

HOL5 - ATP tooling



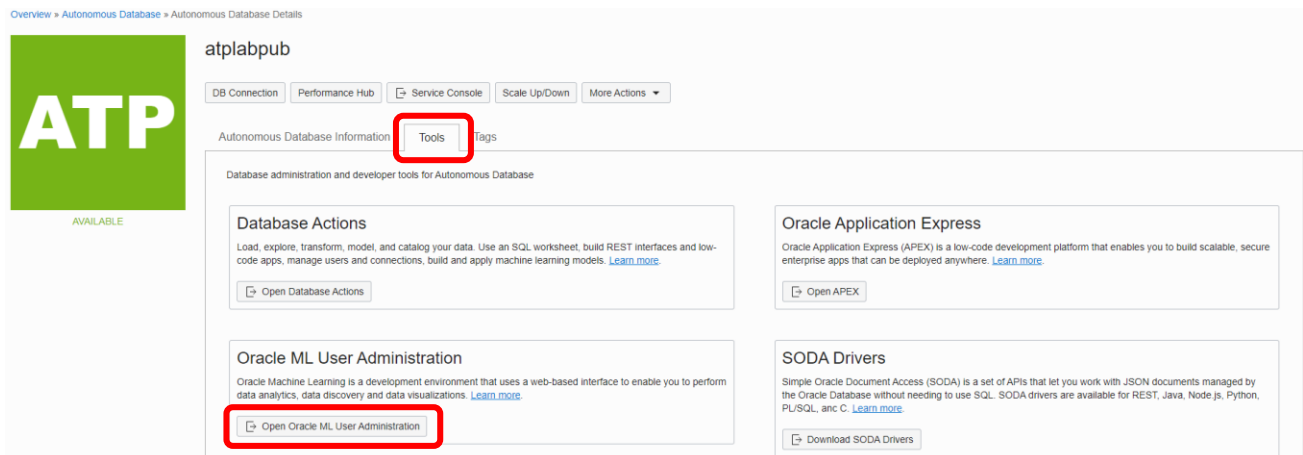
INDICE

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	17
EJERCICIO 4: UTILIZACIÓN DE APEX	21
EJERCICIO 5 (OPCIONAL): CONFIGURACIÓN DE SEGURIDAD DE ACCESO OAUTH2	29

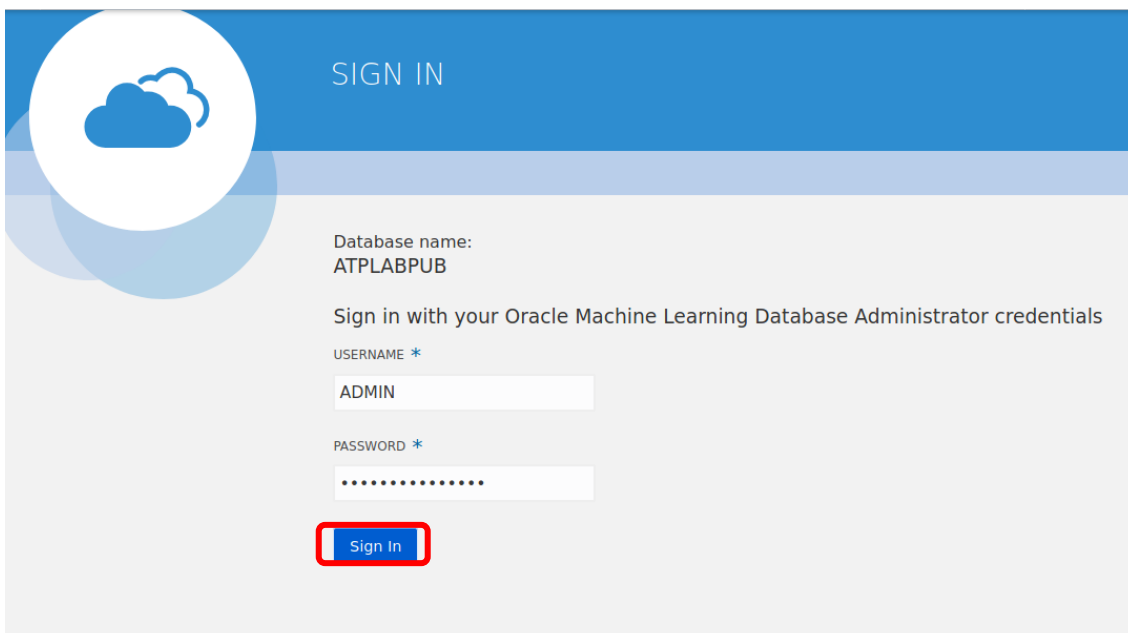


Ejercicio 1: Crear usuario de Machine Learning

En este ejercicio se explica como 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**”



En la pantalla de login, entrar las **credenciales** de **ADMIN**:
Usuario: ADMIN
Contraseña: Autonomous#2020



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



Users

+ Create

Delete

☐ Show All Users

Search...

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

Clicar el botón “**Create**”.

Create User

Create Cancel

* 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, rellenar 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

<div><div><div>+ Create</div><div>X Delete</div><div><input type="checkbox"/> Show All Users</div></div><div>Search...</div></div>					
User Name	Full Name	Role	Email	Created On	Status
ADMIN		Full Name Administrator		1/27/20 11:34 PM	Open
ML_HR		Developer	pp@gmail.com	5/7/20 11:34 AM	Open

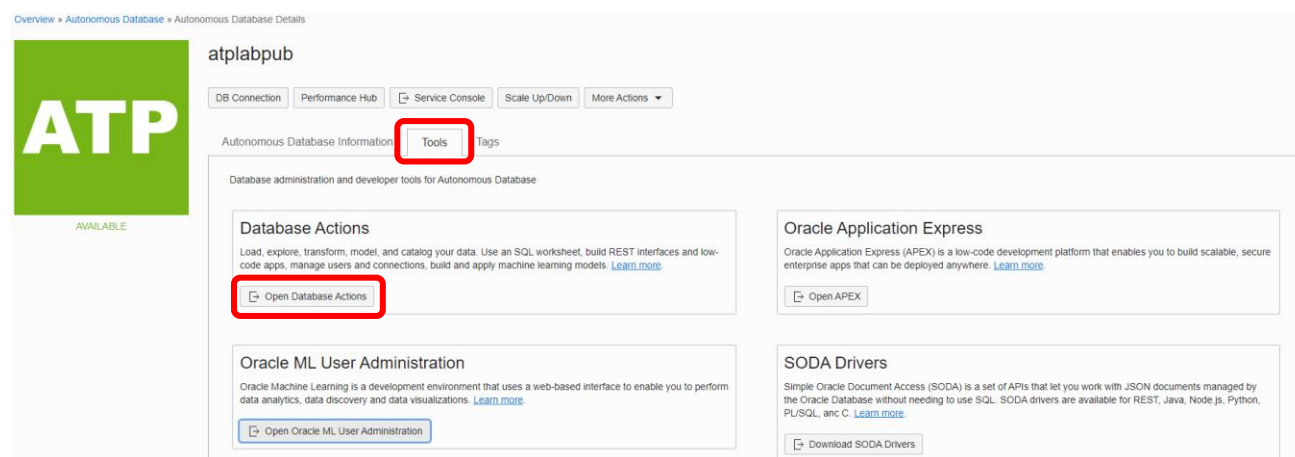


Ejercicio 2: Utilizar Sql*Developer Web

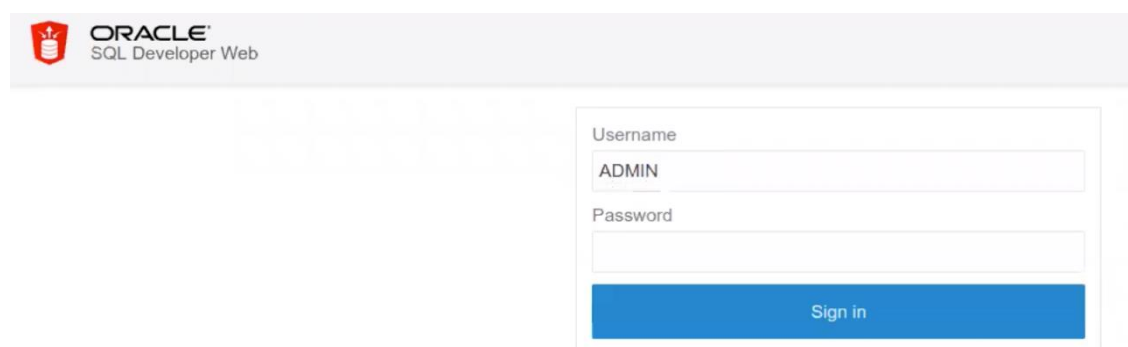
En este ejercicio, vamos a utilizar Sql*Developer Web para:

- 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**”, clicar el botón “**Open Database Actions**”.

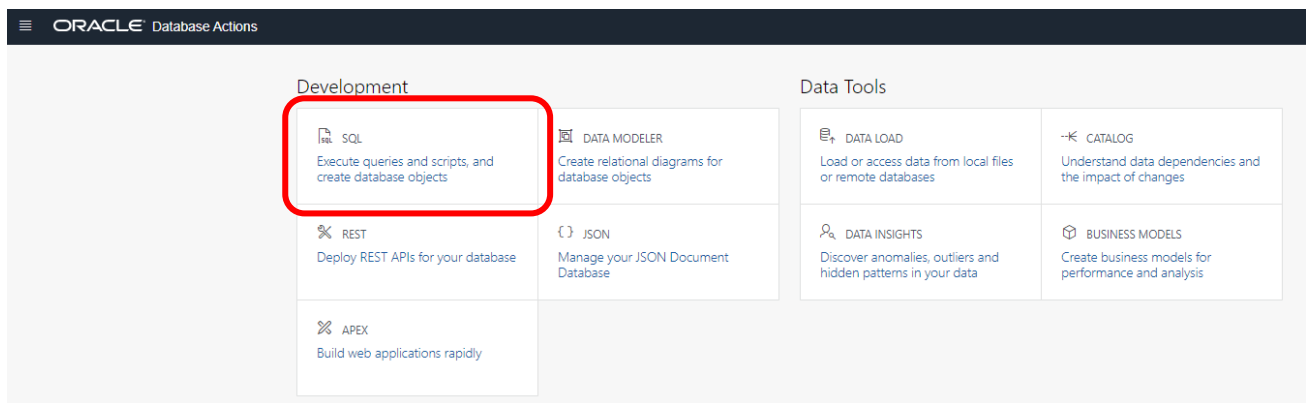


Esto nos lleva a la pantalla de login: conectarse como usuario ADMIN, password “Autonomous#2020”.

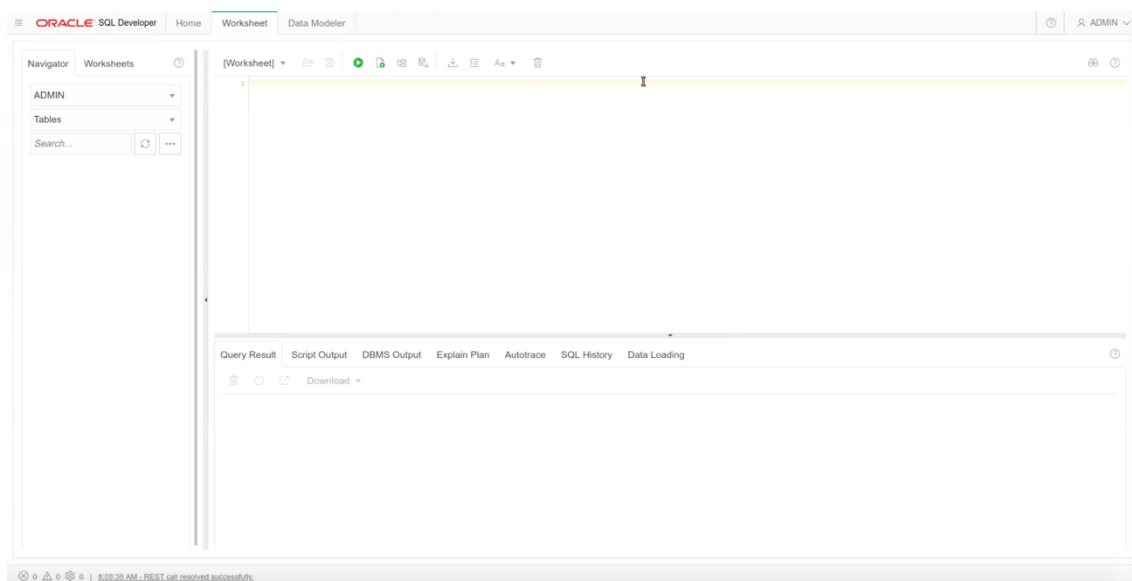


Ahora vemos la pantalla de **Database Actions**, hacemos click en **SQL**.





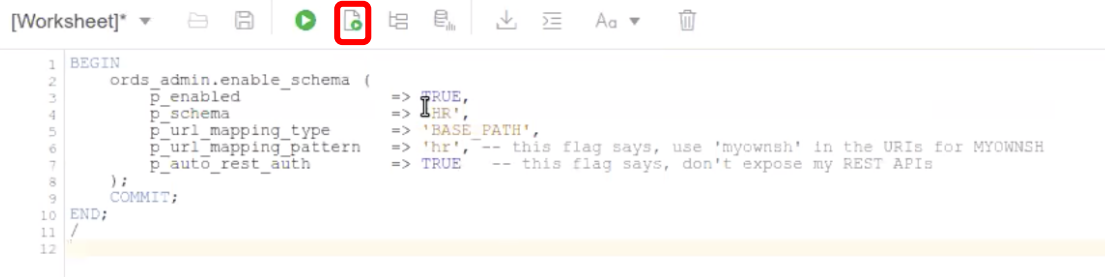
Una vez hecho esto accedemos a la pantalla de **SQL Developer Web**, que nos permitirá interactuar con la base de datos mediante SQL.



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

```
BEGIN
  ords_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;
/
```



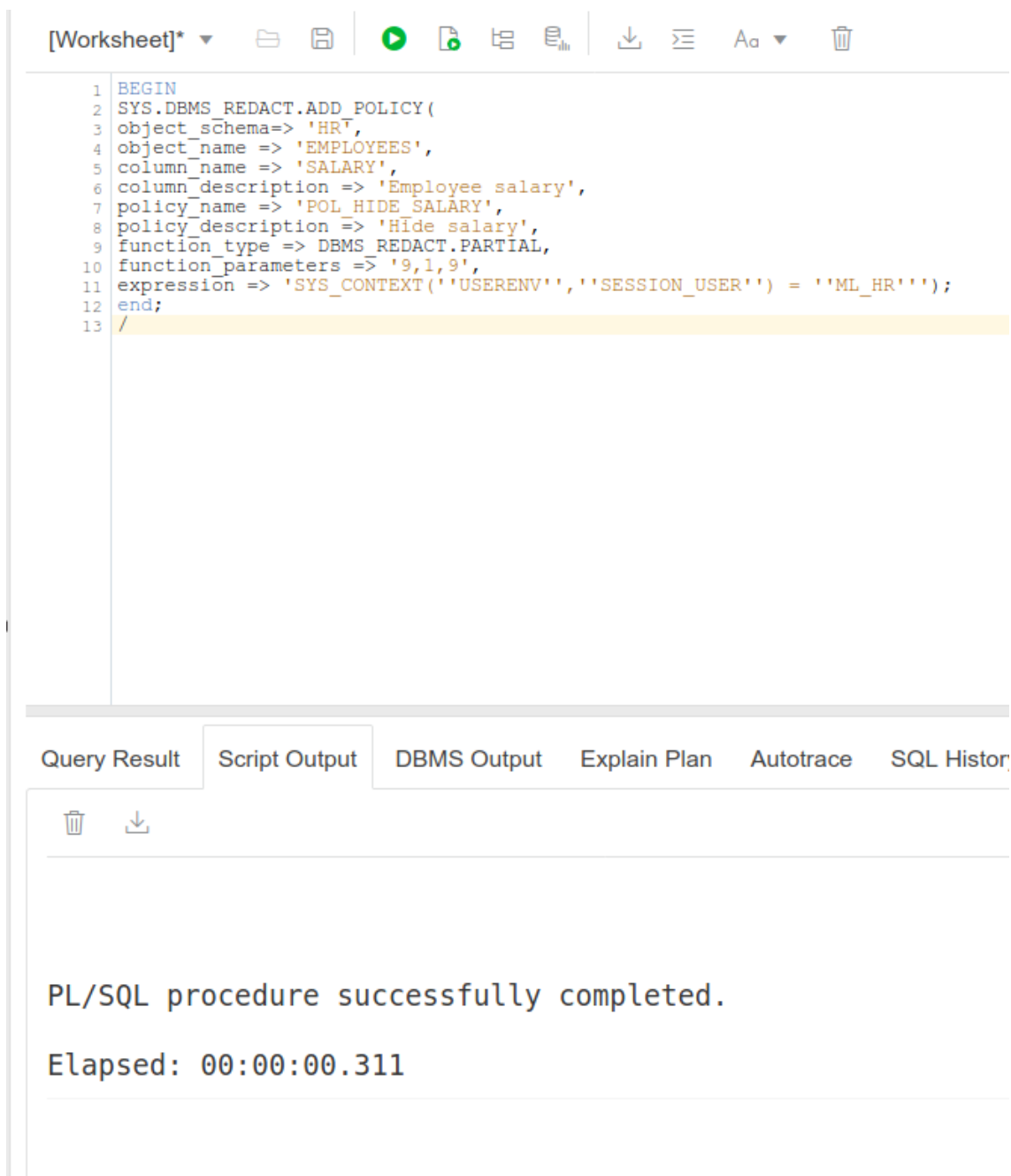


```
1 BEGIN
2   ords_admin.enable_schema (
3     p_enabled => TRUE,
4     p_schema => 'HR',
5     p_url_mapping_type => 'BASE_PATH',
6     p_url_mapping_pattern => 'hr', -- this flag says, use 'myownsh' in the URIs for MYOWNSH
7     p_auto_rest_auth => TRUE -- this flag says, don't expose my REST APIs
8   );
9   COMMIT;
10 END;
11 /
12
```

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 displays the SQL Developer Web interface. At the top, there is a toolbar with icons for file operations, execution, and formatting. Below the toolbar, a code editor contains a PL/SQL script to create a database policy. The script is as follows:

```
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 /
```

Below the code editor, there are tabs for "Query Result", "Script Output", "DBMS Output", "Explain Plan", "Autotrace", and "SQL History". The "Script Output" tab is currently selected, showing the following message:

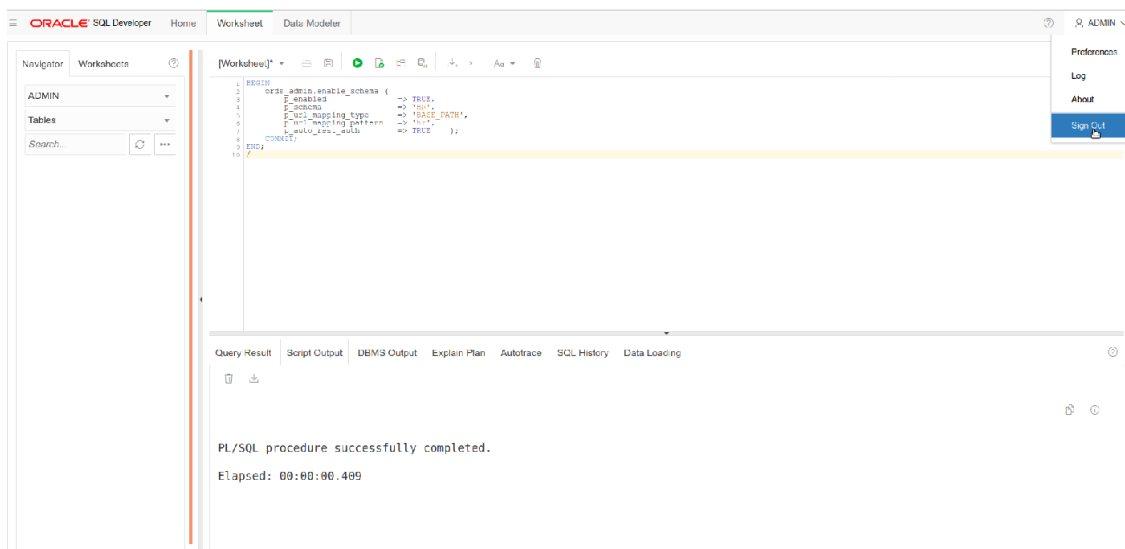
PL/SQL procedure successfully completed.

Elapsed: 00:00:00.311

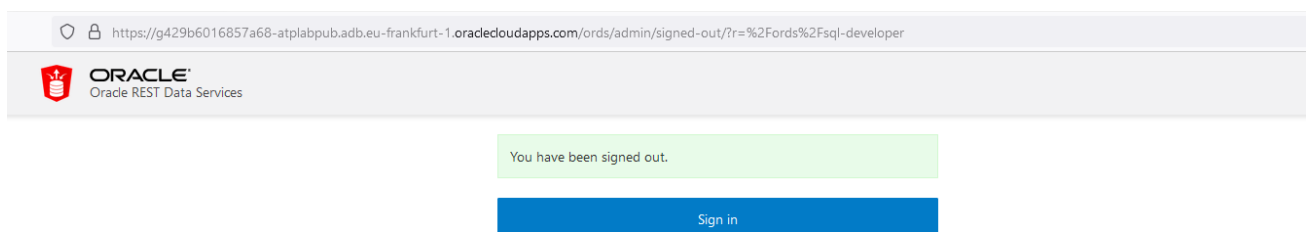
Esta política impedirá que el usuario ML_HR pueda ver el contenido del campo “salary” en la tabla “HR.employees”.

Una vez hecho esto, podemos acceder a SQL Developer Web cambiando el usuario admin por HR. Primero nos desconectamos del Sql*Developer Web, y cerramos la pestaña del navegador.

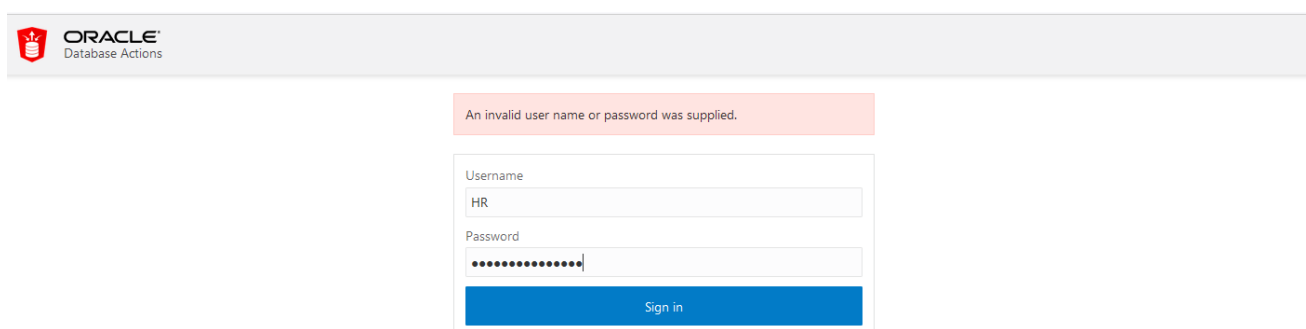




Ahora presionamos el botón **Sing In**.

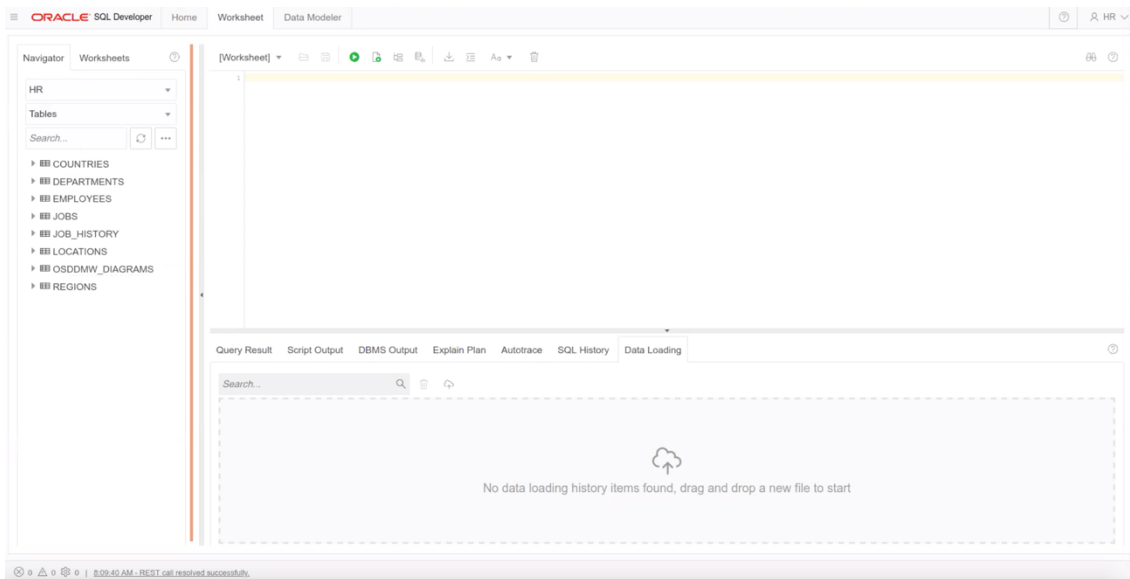


Volverá a aparecer la consola de login e introducir el nombre de usuario y contraseña. En este caso el usuario HR/Autonomous#2020 o hr/Autonomous#2020 .



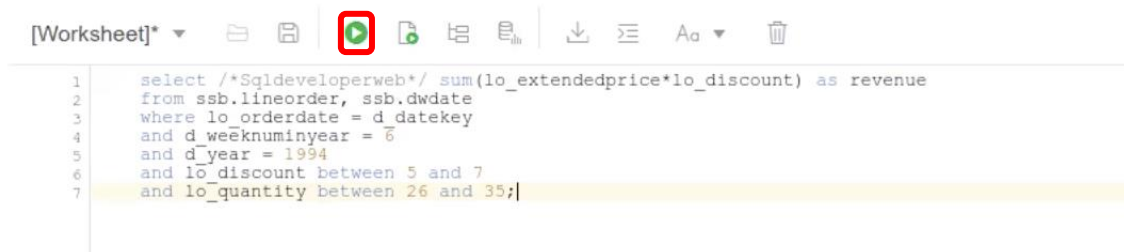
En la siguiente pantalla hacemos click en **“SQL”** para acceder a la misma consola de SQL Developer Web, pero en este caso a la izquierda podemos ver las tablas del esquema HR:



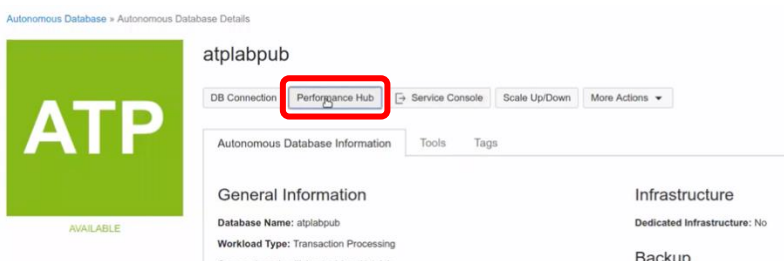


Ejecutamos el siguiente código, como se muestra en la imagen. Es una query sobre el **esquema SSB**, accesible a cualquier usuario:

```
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;
```



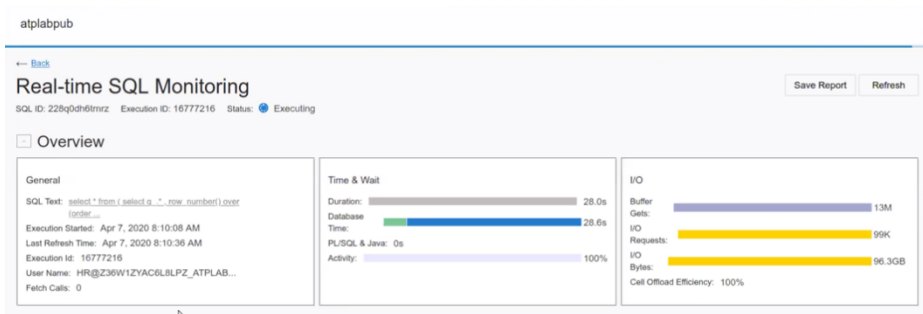
Se puede consultar la ejecución de la query desde la sección Performance HUB, en la pantalla principal del ATP:



Una vez aquí, se puede ver la query ejecutándose:

Status	Duration	Inst ID	SQL ID	SQL Plan Hash	User Name	Parallel	Database Time	I/O Requests	SQL Text
●	20.06s	1	228b0d6trrz	817007416	HR@Z36W1ZYAC6L8LPZ_ATPLABPUB		20.40s	71K	select

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



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

Details

Plan Statistics SQL Text Activity Metrics

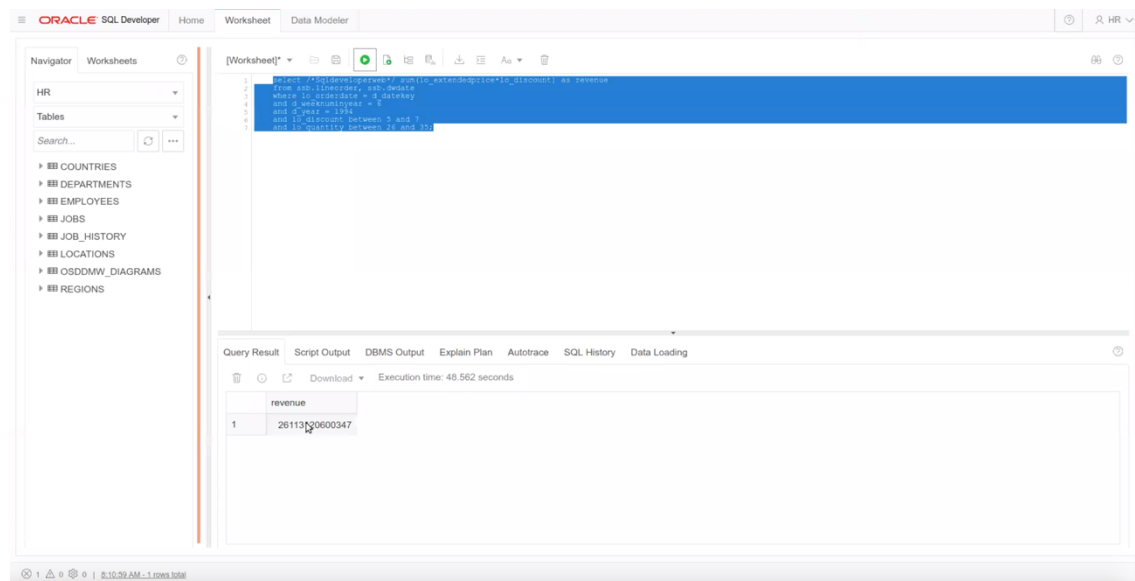
SQL Binds

SQL Text

```
select * from
(select q, row_number() over (order by 1) RN_ from
(select /*+develop*/ sum (a.extendedprice*1-a.discount) as revenue
from shb_linesorder, shb_date
where 1a.orderdate = d_datekey
and d_year = 1994
and 1a.discount between 5 and 7
and 1a.quantity between 25 and 35
) q,
where RN_ between 11 and 12
```



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



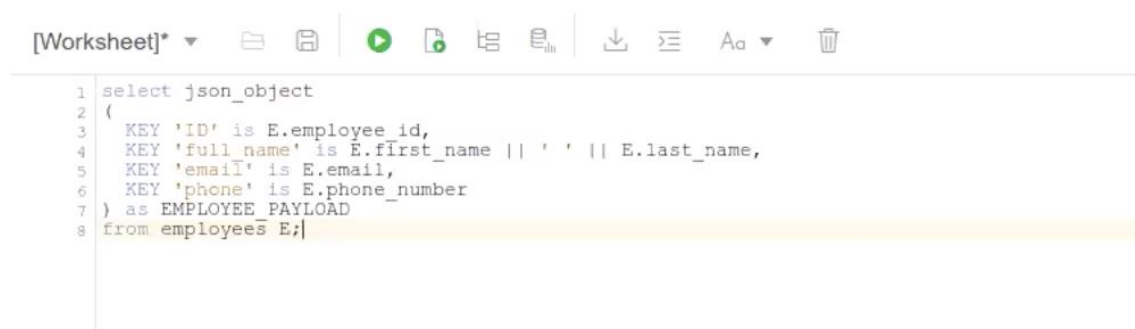
The screenshot shows the Oracle SQL Developer Web interface. On the left is a Navigator pane with a tree view containing 'HR' and 'Tables'. The main workspace displays a SQL query: `select /*SQLDeveloper*/ null as extended_price * 10 discount as revenue from emp, lineitems, shi_devices where li_orderdate = i_datekey and li_shippingkey = 1 and i_email = 'HR' and li_discount between 5 and 7 and li_discount1 between 10 and 100;`. Below the query editor, the 'Query Result' tab is active, showing a table with one row:

revenue
1 2611320600347

. The status bar at the bottom indicates '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:

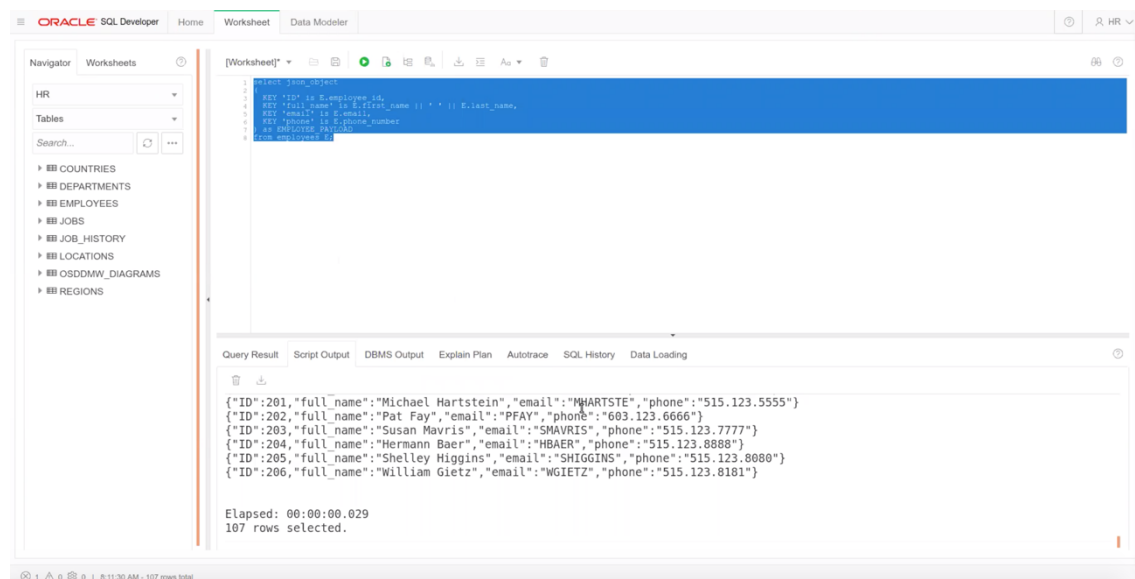
```
select json_object
(
  KEY 'ID' is E.employee_id,
  KEY 'full_name' is E.first_name || ' ' || E.last_name,
  KEY 'email' is E.email,
  KEY 'phone' is E.phone_number
) as EMPLOYEE_PAYLOAD
from employees E;
```



The screenshot shows the Oracle SQL Developer Web interface with a SQL query editor. The query is: `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 query is highlighted in yellow.

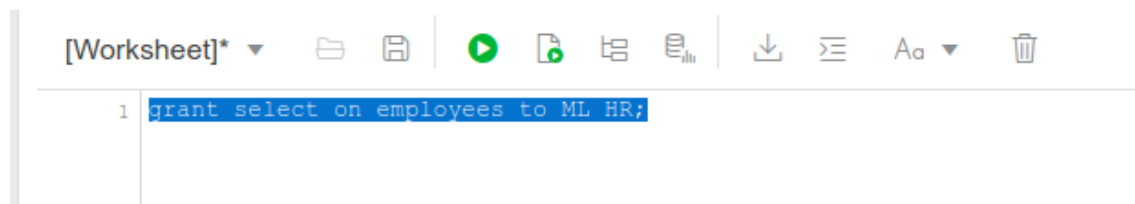


Podemos ver el resultado en formato JSON:



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

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



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

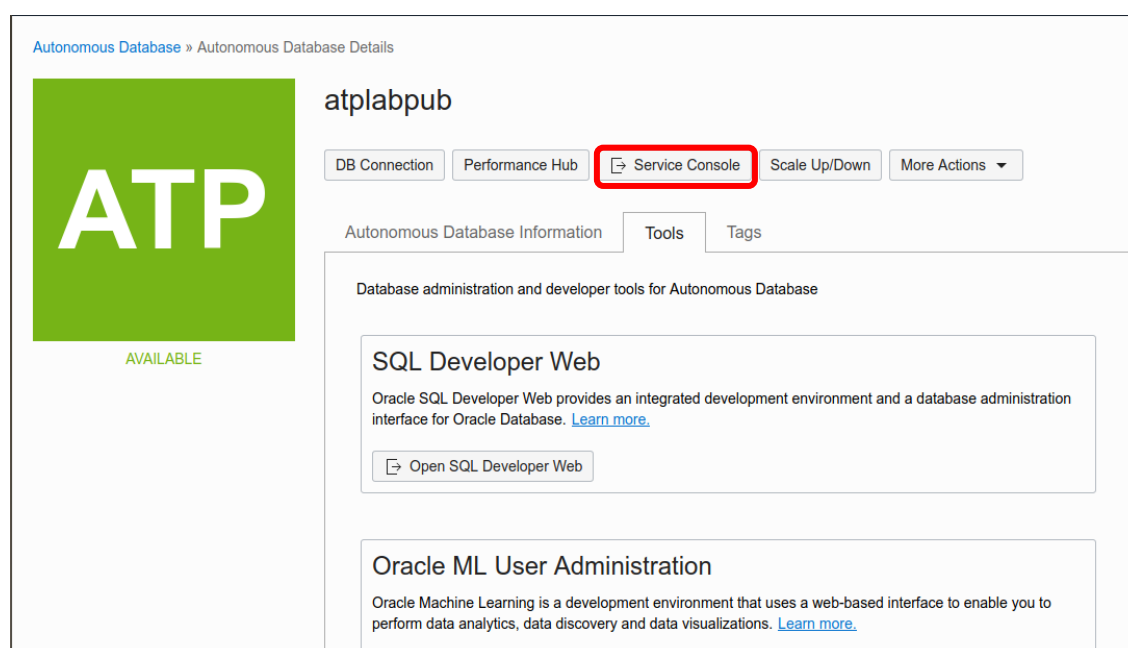
```
DECLARE
  PRAGMA AUTONOMOUS_TRANSACTION;
BEGIN
  ORDS.ENABLE_OBJECT(p_enabled => TRUE,
    p_schema => 'HR',
    p_object => 'EMPLOYEES',
    p_object_type => 'TABLE',
    p_object_alias => 'emp',
    p_auto_rest_auth => FALSE);
  commit;
END;
/
```



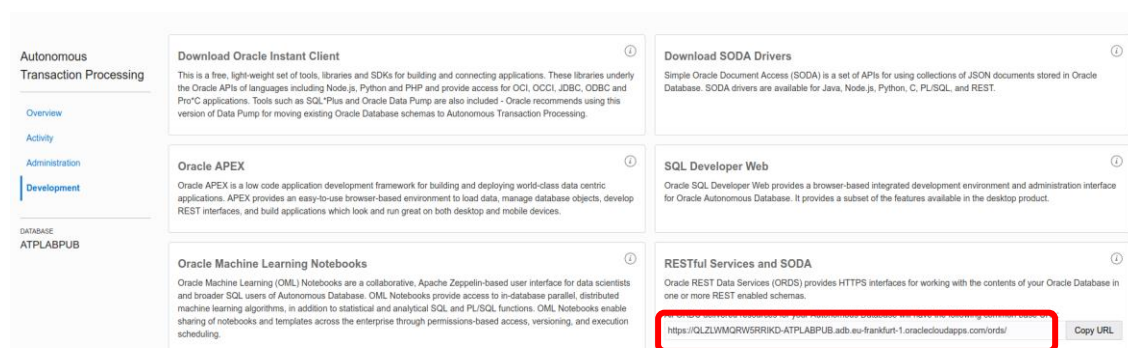
```
[Worksheet]* ▾ | 📁 | 💾 | ▶ | 📄 | 📊 | 📑 | ⬇ | ☰ | Aa ▾ | 🗑
```

```
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”:



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



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 web browser with the URL `https://qlzlwmgw5rrikd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/emp/`. The browser's developer tools are open, displaying the JSON response. The response is a list of two items, each representing an employee record. The first item (ID 100) is for Steven King, and the second item (ID 101) is for Neena Kochhar. The JSON structure includes fields for employee_id, first_name, last_name, email, phone_number, hire_date, job_id, salary, commission_pct, manager_id, and department_id. It also includes a 'links' section with 'self' and 'href' attributes for each record.

```
{
  "items": [
    {
      "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": {
        "self": "https://qlzlwmgw5rrikd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/emp/100"
      }
    },
    {
      "employee_id": 101,
      "first_name": "Neena",
      "last_name": "Kochhar",
      "email": "NKOCHHAR",
      "phone_number": "515.123.4568",
      "hire_date": "2005-09-21T00:00:00Z",
      "job_id": "AD_ASST",
      "salary": 17000,
      "commission_pct": null,
      "manager_id": 100,
      "department_id": 90,
      "links": {
        "self": "https://qlzlwmgw5rrikd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/emp/101",
        "edit": "https://qlzlwmgw5rrikd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/emp/101/edit",
        "describedby": "https://qlzlwmgw5rrikd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/metadata-catalog/emp/item",
        "collection": "https://qlzlwmgw5rrikd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/emp/"
      }
    }
  ]
}
```

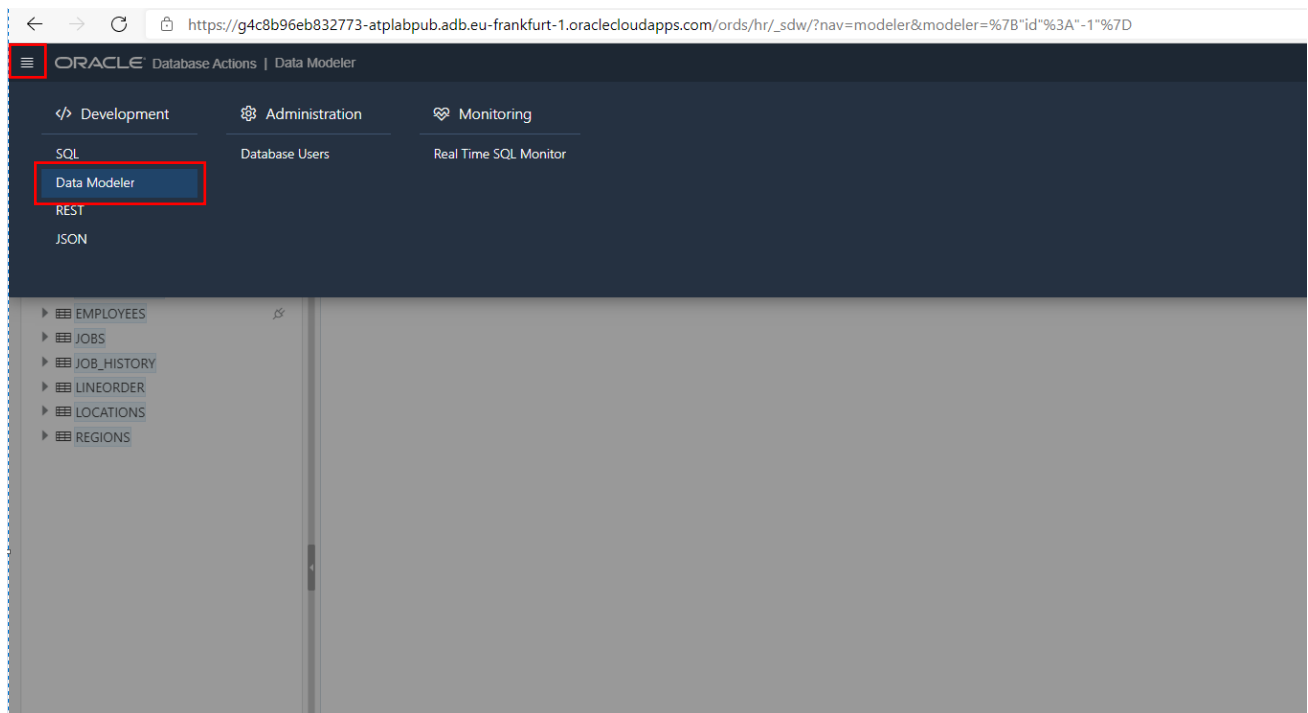
Si queremos ver únicamente el employee ID=100, completamos la URL con “100”:

The screenshot shows a web browser with the URL `https://qlzlwmgw5rrikd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/emp/100`. The browser's developer tools are open, displaying the JSON response. The response is a single employee record for Steven King (ID 100). The JSON structure is identical to the previous screenshot, but it only contains one item. The 'links' section includes 'self', 'edit', 'describedby', and 'collection' attributes.

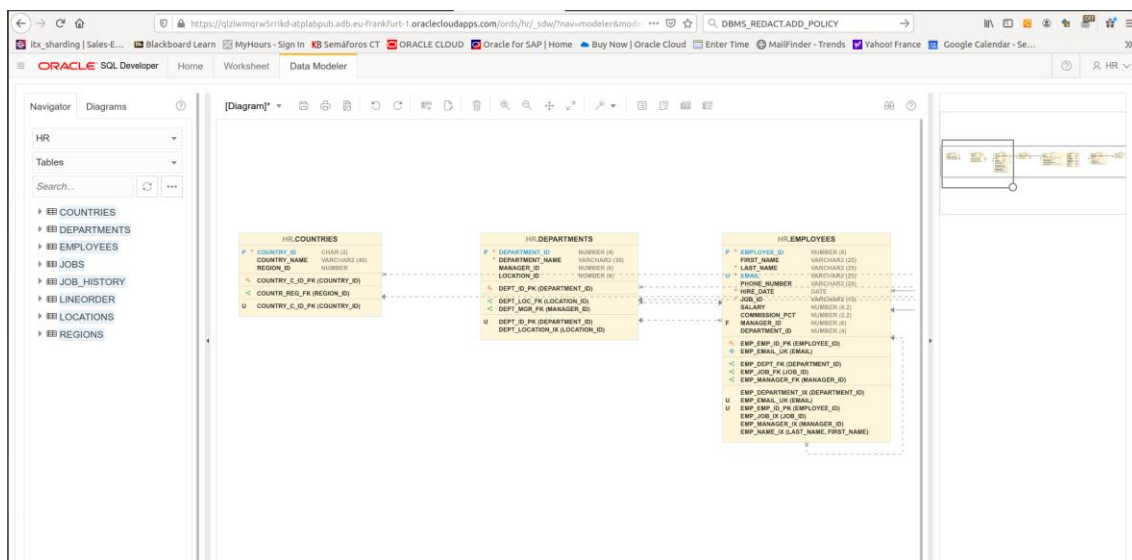
```
{
  "items": [
    {
      "employee_id": 100,
      "first_name": "Steven",
      "last_name": "King",
      "email": "SKING",
      "phone_number": "515.123.4567",
      "hire_date": "2003-06-17T00:00:00Z",
      "job_id": "AD_PRES",
      "salary": 24000,
      "commission_pct": null,
      "manager_id": null,
      "department_id": 90,
      "links": {
        "self": "https://qlzlwmgw5rrikd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/emp/100",
        "edit": "https://qlzlwmgw5rrikd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/emp/100/edit",
        "describedby": "https://qlzlwmgw5rrikd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/metadata-catalog/emp/item",
        "collection": "https://qlzlwmgw5rrikd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/emp/"
      }
    }
  ]
}
```



Finalmente, desde el Sql*Developer Web, hacemos click en icono de las tres rayas paralelas o hamburger y seleccionamos Data Modeler para visualizar el modelo de datos del esquema HR. Arrastramos todas las tablas a la parte central de la pantalla:



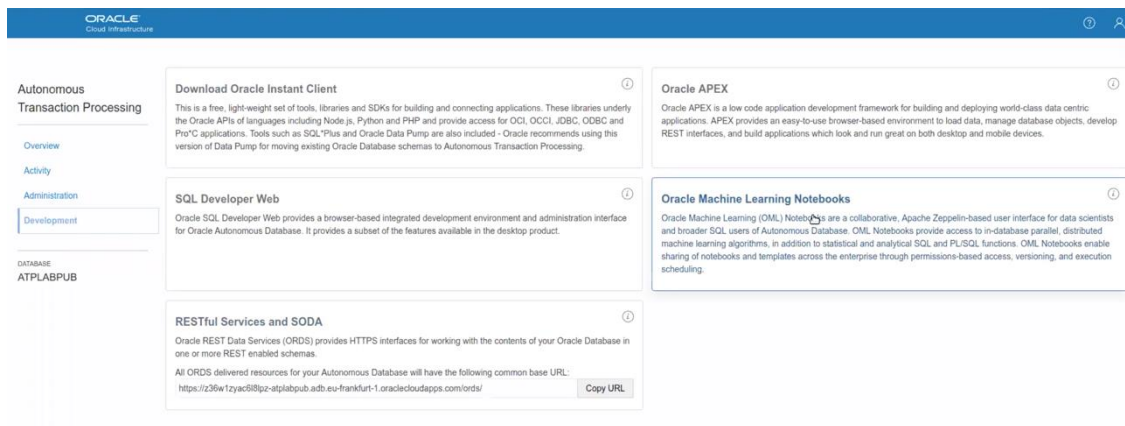
Y visualizamos nuestro modelo relacional:



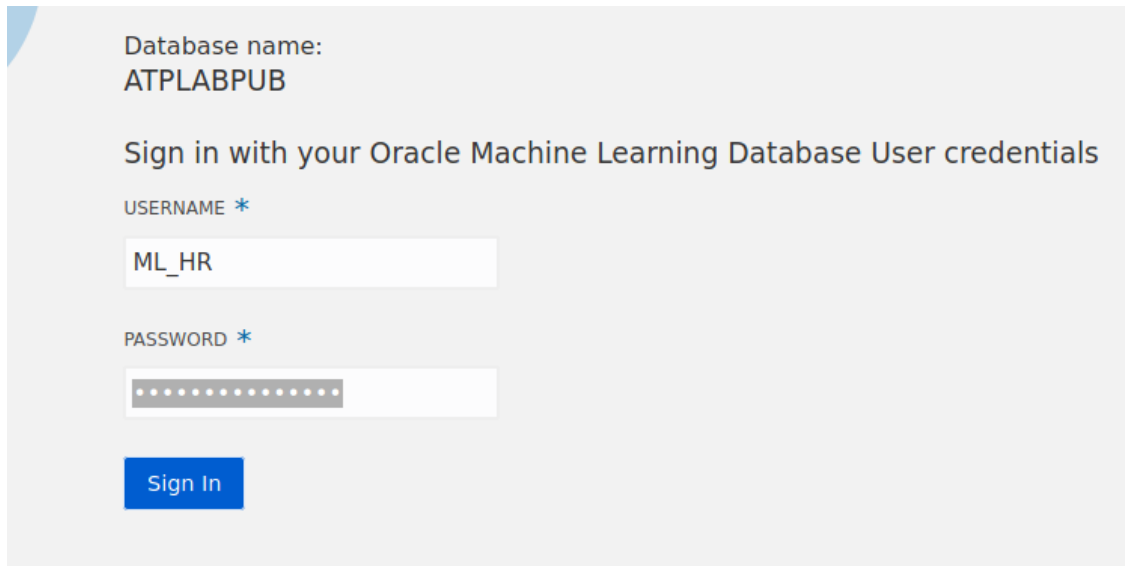
Ejercicio 3: Crear un notebook en Oracle Machine Learning

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”**:

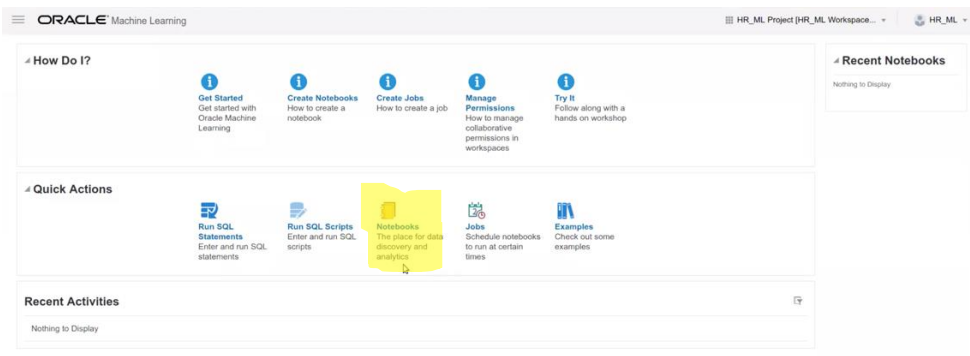


En la pantalla de login, nos conectamos con el usuario **ML_HR/Autonomous#2020**:

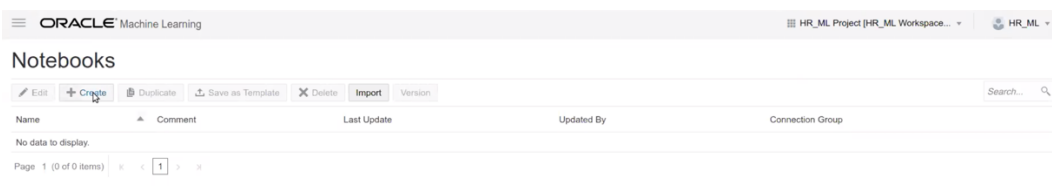


A continuación, aparece la pantalla principal de la sección de Machine Learning, elegimos la opción **“Notebooks”**:

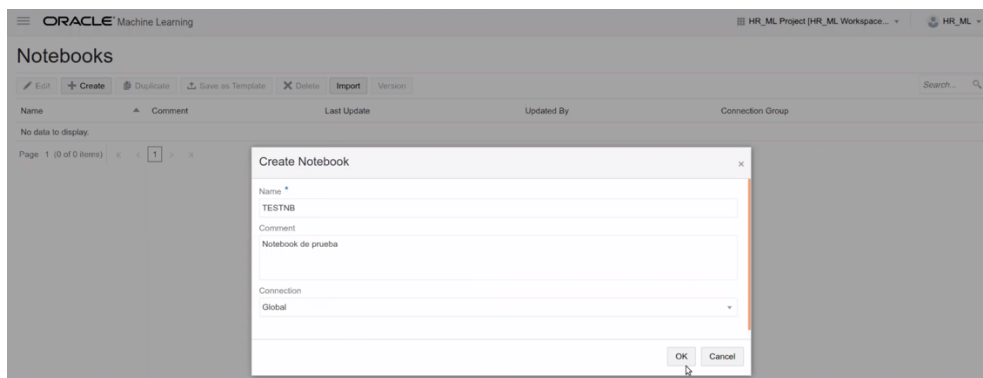




Esto dará paso a la creación de nuestro primer Notebook de Machine Learning. Pulsamos en el botón de crear:



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 pestañas a la izquierda de aquí:



atplabpub

ASH Analytics **SQL Monitoring**

Top 100 by **Last Active Time** Filter by Status, SQL ID or User Name

Kill Session

Status	Duration	Inst ID	SQL ID	SQL Plan Hash	User Name	Parallel	Database Time	I/O Requests	SC
	19.00s	1	9fd9944w9trgc	1992170205	HR_ML@Z36W1ZYAC6L8LPZ_ATPLABPUB	2	37.09s	92K	sel
	48.00s	1	9fd9944w9trgc	3002741515	HR_ML@Z36W1ZYAC6L8LPZ_ATPLABPUB		48.38s	165K	sel

Close

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

atplabpub

ASH Analytics **SQL Monitoring**

Top 100 by **Last Active Time** Filter by Status, SQL ID or User Name

Kill Session

Status	Duration	Inst ID	SQL ID	SQL Plan Hash	User Name	Parallel	Database Time	I/O Requests	SC
	19.00s	1	9fd9944w9trgc	1992170205	HR_ML@Z36W1ZYAC6L8LPZ_ATPLABPUB	2	37.09s	92K	sel
	48.00s	1	9fd9944w9trgc	3002741515	HR_ML@Z36W1ZYAC6L8LPZ_ATPLABPUB		48.38s	165K	sel

Close

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

atplabpub

General SQL Text: <code>select /*MLnotebook*/ sum(to_extendedprice*to_discount)...</code> Execution Plan: 2 Execution Started: Apr 7, 2020 10:32:19 AM Last Refresh Time: Apr 7, 2020 10:32:46 AM Execution ID: 16777220 User Name: HR_ML@Z36W1ZYAC6L8LPZ_ATP... Fetch Calls: 0	Time & Wait Duration: 28.0s Database: 57.6s Time: 57.6s PL/SQL & Java: 0s Activity: 100%	I/O Buffer Gets: 19M I/O Requests: 147K I/O Bytes: 142.9GB Cell Offload Efficiency: 100%
---	--	---

Details

Plan Statistics **Parallel** **SQL Text** Activity Metrics

```

select /*MLnotebook*/ sum(to_extendedprice*to_discount) as revenue
from sdb_lineitem, sdb_supplier
where to_orderdate = d_datekey
and d_weekmonthyear = 5
and d_year = 1994
and to_discount between 5 and 7
and to_quantity between 26 and 35

```

Close



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



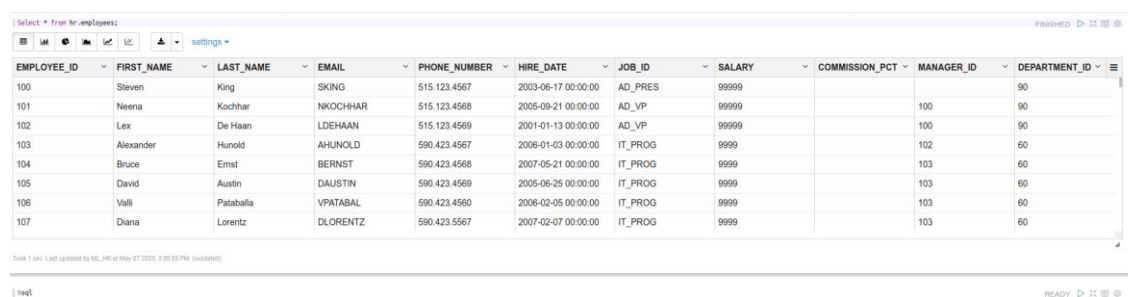
The screenshot shows the Oracle Machine Learning interface. At the top, there's a header with 'ORACLE Machine Learning' and 'HR_ML Project [HR_ML Workspace...]'.

The main area displays a query result for the 'REVENUE' table. The query is: `select 'REVENUE' as(revenue) from hr.employees, job_history where h.job_id = e.job_id and e.salary > 1000 and h.quantity between 5 and 7 and h.quantity between 20 and 30;`

The result is a single row with the value 2611319600347.

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

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

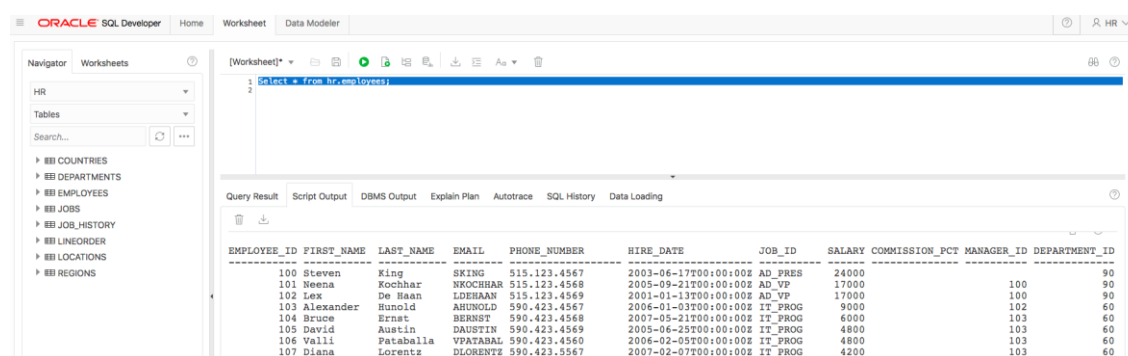


The screenshot shows the Oracle Machine Learning interface displaying the result of the query 'Select * from hr.employees;'. The result is a table with 10 columns: EMPLOYEE_ID, FIRST_NAME, LAST_NAME, EMAIL, PHONE_NUMBER, HIRE_DATE, JOB_ID, SALARY, COMMISSION_PCT, MANAGER_ID, and DEPARTMENT_ID. The SALARY column is masked with '9'.

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	9999			90
101	Neena	Kochhar	NKOCHHAR	515.123.4568	2005-09-21 00:00:00	AD_VP	9999		100	90
102	Lex	De Haan	LDEHAAN	515.123.4569	2001-01-13 00:00:00	AD_VP	9999		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.



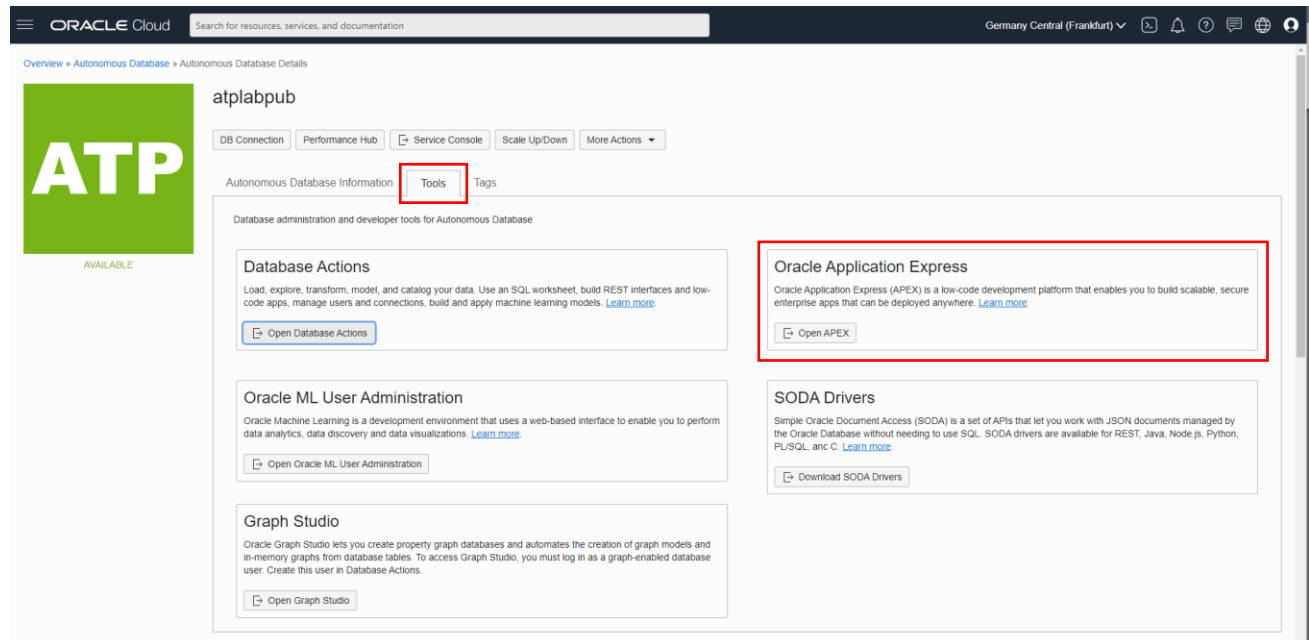
The screenshot shows the Oracle SQL Developer interface displaying the result of the query 'Select * from hr.employees;'. The result is a table with 10 columns: EMPLOYEE_ID, FIRST_NAME, LAST_NAME, EMAIL, PHONE_NUMBER, HIRE_DATE, JOB_ID, SALARY, COMMISSION_PCT, MANAGER_ID, and DEPARTMENT_ID. The SALARY column is masked with '9'.

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

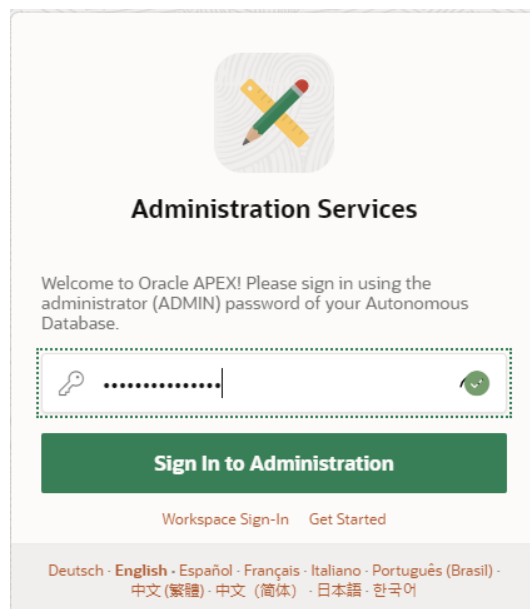


Ejercicio 4: Utilización de APEX

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

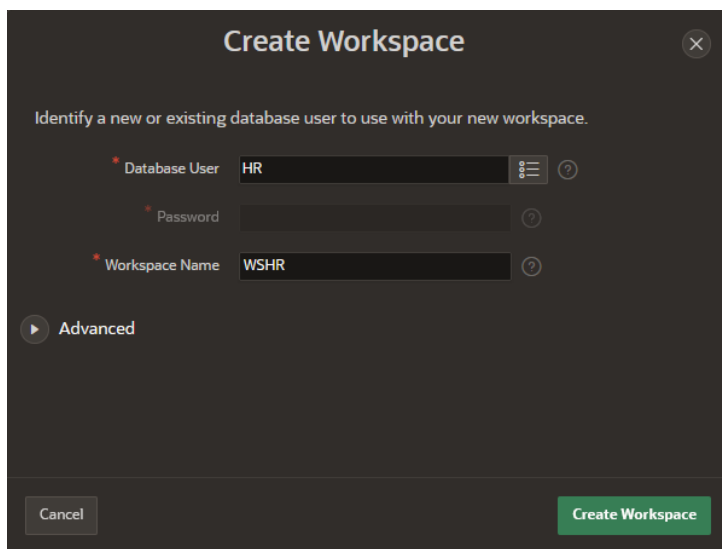
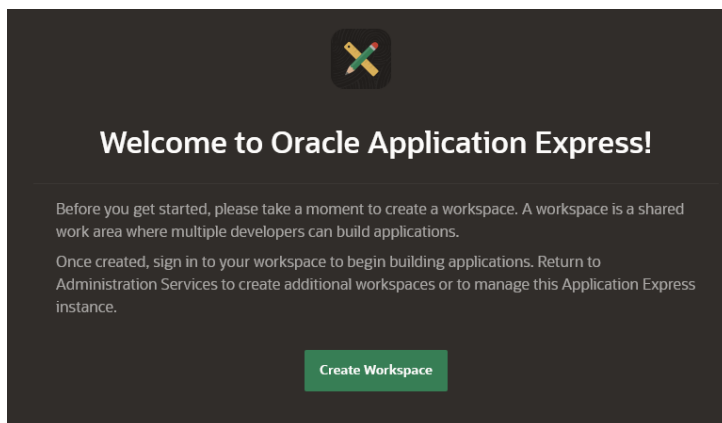


Primero nos conectamos con el usuario ADMIN a la consola de administración de APEX:

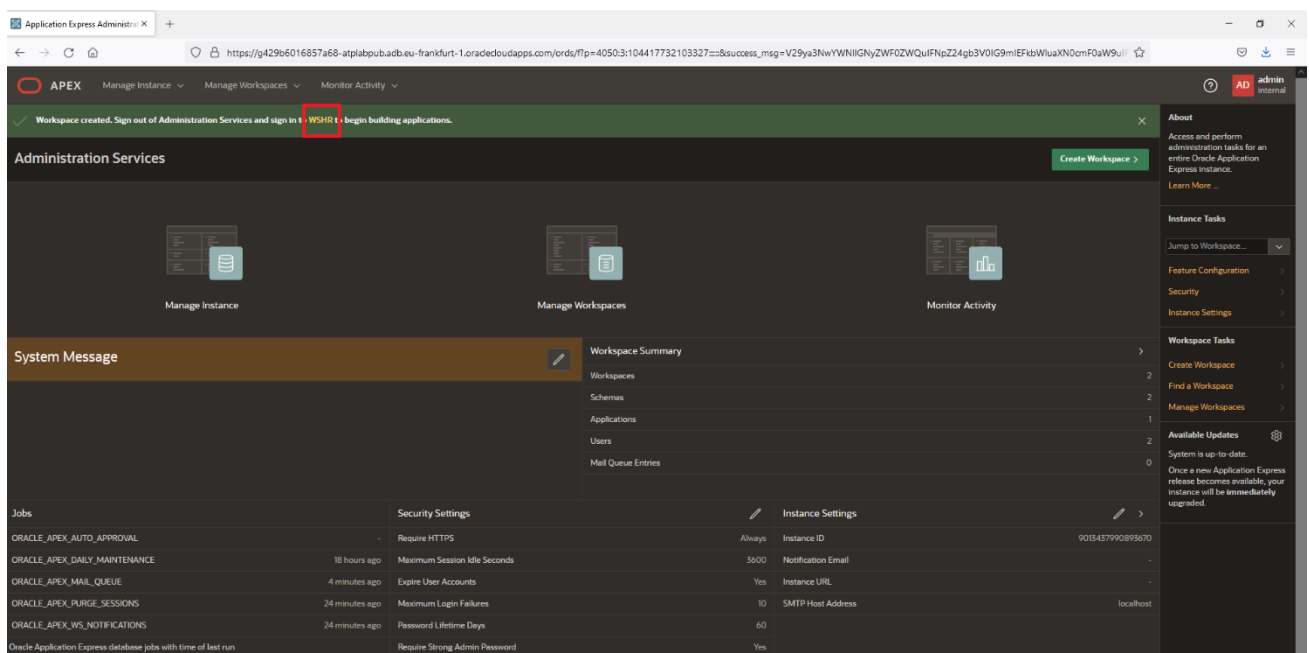


A continuación, creamos un “Workspace” con nombre WSHR para el usuario “HR”

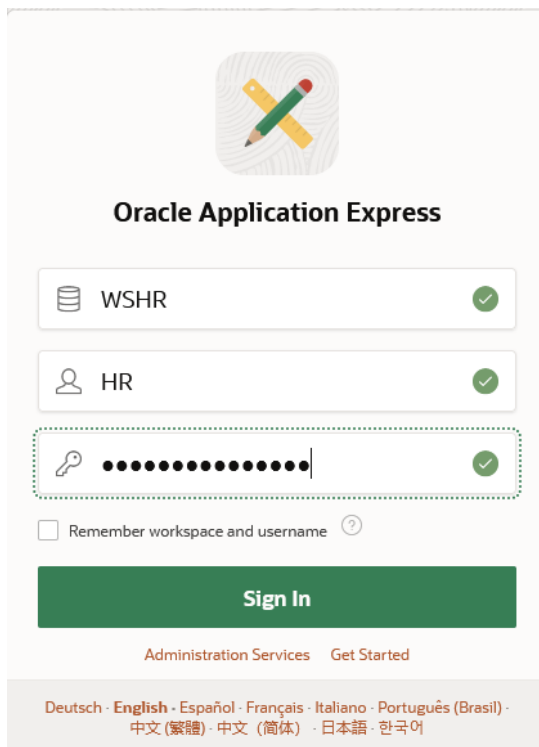




Una vez creado el Workspace, nos conectamos a APEX con el usuario HR, siguiendo el enlace arriba a la izquierda en la pantalla principal de APEX.



Y nos conectamos como HR/Autonomous#2020 o hr/Autonomous#2020 (**contraseña siempre en minúsculas**):



Oracle Application Express

WSHR ✓

HR ✓

●●●●●●●●●● ✓

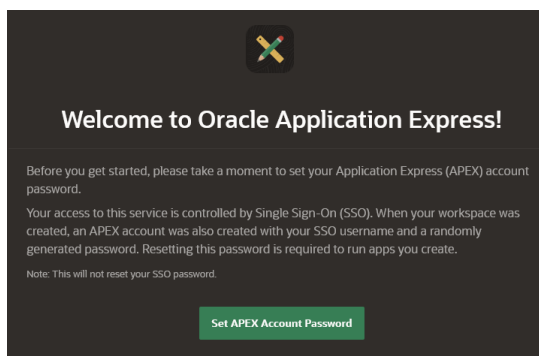
☐ Remember workspace and username ?

Sign In

Administration Services Get Started

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

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

Email Address:

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.

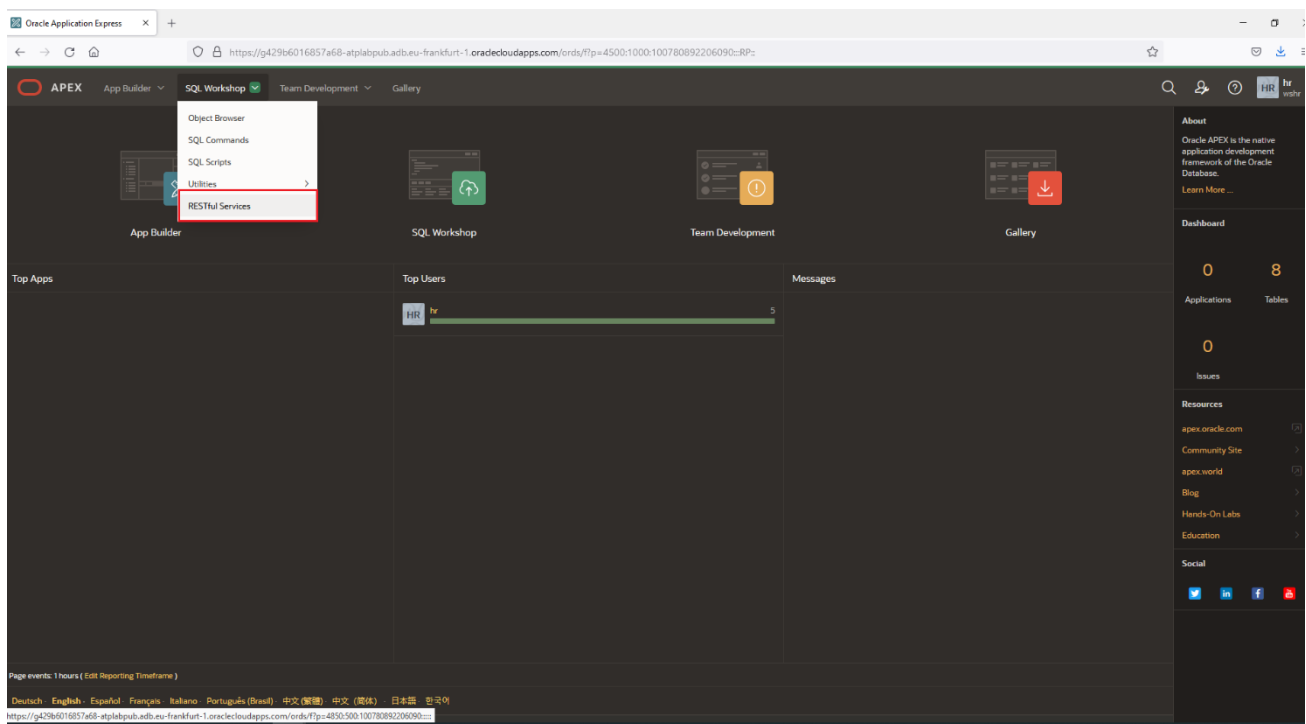
Photo:

Password (For authentication against workspace user account repository only)

If you wish to change your password, enter a new password. Otherwise, leave the password fields empty and the current password will not be changed.

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”**:



The screenshot shows the APEX RESTful Services interface. The left sidebar contains a navigation menu with 'RESTful Data Services', 'Enabled Objects', 'Modules', 'Privileges', and 'Roles'. The main content area shows the 'Enabled Objects' section for the 'HR' schema. It includes a 'De-Register Schema from ORDS' button, 'Install Sample Service', 'Import', and 'Configure' buttons. The dashboard displays the following information:

- Schema Access:** Access Status **ENABLED** (green checkmark icon).
- Metadata Access:** Authorization Required **ENABLED** (green checkmark icon).
- Schema Aliased:** Schema Alias **hr** (yellow warning icon).
- Modules:** 0 Total Modules.
- Privileges:** 4 Total Privileges.
- Roles:** 8 Total Roles.
- Enabled Objects:** 1 Total Enabled Objects (green circle with '1' and a mouse cursor pointing to it).
- Module Status:** No Modules Defined.
- Module Security:** No Modules Defined.
- Object Aliases:** A large teal circle.

The screenshot shows the APEX RESTful Services interface with the 'RESTful Enabled Objects' table. The table has the following columns: Parsing Schema, Parsing Object, Object Alias, Type, Status, Auto REST Auth, Ops Allowed, Type Path, and Aliased. The table contains one row of data for the 'HR' schema.

Parsing Schema	Parsing Object	Object Alias	Type	Status	Auto REST Auth	Ops Allowed	Type Path	Aliased
HR	EMPLOYEES	emp	TABLE	ENABLED	DISABLED	-	ENABLED	✓

Below the table, there is a legend:

- ✓ Object name and alias are different
- ✗ Object name and alias are the same

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



ORDS Enabled Objects

SQL Workshop

Object Browser

SQL Commands

SQL Scripts

Utilities

RESTful Services

RESTful Services

ORDS RESTful Services

RESTful Data Services

Enabled Objects

Modules

Privileges

Roles

Schema

HR

Go

1 Primary Report

Actions

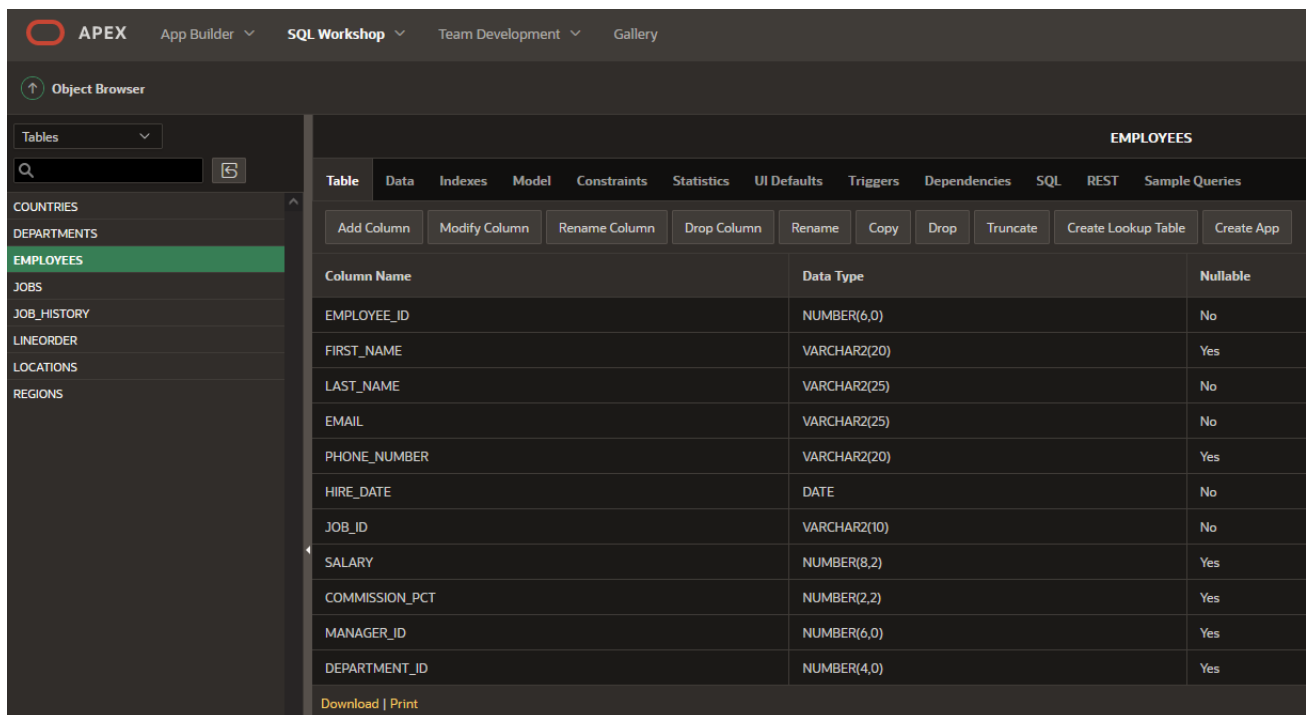
Parsing Schema	Parsing Object	Object Alias	Type	Status	Auto REST Auth	Ops Allowed	Type Path	Aliased
HR	EMPLOYEES	emp	TABLE	ENABLED	DISABLED	-	ENABLED	<div></div>

Legend:

Object name and alias are different
 Object name and alias are the same



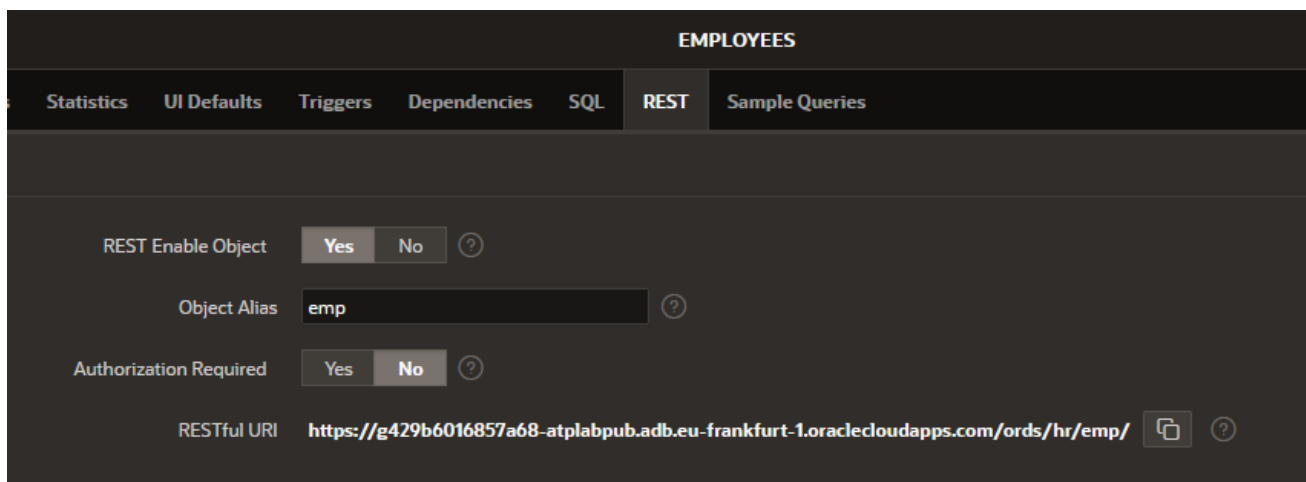
Esto nos lleva a una pantalla donde vemos los objetos del esquema HR. Pulsamos en el objeto “EMPLOYEES”, y accedemos a la **pestaña REST**.



The screenshot shows the APEX SQL Workshop interface. On the left, the 'Object Browser' pane lists database objects, with 'EMPLOYEES' selected. The main pane displays the 'EMPLOYEES' table structure with columns and their data types.

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**:



The screenshot shows the 'REST' configuration page for the 'EMPLOYEES' table. It includes settings for 'REST Enable Object', 'Object Alias', 'Authorization Required', and the 'RESTful URI'.

REST Enable Object: ☒ Yes ☐ No

Object Alias:

Authorization Required: ☒ Yes ☐ No

RESTful URI: <https://g429b6016857a68-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://qlzlwqmrw5rrikd-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://qlzlwqmrw5rrikd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/emp/100"}, {"rel":"edit","href":"https://qlzlwqmrw5rrikd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/emp/100"}, {"rel":"describedby","href":"https://qlzlwqmrw5rrikd-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/emp/100"}]}
```



```
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/"}}
```



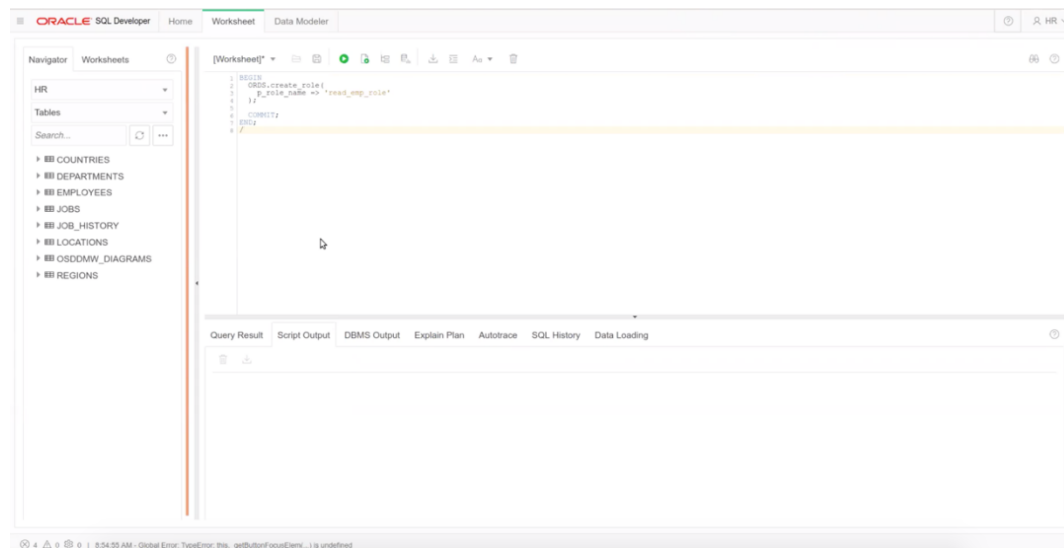
Ejercicio 5 (opcional): Configuración de seguridad de acceso OAuth2

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;
/
```

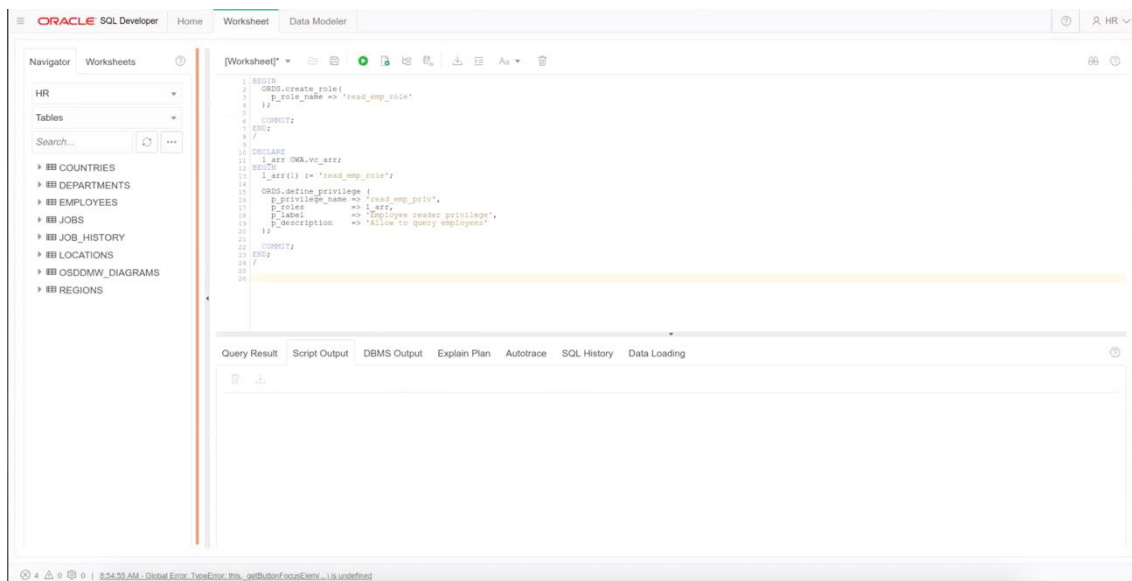


A continuación, se crea un privilegio en ORDS. Este privilegio lo asociamos al role creado en el paso anterior:

```
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;
/
```

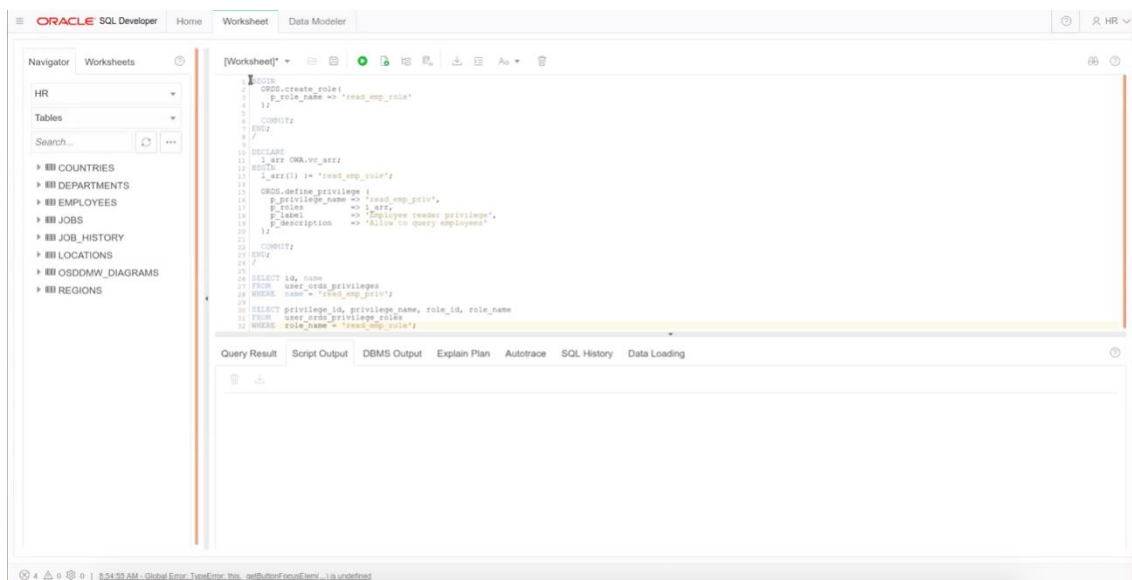




Con las siguientes queries, comprobamos que el rol ha sido correctamente asociado al privilegio de ORDS:

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

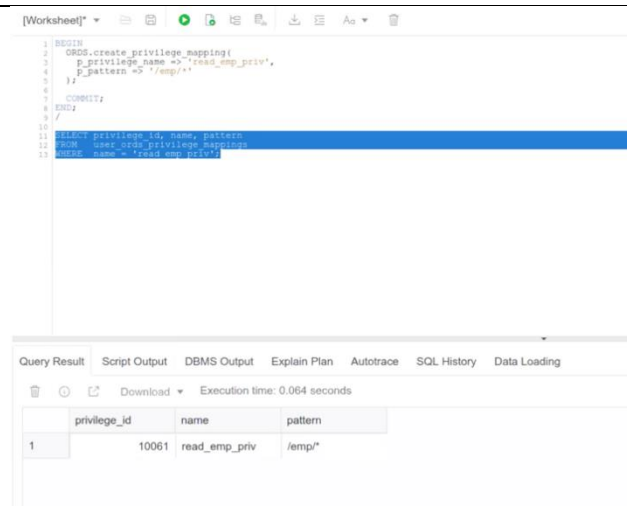
```
SELECT privilege_id, privilege_name, role_id, role_name
FROM user_ords_privilege_roles
WHERE role_name = 'read_emp_role';
```



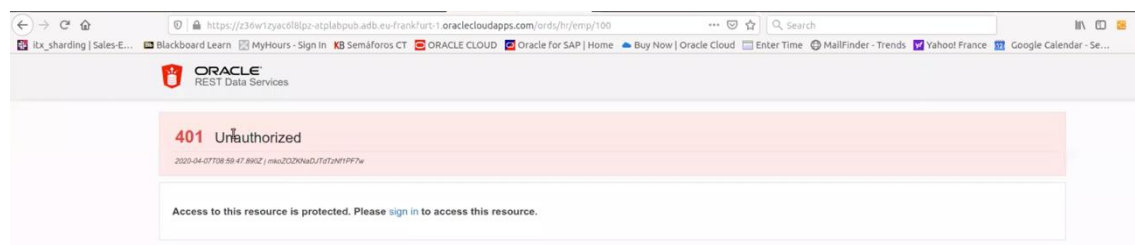
A continuación, mapeamos el privilegio a todas las terminaciones del endpoint “/emp/*”.

```
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';
```



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.



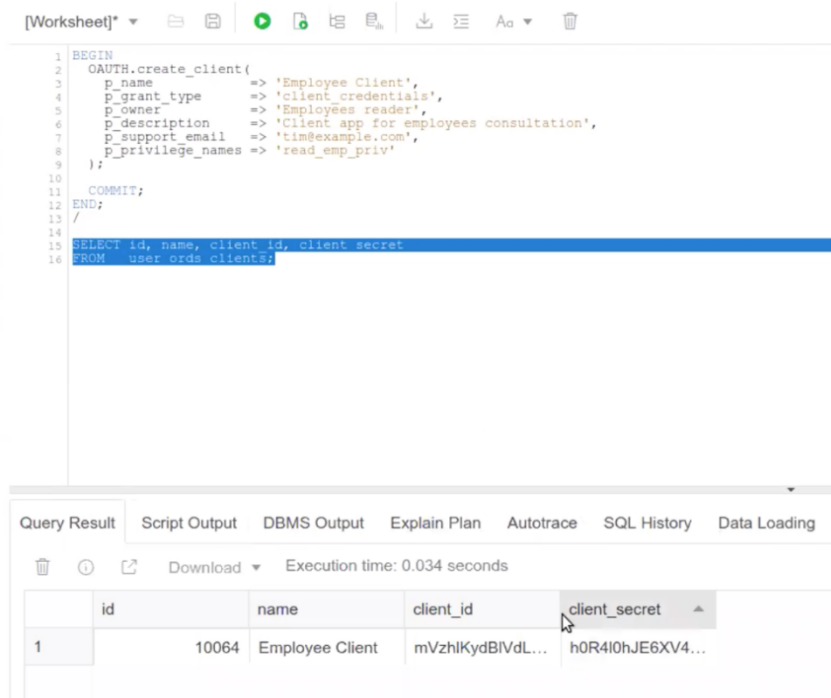
El siguiente paso es crear un token de autorización, valido durante una hora desde su ultima utilización.

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_orcs_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 the Oracle SQL Developer interface. The top pane displays a SQL script with the following content:

```
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_orcs_clients;
```

The bottom pane shows the 'Query Result' tab with the following data:

	id	name	client_id	client_secret
1	10064	Employee Client	mVzhIKydBIVdL...	h0R4I0hJE6XV4...

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_orcs_client_roles;
```



The screenshot shows the SQL Developer interface. The top pane contains SQL code for creating a client, granting a role, and querying the results. The bottom pane shows the 'Query Result' tab with a table containing one row of data.

```

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
11 COMMIT;
12 END;
13 /
14
15 SELECT id, name, client_id, client_secret
16 FROM   user_ords_clients;
17
18 SELECT name, client_name
19 FROM   user_ords_client_privileges;
20
21 BEGIN
22   OAUTH.grant_client_role(
23     p_client_name => 'Employee Client',
24     p_role_name => 'read_emp_role'
25   );
26
27 COMMIT;
28 END;
29 /
30
31 SELECT client_name, role_name
32 FROM   user_ords_client_roles;

```

	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_ords_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://z36w1zyac6l8lpz-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://z36w1zyac6l8lpz-atplabpub.adb.eu-frankfurt-1.oraclecloudapps.com/ords/hr/emp/100
```

