

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

## HOL 2 Machine Learning



# Contenidos

<b>WORKSHOP SPATIAL, GRAPH Y MACHINE LEARNING EN BASE DE DATOS ORACLE .....</b>	<b>1</b>
<b>HOL 2 MACHINE LEARNING .....</b>	<b>1</b>
<b>REQUERIMIENTOS INICIALES .....</b>	<b>3</b>
ESCRITORIO REMOTO CON MICROSOFT WINDOWS .....	3
ESCRITORIO REMOTO CON MacOS .....	4
<b>MACHINE LEARNING.....</b>	<b>7</b>
OML4R (1 HORA) .....	7
Acceso a RStudio .....	7
Contenido de los notebooks .....	14
RESUMEN.....	16
MAS INFORMACIÓN .....	17
OML4PY (1 HORA) .....	18
Acceso a Zeppelin.....	18
Contenido de los notebooks .....	24
RESUMEN.....	26



# 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.

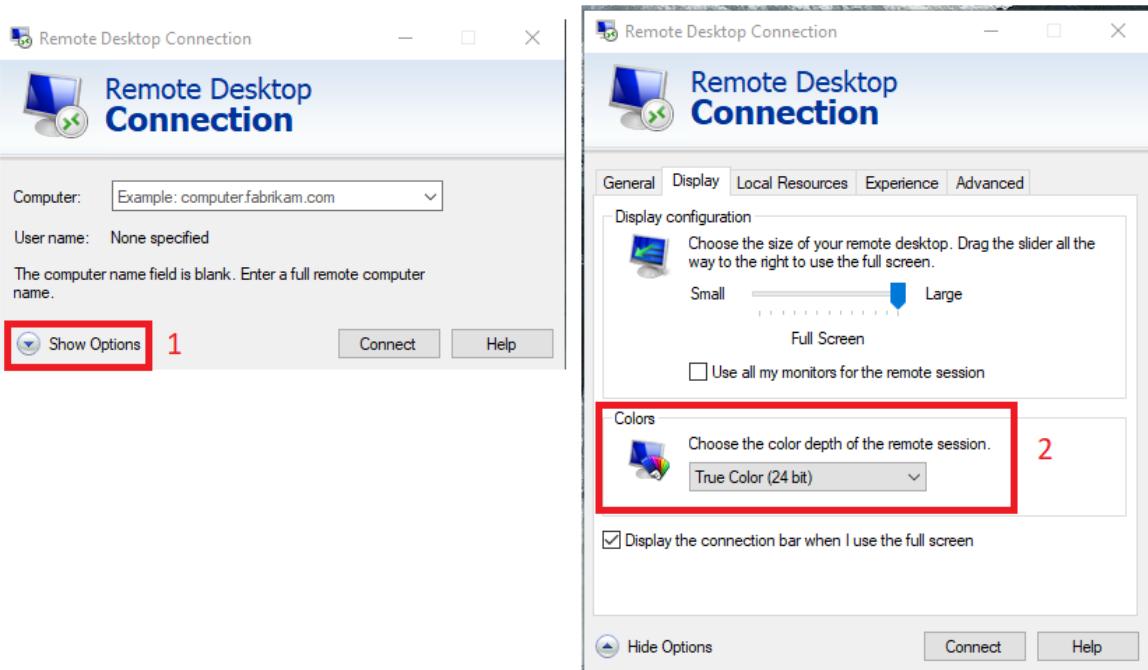
La máquina del workshop se proporciona para uso individual de cada participante del workshop siendo para uso exclusivo del mismo.

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.

## Escrítorio 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:





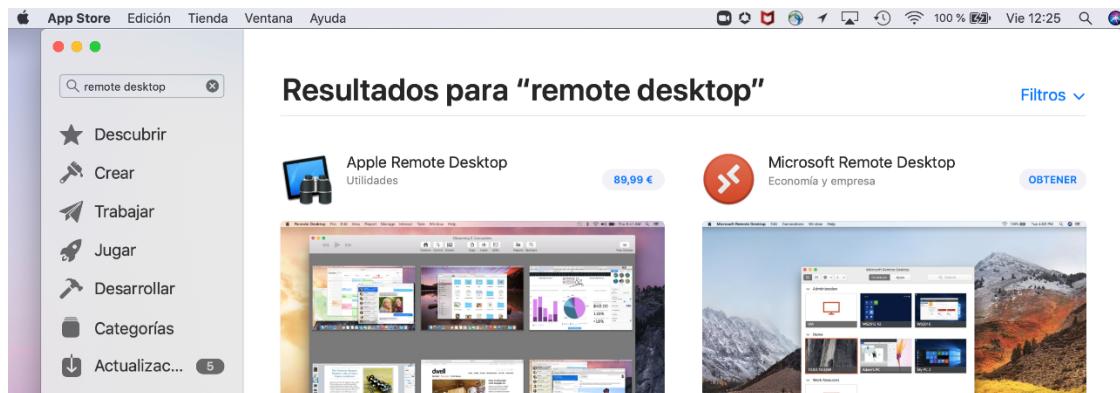
Como parte de la documentación del workshop se facilitarán los siguientes datos:

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

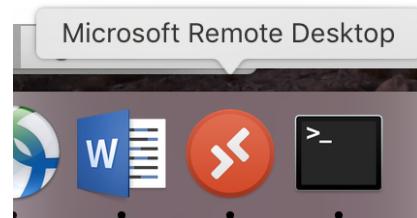
## Escritorio Remoto con MacOS

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:



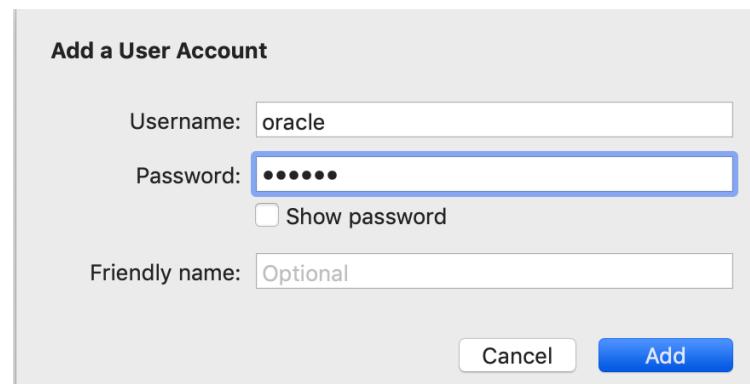
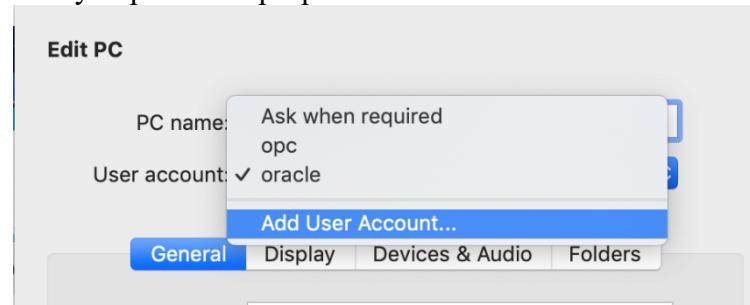
Una vez instalada esta aplicación, aparecerá un ícono como el siguiente en la barra de aplicaciones para poder realizar las conexiones.



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

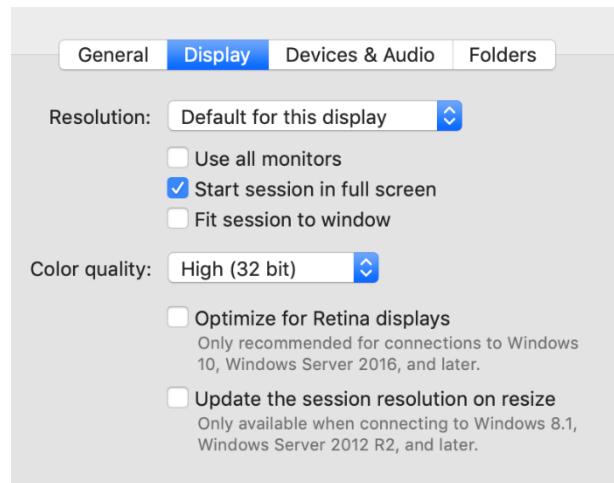
- Nombre de usuario y password
- IP pública de la maquina del workshop

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

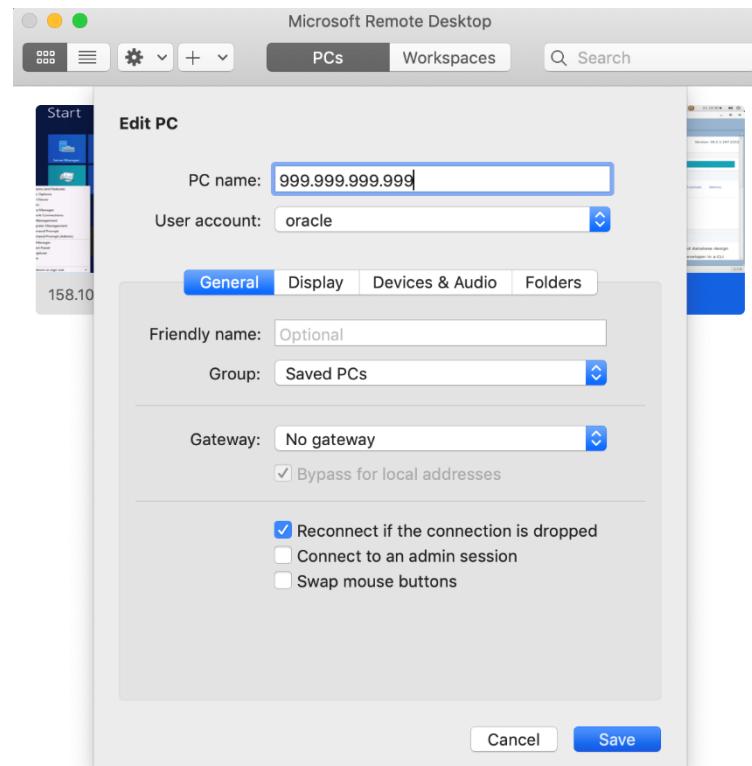


En la pestaña de Display se confirma que la calidad de color está seleccionada a 32 bits:





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:

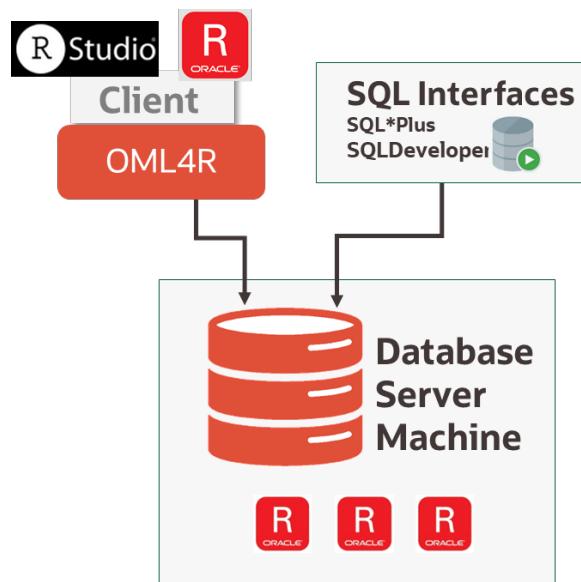
- **OML4Py** - Python (<https://www.python.org/>)
- **OML4R** - R (<https://cran.r-project.org/>)

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

## 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:

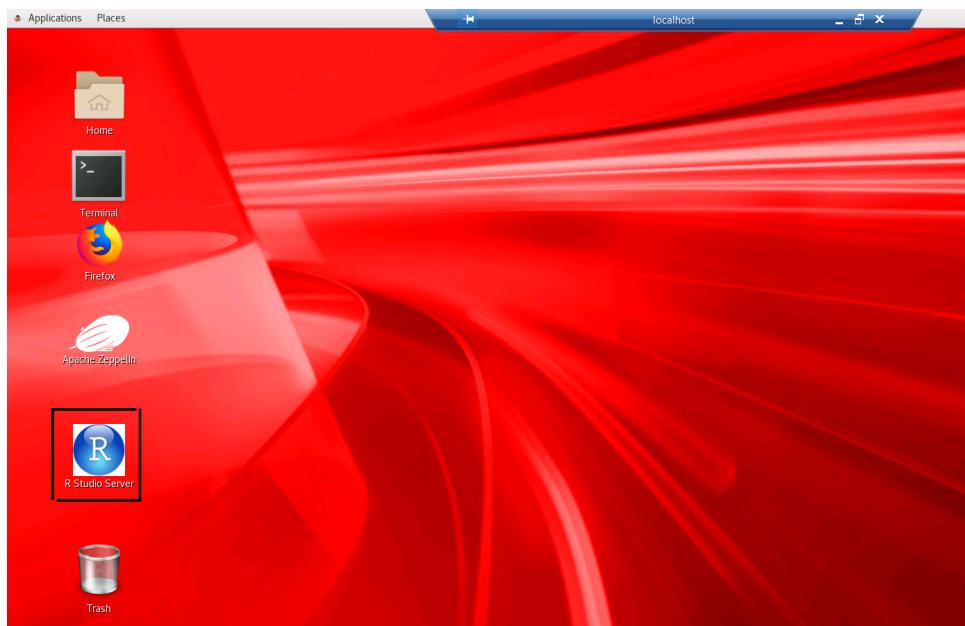


### Acceso a RStudio

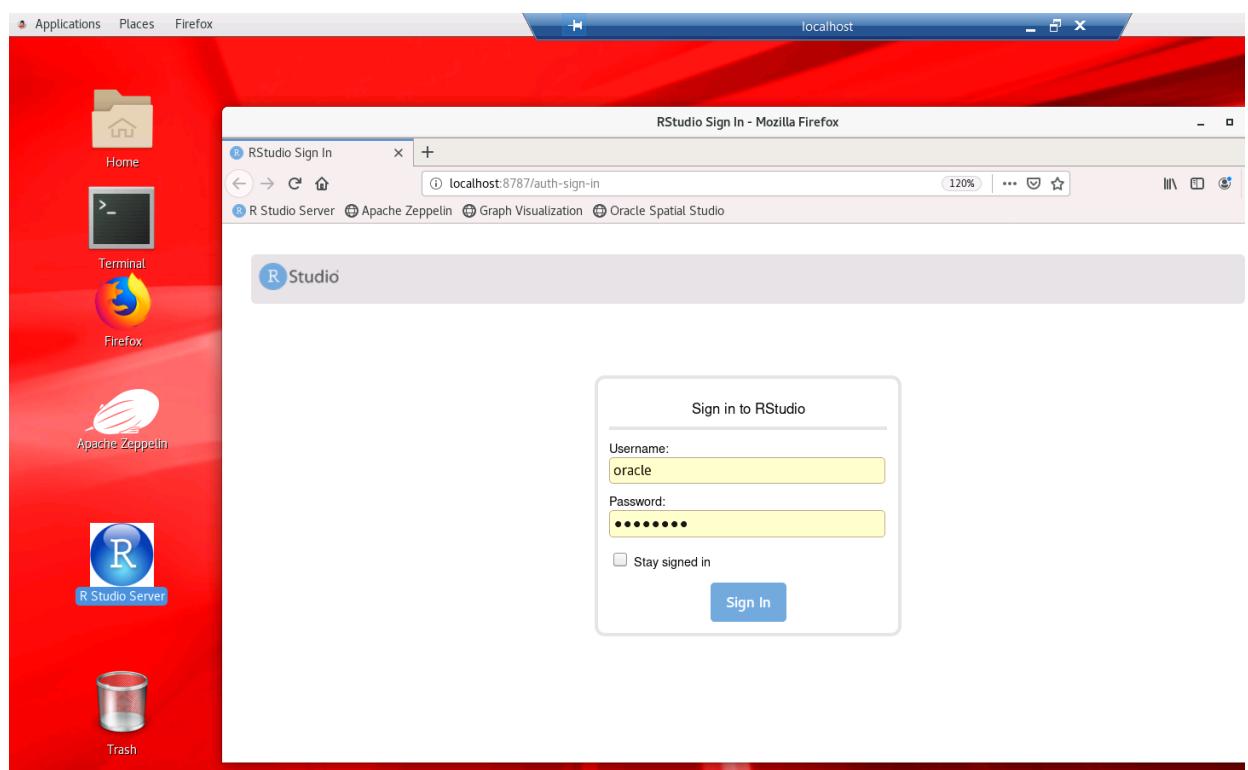
Siga las instrucciones proporcionadas para acceder al escritorio remoto. Se le proporcionará una dirección IP y un usuario/clave.

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*.



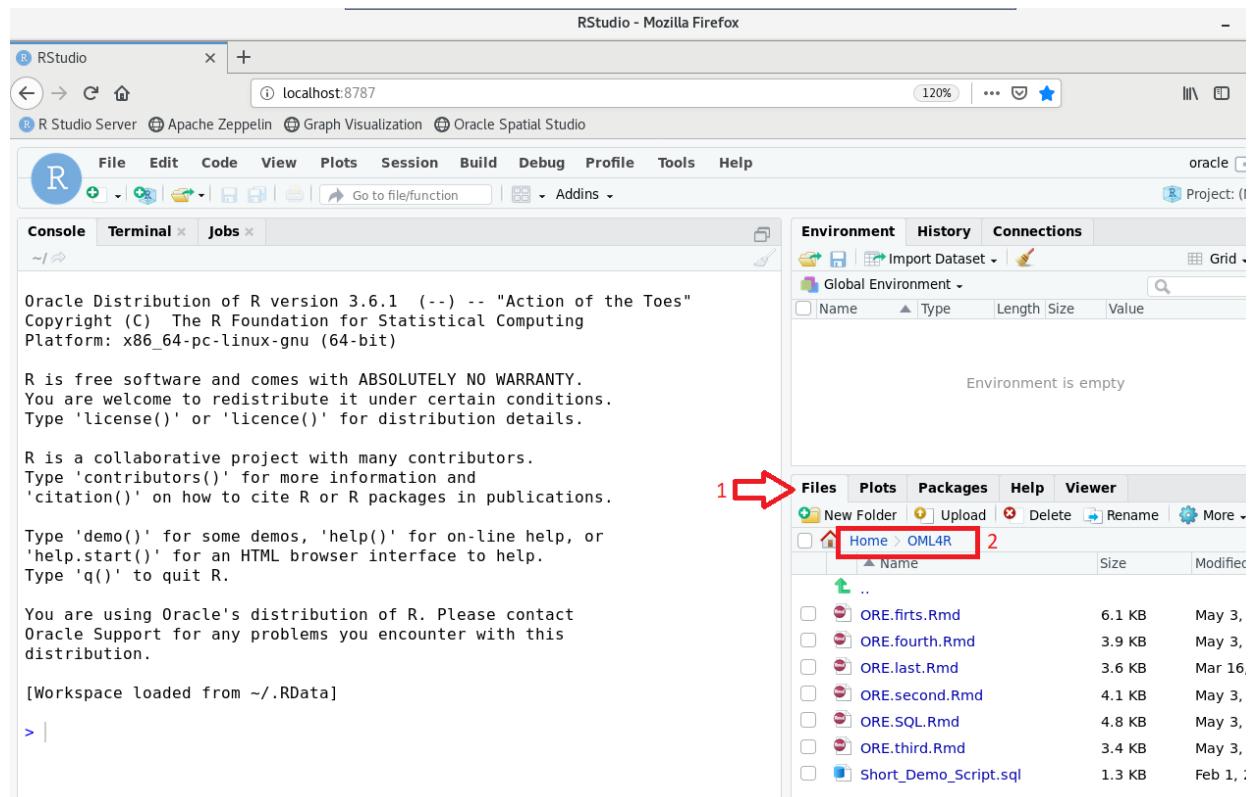
Una vez introducidas las credenciales que se le proporcionen para el curso, aparecerá la interfaz de usuario con varias áreas de trabajo y una sesión de R lista para comenzar las actividades.



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.



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

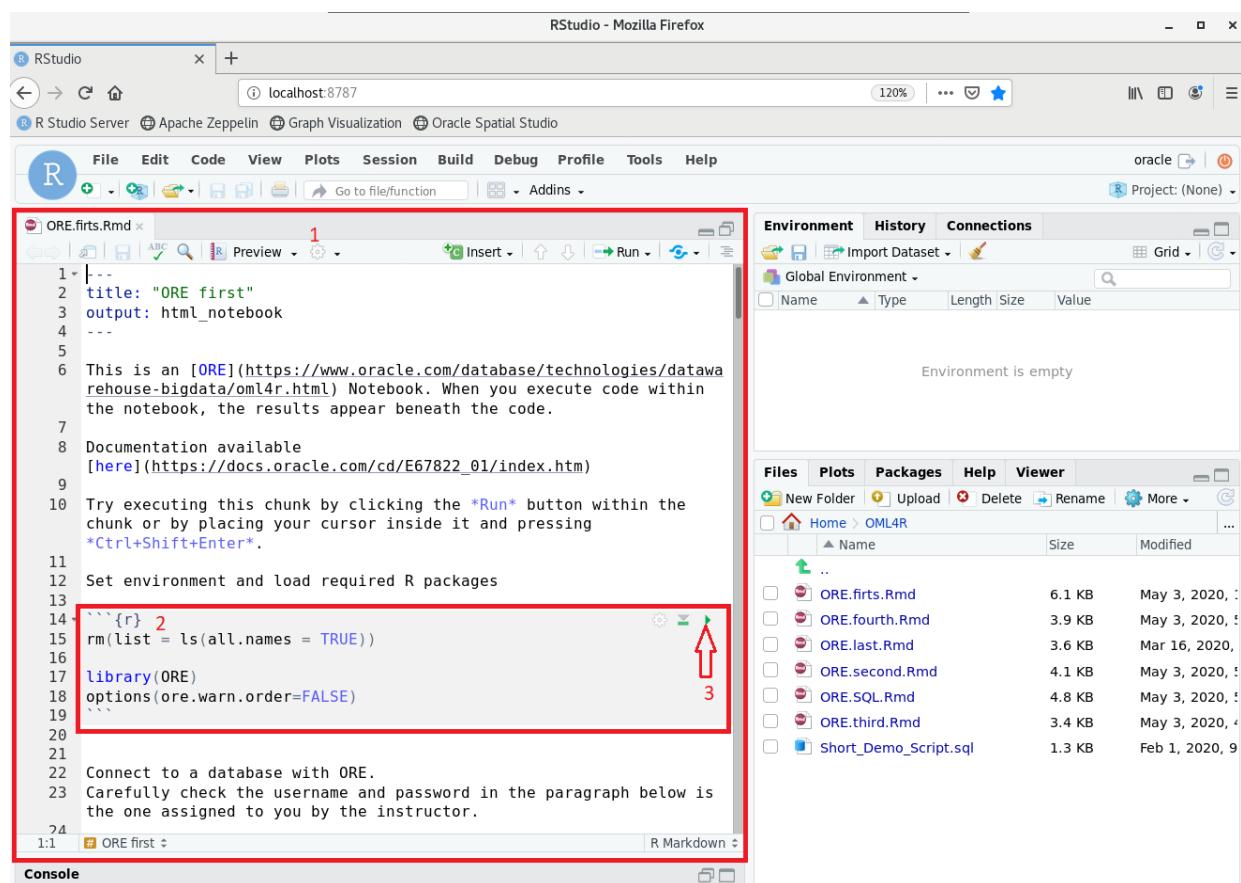
Cada notebook está compuesto por bloques de texto y de código R. Los de código R tienen fondo gris y un botón verde en la zona superior derecha con el que se ejecuta. Los resultados de su ejecución aparecen justo debajo, para su inspección. En algunos casos los



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.

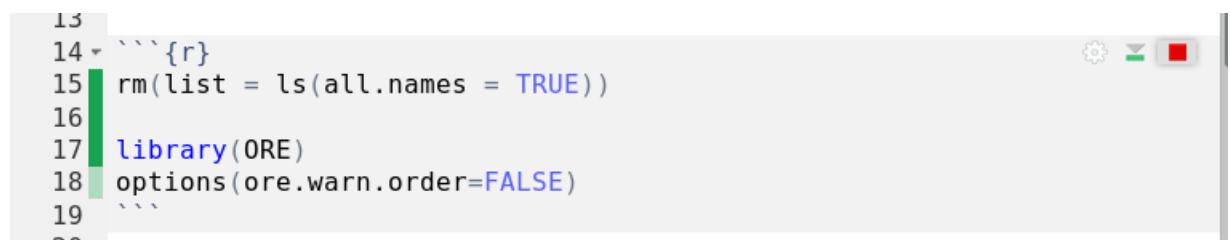


The screenshot shows the RStudio interface with the following details:

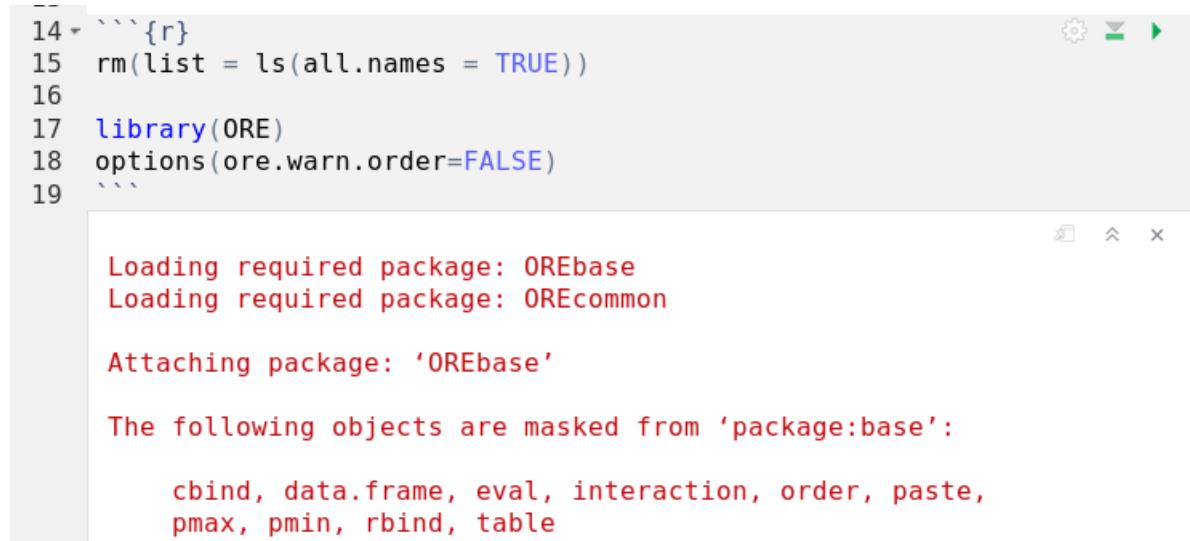
- Title Bar:** RStudio - Mozilla Firefox
- Toolbar:** File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, Help
- Code Editor:** Shows the `ORE.first.Rmd` file content. Line 15, which contains the code `rm(list = ls(all.names = TRUE))`, is highlighted with a red box and has a green progress bar above it, indicating it is currently executing. A red arrow points to this green bar.
- Environment Tab:** Shows the Global Environment table, which is currently empty.
- Files Tab:** Shows a list of files in the current directory:

Name	Type	Length	Size	Value
ORE.firts.Rmd			6.1 KB	May 3, 2020, :
ORE.fourth.Rmd			3.9 KB	May 3, 2020, :
ORE.last.Rmd			3.6 KB	Mar 16, 2020,
ORE.second.Rmd			4.1 KB	May 3, 2020, :
ORE.SQL.Rmd			4.8 KB	May 3, 2020, :
ORE.third.Rmd			3.4 KB	May 3, 2020, :
Short_Demo_Script.sql			1.3 KB	Feb 1, 2020, 9

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:



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.



The screenshot shows an RStudio interface with the following code in the script pane:

```
14 - ````{r}
15 rm(list = ls(all.names = TRUE))
16
17 library(ORE)
18 options(ore.warn.order=FALSE)
19 ````
```

The output pane displays the results of the execution:

```
Loading required package: OREbase
Loading required package: OREcommon

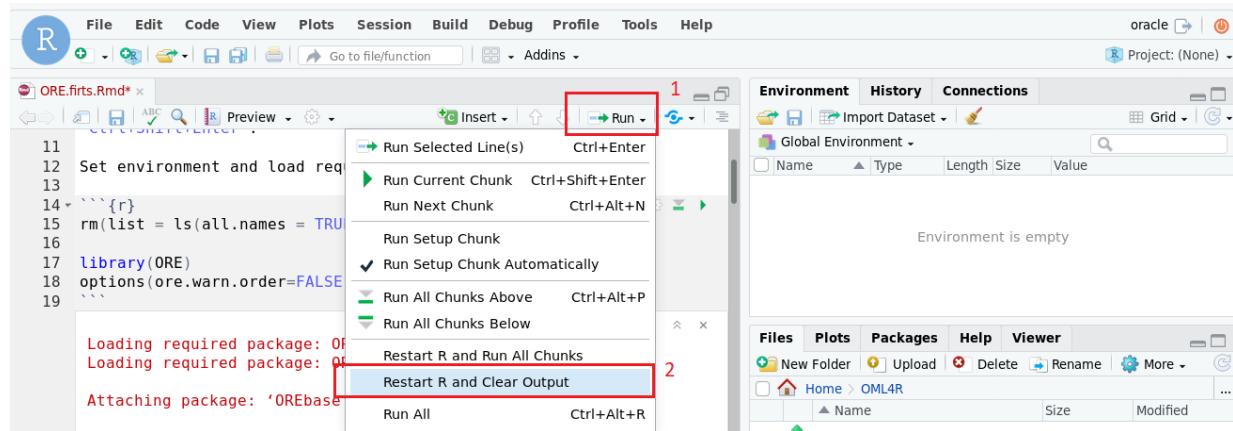
Attaching package: 'OREbase'

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).

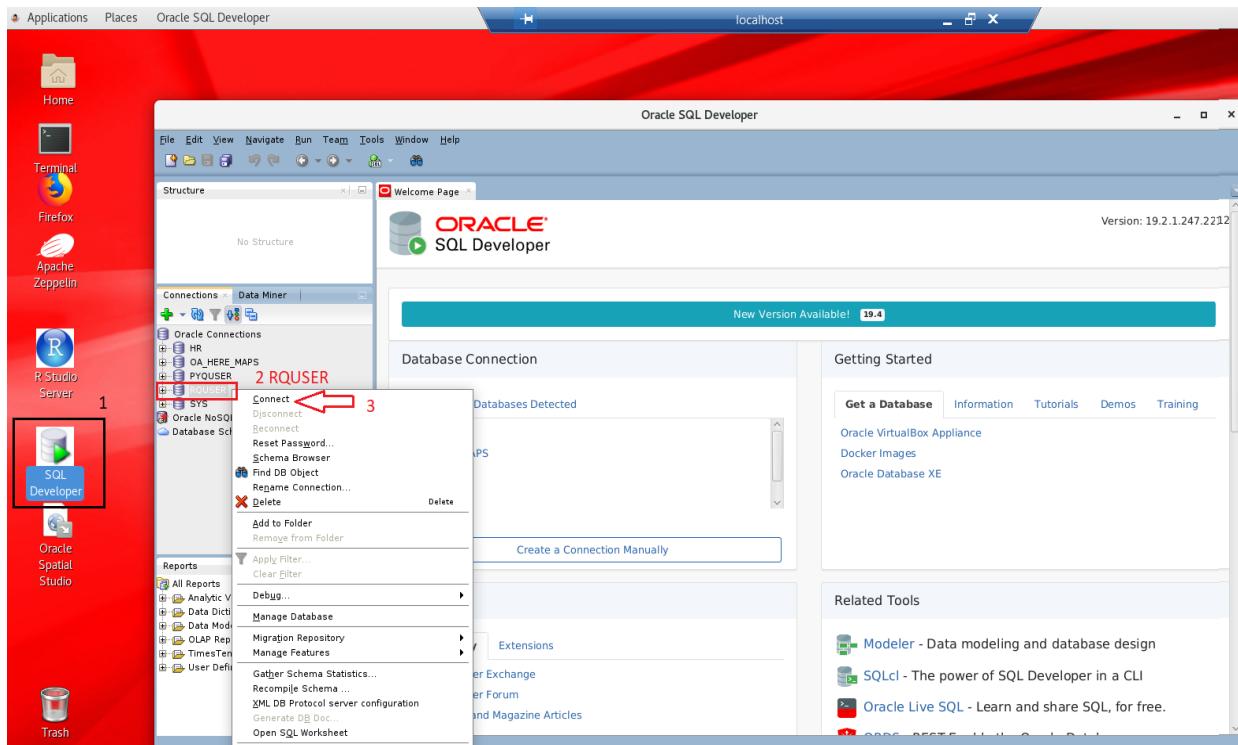


Además del código R proporcionado, en la carpeta OML4R hay un script con extensión sql: Short\_Demo\_Script.sql que contiene sentencias SQL que deben ejecutarse fuera de RStudio. También durante la ejecución de alguno de los notebooks, se le pedirá que ejecute algunas sentencias SQL.

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).



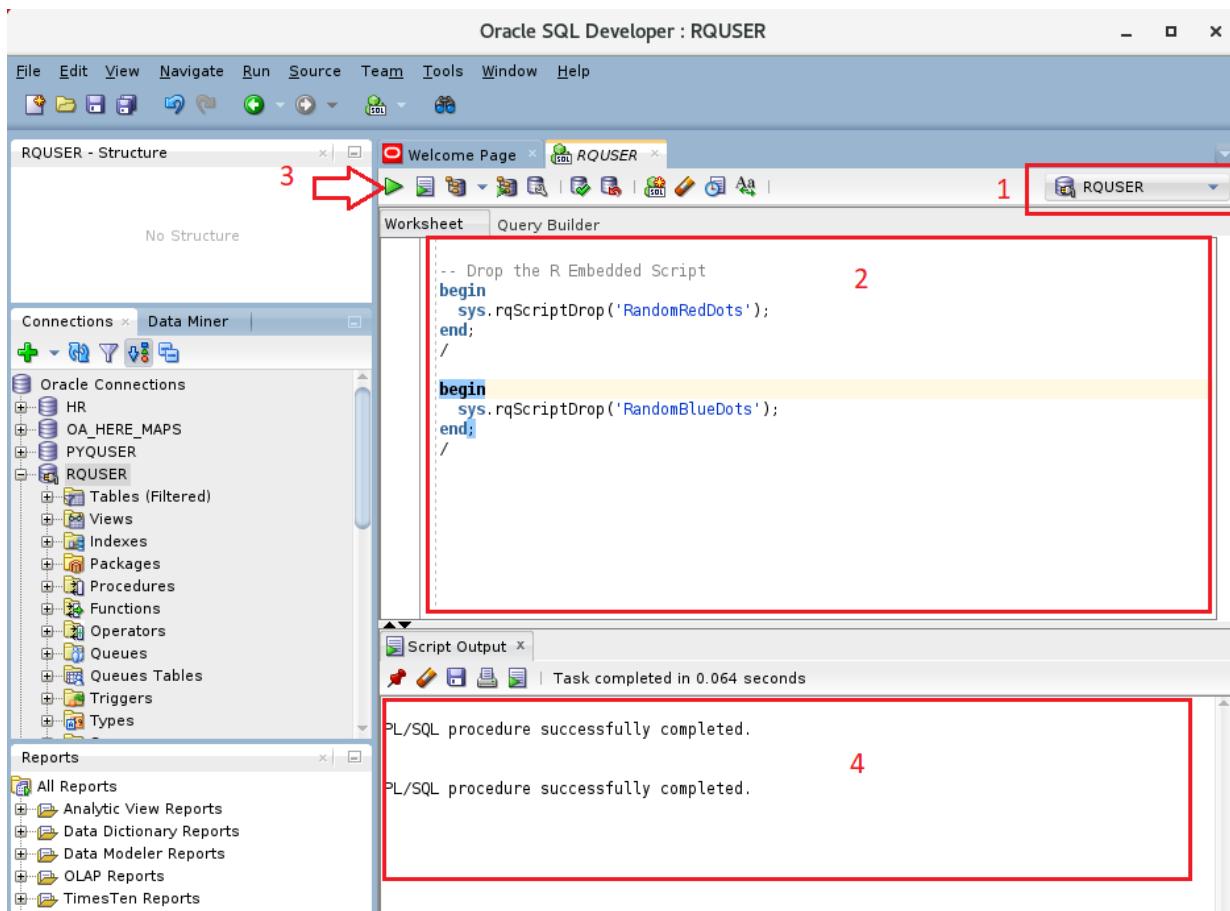
Una vez conectado como RQUSER (1), en el lado derecho de la pantalla de SQL\*Developer tendrá una hoja de trabajo SQL o *Worksheet* (2), en la que puede ir pegando las diferentes sentencias SQL ó el código PL/SQL que se le proporcione en los notebooks.

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.



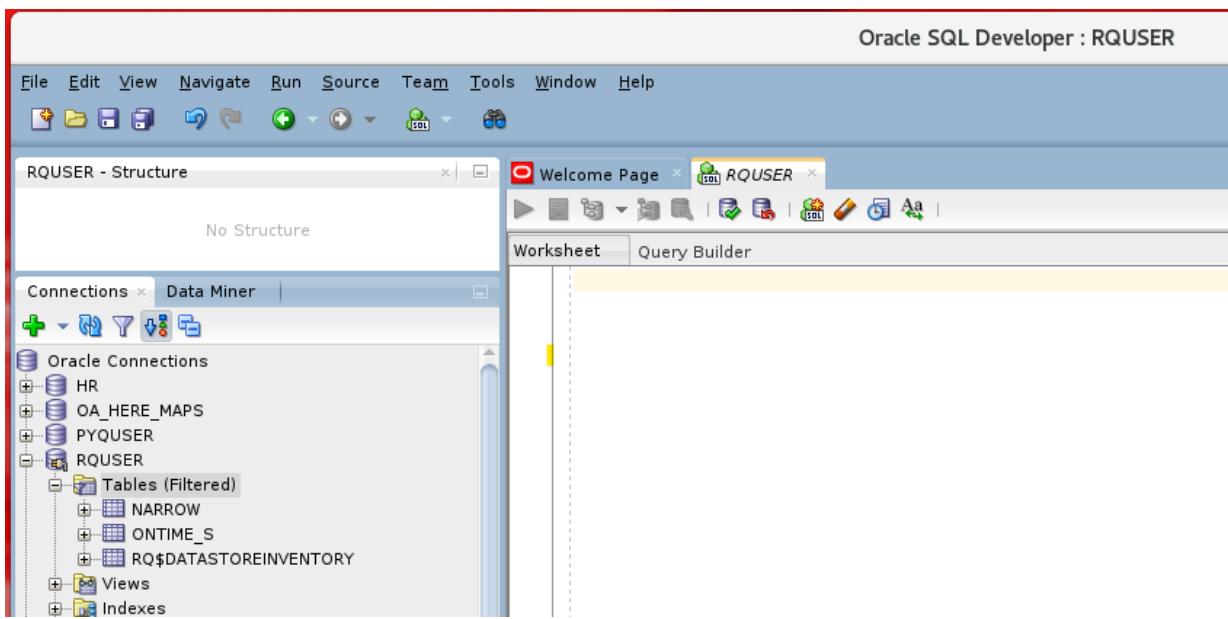


El contenido de cada notebook está escrito de manera que pueda entender cada bloque de código R, aunque no sea un programador experto de R.

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](https://docs.oracle.com/cd/E67822_01/index.htm)

Esta documentación incluye:

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

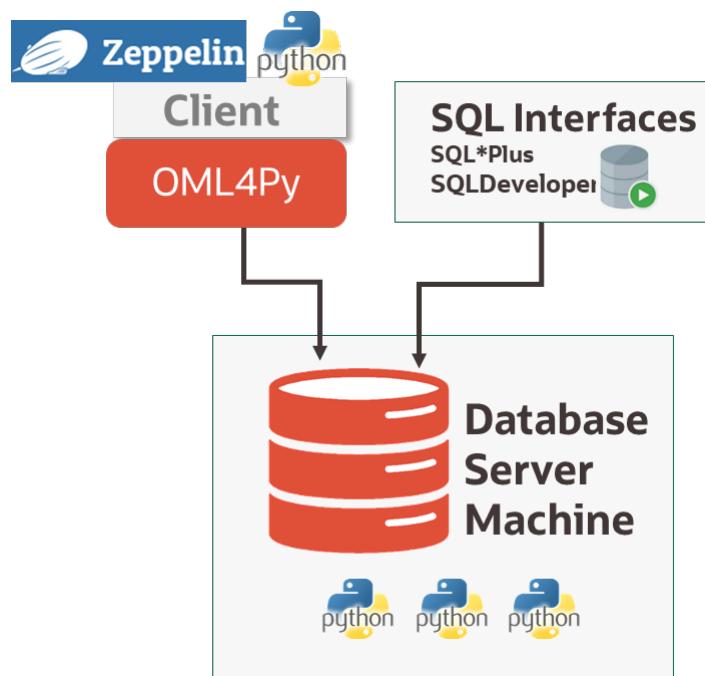
Consulte esta documentación para obtener un mayor conocimiento de ORE.



## OML4Py (1 hora)

Todas las actividades de esta sección están preparadas para ser realizadas a través de *Zeppelin*, 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 Zeppelin

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

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





La página de acceso se mostrará en pantalla:

A screenshot of a web browser window titled "Zeppelin". The address bar shows "localhost:8080/#". The main content area displays the "Welcome to Zeppelin!" message and a brief description of the tool. Overlaid on the main content is a "Login" dialog box. The dialog box has two input fields: "User Name" and "Password", and a "Login" button at the bottom right.

Una vez introducida la credencial de acceso, use el navegador para entrar en la carpeta *OML4Py* donde están los notebooks a usar durante el workshop.



The screenshot shows the Zeppelin web-based notebook interface running in Mozilla Firefox. The title bar reads "Zeppelin - Mozilla Firefox". The address bar shows "localhost:8080/#/". The main content area displays the "Welcome to Zeppelin!" page. On the left, there is a sidebar titled "Notebook" with options like "Import note", "Create new note", and a "Filter" input field. A red box highlights the "OML4Py" option under "Notebook". To the right of the sidebar is a "Help" section with links to documentation, mailing lists, issues tracking, and GitHub. A large blue circular graphic is on the right side of the page.

A continuación, se muestran los 5 notebooks que se usarán durante esta sesión dedicada a OML4Py:

The screenshot shows the Zeppelin web-based notebook interface running in Mozilla Firefox. The title bar reads "Zeppelin - Mozilla Firefox". The address bar shows "localhost:8080/#/". The main content area displays the "Welcome to Zeppelin!" page. On the left, there is a sidebar titled "Notebook" with options like "Import note", "Create new note", and a "Filter" input field. A red box highlights the "OML4Py" section, which contains five sub-notebooks: "0 - OML4Py Tour", "1 - OML4Py - Introduction", "2 - OML4Py Data Selection and Manipulation", "3 - OML4Py Datastore and Script Repository", and "4 - OML4Py Embedded Python Execution". To the right of the sidebar is a "Help" section with links to documentation, mailing lists, issues tracking, and GitHub. A large blue circular graphic is on the right side of the page.



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.

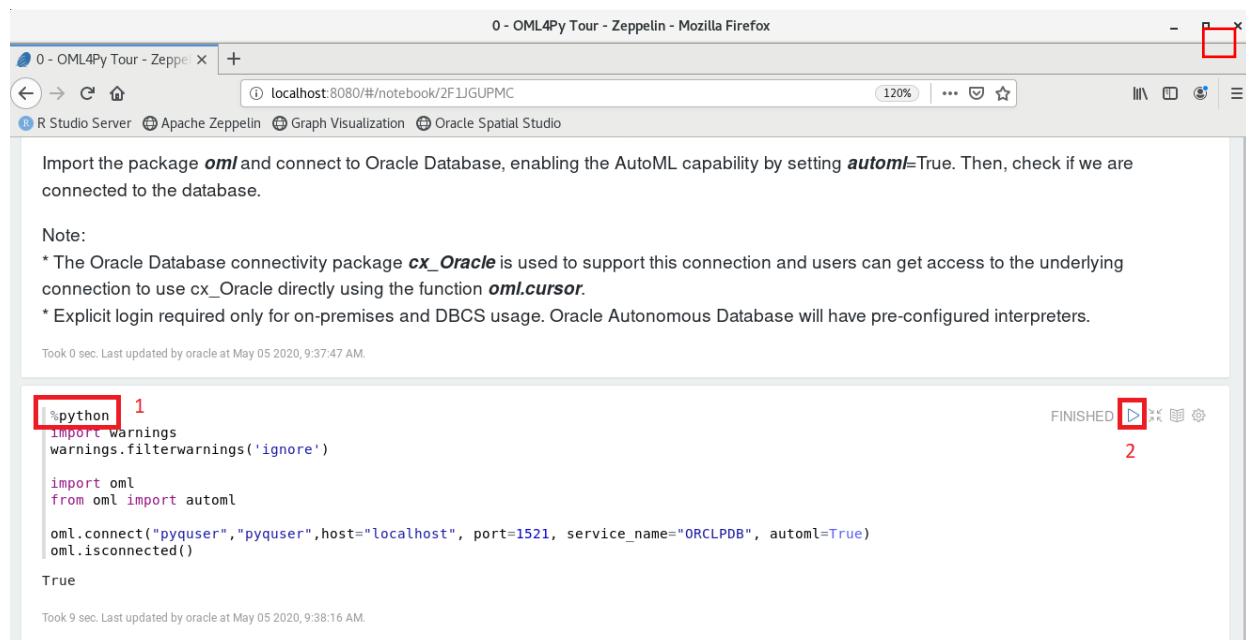
Cada notebook está compuesto por dos tipos de párrafos que contienen

- bloques de texto en formato markdown
- código python

Ambos tienen un botón con el símbolo de *Play* en la zona superior derecha 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: 0-OML4Py Tour abierto en el navegador. Tras los primeros párrafos de markdown está el primer párrafo de Python, que empieza por **%python(1)** y resaltado el botón de ejecución de este primer bloque de código Python (2).



Import the package **oml** and connect to Oracle Database, enabling the AutoML capability by setting **automl=True**. Then, check if we are connected to the database.

Note:

- \* The Oracle Database connectivity package **cx\_Oracle** is used to support this connection and users can get access to the underlying connection to use cx\_Oracle directly using the function **oml.cursor**.
- \* Explicit login required only for on-premises and DBCS usage. Oracle Autonomous Database will have pre-configured interpreters.

Took 0 sec. Last updated by oracle at May 05 2020, 9:37:47 AM.

```
%python 1
import warnings
warnings.filterwarnings('ignore')

import oml
from oml import automl

oml.connect("pyquser","pyquser",host="localhost", port=1521, service_name="ORCLPDB", automl=True)
oml.isconnected()

True
```

Took 9 sec. Last updated by oracle at May 05 2020, 9:38:16 AM.

FINISHED 2

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, su estado cambiar de **FINISHED** a **RUNNING**, como se muestra a continuación:



```
%python
import warnings
warnings.filterwarnings('ignore')

import oml
from oml import autonl

oml.connect("pyquser","pyquser",host="localhost", port=1521, service_name="ORCLPDB", autonl=True)
oml.isconnected()
```

Started a few seconds ago.

Una vez terminada la ejecución, el estado pasa a *FINISHED* (1). Si el código genera algún tipo de salida, se mostrará justo debajo del párrafo (2). En caso de que la salida sean datos se muestran en forma tabular o con algún modo gráfico de visualización.

También se puede ver una línea de estado con el tiempo de duración de la ejecución, así como la fecha en que se completó el párrafo por última vez (3).

```
%python
import warnings
warnings.filterwarnings('ignore')

import oml
from oml import autonl

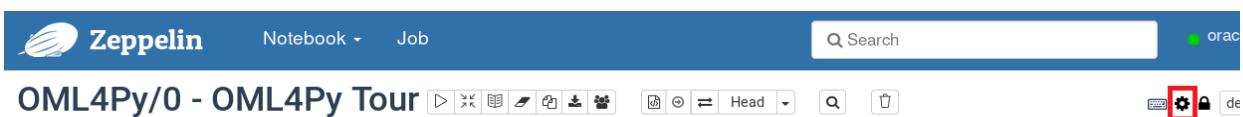
oml.connect("pyquser","pyquser",host="localhost", port=1521, service_name="ORCLPDB", autonl=True)
oml.isconnected()
```

True 2

Took 8 sec. Last updated by oracle at May 26 2020, 2:37:49 PM. 3

Si encuentra problemas con la ejecución, puede ser necesario reiniciar el intérprete 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 intérprete de Python, en la barra superior del notebook haga click en la rueda dentada (*Interpreter Binding*):



Una vez abierta la opción de configuración, se muestran todos los intérpretes disponibles.

Localice el intérprete de python (1) en la lista y haga click en el ícono a su izquierda (2). Se mostrará un diálogo de confirmación antes de hacer el reinicio. Confirme la operación de reinicio.

Pulsando de nuevo en la rueda dentada se cierra por completo el menú de los intérpretes y se comienza desde el inicio la ejecución del notebook.



The screenshot shows the Apache Zeppelin interface with the title "0 - OML4Py Tour - Zeppelin". The URL is "localhost:8080/#/notebook/2F1JGUPMC". The top navigation bar includes links for "R Studio Server", "Apache Zeppelin", "Graph Visualization", and "Oracle Spatial Studio". The main header has "Zeppelin", "Notebook", "Job", a search bar, and a dropdown for "oracle". Below the header is the title "OML4Py/0 - OML4Py Tour". The "Settings" section is open, showing a list of interpreters:

- spark %spark (default), %sql, %dep, %pyspark, %ipyspark, %r
- md %md
- angular %angular
- sh %sh
- livy %livy, %livy.sql, %livy.pyspark, %livy.pyspark3, %livy.sparkr, %livy.shared
- alluxio %alluxio
- file %file
- flink %flink** (highlighted with a red arrow)
- python %python, %python.ipython, %python.sql, %python.conda, %python.docker** (highlighted with a red box)

En este caso se va a usar el esquema PYQUSER que contiene las siguientes tablas:

The screenshot shows the Oracle SQL Developer interface with the title "Welcome Page" and a tab for "PYQUSER". The top menu bar includes "File", "Edit", "View", "Navigate", "Run", "Team", "Tools", "Window", and "Help". The toolbar includes various icons for file operations. The left sidebar is titled "Structure" and shows "No Structure". The main pane displays the "Connections" tab, which lists "Oracle Connections" including "HR", "OA\_HERE\_MAPS", and "PYQUSER". Under "PYQUSER", the "Tables (Filtered)" section is expanded, showing tables: BREASTCANCER, DIGITS, IRIS, NEW\_IRIS, ONTIME\_S, PYQ\$DATASTOREINVENTORY, RF\_COST, and SAMPLE\_IRIS. A red box highlights the "PYQUSER" connection in the connections list.



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 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.

### 0 - OML4Py Tour

Este notebook ofrece una visión general de la funcionalidad que ofrece OML4Py. El objetivo es entender la forma de trabajar con los datos dentro de la Base de Datos, llevando la exploración, análisis previo y ejecución de los algoritmos dentro de la misma, usándola como sistema de ejecución y almacenamiento de los resultados.

Es un notebook bastante extenso que cubre gran parte de lo que ofrece OML4Py.



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
- Algoritmos de ML in-database
- Capa de transparencia
- Persistir información en bbdd
- Manipulación y visualización de datos
- Ejecución embebida

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

## 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 Zeppelin.
- 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.

