PA036: Projekt z DB systémů

Vlastislav Dohnal

Cíle předmětu

- Praktické používání databázového systému
  - Využití rozšířených vlastností
- Vytvoření týmového projektu a jeho prezentace
  - Tým = 3-4 studenti
- Témata projektů: Nové možnosti DB systémů
  - Představení problematiky
  - Otestování výkonnosti
  - Porovnání s jinými možnostmi

Databázové systémy

- Relační – PostgreSQL
- NoSQL – dokumentové, klíč-hodnota, sloupcové, grafové
- Kombinace
  - Využití NoSQL přes PostgreSQL
  - Tzv. Foreign Data Wrappers - https://wiki.postgresql.org/wiki/Foreign_data_wrappers

Dokumentové NoSQL databázové systémy

- MongoDB
  - Datový model - **JSON dokument**
  - Dynamické schéma
  - Primární přístup k dokumentu pomocí ID
  - Sekundární přístup – index nad vybraným atributem dokumentu
- CouchDB
  - Analogický k MongoDB

Klíč-hodnota NoSQL databázové systémy

- Redis
  - Datový model – dvojice klíč-hodnota
  - Primitivní, vysoce výkonné operace nad seznamy, množinami a asoc. poli
  - Často jako in-memory cache – lze nastavit exspiraci záznamů
  - Zpracování GEO souřadnic – včetně rozsahových dotazů
- Riak
  - Analogický k Redis, navíc zvládá JSON dokumenty jako hodnoty, umí sekundární indexy
    - Indexy se musí udržovat ručně

Sloupcové / grafové NoSQL DB systémy

- Cassandra
  - Relační datový model
  - Variabilní schéma – skupiny sloupců
  - CQL jazyk podobný SQL
- Neo4j
  - Datový model – orientovaný multigraf

Témata projektů

- Výkonnost zpracování JSON (1 tým)
- Správa uživatelských profilů (1 tým)
- Aplikační framework a kešování DB (1-2 týmy)
- Notifikace DB serveru a aplikační framework (1-2 týmy)
- Autentizace DB spojení a aplikační framework (1-2 týmy)
- High Availability in PostgreSQL (1-3 týmy)
- Zpracování GEO dat (1-2 týmy)

Projekt: Výkonnost zpracování JSON (1 tým)

- Představte práci s JSON v PostgreSQL i MongoDB
  - Na příkladu demonstrujte použití
- Porovnejte MongoDB a PostgreSQL ve zpracování JSON dokumentů
  - Funkcionalita
  - Výkonnost
- Prezentace projektu:
  - První:
    - Představení práce s JSON
    - Základní porovnání
    - Návrh vhodných experimentů, resp. malé web aplikace, která data ukládá v JSON
  - Druhá:
    - Stručně obsah první
    - Popis experimentu (nastavení, data, …)
    - Výsledky experimentů

Projekt: Správa uživatelských profilů (1 tým)

- Implementujte správu uživatelských profilů webového systému v PostgreSQL
  - Status přihlášení uživatele je každou minutu obnovován
  - Neaktivní uživatel (déle jak 3 min.) je automaticky odhlášen
  - Vypisujte počet přihlášených uživatelů
- Porovnejte s implementací v NoSQL DB (např. Redis)
  - Výkonnost obou systémů vzhledem k celkovému počtu uživatelů, ale i počtu aktivních uživatelů
  - Využijte „foreign data wrapper“ v PostgreSQL

Projekt: Správa uživatelských profilů (pokr.)

- Prezentace projektu:
  - První:
    - Představení implementace v PostgreSQL, včetně www stránky pro login/logout a obnovování stavu
    - Volba NoSQL DB
    - Návrh vhodných výkonnostních experimentů
  - Druhá:
    - Stručně obsah první
    - Popis implementace v NoSQL DB
    - Popis experimentu (nastavení, data, …)
    - Výsledky experimentů
- Uživatelský profil by měl obsahovat:
  - *user_account* (company_key bigint (FK), username varchar (PK), password varchar, ui_lang char(2), ui_settings varchar, person_key bigint (FK), username_db varchar (UNIQUE), employee_account boolean, access_rights char(96), logon_status int (0=logged off, 2=logged on), logon_last_modif timestamp)
  - logon_status a logon_last_modif je nastavován po přihlášení a dále periodicky obnovován; po odhlášení uživatele je nastaven na logged off.

Projekt: Aplikační framework a kešování DB (1-2 týmy)

- Prostudujte možnosti kešování výsledků databázových dotazů pro některý z aplikačních frameworků, např. Laravel nebo Nette pro PHP.
- Implementujte příklad, na kterém předvedete funkčnost a výkonnost řešení.
- Prezentace projektu:
  - První:
    - Základní představení zvoleného frameworku
    - Volba příkladu pro implementaci, resp. web aplikace
    - Návrh vhodných experimentů, měření přístupu k datům v DB (čtených bloků tabulek, disk IOs, memory usage, …)
  - Druhá:
    - Stručně obsah první
    - Způsob realizace kešování
    - Popis a výsledky experimentu (nastavení, data, …)

Projekt: Notifikace DB serveru a aplikační framework (1-2 týmy)

- PostgreSQL umožňuje klientům posílat notifikace při libovolné události pomocí příkazu NOTIFY, resp. přijímat pomocí LISTEN.
- Analyzujte možnosti použití v aplikačních frameworcích (např. Laravel) a porovnejte s nativní implementací pomocí JDBC.
- Prezentace projektu:
  - První:
    - Představení problému, vhodného příkladu a jeho implementace pomocí JDBC
    - Volba frameworku pro možnost implementace do jeho DB abstrakce
    - Představení experimentů (měření doby od vzniku události po aktualizaci dat, zvětšování počtu událostí a porovnání s naivním přístupem)
  - Druhá:
    - Stručně obsah první
    - Způsob realizace ve zvoleném frameworku
    - Popis a výsledky experimentů

Projekt: Autentizace DB spojení a aplikační framework (1-2 týmy)

- Aplikační frameworky poskytují podporu pro získávání/ukládání uživatelských dat, ale i účtů v databázích.
- Autentizace uživatelů je typicky řešena frameworkem a přístup k databázi je pomocí generického účtu.
- Analyzujte možnosti propagace autentizace uživatele až na DB spojení, tj. nikoli pouhé ukládání uživatelských účtů v DB, ale využívání stejných credentials i na ustanovení DB spojení.
- Vhodné řešení implementujte pro PostgreSQL.
  - http://www.postgresql.org/docs/current/static/auth-pg-hba-conf.html

Projekt: Autentizace DB spojení a aplikační framework (pokr.)

- Prezentace projektu:
  - První:
    - Představení problému, vhodného příkladu
    - Volba vhodného frameworku (s ohledem na rozšiřitelnost „factory“ pro realizaci autentizace specifické dané session)
    - Implementace primitivní www aplikace včetně implementace (v PostgreSQL použití views s klazulí „owner=current_user“)
    - Představení experimentů (měření času volání sady různých login requestů v navrženém řešení vs. naivní řešení)
  - Druhá:
    - Stručně obsah první
    - Způsob realizace ve zvoleném frameworku
    - Popis a výsledky experimentů, praktická ukázka

Projekt: High Availability in PostgreSQL (1-3 týmy)

- Nastudujte si problematiku master-slave replikace a nakonfigurujte tuto replikaci pro dvojici serverů
- Navrhněte vhodné experimenty pro ověření funkčnosti a chování řešení při výpadku jednoho ze serverů
  - změřte čas pro „fail-over“ na slave,
  - čas obnovené plné funkčnosti po náběhu chybějícího serveru, …
  - nalezněte případné nekonzistence/chybějící data
- Použijte vybranou možnost:
  - nativní replikace PostgreSQL
  - SkyTools 3 (http://skytools.projects.pgfoundry.org/skytools-3.0/)
  - Slony
  - Pgpool-II
- Prezentace: první – zvolená možnost a její představení, druhá – řešení a výsledky

Projekt: Zpracování GEO dat (1-2 týmy)

- Geografická data o pohybu vozidel a vztah k mapovým systémům
  - Zobrazování a plánování tras vozidel (kde nyní jsem, kde jsem měl být, kdy tam budu ((-;)
  - Analýza dat k využití auta, detekce dopravních omezení, změny trasy
- Využití OLAP v DB, propojení s OpenStreetMap (http://wiki.openstreetmap.org/wiki/Routing)
- Vytvoření aplikace/dashboardu
  - Zobrazení pozic aut na mapě, jejich tras
  - Statistiky času jízdy/stání aut (podle typu komunikace,…), …
  - Zobrazení stavů průjezdnosti silnic (tj. barva silnice podle rychlosti jízdy)
- Prezentace projektu:
  - První: Představení aplikace, její funkcionality; volba implementačního frameworku
  - Druhá: Stručně obsah první, realizace ve zvoleném frameworku, předvedení

Projekt: Zpracování GEO dat (pokr.)

```
CREATE TABLE public.car
(
  car_key bigint NOT NULL DEFAULT nextval('car_car_key_seq'::regclass),
  company_key bigint,
  spz character varying(14) NOT NULL,
  make character varying(50) NOT NULL,
  color character varying(50),
  production_year date NOT NULL,
  type character(1) NOT NULL,
  tonnage numeric(4,1) NOT NULL,
  capacity integer DEFAULT 0,
  length numeric(4,2),
  width numeric(3,2),
  height numeric(3,2),
  pos_vicinity integer NOT NULL DEFAULT 0,
  description character varying(256),
  driver_key bigint,
  home_stand_key character(3),
  last_modif timestamp with time zone DEFAULT now(),
  axles smallint,
  consuption_max numeric(4,1),
  eco_speed integer,
  model integer,
  CONSTRAINT car_pkey PRIMARY KEY (car_key),
  CONSTRAINT car_spz_key UNIQUE (spz)
)
```

```
CREATE TABLE public.tracking
(
  pos_key bigint NOT NULL DEFAULT nextval('tracking_pos_key_seq'::regclass),
  car_key bigint NOT NULL,
  driver_key bigint,
  "time" timestamp with time zone NOT NULL,
  cargo_status character(1) NOT NULL,
  pos_gps point,
  city_key character(3),
  distance numeric(7,2),
  speed numeric(5,2),
  truck_status character(1),
  driver2_key bigint,
  cargo_weight numeric(6,3),
  driving_time integer,
  CONSTRAINT tracking_pkey PRIMARY KEY (pos_key)
)
```