

# Baze Podataka

---

## UVOD

# Šta je Baza Podataka

- Nije bitno koji programski jezik ili API se koristi veštine o poznavanju baze podataka su vitalne.
- Baze podataka je teško izbeći kod razvoja aplikacija!!!
- Baza podataka nam daje strukturu tj. definiše pravila za čuvanje podataka.

Ime (tekst)	Prezime (tekst)	DatumZaposlenja (datum)	Ocena (broj)	Plata (valuta)
Marko	Ilić	09/01/2016	4	75000

**KOLONA**

**RED**

# Namena Baze Podataka

○ Baze podataka su dizajnirane da bi se rešili određeni problemi

## ❑ Veličina

○ Količina podataka sa kojom se radi

## ❑ Tačnost

○ Unos podataka da li prati određenu formu

## ❑ Bezbednost

○ Potrebno je da podaci budu dostupni i usled pada sistema

## ❑ Ažuriranje

○ Šta se dešava ukoliko više korisnika istovremeno pristupa bazi

## ❑ Redudatnost

Postoje više vrsta BP pri čemu se razlika ogleda u navedenim parametrima

# Podela Baze Podataka

---

HIJARARHIJSKI DBMS

MREŽNI DBMS

OBJEKTNO ORJENTISANI DBMS

## RELACIONI DBMS (RDBMS)

- ORACLE
- SQL SERVER
- DB2
- MySQL
- PostgreSQL
- SQLite
- MS ACCESS

## NoSQL DBMS

- MongoDB
- Cassandra
- CouchDB

Fokus stavljamo na **relacione baze** podataka

**Najrasprostranjenije baze podataka**

# Relacione Baze Podataka

---

Ime	Kompanija	Datum Nastanka	Admin Aplikacija	Licenca	Besplatna Verzija
Oracle	Oracle	1979	Oracle SQL Developer	Komercijalna	Express
SQL Server	Microsoft	1989	SQL Server Managment Studio	Komercijalna	Express
DB2	IBM	1983	IBM Data Studio	Komercijalna	Express C
MySQL	Oracle	1994	MySQL Workbench	Open Source	Community
...	...	...	...	...	...

Male baze koje se mogu instalirati na računarima ili Laptopovima do velikih baza koje se koriste u enterprise kompanijama

# Koncept RDBMS

## tbIPREDMETI

SifraPredmeta	Naziv	Datum	Učionica	Kapacitet	...
111	Baze Podataka	21/02/2017	4	35	
112	Mrežni Servisi	06/10/2016	206	30	
113	Razvoj Aplikacija	23/02/2017	8	45	

## tbIPRIJAVA

SifraPredmeta	SifraStudenta
111	50
112	51
111	52
113	50

## tbISTUDENTI

SifraStudenta	Ime	Prezime	Emai	...
50	Dušan	Ilić	di@vts.rs	
51	Jelena	Mitic	jm@vts.rs	
52	Darko	Mitov	dm@vts.rs	

# Koncept RDBMS

---

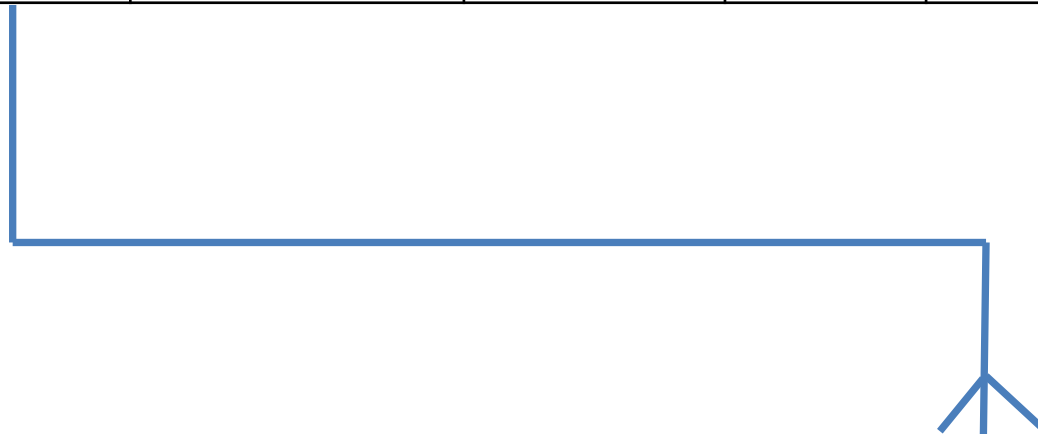
**tblMusterija**

SifraMusterije	Ime	Prezime	Adresa	Email	...
233	Milan	Stanko	SZK	ms@it.rs	
234	Dejan	Mitic	BB	dm@ni.rs	
235	Marina	Pantic	VK9	mp@vts.rs	

**tblPorudzbina**

SifraPorudzbine	Datum	Kolicina	Ukupno	SifraMusterije
101	01/05/2016	23	123	233
102	01/06/2016	10	345	235
103	01/05/2016	3	45	233

**Strani ključ  
Foreign Key**



# PRIKAZ PODATAKA IZ BP - SQL

---

```
SELECT Sime, Prosek, Odluka
FROM Student, Prijava
WHERE Student.Sid=Prijava.Sid AND
Vskole<500 AND
Smer="SRT" AND
Fime="VTS"
```

Sid	Sime	Prosek	Vskole
1	Marko	3.8	400
2	Darko	4.3	400
3	Jelena	4.9	620
4	Sonja	3.6	300

Student

Sid	Fime	Smer	Odluka
1	VTS	SRT	Primljen
1	ELFAK	SRT	Odbijen
2	ELFAK	RTI	Primljen
4	FON	Grafika	Odbijen

Prijava

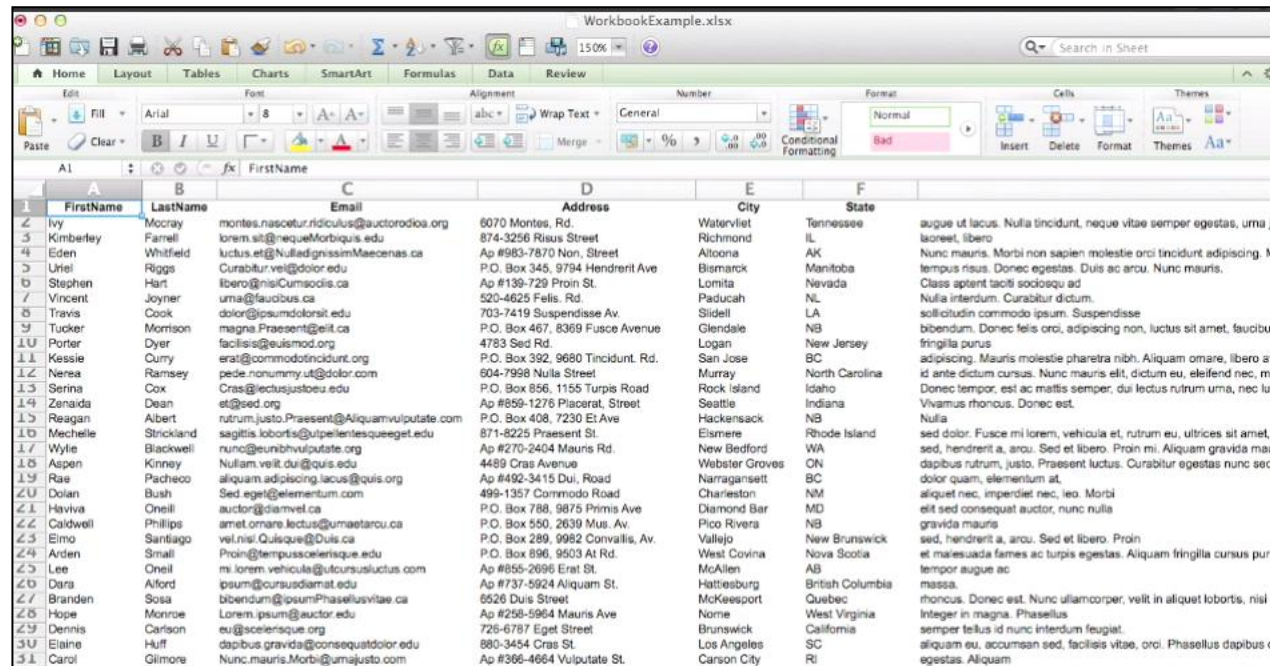
Fime	grad	BrojMesata
VTS	Nis	400
ELFAK	Nis	500
ETF	Bg	500
FON	BG	450

Fakultet



# Jedan od načina čuvanja podataka

- Baze podataka čuvaju podatke o:
  - Klijentima
  - Proizvodima
  - Porudžbenicama
  - Fajlovima
  - Slikama
  - Audio i Video sadržaj
  - Cenama
  - Datumima

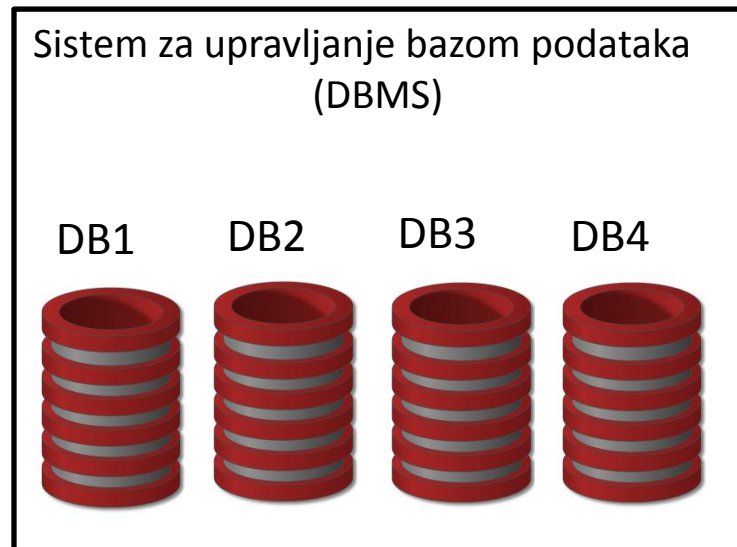


	A	B	C	D	E	F
	FirstName	LastName	Email	Address	City	State
2	Ivy	Mccray	montes.nascetur.ridiculus@auctorodios.org	6070 Montes, Rd.	Watervliet	Tennessee
3	Kimberley	Farnell	lorem.sit@nequeMortibus.edu	874-3256 Riusus Street	Richmond	IL
4	Eden	Whitfield	luctus.et@NulladgnissimMaecenas.ca	Ap #963-7870 Non, Street	Altoona	AK
5	Uriel	Riggs	Curabitur.vel@dolor.edu	P.O. Box 345, 9794 Hendrerit Ave	Bismarck	Manitoba
6	Stephen	Hart	libero@nisiCumsociis.ca	Ap #139-729 Proin St.	Lomita	Nevada
7	Vincent	Joyner	urna@faucibus.ca	523-4625 Felis, Rd.	Paducah	NL
8	Travis	Cook	dolor@ipsumdolorsit.edu	703-7419 Suspendisse Av	Slidell	LA
9	Tucker	Morrison	magna.Praesent@elit.ca	P.O. Box 467, 8369 Fusce Avenue	Glendale	NB
10	Porter	Dyer	facilisis@euismod.org	4783 Sed Rd.	Logan	New Jersey
11	Kessie	Curry	erat@commodotincidunt.org	P.O. Box 392, 9680 Tincidunt, Rd.	San Jose	BC
12	Nenea	Ramsey	pede.nonummy.ut@dolor.com	604-7998 Nulla Street	Murray	North Carolina
13	Serina	Cox	Cras@lectusjusto.eu.edu	P.O. Box 856, 1155 Turpis Road	Rock Island	Idaho
14	Zenaïda	Dean	et@sed.org	Ap #859-1276 Placerat, Street	Seattle	Indiana
15	Reagan	Albert	rutrum.justo.Praesent@Aliquamvulputate.com	P.O. Box 408, 7230 Et Ave	Hackensack	NB
16	Mechelle	Strickland	sagittis.lobortis@utpellenesqueget.edu	871-8225 Praesent St.	Eismere	Rhode Island
17	Wylie	Blackwell	nunc@eunibhulputate.org	Ap #270-2404 Mauris Rd.	New Bedford	WA
18	Aspen	Kinney	Nullam.vel.it.du@quis.edu	4489 Cras Avenue	Webster Groves	ON
19	Rae	Pacheco	aliquam.adipiscing.lacus@quis.org	Ap #492-3415 Dui, Road	Narragansett	BC
20	Dolan	Bush	Sed.egest@elementum.com	499-1357 Commodore Road	Charleston	NM
21	Haviva	O'Neill	auctor@diamvel.ca	P.O. Box 788, 9875 Primis Ave	Diamond Bar	MD
22	Caldwell	Phillips	amet.omnisi.lectus@umastarou.ca	P.O. Box 550, 2639 Mus, Av.	Pico Rivera	NB
23	Erino	Santiago	vel.nisi.Quisque@Duis.ca	P.O. Box 289, 9982 Convallis, Av.	Vailjejo	New Brunswick
24	Artlen	Smith	Proin@tempusscelerisque.edu	P.O. Box 896, 9503 At Rd.	West Covina	Nova Scotia
25	Lee	O'Neill	mi.lorem.vehicula@utcurtusluctus.com	Ap #855-2696 Egest St.	McAllen	AB
26	Dara	Alford	ipsum@coursusdiamat.edu	Ap #737-5924 Aliquam St.	Hattiesburg	British Columbia
27	Branden	Sosa	bibendum@ipsumPhasellusvitae.ca	6526 Duis Street	McKeesport	Quebec
28	Hope	Monroe	Lorem.ipsum@auctor.edu	Ap #258-5964 Mauris Ave	Norne	West Virginia
29	Dennis	Carlson	eu@scelesisque.org	726-6787 Egest Street	Brunswick	California
30	Elaine	Huff	dapibus.gravida@consequatdolor.edu	880-3454 Cras St.	Los Angeles	SC
31	Carol	Gilmore	Nunc.mauris.Morbi@umajusto.com	Ap #366-4664 Vulputate St.	Carson City	RI

# Sistemi za upravljanje bazama podataka

---

- Postoji ogromna razlika između DBMS-a i baze podataka
- DBMS je softver koji upravlja jednom ili više baza podataka koje mogu da imaju različita pravila, podatke, bezbednost,...
- Oracle
- SQL
- MySql
- PostGre
- MongoDB
- SQLite



# Desktop Database Sistemi

- Ove aplikacije nisu primarna meta profesionalaca.
- Koriste ih korisnici u velikim kompanijama (business users)



Office

beta

ME MY OFFICE PRODUCTS SUPPORT IMAGES TE

Search all of Office.com

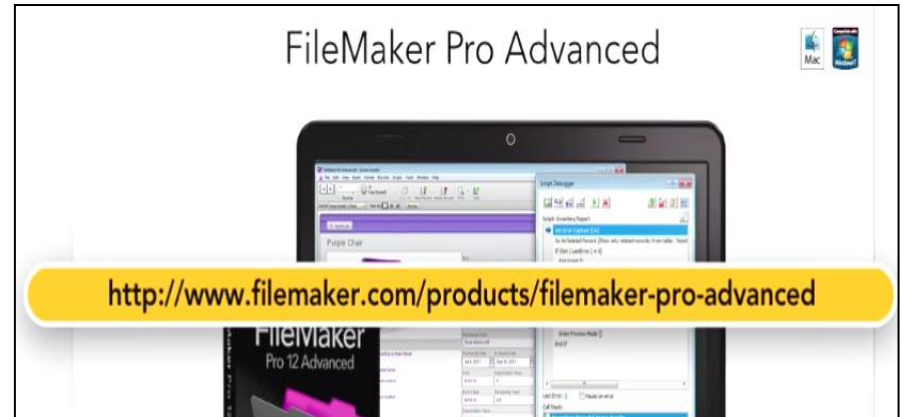
 **Access 2010**

Buy with Office Download a trial


When you need to do **more** than math

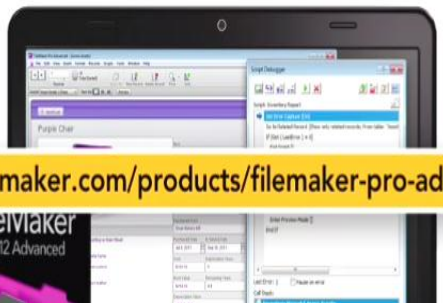
Target customer marketing, track inventory, plan events, and more.

What is Access? Get the basics >



FileMaker Pro Advanced





<http://www.filemaker.com/products/filemaker-pro-advanced>

FileMaker Pro 12 Advanced

# Desktop Database sistemi

---

## DESKTOP DATABASESE SISTEMI (+)

- + Jednostavna instalacija
- + Jednostavni za korišćenje
- + Unapred definisani templejti
- + Baza podataka + User Interface alati
- + Mogućnosti kreiranja izveštaja

## DESKTOP DATABASESE SISTEMI (-)

- Veliki broj korisnika
- Velika količina podataka
- Web baza podataka

# Desktop Database Sistemi - Zaključak

---

- Enterprise baze podataka ne obezbeđuju korisničku aplikaciju koja radi sa bazom podataka, već je takvu aplikaciju potrebno napraviti.
- Za desktop baze podataka najidealnije je ukoliko je koristi samo jedan korisnik
- U Accessu kada se kreira baza kreira se samo jedan fajl koji sadrži kompletnu bazu i on se ne može tako lako deliti kroz mrežu.
- Postoje tehnike koje to omogućavaju ali što je veći broj korisnika baza postaje manje pouzdana.

# Desktop Database Sistemi - Zaključak

---

- Access 2010 ima ograničenje na **250 istovremenih pristupa bazi** od strane korisnika
- U praksi ukoliko se javi potreba za **pristupom više od 10 korisnika** Desktop database sistemi se ne koriste.
- Vrlo je bitna aktivnost tih korisnika, mnogo je lakše podržati 20 korisnika koji samo čitaju bazu nego 10 koji istovremeno manipulišu nad podacima.
- Access nam dozvoljava simultano korišćenje baze od strane većeg broja korisnika ukoliko instaliramo Share Point
- Ukoliko pravimo Web sajt koji u pozadini ima bazu podataka desktop database sistemi nisu rešenje.
- Database sistemi su za interno korišćenje u malim ili srednjim sistemima za malu grupu korisnika i za rad sa malom količinom podataka.

# Relacioni DBMS

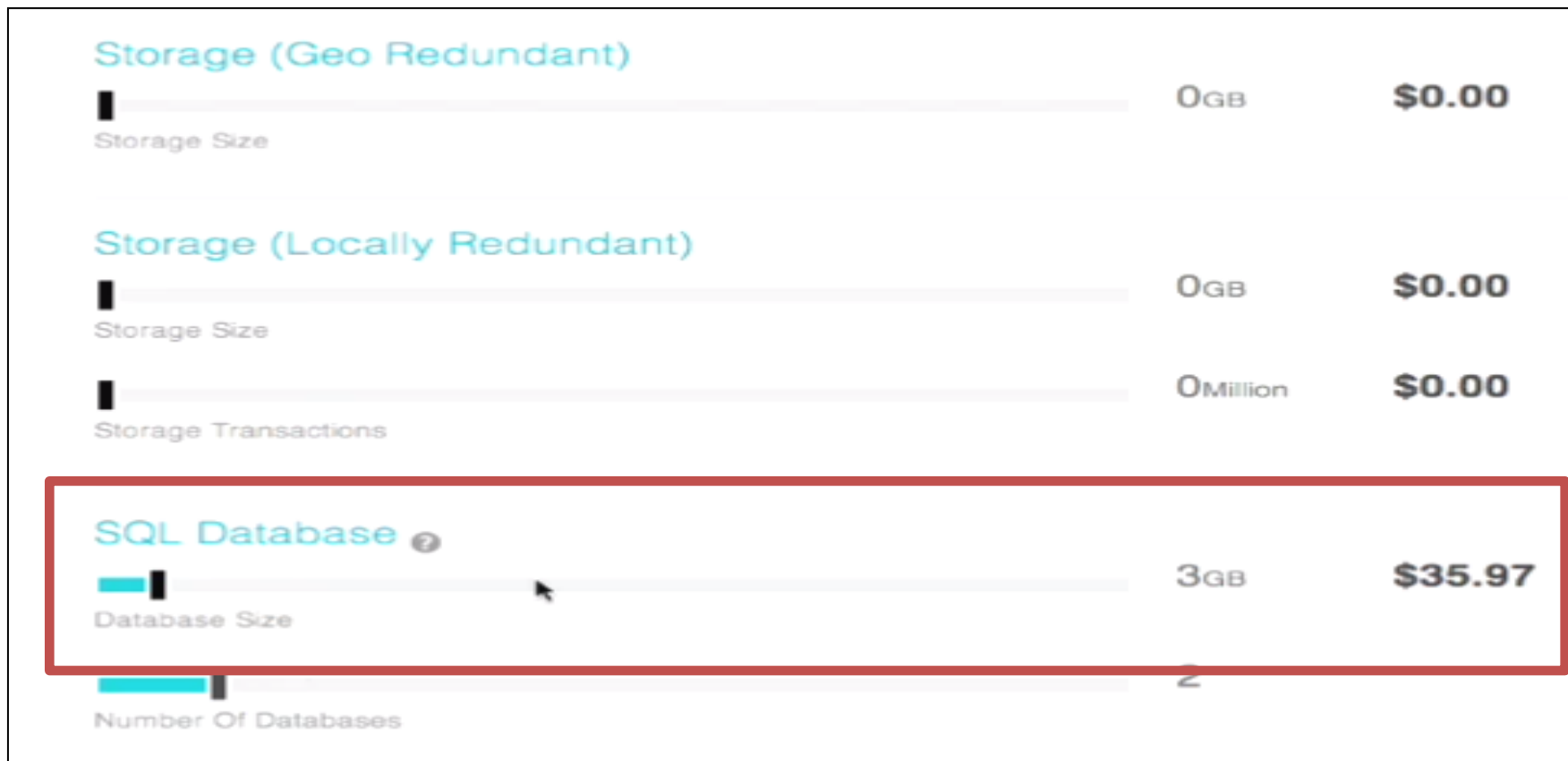
---

- Relacione baze podataka koriste koncept primarnog ključa na osnovu kojeg se kreira veza između tabela
- Koriste SQL za komunikaciju sa bazom
- RDB su najčešće korišćene baze podataka mada ne znači da su uvek i najbolje rešenje (kasnije)

Ime	Kompanija	Datum Nastanka	Admin Aplikacija	Licenca	Besplatna Verzija
Oracle	Oracle	1979	Oracle SQL Developer	Komercijalna	Express
SQL Server	Microsoft	1989	SQL Server Management Studio	Komercijalna	Express
DB2	IBM	1983	IBM Data Studio	Komercijalna	Express C
MySQL	Oracle	1994	MySQL Workbench	Open Source	Community
...	...	...	...	...	...

# Windows Azure SQL server u Cloud-u

---



**PLAĆAMO BRIGU OKO TEHNIČKE INFRASTRUKTURE  
DIZAJN BAZE I APLIKACIJE RADI KORISNIK**



# AMAZON RDS

## Amazon Relational Database Service Pricing

aws.amazon.com/rds/pricing/

stem | Performance...

Oracle Database Express Edition...

IBM – DB2 Express–C

Windows Azure Pricing Calculato...

Amazon Relational



Sign Up

My Account / Console

English

AWS Products & Solutions

AWS Product Information



Developers

Support

### Amazon RDS

- Amazon RDS Overview
- Features
- FAQs
- Pricing**
- Reserved Instances

### Supported DB Engines

- MySQL
- Oracle
- SQL Server

## Amazon Relational Database Service Pricing

MySQL

Oracle

SQL Server

Pay only <sup>active</sup> what you use. There is no minimum fee. Estimate your monthly bill using the [AWS Simple Monthly Calculator](#).

### On-Demand DB Instances

On-Demand DB Instances let you pay for compute capacity by the hour your DB Instance runs with no long-term commitments. This frees you from the costs and complexities of planning, purchasing, and maintaining hardware and transforms what are commonly large fixed costs into much smaller variable costs.

### Standard Deployment

The pricing below applies to a standard DB Instance or Read Replica deployed in a single Availability Zone.

Region:

# Express Edicije



Microsoft  
SQL Server

About SQL Server Solutions & Technologies Editions Get SQL Server Learning Center Partners

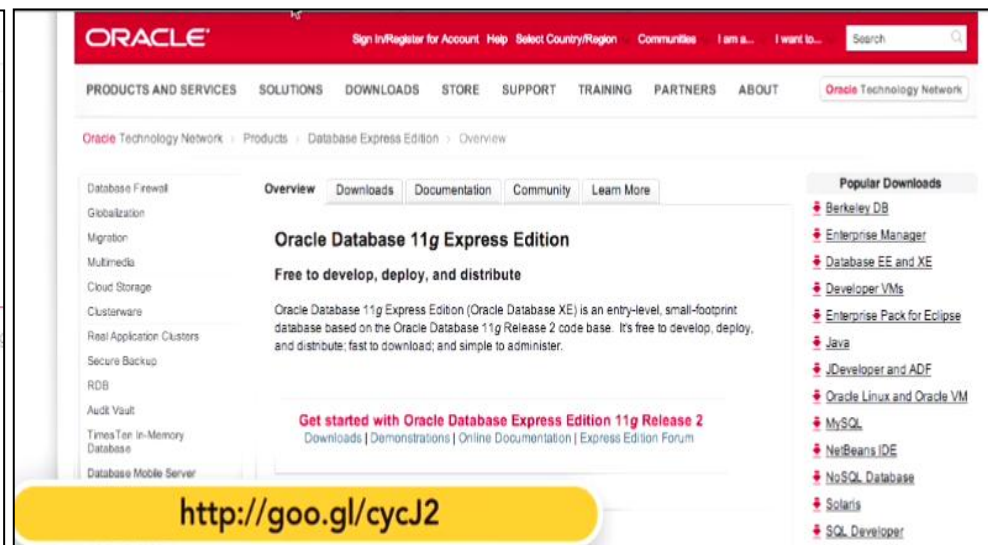
## Express Edition

SQL Server Express is a free edition of SQL Server ideal for developing and powering desktop, web and small server applications.

[Download SQL Server 2012 Express](#)

<http://goo.gl/yGMUE>

Easy to get started. Free to use.



ORACLE

Sign In/Register for Account Help Select Country/Region Communities I am a... I want to... Search

PRODUCTS AND SERVICES SOLUTIONS DOWNLOADS STORE SUPPORT TRAINING PARTNERS ABOUT Oracle Technology Network

Oracle Technology Network > Products > Database Express Edition > Overview

Database Firewall Globalization Migration Multimedia Cloud Storage Clusterware Real Application Clusters Secure Backup RDB Audit Vault TimesTen In-Memory Database Database Mobile Server

Overview Downloads Documentation Community Learn More

## Oracle Database 11g Express Edition

Free to develop, deploy, and distribute

Oracle Database 11g Express Edition (Oracle Database XE) is an entry-level, small-footprint database based on the Oracle Database 11g Release 2 code base. It's free to develop, deploy, and distribute: fast to download; and simple to administer.

[Get started with Oracle Database Express Edition 11g Release 2](#)  
Downloads | Demonstrations | Online Documentation | Express Edition Forum

**Popular Downloads**

- Berkeley DB
- Enterprise Manager
- Database EE and XE
- Developer VMs
- Enterprise Pack for Eclipse
- Java
- Developer and ADF
- Oracle Linux and Oracle VM
- MySQL
- NetBeans IDE
- NoSQL Database
- Solaris
- SQL Developer

<http://goo.gl/cycJ2>



IBM

United States | change | Search

Home Solutions Services Products Support & downloads My IBM Welcome | IBM Sign In | Register

Information Management > Data Management > DB2 Product Family > DB2 for Linux, Unix and Windows

## DB2 Express-C

Free to develop, free to deploy, free to distribute

Information Management

Highlights

- Free DB2 course with videos and exercises on Big Data University
- FREE Book: Getting Started with DB2 Express-C
- ChanneDB2 Community
- PlanetDB2 Blogs

IBM SmartCloud

### DB2 on IBM SmartCloud

DB2 database software offers industry leading performance, scale, and reliability on your choice of platform from Linux.

Free Database! New Version!  
→ Download v10.1.2

Get Help on the Online Forum  
→ Learn more

Full-function relational and XML data server  
Simple, flexible, powerful, and reliable

Download and deploy at no charge  
Support available for added peace-of-mind

<http://goo.gl/N9wbe>

- Express edicije su obično besplatna rešenja komercijalnih baza
- Express edicije nemaju sve funkcionalnosti koje ima komercijalno rešenje
  - Veličina baze nije veća od 4GB
  - DBMS može da koristi samo jedan CPU čak i ukoliko je instalirana na mašini koja ima više procesora
- Express edicije su korisne u pogledu učenja

# XML DB Sistemi

---

- XML je format za razmenu podataka, shodno tome postoje DBMS koji su orijentisani eksplicitno na korišćenje XML-a.
- To su DBMS sistemi koji imaju tunel strukturu u odnosu na tabele i kolone
- XML DB Sistemi (BaseX,Sedna,eXist) čuvaju XML direktno koji uključuje veliku količinu ugnježdenih XML dokumenata
- Xquery se koristi za pretragu XML dokumenata

Ime	Licenca	Query Jezik
BaseX	Open Source	XQuery
Sedna	Open Source	XQuery
eXist	Open Source	XQuery

```
<?xml version="1.0"?>
<library>
  <course id="fop003">
    <author>Allardice, Simon</author>
    <title>Foundations of Programming: Databases</title>
    <genre>Developer</genre>
    <date_published>2013-01-30</date_published>
    <description>Getting started with databases and database technologies.</description>
  </course>
  <course id="java001">
    <author>Gassner, David</author>
    ...
```

# RDBMS XML Podrška

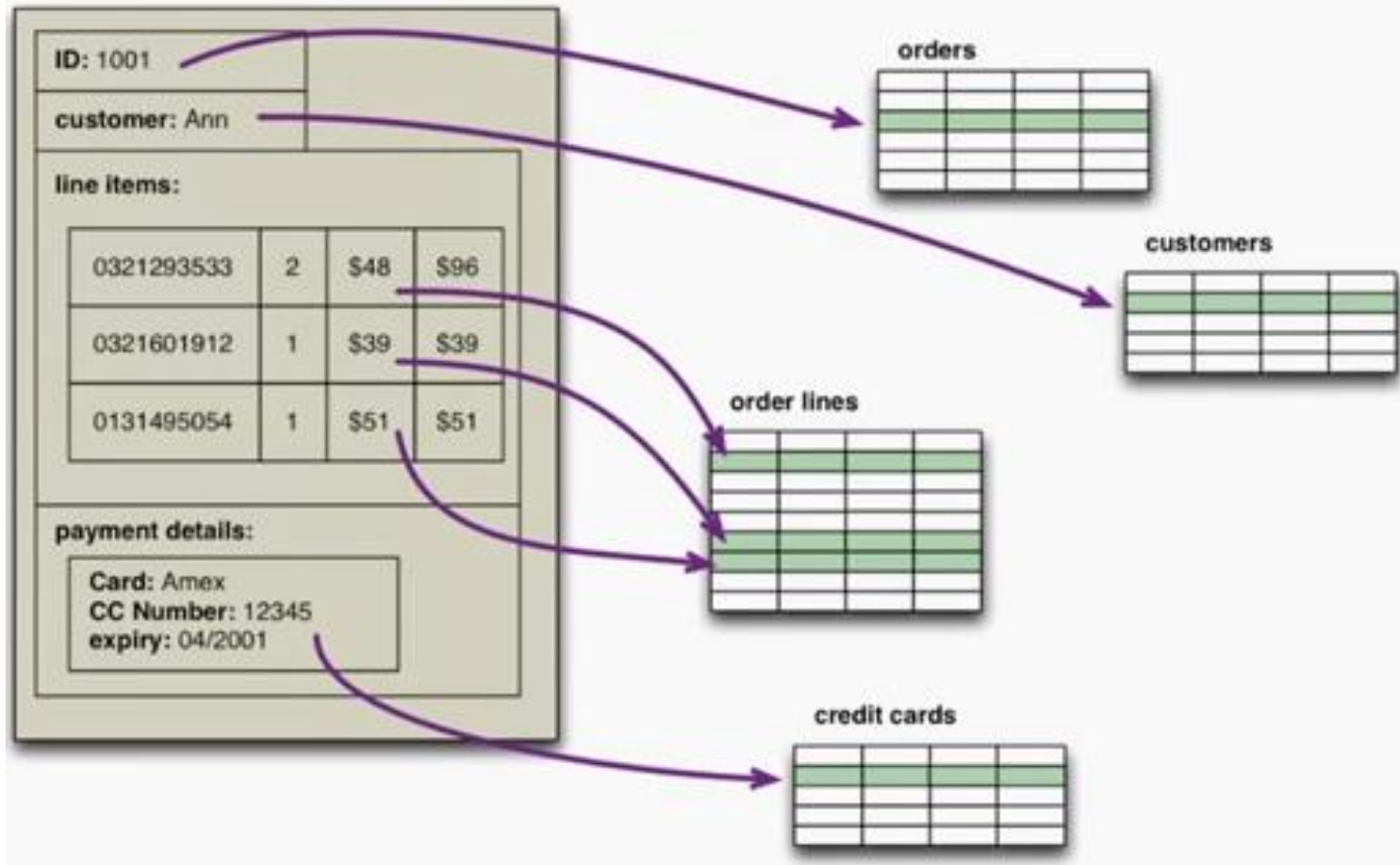
- Relacione baze podataka imaju podršku za XML.
- XML se može čuvati u tabele jer je XML ustvari tekst.
- Većina RDBMS sistema ima tip podataka za XML
- Podrška za XML tip podataka znači da db sistem može da koristi Xquery nad XML kolonama

Ime	XML Tip kolone
Oracle	Da
DB2	Da
SQL Server	Da
MySQL	Ne (čuva se kao text)

CourseID	Details
3	<?xml version="1.0"?><library><course id="fop003"><author>Allardice, Simon</author><title>Foundations of Programming: Databases</title>...

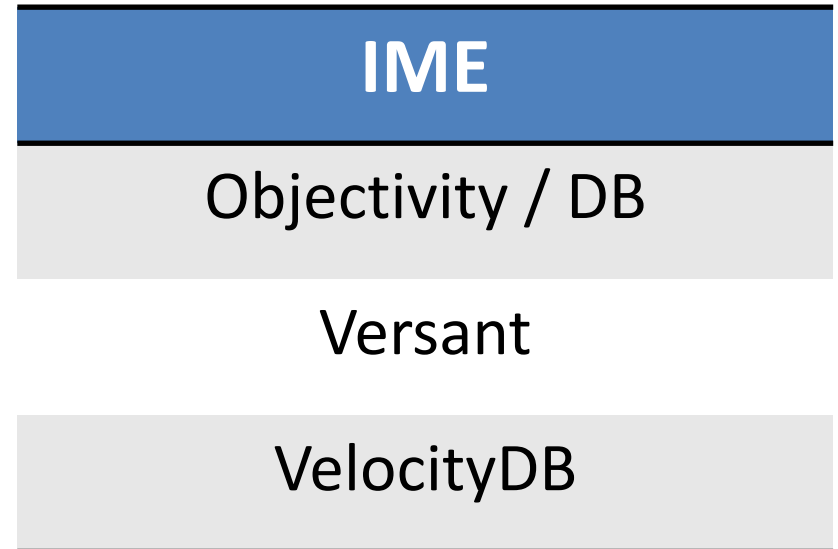
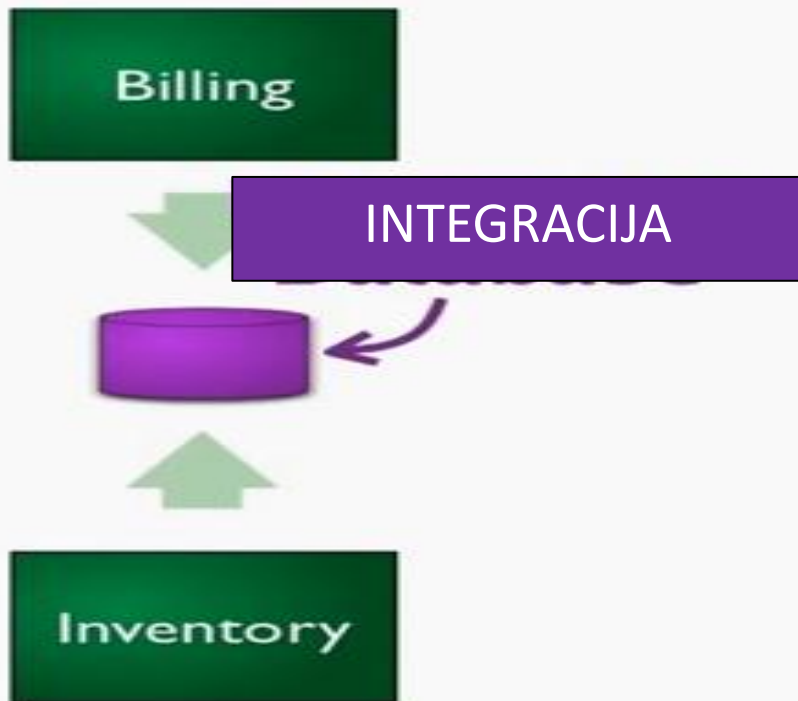
# Objektno Orjentisani DB sistemi

- Osnovni nedostatak DBMS relacionih baza podataka je razvoj objektno orjentisanih programskih jezika koji rade sa objektima koji ne mogu da se mapiraju jednostavno u tabele i kolone.



# Objektno Orjentisani DB sistemi

- Tokom godina su se razvili objektno orjentisani db sistemi, međutim kao i XML DB sistemi oni nikad nisu dostigli popularnost RDBMS.
- Jedan od razloga zasto objektno orjentisane baze podataka nisu iskoristile svoj potencijal jer je taj period predstavljao dominatnost relacionih baza podataka a SQL je postao mehanizam integracije jer su mnogi developeri ugradili SQL kroz svoje aplikacije



# Objektno orjentisano mapiranje (ORM)

---

- Sprega između objektno orjentisanog jezika i relacione baze podataka je ORM koja je izuzetno popularna.
- ORM je softver(framework) koji mapira objekte iz objektno orjentisanog jezika u tabele relacione baze podataka i obrnuto ne razmišljajući mnogo o mapiranju

ORM Framework	Jezik
Hibernate	Java
Core Data	Objective -C
ActiveRecord	Ruby
NHibernate	C# / VB.NET

# NoSql DB Sistemi

---

CouchDB  
MongoDB  
Apache Cassandra  
Hypertable  
HBase  
Neo4J  
BigTable  
Riak  
Project Voldemort  
Redis

## **BITNO!!!!**

- Između NoSql DB sistema postoji velika razlika
- Znanje MongoDB-a ne znači da znate Neo4J što nije slučaj kod RDBMS



# Karakteristike NoSql baze

- Obično ne koriste SQL
- Ne zasniva se na tabelama
- Obično se ne zasniva na relacijama
- Obično ne koristi formalnu šemu
- Ne koristi ACID koncept kod transakcija
- Orjentisana je ka Web aplikacijama
- Orjentisana je ka rešenjima velikih razmera
- Često su Open source
- Cluster friendly

# Kategorizacija NoSql baza

## Document Store Baze

Čuvaju dokumenta u formatu koji ne zahteva kreiranje redova i kolona sa unapred definisanim tipom sadržaja

Dokumenat koji se čuva ima svoj format tj. šemu

Format koji se čuva može biti XML ili JASON.

Svaki podatak koji se čuva može se opisati svojom šemom

Primer dokumenta koji može da se čuva

```
{
  "LastName" : "Brown",
  "FirstName" : "Michelle",
  "Email": [
    {
      "type": "home",
      "number": "michelleb@...com"
    },
    {
      "type": "work",
      "number": "mbrown@acme...com",
      "verified": false
    }
  ],
  "DateHired": "02-17-2009",
  "Department": "Production"
}
```

ing. Databases",

en

cij

integer

Drugi tip dokumenta koji želimo da sačuvamo izgleda potpuno drugačije i sadrži ugnježdene dokumente.

# Kategorizacija NoSql baza

## Document Store Baze

- Svaki dokument koji se čuva ima jedinstven ID
- Imamo potpunu fleksibilnost jer ne moramo da obezbedimo šemu i ne koristimo relacije
- Primeri ovih baza su **MongoDB** i **CouchDB**
- XML baze spadaju u ovu kategoriju

# Kategorizacija NoSql baza

## Key-value Stores Baze

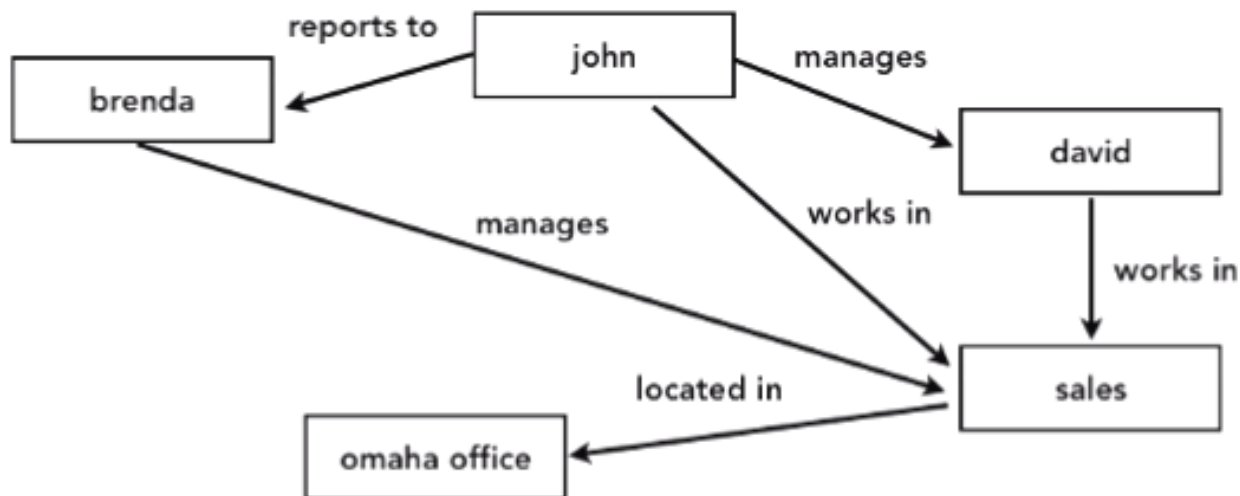
- Sve se čuva kao par key-value(ključ-vrednost)
- Nemaju unapred definisanu šemu
- Čuva i vraća sve na osnovu para kay-value
- Baze koje koriste ovaj koncept su **MemcacheDB, Riak, Project Voldemort**
- Možemo da stavimo šta god želimo za vrednost(JSON, XML,string, integer, binarni podatak,...), ima dosta sličnosti sa MongoDB i CouchDB konceptom

key	value
name1	bob
color	red
company4534_name	Microsoft
company4556_name	Apple
course_43fe6fe	{"title" : "Foundations of Programming: Databases", "rating":10}
u473642_photo	(binary data)
...	...

# Kategorizacija NoSql baza

## Graph Baze

- Koristi se struktura grafa
- Sve se čuva u formi malih povezanih čvorova koji su u relaciji
- Ne postoji master point i sve je povezano sa svim ostalim
- Za razliku od drugih NoSql baza koje ne koriste relacije sa ovom kategorijom je drugačije
- Primeri ovih baza su **Neo4J, AllegroGraph, DB2 NoSQL Graph Store**



# Razlozi za odabir NoSql bazu

## 1. Da li vam je potrebna fleksibilna šema?

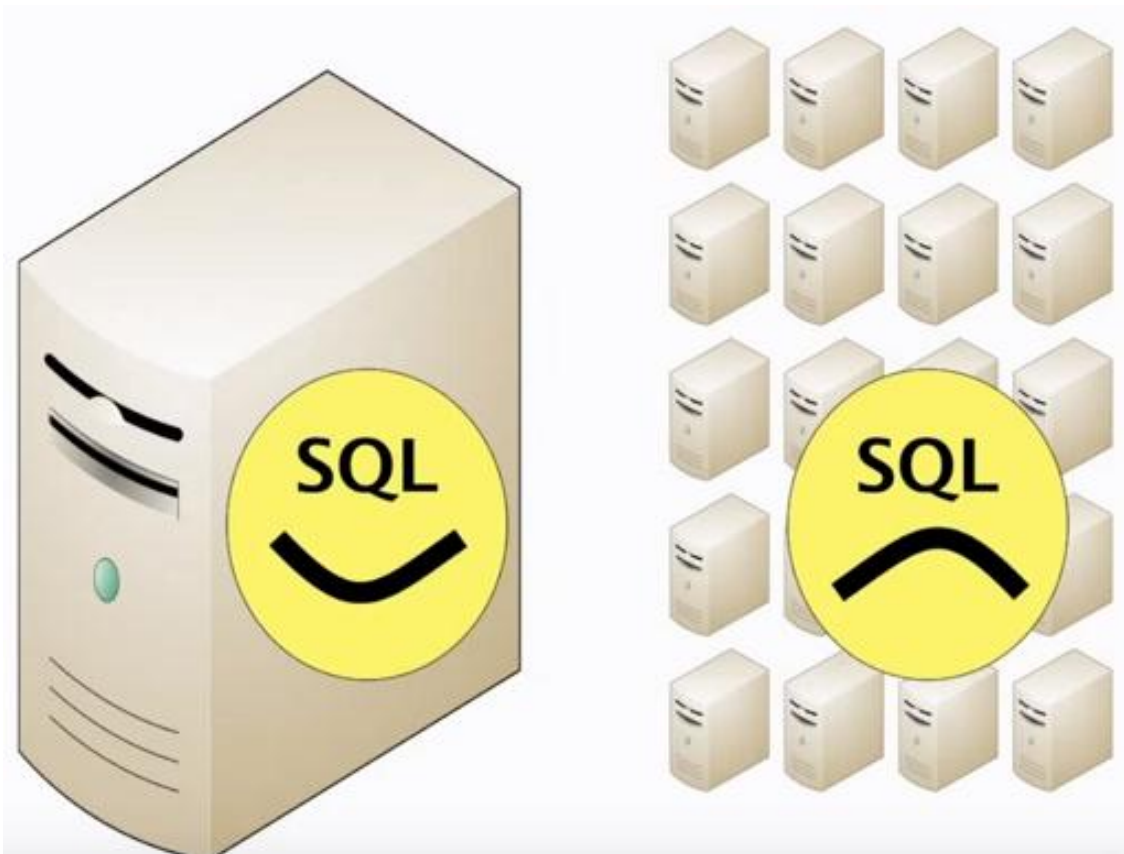
- a) Ukoliko podatke koje primamo se uvek razlikuju tj. ne možemo da napravimo jedinstvenu šemu

## 2. Da li imate ogromnu količinu podataka?

- a) Više od 100 miliona redova, za tabele koje imaju nekoliko miliona ili manje RDBMS je možda bolje rešenje

# NoSql baze

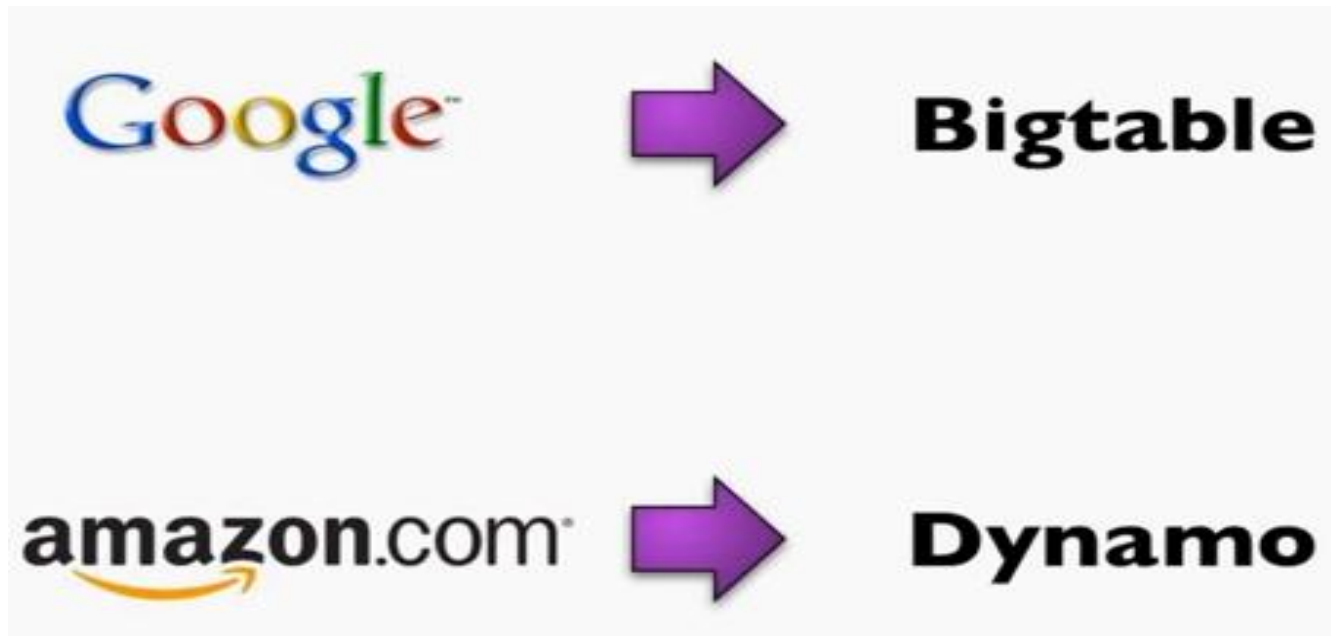
- Promenu u strukturi baza podataka je izazvalo sve više sajtova na Internetu i ogromna količina podataka koja se nalazila na tim sajtovima (Amazon, Google,...)
- Zbog velike količine podataka javila se potreba za skaliranjem
- SQL nije pokazao dobre performanse u radu sa klasterima što su veliki igrači i testirali



# NoSql baze

Velike kompanije su razvile svoj data sistem za skladištenje koji se prilično razlikovao od relacionih baza podataka.

Ovaj korak je predstavljao inspiraciju za pojavu NoSql baze podataka



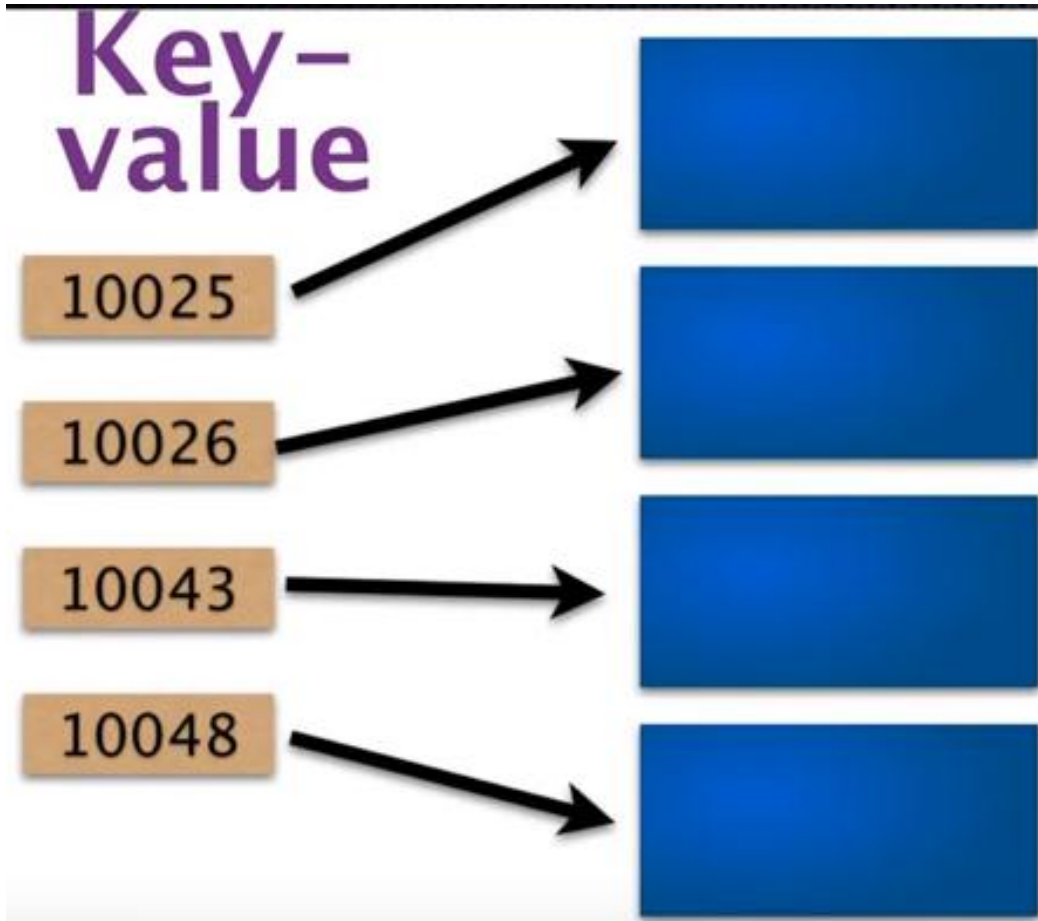


# NoSql baze

- NoSql baze korite DATA model za razliku od RDBMS koje koriste relacioni model
- NoSQL baze se mogu podeliti u 4 kategorije na osnovu DATA modela koji koriste



# Key Value Store Model



# Document Data model

## Document

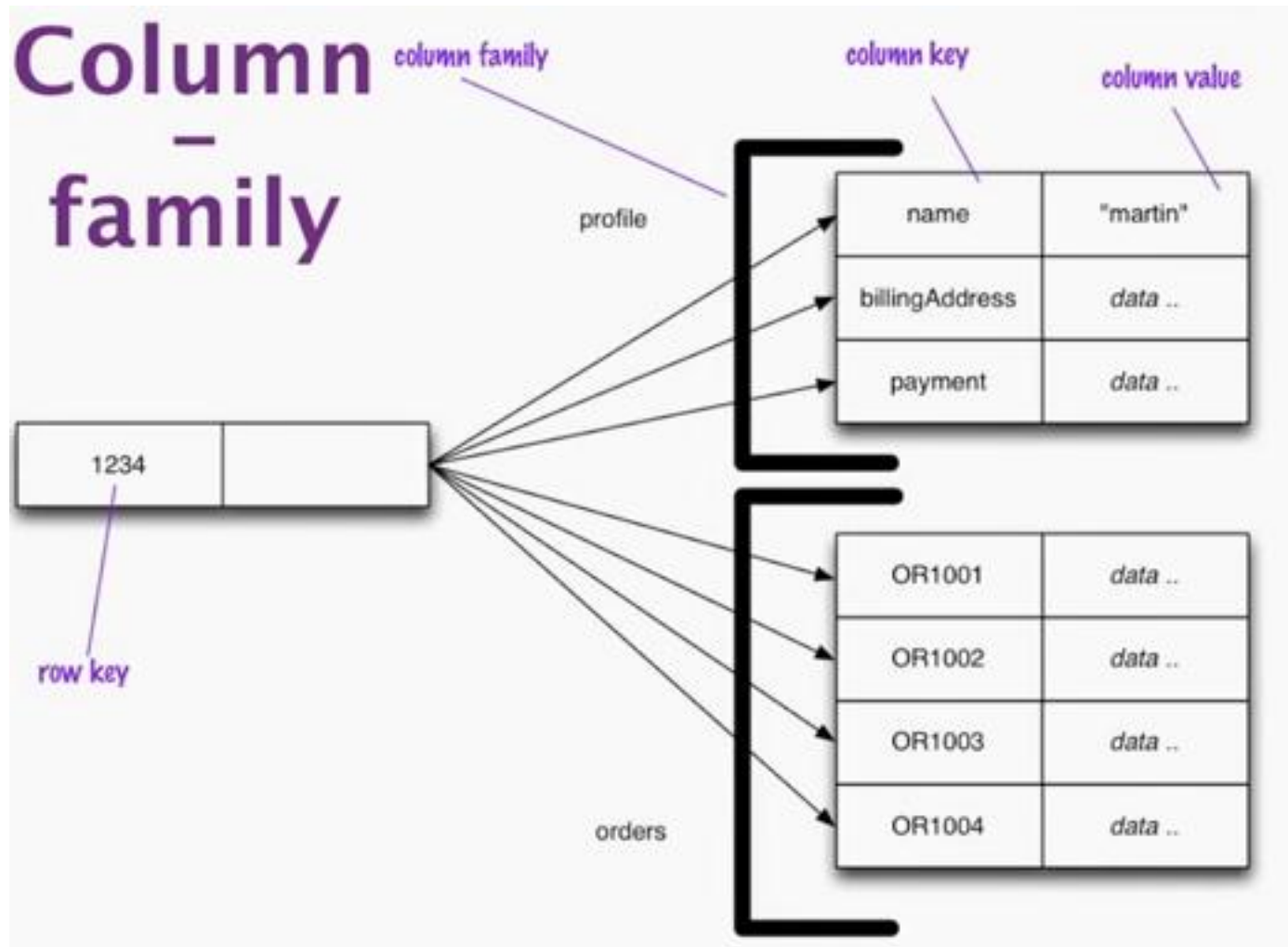
```
{ "id": 1001,  
  "customer_id": 7231,  
  "line-itmes": [  
    { "product_id": 4555, "quantity": 8 },  
    { "product_id": 7655, "quantity": 4 }, { "product_id": 8755,
```

```
    "product_id": 2155,  
  "discount-code": "Y" }
```

**no  
schema**

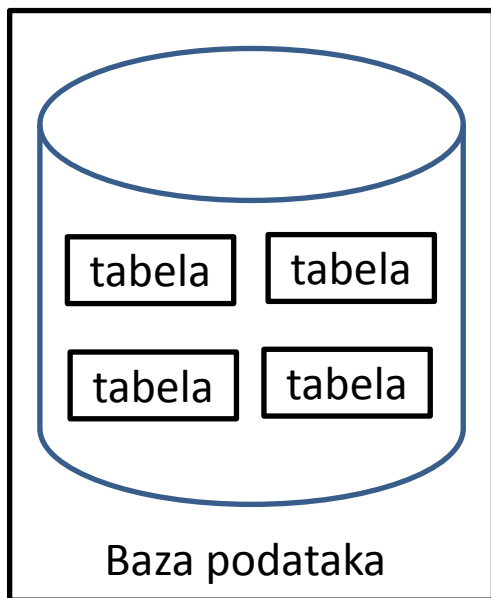


# NoSql baze

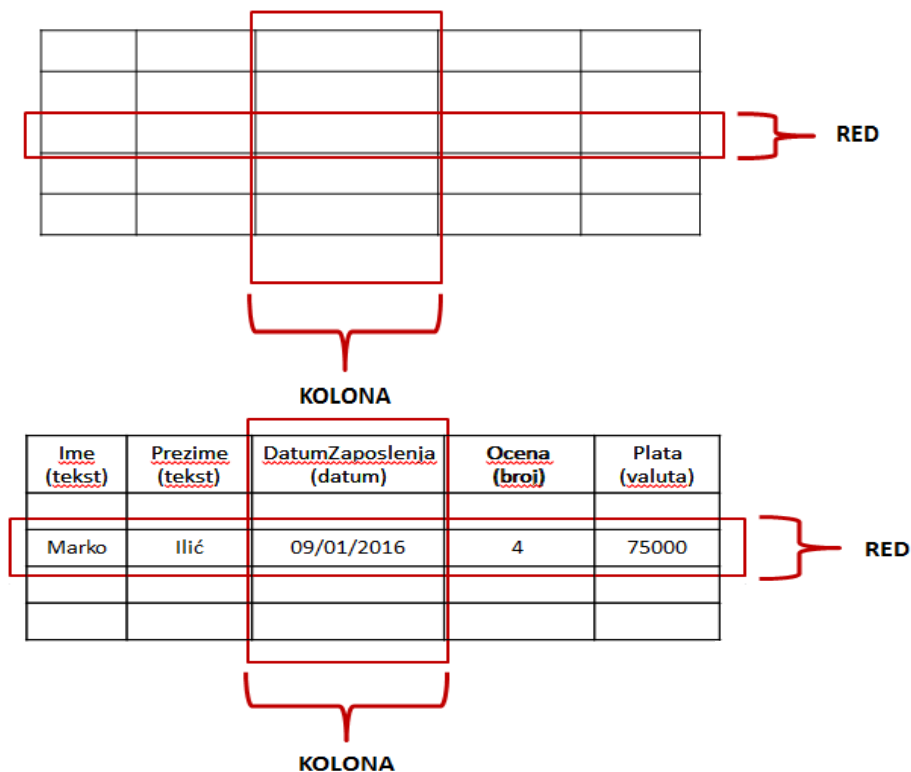


# STRUKTURA RELACIONE BP (RDBMS)

- Baza podataka sastoji se iz jedne ili više tabela
- Tabela je osnovni blok baze podataka



- Svaka tabela se sastoji iz kolona i vrsta
- Svaki red u tabeli predstavlja jednog kupca ili jednog studenta ili jednog zaposlenog,...
- Svaka kolona opisuje ime i tip podataka koji se čuva



# STRUKTURA TABELE

---

- Svaki red u tabeli mora da poštuje strukturu format upisa podataka
- Ako tabela predstavlja radnike:
  - svaki red predstavlja jednog radnika
  - svaka kolona predstavlja specifičnu informaciju o radniku

Ime (tekst)	Prezime (tekst)	DatumZaposlenja (datum)	Ocena (broj)	Plata (valuta)
Jelena	Mitić	01/01/2015	3.5	56000
Marko	Ilić	09/01/2016	4	75000
Danijel	Perić	23/05/2010	3.8	64000
Ana	Pešić	13/08/2012	4.2	80000

# STRUKTURA TABELE

---

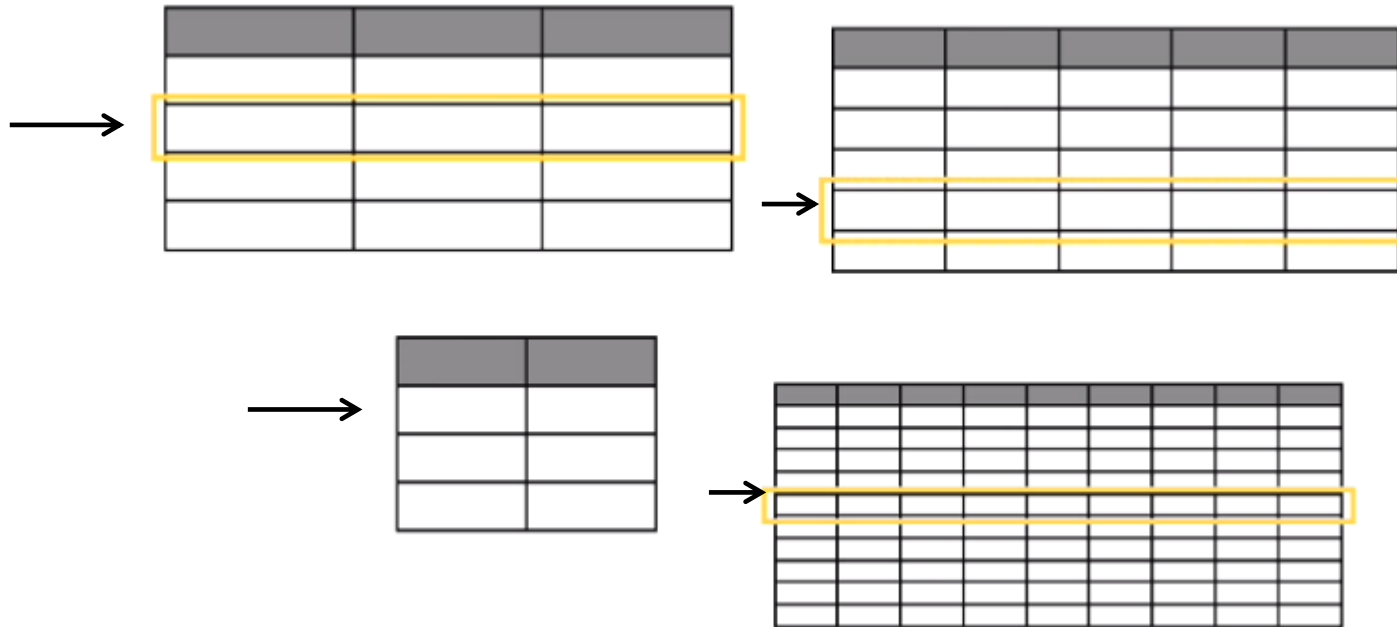
- Tabela se kreira tako što se prvo definišu kolone i ograničenja za kolone a kasnije se unose redovi
- Kada definišemo pravila DBMS traži od nas da ih se i pridržavamo.
  - Ako je kolona Datum Zaposlenja definisana kao datum, DBMS nam neće dozvoliti unos bilo kog podataka

Ime (tekst)	Prezime (tekst)	DatumZaposlenja (datum)	Ocena (broj)	Plata (valuta)
Jelena	Mitić	01/01/2015	3.5	56000
Marko	Ilić	09/01/2016	4	75000
Danijel	Perić	23/05/2010	3.8	64000
Ana	Pešić	13/08/2012	4.2	80000

# STRUKTURA TABELE – IDENTIFIKACIJA REDA

---

- Skoro sve tabele u bazi zahtevaju ključ
- Ključ nam omogućava da identifikujemo željeni red u tabeli
- Ključ je jedna kolona u tabeli čije vrednosti moraju da budu jedinstvene





# STRUKTURA TABELE – IDENTIFIKACIJA REDA

---

- Kolona koja je definisana kao unique koliko god redova da sadrži i koji god tip podataka da koristi (text, broj, datum) nesme da ima istu vrednost jer nam DBMS neće to dozvoliti.
- Kolone koje nisu jedinstvene a većina je takvih mogu da sadrže iste podatke

unique	not unique
153	100
374	100
352	200
376	250
<del>153</del>	100

# PRIMARNI KLJUČ

---

- U svakoj tabeli potrebno je odrediti jedinstvenu kolonu koja može biti kandidat za primarni ključ
  - Ime Radnika ?
  - Datum Zaposlenja ?
  - Broj Socijalnog osiguranja
  - Id Radnika

## PRIMARNI KLJUČ (PK)

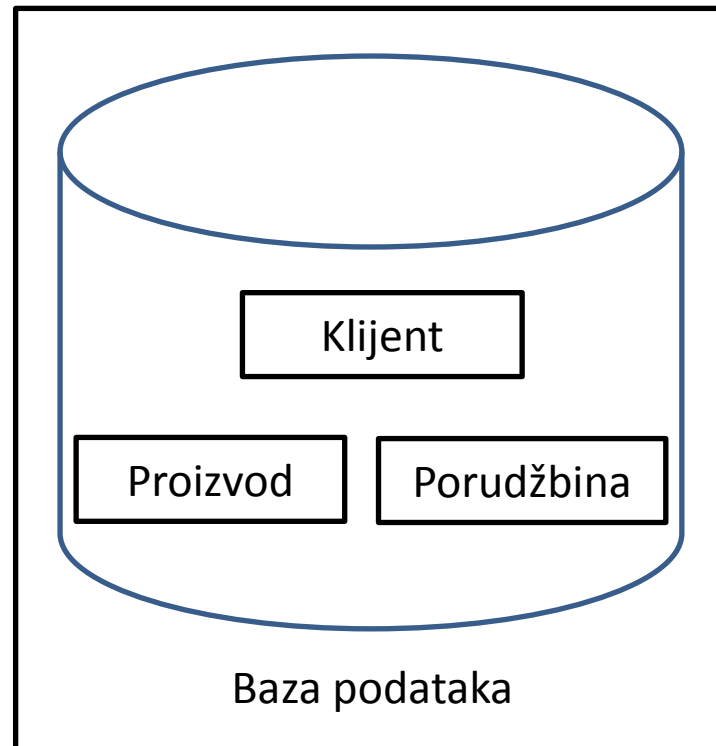
Ime (tekst)	Prezime (tekst)	DatumZaposlenja (datum)	Ocena (broj)	Plata (valuta)
Jelena	Mitić	01/01/2015	3.5	56000
Marko	Ilić	09/01/2016	4	75000
Danijel	Perić	23/05/2010	3.8	64000
Ana	Pešić	13/08/2012	4.2	80000

- Često se koristi generička kolona za primarni ključ koja na jedinstven način identifikuje svaki red u tabeli.
- Sistem generiše vrednosti za ovu kolonu i ona je vidljiva samo sistemu a ne i korisniku.

# VEZE IZMEĐU RELACIJA

---

- Potrebno je opisati vezu između tabela.
- Tabele sadrže različite informacije, međutim između određenih tabela postoji veza koju je potrebno definisati
- Veza između tabela zasniva se na **ključevima**



# VEZE IZMEĐU RELACIJA

---

## PRIMARNI KLJUČ (PK)

SifraMusterije	Ime	Prezime	Adresa	Email	...
233	Milan	Stanko	SZK	ms@it.rs	
234	Dejan	Mitic	BB	dm@ni.rs	
235	Marina	Pantic	VK9	mp@vts.rs	

## PRIMARNI KLJUČ (PK)

SifraPorudzbine	Datum	Kolicina	Ukupno
101	01/05/2016	23	123
102	01/06/2016	10	345
103	01/05/2016	3	45

Svaka porudžbina je porudžbina  
za određenog klijenta

# VEZE IZMEĐU RELACIJA

## PRIMARNI KLJUČ (PK)

SifraMusterije	Ime	Prezime	Adresa	Email	...
233	Milan	Stanko	SZK	ms@it.rs	
234	Dejan	Mitic	BB	dm@ni.rs	
235	Marina	Pantic	VK9	mp@vts.rs	

## PRIMARNI KLJUČ (PK)

SifraPorudzbine	Datum	Kolicina	Ukupno	SifraMusterije
101	01/05/2016	23	123	233
102	01/06/2016	10	345	235
103	01/05/2016	3	45	233

## STRANI KLJUČ FOREIGN KEY (FK) nije jedinstven

- Strani ključ ne mora da bude jedinstven
- **Relacija 1:M**

# VEZA 1:M

---

- 1: M pravilo znaci da jedan klijent moze da naruči više proizvoda ali i ne mora.
- Klijent može da ima samo jednu porudbinu ili da uopšte nema porudbinu.
- Radi se o klijentu koji se samo registrovao ali nista nije naručio
- Sa druge strane 1 porudbinu može da ima samo jedan klijent

# TRANSAKCIJE

---

- Transakcije su vrlo bitne u svetu baza podataka
- PRIMER transakcije u realnom svetu
  - Ukoliko prodavcu knjiga platimo, očekujemo da dobijemo knjigu.
  - Transakcija zahteva da se obe stvari izvrše
  - Ukoliko smo dali novac ocekujemo knjigu, ukoliko prodam knjigu očekujem novac za nju
- U svetu računara klasičan primer transakcije je bankarski sistem.
  - Ulogujemo se na naš račun preko Weba i želimo da prebacimo novac sa jednog računa na drugi.
  - Ova akcija zahteva dve promene nad podacima, sa jednog računa oduzimamo novac a na drugom računu dodajemo novac.
  - Transakcija je uspešna ukoliko su se obe stvari izvršene, ukoliko jedna stvar ne uspe, sistem vraća podatke na početno stanje

BrojRacuna	TipRacuna	StanjeRacuna	...
160-123-3345	Štednja	10000 RSD	

BrojRacuna	TipRacuna	StanjeRacuna	...
160-123-3311	Tekući	1000 RSD	

**transakcija**

- \$200

+ \$200

# TRANSAKCIJE

---

- Termin koji se vrlo često koristi u radu sa transakcijama je ACID i ugrađen je u DBMS
  - **Atomic**
    - Zahteva da se sve akcije u transakciji izvrše ili se sistem vraća na originalno stanje.
    - Razlog zašto se sve akcije u transakciji ne izvrše je nestanak električne energije ili nedovoljno prostora ili ...
    - Atomic znači sve ili ništa
  - **Consistent**
    - Transakcijom baza podataka iz jednog validnog stanja prelazi u drugo na osnovu pravila u bazi
  - **Isolated**
    - Odnosi se na to da podatak koji je uključen u transakciji bude zaključan za vreme trajanja transakcije, tj. ne sme da se dozvoli drugom sistemu da menja isti podatak
  - **Durable**
    - Garantuje izvršenje transakcije čak i ukoliko se desi neki otkaz



# SQL (Structured Query Language)

---

- SQL je jezik koji se primenjuje kod svake relacione baze podataka
- SQL je razvijen 70. godina i danas se koristi
- SQL je deklarativni programski jezik i razlikuje se od proceduralnih i objektnih jezika

IdKnjige	Naslov	DatumPublikacije	Cena	...
100	Dizajn Baze Podataka	01/15/2016	1500	
101	SQL Programiranje	01/09/2015	1750	
102	Android programiranje	23/05/	2000	

**PRIKAZ KNJIGA SKUPLJIH OD 1000 RSD  
u proceduralnom programskom jeziku**

```
For each b in Knjige
  if cena > 1000
    add to knjige_niz
  else
    ignore
  end
return knjihe_niz
```

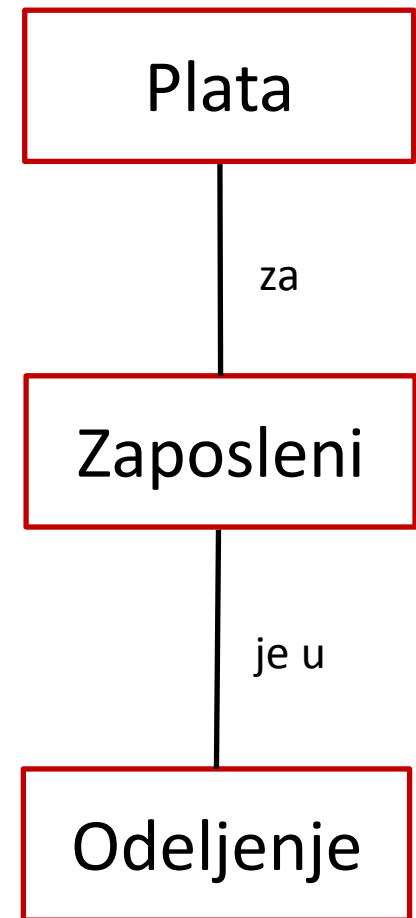
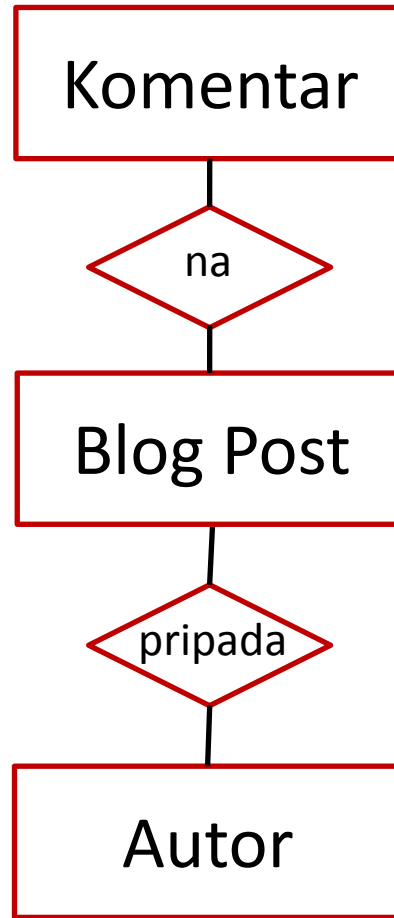
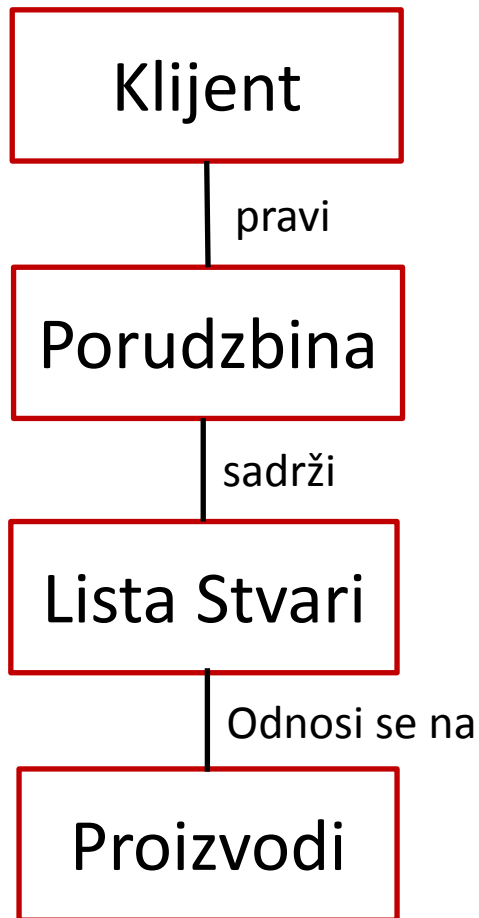
**PRIKAZ KNJIGA SKUPLJIH OD 1000  
RSD u SQL**

```
SELECT Naslov
FROM KNJIGE
WHERE Cena > 1000
```

# ŠEMA BAZE PODATAKA

---

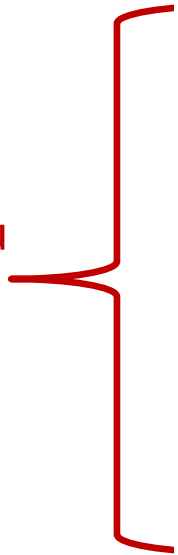
- Tabele (entiteti) osnovni blok svake baze podataka
- Kolone
- Primarni ključevi



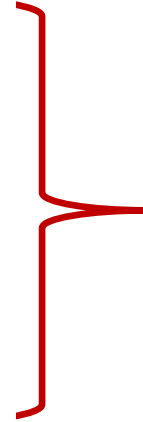
# ENTITETI i ATRIBUTI

---

Kolone u  
tabeli



.



Identifikacija  
tabele

# OBELEŽAVANJE ATRIBUTA

---

## Radnik tabela

- Ime
- Prezime
- DatumZaposlenja (datumzaposlenja, datum\_zaposlenja, dZaposlenja, ...)
- Adresa1
- Adresa2
- Grad
- Drzava
- Email
- Fotografija

# TIP PODATAKA I OSNOVNA OGRANIČENJA ZA SVAKI ATRIBUT

---

## Tabela Radnik

<b>Naziv Atributa</b>	<b>Tip podatka</b>	<b>Tip ograničenja</b>	<b>Default vrednost</b>
Ime	karakter	not NULL	
Prezime	karakter	not NULL	
DatumZaposlenja	datum (date)	not NULL	default: today
Plata	Broj (int)	not NULL	
Adresa1	karakter	not NULL	
Adresa2	karakter	NULL	
Drzava	karakter	not NULL	
Grad	karakter	not NULL	
Email	karakter	not NULL	Patern email: match
Fotografija	binary	NULL	

- Tipovi podataka koje koriste baze se razlikuju od tipova podataka u programskim jezicima.
- Tip podataka u programskom jeziku se odnosi na promenjive dok u bazama podataka se odnosi na kolone

# PRIRODNI PRIMARNI KLJUČ

---

## Tabela Knjiga

ISBN	Naslov	DatumPublikacije	Cena	...
134564868	Dizajn Baze Podataka	01/15/2016	1500	
134533862	SQL Programiranje	01/09/2015	1750	
914556869	Android programiranje	23/05/	2000	

**PRIMARNI KLJUČ (PK)**  
(prirodni ključ)

- Prirodni ključ je kolona koja je planirana u tabeli.

# GENERIČKI PRIMARNI KLJUČ

---

## Tabela Klijent

Integer, auto\_increment

KlijentID	Ime	Prezime	Adresa	Email	...
1	Milan	Stanko	SZK	ms@it.rs	
234	Dejan	Mitic	BB	dm@ni.rs	
235	Marina	Pantic	VK9	mp@vts.rs	

**PRIMARNI KLJUČ (PK)**

- Email kolona nije adekvatna za primarni ključ jer može da se desi da više korisnika deli isti email.

# KOMPOZITNI KLJUČ

- Kompozitni ključ je ključ koji je sastavljen iz vrednosti koje se nalaze u različitim kolonama
- Koristi se u situaciji kada na osnovu vrednosti u jednoj koloni ne možemo da identifikujemo svaki zapis
- Često se umesto kompozitnog ključa koristi generički ključ.

Yearbook

School	Year	ListPrice	PageCount	UnitsInStock	...
Orchard High	2010	\$29.95	144	32	
Orchard High	2011	\$34.95	132	14	
Lawstone Elementary	2010	\$29.95	161	0	
Lawstone Elementary	2011	\$29.95	155	38	
Lawstone Elementary	2012	\$34.95	172	144	
...					

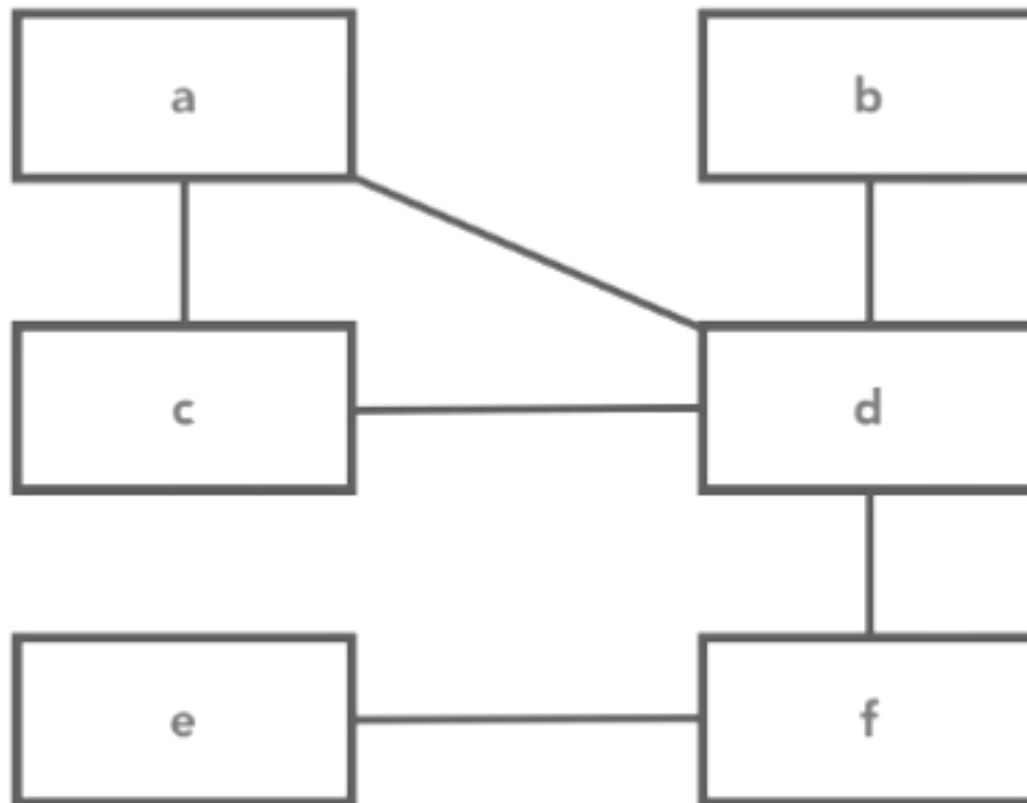
Primary Key (PK)  
(**composite** key)



# VEZE IZMEĐU TABELA

---

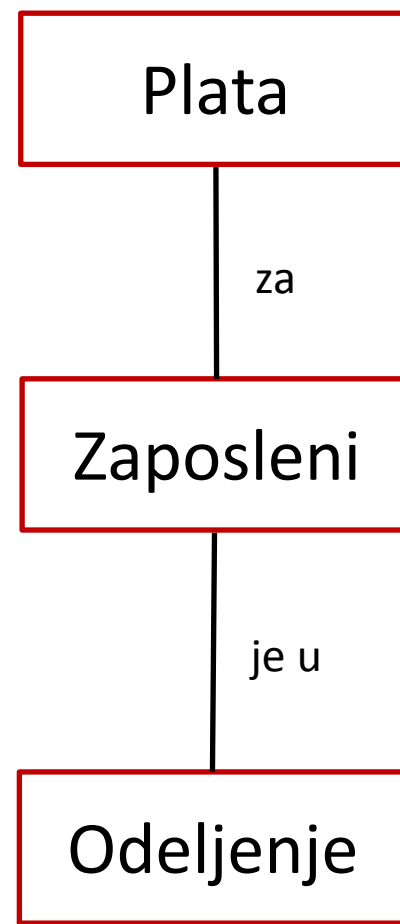
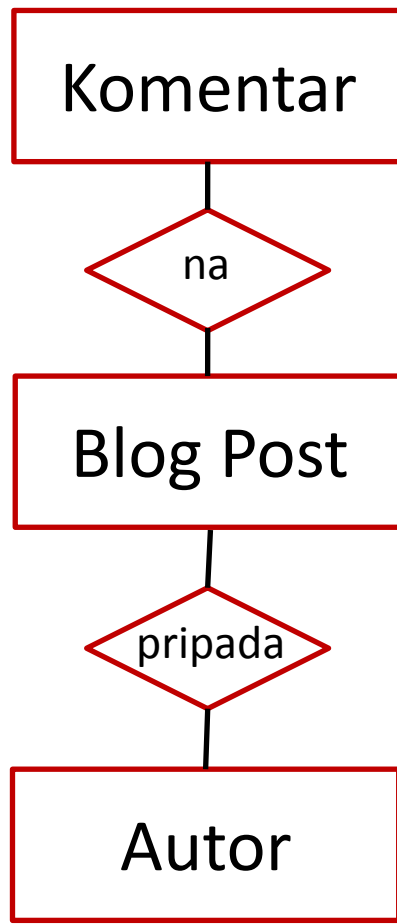
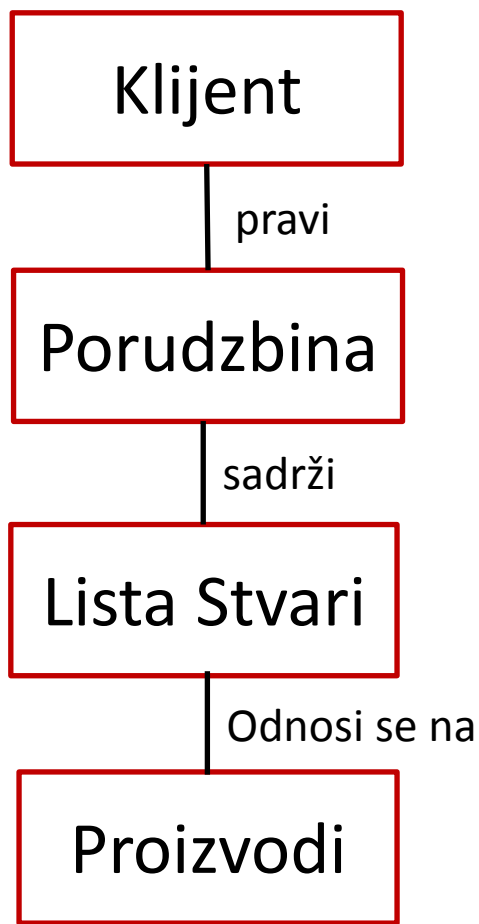
- Baza podataka je sastavljena iz tabela koje su međusobno povezane



# ER DIJAGRAM

---

- Potrebno je pronaći vezu i odrediti tip veze između tabela



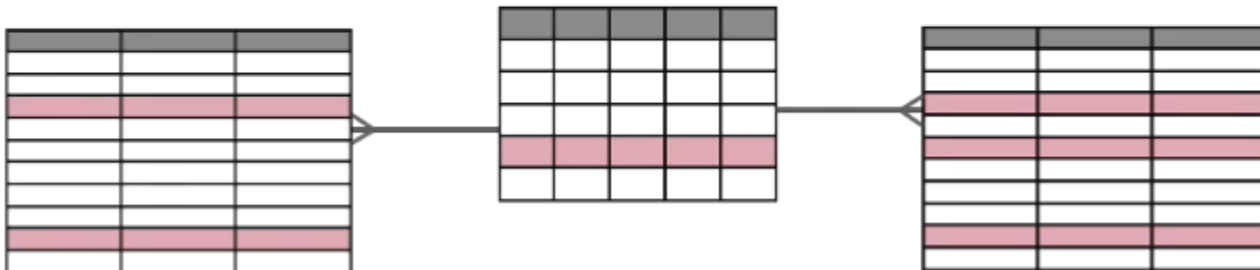
# TIP VEZE (Kardinalnost)

---

1 : 1 (jedan prema jedan)

1 : N (jedan prema više)

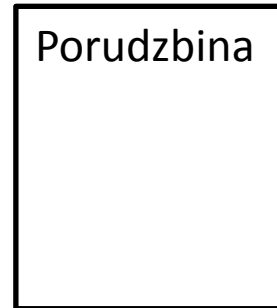
N : M (više prema više)



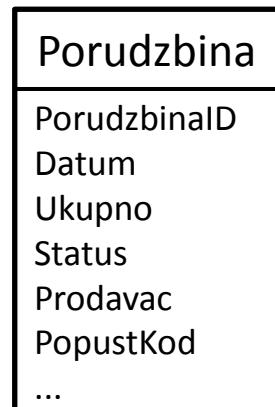
# REDOSLED PROJEKTOVANJA BAZE PODATAKA

---

## 1. Skiciramo (definišemo) entitete (tabele)



## 2. Definišemo attribute(kolone) za tabele



### 3. Odredimo Primarni Ključ (PK) za svaku tabelu

Klijent
<u>KlijentID (PK)</u>
Ime
Prezime
Email
Telefon
Adresa1
...

Porudzbina
<u>PorudzbinaID (PK)</u>
Datum
Ukupno
Status
Prodavac
PopustKod
...

### 3. Relacija između tabela

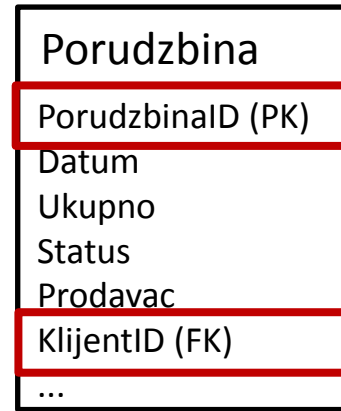
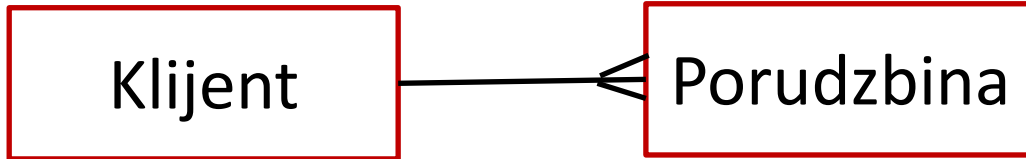
Klijent
KlijentID (PK)
Ime
Prezime
Email
Telefon
Adresa1
...

Porudzbina
PorudzbinaID (PK)
Datum
Ukupno
Status
Prodavac
PopustKod
...



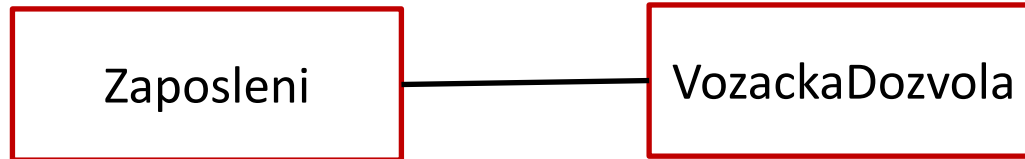
# RELACIJA 1 : N

---



# RELACIJA 1:1

- Ova veza se vrlo retko sreće
- Strani ključ može biti u bilo kojoj tabeli



Strani ključ (fk)

Id	Ime	Prezime	...
101	Dusan	Petrović	
102	Ana	Pešić	
103	Mita	Pešić	

BrDozvole	Grad	DatumVaženja
1234329	Niš	01/01/2017
2221221	Beograd	21/05/2020
9892345	Niš	03/27/2021

- Iako je Vozačka Dozvola drugi entitet, često se ova dva entiteta predstavljaju u formi jednog entiteta

Id	Ime	Prezime	...	BrDozvole	Grad	DatumVaženja
101	Dusan	Petrović		1234329	Niš	01/01/2017
102	Ana	Pešić		2221221	Beograd	21/05/2020
103	Mita	Pešić		9892345	Niš	03/27/2021



# RELACIJE M:N



## Class

ClassID	Title	Date	Classroom	Capacity	...
441	Database Design	3/1/2013	8	12	...
442	SQL for Beginners	3/1/2013	6	24	...
443	Advanced Normalization	4/12/2013	2	18	...

Primarni ključ (pk)

ClassID	StudentID
441	101
441	102
441	103
442	102

ClassStudent

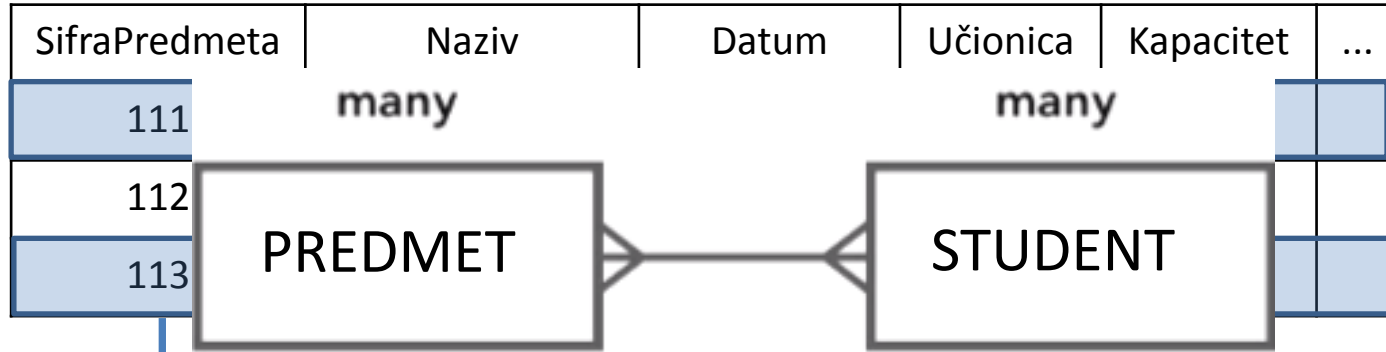
## Student

StudentID	FirstName	LastName	Email	...
101	David	Denton	dd@acme...	
102	Viola	Couch	vc1968@...	
103	Arthur	Price	arthur@me...	

Kreira se nova tabela koja povezuje ove dve tabele

# RELACIJA M:N

## tbIPREDMETI



## tbIPRIJAVA

SifraPredmeta	SifraStudenta
111	50
112	51
111	52
113	50

Kreira se nova tabela koja povezuje ove dve tabele

## tbISTUDENTI

SifraStudenta	Ime	Prezime	Emai	...
50	Dušan	Ilić	di@vts.rs	
51	Jelena	Mitic	jm@vts.rs	
52	Darko	Mitov	dm@vts.rs	

# REFERENCIJALNI INTEGRITET

- Relacija predstavlja vezu kako su dve tabele povezane
- Veza je pravilo koje sadrži ograničenja koja nesmeju da se naruše

Customer

CustomerID	FirstName	LastName	Email	Address	...
367	Michelle	Blackwell	mblackwell@	22 Acacia...	...
368	Lynn	Allen	la1942@...	1016B 1st...	...
369	Lee	Stout	lee@...	47 Main St	...

Order

OrderID	Date	Quantity	TotalDue	CustomerID
101	3/1/2011	17	\$340.00	367
102	3/2/2011	47	\$902.00	367
103	3/2/2011	104	\$1500.00	369
104	3/3/2011	22	\$440	388



- DBMS ne dozvoljava unos narudžbine za klijenta koji ne postoji u tabeli Klijent
- Mora prvo da se kreira klijent a zatim narudžbina za njega
- Obrnuto nije dozvoljeno

# REFERENCIJALNI INTEGRITET

## KASKADNO BRISANJE

- Ukoliko obrišemo klijenta, baza će automatski da obriše sve redove u povezanoj tabeli koji imaju veze sa tim klijentom
- Ukoliko nije uključena opcija za kaskadno brisanje, DBMS nam neće dozvoliti da obrišemo klijenta ukoliko postoje redovi u povezanoj tabeli koji se odnose na tog klijenta.

Customer

CustomerID	FirstName	LastName	Email	Address	...
<del>367</del>	<del>Michelle</del>	<del>Blackwell</del>	<del>mblackwell@...</del>	<del>22 Addison...</del>	<del>...</del>
368	Lynn	Allen	la1942@...	1016B 1st...	...
369	Lee	Stout	lee@...	47 Main St	...

Order

OrderID	Date	Quantity	TotalDue	CustomerID
<del>101</del>	<del>3/1/2011</del>	<del>17</del>	<del>\$310.00</del>	<del>367</del>
<del>102</del>	<del>3/2/2011</del>	<del>47</del>	<del>\$902.00</del>	<del>367</del>
103	3/2/2011	104	\$1500.00	369

# REFERENCIJALNI INTEGRITET

## KASKADNO Anuliranje

- Ukoliko obrišemo klijenta, baza će automatski da upiše null za vrednost stranog ključa u povezanoj tabeli u redovima koji imaju veze sa tim klijentom
- Ova opcija se retko koristi i nemaju svi DBMS ovu opciju

**Customer**

CustomerID	FirstName	LastName	Email	Address	...
367	Michelle	Blackwell	mblackwell@...	22 Acadian...	...
368	Lynn	Allen	la1942@...	1016B 1st...	...
369	Lee	Stout	lee@...	47 Main St	...

**Order**

OrderID	Date	Quantity	TotalDue	CustomerID
101	3/1/2011	17	\$340.00	367
102	3/2/2011	47	\$902.00	367
103	3/2/2011	104	\$1500.00	369

# REFERENCIJALNI INTEGRITET BEZ AKCIJE (No Action)

- Podrazumevana opcija u većini DBMS

Customer

CustomerID	FirstName	LastName	Email	Address	...
367	Michelle	Blackwell	mblackwell@...	22 Acadian...	...
368	Lynn	Allen	la1942@...	1016B 1st...	...
369	Lee	Stout	lee@...	47 Main St	...

Javiće se greška ukoliko pokušamo prvo da obrišemo klijenta jer u povezanoj tabeli postoje redovi koji se odnose na tog klijenta

Order

OrderID	Date	Quantity	TotalDue	CustomerID
101	3/1/2011	17	\$340.00	367
102	3/2/2011	17	\$702.00	367
103	3/2/2011	104	\$1500.00	369

# REFERENCIJALNI INTEGRITET

## Kaskadno ažuriranje

### Customer

CustomerID	FirstName	LastName	Email	Address	...
340	Michelle	Blackwell	mblackwell@	22 Acacia...	...
368	Lynn	Allen	la1942@...	1016B 1st...	...
369	Lee	Stout	lee@...	47 Main St	...

### Order

OrderID	Date	Quantity	TotalDue	CustomerID
101	3/1/2011	17	\$340.00	340
102	3/2/2011	47	\$902.00	340
103	3/2/2011	104	\$1500.00	369

# Normalizacija

- Skup pravila koja se primenjuju nad bazom podataka
- Cilj je projektovanje baze podataka bez anomalija brisanja, ažuriranja i sa minimalnom redundansom.
  - Podacima je lako pristupiti
    - Brisanje
    - Ažuriranje
    - Čitanje





# Prva normalna forma 1NF

- Zahteva da svaka ćelija u tabeli sadrži samo jednu vrednost
- Rešava problem viševrednosnih atributa (radnik može da radi na više računara)

## Employee

EmployeeID	FirstName	LastName	Email	...	ComputerSerial
551	Les	Adams	ladams@	...	XP5435512 , XA5543231
552	Jill	Baker	jbaker@	...	WA2324451
553	Stephen	Jackson	s.jackson@	...	BC32345412 , ZZ87656, XX21312

Ovakav unos  
podataka krši 1NF

# REŠENJE 1NF

- Problem ovakvog rešenja je što svaki zaposleni ima različit broj računara, određena polja ostaće prazna

Employee

EmployeeID	FirstName	LastName	Email	...
551	Les	Adams	ladams@	...
552	Jill	Baker	jbaker@	...
553	Stephen	Jackson	s.jackson@	...

Computer

Serial	EmployeeID	Description	...
XP5435512	551	Dell Laptop	...
XA5543231	551	Apple MacBook	...
WA2324451	552	Acer Desktop	...
BC32345412	553	MacBook Pro	...
ZZ87656	553	HP Server	...
XX21312	553	iPad 3	...

Strani ključ (fk)

REŠENJE za 1NF  
koje se preferira

# Druga normalna forma 2NF

- Opisuje odnos između kolona koje su ključevi i kolona koje to nisu.
- Zahteva se da svaka kolona koja nije ključ zavisi od kolone ili kolona koje su ključ za tabelu
- Sa drugom normalnom formom imamo problema samo ukoliko se koristi kompozitni ključ(primer) u suprotnom možemo da pređemo u 3 NF.
  - U tom slučaju je potrebno sagledati kolone koje nisu ključevi

## Events

Course	Date	CourseTitle	Room	Capacity	Available	...
SQL101	3/1/2013	SQL Fundamentals	4A	12	4	
DB202	3/1/2013	Database Design	7B	14	7	
SQL101	4/14/2013	SQL Fundamentals	7B	14	10	
SQL101	5/28/2013	SQL Fundamentals	12A	8	8	
CS200	4/15/2012	C Programming	4A	12	11	

Kompozitni  
primarni ključ

# REŠENJE ZA 2 NF

## Events

Course	Date	Room	Capacity	Available	...
SQL101	3/1/2013	4A	12	4	
DB202	3/1/2013	7B	14	7	
SQL101	4/14/2013	7B	14	10	
SQL101	5/28/2013	12A	8	8	
CS200	4/15/2012	4A	12	11	

## Course

CourseID	Title	...
SQL101	SQL Fundamentals	...
DB202	Database Design	...
CS200	C Programming	...

# Treća normalna forma 3NF

- Slučaj kada neključni atribut zavisi od drugog neključnog atributa
- Tabele su u 1NF i 2NF.
- Potrebno je sagledati attribute koji nisu ključevi.
  - U našem slučaju su to kolone Prostorija i Broj mesta koje ne zavise od primarnog ključa
  - Primećujemo da se soba 4A i soba 7B ponavljaju nekoliko puta što je nepotrebno, tj. zavise međusobno.

**Events**

Course	Date	Room	Available	...
SQL101	3/1/2013	4A	4	
DB202	3/1/2013	7B	7	
SQL101	4/14/2013	7B	10	
SQL101	5/28/2013	12A	8	
CS200	4/15/2012	4A	11	

**Course**

CourseID	Title	...
SQL101	SQL Fundamentals	...
DB202	Database Design	...
CS200	C Programming	...

**Room**

Room	Capacity
4A	12
7B	14
12A	8

# Treća normalna forma 3NF

- Sledeći primer 3NF
  - Količina i Cena po jedinici su neključni atributi
  - Ukoliko u tabeli čuvamo kolonu Total može da se desi da pogrešno unesemo vrednost koja se dobija kao proizvod količine i cene a da ne možemo da zaključimo da li smo pogrešili u količini ili u ceni.

OrderItem

ID	OrderID	ProductID	Quantity	UnitPrice	Total
10011	454	A43	4	\$10.00	\$40.00
10012	455	B56	2	\$5.00	\$10.00
10013	456	A22	1	\$9.99	\$9.99
10014	456	A43	55	\$10.00	\$550.00
10015	457	C77	10	\$75.00	\$750.00

Calculated kolona, ne čuva se u bazi i read-only je

# Slučajevi kada se ne poštuje normalizacija u potpunosti(denormalizacija)

- U praksi je nekad jednostavnije da dodamo novu kolonu ili kolone iako će neke ćelije ostati prazne nego da kreiramo novu tabelu i uvek kada treba da čitamo određene informacije koristimo uslov za povezivanje tabela.
- U ovom slučaju pravimo izuzetak i nepoštujeemo pravila zbog performansi.
- Normalizacija je prisutna radi manjeg ponavljanja podataka ali nije sjajna u delu koji se odnosi na performanse.

**Employee**

EmployeeID	FirstName	LastName	Email	Phone
551	Les	Adams	ladams@	555-555-1212
552	Jill	Baker	jbaker@	555-543-9876
553	Stephen	Jackson	s.jackson@	555-101-2345

# SQL

- Jezik koji se koristi u radu sa relacionim bazama podataka
- Neije programski jezik i manje je kompleksan.
- Koristi se isključivo u radu za bazama podataka.





# SQL – Select(čitanje podataka)

```
SELECT columns  
FROM table
```

Employee
EmployeeID (PK)
FirstName
LastName
HireDate
Email
Department
Salary

```
SELECT FirstName  
FROM Employee
```

FirstName
Jill
Fred
Reginald
Ray
Lester
Anthony
Angela
Monica
Terri
...

# Identifikacija baze



```
SELECT *  
FROM HumanResources.Employee
```

# SQL – WHERE (Filter)

- Ekvivalentan je IF u HLL programskom jeziku
- U SQL- String podaci se stavljaju pod jednostrukim navodnicima za razliku od HLL jezika gde se koriste dvostruki navodnici
- Jednakost se ispituje “=” za razliku od HLL gde se jednakost ispituje “==”

Employee
EmployeeID (PK)
FirstName
LastName

```
SELECT *  
FROM HumanResources.Employee  
WHERE Salary > 50000;
```

```
SELECT *  
FROM Employee  
WHERE Salary > 50000;'4; ;
```

# SQL RELACIONI OPERATORI

```
SELECT *  
FROM Employee  
WHERE Salary > 50000;
```

>

<

>=

<=

<>

# SQL KOMBINOVANJE VIŠE USLOVA

```
SELECT *  
FROM Employee  
WHERE Department IN  
('Marketing', 'Sales');
```

# SQL – Like komanda

- Komanda se koristi kada radimo operacije sa stringovima
- Za razliku od operacije “=” koja traži tačno mečovanje komanda Like nam dozvoljava veću fleksibilnost i efikasnost prilikom upoređivanja
- Wildcard pretraga
  - % mečuje više bilo kojih karaktera
  - \_ mečuje jedan bilo koji karakter

## Employee

EmployeeID (PK)
FirstName
MiddleInitial
LastName
HireDate
Email
Department
Salary

```
SELECT *  
FROM Employee  
WHERE LastName LIKE 'Sm_oth';
```

# SQL – rad sa NULL poljima

```
SELECT *  
FROM Employee  
WHERE MiddleInitial IS NULL;
```

```
SELECT *  
FROM Employee  
WHERE MiddleInitial IS NOT NULL;
```

## Employee

EmployeeID (PK)  
FirstName  
MiddleInitial  
LastName  
HireDate  
Email  
Department  
Salary

# SQL – Sortiranje rezultata

Product
ProductID (PK)
Description
ListPrice
Color
Weight
Category
SKU
Manufacturer

- Želimo da nam se prvo prikaže najveća cena a na kraju najmanja
- Default sortiranje je u rastućem redosledu

```
SELECT Description,  
ListPrice, Color  
FROM Product ;
```

Description	ListPrice	Color
Extender Cables	4.49	Black
Battery Charger	35.00	Black
Seat Cover	7.98	Red
Headphone Amp	420.00	Silver
...	...	...

```
SELECT Description,  
ListPrice, Color  
FROM Product  
ORDER BY ListPrice DESC ;
```

Description	ListPrice	Color
Premier Headphone Amp	699.00	Gold
Headphone Amp	420.00	Silver
Compressor Unit	399.00	Black
Adjustable LED Lamp	349.98	White
...	...	...

# SQL – Sortirenje na osnovu više kriterijuma

```
SELECT *  
FROM Employee  
WHERE Salary > 50000  
ORDER BY LastName, FirstName;
```

Employee
EmployeeID (PK)
FirstName
LastName
HireDate
Email
Department
Salary
...

- Ukoliko se ne navede DESC ili ASC, default je ASC(rastući redosled)

EmployeeID	FirstName	LastName	HireDate	...
489	Matilda	Aaron	1/14/2001	
24	Maria	Adams	9/24/1992	
551	Siobhan	Adams	6/23/2001	
439	Stephen	Adams	9/19/2000	
1008	Gertrude	Bailey	4/14/2008	
...				



# SQL – Agregacione funkcije

- Agregatne funkcije ili funkcije grupisanja
  - Izvršavaju kalkulaciju nad određenim skupom podataka i **vraćaju jednu vrednost**
  - Vrednost koju vraćaju zavisi od agregatne funkcije koju koristimo

Employee
EmployeeID (PK)
FirstName
MiddleInitial
LastName
HireDate
Email
Department
Salary

- **Ukupan broj redova u tabeli**
- **Broji sve redove iz tabele Zaposleni**

```
SELECT COUNT(*)  
FROM Employee
```

result
547

```
SELECT *  
FROM Employee
```

```
SELECT COUNT(*)  
FROM Employee  
WHERE Salary > 50000
```

result
132

EmployeeID	FirstName	LastName	...
2	Aaron	Cooper	...
4	Lou	Donoghue	
5	Alice	Bailey	
6	Oswald	Hall	
7	John	Velasquez	
...			

# SQL – Agregacione funkcije

```
SELECT MAX(ListPrice) FROM Product ;
SELECT MIN(ListPrice) FROM Product ;
SELECT AVG(ListPrice) FROM Product ;
```

result
699.00

## Agregacione funkcije u kombinaciji sa filtriranjem

```
SELECT SUM(TotalDue)
FROM Order
WHERE CustomerID = 854;
```

```
SELECT COUNT(*)
FROM Product
WHERE Color = 'Red'
```

result
2742.75

# SQL – Grupisanje rezultata

- Grupisanje se koristi samo u kombinaciji sa agregatnim funkcijama
- Koristi se za kategorizaciju rezultata tj. ne vraća samo jednu vrednost.
  - Atribut na osnovu kojeg radimo kategorizaciju
  - Izdvojiće sve vrednosti iz kolone Color, prebrojaće ih i prikazati u zasebnim redovima

```
SELECT COUNT(*) , Color  
FROM Product  
GROUP BY Color
```

result	Color
47	Black
32	Silver
8	White
86	Clear
...	...

# SQL – Inner Join (Spajanje tabela)

M : 1

Employee

Department

ID	FirstName	LastName	HireDate	DepartmentID	...
734	Aaron	Cooper	4/17/09	2	
735	Lou	Donoghue	5/22/05	4	
736	Alice	Bailey	9/1/99	(null)	
737	Oswald	Hall	3/19/11	5	
738	John	Velasquez	4/5/10	4	
...	...				

DepartmentID	Name	Location	BudgetCode	...
1	Production	CA	A4	...
2	R&D	AZ	B17	
3	Marketing	CA	A7	
4	Sales	CA	A7	
5	PR	UK	C9	
...				

Strani ključ (FK)

Primarni ključ (PK)

```
SELECT FirstName, LastName, HireDate,  
       Employee.DepartmentID,  
FROM Employee
```

FirstName	LastName	HireDate	Employee. DepartmentID	Name	Location
Aaron	Cooper	4/17/09	2	R&D	AZ
Lou	Donoghue	5/22/05	4	Sales	CA
Oswald	Hall	3/19/11	5	PR	UK
John	Velasquez	4/5/10	4	Sales	CA
...					

- Inner Join prikazaće samo redove za koje postoji jednakost u kolonama koje su upoređivane.
- Neće prikazati **DepartmentId=1** jer ni jedan radnik ne radi u tom odeljenju ni radnika sa **ID=736** jer ne radi ni u jednom odeljenju

# SQL – Outer Join (Spajanje tabela)

M : 1

Employee

Department

ID	FirstName	LastName	HireDate	DepartmentID	...
734	Aaron	Cooper	4/17/09	2	
735	Lou	Donoghue	5/22/05	4	
736	Alice	Bailey	9/1/99	(null)	
737	Oswald	Hall	3/19/11	5	
738	John	Velasquez	4/5/10	4	
...	...				

DepartmentID	Name	Location	BudgetCode	...
1	Production	CA	A4	...
2	R&D	AZ	B17	
3	Marketing	CA	A7	
4	Sales	CA	A7	
5	PR	UK	C9	
...				

Strani ključ (FK)

Primarni ključ (PK)

```
SELECT FirstName, LastName, HireDate,  
       Employee.DepartmentID  
FROM Employee
```

FirstName	LastName	HireDate	Employee. DepartmentID	Name	Location
Aaron	Cooper	4/17/09	2	R&D	AZ
Lou	Donoghue	5/22/05	4	Sales	CA
Alice	Bailey	9/1/99	(null)	(null)	(null)
Oswald	Hall	3/19/11	5	PR	UK
John	Velasquez	4/5/10	4	Sales	CA
...					

- **LEFT OUTER JOIN** prikazaće sve redove iz leve (Employee) tabele i sve redove za koje postoji jednakost u kolonama koje su upoređivane.
- Na prethodni rezultat dodaće i radnika sa **ID=736** koji ne radi ni u jednom odeljenju

# SQL – Outer Join (Spajanje tabela)

M : 1

Employee

Department

ID	FirstName	LastName	HireDate	DepartmentID	...
734	Aaron	Cooper	4/17/09	2	
735	Lou	Donoghue	5/22/05	4	
736	Alice	Bailey	9/1/99	(null)	
737	Oswald	Hall	3/19/11	5	
738	John	Velasquez	4/5/10	4	
...	...				

DepartmentID	Name	Location	BudgetCode	...
1	Production	CA	A4	...
2	R&D	AZ	B17	
3	Marketing	CA	A7	
4	Sales	CA	A7	
5	PR	UK	C9	
...				

Strani ključ (FK)

Primarni ključ (PK)

```
SELECT FirstName, LastName, HireDate,  
       Employee.DepartmentID  
FROM Employee
```

t

FirstName	LastName	HireDate	Employee. DepartmentID	Name	Location
(null)	(null)	(null)	(null)	Production	CA
Aaron	Cooper	4/17/09	2	R&D	AZ
Lou	Donoghue	5/22/05	4	Sales	CA
Oswald	Hall	3/19/11	5	PR	UK
John	Velasquez	4/5/10	4	Sales	CA
(null)	(null)	(null)	(null)	Marketing	CA

- **RIGHT OUTER JOIN** prikazaće sve redove iz desne (Department) tabele i sve redove za koje postoji jednakost u kolonama koje su upoređivane.
- Dodaće odeljenja u kojima ne radi ni jedan radnik.

# SQL – INSERT, UPDATE i DELETE

**C  
R  
U  
D**





# SQL – Update (ažuriranje podataka)

UPDATE *table*

Želimo da  
setujemo email  
adresu

EmployeeID	FirstName	LastName	HireDate	Email	Department	Salary
734	Joe	Allen	3/12/13	(null)	Sales	45000

UPDATE Employee

Ukoliko ne koristimo WHERE ključnu reč u UPDATE upitu, svaki red biće ažuriran

# SQL – DELETE (brisanje redova)

```
DELETE FROM table
```

```
DELETE FROM Employee  
WHERE EmployeeID = 734
```

EmployeeID	FirstName	LastName	HireDate	Email	Department	Salary
<del>734</del>	<del>Joe</del>	<del>Allen</del>	<del>3/12/13</del>	<del>(null)</del>	<del>Sales</del>	<del>45000</del>

# OPREZNOST PRILIKOM BRISANJA PODATAKA

Brišu se svi  
podaci iz tabele

```
DELETE FROM Employee
```

## Preporuka prilikom brisanja

```
DELETE  
FROM Employee  
WHERE EmployeeID = 734
```

EmployeeID	FirstName	LastName	HireDate	Email	Department	Salary
------------	-----------	----------	----------	-------	------------	--------

# DATA MANIPULATION LANGUAGE (DML)

data manipulation

SELECT  
INSERT  
UPDATE  
DELETE

- Ove komande nam ne dozvoljavaju da izmenimo strukturu tabele  
Komande uz pomoć kojih radimo manipulaciju nad podacima,
- Način definhisanja kolona
  - Način obeležavanja primarnog ključa
  - Način obeležavanja ograničenja
  - ....

# DATA DEFINITION LANGUAGE (DDL)

data definition

CREATE  
ALTER  
DROP

Ove komande nam dozvoljavaju da kreiramo strukturu(šemu) tabele

- Definisnja kolona
- Obeležavanja primarnog ključa
- Obeležavanje stranog ključa
- Obeležavanja ograničenja
- Dodavanje novih kolona
- Brisanje kolona
- ....

# DDL – CREATE

(kreiranje tabele - definicija kolona)

```
CREATE table
```

```
CREATE Employee  
(EmployeeID    INTEGER    PRIMARY KEY,  
  FirstName    VARCHAR(35) NOT NULL,  
  LastName     VARCHAR(100) NOT NULL,  
  Department   VARCHAR(30) NULL,  
  Salary       INTEGER  
);
```

## DDL – ALTER

(kreiranje nove kolone u postojećoj tabeli)

```
ALTER TABLE Employee  
ADD Email VARCHAR(100);
```

## DDL – DROP

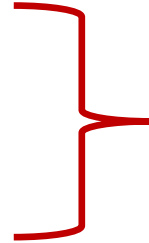
(brisanje tabele)

```
DROP TABLE Employee;
```

# DCL – Data Control Language

data control

GRANT  
REVOKE



Ove komande se koriste za definisanje pristupa bazi tj. za dodelu ili ukidanje privilegija korisnicima



DML ----- DDL ----- DCL

data manipulation

SELECT  
INSERT  
UPDATE  
DELETE

data definition

CREATE  
ALTER  
DROP

data control

GRANT  
REVOKE

# INDEKSI

- Kako baza postaje sve veća dizajn baze postaje sve važniji
- Indeksi igraju značajnu ulogu u brzini rada same baze
- Indeksi nam omogućavaju da brže pronađemo ono što tražimo
- Slični su indeksima koji se koriste u knjizi.
  - Ukoliko nas nešto posebno interesuje u knjizi ne želimo da pročitamo celu knjigu da bi smo pronašli deo koji nas interesuje
- Podaci u tabeli nisu strukturirani i nisu poređani
- Indeksi nam omogućavaju da brže pristupimo određenom redu u velikoj tabeli

CustomerID	FirstName	LastName	Email	Address	...
561	Lee	Stout	lee@...	47 Main St	...
592	Lynn	Allen	la1942@...	1016B 1st...	...
584	Michelle	Blackwell	mblackwell@	22 Acacia...	...
551	Angela	Smith	angie@...	78 Privet Dr	...
...	...	...	...	...	...

CustomerID	FirstName	LastName	Email	Address	...
551	Angela	Smith	angie@...	78 Privet Dr	...
561	Lee	Stout	lee@...	47 Main St	...
584	Michelle	Blackwell	mblackwell@	22 Acacia...	...
592	Lynn	Allen	la1942@...	1016B 1st...	...
...	...	...	...	...	...

clustered index

# Cluster Indeks

- Primarni indeks koji se kreira u svakoj tabeli je cluster indeks
- Redovi u tabeli se organizuju po toj koloni
- Cluster indeks je sličan prezimenu u telefonskom imeniku
- Cluster indeks je u tabeli primarni ključ i većina DBMS će to uraditi automatski jer je primarni ključ označen kao jedinstven preko koga pristupamo svakom redu
- Ukoliko pristupamo podacima koristeći neku drugu kolonu možemo da promenimo cluster indeks.
- Svaka tabela može da ima samo jedan cluster indeks isto kao što telefonski imenik može da bude sortiran na samo jedan način

CustomerID	FirstName	LastName	Email	Address
561	Lee	Stout	lee@...	47 Main St
592	Lynn	Allen	la1942@...	1016B 1st...
584	Michelle	Blackwell	mblackwell@	22 Acacia...
551	Angela	Smith	angie@...	78 Privet Dr
...	...	...	...	...

CustomerID	FirstName	LastName	Email	Address
551	Angela	Smith	angie@...	78 Privet Dr
561	Lee	Stout	lee@...	47 Main St
584	Michelle	Blackwell	mblackwell@	22 Acacia...
592	Lynn	Allen	la1942@...	1016B 1st...
...	...	...	...	...

clustered index

# Cluster Indeks

CustomerID	FirstName	LastName	Email	Address
551	Angela	Smith	angie@...	78 Privet Dr
561	Lee	Stout	lee@...	47 Main St
584	Michelle	Blackwell	mblackwell@	22 Acacia...
592	Lynn	Allen	la1942@...	1016B 1st...
...	...	...	...	...

clustered index

```
SELECT *  
FROM Customer  
WHERE CustomerID = 584;
```

Pretraga će biti vrlo brza jer pretraživanje radimo po primarnom ključu koji je indeksiran

# None Clustered Indeksi

- Može da ih bude više u tabeli

CustomerID	FirstName	LastName	Email	Address
551	Angela	Smith	angie@...	78 Privet Dr
561	Lee	Stout	lee@...	47 Main St
584	Michelle	Blackwell	mblackwell@	22 Acacia...
592	Lynn	Allen	la1942@...	1016B 1st...
...	...	...	...	...

clustered index

```
SELECT *  
FROM Customer  
WHERE LastName = 'Smith';
```

- Pošto DBMS ne zna gde se nalazi traženo prezime on vrši pretragu redom od prvog do poslednjeg reda
- Full table scan, neefikasno pretraživanje traženog podatka
- Neefikasnost raste sa brojem redova u tabeli

# None Clustered Indeksi

- Rešenje je sekundarni indeks (none clustered index) koje se može kreirati.
- Kreira se posebna tabela koja je sortirana na osnovu Prezimena
- Pretraživanje na osnovu prezimena je sada ubrzano
- Ubrzanje nije ekvivalentno cluster indeksu jer je potrebno pretraživanje sprovesti u dve tabele

non-clustered index  
on LastName

LastName	CustomerID
Allen	592
Bailey	432
Blackwell	584
Burns	122
...	...
...	...
Smith	551
...	...

CustomerID	FirstName	LastName	Email	Address
551	Angela	Smith	angie@...	78 Privet Dr
561	Lee	Stout	lee@...	47 Main St
584	Michelle	Blackwell	mblackwell@	22 Acacia...
592	Lynn	Allen	la1942@...	1016B 1st...
...	...	...	...	...

clustered index

```
SELECT *  
FROM Customer  
WHERE LastName = 'Smith';
```

Preporuka je koristi ovaj indeks  
samo za kolone za koje znamo  
daće se dosta koristiti u upitima

# None Clustered Indeksi - Nedostatak

- Postavlja se pitanje zašto ne indeksiramo sve kolone u tabeli i obezbedimo brži prikaz podataka
- Problem je što svaki indeks ima svoju cenu.
  - Prednost indeksa je u čitanju (**select**) podataka
  - Nedostatak je u pisanju(**insert**) i promeni(**update**) podataka
- U našem slučaju pretraživanje na osnovu imena ili prezimena biće brzo jer smo kreirali dva none cluster indeksa nad ovim kolonama
- Ukoliko kreiramo novog zaposlenog umesto jedne operacije fizičkog upisa na disku to će zahtevati tri upisa jer imamo tri tabele (cluster index i dve none cluster indeks).
- Što je veći broj indeksnih kolona veće vreme fizičkog upisa je potrebno i baza postaje neefikasna

clustered index

EmployeeID	FirstName	MiddleInitial	LastName	...
2	Aaron	F	Cooper	...
4	Lou	(null)	Donoghue	
5	Alice	B	Bailey	
6	Oswald	T	Hall	
7	John	(null)	Velasquez	
8	Brenda	A	Daniels	

non-clustered index  
on FirstName

FirstName	EmployeeID
Aaron	2
Alice	5
Brenda	8
John	7
Lou	4
Oswald	6

non-clustered index  
on LastName

LastName	EmployeeID
Bailey	5
Cooper	2
Daniels	8
Donoghue	4
Hall	6
Velasquez	7

# Indeksiranje

- Indeksiranje zahteva od administratora baze podataka konstatno nadgledanje performansi sistema
- Indeksiranje je trgovina između:
  - Bržeg čitanja podataka
  - Sporijeg upisa podataka



# TRANSAKCIJE

- Transakcije su vrlo bitne u svetu baza podataka
- Šta je transakcija u realnom svetu
  - Ukoliko prodavcu knjiga platimo, očekujemo da dobijemo knjigu.
  - To je transakcija ukoliko su se obe stvari desile
  - Ukoliko smo dali novac ocekujemo knjigu, ukoliko prodam knjigu očekujem novac za nju
- U svetu računara klasičan primer transakcije je bankarski sistem.
  - Ulogujemo se na naš račun preko Weba i želimo da prebacimo novac sa jednog računa na drugi.
  - Ova akcija zahteva dve promene nad podacima, sa jednog računa oduzimamo novac a na drugom računu dodajemo novac.
  - Transakcija je uspešna ukoliko su se obe stvari izvršene, ukoliko jedna stvar ne uspe, sistem vraća podatke na početno stanje

Account	AccountType	Balance
A2354542	Savings	\$6000

Account	AccountType	Balance
C9876567	Checking	\$12

**transakcija**

- \$200

+ \$200

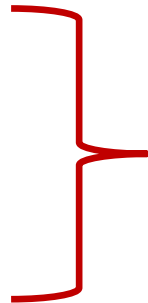
# TRANSAKCIJE

- Termin koji se vrlo često koristi u radu sa transakcijama je ACID i ugrađen je DBMS
  - Atomic
    - Zahteva da se sve akcije u transakciji izvrše ili sistem vraća na originalno stanje.
    - Razlog zašto se sve akcije u transakciji ne izvrše je nestanak električne energije ili nedovoljno prostora ili ...
    - Atomic znači sve ili ništa
  - Consistent
    - Transakcijom baza podataka iz jednog validnog stanja prelazi u drugo na osnovu pravila u bazi
  - Isolated
    - Odnosi se na to da podatak koji je uključen u transakciji bude zaključan za vreme trajanja transakcije, tj. nesme da se dozvoli drugom sistemu da menja isti podatak
  - Durable
    - Garantuje izvršenje transakcije čak i ukoliko se desi neki otkaz

Atomic  
Consistent  
Isolated  
Durable

# TRANSAKCIJE – Isolated Osobina

ID	Nickname	Balance	...
1	Joint	\$9000	...
2	Alice	\$1050	
3	Bob	\$45	
...	...	...	



- Tri računa
  - Joint zajednički račun
  - Alis i Bob su odvojeni računi
- Posmatraju se koraci u transferu novca između Joint računa i Alis računa

Prikaži saldo **Joint** računa (\$10000)

Prikaži saldo **Alis** računa (\$50)

Promeni saldo na **Joint** račun (\$10000) - \$1000

Promeni saldo na **Alis** račun (\$50) + \$1000

# TRANSAKCIJE – Isolated Osobina

Javiće se konflikt jer dva programa istovremeno rade nad istim podatkom, na Joint računu treba da bude \$8000

ID	Nickname	Balance	
1	Joint	\$9000	...
2	Alice	\$1050	
3	Bob	\$1045	
...	...	...	

Slučaj kada Alis i Bob istovremeno rade operacije sa svog i Joint računa.

## Alis

Prikaži saldo **Joint** računa (\$10000)  
Prikaži saldo **Alis** računa (\$50)  
Promeni saldo na **Joint** računu (\$10000) - \$1000  
Promeni saldo na **Alis** računu (\$50) + \$1000

## Bob

Prikaži saldo **Joint** računa (\$10000)  
Prikaži saldo **Bob** računa (\$45)  
Promeni saldo na **Joint** računu (\$10000) - \$1000  
Promeni saldo na **Bob** računu (\$45) + \$1000

Bob veruje da na Joint racunu i dalje ima \$1000 jer je u prvoj iteraciji prilikom prikazivanja dobijo taj podatak

# Transakcija-Rešenje problema

- Potrebno je nekoliko koraka učiniti jedinstvenim tj grupisati u jedan blok koji je nedeljiv praveći transakciju.
- Ukoliko se u toku transakcije javi problem(nestanak el. Energije) sistem se vraća na stanje pre transakcije.
- Transakcije i dalje ne rešavaju *race condition* problem tj. transakcije koje se izvršavaju u isto vreme.
  - Slulaj kada je Bob u fazi čitanja podataka a Alis u fazi izmene podataka (*dirty read*)

## Alis

### BEGIN TRANSACTION

Prikaži saldo Joint računa (\$10000)

Prikaži saldo Alis računa (\$50)

Promeni saldo na Joint račun ( \$10000)- \$1000

Promeni saldo na Alis račun ( \$50) + \$1000

### COMMIT

## Bob

### BEGIN TRANSACTION

Prikaži saldo Joint računa (\$10000)

Prikaži saldo Bob računa (\$45)

Promeni saldo na Joint račun ( \$10000)- \$1000

Promeni saldo na Bob račun ( \$45) + \$1000

### COMMIT

# Transakcija-Pesimistično zaključavanje

- Transakcija nije dovoljna za rešenje problema dirty read, već je potrebno obezbediti i *zaključavanje* od istovremene promene istog podatka.
- Ideja je da čim transakcija startuje podatak sa kojim transakcija radi se zaključava sve dok se ne završi transakcija (commit).

ID	Nickname	Balance	...
1	Joint	\$8000	...
2	Alice	\$1050	
3	Bob	\$1045	
...	...	...	

Bob čeka dok se Joint račun(čelija) ne otključa, transakcija sa leve strane ne završi koja traje 10 deo sekunde

**Alis**

Automatski se zaključava Joint račun

## **BEGIN TRANSACTION**

Prikaži saldo Joint računa (\$10000)

Prikaži saldo Alis računa (\$50)

Promeni saldo na Joint račun (\$10000)- \$1000

Promeni saldo na Alis račun (\$50) + \$1000

**COMMIT**

**Bob**

## **BEGIN TRANSACTION**

Prikaži saldo Joint računa (\$10000)

Prikaži saldo Bob računa (\$45)

Promeni saldo na Joint račun (\$10000)- \$1000

Promeni saldo na Bob račun (\$45) + \$1000

**COMMIT**

# Transakcija-Optimistično zaključavanje

- Pesimistično zaključavanje u našem slučaju je dobro za Alis ali ne i za Boba koji je morao da čeka
- Ovakav način zaključavanja može dovesti do toga da veliki broj ljudi treba da čeka ili do pojave grešaka.
- Međutim sama transakcija ne mora da znači da će doći do menjanja podataka već samo do čitanja.
- Optimistično zaključavanje dozvoljava da se više transakcija izvršavaju istovremeno sa ciljem da neće doći do konflikta

DBMS je otkrio konflikt sa drugom transakcijom, vraća se na početak transakcije

**Alis**

## **BEGIN TRANSACTION**

Prikaži saldo Joint računa (\$10000)

Prikaži saldo Alis računa (\$50)

Promeni saldo na Joint račun ( \$10000)- \$1000

Promeni saldo na Alis račun ( \$50) + \$1000

**COMMIT**

**Bob**

## **BEGIN TRANSACTION**

Prikaži saldo Joint računa (\$10000)

Prikaži saldo Bob računa (\$45)

Promeni saldo na Joint račun ( \$10000)- \$1000

Greška – dirty read je otkrivena - Rollback

# Procedure – Store Procedures

- Ukoliko često pišemo složene SQL upite koje ponavljamo vrlo često rešenje može da bude korišćenje procedura
- Procedura predstavlja skup SQL naredbi koja se čuva u bazi
- Slična je metodi ili funkciji u programskim jezicima
- Može se izvršiti više puta

```
SELECT * FROM Employee  
WHERE Salary > 50000  
ORDER BY LastName, FirstName
```

```
CALL HighlyPaid();
```

Pozivanje procedure kroz aplikaciju kojom se administrira baza ili kroz aplikaciju koja je napisana u nekom programskom jeziku



# Store Procedure – Ulazni Parametri

```
SELECT * FROM Employee  
WHERE Department = 'Sales' ;  
ORDER BY LastName, FirstName
```

# SQL Injection

- Procedure nas štite od SQL Injection napada
- SQL injection omogućava napadaču da dođe do osetljivih podataka ili da obriše postojeće
- Napad karakterističan na Web sajtovima
- Posmatramo jednostavnu formu, order stranicu koja se sastoji od Submit dugmeta i text box-a

Customer Number:

Vrednost iz text box-a preuzima web aplikacija PHP, JAVA, C# i kreira SQL kod

# SQL Injection

Customer Number:

Submit

Napadač unosi parametre na osnovu kojih pokušava da dobije prikaz svih korisnika iz baze tj. Proverava da li je naš sajt otporan na SQL injection

```
sqlString = "SELECT * FROM Orders WHERE CustomerID = 'x'; SELECT * FROM Users; --'"
executeSQL(sqlString);
```

```
SELECT * FROM Orders WHERE CustomerID = 'x';
SELECT * FROM Users;
--'
```

Bazi, aplikacija prosleđuje tri SQL upita

1. Ništa ne radi
2. Lista kompletu tabelu sa korisnicima
3. SQL komentarišemo navodnik da bi izašli iz string

# SQL INJECTION – Brisanje Tabele

Customer Number:

Submit



