Backend Size Matters

Serge Rielau
Principal Architect
srielau@salesforce.com
Vital Stats

> 30,000 PLpgSQL functions
> 1000 tables
> 2000 PLpgSQL triggers
  ++ types, indices, views, ...
> 200 connections

No client session affinity to DB sessions across transactions (spraying)
The FUD

“The database is *leaking* memory like a sieve!”

“We *must* recycle connections every 100 txns!”

“Recycling connections doesn’t cost *anything!*”
The facts

> 30% time spent in compiling queries and PLpgSQL (with frequent backend recycling)

Per backend memory footprint (without recycling connections)
- 140MB Catalog cache
- 550MB PLpgSQL and embedded SQL cache
- 50MB “other”

→ $750\text{MB/backend} \times 200\text{ backends} \Rightarrow 150\text{GB}$
Allocated Memory

- topallocatedsize
- plancacheallocatedsize
- plpgsqlcacheallocatedsize
- preparedplancacheallocatedsize
- catalogcacheallocatedsize
- transactionallocatedsize
- executorallocatedsize
Good news!

It’s just bloating.....

Limiting PLpgSQL function cache
• Set max number of cached function and LRU evict
• Reduced cache by half with no performance impact

Limiting catalog cache
• Set max number of cached catalog entries
• Cut size by 3 with no performance impact

More robust custom/generic plan decisions
• Rather than only compare cost also compare plan shape
  Switch to generic plan if shape (plan-id) remains unchanged

Plan source cache for dynamic SQL
• In conjunction with plan-id reduce compile time by 90%
### Memory estimator

Only fill out blue fields

<table>
<thead>
<tr>
<th>Backend</th>
<th>GUC</th>
<th>Setting</th>
<th>Measure</th>
<th>Multiplier</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>function_cache_size</td>
<td>1000Elements</td>
<td>272,000</td>
<td>272,000,000</td>
</tr>
<tr>
<td></td>
<td>plan_source_cache_size</td>
<td>1000Elements</td>
<td>127,000</td>
<td>127,000,000</td>
</tr>
<tr>
<td></td>
<td>catalog_cache_size</td>
<td>10000Elements</td>
<td>2,400</td>
<td>24,000,000</td>
</tr>
</tbody>
</table>

**Fixed**

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Other caches (relcache, ...)</td>
<td>36MB</td>
<td></td>
<td>37,748,736</td>
</tr>
<tr>
<td>Other</td>
<td>30MB</td>
<td></td>
<td>31,457,280</td>
</tr>
</tbody>
</table>

**Total per Backend**

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>469.40MB</td>
</tr>
</tbody>
</table>

**App Settings**

<table>
<thead>
<tr>
<th>maxconnections</th>
<th>100Connections</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>45.84GB</td>
</tr>
</tbody>
</table>
Food for thought

Due to spraying each backend holds mostly the same cached plans, ...

Any improvement in footprint / backend yields incremental saving

Shared caches would drop footprint by 100x!
Thank you!
srielau@salesforce.com