Slovenská technická univerzita

Fakulta informatiky a informačných technológií

Ilkovičova 3, 842 19 Bratislava 4

Peter Hamar

Lukáš Sekerák

Gergely Králik

Špecifikácia zadania

Github Archive

Študijný program: Informačné systémy / Softvérové inžinierstvo
Ročník: 4 – 5
Predmet: Pokročilé databázové technológie
Vedúci projektu: Ing. Tomáš Kramár
Ak. rok: 20013/2014

Obsah

Opis hlavnej témy:

Na cvičení sme sa dohodli, že našou hlavnou témou budú zdrojové kódy. Všetci budeme získavať dáta zo stránok ako sú Stackoverflow, Github Archive atď. Každá z týchto stránok zoskupuje údaje o projektoch ktoré sú na nich riešené. Konkrétne ide o údaje ako sú: repository, issues, bugs, users a rôzne ďalšie štatistiky. Na základe toho bude v ďalšej časti projektu možné získať zaujímavé výsledky.

Opis primárneho systému:

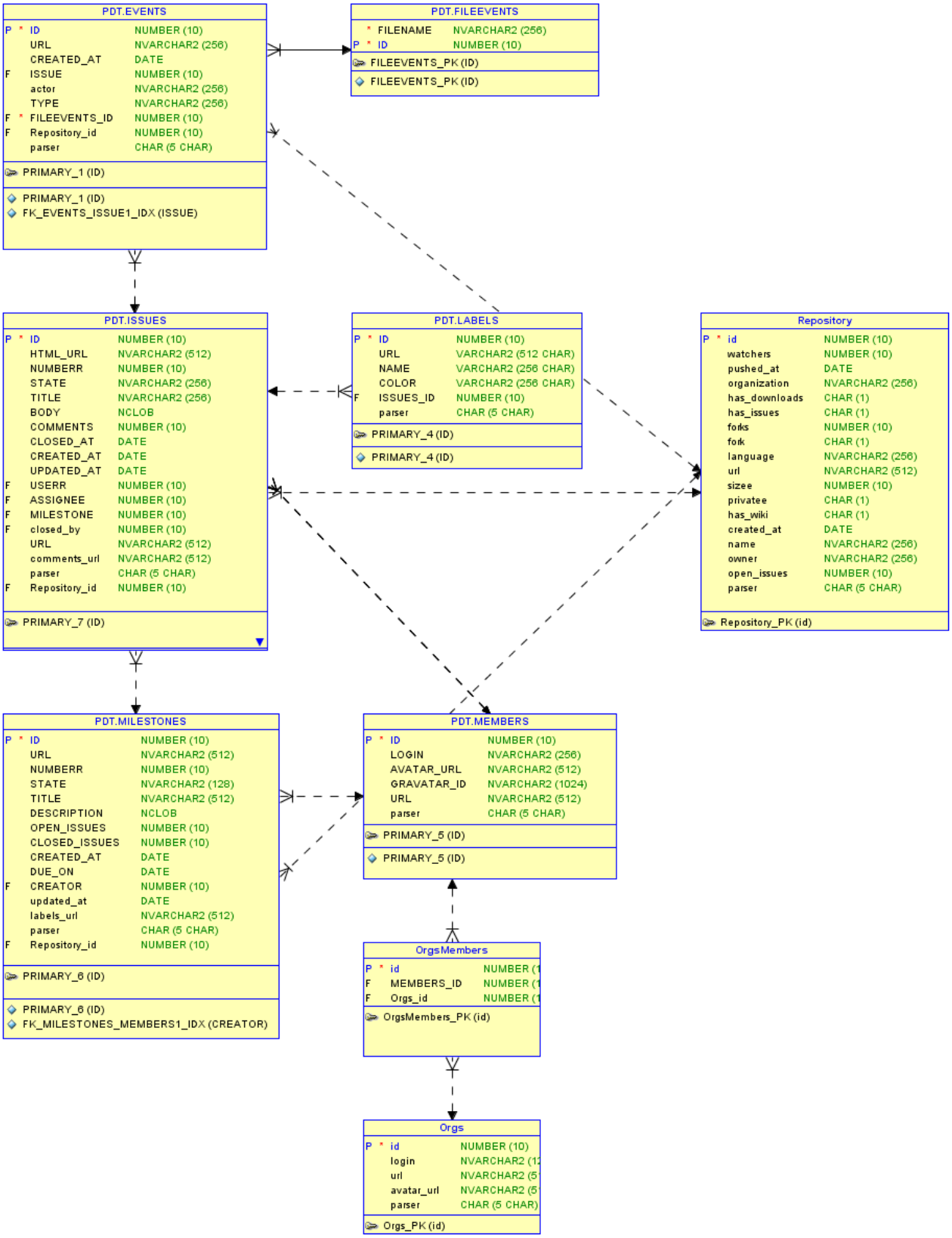
Náš tým sa bude venovať systému Github Archive. Z tohto systému budeme preberať údaje o týchto entitách: Orgs, Members, Issues, Events, Labels, Milestones, Repository. Podrobnejšie informácie o entitách sú v sekcii Popis tabuliek.

- Zdroj dát : Github Archive[1]

Teda v závere dodáme systém, ktorý bude poskytovať informácie o repozitároch, o chybách v týchto repozitároch a to všetko bude prepojené najmä na používateľov a organizácie na githube. Všetky issues budú mat dodatočné údaje v podobe labels a milestones. Hlavným zámerom sú teda issues a ďalšie vzťahy viac vysvetlí dátový model.

[1] www.githubarchive.org

Dátový model



Dátový model obsahuje 9 tabuliek a veľa slabých prepojení. Pri tvorbe diagramu nastalo veľmi veľa problémov. Z dokumentácie entít nebolo úplné jasne prepojenie. Zároveň v dátach entity neboli prepojene kľúčom.

Popis tabuliek

FileEvents – Pomocná tabuľka, pre ukladanie informácií ktoré súbory sme parsovali a teda aj odkiaľ sme jednotlivé dáta získali.
Events – Tabuľka, ktorá reprezentuje udalosť na github. Udalosť obsahuje informáciu, kedy bola vytvorená, kto vykonal akciu, na akom repozitári a o aký typ udalosti ide.
Issues – Obsahuje issues, ktoré boli vytvorené alebo upravené dňa 2012 04 11 atď.
Labels – Jednotlivé označenia pre issues.
Repository – Repozitáre
Milestones – Obdobia, ktoré issues má.
Members – Používatelia, ktorý založili repozitár, issue alebo sú členovia nejakej organizácie atď.
OrgsMember – Medzi-tabuľka medzi Orgs a Members
Orgs – Zoznam organizácií pre časť používateľov, ktorých máme uložených v Members

Postup riešenia

Prvým krokom k splneniu zadania bola podrobná analýza dokumentácie na **GITHUB archive**[2]. Následné sme zistili, že stránka poskytuje API v podobe **REST servisu**, na základe ktorého môžeme efektívne stiahnuť všetky dáta v **JSON**[3] formáte.

API je plne parametrizovateľné a môžeme stiahnuť dáta na základe dátumu a ďalších podmienok. Následné sme rozhodli stiahnuť údaje pre dátum 2012 04 11 a pre 5 hodín tohto dňa. Všetky tieto údaje sú uložené ako „Events“. Teda Events entita obsahuje udalosť, čo sa stalo na github, na akom repozitári a kto akciu vykonal.

S týchto údajov sme vyparsovali väčšinu údajov. Avšak údajov sme mali stále malo. Respektíve Eventov príliš veľa a pre ďalšie entity málo, preto sme sa rozhodli oparsovať :

- Pre malú vzorku Members (používateľov) sme oparsovali všetky organizácie, v ktorých je používateľ zapojený
- Pre malú vzorku repozitárov sme oparsovali, všetky jeho Milestony
- Pre malú vzorku repozitárov sme oparsovali, všetky jeho Issues

[2] Dokumentáciu k GITHUB Archive API - http://developer.github.com/v3/activity/events/types/

[3] Formát JSON - http://en.wikipedia.org/wiki/JSON

Popis aplikácie

Aplikáciu sme pripravili v Jave prostredí, kde sme pripravili triedu ktorá sa stará o GitHubArchive REST službu, teda automaticky sťahuje dáta, následné ich rozbalí a prípravy na ďalšie spracovanie. To všetko sa vykoná v rýchlom streame, takže riešenie je dosť rýchle.

Následné JSON dáta za pomoci knižnice **google-gson**[4] automaticky na-mapuje na naše entity – triedy. Zároveň sme v Jave využili JPA – **Java persistence api**[5], ide o objektový mapovač, ktorý dokáže našu entitu priamo na-mapovať do databázy a jednoduchým príkazom ju uloží.

Vďaka týmto knižniciam naše riešenie je rýchle a zdrojového kódu nie je veľa. JPA sme na-mapovali priamo na **Oracle DB**[6], čím sme opäť použili pokročilejšiu technológiu ako je MySql. Druhou výhodou je lepšia podpora volaní procedúr v databáze. (procedúry budú súčasťou druhého zadania). Treťou výhodou je vyber dát z databázy priamo do našich entít. (na základe objektového mapovača).

Základom projektu sú teda entity, ktoré sme si zadefinovali na začiatku. O všetko ostatne sa postará náš framework, ktorý sme si poskladali z knižníc.

Ďalšie technológie

Na tvorbu fyzického diagramu a prácu s Oracle databázou sme použili program **Oracle SQL Developer**[7]. Pre tvorbu záznamov čo sa deje v aplikácii a čo sa parsuje sme použili knižnicu **log4j**[8]**.** Pre lepšiu prácu so súbormi, ľahké sťahovanie údajov sme použili **Apache commons IO**[9]

CRUD operácie

Aplikácia poskytuje crud operácie, ktoré sme vygenerovali za pomoci **netbeans pluginu**. Crud operacie su poskytnuté v podobe webovej aplikácie, ktorá využíva **glassfish server** a JSF technológiu.

[4] Domáca stránka knižnice google-gson - https://code.google.com/p/google-gson/

[5] Popis technológie JPA - http://www.oracle.com/technetwork/java/javaee/tech/persistence-jsp-140049.html

[6] Link na stiahnutie Oracle Database - http://www.oracle.com/technetwork/database/enterprise-edition/downloads/index.html

[7] Oracle SQL Developer program - http://www.oracle.com/technetwork/developer-tools/sql-developer/overview/index.html

[8] Log4j knižnica - http://logging.apache.org/log4j/1.2/

[9] Apache commons IO knižnica - http://commons.apache.org/proper/commons-io/