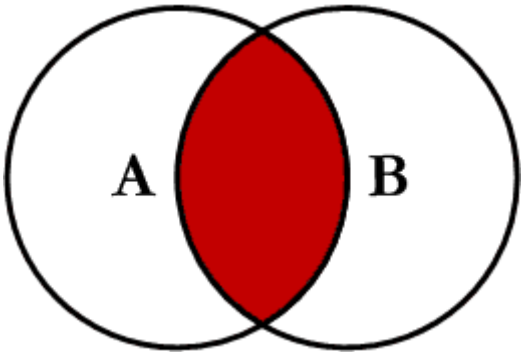
SQL

Join

INNER JOIN



INNER JOIN är den enklaste varianten av JOIN. Den är vanligast och kanske mest begriplig. Resultatet av en INNER JOIN ger poster i den vänstra tabellen (A) som har en matchande post i den högra tabellen (B).

Det är alltid nycklar vi använder för att para ihop data från två olika tabeller. Den vanligaste varianten är att man parar ihop en primärnyckel med en referensnyckelnyckel men det går även med andra varianter.

EX Vi vill se vilka datum en produkt(angiven med namn) har beställts. Då behövs både produkttabellen och prodLevtabellen. De två tabellerna har vars en kolumn vars värde är gemensmat. PID. Då kan vi knyta ihop dem med det värdet och få ett svar som är sant.

```
SELECT Produkt.PId,ProdLev.PId,Produkt.Namn, ProdLev.Datum
FROM ProdLev INNER JOIN Produkt on Produkt.PId= ProdLev.PId
```

Ger svaret:

```
P023        P023        Glitter                2015-02-15
P560        P560        Lamineringsfickor      2015-04-07
P235        P235        Akrylfärg              2015-06-18
P178        P178        Glitter                2015-06-17
P293        P293        Flörtkulor             2015-02-15
P293        P293        Flörtkulor             2015-05-04
P293        P293        Flörtkulor             2015-07-24
P275        P275        Glitter                2015-10-03
P319        P319        Snöre                  2015-01-10
P319        P319        Snöre                  2015-07-11
P415        P415        Band                   2015-11-01
P395        P395        Flörtkulor             2015-10-10
P057        P057        Band                   2015-02-15
P057        P057        Band                   2015-07-24
```

SQL

```
P166        P166        Kartong                 2015-09-15
P377        P377        Band                    2015-05-02
P122        P122        Kaligrafipenna          2015-02-15
P499        P499        Kartong                 2015-08-24
P499        P499        Kartong                 2015-01-22
P007        P007        Band                    2015-04-07
```

Det går även att knyta ihop fler än 2 tabeller så länge de har en kolumn vars värde överenstämmer. Tänk på att knyta ihop tabellerna i rätt ordning.

```
SELECT l.levId, pl.LevId, l.Namn, p.Namn, datum
FROM Leverantor l INNER JOIN ProdLev pl ON l.LevId=pl.LevId
INNER JOIN Produkt p ON p.PID=pl.PId
```

Ger resultatet:

levId	LevId	Namn	Namn	datum
L056	L056	Glitter och glam	Glitter	2015-02-15
L039	L039	Lek och stoj	Lamineringsfickor	2015-04-07
L163	L163	Konstnärsmaterial HB	Akrylfärg	2015-06-18
L056	L056	Glitter och glam	Glitter	2015-06-17
L039	L039	Lek och stoj	Flörtkulor	2015-02-15
L039	L039	Lek och stoj	Flörtkulor	2015-05-04
L056	L056	Glitter och glam	Flörtkulor	2015-07-24
L056	L056	Glitter och glam	Glitter	2015-10-03
L123	L123	Tåtar AB	Snöre	2015-01-10
L123	L123	Tåtar AB	Snöre	2015-07-11
L123	L123	Tåtar AB	Band	2015-11-01
L163	L163	Konstnärsmaterial HB	Flörtkulor	2015-10-10
L123	L123	Tåtar AB	Band	2015-02-15
L123	L123	Tåtar AB	Band	2015-07-24
L163	L163	Konstnärsmaterial HB	Kartong	2015-09-15
L123	L123	Tåtar AB	Band	2015-05-02
L163	L163	Konstnärsmaterial HB	Kaligrafipenna	2015-02-15
L085	L085	Kontoret AB	Kartong	2015-08-24
L163	L163	Konstnärsmaterial HB	Kartong	2015-01-22
L123	L123	Tåtar AB	Band	2015-04-07

SQL

Implicit syntax för INNER JOIN

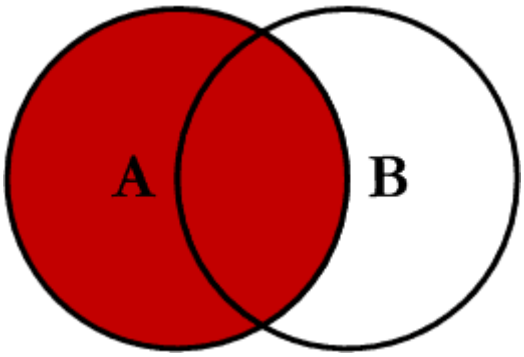
```
SELECT p.Namn,l.Namn
FROM Produkt p, ProdLev pl, Leverantor l
where p.PId= pl.PId AND pl.LevId =l.LevId
```

OUTER JOIN

Typ	**Behåller omatchade rader från**
LEFT OUTER JOIN	Den första(vänstra) tabellen
RIGHT OUTER JOIN	Den andra(högra) tabellen
FULL UOTER JOIN	Bägge tabellerna

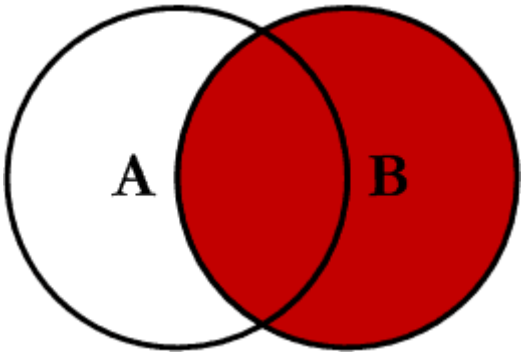
När vi använder LEFT eller RIGHT JOIN blir det av stor betydelse av i vilken ordning vi kombinerar tabellerna!

LEFT JOIN



Denna variant av JOIN ger oss alla poster i den vänstra tabellen (A) oavsett om det finns en matchande post i den högra tabellen (B). Den ger oss även data från den högra tabellen om det går att matcha.

RIGHT JOIN



Denna variant kommer att returnera alla poster i den högra tabellen (B) oavsett om det finns en matchning i den vänstra tabellen (A). Den kommer även att returnera alla matchande poster i den vänstra tabellen.

EX

```
SELECT Produkt.PId,ProdLev.PId,Produkt.Namn, ProdLev.Datum
FROM ProdLev RIGHT OUTER JOIN Produkt on Produkt.PId= ProdLev.PId
```

Ger resultatet

PId	PId	Namn	Datum
P023	P023	Glitter	2015-02-15
P156		Kopieringspapper	
P560	P560	Lamineringsfickor	2015-04-07
P235	P235	Akrylfärg	2015-06-18
P178	P178	Glitter	2015-06-17
P293	P293	Flörtkulor	2015-02-15
P293	P293	Flörtkulor	2015-05-04
P293	P293	Flörtkulor	2015-07-24
P275	P275	Glitter	2015-10-03
P319	P319	Snöre	2015-01-10
P319	P319	Snöre	2015-07-11
P415	P415	Band	2015-11-01
P395	P395	Flörtkulor	2015-10-10
P057	P057	Band	2015-02-15
P057	P057	Band	2015-07-24
P166	P166	Kartong	2015-09-15
P377	P377	Band	2015-05-02
P077		Glitter	
P122	P122	Kaligrafipenna	2015-02-15
P499	P499	Kartong	2015-08-24
P499	P499	Kartong	2015-01-22
P007	P007	Band	2015-04-07
P070		Snöre	

FULL OUTER JOIN

Ex

```
SELECT Produkt.PId,ProdLev.PId,Produkt.Namn, ProdLev.Datum
FROM ProdLev FULL OUTER JOIN Produkt on Produkt.PId= ProdLev.PId
```

Ger resultatet

SQL

PId	PId	Namn	Datum
P007	P007	Band	2015-04-07
P023	P023	Glitter	2015-02-15
P057	P057	Band	2015-02-15
P057	P057	Band	2015-07-24
P070		Snöre	
P077		Glitter	
P122	P122	Kaligrafipenna	2015-02-15
P156		Kopieringspapper	
P166	P166	Kartong	2015-09-15
P178	P178	Glitter	2015-06-17
P235	P235	Akrylfärg	2015-06-18
P275	P275	Glitter	2015-10-03
P293	P293	Flörtkulor	2015-02-15
P293	P293	Flörtkulor	2015-05-04
P293	P293	Flörtkulor	2015-07-24
P319	P319	Snöre	2015-01-10
P319	P319	Snöre	2015-07-11
P377	P377	Band	2015-05-02
P395	P395	Flörtkulor	2015-10-10
P415	P415	Band	2015-11-01
P499	P499	Kartong	2015-08-24
P499	P499	Kartong	2015-01-22
P560	P560	Lamineringsfickor	2015-04-07

Subquery

Det finns även en annan möjlighet att använda flera tabeller i samma fråga och det är via något som kallas subquery. I ett sökkriterium kan ingå att kontrollera om värdet finns i en annan tabell. Detta betyder att i ett sökkriterium kan ingå en annan SQL-fråga, som i sin tur kan bestå av en annan SQL-fråga i sitt sökkriterium. Att ha en SQL-fråga i sökkriteriet kallas för subquery.

Exempel: Skriv ut namnen på den produkt som har levererats i kvantitet om 160.

```
SELECT Namn
FROM Produkt
WHERE pId = (SELECT pid
             From ProdLev
             WHERE antal = 160)
```

Observera att för operatorerna ’=’, ’<’, ’<=’,’>’ ,’>=’, !=’ får underfrågan endast returnera ett värde. Hade det funnits två leveranser i ProdLev enligt exemplet, hade frågan blivit fel.

Operatorn IN fungerar dock bättre och underfrågan kan ge en eller flera svar. Dessutom kan IN kombineras med NOT.

Exempel: Skriv ut namnen på de leverantörer som har levererat en produkt i kvantiteten 20.

```
SELECT Namn
FROM Produkt
WHERE pId IN (SELECT pid
            From ProdLev
            WHERE antal = 20)
```

Underfrågorna kan gå i flera steg.

Exempel: Skriv ut namnen på de leverantörer som har levererat snöre.

```
SELECT Namn FROM Leverantor  WHERE LevId IN (SELECT LevId FROM ProdLev WHERE
Pid IN (SELECT PId FROM Produkt WHERE Namn= 'Snöre') )
```

Ibland är det tvunget att använda subquery.
Vad kommer det att bli för svar på frågan vem som INTE levererat snöre?

```
SELECT DISTINCT l.Namn From  Produkt p
            INNER JOIN ProdLev pl ON p.pid=pl.pId
            INNER JOIN Leverantor l on pl.LevId=l.LevId WHERE p.Namn!='Snöre'
```

Tåtar Ab slinker igenom då de även levererat annat. För att undvika använd:

```
SELECT Namn
FROM Leverantor
WHERE LevId  NOT IN
            (SELECT LevId
            FROM ProdLev
            WHERE Pid IN
                        (SELECT PId
                        FROM Produkt
                        WHERE Namn= 'Snöre') )
```

Vyer, views

Vyer fungerar nästan exakt som tabeller, men är resultatet av en fråga mot en eller flera tabeller eller vyer. Den data som vi kan se i en vy är aldrig fysiskt lagrad, utan sammanställs varje gång vi arbetar mot vyn. Vi kan då bl a använda vyer för att förenkla uttag av data från komplexa frågor. Istället för att varje gång ställa den komplexa frågan, som lätt kan bli fel, så

sparar vi frågan i form av en vy. Ett annat användningsområde är att filtrera bort viss information för en viss användare. Båda tillämpningarna är mycket användbara ur säkerhetssynpunkt. Det första exemplet kan göra att vi undviker att skriva en fråga felaktigt och därmed få konstiga resultat. Den andra tillämpningen är till stor nytta då vi har en grundtabell som innehåller vissa rader som bara får visas för vissa användare. Exempel: Vilka produkter(med namn) har levererats av vilken leverantör(med namn) där priset är 100 eller dyrar?.

```
SELECT Produkt.Namn, Leverantor.Namn
FROM Produkt INNER JOIN ProdLev ON Produkt.PId=ProdLev.PID
            INNER JOIN Leverantor ON Leverantor.LevId= ProdLev.LevId
            WHERE Pris>=100
```

Denna frågan kanske inte känns så komplicerad, men den har ett antal fällor. Vi kan t ex missa ett join-Vi gör därför en vy av den istället.

```
CREATE View Prod_Lev_Pris(ProdNamn, LevNamn, Pris) AS
            SELECT Produkt.Namn, Leverantor.Namn, pris
            FROM Produkt INNER JOIN ProdLev ON Produkt.PId=ProdLev.PID
            INNER JOIN Leverantor ON Leverantor.LevId= ProdLev.LevId
            WHERE Pris>=100
```

Vi kan nu få samma resultat genom att skriva:

```
SELECT *
FROM Prod_Lev_Pris
```

I det andra fallet tänker vi oss att vi har följande schema för tabellen KUNDER.

KUNDER (Id, Namn, Adress, Kontakt, Telefon, Rabatt)

Användaren Kalle arbetar mycket med kundinformation, men ska inte få se vilka rabatter som respektive kund har. Vi skapar en vy utan denna informationen.

```
CREATE VIEW KundInfo
AS
SELECT Id, Namn, Adress, Kontakt, Telefon
FROM KUNDER
```

OBS! Kalle behöver inte ha några rättigheter på tabellen Kunder, men har ändå rätt att göra SELECT på alla rader via vyn KundInfo.

Alter table

Det går att lägga till en kolumn till en tabell

Syntax:

ALTER TABLE Tabell

ADD Kolumn datatype

EX Lägg till Land på leverantör

```
ALTER TABLE Leverantor ADD Land char(25)
```

Det kommer då att bli NULL I den kolumnen för alla existerande rader

Det fungerar även att ta bort en kolumn

Syntax:

ALTER Table Tabell

DROP COLUMN kolumn

Ex Ta bort land

```
ALTER TABLE Leverantor DROP COLUMN Land
```

Defaultvärden

I exemplet har vi knutit ett standardvärde mot en kolumn. Det går även att knyta standardvärden mot användardefinierade datatyper. Alla kolumner som skapas med denna datatyp får då automatiskt detta standardvärde.

Skapa defaultvärde

```
CREATE DEFAULT LandDef AS 'Sverige'
```

Bind det till en kolumn

```
sp_bindefault 'LandDef','Leverantor.Land'
```

Ta bort ett defaultvärde

När man vill plocka bort ett default måste man först knyta loss det från de tabeller defaultvärdet är knutet till innan man kan plocka bort objektet fysiskt från databasen.

sp_unbindefault <defnamn>,<objnamn>

DROP DEFAULT <objnamn>

SQL

Regler, rules

I många fall har vi en mycket väldefinierad domän ur vilken värden till en kolumn kan tas, dess värdemängd. Detta kan vara mängden av alla godkända kundnummer, produktnummer, bokstäver, tal mellan 0 och 24 s v. Vi kan definiera värdemängden för en kolumn med hjälp av regler, rules.

Syntax:

CREATE RULE [<ägare>.]<regel_namn>
AS <villkorsuttryck>

Vi talar om detta med hjälp av ett villkorsuttryck som måste vara sant.

Exempel: Skapa en regel för produktnummer där det gäller att alla nummer har följande utseende; PXX där XX är två siffror.

CREATE RULE rule_PID AS
@Nr LIKE 'P[0-9][0-9][0-9]'

@Nr är en lokal variabel som får värdet i den kolumnen som vi knyter regeln mot. Regeln används vid INSERT och UPDATE. Regler knyts mot kolumner med sp_bindrule.

Syntax:

sp_bindrule <regel_namn>,<objektnamn>[,FUTUREONLY]

Exempel:

```
sp_bindrule rule_PId, 'Produkt.PID'
sp_bindrule rule_PId, 'ProdLev.PId'
```

Då vi försöker sätta in ett värde som bryter mot regeln, så får vi ett felmeddelande och förändringen rullas tillbaka.

Ta bort en regel

När man vill plocka bort en regel måste man först knyta loss den från de tabeller regeln är knuten till innan man tar bort själva objektet fysiskt från databasen.

sp_unbindrule <defnamn>,<objnamn>

DROP RULE <objnamn>