**Osztott Rendszerek vizsgakérdések:**

1. **ACID tulajdonságok**

* Atomosság (Atomicity): vagy minden műveletet végrehajtunk, vagy egyiket sem. Ha a tranzakció nem sikerül, az objektum állapota sértetlen marad.
* Konzisztencia (Consostency): A tranzakció érvényes állapotátmenetet hoz létre. Ez nem zárja ki az érvénytelen köztes pillanatokat a tranzakció végrehajtása során.
* Elkülönítés (Isolation): Az egyidejű tranzakciók nem zavarják egymást. Minden tranzakciónak látszólag úgy kell lefutnia, mintha ez alatt az idő alatt semmilyen másik tranzakciót sem hajtanánk végre.
* Tartósság (Durability): Egy tranzakció végrehajtása után, annak hatásai állandósulnak: ha egyszer egy tranzakció befejeződött, akkor már soha többé nem veszhet el a tranzakció hatása.

1. **Milyen jellemzői vannak az átható,osztott rétegeknek?**

Átható rendszer Sok modern elosztott rendszer kicsi, mobil elemekbol áll. ˝ Néhány jellemzo˝ A környezet megváltozhat: A rendszernek ezt követnie kell. Ad hoc szervezodés: ˝ A rendszer komponenseit nagyon különbözo˝ módokon használhatják a felhasználók. Ezért a rendszernek könnyen konfigurálhatónak kell lennie. Megosztott szolgáltatások: Mivel a rendszer nagyon változékony, az adatoknak könnyen kell áramlaniuk. Ennek elosegítésére a rendszer ˝ elemei általában nagyon egyszer ˝u szerkezet ˝uek.

1. **Hasonlíts össze két kölcsönös kizárási algoritmust**

*Kölcsönös kizárás: decentralizált* Tegyük fel, hogy az eroforrás ˝ n-szeresen többszörözött, és minden replikátumhoz tartozik egy azt kezelo koordinátor. ˝ Az eroforráshoz való hozzáférésr ˝ ol ˝ többségi szavazás dönt: legalább m koordinátor engedélye szükséges, ahol m > n/2. Feltesszük, hogy egy esetleges összeomlás után a koordinátor hamar felépül – azonban a kiadott engedélyeket elfelejti.

*Kölcsönös kizárás: elosztott :* M ˝uködési elv Ismét többszörözött az eroforrás, amikor a kliens hozzá szeretne férni, ˝ kérést küld mindegyik koordinátornak (idobélyeggel). ˝ Választ (hozzáférési engedélyt) akkor kap, ha a koordinátor nem igényli az eroforást, vagy ˝ a koordinátor is igényli az eroforrást, de kisebb az id ˝ obélyege. ˝ Különben a koordinátor (átmenetileg) nem válaszol.

1. **Miben hasonlítanak és térnek el a szálak és a folyamatok**

**Szál** (thread): A processzor egyfajta szoftveres megfeleloje, minimális ˝ kontextussal (környezettel). Ha a szálat megállítjuk, a kontextus elmentheto és továbbfuttatáshoz visszatölthet ˝ o.˝

**Folyamat** (process, task): Egy vagy több szálat összefogó nagyobb egység. Egy folyamat szálai közös memóriaterületen (címtartományon) dolgoznak, azonban különbözo folyamatok nem látják egymás ˝ memóriaterületét.

1. **Milyen lehetséges feladatai szoktak lenni a köztesrétegnek ?**

A köztesrétegbe (middleware) olyan szolgáltatásokat és protokollokat szokás sorolni, amelyek sokfajta alkalmazáshoz lehetnek hasznosak. Sokfajta kommunikációs protokoll Sorosítás ((de)serialization, (un)marshalling), adatok reprezentációjának átalakítása (elküldésre vagy elmentésre) Elnevezési protokollok az eroforrások megosztásának ˝ megkönnyítésére Biztonsági protokollok a kommunikáció biztonságossá tételére Skálázási mechanizmusok adatok replikációjára és gyorsítótárazására

1. **Hogyan keresünk rá egy egyedre,ha hierarchikus névteret használunk ?**

névtér: gyökeres, irányított, élcímkézett gráf, a levelek tartalmazzák a megnevezett egyedeket, a belso csúcsokat ˝ katalógusnak vagy könyvtárnak (directory) nevezzük Az egyedhez vezeto út címkéit összeolvasva kapjuk az egyed egy nevét. A ˝ bejárt út, ha a gyökérbol indul, ˝ abszolút útvonalnév, ha máshonnan, relatív útvonalnév. Mivel egy egyedhez több út is vezethet, több neve is lehet.

1. **Hogyan lehet fenntartani a folytonos konzisztenciát?**

Conit: Ha megtehetjük, a konzisztenciafeltéleket nem a teljes adatbázisra írjuk fel, hanem az adatoknak minél sz ˝ukebb körére. Az olyan adategység, amelyre közös feltételrendszer vonatkozik, a conit (consistency unit).

1. **Mit, milyen jellegű részeket, tartalmaznak az objektumszerverek?**

A rendszer részei a kiszolgálók, a vázak és az adapterek.

1. **RPC kommunikáció során milyen problémák léphetnek fel ?**

1. **Mi a hibajelenség, mi a hiba,hiba ok ?**

Hibajelenség (failure): a komponens nem a tole elvártaknak megfelel ˝ oen ˝ üzemel

Hiba (error): olyan rendszerállapot, amely hibajelenséghez vezethet

Hibaok (fault): a hiba (feltételezett) oka

1. **Hogyan működnek a cluster alapú fájlrendszerek ?**

Nagy fájlrendszerek esetén a kliens-szerver alapú megközelítés nem elég jó ⇒ a fájlokat csíkokra bontjuk (striping), így a részeiket párhuzamosan érjük el. Ennek célja a rendszer gyorsítása és biztonságosabbá tétele.

1. Az önszervező szerver működése
2. **Logikai gyűrű szervezésében, hogy váltunk vezetőt**

Ismét logikai gy ˝ur ˝unk van, és a folyamatoknak vannak sorszámai. A legnagyobb sorszámú folyamatot szeretnénk vezetonek választani. ˝ Bármelyik folyamat kezdeményezhet vezetoválasztást: elindít egy ˝ üzenetet a gy ˝ur ˝un körbe, amelyre mindenki ráírja a sorszámát. Ha egy folyamat összeomlott, az kimarad az üzenetküldés menetébol. ˝ Amikor az üzenet visszajut a kezdeményezohöz, minden aktív folyamat ˝ sorszáma szerepel rajta. Ezek közül a legnagyobb lesz a vezeto; ezt ˝ egy másik üzenet körbeküldése tudatja mindenkivel.

1. **A webszolgáltatásokat, hogy segíti a webproxy**
2. **Hogy épül fel a Recovery Line ?**
3. **Jellemezz egy elosztott tároló rendszert**