Andy LaMora

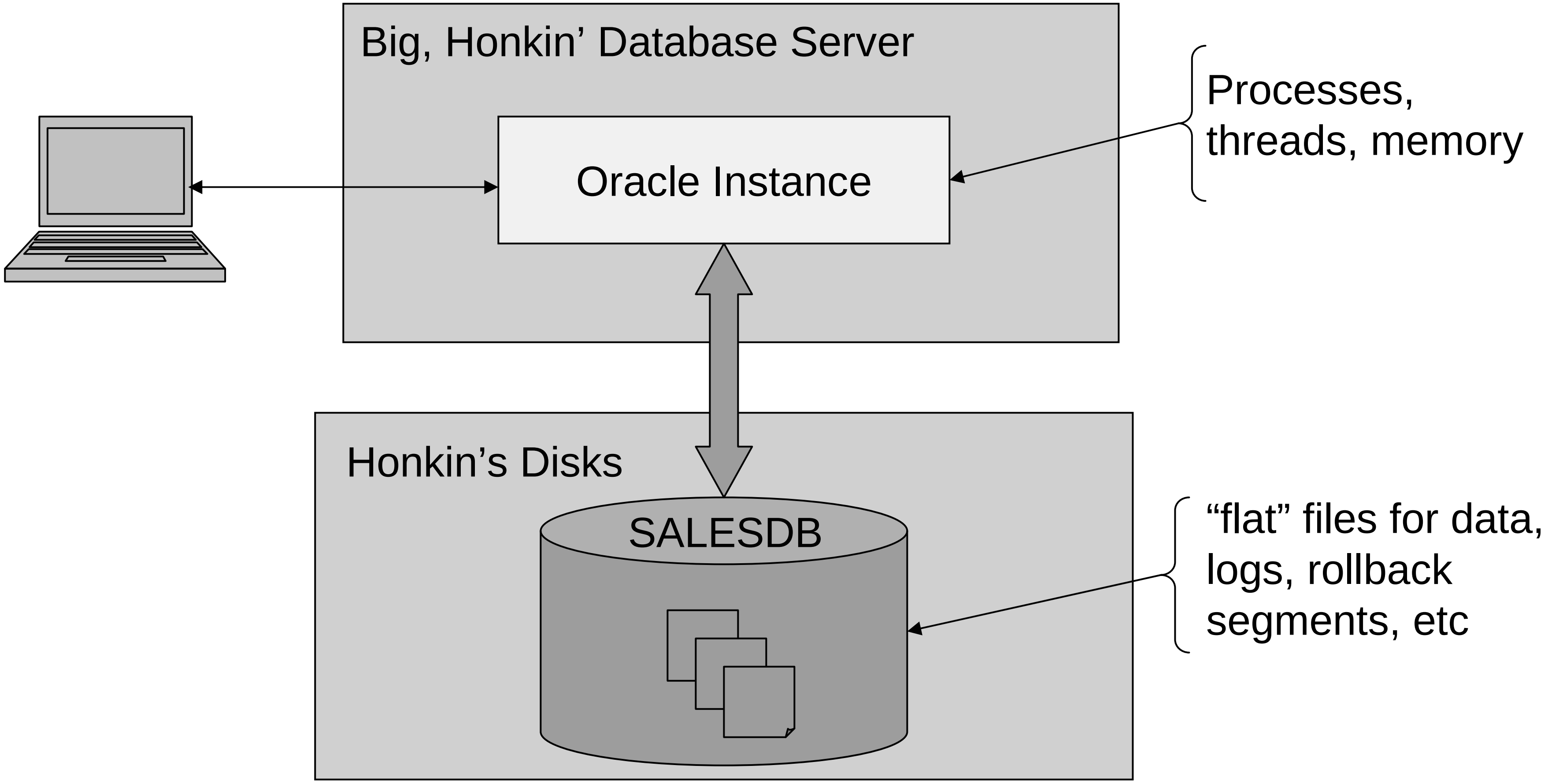
http://www.cs.jhu.edu/~alamora

NEB B21 (student lab)

Alamora@cs.jhu.edu

Oracle Notes

An Oracle Database != An Oracle *Instance*

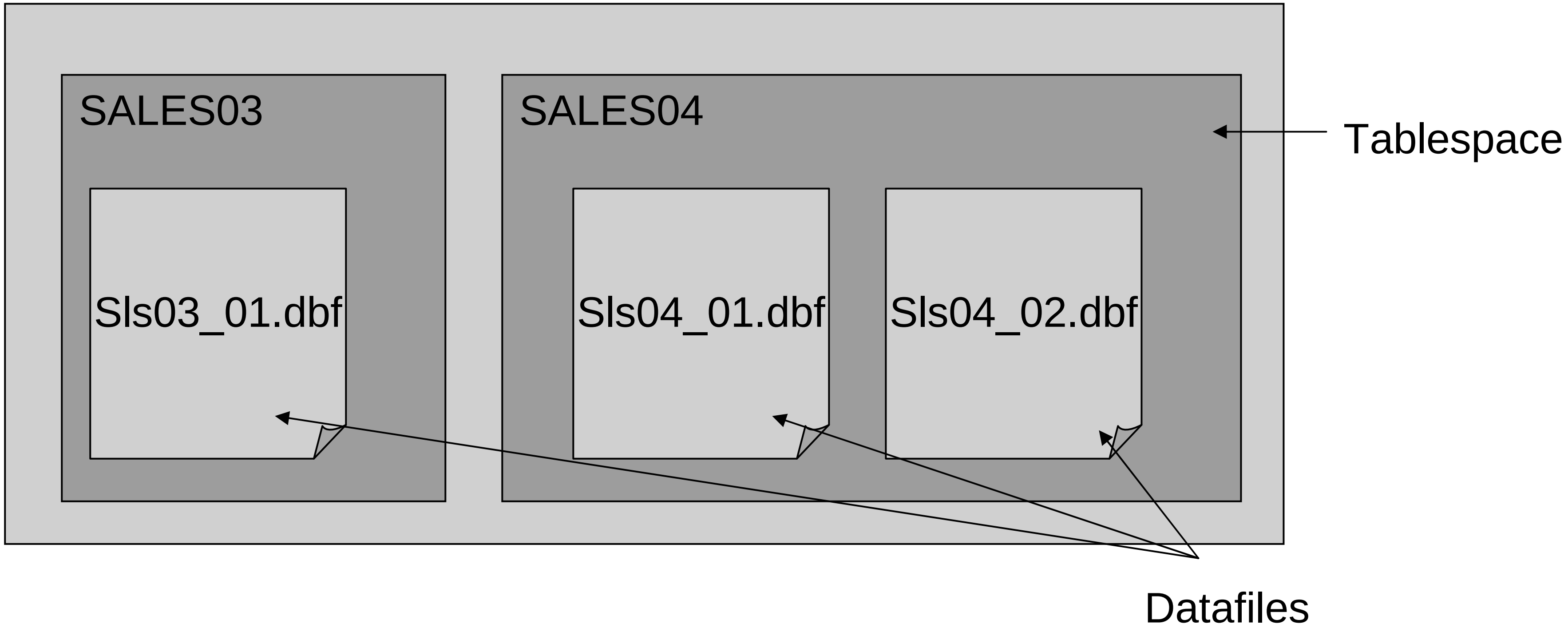

Big, Honkin' Database Server
Oracle Instance
Processes, threads, memory
Honkin's Disks
SALESDB
"flat" files for data, logs, rollback segments, etc

Oracle Notes

A Closer Look: Tablespaces vs Datafiles

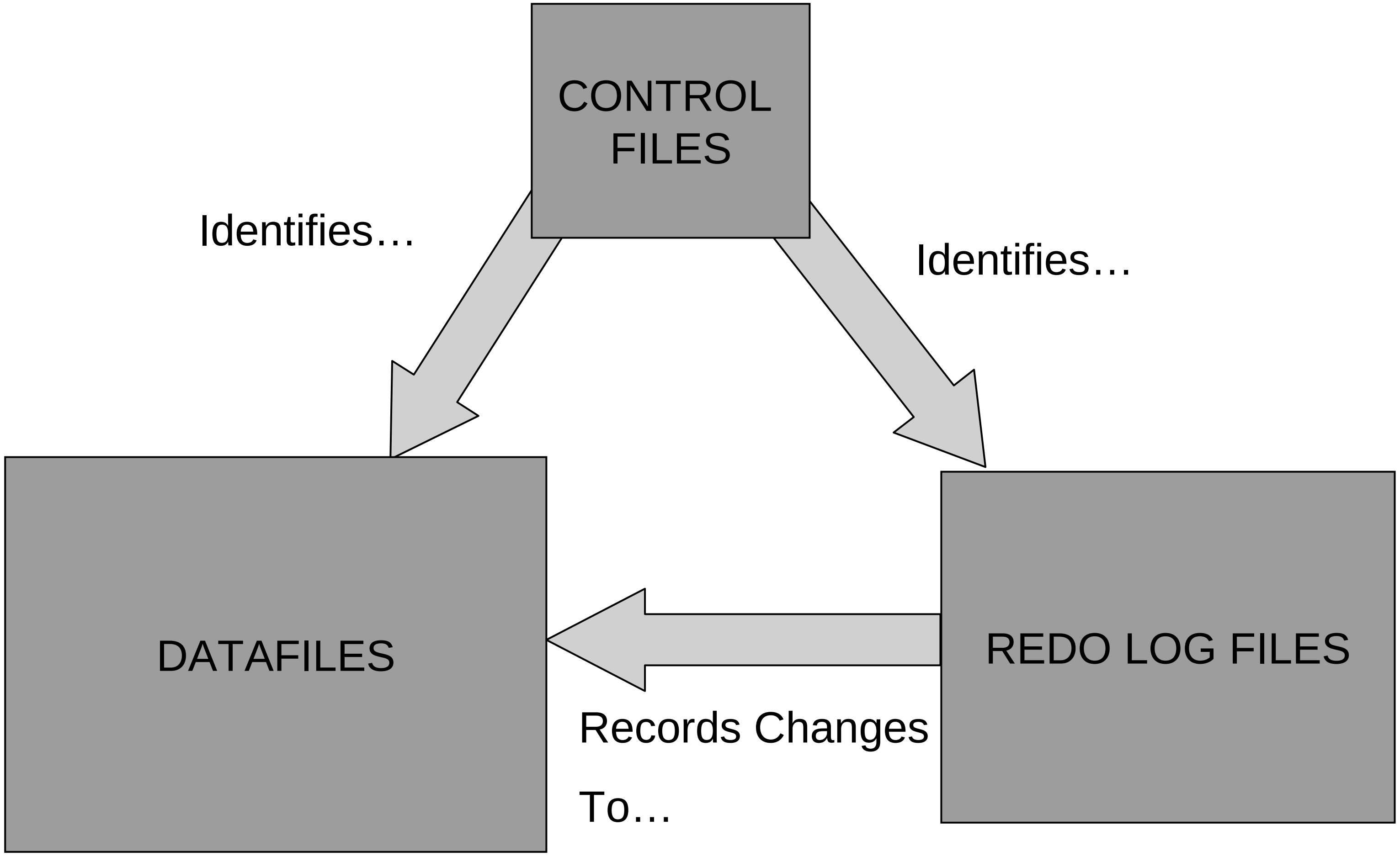
Tablespace: A *LOGICAL* construct, defining a group of data files, a workspace for users of a database

Datafiles: *PHYSICAL* constructs, files residing on system disks, were table and session data are written and stored.



Oracle Notes

Brief Overview: What the Important Files Do



CONTROL FILES

Identifies…

Identifies…

DATAFILES

REDO LOG FILES

Records Changes

To…

Oracle Notes

Sources of the most common problems:

Bad/missing environment variables:

- ORACLE_SID (the server instance)
- ORACLE_HOME (path to the data)
- ORACLE_HOME (path to the binaries and tools)

Insufficient User Permissions:

- Oracle: You don’t have the authority to do (x) – call the DBA
- System: File permissions (occurs when working with shell scripts)

Duh Factor:

- Command doesn’t do what you think it does (what it *should* do, dammit)

Oracle Notes

What my .cshrc file looks like: (or, you can source ~oracle/source.oracle)

```
alamora@dbase(~)> more .cshrc

#if ( -ne $ORACLE_SID ) then
#  source /home/oracle/oracle/oracle.csh
#endif

setenv ORACLE /home/oracle/oracle

setenv ORACLE_BASE $ORACLE/OraHome1
setenv ORACLE_HOME $ORACLE/OraHome1
setenv ORACLE_SID dbase

setenv PATH $ORACLE_BASE/bin:$JAVA_HOME/bin:$PATH
setenv LD_LIBRARY_PATH
$ORACLE_BASE/lib:$LD_LIBRARY_PATH

# Default connection string
setenv TWO_TASK dbase_shared.cs.jhu.edu

# Aliases
alias ls 'ls -l'
```

Oracle Notes:
What ***You*** Need To Do:

1. Log into dbase.cs.jhu.edu with SSH (eg PuTTy)

2. Configure your environment variables (you can just link to ~oracle/source.oracle and run it on login)

3. Run ~oracle/bin/create_oracle_user.php

4. Wait for your email, then…

   (a) Start sqlplus (sqlplus)

   (b) Log in with your uname (<username>_04) and default password (from the email)

   (c) Change your password (prompted automatically)

Oracle Notes:
What ***You*** Need To Do: (cont…)

5. Run the SQL script to create a schema for your homework.

Note: sqlplus gives you a couple ways to run data files.
You can do it from sqlplus itself in two ways:

```
SQL>@my_file.sql  (very convenient if its in your current dir)
SQL>start /long/path/name/to/file.sql
```

Or from the shell:

```
alamora> sqlplus alamora_04/passwd @thefile.sql
```

(but this is a bad idea, right?)

6. You’re done. Have fun!!

Oracle Notes

About Database Roles…

Roles are (sortof) like UNIX user groups, and define broadly what users are allowed to do. Here are some important ones:

SYSDBA: creates databases, schemas, users, anything the SYSDBA desires. This isn't an account but an Oracle-installed, special role.

DBA: **D**ata**B**ase **A**dministrator. Creates objects, files, users, policies, manages logs, and sets user roles/permissions. Also a job title. Root control over all schemas.

ALAMORA_04: just a regular, vanilla user. Devine permissions on own schema, but nobody else's.

Alphabet Soup:
Menu of Databases

Oracle 9i

MS SQL Server

PostgreSQL

DB2

MySQL

Unidata

Informix

MS Access

Popular Relational Database Systems

Enterprise Systems:

- Oracle 8i and 9i
- MS SQL Server 2K
- IBM DB2
- Informix (now IBM)

“Desktop” Systems:

- MySQL
- PostgreSQL
- (MS Access)

Warmup: The ACID Test

Atomicity:

Suppose you have a transaction comprised of an UPDATE, an INSERT and a DELETE statement. Start the transaction, and yank the plug from the wall. In what state is your data now? Atomicity means that either the three statements changed all your data, or none of it.

Consistency:

When the transaction is complete, both the data and internal structures are correct, and accurately reflects the changes effected by the transaction.

Isolation:

One transaction cannot interfere with the processes of another concurrent transaction.

Durability:

All changes are permanent at the end of the transaction.

Warmup: Enterprise Systems

- ALWAYS pass the ACID test with flying colors
- Sophisticated Query Optimization
  - Extensive indexing, usually supports many kinds of indexes
  - Automatic, advantageous use of multiple processors
- Support for Multi-Server Implementations (“Clustering”)
- Enormous Data Capacity (gigabyte, terabyte, more)
  - Can support terabyte (10^12 bytes) or better
  - Extensive data partitioning support
- Reliable Backup Utilities
  - Automated and Schedulable
  - Full or Incremental
- Data-Replication Tools
- Procedural application languages (PL/SQL, T-SQL, 4GL, etc)
  - Usually run in the same application space as the server
- Detailed transaction management methods
- Tools for bulk-loading data
- Extensive options for user management and permissions (separate, and/or in conjunction with OS user permissions)
- Licensed by User or CPU: EXPEN$IVE

Warmup: “Desktop” Systems

- May not pass ACID (until recently, MySQL did not support transactions)
- Usually limited support for SQL 92 (“The” standard for SQL syntax)
    - 80/20 rule: designers implement 20% of the instructions, which are used 80% of the time (again until recently, MySQL did not support nested relationships)
- Cannot usually handle very large databases easily, many cannot handle moderate (1GB+)
    - Don’t support clustering
    - BUT: PostgreSQL has been reported to handle a few 4 TB systems
- Lack sophisticated and/or scalable query optimization
    - Works well, even stellar on small to medium datasets, and/or simple table joins, but performance degrades with increasing data volume or join count
    - Not multi-processor aware
- Typically, provide limited or no procedural languages (exception: postgreSQL)
- Lack sophisticated, very reliable backup utilities tested on enormous data loads (critical failing)
- Do not support multi-server installations
- Limited/Lacking support for data partitioning (necessary for effective RAID usage)

This is my motorcycle!

It is…

Shiny…

Red…

And Fast!! (varoooooom)

Assessing Differences

Architecture

Performance

Maintenance

Application Support (PL/SQL, Java, PHP, etc)

Scalability

Platform Support

A Method

Port (or imagine porting) a richly featured Oracle database to PostGres, MySQL, and SQL Server 2K. What changes must be made to the DDL? To the SQL queries? To the data itself?

The database should feature, at least:

- 6-12 tables
- 1 table of gigabyte size
- 1 table with blobs

Port Oracle to SQL Server?

Common Syntax: very little. This is a big job.

DDL	Some important differences on tablespace distribution. MS uses “Identity” for Oracle “Sequence” Etc.
SQL	Mostly similar – both have robust support of SQL 92. Some changes for “EXCEPT” and “MINUS”.
Procedural Languages	Requires a complete rewrite. Different string and date manipulation functions, procedures and cursors are declared slightly differently. T-SQL != PL/SQL
Architecture	Again some important differences. SQL Server user is not really equivalent to an Oracle user: by default Oracle users have their own schema and tablespace; in SQL everyone is assigned roles, shares a tablespace unless specifically changed.

Note: MS Data Transformation Services does ease the manual work considerably.

Oracle -> PostGreSQL

DDL	PostGreSQL is pretty friendly with Oracle. There will be warnings about syntactical differences. Sequences may fail on you.
SQL	Almost identical
Procedural Languages	Again almost identical. We'll see some differences in a minute.
Architecture	Be wary of user/schema relationships. PG is more like SQL Server here. Tablespace distributions might need to be changed as well to take advantage/avoid PG's implementation of large-file support. (older versions supported only 8K rows, by the way – this could be upped to 32 at compile time, but with a speed hit. The new version presumably fixes this).

Oracle -> MySQL

DDL	Views, Triggers, Stored Procedures, some foreign keys will fail. Beyond that most syntax should run.
SQL	Almost identical
Procedural Languages	N/A
Architecture	Be sure to use InnoDB or BerkeleyDB, not Classic. MySQL should be flexible with regard to the Oracle architecture.

Porting Applications

What are the problem points?

- The procedural language syntax might be different
- The interface drivers (DBI/DBD, JDBC, ODBC) might not be supported as well
- Query parsers might be called differently
- Parsers may handle large result sets differently

Applications, Oracle -> PostGreSQL

PostGreSQL comes packaged with PL/pgSQL, its own version of Oracle’s PL/SQL. It is ALSO (rather handily) packaged with handlers for C, Perl, Tcl and Python.

But: procedures are executed in external C modules. Not the database instance itself (unlike Oracle).

Here are some important differences from Oracle’s PL/SQL:

- No default values for parameters
- You can overload function names
- No need for cursors, use FOR instead (cursors are supported and though the syntax is slightly different, PG recognizes Oracle syntax too)
- Single quotes must be escaped in the procedure body
- Use schemas instead of packages to group your functions.

Applications, Oracle -> MySQL

At this time, MySQL supports no procedural SQL-based languages, stored procedures, or triggers. Ad-hoc queries only (via a query tool, Java, C, Tcl,etc application).

MySQL expects to support a form of PL/SQL, procedures and triggers with version 5.1.

You need to write applications in your (supported) language of choice to query MySQL (via DBI/DBD, ODBC, JDBC, etc)

MySQL: SQL92 Compliance and other Issues

Departures from SQL 92	Known Problems that are Likely to Appear:
- Sub Queries supported in InnoDB only - Transactions – InnoDB only - Foreign Keys – InnoDB only, not checked during bulk loads - No Views -- comment issues (supported only lately, must have trailing space)	- Table corruption during bulk loading – platform dependant - Bulk loader is new - Multiple LEFT JOIN and RIGHT JOINs in a query can return incorrect results - Replication fails - Cannot refer to temporary tables within a query more than once. This fails: SELECT * FROM temp_table, temp_table as T2

Performance

Obviously, along with Maintenance and Scalability, this is the metric we care about most.

- MySQL is extremely fast.

- Oracle is extremely scalable

PG vs. MySQL

Tim Perdue's results:

PostGreSQL 7.0.2

```
concurrency w/pconnects:
10 cli - 10.27 pg/sec 333.69 kb/s
20 cli - 10.24 pg/sec 332.86 kb/s
30 cli - 10.25 pg/sec 333.01 kb/s
40 cli - 10.0 pg/sec 324.78 kb/s
50 cli - 10.0 pg/sec 324.84 kb/s
75 cli - 9.58 pg/sec 311.43 kb/s
90 cli - 9.48 pg/sec 307.95 kb/s
100 cli - 9.23 pg/sec 300.00 kb/s
110 cli - 9.09 pg/sec 295.20 kb/s
120 cli - 9.28 pg/sec 295.02 kb/s (2.2% failure)

concurrency w/10% inserts & pconnects:
30 cli - 9.97 pg/sec 324.11 kb/s
40 cli - 10.08 pg/sec 327.40 kb/s
75 cli - 9.51 pg/sec 309.13 kb/s
```

MySQL 3.22.3

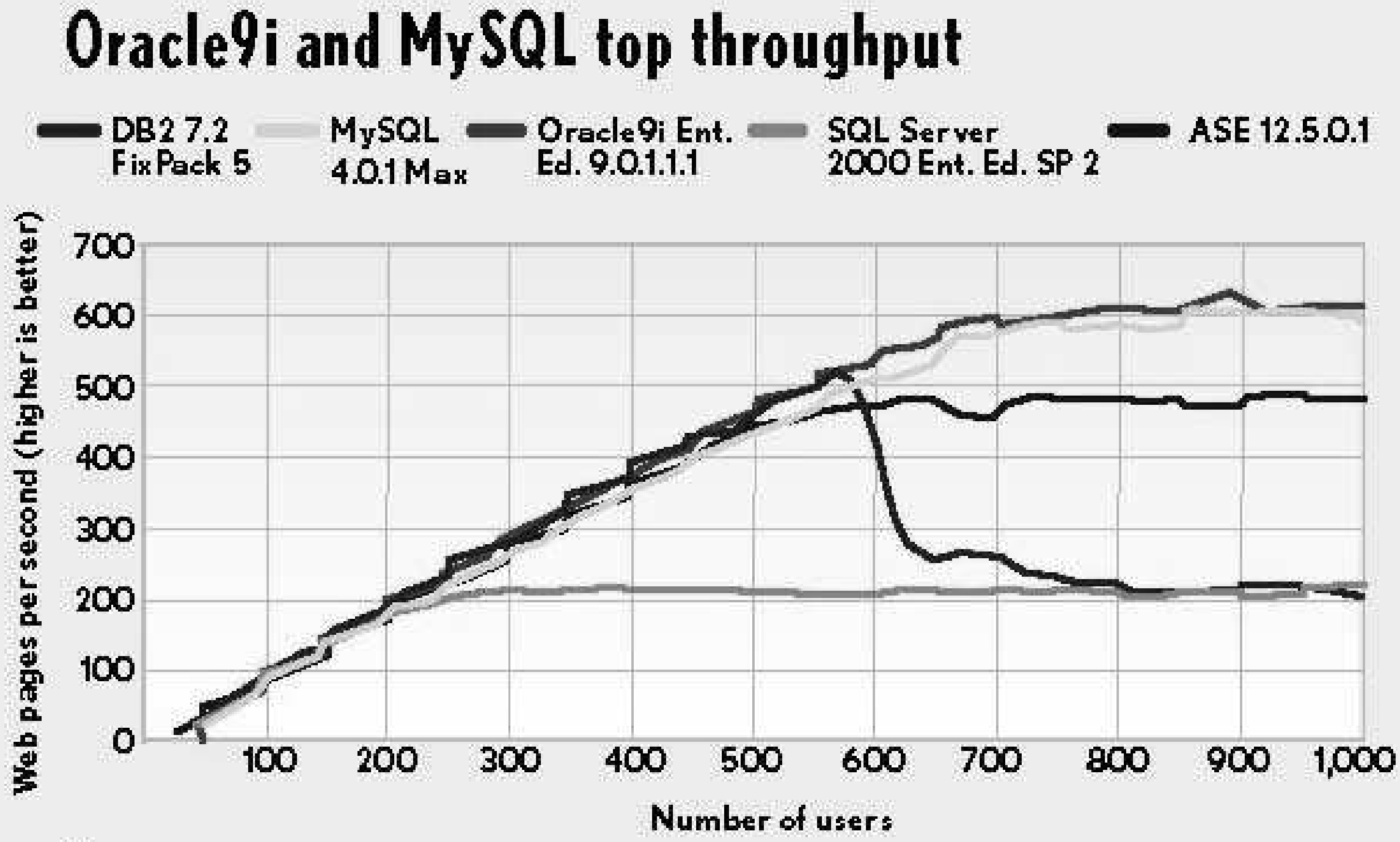
```
Concurrency Tests w/pconnects:
30 cli - 16.03 pg/sec   521.01 kb/s
40 cli - 15.64 pg/sec   507.18 kb/s
50 cli - 15.43 pg/sec   497.88 kb/s
75 cli - 14.70 pg/sec   468.64 kb/s
90 - mysql dies
110 - mysql dies
120 - mysql dies

concurrency w/10% inserts & pconnects:
20 cli - 16.37 pg/sec 531.79 kb/s
30 cli - 16.15 pg/sec 524.64 kb/s
40 cli - 22.04 pg/sec 453.82 kb/sec 378 failures
```

Postgres scales roughly 3x's higher for concurrent connections…

But MySQL is 3xs as fast

Source: http://www.phpbuilder.com/columns/tim20000705.php3?page=2

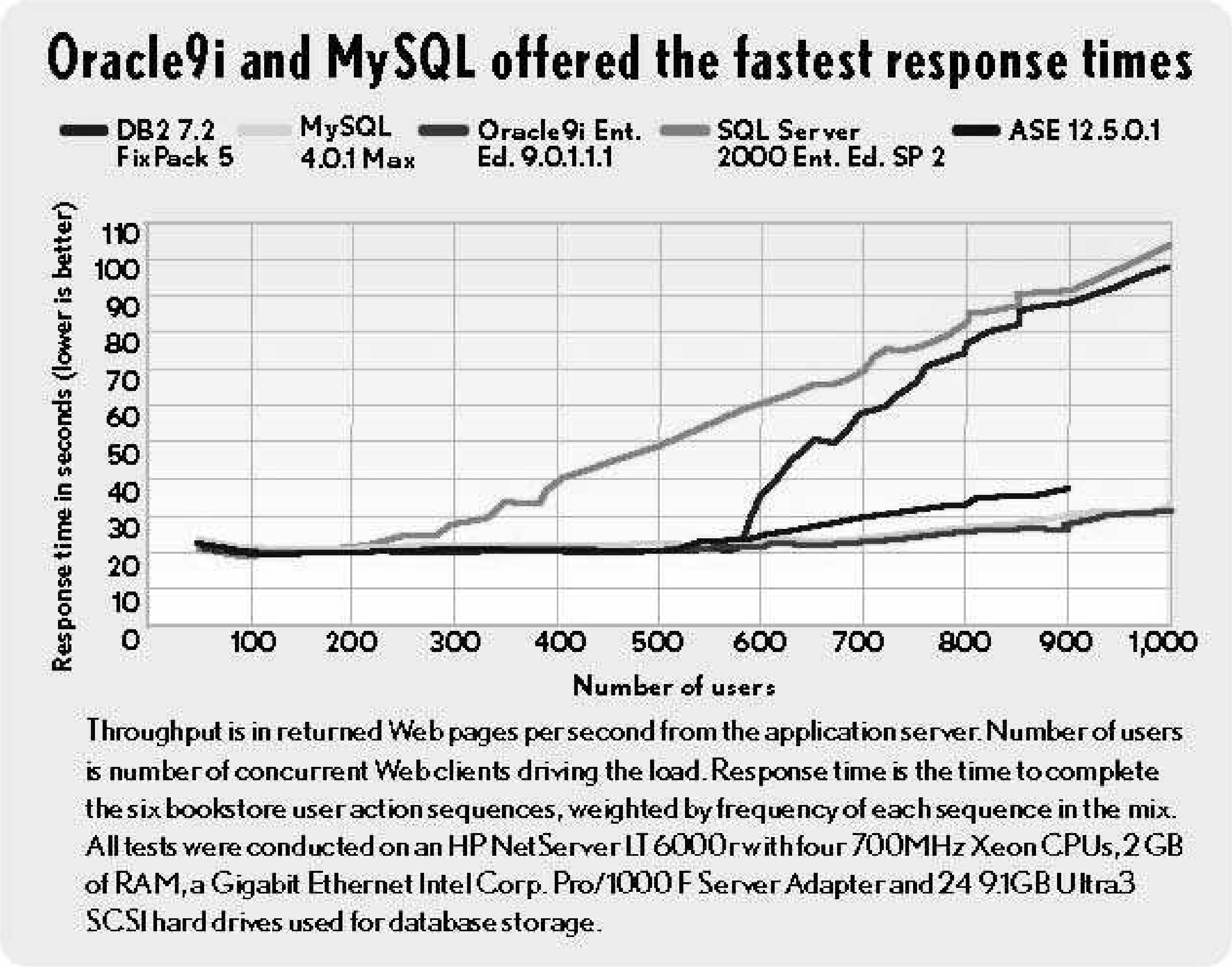
Throughput Benchmark



Throughput is in returned Web pages per second from the application server. Number of users is number of concurrent Web clients driving the load. Response time is the time to complete the six bookstore user action sequences, weighted by frequency of each sequence in the mix. All tests were conducted on an HP NetServer LT 6000r with four 700MHz Xeon CPUs, 2GB of RAM, a Gigabit Ethernet Intel Corp. Pro/1000 F Server Adapter and 24 9.1GB Ultra3 SCSI hard drives used for database storage.

Source: eWeek (http://www.eweek.com)

Response Times



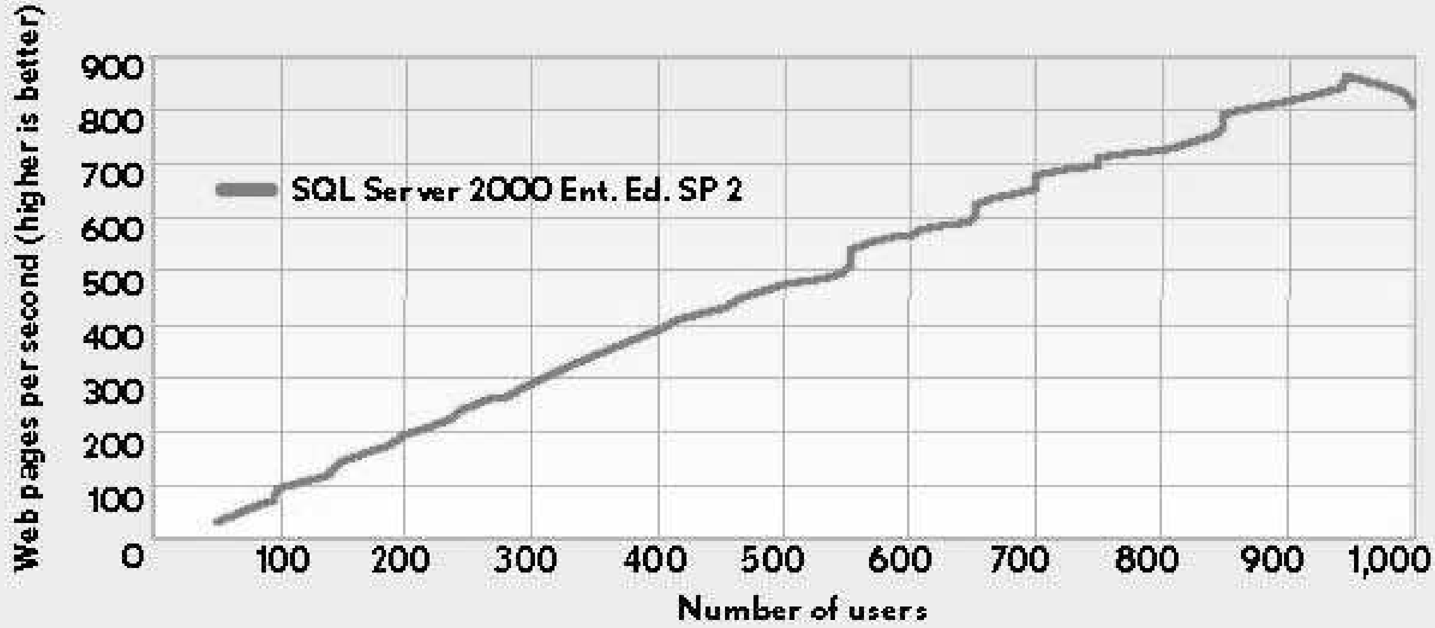
Oracle9i and MySQL offered the fastest response times

Throughput is in returned Web pages per second from the application server. Number of users is number of concurrent Web clients driving the load. Response time is the time to complete the six bookstore user action sequences, weighted by frequency of each sequence in the mix. All tests were conducted on an HP NetServer LT 6000r with four 700MHz Xeon CPUs, 2 GB of RAM, a Gigabit Ethernet Intel Corp. Pro/1000 F Server Adapter and 24 9.1GB Ultra3 SCSI hard drives used for database storage.

Source: eWeek (http://www.eweek.com)

SQL Server on MS

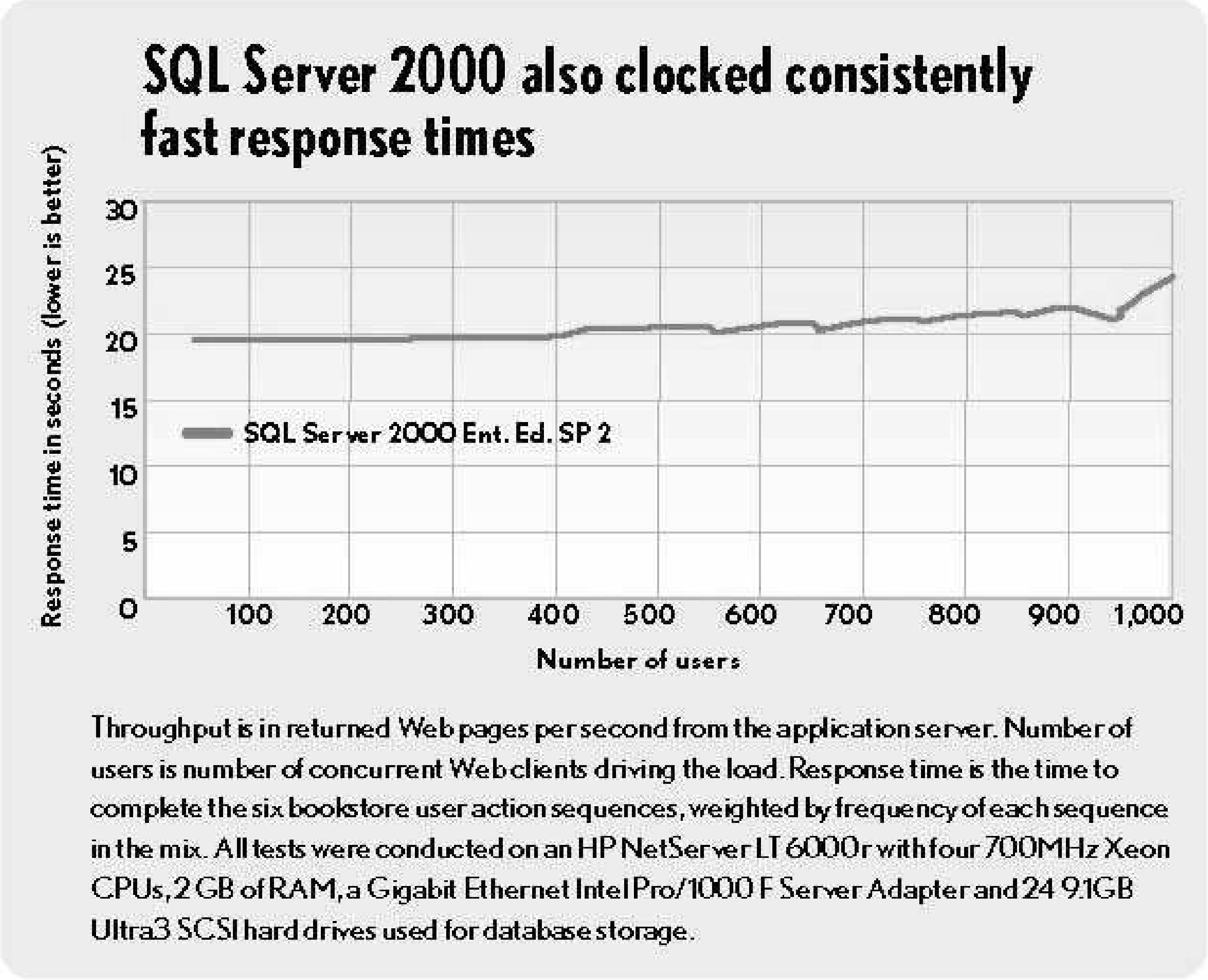
SQL Server 2000 performed very well in ASP .Net-based test on an all-Microsoft software stack



Throughput is in returned Web pages per second from the application server. Number of users is number of concurrent Web clients driving the load. Response time is the time to complete the six bookstore user action sequences, weighted by frequency of each sequence in the mix. All tests were conducted on an HP NetServer LT 6000r with four 700MHz Xeon CPUs, 2 GB of RAM, a Gigabit Ethernet Intel Pro/1000 F Server Adapter and 24 9.1GB Ultra3 SCSI hard drives used for database storage.

Source: eWeek (http://www.eweek.com)

SQL Server Response Times



SQL Server 2000 also clocked consistently fast response times

Throughput is in returned Web pages per second from the application server. Number of users is number of concurrent Web clients driving the load. Response time is the time to complete the six bookstore user action sequences, weighted by frequency of each sequence in the mix. All tests were conducted on an HP NetServer LT 6000r with four 700MHz Xeon CPUs, 2 GB of RAM, a Gigabit Ethernet Intel Pro/1000 F Server Adapter and 24 9.1GB Ultra3 SCSI hard drives used for database storage.

Source: eWeek (http://www.eweek.com)

MySQL: Pricing

MySQL Pro

MySQL Pro includes the InnoDB transactional storage engine, which provides row-level locking.

Number of licenses	Price per copy		
	EUR	USD	GBP
1 .. 9	440.00	495.00	290.00
10 .. 49	315.00	360.00	205.00
50 .. 99	255.00	290.00	165.00
100 .. 249	195.00	220.00	127.00
250 .. 499	155.00	175.00	100.00
500 +	ask for quote, sales@mysql.com		

Price is given per Server (not per CPU)

Source: http://www.mysql.com/products/pricing.html

PostGreSQL: Pricing

Free.

Some Actual Pricing!

(according to Microsoft)

Oracle9*i* Enterprise Edition and SQL Server 2000 Enterprise Edition

With Management Tools, Advanced Security Features, and Business Intelligence Features (May 2003)

# CPUs	Oracle9*i* Enterprise	SQL Server 2000 Enterprise Edition
1	$96,000 US	$19,999 US
2	$192,000 US	$39,998 US
4	$384,000 US	$79,996 US
8	$768,000 US	$159,992 US
16	$1,536,000 US	$319,984 US
32	$3,072,000 US	$639,968 US

Source: http://www.microsoft.com/sql/evaluation/compare/pricecomparison.asp
Date: 9/25/2003

Some Words on Market Acceptance

Oracle and SQL Server	Pretty much control the Fortune 100 market (Oracle=51%), sharing space with DB2
PostGreSQL	Widely used by experienced DBAs seeking affordable Oracle-like solutions. Enjoys formal and rapid development. Large user base across all platforms.
MySQL	Probably has a *larger user base* than PostGres. Blinding performance makes it a popular solution for simple databasing needs and high-read-to-write ratio systems. Also enjoys formal development, a large development community, a very large user base (also across all platforms). Is now supported through formal support contracts.

Case for Middleware and Stored Procedures

Because the workers developing Flash and web media code are not typically database experts, its usually desirable (to say the least) to separate functional logic from presentation code.

One way would be to have your front end developers hand the code to the DBA to implement. But this is clearly not a good solution.

Another is to leave your logic in stored procedures. The only calls your developers make to the database are via procedure calls. But this requires them to manipulate data after they get it.

So the method that has become very common is to establish one or more layers of middleware code that separates the presentation layer, from the data formatting layer, from the data retrieval layer, from the database.

This also makes it easier to compartmentalize your application across many machines

Bottom Line

If your Shop is…	**Use…**
UNIX, Enormous Enterprise	Oracle
NT, Enormous Enterprise	SQL Server
Unix or NT, transaction-heavy data, or require extensive RI (bank, complicated records, etc)	PostGreSQL
Same as above, but speed is king and you're doing mostly reads anyway. Or, you just need an easy, well documented DB	MySQL

Cost: The **REAL** Bottom Line

The quality of your design reigns all.

- DESIGN: **Is it a good fit for your DBMS? Does your DBMS match your existing hardware and software?**
- ISOLATION: Disciplined separation of development from production systems.
- STELLAR DBA(s): Extremely well-read, intelligent, meticulous, carefully authoritative, involved in application development.
- CAREFUL DEVELOPERS: Constantly mindful of the DBMS strengths and weaknesses, table sizes and indexing, query efficiency, interactive with DBA