BI-BIG - semestrální práce

Výsledky voleb 2017
a
demografické informace

Ladislav Martínek 25.12.2017

1. Úvod

Cílem této práce je předvést ukázku práce s vybranými databázovými systémy, které jsou navrženy pro distribuované ukládání dat a jejich paralelní zpracování, tak aby bylo možné pracovat s velkými objemy dat. Práce bude předvedena na mnou 7 vybraných datových sadách, které byly staženy z otevřených dat poskytovanými Českým statistickým úřadem a souřadnice obci z Git Hubu z publikované datové sady. Některá data byla předzpracovaná např. upraveno kódování pro univerzální použití mezi systémy. Pro uložení, dotazování, agregaci a shrnutí dat jsem si vybral nástroj Apache Hive. Pro indexaci, analýzu a vizualizaci jsem zvolil Elasticsearch a Kibanu. V těchto nástrojích zpracuji vybrané datové sady o výsledcích voleb a demografických údajích na území České republiky. Mezi datovými sadami jsou zahrnuty také číselníky pro celé názvy místo identifikačních čísel. Veškeré ukázky jsou prováděné na Linuxu Debian verze 18. Projekt je dostupný na adrese https://gitlab.fit.cvut.cz/martilad/BIG_semestral_work.git na školním Git Labu. Podle dokumentace je možné sestavit celý experiment.

2. Práce

Popis databáze

K práci jsem si vybral především datové sady poskytované Českým statistickým úřadem. https://data.gov.cz.opendata.cz/. Dále také datovou sadu z http://33bcdd.github.io/souradnice-mest/. Konkrétně se jedná o data výsledku voleb do poslanecké sněmovny v roce 2017 pro jednotlivé politické strany. K této datové sadě je aktuální číselník vylosovaných čísel politických stran v těchto volbách a obecné číslo přiřazené každé politické straně. Dalším číselníkem jsou názvy politických stran na obecné číslo přiřazené politické straně při vzniku. Další vybranou datovou sadou jsou data o populaci. Bohužel jsem byl nucen využít zastaralá data ze sčítání lidu v roce 2011. Nad těmito agregovanými datovými sadami jsem vytvářel vizualizace v Kibaně. Další datovou sadou je datová sada s demografickými daty o populaci v roce 2016. V této sadě jsou ukazatele o pohybu, přírůstku a úbytku obyvatelstva v jednotlivých obcích. V této datové sadě lze třeba sledovat procento přistěhovalců a přirozeného přírůstku v jednotlivých obcích. Poslední datovou sadou je datová sada se souřadnicemi jednotlivých obcí v České republice.

Popis a ukázka použitých datasetů

Datové sady jsou obsaženy v repositáři který lze stáhnout:
`git clone git@gitlab.fit.cvut.cz:martilad/BIG_semestral_work.git`
Datové soubory budo dostupné v BIG_semestral_work/data
Všechny datové sady jsou ve formátu CSV a kódování utf-8.

1. **ciselnik_pol_stran.csv (záznamů: 889)**

Datová sada obsahuje názvy politických stran k jejich kódum.
Sloupce popořadě:
VSTRANA - kód volební strany
NAZEVCELK - název volební strany
NAZEV_STRV - název strany 50 znaku
ZKRATKAV30 - zkratka 30 znaku
ZKRATKAV8 - zkratka 8 znaku
POCSTR_SLO - počet politickych subjektů ve složení volební strany
SLOZENI - složenní strany kód
ZKRATKA_OF - oficialni zkratka strany
TYPVS - typ volebni strany S - politicka strana, K - koalice, N - nezavisli, D - zdruzeni
Ukázka:

```
1,Křesťanská a demokratická unie - Československá strana lidová,Křesťan.a
demokrat.unie-Českosl.strana lidová,Křesť.demokr.unie-Čs.str.lid.,KDU-ČSL,1,001,KDU-ČSL,S
```

2. **demograficka_data.csv (záznamů: 52439)**

Datová sata obsahuje demografická data k uvedeným územním částem.
Sloupce popořadě:
IDHOD - unikatni id databáze csu
HODNOTA - zzjištěná hodnota
VUK - kód výstupního ukazatele veřejné databáze
VUK_TEXT - název výstupního ukazatele
STAPRO_KOD -kód statistické proměné ze systému SMS UKAZ
VUZEMI_CIS - kód číselníku pro referenční území, typologie území
VUZEMI_KOD - kód položku pro referenční území
ROK - referenční období
CASREF_OD - začátek intervalu
CASREF_DO - konec intervalu
VUZEMI_TXT - text z položky pro referenční území
Ukázka:

```
758382045,13,DEM0007,Živě narození,4355,43,500011,2016,2016-01-01,2016-12-31,Želechovice nad
Dřevnicí
```

3. **scitani_lidu_2011.csv (záznamů: 6713)**

Datová sada obsahuje údaje nasbírané během sčítání lidu v roce 2011.
Sloupce popřadě:
TYP_UZ_NAZ - typ území, stát, kraj, okres, obec
NAZEV - název území
UZCIS - kód území
UZKOD - kód územního číselníku
U01 - obyvatelstvo celkem
U02 - obyvatelstvo muži
U03 - obyvatelstvo ženy
U04 - lidi věk 0-14
U05 - lidi věk 15-64
U06 - lide 65 a více
U07 - ekonomicky aktivní
U08 - ekonomicky aktivní zaměstnaní
U09 - obydlené domy
U10 - obydlené byty
U11 - hospodařící domácnosti
Ukázka:

```
okres,Zlín,101,40851,190488.0000,92475.0000,98013.0000,26819.0000,131235.0000,31945.0000,94172
.0000,85673.0000,36519.0000,72773.0000,76823.0000
```

4. **souradnice.csv (záznamů: 6253)**

Datová sada se souřadnice jednotlivých obcí.
Sloupce popořadě:
Obec, Kód obce, Okres, Kód okresu, Kraj, Kód kraje, PSČ, Latitude, Longitude
Ukázka:

```
Abertamy,554979,Karlovy Vary,CZ0412,Karlovarský kraj,CZ041,36235,50.368855,12.818377
```

5. **volby_posl_2017.csv (záznamů: 230303)**

Datová sada s výsledku pro každou politickou stranu pro každý územní celek v České republice.
Sloupce popořadě:
ID_OKRSKY - id okrsku
TYP_FORM - data 1 - t/4, 2 - příloha t/4, 3 - ts
OPRAVA - příznak opravy 1 - ano, 0 - ne
CHYBA - příznak chyby 1 - chyba, 0 - ne
OKRES - kód okresu
OBEC - kód obce
OKRSEK - kód okrsku
KC_1 - kontrolní číslo 1
KSTRANA - vylosované číslo strany
POC_HLASU - počet hlasů dané strany
KC_2 - kontrolní číslo 2
HLASY_01 - krožkování pro kandidata 1
…
HLASY_36 - krožkování pro kandidáta 36
KC_3 - kontrolní číslo 3
KC_4 - kontrolní číslo 4
POSL_KAND - poslední číslo kandidáta co měl přednostní hlasy
KC_SUM - sumární kontrolní číslo t4
Ukázka:

```
1,2,0,0,2103,532088,1,0,1,21,22,0,0,2,1,0,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,2,0,2,0,0,0,0,0,3,0,1,
0,0,0,0,0,238,260,31,534745
```

6. **kstrana_vstrana_2017.csv (záznamů: 32)**

Datová sada obsahuje čísleník vylosovaných čísel volebních stran na jejich kódy.
Sloupce popořadě:
KSTRANA - vylosováné číslo strany pro volby 2017
VSTRANA - přídělené číslo při vzniku strany
NAZEVCELK - název
Ukázka:

```
1,53,Občanská demokratická strana
```

7. **volby_scitani.csv (záznamů: 188651)**

Datová sada vytvořena spojením těchto datových sad pro import, indexování a vizualizaci pro technologii Elasticsearch a Kibanu. Datová sada je vytvořena podle příkazu v Kapitole Využití Apache Hive.

Technologie

Apache Hive

Je technologie vyvinutá ve Facebooku pro vytěžování dat. Je spravováno Apache Foundation. Technologie je postavena nad Hadoop. Hive je vrstva poskytující abstrakci pomocí pseudo SQL. Hadoop je open-source software pro distribuované uložiště a zpracování velkého množství dat. Využívá programovací princip MapReduce. Úloha je rozdělena na více částí. Jednou je map, kde jsou na každém uzlu mapovány proměné na key-value. Po této fázi probíhá bariérová synchronizace a Shuffle, kde si uzly vymění data. Poslední fázi je Reduce, který spojí výsledky z Mapové části.
Datový model Hive je rozdělen na databáze, což jsou jmenné prostory, který oddělují tabulky. Dále tabulky, partition, buckety. Tabulky jsou homogenní části dat. Partitions ovlivňuje, kde jsou data fyzicky uložena. Buckety jsou dělení dat v jednotlivých partitions. Hive používá koncept z SQL a datové typy, ale podporuje také složené datové typy například mapy, seznamy, struktury. K serializaci dat je možno specifikovat vlastní serializer. Dotazuje se pomocí Hive query language, které je podmnožinou SQL. Je zde omezení u join operací, je možné používat pouze predikáty rovnosti a specifikovány pomocí ANSI join syntaxe viz. dotazy v kapitole vyžití Apache Hive. Hive nepodporuje insert, každý insert přepisuje data. Chybí i Update i Delete. Je možné i specifikovat vlastní programy pro jednotlivé části např. Map.
Tabulky jsou logické celky, podle partitions jsou uloženy v jeho podadresářích. Buckety jsou jednotlivé soubory. Pokud nejsou partitionovány jsou data uložena na jednom místě. Lze pří uložení specifikovat i množství bucketů.
Hive se skládá z více částí, ale hlavní částí je metastore a query compiler. Metastore je systémový katalog pro Hive. Ukládá informace o tabulkách, partition, atd… Metastore je důležitá část systému. Query compiler vytváří plán vykonávání jednotlivých částí úlohy. Parsování, sémantická kontrola, optimalizace a generování MapReduce úloh.

Elasticsearch a Kibana

Elasticsearch je název pro fulltextový vyhledávač vycházející z Apache Lucene. Disponuje rozhraním poskytující vysokou dostupnost, rychlost a škálovatelnost. Elasticsearch je velmi rychlý. Dále je velmi dobře škálovatelný a umožňuje z provozu vyřazovat servery, které vykazují některé chyby. Vyhledávání je možné podle textu, polohy, podobné záznamy. Využívá bezschémovou databázi, schéma se vytváří na základě vložených dat. Komunikace probíhá prostřednictvím HTTP a požadavky jsou posílány ve formě JSON.
Kibana je plugin pro vizualizace Elasticsearch. Poskytuje možnost vizualizace indexovaného obsahu na clusteru Elasticsearch. Je možné vytvářet různé grafy, mapy k velkým objemům dat. V kibaně je možnost vytvářet celé dashbordy s grafy, které lze exportovat jako JSON.
Poslední součástí je Filebeat, který slouží k posílání dat do Elasticsearch a dávkovat data podle rychlosti indexace.

Využití Apache Hive

Spustění

Pro spuštění je důležité si nainstalovat docker. Vše je nutné dělat pod root.
Debian: https://docs.docker.com/engine/installation/linux/docker-ce/debian/
Dále také docker-compose.

```
curl -L https://github.com/docker/compose/releases/download/1.16.1/docker-compose-`uname -s`-`uname -m` -o /usr/local/bin/docker-compose
```

Přidělit práva.

```
chmod +x /usr/local/bin/docker-compose
```

Dále ve složce naklonované z Git Labu je složka Hive. V této složce rozbalte obsažený adresář a provedeme následující příkazy, kterými nám docker-compose sestaví požadované kontejnery a ty následně spustíme. (Při končení lze kontejnery postupně bezpečně zastavovat v opačném pořadí docker stop <k1> <k2> pro opětovné spouštění)
Příkazy:

```
cd <rozbalený adresář>
docker-compose build
docker-compose up -d namenode hive-metastore-postgresql
docker-compose up -d datanode hive-metastore
docker-compose up -d hive-server
```

Tímto máme spuštěný hive-server lokálně v docker kontejnerech.
Následujícími příkazy spustíme konzoli serveru.

```
docker exec -it hive-server bash
/opt/hive/bin/beeline -u jdbc:hive2://localhost:10000
```

Import dat

Nyní je třeba do kontejneru importovat data.

```
docker cp <soubor v počítači> <kontejner>:<umístění v kontejneru>
docker cp data/ciselnik_pol_stran.csv hive-server:/data/ciselnik_pol_stran.csv
```

Tento příkaz je potřeba provést pro všechny datové sady.

Dále je nutné pro datové sady vytvořit tabuly.

```
CREATE TABLE ciselnik_pol_stran(
VSTRANA int,
NAZEVCELK string,
NAZEV_STRV string,
ZKRATKAV30 string,
ZKRATKAV8 string,
POCSTR_SLO string,
SLOZENI string,
ZKRATKA_OF string,
TYPVS string ) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',';

CREATE TABLE kstrana_vstrana_2017(
KSTRANA int,
VSTRANA int,
NAZEVCELK string ) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',';

CREATE TABLE souradnice(
OBEC string,
KODOBCE int,
OKRES string,
KODOKRESU string,
KRAJ string,
KODKRAJE string,
PSC string,
LATITUDE double,
LONGITUDE double ) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',';

CREATE TABLE demograficka_data(
IDHOD int,
HODNOTA double,
VUK string,
VUK_TEXT string,
STAPRO_KOD int,
VUZEMI_CIS int,
VUZEMI_KOD int,
ROK int,
CASREF_OD string,
CASRED_DO string,
VUZEMI_TXT string) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',';

CREATE TABLE scitani_lidu_2011(
TYPUZ_NAZ string,
NAZEV string,
UZCIS int, UZKOD int,
U01 double, U02 double, U03 double,
U04 double, U05 double, U06 double,
U07 double, U08 double, U09 double,
U10 double, U11 double) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',';
```

Poslední tabulku nechám kvůli rozměrnosti bez formátování.

```
CREATE TABLE volby_posl_2017(ID_OKRSKY int,TYP_FORM int,OPRAVA int,CHYBA int,OKRES int,OBEC
int,OKRSEK int,KC_1 int,KSTRANA int,POC_HLASU int,KC_2 int,HLASY_01 int,HLASY_02 int,HLASY_03
int,HLASY_04 int,HLASY_05 int,HLASY_06 int,HLASY_07 int,HLASY_08 int,HLASY_09 int,HLASY_10
int,HLASY_11 int,HLASY_12 int,HLASY_13 int,HLASY_14 int,HLASY_15 int,HLASY_16 int,HLASY_17
int,HLASY_18 int,HLASY_19 int,HLASY_20 int,HLASY_21 int,HLASY_22 int,HLASY_23 int,HLASY_24
int,HLASY_25 int,HLASY_26 int,HLASY_27 int,HLASY_28 int,HLASY_29 int,HLASY_30 int,HLASY_31
int,HLASY_32 int,HLASY_33 int,HLASY_34 int,HLASY_35 int,HLASY_36 int,KC_3 int,KC_4
int,POSL_KAND int,KC_SUM int) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',';
```

A následně importovat data:

```
LOAD DATA LOCAL INPATH '/data/ciselnik_pol_stran.csv' overwrite into table ciselnik_pol_stran;
LOAD DATA LOCAL INPATH '/data/kstrana_vstrana_2017.csv' overwrite into table
kstrana_vstrana_2017;
LOAD DATA LOCAL INPATH '/data/souradnice.csv' overwrite into table souradnice;
LOAD DATA LOCAL INPATH '/data/demograficka_data.csv' overwrite into table demograficka_data;
LOAD DATA LOCAL INPATH '/data/scitani_lidu_2011.csv' overwrite into table scitani_lidu_2011;
LOAD DATA LOCAL INPATH '/data/volby_posl_2017.csv' overwrite into table volby_posl_2017;
```

Tímto máme vytvořený datové struktury v Apache Hive.

Dotazy

První dotaz agreguje data z datové sady demografických údajů. Počítá průměrnou hodnotu těchto ukazatelů přes všechny obce v České republice. Data dává do vytvořené tabulky.

```
CREATE TABLE prum_dem_ukaz AS
SELECT vuk_text demograficka_udaj,
       AVG(hodnota) AS prum_obec
FROM demograficka_data
WHERE vuzemi_cis = 43
GROUP BY vuk_text;
```

Druhý dotaz vytváří tabulku s celkovým přírůstkem obyvatel v České republice za rok 2016 z datové sady demografických údajů.

```
CREATE TABLE celkovy_prirustek_2016 AS
SELECT vuk_text demograficka_udaj,
       SUM(hodnota) AS prum_obec
FROM demograficka_data
WHERE vuzemi_cis = 43
  AND vuk ='DEM0012'
GROUP BY vuk_text;
```

Třetí dotaz vytváří tabulku s obcemi a procentem žen v těchto obcích z datové sady sčítání lidu.

```
CREATE TABLE procento_zen_obce AS
SELECT nazev,
       (u03/u01)*100 AS zeny
FROM scitani_lidu_2011
WHERE typuz_naz = 'obec';
```

Čtvrtý dotaz agreguje data z voleb do poslanecké sněmovny 2017 a sčítání lidu 2011. Počítá průměrné procentuální využití preferenčních hlasů a procento zastoupení daných kategorií lidí v populaci.

```
CREATE TABLE per_vyuz_prefer AS
SELECT nazev,
       u06/u01 AS duchodci,
       u07/u01 AS ve_veku,
       u04/u01 AS deti,
       AVG((HLASY_01 + HLASY_02 + HLASY_03 + HLASY_04 + HLASY_05 + HLASY_06 + HLASY_07 +
HLASY_08 + HLASY_09 + HLASY_10 + HLASY_11 + HLASY_12 + HLASY_13 + HLASY_14 + HLASY_15 +
HLASY_16 + HLASY_17 + HLASY_18 + HLASY_19 + HLASY_20 + HLASY_21 + HLASY_22 + HLASY_23 +
HLASY_24 + HLASY_25 + HLASY_26 + HLASY_27 + HLASY_28 + HLASY_29 + HLASY_30 + HLASY_31 +
HLASY_32 + HLASY_33 + HLASY_34 + HLASY_35 + HLASY_36)/4/poc_hlasu) * 100 AS per_pref
FROM scitani_lidu_2011
INNER JOIN volby_posl_2017 ON uzkod = obec
GROUP BY nazev,
         u01,
         u06,
         u07,
         u04;
```

Pátý dotaz připravuje data pro index a vizualizaci. Spojuje 5 datových sad a vybírá z nich chtěné údaje. Jednotlivé datové sady a atributy jsou vidět v příkazu níže.

```
CREATE TABLE volby_scitani AS
SELECT a.vstrana AS kod_strany,
       a.nazevcelk AS strana,
       a.zkratka_of AS zkrat_strana,
       c.poc_hlasu AS hlasu,
       d.obec AS obec,
       d.kodobce AS kod_obce,
       d.latitude AS latitude,
       d.longitude AS longitude,
       d.kraj AS kraj,
       d.okres AS okres,
       e.u01 AS obyvatelstvo,
       e.u02/e.u01 AS muzi,
       e.u03/e.u01 AS zeny,
       e.u04/e.u01 AS deti,
       e.u05/e.u01 AS stredni,
       e.u06/e.u01 AS duchodci,
       e.u07/e.u01 AS ek_aktiv,
       e.u08/e.u01 AS zamestnani
FROM ciselnik_pol_stran AS a
INNER JOIN kstrana_vstrana_2017 AS b ON a.vstrana = b.vstrana
INNER JOIN volby_posl_2017 AS c ON c.kstrana = b.kstrana
INNER JOIN souradnice AS d ON d.kodobce = c.obec
INNER JOIN scitani_lidu_2011 AS e ON e.uzkod = d.kodobce;
```

Export dat

Poslední vytvořenou datovou sadu jsem si exportoval do formátu CSV pro export do elasticsearch. Ukázka exportu.

```
insert overwrite local directory '/data' row format delimited fields terminated by ',' select
* from volby_scitani;
docker cp hive-server:/data/<name_of_gnerated_file>
<path_to_BIG_folder/BIG_semestral_work/data>
mv <name_of_gnerated_file> volby_scitani.csv
```

Využití Elasticsearch a Kibany

Spuštění konfigurace

Pro spuštění je nutné mít docker a docker-compose viz kapitola Využití Apache Hive.
Poté se ve staženém adresáři vnoříme do adresáře elastic+kibana a rozbalíme obsažený zip. Zanoříme se do rozbaleného adresáře. Příkazem docker-compose up spustíme Elasticsearch a Kibanu která bude dostupná na adrese http://localhost:5601. Vše je třeba dělat pod root.

Dalším krokem je instalace Filebeat pro import dat. Nasledujícími příkazy:

```
curl -L -O https://artifacts.elastic.co/downloads/beats/filebeat/filebeat-6.0.0-amd64.deb
dpkg -i filebeat-6.0.0-amd64.deb
```

V adresáři vytvoříme poadresář volby_scitani

```
mkdir volby_scitani
```

Do tohoto adresaře kopírujeme vytvořenou a exportovanou datovou sadu.

```
cp ../../data/volby_scitani.csv ./volby_scitani/
```

Zkopírujeme stažené konfigurační soubory.

```
cp ../config/volby_scitani_filebeat.yml ../config/volby_scitani_pipeline.json
../config/volby_scitani_template.json ./
```

Import data a upload vizualizace

Poté soubory importujeme na server.

```
curl --noproxy '*' -XPUT -H 'Content-Type: application/json'
'127.0.0.1:9200/_ingest/pipeline/volby_scitani' -d @volby_scitani_pipeline.json
curl --noproxy '*' -XPUT -H 'Content-Type: application/json'
'127.0.0.1:9200/_template/volby_scitani' -d @volby_scitani_template.json
/usr/share/filebeat/bin/filebeat -e -c volby_scitani_filebeat.yml
```

Teď se pomocí Filebeat importují data do Elasticsearch k indexování.
Kibana je k dispozici na adrese. (http://localhost:5601)

Otevření indexu. Pří otevření indexu lze zvolit timestamp jako časový údaj kdy byly nahrávána.
`Management tab » Index Patterns tab » Create New. Specifikovat volby_vision a timestamp`
Import vizualizací a dashboard.
`Management tab » Saved Objects tab » Import, a vybrat dashboard+vizu.json z konfig složky v elastic+kibana vše podtvrdit`
Poté otevřít dashboard. V souboru dashboard+vizu jsou specifikovány dotazy nad index.

Popis zhora a zleva:

- Průmerné procento žen v obcích znázorněné na mapě.
- Výsledky voleb s počtem hlasů pro jednotlivé strany.
- Průměrné procento důchodců v české republice jako mapa s lokací.
- Poté je sloupcový vertikální graf s procentem ekonomicky aktivních v jednotlivý krajích.
- Poté jsou výsledky voleb v jednotlivých krajích podle počtu hlasů pro pět nejlepších stran.
- Poslední je 10 okresů, ve který se nacházejí obce s největším procentem můžu v české republice.

Dev tools tab » dole je konzole lze psát přímo dotazy. Wild-chart dotaz na všechny obce obsahující klíčové slovo “Ohře”. A chci znát pouze obce.

```
GET _search
{
   "_source": ["obec"],
    "query": {
    "wildcard" : { "obec" : "*Ohř*" }
  }
}
```

3. Závěr

V semestrální práci jsem si vyzkoušel zpracování a analýzu reálných datových sad v nástrojích pro práci s velkými objemy dat distribuovaně. Narazil jsem i na nástrahy veřejné publikace dat, kde šlo především o různé kódování. Hive se mi líbil, protože jde ovládat pomocí SQL a distribuce a naplánování úloh udělá za nás (Například psaní MapReduce úloh). A podmnožina SQL u HQL nebyla pro zpracování vůbec omezující. Elasticsearch a Kibana je pěkná kombinace nástrojů pro pěkné vizualizace dat, které je možné zobrazovat průběžně jak jsou data přidávána a indexována. Práce na semestrální práci mě byla o to zajímavější, že byla prováděna na reálných datových sadách.