# SQL adapter - Communicating with SQL database

fork and edit tutorial (https://github.ibm.com/MFPSamples/DevCenter/tree/master/tutorials/en/foundation/6.3/server-side-development/sql-adapter-communicating-sql-database.html) | report issue (https://github.ibm.com/MFPSamples/DevCenter/issues/new)

#### **Overview**

An IBM MobileFirst Platform Foundation SQL adapter is designed to communicate with any SQL data source.

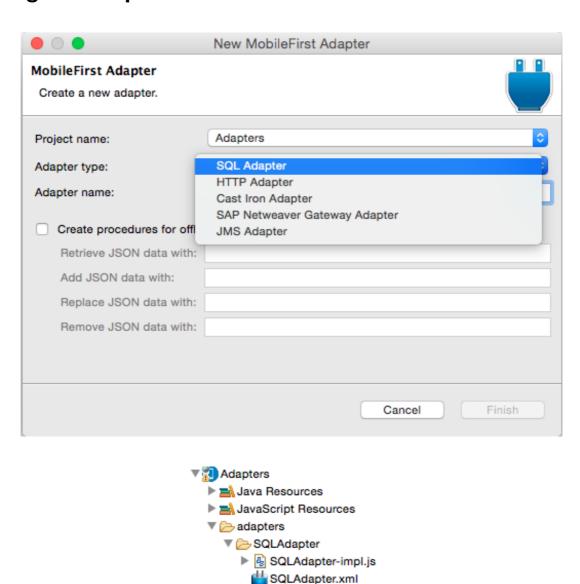
You can use plain SQL queries or stored procedures.

As a developer, you must download the JDBC connector driver for the specific database type separately and add it to the **server\lib**\ folder of a MobileFirst project.

You can download the JDBC connector driver from the appropriate vendor website.

In this tutorial and in the accompanying sample, you learn how to use a MobileFirst adapter to connect to a MySQL database.

#### Creating the adapter



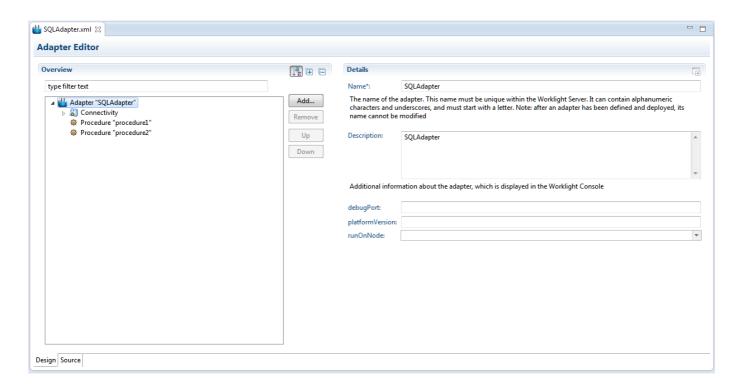
In MobileFirst Studio, create a MobileFirst adapter and choose the SQL adapter type.
 A standard SQL adapter file structure is created.

2. Save the downloaded JDBC connector file in the project **server\lib\** folder.

# **Adapter XML**

Settings and metadata are stored in the adapter XML file.

You can use either the Design or the Source editor to modify the adapter XML file.



- 1. In the adapter XML file, declare the following parameters:
  - Driver Class
  - Database URL
  - Username
  - Password

2. Declare a procedure in the adapter XML file.

### **JavaScript Implementation File**

The adapter JavaScript file is used to implement the procedure logic.

**Important:** The name that is declared in the XML file must be used for the procedure JavaScript function.

There are two ways of running SQL statements:

- SQL statement query
- SQL stored procedure

Use the WL.Server.createSQLStatement method to prepare a SQL query.

The WL.Server.createSQLStatement method must always be called outside the function. Add more parameters, if necessary.

```
//Create SQL query
var getAccountsTransactionsStatement = WL.Server.createSQLStatement(
   "SELECT transactionId, fromAccount, toAccount, transactionDate, transactionAmount, transactionType "
+
   "FROM accounttransactions " +
   "WHERE accounttransactions.fromAccount = ? OR accounttransactions.toAccount = ? " +
   "ORDER BY transactionDate DESC " +
   "LIMIT 20;"
);
```

Use the WL.Server.invokeSQLStatement method to call prepared queries. Return the result to the application or to another procedure.

```
//Invoke prepared SQL query and return invocation result

function getAccountTransactions1(accountld){
    return WL.Server.invokeSQLStatement({
        preparedStatement : getAccountsTransactionsStatement
        ,
        parameters : [accountld, accountld]
        });
    }
```

To run a SQL stored procedure, use the WL.Server.invokeSQLStoredProcedure method.

Specify a SQL stored procedure name as an invocation parameter.

Add more parameters, if necessary.

Return the invocation result to the application or to another procedure.

```
//Invoke stored SQL procedure and return invocation result

function getAccountTransactions2(accountId){
  return WL.Server.invokeSQLStoredProcedure({
    procedure : "getAccountTransactions",
    parameters : [accountId]
});
}<
```

#### **Invocation Results**

```
"isSuccessful": true,
 "resultSet": [{
  "fromAccount": "12345",
  "toAccount": "54321",
  "transactionAmount": 180.00,
  "transactionDate": "2009-03-11T11:08:39.000Z"
  "transactionId": "W06091500863",
  "transactionType": "Funds Transfer"
 }, {
  "fromAccount": "12345",
  "toAccount": null,
  "transactionAmount": 130.00,
  "transactionDate": "2009-03-07T11:09:39.000Z"
  "transactionId": "W214122\/5337",
  "transactionType": "ATM Withdrawal"
 }]
}
```

The result is retrieved as a JSON object.

The isSuccessful property defines whether the invocation was successful.

The resultSet object is an array of returned records.

To access the resultSet object on the client-side: result.invocationResult.resultSet

To access the resultSet object on the server-side: result.ResultSet

# Sample application

The attached sample project contains an SQL adapter.

To run the sample, execute the **mobilefirstTraining.sql** file (which you can find under the Server folder of the sample) on your local MySQL server.

Make sure that the mobilefirst@ user has all access permissions that are assigned to it.

Remember to download and set a MySQL Java Connector in your project.

Click to download

(http://public.dhe.ibm.com/software/products/en/MobileFirstPlatform/docs/v630/MobileFirstAdaptersProject.zip) the Studio project.