**Automate Salesforce Table Creation With Java**

When you ingest data from Salesforce into a relational database, you first need to create a table for the object you want to ingest. Writing a create statement manually is cumbersome and you often need to debug it a few times. Salesforce data types are quite different from database ones. Objects sometimes have a large number of fields (like hundreds).

ETL tools often have an option to create table from Salesforce data ingestion and table creation can done when you ingest the data. However, sometimes your ETL tool does not have an option to create a table with a particular database (e.g. Informatica Cloud does not automatically create tables in Postgres).

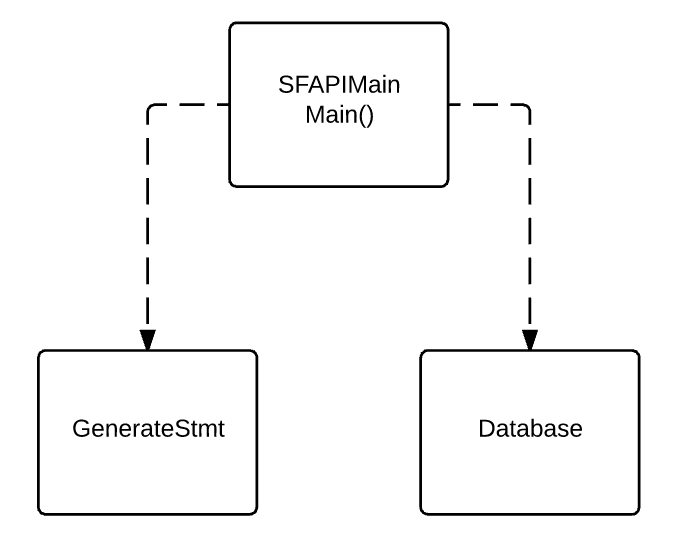
I created a small Java application to create a table in a relational database. The example has a data type hash map for MySQL and Postgres. By editing the hash map and adding the appropriate JDBC, this should work for other databases.

**Salesforce API Reference**

In this post, I used [REST-based API](https://developer.salesforce.com/docs/atlas.en-us.api_rest.meta/api_rest/intro_what_is_rest_api.htm). Salesforce also supports [WSDL-based SOAP API](https://developer.salesforce.com/page/SOAP_API) if you want to use SOAP instead. I found this document ([Setting Up Your Java Developer Environment](http://resources.docs.salesforce.com/210/10/en-us/sfdc/pdf/salesforce_developer_environment_tipsheet.pdf)) very useful to get started with Salesforce API with Java. Using Salesforce API with Python, I recommend you to use Simple Salesforce module in [my previous post](https://www.mydatahack.com/salesforce-api-with-simple-salesforce-for-python/).

**Application Design**

There are 3 classes with the main method included in the SFAPIMain class. The Database class connects to database with JDBC and execute create table statement. The GetMetaData class gets object metadata from Salesforce and converts it to a create table statement. The SFAPIMain class handles Salesforce API calls and get user input and display outcome. Make sure to build path for appropriate JDBC Jar files.



Compile it as a runnable Jar file. Enter the salesforce object and target schema, then it will create a table for you!

The program also generates a metadata csv and create table statement file. The key for this program to work is to have the correct data type mapping in the code. The only difference between MySQL and Postgres for this code to work is the data type for datetime is datetime in MySQL and timestamp in Postgres. Postgres has a data type called money. However, it is best to use decimal with scale of 2 because money type in Postgres becomes String when you query it with other BI tools. This will cause issues when you try to aggregate the field in the BI tool. There is a thread about [postgres datatype for currency](https://stackoverflow.com/questions/15726535/postgresql-which-datatype-should-be-used-for-currency) for further information.

**Code**

**SFPIAMain.java**

**[cc lang="java" tab\_size="4" lines="-1"]**

**[/cc]**

**GetMetaData.java**

**Database.java**