Theres Sundberg Selin, .NET23

Teorihandboken SQL

1. Vad finns det för olika databastyper och var är deras olika för- och nackdelar?

De olika typer av databaser som finns är kort och gott relationella databaser och icke relationella databaser eller NoSQL-databser som de också kan kallas.

Relationsdatabaser så finns det tydliga relationer mellan tabeller. Det är också enkelt att hantera iom att man kan använda SQL vilket är standardiserat. Relationsdatabaser är inte bara strukturerade, utan de ger också ett tydligt och visuellt sätt att se relationerna i databasen eftersom det representeras av ett schema. Nackdelarna med en relationsdatabas är att den är mindre flexibel att hantera olika datatyper och skalbarheten är inte lika enkel som noSQL.

NoSQL har också sina för och nackdelar. Eftersom noSQL innebär att man inte använder sig av SQL så innebär det också att man inte får fördelen med ett standardiserat språk. Det är också svårare att felsöka och är generellt mindre säker eftersom de inte är lika noga med dataintegritet och relationer. Men, det är enklare att skala, det kan också passa bättre för vissa typer av användningsområden, och det finns inte samma krav på skapandet heller.

2. Hur ser datastrukturer ut i de typer av databaser du beskrev ovan?

Data ligger i tabeller med rader och kolumner, och kolumner representerar attributer eller egenskaper och raderna är poster. Relationerna mellan dessa tabeller är också viktiga, primärnycklar är tex alltid unika och relationerna mellan olika tabeller knyter det hela samman. Vi använder oss också av normaliseringsprincipen som innebär att man ska se till att koden som finns är så effektiv som möjligt, alltså använder så lite redundans som möjligt.

I noSQL databaser så kan det bero på vilken typ av databas det handlar om. I mongodb så sparas det tex i dokument och varje dokument innehåller informationen om ett objekt. Men det beror helt på

3. Gör en lite mer fördjupad beskrivning av relationsdatabaser och SQL. Beskriv framför allt vad som gör det till just en *relations*databas, exempelvis primär och sekundär nyckel. Beskriv också vad SQL är för språk och hur det i grunden är uppbyggt

Vad som gör en relationsdatabas til just en relationsdatabas är att den innehåller tabeller med columner och rader, och det finns nyckjlar, så kallade primary keys och foreign keys. Tabellerna har en primary key, och denna nyckels värden ska vara unika för varje rad. Så om en tabell har 125 rader

så måste kolumnen med primärnyckeln innehålla 125 olika värden. En så kallad unik identifierare. En foreign key är en nyckel som importeras från en tabell till en annan, och då är det alltså i den mottagande tabellen som den är foreign. Foreign keys pekar alltid på en primär nyckel, så i den skickande tabellen är det en primasy key och där den landar så är det en foreign key.

SQL, eller structured query language, är ett standardiserat språk som bygger på att beskriver vad du vill göra och låter därefter databasen hämta eller hantera datan och bygger på att du deklarerar det du vill göra, vare sig det är att hämt ,radera eller förändra. Det är uppbyggt av olika delar som sköter olika saker. SQL används i olika relationsdatabashanterare.

4. Vad är databasmodellering (ER och EER) och varför gör vi en sådan?

ER och EER är ett sätt att modellera upp och visualisera hur en databas och dess relationer och allt vad det innebär. Detta görs av en del olika anledningar, tex för att det gör det enklare att utveckla en stabil, effektiv och bra databas eftersom man får en tydlig bild på hur databasen byggs upp och vart brister finns. Det gör det också enklare att förklara och göra sig förstådd för andra. Det gör det tydligare att se brister som kan finnas, och gör därför en säkrare databas. EER är en utökning av ER, då den innehåller aggregation, generalisering och specialisering över de saker som ER innehåller.

5. Vad är relationsalgebra? Ge något enkelt exempel på relationsalgebra och varför det används.

Relationsalgebra handlar om vad vi kan göra med tabellerna som finns i en databas. Tex i ett excel ark så kan man med hjälp av relationsalgebra bestämma vad som ska göras med dessa kolumner och rader, vad som ska välja och visas och så vidare. Vi använder det för att definera operationerna som styr vad som ska göras med databsen i fråga. Det är också så att om du förstår relationsalgebra så är det lättare att förstå SQL och kan därmed göra det till en bättre utvecklare. Det bidrar också till normaliseringsprincipen iom att det ger bra verktyg för att dra ner på koden.

6. Vad är Views? Varför kan dessa vara bra att använda och finns det några nackdelar?

Views är ett sätt att kolla på datan i din databas, och det används genom att du väljer de delar du vill se, och sedan skapas det en visuell representation av den datan som ser ut som ett table. Det finns fördelar och nackdelar med views, precis som med allt annat.

7. Vad är Stored procedures? Varför kan dessa vara bra att använda och finns det några nackdelar?

Stored procedures är något som fins i SQL som är som en återanvändbar kod. Det är en väldigt bra och effektiv funktion, men har som med allt annat här i världen för och nackdelar. Det är alltså

instruktioner som är skriven i SQL. Dessa instruktioner är bra eftersom återanvändbar kod innebär mindre kod, och om en tabell ändras så är det endast i the stored procedure man behöver ändra. Det är dock så att de kan vara specifikt skrivna för en plattform, och kan därför göra det svårare att byta plattform. Det är också komplext och gör det därför mer komplext att felsöka i din SQL.

8. Under kursen har vi kollat på två sätt att kommunicera med databaser från C#. Beskriv dessa två metoder lite kort.

ADO NET är ett bibliotek från Microsoft som ger möjlighet att hantera databaser via .NET kod. Den hanterar allt från anslutningen, till att läsa, redigera och uppdatera databasen. ADO NET stödjer blanda annat transactions vilket gör det enkelt att gruppera operationer och ger möjligheten till "rollback" om det inte skulle fungera att utföra önskade uppgifter.

Entity framework är ett ORM-verktyg (Object-Relational Mapping), också från Microsoft, som gör det möjligt för programmeraren att komma åt databsen via ett objektiorenterat programmeringssätt. Entity framework översätter databasen till programmeringsspråk och gör det möjligt för utvecklaren att jobba med objekten i koden istället för via detabasen i sig. EF stödjer både code first och database first, och ger dig möjlgiheten att använda dig utan LINQ.

9. Vad är några för och nackdelar med dessa olika metoder?

Fördelarna med ADO NET är att det är flexibelt, att det kan användas med olika typer av databaser och inte är bundet. Det ger bra kontroll över åtkomsten och den direkta kontrollen över SQL-en gör det också möjlgit att optimera presntanda(om man vet vad man gör dvs). Nackdelarna är att det ofta kräver mer kod, vilket i sig inte är supereffektivt, kan öka tiden som det tar att utveckla och göra det hela mer komplext. Det kräver också en manuell översättning från databas med tabeller och objektorienterad kod eftersom det inte hanterar data på samma sätt.

Entity framework har fördelarna att det går seamlessly ihop med LINQ, vilket är language-intergrated query, och det innebär att vi får fram en tydlig och läsnar syntsax för att kunna skicka frågor till databasen direkt från din kod. Det har också fördelen att minska koden, och så är det ju såklart en objektorienterad modell där tabellerna från databasen blir till objekt i koden, vilket gör det hela mycket lättare när man som utvecklare ska hantera och förändra saker. Entity framework är dock något lågnsammare, och något svårare. Det kan ta tid att lära sig hur man hanterar entity framework, och det kan absolut vara negativt om det är så att man tex är ny programmerare och ska lära sig, eller bara har svårt att lära sig.