

Process For Adding New JDBC Database

In this section we will describe how to add new JDBC connections for data sources and sinks within IOblend. You will need to have access to the databases setup and granted to you beforehand. You will also need to check the database documentation for specific instructions on the particulars of the process *(read below)*:

1) First, locate the IOblend folder (*where it is installed on your machine*). There you will find an **iob_jars** subfolder on the list.

(i)	- 🔳 View -			
Disk (C:) > Users > euror > Documents > in	oblend_designer	~ C	Q. Search ioblend_designer	
Name	Date modified	Type	Size	
google_cloud_core-2.3.2.dist-info	29/09/2022 18:37	File folder		
google_cloud_storage-2.5.0.dist-info	29/09/2022 18:36	File folder		
google_crc32c	29/09/2022 18:34	File folder		
hadoop	29/09/2022 18:37	File folder		
📁 idlelib	29/09/2022 18:37	File folder		
🚞 images	29/09/2022 18:37	File folder		
io_designer_config	29/09/2022 18:35	File folder		
📁 io_java	29/09/2022 18:36	File folder		
🛛 📁 iob_jars	29/09/2022 18:37	File folder		
📁 ioblend_python	29/09/2022 18:37	File folder		
🚞 java_packages	29/09/2022 18:36	File folder		
📁 numpy	29/09/2022 18:34	File folder		
p3-test	22/11/2022 14:54	File folder		
🚞 pandas	29/09/2022 18:34	File folder		
PIL	29/09/2022 18:34	File folder		

- 2) copy the new JDBC jar for your database into this folder, e.g. for Snowflake <u>https://docs.snowflake.com/en/user-guide/jdbc-download.html</u>. (Note: iob_jars folder has already been added by IOblend into the classpath)
- 3) From the JDBC documentation for your database find the **driver name** that the DB provider instructs you to use
- 4) Using the same JDBC documentation for the new database, work out your **URL connection string**
- 5) Navigate to your **playbook** location and find **config** subfolder. This is a folder IOblend would have created for your projects when you first performed the installation (see the <u>installation</u> and <u>run parameter</u> guides)

) New ~ do	^{CD}		rt - 🔲 View -			
→ × ↑ •	· es	uror > Documents (ioblend_designer > Test_p	sipeline > config	× C a	Search config	
Home	1	Name	Date modified	Type	Size	
🖿 Val - Personal		credentials	13/01/2023 12:38	JSON File	5 KB	
	- 1	io_env_non_win_setup.py	13/01/2023 12:38	PY File	5 KB	
Desktop		io_env_setup.sh	13/01/2023 12:38	SH File	2 KB	
Downloads		ioblend_run_params	13/01/2023 12:43	JSON File	2 KB	
Documents	4	🖬 🚽 jdbc_db_sql	13/01/2023 12:57	JSON File	7 108	
Pictures	4	idbc_db_sql_template - Copy	13/01/2023 12:38	JSON File	3 KB	
Music	4	jdbc_db_sql_template	13/01/2023 12:38	JSON File	3 KB	
Videos	2	jdbc_db_sql_template_old_version	13/01/2023 12:38	JSON File	3 KB	
Google Drive (G:)		idbc_db_sql_template_v1	13/01/2023 12:38	JSON File	3 KB	
Invoices FY23		idbc_db_sql_template_v02	13/01/2023 12:38	JSON File	4 KB	
FY21		spark_submit_command	13/01/2023 12:57	Text Document	1 KB	
FY20						
config	2					

Locate and open the file **jdbc_db_sql** in Notepad. Copy an existing entry for another jdbc db (we suggest you copy the provided bigquery one – see below). Add this copy to the bottom of the file and change the name to be the same as the **drivername** for the new JDBC connection



6) Within this file, under dbToSparkTypes, check that **all** your new database data types that you'll be using have been **mapped to a Spark data type**. If you miss one - don't worry, when you run IOblend it will generate an error and tell you which datatype mapping is missing. Save and exit.

File Edit View "com.simba.googlebigquery.jdbc.Driver": ("mod_col": "alter table #table_name modify column #col_name #col_vale "view": "create or replace view #dbtable as(with cte as (SELECT #t_cols, ROW,MURBER() OVER (PARTITION BV #pke; "primary_key": "alter table #table_name add primary key(#tacloumn_names)", "foresign_key": "alter table #table_name add constraint FK_#table_name foreign key(#fkeys) references #reference "create_table_simple": "Create_statement":"create table IF NOT EXISTS #table_name (#col_def)", "col_name_datatype_sep": ",", col_def":"#col_name #col_data_type"),	
"com.simba.googlebigquery.jdbc.Driver": ("mod_col": "alter table #table_name modify column #col_name #col_val "view": "create or replace view #dbtable as(with cte as (SELECT #t_cols, ROW_NUMBER() OVER (PARTITION BY #pke; "primary_key": "alter table #table_name add primary key(#column_names)", "foreign_key": "alter table #table_name add constraint FK_#table_name foreign key(#fkeys) references #referent "create_table_simple": ("create_statement":"create table IF NOT EXISTS #table_name (#col_def)", "dbloScarkfvoes": ("yirchan": dfrientfvoes": "formate": null).	
<pre>"string": ("sparkType": "StringType", "format": null), "bit": ("sparkType": "ByteType", "format": null), "bool": ("sparkType": "BooleanType", "format": null), "boolean": ("sparkType": "IntegerType", "format": null), "int64": ("sparkType": "IntegerType", "format": null), "integer": ("sparkType": "IntegerType", "format": null), "datetime": ("sparkType": "IntegerType", "format": null), "datetime": ("sparkType": "IntegerType", "format": null), "datetime": ("sparkType": "IntegerType", "format": "dd-HWH-yyyy HH:mm:ss"), "timestamp": ("sparkType": "DubleType", "format": "dd-HWH-yyyy HH:mm:ss"), "timestamp": ("sparkType": "IntegerType", "format": "dd-HWH-yyyy HH:mm:ss"), "timestamp": ("sparkType": "IntegerType", "format": "dd-HWH-yyyy HH:mm:ss"), "timestamp": ("sparkType": "IntegerType", "format": "dd-HWH-yyyy HH:mm:ss"), "dtac54: ("sparkType": "IntegerType", "format": "dd-HWH-yyy HH:mm:ss"), "stank_insert_statement": ("sql": "insert into #table_name(#col_names) values #val_param", "val_param_sep": ","row_seperator_end": ")", "between_row_seperator":","), "nosOfParallelWrites":600),</pre>	<pre>/alue", keys ORDER BY #eventkeys) / ence_key_table(#rkeys)", ence_key_table(#rkeys)", and #eventkey <=#key_to #pi and #eventkey <=#key_to #pi</pre>

7) Run IOblend. Open existing or create a new playbook and add the new source or sink that will be connecting to the new JDBC database Once you've created the playbook component, add an entry into the credentials file for the new source or sink with the username and password or key token, as needed.

Documents > ioblend_designer > Test_p	ipeline > config	~ C q	Search config	
Name ^	Date modified	Туре	Size	
🛯 🗐 credentials	13/01/2023 12:38	JSON File	5 KB	
io_env_non_win_setup.py	13/01/2023 12:38	PY File	5 KB	
io_env_setup.sh	13/01/2023 12:38	SH File	2 KB	
ioblend_run_params	13/01/2023 12:43	JSON File	2 KB	
jdbc_db_sql	13/01/2023 12:57	JSON File	7 K8	
idbc_db_sql_template - Copy	13/01/2023 12:38	JSON File	3 KB	
jdbc_db_sql_template	13/01/2023 12:38	JSON File	3 KB	
idbc_db_sql_template_old_version	13/01/2023 12:38	JSON File	3 KB	
idbc_db_sql_template_v1	13/01/2023 12:38	JSON File	3 KB	
idbc_db_sql_template_v02	13/01/2023 12:38	JSON File	4 K8	
spark_submit_command	13/01/2023 12:57	Text Document	1 KB	

Note: if you are using username and password, this **does not** need to be added to the URL. Please enter these details in the **credentials file**. If using a token, the username **must** be called 'token' (see *example credential files*)

Credentials - Notepad			-	۵	×
File Edit View					ŝ
<pre>"password": "xxxxxxx" }, "sink_sentiment_mysql_jdbc_s3": { "dbtable": "sentiment", "user": "xxxxxxx", "password": "xxxxxxx", "dbtable": "sentiment_jdbc": { "dbtable": "sentiment", "user": "xxxxxxx", "password": "xxxxxxxx", "password": "xxxxxxx", "password": "xxxxxxxx", "password": "xxxxxxxxx", "password": "xxxxxxxxx", "password": "xxxxxxxxx", "password": "xxxxxxxxx", "password": "xxxxxxxxx", "password": "xxxxxxxxxx", "password": "xxxxxxxxx", "password": "xxxxxxxxxxxx, "password": "xxxxxxxxxxxxxx, "password": "xxxxxxxxxxx, "password":</pre>					
Ln 1, Col 1	100%	Windows (CRLF)	UTF-8	3	

9) Congratulations, you can now run your playbook component!