MaxCompute

Tools and Downloads

MORE THAN JUST CLOUD | Alibaba Cloud

Tools and Downloads

Overview

The latest ODPS service provides a new version of the client, compared with the old version, the new client has the following improvements:

- You do not need to enter a command shell to execute a specified command. For example, you can enter the client and run SQL comamnds and security commands directly in new version. You do not need to enter SQL/Security shell at first and run corresponding commands.
- The commands of new version is more closed to Hive commands. Most of comnands are compatible with Hive.

Besides, you must note:

- This document is a part of User Manual for the client in ODPS. It describes how to use basic functions of ODPS by using the command lines of the client.
- Do not perform the analysis operation based on the output format of the client. The output format of the client is not ensured for forward compatibility. Clients in different versions are different in their command formats and behaviors.
- ODPS client is a java program and can run only in the JRE environment. Therefore, it is required to download and install JRE 1.6 version.
- If you need to learn about the use method of the client, refer to Quick Start.

Download and Installation

To downloa ODPS client, please click on Here.

Depress the downloaded file and you can find the following four folders:

```
bin/  conf/  lib/  plugins/
```

There is a file named odps_conf.ini in ‘conf’. Edit this file and fill in related information:

```
project_name=
access_id=<accessid>
access_key=<accesskey>
```

```
end_point=http://xxxx
```

Note:

- Replace the access_id and access_key with the access_id and access_key applied from www.aliyun.com.
- If you often use a project, you can add the name of this project behind of "project_name=", which can avoid executing "use project_name;" command when entering the client.
- end_point. The access URL differs for different regions. For more information, see Access domains and data centers.

After the configuration file has been modified, run 'odpscmd' in bin directory. (If the OS is Linux, run './bin/odpscmd'; if the OS is Windows, run './bin/odpscmd.bat'.)Now you can run ODPS commands:

```
 odps@ test_project> whoami;
Name: ALIYUN$test_user@aliyun.com
End_Point: http://service.ap-southeast-1.maxcompute.aliyun.com/api
Project: test_project
```

Get Help

View the information about help. Commnd Format:

```
odps@ >./bin/odpscmd -h;
```

Or you can type "h" or "help" in an interactive mode. (Case insensitive)

Starting Parameters

When starting the client, you can specify a series of parameters:

```
Usage: odpscmd [OPTION]...
where options include:
--help (-h)for help
--project=<prj_name> use project
--endpoint=<http://host:port> set endpoint
-u <user_name> -p <password> user name and password
-k <n> will skip begining queries and start from specified position
-r <n> set retry times
-f <"file_path;"> execute command in file
-e <"command;[command;]..."> execute command, include sql command
```

```
-C will display job counters
```

Example: (take '-f' as an example)

- Prepare the local script file 'script.txt'. Suppose that the file is located in the disk D and the content is shown as follows:

```
DROP TABLE IF EXISTS test_table_mj;
CREATE TABLE test_table_mj (id string, name string);
DROP TABLE test_table_mj;
```

- Running Command:

```
odpscmd\bin>odpscmd -f d:/script.txt;
```

- Display the execution result:

```
ID = 20150528122432906gux77io3
Log view:
http://webconsole.odps.aliyun-inc.com:8080/logview/?h=http://service-corp.odps.aliyun-
inc.com/api&p=odps_public_dev&i=2015052
8122432906gux77io3&token=RnlrSzJoL242YW43dFFIc1dmb1ZWZzFxQ1RFPSxPRFBTX09CTzoxMDcwMDI1NjI3ODA
1NjI5LDE0MzM0MjA2NzMseyJTdGF0ZW1l
bnQiOlt7IkFjdGlvbiI6WyJvZHBzOlJlYWQiXSwiRWZmZWN0IjoiQWxsb3ciLCJSZXNvdXJjZSI6WyJhY3M6b2RwczoqOnB
yb2plY3RzL29kcHNfcHVibGljX2Rld
i9pbnN0YW5jZXMvMjAxNTA1MjgxMjI0MzI5MDZndXg3N2lvMyJdfV0sIlZlcnNpb24iOiIxIn0=
OK

ID = 20150528122439318gcmkk6u1
Log view:
http://webconsole.odps.aliyun-inc.com:8080/logview/?h=http://service-corp.odps.aliyun-
inc.com/api&p=odps_public_dev&i=2015052
8122439318gcmkk6u1&token=dSt0RXdlV0M5YjZET2I1MnJuUFkzWDN1aWpzPSxPRFBTX09CTzoxMDcwMDI1NjI3O
DA1NjI5LDE0MzM0MjA2ODAseyJTdGF0ZW1l
bnQiOlt7IkFjdGlvbiI6WyJvZHBzOlJlYWQiXSwiRWZmZWN0IjoiQWxsb3ciLCJSZXNvdXJjZSI6WyJhY3M6b2RwczoqOnB
yb2plY3RzL29kcHNfcHVibGljX2Rld
i9pbnN0YW5jZXMvMjAxNTA1MjgxMjI0MzkzMThnY21razZ1MSJdfV0sIlZlcnNpb24iOiIxIn0=
OK

ID = 20150528122440389g98cmlmf
Log view:
http://webconsole.odps.aliyun-inc.com:8080/logview/?h=http://service-corp.odps.aliyun-
inc.com/api&p=odps_public_dev&i=2015052
8122440389g98cmlmf&token=NWlwL0EvQThxUXhzcTRERDc5NFg0b2IxZ3QwPSxPRFBTX09CTzoxMDcwMDI1NjI3O
DA1NjI5LDE0MzM0MjA2ODAseyJTdGF0ZW1l
bnQiOlt7IkFjdGlvbiI6WyJvZHBzOlJlYWQiXSwiRWZmZWN0IjoiQWxsb3ciLCJSZXNvdXJjZSI6WyJhY3M6b2RwczoqOnB
yb2plY3RzL29kcHNfcHVibGljX2Rld
i9pbnN0YW5jZXMvMjAxNTA1MjgxMjI0NDAzODlnOThjbWxtZiJdfV0sIlZlcnNpb24iOiIxIn0=
OK
```

Interactive Mode

Directly running the client will enter the interactive mode.

```
[admin: ~]$odpscmd
Aliyun ODPS Command Line Tool
Version 1.0
@Copyright 2012 Alibaba Cloud Computing Co., Ltd. All rights reserved.
odps@ odps> INSERT OVERWRITE TABLE DUAL SELECT * FROM DUAL;
```

Input the command at the cursor position (take a semicolon as a statement end mark) and press Enter to run.

Continuous Running

- When you use '-e' or '-f' option to run command, you can specify the parameter '-k' if there are multiple statements and you want to start running from a middle statement, which indicates ignoring the previous statements and running from the specified position. If the specified parameter <= 0, start running from the first statement.
- Each statement seperated by a semicolon is considered as a valid statement. The statements which runs successfully or fail to run will be printed at running time.
- For example, there are three SQL statements in the file '/tmp/dual.sql':

```
drop table dual;
create table dual (dummy string);
insert overwrite table dual select count(*) from dual;
```

To ignore the first two statements:

```
odpscmd –k 3 –f dual.sql
```

Get Current Login User

Command Format:

```
whoami;
```

Use Example:

```
odps@ hiveut>whoami;
```

```
Name: odpstest@aliyun.com
ID: 1090142773636588
End_Point: http://10.249.215.1/odps_debug1
Project: lijunsecuritytest
```

Use: get the aliyun account and end point configuration of current log user.

Quit

Command:

```
odps@ > quit;
or q;
```

Eclipse Plugins

To facilitate the development work with Java SDK of MapReduce and UDF, MaxCompute provides Eclipse Development Plug-in.This plug-in can simulate the running process of MapReduce and UDF to provide local debugging methods for users and provide the function for generating a simple template.

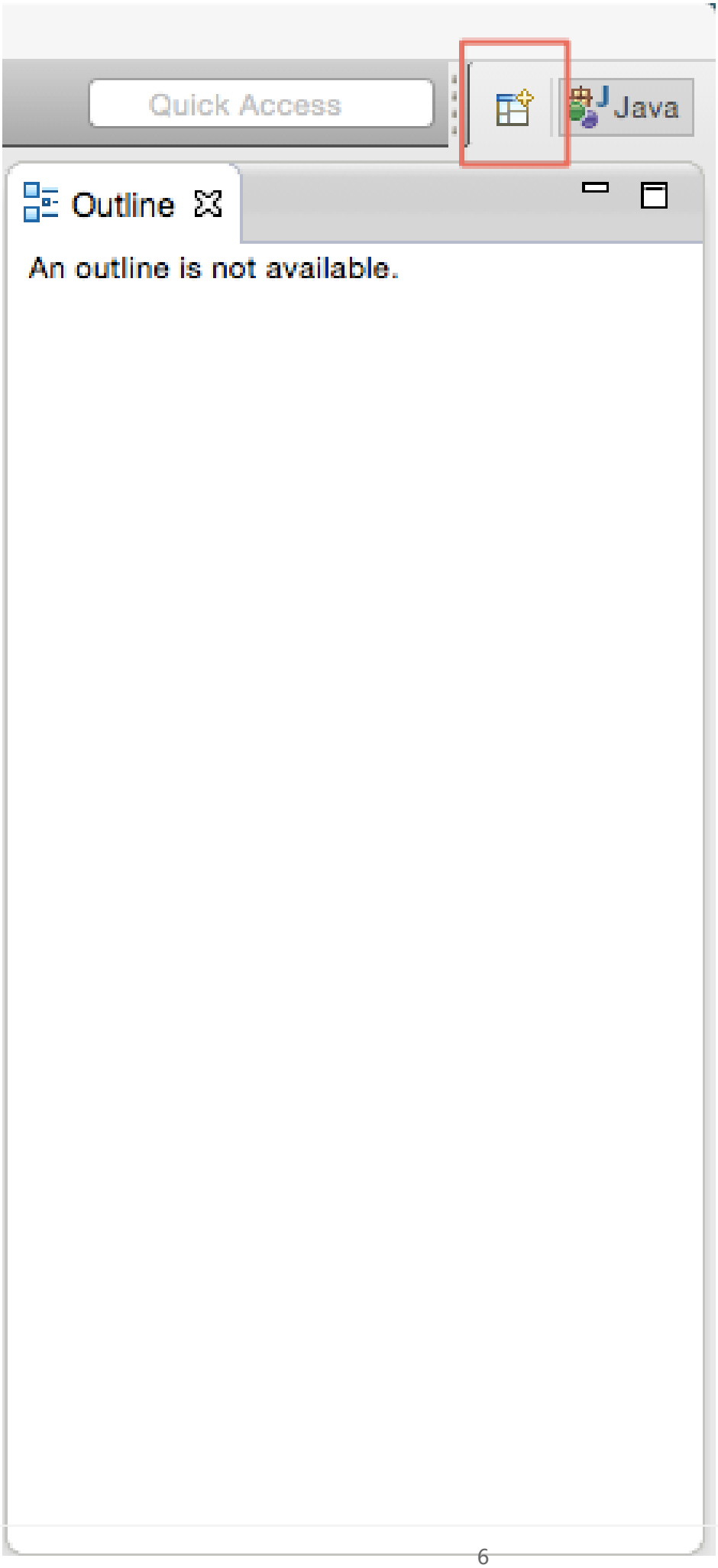
Note:

- To download this plug-in, click Here.
- Unlike the local running mode provided by MapReduce, Eclipse plug-in cannot synchronize data with MaxCompute. The data used by users need to be manually copied to the warehouse directory of Eclipse plugin.

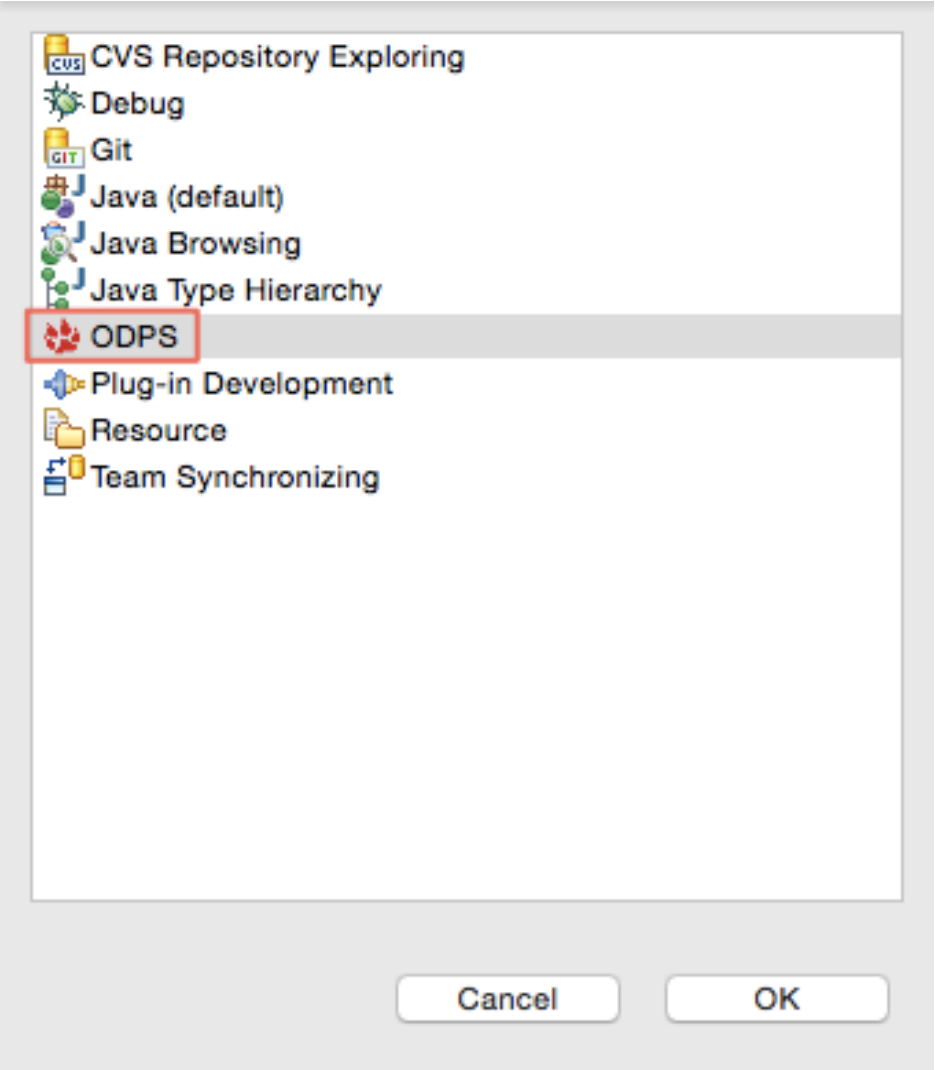
After downloading the Eclipse plug-in, decompress the software package to find the following jar:

```
odps-eclipse-plugin-bundle-0.15.0.jar
```

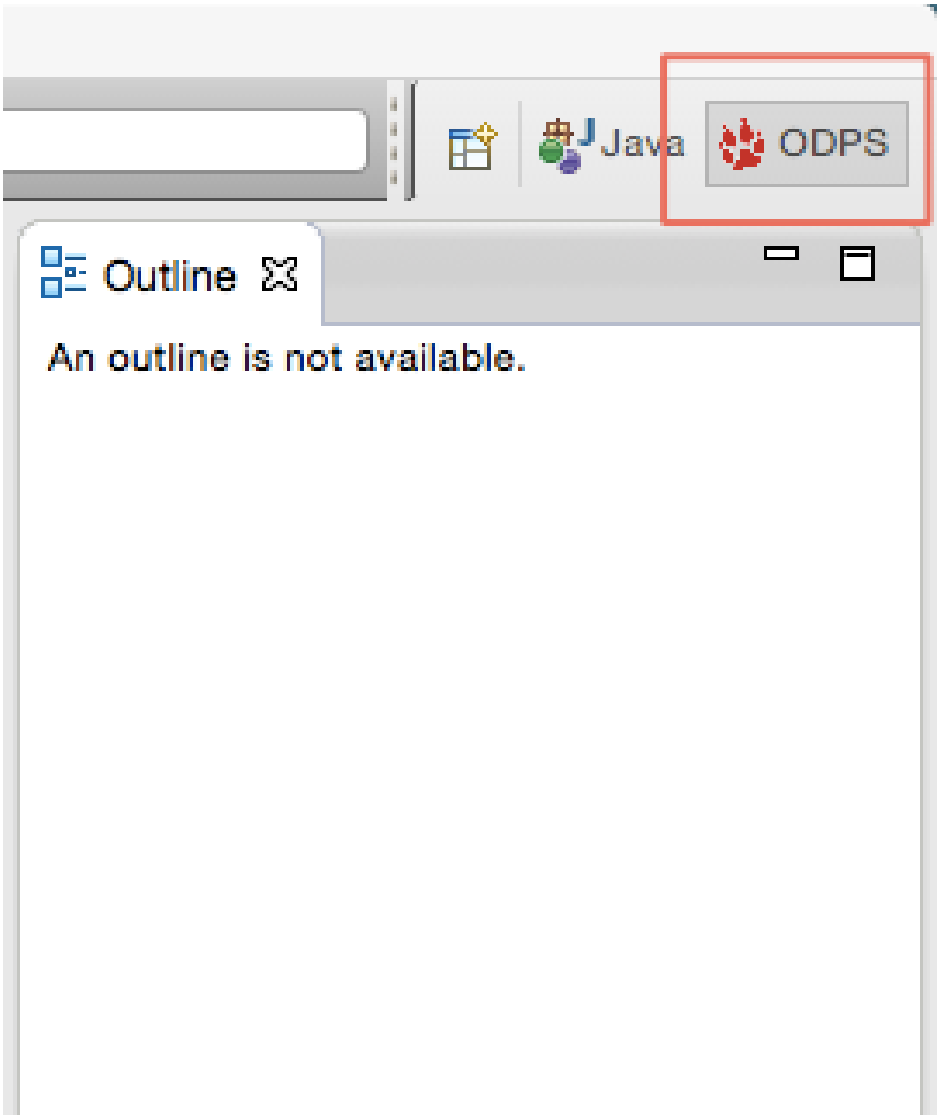
Place the plug-in into the subdirectory ‘plugins’ in Eclipse installation directory. Start the Eclipse plug-in, and click <Open Perspective> at the upper-right corner.


Quick Access
Java
Outline
An outline is not available.

After clicking the button, the following dialog box is displayed:



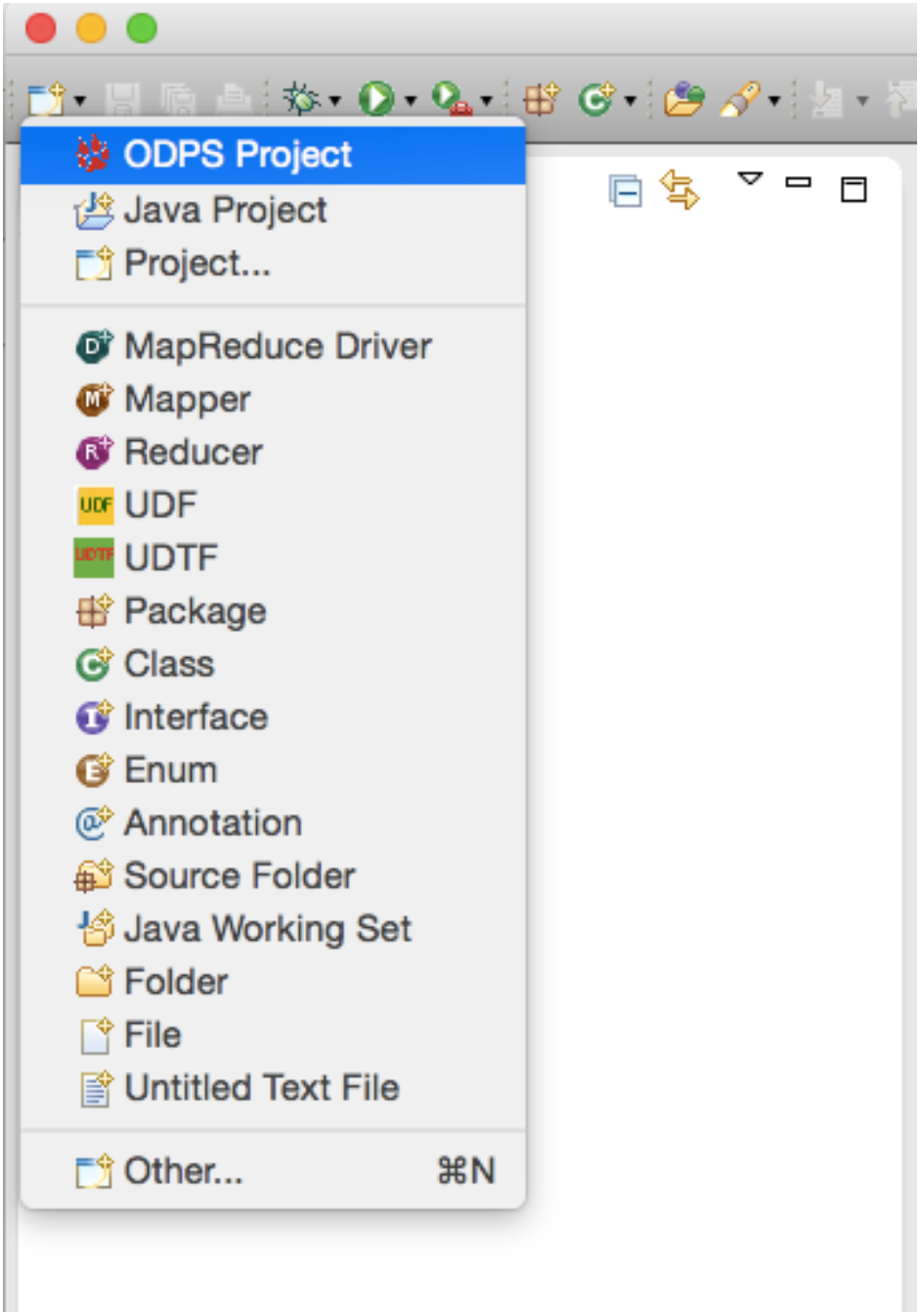
Select [ODPS] and click on **OK**. the MaxCompute icon will appear at the upper-right corner, indicating that the plug-in takes effect.



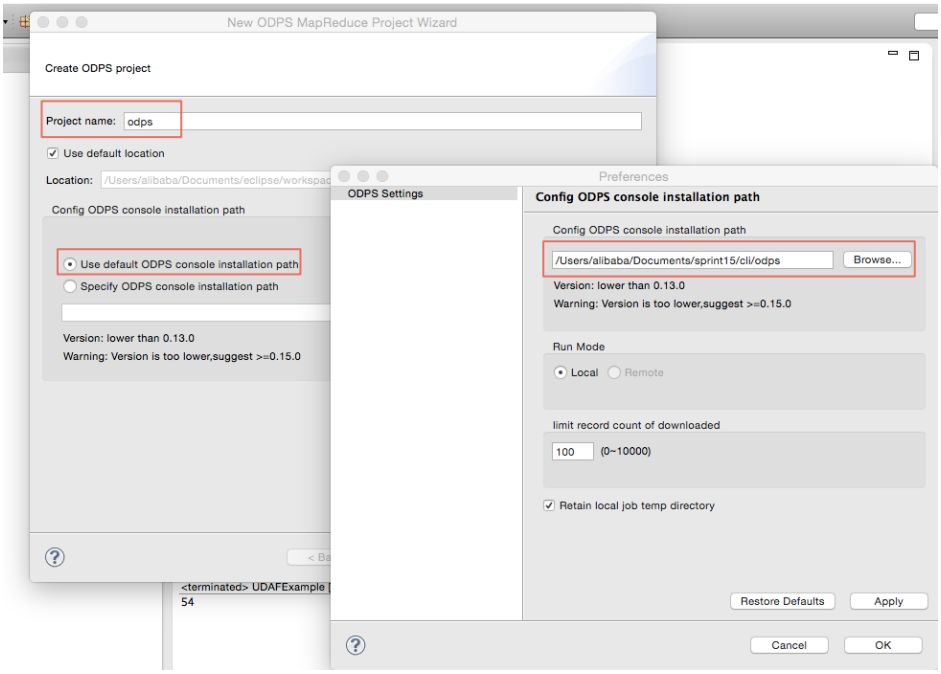
There are two methods to create MaxCompute project.

Method 1

Slect [File > New > Project > MaxCompute > MaxCompute Project] to create the project (in the example, use ‘ODPS’ as the peoject name):



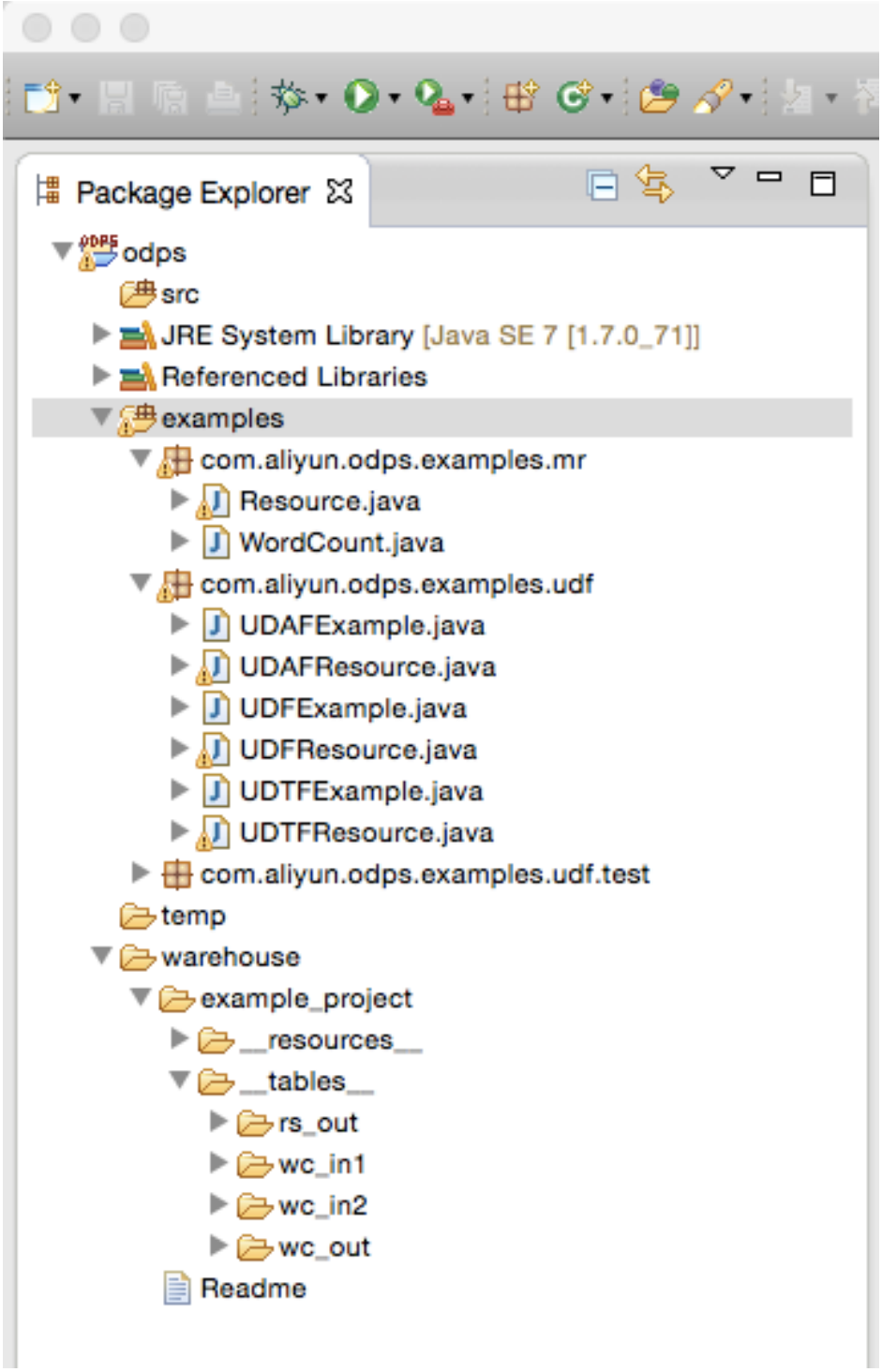
After creating MaxCompute project, the following dialog box will be popped up. Input Project name, and select the path of MaxCompute console. (The console should be uploaded at first.) At last, click <Finish>.



Note:

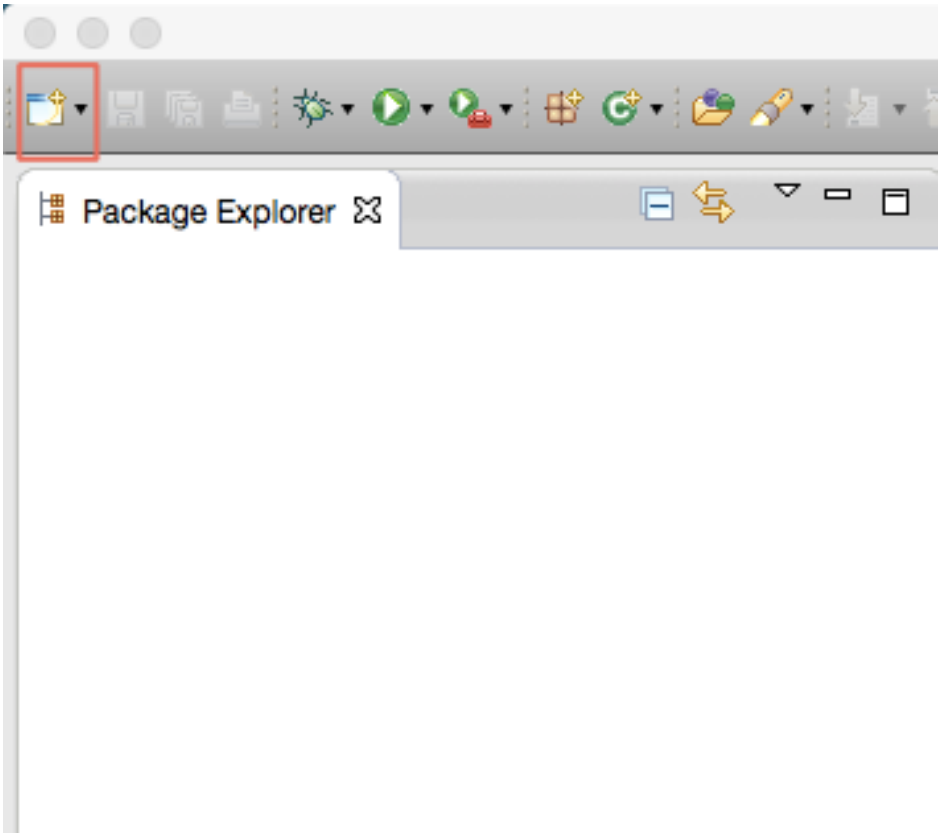
- For the introduction of MaxCompute console, please refer to Console.

creating project is finished, the following directory structure will be viewed in the left ‘Package Explorer’ :

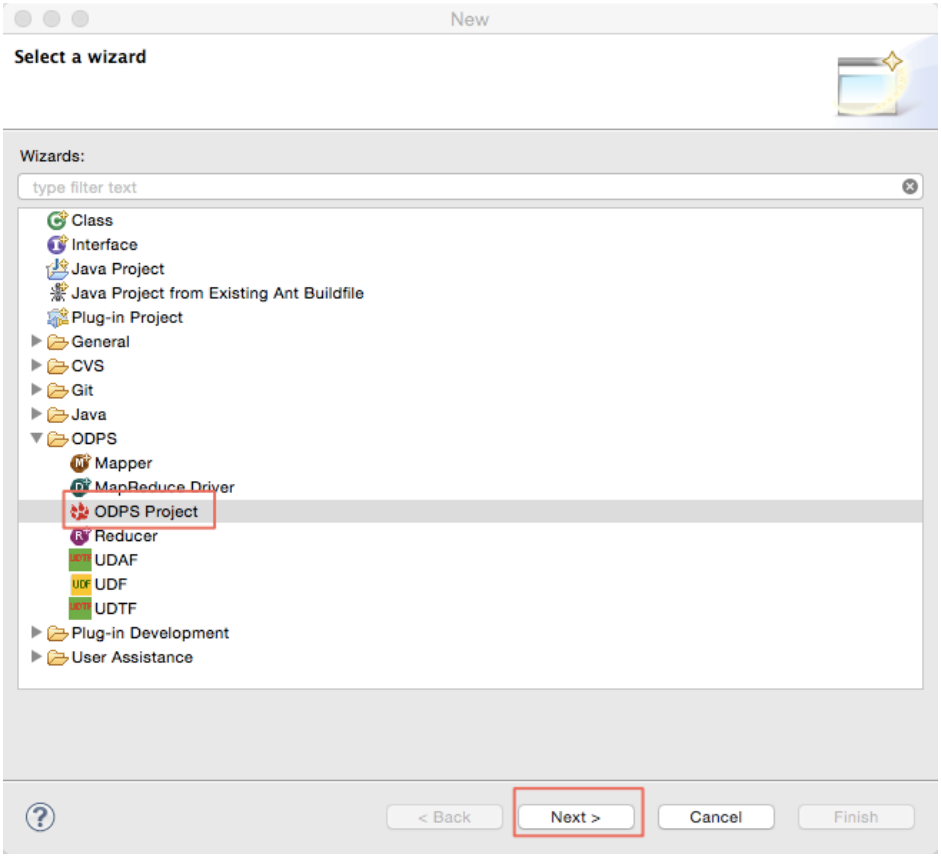


Method 2

Click <New> at the left-upper corner:



After the dialog box is popped up, select ‘ODPS Project’ and click on <Next>:



The subsequent operations are similar with Method 1.

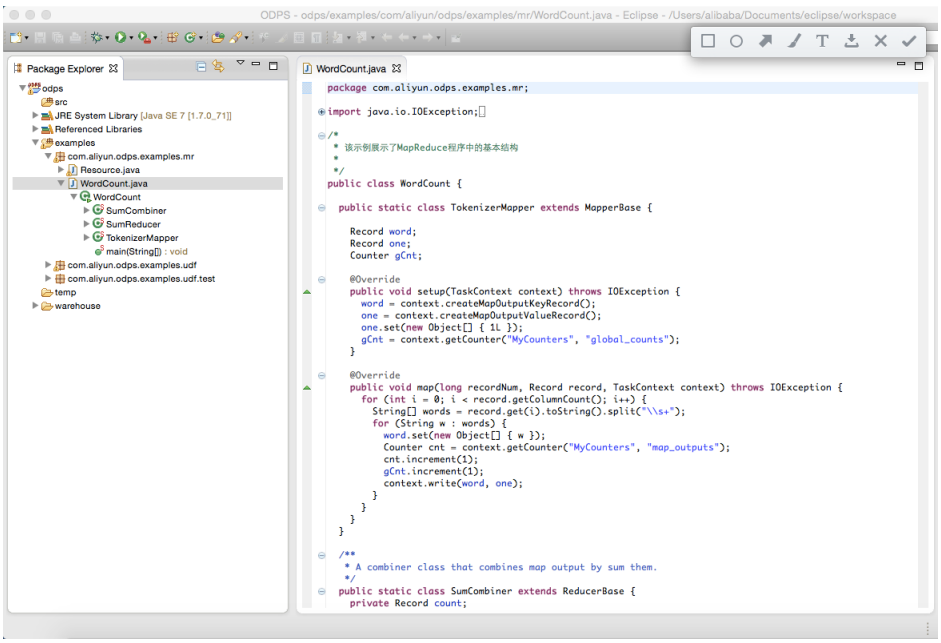
The installation of MaxCompute Eclipse plug-in is completed. User can use this plug-in to write MapReduce or UDF programs.

Note:

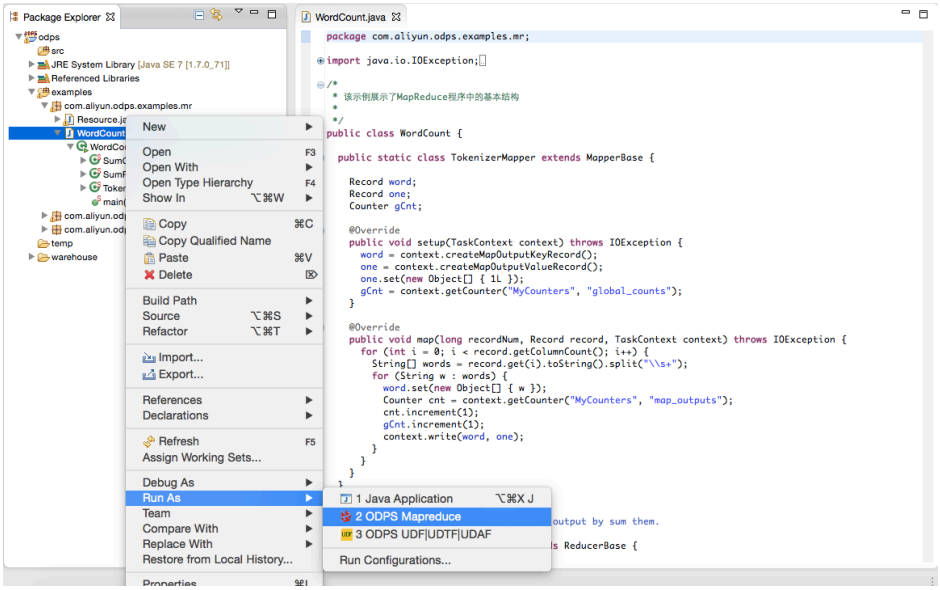
- For the function introduction of MapReduce in the plug-in, refer to MapReduce Development Plug-in Introduction.
- For the UDF program example, please refer to UDF Development Plug-in Introduction.

Run WordCount Example Quickly

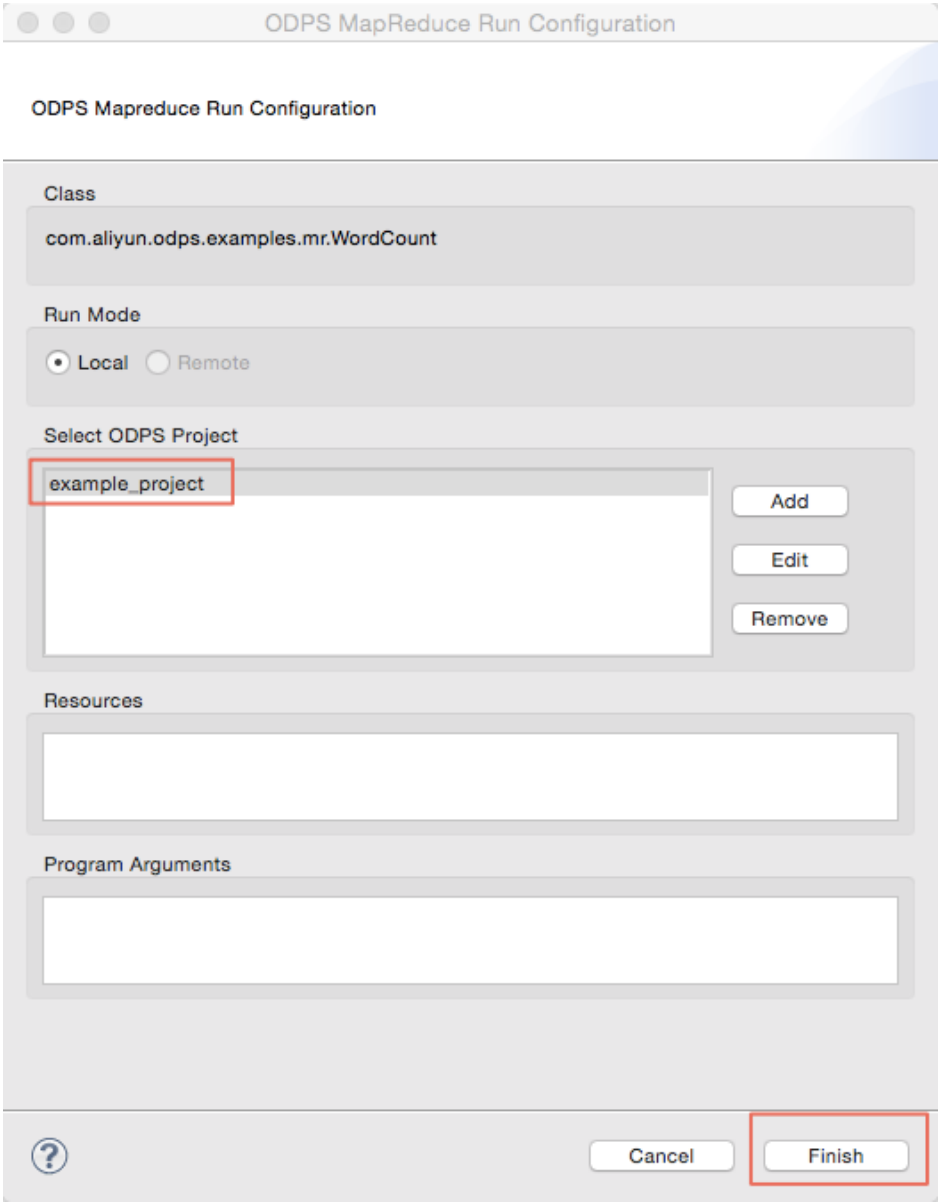
Select WordCount example in MaxCompute project:



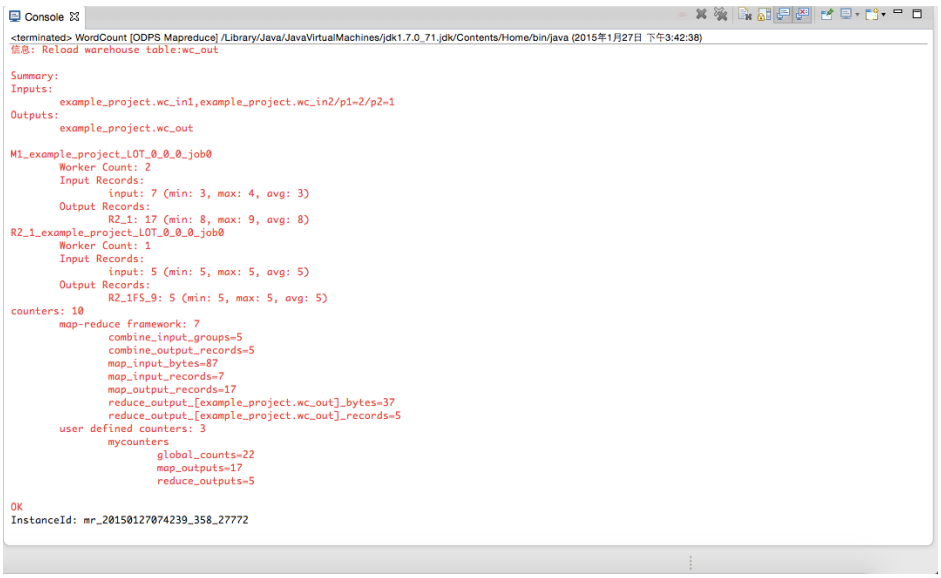
Right-click on 'WordCount.java' and click **Run As** -> **ODPS MapReduce**, as follows:



After the dialog box is popped up, select 'example_project' and click **Finish**:

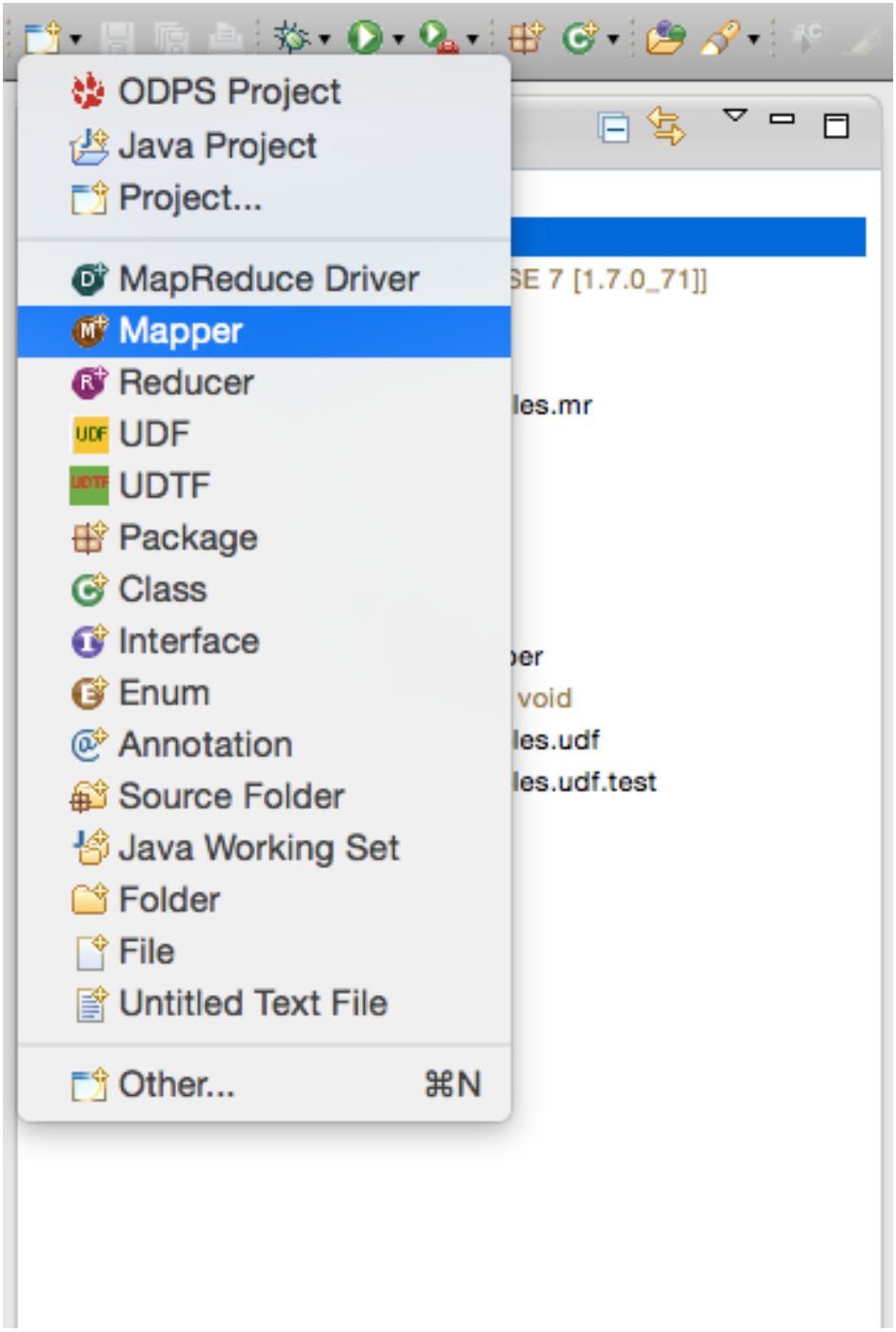


After running is successful, the following result will be displayed:

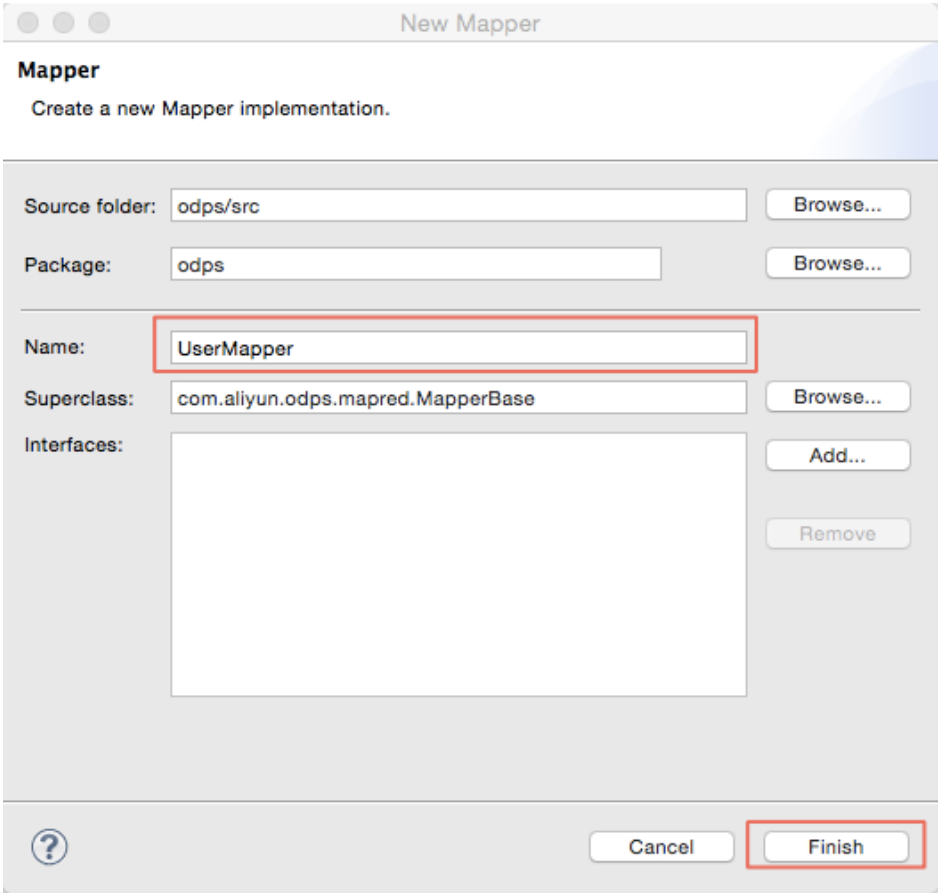


Run Uer-defined MapReduce Program

Right-click ‘src’ directory. Select **New -> Mapper**:



After selecting 'Mapper' and the following dialog box is displayed. Input the name of Mapper class and click **Finish**:



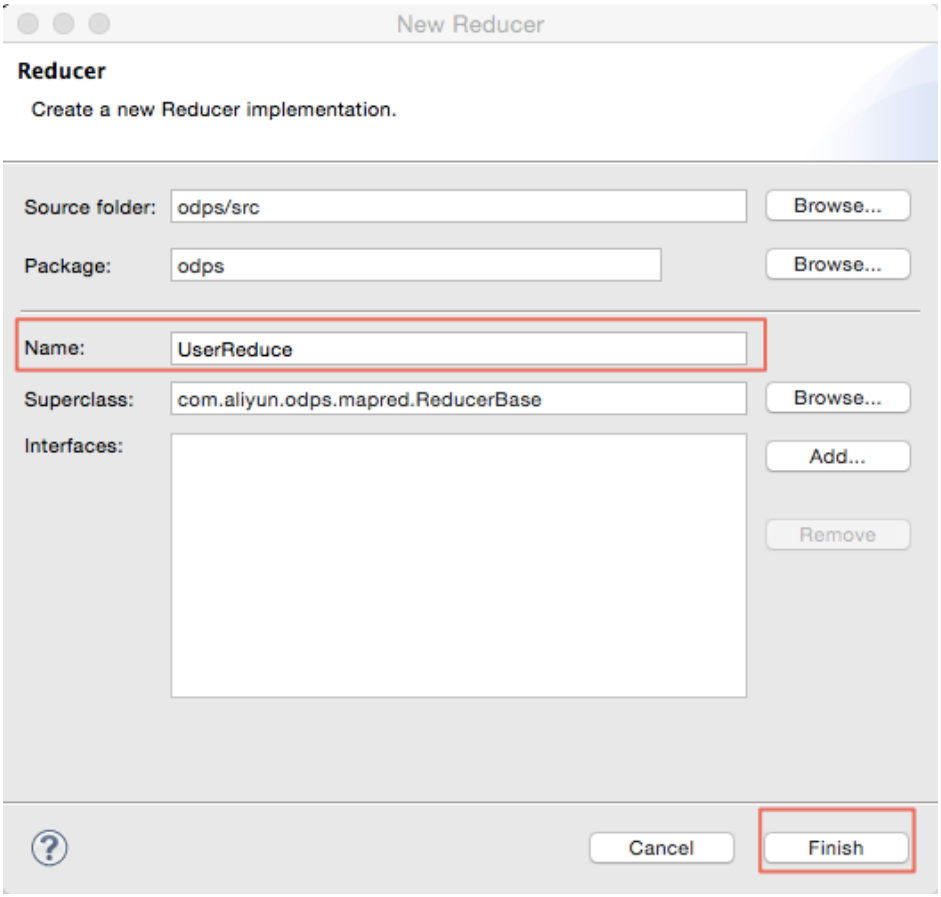
Now you can find a file 'UserMapper.java' is generated in the directory 'src' in 'Package Explorer'. The content of this file is a template of Mapper class:

```
package odps;
import java.io.IOException;
import com.aliyun.odps.data.Record;
import com.aliyun.odps.mapred.MapperBase;
public class UserMapper extends MapperBase {
@Override
public void setup(TaskContext context) throws IOException {
}
@Override
public void map(long recordNum, Record record, TaskContext context)
throws IOException {
}
@Override
public void cleanup(TaskContext context) throws IOException {
}
}
```

In the template, the configured package name defaults to 'odps'. You can modify it according to your actual requirement. Write the template contents as follows:

```
package odps;
import java.io.IOException;
import com.aliyun.odps.counter.Counter;
import com.aliyun.odps.data.Record;
import com.aliyun.odps.mapred.MapperBase;
public class UserMapper extends MapperBase {
Record word;
Record one;
Counter gCnt;
@Override
public void setup(TaskContext context) throws IOException {
word = context.createMapOutputKeyRecord();
one = context.createMapOutputValueRecord();
one.set(new Object[] { 1L });
gCnt = context.getCounter("MyCounters", "global_counts");
}
@Override
public void map(long recordNum, Record record, TaskContext context)
throws IOException {
for (int i = 0; i < record.getColumnCount(); i++) {
String[] words = record.get(i).toString().split("\\s+");
for (String w : words) {
word.set(new Object[] { w });
Counter cnt = context.getCounter("MyCounters", "map_outputs");
cnt.increment(1);
gCnt.increment(1);
context.write(word, one);
}
}
}
@Override
public void cleanup(TaskContext context) throws IOException {
}
}
```

In the same method, right-click ‘src’ directory and select **New -> Reduce**:



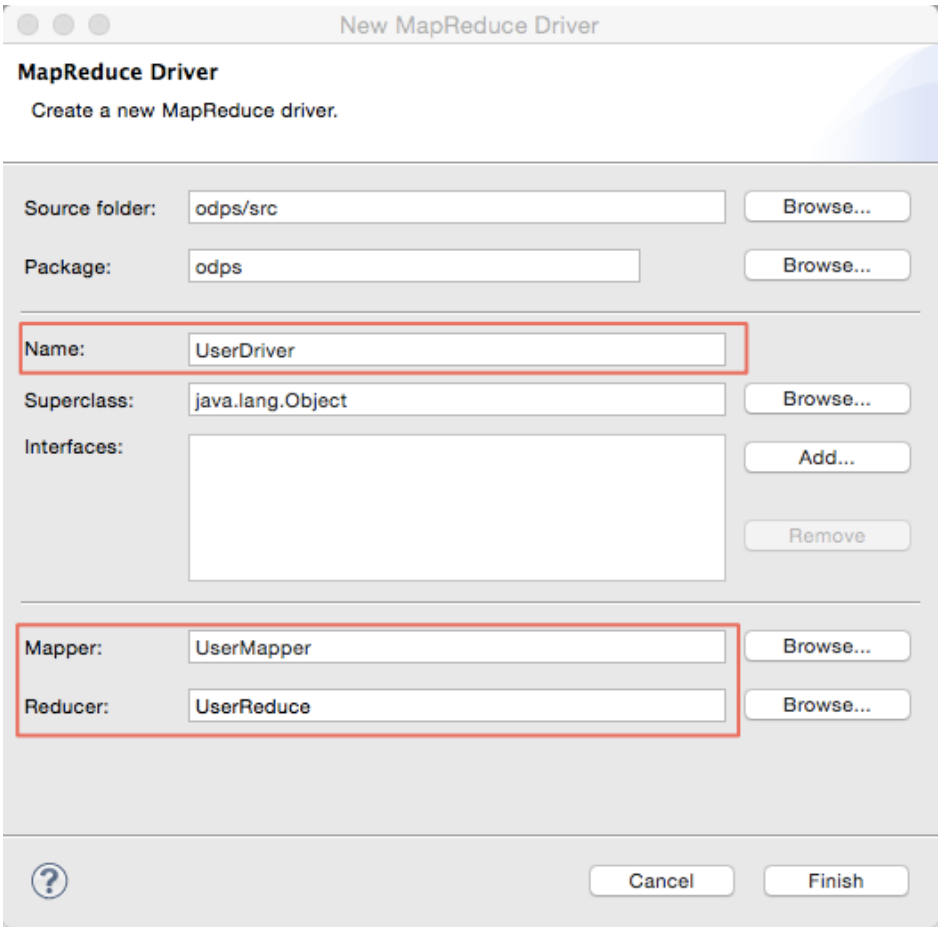
Input the name of Reduce class. (In this example, use 'UserReduce' as the class name):

In 'Package Explorer', a file name 'UseReduce.java' is generated in the directory 'src'. This file content is a template of Reduce class. Edit the template:

```
package odps;
import java.io.IOException;
import java.util.Iterator;
import com.aliyun.odps.counter.Counter;
import com.aliyun.odps.data.Record;
import com.aliyun.odps.mapred.ReducerBase;
public class UserReduce extends ReducerBase {
private Record result;
Counter gCnt;
@Override
public void setup(TaskContext context) throws IOException {
result = context.createOutputRecord();
gCnt = context.getCounter("MyCounters", "global_counts");
}
@Override
public void reduce(Record key, Iterator<Record> values, TaskContext context)
throws IOException {
long count = 0;
while (values.hasNext()) {
Record val = values.next();
```

```
count += (Long) val.get(0);
}
result.set(0, key.get(0));
result.set(1, count);
Counter cnt = context.getCounter("MyCounters", "reduce_outputs");
cnt.increment(1);
gCnt.increment(1);

context.write(result);
}
@Override
public void cleanup(TaskContext context) throws IOException {
}
}
```

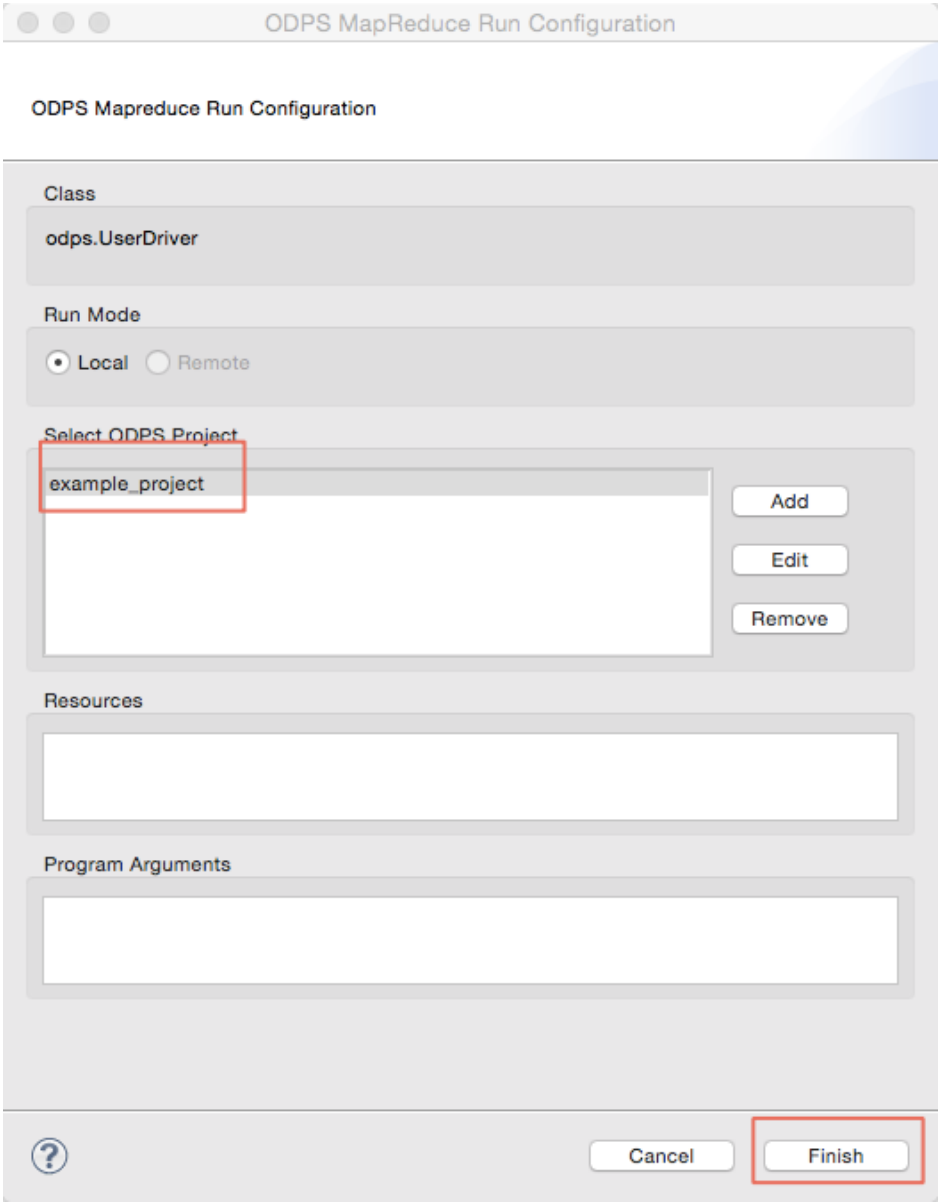
Create 'main' function: right-click 'src' and select **New -> MapReduce Driver**. Fill in Driver Name (in this example, use 'UserDriver' as the name), Mapper and Recduce (in this example use 'UserMapper' and 'UserReduce' as corresponding name) and click **Finish**;. The file 'MyDriver.java file' is also displayed in 'src' directory:



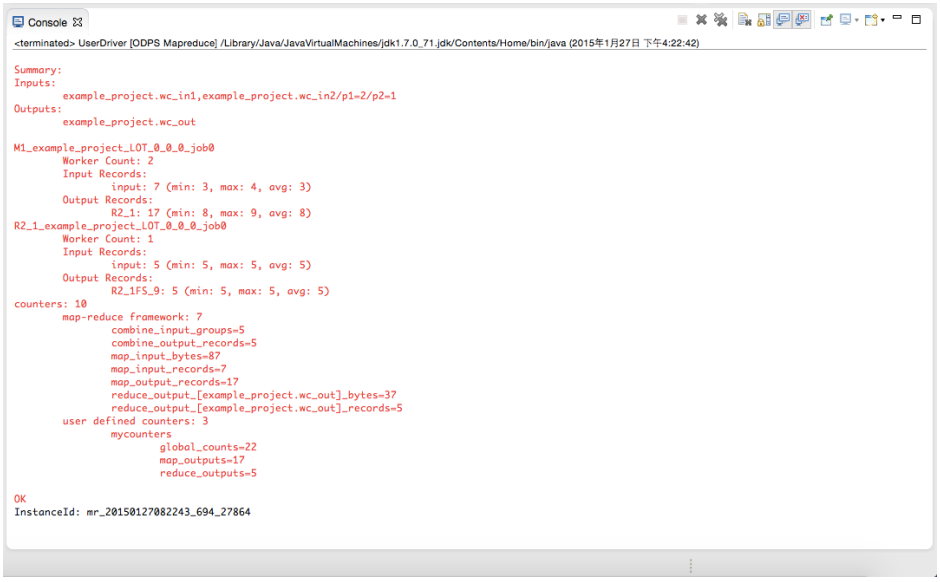
Edit the contents of driver:

```
package odps;
import com.aliyun.odps.OdpsException;
import com.aliyun.odps.data.TableInfo;
import com.aliyun.odps.examples.mr.WordCount.SumCombiner;
import com.aliyun.odps.examples.mr.WordCount.SumReducer;
import com.aliyun.odps.examples.mr.WordCount.TokenizerMapper;
import com.aliyun.odps.mapred.JobClient;
import com.aliyun.odps.mapred.RunningJob;
import com.aliyun.odps.mapred.conf.JobConf;
import com.aliyun.odps.mapred.utils.InputUtils;
import com.aliyun.odps.mapred.utils.OutputUtils;
import com.aliyun.odps.mapred.utils.SchemaUtils;
public class UserDriver {
public static void main(String[] args) throws OdpsException {
JobConf job = new JobConf();
job.setMapperClass(TokenizerMapper.class);
job.setCombinerClass(SumCombiner.class);
job.setReducerClass(SumReducer.class);
job.setMapOutputKeySchema(SchemaUtils.fromString("word:string"));
job.setMapOutputValueSchema(SchemaUtils.fromString("count:bigint"));
InputUtils.addTable(
TableInfo.builder().tableName("wc_in1").cols(new String[] { "col2", "col3" }).build(), job);
InputUtils.addTable(TableInfo.builder().tableName("wc_in2").partSpec("p1=2/p2=1").build(), job);
OutputUtils.addTable(TableInfo.builder().tableName("wc_out").build(), job);
RunningJob rj = JobClient.runJob(job);
rj.waitForCompletion();
}
}
```

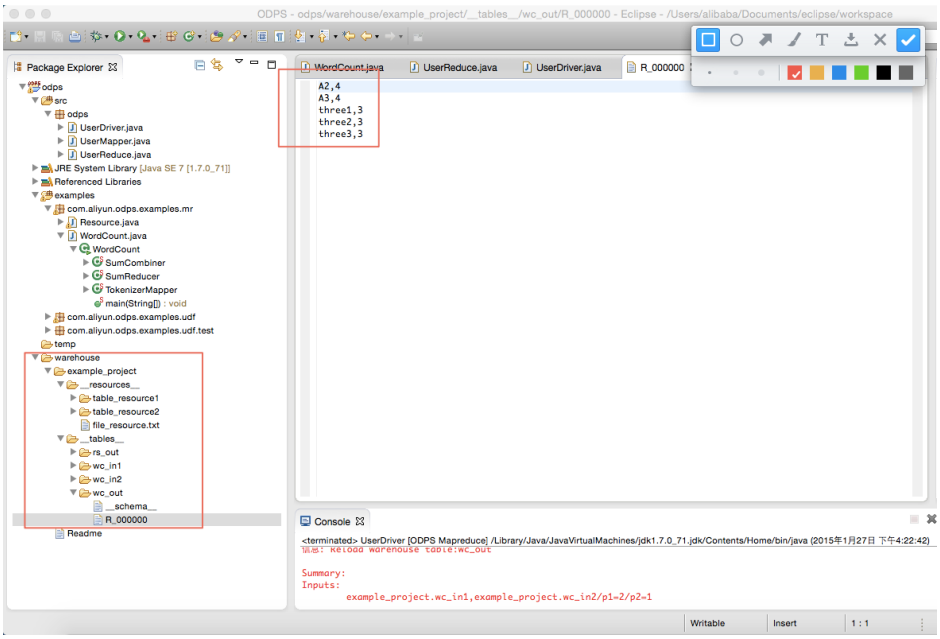
Run MapReduce program. Right-click ‘UserDriver.java’ and select **Run As -> ODPS MapReduce** to pop up the following dialog box:



Select 'example_project' as the MaxCompute Project and click **Finish** to run MapReduce program on the local:



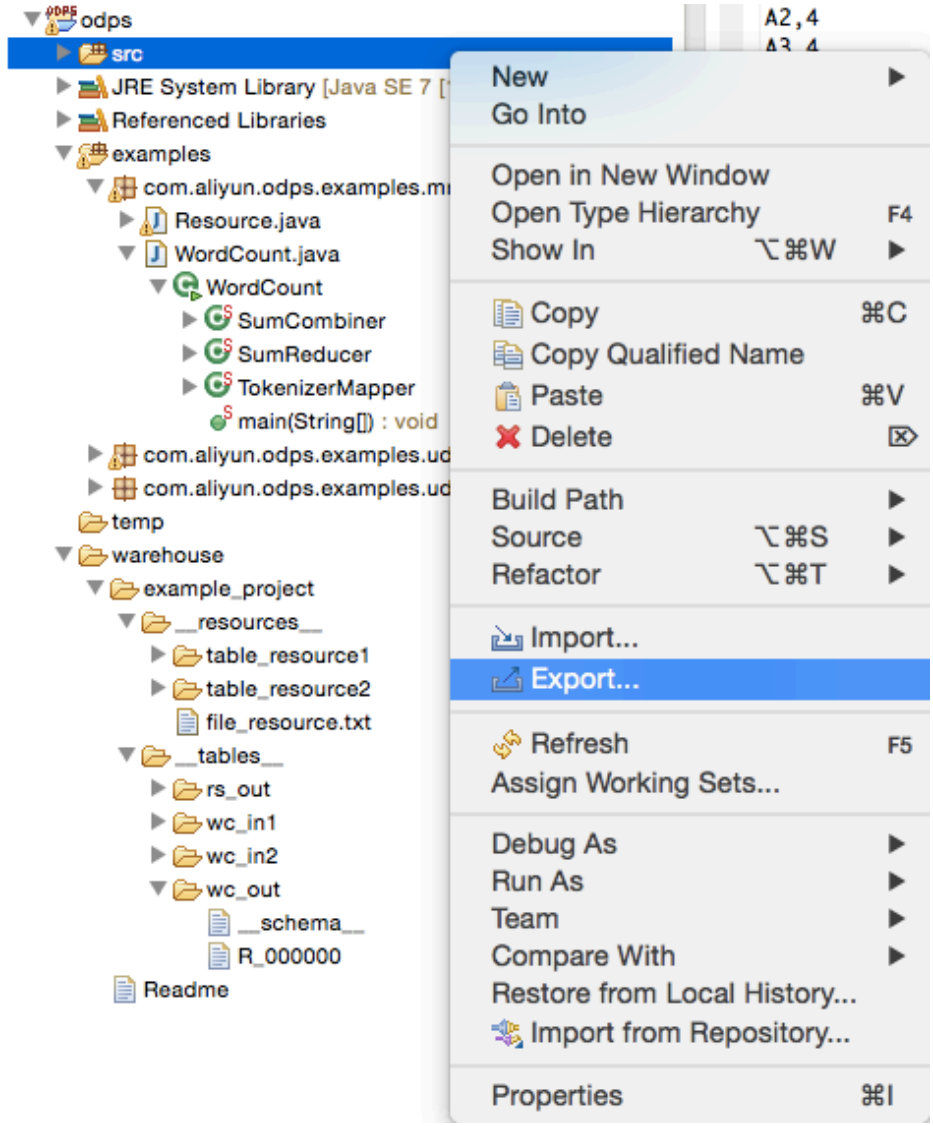
If the information is output as mentioned above, it indicates that local operation runs successfully. The output result is saved in the directory ‘warehouse’. Refresh MaxCompute project:



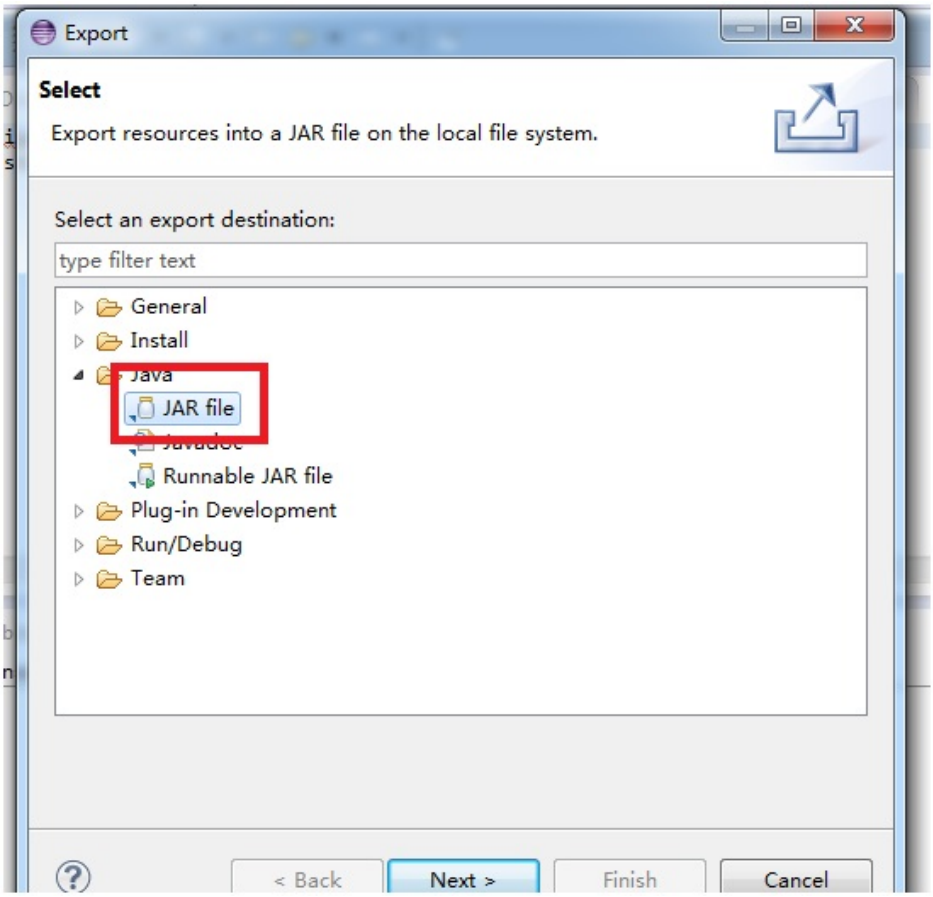
‘wc_out’ is the output directory and ‘R_000000’ is result file. Through local debugging, the result is confrmed to be correct and you can package MapReduce program through Eclipse export function. After it is packaged, upload the jar package to MaxCompute. About how to execute MapReduce in distributed environment, refer to Quick Start.

After the local debugging passed, user can package the codes into jar package through

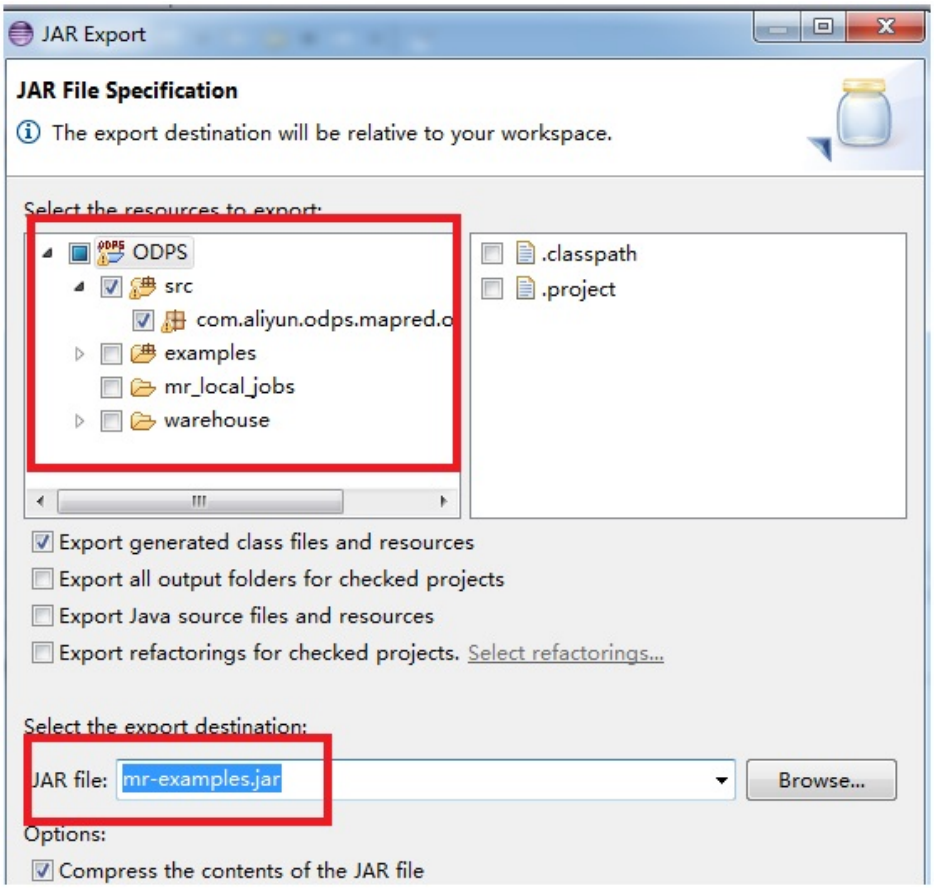
Eclipse Export function, provided for subsequent distributed environment. In this example, we nominate the package 'mr-examples.jar'. Select the directory 'src' and click **Export**:



Select 'Jar File' as an export mode:



You just need to export the package in 'src'. Specify the name of Jar File to be 'mr-examples.jar':



Click **Next** to export the jar file successfully.

If you want to simulate new Project creation in the local, you can create a subdirectory (has same level with example_project) in the directory ‘warehourse’. The directory hierarchy structure is shown as follows:

```
<warehouse>
|___example_project(Project Dirctory)
|___ <__tables__>
| |__table_name1(non-partition table)
| | |___ data(File)
| | |
| | |___ <__schema__> (File)
| |
| |__table_name2(Partition Table)
| |___ partition_name=partition_value(partition directory)
| | |___ data(file)
| |
| |___ <__schema__> (file)
|
|___ <__resources__>
|
|__table_resource_name (table resource)
| |___<__ref__>
|
```

```
|___ file_resource_name ( file resource )
```

'*_schema_* ' Example:

```
Non-partiton table:
project=project_name
table=table_name
columns=col1:BIGINT,col2:DOUBLE,col3:BOOLEAN,col4:DATETIME,col5:STRING
Partition table:
project=project_name
table=table_name
columns=col1:BIGINT,col2:DOUBLE,col3:BOOLEAN,col4:DATETIME,col5:STRING
partitions=col1:BIGINT,col2:DOUBLE,col3:BOOLEAN,col4:DATETIME,col5:STRING
Note:
At present, the following five data formats are supported: bigint,double,boolean,datetime,string, which correspond to the data types in java: -long,double,boolean,java.util.Date,java.lang.String.
```

'data' Example:

```
1,1.1,true,2015-06-04 11:22:42 896,hello world
\N,\N,\N,\N,\N
Note:
The time format is accurate to the millisecond level and all types are represented NULL by '\N'.
```

Note:

- If mapReduce program runs in the local, the default is to search corresponding tables or resources from the directory 'warehouse'. If the tables or resources are not existent, corresponding data will be downloaded from the server and saved in 'warehouse'. Then run MapReduce in the local.
- After running MapReduce is finished, please refresh the directoty 'warehouse' to view generated result.

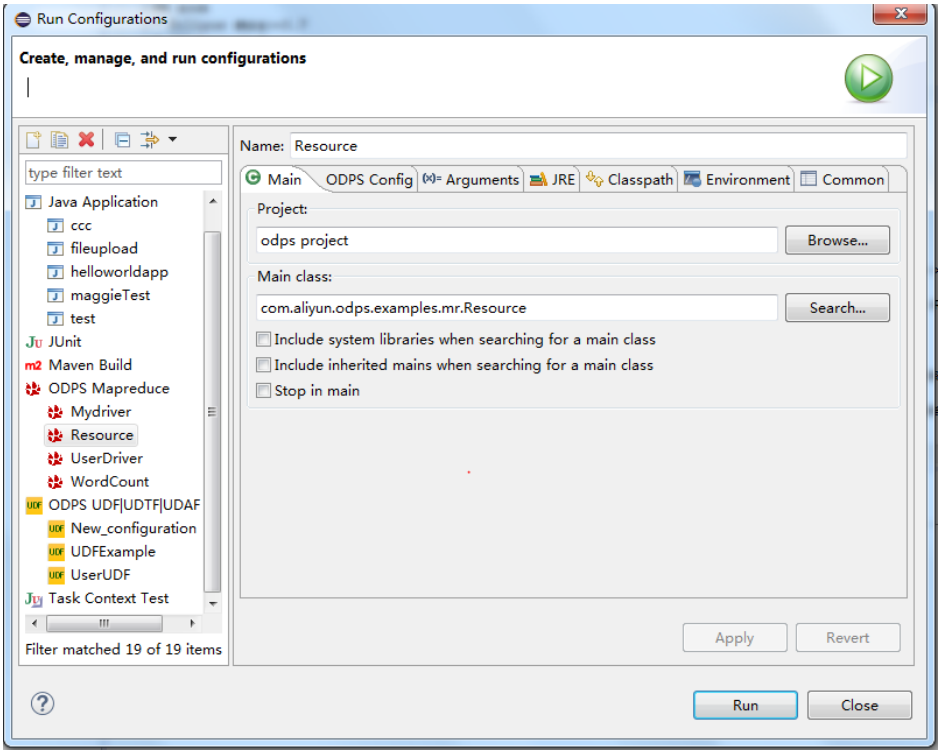
Local Debug UDF Program

This section describes how to develop UDF with the Eclipse plug-in and how to run UDF on local. The preparation and implement process is similar to UDF. You can refer to the example of UDF.

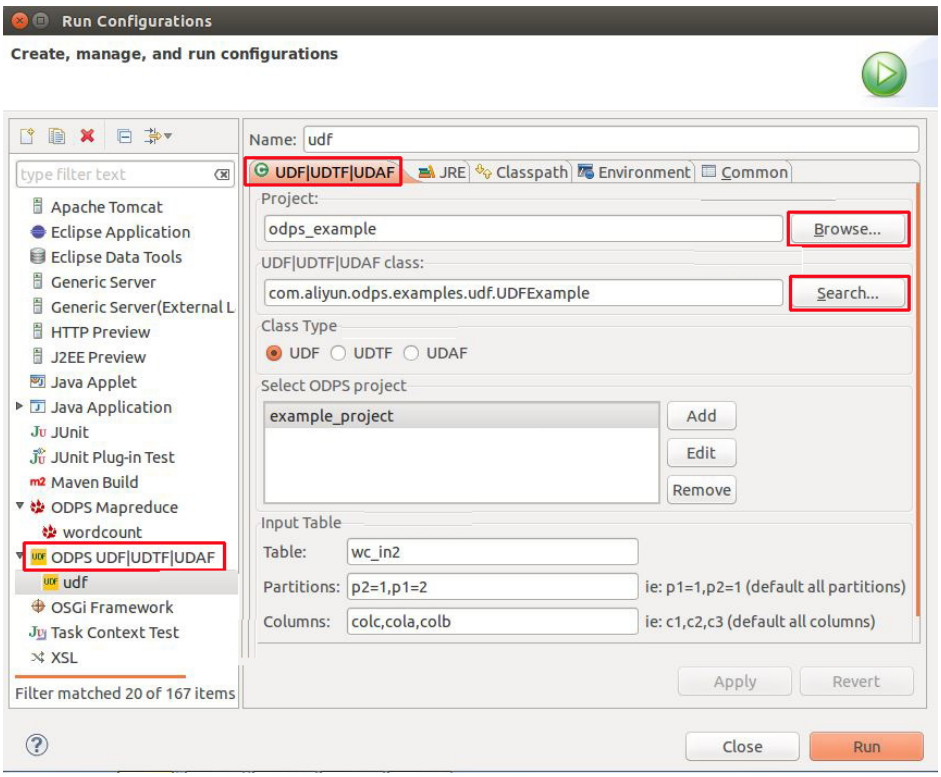
MaxCompute Eclipse plug-in provides two methods to run UDF: Menu Bar and quickly running by rigit-clicking it.

Run UDF through Menu Bar

Select **Run > Run Configurations...** from the menu bar and the following dialog box pops up:



You can create a new Run Configuration. Select the UDF class and type to be executed, select MaxCompute Project and fill in the information of input table. For example:

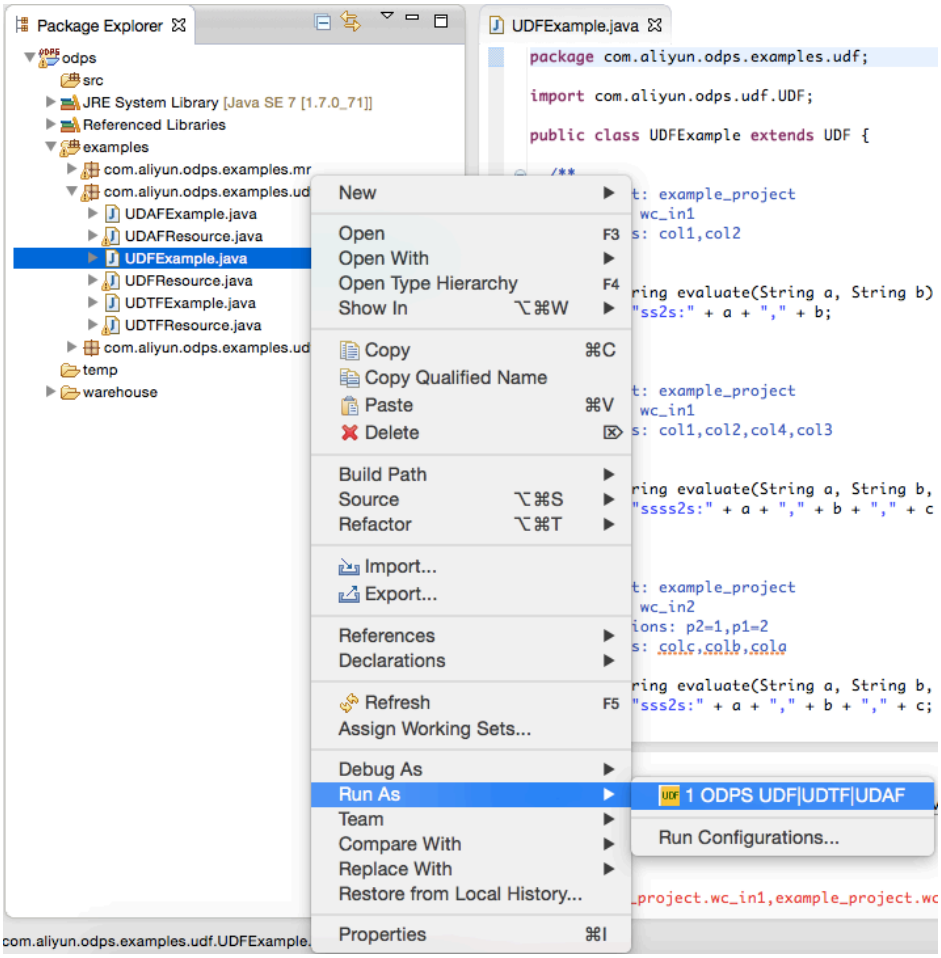


In the configuration mentioned above, “Table” indicates the input table of UDF. “Partitions” indicates the partitions from which the data is read, seperated by commas. “Columns” indicates the columns, which will be considered as the parameters of UDF to be introduced. The columns are seperated by commas.

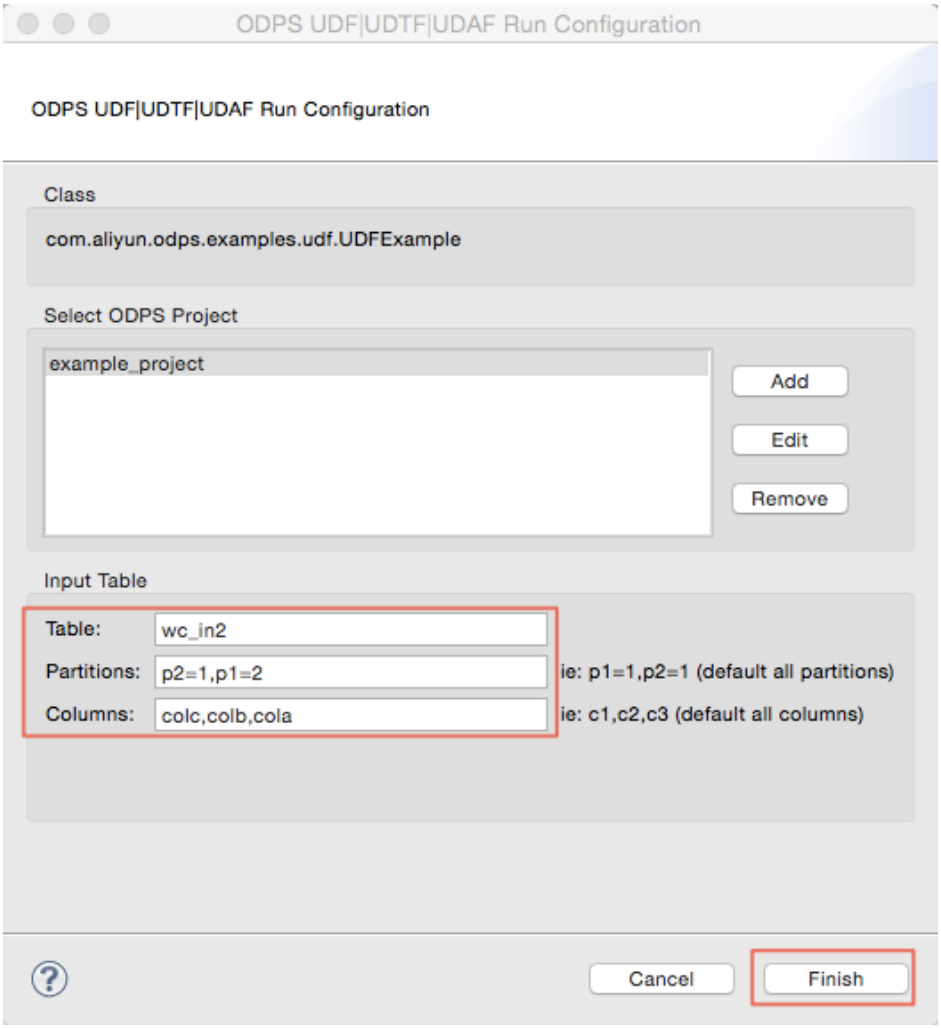
Click **Run** to execute the program and the running result is displayed in the console:

Running Quickly by Right-clicking it

Select a udf.java file (such as: UDFExample.java) and rigit click the mouse. Then select **Run As > Run UDF|UDAF|UDTF**:



The configuration information is shown as follows:



In the configuration mentioned above, "Table" indicates the input table of UDF. "Partitions" indicates the partitions from which the data is read, seperated by commas. "Columns" indicates the columns, which will be considered as the parameters of UDF to be introduced. The columns are seperated by commas.

Click on **Finish** to run UDF and get the output result.

Running Customized UDF Program

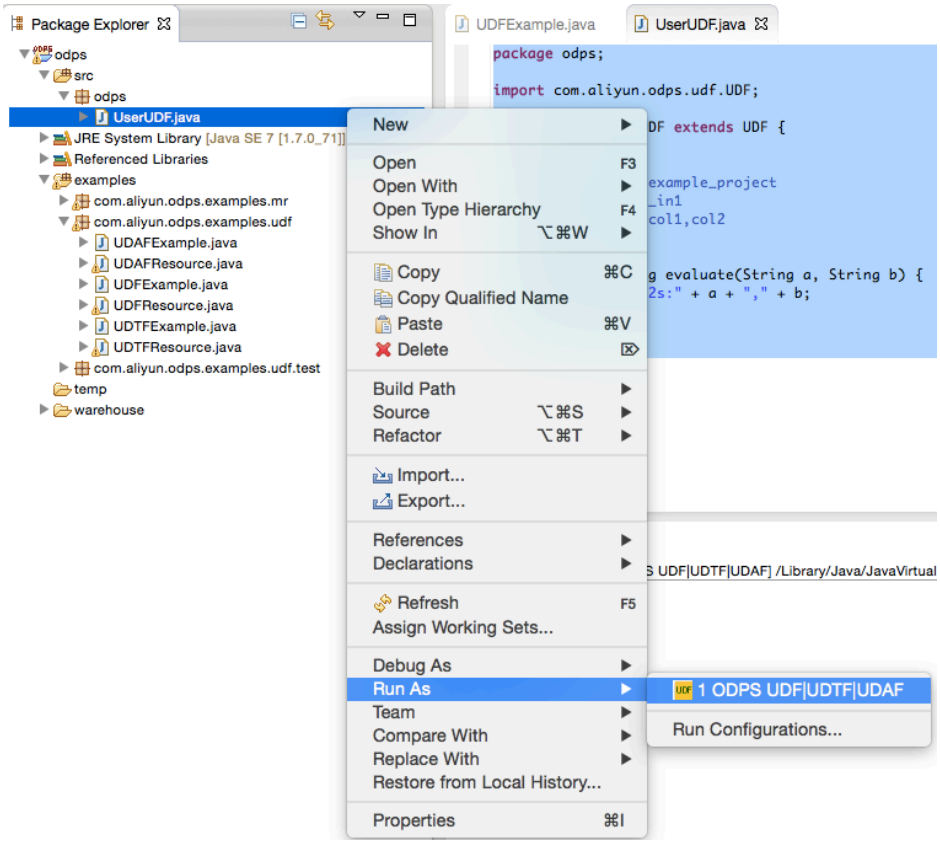
Right click the mouse on a project and select **New >UDF** (or select the menu bar **File > New > UDF**).

Fill in the UDF class name and click "Finish". Generate a Java file in corresponding 'src' directory with the same name as this UDF class. Edit this java file:

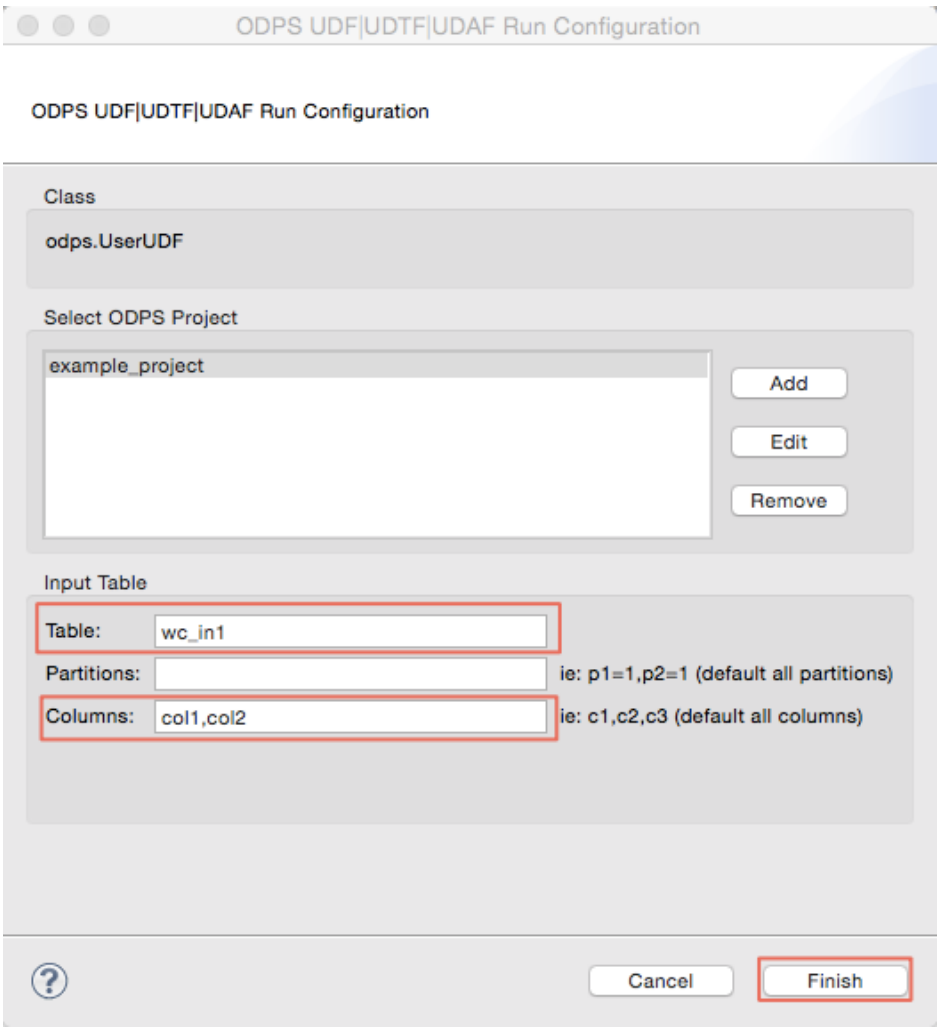
```
package odps;

import com.aliyun.odps.udf.UDF;
```

```
public class UserUDF extends UDF {

/**
* project: example_project
* table: wc_in1
* columns: col1,col2
*
*/
public String evaluate(String a, String b) {
return "ss2s:" + a + "," + b;
}

}
```

Right click this java file (such as UserUDF.java) and select **Run As -> ODPS UDF|UDTF|UDAF**:



Configure the following dialog box:



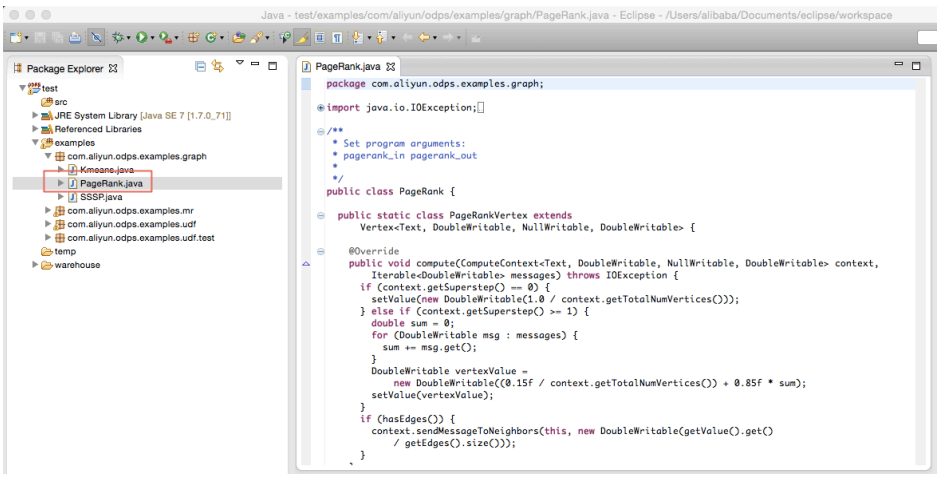
Click **finish** to get the result:

```
ss2s:A1,A2
ss2s:A1,A2
ss2s:A1,A2
ss2s:A1,A2
```

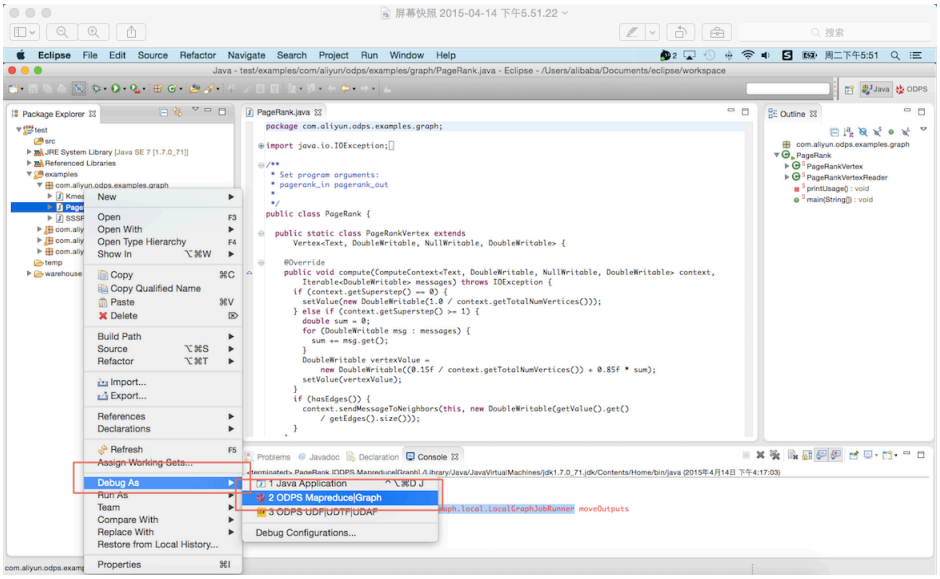
Only the operation instance of UDF is described in this section, and the way of UDTF operating is quite similar with the UDF, therefore, no special descriptions about this situation.

After creating a MaxCompute project, user can write Graph program and complete the local debugging accoring the following steps.In this example, we select “PageRank.java” provided by the plug-in to complete the debugging.

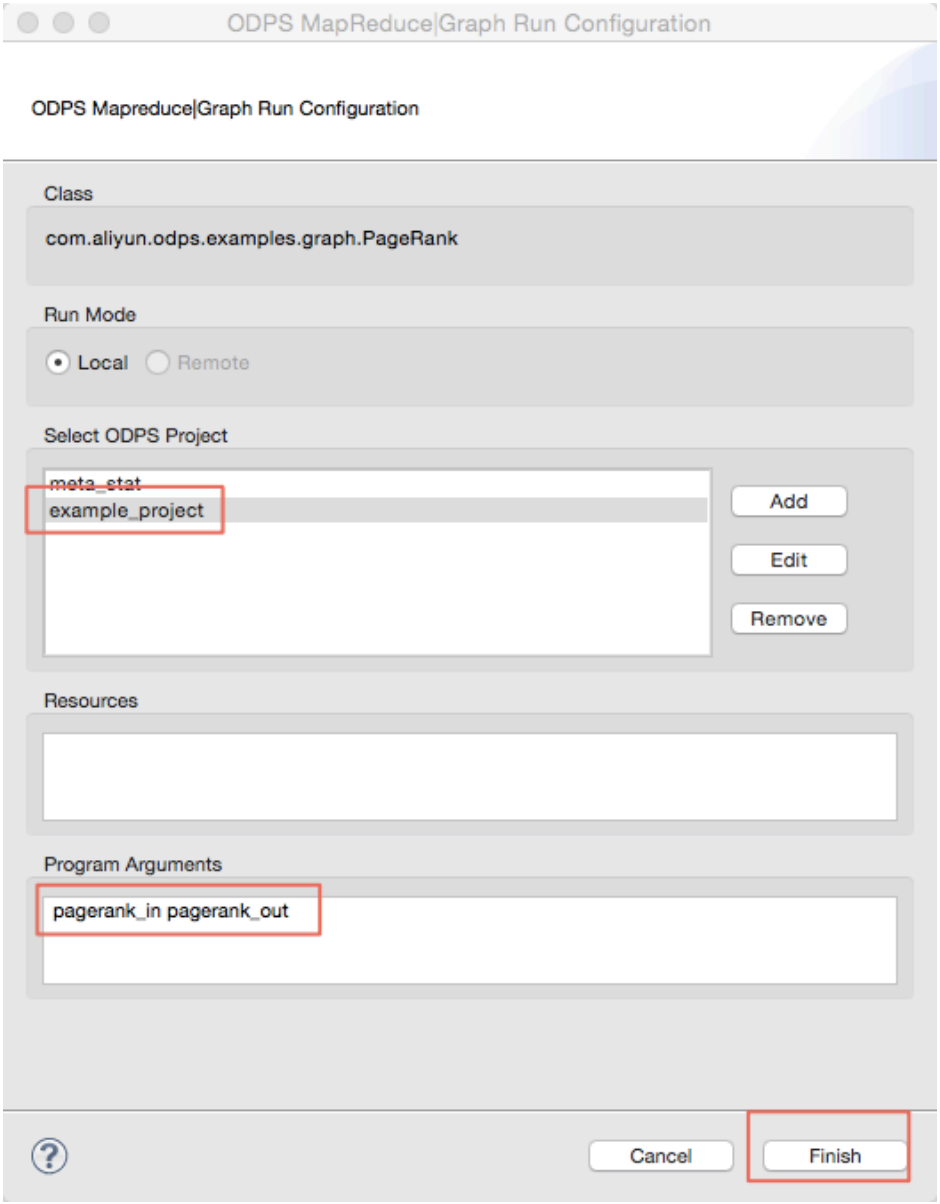
Select **PageRank.java** in **examples**:



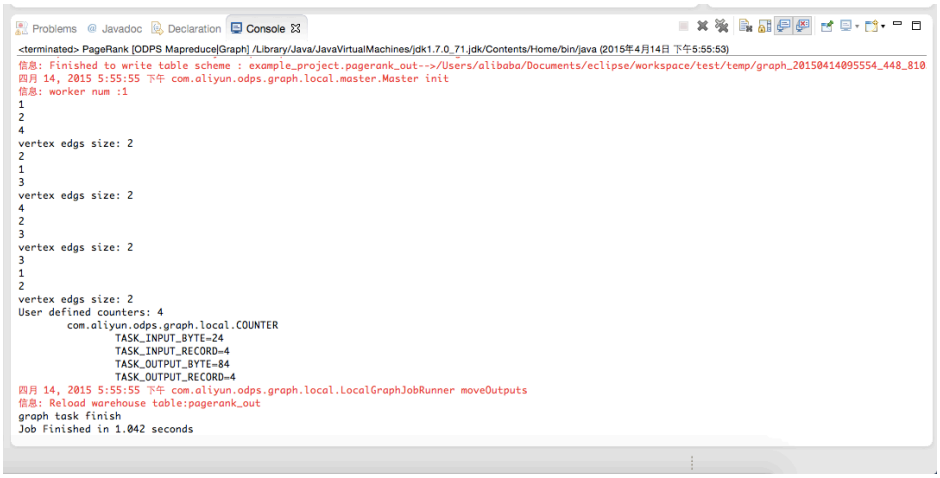
Right click the mouse and select **Debug As -> ODPS MapReduce|Graph**:



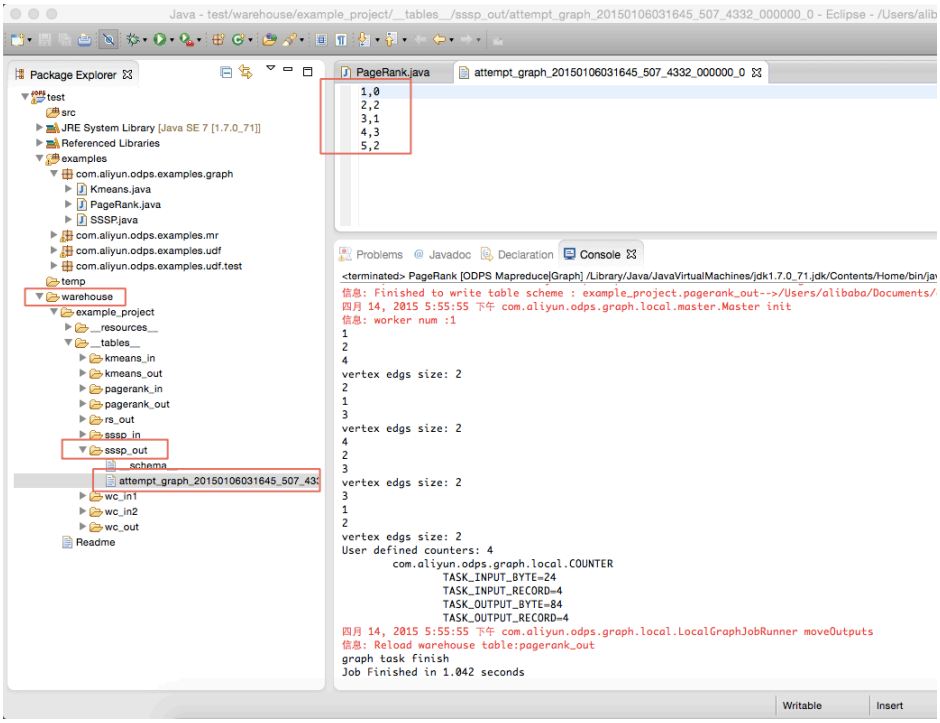
The dialog box appears and configure it as follows:



View the running result:



You can view the computing result on the local:



Afther the debugging passed, you can package the program and upload it to MaxCompute as a Jar resource. Then submit Graph job.

Note:

- For the package process, refer to MapReduce Eclipse Plug-in Introduction.
- For the structure introduction of local result, refer to MapReduce Eclipse Plug-in Introduction.
- For the detaied introcution of uploading Jar resource, refer to ‘Add Resource’ in Basic Introduction.
- For submitting the Graph job, refer to Graph Function.

- SDK Downloads: maven user can search “odps-sdk” from Maven library get MaxCompute Java SDK.
- MaxCompute console: click here.
- Eclipse plugin: click here.
- Intelij plugin: click here.