
Subject: Database performance test

Posted by [Novo](#) on Sun, 16 Sep 2007 01:40:34 GMT

[View Forum Message](#) <> [Reply to Message](#)

I'd like to post a simple database performance test, which is designed after <http://www.sqlite.org/cvstrac/wiki?p=SpeedComparison>

It can be interesting for those who wants to compare performance and memory usage of different databases and database access technologies.

I've attached the test itself and four database drivers:

- 1) Firebird 1.5 (embedded);
- 2) MS JET ADO;
- 3) MS JET ODBC;
- 4) SQLITE (which is an embedded sql server);

A database for Firebird is also included.

Command lines to run:

```
perf_sdb_test.exe --conn_str="/SQLITE" --db_name=test.sqlite  
perf_sdb_test.exe --conn_str="/SQLITE_EMB" --db_name=test.sqlite
```

```
perf_sdb_test.exe --conn_str="/ADO/Microsoft.Jet.OLEDB.4.0" --db_name=test.mdb  
perf_sdb_test.exe --conn_str="/ODBC/Microsoft Access Driver (*.mdb)" --db_name=test.mdb
```

```
perf_sdb_test.exe --conn_str="/InterBase_EMB" --db_name=TEST01.FDB --user_name=test  
--user_passwd=test
```

This test also demonstrates that DLLs on Windows is not hell at all.

Have fun!

File Attachments

1) [perf_sdb_20070915.zip](#), downloaded 464 times

Subject: Re: Database performance test

Posted by [zsolt](#) on Sun, 16 Sep 2007 20:14:06 GMT

[View Forum Message](#) <> [Reply to Message](#)

Can you post your results as well?

Subject: Re: Database performance test

Posted by [Novo](#) on Mon, 17 Sep 2007 00:36:43 GMT

[View Forum Message](#) <> [Reply to Message](#)

zsolt wrote on Sun, 16 September 2007 16:14 Can you post your results as well?

Results for SQLITE ...

Sdb performance tests ...

Test INSERT 01. Insert 4000 records. Autocommit.

INSERT INTO t1 VALUES(:?, :?, :?)

0.032 sec.

Test INSERT 02. Insert 100000 records. Inside of a transaction.

INSERT INTO t2 VALUES(:?, :?, :?)

2.109 sec.

Test INSERT 03. Insert 100000 records into an indexed table. Inside of a transaction.

INSERT INTO t3 VALUES(:?, :?, :?)

4.125 sec.

Test SELECT 02. Select 100 times without an index. Fetch result.

SELECT count(*), avg(b) FROM t2 WHERE b >= :? AND b < :?

7.578 sec.

Test SELECT 03. Select 100 times without an index on a string comparison. Fetch result.

SELECT count(*), avg(b) FROM t2 WHERE c LIKE :?

10.688 sec.

Test INNER JOIN 01. Without an index. Fetch result.

SELECT t1.a FROM t1 INNER JOIN t2 ON t1.b = t2.b

238.02 sec.

Test CREATE INDEX 01. Run 1 time on an ordered field.

CREATE INDEX i2a ON t2(a)

0.672 sec.

Test CREATE INDEX 02. Run 1 time on a nonordered field.

CREATE INDEX i2b ON t2(b)

0.688 sec.

Test SELECT 04. Select 1000 times with an index. Fetch result.

SELECT count(*), avg(b) FROM t2 WHERE b >= :? AND b < :?

0.953 sec.

Test UPDATE 01. Update 400 times without an index. Inside of a transaction.

UPDATE t1 SET b = b*2 WHERE a >= :? AND a < :?

0.734 sec.

Test UPDATE 02. Update 4000 times with an index. Inside of a transaction.

UPDATE t2 SET b = :? WHERE a = :?

0.125 sec.

Test UPDATE 03. Update 4000 times a text value with an index. Inside of a transaction.

UPDATE t2 SET c = :? WHERE a = :?

0.078 sec.

Test INSERT from a SELECT 01. Inside of a transaction.

INSERT INTO t1 SELECT * FROM t2

0.938 sec.

Test INSERT from a SELECT 02. Inside of a transaction.

INSERT INTO t2 SELECT * FROM t1

4.437 sec.

Test INNER JOIN 02. With index on one side. Fetch result.

SELECT t1.a FROM t1 INNER JOIN t2 ON t1.b = t2.b

0.719 sec.

Test SELECT 05. Select 100 times with subqueries. Subquery is using an index. Fetch result.

SELECT t1.a FROM t1 WHERE t1.b IN (SELECT t2.b FROM t2 WHERE t2.b >= :? AND t2.b < :?)

28.828 sec.

Test DELETE 01. Run 1 time without an index.

DELETE FROM t2 WHERE c LIKE '%fifty%'

0.172 sec.

Test DELETE 02. Run 1 time with an index.

DELETE FROM t2 WHERE a > 10 AND a < 95000

2.25 sec.

Test DELETE 03. A big DELETE followed by many small INSERTs.

Run 4000 times. Inside of a transaction.

DELETE FROM t1

INSERT INTO t1 VALUES(:?, :?, :?)

0.437 sec.

File Attachments

1) [perf_sdb_results.zip](#), downloaded 397 times
