

4.2.1.3 Byzer-SQL 和大模型的整合

[安装插件](#)

[启动大模型](#)

[连接大模型](#)

[Byzer-SQL 函数中的 prompt 管理](#)

[Byzer-SQL 和大模型集成的意义](#)

其实你肯定觉得上面的语法枯燥而乏味。实际上我们可以通过和大模型的整合，让编写 Byzer-SQL 变得更加。在这篇中，我们重点介绍下如何在 Byzer-SQL 中连接大模型，使用大模型。

安装插件

假设我们前面的安装目录是叫 BYZER_HOME, 进入 \$BYZER_HOME/plugin 目录，然后打开网`<https://download.byzer.org/byzer-extensions/nightly-build>` 找到最新byzer-llm插件，当前写书时的最新版本是： byzer-llm-3.3_2.12-0.1.9.jar ,下载下来，然后拷贝到 plugin 目录。或者也可以用如下命令直接完成：

```
1 wget https://download.byzer.org/byzer-extensions/nightly-build/byzer-llm-3.3_2.12-0.1.9.jar
```

按相同的方式下载：https://download.byzer.org/byzer-extensions/nightly-build/byzer-yaml-visualization-3.3_2.12-0.1.0-SNAPSHOT.jar 这个插件主要是支持可视化的。此外我们还提供诸如各种对象存储的访问插件，可以访问市面上主流的云对象存储，更多可以查看官方文档。你也可以查看我们的插件项目：<https://github.com/byzer-org/byzer-extension>。

最后，修改 conf/byzer.properties.override文件，找到这么一行配置：

```
1 streaming.plugin.clznames=tech.mlsq.plugins.ds.MLSQExcelApp,tech.mlsq.plugins.assert.app.MLSQLAssert,tech.mlsq.plugins.shell.app.MLSQLShell,tech.mlsq.plugins.mllib.app.MLSQLMllib
```

添加一个 `tech.mlsq.plugins.llm.LLMApp` 和 `tech.mlsq.plugins.visualization.ByzerVisualizationApp` ,修改后的样子:

```
1 streaming.plugin.clznames=tech.mlsq.plugins.llm.LLMApp,tech.mlsq.plugins.visualization.ByzerVisualizationApp,tech.mlsq.plugins.ds.MLSQExcelApp,tech.mlsq.plugins.assert.app.MLSQLAssert,tech.mlsq.plugins.shell.app.MLSQLShell,tech.mlsq.plugins.mllib.app.MLSQLMllib
```

现在需要重启下引擎(在 BYZER_HOME 根目录下), 执行如下指令接口重启:

```
1 ./bin/byzer.sh restart
```

如果你是 MacOS, 并且在后续使用过程中遇到如下问题:

```

1  24/08/01 12:12:09 INFO DriverLogServer: [owner] [hello] [groupId] [99a2f
245-4b91-4510-b7f5-c260f1f2f3e1] __MMMMMM__ objc[1440]: +[__NSTimeZone in
italize] may have been in progress in another thread when fork() was cal
led.
2  24/08/01 12:12:09 INFO DriverLogServer: [owner] [hello] [groupId] [99a2f
245-4b91-4510-b7f5-c260f1f2f3e1] __MMMMMM__ objc[1440]: +[__NSTimeZone in
italize] may have been in progress in another thread when fork() was cal
led. We cannot safely call it or ignore it in the fork() child process. C
rashing instead. Set a breakpoint on objc_initializeAfterForkError to deb
ug.
3  24/08/01 12:12:09 ERROR Executor: Exception in task 0.0 in stage 40.0 (T
ID 412)
4  org.apache.spark.SparkException: Failed to execute user defined function
(Ray$$Lambda$3367/1609353118: (array<string>) => array<string>)
5      at org.apache.spark.sql.errors.QueryExecutionErrors$.failedExecuteUse
rDefinedFunctionError(QueryExecutionErrors.scala:177) ~[spark-catalyst_2.
12-3.3.0.jar:3.3.0]
6      at org.apache.spark.sql.errors.QueryExecutionErrors.failedExecuteUser
DefinedFunctionError(QueryExecutionErrors.scala) ~[spark-catalyst_2.12-3.
3.0.jar:3.3.0]
7      at org.apache.spark.sql.catalyst.expressions.GeneratedClass$Generated
IteratorForCodegenStage1.project_doConsume_0$(Unknown Source) ~[?:?]
8      at org.apache.spark.sql.catalyst.expressions.GeneratedClass$Generated
IteratorForCodegenStage1.processNext(Unknown Source) ~[?:?]
9      at org.apache.spark.sql.execution.BufferedRowIterator.hasNext(Buffered
RowIterator.java:43) ~[spark-sql_2.12-3.3.0.jar:3.3.0]
10     at org.apache.spark.sql.execution.WholeStageCodegenExec$$anon$1.hasNe
xt(WholeStageCodegenExec.scala:760) ~[spark-sql_2.12-3.3.0.jar:3.3.0]
11     at scala.collection.Iterator$$anon$10.hasNext(Iterator.scala:460) ~[s
cala-library-2.12.15.jar:~]
12     at scala.collection.Iterator$$anon$10.hasNext(Iterator.scala:460) ~[s
cala-library-2.12.15.jar:~]
13     at org.apache.spark.sql.Dataset$$anon$1.hasNext(Dataset.scala:3589) ~
[spark-sql_2.12-3.3.0.jar:3.3.0]
14     at scala.collection.Iterator$$anon$10.hasNext(Iterator.scala:460) ~[s
cala-library-2.12.15.jar:~]
15     at org.apache.spark.sql.catalyst.expressions.GeneratedClass$Generated
IteratorForCodegenStage2.processNext(Unknown Source) ~[?:?]
16     at org.apache.spark.sql.execution.BufferedRowIterator.hasNext(Buffered
RowIterator.java:43) ~[spark-sql_2.12-3.3.0.jar:3.3.0]
17     at org.apache.spark.sql.execution.WholeStageCodegenExec$$anon$1.hasNe
xt(WholeStageCodegenExec.scala:760) ~[spark-sql_2.12-3.3.0.jar:3.3.0]
18     at org.apache.spark.sql.execution.SparkPlan.$anonfun$getByteArrayRdd
$1(SparkPlan.scala:364) ~[spark-sql_2.12-3.3.0.jar:3.3.0]
19     at org.apache.spark.rdd.RDD.$anonfun$mapPartitionsInternal$2(RDD.scal
a:890) ~[spark-core_2.12-3.3.0.jar:3.3.0]
20

```

```

21     at org.apache.spark.rdd.RDD.$anonfun$mapPartitionsInternal$2$adapted(
RDD.scala:890) ~[spark-core_2.12-3.3.0.jar:3.3.0]
22     at org.apache.spark.rdd.MapPartitionsRDD.compute(MapPartitionsRDD.sca
la:52) ~[spark-core_2.12-3.3.0.jar:3.3.0]
23     at org.apache.spark.rdd.RDD.computeOrReadCheckpoint(RDD.scala:365) ~[
spark-core_2.12-3.3.0.jar:3.3.0]
24     at org.apache.spark.rdd.RDD.iterator(RDD.scala:329) ~[spark-core_2.12
-3.3.0.jar:3.3.0]
25     at org.apache.spark.scheduler.ResultTask.runTask(ResultTask.scala:90)
~[spark-core_2.12-3.3.0.jar:3.3.0]
26     at org.apache.spark.scheduler.Task.run(Task.scala:136) ~[spark-core_2
.12-3.3.0.jar:3.3.0]
27     at org.apache.spark.executor.Executor$TaskRunner.$anonfun$run$3(Execu
tor.scala:548) ~[spark-core_2.12-3.3.0.jar:3.3.0]
28     at org.apache.spark.util.Utils$.tryWithSafeFinally(Utils.scala:1504)
~[spark-core_2.12-3.3.0.jar:3.3.0]
29     at org.apache.spark.executor.Executor$TaskRunner.run(Executor.scala:5
51) ~[spark-core_2.12-3.3.0.jar:3.3.0]
30     at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecut
or.java:1149) ~[?:1.8.0_151]
31     at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecu
tor.java:624) ~[?:1.8.0_151]
32     at java.lang.Thread.run(Thread.java:748) ~[?:1.8.0_151]
33     Caused by: org.apache.spark.SparkException: Python worker exited unexpect
edly (crashed)
34     at tech.mlsql.arrow.python.runner.BasePythonRunner$ReaderIterator$$an
onfun$1.applyOrElse(PythonRunner.scala:364) ~[byzer-lang-3.3.0-2.12-2.3.9
.jar:?]
35     at tech.mlsql.arrow.python.runner.BasePythonRunner$ReaderIterator$$an
onfun$1.applyOrElse(PythonRunner.scala:353) ~[byzer-lang-3.3.0-2.12-2.3.9
.jar:?]
36     at scala.runtime.AbstractPartialFunction.apply(AbstractPartialFunction
n.scala:38) ~[scala-library-2.12.15.jar:~]
37     at tech.mlsql.arrow.python.runner.ArrowPythonRunner$$anon$2.read(Arro
wPythonRunner.scala:164) ~[byzer-lang-3.3.0-2.12-2.3.9.jar:~]
38     at tech.mlsql.arrow.python.runner.ArrowPythonRunner$$anon$2.read(Arro
wPythonRunner.scala:102) ~[byzer-lang-3.3.0-2.12-2.3.9.jar:~]
39     at tech.mlsql.arrow.python.runner.BasePythonRunner$ReaderIterator.has
Next(PythonRunner.scala:293) ~[byzer-lang-3.3.0-2.12-2.3.9.jar:~]
40     at tech.mlsql.arrow.python.runner.InterruptibleIterator.hasNext(Pytho
nRunner.scala:415) ~[byzer-lang-3.3.0-2.12-2.3.9.jar:~]
41     at scala.collection.Iterator$$anon$11.hasNext(Iterator.scala:491) ~[s
cala-library-2.12.15.jar:~]
42     at scala.collection.Iterator$$anon$10.hasNext(Iterator.scala:460) ~[s
cala-library-2.12.15.jar:~]
43     at scala.collection.Iterator.foreach(Iterator.scala:943) ~[scala-libr
ary-2.12.15.jar:~]

```

```

44     at scala.collection.Iterator.foreach$(Iterator.scala:943) ~[scala-library-2.12.15.jar:?]
45     at scala.collection.AbstractIterator.foreach(Iterator.scala:1431) ~[scala-library-2.12.15.jar:?]
46     at scala.collection.generic.Growable.$plus$plus$eq(Growable.scala:62) ~[scala-library-2.12.15.jar:?]
47     at scala.collection.generic.Growable.$plus$plus$eq$(Growable.scala:53) ~[scala-library-2.12.15.jar:?]
48     at scala.collection.mutable.ListBuffer.$plus$plus$eq(ListBuffer.scala:184) ~[scala-library-2.12.15.jar:?]
49     at scala.collection.mutable.ListBuffer.$plus$plus$eq(ListBuffer.scala:47) ~[scala-library-2.12.15.jar:?]
50     at scala.collection.TraversableOnce.to(TraversableOnce.scala:366) ~[scala-library-2.12.15.jar:?]
51     at scala.collection.TraversableOnce.to$(TraversableOnce.scala:364) ~[scala-library-2.12.15.jar:?]
52     at scala.collection.AbstractIterator.to(Iterator.scala:1431) ~[scala-library-2.12.15.jar:?]
53     at scala.collection.TraversableOnce.toList(TraversableOnce.scala:350) ~[scala-library-2.12.15.jar:?]
54     at scala.collection.TraversableOnce.toList$(TraversableOnce.scala:350) ~[scala-library-2.12.15.jar:?]
55     at scala.collection.AbstractIterator.toList(Iterator.scala:1431) ~[scala-library-2.12.15.jar:?]
56     at tech.mlsq.ets.Ray$.executePythonCode(Ray.scala:430) ~[byzer-lang-3.3.0-2.12-2.3.9.jar:?]
57     at tech.mlsq.ets.Ray.$anonfun$predict$7(Ray.scala:372) ~[byzer-lang-3.3.0-2.12-2.3.9.jar:?]
58     ... 25 more
59 Caused by: java.io.EOFException
60     at java.io.DataInputStream.readInt(DataInputStream.java:392) ~[?:1.8.0_151]
61     at tech.mlsq.arrow.python.runner.ArrowPythonRunner$$anon$2.read(ArrowPythonRunner.scala:133) ~[byzer-lang-3.3.0-2.12-2.3.9.jar:?]
62     at tech.mlsq.arrow.python.runner.ArrowPythonRunner$$anon$2.read(ArrowPythonRunner.scala:102) ~[byzer-lang-3.3.0-2.12-2.3.9.jar:?]
63     at tech.mlsq.arrow.python.runner.BasePythonRunner$ReaderIterator.hasNext(PythonRunner.scala:293) ~[byzer-lang-3.3.0-2.12-2.3.9.jar:?]
64     at tech.mlsq.arrow.python.runner.InterruptibleIterator.hasNext(PythonRunner.scala:415) ~[byzer-lang-3.3.0-2.12-2.3.9.jar:?]
65     at scala.collection.Iterator$$anon$11.hasNext(Iterator.scala:491) ~[scala-library-2.12.15.jar:?]
66     at scala.collection.Iterator$$anon$10.hasNext(Iterator.scala:460) ~[scala-library-2.12.15.jar:?]
67     at scala.collection.Iterator.foreach(Iterator.scala:943) ~[scala-library-2.12.15.jar:?]
68     at scala.collection.Iterator.foreach$(Iterator.scala:943) ~[scala-library-2.12.15.jar:?]

```

```

69     at scala.collection.AbstractIterator.foreach(Iterator.scala:1431) ~[s
cala-library-2.12.15.jar:?]
70     at scala.collection.generic.Growable.$plus$plus$eq(Growable.scala:62)
~[scala-library-2.12.15.jar:?]
71     at scala.collection.generic.Growable.$plus$plus$eq$(Growable.scala:53)
~[scala-library-2.12.15.jar:?]
72     at scala.collection.mutable.ListBuffer.$plus$plus$eq(ListBuffer.scala
:184) ~[scala-library-2.12.15.jar:?]
73     at scala.collection.mutable.ListBuffer.$plus$plus$eq(ListBuffer.scala
:47) ~[scala-library-2.12.15.jar:?]
74     at scala.collection.TraversableOnce.to(TraversableOnce.scala:366) ~[s
cala-library-2.12.15.jar:?]
75     at scala.collection.TraversableOnce.to$(TraversableOnce.scala:364) ~[
scala-library-2.12.15.jar:?]
76     at scala.collection.AbstractIterator.to(Iterator.scala:1431) ~[scala-
library-2.12.15.jar:?]
77     at scala.collection.TraversableOnce.toList(TraversableOnce.scala:350)
~[scala-library-2.12.15.jar:?]
78     at scala.collection.TraversableOnce.toList$(TraversableOnce.scala:350)
~[scala-library-2.12.15.jar:?]
79     at scala.collection.AbstractIterator.toList(Iterator.scala:1431) ~[sc
ala-library-2.12.15.jar:?]
80     at tech.mlsq.ets.Ray$.executePythonCode(Ray.scala:430) ~[byzer-lang-
3.3.0-2.12-2.3.9.jar:?]
81     at tech.mlsq.ets.Ray.$anonfun$predict$7(Ray.scala:372) ~[byzer-lang-
3.3.0-2.12-2.3.9.jar:?]
82     ... 25 more
24/08/01 12:12:09 WARN TaskSetManager: Lost task 0.0 in stage 40.0 (TID
412) (192.168.49.88 executor driver): org.apache.spark.SparkException: Fa
83 iled to execute user defined function (Ray$$Lambda$3367/1609353118: (arra
y<string>) => array<string>)
84     at org.apache.spark.sql.errors.QueryExecutionErrors$.failedExecuteUse
rDefinedFunctionError(QueryExecutionErrors.scala:177)
85     at org.apache.spark.sql.errors.QueryExecutionErrors.failedExecuteUser
DefinedFunctionError(QueryExecutionErrors.scala)
86     at org.apache.spark.sql.catalyst.expressions.GeneratedClass$Generated
IteratorForCodegenStage1.project_doConsume_0$(Unknown Source)
87     at org.apache.spark.sql.catalyst.expressions.GeneratedClass$Generated
IteratorForCodegenStage1.processNext(Unknown Source)
88     at org.apache.spark.sql.execution.BufferedRowIterator.hasNext(Buffered
dRowIterator.java:43)
89     at org.apache.spark.sql.execution.WholeStageCodegenExec$$anon$1.hasNe
xt(WholeStageCodegenExec.scala:760)
90     at scala.collection.Iterator$$anon$10.hasNext(Iterator.scala:460)
91     at scala.collection.Iterator$$anon$10.hasNext(Iterator.scala:460)
92     at org.apache.spark.sql.Dataset$$anon$1.hasNext(Dataset.scala:3589)
93     at scala.collection.Iterator$$anon$10.hasNext(Iterator.scala:460)
94

```



```

105     at org.apache.spark.sql.catalyst.expressions.GeneratedClass$Generated
106     IteratorForCodegenStage2.processNext(Unknown Source)
107     at org.apache.spark.sql.execution.BufferedRowIterator.hasNext(Buffered
108     dRowIterator.java:43)
109     at org.apache.spark.sql.execution.WholeStageCodegenExec$$anon$1.hasNe
110     xt(WholeStageCodegenExec.scala:760)
111     at org.apache.spark.sql.execution.SparkPlan.$anonfun$getByteArrayRdd
112     $1(SparkPlan.scala:364)
113     at org.apache.spark.rdd.RDD.$anonfun$mapPartitionsInternal$2(RDD.scala
114     :890)
115     at org.apache.spark.rdd.RDD.$anonfun$mapPartitionsInternal$2$adapted(
116     RDD.scala:890)
117     at org.apache.spark.rdd.MapPartitionsRDD.compute(MapPartitionsRDD.sca
118     la:52)
119     at org.apache.spark.rdd.RDD.computeOrReadCheckpoint(RDD.scala:365)
120     at org.apache.spark.rdd.RDD.iterator(RDD.scala:329)
121     at org.apache.spark.scheduler.ResultTask.runTask(ResultTask.scala:90)
122     at org.apache.spark.scheduler.Task.run(Task.scala:136)
123     at org.apache.spark.executor.Executor$TaskRunner.$anonfun$run$3(Execu
124     tor.scala:548)
125     at org.apache.spark.util.Utils$.tryWithSafeFinally(Utils.scala:1504)
126     at org.apache.spark.executor.Executor$TaskRunner.run(Executor.scala:5
127     51)
128     at java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecut
129     or.java:1149)
130     at java.util.concurrent.ThreadPoolExecutor$Worker.run(ThreadPoolExecu
131     tor.java:624)
132     at java.lang.Thread.run(Thread.java:748)
133     Caused by: org.apache.spark.SparkException: Python worker exited unexpect
134     edly (crashed)
135     at tech.mlsql.arrow.python.runner.BasePythonRunner$ReaderIterator$$an
136     onfun$1.applyOrElse(PythonRunner.scala:364)
137     at tech.mlsql.arrow.python.runner.BasePythonRunner$ReaderIterator$$an
138     onfun$1.applyOrElse(PythonRunner.scala:353)
139     at scala.runtime.AbstractPartialFunction.apply(AbstractPartialFunction
140     n.scala:38)
141     at tech.mlsql.arrow.python.runner.ArrowPythonRunner$$anon$2.read(Arro
142     wPythonRunner.scala:164)
143     at tech.mlsql.arrow.python.runner.ArrowPythonRunner$$anon$2.read(Arro
144     wPythonRunner.scala:102)
145     at tech.mlsql.arrow.python.runner.BasePythonRunner$ReaderIterator.has
146     Next(PythonRunner.scala:293)
147     at tech.mlsql.arrow.python.runner.InterruptibleIterator.hasNext(Pytho
148     nRunner.scala:415)
149     at scala.collection.Iterator$$anon$11.hasNext(Iterator.scala:491)
150     at scala.collection.Iterator$$anon$10.hasNext(Iterator.scala:460)
151     at scala.collection.Iterator.foreach(Iterator.scala:943)
152     at scala.collection.Iterator.foreach$(Iterator.scala:943)

```

```

125     at scala.collection.AbstractIterator.foreach(Iterator.scala:1431)
    at scala.collection.generic.Growable.$plus$plus$eq(Growable.scala:62)
    at scala.collection.generic.Growable.$plus$plus$eq$(Growable.scala:53)
126 )
    at scala.collection.mutable.ListBuffer.$plus$plus$eq(ListBuffer.scala
127 :184)
    at scala.collection.mutable.ListBuffer.$plus$plus$eq(ListBuffer.scala
128 :47)
    at scala.collection.TraversableOnce.to(TraversableOnce.scala:366)
129
130 at scala.collection.TraversableOnce.toList(TraversableOnce.scala:350)
131 at scala.collection.TraversableOnce.toList$(TraversableOnce.scala:350)
132 )
    at scala.collection.AbstractIterator.toList(Iterator.scala:1431)
133
134 at tech.mlsq.ets.Ray$.executePythonCode(Ray.scala:430)
135 at tech.mlsq.ets.Ray.$anonfun$predict$7(Ray.scala:372)
136
137 ... 25 more
138
139 Caused by: java.io.EOFException
    at java.io.DataInputStream.readInt(DataInputStream.java:392)
140
141 at tech.mlsq.arrow.python.runner.ArrowPythonRunner$$anon$2.read(Arro
wPythonRunner.scala:133)
142
    ... 45 more

24/08/01 12:12:09 ERROR TaskSetManager: Task 0 in stage 40.0 failed 1 ti
mes; aborting job

```

可以添加一个环境变量来启动即可解决问题：

```
1 OBJC_DISABLE_INITIALIZE_FORK_SAFETY=YES ./bin/byzer.sh restart
```

启动大模型

这里我们在命令行里启动一个大模型，假设部署的是一个 SaaS 模型， deepseek-chat:

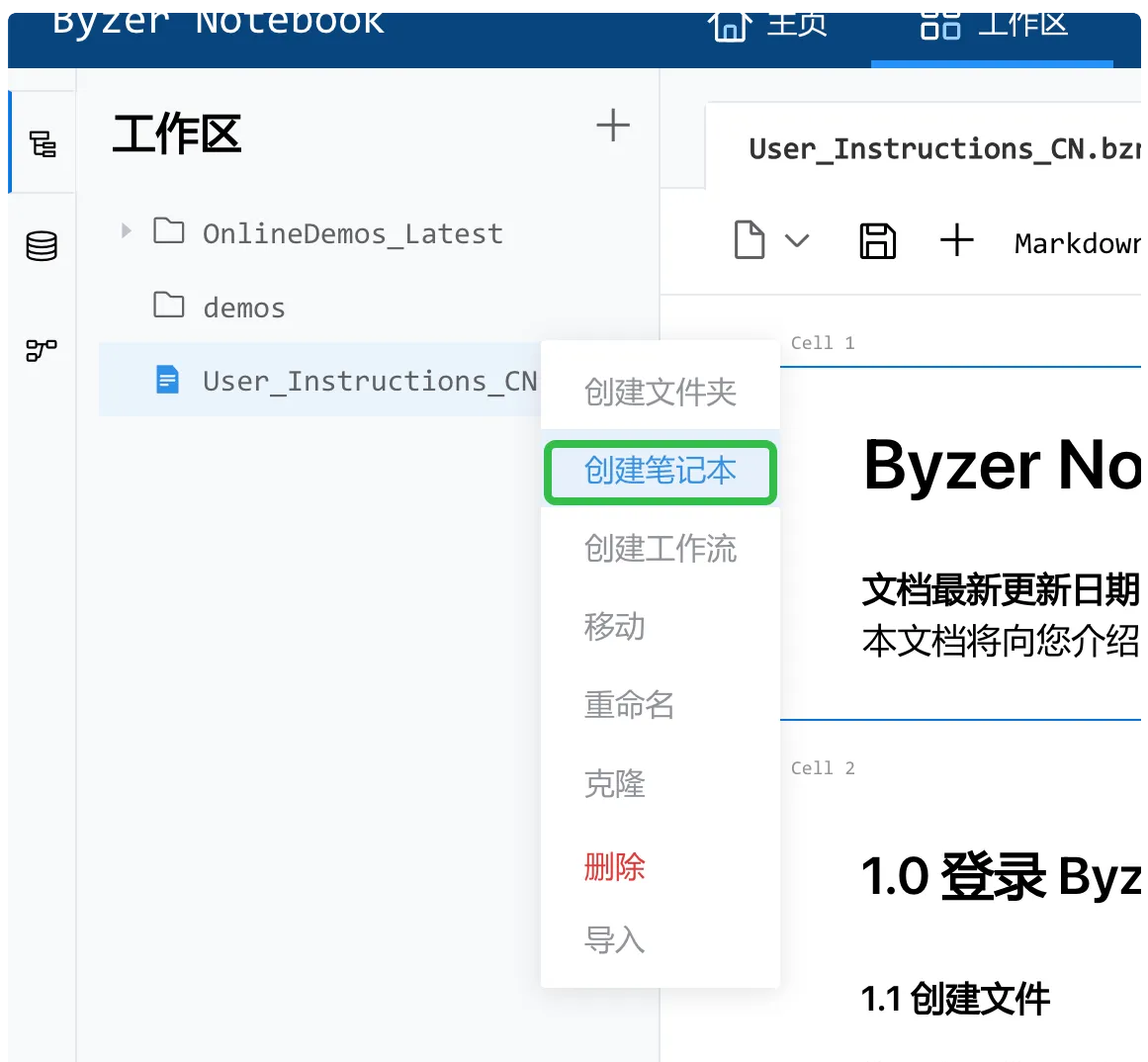

```

1 byzerllm deploy --pretrained_model_type saas/openai \
2 --cpus_per_worker 0.001 \
3 --gpus_per_worker 0 \
4 --worker_concurrency 10 \
5 --num_workers 1 \
6 --infer_params saas.base_url="https://api.deepseek.com/v1" saas.api_key=${MODEL_DEEPSEEK_TOKEN} saas.model=deepseek-chat \
7 --model deepseek_chat

```

连接大模型

在 Byzer-SQL 控制台，右键点击 创建笔记本 来新建一个 Notebook:



然后弹出一个框，输入名字，点击“创建”即可。

创建笔记本

* 笔记本名称

demo

取消

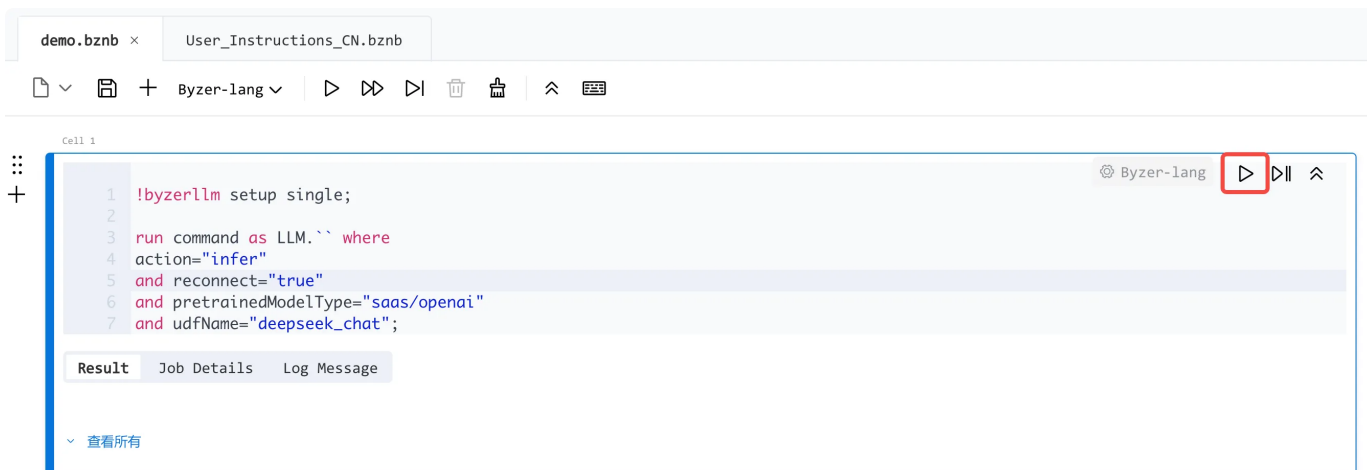
创建

在新创建的笔记本里，输入下面代码：

```
1  !byzerllm setup single;
2
3  run command as LLM.`` where
4  action="infer"
5  and reconnect="true"
6  and pretrainedModelType="saas/openai"
7  and udfName="deepseek_chat";
```

这段话表示，我们在运行 Byzer-SQL 一个叫 LLM 的模块，该模块有四个参数，分别是：

1. action 执行的操作类型，因为我们是推理，填写 "infer" 即可。
2. reconnect, 是否连接已有的模型实例，选择 "true"（在 Byzer-SQL 中万物皆字符串）。
3. pretrainedModelType 连接的模型类型，这里是 "saas/openai"
4. udfName 连接的模型实例名，这里是前面启动的 deepseek_chat。为什么叫 udfName, 是因为该实例名称在 Byzer-SQL 中直接可以当做函数来用，这个我们会在后面看到。



点击上面的红框部分的执行按钮，如果没有任何输出异常，一般就是正确注册上了。不过我们还是来验证下，在新的输入框填入如下信息：

```
1  --%chat
2  --%model=deepseek_chat
3  --%output=q1
4
5  你好
```

这里使用了Byzer 控制台独有的一种语法，

1. `--%chat` 表示这是一个对话框
2. `--%model` 配置我们需要对话的模型实例名。
3. `--%output` 配置我们的对话信息存储的表名，可以随意，方便后续我们支持多轮对话。

点击执行按钮后的输出：

```

1  --%chat
2  --%model=deepseek_chat
3  --%output=q1
4
5  你好

```

Result Job Details Log Message

你好！欢迎使用人工智能助手。有什么我可以帮助你的吗？无论是问题解答、信息查询还是其他任何事情，请随时告诉我。

[查看大图](#)

实际上我们也可以进行多轮对话，你可以通过 output/input 来衔接两个对话，比如：

```

1  --%chat
2  --%model=deepseek_chat
3  --%output=q1
4
5  你好，请记住我是祝威廉

```

Result Job Details Log Message

你好，祝威廉！很高兴认识你。如果你有任何问题或需要帮助，请随时告诉我。我会尽力为你提供信息和支持。

[查看大图](#)

```

1  --%chat
2  --%model=deepseek_chat
3  --%input=q1
4  --%output=q2
5
6  请问我的名字是谁？

```

Result Job Details Log Message

你的名字是祝威廉。我已经记住了，如果你有任何问题或需要帮助，请随时告诉我。

[查看大图](#)

我在第二个对话里，可以看到大模型记住我的第一个对话。这里主要是通过 input/output 来衔接的。

除了上面这种使用方式以外，你当然也可以直接在 SQL 中使用，下面可以达到上面相同的结果：

```
1  select
2  deepseek_chat(llm_param(map(
3      "instruction", '我是威廉，请记住我是谁。'
4  )))
5
6  as response as table1;
7
8  select llm_result(response) as result from table1 as output;
9
10
```

下面是执行后的输出结果：

Cell 4

```
1  select
2  deepseek_chat(llm_param(map(
3      "instruction", '我是威廉，请记住我是谁。'
4  )))
5
6  as response as table1;
7
8  select llm_result(response) as result from table1 as output;
9
10
```

Result

Job Details

Log Message

result

当然，威廉，我已经记住了你的名字。如果你有任何问题或需要帮助，随时可以告诉我。

共 1 条 < 1 > 前往 1 页

↓

使用上涉及到几个知识点：

1. 参数需要通过两层包裹: `llm_param(map(...))` 才能传递给模型方法。

2. 返回的结果字段实际上需要通过 llm_result 来进行文本抽取。

response 字段其实是一个比较复杂的数组，数组的每个元素是一个复杂的json字符串。你可以通过下面的命令查看：

```
1 select * from table1 as output;
```

下面是输出：



The screenshot shows a SQL query execution interface. At the top, the query `select * from table1 as output;` is entered. Below the query, there are tabs for "Result", "Job Details", and "Log Message". The "Result" tab is selected, showing a JSON array of results. The first result is a JSON object with the following fields: `"output"`, `"input"`, `"history"`, `"temperature"`, `"top_p"`, `"max_length"`, `"metadata"`, `"request_id"`, `"input_tokens_count"`, `"generated_tokens_count"`, `"time_cost"`, `"first_token_time"`, and `"speed"`. The output field contains a string of Chinese text. At the bottom, there is a pagination bar showing "共 1 条" (Total 1 item) and "1" page.

可以看到 llm_result 可以把大模型的实际回复抽取出来。

另外，Byzer-SQL 也可以实现多轮对话，比如我可以接着 table1 继续对话：

```
1 select
2   deepseek_chat(llm_stack(response,llm_param(map(
3     "instruction",'请问我是谁? '
4   )))
5
6 as response from table1
7 as table2;
8
9 select llm_result(response) as result from table2 as output;
```

执行后输出：



```

1 select
2 deepseek_chat(llm_stack(response,llm_param(map(
3     |         |         |         |
4 )))
5
6 as response from table1
7 as table2;
8
9 select llm_result(response) as result from table2 as output;

```

Result Job Details Log Message

result

你是威廉，我之前已经记住了你的名字。如果你有任何其他问题或需要进一步的帮助，请随时告诉我。

共 1 条 < 1 > 前往 1 页



可以看到，对话可以通过 Byzer-SQL 表之间进行传递。这里唯一需要注意的是，我们需要通过 `llm_stack` 将 `table1` 中的 `response`(或者你改名后的字段名称) 和 新的问题一起拼接起来，这样系统才能拿到完整的对话。

`llm_stack` 函数会自动帮你把 `table1` 中的对话和新的对话做拼接。

Byzer-SQL 函数中的 prompt 管理

正如之前我们在 `byzerllm` 提到的，`prompt` 管理是一件复杂和有挑战的事情。对于复杂而超长的 `prompt`, `Byzer-SQL` 也提供了 `llm_prompt` 函数 来实现 `prompt` 的模板化渲染。

来看一个例子：


```

1  select "Byzer-SQL 是一门SQL语言, 可以和大模型友好的协作, 更高效的完成数据分析任务。"
   as
2  context as rag_table;
3
4  select
5  deepseek_chat(llm_param(map(
6      "instruction", llm_prompt('
7
8  根据下面提供的信息, 回答用户的问题。
9
10 信息上下文:
11  ``
12  {0}
13  ``
14
15  用户的问题:  Byzer-SQL 是什么?
16
17  ',array(context))
18
19  )))
20
21  as response from rag_table as q3;
22
23  select llm_result(response) as result from q3 as output;
24

```

第一条 Byzer-SQL 语句, 我们模拟了一张表, 表里面有个 context 字段。接着第二条语句, 我们写了一个 prompt 语句, 然后里卖弄支持位置变量渲染, `{0}` 会被 llm_prompt 的第二个参数, 也就是一个数组的第一个元数替代。我们相当于简单实现一个RAG。我们来看看执行结果:

```
1 select "Byzer-SQL 是一门SQL语言，可以和大模型友好的协作，更高效的完成数据分析任务。" as
2 context as rag_table;
3
4 select
5 deepseek_chat(llm_param(map(
6   | | | "instruction", llm_prompt('
7   根据下面提供的信息，回答用户的问题。
8
9   信息上下文：
10  ...
11  {0}
12  ...
13
14  用户的问题：Byzer-SQL 是什么?
15
16  ', array(context)))
17
18 )))
19
20 as response from rag_table as q3;
21
22 select llm_result(response) as result from q3 as output;
23
24
25
```

Result Job Details Log Message

result

Byzer-SQL 是一门SQL语言，它能够与大模型（可能是指大型数据模型或机器学习模型）友好协作，从而更高效地完成数据分析任务。这种语言的设计旨在优化与大型数据处理和分析系统的交互，提高数据操作的效率和性能。

这里简单再介绍下 llm_prompt 函数，第一个参数是一个模板，模板里可以使用位置变量，类似 {0}，{1} 这样的特殊字符，然后 llm_prompt 的第二个参数是一个数组，这个数组的值可以是字面量，也可以是来自其他表的数据，最后 llm_prompt 会渲染成一个纯文本给到 deepseek_chat 模型函数。

Byzer-SQL 和大模型集成的意义

你可能会好奇，在 Byzer-SQL 中这样集成大模型的意义在于哪？

1. 通过 --%chat 你可以不用离开控制台就可以和大模型聊天，比如让大模型帮你写 Byzer-SQL。
2. 通过 Byzer-SQL 可以对数据库或者其他数据源的数据，使用大模型做数据加工处理。大部分企业的数据都是纯文本，实际上很难被实际利用起来，我们可以通过 Byzer-SQL 工程化的对这些数据使用大模型处理，比如抽取QA问答对，实现数据的半合成化，然后再去微调大模型。
3. Byzer-SQL 实现了企业数据流动的闭环。可以将各种数据源的数据做聚合和清洗（ETL），然后在使用大模型做半合成加工，微调大模型，部署大模型，然后再用新的大模型处理数据。

在下面一个章节，我们会来一个实际的例子，如何使用 Byzer-SQL 加大模型对数据进行分析。

