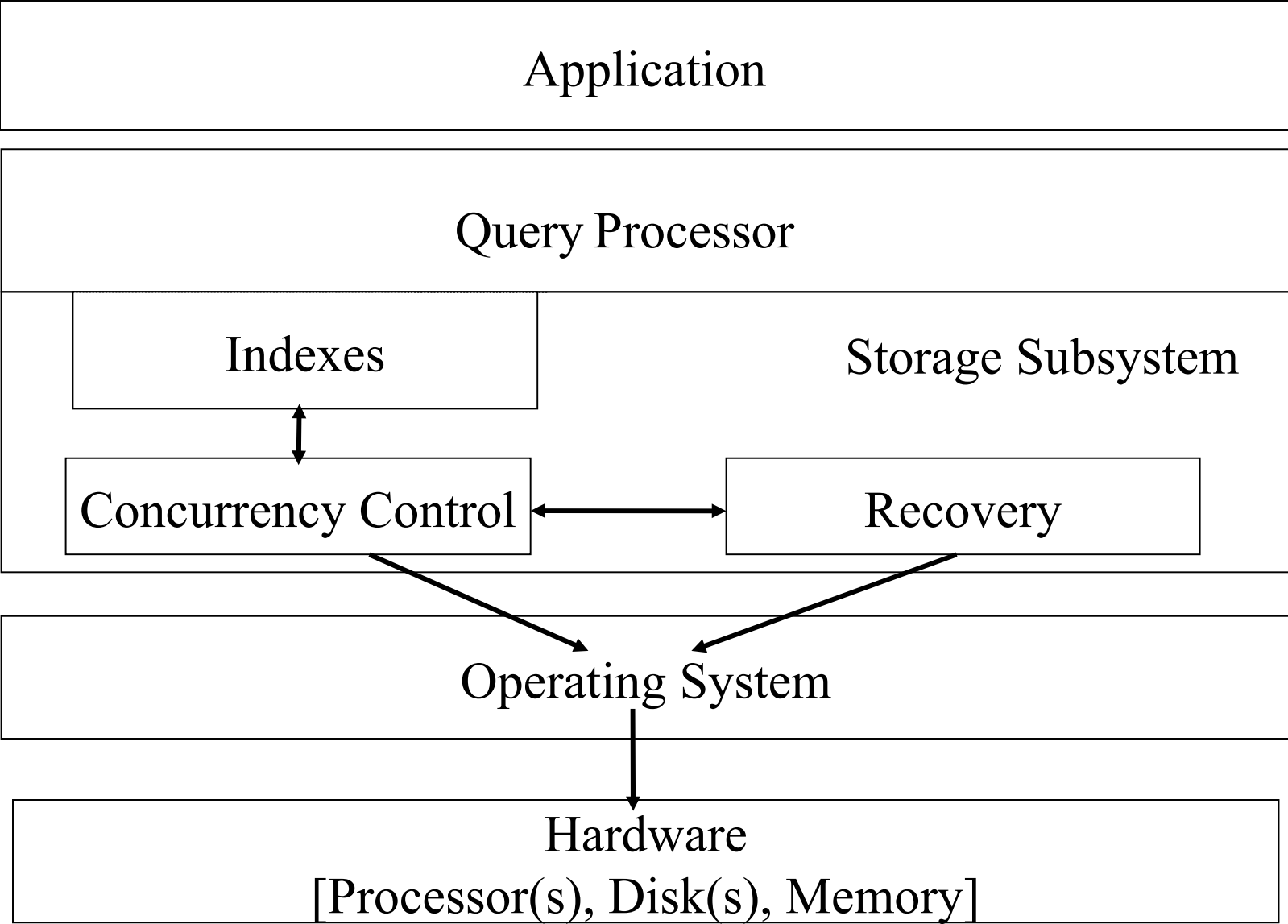


Communicating with the Outside



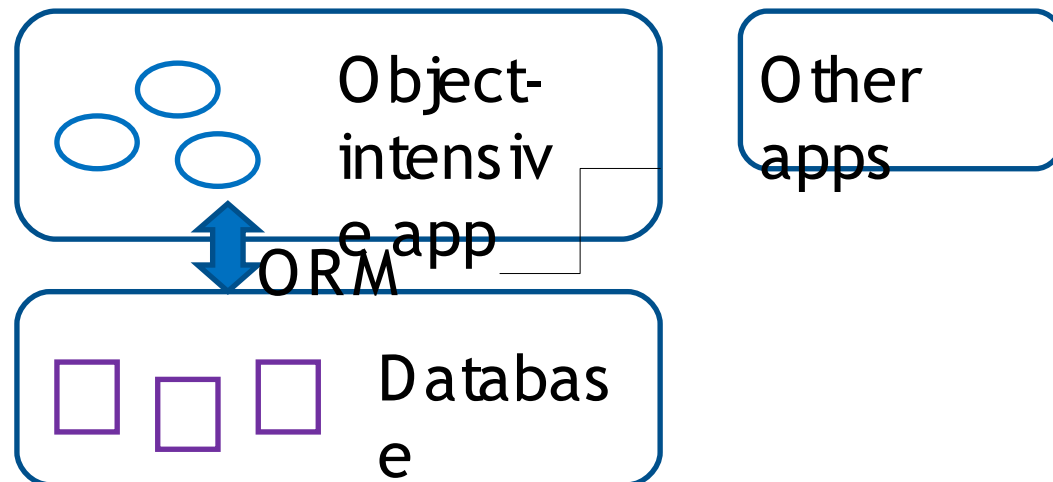
Database Programming

- Programming language + Call Level Interface
 - ODBC: Open DataBase Connectivity
 - JDBC: Java based API
 - OCI (C++/Oracle), CLI (C++/ DB2)
 - Perl/DBI
- ORM: Object-relational mapping

Object-Relational Mapping

(ORM)

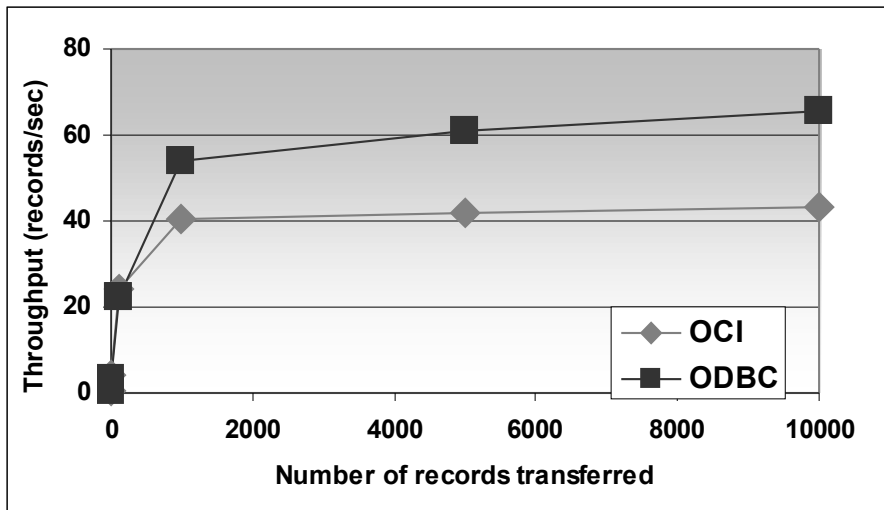
- A software system that shuttles data back and forth between database rows and objects
- Appears as a normal database user to the database
- Can share the database and tables with other apps



API pitfalls

- Cost of portability
 - Layer of abstraction on top of ODBC drivers to hide discrepancies across drivers with different conformance levels.
 - Beware of performance problems in this layer of abstraction:
 - Use of meta-data description when submitting queries, accessing the result set
 - Iterations over the result set

ODBC vs. OCI



- ODBC vs. OCI on Oracle8iEE on Windows 2000
- Iteration over a result set one record at a time. Prefetching is performed.
- Low OCI overhead when number of records transferred is small
- ODBC performs better when number of records transferred. Better job at prefetching.

Client-Server Mechanisms

- Connection pooling and multiplexing when multiple clients access a server
- Communication buffer on the database server. One per connection.
 - If a client does not consume results fast enough, then the server holds resources until it can output the result.
 - Data is sent either when the communication buffer is full or when a batch is finished executing.
 - Small buffer – frequent transfer overhead
 - Large buffer – time to first record increases.
 - No actual impact on a 100 Mb network. More sensitive in an intranet with low bandwidth.

Object-Orientation Considered Harmful

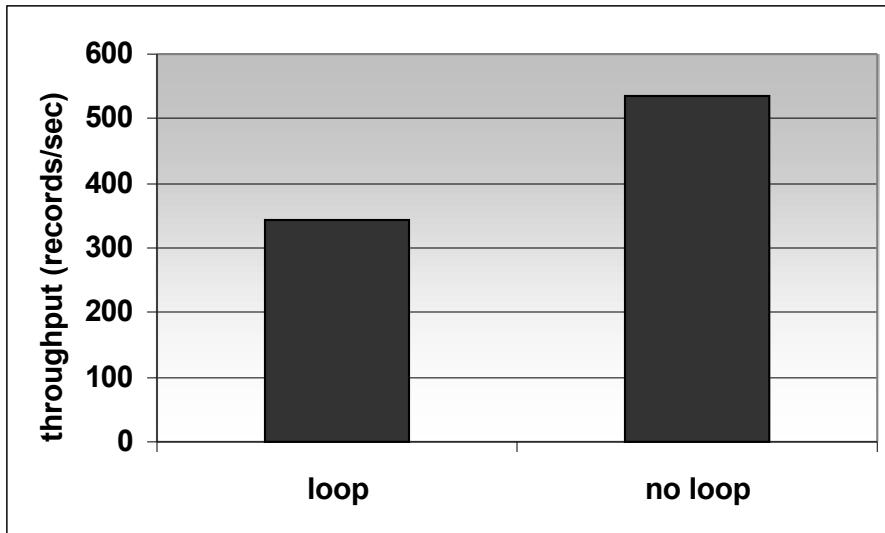
- `authorized(user, type)`
- `doc(id, type, date)`
- What are the document instances a user can see?
- SQL:

```
select doc.id, doc.date  
from authorized, doc  
where doc.type =  
authorized.type  
and authorized.user = <input>
```
- If each document is encapsulated in an object, the risk is the following:
 - Find types *t* authorized for user *input*

```
select doc.type as t  
from authorized  
where user = <input>
```
 - For each type *t* issue the query

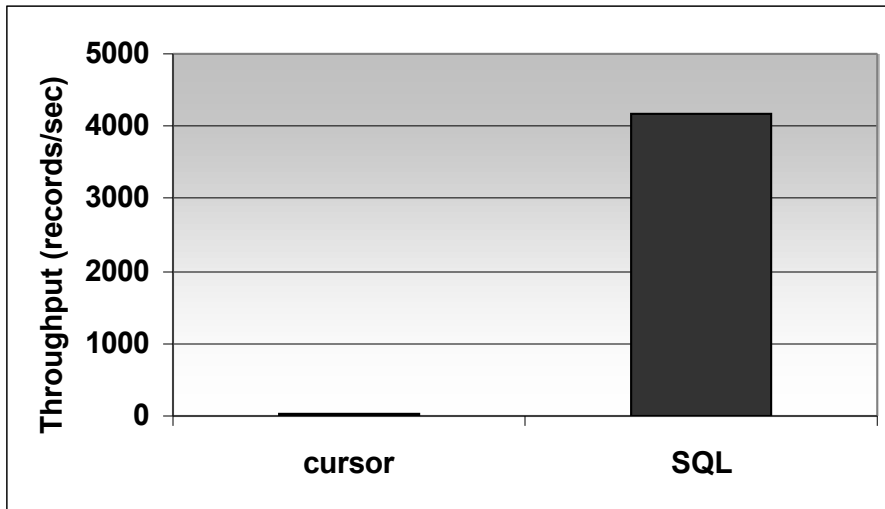
```
select id, date  
from doc  
where type = <t>;
```
 - The join is executed in the application and not in the DB!

Loops



- Fetch 2000 records
- Loop: 200 queries
- No loop: 1 query
- Crossing the application interface too often hurts performances.

Cursors



- Query fetches 200000 56 bytes records
- Response time is a few seconds with a SQL query and more than an hour iterating over a cursor.

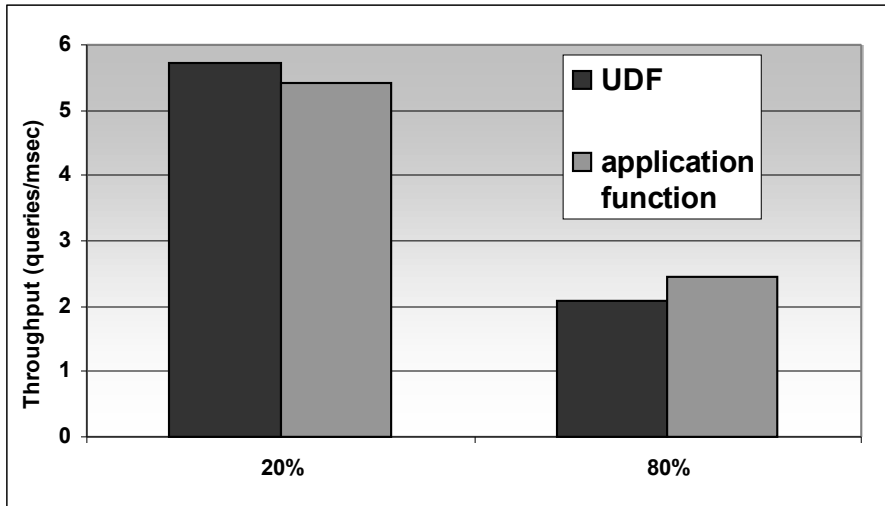
Avoid User Interaction within a Transaction

- User interaction within a transaction forces locks to be held for a long time.
- Careful transaction design (possibly transaction chopping) to avoid this problem.

Minimize the Number of Roundtrips to the Database

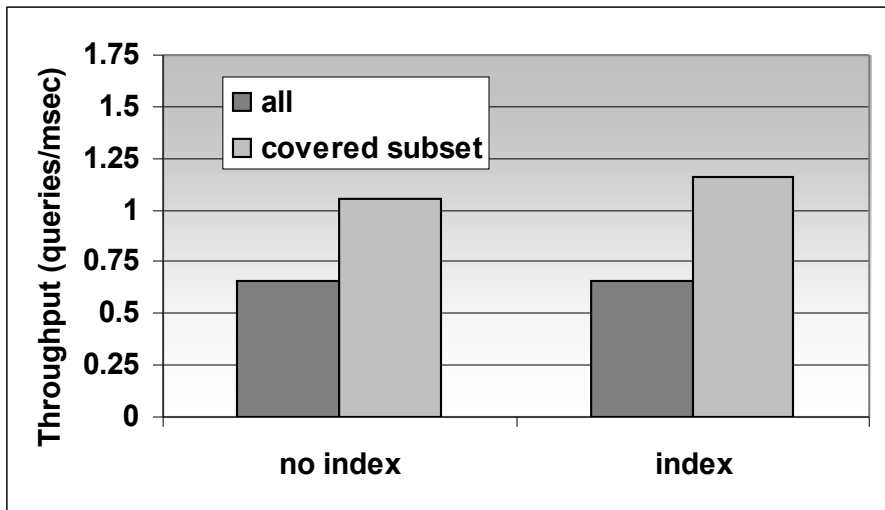
- Avoid Loops:
 - Application programming languages offer looping facilities (SQL statements, cursors, positioned updates)
 - Rigid object-oriented programming might force such loops.
- Package several SQL statements within one call to the database server:
 - Embedded procedural language (Transact SQL) with control flow facilities.
- Use User Defined Functions (UDFs) when they select out a high number of records.

User Defined Functions



- Function computes the number of working days between two dates.
- Function executed either on the database site (UDF) or on the application site
- Applying the UDF yields good performances when it helps reduce significantly the amount of data sent back to the application.

Retrieve Needed Columns Only

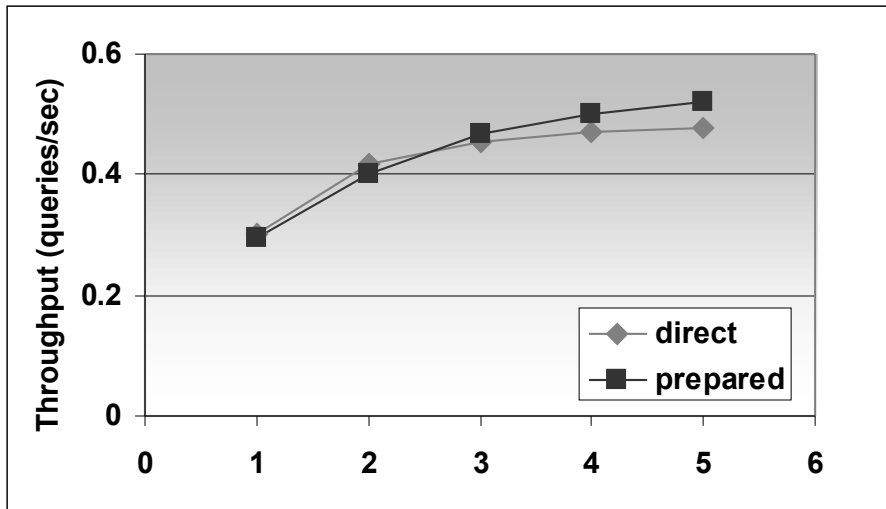


- Avoid transferring unnecessary data
- Might prevent the use of a covering index.
- In the experiment the subset contains $\frac{1}{4}$ of the attributes.
 - Reducing the amount of data that crosses the application interface yields significant performance improvement.

Retrieve Needed Rows Only

- If the user is only viewing a small subset of a very large result set, it is best to
 - Only transfer that subset
 - Only compute that subset
- Applications that allow the formulation of ad-hoc queries should permit users to cancel them.

Minimize the Number of Query Compilations



Experiment performed on Oracle8iEE on Windows 2000.

- Prepared execution yields better performance when the query is executed more than once:
 - No compilation
 - No access to catalog.
- Prepared execution plans become obsolete if indexes are added or the size of the relation changes.

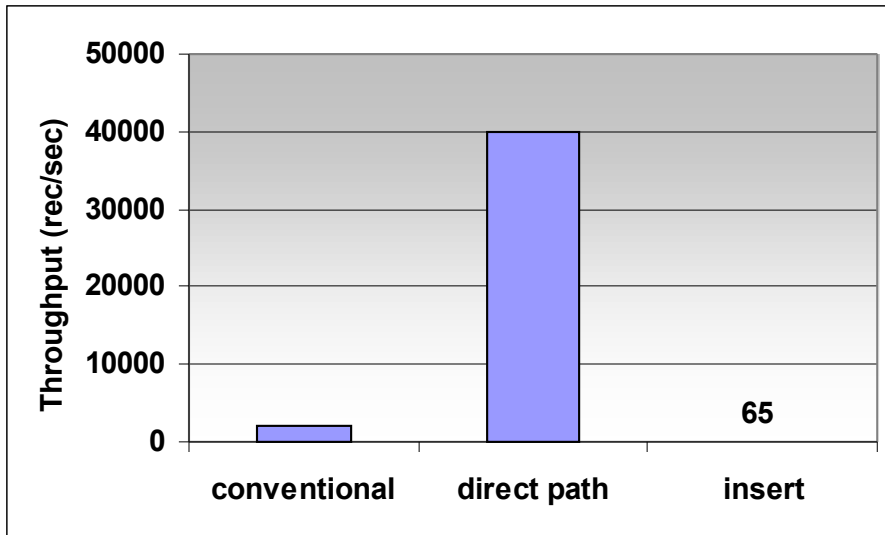
Tuning the Application Interface

- Avoid user interaction within a transaction
- Minimize the number of roundtrips between the application and the database
- Retrieve needed columns only
- Retrieve needed rows only
- Minimize the number of query compilations

Bulk Loading Data

- Tools to bulk load data in each system.
- Tool parameters:
 - Bypass query engine
 - Avoid logging
 - No index update
 - No constraint check
 - Frequency of commits

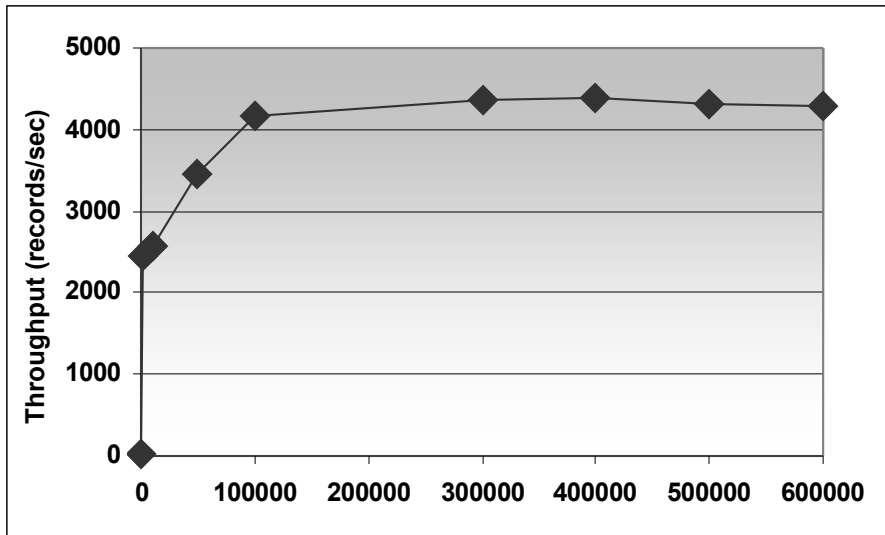
Direct Path



Experiment performed on
Oracle8iEE on Windows 2000.

- Loading 600000 records into the lineitem relation from TPCH
- Direct path loading bypasses the query engine and the storage manager. It is orders of magnitude faster than conventional path (with a commit every 100 records) and inserts (with a commit for each record).

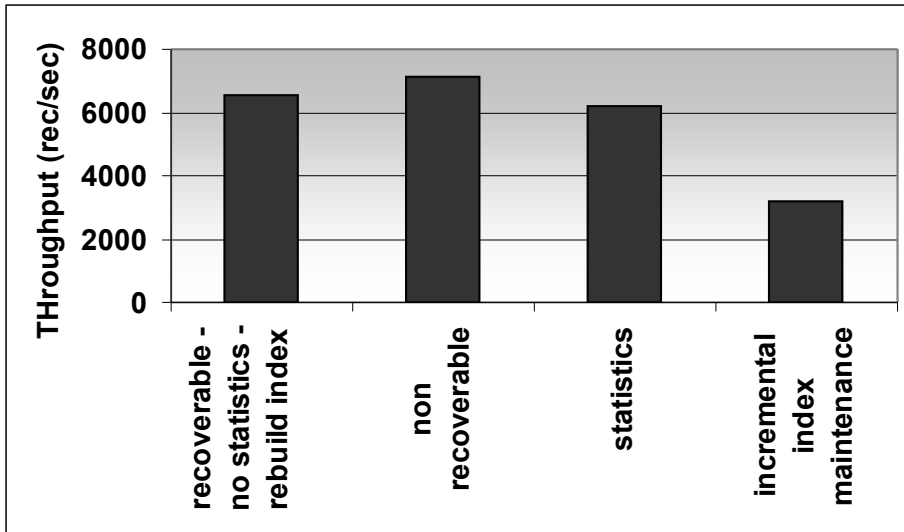
Batch Size



Experiment performed on
SQL Server 2000
on Windows 2000.

- Bulk load of 600000 records.
- Throughput increases steadily when the batch size increases to 100000 records. Throughput remains constant afterwards.
- Trade-off between performance and amount of data that has to be reloaded in case of problem.

Storage Engine Parameters



Experiment performed on
IBM DB2 UDB V7.1
on Windows 2000.

- Bulk load of 600000 records.
- As expected:
 - Turning off logging helps.
 - Collecting statistics hurts
 - Maintaining indexes incrementally hurts a lot.

Connecting to Multiple Databases

- Shared connection to reduce start-up cost
 - Connection pooling
- Pass-through statements when performance is CPU bound
 - Eliminates query rewriting to adapt to specific SQL dialect
- Transfer large amounts of data when performance is network bound