Webbserverprogrammering 1

Table of Contents

1. Introduktion till databaser

1.1. Vad är en databas?

En databas är en välorganiserad, det vill säga uppbyggd enligt nedskrivna regler, centraliserad, samling av data.

Data *vs* **information**.

Information är behandlad data - det vill säga data, som någon kan använda för att tillföra något slags värde i någon slags process eller sammanhang.

Man kan säga att data blir information när någon tittar på eller behandlar den.

1.2. Databashanterare / Database Management System

En Databashanterare/Databashanteringssystem, eller DBMS (Database Management System), är mjukvara som gör det möjligt att på ett effektivt och säkert sätt definera och modifiera innehåll i en databas.

Man kan inte kommunicera direkt med databasen; all kommunikation sker via databashanteraren.

Det finns i dag två huvudsakliga typer av databaser: relationsdatabaser (vilket är vad vi kommer använda i den här kursen) och dokument/nod-baserade (berörs inte i den här guiden). Andra ord som används för att skilja på dessa två typer av databaser är SQL (relationella) och NoSQL (dokument/nod-baserade).

Ordet "relation" är taget från matematiska begreppet "relation" - som betyder just tabell.

Relationsdatabaser bygger på relationsalgebra. Som tur är behöver man inte förstå all bakomliggande teori för att kommunicera med en databashanterare.

Det finns flera olika databashanterare för relationsdatabaser, t.ex SQLite, MySQL, MariaDB, PostgreSQL, Oracle Database, Microsoft SQL Server.

I stort sett alla applikationer använder i någon grad en databas - t.ex. lagrar webbläsare sin historik, chattprogram sina meddelanden och schoolsoft dina betyg i databaser.

Världens överlägset mest använda databashanterare (och den vi kommer använda i större delen av kursen) är SQLite, som finns installerad på alla iOS-enheter, Android-enheter, och används i både Chrome och Firefox.

1.3. Varför använda en relationsdatabas

Det finns flera anledningar till att använda en (relations)databas:

1.3.1. Normalisering av data

Normalisering, eller normaliserad data innebär att man minskar förekomsten av redundant data.

Redundant data innebär att samma data kan finnas på flera ställen samtidigt. Detta tar mer plats än om datan finns på enbart ett ställe.

1.3.2. Konsistent data

Om samma data finns på flera ställen är det stor risk att datan blir *inkonsistent*, det vill säga, beroende på vilket ställe man kollar kan man få olika svar.

Genom normalisering kan man förhindra eller minska risken för inkonsistent data.

1.3.3. Enkel delning av data

Flera applikationer kan använda samma databas, och därmed använda samma data.

1.3.4. En väldokumenterad standard för informationsutbyte

SQL är ett väldokumenterat språk som väldigt många utvecklare behärskar. Det är därför lätt för nya utvecklare att sätta sig in i ett nytt system.

1.3.5. Säkerhet och integritet

De flesta databashanterare innehåller avancerade kontroller av vem som har rättighet att utföra olika operationer i databasen.

1.3.6. Validering av data

I de flesta databashanterare kan man skriva regler för hur rader och kolumner i tabeller ska se ut.

T.ex kan man bestämma att alla rader i användartabellen måste innhålla ett användarnamn, och att varje användarnamn måste vara unikt. Databashanteraren gör det i så fall omöjligt att skriva rader i användatabellen som saknar användarnamn, eller där användarnamnet redan finns på någon annan rad i tabellen.

1.3.7. Transaktionssäkerhet

Ofta behöver man utföra operationer i flera, av varandra beroende steg.

Säg till exempel att du skriver mjukvara för att hålla koll på bankkonton, och du vill föra över pengar från ett konto till ett annat konto. Då måste du:

1. Ta bort pengarna från konto 1.
2. Föra in pengerna på konto 2.

Alternativt

1. Föra in pengarna på konto 2.

2. Ta bort pengarna från konto 1.

Säg att något går fel i steg 2 - kanske har någon angett ett felaktigt kontonummer, eller under tiden som instruktionerna utförs har konto 2 tagits bort. I ett system utan transaktionssäkerhet skulle pengarna bara ha försvunnit i tomma intet (eller i alternativ 2, skapats från tomma intet)

I de flesta databashanterare finns därför `transaktioner` - om man lägger sina instruktioner i en transaktion kollar databasen att samtliga instruktioner i transaktionen går att utföra innan de faktiskt utförs. Skulle felet i steg 2 hända i en transaktion skulle steg 1 aldrig utföras - och pengarna skulle finnas kvar.

2. Tabeller och relationer

En relationsdatabas består av en eller flera `tabeller`. Varje tabell har ett namn, och består av `rader` och `kolumner`.

Table 1. Exempel på en tabell för att hålla koll på böcker

books		
id	**title**	**page_count**
1	'Catch 22'	464
2	'To Kill a Mockingbird'	336
3	'1984'	328
4	'The Stranger'	123

Tabellen ovan har namnet `books`, och har 4 rader och 3 kolumner.

2.1. Tupler

Varje rad i tabellen kallas för en `tupel` och beskriver en `post` i tabellen. I vårt exempel lagrar varje tupel information om en bok.

2.2. Attribut

Varje kolumn beskriver en egenskap, eller `attribut` för posten. I vårt fall är "id", "title" och "page_count" attribut.

Attribut måste vara `atomära`, det vill säga i varje "cell" eller "fält" får det maximalt finnas ett värde - man kan t.ex inte lagra flera titlar i en och samma cell.

2.3. Primärnyckel

Varje rad i en tabell måste vara unik, det vill säga, i varje tabell får det inte förekomma två rader där samtliga kolumner är identiska (detta är en följd av relationsalgebran).

Detta löses genom att varje tabell har en attribut som kallas för `primärnyckel` eller `primary key`. I tabellen ovan är det attributet med namnet `id` som är primärnyckel

När man skapar tabellen talar man om vilken av attributen som kommer vara primärnyckeln.

Detta innebär att om någon annan skulle skriva en bok som heter the stranger, och som råkar ha 123 sidor även den kommer raden fortfarande vara unik, eftersom den kommer få en unikt värde på primärnyckeln

Table 2. Varje tupel är fortfarande unik, trots att det finns två böcker med samma titel och sidantal.

books		
id	**title**	**page_count**
1	'Catch 22'	464
2	'To Kill a Mockingbird'	336
3	'1984'	328
4	'The Stranger'	123
5	'The Stranger'	123

Det är databashanterarens roll att kontrollera att varje tupel är unik.

Namngivning av primärnyckeln

I den här boken kommer vi:

- **alltid** använda namnet **id** för primärnyckeln, men primärnyckeln *kan* heta vad som helst.
- **alltid** använda **automatiskt inkrementerande heltal** för primärnyckeln, men primärnyckeln *kan* vara vad som helst.

2.4. Normalisering

Säg att vi vill även vill lagra författarens namn. Vi lägger därför till en kolumn:

books			
id	**title**	**page_count**	**author**
1	'Catch 22'	464	'Joseph Heller'
2	'To Kill a Mockingbird'	336	'Harper Lee'
3	'1984'	328	'George Orwell''
4	'The Stranger'	123	'Albert Camus'

Inga problem, men säg att vi även vill lagra författarens nationalitet, födelseår och skostorlek.

books						
id	**title**	**page_count**	**author**	**nationality**	**birth_year**	**shoe_size**
1	'Catch 22'	464	'Joseph Heller'	'American'	1923	42
2	'To Kill a Mockingbird'	336	'Harper Lee'	'American'	1926	36
3	'1984'	328	'George Orwell'	'English'	1903	41
4	'The Stranger'	123	'Albert Camus'	'French'	1913	44

Nu börjar tabellen se lite konstig ut: Tabellen heter "books" men innehåller attributen "nationality", "birth_year" och "shoe_size". De låter inte som attribut (egenskaper) en bok har. Vi kan förstås lägga in "author_" framför varje attribut som är kopplat till en författare snarare än en bok:

books						
id	**title**	**page_count**	**author**	**author_nationality**	**author_birth_year**	**author_shoe_size**
1	'Catch 22'	464	'Joseph Heller'	'American'	1923	42
2	'To Kill a Mockingbird'	336	'Harper Lee'	'American'	1926	36
3	'1984'	328	'George Orwell'	'English'	1903	41
4	'The Stranger'	123	'Albert Camus'	'French'	1913	44

Det känns bättre, men det är fortfarande konstigt att om man vill veta vilken skostorlek en författare har så ska man kolla i boktabellen. Men vad händer när vi börjar lagra flera böcker för en författare?

books

id	title	page_count	author	author_nationality	author_birth_year	author_shoe_size
1	'Catch 22'	464	'Joseph Heller'	'American'	1923	42
2	'To Kill a Mockingbird'	336	'Harper Lee'	'American'	1926	36
3	'1984'	328	'George Orwell'	'English'	1903	41
4	'The Stranger'	123	'Albert Camus'	'French'	1913	44
5	'Closing Time'	382	'Joseph Heller'	'American'	1923	42
6	'Animal Farm'	218	'George Orwell'	'English'	1903	41
7	'The Plague '	312	'Albert Camus'	'French'	1913	44
8	'Coming Up for Air'	393	'George Orwell'	'English'	1903	41

Nu förekommer varje författares nationalitet, födelseår och skostorlek flera gånger - vi har med andra ord redundant data i databasen.

Vi vill i största möjliga utsträckning undvika redundant data, och behöver därför `normalisera` datan, genom att dela upp datan i **två** tabeller - **books** och **authors**

authors				
id	name	nationality	birth_year	shoe_size
1	'Joseph Heller'	'American'	1923	42
2	'Harper Lee'	'American'	1926	36
3	'George Orwell'	'English'	1903	41
4	'Albert Camus'	'French'	1913	44

books

id	title	page_count
1	'Catch 22'	464
2	'To Kill a Mockingbird'	336
3	'1984'	328
4	'The Stranger'	123
5	'Closing Time'	382
6	'Animal Farm'	218
7	'The Plague '	312
8	'Coming Up for Air'	393

Nu har vi normaliserat datan; data om författare i en tabell, och data om böcker i en annan. Ingen data är duplicerad. Dessvärre vet vi inte längre vilken författare som skrivit vilken bok.

2.5. Relationer

För att knyta samman två tabeller behöver vi skapa en `relation`, det vill säga, en koppling, mellan dem.

I exemplet med böcker och författare behöver vi skapa en `en-till-många-relation` mellan authors och books - *en* för fattare kan ha skrivit *många* böcker (men en bok kan bara ha en författare).

2.6. Främmande nyckel

För att skapa en relation mellan två tabeller använder man en `främmande nyckel`. Man skapar en främmande nyckel genom att kopiera in värdet från `primärnyckeln` i `en-änden` av relationen i en ny kolumn i `många-änden` av relationen.

I vårt fall innebär det att vi ska lägga till primärnyckeln från **author**-tabellen i **books**-tabellen (en för fattare kan ha skrivit flera böcker).

books			
id	title	page_count	author_id
1	'Catch 22'	464	1
2	'To Kill a Mockingbird'	336	2
3	'1984'	328	3
4	'The Stranger'	123	4

5	'Closing Time'	382	1
6	'Animal Farm'	218	3
7	'The Plague '	312	4
8	'Coming Up for Air'	393	3

Namngivning av den främmande nyckeln

Den främmande nyckelns kolumn kan heta precis vad som helst, men i den här boken kommer den främmande nyckelns kolumn-namn **alltid** döpas enligt följande: **namnet på en-ändens tabellnamn *i singular*** följt av ett understreck och sen **id (author_id)**

2.7. Övningar

2.7.1. Datorregister

Alla elever på IT-Gymnasiet Göteborg får låna en dator. Det finns ett inventeringssystem med en databas där alla elever, datorer och lån finns registrerade.

Systemet håller koll på:

- Datorers modell och serienummer
- Elevers namn och personnummer
- Vilken elev som har lånat vilken dator

Rita upp de tabeller och relationer (inklusive främmande nycklar) som databasen behöver. Fyll i ett par rader i varje tabell.

2.7.2. Matkort

I årskurs tre får eleverna på IT-Gymnasiet Göteborg matkort. Företaget som skapar och säljer matkorten till skolorna har (förhoppningsvis) en databas som håller koll på skolor, kort, och låntagare.

De behöver kunna hålla kolla på:

- Skolors adress och kontaktperson (så de kan skicka ut korten till rätt adress).
- Korts serienummer och pinkod
- Elevers namn och klass
- Vilket kort som tillhör vilken elev
- Vilken skola och klass en elev går i

Rita upp de tabeller och relationer (inklusive främmande nycklar) som databasen behöver. Fyll i ett par rader i varje tabell.

3. ER-diagram

För att snabbare kunna modellera databaser finns Entity Relationship Diagrams (ER-diagram).

ER-diagram illustrerar en databas logiska uppbyggnad, det vill säga vilka tabeller, attribut och relationer som finns.

3.1. Entitet

Entiteter representerar tabeller i databasen. Entiteter ritas som rektanglar, med namnet (i singular) i mitten.

Figure 1. Två entiteter

3.2. Attribut

Attribut representerar ett attribut på en tabell. Attribut ritas som en ovalel, med namnet i mitten. Alla attribut tillhör någon entitet, och man drar ett streck mellan entiteten och attributet för att visa vilken entitet ett attribut tillhör.

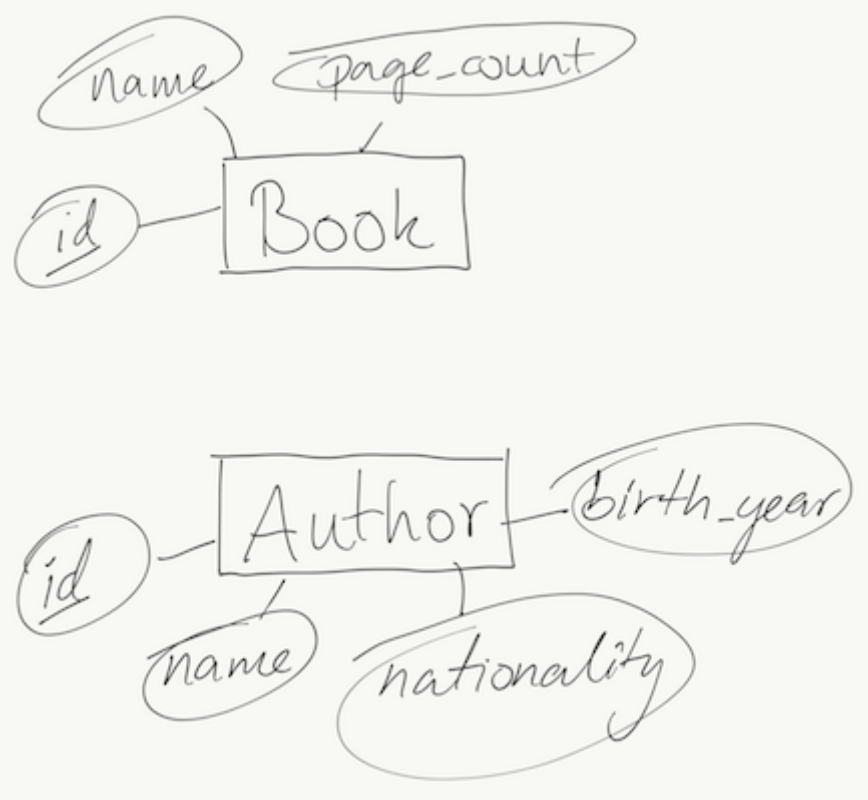

Figure 2. Två entiteter med attribut

`Primärnycklarna` är alltid understrukna.

Främmande nycklar

Främmande nycklar ska aldrig ritas ut i ER-diagrammet, deras placering framgår av relationerna (se nästa rubrik)

3.3. Relation

Relationer visar på kopplingar mellan två entiteter. Relationer ritas som romber. I mitten av romben står ett ett eller flera ord som beskriver relationen (oftast från ena entitetens perspektiv). Varje relation är kopplad med streck till de ingående entiteterna. I varje ände av strecket framgår relationens `kardinalitet`, som låter oss förstå om det t.ex. är en en-till-många-relation.

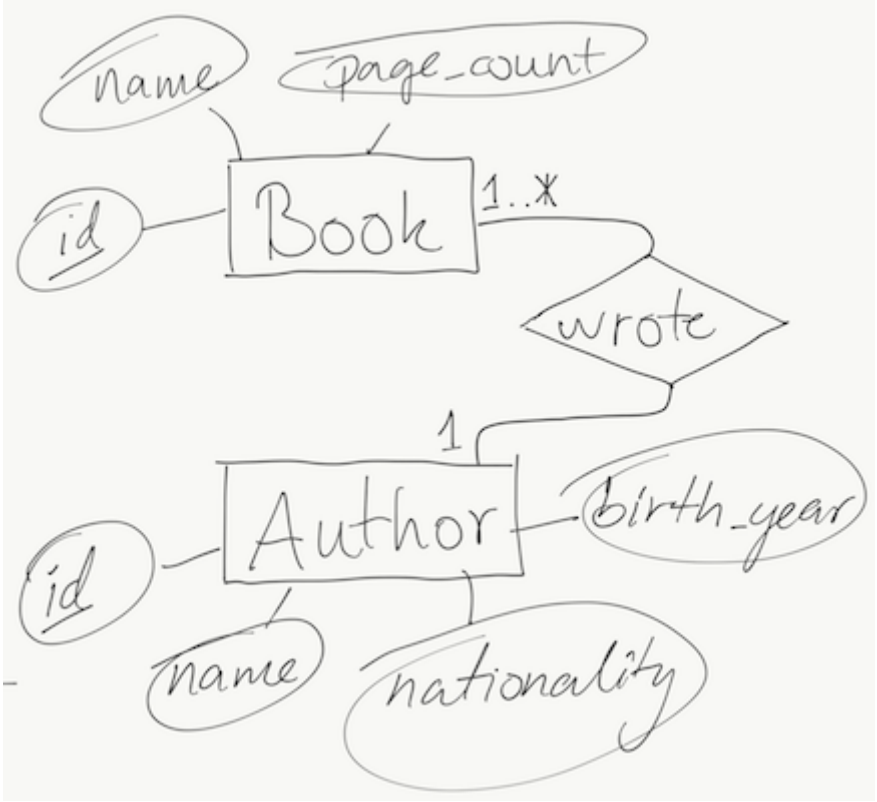

Figure 3. Två entiteter med attribut och relation

I exemplet ovan kan man utläsa att **en** författare kan skriva **många** (men minst 1) böcker, men en bok kan bara ha en författare. Det är med andra ord en en-till-många-relation.

Relationen är i exemplet ovan skrivet från författarens perspektiv ("wrote"), men skulle lika gärna kunna vara skriven från en boks perspektiv (t.ex "written by" eller "belongs to").

Från relationens kardinalitet kan vi utläsa att det i databasens books-tabell ska skapas en främmande nyckel med namnet "author_id"

4. Databasmodellering med ER-diagram

I det här kapitlet ska vi med hjälp av ER-diagram och ett par frågor modellera en virtuell bokhylleapplikation.

Den virtuella bokhyllan ska ska hålla koll på en användares alla titlar.

- En titel kan tillhöra flera olika genres (t.ex. "Fantasy" och "Science Fiction").
- En titel har ett namn, ett sidantal, och är skriven av en författare.
- En författare kan ha skrivit flera titlar.
- En genre kan innehålla flera titlar.
- Användaren ska kunna
  - Se alla titlar i sin bokhylla,
  - Skriva anteckningar om en titel,
  - Söka bland titlar i sin bokhylla

- Se sina anteckningar om en specifik titel.

4.1. Identifiera substantiv

Ett trick man kan använda för att identifiera sina entiteter är att leta efter substantiv i beskrivningen av sin problemdomän.

Beskrivningen ovan innehåller följande substantiv: "bokhylla" "användare", "titel", "genre", "namn", "sidantal", "författare", "anteckning".

Vissa substantiv kommer bli våra entiteter, andra kommer att bli attribut våra entiteter har, och vissa kan vi ignorera.

I vårt fall kan vi ignorera substantivet "bokhylla", eftersom hela applikationen är "bokhyllan". I vårt fall kanske hela applikationen skulle kunna heta t.ex "Bookshelf"

4.2. Frågor för att skilja entiteter från attribut

Första steget är att räkna ut vilka av ens substantiv som blir entiteter (och, i förlängningen, tabeller i databasen), och vilka som blir attribut på entiteterna.

Frågor man kan ställa för att klura ut vilka substantiv som bör bli egna entiteter är:

1. Har substantivet flera saker som skulle kunna vara attribut på det?
2. Har substantivet verb kopplade till sig (utöver skapa/ta bort/uppdatera)?
3. Om substantivet är ett attribut på en (eller flera) annan entitet, skulle det i så fall förekomma på flera rader i den entitetens tabell?
4. Om substantivet är ett attribut på en (eller flera) annan entitet, skulle varje rad i den entitetens tabell kunna ha flera av substantivet?

Om svaret är "Ja" på en eller flera av dessa frågor är det troligt att det ska bli en entitet.

Vi går igenom Bokylleapplikationens substantiv och ställer frågorna till dem.

4.2.1. Användare

Har användare flera saker som skulle kunna vara attribut?

Ja, användare har (antagligen) email, användarnamn, lösenord och information om när hen senast var inloggad.

Har användare verb kopplade till sig?

Ja, användare kan logga in/ut, och skriva anteckningar. En användare kan antagligen också lägga till/ta bort böcker.

Om användare var attribut på en annan entitet, skulle de i så fall förekomma flera i den entitetens tabell?

Ja, om användare var attribut på t.ex titel-entiteten skulle de förekomma flera gånger i titel-tabellen, eftersom en användare kan äga flera titlar.

Table 3. Title-tabell där samma användardata dupliceras på olika rader

id	name	isbn	user_name	user_pwd	user_last_login
1	"Grillboken"	123	"Grill"	pwd1	2017-08-01
2	"BBQ Book"	234	"Grill"	pwd1	2017-08-01
3	"Würstbuch"	345	"Korv"	pwd2	2017-09-23
4	"Kokboken"	456	"Korv"	pwd2	2017-09-23

Title-tabellen innehåller duplicerad data om användaren (rad 1 & 2 och rad 3 & 4), vilket vi vill undvika.

Om användare var attribut på en annan entitet, skulle det i så fall kunna finnas flera användare på varje rad i den entitetens tabell?

Ja, om användare var attribut på t.ex. en titel skulle varje titel kunna ha flera användare, eftersom flera användare kan äga samma titel.

Table 4. Title-tabell där varje rad innehåller data om flera användare

id	name	isbn	user_names	user_pwds	user_last_logins
1	"Grillboken"	123	"Grill", "Korv", …	…, …, …	2017-08-01, 2017-09-23, …

En cell i en tabell får endast innehålla ett värde, så det här fungerar inte.

Slutsats Användare:

Samtliga frågor har besvarats med "Ja". Användare bör därmed vara en egen entitet.

4.2.2. Titel

Har titlar flera saker som skulle kunna vara attribut?

Ja, en titel har namn, isbn, författare, utgivningsår, genres, sidantal, anteckningar

Har titlar verb kopplade till sig?

Nej, inte utöver skapa/ta bort/uppdatera

Om en titel var attribut på en annan entitet, skulle de i så fall förekomma på flera rader i den entitetens tabell?

Ja, om titlar var attribut på användare så skulle varje titel förekomma på flera rader, eftersom flera användare kan ha samma titel:

Table 5. User-tabell där samma titel-data dupliceras på olika rader

id	username	pwd	title_name	title_isbn	title_pagecount
1	"Grill"	pwd1	"Grillboken"	123	1337
2	"Korv"	pwd2	"Grillboken"	123	1337
3	"Senap"	pwd3	"BBQ Book"	234	42
4	"Ketchup	pwd4	"Grillboken"	123	1337

Grillbokens information är dublicerad i tabellen, vilket vi vill undvika.

Om titlar var attribut på författare skulle det kunna fungera (men se fråga 4 nedan):

Table 6. Author-tabell med titel som attribut (ingen duplicering)

id	name	country	title_name	title_isbn	title_pagecount
1	"F. Örfattare"	"Sweden"	"Grillboken"	123	1337
2	"A. Uthor"	"Great Britain"	"BBQ Book"	234	42
3	"V. Erfasserin"	"Germany"	"Würstbuch"	345	7331

Om en titel var ett attribut på en annan entitet, skulle de i så fall kunna förekomma flera gånger på en rad?

Ja, om en titel var ett attribut på författare eller användare skulle varje författares eller användares rad behöva ha flera titlar.

Table 7. Author-tabell där varje rad innehåller data om flera titlar

id	name	country	title_names	title_isbns	title_pagecounts
1	"F. Örfattare"	"Sweden"	"Grillboken", "Kokboken", …	123, 456, …	1337, 21, …
2	"A. Uthor"	"Great Britain"	"BBQ Book" , "Cookbook", ….	234, 567, …	42, 512, …

Table 8. User-tabell där varje rad innehåller data om flera titlar.

id	username	pwd	title_names	title_isbns	title_pagecounts
1	"Grill"	pwd1	"Grillboken", "Kokboken", …	123, 456, …	1337, 21, …
2	"Korv"	pwd2	"BBQ Book" , "Cookbook", …	234, 567, …	42, 512, …

Båda tabellerna ovan innehåller celler med mer än ett värde, vilket inte är tillåtet.

Slutsats

Flera av frågorna har besvarats med "Ja". Titel bör vara en egen entitet

4.2.3. Genre

Har genres flera saker som skulle kunna vara attribut?

Nja, en genre har antagligen bara ett attribut: namn.

Har genres verb kopplade till sig?

Nej, inte utöver skapa/ta bort/upddatera.

Om en genre var attribut på en annan entitet, skulle de i så fall förekomma på flera rader i den entitetens tabell?

Ja, om genres var entiteter på t.ex. titlar skulle varje genre förekomma flera gånger i tabellen, eftersom en genre kan ha många titlar

Table 9. Title-tabell där samma genre-data dupliceras på olika rader

id	name	isbn	pagecount	genre
1	"Grillboken"	123	1337	"Grillning"

id	name	isbn	pagecount	genre
2	"BBQ Book"	234	42	"Grillning"
3	"Würstbuch"	345	7331	"Korv"

I tabellen ovan dupliceras datan om genre, men eftersom det enbart är ett enstaka fält, skulle det kunna vara OK.

Om en genre var attribut på en annan entitet, skulle de i så fall förekomma flera gånger på samma rad?

Ja, om genre var ett attribut på en titel skulle genren förekomma flera gånger på varje titels rad, eftersom varje titel kan ha flera genres

Table 10. Titel-tabell där varje rad innehåller information om flera genres

id	name	isbn	pagecount	genres
1	"Grillboken"	123	1337	"Matlagning", "Grillning"
2	"BBQ Book"	234	42	"Matlagning", "Grillning"
3	"Würstbuch"	345	7331	"Matlagning", "Korv"

En cell i en tabell får endast innehålla ett värde, så det här fungerar inte.

Slutsats:

Flera av frågorna har besvarats med "Ja". Genre bör vara en egen entitet

4.2.4. Namn

Har namn flera saker som skulle kunna vara attribut?

Nej (eventuellt skulle en titel kunna ha olika namn på olika språk).

Har namn verb kopplade till sig?

Nej, inte utöver skapa/ta bort/uppdatera

Om namn var attribut på en annan entitet, skulle de i så fall förekomma på flera rader i den entitetens tabell?

Nej (teoretiskt sett kan det finnas titlar med samma namn, men det är väldigt ovanligt).

Om namn var attribut på en annan entitet, skulle de i så fall förekomma flera gånger på samma rad?

Nej. En titel har bara ett namn (återigen: eventuellt kan namnet kan finnas på flera språk).

Slutsats:

Samtliga fyra frågor besvarades med "Nej". Namn bör vara ett attribut (på titel).

4.2.5. Sidantal

Har sidantal flera saker som skulle kunna vara attribut?

Nej.

Har sidantal verb kopplade till sig?

Nej, inte utöver skapa/ta bort/uppdatera.

Om sidantal var attribut på en annan entitet, skulle de i så fall förekomma på flera rader i den entitetens tabell?

Nja, det finns antagligen flera böcker som har samma sidantal. Men det skulle inte innebära någon vidare duplikation av data.

Om sidantal var attribut på en annan entitet, skulle de i så fall förekomma flera gånger på samma rad?

Nej. En titel har bara ett sidantal (olika utgåvor av en bok kanske har olika sidantal dock).

Slutsats:

Ingen av frågorna besvarades med "Ja". Sidantal bör vara ett attribut (på titel).

4.2.6. Författare

Har författare flera saker som skulle kunna vara attribut?

Ja, förnamn, efternamn, nationalitet, födelseår.

Har författare verb kopplade till sig?

Nej, inte utöver skapa/ta bort/uppdatera.

Om författare var attribut på en annan entitet, skulle de i så fall förekomma på flera rader i den entitetens tabell?

Ja, om författare var ett attribut på en titel skulle författaren förekomma på flera rader i titeltabellen eftersom författaren kan skriva flera böcker.

Table 11. Title-tabell där samma författar-data dupliceras på flera rader

id	name	isbn	pagecount	author_name	author_country
1	"Grillboken"	123	1337	"F. Örfattare"	"Sweden"

id	name	isbn	pagecount	author_name	author_country
2	"BBQ Book"	234	42	"A. Uthor"	"Great Britain"
3	"Würstbuch"	345	7331	"V. Erfatterin"	"Germany""
4	"Kokboken"	456	512	"F. Örfattare"	"Sweden"

I tabellen ovan dupliceras datan om varje författare, vilket vi vill undvika.

Om författare var attribut på en annan entitet, skulle de i så fall förekomma flera gånger på samma rad?

Nja. Författare skulle kunnna vara ett attribut på en titel, eftersom en titel bara har en författare (i vår applikation, men det kan vara önskvärt att bygga ut funktionalitet för att tillåta flera författare på en bok - det är inte ovanligt att författare samarbetar).

Table 12. Title-tabell där samma författar-data dupliceras på olika rader

id	name	isbn	pagecount	author_name	author_country
1	"Grillboken"	123	1337	"F. Örfattare"	"Sweden"
2	"BBQ Book"	234	42	"A. Uthor"	"Great Britain"
3	"Würstbuch"	345	7331	"V. Erfatterin"	"Germany""

I tabellen ovan finns det bara en författare per rad, vilket skulle kunna vara OK (men se fråga 3).

Slutsats:

Flera av frågorna har besvarats med "Ja". Författare bör vara en entitet

4.2.7. Anteckning

Har anteckningar flera saker som skulle kunna vara attribut?

Ja, en användare. Och kanske en titel, och datum den är skapad?

Har anteckningar verb kopplade till sig?

Nej, inte utöver skapa/ta bort/uppdatera.

Om anteckning var attribut på en annan entitet, skulle de i så fall förekomma på flera rader i den entitetens tabell?

Nej, om anteckning var ett attribut på användare eller titel skulle antagligen varje anteckning fortfarande vara unik (eventuella dubletter skulle vara ett sammanträffande).

Table 13. Title-tabell med anteckning som attribut

id	name	isbn	pageount	user_note	user_id
1	"Grillboken"	123	1337	"This book smells funny"	1
2	"BBQ Book"	234	42	"This book has a weird taste"	2
3	"Würstbuch"	345	7331	"I like turtles!"	1

I tabellen ovan dupliceras ingen data, men se fråga 4.

Om anteckning var attribut på en annan entitet, skulle de i så fall förekomma flera gånger på samma rad?

Ja. En om anteckning är ett attribut på användare eller titel skulle varje rad i användar- eller titel-tabellen behöva innehålla flera anteckningar, eftersom både titlar och användare kan ha flera anteckningar.

Table 14. Title-tabell där varje rad innehåller data om flera anteckningar

id	name	isbn	pageount	user_notes	user_ids
1	"Grillboken"	123	1337	"This book smells funny", "To Grill or not to Grill", …	1, 2, …
2	"BBQ Book"	234	42	"This book has a weird taste", "BBQ is Life!", …	2, 3, …

Tabellen ovan innehåller flera anteckningar per cell, vilket inte är tillåtet.

Slutsats:

Flera av frågorna har besvarats med "Ja". Anteckning bör vara en entitet.

4.3. Entiteter med exempelattribut

Baserat på slutsatserna kan vi skapa en skiss över våra entiteter med exempelattribut:

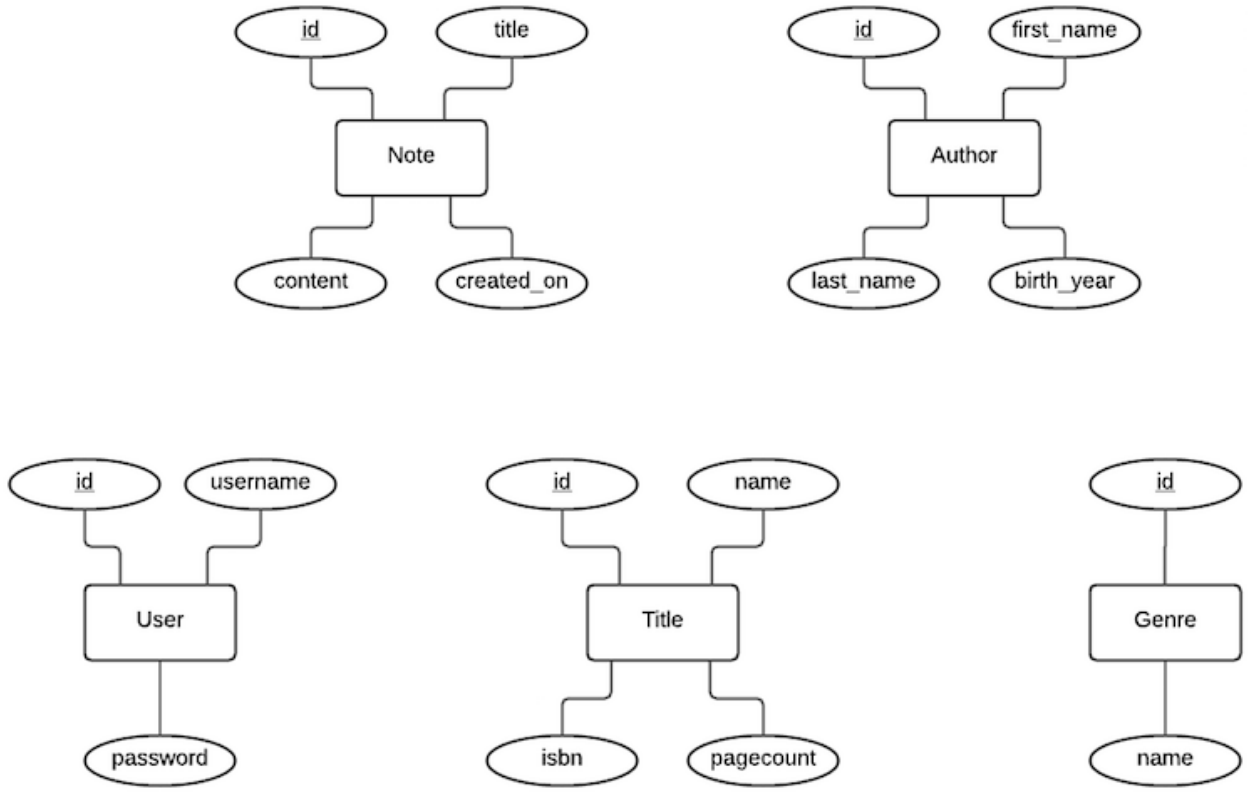

Figure 4. Exempelentiteter med attribut

4.4. Associationer & Kardinalitet

Nästa steg är att markera kardinaliteten mellan entiteterna.

4.4.1. En-till-många-relationer

En titel kan ha många anteckningar, men varje anteckning tillhör bara en titel.

I ER-diagrammet markeras en en-till-många-relation med hjälp av en etta (1) i "ett-änden" av relationen, och en asterisk (*) i många-änden av relationen:

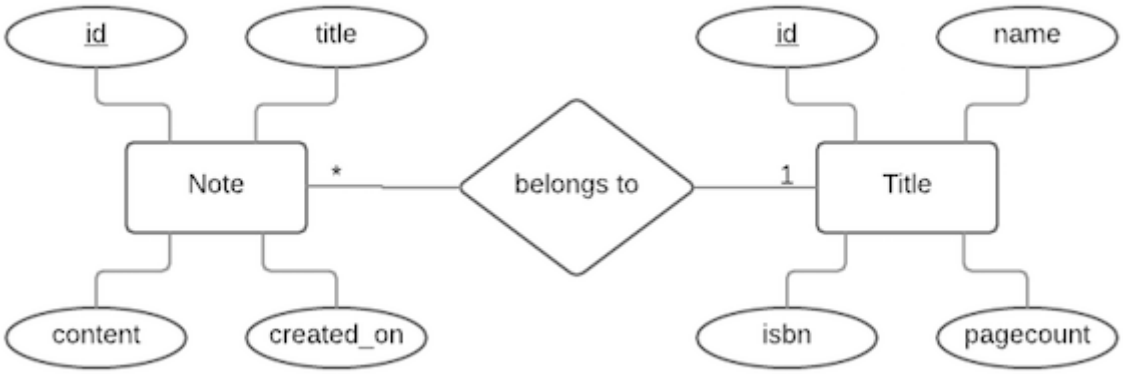

Figure 5. En-till-många-relation mellan title och note

I databasen kommer relationen skapas genom att det i många-ändens-tabell läggs till en främmande nyckel som innehåller en-ändens-primärnyckel.

Table 15. Title-tabellen ("en-änden" i relationen).

id	name	isbn	pagecount
1	"Grillboken"	123	1337
2	"BBQ Book"	234	42
3	"Würstbuch"	345	7331

Table 16. Note-tabellen ("många-änden" i relationen).

id	content	title_id
1	"This book tastes weird"	1
2	"This book smells funny"	1
3	"I Like turtles!"	2

Vissa entiteter, som t.ex anteckning, kommer vara i en-änden av flera relationer: En användare kan ha flera anteckningar, men en anteckning tillhör bara en användare.

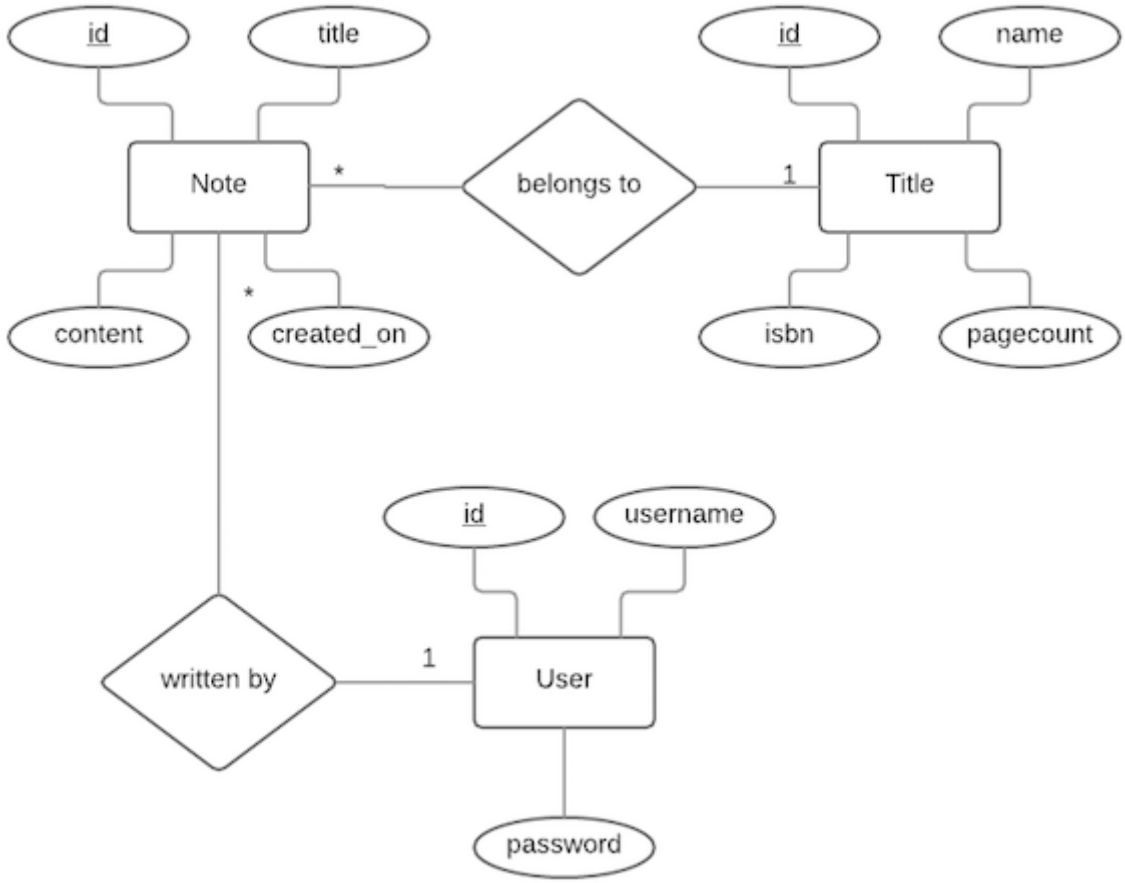

Figure 6. En-till-många-relationer mellan note och title och note och user

Table 17. User-tabell ("en-änden" av relationen)

id	username	pwd
1	"Grill"	pwd1
2	"Korv"	pwd2
3	"Senap"	pwd3
4	"Ketchup"	pwd4

I databasen kommer notes-tabellen ha både "user_id" och "title_id":

Table 18. Note-tabellen ("många-änden" i **två** *relationer).*

id	**content**	**user_id**	**title_id**
1	"This book tastes weird"	1	1
2	"This book smells funny"	1	2
3	"I Like turtles!"	2	1

4.4.2. Många-till-många-relationer

En titel kan tillhöra många genres, och varje genre kan ha flera titlar.

I ER-diagrammet markeras en många-till-många-relation med hjälp av en asterisker (*) i båda-ändarna av relationen:

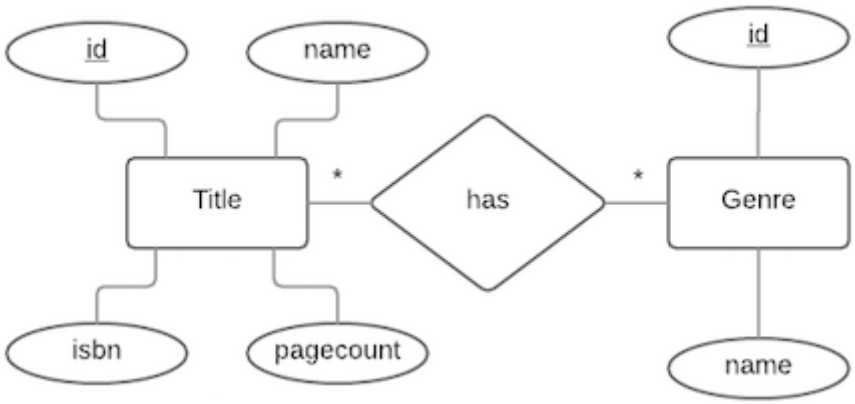

Figure 7. Många-till-många-relation mellan genre och titel

Många till-många-relationer går inte att skapa i databastabeller, utan kräver en extra tabell; en så kallad *relationstabell*.

En relationstabell är en extra tabell som ligger mellan de två ingående entiteterna och omvandlar relationen till två stycken en-till-många-relationer istället. Om man inte kan komma på vad man ska döpa relationen kan man döpa den till de två ingående entiteternas namn (i vårt fall "Genre-Titles")

Table 19. Title-tabellen (ena "en-änden" i många-till-många-relationen).

id	name	isbn	pagecount
1	"Grillboken"	123	1337
2	"BBQ Book"	234	42
3	"Würstbuch"	345	7331

Table 20. Genre-tabellen (andra "en-änden" i många-till-många-relationen).

id	name
1	"Matlagning"
2	"Grillning"
3	"Korv"

Table 21. Genre-Title-tabellen (relationstabellen i många-till-många-relationen)

title_id	genre_id
1	1
1	2
2	1
2	2
3	3

5. SQL

För att kommunicera med relationella databashanterare används databasspråket SQL (Structured Query Language).

SQL kan uttalas "ess-ku-ell" (på svenska), "es-que-ell" (på engelska), som Engelska ordet "sequel" eller som "seek well".

SQL skiljer sig från programmeringsspråk som t.ex Ruby eller Javascript genom att man definerar vilken information man vill ha, och lämnar upp till databashanteraren att avgöra **hur** den på bästa (snabbaste och säkraste) sätt ska hämta efterfrågad data.

Ruby - rad 8-14 beskriver **hur** *datan ska hämtas (i det här fallet med hjälp av en while-loop).*

```
titles = [ {name: 'BBQ Book', isbn: '123', page_count: 1337},
           {name: 'Grillboken', isbn: '234', page_count: 42},
           ...
         ]

long_titles = []

i = 0
while i < titles.length
    if titles[i][:page_count] > 500
        long_titles << books[i]
    end
    i += 1
end

p long_titles
```

SQL - **hur** *databashanteraren ska hämta den efterfrågade informationen framgår inte i frågan*

```
SELECT * FROM titles WHERE page_count > 500; ①
```

① Förutsatt att det i databasen finns en tabell vid namn titles

6. Vagrant

Vagrant är ett program som underlättar att bygga och starta virtuella operativsystemsmaskiner.

I den här kursen använder vi Vagrant för att snabbt kunna installera, konfigurera, och starta virtuella linuxmaskiner att köra våra webbapplikationer på.

6.1. Virtualiseringsstöd i BIOS

1. Stäng av datorn (starta **inte** om).
2. Starta datorn igen. Följ sedan instruktionerna nedan:

När du ser texten *To Interrupt normal startup, press Enter*, tryck på `Enter` .

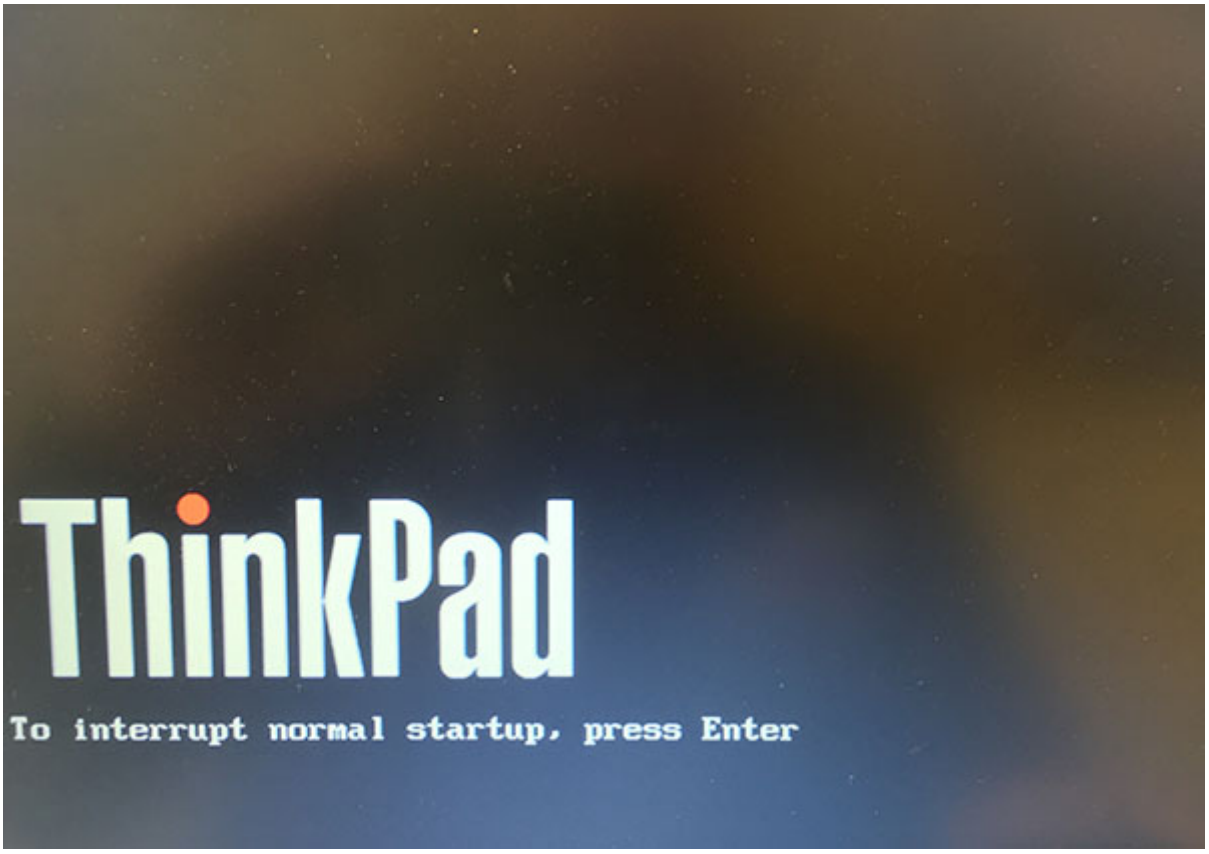

Figure 8. BIOS Interrupt

Om datorn går vidare till Windows-uppstarten får du stänga av datorn och försöka igen.

När du ser *Startup Interrupt Menu* trycker du på F1.

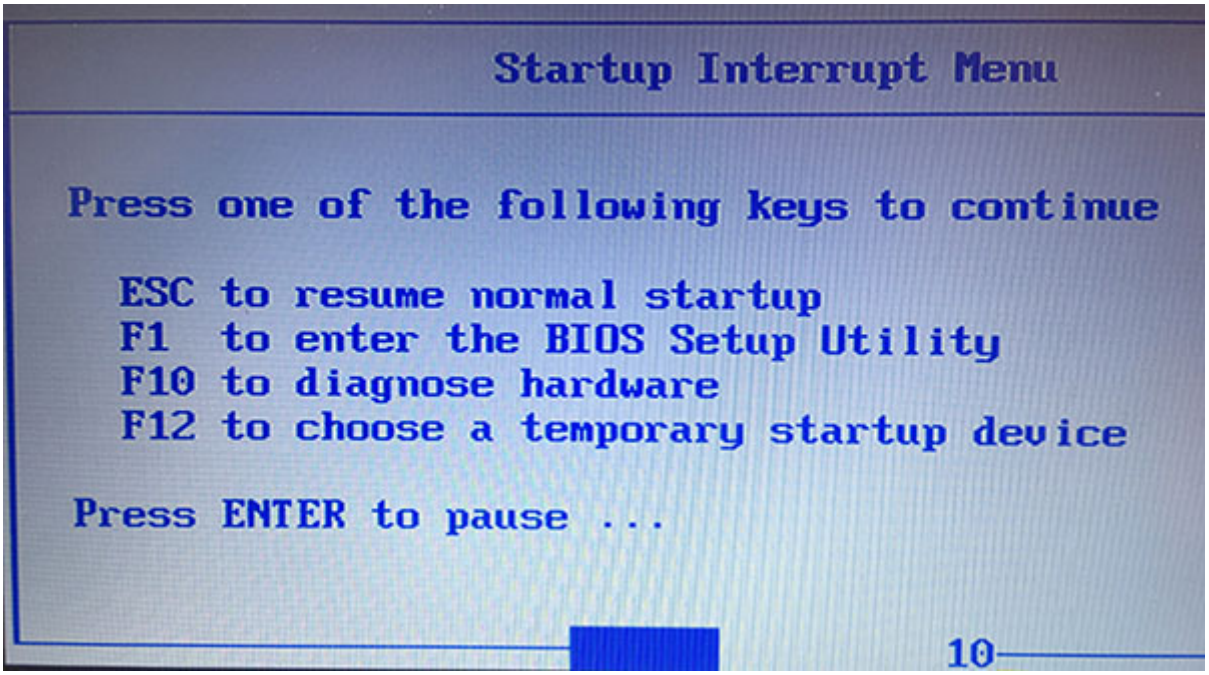

Figure 9. Startup Interrupt Menu

Du får nu se se *Main-fliken i bios*:

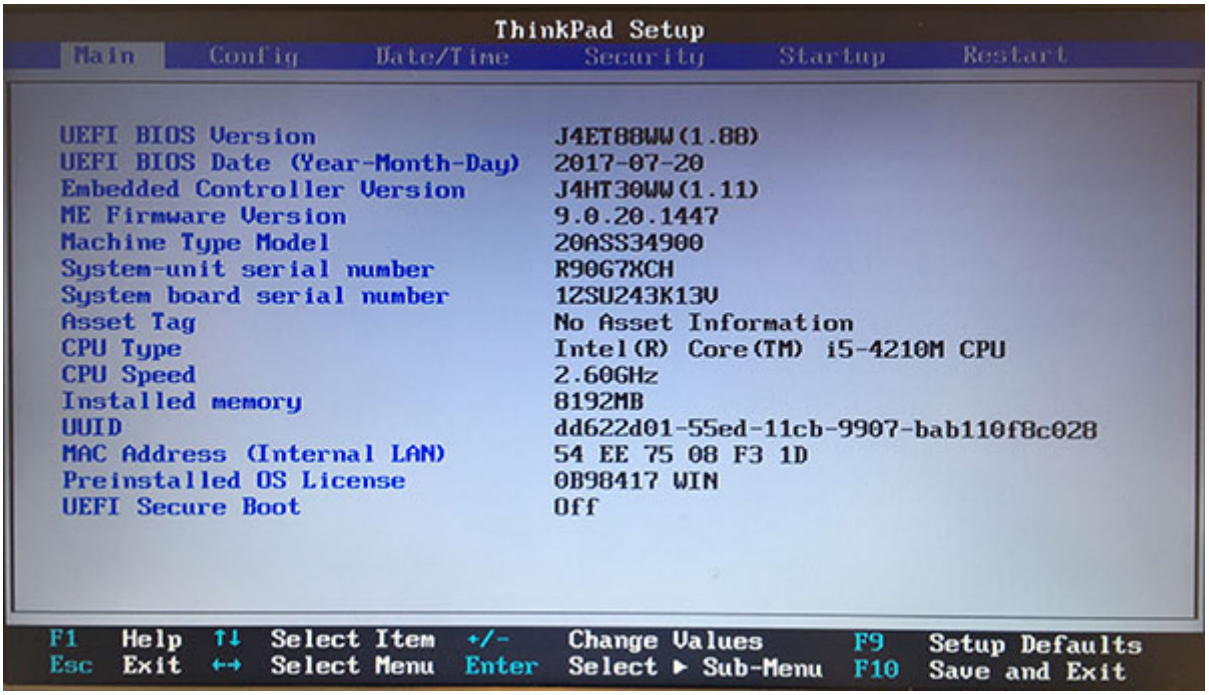

Figure 10. Main-fliken i BIOS

Använd piltangenterna ◀▶ för att navigera till *Security*.

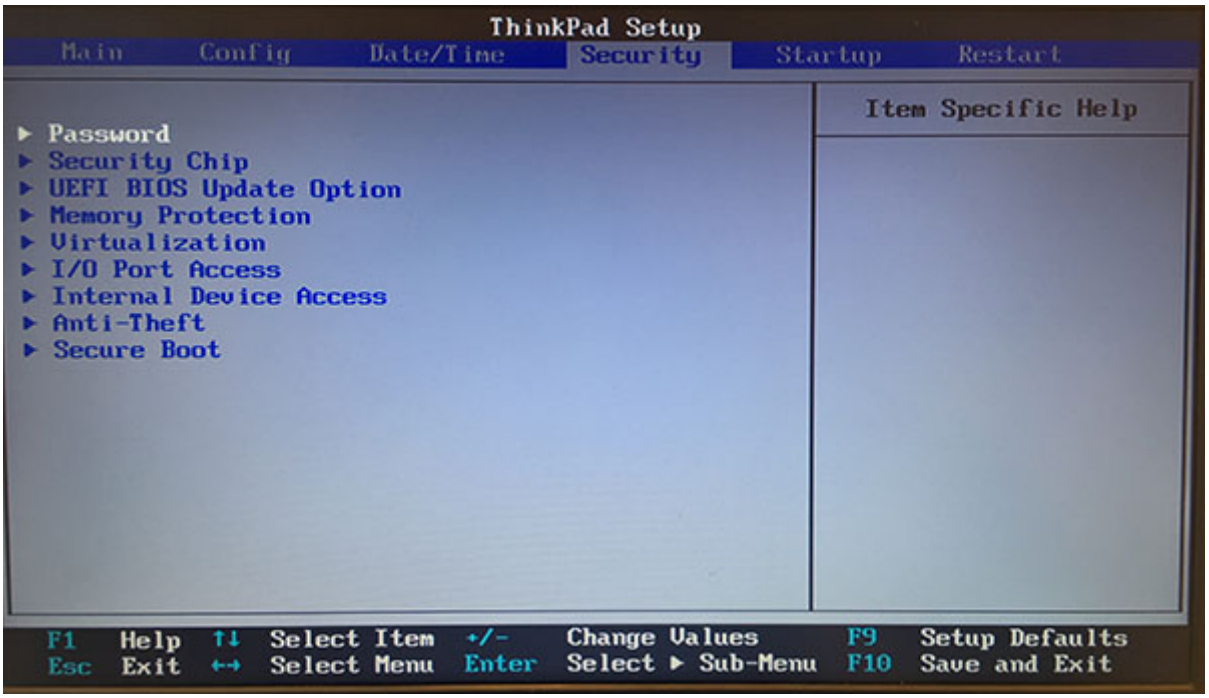

Figure 11. Security-fliken i BIOS

Använd piltangenterna ▲▼för att markera *Disabled*. Tryck på Enter .

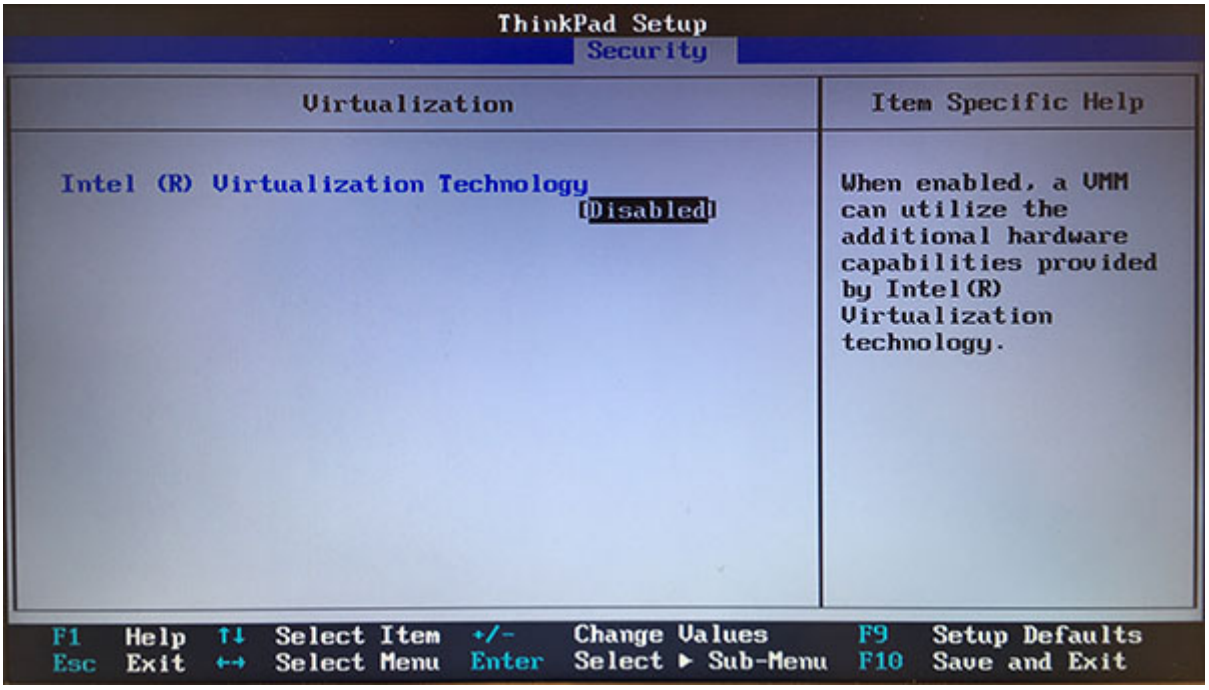

Figure 12. Intel Virtualization Technology Disabled

Använd sen piltangenterna ▲▼för att välja *Enabled*, och sen `Enter` igen.

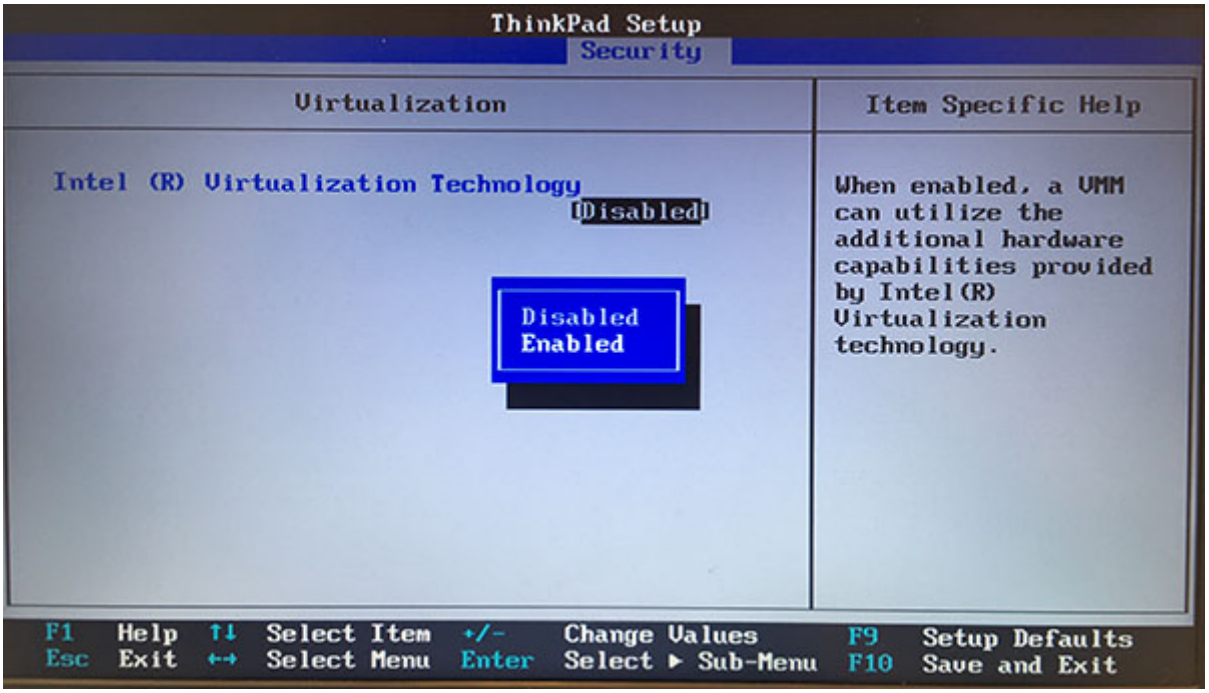

Figure 13. Intel Virtualization Technology Enabled

Spara ändringarna i BIOS genom att trycka på `F10`.

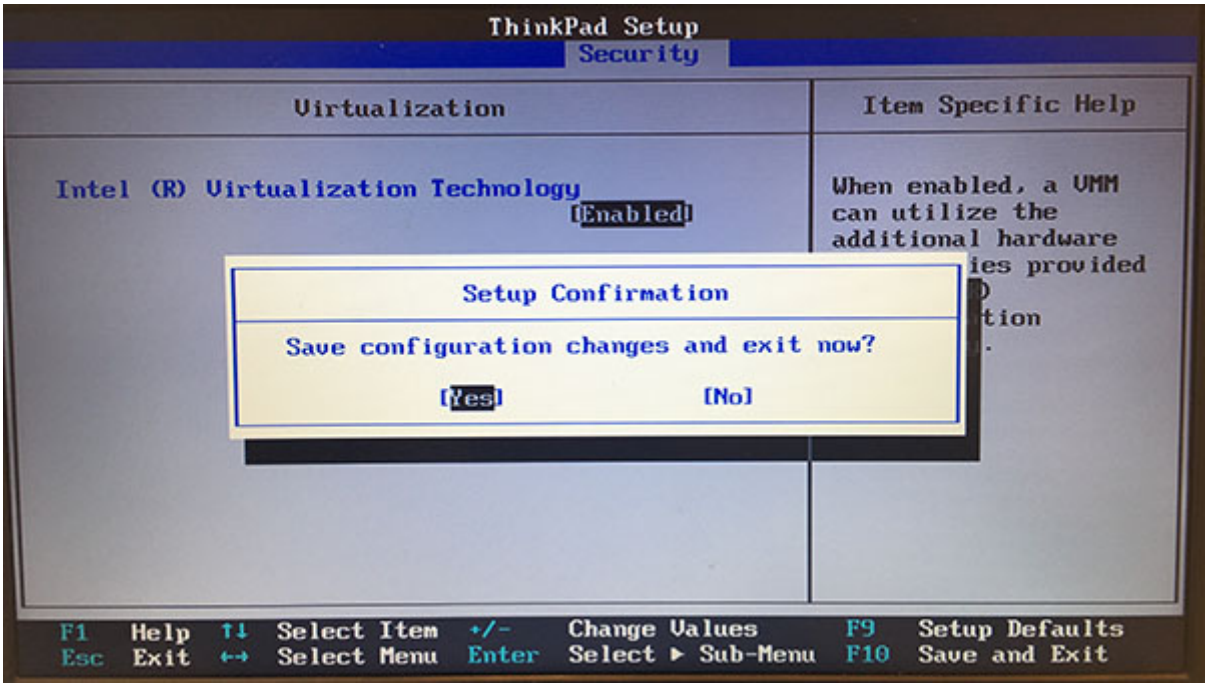

Figure 14. Setup Confirmation

Välj *Yes* med piltangenterna ◀▶ och tryck på `Enter` . Datorn startar nu om.

6.2. Registrering av virtuell maskin-fil.

För att registrera den virtuella maskinen så Vagrant kan använda den behöver vi importera filen till vagrant. (filen som innehåller maskinen finns med i imagen för din dator (i mappen `c:\vagrant_boxes`)).

I terminalen, skriv

```
vagrant box add c:\vagrant-boxes\namnet-på-filen.box --name ubuntu-server ① ②
```

① Byt ut `namnet-på-filen` mot filens riktiga namn

② Värdet efter `--name` är namnet den virtuella maskinen kommer registreras som i vagrants "databas".

Vagrant kommer nu packa upp filen och registrera den i sin databas.

6.3. Skapande av virtuell maskin

Skapa en mapp för din webapplikation. Navigera till mappen i terminalen och skriv:

```
vagrant init ubuntu-server ①
```

① byt ut `ubuntu-server` mot namnet du registrerade din virtuella maskin med.

Vagrant skapar nu en `Vagrantfile` i mappen du körde kommandot:

```
A `Vagrantfile` has been placed in this directory. You are now
ready to `vagrant up` your first virtual environment! Please read
the comments in the Vagrantfile as well as documentation on
`vagrantup.com` for more information on using Vagrant.
```

6.4. Konfigurering av nätverk för den virtuella maskinen

För att kunna komma åt webbservern på din virtuella maskin behöver du konfigurera nätverket.

Öppna `Vagrantfile` i en editor.

Leta reda på raden `# config.vm.network "public network"` och avkommentera den.

```
...
  # Create a public network, which generally matched to bridged network.
  # Bridged networks make the machine appear as another physical device on
  # your network.
  config.vm.network "public_network" ①
...
```

① Ta bort kommentaren på den här raden.

När din virtuella maskin bootar kommer den få en egen ip-adress.

Alla (på samma nätverk) kan komma åt den virtuella maskinen med hjälp av dess ip-adress.

6.5. Starta den virtuella maskinen

I samma mapp som `Vagrantfile`, skriv

```
vagrant up
```

Den virtuella maskinen bootas och du kommer se en utskrift i terminalen liknande denna:

```
==> default: Importing base box 'ubuntu-server'...
Progress: 30%
```

När maskinen är importerad kommer den fråga vilket nätverkskort den ska använda:

```
==> default: Matching MAC address for NAT networking...
==> default: Setting the name of the VM:
intro_vagrant_och_sinatra_default_1506500026362_91339
==> default: Clearing any previously set network interfaces...
==> default: Available bridged network interfaces:
1) en5: Thunderbolt Ethernet Slot 1
2) en0: Wi-Fi (AirPort) ①
3) en1: Thunderbolt 1
4) en2: Thunderbolt 18
5) bridge0
6) p2p0
7) awdl0
==> default: When choosing an interface, it is usually the one that is
==> default: being used to connect to the internet.
    default: Which interface should the network bridge to?
```

① Välj alternativet som verkar vara ditt wifi-nätverskort.

Efter du valt nätverkskort fortsätter maskinen att boota:

```
==> default: Preparing network interfaces based on configuration...
    default: Adapter 1: nat
    default: Adapter 2: bridged
==> default: Forwarding ports...
    default: 22 (guest) => 2222 (host) (adapter 1)
==> default: Booting VM...
==> default: Waiting for machine to boot. This may take a few minutes...
    default: SSH address: 127.0.0.1:2222
    default: SSH username: vagrant
    default: SSH auth method: private key
==> default: Machine booted and ready!
==> default: Checking for guest additions in VM...
==> default: Configuring and enabling network interfaces...
==> default: Mounting shared folders...
    default: /vagrant => /Users/daniel.berg/Documents/Kurser/Webbserverprogrammering
1/intro_vagrant_och_sinatra
```

Om du vill kan du dubbelkolla att maskinen faktiskt körs:

```
vagrant status
```

Vilket förhoppningsvis ger följande utskrift:

```
Current machine states:

default                   running (virtualbox)

The VM is running. To stop this VM, you can run `vagrant halt` to
shut it down forcefully, or you can run `vagrant suspend` to simply
suspend the virtual machine. In either case, to restart it again,
simply run `vagrant up`.
```

6.6. Logga in

För att kontrollera den virtuella maskinen kommer du använda dig av SSH; **S**ecure **SH**ell:

```
vagrant ssh
```

Du loggas nu in i den virtuella maskinen, och möts av en linuxterminal:

```
Welcome to Ubuntu 16.04.3 LTS (GNU/Linux 4.4.0-62-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

56 packages can be updated.
30 updates are security updates.

Last login: Wed Aug  9 15:43:57 2017 from 10.0.2.2
[Oh My Zsh] Would you like to check for updates? [Y/n]: ①
    ~
```

① Tryck på `N` (du kan uppdatera nästa gång om du vill).

6.7. Delad mapp

Vagrant delar automagiskt innehållet i mappen med `Vagrantfile` mellan Windows och linux.

Detta innebär att du kan jobba med filerna i Windows, och webbservern som körs i den virtuella linuxmaskinen kan komma åt filerna. I den virtuella maskinen hittar du den delade mappen i `/vagrant`

"dir" i den delade mappen i Windows

```
C:\Users\admin\Documents\Webbserverprogrammering\min_webapp>dir
 Volume in drive C has no label.
 Volume Serial Number is 24BD-F54D

 Directory of C:\Users\admin\Documents\Webbserverprogrammering\min_webapp

2017-09-27  12:50    <DIR>          .
2017-09-27  12:50    <DIR>          ..
2017-09-27  12:49               166 config.ru
2017-09-27  12:49                50 Gemfile
2017-09-27  12:49                63 min_webapp.rb
2017-09-27  12:47              3018 Vagrantfile
               4 File(s)              0 bytes
               2 Dir(s)  15 661 158 400 bytes free

C:\Users\admin\Documents\Webbserverprogrammering\min_webapp>
```

"ls -la" i /vagrant i den virtuella maskinen

```
   /vagrant ls -la
total 20
drwxr-xr-x  1 vagrant vagrant  238 Sep 27 12:55 .
drwxr-xr-x 24 root    root    4096 Aug  9 15:15 ..
-rw-r--r--  1 vagrant vagrant  166 Sep 26 09:26 config.ru
-rw-r--r--  1 vagrant vagrant   50 Sep 26 09:26 Gemfile
-rw-r--r--  1 vagrant vagrant   63 Sep 26 09:24 min_webapp.rb
drwxr-xr-x  1 vagrant vagrant  102 Sep 27 10:13 .vagrant
-rw-r--r--  1 vagrant vagrant 3018 Sep 27 10:13 Vagrantfile
```

6.8. Logga ut

För att logga ut från den virtuella maskinen skriver du:

```
   exit
Connection to 127.0.0.1 closed.
```

När uppkopplingen stängts är du tillbaks i Windows.

6.9. Stänga av

För att stänga av den virtuella maskinen skriver du i maskinens mapp i Windows:

```
vagrant halt
==> default: Attempting graceful shutdown of VM...
```

7. Linux

7.1. Navigation i filsystemet

För att navigera i filsystemet använder du följande kommandon:

Observera att linux använder `/` (slash) för att dela upp sökvägen, inte `\` (backslash).

```
pwd       ①
ls        ②
ls -la    ③
cd ..     ④
cd /      ⑤
cd mapp   ⑥
```

① **P**rint **W**orking **D**irectory. Skriver ut absoluta sökvägen till aktuell mapp.

② **Lis**t. Skriver ut namn på icke-dolda filer och mappar i aktuell mapp. Motsvarar `Dir /b` i Windows.

③ **Li**st **L**ist **All**. Skriver ut en detaljerad "tabell" med namn på samtliga filer och mappar i aktuell mapp. Motsvarar `Dir` i Windows.

④ **C**hange **D**irectory. Går upp en nivå i filträdet - precis som Windows.

⑤ **C**hange **D**irectory. Går till roten av filsystemet - motsvarande `cd c:\` i Windows.

⑥ **C**hange **D**irectory. Går till mappen definerad i sökväg - precis som i Windows.

7.2. Manipulering av filer och mappar

```
touch filnamn    ①
mkdir mappnamn   ②
mv namn nyttnamn ③
rm filnamn       ④
```

① Skapar en tom fil med namnet "filnamn" (eller uppdaterar datum filen senast blev manipulerad, om filen redan finns).

② skapar en tom mapp med namnet "mappnamn"

③ byter namn på en fil/mapp med namnet "namn" till namnet "nyttnamn"

④ tar bort filen med namnet "filnamn" - obs - den hamnar inte i papperskorgen!

7.3. Ip-adress

För att kunna surfa till din webbapplikation behöver du veta dess ip-adress.

Kommandot `ifconfig` listar relevant nätverksinformation:

```
ifconfig ①
enp0s3    Link encap:Ethernet  HWaddr 08:00:27:ac:8c:09
          inet addr:10.0.2.15  Bcast:10.0.2.255  Mask:255.255.255.0 ②
          inet6 addr: fe80::a00:27ff:feac:8c09/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:586 errors:0 dropped:0 overruns:0 frame:0
          TX packets:422 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:64092 (64.0 KB)  TX bytes:59134 (59.1 KB)

enp0s8    Link encap:Ethernet  HWaddr 08:00:27:64:6a:ee
          inet addr:192.168.xxx.xxx  Bcast:192.168.195.255  Mask:255.255.252.0 ③
          inet6 addr: fe80::a00:27ff:fe64:6aee/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1591 errors:0 dropped:8 overruns:0 frame:0
          TX packets:19 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:223955 (223.9 KB)  TX bytes:2584 (2.5 KB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0 €④
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:36 errors:0 dropped:0 overruns:0 frame:0
          TX packets:36 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:20865 (20.8 KB)  TX bytes:20865 (20.8 KB)
```

① Motsvarar `ipconfig` i windows

② Det här är en speciell ip-adress virtualbox använder

③ 192.168.xxx.xxx är adressen på elevnätet - det är den här du vill ha.

④ 127.0.0.1 är den virtuella maskinens interna ip-adress.