SQL-OHJELMOIJAN MUISTILISTA (DB2 V6R1)

A. Yllättävät saantipolut

1. Käytä muuttujalle ja literaalille sarakkeen tietomuotoa ja pituutta.

2. Vältä skalaarifunktioita WHERE-ehdossa niille hakemistosarakkeille, joita halutaan käyttää haussa. Esim. SUBSTR voidaan usein korvata BETWEEN:llä.

3. Vältä aritmeettisten lausekkeiden käyttöä WHERE-ehdossa. Laske arvo valmiiksi muuttujaan.

4. Käytä ehtoparia >=, <=, kun arvovälikyselyssä vertaat muuttujaa kahteen sarakkeeseen. Esim: Verrattaessa saraketta kahteen muuttujaan, on BETWEEN yleensä yhtä tehokas kuin ehtopari >=,<=. Esim: WHERE COL BETWEEN :m1 AND :m2.

5. Vältä negaatiota WHERE-ehdossa niille hakemistosarakkeille, joita halutaan käyttää haussa.

6. Muista, että alikyselyn kirjoitustapa määrää suoritustavan ja -järjestyksen.

7. Käytä OPTIMIZE FOR n ROWS, kun ohjelma ei lue koko tulosjoukkoa. Jos SELECT-lauseen tuloksena syntyy suuri tulosjoukko, mutta ohjelma tarvitsee siitä vain pienen osan, SELECT-lauseeseen voidaan lisätä: OPTIMIZE FOR n ROWS. Lisäyksen jälkeen optimoija saattaa valita ohjelman kannalta tehokkaamman saantipolun.

8. Muista, että OR-rakenteesta, jota ei voi konvertoida IN-listaksi (esim.WHERE x < ...OR x > ...), seuraa parhaimmillaankin MIA-saantipolku.

9. Jos DB2 yhden rivin haussa INDEX ONLY -hakupolun sijaan lukee tiedot taulusta, lisää SELECT-listaan, jos mahdollista, sarakefunktio.

10. Tarkista MIN- ja MAX -funktioiden hakupolku EXPLAIN:sta.

11. Jos NOT NULL -saraketta verrataan alikyselyn tulokseen, joka voi saada NULL -arvon, käytä alikyselyssä COALESCE (VALUE) -funktiota.

12. Älä tutki isäntämuuttujien arvoja WHERE –predikaatilla.

B. Turhat lajittelut

1. Käytä ORDER BY -listassa hakemiston järjestystä, jos mahdollista.

2. Muista, että OPEN CURSOR –lauseen suoritusvaiheessa kaikki hakuehdon täyttävät rivit lajitellaan tilapäistaulukkoon, mikäli tarvittavaa järjestystä ei saada hakemiston avulla.

3. Älä käytä DISTINCT-määrettä, jos se ei ole tarpeellista.

4. Määrittele yhdiste UNION ALL paitsi, jos duplikaattien poisto on välttämätöntä.

5. Huomioi, että ylimääräisen, hakemistosta puuttuvan sarakkeen lisääminen lajittelutekijäksi aiheuttaa lajittelun, vaikka sarakkeeseen olisi voimassa yhtäsuuruusehto.

6. Jos yhdistelyssä (JOIN) lajittelutekijät kohdistuvat useampaan kuin yhteen tauluun tai pelkästään sisempään tauluun, DB2 voi tehdä turhan lajittelun.

C. CPU-ajan säästö

1. Massalisäyksissä LOAD-apuohjelma on tehokkaampi kuin INSERT-lause. Huomaa, ettei taulut / kanta ole käytettävissä LOADin aikana (24 h käytettävyys).

2. Jokerimerkki ('%', _') ensimmäisenä merkkinä LIKE-ehdossa sulkee pois MIS-saantipolun. Sen käyttö on silti tehokkaampaa kuin rivien luku FETCH:llä ja ehdon tarkistaminen ohjelmassa.

3. Määrittele kohdistin eräajoissa WITH HOLD -optiolla.

4. Käytä olemassaolon tarkistukseen **KOHDISTIN**ta, jos tulosjoukko voi olla suurempi kuin yksi rivi.

5. Luettele SELECT –lauseessa vain ne sarakkeet, joiden tietoa tarvitset. WHERE –ehtojen sarakkeita ei pidä turhaan toistaa.

6. Vältä UPDATE –lauseessa tietoarvoltaan muuttumattomien sarakkeiden luetteleminen. Ole tarkka varsinkin avain-, viiteavain- ja muiden hakemistosarakkeiden kohdalla.

7. Vältä turhien SQL-lauseiden suorittamista.

D. Turvallisuus

1. Käytä rivien päivityksessä ja poistossa kohdistinta. Poikkeus: Käsitellään yhtä riviä, jonka perusavain tunnetaan.

2. Jos käytät näkymän määrittelyssä WHERE-ehtoja ja näkymää käytetään rivien päivitykseen/lisäykseen, määrittele näkymä WITH CHECK OPTION.

3. Käytä aina **:** -merkkiä isäntämuuttujan edessä. DB2 V6:ssa se on pakollista ja vanhat merkinsäästöt on korjattava sitä ennen.

E. Selkeys ja ylläpidettävyys

1. Älä käytä SELECT * -lausetta.

2. Kirjoita SELECT-lauseeseen vain ne sarakkeet, joita todella tarvitset.

3. Luettele INSERT-lauseessa ne näkymän sarakkeet, joihin viet tietoa.

4. Määrittele FOR UPDATE OF -sarakelistaan vain ne sarakkeet, joita aiot päivittää. Jos rivejä poistetaan, luettele perusavaimen sarakkeet.

5. Käytä mieluummin IN-listaa usean OR-ehdon sijaan.

6. Käytä isäntämuuttujina DCLGEN-struktuureja.

7. Luettele ORDER BY -sarakelistassa sarakkeet nimeltä.

8. Sarakkeen nimen on hyvä olla kuvaava. Näe samalla kertaa vaivaa ja kommentoi sekä anna otsakenimi (LABEL) kullekin sarakkeelle. Käytä samasta tiedosta aina samaa nimeä, jolloin DB2-katalogi toimii ikäänkuin köyhän miehen tietohakemistona.

F. Lukitukset

1. Auki olevan **LUKEVAN** kursorin kohteena olevien taulujen päivittäminen kursorin ulkopuolelta samassa yhteydessä (UOW) voi häiritä kursorin etenemistä. Rivi on päivitystä varten luettava SELECT FOR UPDATE –optiolla. Ohjeen noudattamatta jättämisellä voit saada aikaan ns. kengurukursorin. Tällöin päivitykset 'siirtävät' rivin paikkaa. Oireina ovat mm. hypyt rivien yli, saman rivin uudelleen luku, pahimmillaan ikikieriö (LOOP).

2. Samanaikaisesti tosiaikaohjelmien kanssa suoritettavissa eräohjelmissa voi deadlock- ja timeout-paluukoodin jälkeen (-911) harkita kesken jääneen LUW:n suoritusta uudelleen muutamia kertoja. Toistuvasti lukkiumaan johtavan tapahtuman voi ehkä ohittaa ja siirtää uudelleensuoritettavaksi.

3. Käytä kohdistimen määrittelyssä FOR FETCH ONLY -määrettä, jos rivejä aiotaan vain lukea.

4. Harkitse lukevissa lauseissa UR -optiota. Tämä parantaa samanaikaisuutta kyselijöitten ja päivittäjien välillä, mutta voi tuottaa lukevalle ohjelmalle loogisesti epäyhtenäisen tuloksen.

5. Tee commit eräohjelmassa 2-5 sekunnin välein, jolleivat painavat syyt muuta vaadi.

G. Yllättävät tulokset

1. Käytä ORDER BY -määrettä aina silloin, ja vain silloin, kun tulosjoukko halutaan tietyssä järjestyksessä.

2. Varo väärää päätelmää sarake -funktioiden luvusta.

3. Jos sarakkeen arvo on NULL, pätee siihen vain IS NULL -ehto, eikä mikään muu ehto. Kaikista muista kyselyistä rivi jää huomioimatta.

4. Ole varovainen WHERE –ehtojen kanssa ulkoliitoksen yhteydessä

H. Client-Server

1. OPTIMIZE FOR N ROWS

2. Käytä hajautetussa ympäristössä kohdistimen määrityslauseessa FOR FETCH ONLY -määrettä, jos rivejä aiotaan vain lukea.

I. Data Sharing

1. Commit-väli & LOCK AVOIDANCE –optio.

2. Ohjelman suunnitttelu mukaan; hyvä ei riitä, pitää olla loistava.

3. Käytä ehdottomasti LOCK AVOIDANCE –optiota, vaikutus tuntuu paljon laajemmin käydessään kaikkien koneiden läpi.