PRIMER – JDBC

LEKARSKA ORDINACIJA

Posmatrani sistem je jedna lekarska ordinacija koja opslužuje pacijente koji se vode u evidenciji i u kojoj svaki od određenog broja lekara može da pruža sve usluge. Ordinacija radi tako što pacijenti prvo zakazuju termin u fiksnom trajanju od jednog sata. Tom prilikom, operater u ordinaciji prvo evidentira pacijenta ako već nije u evidenciji, a zatim određuje prvi raspoloživi termin, pri čemu tada nije poznato koji će ga lekar opslužiti. Svaki zakazani termin može biti ostvaren ili neostvaren (ako pacijent ne dođe). Prilikom svakog ostvarenog termina određeni lekar prvo utvrđuje koje će sve usluge obaviti a zatim pruža te usluge u vidu tretmana. Na kraju ostvarenog termina a po okončanju svih tretmana pacijentu se ispostavlja račun koji se plaća u celini odmah. Račun sadrži stavke od kojih svaka odgovara jednom tretmanu.

Šema relacione baze podataka je:

USLUGA (IDUsl,Naziv,Cena)
LEKAR (IDLek,PrezimeIme)
PACIJENT (IDPac,PrezimeIme,KontaktPodaci)
TERMIN (IDTer,Datum,Vreme,Ostvarenost,IDPac,IDLek)
TRETMAN (IDTer,RedBr,IDUsl)
RACUN (IDRac,Datum,IDTer, Status)
STAVKA (IDRac,RedBr,IDUsl)

Sastaviti Java metodu koja koristeći JDBC implementira funkcionalnost ObradaTermina, koja se obavlja samo ukoliko je pacijent došao u svom zakazanom terminu. Pretpostaviti da konekcija već uspostavljena.

```
public boolean ObradaTermina(Connection konekcija, int idPac, int idLek){

	//provera ispravnosti unetih podataka

	int idTermina;

	Statement upitTermin = konekcija.createStatement();

	ResultSet rezultatTermin = upitTermin.executeQuery("
	SELECT IDTer
	FROM TERMIN
	WHERE IDPac =” + idPac + “AND IDLek =” + idLek + “AND Ostvarenost = ‘N’");

	while(rezultatTermin.next()){
	     idTermina = rezultatTermin.getInt(1);
	}
	upitTermin.close();
	rezultatTermin.close();

	//pretpostavka je da je dovoljno da postoji zakazan termin
	//mogla je i provera datuma za kada je zakazan termin

	if(idTermina == NULL) return false;

	//evidentiranje pocetka termina

	Statement osveziTermin = konekcija.createStatement();

	osveziTermin.executeUpdate("UPDATE TERMIN
	SET Ostvarenost = ‘D’ WHERE IDTer =” + idTermin);

	osveziTermin.colose();

	//utvrdjivanje potrebnih usluga

	int[] idPotrebnihUsluga = UcitajPotrebneUsluge();

	//pruzanje usluga

	int redBr = 0;

	PreparedStatement dodajTretman = konekcija.prepareStatement(“INSERT INTO
TRETMAN VALUES (?, ?, ?)”);

	for (int idUsluge : idPotrebnihUsluga){

		redBr++;

		dodajTretman.setInt(1, idTermina);
		dodajTretman.setInt(2, redBr);
		dodajTretman.setInt(1, idUsluge);

		dodajTretman.executeUpdate();

	}

	dodajTretman.close();

	//izrada racuna

	redBr = 0;
	idRacuna = 0;
```

```
        Statement upitRacun = konekcija.createStatement();

        ResultSet rezultatRacun = upitRacun.executeQuery("SELECT MAX(IDRac)+1
        FROM RACUN);

        while(rezultatRacun.next()){

             idRacuna = rezultatRacun.getInt(1);
        }

        upitRacun.close();
        rezultatRacun.close();

        Statement unosRacun = konekcija.createStatement();

        unosRacun.executeUpdate("INSERT INTO RACUN
        VALUES (“ + idRacuna +”, CURRENT_DATE,” + idTermina + ”, ‘N’)”);

        unosRacun.close();

        PreparedStatement dodajStavku = konekcija.prepareStatement(
        “INSERT INTO STAVKA VALUES (?, ?, ?)”);

        for (int idUsluge : idPotrebnihUsluga){

               redBr++;

               dodajTretman.setInt(1, idRacuna);
               dodajTretman.setInt(2, redBr);
               dodajTretman.setInt(1, idUsluge);

               dodajTretman.executeUpdate();
        }

        dodajStavku.close();

        //naplata racuna

        Statement osveziRacun = konekcija.createStatement();

        osveziRacun.executeUpdate("UPDATE RACUN
        SET Status = ‘D’ WHERE IDRac = “ + idRacuna);

        osveziRacun.close();

        return true;

}
```