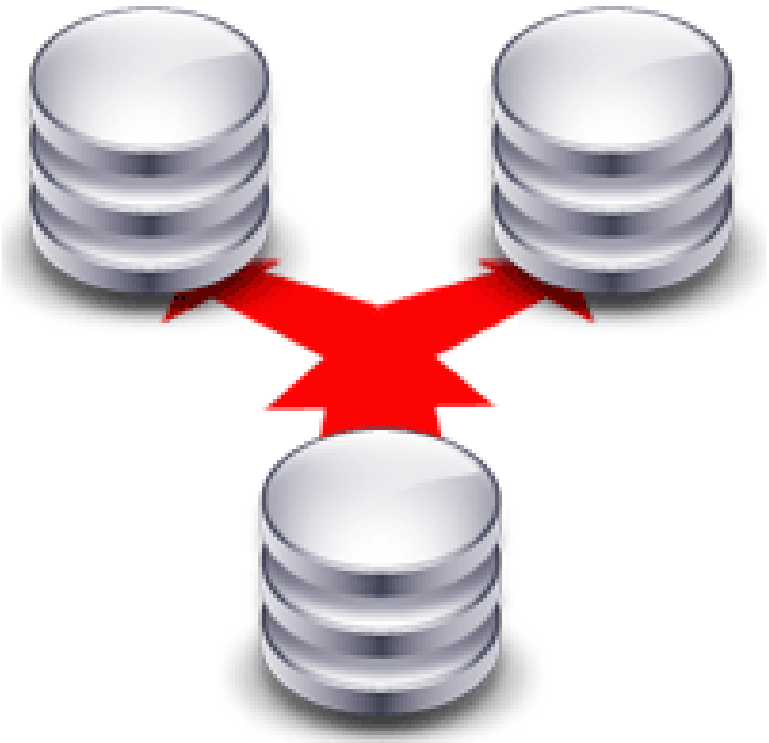
# Replikace databází



*“Three things are certain in life:*
*Taxes, death, and data loss.*
*Guess which has just occurred.”*

*-- Operating system error message*

Ondřej Čečák, ČVUT FEL, Y36DBA

# Replikace databází, Obsah

- představení
- motivace
- “nepřímé” replikace
- replikace v MySQL a v PostgreSQL
- v celém příspěvku tisíc a jeden buzzword :)
- replikace v PostgreSQL pomocí Slony-I

# Replikace

- Replication (in computer science) is the process of sharing information so as to ensure the consistency between redundant resources, such as software or hardware components, to improve reliability, fault-tolerance or accessibility.

http://en.wikipedia.org/wiki/Replication_(computer_science)

# Motivace

*"The value of data stored on a computer, often far exceeds the value of the computer! Hardware can be quickly and easily replaced, but data cannot! Companies often insure there computer hardware against theft and loss, but fail to backup their data, which is far more precious!"*

*-- T.E. Ronneberg*

# Motivace, možné scénáře

- “stand-by backup”/”hot-spare backup”
- “zero-downtime upgrades”
- development servers
- load balancing

# Účel

- pozor, replikace není zálohování, ale spíše disaster-recovery
  - zálohy (uživatel si smaže soubor)
  - archivace (uživatel chce rok starou verzi dat)
  - vysoká dostupnost (HA, high-availability)
- může souviset s rozkládáním zátěže (load balancing)

# Replikace nepřímo

- téměř vždy můžeme replikovat na úrovni operačního systému nebo databázového systému
  - tedy replikace databáze nepřímo
    - replikace souborového systému
    - sdílený prostor s daty
  - pravidelnou zálohou a obnovou databáze

# Replikace databází

- hlavní koncepty:
  - podle typu replikace
    - master/slave
      - vždy pouze jeden master, který provádí změny, které se dál šíří na slaves
      - heartbeat failover
      - riziko “split-brain situation”, kdy si několik systémů myslí, že jsou master – řešením je pak typicky fencing a STONITH (Shoot The Other Node In The Head)

# Replikace databází

- hlavní koncepty:
  - podle typu replikace
    - master/slave
    - multi-master (master-master)
      - technologicky více sexy, změnu může udělat každý node
      - náročnější implementace a provoz (typicky aktivní ochrana před konflikty transakcí)

# Replikace databází

- hlavní koncepty:
  - podle typu replikace
    - master/slave
    - multi-master
  - podle způsobu uložení dat
    - asynchronní
      - změna je potvrzena, jakmile se provede lokálně
      - vysoký výkon za cenu možné ztráty dat

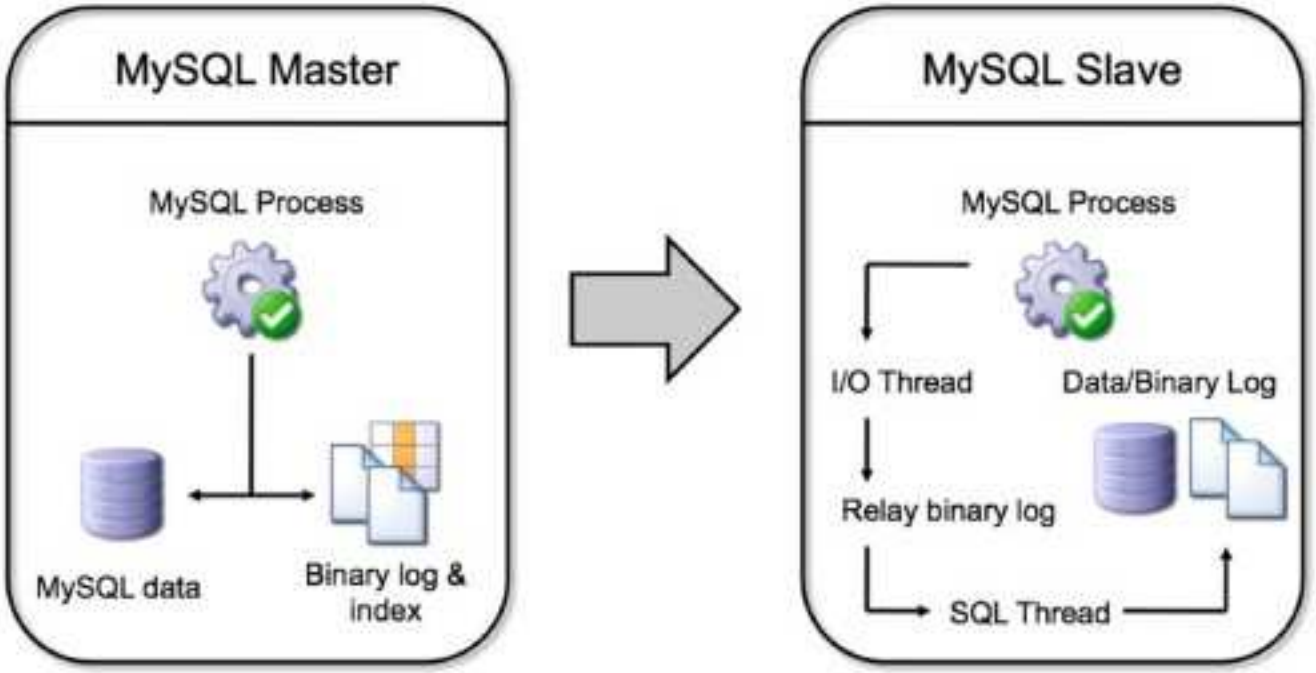
# Replikace databází

- hlavní koncepty:
  - podle způsobu uložení dat
    - asynchronní
    - synchronní
      - garantuje nulovou ztrátu dat
      - nepřekonatelná zpoždění – například rychlost světla (cca 67 μs na 10 km, u lokálního potvrzení zápis bloku za cca 10-20 μs)
      - systém zamrzne už při ztrátě jednoho node a nebo spojení, z definice (!)

# Replikace MySQL

- out-of-the-box, podpora od verze 3.23
- asynchronní
- mnoho replikačních scénářů
  - single master, single slave
  - single master, multiple slaves
  - kaskádová replikace od jednoho mastera
  - master-master

# Replikace MySQL




# Replikace MySQL

- do verze 5.1 byla replikace založená na přenosu SQL dotazů
- v současné době se přenáší binární log row-based (konfigurovatelné, různé chování vzhledem k aplikaci) změn v celém systému, databázi a nebo tabulkách
- single-threaded (bottleneck!)

# Nastavení replikace MySQL

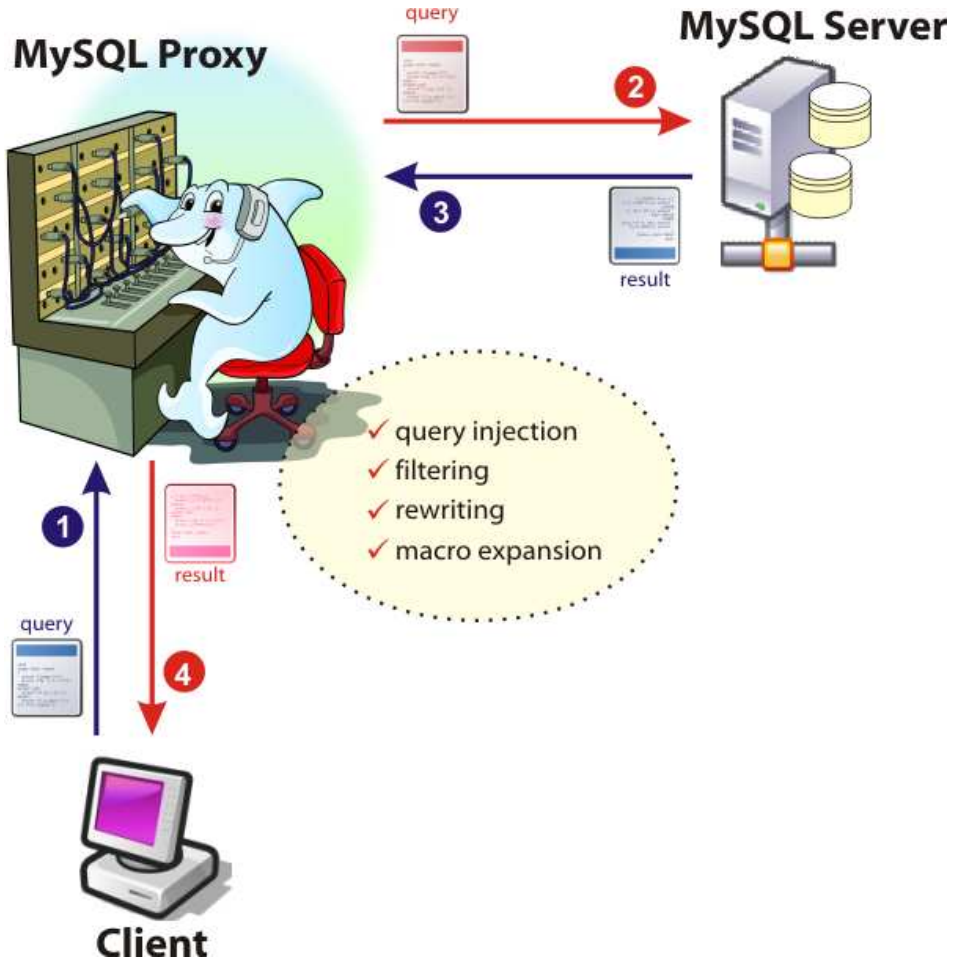
- master-slave replikace
  - MySQL masteru musí být síťově dostupná
  - `GRANT REPLICATION SLAVE ON *...`
  - `binlogdo_db = replicated`
  - `mysqldump` & restore
  - `server-id = 2`
  - `CHANGE MASTER, START SLAVE`

# Co nás čeká v MySQL 5.5

- MySQL 5.5 je momentálně pre-release
- umožňuje semi-synchronní replikace
  - data z masteru jsou přenesena na slave, kde jsou pouze zapsána do logu
  - jakmile master ví, že data jsou v logu slave, provede commit

# Další možnosti replikování MySQL

- MySQL Proxy



# Další možnosti replikování MySQL

- MySQL Proxy
- MySQL NDB Cluster
  - shared-nothing architecture
  - in-memory
  - synchronní replikace (dvoj-fázový commit)
  - no SPOF (Single Point of Failure)
  - některé funkce chybí (např. cizí klíče)

# Další možnosti replikování MySQL

- MySQL Proxy
- MySQL NDB Cluster
- Continuent Tungsten middleware
  - aplikační server nad MySQL
  - synchronní replikace

# Další možnosti replikování MySQL

- MySQL Proxy
- MySQL NDB Cluster
- Continuent Tungsten middleware
- Federated storage engine
  - tabulky na vzdálených strojích

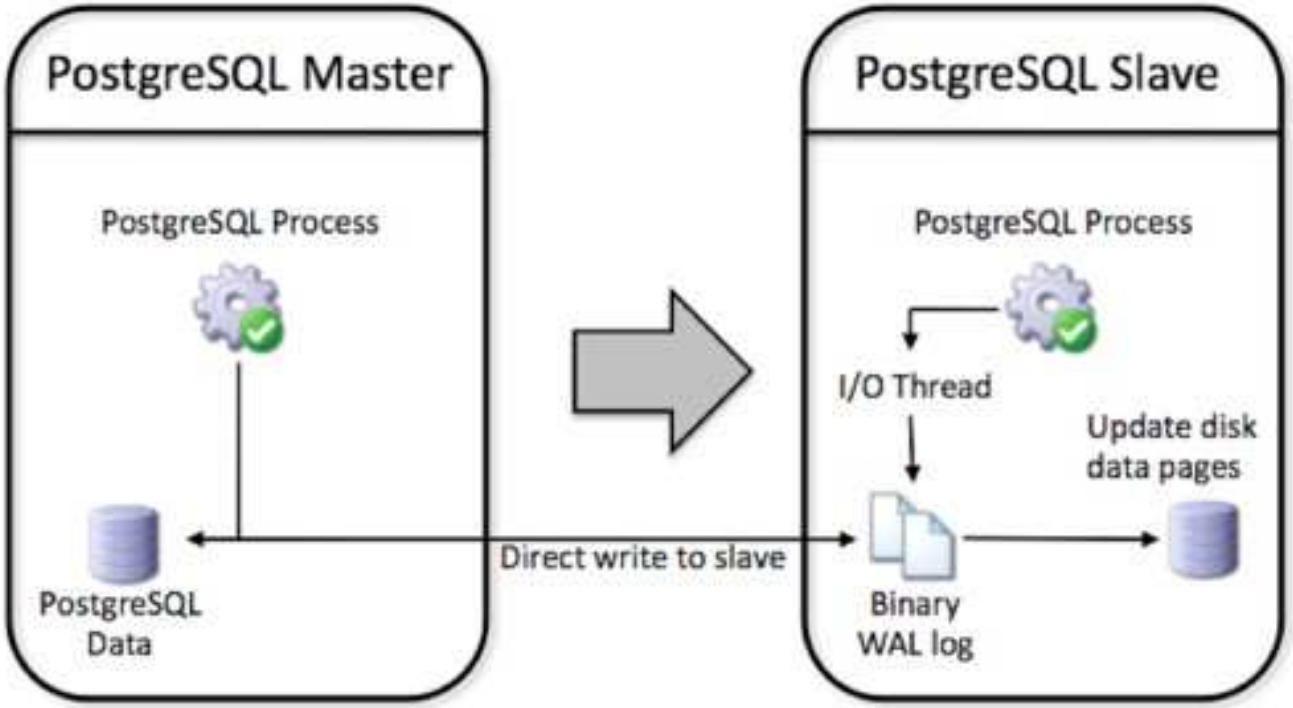
# Další možnosti replikování MySQL

- MySQL Proxy
- MySQL NDB Cluster
- Continuent Tungsten middleware
- Federated storage engine
- HiveDB
  - partitioning

# Replikace PostgreSQL

- WAL (Write-Ahead Log) od verze 7.1
- asynchronní
- plná podpora až od verze 9.0
- skromné replikační scénáře
  - single master, single slave
  - single master, multiple slaves
  - ... a nic víc :)

# Replikace PostgreSQL




# Replikace PostgreSQL

- úpravy z masteru jsou zapisovány přímo do WAL na slave
- slave promítá změny a to přímo raw data, ne jednotlivé dotazy kvůli rychlosti, tzn. je bezpečnější pro `INSERT`y s `NOW()`, `function()` apod.)
- také single-threaded

# Nastavení replikace PostgreSQL

- master-slave replikace
  - PostgreSQL na masteru musí být síťově dostupný
  - `wal_level = hot_standby`
  - `archive_mode = on`
  - `pg_dump` & restore
  - `hot_standby = on`
  - `standby_mode = on, primary_conninfo`

# Co nás čeká v nové verzi

- synchronní replikace v 9.1
- “no formal requirements for each PostgreSQL release” :)
- Postgres-R
  - multi-master
- Postgres-XC
  - synchronní multi-master shared-nothing

# Další možnosti replikování PgSQL

- Slony-I
  - trigger-based replikace
  - master-slave, s širší podporou replikačních scénářů

# Další možnosti replikování PgSQL

- Slony-I
- Londiste
  - replikační systém nad PostgreSQL
  - součástí SkyTools od Skype

# Další možnosti replikování PgSQL

- Slony-I
- Londiste
- Bucardo
  - také multi-master replikace
  - aplikační server v Perlu

# Další možnosti replikování PgSQL

- Slony-I
- Londiste
- Bucardo
- pgpool-II
  - aplikační proxy

# Další možnosti replikování PgSQL

- Slony-I
- Londiste
- Bucardo
- pgpool-II
- Continuent Tungsten middleware
  - jako u MySQL

# Další možnosti replikování PgSQL

- Slony-I
- Londiste
- Bucardo
- pgpool-II
- Continuent Tungsten
- EnterpriseDB

# Srovnání MySQL a PostgreSQL

- MySQL
  - statement-based
    - menší logy, ale pozor na UDF, DML apod.
  - row-based
- PostgreSQL
  - latter-based
    - nejbezpečnější

# Srovnání MySQL a PostgreSQL

- MySQL
  - mnoho replikačních scénářů
    - včetně “ring replication”
- PostgreSQL
  - pouze pár replikačních scénářů

# Srovnání MySQL a PostgreSQL

- MySQL
  - umožňuje filtrování replikovaných dat (například replikace konkrétní tabulky)
- PostgreSQL
  - filtrování není podporováno

# Srovnání MySQL a PostgreSQL

- MySQL
  - single-threaded replication
- PostgreSQL
  - single-threaded replication

# Srovnání MySQL a PostgreSQL

- MySQL
  - `SHOW` příkazy pro správu a monitoring
- PostgreSQL
  - možnost sledovat datový rozdíl (vzdálenost) mezi master a slave

# Srovnání MySQL a PostgreSQL

- oba systémy jsou dobrou volbou na replikování
- každý má svá pro a proti

# Replikace pomocí Slony-I

- Slony-I umožňuje asynchronní, master-slave replikaci založenou na TRIGGERech v databázi s daty
- sleduje změny v datech a následně je provádí v celém replikačním clusteru
- jeho konfigurace je poměrně složitá

# Principy replikace pomocí Slony-I

- master node s několika slaves
- `AFTER ROW` triggery na unikátních řádcích
- replikační daemoni (Slon)
- replikace events
- konfigurační jazyk Slonik

# Příklad

- jednoduchá replikace mezi dvěma nody
- instalace PostgreSQL, Slony
- konfigurace PostgreSQL pro síťovou komunikaci
- vytvoření uživatelů pro Slony
- podpora PL/pgSQL
- konfigurace pro Slonik

# Konfigurace pro Slonik

```
#!/bin/sh

slonik << EOF

cluster name = first;

node 1 admin conninfo = 'dbname=pgbench host=192.168.1.1 ...';

node 2 admin conninfo = 'dbname=pgbenchslave host=192.168.1.2 ...'

init cluster (id=1, comment = 'Master Node');

...

create set (id=1, origin=1, comment='All pgbench tables');

set add table (set id=1, origin=1, id=1,

fully qualified name = 'public.accounts', comment='accounts table');

...

EOF
```

# Konfigurace pro Sloník – SQL

- Slonik překládá do SQL, které se dá volat také přímo:

```
-- Initialize local node

SELECT @NAMESPACE@.initializelocalnode(@MASTER_NODE_ID@,
   '@MASTER_NODE_NAME@');

SELECT @NAMESPACE@.enablenode(@MASTER_NODE_ID@);

-- Create replication set

SELECT @NAMESPACE@.storeset(@REPSET_ID@, '@REPSET_NAME@');
```

# Konfigurace pro Slonik

```
#!/bin/sh

slonik << EOF

cluster name = first;

node 1 admin conninfo = 'dbname=pgbench host=192.168.1.1 ...';

node 2 admin conninfo = 'dbname=pgbenchslave host=192.168.1.2 ...'

init cluster (id=1, comment = 'Master Node');

...

create set (id=1, origin=1, comment='All pgbench tables');

set add table (set id=1, origin=1, id=1,

fully qualified name = 'public.accounts', comment='accounts table');

...

EOF
```

# ... děkuji za pozornost

Použité zdroje:

- Wikipedia [http://www.slony.info/]
- Slony-I Homepage [http://www.slony.info/]
- http://www.theserverside.com/feature/Comparing-MySQL-and-Postgres-90-Replication
- MySQL.com, PostgreSQL.com