CSCI317 Database Performance Tuning

Tuning Database Interfaces

Dr Janusz R. Getta

School of Computing and Information Technology -
University of Wollongong

Tuning Database Interfaces

Outline

ODBC, JDBC, ...DBC versus native interfaces

Client-server mechanisms

Object-oriented programming

Application development tools

User interactions with a transactions

Round-trips between application and database server

Amounts of data retrieved from a database

Query compilations

Bulk loading of data

Accessing multiple databases

ODBC, JDBC, ...DBC versus native interfaces

ODBC (Open Database Connectivity) enables access to the relational databases from C/C++ applications; ODBC interface is based on a driver manager usually integrated with an operating system

JDBC (Java Database Connectivity) enable access to the relational database from Java applications; JDBC is based on a driver manager

A native interface (Pro*C/C++, OCI8, SQLJ) enables access to a particular database from C/C++/Java applications

ODBC/JDBC drivers provide transparent portability but worse performance then native interfaces; ODBC/JDBC drivers are built on top of native interfaces

Tuning Database Interfaces

Outline

- ODBC, JDBC, ...DBC versus native interfaces
- Client-server mechanisms
- Object-oriented programming
- Application development tools
- User interactions with a transactions
- Round-trips between application and database server
- Amounts of data retrieved from a database
- Query compilations
- Bulk loading of data
- Accessing multiple databases

Client-server mechanisms

Communication between a client and a database server is performed through a buffer on the server side

It may cause the following performance problems:

- A client does not consume results produced by a database server fast enough
- The resources and locks must be kept longer by a server
- Too large buffer delays the transmission of the first results
- Too small buffer forces traversing a network stack too frequently
- A buffer should fit into a frame of the underlying transport layer

Three-tier architecture client-application server- database server solves the problem

Tuning Database Interfaces

Outline

- ODBC, JDBC, ...DBC versus native interfaces
- Client-server mechanisms
- Object-oriented programming
- Application development tools
- User interactions with a transactions
- Round-trips between application and database server
- Amounts of data retrieved from a database
- Query compilations
- Bulk loading of data
- Accessing multiple databases

Object-oriented programming

Object-oriented programming forces encapsulation of object classes

Encapsulation of object classes forces one object at a time technique of object processing

Consider the relational tables

DOCUMENT schema of relational table

```
DOCUMENT(title, class, edate)
```

ACCESS schema of relational table

```
ACCESS(user,class)
```

A query `Find all documents available to a given user` is implemented as the following `SELECT` statement

SELECT statement

```
SELECT title
FROM DOCUMENT JOIN ACCESS
              ON DOCUMENT.class = ACCESS.class
WHERE user = ...;
```

SQL query optimizer is able to pick an appropriate algorithm for implementation of `join` operation

Object-oriented programming

Object-oriented programming forces encapsulation of object classes

Encapsulation of objects classes forces one object at a time technique of object processing

Document class

```
class Document{ String title;
                String className;
                Date edate )
```

Access class

```
class Access( String user,
              String className)
```

A query `find all documents available to a user` is implemented as

Object oriented implementation of a query

```
for(a in Access)
      for( d in Document)
            if a.user = ... and d.className = a.className then
                out(d.title);
```

Object-oriented programming

In object-oriented view, `join` is performed by an application and not by a database server

One object at a time technique of object processing forces nested loop implementation of `join` operation

When object-oriented application is processed remotely from a database systems the relational tables must be transmitted to a remote site

Database application programmers should directly access a bulk object (e.g. a collection of the documents) instead of forming the member objects individually and grouping them into a bulk object inside an application code

A query must be stored at and it must be processed by a database server

Tuning Database Interfaces

Outline

- ODBC, JDBC, ...DBC versus native interfaces
- Client-server mechanisms
- Object-oriented programming
- Application development tools
- User interactions with a transactions
- Round-trips between application and database server
- Amounts of data retrieved from a database
- Query compilations
- Bulk loading of data
- Accessing multiple databases

Application development tools

Application development tools may access metadata (data dictionary) to get information about column names and column types each time an application tries to access an object mapped into a row in a relational table

It takes too much time and the benefits are practically nonexistent !

A good idea is to turn off the options that allow access to metadata and to test each SQL statement outside an application

Beware of `AUTOCOMMIT`

Tuning Database Interfaces

Outline

- ODBC, JDBC, ...DBC versus native interfaces
- Client-server mechanisms
- Object-oriented programming
- Application development tools
- User interactions with a transactions
- Round-trips between application and database server
- Amounts of data retrieved from a database
- Query compilations
- Bulk loading of data
- Accessing multiple databases

User interactions within a transaction

Database transactions should be organized such that they do not include user interactions, for example manual screen update, or manual data entry

To avoid such a problem transaction should whenever it is possible divide an update into read transaction and local update into write transaction (transaction chopping)

Tuning Database Interfaces

Outline

- ODBC, JDBC, ...DBC versus native interfaces
- Client-server mechanisms
- Object-oriented programming
- Application development tools
- User interactions with a transactions
- Round-trips between application and database server
- Amounts of data retrieved from a database
- Query compilations
- Bulk loading of data
- Accessing multiple databases

Round-trips between application and db server

Transmission of data between an application and database server is expensive

Solutions:

- (1) Do not put SQL statements inside a loop; retrieve a significant amount of data outside a loop and use a loop to process the data
- (2) "Package" a number of SQL statements into one interaction, e.g. use PL/SQL inside an application implemented in a procedural programming language
- (3) Use stored functions and procedures
- (4) Use positioned updates, e.g. a cursor with FOR UPDATE clause

Tuning Database Interfaces

Outline

- ODBC, JDBC, ...DBC versus native interfaces
- Client-server mechanisms
- Object-oriented programming
- Application development tools
- User interactions with a transactions
- Round-trips between application and database server
- Amounts of data retrieved from a database
- Query compilations
- Bulk loading of data
- Accessing multiple databases

Amounts of data retrieved from a database

- Retrieve needed rows only
- Retrieve needed columns only to reduce transmission size
- Transfer only data viewed by a user
- First transfer only a subset of the set of interest then transfer the remaining rows while a user views the first results
- Make sure that cursors do not transmit one row at a time (verify cursor parameters)
- Enable index-only access to a relational table
- Allow for cancellation of ad hoc queries

Tuning Database Interfaces

Outline

- ODBC, JDBC, ...DBC versus native interfaces
- Client-server mechanisms
- Object-oriented programming
- Application development tools
- User interactions with a transactions
- Round-trips between application and database server
- Amounts of data retrieved from a database
- Query compilations
- Bulk loading of data
- Accessing multiple databases

Query compilations

Query compilation requires parsing, semantic analysis, verification of access privileges, optimization, and read access to data dictionary

If compilation of complex queries is frequently repeated then it could be too time consuming

Allow for reuse of parsed queries and optimized query execution plans (use correlation variables, share_cursor parameters, use prepared SQL)

Tuning Database Interfaces

Outline

- ODBC, JDBC, ...DBC versus native interfaces
- Client-server mechanisms
- Object-oriented programming
- Application development tools
- User interactions with a transactions
- Round-trips between application and database server
- Amounts of data retrieved from a database
- Query compilations
- Bulk loading of data
- Accessing multiple databases

Bulk loading of data

Bulk loading of data means that data is loaded directly from text files without application of SQL

Bulk loading tools like `SQL*Loader` are more effective than sequences of `INSERT` statements

Bulk loading tools disable consistency constraints, disable logging, disable index maintenance, disable triggers, disable statistics, skip data buffer cache when storing data blocks

Tuning Database Interfaces

Outline

- ODBC, JDBC, ...DBC versus native interfaces
- Client-server mechanisms
- Object-oriented programming
- Application development tools
- User interactions with a transactions
- Round-trips between application and database server
- Amounts of data retrieved from a database
- Query compilations
- Bulk loading of data
- Accessing multiple databases

Accessing multiple databases

The operations required when accessing multiple and heterogeneous databases include physical reorganization, data cleaning, semantic reconciliation

Database gateways provide transparent access to external relational and non relational data sources

Database gateways support distribution transparency, heterogeneity transparency

Performance tuning includes

- shared connections to reduce startup-costs,
- pass-through SQL, (SQL statements written in a dialect of an external data source) to reduce CPU costs statements, large transfer blocks to reduce transmission costs (minimize a number of round trips)

References

D. Shasha and P.Bonnet Database Tuning Principles, Experiments, and Troubleshooting Techniques, Morgan Kaufmann, 2003, chapters 5, 6