zalando

zalando

# Elefanten sind in Mode

PostgreSQL bei Zalando

zalando

# Über mich

**Jan Mußler** Dipl.-Inform.

Database Engineer @ Zalando

E-Mail: jan.mussler@zalando.de

Technology Blog: tech.zalando.com

# Zalando in Zahlen

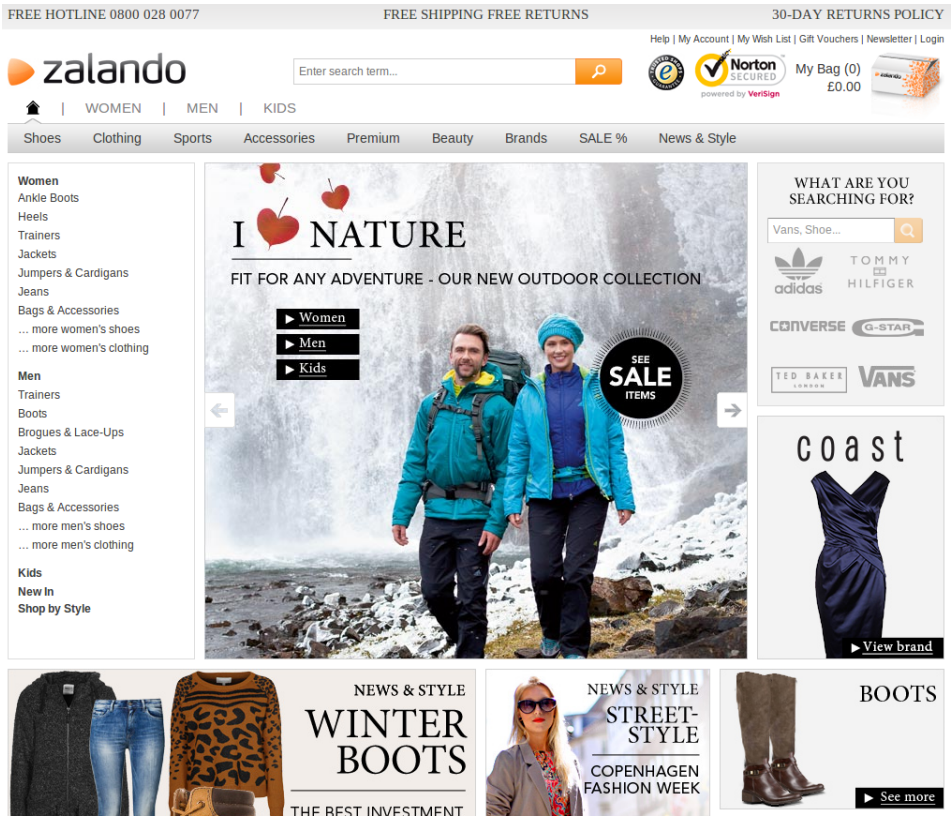
1 Mrd. € Nettoumsatz

14 Länder

15+ Mio Kunden

3 Logistikstandorte

Mehr als 150.000 Artikel



zalando

# Herausforderungen

- Schnelle Entwicklung und Flexibilität
- Hohe Verfügbarkeit
- Hohe Performance
- Skalierbarkeit

zalando

# ZEOS Platform in kurz

Open Source Software Stack

Java dominiert, etwas Python für internes Tooling

PostgreSQL Datenbanken

mehr als 400 Technology Kollegen

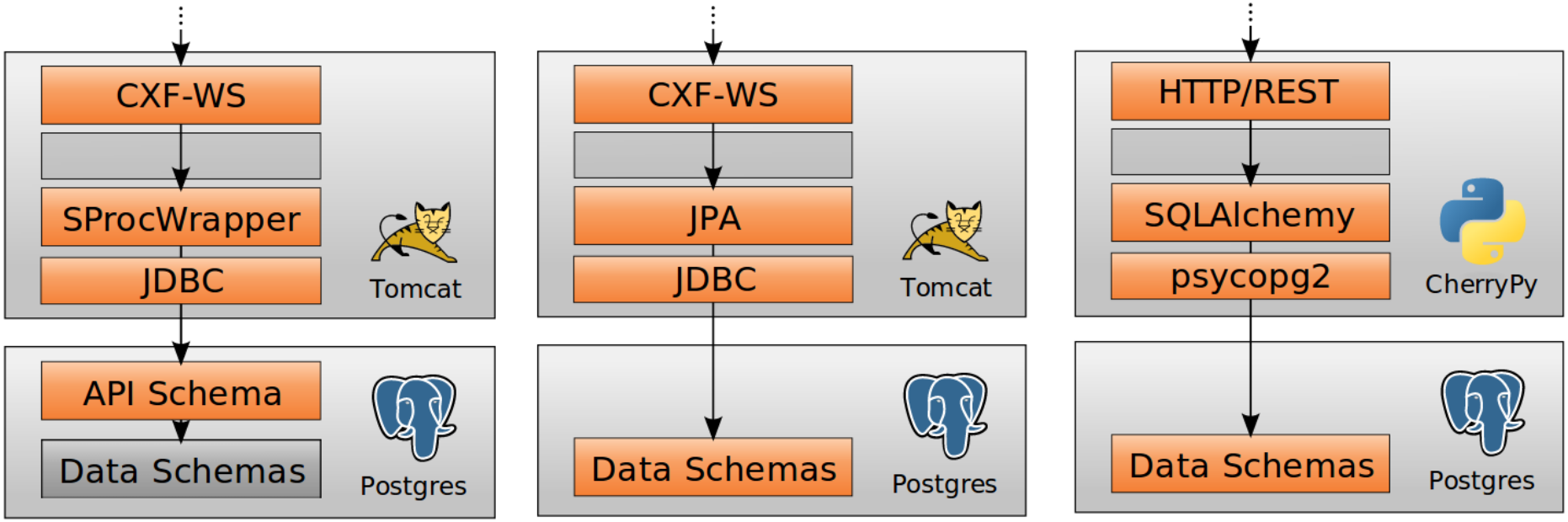
zalando

# ZEOS Platform in Zahlen

- 800+ Live Tomcat Instanzen
- 50+ unterschiedliche Datenbanken
- 90+ PostgreSQL Master Datenbanken
- 5 TB in PostgreSQL und wachsend

zalando

# Datenbankzugriff ...



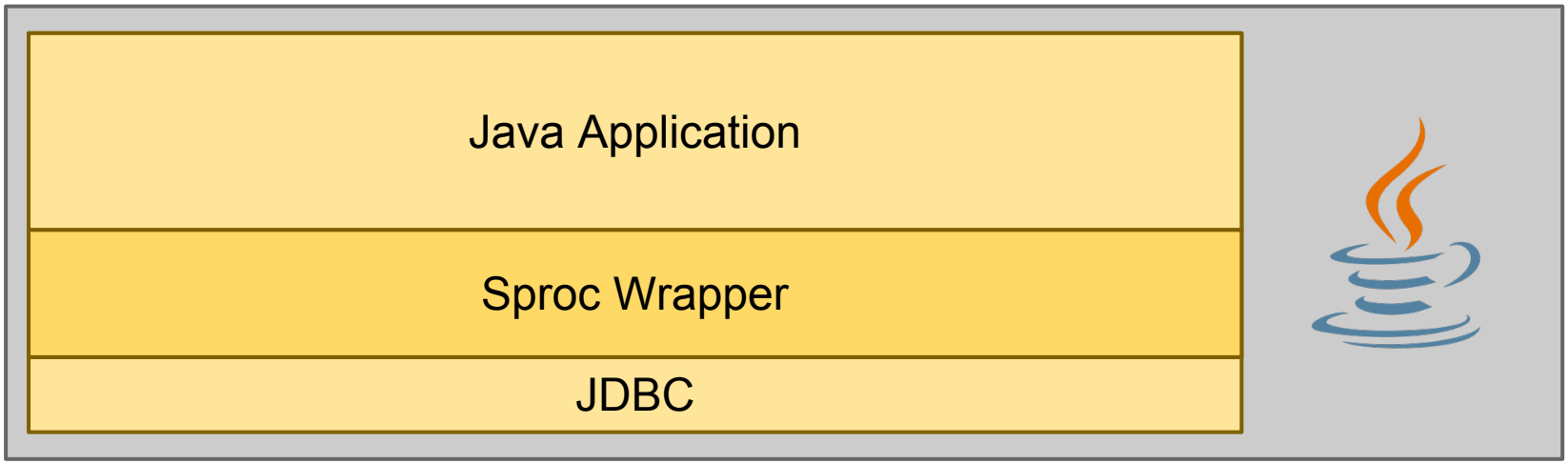
zalando

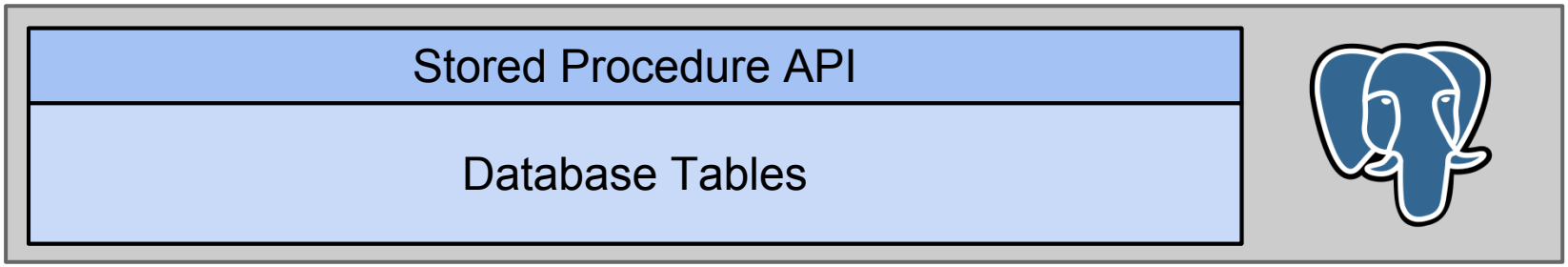
# Stored Procedures

- Performance Verbesserung
- Validierung nah an Daten
- Einfacher Transaktionskontext
- Verwendbar in mehreren Sprachen
- Einfach zu modifizieren in “Production”
- Rückgabe von Aggregaten von Entitäten

zalando

# Java Sproc Wrapper




zalando

zalando

# Stored Procedure Wrapper

JAVA

```java
@SProcService
public interface CustomerSProcService {
    @SProcCall
    int registerCustomer(@SProcParam String email, @SProcParam Gender gender);
}
```

SQL

```sql
CREATE FUNCTION register_customer(p_email text, p_gender z_data.gender)
RETURNS int AS
$$
    INSERT INTO z_data.customer (c_email, c_gender)
         VALUES (p_email, p_gender)
      RETURNING c_id
$$
LANGUAGE 'sql' SECURITY DEFINER;
```

zalando

# Stored Procedures Wrapper

JAVA

```java
@SProcService
public interface CustomerSProcService {
    @SProcCall
    int registerCustomer(@SProcParam String email, @SProcParam Gender gender);
}
```

SQL

```sql
CREATE FUNCTION register_customer(p_email text, p_gender z_data.gender)
RETURNS int AS
$$
    INSERT INTO z_data.customer (c_email, c_gender)
         VALUES (p_email, p_gender)
      RETURNING c_id
$$
LANGUAGE 'sql' SECURITY DEFINER;
```

zalando

# Schema Aufbau

## Tabellen, Views, Enums und Typen

Data Schema

.customer

.order

.order_address

## Funktionen und Ergebnistypen

API Schema

.register_customer()

.find_orders()

.create_order()

zalando

# API Schema Versionierung

Data Schema

.customer

.order

.order_address

api_r13_05

.register_customer(text,...)

.find_orders(text,...)

.create_order(int, ...)

api_r13_06

.register_customer(int,...)

.find_orders(json)

.create_order(hstore)

zalando

# API Schema Versionierung

- Testing auf der gesamten API Version
- Keine Migration von Schema notwendig
- Deployment ist automatisiert

zalando

# Segmentiertes Deployment

[ Instanz1 , … ]

[ Instanz2 , … ]

Application v13_00_05

Application v13_00_06

API_13_00_05

API_13_00_06

PostgreSQL DB

**PostgreSQL “search_path = api_13_00_… , public;”**

zalando

# Downtime?

“Keine” Downtime durch Migration oder Deployments

zalando

# Schema Änderungen

1. Exclusive Lock Zeit minimieren
2. Migrationsaufwand minimieren
3. Migration in kleinen Chunks und monitoren

zalando

# Tabellen Erweiterung

```
BEGIN;
    SELECT _v.register_patch('TICKET-1337');

    -- fast catalog lock only
    ALTER TABLE data."order" ADD COLUMN o_tax numeric;

    -- lock already held
    -- fast change of default, no rewrite
    ALTER TABLE data."order" ALTER COLUMN o_tax
                                SET DEFAULT 0.19 * o_value;
COMMIT;

-- Do chunk migration here or use coalesce()
```

zalando

# Daten Schema Migration

DBDIFF SQL

```
BEGIN;
    SELECT _v.register_patch('TICKET-5432.order');

    CREATE TABLE data.order_address (
      oa_id int SERIAL,
      oa_country z_data.country,
      oa_city varchar(64),
      oa_street varchar(128), ...
    );

    ALTER TABLE data."order" ADD o_shipping_address_id int
      REFERENCES data.order_address (oa_id);
COMMIT;
```

zalando

# Daten Schema Migration

DBDIFF SQL

```
BEGIN;
    SELECT _v.register_patch('TICKET-5432.order');

    \i order/database/order/10_tables/10_order_address.sql

    ALTER TABLE z_data."order" ADD o_shipping_address_id int
      REFERENCES z_data.order_address (oa_id);
COMMIT;
```

zalando

# Review Prozess für Änderungen

## Overview of R13_00_44

**Warning!** 11 patch names exists in multiple files!

| Project | Database | Diff | Reviewed | Integration | Release | Patch | LIVE |
|---|---|---|---|---|---|---|---|
| | | **backend - 18/19** | | | | | |
| de.zalando.admin/admin-backend | | | | | | | |
| | admin | | | | | | |
| | | ZEOS-24617.admin | A S | 1 / 1 | 1 / 1 | 1 / 1 | 1 / 1 |
| de.zalando/bm | | | | | | | |
| | bm | | | | | | |
| | | ORDER-453.bm | S A | 0 / 1 | 1 / 1 | 1 / 1 | 1 / 1 |
| de.zalando/config-service | | | | | | | |
| | config | | | | | | |
| | | ZEOS-21566.data | A A S | 1 / 1 | 1 / 1 | 1 / 1 | 1 / 1 |
| | | ZEOS-24840.data | S A | 1 / 1 | 1 / 1 | 1 / 1 | 1 / 1 |
| | | ZEOS-25486.data | A | 0 / 1 | 1 / 1 | 0 / 1 | 1 / 1 |

zalando

# Horizontal skalieren ...

Wieso?

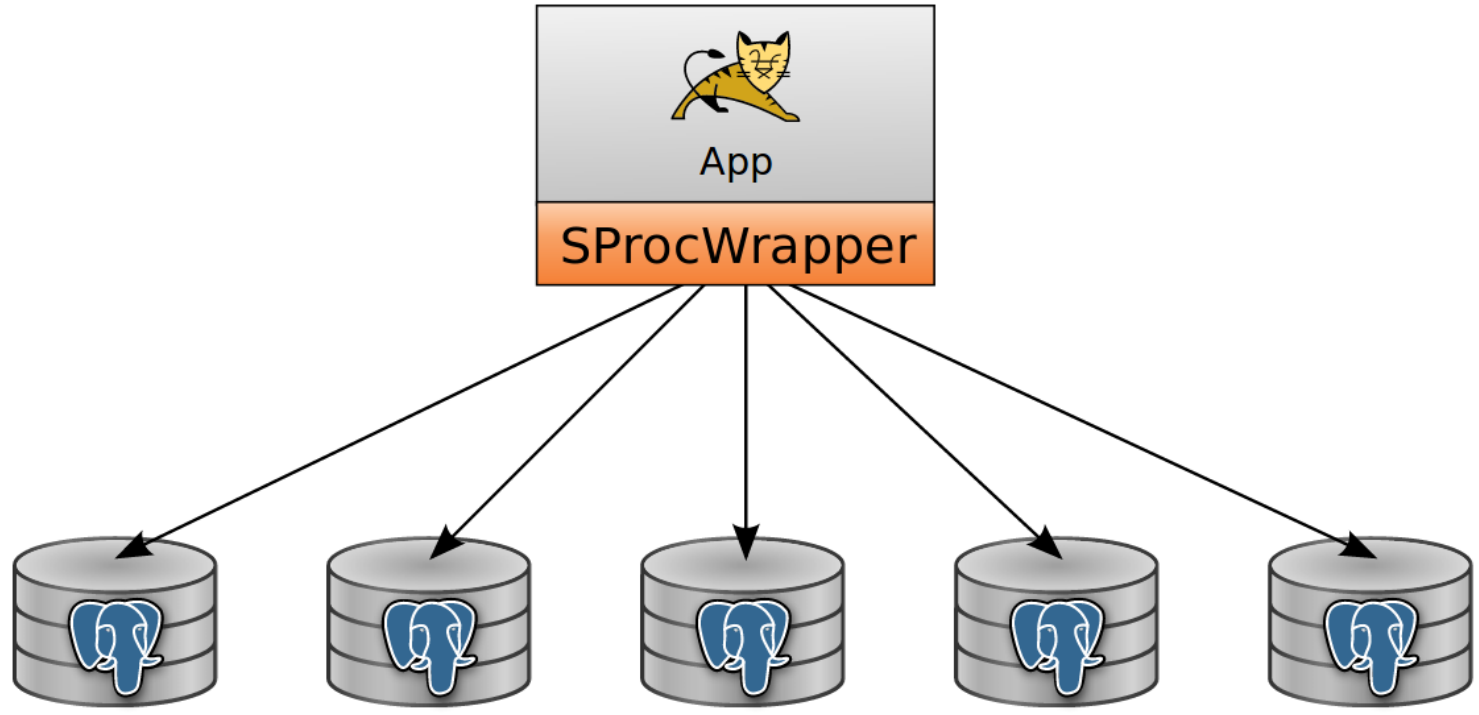
- Performance
- Verfügbarkeit

Probleme?

- Unique IDs
- Cross Shard Queries
- Verwaltung / Wartungsaufwand

zalando

# Horizontal skalieren ...



zalando

# Horizontal skalieren ...

zalando

# Server Konfiguration

## Binary Deployment

```
/server/postgresql/9.0 -> ../9.0.11
/server/postgresql/9.0.10
/server/postgresql/9.0.11

/server/postgresql/9.1 -> ../9.1.4
/server/postgresql/9.1.3
/server/postgresql/9.1.4
```

## Filesystem Setup

System Partition
RAID 1

XLOG Partition
RAID 1

Daten Partition
RAID 1+0

zalando

# Monitoring

zalando

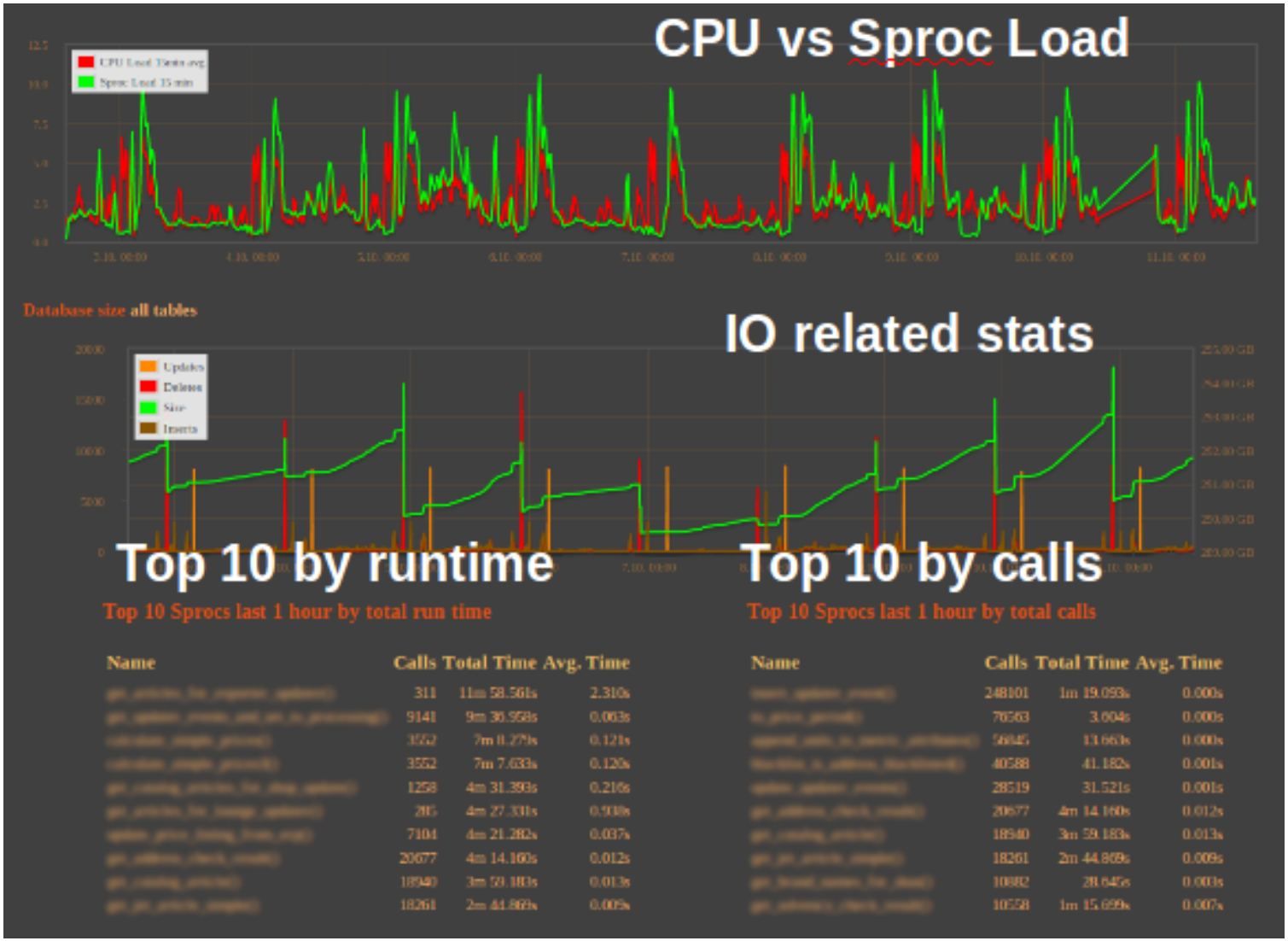
# PG View

```
pgdevx up 21:26:07 2 cores Linux 2.6.32-5-amd64 load average 0.15 0.03 0.01          14:20:10
sys: utime 23.8  stime 28.1  idle 44.3  iowait 0.0   ctxt 360 run 2   block 0
mem: total 1002.7MB free 137.8MB buffers 41.2MB cached 725.9MB dirty 21.0MB limit 1.2GB  as 270
dev 9.0 database connections: 2 total, 1 active
type dev    fill total    left   read  write    await path_size path
data sda1    0.0 7.5GB 271.2MB    0.0   97.8    578.2     9.7GB /data/postgres/pgsql_dev/9.0...
xlog sda1    0.0 7.5GB 271.2MB    0.0   97.8    578.2     8.8GB /data/postgres/pgsql_dev/9.0...
  pid type    s utime stime guest read write   age db       user     query
12063 backend S   0.0   0.0   0.0  0.0   0.0 04:45 postgres postgres idle in transaction
dev 9.1 database connections: 3 total, 2 active
type dev    fill until_full    total    left   read  write    await path_size path
data sda1  106.8 00:03         7.5GB 269.6MB    0.0   96.8    572.4     1.3GB /data/postgres...
xlog sda1    0.0               7.5GB 269.6MB    0.0   96.8    572.4    64.1MB /data/postgres...
  pid type    s utime stime guest read write   age db       user     query
12716 backend S   0.0   0.0   0.0  0.0   0.0 02:23 postgres postgres lock table test;
12173 backend R  45.5  54.1   0.0  0.0 106.9 02:15 postgres postgres INSERT INTO test (id, n...

<s>System processes  <f>Output freeze  <u>Measurement units  <a>Autohide fields  <h>Help
```

zalando

# PGObserver



zalando

# Open Source Projekte

SProcWrapper github.com/zalando/java-sproc-wrapper

PGObserver

github.com/zalando/PGObserver

pg_vie
github.com/zalando/pg_view

zalando

# Danke für’s Zuhören

tech.zalando.com