(c) Vaikka P_1 ei voi sitouttaa alitransaktiotansa, sekä P_2 että P_3 ovat valmiita sitouttamaan alitransaktionsa. Kun koordinaattori C on tehnyt päätöksensä, minkä viestin se lähettää ja kenelle? Onko mikään kolmesta osallistujasta epävarmuuden tilassa? *Even though P_1 cannot commit its subtransaction, both P_2 and P_3 are ready to commit their subtransactions. When C has made its decision, what message does it send and to whom? Are any of the three participants in a state of uncertainty?*

(d) Eräs osallistuja romahtaa ja elpyessään löytää lokistaan valmiuskirjauksen. Mitä se tekee seuraavaksi? Onko täma tilanne mielestäsi 2PC protokollan heikoin lenkki? Perustele vastauksesi lyhyesti. *A certain participant crashes and upon recovery, finds a prepare log entry in its log. What does it do next? Would you consider this situation to be the weakest spot in the 2PC protocol? Justify you answer briefly.*

4. (a) Merkitse seuraavat väittämät joko todeksi (T) tai epätodeksi (E). Jos väittämä on epätosi, perustele lyhyesti miksi se on epätosi. Kohta (b) liittyy sovittuun päätösvaltakäytäntöön, jossa p on lukuvallan koko.
Label the following statements as True (T) or False (F). If a statement is false, briefly state why it is so. Item(b) refers to the Quorum Consensus protocol where p is the size of the Read Quorum.

(i) Innokkaassa missä-tahansa-päivittävässä-toisintamisessa transaktio voi yleensä sitoutua nopeammin kuin Laiskassa missä-tahansa-päivittävässä-toisintamisessa. *In Eager update-anywhere-replication, the transaction can generally commit faster than with Lazy update-anywhere replication.*

(ii) Eräs ROWA protokollan eduista on, että päivitys toisinteissa voi jatkua vaikka yksi pisteistä olisi kaatunutkin. *An advantage of the ROWA protocol is that updating at the replicas can proceed even if one site is unavailable.*

(iii) Kunhan vinoumaa ei esiinny, kiertovuoro-ositus on tehokas alla olevalle kyselylle:
`Select * from ORDERS Where OrderID=557`
As long as there is no skew, round-robin partitioning is effective for the query above.

(iv) Alla olevat kaksi riviä ovat oikeellinen esimerkki pystysuoraosittaa relaatio STAFF tehtävästä 2.
`STAFF1 (StaffId, FirstName, LastName, ManagerId)`
`STAFF2 (StaffId, FirstName, LastName, DateJoined)`
The above two lines are a valid example of a vertical partition for relation STAFF from problem nr 2.

(v) Pääkopiotoisintamisessa voi esiintyä pitkiäkin viiveitä jos asiakassovellus sijaitsee kaukana pääkopiosta. *In Primary Copy replication, there may be even long delays if the client application is located far away from the primary copy.*

(b) Olkoon $n = 4$ ja $p = 2$. Luettele kaikki eri mahdolliset lukuvallat. Todetaan, että piste S_2 on muita alttiimpi romahtamaan. Mikä on todennäköisyys sille, että piste S_2 esiintyy jossakin mahdollisessa lukuvallassa? Jotta kirjoittaminen olisi mahdollisimman helppoa, mitä q:n tulisi olla? Jos nyt piste S_2 kaatuisikin, voisiko päätösvalta silti toimia oikein? Miksi tai miksi ei?
Let $n = 4$ and $p = 2$. List all possible Read Quorums. Given that site S_2 is more likely to crash, what is the probability that S_2 will appear in some possible Read Quorum? For writing to be as easy as possible, what should q then be? If now site S_2 should actually fail, could the Quorum Consensus still function correctly? Why or why not?

5. Kohdat (a)-(b) liittyvät asiakas-palvelin järjestelmään, joka perustuu sivupalvelimeen. *Items (a)-(b) relate to a Client-Server system based on a page server.*

(a) Mitä tarkoitetaan sivun p nykyversiolla? *What is meant by the current version of a page p?* Missä mielessä laiska/tarvehakuinen päivitysten levityskäytäntö on samankaltainen kuin asiakkaan Älä-Pakota käytäntö? *How is lazy update propagation/demand-driven update propagation similar to the client's No-Force policy?*

(b) Kuvaile tilanne, jossa romahduksen jälkeisessä elvytyksessä, sivu, joka oli asiakkaalla likainen, ei olekaan likainen palvelimen puolella. Mikä ongelma tästä voi seurata sivun päivityksille?
Describe a scenario where at recovey after a crash, a page that was dirty at the client is not considered dirty at the server side. What problem can this cause for the page updates?

(c) Miten toimii kaikkein hitain sivujen levityskäytäntö yhteislevyjärjestelmässä? Mikä on sen etu elvytyksessä verrattuna kaikkein nopeimpaan levityskäytäntöön? *How does the slowest page propagation work in the shared disk environment? What is its advantage at recovery over the fastest scheme?*