PostgreSQL Replication Solutions

BRUCE MOMJIAN

Replication is a complex feature. POSTGRESQL supports a variety of replication options.

https://momjian.us/presentations

Creative Commons Attribution License

Last updated: November 2021
Uses for Replication

https://www.flickr.com/photos/eugenius/
Fail Over
Data Warehousing
Load Balancing
Remote Servers
Mobile Servers
Shared Storage

• No overhead
• No data loss on fail-over
• Slave cannot execute queries
Storage Mirroring

- No overhead on master
- Synchronous or asynchronous
- Possible data loss on fail-over when using asynchronous
- Slave cannot execute queries
Streaming Replication

- No overhead on master
- Slaves can execute queries
- Possible data loss on fail-over when using asynchronous mode
- Synchronous option available
• Triggers add overhead to the master
• Possible data loss on fail-over
• Replication possible even over slow links
• Slave can execute read-only queries
• Table-level granularity allows complex data partitioning configurations
CREATE TABLE ...

Slonik

User

Origin

sl_log

User

Subscriber

Other Subscribers

CREATE TRIGGER fills sl_log

CREATE TRIGGER prevents modifications
Slony Master Switching
Bucardo

• Similar to Slony, except multi-master with conflict resolution
• Conflict resolution rules are user-configurable
Pgpool II

- Automatically load-balances read queries
- Queries with non-deterministic behavior can cause inconsistency
- Allows parallel query execution on all nodes
- Also does connection pooling and query caching

SELECT INSERT, UPDATE, DELETE, MERGE to all hosts

pgpool

INSERT, UPDATE, DELETE, MERGE to all hosts

SELECT to any host
Pgpool II With Streaming Replication

Streaming replication avoids the problem of non-deterministic queries producing different results on different hosts.
## Summary

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Most Popular Implementation</td>
<td>NAS</td>
<td>DRBD</td>
<td>Log shipping</td>
<td>Slony</td>
<td>pgpool-II</td>
<td>Bucardo</td>
<td>Table rows &amp; row locks</td>
</tr>
<tr>
<td>Communication Method</td>
<td>shared disk</td>
<td>disk blocks</td>
<td>WAL</td>
<td>table rows</td>
<td>SQL</td>
<td>table rows</td>
<td></td>
</tr>
<tr>
<td>No Special hardware required</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Allows multiple master servers</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No master server overhead</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No waiting for multiple servers</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Master failure will never lose data</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Slaves accept read-only queries</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Per-table granularity</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No conflict resolution necessary</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

https://momjian.us/presentations