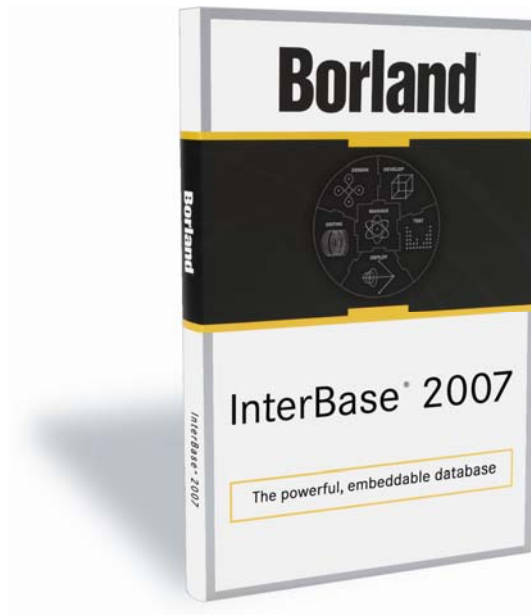


# The Ideal Database for Your Business

## InterBase® 2007

---

Ensure durability and reliability leveraging the powerful database for embedded and enterprise applications



A CodeGear White Paper

January 2007

---

**CodeGear**

## Table of contents

<b>Introduction</b> .....	<b>3</b>
<b>Crash Recovery</b> .....	<b>3</b>
<b>Data Protection and Recovery</b> .....	<b>4</b>
<b>Minimizing Cost</b> .....	<b>4</b>
<b>Database Maintenance</b> .....	<b>5</b>
<b>System Resource Requirements</b> .....	<b>5</b>
<b>Multi-Platform Support</b> .....	<b>5</b>
<b>Training Time</b> .....	<b>6</b>
<b>Monitoring Performance</b> .....	<b>6</b>
<b>Vendor Support</b> .....	<b>6</b>
<b>Licensing Cost</b> .....	<b>7</b>
<b>Minimize Development Time</b> .....	<b>7</b>
High Concurrency .....	7
Transaction Support.....	7
SQL Support .....	8
Stored Procedures, Triggers & Views.....	8
Events .....	8
Metadata .....	9
Security .....	10
Dependency Checking.....	10
<b>A Look at Alternatives</b> .....	<b>10</b>
Oracle 10g.....	11
Microsoft SQL Server 2005.....	12
MySQL 5.1 .....	14
Firebird 1.5.....	15
PostgreSQL 8.1 .....	16
<b>Get the Features You Need</b> .....	<b>17</b>
<b>Conclusion</b> .....	<b>21</b>

## Introduction

While large enterprises require complex database systems that support thousands of users, terabytes of data, and need a highly trained support staff, the needs of most businesses are more modest. InterBase is a powerful, reliable, low cost database that can easily support hundreds of concurrent users and gigabytes of data with no support during normal operation.

The most important characteristics of a database for the smaller enterprise are reliability and a low total cost of ownership (TCO). This paper examines the features that determine reliability and total cost of ownership and shows you why InterBase is the best choice for your business. To evaluate the reliability of a database you need to answer the following questions.

- Can I lose data if a workstation crashes?
- Can I lose data if the server crashes?
- Can I lose data if the hard drive in the database server fails?
- Can I lose data if the database server is stolen or destroyed?

## Crash Recovery

Because InterBase is a database server, client workstation crashes have no effect on the database. If you have a server crash, caused by loss of power or some other event, InterBase recovers reliably and immediately. When you restart InterBase all active transactions are automatically rolled back and the database is returned to a logically consistent state without any action on your part.

InterBase not only recovers automatically on restart after a server crash, it also recovers instantly as there is no change log that must be processed to rollback active transactions. Instead, InterBase simply changes the status bits for each active transaction to rolled back and leaves the row versions created by the rolled back transactions in the database. The versioning engine automatically ignores rolled back row versions and the garbage collector removes them automatically as tables are visited during normal database use.

## Data Protection and Recovery

InterBase provides a full suite of features to protect your data from disaster. The database can be backed up while in use. Both full and incremental backups are supported to minimize backup time. Backups automatically maintain a duplicate copy of the database on another machine. You can even use the backup database, in read only mode, while backups from the production server are updating it. A single command will switch the backup database to read/write mode so you can put it into production in an emergency.

The InterBase incremental backup system makes managing large databases easier because you do not have to do periodic full backups. Other database systems that support incremental backups require you to perform a full backup then periodic incremental backups. However, to recover the database you must restore the full backup then restore each incremental backup. This means you must do periodic full backups to limit the number of incremental backups that must be restored and thus limit the recovery time. Since InterBase incremental backups update a working copy of the database you never have to do a full backup after the first one and recovery is instantaneous; just switch the backup database from read only to read/write mode and you are back on line. There is no restore operation.

Enabling transaction journaling records all database changes in the InterBase 'write ahead' log. The journal file can be placed on a separate physical disk drive for protection from disk failure. Journal files can be automatically archived to a remote server and the database can be recovered from the remote server even if the production server has been physically destroyed. Journaling also allows point-in-time recovery to protect from accidental deletion of data or other logical damage to the database.

## Minimizing Cost

There are many factors that contribute to the TCO of a database. When evaluating the TCO ask yourself the following questions:

- How much maintenance does the database require?
- What are the hardware requirements?

- Does the database give me a choice of operating system platforms?
- How much will it cost to train my development staff to use the database?
- How easy is it to diagnose a performance problem?
- What are the licensing costs?
- What level of support is available from the database vendor?
- Does the database provide features to accelerate development time?

## Database Maintenance

A database that requires routine maintenance adds to TCO in one of two ways. You may be able to automate the maintenance tasks but automating routine maintenance adds to development time and cost. If the database requires maintenance that can not be automated you will need staff to perform the maintenance tasks and the cost will be ongoing.

InterBase databases require no maintenance. InterBase has features that are designed specifically for use at sites where no DBA or other support is available. Because InterBase is self-tuning and self-maintaining there is nothing for a DBA to do during normal operation.

## System Resource Requirements

The more hardware your database requires the higher the cost. A full installation of InterBase requires less than 39 MB on disk. If you do not install the documentation and examples InterBase requires less than 19 MB of disk space. Furthermore, InterBase requires just 32 megabytes of memory on all platforms.

## Multi-Platform Support

To minimize hardware costs your database needs to run on a platform that is already available at your site or it must give you the option to deploy a low cost platform such as Linux. InterBase runs on Windows®, Linux and Solaris™ to give you platform flexibility now and in the future.

## Training Time

The time required to train your development team is a significant component of both the total cost of ownership and the time it takes to deliver your project. It is also a good measure of product complexity.

InterBase is so easy to learn and administer that the most comprehensive training course for it, designed to train complete database novices to the point that they can confidently administer InterBase databases, lasts just five days. The jump-start course for experienced SQL database developers lasts just two days and many experienced developers find that they require no training at all. Instead, the InterBase manuals are all they need.

## Monitoring Performance

Once you deploy a multi-user database system there is always a chance that users will experience poor performance. Users may get access to the database with a third-party query and reporting tool then concoct queries that place a huge load on the database.

The important question is, when the unexpected happens, how many man-hours will it take to diagnose and correct the problem? InterBase includes a sophisticated performance monitor that offers both graphical and SQL interfaces. Not only can you see exactly what is happening in the InterBase server at any time, you can also log selected parameters over time for later review. This makes diagnosing performance problems a snap.

## Vendor Support

Many of today's database applications are mission critical. In a critical environment support from your database vendor should be available whenever you need it. InterBase offers a full suite of support options from 'per incident' support to 'around the world around the clock' support contracts.

## Licensing Cost

InterBase licensing costs are among the lowest of all commercial databases. Its low licensing cost combines with the other features described in this paper to give InterBase the lowest TCO for business applications.

## Minimize Development Time

A major component of database TCO is the cost of developing, maintaining and enhancing your application. If your database does not provide the features that make development easy your development team can waste a lot of time coding around the limitations of the database.

InterBase provides a rich feature set that makes developing any application fast and easy.

## High Concurrency

InterBase provides high concurrency in a mixed read/write environment because readers get a stable snapshot of the database without blocking writers. Inserts, updates and deletes never have to wait on read transactions regardless of their isolation level. In fact, snapshot transaction isolation is the default in InterBase. No matter how many SELECT statements you execute during the life of a snapshot transaction you will always get a stable consistent snapshot of the entire database as it was at the instant the transaction started.

To prevent a row that has been updated by an active transaction from being changed by another transaction, InterBase uses row level locking. The “locks” are actually implemented through the versioning engine, not by traditional two-phase locking, so you never have to worry about lock escalation, conversion deadlocks, or a large number of locks causing a severe resource load on the server. These problems simply do not exist in InterBase.

## Transaction Support

InterBase provides full ACID transaction support with both snapshot and read committed isolation levels. InterBase also supports two phase commit allowing a single transaction to include changes to two or more databases.

## SQL Support

InterBase provides a rich SQL implementation including ANSI 92 support with many extensions. A powerful cost-based optimizer provides optimum SQL execution speed. Using bitmap technology, indexes are combined to optimize performance with complex selection criteria. InterBase lets you extend SQL on any platform with user-defined functions. User-defined functions are implemented as DLLs on Windows and as shared object libraries on Linux and Solaris so you can create them using the language you prefer.

## Stored Procedures, Triggers & Views

InterBase provides both traditional executable stored procedures and selectable stored procedures. Selectable stored procedures are used in SQL SELECT statements in place of tables and, like a table, return a set of rows. You define the columns that comprise the returned rows when you write the stored procedure, then use any combination of SQL statements and computations to supply a value for each column. Using selectable stored procedures, developers save hours by easily creating virtual tables that are impossible to create with views.

For example, assume you need to produce a report that shows total sales for each of the last four quarters and for the last 12 months grouped by territory. With most databases you would create a temporary table with columns for the sales territory, the four quarterly totals and the twelve month total. Next you would insert a row for each territory; run a series of UPDATE/SELECT statements to compute the totals for each territory and insert them into the temporary table; and finally select the data from the temporary table to produce the report.

Using an InterBase selectable stored procedure, you declare output parameters to hold the sales territory, quarterly totals and twelve month total. The body of the stored procedure loops through the sales territories and selects the totals directly into the output parameters. That is all there is to it. To produce the report you just `SELECT * FROM PROCEDURE_NAME`.

InterBase's trigger implementation also reduces the amount of code developers have to write. InterBase provides both before and after triggers for *insert*, *update* and *delete* operations. The old and new value of each column is available in both *before* and *after triggers*. *Before insert* and

*before update* triggers let you validate or change the value in any column before the row is inserted or updated. You can also prevent the *insert*, *update* or *delete* from taking place by raising an exception in a *before trigger*. InterBase triggers are easy to write because the trigger fires for each row changed, not once for each statement, so you do not have to write looping code to iterate through the affected rows. InterBase also allows multiple triggers for each event, with complete control of trigger firing order, so you can modularize code for easy development, testing and maintenance.

Like most databases, InterBase lets you define a view using any valid SQL SELECT statement. Views are frequently used to flatten or denormalize the data model by joining multiple tables. However, many views that use joins or aggregate functions cannot be updated. This can force you to write additional code in your client applications to handle changes to the data. InterBase solves this problem by letting you create *before triggers* on a view. When a user makes a change the trigger fires and you can easily control which tables, columns and rows are changed.

## Events

When client applications must be notified of events in the database, many database management systems require you to poll the database at regular intervals or write complex add-on components. Events are built into InterBase. If your application needs to be notified when a part's quantity on hand drops below the reorder point all you need is an after update trigger to check the quantity and raise an event. Your client application simply registers interest in the event and is notified immediately when the event fires. Of course, you never have to worry about spurious events fired by transactions that subsequently rollback. No matter where you raise an InterBase event, the event is not actually sent to the client until the transaction commits.

## Metadata

InterBase provides a full range of data types for floating point, fixed point and integer numeric data; date, time and timestamp data, Boolean data, fixed and variable length character data and BLOBs for regular binary and character data. InterBase supports the null state for all data types. User defined data types (called domains) save both development and maintenance time by letting you define standard types for names, addresses, part numbers and other values that may appear

in multiple tables. Altering the domain changes the type everywhere it is used. InterBase supports multiple character sets including Unicode. A character set can be specified at the database, table or column level.

InterBase tables also provide computed columns so you never have to store values that can be computed on the fly. Default value, primary key, foreign key, and check constraints make both referential integrity and domain integrity enforcement automatic.

## Security

InterBase provides both server-wide and embedded database security models. With server wide security any user with a valid user name and password can log onto the InterBase server then connect to any database. If you use embedded user authentication, user names and passwords are stored in the database making it impossible for someone to kidnap your database and open it on another server. InterBase provides role based access control to database objects using the standard SQL GRANT and REVOKE commands.

## Dependency Checking

InterBase will not allow you to drop or alter any database object when another object depends on it. For example, you cannot drop a table if that table is used in a view or stored procedure. Likewise, you cannot drop a stored procedure that is called from a trigger or another stored procedure, nor can you drop a view that is used by a stored procedure or trigger. With InterBase you never have to worry about making a metadata change that will cause another object to fail.

## A Look at Alternatives

There are other databases you can consider for your applications but none of them were designed specifically for the smaller enterprise. InterBase alternatives can be divided into

commercial products, such as Oracle® 10g, Microsoft® SQL Server 2005 and MySQL 5.1, and open source databases, for example, PostgreSQL 8.1 and Firebird 1.5. MySQL can only be distributed free of charge as part of your application if the entire application is distributed under the GPL or an OSI approved open source license. Since very few business applications are distributed under an open source license MySQL is included with the other commercial databases.

Commercial databases are developed and supported by a single organization that can give you quick in-depth support in an emergency because the support staff has direct and immediate access to the R&D engineers who develop the product. If you encounter a bug or other problem with a commercial database you can get a patch or work-around without waiting for the next scheduled product release.

Open source databases appear to have a significant cost advantage because there is no deployment cost but open source databases also have a significant disadvantage in support. Open source databases are developed by groups of volunteers. There is no vendor organization to provide support. Instead, you must rely on third-party support vendors that do not have the level of access to the development team which you find with commercial products.

Because none of these commercial or open source products were designed specifically to support the smaller enterprise they all have weaknesses when compared to InterBase. The following sections examine some of their limitations.

## Oracle 10g

Oracle is one of the most powerful, complex and expensive databases available today. If you need to support thousands of users and terabytes of data Oracle is an excellent choice but applications in the smaller enterprise do not need to scale to that level. Oracle has hundreds of

---

<sup>1</sup> MySQL is not free when deployed with your application unless your entire application is deployed under the GPL or an OSI approved open source license. In all other cases you must purchase OEM licenses to deploy MySQL. Reference: <http://www.mysql.com/company/legal/licensing>

tuning and configuration parameters and is so complex Oracle certifies DBAs at three different skill levels<sup>2</sup>. Oracle XE can be deployed at no cost but has the following limitations.

- One database per machine
- One instance per machine
- Uses only one CPU
- Uses a maximum of one gigabyte of memory
- Database size is limited to four gigabytes of user data

In reality, Oracle XE cannot be used for most applications in any but the smallest business due to the first two restrictions. Consider what happens when you try to add a second application to an existing application server and discover that another application using Oracle XE is already installed on that machine. Since Oracle XE only allows one database per machine and one Oracle XE instance per machine you cannot install a second application that uses Oracle XE. Your only choices are to install another application server or upgrade to the full Oracle product. Suddenly Oracle XE has become a very expensive choice. In addition, Oracle XE has a disk footprint of over 1.1 gigabytes. That is 30 times the size of a full InterBase installation and 62 times the size of InterBase without documentation and examples.

## Microsoft SQL Server 2005

Microsoft SQL Server is another enterprise database with the features and complexity necessary to handle very large numbers of users and very large databases. SQL Server uses a complex locking model that can cause poor concurrency if developers do not understand how to avoid its traps. "As the Microsoft SQL Server Database Engine acquires low-level locks, it also places intent locks on the objects containing the lower-level objects."<sup>3</sup> "The Database Engine does not escalate row or key-range locks to page locks, but escalates them directly to table locks.

---

<sup>2</sup> [http://education.oracle.com/pls/web\\_prod-plq-dad/db\\_pages.getpage?page\\_id=47&p\\_org\\_id=1001&lang=US](http://education.oracle.com/pls/web_prod-plq-dad/db_pages.getpage?page_id=47&p_org_id=1001&lang=US)

<sup>3</sup> <http://msdn2.microsoft.com/en-us/library/ms184286.aspx>

Similarly, page locks are always escalated to table locks.”<sup>4</sup> Having entire tables locked without warning causes serious concurrency limitations in a multi-user environment.

To avoid the pitfalls of lock escalation and conversion deadlocks you will need one or more certified DBAs on your development team. Certification as a Microsoft SQL Server developer and DBA requires 18 days of classroom instruction plus three examinations.

Developing applications with Microsoft SQL Server typically requires substantially more code compared to InterBase. One reason is Microsoft SQL Server’s trigger implementation. Microsoft SQL Server does not have *before* triggers. If you need to calculate or modify values being inserted or updated, you must create an *instead of* trigger and write the code to perform the insert or update, in addition to your data validation or calculation code. Even that option may not be available. “*Instead of update* and *delete* triggers are not allowed on tables that are targets of cascaded referential integrity constraints.”<sup>5</sup> In addition, only one *instead of* trigger is allowed for each action so you cannot modularize your code. Microsoft SQL Server triggers fire once per statement, not once for each affected row, so, in addition to the code you would have to write with InterBase, you must declare a cursor and write code to iterate over the affected rows.

Microsoft SQL Server 2005 Express Edition can be deployed at no cost but it has limitations.

- Uses only one CPU
- Database size is limited to 4 gigabytes
- Buffer pool memory is limited to 1 gigabyte

Many of today’s applications must store more than four gigabytes of data. If you need to store BLOB data such as text documents, scanned images, photographs, sound or video you can fill a four gigabyte database very quickly. If there is any possibility that any of your sites may exceed these limits you will have to monitor your installations carefully so you can upgrade to the full version of Microsoft SQL Server before you run out of space in the database or performance degrades. Since SQL Server 2005 Express Edition is just a limited version of Microsoft SQL

---

<sup>4</sup> <http://msdn2.microsoft.com/en-us/library/ms184286.aspx>

<sup>5</sup> Books Online – DML Trigger planning guidelines

Server, developers require the same level of knowledge and skill to develop applications using the Express Edition that they must have to work with the full product.

## MySQL 5.1

MySQL is not a typical database management system but rather a front end for a family of nine different storage engines<sup>6</sup>. All of the engines have different table formats, concurrency control mechanisms and other features. This means that you cannot simply decide to use MySQL. You must evaluate the storage engines against your application's requirements and select the one whose characteristics best meet your application's needs. For example, the MyISAM engine was designed for fast processing of SELECT statements but uses table locks for insert, update and delete statements. This can cause poor performance in a multi-user environment where data is being changed.

When selecting a storage engine you must also carefully consider its longevity. For example, the ISAM engine has been deprecated and replaced by the MyISAM engine while the Berkeley DB engine has been dropped. In addition, some of the storage engines are third-party products. The InnoDB engine, arguably the most feature rich of all of the storage engines, is now owned by Oracle Corporation. Will the InnoDB engine still be available five years from now and, if so, at what cost? Uncertainty about the future adds risk to the decision to use a third-party storage engine.

MySQL's implementation of triggers and stored procedures is limited. Triggers are not activated by the action of foreign keys<sup>7</sup>. If you need to provide an audit trail and you have foreign keys with the ON DELETE CASCADE or ON UPDATE CASCADE option, you cannot use a trigger to log changes because the child table's trigger will not fire when your updates are cascaded. In addition, triggers cannot modify any table that is already being used, either for reading or writing, by the statement that invoked the trigger<sup>8</sup>. You must also be careful about how you write and use

---

<sup>6</sup> MySQL manual page 561

<sup>7</sup> <http://dev.mysql.com/doc/refman/5.0/en/routine-restrictions.html>

<sup>8</sup> <http://dev.mysql.com/doc/refman/5.0/en/routine-restrictions.html>

stored procedures as all the restrictions that apply to triggers, but not to stored procedures, do apply to stored procedures that are called by triggers.

Views have a number of surprising limitations. In some cases indexes on base tables are not used<sup>9</sup>. You can drop or alter a table that is used in a view without warning. You will not learn that the view is now invalid until you try to use it<sup>10</sup>. A database backup using the mysqldump utility will fail if a view exists that was created by a user that does not have the SHOW VIEW privilege<sup>11</sup>.

MySQL has many other limitations that make it a poor choice for applications in the smaller enterprise.

- MySQL has very limited performance monitoring.
- MySQL does not support multi-instance installations. That makes migration from one version to the next more difficult and prevents isolation of one application from other applications. It also prevents using multiple instances for load balancing.
- MySQL does not have an integrated mechanism to notify clients of database events.
- MySQL does not have incremental backup. Instead, you must start a new binary log then copy the old log files to the backup location<sup>12</sup>.
- MySQL does not support user defined data types.
- MySQL includes over 100 system variables plus numerous configuration parameters that control its operation; it is not self-tuning.
- MySQL has a query cache, key cache, table cache, InnoDB buffer and InnoDB log buffer. You must decide how to allocate memory among these five caches to get the best performance.

## Firebird 1.5

Firebird is an open source database that has limited scalability because it does not have multi-processor support. To take advantage of multiple CPUs you must use the “classic” version, which

---

<sup>9</sup> <http://dev.mysql.com/doc/refman/5.0/en/view-restrictions.html>

<sup>10</sup> <http://dev.mysql.com/doc/refman/5.0/en/view-restrictions.html>

<sup>11</sup> <http://dev.mysql.com/doc/refman/5.0/en/view-restrictions.html>

<sup>12</sup> MySQL manual page 359-360

spawns a separate process for each user. However, the classic architecture does not scale well. Having a separate Firebird server process and database cache for each user consumes resources rapidly as the number of users grows.

A more serious limitation is that Firebird provides limited data recovery in case of media failure or any other event that damages the database. You can restore from your last backup but all changes since the last backup will be lost. The only way to provide recovery with little or no data loss is to purchase a third-party replication tool and implement replication to another server. This will protect your data but it also increases development time and deployment cost and degrades performance.

Firebird has no performance monitoring tools. Diagnosing performance problems at remote sites is very difficult. In addition, Firebird does not support temporary tables. Tasks that can be done easily with temporary tables require additional coding when using Firebird.

Like other open source databases, the only source of support for Firebird is third-party organizations. Unfortunately, third-party support providers do not have the access to and integration with the engineers developing the product that you find with commercial databases like InterBase and this limits both the level of support you can get and the speed with which you can get it.

## PostgreSQL 8.1

PostgreSQL is another open source database that uses versioning architecture. However, PostgreSQL is neither self-tuning nor low maintenance. PostgreSQL has over 170 configuration options that must be set to control performance. In addition, it does not automatically perform garbage collection as data is accessed. Instead, you must perform a vacuum operation frequently to remove old row versions from the database<sup>13</sup>.

PostgreSQL logs information to a log file. “However, the log output tends to be voluminous (especially at higher debug levels) and you won’t want to save it indefinitely. You need to “rotate”

---

<sup>13</sup> PostgreSQL manual 22.1.2

the log files so that new log files are started and old ones removed after a reasonable period of time.”<sup>14</sup>

Like Firebird and other open source databases, support for PostgreSQL is only available from third-party providers. Consider carefully whether you can get the level of support you need before you commit to any open source database.

## Get the Features You Need

Minimizing the TCO of your database means getting the features you need to minimize maintenance, support, hardware, operating system, training, development and troubleshooting costs. The feature matrix below shows why InterBase is truly the ideal database for the smaller enterprise.

<b>Feature</b>	<b>InterBase 2007</b>	<b>Oracle 10g</b>	<b>SQL Server 2005</b>	<b>MySQL<sup>15</sup> 5.1</b>	<b>Firebird 1.5</b>	<b>PostgreSQL 8.1</b>
Zero maintenance	√				√	
Requires certified DBA		√	√			
Support contracts available from vendor	√	√	√	√		
Transaction support	√	√	√	√	√	√
SQL support	√	√	√	√	√	√
Multi-user and single user versions	√				√	

---

<sup>14</sup> PostgreSQL manual 22.3

<sup>15</sup> MySQL features assume you are using Oracle Corporation's InnoDB storage engine

---

User-defined functions	√	√	√		√	√
Cost-based optimizer	√	√	√	√	√	√
Provides a consistent snapshot of data without blocking updates	√	√	√	√	√	√
Row level locking	√	√	√ <sup>16</sup>	√	√	√
No lock escalation	√	√		√	√	√
No conversion deadlocks	√	√		√	√	√
Snapshot transaction isolation by default	√	√		√	√	√
Two phase commit	√	√	√	√	√	√
SQL based performance monitoring	√	√				√
Graphical performance monitoring	√	√	√			
SMP support	√	√	√	√		√
Multi-core support	√	√	√	√		√
Hyperthreading support	√	√	√	√		√
Executable stored procedures	√	√	√	√	√	√
Selectable stored procedures	√				√	

---

<sup>16</sup> Row locks can escalate to table locks without warning

Before triggers	√	√		√	√	√
After triggers	√	√	√	√	√	√
Old & new values available in before & after triggers	√			√	√	√
Multiple triggers on same event	√	√			√	√
Control of trigger firing order	√			√	√	
Triggers fire for cascaded updates & deletes	√	√	√		√	√
Views	√	√	√	√	√	√
Before triggers on views	√				√	
Temporary tables	√	√	√	√		√
Fire client events	√	√			√	√
Automatic key generation	√	√	√	√	√	√
Unicode support	√	√	√	√	√	√
Supports the null state for all data types	√		√	√	√	√
Declarative referential integrity	√	√	√	√	√	√
Check constraints	√	√	√	√	√	√
Default values	√	√	√	√	√	√

Computed columns	√	√	√	√	√	√
User defined data types	√		√		√	√
Online backup	√	√	√	√ <sup>17</sup>	√	√
Online incremental backup	√	√	√		√	√
Backups maintain a backup database on another server	√					
Online metadata changes	√	√	√	√	√	√
Immediate crash recovery	√				√	
Journaling for point-in-time recovery	√	√	√	√		√
Archive journal files to another server	√	√	√	√ <sup>18</sup>		√
Role based access control	√	√	√		√	√
Multiple instances running on the same server simultaneously	√	√	√			√ <sup>19</sup>
Runs on Windows, Linux & Solaris	√	√		√	√	√
Low licensing cost	√			√	√	√
Memory required	32 MB	Not Published	512 MB	Not Published	32 MB	Not Published

<sup>17</sup> Requires InnoDB Hot Backup available at extra cost from Oracle Corporation

<sup>18</sup> Replication can be used to maintain a backup on another server. MySQL Manual page 947

<sup>19</sup> Requires compilation from source if instances are different versions

Disk footprint	19 MB without examples or manuals. 39 MB with.	>1.1 GB	350 MB 775 MB with documentation & sample databases	140 MB	15.5 MB	77.5 MB
----------------	--	---------	---	--------	---------	---------

## Conclusion

This whitepaper has shown the benefits of InterBase for the SME market to outweigh the high cost of other databases designed for the large enterprise. InterBase gives you high reliability, built-in disaster recovery, zero database maintenance and a host of features that make application development and deployment easy and reliable at the lowest total cost.

### About CodeGear

CodeGear, formerly Borland's Developer Tools Group, delivers innovative, high-productivity development tools for a wide spectrum of software developers ranging from individuals to enterprise teams. CodeGear's products enable developers to freely develop on their platform of choice while focusing on simplifying complex technologies and tasks so they can concentrate on application design, not infrastructure, to ensure on-time project delivery.

Borland, InterBase and all other Borland brand and product names are service marks, trademarks or registered trademarks of Borland Software Corporation or its subsidiaries in the United States and other countries. Microsoft, Windows and all other Microsoft brand and product names are service marks, trademarks or registered trademarks of Microsoft Corporation or its subsidiaries in the United States and other countries. Solaris and all other Sun brand and product names are service marks, trademarks or registered trademarks of Sun Microsystems, Inc. or its subsidiaries in the United States and other countries. Oracle and all other Oracle brand and product names are service marks, trademarks or registered trademarks of Oracle Corporation or its subsidiaries in the United States and other countries. All other marks are the property of their respective owners.

### Safe Harbor Statement

This document contains "forward-looking statements" as defined under the U.S. Federal Securities Laws, including the Private Securities Litigation Reform Act of 1995 and is subject to the safe harbors created by such laws. Forward-looking statements may relate to, but are not limited to, the features available in, and the potential benefits to be derived from, Borland and CodeGear products and solutions, and the release dates, plans and market acceptance of such products and solutions. Such forward-looking statements are based on current expectations that involve a number of uncertainties and risks that may cause actual events or results to differ materially. Factors that could cause actual events or results to differ materially include, among others, the following: rapid technological change that can adversely affect the demand for Borland and CodeGear products, shifts in customer demand, shifts in strategic relationships, delays in Borland's or CodeGear's ability to deliver their products and services, software errors or announcements by competitors. These and other risks may be detailed from time to time in Borland periodic reports filed with the Securities and Exchange Commission, including, but not limited to, its latest Annual Report on Form 10-K and its latest Quarterly Report on Form 10-Q, copies of which may be obtained from [www.sec.gov](http://www.sec.gov). Neither Borland nor CodeGear is under any obligation to (and expressly disclaims any such obligation to) update or alter its forward-looking statements whether as a result of new information, future events or otherwise. Information contained elsewhere in our website is not incorporated by reference in, or made part of this document.