Michael Glaesemann

myYearbook.com

michael.glaesemann@myyearbook.com
We were somewhere around Barstow on the edge of the desert when the drugs began to take hold...
Visualizing Postgres
PGCon 2009
Ottawa
2009-05-21
What’s going on here?
measure
explain-analyze.info
statistics
rocket
science
O-ring damage index, each launch

26°–29° range of forecasted temperatures (as of January 27, 1986) for the launch of space shuttle Challenger on January 28

Temperature (°F) of field joints at time of launch
Tufte
ANALYZE
pg_stats
pg_class
cpu
memory
io
bloat
pg_database_size
pg_relation_size
pg_total_relation_size
pg_column_size
pg_size_size
pg_size_pretty
bloat report
CREATE VIEW utility.index_byte_sizes AS
SELECT rel.oid AS relid, pg_index.indexrelid, pg_namespace.nspname, rel.relname, idx.relname AS indexrelname,
    pg_index.indisunique AS is_key,
    ((ceil(idx.reltuples
        * (consts.index_tuple_header_size
            + consts.item_id_data_size
            + \ CASE WHEN (consts.coalesce(sum(case when statts.staattnotnull then 0 else 1 end)), 0::BIGINT)
                + ((select consts.coalesce(sum(case when atts.attnotnull then 0 else 1 end)), 0::BIGINT)
                    FROM pg_attribute atts
                JOIN (select pg_index.indkey[i] AS attnum
                        FROM generate_series(0, pg_index.indnatts - 1) i)
                WHERE atts.attrelid = pg_index.indrelid)) > 0
            THEN (select the.null_bitmap_size + consts.max_align
                    - case when (the.null_bitmap_size % consts.max_align) = 0 then consts.max_align
                    else the.null_bitmap_size % consts.max_align end
                    FROM (values (pg_index.indnatts / 8
                        + case when (pg_index.indnatts % 8) = 0 then 0 else 1 end))
                WHERE atts.starelid = pg_index.indrelid)
        ELSE 0 END) :: DOUBLE PRECISION
        + consts.coalesce(sum(statts.stawidth::DOUBLE PRECISION * (1::DOUBLE PRECISION - statts.stanullfrac)), 0::DOUBLE PRECISION)
        + consts.coalesce((select sum(atts.stawidth::DOUBLE PRECISION * (1::DOUBLE PRECISION - atts.stanullfrac))
                FROM pg_statistic atts
            JOIN (select pg_index.indkey[i] AS attnum
                        FROM generate_series(0, pg_index.indnatts - 1) i)
            WHERE atts.starelid = pg_index.indrelid), 0::DOUBLE PRECISION)
        ) / (consts.block_size - consts.page_header_data_size::NUMERIC - consts.special_space::NUMERIC) :: DOUBLE PRECISION
        + consts.index_metadata_pages::DOUBLE PRECISION)
    * consts.index_metadata_pages ::BIGINT AS ideal_idxsize,
    (idx.relpages::NUMERIC * consts.block_size ::BIGINT) ::BIGINT AS idxsize
FROM pg_index
JOIN pg_class idx ON pg_index.indexrelid = idx.oid
JOIN pg_class rel ON pg_index.indrelid = rel.oid
JOIN pg_namespace ON idx.relnamespace = pg_namespace.oid
LEFT JOIN (select current_setting('block_size', ::TEXT) :: NUMERIC AS block_size,
            case when substring(version(), 12, 3) = ANY (ARRAY['8.0': TEXT, '8.1': TEXT, '8.2': TEXT]) then 27
            else 23 end ::NUMERIC AS tuple_header_size,
            case when version() ~ 'mingw32' :: TEXT THEN 8
            else 4 end ::NUMERIC AS max_align,
            8 AS index_tuple_header_size,
            4 AS item_id_data_size,
            24 AS page_header_data_size,
            0 AS special_space,
            1 AS index_metadata_pages) ::NUMERIC AS constants
    CROSS JOIN (select pg_statistic.starelid, pg_statistic.staattnum,
            pg_statistic.stanullfrac, pg_statistic.stawidth,
            pg_attribute.attnotnull AS stattnotnull
            FROM pg_statistic atts
    JOIN (select pg_index.indkey[i] AS attnum
                        FROM generate_series(0, pg_index.indnatts - 1) i)
            WHERE atts.starelid = pg_index.indrelid)
            JOIN pg_attribute ON pg_statistic.starelid = pg_attribute.attrelid
                AND pg_statistic.staattnum = pg_attribute.attnum) ::NUMERIC AS constants
GROUP BY pg_namespace.nspname, rel.relname, rel.oid, idx.relname, idx.rel tuples, idx.relpages,
SELECT total_relsize_bytes, replace(pg_size_pretty(total_relsize_bytes), 'bytes', 'B') AS total_relsize,
relsize_bytes, replace(pg_size_pretty(relsize_bytes), 'bytes', 'B') AS relsize,
free_space_bytes, replace(pg_size_pretty(free_space_bytes), 'bytes', 'B') AS free_space,
(table_byte_sizes.free_space_bytes::numeric / table_byte_sizes.relsize_bytes::numeric)::numeric(4,3) AS bloat_rate,
idxsize_bytes, replace(pg_size_pretty(idxsize_bytes), 'bytes', 'B') AS idxsize,
(idxsize_bytes::numeric / total_relsize_bytes::numeric) AS index_rate,
toast_relsize_bytes, replace(pg_size_pretty(toast_relsize_bytes), 'bytes', 'B') AS toast_relsize,
toast_idxsize_bytes, replace(pg_size_pretty(toast_idxsize_bytes), 'bytes', 'B') AS toast_idxsize,

key_idxsize_bytes, replace(pg_size_pretty(key_idxsize_bytes), 'bytes', 'B') AS key_idxsize,
CASE WHEN key_idxsize_bytes - ideal_key_idxsize_bytes < 0 THEN 0
 ELSE key_idxsize_bytes - ideal_key_idxsize_bytes END AS free_key_idxsize_bytes,
replace(pg_size_pretty(CASE WHEN key_idxsize_bytes - ideal_key_idxsize_bytes < 0 THEN 0
 ELSE key_idxsize_bytes - ideal_key_idxsize_bytes END), 'bytes', 'B') AS key_idx_bloat_rate,

nonkey_idxsize_bytes, replace(pg_size_pretty(nonkey_idxsize_bytes), 'bytes', 'B') AS nonkey_idxsize,
CASE WHEN nonkey_idxsize_bytes - ideal_nonkey_idxsize_bytes < 0 THEN 0
 ELSE nonkey_idxsize_bytes - ideal_nonkey_idxsize_bytes END AS free_nonkey_idxsize_bytes,
replace(pg_size_pretty(CASE WHEN nonkey_idxsize_bytes - ideal_nonkey_idxsize_bytes < 0 THEN 0
 ELSE nonkey_idxsize_bytes - ideal_nonkey_idxsize_bytes END), 'bytes', 'B') AS nonkey_idx_bloat_rate,

CASE WHEN key_idxsize_bytes = 0
 OR key_idxsize_bytes - ideal_key_idxsize_bytes < 0 THEN 0
 ELSE (key_idxsize_bytes - ideal_key_idxsize_bytes)::numeric / key_idxsize_bytes END) AS numeric(4,3) AS key_idx_bloat_rate,

CASE WHEN nonkey_idxsize_bytes = 0
 OR nonkey_idxsize_bytes - ideal_nonkey_idxsize_bytes < 0 THEN 0
 ELSE (nonkey_idxsize_bytes - ideal_nonkey_idxsize_bytes)::numeric / nonkey_idxsize_bytes END) AS numeric(4,3) AS nonkey_idx_bloat_rate,

FROM utility.table_byte_sizes
LEFT JOIN (SELECT nspname, relname,
 CAST(SUM(CASE WHEN is_key THEN ideal_idxsize ELSE 0 END) AS BIGINT) AS ideal_key_idxsize_bytes,
 CAST(SUM(CASE WHEN NOT is_key THEN ideal_idxsize ELSE 0 END) AS BIGINT) AS ideal_nonkey_idxsize_bytes,
 CAST(SUM(CASE WHEN is_key THEN idxsize ELSE 0 END) AS BIGINT) AS key_idxsize_bytes,
 CAST(SUM(CASE WHEN NOT is_key THEN idxsize ELSE 0 END) AS BIGINT) AS nonkey_idxsize_bytes
 FROM utility.index_byte_sizes
 GROUP BY nspname, relname, idx_sizes USING (nspname,relname)
) idx_sizes
WHERE table_byte_sizes.nspname <> ALL (ARRAY['pg_catalog::name', 'information_schema::name'])
ORDER BY total_relsize_bytes DESC,
 free_space_bytes IS NULL,
 free_space_bytes DESC,
 relsize_bytes DESC,
bloat_rate DESC,
 idxsize_bytes DESC;
DTrace
logs
log_min_duration_statement
log_duration
log_lock_waits
deadlock_timeout
log_temp_files
log_connections

log_disconnections
track_activities
track_activity_query_size*
track_counts
track_functions*
stats_temp_directory*
log_statement_stats
log_parser_stats
log_planner_stats
log_executor_stats
LOG: EXECUTOR STATISTICS
DETAIL: ! system usage stats:
  ! 0.017621 elapsed 0.004762 user 0.000816 system sec
  ! [6.012501 user 0.336354 sys total]
  ! 0/0 [0/0] filesystem blocks in/out
  ! 0/0 [0/0] page faults/reclaims, 0 [0] swaps
  ! 0 [1] signals rcvd, 0/10 [4/14944] messages rcvd/sent
  ! 2/0 [210/0] voluntary/involuntary context switches
! buffer usage stats:
!  Shared blocks:         9 read,          0 written, buffer hit rate = 0.00%
!  Local    blocks:      0 read,          0 written, buffer hit rate = 0.00%
!  Direct blocks:       0 read,          0 written
STATEMENT: select * from posuta.index_statistics limit 1000;
LOG: duration: 42.422 ms
CSV
SELECT * from posuta.index_statistics where index_id = 265 limit 1000;
contrib
pg_freespacemap
pg_buffercache
pgrowlocks
pgstattuple
statistics

collector
pg_stat_activity

pg_locks
pg_stat_get_numscans
pg_stat_get_tuples_returned
pg_stat_get_tuples_fetched
pg_stat_get_tuples_inserted
pg_stat_get_tuples_updated
pg_stat_get_tuples_hot_updated
pg_stat_get_tuples_deleted
pg_stat_get_live_tuples
pg_stat_get_dead_tuples
pg_stat_get_blocks_fetched
pg_stat_get_blocks_hit
pg_stat_get_last_vacuum_time
pg_stat_get_last_autovacuum_time
pg_stat_get_last_analyze_time
pg_stat_get_last_autoanalyze_time
pg_stat_get_function_calls*
pg_stat_get_function_time*
pg_stat_get_function_self_time*
pg_stat_get_db_xact_commit

pg_stat_get_db_xact_rollback
pg_stat_get_bgwriter_timed_checkpoints
pg_stat_get_bgwriter_requested_checkpoints
pg_stat_get_bgwriter_buf_written_checkpoints
pg_stat_get_bgwriter_buf_written_clean
pg_stat_get_bgwriter_maxwritten_clean
snapshot
topHeapHitters
topIndexHitters
Those who cannot remember the past are condemned to repeat it.
posuta
Postgres statistics
ポスタ
Ruby
web.py
LISP IS OVER HALF A CENTURY OLD AND IT STILL HAS THIS PERFECT, TIMELESS AIR ABOUT IT.

I WONDER IF THE CYCLES WILL CONTINUE FOREVER.

A FEW CODERS FROM EACH NEW GENERATION RE-DISCOVERING THE LISP ARTS.

THESE ARE YOUR FATHER'S PARENTHESES.

ELEGANT WEAPONS FOR A MORE... CIVILIZED AGE.
clojure
compojure
(defn request-accepts-re [request re]
  (let [accept-headers (str-utils/re-split "," ((request :headers) "accept"))]
    (some #(not (= () (re-seq re %))) accept-headers)))

(defroutes posuta
  (GET "/targets/:target/databases/:database/schemas/:schema/relations/:relation/stats/analyses"
    (if (request-accepts-re request #"application/json")
      (analysis-statistics-controller/jsonp (params :callback)
        (params :target) (params :database)
        (params :schema) (params :relation)
        (params :offset) (params :duration))
      (analysis-statistics-controller/index (params :target) (params :database)
        (params :schema) (params :relation)))))

(POST "/targets/:target/databases/:database/schemas/:schema/relations/:relation/stats/vacuums"
  (if (request-accepts-re request #"application/json")
    (vacuum-statistics-controller/jsonp (params :callback)
      (params :target) (params :database)
      (params :schema) (params :relation)
      (params :offset) (params :duration))
    (vacuum-statistics-controller/index (params :target) (params :database)
      (params :schema) (params :relation)))))

(GET "/"
  (targets-controller/index))

(GET "/*
  (or (serve-file (params :*)) :next))

(ANY "/*" (error-404 (params :*))))
(defn index [target-label database-name schema-name relation-name]
  (let [title (str relation-name " vacuums")
        relation (relation/relation target-label database-name schema-name relation-name))
  (page (h (str relation-name " vacuums"))
    (html (javascript-tag (str "$(document).ready(function(){initVacuumStatistics("""
      (json-str {
        "target" target-label
        "database" database-name
        "schema" schema-name
        "relation" relation-name
      });});"""
    )))
    (html
      (link-to (relation-statistics-uri relation) relation-name)
      [:dl#charts {:class "chart"}]
      [:pre#debug]))))))

(defnjsonp [callback target-label database-name schema-name relation-name offset period-str]
  (let [day-offset (Integer/parseInt offset)
        duration (as-sql-interval (parse-iso-period period-str))
        stats (vacuum-statistics/vacuum-statistics
              target-label database-name schema-name relation-name
              day-offset duration)
        chart-data (if (empty? stats) []
                     (let [bounds (let [row (first stats)]
                                  (vector (row :lower_bound_js_epoch)
                                         (row :upper_bound_js_epoch)))
                               get-series (fn [row col-name]
                                             (vector (row :occurred_at_js_epoch) (row col-name)))
                               map-stats (fn [col-name stats] (map #(get-series % col-name) stats))]
                           (hash-map "vacuum" (map-stats :vacuum stats)
                                      "auto-vacuum" (map-stats :autovacuum stats)
                                      "bounds" bounds
                                      "label" day-offset)))]
    (jsonp-response callback chart-data)))
(defn banner []
  (html [:div#banner (link-to application-base-path
       [:img {:alt "posuta" :src "/images/logotype.png"}])]))

(defn page
  ([title body]
   (page title nil body))
  ([title head-elts body]
   (html (doctype :xhtml-strict)
     [:head [:title title]
      (include-css "/css/reset.css" "/css/layout.css"
      (include-js "/js/debug.js" "/js/jquery.js"
        "/js/jQuery.flot.js" "/js/posuta.js"
        "/js/jQuery-ui-1.7.1.custom.min.js")
     head-elts]
     (banner)
     [:body
      [:h1#title (h title)]
      [:div#content body
       [:pre#debug]]]))

(defn jsonp-response [callback content]
  (let [response (str callback "(" (json-str content ")")")]
    [200 {:headers {"Content-Type" "application/json"
                     "Content-Length" (str (.length response))
                     "X-Server" "Posuta"} response}]))
(defn vacuum-statistics
  [target-label database-name schema-name relation-name day-offset duration]
  (db/sql-query-join
   [
    "SELECT posuta.js_epoch(lower_bound + bounds.shift) AS lower_bound_js_epoch,
     "posuta.js_epoch(upper_bound + bounds.shift) AS upper_bound_js_epoch,
     "posuta.js_epoch(vacuumed_at + bounds.shift) AS occurred_at_js_epoch,
     "target, database_name, schema_name, relation_name,
     "CASE WHEN is_autovacuum THEN 0 ELSE 1 END AS vacuum,
     "CASE WHEN is_autovacuum THEN 1 ELSE 0 END AS autovacuum"
     FROM posuta.vacuums_view"
     NATURAL JOIN (SELECT target, database_name, schema_name, relation_name,"
       "(latest_occurred_at - latest.shift - CAST(? AS INTERVAL))"
       "AS lower_bound,"
       "(latest_occurred_at - latest.shift) AS upper_bound,"
       "latest.shift"
       FROM (SELECT target, database_name,"
         "schema_name, relation_name,"
         "MAX(vacuumed_at) AS latest_occurred_at,"
         "p.shift"
         FROM posuta.vacuums_view"
         NATURAL JOIN (SELECT (? * INTERVAL '24 hours') AS shift) AS p"
         GROUP BY target, database_name,"
         "schema_name, relation_name,"
         "p.shift) AS latest) AS bounds"
       WHERE (target, database_name, schema_name, relation_name) = (?, ?, ?, ?)"
       "AND vacuumed_at BETWEEN lower_bound AND upper_bound"
       "ORDER BY vacuumed_at")
    duration day-offset
    target-label database-name schema-name relation-name)
Postgres
http://postgresql.org

Ruby
http://www.ruby-lang.org

Clojure
http://clojure.org

Compojure
http://github.com/weavejester/compojure

jQuery
http://jquery.com

flot
http://code.google.com/p/flot
Visualizing Postgres

posuta

michael.glaesemann@myyearbook.com
Inspirational art by Ralph Steadman and xkcd.com.
Challenger chart by Edward Tufte.
Everything used without permission.