

# Der Entwickler 01/1999

## Werkzeuggestützter Entwurf von SQL Datenbanken

von Andreas Tengicki und Henry Wolf

- |   |   |
|---|---|
| <a href="#">1. Theorie relationaler Datenbanken</a>         | <a href="#">5. Marathon und InterBase Difference Finder</a> |
| <a href="#">2. Anforderungen an DB-Management Werkzeuge</a> | <a href="#">6. Marathon</a>                                 |
| <a href="#">3. Pflege von bestehenden Datenbanken</a>       | <a href="#">7. InterBase Difference Finder</a>              |
| <a href="#">4. Automatische Applikationsgenerierung</a>     | <a href="#">8. Vergleich der Konzepte</a>                   |

Bei der Entwicklung von Datenbank Anwendungen speziell von Client-Server Anwendungen ist neben der Implementation der Oberfläche ( Client ) die Programmierung der Datenbank (Server) notwendig. Dabei sollte auf einige theoretische Aspekte Rücksicht genommen. Bei der Praxis der Implementierung wiederum existieren verschiedene mächtige Werkzeuge, zur Unterstützung des Entwicklers. Im folgenden wollen wir sowohl auf die theoretischen Grundlagen als auch auf zwei typische Werkzeugvertreter eingehen.

### 1. Theorie relationaler Datenbanken

Alle modernen Datenbankmanagementsysteme beruhen auf dem Modell der **relationalen Datenbank** ( entsprechend Definition laut E.F. Codd 1970 ). Andere Modelle für Datenbanken haben sich für viele Problemstellungen gegenüber dem relationalen Modell unterlegen erwiesen ( netzartige Datenbanken mit Einschränkungen hierarchische Datenbanken ) bzw. sind in Ihrer Verbreitung äußert beschränkt. ( objektorientierte Datenbanken ). Besonders das Modell der objektorientierten Datenbanken wird im Zusammenspiel mit den relationalen Datenbankmodell ( Oracle ORDBMS 8, POET u.a. ) zukünftig eine große Rolle spielen. Den Stand der Technik im Praxiseinsatz heute stellen allerdings nach wie vor relationale Datenbanken dar. Im folgenden sollen für das bessere Verständnis einige Grundzüge des Aufbaus eines relationalen Datenbankmanagementsystems dargestellt werden.

Das Basisobjekt einer relationalen Datenbank ist die **Tabelle**. Jede relationale Datenbank besteht aus der Sicht des Nutzers nur aus Tabellen ( logische Sicht ). Eine Tabelle setzt sich aus ein oder mehreren Spalten ( Feldern ) bzw. aus ein oder mehreren Reihen bzw. Zeilen ( Datensatz / Rows ) zusammen. Das Objekt, welches zu einer Reihe und einer Spalte eindeutig zugeordnet wird, heißt Datenwert bzw. Datum. Für eine Tabelle gelten nach dem Modell der relationalen Datenbank folgende Regeln:

1. Die Reihen innerhalb einer Tabelle können eine beliebige Reihenfolge haben.
2. Alle Werte einer Spalte haben genau den selben Datentyp.
3. Jede Spalte in einer Tabelle besitzt einen eindeutigen und in der Tabelle einmaligen Namen ( Feldbezeichner ).
4. Jedes Datum ist innerhalb der Tabelle durch einen einzigen Wert dargestellt.
5. In jeder Tabelle existiert ein ( oder mehrere ) Bezeichner, der jede Reihe ( Zeile ) der Tabelle eindeutig definiert. Dieser Bezeichner besteht aus ein oder mehreren Spalten. Dieser wird als Primärschlüssel bezeichnet.
6. In jeder Tabelle existieren nie zwei identische Reihen (4 Normalform).

Die Definition der Datenbank ( Datenbankdesign ) stellt eine wichtige Phase bei der Erstellung eines Anwendungssystems dar. Im Bereich der relationalen Datenbanken wird meist das Verfahren der Normalisierung angewandt. Dadurch wird die Redundanz der Daten stufenweise reduziert.

Gleichzeitig ist die logische Unabhängigkeit der Daten ein Ziel des Verfahrens. Das Verfahren der

Normalisierung umfaßt 4 Phasen und basiert auf den Tabellen.

1. **Erste Normalform**

eine Tabelle befindet sich in der ersten Normalform, wenn alle Spalten nur atomare Werte enthalten ( kein Schlüssel bzw. Index ).

2. **Zweite Normalform**

eine Tabelle befindet sich in der zweiten Normalform, wenn jede Spalte dieser Tabelle, welche nicht den Primärschlüssel bildet, voll funktional abhängig von dem Primärschlüssel ist.

3. **Dritte Normalform**

eine Tabelle befindet sich in der dritten Normalform, wenn zwischen den Spalten einer Tabelle, die nicht den Primärschlüssel bilden, keine Abhängigkeiten existieren. Ausgegangen wird dabei von Tabellen der zweiten Normalform. Datenbanken, welche in der dritten Normalform vorliegen, enthalten weitgehend nicht redundante Datenbestände.

4. **Vierte Normalform**

Diese beseitigt mehrwertige Abhängigkeiten in den Tabellen einer Datenbank. Seien z.B. Spalte A und B funktional von Spalte C abhängig, untereinander besteht jedoch keine Abhängigkeit. Dann erfolgt eine Trennung in zwei Tabellen. Tabelle 1: Spalte C + Spalte A; Tabelle 2 : Spalte C + Spalte B

Weitere Ausführungen zum Modell der relationalen Datenbank sollen an dieser Stelle nicht erfolgen. Es sei jedoch erwähnt, daß die Sinnhaftigkeit eines Datenbankmodells in der vierten Normalform äußerst zweifelhaft ist. Normal ist die Implementierung eines Datenbankmodells in der Dritten Normalform.

Alle marktrelevanten Datenbanken unterstützen die Implementierung von Datenbankmodellen in der 3. Normalform durch verschiedene Mechanismen ( Primary Key, Foreign Keys, Trigger u.a. ).

Neben ehemaligen Datenhaltungssystemen wie z.B. dBase, werden in SQL-Datenbanken ( Oracle, Informix, Sybase u.a. ) innerhalb der Datenbanken bedeutend mehr Funktionalität als die reine Datenhaltung unterstützt. D.h., es sind zusätzliche Mechanismen und zahlreiche weitergehende Funktionen zur Unterstützung der Implementierung der 3. Normalform vorhanden. Diese Datenbanksysteme dienen nicht mehr der reinen Haltung von Daten, sondern verstehen sich als "Relationale Datenmanagementsysteme". An die Funktionalität solcher Systeme werden größere Anforderungen gestellt als an Datenhaltungssysteme.

- **Logische Datenunabhängigkeit**

Jeder Nutzer kann seine eigene logische Sicht auf die Daten erzeugen, ohne die logische Gesamtstruktur der Datenbasis zu ändern.

=> "Views"

- **Physikalische Datenunabhängigkeit**

Die physikalische Struktur der Datenbank kann beliebig geändert werden, ohne die logische Struktur zu berühren.

=> in einer solchen Datenbank werden Tabellen nicht als physikalische Datei, sondern als logische Einheit betrachtet

- **Prozedurale und nicht prozedurale Schnittstellen**

Dem Entwickler werden zur Erstellung von Anwendungen Programmiersprachen ( Schnittstellen ) zur Verfügung gestellt. Werden von der Schnittstelle Kontrollstrukturen ( z.B. IF..THEN..ELSE) angeboten, wird von einer prozeduralen Schnittstelle ( Programmiersprache ) gesprochen. Das herausragende Beispiel für die Implementierung einer solchen Datenbankbasierten Programmiersprache ist Oracle PL/SQL Dem Endbenutzer sind zur Erstellung von ad - hoc Anfragen Tools auf der Basis nicht prozeduraler Schnittstellen zur Verfügung zu stellen ( Report-Smith , Visual dBase C/S ,

ACCESS , Excel u.a. ).

- **Effiziente Abarbeitung von Datenbankoperationen**  
Das Datenbanksystem stellt effiziente Algorithmen und Optimierungsmöglichkeiten zur Realisierung von effizienten Datenbankoperationen zur Verfügung. Dies erfolgt entweder manuell durch Administrator, z.B. bei Parametern der Speicherverwaltung oder aber automatisch durch das System, z.B. beim internen SQL-Optimierer.
- **Minimale Redundanz**  
der Datenbestände Mehrfach erfaßte Daten verbrauchen mehr Speicher und ziehen einen erhöhten Pflegeaufwand nach sich. Das Datenbanksystem sollte Möglichkeiten zur Minimierung solcher Redundanzen bieten ( 3. Normalform des relationalen Datenbankmodells ).
- **Datenintegrität**  
Das Datenbanksystem muß offensichtlich unsinnige Daten erkennen und zurückweisen bzw. die Definition von Schranken und Regeln über die Sinnhaftigkeit von Daten erlauben. Neben solchen Schranken sollten Formatangaben in Bezug auf die zu erfassenden Daten möglich sein ( z.B. Datum, Zeichensatz, u.a. Unterstützung der jeweiligen Länderspezifika ).
- **Unterstützung konkurrierender Zugriffe auf den Datenbestand**  
Im Regelfall greifen mehrere Benutzer auf Datenbestände mit dem Zweck Daten zu lesen, bzw. zu manipulieren zu. Alle sich daraus ergebenden Konsequenzen hinsichtlich der Verwaltung solcher Zugriffe unter besonderer Berücksichtigung der Integrität des Datenbestandes und der Transparenz des Informationsgehaltes von Zugriffen auf den Datenbestand sind hierbei zu beachten.
- **Datensicherheit / Datenschutz**  
Das Datenbanksystem muß die Daten gegen unerlaubten Zugriff schützen können, jeden Nutzer jedoch den Zugriff auf für ihn notwendige und erlaubten Datenbestände sichern. Dies erfolgt über die Vergabe von entsprechenden Rechten auf alle Objekte einer Datenbank.
- **Hohe Verfügbarkeit der Datenbestände**  
Die Wartung der Datenbestände ( Backup, Recovery ) ist durch ein Datenbanksystem effizient zu lösen.

Tabelle 1 : Beispiel zur Umsetzung des relationalen Datenbankmodells durch Oracle

Eigenschaft	Umsetzung in Oracle V7/8	InterBase 5.
Logische Datenunabhängigkeit	Tabellen, Views, Synonyme, Datenbanklinks u.a.	Tabellen, Views, Stored Procedures
Physikalische Datenunabhängigkeit	Trennung von physikalischer ( Systemdateien ) und logischer Datenbankstruktur ( Tablespace )	( auf die physikalische Struktur hat der DBA keinen Einfluß )
prozedurale und nicht prozedurale Schnittstellen	Unterstützung der Datenbanksprache SQL und deren prozedurale Erweiterung durch Oracle PL/SQL	Unterstützung der Datenbanksprache SQL und einer prozedurale Erweiterung durch InterBase
effiziente Abarbeitung von Datenbankoperationen	Automatische Optimierung von Datenbankzugriffen manuelle Optimierungsmöglichkeiten, Analysewerkzeuge	Automatische Optimierung von Datenbankzugriffen manuelle Optimierungsmöglichkeiten

minimale Redundanz von Datenbeständen	Unterstützung des relationalen Datenbankmodells	
Datenintegrität	Commit ( fest schreiben ) Rollback ( Änderungen von nicht vollständig durchgeführten Transaktionen rückgängig machen ) <ul style="list-style-type: none"> <li>• Referenzielle Integrität</li> <li>• Two-Phase-Commit</li> </ul> Unterstützung länderspezifischer Datenformate	
Unterstützung konkurrierender Zugriffe auf Datenbestände	durch interne Verwaltung und explizite durchzuführende Funktionen realisiert	Realisierung durch Multi-Generations-Architektur, diese bewirkt optimistisches Locking
Datensicherheit / Datenschutz	Unterstützung des C2-Standards Rollen - ( Gruppen ) Konzept bei der Nutzerverwaltung	
Hohe Verfügbarkeit der Datenbestände	On-Line Sicherung Online - Verwaltung der Datenobjekte	On-Line Sicherung
Übereinstimmung mit Industriestandards	Unterstützung aller gängigen Industriestandards ( ANSI, SQL, ODBC, BDE, OLE u.a. )	
Portabilität	Verfügbar auf allen gängigen Plattformen, vom Großrechner bis hin zum Einzel-PC unter DOS/Windows	
Skalierbarkeit	Portabilität, protokollunabhängiges Kommunikationsmodul, Datenaustausch zwischen Datenbanken auf unterschiedlichen Plattformen problemlos möglich ( Export/Import ), Multiprozessorunterstützung	Portabilität, Datenaustausch zwischen Datenbanken auf unterschiedlichen Plattformen problemlos möglich ( Export/Import ), Multiprozessorunterstützung
Zusammenarbeit über Systemgrenzen hinweg	Protokollunabhängiges Kommunikationsmodul SQL*NET, Gateways zu anderen Datenbankmanagementsystemen, Unterstützung der Standards ODBC/IDAPI	Gateways zu anderen Datenbankmanagementsystemen, Unterstützung der Standards ODBC/IDAPI

## 2. Anforderungen an DB-Management Werkzeuge

Die oben beschriebenen Grundfunktionalitäten müssen nicht nur von dem Datenbanksystem zur Verfügung gestellt werden, sie müssen auch angewendet werden. Sowohl die Implementierung als auch besonders die Pflege eines applikationsspezifischen Datenbankmodells macht die Bearbeitung sehr vieler einzelner Datenbankobjekte erforderlich d.h. der Entwurf und die Pflege einer Datenbank umfaßt weit mehr als den reinen Entwurf der Tabellen und ihrer Beziehungen. Hierfür sind, gerade bei komplexen Anwendungssystemen ( Data Warehouse ) effiziente Werkzeuge notwendig. Ebenso ist die Frage wichtig, was mit bereits existierenden Datenbanken passiert. Kann ich eine existierende Datenbank mit einem solchen Werkzeug bearbeiten / pflegen?

Bei all dem ist sowohl die physikalische als auch die logische Anordnung von Daten und deren Managementfunktionalitäten zu berücksichtigen.

Eine besondere Rolle spielt die häufig auftretende Forderung nach "Datenbank unabhängiger Programmierung", welche von zahlreichen Herstellern von Entwicklungswerkzeugen mehr oder weniger sinnvoll unterstützt wird. Dabei stellt der ungleiche Funktionsumfang, der unterschiedliche Sprachumfang sowie eine teilweise unterschiedliche Syntax eine hohe zusätzliche Forderung an entsprechende Werkzeuge dar. SQL wurde zwar in verschiedenen Gremien zu verschiedenen Zeitpunkten standardisiert, doch dessen ungeachtet implementiert jeder Datenbankhersteller seine eigenen Zusatzfeatures, die die Kompatibilität beeinträchtigen.

Hierbei sei erwähnt, daß Anwendungen teilweise parallel für Desktopdatenbanken und für SQL-Datenbanken entwickelt werden und dies häufig frei nach dem Motto: "Wir suchen den kleinsten gemeinsamen Nenner zwischen dBase und Oracle".

Diesen nicht ganz unproblematischen Ansatz wollen wir zu einem späteren Zeitpunkt untersuchen. An dieser Stelle möchten wir uns zwei sehr unterschiedlichen Werkzeugen zur Unterstützung der Implementierung von DB-Modellen beschäftigen. Dabei sollen die Werkzeuge die oben aufgeführten Anforderungen an einen Datenbankentwurf bzw. an eine Datenbank unterstützen. Die Implementation eines Datenbankmodells erfolgt in jedem Fall plattformspezifisch.

### **Powerdesigner 6.1 ( S-Designer )**

Die Betrachtung möglicher Werkzeuge zur Erstellung und Pflege von Datenbankmodellen beginnen wir mit dem Powerdesigner der Firma Sybase / Powersoft. Der Powerdesigner ist die Familie von integrierten Werkzeugen, früher unter dem Namen S-Designer/DataArchitect bekannt, zur werkzeuggestützten Entwicklung von komplexen Datenbankanwendungen aus dem Hause Sybase bestehend aus ProcessAnalyst, WarehouseArchitect, DataArchitect, AppModeller und MetaWorks.

Der DataArchitect ist das Werkzeug zum Design von Datenbankmodellen.

Als eine sinnvolle Ergänzung hierfür ist der AppModeller, welcher zur Generierung von Formularen für entsprechende Entwicklungswerkzeuge wie Delphi dient.

Die Werkzeuge ProcessAnalyst und WarehouseArchitect dienen zur Unterstützung der Abbildung von Prozeß und Datenflusswegen in komplexen Datenbankanwendungen, bzw. zur Unterstützung bei der Entwicklung von Anwendungssystemen mit dem Anspruch eines Data Warehouse ( OLTP u.a. ).

MetaWorks unterstützt die Teamarbeit an Projekten. Es erzeugt ein eigenes Dictionary auf deren Basis alle Mitarbeiter des Teams gemeinsam an einem Projekt arbeiten können.

Aus Sicht der gewählten Aufgabenstellung möchten wir uns an dieser Stelle mit den Produkten DataArchitect und AppModeller beschäftigen.

Der DataArchitect dient der graphisch unterstützten Entwicklung von Datenbankmodellen.

### **Hierbei bietet der DataArchitect drei Vorgehensweisen an:**

1. Reverse Engineering ? Integration vorhandener Datenbanken in das Werkzeug, vorhandene Datenbanken werden ausgelesen, die Informationen ausgewertet und zur weiteren Bearbeitung zur Verfügung gestellt. Ergebnis ist ein DBM-system spezifisches Datenbankmodell.
2. DBMS-spezifisches Datenbankdesign ? Für mehr als 30 verschiedene DBM-System ( dBase, Oracle, InterBase u.a. ) können Datenbankmodelle erzeugt, bearbeitet und gepflegt werden.
3. DBMS-unabhängiges Datenbankdesign - Es wird ein DBMS-unabhängiges logisches oder konzeptionelles) Datenbankmodell erstellt, aus welchem später in die Datenbank spezifischen (physikalischen Modelle für die jeweilige Plattform erzeugt werden können.

Physikalisches und konzeptionelles Datenbankmodell unterscheiden sich in erster Linie dadurch, daß Tabellen mit virtuellen Datentypen erstellt werden und bei der Umwandlung in ein physikalisches Datenbankmodell durch die jeweiligen DBM-System spezifischen Datentypen ersetzt werden.

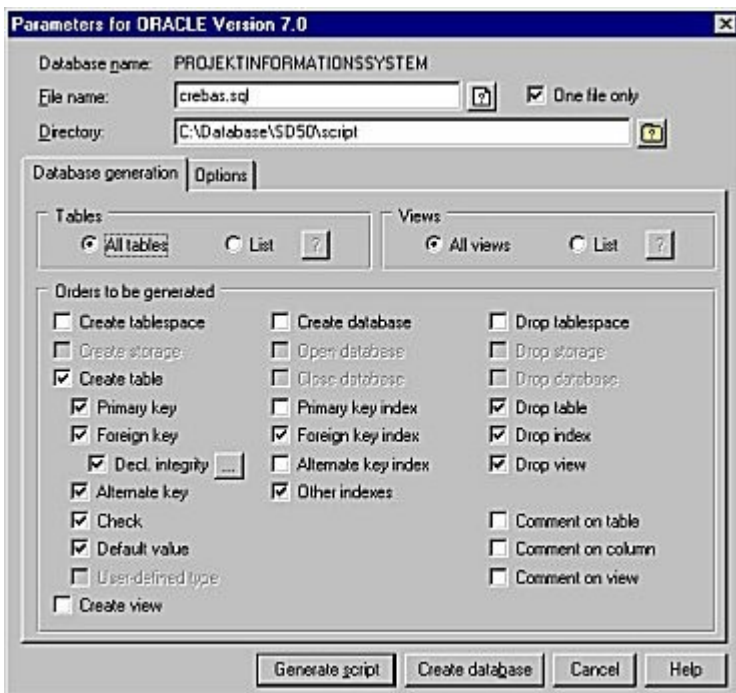
Es ist auch möglich, aus jedem physikalischen Datenbankmodell ein logisches Datenbankmodell zu erzeugen. Somit ist in diesem Werkzeug eine hervorragende Unterstützung zur Entwicklung von Anwendungen auf verschiedenen DB-Plattformen und zur Portierung von Datenbankmodellen auf andere DB-Plattformen gegeben.

Der DataArchitect unterstützt die Implementierung von allen wichtigen Datenbankobjekten, der gesamten Logik und Physis einer Datenbank, im physikalischen und konzeptionellen Modell sowie die Generierung von Vorlagen für Datentypen von Tabellenspalten, Triggern, Prozeduren u.v.a..

Außerdem bietet er einige Prüfroutinen zur Stimmigkeit des DB-Modells an. Diese Prüfungen beinhalten keine Syntaxprüfungen für die spezielle Datenbank. Solche Fehler werden erst beim Prozeß des eigentlichen Erzeugens der Datenbank sichtbar.

Auf der Grundlage des physikalischen Datenbankmodells werden entsprechend des Datenbanksystems Scripte zur eigentlichen Erzeugung der Datenbank generiert. Die Erzeugung der eigentlichen Datenbank kann direkt aus dem Werkzeug heraus erfolgen oder auf der Basis der Scripte erfolgen. Für SQL-Datenbanken werden entsprechende SQL-Scripte erzeugt, für andere z.B. dBase Scripte in der entsprechenden Datenbanksprache ( eben dBase ) erzeugt.

### 3. Pflege von bestehenden Datenbanken



Ein Highlight im DataArchitect ist komfortable Pflege von Datenbankmodellen. Durch Archivierung wird ein bestimmter Stand der Entwicklung fixiert. Auf der Basis dieses Standes werden bei Änderung des Datenbankmodells komfortable Algorithmen ( Scripte ) zur Anpassung bestehender Datenbanken erzeugt.

### 4. Automatische Applikationsgenerierung

Der AppModeller erzeugt auf der Basis des DB-Modells Formulare zur einfachen Bearbeitung der Daten. Hierbei werden verschiedenen Entwicklungstools unterstützt.

So z.B. Delphi 2, Powerbuilder, Visual Basic und Power++.

Die Qualität der erzeugten Formulare entspricht nicht unbedingt den hohen Ansprüchen an ein

perfektes Design von Anwendungen. Dieses kann jedoch durch Nachbearbeitung in den entsprechenden Werkzeugen individuell verbessert werden.

Für die Internetentwicklung wird MS-Active-Serverpages und der Sybase Power-Dynamoserver 2.0 unterstützt.

## 5. Marathon und InterBase Difference Finder

Mit dem Powerdesigner wurde eine Entwicklungsumgebung vorgestellt, die sowohl das logische als auch das physikalische Modell unterstützt. Durch individuelle Konfigurationsfiles ist der Powerdesigner für verschiedene Datenbanksysteme einsetzbar.

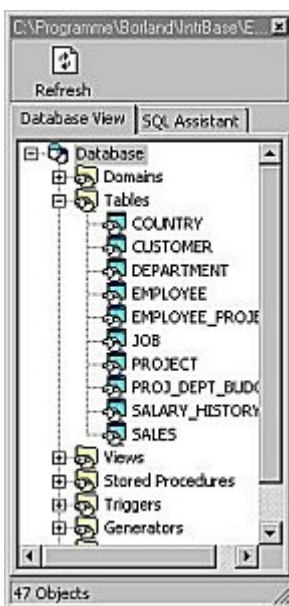
Einen völlig anderen Ansatz verfolgt Marathon von Gimbal. Dies ist ein SQL-Tool speziell für InterBase, das direkt auf dem physikalischen Modell operiert. Die Nachteile liegen damit direkt auf der Hand.

- Unterstützung einer einzigen Datenbank
- kein logisches Datenmodell

Zudem enthält Marathon keine Möglichkeit ein Upgrade-Script analog zur Archivierung beim Powerdesigner zu erstellen. Aus diesem Dilemma wird uns der InterBase Difference Finder befreien.

## 6. Marathon

Für den Einsatz von Marathon spricht der im Vergleich zum Powerdesigner geringe Preis sowie die einfache Handhabung. In einem zentralen Explorer Fenster werden alle Datenbankobjekt in einem TreeView dargestellt.



Aus diesem Explorer heraus können dann die einzelnen Objekte bearbeitet werden, z.B. eine Tabelle. Dabei stehen mehrere Ansichten zur Verfügung, z.B. die Definition

Field Name	Field Type	Domain	Field Nullable Flag	Default Source
CUST_NO	INTEGER	CUSTNO	False	
CUSTOMER	VARCHAR(25)	RDB\$18	False	
CONTACT_FIRST	VARCHAR(15)	FIRSTNAME	True	
CONTACT_LAST	VARCHAR(20)	LASTNAME	True	
PHONE_NO	VARCHAR(20)	PHONENUMBER	True	
ADDRESS_LINE1	VARCHAR(30)	ADDRESSLINE	True	
ADDRESS_LINE2	VARCHAR(30)	ADDRESSLINE	True	
CITY	VARCHAR(25)	RDB\$19	True	
STATE_PROVINCE	VARCHAR(15)	RDB\$20	True	
COUNTRY	VARCHAR(15)	COUNTRYNAME	True	
POSTAL_CODE	VARCHAR(12)	RDB\$21	True	
ON_HOLD	CHAR(1)	RDB\$22	True	DEFAULT NULL

und die Daten.

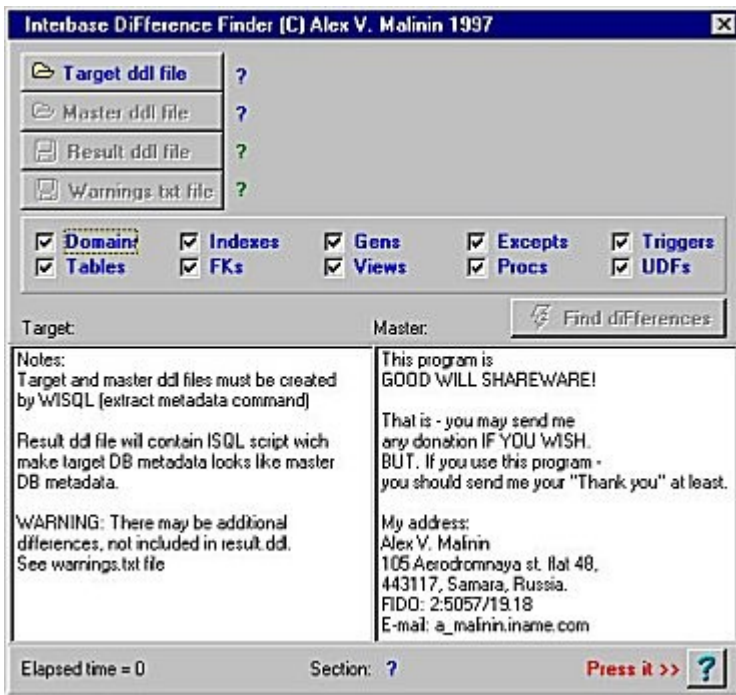
CUST_NO	CUSTOMER	CONTACT_FIRST	CONTACT_LAST	PHONE_NO
1001	Signature Design	Dale J.	Little	(519) 530-2710
1002	Dallas Technologies	Glen	Brown	(214) 960-2233
1003	Buttle, Griffith and Co.	James	Buttle	(517) 488-1864
1004	Central Bank	Elizabeth	Brocket	61 211 99 88
1005	DT Systems, LTD.	Tai	Wu	(852) 850 43 98
1006	DataServe International	Tomas	Bright	(513) 229 3323
1007	Mrs. Beauvais		Mrs. Beauvais	
1008	Anini Vacation Rentals	Leilani	Eriggs	(808) 835-7605
1009	Max	Max		22 01 23

In den anderen Seiten der Tabellendarstellung kann man die definierten Bedingungen (Constraints), Indizes, Abhängigkeiten und Trigger einsehen. Alle Datenbankobjekte haben zudem die Möglichkeit Dokumentationsinformationen zu hinterlegen. Insbesondere die Darstellung, der an der Tabelle definierten Trigger und die Übersicht der bestehenden Abhängigkeiten, d.h. es wird gezeigt in welchen Views, Prozeduren u.a. die Tabelle verwendet wird, bietet eine gute Navigationsmöglichkeit auch durch größere Datenbanken.

Ebenso wie für die Tabellen ist für alle anderen Datenobjekte eine Ansicht definiert. Dabei erfolgt eine Syntaxhervorhebung bei der Eingabe von SQL-Befehlen. Auch gibt es einen Experten, der einen bei Aufbau eines SQL-Statements behilflich ist.

## 7. InterBase Difference Finder

Mit den Werkzeugen von InterBase läßt sich ein Script File erstellen, daß alle Elemente der Datenbank enthält. Der InterBase Difference Finder von Alex.V. Malinin vergleicht zwei dieser Datenbankbeschreibungen und erstellt ein Script welches die Migration von einer Version zur nächsten realisiert. Damit existiert auch beim Arbeiten mit Marathon oder anderen Tools, für InterBase ein Werkzeug analog der Archivierung und Upgrade Erzeugung des Powerdesigners.



## 8. Vergleich der Konzepte

Der Unterschied zum Powerdesigner liegt in der Sicht auf die Dinge. Mit dem Powerdesigner wird ein konzeptioneller Blick auf das Datenbankkonzept geworfen und man hat die Möglichkeit auch bis in die kleinen Details alle Eigenschaften der Datenbank zu definieren.

Bei Marathon befindet man sich mitten in den Details und hat keine Möglichkeit das ganze Konzept zu visualisieren. Das Konzept muß sich in der Vorstellungskraft des Entwicklers befinden. Dafür wird man bei den Eigenheiten der speziellen Datenbank (hier InterBase) hervorragend unterstützt. Außerdem arbeitet man ohne Scripte und andere Hilfsmittel direkt auf der zu bearbeitenden Datenbank.

Damit dürfte auch deutlich werden wann welches Tool besser eingesetzt wird. Für die kontinuierliche Arbeit an großen Projekten, die Entwicklung, Darstellung und Dokumentation sind Produkte wie der Powerdesigner, die besseren Werkzeuge. Ähnlich wie der Powerdesigner arbeitet zum Beispiel noch das ER/Studio 3 von Embarcadero, sowie Erwin und einige andere Produkte, die insbesondere natürlich von den Datenbankhersteller und ihrem Umfeld angeboten werden. Diese Werkzeuge hier auch vorzustellen hätte den Rahmen des Artikels aber bei weitem überschritten und könnte unter Umständen einmal anlässlich einen entsprechenden Vergleiches stattfinden.

Marathon und andere SQL-Explorer unterstützen den Entwickler direkt bei seinem Arbeiten an der Datenbank und zielen weniger auf die konzeptionelle Ebene.