TEORIJA SA ZAVRŠNIH ISPITA

(3 boda)

2PC protokol

1. Što je **2PC** protokol i čemu služi?
2. Kod kojih sustava se koristi?
3. Koja su njegova svojstva?

Correct answer:

1. protokol koji osigurava atomarnost globalne transakcije (ili su sve subtransakcije obavljene ili nijedna)
2. u distribuiranim bazama podataka
3. 2PC protokol je blokirajući protokol, nije nezavisan s obzirom na mogućnost obnove

---

(2 boda)

Objasnite zašto se, u svijetu NoSQL tehnologija, smatra da **grafovske baze podataka odudaraju** od ostalih ("*Strange fish in SQL pond*")? Navedite i objasnite po stavkama što je grafovskim bazama podataka zajedničko, a što nije, s ostalim NoSQL tehnologijama.

Correct answer:

Objasnite zašto se, u svijetu NoSQL tehnologija, smatra da grafovske baze podataka odudaraju od ostalih ("Strange fish in SQL pond")?

- Razgrađuje podatke u još manje jedinice od RBP-a
- Nema agregatnog modela
- Nisu pogodne za distribuciju
- Sličnije relacijskim bazama, podržavaju transakcije (ACID)

Navedite i objasnite po stavkama što je grafovskim bazama podataka zajedničko, a što nije, s ostalim NoSQL tehnologijama.

- Zajedničko s ostalima: ne-relacijski model, popularnost, open source
- Što nije zajedničko - Pogodni za složene, polu-strukturirane, jako povezane podatke

(3 boda)

Navedite i objasnite kategorije konzistencije u sustavima tokova podataka.
Koju je najteže ostvariti?
Kako je ta konzistencija povezana s izvorištem i odredištem tokova?

Correct answer:

- Navedite i objasnite kategorije konzistencije u sustavima tokova podataka.

Tri kategorije konzistencije (consistency, correctness, accuracy,...):

1. At-most-once-processing
2. Al-least-once-processing
3. Exactly-once-processing/effectively-once processing
    - To ne znači da će sustav samo jednom (pokušati) obraditi događaj, već da će na izlaz biti proslijeđena samo jedna od tih obrada

- Koju je najteže ostvariti?

Exactly-once-processing/effectively-once processing

- Kako je ta konzistencija povezana s izvorištem i odredištem tokova?

Treba ostvariti exactly-once i u interakciji s izvorištima i odredištima

---

2 boda/2 points

Objasnite ulogu i osobine činjeničnih tablica te ulogu i osobine dimenzijskih tablica u zvjezdastom dimenzijskom modelu podataka.
Usporedite zauzeće prostora (memorije) od strane činjeničnih i dimenzijskih tablica

- za jednu ntorku
- po pitanju ukupnog broja ntorki u svakoj

Correct answer:

Činjenična tablica odgovara procesu koji se prati u skladištu podataka.
Sadrži dvije skupine brojčanih atributa:

- ključevi dimenzijskih tablica
- mjere (engl. measures)
  Često je (gotovo) identična odgovarajućoj tablici u relacijskom izvoru podataka - normalizirane je sheme.

Dimenzijska tablica predstavlja subjekt/objekt koji sudjeluje u procesu koji se prati u skladištu podataka.
Objašnjavaju činjenice pohranjene u činjeničnoj tablici, odnosno opisuju kontekst, npr.: koji student ja kada položio predmet kod kojeg nastavnika.
Dimenzijska tablica nije normalizirana.
Često nastaje spajanjem više relacijskih tablica.

Zauzeće memorije:

Činjenične tablice: velik broj zapisa (10^5, 10^6, 10^7, ...) jedan redak (n-torka) ne zauzima puno prostora (normalizirana tablica, numerički atributi)

Dimenzijske tablice: mali broj zapisa (<10^5, većinom <10^2) veliko zauzeće prostora po jednom retku (nenormalizirana, npr. 30 atributa, od toga 15-tak znakovni nizovi: 15 * 4 + 15 * 100 = 1560 byte)

(3 boda)

2PL protokol

1. Što je **2PL** protokol i čemu služi?
2. Kod kojih sustava se koristi?
3. Koja su njegova svojstva?
4. Postoji li alternativa tom protokolu i koja?

Correct answer:

1. Protokol zasnovan na zaključavanju za osiguravanje **serijalizabilnosti** transakcija u bazama podataka. Ima dvije faze: fazu rasta (postavljanje ključeva) i fazu sužavanja (otpuštanje ključeva)
2. Koristi se primarno u relacijskim bazama podataka, ali mogao bi se koristiti i šire.
3. Pesimističan, koristi zaključavanje, loše performanse
4. Drugi protokoli za serijalizabilnost transakcija, poput PostgreSQL-ovog SSI zasnovanog na MVCC-u ili trivijalnog serijskog obavljanja transakcija kao u VoltDB

(1+3 boda)

Fragmentacija (*sharding*)

a) (1 bod)
Objasnite problem preraspodjele fragmenata u repliciranom sustavu; kojima načelima se treba voditi prilikom preraspodjela?

b) (3 boda)
Navedite i objasnite strategije preraspodjele fragmenata.

Correct answer:

// ovo pitanje nije o tome kako napraviti sharding, tj. hash based i range based i sl. to se **NE traži**

(a) Vremenom opterećenje (količina podataka) raste te rastu veličina odnosno broj fragmenata i potrebno je napraviti preraspodjelu fragmenata, pri čemu se očekuje:

- Baza mora biti operativna za vrijeme preraspodjele
- Pravedna raspodjela opterećenja nakon preraspodjele
- Minimizirati mrežni promet prilikom preraspodjele Za to je potrebno koristiit neku vrstu consistent hashinga

(b) Postoje sljedeće strategije (vidjeti NoSQL4 S22-):

1. Fiksni broj fragmenata koji je napravljeni unaprijed, (npr. Riak ring)
    - Novi čvor preuzima svoj udio cijelih fragmenata od postojećih čvorova
2. Dinamički broj fragmenata (npr. Mongo)
    - Svaki fragment pripada jednom čvoru
    - Ako fragment naraste preko definirane veličine
        - Razdijeliti na dva jednaka dijela
        - Potencijalno jedan fragment prebaciti na drugi čvor
3. Fiksni broj fragmenata po čvoru (npr. Cassandra)
    - Kada se novi čvor priključi, odabire slučajno postojeći čvor i podijeli s njim opterećenje
        - Prije: postojeći čvor s N fragmenata, ukupno X MB
        - Poslije:
            - postojeći čvor s N fragmenata, ukupno ~ X/2 MB
            - novi čvor s N fragmenata, ukupno ~ X/2 MB

(2+1 boda)

Tokovi podataka:

a) (2 boda)
Koje pristupe prijenosu događaja preko mreže znate? Komentirajte njihova svojstva i razlike, navedite primjer sustava ako znate.

a) (1 boda)
Općenito, što se može dogoditi kada proizvođači šalju poruke brže nego što ih potrošači mogu primati? Komentirajte opcije.

Correct answer:

(a) Dva pristupa:

1. **Direktna komunikacija** od proizvođača do potrošača - moguć gubitak poruka, potrebno je napraviti vlastito aplikacijsko rješenje da se to spriječi ako se želi - npr. ZeroMQ
2. **Posrednički sustavi** - Posrednik je poslužitelj, a proizvođači i potrošači klijenti - Tipično asinkrona komunikacija, proizvođač samo preda poruku posredniku, koja naknadno biva dostavljena potrošaču - Ovdje imaju dva podpristupa:
    1. Message Broker/Queue
        - postoje i dva standarda: AMQP, JMS
        - npr. RabbitMQ
    2. Log-based
        - Proizvođači dodaju u dnevnik
        - Potrošači čitaju (ali ne brišu)
        - Radi performansi - dnevnik se razlaže u particije (i logički - teme)

(b) Moguće je:

- Odbaciti poruke
- Spremati poruke u red (dokad?)
- Kočiti proizvođače (backpressure, flow control), npr. unix pipes
- (kombinacija pristupa)

(4 boda)

Navedite i **objasnite** četiri kategorije obrade neograničenih tokova (*unbounded stream processing*).

Correct answer:

Četiri kategorije:

- A1: vremenski agnostičko (time-agnostic)
- A2: aproksimativno
- B1: pomoću prozora u vremenu obrade (processing time windowing)
- B2: pomoću prozora u vremenu događaja (event time windowing)

(dalje pogledati u predavanjima)

---

5. **(3 boda)** Što je svojstvo trajnosti (eng. *durability*)?
Komentirajte svojstvo trajnosti u relacijskim i NoSQL sustavima.
Što je i čemu služi oslabljivanje trajnosti (eng. relaxing durability)? Navedite primjer.

Durability - izdržljivost - ako je transakcija obavila svoj posao, njezini efekti ne smiju biti izgubljeni ako se dogodi kvar sustava, čak i u situaciji kada se kvar dogodi neposredno nakon završetka transakcije

Obnova baze podataka povezana je sa svojstvom izdržljivosti (durability). Osnovno pravilo koje omogućuje obnovu je redundancija – svaki se podatak mora moći rekonstruirati iz nekih drugih informacija redundantno pohranjenih negdje drugdje u sustavu. Redundancija se postiže dnevnicima izmjene - služe za poništavanje transakcija (očuvanje atomarnosti/nedjeljivosti) i ponovno obavljanje transakcija (očuvanje durability/izdržljivosti).

Relacijske BP:

Write-Ahead Log rule: izmjena se prvo zapisuje u dnevnik, a tek se onda provodi! – to omogućuje postojanje svojstva izdržljivosti. Dakle, prvo se zapisuje u dnevnik (koji je isto neka vrsta trajne memorije) pa se onda potvrdi promjena te se tek onda promjena provodi i zapisuje na disk.

NoSQL => durability vs performance trade-off – oslabljuje se trajnost kako bi se dobilo na performansama

Ekstremno oslabljivanje trajnosti u NoSQL-u su in-memory BP.

Oslabljivanje izdržljivosti/trajnosti događa se čak htjeli mi to ili ne htjeli u master-slave scenariju. Dakle, master obavi update i prestane raditi prije nego što promjene propagira svojim slave-ovima pa oni odabiru novog mastera => narušavanje trajnosti jer mi očekujemo da su sve potvrđene promjene stvarno obavljene. Kada pravi master opet proradi => trade-off: možemo zahtijevati da master obavi barem jednu replikaciju prije nego potvrdi klijentu

Primjer za Riak (skroz suprotno od Write-Ahead Log-rule u relacijskim bazama): objekt se prvo zapiše u memoriju (buffer – dakle, ne na trajni medij kao što je log dnevnik) pa se onda potvrdi uspješna promjena/pisanje pa se tek onda ta promjena zapisuje na disk => više prostora da nešto pođe po zlu u smislu očuvanja izdržljivosti

6. (**3 boda**) Objasnite mehanizam kvoruma kod ostvarivanja konzistencije. Navedite primjere i za čitanje i za pisanje.

6. Zadatak

Za konzistenciju kod pisanja dovoljna je većina: W > N/2
Za konzistenciju kod čitanja dovoljno je: W + R > N (konzistencija kod čitanja ovisi o W)

Možemo imati konzistenciju kod čitanja i kad nemamo kod pisanja - čitati sve čvorove! Što više čvorova je uključeno u operaciju bolja je konzistencija, ali veća latencija i obratno. Primjer: ako želimo brzo čitanje onda žrtvujemo pisanje: N=3, W=3, R=1

---

5. (5 bodova)
Što je Bloomov filter i čemu služi? Koja su njegova svojstva. Navedite parametre koje uzimamo u obzir kad projektiramo Bloomov filter i kako one utječu na njegovu točnost. Navedite primjer.

BF je probabilistička podatkovna struktura koja omogućuje kompaktno pohranjivanje informacije o članstvu u skupu na račun točnosti. BF može dati lažne pozitivne odgovore (da element jest član skupa), ali ne i lažne negativne. Razmatramo m - broj bitova BF-a, k - broj hash funkcija i n – očekivani broj elemenata u skupu. Veći broj bitova povećava točnost BF-a, veći n naravno smanjuje, a k ima neki optimum za m i n.

Teorijska pitanja iz starih završnih ispita

ZI 19/20

(2 boda) Koje vrste vremenskih prozora poznajete?

(3 boda) Što je svojstvo trajnosti (eng. durability) i kako se ostvaruje? Komentirajte svojstvo trajnosti u relacijskim i NoSQL sustavima. Što je i čemu služi oslabljivanje trajnosti (eng. relaxing durability)? Navedite primjer.
> Durability - izdržljivost - ako je transakcija obavila svoj posao, njezini efekti ne smiju biti izgubljeni ako se dogodi kvar sustava, čak i u situaciji kada se kvar desi neposredno nakon završetka transakcije.

Ostvaruje se tako da se podatci prvo pišu u logički dnevnik u trajnoj memoriji prije nego što se klijentu potvrdi pisanje, odnosno ostvaruje se pomoću logičkog dnevnika i WAL pravila.

U relacijskim sustavima D je standardno svojstvo u okviru ACID svojstava koje relacijski sustavi pružaju. U NoSQL sustavima durability nije nužno defaultno uključen, npr. u Riaku je po defaultu „relaksiran" tj. pisanja nisu durable pisanja. Mongo također omogućuje da se ne piše durable, odnosno to je parametar na razini zahtjeva.

Durability se tako može „oslabiti", naravno – kako bi se povećale performanse. Oslabiti znači ne poštivati WAL pravilo i kasnije, asinkrono, pisati u trajnu memoriju. Npr. Riak prvo potvrđuje pisanje tj. odgovara klijentu a tek naknadno pohranjuje u trajnu memoriju.

(4 boda)

a) **Koje vrste replikacijskih protokola poznajte? Kratko ih opišite.**
b) **Koje su im prednosti, a koji nedostatci?**
c) **Koju vrstu replikacijskih protokola koristi MongoDB, a koju Riak?**

Sync:

- ✓ osigurana potpuna konzistentnost
- ✓ dobre performanse pri čitanju
- **-** slabije pri izmjeni
- **-** produljenje trajanja transakcije, povećanje broja potpunih zastoja, niska dostupnost (ispadom samo jednog čvora operacije izmjene postaju neizvedive)

Async:

- ✓ visoka dostupnost, dobre performanse
- **-** velika mogućnost pojave nekonzistentnih podataka

1way (npr. MasterSlave):

- ✓ dobre performanse pri čitanju
- ✓ konzistencija
- **-** loše performanse kod pisanja, single point of failure

2way (tj. P2P):

- ✓ dobre performanse pri čitanju i pisanju
- **-** problemi s konzistencijom, posebice inconsistent writes

Mongo je MS, a Riak P2P.

ZI 18/19

(4 boda)

a) **Što je svojstvo izolacije u ACID kratici?**
   Kada se paralelno obavljaju dvije ili više transakcija, njihov učinak mora biti jednak kao da su se obavljale jedna iza druge.
b) **Koji su motivi za uvođenje izolacije, odnosno koje su prednosti i nedostatci?**
   Motivacija je da se što više koriste resursi poslužitelja. Izolacija omogućuje paralelizaciju. Ako NEMAMO izolaciju, odnosno imamo serijsko obavljanje transakcija onda resursi poslužitelja tipično stoje neiskorišteni (npr. CPU, memorija, itd.). Dakle, želi se obavljati više operacija istovremeno.
   Nedostatak je što je izolaciju složeno implementirati te može u ekstremu i narušavati performanse – npr. dvije transakcije mogu ući u potpuni zastoj (deadlock).
c) **Kakav je to serijalizabilan poredak obavljanja naredbi transakcija? Navedite primjer.**
   Poredak obavljanja naredbi transakcije koji nije serijski, ali ima učinak izvršavanja jednak nekom serijskom poretku obavljanja ukupnih transakcija. Primjer: P06S34
d) **Koje pristupe implementaciji serijalizabilnosti/izolacije poznajete? (ne treba objašnjavati u detalje)?**

   1. Trivijalno -> serijsko (linearno) izvođenje (npr. VoltDB)
      a. Vrlo pesimističan ~ lock cijele baze!
   2. 2PL - Two Phase Locking protokol zasnovan na zaključavanju
      a. Pesimističan
      b. Loše performanse
   3. Serializable Snapshot Isolation
      a. Snapshot -> MVCC
      b. Optimističan
      c. Bolje performanse (ovisno o načinu korištenja, npr. dobar za dugačke readonly transakcije)

(2 boda)

Što je 2PC protokol i čemu služi (ne treba objašnjavati protokol u detalje)? Kod kojih sustava se koristi? Koja su njegova svojstva?

2PC protokol je protokol potvrđivanja globalne transakcije. Koristi se (ali ne samo) u distribuiranim relacijskim bazama podataka.

- 2PC je blokirajući protokol.
- 2PC nije nezavisan s obzirom na mogućnost obnove.

(2 boda) Što su vektorski satovi i čemu služe? U kojim sustavima ih koristimo?

Vektorski satovi su podatkovne strukture koje služe za detekciju (ali ne i razrješavanje!) konflikata u distribuiranom multi-master (P2P) sustavu.

Koristimo ih kod multi-master (P2P) sustava (npr. NE koristimo ih kod M-S).

a) Definirajte pojam watermark u domeni obrade tokova.

F(P) -> E

- Za vrijeme P vraća event time E
- Za to vrijeme E sustav vjeruje da su svi događaji s e < E već viđeni

b) Koje vrste watermarka poznajte? Objasnite na primjeru.

1. Idealni watermark

- Kad imamo savršeno poznavanje ulaznih podataka nekad možemo definirati savršeni watermark, takav da nema zakašnjelih događaja

2. Heuristički watermark

- Educated guess, pokušavamo predvidjeti kada je obrađen najstariji zapis, tipično postoje zakašnjeli podatci

ZI 17/18

(4 boda) Što je Write-Ahead Log rule? Čemu služi? Kod kojih sustava se koristi?

(5 bodova)

a) **Što je replikacija?**
b) **Koji su motivi za uvođenje replikacije?**
c) **Koje dvije osnovne vrste replikacijskih protokola poznajte s obzirom na brzinu propagiranja promjena (opišite načelo) i koje su njihove odlike?**

(5 bodova)

a) **Iznesite CAP teorem uz objašnjenje C, A i P pojmova.**
b) **Objasnite na primjeru.**
c) **U kakvoj je vezi latencija (odgovora) s CAP teoremom?**

(3 boda)

Definirajte pojam ontologija u kontekstu semantičkog weba. Detaljno objasnite definiciju, tako da objasnite značenje svakog pojedinog dijela definicije.

ZI 16/17

(3 boda) Navedite replikacijske modele kod NoSQL sustava i komentirajte njihove prednosti i mane.

(3 boda) Što je combinable reducer i koje su mu prednosti i ograničenja. Objasnite na primjeru.

(2 boda) Neformalno opišite 2PC (two-phase commit) protokol, čemu služi i kada se koristi, te koja su mu svojstva (ne morate reproducirati dijagram).

(2 boda) Što je shema distribucije kod DSUBP-a i od čega se sastoji. Objasnite na primjeru.

(2 boda) Objasnite svojstvo D iz akronima ACID, te objasnite kako se ono ostvaruje u relacijskim bazama podataka?

ZI 15/16

(3 boda) Koja od svojstava transakcije u relacijskim sustavima je teško ostvariti u distribuiranoj okolini i zašto?

(3 boda) Objasnite CAP teorem. Navedite primjer u kojem u slučaju mrežne particije nemamo samo binarni izbor.

(3 boda) Što je combinable reducer i koje su mu prednosti i ograničenja. Objasnite na primjeru.
(3 boda) Što je ontologija (u kontekstu Semantičkog weba) i čemu služi? Od čega se sastoji (što je moguće opisati pomoću) OWL-a?

ZI 14/15

(4 boda)

a) Komentirajte potporu za transakcije u NoSQL sustavima.
b) Kakva je povezanost modela podataka s (ne)mogućnošću obavljanja transakcija u NoSQL sustavima?
c) Objasnite pomoću primjera.

(4 boda) Definirajte pojmove aditivna i semiaditivna mjera. Na primjeru komentirajte svojstvo

semiaditivnosti mjere.

(3 boda) Objasnite „Open World“ i „Unique Name“ pretpostavke. Vrijede li one u OWL-u?