Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Nedim Husaković

Primerjava in analiza učinkovitosti podatkovnih baz DB2 in MySQL

diplomsko delo

visokošolski študijski program
prve stopnje
računalništvo in informatika

Mentor: viš. pred. dr. Aljaž Zrnec

Ljubljana, 2017

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Tematika naloge:

Danes obstaja veliko relacijskih podatkovnih baz. IBM DB2 in MySQL sta zaradi prilagodljivosti uporabljeni v številnih velikih sistemih. V okviru diplomske naloge predstavite in analizirajte podatkovni bazi DB2 in MySQL, kjer izpostavite prednosti in slabosti obeh in ključne razlike. V okviru praktičnega dela izvedite performančno analizo obeh baz, kjer za shranjevanje podatkov uporabite klasičen disk in disk SSD. Glede na rezultate analize podajte priporočila glede uporabe obeh baz.

Iskreno se zahvaljujem viš. pred. dr. Aljažu Zrnecu za pomoč, potrpežljivost, izjemno odzivnost in usmerjanje pri pisanju diplomske naloge. Rad bi se zahvalil tudi moji družini za neprestano podporo in spodbudo tekom celotnega študija. Iskrena zahvala gre tudi Evi Andolšek, ki je poskrbela za vse moje slovnične napake.

Kazalo

Seznam uporabljenih kratic

kratica	angleško	slovensko
DDL	Jezik za definiranje podatkov	Data Definition Language
DML	Data Manipulation Language	Jezik za rokovanje s podatki
CODASYL	onference on Data Systems Languages	Konferenca o jezikih podatkovnih sistemov
CPU	Central processing unit	Centralna procesorska enota
IMS	Information Management System	Sistem za upravljanje informacij
DBMS	Database management system	Sistem za upravljanje podatkovnih baz
RDBMS	Relational database management system	Sistem za upravljanje podatkovnih baz
SQL	Structured Query Language	Strukturirani poizvedovalni jezik
XML	Extensible Markup Language	Razširljivi označevalni jezik
ANSI	American National Standards Institute	Ameriški nacionalni inštitut za standarde
ISO	International Organization for Standardization	Mednarodna organizacija za standardizacijo
DCL	Data Control Language	Jezik za nadzor nad podatki
PHP	Hypertext Preprocessor	Strežniški skriptni jezik
SMP	Symmetric multiprocessing	Simetrično multiprocesiranje
JSON	JavaScript Object Notation	Opis JavaScript objekta
ISAM	Indexed Sequential Access Method	Metoda zaporedne indeksirane dostopnosti
API	Application program interface	Vmesnik aplikacijskega programa
EEE	Extended Enterprise Edition	Razširjena izdaja
MPP	Massively parallel processing	Masovna vzporedna obdelava

Povzetek

Naslov: Primerjava in analiza učinkovitosti podatkovnih baz DB2 in MySQL

Avtor: Nedim Husaković

Kljub temu, da obstaja veliko drugih različic podatkovnih baz, ki ne uporabljajo relacijskega podatkovnega modela, pa je le-ta trenutno najbolj uporabljen na svetu. Relacijske podatkovne baze uporabljajo strukturirani poizvedovalni jezik SQL in zagotavljajo ACID lastnosti transakcij. Cilj diplomske naloge je predstaviti relacijski podatkovni model in analizirati dve relacijski podatkovni bazi, in sicer DB2 ter MySQL, ki sta zaradi svoje varnosti in fleksibilnosti med najbolj uporabljenimi podatkovnimi bazami na svetu. Podatkovni bazi predstavimo tako s teoretičnega, kot s praktičnega vidika. V okviru teoretičnega dela na osnovi znanstvene literature predstavimo pojem podatkovne baze in opišemo njene značilnosti, zgodovino ter lastnosti. Glavni namen je predstaviti prednosti ter pomanjkljivosti relacijskih podatkovnih baz. Poleg opisa sistema je podana tudi primerjava med dvema popularnima podatkovnima bazama MySQL in DB2. V okviru praktičnega dela pa na osnovi testnih scenarijev predstavimo performančni analizi MySQL in DB2 na klasičnem ter na SSD disku. Na koncu, v zaključku sledijo sklepne ugotovitve in usmeritve glede izbora najprimernejše relacijske podatkovne baze za določen problem.

Ključne besede: relacijska podatkovna baza, ACID, SQL, MySQL, DB2.

Abstract

Title: Comparison and performance analysis of DB2 and MySQL databases

Author: Nedim Husaković

Despite the fact that there are a lot of other versions of databases that do not use relation data model, this one is the most used around the world. Relation databases use SQL, a structured query language, and provide ACID transaction properties. The goal of this diploma thesis is to present the relation database model and analyse two relation databases, DB2 and MySQL that are due to their security and flexibility among the most used in the world. The databases are presented from both theoretical and practical aspect. In the theoretical part, the term database, which is based on scientific literature and its features, history and properties are presented. The main goal is to present the advantages and disadvantages of relation databases. In addition to the description of the system, is a comparison between two popular databases, MySQL and DB2. In the practical portion, the performance of MySQL and DB2 is compared based on test scenarios on a HDD and SSD drives. The conclusion includes the findings and guides to choose the most suitable relation database for a specific problem.

Keywords: relational database, ACID, SQL, MySQL, DB2.

Poglavje 1

Uvod

Podjetje v katerem delam je eden izmed razlogov za izdelavo tega diplomskega dela. Podjetje podatke shranjuje v relacijski podatkovni bazi IBM DB2. Vodstvo podjetja je začelo razmišljati, da bi začeli uporabljati drugo podatkovno bazo, zato je na dan prišla ideja o testiranju IBM DB2-a z njegovim velikim tekmecem Oracle MySQL. Oba relacijska sistema imata svojo odprtokodno različico, ki je podprta z licenco GPL. Odprtokodna različica se razlikuje od plačljive različice v funkcionalnostih in dodatnih orodjih. Odločili smo se raziskati, analizirati in testirati hitrost obeh podatkovnih baz ter jih predstaviti v okviru diplomske naloge.

Odločitev je sprva vodila do prebiranja člankov ter dokumentacije omenjenih relacijskih podatkovnih baz. Že na začetku se je zdela MySQL zelo zanimiva, saj je bila celotna podatkovna baza z vsemi orodji in funkcionalnostmi do leta 2010 zastonj. Ko je podjetje Oracle kupilo sistem, je to vodilo do prekinitve podpore odprtokodnim orodjem in funkcionalnostim. Seveda pa obstajajo tudi pozitivne posledice prodaje MySQL-a podjetju Oracle, kot so na primer izboljšana varnost podatkovne baze, hitrost in redne posodobitve. Na drugi strani pa imamo DB2 relacijski sistem, ki je leta 2006 sklenil izdati svojo prvo odprtokodno različico podatkovne baze DB2 Express-C. S to izdajo so želeli pridobiti zaupanje manjših organizacij, ki so jim ponudili relacijsko po-

datkovno bazo s številnimi zmogljivimi funkcijami, ki se nahajajo v dražjih različicah DB2 sistema. Organizacije oziroma podjetja se bodo zaradi primerjave in analize lažje odločili za določeno podatkovno bazo.

Diplomska naloga je sestavljena iz šestih delov. Prvi del je namenjen predstavitvi pojma "podatkovna baza"in zgodovini o podatkovnih bazah. V nadaljevanju bomo opisali značilnosti relacijskih podatkovnih baz in iskali prednosti relacijskih pred drugimi bazami. Proučili bomo tudi lastnosti transakcij ACID in SQL poizvedovalnega jezika.

V drugem delu se bomo podrobneje seznanili s podatkovnima bazama MySQL in DB2, za kateri bomo predstavili zgodovino in funkcionalnosti. Večji razmislek bomo namenili prednostim ter razlikam med plačljivimi in odprtokodnimi različicami pri MySQL in DB2 relacijskima sistemoma.

V tretjem poglavju z naslovom HDD in SSD diski bomo predstavili prednosti ter slabosti klasičnih in SSD diskov. Spoznali bomo tudi delovanje klasičnih in SSD diskov.

Četrti del je namenjen testiranju podatkovnih baz. V okviru tega bomo predstavili okolja za upravljanje podatkovnih baz MySQL in DB2, ki jih priporočata podjetji Oracle in IBM. Podrobneje bomo opisali okolje DBVisuliser, v katerem je bilo opravljeno celotno testiranje. Testiranje je potekalo na petih različnih scenarijih, in sicer: vstavljanje podatkov v prazno tabelo, vstavljanje podatkov v tabelo s podatki, brisanje podatkov, branje podatkov in branje podatkov z uporabo indeksov. Za vsak scenarij smo izdelali preglednico in graf na podlagi rezultatov, ki nam ga je podalo testiranje.

Sledi še zadnje poglavje oziroma zaključek diplomske naloge, v katerem bomo združili povzetek ugotovitev na osnovi testiranja MySQL in DB2 ter premislek o prihodnosti relacijskih podatkovnih baz.

Poglavje 2

Podatkovna baza

2.1 Kaj je podatkovna baza

Podatkovna baza je zbirka medsebojno logično povezanih podatkov, ki zadovoljujejo informacijske potrebe organizacije in njenih poslovnih procesov [1]. Podatki so zapisana dejstva, ki so shranjena na nekem računalniškem sistemu, ki se jim lahko pripiše pomen [2]. Vsaka zbirka podatkov ni podatkovna baza, saj mora le-ta ustrezati določenim pogojem kakovosti, kot so prilagodljivost, natančnost, uporabnost. Te pogoje dosežemo s sistemi za upravljanje podatkovnih baz (SUPB) ali angleško DBMS. SUPB je skupek programske opreme, ki omogoča kreiranje, vzdrževanje in nadzor nad podatki v podatkovni bazi. Kreiranje podatkovnih struktur je omogočeno preko DDL (Data Definition Language), vzdrževanje podatkov pa z ukazi Create, Insert, Update, Delete preko DML (Data Manipulation Language).
Dostop do podatkov je lahko centraliziran ali porazdeljen. Centralizirane podatkovne baze omogočajo lažji nadzor in upravljanje podatkov. Te podatkovne baze so bolj ranljive, saj so odvisne od enega centralnega sistema. Pri porazdeljenih podatkovnih bazah pa je zbirka podatkov logično povezana in fizično porazdeljena po vozliščih računalniškega sistema.

Za lažje razumevanje in opisovanje podatkovne baze si ustvarimo model, na osnovi katerega podatkovno bazo izdelamo [3]. Podatkovni model je abstraktna, a vendar razumljiva predstavitev potrebnih podatkov celotne organizacije oziroma njenega dela, kar pomeni predstavitev same organizacije oziroma njenega dela z vidika potrebnih podatkov. Danes obstaja in uporabljamo več podatkovnih modelov. To so:

- relacijski,
- objektno-relacijski in
- objektni podatkovni model.

Poznamo še dva podatkovna modela, hierarhičnega in mrežnega, ki pa zaradi svoje zastarelosti iz prakse izginjata. Najbolj raširjen podatkovni model izmed zgoraj naštetih je relacijski model.

2.2 Razvoj podatkovnih baz

Shranjevanje podatkov sega dolgo nazaj v preteklost. Podatkovne baze smo začeli uporabljati zaradi potrebe po hitrem dostopu do podatkov. Baze so sprva uporabljale vladne službe, knjižnice in bolnice. V 60. letih prejšnjega stoletja so izvajali projekt Apollo, katerega cilj je bil pristajanje človeka na luni. V tem času ni bilo nobenega sistema, ki bi obdeloval in upravljal s tako veliko količino podatkov, zato so pri severnoameriškem letalstvu (ang.: North American Aviation) razvili programsko opremo znano kot GUAM (ang.: Generalized Update Access Method). Struktura te programske opreme je ustrezala drevesu obrnjenemu na glavo, oziroma je znana kot hierarhična struktura. V teh letih je uporaba računalnikov postala stroškovno dostopnejša za zasebna podjetja in je to eden izmed razlogov, da so se začele podatkovne baze masovno uporabljati. V tem desetletju sta se največ uporabljala omrežni model CODASYL (ang.: Conference/Committee on Data Systems Languages) in hierarhični model IBM IMS (ang.: Information Management

System), ki je predstavljal osnovo za hierarhični podatkovni model. Največji uspeh v poznih šestdesetih letih je doživel sistem SABRE, ki ga je IBM razvil za upravljanje rezervacij v podjetju American Airlines.

Leta 1970 je Edvard Codd objavil pomemben dokument, s katerim je predlagal uporabo relacijskega podatkovnega modela. Njegove ideje so spremenile način razmišljanja pri ljudeh in postavile standarde za razvoj relacijskih podatkovnih baz.

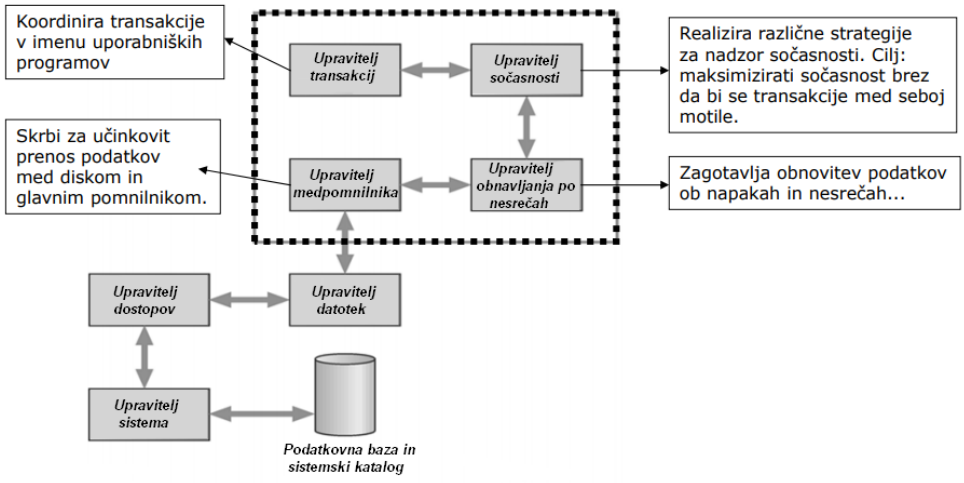
Največja rast industrije podatkovnih baz je bila okoli leta 1990, zaradi pojava interneta. Internet je uporabo podatkovnih baz privedel do eksponentne rasti zaradi povečanja rasti podatkov. Internetna industrija je v začetku leta 2000 doživela padec, ampak je povpraševanje po podatkovnih bazah še vedno naraščalo. V teh letih so se razvile še nerelacijske podatkovne baze tipa NoSQL, ki so prirejene posebnim potrebam delovanja, ter baze, ki so združevale prednosti tako relacijskih kot NoSQL, ti. NewSQL podatkovne baze.

Trenutno se na svetu najbolj uporabljajo prav relacijske podatkovne baze [4]. Štiri vodilne relacijske podatkovne baze na svetu so Oracle Database, IBM DB2, Microsoft SQL Server in Oracle MySQL [5].

2.3 Sistem za upravljanje podatkovnih baz (SUPB)

Sistem za upravljanje podatkovnih baz je programska oprema, ki omogoča definiranje, kreiranje in vzdrževanje podatkovne baze ter zagotavlja hkraten in nadzorovan dostop do podatkov več uporabnikom. Ta programska oprema je povezana z aplikacijskimi programi uporabnikov in samo podatkovno bazo. Aplikacijski program pa je računalniški program, ki komunicira s podatkovno bazo z ustrezno zahtevo (običajno SQL stavek), ki jo posreduje sistemu za upravljanje s podatkovno bazo.

Sistem za upravljanje podatkovnih baz je razdeljen na več komponent programske opreme, od katerih ima vsaka svojo nalogo. Slika 2.2 nam prikazuje glavne komponente programske opreme SUPB-ja. V črtkanem pravkokotniku so predstavljene komponente za obvladovanje transakcij, nadzor sočasnosti in obnovitev podatkov.



Slika 2.1: Moduli SUPB [6].

Upravitelj datotek (ang.: File Manager) upravlja z osnovnimi datotekami za shranjevanje in skrbi za dodelitev prostora za shranjevanje na disku.

Upravitelj transakcij (ang.: Transaction Manager) opravlja zahtevano obdelavo operacij, ki jih dobiva iz transakcij.

Upravitelj sočasnosti (ang.: Scheduler) je komponenta, ki je odgovorna za zagotavljanje sočasnega izvajanja operacij. Nadzira vrstni red izvajanja operacij znotraj transakcij.

Upravitelj obnavljanja po nesrečah (ang.: Recovery manager) zago-

tavlja, da v primeru napak podatkovna baza ostane v konsistentnem stanju.

Upravitelj z medpomnilnikom (ang.: Buffer manager) je komponenta, ki je odgovorna za prenos podatkov med glavnim in sekundarnim pomnilnikom. Od upravitelja datotek prejema bloke podatkov iz diska in izbira stran v glavnem pomnilniku, v katero se bodo podatki zapisali.

Upravitelj dostopov je komponenta, ki nadzorjuje dostope in celovitost podatkov.

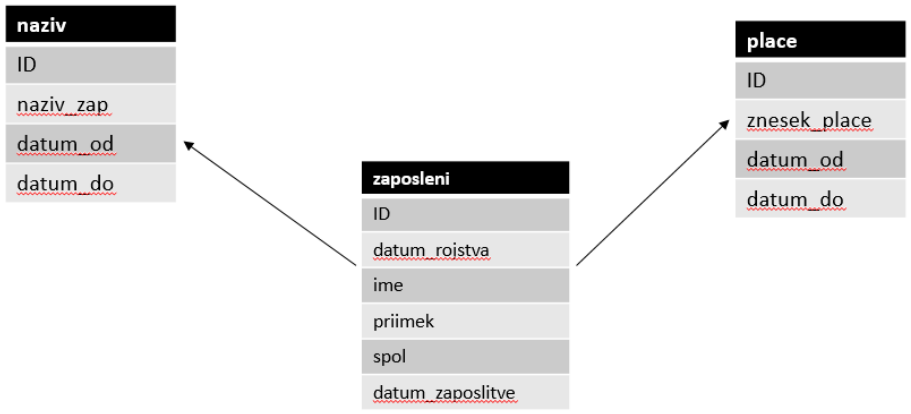
Upravitelj sistemov je komponenta, ki upravlja s podatki iz podatkovne baze.
Na svetu je trenutno največ relacijskih podatkovnih baz, saj le-te zagotavljajo več prilagodljivosti pri pisanju poizvedb. Ključna razlika od ostalih SUPB je, da relacijski sistem za upravljanje podatkovnih baz shranjuje podatke v obliki tabele ter, da so podatki logično in fizično neodvisni.

2.4 Relacijska podatkovna baza

Relacijska podatkovna baza se pojavi leta 1970, ko je razvijalec IBM-a, E. F. Codd, objavil dokument "A Relational Model of Data for Large Shared Data Banks"[7], v katerem je predstavil relacijski podatkovni model. V tem dokumentu je definiral, kaj »relacijska« sploh pomeni in na podlagi tega dokumenta so se razvile relacijske podatkovne baze. Relacijska podatkovna baza temelji na relacijskem modelu in je predstavljena z množico relacij, kjer je vsaka relacija tabela z vrsticami in stolpci. Kasneje je Codd objavil nadaljevalni dokument, v katerem navaja dvanajst pravil, ki jim morajo slediti vse podatkovne baze. Mnoge sodobne baze ne upoštevajo vseh dvanajstih pravil, vseeno pa se štejejo kot relacijske podatkovne baze. Programska oprema, ki omogoča izvajanje operacij nad podatki in definiranje same podatkovne baze v relacijskih podatkovnih bazah se imenuje sistem za upravljanje rela-

cijskih podatkovnih baz (ang.: RDBMS, Relational Database Management Systems).

Skoraj vse relacijske podatkovne baze uporabljajo strukturiran poizvedovalni jezik SQL (Structured Query Language), čeprav je bil ta razvit po predstavitvi relacijskega modela in ni zapisan v Coddovih pravilih. Ta jezik se je razvil v osemdesetih letih in je namenjen poizvedovanju po podatkih v relacijskih podatkovnih bazah.
Vsaka tabela v relacijski podatkovni bazi ima enolično ime. Tabele so predstavljene s stoplci oziroma atributi in vrsticami oziroma zapisi. Vsak atribut v tabeli ima enolično ime in svojo domeno. Atribut omejimo z domeno, ta pa predstavlja zalogo vrednosti, ki jih lahko zavzame atribut (celo število, realno število, niz...). Domena določa tudi celovitost podatkov (npr. obvezen vnos, točno 5 znakov itd.).



ID	datum_rojstva	Ime	priimek	spol	datum_zaposlitve
1	1973-05-20	Jaka	Pintar	M	1993-04-10
2	1981-10-01	Luka	Novak	M	1993-05-20
3	1975-12-03	Matic	Kralj	M	1993-05-20
4	1983-06-20	Gregor	Praznik	M	1993-06-01

Slika 2.2: Relacijski podatkovni model in tabela.

Slika 2.3 nam prikazuje tipičen primer relacijskega modela, ki prikazuje podatke o zaposlenih, njihovih nazivih in plačah. Na sliki je tudi tabela zaposleni, ki ima štiri zapise.

Vsaka vrstica v tabeli je enolično določena s primarnim ključem, ki določa zapis v tabeli. Vrednost primarnega ključa ne sme biti prazna (null) ali podvojena. Vsaka tabela ima lahko le en primarni ključ. Poleg primarnega ključa v relacijskih podatkovnih bazah poznamo še sekundarni ključ, tuji ključ, super ključ in speti ključ [8].

Sekundarni ključ olajša ali pospeši dostop do posameznega pojava entitete. Pri tem pogoj enoličnost tudi velja.

Tuji ključ igra posebno vlogo atributa. Gre za atribut pri posamezni relaciji, s pomočjo katerega vzpostavljamo povezave z drugimi primerki tipov relacij. Tabele so med seboj povezane z uporabo tujega ključa. Vrednost je lahko prazna in podatki v njem se lahko podvajajo.

Super ključ je nadmnožica ključa. To pomeni, da vsebuje ključ in dodaten oziroma dodatne atribute.

Sestavljen ključ je sestavljen iz več atributov.

Povezanost tabel med seboj olajša uporabnikom iskanje podatkov. Zagotavljanje celovitosti podatkov v povezanih tabelah nam omogoča referenčna integriteta. Ta zahteva, da lahko tuji ključ v podrejeni tabeli zavzame samo tiste vrednosti, ki jih ima primarni ključ v nadrejeni tabeli.
Slabost relacijskih podatkovnih baz je, da se lahko v njih pojavi podvajanje podatkov ali redudanca. Redundanca pomeni, da so isti podatki shranjeni večkrat. To lahko odpravimo z normalizacijo. Normalizacija je proces odkrivanja in odstranjevanja podvojenih podatkov v podatkovni bazi. Normali-

zirana podatkovna baza ne vsebuje redundančnih podatkov, zaradi česar jo je enostavneje vzdrževati in nadgrajevati. Obratni proces se imenuje denormalizacija, ki se lahko uporabi v primerih, ko se zahteva večja učinkovitost podatkovne baze.

Velika prednost, ki jo omogoča relacijski sistem za upravljanje s podatkovno bazo je možnost sočasnega dostopa uporabnikov do podatkov. Zaklepanje podatkov in upravljanje transakcij nam zagotavljata, da ne pride do trčenja med uporabnikoma, ki hkrati posodabljata podatke in, da uporabnik ne dostopa do delno posodobljenih zapisov. Vgrajene avtorizacijske funkcije omogočajo skrbniku podatkovne baze omejiti dostop in dodeljevati privilegije uporabnikom na podlagi nalog, ki jih opravljajo. Dovoljenje za dostopanje je na primer možno definirati na podlagi IP odjemalca. Skrbniki relacijskih podatkovnih baz imajo na voljo veliko orodij za preprosto vzdrževanje, testiranje in varnostno kopiranje.

Naj omenimo še, da relacijski model ne definira najučinkovitejše podatkovne strukture, vendar pa predstavlja zelo dober kompromis z vidika kompleksnosti sistema in učinkovitosti. Vgrajene funkcije in optimizacije povečajo zmogljivost in omogočajo aplikacijam, da dovolj hitro dobijo potrebne podatke iz podatkovne baze. Nenehne izboljšave strojne opreme, kot so večja hitrost procesorja, zmanjševanje stroškov pomnilnika in shranjevanje, omogočajo sistemskim operaterjem, da zgradijo učinkovite relacijske podatkovne sisteme.

2.5 Poizvedovalni jezik SQL

SQL (Structured Query Language) je do sedaj edini standardiziran (ANSI/ISO) poizvedovalni jezik za izvajanje različnih operacij v relacijskih podatkovnih bazah. Nastal je okoli leta 1970, sprva pa so ga uporabljali administratorji in razvijalci podatkovnih baz. Za strukturiran poizvedovalni jezik (SQL) lahko rečemo, da omogoča dostop do podatkovnih baz s programskimi stavki, ki

posnemajo naravni jezik[9]. Glavne kategorije stavkov SQL so: DML (ang.: Data Manipulation Language), DDL (ang.: Data Definition Language), DCL (ang.: Data Control Language) in TCL (ang.: Transactional Control Language).

Data Manipulation Language vsebuje stavke SQL, ki se uporabljajo najpogosteje in so namenjeni za manipulacijo s podatki v podatkovni bazi. Štirje najpogostejši ukazi za pridobivanje podatkov so: dodajanje novih podatkov v bazo (ukaz INSERT), pridobivanje podatkov iz baze (ukaz SELECT), spreminjanje informacij že shranjenih podatkov v bazi (ukaz UPDATE) in odstranjevanje podatkov (ukaz DELETE).

Data Definition Language vsebuje ukaze, ki so namenjeni spreminjanju dejanske strukture podatkovne baze. Najpogostejši ukazi so generiranje nove tabele (ukaz CREATE TABLE), spreminjanje tabele v bazi (ukaz ALTER TABLE), brisanje tabele (ukaz DROP TABLE) in kreiranje indeksov (ukaz CREATE INDEX).

Data Control Language se uporablja za upravljanje uporabniškega dostopa do podatkovne baze. Sestavljen je iz dveh ukazov. Ukaz GRANT, ki se uporablja za dodajanje pravic za uporabnika in ukaz REVOKE, ki se uporablja za odstranjevanje obstoječih pravic.

Transactional Control Language vsebuje ukaze za nadzor in upravljanje transakcij. Sem spadajo ukazi za začetek transakcije (ukaz BEGIN), uspešen zaključek transakcije (ukaz COMMIT) in razveljavitev transakcije (ukaz ROLLBACK).

SQL je najbolj razširjen poizvedovalni jezik. Enostaven je za uporabo, saj se osnovne sintakse ni težko naučiti. Uporabnik pove, kaj želi, SUPB pa ukaze interpretira, prevede logično sklicevanje na fizično izvedbo ukaza, in ta vrne

iskane podatke [10].
SQL jezik je sestavljen iz uporabniško definiranih in rezerviranih besed. Rezervirane besede so posebne besede, s katerimi povemo, kaj želimo narediti s podatki. Te besede ne moremo uporabiti za identifikator, kot je ime spremenljivke ali funkcije. Za boljšo berljivost pišemo rezervirane besede z velikimi črkami, ostale besede pa z malimi. Izjave oziroma poizvedbe lahko trajno vplivajo na podatke v bazi. Standard SQL določa, da moramo vsak stavek zaključiti z znakom »;«, čeprav se nekateri relacijski SUPB-ji tega ne držijo, zaradi česar stavka ni potrebno zaključiti s podpičjem. Standard tudi določa, da se več presledkov v stavku ignorira, kar omogoča lažje formatiranje kode.

2.6 Lastnosti transakcij ACID

Pomembna značilnost relacijskih podatkovnih baz je sposobnost, da za transakcije zagotovijo ACID lastnosti. Lahko rečemo, da lastnosti ACID zagotavljajo, da se transakcije v podatkovni bazi zanesljivo izvedejo tudi v primeru napak. Transakcija je zaporedje operacij nad podatkovno bazo, ki se mora izvesti kot celota (atomarna enota) po principu »vse ali nič«. ACID model določa štiri lastnosti, ki jih mora imeti vsak zanesljiv sistem za upravljanje podatkovnih baz. Kratica ACID pomeni atomarnost (ang.: Atomicity), konsistentnost (ang.: Consistency), izolacijo (ang.: Isolation) in trajnost (ang.: Durability) [11]. Relacijsko podatkovno bazo, ki ne izpolnjuje enega od navedenih ciljev, ni mogoče šteti za zanesljivo podatkovno bazo.

Atomarnost določa, da morajo spremembe slediti pravilu "vse ali nič". Če samo en del transakcije ne uspe, potem je potrebno celotno transakcijo razveljaviti.

Konsistentnost določa, da so le veljavni podatki zapisani v podatkovni bazi. Če je iz nekega razloga izvršena transakcija, ki krši pravila konsistentnosti, potem bo celotna transakcija povrnjena nazaj in podatkovna baza se bo po-

novno vrnila v stanje, ki je skladno s temi pravili.

Izolacija zahteva, da se pri izvajanju več transakcij hkrati vse izvajajo neodvisno ena od druge. Delni rezultati ne smejo biti vidni drugim transakcijam.

Trajnost pa je lastnost, ki zagotavlja, da vse spremembe v podatkovni bazi ostanejo trajne. Zahteva trajnosti je, da so vse spremembe zapisane na trajni pomnilnik (npr. trdi disk) preden je transakcija uspešno potrjena s strani podatkovne baze. V primeru izpada oziroma odpovedi sistema se potrjeni podatki ne izgubijo.

Za zagotavljanje atomarnosti, konsistentnosti, izolacije in trajnosti skrbi sistem za upravljanje podatkovnih baz. Upravitelj transakcij skrbi za konsistentnost in izolacijo, enota za obnavljanje podatkov po nesreči pa za atomarnost in trajnost.

Poglavje 3

MySQL in DB2

V okviru tega poglavja bomo predstavili dve relacijski podatkovni bazi, MySQL in DB2. Osredotočili se bomo na njune prednosti in slabosti ter izpostavili ključne razlike. Obe omenjeni podatkovni bazi bomo kasneje tudi uporabili v okviru testiranja učinkovitosti.

3.1 MySQL

MySQL je relacijski SUPB. Sistem je priljubljen zaradi prenosljivosti, saj deluje praktično na vseh platformah oziroma operacijskih sistemih. Pri tem sistemu imamo na voljo tako odprtokodno različico podatkovne baze kot tudi licenčno različico, ki jo je potrebno kupiti.

Sistem je bil prvotno zasnovan znotraj Švedskega podjetja MySQL AB, ki ga je leta 2008 kupilo podjetje Sun Microsystems. Slednjega je leta 2010 kupilo podjetje Oracle, ki je tako današnji lastnik sistema MySQL. Sistem deluje na način odjemalec-strežnik, na voljo pa je tudi kot vgrajena knjižica, namenjena manjšim samostojnim produktom. Odprotkodna programska oprema MySQL je zaščitena z licenco GPL (General Public Licence), ki je dostopna na povezavi: "http://www.fsf.otg/licenses/" [12]. Licenca določa, kaj lahko spreminjamo oziroma kaj lahko in česa ne smemo početi s programsko

opremo. Če nam GPL ne ustreza ali želimo uporabiti kodo MySQL v komercialne namene, moramo kupiti komercialno licenčno različico. Prednosti komercialne licenčne različice so na primer podpora proizvajalca in dodatne funkcionalnosti ter orodja, ki jih bomo predstavili v poglavju "Različice MySQL".

MySQL podatkovna baza se uporablja za različne namene, najpogosteje pa ta sistem uporabljajo spletne strani za shranjevanje podatkov. Spletne strani, ki uporabljajo MySQL nastanejo dinamično, tako da pri generiranju uporabijo podatke iz podatkovne baze. Veliko dinamičnih spletnih strani, ki uporabljajo MySQL, uporabljajo tudi skriptni jezik PHP za dostop do podatkov. PHP koda omogoča generiranje dela ali celote spletne strani iz podatkovne baze. MySQL je najbolj priljubljena podatkovna baza, ki se uporablja v navezi z jezikom PHP.

3.1.1 Zgodovina MySQL

Ime MySQL je nastalo iz kombinacije dveh besed. "My", kar je ime hčerke soustanovitelja podjetja MySQL AB Michaela Wideniusa in besede "SQL", kar je kratica za "Structured Query Language" in pomeni strukturirani poizvedovalni jezik. Kot smo že omenili, je sistem MySQL nastal na Švedskem v podjetju MySQL AB. Sistem so ustanovili David Axmark, Allan Larsson in Michael "Monty"Widenius. Z razvojem so ustanovitelji začeli leta 1994 z namenom osebne uporabe prvega nizkocenovnega sistema za upravljanje podatkovnih baz mSQL, ki so ga priključili IBM-ovemu datotečnemu sistemu ISAM[13]. Ugotovili so, da je mSQL počasen in nezanesljiv za potrebe zahtevnih uporabnikov, zato so ustvarili nov vmesnik SQL, hkrati pa obdržali API kot mSQL. Z ohranjanjem vmesnika so lahko razvijalci kodo, ki je bila napisana za mSQL, prenesli v MySQL.

Od svoje prve izdaje dalje so razvijalci namenili poseben poudarek delovanju MySQL podatkovne baze in njeni razširjenosti. Rezultat je bil zelo

optimizirana programska oprema, ki nima funkcij, ki so običajne v drugih podatkovnih bazah, kot so na primer shranjene procedure, prožilci in transakcije. Izdelek je dobil pozornost velikega števila uporabnikov, ki so bili bolj zainteresirani za hitrost in prenosljivost, ne pa za funkcionalnosti, ki jih uporabniki ne bodo nikoli uporabili v praksi. Uporabniki MySQL podatkovne baze so nekatere od najbolj znanih podjetij in organizacij na svetu, kot so Yahoo!, CNET, NASA in Google.

3.1.2 Razvoj in lastnosti MySQL

MySQL se odlikuje po hitrosti in v preteklosti ni imel nekaterih funkcij, ki so jih imeli drugi konkurenčni izdelki. Seveda pa je njegova velika priljubljenost potrdila, da za milijone uporabnikov te napredne funkcionalnosti niso kritičnega pomena. Prva različica (MySQL 3.19) je bila za javnost izdana v začetku leta 1995. Ta različica ne izpolnjuje Coddovih pravil, zato je ne moremo upoštevati za relacijski sistem za upravljanje podatkovnih baz.

Naslednja različica (MySQL 4.0) je bila izdana marca leta 2003 (preizkusna različica je bila izdana avgusta leta 2002) in šteje za velik mejnik v zgodovini razvoja SUPB. Po več letih razvoja je končni izdelek postal javno dostopen. Različica 4.0 je prinesla več funkcij, ki so bile v tem času kot standardne prisotne že v drugih konkurenčnih podatkovnih bazah. V standardno distribucijo so dodali pogon InnoDB. Pomembne nove funkcionalnosti v različici 4 so bile R-drevo, B-drevo, podpora Unicode in pripravljeni stavki (ang.: prepared statment). Dodali so nove spremembe, ki so bile potrebne za izpolnjevanje standardov SQL, kot je nespremenjena dolžina niza pri podatkovnem tipu TIMESTAMP.

Uporabniki so se dolgo časa pritoževali zaradi pomanjkanja napredne funkcionalnosti, kot so obstajanje pogledov, prožilcev, shranjenih postopkov itd. Razvijalci so dolgo razmišljali o vgradnji teh funkcionalnosti, ampak so želeli predvsem ohraniti hitrost podatkovne baze. Vse zgoraj navedene funkcional-

nosti so dodali v najbolj pomembno različico MySQL podatkovne baze 5.0. Različica 5.0 je bila uradno objavljena oktobra 2005. Od te različice naprej lahko MySQL sistemu rečemo relacijski sistem za upravljanje podatkovnih baz, saj je vključil referenčno integriteto. Referenčna integriteta je eno izmed Codd-ih pravil za relacijsko podatkovno bazo. Pomembna nova funkcionalnost je shranjen postopek. To je nabor ukazov SQL, zapisanih v bazi podatkov, ki jih je mogoče uporabiti kot funkcijo SQL. Primera shranjenih postopkov sta funkciji min() in rand(). Dodali so nove pogone ARCHIVE in FEDERATED ter izboljšali zmogljivost pogona InnoDB. Ostale nove funkcionalnosti so kazalniki, prožilci, pogledi itd. Edina pomanjkljivost različice 5.0 je slaba zmogljivost pri shranjevanju podatkov zaradi nezmožnosti uporabe več CPU jeder za obdelavo ene poizvedbe. To so rešili pri naslednjih izdajah.

Po prodaji MySQL sistema podjetju Oracle je bila na koncu leta 2010 izdana različica 5.5. Vsebovala je izboljšave in nove funkcije, ter novo licenčno različico. Od te različice naprej je privzeti datotečni pogon InnoDB (prej je bil MyISAM). Pomembna posodobitev je vključeno delovanje SMP (ang.: Symmetric multiprocessing), s čimer so izboljšali zmogljivost podatkovne baze. Dodali so še komplet znakov Unicode: utf16, utf32 in utf8mb4. Leta 2013 so izdali še različico MySQL 5.6, v kateri so izboljšali upravljanje in poizvedovanje iz velikih tabel ter dodali nove API-je.

Najnovejša različica nosi številko 8.0 in izboljšuje združljivosti z jezikom JSON. V tej različici so izboljšali standardne vloge ter dodali podatkovni slovar, ki hrani podatke o objektih podatkovne baze.

Uporabniki MySQL podatkovne baze so izrazili svoje mnenje, da je namestitev MySQL-a najenostavnejša glede na druge podatkovne baze. Poleg tega ima zelo zanimiv način plačila ter je podprta za najbolj popularne operacijske sisteme in knjižice, preko katerih dostopamo do podatkov. Odkar je lastnik sistema MySQL podjetje Oracle mnogi uporabniki menijo, da MySQL ne

sodi več v kategorijo brezplačnega odprtega vira, saj je veliko pomembnih funkcionalnosti plačljivih [14]. Omogoča tudi delovanje na več platfomah, kot so npr. FreeBSD, Linux, OS X, Solaris in Windows. Podatkovna baza podpira tri aplikacijske vmesnike za dostop do sistema ADO.NET, JDBC in ODBC. Če za določeno platformo ne obstaja različica, je možen prenos izvorne kode za MySQL, čemur sledi samostojen prevod. Podprti programski jeziki MySQL strežnika so Ada, D, Haskell, Caml, Ruby, C, Delphi, Java, Perl, Scheme, C, Eiffel, JavaScript, PHP, Tcl, C++, Erlang, Objective-C in Python.

Pri MySQL lahko izbirate med šestimi datotečnimi sistemi za upravljanje s podatki. Relacijski sistem za upravljanje podatkovnih baz uporablja datotečni pogon za ustvarjanje, branje in posodabljanje podatkov v podatkovni bazi. Tem modulom običajno rečemo pogon za shranjevanje. Pred različico MySQL 5.5 je bil privzeti datotečni pogon MyISAM (okolje za vse operacijske sisteme razen za Windows). Ta pogon ni podpiral transakcij in je ponujal zaklepanje na ravni tabele. Za novejše različice od 5.5 je privzeti pogon InnoDB, ki je tudi najbolj razširjen mehanizem za shranjevanje podatkov s podporo za transakcije. Omenjeni pogon podpira zaklepanje na ravni vrstic in obnavljanje podatkov po nesrečah. MySQL podpira še hitri pogon MEMORY (prejšnje ime HEAP), ki je idealen za ustvarjanje začasnih tabel in hitrih poizvedb. Podatki pri tem pogonu se izgubijo, ko se podatkovna baza ponovno zažene. Ostali pogoni, ki jih podpira MySQL so Merge, Archive in CSV [15].

3.1.3 Različice MySQL

MySQL Community Edition

MySQL Community Edition je najbolj priljubljena brezplačna različica podatkovne baze. Na voljo je pod odprtokodno licenco GPL in jo podpira velika skupnost razvijalcev. GPL licenca dovoljuje prosto razmnoževanje in ureja-

nje kode programa. Razlog za popularnost MySQL Community Edition se nahaja v aktivni podpori oziroma skupnosti MySQL uporabnikov. Različica nam omogoča izbiro med datotečnimi pogoni, ki smo jih prej omenili. Ponuja tudi druge funkcionalnosti shranjenjenih procedur, prožilcev, pogledov, MySQL replikacije, MySQL particije itd. MySQL replikacija je proces, ki omogoča enostavno vzdrževanje kopij podatkov, tako da jih samodejno kopira iz gospodarja na sužnje. Funkcionalnost MySQL Particije pa omogoča horizontalno razbitje ene tabele na več manjših.

MySQL plačljive različice

Če želimo imeti aktivno tehnično podporo podjetja Oracle in javno distribuirati svoj projekt, ki uporablja MySQL podatkovno bazo, potem moramo kupiti licenčno različico MySQL strežnika. Ena izmed licenčnih različic je MySQL Standard Edition. Ta temelji na MySQL Community Edition. MySQL Standard Edition vključuje celovit nabor funkcij, orodij za upravljanje ter doseganja najvišje varnosti. Pri nakupu te različice dobimo popolno različico programa MySQL Workbench, ki je razvojno, načrtovalno in administrativno okolje za upravljanje podatkovne baze.
Najbolj zanesljiva in varna različica MySQL sistema je MySQL Enterprise Edition. Ta različica vključuje še nabor funkcij in orodij za upravljanje podatkovne baze kot Standard Edition. Uporabnikom pomaga MySQL tehnična podpora, ki svetuje pri razvijanju in načrtovanju ter odgovarja na tehnična vprašanja. Različica ima v svojem paketu še nekaj programov oziroma orodij, ki izboljšajo podatkovno bazo in olajšajo delo razvijalcem in uporabnikom. Eden izmed teh programov je MySQL Query Analyzer, ki je namenjen optimizaciji in analizi poizvedb. Ta program razvijalcem olajša identifikacijo dragih poizvedb, pomaga odpravljati problematično kodo SQL ter spremlja izvajanje poizvedb v realnem času. Drugo pomembno orodje je MySQL Enterprise Firewall, ki blokira napade na podatkovno bazo, kot je SQL Injection. Paket vsebuje tudi MySQL Enterprise Scalability, ki je namenjen izboljšanju zmogljivosti podatkovne baze pri večjem številu uporabnikov in poizvedb.

3.2 DB2

DB2 predstavlja družino izdelkov sistema za upravljanje relacijskih podatkovnih baz, ki tečejo na različnih operacijskih sistemih. IBM je DB2 prvotno uvedel leta 1983 za delovanje na svoji platformi MVS (Multiple Virtual Storage). Sprva je bil namenjen izključno delu na IBM-ovih velikih platformah, kasneje pa so ga prenesli v druge razširjene operacijske sisteme, kot so UNIX, Windows in Linux [16]. Nudi podporo številnim programskim jezikom in je združljiv z jeziki, kot sta XML in JSON. DB2 podatkovno bazo priporočajo podjetja, ki imajo opravka z veliki obremenitvami. Omogoča obdelavo velikih količin podatkov in istočasno služi številnim uporabnikom. Po podatkih IBM-a je podatkovna baza DB2 vodilna na trgu za operacijske sisteme Linux.

3.2.1 Zgodovina DB2

Začetki sistema DB2 segajo v sedemdeseta leta, ko je Edgar F. Codd, raziskovalec, ki je delal za IBM, opisal teorijo relacijskih podatkovnih baz. Leta 1974 je raziskovalni center IBM v San Joseju razvil relacijski sistem za upravljanje podatkovnih baz, System R, za izvajanje Coddovih konceptov. Podjetje IBM na začetku ni verjelo Coddovim idejam, zato so ustvarili oddelek, kjer niso bili pod nadzorom Codda in so kršili več pravil njegovega modela. Rezultat je bil strukturiran poizvedovalni jezik SEQUEL. Kasneje so SEQUEL preimenovali v SQL (ang.: Structured Query Language), saj je bila kratica SEQUEL blagovna znamka letalske družbe iz Združenega kraljestva.

Prvi komercialni produkt relacijske podatkovne baze so izdali leta 1981 in je bil namenjen za operacijske sisteme DOS, VSE in CMS. Ime DB2 je bilo prvič dodeljeno sistemu za upravljanje podatkovnih baz leta 1983, ko je IBM izdal DB2 na svoji glavni platformi MVS. V preteklosti je IBM za vsakega od svojih glavnih operacijski sistemov izdelal specifični DB2 izdelek. V začetku devetdesetih je IBM razvil DB2 tudi za ostale platforme, vključno z OS / 2, UNIX, Windows, nato Linux in dlančnike. Sredi devetdesetih let prejšnjega

stoletja je IBM izdal združen sistem imenovan DB2 Parallel Edition, ki ga je na začetku izvajal operacijski sistem AIX. To je bila velika zbirka podatkov, razdeljena na več strežnikov, ki komunicirajo preko hitrega omrežja. Ta izdaja se je sčasoma prenesla na vse platforme Linux, UNIX ter Windows in je bila preimenovana v DB2 (ang.: Extended Enterprise Edition, EEE) [17]. Slednja je lahko delovala na različni strojni opremi, od enojedrnih do multiprocesorskih sistemov (ang.: Symmetric multiprocessing, SMP) ter sistemov za množično vzporedno obdelavo (ang.: massively parallel processing, MPP).

3.2.2 Lastnosti in razvoj DB2

Leta 2006 je IBM izdal različico DB2 9.0, v kateri so izboljšali metodo shranjevanja dokumentov XML, saj prej ni bila idealna ter ni podpirala nekaterih struktur kot je drevo (ang.: tree). Kasneje, z različico 9.5 so prišle tudi izboljšave pri stiskanju podatkov za indekse, začasnih tabelah itd. Ta izdaja podpira tudi globalne spremeljivke in IDE knjižnjico. Preko IDE knjižnice so podprti programski jeziki PHP, Ruby in Perl. Vpeljali so tudi več funkcij, ki uporabnikom Oracle Database olajšajo delo z DB2 pri prenašanju podatkov. Vse DB2 različice podpirajo relacijski model, vendar pa so bili v zadnjih letih nekateri izdelki tudi razširjeni za podporo objektno-relacijskim funkcijam in ne relacijskim strukturam, kot je XML.
Leta 2012 je izšla verzija DB2 10.5, ki je znana po orodju BLU Acceleration, ki omogoča hitrejši dostop do podatkov. S pomočjo orodja BLU organiziramo tabele po stolpcu namesto po vrsticah, kar privede do znatnih izboljšav pri poizvedbah [18]. Najnovejša različica je DB2 11.1 in smo jo uporabili v okviru testiranja. Ta ponuja zelo enostavno selitev podatkov iz drugih različic DB2 podatkovnih baz. Novosti so številne SQL funkcije in regularne funkcije REGEXP.

V DB2 sistemu imajo vse različice enako jedro. Edine tehnične razlike med različicami DB2 so omejitve virov in napredne funkcije oziroma nabori orodij.

Skupno jedro oziroma skupna baza je namerno ustvarjena. Na ta način lahko aplikacije, ki so napisane za katerokoli izdajo DB2 preprosto zamenjamo v katerokoli drugo izdajo DB2, na katerikoli platformi operacijskega sistema, ki jo podpira DB2. Ta prilagodljivost velja tudi za skrbnika baze podatkov (DBA). DBA bo z veščinami na eni izdaji DB2 v trenutku produktiven katerikoli drugi različici izdaje DB2.

Sistem DB2 omogoča različne tehnologije za povezovanje in integracijo. Tako zagotavlja konkurenčnost drugim relacijskim sistemom za upravljanje podatkovnih baz. DB2 se lahko izvaja na vseh večjih operacijskih sistemih kot so AIX, HP-UX, Linux, Solaris, Windows in z/OS. Podprti programski jeziki za DB2 so C, C++, C, Cobol, Java, Pyton, Delphi, Perl, Ruby, Fortran, PHP in Visual Basic.

3.2.3 Različice DB2

DB2 Express-C

DB2 Express-C je brezplačna izdaja podatkovne baze DB2 za razvijalce in partnersko skupnost. Je enostaven tako za uporabo, kot tudi za namestitev in je zasnovan tako, da deluje v nekaj minutah. Projekte, ki imajo shranjene podatke v DB2 Express-C lahko brez težav prestavimo v drugo različico DB2 brez potrebnih sprememb kode aplikacije. Sistem za upravljanje podatkovnih baz DB2 Express-C se lahko uporablja za razvoj in uporabo brezplačno in se lahko distribuira z rešitvami tretjih oseb brez licenčnih pravic za IBM. Lahko se namesti na fizične ali virtualne sisteme ter je optimiziran za uporabo do največ dveh jeder in 16 GB notranjega pomnilnika. Velika prednost pred drugimi ponudniki je ta, da IBM DB2 Express-C nudi podporo za svoje uporabnike. Uporabniki, ki potrebujejo večjo podporo ali dodatne zmogljivosti, kot so na primer funkcije za večjo razpoložljivost in replikacijo, morajo izvesti nadgradnjo na druge izdaje DB2. Pri testiranju smo uporabili DB2 11.1 različico podatkovne baze.

Plačljive različice DB2

Najuporabnejša plačljiva različica DB2 podatkovne baze je IBM DB2 Workgroup Server Editon, ki je namenjena za mala in srednje velika podjetja. Zagotavlja visoko razpoložljivost in varnostne funkcije. Prednost plačljive DB2 Workgroup Server Edition je večdimenzionalno združevanje podatkov v gruče, kar izboljša zmogljivost poizvedbe. Poleg tega znatno zmanjša stroške vzdrževanja podatkov, kot so reorganizacija in vzdrževanje indeksov med postopki vstavljanja, posodabljanja in brisanja. Pomaga pri poenostavitvi upravljanja projektov podatkovnih aplikacij podjetij, hkrati pa zagotavlja naprednejšo varnost in šifriranje. Ponuja funkcije združljivosti SQL, ki pomagajo zmanjšati stroške in tveganje selitve starih aplikacij baz podatkov Oracle v DB2. Glede na zgoraj navedeno lahko trdimo, da je DB2 Express-C izjemno podobna različici DB2 Workgroup Server Editionu, razlika je samo v dodatnih funkcijah in omejitvi pomnilnika. DB2 Workgrop ima začetno omejitev do 64 GB pomnilnika ter štirih jeder, če pa potrebujemo več pomnilnika, ga lahko enostavno povečamo.

DB2 Enterprise Server Edition je najbolj popolna podatkovna baza podjetja IBM. Vključuje vse funkcije DB2 Workgroup Server Edition in hkrati omogoča napredno upravljanje s podatki, visoko razpoložljivost, obnavljanje podatkov po nesrečah, ki naj bi zagotovile visoko zmogljivost in večjo učinkovitost. Rešitev je namenjena srednjim in velikim podjetjem. Ima svoje dodatne pakete, ki povečajo zmogljivost podatkovne baze. Eden izmed teh paketov je DPF paket (ang.: Database Partitioning Feature), ki omogoča razdelitev podatkov na enega ali več strežnikov, ki tečejo na istem operacijskem sistemu.
Pomembna različica je DB2 Everyplace Edition. Ta različica je namenjena za manjše naprave, kot so dlančniki (ang.: personal digital assistant, PDA), ročni računalniki, vgrajene naprave in podobno. Značilno za to različico je, da se podatki najprej shranijo na mobilni podatkovni bazi in se kasneje prenesejo na strežnik z DB2 Everyplace Sync, ki je nameščen na drugi napravi.

Ta različica deluje na operacijskih sistemih kot so Linux, Symbian OS, Windows Mobile for Pocket PC.

Tabela 3.1 prikazuje primemerjavo glavnih lastnosti med MySQL in DB2 bazo.

Podatkovna baza	MySQL	DB2
Objavljena leta	1995	1983
Napisana v programskem jeziku	C in C++	C in C++
Podprti operacijski sistemi	AIX, HP-UX, Linux, Solaris, Windows, z/OS	FreeBSD, Linux, OS X, Solaris, Windows
Podprti sekundarni indeksi	DA	DA
Pogledi	DA	DA
Prožilci	DA	DA
Podpora JSON	DA	DA
Podpora XML	NE	DA
Model replikacije	Model gospodar-suženj	Model gospodar-suženj, HADR
Šifriranje podatkov	DA	DA
Podpora grafov	NE	DA
Tehnična podpora 24/7	DA	DA
Najnovejša različica	MySQL 5.7.20	DB2 11.1

Tabela 3.1: Primerjava lastnosti med MySQL in DB2 bazi.

Poglavje 4

HDD in SSD diski

V tem poglavju predstavljamo sestavne dele, prednosti in delovanje klasičnih trdih diskov in SSD diskov.

4.0.1 HDD in SSD disk

Pogost problem pri hitrosti podatkovne baze predstavlja veliko število V/I operacij povezanih z diski. Kljub temu, da se te naprave nenehno razvijajo, je hitrost delovanja podatkovne baze še vedno omejena s hitrostjo branja in pisanja z diska. Včasih se dogaja, da sistem čaka na zaključek operacij, ki jih izvaja disk. Pri kreiranju podatkovne baze se moramo zavedati, da disk predstavlja ozko grlo in glede na njegove zmogljivosti je potrebno pravilno načrtovati podatkovno bazo. Zaradi razlik v hitrosti med HDD in SSD, smo obe vrsti diskov uporabili pri testiranju.

Klasični trdi disk (ang.: Hard Disk Drive, HDD) je najbolj razširjena vrsta zunanjega pomnilnika. Sestavljen je iz več okroglih kovinskih plošč, ki se med delovanjem vrtijo, prevlečene pa so z magnetno snovjo. Nad diskom je bralno-pisalna glava, ki lebdi nekaj nanometrov nad površino plošče s pomočjo samega zraka, ki ga generira disk. Današnji trdi diski dosežejo hitrost prenosa podatkov do 50 MB/s [19].

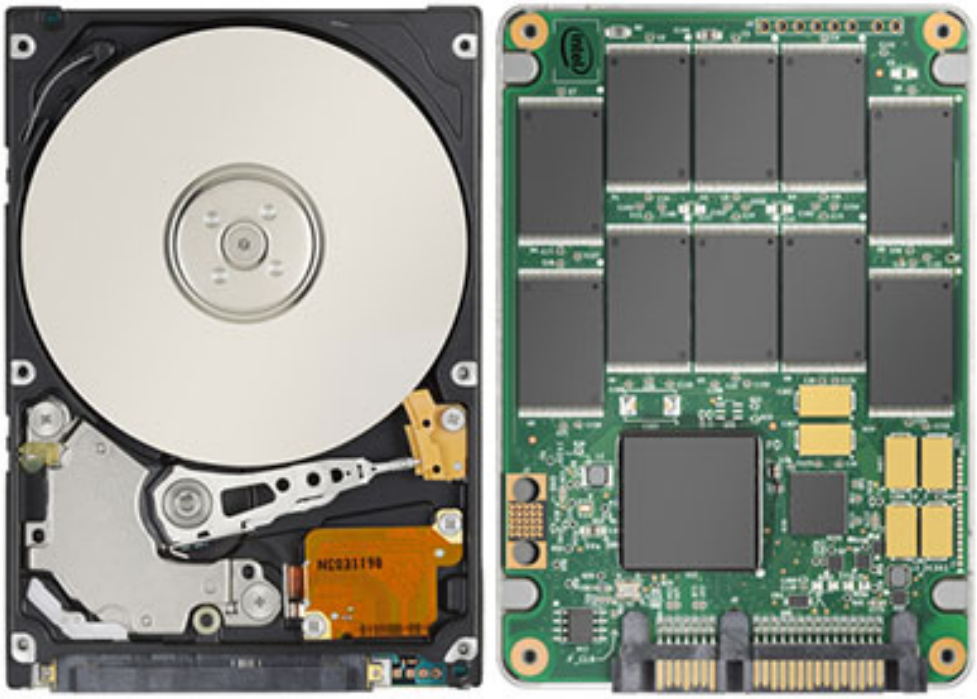
Pri zapisovanju v podatkovno bazo je bolje, da imamo na voljo večjo kapaciteto diska, saj tako algoritmi hitreje najdejo prazen prostor za shranjevanje. Podatki so razporejeni v koncentričnih krožnicah, ki jim rečemo sledi. Vsaka sled je razdeljena na manjše enote, ki jih imenujemo sektorji, ti pa tvorijo zaključene podatkovne enote. Na začetku vsakega razdelka diska je tabela, v kateri je zapisano kateri sektorji so prosti in kateri zasedeni (v operacijskem sistemu Windows se tej datotečni tabeli reče FAT32 in NTFS). Če računalnik želi shraniti nove podatke, si najprej ogleda datotečno tabelo in poišče proste sektorje, nato naroči glavi za branje in pisanje podatkov, da se premakne točno na pravo mesto in podatek zapiše. Za branje informacij je proces enak, samo da poteka v obratnem vrstnem redu. Najbolje je, da so podatki, ki jih velikokrat iščemo, zapisani skupaj.
Za dostopanje do podatkov shranjenih na disku potrebujemo čas, ki ga imenujemo dostopni čas. Izračunamo ga tako, da seštejemo iskalni čas, vrtilno zakasnitev in čas prenosa podatkov. Iskalni čas je čas potreben za premik glave nad ustrezen izsek, vrtilna zakasnitev pa je čas, ki ga potrebuje bralno-pisalna glava.

SSD disk (ang.: Solid-state drive) je naslednik trdega diska, deluje podobno kot USB pomnilnik in nima gibljivih delov. Za hrambo podatkov SSD uporablja med seboj povezane pomnilniške flash čipe. SSD nima gibljivih delov in je do 30% hitrejši v primerjavi z HDD diskom. Tehnologija v SSD disku je veliko bolj napredna in dovršena kot v njegovem predhodniku, zato pa je SSD disk trikrat dražji [20].

Obstajajo tudi hibridni diski, ki so mešanica HDD in SSD diska, rečemo jim SSHD (ang.: solid-state hybrid drive) diski. Razlike v hitrosti med hibridnim in HDD diskom se na začetku ne opazi, saj programska oprema šele kasneje ugotovi, kateri podatki so pogosto uporabljeni in jih prenese na SSD del diska. Hibridni diski so bolj vzdržljivi od HDD diskov, saj zaradi prenosa podatkov na SSD disk ne pride do veliko enakih zapisov, ki fizično vničujejo

disk. Ti diski so hitrejši od klasičnih trdih diskov in počasnejši od SSD diskov.

Slika 4.1 prikazuje izgled klasičnega trdega diska in SSD diska. Levi del slike prikazuje klasični disk. Glavne komponente klasičnega diska so osnovno ohišje, magnetne plošče, bralno-pisalna glava in koračni motorji, ki premikajo plošče in glave. Drugi del slike oziroma desna stran slike nam prikazuje sestavne dele SSD diska. Kot vidimo je SSD disk sestavljen iz flash čipov, predpomnilnika in krmilnika.



Slika 4.1: HDD in SSD disk [20].

4.0.2 Prednosti SSD in HDD diska

Prednosti SSD diska so:

- precej krajši dostopni čas (približno 100 krat),
- hitrejši prenos podatkov, tipično 100 MB/s do 600 MB/s (HDD disk ima največjo hitrost 50 MB/s),
- manjša poraba energije in manjše segrevanje ter
- neobčutljivost na udarce.

Prednosti HDD diska so:

- cenovno dostopnejši kot SSD disk (cena SSD diska z enako kapaciteto kot HDD disk je dvakrat večja),
- večja kapaciteta diska,
- večja življenjska doba (gre se za pisanje operacij) in
- možnost dolgotrajnega skladiščenja podatkov brez elektrike (pri SSD-ju se dogaja, da se podatki izbrišejo po približno enem letu).

Poglavje 5

Testiranje in analiza

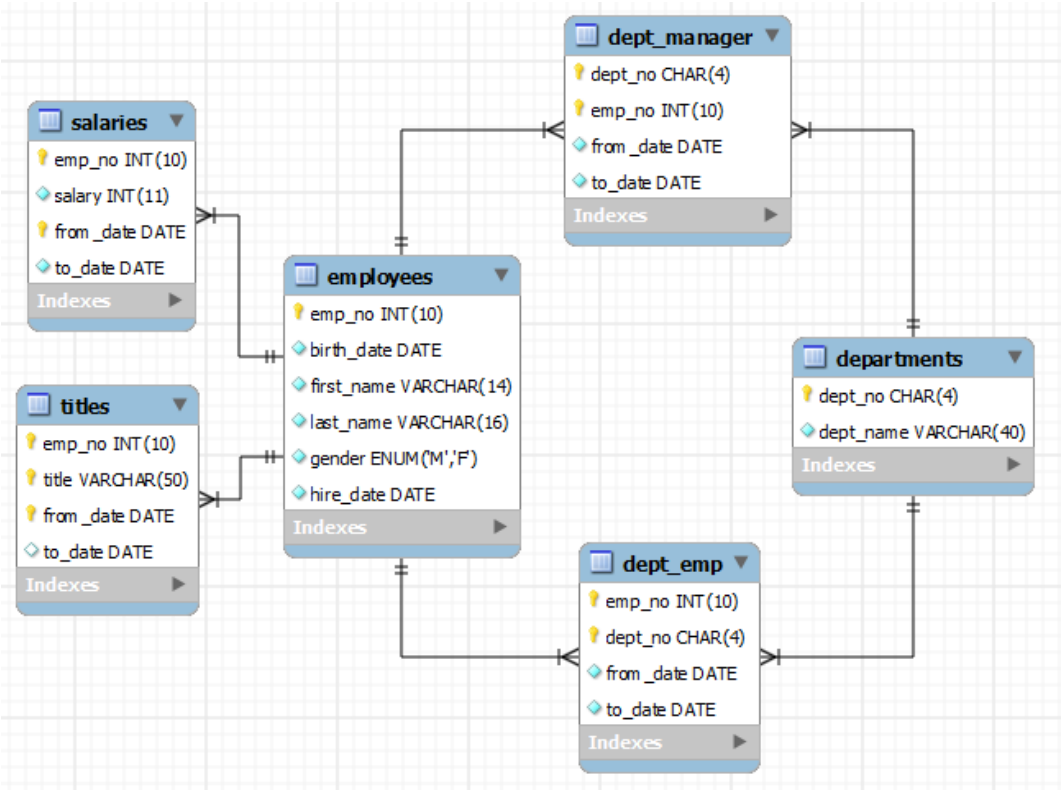
Pred začetkom analize smo namestili podatkovni bazi na klasičen in SSD disk. V tem poglavju bomo opisali, kako je testiranje potekalo ter katere programe smo uporabili.

5.1 Namestitev programov

Performančno testiranje podatkovnih baz smo pričeli z namestitvijo obeh podatkovnih baz na računalnik. Pred začetkom testiranja smo se morali odločiti, kateri programski jezik uporabiti za izdelavo testov. Odločili smo se, da bomo uporabli programski jezik Java, saj ga najbolj poznamo in največ uporabljamo. Celotno testiranje hitrosti podatkovnih baz je potekalo z uporabo programa DBVisualizer in programa Netbeans. Oba programa sta odprtokodna in zastonj za uporabnike. Program DBVisulizer je bil uporabljen za kreiranje tabel, vstavljanje in brisanje podatkov. Drugi omenjeni program Netbeans pa je bil uporabljen za testiranje hitrosti branja podatkov. Za testiranje v zgoraj navedenih programih smo se odločili, zato ker nimata nobene povezave z podjetjema IBM in Oracle.

Za potrebe testiranja je bilo potrebno namestiti naslednje programe:

- MySQL Server,
- Db2 Express-C (Version 11.1),
- DBVisualizer 9.5.7 in
- NetBeans IDE 8.2.



Slika 5.1: Logični model testne podatkovne baze [21].

Slika 5.1 prikazuje podatkovni model testne podatkovne baze. Kot vidimo, baza vsebuje šest tabel. To so zaposleni (ang. employees), plače (ang. salaries), naziv (ang. titles), oddelki (ang. departments), oddelek zaposlenega (ang.dept_emp) in upravitelj oddelka (ang. dept_manager). Največ vrstic ima tabela salaries, ki vsebuje 970011 zapisov. Najmanj zapisov pa tabela

departments, ki jih ima 9. Tabela employees vsebuje 300024 zapisov, tabela titles 43308, tabela dept_emp 331603 in tabela dept_manager 24. Vseh podatkov skupaj je 2.144.979 vrstic, kar predstavlja obsežno množico podatkov. Slika 5.2 prikazuje deset zapisov v tabeli employees.

*	emp_no	birth_date	first_name	last_name	gender	hire_date
1	10001	1953-09-02	Georgi	Facello	M	1986-06-26
2	10002	1964-06-02	Bezalel	Simmel	F	1985-11-21
3	10003	1959-12-03	Parto	Bamford	M	1986-08-28
4	10004	1954-05-01	Chirstian	Koblick	M	1986-12-01
5	10005	1955-01-21	Kyoichi	Maliniak	M	1989-09-12
6	10006	1953-04-20	Anneke	Preusig	F	1989-06-02
7	10007	1957-05-23	Tzvetan	Zielinski	F	1989-02-10
8	10008	1958-02-19	Saniya	Kalloufi	M	1994-09-15
9	10009	1952-04-19	Sumant	Peac	F	1985-02-18
10	10010	1963-06-01	Duangkaew	Piveteau	F	1989-08-24

Slika 5.2: Tabela employees.

5.1.1 Namestitev MySOL podatkovne baze

Najprej smo namestili podatkovno bazo MySQL Community Server. Za namestitev omenjene baze je nujna namestitev MySQL Installerja. S pomočjo programa MySQL Installer lahko namestimo celoten paket programov potrebnih za delovanje MySQL serverja. Če smo kdaj nameščali kakšno drugo podatkovno bazo potem vidimo, kako je MySQL baza enostavna za namestitev. Potrebno je le slediti korakom po navodilih. Pri MySQL strežniku lahko izbiramo med 32- in 64-bitno različico. Pri namestitvi je potrebno izbrati Developer Machine, ostale nastavitve lahko pustimo nespremenjene. Skrbniki imamo avtomatsko dodeljeno uporabniško ime »root«, geslo pa moramo sami vnesti. Po namestitvi MySQL serverja nam program MySQL Installer ponudi še namestitev drugih programov, ki pomagajo pri upravljanju podatkovne baze kot so MySQL Workbench, Shell, Notifier in MySQL

Documentation.
Za testiranje podatkovne baze namestimo še mysql-connector-java-5.1.43 gonilnik, s katerim naredimo povezavo med MySQL strežnikom in testno skripto preko katere beremo podatke. MySQL Connector je tipa JDBC in s tem API-jem povežemo našo testno skripto, ki je napisana v programskem jeziku Java. Podatkovno bazo MySQL povežemo s spodnjimi vrsticami.

private static final String USERNAME = "root";
private static final String PASSWORD = "diplomska12";
private static final String CONN_STRING =
"jdbc:mysql://localhost:3306/employees?autoReconnect=trueuseSSL=false";

5.1.2 DB2

Za namestitev podatkovne baze DB2 je bila potrebna registracija na IBM spletni strani. Po uspešni prijavi v IBM platformo je sledil prenos podatkovne baze DB2 Express-C (Version 11.1). Sama namestitev je zelo enostavna, kot pri MySQL je potrebno le sledili navodilom. Pred koncem namestitve je potrebno vnesti uporabniško ime in geslo, zato smo vnesli uporabniško ime "db2admin"in geslo "diplomska12".Na koncu namestitve nam je ponujena testna podatkovna baza »SAMPLE«, da testiramo, če je namestitev strežnika uspela. Za namestitev programa IBM Data Studio smo morali naprej namestiti IBM Installation Manager, preko katerega smo prenesli program Data Studio, s katerim IBM priporoča upravljenje njihove podatkovne baze. Za povezavo s testno skripto za branje podatkov smo prenesli JDBC 4.0 DB2 Driver.

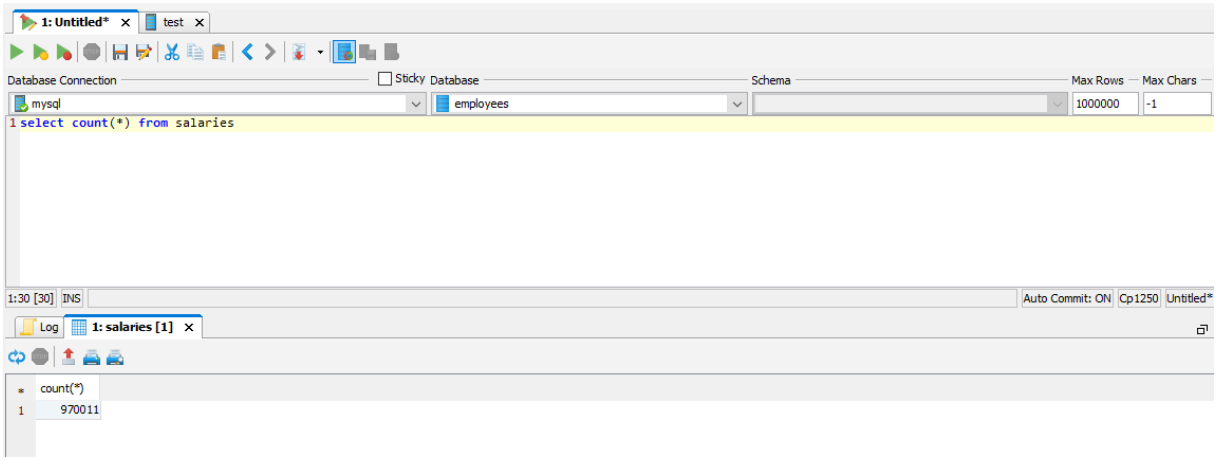
Na podatkovno bazo DB2 smo se povezali z DriverManager.getConnection metodo. Spodnje vrstice prikazujejo potrebne podatke za povezavo na podatkovno bazo DB2.

```
Class.forName("com.ibm.db2.jcc.DB2Driver");
conn = DriverManager.getConnection
("jdbc:db2:employee","db2admin","diplomska12");
```

5.1.3 DBVisualizer in NetBeans

V okviru faze testiranja smo uporabljali neodvisno orodje za upravljanje podatkovnih baz DBVisualizer, saj ta omogoča upravljanje DB2 in MySQL podatkovnih baz. Progam podpira še druge podatkovne baze, kar je njegova prednost. Omogoča uporabnikom, da so lahko hkrati povezani na več podatkovnih baz. Za testiranje smo prenesli in namestili brezplačno različico 9.5.7. Ob želji se lahko dokupi licenčna različica programa, ki omogoča dodatne funkcionalnosti, kot so uvoz podatkov iz CSV datotek, podpora za samodejno dokončanje SQL poizvedb itd. Okolje ima zelo pregleden in dobro organiziran grafični vmesnik. Ob povezavi na svojo podatkovno bazo je možen enostaven ogled tabel, pogledov, sistemske tabele, procedure itd. Program lahko upravlja spodnje podatkovne baze [22] DB2, D, Infomix, JavaDB, H2, Mimer, SQL, MySQL, Netezza, NuoDB, Oracle, PostgreSQL, Redshift, SQL Server, SQLite, Sybase in Vertica.

Slika 5.3 prikazuje orodje DBVisualizer za upravljanje podatkovnih baz.



Slika 5.3: DBVisualizer.

Drugi program, ki smo ga uporabili pri branju podatkov iz podatkovnih baz, je NetBeans. NetBeans je odprtokodno razvijalno okolje za razvoj javanskih aplikacij. Na voljo so dodatki, ki omogočajo razvoj v drugih programskih jezikih kot sta C in C++. Omogoča razvoj aplikacij na platformah kot so Linux, macOS, Windows in Solaris.

5.2 Rezultati testiranja

V prejšnjem poglavju smo omenili, da smo za testiranje podatkovnih baz uporabili testno podatkovno bazo s šestimi tabelami. Ta podatkovna baza je zastonj in je načeloma narejena za podatkovno bazo MySQL, zato smo morali testne podatke najprej nekoliko prilagoditi, da so ustrezali DB2 bazi. Čeprav obe podatkovni bazi uporabljata enak poizvedovalni jezik (SQL jezik), nimata enakih podatkovnih tipov in se razlikujeta v sintaksi pisanja poizvedb. Vsi podatki zapisani v podatkovni bazi so izmišljeni. Še preden smo začeli s testiranjem, smo morali napisati poizvedbe, ki smo jih izvajali nad podatki. Ko smo imeli poizvedbe napisane, smo jih sprva testirali

na manjši količini podatkov, zaradi preverjanja pravilnosti izpisa rezultata. Testiranje smo opravili v obliki testnih scenarijev. Pri vsakem scenariju testiranja je napisan primer testne poizvedbe. Pomembno je opozoriti, da je bilo testiranje opravljeno na različnih scenarijih, ki so izdelani na podlagi realnih dogodkov, s čimer želimo predstaviti prednosti in slabosti podatkovnih baz MySQL in DB2. Za še večjo verodostojnost podatkov smo vsak test oziroma poizvedbo pognali petkrat. Največjo in najmanjšo vrednost smo odstranili in na podlagi ostalih treh je izračunana srednja vrednost. Srednje vrednosti časa izvajanja poizvedb so napisane v tabelah in prikazane na grafih. Za boljše preglednosti smo pri vseh scenarijih narisali graf do 25000 testnih primerov. Če bi narisali večje vrednosti od 25000 bi postal graf nepregleden. Testna koda se nahaja v prilogi in v testnih scenarijih. Vsi rezultati meritev so podani v sekundah.

Uporabljeni so bili naslednji testni scenariji:

- vstavljanje podatkov v prazno tabelo,
- vstavljanje v tabelo s podatki,
- brisanje podatkov,
- branje podatkov in
- branje podatkov v indeksiranih tabelah.

Podatkovne baze smo testirali na operacijskem sistemu Windows 10 64 bit na računalniku z Intel Core^{TM} i5 2.60GHz procesorjem in 8,00 GB notranjega. Teste smo pognali dvakrat, enkrat z uporabo klasičnega in drugič z uporabo SSD diska. Klasični disk je imel kapaciteto 1 TB, 5400 vrtljajev v minuti ter povprečno hitrost branja in pisanja 150 MB na sekundo. SSD disk UV400 pa ima kapaciteto 120 GB, hitrost branja podatkov 450 MB/S in pisanja 180 MB/s.

5.2.1 Scenarij vstavljanja podatkov v prazno tabelo

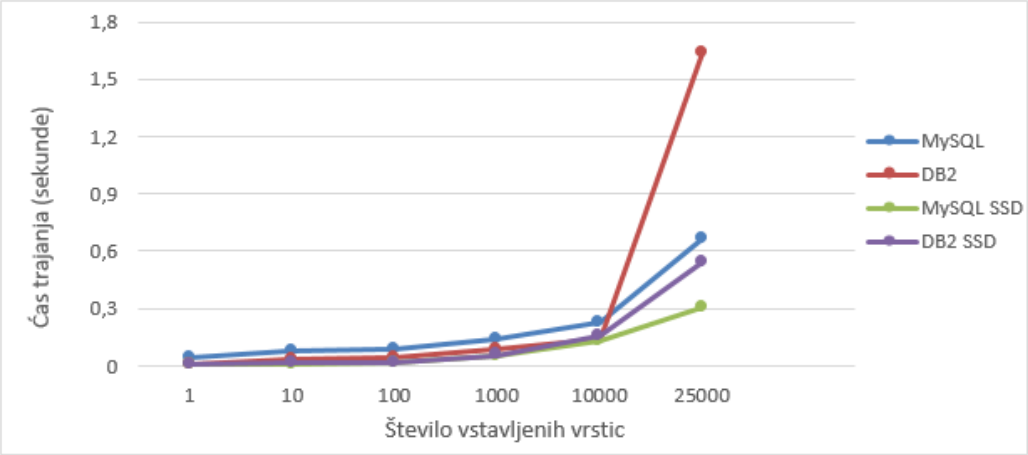
Prvi izmed scenarijev testiranja je bil vstavljanje podatkov v prazno tabelo. Pri tem scenariju smo merili čas izvajanja pri vstavljanju podatkov v prazno tabelo »salaries«. Po vstavljanju smo podatke pobrisali z ukazom »DELETE FROM ImeTabele« in ponovili ukaz za vstavljanje. Tabela »salaries« ima naslednje atribute:

emp_no INT
salary INT
from_date DATE
to_date DATE

Primer ukaza za vstavljanje podatkov je: **INSERT INTO salaries VALUE (10001,60117,'1986-06-26','1987-06-26');**

Zgornji primer ukaza prikazuje, iz katerih atributov je sestavljen en zapis v tabeli »salaries«. Ta ukaz za vstavljanje podatkov deluje v obeh bazah. Tabela »salaries« vsebuje podatke o plačah za vsakega zaposlenega v določenem obdobju. V testni bazi smo za vsako tabelo posebej imeli pripravljene podatke v besedilnem formatu. Podatke smo kopirali v program DBVisualizer, s katerim smo vstavljali v bazo. Logični model, ki je na sliki 5.1, nam je služil kot dokumentacija, saj prikazuje, kako mora biti načrtovana naša testna baza »zaposleni«. Naenkrat smo vstavili največ 25000 vrstic v bazo.

Testni primeri	MySQL	DB2	MySQL SSD	DB2 SSD
1	0,04	0,007	0,005	0,007
10	0,082	0,034	0,011	0,015
100	0,089	0,041	0,014	0,016
1000	0,139	0,086	0,054	0,056
10000	0,23	0,142	0,133	0,159
25000	0,661	1,168	0,302	0,542



Slika 5.4: Graf merjenja časa pri vstavljanju podatkov.

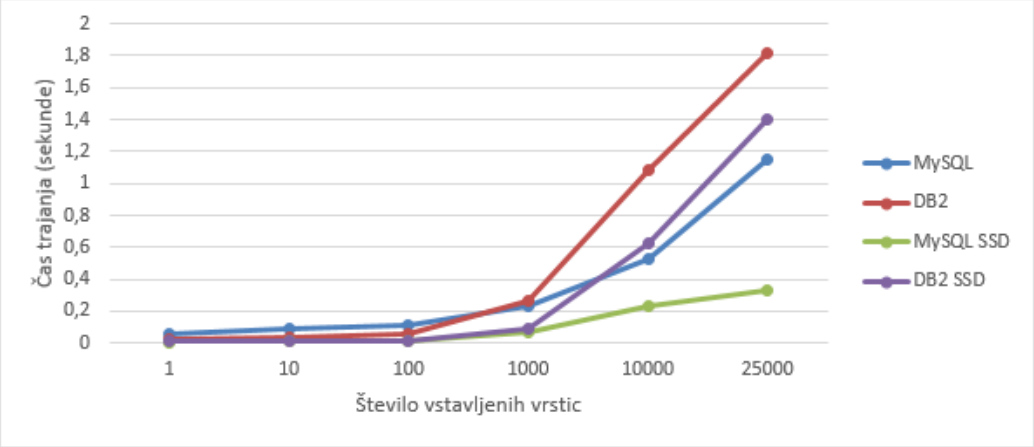
Kot je že razvidno iz zgornje tabele, smo vstavljali 1, 10, 100, 1000, 10000 in 25000 vrstic. V tabeli so vnesene srednje vrednosti časa vstavljanja podatkov v tabelo. Na podlagi podatkov iz tabele smo narisali graf. Glede na rezultate lahko rečemo, da se pri vstavljanju podatkov bolj izkaže MySQL podatkovna baza. Pri majhni količini podatkov ji sledi DB2, ki je malce počasnejša. Pri vstavljanju večje količine podatkov se podatkovna baza DB2 ne izkaže, saj je čas veliko daljši v primerjavi z MySQL podatkovno bazo. Po predvidevanjih bi z večanjem vstavljanja testnih primerov krivulja grafa postala bolj strma in nakazovala časovno potratnost DB2 podatkovne baze shranjene na navadnem trdem disku. Čas vstavljanja pri DB2 se izboljša pri podatkovni bazi shranjeni na SSD disku.

5.2.2 Scenarij vstavljanja podatkov

Razlika med prejšnjim scenarijem je, da smo pri tem testu podatke vstavljali v tabelo, ki ni prazna. Spet smo vstavljali v tabelo "salaries", ki že vsebuje 970011 zapisov.
Podatki so enako generirani kot pri prejšnjem scenariju vstavljanja. Primer za vstavljanje dveh vrstic naenkrat v tabelo je

INSERT INTO salaries VALUE
(10001,88958,'2002-06-22','9999-01-01'),
(10002,65828,'1996-08-03','1997-08-03');

Testni primeri	MySQL	DB2	MySQL SSD	DB2 SSD
1	0,059	0,021	0,007	0,009
10	0,086	0,035	0,019	0,016
100	0,115	0,063	0,026	0,026
1000	0,232	0,261	0,072	0,094
10000	0,525	1,08	0,229	0,629
25000	1,155	1,819	0,336	1,399



Slika 5.5: Graf merjenja časa pri vstavljanju podatkov v tabelo, ki ni prazna.

Rezultati testa vstavljanja v tabelo ki ni prazna pokažejo, da je podatkovna baza MySQL tudi tokrat hitrejša od DB2 podatkovne baze. Pri majhni količini podatkov so časi merjenja vstavljanja podatkov skoraj enaki. To pomeni, da nobena baza ni boljša ali slabša pri manjši količini podatkov. Pri večji količini pa lahko vidimo, da je podatkovna baza DB2 počasnejša. Podatkovni bazi shranjeni na SSD disku so skoraj dvakrat hitrejše od podatkovnih baz shranjenih na navadnih HDD diskih.

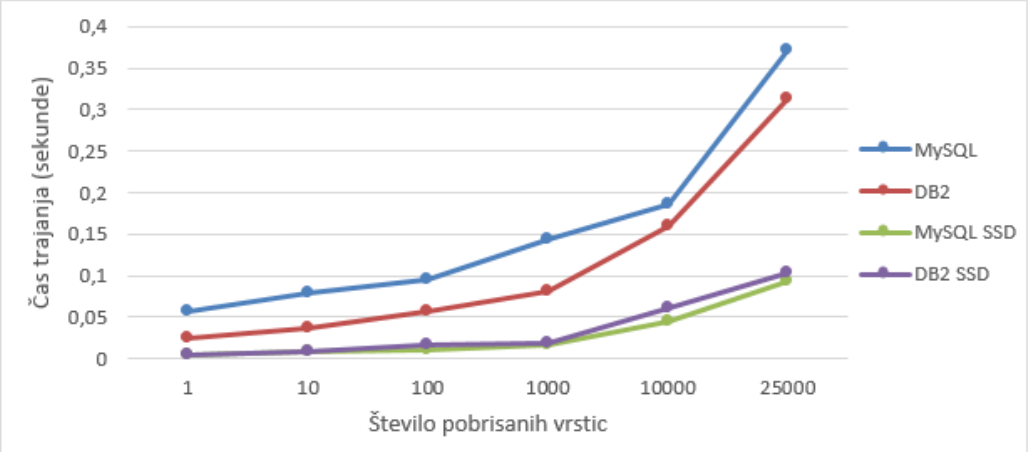
5.2.3 Scenarij brisanja podatkov

Pri tem scenariju smo merili čas brisanja podatkov iz tabele. Sprva smo vstavili podatke v prazno tabelo in jih nato izbrisali. To pomeni, da smo vsakič izbrisali vse podatke iz tabele. Zopet smo testirali brisanje od 1 do 25000 vrstic. Spodaj je primer ukaza, ki pobriše vse podatke iz tabele. Ta ukaz je enak pri obema podatkovnima bazama.

Primer ukaza: **DELETE FROM salaries;**

V tabeli so napisane dobljene vrednosti pri brisanju podatkov in na podlagi nje je narisan graf.

Testni primeri	MySQL	DB2	MySQL SSD	DB2 SSD
1	0,056	0,025	0,004	0,005
10	0,079	0,037	0,008	0,008
100	0,096	0,056	0,011	0,016
1000	0,144	0,081	0,016	0,019
10000	0,185	0,16	0,045	0,06
25000	0,371	0,313	0,094	0,103



Slika 5.6: Graf merjenja časa pri brisanju podatkov.

Iz grafa je razvidno, da je podatkovna baza DB2 hitrejša pri brisanju podatkov od MySQL baze. Graf kaže tudi, da je v primeru podatkovne baze shranjene na SSD disku čas brisanja podatkov krajši znotraj MySQL baze. Glede na pridobljene rezultate lahko rečemo, da imata obe podatkovni bazi skoraj da enak rezultat.

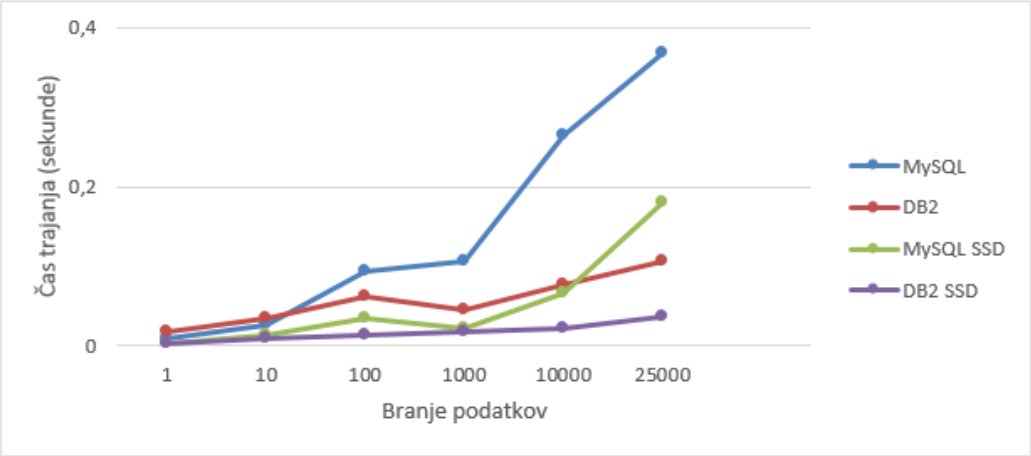
5.2.4 Scenarij branja podatkov

Pri tem scenariju preberemo določeno število podatkov. Podatke beremo iz tabel zaposleni in nazivi, ki sta povezani preko skupnega atributa emp_no. Branje podatkov je ena najbolj potratnih časovnih operacij, saj ponavadi beremo podatke iz več relacij. Primer ukaza je:

SELECT last_name, birth_date, first_name, title
FROM employees
JOIN titles ON employees.emp_no = titles.emp_no
WHERE employees.emp_no >11170 and employees.emp_no <11970
LIMIT 1000;

Zgornji primer poizvedbe bo vrnil rojstni datum, ime, priimek in naziv zaposlenega. Pri tem primeru smo poizvedbo omejili na tisoč zaposlenih z ukazom LIMIT. Z ukazom LIMIT smo določili, koliko podatkov želimo prebrati. Pri ostalih testnih primerih smo vrednost ukaza LIMIT in atributa ID_zaposlenega povečevali oziroma zmanjševali. Testna skripa, ki prikazuje primer branja podatkov je v prilogi.

Testni primeri	MySQL	DB2	MySQL SSD	DB2 SSD
1	0,009	0,018	0,003	0,003
10	0,027	0,035	0,014	0,009
100	0,093	0,063	0,035	0,013
1000	0,107	0,045	0,021	0,018
10000	0,265	0,077	0,066	0,021
25000	0,367	0,107	0,179	0,037



Slika 5.7: Graf merjenja časa pri branju podatkov.

Iz tega grafa je takoj razvidna razlika, in sicer vidimo, da je DB2 podatkovna baza hitrejša tako na HDD kot tudi na SSD disku. Večina relacijskih podatkovnih baz je hitrih pri branju podatkov, ampak so hkrati počasne pri vrnitvi rezultata, kar je eden izmed razlogov počasnosti MySQL podatkovne baze.

5.2.5 Scenarij branja indeksiranih podatkov

Indeks je podatkovna struktura, ki izboljša hitrost pridobivanja podatkov iz tabele, in sicer na račun dodatnih zapisov in prostora za shranjevanje, da se ohrani indeksna struktura podatkov. Indekse praviloma kreiramo na atribu-

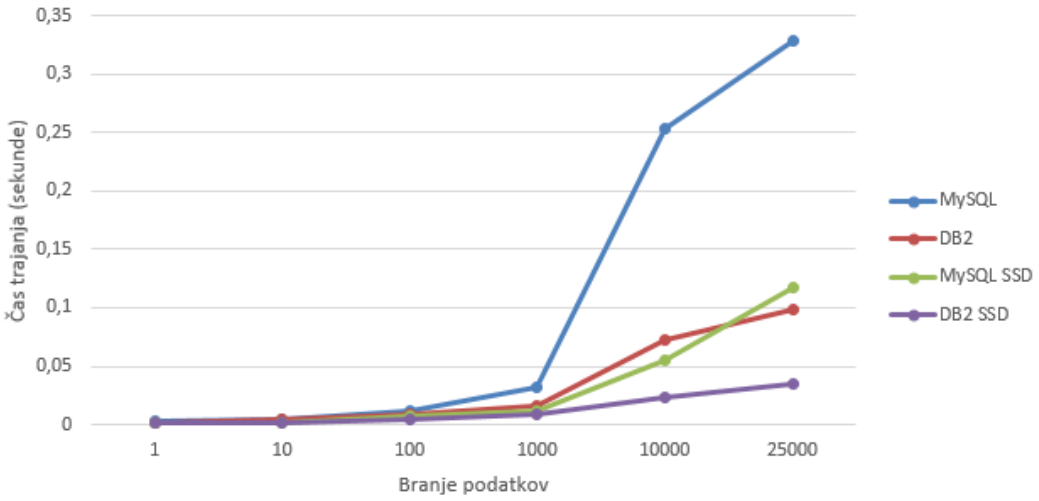
tih, ki jih pogosto uporabljamo kot kriterij pri branju zapisov iz tabele. Za primer vzemimo tabelo employees, ki ima kot primarni ključ številko zaposlenega (emp_no). Na tem atributu je že avtomatsko kreiran indeks. Razlika med indeksirano tabelo in ne indeksirano tabelo je, da pri tabeli, ki ni indeksirana, mora sistem za upravljanje podatkovnih baz pregledati vsako vrstico po vrsti, dokler ne pride do iskalnih podatkov. Za potrebe našega testiranja bomo kreirali dodatni oziroma sekundarni indeks v tabeli »employees« na atributu priimek (last_name), s katerim bomo pospešili iskanje podatkov. Za ta atribut smo se odločili, saj je v naši podatkovni bazi relativno majhna ponovitev enakih priimkov. Sekundarni indeks nam naredi učinkovitejše in hitrejše iskanje po določenih pogojih. Tabele lahko imajo več sekundarnih indeksov. Indeks smo kreirali z ukazom:

CREATE INDEX priimek_index
ON employees (last_name);

Tudi pri tem scenariju določili omejitev prebranih podatkov z ukazom LIMIT. Primer ukaza za branje iz indeksirane tabele je:

SELECT last_name, birth_date, first_name, title
FROM employees
JOIN titles ON employees.emp_no = titles.emp_no
WHERE employees.emp_no >11170 and employees.emp_no <11970
LIMIT 1000;

Testni primeri	MySQL	DB2	MySQL SSD	DB2 SSD
1	0,003	0,002	0,001	0,001
10	0,005	0,004	0,002	0,002
100	0,012	0,009	0,007	0,005
1000	0,032	0,016	0,012	0,009
10000	0,253	0,073	0,055	0,023
25000	0,329	0,098	0,117	0,035



Slika 5.8: Graf merjenja časa pri branju indeksiranih podatkov iz tabele.

Pri testiranju branja podatkov indeksiranega atributa smo spet merili čas izvajanja od 1 do 25000 vrstic. Ob primerjavi podatkov, ki smo jih prebrali iz tabele, ki ni imela sekundarnega indeksa, in tablele s sekundarnim indeksom smo ugotovili, da je branje podatkov hitrejše s sekundarnim indeksom. Ne smemo pa zanemariti, da bo sedaj vstavljanje v to tabelo dražje kot prej, saj vsakič, ko vstavimo zapis v podatkovno bazo, se bo posodobila indeksna struktura. Glede na podatke iz tabele in grafa lahko potrdimo, da je branje podatkov iz podatkovne baze DB2 hitrejše kot branje iz MySQL podatkovne baze. Iz grafa lahko odčitamo, da bo krivulja branja podatkov MySQL podatkovne baze, ki je shranjena na navadnem trdem disku strmo naraščala. Pridobivanje podatkov iz baz shranjenih na SSD disku je skoraj trikrat hitrejše kot iz baz shranjenih HDD disku.

5.3 Sklepne ugotovitve

Tako MySQL kot DB2 se še naprej razvijata. Obe podatkovni bazi stremita k načrtovanju in izgradnji nerelacijskih rešitev, ki jih lahko uporabniki uporabljajo v kombinaciji z obstoječimi relacijski rešitvami. Rečemo lahko, da MySQL podatkovna baza ni obstajala v času, ko je bila DB2 prvič ustvarjena, na priljubljenosti pa je MySQL dobila zaradi odprtokodne narave. Tako rečeno je bila v začetku bolj nezanesljiva in ranljiva, čez leta pa se je razvila v stabilen relacijski podatkovni sistem. Čeprav je podatkovno bazo DB2 možno izvajati na več operacijskih sistemih kot MySQL, pa je slednja podprta v več programskih jezikih. Če primerjamo brezplačne različice lahko rečemo, da ima MySQL podatkovna baza na voljo več prostora za shranjevanje, ampak DB2 sistem ima na voljo več funkcij. Prva ima tudi več podprtih podatkovnih struktur.

Testiranje je bilo uspešno in je potekalo brez večjih zapletov. Prišli smo do ugotovitve, da sta si bazi podobni. Glede na opravljena testiranja lahko trdimo, da je vstavljanje in brisanje podatkov hitrejše v MySQL sistemu. Za branje podatkov pa tega ne moremo trditi, saj je v tem primeru DB2 hitrejši. MySQL je bolje uporabiti, kadar imamo veliko posodobitev podatkov oziroma vnosov. Če pa veliko beremo podatke in želimo močno podporo proizvajalca, je DB2 boljša izbira. V publikacijah je rečeno, da je pridobivanje podatkov v DB2 približno trikrat hitrejše od MySQL-a, kar je tudi dokazalo testiranje v praktičnem delu. Če shranjujemo zaupne oziorma kritične podatke, je zaradi varnostnih mehanizmov boljše uporabiti DB2 bazo.
DB2 ima odlične zmogljivosti za združevanje podatkov iz več virov, vendar je za to storitev potrebno plačati.
Odločilno vlogo pri hitrosti pridobivanja podatkov ima poleg strukture relacijskega sistema hitrost trdega diska ali drugega medija, kjer so podatki shranjeni. V diplomski nalogi smo dokazali, da je najmanj dvakrat hitrejše zapisovanje in branje podatkov na SSD disku v primerjavi s klasičnim diskom. Ne smemo pa zanemariti, da še vedno imajo SSD diski omejeno število

zapisovalnih ciklov, ki jih lahko prenesejo.

Poglavje 6

Zaključek

Pred vsako izbiro podatkovnega modela primernega za določen projekt oziroma aplikacijo je potrebno dobro premisliti. Izbor relacijske podatkovne baze, ki je za določen problem najprimernejša, mora biti osnovana na podlagi namena, za katerega jo bomo uporabili. V diplomskem delu smo spoznali relacijske podatkovne baze ter predstavili njihove prednosti in lastnosti. Opisali smo tudi dve večji relacijski podatkovni bazi MySQL in DB2. Uvodoma smo naredili navodila za namestitev teh dveh podatkovnih baz, v nadaljevanju smo izvedli primerjavo med tema dvema podatkovnima bazama in merili hitrost izvajanja pri vstavljanju, brisanju in branju podatkov. Pridobljene rezultate smo predstavili v obliki tabel in grafov. Spoznali smo, da čeprav obe podatkovni bazi uporabljata enak jezik, vseeno obstajajo razlike v pisanju poizvedb in funkcij.

Diplomsko delo bo v pomoč uporabnikom, inženirjem in razvijalcem podatkovnih baz pri spoznavanju relacijskih podatkovnih baz in njihovih prednosti. Podjetja in organizacije se bodo lažje odločili med podatkovnima bazama MySQL in DB2. Ugotovili smo, da nobena od proučevanih podatkovnih baz ne zaostaja v razvoju ter, da obe poskušata biti konkurenčni tudi drugim novejšim podatkovnim bazam. Poudarili bi, da razvoj novih modelov podatkovnih baz ne smemo razumeti kot zavračanje relacijih podatkovnih baz

ampak za njihovo dopolnitev. Menimo, da še vedno obstaja bojazen pred odprtokodnimi različicami podatkovnih baz, predvsem za posameznike in manjša podjetja, zato bosta komercialni različici MySQL in DB2 še naprej povečevali svoj tržni delež.

Literatura

[1] Univerza v Mariboru. Podatkovna zbirka. Dosegljivo: http://studentski.net/gradivo/umb_fer_mk2_pbm_sno_zapiski_01?r=1. [Dostopano: 07. 08. 2017]].

[2] FRI. Podatkovna model. Dosegljivo: http://colos1.fri.uni-lj.si/ERI/RACUNALNISTVO/PODATKOVNE_BAZE/opredelitev_termina_podakovna_baza.html. [Dostopano: 11. 09. 2017].

[3] Podatkovni model. Dosegljivo: http://colos1.fri.uni-lj.si/ERI/RACUNALNISTVO/PODATKOVNE_BAZE/podatkovni_model.html. [Dostopano: 2017].

[4] Wikipedija. Zgodovina podatkovnih baz. Dosegljivo: http://www.quickbase.com/articles/timeline-of-database-history. [Dostopano: 11. 08. 2017].

[5] DB-Engines Ranking. Najbolj popularne pb. Dosegljivo: https://db-engines.com/en/ranking. [Dostopano: 07. 08. 2017].

[6] FRI. Komponente supb. Dosegljivo: https://ucilnica.fri.uni-lj.si/course/view.php?id=144. [Dostopano: 1. 11. 2017].

[7] E.F. Codd. A relational model of data for large shared data banks. Dosegljivo: http://webcache.googleusercontent.com/search?q=cache:2ku4yv-ZcMMJ:citeseerx.ist.psu.edu/viewdoc/download%3Fdoi%3D10.1.1.98.5286%26rep%3Drep1%26type%3Dpdf+&cd=3&hl=en&ct=clnk&gl=si&client=firefox-b-ab. [Dostopano: 12. 09. 2017].

[8] Carolyn Begg Thomas Connolly. *Database systems*. British Library Cataloguing-in-Publication Data, London, England, 2005.

[9] FRI. Sql jezik. Dosegljivo: `http://colos1.fri.uni-lj.si/ERI/RACUNALNISTVO/PODATKOVNE_BAZE/jeziki_in_prihodnost_podatkovnih_baz.html`. [Dostopano: 23. 09. 2017].

[10] FRI. Strukturirani poizvedovalni jezik sql. Dosegljivo: `www.fpp.uni-lj.si/mma_bin.php?id=2012120512352180`. [Dostopano: 23. 10. 2017].

[11] Wikipedija. Scid. Dosegljivo: `https://en.wikipedia.org/wiki/ACID`. [Dostopano: 11. 08. 2017].

[12] Oracle. Mysql. Dosegljivo: `https://www.mysql.com/`. [Dostopano: 13. 08. 2017].

[13] MySQL. Zgodovina mysql. Dosegljivo: `https://dev.mysql.com/doc/refman/5.7/en/history.html`. [Dostopano: 13. 08. 2017].

[14] Larry Ullman. *MySQL, Second Edition*. Peachpit Press, California, United States, 2008.

[15] Larry Ullman. *PHP and MySQL for Dynamic Web Sites, Fourth edition*. Peachpit Press, California, United States, 2016.

[16] TutorialsPoint. Db2. Dosegljivo: `https://www.tutorialspoint.com/db2/db2_introduction.htm`. [Dostopano: 01. 08. 2017].

[17] TechTarget. Db2. Dosegljivo: `http://searchdatamanagement.techtarget.com/feature/IBM-DB2-relational-DBMS-overview`. [Dostopano: 01. 08. 2017].

[18] Sam Lightstone George Baklarz Aamer Sachedina Paul Zikopoulos, Matthew Huras. *DB2 10.5 with BLU Acceleration*. McGraw Hill Companies, New York, United States, 2013.

[19] TechTarget. Trdi disk hdd. Dosegljivo: http://searchstorage.techtarget.com/definition/hard-disk-drive. [Dostopano: 13. 08. 2017].

[20] TechTarget. Ssd disk. Dosegljivo: https://en.wikipedia.org/wiki/Solid-state_drive. [Dostopano: 15. 10. 2017].

[21] http://www.mysqltutorial.org/php-mysql-create-table/http://www.mysqltutorial.org/php-mysql-create table/. Mysqltutorial. http://www.mysqltutorial.org/php-mysql-create-table/.

[22] Dbvisuliser. Dosegljivo: https://www.dbvis.com/. [Dostopano: 11. 08. 2017].

Dodatek A

Testna skripta MySQL

```
import com.mysql.jdbc.Connection;
import java.sql.SQLException;
import java.sql.*;

public class Diplomska_naloga {
      private static final String USERNAME = ”root”;
      private static final String PASSWORD = ”diplomska12”;
      private static final String CONN_STRING = ”jdbc:mysql://localhost:”
                  + ”3306/employees?autoReconnect=trueuseSSL=false”;

      public static void main(String[] args) {
        Connection conn = null;
        try {
           conn = (Connection) DriverManager.getConnection(CONN_STRING,
                  USERNAME, PASSWORD);
           System.out.println(”Povezava na podatkovno bazo vzpostavljena”);
           Statement povezava = (Statement) conn.createStatement();
           long zacetek = System.currentTimeMillis();
           ResultSet rezultat = povezava.executeQuery(“SELECT
```

```
            + "last_name, first_name, birth_date, title "
            + "FROM employees "
            + "JOIN titles ON employees.emp_no = titles.emp_no "
            + "WHERE employees.emp_no= 11170");
        while (rezultat.next()) {
          System.out.print(rezultat.getString("last_name") + " ");
          System.out.print(rezultat.getString("first_name") + " ");
          System.out.print(rezultat.getDate("birth_date") + " ");
          System.out.print(rezultat.getString("title") + " ");
        }
        long konec = System.currentTimeMillis();
        double cas = (konec - zacetek) * 0.001;
        System.out.println("Čas izvajanje poizvedbe je "+ (cas) + " s.");
        rezultat.close();
        povezava.close();
        conn.close();
      } catch (SQLException e) {
        System.err.println(e);
      } finally {
        try {
          if (conn != null)
              conn.close();
          }
        } catch (SQLException aa) {
        }
      }
    }
}
```

Testna skripta DB2

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.*;

public class DiplomskaNalogaDB2 {
    public static void main(String[] args) {
      try {
        Class.forName("com.ibm.db2.jcc.DB2Driver");
      } catch (ClassNotFoundException e) {
        System.out.println("Vnesi pot do JDBC driverja");
        e.printStackTrace();
        return;
      }       try {
        Connection conn = (Connection) DriverManager.getConnection
            ("jdbc:db2:employee", "db2admin", "diplomska12");
        long zacetek = System.currentTimeMillis();
        Statement povezava = (Statement) conn.createStatement();
        ResultSet rezultat = povezava.executeQuery
            ("SELECT last_name, birth_date, first_name, title "
            + "FROM employees "
           + "JOIN titles ON employees.emp_no = titles.emp_no "
           + "WHERE employees.emp_no= 11170");
        while (rezultat.next()) {
          System.out.print(rezultat.getString("last_name") + " ");
          System.out.print(rezultat.getString("first_name") + " ");
          System.out.print(rezultat.getDate("birth_date") + " ");
          System.out.print(rezultat.getString("title") + " ");
```

```
            }
            long konec = System.currentTimeMillis();
            double cas = (konec - zacetek) * 0.001;
            System.out.println("Čas izvajanje poizvedbe je "+ (cas) + " s.");
            rezultat.close();
            povezava.close();
            conn.close();
        } catch (SQLException e) {
            System.out.println("DB2 Database connection Failed");
            e.printStackTrace();
            return;
        }
    }
}
```