

Workshop Spatial, Graph y Machine Learning en base de Datos Oracle

HOL 2 Machine Learning

Contenidos

| | |
|---|----------|
| WORKSHOP SPATIAL, GRAPH Y MACHINE LEARNING EN BASE DE DATOS ORACLE | 1 |
| HOL 2 MACHINE LEARNING..... | 1 |
| REQUERIMIENTOS INICIALES..... | 3 |
| ESCRITORIO REMOTO CON MICROSOFT WINDOWS | 3 |
| ESCRITORIO REMOTO CON MacOS | 4 |
| MACHINE LEARNING | 7 |
| DESCRIPCIÓN Y OBJETIVO DEL TALLER | 7 |
| OML4R (1 HORA)..... | 8 |
| Acceso a RStudio | 8 |
| Contenido de los notebooks | 15 |
| RESUMEN..... | 17 |
| MAS INFORMACIÓN..... | 18 |
| OML4PY (1 HORA)..... | 19 |
| Acceso a Jupyter | 19 |
| Contenido de los notebooks | 25 |
| RESUMEN..... | 28 |



Requerimientos iniciales

Para la realización de este workshop se necesita un cliente de *Remote Desktop* de Windows.

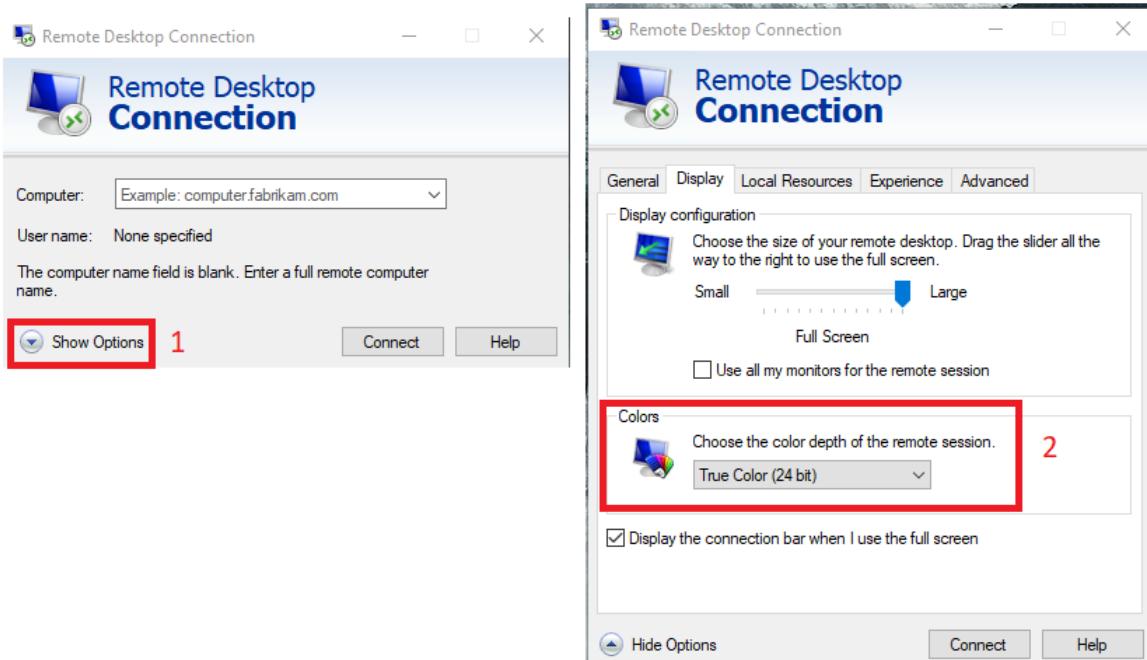
Este cliente está instalado por defecto en el sistema operativo Microsoft Windows, existiendo también clientes compatibles para MacOs y Linux.

La máquina del workshop se proporciona para uso individual de cada participante del workshop siendo para uso exclusivo del mismo.

Esta máquina está alojada en la nube pública **Oracle Cloud Infrastructure** (OCI) estando prohibida la reproducción o alteración de sus contenidos fuera de lo previsto en este manual de usuario.

Escritorio Remoto con Microsoft Windows

En la configuración del cliente es importante especificar el uso de *True Color (24 bit)* para evitar problemas con algunas herramientas. Desde el botón *Show Options*, como se muestra a continuación:



Como parte de la documentación del workshop se facilitarán los siguientes datos:

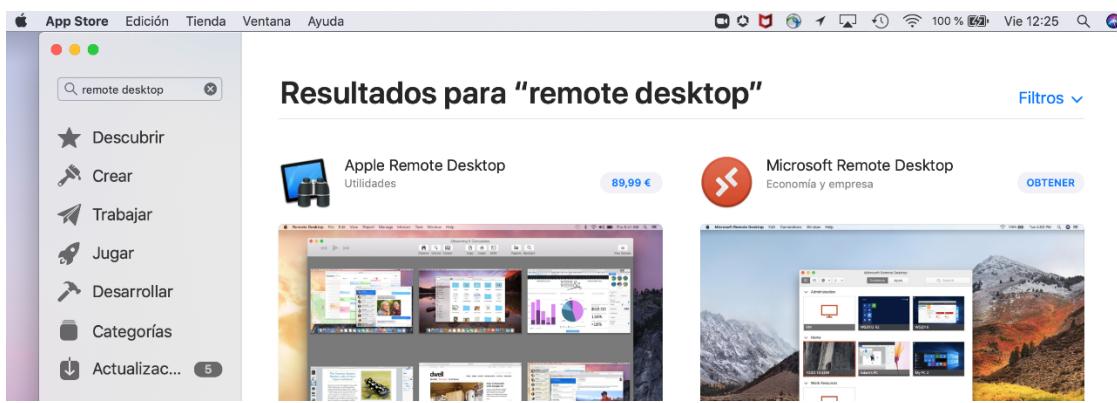


- Nombre de usuario y password
- IP pública de la maquina del workshop

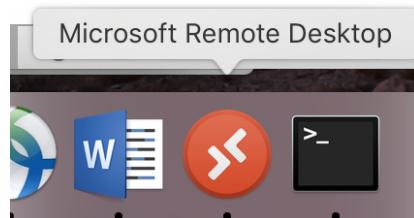
Escritorio Remoto con MacOS

En MacOs la aplicación para conectar a escritorio remoto de Windows no viene instalada por defecto, pero está disponible en el App Store de manera gratuita.

Buscando “remote desktop” se encuentra como “*Microsoft Remote Desktop*” tal y como se muestra a en la siguiente captura:



Una vez instalada esta aplicación, aparecerá un ícono como el siguiente en la barra de aplicaciones para poder realizar las conexiones.

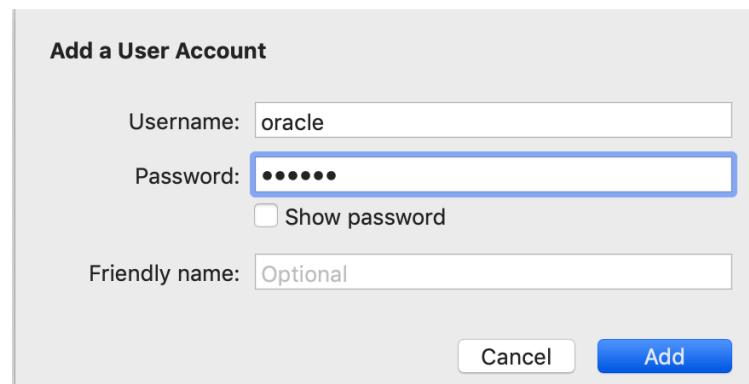
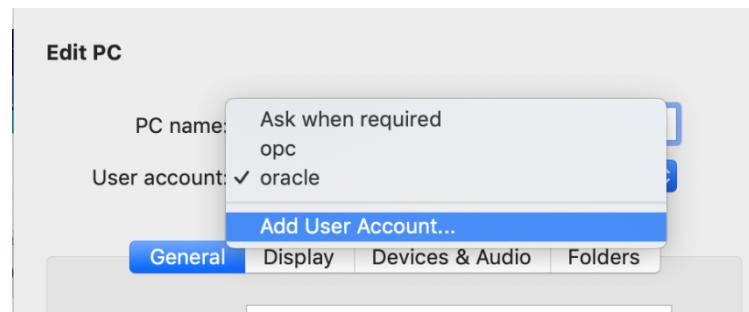


Como parte de la documentación del workshop se facilitarán los siguientes datos:

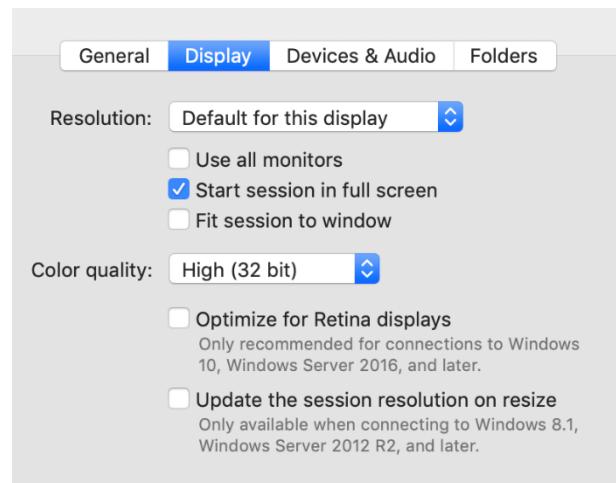
- Nombre de usuario y password
- IP pública de la maquina del workshop

Con los cuales se configura como se muestra a continuación.
Añadimos el usuario y la password proporcionados:



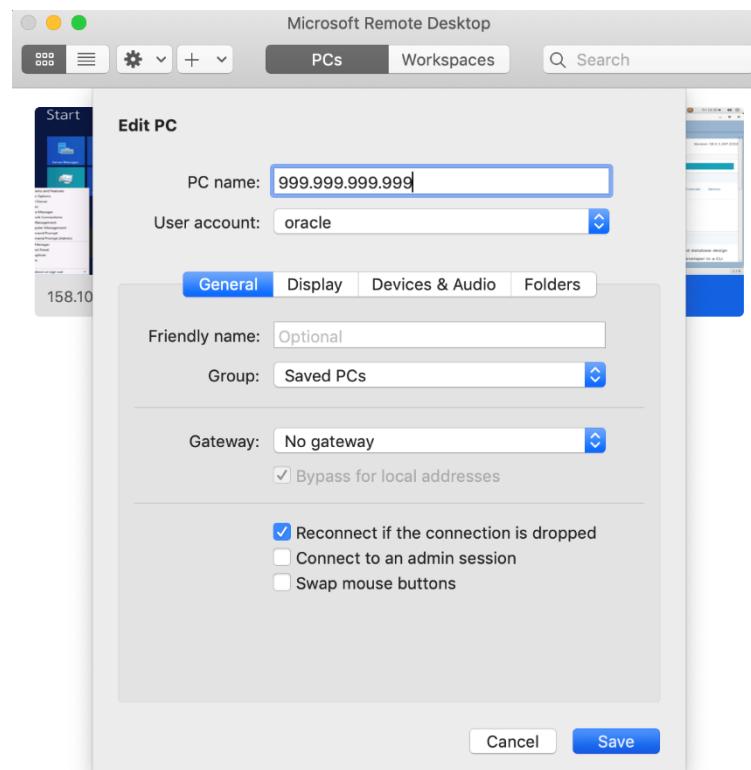


En la pestaña de Display se confirma que la calidad de color está seleccionada a 32 bits:



Se introduce la IP pública en ‘PC name’, se guarda el acceso con el botón Save y ya está preparado el acceso a la máquina virtual del workshop.





Machine Learning

La funcionalidad de Machine Learning está disponible en dos paquetes diferentes:

- **OML4Py** - Python (<https://www.python.org/>)
- **OML4R** - R (<https://cran.r-project.org/>)

Cada uno de ellos emplea un lenguaje de programación y unas herramientas de desarrollo diferentes, que están descritas en las siguientes secciones.

Descripción y objetivo del taller

R y **Python** son los lenguajes preferidos de los científicos de datos. Ambos tienen una amplia comunidad de usuarios y entusiastas, que han generado gran cantidad de paquetes y código los cuales están a disposición de todos.

Todos los algoritmos de Machine Learning necesitan datos para poder entrenarse y ‘aprender’, es por ello por lo que a lo largo de este taller va a comprobar cómo OML4Py y OML4R facilitan el uso de datos almacenados en base de datos Oracle en estos lenguajes de programación. Es lo que denominamos llevar los algoritmos al dato y no al revés.

Para hacerlo, usará **RStudio** en el caso de R y **Jupyter** en el caso de Python, pero en general usted podrá usar aquel IDE con el que se encuentre más cómodo cuando trabaja con estos lenguajes.

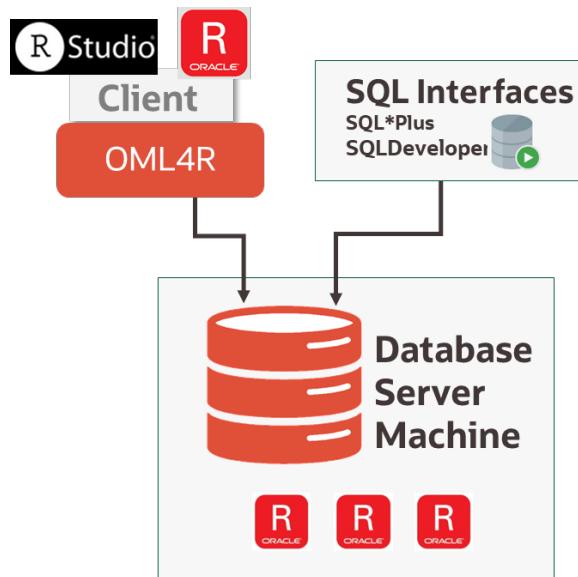
A continuación, puede ver las diferentes actividades que usted llevará a cabo en esta sesión.



OML4R (1 hora)

Todas las actividades de esta sección se realizan a través de la herramienta **RStudio Server** (<https://rstudio.com>), la cual debe ser accedida usando un navegador web.

Para esta sección de OML4R los elementos implicados están representados en el siguiente diagrama:



Acceso a RStudio

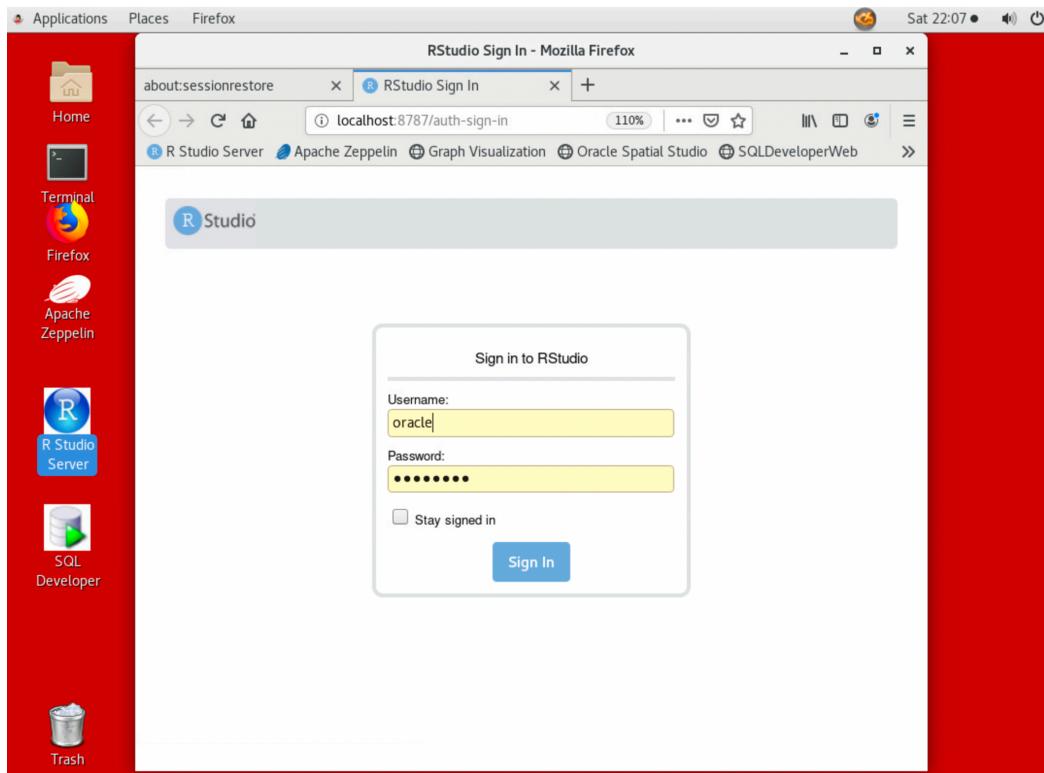
Siga las instrucciones proporcionadas para acceder al escritorio remoto. Se le proporcionará una dirección IP y un usuario/clave.

En el escritorio remoto hay un acceso a *RStudio*, haga doble click para acceder:





Aparecerá una ventana de navegador web con la página de acceso a *RStudio*.



Una vez introducidas las credenciales que se le proporcionen para el curso, aparecerá la interfaz de usuario con varias áreas de trabajo y una sesión de R lista para comenzar las actividades.

En la carpeta del usuario están precargados los notebooks necesarios para el workshop, con los siguientes nombres:

- ORE.first.Rmd
- ORE.second.Rmd
- ORE.third.Rmd
- ORE.fourth.Rmd
- ORE.last.Rmd
- ORE.SQL.Rmd

La captura siguiente muestra la pestaña del navegador de archivos (1), donde debe posicionarse a la carpeta Home → OML4R (2), que es la que contiene los notebooks que se acaban de enumerar.

The screenshot shows the RStudio interface running in a Mozilla Firefox browser window. The terminal pane displays the standard Oracle R distribution startup message. The file browser pane on the right shows a directory structure under 'Home > OML4R'. A red arrow labeled '1' points to the 'Files' tab in the browser header. Another red arrow labeled '2' points to the 'OML4R' folder in the file list. The file list contains the following files:

| Name | Size | Modified |
|-----------------------|--------|----------|
| ORE.first.Rmd | 6.1 KB | May 3, |
| ORE.fourth.Rmd | 3.9 KB | May 3, |
| ORE.last.Rmd | 3.6 KB | Mar 16, |
| ORE.second.Rmd | 4.1 KB | May 3, |
| ORE.SQL.Rmd | 4.8 KB | May 3, |
| ORE.third.Rmd | 3.4 KB | May 3, |
| Short_Demo_Script.sql | 1.3 KB | Feb 1, : |

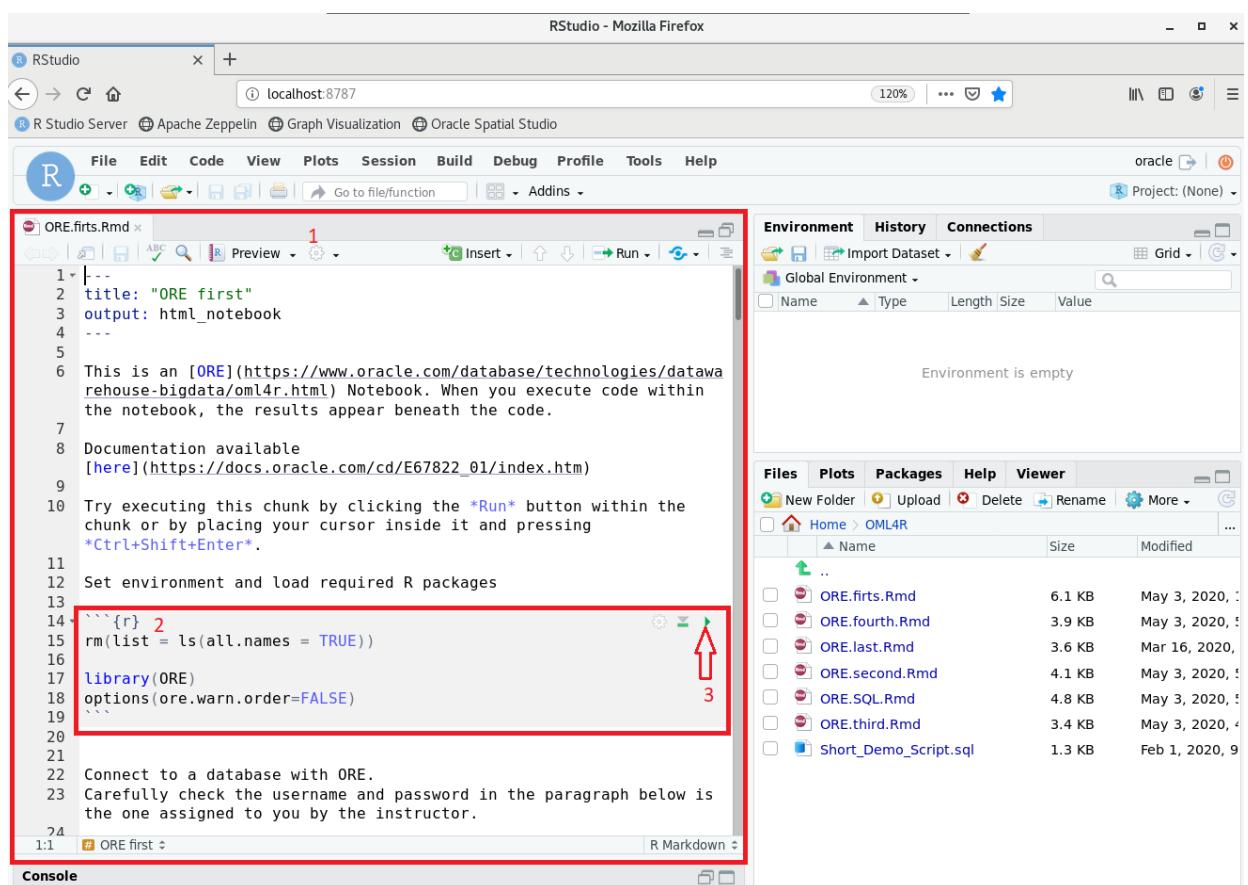


Al hacer click sobre cada notebook, se abren en el editor de la interfaz de usuario y están listos para comenzar su ejecución.

Cada notebook está compuesto por bloques de texto y de código R. Los de código R tienen fondo gris y un botón verde en la zona superior derecha con el que se ejecuta. Los resultados de su ejecución aparecen justo debajo, para su inspección. En algunos casos los resultados serán texto, en otros casos serán visualizaciones y cuando sean una combinación de ambos aparecerán en varias pestañas.

La siguiente captura muestra el primer notebook `ORE.first.Rmd` abierto en una pestaña de trabajo (1), con un primer bloque de texto y una segunda con el primer bloque de código R (2) donde se ha señalado el botón verde de ejecución (3).

Los notebooks se pueden editar, de manera que usted puede realizar modificaciones sobre el código original y ver los nuevos resultados.



Cuando se pulsa el botón de ejecución aparece en la izquierda del párrafo una barra verde de progreso que indica qué línea está ejecutándose, tal y como se puede ver en esta captura:



```

13 ````{r}
14 rm(list = ls(all.names = TRUE))
15
16 library(ORE)
17 options(ore.warn.order=FALSE)
18 ````
```

Observe que las líneas de todos los párrafos están numeradas, pero las áreas donde se presentan los resultados de las ejecuciones no lo están, como se puede ver a continuación.

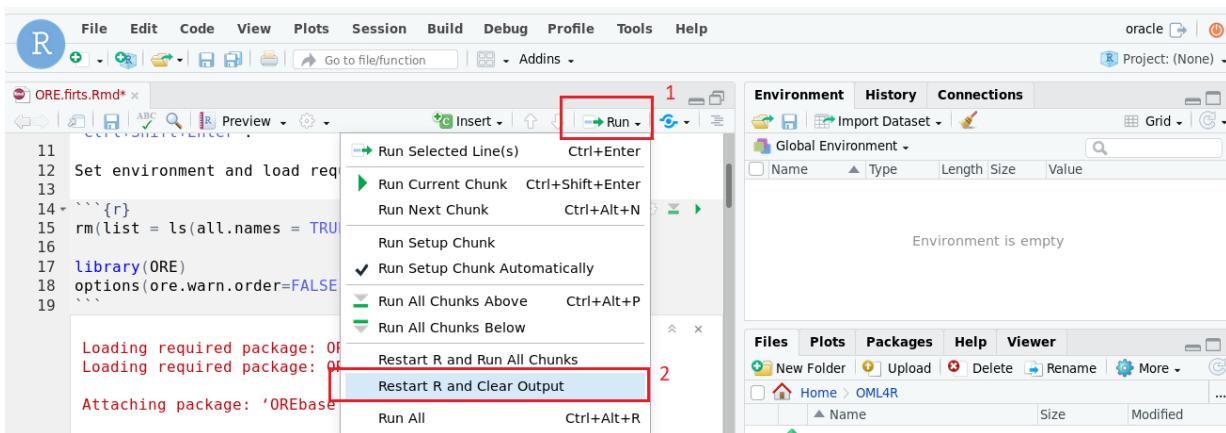
```

-- ````{r}
14 rm(list = ls(all.names = TRUE))
15
16 library(ORE)
17 options(ore.warn.order=FALSE)
18 ````
```

Loading required package: OREbase
 Loading required package: OREcommon
 Attaching package: 'OREbase'
 The following objects are masked from 'package:base':
 cbind, data.frame, eval, interaction, order, paste,
 pmax, pmin, rbind, table

Es importante ejecutar los párrafos en orden, ya que cada párrafo tiene dependencia de los anteriores.

Si en algún momento necesita comenzar la ejecución desde el principio, desde el menú *Run* (1) del notebook ejecute *Restart R and Clear Output*(2).



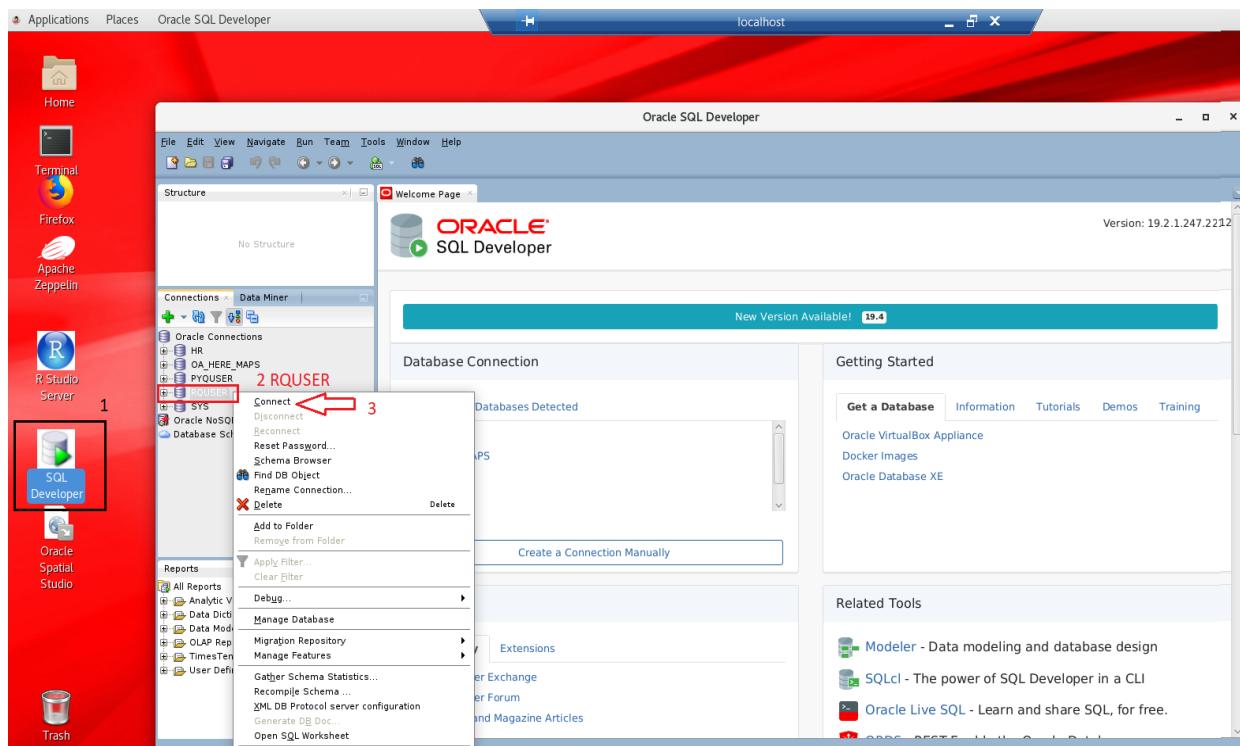
Además del código R proporcionado, en la carpeta OML4R hay un script con extensión sql: *Short_Demo_Script.sql* que contiene sentencias SQL que deben ejecutarse fuera de



RStudio. También durante la ejecución de alguno de los notebooks, se le pedirá que ejecute algunas sentencias SQL.

Es muy importante que estas sentencias se ejecuten sobre el esquema RQUSER, use SQL*Developer para hacerlo.

En el escritorio hay un acceso directo a *SQL*Developer*(1), en la lista de conexiones haga botón derecho sobre RQUSER(2) y seleccione *Connect* (3).



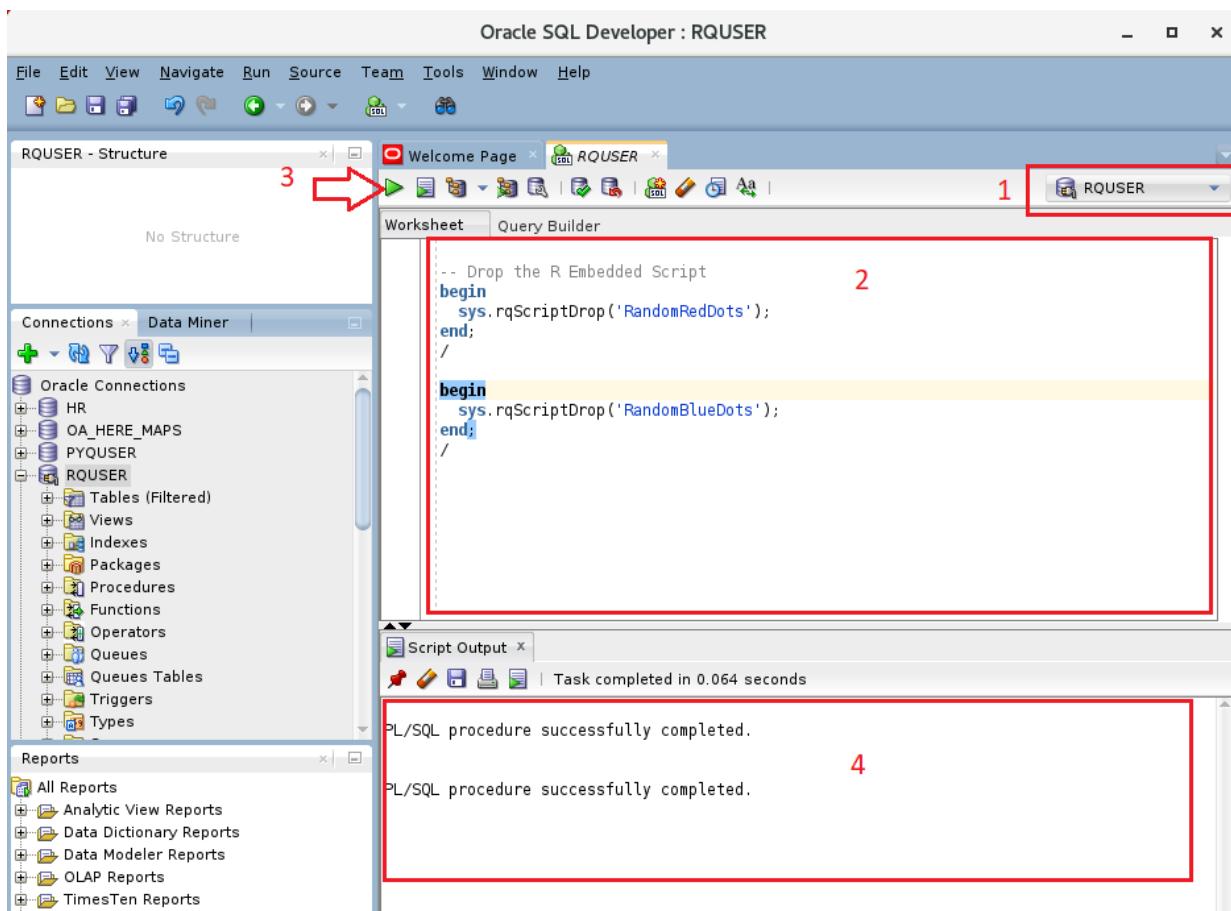
Una vez conectado como RQUSER (1), en el lado derecho de la pantalla de SQL*Developer tendrá una hoja de trabajo SQL o *Worksheet* (2), en la que puede ir pegando las diferentes sentencias SQL ó el código PL/SQL que se le proporcione en los notebooks.

En la barra superior está el botón de ejecución (3) con el que se lanza aquella sentencia o bloque de código donde tenga posicionado el cursor dentro de la hoja de trabajo (2).

Los resultados de las ejecuciones aparecen en el panel inferior derecho (4).

Para estas actividades asegúrese de estar conectado como RQUSER.



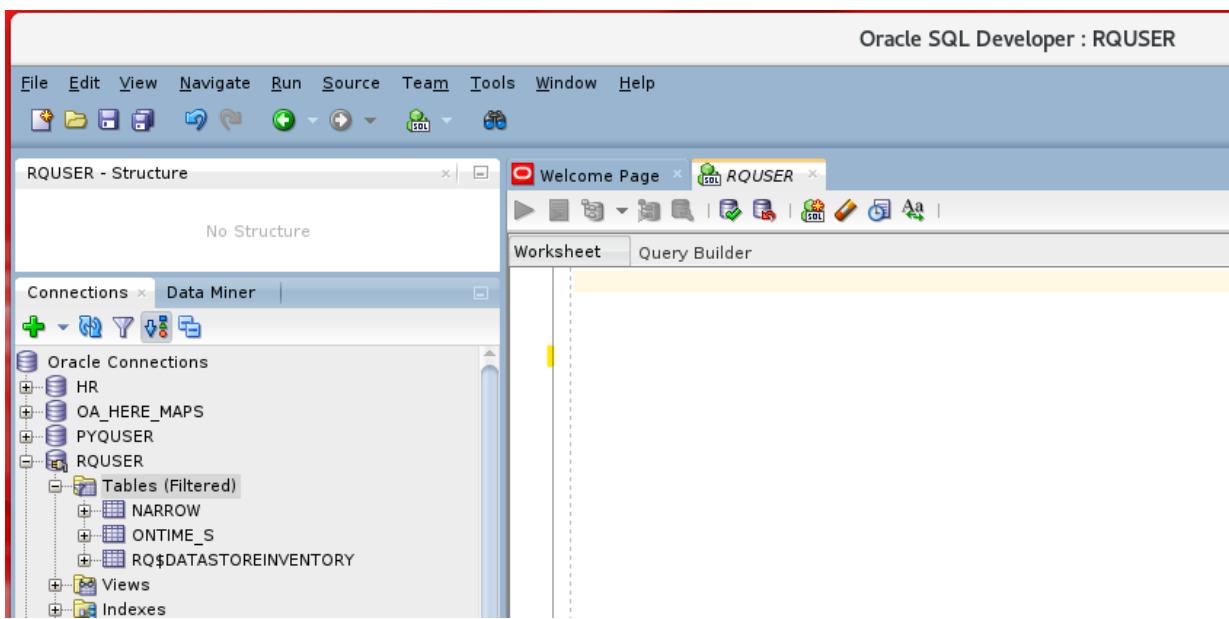


El contenido de cada notebook está escrito de manera que pueda entender cada bloque de código R, aunque no sea un programador experto de R.

No es necesario modificar los notebooks para llevar a cabo el workshop, aunque puede experimentar haciendo cambios.

En esta sesión se va a usar el usuario RQUSER que contiene las siguientes tablas:





Contenido de los notebooks

A continuación, se muestran los contenidos y objetivos de cada uno de los notebooks. Para completar esta sección del workshop sólo debe usar RStudio y SQL*Developer.

[ORE.first.Rmd](#)

Para usar OML4R es necesario cargar el paquete ORE y crear una conexión a la base de datos para trabajar con ella.

El objetivo de este primer script es familiarizarse con esta forma de trabajo donde los juegos de datos están en tablas de la Base de Datos Oracle, pero son manejados como si se tratase de *data frames* en la memoria de la sesión R.

Para este cometido se usa la *capa de transparencia*, donde el desarrollador de R trabaja con los *ORE data frames* que representan objetos e información en la base de datos de la misma manera que si lo hiciese con *data frames* locales.

Los métodos de filtrado y otras tareas analíticas están sobrecargados para que, de manera transparente, la base de datos se encargue del procesamiento sin necesidad de escribir SQL.

[ORE.second.Rmd](#)

En este notebook muestra con varios ejemplos cómo hacer uso de paquetes de CRAN y del propio OML4R (ORE) para construir unos modelos de regresión lineal muy sencillos.



De esta manera se pueden usar las implementaciones de estos algoritmos que ofrece la Base de Datos Oracle desde lenguaje R y también se puede optar por las implementaciones de CRAN.

Los resultados obtenidos por estos dos modos de trabajo se manejan de la misma manera desde R.

[ORE.third.Rmd](#)

En este notebook se introducen dos conceptos:

- Ejecución embebida de código R en la base de datos
- Repositorio de scripts R en la base de datos

A través de algunos ejemplos se realiza la ejecución de funciones escritas en lenguaje R en la base de datos. Se envía el código a la base de datos para su ejecución y se recupera el resultado al entorno local.

También se muestra cómo interactuar con el repositorio de scripts tanto desde RStudio como desde SQL*Developer.

Este repositorio de scripts es por defecto privado al usuario, siendo posible hacer públicas o compartidas con otros usuarios algunas de las funciones que se almacenen en él.

[ORE.foruth.Rmd](#)

En este notebook se profundiza en los diferentes métodos de ejecución embebida en la base de datos:

- indexApply
- groupApply
- tableApply

Cada uno de ellos tiene sentido para diferentes tareas dentro del proceso de trabajo estadístico, de entrenamiento de un modelo o de scoring de nuevos datos.

Con ellos es posible paralelizar algunas de estas etapas.

[Ore.last.Rmd](#)

En este notebook se muestra con más detalle el uso de los repositorios de la base de datos:

- almacén de datos



- repositorio de scripts

Estos dos almacenes permiten guardar en la base de datos objetos de la sesión R y funciones de código, que al estar en tablas de la base de datos pueden ser protegidas por copias de seguridad con los métodos habituales de la base de datos.

Se muestra cómo cambiar la visibilidad de los repositorios para compartir datos o código con otros usuarios.

[Short_Demo_Script.sql](#)

Una funcionalidad muy interesante de OML4R es que permite la ejecución de código R desde una sentencia SQL. En este script se muestran unos ejemplos muy sencillos de cómo hacerlo.

Concretamente se muestran estas dos funcionalidades.

- Ejecución de código R desde SQL.
- Interacción con el repositorio de código R desde SQL

Este script se indica en ORE.third.Rmd que se haga su ejecución, pero puede hacerse por separado también.

Resumen

¡Enhorabuena! Ha conseguido llegar al final de las actividades de esta sección del workshop.

En este workshop usted ha conseguido los siguientes retos:

- Usar OML4R para conectar a una Base de Datos Oracle usando la interfaz habitual de R RStudio Server.
- Experimentar con la capa de transparencia, usando juegos de datos almacenados en la Base de Datos Oracle como si estuvieran en local
- Comparar cómo trabajar con la funcionalidad de OML4R frente a la forma tradicional de R
- Realizar analítica avanzada sin mover los datos fuera de la Base de Datos Oracle.
- Almacenar y ejecutar código R en la Base de Datos Oracle, tanto desde R como desde SQL
- Almacenar objetos de sesión y código en la Base de Datos y cómo compartirlo con otros usuarios de la Base de Datos.



Mas información

La documentación oficial de OML4R (**Oracle R Enterprise**) está disponible públicamente en el siguiente sitio web: https://docs.oracle.com/cd/E67822_01/index.htm

Esta documentación incluye:

- Instrucciones de instalación
- Manual de usuario
- Notas de la versión

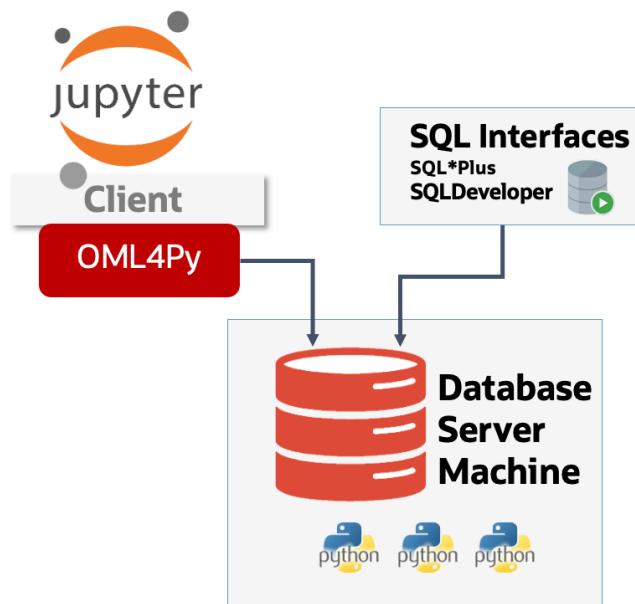
Consulte esta documentación para obtener un mayor conocimiento de ORE.



OML4Py (1 hora)

Todas las actividades de esta sección están preparadas para ser realizadas a través de *Jupyter*, el cual debe ser accedido usando un navegador web.

Este diagrama muestra las diferentes partes que estarán en uso durante las actividades de esta sección:

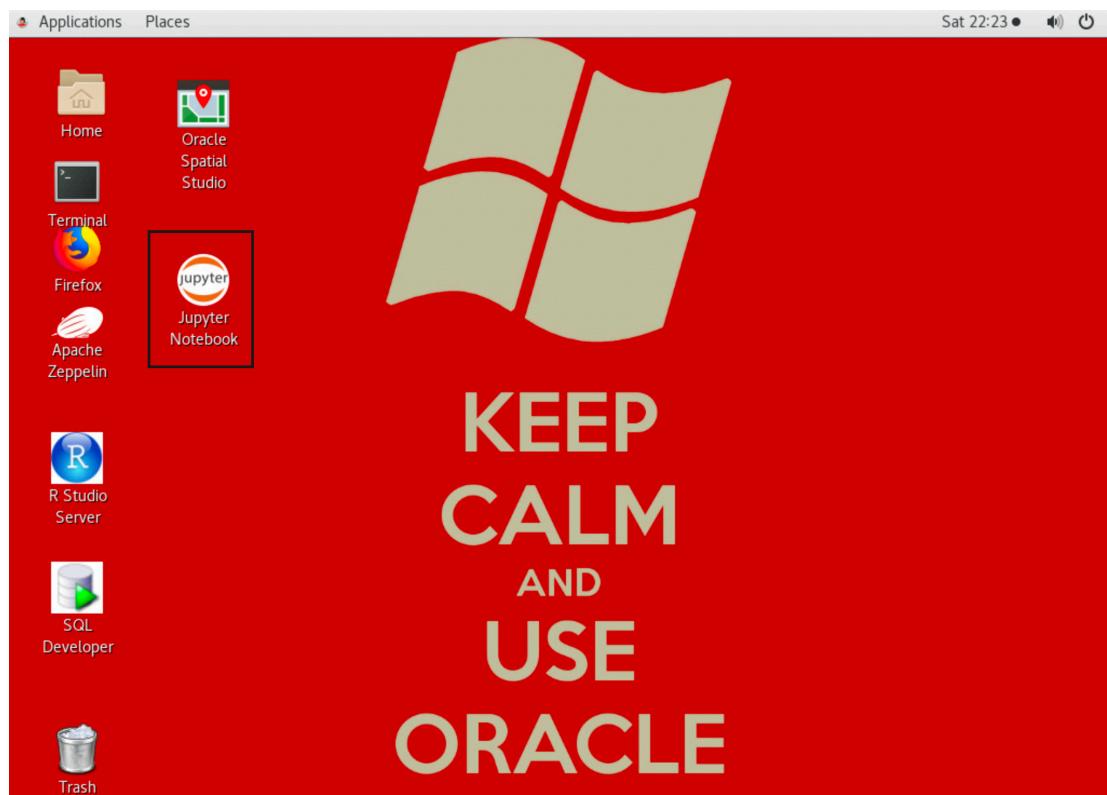


Acceso a Jupyter

Una vez se haya accedido al servidor con el cliente de *Remote Desktop*, en el escritorio hay un acceso directo a Jupyter.

Haga doble click para abrir un navegador que directamente abrirá la dirección de Jupyter.





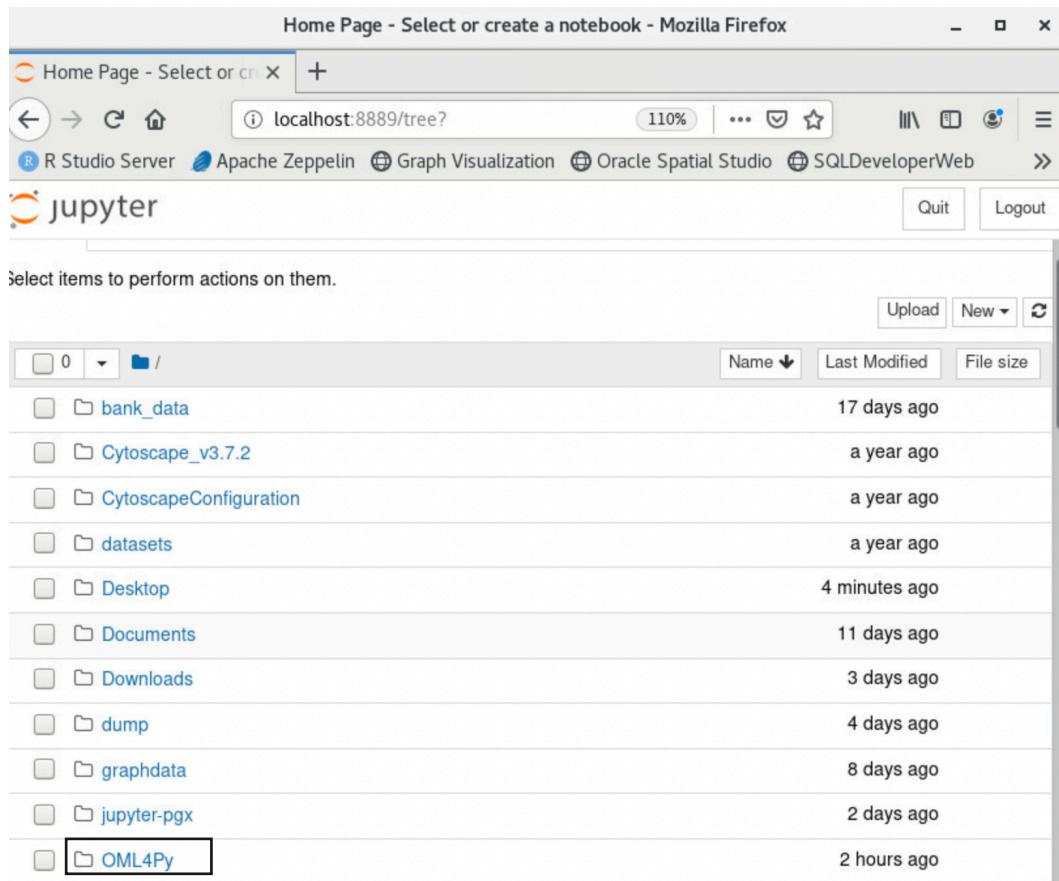
La página de acceso se mostrará en pantalla:

A screenshot of a Mozilla Firefox browser window. The title bar says "Jupyter Notebook - Mozilla Firefox". The address bar shows the URL "localhost:8889/login". Below the address bar, there is a toolbar with various icons. The main content area of the browser shows a login form for Jupyter Notebook. It features a "jupyter" logo at the top, followed by a "Password:" label and an input field with a single character typed into it. Below the input field is a "Log in" button.



Una vez introducida la credencial de acceso, aparecerá en la pantalla un navegador de ficheros que muestra el contenido del directorio casa del usuario Oracle.

Use este navegador para localizar la carpeta *OML4Py* y acceder a ella, puesto que allí están los notebooks a usar durante el workshop.



A continuación, se muestran los 5 notebooks que se usarán durante esta sesión dedicada a OML4Py. También existe un sexto fichero con código SQL, que puede abrirse con Jupyter para leer su contenido, pero no para ejecutarlo.



| Name | Last Modified | File size |
|--|---------------|-----------|
| .. | seconds ago | |
| img | a year ago | |
| zeppelin | a day ago | |
| 1 - OML4Py Introduction.ipynb | a day ago | 15.7 kB |
| 2 - OML4Py Data Selection and Manipulation.ipynb | a day ago | 29.3 kB |
| 3 - OML4Py Datastore and Script Repository.ipynb | 2 hours ago | 20.9 kB |
| 4 - OML4Py Embedded Python Execution.ipynb | 2 hours ago | 25.2 kB |
| 5 - OML4Py Automatic Machine Learning.ipynb | 2 hours ago | 18.2 kB |
| OML4Py_demo.sql | 2 years ago | 7 kB |

Al hacer click sobre cada uno de los notebooks, se abre en una nueva pestaña del navegador el editor de la interfaz de usuario y están listos para comenzar su ejecución.

Cada notebook está compuesto por dos tipos de párrafos que contienen

- bloques de texto en formato *markdown*
- código python

Ambos disponen de un botón *Run* en la barra superior con el que se ejecutan. Al ejecutarse los resultados aparecen justo debajo para su inspección.

Nota: No es necesario ejecutar los párrafos de markdown.

La siguiente captura muestra el primer notebook: 1 - OML4Py Introduction abierto en el navegador. Tras los primeros párrafos de markdown está el primer párrafo de Python, que empieza tiene a la izquierda el texto IN [] y números de línea, así como el código Python resaltado en diferentes colores.



The screenshot shows a Jupyter Notebook interface in a Mozilla Firefox browser window. The title bar reads "1 - OML4Py Introduction - Jupyter Notebook - Mozilla Firefox". The address bar shows the URL "localhost:8889/notebooks/OML4Py/1 - OML4Py Introduction.ipynb". The notebook content includes a list of steps and a code cell for importing the OML4Py library.

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 Logout

• create database tables
• use the transparency layer
• rank attributes for predictive value using the in-database attribute importance algorithm
• build a predictive model
• score data using this model

Import the OML4Py library and connect to Oracle Database

To use OML4Py, first import the package `oml`. OML4Py supports a variety of connection specification options, including Oracle Wallet. Once connected to an Oracle Database that has OML4Py installed, invoking `oml.isconnected` returns true.

```
In [ ]: 1 import warnings  
2 warnings.filterwarnings('ignore')  
3  
4 import oml  
5 oml.connect("pyquser","pyquser",  
6 dsn='(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=localhost)(PORT=1521))(CONNECT  
7 oml.isconnected()
```

Create a Pandas DataFrame and load into Oracle

El código de cada párrafo de los notebooks se puede editar, de manera que usted puede realizar modificaciones sobre el código original y ver los nuevos resultados.

No es necesario realizar ninguna modificación para completar el taller.

Cuando se pulsa el botón de ejecución de un párrafo Python, el texto a su izquierda cambia a IN [#] con el orden de ejecución y justo debajo puede aparecer la salida, como se muestra a continuación:

The screenshot shows a Jupyter Notebook cell with the input code and its resulting output. The input cell is labeled "In [1]:" and contains the same code as the previous screenshot. The output cell, labeled "Out[1]:", displays the result "True".

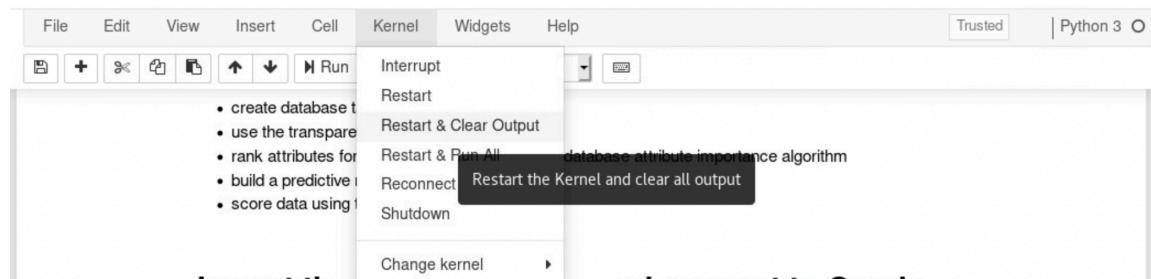
```
In [1]: 1 import warnings  
2 warnings.filterwarnings('ignore')  
3  
4 import oml  
5 oml.connect("pyquser","pyquser",  
6 dsn='(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=localhost)(PORT=1521))(CONNECT  
7 oml.isconnected()
```

Out[1]: True

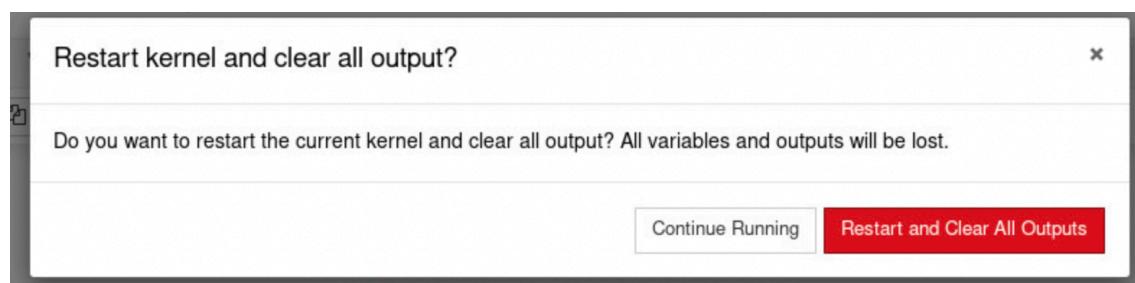


En caso de que la salida sean datos se muestran en forma tabular o con algún modo gráfico de visualización.

Si encuentra problemas con la ejecución, puede ser necesario reiniciar el kernel de Python. Cuando se hace esto, todas las variables y resultados de la sesión se pierden, por lo que es necesario repetir la ejecución completa del notebook puesto que, en general, cada párrafo depende de los resultados de los anteriores.

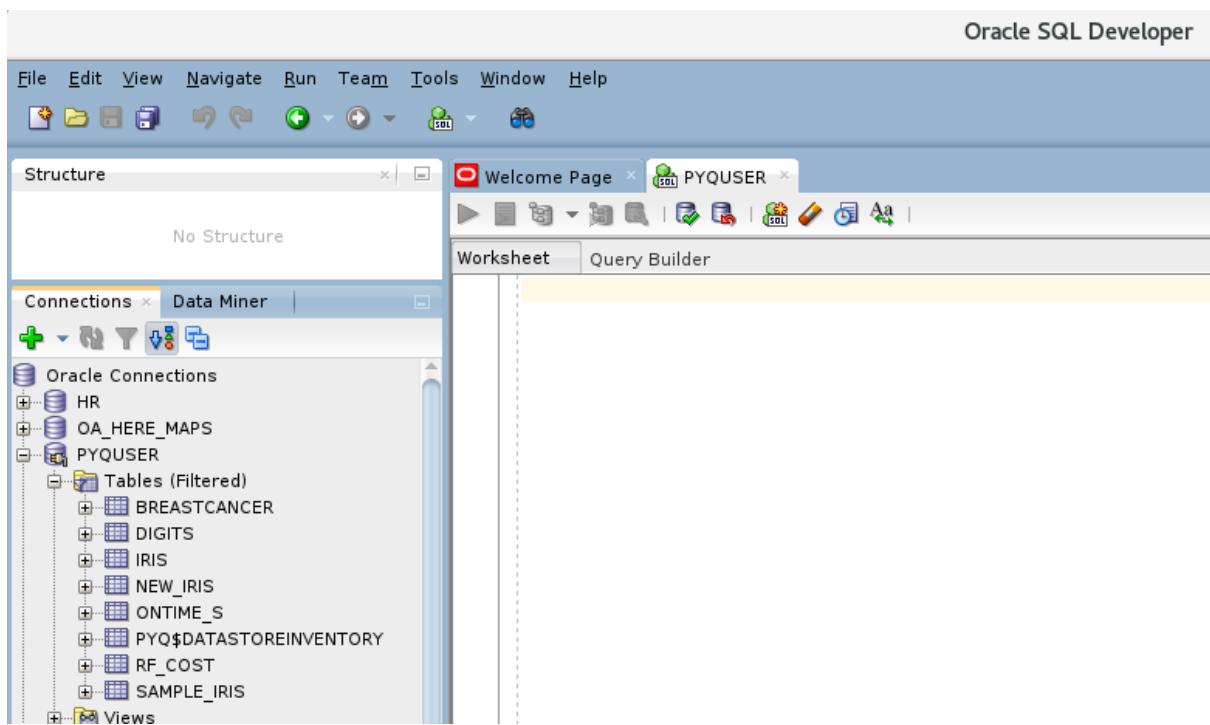


Para reiniciar el kernel de Python haga click en el botón ‘Restart and Clear All Outputs’:



En este apartado se va a usar el esquema de base de datos PYQUSER que contiene varias tablas precargadas con los datos necesarios para las diferentes actividades:





Estas tablas contienen los dataset con los que se trabaja desde OML4Py.

Contenido de los notebooks

A continuación, se muestran los contenidos en cada uno de los notebooks, así como el objetivo de estos.

1 - OML4Py Introduction

Con OML4Py se consigue la integración del lenguaje Python con la Base de datos Oracle construyendo un entorno para uso empresarial de toda la parte estadística, *machine learning* y análisis gráfico de información almacenada en tablas y vistas.

El objetivo de este notebook es mostrar los siguientes aspectos de su forma de trabajar:

- Conectar a Base de Datos Oracle
- Crear tablas en Oracle
- Usar la capa de transparencia
- Usar el algoritmo de importancia de atributos
- Construir un modelo predictivo
- Usar el modelo predictivo



Se ofrece un entorno de trabajo nativo al programador Python, de manera está trabajando con datos desde y hacia tablas de la Base de Datos Oracle sin necesidad de tener conocimientos SQL para la creación de objetos o para la consulta avanzada. Todo se realiza con funciones Python que se traducen al motor de la base de datos de forma transparente al programador.

2 OML4Py Data Selection and Manipulation

En este notebook se muestra cómo trabajar con información sin moverla de la base de datos en tareas de selección y manipulación de datos.

De esta manera se delegan en la base de datos que es capaz de manejar volúmenes de datos masivos que no puede almacenarse en la memoria RAM.

Se consigue evitar extracciones de datos, pérdida de gobierno sobre esas extracciones de datos, trabajar con juegos de datos anticuados ... son varias las ventajas.

Para la selección y manipulación se usan los mismos mecanismos y funciones que un programador Python usaría para trabajar con un juego de datos cargado en local en la memoria de su puesto de trabajo.

Esto evita tener que aprender a trabajar con un paquete nuevo, con un escalón de entrada muy bajo ya que los conocimientos existentes son perfectamente útiles con OML4Py. Lo que se hace es sobrecargar los métodos habituales y que sea la base de datos la que se encargue de hacer el trabajo.

3 - OML4Py Datastore and script repository

El objetivo de este notebook es mostrar cómo usar la Base de Datos Oracle como repositorio de objetos de la sesión Python (*datastore*), así como de funciones de código (*script repository*) para ser invocadas directamente en la base de datos.

Estos almacenes de datos y código son propios de cada usuario de base de datos, aunque pueden compartirse con otros usuarios o convertirse en públicos.

Con el *datastore*, los usuarios pueden:

- Guardar objetos de una sesión para recuperarlos en otra
- Pasar argumentos a las funciones Python o guardar resultados de las mismas

Con el *script repository*, los usuarios pueden:

- Almacenar el código de sus funciones en la base de datos
- Compartir esas funciones con otros usuarios de la base de datos
- Traer esas funciones a la sesión Python o ejecutarlas desde SQL

Aunque en el notebook se muestra esta funcionalidad desde Python, existe una interfaz equivalente en SQL para trabajar con el repositorio de scripts.



4 - OML4Py Embedded Python Execution

En este notebook se experimenta con la ejecución de Python en la Base de Datos Oracle. Bien a pasando el código de una función de la sesión a la base de datos para su ejecución, bien invocando una de las funciones almacenadas en el *script repository*. Como datos de entrada a estas ejecuciones se usará un juego de datos que se encuentre dentro de la base de datos. Con este modo de trabajo se consigue llevar la ejecución a los datos y no los datos a la ejecución.

En el notebook se muestran diferentes estrategias de ejecución embebida en función del problema que se esté resolviendo.

5 - OML4Py Automatic Machine Learning

Este notebook ofrece una visión general de la funcionalidad que ofrece AutoML.

Se muestran los siguientes aspectos:

- Conexión a Base de datos Oracle. Los datos de trabajo están en tablas, para poder acceder a ellos son necesarias unas credenciales.
- Generación de objetos proxy.
- Empleo de AutoML
 - selección automática de algoritmo
 - selección automática de atributos
 - selección automática de hiperparámetros

Cada uno de estos conceptos está explicado en mayor detalle en los bloques de *markdown* del notebook.

OML4Py_demo.sql

Una funcionalidad muy interesante de OML4Py es que permite la ejecución de código Python desde una sentencia SQL. En este script se muestran unos ejemplos muy sencillos de cómo hacerlo.

Concretamente se muestran estas dos funcionalidades.

- Ejecución de código Python desde SQL.
- Interacción con el repositorio de código Python desde SQL

Para ejecutar el código de este script, editelo desde Jupyter y copie su contenido a una hoja de trabajo SQL en SQL Developer después de abrir una sesión con el usuario PYQUSER que se encuentra preconfigurado.



The screenshot shows two windows side-by-side. On the left is a Jupyter Text Editor window titled 'OML4Py_demo.sql - Jupyter Text Editor'. It contains Python code for generating scatter plots. On the right is an 'Oracle SQL Developer : PYQUSER' window. The 'Connections' tab shows a connection to 'PYQUSER' highlighted with a red box. The 'Worksheet' tab displays the same Python code as the Jupyter editor. A large red arrow points from the Jupyter editor towards the Oracle SQL Developer window.

```

7 --#
8 --#####
9
10 --## Random Red Dots
11 BEGIN
12     sys.pyqScriptCreate('pyqFun1',
13     'def pyqFun1 ()':
14         import numpy as np
15         import pandas as pd
16         import matplotlib.pyplot as plt
17
18         d = {'id': range(1,10), 'val': [x/100 for
19             df = pd.DataFrame(data=d)
20             fig = plt.figure(1)
21             ax = fig.add_subplot(111)
22             ax.scatter(range(0,100), np.random.rand(100),c='r')
23             fig.suptitle("Random Red Dots")
24
25             fig2 = plt.figure(2)
26             ax2 = fig2.add_subplot(111)
27             ax2.scatter(range(0,10), np.random.rand(10),c='r')
28             fig2.suptitle("Random Red Dots")
29             return df', NULL, TRUE);
30 END;
31 /
32
33 SELECT * FROM table(pyqEval(NULL,'PNG','pyqFun1'))
34 SELECT * FROM table(pyqEval(NULL,'select 1 id, 1 \'

```

```

BEGIN
    sys.pyqScriptCreate('pyqFun1',
    'def pyqFun1 ()':
        import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt

        d = {'id': range(1,10), 'val': [x/100 for
            df = pd.DataFrame(data=d)
            fig = plt.figure(1)
            ax = fig.add_subplot(111)
            ax.scatter(range(0,100), np.random.rand(100),c='r')
            fig.suptitle("Random Red Dots")

            fig2 = plt.figure(2)
            ax2 = fig2.add_subplot(111)
            ax2.scatter(range(0,10), np.random.rand(10),c='r')
            fig2.suptitle("Random Red Dots")
            return df', NULL, TRUE);
END;
/

```

Resumen

¡Enhorabuena! Ha conseguido llegar al final de las actividades de esta sección del workshop.

En este workshop usted ha conseguido los siguientes retos:

- Usar OML4Py para conectar a una Base de Datos Oracle usando la interfaz Jupyter.
- Experimentar con la capa de transparencia, usando juegos de datos almacenados en la Base de Datos Oracle como si estuvieran en local.
- Comparar cómo trabajar con la funcionalidad de OML4Py frente a la forma tradicional de Python.
- Realizar analítica avanzada sin mover los datos fuera de la Base de Datos Oracle.
- Almacenar y ejecutar código Python en la Base de Datos Oracle, tanto desde Python como desde SQL.
- Almacenar objetos de sesión y código Python en la Base de Datos y cómo compartirlo con otros usuarios de la Base de Datos.

