A BORLAND WHITE PAPER

OCTOBER 18, 2005

RE-INTRODUCTION TO BORLAND® INTERBASE®

COMPILED BY: MICHAEL J. ROZLOG
CHIEF TECHNICAL ARCHITECT

Borland®

Contents

Executive Summary:

Do you have a database? If you are like most of the companies in the world, the answer to that question would be yes. Can you remember what the reasons behind choosing the particular database you are currently using? Were some of the selection criteria things like platform support, political issues, speed, specific features or capabilities? These are all reasons to pick a particular technology like a database, but if you are like most companies, you have more then one database for various reasons.

For the past 20 years Borland® InterBase® has been developed to handle all of the above points, and in those years, it has amassed an impressive customer list. Small companies to enterprise size organizations employ InterBase as a mission-critical product that helps run their businesses. Some examples of organizations that rely on InterBase include:

- DataFlow – embedded / resale accounting system
- Ultrasoft – packaged/embedded resale Customer Relationship Management (CRM) product
- YellowTuna Networks – database for enterprise transactions

When you look at the platforms and frameworks that InterBase supports out of the box, it is really impressive. From full support for the Windows® platform, Solaris™, and Linux® hardware platforms, to the support of Win32®, .NET, ODBC, Java and direct connection through C++ interfaces, InterBase is highly versatile. This highly performing database has been used in everything from large stock trading systems to being embedded in military hardware and deployed into remote locations. Its versatility is highlighted by the truly awesome list of capabilities it offers:

- Silent installer: Ability to install the product in an embedded fashion.
- Small footprint: Because it uses very little memory and disk space, InterBase can be run on inexpensive computers and is very portable.
- Zero administration: Self tuning features mean that no database administrator (DBA) or IT support is required.
- Automatic crash recovery: If the host computer crashes or suffers a power outage, InterBase will automatically start up again when the computer does and recover almost instantly, with no loss of data or data corruption.
- Easy to install: InterBase is easy to install. This is important when applications must be installed in situations where there is no DBA or IT support. InterBase can also be installed "invisibly" so that the application user doesn't even know that InterBase is there.
- High performance: InterBase is one of the fastest native databases available. It is also SQL92 compliant, which means that developers can build the most complex applications using InterBase without fear of running into barriers.
- Portable: InterBase is certified to run without modification on Windows, Linux and Solaris. This means that developers can deploy to a wide variety of platforms without having to rewrite code.
- Economical: InterBase is easy to develop with, does not require an expensive DBA or IT support, runs on inexpensive systems, and has a low purchase cost. As a result, the total cost of ownership for InterBase is very low, and users can realize a quick return on their investment.

A Borland White Paper

RE-INTRODUCTION TO BORLAND® INTERBASE®

COMPILED BY: MICHAEL J. ROZLOG
CHIEF TECHNICAL ARCHITECT

October 18, 2005

Borland®

Contents

Executive Summary:

Do you have a database? If you are like most of the companies in the world, the answer to that question would be yes. Can you remember what the reasons behind choosing the particular database you are currently using? Were some of the selection criteria things like platform support, political issues, speed, specific features or capabilities? These are all reasons to pick a particular technology like a database, but if you are like most companies, you have more then one database for various reasons.

For the past 20 years Borland® InterBase® has been developed to handle all of the above points, and in those years, it has amassed an impressive customer list. Small companies to enterprise size organizations employ InterBase as a mission-critical product that helps run their businesses. Some examples of organizations that rely on InterBase include:

- DataFlow – embedded / resale accounting system
- Ultrasoft – packaged/embedded resale Customer Relationship Management (CRM) product
- YellowTuna Networks – database for enterprise transactions

When you look at the platforms and frameworks that InterBase supports out of the box, it is really impressive. From full support for the Windows® platform, Solaris™, and Linux® hardware platforms, to the support of Win32®, .NET, ODBC, Java and direct connection through C++ interfaces, InterBase is highly versatile. This highly performing database has been used in everything from large stock trading systems to being embedded in military hardware and deployed into remote locations. Its versatility is highlighted by the truly awesome list of capabilities it offers:

- Silent installer: Ability to install the product in an embedded fashion.
- Small footprint: Because it uses very little memory and disk space, InterBase can be run on inexpensive computers and is very portable.
- Zero administration: Self tuning features mean that no database administrator (DBA) or IT support is required.
- Automatic crash recovery: If the host computer crashes or suffers a power outage, InterBase will automatically start up again when the computer does and recover almost instantly, with no loss of data or data corruption.
- Easy to install: InterBase is easy to install. This is important when applications must be installed in situations where there is no DBA or IT support. InterBase can also be installed "invisibly" so that the application user doesn't even know that InterBase is there.
- High performance: InterBase is one of the fastest native databases available. It is also SQL92 compliant, which means that developers can build the most complex applications using InterBase without fear of running into barriers.
- Portable: InterBase is certified to run without modification on Windows, Linux and Solaris. This means that developers can deploy to a wide variety of platforms without having to rewrite code.
- Economical: InterBase is easy to develop with, does not require an expensive DBA or IT support, runs on inexpensive systems, and has a low purchase cost. As a result, the total cost of ownership for InterBase is very low, and users can realize a quick return on their investment.

We have established that InterBase is a high performance, feature-rich, technically capable database. What is stopping you from using it? The only barrier left must be cost. There are open source databases and free databases from commercial vendors, but are they as good or have the same capabilities? What about the intangibles like support, integration into multiple technologies, and finally, the largest factor - vendor lock-in to the technology?

Many of the open source databases, while they look nice on the surface, are based on a support model, which means the database is not really free. Others give away the database with the hope that its compatibility with the for-pay version will be the logical step for the customer to make when they hit the ceiling with the free technology.

Finally, there are hidden costs. Do you need a DBA to run, tune, and maintain the other databases? Most likely the answer is yes. InterBase is a self-tuning, no DBA required database, that provides cost savings every year. However, don't let this unique feature scare you that InterBase is not customizable – the database is fully tunable to your desires, able to run with minimal intervention.

InterBase has one of the lowest cost of ownership models in the database market available today. The average cost for an Enterprise level unlimited user connection server either on Windows, Solaris, or Linux with 2-CPU(s) is less then $5,500. Let's look at the database cost as it pertains to the market today:

Open Source Competitors:

Average support prices range from:	$300.00 – $4,000.00 per server, every year.

Lite or Free Commercial Move to "Enterprise" Scale:

Average price range from	$8,000.00 - $20,000.00 per year + maintenance
Average price per added user:	$200.00 and up depending on volume
Additional processor cost:	$3,000.00 and up depending on OS

Commercial Data Solutions:

Average price range from:	$8,000.00 - $20,000.00 per year + maintenance
Average price per added user:	$200.00 and up depending on volume
Additional processor cost:	$3,000.00 and up depending on the OS

InterBase "Enterprise" Solution:

Average price range from:	$60- $200 for single user; 20% more for support
Average price per added user:	<$120 and $4,000 maximum
Additional processor cost:	$1,000-$1,500

Knowing this information will hopefully interest you enough to review the rest of this paper for technical details. Then, once it's clear that this may be a better and more affordable choice than the products currently being used, try using the product.

Why InterBase?

Borland InterBase is a high-performance, cross-platform embeddable database that is relied upon by millions of users around the world. Combining easy installation, automatic crash recovery, and minimal maintenance, the InterBase database is extremely well suited for embedding within distributed applications.

Multiprocessor support and a sophisticated architecture make it a great choice for high-powered, business critical applications that have many connected users. Its power, ease of use, support for the Windows, Linux, and Solaris platforms—as well as for development environments such as Borland Delphi®, Kylix®, and others—make the InterBase database a favorite with developers and a low cost option for business.

- InterBase has the small footprint, zero maintenance, high performance and low cost that is the essential characteristics of a great embedded database.
- InterBase support for SMP and its connection monitoring facilities improve application performance and accelerates the application development and deployment.
- InterBase is the best-of-breed native database component of Borland's application lifecycle management strategy. When it is time to deploy, Borland's InterBase is a choice that makes technical and business sense.

What can I do with InterBase?

The more appropriate question is, what can't be accomplished with InterBase? InterBase brings exceptional capabilities for single-tier database applications that are deployed to a single machine but have all the advanced enterprise database server features required for mission-critical endeavors. Plus, it has key capabilities that make it perfect for embedded solutions. Now, InterBase is packaged in two versions: InterBase Desktop and InterBase Server. However many configurations are possible, the most common ways people use the product are explained below:

Local (InterBase Desktop):

The Desktop edition is normally used for "stand-alone" applications. A developer may create this stand alone application and then wish to deploy it to multiple users. The developer would first obtain the Desktop Edition application by purchasing the Desktop Edition CD. Additional "user packs" would then be purchased for deployment to the appropriate number of users.

Note: Choose the Desktop Edition for applications that will not be part of a networked environment.

Server (InterBase Server):

The Server Edition is normally used for applications in which a number of users will connect simultaneously. Customers who wish to purchase the InterBase Server and one or more simultaneous user licenses, must purchase a Server License and licenses for the appropriate number of simultaneous users (or the unlimited user license).

Embedded:

InterBase combines simple installation, automatic crash recovery, and minimal maintenance, which makes it well suited for embedding in remotely deployed. It is also important for ISVs, OEMs, and VARs, who build "packaged" applications that are sold to end-users in the small to medium enterprises, to have a database that boasts a small footprint and have the sophisticated features to support high-powered, business-critical applications. This allows InterBase to be embedded in many types of applications, particularly popular in the fields of transportation, finance, medical and retail operations and gives the IT managers a way they can save significantly on the purchase price and on-going maintenance costs.

NOTE: A VAR agreement must be signed in order to re-sell the InterBase product as an embedded product.

InterBase Enterprise Limits:

Maximum number of clients connected to one server	There is no single number for the maximum number of clients the InterBase server can serve—it depends on a combination of factors including capability of the operating system, limitations of the hardware, and the demands that each client puts on the server. Applications that engage in high levels of contention or that perform complex or high-volume operations could cause the practical number of clients to be fewer. In applications that don't generate much contention, InterBase can support a large number of users, where "large" is not well-defined.
Maximum database size	No limit is imposed by InterBase. Maximum size is defined by the operating system.
Maximum number of files per database	By design, 216 (65,536) because the files are enumerated with an unsigned 16-bit integer. Shadow files count toward this limit. This is a design parameter of InterBase, but most operating systems have a much lower limit on the number of files that a single process can have open simultaneously. In some cases, the OS provides a means to raise this limit. Refer to your OS documentation for the default open files limit, and the means to raise this limit.
Maximum number of cache pages per database	65,536 for the sake of performance. A more practical upper limit would be 10,000. Total size of cache pages should never exceed 50% of memory.
Maximum number of databases open in one transaction	No restriction. The parameters in a transaction parameter buffer comprise a linked list, so there is no limit except that imposed by system resources.

Maximum number of tables per database	32,640
Maximum versions per table	255. Then, no more metadata changes until the database has been backed up and restored.
Maximum row size	64KB. Each Blob and array contributes eight bytes to this limit in the form of their Blob handle. Systems tables (tables maintained by the InterBase engine for system data) have a row size limit of 128KB.
Maximum number of rows and columns per table	By design, 232 rows because rows are enumerated with a 32-bit unsigned integer per table. Number of columns in a row depends on datatypes used. One row can be 64K. For example, you can define 16,384 columns of type INTEGER (four bytes each) in one table.
Maximum number of indexes per table	By design, 216 (65,536) because indexes per table are enumerated with a 16-bit unsigned integer.
Maximum number of indexes per database	By design, 232 because you can create 216 tables per database, and each table can have 216 indexes.
Maximum index key size	Multi-column keys. Subtract four bytes for each additional column. Example: a single-column CHAR key can be up to 256 – 4 = 252 bytes; a three-column key must add up to 200 – 12 = 188 bytes. Note that multi-byte character sets must fit within the key by counting bytes, not by counting characters. For example, a single-column key using 3-byte UNICODE_FSS characters can have a maximum of (256 – 4) / 3 = 84 characters. It is good practice to keep index keys as small as possible. This limits the depth of indexes and increases their efficiency.

Maximum number of events per stored procedure	No restriction by design, but there is a practical limit, given that there is a limit on the length of code in a stored procedure or trigger (see below).
Maximum stored procedure or trigger code size	48KB of BLR, the bytecode language compiled from stored procedure or trigger language
Maximum Blob size	The size of the largest single Blob depends on the database page size: 1KB page size: largest Blob is 64MB 2KB page size: largest Blob is 512MB 4KB page size: largest Blob is 4GB 8KB page size: largest Blob is 32GB A Blob is a stream of many segments. The maximum Blob segment size is 64KB.
Maximum tables in a JOIN	No restriction by design, but the task of joining tables is exponential in relation to number of tables in the join. The largest practical number of tables in a JOIN is about 16, but experiment with your application and a realistic volume of data to find the most complex join that has acceptable performance.
Maximum levels of nested queries	There is no restriction by design. The practical limit depends on the type of queries you nest. Experiment with your application and a realistic volume of data to find the deepest nested query that has acceptable performance.
Maximum number of columns per one composite index	16
Levels of nesting per stored procedure or trigger	■ 750 on Windows platforms ■ 1000 for UNIX platforms

Maximum size of key in SORT clause	32 KB
Maximum size of external table file	4 GB on Windows NT/2000/XP; 2 GB on Solaris, Linux, and Windows 98/ME
Range of date values	January 1, 100 A.D. to February 29, 32768 A.D.

Special Features of InterBase for Delphi® and C++Builder® users:

InterBase features multiple instances on a single machine, offering easier deployment and a more sophisticated development environment. With added automatic re-routing of databases, server-side database alias and embedded database user authentication, deployment of your applications is even easier. Along with functionality that allows C# Builder® and Microsoft® Visual Studio® developers to use InterBase for their .NET development. InterBase provides a high quality, low cost alternative to many of the competitors. Many Visual Studio developers also resent the "lock in" strategy of proprietary syntax and restrictive licensing associated with the Microsoft product sets.

Developers who are building applications using CBuilderX, Delphi® 2005, C++, Kylix, and the new Borland Developer Studio are natural users of InterBase. The product is designed for easy and productive use with those products and is distributed in development-trial version with each of them.

Special Features for InterBase for JBuilder® and Java® users:

Features to connect InterBase with your JBuilder® application or applet have been minimized by using the Database component to establish the connection, a DataSet component (such as a TableDataSet or QueryDataSet component) to provide the data, and a data-aware control (such as a GridControl) to display the results.

Besides, having an all-Java® JDBC driver (Type 4), InterClient enables platform-independent, client/server development for the Internet and corporate intranets. The advantage of an all-Java driver versus a native-code driver is that you can deploy InterClient-based applets without having to manually load platform-specific JDBC drivers on each client system. Web servers automatically download the InterClient classes along with the applets. Therefore, there is no need to manage local native database libraries, which simplifies administration and maintenance of customer applications. As part of a Java applet, InterClient can be dynamically updated, further reducing the cost of application deployment and maintenance.

IBExpress (IBX) or InterBase Express:

InterBase 7.5 is a significantly improved version of Borland's excellent embedded database. Since Delphi, C++, Kylix, and Borland Developer Studio each ships with built-in IBExpress components to allow flawless, native connection to InterBase, it makes sense to use this new, improved version when it comes time to deploy your application.

Special features for Embedding InterBase:

Embedded database user authentication is a security enhancement new to InterBase 7.5, which now can manage multiple databases for unrelated applications. This feature allows database-specific user account management. Because most applications do not share databases, this feature also leads to customizable application-specific user management.

TIBInstall and its ancestor, TIBSetup provide properties to allow you to build an InterBase install component into your application. TIBInstall allows you to set the installation source and destination, displays your own installation messages, and set the individual InterBase components to be installed.

How do I get started?

Below are some additional references and where you can download the trial product from Borland Software Corporation. In the Appendix of this document, there are two sections, one for installing and configuring the product and one with an example using Borland Developer Studio 2006 (BETA).

References:

Here is where the background information that was used to help compile this re-introduction to InterBase paper is located.

Case Studies (on the Borland Web site):

- Dataflow.pdf
- Ultrasoft.pdf
- YellowTuna Networks.pdf

Getting InterBase / Demo / Real World Examples:

All of the below have applications that run against the InterBase databases on the backend. Each one supports the Borland Software Corporation Community every single day.

- Code Central page
- Quality Central page
- Quality Control statistics on the Borland Developer Network (BDN) site

Getting InterBase Trial:

- Go to the Downloads page on the Borland web site.

Other:

- Internal Borland Software Corporation documentation
- Borland Software Corporation InterBase Developer Guide
- Borland Software Corporation InterBase OpGuide
- Borland Software Corporation Developer's Network

Summary:

Borland challenges each and every one of the readers of this paper to spend an afternoon reviewing 20 years of development of a product like InterBase – you will be impressed. The amount of features and true functionality found in this database is staggering. Keep in mind, that InterBase Server is designed to run on Windows, Solaris, and Linux operating systems. The product comes on a single CD, takes a whopping 20 megabytes of hard drive space when installed and equally impressive, and has a 32 megabyte minimum memory requirement. Those types of resources are the kinds that were prevalent in the mid 90's. However, just because the minimum memory footprint is around 32 megabytes, this is not all that InterBase will use. If additional memory is available, temporary tables, caches, and garbage collections activities all increase in speed. Very Large Data Base (VLDB) is a feature that allows for 64 bit I/O and the ability for the databases in a single file to grow to terabytes in length.

So the database is small and powerful. What about real world situations? Take the example of power outages and the time and effort most databases need to recover from such an event. Even in today's market, where UPC and power backup are everywhere, power loss or system failure is still a valid concern in these interesting times. In most companies, the ability to recover from one of these types of events is truly mission critical, plus the need to have experienced staff on hand or available to help recover data in a timely fashion is at the top of the customer "needs" list. With InterBase, the only thing needed to resolve and recover from catastrophic events such as these are power and a running system – no DBA required.

Since InterBase has been a stable of Borland for so long, it is no surprise that it supports multiple components and wizards for tools like Borland Windows platform, Borland Developer Studio 2006, which supports Delphi for Win32® and Delphi for .NET, .NET (C#), C++ and the BDP components for data access. It also comes with the latest support for JDBC by supplying a Type 4 all-Java driver that may be used by any Java application and can be used to take full advantage of InterBase's power on the Java platform. So beyond being small, reliant, fast, and easy to use, what additional advanced features does the database support? Thing like dependency tracking of User Defined Functions (UDFs), XML data generation for enhanced support for XML based applications, embedded SQL, SQL Rows support, and programmable services to help administer and monitor the database when it is deployed.

By now, hopefully you are convinced that InterBase can be used anywhere from embedded, departmental, to full-out enterprise databases. With its feature-rich history and feature additions with every new release, this is a cutting-edge product with years of maturity and optimization under its belt. With a price point to fit any company's budget, you cannot go wrong with InterBase.

The next time there is a need for a new database, replacement of an existing implementation, or the need for a specialized database, give InterBase a solid look. The features included in InterBase are too numerous to list, but some of the key characteristics include:

- **Small footprint:** Because it uses memory and disk space very efficiently, InterBase can be run on inexpensive computers and is very portable.
- **Zero administration:** Self tuning features mean that no database administrator or IT support is required.
- **Automatic crash recovery:** If the host computer crashes or suffers a power outage, InterBase will automatically start up again when the computer does and recover almost instantly, with no loss of data or data corruption.
- **Easy to install:** InterBase is easy to install. This is important when applications must be installed in situations where there is no DBA or IT support. InterBase can also be installed "invisibly" so that the application user doesn't even know that InterBase is there.

- **High performance:** InterBase is one of the fastest native databases available. It is also SQL92 compliant which means that developers can build the most complex applications using InterBase without fear of running into barriers.
- **Portable:** InterBase is certified to run without modification on Windows, Linux and Solaris. This means that developers can deploy to a wide variety of platforms without having to rewrite code.
- **Economical:** Because InterBase is easy to develop with, requires no expensive DBA or IT support, runs on inexpensive systems, and has a low purchase cost, the total cost of ownership for InterBase is extremely low and the return on investment is very fast.

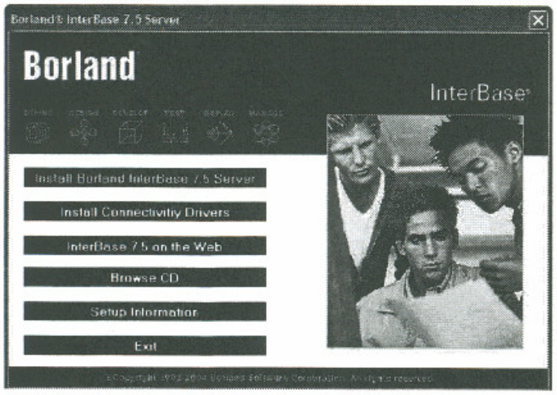
This proven database has been around for more than 20 years and keeps improving with age, running a large range of applications from mission-critical systems to remote devices across the world. Take a look for yourself. And please call us after you do.

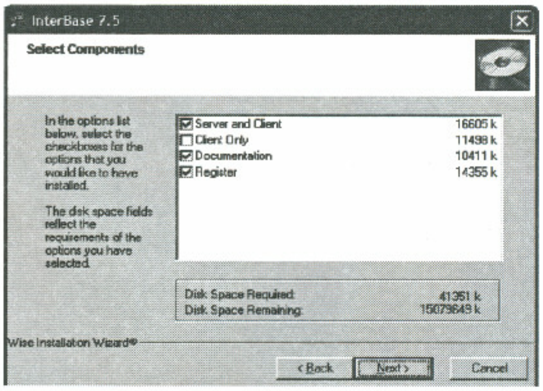
Installing InterBase:

One of the key features of InterBase is the ease of installation and setup. If you need to get a trial of InterBase, please go to the resources section of this paper for the details on doing that. Installing InterBase on Windows and setting it up can be accomplished in a few simple steps outlined below:

Windows:

Installing the product on Windows is simply enough, just put the CD into the CD-tray and off it goes. The following are the progressions of the installer:


Borland® InterBase 7.5 Server
Borland
InterBase
Install Borland InterBase 7.5 Server
Install Connectivity Drivers
InterBase 7.5 on the Web
Browse CD
Setup Information
Exit
© Copyright 1993-2004 Borland Software Corporation. All rights reserved.



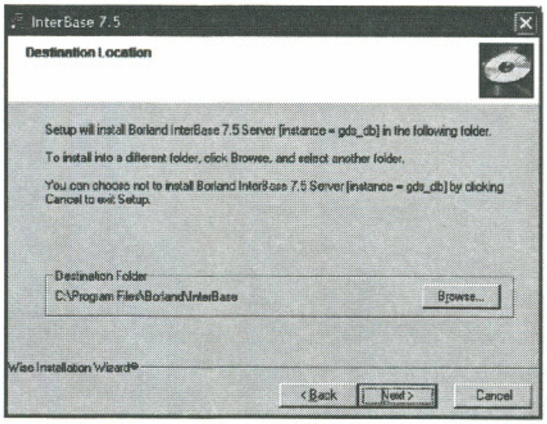
In the options list below, select the checkboxes for the options that you would like to have installed.

The disk space fields reflect the requirements of the options you have selected.

☑ Server and Client	16605 k
□ Client Only	11498 k
☑ Documentation	10411 k
☑ Register	14355 k

Disk Space Required:	41351 k
Disk Space Remaining:	15079649 k

Since InterBase has the ability to have multiple copies loaded on the same machine, the installer will ask if that is a configuration that is of interest. For this example, click NO and continue. It will then show what is going to be loaded, like the second screen above.

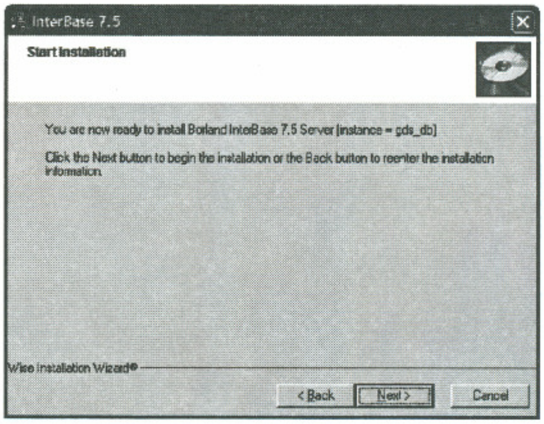


Setup will install Borland InterBase 7.5 Server [instance = gds_db] in the following folder.

To install into a different folder, click Browse, and select another folder.

You can choose not to install Borland InterBase 7.5 Server [instance = gds_db] by clicking Cancel to exit Setup.

Destination Folder
C:\Program Files\Borland\InterBase



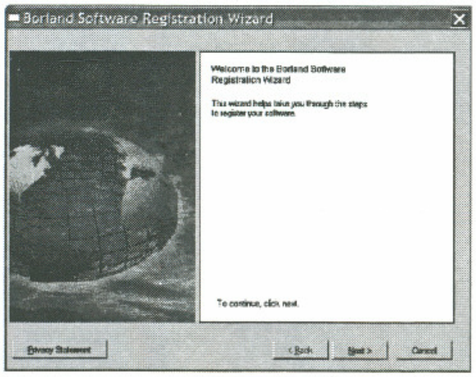
You are now ready to install Borland InterBase 7.5 Server [instance = gds_db]

Click the Next button to begin the installation or the Back button to reenter the installation information.

The installer will verify the location and the last Next> buttonwill start the install. Once the install is complete, it will start the Registration process.

Registration:

The registration process is the one that is generally followed by Borland products.

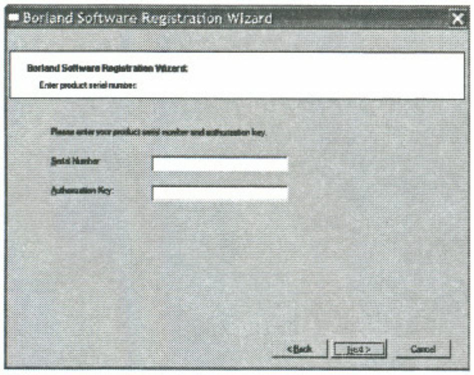


Welcome to the Borland Software Registration Wizard

This wizard helps take you through the steps to register your software.

To continue, click next.

1. Once the Next button is pressed, the Serial number and Key screen will be displayed as below:



Borland Software Registration Wizard:

Enter product serial number:

Please enter your product serial number and authorization key.

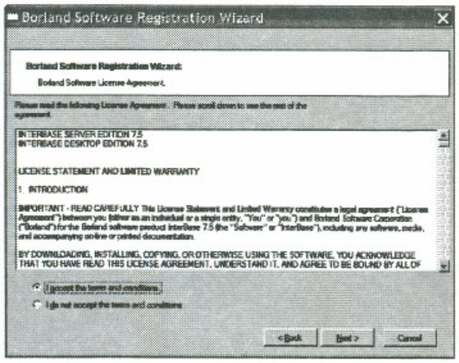
Serial Number

Authorization Key:

2. It is highly recommended that you turn on CAPS LOCK to input the key. Also, make sure you add the dashes (-) in the proper location so the key is accepted. If a mistake is found in the key, a red warning under the key will be displayed.

3. After adding the serial and key, press the Next button to continue.

4. The license agreement will be displayed again.



Borland Software Registration Wizard:

Borland Software License Agreement.

Please read the following License Agreement. Please scroll down to see the rest of the agreement.

INTERBASE SERVER EDITION 7.5
INTERBASE DESKTOP EDITION 7.5

LICENSE STATEMENT AND LIMITED WARRANTY

1. INTRODUCTION

IMPORTANT - READ CAREFULLY This License Statement and Limited Warranty constitutes a legal agreement ("License Agreement") between you (either as an individual or a single entity, "You" or "you") and Borland Software Corporation ("Borland") for the Borland software product InterBase 7.5 (the "Software" or "InterBase"), including any software, media, and accompanying on-line or printed documentation.

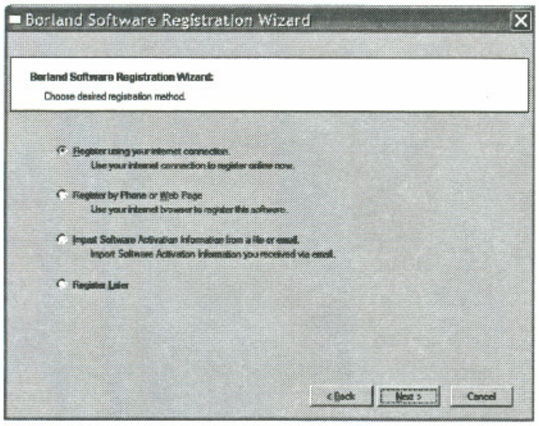
BY DOWNLOADING, INSTALLING, COPYING, OR OTHERWISE USING THE SOFTWARE, YOU ACKNOWLEDGE THAT YOU HAVE READ THIS LICENSE AGREEMENT, UNDERSTAND IT, AND AGREE TO BE BOUND BY ALL OF

● I accept the terms and conditions.

○ I do not accept the terms and conditions

NOTE: It is always important to read and understand the license presented. Please read the license before accepting and continuing.

5. A series of screens will be displayed for how you want to register. Below is the progression using the Internet and a bdn.borland.com account.



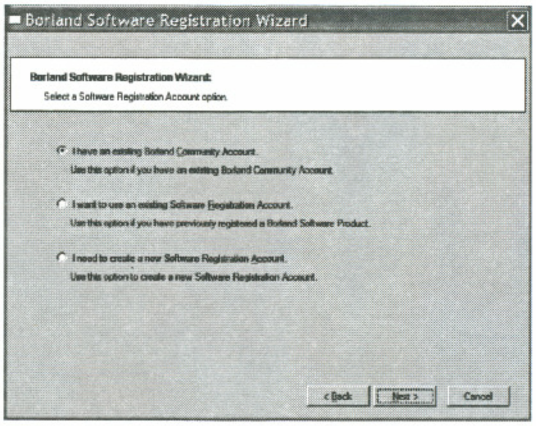
Borland Software Registration Wizard:
Choose desired registration method.

● Register using your internet connection.
Use your internet connection to register online now.

○ Register by Phone or Web Page
Use your internet browser to register this software.

○ Import Software Activation Information from a file or email.
Import Software Activation Information you received via email.

○ Register Later

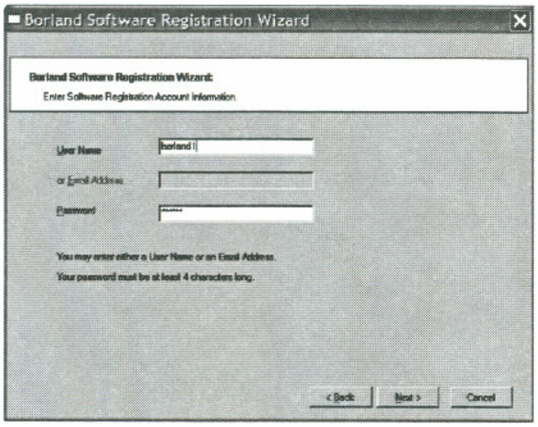


Borland Software Registration Wizard:
Select a Software Registration Account option.

● I have an existing Borland Community Account.
Use this option if you have an existing Borland Community Account.

○ I want to use an existing Software Registration Account.
Use this option if you have previously registered a Borland Software Product.

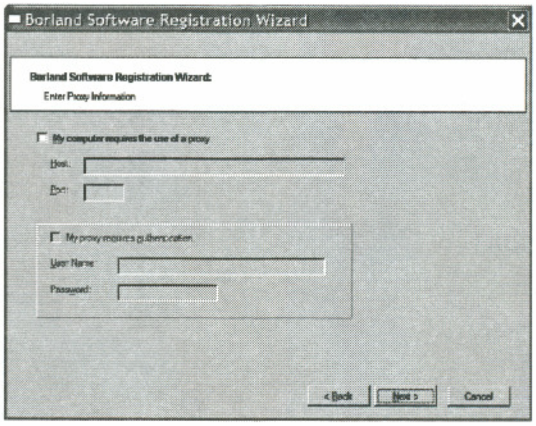
○ I need to create a new Software Registration Account.
Use this option to create a new Software Registration Account.



Borland Software Registration Wizard:
Enter Software Registration Account information.

User Name: borland I
or Email Address:
Password: ******

You may enter either a User Name or an Email Address.
Your password must be at least 4 characters long.



Borland Software Registration Wizard:
Enter Proxy Information

□ My computer requires the use of a proxy

Host:
Port:

□ My proxy requires authentication

User Name:
Password:

6. A confirmation dialog will be displayed and when the product is registered it will return with the Product Key. It is always good to copy down the Serial, Key, and Product Key, and keep that information in a safe place.

7. Once the product is registered, you can then go to the InterBase Setup for Windows section of the paper.

InterBase Setup for Windows:

Setup is just as easy as the install. Follow the steps below to get it setup.

1. Start|All Programs|Borland InterBase 7.5 Server [instance = gds_db]|InterBase Server Manager [instance = gds_db]

2. This start the server manager.

The above is the standard installs properties. At this point, you can not click the Server Properties or Guardian Properties. This dialog is a little confusing:

Startup Mode: Automatic is the used in conjunction with Run as a Server in the Status.
Manual is if you want to start it every single time by coming to this dialog.

Root Directory: This should be the install directory of InterBase.

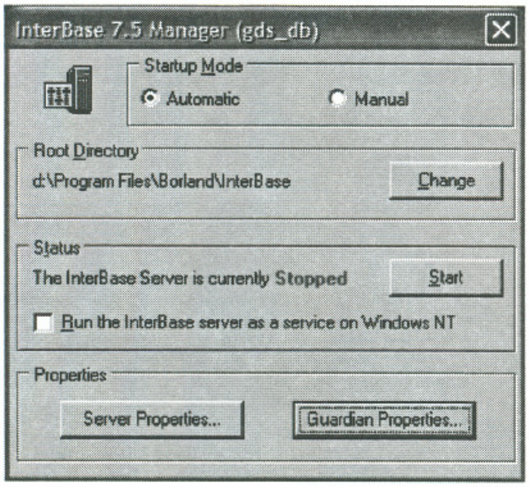
Status: If the InterBase Server is currently Stopped, then before you do anything, you want to set the other properties again before you start it.

Run as Service: Check this if you want it to run as a windows service. This should be checked before starting.

Properties: Server Properties…; this will load once the server is started.

Guardian Properties…; this will also load once the server is started.

3. At this point, click the start button.



InterBase 7.5 Manager (gds_db)

Startup Mode

● Automatic ○ Manual

Root Directory

d:\Program Files\Borland\InterBase

Status

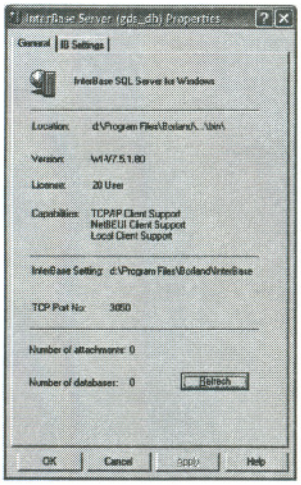
The InterBase Server is currently Stopped

□ Run the InterBase server as a service on Windows NT

Properties

Notice the server is now running. Also notice that the options for a Service are disabled! This must be set before you start the server.

4. Now, looking at the Server Properties:



InterBase SQL Server for Windows

Location: d:\Program Files\Borland\...\bin\

Version: WI-V7.5.1.80

License: 20 User

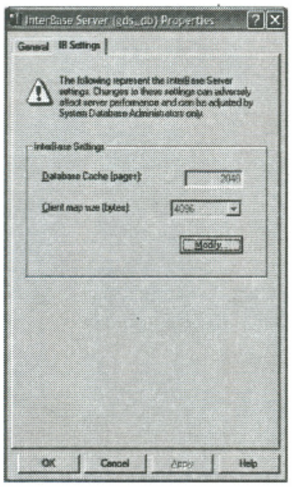
Capabilities: TCP/IP Client Support
NetBEUI Client Support
Local Client Support

InterBase Setting: d:\Program Files\Borland\InterBase

TCP Port No: 3050

Number of attachments: 0

Number of databases: 0

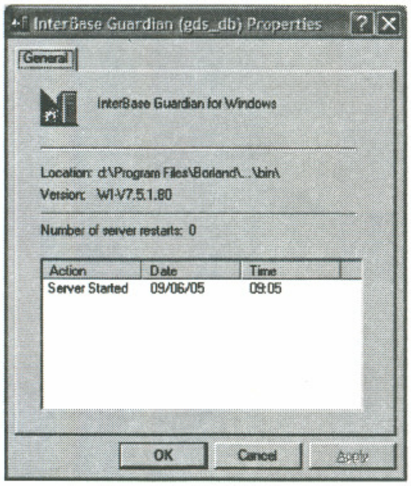


The following represent the InterBase Server settings. Changes to these settings can adversely affect server performance and can be adjusted by System Database Administrators only.

Database Cache (pages): 2048

Client map size (bytes): 4096

5. This will display the above dialogs. It basically tells you how the InterBase instance is actually configured. You can modify basically only the IB Settings, which are the cache issues and Client communication in bytes.



InterBase Guardian for Windows

Location: d:\Program Files\Borland\...\bin\
Version: WI-V7.5.1.80

Number of server restarts: 0

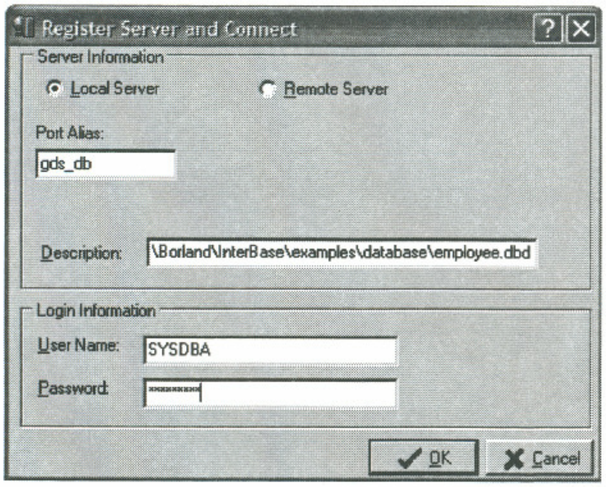
Action	Date	Time
Server Started	09/06/05	09:05

6. The Guardian dialog basically is there to show when the server was started and where its located.

Local Server:

1. Now that the server is started, we can go into the IbConsole and get things working.

2. In the Console, go to Server|Register.



● Local Server ○ Remote Server

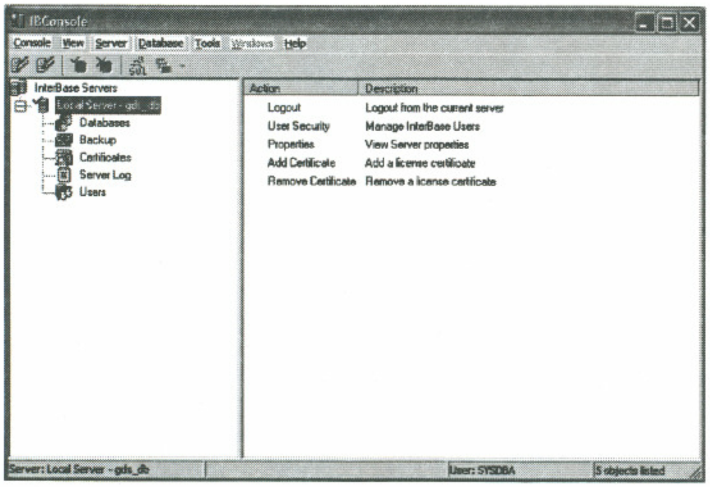
Port Alias: gds_db

Description: \Borland\InterBase\examples\database\employee.dbd

User Name: SYSDBA

Password: *********

3. Once the OK button is pressed, the Local Server should be listed as below.



Action	Description
Logout	Logout from the current server
User Security	Manage InterBase Users
Properties	View Server properties
Add Certificate	Add a license certificate
Remove Certificate	Remove a license certificate

Server: Local Server - gds_db | User: SYSDBA | 5 objects listed

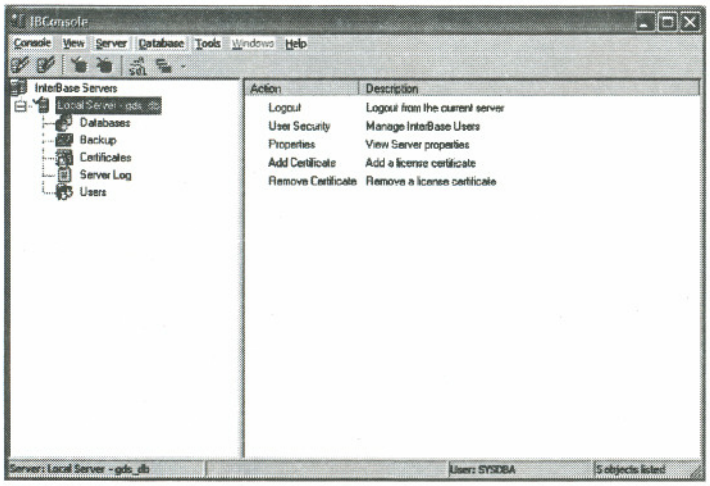
4. In the above IBConsole, click the plus sign to see the Local Server Properties.

5. Right mouse click on the Database icon, and select Register.

6. Click the … button and select the employee.gdb that is sent with InterBase, and click OK on the button. This will also setup the employee.gdb Alias. Login Information is:

User Name: SYSDBA – this is the standard root user
Password: masterkey – this is the standard root users password

7. Press the OK button, and the following IBConsole should appear:



Action	Description
Logout	Logout from the current server
User Security	Manage InterBase Users
Properties	View Server properties
Add Certificate	Add a license certificate
Remove Certificate	Remove a license certificate

Server: Local Server - gds_db | User: SYSDBA | 5 objects listed

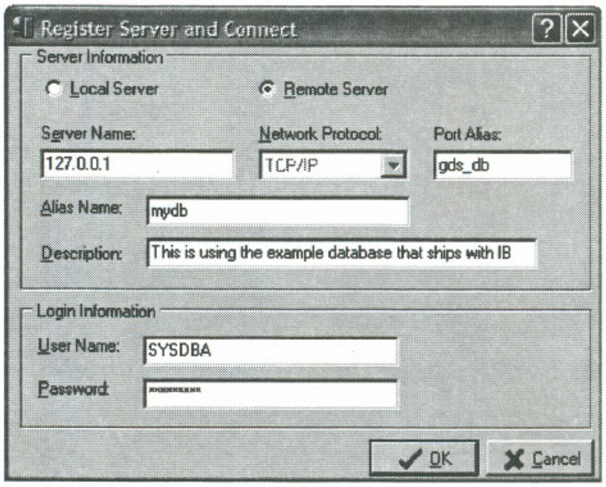
8. This is how you start a local (desktop) edition of InterBase.

Server

The Server is almost the same as the Local InterBase (Desktop) edition. The difference is when you want to add a Database. The following will walk you through those steps.

9. Making the assumptions that the InterBase database is started as followed in steps 1 – 8 of this document.

10. Doing the Server|Register from the IBConsole



○ Local Server ● Remote Server

Server Name: 127.0.0.1

Network Protocol: TCP/IP

Port Alias: gds_db

Alias Name: mydb

Description: This is using the example database that ships with IB

User Name: SYSDBA

Password: *********

11. The server name could be the name of the machine.

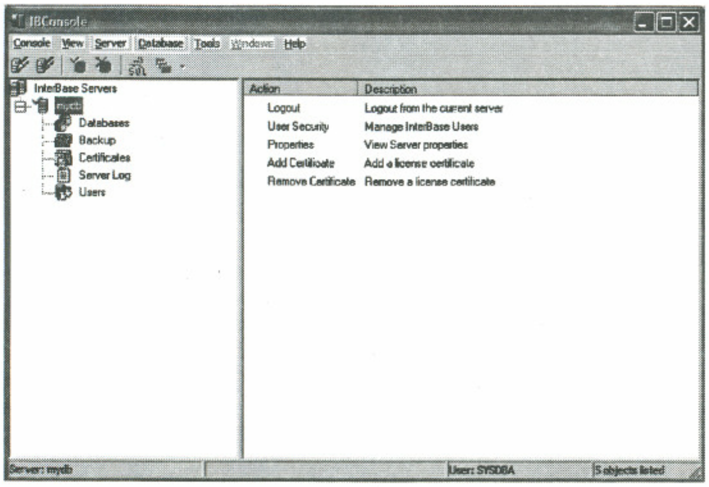
12. Fill in the Alias and Description.

13. Login Information is as follows:

User Name: SYSDBA – used as the root user

Password: masterkey – used as the root password for the root user

14. The IBConsole should look like the following. I have clicked the + inside the IBConsole explorer DB_Alias.

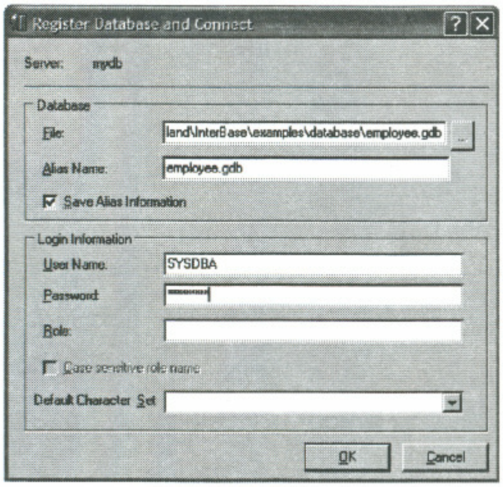


Action	Description
Logout	Logout from the current server
User Security	Manage InterBase Users
Properties	View Server properties
Add Certificate	Add a license certificate
Remove Certificate	Remove a license certificate

Server: mydb
User: SYSDBA
5 objects listed

15. Right-mouse click on the Database and select the Register menu item.

16. Now fill in the needed information like below:



Server: mydb

Database

File: land\InterBase\examples\database\employee.gdb

Alias Name: employee.gdb

☑ Save Alias Information

Login Information

User Name: SYSDBA

Password: ********

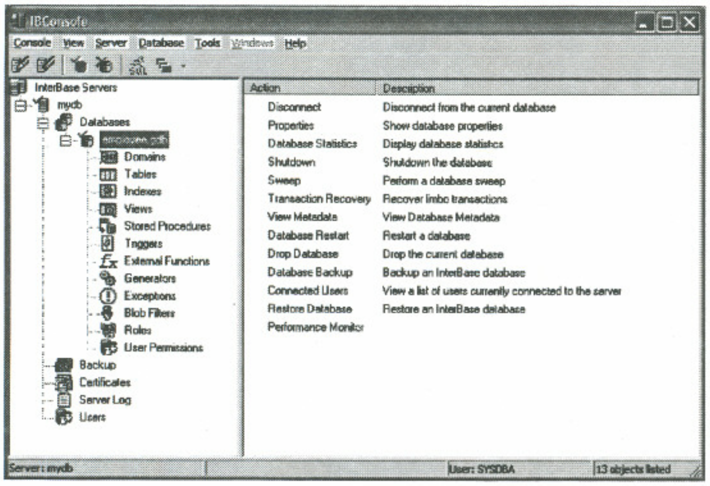
Role:

☐ Case sensitive role name

Default Character Set:

17. Remember to fill in the UserName and Password.

18. The IBConsole should appear as follows:

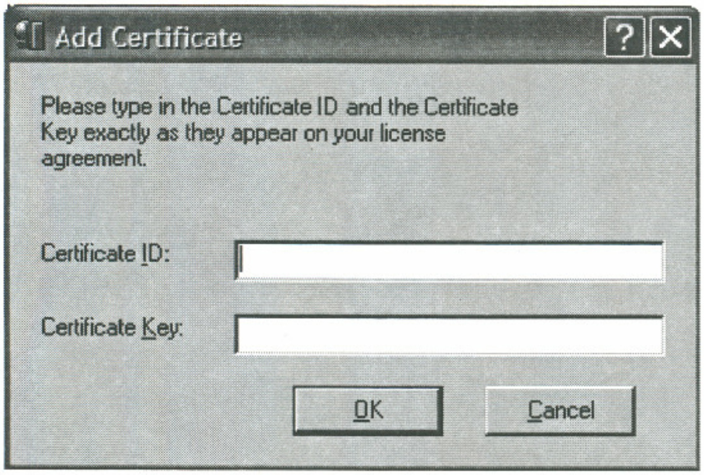


Action	Description
Disconnect	Disconnect from the current database
Properties	Show database properties
Database Statistics	Display database statistics
Shutdown	Shutdown the database
Sweep	Perform a database sweep
Transaction Recovery	Recover limbo transactions
View Metadata	View Database Metadata
Database Restart	Restart a database
Drop Database	Drop the current database
Database Backup	Backup an InterBase database
Connected Users	View a list of users currently connected to the server
Restore Database	Restore an InterBase database
Performance Monitor	

Server: mydb | User: SYSDBA | 13 objects listed

19. This is how you setup an InterBase Server.

20. If you have to add Users to the InterBase Server, this can be done by clicking the Server|Add Certificate.



Please type in the Certificate ID and the Certificate Key exactly as they appear on your license agreement.

Certificate ID:

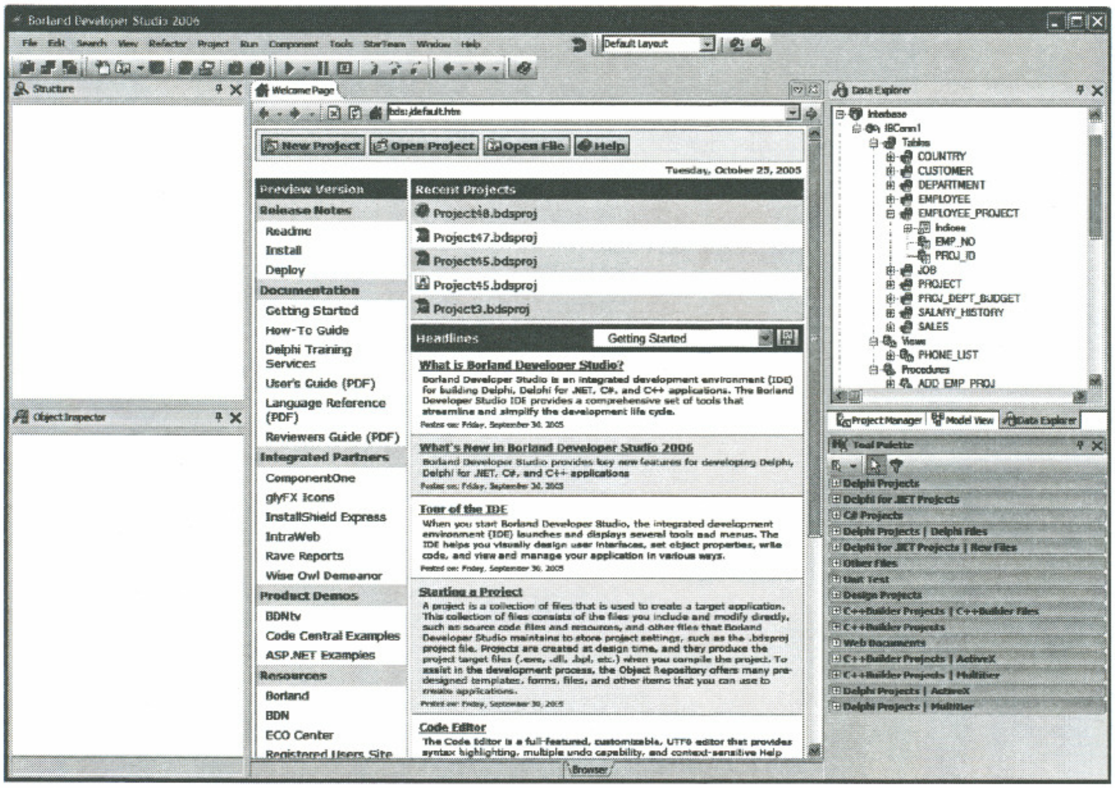
Certificate Key:

21. You add as many certificates as you have by repeating step 20 for the number of certificates you have to add.

Quick Examples using Borland Developer Tools:

Using InterBase 7.5 with Borland Developer Suite 2006:

This example is using the beta version of Delphi 2006. Things could change, but it has also been tested on Delphi 2005 and works.



What is Borland Developer Studio?

Borland Developer Studio is an integrated development environment (IDE) for building Delphi, Delphi for .NET, C#, and C++ applications. The Borland Developer Studio IDE provides a comprehensive set of tools that streamline and simplify the development life cycle.

Posted on: Friday, September 30, 2005

What's New in Borland Developer Studio 2006

Borland Developer Studio provides key new features for developing Delphi, Delphi for .NET, C#, and C++ applications

Posted on: Friday, September 30, 2005

Tour of the IDE

When you start Borland Developer Studio, the integrated development environment (IDE) launches and displays several tools and menus. The IDE helps you visually design user interfaces, set object properties, write code, and view and manage your application in various ways.

Posted on: Friday, September 30, 2005

Starting a Project

A project is a collection of files that is used to create a target application. This collection of files consists of the files you include and modify directly, such as source code files and resources, and other files that Borland Developer Studio maintains to store project settings, such as the .bdsproj project file. Projects are created at design time, and they produce the project target files (.exe, .dll, .bpl, etc.) when you compile the project. To assist in the development process, the Object Repository offers many pre-designed templates, forms, files, and other items that you can use to create applications.

Posted on: Friday, September 30, 2005

Code Editor

The Code Editor is a full-featured, customizable, UTF8 editor that provides syntax highlighting, multiple undo capability, and context-sensitive Help

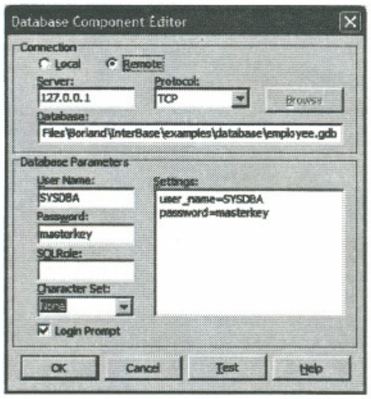
This is going to be a very simple and quick example of how to use the InterBase Express components (IBExpress) to verify InterBase is working correctly with Borland Developer Studio 2006 (BDS 2006). It makes the assumption that BDS 2006 and InterBase is loaded onto the machine to be used.

There are so many new features inside the Next version of the Borland Developer Studio, some of them include:

- C++ Personality, the next installment of C++Builder for the Windows platform.
- New memory Manager for a better performing product.
- New designer enhancements to make interface layout faster and more productive.
- Enhanced Together support for modeling.
- Enhanced Database, ECO (Enterprise Core Objects), ASP.NET development and many more.
- Enhanced Starteam® and CaliberRM™ integrations.

Create Delphi.NET project:

1. Click the File|New|VCL Forms Application – Delphi for .NET menu item.
2. Click the InterBase expander on the tools palette, and then drop a TIBDatabase component on the form.
3. Click and drop a TIBQuery on the form.
4. Click and drop a TIBTransaction on the form.
5. Right-mouse click on the TIBDatabase component and select the Database Editor… menu item. The following dialog will be displayed:



○ Local ● Remote

Server: 127.0.0.1

Protocol: TCP

Database: Files\Borland\InterBase\examples\database\employee.gdb

User Name: SYSDBA

Password: masterkey

SQLRole:

Character Set: None

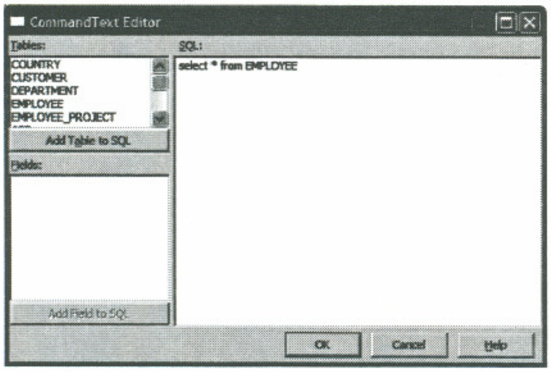
Settings:
user_name=SYSDBA
password=masterkey

☑ Login Prompt

Set the above properties as shown. Press the Test button to ensure connection to the database.

6. Click the OK button to continue.
7. Change the Connected property to True.
8. Click on the TIBQuery and set the following properties in the Properties editor.

Database: IBDatabase1
Transaction: IBTransaction1
SQL: select * from employee;
Active: True



Tables:
COUNTRY
CUSTOMER
DEPARTMENT
EMPLOYEE
EMPLOYEE_PROJECT

SQL:
select * from EMPLOYEE

9. Click the OK button to continue.

10. Click on the TIBTransaction component, and set the DefaultDatabase to IBDatabase1.

11. Now click the Data Access expander, select the TDataSource, and drop it on the form.

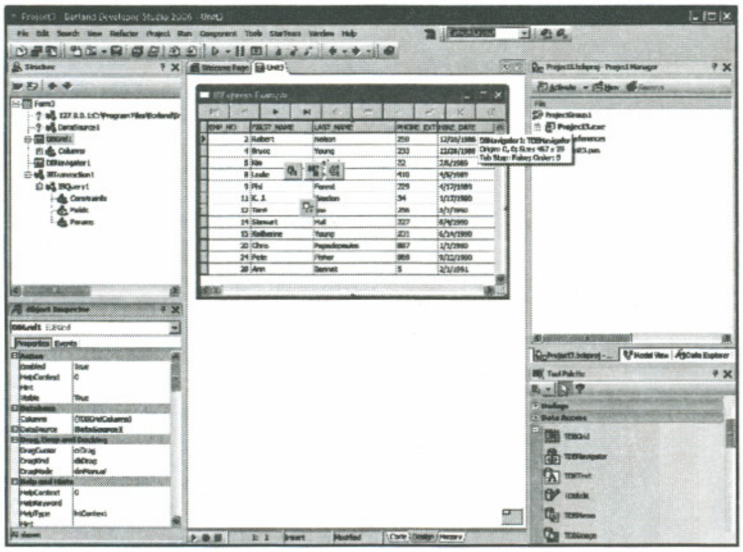
12. On the TDataSource, set the DataSet property to IBQuery.

13. Click the Data Controls, and drop a TDBNavigator at the top of the form. Set the Align property to alTop.

14. Next, drop a TDBGrid on the form, and set its Align property to alClient.

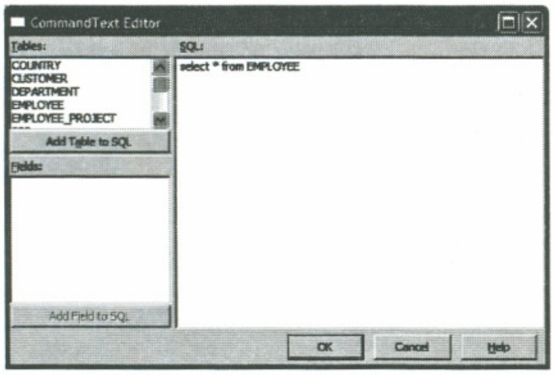
15. Hold the Shift-key and mouse-click on the TDBNavigator and TDBGrid, and set the DataSource property in the property editor to DataSource1.

16. The environment should look like the following:



EMP_NO	FIRST_NAME	LAST_NAME	PHONE_EXT	HIRE_DATE
2	Robert	Nelson	250	12/28/1988
4	Bruce	Young	233	12/28/1988
5	Kim		22	2/6/1989
8	Leslie		410	4/5/1989
9	Phil	Forest	229	4/17/1989
11	K. J.	Weston	34	1/17/1990
12	Terri	Lee	256	5/1/1990
14	Stewart	Hall	327	6/4/1990
15	Katherine	Young	231	6/14/1990
20	Chris	Papadopoulos	887	1/1/1990
24	Pete	Fisher	888	9/12/1990
28	Ann	Bennet	5	2/1/1991

17. The final thing to do is test the quick application. Hit the run button, and the following application should appear:



COUNTRY
CUSTOMER
DEPARTMENT
EMPLOYEE
EMPLOYEE_PROJECT

```
select * from EMPLOYEE
```

18. Example done.

Using InterBase 7.5 with JBuilder 2006 Enterprise:

This edition of JBuilder 2006 has some of the most interesting features to be introduced since JBuilder 4.0. The 2006 edition has many new things that make it an awesome upgrade or a stellar purchase, including:

- Peer to Peer Collaboration, allowing for pair programming, developer chat, and even multiple developers debugging sessions.
- A lot of new coding productivity tools have been added.
- Full support for JDK 5.0. The IDE is actually now hosted on that particular JDK.

Some people have problems getting InterBase configured to be used with JBuilder. The following example should remove any questions on how to configure JBuilder to work with InterBase.

Where's InterBase?

As many Java developers know, the classpath is everything in Java. If it is not configured properly, nothing is going to work very well. This is especially true with JBuilder, because it can access libraries from many different places. The three most common include:

- The JBuilder Environment - This is everything that can be executed from the JBuilder\bin directory.
- The Application Environment - This is inside the JBuilder environment.
- The running application - This is a special case, because classpaths can be loaded dynamically so programs that execute in this manner don't have concern for the two other options. This example is going to focus on the first two options only.

Making InterBase available to the JBuilder environment:

The most important files to remember when adding Java classes, .zip(s), and .jar(s) to the JBuilder environment include the **jbuilder.config** and the **jdk.config** files located inside the **{install drive}:\jbuilder2006\bin** directory. The first file of importance is the jbuilder.config. The following is a small excerpt:

```
# +---------------------------------------------------------------+
# |                    |
# | This is the configuration file for the JBuilder IDE |
# |                    |
# |  Do NOT make modifications without maintaining a  |
# |   backup copy. See config_readme.html for a  |
# |   description of the recognized directives.  |
# |                    |
# +---------------------------------------------------------------+
 # Add any configuration files located in the patch directory
 includedir ../patch

# Enable peer to peer debugging
addpath ../lib/jdi-extension.jar

# Enable HotSpot Serviceablity Agent
include sa.config
```

```
# Read the shared JDK definition
include jdk.config

...

# Add all JAR files located in the patch, lib and lib/ext directory
addjars ../patch
addjars ../lib
addjars ../lib/bes
addjars ../lib/servers
addjars ../lib/p2p
addjars ../lib/ext
```

As the comments in the config file point out, if you want more in-depth information on the .config files used by the JBuilder environment, check out the config_readme.html for the latest information and explanations about what is found in these files.

The goal is to add InterBase to the JBuilder environment, so the focus should be on the Add all JAR files section, which is in bold above. This is where a modification would be done to include **InterClient.jar**, which is the all Java type 4 database driver. This file will be located in a normal install in the lib directory under the main InterBase directory. To add the .jar file to the environment add the following line:

```
addpath C:\Program Files\Borland\InterBase\lib\interclient.jar
```

NOTE: JBuilder should be closed when making changes to the config files.

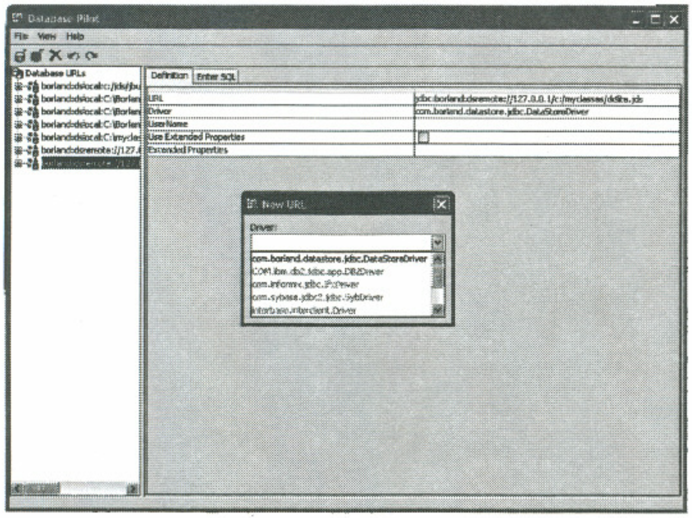
NOTE: Make sure the addpath is lower case.

Now if you were only going to use InterBase inside JBuilder the above fix might be OK, but there are other programs that might use it, like DatabasePilot. To make the interclient.jar file available to that tool, one could add the same line above to its config file. The other option is to make it available to all tools. Notice in the above excerpt that another set of lines were bolded.

```
# Read the shared JDK definition
include jdk.config
```

This uses the include statement, which will add all of the switches defined in the JDK.config file to whichever file calls it. So, if the change is going to be universal or the features should be available everywhere, then this is the file to update. Add the above Addpath line to the JDK.config, and everything should be set.

Without adding the line:


Database Pilot
File View Help
Database URLs
borland:dslocal:c:/jds/jbu
borland:dslocal:C:\Borlan
borland:dslocal:C:\Borlan
borland:dslocal:C:\Borlan
borland:dslocal:C:\myclas
borland:dsremote://127.0
Definition
Enter SQL
New URL

URL	jdbc:borland:dsremote://127.0.0.1/c:/myclasses/ddlite.jds
Driver	com.borland.datastore.jdbc.DataStoreDriver
UserName	
Use Extended Properties	□
Extended Properties	

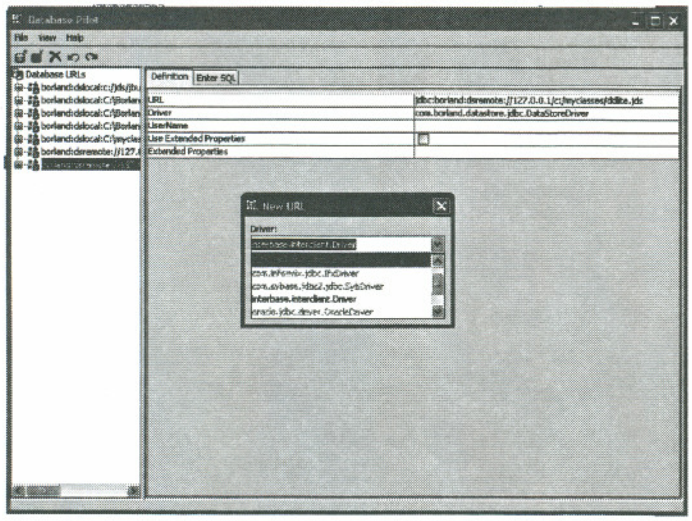
Driver:

com.borland.datastore.jdbc.DataStoreDriver
COM.ibm.db2.jdbc.app.DB2Driver
com.informix.jdbc.IfxDriver
com.sybase.jdbc2.jdbc.SybDriver
interbase.interclient.Driver

This is an example of where the environment does not know about the interclient.jar file. Once the interclient.jar file is added to the JBuilder.config or the **JDK.config**, the red will be removed as shown in the example below.

By adding the .jar file to the JDK.config:

Notice now that the **interBase.interclient.Driver** is now black and ready for use. This means the .jar file has successfully been added to the system.


Database Pilot
File View Help
Database URLs
borland:dslocal:c:/jds/jbu
borland:dslocal:C:\Borlan
borland:dslocal:C:\Borlan
borland:dslocal:C:\Borlan
borland:dslocal:C:\myclas
borland:dsremote://127.0
Definition
Enter SQL
New URL

URL	jdbc:borland:dsremote://127.0.0.1/c:/myclasses/ddlite.jds
Driver	com.borland.datastore.jdbc.DataStoreDriver
UserName	
Use Extended Properties	□
Extended Properties	

Driver:

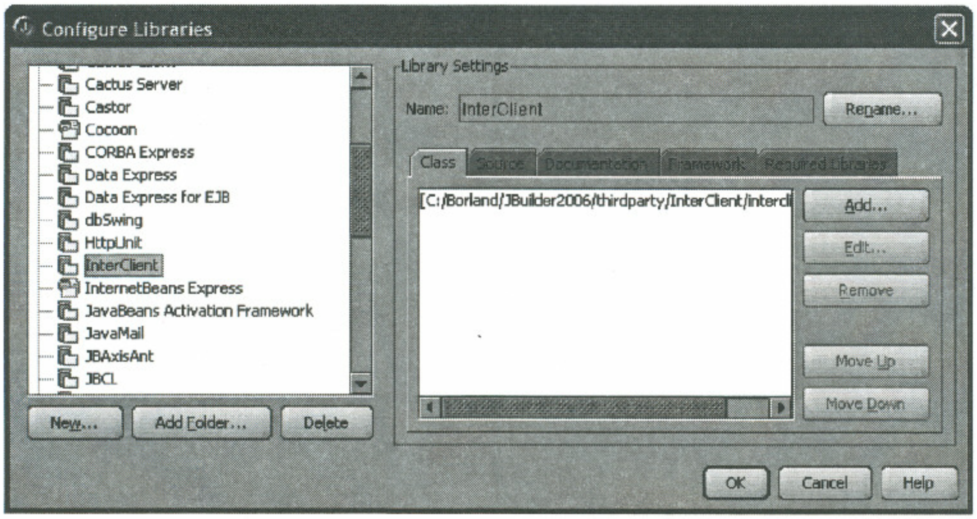
interbase.interclient.Driver

com.informix.jdbc.IfxDriver
com.sybase.jdbc2.jdbc.SybDriver
interbase.interclient.Driver
oracle.jdbc.driver.OracleDriver

Making InterBase available inside the JBuilder environment:

The next area that needs to be set up is the JBuilder libraries. These are files that logically manage the classes, .jar, .war, etc., so that each project can find the needed libraries. Remember, JBuilder constructs the actual classpath used by the program being created. This will be highlighted later in the JBuilder example.

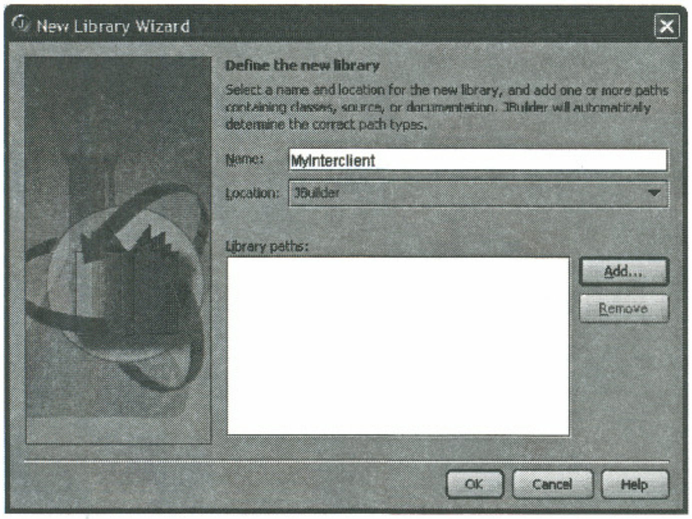
Managing libraries in JBuilder is rather easy, using the Tools|Configure|Libraries menu item. This will produce the following dialog:



Name: InterClient

[C:/Borland/JBuilder2006/thirdparty/InterClient/intercl

InterClient library may already be defined if you are using JBuilder 2006. When reviewing the libraries, if InterClient does not exist, press the New… button, and add the following parameters:



Select a name and location for the new library, and add one or more paths containing classes, source, or documentation. JBuilder will automatically determine the correct path types.

Name: MyInterclient

Location: JBuilder

In the library paths, click the Add… button, and add the InterClient.jar file found inside the InterBase lib directory. Then press OK to continue and press OK to finish out of the Library management area.

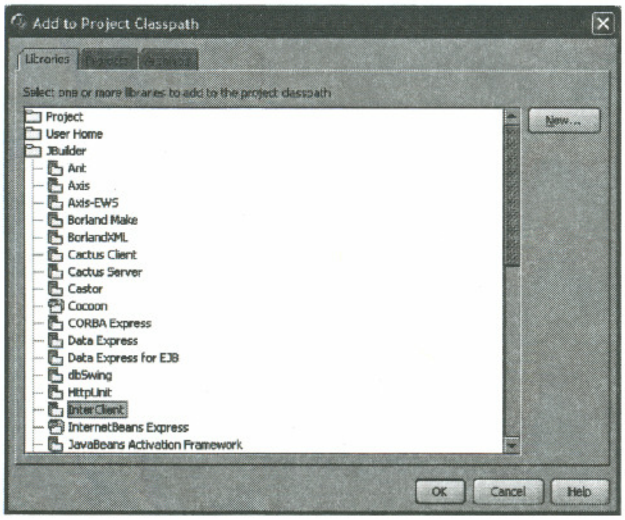
If there is an InterClient library defined, ensure that it is pointing to the latest driver available for use. If everything looks correct, it will be time to use InterBase inside JBuilder.

JBuilder Example:

This example is going to show how to use InterBase in a Java application, using the wizards to do most of the work. The main reason for this is to ensure that JBuilder is configured correctly to use InterBase.

1. Create a new project by pressing the File|New Project menu item.
2. In step one, give the project some meaningful name like InterBaseExample, and press the Next button.
3. In step two of the wizard, click on the required libraries tab. In a normal situation, that will be blank.

   a. Click the Add… button, and the following dialog should be displayed:

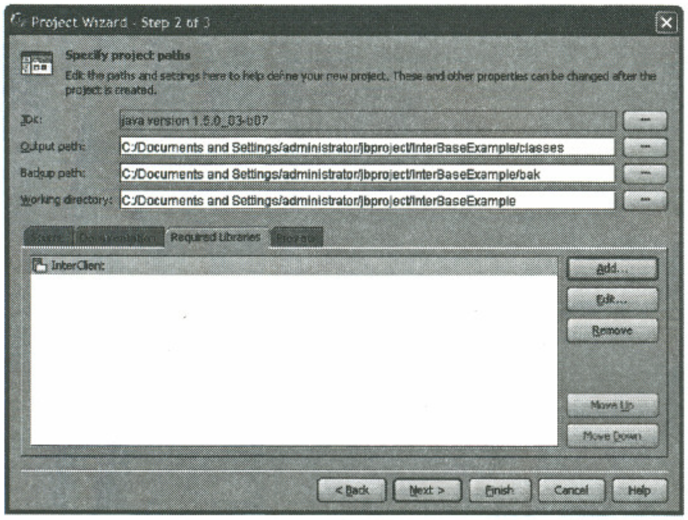


Add to Project Classpath

Select one or more libraries to add to the project classpath

   b. Click the InterClient library, and click the OK button.

4. This will then be added to the Required Libraries for the project.



Project Wizard - Step 2 of 3

Specify project paths

Edit the paths and settings here to help define your new project. These and other properties can be changed after the project is created.

JDK: java version 1.5.0_03-b07

Output path: C:/Documents and Settings/administrator/jbproject/InterBaseExample/classes

Backup path: C:/Documents and Settings/administrator/jbproject/InterBaseExample/bak

Working directory: C:/Documents and Settings/administrator/jbproject/InterBaseExample

Required Libraries: InterClient

5. Then click the Finish button. In the last step, nothing needs to be modified, so it can be skipped for this example.

Create the Application:

6. Now click the File|New, and select the General item from the left-hand column. Then, select the Application Icon, and click OK.

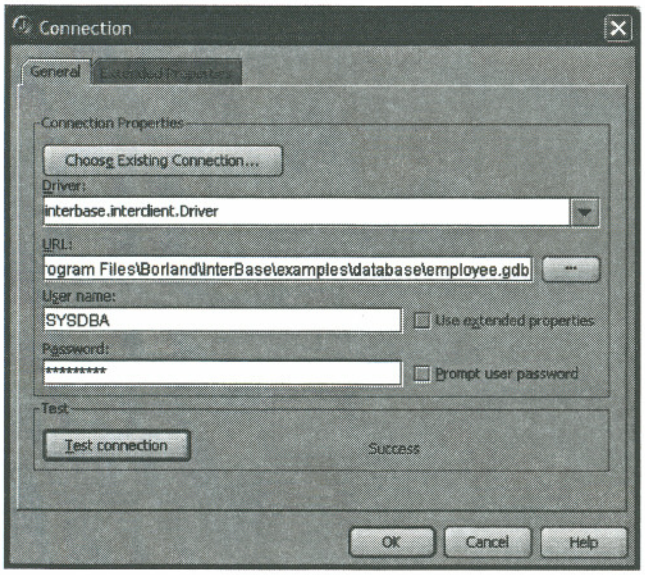
7. This will start the Application creation wizard. For this example, click the Finish button to continue.

8. This will create a Frame1.java, and it will be presented in the Content pane on the right-hand side of the environment.

9. Click the Design tab located under the Content pane, enabling the drag-&-drop component environment.

10. Now click the DataExpress expander, select a DataBase component, and drop it on the form. It will show in the Structure pane under Data Access folder.

   a. Now click on the connection property in the properties editor.



Driver: interbase.interclient.Driver

URL: rogram Files\Borland\InterBase\examples\database\employee.gdb

User name: SYSDBA

□ Use extended properties

Password: *********

□ Prompt user password

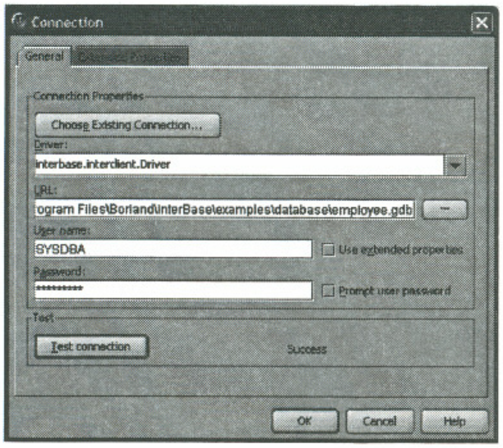
Success

   b. Set the properties:
   Driver: InterBase.interclient.Driver
   URL: {location of the database}
   User name: SYSDBA
   Password: masterkey

   c. Click the Test connection. Success should be displayed as above. Click the OK button to continue.

11. Next select a QueryDataSet component, and drop it on the form. Notice that it is also added to the Structure pane under the Data Access folder.

12. Click the query property in the Properties editor, displaying the following dialog:



Connection

General

Connection Properties

Choose Existing Connection...

Driver: interbase.interclient.Driver

URL: ogram Files\Borland\InterBase\examples\database\employee.gdb

User name: SYSDBA

□ Use extended properties

Password: *********

□ Prompt user password

Test

Test connection   Success

OK   Cancel   Help

13. Select the database1 from the drop-down selection box. Then add the SQL statement to the edit box. Select * from employee.

14. Click the Test Query button. Success should be shown as above, then click the OK button to continue.

15. Now click the dbSwing expander on the component palette, select the JdbNavToolBar, and drop it at the top of the form.

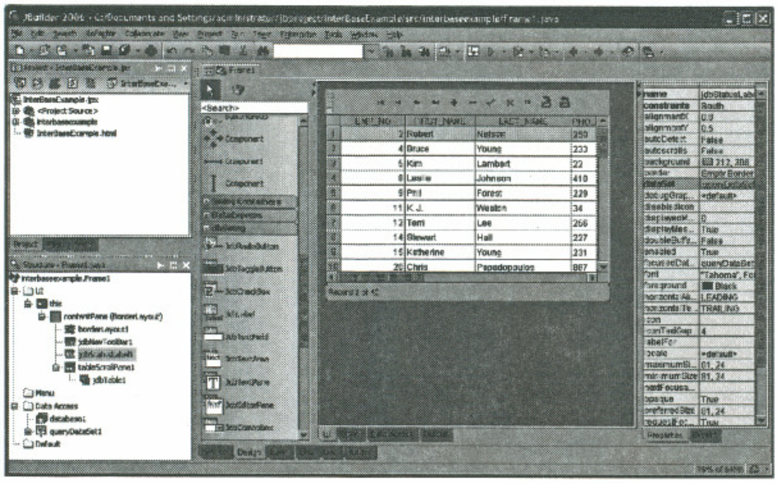
16. Now select the JdbStatusLabel, and drop it at the bottom of the form.

17. Select the TableScrollPane, and drop it into the center of the form.

18. Next, select the JdbTable, and drop it into the center of the TableScrollPane.

19. Now, holding the shift key, click the JdbNavToolBar, JdbTable, and the JdbStatusLabel, and set the dataset property in the Properties Editor to QueryDataSet1.

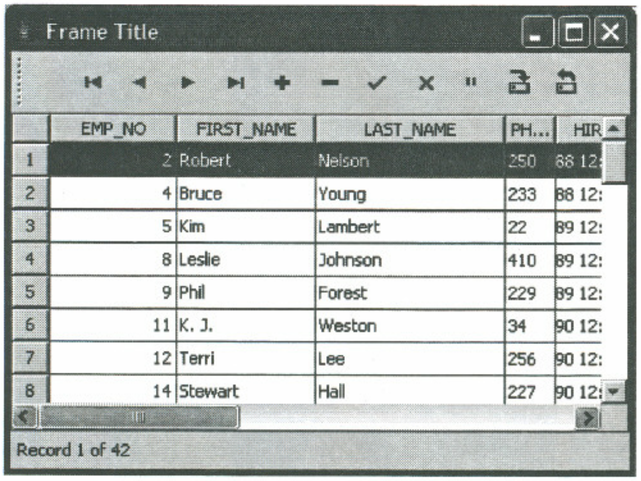
20. It should look something like the following:



	EMP_NO	FIRST_NAME	LAST_NAME	PHO
1	2	Robert	Nelson	250
2	4	Bruce	Young	233
3	5	Kim	Lambert	22
4	8	Leslie	Johnson	410
5	9	Phil	Forest	229
6	11	K.J.	Weston	34
7	12	Terri	Lee	256
8	14	Stewart	Hall	227
9	15	Katherine	Young	231
10	20	Chris	Papadopoulos	887

21. Notice, since the JBuilder environment understands the InterBase setup, it has live data inside the designers.

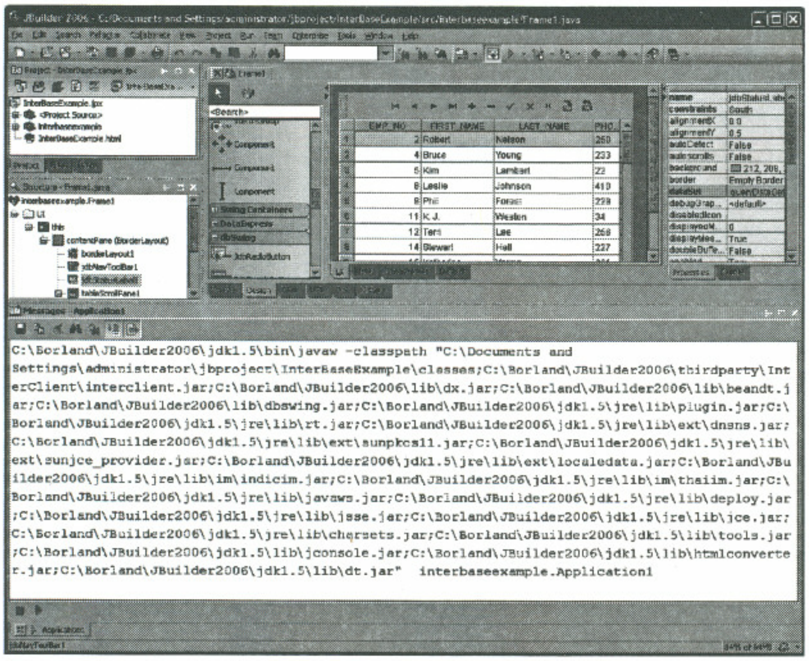
22. The last thing to do is run the application. Click the Run icon, and the following should be displayed:



	EMP_NO	FIRST_NAME	LAST_NAME	PH...	HIR
1	2	Robert	Nelson	250	88 12:
2	4	Bruce	Young	233	88 12:
3	5	Kim	Lambert	22	89 12:
4	8	Leslie	Johnson	410	89 12:
5	9	Phil	Forest	229	89 12:
6	11	K. J.	Weston	34	90 12:
7	12	Terri	Lee	256	90 12:
8	14	Stewart	Hall	227	90 12:

Record 1 of 42

23. Remember that the message pane has the exact command used to run the application. This command could be copied to a cmd or script file for execution outside JBuilder.



```
C:\Borland\JBuilder2006\jdk1.5\bin\javaw -classpath "C:\Documents and
Settings\administrator\jbproject\InterBaseExample\classes;C:\Borland\JBuilder2006\thirdparty\Int
erClient\interclient.jar;C:\Borland\JBuilder2006\lib\dx.jar;C:\Borland\JBuilder2006\lib\beandt.j
ar;C:\Borland\JBuilder2006\lib\dbswing.jar;C:\Borland\JBuilder2006\jdk1.5\jre\lib\plugin.jar;C:\
Borland\JBuilder2006\jdk1.5\jre\lib\rt.jar;C:\Borland\JBuilder2006\jdk1.5\jre\lib\ext\dnsns.jar;
C:\Borland\JBuilder2006\jdk1.5\jre\lib\ext\sunpkcs11.jar;C:\Borland\JBuilder2006\jdk1.5\jre\lib\
ext\sunjce_provider.jar;C:\Borland\JBuilder2006\jdk1.5\jre\lib\ext\localedata.jar;C:\Borland\JBu
ilder2006\jdk1.5\jre\lib\im\indicim.jar;C:\Borland\JBuilder2006\jdk1.5\jre\lib\im\thaiim.jar;C:\
Borland\JBuilder2006\jdk1.5\jre\lib\javaws.jar;C:\Borland\JBuilder2006\jdk1.5\jre\lib\deploy.jar
;C:\Borland\JBuilder2006\jdk1.5\jre\lib\jsse.jar;C:\Borland\JBuilder2006\jdk1.5\jre\lib\jce.jar;
C:\Borland\JBuilder2006\jdk1.5\jre\lib\charsets.jar;C:\Borland\JBuilder2006\jdk1.5\lib\tools.jar
;C:\Borland\JBuilder2006\jdk1.5\lib\jconsole.jar;C:\Borland\JBuilder2006\jdk1.5\lib\htmlconverte
r.jar;C:\Borland\JBuilder2006\jdk1.5\lib\dt.jar"  interbaseexample.Application1
```

24. Example done.

Using InterBase 7.5 with Together® Architect 2006:

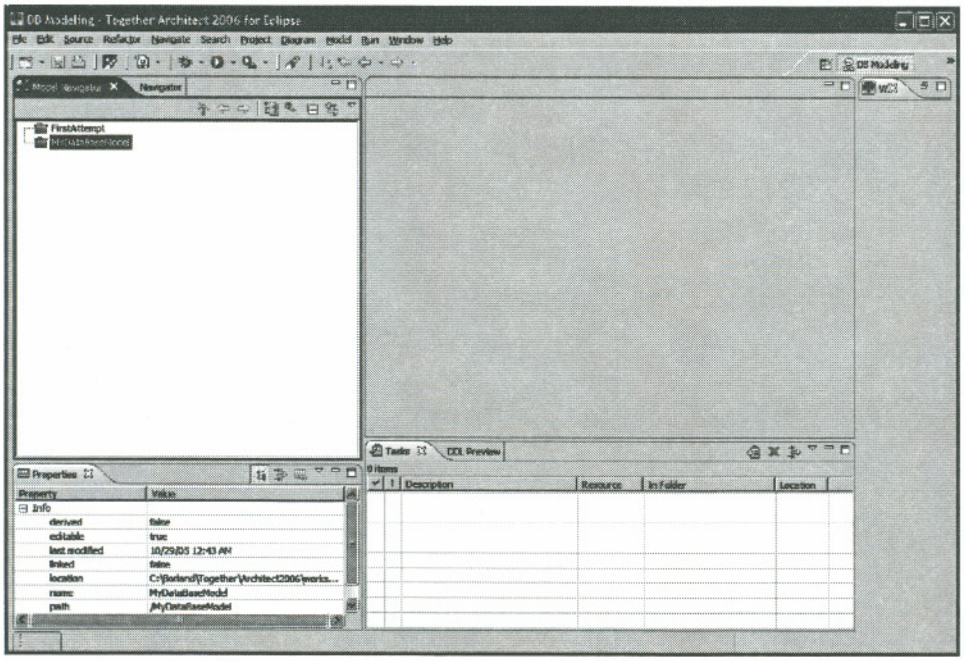
The Architect 2006 edition has a lot to offer developers. It is based on the Eclipse 3.1 framework and adds serious capabilities to help any IT shop get the most out of any one tool. This edition adds key technical capabilities like:

- MDA – Model Driven Architecture
- BPM – Business Process Modeling
- OCL – Object Constraint Language
- UML® 2.0 – Unified Modeling Language 2.0
- Enhanced pattern handling
- Enhanced Documentation generation
- Enhanced Data Modeling

It is recommended that readers of this paper download a copy of Together Architect 2006, and try it out. The example that will be shown will show how to create a new Data Modeling project, and Import the database Schema from an existing InterBase 7.5 database.

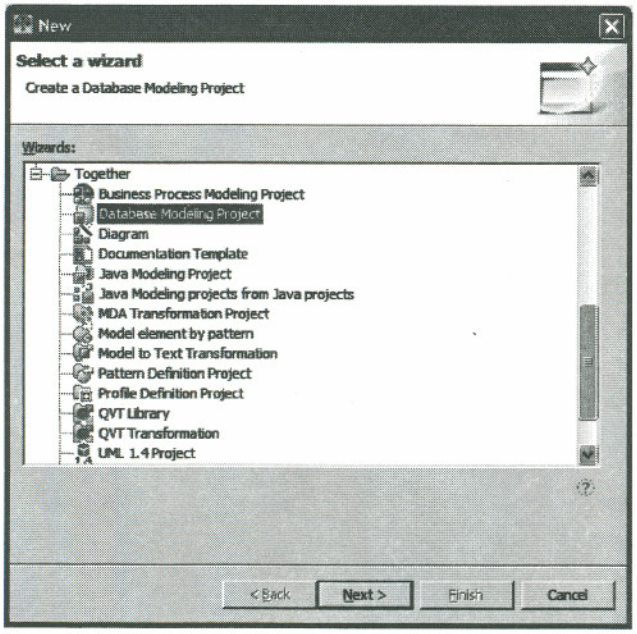
Creating a Project:

1. After installing and starting Together Architect 2006 on your machine, you should have a display similar to the following:



Property	Value
Info	
derived	false
editable	true
last modified	10/29/05 12:43 AM
linked	false
location	C:\Borland\Together\Architect2006\works...
name	MyDataBaseModel
path	/MyDataBaseModel

2. Create a new Data Modeling project by selecting the File|New|Other menu item, displaying the following:



New

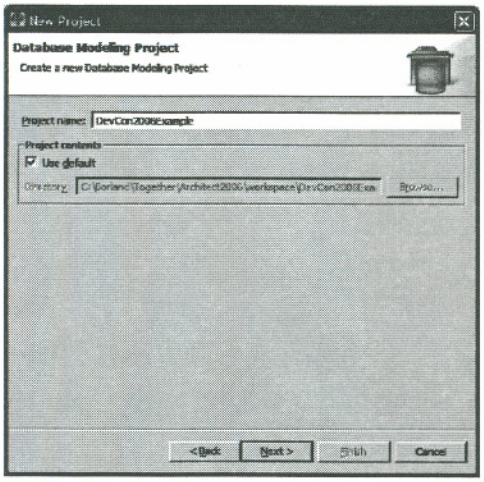
Select a wizard

Create a Database Modeling Project

Wizards:

- Together
  - Business Process Modeling Project
  - Database Modeling Project
  - Diagram
  - Documentation Template
  - Java Modeling Project
  - Java Modeling projects from Java projects
  - MDA Transformation Project
  - Model element by pattern
  - Model to Text Transformation
  - Pattern Definition Project
  - Profile Definition Project
  - QVT Library
  - QVT Transformation
  - UML 1.4 Project

3. Select the Database Modeling Project item under the Together folder. When the Next button is pressed, it will start the new project wizard.



New Project

Database Modeling Project

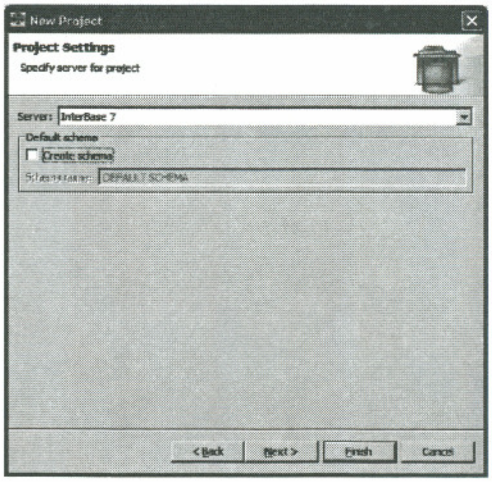
Create a new Database Modeling Project

Project name: DevCon2006Example

Project contents

☑ Use default

Directory: C:\Borland\Together\Architect2006\workspace\DevCon2006Exa



New Project

Project Settings

Specify server for project

Server: InterBase 7

Default schema

□ Create schema

Schema name: DEFAULT SCHEMA

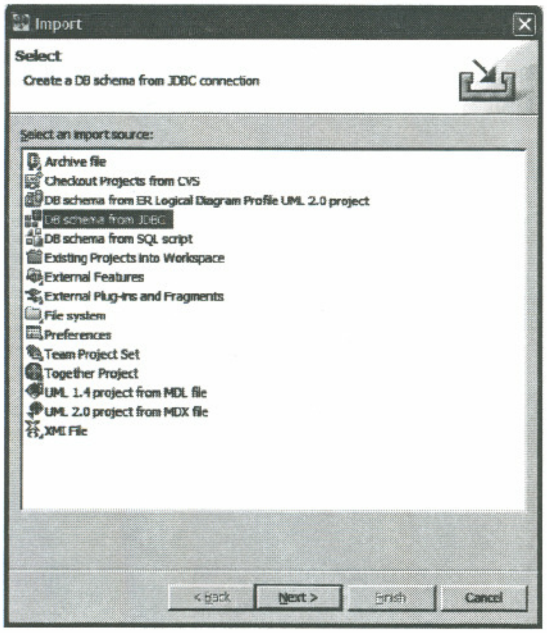
4. For this example, name the Project DevCon2006, and leave all other options selected by default as above.

5. Then in the Server, select InterBase 7, and deselect the Default Schema option. This is not needed since the steps outlined in the example will generate a schema from an existing database.

6. Select the Finished Button. This will complete the New Project wizard. It could ask, "Do you want to start a Data Model Perspective?" Click Yes to continue. If that option is not presented, the perspective will be changed to Data Model and be ready to continue.

Importing an existing database:

7. Now that there is an open Data Modeling project, the next step is to import a schema from an existing database. This can be accomplished by clicking the File|Import… menu item.



Import

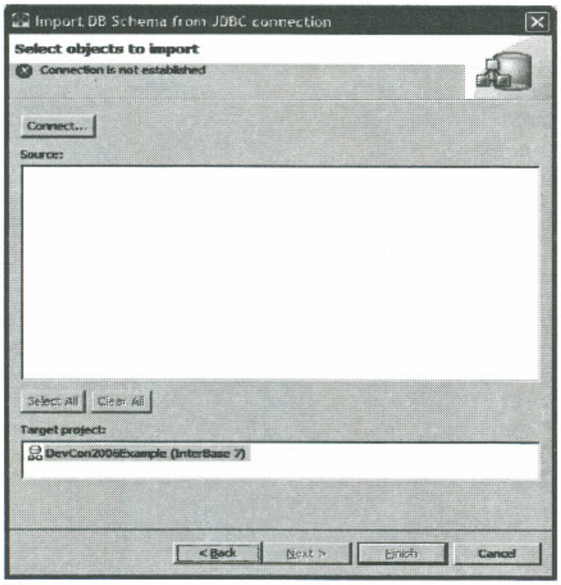
Select

Create a DB schema from JDBC connection

Select an import source:
Archive file
Checkout Projects from CVS
DB schema from ER Logical Diagram Profile UML 2.0 project
DB schema from JDBC
DB schema from SQL script
Existing Projects into Workspace
External Features
External Plug-ins and Fragments
File system
Preferences
Team Project Set
Together Project
UML 1.4 project from MDL file
UML 2.0 project from MDX file
XMI File
< Back
Next >
Finish
Cancel

8. Select the DB schema from JDBC and click the Next button.

   a. Currently this can only be done through JDBC, and you must know the location of InterClient.jar, which is InterBase's type 4 all Java driver.



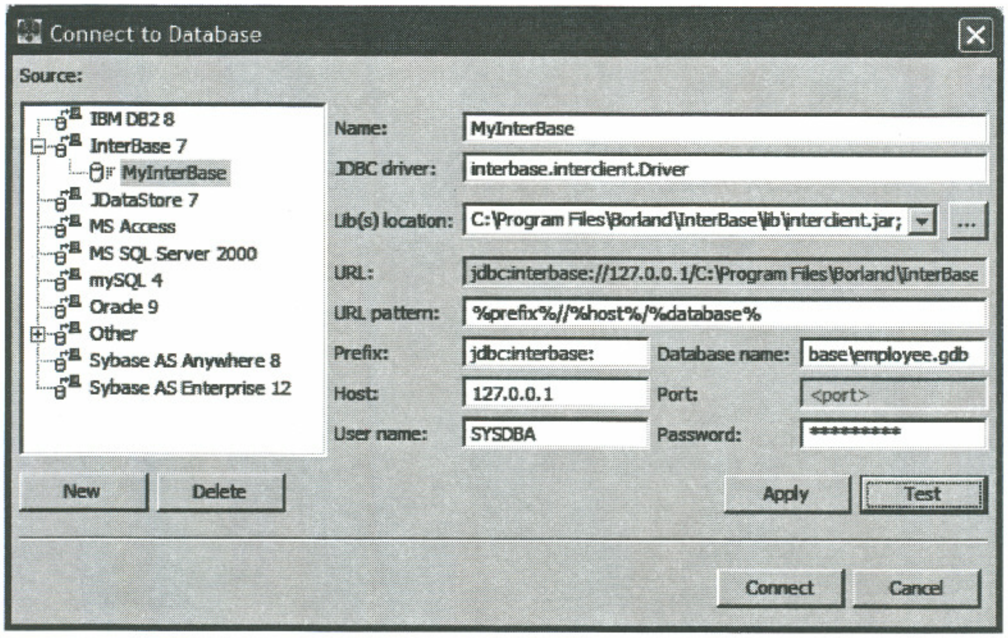
Import DB Schema from JDBC connection

Select objects to import

Connection is not established

Target project: DevCon2006Example (InterBase 7)

9. No connection may be found. Click the Connect… button. This will start the Connect Database wizard, where the configuration to InterBase will be established.



Name: MyInterBase

JDBC driver: interbase.interclient.Driver

Lib(s) location: C:\Program Files\Borland\InterBase\lib\interclient.jar;

URL: jdbc:interbase://127.0.0.1/C:\Program Files\Borland\InterBase

URL pattern: %prefix%//%host%/%database%

Prefix: jdbc:interbase:

Database name: base\employee.gdb

Host: 127.0.0.1

Port: <port>

User name: SYSDBA

Password: **********

10. Click the InterBase 7 item under the Source: pane, then click the New button under the same pane. This will mark the fields on the right to be editable. The following are the parameters used for this configuration:

- **Name:** MyInterBase
- **JDBC driver:** no change
- **Lib(s) location:** interclient.jar; This needs to be changed to the exact location of that file. In this example, it was located on the demo machine in the following location:

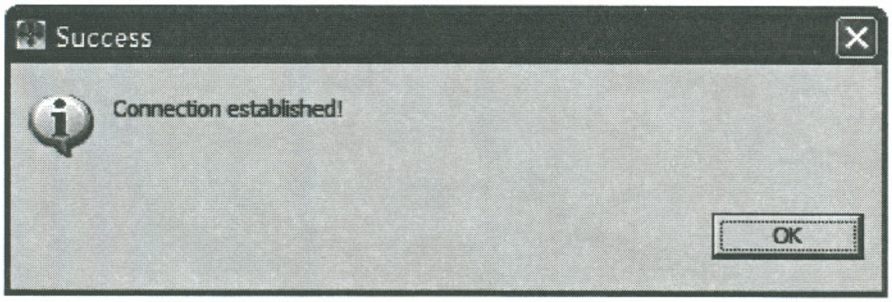
  C:\Program Files\Borland\InterBase\lib

  NOTE: This may have to be changed depending on the location of the install of InterBase.
- **URL:** - not modifiable
- **URL pattern:** This is showing how the connection string will be assembled.
- **Prefix:** - no change
- **Database name:** This needs to be changed to the location of the database. For this example, the standard employee.gdb was used from the examples:

  C:\Program Files\Borland\InterBase\examples\database\employee.gdb

- Host: 127.0.0.1
- Port: - not modifiable
- User name: SYSDBA
- Password: masterkey

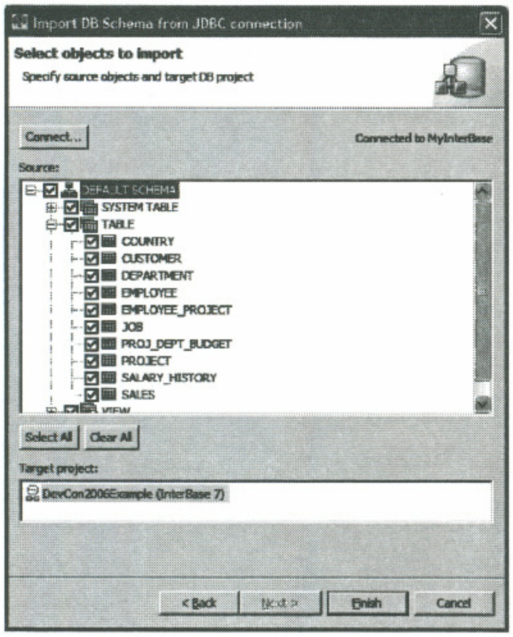
11. Once the above configuration is set, press the Apply button, and then press the Test button. A success dialog should be displayed.



Success

Connection established!

12. Next the Connect button should be pressed. This will establish the connection to the database, and return all of the meta-data known from the database and show the following selection dialog:



Import DB Schema from JDBC connection

Select objects to import

Specify source objects and target DB project

Connected to MyInterBase

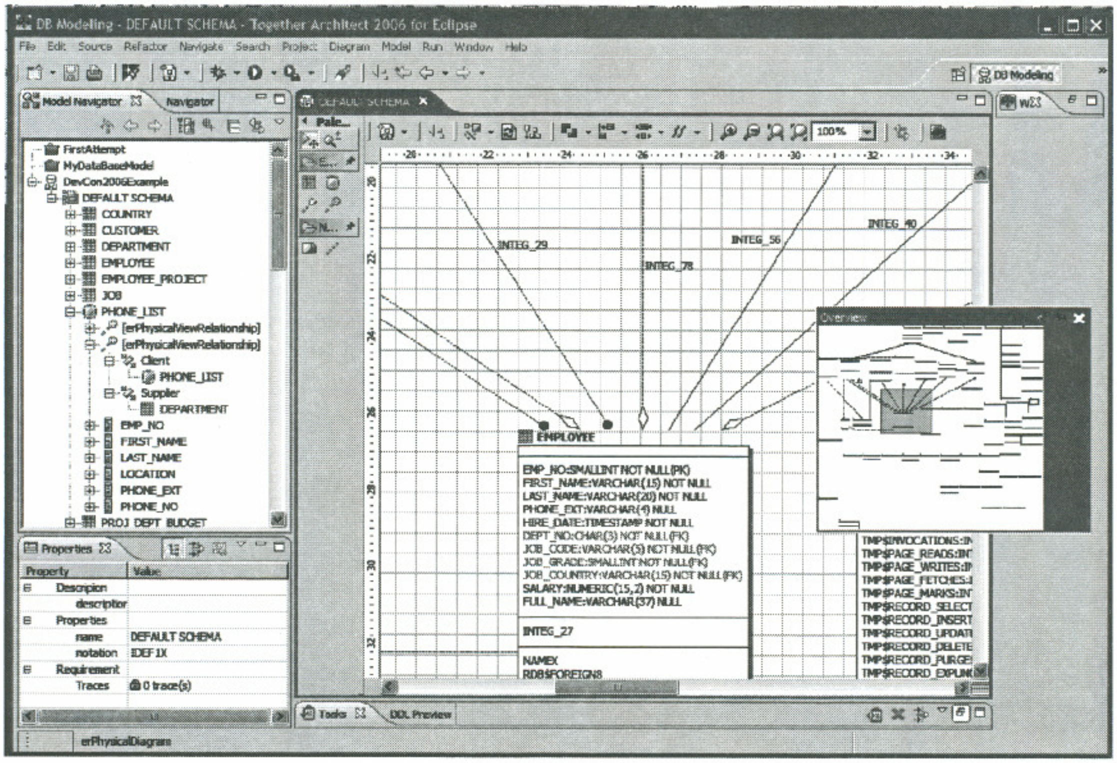
Source:

- ☑ DEFAULT SCHEMA
  - ☑ SYSTEM TABLE
  - ☑ TABLE
    - ☑ COUNTRY
    - ☑ CUSTOMER
    - ☑ DEPARTMENT
    - ☑ EMPLOYEE
    - ☑ EMPLOYEE_PROJECT
    - ☑ JOB
    - ☑ PROJ_DEPT_BUDGET
    - ☑ PROJECT
    - ☑ SALARY_HISTORY
    - ☑ SALES
  - ☑ VIEW

Target project:

DevCon2006Example (InterBase 7)

13. Items can be selected and deselected in this dialog. For this example, leave everything selected and click the Finish button. The schema should be generated and the following model should be displayed:



Property	Value
Description	
descriptor	
Properties	
name	DEFAULT SCHEMA
notation	IDEF1X
Requirement	
Traces	0 trace(s)

EMPLOYEE

EMP_NO:SMALLINT NOT NULL(PK)
FIRST_NAME:VARCHAR(15) NOT NULL
LAST_NAME:VARCHAR(20) NOT NULL
PHONE_EXT:VARCHAR(4) NULL
HIRE_DATE:TIMESTAMP NOT NULL
DEPT_NO:CHAR(3) NOT NULL(FK)
JOB_CODE:VARCHAR(5) NOT NULL(FK)
JOB_GRADE:SMALLINT NOT NULL(FK)
JOB_COUNTRY:VARCHAR(15) NOT NULL(FK)
SALARY:NUMERIC(15,2) NOT NULL
FULL_NAME:VARCHAR(37) NULL

INTEG_27

NAMEX
RDB$FOREIGN8

TMP$INVOCATIONS:IN
TMP$PAGE_READS:INT
TMP$PAGE_WRITES:IN
TMP$PAGE_FETCHES:I
TMP$PAGE_MARKS:INT
TMP$RECORD_SELECT
TMP$RECORD_INSERT
TMP$RECORD_UPDATI
TMP$RECORD_DELETE
TMP$RECORD_PURGE
TMP$RECORD_EXPUNG

From this point on, you can use Together as your Data Modeling tool of choice.

14. Example done.

Borland®

www.borland.com