Nutzerforum - PostgreSQL

Der Datenbankdienst des URZ bekommt Nachwuchs

Andreas Heik, Daniel Schreiber

TU-Chemnitz, Universitätsrechenzentrum

22. April 2015

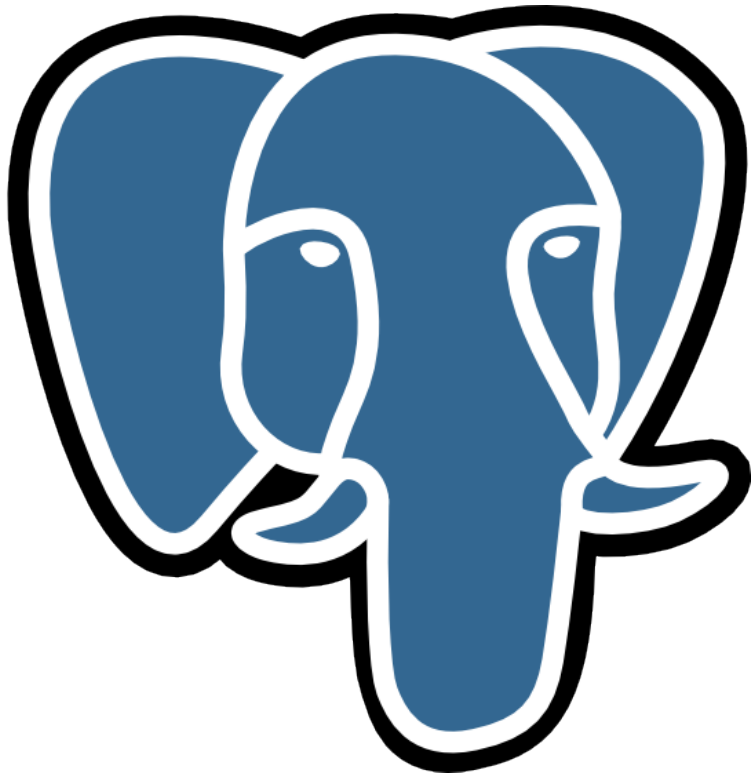
MySQL-Datenbankdienst

- Datenbankdienst um 1999/2000 etabliert
- 555 aktive Datenbanken belegen $\sum$ 35 GB [1]
- Datenbanken als *„shared service“*
  - auf einem virtuellen Server
  - mit 8 CPUs und 12 GB RAM
  - MySQL 5.1.x unter Scientific Linux 6
- zentrale Systempflege (LADM)
  (Betriebssystem und Datenbanksoftware)
- tägliche Datensicherung `(mysqldump)`

[1] Stand: 04/2015

Motivation für PostgreSQL

- Anfragen aus Nutzerkreis
- Eigenbedarf im URZ und ZUV
- ausgereifte, stabile Datenbanksoftware
- OpenSource Lizenz
  $\rightarrow$ **uneingeschränkt nutzbar**
- 06/2014 HiWi-Projekt ausgeschrieben



Entwicklungsziele

- Datenbankservice auf Basis von PostgreSQL
  (ähnlich dem MySQL-Datenbankservice)
- einfacher Zugang, einfache Benutzung
  (automatisiertes Anlegen von Datenbanken und Datenbankbenutzern)
- Integration in IdM-Portal:
  - Beauftragung und Modifikation
  - Verwaltung (Ressourcenverantwortlicher, Laufzeit, . . .)
  - Freigabe der Ressourcen
- Betrieb eines zentralen Servers im URZ

technische Basis

- PostgreSQL 9.4
  - Softwareupdates von `http://www.postgresql.org/`
- Datenbanken als *„shared service“*
  - auf einem virtuellen Server
  - mit 2 CPUs und 8 GB RAM
  - unter Scientific Linux 6 (LADM)
- Service Monitoring auf Basis von *xymon*
- tägliche Datensicherung
- Replikation zu Slave-System

Datenbank beauftragen

- zentraler Dienst des URZ mit Fokus auf Forschung und Lehre
  (keine kommerzielle Nutzung)
- Nutzerkreis: Studenten und Mitarbeiter
- Beauftragung im IdM-Portal:
  `https://idm.hrz.tu-chemnitz.de/user/`
  - minimale Parameterabfrage
    (Name, Beschreibung, Ablaufdatum)
  - schreibberechtigter Datenbanknutzer aus DB-Name gebildet
  - *optional:* leseberechtigter Datenbanknutzer
  - Datenbankpassworte werden vom IdM nach aktuellen Sicherheitsrichtlinien erzeugt und einmalig angezeigt

Nutzungshinweise

- Datenbankressourcen sind befristet
  (Verlängerung der Laufzeit im IdM-Portal)
  - Sperrung der Ressource am Laufzeitende
  - Löschen der Ressource 60 Tage nach Sperrung
- *„shared service“* - Datenbanken teilen sich einen DB-Server
  - Datenvolumen und Abfragefrequenz berücksichten
- keine IP-basierte Beschränkung im Campusnetz
  (bisher häufigste Fehlerursache)
- **Tipp:** für nichtpersönliche Datenbankprojekte zusätzlichen Ressourcenverantwortlichen oder IdM-Gruppe festlegen

Verbesserungen von PostgreSQL

- viel mehr Standard SQL als MySQL
- Statistik basierter Optimierer
- Window functions
- Rekursive Abfragen
- Nutzerdefinierte Funktionen
- Nutzerdefinierte Aggregate
- Trigger feuern immer (standardkonform)
- Volle Freiheit bei LIMIT, OFFSET, Subselects, ORDER BY
- Partielle Indizes

Datensicherung

- tägliche Datensicherung aller Datenbanken ab 0:15 Uhr
  (mittels `pg_dump`)
- Aufbewahrung der Dumps nach Vorgaben des *RSYNC Backup-Dienstes*
  (6 Monate, davon 12 wöchentliche und 28 tägliche Dumps)
- Restorefall:
  - Auslieferung von Datenbank-Dumps an Ressourcenverantwortliche
    Restore mittels *pg_restore*
  - Einspielen eines Datenbank-Dumps auf Wunsch durch URZ
- Havariefall:
  - Aktivierung des Slave-Systems, Umschalten des DNS-Alias
    (kein automatischer Prozess)

Clientenunterstützung

Verbindungsinformationen werden im IdM-Portal angezeigt

- Kommandozeile: `psql`
- grafisches Werkzeug: `pgadmin3`
- Webanwendungen: `php, pg_connect()`
- Treiber für verschiedene Programmiersprachen
  (PyGreSQL, python-psycopg2, ...)
- SDBC-Treiber (LibreOffice Integration)
- ODBC, JDBC-Treiber

Demo

- Datenbank beauftragen
- Daten laden
- pgadmin3
- PHP
- SQL

psql

- Interaktiver SQL Prompt
- \? psql Hilfe
- \h SQL Hilfe
- \e letzen Befehl bearbeiten
- \d Tabellen anzeigen
- \i Code aus Datei laden und ausführen
- CSV Daten laden: \copy t_oil from t_oil.csv with delimiter ','

PHP PSQL

```
// Verbindungsaufbau und Auswahl der Datenbank
$dbconn = pg_connect("host=pgsql.hrz dbname=mydb user=mydb_rw password=****")
   or die('Verbindungsaufbau fehlgeschlagen: ' . pg_last_error());

// SQL-Abfrage
$query = 'SELECT * FROM authors';
$result = pg_query($query)
   or die('Abfrage fehlgeschlagen: ' . pg_last_error());

// Ergebnisse HTML-formatiert ausgeben
echo "<table>\n";
while ($line = pg_fetch_array($result, null, PGSQL_ASSOC)) {
   echo "\t<tr>\n";
   foreach ($line as $col_value) {
      echo "\t\t<td>$col_value</td>\n";
   }
   echo "\t</tr>\n";
}
echo "</table>\n";

// Speicher freigeben
pg_free_result($result);

// Verbindung schliessen
pg_close($dbconn);
```

Fallstricke für MySQL-Nutzer

- Quoting: "$\rightarrow$ Bezeichner, ' $\rightarrow$ Text
- GROUP BY: Alle Spalten müssen angegeben werden, wenn kein Aggregat
- kein AUTO_INCREMENT $\rightarrow$ Datentyp SERIAL verwenden
- CHECK Klauseln werden angewendet
- mysql -u $\rightarrow$ psql -U
- Sortierung NULL-Werte per Default anders. PostgreSQL: NULLS LAST
- LIKE ist case sensitive. Alternative ILIKE
- || in MySQL OR, Standard SQL Textverknüpfung

SQL Basics

```
SELECT * FROM t_oil;
```

```
   country      | year |  production
----------------+------+-------------
 Saudi Arabia   | 1980 |      3623400
 Saudi Arabia   | 1981 |      3582475
 Saudi Arabia   | 1982 |      2366295
 Saudi Arabia   | 1983 |      1856390
...
 Saudi Arabia   | 2013 |      3538000
 USA            | 1980 |      3146502
 USA            | 1981 |      3128780
 USA            | 1982 |      3156885
...
```

SQL Basics

```
select sum(production), country from t_oil group by
    country;

   sum     |    country
-----------+---------------
 84529125 | USA
 96284029 | Saudi Arabia
(2 Zeilen)
```

SQL Basics

```
select year/10*10, country, sum(production)
from t_oil
group by country, year/10
order by year/10, country;
```

```
 ?column? |    country     |    sum
----------+----------------+----------
     1980 | Saudi Arabia   | 21415949
     1980 | USA            | 31101036
     1990 | Saudi Arabia   | 29292522
     1990 | USA            | 24490633
     2000 | Saudi Arabia   | 31738128
     2000 | USA            | 19782289
     2010 | Saudi Arabia   | 13837430
     2010 | USA            |  9155167
```

Group Filter

```
select year,
    sum(production) filter (WHERE country='Saudi Arabia'
        ) as "SA",
    sum(production) filter (WHERE country='USA') as USA
from t_oil
group by year
order by 1;
```

```
 year |    SA     |   usa
------+-----------+---------
 1980 | 3623400 | 3146502
 1981 | 3582475 | 3128780
 1982 | 2366295 | 3156885
 1983 | 1856390 | 3171120
...
```

Daten generieren

```
select * from generate_series('2014-01-01'::date, '
    2014-01-31'::date, '1 days'::interval);

     generate_series
------------------------
 2014-01-01 00:00:00+01
 2014-01-02 00:00:00+01
 2014-01-03 00:00:00+01
 2014-01-04 00:00:00+01
...
 2014-01-31 00:00:00+01
(31 Zeilen)
```

Daten generieren

```
select g.g::date from generate_series('2014-01-01'::date
    , '2014-01-31'::date, '2 weeks'::interval) as g;
```

```
     g
------------
 2014-01-01
 2014-01-15
 2014-01-29
```

Daten generieren (rekursiv)

```
WITH RECURSIVE
x(i)
AS (
    VALUES(0)
UNION ALL
    SELECT i + 1 FROM x WHERE i < 10
)
SELECT * from x;
```

```
 i
----
  0
  1
  2
...
  8
  9
 10
(11 Zeilen)
```

Daten generieren (rekursiv)

```
WITH RECURSIVE t(a,b) AS (
        VALUES(0,1)
    UNION ALL
        SELECT greatest(a,b), a + b AS a FROM t
        WHERE b < 10
)
select a from t;
```

```
 a
---
 0
 1
 1
 2
 3
 5
 8
(7 Zeilen)
```

Nutzerdefinierte Funktionen

```
CREATE OR REPLACE FUNCTION fib(f integer)
RETURNS SETOF integer
LANGUAGE SQL
AS $$
WITH RECURSIVE t(a,b) AS (
        VALUES(0,1)
    UNION ALL
        SELECT greatest(a,b), a + b AS a FROM t
        WHERE b < $1
   )
SELECT a FROM t;
$$;

select * from fib(10);
 fib
-----
   0
   1
...
```

Window Functions

```
select year, country, production,
        (first_value(t_oil) over w).production,
        (first_value(t_oil) over w).year as max_year
from t_oil
window w as (partition by year/10, country order by
   production desc)
order by country, year;
```

```
 year |   country    | production | production | max_year
------+--------------+------------+------------+----------
 1980 | Saudi Arabia |    3623400 |    3623400 |     1980
 1981 | Saudi Arabia |    3582475 |    3623400 |     1980
...
 1997 | Saudi Arabia |    3052142 |    3061950 |     1998
 1998 | Saudi Arabia |    3061950 |    3061950 |     1998
 1999 | Saudi Arabia |    2859187 |    3061950 |     1998
...
 1985 | USA          |    3274415 |    3274415 |     1985
 1986 | USA          |    3168200 |    3274415 |     1985
...
```

Hilfe

- https://www.tu-chemnitz.de/urz/storage/db/
- http://www.postgresql.org/docs/9.4/interactive/index.html
- https://wiki.postgresql.org/wiki/Main_Page
- http://php.net/manual/de/book.pgsql.php
- support@hrz.tu-chemnitz.de

Nutzerforum - PostgreSQL

VIELEN DANK FÜR
IHRE AUFMERKSAMKEIT!