Firebird OLTP Test

Pavel Zotov, Firebird QA
THANK YOU!
ABOUT THIS TEST

• Emulates work of real-life app (car service)
  – Settings for init pop., warm-up & measure.
• Does not require 3rd party utilities.
• Workload modes: small, medium, heavy.
• Main purpose: get maximal performance.
• Performance report auto creating.
  – Log every unit run and its result.
• Test home: www.firebirdtest.com (to be published)
• Author: Pavel Zotov, p519446@yandex.ru
WHY THIS TEST

• Stress-test of Firebird stability
• Logic as in real-life, workload much harder
• Compare performance:
  * “hardware-1” vs “hardware-2”
  * Firebird 3.0 vs 2.5
  * SS vs SC vs CS
    * database settings: page_size, FW, etc
• Create client-side app for distribution as example (planning).
MODEL: INTRO

Main entities:

• Catalogue of products & shopping cart
• Contractor
• Document header
• Document line
• Turnover log
• Aggregated remainder
• FIFO distribution: source & target
1. **Only INSERTS** occur in turnover log
2. Aggregating is “serialized” with high frequency
MODEL: CONTRACTORS

**CONTRACTOR**

- **Id**
- **Name**
- **flag**

**our firm**

**external**

**COST TURNOVER LOG**

- **ID** (for 1NF)
- **Contractor_id**
- **Operation_id**
- **Cost_total (+ / - )**
- **Timestamp**

**COST AGGREGATED REMAINDEERS**

- **Contractor_id**
- **Saldo-1** (for supplier)
- **Saldo-2** (for customer)

1. **Only INSERTS** occur in turnover log
2. Aggregating is “serialized” among conns.
3. “Our firm” => NO reserve after shipping
DATABASE SCHEMA

- Documents flow (operations)
- Turnovers and balances
- Effect from operations
- Producer - consumer logic
START: client order

Our order to supplier

Invoice from supplier

Invoice accept?

client reserve?

FINISH: shipping

Our (pre-) payments

Invoice

client reserve?

OUT

OUT

OUT
TURNOVERS AND BALANCES

STOCK BALANCES:
- CLO: client order
- ORD: order to supplier
- SUP: invoice from supplier
- AVL: available
- RES: reserve for client

STOCK TURNOVERS:
- INC: total incomings
- OUT: total outgoings

ACCOUNTANT BALANCE:
= INC – OUT

MONETARY BALANCE & TURNOVER:
- balance of contractor as supplier or as customer:
  is calculated in purchasing or retailing prices
- (pre-) payment to supplier / (pre-)payment from client
CHANGES OF REMAINDERS

- Catalogue -> **Client** order
- Supplier **order**: gather rows from client orders
- Supplier invoice: gather rows from **supplier** orders
- Check and accepting invoice: add its content to **available** remainders
- Search for client orders that still need some goods to be **reserved**
- Create **reserve**
- Products shipping to customers
### EFFECT FROM OPERATIONS

<table>
<thead>
<tr>
<th>BUSINESS OPERATION</th>
<th>CLO</th>
<th>ORD</th>
<th>SUP</th>
<th>AVL</th>
<th>RES</th>
<th>INC</th>
<th>OUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUSTOMER ORDER</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OUR ORDER TO SUPPLIER</td>
<td>-1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WE GET INVOICE FROM SUPPLIER</td>
<td>-1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>WE VERIFIED &amp; ACCEPT INVOICE</td>
<td>-1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>RESERVE FOR CLIENT</td>
<td>-1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SALE OF PRODUCTS</td>
<td>-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

**Legend:**

- **CLO** = remainder in customer orders
- **ORD** = remainder in orders to supplier
- **SUP** = not-delivered invoices
- **AVL** = ‘on-hand’ remainder
- **RES** = remainder of reserved goods
- **INC** = total incomings
- **OUT** = total outgoings
Producer-Consumer: why?

Too many row-level lock conflicts:

- When need to update remainder
- When need to change amount in document line
- When need to change total cost in document header

Performance impact:

Earlier test versions: approx. 80% of application unit calls failed with lock conflict.
No sense to measure performance in this case.
Producer-Consumer: BALANCES

Attempt to apply same schema as for contractor balances:

- DOC_LINES:
  - INSERT …
  - UPDATE …
  - DELETE …

- TURNOVER LOG

- SCAN LOG & AGGREG.

- AGGREGATED REMAINDERS

- do **inserts only** into turnover log
- only **one attach** runs aggregation
- clear turnover log after aggregation finishes

**Q:** how to provide constraint “REMAINDER >= 0”?
Producer-Consumer: QUANTITIES

Producer:

```
INSERT INTO DOC_LINES(..., QTY)
VALUES(..., 5);
```

Source for future qty spreading:

```
ADD 5 ROWS FOR QTY = 5
```

Consumers:

- Tx1: wants Q=3
- Tx2: wants Q=2
- Tx3: wants Q=4

FREE

LOCKED

Don’t allow to take more than source can give!
Producer-Consumer: DOC TOTALS

Producer:
DOC_HEADER
INSERT(..., COST) VALUES(..., 5400)

Source for future cost spreading:

id=1 $1000
id=2 $1000
id=3 $1000
id=4 $1000
Id=5 $1000
Id=6 $400

Consumers:

Tx1: wants $2100
Tx2: wants $1700
Tx3: wants $800

FREE

LOCKED
PRODUCER–CONSUMER: OVERALL

BASE OPER.

CONTRACTOR

CATALOGUE

DOC. HEADER

COST = P, SPLIT IT ON 10 PARTS

DOC. LINES

QTY = N

SOURCE FOR COST “SPREADING”

RESULT OF COST “SPREADING”

MONETARY TURNOVER LOG

SOURCE FOR QUANTITY “SPREADING”

RESULT OF QUANTITY “SPREADING”

STOCK TURNOVER LOG
HOW TEST WORKS

• Phases of test run

• Sketch of measurement

• Auto make performance report
PHASES OF TEST RUN

Read config:
get “init_docs“
=> save to “A”

How many docs
now we have ?
=> save to “B”

“A” > “B”
YES

NO
Fill up to
“A”

REPORT

MEASURE

WARM-UP

MON$ - 2

MON$ - 1
Big script with ~300 transactions
Repeat this after finish (loop in .bat)
Batch checks whether one need to exit from loop and terminate itself
Read config:
get \%tmpdir\% setting (path)

1run_oltp_emul.bat

start

worker

worker

worker

worker

1

ISQL

2

ISQL

3

ISQL

4

ISQL

Report

File: \%TMPDIR\% \oltp_NN_performance

DB

Report SP
PERFORMANCE: RESULTS

• Performance: how to measure?

• Results:
  performance overall
  dynamic change of performance
  explanation

• What it was tested?
  Hardware, Firebird & database settings

• Graphics
PERFORMANCE: IN WHAT “UNITS”? 

Performance rating:  \( P = \frac{S}{M} \),

where:

\( S = \text{number of successfully completed actions, when gdscode is NULL} \)

\( M = \text{durability of workload period, in minutes} \)
PERFORMANCE: REPORTS

Following reports can be created:

• 1) overall;

• 2) dynamic (“how Firebird gets tired”);

• 3) detailed;

• 4) exceptions occurred
WHAT IS WAS TESTED?

- Server:
  12 core CPU, 2GHz, RAM 32 Gb HDD IBM SCSI
  OS: Linux RHEL, kernel 2.6.39

- Firebird versions: 2.5 SS, 2.5 SC, 3.0 SS, 3.0 SC
- Database settings:
  FW = ON and OFF
  page_size = 8192

Number of attaches: 25, 50, 100, 150

- Initial number of documents: 30000. Database size: ~410 Mb
- Database warm-up time: 10 minutes (3.0), 15 minutes (2.5)
- Measured time: mostly 180 minutes, several times 12 hours
Changes in firebird.conf

Following parameters need to be changed:

• **ExternalFileAccess = Restrict <path>**  
  (place when ‘STOPTEST.TXT’ will live)

• **DefaultDBCachePages**  
  increase at least to 512 for SC or CS  
  increase at least to 65535 for SS

• **LockHashSlots**  
  increase to 22111

• **TempCacheLimit**  
  increase at least to 256M
RESULTS - PERFORMANCE OVERALL

Successful business actions per minute, in average:

Number of attaches: 100
Warm-up time: 10 minutes
Measurement time: 180 minutes

DefaultDbCachePages = 512K
LockHashSlots = 22111
TempCacheLimit = 2 Gb
How Firebird Gets Tired When FW = ON

Time, minutes

sop / min

2.5 SS
2.5 SC
3.0 SC
3.0 SS

(succesful business actions per minute, in average)
How Firebird Gets Tired When FW = OFF

(succesful business actions per minute, in average)
Attaches: 100. Warm-up: 10 min. Measure: 180 min.

Table: **producer** of quantity for FIFO distribution

**Total Versions / Total Records, %:**

![Bar chart showing total versions and records for different quantities.]

- **3.0 SC, FW = ON**
- **3.0 SS, FW = ON**
- **2.5 SC, FW = ON**
- **2.5 SS, FW = ON**
Attaches: 100. Warm-up: 10 min. Measure: 180 min.

Table: **consumer** of quantity after FIFO handling

**Total Versions / Total Records, %:**

![Bar chart showing the distribution of different versions of consumers.](chart.png)

- **3.0 SC, FW = ON**
- **3.0 SS, FW = ON**
- **2.5 SC, FW = ON**
- **2.5 SS, FW = ON**
Attaches: 100. Warm-up: 10 min. Measure: 180 min.

Table: **producer** of quantity for FIFO distribution

**Total Versions / Total Records, %:**

![Bar chart showing quantity producer for 3.0 SC, FW = OFF, 3.0 SS, FW = OFF, 2.5 SC, FW = OFF, 2.5 SS, FW = OFF]
Attaches: 100. Warm-up: 10 min. Measure: 180 min.

Table: **consumer** of quantity after FIFO handling

**Total Versions / Total Records, %:**

![Chart showing consumer of quantity after FIFO handling]
BUGS & ODDITIES

Total detected bugs: more than 30 (see doc & tracker).

Not fixed yet:

Spontaneous crashes, 3.0 SC, without adding any message in firebird.log. Database appears broken after this. No bugchecks.

Bugchecks with text about ‘wrong record length’, in 3.0 only

“Page type 4 (or 5) lock denied” in firebird.log, in 3.0 only

“I/O error during read file “fb_table_***”, file exists”, in 3.0 only

Standard error messages that should be shipped to client occurs in firebird.log

Attempts of PK violations where bulk of undo occurs (when one of testing machines hangs etc). Firebird 3.0 crashes when workload more than 200 attaches.
GOOD NEWS

- Monitoring was greatly improved in 3.0;
- New monitoring counters and especially table mon$table_stats - the “golden key” in search of performance bottleneck;
- Overall impression about current 3.0: much stable than it was in aug. 2013
- Sounds like paradox but: currently 3.0 SuperServer is more stable than all others (2.5 and 3.0 SuperClassic!)
QUESTIONS?