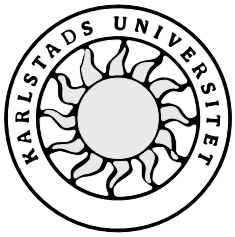



Datavetenskap

---

**Christian Söderblom**

**Thomas Andersson**

# Att flytta en databas från Sybase SQL Anywhere 5 till Oracle8

---

Examensarbete, C-nivå

2000-27

# Att flytta en databas från Sybase SQL Anywhere 5 till Oracle8

**Christian Söderblom**

**Thomas Andersson**

Denna rapport är skriven som en del av det arbete som krävs för att erhålla en kandidatexamen i datavetenskap. Allt material i denna rapport, vilket inte är mitt eget, har blivit tydligt identifierat och inget material är inkluderat som tidigare använts för erhållande av annan examen.

Christian Söderblom

Thomas Andersson

Godkänd, 2000-05-30

Handledare: Hua Shu

Examinator: Stefan Lindskog

# Sammanfattning

Detta dokument presenterar resultatet av ett 10-poängs examensarbete utfört vid Analys-Consult KB i Karlstad under vårterminen 2000. Företaget har utvecklat ett affärssystem vid namn Even(t) AC och för att köra detta system krävs en rätt konfigurerad databashanterare. Systemet är utvecklat mot Sybase SQL Anywhere och körs i dagsläget endast mot denna databashanterare.

För att kunna applicera systemet mot en databashanterare av något annat märke än Sybase SQL Anywhere så behövs en undersökning över huruvida detta är möjligt. Vi har valt att fokusera undersökningen på vilka metoder som kan användas för att flytta över en databas skapad för Even(t) AC från Sybase SQL Anywhere 5 till Oracle8.

# Migration of a database

# –From Sybase SQL Anywhere 5 to Oracle8


## Abstract

This document presents the result of our bachelor project done at Analys-Consult KB in Karlstad (Sweden) during spring term 2000. The company has developed a business system called Even(t) AC. This system requires a correctly configured database management system to work properly. Currently Even(t) AC is based upon a Sybase SQL Anywhere platform and runs only on this database management system.

The ultimate goal of this project is to investigate the possibility of running Even(t) AC on other database management systems than Sybase SQL Anywhere. Our focus is on methods to migrate the database created for Even(t) AC from Sybase SQL Anywhere to Oracle8.

# Innehållsförteckning

# Figurförteckning

# 1 Inledning

Detta dokument presenterar ett 10-poängs examensarbete på C-nivå utfört under sista terminen på dataingenjörsprogrammet vid Karlstads Universitet. Vi har fått ett uppdrag hos Analys-Consult KB i Karlstad, ett företag som sysslar med utveckling av programvara för affärssystem. Företaget har i dagsläget två anställda programmerare/utvecklare samt chefen Leif Olsson. Analys-Consult har sina lokaler på Gjuterigatan i Karlstad och där har vi arbetat med detta examensarbete.

## 1.1 Bakgrund

Analys-Consult KB har utvecklat ett affärssystem för administrering av information lagrad i en databas. Affärssystemet heter **Even** even**(t) A**dministration **C**ontrol, Even(t) AC. Systemet har utvecklats i 12 manårs tid och börjar nu att komma ut på marknaden. De första systemen ligger idag ute hos kunder på olika platser i Sverige. Even(t) AC är utvecklat i kodverktyget Clarion. Clarion är ett program som speciellt används för att skapa applikationer mot databaser.

## 1.2 Problem

Koden som genereras av Clarion liknar Pascal-kod och är enkelt att förstå och läsa. För att hantera databaser använder sig Clarion av embedded SQL.

Systemet körs idag mot Sybase SQL Anywhere genom ODBC, men för att bli riktigt slagkraftigt så måste det givetvis gå att köra det mot fler databashanterare än Sybase. Då hela systemet är utvecklat mot Sybase så ligger problemet i att de inte vet om deras systemet kan köras mot exempelvis Oracle och Microsoft SQL Server. För att kunna köra Even(t) AC så behövs det flera systemtabeller i den databashanterare man vill köra mot. I dagsläget så sker all överföring manuellt från en hanterare till en annan. Alla filer med data granskas för att se i vilken tabell de ska implementeras och vad kolumnerna ska heta, samt av vilken typ de ska vara. Detta är ett tidskrävande jobb där den mänskliga faktorn alltid kan ställa till problem.

## 1.3 Uppgift

Analys-Consult vill att deras system Even(t) AC ska kunna appliceras mot alla de stora databashanteringssystemen med minimala ändringar i programmet.

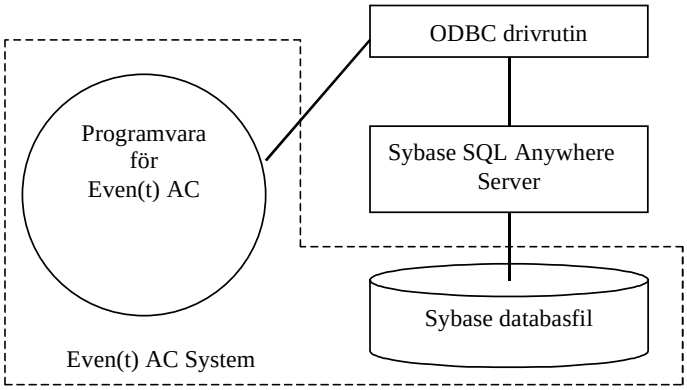
Problemet kan tänkas ligga i att de olika databashanterare använder olika typer av SQL och innehåller olika funktioner. På papper finns det en universell tolk, ODBC för alla SQL dialekter. Med ODBC är det meningen att man ska kunna köra sitt databashanteringssystem mot vilken databas som helst via ODBC.

För cirka fyra år sedan provade företaget att köra sitt då ganska ofullständiga system mot Oracle. Testet dokumenterades inte, men enligt de som var med och provade så fungerade det inte helt tillfredsställande. Mycket har hänt under dessa fyra år på alla fronter inom databaser och användning av dessa. Even(t) AC och framförallt ODBC har utvecklats enormt.

Affärssystemet Even(t) AC består av två huvuddelar, se figur 1.1. Den ena delen är programmet och alla dess ingående komponenter. Den andra delen består av databasen som innehåller systemtabellerna för programmet. Dessa tabeller är färdigutvecklade och det är endast data i dessa tabeller som förändras men strukturen förblir densamma. Systemtabellerna behövs för att köra programmet Even(t) AC och det krävs en databas med dessa tabeller för varje databashanteringssystem, till exempel Sybase eller Oracle.

Vår uppgift innebär att vi ska kontrollera om det är möjligt att anpassa Even(t) AC till andra databashanterare och denna anpassning ska ge likvärdig funktionalitet. Vi ska prova Even(t) AC mot en Oracle databas och kontrollera dess funktion. För att kunna göra detta måste vi finna någon metod för att flytta databasen från Sybase SQL Anywhere till Oracle.

Om det fungerar tillfredsställande utan problem utvecklas examensarbetet till att även kontrollera hur man kan byta databaser ”on the fly” det vill säga att man behåller sitt gamla system och kan importera en databas till en annan databas för att på så sätt utöka eller uppgradera sitt system. På denna uppgift följer tester av prestanda och tillförlitlighet, samt en eventuell rekommendation åt Analys-Consult av vilken systemkonfiguration som ger bäst prestanda.




*Figur 1.1: Even(t) AC systembeskrivning*

## 1.4 Dokumentöversikt

I kapitel ett introducerar vi uppgiften till läsaren. Vi framför i kapitlet viktig bakgrunds-information och presenterar uppgiften med de problem som den medför.

Kapitel två förklarar viktiga begrepp och skapar förståelse för dessa. De begrepp som förklaras är databas, SQL och ODBC.

De förberedelser och installationer av mjuk- och hårdvara som krävdes för vår uppgift beskrivs i kapitlet om systemkonfiguration.

I kapitel fyra redovisar vi vår lösning till uppgiften. Vi diskuterar och beskriver de fyra metoderna vi använt och sedan redovisar vi resultatet av dessa metoder.

Kapitel fem innehåller våra slutsatser och rekommendationer och kapitel sex innehåller vårt slutord.

# 2 Begreppsbeskrivning grundläggande

## 2.1 Vad är en databas?

I dataordbokens definieras ordet databas som en ”mängd av data som består av åtminstone en fil och som är tillräcklig för ett visst ändamål eller för ett visst databehandlingssystem.” [1]

Databaser används för lagring av information inom många olika områden. Databaser lagrar och distribuerar information som vi är beroende av – det kan gälla allt från tågbolagets bokningsinformation till pappas samling av musikskivor. Den vanligaste databas vi kommer i kontakt med är telefonkatalogen. Katalogen är ett register utskrivet på papper från en stor databas där alla hushåll och företag med telefon finns registrerade.

En databas är därmed något som avbildar och beskriver en del av världen runt omkring oss. I vårt fall lagras och hanteras databasen av en dator.

En databas kan delas upp i två delar. Den ena delen består av schemat som man kan säga är strukturen i databasen, själva skelettet. Den andra delen är innehållet i databasen det vill säga data som finns i tabellerna.

För att lagra och hantera databaser används databashanterare som till exempel Oracle, Microsoft SQL Server och Microsoft Access. Detta är ofta stora, komplicerade och kraftfulla program de underlättar all kommunikation med databaser och ger användare ett lätt sätt att manipulera data i databasen.

## 2.2 Vad är SQL?

Structured Query Language (SQL) utvecklades ursprungligen av IBM för deras relationsdatabashanteringssystem DB2 i slutet av sjuttiotalet.

SQL är ett icke procedurbaserat språk, till skillnad från de procedurbaserade, eller 3:e generationens språk typ COBOL och C. Att ett språk är icke procedurorienterat betyder att det beskriver *vad* snarare än *hur*. SQL beskriver till exempel vilka data som ska hämtas, raderas eller infogas, snarare än hur det ska ske. Det finns två standardiseringsorgan som upprätthåller en industristandard – American National Standards Institute (ANSI) och International Standards Organization (ISO). Även om dessa organ skapar standarder som systemutformarna ska följa så skiljer sig alla databasprodukter något från standarden. Dessutom innehåller de

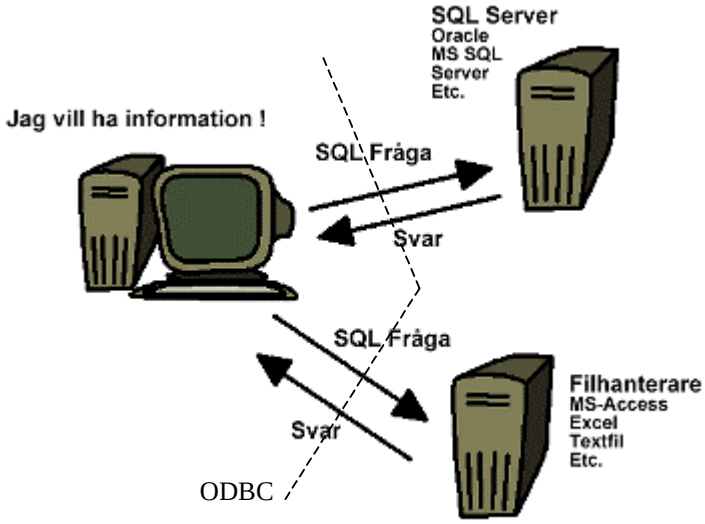
flesta system några egna tillägg till SQL så som triggers och stored procedures vilket gör att språket närmar sig ett procedurorienterat språk.[2]

## 2.3 Vad är ODBC?

Detta avsnitt har hämtats från Filemaker Forum [3]. Dock har vi redigerat texten för att passa vårt syfte.

Open Database Connectivity (ODBC) är ett standardiserat Application Programming Interface (API) som ger en applikation möjlighet att hämta databasinformation från andra datakällor. ODBC ger olika typer av applikationer, som till exempel frågeverktyg, databaser och kalkylblad, ett gemensamt språk och regelverk för att komma åt information från olika typer av datakällor. Exempel på sådana datakällor är Oracle, Microsoft SQL Server och Sybase för att nämna några. ODBC baseras på specifikationen Call-Level Interface (CLI) från X/Open och ISO/IEC och använder sig av SQL som databasspråk.

ODBC är inget filformat, utan ett protokoll som klienter använder för att fråga efter information från en eller flera servrar.

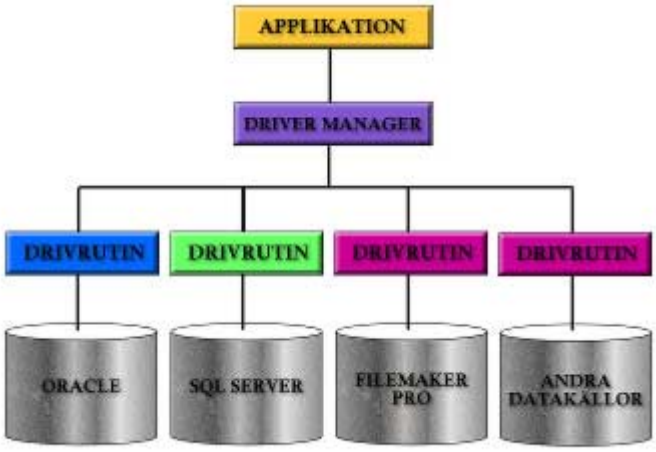



*Figur 2.1: ODBC ett exempel*

Applikationer som är kapabla att kunna skicka frågor till ODBC-Managern, på den lokala maskinen, bestämmer vilken datakälla som finns tillgänglig. Frågorna skickas från ODBC-Managern till drivrutinen för databasen och svaret tas genom drivrutinen till ODBC-Managern för att sedan skickas över till applikationen som ställde frågan.

ODBC realiseras med fyra delkomponenter, se figur 2.2 för deras beroende av varandra. Dessa är följande:

- Datakällan
- ODBC Drivrutin
- ODBC Driver Manager
- Applikation




*Figur 2.2: De olika ODBC-komponenterna*

**Datakälla**

Datakällan kan till exempel vara en Oracledatabas, Sybasedatabas, Excelfil eller annan källa som stödjer ODBC.

**Drivrutin**

Drivrutinerna är Dynamically Linked Libraries (DLL) i Windows eller shared libraries under Mac OS PPC. Denna drivrutin processar de anrop som förekommer mellan Driver Managern och datakällan. Applikationen säger till drivrutinen att koppla upp sig mot datakällan, skicka över SQL-frågan med eventuell konvertering, samt ta emot svaret när detta kommer.

**Driver Manager**

Driver Managern som är en DLL i Windows eller shared library i Mac OS PPC, ger applikationen tillträde till respektive drivrutin. Driver Managern processar applikationens API-anrop och dirigerar dessa mot rätt drivrutin.

**Applikation**

Applikationen är det program som bearbetar informationen. Detta program kan använda sig av de ODBC-funktioner som erbjuds via Driver Managern för att få tag i information från olika datakällor.

**ODBC ett exempel**

1. Applikationen överför en SQL-fråga till ODBC-Managern. Frågan kan vara: Hämta alla kunders namn från försäljningsdatabasen.
2. ODBC-Managern väljer den datakälla som informationen ska hämtas ur. I detta fall kan försäljningsdatabasen finnas i en Oracle server. Innan denna fråga kan sändas iväg mot Oracleservern så måste givetvis Oracles ODBC-driver med mera, vara konfigurerad på rätt sätt.
3. ODBC-Managern skickar över frågan till ODBC-drivrutinen för den databas som ska användas.
4. ODBC-drivrutinen tar emot ODBC-frågan och översätter frågan till den riktiga syntaxen för den databas vi frågar mot. I detta fall så översätts frågan till Oracles eget språk och det kan vara följande SQL-sats: SELECT ALL customer_namnes FROM cust_table.
5. Databasen returnerar svaret genom ODBC-drivrutinen och ODBC-Managern till applikationen och svaret kan se ut som följer: "Sven Svensson"; "Nisse Nilsson"; "Axel Axelsson..."

Denna dynamiska process omfattar oftast flera program och tjänster som arbetar samtidigt. Ett typiskt resultat blir att informationen returneras som rader från databastabellerna.

**Vilka möjligheter finns med ODBC?**

Den övergripande möjligheten med ODBC är att ansluta till olika datakällor, detta innebär:

- Att få specifik information från en annan datakälla.
- Att importera mängder av information från en annan datakälla.
- Att från serverns sida skicka specifik information via ODBC till olika typer av applikationer.
- Att exportera stora mängder av information via ODBC till olika typer av applikationer.

- Att hämta information om datastrukturer (tabeller, fält, etc.)
- Att köra databasspecifika funktioner som inte stöds direkt av ODBC ("pass-through mode")
- Att bibehålla en koppling till en datakälla.

**Vilka applikationer stöder ODBC?**

Många olika typer av applikationer hanterar ODBC. Detta stöd kan finnas dels som ett program (klient) som hämtar information från en datakälla, eller som en server som kan lämna information via ODBC. Vissa typer av applikationer kan agera både som klient och server.

- **SQL-orienterade databaser**, som Oracle, Sybase, Microsoft SQL Server, Microsoft Access, Informix, DB2, etc agerar som typiska servrar. De kan lämna ut tabeller med data till klienter som kan arbeta via ODBC eller via det språk som dessa databaser förstår (Oracle PL/SQL, Sybase Transact-SQL, CICS for IBM mainframe DB2).

- **Applikationer för personligt bruk**, som FileMaker Pro 4.1, Microsoft Word, Microsoft Excel. Dessa är typiska klienter som skickar frågor och kan ta emot svar via ODBC. Microsoft Excel kan också agera som en server eftersom den kan lämna ifrån sig information via ODBC.

- **Webb-till-databas applikationer**, som NetDynamics, Microsoft Active Server Pages, etc. Dessa applikationer ligger oftast i bakgrunden och hämtar information via ODBC-databaser för att sedan generera data till webbsidor som visas i en webbläsare.

- **Frågeverktyg (Query-Tools)**, som till exempel BrioQuery och Crystal Reports är skrivbordsverktyg vars syfte är att hämta stora mängder data. Dessa typer av produkter agerar oftast som frontend-applikationer där slutresultatet blir en rapport av något slag.

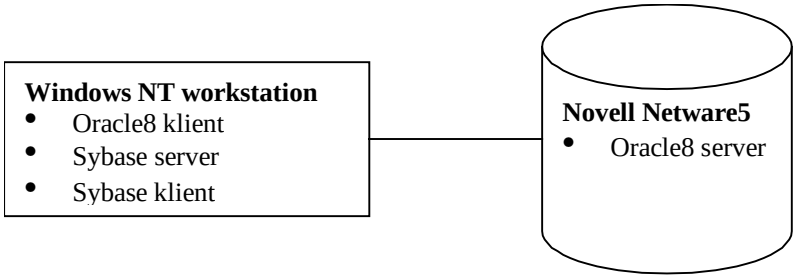
- **Utvecklingsverktyg**, som till exempel Microsoft Visual Basic, PowerSoft PowerBuilder. Det finns möjlighet att skriva SQL-frågor och sätta in svaren i användarnas applikationer med dessa verktyg. Detta medför också att en

applikationsutvecklare kan skräddarsy färdiga applikationer som utför hela processen utan att användaren ser vad som händer. En fördel i detta led är att det oftast går att använda flera datakällor.

**Var kommer ODBC-drivrutiner och ODBC-Managern ifrån?**

MERANT (tidigare INTERSOLV) är den ledande tillverkaren av ODBC-drivrutiner och skapar en stor mängd av dessa för olika typer av datakällor. Dessutom levererar MERANT samma drivrutiner till andra databasleverantörer (till exempel Sybase). En del tillverkare av databasservrar tillverkar egna drivrutiner, som ska följa den standard som finns och kallas då ODBC – installationer. [4]

# 3 Systemkonfiguration - hård och mjukvara




*Figur 3.1: Systemkonfiguration arbetsplats*

## 3.1 Montering av arbetsstation

För att kunna genomföra vårt arbete behövde vi en egen arbetsstation. Vi byggde en arbetsstation med en AMD K6 400 MHZ processor och 64 MB ramminne. Denna hårdvarukonfiguration gav oss tillräckligt med prestanda för uppgiften.

### 3.1.1 Installation av Windows NT samt annan nödvändig mjukvara

På vår arbetsstation installerade vi bland annat följande komponenter:

- Microsoft Windows NT 4.00.1381
- Office 97 paketet
- Internet Explorer 5.0
- Sybase SQL Anywhere server release 5.0.03
- Sybase Central version 2.0 (klient)
- Oracle8 (klient)
- Migration Workbench
- Microsoft Access 97
- Data Junction 6.0

## 3.2 Installation av Oracle8 server release 8.0.3.0.6. på en Novell Netware 5 server.

Det finns mycket att tänka på om man ska installera Oracle8 på en Novell Netware 5 konfiguration. Speciellt viktigt är det att tänka på att alla typer av komprimering är avstängd och att kontrollera detta innan installation. Detta innebär att file compression bör vara avstängd. Ett exempel på file compression är att det finns två stora filer (100 MB) A och B, vilka inte varit öppnade under en tid. Om file compression är på så är dessa komprimerade. Den plats som frigjorts när filerna komprimerats kan således användas av andra nya filer. När sedan databasen öppnas och filerna A och B ska användas måste de dekomprimeras och då finns det ingen plats på enheten för detta.

I installationsanvisningarna står det att Oracle bör installeras på en separat enhet så att file compression kan stängas av på hela enheten när denna skapas. På så sätt behöver man inte stänga av file compression för varje Oracle fil eller katalog för sig.

Vidare står det i installationsanvisningarna för Oracle8 att block suballocation måste vara avstängd. Block suballocation innebär att ett block kan innehålla data från mer än en fil, vilket gör att diskutrymmet kan utnyttjas mer effektivt med en relativt liten tidsförlust.

Att block suballocation är avstängt är oftast inte något problem då Oracle filer oftast är stora och det är endast vid många små filer som man tjänar plats på block suballocation.

Block suballocation är en inställning som väljs vid varje ny installation av en disk i Novell. För att ändra denna egenskap på disken så måste en ominstallation av denna göras. Detta är inget problem så länge som det inte påverkar SYS disken (huvuddisken för Novell) där Novell ligger installerat. Om så är fallet så måste en ominstallation av hela systemet göras.

I vårt fall fanns endast Novells SYS-disk som alternativ. Eftersom det ligger flera databaser och hanterare på denna disk så ville vi inte ta ner den och installera om allt igen. Istället monterade vi i en extra disk där vi skapade en rätt konfigurerad enhet. Sedan gick det att installera Oracle8 på Novellservern.

Systemet Novell kräver 48 MB ramminne för att köra Oracle8. [5] Vid kontroll av minnesanvändning så märkte vi att Sybase, som körs på samma server, använde en stor del av minnet. Den server vi körde mot hade endast 64 MB ramminne och det räckte inte till både Sybase och Oracle. Vi monterade därför i ytterligare 32 MB ramminne, totalt 96 MB, och det är tillräckligt för att köra de båda.

Efter att ha haft Oracle i drift ett par dagar så ansåg vi att det inte var riktigt så stabilt som det borde vara. Vi funderade på vad som kunde tänkas orsaka problemen och kom fram till att det skulle kunna vara ett kommunikationsproblem. Dessa misstankar grundade vi på att vi vid serverinstallationen valde kommunikationsprotokollet TCP/IP, då vi felaktigt fått anvisningar om att detta var det protokoll som användes. När vi sedan fick reda på att det var SPX protokoll som användes för att kommunicera över det lokala nätverket, antog vi att en ominstallation av Oracle eventuellt skulle ta hand om alla småfel. Efter ominstallation av servern, där SPX valdes, så fungerade allt som det skulle.

Klienten till Oracle på en Windows NT workstation var enkel att installera och inga problem uppstod under installationen.

## 3.3 Installation av Sybase SQL Anywhere server release 5.0.03 på Windows NT

Vi beslutade att installera Sybase servern lokalt på vår Windows NT workstation eftersom den Novell server vi installerat Oracle på och som även företaget kör sin Sybase server mot var tillräckligt belastad. Både servern och klienten till Sybase gick snabbt att installera och jämfört med Oracle var det en mycket enklare process. Det var bara att starta installationsprogrammet och följa anvisningarna.

# 4 Undersökning av olika lösningsmetoder

Vi har undersökt fyra stycken olika metoder för att lösa vår uppgift, men vi lyckades inte lösa uppgiften fullt tillfredsställande med någon av metoderna. Den lösning som vi kom längst med var den fjärde och sista. Med denna lösning så lyckades vi med överföringen av den kompletta databasen, men vi lyckades inte få databasen att fungera mot Even(t) AC eftersom vi inte haft tid, eller kunnat få hjälp med konfigurationen av programmet.

## 4.1 Metod 1: Direktimport

Tänk om det vore så enkelt att man kunde ta hela schemat från en databas och exportera den till ett gemensamt format som alla databashanterare förstod och accepterade. Eller om man kunde exportera en tabell med alla kolumner, kopplingar och data och sedan importera den till den databashanterare man vill använda.

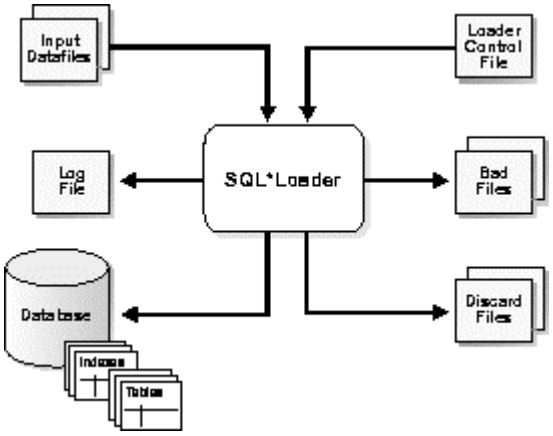
I dagsläget med de verktyg vi provat fungerar det inte riktigt så enkelt. Vi har försökt att använda en enkel direktimportlösning för att kunna exportera en komplett databas från Sybase till Oracle, men det visade sig att ingen av databashanterarna stödjer denna funktion.

Om man vill exportera en databas från Sybase, det vill säga, unload database så exporteras det ut två typer av filer. Den ena filen är en fil som heter reload.sql och den innehåller syntaxen för att återskapa hela databasen. Trots sitt namn med filändelsen sql så innehåller inte filen endast SQL uttryck utan även en del Sybase specifik syntax (se bilaga B.1). De SQL uttryck som förekommer i filen är även de genererade med en Sybase dialekt. Reload.sql är mycket enkel att läsa och förstå och man ser på ett enkelt sätt i vilken ordning och även hur databasen återskapas om man exekverar filen. Den andra typen av filer som genereras när man kör unload är av typen ”nummer.dat” där nummer är ett löpnummer. I vårt fall med start på 161 och slut på 374. Dessa filer innehåller data till tabellerna (se bilaga B.2).

När man sedan vill återskapa databasen i Sybase så kan man exekvera reload.sql som en SQL fråga i Sybaseapplikationen isql (interactive sql). Allt sköts helt automatiskt och en ny databas med samma egenskaper som ursprunget skapas. Problemet är att om man försöker köra denna fil i Oracle så är det mycket som inte stämmer överens.

## 4.2 Metod 2: SQL*Loader

SQL*Loader är ett Oracleprogram som läser data från externa filer till tabeller i en Oracledatabas. Filernas format kan variera mycket och programmet kan läsa data från flera filer under samma session. Programmet kan även läsa data till flera tabeller. Inställningarna är många (görs i en kontrollfil) och flexibiliteten är stor. Data i filerna kan även manipuleras innan inläsning i Oracle med SQL-uttryck.




*Figur 4.1: SQL*Loader översikt*

### 4.2.1 Idé

Som vi sett tidigare finns det en funktion i Sybase som exporterar databasen till en sqlfil och flera datfiler. Eftersom SQL*Loader kan läsa in data från flera filer samtidigt till Oracle så kunde detta program användas. För att överföringen ska vara möjligt måste en kontrollfil skrivas till SQL*Loader. Stöd för att skriva kontrollfilen kunde fås från reload.sql.

### 4.2.2 Resultat

Vi övergav metoden rätt snabbt även om SQL*Loader är ett utmärkt verktyg. Hade det bara rört sig om en tabell så hade kontrollfilen kunnat konstrueras relativt snabbt. Men då vi hade många tabeller insåg vi att det skulle bli ett tidsödande arbete att konstruera kontrollfilen för alla dessa tabeller, speciellt som alla kolumner i tabellerna måste specificeras i kontrollfilen.. Ytterligare ett problem med metoden, förutom det omfattande arbetet med att skriva kontrollfilen var att SQL*Loader bara överför tabelldata och inget databasschema till Oracle.
Det skulle inte vara något problem att strukturera upp databasschemat manuellt efter dataöverföringen eftersom Sybasedatabasen finns dokumenterad, men det hade blivit ett mycket tidsödande arbete.

På grund av att nästan allt arbete med att konstruera databasschemat måste ske manuellt vid användning av SQL*Loader så övergav vi denna metod.

## 4.3 Metod 3: Migration Workbench

Migration Workbench är ett program som Oracle erbjuder för att överföra hela databaser mellan olika databashanterare (servers). Verktyget är gratis och går att ladda ner från Oracles teknologisida [6].

Efter att ha laddat ner och installerat programmet på en Windows NT klient så undersökte vi dess möjligheter att lösa vårt problem. Vi läste i dokumentationer på nätet och även de som följde med programmet och detta program verkade vara precis vad vi letade efter. Men hur vi än försökte så hittade inte Migration Workbench den källdatabas vi använder, det vill säga, Sybase SQL Anywhere server release 5.0.03. Eftersom detta var en av de första lösningsförsök vi gjorde och programmet verkade lovande så trodde vi naturligtvis att det var vi som gjorde något fel. Problemet med att hitta vår källa kvarstod och vi började tveka på huruvida det verkligen var vi som gjorde fel eller om det var någonting annat. Vi skickade e-post till alla supportavdelningar inom Oracle som vi kunde hitta, både svenska och utländska, tyvärr så var det inte många som hade en aning om vad vi talade om. Inte ens supporten för Migration Workbench visste helt säkert vad programmet klarar av. Trots att vi uppgett den exakta versionen av de databashanterare vi använder så verkade Oracles support mycket förvirrade och hade en dålig uppfattning om vad som gällde hos deras konkurrenter Sybase och vilka olika databashanterare Migration Workbench stödjer.

Efter hård e-postdebatt med supporten kom till sist efter flera veckor detta e-postmeddelande:

I normally assume the basic Sybase (adaptive server) unless otherwise stated. Sybase SQL Anywhere 6.0 is the latest however 5.5 is a recent release. We are presently working on creating a SQL Anywhere 5.5 plugin to the Migration Workbench to support this migration however it is presently in the early alpha stages and is not ready for external use.

What sort of time frame are you looking at? I can keep your name on file and as soon as we are nearer to beta release time and have the beta criteria defined I can let you know. Perhaps you could be involved in the beta program and get early access to the product.

Let me know how your time frames are looking.

Regards,

Marie

*Figur 4.2: Brev ett från Oracles support*

En månad senare efter ytterligare påstötningar från oss kom detta e-post.

We have run into some technical problems that we are working through and as a result

I cannot commit to a beta release date at this point.

My apologies,

Marie

*Figur 4.3: Brev två från Oracles support*

Till sist fick vi alltså reda på att Oracles program Migration Workbench bara fungerar med Sybase Adaptive Server Enterprise 11 och senare versioner. Denna server är en mer internet-anpassad server än Sybase SQL Anywhere server som vi använder.

Med ovanstående e-postmeddelande avslutades våra försök med Migration Workbench då vi kände att vi varken ville vara med och utveckla något nytt verktyg eller att vi hade tid att vänta på att det ska bli klart.

## 4.4 Metod 4: Clarion och Data Junction

Med hjälp av en medarbetare på AC System upptäckte vi en funktion i Clarion som exporterar databasschemat från Sybase till Oracle. Denna funktion skapar sex filer som sedan kan användas i Worksheet för Oracle eller i något annat program som kan tolka SQL-uttryck. Dessa filer innehåller allt för att återskapa databasschemat. Clarion är ett väletablerat utvecklingsverktyg som används på företaget för att utveckla Even(t) AC. De två filer vi har använt oss av är till för att återskapa tabeller med primärnycklar och index samt för att sätta främmande nycklar. De övriga fyra filerna från Clarion har vi inte använt då de återskapar funktonalitet som inte fanns i den aktuella Sybasedatabasen. Vår idé var att först återskapa tabellerna i Oracle från den clarionskapade filen book_tbl.sql, vilken består av SQL-uttryck. Filen book_tbl.sql skapar vid körning alla tabeller från Sybasedatabasen, med de önskade primärnycklarna. När detta är gjort får man använda något annat verktyg för att överföra tabelldata. I vårt fall tänkte vi använda Data Junction. Eventuellt kunde Access ha använts men då finns inte någon möjlighet att spara operationerna utan allt får göras om varje gång man vill utföra dataöverföringen. Efter att alla tabeller och all data fanns i Oracle, körde vi filen book_fky.sql för att koppla samman tabellerna med hjälp av främmande nycklar.

### 4.4.1 De skapade clarionfilerna

I Clarion finns ett menyalternativ varifrån Sybases databasschema kan exporteras. Här finns möjligheter att ställa in de föredragna inställningar för de exporterade filerna, till exempel namn, prefix och suffix. Vi valde namnet *book* därför heter alla våra filer något som börjar med *book* och avslutas med ett suffix och filändelsen sql (se bilaga C). Våra sex filer heter: book_err.sql, book_log.sql, book_fky.sql, book_tbl.sql, book_ndx.sql och book_trg.sql. Då Sybasedatabasen inte innehåller några triggers så är följaktligen book_trg.sql tom. Book_ndx.sql är också tom eftersom Sybasedatabasen inte innehåller några index för stigande och fallande ordning. Eftersom filerna är tomma har vi inte använt oss av dem. De två filerna book_err.sql och book_log.sql är till för att skriva fel i och används inte vid förflyttningen men kommer väl tillhands efter förflyttningen för att kontrollera fel som har uppstått. Speciellt logfilen användes flitigt då det visade sig att book_tbl.sql inte gick att köra

problemfritt i Oracle Worksheet. Detta genererade felmeddelanden i logfilen som gjorde att vi var tvungna att ändra en del i book-tbl.sql manuellt efteråt.

**Strängar**

I book_tbl.sql (se bilaga C.1) finns det uttryck som anger att kolumner med strängar ska skapas i Oracle eftersom tabellerna i Sybase delvis innehåller denna typ. Oracle stödjer bara strängar med upp till 2000 tecken och därför måste alla förekomster av char(2048) ersättas med char(2000) istället. Ett alternativ hade varit att byta datatypen till varchar istället vilken stödjer ända upp till 4000 tecken men detta hade inte varit bra då typen inte blir exakt densamma [7].

**Tablespace**

I book_tbl.sql (se bilaga C.1) finns uttryck som anger att tabellerna ska skapas i TABLESPACE OPTOO_*tabellprefix*, där tabellprefix är de tre första bokstäverna i tabellens namn. Tablespace är det logiska utrymmet där en tabell lagras. I ett tablespace kan en eller flera tabeller inrymmas. Funktionen i Clarion som skapar book_tbl.sql har valt ett olyckligt namn på det tablespace där en tabell ska lagras. Anledningen till att det är olyckligt är att det på grund av tabellprefixet blir ett tablespace för varje tabell och tabellerna är väldigt många. Att skapa så många tabelspace manuellt i Oracle, alla med olika namn, tar lång tid och är arbetsamt. Vi valde att istället för att ägna stor tid åt att använda tabellutrymmet USER_DATA som redan fanns i Oracle. För att använda tabellutrymmet USER_DATA ersatte vi alla förekomster av OPTOO_tabellprefix i filen book_tbl.sql så att alla tabeller sparas i USER_DATA istället.

**Index**

I book_tbl.sql (se bilaga C.1) skapas också uttrycket INDEX OPTOO_*tabellprefix*.inx och på samma vis som i föregående avsnitt valde vi att använda tabellutrymmet USER_DATA. Det fanns vidare några strängar i Sybase på 2048 tecken som var indexerade och Oracle kan endast indexera upp till 748 tecken. Dessa felmeddelanden gick inte att undvika eller hitta någon lösning på. Det går bra att använda Event AC ändå mot Oracledatabasen eftersom index bara ökar hastigheten på sökningen. Det finns antagligen ett sätt att lösa problemet med index, men uppsatsens tidsbegränsning gjorde att vi inte hann undersöka detta närmare.

**Skräp**

Filen skapar också en hel del skräptecken till exempel |  efter vissa nyckelnamn. Vi tog bort alla sådana skräptecken manuellt då dessa medför att namnen blir felaktiga. Att ta bort saker är naturligtvis inte bra men funktionen i Clarion som skapar filerna fungera inte helt tillfredsställande och innehåller uppenbarligen en del felaktigheter.

### 4.4.2 Återskapa databasschemat i Oracle

Börja med att starta Oracle Enterprise Manager för att kunna skapa en ny användare. Logga in som databasadministratör till den aktuella databasen.

När programmet startat navigera fram till users under databasen. Högerklicka och välj create. Fyll i namn, password, tablespaces som användaren ska använda. På fliken roles/priviliges väljs de rättigheter användaren ska ha. I vårt fall valde vi connect, resource och dba.

Starta sedan Oracle SQL Worksheet och logga in med användaren som nyss skapats. Öppna den exporterade filen book_tbl.sql och kör sedan denna. Tabeller, nycklar och index skapas i Oracle. Titta i loggfilen som skapas för att upptäcka eventuella fel.

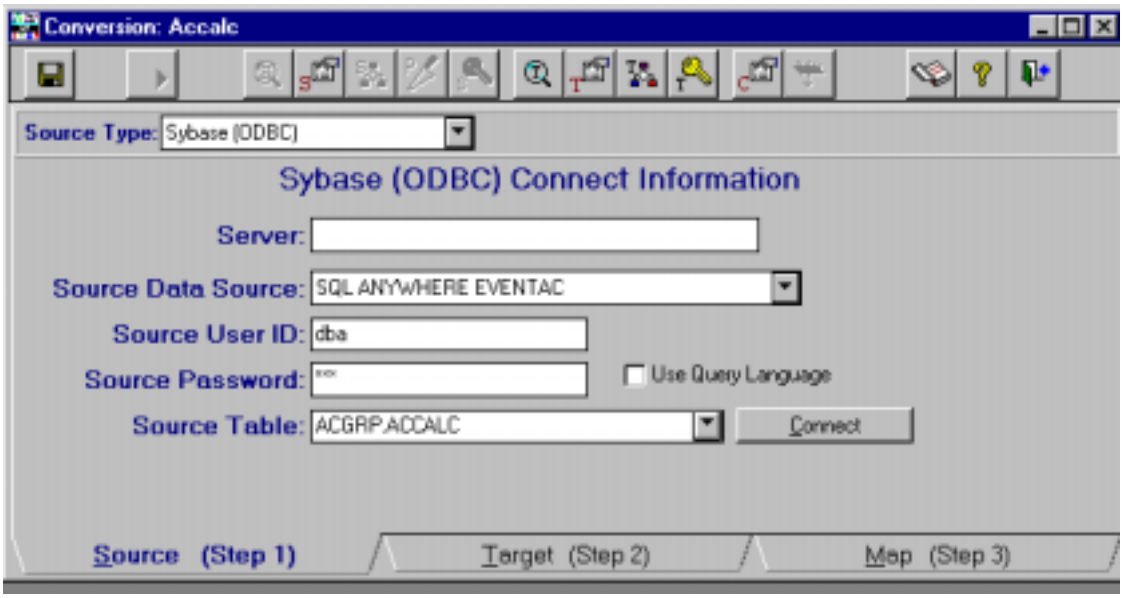
### 4.4.3 Flytta data med Data Junction till Oracle

Data Junction är ett verktyg för Windows med vars hjälp data kan flyttas mellan en stor mängd av olika filtyper. För att förflytta data med Data Junction designar du och kör en så kallad översättning, engelskans conversion. Varje översättning innehåller all information som Data Junction behöver för att flytta data från en existerande fil eller tabell till en ny målfil eller tabell. Vår källdatabas innehåller 210 tabeller därför måste 210 olika översättningar skapas i Data Junction.

**Skapa en ny översättning**

En ny översättning skapas enligt hjälpfilerna till Data Junction i tre steg. Först väljs datakällan, sedan målkällan och slutligen sker en mappning av datatyperna.

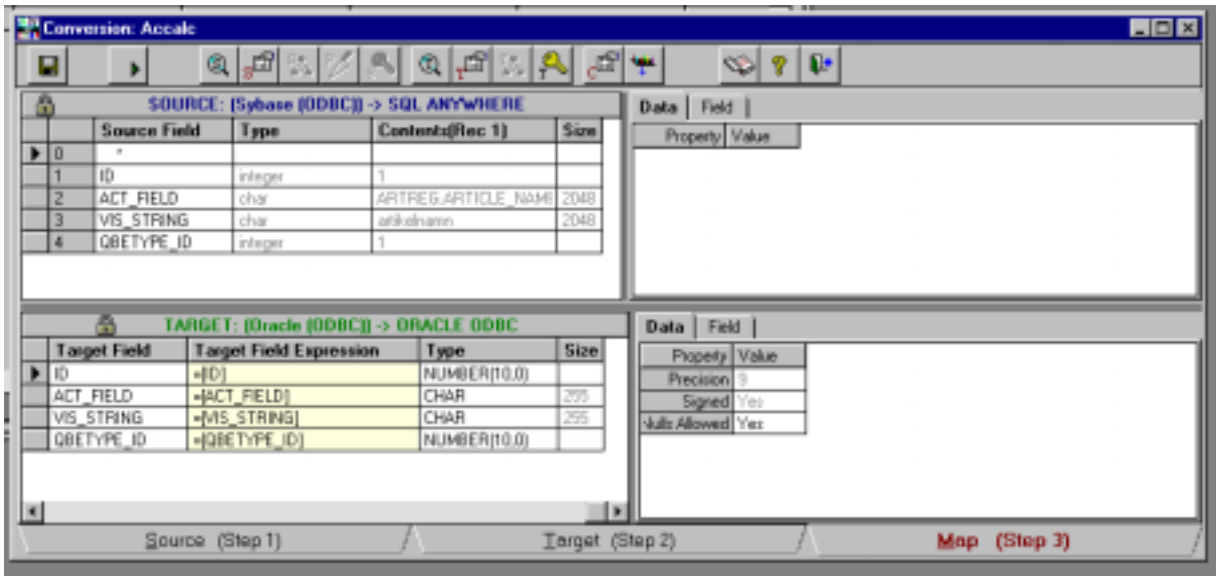
I steg ett valde vi en datakälla av typ Sybase ODBC och ställde in all information som krävdes för att ansluta till Sybase via ODBC-drivrutinen. Denna installeras och konfigureras under ODBC datakällor i Windows NT. Nedan ses inställningarna för tabellen accalc vilka för övrigt är samma för de andra tabellerna också.




*Figur 4.4: Data Junction, steg ett av en översättning*

Steg två följer samma mönster som steg 1 fast här ska målkällan anges istället, dvs Oracle ODBC. Steg två ser även ut ungefär som figur 5-2 men med de skillnaderna att datakällan är Oracle ODBC istället för Sybase ODBC och att det dessutom finns ett till fält. I detta fält sker inställningen av om en existerande tabell ska ersättas, om bara datan ska ersättas, om data ska läggas till i slutet på en existerande tabell etc. Det finns fyra olika alternativ om den valda datakällan är ODBC.

I steg 3, se Figur 4.5, sker mappningen av data mellan de två olika tabellerna. Här kan man se typerna i ursprungstabellen och de olika kolumnernas egenskaper, det vill säga om det är heltal med ett visst antal decimaler etc. Om det finns en måltabell, som i vårt fall, så syns dess egenskaper i targetrutan. Om tabellen inte finns måste en ny tabell skapas. Detta kan ske genom en kopiering av egenskaperna för tabellen man vill flytta eller genom att göra inställningarna manuellt, rad för rad, i tabellen. Inställningsmöjligheterna varierar med valda alternativ i steg två. Vi valde att ersätta enbart data och kunde följaktligen inte ändra datatyperna i måltabellen eftersom dessa blev låsta. Att tabellens egenskaper är låsta beror på att vi valde att endast ersätta data i tabellen och följaktligen är tabellens egenskaper redan fastställda av den redan existerande tabellen.




*Figur 4.5: Data Junction, steg tre av en översättning*

**Att skapa och köra en batch**

Sista steget av dataöverföringen görs lättast genom att skapa en batch i Data Junction. En batch är en mängd konverteringar som körs efter varandra utan mänsklig hjälp. Detta steg genomförs utan problem om inte programmet rapporterar några fel. Tiden för det hela varierar beroende på mängden data, men man får räkna med att det tar några timmar.

Vi fann att vi inte kunde genomföra detta utan att fel uppstod i några av tabellerna. Felet bestod i att Data Junction och Oracle klagade på datatyper som inte matchade. Detta kontrollerades och de datatyper som fanns i Oracle motsvarades av de som fanns i Sybase. Enligt Oracles dokumentation finns det stöd för dessa datatyper i Oracle så dataöverföringen ska fungera utan fel, men i vårt fall så uppstod det problem.

Problemet bestod av ett minustecken i en tabell och en decimalpunkt i de övriga. Då datatyperna i Oracle stödjer både minus och tio decimaler, samma som i Sybase för övrigt, var detta svårt att förklara. Vi löste problemet genom att ändra det negativa talet till ett positivt tal, göra överföringen och sedan ändra tillbaka till ett negativt tal i Oracle. Problemet med decimalerna löste vi genom att ta bort decimalerna då de ändå bara innehöll nollor. Sedan gick det bra att köra batchen.

Eftersom Oracle stöder de datatyper som orsakade våra problem och det gick bra att ändra datatyperna i Oracle efter överföringen så misstänker vi att problemet inte låg i Oracle utan i programmet som skulle flytta över datat, i vårt fall Data Junction. Vår tes om att det skulle vara Data Junction som orsakade problemen föll när det visade sig att samma problem

uppstod då vi försökte med en exakt likadan överföring med hjälp av Access och fick samma fel. Detta får oss att tro att felet ligger i någon av ODBC-drivrutinerna. Vi brydde oss aldrig om att ladda hem en nyare drivrutin då det visade sig att det var svårt att få tag på rätt version. Därför kunde vi inte testa om det var ODBC-drivrutinerna som det var fel på.

### 4.4.4 Resultat

Slutligen så tänkte vi pröva vår nya Oracledatabas mot Even(t) AC så vi började med att installera alla filer för att kunna köra Even(t) AC. Sedan fortsatte vi med att konfigurera ODBC-källan, det vill säga konfigurera AC att köra mot Oracle ODBC istället för Sybase. För att lyckas med att starta Even(t) AC överhuvudtaget var vi tvungna att skapa användaren ”adm” i Oracle. Detta för att Even(t) AC använder ”adm” för att ansluta till databasen. Vi fick mycket riktigt igång systemprogrammet, men sedan var det inte mycket som fungerade. Detta var inte helt förvånande med tanke på de fel som uppstod under överföringen beskrivna i tidigare avsnitt. Så även om vi kom ganska långt med denna lösningen så kom vi inte riktigt ända fram. Med manuella justeringar i databasen efteråt kan förhoppningsvis ett fungerande system uppnås.

## 4.5 Sammanfattning

I det här kapitlet har vi fokuserat på olika lösningar för att skapa en Oracledatabas utifrån en redan existerande Sybasedatabas. De metoder vi använt för att flytta *data* från en databas till en annan är: SQL*Loader, Data Junction, Microsoft Access och Sybasefunktionen export database. Att flytta data är den mindre problemfyllda delen av uppgiften. Den riktigt svåra delen av uppgiften är att flytta databasschemat från en databas till en annan. För att lösa detta har vi provat flera alternativa lösningar bland annat SQL*Loader, Migration Workbench, Clarion och Sybasefunktionen export database.

# 5 Slutsatser

Vi började vår uppgift med övertygelse om att flytta en databas från Sybase till Oracle skulle vara gjort på två kanske tre veckor. När vi flyttat databasen tänkte vi fortsätta med att undersöka och testa hur väl ODBC fungerar. Vår ursprungliga plan slogs snabbt i spillror när det visade sig att överföringen av databasen från Sybase till Oracle inte lät sig genomföras så lätt. Vår erfarenhet efter att provat flera metoder är att uppgiften är mycket svårlöst.

Det viktigt i sammanhanget är att det inte är data i tabellerna som är svåra att flytta. Med det menas att flytta data från en tabell i Sybase till en tabell i Oracle låter sig göras på ett smidigt sätt. Till detta kan flera olika program användas så som SQL*Loader för Oracle, Data Junction och Microsoft Access. Om det rör sig om många tabeller försvåras uppgiften men är ändå genomförbar.

Det svåra ligger i att flytta över databasschemat. Databasschemat är det som beskriver vilka tabeller som finns, nycklar i tabeller, kopplingar mellan tabellerna och så vidare. För att flytta databasschemat från en produkt till en annan har vi ej funnit någon generell metod.

Något som ytterligare försvårar uppgiften är alla de olika produkter som finns på marknaden idag. Sybase och Oracle har inte bara en databashanterare. Endast Sybase har minst två olika databashanterare med väldigt stora individuella skillnader, detta trots att namnen är förvillande lika. Av dessa två varianter finns sedan en uppsjö olika versioner. Den senaste versionen av Sybase SQL Anywhere är Adaptive Server Anywhere 6.0. Utöver denna server finns Sybase Adaptive Enterprise som är den mest avancerade och största av Sybase databassystem. Adaptive Enterprise finns idag i version 11 och 12.

Sammantaget så gör alla dessa produkter att det som fungerar mellan två av dem inte fungerar mellan två andra. Allt blir bara en stor förvirrande röra på grund av alla olika databashanteringssytem.

I Migration Workbench finns idag funktionalitet för att flytta en databas från Sybase Adaptive Server 11 till Oracle8i. Detta har inte hjälp oss då versionen som används på Analys-Consult är Sybase SQL Anywhere 5.0. Oracle arbetar dock på att ta fram en plugin till Migration Workbench som ska göra det möjligt att flytta databaser mellan Adaptive server Anywhere 6.0 och Oracle8i. Denna utveckling har för tillfället lagts på is på grund av att

Oracle har stött på tekniska problem. Detta visar än en gång att uppgiften att flytta databaser från en produkt till en annan inte är lätt att lösa.

Vad vi ser som en möjlig lösning är att använda sig av Migration Workbench när utvecklingen tar fart igen. Steg ett vore att uppgradera databasen från Sybase SQL Anywhere 5.0 till Adaptive Server Anywhere 6.0 och därefter använda Migration Workbench för att flytta databasen till Oracle8i. På detta sätt skulle databasschema och data gå att överföra i ett steg. Utvecklingen av tillägget till Migration Workbench är i dagsläget inte färdigutvecklat men detta tillägg kommer antagligen att göra överföring mellan Adaptive Server Anywhere 6.0 och Oracle8i möjlig. Denna hypotes har vi naturligtvis ännu inte kunnat styrka. Efter att ha arbetat med Sybase och Oracle en längre tid så anser vi att Migration Workbench förmodligen är den bästa lösningen.

Ett alternativ kunde vara att uppgradera Sybase SQL Anywhere 5.0 till Sybase Adaptive Server 11 men detta är ett steg som inte går att genomföra lika lätt som uppgradering till Adaptive Server 6.0 enligt Sybase support.

# 6 Slutord

Vi har lärt oss massor om olika databaser hur de fungerar, vilka problem som kan uppstå och hur lätta respektive svåra de är att administrera. Även om vi inte lyckades att lösa problemet helt och hållet så känns det inte som ett misslyckat examensarbete. Uppgiften som kändes konkret från början blev mer och mer en undersökning i hur kompatibla de olika databashanterarna är samt vilka verktyg det finns för att flytta data mellan olika databaser.

# Referenser


[1] Dataordboken, Svensk Standard SS 01 16 01 Utgåva 4

[2] Teach Yourself SQL in 21 Days, 2nd Edition, Ryan K.Stephens, Ronald R. Piew, Bryan Morgan, Jeff Perkins

[3] http://www.watz..se/odbc.html

[4] http://www.merant.com

[5] Novell NetWare 5; Overview and Installation. (manual).

[6] http://technet.oracle.com/software/tech/migration/workbench/software_index.htm

[7] Oracle8 Documentation, http://oradoc.photo.net/ora8doc/

# A Begrepp och förkortningar

| | |
|---|---|
| ANSI | American National Standards Institute |
| Clarion | Avser version 2.0 |
| CLI | Call-Level Interface |
| DB2 | Relationsdatabashanteringssystem |
| IBM | Internal Buisness Machines |
| IEC | International Electrotechnical Commission |
| ISO | International Standards Organization |
| ODBC | Open Database Connectivity |
| Oracle8 | Avser Oracle8 server release 8.0.3.0.6. |
| SQL | Structured Query Language |
| Sybase SQL Anywhere | Avser Sybase SQL Anywhere server release 5.0.03. |
| DLL | Dynamically Linked Libraries |

# B Exempel från Sybase unload database

Bilagorna placeras efter referenserna. Använd Alt+Q, W, E resp. R för att sätta rätt rubriktyp och numrering på bilagerubriker och första, andra resp. tredje nivå underrubriker Om du önskar att en bilaga ska börja på ny sida kan du använda Ctrl+Enter framför rubriken eller helst Format | Stycke | Textflöde | Sidbrytning före inuti rubriken.

## B.1 Utdrag från filen Reload.sql

```
% Usage:          isql read D:\UNLOADEDDB\RELOAD.SQL
%
% This command file reloads a database that was unloaded using "dbunload".
%
% (version:  5.5.01 Build #1333)
%

SET OPTION Statistics = 3
go
SET OPTION Date_order = 'YMD'
go

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%    Create userids and grant user permissions
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

GRANT CONNECT TO "DBA" IDENTIFIED BY "SQL"
go
GRANT RESOURCE, DBA, SCHEDULE TO "DBA"
go
GRANT CONNECT TO "dbo"
go
GRANT GROUP TO "dbo"
go
GRANT RESOURCE, DBA TO "dbo"
go
GRANT CONNECT TO "ACGRP" IDENTIFIED BY ENCRYPTED
'\xB9\x71\x13\x69\x3E\xC6\x5C\x47\x4C\x73\xDB\x00\xAB\x35\x70\x0E\x6C\xC4\xEC\x14\xC4\x66\xE1\xC
7\xB9\x04\x75\x66\xCA\xA1\x23\x3B\xE2\x6B\xFE\xE8'

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%    Create tables
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

GRANT MEMBERSHIP IN GROUP "ACGRP" TO "EVENTACGRP"
go
GRANT MEMBERSHIP IN GROUP "EVENTACGRP" TO "ADM"
go
GRANT MEMBERSHIP IN GROUP "EVENTACGRP" TO "USER1"
go
GRANT MEMBERSHIP IN GROUP "EVENTACGRP" TO "USER2"
```

```
go
GRANT MEMBERSHIP IN GROUP "EVENTACGRP" TO "USER3"
go
CREATE TABLE "EVENTACGRP"."CLIENTFILE"
(
                "ID"                                    integer NOT NULL,
                "CLIENT_NAME"                           varchar(50) NULL,
                "DATASOURCENAME"                        varchar(200) NULL,
                "TABLEOWNER"                            varchar(30) NULL,
                "DATAFILEPATH"                          varchar(200) NULL,
                "DATAFILEEXT"                           varchar(5) NULL,
                PRIMARY KEY ("ID")
)
go
CONNECT "DBA" IDENTIFIED BY "SQL"
go
GRANT SELECT, INSERT, DELETE, UPDATE, ALTER, REFERENCES ON
"EVENTACGRP"."CLIENTFILE" TO "EVENTACGRP"
go
CREATE TABLE "ACGRP"."ADDRESSFILE"
(
                "ID"                                    integer NOT NULL,
                "PERSONFILE_ID"                         integer NULL,
                "ADDRESSTYPE_ID"                        integer NULL,
                "ADDRESS1"                              varchar(50) NULL,
                "ADDRESS2"                              varchar(50) NULL,
                "POSTCODE"                              varchar(20) NULL,
                "CITY"                                  varchar(50) NULL,
                "REGION"                                varchar(50) NULL,
                "STATE"                                 varchar(20) NULL,
                "ADDRESSPOSITION"                       varchar(50) NULL,
                PRIMARY KEY ("ID")
)
go
GRANT SELECT, INSERT, DELETE, UPDATE, ALTER, REFERENCES ON "ACGRP"."ADDRESSFILE"
TO "EVENTACGRP"

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%    Reload data
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

LOAD TABLE "ACGRP"."ADDRESSFILE"
                FROM 'D:\\UNLOADEDDB\\UNLOAD\\192.dat'
                FORMAT 'ASCII'
                QUOTES ON ESCAPES ON STRIP OFF
                DELIMITED BY ','

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%    Add foreign key definitions
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

go
CREATE INDEX "ADD_ADD_PERSONFILE_ID_FK" ON "ACGRP"."ADDRESSFILE"
(
                "PERSONFILE_ID" ASC
)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%    Set option values
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
SET OPTION Statistics =
go
SET OPTION Date_order =
go


%
%SQL Option Statements for user
%

SET OPTION "DBA"."Statistics" = '3'
go
SET OPTION "DBA"."Date_order" = 'YMD'
go


%
%SQL Option Statements for user
%

SET OPTION "PUBLIC"."Blocking" = 'ON'
go
SET OPTION "PUBLIC"."Checkpoint_time" = '60'
go
SET OPTION "PUBLIC"."Conversion_error" = 'ON'
go
SET OPTION "PUBLIC"."Date_format" = 'YYYY-MM-DD'
go
SET OPTION "PUBLIC"."Date_order" = 'YMD'
go
SET OPTION "PUBLIC"."Isolation_level" = '3'
go
SET OPTION "PUBLIC"."Lock_rinteected_rows" = 'OFF'
go
SET OPTION "PUBLIC"."Precision" = '30'
go
SET OPTION "PUBLIC"."Recovery_time" = '2'

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%    Create user messages
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

commit work
go

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%    Create procedures
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

CONNECT "DBA" IDENTIFIED BY "SQL"
go
CONNECT "ACGRP"
go

create procedure
ACgrp.SP_RestAmountPerson(@Personfile_ID integer)
as
begin
  select SUM(InvoiceAmount-PaidAmount)
```

```
   from ORDERFILE
   where PERSONFILE_ID=@Personfile_ID
end
go
GRANT EXECUTE ON "ACGRP"."SP_RestAmountPerson" TO "EVENTACGRP"
go
CONNECT "DBA" IDENTIFIED BY "SQL"
go
commit work
go

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%    Create triggers
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

commit work
go

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%    Create SQL remote definitions
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

CREATE REMOTE TYPE "FILE" ADDRESS "
go
CREATE REMOTE TYPE "MAPI" ADDRESS "
go
CREATE REMOTE TYPE "VIM" ADDRESS "
go
CREATE REMOTE TYPE "SMTP" ADDRESS "
go
commit work
go

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%    Check view definitions
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

GRANT CONNECT TO "DBA" IDENTIFIED BY ENCRYPTED
'\x79\x43\x12\x86\x52\x19\xD5\xF7\xF5\xA2\xC3\xC5\x10\x8C\x77\x19\x6D\x01\xCB\xB5\xE9\x75\xDA\xF
B\xFD\xEA\x6E\xB8\xEB\xC0\x65\x55\xDC\xDA\x60\xDF'
go
commit work
go
```

## B.2 Utdrag från filen 192.dat

För att lättare få en överblick av filen har vi skrivit ut namnet på kolumnerna samt i vilken tabell de finns.

Namn på kolumnerna i tabellen ADDRESSFILE:

ID, PERSONFILE_ID, ADDRESSTYPE_ID, ADDRESS1, ADDRESS2, POSTCODE, CITY, REGION, STATE, ADDRESSPOSITION

1,1,2,'Gjuterigatan 28',,'65221','Karlstad',,,

2,1,1,'Gjuterigatan 28',,'65221','Karlstad',,,

16,6,2,'Norra Stationsgatan 93','Box 23009','10435','Stockholm',,,

19,7,2,'Kvarngatan 22 Box 264',,'68625','Sunne',,,

20,7,1,'Kvarngatan 22 Box 264',,'68625','Sunne',,,

33,10,3,'Box 596',,'65109','Karlstad',,,

34,11,2,'Box 10165','Kabelgatan 10','43422','Kungsbacka',,,

35,11,1,'Box 10165','Kabelgatan 10','43422','Kungsbacka',,,

# C Utdrag ur de sex sqlfilerna skapade av Clarion

## C.1 Book_tbl.sql före korrigeringar

```
--+------------------------------------------------------------
SPOOL BOOK_LOG.SQL
--+------------------------------------------------------------
--+------------------------------------------------------------
--+ File: ACC8 - ACCALC (File Definition)
--+ Desc: Accounting (-rules) Calculators
--+------------------------------------------------------------

CREATE TABLE ACCALC
    (
     ID          NUMBER  (  10),              --+ LONG,NAME('ID')
     ACT_FIELD    CHAR    (2048),              --+ STRING(2048),NAME('ACT_FIELD')
     VIS_STRING   CHAR    (2048),              --+ STRING(2048),NAME('VIS_STRING')
     QBETYPE_ID   NUMBER  (  10)               --+ LONG,NAME('QBETYPE_ID')
    )
    STORAGE ( INITIAL     16K
              NEXT        16K
              MINEXTENTS  1
              MAXEXTENTS  99
              PCTINCREASE 0 )
TABLESPACE USER_DATA;

NAME('"' & CLIP(TableOwner) & '".' & '"ACCALC"')

ALTER TABLE ACCALC
ADD (CONSTRAINT ACC8_ID_PK PRIMARY KEY (ID|
 )
    USING INDEX TABLESPACE USER_DATA
    STORAGE ( INITIAL     16K
              NEXT        16K
              MINEXTENTS  1
              MAXEXTENTS  99
              PCTINCREASE 0 )
   );

CREATE INDEX ACC8_QBETYPE_ID_FK ON ACCALC (QBETYPE_ID)
STORAGE ( INITIAL     16K
          NEXT        16K
          MINEXTENTS  1
          MAXEXTENTS  99
          PCTINCREASE 0 )
TABLESPACE USER_DATA;

COMMIT;
```

## C.2 Book_tbl.sql efter korrigeringar

```
--+----------------------------------------------------------
SPOOL BOOK_LOG.SQL
--+----------------------------------------------------------
--+----------------------------------------------------------
--+ File: ACC8 - ACCALC (File Definition)
--+ Desc: Accounting (-rules) Calculators
--+----------------------------------------------------------

CREATE TABLE ACCALC
    (
     ID          NUMBER  (  10),              --+ LONG,NAME('ID')
     ACT_FIELD    CHAR    (2000),              --+ STRING(2048),NAME('ACT_FIELD')
     VIS_STRING   CHAR    (2000),              --+ STRING(2048),NAME('VIS_STRING')
     QBETYPE_ID   NUMBER  (  10)               --+ LONG,NAME('QBETYPE_ID')
    )

TABLESPACE USER_DATA;

ALTER TABLE ACCALC
ADD (CONSTRAINT ACC8_ID_PK PRIMARY KEY (ID)
    USING INDEX TABLESPACE USER_DATA
);

CREATE INDEX ACC8_QBETYPE_ID_FK ON ACCALC (QBETYPE_ID)
TABLESPACE USER_DATA;

COMMIT;
```

## C.3 Book_fky.sql

```
--+----------------------------------------------------------
SPOOL BOOK_LOG.SQL
--+----------------------------------------------------------
--+----------------------------------------------------------
--+ File: ACC8 - ACCALC (Foreign Key Definitions)
--+ Desc: Accounting (-rules) Calculators
--+----------------------------------------------------------
ALTER TABLE ACCALC
ADD (CONSTRAINT FK_Q_T_ACC8_QBETYPE_ID_FK FOREIGN KEY (Qbetype_ID)
    REFERENCES QBETYPE (ID)
   );

COMMIT;

--+----------------------------------------------------------
--+ File: ACD2 - ACCDAY (Foreign Key Definitions)
--+ Desc: **** Account per day
--+----------------------------------------------------------
ALTER TABLE ACCDAY
ADD (CONSTRAINT FK_ACP_ACD2_ACCPERIOD_ID_FK FOREIGN KEY (AccPeriod_ID)
    REFERENCES ACCPERIOD (ID)
   );

COMMIT;

--+----------------------------------------------------------
--+ File: AAC - ACCDAYCUR (Foreign Key Definitions)
--+ Desc: **** Account per day currency
--+----------------------------------------------------------
ALTER TABLE ACCDAYCUR
ADD (CONSTRAINT FK_CUR_AAC_CURRENCY_ID_FK FOREIGN KEY (Currency_ID)
    REFERENCES CURRENCY (ID)
   );

ALTER TABLE ACCDAYCUR
ADD (CONSTRAINT FK_ACD2_AAC_ACCDAY_ID_FK FOREIGN KEY (Accday_ID)
    REFERENCES ACCDAY (ID)
    ON DELETE CASCADE
   );

COMMIT;
```

# D Oracle felkoder

**ORA-01450 maximum key length exceeded**

Cause: The combined length of all the columns specified in a CREATE INDEX statement exceeded the maximum index length. The maximum index length varies by operating system. The total index length is computed as the sum of the width of all indexed columns plus the number of indexed columns. Date fields have a length of 7, character fields have their defined length, and numeric fields have a length of 22. Numeric length = (precision/2) + 1. If negative, add +1.

Action: Select columns to be indexed so the total index length does not exceed the maximum index length for the operating system. See also your operating system-specific Oracle documentation.

**ORA-01408 such column list already indexed**

Cause: A CREATE INDEX statement specified a column that is already indexed. A single column may be indexed only once. Additional indexes may be created on the column if it is used as a portion of a concatenated index, that is, if the index consists of multiple columns.

Action: Do not attempt to re-index the column, as it is unnecessary. To create a concatenated key, specify one or more additional columns in the CREATE INDEX statement.

**ORA-00957 duplicate column name**

Cause: A column name was specified twice in a CREATE or INSERT statement. Column names must be unique within a table, view, or cluster.

Action: In a CREATE statement, change one of the column names to a new, unique column name. In an INSERT statement, remove one of the duplicate names.

**ORA-00942 table or view does not exist**

Cause: The table or view entered does not exist, a synonym that is not allowed here was used, or a view was referenced where a table is required. Existing user tables and views can be listed by querying the data dictionary. Certain privileges may be required to access the table. If an application returned this message, the table the application tried to access does not exist in the database, or the application does not have access to it.

Action: Check each of the following:

- the spelling of the table or view name.
- that a view is not specified where a table is required.
- that an existing table or view name exists. Contact the database administrator if the table needs to be created or if user or application privileges are required to access the table.

Also, if attempting to access a table or view in another schema, make certain the correct schema is referenced and that access to the object is granted.

**ORA-02270 no matching unique or primary key for this column-list**

Cause: An attempt was made to reference a unique or primary key in a table with a CREATE or ALTER TABLE statement when no such key exists in the referenced table.

Action: Add the unique or primary key to the table or find the correct names of the columns with the primary or unique key, and try again.

**ORA-01722 invalid number**

Cause: The attempted conversion of a character string to a number failed because the character string was not a valid numeric literal. Only numeric fields or character fields containing numeric data may be used in arithmetic functions or expressions. Only numeric fields may be added to or subtracted from dates.

Action: Check the character strings in the function or expression. Check that they contain only numbers, a sign, a decimal point, and the character "E" or "e" and retry the operation.