Import and export data from your Bluemix SQL Database with SQL Database service

Senior Technical Staff Member, Information Management Cloud Services
IBM

Learn how to import data into the Bluemix SQL Database. See a sample application to show how data can be exported.

Maybe you have played around with the new IBM Bluemix platform and noticed the SQL Database service listed with the other database services such as IBM Cloudant, MySQL, MongoDB, and PostgreSQL. SQL Database provides an on-demand relational database for transactional workloads.

This tutorial describes how to build and deploy a small Java™ app that uses the SQL Database service. Learn how to create a Java runtime, bind a SQL Database service instance to it, upload a Java app, and access the SQL Database console to create some tables and load up some data. Because the console does not enable you to export a full table, use the Java app to export data from a Bluemix SQL Database table to the user's machine.

"The SQL Database service is an easy and quick way to add a powerful relational database to your web or mobile application. Under the covers, it is powered by IBM DB2."

The Bluemix documentation contains a whole section on getting started with building a web application. It explains everything from how to install the Bluemix cf tool to how to upload your app. Most of these steps can be performed through the Bluemix user interface.

Run the app
Get the code

Note: This sample application allows users to export any of their tables and as such provides web users with access to all of the SQLDB data. In a real world environment you would need to secure
the application with authentication to make sure only certain users have access. In addition, you might need to limit access to different sets of tables to different groups of users through some set of authorization rules. Refer to the Bluemix documentation to determine how to add authentication to an application.

**Build a simple Java app**

To build a simple Bluemix application, take these steps:

- Create a runtime for your app. The runtime can be a Java, Node.js, Ruby Sinatra, Ruby Rails, or a custom build-pack runtime.
- Create and bind the appropriate services to the application. In this case, bind the SQL Database service to the app.
- Code and build or compile your application locally on your workstation and then upload it to Bluemix using the `cf push` command line tool.

Choose among several tools and methods to develop code using Bluemix.

- Use the `cf` tool from the command line to do almost anything.
- Use the Bluemix user interface to create and bind the runtimes and services.
- Use Eclipse to create your code and use the Eclipse Bluemix plugin to send it directly to Bluemix. You can also use your favorite editor or IDE.
- Use IBM DevOps Services, an online place where you can collaborate to develop, track, plan and deploy straight into Bluemix.

I'll show you one way to get things done but there are plenty of alternatives to suit your needs and preferences.

**Create the runtime**

In the Bluemix user interface go to your dashboard and click + **CREATE AN APP**. Select the **Liberty for Java** runtime from the catalog. Give your application a name. This name needs to be unique across all of Bluemix because it becomes part of your application’s URL. Click **CREATE**. On your dashboard, you see something similar to the following figure:
Note: The icon might be different from the one shown here.

**Bind a SQL Database service instance**
To see the app runtime overview, click the app. On that view, click + ADD A SERVICE. A view of the Bluemix catalog opens. Select the SQL Database service under the the Data Management section.

On the subsequent window, find a description of the service and the available plans. Select the Small plan. On the right side of the window, make sure that the app that is mentioned is the one you bind this service to. You can then change the name of the SQL Database service instance (which is the name of your database), or leave the default and click CREATE. If prompted to restart the application at this time, click OK.

At this point, Bluemix does two steps for you. It procures an instance of a DB2® database for you, available as a SQL Database service instance. Then it binds this database to the application runtime. This binding step basically enables the runtime to use the database.

The Overview window of your app should now look similar to this image:

![Overview window of an app with a SQL Database instance bound](image)

**Examine the VCAP_SERVICES structure**
**Important:** The user name and password part of the VCAP_SERVICES structure changes each time you bind the service to an application.

Click Runtime in the menu on the left and scroll to the bottom where you see the VCAP_SERVICES information displayed. VCAP_SERVICES is an environment variable, available to your application, that
contains all of the information required to access the database. In a later step when you get to the Java code, you use this information to establish a database connection.

```json
{
  "sqldb": [
    {
      "name": "SQL Database-pvr",
      "label": "sqldb",
      "plan": "sqldb_small",
      "credentials": {
        "port": 50000,
        "db": "I_847692",
        "username": "jdfiwncbg",
        "host": "23.246.228.235",
        "hostname": "23.246.228.235",
        "jdbcurl": "jdbc:db2://23.246.228.235:50000/I_847692",
        "uri": "db2://jdfiwncbg:snfraomfrsbtp@23.246.228.235:50000/I_847692",
        "password": "snfraomfrsbtp"
      }
    }
  ]
}
```

Use the SQL Database console

Click the SQL Database box to go to the Managed Database console starting screen. To open the SQL Database console, click **LAUNCH**.

![Welcome to the Managed Database console](image)

The managed database service, SQL Database, greatly simplifies creating and managing a full relational database. Several typical database tasks, such as running SQL statements, creating backups, loading data, and monitoring usage can be performed through the console. Try out a few of these in the next steps.

Run the DDL

Click **Work with Database Objects** on the Getting started page or select this option from the **Manage** menu to open the data viewer. At the top left, see the drop-down box with the name of your database and its internal DB2 name with the format of `I_123456`. If you own multiple SQL Database instances, they are accessible here.

Use the data viewer to browse the contents of your database. Because you just created this database, it is currently empty, except for the DB2 catalog tables you see in the viewer in schemas such as **SYSCAT** and **SYSIBM**. At times, options in the menu on the right are grayed out. To access...
them you first have to select an item in the table on the left. For example, click the SYSIBM schema to see all the tables in this schema. When you select one of the table's several menu options such as Sample Data, Column Masks becomes available.

The following image shows the data viewer after selecting the SYSIBM schema and the SYSTABLES table. After you select those entries, click the Columns menu item on the left to see the columns.

Click **Run DDL** at the bottom of the menu on the left. (The menu item **RUN DDL** is beneath the section that is visible in the previous screen capture.)

Click **Browse** to upload the employee.sql file from the sample. (Obtain the full code for this sample by clicking **Get the code** at the beginning of this tutorial.) This file contains a simple EMPLOYEE table from the DB2 sample database. The DDL shown in the edit window should look similar to the following code listing.

**Best practice:** When creating or accessing tables in the Bluemix SQL Database, use explicit schema names. This approach keeps the tables from defaulting to adopting the user name as the schema.
CREATE TABLE USER01."EMPLOYEE" (  "EMPNO" CHAR(6) NOT NULL,  "FIRSTNME" VARCHAR(12) NOT NULL,  "MIDINIT" CHAR(1),  "LASTNAME" VARCHAR(15) NOT NULL,  "WORKDEPT" CHAR(3),  "PHONENO" CHAR(4),  "HIREDATE" DATE,  "JOB" CHAR(8),  "EDLEVEL" SMALLINT NOT NULL,  "SEX" CHAR(1),  "BIRTHDATE" DATE,  "SALARY" DECIMAL(9, 2),  "BONUS" DECIMAL(9, 2),  "COMM" DECIMAL(9, 2) );

ALTER TABLE USER01."EMPLOYEE" ADD CONSTRAINT "PK_EMPLOYEE" PRIMARY KEY ("EMPNO");

Click **Run DDL** to run this script and create the **EMPLOYEE** table in the **USER01** schema in your database. You should get the message **The DDL ran successfully**.

**Import the data**

To load data into this table, click **Manage** in the top-level menu and select **Load Data**.

### Step 1. Upload the file

- Click **Browse files** to select the employee.csv data file from your local system.
- Row 1 of this file does not contain the column names; therefore, click **No** to answer that question.
- Set the separator character to **Comma** because this is the character used to distinguish the column values in the file.
Import and export data from your Bluemix SQL Database with SQL Database service

- This load file contains a number of dates; therefore, click Yes to the question Does the file have columns that contain dates or times? and then enter YYYYMMDD as the date format. If this pattern is not in the drop-down list, you can add it yourself.
- Click Load File.

The console now loads the file and shows you a preview. The maximum file size is 20MB.

**Step 2. Choose a target**
- Select Load into an existing table to load the data into the table you created in the previous section.
- Click Next.

**Step 3. Select a table**
- Click the USER01 schema and click the EMPLOYEE table to select the table to load the data into. You see a preview list of the columns of the table.
- Select Append new data into the table at the bottom of the dialog box. This option adds the data to the table. The other option erases the table and replaces it with the content of the file.
- Click Finish.

**Step 4. Load the data into the table**
You should now see a message stating that the load succeeded and giving a summary of how many records were loaded.

Write a Bluemix Java application to export data

The next step is to build a Java application to export the contents of a database table to a file that can be downloaded. (Obtain the full code for this sample by clicking Get the code at the beginning
of this tutorial.) The app displays a list of tables found in the database. The list ignores any DB2 system tables. The user can select one of these tables.

The app writes the records from the table to a file on the server in CSV format. The file can be downloaded by the user.

### Connect to the Bluemix SQL Database

The SQLDBConnection class in the com.ibm.sqljs.db.util package contains the logic to connect to the SQL Database in three different ways: from information in the servlet context, from information in the VCAP_SERVICES variable, or by explicitly supplied connection information. This article uses the VCAP_SERVICES variable information here. The constructor of the class takes the value of serviceName, in this case sqldb, as a parameter. The class contains private instance variables with setters and getters for all the connection information elements.

```java
public SQLDBConnection(final String serviceName) throws SQLDBConnectionException

To get the environment variable, the System.getenv method is used:

```java
vcapServices = System.getenv("VCAP_SERVICES");
```

The VCAP_Services environment variable contains a JSON structure (as shown earlier in this article) which is parsed with the BasicDBObject class from the com.ibm.nosql.json.api package provided in the nosqljson.jar file with the sample. The VCAP_SERVICES structure can potentially contain multiple services because more than one service can be bound to an application. The code traverses the structure looking for a service that contains sqldb.

```java
BasicDBObject obj = (BasicDBObject) JSON.parse(vcapServices);
// get all the keys from the JSON structure
Set<String> keys = obj.keySet();
// Look for the VCAP key that holds the serviceName information
for (String eachkey : keys) {
    // iterate and look for the key, stop on the first match
    if (eachkey.toUpperCase().contains(serviceName.toUpperCase())) {
        setVCAPKey(eachkey);
        break;
    }
}
```
If such a key is found, it can be parsed to extract the required database connection information, such as the host name, the port number, the database name, and the user name and password.

```java
BasicDBList list = (BasicDBList) obj.get(getVCAPKey());
obj = (BasicDBObject) list.get("0");
// if values are not found they'll be set to null
setName((String) obj.get(Constants.NAME));
setLabel((String) obj.get(Constants.LABEL));
setPlan((String) obj.get(Constants.PLAN));
obj = (BasicDBObject) obj.get(Constants.CREDENTIALS);
setHostName((String) obj.get(Constants.HOSTNAME));
setHost((String) obj.get(Constants.HOST));
setPort((int) obj.get(Constants.PORT));
setUsername((String) obj.get(Constants.USERNAME));
setPassword((String) obj.get(Constants.PASSWORD));
setDatabaseName((String) obj.get(Constants.DBNAME));
```

These connection settings can be used to establish a connection to the SQL Database service instance that was bound to the application as follows:

```java
DB2SimpleDataSource dataSource = new DB2SimpleDataSource();
if (getHost() != "") {
    dataSource.setServerName(getHost());
} else {
    dataSource.setServerName(getHostName());
}
dataSource.setPortNumber(getPort());
dataSource.setDatabaseName(getDatabaseName());
dataSource.setUser(getUsername());
dataSource.setPassword(getPassword());
dataSource.setDriverType(4);
con = dataSource.getConnection();
con.setAutoCommit(autoCommit);
```

### Export data

The bulk of the CSV export processing is done in the export method of the `ExportTable` class. The string for the `SELECT` statement is constructed and the statement is run. If the inclusion of a header in the CSV file is required, the result set meta data is used to construct a list of the column names. Subsequently, the result set is processed and all the column values for all the rows are written to the output file.

```java
public int export (String projection) throws SQLException, FileNotFoundException {
    writer = new PrintWriter(this.filename);
    if (projection == "") projection = "";
    String execStatement = "SELECT " + projection + " FROM " + this.tablename;
    Statement stmt = con.createStatement();
    stmt.execute(execStatement);
    ResultSet rs = stmt.getResultSet();
    // iterate through the meta data set, display column headings
    ResultSetMetaData rsmd = rs.getMetaData();
    int columnCount = rsmd.getColumnCount();
    if (header) {
        String columnHeading = rsmd.getColumnName(1);
        for (int i = 2; i <= columnCount; i++) {
            columnHeading = columnHeading + separator + rsmd.getColumnName(i);
        }
        writer.println(columnHeading);
    }
    // iterate through the result set
```
int count = 0;
String row = "";
while (rs.next()) {
    count++;
    row = rs.getString(1);
    // iterate through the columns
    for (int i = 2; i <= columnCount; i++) {
        int type = rsmd.getColumnType(i);
        if (type == Types.VARCHAR || type == Types.CHAR)
            row = row + separator + quotation + rs.getString(i) + quotation;
        else
            row = row + separator + rs.getString(i);
    }
    writer.println(row);
}
// Close the ResultSet
rs.close();
writer.close();
return count;
}

When the table processing is complete and the file is written to the file system, a link is presented to the user so the file can be downloaded.

IBM SQLDB CSV Export

Connection: jdbc:db2://23.246.228.235:50000/I_838143
Exporting table: PVANRUN_CA_IBM_COM.EMPLOYEE
Export completed successfully.
42 records exported
Download file here: PVANRUN_CA_IBM_COM.EMPLOYEE
Go back

Conclusion

Because of the managed nature of the SQL Database in Bluemix, a user does not have access to the DB2 command line interface to perform Load/Unload or Import/Export commands. Instead, the console needs to be accessed to perform these kinds of operations. This article shows how to import data into the Bluemix SQL Database service through the console application. It gives the steps to build a sample Bluemix Java application to export data from a table to your workstation.

SQL Databasehttps://ace.ng.bluemix.net/#/store/cloudOEPaneId=store&serviceOfferingGuid=0d5a104d-d700-4315-9b7c-8f84a9c85ae3&fromCatalog=trueadds an on-demand relational database to your application.

RELATED TOPICS: SQLDB  Bluemix resources  Cloudant  Data Management solutions on Bluemix
About the author

Paul Van Run

@pvanrun on Twitter
Find me on LinkedIn

© Copyright IBM Corporation 2014
Trademarks
(www.ibm.com/developerworks/ibm/trademarks/)