1. NoSQL baze podataka

Tuesday, 27 October 2020 11:18

Predavanje:

OO DB
- omogucavale da cuvate kompleksne objekte I ponasanje za objekte.
- Kao Objekat I metode.
- Nisu uspele da se standardizuju.

Relacioni model
- 2d cuvanje podataka, efikasno je
- Imale su upite, pise se upit I dobijaju se informacije uzivo na osnovu tih upita

NoSQL baze
- Specificni zahtevi za web, mobilne I cloud aplikacije
- Distribuiranost
- Skalabilnost

NewSQL baze
- RDBMS (Relational Database Management System) - programski jezik koji se koristi da se komunicira sa podacima koji se nalaze u bazi (SQL)
  - Prednosti RDBMS:
    - Efikasno skladistenje podataka
    - Podrska za ACID transakcije
    - Podrska za kompleksne SQL upite
    - Ogromna tehnoloska baza (razliciti DBMS alati, programski interfejsi itd.)

  - ACID***
    - Atomicity (Atomicnost) - Sve ili nista. Cela transakcija da se izvrsi ili nijedna. Ako ima greske cela transakcija se ponistava.
      - Transfer novca u banci, ili se izvrsi cela transakcija ili se ukida cela

    - Consistency (Konzistentnost) - Mora sva ogranicenja I pravila da se postuju.
      - Ako je nesto not null ne moze da je null, nesto je broj a mi zelimo da pisemo tekst itd.

    - Isolation (izolacija) - Nijedna transakcija nema pristup nekoj drugoj transakciji koja nije zavrsena
      - Transakcije su vidljive samo u toj ne u drugim transakcijama sve dok se ne komituje.
      - Svaka transakcija je nezavisna za sebe.

    - Durability (trajnost) - kad se transakcija zavrsi, izmene transakcije su trajne ne mogu da se izgube.
      - Cak I da padne sistem ti podaci posle transakcije su zapisani

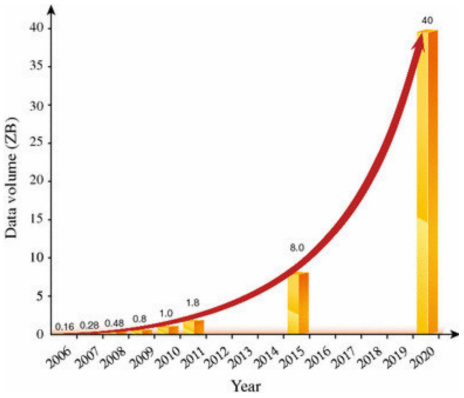
    - Sistem je sve ispuni ili nista (razlika od CAP dalje)

  - Transakcije definicija
    - Niz naredbi za manipulaciju podacima povezanih u jednu logicku celinu
      - Primer: Transakcija u banci, prebacivanje novca sa racuna na racun
        - Naredba je skidanje novca sa racuna 1 I dodavanje novca na racun 2
        - Ove dve naredbe cine smislenu celinu pa je zato dobra logicka celina

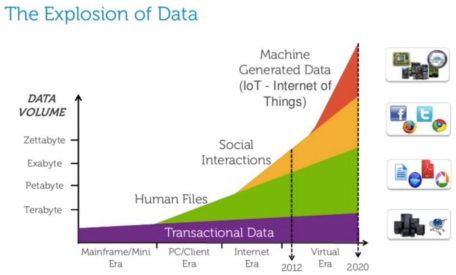
Podaci na webu

- Cetiri osnovne karakteristike podataka na Web-u:
  - Velika kolicina podataka
  - Povezanost podataka (relacije)
  - Polustruktuiranost podataka
  - Arhitektura aplikacija koje koriste podatke

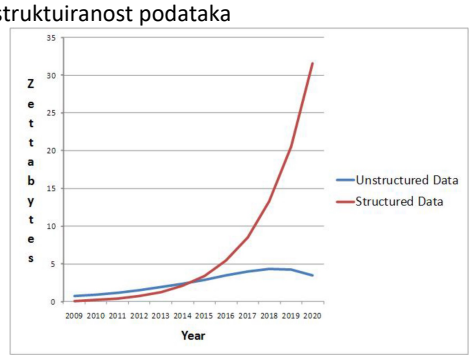
  - Velika kolicina podataka



Povezanost podataka (relacije)

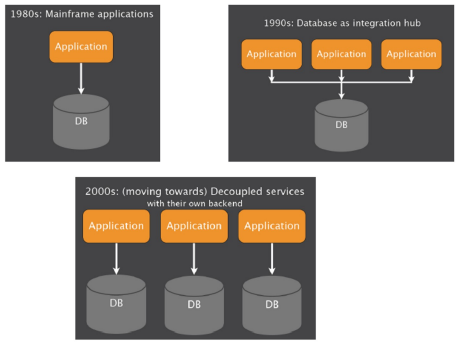


Polustruktuiranost podataka

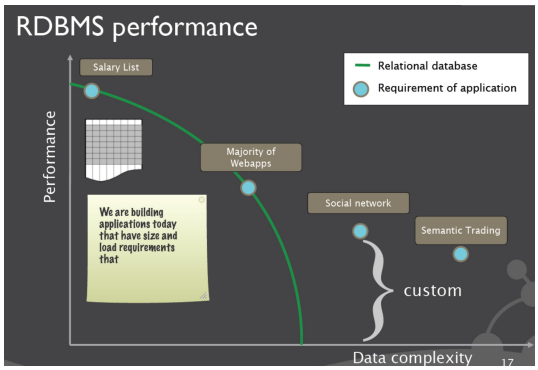


Crveno je nestruktuirano (greska u slici) to su podaci ljudi na socijalnim mrezama

Arhitektura aplikacija koje koriste podatke



Danas jedan app ima vise nanoservisa koji svaki ima svoju bazu



Kada koristiti relacione baze - kad je nesto prostije kao upis I transakcija plate (primer Salary list)

A kasnije se u zavisnsti od aplikacije koristi nesto drugo jer preformansa opada sto je kompleksnost veca

Web aplikacije zahtevaju:
- Ekstremno veliki broj transakcija u jedinici vremena
  - Amazon transakcije - veliki br ljudi kupuje uvek
- Dinamička analiza velikih količina podataka
  - Amazon recommend - sistem gde se analizira korisnik da se predloze proizvodi na osnovu pogledanog
- Kratko i predvidivo vreme odziva (latency)
  - Brz odaziv sadrzaja, skoro instantno
- Skalabilnost (po niskoj ceni)
  - Dodavati resurse kako rastemo al jeftino
- Visok nivo dostupnosti (high availability)
  - Uvek dostupno 24h/7 (kao facebook, goodle itd)
- Fleksibilnu šemu / polustruktuirane podatke
  - Fleksibilna sema - moze lako da se menja (kad se update radi ili menja app menja se I baza)
    - Kod relacionih nije laka izmena baze pa je to bio problem
  - Ima podrsku za struktuirane I nestruktuirane podatke
- Geografska distribuiranost (veći broj čvorova u kojima se podaci obrađuju, mreža kao problem)
  - Serveri globalni sto blizi korisniku da se smanji odaziv sajta

Skalabilnost - sposobnost sistema da nastavi da radi kad se dodaju novi resursi u cilju resavanja problema
- Povecan br korisnika/zahteva
- Povecan kolicina podataka
- Povecan br funkcijonalnosti I kompleksnosti koje se nude korisnicima

Scalink up
- Dodavanje resursa (dodavanje cpu, memorije, brzina konekcije …)

Prednost
- Brzo I jednostavno
- Aplikacija se automatski skalira

Nedostaci
- Prevazidje se kapacitet najaceg sistema - npr popunimo memoriju u racunaru, nema jaci cpu itd
- Cena - ne mozemo da biramo resenje nego resenje koje se uklapa u konfiguraciju koju imamo
- Zavisnost od samo jednog proizvodjaca - Ako imamo brend racunara I moramo njihovu memoriju da kupimo a skuplja je da bi scale up
- Single point of failure SPOF - sve ide na jedan racunar I ako taj padne sve pada

Scaling out
- Nastoji da resi probleme Scaling up-a

Availability (dostupnost sistema)
- Definiše se kao sposobnost korisnika da komuniciraju sa sistemom (slanje, ažuriranje ili preuzimanje podataka)

Downtime – period kada sistem nije dostupan
- Planirani – održavanje i nadogradnje sistema
- Neplanirani – otkazi usled hardverskih i softverskih grešaka

High availability = 24x7x365
- U praksi nastoji se da se minimizira downtime sistema i aplikacija (teži se 0)
  - NINES

Availability %	Downtime per Year	Downtime per Month	Downtime per Week	Downtime per Day
90%	36.5 days	72 hours	16.8 hours	2.4 hours
99%	3.65 days	7.20 hours	1.68 hours	14.4 minutes
99.9%	8.76 hours	43.8 minutes	10.1 minutes	1.44 minutes
99.99%	52.56 minutes	4.38 minutes	1.01 minutes	8.66 seconds
99.999%	5.26 minutes	25.9 seconds	6.05 seconds	864.3 milliseconds

Gleda se 3 ili 4 devetke da se nadje kod provajdera, 5 je najbolje al retko se nudi

High availability principi
- Redundantnost (Redundancy) - dizajn da sistem moze da nastavi da radi neometano cak I ako se desi otkaz nekog dela
  - Nema Single point of failure
  - Redundantnost je I softverska I hardverska
- Detekcija grešaka (Fault detection) -
- Oporavak (Repair/Recovery)
- Automatizovani i nenadgledani sisgtemi (Automated & Unattended)

RPO (recovery point objective) – količina podataka čiji se gubitak može tolerisati
RTO (recovery time objective) – downtime koji se može tolerisati
DB redundansa – softver i procesi

Vertikalna skalabilnost
- Jedan cvor za naplacivanje, jedan cvor za dodavanje u korpu itd

Horizontalna skalabilnost
- Svaki cvor radi istu stvar

Distribuirane baze podataka
- Više čvorova (servera) ponasaju se kao jedna baza podataka
- Fragmentacija - podaci su podeljeni po cvorovma
  - Vertikalna - izdvajaju se kolone
    - Svaka kolona ima primarni kljuc
    - PROJECT da podelimo tabelu
    - JOIN da spojimo tabelu
  - Horizontralna
    - Filtriramo redove na neki nacin
    - RESTRICT odvajamo redove
    - UNION spajamo redove
  - Mesano vertikalna I horizontalna

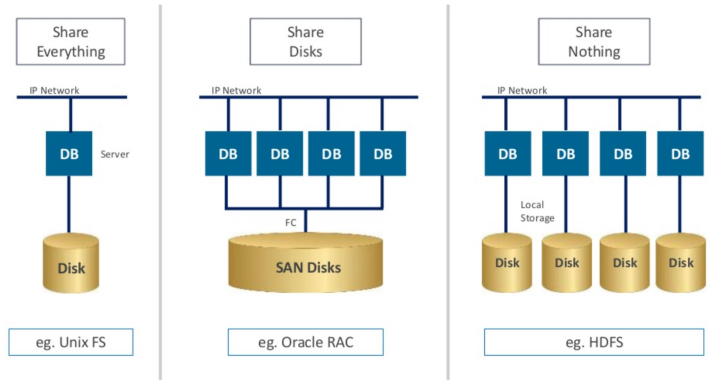
CustomerID	Name	Area	PaymentType	Sex
6	Smith	London	Cash	M
5	Patel	London	Card	F
8	Singh	Manchester	Card	F
9	Kodogo	Birmingham	Card	F
2	Rice	Manchester	Cash	M

CustomerID	Name	PaymentType
8	Singh	Card
2	Rice	Cash

Vertical and Horizontal Fragmentation

Customers in Manchester and their Payment Type

    - Horizontalna za ljude iz Manchester-a
    - Vertikalna za Name I PaymentType
- Replikacija - svaki podatak se moze javiti u vise kopija
  - Da bi se napravila kopija ili azurirale sve kopije treba mnogo vremena pogotovo ako su geografski udaljeni serveri



  - Share everything - sve na isto mesto u istoj bazi na istom disku
  - Share disk - vise database na jedan disk (kao kod kuce kad napravimo servere0
  - Share nothing - svaka baza ima svoj disk al treba da omogucimo adekvatno azuriranje podatka
- Skalabilnost
- Performanse

High availability
Otpornost na greške
Sa stanovišta korisnika ponaša se kao centralizovana baza podataka (jedna baza)

Prednosti:
- Bolja kontrola
- Bolje performanse
- Povećana dostupnost sistema
- Lakše skaliranje sistema

Nedostaci:
- Kompleksnost
- Cena
- Sigurnost - ne username I password nego pristup da odredjene adrese
- Otežana kontrola integriteta

Zahtevi koje moraju da ispune distribuirane baze podataka (CAP teorema)

Consistency - konzistentniost podataka posle svake operacije, svi vide iste podatke
Availability - uvek dostupan sistem, tolerancija na promene I otkaze
Partition tolerance - sistem nastavlja da radi cak I ako deo otkaze

CAP teorema: Sistem moze da ispuni 2 od 3 nije kao ACID sve ili nista
Zadovoljice se 2 pravila oko treceg se nalazi kompromis
Moze sva 3 al je preskupo da se zadovolji

CA: Consistency & Availability
Kompromis oko Partition tolerance, radi se kod single site cluster resenja
Primer dvofazni comit (2PC)
1. kordinator prima comit I kordinise sve cvorove da komituju
2. I kad svi se jave da su komitovali obavestava da je zavrseno komitovanje
   a) Ako jedan nije uspeo salje kordinator rollback svima da poniste transakciju

CP: Consistency & Partitioning
Kompromis oko Availavility, Pojedinim podacima privremeno moze pristup biti ogranicen ili onemogucen

AP: Availability & Partitioning
Kompromis kod Consistency, najcesce koriscen
Podaci mogu biti temporarily not up to date, zahteva conflict resolution jer ponekad su info razliciti pa koj podatak prihvatiti
Primer:
DNS kad zakupimo URL upisuje setaj URL u DNS server I treba vreme da se svima u svetu updatuje da je zakupljen

CAE trade-off (Amazon verzija CAP teoreme)

Cost-efficiency
High **A**vailability
Elasticity

Biraju se bilo koja dva (C, A, E)
Klijent čeka kada je sistem opterećen (C i E)
Ukoliko je moguće predvideti opterećenje, moguće je obezbediti A i C rezervisanjem resursa unapred
Nepotrebni resursi (over-provisioning) – A i E

BASE - CAP varijanta ACID svojstava

Basically Available - sistem odgovara na svaki zahtev, moze da se desi greska ili los podatak al vraca uvek odgovor
Soft State - Stanje sistema se menja tokom vremena čak i kada nema ulaznih podataka zahvaljujući svojstvu eventual consistency
Eventually Consistent - Sistem će dostići konzistentnost u određenom trenutku nakon što prestane da prima nove ulazne podatke

ACID forsira konzistentnost podataka dok BASE prihvata da će se konflikti desiti. Bitno je Soft State funkcija

NoSQL baze podataka **(video 3)**

NoSQL = Not Only SQL (sve nerelacione baze)

NoSQL su pokusaj da se poprave nedostaci relacionih baza podataka

Problemi koji se popravljaju:

Velika kolicina podataka
Veliki br transakcija u jedinici vremena
Transakcije su proste
Potreba za promenom baze podataka
Itd

Google pravi 2006 Bigtable whitepaper kao resenje nedostatka relacionih baza za potrebe indeksiranja internet pretraga
Spada u wide column baze podataka
Dynamo whitepapare je Amazonovo resenje 2007 spada u key value bazu podataka
Cassandra je 2008 Facebook napravio kao mix Googlovog I amazonovog resenja al je odustao od projekta
Voldemort je 2009 LinkedIn-ovo resenje

Postoje:
relacione baze podataka
NoSql - nerelacione baze podataka

Tipicna primena NoSQL baza
Velika kolici na podataka - koristi se distribuirana arhitektura (mnogo servera)
Veliki br upita - prosti upiti, redudantnost je potrebna kako bi se spojebi obezbedili
Shema evolution - obezbedi lako menjanje seme podataka

Prednosti:
Fleksibilnost
Skalabilnost - distribuirani sistemi koji se lako skaliraju
Eventualy consistent - CAP teorema, sistem uvek dostupan, ako particija padne ne pada sve
Jeftine - potrebno je uloziti mnogo ali je jeftino u celini
Prilagodjene potrebama Web aplikacija

Lose strane:
Nije stabilna - jer je nova tehnologija
Nema zajednickih standarda - svaki program je drugaciji
Losa podrska za transakcije - jer se ne skaliraju lako ili nemaju podrsku za transakcije
Losa podrska za pretrazivanje podataka - pretrazujes po kljucu, za komplikovane pretrage je problem
Zahteva promenu nacina razmisljanja - unapred se razmislja o upitima pa onda strukturu podesiti po upitima
Tesko naci dva identicna scenarija primene - svaka baza ima specifican usecase, razmisljaj o arhitekturi sistema

Taksonomija/ tipovi NoSQL baza
Key/value stores - pretraga je po kljucu, brzo je, a podaci su sta god (blob) baza ne zna strukturu blood-a samo vracaju ceo blob, Redis
Pretraga ne moze po sadrzaju
Column stores (Extensible records/ WhideColumn Stores) - Google big table - vise kolona ima, Cassandra
imamo kljuc I niz kolona sa podacima umesto bloob
Dve vrste ne moraju da imaju iste kolone
Document stores - kljuc dokument par, baza zna strukturu dokumenta I moze da ga obradjuje, moze pretraga I po sadrzaju dokumenta
MongoDB
Graph databases - ima cvorove I poteze, graf je, Neo4J