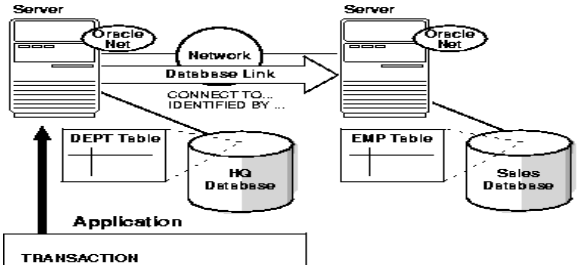
Proof of Concept

Cilj koncepta: Predstaviti arhitekturu aplikacije koja će optimalno funkcionisati u situaciji kada broj istovremenih korisnika preraste mogućnosti jednog servera.

Vreme odziva i stalna dostupnost su dva ključna elementa koje razdvajaju dobre od loših World Wide Web (Web) sajtova. Pojedina istraživanja pokazala su da sporo reagovanje Web sajtova prilikom učitavanja web stranice može da dovede do značajnih finansijskih gubitaka. To ukazuje na značaj brzog vremena odziva Web sajta za retenciju klijenata. Ovakve neželjene okolnosti, prvenstveno prouzrokovane postojanjem velikog broja korisnika i upita, mogu se ublažiti ili potpuno eliminisati primenom različitih tehnologija, metoda i tehnika koje se oslanjaju na vrhunske performanse kako bi opslužili veliki broj korisnika("traffic").

Indeksiranje baze podataka - Indeksiranjem baze podataka formira se struktura podataka koja poboljšava brzinu rada i pronalaženje podataka u tabeli baze podataka. Međutim, ovo unapređenje ima svoju „cenu" koja predstavlja dodatni upise i prostor za skladištenje kako bi se održala struktura indeksa baze podataka.

Distribuirana baza podataka - je kolekcija velikog broja, logički integrisanih baza podataka, distribuiranih preko određene heterogene računarske mreže. Sistem za upravljanje distribucionim bazama podataka je softver koji upravlja mehanizmom pristupa distribuiranim bazama podataka i čini ga transparentnim za sve korisnike. To je više ili manje spregnut multiprocesorski računarski sistem sa sinhronizovanom vremenskom podelom zadataka. Za što efikasnije iskorišćavanje softverskih, hardverskih i komunikacionih resursa prilikom projektovanja distribuirane baze podataka neophodno je izvršiti pravilnu fregmentaciju, alokaciju i replikaciju fragmenata i optimizaciju baze podataka. U nekim specifičnim slučajevima potrebno je izvršiti optimizaciju postavljanja korisničkih upita. Replikacija znači da se na određenim lokacijama čuva više kopija baze podataka, relacija, baze podataka ili relacionih fragmenata.
Za što efikasnije iskorišćavanje softverskih, hardverskih i komunikacionih resursa prilikom projektovanja distribuirane baze podataka neophodno je izvršiti pravilnu fregmentaciju, alokaciju i replikaciju fragmenata i optimizaciju baze podataka. U nekim specifičnim slučajevima potrebno je izvršiti optimizaciju postavljanja korisničkih upita.



Slika 1: Oracle distribuirana baza podataka

Klasterovanje (Computer Clusters) – Klaster predstavlja skup računara koji su povezani jedni sa drugim preko veoma brze LAN mreže gde svaki čvor(računar korišćen kao server) koristi svoju instancu operativnog sistema. Ovime je omogućeno da klasterski sistem poseduje znatno bolji odnost cene i performanse u odnosu na pojedinačni računar slične brzine. Klasterska arhitektura je prvenstveno namenjena: (1) kao podrška web servisa; (2) za primenu u računanju naučnih proračuna. Dve najčešće korišćene vrste klastera su:

- **Load-Balancing Clusters** – U ovoj arhitekturi, klaster čvorovi dele opterećenje kako bi poboljšali ukupne performanse sistema. Na primer klaster može dodeliti različite upite različitim klsaterima, kako bi vreme odgovora (response time) bilo optimizovano. Algoritam raspodele može se bitno razlikovati u zaisnosti od aplikacije. Klaster visokih performansi koji se koristi za naučne proračune će balansirati rad različitim algoritom od na primer Web-server klastera koji može koristiti jednostavan "round-robin" algoritam gde se svaki novi zahtev dodeljuje drugačijem čvoru.
- **High-Availability (Failover) Clusters** – Oni nude najveću verovatnoću da će svi resursi biti u potpunoj funkciji u svakom trenutku. Bez ove arhitekture, ako se kvar dogodi, kao što slučaj kada „sistem padne" onda su izgubljeni svi upiti koji su bili u toku. HA klasteri pokušavaju da spreče ovakvu situaciju pomoću automatskog detektovanja hardverskih ili softverskih kvarova i u istom trenutku ponovno pokreću aplikaciju na drugom sistemu (proces poznatiji kao "Failover"). Ovaj tip klastera ima široku upotrebu u file-sharing aplikacijama, biznis i e-commerce web sajtovima.

Proof of Concept

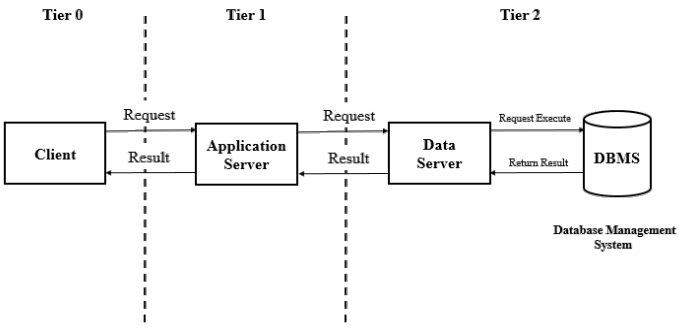
Prednosti korišćenja klaster računara su mnogobrojne, a najvažnije su:

- **Apsolutna skalabilnost** – Moguće stvoriti velike serverske klastere koji imaju veću moć (procesorsku, memorijsku) čak i od najvećih samostalnih računara. klaster može imati desetine multiprocesorskih mašina.
- **Dodatna skalabilnost** – Klaster je konfigurisan tako da je moguće da dodaju novi sistemi uz minimalnu nadogradnju. Klasteri imaju mogućnost horizontalnog proširenja, a to znači da više računara mogu da se dodaju klasteru a da pri tome klaster poboljša svoje performanse, redundantnost ili toleranciju greške.
- **Visoka dostupnost** – Svaki čvor u klasteru je samostalan kompjuter, neuspeh jednog čvora ne znači gubitak servisa. Jedan čvor može da se isključi iz postojeće arhitekture i odnese na održavanje, dok ostatali deo klastera preuzima zadatke tog pojedinačnog čvora.
- **Prihvatljiv odnos cena/performanse** – Klasteri su obično podešen da poboljšaju performanse i dostupnost u odnosu na pojedinačni računar, a obično je mnogo isplativiji od pojedinačnih računara u odnosu na istu brzinu i dostupnost.

Connection pooling - Otvaranje i održavanje konekcije pojedinačnog internet korisnika i generisanje dinamičke database-driven Web sajt aplikacije korišćenjem baze podataka, predstavlja skupu aktivnost i nepotrebno trošenje celokupnih resursa sistema. U cilju smanjivanja trošenja ovih resursa, stvara se keš konekcija baze podataka. Nakon što konekcija uspostavljena, ona se smešta u "bazen" (Connection pool), gde budući zahtevi umesto kreiranja nove konekcije, koriste onu koje je smeštena u bazen. Ukoliko postojeća konekcija nije dostupna, uspostavlja se nova i dodaje u bazen. Na ovaj način se smanjuje vreme koje korisnik mora da čeka kako bi uspostavio konekciju prema bazi podataka.

Tehnike keširanja - (caching ili data caching) su razvijene kao odgovor na zahteve za povećanje performansi web aplikacija koje pristupaju bekend bazama podataka. Keširanje obično postiže određene uspehe u poboljšanje skalabilnosti na aplikativnim i web serverima koji rade na više relativno jeftinijih računarskih sistema. Međutim, ovime se ne rešava problem skalabilnosti za servere baze podataka. Jedan od mogućih rešenja ovog problema je primena srednjeg (middle-tier) sloja baze podataka, koji se najčešće primenjuje na web i/ili serveru aplikacija, kao multi-tier (uglavnom three-tier) arhitektura.

Multi-tier web sajt arhitektura sa keširanjem srednjeg nivoa baza podataka, omogućava sledeća poboljšanja sistema: (a) *Horizontalna skalabilnost*: deljenje opterećenja upita jedinstvenom serveru baze podataka na više jeftinih računarskih sistema (klastera) pomoću Web/Aplikativnih servera i multi-tier keširanja baze. (b) *Performanse*: ujednačavanje vrhova("peak") opterećenja koje su posledica znatno većeg opterećenja celokupne infrastrukture sistema od uobičajenog. (c) *Dostupnost*: nastavak servisa za aplikacije koje zavise samo od keširanih tabela čak i ako je server baze podataka nedostupan.



Slika 2: Three tier arhitektura

Client-Side Web Caching – Ovakva vrsta keša se nalazi izvan serverske mreže, uglavnom u pretraživačima i u bilo kojoj web sredini između korisnika i servera(na primer na strani ISP provajdera). Glavni cilj web keširanja je da smanji server lag. Web keš je podeljen na „kratkoročni keš" koji prima web objekte sa Interneta direktno, i dugoročne memorije koja prima web objekte iz kratkoročnog keša. Kada se korisnik usmerava ka određenoj web stranici, svi web objekti ugrađeni u stranice, pre svega, se čuvaju u kratkoročnom kešu. Web objekti koji su posećeni više puta biće izmešteni u dugoročnu memoriju za duže keširanje, ali sa druge strane drugi najstariji objekati će biti uklonjeni iz dugoročnog keša. ClientSide Web Caching obezbeđuje da su poželjni web objekti sačuvani što duže vreme, dok su manje korišćeni objekti ranije uklonjeni.