Distributed Database Querying System

Abstract

Aim:

The project aims to develop an application which could query multiple heterogeneous Databases in a distributed fashion.

The project would provide me familiarity with the RMI mechanism as implemented in Java. It will also give me an insight into some of the challenges involved with designing and building a distributed application.

Objective:

The problem involves developing a distributed database querying system. The system has three databases. These databases are MS Access databases as used in this Implementation. However, these databases could be of varying architectures like Oracle or MS SQL server and so on.

The system has three Access databases on three different machines. The database structures are as described in the problem statement earlier. Just above the databases on each machine is a Query Database object. The Query Database object can talk to the Database and make queries on it. Since the Query Database objects are on the same Machine as the database they are interacting with, it is assumed that the individual Query Database objects would know the structure of the database they are querying. The Query Database object talks to the database

using JDBC (Java Database Connectivity) calls. Unfortunately, the Access databases do not recognize JDBC calls. However, Access understands another standard i.e. ODBC (Open Database Connectivity). Thus the Query Database objects use a JDBC-ODBC bridge to connect to the database.

Each machine implements a server that implements a MyServer interface. These servers have the job of binding the name of the particular machine with the RMI registry. The servers implement among other things, a method called ProcessData. The Process Data method is the remote method that would be called using RMI.

So the execution goes on like this,

1. A client takes a query from the user. This query is a sid of a student whose information is required.

2. The client calls the ProcessData method of the second machine remotely and passes it the query string and a result string (which is initially null). The ProcessData calls the corresponding QueryDatabase method and store the results got into the result string. It then calls the ProcessData of the third machine remotely.

3. The ProcessData method on the third machine does a similar job and calls the ProcessData method on the first machine (on which the client was run). It then contacts the QueryDatabase object and gets the results from the first database which are appended to the result string and finally, the ProcessData method on the first machine will write out the result string to a file which is named as query.

Existing System

We can't access more than one databases existing in different servers.

Proposed System

By using this we can overcome the drawback of existing one.

Scope of the System

We can also implement internationalization (i18n) to support user interface in various/local languages.

Module Description

The system **"DDQS"** consists of 4 modules

1. Personal Info Module.
2. Finance Info Module.
3. Account Info Module.
4. Schema Management Module

1) Personal Info:

This module stores the names of students, their phone numbers and a unique pid. (Similar to VT)

2) Finance Info:

This module stores the current Hokie account credit and insurance amounts for each student (identified with the unique pid field)

3) **Account Info:**

VT has decided to let students take copies and printouts up to a certain limit. This database holds the current number of copies and printouts taken by each student identified with the unique pid field

4) **Schema management Module:**

This module manages the schema of the database spreaded across in all the DBs. It helps if this system want to send the same amount of data to interoperable systems then this schema document is submitted to other platforms to make them understand that schema/structure in which data is organized.

Features to be implemented

- ***Normalized database***
- ***Maintainability***
- ***Exception handling***
- ***Client-side validations***

Technologies to be used

- ***Web Presentation: HTML, CSS***
- ***Client – side Scripting: JavaScript***
- ***Programming Language: Java***
- ***Web based Technologies: JNDI, RMI,DAO***
- ***Database Connectivity API: JDBC***
- ***Debug Tool: Log 4J***
- ***CASE tool: Rational Rose, Visual Paradigm, Enterprise Architect***
- ***Backend Database: Oracle/SQL Server/MY SQL/MS Access***
- ***Operating System: Windows XP/2000/2003, LINUX, Solaris***
- ***IDEs: Eclipse with My Eclipse plug-ins/Net Beans/RAD***

Hardware requirements

- *Pentium processor -------- 233 MHZ or above*
- *RAM Capacity -------- 128MB*
- *Hard Disk -------- 20GB*
- *Floppy disk -------- 1.44 MB*
- *CD-ROM Drive -------- 32 HZ*
- *KEYBOARD -------- 108 Standard*