DISTRIBUTED DATABASES MANAGEMENT USING REMOTE ACCESS METHOD

MIRCEA PETRINI [*]

ABSTACT: *Because of the formidable obstacles to realizing the ideal distributed database, DBMS vendors have taken a step-by-step approach to databases and networking. They have focused on specific forms of network database access, data distribution, and distributed data management that are appropriate for particular application scenarios. This paper studies the remote access method as a component of the distributed databases management.*

KEY WORDS: *Distributed databases, Remote Access, Oracle*

1. DISTRIBUTED DATA

Commercial data processing in a modern corporation has evolved a long way from the centralized environment of the 1970s. Figure 1 shows a portion of a computer network that you might find in a manufacturing company, a financial services firm, or in a distribution company today. Data is stored on a variety of computer systems in the network:

- Mainframes;
- Workstations and UNIX and Linux-based servers;
- LAN server;
- Desktop personal computers;
- Mobile laptop PC's;
- Handheld device;
- Internet connections.

With data spread over many different systems, it's easy to imagine requests that span more than one database, and the possibility for conflicting data among the databases:

- An engineer needs to combine lab test results (on an engineering workstation) with production forecasts (on the mainframe) to choose among three alternative technologies.

[*] *Lecturer, Ph.D. Student, University of Petroşani, Romania, petrini_mircea@yahoo.com*

- A financial planner needs to link financial forecasts (in an Informix database) to historical financial data (on the mainframe).
- A product manager needs to know how much inventory of a particular product is in each distribution center (data stored on six Linux servers) to plan product obsolescence.
- Current pricing data needs to be downloaded daily from the mainframe to the distribution center servers, and also to all of the sales force's laptop computers.
- Orders need to be uploaded daily from the laptop systems and parceled out to the distribution centers; aggregate order data from the distribution centers must be uploaded to the mainframe so that the manufacturing plan can be adjusted.
- Salespeople may accept customer orders and make shipment date estimates for popular products based on their local databases, without knowing that other salespeople have made similar commitments. Orders must be reconciled and prioritized, and revised shipment estimates provided to customers.
- Engineering changes made in the workstation databases may affect product costs and pricing. These changes must be propagated through the mainframe systems and out to the web site, the distribution centers, and the sales force laptops.
- Managers throughout the company want to query the various shared databases using the PCs on their desktops.

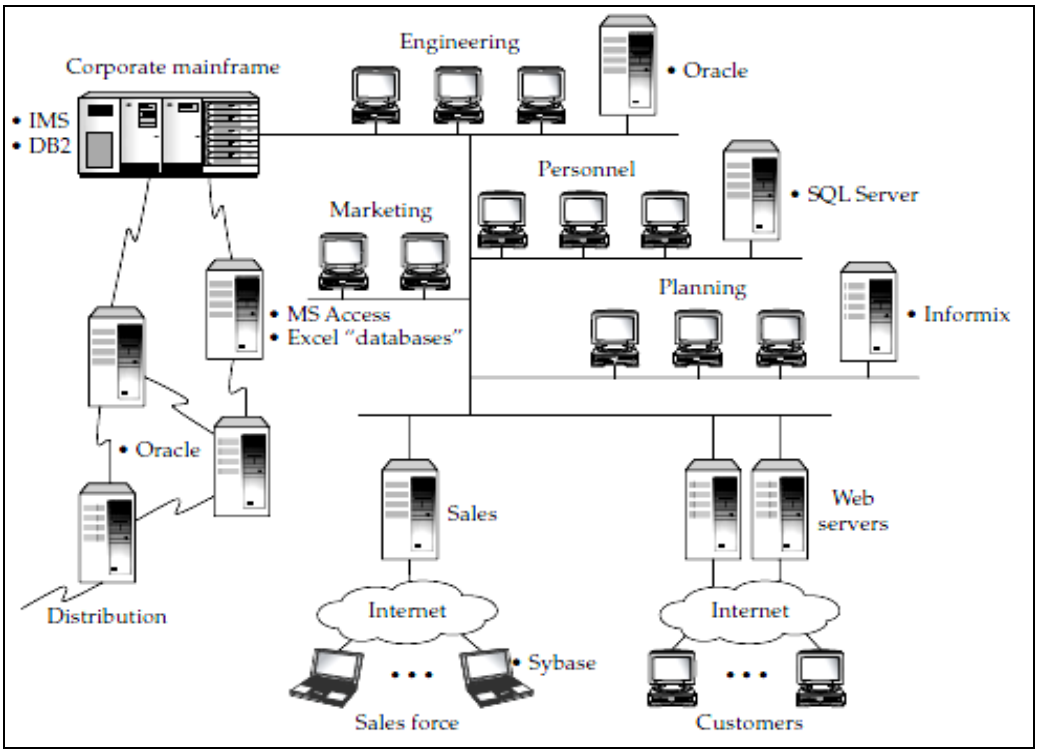

Figure 1. DBMS usage in a typical corporate network

As these examples suggest, effective ways of distributing data, managing distributed data, and providing access to distributed data have become critical as data processing has moved to a distributed computing model. The leading DBMS vendors are committed to delivering distributed database management and currently offer a variety of products that solve some of the distributed data.

2. REMOTE DATABASE ACCESS

One of the simplest approaches to managing data stored in multiple locations is remote data access. With this capability, a user of one database is given the ability to reach out across a network and retrieve information from a different database. In its simplest form, this may involve carrying out a single query against the remote database, as shown in figure 2. It may also involve performing an INSERT, UPDATE, or DELETE statement to modify the remote database contents. This type of requirement often arises when the local database is a satellite database (such as a database in a local sales office or distribution center) and the remote database is a central, corporate database.

In addition to the remote data access request, figure 2 also shows a client/server request to the remote database from a (different) PC user. From the standpoint of the remote database, there is very little difference between processing the request from the PC client and processing the remote database access request. In both cases, a SQL request arrives across the network, and the remote database determines that the user making the request has appropriate privileges and then carries out the request. In both cases, the status of the SQL processing is reported back across the network.

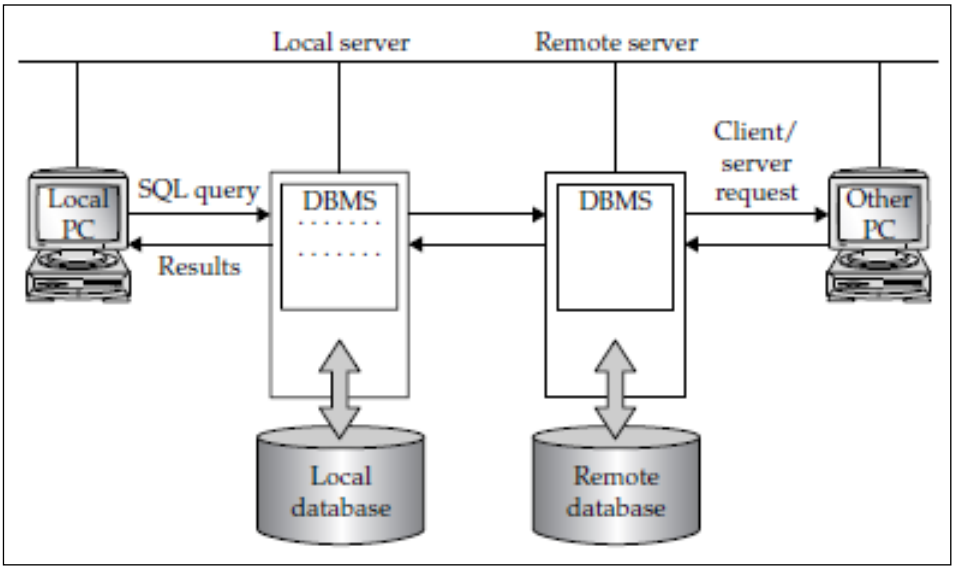

Figure 2. A remote database server access request

The local database in figure 2 must do some very different work than the process it normally uses to process local database requests, however. There are several complications for the local DBMS:

- It must determine which remote database the user wants to access, and how it can be accessed on the network.
- It must establish a connection to the remote database for carrying out remote requests.
- It must determine how the local user authentication and privilege scheme maps to the remote database. That is, does it simply pass the user name/password supplied for local database access to the remote database, or is a different remote user name/ password supplied, or should some kind of automatic mapping be performed?

Several of the leading enterprise DBMS vendors offer the kind of remote database access capability shown in figure 2. They differ in the specific way that remote access is presented to the user and to the database administrator. In some cases, they involve extensions to the SQL language accepted by the DBMS. In others, the extra mechanisms for establishing remote access are mostly external to the SQL language.

Sybase Adaptive Server Enterprise (ASE) offers a simple entry-level remote database access capability. While connected to a local Sybase installation, the user can issue a CONNECT TO SQL statement, naming a remote server that is known to the local server. For example, if a remote server named CENTRALHOST contains a copy of the sample database, then this statement:

```
CONNECT TO CENTRALHOST
```

makes that remote server the current server for the session. The local server in effect enters a pass-through mode, sending all SQL statements to the remote server. The remote database can now be processed directly over the connection, with standard, unmodified queries and data manipulation statements:

Get the names and sales numbers of all salespeople who are already over quota.

```
SELECT NAME, QUOTA, SALES
    FROM SALESREPS
WHERE SALES > QUOTA;
```

Oracle takes an approach to remote database access similar to the capabilities provided by other DBMS brands. It requires that Oracle's SQL*Net networking software be installed along with the Oracle DBMS on both the local and the remote system. The database administrator is responsible for establishing one or more named database links from the local database to remote databases. Each database link specifies:

- Network location of the target remote computer system.
- Communications protocol to use.
- Name of the Oracle database on the remote server.
- Remote database user name and password.

To access a remote database over a database link, the local system user uses standard SQL statements. The name of the database link is appended to the remote

table and view names, following an "at" sign (@). For example, assume you are on a local computer system that is connected to a copy of the sample database on a remote system over a database link called CENTRALHOST. This SQL statement retrieves information from the remote SALESREPS table:

Get the names and sales numbers of all salespeople who are already over quota.

```
SELECT NAME, QUOTA, SALES
        FROM SALESREPS@CENTRALHOST
WHERE SALES > QUOTA;
```

Oracle supports nearly all of the query capabilities that are available for the local database against remote databases. The only restriction is that every remote database entity (table, view, etc.) must be suffixed with the database link name. Also, Oracle does not support DDL or database updates via a database link. Here is a two-table join, executed on the remote Oracle database:

Get the names and office cities of all salespeople who are already over quota.

```
SELECT NAME, CITY, QUOTA, SALES
        FROM SALESREPS@CENTRALHOST, OFFICES@CENTRALHOST
WHERE SALES > QUOTA AND REP_OFFICE = OFFICE;
```

3. REMOTE DATA TRANSPARENCY

With any of the remote database naming conventions that extend the usual SQL table and view names, the additional qualifiers can quickly become annoying or confusing. For example, if two tables in the remote database have columns with the same names, any query involving both tables must use qualified column names - and the table name qualifiers now have the remote database qualification as well.

A single column reference has grown to half a line of SQL text. For this reason, table aliases are frequently used in SQL statements involving remote database access. Synonyms and aliases are also very useful for providing more transparent access to remote databases. Here's an Informix synonym definition that could be established by a user or a database administrator:

```
CREATE SYNONYM REMOTE_REPS
        FOR SAMPLE@CENTRALHOST.JOE.SALESREPS;
```

The equivalent Oracle synonym definition is

```
CREATE SYNONYM REMOTE_REPS FOR JOE.SALESREPS@CENTRALHOST;
```

With this synonym in place, the preceding qualified column name becomes simply:

```
REMOTE_REPS.NAME
```

Several DBMS brands take the synonym capability for transparent database access one step further and permit views in the local database that are defined in terms of remote database tables. Here is an Oracle view definition that creates a view called EAST_REPS in the local database. The view is a subset of information from the remote sample database:

Create a local view defined in terms of two remote tables.

```
CREATE VIEW EAST_REPS AS
        SELECT EMPL_NUM, NAME, AGE, CITY
```

```
    FROM SALESREPS@CENTRALHOST, OFFICES@CENTRALHOST
  WHERE REP_OFFICE = OFFICE
      AND REP_OFFICE BETWEEN 11 AND 19;
```

After this view has been defined, a user can pose queries in terms of the EAST_REPS view, without worrying about database links or remote table names. The view not only provides transparent remote access, but also hides from the user the remote join operation between the OFFICES and SALESREPS tables.

Transparent access to remote data, provided by views and synonyms, is usually considered a very desirable characteristic. It does have one drawback, however. Because the remote aspect of the database access is now hidden, the network overhead created by the access is also hidden. Therefore, the possibility of a user or programmer inadvertently creating a great deal of network traffic through very large queries is increased. The database administrator must make this trade-off when deciding whether to permit remote transparent synonyms and views.

4. CONCLUSIONS

Supporting such distributed queries and transactions adds a major new level of complexity (and potentially huge network data transmission overhead) to the remote access. Because of this, although several commercial DBMS systems support distributed queries and transactions, they are not heavily used in practice.

REFERENCES:

[1]. Fotache, M.; Strîmbei, C.; Cretu, L. - *ORACLE 9i2 - Ghidul dezvoltării aplicațiilor profesionale*, Editura Teora, Bucureşti, 2005
[2]. Fotache, M. - *Dialecte SQL*, Editura Gh. Asachi, Iaşi, 2002
[3]. Oracle Co. - *Oracle Database*, Administrator's Guide, 11g
[4]. Oszu, T.; Valduriez, P. - *Principles of Distributed Database Systems*, 2nd Edition, Editura Pretince Hall, 1999
[5]. Petrini, M. - *Aplicații în SQL*, Editura Focus, Petroşani, 2007