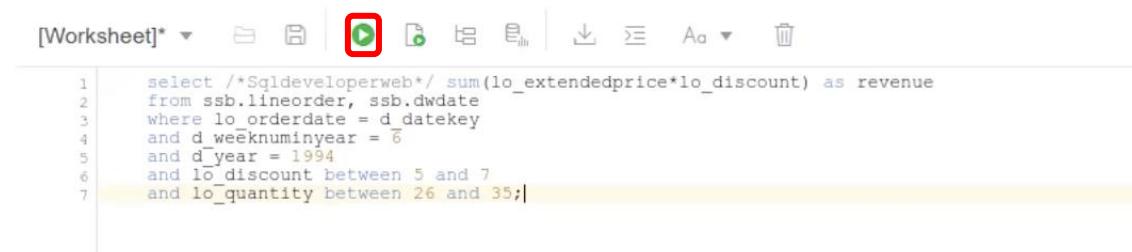
En la siguiente pantalla podemos ver la misma consola de SQL Developer Web, pero en este caso a la izquierda podemos ver las tablas del esquema HR:





Ejecutamos el siguiente código, como se muestra en la imagen. Es una query sobre el **esquema SSB**, accesible a cualquier usuario:

```
select /*Sqldeveloperweb*/ sum(lo_extendedprice*lo_discount) as revenue
from ssb.lineorder, ssb.dwdate
where lo_orderdate = d_datekey
and d_weeknuminyear = 6
and d_year = 1994
and lo_discount between 5 and 7
and lo_quantity between 26 and 35;
```



Se puede consultar la ejecución de la query desde la sección Performance HUB, en la pantalla principal del ATP:



Una vez aquí, se puede ver la query ejecutándose:

ASH Analytics SQL Monitoring

Top 100 by Last Active Time

Kill Session

Status	Duration	Inst ID	SQL ID	SQL Plan Hash	User Name	Parallel	Database Time	I/O Requests	SQL Text
Executing	20.00s	1	228q0dh6lrmz	817007416	HR@Z36W1ZYAC6L8LPZ_ATPLABPUB	1	20.40s	71K	select * from i select q_ ... row_number() over ...

Se puede seleccionar el SQL ID de la query, y acceder a sus detalles

atplabpub

← Back

Real-time SQL Monitoring

SQL ID: 228q0dh6lrmz Execution ID: 16777216 Status: Executing

Save Report Refresh

Overview

General

SQL Text: select * from i select q_ ... row_number() over ...

Execution Started: April 7, 2020 8:10:08 AM

Last Refresh Time: April 7, 2020 8:10:36 AM

Execution Id: 16777216

User Name: HR@Z36W1ZYAC6L8LPZ_ATPLAB...

Fetch Calls: 0

Time & Wait

Duration	Time
28.0s	28.6s

I/O

Buffer	Gets	IOs	Requests	Bytes
13M	99K	96.3GB	100%	Call Offload Efficiency: 100%

Más abajo, se puede ver el código de la query, en la pestaña SQL Text:

Details

Plan Statistics SQL Text Activity Metrics

SQL Binds

SQL Text

```
select * from
(select q_.*, row_number() over (order by 1) RN_
from (
select q_.*, l_extendedprice*l_discount as revenue
from ssa.lineorder, ssa.dates
where ls_orderdate = d_datekey
and ls_suppkey = s_suppkey
and d_year = 1994
and ls_discount between 5 and 7
and ls_quantity between 26 and 35
) q_
where RN_ between :1 and :2
)
```



Volviendo a SQL Developer Web, cuando la query haya terminado, podemos ver el resultado de la consulta:

The screenshot shows the Oracle SQL Developer Web interface. In the top navigation bar, 'Worksheet' is selected. The left sidebar shows a tree view of tables under the 'HR' schema, including COUNTRIES, DEPARTMENTS, EMPLOYEES, JOBS, JOB_HISTORY, LOCATIONS, OSDDMW_DIAGRAMS, and REGIONS. The main workspace contains a SQL query in the 'Worksheet' tab:

```
1 select /*+index(developer, statistic_extendedprice)*/ l.revenue
  2   from developer.dates d
  3      ,developer.products p
  4      ,developer.suppliers s
  5      ,developer.orders o
  6      ,developer.order_items i
  7      ,developer.line_items l
  8     where l.orderdate = d.datekey
  9       and d.weekendyear = 8
 10      and p.product_id = i.product_id
 11      and l.discount between 3 and 7
 12      and l.quantity between 20 and 30;
```

The results are displayed in the 'Query Result' tab, showing one row of data:

revenue
2611320600347

Execution time: 48.562 seconds.

A continuación, ejecutamos una segunda consulta, en este caso la consulta devuelve un **objeto JSON** a partir de los datos de la consulta SQL:

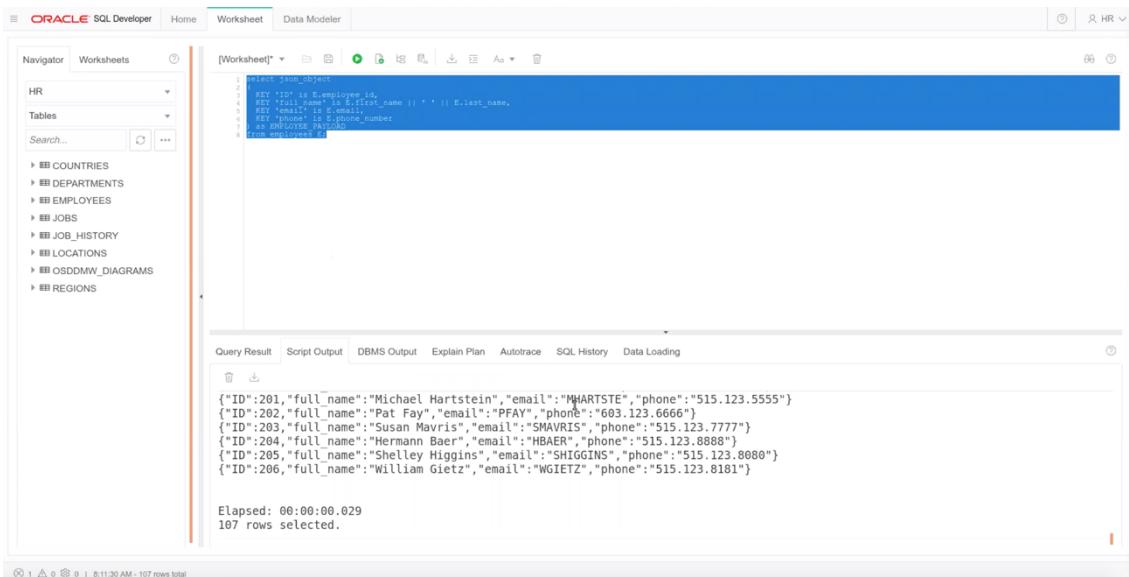
```
select json_object
(
  KEY 'ID' is E.employee_id,
  KEY 'full_name' is E.first_name || ' ' || E.last_name,
  KEY 'email' is E.email,
  KEY 'phone' is E.phone_number
) as EMPLOYEE_PAYLOAD
from employees E;
```

The screenshot shows the Oracle SQL Developer Web interface with the same layout as the previous one. The main workspace contains the same SQL query as above:

```
1 select json_object
  2  (
  3    KEY 'ID' is E.employee_id,
  4    KEY 'full_name' is E.first_name || ' ' || E.last_name,
  5    KEY 'email' is E.email,
  6    KEY 'phone' is E.phone_number
  7  ) as EMPLOYEE_PAYLOAD
  8 from employees E;
```



Podemos ver el resultado en formato JSON:



The screenshot shows the Oracle SQL Developer interface. In the top navigation bar, 'ORACLE SQL Developer' is selected. Below it, the 'Worksheet' tab is active. On the left, the 'Navigator' pane shows the 'HR' schema with tables like COUNTRIES, DEPARTMENTS, EMPLOYEES, JOBS, JOB_HISTORY, LOCATIONS, OSDDMW_DIAGRAMS, and REGIONS. The main workspace displays a query in the 'Worksheet' tab:

```
SELECT json_object(
    KEY 'ID' IS employee_id,
    KEY 'full_name' IS E.first_name || ' ' || E.last_name,
    KEY 'email' IS E.email,
    KEY 'phone' IS E.phone,
    KEY 'emp_no' IS E.emp_no
) AS EMPLOYEE_PAYLOAD
FROM employees E;
```

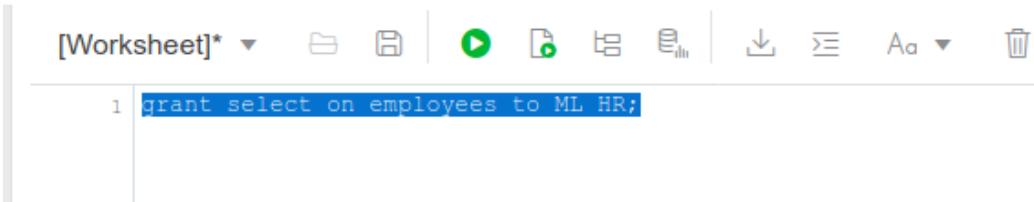
Below the query, the 'Query Result' tab is selected, showing the output in JSON format:

```
[{"ID":201,"full_name":"Michael Hartstein","email":"MARTSTE","phone":"515.123.5555"}, {"ID":202,"full_name":"Pat Fay","email":"PFAY","phone":"603.123.6666"}, {"ID":203,"full_name":"Susan Mavris","email":"SMAVRIS","phone":"515.123.7777"}, {"ID":204,"full_name":"Hermann Baer","email":"HBAERI","phone":"515.123.8888"}, {"ID":205,"full_name":"Shelley Higgins","email":"SHIGGINS","phone":"515.123.8680"}, {"ID":206,"full_name":"William Gietz","email":"WGIETZ","phone":"515.123.8181"}]
```

At the bottom, it says 'Elapsed: 00:00:00.029' and '107 rows selected.'

Para preparar los ejercicios siguientes, otorgamos privilegios al **usuario ML_HR** sobre la tabla “employees”:

```
grant select on employees to ML_HR;
```



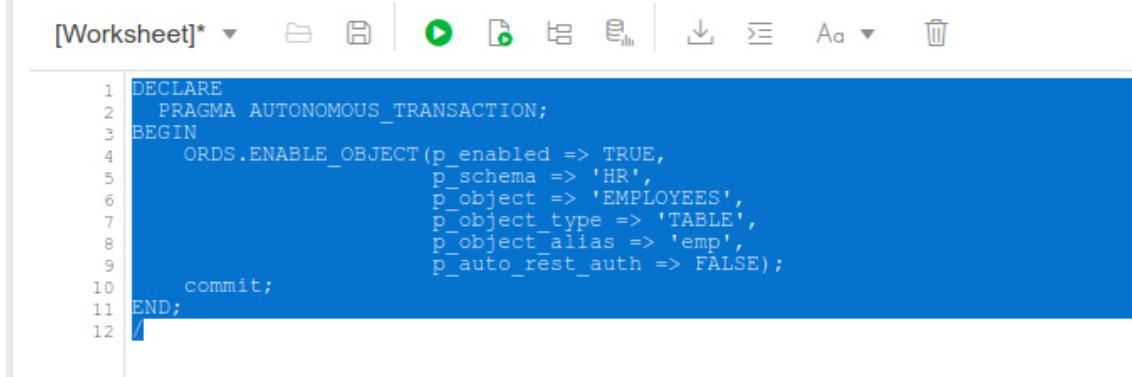
The screenshot shows the Oracle SQL Developer interface with the 'Worksheet' tab active. The query window contains the following command:

```
1 grant select on employees to ML_HR;
```

Y habilitamos ORDS sobre la tabla “employees”, para permitir el acceso por REST a sus datos:

```
DECLARE
  PRAGMA AUTONOMOUS_TRANSACTION;
BEGIN
  ORDS.ENABLE_OBJECT(p_enabled => TRUE,
                      p_schema => 'HR',
                      p_object => 'EMPLOYEES',
                      p_object_type => 'TABLE',
                      p_object_alias => 'emp',
                      p_auto_rest_auth => FALSE);
  commit;
END;
/
```



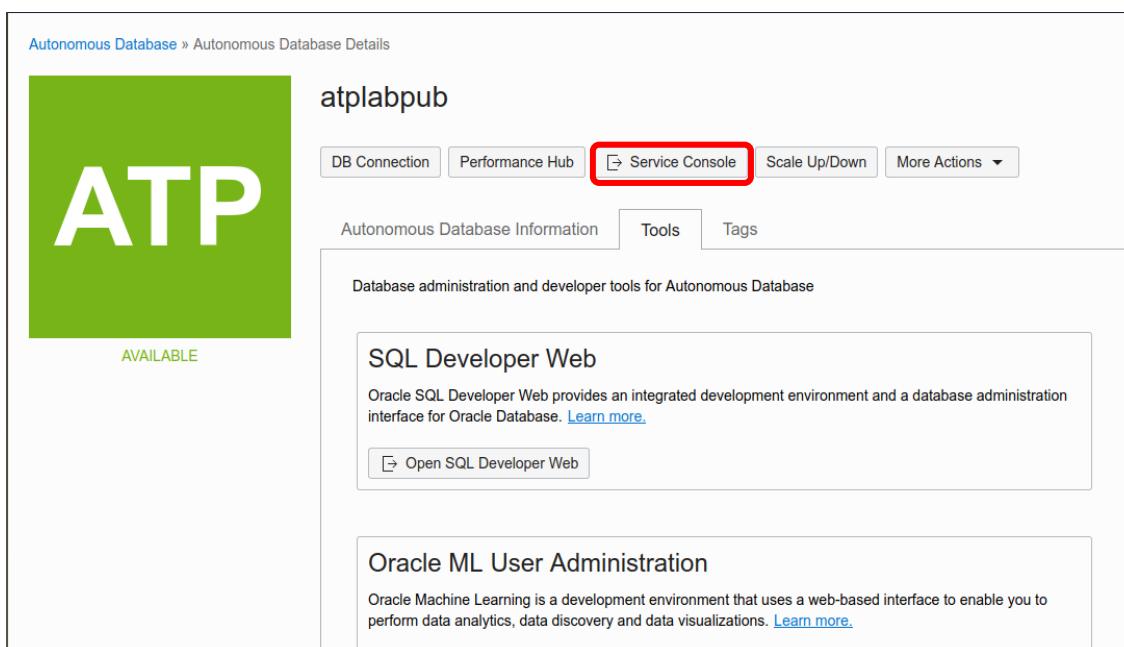


```

1  DECLARE
2      PRAGMA AUTONOMOUS_TRANSACTION;
3  BEGIN
4      ORDS.ENABLE_OBJECT(p_enabled => TRUE,
5                          p_schema => 'HR',
6                          p_object => 'EMPLOYEES',
7                          p_object_type => 'TABLE',
8                          p_object_alias => 'emp',
9                          p_auto_rest_auth => FALSE);
10     commit;
11 END;
12 /

```

Ahora podremos consultar la tabla “employees” mediante REST API. Para ello recuperamos el REST Endpoint desde la pagina principal del ATP. Hacer click en el botón “Service Console”:



Autonomous Database » Autonomous Database Details

atplabpub

DB Connection Performance Hub **Service Console** Scale Up/Down More Actions ▾

Autonomous Database Information Tools Tags

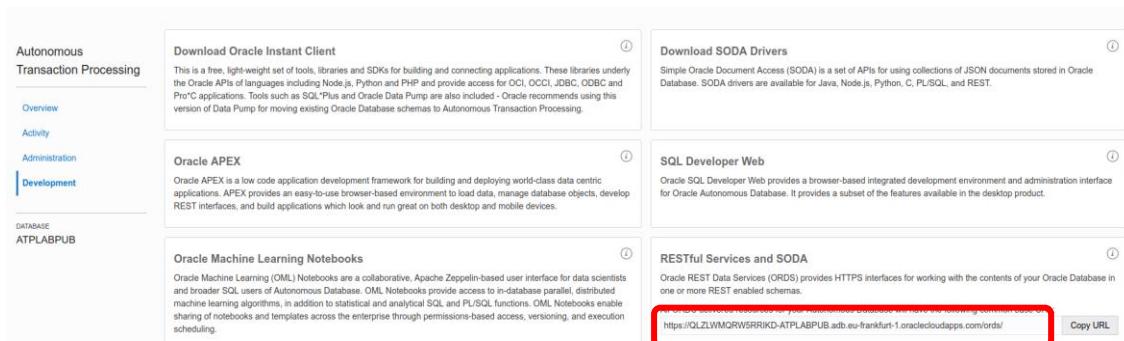
Database administration and developer tools for Autonomous Database

SQL Developer Web
Oracle SQL Developer Web provides an integrated development environment and a database administration interface for Oracle Database. [Learn more.](#)

[Open SQL Developer Web](#)

Oracle ML User Administration
Oracle Machine Learning is a development environment that uses a web-based interface to enable you to perform data analytics, data discovery and data visualizations. [Learn more.](#)

En la pantalla siguiente, hacer click en “Development”. En el apartado “**RESTful Services and SODA**”, vemos nuestro **REST Endpoint**:



Autonomous Transaction Processing

Overview

Activity

Administration

Development

DATABASE ATPLABPUB

Download Oracle Instant Client
This is a free, light-weight set of tools, libraries and SDKs for building and connecting applications. These libraries underly the Oracle APIs of languages including Node.js, Python and PHP and provide access for OCI, OCCI, JDBC, ODBC and Pro*C applications. Tools such as SQL*Plus and Oracle Data Pump are also included - Oracle recommends using this version of Data Pump for moving existing Oracle Database schemas to Autonomous Transaction Processing.

Download SODA Drivers
Simple Oracle Document Access (SODA) is a set of APIs for using collections of JSON documents stored in Oracle Database. SODA drivers are available for Java, Node.js, Python, C, PL/SQL, and REST.

Oracle APEX
Oracle APEX is a low code application development framework for building and deploying world-class data centric applications. APEX provides an easy-to-use browser-based environment to load data, manage database objects, develop REST interfaces, and build applications which look and run great on both desktop and mobile devices.

SQL Developer Web
Oracle SQL Developer Web provides a browser-based integrated development environment and administration interface for Oracle Autonomous Database. It provides a subset of the features available in the desktop product.

Oracle Machine Learning Notebooks
Oracle Machine Learning (ML) Notebooks are a collaborative, Apache Zeppelin-based user interface for data scientists and broader SQL users of Autonomous Database. ML Notebooks provide access to in-database parallel, distributed machine learning algorithms, in addition to statistical and analytical SQL and PL/SQL functions. ML Notebooks enable sharing of notebooks and templates across the enterprise through permissions-based access, versioning, and execution scheduling.

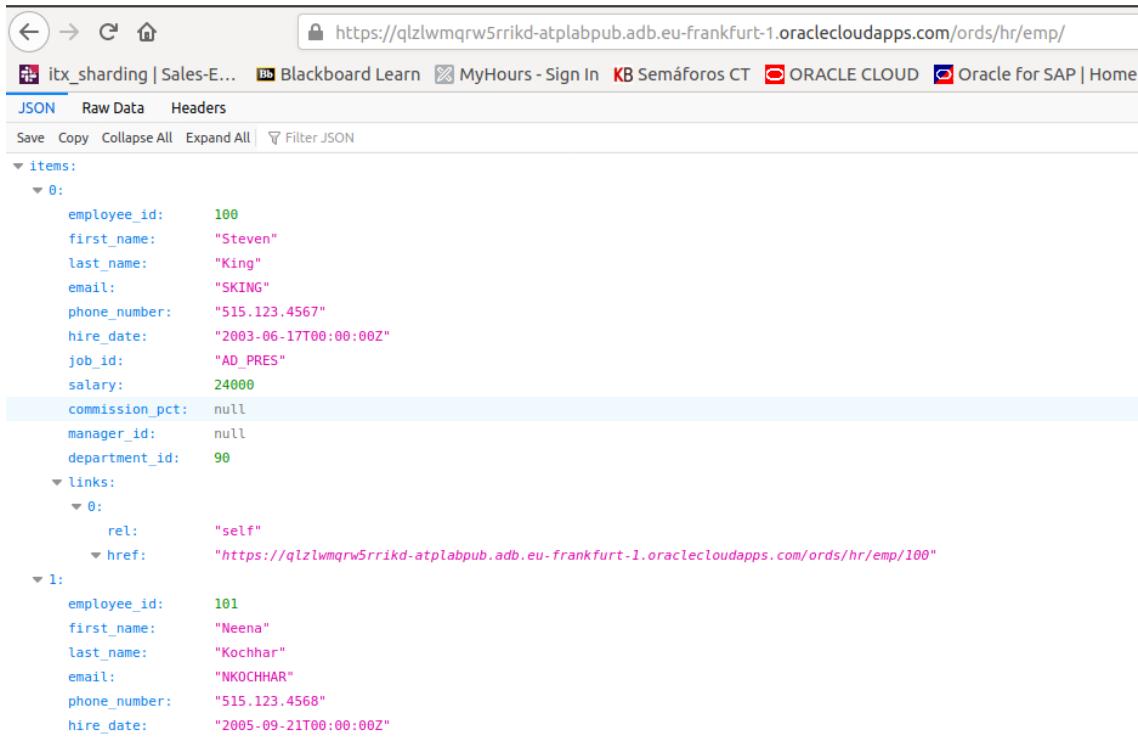
RESTful Services and SODA
Oracle REST Data Services (ORDS) provides HTTPS interfaces for working with the contents of your Oracle Database in one or more REST-enabled schemas.

<https://QLZLWMQRW5RRIKD-ATPLABPUB.adb.eu-frankfurt-1.oraclecloudapps.com/ords/> [Copy URL](#)

Para consultar nuestra tabla, a la URL de REST Endpoint le añadimos “[/hr/emp/](#)”, por ejemplo:
<https://QLZLWMQRW5RRIKD-ATPLABPUB.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/emp/>



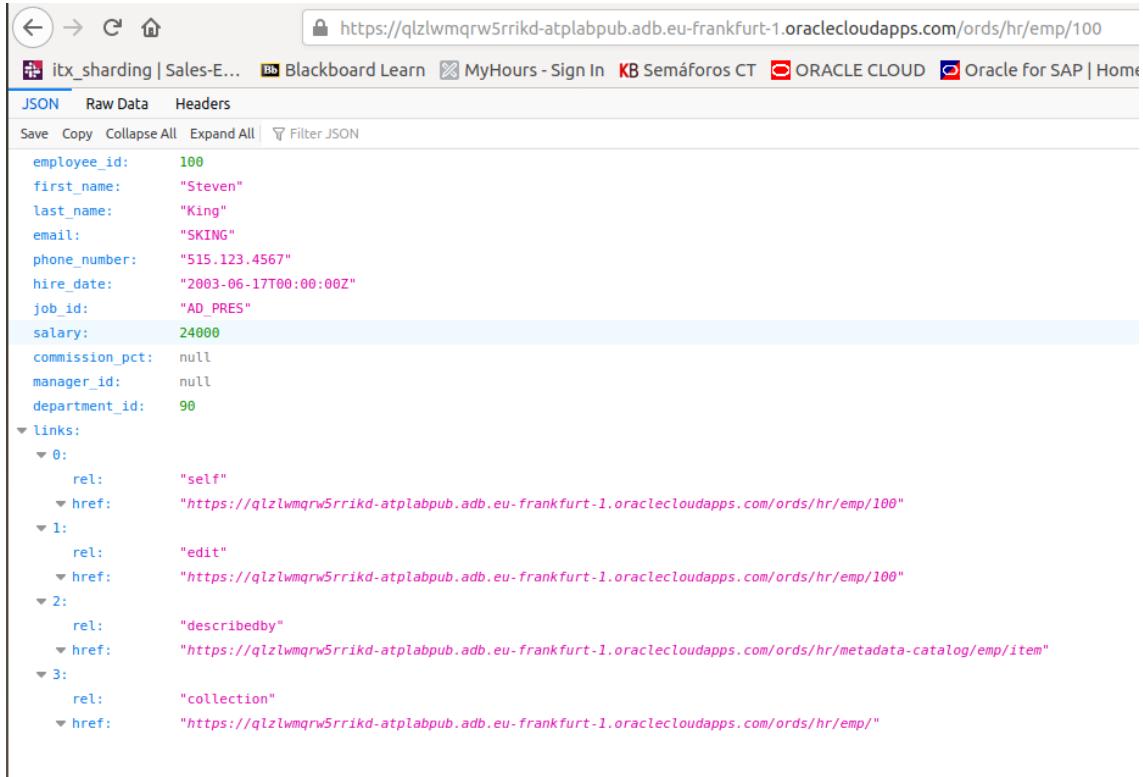
Si pegamos esta URL en un navegador Web, vemos los datos de la tabla “employees”:



The screenshot shows a browser window with the URL <https://qlzlwmqrw5rrikd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/emp/>. The page displays a JSON response with two items (employees). Item 0 has employee_id 100, first_name "Steven", last_name "King", email "SKING", phone_number "515.123.4567", hire_date "2003-06-17T00:00:00Z", job_id "AD_PRES", salary 24000, commission_pct null, manager_id null, and department_id 90. It also contains a link to its own detail page. Item 1 has employee_id 101, first_name "Neena", last_name "Kochhar", email "NKOCHHAR", phone_number "515.123.4568", hire_date "2005-09-21T00:00:00Z", and other details.

```
items:
  0:
    employee_id: 100
    first_name: "Steven"
    last_name: "King"
    email: "SKING"
    phone_number: "515.123.4567"
    hire_date: "2003-06-17T00:00:00Z"
    job_id: "AD_PRES"
    salary: 24000
    commission_pct: null
    manager_id: null
    department_id: 90
  links:
    0:
      rel: "self"
      href: "https://qlzlwmqrw5rrikd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/emp/100"
  1:
    employee_id: 101
    first_name: "Neena"
    last_name: "Kochhar"
    email: "NKOCHHAR"
    phone_number: "515.123.4568"
    hire_date: "2005-09-21T00:00:00Z"
```

Si queremos ver únicamente el employee ID=100, completamos la URL con “100”:

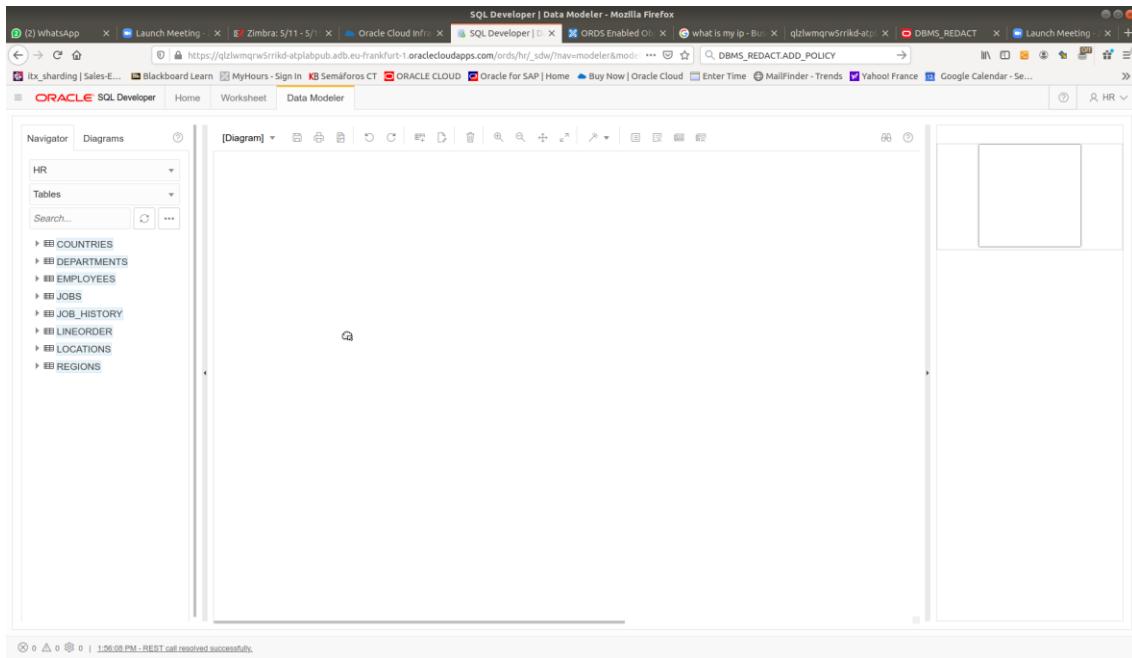


The screenshot shows a browser window with the URL <https://qlzlwmqrw5rrikd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/emp/100>. The page displays a JSON response for employee ID 100, including links for edit, describedby, collection, and the main collection.

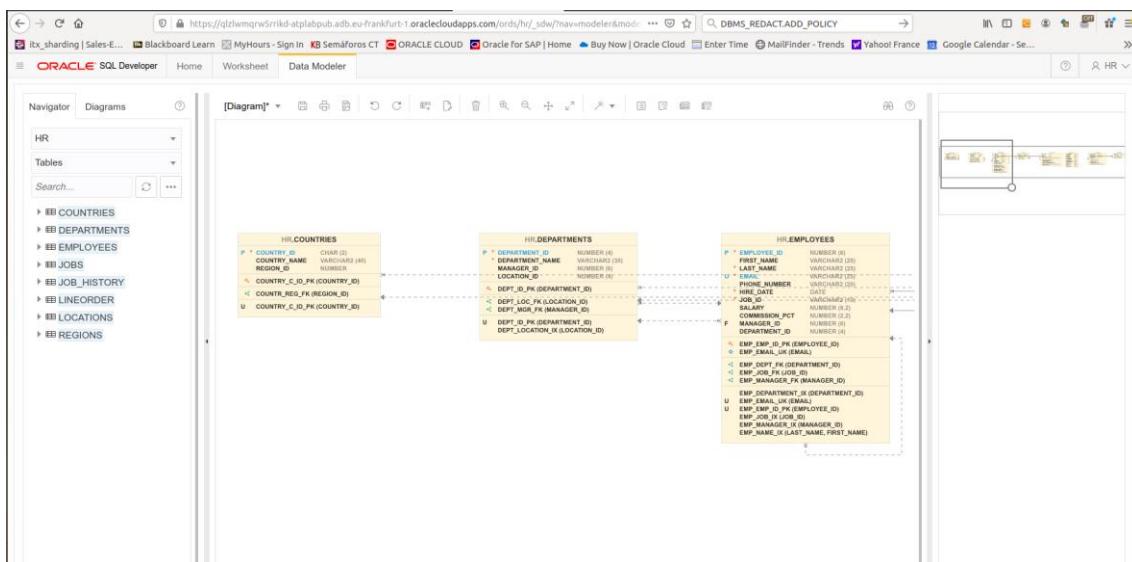
```
employee_id: 100
first_name: "Steven"
last_name: "King"
email: "SKING"
phone_number: "515.123.4567"
hire_date: "2003-06-17T00:00:00Z"
job_id: "AD_PRES"
salary: 24000
commission_pct: null
manager_id: null
department_id: 90
links:
  0:
    rel: "self"
    href: "https://qlzlwmqrw5rrikd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/emp/100"
  1:
    rel: "edit"
    href: "https://qlzlwmqrw5rrikd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/emp/100"
  2:
    rel: "describedby"
    href: "https://qlzlwmqrw5rrikd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/metadata-catalog/emp/item"
  3:
    rel: "collection"
    href: "https://qlzlwmqrw5rrikd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/emp/"
```

Finalmente, desde el Sql*Developer Web, hacemos click en la pestaña Data Modeler para visualizar el modelo de datos del esquema HR. Arrastramos todas las tablas a la parte central de la pantalla:





Y visualizamos nuestro modelo relacional:



Ejercicio 3: Crear un notebook en Oracle Machine Learning

En este ejercicio vamos a conectarnos a OML con el usuario ML_HR que hemos creado anteriormente. Desde la pantalla principal del ATP. Pulse en el botón “**Service Console**”.

Vamos a la parte de desarrollo (Development) dentro de la consola de servicio, y pulsamos en “**Oracle Machine Learning Notebooks**”:

The screenshot shows the Oracle Service Console interface. On the left, there's a sidebar with navigation links: Autonomous Transaction Processing (Overview, Activity, Administration, Development), DATABASE (ATPLABPUB), and ORACLE Cloud Infrastructure. The main content area has several sections: "Download Oracle Instant Client", "Oracle APEX", "SQL Developer Web", and "Oracle Machine Learning Notebooks". The "Oracle Machine Learning Notebooks" section is highlighted with a red box. It contains a brief description of what Oracle Machine Learning Notebooks are and how they can be used. Below the description is a "Copy URL" button.

En la pantalla de login, nos conectamos con el usuario **ML_HR/Autonomous#2020**:

The screenshot shows a login page titled "Sign in with your Oracle Machine Learning Database User credentials". It has fields for "USERNAME *" containing "ML_HR" and "PASSWORD *" containing a masked password. A blue "Sign In" button is at the bottom.

A continuación, aparece la pantalla principal de la sección de Machine Learning, elegimos la opción “**Notebooks**”:



Esto dará paso a la creación de nuestro primer Notebook de Machine Learning. Pulsamos en el botón de crear:

Damos un nombre al nuevo Notebook, en este caso **TESTNB**:

A continuación, ejecutamos una query en el nuevo notebook:

```
select /*MLnotebook*/ sum(lo_extendedprice*lo_discount) as revenue
from ssb.lineorder, ssb.dwdate
where lo_orderdate = d_datekey
and d_weeknuminyear = 6
and d_year = 1994
and lo_discount between 5 and 7
and lo_quantity between 26 and 35;
```

Podemos monitorizar la ejecución de la query en el “Performance Hub” desde la pestaña de ATP, dos pestanas a la izquierda de aquí:



Dentro de la pestaña SQL Monitoring, podemos ver la query ejecutada. Si entramos dentro de esta query se pueden ver los detalles:

Podemos ver en la pestaña SQL Text que, en este caso, el motor de Machine Learning no ha reescrito la query. Tambien podemos ver los detalles asociados a esta query, como el plan de ejecución, estadísticas, actividad, métricas, etc



Finalmente volvemos al Notebook y comprobamos el resultado de la query:

The screenshot shows the Oracle Machine Learning Notebook interface. A query has been run in the 'REVENUE' cell, resulting in a single row of data: '261137 2600347'. The notebook interface includes tabs for 'TESTNB', 'HR_ML Project [HR_ML Workspace...]', and 'Connected'. The status bar at the bottom indicates 'Took 32 sec. Last updated by HR_ML at April 07 2020, 12:02:04 PM (unstaged)'.

Vamos a ejecutar ahora una query contra la tabla HR.employees desde el mismo Notebook:

```
Select * from hr.employees;
```

The screenshot shows the results of the query 'Select * from hr.employees' in the notebook. The results are displayed as a table with 10 columns: EMPLOYEE_ID, FIRST_NAME, LAST_NAME, EMAIL, PHONE_NUMBER, HIRE_DATE, JOB_ID, SALARY, COMMISSION_PCT, and MANAGER_ID. The table contains 10 rows of data. The 'SALARY' column values are all masked with '9999'. The status bar at the bottom indicates 'Took 1 sec. Last updated by ML_HR at May 07 2020, 9:30:55 PM (unstaged)'.

Observamos que el campo “salary” esta **enmascarado** con “9”, ocultando el valor real del campo en todas las filas. Esto es el efecto de la **política de Data Redaction** que hemos implementado anteriormente.

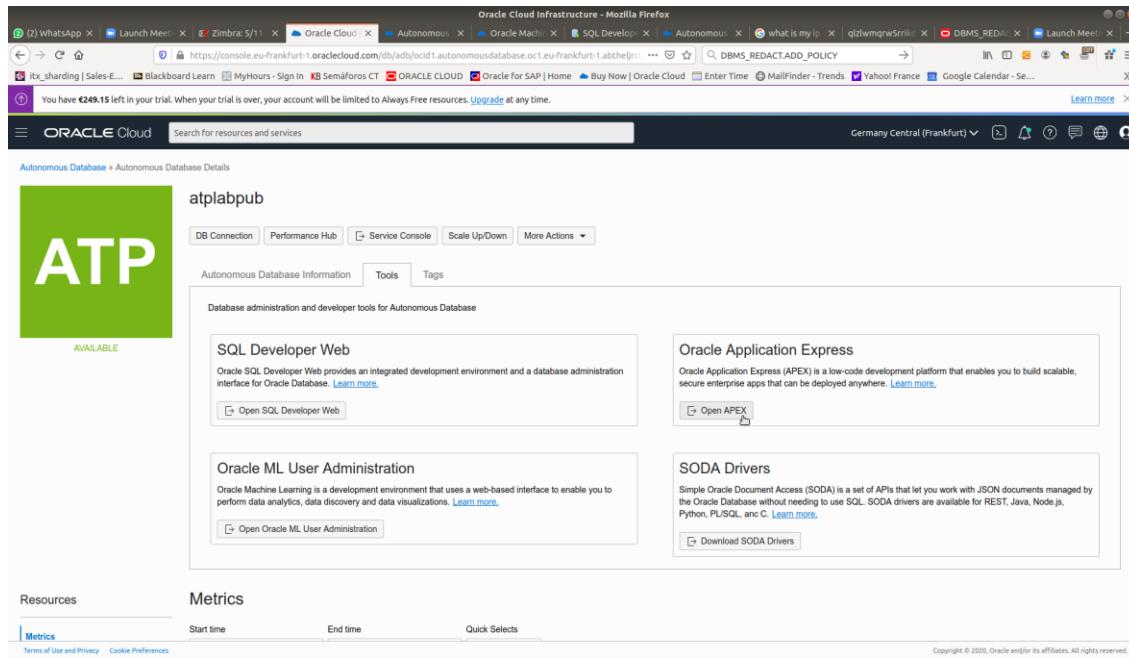
Podemos compararlo con la consulta que se hace con el usuario HR desde SQL Developer web.

The screenshot shows the results of the query 'Select * from hr.employees' in Oracle SQL Developer web. The results are displayed as a table with 10 columns: EMPLOYEE_ID, FIRST_NAME, LAST_NAME, EMAIL, PHONE_NUMBER, HIRE_DATE, JOB_ID, SALARY, COMMISSION_PCT, and MANAGER_ID. The table contains 10 rows of data. The 'SALARY' column values are not masked. The status bar at the bottom indicates 'Took 1 sec. Last updated by ML_HR at May 07 2020, 9:30:55 PM (unstaged)'.



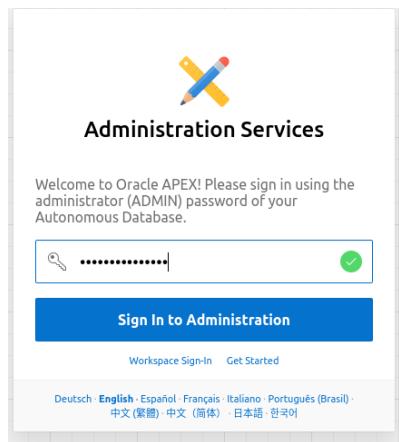
Ejercicio 4: Utilización de APEX

En el ejercicio siguiente, vamos a utilizar APEX. Desde la pantalla principal del ATP, en la pestaña “Tools”, elegimos “Oracle Application Express”:



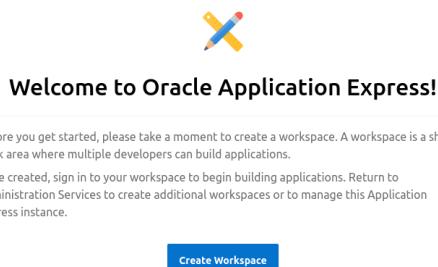
The screenshot shows the Oracle Cloud Infrastructure interface. In the top navigation bar, there is a link to "Oracle Application Express". Below the navigation, there is a section titled "Autonomous Database Information" which includes links to "SQL Developer Web", "Oracle Application Express", "Oracle ML User Administration", and "SODA Drivers". The "Oracle Application Express" section is specifically highlighted with a red box around its title and description.

Primero nos conectamos con el usuario ADMIN a la consola de administración de APEX:



The screenshot shows the "Administration Services" sign-in page for Oracle APEX. It has a logo at the top, a password input field with a mask, and a "Sign In to Administration" button. Below the button are links for "Workspace Sign-In" and "Get Started". At the bottom, there are language selection options: Deutsch, English, Español, Français, Italiano, Português (Brasil), 中文 (繁體), 中文 (简体), 日本語, and 한국어.

A continuación, creamos un “Workspace” con nombre WSHR para el usuario “HR”



The screenshot shows the "Welcome to Oracle Application Express!" page. It has a logo at the top, a message about creating a workspace, and a "Create Workspace" button at the bottom. Below the button, there is a copyright notice: "Copyright © 2020, Oracle and/or its affiliates. All rights reserved. | Oracle Confidential".

Create Workspace

Identify a new or existing database user to use with your new workspace.

* Database User: HR

* Password:

* Workspace Name: WSHR

Advanced

Una vez creado el Workspace, nos conectamos a APEX con el usuario HR, siguiendo el enlace arriba a la izquierda en la pantalla principal de APEX.

Application Express Administration Services - Mozilla Firefox

https://qlzlwmgw5rrkd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/f?p=4550:3:11213692322077

ORACLE APEX Manage Instance Manage Workspaces Monitor Activity

Workspace created. Sign out of Administration Services and sign in to [APEX](#) to begin building applications.

Instance Administration

Manage Instance Manage Workspaces Monitor Activity

System Message

Workspace Summary	
Workspaces	2
Schemas	2
Applications	2
Users	2
Mail Queue Entries	0
Websheets	0

Jobs

ORACLE_APEX_AUTO_APPROVAL	7 weeks ago
ORACLE_APEX_DAILY_MAINTENANCE	13 hours ago
ORACLE_APEX_MAIL_QUEUE	5 minutes ago
ORACLE_APEX_PURGE_SESSIONS	40 minutes ago
ORACLE_APEX_WS_NOTIFICATIONS	10 minutes ago

Oracle Application Express database jobs with time of last run

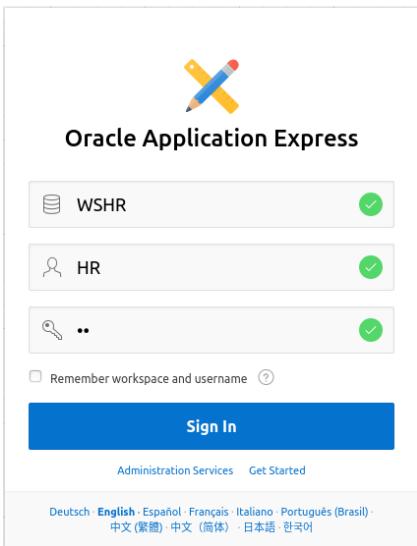
Deutsch English Español Français Italiano Português (Brasil) 中文 (简体) 中文 (简体) 日本語 한국어

https://qlzlwmgw5rrkd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/f?p=4550:1::F4550_P1_COMPANY,F4550_P1_USERNAME:WSHR,HR reserved.

Application Express 19.2.0.00.18

Y nos conectamos como HR/hr o hr/hr (**contraseña siempre en minúsculas**):





Seguimos los pasos siguientes:



Welcome to Oracle Application Express!

Before you get started, please take a moment to set your Application Express (APEX) account password.

Your access to this service is controlled by Single Sign-On (SSO). When your workspace was created, an APEX account was also created with your SSO username and a randomly generated password. Resetting this password is required to run apps you create.

Note: This will not reset your SSO password.

[Set APEX Account Password](#)

Completamos el perfil del usuario HR:

Edit Profile

Profile Details

Workspace	WSHR
Username	HR
Email Address	pp@gmail.com
First Name	
Last Name	

Profile Photo

Your profile photo personalizes your activity by showing up in the Top Users list. Add, change, or remove your photo.

Photo No file selected.

Esto nos lleva a la pantalla principal del workspace, desde donde podremos crear aplicaciones nuevas, gestionar el acceso por REST, etc ...

En esta pantalla, Pulsamos sobre el **menu “SQL Workshop”**, opción **“Restful Services”**:



Vemos que **ORDS** está habilitada sobre el esquema HR, que su alias es “hr”, y que tiene un objeto habilitado para REST. Pulsamos sobre “**Total Enabled Objects**”:



Ahora volvemos al menú SQL Workshop, y elegimos la opción “Object Browser”:



Esto nos lleva a una pantalla donde vemos los objetos del esquema HR. Pulsamos en el objeto “EMPLOYEES”, y accedemos a la **pestaña REST**.

Column Name	Data Type	Nullable
EMPLOYEE_ID	NUMBER(6,0)	No
FIRST_NAME	VARCHAR2(20)	Yes
LAST_NAME	VARCHAR2(25)	No
EMAIL	VARCHAR2(25)	No
PHONE_NUMBER	VARCHAR2(20)	Yes
HIRE_DATE	DATE	No
JOB_ID	VARCHAR2(10)	No
SALARY	NUMBER(8,2)	Yes
COMMISSION_PCT	NUMBER(2,2)	Yes
MANAGER_ID	NUMBER(6,0)	Yes
DEPARTMENT_ID	NUMBER(4,0)	Yes

Aquí vemos la **URL** a utilizar para acceder a la tabla mediante **API REST**:

REST Enable Object	<input checked="" type="checkbox"/> Yes	<input type="checkbox"/> No	(?)
Object Alias	emp (?)		
Authorization Required	<input type="checkbox"/> Yes	<input checked="" type="checkbox"/> No	(?)
RESTful URI	https://qlzlwqmqrw5rrikd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/emp/ (?)		

Si copiamos esta URL y la pegamos en un navegador, vemos los datos de la tabla, al igual que en un ejercicio anterior. Alternativamente, desde cualquiera de las máquinas “bastion”, podemos acceder a esta URL mediante cURL:

```
curl https://qlzlwqmqrw5rrikd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/emp/100
```

```
{"employee_id":100,"first_name":"Steven","last_name":"King","email":"SKING","phone_number":"515.123.4567","hire_date":"2003-06-17T00:00:00Z","job_id":"AD_PRES","salary":24000,"commission_pct":null,"manager_id":null,"department_id":90,"links":[{"rel":"self","href":"https://qlzlwqmqrw5rrikd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/emp/100"}, {"rel":"edit","href":"https://qlzlwqmqrw5rrikd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/emp/100"}, {"rel":"describedby","href":"https://qlzlwqmqrw5rrikd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/metadata-catalog/emp/item"}, {"rel":"collection","href":"https://qlzlwqmqrw5rrikd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/emp/"}]}
```



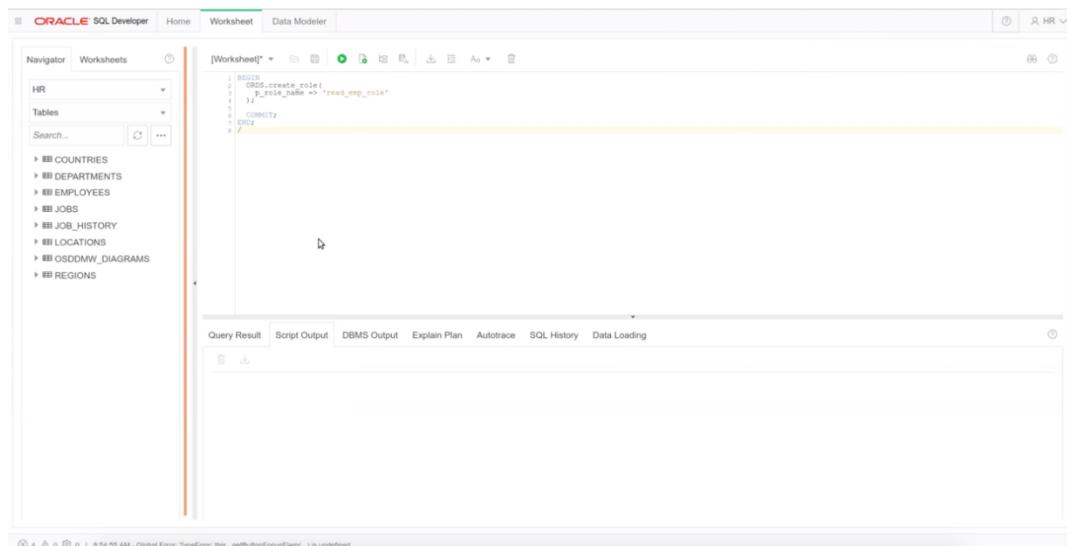
Ejercicio 5 (opcional): Configuración de seguridad de acceso OAuth2

En este ejercicio se explica como configurar seguridad de autenticación para acceder a los datos a través de REST API con un token de autenticación. Vamos a dotar el acceso a la tabla “employees” de seguridad mediante autenticación por token.

Nos conectamos al Sql*Developer Web como usuario HR, igual que en el ejercicio 2.

En primer lugar, hay que crear un rol, que se asociará al usuario HR y nos permitirá acceder al endpoint “/emp”:

```
BEGIN
  ORDS.create_role(
    p_role_name => 'read_emp_role'
  );
  COMMIT;
END;
/
```

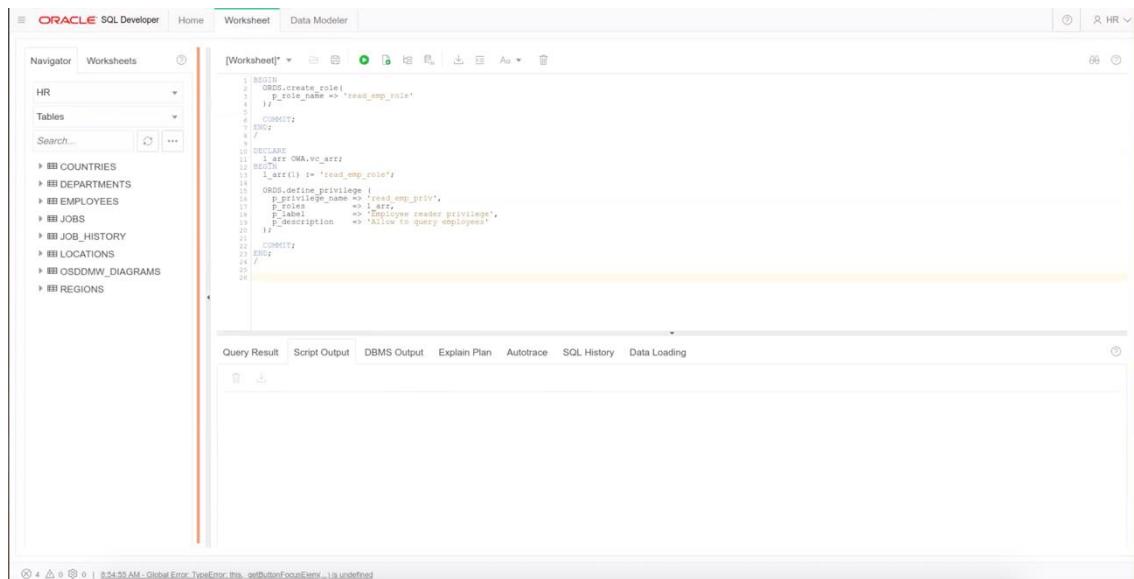


A continuación, se crea un privilegio en ORDS. Este privilegio lo asociamos al role creado en el paso anterior:

```
DECLARE
  l_arr OWA.VC_ARR;
BEGIN
  l_arr(1) := 'read_emp_role';

  ORDS.DEFINE_PRIVILEGE (
    p_privilege_name => 'read_emp_priv',
    p_roles          => l_arr,
    p_label          => 'Employee reader privilege',
    p_description    => 'Allow to query employees'
  );
  COMMIT;
END;
/
```





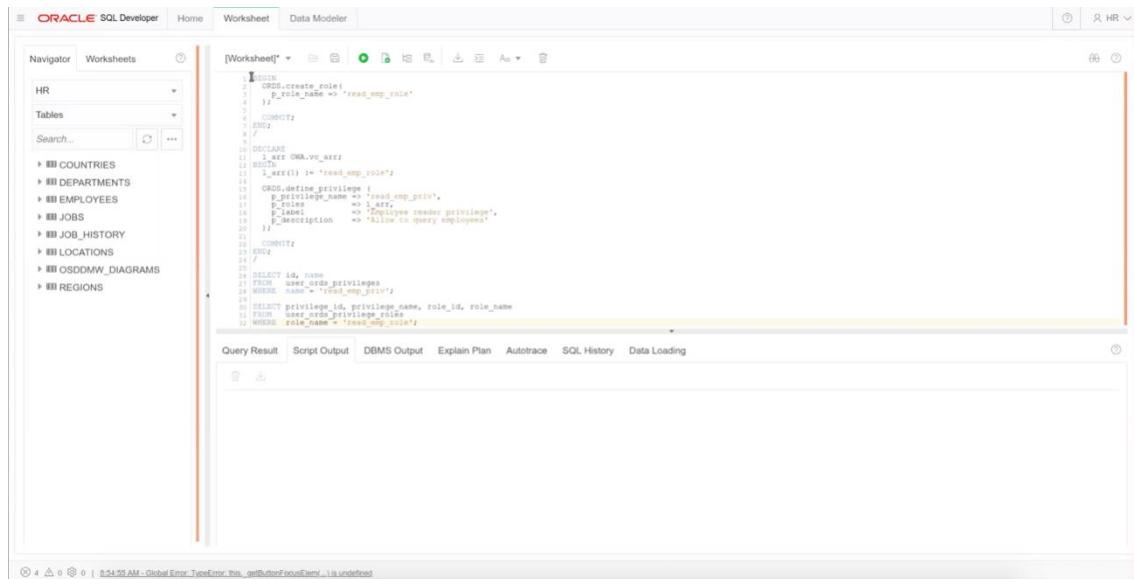
The screenshot shows the Oracle SQL Developer interface with the 'Worksheet' tab selected. In the central workspace, there is a PL/SQL script named '[Worksheet]*'. The script creates a role 'read_emp_role' and defines a privilege 'read_emp_priv' for it. The code is as follows:

```
1 BEGIN
2   ORDS.create_role(
3     p_role_name => 'read_emp_role'
4   );
5   COMMIT;
6 END;
7 /
8
9 DECLARE
10   l_attr OMA.VC_ATTR;
11   l_attr1 := 'read_emp_role';
12   l_attr1 := 'read_emp_priv';
13   ORDS.define_privilege (
14     p_privilege_name => 'read_emp_priv',
15     p_privilege_type => 'attr',
16     p_label      => 'Employees reader privilege',
17     p_description => 'Allow to query employees'
18   );
19   COMMIT;
20 END;
21 /
22
23 COMMIT;
24 END;
25 /
26
```

Con las siguientes queries, comprobamos que el rol ha sido correctamente asociado al privilegio de ORDS:

```
SELECT id, name
FROM user_ords_privileges
WHERE name = 'read_emp_priv';

SELECT privilege_id, privilege_name, role_id, role_name
FROM user_ords_privilege_roles
WHERE role_name = 'read_emp_role';
```



The screenshot shows the Oracle SQL Developer interface with the 'Worksheet' tab selected. The workspace contains the same PL/SQL script as before, followed by the two queries from the previous section. The 'Query Result' tab is selected, showing the output of the second query:

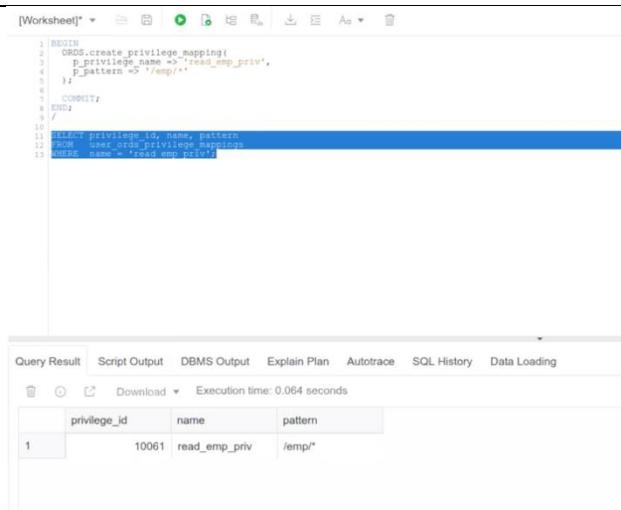
```
SELECT id, name
FROM user_ords_privileges
WHERE name = 'read_emp_priv';

SELECT privilege_id, privilege_name, role_id, role_name
FROM user_ords_privilege_roles
WHERE role_name = 'read_emp_role';
```



A continuación, mapeamos el privilegio a todas las terminaciones del endpoint “/emp/*”.

```
BEGIN
  ORDS.create_privilege_mapping(
    p_privilege_name => 'read_emp_priv',
    p_pattern => '/emp/*'
  );
  COMMIT;
END;
/
SELECT privilege_id, name, pattern
FROM user_ords_privilege_mappings
WHERE name = 'read_emp_priv';
```



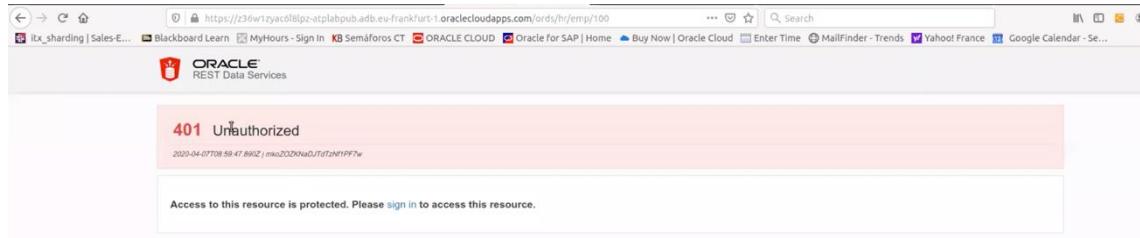
The screenshot shows a SQL worksheet with the following content:

```
1 BEGIN
2   ORDS.create_privilege_mapping(
3     p_privilege_name =>'read_emp_priv',
4     p_pattern => '/emp/*'
5   );
6   COMMIT;
7 END;
8 /
9
10
11 SELECT privilege_id, name, pattern
12 FROM user_ords_privilege_mappings
13 WHERE name='read_emp_priv';
```

Execution time: 0.064 seconds

privilege_id	name	pattern
1	10061	/emp/*

Comprobamos si podemos acceder a los datos de la tabla employees a través de un navegador, con la URL utilizada en ejercicios anteriores:



La URL falla, por fallo de autorización.



El siguiente paso es crear un token de autorización, valido durante una hora desde su ultima utilización.

Ejecutamos lo siguiente:

```
BEGIN
  OAUTH.create_client(
    p_name      => 'Employee Client',
    p_grant_type => 'client_credentials',
    p_owner      => 'Employees reader',
    p_description => 'Client app for employees consultation',
    p_support_email => 'tim@example.com',
    p_privilege_names => 'read_emp_priv'
  );
  COMMIT;
END;
/
```

```
SELECT id, name, client_id, client_secret
FROM   user_ords_clients;
```

La última sentencia SELECT devuelve un client ID y client secret. Los copiamos en un fichero de texto para introducirlos en la próxima llamada REST para conseguir un token de autenticación.

The screenshot shows a SQL worksheet with the following code:

```
1 BEGIN
2   OAUTH.create_client(
3     p_name      => 'Employee Client',
4     p_grant_type => 'client_credentials',
5     p_owner      => 'Employees reader',
6     p_description => 'Client app for employees consultation',
7     p_support_email => 'tim@example.com',
8     p_privilege_names => 'read_emp_priv'
9   );
10  COMMIT;
11 END;
12 /
13
14
15 SELECT id, name, client_id, client_secret
16 FROM   user_ords_clients;
```

Below the code, the "Query Result" tab is selected, showing the output:

	id	name	client_id	client_secret
1	10064	Employee Client	mVzhIKydBlVdL...	h0R4l0hJE6XV4...

A continuación, mapeamos el token de autenticación con el rol que se ha creado anteriormente para la tabla de empleados:

```
BEGIN
  OAUTH.grant_client_role(
    p_client_name => 'Employee Client',
    p_role_name    => 'read_emp_role'
  );
  COMMIT;
END;
/
```

```
SELECT client_name, role_name
FROM   user_ords_client_roles;
```



```

1 BEGIN
2   OAUTH.create_client(
3     p_name      => 'Employee Client',
4     p_grant_type => 'client_credentials',
5     p_owner      => 'Employees reader',
6     p_description => 'Client app for employees consultation',
7     p_support_email => 'tim@example.com',
8     p_privilege_names => 'read_emp_priv'
9   );
10  COMMIT;
11 END;
12 /
13
14 SELECT id, name, client_id, client_secret
15 FROM user_ordbs_clients;
16
17 SELECT name, client_name
18 FROM user_ordbs_client_privileges;
19
20 BEGIN
21   OAUTH.grant_client_role(
22     p_client_name =>'Employee Client',
23     p_role_name  =>'read_emp_role'
24   );
25 END;
26
27 COMMIT;
28 END;
29 /
30
31 SELECT client_name, role_name
32 FROM user_ordbs_client_roles;

```

Query Result

	client_name	role_name
1	Employee Client	read_emp_role

El siguiente paso es comprobar mediante el comando “curl”, desde una terminal Linux, si se puede acceder introduciendo el token de autenticación. Esto lo podemos hacer desde cualquiera de las máquinas de bastion.

Si no hemos apuntado el client_id y el secret en uno de los pasos anteriores, lo podemos consultar de nuevo con la siguiente query:

```
SELECT id, name, client_id, client_secret
FROM user_ordbs_clients;
```

A continuación, ejecutamos el siguiente comando, desde cualquiera de las máquinas bastion, para conseguir un token. Observamos en el comando de cURL el uso del parametro “**--user**”, con el valor **<client_id>:<secret>**:

La URL de oAUTH es nuestro REST Endpoint ya utilizado en varios ejercicios anteriores, completado por “**/hr/oauth/token**”:

```
#CLIENT_ID : x3n1g7heGXI0zxN_DJrIXw..
#CLIENT_SECRET : Az4WOTviFaDjgHgSMq-KLg..
#OAUTH URL : https://z36w1zyac618lpz-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/oauth/token
curl -i -k --user x3n1g7heGXI0zxN_DJrIXw..:Az4WOTviFaDjgHgSMq-KLg.. --data "grant_type=client_credentials"
https://<use su ATP ORDS URL>/ords/hr/oauth/token
```

El comando anterior nos devuelve un token, que utilizamos ahora para consultar la tabla “employees” (sustituir **<TOKEN>** por el token devuelto por el paso anterior):

```
curl -i -k -H"Authorization: Bearer <TOKEN>" https://z36w1zyac618lpz-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/emp/100
```

