

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 Oracle Autonomous Database Details page for the database 'atplabpub'. At the top, there is a green 'ATP' icon with 'AVAILABLE' status. Below it, the database name 'atplabpub' is displayed. The 'Tools' tab is selected, indicated by a red box around its label. Under the 'Tools' tab, there are four sections: 'SQL Developer Web', 'Oracle Application Express', 'Oracle ML User Administration', and 'SODA Drivers'. The 'Open Oracle ML User Administration' button is also highlighted with a red box.

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

Usuario: ADMIN

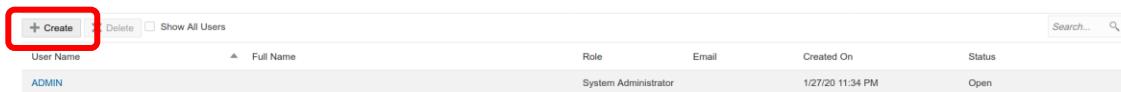
Contraseña: Autonomous#2021

The screenshot shows the Oracle Machine Learning Database Administrator login screen. It features a blue header with a cloud icon and the text 'SIGN IN'. Below the header, it says 'Database name: ATPLABPUB'. The main area is titled 'Sign in with your Oracle Machine Learning Database Administrator credentials'. It has fields for 'USERNAME *' (containing 'ADMIN') and 'PASSWORD *' (containing a series of dots). A large red box highlights the 'Sign In' button at the bottom.

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



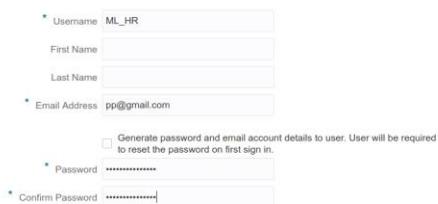
Users



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

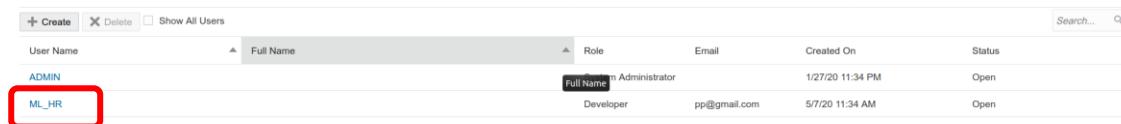
Password: Autonomous#2021

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

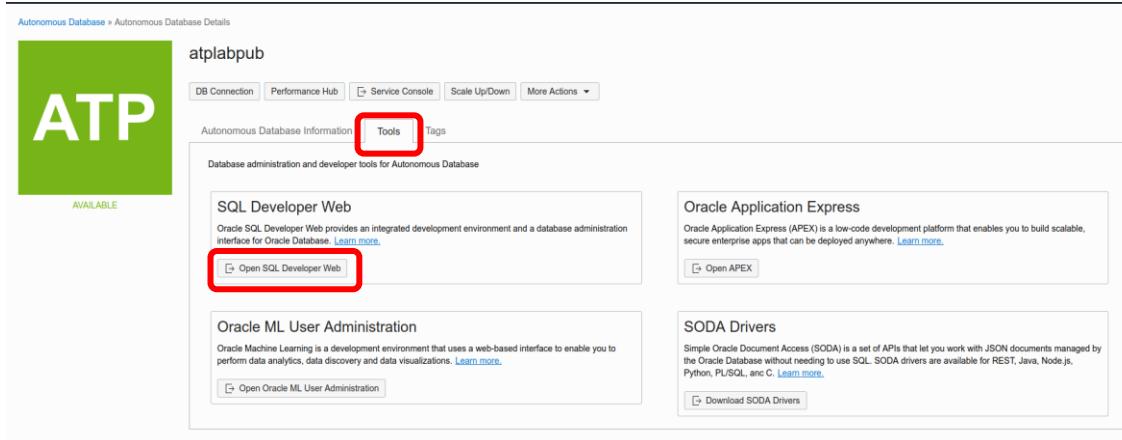


Ejercicio 2: Utilizar Sql*Developer Web

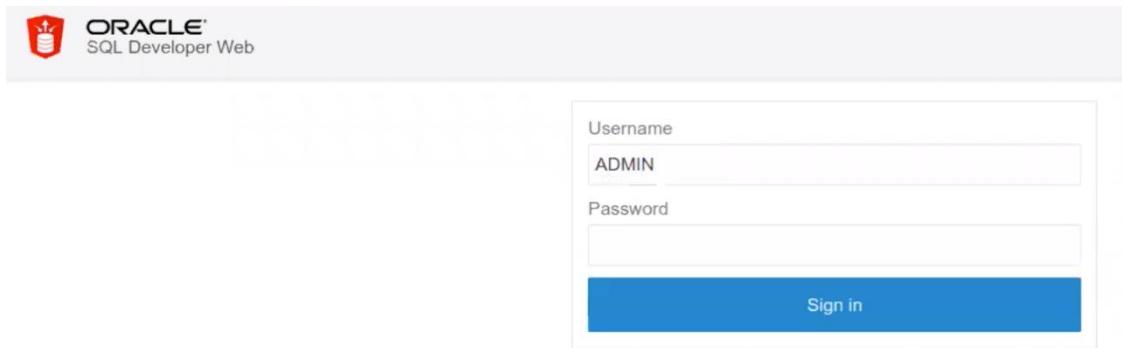
En este ejercicio, vamos a utilizar Sql*Developer Web para:

- 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”.

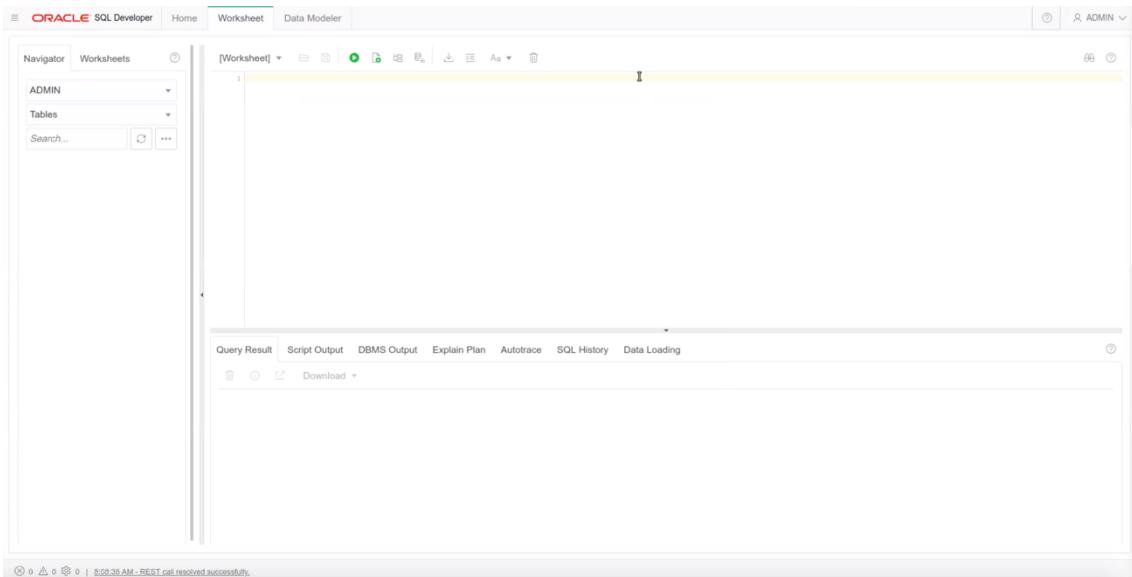


Esto nos lleva a la pantalla de login: conectarse como usuario ADMIN, password “Autonomous#2021”.



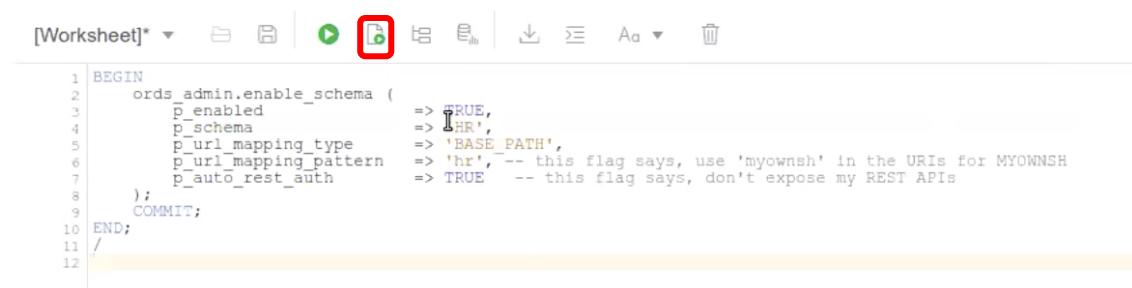
Una vez hecho esto, accedemos a la pantalla de **SQL Developer Web**, Que nos dará acceso para interactuar con la base de datos mediante SQL.





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

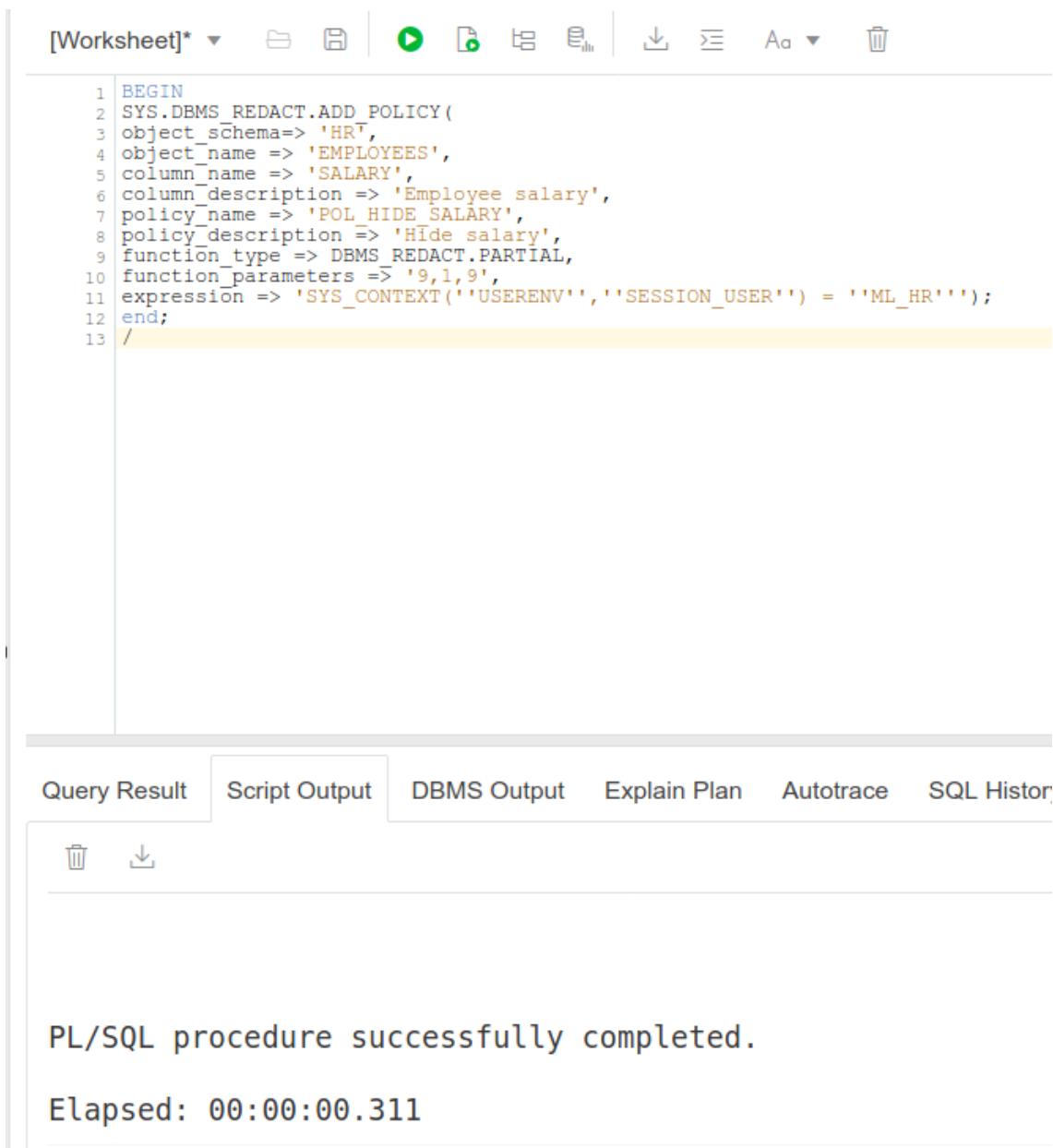
```
BEGIN
    ord.admin.enable_schema (
        p_enabled          => TRUE,
        p_schema            => 'HR',
        p_url_mapping_type => 'BASE_PATH',
        p_url_mapping_pattern => 'hr',
        p_auto_rest_auth   => TRUE      );
    COMMIT;
END;
/
```



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_all_tabs => 'TRUE',
7     p_grant_on_all_schemas => 'TRUE' );
8   COMMIT;
9 END;
/

```

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

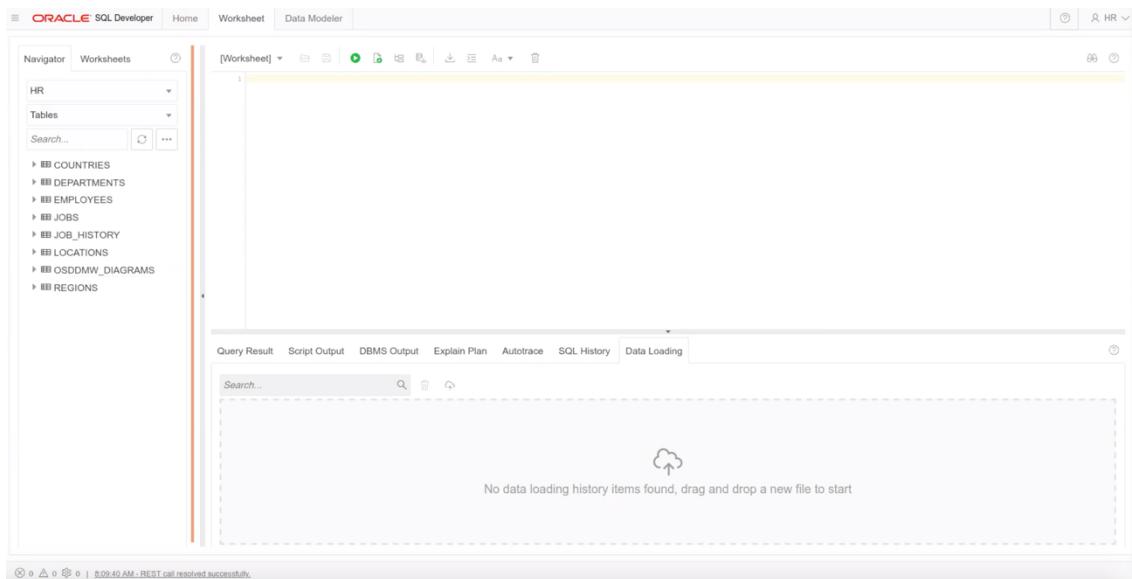
Luego desde la pagina principal del ATP, pestaña “**Tools**”, volvemos a cliquar “**Open Sql Developer Web**”, para volver a la pantalla de login. En la URL, **cambiamos “admin” por “hr”**:



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**).

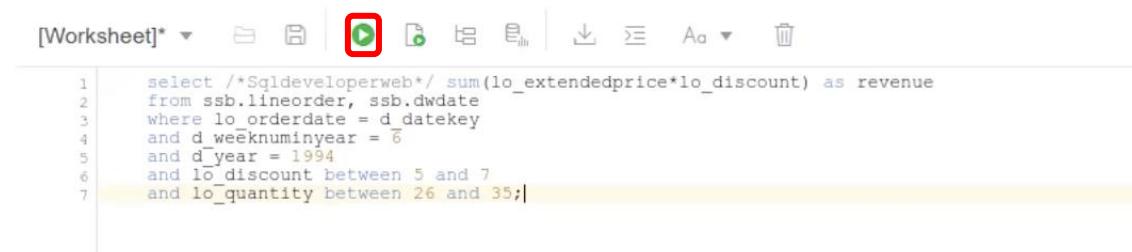
En la siguiente pantalla podemos ver la misma consola de SQL Developer Web, pero en este caso a la izquierda podemos ver las tablas del esquema HR:





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:

The screenshot shows the ASH Analytics interface with the 'SQL Monitoring' tab selected. A table lists a single session: Status: Executing, Duration: 20.00s, Inst ID: 1, SQL ID: 228q0dh6lrmz, User Name: HR@Z36W1ZYAC6L8LPZ_ATPLABPUB. The session is running a 'select' statement. Metrics shown include Database Time: 20.40s, I/O Requests: 71K.

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

The screenshot shows the Real-time SQL Monitoring page for SQL ID 228q0dh6lrmz. It includes an 'Overview' section with metrics like Duration (28.0s), Database Time (28.6s), and Activity (100%). Below it is a 'Time & Wait' chart and an 'I/O' chart showing buffer, IO, requests, and bytes. The 'SQL Text' section displays the query: 'select * from l\$sel\$_q_1\$\$_ row_number() over (order by 1) RN____ from l\$sel\$_q_1\$\$_ left join l\$sel\$_q_1\$\$_ to extendedprice*ls.discount) as revenue from l\$sel\$_lineorder, l\$sel\$_datekey where ls.orderdate = d.datekey and ls.orderdate between 1994 and 1995 and ls.discount between 5 and 7 and ls.quantity between 26 and 35 ; q; where RN____ between :1 and :2'.

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

The screenshot shows the 'SQL Text' tab of the SQL monitoring details. It displays the full query code: 'select * from l\$sel\$_q_1\$\$_ left join l\$sel\$_q_1\$\$_ to extendedprice*ls.discount) as revenue from l\$sel\$_lineorder, l\$sel\$_datekey where ls.orderdate = d.datekey and ls.orderdate between 1994 and 1995 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, the 'Worksheet' tab is selected. On the left, the Navigator pane shows the 'HR' schema with tables: 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.locations l
  4      ,developer.products p
  5      ,developer.suppliers s
  6      ,developer.orders o
  7      ,developer.order_items i
  8      ,developer.line_items l
  9     where l.orderdate = d.datekey
    10    and d.weekendquarter = 8
    11    and p.product_id = i.product_id
    12    and l.discount between 3 and 7
    13    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 'Worksheet' tab is selected. The SQL query is identical to the one 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 contains a PL/SQL script and its results. The script is as follows:

```
1  ETV 'ID', Employee_id;
2  KEY 'full_name' IS E.first_name || ' ' || E.last_name,
3  KEY 'email' IS E.email;
4  TYPE 'emp' IS EMPLOYEE;
5  TYPE 'emp' IS EMPLOYEE;
6  AS EMPLOYEE_Payload;
7  FROM employees;
8
```

The results pane shows the output of the query, which is a JSON array of employee records:

```
[{"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"}]
```

Below the results, the message 'Elapsed: 00:00:00.029' and '107 rows selected.' is displayed.

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 code 'grant select on employees to ML_HR;' is entered in the worksheet area. The code is highlighted in blue.

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

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 (OML) Notebooks are a collaborative, Apache Zeppelin-based user interface for data scientists and broader SQL users of Autonomous Database. OML Notebooks provide access to in-database parallel, distributed machine learning algorithms, in addition to statistical and analytical SQL and PL/SQL functions. OML 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.

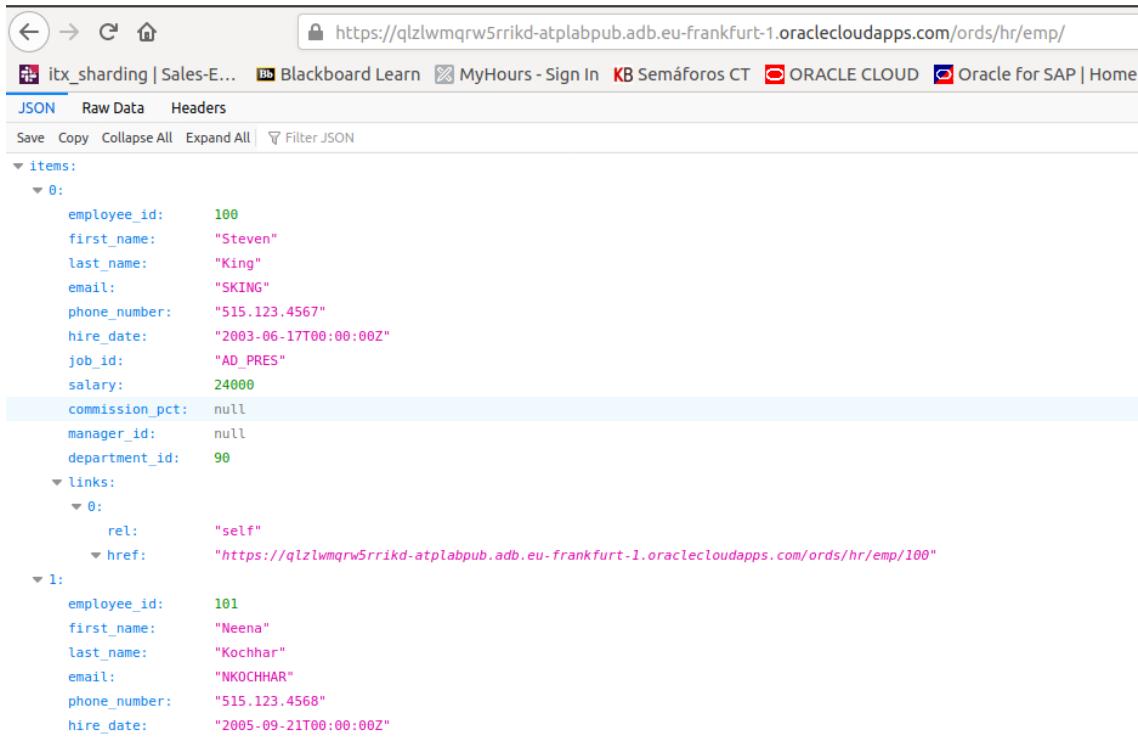
<https://QLZLWMQRW5RRIKD-ATPLABPUB.adb.eu-frankfurt-1.oraclecloudapps.com/ords/>

[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/>



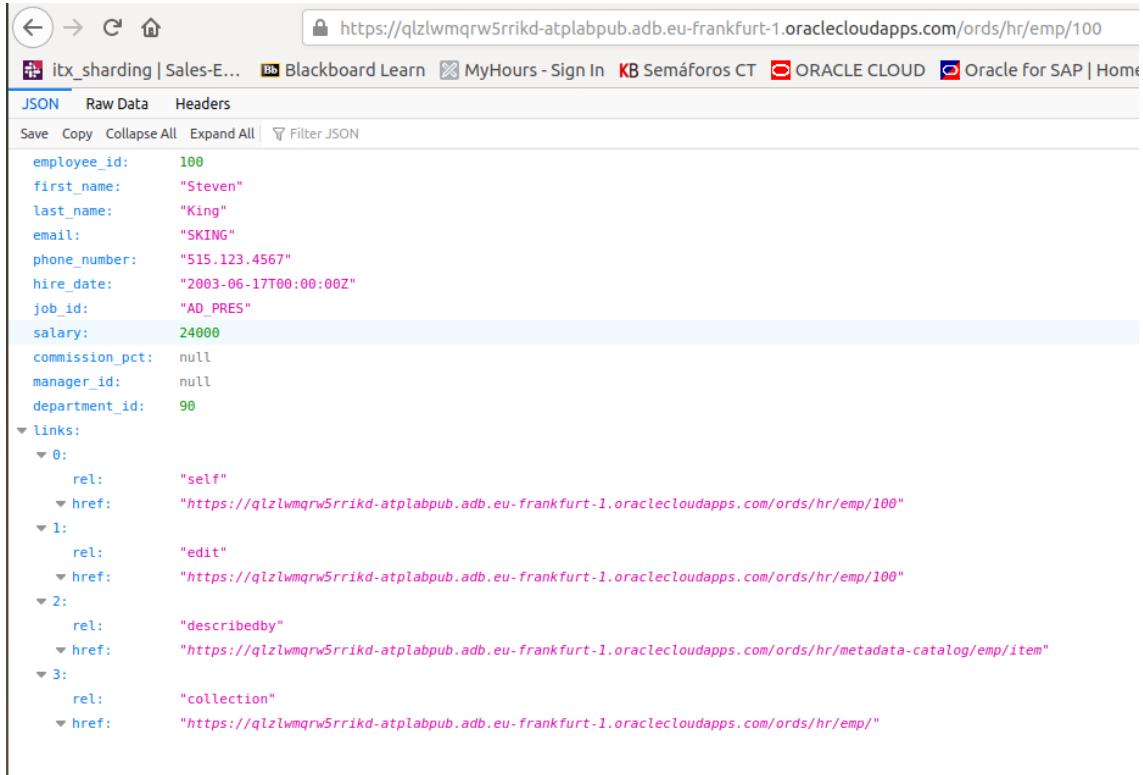
Si pegamos esta URL en un navegador Web, vemos los datos de la tabla “employees”:



The screenshot shows a browser window with the URL <https://qlzlwmqrw5rrikd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/emp/>. The page title is "itx_sharding | Sales-E...". The JSON response is displayed, showing two employees (0 and 1) with their details like employee_id, first_name, last_name, etc. Employee 0 has an employee_id of 100 and Employee 1 has an employee_id of 101. The JSON structure includes items, links, and specific employee 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_PRE"
    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”:

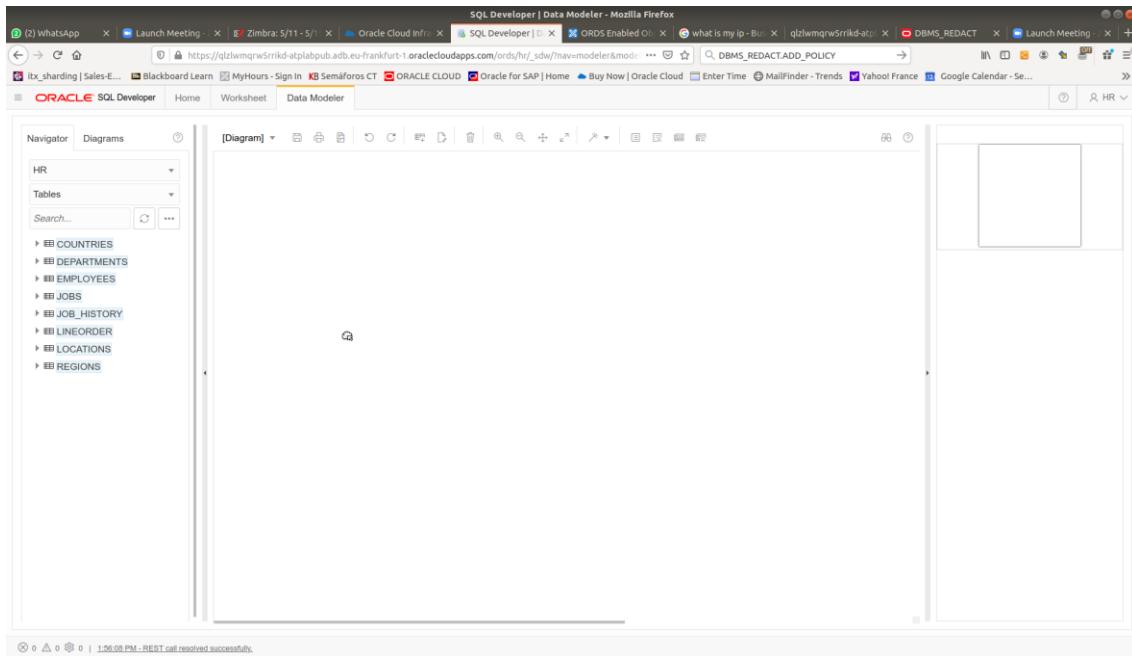


The screenshot shows a browser window with the URL <https://qlzlwmqrw5rrikd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/emp/100>. The page title is "itx_sharding | Sales-E...". The JSON response is displayed, showing Employee 0 with an employee_id of 100. The JSON structure includes items, links, and specific employee details.

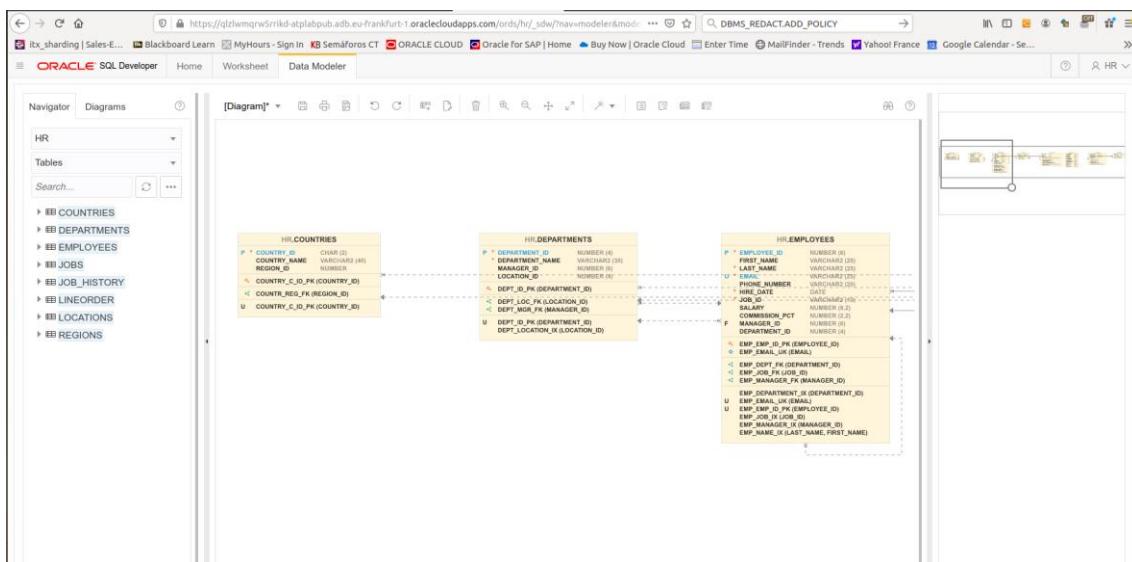
```
employee_id: 100
first_name: "Steven"
last_name: "King"
email: "SKING"
phone_number: "515.123.4567"
hire_date: "2003-06-17T00:00:00Z"
job_id: "AD_PRE"
salary: 24000
commission_pct: null
manager_id: null
department_id: 90
links:
  0:
    rel: "self"
    href: "https://qlzlwmqrw5rrikd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/emp/100"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

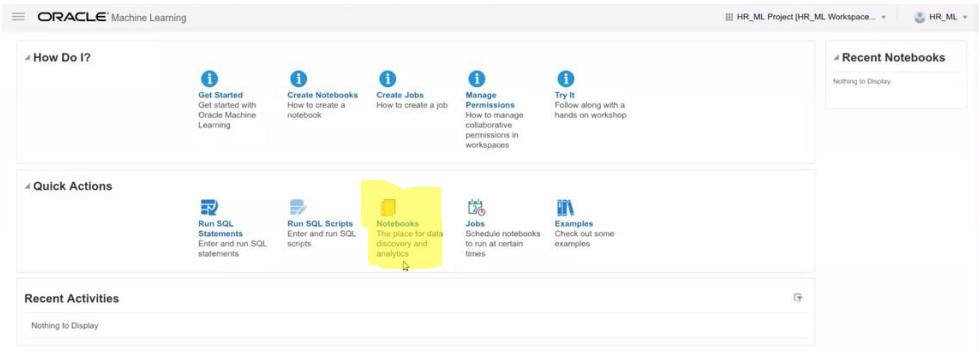
The screenshot shows the Oracle Service Console interface. On the left, there's a sidebar with 'Autonomous Transaction Processing' selected. Underneath it, 'Development' is highlighted. The main area has several cards: 'Download Oracle Instant Client', 'Oracle APEX', 'SQL Developer Web', and 'Oracle Machine Learning Notebooks'. The 'Oracle Machine Learning Notebooks' card is expanded, showing its description and a 'Copy URL' button.

En la pantalla de login, nos conectamos con el usuario **ML_HR/Autonomous#2021**:

The screenshot shows a login form. It asks for a 'Database name:' which is filled with 'ATPLABPUB'. Below it, it says 'Sign in with your Oracle Machine Learning Database User credentials'. There are fields for 'USERNAME *' containing 'ML_HR' and 'PASSWORD *' with a masked password. At the bottom is a blue 'Sign In' button.

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 also displays some configuration details at the top.

```
select /*+rownum*/ sum((extendedprice*discount)) as revenue
from stg_lineorder, stg_order
where stg_order.o_orderkey = stg_lineorder.l_orderkey
and stg_order.o_shipinstruct = 'C'
and stg_order.o_orderstatus = 'F'
and l_discount between 1 and 7
and l_extendedprice > 50;
```

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 in the notebook. The salary column values are all masked with '9999'. The table includes columns: EMPLOYEE_ID, FIRST_NAME, LAST_NAME, EMAIL, PHONE_NUMBER, HIRE_DATE, JOB_ID, SALARY, COMMISSION_PCT, MANAGER_ID, and DEPARTMENT_ID.

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
100	Steven	King	SKING	515.123.4567	2003-06-17 00:00:00	AD_PRES	99999		90	
101	Neena	Kochhar	NKOCHHAR	515.123.4568	2005-09-21 00:00:00	AD_VP	99999		100	90
102	Lex	De Haan	LDEHAAN	515.123.4569	2001-01-13 00:00:00	AD_VP	99999		100	90
103	Alexander	Hunold	AHUNOLD	590.423.4567	2006-01-03 00:00:00	IT_PROG	9999		102	60
104	Bruce	Ernst	BERNST	590.423.4568	2007-05-21 00:00:00	IT_PROG	9999		103	60
105	David	Austin	DAUSTIN	590.423.4569	2005-06-25 00:00:00	IT_PROG	9999		103	60
106	Valli	Pataballa	VPATABAL	590.423.4560	2006-02-05 00:00:00	IT_PROG	9999		103	60
107	Diana	Lorentz	DLORENTZ	590.423.5567	2007-02-07 00:00:00	IT_PROG	9999		103	60

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.

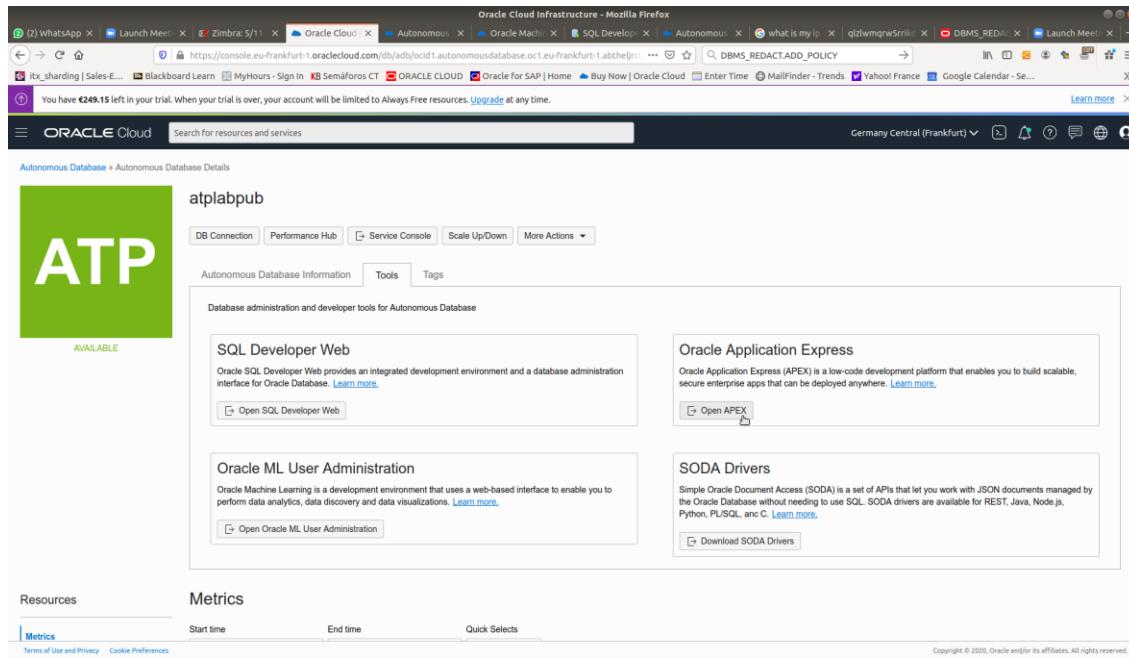
The screenshot shows the results of the query in SQL Developer web. The salary column values are not masked. The table structure is identical to the one in the notebook.

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
100	Steven	King	SKING	515.123.4567	2003-06-17T00:00:00Z	AD_PRES	24000		90	
101	Neena	Kochhar	NKOCHHAR	515.123.4568	2005-09-21T00:00:00Z	AD_VP	17000		100	90
102	Lex	De Haan	LDEHAAN	515.123.4569	2001-01-13T00:00:00Z	AD_VP	17000		100	90
103	Alexander	Hunold	AHUNOLD	590.423.4567	2006-01-03T00:00:00Z	IT_PROG	9000		102	60
104	Bruce	Ernst	BERNST	590.423.4568	2007-05-21T00:00:00Z	IT_PROG	6000		103	60
105	David	Austin	DAUSTIN	590.423.4569	2005-06-25T00:00:00Z	IT_PROG	4800		103	60
106	Valli	Pataballa	VPATABAL	590.423.4560	2006-02-05T00:00:00Z	IT_PROG	4800		103	60
107	Diana	Lorentz	DLORENTZ	590.423.5567	2007-02-07T00:00:00Z	IT_PROG	4200		103	60



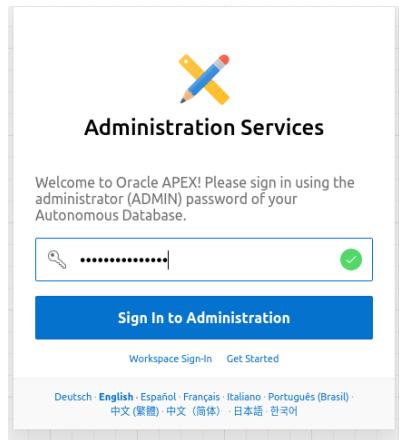
Ejercicio 4: Utilización de APEX

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



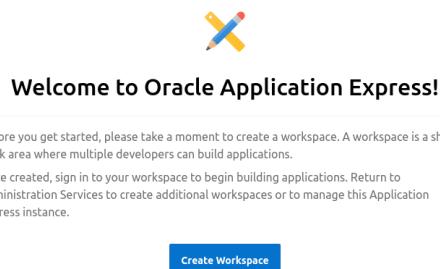
The screenshot shows the Oracle Cloud Infrastructure interface. In the top navigation bar, there is a link to "Oracle Application Express". Below the navigation, there is a section titled "Autonomous Database Information" which includes links to "SQL Developer Web", "Oracle Application Express", "Oracle 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" page of Oracle APEX. It has a logo at the top, followed by a message: "Welcome to Oracle APEX! Please sign in using the administrator (ADMIN) password of your Autonomous Database.". Below this is a password input field with a magnifying glass icon and a green checkmark icon. A large blue button labeled "Sign In to Administration" is centered. At the bottom, there are links for "Workspace Sign-In" and "Get Started", and language options: Deutsch, English, Español, Français, Italiano, Português (Brasil), 中文 (繁體), 中文 (简体), 日本語, 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, followed by a message: "Before you get started, please take a moment to create a workspace. A workspace is a shared work area where multiple developers can build applications." Below this is another message: "Once created, sign in to your workspace to begin building applications. Return to Administration Services to create additional workspaces or to manage this Application Express instance." At the bottom, there is a blue button labeled "Create Workspace".



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://qlzlwmgqw5rrkd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/f?p=4550:1::F4550_P1_COMPANY,F4550_P1_USERNAME:WSHR,HR

ORACLE APEX Manage Instance Manage Workspaces Monitor Activity

Workspace created. Sign out of Administration Services and sign in to [WSHR](#) 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

About

Access and perform administration tasks for an entire Oracle Application Express instance.

Learn More ...

Instance Tasks

Feature Configuration

Workspace Tasks

Create Workspace

Find a Workspace

Manage Workspaces

Available Updates

System is up-to-date. Once a new Application Express release becomes available, your instance will be immediately upgraded.

Oracle Application Express database jobs with time of last run

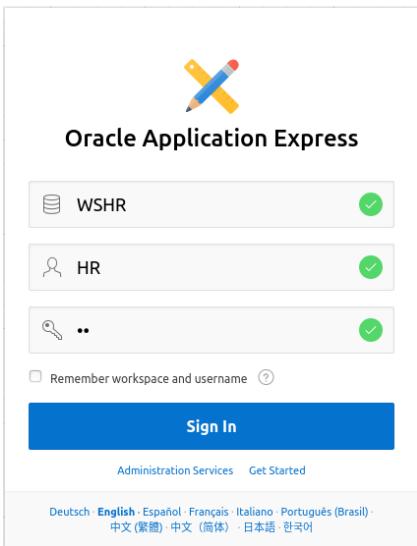
Deutsch English Español Français Italiano Português (Brasil) 中文 (简体) 中文 (简体) 日本語 한국어

https://qlzlwmgqw5rrkd-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”**:



The screenshot shows the Oracle Application Express (APEX) interface. The top navigation bar includes links like 'Launch Me!', 'Zimbra', 'Oracle Cloud', 'Oracle App...', 'Autonomous...', 'Oracle Ma...', 'SQL Develop...', 'Autonomous...', 'what is my...', 'DBMS_REDACT...', 'Launch Me...', and 'ORACLE CLOUD'. The main menu has sections for 'ORACLE APEX', 'App Builder', 'SQL Workshop', 'Team Development', 'App Gallery', and 'About'. The 'SQL Workshop' section is active, showing icons for 'Object Browser', 'SQL Commands', 'SQL Scripts', 'Utilities', and 'RESTful Services'. A tooltip points to the 'RESTful Services' icon. To the right, there's a dashboard with metrics for 'Applications' (0), 'Tables' (8), 'Productivity Apps' (0), 'Features' (0), and 'Resources' (Community Site, Blog, Hands-On Labs, Education, Social). Below the dashboard, there are links for 'Community Site', 'Blog', 'Hands-On Labs', 'Education', and 'Social' with icons for Twitter, LinkedIn, Facebook, and YouTube.

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**”:

The screenshot shows the ORDS RESTful Services interface. The top navigation bar includes links like 'Launch Me!', 'Zimbra', 'Oracle Cloud', 'ORDS REST', 'Autonomous...', 'Oracle Ma...', 'SQL Develop...', 'Autonomous...', 'what is my...', 'DBMS_REDACT...', 'Launch Me...', and 'ORACLE CLOUD'. The main menu has sections for 'ORACLE APEX', 'App Builder', 'SQL Workshop', 'Team Development', 'App Gallery', and 'About'. The 'ORDS REST' section is active, showing the 'ORDS Version' as 19.4.3.r1061746. The 'RESTful Data Services' section is expanded, showing 'Enabled Objects' (selected), 'Modules', 'Privileges', and 'Roles'. The 'Schema Access' section shows 'Access Status' as 'ENABLED' with a green checkmark. The 'Metadata Access' section shows 'Authorization Required' as 'ENABLED' with a green checkmark. The 'Enabled Objects' section shows a summary: 'Total Modules' (0), 'Total Privileges' (4), 'Total Roles' (8), and 'Total Enabled Objects' (1). Below this, there are sections for 'Module Status' (No Modules Defined), 'Module Security' (No Modules Defined), and 'Object Aliases' (No Aliases Defined).



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://qlzlwqrw5rrikd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/emp/ (copy) (?)		

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://qlzlwqrw5rrikd-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://qlzlwqrw5rrikd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/emp/100"}, {"rel":"edit","href":"https://qlzlwqrw5rrikd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/emp/100"}, {"rel":"describedby","href":"https://qlzlwqrw5rrikd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/metadata-catalog/emp/item"}, {"rel":"collection","href":"https://qlzlwqrw5rrikd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/emp/"}]}
```



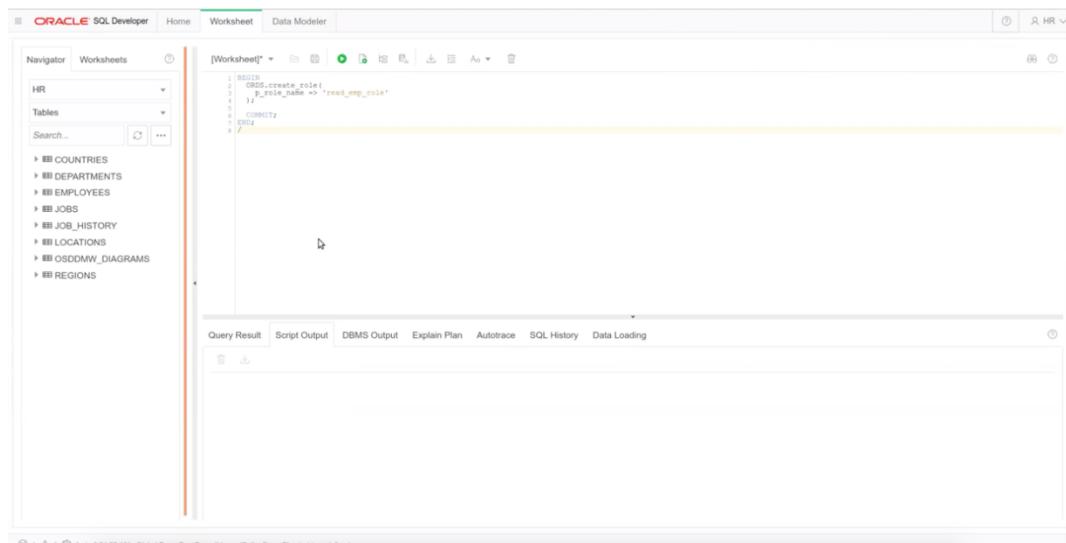
Ejercicio 5 (opcional): Configuración de seguridad de acceso OAuth2

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. The top menu bar includes 'ORACLE SQL Developer', 'Home', 'Worksheet', 'Data Modeler', and various icons. On the left, the Navigator pane shows 'HR' selected, with tables like COUNTRIES, DEPARTMENTS, EMPLOYEES, JOBS, JOB_HISTORY, LOCATIONS, OSDMW_DIAGRAMS, and REGIONS listed. The main workspace is titled '[Worksheet]*' and contains the following PL/SQL code:

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

Below the code, the 'Query Result' tab is active, followed by 'Script Output', 'DBMS Output', 'Explain Plan', 'Autotrace', 'SQL History', and 'Data Loading'. The status bar at the bottom shows '4 ▲ 0 ⚙ 0 | 8:24:55 AM - Global Error: TypeError: this._perfbuttonFocusEvent_ is undefined'.

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. The top navigation bar includes tabs for Navigator, Worksheets, Home, and Worksheet (which is currently selected). Below the navigation is a sidebar with sections for HR and Tables, and a search bar. The main workspace displays a PL/SQL script for creating a role and defining privileges. The code is as follows:

```
1 BEGIN
2   GRDS.create_role(
3     p_role_name => 'read_emp_role'
4   );
5 END;
6 /
7 COMMIT;
8 /
9 /
10 DECLARE
11   l_arr OWA.VC_ARR;
12   BEGIN
13     l_arr(1) := 'read_emp_role';
14   GRDS.define_privilege (
15     p_privilege_name => 'read_emp_priv',
16     p_privilege_desc => 'read_emp_priv',
17     p_label      => 'Employee reader privilege',
18     p_description => 'Allows to query employees'
19   );
20   END;
21 /
22 COMMIT;
23 /
24 /
25 SELECT l4.name
26   FROM user_ords_privileges
27  WHERE name = 'read_emp_priv';
28 /
29 SELECT privilege_id, privilege_name, role_id, role_name
30   FROM user_ords_privileges
31  WHERE role_name = 'read_emp_role';
32 /
33 
```

Below the code, there are tabs for Query Result, Script Output, DBMS Output, Explain Plan, Autotrace, SQL History, and Data Loading. The status bar at the bottom indicates the time as 8:24:50 AM and shows a Global Error message.



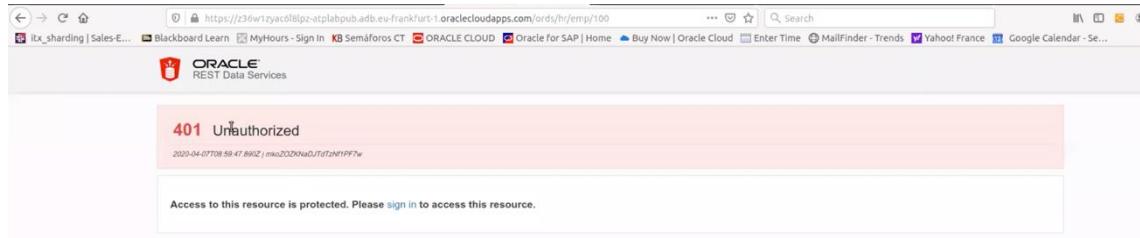
A continuación, mapeamos el privilegio a todas las terminaciones del endpoint “/emp/*”.

```
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 developer interface. At the top, there is a code editor window containing the provided PL/SQL script. Below it is a query result window titled 'Query Result' which displays the output of the last SELECT statement. The output is a single row in a table:

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
```

