Slovenská technická univerzita
Fakulta informatiky a informačných technológii
Ilkovičova 3, 812 19 Bratislava

Databázové systémy

Projekt

Elena Štefancová

Cvičiaci: Ing. Róbert Móro
Študijný odbor: Informatika
Ročník: 2. Bc
Akademický rok: 2015/2016

Obsah

1. Znenie zadania

Vo vami zvolenom prostredí vytvorte databázovú aplikáciu, **ktorá komplexne rieši minimálne 6 scenárov** vo vami zvolenej doméne. Presný rozsah a konkretizáciu scenárov si dohodnete s Vašim cvičiacim na cvičení. Aplikáciu vytvoríte v dvoch iteráciach. V prvej iterácii, postavenej nad relačnou databázou, musí aplikácia realizovať tieto všeobecné scenáre:

- Vytvorenie nového záznamu,
- Aktualizácia existujúceho záznamu,
- Vymazanie záznamu,
- Zobrazenie prehľadu viacerých záznamov (spolu vybranou základnou štatistikou),
- Zobrazenie konkrétneho záznamu,
- Filtrovanie záznamov spĺňajúcich určité kritériá zadané používateľom.

Aplikácia môže mať konzolové alebo grafické rozhranie. Je dôležité aby scenáre boli realizované realisticky - teda aby aplikácia (a teda aj jej používateľské rozhranie) naozaj poskytovala časť funkcionality tak, ako by ju očakával zákazník v danej doméne.

Scenáre, ktoré menia dáta musia byť realizované **s použitím transakcií** a aspoň jeden z nich musí zahŕňať **prácu s viacerými tabuľkami** (typicky vytvorenie záznamu a naviazanie cudzieho kľúča).

2. Konkretizácia zadania

Moja zvolená doména je vytvorenie systému na zadávanie úloh (ticket). V tomto prostredí sú hlavnými entitami zamestnanec firmy (staff), zadávateľ (customer) a požiadavka (ticket). Okrem toho, každý požiadavka má jeden z predefinovaných stav (stats = new / in progress / done) a každý zákazník prináleží nejakému oddeleniu.

Po spustení sa otvorí menu. Pozostáva zo šiestich volieb.

Pod tlačidlom „HR“ je správa zamestnancov. Po označení mena zamestnanca je ho možné zmazať. (*Scenár zmazania záznamu.*) Mená zobrazuje aj s identifikačným číslom zamestnanca. Taktiež je možné pridať nového zamestnanca.

Tlačidlo „Add a customer“ umožňuje pridať nového zákazníka z konkrétneho oddelenia.

„Show statistics“ zobrazuje, koľko dní priemerne trvá jednotlivým zamestnancov vyriešiť požiadavky (dostať ich do stavu Done od vytvorenia). (*Scenár zobrazenia viacerých záznamov a ich štatistiky.*)

„Filter tickets“ umožňuje filtrovať požiadavky podľa zamestnanca, na ktorého sú priradení, alebo aj ich aktuálneho statusu. (*Scenár filtrovania záznamov.*)

„Show tickets“ dokáže po kliknutí na požiadavku a tlačidlo „Open“ zobraziť základné údaje o konkrétnom zázname(*Scenár zobrazenia záznamu.*), pričom sa niektoré dajú aj editovať a následne uložiť tlačidlom „Save“. (*Scenár editovania, aktualizácie záznamu.*)

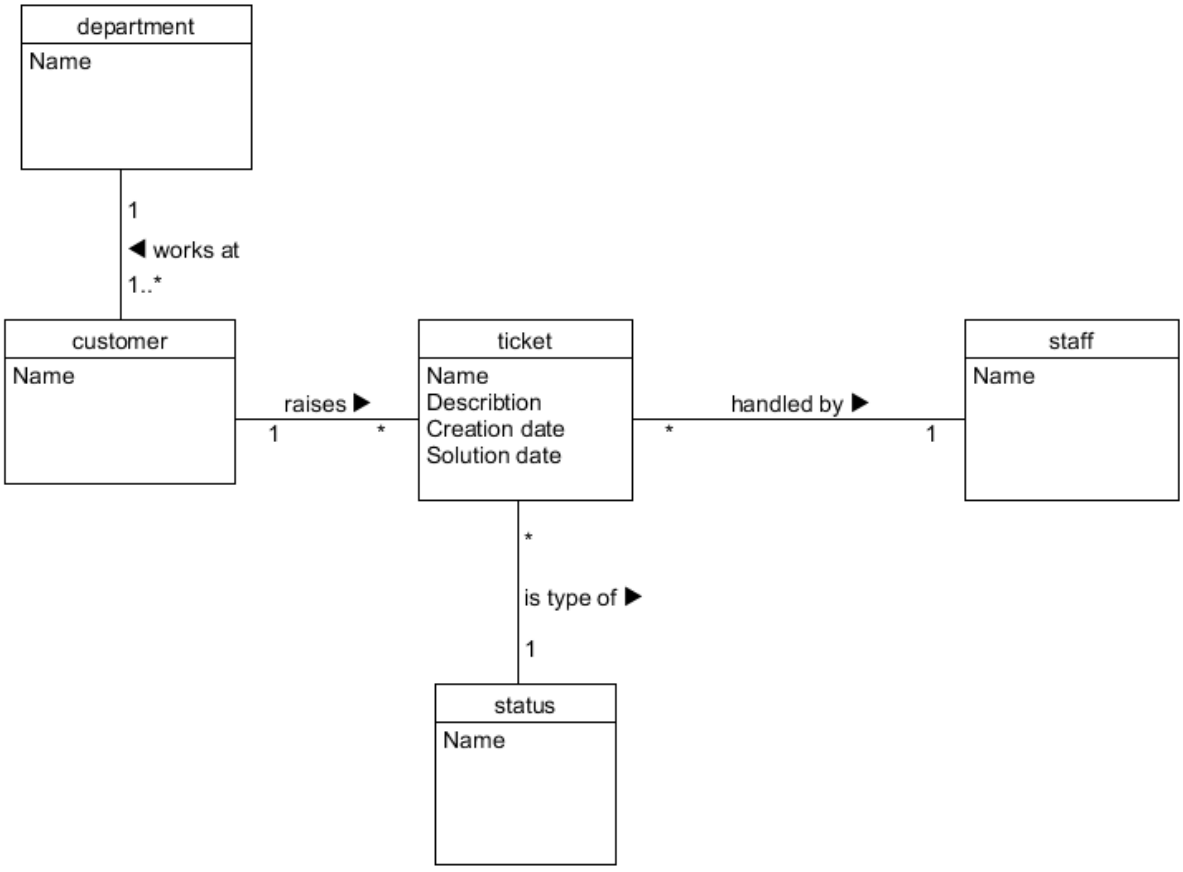
A konečne, „Add a new ticket“ dokáže vytvoriť záznam s určitým názvom, popisom, zadávateľom zo zákazníkov a zamestnancom, ktorý ho bude mať na starosti. (*Scenár vytvorenia záznamu.*) „Ticket“ na seba viaže teda najviac cudzích kľúčov zo všetkých entít.

Ako je vidno, spracovanie pozostáva z viacerých ako len povinných šiestich scenárov.

3. Analýza

Na implementáciu zadania som sa rozhodla používať programovací jazyk Java (vyvíjané v prostredí Mars.1 Release (4.5.1)) a čo sa databázovej strany týka SQL – PostgreSQL (v prostredí pgAdmin v1.22.1).

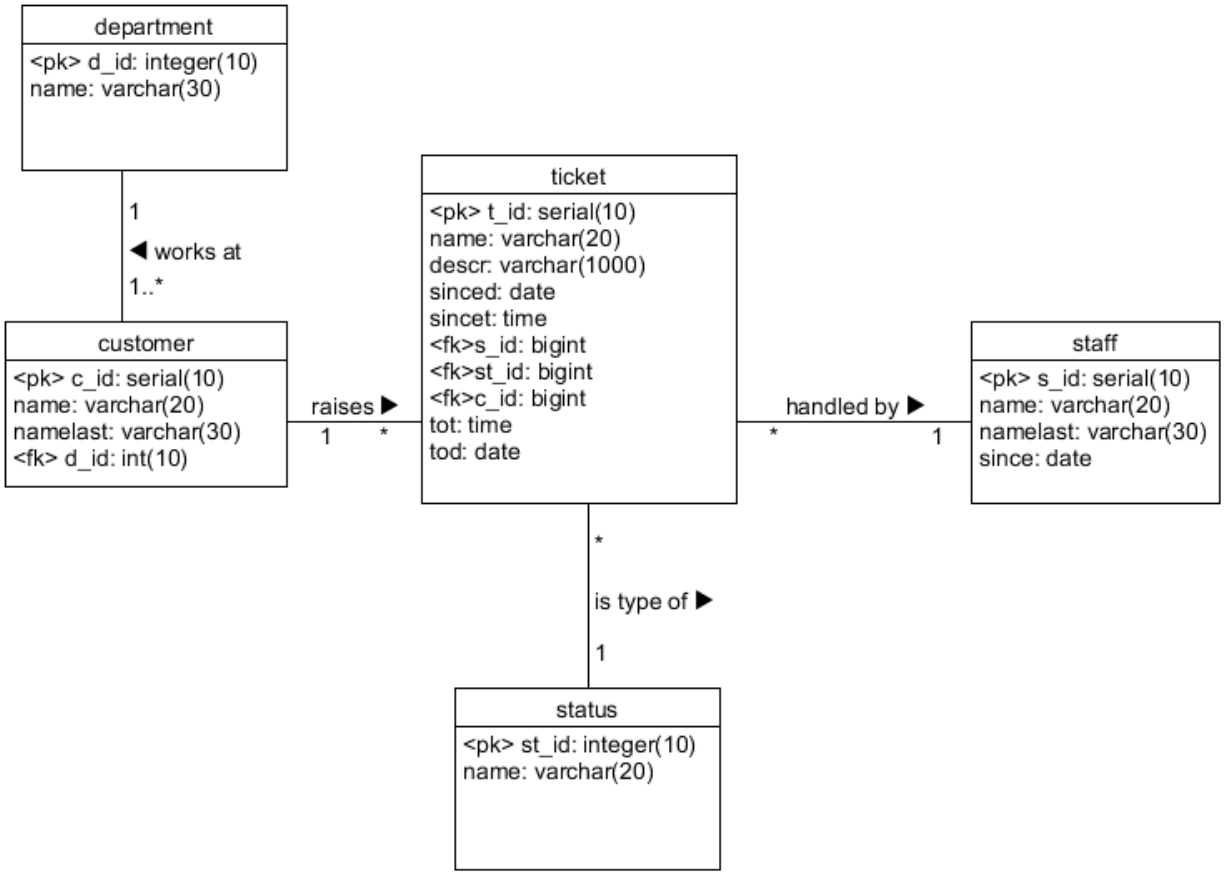
Databáza vychádza z nasledovného iagram:



Obrázok 1. – Diagram logického modelu

Ako je vidno, každý “ticket” je vznesený zákazníkom (“customer”, ktorý patrí k nejakému oddeleniu – “department”) a je riešený nejakým zamestnancom (“staff”). Taktiež má daný aktuálny status – tie sú dopredu určené 3 – New, In progress, Done (Nový, Pracuje sa na ňom, Hotový).

Podrobnejšie môžeme vzťahy vidieť na obrázku 2.



Obrázok 2. – Diagram fyzického modelu

Ako je vidno, každý prvok má svoje jedno či dvojčlenný názov, “ticket” má aj popis v maximálnej hodnote 1000 znakov. Okrem toho má aj čas a datum vytvorenia a vyriešenia (kedy sa mu zmení status na “Done”) – “sincet, sinced, tot, tod” – ako “since” a “to” plus “time” a “date”, teda čas a datum.

4. Implementácia

Aplikácia má grafické používateľské rozhranie, ktoré je vytvorené pomocou java swing komponentov. Rozhranie je ošetrené natoľko, aby bol používateľ vždy upozornení pri nesprávnom vyplnení údajov alebo keby sa pokúšal o nemožnú operáciu (napríklad pridať požiadavku na zamestnanca, ktorý je už zmazaný).

Zároveň dodržiava zásady MVC, teda sa snaží oddelovať logiku od grafiky v návrhu projektu.

Na pristupovanie k database sa využíva najmä trieda Connection.

4.1 Trieda Start

Trieda, ktorá slúži ako Main k celému program. Zabezpečí sa spojenie s databázov.

Potom sa otvorí hlavné menu, teda už časť grafického rozhrania.

4.2 Trieda model.EmptyException

Vlastná výnimka, ktorá sa využíva v prípade nevyplnených okien od používateľa v grafickom rozhraní, ktoré je ale nutné vyplniť.

4.3 HR

Funkcie týchto tried a zároveň aj rovnomenného okna v grafickom rozhraní, je vkladať a mazať zamestnancov, ak nastal nejaký preklep alebo podobne. Mazanie uskutočnené pomocou transakcie, ktorá zabraňuje, aby bol omylom zmazaný zamestnanec s prebranými požiadavkami.

Mazanie je určené najmä pre úpravu v prípade preklepu v mene pri nahadzovaní údajov do systému.

```
String sql = "BEGIN; DELETE from staff where s_id="+id+"; COMMIT;";
```

Obrázok 3. – Zdrojový kód mazania člena "staff"

4.4 Trieda Customer

Vytvorenie nového zákazníka (obrázok 7), ktorému musí byť ale tiež priradené oddelenie v ktorom pracuje.

4.5 Trieda NewTicket

Nový ticket môže byť vytvorený len po vybraní zamestnanca, riešiteľa a názvu s popisom.Vkladanie je ošetrené transakciou, aby nenastala chyba, kedy je ticket priradený už neexistujúcemu zamestnancovi. Pre ticket je tiež uložená časové a dátumová značka vytvorenia.

```
Savepoint savepoint = null;
try {
    savepoint = conn.setSavepoint();
    try {
        theModel.add(theView.textField.getText(), theView.textPane.getTex
        theView.textField.setText(null);
        theView.textPane.setText(null);
    } catch (EmptyException e) {
        theView.displayErrorMessage("Missing name/last name");
    }

} catch (SQLException e) {
    theView.displayErrorMessage("The member of staff does not exist.");
    if(conn != null && savepoint != null) {
        try {
            conn.rollback(savepoint);
        } catch (SQLException e1) {

        }
    }
```

Obrázok 4. – Rollback transakcie, keby zamestnanec neexistoval

4.6 Trieda ShowTicket

Aktualizácia už existujúceho záznamu, resp. v tomto prípade jeho popisu. Najprv ho ale treba zobraziť, čo tiež robí táto trieda.

Update záznamu nastáva nasledovným príkazom:

“`sql = "BEGIN; UPDATE ticket set name = '"+name+"', descr = '"+descr+"', st_id = '"+status+"', tod = NULL, tot = NULL  where t_id="+id+"; COMMIT;";`”

To ale len v prípade, že pôvodný stav ticket nebol done. V prípade že sa mení status ticket, z Done na InProgress, premazáva sa aj jeho čas uzavretia.

4.7 Trieda FilterTickets

Táto trieda umožňuje triediť záznamy ticket podľa ich status a zamestnanca, ktorý ich rieši.

```
else {
    s = "SELECT ticket.t_id, ticket.name, customer.name AS cn, customer.namelast AS cnl, status.name AS stn, staff.name AS sn, "
            + "staff.namelast AS snl,"
            + " department.name AS dn "
            + "FROM ticket INNER JOIN customer ON ticket.c_id=customer.c_id INNER JOIN status "
            + "ON ticket.st_id=status.st_id INNER JOIN staff ON ticket.s_id=staff.s_id INNER JOIN department "
            + "ON customer.d_id=department.d_id "
            + "WHERE ";
    //if a specific member of staff is selected
    if (id != -1){
        s = (s+"staff.s_id="+id+" ");
        pom=true;
    }

    //if new is selected
    if (selected1){
        if (pom){
            s = (s+"AND (");
            pom=false;
        }
        else
            s = (s+" (");

        s = (s+"status.st_id=1 ");
        temp = true;
    }
    //if InProgress is selected
    if (selected2){
        if (pom){
            s = (s+"AND (");
        }
        if (temp){
            s = (s+"OR ");
        }
        else if (!pom){
            s = (s+" (");
        }
        pom=false;

        s = (s+"status.st_id=2 ");
        temp = true;
    }
    //if Done is selected
    if (selected3){
```

Obrázok 4. – Časť selectu umožnujúceho filtrovanie ticketov

Využíva na to niekoľko JOINov.

4.8 Trieda ShowStatistics

ShowStatistics umožňuje zobraziť všetkých, alebo len vybraného zamestnanca, so štatistikou, ako priemerne dlho rieši jeden incident (od jeho vytvorenia po označenie “Done”).

```
String s="SELECT staff.name, staff.namelast, tabii.days FROM (SELECT s_id, AVG(datediff) AS days FROM (SELECT  datediff, t_id, st_id , "
        + "s_id FROM (SELECT DATE_PART('day', ticket.tod::timestamp - ticket.sinced::timestamp) AS datediff, ticket.t_id, ticket.st_id , "
        + "ticket.s_id FROM ticket WHERE ticket.st_id=3) AS tab) AS tabi GROUP BY s_id) AS tabii INNER JOIN staff ON tabii.s_id=staff.s_id "
        + "";
```

Obrázok 5. – Príkaz na zobrazenie danej štatistiky

Využíva na to agregačnú funkciu AVG a GROUP BY.

4.9 Prístup ku database

Na prístup sa využíva najmä trieda connection.

```
static final String JDBC_DRIVER = "org.postgresql.Driver";
static final String DB_URL = "jdbc:postgresql://localhost:5432/xstefancovae";

static final String USER = "postgres";
static final String PASS = "mojeN8Sheslo";

public static void main(String[] args) {

    try {
        //making new connection conn
        Class.forName("org.postgresql.Driver");
        Connection conn = DriverManager.getConnection(DB_URL, USER, PASS);
        conn.setAutoCommit(false);
```

Obrázok 6. – Vytvorenie spojenia

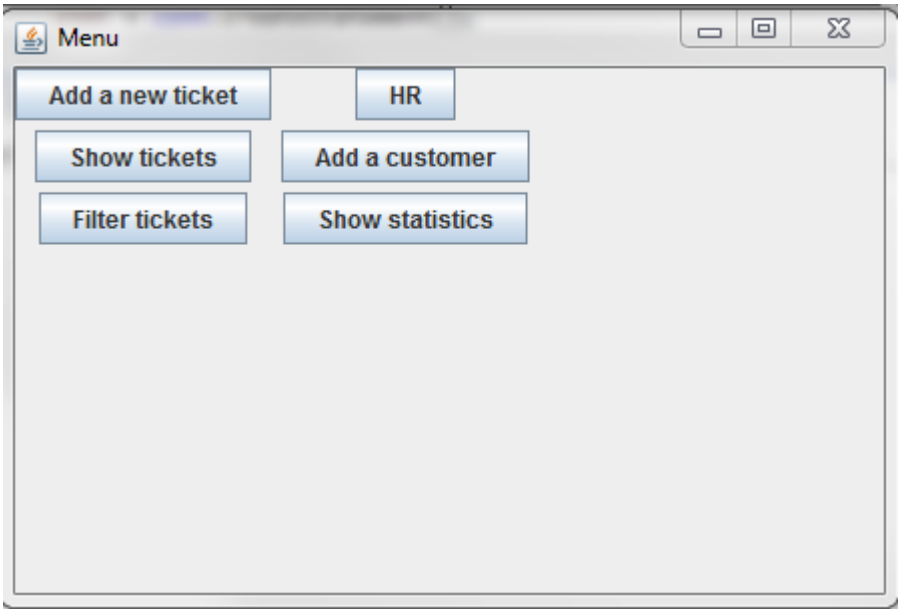
```
//add new customer
Statement stmt = null;
stmt = conn.createStatement();

String sql = "INSERT INTO customer (name,namelast,d_id)"
      + "VALUES ('"+name+"','"+lastname+"','"+id+"');";

stmt.executeUpdate(sql);
stmt.close();
```

Obrázok 7. – Implementovanie príkazu k database z javy

4.10 Používateľské rozhranie

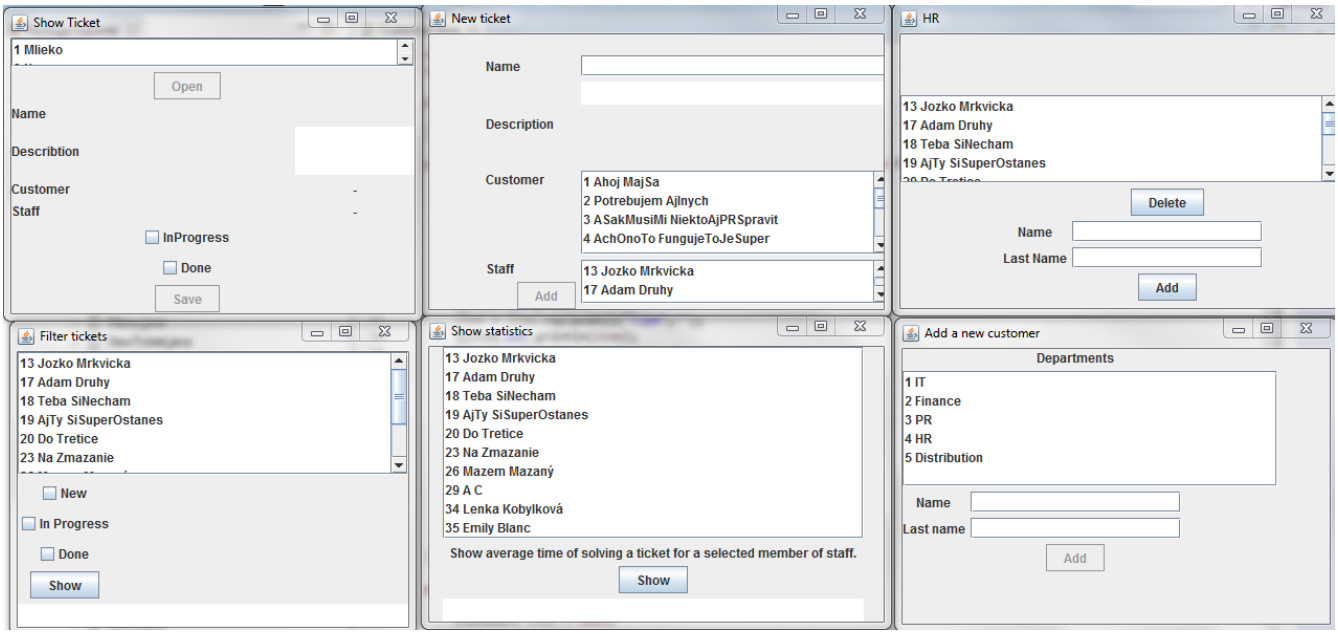
Rozhranie je riešené graficky.

Program po spustení zobrazí menu (obrázok 8).



Obrázok 8. – Menu

Tlačidlá umožňujú spustiť nasledovné časti aplikácie:



Obrázok 9. – Všetky vyskakovacie okná

5. Testovanie

Bolo otestovaných mnoho scenárov, ktoré zahŕňali vkladanie ticketu na neexistujúceho zamestnanca, zmena status z Done na InProgress a naspäť na Done, z toho vyplývajúce rôzne scenáre rátania štatistiky doby riešenia v dňoch, vkladanie prázdnych stringov do mien a podobne.

Všetky scenáre program zvládol.

6. Záver

Moja implementácia daného problému splnila základné požiadavky ustanovené v zadaní, zároveň niektoré veci doplnila, čo vyplývalo z tému, na ktorú sa project zameral.