Lekcija8 – Upravljanje performansama

1. **Navesti i objasniti 5 faktora koji utiču na performansu baze podataka**
2. *Obim posla*– kombinacija online transakcija, batch poslova, ad hoc upiti, analiza skladišta podataka, i sistemske komande usmerene kroz sistem u bilo kom trenutku

*2. Propusna moć*- definiše uopštenu mogućnost kompjutera da obradi podatke. To je mešavina I/O brzine, CPU brzine, mogućnosti mašine za paralelnim izvršavanjem, i efikasnost operativnog sistema i sistemskog softvera.

*3. Resursi–* kombinacija hardverskih i softverskih alata koji su na raspolaganju sistemu

*4. Optimizacija* – proces omogućavanja da se stvari izvršavaju što efektivnije moguće

5. Takmičenje– stanje gde dve ili više komponenti od ukupnog posla nameravaju da koriste jedan resurs na konfliktan način

**2. Definisati pojam performanse baze podataka.**

Performansa baze podataka može da se definiše kao optimizacija korišćenja resursa da bi se povećala prohodnost i minimiziralo takmičenje, omogućavajući da se najveći mogući obim posla odradi.

**3.Koji elementi sistema i kojim redosledom se analiziraju kada se rešava problem performanse baze?**

Pravilo 80/20

80% rezultata podešavanja dolazi od 20% napora podešavanja

Progres podešavanja-

*Razrešavanje problema*

* **Aplikacija**
* SQL
* Host Language Code
* **Baza podataka**
* Indexing
* Organizacija baze i indeksa
* Dizajn baze podataka (normalizacija / denormalizacija)
* **Sistem / Podsistem**
* Sistemski param, Pools, Lokovanje, address spaces, etc.
* **Okruženje**
* Network
* TP Monitor
* Disk
* Operating System

**4. Šta je monitoring baze podataka i koja je njegova uloga ?**

* Proces identifikovanja problema

• Monitor sakuplja važne informacije koje će služiti za odlučivanje u vezi podešavanja performanse, ali je suštinski neinteligentan.

**5. Šta obuhvata proces upravljanja performansama baze podataka?**

* Upravljanje performansama baze podataka obuhvata sve navedeno:

– Monitoring za pronalaženje problema

– Analiza za identifikaciju korektivnih akcija

– Optimizacija koja donosi promene za sisteme i aplikacije koji bi bili sa boljim performansama

• Pravo upravljanje promenama može da se postigne samo korišćenjem proaktivnog plana performanse.

**6. Koja je razlika između proaktivnog I reaktivnog upravljanja performansama baza podataka?**

* **Proaktivno**

– Promišljenost

– Planiranje

– Korektivne akcije se izvode pre nego što se dese problemi

– Automatizovani

– Minimizira reaktivno monitorovanje i zahteve za podašavanje

**• Reaktivno**

– „Gašenje požara

“ – Problemi postoje kada treba da se isprave

– Neplanirani problemi

– Ne mogu biti totalno eliminisani

**\***Proaktivno upravljanje performansama redukuje količinu vremena, truda, i ljudske greške koje su uključene u implementaciji i održavanju efikasnih sistema baze podataka

**7. Definisati service-level management. Dati primer izjave servisnog nivoa.**

* Service-level management (SLM) je “disciplinovana, proaktivna metodologija i procedure koje se koriste da osiguraju adekvatni nivo servisa i koje su isporučene svim IT korisnicima u skladu sa prioritetima poslovanja i po prihvatljivoj ceni.”\*

• Servisni nivo je mera operacionog ponašanja.

• SLM može da se fokusira na dostupnost, performanse ili oba.

**Primeri izjava servisnog nivoa**

• Primeri u terminima dostupnosti:

– “99.95% uptime od 9:00 A.M. do 10:00 P.M. radnim danima.”

• Servisni nivo može biti i precizniji:

– “Prosečno vreme odgovora za transakcije će biti 2s ili manje, za obim posla od 500 .”

**8. Koji su faktori uspeha za service level agreement?**

* Da bi service-level agreement (SLA) bio uspešan, svi učesnici moraju da se dogovore o iskazanim ciljevima za dostupnost i performansu.

– Krajnji korisnici moraju da budu zadovoljni sa performansom svojih aplikacija

– DBA i tehnička lica moraju da budu sigurni u svoje sposobnosti da upravljaju sistemom da bi ga doveli do postavljenog cilja

– Kompromis je esencijalan da bi se dostigao korisni SLA

**9. Navesti i objasniti vrste podešavanja performansi.**

* Podešavanje sistema

– Instance baze podataka ili samog podsistema i bilo kojih parametara sistema i detalja postavljanja

• Podešavanje baze podataka

– Strukture baze podataka, DDL/parametri i organizacija podataka

• Podešavanje Aplikacije

– Kod aplikacije i SQL

Lekcija9 – Performanse sistema

**10. U kakvoj vezi su performance sistema sa performansama baze podataka?**

**11. Na koji način performanse I/O uređaja utiču na performance DBMSa?**

* Jedan od najvećih „uskih grla“ za performansu baze podataka jeste fizička cena I/O operacija
* Podaci se nalaze na disku, a disk je mehanički uređaj.
* SSD (solid state device) mogu da poboljšaju performanse.

– SSD je ustvari memorija kompjutera koja je konfigurisana da radi kao disk uređaj

**12. Navedi vrste keša.**

• Keš podataka

• Kež procedure

• Keš sortiranja

• Keš interne strukture

• Keš loga baze podataka

**13. Objasniti keširanje internih struktura.**

* Da bi izvršio relacione operacije, DBMS možda treba da kreira interne strukture koje nisu obavezno vidljive krajnjem korisniku.
* Međutim, DBA, i ponekad programeri, trebaju da znaju nešto o internim strukturama.
* Ove interne strukture su često keširanje u memoriji kako su korišćene.

– Primer: DBD (database descriptor)

**14. Zašto se koristi keširanje logovanja upisa?**

Keširanje logovanja upisa se koristi za ubrzanje modifikacije baze podataka. Baferovanje logovanja upisa, logovanje baze postaje manje „usko grlo“ sistema i performanse aplikacije.

**15. Za šta se koristi keširanje logovanja čitanja?**

* Keširanje logovanja čitanja se koristi za operacije ROLLBACK i RECOVER

– Rollback ili recovery treba da pristupa logu da bi se poništile ili ponovo izvršile promene u bazi podataka

– Kako se ovi log rekordi traže, oni će biti baferovani u memoriji, u kešu logovanja čitanja

**16. Koji su razlozi za kreiranje dodatnog prostora u memoriji (navesti bar 4)?**

* **Konekcije korisnika.**

– Svaka konkurentna konekcija korisnika na DBMS, bez obzira na vrstu konekcije klijenta, zahteva memoriju da bi DBMS održavao i upravljao konekcijom.

* **Uređaji.**

– Uređaji koje koriste baze podataka, mogu da zahtevaju sistemsku memoriju za održavanje u korišćenje.

* **Otvorene baze**

– Većina DBMS-a omogućava parametar kojim se određuje maksimalni broj baza koje mogu da budu otvorene u isto vreme. Svaka otvorena baza zahteva DBMS memoriju.

* **Otvoreni objekti.**

– Drugi parametrni mogu da postoje koji će identifikovati maksimalan broj objekata baze podataka koji mogu biti otvoreni u jednom trenutku. Uključujući tabele, indekse i druge objekte baze podataka koje se koriste. Svaki otvoreni objekat zahteva memoriju.

* **Brave.**

– Svaka brava koja se konkurentno drži će zahtevati memoriju. DBMS treba da pruži konfiguracioni parametar za broj konkurentnih brava koje se istovremeno drže.

* **Keš.**

– Razne vrste keš memorije su već diskutovane.

**17. Od čega zavisi efikasnost keša podataka?**

* Efikasnost keša podataka zavisi od ispravne procene veličine keša

– Suviše veliki: troši memoriju

– Suviše mali: česti upisi i zamena

* Neki DBMS-ovi, kao što su DB2, pružaju više bafera koji mogu da se konfigurišu i podešavaju nezavisno sa više parametara
* Drugi, kao što su SQL Server, su više osnovni, sa kešom podataka po bazi podataka. Ali bez obzira na DBMS, DBA bi trebalo da monitoruje efikasnost svakog keša podataka ili bafera.
* Efikasnost čitanja keša podataka je procenat koji prati koliko keš obavlja svoju osnovnu funkciju-izbegavanje I/O operacija

**18. Zašto je važno efikasno čitanje? Kako rešiti problem pada efikasnosti čitanja ispod 80%?**

• Efikasno čitanje je važno zato što pokazuje procenat koliko pita je stranica podatka pronađena u kešu podataka (ili baferu)

– Ciljati na 80% ili bolju efikasnost čitanja.

• Što je procenat veći, to je bolja efikasnost bafera. Kada se stranice podatka nađu u memoriji, bez zahteva za fizičkim I/O, performanse se povećavaju

• Stvaran broj I/O zahteva i fizičke I/O operacije se mogu naći u DBMS zapisima ili koristeći monitor za performanse baze podataka.

– Budite sigurni da ćete proučiti sve I/O zahteve, sinhrone kao i asinhrone

– Mnogi DBMS-ovi su u mogućnosti da predvide I/O zahteve i zakažu čitanje blokova pre nego što budu zahtevani.

**19. Objasniti log offloading.**

* Tokom vremena, log fajl će se napuniti podacima
* Log offloading je proces arhiviranja aktivnog loga u arhivni log i zamena upisa loga u novi aktivni log
* DBMS može da izvede automatski log offloading ili možete da ga izvedete manuelno (korišćenjem DBA komande)
* DBA mora da nauči kako svaki DBMS koji se koristi upravlja bekapom loga baze podataka

**20. Kada se dešava suspenzija brave?**

Suspenzija brave se dešava kada proces aplikacije traži bravu koju već drži drugi proces aplikacije i koji ne može da se deli. Suspendovan proces privremeno zaustavlja izvršavanje sve dok zahtevana brava ne postane slobodna.

**21. Kada se dešava timeout brave?**

Timeout se dešava kada se proces aplikacije prekine i završi zato što je bio suspendovan duže nego što je dodeljeni interval. Ovaj interval se obično podešava uz pomoć konfiguracionog parametra.

**22. Kada se dešava deadlock brave?**

Deadlocks se dešava kada sva ili više procesa aplikacije drže brave nad resursima koje drugi trebaju i bez kojih ne mogu da nastave sa radom. Ciklus za detekciju deadlocka je vremenski interval između provere za deadlockom-može takođe da se podesi koristeći konfiguracioni parametar.

**23. Šta je katalog Sistema? Navesti njihove nazive u bar 2 DBMSa.**

* Relacioni DBMS zahteva katalog sistema, ali on može da se zove drugačije, u zavisnosti od određenog DBMS-a koji se koristi.

– U Oracle se naziva Data Dictionary.

– U SQL Serveru se naziva System Catalog.

– U DB2 se naziva ili DB2 Catalog ili System Catalog.

– U MySQL, se naziva Information Schema

**24. Kako može da se reši problem performance vezane za katalog sistema?**

* Fizička lokacija i podešavanje kataloga sistema će imati uticaj na performanse sistema.
* DBA mora da odluči gde će biti instaliran, na kojem disku i koliko prostora će alocirati.
* Ove odluke se obično donose u vreme instalacije.
* Smestiti katalog sistema na odvojenom disku da bi moglo njime da se upravlja i podešava odvojeno od ostalih podataka aplikacije.

**25. Šta su ugnježdeni pozivi trigera? Koje probleme mogu da izazovu? Kako se mogu kontrolisati?**

Ugnježdeni poziv trigera je kada jedan triger čini da se drugi triger startuje. Neki DBMS-ovi pružaju dodatnu kontrolu ugnježdavanja trigera tako što daju maksimalan broj. Postavljanjem ovog broja, DBA može da kontroliše koliko nivoa ugnježdenog poziva trigera je dozvoljeno.

Lekcija10 – Performanse baze

**26. Šta je particionisanje u bazama podataka?**

* Tabela baze podataka je logička manifestacija skupa podataka koji su fizički smešteni u kompjuterskom skladištu.
* Svaki DBMS pruža različite mehanizme za fizičke fajlove u tabelama podataka.
* DBA mora da izabere od sledećih mogućnosti preslikavanja za svaku tabelu:

– Jedna tabela u jedan fajl

– Jedna tabela u više fajlova

– Više tabela u jedan fajl

**27. Šta je paralelizam u bazama podataka? Kako particionisanje može da pomogne u postizanju paralelizma?**

* Paralelizam je proces korišćenja više zadataka za pristup bazi u paraleli.
* Particionisanje pomaže postizanju paralelizma.
* Paralelni zahtev može da se izda za korišćenje većeg broja simultanih sistema za obradu čitanja, za samo jedan SQL zahtev.

– Paralelizam je poželjan zato što može da redukuje vreme potrebno za izvršavanje upita.

**28. Navesti vrste paralelizma u bazama podataka.**

* Mnogi tipovi paralelizma su bazirani na resursima koji mogu biti pozvani u paraleli.

– Jedan upit može biti razbijen na nekoliko zahteva od kojih svaki koristi različito CPU jezgro u paraleli

– Kao dodatak, paralelizam može da se poboljša proširivanjem posla duž nekoliko instanci baza podataka.

– Svaki DBMS nudi druge nivoe podrske za paralelne upite.

• Da bi se optimizovala performansa baze podataka, DBA bi trebalo da bude svesan ponuđene podrške koju nudi sam DBMS kojim se upravlja i mogućnosti da ekspolatiše mogućnosti paralelnog upita

**\***Procesor sa više jezgra je jedna komponenta kompjutera sa dva ili više nezavisnih procesora (koji se zovu „jezgra“)

**29. Zašto su indeksi korisni?**

Indeksi se koriste da bi se poboljšale performanse Kreiranje korektnih ideksa nad tabelama u bazi podataka je možda najveća tehnika podešavanje optimizacije koju može DBA da izvede. Indeksi su posebno korisni za:

• Lociranje redova po vrednostima u kolonama

• Pravljenje spajanja efikasnijim (kada je indeks definisan nad kolonama spajanja)

• Korelaciju podataka duž tabela

• Agregaciju podataka

• Sortiranje podataka da bi se zadovoljio upit

**30. Kako indeksi prekoračuju liniju između podešavanja baze i aplikacije?**

* Bez indeksa, sav pristup podacima u bazi podataka bi trebalo da bude izvršen pretraživanjem svih raspoloživih redova.
* Pretraživanje je veoma neefikasno za velika tabele.
* Dizajniranje i kreiranje indeksa za tabele baze podataka ustvari prelazi liniju između podešavanja performanse baze podataka i podešavanja performanse aplikacije.
* Indeksi i objekti baze podataka su kreirani od strane DBA sa DDL-om baze podataka. Međutim, indeksi su kreirani da bi se SQL iskazi u aplikaciji brže izvršavali.

**31. Kako se može proceniti efekat dodavanja indeksa?**

* DBA bi trebalo da razume obrasce pristupa tabeli nad kojom se indeks kreira.
* Korisne informacije uključuju:

– Procenat upita koji pristupaju podacima a ne ažuriraju ih

– Granica performanse koja je postavljena u SLA, a vezana je za upite nad tabelom

– Uticaj dodavanja novog indeksa na izvršavanje pomoćnih zadataka baze podataka, kao što su učitavanje, reorganizacija i oporavak

**32. Kada indeksi poboljšavaju pefromanse a kada ih degradiraju?**

* Indeksi mogu da poboljšaju performanse upita
* Indeksi će degradirati performanse brisanja i dodavanja
* Indeksi mogu da degradiraju performanse ažuriranja

**33. Kako se odlučuje i šta se indeksira?**

* Većina indeksa bi trebalo da bude dizajnirana tako da podržava što više upita.

– Detaljan test performansi upita koje svaki indeks podržava.

* Eksperimentisati sa različitim kombinacijama indeksa i meriti rezultate.
* Kreiranje indeksa da podržava samo jedan upit jeste prihvatljivo ukoliko je taj upit dovoljno važan za poslovanje.

– Ukoliko se upit izvršava retko, razmisliti o kreiranju indeksa pre početka obrade i uklanjanju indeksa kada se obrada završi.

**34. Kada treba izbegavati indekse?**

* + Kada svi pristupi dohvataju svaki red u tabeli.

– Pošto se svaki red dohvata svaki put kada se koristi tabela, ako se koristi indeks time će se dodati ekstra I/O pristup i performanse će se smanjiti, a ne povećavaju. Iako to nije često, možda se desi da naiđete na takve tabele u vašoj organizaciji.

* + Za veoma male tabele koje imaju samo nekoliko strana podataka i bez primarnih ključeva ili zahteva za jedinstvenost

– Veoma mala tabela (možda 10 ili 20 stranica) možda i ne treba indeks zato što je jednostavno čitanje svih stranica već veoma efikasno.

* + Kada performanse nisu važne i tabelama se retko pristupa.

– Da li ima takvih tabela u realnom životu?

* + Možda želite da izbegnete indeksiranje kolona koje imaju promenljivu dužinu. Proširivanje može da utiče da indeksi potroše prekomernu količinu prostora na disku.

– Međutim, ukoliko se kolone sa promenljivom dužinom koriste u SQL WHERE klauzuli, trošak prostora na disku mora da se uporedi sa troškom pretraživanja. Kupovina nekog dodatnog diska je često jeftinija nego trošak CPU resursa za pretraživanje redova.

* + Kolone sa niskom kardinalnošću.

– Na primer: POL

**35. Šta je i zašto je nekada poželjno “preopteretiti” indekse.**

* Ponekad se performanse upita mogu povećati preopterećenjem indeksa.
* Posmatrajte indeks nad SALARY i sledeći SQL upit:

Select emp\_no,last\_name,salary

From employee

Where salary > 15000.00;

* Dodavanjem EMP\_NO i LAST\_NAME u indeks, svi podaci se mogu dohvatiti iz indeksa čime se eliminiše I/O tabele i poboljšavaju performanse.

**36. Koja dva načina primene klasterovanja postoje? Objasniti.**

* Klasterovana tabela će smeštati redove fizički na disk sortirane po nekoj određenoj koloni ili kolonama.
* U zavisnosti od DBMS-a, podaci ne moraju uvek da budu fizički održavani u tačnoj sekvenci klastera.
* Kada je sekvenca klastera definisana za tabelu, DBMS će se ponašati na jedan od sledeća dva načina da bi primenio klasterovanje:

1. Kada se novi redovi dodaju, DBMS će fizički da premesti redove i stranice da bi postavio nove redove u definisanu sekvencu klastera.

2. Kada se novi redovi dodaju, DBMS će pokušati da smesti podatke u definisanu sekvencu klastera, ali ukoliko prostor nije na raspolaganju na zahtevanoj stranici, podaci se mogu smestiti negde drugde.

**37. Kada klasterovanje poboljšava performance?**

* Klaster nad kolonama spajanja, da bi optimizovali SQL spajanja gde više redova odgovara za jednu ili drugu tabelu koje učestvuju u spajanju
* Klaster nad kolonama stranog ključa zato što često učestvuju u spajanjima i DBMS pristupa vrednostima stranog ključa za vreme provere referencijalnog integriteta.
* Klaster nad predikatima u WHERE klauzuli
* Klaster nad kolonama opsega
* Klaster nad kolonama koje se često ne menjaju

– Redukuje fizičko reklasterovanje

* Klaster nad kolonama koje su često grupisane ili sortirane u SQL iskazima

**38. Da li klasterovati kolone primarnog ključa?**

* Klasterovanje generalno nije preporučivo za kolone primarnog ključa zato što je primarni ključ, po definiciji, jedinstven.

– Međutim, ukoliko su opsezi redova često selektovani i sortirani po primarnom ključu, klaster indeks može da bude delotvoran

**39. Šta je deljenje stranice? Koje vrste deljenja postoje? Objasniti svaku.**

* Proces kreiranja novih stranica da bi smestile ubačene podatke se zove deljenje stranice.
  + **Monotonično Deljenje Stranice**
  + Monotonično deljenje strana je mnogo jednostavniji proces, koji zahteva samo dva koraka. DBMS

1. Kreira novu stranu između pune strane i sledeće strane.

2. Ubacuje nove vrednosti u novu stranu.

* + Korisno kada su redovi uneseni u rastućoj sekvenci.

– Izbegavati trošenje prostora

**40. Zašto se ostavlja slobodan proctor?**

* Slobodan prostor se koristi da se ostavi deo prostora tabele ili indeksa praznim i na raspolaganju za smeštanje novo dodatih podataka.
* Ostavljanje odgovarajućeg slobodnog prostora može da:

– Redukuje učestalost reorganizacije

– Redukuje utrku za resursima

– Poveća efikasnost inserta

* Svaki DBMS nudi metode za specifikaciju slobodnog prostora za objekte baze podataka u CREATE i ALTER iskazima

**41. Navesti prednosti I mane slobodnog prostora.**

* Osiguranje odgovarajuće količine slobodnog prostora za svaki objekat baze podataka daje sledeće benefite:
  + • Insert je brži kada je dostupan slobodan prostor
  + • Kako se novi redovi dodaju, oni mogu da se odgovarajuće klasteruju
  + • Redovi promenljive dužine i izmenjeni redovi imaju prostora za proširenje , potencijalno redukovajući broj realociranih redova
  + • Manje redova na strani rezultuje boljom konkurencijom zato što je manje podataka nedostupno drugim korisnicima kada je stranica zaključana.
* Međutim, slobodan prostor takođe ima nekoliko mana.
  + Zahtevi za diskom su veći
  + Pretraživanja traju duže
  + Manje redova po strani može da zahteva više I/O operacija za pristup zahtevanim informacijama.
  + Pošto broj redova po strani opada, efikasnost keširanja podataka može da opada zato što se manje redova dohvata po I/O operaciji.

**42. Kako se određuje veličina slobodnog prostora?**

Tačna veličina slobodnog prostora mora da se bazira na sledećim kriterijumima:

* + Učestalost inserta i modifikacija
  + Broj sekvencijalnih u odnosu na nasumičan pristup
  + Uticaj pristupanja neklasterovanim podacima
  + Vrsta obrade
  + Verovatnoća ulančavanja redova, migracije redova i deljenje strane

**43. Šta je kompresija u bazama podataka I kakav uticaj ima na performance?**

* Kompresija može da se koristi da bi se smanjila veličina baze podataka.

– Kada se kompresija specifikuje, podaci su algoritamski kompresovani po insertu u bazu i dekompresovani kada se čitaju.

* Kompresija uvek zahteva kompromis koji DBA mora da analizira.

– Sa pozitivne strane, imamo uštedu diska, i potencijalno smanjenje I/O troškova (zato što se više redova može smestiti na strani ili bloku).

– Sa negativne strane, imamo dodatan trošak CPU koji zahteva kompresiju i dekompresiju podataka.

**44. Kako treba smestiti logove baze podataka da bi se poboljšale performance. Objasniti.**

* Smeštanje loga transakcije na odvojeni disk uređaj od diska sa podacima, dozvoljava DBA da bekapuje logove transakcije nezavisno od baze podataka.

– Takođe minimizuje duple upise na isti disk

– Upis podataka u dva fajla, na istom disku, u isto vreme, će degradirati performanse čak i više nego čitanje podataka iz dva fajla, na istom disku, u isto vreme.

– Setite se, svaka modifikacija baze (upis) je smešten u log transakcije

**45. Šta su strane? Koje su standardne veličine strana u DBMSovima. Dati primer.**

* Redovi su smešteni fizički u stranicu tabele ili bloku.
* Većina DBMS-a ograničava veličinu strane/bloka koji može da se izabere.
* Primeri:

– DB2 ograničava veličine strana na 4K, 8K, 16K, or 32K

– SQL Server samo podržava 8K

* DBA mora da izračuna najbolju veličinu strane na osnovu veličine reda, broja redova po strani, i DBA zahteva za slobodnim prostorom.
* Primer veličine stranice:
  + U DB2, koja veličina strane bi trebalo da bude izabrana ukoliko je zahtevano 0% slobodnog prostora i veličina zapisa je 2500 B?
  + Najjednostavniji odgovor je 4K, alo možda nije i najbolji.
  + 4K strana bi držala jedan 2500-byte rekord po strani, ali 8K strana bi držala tri.
  + 8K strana može da pruži efikasniji I/O, posebno ukoliko se podatku pristupa sekvencijalno (umesto randomly).

– Ovo je zato što čitanje 8K podataka bi vratilo 3 reda, dok čitanje 8K podataka koristeći dve 4K stranice bi vratilo samo dva reda.

**46. Koji su uzroci dezorganizacije baze podataka? Navesti I objasniti.**

* Uzroci dezorganizacije baze podataka:
  + Neklasterovani podaci. Klasterovana tabela ili indeks može da postane neklasterovan kako se podaci dodaju i menjaju. Ukoliko podaci postaju značajno neklasterovani, upiti koji su optimizovani za pristup podacima, ne mogu da imaju benefit klasterovane sekvence što ima uticaja na performanse.
  + Fragmentacija je stanje u kome postoje puno rasutih prostora za smeštanje u bazi podataka koji su mali da bi se mogli koristiti produktivno. To rezultuje izgubljenim prostorom, koji može da ugrozi performanse zato što su dodatni I/O zahtevani da bi se isti podaci dohvatili.
  + Ulančavanje reda ili migracija reda se dešava kada ažurirani podaci ne odgovaraju prostoru koji trenutno okupira, i DBMS treba da nađe prostor na red. U svakom slučaju, pokazivač se koristi da locira ili ostatak reda ili ceo red, čime se performanse smanjuju.

– Sa ulančavanjem reda, DBMS pomera deo novog, većeg reda, na lokaciju unutar prostora tabele gde postoji slobodan prostor.

– Kod migracije reda, ceo red se smešta negde drugde u prostoru tabele.

– Deljenje strane može da uzrokuje dezorganizovane baze podataka, takođe. Ukoliko DBMS izvodi monotonična deljenja stranice onda kada treba da uradi normalnu podelu stranice, ili obratno, prostor je izgubljen. Kada je prostor izgubljen, manje redova postoji na svakoj stranici, čime DBMS treba da uradi više I/O zahteva da bi dohvatio podatke. Zbog toga, performansa opada,

* + Proširenje fajla može negativno da utiče na performanse. Proširenje je dodatni fajl koji je povezan sa originalnim fajlom i može da se koristi samo zajedno sa originalnim. Kada nema mesta u fajlu koji se koristi za prostor tabele, proširenje se dodaje da bi se fajl produžio.

**47. Čemu služi naredba REORG? Da li se prilikom izvršavanja naredbe baza zaustavlja ili ne?**

* Prostori tabela i indeksi mogu da budu reorganizovani.
* Kako DBA izvršava REORG alat zavisi od DBMS-a.

– Neki DBMS proizvodi se isporučuju sa ugrađenim alatima za reorganizaciju.

– Drugi zahtevaju da klijent kupi alat.

– Neki tvrde da klijentu ne treba alat sve dok koristi njihov DBMS.

* + U praksi, ova poslednja tvrdnja nije tačna. Svaki DBMS nosi neki nivo dezorganizacije kako se podaci dodaju i modifikuju..i zato mogu da imaju korist od reorganizacije.

Lekcija11 – Performanse aplikacije

**48. Koji problemi dizajna aplikacije se analiziraju kada su performance aplikacije loše?**

Problemi dizajna koji treba da se analiziraju kada su performanse aplikacije

loše

• Vrsta SQL-a. Da li je ispravan tip SQL-a(planiran ili neplaniran, dinamički ili statički, embedovan ili samostalan)

korišten za ovu konkretnu aplikaciju?

•Programski jezik. Da li je programski jezik u stanju da dostigne zahtevane performanse i da

li je jezik optimizovan za pristup bazama podataka?

•Dizajn i obrada transakcija. Da li su transakcije u programu ispravno dizajnirane da bi

osigurale ACID osobine, i da li program koristi izabrani procesor transakcija efikasno i

ispravno?

•Strategija zaključavanja. De li aplikacija drži pogrešan tip brave, ili da li drži ispravan tip

brave predugo?

•COMMIT strategija.Da li svaki program aplikacije izdaje SQL COMMIT iskaze da bi minimizirao uticaj zaključavanja?

•Obrada Batch programa.Da li su batch programi dizajnirani odgovarajuće da bi imali korist

od osobina sekvencijalne obrade DBMS-a?

•Online obrada. Da li su onlajn aplikacije dizajnirane da bi vratile korisne informacije i da bi

minimizirale količinu informacija koje su prikazane na korisnikovom ekranu pri pozivu

programa?

**49.Šta je optimizator i čemu služi?**

* Optimizator je srce relacionog DBMS-a.

-Optimizator je engine zaključivanja koji određuje strategiju navigacije baze podataka.

* Developer definiše koji podaci su mu potrebni kodiranjem SQL iskaza.
* ... DBMS daje informacije o tome gde su podaci locirani, i...
* ...relacioni optimizator odlučuje kako efikasno da se navigira kroz bazu.
* Krajnji korisnik ne treba da zna gde i kako su podaci smešteni. Optimizator zna ove informacije.

**50.Na šta se odnosi fizička nezavisnost podataka u DBMS-u?**

* Relaciona optimizacija dozvoljava upitima da se prilagode na promenljivo okruzanje baze podataka.
* Optimizator moze da reaguje na promene formulisanjem novih putanja pristupa bez zahteva za promenom u kodu aplikacije.
  + Aplikacija moze da bude fleksibilna kako se tabele prosiruju ili smanjuju, kako se indeksi dodavaju ili uklanjaju, I kako baza postaje dezorganizovana ili reorganizovana.
* Ovo razdvajanje kriterijuma pristupa od karakteristika fizičkog smeštanja se zove fizička nezavisnost podataka.

**51.Šta je optimizacija upita?**

•Iako svaki RDBMS ima relacioni optimizator koji pretvara SQL u izvršne putanje pristupa, svaki optimizator radi nešto drugačije, sa različitim koracima i koristeći druge informacije.

•Ali je suština procesa ista:

–Optimizator parsuje SQL iskaz i izvršava različite faze optimizacije

•Tipično uključujući verifikaciju sintaksne i semantičke ispravnosti

–Onda se upit analizira.

–I putanje pristupa se kreiraju za upit.

**52.Kada se priča o koštanju upita, na šta se to odnosi?**

* CPU I I/O troskovi
* Statistika baze podataka
  + Broj redova u prostoru tabela, u tabeli, indexu
  + Broj jediinstvenih vrednosti smestenih u koloni
  + Najcesce koriscene vrednosti za kolone
  + Gustina indexa
    - Prosecan procenat duplikatnih vrednosti
* Detalji odnosa klasteringa za klasterovane tabele
* Korelacija kolona sa ostalim kolonama
* Strukturno stanje indexa ili prostora tabela
* Kolicina prostora koji koristi objekat baze podataka

**53.Kako sortiranje u upitu utice na performance? Kako se izbegava sortiranje?**

* Izbegavati sortiranje kad god je to moguce
* Kada su preformanse vazne, zapamtite da potrazite sortiranje I nadjete nacine dag a eliminisete.
* Mozete koristiti indexed a bi izbegli sortiranje za sledece SQL konstrukte vecine DBMS-OVA:

-ORDER BY

-GROUP BY

-DISTINCT

-UNION

**54.Kako manjak ili višak COMMIT naredbi utiče na performance?**

* COMMIT iskaz finalizira bilo koju modifikaciju u bazi podataka.
* Izmenjeni podaci su zakljucani sve dok se ne komituju.
* Brave imaju uticaj na konkurenciju I degradiraju preformanse (I verovatno dostupnost)

Kao DBA morate da osigurate da developer aplikacije izda dovoljno COMMIT-a da bi minimizirali uticaj zakljucavanja na dostupnosti I da bi održači segmente rollback segmenta na upravljivu veličinu.

**55.Kako stored procedure utiču na performance?**

* Stored procedure mogu da se koriste za smanjenje mreznog saobracaja I poboljsanja preformanse.
  + Stored procedura moze da sadrzi vise SQL iskaza
  + Samo jedan prolaz skroz mrezu je zahtevan za izvrsenje stored procedure
  + Vise prolaza kroz mrezu bi zahtevalo da bi se izvrsio svaki, individualni, SQL iskaz…

Lekcija12 –Sigurnost baze podataka

**56.Šta je kompromitovanje podataka?**

Kompromitovanje podataka je neovlašćeno objavljivanje informacija koje kompromituje bezbednost, poverljivost, integritet ličnih informacija.

**57.Šta je osnova sigurnosti podataka?**

* Svi resursi baze podataka su kontrolisani od strane DBMS-a
* Da bi korisnik mogao da obavlja bilo koju operaciju funkciju DBMS-a, jedan od sledecih funkcija mora postojati:
  + Korisnik je dobio mogucnost da obavlja tu funkciju ili operaciju, ili
  + Ta je operacija ili funkcija genericki data svim korisnicima

**58.Šta je autentikacija u DBMS-u?**

* Jaka autentifikacija je kamen temeljac svakog plana realizacije bezbednosti
* Nemoguce je kontrolisati ovlascenje I pracenje koriscenja bez nje
* Pre odobrenog ovlascenja za koriscenje resursa baze podataka, za svakog korisnika treba da se uspostavi prijava na DBMS.
  + Prijave se ponekad nazivaju nalozi, ili identifikatori korisnika
  + Prijava ce imati lozinku, tako da samo oni koji znaju lozinku mogu da koriste taj identifikator
  + Neki DBMS-ovi koriste login ID I lozinku operativnog sistema kao ID DBMS-a; drugi zahtevaju da se posebno kreira dodatni login ID I lozinka za pristup bazi podataka I bezbednost.

**59.Objasniti GRANT I REVOKE iskaze. Šta označava WITH GRANT?**

* DBA kontroliše bezbednost baze podataka i ovlašćenje koristeći Data Control Language ( DCL) . DCL iskazi sadrže dva osnovna tipa :

- GRANT dodeljuje dozvolu korisniku baze podataka.

- REVOKE ukida dozvolu korisniku baze podataka

* GRANT iskaz se izdaje kada lista privilegija treba da bude dodeljena listi korisnika • WITH GRANT opcija omogućava korisniku da preda ovlašćenje korisniku da on može da odobri privilegije drugima

– Generalno, upotreba ove klauzule zavisi od toga da li centralizovana ili decentralizovana administracija privilegija

• Decentralizovanu administraciju je generalno lakše uspostaviti, ali teže kontrolisati. Kako sve više i više korisnika dobija ovlašćenja da odobrava privilegije, obim ovlašćenja je proširen i postaje nezgrapan.

• Centralizovana administracija je generalno lakše za upravljanje, ali stavlja teret na centralizovanog administratora kao jedinog arbitra privilegija unutar okruženja.

**60. Koje vrste privilegija postoje? Objasniti.**

* + Tabela: da kontroliše ko može da pristupi i modifikuje podatke u tabelama
  + Objekat Baze: kontroliše ko može da kreira nove objekte baze podataka i uklanja postojeće objekte baze podataka
  + Sistem: kontroliše ko može da obavlja određene vrste aktivnosti u sistemu
  + Program: kontroliše ko može kreirati, modifikovati, i korisiti programe baze podataka
  + Uskladištene procedure: kontroliše ko može da izvrši određene funkcije i uskladištene procedure

**61. Objasniti privilegije tabele**

* Privilegije Tabela se odobravaju korisnicima da bi im omogućile da pristupe tabelama, pogledima, i kolonama u tabelama. Sledeći privilegije mogu da se dodele za tabele i poglede:
  + SELECT: da omogući korisniku da pročita iz ove tabele/pogleda
  + INSERT : da omogući korisniku da ubaci redove u ovu tabelu/pogled
  + UPDATE : da omogući korisniku da ažurira ovu tabelu/pogled
  + DELETE : da omogući korisniku da izbriše redove iz ove tabele/pogleda
  + ALL : da omogući korisniku da selektuje , ubaci , ažurira i briše koristeći ovu tabelu/pogled

**62. Šta omogućavaju sistemske privilegije?**

* Sistemski privilegije kontrolišu koji korisnici mogu da koriste određene funkcije DBMSa i izvršavaju određene DBMS komande
* Dostupne privilegije sistema će se razlikovati od DBMS do DBMS ali mogu uključivati

– Mogućnosti da se arhiviraju logovi baze podataka

– Isključenja i restart servera baze podataka

– Početak praćenja za monitoring

– Upravljanje smeštanjem

– Upravljanje kešom baze podataka

**63. Šta su privilegije programa I procedure? Zašto su ponekad poželjnije ove vrste privilegija?**

* + Davanje EXECUTE privilegije daje dozvolu korisniku za izvršenja programa ili uskladištene procedure.
  + Davanje privilegije korisnicima nad programima i procedurama je lakše kontrolisati nego davanje privilegija na zasebnim tabelama i kolonama.
  + Proceduralni logika u programu i postupku kontroliše koje specifične tabele i kolone se mogu modifikovati.
  + DBA može bolje održavati integritet podataka na izvršnom okruženju ako je jedini način da se promeni programski

**64. Šta su I čemu služe uloge u procesu autorizacije?**

* Jednom definisana, uloga se može koristiti da se dodeli jedna ili više prethodno dodeljenih privilegija korisniku.
* Uloga je u suštini zbirka privilegija

**65. Koje grupe su često unapred definisane u DBMSovima?**

* Autorizacija na nivou grupa je slična ulogama.
* Svaki DBMS obezbeđuje ugrađene grupe koje ne mogu da se promene. Sledeće grupe su uobičajene među glavnim DBMSovima:

– Sistem administrator. Ponekad skraćeno SA ili SYSADM, sistem administrator grupa je najmoćnija u okviru DBMSa.

– Administrator baze podataka. Ponekad skraćeno DBADM ili DBA, grupa administratora baze daje sve privilegije u određenoj bazi podataka, plus mogućnost da pristupa, ali ne modifikuje, podatke u tabelama u toj bazi podataka.

– Održavalac baza podataka. Ponekad skraćeno DBMAINT, grupa za održavanje baza podataka sadrži konkretne privilegije baze podataka za održavanje objekata baze podataka (kao što je mogućnost za pokretanje alata i izdavanje naredbi).

– Administrator bezbednosti Ova uloga, zvana SSO ili SECADM, ima skup privilegija koja dozvoljava davanje i ukidanje bezbednosti baze duž svih DBMS. Svaka aktivnost vezana za bezbednost baze podataka može se izvesti od strane administratora bezbednosti.

– Kontrola operacija. Ponekad se nazivaju OPER ili SYSOPR, uloga kontrole operacija ima ovlašćenje da vrši operativne zadatke u bazi podataka, kao što su backup i oporavak, ili ukida odbegle zadatke.

**66. Šta je SA korisnik? Koliko SA korisnika treba da bude u organizaciji?**

* Jedna organizacija treba da ograniči broj korisnika kojima je dodeljena SA uloga ili grupa

– Korisnik sa SA mogućnostima je veoma moćan

• Samo korporativnim DBA i sistem programerima treba odobriti ovaj nivo ovlašćenja

• Krajnji korisnici , menadžeri i developeri aplikacije ne trebaju SA ovlašćenja da bi radili svoj posao

• Ovo je posebno uznemirujući problem sa kupljenim aplikacijama od trećih strana

– Mnoge takve aplikacije se oslanjaju na SA ovlašćenja jer je lakše

– Međutim, nije sigurnosna praksa .

**67. Šta je kaskadni REVOKE?**

* Neki korisnici koji su imaju privilegije grupe mogu odobriti privilegije drugima

– Ako autoritet na nivou grupe ima oduzetu privilegiju onda će sve privilegije koje je korisnik odobrio biti ukinute

– To je slično kaskadnom GRANTu koje se javlja kao posledica opcije WITH GRANT

* • Pre ukidanja ovlašćenje grupe od korisnika, obavezno utvrditi uticaj kaskadnih GRANTova, i budite spremni da ponovo primenite potrebne privilegije koje će biti uklonjene zbog kaskadnog efekta.

**68. Kako korišćenje pogleda poboljšava sigurnost? Šta su vertikalna I horizontalna ograničenja?**

* Pogled može biti kreiran tako da izostavlja osetljive informacije

– Stvaranjem pogleda bez osetljivih kolona, korisnici mogu dobiti SELECT privilegiju na pogledu i moći će da pristupe informacijama o zaposlenima koje se smatraju prikladnim

* Vertikalna Ograničenja

– Kada pogled eliminiše kolone iz osnovne tabele

– Korišćenje listing kolona

* Horizontalna Ograničenja

– Kada pogled eliminiše redove iz osnovne tabele

– Korišćenje WHERE klauzule

**69. Kako se koriste stored procedure za pružanje dodatnog nivoa sigurnosti?**

* Stored procedure se mogu koristiti za pružanje dodatnog nivoa bezbednosti .
* Stored procedure mogu biti kodirane tako da pristupe samo redovima i/ili kolonama podskupa podataka
* Sposobnost da se izvrši ova stored procedura se zatim može odobriti korisnicima
* Ako nema privilegije nad osnovnim tabelama, korisnici će moći da pristupe podacima samo izvršavanjem stored procedura, čime se obezbeđuje potrebna sigurnost

**70. Šta je enkripcija? Koje dve vrste enkripcije se koriste u bazama podataka?**

* Enkripcija pretvara podatak čineći je nečitljivim svima bez ključa za dešifrovanje.
* Postoje dve vrste šifrovanja

– U mirovanju

– U tranzitu

**71. Šta je SQL injection? Objasniti.**

* SQL injekcija je oblik web hakovanja kojim su SQL iskazi navedeni u poljima online formulara da bi izazvali loše dizajniranu web aplikaciju za prebacivanje sadržaja baze podataka na napadača.
* SQL injekcija je omiljena tehnika web hakera.
* Web aplikacija korišćenjem dinamičkog SQL zahteva od korisnika da se prijavi sa svojim e-mail adresom i lozinkom.

SELECT userid,password

FROM uid\_pwd\_table

WHERE field = ’$EMAIL’

* Pametan haker može da pokuša napad SQL injekcijom ovako:

Anything’ OR ‘1’=’1

* Ako aplikacija ne proverava pravilno unos, injekcija uzrokuje da SQL sad izgleda ovako:
* SELECT userid,password
* FROM uid\_pwd\_table
* WHERE field = ’anything’ OR ‘1’=’1’;
* Drugi oblik SQL injekcije se oslanja na nepravilno kucanje, na primer ne proverava se da li podaci koje bi trebalo da budu numerički su zaista numerički.

Statement := “SELECT \* FROM userinfo WHERE id=”+in\_var+ “;”

* Haker može da unese:

4;DROP TABLE customer

**72. Kako sprečiti SQL injection?**

* Koristeći dobro dizajnirane prevodioce za upitne jezike i kodiranjem aplikacije na odgovarajući način može da se spreče napadi SQL Injekcijom.

– Kada god je moguće, koristite statički SQL.

– Uvek proverite korisnikov unos testiranjem tipa , dužine, formata, i opsega

– Program bi trebalo da nema nikakve pretpostavke o veličini, vrsti ili sadržaju podataka koji su primljeni.

– Izbegavajte nadovezivanje korisnikovog unosa koji nije proveren

– Analizirajte unos i sve što sadrži specijalne karaktere kao što su ; , ' , komentari ( - , / \* ... \* / ) , V$ ( početak Oracle DBA pogleda) , i xp\_ (početak SQL stored procedura server kataloga).

– Za korisnike Microsoft SQL Servera, čuvajte se Transact-SQL procedure xp\_cmdshell. Ova procedura prolazi Windovs komandnu ljusku i daje string za izvršavanje

Lekcija13 – Bekap podataka

**73. Šta je pun, a šta inkrementalni bekap? Koje su prednosti a koje mane jednog i drugog?**

* Pun bekap kopije slike je potpuna kopija svih podataka u objektima baze podataka u vreme kada je kopija slike pokrenuta.
* Inkrementalni bekap kopije slike (aka diferencijalni bekap) sadrži samo podatke koji su se promenili od poslednjeg punog ili inkrementalnog bekapa kopije slike.

– Prednost uzimanja inkrementalnog, a ne kompletnog bekapa kopije slike je da se ponekad može napraviti brže, i zahteva manje prostora na disku (ili traci).

– Nedostatak je da oporavak zasnovan na inkrementalnim kopijama može trajati duže, jer, u nekim slučajevima, isti red se ažurira nekoliko puta pre nego što se poslednje promene obnove.

**74. Kada koristiti inkrementalni a kada puni bekap?**

* Favorizujte upotrebu potpune kopije slike za male objekte baze podataka.

– Definicija "mali" će varirati od mesta do mesta i od DBMSa do DBMSa.

* Razmislite o korišćenju inkrementalne kopije slike za smanjenje prozora obrade batcha, za veoma velike objekte baze podataka koji su minimalno modifikovani , između bekapova kopija slika.

– DBA bi trebalo da bazira odluku pun vs inkrementalni na osnovu procenta blokova podataka koji su modifikovani, ne na osnovu broja redova koja su modifikovani

* Neki scenariji nisu kompatibilni sa inkrementalnim bekapom kopije slike.

– Neki DBMSovi dozvoljavaju korisniku da onemogući logovanje tokom nekih operacija i uslužnih programa. Kad god se preduzme akcija koja dodaje ili menja podatake bez logovanja, potrebna je puna kopija slike.

**75. Da li se radi bekap indeksa? Koji problem može da nastane ukoliko se ne uradi bekap indeksa?**

* Neki DBMSovi podržavaju pravljenje bekapa indeksa.
* – Zaista, neki DBMSovi zahtevaju indekse da se bekapuju, dok je za druge bekap indeksa opciono.

– Bekap indeksa može biti opciono jer DBMS može obnoviti indeks od podataka iz tabele

* Moraćete da ispitate kompromise kopiranja indeksa.
* Budite sigurni da obavljate bekap podataka i indeksa istovremeno, ako se odlučite da bekapujete, a ne obnavljate svoje indekse.

– Ako to ne učinite, možete dovesti do indeksa koji se ne poklapaju sa obnovljenim podacima

**76. Šta je vruć a šta hladan bekap? Koji problem mogu da nastanu kod vrućeg bekapa?**

* Hladan bekap se postiže zaustavljanjem instance baze podataka i pravljenjem bekapa fajlova relevantnih baza podataka
* Vrući backup se obavlja dok je instanca baze onlajn , što znači da je moguć istovremeni pristup
* U zavisnosti od mogućnosti DBMS a koji koristite, vruć backup može biti problematičan jer

– Oni mogu biti složeniji za implementaciju

– Oni mogu izazvati dodatnu potrošnju u vidu većeg CPU, dodatnog I/O , i dodatnim smeštanjem loga baze

– Oni mogu zahtevati da DBA da stvori skripte specifične za sajt, da bi izvršio hot backup

– Oni zahtevaju veliko testiranje kako bi se osiguralo da su bekapi održivi za oporavak

**77. Navesti šta se, osim tabela, bekapuje.**

• Obavezno napravite bekap svih bitnih komponenti instance baze podataka, uključujući –DBMS fajlove –Katalog sistema i objekte direktorija –Logovi (arhive) baze podatala –Fajlovi za konfiguraciju i podešavanje –Sistemske biblioteke –Biblioteke za upravljanje trakama –Izvršni program biblioteke i izvršne biblioteke

**78. Šta je oporavak baze podataka? Kada se oporavak smatra uspešnim?**

•Oporavak podrazumeva mnogo više od pukog vraćanje slike podataka kako je izgledala u nekom ranijem trenutku. •Oporavak baze podataka uključuje dovođenje podataka nazad u svoje stanje u (ili pre) vremena problema. Uspešan oporavak je onaj u kome ste dobili podatke aplikacije do stanja koje želite, da li je stanje kako je bilo prošle nedelje, juče, ili samo trenutak pre.

**79. Kako može da utiče migracija verzija DBMSa na oporavak?**

•Ponekad su DBMS prodavci promenili format fajlova bekapa kopije slike, tako da je stari format neupotrebljiv. –U zavisnosti od DBMSa i posebno nove verzije, bekap uzet u prethodnom releasu ne može biti upotrebljiv za oporavak posle migracije. –Alternativno, bekap uzet posle migracije koja pokušava da se koristi nakon povratka nazad u stariju verziju DBMSa takođe možda neće biti koristan za oporavak.

**80. Navesti i objasniti tipove oporavka.**

•Oporavak do trenutnog •Point-in-Time (PiT) oporavak •Oporavak transakcije

•Oporavak transakcija omogućava korisniku da oporavi određeni deo baze podataka na osnovu korisnički definisanih kriterijuma. Ovo može biti na nivou: –transakcija ili –nivoa aplikacije

**81. Koje su opcije za oporavak indeksa. Objasniti.**

Opcije: •Rekreiranje indeksa iz podataka tabele, ili •Oporavak indeksa iz kopije bekapa samog indeksa.

•Rekreiranje indeksa iz podataka tabele •Svaki DBMS identifikuje objekte baze podataka pod nejgovom kontrolom unutrašnjim identifikatorom –Kada je objekat uklonjen i ponovo kreiran, interni identifikator za taj objekat će se obično promeniti –Stoga, ponovo kreiranje objekta koristeći isti DDL i izvršavanje oporavka pomoću prethodnog bekapa kopije slike neće raditi. •Da bi se oporavio uklonjen objekat baze podataka, DBA će možda morati da prevede interni identifikator starog objekta baze u unutrašnji identifikator novog objekta baze podataka.

•Oporavak indeksa iz kopije bekapa samog indeksa •Oštećen blok ili strana je deo jednog prostora tabela ili indeks koji sadrži loše ili nekonzistentne podatke. •Da biste oporavili indeks sa oštećenom stranicom možete jednostavno obnoviti indeks iz podataka u stranici tabele. –Ponekad jednostavno zaustavljanje i pokretanje prostora tabela ili recikliranja DBMS instance može popraviti oštećenu stranicu. –Neki DBMSovi dolaze sa alatom za popravku koji se može koristiti da ukaže lokacije u datoteci na osnovu pomeraja i zamenom podatka na nivou bita ili bajta.

–Neki DBMSovi dolaze sa alatom za popravku koji se može koristiti da ukaže lokacije u datoteci na osnovu pomeraja i zamenom podatka na nivou bita ili bajta. –Kada ste popravili informacije, možda ćete morati da povratite stranice tabele do trenutnih.

**Lekcija15– Upravljanje podacima i skladištenjem**

**82. Šta je važnije, integritet ili dostupnost? Objasniti.**

•Za kritične aplikacije, integritet podataka može biti važniji od dostupnosti podataka. –Ako je medij za skladištenje nepouzdan i kvar izaziva oštećenje podataka, izgubljeni podatak može biti veći problem nego zastoj u izvršavanju. –Rešenja za skladištenje baze podataka mora da zaštiti podatke po svaku cenu

**83. Koji su ciljevi pri kreiranju sistema za smeštanje podataka.**

•Sprečavanje gubitka podataka •Obezbeđivanje dostupnosti adekvatnog kapaciteta, uz mogucnost prosirenja, kako potreba za skladistenjem raste •Izbor rešenja koje obezbeđuje brz pristup podacima sa minimalnim ili bez prekida servisa •Izbor rešenja za skladištenje koja su tolerantna na kvar i koja se mogu brzo popraviti kada se dogodi kvar ili otkaz •Izbor rešenja za čuvanje gde možete dodati ili zameniti diskove bez prekida rada •Kombinovanje svih gore navedenih u isplativo rešenje za skladištenje koje vaša kompanija može priuštiti

**84. Koji razlozi mogu biti da se DBA odluči za korišćenje više uređaja?**

Razlozi su da bi: •Uskladio zahteve performansi fajla sa odgovarajućim disk uređajem. •Razdvojio indekse od podataka zbog performansi •Izolovao log transakcije na drugom i vrlo brzom uređaju •Izolovao privremene i radne fajlove na jednom mestu •Podelili podatke na više uređaja za olakšavanje paralelnog pristupa

**85. Ako imate DBMS na Unixu, da li treba koristiti row fajlove ili sistemske fajlove. Objasniti razlog.**

**86. Navesti opcije za skladištenje podataka.**

Opcije za skladistenje: •SSD

•RAID

•JBOD

•SAN

•NAS

•Tiered Storage

**87. Objasniti prednosti I mane SSD diska.**

•SSD-ovi koriste SSD memoriju za skladištenje podataka. •Iako SSD disk ima bolje performanse nego tradicionalni, i manje su podložni fizičkim oštećenjima, oni su znatno skuplji i imaju manje kapacitete

**88. Koji nivoi RAID-a su pogodni za korišćenje u bazama podataka? Objasniti zašto.**

•Preferirajte nivoe RAIDa koji su otporni na greške, za datoteke baze podataka. •Izaberite odgovarajući sistem diska za vrstu delatnosti svakog objekta baze podataka. Na primer, možda ćete želeti da implementirate dva odvojena RAID sistema - jedan na RAID-5 za podatke koje se najviše čitaju, kao što su za analize i izveštavanja, i drugi da bude RAID-1 (ili RAID-0+1) za transakcije podataka koje se često upisuju i ažuriraju. RAID10 za performanse i toleranciju greške. •Ako imate budžet na raspolaganju, razmislite o RAID-0+1 , jer ima brzo čitanje , brzo pisanje, i toleranciju greške

**Lekcija16 – Distribucija podataka**

**89. Čemu služi LOAD I UNLOAD. Da li je alat uvek deo DBMS-a?**

Load I Unload su alati koji dolaze DBMS-om i koriste se da bi DBA mogao da pomeri podatke sa jednog mesta na drugo.

–LOAD uslužni alat se koristi za popunjavanje tabela sa novim podacima. –UNLOAD uslužni alat se koristi da se pročitaju podaci iz tabele i smeste u fajl sa podacima.

**90. Koji problem može nastati sa trigerima, prilikom LOAD-a podataka?**

•Kod većine baza podataka LOAD alati ne uzrokuju okidanje trigera. –To može da izazove probleme integriteta podataka. –DBA i programeri aplikacija će morati da razviju programe ili skripte koje oponašaju akcije trigera ukoliko se podaci redovno učitavaju i trigeri okidaju.

**91. Čemu služi EXPORT I IMPORT? Koja je razlika u odnosu na LAOD I UNLOAD?**

•EXPORT alat čita podatke iz tabele i stavlja ih u spoljni fajl. •IMPORT alat će čitati eksterni fajl kreiran od strane EXPORT alata i ubaciće podatke u tabelu.

•IMPORT i EXPORT alati obično rade sa više stvari, ne samo sa podacima, mada –Fajl EXPORTa obično sadrži šemu za tabelu zajedno sa podacima

–Ponekad EXPORT fajl sadrži više od samo jedne tabele –Neki IMPORT/EXPORT objekti pružaju i UNLOAD karakteristike da probaju, podele, i ograniče podatke koji se izvoze (i uvoze). •Ne nudi svaki DBMS alate za EXPORT i IMPORT

**92. Objasniti distribuiran zahtev?**

•Krajnji i najrobusniji oblik distribuiranog pristupa je distribuiran zahtev

•Jedna SQL naredba pristupa podacima sa više lokacija u isto vreme

–SQL naredbe mogu da pristupaju više servera baze podataka i višestruki SQL zahtevi, kako distribuirani tako i nedistribuirani, mogu da se nađu u jednoj jedinici rada

**93. Šta je distribuirana jedinica rada?**

•Svaka jedinica rada može da pristupi podacima sa više lokacija •Ovaj tip pristupa se naziva distribuirana jedinica rada. •Više od jednog DBMS se može pristupiti po jedinici rada •Višestruki SQL iskazi mogu čitati i/ili modifikovati podatke na više servera baze podataka u jednoj jedinici rada.