Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Jakob Marušič

Primerjava performance SQL podatkovnih baz

Seminarska naloga pri predmetu Tehnologija upravljanja podatkov

Ljubljana, 2020

Uvod

V poplavi različnih sistemov za upravljanje s podatkovnimi bazami (SUPB) razvijalci pogosto izbirajo glede na dosedanje izkušnje pri delu s SUPB-ji ali glede na poslovna pravila določena s strani organizacije.

Odločitev je pogosto sprejeta s predpostavko, da večina rešitev za upravljanje s podatki omogoča primerjivo performanco. V seminarski nalogi, ki je nastala v okviru predmeta Tehnologija upravljanja podatkov, želim preveriti to predpostavko s testiranje primerjivih operacij v različnih SUPB-jih.

Poglavje 1

Način testiranja

1.1 Izbrani SUPB

Testiranje se bo izvajalo na treh prosto dostopnih podatkovnih bazah: MySQL, PostgreSQL in Micorsoft SQL Server.

Vse tri podatkovne baze se bodo izvajale istočasno na Dockerju.

1.1.1 Zagon Docker slik

1.1.2 MySql

```
docker run --name TUP-sem-mysql -p 7202:3306 -e
        MYSQL_ROOT_PASSWORD=root -d mysql:latest
```

Baza je dostopna na naslovu http://localhost:7202

1.1.3 PostgreSQL

```
docker run --name tup-sem-postgres
      -e POSTGRES_USER=root -e POSTGRES_PASSWORD=root
      -p 7200:5432 -d postgres
```

Baza je dostopna na naslovu http://localhost:7200

1.1.4 Microsoft SQL Server

```
docker run -e 'ACCEPT_EULA=Y' --name tup-sem-mssql
   -e 'SA_PASSWORD=root_ROOT' -p 7201:1433
   -d mcr.microsoft.com/mssql/server:2017-CU8-ubuntu
```

Baza je dostopna na naslovu http://localhost:7201.

1.2 Način poganjanja podatkovne baze

Podatkovna baza je bila nameščena in se je izvajala na računalniku s sledečimi specifikacijami:

Procesor	Intel Core i7-8705G 3.10GHz
Delovni pomnilnik	16 GB
Virtualni delovni pomnilnik	28,9 GB
Operacijski sistem	Windows 10 Education
Verzija operacijskega sistema	10.0.18363 Build 18363
Verzija Docker okolja[1]	19.03.5
Verzija MySQL docker slike [2]	8.0.18
Verzija PostgreSQL docker slike	12.1
Verzija MS SQL Server docker slike	2017

1.3 Način točkovanja testiranja in vrste testov

Vsak test se točkuje z določenim število točk pri čemer je maksimalno število točk določeno po ključu, kjer je glavni faktor pogostost izvajanja določene operacije realnem svetu *(primer: zdravstveni dom)*.

Testira se vse vidike operacij nad podatki podatkovne baze, katere izvajamo v okviru *CRUD (Create, Read, Update, Delete)* operacij.

Operacija	**Maksimalno število točk**	**Primer uporabe**
Vstavljanje[3] podatkov	25	migracija podatkov
Brisanje[4] podatkov	25	praznjenje tabel
Posodabljanje[5] podatkov	50	sprememba KZZ
Branje, analiza podatkov	$4 \cdot 50 = 200$	statistika

1.4 Način izvedbe testov in določitve točk

Pri vsakem testu, ki se bo izvedel večkrat, bomo po definiranem ključu določili zmagovalca, ki v dani kategoriji dobil maksimalno število točk. Naslednji dve podatkovni bazi pa bosta od maksimalnega števila točk dobila odbitek, ki je enak odstotku razlike med lastnim rezultatom in rezultatom

[1] https://docs.docker.com/docker-for-windows/install/
[2] https://hub.docker.com/_/mysql
[3] Vstavljanje velike količine podatkov
[4] Brisanje velike količine podatkov
[5] Posodabljanje velike količine podatkov

zmagovalca kategorije.

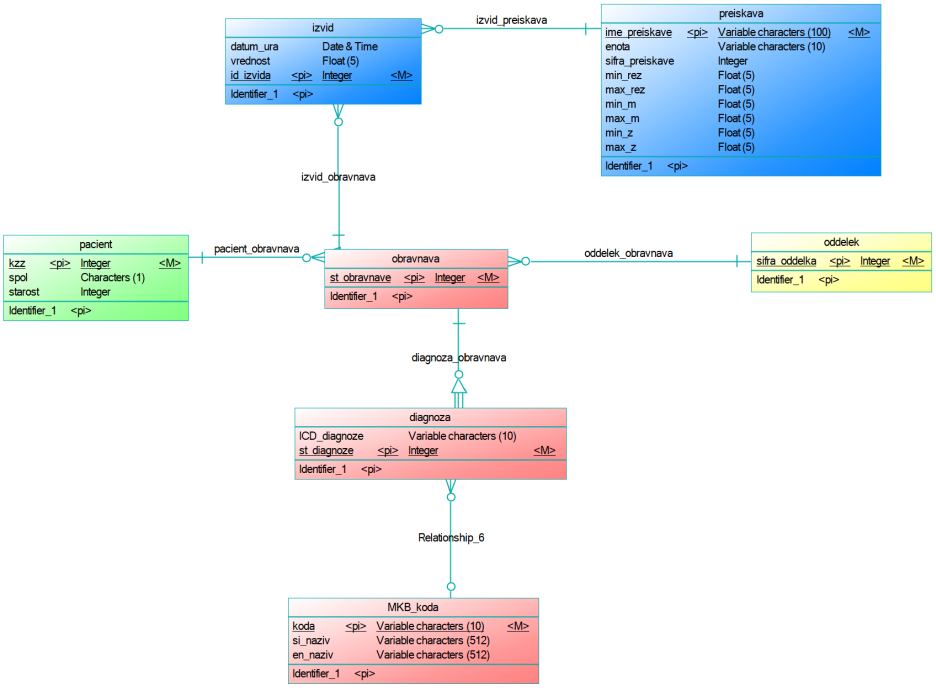
V zaključku raziskave se bodo točke seštele in tako objektivno določile performančnega zmagovalca testov. Vseeno pa so točke podeljene zgolj informativne narave, saj se izkaže, da pri izbiri pravega SUPB-ja ni pomembna zgolj performanca, temveč tudi izbira posameznika, naj si bo naročnika, upravljalca ali programerja podatkovne baze. Več o tem sledi v zaključku naloge.

Poglavje 2

Podatkovna baza - zdravstveni dom

2.1 Konceptni model

Podatkovna baza za potrebe testiranja je podatkovna baza zdravstvenega doma, ki se je uporabljala pri 3. domači nalogi. Podatkovna baza je razdeljena na 6 tabel in ima skupno xx vrstic.



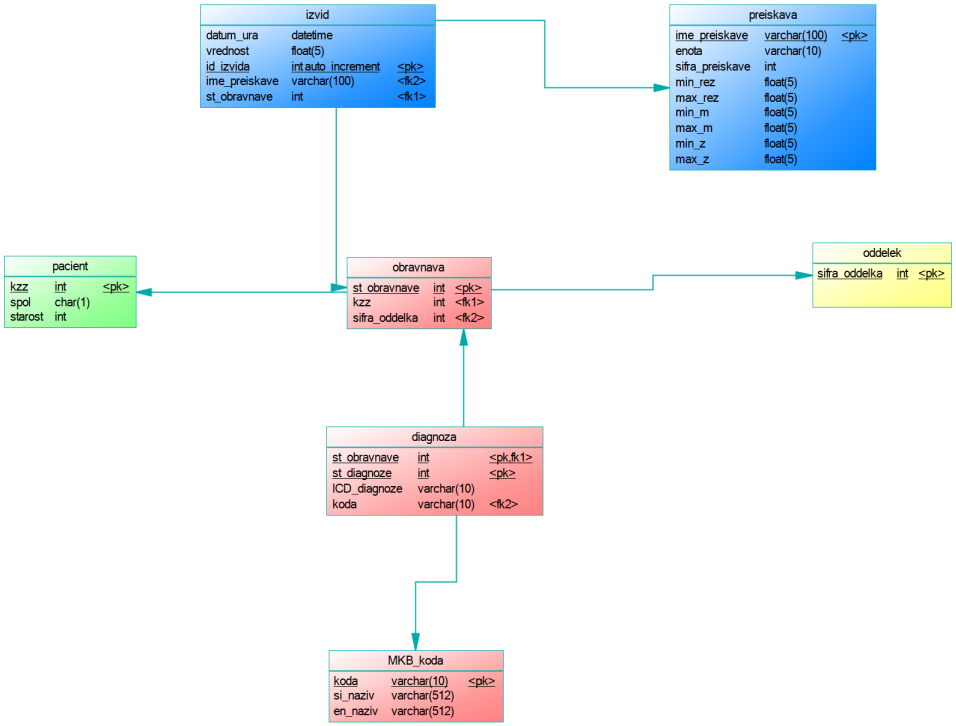
Slika 2.1: Konceptni model podatkovne baze

2.1.1 Sestava podatkovne baze

Tabela	Število zapisov v tabeli
Pacient	6 071
Obravnava	6 643
Diagnoza	34 274
Oddelek	10
Izvid	846 906
Preiskava	274

2.2 Fizični model

2.2.1 MySQL

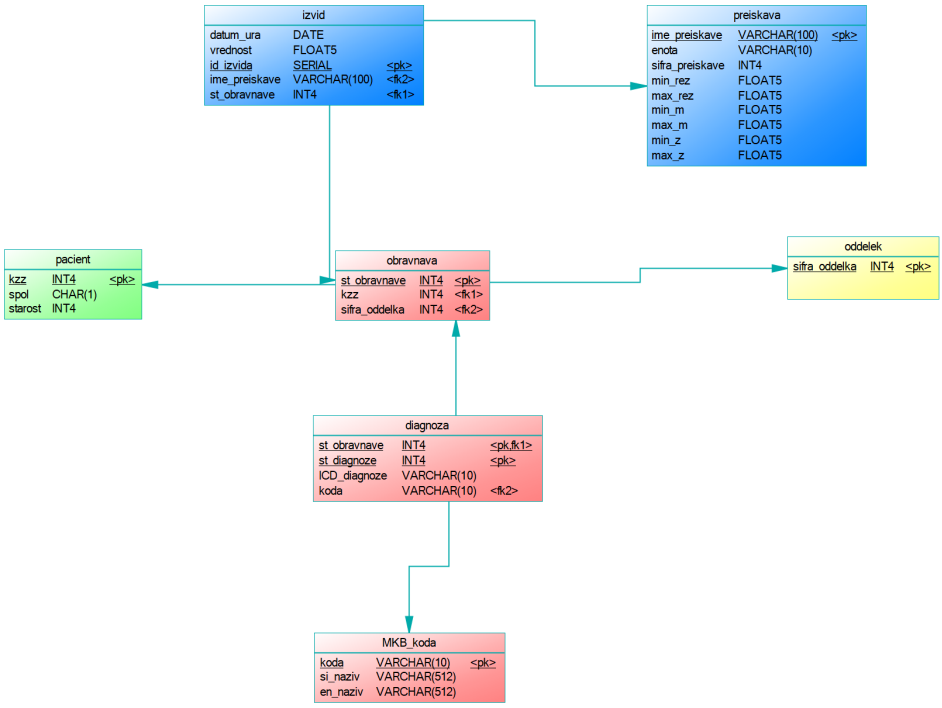


Slika 2.2: Fizični model podatkovne baze za MySQL

Skripta za ustvarjanje podatkovne baze je shranjena v datoteki *./sql-skripte/mysql-init.sql*.

2.2.2 PostgreSQL

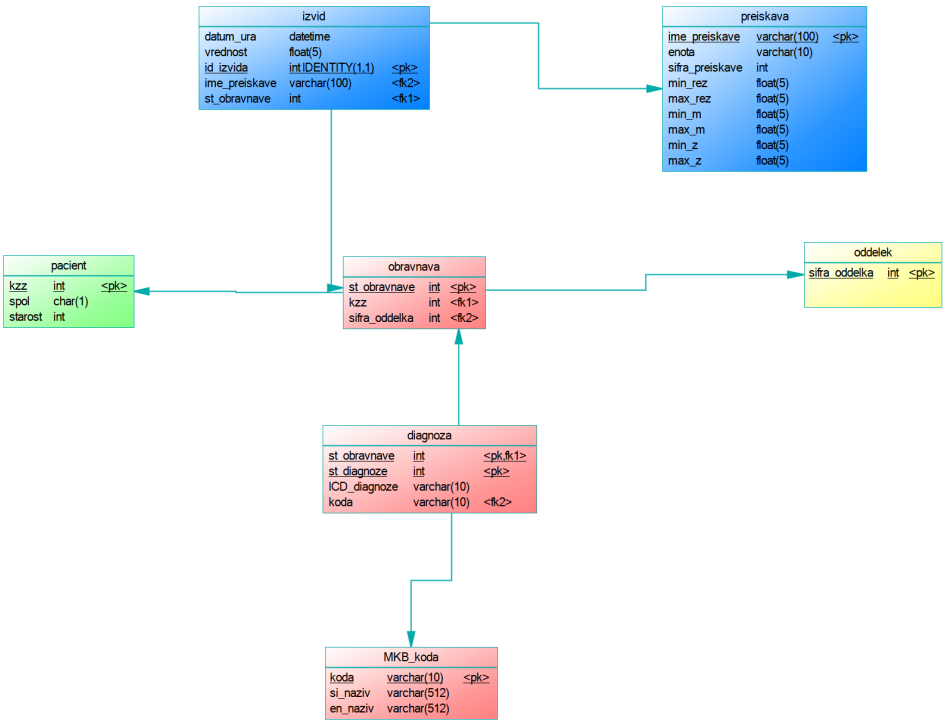
Skripta za ustvarjanje podatkovne baze je shranjena v datoteki *./sql-skripte/postgres-init.sql*.



Slika 2.3: Fizični model podatkovne baze za PostgreSQL

2.2.3 Microsoft SQL Server

Skripta za ustvarjanje podatkovne baze je shranjena v datoteki *./sql-skripte/mssql-init.sql*.



Slika 2.4: Fizični model podatkovne baze za MS SQL Server

Poglavje 3

Testiranje

3.1 Kreiranje podatkovne baze

3.2 Vstavljanje (velike količine) zapisov v podatkovno bazo

V podatkovno bazo se je vstavilo skupaj xxx zapisov. Vstavljanje je potekalo prek Python skripte filanje.py po postopku:

1. Iz csv (coma seperated values) datoteke je program prebral vrstico, jo razdeli in pretvoril v pravilen zapis spremelnjivke (s pomočjo vgrajenih ali pomožnih funkcij)
2. Program shrani trenutni sistemski čas
3. Program izvede INSERT
4. Program od trenutnega sistemskega časa odšteje sistemski čas iz točke 3 in ga prišeje k števcu skupnega časa
5. Program se premakne na naslednjo vrstico csv datoteke
6. Ko program obdela vse datoteke, funkcija vrne celoten seštevek časov izvajanja

Test se je avomatsko izvedel desetkrat na vseh treh testnih bazah desetkrat v vrstnem redu: Microsoft SQL Server, MySql in PostgreSQL. Po vrsti je program polnil tabele:

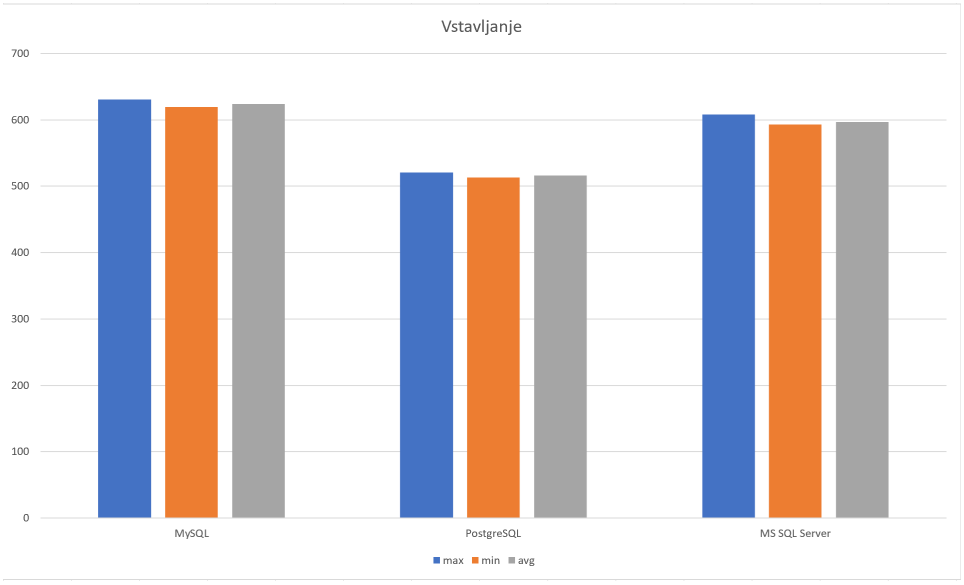
1. Pacient
2. Oddelek
3. Obravnava

4. MKB_koda
5. Diagnoza
6. Preiskava
7. Izvid

Po polnjenju vseh treh podatkovnih baz je program podatkovne baze izpraznil pri čemer je bil uporabljen ukaz *DELETE * FROM —ime tabele—* .

Pri obdelavi rezultatov se najboljši in najslabši čas nista upoštevala, iz ostalih pa se je izračunalo povprečje.

SUPB	**Najboljši čas** [s]	**Najslabši čas** [s]	**Povprečje** [s]
PostgreSQL	512,716	520,791	515,921
Microsoft SQL Server	592,745	608,040	596,453
MySQL	619,255	630,775	623,806



Slika 3.1: Primerjava "best, worst & average"vstavljanja

Iz rezultatov lahko vidimo, da je povprečje PostgreSQL-a ne le najboljše, ampak je celo za *15 %* boljši od najboljšega časa naslednjega zasledovalca, tj. Microsoft SQL Server, in za *20%* od MySQL.

V povprečju je sicer PostgreSQL potreboval približno 8 minut in pol, Microsoft SQL Server nekaj manj, MySql pa nekaj več kot 10 minut.

Ob tem gre poudariti, da se je pri vseh uporabljal *običajen INSERT* in ne kakšna posebna oblika vstavljanja velike količine podatkov (npr. *BULK INSERT* pri Microsoft SQL Server).

Iz zgoraj navedenih rezultatov se, kot je zapisano v sekciji 1.4.1, določi točke po naslednjem ključu:

SUPB	Rezultat testiranja	Odstotek točk [%]	Število točk
PostgreSQL	515,921	100	25
Micorsoft SQL Server	596,453	86,5	21,6
MySql	623,806	82,7	20,7

3.3 Brisanje (velike količine) podatkov iz podatkovne baze

Brisanje podatkov je potekalo avtomatsko z uporabo Python skripte *filanje.py*. Testiranje se je izvedlo destkrat.

Testiranje je potekalo po naslednjem postopku:

1. Program za izvede ukaz s katerim pridobi imena vseh tabel v podatkovni bazi
2. Program za vsako izmed tabel izvede:
3. Program shrani trenutni sistemski čas
4. Program izvede ukaz *DELETE FROM —ime-tabele—*
5. Program od trenutnega sistemskega časa odšteje shranjen čas
6. Program vrne seštevek vsote časov izvajanja funkcije brisanja podatkov

Rezultati izračunani iz povprečja so naslednji:

SUPB	Testiranje [ms]	Odstotek točk [%]	Število točk
Microsoft SQL Server	1,087	100	25
PostgreSQL	1,490	72,9	18,2
MySQL	5,373	20,2	5,1

Iz zgornjih rezultatov lahko sklepamo, da je brisanje podatkov iz podatkovne baze veliko hitrejši postopek od vstavljanja. Ravno zaradi tega so relativne razlike med testiranimi SUPB-ji izjemno velike v primerjavi z razlikami pri vstavljanju, čeprav so absolutne razlike (predvsem med MS SQL Server in PostgreSQL) zanemarljive.

3.4 Branje podatov iz podatkovne baze

3.4.1 Povezovanje tabel in primerjava med *WHERE* in *INNER JOIN*

SQL poizvedba Poizvedba za vsakega izmed pacientov prikaže pacientovo številko zdravstvenega zavarovanja, spol, številko obravnave ter kakšna vrsta preiskave je bila opravljena in njeno vrednost. V testiranju sta bili uporabljeni dve različni poizvedbi ena z uporabo *INNER JOIN* (poizvedba 1) in druga z uporabo *WHERE* stavka (poizvedba 2).

```
Poizvedba 1:
   select
      p.kzz,
      p.spol,
      o.st_obravnave,
      i.ime_preiskave,
      i.vrednost
   from pacient p
   inner join obravnava o on o.kzz = p.kzz
   inner join izvid i on i.st_obravnave=o.st_obravnave
   order by p.kzz, o.st_obravnave
```

```
Poizvedba 2:
   select
       p.kzz,
       p.spol,
       o.st_obravnave,
       i.ime_preiskave,
       i.vrednost
   from
        pacient p,
        obravnava o,
```

```
        izvid i
    where
        p.kzz = o.kzz and
        o.st_obravnave = i.st_obravnave
    order by p.kzz, o.st_obravnave
```

Rezultati poizvedbe 1 (*INNER JOIN*) Test se je izvedel desetkrat, avtomatsko z uporabo Python skripte *poizvedbe.py* za vse tri podatkovne baze.

SUPB	Najboljši [ms]	Najslabši [ms]	Povprečen rezultat [ms]
Microsoft SQL Server	361	389	372
MySql	710	842	782
PostgreSQL	1052	1352	1135

Primerjava hitrsoti izvedbe SQL stavka z uporabo *INNER JOIN*

Rezultati poizvedbe 1 (*WHERE*)

SUPB	Najboljši [ms]	Najslabši [ms]	Povprečen rezultat [ms]
Microsoft SQL Server	367	420	393
MySql	795	1000	874
PostgreSQL	1056	1313	1085

Primerjava hitrsoti izvedbe SQL stavka z uporabo *WHERE*

Microsoft SQL Server je tako pri poizvedbi 1 kot pri poizvedbi 2 za (vsaj) 41% hitrejši od MySql ter vsaj 60% od PostgreSQL, kateri se izkaže za veliko počasnejšega od konkurence.

Razlike med hitrostjo poizvedb pri uporabi *WHERE* in *INNER JOIN* so pri vseh treh bazah zanemarljive, vendar vseeno opazne.

SUPB	*INNER JOIN* [ms]	*WHERE* [ms]	Razlika [%]
Microsoft SQL Server	392	372	-4,5
MySql	781	874	10,6
PostgreSQL	1084	1135	7,7

Prikaz razlike v povprečni hitrosti izvedbe SQL poizvedbe z uporabo *INNER JOIN* in *WHERE* stavka.

Relativne razlike v hitrosti so najmanj razvidne pri Microsoft SQL Server, kjer se zdi, da je *WHERE* poizvedba bolje optimizirana, vendar so razlike premajhne, da bi lahko kaj takega trdili z dovolj veliko verjetnostjo. Pri PostgreSQL in predvsem pri MySql se zdi, da je *INNER JOIN* boljša izbira.

Določitev točk Čeprav se poizvedba deli na dve sintaktično različni, je razultat obeh poizvedb identičen. Kot takega ga torej štejemo kot eno postavko izračuna točk, pri čemer se upošteva boljši rezultat obeh poizvedb.

SUPB	Testiranje	Odstotek točk [%]	Število točk
Microsoft SQL Server	372	100	50
MySql	781	50,3	23,1
PostgreSQL	1085	36,2	17,2

3.4.2 Uporaba funkcije *COUNT* v povezanih tabelah

Test se je izvedel desetkrat, avtomatsko z uporabo Python skripte *poizvedbe.py* za vse tri podatkovne baze.

SQL poizvedba Poizvedba izpiše ime preiskave in število izvidov, katerih vrednost je višja od priporočene vrednosti glede na spol. Izpis je urejen padajoče po številu prekoračenih izvidov.

```
select
    i.ime_preiskave,
    count(i.ime_preiskave)
from
    izvid i
inner join obravnava o on
        i.st_obravnave = o.st_obravnave
inner join pacient p
        on p.kzz = o.kzz
inner join preiskava p2
        on i.ime_preiskave = p2.ime_preiskave
where
    (p.spol = 'M' and p2.max_m <= i.vrednost
        and p2.max_m is not null)
    or
    (p.spol = 'Z' and p2.max_z <= i.vrednost
        and p2.max_z is not null)
group by i.ime_preiskave
order by count(i.ime_preiskave) desc
```

SUPB	Najboljši [ms]	Najslabši [ms]	Povprečen rezultat [ms]
PostgreSQL	378	504	413
Microsoft SQL Server	606	646	633
MySql	1653	1778	1704

SUPB	**Testiranje**	**Odstotek točk [%]**	**Število točk**
PostgreSQL	413	100	50
Microsoft SQL Server	633	65,2	32,6
MySql	1704	24,2	12,1