SOLR 6.1

# Installation

Download the zip file from the following location and extract it to a direcctory.

http://redrockdigimark.com/apachemirror/lucene/solr/6.1.0

# How to Start SOLR in foreground mode

Go to the directory, in this case, my directory is given below, ***E:\devsoftwares\solr-6.1.0\bin***.

go to the command prompt in the above directory and run the following command.

**solr start -f**

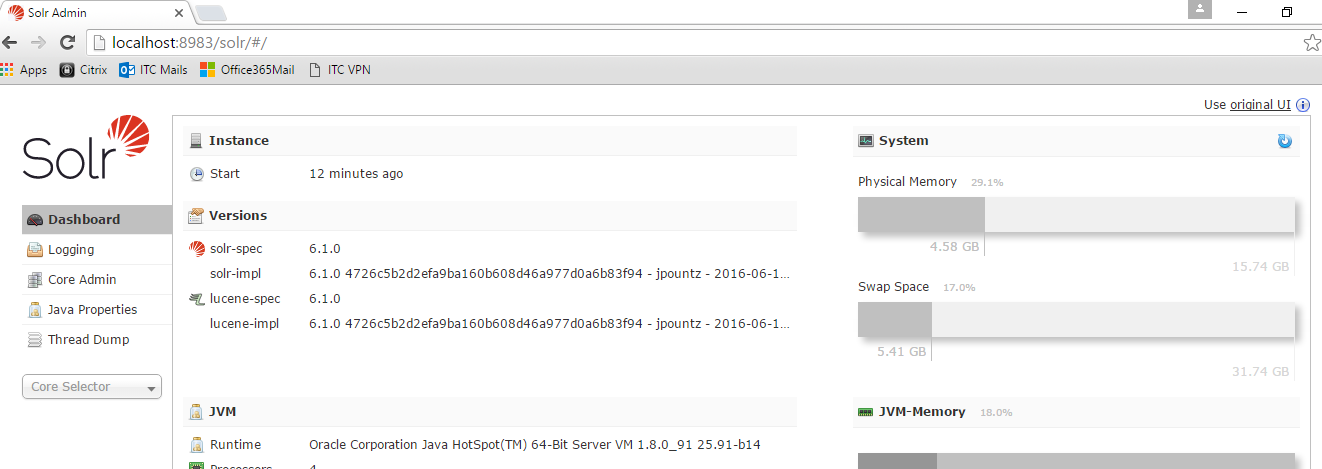
Solr can also be started like “**solr start**”. It will start in background mode.

Open your browser and go to the url

**http://localhost:8983/solr**

The default port for SOLR is 8983. However SOLR can also be started by providing port number.

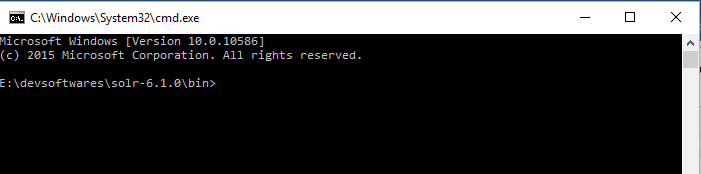
If everything is fine, the SOLR UI looks like the below.



# How to create a new SOLR CORE

**Basic way of creating SOLR CORE**

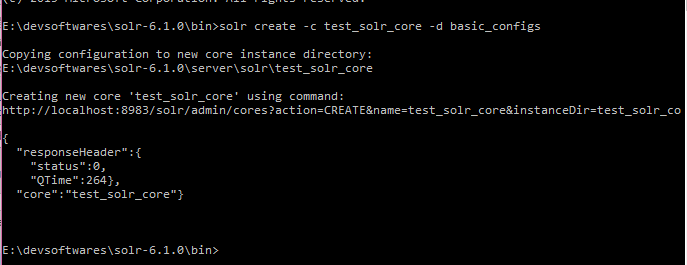
Go to the bin directory of SOLR installation folder and open the command prompt as shown below.



Now type the following command. This is used for the first time to create a basic core in SOLR.

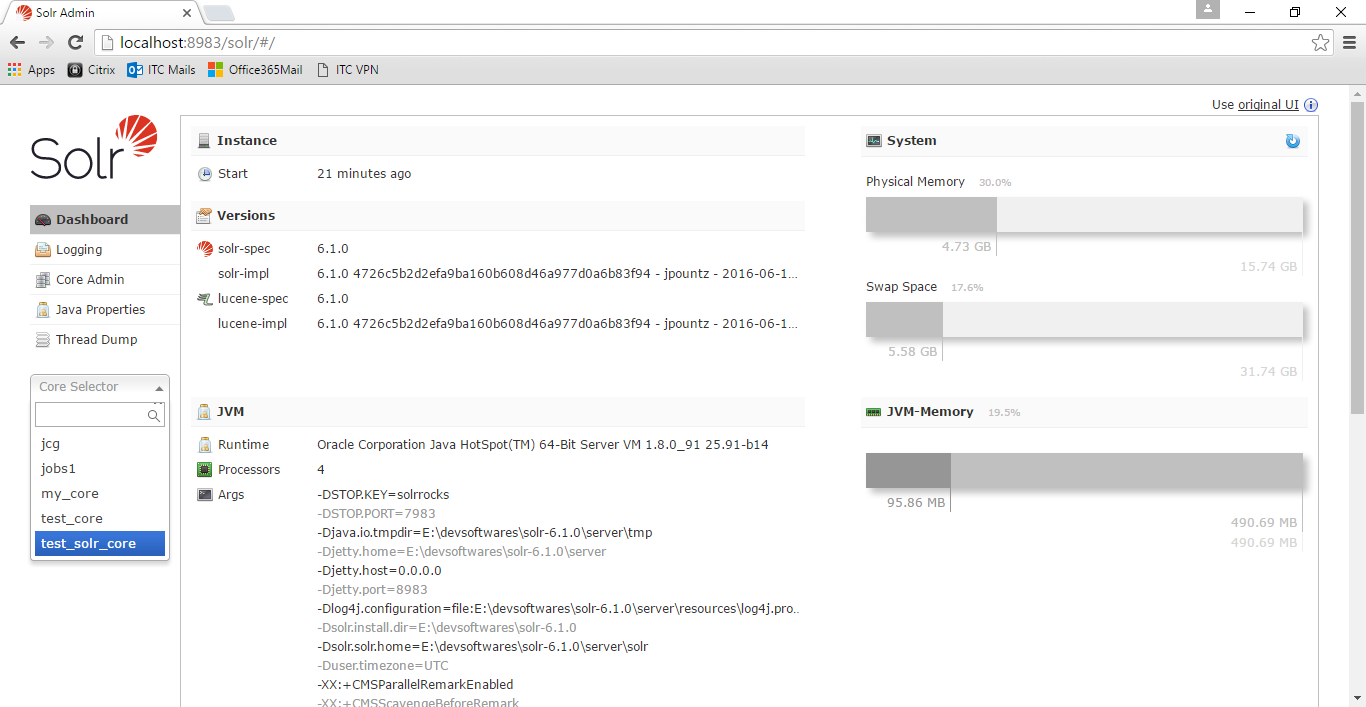
“**solr create -c test\_solr\_core -d basic\_configs**”

You will get the following response from the command prompt as shown below.



Now refresh the page displayer in your browser.

Now you can see the following screen.



# How to create push data in SOLR Core

We are able to create a solr core in the system, now let us push some data in SOLR Indexing system.

Let us push the following data in solr system.

{

"id":"1",

"title\_s":"The Way of Kings",

"author\_s":"Brandon Sanderson"

}

Why always \_s in the fields ?

Because of the following definitions in schema.xml

<dynamicField name="\*\_i" type="int" indexed="true" stored="true"/>

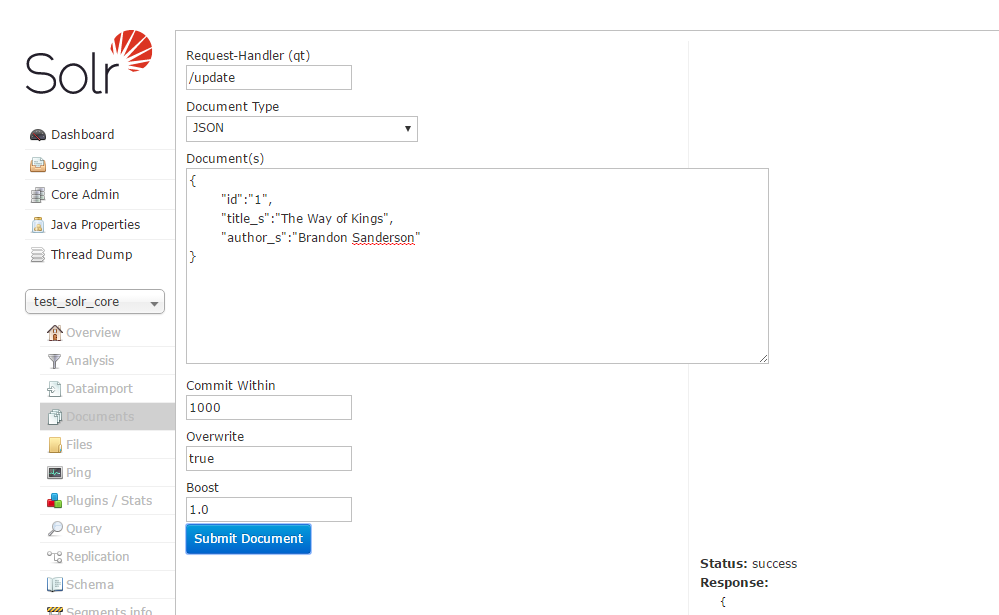
<dynamicField name="\*\_is" type="int" indexed="true" stored="true" multiValued="true"/>

<dynamicField name="\*\_s" type="string" indexed="true" stored="true" />

<dynamicField name="\*\_ss" type="string" indexed="true" stored="true" multiValued="true"/>

<dynamicField name="\*\_l" type="long" indexed="true" stored="true"/>

SOLR UI Console provides a flexible way to push the data. The screenshot is given below.



You can also make the following rest call to update the data in solr system using Firefox REST client.

POST

<http://localhost:8983/solr/test_solr_core/update?commitWithin=1000&overwrite=true&wt=json>

**Request Body**

[{

"id":"2",

"title\_s":"Death in Sky",

"author\_s":"Agatha Cristie"

}]

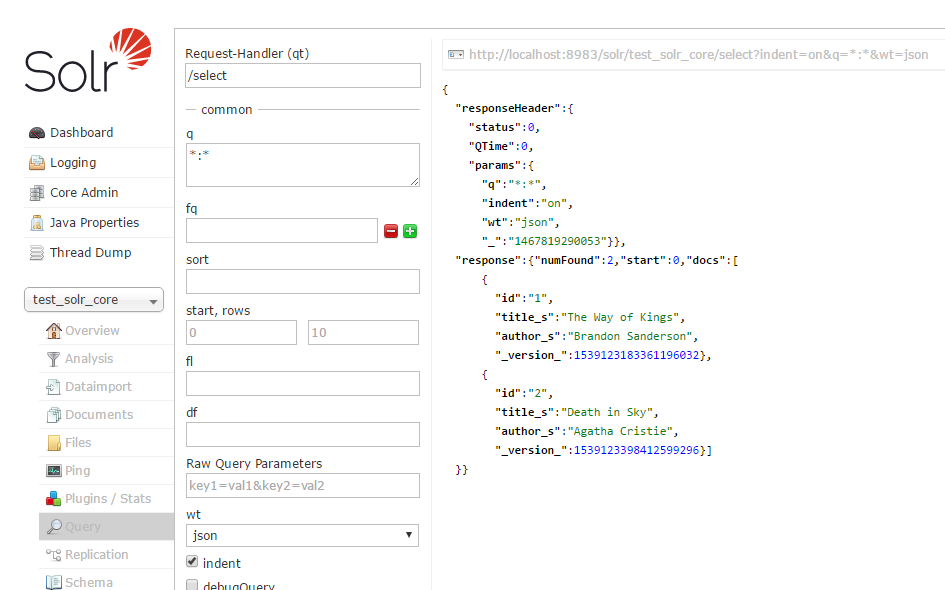
Request Headers

Content-Type : application/json

Accept: application/json

# How to see the data in SOLR System

First of all select the core ie “test\_solr\_core” and go to the query link and click the button “Execute Query”



How to update a bunch of JSON Data in SOLR System

The JSON data is given below.

{

"id":"3",

"title\_s":"Book-1",

"author\_s":"Deb"

},

{

"id":"4",

"title\_s":"My Life",

"author\_s":"DD Mishra"

},

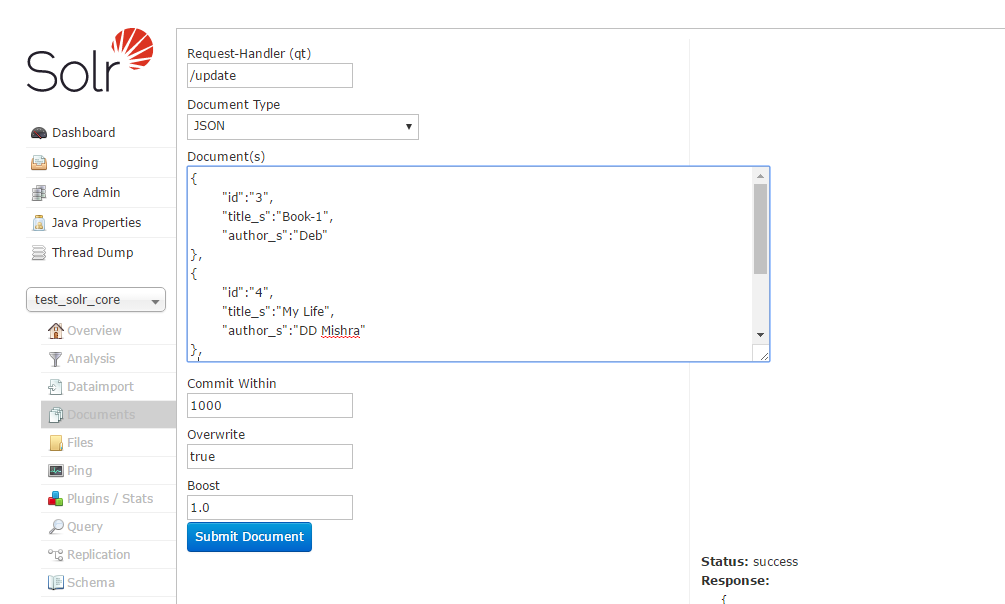
{

"id":"5",

"title\_s":"Changing Life",

"author\_s":"Debadatta Mishra"

}

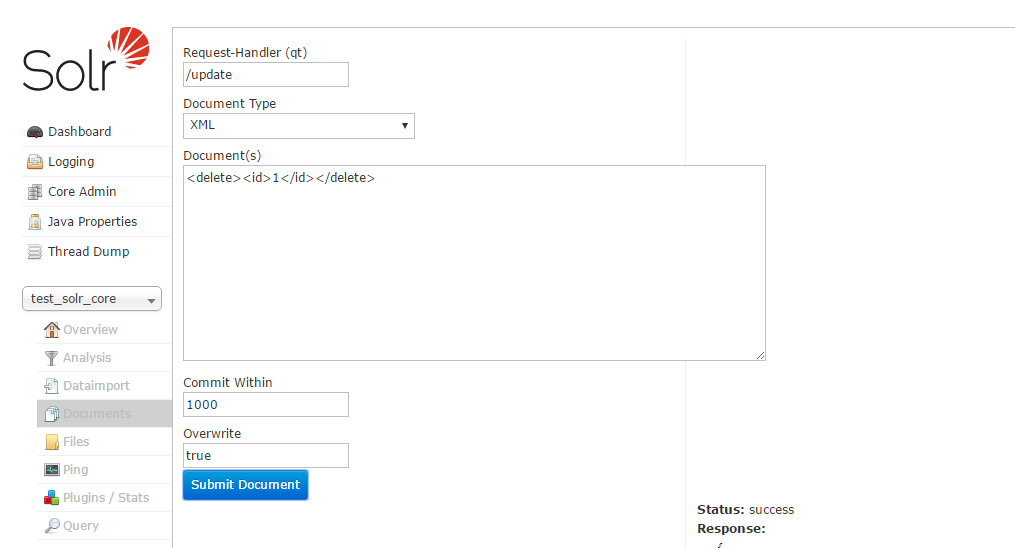


Similarly, check the network call in Google Chrome and make rest call accordingly.

# How to Delete data from SOLR

You can delete the data in XML as shown below.

<delete><id>1</id></delete>



How to delete data in Json format. This is bit tricky one.

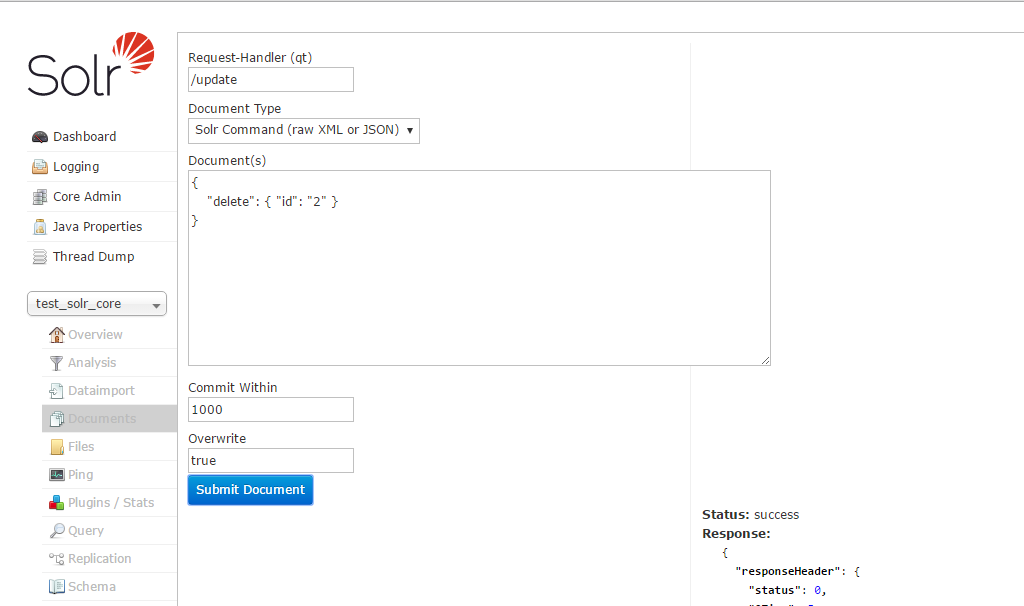
The json is given below. Solr provides to delete by Id.

**{**

**"delete": { "id": "2" }**

**}**

In solr console, you have to select “**Solr Command (raw XML and JSON)**”. The screenshot is given below.



To delete all the data, the following query in XML format

**<delete><query>\*:\*</query></delete>**

How to update a document in SOLR

The json data to be updated is given below.

{

"id":"3",

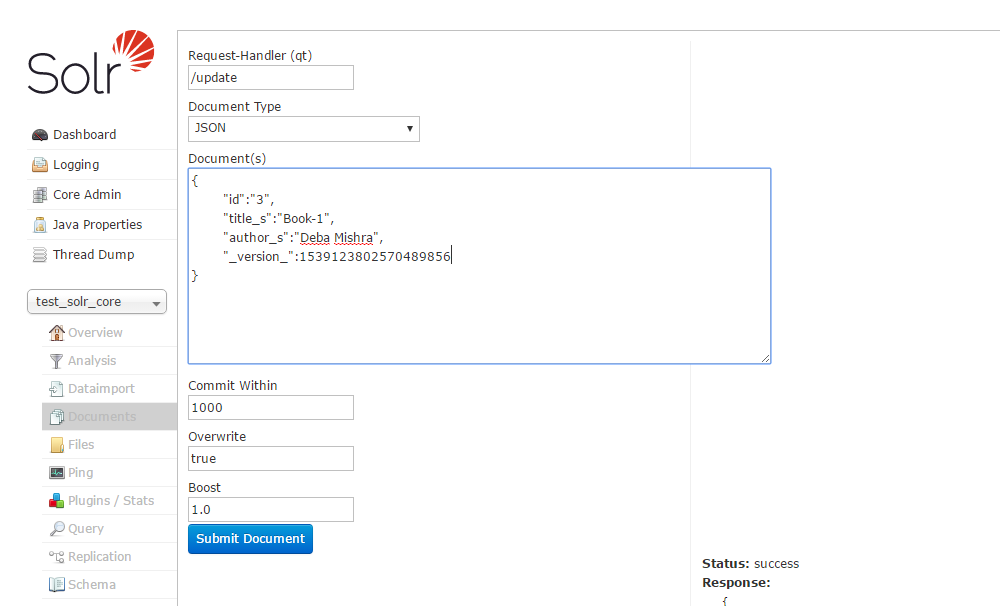
"title\_s":"Book-1",

"author\_s":"Deba Mishra",

"\_version\_":1539123802570489856

}

The screen shot is given below.



Managed-schema File Details

<field name="id" type="string" indexed="true" stored="true" required="true" multiValued="false" />

<field name="title\_s" type="string" indexed="true" stored="true" />

<!-- New Additions of Fields for the required pojo -->

<field name="fname" type="string" indexed="true" stored="true" />

<field name="lname" type="string" indexed="true" stored="true" />

<field name="age" type="int" indexed="true" stored="true" />

<field name="streetNo" type="string" indexed="true" stored="true" />

<field name="city" type="string" indexed="true" stored="true" />

<field name="country" type="string" indexed="true" stored="true" />

<field name="emailIds" type="text\_general" indexed="true" stored="true" multiValued="true"/>

<field name="salary" type="int" indexed="true" stored="true" />

# SOLRJ Usage

All kinds of CRUD operation is given below using **solrJ** .

**Maven Configuration**

<project xmlns=*"http://maven.apache.org/POM/4.0.0"* xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd"*>

<modelVersion>4.0.0</modelVersion>

<groupId>solr-core-dataimport</groupId>

<artifactId>solr-core-dataimport</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<name>solr-core-dataimport</name>

<url>http://maven.apache.org</url>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

</properties>

<dependencies>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>3.8.1</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.apache.solr</groupId>

<artifactId>solr-solrj</artifactId>

<version>6.1.0</version>

</dependency>

<dependency>

<groupId>org.apache.solr</groupId>

<artifactId>solr-core</artifactId>

<version>6.1.0</version>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.2</version>

<configuration>

<source>1.8</source>

<target>1.8</target>

</configuration>

</plugin>

</plugins>

</build>

</project>

**AddressBean.java**

**package** com.ddlab.rnd.solr;

**import** org.apache.solr.client.solrj.beans.Field;

/\*\*

\* Created by PIKU on 7/9/2016.

\*/

**public** **class** AddressBean {

@Field("streetNo")

**private** String streetNo;

@Field("city")

**private** String city;

@Field("country")

**private** String country;

**public** String getCity() {

**return** city;

}

**public** **void** setCity(String city) {

**this**.city = city;

}

**public** String getStreetNo() {

**return** streetNo;

}

**public** **void** setStreetNo(String streetNo) {

**this**.streetNo = streetNo;

}

**public** String getCountry() {

**return** country;

}

**public** **void** setCountry(String country) {

**this**.country = country;

}

}

**EmpBean.java**

**package** com.ddlab.rnd.solr;

**import** org.apache.solr.client.solrj.beans.Field;

**import** java.util.Arrays;

/\*\*

\* Created by PIKU on 7/9/2016.

\*/

**public** **class** EmpBean **extends** AddressBean {

@Field("id")

**private** String id;

@Field("fname")

**private** String firstName;

@Field("lname")

**private** String lastName;

@Field("age")

**private** **int** age;

@Field("emailIds")

**private** String[] emailIds;

@Field("salary")

**private** **int** salary;

**public** **int** getSalary() {

**return** salary;

}

**public** **void** setSalary(**int** salary) {

**this**.salary = salary;

}

**public** String[] getEmailIds() {

**return** emailIds;

}

**public** **void** setEmailIds(String[] emailIds) {

**this**.emailIds = emailIds;

}

**public** String getId() {

**return** id;

}

**public** **void** setId(String id) {

**this**.id = id;

}

**public** String getFirstName() {

**return** firstName;

}

**public** **void** setFirstName(String firstName) {

**this**.firstName = firstName;

}

**public** String getLastName() {

**return** lastName;

}

**public** **void** setLastName(String lastName) {

**this**.lastName = lastName;

}

**public** **int** getAge() {

**return** age;

}

**public** **void** setAge(**int** age) {

**this**.age = age;

}

@Override

**public** String toString() {

**return** "EmpBean{" + "id='" + id + '\'' + ", firstName='" + firstName + '\'' + ", lastName='" + lastName + '\''

+ ", age=" + age + ", emailIds=" + Arrays.*toString*(emailIds) + ", salary=" + salary + '}';

}

}

**SolrCRUDOperations.java**

**package** com.ddlab.rnd.solr;

**import** org.apache.solr.client.solrj.SolrClient;

**import** org.apache.solr.client.solrj.SolrQuery;

**import** org.apache.solr.client.solrj.SolrServerException;

**import** org.apache.solr.client.solrj.impl.HttpSolrClient;

**import** org.apache.solr.client.solrj.response.QueryResponse;

**import** org.apache.solr.client.solrj.response.UpdateResponse;

**import** org.apache.solr.common.SolrDocument;

**import** org.apache.solr.common.SolrDocumentList;

**import** org.apache.solr.common.params.SolrParams;

**import** java.io.IOException;

**import** java.util.\*;

/\*\*

\* Created by PIKU on 7/9/2016.

\*/

**public** **class** SolrCRUDOperations {

**private** **static** String *SOLR\_BASE\_URL* = "http://localhost:8983/solr";

**private** **static** String *SOLR\_CORE\_NAME* = "test\_solr\_core";

**private** **static** String *SOLR\_CORE\_URL* = *SOLR\_BASE\_URL* + "/" + *SOLR\_CORE\_NAME*;

/\*\*

\* Execute query by Id

\*/

**public** **static** **void** executeQuery() {

SolrClient solrClient = **new** HttpSolrClient.Builder(*SOLR\_CORE\_URL*).build();

SolrQuery query = **new** SolrQuery();

// query.add("id", "5");

query.set("q", "id:5");

**try** {

QueryResponse qResponse = solrClient.query(query);

**int** status = qResponse.getStatus();

**if** (status == 0)

System.***out***.println("Query executed successfully ...");

System.***out***.println(qResponse);

// SolrDocumentList sdList = qResponse.getResults();

// Iterator itr = sdList.iterator();

// while( itr.hasNext() ) {

// SolrDocument sdoc = (SolrDocument) itr.next();

// System.out.println(sdoc.getFieldValue("fname"));

// }

// OR

List<EmpBean> empBeanList = qResponse.getBeans(EmpBean.**class**);

**for** (EmpBean eBean : empBeanList)

System.***out***.println(eBean.toString());

} **catch** (SolrServerException e) {

e.printStackTrace();

} **catch** (IOException e) {

e.printStackTrace();

}

}

/\*\*

\* Query -> salary: [2000 TO \*] -> greater than 2000 Query -> salary: [\* TO

\* 500] -> Less than 500

\*/

**public** **static** **void** executeQuery\_1() {

SolrClient solrClient = **new** HttpSolrClient.Builder(*SOLR\_CORE\_URL*).build();

SolrQuery query = **new** SolrQuery();

query.set("q", "id:[2000 TO \*]");

**try** {

QueryResponse qResponse = solrClient.query(query);

**int** status = qResponse.getStatus();

**if** (status == 0)

System.***out***.println("Query executed successfully ...");

System.***out***.println(qResponse);

// SolrDocumentList sdList = qResponse.getResults();

// Iterator itr = sdList.iterator();

// while( itr.hasNext() ) {

// SolrDocument sdoc = (SolrDocument) itr.next();

// System.out.println(sdoc.getFieldValue("fname"));

// }

// OR

List<EmpBean> empBeanList = qResponse.getBeans(EmpBean.**class**);

**for** (EmpBean eBean : empBeanList)

System.***out***.println(eBean.toString());

} **catch** (SolrServerException e) {

e.printStackTrace();

} **catch** (IOException e) {

e.printStackTrace();

}

}

**public** **static** **void** saveOrUpdatePojos(String solrURL, List<EmpBean> emps) {

SolrClient solrClient = **new** HttpSolrClient.Builder(solrURL).build();

**try** {

solrClient.addBeans(emps);

UpdateResponse response = solrClient.commit();

**int** status = response.getStatus();

**if** (status == 0)

System.***out***.println("Saved successfully ...");

} **catch** (IOException e) {

e.printStackTrace();

} **catch** (SolrServerException e) {

e.printStackTrace();

} **finally** {

**try** {

solrClient.close();

} **catch** (IOException e) {

e.printStackTrace();

}

}

}

**public** **static** **void** updatePojo(String solrURL, EmpBean emp) {

SolrClient solrClient = **new** HttpSolrClient.Builder(solrURL).build();

**try** {

solrClient.addBean(emp);

UpdateResponse response = solrClient.commit();

**int** status = response.getStatus();

**if** (status == 0)

System.***out***.println("Updated successfully ...");

} **catch** (IOException e) {

e.printStackTrace();

} **catch** (SolrServerException e) {

e.printStackTrace();

} **finally** {

**try** {

solrClient.close();

} **catch** (IOException e) {

e.printStackTrace();

}

}

}

**public** **static** **int** getSalary() {

Random rand = **new** Random();

**return** rand.nextInt(10000);

}

**public** **static** **void** deleteAll() {

SolrClient solrClient = **new** HttpSolrClient.Builder(*SOLR\_CORE\_URL*).build();

**try** {

solrClient.deleteByQuery("\*:\*");

UpdateResponse response = solrClient.commit();

**int** status = response.getStatus();

**if** (status == 0)

System.***out***.println("Deleted all successfully ...");

} **catch** (SolrServerException e) {

e.printStackTrace();

} **catch** (IOException e) {

e.printStackTrace();

}

}

**public** **static** **void** deleteById(**int** id) {

SolrClient solrClient = **new** HttpSolrClient.Builder(*SOLR\_CORE\_URL*).build();

**try** {

solrClient.deleteById(String.*valueOf*(id));

UpdateResponse response = solrClient.commit();

**int** status = response.getStatus();

**if** (status == 0)

System.***out***.println("Deleted id " + id + " successfully ...");

} **catch** (SolrServerException e) {

e.printStackTrace();

} **catch** (IOException e) {

e.printStackTrace();

}

}

**public** **static** **void** save() {

List<EmpBean> empList = **new** ArrayList<EmpBean>();

**for** (**int** i = 0; i < 20; i++) {

EmpBean emp = **new** EmpBean();

emp.setId(String.*valueOf*(i));

emp.setStreetNo("#" + i);

emp.setCity("CT Center-" + i);

emp.setCountry("India");

emp.setAge(i);

emp.setFirstName("Name-" + i);

emp.setLastName("Surname-" + i);

emp.setEmailIds(**new** String[] { "a@a.com", "b@b.com" });

emp.setSalary(*getSalary*());

empList.add(emp);

}

*saveOrUpdatePojos*(*SOLR\_CORE\_URL*, empList);

}

**public** **static** **void** update() {

// To update Employee information

EmpBean emp = **new** EmpBean();

emp.setId(String.*valueOf*(3));

emp.setStreetNo("#" + 3);

emp.setCity("CT Center-" + 3);

emp.setCountry("Norway");

emp.setAge(37);

emp.setFirstName("Sabyasachee");

emp.setLastName("Mishra");

emp.setEmailIds(**new** String[] { "saby@ge.com", "sabyasachee@motilla.com" });

*updatePojo*(*SOLR\_CORE\_URL*, emp);

}

**public** **static** **void** main(String[] args) {

// deleteAll();//To delete everything

// deleteById(0);//To delete an Id

// save();

// update();

// executeQuery();

*executeQuery\_1*();

}

}

**SolrImport.java**

**package** com.ddlab.rnd.solr;

**import** org.apache.solr.client.solrj.SolrClient;

**import** org.apache.solr.client.solrj.SolrServerException;

**import** org.apache.solr.client.solrj.impl.HttpSolrClient;

**import** org.apache.solr.client.solrj.response.QueryResponse;

**import** org.apache.solr.client.solrj.response.UpdateResponse;

**import** org.apache.solr.common.params.ModifiableSolrParams;

**import** java.io.IOException;

/\*\*

\* Created by PIKU on 7/16/2016.

\*/

**public** **class** SolrjImport {

**private** **static** String *SOLR\_BASE\_URL* = "http://localhost:8983/solr";

**private** **static** String *SOLR\_CORE\_NAME* = "core1";

**private** **static** String *SOLR\_CORE\_URL* = *SOLR\_BASE\_URL* + "/" + *SOLR\_CORE\_NAME*;

**public** **static** **void** executeFullImport() {

SolrClient solrClient = **new** HttpSolrClient.Builder(*SOLR\_CORE\_URL*).build();

ModifiableSolrParams params = **new** ModifiableSolrParams();

params.set("qt", "/dataimport");

params.set("command", "full-import");

params.set("clean", "true");// false

**try** {

QueryResponse response = solrClient.query(params);

**int** status = response.getStatus();

**if** (status == 0)

System.***out***.println("Imported data successfully ...");

} **catch** (IOException e) {

e.printStackTrace();

} **catch** (SolrServerException e) {

e.printStackTrace();

} **finally** {

**try** {

solrClient.close();

} **catch** (IOException e) {

e.printStackTrace();

}

}

}

**public** **static** **void** executeDeltaImport() {

SolrClient solrClient = **new** HttpSolrClient.Builder(*SOLR\_CORE\_URL*).build();

ModifiableSolrParams params = **new** ModifiableSolrParams();

params.set("qt", "/dataimport");

params.set("command", "delta-import");

params.set("clean", "true");// false

**try** {

QueryResponse response = solrClient.query(params);

**int** status = response.getStatus();

**if** (status == 0)

System.***out***.println("Delta Import Executed successfully ...");

} **catch** (IOException e) {

e.printStackTrace();

} **catch** (SolrServerException e) {

e.printStackTrace();

} **finally** {

**try** {

solrClient.close();

} **catch** (IOException e) {

e.printStackTrace();

}

}

}

**public** **static** **void** main(String[] args) {

// executeFullImport();

*executeDeltaImport*();

}

}

**How to create a SOLR CORE and PULL DATA from Oracle Table**

# How to create a SOLR Core

1. Go to the directory **%SOLR\_HOME%/server/solr**. In my case the example is “**E:\devsoftwares\solr-6.1.0\server\solr**”
2. Create a directory of core name, let us create a core named “**core1**” and create a directory called “**core1**”.
3. Create a directory called “**lib**” and place the jar file “**ojdbc6.jar**”.
4. Create a properties file “core.properties” and add the followings.

**#Written by CorePropertiesLocator**

**#Tue Jul 05 10:15:45 UTC 2016**

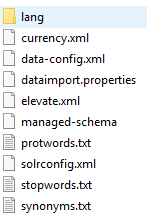
**name=core1**

**config=solrconfig.xml**

**schema=schema.xml**

**dataDir=data**

1. As you see above give the name as “**core1**”.
2. Now create a directory called “**conf**” and put all the files. Find below the screenshot.



1. Now open the file “**data-config.xml**” and add the following as per your requirement.

**<dataConfig>**

**<dataSource name="jdbc" driver="oracle.jdbc.OracleDriver" url="jdbc:oracle:thin:@localhost:1521/xe" user="test" password="test" />**

**<document>**

**<entity name="jobs" query="select \* from jobs"**

**deltaImportQuery="select \* from jobs where id='${dataimporter.delta.ID}'"**

**deltaQuery="select id from jobs where TO\_CHAR(UPDATED\_DATE, 'yyyy-mm-dd hh12\:mi\:ss') > '${dataimporter.last\_index\_time}'">**

**<field column="ID" name="id" />**

**<field column="NAME" name="name" />**

**<field column="DESCRIPTION" name="description" />**

**<field column="LOCATION" name="location" />**

**<field column="COUNTRY" name="country" />**

**<field column="UPDATED\_DATE" name="updated\_date" />**

**</entity>**

**</document>**

**</dataConfig>**

1. Open the file “managed-schema” and add the following specific to database table structure.

<!-- Add fields specific to database table -->

<!-- id, name are already available -->

<field name="DESCRIPTION" type="text\_general" indexed="true" stored="true"/>

<field name="LOCATION" type="text\_general" indexed="true" stored="true"/>

<field name="COUNTRY" type="text\_general" indexed="true" stored="true"/>

<field name="UPDATED\_DATE" type="date" indexed="true" stored="true"/>

1. Finally open the file “solrconfig.xml” and the followings.

<requestHandler name="/dataimport" class="solr.DataImportHandler">

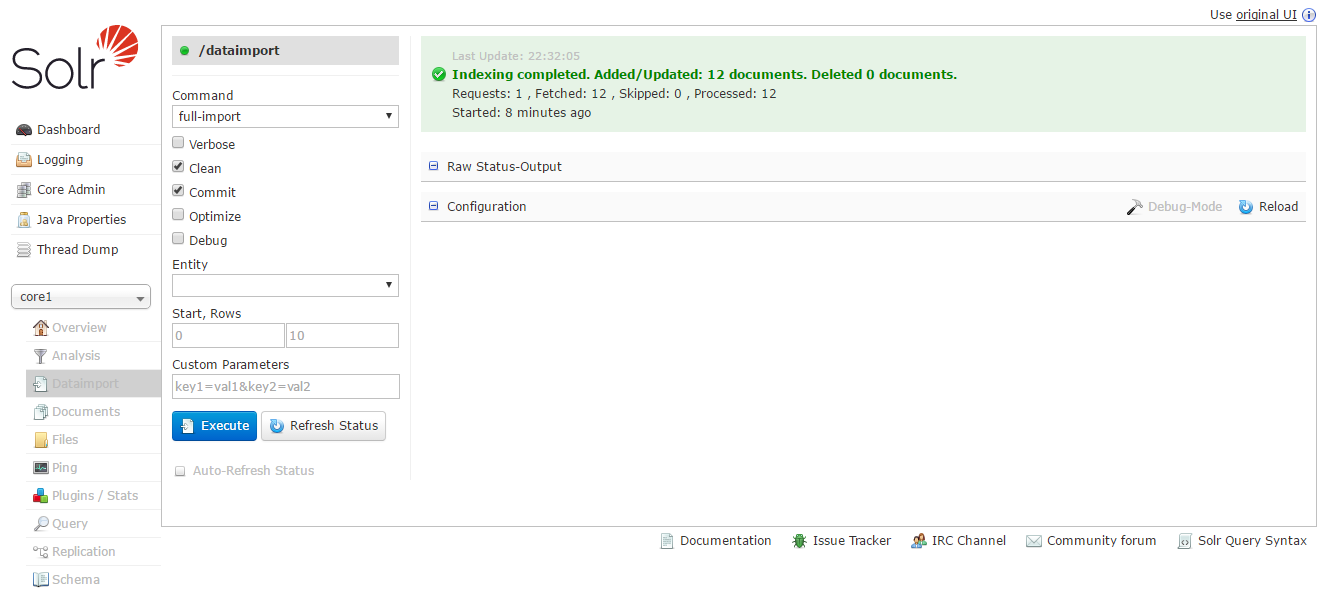
<lst name="defaults">

<str name="config">data-config.xml</str>

</lst>

</requestHandler>

1. Now restart the server and browse the url and select the core and finally click on the dataimport and execute it to get the data. See below the screenshot.



# Database Changes

The table creation script is given below.

-- Drop the existing sequence

**DROP** SEQUENCE job\_sequence;

-- Drop the existing table

**DROP** **TABLE** jobs;

-- Create the sequence

**CREATE** SEQUENCE job\_sequence

START **WITH** 1

INCREMENT **BY** 1

MINVALUE 1

MAXVALUE 10000;

-- Create the table

**CREATE** **TABLE** jobs

(

id NUMBER(3) **PRIMARY** **KEY**,

name VARCHAR2(200),

description VARCHAR2(500),

location VARCHAR2(50),

country VARCHAR2(50),

updated\_date **DATE**

);

--Insert the required data

**INSERT** **INTO** jobs

**VALUES** ( job\_sequence.NEXTVAL,

'Project Manager',

'Opening for PM',

'Bangalore',

'India',

To\_date('03/05/2003 09:02:44', 'dd/mm/yyyy hh12:mi:ss') );

**INSERT** **INTO** jobs

**VALUES** ( job\_sequence.NEXTVAL,

'Product Manager',

'Opening for Product development',

'Bangalore',

'India',

To\_date('05/07/2016 09:02:44', 'dd/mm/yyyy hh12:mi:ss') );

**INSERT** **INTO** jobs

**VALUES** ( job\_sequence.NEXTVAL,

'Sales Executive',

'Opening for a prospective sales company',

'Mangalore',

'India',

To\_date('05/07/2016 09:45:44', 'dd/mm/yyyy hh12:mi:ss') );

**INSERT** **INTO** jobs

**VALUES** ( job\_sequence.NEXTVAL,

'Call Center Executive',

'Opening for a prospective sales company',

'Bhubaneswar',

'India',

To\_date('05/07/2016 09:55:44', 'dd/mm/yyyy hh12:mi:ss') );

**INSERT** **INTO** jobs

**VALUES** ( job\_sequence.NEXTVAL,

'Marketing Officer',

'Opening for a Sales and Marketing',

'Jaipur',

'India',

To\_date('05/07/2016 09:45:44', 'dd/mm/yyyy hh12:mi:ss') );

**INSERT** **INTO** jobs

**VALUES** ( job\_sequence.NEXTVAL,

'Operation Officer',

'Opening for Operations',

'Lalbagh',

'India',

To\_date('05/07/2016 10:45:44', 'dd/mm/yyyy hh12:mi:ss') );

**INSERT** **INTO** jobs

**VALUES** ( job\_sequence.NEXTVAL,

'Doctor',

'Orthopedic department',

'New Delhi',

'India',

To\_date('05/07/2016 10:45:44', 'dd/mm/yyyy hh12:mi:ss') );

**INSERT** **INTO** jobs

**VALUES** ( job\_sequence.NEXTVAL,

'Mechanical Engineer',

'Steel Plant',

'Rourkela',

'India',

To\_date('05/07/2016 10:45:44', 'dd/mm/yyyy hh12:mi:ss') );

**INSERT** **INTO** jobs

**VALUES** ( job\_sequence.NEXTVAL,

'High Steel Mechanical Engineer',

'Steel Plant',

'Rourkela',

'India',

To\_date('06/07/2016 10:45:44', 'dd/mm/yyyy hh12:mi:ss') );

**INSERT** **INTO** jobs

**VALUES** ( job\_sequence.NEXTVAL,

'Airtel Engineer',

'Airtel Company',

'Nuagarh',

'India',

To\_date('06/07/2016 10:45:44', 'dd/mm/yyyy hh12:mi:ss') );

**INSERT** **INTO** jobs

**VALUES** ( job\_sequence.NEXTVAL,

'Airtel Engineer',

'Airtel Company',

'Nuagarh',

'India',

To\_date('07/07/2016 10:45:44', 'dd/mm/yyyy hh12:mi:ss') );

**INSERT** **INTO** jobs

**VALUES** ( job\_sequence.NEXTVAL,

'Vodafone Engineer',

'Vodafone Company',

'Nuagarh',

'India',

To\_date('09/07/2016 05:45:44', 'dd/mm/yyyy hh12:mi:ss') );

**INSERT** **INTO** jobs

**VALUES** ( job\_sequence.NEXTVAL,

'Vodafone Engineer',

'Vodafone Company',

'Nuagarh',

'India',

To\_date('09/07/2016 05:45:44', 'dd/mm/yyyy hh12:mi:ss') );

**COMMIT**;

/\* ALL Select Queries \*/

**SELECT** \*

**FROM** jobs;

**SELECT** \*

**FROM** jobs

**WHERE** To\_char(updated\_date, 'yyyy-mm-dd') > '2016-01-16';

**SELECT** \*

**FROM** jobs

**WHERE** To\_char(updated\_date, 'yyyy-mm-dd hh12\:mi\:ss') >

'2016-03-16 09\:18\:01';

**SELECT** \*

**FROM** jobs

**WHERE** To\_char(updated\_date, 'yyyy-mm-dd') > '2016-05-16';