XQuark Fusion 1.1

API Tutorial

XQUARK FUSION 1.1

API TUTORIAL

Document version 1.1

Table of contents

Part

Abstract

This document is an introduction to the XQuark Fusion API, called XML/DBC.

Part

2

Introduction

Overview

This tutorial describes in detail a single complete example of the use of XQuark Fusion. The intended audience is a programmer that is learning the API for the product. The scenario includes three data sources, each containing one of the following elements : users, items or bids of users on items. Each type of elements is supposed to be stored a different relational database but for the sake of simplicity in this tutorial is stored in the same relational database, but accessed through different accessors. The product provides a powerful mechanism to extract XML from several relational databases.

As the tutorial shows, only a few lines of code are required to perform these operations where as a “hard-coded” implementation would require hundreds of lines of code. The remainder of this tutorial describes the XML data, the relational schema, and example code for querying the data sources.

Part

3

Creating the Relational Data

The relational data consists of three tables, a `users` table, a `items` table and a `bids` table. The insertion example is given for **MySql**.

```
-- Table creation

create table users (
  userid char(3) not null,
  name varchar(20),
  rating char(1),
  unique(userid, name),
  primary key(userid)
);

-- Table filling

insert into users values('U01','Tom Jones','B');
insert into users values('U02','Mary Doe','A');
insert into users values('U03','Dee Linquent','D');
insert into users values('U04','Roger Smith','C');
insert into users values('U05','Jack Sprat','B');
insert into users values('U06','Rip Van Winkle','B');


-- Table creation

create table items (
  itemno char(4) not null,
  description varchar(30),
  offered_by char(3),
  start_date datetime,
  end_date datetime,
  reserve_price decimal(10),
  unique(itemno),
  primary key(itemno)
);


-- Table filling

insert into items values(1001,'Red Bicycle','U01','2002-
02-05','2002-02-20',40);
insert into items values(1002,'Motorcycle','U02','2002-
03-11','2002-04-15',500);
insert into items values(1003,'Old Bicycle','U02','2002-
02-10','2002-03-20',25);
insert into items values(1004,'Tricycle','U01','2002-03-
25','2002-04-08',15);
```

```
insert into items values(1005,'Tennis
Racket','U03','2002-04-19','2002-05-30',20);
insert into items values(1006,'Helicopter','U03','2002-
06-05','2002-06-25',50000);
insert into items values(1007,'Racing
Bicycle','U04','2002-02-20','2002-03-20',200);
insert into items values(1008,'Broken
Bicycle','U01','2002-03-05','2002-04-06',25);

-- Table creation

create table bids (
  userid char(3),
  itemno char(4),
  bid decimal(10) not null,
  bid_date datetime
);

-- Table filling

insert into bids values('U01',1002,400,'2002-03-14');
insert into bids values('U01',1004,40,'2002-04-05');
insert into bids values('U02',1001,35,'2002-02-07');
insert into bids values('U02',1001,45,'2002-02-11');
insert into bids values('U02',1001,55,'2002-02-15');
insert into bids values('U02',1002,600,'2002-03-16');
insert into bids values('U02',1002,1200,'2002-04-02');
insert into bids values('U03',1002,800,'2002-03-17');
insert into bids values('U03',1007,175,'2002-02-25');
insert into bids values('U04',1001,40,'2002-02-08');
insert into bids values('U04',1001,50,'2002-02-13');
insert into bids values('U04',1002,1000,'2002-03-25');
insert into bids values('U04',1003,15,'2002-02-22');
insert into bids values('U04',1007,225,'2002-03-12');
insert into bids values('U05',1003,20,'2002-03-03');
insert into bids values('U05',1007,200,'2002-03-08');
```

Part 4

Creating the configuration file

The XQuark Fusion configuration file contains information on its wrapped data sources. Each data source has its own configuration file. So four files have to be edited. For more information on configuration files please see the reference guides of XQuark Fusion and XQuark Bridge.

XQuark Fusion configuration file :

```
<accessor xmlns="http://www.xquark.org/Mediator"
type="mediator" name="Fusion">
 <launcher type="jvm"/>
 <specific/>
 <subaccessors>
  <subaccessor name="XQBridge1">
   <driver>org.xquark.extractor.ExtractorDriver</driver>
   <connection>
      xdbc:xquark:extractor:file:XQBridge1.xml
   </connection>
  </subaccessor>
  <subaccessor name=" XQBridge2">
   <driver>org.xquark.extractor.ExtractorDriver</driver>
   <connection>
      xdbc:xquark:extractor:file:XQBridge2.xml
   </connection>
  </subaccessor>
  <subaccessor name=" XQBridge3">
   <driver>org.xquark.extractor.ExtractorDriver</driver>
   <connection>
      xdbc:xquark:extractor:file:XQBridge3.xml
   </connection>
  </subaccessor>
 </subaccessors>
</accessor>
```

First data source configuration file : XQuark Bridge on **MySQL**. The only table selected in the table `users` in the schema `user`.

```
<ds:datasource name="XQBridge1"
xmlns:ds="http://www.xquark.org/Bridge/1.0/Datasource">
 <description>XQuark Bridge</description>
 <url>jdbc:mysql://localhost/test</url>
 <user>user</user>
 <password>password</password>
 <catalog name="test">
  <schema targetNamespace="http://USERS">
   <includes>
    <table name="users"/>
```

```
    </includes>
   </schema>
  </catalog>
</ds:datasource>
```

Second data source configuration file : XQuark Bridge on **MySQL**. The only table selected in the table `items` in the schema `user`.

```
<ds:datasource name="XQBridge2"
xmlns:ds="http://www.xquark.org/Bridge/1.0/Datasource">
 <description>XQuark Bridge</description>
 <url>jdbc:mysql://localhost/test</url>
 <user>user</user>
 <password>password</password>
 <catalog name="test">
  <schema targetNamespace="http://ITEMS">
   <includes>
    <table name="items"/>
   </includes>
  </schema>
 </catalog>
</ds:datasource>
```

Third data source configuration file : XQuark Bridge on **MySQL**. The only table selected in the table `bids` in the schema `user`.

```
<ds:datasource name="XQBridge3"
xmlns:ds="http://www.xquark.org/Bridge/1.0/Datasource">
 <description>XQuark Bridge</description>
<url>jdbc:mysql://localhost/test</url>
 <user>user</user>
 <password>password</password>
 <catalog name="test">
  <schema targetNamespace="http://BIDS">
   <includes>
    <table name="bids"/>
   </includes>
  </schema>
 </catalog>
</ds:datasource>
```

5

Constructing XML Documents via a Query

XQuark Fusion provides an implementation of XQuery.

Appendix A lists a complete Java program that queries the virtual XML view using XQuery. To issue an XQuery, four steps are involved. The first step obtains a connection to a XQuark Fusion object. The second step obtains a statement object. The third step issues a query and the fourth step iterates over the result.

The first step gets a connection to the database

Setting up of the connection.

```
XMLConnection xc = XMLDriverManager.getConnection(uri);
```

Getting the statement object.

```
XMLStatement xs = xc.createStatement();
```

The third step issues an XQuery. This query lists all users in alphabetic order by name and for each user, include descriptions of all the items (if any) that were bid on by that user, in alphabetic order. The boolean `result` is true if the query generated a result.

```
XMLResultSet xrs = xs.executeQuery(
  "<result> {" +
  "  for $u in collection(\"*:users\")/*:users" +
  "  order by $u/name" +
  "  return" +
  "  <user>" +
  "    { $u/name }" +
  "    { for $b in distinct-values(" +
  "          collection(\"*:bids\")/*:bids" +
  "             [userid = $u/userid]/itemno" +
  "                                 )," +
  "          $i in collection(\"*:items\")/*:items" +
  "                   [itemno = $b]" +
  "      order by $i/description/text()" +
  "      return" +
  "        <bid_on_item>" +
  "           { $i/description/text() }" +
  "        </bid_on_item>" +
  "     }" +
```

```
  "  </user>" +
  "}" +
  "</result>");
```

Finally, iteration over the result set of the query produces the XML result generated by the XQuery. In this example, there is only a single result, as the query starts with an enclosing `result` tag.

```
while (xrs.hasNext()) {
  String result = xrs.nextAsString();
  System.out.println(result);
}
```

The XML document produced by this query for the entire data set is the following:

```
<result>
    <user>
          <name>Dee Linquent</name>
          <bid_on_item>Motorcycle</bid_on_item>
          <bid_on_item>Racing Bicycle</bid_on_item>
    </user>
    <user>
          <name>Jack Sprat</name>
          <bid_on_item>Old Bicycle</bid_on_item>
          <bid_on_item>Racing Bicycle</bid_on_item>
    </user>
    <user>
          <name>Mary Doe</name>
          <bid_on_item>Motorcycle</bid_on_item>
          <bid_on_item>Red Bicycle</bid_on_item>
    </user>
    <user>
          <name>Rip Van Winkle</name>
    </user>
    <user>
          <name>Roger Smith</name>
          <bid_on_item>Motorcycle</bid_on_item>
          <bid_on_item>Old Bicycle</bid_on_item>
          <bid_on_item>Racing Bicycle</bid_on_item>
          <bid_on_item>Red Bicycle</bid_on_item>
    </user>
    <user>
          <name>Tom Jones</name>
          <bid_on_item>Motorcycle</bid_on_item>
          <bid_on_item>Tricycle</bid_on_item>
    </user>
</result>
```

Appendix A - Example

```
import org.xquark.xml.xdbc.*;

public class Eight {
  static public void main(String argv []) throws Exception {
    String driver = "org.xquark.mediator.MediatorDriver" ;
    String uri = "xdbc:xquark:mediator:file:med-conf.xml";
    Class.forName (driver);
    XMLConnection xc = XMLDriverManager.getConnection(uri);
    XMLStatement xs = xc.createStatement();
    XMLResultSet xrs = xs.executeQuery(
      "<result> {" +
      "  for $u in collection(\"*:users\")/*:users" +
      "  order by $u/name" +
      "  return" +
      "  <user>" +
      "    { $u/name }" +
      "    { for $b in distinct-values(" +
      "          collection(\"*:bids\")/*:bids" +
      "             [userid = $u/userid]/itemno" +
      "                                 )," +
      "          $i in collection(\"*:items\")/*:items" +
      "                    [itemno = $b]" +
      "      order by $i/description/text()" +
      "      return" +
      "        <bid_on_item>" +
      "           { $i/description/text() }" +
      "        </bid_on_item>" +
      "     }" +
      "  </user>" +
      "}" +
      "</result>");
    while (xrs.hasNext()) {
          String result = xrs.nextAsString();
          System.out.println(result);
    }
    xc.close();
  }
}
```