VAASAN AMMATTIKORKEAKOULU
VASA YRKESHÖGSKOLA
UNIVERSITY OF APPLIED SCIENCES

Peter Råstu

UTVECKLING AV ETT FAKTURERINGSPROGRAM I MICROSOFT ACCESS

Case Meles Bike

Företagsekonomi och turism
2013

VASA YRKESHÖGSKOLA
Utbildningsprogrammet för informationsbehandling

ABSTRAKT

Författare	Peter Råstu
Lärdomsprovets titel	Utveckling av ett faktureringsprogram i Microsoft Access - Case Meles Bike
År	2013
Språk	svenska
Sidantal	44
Handledare	Kenneth Norrgård

Detta lärdomsprov hade som mål att på ett begripligt och överskådligt sätt beskriva en del av de skeden som genomgåtts för att planera och utveckla ett faktureringsprogram för småföretagare. Faktureringsprogrammet utvecklades enligt uppdragsgivarens önskemål och behov. Det centrala kravet på programmet var att det skulle vara användarvänligt och lätt att uppdatera.

Faktureringsprogrammet gjordes i Microsoft Access 2007. Relationsdatabasen bestod av två databasfiler, där den ena filen innehöll funktionaliteten, programkoden och rapporterna medan den andra innehöll själva databasen.

Planeringen av ett program är av central betydelse för hur väl det i längden motsvarar användarens behov och det är därför viktigt att i detalj analysera olika scenarion som kan bli aktuella för användaren. Databasens struktur och normaliseringen av databasen har underlättat de uppdateringar som hittills krävts till följd av t.ex. ändrade skattesatser.

Programmet har utan större problem kunnat anpassas till företag i olika branscher. Planeringen och utvecklingen av programmet har kommit till praktisk användning och produkten används idag som huvudsaklig faktureringsmjukvara av tre företag.

Ämnesord Access, fakturering, relationsdatabaser, programmering

VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES
Utbildningsprogrammet för informationsbehandling

ABSTRACT

Author	Peter Råstu
Title	Development of an Invoicing Software in Microsoft Access - Case Meles Bike
Year	2013
Language	Swedish
Pages	44
Name of Supervisor	Kenneth Norrgård

The aim of this thesis was to describe the process of planning and development of an invoicing software for small businesses. The software package was developed according to the client's wishes and needs. The key requirements of the program were that it should be easy to use and to update, and suitable for small businesses.

The invoicing software was built using Microsoft Access 2007. The relational database consisted of two database files, with one file containing its functionality, programming code and reports, and the other containing the database itself.

The planning process of any software will ultimately determine how well it will correspond to the user's needs in the long run. Hence, it is important to analyze in detail the different scenarios that may be of relevance to the user. As shown in the present thesis, the structure and normalization of the database made it easier to make necessary configurations and updates (due to e.g. changes in tax rates).

The software has been adapted to a few small businesses in different sectors without major problems. To date, it is still in use as the principal invoicing tool of three enterprises.

Keywords Access, invoicing, relational databases, programming

INNEHÅLL

FÖRTECKNING ÖVER FIGURER OCH TABELLER

1 INLEDNING

Datoriseringen av fakturahantering och bokföring som skett under de senaste decennierna har medfört en övergång från ett uniformt system enligt givna riktlinjer till ett system där olika företag och aktörer gör sina egna fakturabottnar. Tidigare såg alltså alla fakturor i princip likadana ut (på så vis att slutsumman, fakturans förfallodag, eventuellt referensnummer, och mottagarens kontonummer fanns på samma ställe) medan dagens fakturor varierar avsevärt sett till utseende och layout. Detta innebär att det idag kan vara mer arbetskrävande att betala en faktura. Dessutom upplevs vissa av de faktureringshjälpmedel som idag erbjuds till företagare som komplicerade och besvärliga att använda, vilket jag själv upplevt dels i egenskap av företagare, men också fått höra av företagarkollegor.

Eftersom det fanns ett tydligt behov av ett faktureringsprogram som är användarvänligt för den som skapar fakturorna, men även för den som mottar fakturan och skall göra själva betalningen, bestämde jag mig för att utveckla en sådan mjukvara. Utgångspunkten var att mjukvaran skall gå att använda med minimal skolning, d.v.s. att användaren skall kunna använda programmet efter en introduktion på mindre än 30 minuter. Målet var att skapa en komplett produkt avsedd för omedelbar ibruktagning, varför mjukvaran utvecklades i samarbete med en uppdragsgivare som var beredd att både testa och ta i bruk programmet då det var färdigt.

1.1 Uppdragsgivaren

Cykelaffären Meles Bike/ Kb Melker Stenbacka Ky kontaktades som möjlig uppdragsgivare. Företagets ägare Melker Stenbacka uttryckte intresse för ett dylikt program, eftersom han ditintills gjort företagets alla fakturor manuellt i Microsoft Excel. Det visade sig att han letat efter ett simpelt program för ändamålet, men ännu inte hittat något som motsvarade företagets behov. Han behövde även ett kundregister i bakgrunden, då hans faktureringskunder till stor del är återkommande företag. Vidare önskades stöd för att ta ut rapporter för att underlätta bokföringen. Ett annat önskemål var att införskaffningspriset, kostnader för framtida underhåll och licensavgifter inte skulle vara höga. Stenbacka hade

tidigare konstaterat att befintliga program ofta var både dyra och onödigt omfattande för hans företags behov.

1.2 Uppdragsbeskrivning

Efter inledande diskussioner med klienten formulerades programmets grundläggande funktioner som följer:

- Ett lättanvänt faktureringsprogram, som skall fungera i operativsystemet Microsoft Windows XP och i nyare versioner av Windows.
- Kundregister med en kunddatabas där tidigare fakturor skall finnas tillgängliga.
- Layouten på fakturorna skall utseendemässigt motsvara den finska bankstandarden/ Finansbranschens Centralförbund.
- Det skall vara möjligt att pricka av fakturor med s.k. reskontra.
- Programmet skall automatiskt ta kopior på databasen (dessa kopior lagras lokalt på samma hårddisk som programmet är installerat på).
- Det skall finnas möjlighet att ta ut rapporter som underlättar arbetet för bokföraren då faktureringsverifikat skall införas i bokföringen.
- Ändring av och möjlighet att lägga till mervärdesskattesatser skall finnas.
- På de befintliga rapporterna skall det vid behov vara möjligt att göra ändringar i efterhand.
- Det skall finnas möjlighet att senare kunna konstruera helt nya rapporter utöver de befintliga.

Som personlig reflektion kunde nämnas, att de funktioner och den uppbyggnad som jag själv konstaterat att vore rimliga för ett dylikt program i hög grad motsvarade klientens, en erfaren egenföretagare, önskemål.

2 VAL AV VERKTYG

För att utveckla ett program i enlighet med uppdragsbeskrivningen fanns det två olika tillvägagångssätt att välja mellan. Det ena alternativet var att använda Microsofts Visual Basic.NET (förkortas härefter VB.NET) för att göra det grafiska användargränssnittet, och Microsoft SQL Server, Microsoft Access 2007 (förkortas härefter Access 2007) eller Microsoft Access 2010 (förkortas härefter Access 2010) för att göra databasen.

Det andra alternativet var att göra både det grafiska användargränssnittet och databasen i Access 2007 eller Access 2010.

2.1 VB.NET eller Access

Det naturliga valet att göra gränssnittet med borde ha varit VB.NET eftersom det är det programmeringsspråk jag hade använt mig mest av och det vi använt i Vasa yrkeshögskola.

På min arbetsplats hade jag kommit i kontakt med flera applikationer som helt och hållet var gjorda i Access. Jag har noterat att det är en ganska vanlig uppfattning bland programmerare att Access inte kan användas till att utveckla ”riktig” mjukvara.

På arbetsplatsen där jag arbetade hade vi ett löneberäkningsprogram som sex olika avdelningar kunde använda samtidigt, och där antalet anställda uppgick till fler än 200 personer. Detta löneberäkningsprogram var gjort i Access 2003.

Ett annat program som fortfarande används vid samma arbetsplats är ett s.k. POS (Point of Sale)-program, d.v.s. ett kassaprogram som från början var gjort i Access 97 och senare konverterades till Access 2003 (vilket är den versionen som används än idag). I skrivande stund har programmet varit i bruk tio år, 24 timmar om dygnet, utan att det uppstått problem med vare sig antal inmatade poster i databasen, eller andra problem som kunde bero på valet av databas.

Mitt val av program blev slutligen Access. En bidragande orsak är att det grafiska

användargränssnittet i Access binds ihop med databasen mer automatiskt, och att det därför går betydligt snabbare och enklare att få produkten färdig och klar att användas.

2.2 Access 2007 eller 2010

I Access 2007 finns det en del begränsningar som har med själva layouten av det grafiska användargränssnittet att göra, men de är i det här fallet främst kosmetiska. Som exempel kan nämnas att man inte kan ändra färg på ”knapparna” i Access 2007.

Valet av program var slutligen enkelt, eftersom både klienten och jag själv vid tidpunkten använde programpaketet Microsoft Office 2007 Professional, i vilket Access 2007 ingår. Det var därför kostnads- och underhållsmässigt det bästa valet.

En nackdel med Access 2007 jämfört med Access 2010 är att det i Access 2010 är enklare att göra ett distributionspaket, vilket innebär att man kan installera ett program gjort i Access 2010 utan att själva Access 2010 finns installerat på datorn. Detta är möjligt eftersom det i Access 2010 distributionspaket finns ett tillval för att inkludera databasmotorn, så att applikationen kan köras trots att det fullständiga Access 2010 inte är lokalt installerat. (Access 2007 förkortas härefter Access.)

3 PROBLEMOMRÅDEN OCH LÖSNINGAR

Här beskrivs en del av de olika problemen och de lösningar som gjort att programmet till slut har blivit en funktionerande helhet.

Källor som inte direkt kan hänvisas till från texten i detta arbete men som använts otaliga gånger under utvecklingen av faktureringsprogrammet är hjälpfunktionen i Access. Denna funktion nås enklast genom att man trycker på funktionstangenten F1 i programmet, varpå förslag på hjälp för det mest aktuella visas. Hjälpfunktionen i Access är numera integrerad online med (MSDN Microsoft Developer Network 2013) som är ett omfattande bibliotek för utvecklare av alla Microsofts produkter. Förutom den har (Allen Browne's tips for Microsoft Access) också använts flitigt.

3.1 Programmet

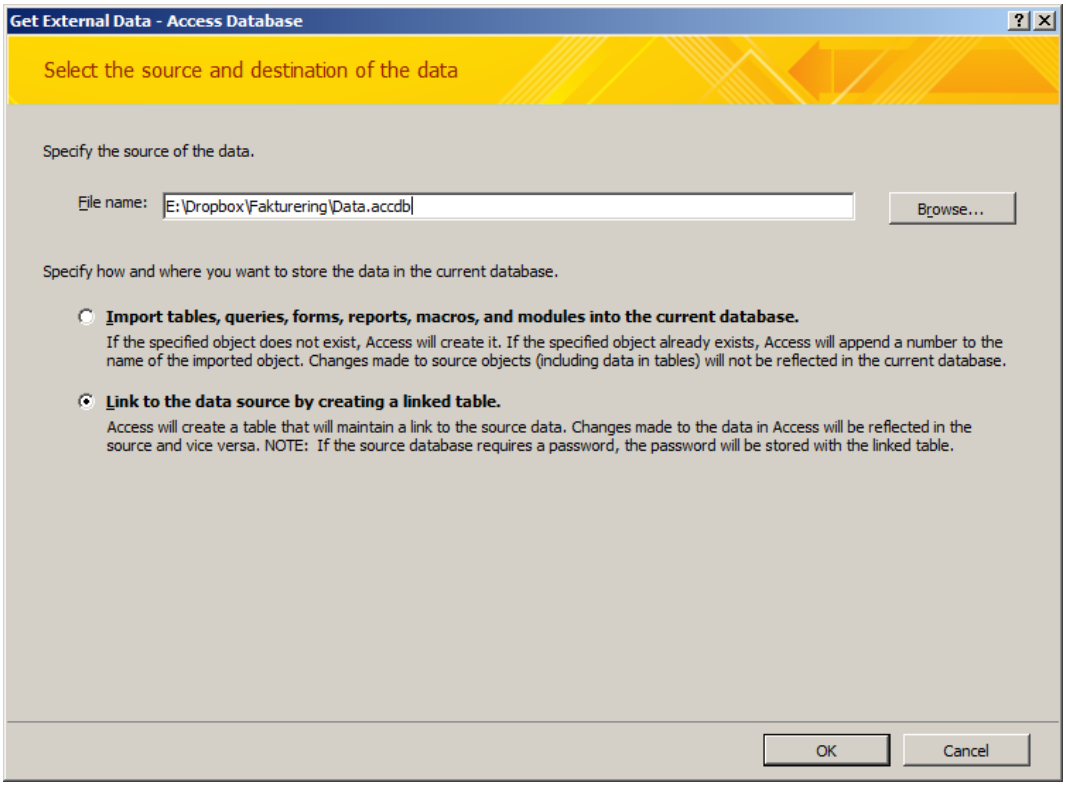
Faktureringsprogrammet består egentligen av två databasfiler. Båda filerna är gjorda i Access och har filsuffixet .accdb som är en förkortning av *acc*ess *d*ata*b*ase. I den ena filen finns funktionaliteten, programkoden och rapporterna. Denna del har jag valt att kalla Fakturering.accdb och denna del utgör själva programmet, i vilket inga data som matas in av användaren, lagras överhuvudtaget. Det är med denna fil som programmet startas.

I den andra delen finns endast databasen, som jag valt att kalla Data.accdb. Orsaken till att programmet är uppdelat på två olika filer är att det skall vara lättare att testa, göra uppdateringar i funktioner och rapporter m.m. på min personliga dator utan att riskera att några data i kundens databas blir överskriven då den uppdaterade delen levereras till användaren.

3.2 Hopkopplingen av databaserna

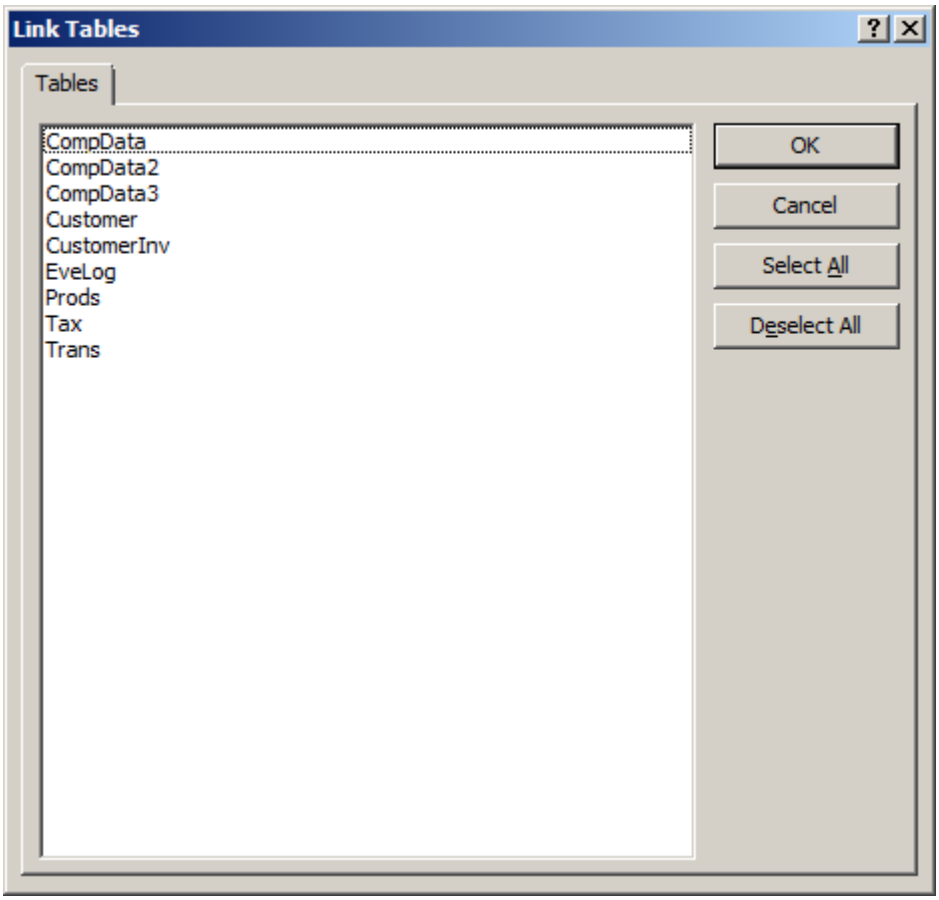
Eftersom databaserna skall hållas helt skilda går det inte att importera data från filen Data.accdb. Databaserna måste därför länkas ihop, i detta fall med hjälp av ”Import Access database” som finns under fliken ”External Data” i Access. I figur 1 på nästa sida kan man se hur man väljer vilken fil som skall länkas ihop med

databasen. Därefter väljs alternativet ”Link to the data source...” och sedan ”OK”.



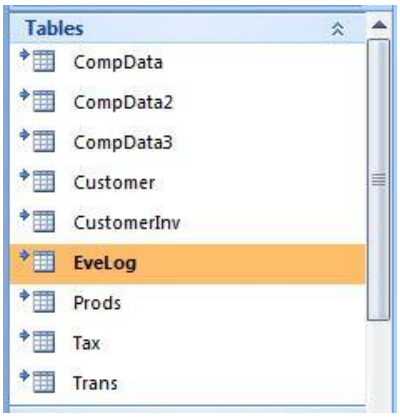
Figur 1. Skärmdump på hopkoppling med en extern databas.

Figur 2 på nästa sida visar en skärmdump där det frågas vilka tabeller som skall väljas. I mitt fall är det alla tabeller som måste finnas i programdelen av faktureringsprogrammet. Man kan välja en och en eller ”Select all” och sedan klicka på ”Ok”.



Figur 2. Befintliga tabeller i den externa databasen.

Efter att tabellerna är länkade till databasen visas de nästan som om de skulle vara inne i den. Enda skillnaden är att det med små blåa pilar illustreras att de inte fysiskt finns i databasfilen, utan att de är länkade från en annan fil. Alla data kan ändå användas på samma sätt som om de fysiskt skulle finnas i filen. Detta illustreras i figur 3 på nästa sida.



Figur 3. Tabellerna som de visas efter att de blivit länkade till databasen.

3.3 Databasen

Databasen är en s.k. relationsdatabas. Detta innebär att tabellerna i databasen länkas ihop med relationer. Dessa relationer består egentligen av värden som är lika i de olika tabellerna och på så vis länkas ihop. De allra flesta databashanterarna, d.v.s. program för databashantering, är idag relationsdatabashanterare. Exempel på sådana är: Oracle, MySQL, Microsoft SQL Server och Microsoft Access.

I detta kapitel beskrivs databasen mera ingående. För att enklare skilja åt tabeller, kolumner och innehållet i kolumnerna används följande tecken: parenteser för (Tabellnamn), klamrar för [Kolumnnamn] och citattecken vid ”Postinnehåll”.

De olika datatyperna som använts, d.v.s. vilken sorts data som kan sparas i de olika kolumnerna är följande:

- *AutoNumber*, ett nummer som databasmotorn automatiskt genererar då man lägger till t.ex. en ny kund. Den använder det senaste numret och adderar det med 1.
- *Number*, ett tal. I denna datatyp måste man bestämma hurudana tal som skall lagras. I figur 4 på nästa sida visas en skärmdump från hjälpen i Access.

- *Currency*, valuta. Här kan bl.a. anges med hur många decimalers noggrannhet värdena skall sparas.
- *Text*, i detta fält kan alla sorts tecken sparas.
- *Date/Time*, här kan man välja mellan olika datumformat och jag har valt att använda något som kallas General Date, vilket innebär att data sparas i formatet ”dd.mm.åååå tt.mm.ss”.
- *Yes/No*, i ett sådant fält kan sparas antingen -1 som står för True eller 0 som står för False.

Setting	Description	Decimal precision	Storage size
Byte	Stores numbers from 0 to 255 (no fractions).	None	1 byte
Decimal	Stores numbers from –10^38–1 through 10^38–1 (.adp) Stores numbers from –10^28–1 through 10^28–1 (.mdb, .accdb)	28	2 bytes
Integer	Stores numbers from –32,768 to 32,767 (no fractions).	None	2 bytes
Long Integer	(Default) Stores numbers from –2,147,483,648 to 2,147,483,647 (no fractions).	None	4 bytes
Single	Stores numbers from –3.402823E38 to –1.401298E–45 for negative values and from 1.401298E–45 to 3.402823E38 for positive values.	7	4 bytes
Double	Stores numbers from –1.79769313486231E308 to –4.94065645841247E–324 for negative values and from 4.94065645841247E–324 to 1.79769313486231E308 for positive values.	15	8 bytes
Replication ID	Globally unique identifier (GUID)	N/A	16 bytes

Figur 4. Skärmdump på de numeriska datatyperna.

En relationsdatabas skall i regel hållas så enkel som möjligt för att undvika så kallad dubbellagring eller redundans. Det finns fem normalformer eller normaliseringsformer (förkortas härefter NF) som de också kallas. De tre förstnämnda är av störst relevans rent praktiskt (Padron-McCarthy 2005). På följande sätt beskrivs fem NF av (Padron-McCarthy 2005):

- 1:a NF. *”En tabell som är i 1NF får bara innehålla atomära värden.”* Detta innebär att det endast får finnas ett värde per post, t.ex. en kolumn [Adress] skall inte innehålla ”65100 Vasa” utan skall delas upp i [Postnr] ”65100” och [Kommun] ”Vasa”.
- 2:a NF. *”En tabell som är i 2NF ska vara i 1NF, dessutom måste varje icke-nyckelattribut vara fullständigt funktionellt beroende av alla kandidatnycklar.”*

- 3:e NF. *”En tabell som är i 3NF ska vara i 2NF, och dessutom får inget icke-nyckelattribut vara fullständigt funktionellt beroende av något annat icke-nyckelattribut.”*
- 4:e NF. *”En tabell som är i 4NF ska vara i BCNF, och dessutom får den inte innehålla några icke-triviala flervärda beroenden. Fjärde normalformen har ganska liten praktisk betydelse när man konstruerar databaser, men den kan vara användbar för att analysera vad som är fel med en dåligt designad databas.”* BCNF (Boyce-Codd's Normal Form) är *”En av de normalformer som används i relationsmodellen.”*
- 5:e NF. *”Ännu en av de normalformer som används i relationsmodellen. Har ganska liten praktisk betydelse.”*

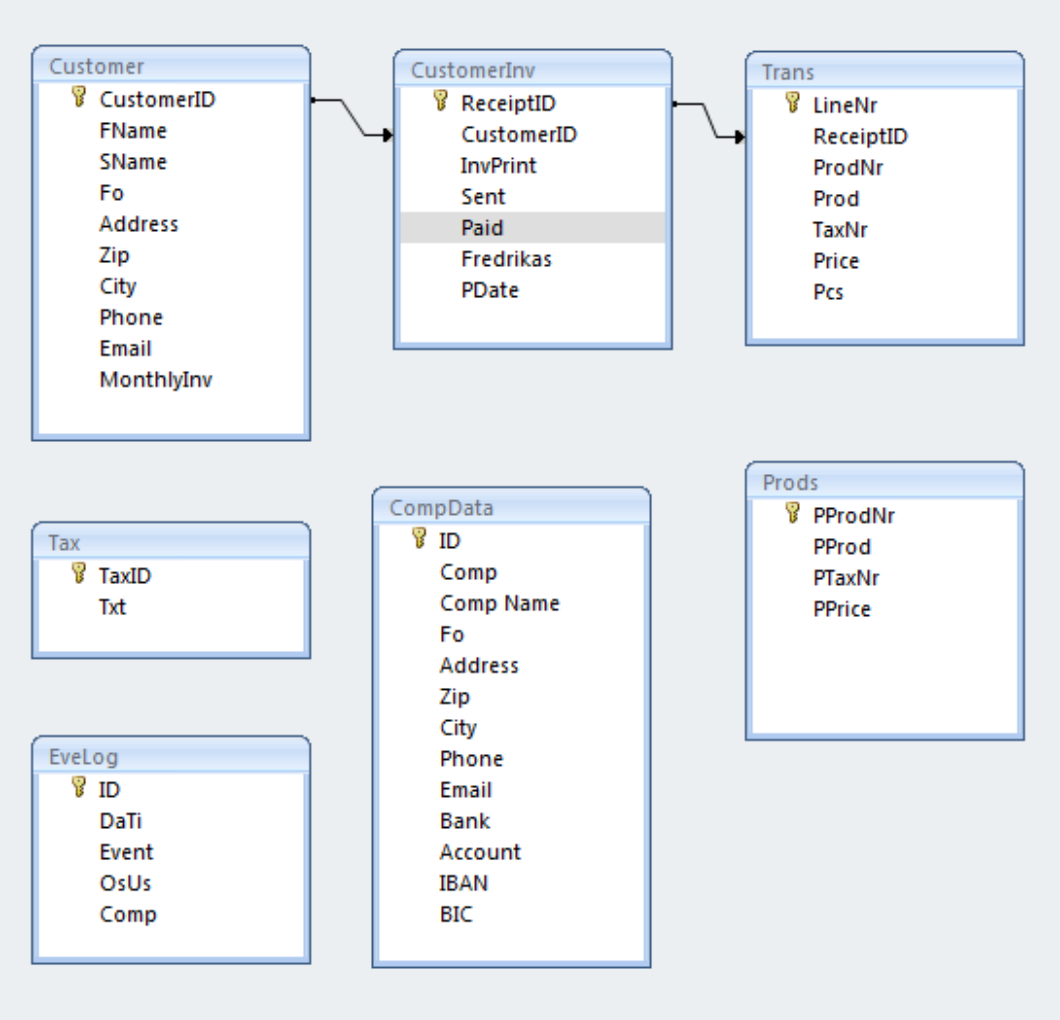
Som man kan se i tabell 1 på sidan 19 strider kolumnerna [Zip] och [City] i tabellen (Customer) mot den tredje normaliseringsformen, eftersom en viss stad eller kommun alltid hör ihop med ett visst postnummer. Enligt (Padron-McCarthy 2005) är det ibland motiverat att inte normalisera allt: *”Ibland är det bra att inte normalisera. I ett adressregister vill man kanske ha med både postnummer och ort i samma tabell, trots att det egentligen strider mot 3NF. Designen blir förmodligen klarare då.”*

Speciellt då det gäller mindre databaser anser jag att de högre normaliseringsformerna inte har lika stor betydelse idag som för 40 år sedan, då normaliseringsmodellen uppfanns av Edgar F. Codd. Detta eftersom datorerna blivit mycket snabbare och har en lagringskapacitet som vida överstiger det som fanns tillgängligt på 1970-talet. Ändå är det alltid bra att försöka normalisera så långt som möjligt även om databasen är liten, för att inte få problem med det framtida underhållet, och med dubbellagring.

3.3.1 Tabellerna

Här finns tabellerna som finns i databasen Data.accdb beskrivna med förklaringar över vad de används till. I figur 5 på sidan 17 finns en schematisk bild av databasen i sin helhet:

- Här lagras uppgifterna om alla kunderna (Customer), som alla kan ha flera fakturor.
- Varje faktura (CustomerInv) kan i sin tur ha flera produkter.
- Tabellen (Trans) är den som innehåller produkterna och bildar raderna på fakturan.
- De olika skattesatserna finns sparade i tabellen (Tax).
- I tabellen (CompData) finns uppgifter om företaget som använder programmet.
- Tabellen (EveLog) används för att logga vad användare gör i programmet.
- Tabellen (Prods) används inte i programmets nuvarande form. Den är färdigt skapad för eventuella framtida behov, om man skulle vilja ta i bruk en produktdatabas.



Figur 5. Schematisk bild av databasen.

3.3.2 Kolumnerna

Här beskrivs hurudan data som lagras i tabellernas kolumner. Det beskrivs också vilken kolumn som är primärnyckel och vilken kolumn som är främmande nyckel för varje tabell. I figuren ovan visas även grafiskt hur de tre viktigaste tabellerna av databasen är hopkopplade. Detta illustreras med en pil, en kund kan ha flera fakturor och en faktura kan ha flera produkter. Nycklarna används för att koppla ihop data mellan olika tabeller. Primärnyckeln måste vara unik, det kan inte finnas två likadana värden i samma kolumn. Den ”kopplas” ihop med en s.k. främmande nyckel som har samma värde i en annan tabell.

Först en förklaring från (Padron-McCarthy 2005), som har en ordlista där primär- och främmande nycklar beskrivs på följande sätt:

> ***”Primärnyckel*** *(engelska:* ***primary key****).*
>
> 1. *I en relationsdatabas: En kolumn, eller en kombination av kolumner, som alltid har ett unikt värde för varje rad i tabellen. (Man får dock inte ta med några onödiga kolumner.) Om det finns flera möjliga primärnycklar säger man att man har flera kandidatnycklar, och man väljer en av dem som primärnyckel.*
> 2. *När man talar om fysiska lagringsstrukturer: Ett fält, eller en kombination av fält, som alltid har ett unikt värde för varje post i filen, och som filen dessutom är sorterad efter.*
> 3. *I verkligheten, eller i en konceptuell datamodell: Något som unikt identiferar* (sic) *en viss sak, till exempel personnumret på en person.”*
>
> ***”Främmande nyckel*** *(engelska:* ***foreign key****). Samma sak som referensattribut.”*
>
> ***”Referensattribut*** *(engelska:* ***reference attribute****). Ett attribut (dvs en kolumn) i en tabell som refererar till (dvs "pekar ut rader i") en annan (eller ibland samma) tabell. Det är inga "pekare" av samma typ som man har i många programmeringsspråk, utan referensen består i att det står ett värde, och sen ska det stå samma värde på en rad i den refererade tabellen. Kallas även* ***främmande nyckel*** *(engelska:* ***foreign key****).”*

Tabell 1. Kolumnerna i tabellen för kunder (Customer).

Tabellen (Customer)	
Kolumn:	**Förklaring:**
CustomerID	AutoNumber, Primärnyckel, genereras automatiskt
FName	Text, namnet på kunden eller företaget
SName	Text, ett till namn eller kontaktperson om det är ett företag
Fo	Text, FO-nummer på företag, det måste vara ett textfält då FO-nummer innehåller tecknet -
Address	Text, gatuadressen
Zip	Text, postnummer eller postbox, det måste vara ett textfält då postboxadresser innehåller bokstäver
City	Text, stad eller kommun
Phone	Text, telefonnummer, kan ha andra tecken än siffror
Email	Text, e-postadress
MonthlyInv	Yes/No, kryssas i om kunden faktureras vid månadens slut

Tabell 2. Kolumnerna i tabellen för fakturor (CustomerInv).

Tabellen (CustomerInv)	
Kolumn:	**Förklaring:**
ReceiptID	AutoNumber, Primärnyckel, genereras automatiskt
CustomerID	Number, främmande nyckel som kopplar ihop fakturan med en viss kund, genereras automatiskt
InvPrint	Date/Time, anger när fakturan är skapad och skickad
Sent	Yes/No, anger om en faktura har blivit skickad
Paid	Yes/No, detta är den enda kolumnen i databasen som inte används av något av de företag som idag använder programmet. Det har tillkommit en ny kolumn med namnet [PDate] som gör kolumnen [Paid] överflödig. Jag har ändå låtit den vara kvar, dels pga. att relativt mycket kod skulle kräva uppdatering om den togs bort. Bokföraren som hjälpt

	mig med rapporterna behöver information om det specifika datumet när en faktura har blivit inbetald på kontot, det räcker inte att veta att den blivit betald.
Fredrikas	Yes/No, denna kolumn blev tillsatt då ett företag med två skilda försäljningsställen började använda programmet. Företaget hade separat bokföring på de olika avdelningarna, vilket löstes med en extra kolumn där det lagras data om vilken avdelning som fakturan tillhör.
PDate	Date/Time, datum för när en faktura blivit betald; och när pengarna de facto syns på mottagarens bankkonto

Tabell 3. Kolumnerna i tabellen för fakturornas rader (Trans).

Tabellen (Trans)	
Kolumn:	**Förklaring:**
LineNr	AutoNumber, Primärnyckel, genereras automatiskt
ReceiptID	Number, främmande nyckel som kopplar ihop produkten med en viss faktura, genereras automatiskt
ProdNr	Number, här lagras inget för tillfället då ingen av användarna tillsvidare använder sig av produktdatabasen
Prod	Text, själva produkttexten är helt öppen och skall skrivas in manuellt enligt uppdragsgivarens önskemål
TaxNr	Number, koden för skattesatsen lagras här men användaren ser bara själva skatteprocenten, inte själva koden som lagras här
Price	Currency, styckepriset på produkten
Pcs	Number, antal produkter

Tabell 4. Kolumnerna i tabellen för skatter (Tax).

Tabellen (Tax)	
Kolumn:	**Förklaring:**
TaxID	Number, Primärnyckel
Txt	Text, skatteprocenten

Tabell 5. Kolumnerna i tabellen för det egna företaget (CompData).

Tabellen (CompData)	
Kolumn:	**Förklaring:**
ID	AutoNumber, Primärnyckel. Det finns bara en post per kolumn eftersom det bara är ett företag som använder det specifika programmet.
12 olika kolumner alla med samma datatyp	Här lagras alla data om det fakturerande företaget. Det finns 12 kolumner som alla har datatypen Text. Trots att flera av posterna innehåller siffror används typen Text då inga beräkningar skall utföras med siffrorna. De är endast till för att skrivas ut på fakturan. Kolumnerna är: [Comp] bolagets firma, [Comp Name] eventuellt marknadsföringsnamn, de flesta bolag heter något annat än det som kunden känner till, [Fo] bolagets FO-nummer, [Address] gatuadressen, [Zip] postnummer, [City] stad eller kommun, [Phone] telefonnummer, [Email] e-postadress, [Bank] bankens namn, [Account] kontonummer i det gamla formatet, [IBAN] kontonummer i IBAN format inklusive FI före sifferserien, [BIC] bankens BIC nummer/ namn

Tabell 6. Kolumnerna i tabellen för loggen (EveLog).

Tabellen (EveLog)	
Kolumn:	**Förklaring:**
ID	AutoNumber, Primärnyckel, genereras automatiskt
DaTi	Date/Time, datum och tid med en sekunds noggrannhet
Event	Text, beskriver händelsen t.ex. att programmet startas eller vilken rapport som öppnas
OsUs	Text, anger vilken användare som gör någonting. Denna information tas rakt ur Windows, d.v.s. användarnamnet rekvireras från operativsystemet.
Comp	Text, datornamnet tas även rakt från operativsystemet

Tabell 7. Kolumnerna i tabellen för produkter (Prods).

Tabellen (Prods)	
Kolumn:	**Förklaring:**
PprodNr	Number, Primärnyckel
PProd	Text, produkttexten
PTaxNr	Number, koden för skattesatsen
PPrice	Currency, styckepriset på produkten

3.4 Referensnummer

Ett referensnummer på fakturor är en nummerserie som förenklar reskontra – alltså avprickning av en viss faktura då den är betald. Enligt (Finansbranschens Centralförbund 2011: 5-6) skall referensnumret vara så kort som möjligt, dock minst fyra tecken (tre tecken + kontrollsiffra). Den längsta tillåtna längden är 20 siffror.

> *”7. Referensnummer*
> *Med referensnummer specificerar fakturautställaren den faktura som skickas till kunden. En betalning som har referensnummer*

förmedlas till betalningsmottagarens konto som är avsett för mottagning av betalningstransaktioner med referens. Fakturautställaren kan fritt utforma referensnumret antingen enligt den nationella finländska standarden eller enligt den internationella RF-standarden.

7a) Referensnummer enligt den inhemska standarden

Inhemskt referensnummer används på inhemska fakturor. För att undvika fel vid sifferinmatning ska referensnumret vara kort, dock minst 4 siffror långt (3 + kontrollsiffra). Den maximala tillåtna längden på referensnumret är 19 + 1 siffror. Referensnumret skrivs ut i det reserverade fältet i grupper om fem siffror med en tom teckenplats mellan grupperna. Förnollor skrivs inte ut. Referensnumret måste också synas på den eventuella fakturadelen. Se bilaga om bildande av referensnummer.

Exempel, referensnummer:
12 34561 (klartextform)
00000000000001234561 (maskinläsbar form)"

Rekommendationen att längden skall hållas så kort som möjligt tycks inte de flesta andra tillverkare av faktureringsprogram ha tagit till sig. Så gott som alla fakturor jag själv betalat både privat och i arbetet är sällan kortare än 15 men ofta upp till 20 tecken långa.

Referensnumret på fakturorna i detta program är uppbyggt enligt följande:

- Första delen är kundnumret, ett löpande nummer som genereras vid inmatning av en ny kund.
- Därefter följer fakturanumret, även det ett löpande nummer som genereras vid inmatning av en ny faktura.
- Slutligen läggs kontrollsiffran till. Här används den nationella finländska standarden för referensnummer, eftersom de som använder programmet så gott som uteslutande fakturerar inom Finland.

Då både kundnumret och fakturanumret är löpande innebär det att referensnumret blir aningen längre med tiden då flera kunder tillkommer och antalet fakturor ökar. Referensnumret kommer ändå i de flesta fall att vara kortare än åtta siffror

och garanterat kortare än de 20 siffror som är det längsta tillåtna.

Hur man räknar ut kontrollsiffran enligt den nationella finländska standarden fanns tidigare beskrivet i Gireringsguiden. I nyaste versionen av guiden är uträkning borttagen. Beskrivningen finns nu i dokumentet ”Inhemska referensnummers uppbyggnad” (Finansbranschens Centralförbund 2009):

> ***”Uträkning av kontrollsiffra för referensnummer***
> *I referensnumret ingår alltid en kontrollsiffra. Den räknas ut enligt följande:*
> - *Siffrorna i grundreferensuppgiften (t.ex. kundnummer eller fakturanummer) multipliceras med vikterna 7, 3, 1, 7, 3,1 …. Från höger till vänster.*
> - *Produkterna adderas och summan dras av från följande jämna tiotal.*
> - *Skillnaden är den kontrollsiffra som skrivs ut som sista siffran i referensnumret.*
> - ***Om skillnaden är 10, är kontrollsiffran 0.***
>
> ***Exempel på uträkning av kontrollsiffra för referensnummer***
> *Grundreferensuppgift 1 2 3 4 5 6*
> *Vikterna från höger till vänster 1 3 7 1 3 7*
> *Produkternas summa 1+6+21+4+15+42 = 89*
> *Summan dras av från följande jämna tiotal (90)*
> *90 - 89 = 1*
> ***Skillnaden är referensnumrets kontrollsiffra 1***
> *I exemplet blir referensnumret inkl. kontrollsiffran 12 34561.*
> *Det skrivs ut på giroblanketten i grupper av fem siffror antingen från vänster till höger eller från höger till vänster.*
> *Mellan siffergrupperna lämnas en tom teckenplats.* ***Förnollor skrivs inte ut på giroblanketten.”***

I faktureringsprogrammet har uträkningen av kontrollsiffran lösts med programkod i VBA (*Visual Basic for Applications;* ett språk för makroprogrammering som ingår i alla program i Microsoft Office-paketet). Koden är skriven så att kontrollsiffran räknas ut korrekt oberoende längden på referensnumret. Figur 6 på nästa sida visar en skärmdump av koden, som räknar ut kontrollsiffran och sparar hela referensnumret som en sträng.

```
Public Function newViite(sViite As String) As String
    Dim iCounter As Integer
    Dim iSum As Integer
    Dim bMulti As Byte
    bMulti = 7
    For iCounter = Len(sViite) To 1 Step -1
        iSum = iSum + Val(Mid(sViite, iCounter, 1)) * bMulti
        If bMulti = 7 Then bMulti = 3 Else If bMulti = 3 Then bMulti = 1 Else If bMulti = 1 Then bMulti = 7
    Next
    newViite = sViite & Right(Str(10 - (iSum Mod 10)), 1)
End Function
```

Figur 6. Skärmdump av VBA koden från Access.

3.5 Rapporter

Alla rapporter är gjorda med verktyget ”Report” som finns inbyggt i Access. Man kan antingen designa rapporterna från början, rita dem på ett tomt ”papper” och sedan koppla ihop de olika fälten med databasen eller ta hjälp av ett guideverktyg (eng. *wizard*).

Vid designen av rapporterna valdes båda eller egentligen en blandning av de båda tillvägagångssätten. Oftast gör jag en ganska rå botten med hjälp av guiden och ändrar sedan designen och lägger till fält för att få ett resultat som jag är nöjd med. I de flesta rapporterna har jag dessutom använt SQL-satser för att kunna utföra de uträkningar som behövs. Senare i detta kapitel, i underkapitel 3.5.4 finns några exempel på SQL-satser. SQL definieras enligt ordlistan i (Padron-McCarthy 2005): *”SQL. Av Structured Query Language. Ett deklarativt frågespråk som används i de flesta databashanterare. Hette från början SEQUEL, och uttalas fortfarande så av en del.”*

Vid utskrift av rapporter visas rapporten först i ett fönster för förhandsgranskning. Man kan sedan skriva ut den antingen på någon av de skrivare man har installerat eller välja en s.k. PDF-skrivare om man skall skicka rapporten per e-post eller spara den på datorn. Jag har installerat PDF-skrivarprogram åt alla som använder faktureringsprogrammet, dessa finns fritt tillgängliga på internet.

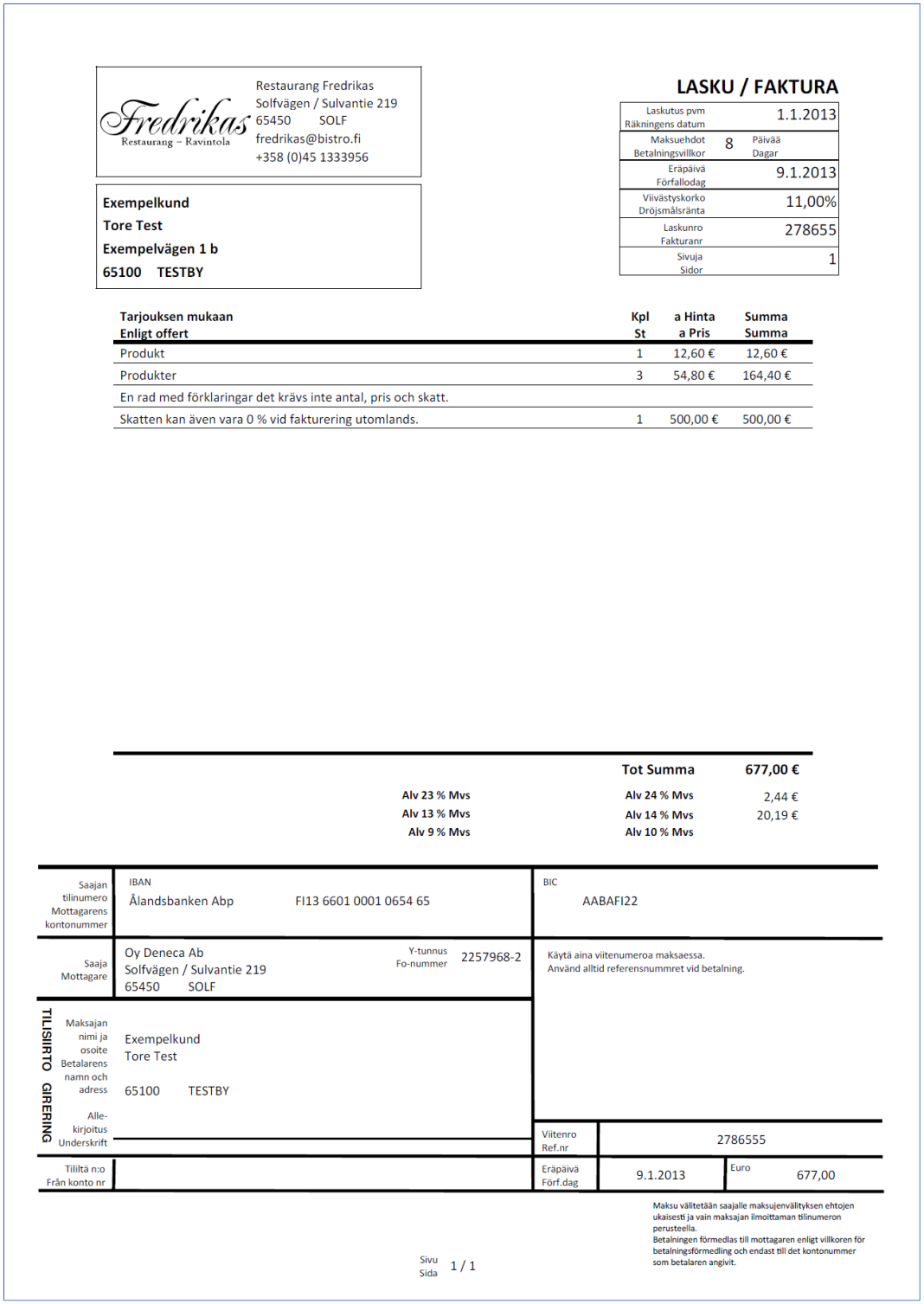
Följande rapporter finns för tillfället med i faktureringsprogrammet:

- faktura
- påminnelse om försenad faktura
- bokföringsrapport
- obetalda fakturor
- försenade fakturor
- oskickade fakturor
- kundlista, rapport på alla kunder inklusive kontaktuppgifter
- användarlogg

Av dessa rapporter kommer fakturan, bokföringsrapporten och användarloggen att få en mer detaljerad beskrivning.

3.5.1 Fakturan

Den viktigaste av alla rapporter är givetvis själva fakturan. Jag har i detta arbete valt att kalla även själva fakturan för en rapport, eftersom det är med ”Report” verktyget i Access som den är konstruerad. Ett av kraven på programmet var att layouten på fakturan skall vara gjord i enlighet med standarden som beskrivs i gireringsguiden utgiven av (Finansbranschens Centralförbund 2011). Både fakturan och en liten del av koden i programmet har ändrats några gånger de senaste åren då bl.a. momssatserna i Finland har justerats, senast 1.1.2013. Figur 7 på nästa sida föreställer en exempelfaktura.



Fredrikas
Restaurang – Ravintola

Restaurang Fredrikas
Solfvägen / Sulvantie 219
65450 SOLF
fredrikas@bistro.fi
+358 (0)45 1333956

Exempelkund
Tore Test
Exempelvägen 1 b
65100 TESTBY

LASKU / FAKTURA

Laskutus pvm Räkningens datum		1.1.2013
Maksuehdot Betalningsvillkor	8	Päivää Dagar
Eräpäivä Förfallodag		9.1.2013
Viivästyskorko Dröjsmålsränta		11,00%
Laskunro Fakturanr		278655
Sivuja Sidor		1

Tarjouksen mukaan **Enligt offert**	**Kpl** **St**	**a Hinta** **a Pris**	**Summa** **Summa**
Produkt	1	12,60 €	12,60 €
Produkter	3	54,80 €	164,40 €
En rad med förklaringar det krävs inte antal, pris och skatt.			
Skatten kan även vara 0 % vid fakturering utomlands.	1	500,00 €	500,00 €

		Tot Summa	**677,00 €**
Alv 23 % Mvs		**Alv 24 % Mvs**	2,44 €
Alv 13 % Mvs		**Alv 14 % Mvs**	20,19 €
Alv 9 % Mvs		**Alv 10 % Mvs**	

Saajan tilinumero Mottagarens kontonummer: IBAN Ålandsbanken Abp FI13 6601 0001 0654 65 — BIC AABAFI22

Saaja Mottagare: Oy Deneca Ab, Solfvägen / Sulvantie 219, 65450 SOLF — Y-tunnus Fo-nummer 2257968-2

Käytä aina viitenumeroa maksaessa.
Använd alltid referensnummret vid betalning.

TILISIIRTO GIRERING

Maksajan nimi ja osoite Betalarens namn och adress: Exempelkund, Tore Test, 65100 TESTBY

Allekirjoitus Underskrift

Viitenro Ref.nr: 2786555

Tililtä n:o Från konto nr

Eräpäivä Förf.dag: 9.1.2013 — Euro 677,00

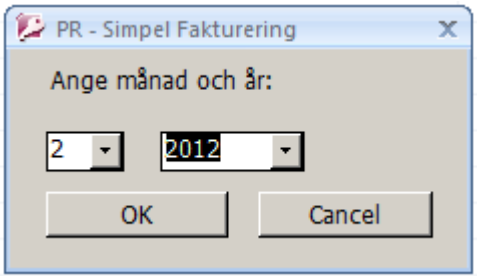
Maksu välitetään saajalle maksujenvälityksen ehtojen ukaisesti ja vain maksajan ilmoittaman tilinumeron perusteella.
Betalningen förmedlas till mottagaren enligt villkoren för betalningsförmedling och endast till det kontonummer som betalaren angivit.

Sivu Sida 1 / 1

Figur 7. Exempel på en faktura.

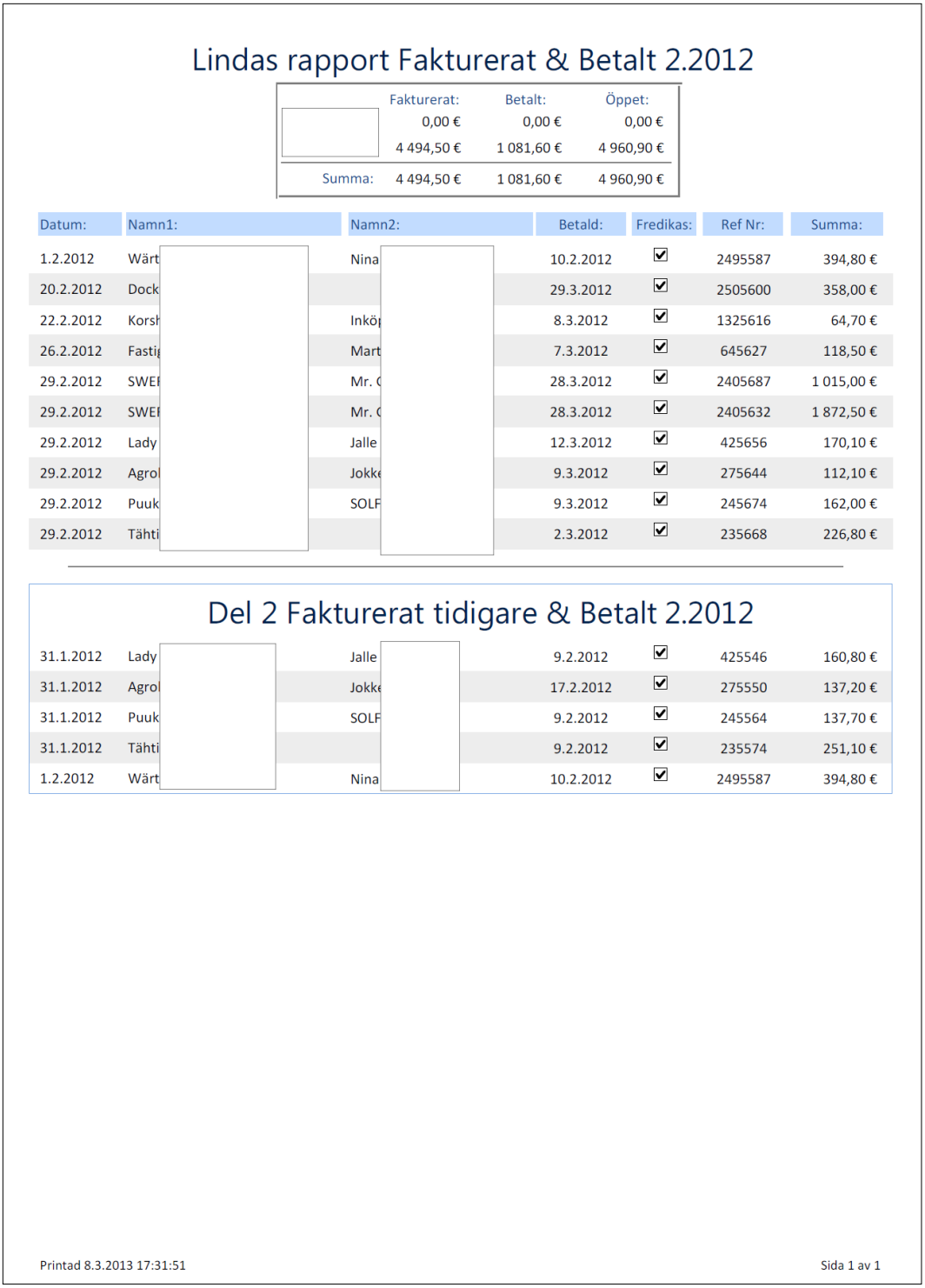
3.5.2 Bokföringsrapporten

Månadsrapporten för bokföring skrivs ut genom att man först trycker på knappen för månadsrapport. Därefter visas ett frågefönster där man kan välja månad samt år för rapporten, standardvärdena är nuvarande månad och år. Efter att man valt dessa och klickat på ”OK” visas en förhandsgranskning av rapporten. En skärmdump på frågefönstret ses härunder i figur 8.



Figur 8. Skärmdump på frågefönster.

Figur 9 på följande sida är ett exempel på en bokföringsrapport för februari månad 2012 som är utskriven 8.3.2013. Utformningen och innehållet är framtagna efter konsultation med en lokal bokförare i Vasa. En av de ändringar bokföraren ville ha efter att ha bekantat sig med en tidig version, var att det borde vara möjligt att utläsa både när fakturan skall betalas, och vilket datum betalningen kommit in på mottagarens konto. Då rapporten blir längre än en sida delar programmet upp de olika delarna på varsin sida och lägger vid behov till flera sidor.



Lindas rapport Fakturerat & Betalt 2.2012

	Fakturerat:	Betalt:	Öppet:
	0,00 €	0,00 €	0,00 €
	4 494,50 €	1 081,60 €	4 960,90 €
Summa:	4 494,50 €	1 081,60 €	4 960,90 €

Datum:	Namn1:	Namn2:	Betald:	Fredikas:	Ref Nr:	Summa:
1.2.2012	Wärt	Nina	10.2.2012	☑	2495587	394,80 €
20.2.2012	Dock		29.3.2012	☑	2505600	358,00 €
22.2.2012	Korsh	Inkö	8.3.2012	☑	1325616	64,70 €
26.2.2012	Fasti	Mart	7.3.2012	☑	645627	118,50 €
29.2.2012	SWEI	Mr. (	28.3.2012	☑	2405687	1 015,00 €
29.2.2012	SWEI	Mr. (	28.3.2012	☑	2405632	1 872,50 €
29.2.2012	Lady	Jalle	12.3.2012	☑	425656	170,10 €
29.2.2012	Agrol	Jokk	9.3.2012	☑	275644	112,10 €
29.2.2012	Puuk	SOLF	9.3.2012	☑	245674	162,00 €
29.2.2012	Tähti		2.3.2012	☑	235668	226,80 €

Del 2 Fakturerat tidigare & Betalt 2.2012

31.1.2012	Lady	Jalle	9.2.2012	☑	425546	160,80 €
31.1.2012	Agrol	Jokk	17.2.2012	☑	275550	137,20 €
31.1.2012	Puuk	SOLF	9.2.2012	☑	245564	137,70 €
31.1.2012	Tähti		9.2.2012	☑	235574	251,10 €
1.2.2012	Wärt	Nina	10.2.2012	☑	2495587	394,80 €

Printad 8.3.2013 17:31:51 Sida 1 av 1

Figur 9. Exempel på bokföringsrapporten.

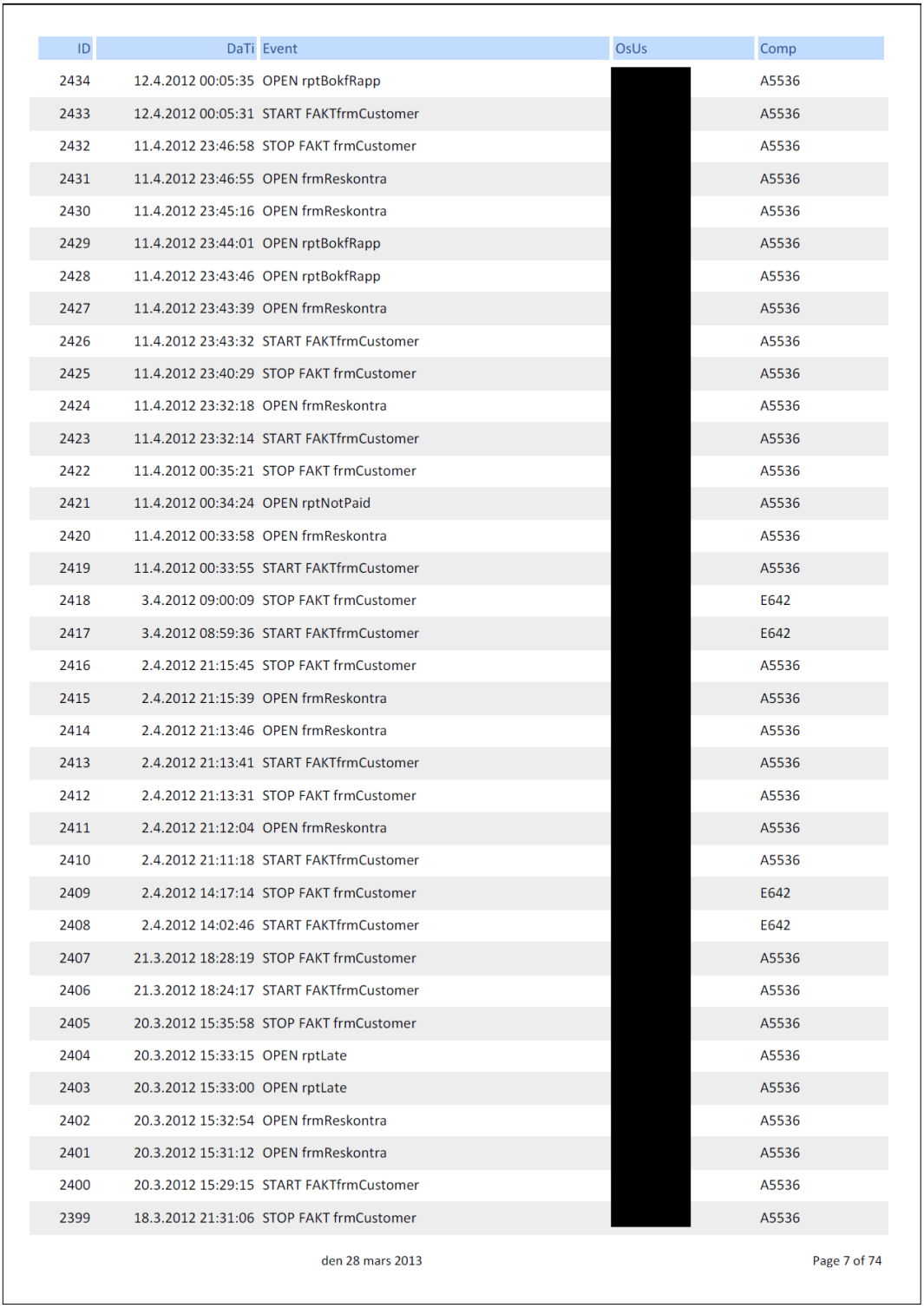
3.5.3 Användarloggen

Användarloggen har kommit till pga. att ett av företagen som använder programmet har två olika avdelningar som finns på två geografiskt olika ställen. Man ville veta vilket ställe som senast varit in i programmet och vilka rapporter användaren varit in i.

Loggen öppnas genom att man dubbelklickar på en viss text i programmet, som inte är utformad som en knapp. Meningen är att bara vissa förmän skall kunna granska loggen. Det går givetvis inte att göra några som helst ändringar eller på annat sätt manipulera denna rapport, utan det går bara att granska och skriva ut den. De olika objekten som registreras i denna rapport är:

- ID i form av ett löpande nummer
- datum och tid med en sekunds noggrannhet
- vad som gjorts i programmet, när det startats, stängts, vilka rapporter som öppnats osv
- användarnamn, namnet med vilken användaren har loggat in i operativsystemet
- datornamnet som tas rakt från operativsystemet.

I figur 10 på nästa sida ser man en sida ur användarloggen, detta exempel är sidan 7 av 74, som är utskriven 28.3.2013.



ID	DaTi	Event	OsUs	Comp
2434	12.4.2012 00:05:35	OPEN rptBokfRapp		A5536
2433	12.4.2012 00:05:31	START FAKTfrmCustomer		A5536
2432	11.4.2012 23:46:58	STOP FAKT frmCustomer		A5536
2431	11.4.2012 23:46:55	OPEN frmReskontra		A5536
2430	11.4.2012 23:45:16	OPEN frmReskontra		A5536
2429	11.4.2012 23:44:01	OPEN rptBokfRapp		A5536
2428	11.4.2012 23:43:46	OPEN rptBokfRapp		A5536
2427	11.4.2012 23:43:39	OPEN frmReskontra		A5536
2426	11.4.2012 23:43:32	START FAKTfrmCustomer		A5536
2425	11.4.2012 23:40:29	STOP FAKT frmCustomer		A5536
2424	11.4.2012 23:32:18	OPEN frmReskontra		A5536
2423	11.4.2012 23:32:14	START FAKTfrmCustomer		A5536
2422	11.4.2012 00:35:21	STOP FAKT frmCustomer		A5536
2421	11.4.2012 00:34:24	OPEN rptNotPaid		A5536
2420	11.4.2012 00:33:58	OPEN frmReskontra		A5536
2419	11.4.2012 00:33:55	START FAKTfrmCustomer		A5536
2418	3.4.2012 09:00:09	STOP FAKT frmCustomer		E642
2417	3.4.2012 08:59:36	START FAKTfrmCustomer		E642
2416	2.4.2012 21:15:45	STOP FAKT frmCustomer		A5536
2415	2.4.2012 21:15:39	OPEN frmReskontra		A5536
2414	2.4.2012 21:13:46	OPEN frmReskontra		A5536
2413	2.4.2012 21:13:41	START FAKTfrmCustomer		A5536
2412	2.4.2012 21:13:31	STOP FAKT frmCustomer		A5536
2411	2.4.2012 21:12:04	OPEN frmReskontra		A5536
2410	2.4.2012 21:11:18	START FAKTfrmCustomer		A5536
2409	2.4.2012 14:17:14	STOP FAKT frmCustomer		E642
2408	2.4.2012 14:02:46	START FAKTfrmCustomer		E642
2407	21.3.2012 18:28:19	STOP FAKT frmCustomer		A5536
2406	21.3.2012 18:24:17	START FAKTfrmCustomer		A5536
2405	20.3.2012 15:35:58	STOP FAKT frmCustomer		A5536
2404	20.3.2012 15:33:15	OPEN rptLate		A5536
2403	20.3.2012 15:33:00	OPEN rptLate		A5536
2402	20.3.2012 15:32:54	OPEN frmReskontra		A5536
2401	20.3.2012 15:31:12	OPEN frmReskontra		A5536
2400	20.3.2012 15:29:15	START FAKTfrmCustomer		A5536
2399	18.3.2012 21:31:06	STOP FAKT frmCustomer		A5536

Figur 10. En sida ur användarloggen.

3.5.4 SQL-satser i rapporterna

SQL är ett frågespråk som används i relationsdatabaser. Det används för att söka upp data, göra beräkningar och att presentera data med. I faktureringsprogrammet har SQL-satser bl.a. använts för att kunna göra rapporter. Nedan ses tre korta och enkla SQL-satser. Som källa för hjälp för SQL-satserna, utöver Access hjälpfunktion har (TECH on the Nets SQL hjälp; SQL-Tutorial.net) använts.

Figur 11 nedan föreställer en SQL-sats som plockar ut all data ur tabellen EveLog. Dessa ordnas i fallande ordning, vilket i det här fallet betyder att den nyaste händelsen visas överst. I figur 10 på föregående sida ser man resultatet av SQL-satsen då dess data har överförts till en rapport.

```
SELECT EveLog.ID, EveLog.DaTi, EveLog.Event, EveLog.OsUs, EveLog.Comp
FROM EveLog
ORDER BY EveLog.ID DESC;
```

Figur 11. SQL-sats för att visa och ordna användarloggen.

Förutom att plocka ut eller välja data med en SQL-sats kan man också göra beräkningar. För att undvika att spara onödig data i databasen, används SQL-satser för att göra beräkningarna. Nedan i figur 12 ses en SQL-sats som räknar ut produkten på ett pris [Price] som multipliceras med ett antal [Pcs] och produkten av uträkningen sparas i variabeln Summa.

```
SELECT Sum([Trans].[Price]*[Pcs]) AS Summa
FROM Trans
WHERE ((Trans.ReceiptID)=[Forms]![frmCustomer]![txtRID]);
```

Figur 12. SQL-sats för en enkel beräkning.

I figur 13 ses en SQL-sats som används till en rapport som visar firma, namn, fakturanummer och total summa m.m. på alla fakturor som är försenade mer än 12 dagar.

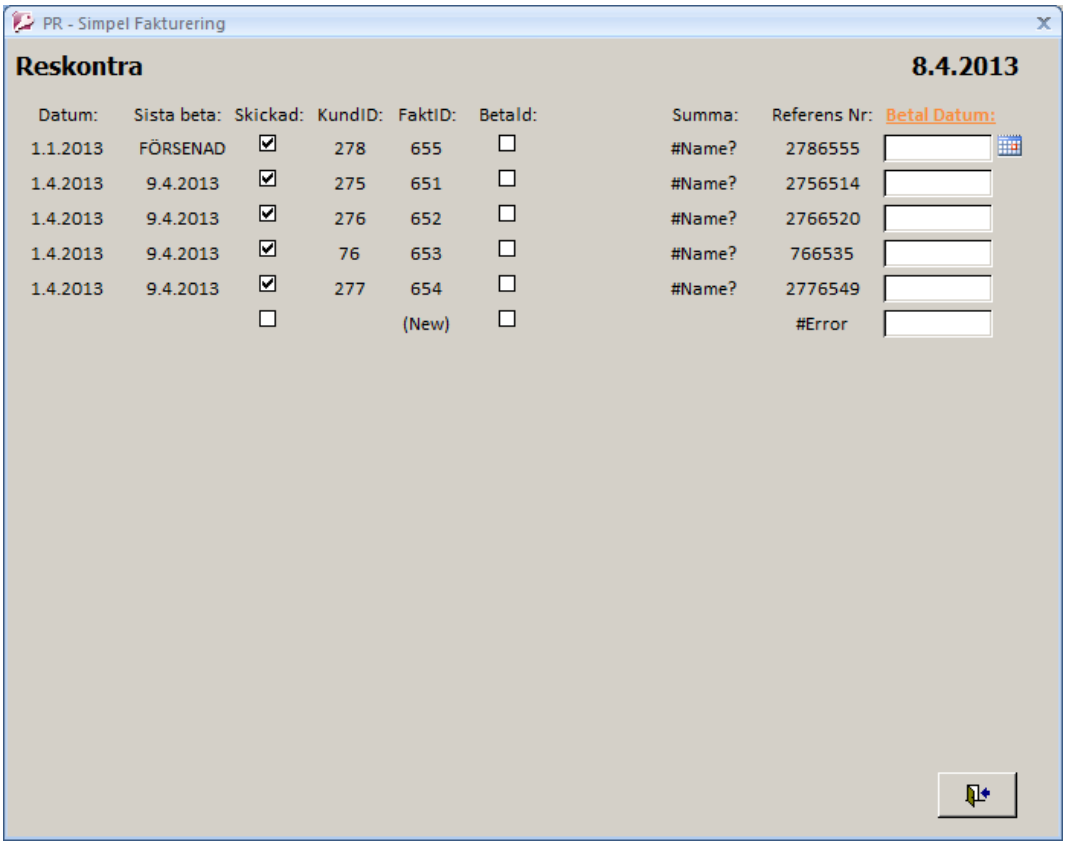
```
SELECT Customer.CustomerID, Customer.FName, Customer.SName, CustomerInv.ReceiptID,
CustomerInv.InvPrint, CustomerInv.Sent, CustomerInv.Paid, Sum(Trans.[Price]*[Pcs]) AS Sum2
FROM (Customer INNER JOIN CustomerInv ON Customer.CustomerID = CustomerInv.CustomerID)
INNER JOIN Trans ON CustomerInv.ReceiptID = Trans.ReceiptID
GROUP BY Customer.CustomerID, Customer.FName, Customer.SName, CustomerInv.ReceiptID,
CustomerInv.InvPrint, CustomerInv.Sent, CustomerInv.Paid, [CustomerInv].[InvPrint]+15
HAVING (((CustomerInv.Paid)=False) AND (([CustomerInv].[InvPrint]+12)<Now()));
```

Figur 13. SQL-sats för beräkningar.

3.6 Reskontra

Formuläret Reskontra som syns i figur 14 på nästa sida visar alla skickade fakturor som ännu inte betalats. Där framkommer när en faktura är skickad och vilket datum som är sista betalningsdag, om fakturan är försenad syns endast texten ”FÖRSENAD”. Man ser också kundnumret, faktureringsnumret, totalsumman och referensnumret.

Fakturorna måste reskontreras manuellt. Detta sker enklast så att man söker upp referensnumret på ett elektroniskt kontoutdrag eller på nätbankens kontotransaktioner. Därefter fyller man antingen i fältet ”Betal Datum” på formuläret med de numeriska tangenterna eller så klickar man på den lilla kalenderikonen höger om datumfältet och väljer rätt datum från den grafiska kalendern som dyker upp. Programmet registrerar fakturan som betald och sparar automatiskt datumet.



Figur 14. Skärmdump av formuläret för reskontra.

3.7 SEPA (Single Euro Payments Area)

SEPA är ett av den Europeiska unionens initiativ för att skapa ett enhetligt betalningssystem för alla 27 medlemsländer, samt dessutom Island, Norge, Schweiz, Lichtenstein och Monaco.

SEPA har trätt i kraft i olika skeden, vilket har krävt ändringar i faktureringsprogrammet. Som källor för dessa justeringar har jag bl.a. använt websidan om SEPA (Finansbranschens Centralförbund 2013) och Europeiska Centralbankens websida (ECBS European Committee for Banking Standards).

Här redovisas några av uppdateringarna som gjorts i faktureringsprogrammet till följd av SEPA. På själva fakturan eller *”Giroblanketten för betalningar i euro inom det gemensamma eurobetalningsområdet (SEPA-gireringar)”* som

(Finansbranschens Centralförbund 2011) kallar den, är de viktigaste ändringarna kontonumren och bankkoden, vilka senast skulle vara ändrade 31.12.2012. Kontonumret har fått namnet IBAN, vilket står för ”International Bank Account Number” och betyder internationellt bankkontonummer på svenska. Fast det heter internationellt bankkontonummer är det i första hand tänkt att fungera inom Europa och fungerar t.ex. inte i USA. IBAN numret börjar alltid med två bokstäver och i Finland är dessa FI, i enlighet med standarden ISO 3166. Förutom detta skall bankens identifieringskod BIC (Bank Identifier Code) anges. BIC koden kallades tidigare SWIFT-nummer och är också en ISO-standard, ISO 9362.

Layouten på fakturan har också fått några uppdateringar. Vissa finstilta texter har flyttats och ändrats och streckkoden har flyttats från mitten i den nedre kanten till den vänstra sidan i den nedre kanten. I faktureringsprogrammet slopades streckkoden helt. Orsaken till detta är att antalet betalautomater och deras användning drastiskt har minskat i.o.m. att de flesta betalar sina fakturor hemifrån via bankernas websidor eller med mobiltelefoner.

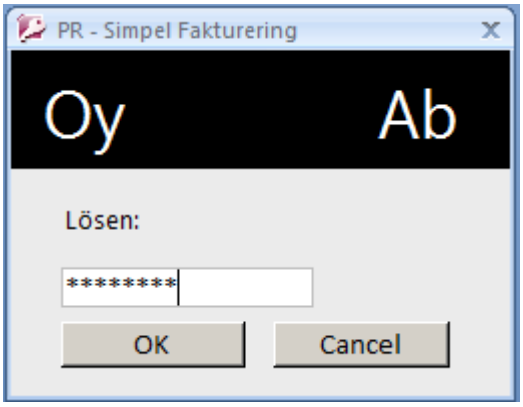
3.8 Säkerhet

Det beslöts i programmets tidiga utvecklingsskede i samråd med klienten att programmet som sådant inte behövde någon särskild inloggning eller kryptering. Detta bl.a. för att alla som arbetar på företaget känner till kunderna, vad de olika produkterna kostar och hur mycket rabatt det är möjligt att ge på de olika produkterna. Det beslöts alltså att det räcker med den inloggning som krävs för att komma in i operativsystemet Windows.

Dock framkom det ganska snart ett behov av en inloggningsfunktion. Denna tillkom vid en av de första uppdateringarna. Orsaken till att inloggningsfunktionen togs i bruk var bl.a. för att förhindra att säsongsanställda och andra vikarier inte skulle ha tillgång till data som för dem inte är relevant, och alltså för att skydda kundernas integritet.

I detta inloggningsfönster, se figur 15 på nästa sida, kan man inte logga in med olika användare. Programmet frågar endast efter ett lösenord och om fel lösenord

anges stängs inloggningsfönstret och programmet måste startas på nytt.



Figur 15. Skärmdump av inloggningsfönstret.

3.8.1 Lokal säkerhetskopiering

Säkerhetskopiering var något som ansågs vara mycket viktigt, eftersom uppdragsgivaren tidigare varit med om en hårddiskkrasch, vilket kan få fatala följder. Detta eftersom all data som finns lagrad på hårddisken kan förstöras och bli obrukbar (Technilab Datarekonstruktion 2013; Aurora IT Systems AB 2013).

Själva programdelen behöver inte säkerhetskopieras eftersom inga data tillkommer, ändras eller uppdateras, utan endast innehåller användargränssnittet och funktionaliteten. Om programmet skulle bli korrupt, raderas eller försvinna, har jag alltid en kopia som det bara är att installera på nytt.

Databasen däremot blir uppdaterad varje gång användaren startar programmet, lägger till en kund, en faktura eller gör ändringar på något. Denna bör det alltså alltid finnas en uppdaterad säkerhetskopia på. Detta har lösts med VBA-kod som tar en säkerhetskopia varje gång någon startar eller stänger fakturerings-programmet. Denna kopia sparas lokalt i ett underbibliotek kallat backup. Där finns kopior på databasen som den såg ut de 10 senaste gångerna programmet startades och de 10 senaste gångerna programmet stängdes.

Filnamnet är uppbyggt så att det börjar med ”Data_”, sedan datumet enligt formen år, månad, dag, därefter tiden och sist ordet ”_LOGIN” eller ”_CLOSE” beroende på om kopian är tagen vid start- eller stängningsögonblicket. Så här kan filnamnet se ut i sin helhet ”Data_20130409_182111_CLOSE.accdb”. Nedan i figur 16 ses VBA koden, med vilken den äldsta säkerhetskopian raderas och en ny kopia av databasfilen tas vid start av programmet.

```
Public Sub Backup_Delete_Start()
Dim s As String
Dim OldFile As String
Dim i As Integer
Dim fso
Dim fsoFile
Dim fsoFolder
Dim sDate As Date
Set fso = CreateObject("Scripting.FileSystemObject")
If Dir(CurrentProject.Path & "\Fakt_Backup", vbDirectory) = "" Then MkDir (CurrentProject.Path & "\Fakt_Backup")
Set fsoFolder = fso.GetFolder(CurrentProject.Path & "\Fakt_Backup")
s = Format(Now(), "yyyymmdd_hhmmss")
Set fso = CreateObject("Scripting.FileSystemObject")
    fso.CopyFile CurrentProject.Path & "\Data.accdb", CurrentProject.Path & "\Fakt_Backup\Data_" & s & "_LOGIN.accdb"
    For Each fsoFile In fsoFolder.Files
        i = i + 1
        sDate = fsoFile.DateCreated
    Next
    If i > 20 Then
        For Each fsoFile In fsoFolder.Files
            If DateDiff("s", sDate, fsoFile.DateCreated) < 0 Then
                sDate = fsoFile.DateCreated
                OldFile = fsoFile.Name
            End If
        Next
        Kill CurrentProject.Path & "\Fakt_Backup\" & OldFile
    End If
    Set fsoFile = Nothing
    Set fsoFolder = Nothing
    Set fso = Nothing
End Sub
```

Figur 16. Skärmdump av VBA koden från Access.

3.8.2 Säkerhetskopiering till molnet

Eftersom faktureringsprogrammen i denna typ av verksamheter inte innehåller några större affärshemligheter, gjordes valet att prova någonting som vid tidpunkten för programmets implementering var relativt nytt. Efter att ha studerat olika lösningar för säkerhetskopiering till molnet beslöts det att testa en tjänst som heter Dropbox. I skrivande stund vet antagligen alla eller åtminstone de flesta vad molnet är och hur det fungerar men 2009 var det någonting nytt, som få hade provat. Idag har de flesta molnlagrings- och backuptjänster även versioner för företag. Källor som använts för molnet och Dropbox är bl.a. (IDG.se, artiklar om

Dropbox; IDG.se, artiklar om molnet; Dropbox hjälpen).

Till molnet, i detta fall till Dropbox servrar, synkroniseras alla filer, mappar och undermappar som finns i Dropboxmappen. Faktureringsprogrammet körs från mappen ”Fakturering”, där även databasfilen finns. I mappen ”Fakturering” finns det även en undermapp med namnet ”Backup”, där de 20 senaste säkerhetskopiorna på databasfilen finns sparade (se föregående kapitel).

Dropbox fungerar så att ett litet program känner av om någon fil har ändrats och synkroniserar då dessa filer och mappar till sina servrar, i dagligt tal kallat molnet. Om man har flera olika datorer kopplade till samma Dropboxkonto synkroniseras även dessa. På detta sätt finns flera olika säkerhetskopior av filerna, även på alla de egna datorerna utöver att filerna finns i molnet. Det enda som krävs är att datorerna är uppkopplade mot internet och har Dropbox programmet installerat samt att alternativet ”Start Dropbox on system startup” är ikryssat.

4 ÖVRIGA ANVÄNDARE

Efter att personalen på Meles Bike har använt och varit nöjda med programmet har även två andra företag visat sitt intresse för programmet. De är båda verksamma inom restaurangbranschen och deras intresse väcktes då jag berättat att det inte finns några hinder för att använda programmet i en annan bransch. Detta har inte varit något problem eftersom alla i Finland använda skattesatser funnits med i programmet från början.

De andra två företagen förutom Meles Bike som använder programmet är:

Oy Deneca Ab Tom Söderholm Solfvägen 219 65450 SOLF Tel +358 45 1333946	Restaurang Olympic Johan Österback Olympiagatan 3 b 65100 VASA Tel +358 50 5398173

De uppgifter som måste matas in eller ändras då ett nytt företag tar i bruk programmet är givetvis alla uppgifter rörande det fakturerande bolaget. Det är 12 poster, vilka kan ses i tabell 5, kapitel 3.3.2 och illustreras i figur 5. Förutom detta så skall givetvis även företagets logotyp på rapporten för fakturan ändras. Alla de övriga uppgifterna på fakturan tas automatiskt från databasen.

Ett undantag var då företaget Oy Deneca Ab skulle ta i bruk programmet. Detta företag hade två försäljningsställen, och de två avdelningarna hade separat bokföring. Programmet var inte ursprungligen avsett för att användas för fler än ett försäljningsställe, och hade heller inte stöd för en sådan funktion. Vidare ville man ha endast ett gemensamt faktureringsprogram, eller en gemensam databas för de olika avdelningarna. Detta innebar att man på något vis måste kunna skilja på de olika avdelningarna. Det här löstes med en ny kolumn i tabellen för fakturor (CustomerInv). Den nya kolumnen fick namnet [Fredrikas] och kan bara ha värdena Yes och No, d.v.s. antingen hör fakturan till avdelningen ”Fredrikas”, eller så gör den det inte.

5 SLUTLEDNING

Det har varit viktigt att hålla sig uppdaterad med regler, förordningar och lagar både under programmeringsprojektet och även efter slutförandet. Bl.a. skattesatserna har ändrat flera gånger under de senaste sex åren. En av dessa ändringar var den nya frisörs- och småreparations- eller ”cykelskattesatsen” som infördes på prov år 2007. Också skattesatserna för restauranger har ändrat ofta de senaste åren. Vid alla ändringstillfällen har jag uppdaterat alla skattesatser. Skattesatserna måste ändras i databasfilen, men detta har inte varit ett problem då databasen var planerad så att ändringarna bara behövdes göras på ett ställe i en tabell. Detta har underlättat uppdateringarna avsevärt och visar att normalisering av databasen lönar sig.

Utvecklingsverktyget Access som användes i detta projekt har fungerat bra och visat sig vara ett bra val för denna typ av program, som består av en databas och ett lite enklare användargränssnitt. Jag är nu också övertygad om att man direkt i Access kan göra relativt komplexa databasapplikationer.

Trenden inom fakturering idag är att företagen i allt större utsträckning sänder s.k. e-fakturor (d.v.s. elektroniska fakturor, som inte skickas ut i pappersform). Detta kräver dock ett avtal mellan företaget och fakturamottagaren, där den senare godkänner att motta e-fakturor från just det specifika företaget. Detta avtal kan göras direkt i nätbanken. E-fakturor är praktiska främst då det gäller löpande och regelbundna fakturor. Om ett företag eller en privatperson köper varor eller använder tjänster oregelbundet eller sporadiskt (t.ex. köper en cykel eller använder cateringtjänster), finns det ingen orsak att göra ett s.k. e-fakturaavtal. Därför är det rimligt att anta att traditionella pappersfakturor inte kommer att försvinna inom den närmaste framtiden.

5.1 Förslag till vidareutveckling

Det skulle kanske vara motiverat att återinföra streckkodsfunktionen som togs bort i en tidigare uppdatering, eftersom det finns streckkodsläsare för hemmabruk och vissa klienter skulle möjligtvis använda en sådan då de betalar räkningar. Det

är inte heller omöjligt att det i framtiden kommer att finnas tvådimensionella koder på fakturor, sådana har börjat synas i allt större utsträckning på de mest olika ställen. De används bl.a. på biljetter for kollektivtrafik och inom flygbranschen. En fördel med dessa är att de kan innehålla en betydligt större datamängd, och att de går att avläsa med de flesta mobiltelefoner efter att man laddat ner en s.k. App (applikation) för ändamålet. Detta kräver dock att kommittén för europeiska banktillsynsmyndigheten (ECBS) enas om vilken av alla de tvådimensionella koderna som finns skall vara standard för fakturor.

Om jag i framtiden gjorde ett nytt liknande program skulle det möjligtvis vara uppbyggt på ett helt plattformsoberoende användargränssnitt. Det skulle antagligen vara byggt för att kunna användas med valfri internetbrowser. Databasen skulle kunna göras i t.ex. den nyaste versionen av Access eller möjligen med MariaDB som är en ersättning eller vidareutveckling av MySQL-databasen (MariaDB Foundation 2013). I ett sådant fall skulle det vara skäl att ha helt andra och större krav på datasäkerhet. I ett helt webbaserat system med inloggningsmöjligheter från en dator var helst i världen, är kraven något helt annat än på ett program som är installerat på en lokal dator, som är inlåst på ett företag.

Om jag utvecklade en ny variant av ett faktureringsprogram för Windows skulle det troligtvis bli i den nyaste versionen av Access. I skrivande stund är det en version som är två generationer nyare än Access 2007, som användes för detta projekt. Den nyaste versionen heter Access 2013, vilken släpptes 29.1.2013 och är version 15.

5.2 Reflektion

Själva idén som jag hade med ett användarvänligt faktureringsprogram för små företag tycker jag att har genomförts bättre än förväntat. Det har varit intressant och givande att ha ett konkret projekt att arbeta med. Det som bidragit till att det varit ett så intressant projekt, är att det är ett riktigt program med konkreta och specifika funktioner och att det från början varit tänkt att användas av ett företag och inte enbart en teoretisk uppgift. En av de största orsakerna till att jag varit så motiverad att slutföra programmet är att jag haft en uppdragsgivare som varit

motiverande under hela planerings- och utvecklingsprocessen, ända tills jag levererat den funktionerande slutprodukten.

Faktureringsprogrammet började jag fundera på redan vid inledandet av studierna, som ett litet projekt och som en del av ett lärdomsprov. Arbetet med programmet har varit mycket givande och intressant men även krävande vad gäller tid, framför allt till följd av alla uppdateringar och ändringar som krävts under processens lopp. Själva programmet har krävt betydligt mera tid än det var tänkt, vilket också varit lärorikt – ett program görs inte på några timmar. Jag tycker själv att det varit en mera lyckad produkt än jag hade vågat hoppas – totalt har fyra företag använt sig av programmet, och det är fortfarande i bruk hos tre företag i skrivande stund.

KÄLLOR

Elektroniska publikationer

Allen Browne's tips for Microsoft Access. Hänvisat 10.4.2009. http://allenbrowne.com/tips.html

Aurora IT Systems AB 2013. Hänvisat 10.4.2013. http://www.aurora.se/harddiskkrasch.htm

Dropbox hjälpen. Hänvisat 15.3.2012. https://www.dropbox.com/help

ECBS European Committee for Banking Standards. Hänvisat 1.2.2013. http://www.ecbs.org/iban.htm

Finansbranschens Centralförbund 2009. Inhemska referensnummers uppbyggnad 1.11.2009. Hänvisat 6.12.2012. http://www.fkl.fi/teemasivut/sepa/tekninen_dokumentaatio/Dokumentit/Inhemska_referensnummers_uppbyggnad.pdf

Finansbranschens Centralförbund 2011. Gireringsguide (I bruk från 1.4.2011), Hänvisat 6.12.2012. http://www.fkl.fi/sv/materialbank/publikationer/Publications/Gireringsguide_4_2011.pdf

Finansbranschens Centralförbund 2013. SEPA Gemensamt eurobetalningsområde, Hänvisat 4.3.2013. http://www.fkl.fi/sv/temasidor/sepa/Sidor/default.aspx på samma sida finns även alla SEPA nyhetsbrev och publikationer http://www.fkl.fi/sv/temasidor/sepa/SEPA_nyhetsbrev/Sidor/default.aspx

IDG.se, artiklar om Dropbox. Hänvisat 17.4.2013. http://www.idg.se/2.1085/1.50095?actionType=search&queryText=dropbox&articleType=0&publicationSelect=0&dateRange=4&sort=0

IDG.se, artiklar om molnet. Hänvisat 17.4.2013. http://www.idg.se/2.1085/1.50095?actionType=search&queryText=molnet&articleType=0&publicationSelect=0&dateRange=4&sort=0

MariaDB Foundation 2013. About MariaDB. Hänvisat 11.3.2013. https://mariadb.org/en/about/

MSDN Microsoft Developer Network 2013, Microsoft. Hänvisat 12.4.2013 http://msdn.microsoft.com/en-us/library/office/bb149076(v=office.12).aspx

Padron-McCarthy T. 2005. En webbkurs om databaser. Örebro, Sverige. Hänvisat 11.12.2012. http://www.databasteknik.se/webbkursen/

SQL-Tutorial.net. Hänvisat 11.3.2010. http://www.sql-tutorial.net/SQL-tutorial.asp

Technilab Datarekonstruktion 2013. Hänvisat 10.4.2013. http://www.technilab.se/diagnostic/analys-haarddisk.htm

TECH on the Nets SQL hjälp. Hänvisat 23.3.2009. http://www.techonthenet.com/sql/index.php