A look at the Elephants Trunk

PostgreSQL 11

PostgresOpen SV 2018
San Francisco, USA

Magnus Hagander
magnus@hagander.net
Magnus Hagander

- Redpill Linpro
  - Principal database consultant
- PostgreSQL
  - Core Team member
  - Committer
  - PostgreSQL Europe
PostgreSQL 11
PostgreSQL 11

- Not done yet
  - Almost!
- Some things exist already
  - But may be removed
Development schedule

- August 2017 - branch 10
- September 2017 - CF1
- November 2017 - CF2
- January 2018 - CF3
- March 2018 - CF4
- August 2018 - Beta3
- Target: Oct 2018 - release
New features

- DBA and administration
- SQL and developer
- Backup and replication
- Performance
WAL segment size configurable

- Change from 16MB without recompile
- Useful in some high-WAL environments
- Or very constrained ones

$ initdb -D /pgdata --wal-segsize=32
pg_stat_statements

- queryid is now 64-bit
- *Much* less risk of collision
- (25% risk after 3bn instead of 50k)
- Possible breaking change!
Expression index stats

- SET STATISTICS can be done for expression index
- Defined on columns by ordinal

```sql
CREATE INDEX coord_idx ON measured (x, y, (z + t));
ALTER INDEX coord_idx ALTER COLUMN 3 SET STATISTICS 1000;
```
INCLUDE indexes

- Add extra columns to index
- Not in key
- Only used for index only scans

```
CREATE UNIQUE INDEX myidx ON mytable
    USING btree (id) INCLUDE (secondfield);
```
Automatic prewarm

- pg_prewarm
  - Already exists
- Automatically dump list
  - Regular intervals
  - Default: 5 minutes
- Automatically load on start
More default roles

- `pg_read_server_files`
- `pg_write_server_files`
- `pg_execute_server_program`
ALTER TABLE ADD COLUMN

- *With* NOT NULL DEFAULT values
- Now fast!
- Avoids rewrite
- New rows gets materialized value
- Must be non-volatile
New features

- DBA and administration
- SQL and developer
- Backup and replication
- Performance
websearch_to_tsquery

- Like phraseto_tsquery()
- But less picky
- More like typical search engines
- Quotes, AND/OR, and negation
Domain enhancements

- ARRAYs over domains
- Domains over composite types
Window frame clauses

- Now full SQL:2011 support
- RANGE BETWEEN
  - Previously, just ROWS
  - Now handles values
- Exclusion clauses
  - Exclude current row
  - Exclude ties
Window frame clauses

postgres=# SELECT i,  
    SUM(i) OVER (ORDER BY i ROWS  
        BETWEEN 2 PRECEDING AND 2 FOLLOWING),  
    SUM(i) OVER (ORDER BY i RANGE  
        BETWEEN 2 PRECEDING AND 2 FOLLOWING)  
FROM numb

<table>
<thead>
<tr>
<th>i</th>
<th>sum</th>
<th>sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>16</td>
<td>9</td>
</tr>
<tr>
<td>5</td>
<td>25</td>
<td>15</td>
</tr>
<tr>
<td>7</td>
<td>35</td>
<td>21</td>
</tr>
</tbody>
</table>
postgres=# SELECT i,
    SUM(i) OVER (ORDER BY i ROWS BETWEEN 2 PRECEDING AND 2
    EXCLUDE CURRENT ROW),
    SUM(i) OVER (ORDER BY i RANGE BETWEEN 2 PRECEDING AND 2
    EXCLUDE CURRENT ROW) FROM numbers;

<table>
<thead>
<tr>
<th>i</th>
<th>sum</th>
<th>sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>13</td>
<td>6</td>
</tr>
<tr>
<td>5</td>
<td>20</td>
<td>10</td>
</tr>
<tr>
<td>7</td>
<td>28</td>
<td>14</td>
</tr>
</tbody>
</table>
Stored procedures

- Not just void-returning functions
- SQL standard syntax
  - Uses CALL
- Transaction control
  - Not just savepoints
postgres=# CREATE PROCEDURE myproc() LANGUAGE plpgsql AS $$
BEGIN
    INSERT INTO mytable VALUES (1);
    COMMIT;
    INSERT INTO mytable VALUES (2);
    ROLLBACK;
END;
$$
Stored procedures

postgres=# CALL myproc();
CALL
postgres=# SELECT * FROM mytable;
a
---
1
(1 row)
New features

- DBA and administration
- SQL and developer
- Backup and replication
- Performance
Advance replication slots

- Without consuming
- Keep slots in sync across nodes
- Mainly for cluster management

```
SELECT * FROM pg_replication_slot_advance('test_slot', '0/1678BC8')
```
Logical replication of TRUNCATE

- Separately enabled in publication
- Published by default
Exclude unlogged tables

- Unlogged tables excluded from base backups
- Deleted on restore anyway...
- (temp tables also excluded)
Validate checksums

- Base backups validate checksums by default
- Cheap since I/O is already paid
New features

- DBA and administration
- SQL and developer
- Backup and replication
- Performance
Parallelism

- 9.6 added parallelism
- 10 made it useful
- 11 makes it even better!
Parallelism

- General enhancements
- Parallel append plan nodes
- Parallel aware hash joins
Parallell CREATE INDEX

- btrees indexes only
- Often CPU bound
  - Much faster now!
- max_parallel_maintenance_workers=2
Partitioning

- Declarative partitioning in 10
- Syntax and basic functionality
- 11 makes it much more powerful!
Default partitions

- Where to put rows that match no other partition

```
postgres=# CREATE TABLE p_def PARTITION OF p DEFAULT;
CREATE TABLE
```
Allow UPDATE to move rows

- Change the value in partition key
- Previously only within partition
- Including in and out of default
- Some concurrency issues
Local partitioned indexes

- Indexes can be created on master table
- Automatically added to partitions
- Both existing and new
- Can still do individual indexes too
Cross partition UNIQUE

- UNIQUE indexes on parent
  - PRIMARY KEY
- Must include *all* partition keys
- (foreign keys only one way)
INSERT ON CONFLICT

• Now on partitioned tables
Better partition pruning

- Done in executor
- Once at start
  - For parameters
- Once at runtime
  - For subqueries etc
Hash partitioning

- Partition by automatic hash value

```sql
postgres=# CREATE TABLE p2(i int, t text)
postgres-# PARTITION BY HASH (i);
CREATE TABLE
postgres=# CREATE table p2_1 PARTITION OF p2
postgres-# FOR VALUES WITH (MODULUS 4, REMAINDER 0);
CREATE TABLE
```
Partition wise join

- Join of tables on partition key
- \textit{Identical} partition key
- Joining on complete partition key
- Default: off
Partition wise aggregates

- Partition key part of GROUP BY
- Run aggregates per partition
- Summarize at the end
Other performance
JIT compilation

- LLVM based JIT compilation
  - Availability depends on packaging
- Optimized expression processing
  - Big speedup for some analytical
  - E.g. large computational aggregates
- Automatically enabled for expensive queries
That's a lot!
There's always more

- Lots of smaller fixes
- Performance improvements
- etc, etc
- Can't mention them all!
Please help!
Please help!

- Download and test!
  - apt packages available
  - rpm/yum packages available
Thank you!

Magnus Hagander
magnus@hagander.net
@magnushagander
https://www.hagander.net/talks/