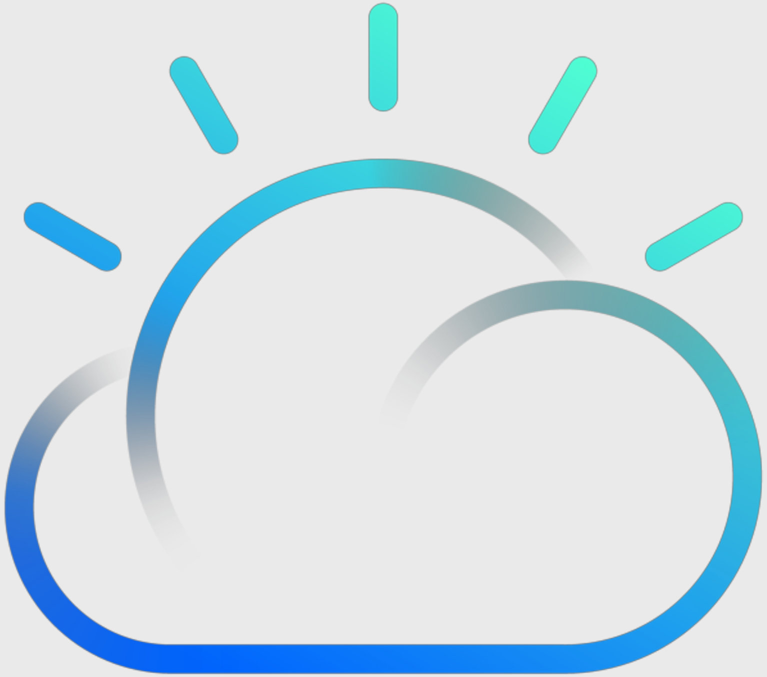
IBM Informix & Docker

Informix User Group Meeting – Atlanta
Wednesday, April 17, 2019



Pradeep Natarajan
Head of Engineering,
Informix R&D – HCL

@pradeepnatara

Disclaimers

Contents

Containerization Basics

A method to package up an application with all the dependencies and libraries needed to run the application as a single file

What is Containerization

- A method of creating a lightweight, standalone, executable package of software that includes everything needed to run an application.
- Images become containers at runtime
- Docker is not the only containerization vehicle available. But it is the most popular.

Image

An inert, immutable object that is created with the docker build command, containing your application and all dependencies required for the application.

Container

A running instance of a docker image. Multiple instances of a docker image can run at any given time.

Benefits of Containerization

- Build once ….. and run anywhere
- Portable environment
- Similar to a Virtual Machine without the overhead of a VM
- Single package with all dependencies
- Build package and store in repository for later retrieval

Dockerhub

Public docker registry where docker images can be stored.

Private Registry

A storage location for docker images that is not available to the public. EX. A private registry could be used within your company organization.

Performance of Containers

- Internal Testing was performed on Linux x86 with 8 Core and 16GB RAM
- Testing was performed with both cooked files and raw devices
- Read tests, write tests, read+write tests

Docker

Testing in a docker container we found that degradation was just 1% - 3% when compared to bare metal testing.

Virtual Machine

We did similar testing in a virtual machine environment and found that the degradation of a VM was approximately 15% when compared to bare metal.

Pull Informix Docker Images

We’ve made a handful of Informix docker images available on Dockerhub.

Informix Dockerhub Images

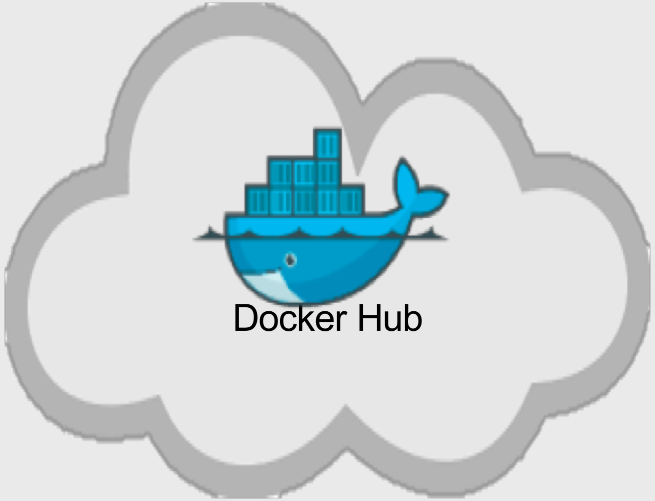
We've made the IBM Informix Innovator-C and the IBM Informix Developer Edition of Informix available on Dockerhub. We've also created a developer sandbox to help Informix developers get up and Running within minutes.

https://hub.docker.com/r/ibmcom/informix-developer-database/

https://hub.docker.com/r/ibmcom/informix-innovator-c/

https://hub.docker.com/r/ibmcom/informix-developer-sandbox/

To use these images you would use the docker pull command

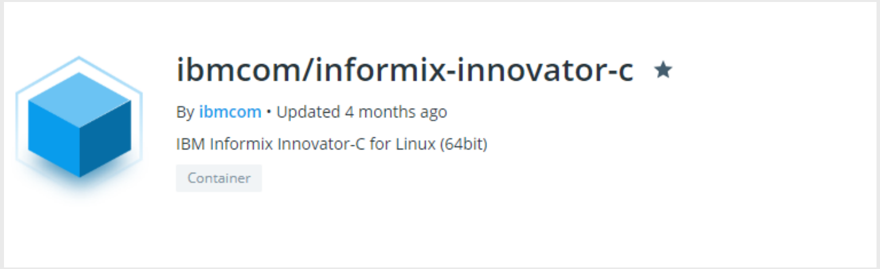


Pull Informix Server from Docker

Within a working docker environment use the docker pull command

IBM Informix Innovator-C

Free IBM Informix Database for limited production use.



ibmcom/informix-innovator-c ★

By ibmcom • Updated 4 months ago

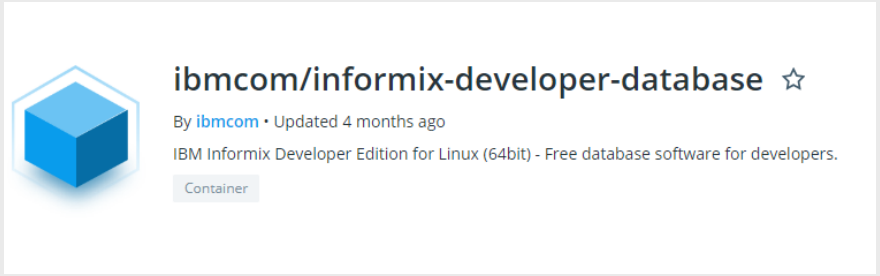
IBM Informix Innovator-C for Linux (64bit)

Container

```
$docker pull ibmcom/informix-innovator-c
```

IBM Informix Developer Edition

Fully featured IBM Informix Database free for non-production use.



ibmcom/informix-developer-database ☆

By ibmcom • Updated 4 months ago

IBM Informix Developer Edition for Linux (64bit) - Free database software for developers.

Container

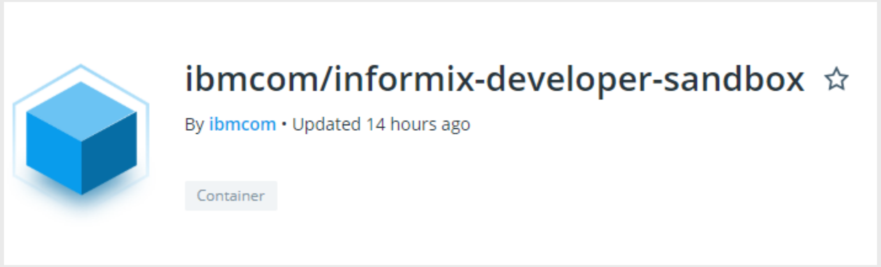
```
$docker pull ibmcom/informix-developer-database
```

Pull Informix Application Developer Sandbox from Docker Hub

Within a working docker environment use the docker pull command

Informix Developer Sandbox

A sandbox with a development environment and examples for Java, NodeJS, and Python.


ibmcom/informix-developer-sandbox

By ibmcom • Updated 14 hours ago

```
$docker pull ibmcom/informix-developer-sandbox
```

Up & Running with Informix Database

Using the IBM Informix Docker image you can have a database system up and available in minutes.

Start Informix Server Docker Image

After you have pulled the docker image. You use the following command to run the docker image.

If you haven’t already pulled the image, the initial docker run will pull the image.

Docker run

This command will run a docker container, setup connectivity ports, accept the license agreement and name the container ‘**server**’.

For more information on these options review the Dockerhub page for the Image.

```
$docker run -td --name server -e LICENSE=accept -p 9088:9088 -p 9089:9089 -p 27017:27017 -p 27018:27018 -p 27883:27883 -e TYPE=oltp  ibmcom/informix-developer-database
```

Information on the run options used

The –td starts the container as a daemon. You can then start a shell with the docker exec command and specify the name of the container to attach to.

```
$docker exec –it server bash
```

The –name server, will name this container ‘**server**’

The –e LICENSE=accept is necessary to accept the license agreement.

The –p options map the ports within the container to the host system. TCP-9088, DRDA-9089, Mongo-27017, REST-27018, MQTT-27883

The –e TYPE=oltp indicates how to configure the Informix Database.

Build you own Informix Server Docker Image

Docker

This repo contains docker files for various projects.

Folder informix-db contains the docker files to build an Informix Server docker image.

https://github.com/informix/Informix-dockerfiles

```
$git clone https://github.com/informix/informix-dockerfiles
```

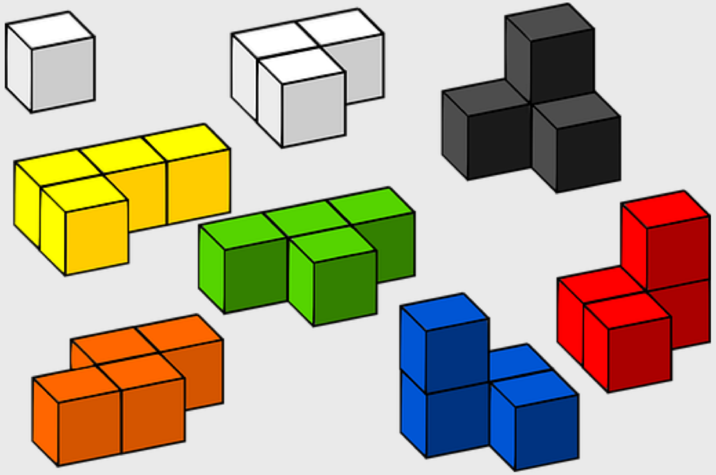
Up & Running with Informix Application Development

Using the IBM Informix Developer Sandbox Docker image you can have a development system up and running in minutes.

Building blocks

Client Sandbox includes,

- Java 1.8
- Informix JDBC
- Python 2.7
- NodeJS 8.10.0
- Informix Python Driver
- Informix NodeJS Driver
- Informix ODBC Driver



Start Informix Developer Sandbox Docker Image

After you have pulled the docker image. You use the following command to run the docker image.

If you haven't already pulled the image, the initial docker run will pull the image.

Docker run

This command will run a docker container, setup connectivity ports, accept the license agreement and name the container '**client**'.

For more information on these options review the Dockerhub page for the Image.

```
$docker run -td --name client -p 9001:9001 ibmcom/informix-developer-sandbox
```

Information on the run options used

The –td starts the container as a daemon. You can then start a shell with the docker exec command and specify the name of the container to attach to.

The –name server, will name this container ‘**client**’

The –p options map the ports within the container to the host system. Port 9001 is needed for a demo needing http access.

```
$docker exec –it client bash
```

Application Development Examples

Using the IBM Informix Docker image you can have a database system up and available in minutes.

Java Examples – Initial setup

1. Initial setup

   - Login to the client Docker container
     - (docker exec –it client bash)
   - cd informix-db-examples/java
   - chmod 777 gradlew
   - ./gradlew clean jar

`https://github.com/informix/informix-db-examples/tree/master/java`

Java Example – Create database

2. Run the following to create a database for use with the examples.

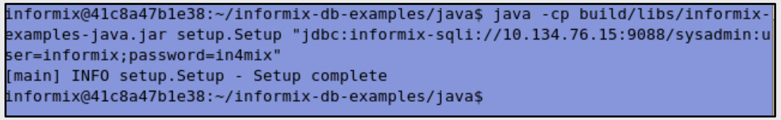
```
$java –cp build/libs/informix-examples-java.jar setup.Setup
"jdbc:informix-
sqli://IPADDRESS:9088/sysadmin:user=informix;password=in4mix"
```

- This example will create a database named **banktest**. This can be used for the other examples. And will be used for the Demo program referred to later on.

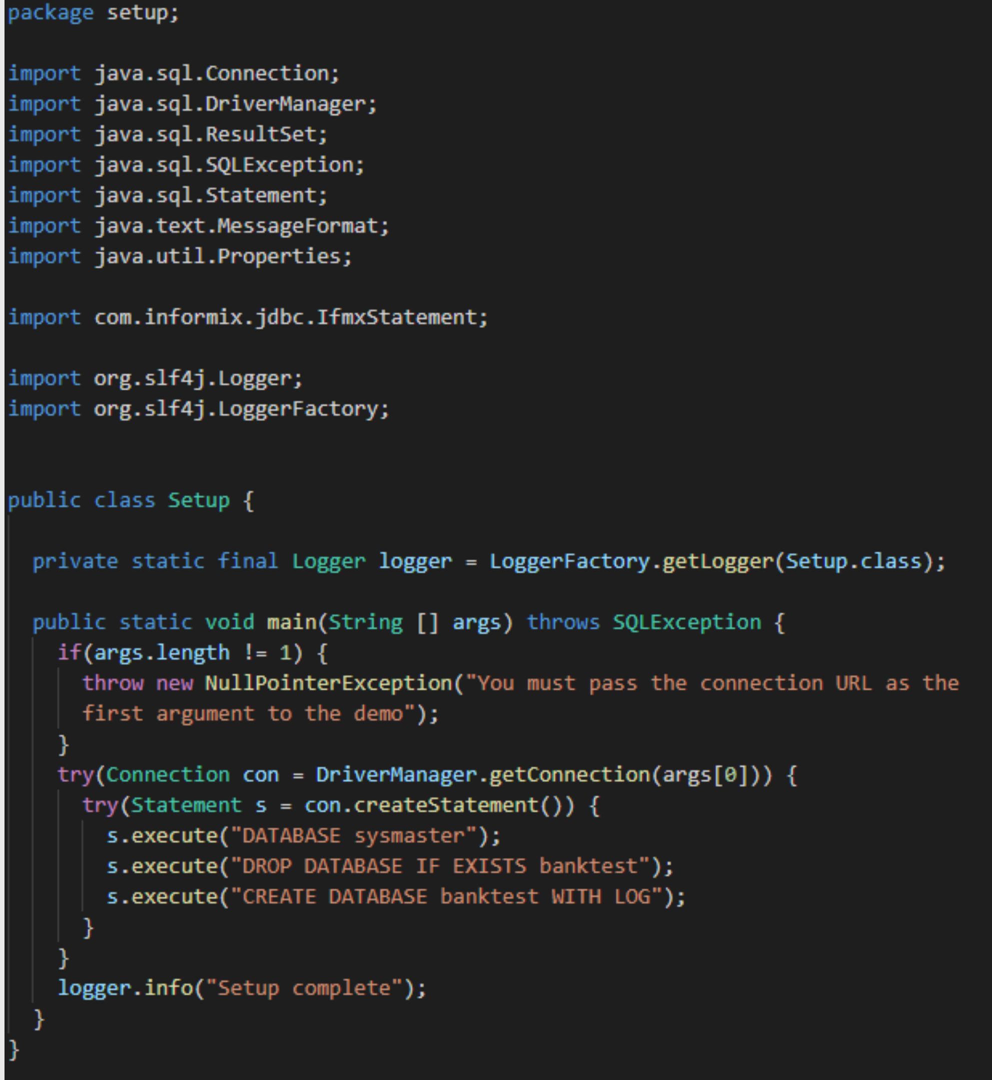
```
https://github.com/informix/informix-db-examples/tree/master/java
```

Java Example – Code - Create database

- src/main/java/setup/Setup.java



```
informix@41c8a47b1e38:~/informix-db-examples/java$ java -cp build/libs/informix-
examples-java.jar setup.Setup "jdbc:informix-sqli://10.134.76.15:9088/sysadmin:u
ser=informix;password=in4mix"
[main] INFO setup.Setup - Setup complete
informix@41c8a47b1e38:~/informix-db-examples/java$
```



```java
package setup;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.text.MessageFormat;
import java.util.Properties;

import com.informix.jdbc.IfmxStatement;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;


public class Setup {

  private static final Logger logger = LoggerFactory.getLogger(Setup.class);

  public static void main(String [] args) throws SQLException {
    if(args.length != 1) {
      throw new NullPointerException("You must pass the connection URL as the
      first argument to the demo");
    }
    try(Connection con = DriverManager.getConnection(args[0])) {
      try(Statement s = con.createStatement()) {
        s.execute("DATABASE sysmaster");
        s.execute("DROP DATABASE IF EXISTS banktest");
        s.execute("CREATE DATABASE banktest WITH LOG");
      }
    }
    logger.info("Setup complete");
  }
}
```

Java Example – BSON Example

3. Run the following to BSON example.

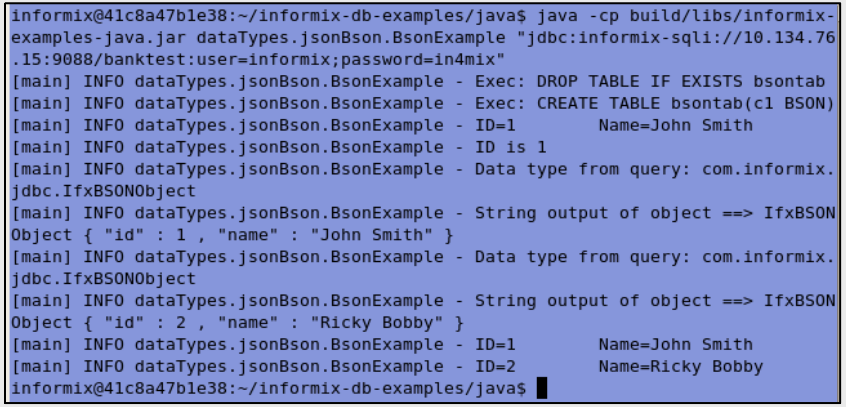
```
$java –cp build/libs/informix-examples-java.jar
dataTypes.jsonBson.BsonExample “jdbc:informix-
sqli://IPADDRESS:9088/banktest:user=informix;password=in4mix”
```

- This example will create a table named **bsonTab**. It creates a bson object and performs a couple inserts using different options.

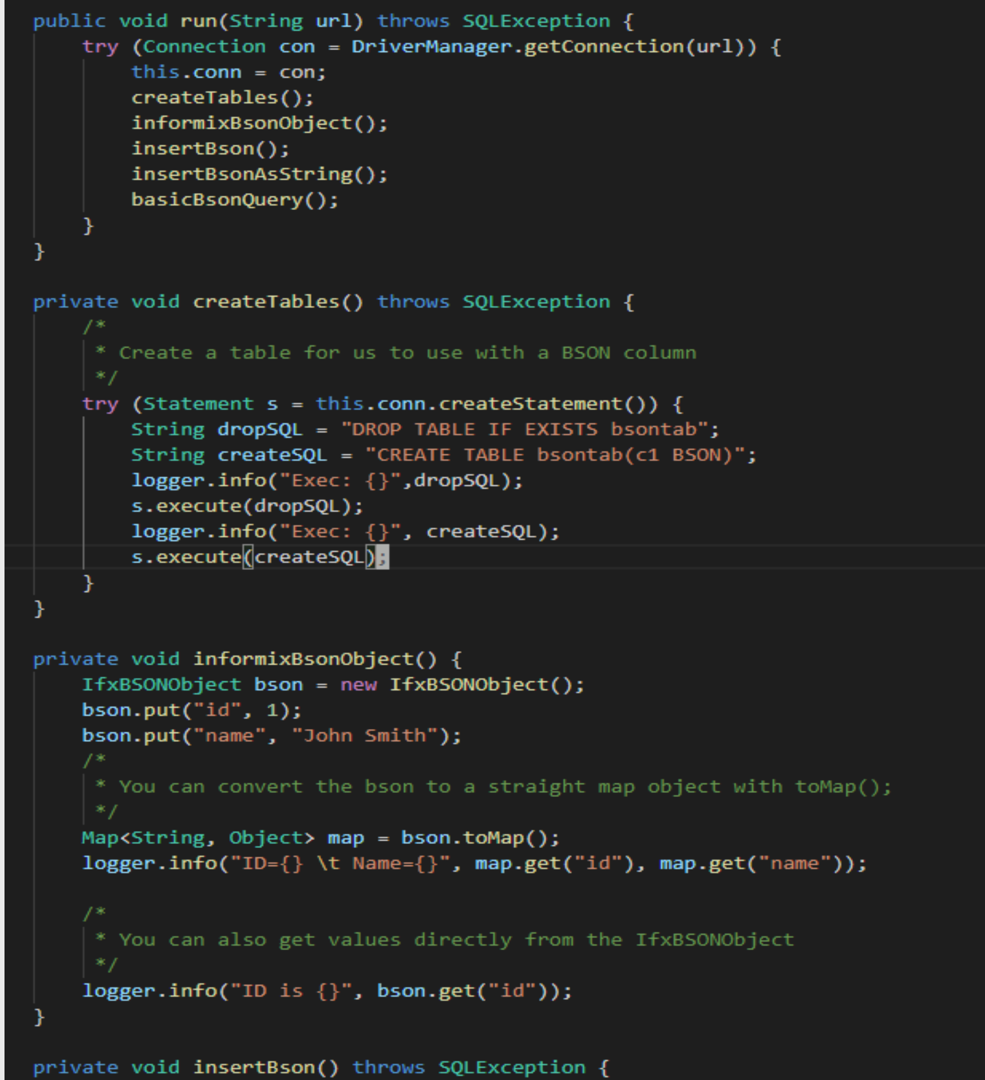
```
https://github.com/informix/informix-db-examples/tree/master/java
```

Java Example – Code – BSON example

- src/main/java/dataTypes/jsonBson/BsonExample.java



```
informix@41c8a47b1e38:~/informix-db-examples/java$ java -cp build/libs/informix-
examples-java.jar dataTypes.jsonBson.BsonExample "jdbc:informix-sqli://10.134.76
.15:9088/banktest:user=informix;password=in4mix"
[main] INFO dataTypes.jsonBson.BsonExample - Exec: DROP TABLE IF EXISTS bsontab
[main] INFO dataTypes.jsonBson.BsonExample - Exec: CREATE TABLE bsontab(c1 BSON)
[main] INFO dataTypes.jsonBson.BsonExample - ID=1        Name=John Smith
[main] INFO dataTypes.jsonBson.BsonExample - ID is 1
[main] INFO dataTypes.jsonBson.BsonExample - Data type from query: com.informix.
jdbc.IfxBSONObject
[main] INFO dataTypes.jsonBson.BsonExample - String output of object ==> IfxBSON
Object { "id" : 1 , "name" : "John Smith" }
[main] INFO dataTypes.jsonBson.BsonExample - Data type from query: com.informix.
jdbc.IfxBSONObject
[main] INFO dataTypes.jsonBson.BsonExample - String output of object ==> IfxBSON
Object { "id" : 2 , "name" : "Ricky Bobby" }
[main] INFO dataTypes.jsonBson.BsonExample - ID=1        Name=John Smith
[main] INFO dataTypes.jsonBson.BsonExample - ID=2        Name=Ricky Bobby
informix@41c8a47b1e38:~/informix-db-examples/java$ 
```



```java
public void run(String url) throws SQLException {
    try (Connection con = DriverManager.getConnection(url)) {
        this.conn = con;
        createTables();
        informixBsonObject();
        insertBson();
        insertBsonAsString();
        basicBsonQuery();
    }
}

private void createTables() throws SQLException {
    /*
     * Create a table for us to use with a BSON column
     */
    try (Statement s = this.conn.createStatement()) {
        String dropSQL = "DROP TABLE IF EXISTS bsontab";
        String createSQL = "CREATE TABLE bsontab(c1 BSON)";
        logger.info("Exec: {}",dropSQL);
        s.execute(dropSQL);
        logger.info("Exec: {}", createSQL);
        s.execute(createSQL);
    }
}

private void informixBsonObject() {
    IfxBSONObject bson = new IfxBSONObject();
    bson.put("id", 1);
    bson.put("name", "John Smith");
    /*
     * You can convert the bson to a straight map object with toMap();
     */
    Map<String, Object> map = bson.toMap();
    logger.info("ID={} \t Name={}", map.get("id"), map.get("name"));

    /*
     * You can also get values directly from the IfxBSONObject
     */
    logger.info("ID is {}", bson.get("id"));
}

private void insertBson() throws SQLException {
```

Java Example – SmartTrigger example

4. Run the following to SmartTrigger example.

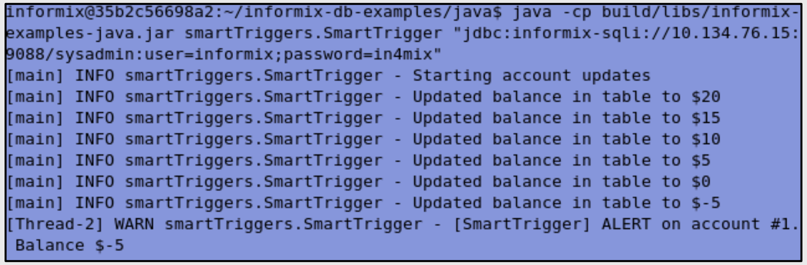
```
$java –cp build/libs/informix-examples-java.jar
smartTriggers.SmartTrigger "jdbc:informix-
sqli://IPADDRESS:9088/sysadmin:user=informix;password=in4mix"
```

- This example will create a smart Trigger on an account table. It will periodically update the account balance, decreasing the total. When the account goes below 0 an Alert will trigger on the update.
  - Important to use connect to sysadmin for this example

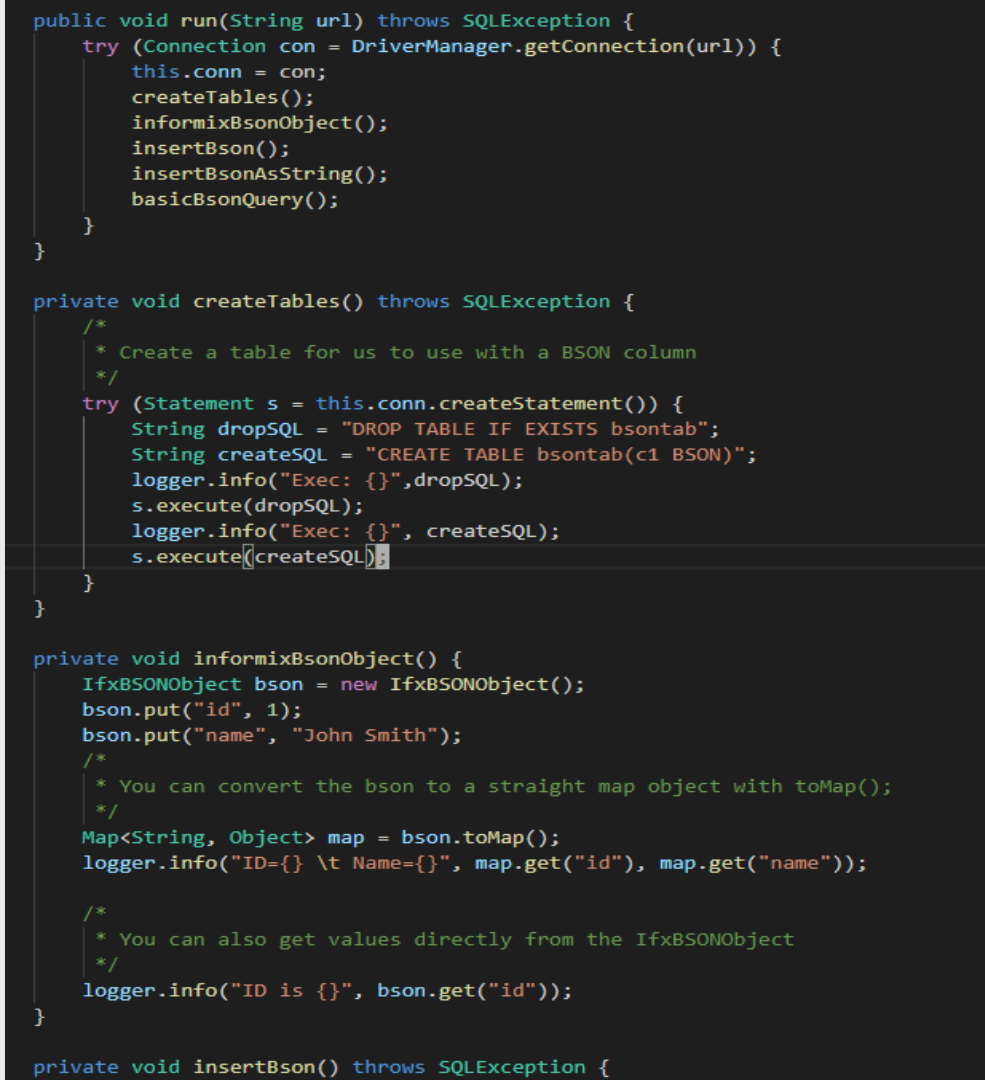
```
https://github.com/informix/informix-db-examples/tree/master/java
```

Java Example – Code – SmartTrigger example

- src/main/java/smartTriggers/SmartTrigger.java



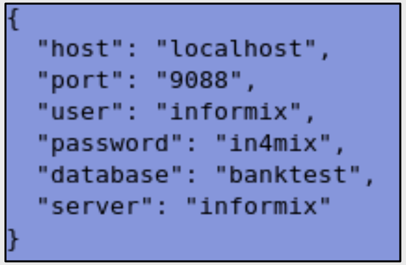
```
informix@35b2c56698a2:~/informix-db-examples/java$ java -cp build/libs/informix-
examples-java.jar smartTriggers.SmartTrigger "jdbc:informix-sqli://10.134.76.15:
9088/sysadmin:user=informix;password=in4mix"
[main] INFO smartTriggers.SmartTrigger - Starting account updates
[main] INFO smartTriggers.SmartTrigger - Updated balance in table to $20
[main] INFO smartTriggers.SmartTrigger - Updated balance in table to $15
[main] INFO smartTriggers.SmartTrigger - Updated balance in table to $10
[main] INFO smartTriggers.SmartTrigger - Updated balance in table to $5
[main] INFO smartTriggers.SmartTrigger - Updated balance in table to $0
[main] INFO smartTriggers.SmartTrigger - Updated balance in table to $-5
[Thread-2] WARN smartTriggers.SmartTrigger - [SmartTrigger] ALERT on account #1.
 Balance $-5
```



```java
public void run(String url) throws SQLException {
    try (Connection con = DriverManager.getConnection(url)) {
        this.conn = con;
        createTables();
        informixBsonObject();
        insertBson();
        insertBsonAsString();
        basicBsonQuery();
    }
}

private void createTables() throws SQLException {
    /*
     * Create a table for us to use with a BSON column
     */
    try (Statement s = this.conn.createStatement()) {
        String dropSQL = "DROP TABLE IF EXISTS bsontab";
        String createSQL = "CREATE TABLE bsontab(c1 BSON)";
        logger.info("Exec: {}",dropSQL);
        s.execute(dropSQL);
        logger.info("Exec: {}", createSQL);
        s.execute(createSQL);
    }
}

private void informixBsonObject() {
    IfxBSONObject bson = new IfxBSONObject();
    bson.put("id", 1);
    bson.put("name", "John Smith");
    /*
     * You can convert the bson to a straight map object with toMap();
     */
    Map<String, Object> map = bson.toMap();
    logger.info("ID={} \t Name={}", map.get("id"), map.get("name"));

    /*
     * You can also get values directly from the IfxBSONObject
     */
    logger.info("ID is {}", bson.get("id"));
}

private void insertBson() throws SQLException {
```

Python Examples – Initial setup

1. Initial setup
   - Login to the client Docker container
     - (docker exec –it client bash)
   - cd informix-db-examples
   - Modify connections.json accordingly. Set the host in the connections.json to your IP address of the HOST where the docker containers are running
   - cd python



```
{
  "host": "localhost",
  "port": "9088",
  "user": "informix",
  "password": "in4mix",
  "database": "banktest",
  "server": "informix"
}
```

https://github.com/informix/informix-db-examples/tree/master/python

Python Examples – basicQuery example

2. Run basicQuery.py

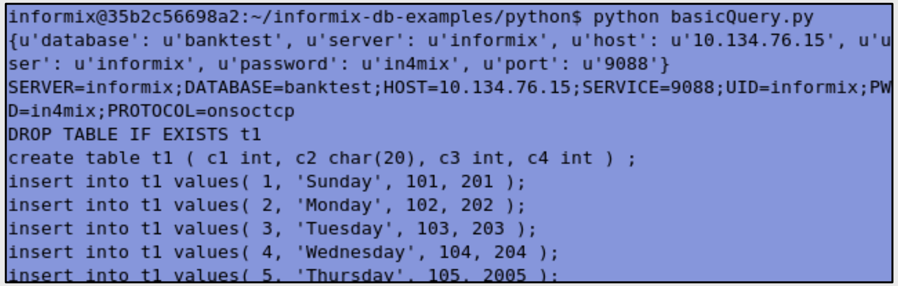
```
$python basicQuery.py
```

- This example will create a table named **t1**, insert some data and query the data.

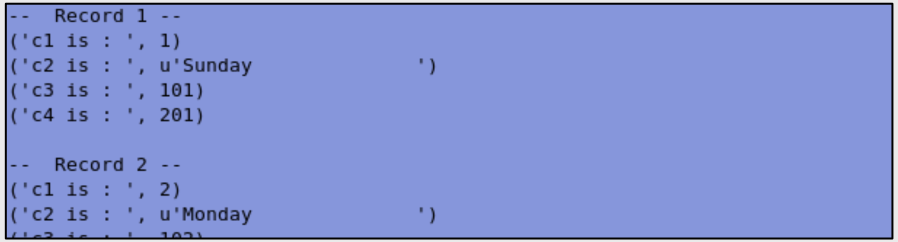
```
https://github.com/informix/informix-db-examples/tree/master/python
```

Python Example – Code – basicQuery example

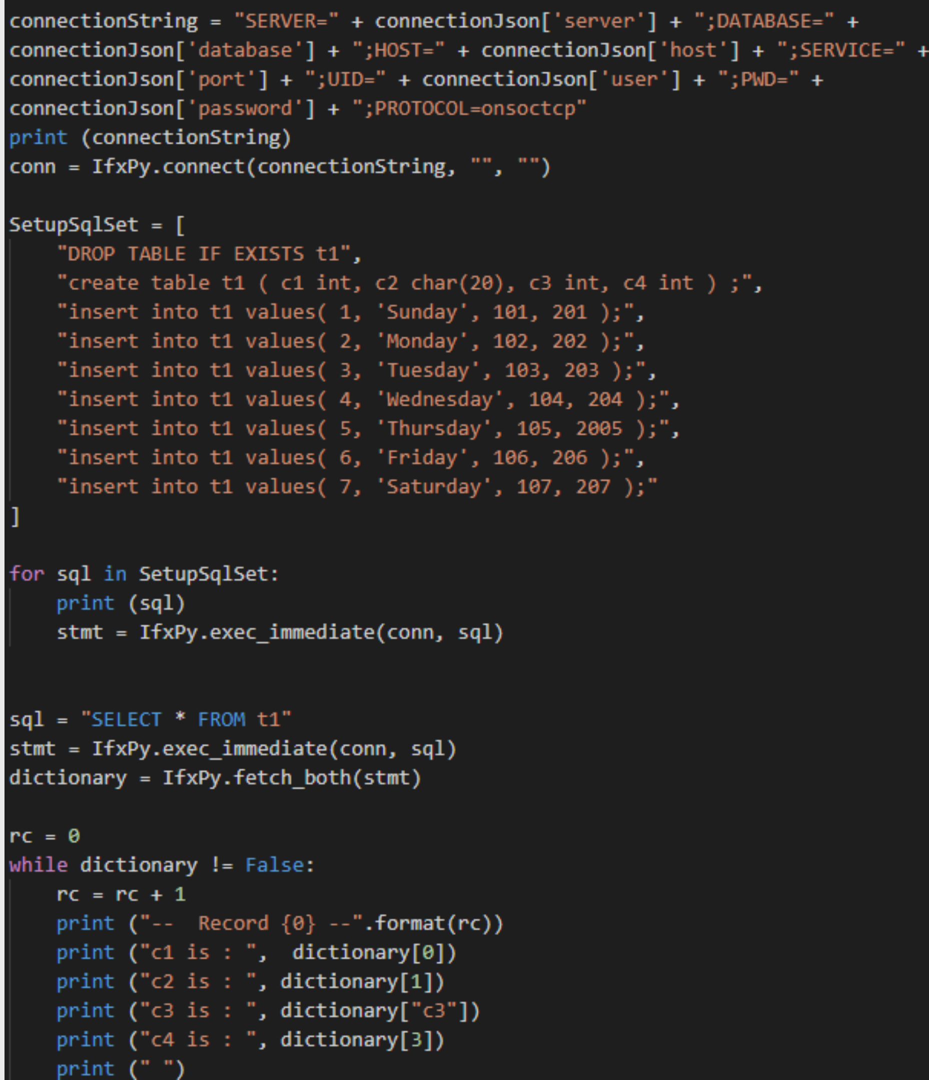
- basicQuery.py



```
informix@35b2c56698a2:~/informix-db-examples/python$ python basicQuery.py
{u'database': u'banktest', u'server': u'informix', u'host': u'10.134.76.15', u'u
ser': u'informix', u'password': u'in4mix', u'port': u'9088'}
SERVER=informix;DATABASE=banktest;HOST=10.134.76.15;SERVICE=9088;UID=informix;PW
D=in4mix;PROTOCOL=onsoctcp
DROP TABLE IF EXISTS t1
create table t1 ( c1 int, c2 char(20), c3 int, c4 int ) ;
insert into t1 values( 1, 'Sunday', 101, 201 );
insert into t1 values( 2, 'Monday', 102, 202 );
insert into t1 values( 3, 'Tuesday', 103, 203 );
insert into t1 values( 4, 'Wednesday', 104, 204 );
insert into t1 values( 5, 'Thursday', 105, 2005 );
```



```
--  Record 1 --
('c1 is : ', 1)
('c2 is : ', u'Sunday              ')
('c3 is : ', 101)
('c4 is : ', 201)

--  Record 2 --
('c1 is : ', 2)
('c2 is : ', u'Monday              ')
```

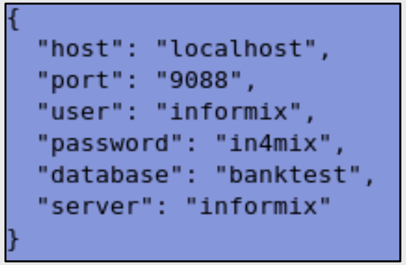


```python
connectionString = "SERVER=" + connectionJson['server'] + ";DATABASE=" +
connectionJson['database'] + ";HOST=" + connectionJson['host'] + ";SERVICE=" +
connectionJson['port'] + ";UID=" + connectionJson['user'] + ";PWD=" +
connectionJson['password'] + ";PROTOCOL=onsoctcp"
print (connectionString)
conn = IfxPy.connect(connectionString, "", "")

SetupSqlSet = [
    "DROP TABLE IF EXISTS t1",
    "create table t1 ( c1 int, c2 char(20), c3 int, c4 int ) ;",
    "insert into t1 values( 1, 'Sunday', 101, 201 );",
    "insert into t1 values( 2, 'Monday', 102, 202 );",
    "insert into t1 values( 3, 'Tuesday', 103, 203 );",
    "insert into t1 values( 4, 'Wednesday', 104, 204 );",
    "insert into t1 values( 5, 'Thursday', 105, 2005 );",
    "insert into t1 values( 6, 'Friday', 106, 206 );",
    "insert into t1 values( 7, 'Saturday', 107, 207 );"
]

for sql in SetupSqlSet:
    print (sql)
    stmt = IfxPy.exec_immediate(conn, sql)


sql = "SELECT * FROM t1"
stmt = IfxPy.exec_immediate(conn, sql)
dictionary = IfxPy.fetch_both(stmt)

rc = 0
while dictionary != False:
    rc = rc + 1
    print ("--  Record {0} --".format(rc))
    print ("c1 is : ",  dictionary[0])
    print ("c2 is : ", dictionary[1])
    print ("c3 is : ", dictionary["c3"])
    print ("c4 is : ", dictionary[3])
    print (" ")
```

NodeJS Examples – Initial setup

1. Initial setup

- Login to the client Docker container
  - (docker exec –it client bash)
- cd informix-db-examples
- Modify connections.json accordingly. Set the host in the connections.json to your IP address of the HOST where the docker containers are running
- cd nodejs



```
{
  "host": "localhost",
  "port": "9088",
  "user": "informix",
  "password": "in4mix",
  "database": "banktest",
  "server": "informix"
}
```

https://github.com/informix/informix-db-examples/tree/master/nodejs

NodeJS Examples – basicQuery example

2. Run basicQuery.js

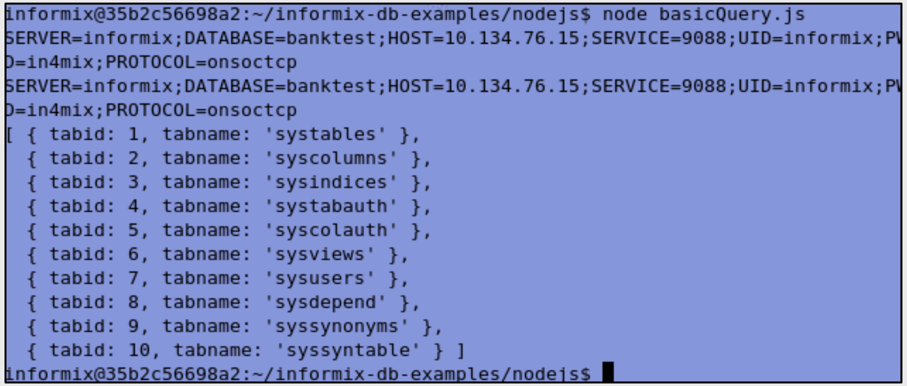
```
$node basicQuery.js
```

- This example will query the first 10 rows from systables.

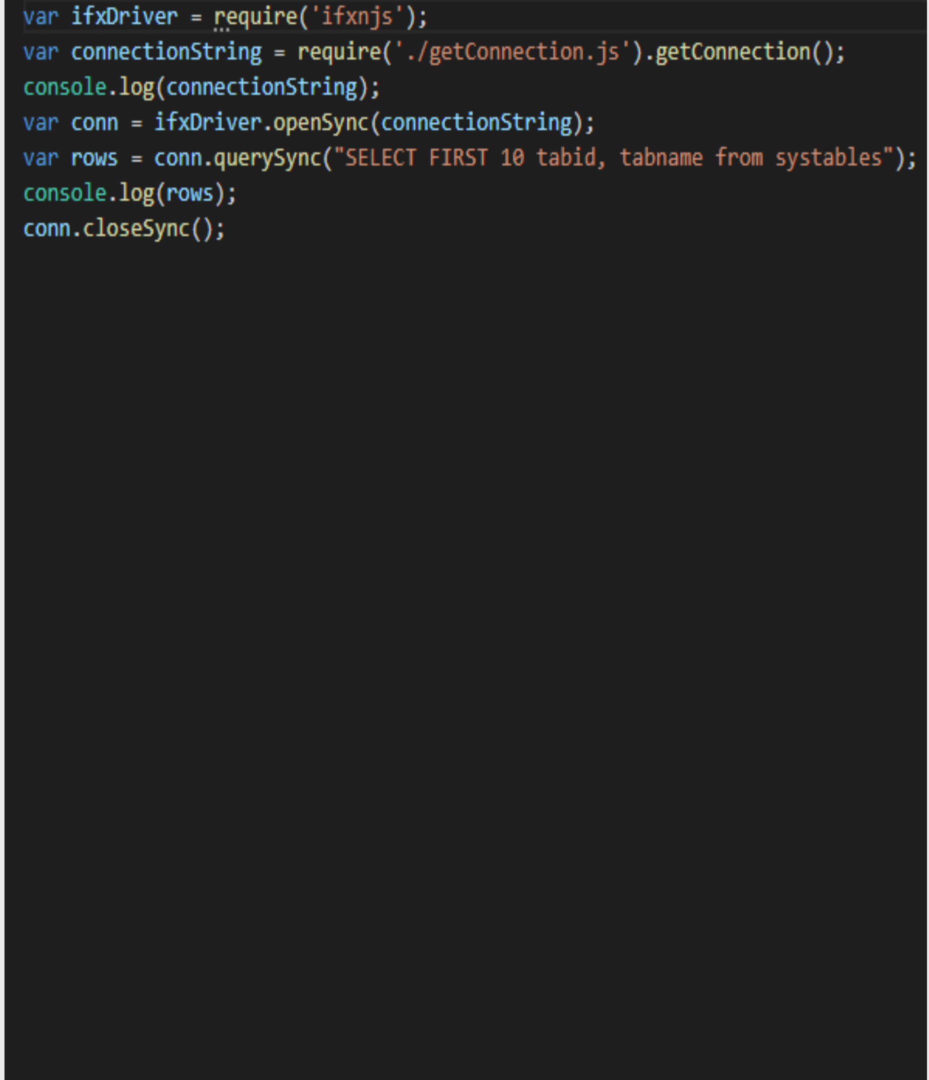
```
https://github.com/informix/informix-db-examples/tree/master/nodejs
```

NodeJS Example – Code – basicQuery example

- basicQuery.js



```
informix@35b2c56698a2:~/informix-db-examples/nodejs$ node basicQuery.js
SERVER=informix;DATABASE=banktest;HOST=10.134.76.15;SERVICE=9088;UID=informix;PW
D=in4mix;PROTOCOL=onsoctcp
SERVER=informix;DATABASE=banktest;HOST=10.134.76.15;SERVICE=9088;UID=informix;PW
D=in4mix;PROTOCOL=onsoctcp
[ { tabid: 1, tabname: 'systables' },
  { tabid: 2, tabname: 'syscolumns' },
  { tabid: 3, tabname: 'sysindices' },
  { tabid: 4, tabname: 'systabauth' },
  { tabid: 5, tabname: 'syscolauth' },
  { tabid: 6, tabname: 'sysviews' },
  { tabid: 7, tabname: 'sysusers' },
  { tabid: 8, tabname: 'sysdepend' },
  { tabid: 9, tabname: 'syssynonyms' },
  { tabid: 10, tabname: 'syssyntable' } ]
informix@35b2c56698a2:~/informix-db-examples/nodejs$
```



```
var ifxDriver = require('ifxnjs');
var connectionString = require('./getConnection.js').getConnection();
console.log(connectionString);
var conn = ifxDriver.openSync(connectionString);
var rows = conn.querySync("SELECT FIRST 10 tabid, tabname from systables");
console.log(rows);
conn.closeSync();
```

Thank You

Questions

@pradeepnatara