IBM® DB2 Universal Database™

Visual Explain - Veiledning

Versjon 8

IBM® DB2 Universal Database™

Visual Explain - Veiledning

Versjon 8

Før du bruker opplysningene i denne boken og produktet det blir henvist til, må du lese *Merknader*.

Du kan bestille IBM-publikasjoner elektronisk eller via IBM-representanten.

- Hvis du vil bestille publikasjoner elektronisk, går du til IBM Publications Center på www.ibm.com/shop/publications/order
- IBM-representanten finner du ved å gå til IBM Directory of Worldwide Contacts på www.ibm.com/planetwide

Hvis du vil bestille DB2-publikasjoner fra DB2 Marketing and Sales i USA eller Canada, må du ringe 1-800-IBM-4YOU (426-4968).

Når du sender informasjon til IBM, gir du IBM en ikke-eksklusiv rett til å bruke eller distribuere informasjonen på den måten IBM mener er best, uten forpliktelser i noen retning.

Innhold

Om denne veiledningen

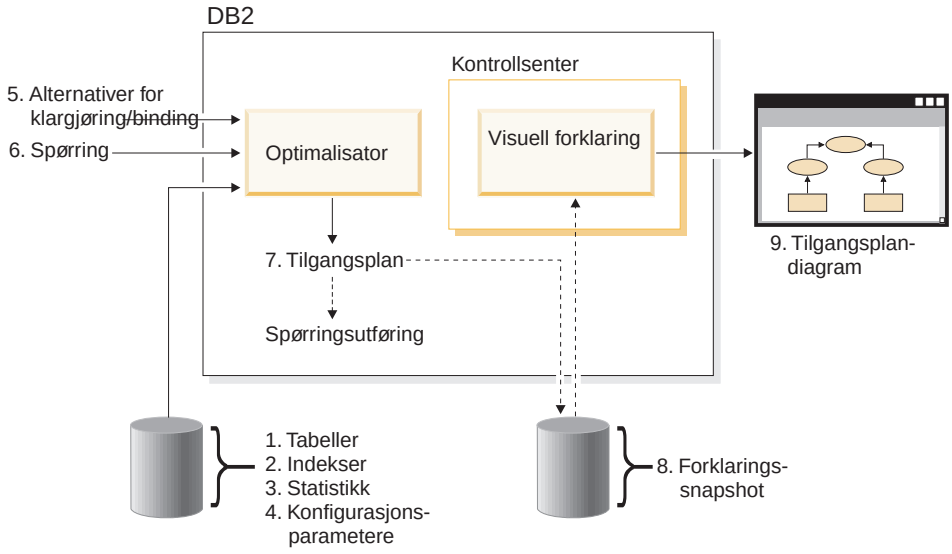
Denne veiledningen inneholder en innføring i funksjonene i DB2 Visuell forklaring. Ved å fullføre leksjonene i denne veiledningen, vil du lære hvordan du kan bruke Visuell forklaring til å vise tilgangsplanen for forklarte SQL-setninger som et diagram. Du vil også lære å bruke informasjonen som er tilgjengelig fra et slikt diagram, til å justere SQL-spørringene slik at de gir bedre ytelse.

Ved hjelp av optimalisatoren undersøker DB2 SQL-spørringene og finner ut hvordan du får tilgang til dataene på best mulig måte. Denne banen til dataene kalles tilgangsplanen. Ved hjelp av DB2 kan du se hva optimalisatoren har gjort, ved at du får se på tilgangsplanen som er valgt til å utføre en bestemt SQL-spørring. Du kan bruke Visuell forklaring til å vise tilgangsplanen som et diagram. Diagrammet er en visuell presentasjon av databaseobjektene som er involvert i en spørring (for eksempel tabeller og indekser.) Det omfatter også operasjonene som er utført for disse objektene (for eksempel skanninger og sorteringer), og det viser dataflyten.

Du kan forbedre en spørrings tilgang til data ved å utføre ett av eller alle de følgende justeringsaktivitetene:
1. Justere tabellutformingen og omorganisere tabelldataene.
2. Opprette riktige indekser.
3. Bruke **runstats**-kommandoen til å gi optimalisatoren gjeldende statistikk.
4. Velge riktige konfigurasjonsparametere.
5. Velge riktige bindingsalternativer.
6. Utforme spørringer som bare henter nødvendige data.
7. Arbeide med en tilgangsplan.
8. Opprette forklarings-snapshot.
9. Bruke et tilgangsplandiagram til å forbedre en tilgangsplan.

Disse ytelsesrelaterte aktivitetene samsvarer med som de blir vist i den følgende illustrasjonen. (Brutte linjer viser trinn som er nødvendige for Visuell

forklaring.)



Denne veiledningen inneholder leksjoner om det følgende:

- Opprette forklarings-snapshot. Dette er kravene for å vise tilgangsplandiagrammer.
- Vise og manipulere et tilgangsplandiagram.
- Utføre justeringsaktiviteter og utforske hvordan disse forbedrer tilgangsplanen.

Merk: Ytelsesjusteringen er delt i en leksjon for databasemiljøer med enkeltpartisjon og en leksjon for partisjonerte databasemiljøer.

Du kommer til å bruke eksempeldatabasen for DB2 når du arbeider med leksjonene. Se *Administration Guide* hvis du ikke allerede har opprettet eksempeldatabasen.

Environment-specific information

Information marked with this icon pertains only to enkeltpartisjonsdatabase environments.

Information marked with this icon pertains only to partisjonert database environments.

Leksjon 1. Opprette forklarings-snapshot

I denne leksjonen skal du opprette forklarings-snapshot. Funksjonen for SQL-forklaring blir brukt til å registrere informasjon om systemet der en statisk eller dynamisk SQL-setning blir kompilert. Ved hjelp av den registrerte informasjonen vil du forstå strukturen og den potensielle utføringsytelsen for SQL-setningene. Et forklarings-snapshot er komprimert informasjon som blir samlet inn når en SQL-setning blir forklart. Det lagres som et stort binærobjekt (BLOB) i tabellen EXPLAIN_STATEMENT, og inneholder følgende informasjon:

- Internfremstillingen av tilgangsplanen, i tillegg til operatorene, tabellene og indeksene som er brukt.
- Beslutningskriterier som optimalisatoren har brukt, samt statistikk for databaseobjekter og kumulative kostnader for hver operasjon.

Hvis du vil vise et tilgangsplandiagram, trenger Visuell forklaring informasjonen som ligger i et forklarings-snapshot.

Opprette forklaringstabellene

Når du skal opprette forklarings-snapshot, må du passe på at forklaringstabellene nedenfor finnes for bruker-IDen din:

- EXPLAIN_INSTANCE
- EXPLAIN_STATEMENT

Bruk kommandoen **DB2 list tables** når du skal kontrollere om de finnes. Hvis disse tabellene ikke finnes, må du opprette dem ved å bruke de følgende instruksjonene:

1. Hvis DB2 ikke allerede er startet, gir du kommandoen **db2start**.
2. Fra DB2-klarmeldingen i kommandolinjebehandleren knytter du deg til databasen du vil bruke. For denne veiledningen knytter du deg til SAMPLE-databasen ved hjelp av kommandoen **connect to sample**.
3. Opprett forklaringstabellene ved hjelp av eksempelkommandofilen EXPLAIN.DDL. Denne filen ligger i katalogen sqllib\misc. Når du skal kjøre kommandofilen, går du til denne katalogen og gir kommandoen **db2 -tf EXPLAIN.DDL**. Denne kommandofilen lager forklaringstabeller som har den tilknyttede bruker-IDen som prefiks. Denne bruker-IDen må ha CREATETAB-rettighet i databasen eller SYSADM- eller DBADM-autorisasjon.

Bruke forklarings-snapshot

Det finnes fire eksempel-snapshot som kan hjelpe deg å lære om Visuell forklaring. De følgende delene inneholder informasjon om hvordan du oppretter egne snapshot, men du trenger ikke å opprette egne snapshot for å arbeide med denne veiledningen.

- Opprette forklarings-snapshot for dynamiske SQL-setninger
- Opprette forklarings-snapshot for statiske SQL-setninger

Spørringen som ble brukt til eksempel-snapshotene, viser navnet, avdelingen og inntekten for alle ansatte som ikke er ledere, og som tjener mer enn 90 % av lønnen til den best betalte lederen.

```
SELECT S.ID,S.NAVN,O.AVDELING,LØNN+PROV
FROM ORG O, STAB S
WHERE
  O.AVDNR = S.AVD AND
  S.JOBB <> 'Leder' AND
  S.LØNN+S.PROV > ALL( SELECT ST.LØNN*.9
                       FROM STAB ST
                       WHERE ST.JOBB='Leder' )
ORDER BY S.NAVN
```

Spørringen har to deler:

1. Delspørringen (i parentes) produserer datarader som består av 90 % av lønnen til hver enkelt leder. Siden delspørringen er kvalifisert med ALL, blir bare den største verdien fra denne tabellen hentet.
2. Hovedspørringen kombinerer alle radene i ORG- og STAB-tabellene der avdelingsnumrene er de samme, JOBB ikke er lik 'Leder', og lønn pluss kommisjon er større enn verdien som ble returnert fra delspørringen.

Hovedspørringen inneholder disse tre predikatene (sammenlikninger):

```
1.   O.AVDNR = S.AVD
2.   S.JOBB <> 'Leder'
3.   S.LØNN+S.PROV > ALL ( SELECT ST.LØNN*.9
                       FROM STAB ST
                       WHERE ST.JOBB='Leder' )
```

Disse predikatene viser til:

1. Et kombinasjonspredikat som kombinerer ORG- og STAB-tabellene der avdelingsnumrene er like
2. Et lokalt predikat på JOBB-kolonnen til STAB-tabellen
3. Et lokalt predikat på LØNN- og PROV-kolonnene til STAB-tabellen som bruker resultatet av delspørringen

Slik laster du inne eksempel-snapshotene:

1. Hvis DB2 ikke allerede er startet, gir du kommandoen **db2start**.

2. Kontroller at forklaringstabellene finnes i databasen din. Følg instruksjonene under Opprette forklaringstabeller når du skal gjøre dette.
3. Koble til databasen du vil bruke. I denne veiledningen skal du koble til SAMPLE-databasen. Når du skal koble deg til SAMPLE-databasen fra CLP-klarmeldingen for DB2, gir du kommandoen **connect to sample**.

   Hvis den ikke allerede er opprettet, kan du se i avsnittet om installering av SAMPLE-databasen i *Administration Guide*.
4. Når du skal importere de forhåndsdefinerte snapshotene, kjører du DB2-kommandofilen VESAMPL.DDL.

   -  Denne filen ligger i katalogen sqllib\samples\ve.
   -  Denne filen ligger i katalogen sqllib\samples\ve\inter.

   Når du skal kjøre kommandofilen, går du til denne katalogen og gir kommandoen **db2 -tf vesampl.ddl**.
   - Denne kommandofilen må kjøres fra den samme bruker-IDen som ble brukt til å lage forklaringstabellene.
   - Denne kommandofilen importerer de forhåndsdefinerte snapshotene. Den oppretter ikke tabeller eller data. Justeringsaktivitetene som beskrives senere (for eksempel CREATE INDEX og runstats), vil bli kjørt for tabeller og data i SAMPLE-databasen.

Du er nå klar til å vise og bruke tilgangsplandiagrammene.

Opprette forklarings-snapshot for dynamiske SQL-setninger

Merk: Opplysningene om hvordan du oppretter forklarings-snapshot i denne delen, er gitt bare for informasjon. Siden det følger med eksempler på forklarings-snapshot, trenger du ikke å utføre denne oppgaven når du skal arbeide med veiledningen.

Følg disse trinnene når du skal opprette et forklarings-snapshot for en dynamisk SQL-setning:

1. Hvis DB2 ikke allerede er startet, gir du kommandoen **db2start**.
2. Kontroller at forklaringstabellene finnes i databasen din. Følg instruksjonene under Opprette forklaringstabeller når du skal gjøre dette.
3. Fra DB2-klarmeldingen i kommandolinjebehandleren knytter du deg til databasen du vil bruke. Når du for eksempel skal koble deg til SAMPLE-databasen, gir du kommandoen **connect to sample**.

   Hvis du vil opprette SAMPLE-databasen, kan du se i avsnittet om installering av SAMPLE-databasen i *Administration Guide*.

4. Opprett et forklarings-snapshot for en dynamisk SQL-setning ved å bruke en av disse kommandoene fra DB2-klarmeldingen i kommandolinjebehandleren (CLP).
   - Hvis du vil opprette et forklarings-snapshot uten å utføre SQL-setningen, kan du gi kommandoen **set current explain snapshot=explain**.
   - Hvis du vil opprette et forklarings-snapshot og utføre SQL-setningen, gir du kommandoen **set current explain snapshot=yes**.

   Denne kommandoen definerer spesialregisteret for Explain. Når det er definert, har det innvirkning på alle etterfølgende SQL-setninger. Hvis du vil ha flere opplysninger, kan du >se i delene som omhandler gjeldende forklarings-snapshot i *SQL Reference*.
5. Gi SQL-setningene fra DB2-klarmeldingen i kommandolinjebehandler (CLP).
6. Hvis du vil se tilgangsplandiagrammet for snapshotet, fornyer du vinduet Historikk over forklarte setninger (tilgjengelig fra Kontrollsenter), og dobbeltklikker på snapshotet.
7. Valgfritt: Hvis du vil slå av snapshot-funksjonen, gir du kommandoen **set current explain snapshot=no** etter at du har sendt SQL-setningene.

Opprette forklarings-snapshot for statiske SQL-setninger

Merk: Opplysningene om hvordan du oppretter forklarings-snapshot i denne delen, er gitt bare for informasjon. Siden det følger med eksempler på forklarings-snapshot, trenger du ikke å utføre denne oppgaven når du skal arbeide med veiledningen.

Følg disse instruksjonene når du skal lage et forklarings-snapshot for en statisk SQL-setning:

1. Hvis DB2 ikke allerede er startet, gir du kommandoen **db2start**.
2. Kontroller at forklaringstabellene finnes i databasen din. Følg instruksjonene under Opprette forklaringstabeller når du skal gjøre dette.
3. Fra DB2-klarmeldingen i kommandolinjebehandleren knytter du deg til databasen du vil bruke. Når du for eksempel skal koble deg til SAMPLE-databasen, gir du kommandoen **connect to sample**.
4. Opprett et forklarings-snapshot for en statisk SQL-setning med parameteren EXPLSNAP når du binder eller klargjør applikasjonen. Du kan for eksempel gi kommandoen **bind** *din fil* **explsnap yes**.
5. Valgfritt: Hvis du vil se tilgangsplandiagrammet for snapshotet, fornyer du vinduet Historikk over forklarte setninger (tilgjengelig fra Kontrollsenter), og dobbeltklikker på snapshotet.

Hvis du vil vite hvordan du bruker parameteren EXPLSNAP for tilsvarende programmeringsgrensesnitt (APIer), kan du se under delene for hver av disse i *Application Development Guide*.

Det neste

I "Leksjon 2. Vise og bruke et tilgangsplandiagram" på side 7 får du lære hvordan du viser et tilgangsplandiagram, og hvordan du skal forstå innholdet.

Leksjon 2. Vise og bruke et tilgangsplandiagram

I denne leksjonen skal du bruke vinduet Tilgangsplandiagram til å vise og bruke et tilgangsplandiagram. Et tilgangsplandiagram er en grafisk representasjon av en tilgangsplan. Fra et tilgangsplandiagram kan du vise detaljene for:

- tabeller (og tilknyttede kolonner) og indekser
- Operatorer (for eksempel tabellsøkinger, sorteringer og kombineringer)
- tabellplasser og -funksjoner

Du kan vise et tilgangsplandiagram ved hjelp av disse metodene:

- Velge fra en liste over tidligere forklarte setninger.
- Velge fra en liste over forklarlige setninger i en pakke.
- Forklare en SQL-setning dynamisk.

Siden du skal arbeide med tilgangsplandiagrammer for de eksemplene på forklarings-snapshot som du lastet inn i leksjon 1, skal du velge fra en liste over tidligere forklarte setninger. Hvis du vil ha opplysninger om andre metoder for å vise tilgangsplandiagrammer, kan du se i hjelpen for Visuell forklaring.

Vise et tilgangsplandiagram ved å velge fra en liste over tidligere forklarte SQL-setninger

Slik viser du et tilgangsplandiagram ved å velge fra en liste over tidligere forklarte setninger:

1. Utvid objektoversikten fra Kontrollsenter til du finner SAMPLE-databasen.
2. Høyreklikk på databasen og velg **Vis historikk over forklarte setninger** fra objektmenyen. Vinduet Historikk over forklarte setninger blir åpnet.
3. Du kan bare vise et tilgangsplandiagram for en setning som har et forklart snapshot. Kvalifiserte setninger vil ha JA (YES) i kolonnen **Forklarings-snapshot**. Dobbeltklikk på posten som er identifisert som spørring nummer 1 (du må kanskje bla til høyre for å finne kolonnen **Nummer på spørring**). Vinduet Tilgangsplandiagram for setningen blir åpnet.

Merk: Diagrammet leses fra bunnen til toppen. Det første trinnet i spørringen blir vist nederst i diagrammet, og det siste trinnet blir vist øverst.

Lese symbolene i et tilgangsplandiagram

Tilgangsplandiagrammet viser et tre som fremstiller strukturen til en tilgangsplan. *Nodene* i treet fremstiller dette:

- Tabeller, vist som rektangler
- Indekser, vist som ruter
- Operatorer, vist som åttekanter TQUEUE-operatorer, vist som parallellogrammer
- Tabellfunksjoner, vist som sekskanter

For operatorer er tallet i hakeparentes til høyre for operatortypen en entydig ID for hver node. Tallet nedenfor operatortypen er den kumulative kostnaden.

Bruk zoomeskalaen til å forstørre deler av et diagram

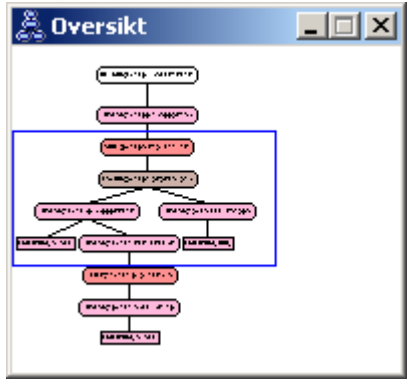
Når du viser et tilgangsplandiagram, blir hele diagrammet vist, slik at du kanskje ikke er i stand til å se detaljene som skiller hver enkelt node.

Fra vinduet Tilgangsplandiagram bruker du **zoomeskalaen** til å forstørre deler av et diagram:

1. Plasser musepekeren over den lille posisjonsviseren i zoomeskalaen på venstre side av diagrammet.
2. Venstreklikk og dra skalaen til diagrammet har det forstørrelsesnivået du ønsker.

Hvis du vil se andre deler av diagrammet, bruker du blafeltet.

Hvis du vil vise et stort og komplisert tilgangsplandiagram, bruker du vinduet Diagramoversikt. Du kan bruke dette vinduet for å se hvilke deler av diagrammet som vises, og zoome inn på eller bla gjennom diagrammet. Delen i zoomefeltet blir vist i tilgangsplanen.



Hvis du vil bla gjennom diagrammet, plasserer du musepekeren over det uthevede området i vinduet Diagramoversikt, trykker og holder nede museknapp 1. Flytt deretter musen til du ser den delen av tilgangsplandiagrammet du ønsker.

Få flere detaljer om objektene i et diagram

Du kan få tilgang til mer informasjon om objektene i et tilgangsplandiagram. Du kan vise dette:

- Systemkatalogstatistikk for objekter, for eksempel:
  - tabeller, indekser eller tabellfunksjoner
  - Informasjon om operatorer, for eksempel kostnad, egenskaper og inndataargumenter
  - Innebygde funksjoner eller brukerdefinerte funksjoner
  - Tabellplasser
  - kolonner det er referert til i en SQL-setning
- Informasjon om konfigurasjonsparametere og bindingsalternativer (optimaliseringsparametere).

Få statistikk for tabeller, indekser og tabellfunksjoner

Hvis du vil vise katalogstatistikk for en enkelttabell (rektangel), en indeks (rute) eller en tabellfunksjon (sekskant) i et diagram, dobbeltklikker du på den tilhørende noden. Det blir åpnet et statistikkvindu for de valgte objektene, der du får opplysninger om statistikken som gjaldt da snapshotet ble opprettet, i tillegg til statistikken som for tiden finnes i systemkatalogtabellene.

Hvis du vil vise katalogstatistikk for *flere* tabeller, indekser eller tabellfunksjoner i et diagram, velger du hver enkelt ved å klikke på den (fargen endres). Velg deretter **Node–>Vis statistikk**. Det blir åpnet et statistikkvindu for hver enkelt av de valgte objektene. (Vinduene kan være stablet, og det er mulig at du må dra og slippe noen av dem for å få tilgang til alle.)

Hvis posten for **STATS_TIME** i kolonnen **Forklart** inneholder posten **Statistikken er ikke oppdatert**, fantes det ikke statistikk da optimalisatoren opprettet tilgangsplanen. Hvis optimalisatoren trengte bestemte statistikkdata for å lage tilgangsplanen, brukte den derfor standardverdier. Hvis standardstatistikken ble brukt av optimalisatoren, blir den identifisert som **(standard)** i kolonne Forklart.

Få detaljer om operatorer i et diagram

Hvis du vil vise katalogstatistikk for en enkeltoperator (åttekant), dobbeltklikker du på den tilhørende noden. Et Operatordetaljer-vindu blir åpnet for den valgte operatoren, der denne typen informasjon vises:

- Anslått kumulativ kostnad (I/U, CPU-instruksjoner og total kostnad)
- Kardinalitet (det vil si det anslåtte antall rader det er søkt i) hittil
- Tabeller som er brukt og kombinert hittil i planen
- Kolonnene til tabellene som er brukt hittil
- Predikatene som er brukt hittil, i tillegg til deres anslåtte selektivitet
- Inndataargumentene for hver operator

Hvis du vil vise detaljer for *flere* operatorer, velger du hver enkelt ved å klikke på den (den blir uthevet). Deretter velger du **Node–>Vis opplysninger**. Det blir åpnet et statistikkvindu for hver enkelt av de valgte objektene. (Vinduene kan være stablet, og det er mulig at du må dra og slippe noen av dem for å få tilgang til alle.)

Få statistikk for funksjoner

Hvis du vil vise katalogstatistikk for innebygde og brukerdefinerte funksjoner, velger du **Setning–>Vis statistikk–>Funksjoner**. Velg en eller flere poster fra listen som vises i vinduet Funksjoner, og klikk på **OK**. Vinduet Funksjonsstatistikk åpnes for hvert av funksjonene som er valgt.

Få statistikk for tabellplasser

Hvis du vil vise statistikk for tabellplasser, velger du **Setning–>Vis statistikk–>Tabellplasser**. Velg en eller flere poster fra listen som vises i vinduet Tabellplasser, og klikk på **OK**. Vinduet Tabellplasstatistikk åpnes for hvert av tabellplassene som er valgt.

Få statistikk for kolonnene i en SQL-setning

Slik henter du statistikk for kolonnene det er referert til i en SQL-setning:

1. Dobbeltklikk på en tabell i tilgangsplandiagrammet. Vinduet Tabellstatistikk blir åpnet.
2. Klikk på skjermtasten **Refererte kolonner**. Du får frem vinduet Refererte kolonner, som viser kolonnene i tabellen.
3. Velg en eller flere kolonner fra listen, og klikk på **OK**. Vinduet Statistikk for refererte kolonner åpnes for hver av kolonnene som er valgt.

Få informasjon om konfigurasjonsparameter og bindingsalternativer

Hvis du vil vise informasjon om konfigurasjonsparametere og bindingsalternativer (optimaliseringsparametere), velger du **Setning–>Vis optimaliseringsparametere** fra vinduet Tilgangsplandiagram. Vinduet Optimaliseringsparametere åpnes, der du får informasjon om parameterverdiene som gjaldt da snapshotene ble laget, i tillegg til de gjeldende verdiene.

Endre utseende til et diagram

Slik endrer du verdiene som bestemmer hvordan et diagram ser ut:

1. Fra vinduet Tilgangsplandiagram velger du **Vis –> Innstillinger**. Notisboken Innstillinger for tilgangsplandiagram blir åpnet.
2. Hvis du skal endre bakgrunnsfargene, velger du flippen Diagram.
3. Hvis du vil endre fargen for forskjellige operatorer, bruker du flippene Grunnleggende, Utvid, Oppdatering og Diverse.
4. Bruk flippen Operand når du skal endre fargen på tabeller, indekser eller tabellfunksjoner.

5. Hvis du vil velge hvilken type informasjon som blir vist i operatornodene (type kostnad eller kardinalitet, som er det anslåtte antall rader som er returnert hittil), velger du flippen Operator.
6. Hvis du vil oppgi om skjemanavn eller bruker-IDer skal vises i tabellnodene, velger du flippen Operand.
7. Hvis du vil oppgi om nodene skal vises todimensjonalt eller tredimensjonalt, velger du flippen Node.
8. Hvis du vil oppdatere diagrammet med alternativene du valgte og lagre innstillingene, klikker du på **Bruk**.

Det neste

Hvis du arbeider i et databasemiljø med enkeltpartisjon, går du til "Leksjon 3. Forbedre en tilgangsplan i et databasemiljø med enkeltpartisjon" på side 13, der du får lære hvordan de ulike justeringsaktivitetene kan endre og forbedre en tilgangsplan.

Hvis du arbeider i et partisjonert databasemiljø, går du til "Kjøre en spørring uten indekser og statistikk" på side 14, der du får lære hvordan de ulike justeringsaktivitetene kan endre og forbedre en tilgangsplan.

Leksjon 3. Forbedre en tilgangsplan i et databasemiljø med enkeltpartisjon

I denne leksjonen får du lære hvordan tilgangsplanen og de tilhørende vinduene for den grunnleggende spørringen blir endret når du utfører ulike justeringsaktiviteter. Ved hjelp av en rekke eksempler og tilhørende illustrasjoner får du lære hvordan den beregnede totalkostnaden for tilgangsplanen til selv en enkel spørring, kan forbedres ved å bruke kommandoen **runstats** og tilføye passende indekser.

Når du har arbeidet med Visuell forklaring en stund, vil du finne ut andre måter å tilpasse spørringene på.

Arbeide med tilgangsplandiagrammer

Ved hjelp av de fire eksempel-snapshotene får du lære at justering er en viktig del av databaseytelsen.

Spørringene som er knyttet til forklarings-snapshotene, er nummerert fra 1 – 4. Hver spørring bruker en samme SQL-spørringen (beskrevet i Leksjon 1):

```
SELECT S.ID,S.NAVN,O.AVDELING,LØNN+PROV
FROM ORG O, STAB S
WHERE
  O.AVDNR = S.AVD AND
  S.JOBB <> 'Leder' AND
  S.LØNN+S.PROV > ALL( SELECT ST.LØNN*.9
                         FROM STAB ST
                         WHERE ST.JOBB='Leder' )
ORDER BY S.NAVN
```

Hver gjentakelse av spørringen bruker imidlertid flere justeringsteknikker enn den forrige utføringen. Spørring 1 har for eksempel ingen ytelsesjustering, mens spørring 4 har flest. Ulikhetene i spørringene blir beskrevet nedenfor:

Spørring 1
: Kjøre en spørring uten indekser og statistikk

Spørring 2
: Samle gjeldende statistikk for tabellene og indeksene i en spørring

Spørring 3
: Opprette indekser på kolonner som brukes til å kombinere tabeller i en spørring

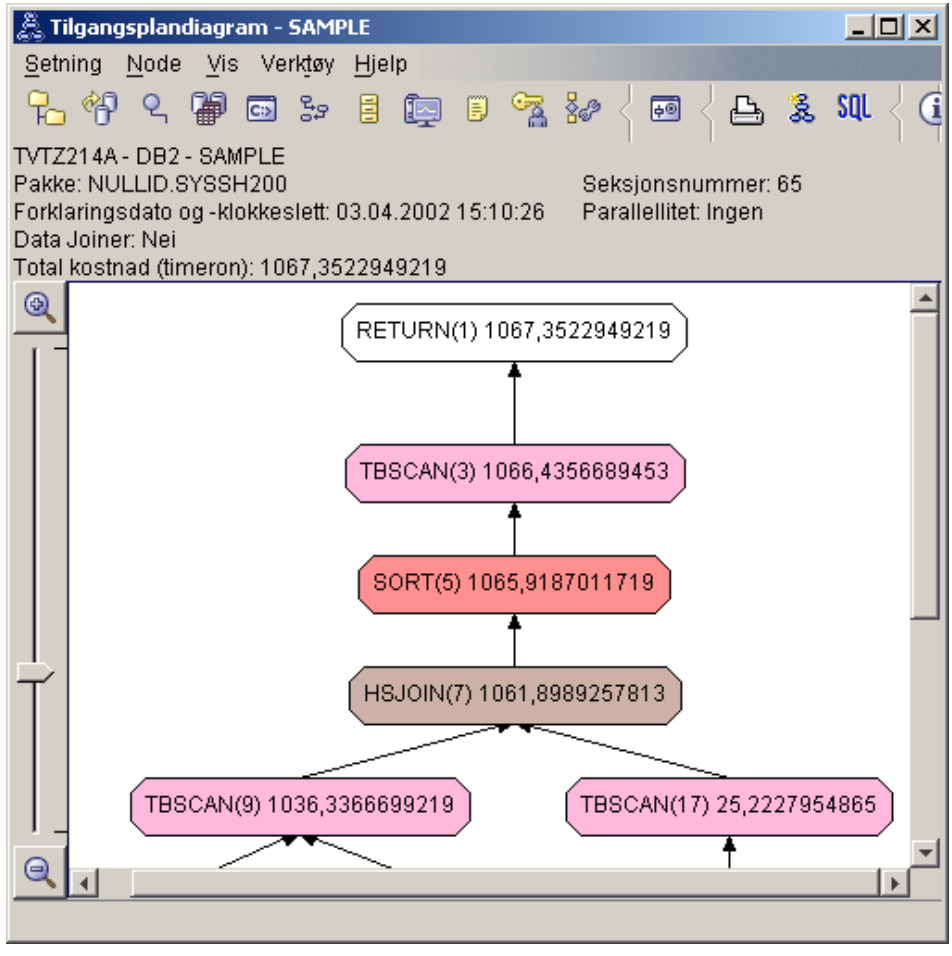
Spørring 4
: Opprette tilleggsindekser på tabellkolonner

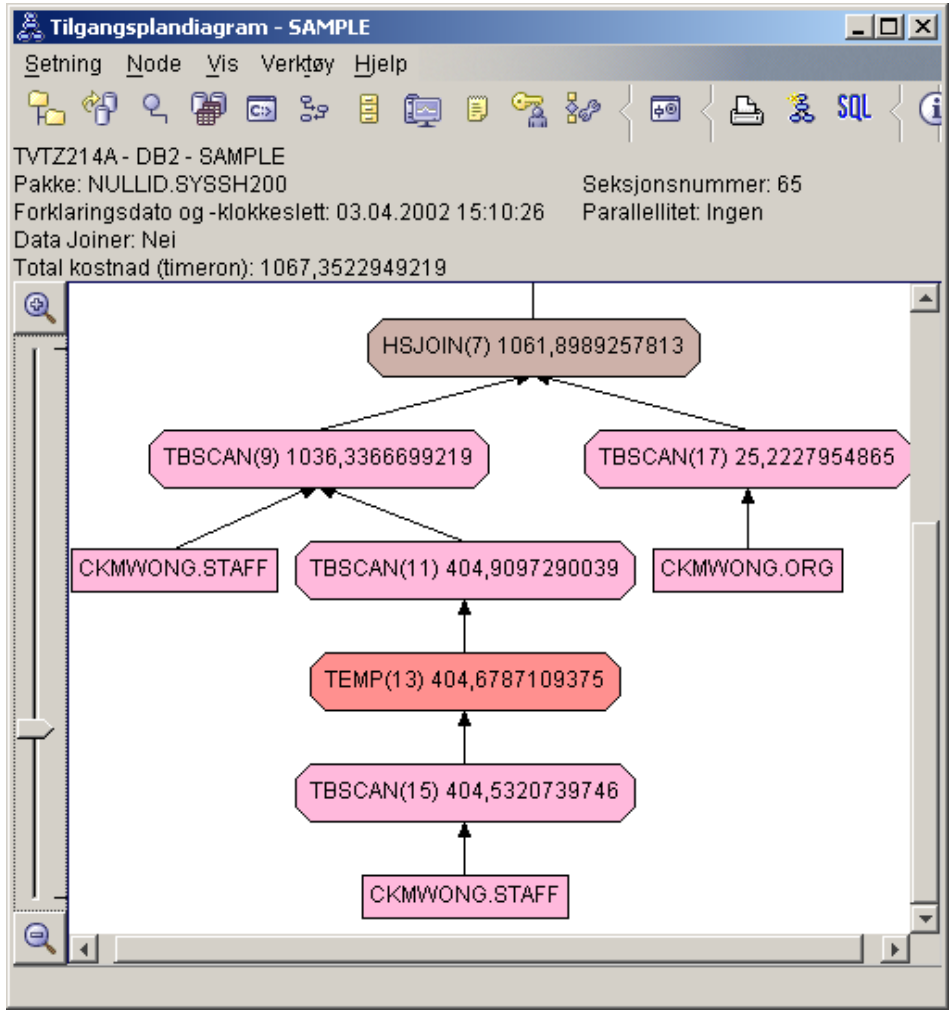
Kjøre en spørring uten indekser og statistikk

I dette eksempelet ble tilgangsplanen opprettet for SQL-spørringen uten indekser og statistikk.

Slik viser du tilgangsplandiagrammet for denne spørringen (spørring 1):

1. Utvid objektoversikten fra Kontrollsenter til du finner SAMPLE-databasen.
2. Høyreklikk på databasen og velg **Vis historikk over forklarte setninger** fra objektmenyen. Vinduet Historikk over forklarte setninger blir åpnet.
3. Dobbeltklikk på posten som er identifisert som spørring nummer 1 (du må kanskje bla til høyre for å finne kolonnen **Nummer på spørring**). Vinduet Tilgangsplandiagram for setningen blir åpnet.





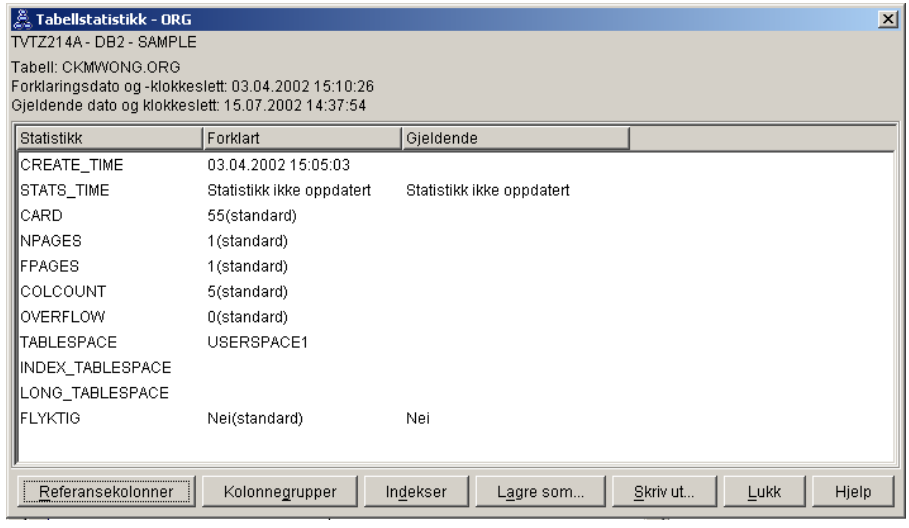
Hvis du svarer på de følgende spørsmålene, vil det hjelpe deg med å forstå hvordan du kan forbedre spørringen.

1. Finnes det gjeldende statistikk for hver tabell i spørringen?

   Hvis du vil finne ut om det finnes gjeldende statistikk for hver tabell i spørringen, dobbeltklikker du på hver tabellnode i tilgangsplandiagrammet. I det Tabellstatistikk-vinduet som blir åpnet, inneholder raden **STATS_TIME** under kolonnen **Forklart** ordene "Statistikk ikke oppdatert" hvis det ikke var samlet inn statistikk på det tidspunktet snapshotet ble opprettet.

Hvis det ikke finnes gjeldende statistikker, bruker optimalisatoren standardstatistikker som kan være forskjellige fra de faktiske statistikkene. Standardstatistikker er merket med ordet "standard" i kolonnen **Forklart** i vinduet Tabellstatistikk.

I følge opplysningene i vinduet Tabellstatistikk for ORG-tabellen, brukte optimalisatoren standardstatistikker (slik det er avmerket ved siden av de forklarte verdiene). Det ble brukt standardstatistikker fordi det ikke fantes noen faktiske statistikker tilgjengelig da snapshotet ble laget (slik det er avmerket i raden **STATS_TIME**).



2. Bruker denne tilgangsplanen de mest effektive metodene for å få tilgang til data?

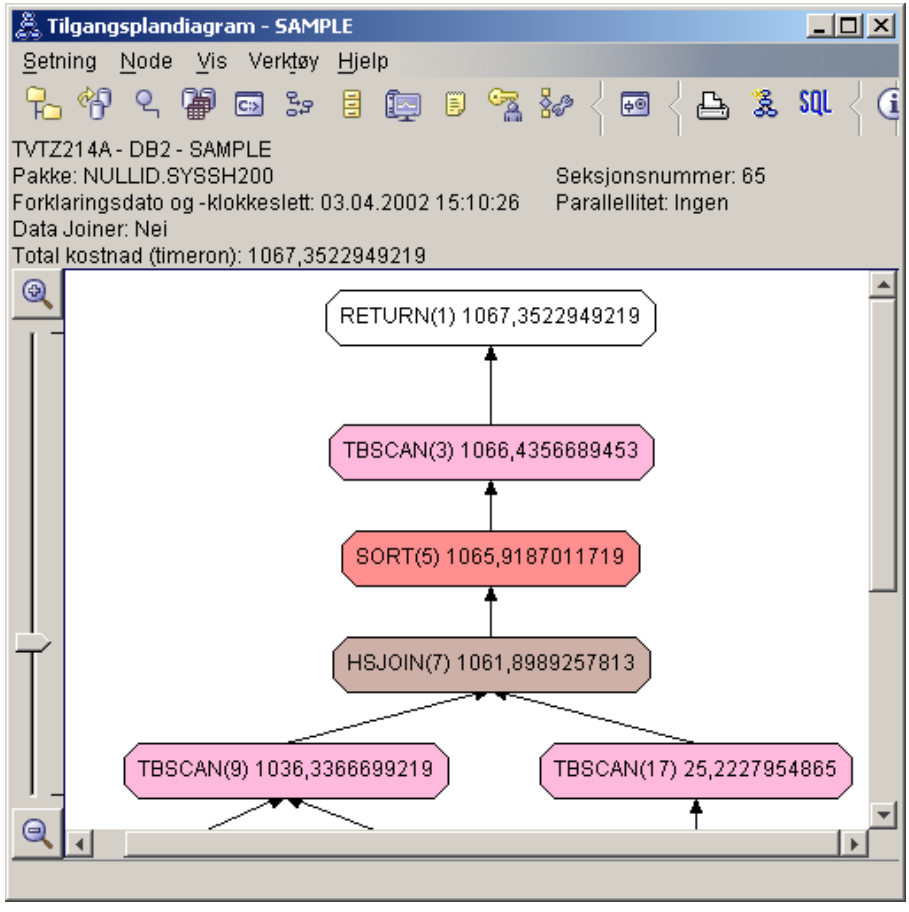
   Denne tilgangsplanen inneholder tabellsøk, og ikke indekssøk. Tabellsøk vises som åttekanter og er merket med TBSCAN. Hvis det var brukt indekssøk, ville de blitt vist som ruter, og de ville vært merket med IXSCAN. Bruken av en indeks som ble opprettet for en tabell, er mer kostnadseffektivt enn et tabellsøk hvis små mengder med data blir trukket ut.

3. Hvor effektiv er denne tilgangsplanen?

   Du kan bare finne ut hvor effektiv en tilgangsplan er hvis den er laget på grunnlag av faktiske statistikker. Siden optimalisatoren brukte standardstatistikker i denne tilgangsplanen, kan du ikke finne ut hvor effektiv den er.

   Som en regel bør du notere deg den totale anslåtte kostnaden for tilgangsplanen slik at du kan sammenlikne den med endrede tilgangsplaner. Kostnaden som blir vist i hver node, er kumulativ, fra det første trinnet i spørringen til og med noden.

I vinduet Tilgangsplandiagram er totalkostnaden omtrent 1 067 timeron, som blir vist i **RETURN (1)** øverst i diagrammet. Den totale anslåtte kostnaden blir også vist øverst i vinduet.



4. Hva nå?

   Spørring 2 ser på en tilgangsplan for den grunnleggende spørringen etter at kommandoen **runstats** er kjørt. Bruk av kommandoen **runstats** gir optimalisatoren gjeldende statistikk for alle tabellene som spørringen har tilgang til.

Samle gjeldende statistikk for tabellene og indeksene ved hjelp av runstats

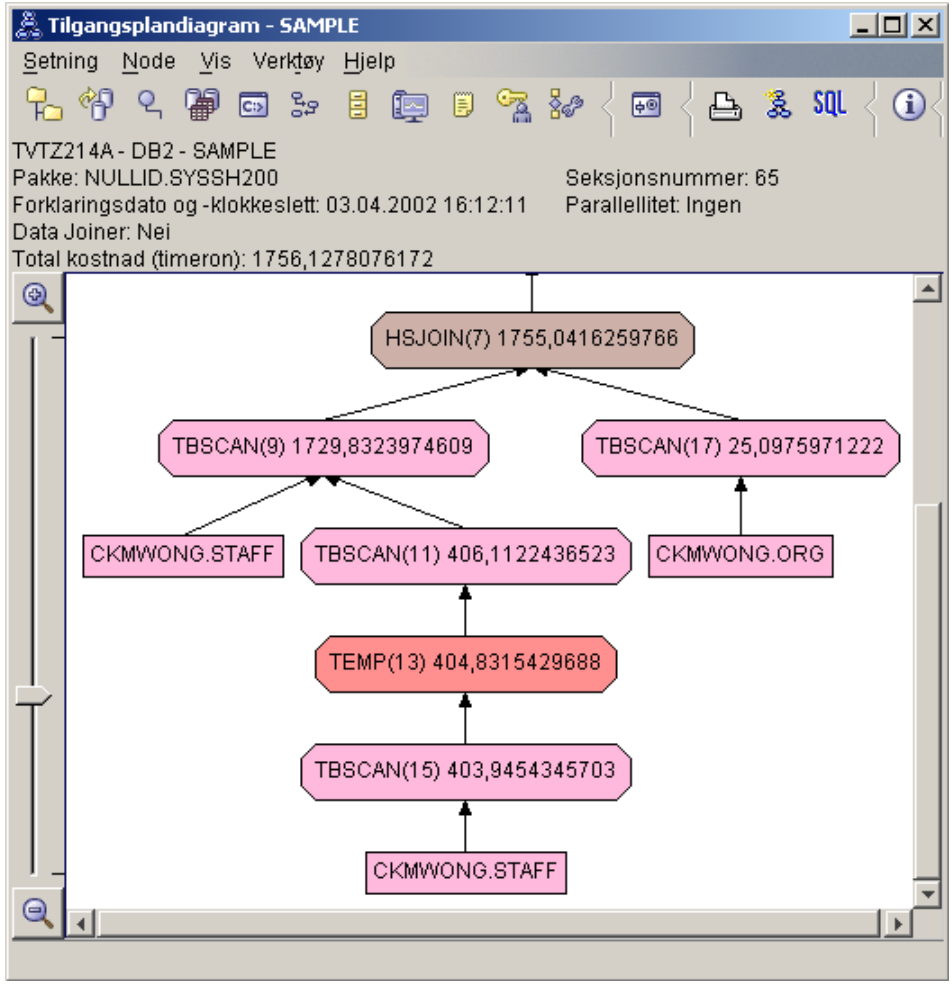
Dette eksempelet bygger på tilgangsbanen som er beskrevet i spørring 1 ved å samle inn gjeldende statistikk med kommandoen **runstats**.

Det anbefales at du bruker **runstats**-kommandoen til å samle gjeldende statistikk på tabeller og indekser, spesielt hvis det er foretatt en oppdatering

eller det er opprettet nye indekser siden sist gang **runstats**-kommandoen ble utført. Dette gir optimalisatoren de mest nøyaktige opplysningene som den kan bruke til å velge den beste tilgangsplanen.Hvis gjeldende statistikk ikke er tilgjengelig, kan optimalisatoren velge en ineffektiv tilgangsplan basert på unøyaktig standardstatistikk.

Husk å bruke **runstats** *etter* at du har oppdatert tabellene. Hvis ikke, tror optimalisatoren at tabellen er tom. Problemet er åpenbart hvis kardinaliteten i vinduet for operatordetaljer er null. I dette tilfellet må du fullføre oppdateringene av tabellen, kjøre kommanoen **runstats** på nytt og gjenoppette forklarings-snapshotene for de tabellene det gjelder.

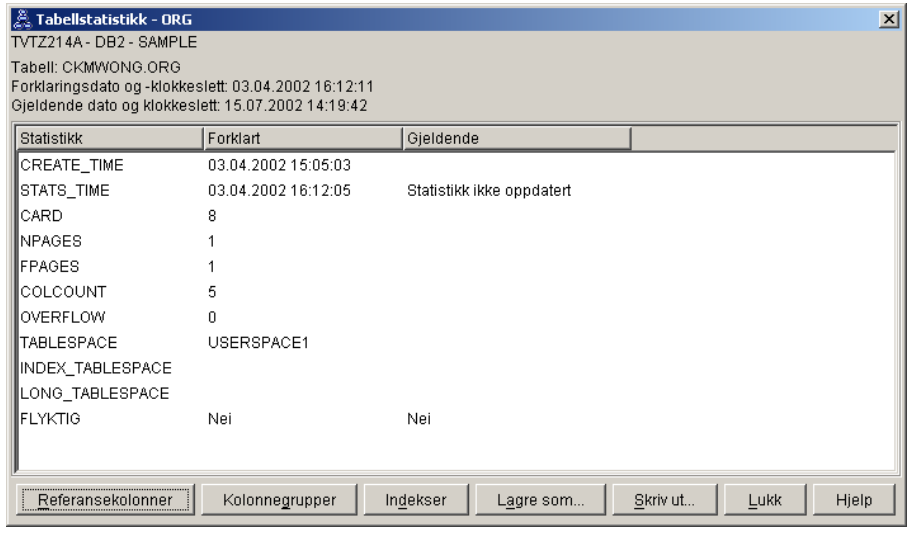
Hvis du vil vise tilgangsplandiagrammet for denne spørringen (spørring 2), dobbeltklikker du på posten som er identifisert som spørring nummer 2, i vinduet Historikk over forklarte setninger. Vinduet Tilgangsplandiagram for denne utføringen av setningen, blir åpnet.



Hvis du svarer på de følgende spørsmålene, vil det hjelpe deg med å forstå hvordan du kan forbedre spørringen.

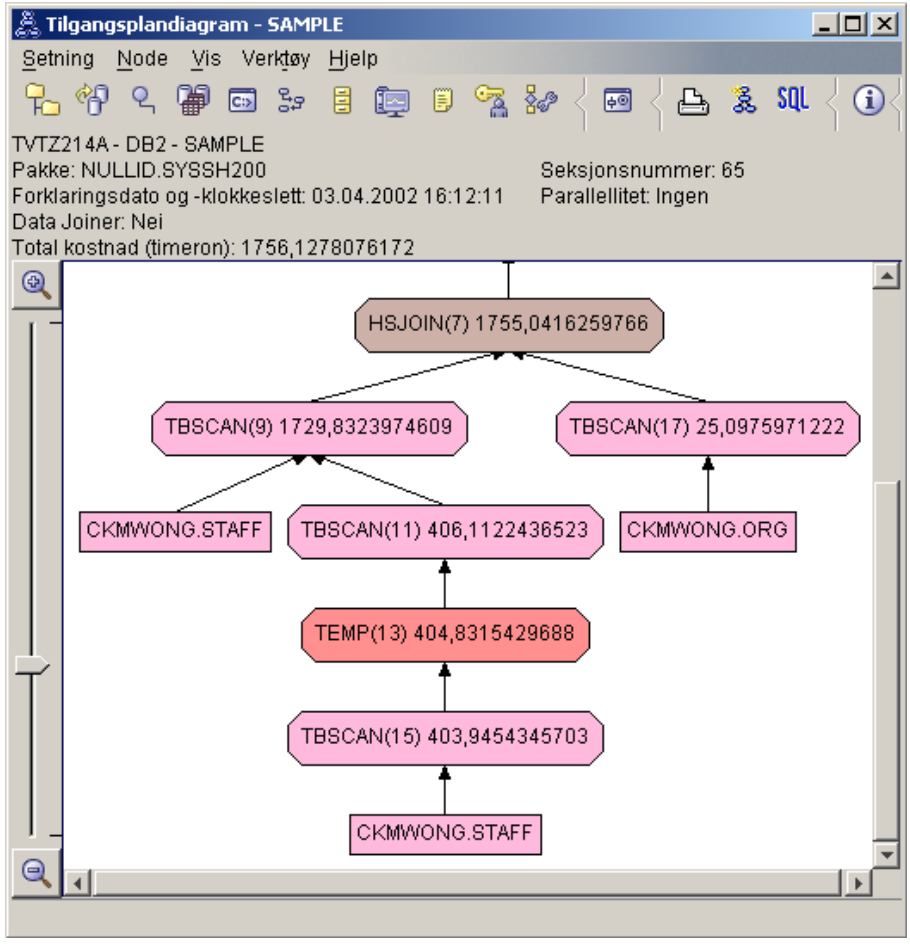
1. Finnes det gjeldende statistikk for hver tabell i spørringen?

   Vinduet Tabellstatistikk for tabellen ORG viser at optimalisatoren brukte faktisk statistikk (verdien **STATS_TIME** er den faktiske tiden for innsamlingen av statistikken). Statistikkens nøyaktighet avhenger av om det det er gjort vesentlige endringer i innholdet i tabellene etter at kommandoen **runstats** er kjørt.

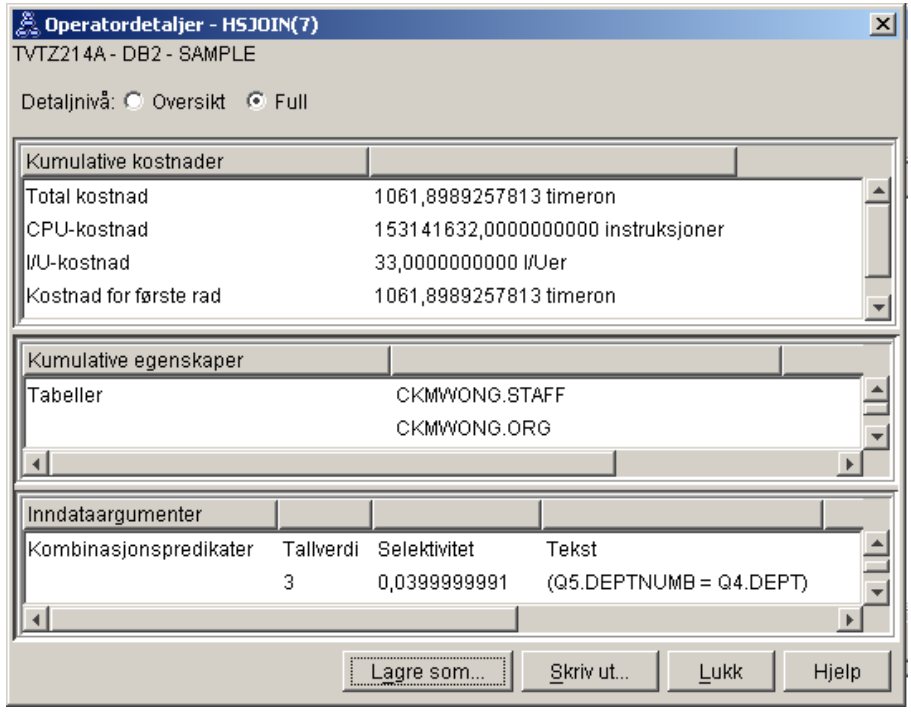


2. Bruker denne tilgangsplanen de mest effektive metodene for å få tilgang til data?

   I likhet med spørring 1, bruker tilgangsplanen i spørring 2 tabellsøk (TBSCAN) og ikke indekssøk (IXSCAN). Selv om det finnes gjeldende statistikk, ble det ikke utført noe indekssøk fordi det ikke finnes noen indekser på kolonnene som brukes av spørringen. En måte å forbedre spørringen på, vil være å gi optimalisatoren indekser på kolonner som brukes til å kombinere tabeller (det vil si kolonner som brukes i kombinasjonspredikater). I dette eksempelet er dette den første samkjøringskombinasjonen: HSJOIN (7).



I vinduet Operatordetaljer for operatoren HSJOIN (7) ser du på **Kombinasjonspredikater** under **Inndataargumenter**. Kolonnene som blir brukt i denne kombinasjonen, står under kolonnen **Tekst**. Her er det kolonnene AVDNR og AVD som er brukt.



3. Hvor effektiv er denne tilgangsplanen?

   Tilgangsplaner som er basert på oppdatert statistikk, produserer alltid en anslått kostnad (målt i timeron) som er realistisk. Siden den anslåtte kostnaden i spørring 1 var laget på grunnlag av standardstatistikk, kan vi ikke sammenlikne kostnadene i disse to tilgangsplandiagrammene for å finne ut hvilken som er mest effektiv. Det spiller ingen rolle om kostnaden er høyere eller lavere. Du må sammenlikne kostnaden i tilgangsplaner som er laget på grunnlag av faktisk statistikk, for å få et reelt mål på effektiviteten.

4. Hva nå?

   Spørring 3 ser på virkningen av å tilføye indekser for kolonnene AVDNR og AVD. Hvis du tilføyer indekser for kolonnene som blir brukt i kombinasjonspredikater, kan ytelsen bli forbedret.

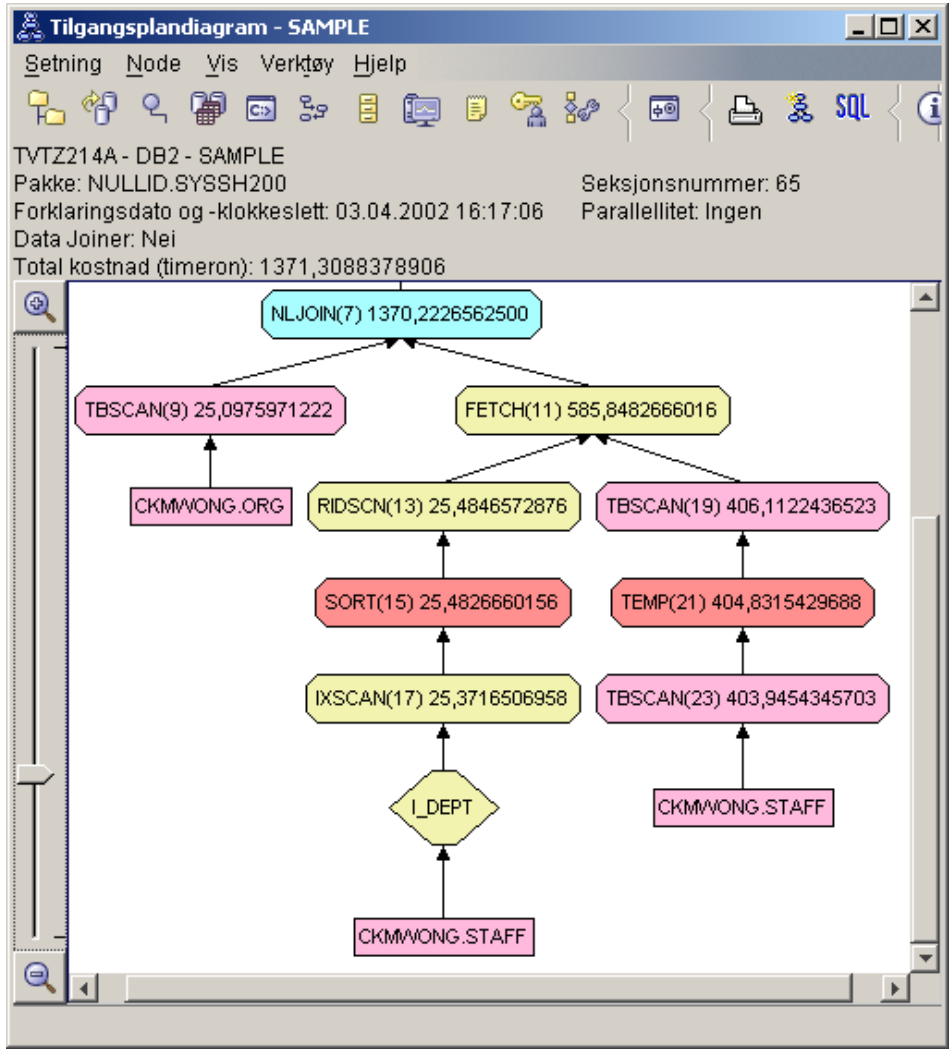
Opprette indekser på kolonner som brukes til å kombinere tabeller i en spørring

Dette eksempelet bygger på tilgangsplanen som er beskrevet i spørring 2, ved å opprette indekser for kolonnen AVD i STAB-tabellen, og for kolonnen AVDNR i ORG-tabellen.

Merk: I versjon 8 kan anbefalte indekser opprettes ved hjelp av veiviseren for arbeidsbelastningsytelse.

Hvis du vil vise tilgangsplandiagrammet for denne spørringen (spørring 3), dobbeltklikker du på posten som er identifisert som spørring nummer 3, i vinduet Historikk over forklarte setninger. Vinduet Tilgangsplandiagram for denne utføringen av setningen, blir åpnet.

Merk: Selv om det ble opprettet en indeks for AVDNR, ble den ikke brukt av optimalisatoren.

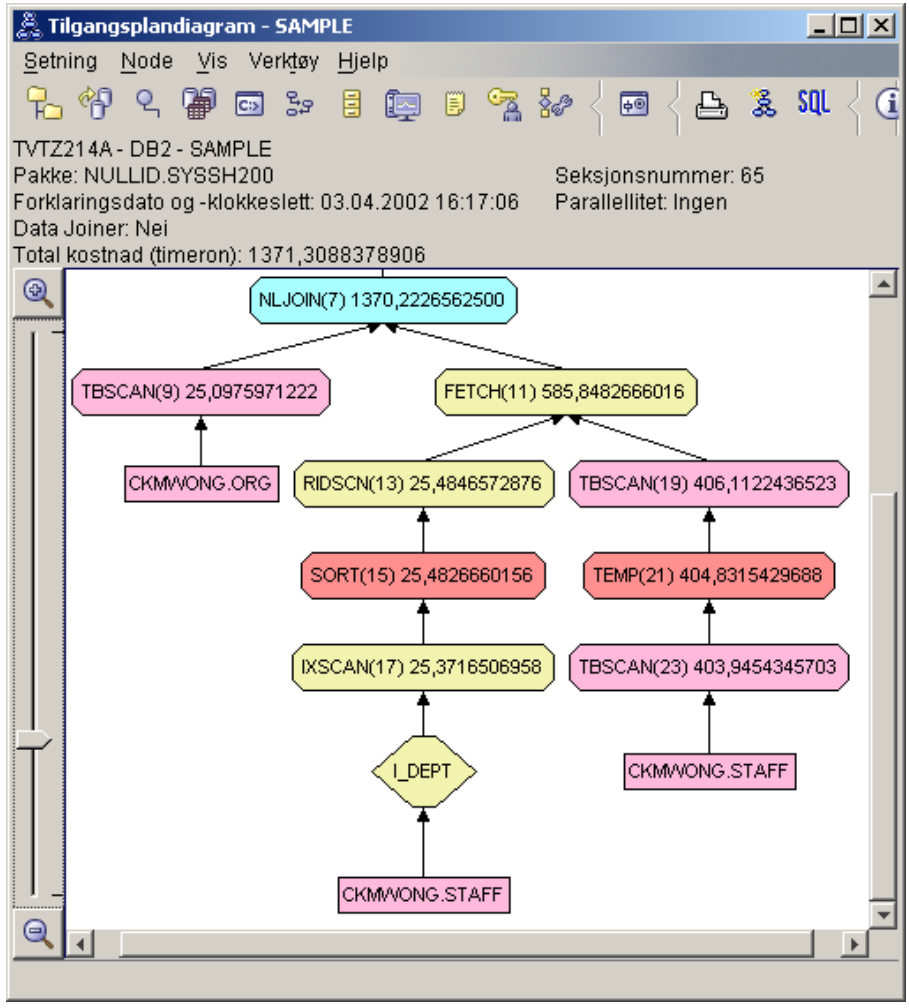


Hvis du svarer på de følgende spørsmålene, vil det hjelpe deg med å forstå hvordan du kan forbedre spørringen.

1. Hva er endret i tilgangsplanen med indekser?

En nestet sløyfekombinasjon, NLJOIN (7), har erstattet samkjøringsrutinen HSJOIN (7) som ble brukt i spørring 2. Bruk av en nestet sløyfekombinasjon resulterte i en lavere beregnet kostnad enn en samkjøringskombinasjon, fordi denne typen kombinasjon ikke krever noen sorteringstabeller eller midlertidige tabeller.

En ny ruteformet node, **I_AVD**, er tilføyd like over STAB-tabellen. Denne noden representerer indeksen som ble opprettet for AVD, og den viser at optimalisatoren har brukt et indekssøk i stedet for et tabellsøk for å finne ut hvilke rader som skulle hentes.



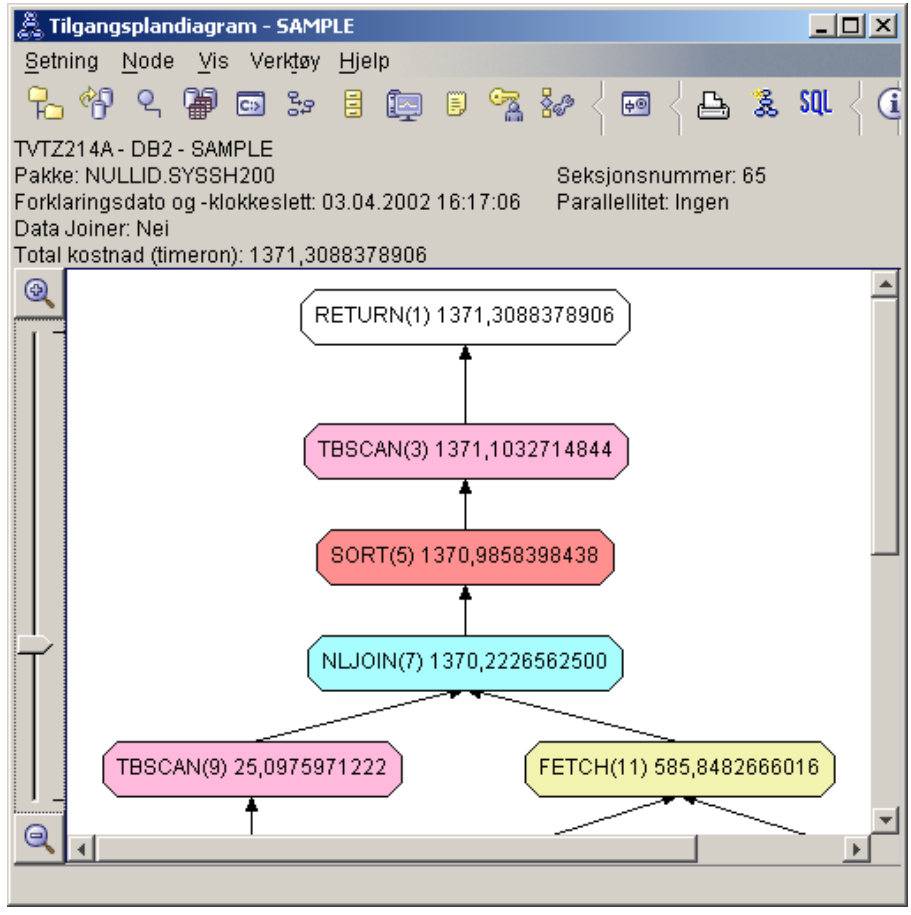
I denne delen av tilgangsplandiagrammet kan du legge merke til at det ble opprettet en ny indeks (I_AVD) på AVD-kolonnen, og at IXSCAN (17) ble brukt for å få tilgang til STAB-tabellen. I spørring 2 ble det brukt et tabellsøk for å få tilgang til STAB-tabellen.

2. Bruker denne tilgangsplanen de mest effektive metodene for å få tilgang til data?

   Fordi du har tilføyd indekser, ble en IXSCAN-node, IXSCAN (17), brukt til å få tilgang til STAB-tabellen. Spørring 2 hadde ingen indeks, og derfor ble det utført et tabellsøk i det eksempelet.

   FETCH-noden, FETCH (11), viser at i tillegg til å bruke indekssøket til å hente kolonnen AVD, hentet optimalisatoren andre kolonner fra STAB-tabellen med indeksen som peker. I dette tilfellet ble det beregnet at kombinasjonen av indekssøk og henting hadde lavere kostnad enn det fullstendige tabellsøket som ble brukt i de tidligere tilgangsplanene.

   Merk: Noden for STAB-tabellen vises to ganger, slik at den viser forbindelsen til både indeksen for AVD og til FETCH-operasjonen.

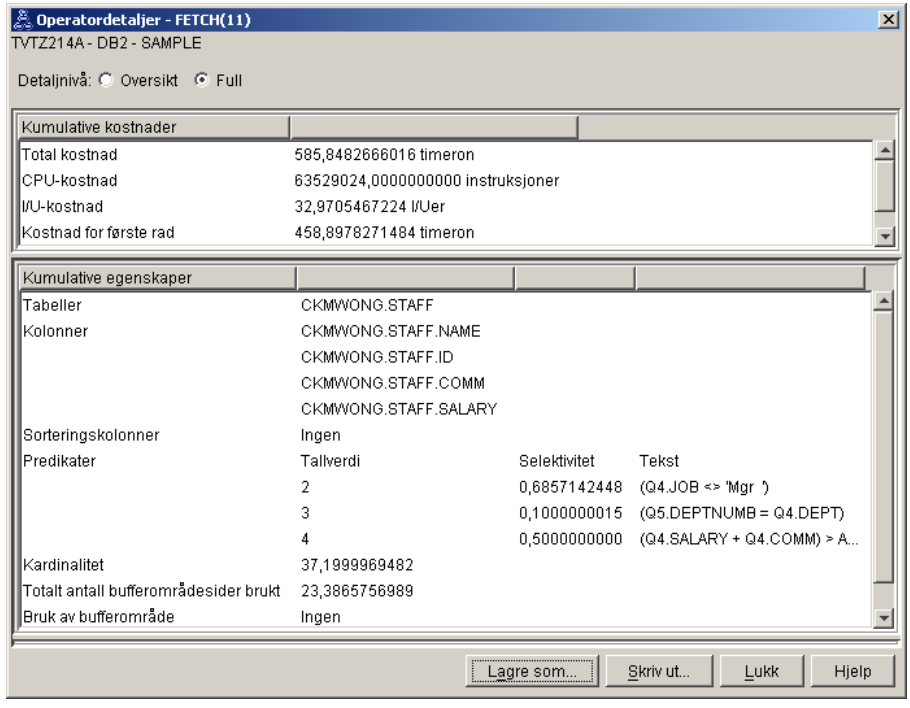


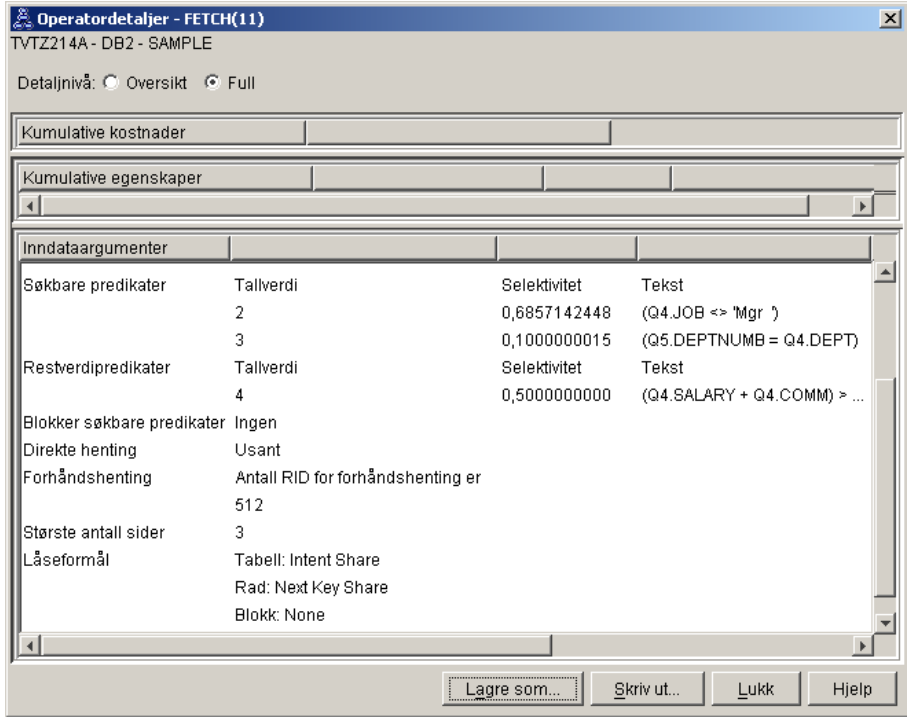
Tilgangsplanen for denne spørringen viser effekten av å opprette indekser på kolonner som brukes i kombinasjonspredikater. Indekser kan også øke

hastigheten på behandlingen av lokale predikater. La oss se på de lokale predikatene for hver tabell i denne spørringen for å finne ut hvordan tilgangsplanen blir påvirket hvis vi føyer indekser til kolonnene som det er referert til i lokale predikater.

I vinduet Operatordetaljer for operatoren FETCH (11) ser du på kolonnene under **Kumulative egenskaper**. I dette predikatet for denne henteoperasjonen brukes kolonnen JOBB, slik det er vist i delen som omhandler predikater.

Merk: Selektiviteten for dette predikatet er .69. Det betyr at med dette predikatet vil 69 % av radene bli valgt for videre behandling.





Vinduet Operatordetaljer for operatoren FETCH (11) viser kolonnene som brukes i denne operasjonen. Du kan se at AVDELING står i den første raden ved siden av **Kolonne som er hentet** under **Inndataargumenter**.

3. Hvor effektiv er denne tilgangsplanen?

   Denne tilgangsplanen er mer kostnadseffektiv enn planen i det forrige eksempelet. Den kumulative kostnaden er redusert fra rundt 1 755 timeron i spørring 2 til rundt 959 timeron i spørring 3.

   Tilgangsplanen for spørring 3 viser imidlertid et indekssøk IXSCAN (17) og en FETCH (11) for STAB-tabellen. Et indekssøk kombinert med en henteoperasjon har lavere kostnad enn et fullstendig tabellsøk, men det betyr at for hver rad som blir hentet, må programmet ha tilgang til tabellen en gang og indeksen en gang. La oss prøve å fjerne denne doble tilgangen til STAB-tabellen.

4. Hva nå?

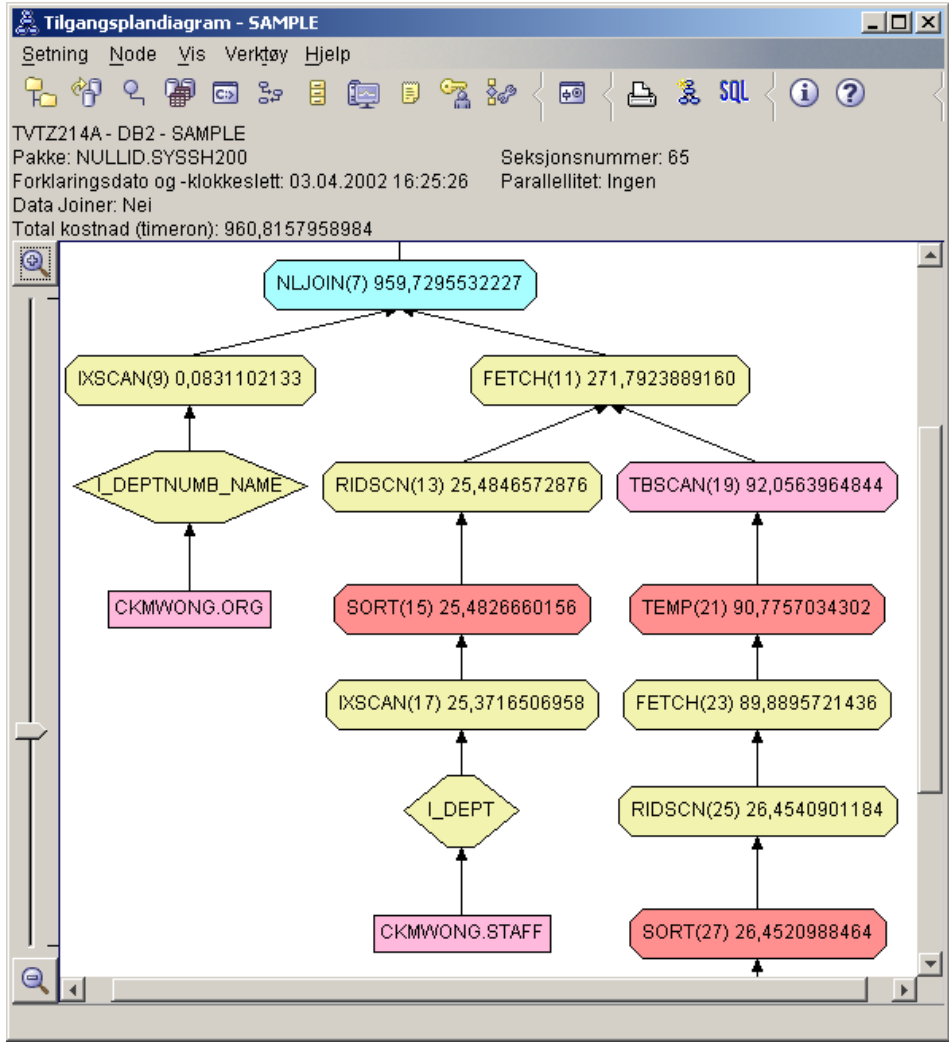
   Spørring 4 reduserer hentingen og indekssøket til et enkelt indekssøk uten henting. Hvis du oppretter flere indekser, kan det redusere den beregnede kostnaden for tilgangsplanen.

Opprette tilleggsindekser på tabellkolonner

Dette eksempelet bygger på tilgangsplanen som er beskrevet i spørring 3, ved å opprette en indeks for JOBB-kolonnen i STAB-tabellen, og tilføye

AVDELING i den eksisterende indeksen i ORG-tabellen. (Hvis du tilføyer en separat indeks, kan det gi ekstra tilgang.)

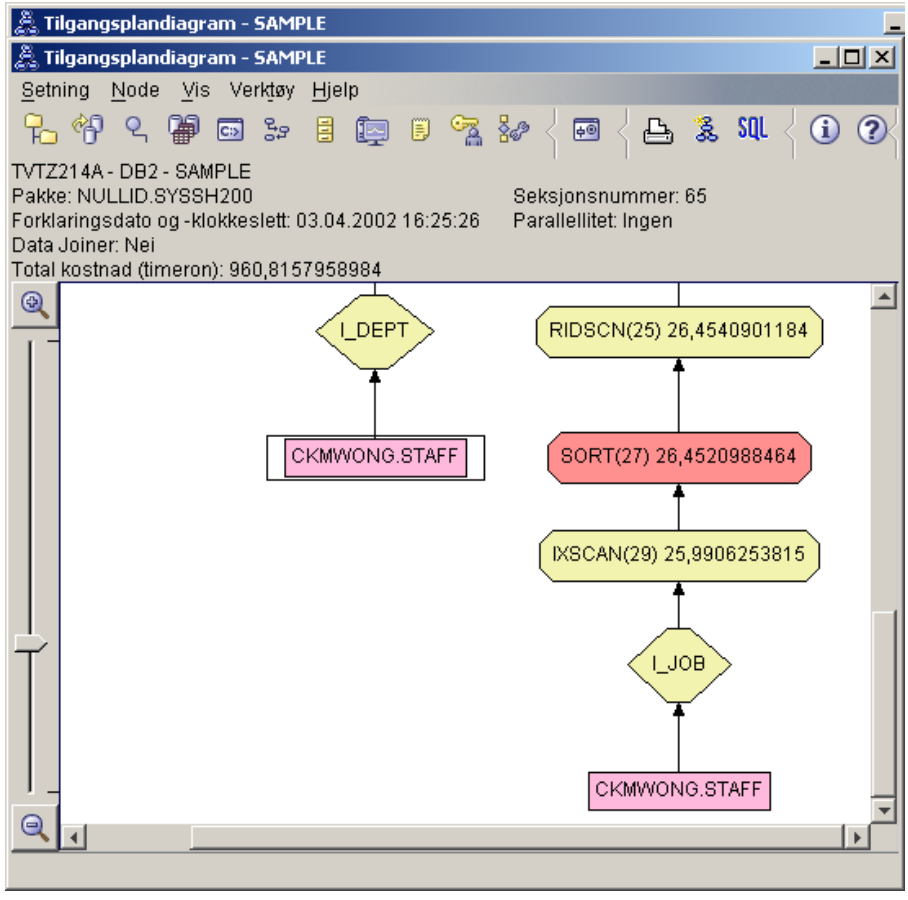
Hvis du vil vise tilgangsplandiagrammet for denne spørringen (spørring 4), dobbeltklikker du på posten som er identifisert som spørring nummer 4, i vinduet Historikk over forklarte setninger. Vinduet Tilgangsplandiagram for denne utføringen av setningen, blir åpnet.



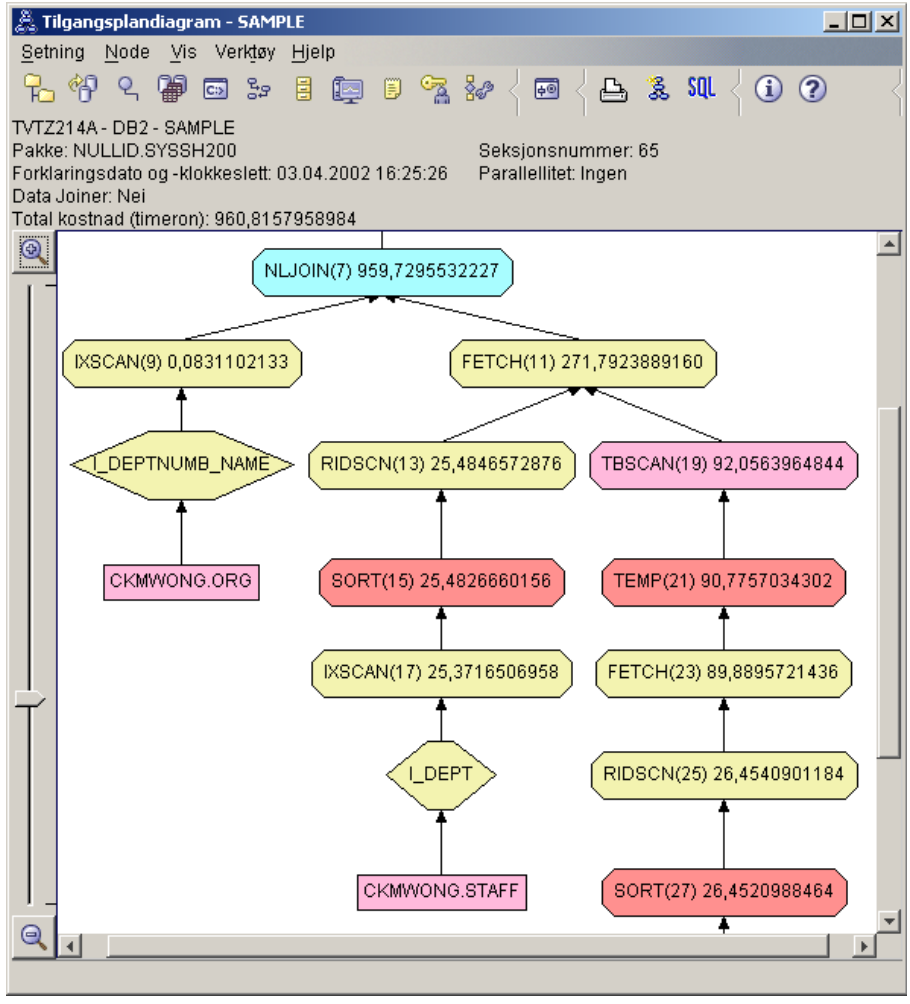
Hvis du svarer på de følgende spørsmålene, vil det hjelpe deg med å forstå hvordan du kan forbedre spørringen.

1. Hva er endret i denne tilgangsplanen på grunn av at det ble opprettet tilleggsindekser?

   Optimalisatoren har benyttet indeksen som ble laget på JOBB-kolonnen (representert av en rute med navnet **I_JOBB**) til å forbedre denne tilgangsplanen ytterligere.



I den midterste delen av tilgangsplandiagrammet kan du legge merke til at for ORG-tabellen er det forrige indekssøket og den forrige hentingen blitt endret til bare et indekssøk (IXSCAN 9). Siden AVDELING-kolonnen er føyd til indeksen på ORG-tabellen, kunne optimalisatoren fjerne den ekstra tilgangen for hentingen.



2. Hvor effektiv er denne tilgangsplanen?

   Denne tilgangsplanen er mer kostnadseffektiv enn planen i det forrige eksempelet. Den kumulative kostnaden er redusert fra rundt 1 370 timeron i spørring 3 til rundt 959 timeron i spørring 4.

Det neste

Administration Guide inneholder detaljert informasjon om andre trinn du kan bruke for å bedre ytelsen. Du kan gå tilbake til Visuell forklaring for å vurdere virkningen av handlingene.

Leksjon 4. Forbedre en tilgangsplan i et partisjonert databasemiljø

I denne leksjonen får du lære hvordan tilgangsplanen og de tilhørende vinduene for den grunnleggende spørringen blir endret når du utfører ulike justeringsaktiviteter. Ved hjelp av en rekke eksempler og tilhørende illustrasjoner får du lære hvordan den beregnede totalkostnaden for tilgangsplanen til selv en enkel spørring, kan forbedres ved å bruke kommandoen **runstats** og tilføye passende indekser.

Når du har arbeidet med Visuell forklaring en stund, vil du finne ut andre måter å tilpasse spørringene på.

Arbeide med tilgangsplandiagrammer

Ved hjelp av de fire eksempel-snapshotene får du lære at justering er en viktig del av databaseytelsen.

Spørringene som er knyttet til forklarings-snapshotene, er nummerert fra 1 – 4. Hver spørring bruker en samme SQL-spørringen (beskrevet i Leksjon 1):

```
SELECT S.ID,S.NAVN,O.AVDELING,LØNN+PROV
FROM ORG O, STAB S
WHERE
  O.AVDNR = S.AVD AND
  S.JOBB <> 'Leder' AND
  S.LØNN+S.PROV > ALL( SELECT ST.LØNN*.9
                         FROM STAB ST
                         WHERE ST.JOBB='Leder' )
ORDER BY S.NAVN
```

Hver gjentakelse av spørringen bruker imidlertid flere justeringsteknikker enn den forrige utføringen. Spørring 1 har for eksempel ingen ytelsesjustering, mens spørring 4 har flest. Ulikhetene i spørringene blir beskrevet nedenfor:

Spørring 1
: Kjøre en spørring uten indekser og statistikk

Spørring 2
: Samle gjeldende statistikk for tabellene og indeksene i en spørring

Spørring 3
: Opprette indekser på kolonner som brukes til å kombinere tabeller i en spørring

Spørring 4

Opprette tilleggsindekser på tabellkolonner

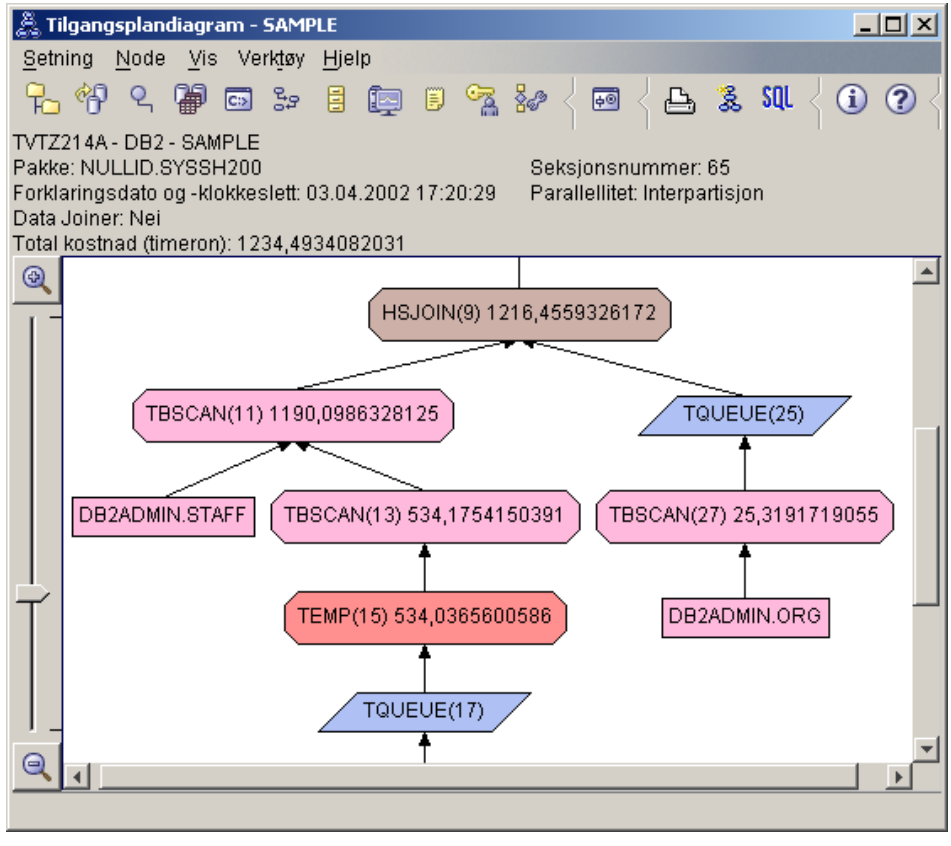
Disse eksemplene er produsert på en RS/6000 SP-maskin med 7 fysiske noder som bruker interpartisjonparallellitet.

Kjøre en spørring uten indekser og statistikk

I dette eksempelet ble tilgangsplanen opprettet for SQL-spørringen uten indekser og statistikk.

Slik viser du tilgangsplandiagrammet for denne spørringen (spørring 1):

1. Utvid objektoversikten fra Kontrollsenter til du finner SAMPLE-databasen.
2. Høyreklikk på databasen og velg **Vis historikk over forklarte setninger** fra objektmenyen. Vinduet Historikk over forklarte setninger blir åpnet.
3. Dobbeltklikk på posten som er identifisert som spørring nummer 1 (du må kanskje bla til høyre for å finne kolonnen **Nummer på spørring**). Vinduet Tilgangsplandiagram for setningen blir åpnet.



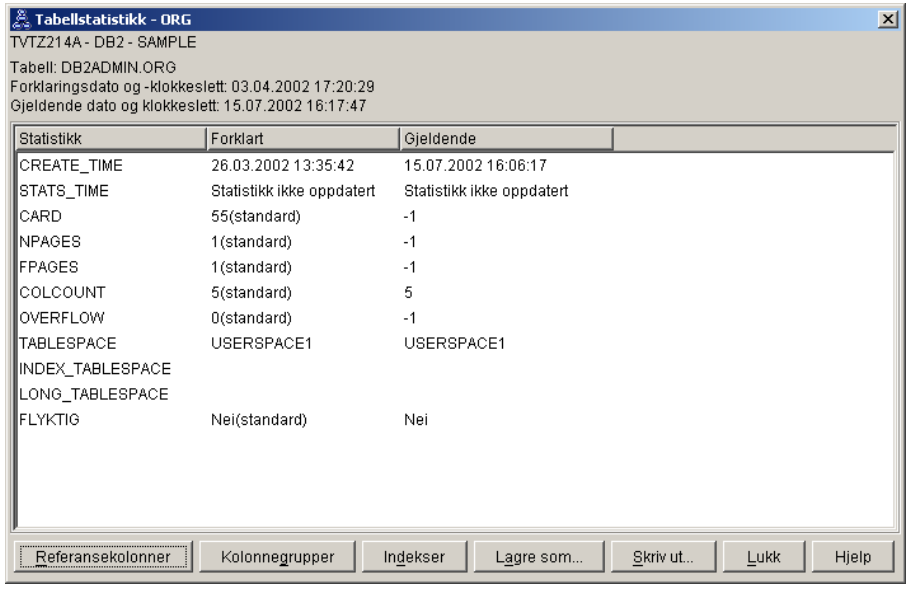
Hvis du svarer på de følgende spørsmålene, vil det hjelpe deg med å forstå hvordan du kan forbedre spørringen.

1. Finnes det gjeldende statistikk for hver tabell i spørringen?

   Hvis du vil finne ut om det finnes gjeldende statistikker for hver tabell i spørringen, dobbeltklikker du på hver tabellnode i diagrammet for tilgangsplanen. I det tilsvarende Tabellstatistikk-vinduet som blir åpnet, inneholder raden **STATS_TIME** under kolonnen **Forklart** ordene "Statistikk ikke oppdatert", noe som betyr at det ikke var samlet inn statistikk på det tidspunktet snapshotet ble opprettet.

   Hvis det ikke finnes gjeldende statistikker, bruker optimalisatoren standardstatistikker som kan være forskjellige fra de faktiske statistikkene. Standardstatistikker er merket med ordet "standard" i kolonnen **Forklart** i vinduet Tabellstatistikk.

   I følge opplysningene i vinduet Tabellstatistikk for ORG-tabellen, brukte optimalisatoren standardstatistikker (slik det er avmerket ved siden av de forklarte verdiene). Det ble brukt standardstatistikker fordi det ikke fantes noen faktiske statistikker tilgjengelig da snapshotet ble laget (slik det er avmerket i raden **STATS_TIME**).



2. Bruker denne tilgangsplanen de mest effektive metodene for å få tilgang til data?

   Denne tilgangsplanen inneholder tabellsøk, og ikke indekssøk. Tabellsøk vises som åttekanter og er merket med TBSCAN. Hvis det var brukt indekssøk, ville de blitt vist som ruter, og de ville være merket med

IXSCAN. Bruken av en indeks som ble opprettet for en tabell, er mer kostnadseffektivt enn et tabellsøk hvis små mengder med data blir trukket ut.

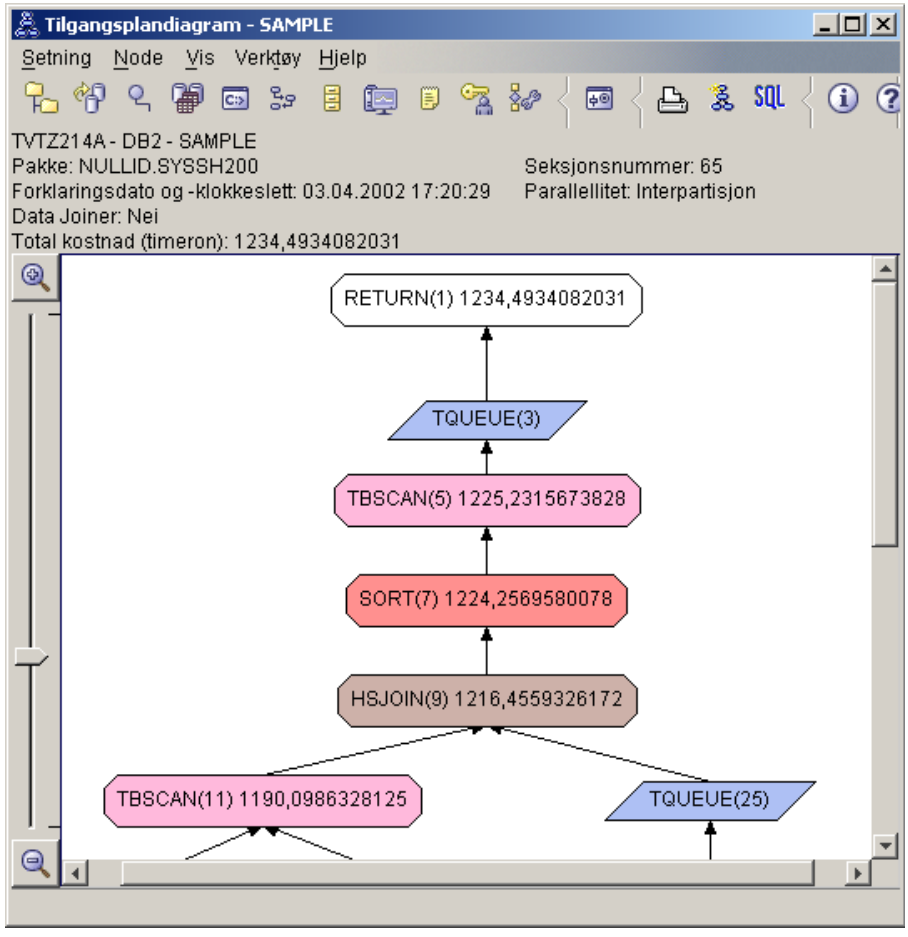
3. Hvor effektiv er denne tilgangsplanen?

   Du kan bare finne ut hvor effektiv en tilgangsplan er hvis den er laget på grunnlag av faktiske statistikker. Siden optimalisatoren brukte standardstatistikker i denne tilgangsplanen, kan du ikke finne ut hvor effektiv den er.

   Som en regel bør du notere deg den totale anslåtte kostnaden for tilgangsplanen slik at du kan sammenlikne den med endrede tilgangsplaner. Kostnaden som blir vist i hver node, er kumulativ, fra det første trinnet i spørringen til og med noden.

   Merk: For partisjonerte databaser er dette den kumulative kostnaden for den noden som bruker flest ressurser.

   I vinduet Tilgangsplandiagram er totalkostnaden omtrent 1 234 timeron, som blir vist i **RETURN (1)** øverst i diagrammet. Den totale anslåtte kostnaden blir også vist øverst i vinduet.



4. Hva nå?

   Spørring 2 ser på en tilgangsplan for den grunnleggende spørringen etter at kommandoen **runstats** er kjørt. Bruk av kommandoen **runstats** gir optimalisatoren gjeldende statistikk for alle tabellene som spørringen har tilgang til.

Samle gjeldende statistikk for tabellene og indeksene ved hjelp av runstats

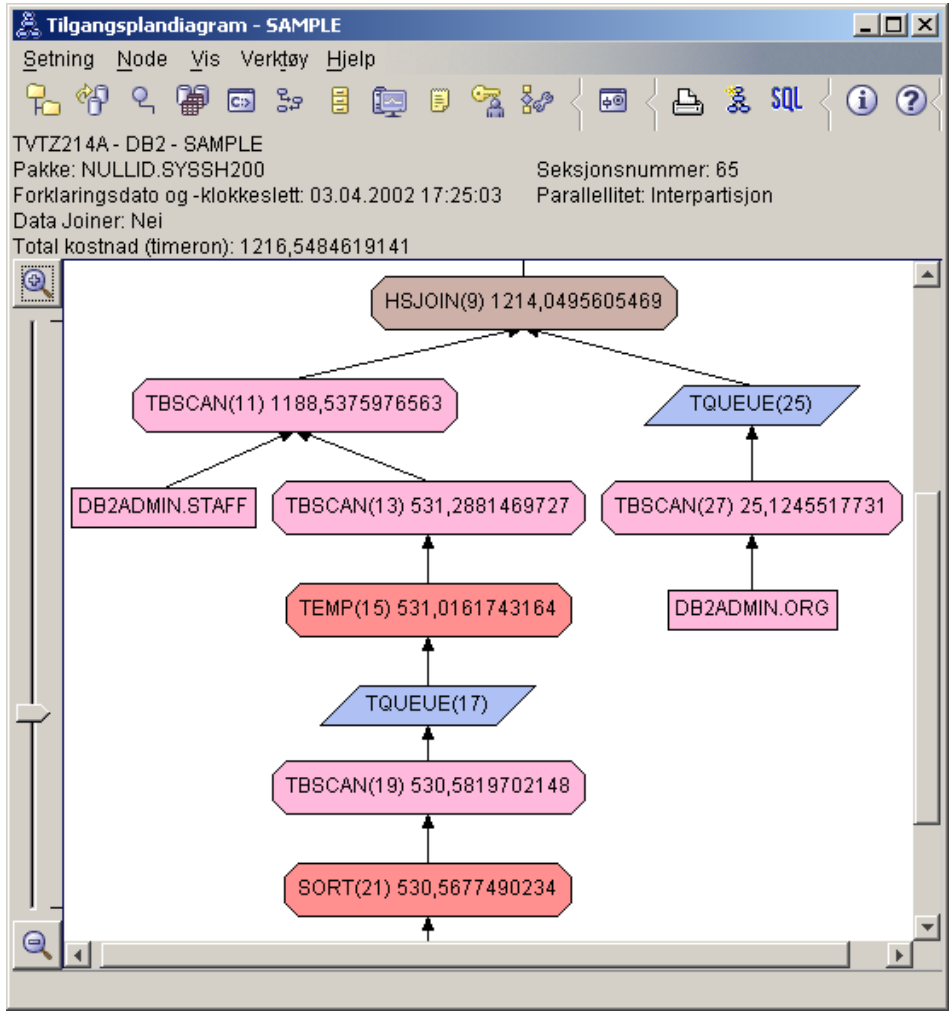
Dette eksempelet bygger på tilgangsbanen som er beskrevet i spørring 1 ved å samle inn gjeldende statistikk med kommandoen **runstats**.

Det anbefales at du bruker **runstats**-kommandoen til å samle gjeldende statistikk på tabeller og indekser, spesielt hvis det er foretatt en oppdatering eller det er opprettet nye indekser siden sist gang **runstats**-kommandoen ble utført. Dette gir optimalisatoren de mest nøyaktige opplysningene som den

kan bruke til å velge den beste tilgangsplanen.Hvis gjeldende statistikk ikke er tilgjengelig, kan optimalisatoren velge en ineffektiv tilgangsplan basert på unøyaktig standardstatistikk.

Husk å bruke **runstats** *etter* at du har oppdatert tabellene. Hvis ikke, tror optimalisatoren at tabellen er tom. Problemet er åpenbart hvis kardinaliteten i vinduet for operatordetaljer er null. I dette tilfellet må du fullføre oppdateringene av tabellen, kjøre kommanoen **runstats** på nytt og gjenoppette forklarings-snapshotene for de tabellene det gjelder.

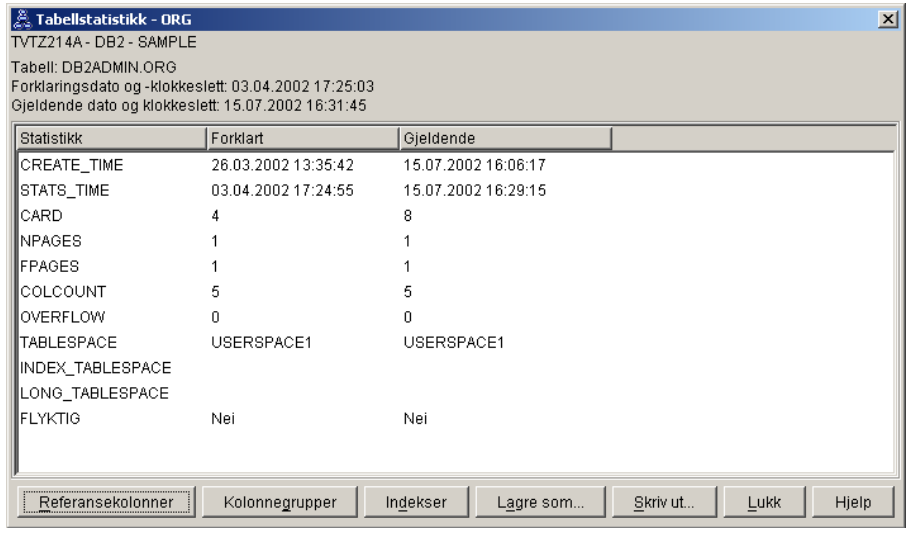
Hvis du vil vise tilgangsplandiagrammet for denne spørringen (spørring 2), dobbeltklikker du på posten som er identifisert som spørring nummer 2, i vinduet Historikk over forklarte setninger. Vinduet Tilgangsplandiagram for denne utføringen av setningen, blir åpnet.



Hvis du svarer på de følgende spørsmålene, vil det hjelpe deg med å forstå hvordan du kan forbedre spørringen.

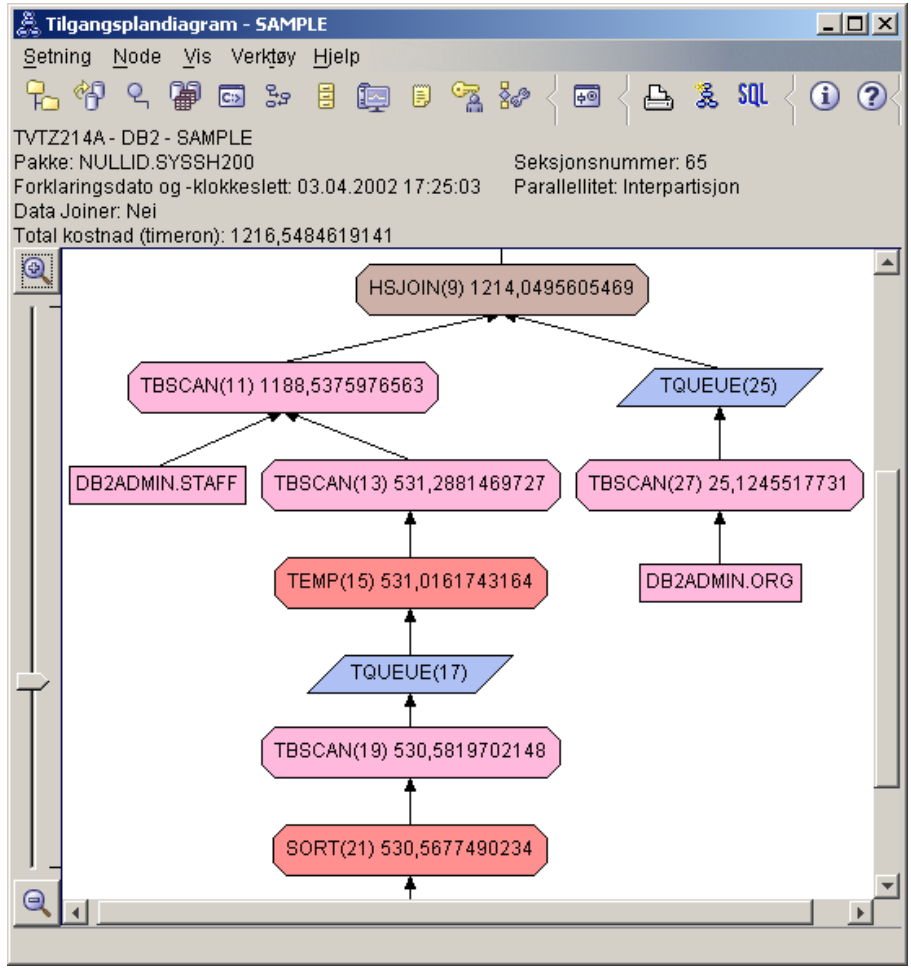
1. Finnes det gjeldende statistikk for hver tabell i spørringen?

   Vinduet Tabellstatistikk for tabellen ORG viser at optimalisatoren brukte faktisk statistikk (verdien **STATS_TIME** er den faktiske tiden for innsamlingen av statistikken). Statistikkens nøyaktighet avhenger av om det det er gjort vesentlige endringer i innholdet i tabellene etter at kommandoen **runstats** er kjørt.

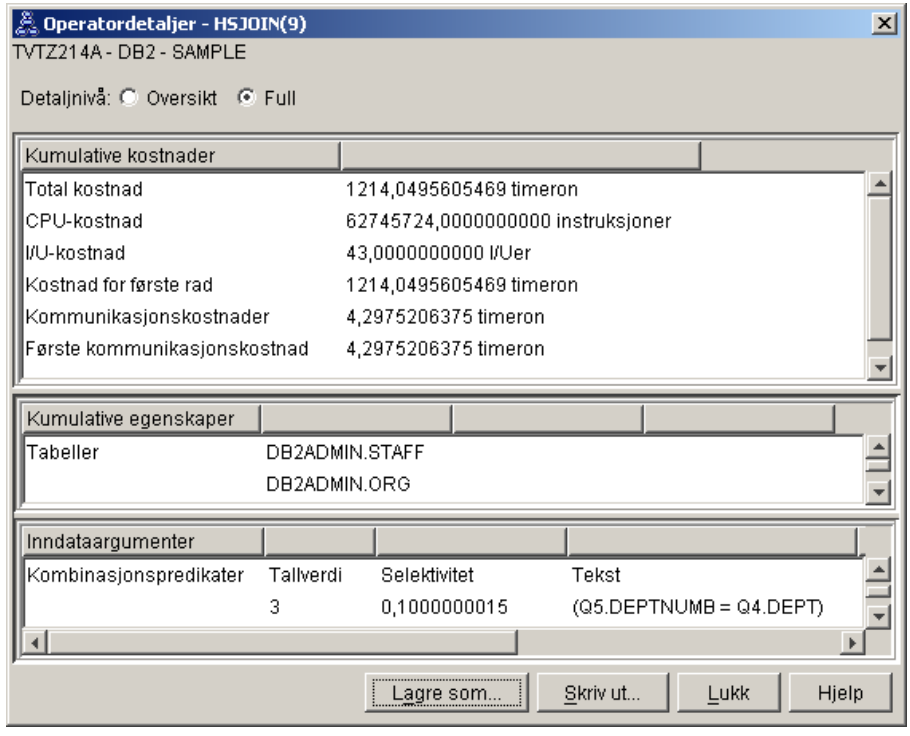


2. Bruker denne tilgangsplanen de mest effektive metodene for å få tilgang til data?

   I likhet med spørring 1, bruker tilgangsplanen i spørring 2 tabellsøk (TBSCAN) og ikke indekssøk (IXSCAN). Selv om det finnes gjeldende statistikk, ble det ikke utført noe indekssøk fordi det ikke finnes noen indekser på kolonnene som brukes av spørringen. En måte å forbedre spørringen på, vil være å gi optimalisatoren indekser på kolonner som brukes til å kombinere tabeller (det vil si kolonner som brukes i kombinasjonspredikater). I dette eksempelet er dette den første samkjøringskombinasjonen: HSJOIN (9).



I vinduet Operatordetaljer for operatoren HSJOIN (9) ser du på **Kombinasjonspredikater** under **Inndataargumenter**. Kolonnene som blir brukt i denne kombinasjonen, står under kolonnen **Tekst**. Her er det kolonnene AVDNR og AVD som er brukt.



3. Hvor effektiv er denne tilgangsplanen?

   Tilgangsplaner som er basert på oppdatert statistikk, produserer alltid en anslått kostnad (målt i timeron) som er realistisk. Siden den anslåtte kostnaden i spørring 1 var laget på grunnlag av standardstatistikk, kan vi ikke sammenlikne kostnadene i disse to tilgangsplandiagrammene for å finne ut hvilken som er mest effektiv. Det spiller ingen rolle om kostnaden er høyere eller lavere. Du må sammenlikne kostnaden i tilgangsplaner som er laget på grunnlag av faktisk statistikk, for å få et reelt mål på effektiviteten.

4. Hva nå?

   Spørring 3 ser på virkningen av å tilføye indekser for kolonnene AVDNR og AVD. Hvis du tilføyer indekser for kolonnene som blir brukt i kombinasjonspredikater, kan ytelsen bli forbedret.

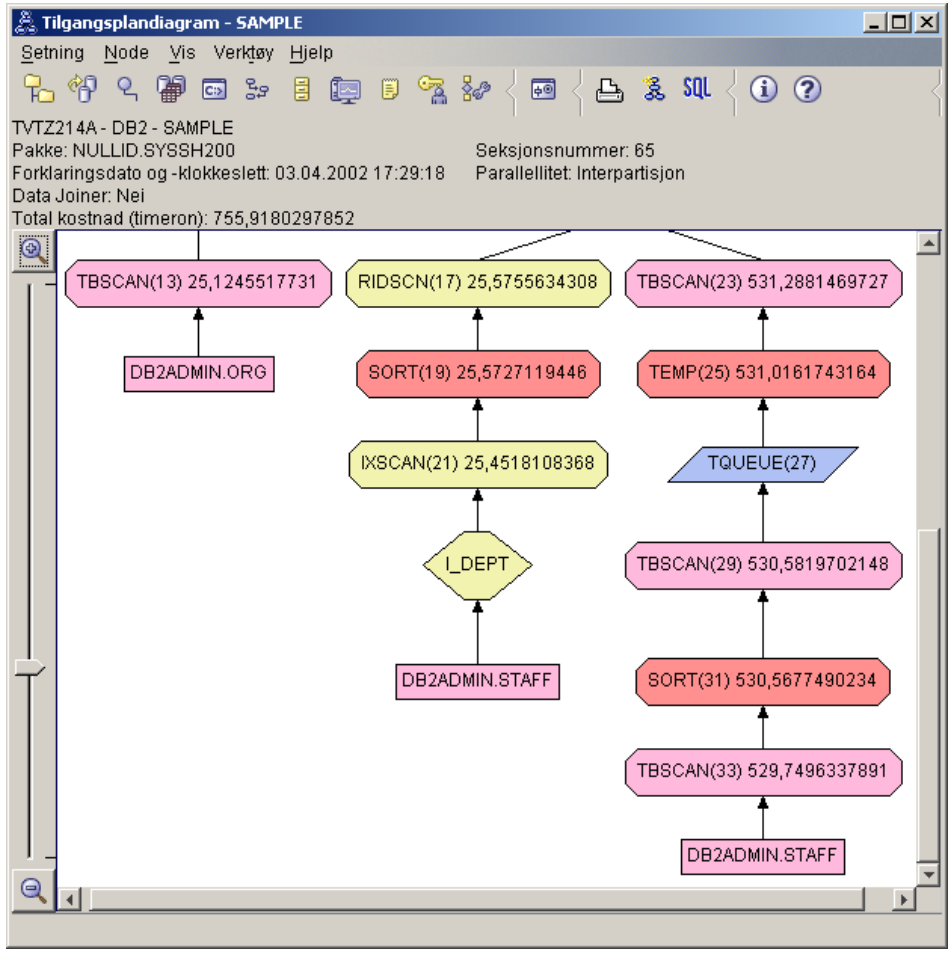
Opprette indekser på kolonner som brukes til å kombinere tabeller i en spørring

Dette eksempelet bygger på tilgangsplanen som er beskrevet i spørring 2, ved å opprette indekser for kolonnen AVD i STAB-tabellen, og for kolonnen AVDNR i ORG-tabellen.

Merk: I versjon 8 kan anbefalte indekser opprettes ved hjelp av veiviseren for arbeidsbelastningsytelse.

Hvis du vil vise tilgangsplandiagrammet for denne spørringen (spørring 3), dobbeltklikker du på posten som er identifisert som spørring nummer 3, i vinduet Historikk over forklarte setninger. Vinduet Tilgangsplandiagram for denne utføringen av setningen, blir åpnet.

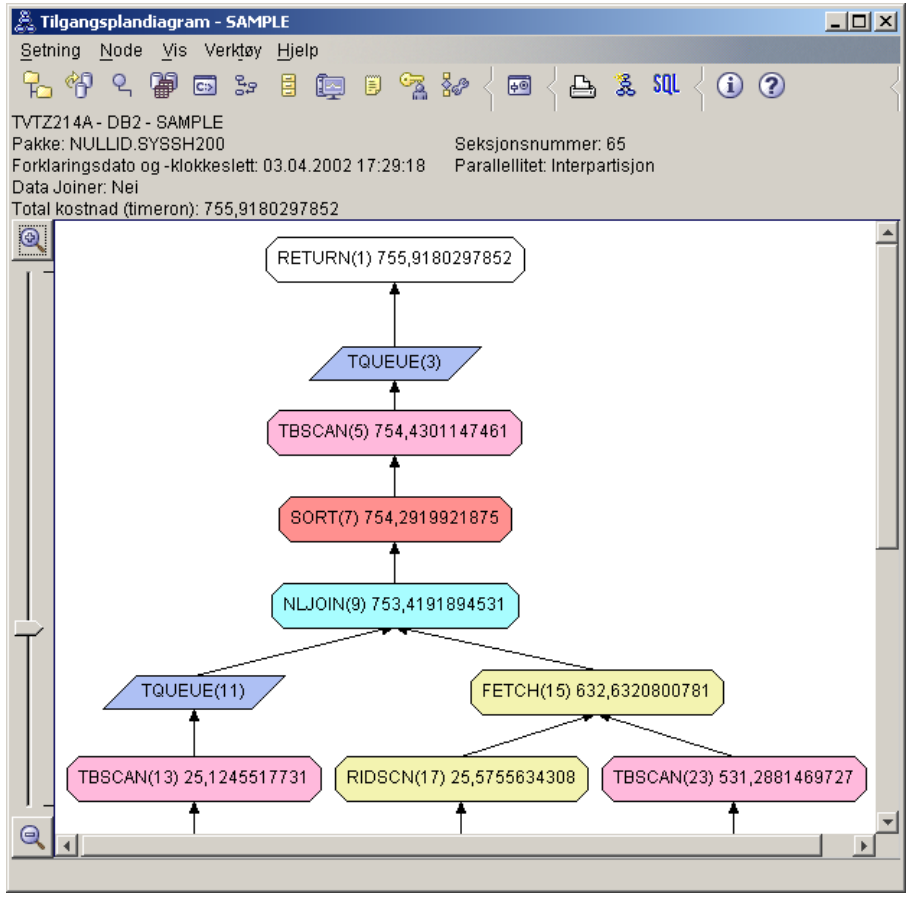
Merk: Selv om det ble opprettet en indeks for AVDNR, ble den ikke brukt av optimalisatoren.



Hvis du svarer på de følgende spørsmålene, vil det hjelpe deg med å forstå hvordan du kan forbedre spørringen.

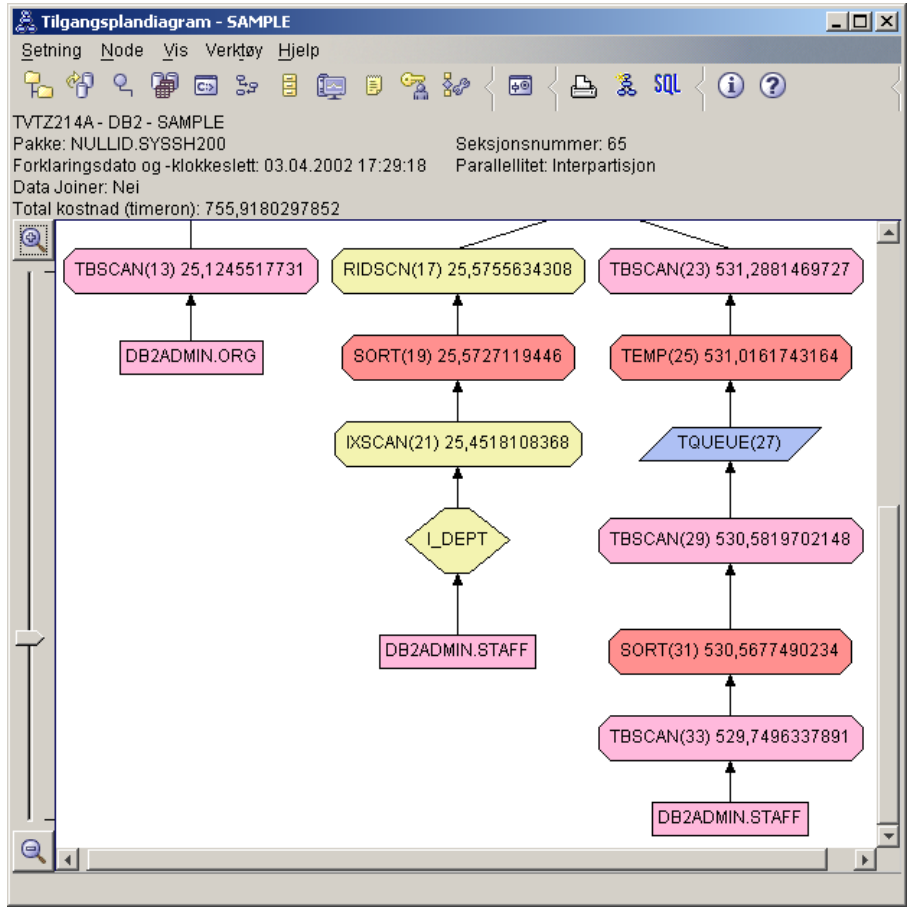
1. Hva er endret i tilgangsplanen med indekser?

En ny ruteformet node, **I_AVD**, er tilføyd like over STAB-tabellen. Denne noden representerer indeksen som ble opprettet for AVD, og den viser at optimalisatoren har brukt et indekssøk i stedet for et tabellsøk for å finne ut hvilke rader som skulle hentes.

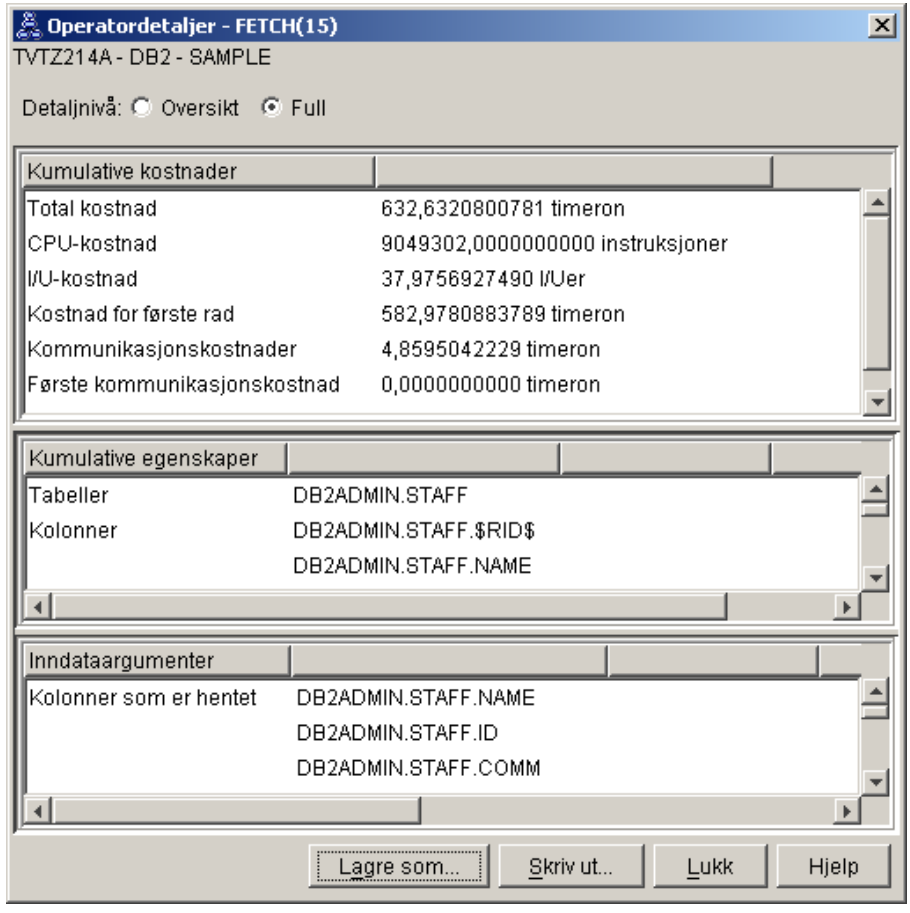


2. Bruker denne tilgangsplanen de mest effektive metodene for å få tilgang til data?

   Tilgangsplanen for denne spørringen viser virkningen av å opprette indekser for kolonnen AVDNR i ORG-tabellen, som gir resultatet FETCH (15) og IXSCAN (21), og for kolonnen AVD i STAB-tabellen. Spørring 2 hadde ingen indeks, og derfor ble det brukt et tabellsøk i det eksempelet.



Vinduet Operatordetaljer for operatoren FETCH (15) viser kolonnene som brukes i denne operasjonen.



Kombinasjonen av indeks og henting er beregnet til ha lavere kostnad enn det fullstendige tabellsøket som ble brukt i de tidligere tilgangsplanene.

3. Hvor effektiv er denne tilgangsplanen?

   Denne tilgangsplanen er mer kostnadseffektiv enn planen i det forrige eksempelet. Den kumulative kostnaden er redusert fra rundt 1 214 timeron i spørring 2 til rundt 755 timeron i spørring 3.

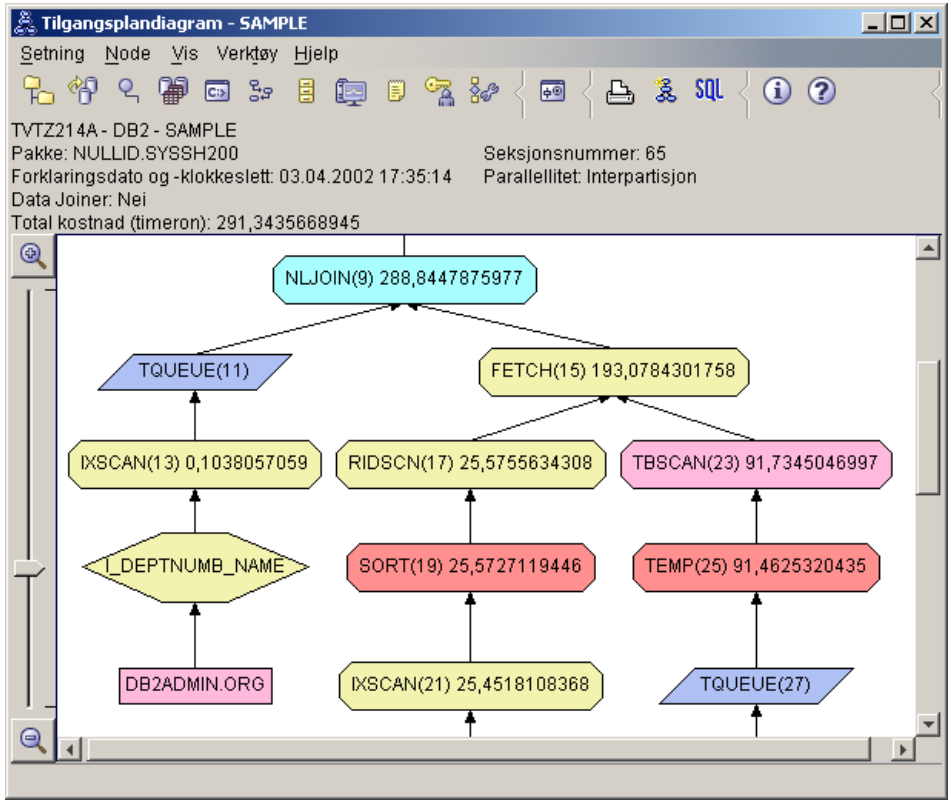
4. Hva nå?

   Spørring 4 reduserer hentingen og indekssøket til et enkelt indekssøk uten henting. Hvis du oppretter flere indekser, kan det redusere den beregnede kostnaden for tilgangsplanen.

Opprette tilleggsindekser på tabellkolonner

Dette eksempelet bygger på tilgangsplanen som er beskrevet i spørring 3, ved å opprette en indeks for JOBB-kolonnen i STAB-tabellen, og tilføye AVDELING i den eksisterende indeksen i ORG-tabellen. (Hvis du tilføyer en separat indeks, kan det gi ekstra tilgang.)

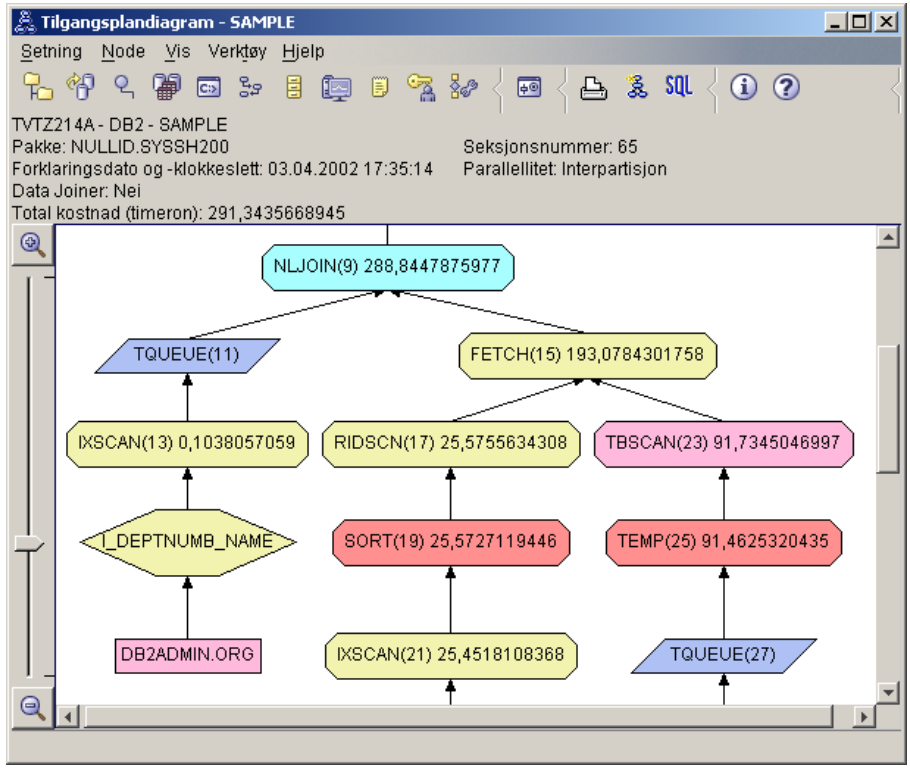
Hvis du vil vise tilgangsplandiagrammet for denne spørringen (spørring 4), dobbeltklikker du på posten som er identifisert som spørring nummer 4, i vinduet Historikk over forklarte setninger. Vinduet Tilgangsplandiagram for denne utføringen av setningen, blir åpnet.



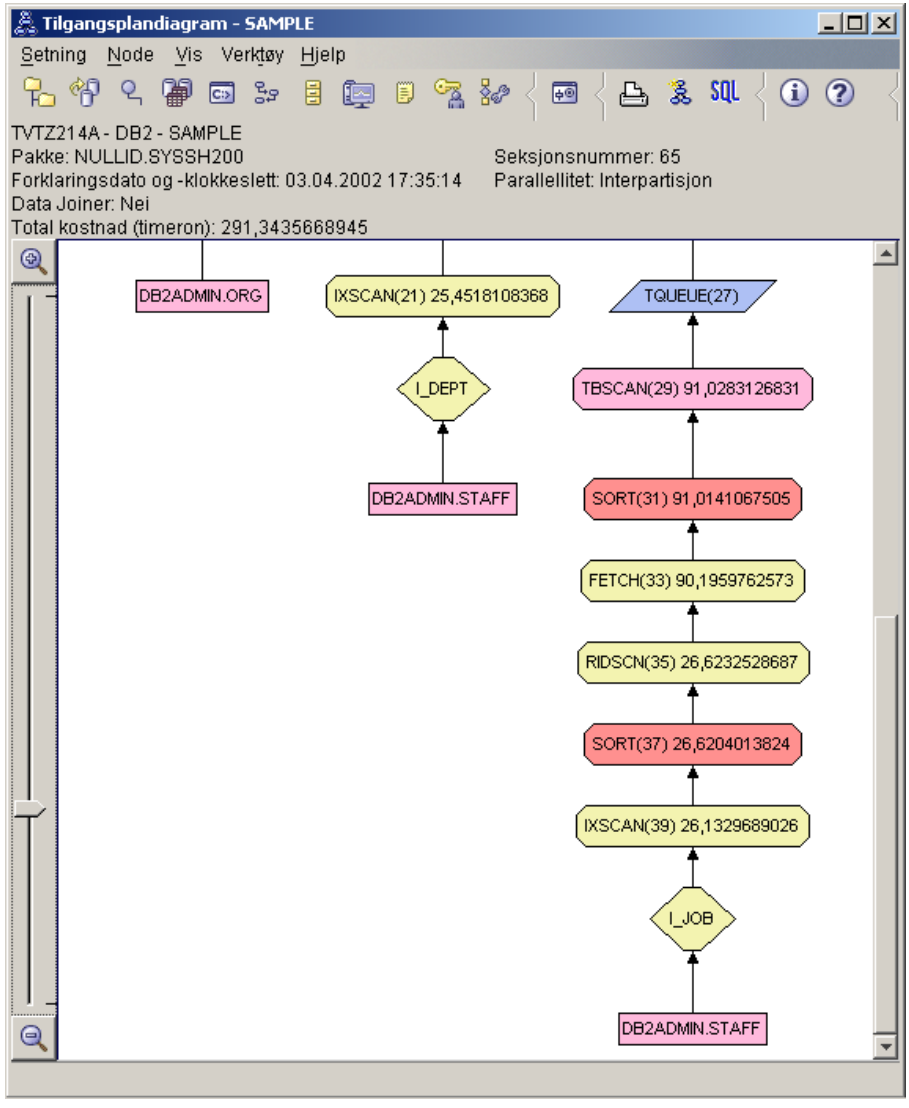
Hvis du svarer på de følgende spørsmålene, vil det hjelpe deg med å forstå hvordan du kan forbedre spørringen.

1. Hva er endret i denne tilgangsplanen på grunn av at det ble opprettet tilleggsindekser?

   I den midterste delen av tilgangsplandiagrammet kan du legge merke til at for ORG-tabellen er det forrige tabellsøket endret til et indekssøk (IXSCAN 7). Siden kolonnen AVDELING er føyd til indeksen for ORG-tabellen, kunne optimalisatoren forbedre tilgangen for tabellsøket.



I den nederste delen av tilgangsplandiagrammet kan du legge merke til at for STAB-tabellen er det forrige indekssøket og den forrige hentingen endret til bare et indekssøk (IXSCAN 39). Siden indeksen JOBB ble opprettet for STAB-tabellen, kunne optimalisatoren fjerne den ekstra tilgangen for hentingen.



2. Hvor effektiv er denne tilgangsplanen?

   Denne tilgangsplanen er mer kostnadseffektiv enn planen i det forrige eksempelet. Den kumulative kostnaden er redusert fra rundt 753 timeron i spørring 3 til rundt 288 timeron i spørring 4.

Det neste

Administration Guide inneholder detaljert informasjon om andre trinn du kan bruke for å bedre ytelsen. Du kan gå tilbake til Visuell forklaring for å vurdere virkningen av handlingene.

Tillegg A. Visuell forklaring - begreper

Tilgangsplan

Bestemte data er nødvendige for å behandle en forklarlig SQL-setning. En *tilgangsplan* oppgir en operasjonsrekkefølgen for tilgang til disse dataene. Tilgangsplanen bruker du til å se på statistikk for valgte tabeller, indekser eller kolonner, operatoregenskaper, generell informasjon som for eksempel statistikk for tabellplasser og funksjoner, og konfigurasjonsparametere som er relevante for optimalisering. Med Visuell forklaring kan du se tilgangsplanen for en SQL-setning i grafisk form.

Optimalisatoren oppretter en tilgangsplan hver gang en forklarlig SQL-setning blir kompilert. Dette skjer under klargjørings- og tilkoblingstiden for statiske setninger og under kjøring for dynamiske setninger.

Det er viktig å forstå at en tilgangsplan er et *estimat* basert på den informasjonen som er tilgjengelig. Optimalisatoren baserer sine beregninger blant annet på denne typen opplysninger:

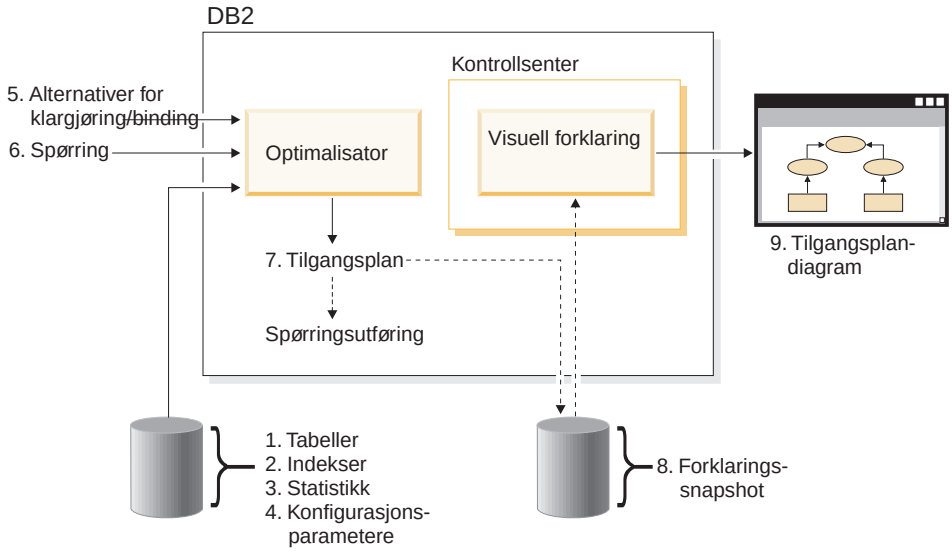
- Statistikk i systemkatalogtabeller (hvis statistikken ikke er oppdatert, kan du oppdatere den med runstats-kommandoen.)
- Konfigurasjonsparametere
- Bindingsalternativer
- Klasse for optimalisering av spørring

Kostnadsinformasjonen som er tilknyttet en tilgangsplan, er optimalisatorens *beste estimat* av ressursbruken for en spørring. Tiden som faktisk medgår til en spørring, kan variere avhengig av faktorer som ligger utenfor området til DB2 (for eksempel hvor mange andre applikasjoner som kjøres samtidig). Tiden som faktisk medgår kan måles mens spørringen kjøres, ved hjelp av ytelsesovervåking.

Tilgangsplandiagram

Visuell forklaring bruker opplysninger fra en rekke kilder for å opprette et tilgangsplandiagram, slik som vist i illustrasjonen nedenfor. Optimalisatoren velger en tilgangsplan basert på forskjellige inndata, og Visuell forklaring viser det i et *tilgangsplandiagram*. Nodene i diagrammet representerer tabeller og indekser og hver operasjon på dem. Linjene mellom nodene representerer

dataflyten.



Listen over oppgaver nedenfor samsvarer med oppgavene som blir vist i illustrasjonen ovenfor. (Brutte linjer viser trinn som kreves for Visuell forklaring.)

1. Juster tabelloppsettet og omorganiser tabelldataene.
2. Opprett riktige indekser.
3. Bruk runstats-kommandoen til å forsyne optimalisatoren med gjeldende statistikk.
4. Velg riktige konfigurasjonsparametere.
5. Velg riktige bindingsalternativer.
6. Utform spørringer som bare henter de ønskede dataene .
7. Opprette en tilgangsplan.
8. Opprett forklarings-snapshot.
9. Vis og bruk et tilgangsplandiagram.

Hvis du for eksempel skal bruke Visuell forklaring, må du først oppdatere gjeldende statistikk med kommandoen **runstats** på tabellene og indeksene som brukes av setningen. Denne statistikken, konfigurasjonsparameterne, bindingsalternativene og selve spørringen bruker optimalisatoren til å lage en tilgangsplan og et forklarings-snapshot når pakken blir bundet. Visuell forklaring bruker dette forklarings-snapshotet til å få frem tilgangsplandiagrammet for setningen.

Tilgangsplandiagram

Tilgangsplandiagrammet viser en oversikt over *noder*. Disse nodene representerer:

- Tabeller, vist som rektangler
- Indekser, vist som diamanter
- Operatorer, vist som åttekanter (8 sider). TQUEUE-operatorer, vist som parallellogrammer
- Tabellfunksjoner, vist som sekskanter (6 sider)

Gruppering

Oppdateringer kan over tid forårsake endring av plasseringen til rader på datasider. Dette forringer graden av *gruppering* som finnes mellom en indeks og datasidene. Når du omorganiserer en tabell med hensyn til en valgt indeks, blir dataene omgruppert. En gruppeindeks er mest nyttig når du arbeider med kolonner som har områdepredikater, fordi den muliggjør en bedre sekvensiell tilgang til data i basistabellen. Dette resulterer i at færre sider blir hentet inn, siden like verdier står på samme dataside.

Vanligvis er det bare en indeks i en tabell som kan ha en høy grupperingsgrad.

Hvis du vil kontrollere grupperingsgraden for en indeks, dobbeltklikker du på noden til indeksen for å få frem vinduet Indeksstatistikk. Verdien for grupperingsforholdstall eller grupperingsfaktor vises i dette vinduet. Hvis verdien er lav, bør du vurdere å omorganisere dataene i tabellen.

Hvis du ønsker flere opplysninger, kan du slå opp i delen om omorganisering av tabelldata i *Administration Guide*.

Container

En *container* er en fysisk lagerplass for data. Den er tilknyttet en tabellplass og kan være en fil, en katalog eller en enhet.

Kostnad

I forbindelse med Visuell forklaring er *Kostnad* den antatte totale bruk av ressurser som er nødvendig for å utføre tilgangsplanen for en setning (eller elementene i en setning). Kostnad er utledet fra en kombinasjon av CPU-kostnad (i antall instruksjoner) og I/U (i antall søk og sideoverføringer).

Kostnadsenheten er *timeron*. En timeron er ikke det samme som faktisk medgått tid, men gir et grovt relativt overslag over ressursene (kostnadene) som kreves av databasesystemet for å kunne utføre to planer for den samme spørringen.

Kostnaden som vises i hver operatornode til et tilgangsplandiagram, er den kumulative kostnaden, fra tilgangsplanen startet utføringen til og med utføringen av den bestemte operatoren. Faktorer som arbeidsbelastning på systemet eller kostnadene ved å returnere rader med data til brukeren, gjenspeiles ikke av begrepet kostnad.

Pekerblokking

Pekerblokking er en teknikk som reduserer overflødig behandling ved at databasesystemet henter tilbake en *blokk* med rader i en enkelt operasjon. Radene blir lagret i en hurtigbuffer mens de blir behandlet. Hurtigbufferen blir tildelt når en applikasjon sender en OPEN CURSOR-forespørsel, og tildelingen opphører når pekeren lukkes. Når alle radene er behandlet, blir en annen blokk med rader hentet.

Bruk BLOCKING-alternativet i **PREP**- eller **BIND**-kommandoer sammen med følgende parametere for å oppgi hvilken type pekerblokking du vil bruke:

UNAMBIG
Bare utvetydige pekere blir blokket (standard).

ALL Både tvetydige og utvetydige pekere blir blokket.

NO Pekere blir ikke blokket.

Hvis du ønsker flere opplysninger, kan du slå opp i avsnittet om pekerblokking i *Administration Guide*.

DMS-tabellplass (databasestyrt plass)

Det er to typer tabellplasser som kan finnes i en database: databasestyrt plass (DMS) og systemstyrt plass (SMS).

DMS-tabellplasser styres av databasesystemet og er utformet og justert for å tilfredsstille databasesystemets krav.

Definisjonen av DMS-tabellplassen inneholder en liste over filer (eller enheter) som lagrer databasedata i DMS-tabellplassformat.

Du kan føye forhåndstildelte filer (eller enheter) til en eksisterende DMS-tabellplass for å øke lagringskapasiteten. Databasesystemet balanserer automatisk eksisterende data i alle containere som hører til den tabellplassen.

DMS- og SMS-tabellplasser kan eksistere sammen i en database.

Dynamisk SQL

Dynamiske SQL-setninger er SQL-setninger som blir klargjort og utført i et applikasjonsprogram mens programmet kjører. I dynamiske SQL-setninger

- utsteder du SQL-setningen interaktivt ved hjelp av CLI eller CLP
- er SQL-kilden inneholdt i vertsspråkvariabler som er innfelt i et applikasjonsprogram

Når DB2 kjører en dynamisk SQL-setning, oppretter den en tilgangsplan som er basert på gjeldende katalogstatistikk og konfigurasjonsparametere. Denne tilgangsplanen kan endre seg fra en utføring av setningene i et applikasjonsprogram til en annen.

Alternativet til dynamisk SQL er statisk SQL.

Forklarings-snapshot

Med Visuell forklaring kan du undersøke innholdet i et forklarings-snapshot.

Et *forklarings-snapshot* er komprimert informasjon som blir registrert når en SQL-setning er forklart. Det lagres som et stort binærobjekt (BLOB) i tabellen EXPLAIN_STATEMENT, og inneholder følgende informasjon:

- Internfremstilling av tilgangsplanen, samt operatorene, tabellene og indeksene som er brukt
- Beslutningskriterier som optimalisatoren har brukt, samt statistikk for databaseobjekter og kumulative kostnader for hver operasjon

Et forklarings-snapshot kreves hvis du vil se en grafisk fremstilling av tilgangsplanen til en SQL-setning. Slik forsikrer du deg om at et forklarings-snapshot blir opprettet:

1. Databasesystemet må ha forklaringstabeller som kan lagre forklarings-snapshotene. Informasjon om hvordan du oppretter disse tabellene se Opprette forklaringstabeller i hjelpen.
2. Når du binder eller klargjør en pakke som inneholder statiske SQL-setninger, må du definere EXPLSNAP til ALL eller YES. Da får du et forklarings-snapshot for alle forklarlige SQL-setninger i pakken. Du finner mer informasjon om kommandoene **BIND** og **PREP** i *Command Reference*.
3. For dynamiske SQL-setninger setter du EXPLSNAP-alternativet til ALL når du binder applikasjonen som utsteder dem, eller du kan sette spesialregisteret CURRENT EXPLAIN SNAPSHOT til YES eller EXPLAIN før du utsteder dem interaktivt. Hvis du ønsker flere opplysninger, slår du opp i avsnittet om gjeldende forklarings-snapshot i *SQL Reference*.

Forklarlig setning

En *forklarlig setning* er en SQL-setning som en forklaringsoperasjon kan utføres på.

Forklarlige SQL-setninger er:
- SELECT
- INSERT
- UPDATE
- DELETE
- VALUES

Forklart setning

En *forklart setning* er en SQL-setning som en forklaringsoperasjon er utført på. Forklarte setninger vises i vinduet Historikk over forklarte setninger.

Operand

En operand er en enhet som det blir utført en operasjon på. En tabell eller en indeks er for eksempel en operand av forskjellige operatorer som for eksempel TBSCAN og IXSCAN.

Operator

En *operator* er enten en handling som må utføres på data, eller utdataene fra en tabell eller en indeks, når tilgangsplanen til en SQL-setning er utført.

Følgende operatorer kan vises i tilgangsplandiagrammet:

DELETE
Sletter rader i en tabell.

EISCAN
Skanner en brukerdefinert indeks for å produsere en redusert strøm av rader.

FETCH
Henter kolonner fra en tabell med en bestemt postidentifikator.

FILTER
Filtrerer data ved å utføre ett eller flere predikater på dem.

GRPBY
Grupperer rader etter vanlige verdier for definerte kolonner eller funksjoner, og evaluerer definisjonsfunksjoner.

HSJOIN
Representerer en nøkkelkombinasjon (hash join) der to eller flere tabeller indekseres i kombinasjonskolonnene.

INSERT
Setter inn rader i en tabell.

IXAND
Føyer sammen radidentifikatorer (RIDer) fra to eller flere indekssøk.

IXSCAN
Blar gjennom en indeks for en tabell med valgfrie start- og stoppvilkår, og lager sorterte rader.

MSJOIN
Representerer en samkjøringskombinasjon (merge join) der både eksterne og interne tabeller må være i kombinasjonspredikat-rekkefølge.

NLJOIN
Representerer en nestet sløyfekombinasjon som går inn i en intern tabell en gang for hver rad i den eksterne tabellen.

RETURN
Representerer returnering av dataene fra spørringen til brukeren.

RIDSCN
Blar gjennom en liste over radidentifikatorer (RID) som ble funnet i en eller flere indekser.

SHIP Henter data fra en fjernliggende databasekilde. Brukes i det forente systemet.

SORT Sorterer rader i den rekkefølgen kolonnene er oppgitt i, og fjerner poster som er like (valgfritt).

TBSCAN
Henter rader ved å lese alle nødvendige data direkte fra datasidene.

TEMP Lagrer data i en midlertidig tabell som skal leses tilbake (muligens flere ganger).

TQUEUE
Overfører tabelldata mellom databaseagenter.

UNION
Slår sammen rader fra flere tabeller.

UNIQUE
Fjerner rader som har like verdier for oppgitte kolonner.

UPDATE
Oppdaterer rader i en tabell.

CMPEXP

Operatornavn: CMPEXP

Beskrivelse: Beregningen av uttrykk som er nødvendig for midlertidige eller endelige resultater.

(Denne operatoren er bare for feilsøkingsmodus.)

DELETE

Operatornavn: DELETE

Beskrivelse: Sletter rader fra en tabell.

Denne operatoren representerer en nødvendig operasjon. Hensikten er å redusere tilgangsplankostnadene ved å fokusere på andre operatorer (som søk og kombinasjoner) som definerer settet med rader som skal slettes.

Gjør slik:

- Hvis du sletter alle radene fra en tabell, bør du bruke kommandoen DROP TABLE eller **LOAD REPLACE**.

EISCAN

Operatornavn: EISCAN

Beskrivelse: Denne operatoren søker igjennom en brukerdefinert indeks for å lage en redusert datastrøm av rader. Søket bruker de mange start/stop-betingelsene fra Range producer-funksjonen som er oppgitt av brukeren.

Denne operasjonen utføres for å redusere settet med kvalifiserende rader før basistabellen lastes (basert på predikater).

Gjør slik:

- Over tid kan databaseoppdateringer resultere i at en indeks blir fragmentert, noe som igjen fører til at du får flere indekssider enn nødvendig. Du kan rette dette ved å slette eller opprette indeksen på nytt, eller ved å reorganisere indeksen.
- Hvis statistikken ikke er oppdatert, kan du oppdatere den med runstats-kommandoen.

FETCH

Operatornavn: FETCH

Beskrivelse: Henter kolonner fra en tabell ved hjelp av en bestemt radidentifikator (RID).

Gjør slik:

- Ta med kolonnene som er hentet i indeksnøkkelen, slik at du slipper å bruke datasidene.
- Finn indeksen som er knyttet til hentingen, og dobbeltklikk på noden for å få frem statistikkvinduet. Kontroller at graden av gruppering er høy for indeksen.
- Øk bufferstørrelsen hvis inndataene/utdataene (I/U) som ble hentet, overskriver antall sider i tabellen.
- Hvis statistikken ikke er oppdatert, kan du oppdatere den med runstats-kommandoen.

  Statistikken for kontrollverdi og hyppig verdi gir opplysninger om selektivitet for predikater, som bestemmer når det skal søkes i en indeks og ikke i en tabell. Når du skal oppdatere denne statistikken, må du bruke kommandoen **RUNSTATS** på en tabell med WITH DISTRIBUTION-leddet.

FILTER

Operatornavn: FILTER

Beskrivelse: Bruk av restverdipredikater slik at data blir filtrert etter de kriteriene som predikatene tilsier.

Gjør slik:

- Kontroller at du har brukt predikater som bare henter de dataene du trenger. Kontroller for eksempel at selektivitetsverdien for predikatene representerer den delen av tabellen som du vil ha returnert.
- Kontroller at optimaliseringsklassen er minimum 3 slik at optimalisatoren bruker en kombinering (JOIN) i stedet for en delspørring. Hvis det ikke er mulig, kan du prøve å skrive om SQL-spørringen på nytt for hånd, slik at du eliminerer delspørringen. Du finner et eksempel hvis du slår opp i kapittelet som omhandler omskriving av spørringer med SQL-kompilatoren, i *Administration Guide*.

GENROW

Operatornavn: GENROW

Beskrivelse: En innebygd funksjon som genererer en tabell med rader uten å bruke noen form for inndata fra tabeller, indekser eller operatorer.

Optimalisatoren kan bruke GENROW når den skal lage rader med data (for eksempel for en INSERT-setning eller for noen IN-lister som er overført til kombinasjoner).

Når du vil se på statistikken som er beregnet for tabeller som GENROW-funksjonen har laget, dobbeltklikker du på noden.

GRPBY

Operatornavn: GRPBY

Beskrivelse: Gruppering av rader i henhold til vanlige verdier for definerte kolonner eller funksjoner. Operasjonen er nødvendig for å lage en gruppe verdier, eller for å evaluere definerte funksjoner.

Selv om det ikke er oppgitt en GROUP BY-kolonne, kan GRPBY-operatoren brukes hvis det finnes samlingsfunksjoner i SELECT-listen som oppgir at hele tabellen er behandlet som en enkelt gruppe under samlingen.

Gjør slik:

- Denne operatoren representerer en nødvendig operasjon. Hensikten er å redusere tilgangsplankostnadene ved å fokusere på andre operatorer (som søk og kombinasjoner) som definerer hvilket sett med rader som skal grupperes.
- Hvis du vil forbedre ytelsen til en SELECT-setning som inneholder en enkelt samlefunksjon, men ingen GROUP BY-ledd, kan du forsøke følgende:
  - For en MIN(C)-samlefunksjon lager du en stigende indeks på C.
  - For en MAX(C)-samlefunksjon lager du en synkende indeks på C.

HSJOIN

Operatornavn: HSJOIN

Beskrivelse: En nøkkelkombinasjon (hash join) der kvalifiserte rader fra tabeller indekseres slik at direkte kombinering uten forhåndssortering av innholdet er mulig.

En kombinasjon er nødvendig når et FROM-ledd viser til flere tabeller. En nøkkelkombinasjon er mulig hvis det finnes et kombinasjonspredikat som sammenligner kolonner fra to forskjellige tabeller. Kombinasjonspredikatene må ha nøyaktig samme datatype. Nøkkelkombinasjoner kan også komme fra en delspørring som er skrevet om, som for eksempel NLJOIN.

Nøkkelkombinasjoner krever ikke at tabellene sorteres på forhånd. Kombineringen utføres ved å søke gjennom den indre tabellen og lage en

oppslagstabell ved å indeksere kombinasjonskolonneverdiene. Deretter leses den ytre tabellen, kombinasjonskolonneverdiene blir indeksert (hashed) og oppslagstabellen for den interne tabellen blir kontrollert.

Hvis du ønsker flere opplysninger, kan du slå opp i kapittelet om kombinasjonsbegreper i *Administration Guide*.

Gjør slik:

- Bruk lokale predikater (det vil si predikater som viser til en tabell) når du skal redusere antall rader som skal kombineres.
- Øke størrelsen til minneområdet for sortering slik at det blir stort nok til å romme oppslagstabellen.
- Hvis statistikken ikke er oppdatert, kan du oppdatere den med runstats-kommandoen.

INSERT

Operatornavn: INSERT

Beskrivelse: Setter rader inn i en tabell.

Denne operatoren representerer en nødvendig operasjon. Hensikten er å redusere tilgangsplankostnadene ved å fokusere på andre operatorer (som søk og kombinasjoner) som definerer hvilket sett med rader som skal settes inn.

IXAND

Operatornavn: IXAND

Beskrivelse: Føyer sammen resultatene av flere indekssøk ved hjelp av dynamiske bitmønsterteknikker. Operatoren gjør det mulig for predikater det er utført AND på, å bli brukt på flere indekser for å redusere tilgang til underliggende tabeller til et minimum.

Denne operatoren utføres for å

- begrense settet med rader før basistabellen lastes
- føye sammen (AND) predikater som er brukt i flere indekser
- føye sammen (AND) resultatene fra delkombinasjoner, som er brukt i stjernekombinasjoner

Gjør slik:

- Over tid kan databaseoppdateringer resultere i at en indeks blir fragmentert, noe som igjen fører til at du får flere indekssider enn nødvendig. Du kan rette dette ved å slette eller opprette indeksen på nytt, eller ved å reorganisere indeksen.

- Hvis statistikken ikke er oppdatert, kan du oppdatere den med runstats-kommandoen.
- Vanligvis er indekssøk mest effektivt når du bare har få kvalifiserte rader. Optimalisatoren bruker statistikken som er tilgjengelig for kolonnene det er referert til i predikater, når den skal beregne antall kvalifiserte rader. Hvis enkelte verdier opptrer oftere enn andre, er det viktig at du ber om fordelingsstatistikk. Det gjør du ved å bruke WITH DISTRIBUTION-leddet sammen med kommandoen **RUNSTATS**. Hvis du bruker den ujevne fordelingsstatistikken, kan optimalisatoren skille mellom verdier som opptrer ofte, og verdier som opptrer sjelden.
- IXAND kan best utnytte indekser med enkeltkolonner, siden start- og stoppnøklene er helt nødvendig ved bruk av IXAND.
- Når det gjelder stjernekombinasjoner, bør du opprette indekser med enkeltkolonner for hver av de mest selektive kolonnene i produkttabellen og de beslektede formattabellene.

IXSCAN

Operatornavn: IXSCAN

Beskrivelse: Søker i en indeks for å generere en redusert datastrøm av rader. Søkingen kan bruke valgfrie start- og stoppvilkår, eller den kan brukes på indekserbare predikater som viser til kolonner i indeksen.

Denne operasjonen utføres for å redusere settet med kvalifiserende rader før basistabellen lastes (basert på predikater).

Hvis du ønsker flere opplysninger, kan du slå opp i kapittelet om indekssøk i *Administration Guide*.

Gjør slik:

- Over tid kan databaseoppdateringer resultere i at en indeks blir fragmentert, noe som igjen fører til at du får flere indekssider enn nødvendig. Du kan rette dette ved å slette eller opprette indeksen på nytt, eller ved å reorganisere indeksen.
- Når du bruker to eller flere tabeller, får du mer effektiv tilgang via en indeks til den interne tabellen hvis du bruker en indeks for kombinasjonskolonnen til den eksterne tabellen.

  Hvis du ønsker flere opplysninger om indekser, kan du se i hjelpen for Visuell forklaring.
- Hvis statistikken ikke er oppdatert nylig, må du oppdatere den ved hjelp av runstats-kommandoen.
- Vanligvis er indekssøk mest effektivt når du bare har få kvalifiserte rader. Optimalisatoren bruker statistikken som er tilgjengelig for kolonnene det er

referert til i predikater, når den skal beregne antall kvalifiserte rader. Hvis enkelte verdier opptrer oftere enn andre, er det viktig at du ber om fordelingsstatistikk. Det gjør du ved å bruke WITH DISTRIBUTION-leddet sammen med kommandoen **RUNSTATS**. Hvis du bruker den ujevne fordelingsstatistikken, kan optimalisatoren skille mellom verdier som opptrer ofte, og verdier som opptrer sjelden.

MSJOIN

Operatornavn: MSJOIN

Beskrivelse: En samkjøringskombinasjon der de kvalifiserte radene fra både eksterne og interne tabeller må være i samme rekkefølge som i det kombinerte predikatet. En samkjøringskombinasjon kalles også en *kombinasjon av samkjørte søk* eller en *kombinasjon av sorterte søk*.

En kombinasjon er nødvendig når et FROM-ledd viser til flere tabeller. En samkjøringskombinasjon er alltid mulig hvis det finnes et kombinasjonspredikat som utlikner kolonnene fra to forskjellige tabeller. Det kan også komme fra en delspørring som er skrevet om.

En samkjøringskombinasjon krever at inndataene i kombinasjonskolonner er sortert, fordi tabellene bare blir bladd gjennom en gang. Du får tilgang til sorterte inndata ved hjelp av en indeks eller en sortert tabell.

Hvis du ønsker flere opplysninger, kan du slå opp i kapittelet om kombinasjonsbegreper i *Administration Guide*.

Gjør slik:

- Bruk lokale predikater (det vil si predikater som viser til en tabell) når du skal redusere antall rader som skal kombineres.

  Du finner flere retningslinjer om indekser under Opprette riktige indekser i hjelpen til Visuell forklaring.
- Hvis statistikken ikke er oppdatert, kan du oppdatere den med runstats-kommandoen.

NLJOIN

Operatornavn: NLJOIN

Beskrivelse: En nestet sløyfekombinasjon som blar gjennom (vanligvis med et indekssøk) den interne tabellen en gang for hver rad i den eksterne tabellen.

En kombinasjon er nødvendig når et FROM-ledd viser til flere tabeller. En nestet sløyfekombinasjon krever ikke et kombinasjonspredikat, men utføringen blir bedre hvis du bruker det.

En nestet sløyfekombinasjon blir utført
- enten ved at programmet blar gjennom den interne tabellen for alle rader som er brukt i den eksterne tabellen
- eller ved at programmet slår opp i en indeks for den interne tabellen for alle rader som er brukt i den eksterne tabellen

Hvis du ønsker flere opplysninger, kan du slå opp i kapittelet om kombinasjonsbegreper i *Administration Guide*.

Gjør slik:

- En nestet sløyfekombinasjon blir sannsynligvis mer effektiv hvis det finnes en indeks på kombinasjonspredikatkolonnene i den interne tabellen (tabellen som vises til høyre for NLJOIN-operatoren). Kontroller om den interne tabellen er en TBSCAN, og ikke en IXSCAN. Hvis den er en TBSCAN, bør du vurdere å tilføye en indeks på kombinasjonskolonnene til tabellen.

  En annen (mindre viktig) måte å gjøre kombinasjonen mer effektiv på, er at du oppretter en indeks på kombinasjonskolonnene i den eksterne tabellen, slik at tabellen blir sortert.

  Du finner flere retningslinjer om indekser under Opprette riktige indekser i hjelpen til Visuell forklaring.
- Hvis statistikken ikke er oppdatert, kan du oppdatere den med runstats-kommandoen.

Annen informasjon:

- Stjernekombinasjon.

PIPE

Operatornavn: PIPE

Beskrivelse: Overføring av rader til andre operatorer uten å endre dem.

(Denne operatoren er bare for feilsøkingsmodus.)

RETURN

Operatornavn: RETURN

Beskrivelse: Returnering av data fra en spørring til brukeren. Dette er den siste operatoren i tilgangsplandiagrammet og viser totalt akkumulerte verdier og kostnader for tilgangsplanen.

Denne operatoren representerer en nødvendig operasjon.

Gjør slik:

- Kontroller at du har brukt predikater som bare henter de dataene du trenger. Kontroller for eksempel at selektivitetsverdien for predikatene representerer den delen av tabellen som du vil ha returnert.

RIDSCN

Operatornavn: RIDSCN

Beskrivelse: Søking i en liste med radidentifikatorer (RIDer) som ble funnet i en eller flere indekser.

Optimalisatoren tar hensyn til denne operatoren under disse forholdene:

- Når predikater er knyttet sammen med OR-nøkkelord, eller hvis det finnes et IN-predikat. Du kan bruke OR i indekser (såkalt "index ORing"), og da blir resultater fra flere indekstilganger mot samme tabell kombinert.
- Når det er en fordel å bruke forhåndshenting fra liste for en enkelt indekstilgang, fordi I/U-operasjonene blir mer effektive hvis du sorterer radidentifikatorene før du går inn på basisradene.

RQUERY

Operatornavn: SHIP

Beskrivelse: En operator som brukes i det forente systemet til å hente data fra en fjerntliggende datakilde. Denne operatoren blir vurdert av optimalisatoren når: En SHIP-operator sender en SQL SELECT-setning til en fjerntliggende datakilde for å hente søkeresultatet. SELECT-setningen blir generert ved hjelp av den SQL-dialekten som støttes av datakilden, og kan inneholde en hvilken som helst gyldig spørring, slik den er tillatt av datakilden.

Forslag til ytelse: Slå opp i kapittel 4 i boken Administration Guide Vol 2, Federated Database Query and Network Tuning Information.

SORT

Operatornavn: SORT

Beskrivelse: Sortering av radene i en tabell i samme rekkefølge som en eller flere av kolonnene i tabellen. Du kan velge om du vil slette poster som er like.

Sortering er nødvendig når det ikke finnes noen indeks som tilfredsstiller rekkefølgen du bad om, eller fordi sortering er mindre kostnadskrevende enn indekssøk. Sortering utføres vanligvis som en siste operasjon etter at de nødvendige radene er hentet, eller for å sortere data før det blir foretatt en kombinering (JOIN) eller en gruppering etter (GROUPBY).

Hvis det er mange rader, eller hvis de sorterte dataene ikke videresendes, krever operasjonen at du lager midlertidige tabeller, som er mer kostnadskrevende.

Hvis du ønsker flere opplysninger om sorteringer, kan du slå opp i *Administration Guide*.

Gjør slik:

- Vurder om du skal tilføye en indeks på sorteringskolonnene.

  Du finner flere retningslinjer om indekser under Opprette riktige indekser i hjelpen til Visuell forklaring.
- Kontroller at du har brukt predikater som bare henter de dataene du trenger. Kontroller for eksempel at selektivitetsverdien for predikatene representerer den delen av tabellen som du vil ha returnert.
- Kontroller at størrelsen på forhåndshentingen for den midlertidige systemtabellplassen er stor nok, det vil si at den ikke er begrenset av I/U. (For å kontrollere dette må du velge **Setning->Vis statistikk->Tabellplasser**.)
- Hvis det ofte er behov for store sorteringer, bør du vurdere om du skal øke verdiene til disse konfigurasjonsparameterne:
  - Minneområde for sortering (sortheap). Hvis du vil endre denne parameteren, klikker du med høyre museknapp på databasen i Kontrollsenter og velger deretter *Konfigurer* fra objektmenyen. Velg **Ytelse** fra notisboken som kommer frem.
  - Sorteringsterskel for minneområde (sheapthres). Hvis du vil endre denne parameteren, klikker du med høyre museknapp på databaseforekomsten i Kontrollsenter og velger *Konfigurer* fra objektmenyen. Velg **Ytelse** fra notisboken som kommer frem.
- Hvis statistikken ikke er oppdatert, kan du oppdatere den med runstats-kommandoen.

TBSCAN

Operatornavn: TBSCAN

Beskrivelse: Et tabellsøk (relasjonssøk) som henter rader ved å lese alle nødvendige data direkte fra datasidene.

Optimalisatoren velger denne typen søk fremfor indekssøk når:
- verdiområdet som blir bladd igjennom, forekommer ofte (det vil si at mesteparten av tabellen må være i bruk)
- tabellen er liten
- det er lav indeksgruppering
- det ikke finnes noen indeks

Hvis du ønsker flere opplysninger om indekssøk, kan du slå opp i *Administration Guide*.

Gjør slik:

- Et indekssøk er mer effektivt enn et tabellsøk hvis tabellen er stor, og hvis de fleste av radene i tabellen ikke er brukt. For å øke muligheten for at optimalisatoren skal bruke et indekssøk i denne situasjonen, bør du vurdere å tilføye indekser på kolonner som det finnes selektive predikater for.

  Du finner flere retningslinjer om indekser under Opprette riktige indekser i hjelpen til Visuell forklaring.
- Hvis det allerede finnes en indeks, men den ikke ble brukt, må du kontrollere at det finnes selektive predikater for hver av de første kolonnene indeksen. Hvis det ikke finnes slike predikater, må du kontrollere at graden av gruppering er høy for indeksen. (Hvis du vil se denne statistikken, åpner du vinduet Tabellstatistikk for tabellen under sorteringen og klikker på skjermtasten *Indekser* for å få frem vinduet Indeksstatistikk.)
- Kontroller at størrelsen på forhåndshentingen for tabellplassen er stor nok, det vil si at den ikke er begrenset av I/U. (For å kontrollere dette må du velge **Setning->Vis statistikk->Tabellplasser**.)

  Hvis du vil ha flere opplysninger, kan du se under delen som omhandler forhåndshenting av data inn i bufferområdet i *Administration Guide*.
- Hvis statistikken ikke er oppdatert, kan du oppdatere den med runstats-kommandoen.

  Kontrollverdien og hyppig verdistatistikk gir informasjon om selektivitet for predikater. Denne statistikken blir for eksempel brukt til avgjøre når indekssøk foretrekkes fremfor tabellsøk. Når du skal oppdatere disse verdiene, bruker du kommandoen **RUNSTATS** på en tabell med WITH DISTRIBUTION-leddet.

TEMP

Operatornavn: TEMP

Beskrivelse: Lagring av data i en midlertidig tabell som skal leses tilbake av en annen operator (muligens flere ganger). Tabellen blir fjernet når SQL-setningen er behandlet, hvis ikke før.

Denne operatoren er nødvendig for å evaluere delspørringer eller for å lagre midlertidige resultater. I enkelte tilfeller (for eksempel når setningen kan oppdateres) kan den være obligatorisk.

TQUEUE

Operatornavn: TQUEUE

Beskrivelse: En tabellkø som brukes til å sende tabelldata fra en databaseagent til en annen når det er flere databaseagenter som behandler en spørring. Flere databaseagenter brukes til å behandle en spørring når parallellitet er involvert.

Tabellkøtyper:

- **Lokal:** Tabellkøen brukes til å sende data mellom databaseagenter innenfor en enkelt node. En lokal tabellkø brukes ved intrapartisjonparallellitet.
- **Ikke lokal:** Tabellkøen brukes til å sende data mellom databaseagenter på forskjellige noder.

UNION

Operatornavn: UNION

Beskrivelse: Sammenkjeding av strømmer av rader fra flere tabeller.

Denne operatoren representerer en nødvendig operasjon. Hensikten er å redusere tilgangsplankostnadene ved å fokusere på andre operatorer (for eksempel søk og kombinasjoner) som definerer settet med rader som skal slås sammen.

UNIQUE

Operatornavn: UNIQUE

Beskrivelse: Sletting av rader som har like verdier i oppgitte kolonner.

Gjør slik:

- Denne operatoren er ikke nødvendig hvis det finnes en entydig indeks for de riktige kolonnene.

  Du finner flere retningslinjer om indekser under Opprette riktige indekser i hjelpen til Visuell forklaring.

UPDATE

Operatornavn: UPDATE

Beskrivelse: Oppdatering av data i radene i en tabell.

Denne operatoren representerer en nødvendig operasjon. Hensikten er å redusere tilgangsplankostnadene ved å fokusere på andre operatorer (for eksempel søk og kombinasjoner) som definerer settet med rader som skal oppdateres.

Optimalisator

Optimalisatoren er den komponenten i SQL-kompilatoren som velger en tilgangsplan for datahåndteringsspråk (DML) av en SQL-setning. Det gjør den ved å skissere utføringskostnadene til mange alternative tilgangsplaner, og så velge den med den antatt laveste kostnaden.

Pakke

En *pakke* er et objekt som ligger lagret i databasen som inneholder den informasjonen som trengs for å behandle SQL-setningene som er tilknyttet en kildefil til et applikasjonsprogram. Den kan genereres på to måter:

- Forkompilering av en kildefil med kommandoen **PREP**
- Binding av en bindingsfil som ble generert av forkompilatoren med kommandoen **BIND**

Predikat

Et *predikat* er et element i en søkebetingelse som uttrykker eller forutsetter en sammenlikningsoperasjon. Predikater finnes i ledd som starter med WHERE eller HAVING.

Eksempel: I SQL-setningen

```
SELECT * FROM SAMPLE
  WHERE NAVN = 'NILSEN' AND
  AVD = 895 AND ÅR > 5
```

er dette predikater: NAVN = 'NILSEN', AVD = 895, og ÅR > 5.

Predikater faller innunder en av følgende kategorier, sortert etter effektivitet (de mest effektive først):

1. Hakeparenteser for start og slutt på betingelser (innsnevrer et indekssøk). (Disse betingelsene kalles også predikater for avgrensning av verdiområde.)

2. Indekssidepredikater (også kalt indekssøkbare predikater) kan evalueres fra en indeks fordi kolonnene som er involvert i predikatet, er del av indeksnøkkelen.
3. Datasidepredikater (også kalt datasøkbare predikater) kan ikke evalueres fra en indeks, men kan evalueres mens rader forblir i bufferen.
4. Restverdipredikater krever vanligvis I/U utover det å få tilgang til en basistabell, og må tas i bruk etter at data er kopiert fra buffersiden. De omfatter predikater som inneholder delspørringer, eller predikater som leser LONG VARCHAR- eller LOB-data som er lagret i filer utenfor tabellen.

Når du lager predikater, bør du forsøke å oppnå høyest mulig selektivitet, slik at færrest mulig rader blir returnert.

Følgende typer predikater er de mest effektive, og også de som er mest brukt:

- Et *enkelt likhetskombinasjonspredikat* er nødvendig i en samkjøringskombinasjon. Det har formatet TABELL1.kolonne = TABELL2.kolonne, og tillater at kolonner i to forskjellige tabeller blir gjort like, slik at tabellene kan kombineres.
- Et *lokalt predikat* brukes bare på en tabell.

Hvis du ønsker flere opplysninger, kan du slå opp i kapittelet om datatilgangsbegreper og optimalisering i *Administration Guide*.

Klasse for optimalisering av spørring

En *klasse for optimalisering av spørring* er et sett med spørreoppdateringsregler og optimaliseringteknikker som brukes til å kompilere spørringer.

Hovedklassene for optimalisering av spørringer:

1 Begrenset optimalisering. Denne er nyttig når minnet og behandlingsressursene er sterkt begrenset. Stort sett ekvivalent med optimaliseringen i Versjon 1.

2 Lett optimalisering. Viser et optimaliseringsnivå som er høyere enn versjon 1, men med betydelig mindre optimaliseringskostnader enn nivåene 3 og over, spesielt for svært komplekse spørringer.

3 Moderat optimalisering. Kommer nærmest et samsvar med spørringsoptimaliseringen som er karakteristisk for DB2 for MVS/ESA.

5 Normal optimalisering. Denne anbefales i et blandet miljø der det brukes både enkle transaksjoner og sammensatte spørringer.

7 Normal optimalisering. Det samme som spørringsoptimalisering 5,

med unntak av at den ikke reduserer størrelsen på spørringsoptimaliseringen for sammensatte, dynamiske SQL-spørringer.

Andre klasser for optimalisering av spørring som kan brukes i spesielle tilfeller:

0 Minimal optimalisering. Denne brukes bare når du trenger liten eller ingen optimalisering (bare for helt enkle spørringer i tabeller som er indeksert på en ordentlig måte).

9 Største optimalisering. Denne bruker atskillig minne og behandlingsressurser. Bruk denne bare hvis klasse 5 er utilstrekkelig (for spørringer som er veldig sammensatte, og som det tar lang tid å kjøre, og som ikke utføres på en vellykket måte med klasse 5).

Generelt bør du bruke en høyere optimaliseringsklasse for statistikkspørringer og for spørringer du antar vil ta lang tid å utføre, og en lavere optimaliseringsklasse for enkle spørringer som sendes dynamisk eller bare kjøres noen få ganger.

Når du skal definere spørringsoptimaliseringen for dynamiske SQL-setninger, oppgir du følgende kommando i kommandolinjebehandleren:

```
SET CURRENT QUERY OPTIMIZATION = n;
```

der 'n' er klassen du vil ha for optimalisering av spørring.

Når du skal definere spørringsoptimaliseringen for statiske SQL-setninger, bruker du QUERYOPT-alternativet i kommandoene **BIND** eller **PREP**.

Hvis du ønsker flere opplysninger, kan du slå opp i avsnittet om justering av optimaliseringsklassen i *Administration Guide*.

Selektivitet for predikater

Selektivitet viser til sannsynligheten for at en rad vil tilfredsstille et predikat (det vil si, være sant).

En selektivitet på for eksempel 0,01 (1%) for et predikat som opererer på en tabell med 1 000 000 rader, innebærer at predikatet anslagsvis returnerer 10 000 rader (1% av 1 000 000) og sletter estimatet på 990 000 rader.

Et svært selektivt predikat (har en selektivitet på 0,10 eller lavere) er ønskelig. Slike predikater returnerer færre rader som fremtidige operatorer må behandle. Dermed kreves det mindre CPU og I/U for å oppfylle kravene i spørringen.

Eksempel

Tenk deg at du har en tabell med 1 000 000 rader, og at den opprinnelige spørringen inneholder et 'ORDER BY'-ledd som krever ekstra sorteringstrinn. Hvis predikatet har en selektivitet på 0,01, må sorteringen gjøres på anslagsvis 10 000 rader. Med et mindre selektivt predikat på 0,50, må sorteringen gjøres på anslagsvis 500 000 rader, og det krever mer CPU- og I/U-tid.

Stjernekombinasjon

Et sett med kombinasjoner betraktes som en stjernekombinasjon når en produkttabell (stor, sentral tabell) kombineres med to eller flere formattabeller (mindre tabeller som inneholder beskrivelser av kolonneverdier i produkttabellen).

En stjernekombinasjon består av 3 hoveddeler:
- Delvise kombinasjoner (semijoins)
- Utføring av Index AND på resultatene av de delvise kombinasjonene
- Fullføring av de delvise kombinasjonene

Den vises som to eller flere kombinasjoner som mater en IXAND-operator.

En delvis kombinasjon (semijoin) er en spesiell type kombinasjon, hvor resultatet av kombineringen bare er radidentifikatoren (RID) til den interne tabellen, i stedet for en kombinasjon av de interne og eksterne tabellkolonnene.

Stjernekombinasjoner bruker delvise kombinasjoner til å forsyne en Index ANDing-operator med radidentifikatorer. Index ANDing-operatoren akkumulerer filtreringseffekten ved de forskjellige kombinasjonene. Utdataene fra Index ANDing-operatoren mates inn i en Index ORing-operator, som sorterer radidentifikatorene og eliminerer like rader som kan ha oppstått under kombinasjonene som matet Index ANDing-operatoren. Radene fra faktatabellen blir så hentet ved hjelp av en henteoperator. Til slutt blir den reduserte faktatabellen kombinert med alle dimensjonstabellene, for å fullføre kombinasjonene.

Gjør slik:
- Opprett indekser på faktatabellen for hver av dimensjonstabellkombinasjonene.
- Kontroller at terskelen for minneområde for sortering er høy nok til å tillate tildeling av Index ANDing-operatorens bitfilter. Stjernekombinasjoner kan kreve opptil 12 MB (eller 3000 sider på 4 kB). Ved intrapartisjonparallellitet tildeles bitfilteret fra samme minnesegment som dbheap, men er avgrenset av minneområdet for sortering (og sheapthres over hele forekomsten). De

delte minnet er dermed styrt av minneområdet for sortering (sortheap) og sheapthres, og kan kreve mer enn 12 MB.

- Bruk filterpredikater mot dimensjonstabellene. Hvis statistikken ikke er oppdatert, kan du oppdatere den med runstats-kommandoen.

Statisk SQL

En *statisk SQL*-setning er innfelt i et applikasjonsprogram. Alle disse innfelte setningene må forkompileres og bindes til en *pakke* før applikasjonen kan kjøres.

Når DB2 kompilerer disse setningene, oppretter det en tilgangsplan for hver av dem som er basert på katalogstatistikken og konfigurasjonsparameterne på det tidspunktet da setningene ble forkompilert og bundet.

Disse tilgangsplanene brukes alltid når applikasjonen kjøres. De forandres ikke før pakken blir bundet igjen.

Alternativet til statisk SQL er dynamisk SQL.

SMS-tabellplasser (systemstyrt plass)

Det er to typer tabellplasser som kan finnes i en database: databasestyrt plass (DMS-plass).

En SMS-tabellplass styres av operativsystemet, som lagrer databasedataene på en plass som tildeles når en tabellplass blir opprettet. Definisjonen av tabellplassen inneholder en liste over en eller flere baner der disse dataene blir lagret.

Filsystemet administrerer tildeling og administrering av lageret.

SMS- og DMS-tabellplasser kan eksistere sammen i en database.

Tabellplass

Det er lettere å styre store databaser hvis du partisjonerer dem i separate deler som kalles *tabellplasser*.

En tabellplass bruker du til å definere plasseringen av data til bestemte logiske enheter eller deler av slike enheter. Når du for eksempel oppretter en tabell, kan du oppgi at indeksene i tabellen eller dens lange kolonner med lange eller store objekter (LOB-data), skal holdes atskilt fra resten av tabelldataene.

En tabellplass kan være fordelt over en eller flere fysiske lagringsenheter (containere) for bedre ytelse. Det anbefales imidlertid at alle enheter eller containere i en tabellplass har like ytelsesegenskaper.

En tabellplass kan styres på to forskjellige måter: som en systemstyrt plass (SMS) eller som en databasestyrt plass (DMS).

Visuell forklaring

Merk: Fra og med versjon 6 kan ikke Visuell forklaring lengre startes fra kommandolinjen. Det kan imidlertid fremdeles startes fra ulike databaseobjekter i Kontrollsenter. For denne versjonen brukes fremdeles navnet Visuell forklaring i dokumentasjonen.

Med Visuell forklaring kan du se på tilgangsplanen for forklarte SQL-setninger, i form av et diagram. Du kan bruke opplysningene i diagrammet til å tilpasse SQL-spørringene slik at du får bedre ytelse.

Et tilgangsplandiagram viser detaljer om:
- tabeller (og tilknyttede kolonner) og indekser
- operatorer (for eksempel tabellsøkinger, sorteringer og kombineringer)
- tabellplasser og -funksjoner

Du kan også bruke Visuell forklaring til å:
- Se på statistikken som ble brukt da optimalisering ble utført. Du kan deretter sammenlikne denne statistikken med den gjeldende katalogstatistikken for å finne ut om du kan få bedre ytelse ved å binde pakken på nytt.
- Finne ut om det ble brukt en indeks til å få tilgang til en tabell. Hvis det ikke ble brukt indeks, kan Visuell forklaring hjelpe deg å finne ut hvilke kolonner det kan være nyttig å indeksere.
- Se på effekten av forskjellige tilpassingsteknikker, ved å sammenlikne versjonene av et tilgangsplandiagram for en spørring før og etter tilpassingen.
- Få informasjon om hver operasjon i tilgangsplanen, inkludert total anslått kostnad og antall rader som ble hentet (kardinalitet).

Tillegg B. Alfabetisk liste over operatorer for Visuell forklaring

CMPEXP

Operatornavn: CMPEXP

Beskrivelse: Beregningen av uttrykk som er nødvendig for midlertidige eller endelige resultater.

(Denne operatoren er bare for feilsøkingsmodus.)

DELETE

Operatornavn: DELETE

Beskrivelse: Sletter rader fra en tabell.

Denne operatoren representerer en nødvendig operasjon. Hensikten er å redusere tilgangsplankostnadene ved å fokusere på andre operatorer (som søk og kombinasjoner) som definerer settet med rader som skal slettes.

Gjør slik:

- Hvis du sletter alle radene fra en tabell, bør du bruke kommandoen DROP TABLE eller **LOAD REPLACE**.

EISCAN

Operatornavn: EISCAN

Beskrivelse: Denne operatoren søker igjennom en brukerdefinert indeks for å lage en redusert datastrøm av rader. Søket bruker de mange start/stop-betingelsene fra Range producer-funksjonen som er oppgitt av brukeren.

Denne operasjonen utføres for å redusere settet med kvalifiserende rader før basistabellen lastes (basert på predikater).

Gjør slik:

- Over tid kan databaseoppdateringer resultere i at en indeks blir fragmentert, noe som igjen fører til at du får flere indekssider enn nødvendig. Du kan rette dette ved å slette eller opprette indeksen på nytt, eller ved å reorganisere indeksen.
- Hvis statistikken ikke er oppdatert, kan du oppdatere den med runstats-kommandoen.

FETCH

Operatornavn: FETCH

Beskrivelse: Henter kolonner fra en tabell ved hjelp av en bestemt radidentifikator (RID).

Gjør slik:

- Ta med kolonnene som er hentet i indeksnøkkelen, slik at du slipper å bruke datasidene.
- Finn indeksen som er knyttet til hentingen, og dobbeltklikk på noden for å få frem statistikkvinduet. Kontroller at graden av gruppering er høy for indeksen.
- Øk bufferstørrelsen hvis inndataene/utdataene (I/U) som ble hentet, overskriver antall sider i tabellen.
- Hvis statistikken ikke er oppdatert, kan du oppdatere den med runstats-kommandoen.

  Statistikken for kontrollverdi og hyppig verdi gir opplysninger om selektivitet for predikater, som bestemmer når det skal søkes i en indeks og ikke i en tabell. Når du skal oppdatere denne statistikken, må du bruke kommandoen **RUNSTATS** på en tabell med WITH DISTRIBUTION-leddet.

FILTER

Operatornavn: FILTER

Beskrivelse: Bruk av restverdipredikater slik at data blir filtrert etter de kriteriene som predikatene tilsier.

Gjør slik:

- Kontroller at du har brukt predikater som bare henter de dataene du trenger. Kontroller for eksempel at selektivitetsverdien for predikatene representerer den delen av tabellen som du vil ha returnert.
- Kontroller at optimaliseringsklassen er minimum 3 slik at optimalisatoren bruker en kombinering (JOIN) i stedet for en delspørring. Hvis det ikke er mulig, kan du prøve å skrive om SQL-spørringen på nytt for hånd, slik at

du eliminerer delspørringen. Du finner et eksempel hvis du slår opp i kapittelet som omhandler omskriving av spørringer med SQL-kompilatoren, i *Administration Guide*.

GENROW

Operatornavn: GENROW

Beskrivelse: En innebygd funksjon som genererer en tabell med rader uten å bruke noen form for inndata fra tabeller, indekser eller operatorer.

Optimalisatoren kan bruke GENROW når den skal lage rader med data (for eksempel for en INSERT-setning eller for noen IN-lister som er overført til kombinasjoner).

Når du vil se på statistikken som er beregnet for tabeller som GENROW-funksjonen har laget, dobbeltklikker du på noden.

GRPBY

Operatornavn: GRPBY

Beskrivelse: Gruppering av rader i henhold til vanlige verdier for definerte kolonner eller funksjoner. Operasjonen er nødvendig for å lage en gruppe verdier, eller for å evaluere definerte funksjoner.

Selv om det ikke er oppgitt en GROUP BY-kolonne, kan GRPBY-operatoren brukes hvis det finnes samlingsfunksjoner i SELECT-listen som oppgir at hele tabellen er behandlet som en enkelt gruppe under samlingen.

Gjør slik:

- Denne operatoren representerer en nødvendig operasjon. Hensikten er å redusere tilgangsplankostnadene ved å fokusere på andre operatorer (som søk og kombinasjoner) som definerer hvilket sett med rader som skal grupperes.
- Hvis du vil forbedre ytelsen til en SELECT-setning som inneholder en enkelt samlefunksjon, men ingen GROUP BY-ledd, kan du forsøke følgende:
  - For en MIN(C)-samlefunksjon lager du en stigende indeks på C.
  - For en MAX(C)-samlefunksjon lager du en synkende indeks på C.

HSJOIN

Operatornavn: HSJOIN

Beskrivelse: En nøkkelkombinasjon (hash join) der kvalifiserte rader fra tabeller indekseres slik at direkte kombinering uten forhåndssortering av innholdet er mulig.

En kombinasjon er nødvendig når et FROM-ledd viser til flere tabeller. En nøkkelkombinasjon er mulig hvis det finnes et kombinasjonspredikat som sammenligner kolonner fra to forskjellige tabeller. Kombinasjonspredikatene må ha nøyaktig samme datatype. Nøkkelkombinasjoner kan også komme fra en delspørring som er skrevet om, som for eksempel NLJOIN.

Nøkkelkombinasjoner krever ikke at tabellene sorteres på forhånd. Kombineringen utføres ved å søke gjennom den indre tabellen og lage en oppslagstabell ved å indeksere kombinasjonskolonneverdiene. Deretter leses den ytre tabellen, kombinasjonskolonneverdiene blir indeksert (hashed) og oppslagstabellen for den interne tabellen blir kontrollert.

Hvis du ønsker flere opplysninger, kan du slå opp i kapittelet om kombinasjonsbegreper i *Administration Guide*.

Gjør slik:

- Bruk lokale predikater (det vil si predikater som viser til en tabell) når du skal redusere antall rader som skal kombineres.
- Øke størrelsen til minneområdet for sortering slik at det blir stort nok til å romme oppslagstabellen.
- Hvis statistikken ikke er oppdatert, kan du oppdatere den med runstats-kommandoen.

INSERT

Operatornavn: INSERT

Beskrivelse: Setter rader inn i en tabell.

Denne operatoren representerer en nødvendig operasjon. Hensikten er å redusere tilgangsplankostnadene ved å fokusere på andre operatorer (som søk og kombinasjoner) som definerer hvilket sett med rader som skal settes inn.

IXAND

Operatornavn: IXAND

Beskrivelse: Føyer sammen resultatene av flere indekssøk ved hjelp av dynamiske bitmønsterteknikker. Operatoren gjør det mulig for predikater det er utført AND på, å bli brukt på flere indekser for å redusere tilgang til underliggende tabeller til et minimum.

Denne operatoren utføres for å
- begrense settet med rader før basistabellen lastes
- føye sammen (AND) predikater som er brukt i flere indekser
- føye sammen (AND) resultatene fra delkombinasjoner, som er brukt i stjernekombinasjoner

Gjør slik:

- Over tid kan databaseoppdateringer resultere i at en indeks blir fragmentert, noe som igjen fører til at du får flere indekssider enn nødvendig. Du kan rette dette ved å slette eller opprette indeksen på nytt, eller ved å reorganisere indeksen.
- Hvis statistikken ikke er oppdatert, kan du oppdatere den med runstats-kommandoen.
- Vanligvis er indekssøk mest effektivt når du bare har få kvalifiserte rader. Optimalisatoren bruker statistikken som er tilgjengelig for kolonnene det er referert til i predikater, når den skal beregne antall kvalifiserte rader. Hvis enkelte verdier opptrer oftere enn andre, er det viktig at du ber om fordelingsstatistikk. Det gjør du ved å bruke WITH DISTRIBUTION-leddet sammen med kommandoen **RUNSTATS**. Hvis du bruker den ujevne fordelingsstatistikken, kan optimalisatoren skille mellom verdier som opptrer ofte, og verdier som opptrer sjelden.
- IXAND kan best utnytte indekser med enkeltkolonner, siden start- og stoppnøklene er helt nødvendig ved bruk av IXAND.
- Når det gjelder stjernekombinasjoner, bør du opprette indekser med enkeltkolonner for hver av de mest selektive kolonnene i produkttabellen og de beslektede formattabellene.

IXSCAN

Operatornavn: IXSCAN

Beskrivelse: Søker i en indeks for å generere en redusert datastrøm av rader. Søkingen kan bruke valgfrie start- og stoppvilkår, eller den kan brukes på indekserbare predikater som viser til kolonner i indeksen.

Denne operasjonen utføres for å redusere settet med kvalifiserende rader før basistabellen lastes (basert på predikater).

Hvis du ønsker flere opplysninger, kan du slå opp i kapittelet om indekssøk i *Administration Guide*.

Gjør slik:

- Over tid kan databaseoppdateringer resultere i at en indeks blir fragmentert, noe som igjen fører til at du får flere indekssider enn nødvendig. Du kan rette dette ved å slette eller opprette indeksen på nytt, eller ved å reorganisere indeksen.
- Når du bruker to eller flere tabeller, får du mer effektiv tilgang via en indeks til den interne tabellen hvis du bruker en indeks for kombinasjonskolonnen til den eksterne tabellen.

  Hvis du ønsker flere opplysninger om indekser, kan du se i hjelpen for Visuell forklaring.
- Hvis statistikken ikke er oppdatert nylig, må du oppdatere den ved hjelp av runstats-kommandoen.
- Vanligvis er indekssøk mest effektivt når du bare har få kvalifiserte rader. Optimalisatoren bruker statistikken som er tilgjengelig for kolonnene det er referert til i predikater, når den skal beregne antall kvalifiserte rader. Hvis enkelte verdier opptrer oftere enn andre, er det viktig at du ber om fordelingsstatistikk. Det gjør du ved å bruke WITH DISTRIBUTION-leddet sammen med kommandoen **RUNSTATS**. Hvis du bruker den ujevne fordelingsstatistikken, kan optimalisatoren skille mellom verdier som opptrer ofte, og verdier som opptrer sjelden.

MSJOIN

Operatornavn: MSJOIN

Beskrivelse: En samkjøringskombinasjon der de kvalifiserte radene fra både eksterne og interne tabeller må være i samme rekkefølge som i det kombinerte predikatet. En samkjøringskombinasjon kalles også en *kombinasjon av samkjørte søk* eller en *kombinasjon av sorterte søk*.

En kombinasjon er nødvendig når et FROM-ledd viser til flere tabeller. En samkjøringskombinasjon er alltid mulig hvis det finnes et kombinasjonspredikat som utlikner kolonnene fra to forskjellige tabeller. Det kan også komme fra en delspørring som er skrevet om.

En samkjøringskombinasjon krever at inndataene i kombinasjonskolonner er sortert, fordi tabellene bare blir bladd gjennom en gang. Du får tilgang til sorterte inndata ved hjelp av en indeks eller en sortert tabell.

Hvis du ønsker flere opplysninger, kan du slå opp i kapittelet om kombinasjonsbegreper i *Administration Guide*.

Gjør slik:

- Bruk lokale predikater (det vil si predikater som viser til en tabell) når du skal redusere antall rader som skal kombineres.

Du finner flere retningslinjer om indekser under Opprette riktige indekser i hjelpen til Visuell forklaring.

- Hvis statistikken ikke er oppdatert, kan du oppdatere den med runstats-kommandoen.

NLJOIN

Operatornavn: NLJOIN

Beskrivelse: En nestet sløyfekombinasjon som blar gjennom (vanligvis med et indekssøk) den interne tabellen en gang for hver rad i den eksterne tabellen.

En kombinasjon er nødvendig når et FROM-ledd viser til flere tabeller. En nestet sløyfekombinasjon krever ikke et kombinasjonspredikat, men utføringen blir bedre hvis du bruker det.

En nestet sløyfekombinasjon blir utført

- enten ved at programmet blar gjennom den interne tabellen for alle rader som er brukt i den eksterne tabellen
- eller ved at programmet slår opp i en indeks for den interne tabellen for alle rader som er brukt i den eksterne tabellen

Hvis du ønsker flere opplysninger, kan du slå opp i kapittelet om kombinasjonsbegreper i *Administration Guide*.

Gjør slik:

- En nestet sløyfekombinasjon blir sannsynligvis mer effektiv hvis det finnes en indeks på kombinasjonspredikatkolonnene i den interne tabellen (tabellen som vises til høyre for NLJOIN-operatoren). Kontroller om den interne tabellen er en TBSCAN, og ikke en IXSCAN. Hvis den er en TBSCAN, bør du vurdere å tilføye en indeks på kombinasjonskolonnene til tabellen.

  En annen (mindre viktig) måte å gjøre kombinasjonen mer effektiv på, er at du oppretter en indeks på kombinasjonskolonnene i den eksterne tabellen, slik at tabellen blir sortert.

  Du finner flere retningslinjer om indekser under Opprette riktige indekser i hjelpen til Visuell forklaring.
- Hvis statistikken ikke er oppdatert, kan du oppdatere den med runstats-kommandoen.

Annen informasjon:

- Stjernekombinasjon.

PIPE

Operatornavn: PIPE

Beskrivelse: Overføring av rader til andre operatorer uten å endre dem.

(Denne operatoren er bare for feilsøkingsmodus.)

RETURN

Operatornavn: RETURN

Beskrivelse: Returnering av data fra en spørring til brukeren. Dette er den siste operatoren i tilgangsplandiagrammet og viser totalt akkumulerte verdier og kostnader for tilgangsplanen.

Denne operatoren representerer en nødvendig operasjon.

Gjør slik:

- Kontroller at du har brukt predikater som bare henter de dataene du trenger. Kontroller for eksempel at selektivitetsverdien for predikatene representerer den delen av tabellen som du vil ha returnert.

RIDSCN

Operatornavn: RIDSCN

Beskrivelse: Søking i en liste med radidentifikatorer (RIDer) som ble funnet i en eller flere indekser.

Optimalisatoren tar hensyn til denne operatoren under disse forholdene:

- Når predikater er knyttet sammen med OR-nøkkelord, eller hvis det finnes et IN-predikat. Du kan bruke OR i indekser (såkalt ″index ORing″), og da blir resultater fra flere indekstilganger mot samme tabell kombinert.
- Når det er en fordel å bruke forhåndshenting fra liste for en enkelt indekstilgang, fordi I/U-operasjonene blir mer effektive hvis du sorterer radidentifikatorene før du går inn på basisradene.

RQUERY

Operatornavn: SHIP

Beskrivelse: En operator som brukes i det forente systemet til å hente data fra en fjerntliggende datakilde. Denne operatoren blir vurdert av optimalisatoren når: En SHIP-operator sender en SQL SELECT-setning til en fjerntliggende datakilde for å hente søkeresultatet. SELECT-setningen blir generert ved hjelp

av den SQL-dialekten som støttes av datakilden, og kan inneholde en hvilken som helst gyldig spørring, slik den er tillatt av datakilden.

Forslag til ytelse: Slå opp i kapittel 4 i boken Administration Guide Vol 2, Federated Database Query and Network Tuning Information.

SORT

Operatornavn: SORT

Beskrivelse: Sortering av radene i en tabell i samme rekkefølge som en eller flere av kolonnene i tabellen. Du kan velge om du vil slette poster som er like.

Sortering er nødvendig når det ikke finnes noen indeks som tilfredsstiller rekkefølgen du bad om, eller fordi sortering er mindre kostnadskrevende enn indekssøk. Sortering utføres vanligvis som en siste operasjon etter at de nødvendige radene er hentet, eller for å sortere data før det blir foretatt en kombinering (JOIN) eller en gruppering etter (GROUPBY).

Hvis det er mange rader, eller hvis de sorterte dataene ikke videresendes, krever operasjonen at du lager midlertidige tabeller, som er mer kostnadskrevende.

Hvis du ønsker flere opplysninger om sorteringer, kan du slå opp i *Administration Guide*.

Gjør slik:

- Vurder om du skal tilføye en indeks på sorteringskolonnene.

  Du finner flere retningslinjer om indekser under Opprette riktige indekser i hjelpen til Visuell forklaring.
- Kontroller at du har brukt predikater som bare henter de dataene du trenger. Kontroller for eksempel at selektivitetsverdien for predikatene representerer den delen av tabellen som du vil ha returnert.
- Kontroller at størrelsen på forhåndshentingen for den midlertidige systemtabellplassen er stor nok, det vil si at den ikke er begrenset av I/U. (For å kontrollere dette må du velge **Setning->Vis statistikk->Tabellplasser**.)
- Hvis det ofte er behov for store sorteringer, bør du vurdere om du skal øke verdiene til disse konfigurasjonsparameterne:
  - Minneområde for sortering (sortheap). Hvis du vil endre denne parameteren, klikker du med høyre museknapp på databasen i Kontrollsenter og velger deretter *Konfigurer* fra objektmenyen. Velg **Ytelse** fra notisboken som kommer frem.

– Sorteringsterskel for minneområde (sheapthres). Hvis du vil endre denne parameteren, klikker du med høyre museknapp på databaseforekomsten i Kontrollsenter og velger *Konfigurer* fra objektmenyen. Velg **Ytelse** fra notisboken som kommer frem.

- Hvis statistikken ikke er oppdatert, kan du oppdatere den med runstats-kommandoen.

TBSCAN

Operatornavn: TBSCAN

Beskrivelse: Et tabellsøk (relasjonssøk) som henter rader ved å lese alle nødvendige data direkte fra datasidene.

Optimalisatoren velger denne typen søk fremfor indekssøk når:
- verdiområdet som blir bladd igjennom, forekommer ofte (det vil si at mesteparten av tabellen må være i bruk)
- tabellen er liten
- det er lav indeksgruppering
- det ikke finnes noen indeks

Hvis du ønsker flere opplysninger om indekssøk, kan du slå opp i *Administration Guide*.

Gjør slik:

- Et indekssøk er mer effektivt enn et tabellsøk hvis tabellen er stor, og hvis de fleste av radene i tabellen ikke er brukt. For å øke muligheten for at optimalisatoren skal bruke et indekssøk i denne situasjonen, bør du vurdere å tilføye indekser på kolonner som det finnes selektive predikater for.

  Du finner flere retningslinjer om indekser under Opprette riktige indekser i hjelpen til Visuell forklaring.
- Hvis det allerede finnes en indeks, men den ikke ble brukt, må du kontrollere at det finnes selektive predikater for hver av de første kolonnene indeksen. Hvis det ikke finnes slike predikater, må du kontrollere at graden av gruppering er høy for indeksen. (Hvis du vil se denne statistikken, åpner du vinduet Tabellstatistikk for tabellen under sorteringen og klikker på skjermtasten *Indekser* for å få frem vinduet Indeksstatistikk.)
- Kontroller at størrelsen på forhåndshentingen for tabellplassen er stor nok, det vil si at den ikke er begrenset av I/U. (For å kontrollere dette må du velge **Setning->Vis statistikk->Tabellplasser**.)

  Hvis du vil ha flere opplysninger, kan du se under delen som omhandler forhåndshenting av data inn i bufferområdet i *Administration Guide*.

- Hvis statistikken ikke er oppdatert, kan du oppdatere den med runstats-kommandoen.

  Kontrollverdien og hyppig verdistatistikk gir informasjon om selektivitet for predikater. Denne statistikken blir for eksempel brukt til avgjøre når indekssøk foretrekkes fremfor tabellsøk. Når du skal oppdatere disse verdiene, bruker du kommandoen **RUNSTATS** på en tabell med WITH DISTRIBUTION-leddet.

TEMP

Operatornavn: TEMP

Beskrivelse: Lagring av data i en midlertidig tabell som skal leses tilbake av en annen operator (muligens flere ganger). Tabellen blir fjernet når SQL-setningen er behandlet, hvis ikke før.

Denne operatoren er nødvendig for å evaluere delspørringer eller for å lagre midlertidige resultater. I enkelte tilfeller (for eksempel når setningen kan oppdateres) kan den være obligatorisk.

TQUEUE

Operatornavn: TQUEUE

Beskrivelse: En tabellkø som brukes til å sende tabelldata fra en databaseagent til en annen når det er flere databaseagenter som behandler en spørring. Flere databaseagenter brukes til å behandle en spørring når parallellitet er involvert.

Tabellkøtyper:

- **Lokal:** Tabellkøen brukes til å sende data mellom databaseagenter innenfor en enkelt node. En lokal tabellkø brukes ved intrapartisjonparallellitet.
- **Ikke lokal:** Tabellkøen brukes til å sende data mellom databaseagenter på forskjellige noder.

UNION

Operatornavn: UNION

Beskrivelse: Sammenkjeding av strømmer av rader fra flere tabeller.

Denne operatoren representerer en nødvendig operasjon. Hensikten er å redusere tilgangsplankostnadene ved å fokusere på andre operatorer (for eksempel søk og kombinasjoner) som definerer settet med rader som skal slås sammen.

UNIQUE

Operatornavn: UNIQUE

Beskrivelse: Sletting av rader som har like verdier i oppgitte kolonner.

Gjør slik:

- Denne operatoren er ikke nødvendig hvis det finnes en entydig indeks for de riktige kolonnene.

  Du finner flere retningslinjer om indekser under Opprette riktige indekser i hjelpen til Visuell forklaring.

UPDATE

Operatornavn: UPDATE

Beskrivelse: Oppdatering av data i radene i en tabell.

Denne operatoren representerer en nødvendig operasjon. Hensikten er å redusere tilgangsplankostnadene ved å fokusere på andre operatorer (for eksempel søk og kombinasjoner) som definerer settet med rader som skal oppdateres.

Tillegg C. DB2-begreper

Databaser

En relasjonsdatabase viser data som en samling av tabeller. En tabell består av et definert sett av kolonner og et hvilket som helst antall rader. Dataene i hver tabell er beslektet logisk, og forholdene kan defineres mellom tabeller. Data kan vises og manipuleres ved hjelp av matematiske prinsipper og operasjoner som kalles relasjoner (for eksempel INSERT, SELECT og UPDATE).

En database er selvbeskrivende i og med at den inneholder en beskrivelse av sin egen struktur, i tillegg til data. Den inneholder et sett av systemkatalogtabeller, som beskriver den logiske og fysiske strukturen til dataene. Den inneholder også en konfigurasjonsfil, som inneholder parameterverdiene som er knyttet til databasen, og en gjenopprettingslogg, som registrerer pågående transaksjoner og transaksjoner som kan arkiveres.

Databaser kan være lokale eller fjerntliggende. En lokal database er fysisk plassert på arbeidsstasjonen som brukes, mens en database på en annen maskin blir kalt fjerntliggende.

Skjemaer

Et skjema er en entydig identifikator som brukes til å gruppere et sett av databaseobjekter (for eksempel tabeller, utsnitt, indekser og tilnavn). Hvis du oppretter en tabell men navnet LØNN, ville det være kjedelig å måtte søke i databasen for å finne ut om en annen bruker allerede har opprettet en tabell med det navnet. Navnet på hvert objekt må være entydig bare i det tilhørende skjemaet.

De fleste databaseobjekter har et objektnavn i to deler. Den første delen er skjemanavnet og den andre delen er navnet på objektet. Når et objekt blir opprettet, kan du tilordne det til et bestemt skjema. Hvis du ikke oppgir et skjema, blir det tilordnet til standardskjemaet, som vanligvis er bruker-IDen for personen som har opprettet objektet. En bruker med navnet Smith kan for eksempel har en tabell med navnet SMITH.LØNN.

Skjemaet blir også et objekt i databasen. Den opprettes når det første objektet i skjemaet blir opprettet. Et skjema kan eies av en enkeltperson, og eieren kan styre tilgangen til dataene og objektene i det.

Tabeller

En relasjonsdatabase viser data som en samling av tabeller. En tabell består av data som er ordnet logisk i kolonner og rader (også kjent som poster).

Hver tabell har et navn, og i en tabell har hver kolonne et navn. Ingen bestemt rekkefølge blir opprettholdt mellom radene i en tabell, men radene kan hentes i en rekkefølge som er fastsatt av verdiene i kolonnene. Dataene i en tabell er logisk beslektet. Alle databasedata og tabelldata er tilordnet til tabellplasser.

Tillegg D. Merknader

Henvisninger til IBMs produkter, programmer eller tjenester betyr ikke at IBM har til hensikt å gjøre dem tilgjengelige i alle land der IBM driver virksomhet. Be din lokale IBM-representant om informasjon om hvilke produkter og tjenester som er tilgjengelige i Norge. Henvisninger til IBMs produkter, programmer eller tjenester betyr heller ikke at det bare er de som kan benyttes. Andre produkter, programmer eller tjenester som har tilsvarende funksjoner, kan brukes i stedet, forutsatt at de ikke gjør inngrep i noen av IBMs patent- eller opphavsrettigheter eller andre lovbeskyttede rettigheter. Vurdering og verifisering ved bruk sammen med andre produkter, programmer eller tjenester enn de som uttrykkelig er angitt av IBM, er brukerens ansvar.

IBM kan ha patent på eller patentsøknader til behandling for de produktene som er omtalt i denne publikasjonen. At du har mottatt denne publikasjonen, innebærer ikke at du får lisensrettighet til disse produktene. Du kan sende spørsmål angående lisenser til

Director of Commercial Relations - Europe
IBM Deutschland GmbH
Schönaicher Str. 220
D - 7030 Böblingen
Tyskland

Lisensforespørsler om dobbeltbyteinformasjon (DBCS) kan rettes til IBMs advokat eller til:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

INTERNATIONAL BUSINESS MACHINES CORPORATION LEVERER DENNE BOKEN SOM DEN ER (“AS IS”) UTEN FORPLIKTELSER AV NOE SLAG.

Denne boken kan inneholde tekniske unøyaktigheter eller typografiske feil. Opplysninger i denne boken kan bli endret. Slike endringer blir tatt med i nye utgaver av boken. IBM kan uten varsel endre produktene og/eller programmene som er beskrevet i denne boken.

Eventuelle henvisninger i denne informasjonen til nettsteder som ikke tilhører IBM, er bare til orientering og innebærer på ingen måte noen godkjennelse

eller støtte til disse nettstedene. Produktene på disse nettstedene er ikke del av dette IBM-produktet, og bruk av disse nettstedene skjer på eget ansvar.

IBM kan bruke eller distribuere informasjonen du gir til IBM på den måten IBM mener er best, uten forpliktelser i noen retning.

Hvis du som lisensinnehaver av dette programmet ønsker informasjon om programmet for å kunne: (i) utveksle informasjon mellom selvstendig utviklede programmer og andre programmer (inklusiv dette) og (ii) dra gjensidig nytte av informasjonen som er utvekslet, kan du kontakte:

IBM Norge AS
Software Marketing
Postboks 500
1411 Kolbotn

Slik informasjon kan være tilgjengelig under gjeldende betingelser, eventuelt mot betaling.

Det lisensierte programmet som er beskrevet i dette dokumentet, og alt lisensiert materiale som er tilgjengelig for programmet, leveres av IBM i henhold til IBM Generelle betingelser, IBMs internasjonale bruksbetingelser eller en tilsvarende avtale mellom partene.

Alle ytelsesdataene du finner i dette dokumentet, ble hentet i et kontrollert miljø. Resultatene du kan oppnå i andre operativmiljøer, kan variere betraktelig. Noen av målingene er foretatt på systemer som er under utvikling, og det er ikke sikkert at du oppnår samme resultat på alminnelige tilgjengelige systemer. Noen av målingene kan dessuten ha blitt beregnet ved hjelp av ekstrapolasjon. De faktiske resultatene kan variere. Brukerne av dette dokumentet bør bekrefte dataene som brukes i sitt bestemte miljø.

Informasjon om ikke-IBM-produkter er innhentet fra leverandørene av produktene, fra deres annonseringer eller fra andre allment tilgjengelige kilder. IBM har ikke testet produktene og kan ikke garantere nøyaktigheten av opplysninger om ytelse og kompatibilitet eller andre opplysninger om ikke-IBM-produkter. Spørsmål om funksjonene i ikke-IBM-produkter må rettes til leverandøren av produktet.

Enhver henvisning til IBMs fremtidige planer eller hensikter kan endres eller trekkes tilbake uten varsel. De er kun ment å være en målsetting.

Denne dokumentasjonen kan inneholde eksempler på data og rapporter som brukes i daglige forretningsoperasjoner. For å illustrere eksemplene så godt som mulig blir det brukt navn på personer, firmaer og produkter. Alle disse navnene er fiktive, og enhver likhet med virkelige navn er tilfeldig.

RETT TIL KOPIERING:

Denne informasjonen inneholder eksempelapplikasjoner, i kildespråk, som viser programmeringsteknikker i forskjellige operativsystemer. Du kan kopiere, endre og distribuere disse eksempelprogrammene i en hvilken som helst form uten betaling til IBM, med den hensikt å utvikle, bruke, markedsføre eller distribuere applikasjoner som følger programmeringsgrensesnittet (API) for operativsystemet som eksempelprogrammene er skrevet for. Disse eksemplene er ikke testet inngående under alle forhold. IBM kan derfor ikke garantere eller antyde at disse programmene er pålitelige, at det tilbys service for dem, eller at de virker.

Hver kopi eller del av disse eksempelprogrammene eller utledet arbeid fra dem, må inneholde en slik merknad om opphavsrett:

© (*ditt firmanavn*) (*år*). Deler av denne koden er utledet fra eksempelprogrammer fra IBM Corp. © Copyright IBM Corp. *_oppgi året eller årene_*. All rights reserved.

Varemerker

Navnene nedenfor er varemerker for International Business Machines Corporation i USA og/eller andre land, og er brukt i minst ett av dokumentene i dokumentasjonsbiblioteket til DB2 UDB.

ACF/VTAM
AISPO
AIX
AIXwindows
AnyNet
APPN
AS/400
BookManager
C Set++
C/370
CICS
Database 2
DataHub
DataJoiner
DataPropagator
DataRefresher
DB2
DB2 Connect
DB2 Extenders
DB2 OLAP Server
DB2 Universal Database
Distributed Relational Database Architecture
DRDA
eServer
Extended Services
FFST
First Failure Support Technology
IBM
IMS
IMS/ESA
iSeries
LAN Distance
MVS
MVS/ESA
MVS/XA
Net.Data
NetView
OS/390
OS/400
PowerPC
pSeries
QBIC
QMF
RACF
RISC System/6000
RS/6000
S/370
SP
SQL/400
SQL/DS
System/370
System/390
SystemView
Tivoli
VisualAge
VM/ESA
VSE/ESA
VTAM
WebExplorer
WebSphere
WIN-OS/2
z/OS
zSeries

Navnene nedenfor er varemerker eller registrerte varemerker for andre selskaper, og er brukt i minst ett av dokumentene i dokumentasjonsbiblioteket til DB2 UDB:

Microsoft, Windows, Windows NT og Windows-logoen er varemerker for Microsoft Corporation i USA og/eller andre land.

Intel og Pentium er varemerker for Intel Corporation i USA og/eller andre land.

Java og alle Java-baserte varemerker er varemerker for Sun Microsystems, Inc i USA og/eller andre land.

UNIX er et registrert varemerke for Open Group i USA og/eller andre land.

Andre navn på firmaer, produkter eller tjenester kan være varemerker for andre selskaper.

Stikkordregister

T

U

V

Z

Kontakte IBM

I USA kan du ringe et av disse numrene:

- 1-800-237-5511 for kundeservice
- 1-888-426-4343 hvis du vil vite mer om tilleggstjenester
- 1-800-IBM-4YOU (426-4968) for DB2-markedsføring og -salg

I Canada kan du ringe et av disse numrene:

- 1-800-IBM-SERV (1-800-426-7378) for kundeservice
- 1-800-465-9600 for å få vite mer om tilgjengelige tilleggstjenester
- 1-800-IBM-4YOU (1-800-426-4968) for DB2-markedsføring og -salg

Når du skal finne et IBM-kontor i nærheten av der du bor, kan du se i IBMs oversikt over kontakter over hele verden på World Wide Web på www.ibm.com/planetwide

Om programmet

Informasjon om DB2 Universal Database-produkter er tilgjengelig på telefon eller på World Wide Web på www.ibm.com/software/data/db2/udb

Dette nettstedet inneholder den nyeste informasjonen om det tekniske biblioteket, bestilling av bøker, nedlastinger, nyhetsgrupper, opprettingspakker, nyheter og koblinger til web-ressurser.

Hvis du er i USA, kan du ringe et av disse numrene:

- 1-800-IBM-CALL (1-800-426-2255) for å bestille produkter eller få generell informasjon.
- 1-800-879-2755 for å bestille publikasjoner.

Du finner ut hvordan du kontakter IBM utenfor USA på siden IBM Worldwide på www.ibm.com/planetwide