Terabytes of Business Intelligence: Design and Administration of Very Large Data Warehouses on PostgreSQL

Josh Berkus
josh@postgresql.org
Joe Conway
mail@joeconway.com
O'Reilly Open Source Convention
August 1–5, 2005

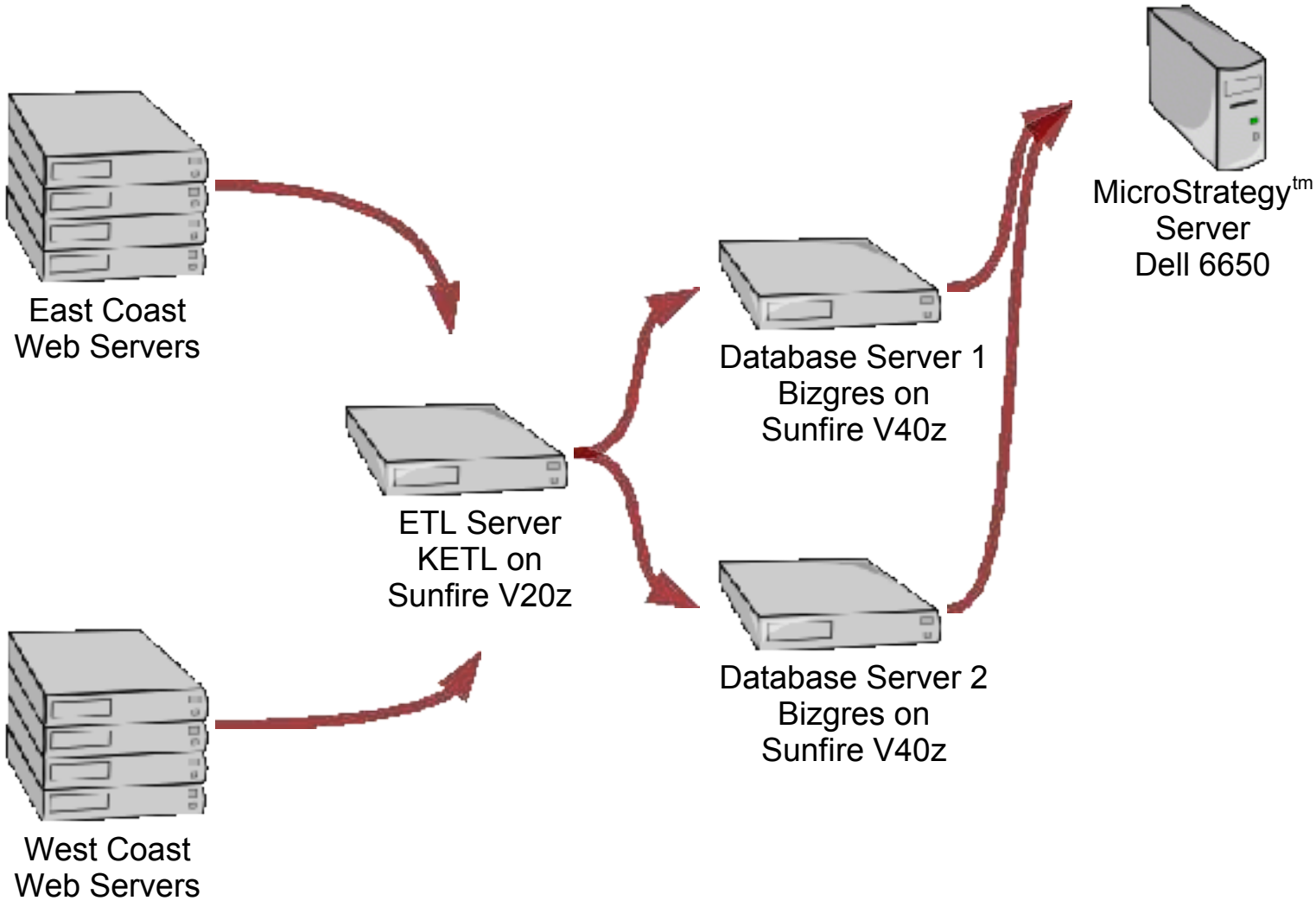
O'REILLY
OPEN
SOURCE
CONVENTION

Agenda

- Case study: weblog analysis data
  - Overview
  - Problem 1: Hardware, Setup and Configuration
  - Problem 2: Data Size
  - Problem 3: Aggregate Reports
  - Problem 4: High Availability
- Case study: equipment performance data
- Upcoming large database features

Agenda

- Case study: weblog analysis data
  - Overview
    - Clickstream Analysis
    - Weblogs from 5 websites
    - 3 million vistors per day
    - High-availibility
    - One year of data
    - Reporting with Microstrategy™
      - 14 defined reports
      - Ad-hoc reports
      - Guarenteed execution times
    - Data loaded nightly in large ETL batch
    - Hourly “intra-day” batches


East Coast
Web Servers
West Coast
Web Servers
ETL Server
KETL on
Sunfire V20z
Database Server 1
Bizgres on
Sunfire V40z
Database Server 2
Bizgres on
Sunfire V40z
MicroStrategy™
Server
Dell 6650

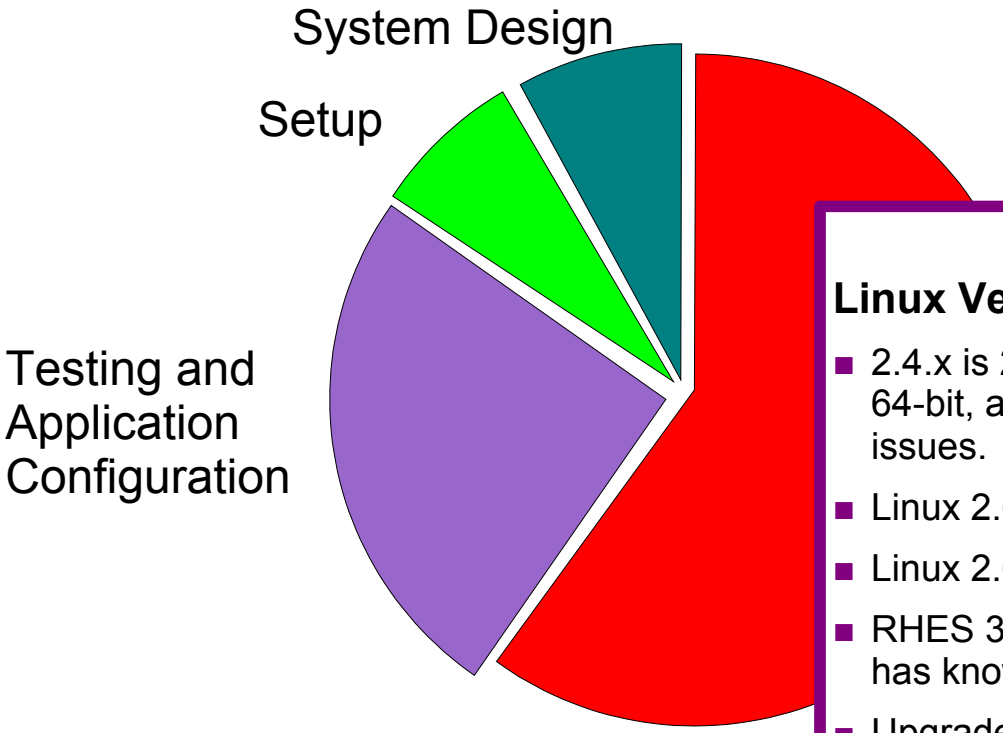
Problem 1: Hardware, Setup and Configuration

Database Servers
- Sun v40z's
- 2 Opteron 844 processors
- 12GB RAM
- 5x 72Gb 10k UW320 SCSI

Database Server Software & Config
- Disk 1:
  - RHES Linux 3 Update 4
  - Bizgres 0.6
  - Swap
  - Database temp space.
- Disk 2: Database Transaction Log
- Disks 3-5: Database, RAID0

- ETL Server
  - Sun v20z
  - 2 Opteron 244 Processors
  - 4GB RAM
  - 2x 143MB 10k UW320 SCSI in RAID-1

Time Spent On Implementing a Large Database Solution



Linux Version Heck

- 2.4.x is 20% slower than 2.6.x, especially on 64-bit, and has virtual memory management issues.
- Linux 2.6.1 to 2.6.3 was unstable.
- Linux 2.6.4 to 2.6.8 had memory leak issues.
- RHES 3 update 4 came with 2.6.9, which has known network performance issues.
- Upgraded to 2.6.11.
- *That* had issues with the fiber cards (3 weeks troubleshooting) We had to install 2.6.12 as soon as it came out!

Problem 1: Hardware, Setup and Configuration

Nine Most Important PostgreSQL.conf Settings For Business Intelligence:

shared_buffers: set to 60,000, or 470MB

work_mem: set to 128MB

maintenance_work_mem: to 512MB

Query planner settings to encourage use of indexes:

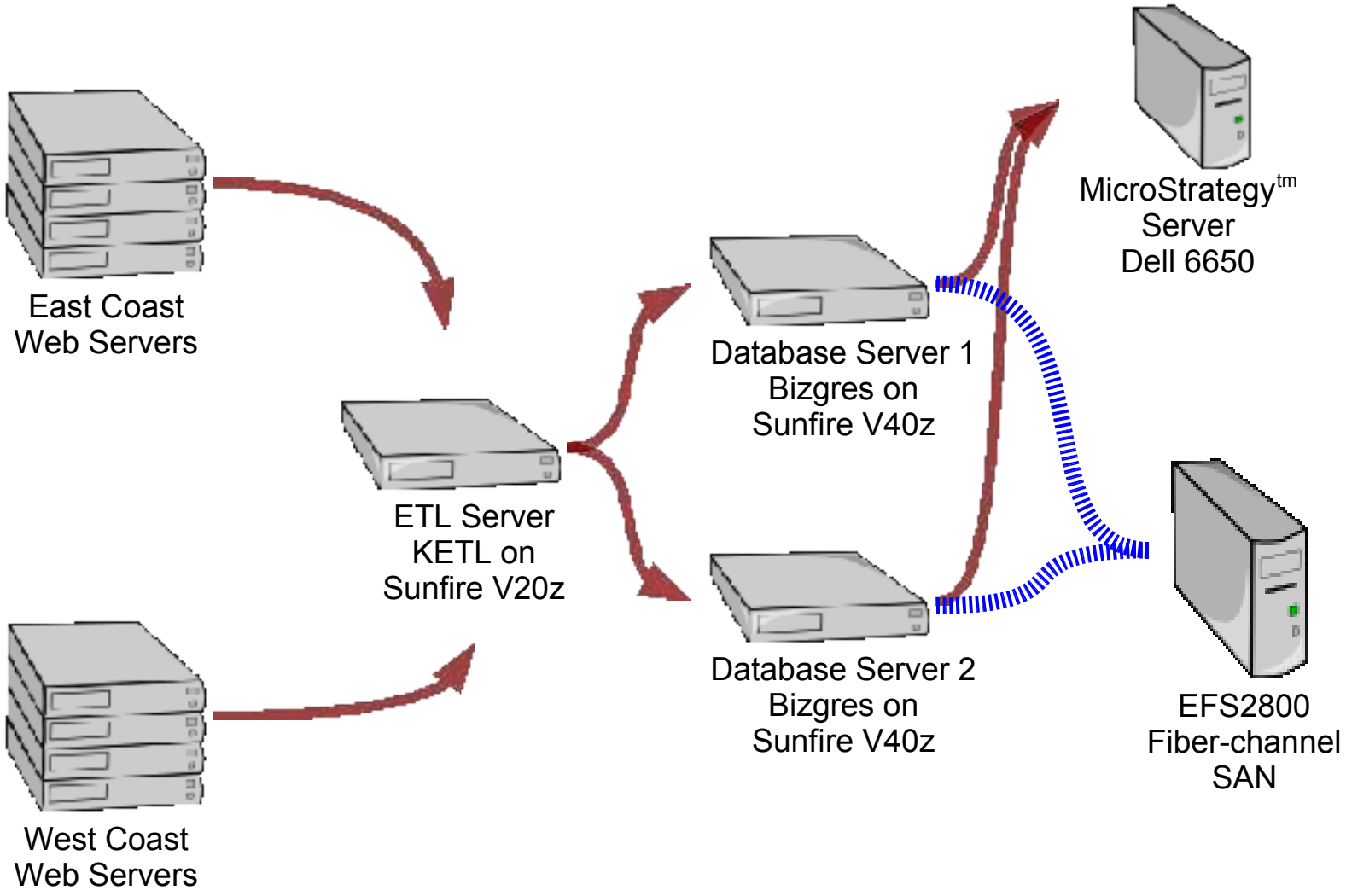
- effective_cache_size: to 1,200,000 (or 9GB)
- random_page_cost: to 2
- cpu_*_cost, lowered

Settings to speed up data loads

- wal_buffers 128
- checkpoint_segments 256 (2 GB)
- checkpoint_timeout 3600 (1 hour)

Problem 2: Data Size

- Estimating Data Size
  - Originally planned: 170GB
  - Based on ERWin data calculator: 350GB
  - Based on 1 week's data: 460GB
    - 750MB new data a day
    - Indexes and aggregates
  - Actual Size of 1 year's data: 520GB
  - Raw data to live data multiple: 1.8
- Storing Data
  - v40z's could only hold 200GB, including xlog
  - Moved all data to EFS2800 Storage Device with two 600GB logical partitions.
  - Re-partitioned v40z's:
    - RAID1-0 = Operating System, Temp space
    - RAID1-1 = Transaction Log (xlog)


East Coast
Web Servers
West Coast
Web Servers
ETL Server
KETL on
Sunfire V20z
Database Server 1
Bizgres on
Sunfire V40z
Database Server 2
Bizgres on
Sunfire V40z
MicroStrategy™
Server
Dell 6650
EFS2800
Fiber-channel
SAN

Problem 3: Aggregate Reports

Microstrategy Produced Numerous Aggregate Reports in this form:

```
select    a12.DAY_OF_WEEK_NBR AS DAY_OF_WEEK_NBR,
     max(TO_CHAR(a12.DATE_DESC ,'Day')) AS CustCol_6,
     a11.DATE_ID AS DATE_ID,
     max(a12.DATE_DESC) AS DATE_DESC,
     a11.FI_ID AS FI_ID,
     max(a13.FI_NAME) AS FI_NAME,
     a12.WEEK_YEAR_ID AS WEEK_YEAR_ID,
     max(a14.SHORT_WEEK_DESC) AS SHORT_WEEK_DESC,
     sum (session_count) AS WJXBFS1,
     sum ( a11_count ) AS WJXBFS2
from ( SELECT DATE_ID, FI_ID, count(distinct SESSION_ID) as session_count, COUNT(*)
as a11_count
          FROM edata.WEB_SITE_ACTIVITY_FA
          WHERE DATE_ID in (2291, 2292, 2293, 2294, 2295)
          GROUP BY DATE_ID, FI_ID )
          a11
     join edata.DATE_LU  a12
       on       (a11.DATE_ID = a12.DATE_ID)
     join edata.DIM_FI   a13
       on       (a11.FI_ID = a13.FI_ID)
     join edata.WEEK_LU  a14
       on       (a12.WEEK_YEAR_ID = a14.WEEK_YEAR_ID)
group by  a12.DAY_OF_WEEK_NBR,
     a11.DATE_ID,
     a11.FI_ID,
     a12.WEEK_YEAR_ID;
```

Problem 3: Aggregate Reports

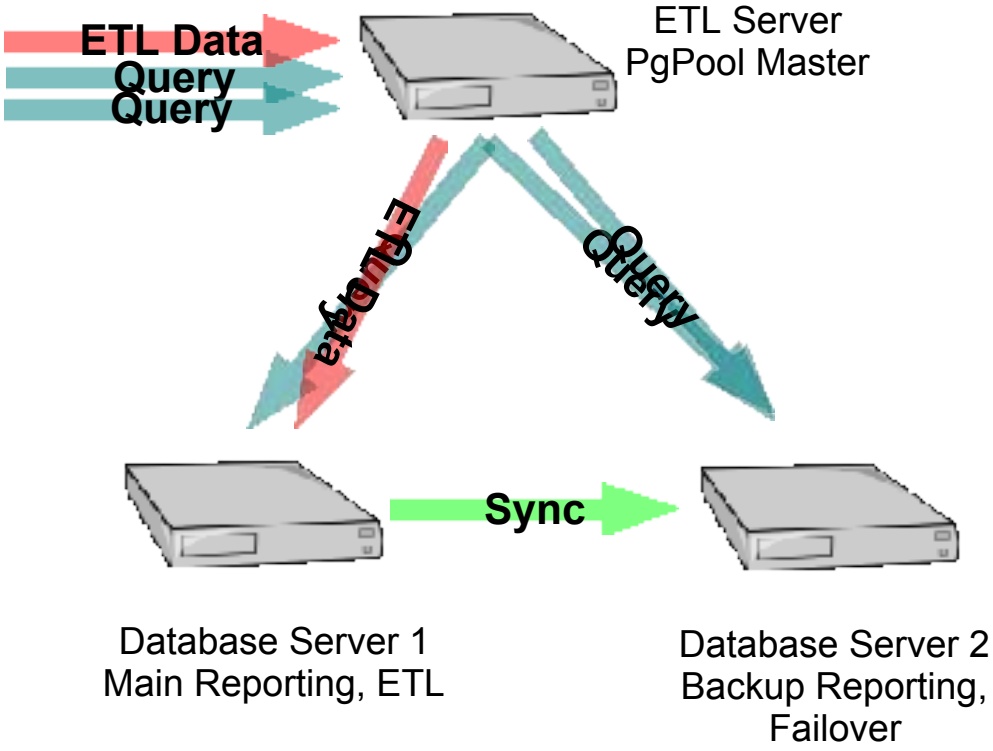
- Solution: Produce “Aggregate Tables”
  - Populated at ETL time
  - Populated cumulatively, no re-generation
  - No VACUUM either!
  - Each aggregate table uses 2 partitions, {agg_table} and {agg_table}_current_week
- Example:

```
        Table "edata.wk_ref_agg"
     Column      |   Type   | Modifiers
-----------------+----------+-----------
 week_year_id    | integer  | not null
 referrer_desc   | text     |
 fi_id           | smallint | not null
 user_cnt_summ   | integer  |
 visit_cnt_summ  | integer  |
 pg_vw_cnt_summ  | integer  |
```

Problem 4: High Availability

- Availability Issues
  - Hardware failures
  - ETL failures
  - Nighly ETL takes 5+ hours
  - Long-running ad hoc reports
- PgPool “Cluster”
  - ETL Server set up with PgPool
  - Switch to Server 2 during nightly ETL
  - Rsync + PITR servers at end of ETL
  - Automated failover if one server fails
  - Server 2 can handle reports on older data
  - Only 20 minutes of downtime per day

Problem 4: High Availability



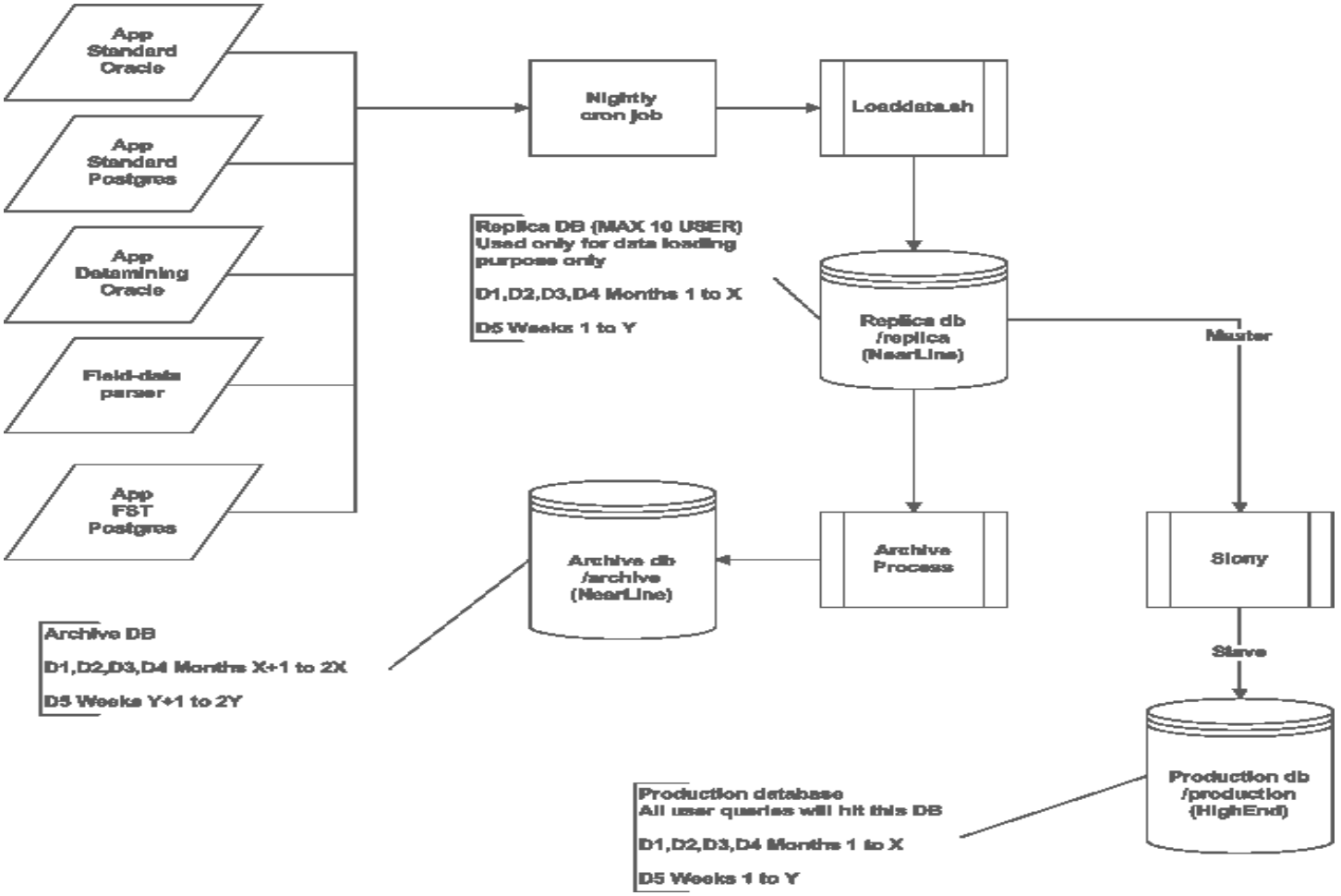
Agenda

- Case study: weblog analysis data
- Case study: equipment performance data
  - Overview
  - Hardware
  - Database schema
  - Data processing
  - Performance
  - Lessons learned
- Upcoming large database features

Agenda

- Case study: weblog analysis data
- Case study: equipment performance data
  - Overview
  - Hardware
  - Database schema
  - Data processing
  - Performance
  - Lessons learned
- Upcoming large database features

Overview


App Standard Oracle
App Standard Postgres
App Datamining Oracle
Field-data parser
App FST Postgres
Nightly cron job
Loaddata.sh
Replica DB (MAX 10 USER) Used only for data loading purpose only
D1,D2,D3,D4 Months 1 to X
D5 Weeks 1 to Y
Replica db /replica (NearLine)
Master
Archive db /archive (NearLine)
Archive Process
Slony
Slave
Archive DB
D1,D2,D3,D4 Months X+1 to 2X
D5 Weeks Y+1 to 2Y
Production db /production (HighEnd)
Production database
All user queries will hit this DB
D1,D2,D3,D4 Months 1 to X
D5 Weeks 1 to Y

Agenda

- Case study: weblog analysis data
- Case study: equipment performance data
  - Overview
  - Hardware
  - Database schema
  - Data processing
  - Performance
  - Lessons learned
- Upcoming large database features

Hardware

- IBM x445
  - 8 Xeon CPUs
  - 8 GB RAM
- Network Appliance
  - 5.5 TB SCSI high performance storage
  - 11 TB IDE “near-line” storage
  - NFS mounted
    - proto=tcp,suid,rw,vers=3,proto=tcp,timeo=600,retrans=2,hard,fg,rsize=8192,wsize=8192
    - MTU=9000

Agenda

- Case study: weblog analysis data
- Case study: equipment performance data
  - Overview
  - Hardware
  - Database schema
  - Data processing
  - Performance
  - Lessons learned
- Upcoming large database features

Database Schema

- Inherited tables design
- Each entity has master parent table
- D1, D2, D3, D4 have monthly partitions
- D5 has weekly partitions
- Partitions are created in advance of loading

Agenda

- Case study: weblog analysis data
- Case study: equipment performance data
  - Overview
  - Hardware
  - Database schema
  - Data processing
  - Performance
  - Lessons learned
- Upcoming large database features

Data Processing

- File extract created at field installation
- File pushed by ftp to central site
- Daily batch process
  - Stamp filename with unique identifier
  - Build work list of files ready to be processed
  - Process files 5 at a time in parallel

Data Processing - per file process

- If file is Oracle export file, process with oraexp2pgexp
- Perform ETL with Perl script
  - Load master data and header record data into temp tables
  - Check for duplicate records based on header data
  - Create new Postgres import files (COPY format)
    - perform crosstab transformation
    - skip any duplicate records
    - original version of this script copied to master table and triggers dispersed rows to appropriate partition table
    - new version of script batches COPY commands directly into partition tables

Agenda

- Case study: weblog analysis data
- Case study: equipment performance data
  - Overview
  - Hardware
  - Database schema
  - Data processing
  - Performance
  - Lessons learned
- Upcoming large database features

Performance

```
dw=> select count(*) from tbl_d1_2005_may;
   count
-----------
 108879137
(1 row)

dw=> explain analyze select diag_value from tbl_d1_2005_may
                     where data_set_date > '10-May-2005' and
                     data_set_date < '11-May-2005' and
                     serial_number = 6132 and
                     diag_num = 151;
                                  QUERY PLAN
-------------------------------------------------------------------
 Index Scan using pk_tbl_d1_2005_may on tbl_d1_2005_may
 (cost=0.00..4.83 rows=1 width=32)
 (actual time=1.120..1.166 rows=1 loops=1)
 Total runtime: 1.271 ms
(4 rows)
```

Performance

```
dw=> select count(*) from eq_d1_data;
   count
------------
 1208530200
(1 row)

dw=> explain analyze select diag_value from eq_d1_data
                     where data_set_date > '10-May-2005' and
                           data_set_date < '11-May-2005' and
                           serial_number = 6132
                           and diag_num = 151;
                                    QUERY PLAN
------------------------------------------------------------------------
[...]
 Total runtime: 43.696 ms
(479 rows)
```

Performance

```
dw=> select count(*) from eq_d5_data;
   count
------------
 1359230913
(1 row)

dw=> explain analyze select eq3 from eq_d5_data
                     where data_set_date > '10-May-2005' and
                     data_set_date < '11-May-2005' and
                     serial_number = 6302;
                                  QUERY PLAN
-----------------------------------------------------------------
[...]
 Total runtime: 9.984 ms
(305 rows)
```

Agenda

- Case study: weblog analysis data
- Case study: equipment performance data
  - Overview
  - Hardware
  - Database schema
  - Data processing
  - Performance
  - Lessons learned
- Upcoming large database features

Lessons Learned

- Duplicate record avoidance is the single largest cost of the ETL process
- Trigger based dispersion of records to partitions was also costly and unnecessary
- A well designed data warehouse running on PostgreSQL can outperform a poorly designed one running on the leading commercial database by several orders of magnitude

Agenda

- Case study: weblog analysis data
- Case study: equipment performance data
- Upcoming large database features
  - Bitmapscan
  - Constraint elimination
  - Improved I/O

Upcoming Features

- BitmapScan
  - Converts B-tree indexes to Bitmaps in RAM
  - Allows combining several indexes in one operation
  - By Tom Lane

```
select * from bt1 where g=2 and e=20;

                              QUERY PLAN
          ---------------------------------------------
Bitmap Heap Scan on bt1
    (cost=2041.47..19807.47 rows=8451 width=159)
   Recheck Cond: ((e = 20) AND (g = 2))
   ->  BitmapAnd  (cost=2041.47..2041.47 rows=8451 width=0)
         ->  Bitmap Index Scan on bt1_btree_e
              (cost=0.00..145.91 rows=25404 width=0)
               Index Cond: (e = 20)
         ->  Bitmap Index Scan on bt1_btree_g
              (cost=0.00..1895.31 rows=332661 width=0)
               Index Cond: (g = 2)
```

Upcoming Features

- Constraint Elimination
  - Allows use of table partitioning in complex queries
  - Partially implements range partitioning on tables
  - By Simon Riggs

```
template1=# explain select * from sales
            where DateKey = date '2005-02-23';

                            QUERY PLAN
 ------------------------------------------------------------
  Result (cost=0.00..60.75 rows=16 width=16)
    -> Append (cost=0.00..60.75 rows=16 width=16)
          -> Seq Scan on sales (cost=0.00..30.38 rows=8 width=16)
                Filter: (datekey = '2005-02-23'::date)
          -> Seq Scan on sales_feb sales (cost=0.00..30.38
                                              rows=8 width=16)
                Filter: (datekey = '2005-02-23'::date)
```

Upcoming Features

- I/O Improvements for data loading, ETL
  - COPY parsing improvements
  - Scratch tables without logging
  - JDBC COPY
  - Bizgres Loader

Contact Information

- This presentation available at: www.powerpostgresql.com
- PostgreSQL: www.postgresql.org
- Joe: mail@joeconway.com
- Josh: josh@postgresql.org
- Bizgres: www.bizgres.org