Linnéuniversitetet
Institutionen för datavetenskap

Laborationsuppgift

Optimerade kontakter

Steg 1, laborationsuppgift 3

Författare Sven Åke Johansson

Kurskod: 1DV409

1. Uppgift

Du har fått i uppgift att utveckla en applikation till vilken du ska använda en databas i Microsoft SQL Server. Kunden har sedan tidigare en databas med data i som ligger i en Microsoft Access databas.

Vi har importerat Access databasen till MS SQL Server och till en egen databas där du kan jobba vidare med den. Den importerade databasen heter DemoMedlem och innehåller endast en tabell som heter Medlem (strukturen hittar du sist i detta dokument) som ska analyseras, kontrolleras och byggas om. Du ska också göra en del tester för fram svarstider för ett antal SQL-frågor.

- Ny databasdesign utifrån den gamla. Normalisera och analysera databasen och skapa sedan en ny konceptuell och fysisk modell. Modellen ska vara optimerad enligt optimeringsmodeller vi tidigare har behandlat. Exempelvis kolumnvis delning, radvis delning, kolumnvis sammanslagning m fl.

  Du ska ha dokumenterat vilka index du tänker använda för att optimera databasen. Alla index som du använder oavsett om det är för Pk, Fk eller andra index du anser din databas behöver.

- Frivillig uppgift. Med hjälp av ett (eller flera) script skapar du den nya databasen med sina nya tabeller och konverterar data in i de nya tabellerna från Medlem.

- Genomför tester mot den gamla databasen och den nya du har. Svarstiderna ska dokumenteras enligt tabellen i slutet av detta dokument. Testerna mot den nya databasen är frivillig.

Slutdokumentation ska innehålla

- Konceptuell datamodell.
- Fysisk datamodell med tabellprecisering och angivna index som du behöver ha. Eventuell denormalisering av databasen ska finnas dokumenterad.
- Dokumentation över de SQL-frågor som du har kört enligt uppgifter under **Tester**. Tillsammans med SQL satserna ska du redovisa uppgifter från executive plan och svarstider.

Uppgifterna 2, 3-4 och 7 är obligatoriska. Uppgift 5 och 6 är frivilliga och ger extra poäng som påverkar ditt betyg.

OBS!

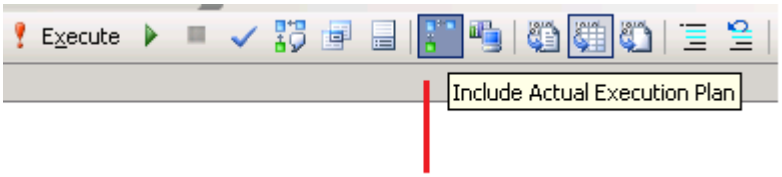
Det är viktigt att du själv reflekterar över tider och sätt att skriva SQL-satser på. Du ska inte enbart skriva en SQL sats och utvärdera resultat utan att det ger rätt värden i retur. Du ska också fundera över om svarstid och därmed kostnad för SQL server är förnuftig eller för krävande.

I vår labbmiljö så kommer en del svarstider kanske vara alldeles för lika jämfört ute i det verkliga livet där också en hel del andra parametrar påverkar. Det kan vara fragmentering på disk/diskar, överföringar via olika komponenter i nätverk etc. Vi använder virtuella maskiner vilket snabbar på en del. Arbetar du på distans mot skolan så förväntas svarstiderna bli längre.

2. Inställningar (obligatoriskt)

Öppna ett nytt Query fönster i SSMS och gör följande samt läsa av förändringar:

1. Kör följande SQL sats:
   SELECT enamn
   FROM DemoMedlem.dbo.Medlem
2. Studera resultatet under Message.
3. Kör följande SET – inställning:
   SET STATISTICS IO ON
   GO
   och kör sedan SQL-satsen under punkt 1 ovan.
   Funktionen visar statistikuppgifter som visar anslutningar/läsningar av data.
4. Studera resultatet under Message.
5. Kör följande SET – inställning:
   SET STATISTICS TIME ON
   GO
   och kör sedan SQL-satsen under punkt 1 ovan.
   Funktionen visar statistikuppgifter med tider för olika händelser.
6. Studera resultatet under Message.
7. Klicka på knappen Include Actual Execution Plan så den blir aktiv:



   och kör sedan SQL-satsen under punkt 1 ovan.
8. Du har nu fått en ny flik som heter Execution Plan. Studera innehållet där.
9. Nu är du klar att köra dina tester.

3. **Tester** (obligatoriskt)

Vid varje SQL fråga så mäter du tiden och studerar Execution plan. Varje fråga kör du två gånger för att se om du får skillnad i tiderna från första till andra gången.

Kör SQL fråga enligt följande:

1. Som visar alla fält och alla poster. Notera första och andra körning.

2. Som endast visar fälten enamn, fnamn, postnr, ort
Jämför tiden med punkt 1.

3. Som visar alla fält och alla poster men som är sorterad på efternamn

4. Som visar alla fält och alla poster men som är sorterad på efternamn och förnamn. Jämför med punkt 1.

5. Som visar endast fälten enamn, fnamn, postnr, ort men som är sorterad på efternamn. Jämför med punkt 2.

6. Som visar endast fälten enamn, fnamn, postnr, ort men som är sorterad på efternamn och förnamn. Jämför med punkt 5.
Nu bestämmer du själv vilka jämförelser du bör göra för att se vad det kostar.

7. Som visar alla medlemmar som bor i Kalmar.

8. Som visar alla medlemmar som bor i en ort som börjar på **Kal**.

9. Som visar alla medlemmar som bor i en ort som börjar på **Kal** och som har ett efternamn där ”son” ingår i efternamnet att den är sorterad är sorterad på fastighetsnr.

10. Visar alla medlemmar som har annullerats under det sista året.

11. Som visar antalet annullerade under respektive år och ort. Här måste du alltså använda aggregatfunktioner och GROUP BY på både årtalet och på ortsnamnet.

12. Ta ut en lista (med en SQL sats) som endast innehåller ortsnamnet som finns registrerad. Ett ortsnamn ska vara unikt och får inte förekomma två gånger. Listan ska vara sorterad i ortsnamnsordning.

13. Ta ut en lista på medlemmar som inte har samma innehåll i telenr1 som i telenr2.

4. Design (obligatoriskt)

Din uppgift här är att skapa en ny design för databasen Demomedlem. För att komma åt att läsa data i DemoMedlem.dbo.Medlem öppnar du ett Query fönster och anropar tabellen med DemoMedlem.dbo.Medlem. Exempelvis:
SELECT * FROM DemoMedlem.dbo.Medlem

1. **Normalisera** och analysera din databas för att komma fram till hur din nya databas ska se ut.

   Generalisera och optimera (kolumnvis/radvis delning/sammanslagning) dina tabeller.

   Tänk på **fältlängder** så du inte har onödigt stora fält.

   Kontrollera om det finns fält som innehåller nullvärden – och då speciellt om det är på många poster i tabellen.

   Sist i detta dokument finns en företeckning över ursprungsfältens innehåll.

   OBS!
   Ska du göra uppgift 5.Konvertering så läs detta också:
   För att hålla ihop data vid fördelning till nya tabeller kan det vara bra att ta med ID från Medlem. När allt är fördelat till nya tabeller kan detta ID tas bort om du inte behöver ha det kvar.
2. Skapa konceptuell modell med tabellspecifikationer och fysisk modell med tabellspecifikationer så du har underlaget klart för dig hur det ska se ut.
3. Fundera över index som du behöver ha i databasen. Dokumentera i den fysiska modellen
4. Som resultat av detta ska du prestera en Konceptuell datamodell med tabellspecifikationer och Fysisk datamodell med tabellspecifikationer och indexförtydligande. Eventuella avvikelser ska vara dokumenterade. Denormaliseringar ska helst undvikas i annat fall ska de dokumenteras.

5. Konvertering (frivillig)

Din uppgift här är att skapa ett script som fördelar om data från Demomedlem till din nya egna databas med din gällande databasdesign.

1. Skapa en ny tom databas som blir din nya databas. Det kan du göra med SSMS. Det är naturligtvis helt OK att göra det med DDL. Namnsätt din nya databas enligt modellen:

[Klassbeteckning]_[konto]_[databasnamn]
Exempelvis: **WP11_xyb222zw_Medlem**

Öppna ett Query fönster som har din nya databas som default.

2. För att hålla ihop data vid fördelning till nya tabeller kan det vara bra att ta med ID från Demomedlem.dbo.Medlem. När allt är fördelat kan detta ID tas bort om du inte behöver ha det.
3. Bygg ett script för att skapa upp tabellerna, relationerna, index och som fördelar om data enligt din nya struktur.
4. Kör scriptet så du får över alla data till dina nya tabeller. Scriptet kommer att innehålla ett antal SQL satser (INSERT INTO). Det är viktigt att ID i den nya medlemstabellen inte har räknare från början eftersom värdet kommer att användas som Fk i andra tabeller.

6. Avslutande tester (frivillig)

Undersök huruvida din nya databasdesign ger dig bättre svarstider än innan ombyggnaden/konverteringen.

Om man jämför uppgift 1 i Test ovan med en sådan test efter uppdelningen så innebär det ett antal JOIN som måste användas. Hur påverkar det svarstiderna?

7. Redovisning (obligatoriskt)

För att blir godkänd på laborationen ska du lämna in följande uppgifter:

1. Konceptuell modell utan tabellspecifikation.

2. Fysik datamodell med tabellprecisering och angivna index som du behöver ha. Eventuell denormalisering av databasen ska finnas dokumenterad. Datamodellen ska vara renritad och tydlig.

3. Dokumentation som visar SQL-sats och svarstider för uppgifter under Tester. Har du gjort uppgift 6 tar du med dessa tider också.

4. Har du gjort uppgift 5 ska scriptet finnas tillgängligt i din databas. Om du har den hemma kan du packa ihop och skicka in den packade filen.

OBS! Data i tabellen nedan är endast exempel.

Uppg	**SQL sats**	**Svarstid (sek)** **1:a**	**2:a**
1	SELECT enamn, fnamn, gata FROM DemoMedlem	57	52
2	SELECT enamn, fnamn, gata FROM Medlem as m INNER JOIN Adress as a ON m.adressid=a.adressid	36	34

8. Tabellbeskrivning DemoMedlem.dbo.Medlem

Databasen innehåller personer, fiktiva, som kan sägas vara medlemmar i en förening som hanterar hyresgäster i fastigheter runt om i landet. Personen som blir medlem blir det oftast genom att en värvare söker upp personen. Värvaren får en ersättning för det arbetet. Avgift för medlemskapet betalas på olika sätt vilket anges i betalkod.

Tillsammans med nedan beskrivning och dina analyser förstår du säkert vad de olika fälten innehåller.

Fält	Förklaring
ID	Pk
Enamn	Efternamn
Fnamn	Förnamn
Conamn	C/O namn (namn på person man är inneboende hos)
Gata	Gataunamn
Gatunr	Gatunummer
Uppgang	Uppgångsbeteckning
Postnr	Postnummer
Ort	Ortsnamn
Anndatum	Annulleringsdatum för medlem som är
Betaltom	Datum för hur länge man betalt till och med medlemsavgift
Indatum	Inträdesdatum som medlem
Fastnr	Fastighetsnummer som medlem bor i
Fastnrgl	Fastighetsnummer - gammal beteckning
Telenr1	Telefonnummer - 1
Telenr2	Telefonnummer - 2
Epost	Epostadress
Enddatum	Datum när senaste ändring på medlemmen ägt rum
Vervkod	Värvarkod - anger hur värvningen har gått till. Varje siffra betyder något speciellt men vi vet inte vad.
Betalkod	En kod som anger sätt som man betalar medlemsavgift på
Inpabet	Datum för när senaste inbetalning ägt rum
Tidning	Om man ska erhålla medlemstidning
Andrahand	Hyr lägenhet i andra hand
Förhandl	Om gemensam förhandling för medlem av hyra ska ske
Medlkort	Anger om medlemskort har skickats till medlem
Huvkortdatum	När Huvudmedlemskort skickats till medlem.
Erskortdatum	Anger när och om ersättningskort har skickats (vilket görs om man tappar sitt huvudmedlemskort).
Vervbetdatum	Om och när föreningen har betalat avgift till värvare. Värvaren får en viss peng när personen blir medlem.