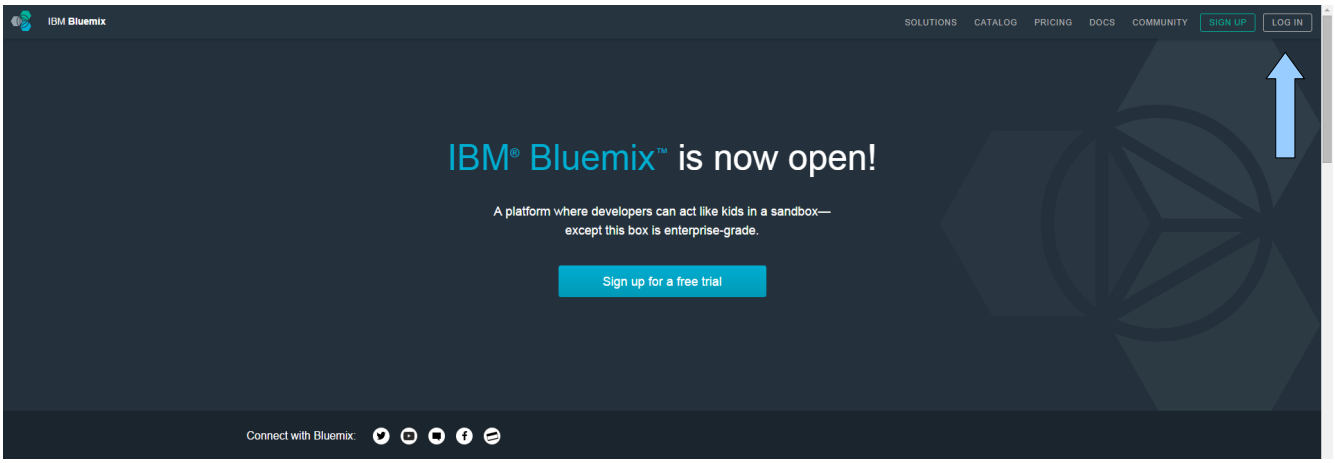
Create a Database Driven Application

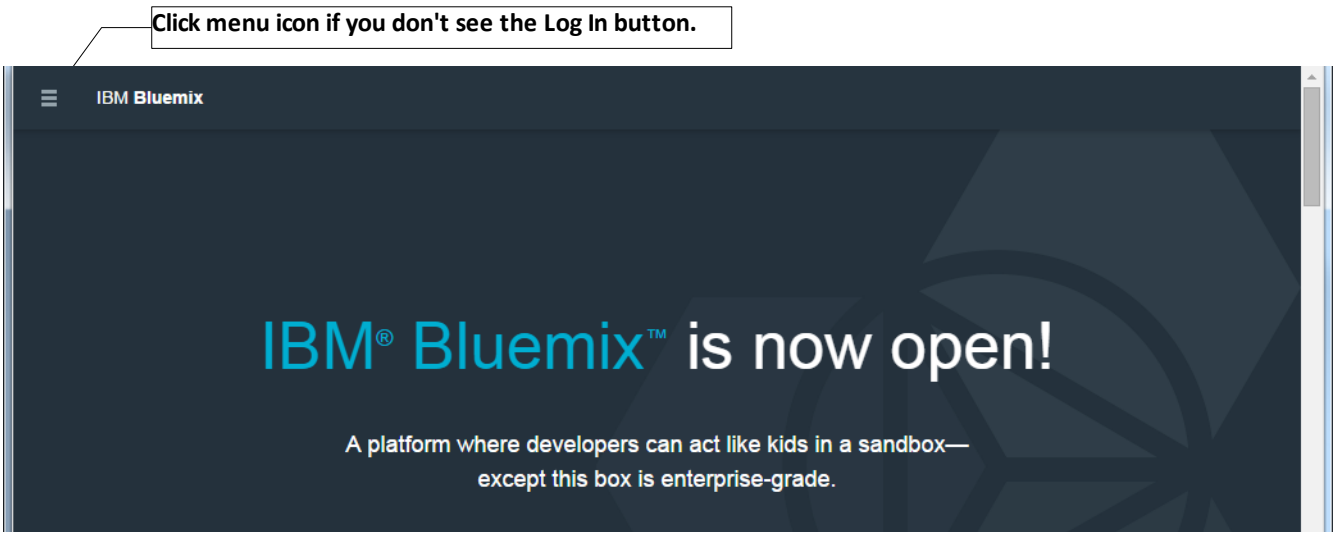
Prerequisites:

- You will need a Bluemix account and an IBM DevOps Services account to complete this project.
- Please review the Registration sushi card for these steps.

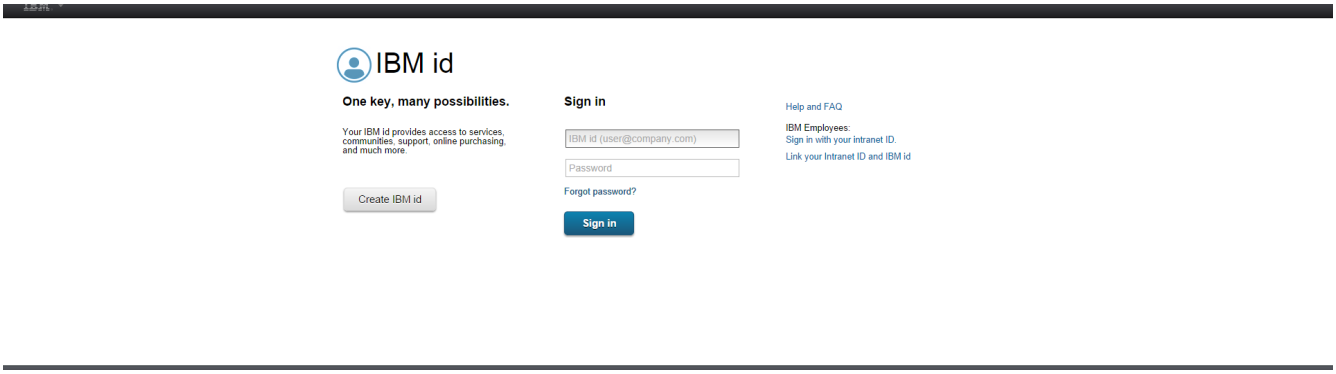
1. Sign In to Bluemix

- Open your web browser (Chrome, Firefox, etc.).
- Enter the following URL in the address bar: https://ace.ng.bluemix.net/
- Click the Log In button. (The Log In button is usually located in the top right-hand corner of the web page. If you do not see the Log In button here, then there may be a 3 bar menu icon in the top left-hand corner of the web page. Click this menu icon, and you should see the Log In button here.)





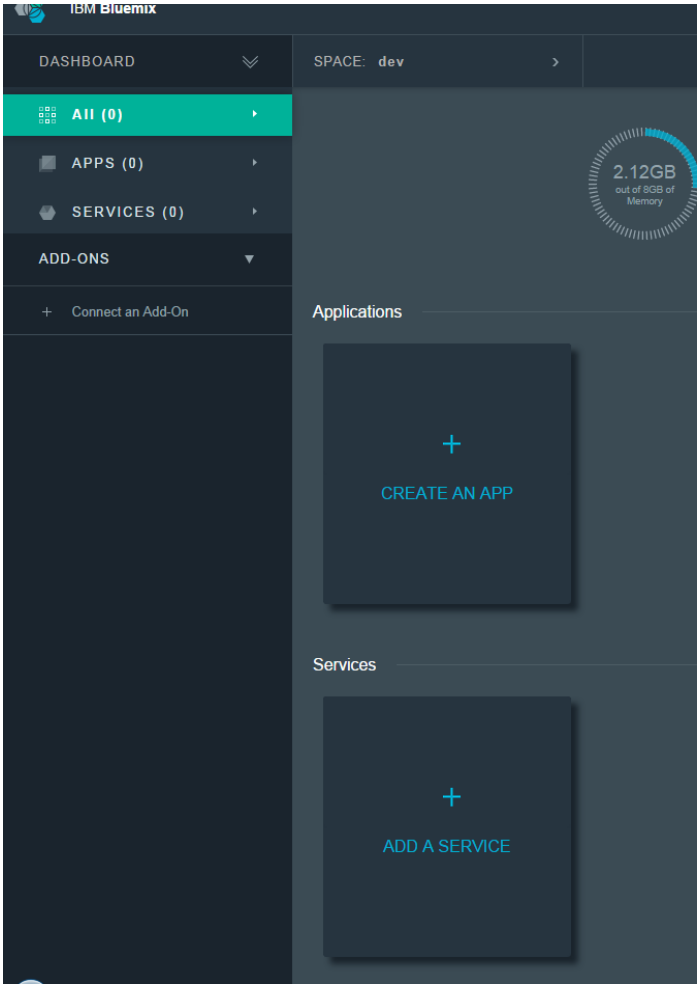
- Enter your log in credentials and click the Sign In button.



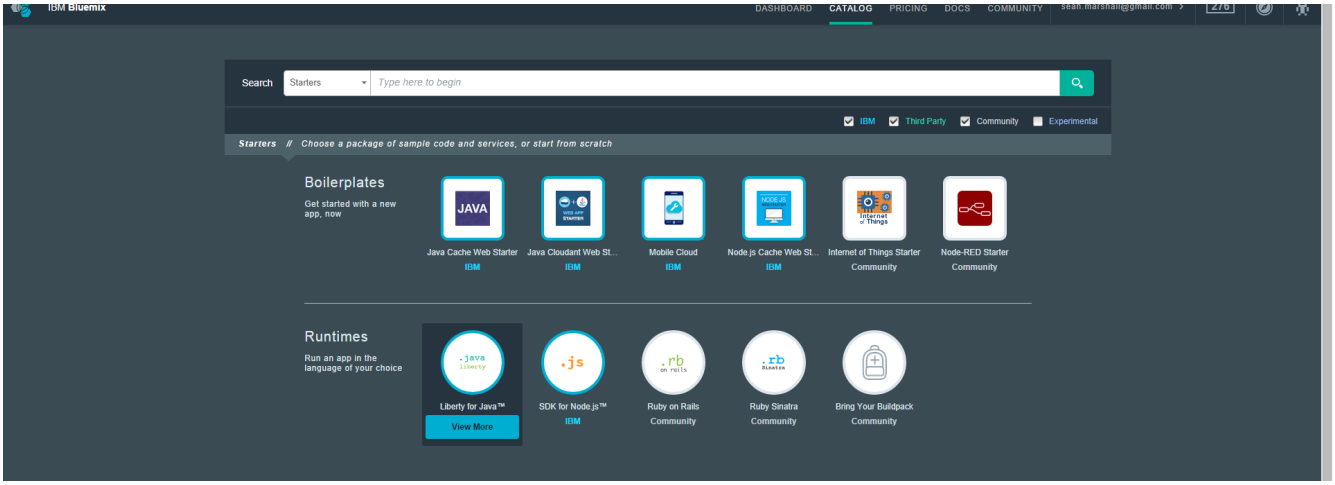
- After you sign in, you will see your Bluemix Dashboard.

2. Create a Java Application

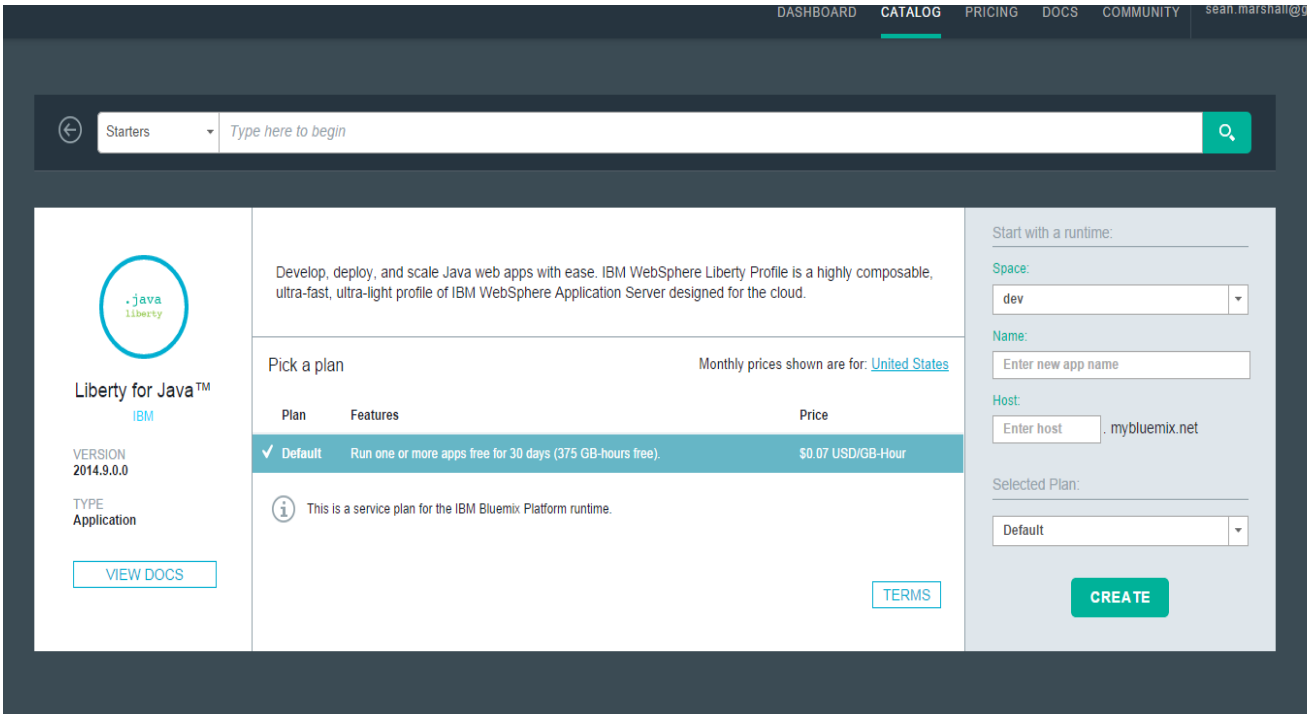
- Click Create an App button on the main dashboard page.



- Select Liberty for Java under the Runtimes section of applications.



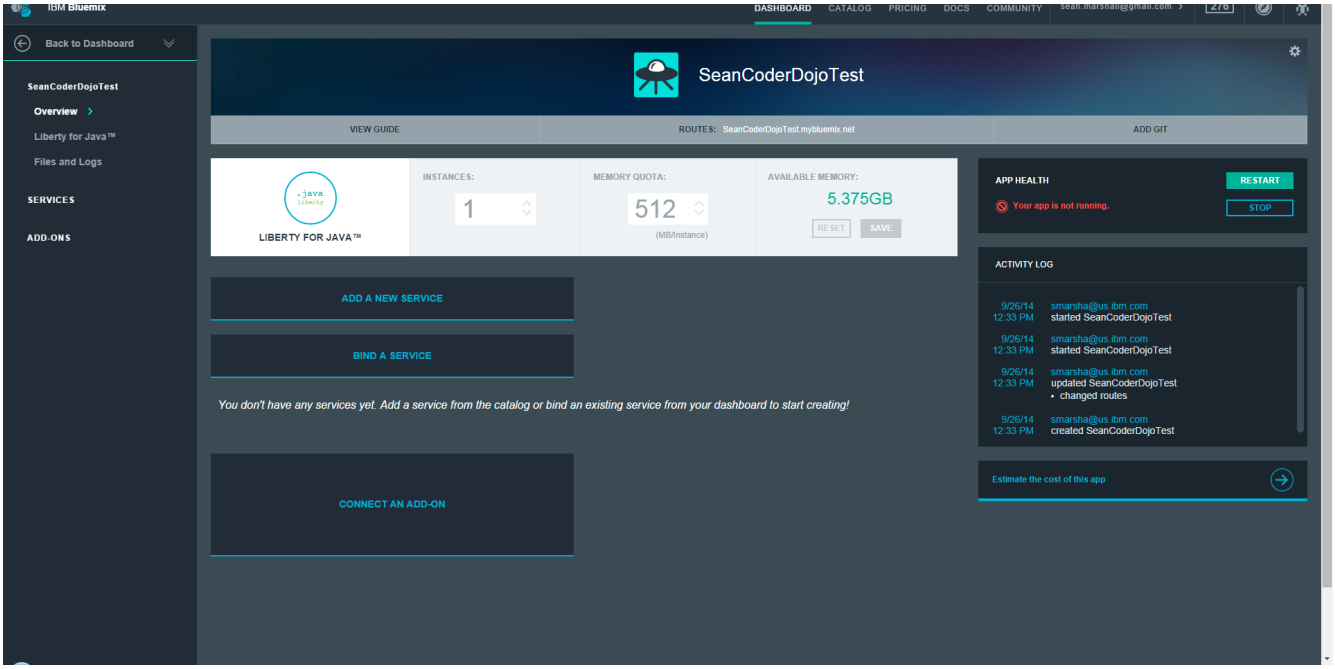
- Enter Runtime details (located on right-hand side of the application details box).
  - [ ] Select the desired Space. (If you only have one space setup for your Bluemix account, then you will not see this dropdown.)
  - [ ] Enter the name of your application. This name needs to be unique across all of Bluemix because it becomes part of your application's URL.
  - [ ] The Host value will default to your application name. There is no need to change this value. This will be the URL for the application.
  - [ ] For Selected Plan, leave as Default.
  - [ ] Click the CREATE button.



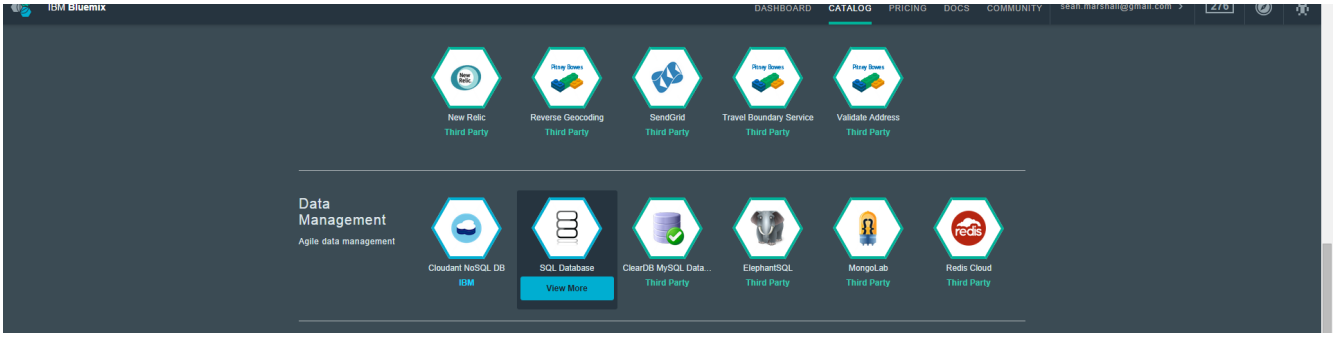
- Congratulations! You created a Java application. Let's continue by adding a database service to the application.

3. Add a Database Service to the Application

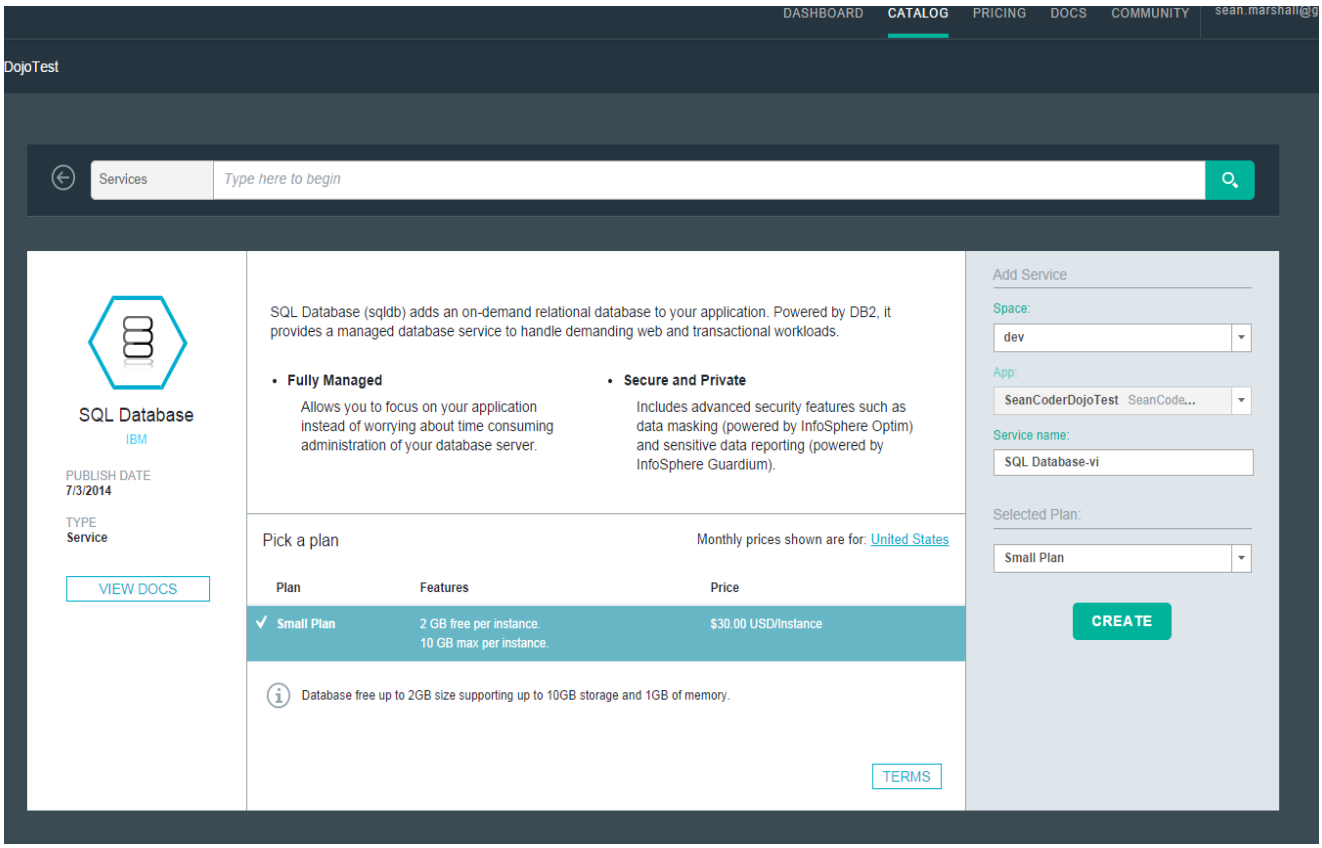
- After creating your Java application, you should be viewing the newly created application. If not, select your application from the main Dashboard.



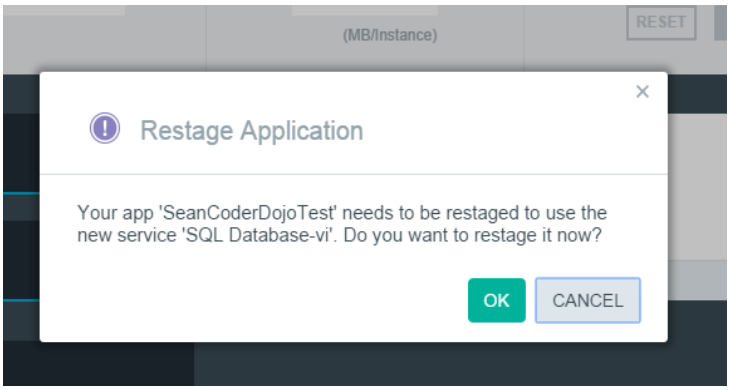
- Click the ADD A NEW SERVICE button.
- Select SQL Database under the Data Management section of services.



- Enter Service details (located on right-hand side of the service details box).
  - [ ] Select the desired Space. (If you only have one space setup for your Bluemix account, then you will not see this dropdown.)
  - [ ] Ensure App Name matches Java app name. This will bind the service to your Java application.
  - [ ] Bluemix will give your database service a default name. There is no need to change this value.
  - [ ] For Selected Plan, select Small Plan.
  - [ ] Click the CREATE button.



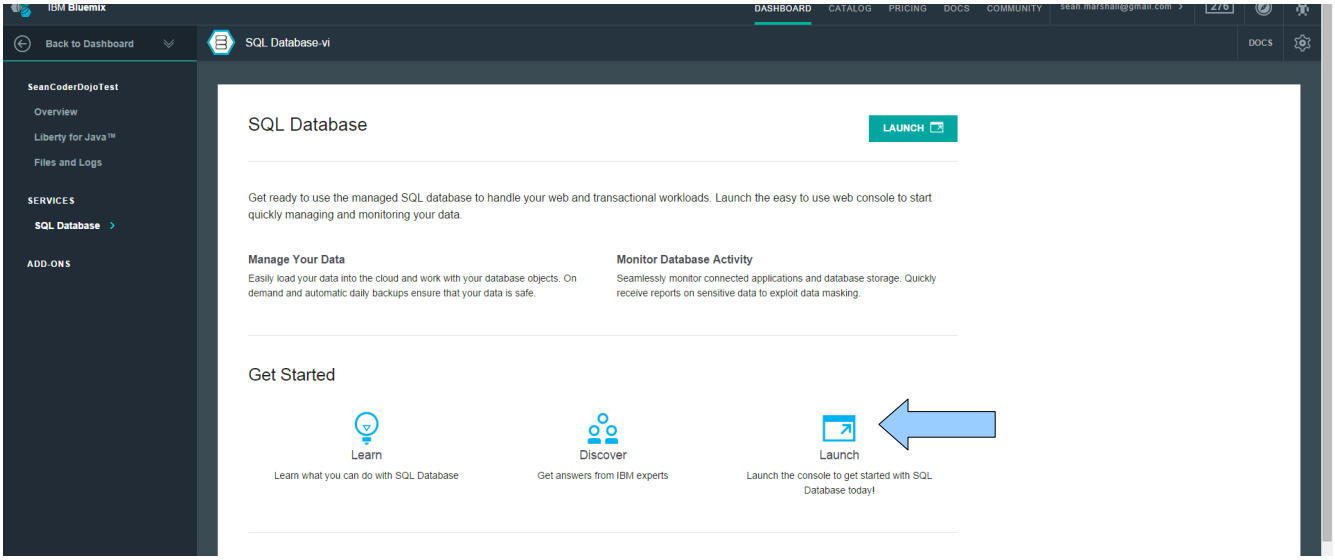
- If prompted to Restage Application, click OK to do so.



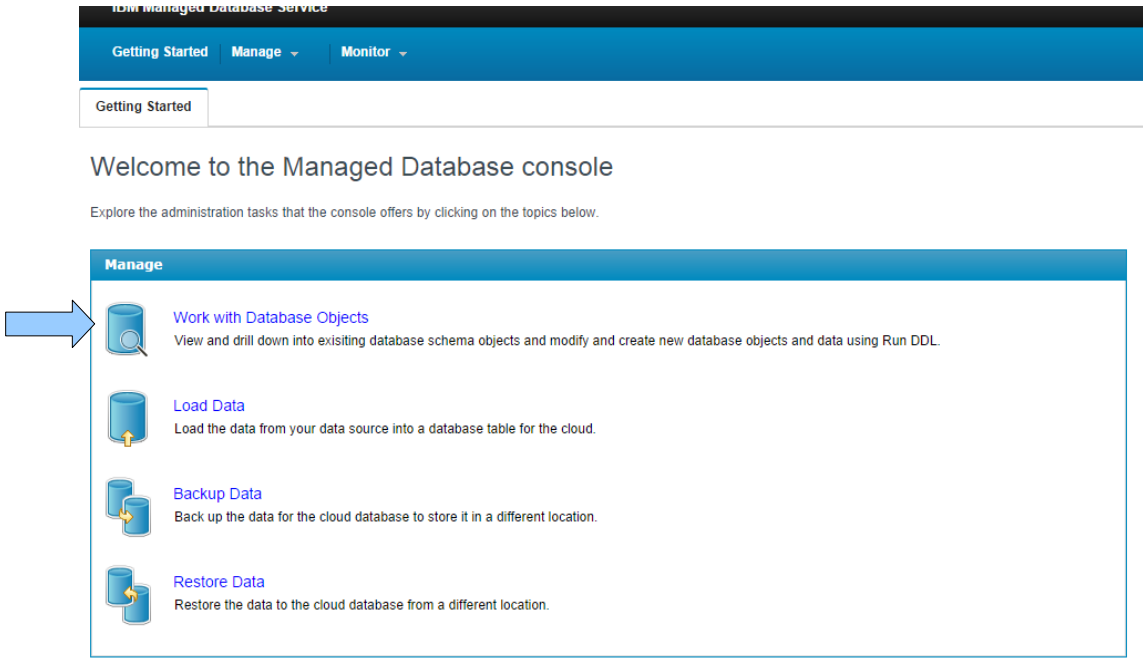
- Congratulations! You have created your application's database and connected the database to your Java application.

4. Setup Database

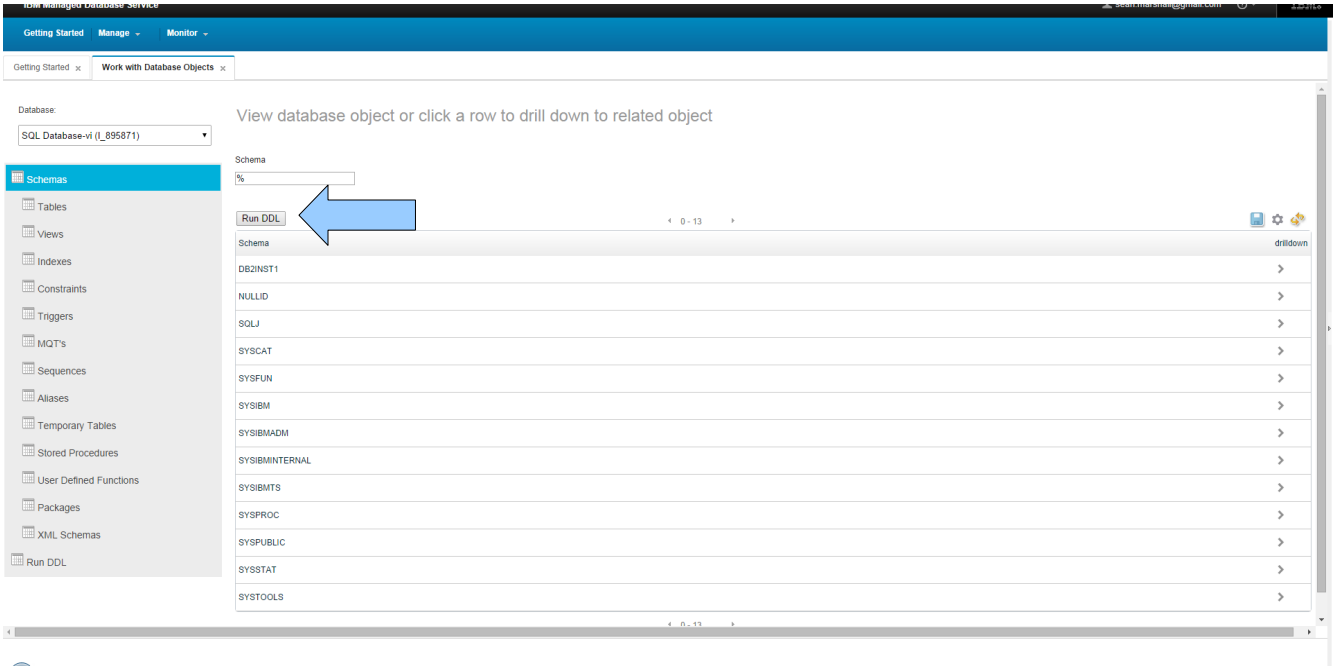
- In the left navigation menu, click SQL Database under Services.
- Under Get Started, click Launch, which should launch the Database Management Console.



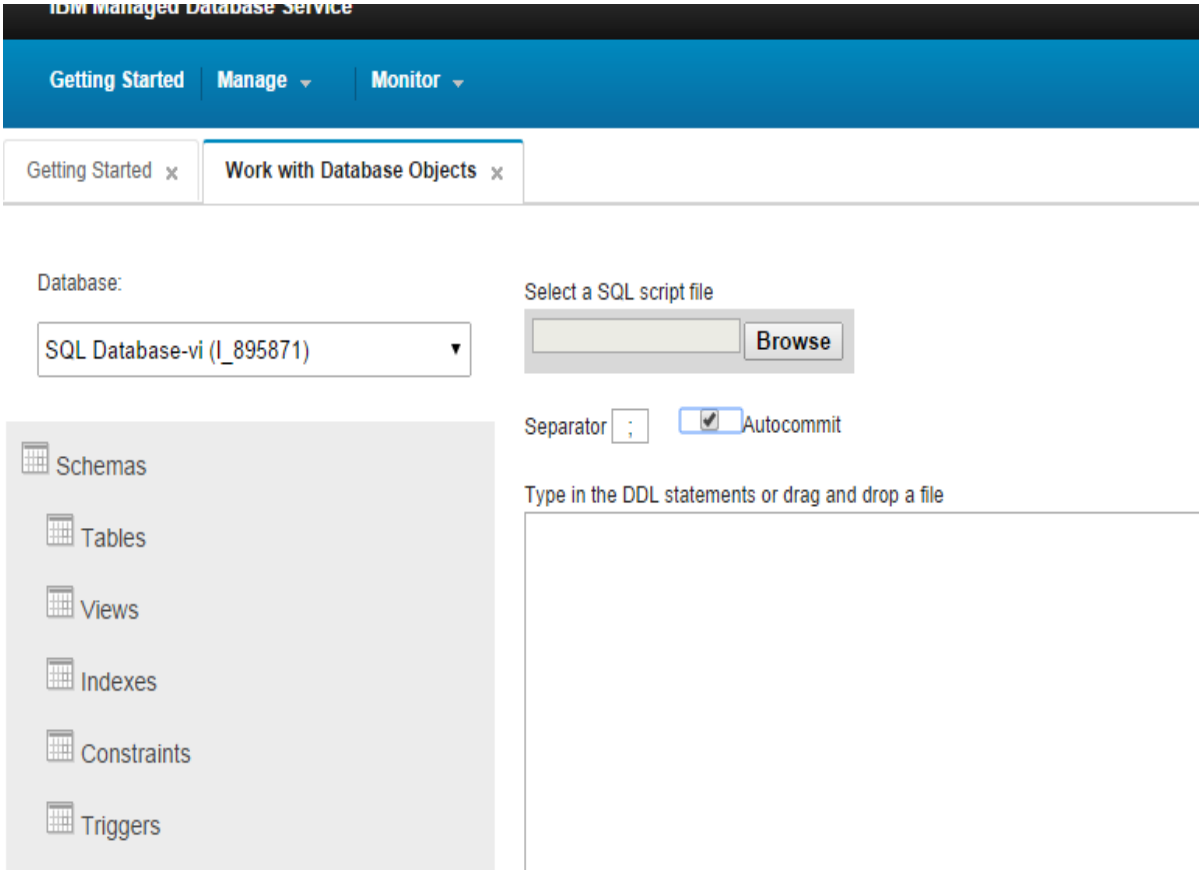
- We need to create some database tables. Click Work with Database Objects.



- Click the Run DDL button.



- Make sure the Autocommit checkbox is checked.

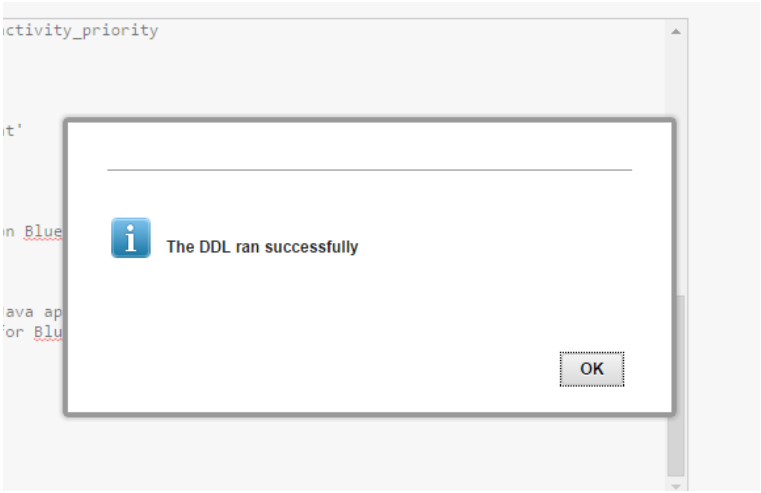


- In the text box below where it states “Type in the DDL statements or drag and drop a file,” copy the SQL scripts below. Start with the line /*** START DB SCRIPTS ***/ and end with the line /*** END DB SCRIPTS ***/.

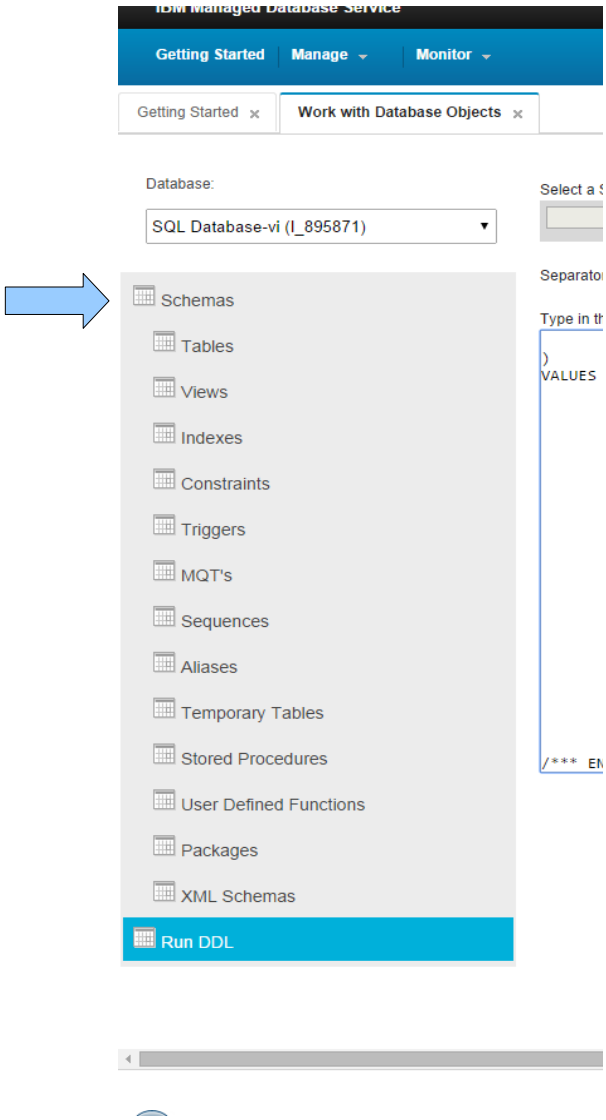
```
/*** START DB SCRIPTS ***/

CREATE TABLE PUBLIC.user_profiles (
	user_profiles_key		BIGINT NOT NULL GENERATED ALWAYS AS IDENTITY (START WITH 1, INCREMENT BY 1, NO CACHE)
	, system_create_date		TIMESTAMP NOT NULL WITH DEFAULT CURRENT TIMESTAMP
	, last_name				VARCHAR(50) NOT NULL
	, first_name			VARCHAR(50) NULL
	, email_address			VARCHAR(100) NULL
);

ALTER TABLE PUBLIC.user_profiles
	ADD CONSTRAINT pk_user_profiles PRIMARY KEY (user_profiles_key);

CREATE TABLE PUBLIC.class_activities (
	class_activities_key		BIGINT NOT NULL GENERATED ALWAYS AS IDENTITY (START WITH 1, INCREMENT BY 1, NO CACHE)
	, system_create_date		TIMESTAMP NOT NULL WITH DEFAULT CURRENT TIMESTAMP
	, activity_name				VARCHAR(50) NOT NULL
	, activity_description		VARCHAR(500) NULL
	, activity_priority			INT NULL
	, activity_due_date			TIMESTAMP NULL
	, activity_start_date		TIMESTAMP NULL
	, activity_end_date			TIMESTAMP NULL
	, is_complete				CHAR(1) NOT NULL WITH DEFAULT 'N'
);

ALTER TABLE PUBLIC.class_activities
	ADD CONSTRAINT pk_class_activities PRIMARY KEY (class_activities_key);

ALTER TABLE PUBLIC.class_activities
	ADD CONSTRAINT is_complete_value_check CHECK (is_complete IN ('Y','N'));


INSERT INTO PUBLIC.class_activities (
	activity_name, activity_description, activity_priority
)
VALUES
	(
		'Create Bluemix Account'
		, 'Register for Bluemix Account'
```

```
        , 1
    )
    , (
        'Create Java app on Bluemix'
        , 'Create a Java application on Bluemix'
        , 2
    )
    , (
        'Create database service for Java app'
        , 'Create a database service for Bluemix Java application'
        , 3
    )
    , (
        'Run Java app'
        , 'Run Java application'
        , 4
    );

/*** END DB SCRIPTS ***/
```

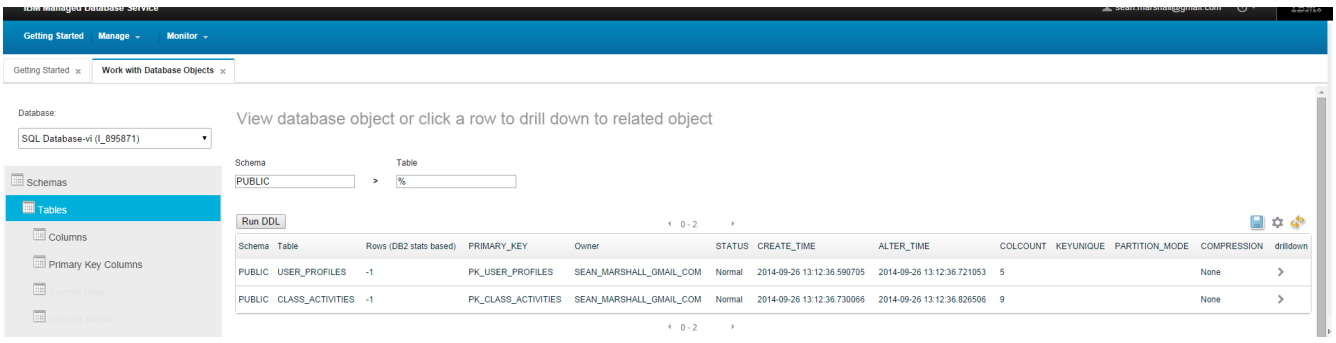
- Click the Run DDL button.
- After the scripts execute, you should receive a confirmation that the DDL ran successfully.



- Let's verify that we successfully created the database tables. Click the Schemas menu option on the left-hand menu bar.



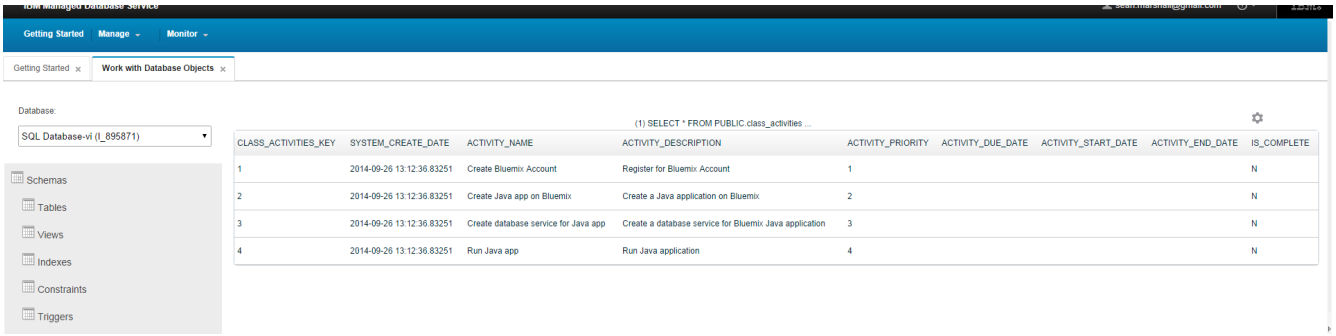
- Since we created the tables in the PUBLIC schema, click on PUBLIC under the list of schemas. You should see 2 tables: USER_PROFILES and CLASS_ACTIVITIES.



- Let's verify that some data was entered into the CLASS_ACTIVITIES table. Click the Run DDL button.
- Just like before, copy and paste the SQL script below into the text box below where it states “Type in the DDL statements or drag and drop a file.”

```
SELECT * FROM PUBLIC.class_activities;
```

- Click the Run DDL button.
- You should see 4 rows of data.



- Congratulations! You successfully created 2 database tables: a table for user profiles and a table for class activities. You also inserted some data into the class activities table.