Kajsa Araskoug
Louise Flou
DA127A

Slutrapport uppgift 4 – Mörrums vitvaror

Tabeller

För uppgift nummer 4 - Mörrums vitvaror används följande tabeller:

customers(pno, customer_name, email, address, postno, region)

pno är kundens personnummer vilken är unik för tabellen och är därmed även primärnyckeln.

postno står för postnummret för adressen.

region anger staden som kunden bor i.

customer_name, email och address är självförklarande.

products(product_id, product_name, description, brand, price, category, image)

product_id är det unika numret som varje vara har och är på så vis primärnyckeln för tabellen.

product_name anger namnet för varan.

description innehåller en beskrivning av varan.

brand anger vilket märke varan är gjord av.

price är priset som varan säljs för hos Mörrums vitvaror.

category anger vilken typ varan är utav, exempelvis tvättmaskin, kaffebryggare osv.

image innehåller den URL som leder till bilden.

staff(staff_id, title, staff_name, phone, email)

staff_id är ett unikt nummer för varje anställd för att kunna identifiera varje person utifrån kvittona.

title anger vilken typ av jobb den anställde har, exemeplvis säljare eller manager.

staff_name, phone och email är självförklarande.

supplier(supplier_name, phone, website)

supplier_name är namnet för återförsäljaren och är därmed unik då vi utgår ifrån att flera företag inte kan ha samma namn.

phone och website är självförklarande.

inventory(product_id, supplier, product_cost, quantity)

(product_id REFERENCES products, supplier REFERENCES supplier)

product_id är det unika numret som refereras ifrån products.

supplier refererar till supplier_name i tabellen supplier och tillsammans med product_id fungerar dessa två som primärnyckeln i tabellen.

product_cost är priset som varan kostar att köpa in för Mörrums vitvaror.

quantity anger hur många exemplar av varan som finns på lager.

sales_details(sales_id, customer_id, staff_id, sales_date, discount, subtotal)

(customer_id REFERENCES customers, staff_id REFERENCES staff)

sales_id är det unika numret som identifierar en försäljning gjord mellan en kund och en anställd. Den är därmed primärnyckeln.

customer_id är det unika numret som refererar från en kund i tabellen customers.

staff_id är den anställdes unika siffra som refereras från tabellen staff.

sales_date är datumet som försäljningen gjordes.

discount anges endast om kunden fick någon rabatt annars kan denna anges som 0.

subtotal är totalsumman för försäljningen.

sales(sales_id, product_id, quantity)

(sales_id REFERENCES sales_details, product_id references products)

sales_id är det unika numret som hämtas ifrån tabellen sales_details.

product_id är den unika siffra som identifierar en vara i tabellen products. Denna fungerar tillsammans med sales_id som den unika primärnyckeln för tabellen.

quantity anger hur många exemplar av varan som såldes.

Språk & ramverk

Programmeringsspråket som de två gränssnitten skapats med är Python 3.6. Detta eftersom det är det program som vi båda har jobbat med sen tidigare. Vi har även jobbat med det under projektet i systemutveckling. Ramverket som har används för att underlätta uppbyggnaden av webbsidan är bottle då den har inbyggda funktioner som renderar templates, skapar routes för varje sida osv. För att skapa varje sida har vi använt oss av HTML5 som med hjälp av taggar skapar sidans layout och för att sedan styla dessa har CSS3 använts. Valet av dessa beror på att vi båda jobbat med detta tidigare under höstterminen och även under våren i projektet. Ytterligare designval har gjorts utifrån ett minimalistiskt perspektiv. Ikonerna på sidan används genom att länka in Open Source CSS-filer från Materialize i HTML-dokumenten. De presenterar sig såhär:
"Created and designed by Google, Material Design is a design language that combines the classic principles of successful design along with innovation and technology. Google's goal is to develop a system of design that allows for a unified user experience across all their products on any platform."
- Materialize

Uppgiften har delats upp till två delar för att hålla isär användargränssnitten för kunder och de anställda (admin).

Systemet utifrån en programmerarsyn

Customers:

Filen "main" är uppbyggd på så sätt att den först importerar de använda ramverken och funktionerna. Sedan skapas en connection till databasen och sparar den som variabeln "conn" så att man genom hela filen kan kalla på den för att nå databasen. Därefter presenteras varje route som genererar en viss funktionalitet för varje sida. Varje funktion avslutas med att rendera en template (alltså köra en HTML-fil) eller redirect som går vidare i en annan angiven funktion. Koden är kommenterad för att man ska kunna förstå vad det är

varje specifik route gör. Ovanstående text är precis likadan för gränssnittet admin för de Anställda.
I filen main, rad 19 och 26: DISTINCT används eftersom i tabellen för produkter finns det flera produkter som har samma värde för category, exempelvis tvättmaskin, kyl osv. I SELECT frågan måste man då ange DISTINCT för att endast hämta unika värden för att undvika att tvättmaskin visas mer än en gång, en gång per produkt som har detta värdet i kategorin. De blir klickbara med hjälp av HTML och hämtar in alla varor med det matchningen som användaren tryckte på med hjälp av en SQL-fråga i python.
SQL-frågorna är utformade på så vis att dem endast hämtar varorna som finns I lager, alltså inte dem där quantity = 0. Detta för att användaren endast ska kunna se varor som är tillgängliga för försäljning.
Ordningen av priset är simpelt konstruerad då den hämtar in samtliga varor och storleksordnar dem efter pris, antingen stigande eller fallande, beroende på vad användaren tryckte på.
Sökfunktionen söker efter namnet på produkten, man kan även leta upp rätt kategori eller märke med hjälp av de klickbara länkarna uppe i menyn. Sökfunktionen fungerar på så sätt att den hämtar in ett värde för sökningen av användaren via HTML:n och jämför det sedan med värdena i databasen. LIKE '%{}%' används för att inte bara söka på ord som börjar eller slutar med vad användaren skrev.

Admin:
Funktionerna som läser in varor, kunder och återförsäljare fungerar med hjälp av en SQL-fråga som finns för varje route (finns kommentarer i koden som identifierar vilken route som är vilken).
Add_customer som registrerar en kund i databasen fungerar genom att hämta infomationen som användaren matade in i formuläret via HTML. Därefter görs en INSERT till databasen för att utföra registreringen. Även add_supplier fungerar på samma sätt.

Add_product fungerar i princip på samma sätt där skillnaden är att den lägger till i två tabeller samtidigt: products och inventory.
För fler kommentarer, se källkod.

Systemet utifrån en användarsyn

Customers:
Användargränsnittet för kunden är väldigt simpelt och har endast de funktionerna som behövs. I menyn, högst upp för varje sida finner användaren alla kategorier och märken som är tillgängliga. Dessa är tryckbara länkar som genom att trycka på dem läses varorna in med det önskade märket eller kategorin. Sökrutan tillåter användaren att söka på varornas namn och visar sedan användaren de varorna som matchade dens sökning.
Admin:
Användaren navigerar sig på exakt samma sätt som på kundsidan genom navigationsbaren högst upp. Där hittar den enkelt alla alternativ som systemet erbjuder. Genom att trycka på “kundlista” kan användaren se alla kunder som är registrerade i systemet och även registrera en ny kund i formuläret. Genom att trycka på “kundvärde” kan man se totalsumman som varje kund har handlat för.
Under ”lager” kan användaren se alla produkter som finns i lagret. Genom att använda menyn som ligger under navigationen kan användaren även se produkter som inte finns i lager då dem är slut eller inte är inköpta. Användaren kan även lägga till en ny vara i systemet på denna sidan genom att fylla i det första formuläret. Här anger användaren även hur många av varan som finns. Om den inte är inköpt utan endast ska registreras i databasen anger admin ”0” för ”antal”. Användaren kan även köpa in fler varor av en vara som redan finns registrerad i systemet genom att fylla i det andra formuläret med produktID, inköpare och antal. På så sätt uppdateras antalet i databasen för produkten. Notera att återförsäljaren måste vara registrerad i databasen för att produkten ska kunna läggas till i databasen.

Genom att trycka på ”kvitton” kan användaren se alla köp som gjorts. På kvittona kan man se vilken kund som handlat, datum, av vilken anställd, vilka varor och vad totalkostnaden för dessa var. Användaren kan även här registrera ett nytt köp genom att fylla i ett kundID och anställdsID och sedan trycka på ”påbörja köp”. Notera att både kunden den anställde måste vara registrerad i databasen för att köpet ska kunna göras. Då registreras köpet och ett tomt kvitto med kundID och anställdsID dyker upp. Därefter matar användaren in i fälten under vilken vara som köpts och hur många. Användaren kan bara mata in en vara i taget och måste trycka på ”lägg till vara i köp” för varje. När samtliga varor är inmatade trycket användaren på ”Avsluta köp!” för att avsluta och totalsumman registreras för alla varor.
Gränssnittet erbjuder till viss del felhantering. Om användaren försöker registrera ett köp när det inte finns tillräckligt många exemplar registrerade på lagret, i tabellen inventory, kommer den att bli informerad om detta och skickad tillbaka till samma sida på nytt. Om användaren försöker nå en sida som inte existerar så hamnar den på en error sida med en länk tillbaka till startsidan.
Under ”Återförsäljare” ser användaren alla återförsäljare som är registrerade i databasen och kan registrera en ny med hjälp av formuläret.
Fliken ”Personal” fungerar på samma sätt som ”Återförsäljare” fast för personal istället.