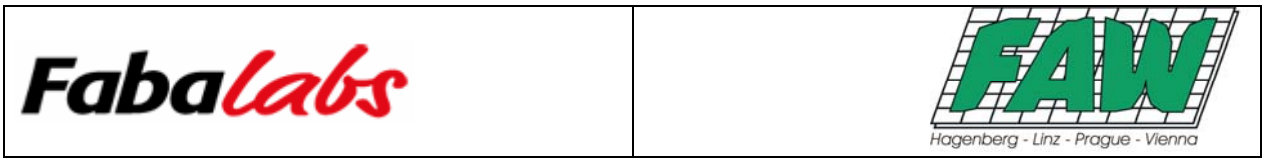


Research Paper

Comparison of the Enterprise Functionalities of
Open Source Database Management Systems

April 26, 2005

Herausgeber und Urheber
Fabalabs Software GmbH, Honauerst. 4 A-4020 Linz

Bearbeiter:
Dipl.-Ing. Daniel Fallmann
Dipl.-Ing. Helmut Fallmann
Dipl.-Ing. Andreas Pramböck
Horst Reiterer
Dipl.-Ing. Martin Schumacher
Dipl.-Ing. Thomas Steinmaurer
Univ.-Prof. Dr. Roland Wagner

Index

Samples

1 Introduction

1.1 Purpose of the Document

Ever since the very beginning, the enterprise-grade database management systems (DBMS) market was clearly dominated by commercial offerings from IBM, Oracle and Microsoft. Meanwhile the situation has changed: Open source DBMS have reached a high grade of evolution and are seen as a cost effective and functionally adequate alternative to commercial DBMS by many business companies. Besides the cost aspects the question if open source DBMS can cope with the demands that are given by enterprise-level applications comes to our mind.

To give an answer to this question, Fabalabs Software GmbH and the Institute For Applied Knowledge Processing (FAW) evaluated Firebird 1.5.2, Ingres r3 3.0.1, MaxDB 7.5.0.23, MySQL 4.1.10 and PostgreSQL 8.0.1 in respect to their applicability on enterprise-level applications.

1.2 Abstract

This study is an in-depth evaluation on open source DBMS in consideration of their applicability on enterprise-level applications. Based on a catalog of requirements typical for enterprise-level applications, product manuals, books, websites, newsgroups and other DBMS-related resources have been worked through to give an extensive answer on requirements like licensing, operating system support, certain limits of a database, database interfaces, programming features, text searching capabilities, performance of the database, distributed transaction support, replication, high availability features, ease of administration and a technical appraisal in regard to the project activity, size of the development team and support availability. Additionally, three so-called “limit tests” have been developed and executed against each DBMS to test limits relevant for complex query execution. These tests are discussed in section 4 in more detail. No total cost of ownership (TCO) analysis has been done however and not every feature has been tested practically.

The following open source DBMS have been evaluated: Firebird 1.5.2, Ingres r3 3.0.1, MaxDB 7.5.0.23, MySQL 4.1.10 and PostgreSQL 8.0.1

1.3 Feedback and Further Information

Feel free to cantact us for feedback or further information:

Fabalabs Software GmbH
Honauerstr. 4
A – 4020 Linz

Dipl.-Ing. Daniel Fallmann
daniel.fallmann@fabalabs.org

1.4 Demands of Enterprise Applications on Database Systems

The database world has changed over the last years. Behind the scenes the world of relational DBMS is not as easy as it sometimes maybe seems to be. A DBMS has lots of components, or modules, including modules for implementing the catalog, processing the interface, processing a query

language, accessing data, buffering data, controlling concurrency and last but not least it has to handle data recover and security in an adequate way, too.

This all is quite common knowledge to DBMS system designers and implementers. There are indeed many ways to solve problems, but one should still keep in mind that different solutions often have a different impact to performance, stability and usability. Nowadays we are dealing with DBMS up to many terabytes of footprint on disk. When dealing with actual enterprise applications we have to concern that new problems arrive.

What is it that we have to draw our attention to? Well, actual enterprise applications and especially the databases in the background have to deal with ten thousands of tables, hundred thousands of columns, huge footprints in memory and disk, many different languages, hundreds or sometimes thousands of transactions per second, hundred thousands of DB calls per second and not to forget today's hardware, like perfect SMP scalability, shared memory access, etc. Nevertheless, there has a lot more to be considered for databases than the usage of the right architecture and the most appropriate algorithms. For the daily work one needs useful and therefore easy to handle tools for debugging, profiling, backup data, recover data and for maintenance. This study looks at five open source databases and looks into the aspects of enterprise applications that use huge database systems.

We deal with the most important aspects we consider when we decide which database is most suitable for our needs. This addresses for example full-text search, soundex, XA-compliance, limits of the database (number of tables, size of columns, kinds of data to be stored, …), indexing, online reorganisation, backup (full and incremental), load balancing, internal data structures, ease of maintenance (SNMP support, …), complex queries (maximum number of conditional connections in one statement, maximum number of tables that can be joined, maximum size of IN list, …), connection interfaces (native, ODBC, JDBC, …), locking (grain), isolation levels, triggers (level, timing, nested), connectivity (gateways to other DBMS) and much more.

2 Evaluation and Comparison

The following section is the core part of this document, comparing open source DBMS in different areas.

The evaluation starts in section 2.1 with a description of license terms followed by an extensive list of supported operating systems and a list of supported 64-bit architectures in section 2.2.

Section 2.3 discusses limits of the products including per database, table, index, and data type limits, and if the DBMS supports Unicode. The section ends with the so-called “limit test” trying to determine different limits important for complex queries. The implemented tests are discussed in more detail in section 4.

Section 2.4 proceeds with an analysis of the supported SQL standard, available language interfaces for database developers and the support of database links and gateways.

After that, ACID-compliant transaction support, transaction isolation levels, support for prepared statements, the capability to run code modules on the server via stored procedures, triggers and much more is evaluated in section 2.5.

Section 2.6 is dedicated to evaluating enhanced search capabilites in character data like full-text and soundex search.

Section 2.7 discusses data structures for table and index storage and outlines scalability and load balancing capabilities as well as SMP (Symmetric Multiprocessing), multi volume and large memory support.

Distributed transaction support is another important feature for an enterprise-ready DBMS. Section 2.8 explores the support of distributed transactions, and if the used protocol is XA-compliant as well as the ability to recover a multi-database transaction in case of a server or network failure.

Section 2.9 covers replication facilities available in the evaluated DBMS.

Some high availability features are discussed in section 2.10, including online backup, online reorganization and cluster support.

Section 2.11 covers the area of administrating the DBMS. This includes SNMP (Simple Network Management Protocol) support, the need for a full-time database administrator, availability of graphical database administration and development tools, performance tuning and profiling tools. This section will also discuss some basic security related topics.

Section 2.12 highlights the activity of the open source project, estimated values regarding release cycle, size of the development team, availability of free and commercial support and official future trends including planned features for forthcoming versions.

2.1 Licensing

Before comparing the products from a technical point of view, the applicable open source license is listed. This is important because open source doesn’t necessarily mean that the DBMS can be used in commercial applications for free, which might be an important business requirement when following the open source route, instead of choosing a commercial product.

Firebird 1.5.2	The original modules released by Inprise are licensed under the InterBase Public License (IPL). [FIPl00] New code modules added to Firebird are licensed under the Initial Developer's Public License (IDPL). [FIDPL04] Both licenses are modified versions of the Mozilla Public License v.1.1. [MoPl05] Firebird is distributed completely free of registration or deployment fees, even for commercial applications.
Ingres r3 3.0.1	Ingres is available under the Computer Associates Trusted Open Source License. Ingres is distributed completely free of registration or deployment fees, even for commercial applications. [InLIC05]
MaxDB 7.5.0.23	MaxDB is available under the MySQL Dual Licensing Model. - Commercial License [MaCl04] - GNU General Public License [MaOl04] Pricing information for the commercial license is available at [MaPr05].
MySQL 4.1.10	MySQL is available under the MySQL Dual Licensing Model. - Commercial License [MaCl04] - GNU General Public License [MaOl04] Pricing information for the commercial license is available at [MaPr05].
PostgreSQL 8.0.1	PostgreSQL is released under the BSD license. PostgreSQL is distributed completely free of registration or deployment fees, even for commercial applications. [PGLC05]

2.2 Operating Systems Support

This section covers the supported operating systems and builds for 64-bit architectures, if available.

2.2.1 Supported Platforms

The most important platforms supported by the DBMS are shown in the following table. If you are looking for a particular operating system / architecture which is missing, then please make sure that you visit the provided resources.

Firebird 1.5.2	Firebird supports the following operating systems and architectures [FbBo04] [IpFd05]:

Operating system	**Architecture**
Microsoft Windows 95 / 98 / ME	X86_32
Microsoft Windows 2000	X86_32

	Microsoft Windows XP: X86_32 Microsoft Windows Server 2003: X86_32 Linux*: X86_32 FreeBSD: X86_32 Sun Solaris: Sparc, X86_32 HP-UX: PA-RISC Mac OS X: PPC_32 * Linux: NPTL (Native POSIX Thread Library) – specific builds are required for Red Hat 9 or higher, SuSE 8.10 or higher and Mandrake 9.0 or higher. Firebird builds for NTPL-based Linux distributions are available.
Ingres r3 3.0.1	Ingres supports the following operating systems and architectures [InGSfL05] [OSPSI05]: **Operating system** / **Architecture** Microsoft Windows 2000: X86_32 Microsoft Windows XP: X86_32 Microsoft Windows Server 2003: X86_32 Linux: X86_32 Sun Solaris: Sparc, 32-bit and 64-bit HP-UX*: PA-RISC, 32-bit and 64-bit * HP-UX: currently in beta
MaxDB 7.5.0.23	MaxDB supports the following operating systems and architectures [MaDo05]: **Operating system** / **Architecture** Microsoft Windows 2000: X86_32 Microsoft Windows XP: X86_32 Microsoft Windows Server 2003: X86_32, X86_64, IA64 Linux: X86_32, X86_64, IA64, PPC_64

	Sun Solaris: Sparc, 64-bit HP-UX: PA-RISC, IA64 HP Tru64 UNIX: Alpha IBM AIX: PPC_64
MySQL 4.1.10	MySQL supports the following operating systems and architectures [MyRM05] [MySSP05]:

Operating system	**Architecture**
Microsoft Windows 95 / 98 / ME	X86_32
Microsoft Windows 2000	X86_32
Microsoft Windows XP	X86_32
Microsoft Windows Server 2003	X86_32
Linux	X86_32, X86_64, IA64, Alpha, S/390
FreeBSD	X86_32
Sun Solaris	X86_32, Sparc 32-bit and 64-bit
HP-UX	PA-RISC, IA64
HP Tru64 UNIX	Alpha
Mac OS X	PPC_32
IBM AIX	RS6000
QNX	X86_32
Novell Netware	X86_32

PostgreSQL 8.0.1 — PostgreSQL supports the following operating systems and architectures [PGMAN05]:

Operating system	**Architecture**
Microsoft Windows 2000	X86_32
Microsoft Windows XP	X86_32

Microsoft Windows 2003 Server	X86_32
Linux	X86_32, X86_64, ARM, IA64, PA-RISC, MIPS, PPC_32, PPC_64, Sparc, S/390
FreeBSD	X86_32
Sun Solaris	Sparc, X86_32
HP-UX	PA-RISC, IA64
HP Tru64 UNIX	Alpha
Mac OS X	PPC_32
IBM AIX	PPC_32, RS6000

A full list of all supported operating systems and architectures is available here [PGSP05].

2.2.2 64-bit OS Support

32-bit operating systems can address only 4 GB of data in memory. Especially for query intensive tasks with a huge amount of data like in data warehouse and analysis applications, accessed data might exceed the 4 GB limit. If main memory is exhausted, data needs to be swapped onto disk, which is a serious bottleneck for high responsive environments. A 64-bit operating system is capable of addressing memory beyond the 4 GB limit. The availability of versions for 64-bit operating systems is shown below.

Firebird 1.5.2

Firebird isn't currently available for 64-bit OS. [IpFd05]

Ingres r3 3.0.1

Ingres supports the following 64-bit OS [OSPSI05]:

Operating system	**Architecture**
HP-UX*	PA-RISC
Sun Solaris	Sparc

* HP-UX: currently in beta

MaxDB 7.5.0.23

MaxDB supports the following 64-bit OS [MaDo05]:

Operating system	**Architecture**
HP-UX	PA-RISC, IA64
Linux	X86_64, IA64, PPC_64

	Sun Solaris: Sparc HP Tru64 UNIX: Alpha Microsoft Windows Server 2003: X86_64, IA64
MySQL 4.1.10	MySQL supports the following 64-bit OS [MyDL05]:

Operating system	**Architecture**
HP-UX	PA-RISC, IA64
Linux	X86_64, IA64, Alpha
Sun Solaris	Sparc
HP Tru64 UNIX	Alpha

PostgreSQL 8.0.1	PostgreSQL supports the following 64-bit OS [PGMAN05]:

Operating system	**Architecture**
HP-UX	PA-RISC, IA64
Linux	X86_64, IA64, PPC_64
Sun Solaris	Sparc
HP Tru64 UNIX	Alpha

2.3 Limits of the Database

Several per database, table and index limits are listed in this section. Additionally, numerical and string limits are evaluated, which include the maximum precision of fixed decimal types and support for a 64-bit integer data type as well as the maximum size of character data types including character large objects. Another important issue is if the products support Unicode. Several complex query limits are shown at the end of this chapter. The limits were determined by implementing and running limit tests, which are explained in more detail in section 4.

2.3.1 Limits per Database

Especially in the area of very large databases (VLDB), a database administrator and developer must be aware of fundamental limits of a particular DBMS. Limits on a per database include: Maximum database size, maximum number of files per database, maximum file/volume size, maximum number of tables and more. Use the term “Unlimited” with cautious. Of course, physical limits like available disk space still apply. The most important database limits are listed in the table below.

Firebird 1.5.2	Maximum number of tables: 32767 Maximum database size: Theoretical limit of 7 TB Maximum number of files per database: Theoretically 65536 (2^{16}) including shadow files Maximum page size: 16384 bytes Maximum cache buffers: 65536 pages [FbBo04]
Ingres r3 3.0.1	Maximum number of tables: 67108863 Maximum database size: Unlimited Maximum number of files per database: Unlimited Maximum page size: 65536 bytes [InDAG05]
MaxDB 7.5.0.23	Maximum number of tables: Unlimited Maximum database size: 32 TB (with 8 KB page size) Maximum number of files/volumes per database: 64 – 4096, specified by a configuration parameter Maximum file/volume size (data): 518 MB – 8 GB Maximum file/volume size (log): 16 TB [MaTf05]
MySQL 4.1.10	Maximum number of tables: Unlimited Maximum database size: Unlimited Maximum number of files per database: Unlimited The maximum size of a MySQL tablespace is 64 TB (InnoDB) [MyRM05]
PostgreSQL 8.0.1	Maximum number of tables: Unlimited Maximum database size: Unlimited Maximum number of files per database: Not determinable [PGMAN05] [PGLIM05]

2.3.2 Limits per Table

Other limits exist per table. This includes: Maximum number of rows, maximum row size, maximum number of columns/fields and the maximum number of indexes per table. Use the term “Unlimited” with cautious. Of course, physical limits like available disk space still apply. These limits are shown in the following table.

Firebird 1.5.2	Maximum number of rows per table: 4294967296 (2^{32}) Maximum row size per table: 64 KB Maximum number of fields per table: Depends on data types used (see

	Maximum row size) Maximum number of indexes per table: 256 [FbBo04]
Ingres r3 3.0.1	Maximum number of rows per table: 4294690816 (even of a factor 65536 larger when partioning tables) Maximum row size per table: 32 KB Maximum number of fields per table: 1024 Maximum number of indexes per table: Unlimited [InDAG05] [InRS05]
MaxDB 7.5.0.23	Maximum number of rows per table: Limited by database size Maximum row size per table: 8088 bytes Maximum number of fields per table (with KEY): 1024 Maximum number of fields per table (without KEY): 1023 Maximum number of indexes per table: 255 More at [MaTf05]
MySQL 4.1.10	Maximum number of rows per table: Limited by maximum table space (64 TB; InnoDB) Maximum row size per table: 8000 bytes Maximum number of fields per table: 1000 Maximum number of indexes per table: 64 [MyRM05]
PostgreSQL 8.0.1	Maximum number of rows per table: Unlimited Maximum row size per table: 1.6 TB Maximum number of fields per table: 250 – 1600 depending on field types Maximum number of indexes per table: Unlimited The maximum table size of 16 TB does not require large file support from the operating system. Large tables are stored as multiple 1 GB files so file system size limits are not important. The maximum table size and maximum number of fields can be increased if the default block size is increased to 32k. [PGMAN05] [PGLIM05]

2.3.3 Limits per Index

Even limits per index might be important, especially when indexing character data, the maximum size of an index might hit. This limit is given in bytes. When using a 1-byte character set/collation, the number of bytes can be directly mapped to the maximum number of characters. If it is a multi-byte character set/collation, the maximum number of indexed character data is less than the maximum possible index size, depending on how many bytes are needed to store one character.

Firebird 1.5.2	Maximum size: 252 bytes Maximum number of fields: 16 [FbBo04]
Ingres r3 3.0.1	Maximum size: 1003 bytes Maximum number of fields: 32 [InSRG05]
MaxDB 7.5.0.23	Maximum size: 1024 bytes Maximum number of fields: 16 [MaTf05]
MySQL 4.1.10	Maximum size: 1000 bytes Maximum number of fields: 16 [MyRM05]
PostgreSQL 8.0.1	Maximum size: No information available Maximum number of fields: 32 Multi-field indexes are supported by the B-Tree and GiST implementation. [PGMAN05] [PGLIM05]

2.3.4 Numeric Limits

Two questions in this study regarding numerical limits are: 64-bit integer support and the maximum number of precisions for fixed decimal data types like `NUMERIC` and/or `DECIMAL`.

Firebird 1.5.2	64-bit integer values are supported in a Firebird dialect 3 database through the `BIGINT` data type. Fixed-Decimal types are supported via the `NUMERIC(p,s)` / `DECIMAL(p,s)` data types with $1 \leq p \leq 18$, $0 \leq s \leq p$. [FbBo04]
Ingres r3 3.0.1	`SMALLINT`, `INTEGER` and 64-bit integer values are supported. The maximum range of precision of fixed point values of type `DECIMAL` is 31. The range of `FLOAT` values is processor dependent, whereas the precision of the mantissa is up to 16 digits. [InSRG05]

MaxDB 7.5.0.23	The data type `INT[EGER]` is equivalent to `FIXED(10.0)`. In contrast to data type `FIXED(10.0)` however, only values between -2147483648 and 2147483647 are permitted for the data type `INT[EGER]`. There is no `BIGINT` keyword for defining a 64-bit integer, but the `FIXED(p)` data type can be used with a precision p up to 38 digits. The data type `FIXED (p,s)` defines a fixed point number (fixed_point_literal). A column is defined that has a fixed point number with precision p and s number of decimal places ($0 < p \leq 38$, $s \leq p$). `NUMERIC` and `DECIMAL` are permitted in a column definition and are mapped to the `FIXED` data type. [MaLi05]
MySQL 4.1.10	64-bit integer values are supported through the `BIGINT` data type, but arithmetics are always done as signed `BIGINT` and can therefore cause rounding errors. Floating point values are supported through `FLOAT(p)` where p is in the range of 25 to 53. [MyRM05]
PostgreSQL 8.0.1	Variable length numerics are supported through the numeric datatype. The limit in precision is 1000 digits. Within this range, there is no limit in the size of the scale. The numeric type is less efficient than the more constricted floating point types. [PGMAN05]

2.3.5 String Limits

This section evaluates the maximum size of conventional character data types like `[N][VAR]CHAR`, and the availability of character large object (`CLOB`) data types and their maximum size.

Firebird 1.5.2	The maximal length of a CHAR field is 32767 bytes and 32765 bytes for a `VARCHAR` field, because Firebird adds a 2-byte size element to each `VARCHAR` item. Firebird also supports a `BLOB` data type with `SUB_TYPE 1`, equivalent to `CLOB` and `MEMO` types implemented by some other DBMS. The maximum `BLOB` size depends on the database page size used: 2 KB page size ⇒ 512 MB 4 KB page size ⇒ 4 GB 8 KB page size ⇒ 32 GB 16 KB page size ⇒ 256 GB [FbBo04]
Ingres r3 3.0.1	`CHAR` and `VARCHAR` are limited to 32000 bytes. Ingres internally converts `VARCHAR` types to `CHAR` types in the on-disk representation. `LONG VARCHAR` types can be up to 2 GB in size. [InSRG05]

MaxDB 7.5.0.23	The data type `[VAR]CHAR` defines an alphanumeric column. Specifying the length attribute n is optional. If no other length attribute is specified, then n=1. `[VAR]CHAR [(n)]`: $0 < n \leq 8000$ `[VAR]CHAR [(n)] UNICODE`: $0 < n \leq 4000$ MaxDB also supports a `LONG VARCHAR` data type with a maximum size of 2 GB of character data. [MaLi05]
MySQL 4.1.10	`CHAR` and `VARCHAR` are limited to 255 bytes. To store larger text, the datatype `TEXT` can be used. Several different sized `TEXT` dataypes are available, whereas `LONGTEXT` can be up to 4 GB large. `TEXT` data types do have a constraint, that no `DEFAULT` values can be set. The exact amount of characters that can be stored in string datatypes depend on the character set used. [MyRM05]
PostgreSQL 8.0.1	The longest character string that can be stored is about 1 GB. If long strings with no specific upper limit should be stored, it is recommended to use the text or character varying datatype without a length specifier rather than specifying an explicit limit. [PGMAN05]

2.3.6 Unicode Support

For specific application domains, Unicode support in a DBMS represents an essential requirement. Multilingual texts, worldwide data exchange eliminating code page conversion problems and Internet based applications are examples where Unicode plays an important role. The following table shows whether a product supports Unicode characters or not, and which Unicode implementation is applicable. More information on the consecutively used terms UTF-8, UTF-16, UTF-32, UCS-2 and the difference between Unicode and the ISO 10646 standard, which defines the Universal Character Set (UCS), can be found in [UnIs05]. More general information on Unicode can be found in [UnCo05].

Firebird 1.5.2	Firebird has a special character set called `UNICODE_FSS` to store Unicode characters. Developers need to know that it is effectively a UTF-8 implementation. Users need to know that it can be used to store UTF-16 but not UTF-32 characters (that would take up to 6 bytes per character). No collation sequence other than the default binary one is available. [FbBo04]
Ingres r3 3.0.1	USC-2 is the only supported Unicode character set and there is no coercion between Unicode data types and non-Unicode data types such as `CHAR` and `VARCHAR`. [InSRG05] [InMG05]
MaxDB 7.5.0.23	The MaxDB database system supports Unicode in accordance with ISO 10646. Database instances can use Unicode for the database catalog and application data. The data for this is stored internally in the UTF-16/UCS-2 format. [MaLi05]

MySQL 4.1.10	MySQL supports UCS-2 and UTF-8 character sets, but the UTF-8 support is limited to a maximum of three bytes per character. Larger byte-sequences are not supported. UCS-2 cannot yet be used as a character set in MySQL client tools. Character sets can be defined per column, table and database. [MyRM05]
PostgreSQL 8.0.1	Several single-byte character sets and some multi-byte character sets are supported. UTF-8 is the only supported Unicode character set. PostgreSQL supports automatic character set conversion between server and client for certain character sets. Unlike the database, not all APIs support all the listed character sets. [PGMAN05]

2.3.7 Complex Query Limits

Three limit tests were implemented that are relevant for complex queries. These are the maximum number of members in an `IN` list, the maximum number of joinable tables and the maximum number of `AND/OR` operators in the `WHERE` clause of a `SELECT` statement. More information on how the various tests are implemented can be found in section 4.

Firebird 1.5.2	Maximum size of a SQL statement: 64 KB Maximum number of members in an `IN` list: 1499 Maximum number of joinable tables: 255 Maximum number of logical operators in `WHERE` clause: `AND`: 255; `OR`: No limitation found (test run has been stopped by the user at 1593); `RANDOM`: No limitation found (test run has been stopped by the user at 1636). [FbBo04] [LiTe05]
Ingres r3 3.0.1	Maximum size of a SQL statement: Configurable (even ≥ 64 KB) Maximum number of members in an `IN` list: No limitation found (test run has been stopped by the user at 2676) Maximum number of joinable tables: 126 Maximum number of logical operators in `WHERE` clause: `AND`: 1023; `OR`: 1617 (server crashed); `RANDOM`: No limitation found (test run has been stopped by the user at 1708) [LiTe05]

MaxDB 7.5.0.23	Maximum size of a SQL statement: ≥ 16 KB (Default value 64 KB, specified by a system variable) Maximum number of members in an `IN` list: 2041 using the default value for the parameter PACKET_SIZE. Maximum number of joinable tables: 64 Maximum number of join conditions in a `WHERE` clause of a `SELECT` statement: 128 Maximum number of logical operators in `WHERE` clause: `AND`, `OR`, `RANDOM`: 511 using the default PACKET_SIZE parameter value. [MaTf05] [LiTe05]
MySQL 4.1.10	Maximum size of a SQL statement: 1 GB, 16 MB is default. This size is dependent on the max_allowed_packet parameter. Maximum number of members in an `IN` list: No limitation found (test run has been stopped by the user at 27042). Maximum number of joinable tables: 61 Maximum number of logical operators in `WHERE` clause: `AND`: No limitation found (test run has been stopped by the user at 5474) `OR`: No limitation found (test run has been stopped by the user at 5354) `RANDOM`: No limitation found (test run has been stopped by the user at 6772) [LiTe05]
PostgreSQL 8.0.1	Maximum size of a SQL statement: No information available Maximum number of members in an `IN` list: 10917 using the default "max_stack_depth" parameter value. Maximum number of joinable tables: No limitation found (test run has been stopped by the user at 152 due to slow query execution) Maximum number of logical operators in `WHERE` clause: `AND`: No limitation found (test run has been stopped by the user at 1006) `OR`: No limitation found (test run has been stopped by the user at 5008) `RANDOM`: No limitation found (test run has been stopped by the user at 5012) [LiTe05]

2.4 Database Interfaces

This section covers different aspects for interfacing to the DBMS. First, we will have a look on what SQL standards the products in general support. Second, the available programming language interfaces will be listed, including native APIs and higher-level language drivers. Evaluating the support for database links and gateways will conclude this section.

2.4.1 SQL-Standards

The main question here is which SQL standards are support in general, and whether the SQL-92-compliant `JOIN` syntax is supported or not.

Firebird 1.5.2	Firebird's SQL language adheres closely to the SQL-92 standards (ISO/IEC 9075:1992) at entry level. Firebird 1.5 introduced a number of features in accordance with the more recently released SQL-99 standard. Although FirebirdSQL follows standard closely, there are small differences. [FbBo04] Firebird supports the `INNER` and `OUTER` join syntax, for instance.
Ingres r3 3.0.1	The SQL-92 standard (ISO/IEC 9075:1992) is supported at entry level. SQL-92 standard behaviour can be switched off for compatibility with older versions of Ingres. Inner and all three types of outer joins are supported. [InSRG05]
MaxDB 7.5.0.23	You can operate the MaxDB database system in one of the following SQL modes: - INTERNAL: Database system-internal definition - ANSI: ANSI standard according to ANSI X3.135-1992, Entry SQL - DB2: Definition of DB2 Version 4 - ORACLE: Definition of ORACLE7 MaxDB does not support the `INNER` and `OUTER` join syntax. An outer join is represented by the operator (+). [MaLi05]
MySQL 4.1.10	Several differences between MySQL and standard SQL are listed. A warning is given, that “Concatenating many `RIGHT JOINS` or combining `LEFT` and `RIGHT` join in the same query may not give a correct answer”. Unfortunately “many” is not further detailed. Full outer joins are not supported. [MyRM05]
PostgreSQL 8.0.1	PostgreSQL supports a subset of both the SQL-92 and SQL-99 standards. It gives detailed explanations of what is supported and what is not. All Outer join types are supported. [PGMAN05]

2.4.2 Programming Language Interfaces

The available programming language interfaces are an important requirement for database developers to access databases via several programming languages and integrated development environments. In general, a DBMS makes its feature set available through a native API (application programming interface). High-level drivers either expose the client API in an easy to use object model or they completely re-implement the communication protocol between the client and the server (e.g. JDBC Type 4 driver). The available programming language interfaces are shown below.

Firebird 1.5.2	The "native" interface to Firebird to the client library is through C functions and parameter structures exposed by the API. The C header file ibase.h is distributed with Firebird in the /include directory. This header file can be used when writing C programs that use the client library, but it is also a useful reference when developing interfaces for the library from other languages. There are several "high-level" drivers for Firebird available: JDBC,

	ODBC, .NET Provider, an object-oriented framework for C++ development called "IBPP", component sets for Delphi, Kylix and Borland C++ Builder and drivers for PHP, Python and Perl. [FbBo04]
Ingres r3 3.0.1	Drivers are available for JDBC, ODBC and .NET. OpenAPI libraries can be used as an alternative to using embedded SQL which is available for C, Ada, Cobol, Fortran, Pascal and PL1. The actual support of each of these ESQL libraries is dependant on the platform used. [InSRG05] [InGSfL05]
MaxDB 7.5.0.23	MaxDB is open to many programming languages via open interfaces which separate database and application logic for better scalability. For developing MaxDB, the MaxDB ODBC Driver, SQL Database Connectivity (SQLDBC), JDBC Driver, Perl and Python modules and a MaxDB PHP extension, which provides access to the MySQL MaxDB databases using PHP, are available. Third Party Programming Interfaces: Support for OLE DB, ADO, DAO, RDO and .NET through ODBC. [MaLi05]
MySQL 4.1.10	There are APIs available for C, PHP, Perl, C++, Python, Tcl. The steps that have to be taken to make clients thread safe are listed. Drivers are available for JDBC, ODBC and .NET. [MyRM05]
PostgreSQL 8.0.1	Several different client interfaces are available. Among them are ODBC, JDBC and ECPG, an embedded SQL interface in C. A Tcl Binding Libary is available too. There are two ODBC drivers available, PsqlODBC and OpenLink ODBC. There are special steps necessary to make libpq reentrant and thread-safe. Nevertheless, the restriction that two threads aren't allowed to manipulate the same database connection object at the same time still applies. The .Net Data Provider for PostgreSQL is currently under development and in beta status. [PGFAQ05] [PGMAN05] [PGJDBC05] [PGNPG05] [PGCON05]

2.4.3 Database Links

A database link is a database/schema object in a database, which enables access to another, potentially separate and geographically dispersed database inside a single SQL statement, stored procedure or trigger.

Firebird 1.5.2	Firebird doesn't support database links.
Ingres r3 3.0.1	Ingres doesn't support database links.

MaxDB 7.5.0.23	MaxDB doesn't support database links.
MySQL 4.1.10	MySQL doesn't support database links.
PostgreSQL 8.0.1	It is possible to select from and manipulate data in different PostgreSQL databases through the contributed dblink feature. Dblinks are not transparent to the user, they act as function calls operating on rows. [PGDBL05]

2.4.4 Gateways

The term "gateway" is understood as a possibly pluggable mechanism to access databases of different DBMS products.

Firebird 1.5.2	Firebird doesn't support gateways to other DBMS.
Ingres r3 3.0.1	Ingres doesn't support gateways to other DBMS.
MaxDB 7.5.0.23	MaxDB doesn't support gateways to other DBMS.
MySQL 4.1.10	MySQL doesn't support gateways to other DBMS.
PostgreSQL 8.0.1	PostgreSQL doesn't support gateways to other DBMS.

2.5 Programming Features

The most important programming features for database developers are covered in this section. The following features are included: Transactions, isolation levels, prepared statements, bind variables, stored procedures with bind variables, indicator host variables, array inserts / array fetches, row-level locks, timeout functionality, error code expressiveness, available date/time literals, string to date/time conversion, numeric conversion, stored procedure and trigger support.

2.5.1 Transactions

One of the most crucial features in a DBMS is support for transactions. A transaction is a single atomic entity, which ensures that each operation in context of a transaction either succeeds or fails. Typically, this is an important requirement for financial applications but it might be important for other application domains as well. Enterprise-level transactions are often referred to as being ACID-compliant. The term ACID is simply an abbreviation for Atomicity, Consistency, Isolation and Durability.

- Atomicity: A transaction is an atomic entity, which will either entirely succeed or fail.
- Consistency: A transaction should transform the database from one consistent state into another.
- Isolation: A transaction must not influence other transactions in any way.
- Durability: Committed transactions should remain durable, even in case of a system failure.

Several DBMS also support nested transactions and so-called savepoints. Savepoints are simply "bookmarks" in context of a running transaction, which can be set using a savepoint label/name. Data changes can be rolled back to a particular savepoint without quitting the currently running transaction entirely.

Firebird 1.5.2	Firebird supports ACID-compliant transactions including savepoints. [FbBo04]
Ingres r3 3.0.1	Ingres supports ACID-compliant transactions including savepoints. [InSRG05]
MaxDB 7.5.0.23	MaxDB supports ACID-compliant transactions including sub-transactions and/or savepoints. [MaLi05]
MySQL 4.1.10	MySQL supports ACID-compliant transactions including savepoints in InnoDB tables. MyISAM tables do not support transactions. [MyRM05]
PostgreSQL 8.0.1	PostgreSQL supports ACID-compliant transactions including savepoints. [PGMAN05]

2.5.2 Transaction Isolation Levels

A transaction isolation level defines the interaction and visibility of work performed by simultaneously running transactions. The SQL standard defines four transaction isolation levels:

- Read Uncommitted: A transaction sees changes done by uncommitted transactions.
- Read Committed: A transaction sees only data committed before the statement has been executed.
- Repeatable Read: A transaction sees during its lifetime only data committed before the transaction has been started.
- Serializable: This is the strictest isolation level, which enforces transaction serialization. Data accessed in the context of a serializable transaction cannot be accessed by any other transaction.

A critical discussion on the ANSI SQL definitions of transaction isolation levels is available here [IsLe95].

The supported transaction isolation levels are shown below.

Firebird 1.5.2	Isolation levels can be `READ COMMITTED`, `SNAPSHOT`, or `SNAPSHOT TABLE STABILITY`. Within `READ COMMITTED`, two sub-levels are available: `RECORD_VERSION` and `NO_RECORD_VERSION`. `READ UNCOMMITTED` is not supported in Firebird at all. `READ COMMITTED` conforms to the standard. At the two deeper levels (`RECORD_VERSION`, `NO_RECORD_VERSION`), the nature of the Multi-Generational-Architecture (MGA) prevails over the two-phase locking limitations implied by the standard. Mapping to the standard governance of `REPEATABLE READ` and `SERIALIZABLE` is not possible. The default isolation level in Firebird is `SNAPSHOT`. [FbBo04] The difference between `SNAPSHOT` and the ANSI `REPEATABLE READ` isolation level is discussed here [IsLe95].

Ingres r3 3.0.1	All four isolation levels are available in Ingres. [InSRG05]
MaxDB 7.5.0.23	All four isolation levels are available in MaxDB. The default isolation level depends on the SQL mode: - ANSI: `SERIALIZABLE` - All others: `READ COMMITTED` [MaLi05]
MySQL 4.1.10	All four isolation levels are supported in InnoDB tables, the only table type which supports transations at all. The default isolation level in MySQL is `REPEATABLE READ`. Isolation levels can be set for all connections, for a single session or all new incoming connections. [MyRM05]
PostgreSQL 8.0.1	PostgreSQL offers the `READ COMMITTED` and `SERIALIZABLE` isolation levels. `READ COMMITTED` is the default isolation level in PostgreSQL. The `READ UNCOMMITTED` and `REPEATABLE READ` isolation levels are not supported in PostgreSQL. [PGMAN05]

2.5.3 Prepared Statements

Support for prepared statements in a DBMS is an important feature when executing the same SQL statement again and again, because using a prepared statement will normally reduce execution time noticeably. When a client requests a prepared statement, the SQL statement will be sent to the server where it gets syntactically analyzed, compiled and optimized. As a result, the client gets a pre-compiled statement back. This means, when a prepared statement gets executed, the server doesn't need to analyze, compile and optimize the statement again. The server executes the pre-compiled statement. Usually, prepared statements will be used for SQL statements that take parameters. Thus, you can use the same SQL statement with different parameter values each time the prepared statement gets executed. This results in an efficient way to execute the same SQL statement with different values used in batched data manipulation operations, for instance. Prepared statements are accompanied by bind variables.

Firebird 1.5.2	Prepared statements are supported. A "non-parametrized" query uses constants in expressions for search conditions. Data access interfaces that implement the Firebird API have the capability to process the constants in search conditions as "replaceable parameters". The API allows a statement to be submitted to the server as a kind of template that represents these parameters as placeholders. The client request asks for the statement be "prepared" – by obtaining syntax and metadata validation – without actually executing it. [FbBo04] A single prepared statement cannot be used by multiple concurrent clients.
Ingres r3 3.0.1	Prepared statements are supported. The scope of a prepared statement is the owning transaction. [InSRG05]

MaxDB 7.5.0.23	Prepared statements are supported. MaxDB even can use shared SQL to avoid multiple parsing and multiple storage of parse information (such as execution plans and descriptions of result sets). When shared SQL is activated, all parsed SQL statements and related information are stored in a global cache shared by all users, and can be used again if necessary. You can activate shared SQL with the database parameter SHARED_SQL. [MaLi05]
MySQL 4.1.10	Prepared statements are supported. The scope of a prepared statement is the client session within which it is created. Other clients cannot use the prepared statement. [MyRM05]
PostgreSQL 8.0.1	Prepared statements are supported. Statements can be prepared before execution, but prepared statements are only stored in and for the duration of the current database session. A single prepared statement cannot be used by multiple concurrent clients. [PGMAN05]

2.5.4 Bind Variables

A bind variable is a mechanism to bind a value to a (host) variable or to a parameter in a prepared statement. Support for bind variables is a crucial requirement for a DBMS supporting prepared statements (see section 2.5.3).

Firebird 1.5.2	Bind variables are supported. [FbBo04]
Ingres r3 3.0.1	Bind variables are supported. [InSRG05]
MaxDB 7.5.0.23	Bind variables are supported. [MaLi05]
MySQL 4.1.10	Bind variables are supported. [MyRM05]
PostgreSQL 8.0.1	Bind variables are supported in the so-called “extended query” protocol. [PGMAN05]

2.5.5 Stored Procedures with Bind Variables

Prepared statements aren’t necessarily simple SQL statements only. It might be possible, that even stored procedures can get prepared, input parameters are filled with parameter values and the stored procedures gets executed. Any scalar result value might be available through output parameters as well. The following table outlines bind variables support for stored procedures.

Firebird 1.5.2	Stored procedures can be used with parameters/bind variables. [FbBo04]

Ingres r3 3.0.1	Procedures with bind variables are not supported as the statement to call stored procedures cannot be prepared. [InSRG05]
MaxDB 7.5.0.23	Stored procedures can be used with parameter/bind variables. [MaLi05]
MySQL 4.1.10	MySQL does not support bind variables in procedures as it does not support stored procedures. [MyRM05]
PostgreSQL 8.0.1	Stored procedures can be used with parameter/bind variables. [PGMAN05]

2.5.6 Indicator Host Variables

Indicator host variables are necessary to handle `NULL` in an ESQL (Embedded SQL) programm correctly. Usually, the indicator host variable is set to a particular value, if the returned field value is `NULL`, which means an unknown value/state. The used value for indicating `NULL` might be different in different DBMS.

Firebird 1.5.2	Indicator host variables are supported in ESQL. [IbEs00]
Ingres r3 3.0.1	Indicator host variables are supported in ESQL. [InSRG05]
MaxDB 7.5.0.23	Indicator host variables are supported in ESQL. [MaLi05]
MySQL 4.1.10	Indicator host variables are not supported, as there is no ESQL available. [MyRM05]
PostgreSQL 8.0.1	Indicator host variables are supported in ESQL. [PGMAN05]

2.5.7 Array Inserts / Array Fetches

An array insert or fetch is a mechanism to insert or fetch a number of rows with one database call from the client application using only one network round-trip.

Firebird 1.5.2	Array inserts and fetches are not supported.
Ingres r3 3.0.1	Array inserts and fetches are not supported.
MaxDB 7.5.0.23	Array inserts and fetches are supported. [MaDc05]
MySQL 4.1.10	Array inserts and fetches are not supported.
PostgreSQL 8.0.1	Array inserts and fetches are not supported.

2.5.8 Row-Level Locks

DBMS provide various lock types to ensure data integrity in a multi-user environment. Typical lock types are table, page and row-level locks. The latter is the most fain-grained lock type, and provides maximum concurrent data access, because only updated rows are locked for other transactions, and not the entire table or data page.

In contrast to the classic locking model in DBMS, another approach is the so-called MVCC (Multi-Version Concurrency Control) model. This model overcomes the problems found in environments which have to serve as OLTP (Online Transaction Processing) and OLAP (Online Analytical Processing) systems with mixed update operations and long running read transactions by avoiding unneccesary locking. The MVCC model can be summarized as: “Writers don’t block readers and readers don’t block writers”.

Firebird 1.5.2	Row-level locks are supported. Firebird also supports the MVCC model. [FbBo04]
Ingres r3 3.0.1	Row-level locks are supported. [InDAG05] Ingres doesn’t support the MVCC model.
MaxDB 7.5.0.23	Row-level locks are supported. An optimistic lock must be requested explicitly using a `LOCK` statement. If an exclusive lock already exists for a database object, an optimistic lock can be set (lock conflict). If an optimistic lock exists for a database object, other users can continue to set exclusive, shared, or additional optimistic locks. [MaLi05] MaxDB doesn’t support the MVCC model.
MySQL 4.1.10	Row-level locks are supported for InnoDB tables only. Table-level locking is available for ISAM, MyISAM, and MEMORY (HEAP) tables. Page-level locking is available for BDB tables. InnoDB tables also supports the MVCC model. [MyRM05]
PostgreSQL 8.0.1	Row-level locks are supported. PostgreSQL also supports the MVCC model. [PGMAN05]

2.5.9 Timeout Functionality

For long running queries it may be useful to be able to either kill a currently running statement, or to set a maximum duration until a statement gets cancelled by the server automatically. Latter is usually implemented via a timeout parameter on a per server/database/instance/statement basis. The capability to specify a timeout parameter is listed below.

Firebird 1.5.2	Firebird doesn't support something like a timeout parameter, which will terminate a long-running statement after a specific duration.

Ingres r3 3.0.1	Timeout functionality for long running SQL Statements is supported and customizable. [InSRG05]
MaxDB 7.5.0.23	MaxDB doesn't have a timeout parameter for a particular statement, but it has a REQUEST_TIMEOUT parameter which is defined as follows [MaLi05]: Maximum wait time for a lock to be released (in s); for all database sessions, restricts waiting for a lock to be released by another user. If a lock request cannot be fulfilled in the time defined here, a message is sent to the waiting database session. The changes that may have been made within the transaction earlier are rolled back. In MaxDB, a particular statement can be cancelled via the cancel <taskindex> command in XCONS. [MaLi05]
MySQL 4.1.10	There is no timeout functionality in MySQL for long running SQL Statements. [MyRM05]
PostgreSQL 8.0.1	A timeout value can be set for long running statements. [PGMAN05]

2.5.10 Error Code Expressiveness

Executing a syntactically wrong SQL statement or constraint violations are only two examples where the database engine will produce an error message. There are a number of other situations where the engine will return an error either to the calling application or a server-side code module like a stored procedure or trigger. From a database developer point of view, it's important that unique error codes and error messages are returned from the database engine, to handle any database related error messages properly.

Firebird 1.5.2	There is a broad range of error codes/messages available. [FbBo04] Does not follow the SQL standard's convention for `SQLSTATES` codes.
Ingres r3 3.0.1	There is a broad range of error codes/messages available. [InSRG05] Follows the SQL standard's convention for `SQLSTATES` codes.
MaxDB 7.5.0.23	There is a broad range of error codes/messages available. [MaLi05] Does not follow the SQL standard's convention for `SQLSTATES` codes.
MySQL 4.1.10	There is a broad range of error codes/messages available. [MyRM05] Does not follow the SQL standard's convention for `SQLSTATES` codes.
PostgreSQL 8.0.1	There is a broad range of error codes/messages available. [PGMAN05] Follows the SQL standard's convention for `SQLSTATES` codes.

2.5.11 Date/Time Literals

A date/time literal is a stream of characters representing a valid date/time format. A valid date literal, for example, is 'YYYY-MM-DD' (YYYY = four digit year; MM = two digit month; DD = two digit day).

The following table only gives a hint, if there is support for date/time literals at all. Further information can be found in the provided references.

Firebird 1.5.2	Firebird supports a broad range of date/time literal formats. [FbBo04]
Ingres r3 3.0.1	Several standard date/time literal formats are supported according to the currently activated language settings which can be changed within the session. [InSRG05]
MaxDB 7.5.0.23	MaxDB supports the following date/time literal formats: EUR | INTERNAL | ISO | JIS | USA The ISO date and time format is used by ODBC and JDBC applications and cannot be replaced by a different date and time format. The DATE_TIME_FORMAT parameter defines the system default for the date and time format. Each user can change the date and time format for a session by making corresponding entries in the database tool (such as in the Loader using standard_date_mask) or by making corresponding entries in SQL statements (datetimeformat). [MaLi05]
MySQL 4.1.10	Several standard date/time literal formats are supported according to the currently activated language settings. Many more date and time literal combinations can be built by hand by explicitly specifying the format of the literal. [MyRM05]
PostgreSQL 8.0.1	PostgreSQL supports a broad range of date/time literal formats. [PGMAN05]

2.5.12 Conversion from String to Date/Time

In certain situations, a string, compliant to a date/time literal needs to be converted to a real date/time data type. This might be important for date/time arithmetics for instance. The capability of the database engine to convert a string to a date/time and whether there is an Oracle-like `TO_DATE` function or not, is evaluated in the table below.

Firebird 1.5.2	Firebird allows casting a valid date/time literal to a date/time data type. [FbBo04] Firebird doesn't have a built-in function like `TO_DATE(...)` in Oracle, which allows to specify a user-defined date/time format.
Ingres r3 3.0.1	Ingres allows casting a valid date/time literal to a date/time data type according to current language settings, but there is no possibility to explicitly specify how to convert a given literal. [InSRG05]
MaxDB 7.5.0.23	MaxDB implicitly cast a date/time literal to a date/time type through the available date/time literals. [MaLi05] MaxDB doesn't have a built-in function like `TO_DATE(...)` in Oracle, which allows to specify a user-defined date/time format.

MySQL 4.1.10	MySQL allows casting a valid date/time literal to a date/time data type according to current language settings, but there is no possibility to explicitly specify how to convert. [MyRM05]
PostgreSQL 8.0.1	Date and time input is accepted in almost any reasonable format, including ISO 8601, SQL-compatible, traditional POSTGRES, and others. The default datestyle can be chosen from a list of styles. Additionally PostgreSQL supports a wide range of formating options for conversion of date literals to date types and vice versa via the `TO_DATE(…)` and `TO_TEXT(…)` functions. [PGMAN05]

2.5.13 Conversion from String to Numeric

Similar to the requirement above, it has been evaluated how numeric literals can be converted to a numeric data type. Usually, a numeric literal can be “casted” to a numeric data type, either by an implicit (= by the database engine itself) or explicit (= by the user) cast. The existence of an Oracle-like `TO_NUMBER` function has been verified as well.

Firebird 1.5.2	Firebird allows casting a valid numeric literal to a numerical data type. [FbBo04] Firebird uses "." as decimal separator. There is no way to change that. For display purposes, changing the "." to "," might be easy in the client application, or by using a third-party UDF library. Firebird doesn't have a built-in function like `TO_NUMBER(...)` in Oracle with user-defined formatting literals.
Ingres r3 3.0.1	Ingres allows casting a valid numeric literal to a numeric data type according to current language settings, but there is no possibility to explicitly specify how to convert a given literal. [InSRG05]
MaxDB 7.5.0.23	`NUM(a)` is a conversion function that converts the expression “a” to a numeric value. NUM can be applied to character strings (data type `CHAR` or `VARCHAR`) with the code attribute `ASCII`, `UNICODE`, or `BYTE`, to date values, time values, timestamp values, and to numeric and Boolean values (`BOOLEAN`). [MaLi05] MaxDB doesn't have a built-in function like `TO_NUMBER(...)` in Oracle with user-defined formatting literals.
MySQL 4.1.10	MySQL allows casting a valid numeric literal to a numeric data type according to current language settings, but there is no possibility to explicitly specify how to convert. [MyRM05]
PostgreSQL 8.0.1	PostgreSQL supports a wide range of formating options for conversion of numeric literals to a numeric data type and vice versa via the `TO_NUMBER(…)` and `TO_TEXT(…)` functions. [PGMAN05]

2.5.14 Stored Procedure Support

Support for stored procedures and triggers (see below) is an important requirement for enterprise-ready DBMS. A stored procedure is a code module, which can be executed and run on the server. The programming language for writing serverside code modules is in most cases SQL, with extensions for local variable declarations, control-flow statements, cursor declarations for navigating through result sets, input and output parameters, context variables for accessing old and new field values (triggers only), and more. The main benefits of using stored procedures are:

- Reduce network traffic
- Centralize business rules
- Reuse code
- Enhance maintainability

Stored procedure support is evaluated in the table below.

Firebird 1.5.2	Firebird supports stored procedures. The procedural language for writing stored procedures is called PSQL. Stored Procedures in Firebird can be categorized into "Selectable" and "Executable" stored procedures, depending on the result set a stored procedures returns. [FbBo04]
Ingres r3 3.0.1	Stored procedures are supported with SQL as programming language plus local variable declarations, control flow statements and status statements. The syntax is validated at runtime. [InSRG05]
MaxDB 7.5.0.23	MaxDB supports stored procedures. The database system provides a language (special SQL syntax that has been extended to include variables, control structures, and troubleshooting measures) that allows a SQL access layer to be formulated on the server side. This special SQL syntax can be used to define database procedures. [MaLi05]
MySQL 4.1.10	Stored procedures are not supported in MySQL. [MyRM05]
PostgreSQL 8.0.1	Stored procedures are supported through user-defined functions which can be written in SQL, PL/Tcl, PL/pgSQL and C. Function overloading is possible. User-defined functions written in C are compiled into dynamically loadable objects (shared libraries) and are loaded by the server on demand. [PGMAN05]

2.5.15 Trigger Support

Triggers are similar to stored procedures as mentioned above, except that they aren't executed by a client application or another server-side code module directly, but automatically in response to a data manipulation operation like delete, insert or update. Triggers usually don't have input and output parameters. Trigger implementations of DBMS varies in different aspects. For example:

- Row-level versus statement-level triggers: Row-level triggers fire for each record, whereas a statement-level trigger fires only once for a data manipulation operation, independent of how many records have been deleted, inserted or updated.
- Phase: Common phases are: `BEFORE`, `INSTEAD OF` or `AFTER` triggers.
- Timing/event: These are the data manipulation operations delete, insert and update.
- One versus multiple triggers per phase, timing/event for a particular table
- Nested triggers: Is it possible that triggers will fire in response of triggers on other tables?

Trigger support for all participants is covered in the following table.

Firebird 1.5.2	Firebird supports triggers. Firebird allows multiple row-level trigger modules for any phase (`BEFORE`, `AFTER`) / event (`DELETE`, `INSERT`, `UPDATE`) combination. There is probably some practical limit, but it is safe to say you can create as many as you need, using whole numbers between 0 and 32767. The default is sequence number ("`POSITION`") is zero. Firebird supports updating rows in other tables which may invoke other triggers. But, before considering using a trigger to update other rows in the same table, look carefully at the effect of setting off a cycle of nested acitivity. If a trigger performs an action that causes it to fire again, or it fires another trigger that performs an action that causes it to fire, an infinite loop results. For this reason, it is important to ensure that a trigger's actions never cause the trigger to launch itself, even indirectly. [FbBo04]
Ingres r3 3.0.1	Triggers are supported through Ingres Rules. The only timing allowed is after delete, insert or update statements on up to 1024 specified columns. A qualification rule can also be specified and trigger nesting is supported up to 20 levels by default which can be increased by setting a configuration parameter. [InSRG05]
MaxDB 7.5.0.23	MaxDB supports triggers. MaxDB allows one row-level `AFTER` trigger module per event (`DELETE`, `INSERT`, `UPDATE`) and table. A trigger can call further triggers implicitly. Within an regular `UPDATE` statement, one can use the `IGNORE TRIGGER` option so that the trigger event `UPDATE` is ignored. The database system provides a language (special SQL syntax that has been extended to include variables, control structures, and troubleshooting measures) that allows an SQL access layer to be formulated on the server side. This special SQL syntax can be used to define triggers. [MaLi05]
MySQL 4.1.10	Triggers are not supported in MySQL. [MyRM05]

PostgreSQL 8.0.1	Triggers can be written in SQL, PL/Tcl, PL/pgSQL and C. `INSERT`, `UPDATE`, and `DELETE` triggers are supported per row or per statement. The functionality of `INSTEAD OF` triggers can also be implemented. Cascading triggers are supported to an unkown level. Infinite recursions of triggers firing are not avoided by PostgreSQL. [PGMAN05]

2.6 Text Searching Capabilities

This section covers two requirements in the area of text searching. These are a built-in full-text search facility and the possibility to do a soundex search.

2.6.1 Full-Text Search

A full-text search is hereby meant as a mechanism to index, search, and analyze large pieces of textual data stored in the database by using SQL. The feature set, syntax and implementation of a full-text search facility may vary in different DBMS. The table below only evaluates the availability of a built-in full-text search, without going into detail how the implementation looks like.

Firebird 1.5.2	Firebird does not have a built-in full-text search facility. The comparison predicator `CONTAINING` allows to search for a given sequence of characters in a character large object, but without using an index, because a field based on an character large object data type in Firebird can't be indexed. [FbBo04] Different third-party products allow you to integrate a FTS (Full-Text-Search) Engine into a database, by using a mix of User-Defined-Functions and implemented triggers.
Ingres r3 3.0.1	Ingres does not have a built-in full-text search facility.
MaxDB 7.5.0.23	MaxDB does not have a built-in full-text search facility. [MaTq05]
MySQL 4.1.10	MySQL does support full-text searching, but only on MyISAM type tables. These limitations can lead to problems, as MyISAM type tables do not support transactions! All three string datatypes `CHAR`, `VARCHAR` and `TEXT` can be indexed and a relevance value is returned for every retrieved row. [MyRM05]
PostgreSQL 8.0.1	PostgreSQL doesn't include full-text indexing as part of the core distribution, but the PostgreSQL community has contributed the TSearch2 module which is listed in the official core documentation as a contribution. Full indexing is only available on columns of the datatype `tsvector` and has it's own operator. The SQL 'like' operator is not supported. [PGTS2G05] [PGMAN05]

2.6.2 Soundex Search

Soundex is a phonetic algorithm which takes an English word and returns a hash of it. This hash value is intended to be the same for words that sound similar. The availability of a soundex function is evaluated in this section.

Firebird 1.5.2	Firebird doesn't have a built-in `SOUNDEX` function, but there are different UDF (User-Defined-Function) libaries, which do have such a function implemented that can be declared and used in a database. [IpSf05]
Ingres r3 3.0.1	Soundex Search is supported through the included `SOUNDEX(x)` function that can be used on any character data type. [InSRG05]
MaxDB 7.5.0.23	MaxDB has a built-in function `SOUNDEX(x)`. `SOUNDEX(x)` is a string function that converts a character string x (string specification) to a format that is generated by the soundex algorithm. Only expressions that have an alphanumeric value as a result are allowed as a string specification (string spec). [MaLi05]
MySQL 4.1.10	Soundex Search is supported through the included `SOUNDEX(x)` Function. The implementation of the MySQL soundex function is a bit different compared to other implementations, because it returns arbitrarily long strings instead of the standard four characters long strings. [MyRM05]
PostgreSQL 8.0.1	A Soundex function is available, but only through the contributed packages. It is not part of the PostgreSQL core. [PGMAN05]

2.7 Performance of the Database

Another crucial requirement for a DBMS is performance. This section covers some performance related requirements from a theoretical point of view. This includes: Used data structures for table and index storage, scalability and SMP (Symmetric Multiprocessing) support, load balancing capabilities, multi-volume and large memory support. No practical performance tests have been made.

2.7.1 Table Structures

Data structures used for table storage are shown below.

Firebird 1.5.2	The data structure in Firebird used for table storage can't be summarized with well-known data structures. The used table structure in Firebird is described in [FbId04] in more detail.
Ingres r3 3.0.1	Ingres allows tables to be organized as Heap, Hash, Isam and B-Tree table types. Each of them is available as a compressed version too. [InDAG05]

MaxDB 7.5.0.23	To store data so that it is structured, the database system uses B* trees. Data Access Using a B* tree is more efficient than other methods such as sequential scanning. Undo information is not stored in B* trees, nor are C++ objects that are made persistent in liveCache database instances; the database system uses page chains. The database system creates B* trees for the following database objects: - One B* tree for each table (primary data) - One or more additional B* trees for tables with `LONG` columns (primary data) - An additional B* tree for each index of a table (secondary data) A B* tree consists of a root level, one or more B* tree index levels and a leaf level. Each level consists of one or more pages. [MaLi05]
MySQL 4.1.10	An InnoDB table is index clustered. This means, a table gets stored as a B-Tree index, where data of a table row is stored in the tree nodes. [MyRM05]
PostgreSQL 8.0.1	PostgreSQL supports HEAP tables. [PGMAN05]

2.7.2 Index Structures

Data structures used for index storage are shown below.

Firebird 1.5.2	Firebird builds binary trees (B-Tree) for indexes. [FbBo04]
Ingres r3 3.0.1	Indexes in Ingres can be of the types B-Tree, ISAM, Hash and R-Tree. ISAM is the default index type. Except R-Trees each of the index types is also available as a compressed version. Multi-segment indexes are supported with up to 32 columns, but there are no expression indexes available. [InSRG05]
MaxDB 7.5.0.23	The database system creates a B* tree for every index for which a base table is defined. [MaLi05]
MySQL 4.1.10	Indexes on InnoDB tables are B-Tree indexes. In memory tables are indexed by hash indexes. There is no expression index available. [MyRM05]
PostgreSQL 8.0.1	PostgreSQL provides several index types: B-Tree, R-Tree, GiST, and Hash. An index definition may specify an operator class for each column of an index. Partial indexes built over a subset of a table are supported. The subset is defined by a conditional expression. [PGMAN05]

2.7.3 Scalability / SMP Support

This section outlines scalability and SMP (Symmetric Multiprocessing) support in general. We didn't run any practical tests to measure how good a particular DBMS scales under load, when adding additional hardware. A good definition of the scalability term can be found in [WiSc05]. Additionally, the capability to run a single statement in parallel (= parallel query execution) has been evaluated too.

Firebird 1.5.2	Firebird 1.5.x comes in three different architectures. Classic Server, SuperServer and Embedded Server. [FbBo04] Classic server uses one server process per connection. When a client attempts to connect to a Firebird database, an instance of the fb_inet_server executable is initiated and remains dedicated to that client connection for the duration of the connection. When the client detaches from the database, the server process instance ends. Classic Server supports SMP systems. [FbBo04] SuperServer runs as a single instance of the fbserver executable. fbserver is started once, by a system boot script or by the system administrator, and stays running, waiting for connection requests. The process is terminated by an explicit shutdown. SuperServer currently doesn't support SMP systems. [FbBo04] Embedded Server can use only the "Windows local" access method and supports one and only one connection to each local database. The embedded client can connect to multiple local databases exclusively and access them through the embedded server. The embedded client can also connect as a regular network client to databases on other servers. Embedded Server is only available for Windows. It also doesn't support SMP systems, as it is based on the SuperServer architecture. [FbBo04] Firebird isn't capable to run a single statement in parallel. [FbDn04]
Ingres r3 3.0.1	Ingres is offered with a Grid Option that enables clustering of Ingres databases. SMP-support by operating system threads is offered for Linux on Intel, Alpha, S/390 and IA64 Itanium platforms. Parallel Query execution is possible with a customizable number of CPUs involved. [InMG05]
MaxDB 7.5.0.23	A database instance uses several operating system threads. The database system differentiates between the following types of threads: - User Kernel Threads (UKTs) process client requests. They are the main consumers of the processor time. A database instance generally has several user kernel threads. - Special threads provide services for the user kernel threads that are processed asynchronously to the user kernel threads. These include connection requests, reading or writing to data volumes and backups. Special threads execute all tasks that access external resources or that can be blocked. SQL statements are processed in the user kernel threads in the form of tasks. In general, several tasks share a user kernel thread. The database system controls the sequence in which the tasks run in a user kernel thread, and can therefore avoid conflicts with accesses to

	resources. To distribute tasks optimally to the user kernel threads, the database system performs load balancing. In a user kernel thread, a maximum of one task can use the processor, that is, be running. All other tasks can either run or are waiting for an external event, such as the end of an I/O operation and a corresponding message so that they can run again. The change between the tasks is known as task switching. Task switching in a user kernel thread always takes place cooperatively, that is, it is triggered by the task that is currently running. MaxDB supports SMP environments. To make optimum use of computers with multiple processors, the database system supports configurable tasking. The aim of this tasking is to support the largest possible number of database sessions with a minimum number of operating system threads. You can configure the tasking with the two general database parameters `MAXUSERTASKS` and `MAXCPU`. [MaLi05] MaxDB can't run a single SQL statement in parallel. [MaTq05]
MySQL 4.1.10	SMP is supported by using OS threads when using popular platforms. As MySQL connections are being executed in only one thread, queries are not executed in parallel on multiple CPUs. [MyRM05] [MyCL05]
PostgreSQL 8.0.1	PostgreSQL takes advantage of multi processor architectures as the operating system is able to spread the processes of different connections across the available CPUs. PostgreSQL connections are not multi-threading able. Each database connection uses only one CPU at any time. [PGMAN05]

2.7.4 Load Balancing

The capability to spread workload between computers in a clustered environment is hereby called load balancing. Whether the participants support load balancing in a clustered environment or not, is shown below.

Firebird 1.5.2	Firebird doesn't have a built-in load balancing facility, that means, one request can't be processed by two or more Firebird instances/installations on different machines. For reporting and data warehousing services, a redundant database could be maintained either by using a third-party replication solution, or by a hand-made solution using triggers and applying logged operations on the backup database.
Ingres r3 3.0.1	Load balancing happens in Ingres when using the Grid Option but there are no details given how load balancing is implemented. Ingres databases can be either used in Cluster- or in T2PC-mode, not in both. Automatic node failover is available on Solaris and Windows but not on Linux. [InROUG05] [InGSfL05] [InRS05]

MaxDB 7.5.0.23	MaxDB doesn't support load balancing in an clustered environment. [MaTq05] But MaxDB uses load balancing to distribute tasks in an optimum way to the available user kernel threads. [MaLi05]
MySQL 4.1.10	Load balancing happens in MySQL clusters when full table scans are divided into parallel partial table scans on different nodes and when the nodes to handle the transactions are switched in a round robin manner. [MyRM05] [MyCL05]
PostgreSQL 8.0.1	Load balancing between two nodes can be built up with the "pgpool" add-on. Select queries are distributed among the master and slave server in random manner. Other operations are being executed against the master and replicated to the slave. [PGPOOL05] [PGMAN05]

2.7.5 Multi-volume Support

A logical database may consist of several physical database files or data and log files. These are two separately stored entities like swapping and paging areas used by the DBMS. Especially when different stored entities are involved to complete an operation, these entities might be stored on different disks to allow parallel I/O operations to achieve better performance.

Firebird 1.5.2	Firebird doesn't have a transaction log to recover from a database crash. This is due to the multi-generational architecture of Firebird. This means that multiple versions of data rows are stored directly on the data pages. With multi-file databases, you can avoid confining databases to the size of a single disk if the system does not support spanning a single, huge file across multiple disks. There will be no problems installing a RAID array and distributing a multi-file Firebird database across several disks on any supported platform. Note: All files must be located on disks that are under the direct physical control of the Firebird server's host machine. [FbBo04]
Ingres r3 3.0.1	Transaction logs and the database are separated in Ingres. A concept similar to tablespaces named locations is available in Ingres. A Setup with four or more disks is recommended to spread the workload. [InGSfL05]
MaxDB 7.5.0.23	MaxDB has multi-volume support to use different disks for data, log volumes and swap, paging areas. [MaLi05]
MySQL 4.1.10	Data files for table and index data and binary log files are supported in MySQL. [MyRM05]
PostgreSQL 8.0.1	Tablespaces, data files and binary log files are supported in PostgreSQL. [PGMAN05]

2.7.6 Large Memory Support

Support for 64-bit operating systems and addressable memory beyond the 4 GB limit has been already discussed in section 2.2.2. Here, we will evaluate large memory support for 32-bit systems.

Firebird 1.5.2	More than 2 GB on 32-bit platforms. [FbDn04]
Ingres r3 3.0.1	More than 2 GB on 32-bit platforms. [InMG05]
MaxDB 7.5.0.23	More than 2 GB on 32-bit platforms. [MaTq05]
MySQL 4.1.10	Large Memory Support on 32-bit platforms is not assured.
PostgreSQL 8.0.1	Large Memory Support on 32-bit platforms is not assured.

2.8 Distributed Transactions

A distributed transaction is a database transaction that must be synchronized among multiple participating databases which are distributed among different physical locations. A common algorithm for ensuring correct completion of a distributed transaction is the two-phase commit. [WiDt05]

2.8.1 Two-Phase Commit

The two-phase commit (2PC) is a well-known mechanism to ensure that a distributed transaction either succeeds or fails entirely. There are two phases involved to complete a distributed transaction properly [TpCp00]:

- Phase 1: A coordinator process is started (usually at the site where the transaction is initialized), writes a begin commit record in its log, sends a prepare message to the participants, and enters the wait state. This message also contains a unique transaction id (TID), which is also in all further messages involved in a 2PC. When a participant receives a prepare message, it checks if it can commit the transaction. If it can, the participant writes a ready record in its log, sends a vote_commit message to the coordinator, and enters the ready state. Otherwise, the participants decides to unilaterally abort the transaction--it writes an abort record in the log and sends a vote_abort message to the coordinator. It enters the abort state and can forget about the transaction.
- Phase 2: After the coordinator has received votes from all participants it decides whether to commit or abort according to the global commit rule, and writes this decision in the log. If the decision is to commit, it sends a global_commit message to all sites. Otherwise, it sends a global_abort message to all sites that voted to commit. Finally, it writes an end of transaction record in its log. The participants finish the transaction according to the decision and write the result in their logs.

2PC support is discussed below.

Firebird 1.5.2	Firebird supports the 2PC protocol, which is exposed via the Firebird API. [FbBo04]

Ingres r3 3.0.1	Ingres SQL provides two statements that support a 2PC: `PREPARE TO COMMIT` and `CONNECT`. [InSRG05]
MaxDB 7.5.0.23	MaxDB doesn't support the 2PC protocol. [MaTq05]
MySQL 4.1.10	MySQL doesn't support the 2PC protocol. [MyRM05]
PostgreSQL 8.0.1	PostgreSQL doesn't support the 2PC protocol. [PGFAQ05]

2.8.2 XA-Protocol Support

The protocol used for a 2PC can be either a proprietary DBMS vendor implementation or in compliance with the so-called XA-protocol. The XA-protocol is used by a coordinating transaction manager, to span a single transaction over two or more resources. A resource in this context doesn't necessarily mean a database. Other resources involved in a single global transation via the XA-protocol might be a JMS (Java Messaging Service) connection or a JCA (Java Connector Architecture) resource, for instance.

Firebird 1.5.2	Firebird does support the 2PC protocol. The protocol used isn't the XA protocol though. Firebird could be slotted into an XA-compliant DTP environment as illustrated in [FbBo04] on p.90.
Ingres r3 3.0.1	The XA-protocol is supported, but several XA options are not available. Among them are: Multithreaded database client libraries and asynchronous XA operations. Libraries for providing entry points for the transaction monitors Encina, CICS and Tuxedo are included. [InDTPUG05] [InRS05]
MaxDB 7.5.0.23	MaxDB doesn't support the XA-protocol. [MaTq05]
MySQL 4.1.10	MySQL doesn't support the XA-protocol. [MyRM05]
PostgreSQL 8.0.1	PostgreSQL doesn't support the XA-protocol. [PGFAQ05] [PGJXA05]

2.8.3 Consistent Multi-Database-Recovery

During a 2PC: If network interruption or a disk crash makes one or more databases unavailable, causing the two-phase commit to fail during the second phase, sub-transactions left behind will be in a transient state, flagged as neither committed nor rolled back. [FbBo04]

A consistent multi-database-recovery shall ensure, that in case of a failure during a 2PC, each participating database will be transfered into a consistent state again.

Firebird 1.5.2	Within each individual database, these sub-transactions that never completed the second phase (became committed or rolled back) are recognized as being in a "limbo" state. Until a limbo transaction is finished (by being committed or rolled back) it remains "interesting" to Firebird, which keeps statistics on unfinished transactions. Recovering a limbo transaction means committing it or rolling it back. The gfix tool can recover limbo transactions and let you deal with them interactively. [FbBo04]
Ingres r3 3.0.1	There is no tool support for consistent multi-database-recovery. [InDAG05]
MaxDB 7.5.0.23	Not relevant, because MaxDB doesn't support distributed transactions. [MaTq05]
MySQL 4.1.10	Not relevant, because MySQL doesn't support distributed transactions. [MyRM05]
PostgreSQL 8.0.1	Not relevant, because PostgreSQL doesn't support distributed transactions. [PGFAQ05]

2.9 Replication

Replication is a mechanism to keep parts or the entire database with another database in-sync, either for redundant environments or as a business requirement, for example for syncing a field worker's database with a centralized company database.

A sophisticated replication facility might be an important requirement for enterprise-level applications, thus replication support is listed in the table below.

Firebird 1.5.2	There is no replication facility in the Firebird core package, but there are different third-party replication solutions for Firebird available. [IpRt05]
Ingres r3 3.0.1	Asynchronous Master-Master (Full Peer) replication is available through the Ingres Replicator Option which builds upon TPC. Ingres Replicator Option interoperates with non-Ingres databases through Advantage Enterprise Access products. [InROUG05]
MaxDB 7.5.0.23	Replication is supported as standby database and hot standby system. [MaLi05] A much more flexible replication solution called "Synchronization Manager" will be available in MaxDB 7.6. [MaRm04]
MySQL 4.1.10	Master-Slave replication is supported. Chained replication servers are also supported. [MyRM05]
PostgreSQL 8.0.1	Replication is possible through the "Slony1" PostgreSQL database replication solution, which is not a part of the core PostgreSQL distribution. Another way to set up replication is through the erserver project. The "eRServer" project delivers a trigger-based single-master/multi-slave asynchronous replication system.

	[PGSLONY05] [PGERSRV05]

2.10 High Availability Features

This section covers some basic high availability requirements necessary for a smooth 24x7 operation of the DBMS. This includes: online backup, online reorganization and cluster support.

2.10.1 Online Backup

This requirement defines the capability of creating backups of data and/or log files without requiring a database or database server shutdown. An online backup facility is an extremely important requirement for database or system administrators to include database files in an available server backup plan, because database files shouldn't be touched by regular backup software while there are active connections to the database in question. Usually, the online backup facility of the DBMS is used to create a database backup file, which can then be archived using third-party backup software.

Firebird 1.5.2	Firebird is capable of creating hot backups via the command-line tool "gbak" or via the so-called Services API. [FbBo04] Firebird 1.5.2 doesn't support incremental backups. This will be available in Firebird 2.0. [FbRm05]
Ingres r3 3.0.1	Ingres is capable of creating hot backups via snapshots of the database, so-called checkpoints and by keeping the changes since the last checkpoint in a journaling file. [InDAG05]
MaxDB 7.5.0.23	MaxDB is capable of creating incremental data and log backups when the database instance is in operational state `ONLINE` or `ADMIN`. [MaLi05]
MySQL 4.1.10	Online Backup in MySQL can be done through the tool Mysqldump which generates a SQL script for recreating the database objects and the data in them. As binary log files are being written too, the combination of restoring the Mysqldump file and the since then generated binary logs is a somewhat basic online backup and recovery solution that is suitable for point-in-time recovery with InnoDB type tables. Another solution is the tool InnoDB Hot Backup, a commercial add-on. InnoDB Hot Backup does not set any locks or disturb the normal database processing. On MySQL Clusters the committed log contains the committed transactions made during the backup so that no transactions get lost during backup. [MyRM05] [MyCL05]
PostgreSQL 8.0.1	Write-ahead logs are saved automatically and continuously. Together with the PostgreSQL point-in-time recovery feature, a continuous backup of the server is possible. Recovery can happen either up to the point of failure or to some transaction in the past. Setting up a Hot Standby System is possible. [PG800RN05] [PGMAN05]

2.10.2 Online Reorganization

Beside the ability to run online backups, another important issue is online reorganization of databases. Reorganization in this context mainly means creating, altering and dropping indexes as well as updating index statistics either automatically by the DBMS or manually by the database administrator/developer without shutting down the database or the entire database server.

Firebird 1.5.2	An index can be created via the `CREATE INDEX` statement at any time. An index can be deactivated/activated via the `ALTER INDEX` statement as long as the index isn't used by a concurrent transaction. An index can be dropped via the `DROP INDEX` statement, as long as the index isn't used by a concurrent transaction. Use the command-line utility "gfix" to perform a variety of housekeeping and recovery tasks. For several tasks you need to have exclusive access to the database. [FbBo04]
Ingres r3 3.0.1	Indexes can be created, dropped and re-built. Creating an index requires an exclusive lock on the table. [InSRG05]
MaxDB 7.5.0.23	An index can be created via the `CREATE INDEX` statement at any time. Enabling/disabling and dropping is only possible if the index isn't in use by a concurrent transaction. The database system recognizes uneven distributions of B* tree structures and evens them out automatically (automatic reorganization). In doing so, the database system moves some entries to new positions and updates the pointers to the pages. Due to this, rebalancing the size of a table is not limited by the page size, but only by the total storage space that is available. Checking database structures is used to find errors that are caused by hardware defects. A database structure check validates the consistency of the individual database structures (such as the consistency of the B* trees), but not the semantics of the data model. You cannot, therefore, find any logical errors with a database structure check.
MySQL 4.1.10	Indexes in MySQL can be created and dropped but they cannot be rebuilt. Beware that `ANALYZE TABLE` on a BDB table may in some cases make the table unusable until you restart the "mysql" deamon. [MyRM05]
PostgreSQL 8.0.1	Some configuration parameters can be changed at runtime for all currently running processes or for a single process. Others require a restart to become effective. Indexes can be added to and removed from tables at any time. The system will update the index when the table is modified, but the `ANALYZE` command has to be run regularly to update statistics for the query planner to allow it to make educated decisions on when to use the index in queries. [PGMAN05]

2.10.3 Cluster Support

In section 2.7.4, we've already discussed load balancing capabilities in a clustered environment. Here, we are asking for support for a failover mechanism in a clustered environment.

Firebird 1.5.2	Firebird doesn't support clustering/failover natively. Clustering might be possible via a clustering middleware solution like C-JDBC. [CJDBC04]
Ingres r3 3.0.1	Clusters are supported in Ingres using the grid option. Ingres clusters are transparent to applications. Special hard- and software is required to run Ingres in cluster mode, among them is a storage area network (SAN) and a cluster file system. Several features are not available in Cluster mode. These features include row-level locking, two-phase commit and replication. An increased number of deadlocks and contention may occur. [InGSfL05]
MaxDB 7.5.0.23	With MaxDB, it is possible to set up a hot-standby system in a clustered environment. The requirements are [MaLi05]: - Database Software. MaxDB database instances from Version 7.5.0.8 - Cluster Solution - Memory Management System MaxDB doesn't support load balancing in an clustered environment though. [MaTq05]
MySQL 4.1.10	Clusters are available in MySQL on several platforms. They function as in-memory databases in a shared-nothing system. Data is replicated synchronously to up to four replicas. If one node fails, the current transaction fails for the application even if all other nodes survive. [MyRM05] [MyCL05]
PostgreSQL 8.0.1	There is an add-on named "pgpool" available that acts as a synchronous replication server rather than a cluster. If one of the PostgreSQL servers goes down, "pgpool" tries to continue the service with the remaining live server. Before coming back to replication mode, the two databases have to be brought back to exactly the same DB contents, which in most cases causes downtime of both servers. Another add-on to PostgreSQL, named "PGCluster" is constituted by Load Balance Server, Cluster DB Server, and Replication Server. With each server's combination, "PGCluster" can build two kinds of systems, a load balancing system and a highly available system. In case of a failure, cluster DBs can be started with a recovery option to come back to a replication system. In order to avoid service disruption during recovery, a cluster must consist of at least three operating databases. [PGPOOL05] [PGCL05] [PGMAN05]

2.11 Ease of Administration

Incidental expenses in an environment using a fully-fledged commercial DBMS like Oracle, DB2 or Microsoft SQL Server are expenses for a full-time database administrator employment. Reducing

licensing costs when moving from a commercial towards an open source DBMS with an available comparable feature set might be one good reason, but another, possibly even better reason will be to decrease ongoing personnel costs by cutting down expenses for a full-time database administrator. This section outlines the ease of administration in general, which includes: system management, database management, performance tuning / profiling tools and security aspects. These areas are discussed very briefly. If you need further information on specific topcis regarding administration, we would like to encourage reading the references given.

2.11.1 System Management

The system management requirement is related to running database servers/instances for example in a data center, and how easy it is to monitor the availability and health of database servers/instances from a central location. This also includes support for enterprise-level management protocols like Simple Network Management Protocol (SNMP).

Firebird 1.5.2	Firebird does not come with visual tools. The Firebird core package includes a set of command-line tools, which can be used to query particular database health information like transaction statistics, in-depth information about indexes and more. There is no built-in way to check if a remote Firebird server is currently up and running, except when trying to connect to a remote database with the command-line tool “isql” for instance. Firebird does not support SNMP or any other management protocol natively.
Ingres r3 3.0.1	“Ingres Visual Manager” captures events that are occurring in the system. The DBA has the possibility to specify which messages must be discarded, which are displayed, and which additionally trigger special alerts. Special alerts include playing sounds and displaying message boxes and forwarding of message texts to the operating system event handling system for further treatment. The Ingres management architecture is compatible with standard management protocols like SNMP, Distributed Management Environment (DME) and Common Management Interface Protocol (CMIP). [InDAG05]
MaxDB 7.5.0.23	The “Database Manager” tool is used to manage database instances. You can use it among other things to create, monitor, back up, and restore database instances. The “Database Manager” GUI has a graphical user interface and can be used only on Microsoft Windows. However, you can use the Database Manager GUI also to manage database instances that are running on a remote computer with a different operating system. There is also a web edition of “Database Manager” as well as a command-line version. MaxDB does not support SNMP or any other management protocol natively. [MaMq05]
MySQL 4.1.10	The tool "MySQL Administrator" enables the DBA, among other administrative tasks, to visually monitor the database health. Aside from other database health informations, the number of open connections, the amount of traffic, and the number of SQL queries is shown. There is no built in mechanism of warning the DBA when certain limits are being reached or the connection to the database is lost.

	MySQL does not support SNMP or any other management protocol natively but SNMP support can be gained through the commercial tool Micro Agent for MySQL. [MyRM05] [MySNMP05]
PostgreSQL 8.0.1	The graphical administration tool "pgAdmin III" has a server status window where the DBA can see a list of open connections along with further detailed information. The server status window is capable of refreshing itself periodically. There is no built in mechanism of warning the DBA when certain limits are being reached or the connection to the database is lost. PostgreSQL does not support SNMP or any other management protocol natively.

2.11.2 Database Management

This section mainly mentions tool support for administrating/managing a database instance.

Firebird 1.5.2	Unlike many other relational database systems, Firebird never has need of a full-time DBA with an armory of algorithms for keeping databases running smoothly. For the most part, well-kept Firebird databases just "keep on keeping on. Firebird also does not require the intense and constant reconfiguration that many other heavy-duty RDBMSs do. [FbBo04] The Firebird kit does not come with a GUI admin tool. It does have a set of command-line tools, executable programs that are located in the ./bin directory of your Firebird installation. [FbBo04] A list of full-featured third-party graphical administration and development tools for Firebird is available at [IpAt05].
Ingres r3 3.0.1	There are a lot of operating system shell tools available for several administrative tasks along with several graphical tools. [InSAG05]
MaxDB 7.5.0.23	Some characteristics of MaxDB underline it's ease of administration [MaRm04]: - Few configuration parameters - No size estimates for individual database objects - Automatic space management - Automatic balancing of disk I/O - No permanent attention required MaxDB comes with a set of (graphical) tools to administer a MaxDB database instance. [MaLi05]
MySQL 4.1.10	A graphical administration tools is available as well as a graphical query tools. Several commercial administration tools are available. [MyRM05]

PostgreSQL 8.0.1	A graphical administration tool is available with an included query tool. A web based tool built upon PHP is available as well. Commercial administration tools can also be aquired. [PGHP05] [PGFAQ05]

2.11.3 Performance Tuning / Profiling Tools

Especially in a 24x7 environment, it might be useful for a database administrator to see what's going on in a database and how performance can be improved. Database developers are mainly interested in getting the execution plan for certain SQL statements, either in a textual or graphical representation.

Firebird 1.5.2	Firebird has a few configuration parameters to tweak performance. For example: `CPUAffinityMask`, `SortMemBlockSize`, `SortMemUpperLimit`. To process a SELECT statement or a search condition, the Firebird engine processes a statement using a set of internal algortihms known as the query optimizer. Each time the statement is prepared for execution, the optimizer computes a retrieval plan. The query plan provides a kind of road map which tells the engine the least costly route through the required process of searching, sorting, and matching to arrive at the request output. The more efficient a plan the optimizer can construct, the faster the statement will begin returning results. Many graphical database admin and SQL monitor tools provide the capability to inspect the optimizer's plan when a statement is prepared. Firebird's own isql utility provides two interactive commands for viewing plans. Indexes play an important role in the performance of a database. It is important to recognize that they are dynamic structures that, like moving parts in an engine, need to be "cleaned and lubed" from time to time. Indexes can be rebalanced and tuned in a number of ways to restore performance to optimal levels. [FbBo04] Firebird does not have any type of profiling/monitoring tables to keep track of what's going in the database, like which statements are currently running, and so on.
Ingres r3 3.0.1	Fine grained auditing, session- and I/O-tracing is available in Ingres but there is no general database-wide trace mode. [InDAG05] The statistics-based optimizer returns the query execution plan which can be inspected in text mode or with graphical database administration and SQL monitor tools. Lock monitoring tools are available too. Configuration parameters like sort area sizes can not be changed. [InDAG05]
MaxDB 7.5.0.23	To analyze performance during operation, the so-called "Database Analyzer" and other database tools can be used. [MaLi05] The `EXPLAIN` statements can be used to find out which search strategy the optimizer selected for an SQL statement. [MaLi05]

	Some hints for better performance when setting up a new database instance are available here [MaLi05]. The "Database Analyzer" database tool is used to analyze the performance of database instances. It can be used to identify problems in the following areas: - Database configuration (caches, parameters) - Synchronization (locks, critical sections, heap) - Processing of SQL statements (for example, search strategies, indexes, optimization statistics) - Hardware configuration "Database Analyzer" can be used on all operating systems supported by the database system. You can analyze all database instance types and versions of the database software and also access database instances on a remote computer. [MaLi05]
MySQL 4.1.10	MySQL has a set of configuration parameters to tweak performance (e.g. buffer sizes). [MyRM05] Profiling of all code or selected functions is available. [MyRM05]
PostgreSQL 8.0.1	Usage of indexes can be examined by using the `EXPLAIN` command. The optimizer can be provided with information about the tables using the `ANALYZE` command. There are even run-time parameters that can turn off various plan types. It is possible to control the query planner to some extent by using the explicit `JOIN` syntax. PostgreSQL has a set of configuration parameters to tweak performance (e.g. buffer sizes). As a crude profiling instrument performance statistics can be written to the server log for statements, the parser, the planner and the executor. [PGMAN05]

2.11.4 Security

This section discusses some security related aspects of the DBMS in general.

Firebird 1.5.2	The server installation includes a user authentication database for storing definitions of all users that have access to the Firebird Server. A case-sensitive password must be defined for each user and used to gain access to the server. The command-line tool for maintaining the user database is gsec. User authentication is required whenever a remote or local client connects to a Firebird database. Passwords can be up to 32 characters, but only the first eight characters are significant. Hence, for example, the passwords masterkey and masterkeeper are seen by the server as identical. Passwords are case sensitive. Passwords are encrypted by the Firebird API at the client, before they are sent to the server. [FbBo04] Firebird doesn't support other authentication mechanism like Kerberos or LDAP for instance.
Ingres r3 3.0.1	Kerberos is supported as an authentication mechanism. There is no built-in SSL support available. Ingres Installations can be administered in compliance with the National Computer Security Center (NCSC) C2 security standard. [InDAG05] [InCG05]
MaxDB 7.5.0.23	The MaxDB database system differentiates between two types of users [MaLi05]: - Database users access the data in the database instance using SQL statements - DBM operators (Database Manager operators) manage database instances using the Database Manager tool The database system administrator (SYSDBA user) and the first DBM operator have special authorizations. User names can be up to 32 characters. Passwords have a maximal length of 18 (`ASCII` character set) or 9 (`UNICODE` character set) characters. [MaLi05] MaxDB uses a proprietary authentication protocol and doesn't support other authentication mechanism like Kerberos or LDAP for instance. [MaMq05] Passwords aren't sent in a human-readable way from the client to the server. [MaMq05]
MySQL 4.1.10	MySQL uses a proprietary authentication protocol using a user/password information. Additionally, MySQL can check X509 certificate attributes but there is no support for other authentication mechanism like Kerberos or LDAP for instance. SSL-encrypted connections between MySQL clients and servers are supported. [MyRM05]

PostgreSQL 8.0.1	Kerberos is supported. At a given time either Kerberos version 4 or 5 can be used, not both together. SSL, PAM, MD5 and SSH are supported for authentication and encrypting connections. It is possible to assign a password to the database superuser operating system user account to divide security handling between the operating system and the database management system. [PGMAN05]

2.12 Technical Appraisal

This section covers some general aspects of the project like project activity/release cycle, size of development team, support availability and future trends.

2.12.1 Project Activity / Release Cycle

The table below discusses some items related to activity and past release cycles of the project.

Firebird 1.5.2	The release cycle depends on the number of new features and enhancements targeted for inclusion in the next release. The following is a list of release dates: Firebird 1.0.0: March 2002 Firebird 1.0.1: September 2002 Firebird 1.0.2: December 2002 Firebird 1.0.3: June 2003 Firebird 1.5.0: February 2004 Firebird 1.5.1: July 2004 Firebird 1.5.2: December 2004 The Firebird project is being actively developed. CVS check-ins are listed here [FbCi05].
Ingres r3 3.0.1	The r3 3.0.0 release of Ingres is the first open source release. Nearly three years have passed between the last two major releases of Ingres. [InCAH05]
MaxDB 7.5.0.23	Minor releases (e.g. 7.5.0.19 to 7.5.0.23) are released frequently. MaxDB 7.6 is currently (when making the study officially available) in beta. There is no release date for 7.6 available though. [MaCh05] MaxDB is actively developed by SAP. The developer's mailing list shows the activity of the project. [MaDl05]
MySQL 4.1.10	For about one year new minor releases (4.1.x) have been released nearly every month. MySQL 4.1.10 and 5.0 are being developed in parallel, with MySQL 5 having new features like stored procedures, triggers, and many more. The stable MySQL 4.1 branch is for bugfixing only. [MyRM05] [MyNews05]
PostgreSQL 8.0.1	Major releases of PostgreSQL, the last ones having the release numbers 7.3, 7.4 and 8.0, are being released about once a year around christmas time. [PGFAQ05] [PGNEWS05]

2.12.2 Size of Development Team

The following table tries to give an impression on how many people are working on the product. Use these numbers with care, because they were gathered from presentations, check-in statistics and other sources. This information might even vary over time.

Firebird 1.5.2	Core Engine: ~15 .NET Provider: 1 ODBC: 1 JDBC: 1-3 Misc (packaging, documentation, ...): ~5-15 The information is based on following the firebird-checkins list [FbCi05].
Ingres r3 3.0.1	There are no official numbers available on how many people are working on Ingres.
MaxDB 7.5.0.23	There are no concrete numbers on how many SAP employees are working on MaxDB, but [MaRm04]: - SAP continues to own MaxDB - SAP continues to develop MaxDB - SAP continues to support MaxDB - SAP continues to offer MaxDB
MySQL 4.1.10	More than 60 developers are listed by name in the bug database but there is no mention on which part of MySQL they are working on. [MyBugs05]
PostgreSQL 8.0.1	Around 30 developers are actively developing PostgreSQL and around 20 contributing people are listed by name. [PGMAN05]

2.12.3 Support Availability

One reason (beside others) why commercial entities aren't using open source products is the assumed lack of support services required for mission critical applications. Commercial DBMS vendors usually offer superb commercial support including guaranteed response times, for instance. This section discusses available support options for a particular open source DBMS.

Firebird 1.5.2	Commercial support is available through IBPhoenix [IbPh05]. IBPhoenix is a web site and an organization that provides information and services to InterBase® and Firebird developers and users, those who develop applications on InterBase® or Firebird, and those who develop and use the underlying database engine itself. The IBPhoenix team has an unparalleled depth and breadth of experience with InterBase® and Firebird, as developers, as users, as consultants, and in providing accurate, useful answers to questions about either product. Free support is available through community groups, lists and forums active on the Firebird Open Source projects. A list of available groups is available here [IpCg05].

Ingres r3 3.0.1	CA.com offers extensive Ingres support in many countries. Also several courses in administration and application development can be taken in cities around the world. [InCPI05] [InCAH05]
MaxDB 7.5.0.23	Commercial support contract with SAP [MaLi05]: If you are using the database system in an SAP application, then you are given access to the whole of SAP AG's support infrastructure through the support contract with SAP. Commercial support contract with MySQL [MaLi05]: If you are using the database system in other applications (not SAP applications), then you can conclude a support contract with MySQL. Freely available support via the MaxDB mailing list [MaDl05].
MySQL 4.1.10	Free support can be obtained from community groups, lists and forums. Commercial support can be obtained from partners worldwide or directly from MySQL AB. Training, consulting and 24x7 telephone hotline support are available. [MySp05]
PostgreSQL 8.0.1	There are several companies worldwide offering PostgreSQL support and professional services. [PGCON05] [PGCGS05]

2.12.4 Future Trends

This section outlines some features expected to arrive in forthcoming releases. More information is available in the provided references.

Firebird 1.5.2	Firebird 2.0 Alpha 1, with many new features (new index structure, incremental backups, derived tables, `EXECUTE BLOCK` syntax, expression index, ...) has been released in March 2005. Vulcan, an experimental branch based on an early Firebird 2.0 code base, will include fine grained multi-threading, SMP and 64-bit operating system support. Firebird 3.0 is planned to be a merger of Firebird 2.0 + Vulcan + Yaffil (Russian fork based on Firebird 1.0). The entire roadmap is available here [FbRm05].
Ingres r3 3.0.1	HP-UX und OpenVMS ports of Ingres are being developed. [InSC05]
MaxDB 7.5.0.23	MaxDB 7.6 is currently in beta. Planned enhancements include SSL, IPv6 support for client/server communication, improved join performance, user defined extensions (UDE) based on Java, logging of DBM server protocol, schema support, `BEFORE` statement triggers and a so-called Synchronization Manager (a much more flexible replication solution), and more. [MaRm04]
MySQL 4.1.10	MySQL 5.0 is currently in beta. It supports stored procedures, views, cursors, triggers, full outer joins and constraints. [MyRM05]

PostgreSQL 8.0.1	A large todo list is available that contains features in the areas of administration, indexes, commands and others. Optimizations and feature improvements appear in several areas and two-phase commit is listed under the list item of exotic features to be developed. [PGTD05]

3 Feature Matrix

	Firebird 1.5.2	Ingres r3 3.0.1	MaxDB 7.5.0.23	MySQL 4.1.10	PostgreSQL 8.0.1
Licensing					
Applicable License	Original code modules released by Inprise are licensed under the InterBase Public License (IPL). New code modules added to Firebird are licensed under the Initial Developer's Public License (IDPL). Completely free, even for commercial application	Computer Associates Trusted Open Source License. Completely free, even for commercial application	MySQL Dual Licensing Model: - Commercial License, and - Open Source License	MySQL Dual Licensing Model: - Commercial License, and - Open Source License	BSD Licence Model Completely free, even for commercial application
Operating Systems Support					
Supported Platforms	Large list of platforms	Large list of platforms	Large list of platforms	Large list of platforms	Large list of platforms
64-bit OS Support	No official release yet	Solaris on Sparc, HP-UX on PA-RISC (currently in beta)	Large list of platforms	Several platforms	Large list of platforms
Limits of the Database					
Limits per Database	Database size: 7 TB Number of tables: 32767 Files per database: 65536 (2^{16})	Database size: Unlimited Number of tables: 67108863 Files per database: Unlimited	Database size: 32 TB Number of tables: Unlimited Files/volumes per database: 64 – 4096	Database size: Unlimited Number of tables: Unlimited Files per database: Unlimited	Database size: Unlimited Number of tables: Unlimited Files per database: Not determinable
Limits per Table	Number of rows: 4294967296 (2^{32}) Maximum row size: 64 KB Number of fields: Depends on data types used Indexes per table: 256	Number of rows: 4294690816 Maximum row size 32 KB Number of fields: 1024 Indexes per table: Unlimited	Number of rows: Limited by database size Maximum row size: 8088 bytes Number of fields (with KEY): 1024 Number of fields (without KEY): 1023 Indexes per table: 255	Number of rows: Limited by maximum table space (64 TB) Maximum row size: 8000 bytes Number of fields: 1000 Indexes per table: 64	Number of rows: Limited by table size Maximum row size: 1.6 TB Number of fields: 250 – 1600 Indexes per table: Unlimited
Limits per Index	Maximum size: 252 bytes Number of fields: 16	Maximum size: 1003 bytes Number of fields: 32	Maximum size: 1024 bytes Number of fields: 16	Maximum size: 1000 bytes Number of fields: 16	Maximum size: No information available Number of fields: 32
Numeric Limits	Support for 64-bit integer 18 digit precision for `NUMERIC/DECIMAL`	Support for 64-bit integer 53 digit precision for `NUMERIC/DECIMAL`	Support for 64-bit integer 38 digit precision for `NUMERIC/DECIMAL`	64-bit integer only without arithmetics 53 digit floating point precision	1000 digit precision
String Limits	`VARCHAR`: 32765 bytes `CHAR`: 32767 bytes `BLOB SUB_TYPE 1`: Maximum `BLOB` size depends on the used page size: 2 KB page size ⇒ 512 MB	`VARCHAR`: 32000 bytes `CHAR`: 32000 bytes `LONG VARCHAR`: 2 GB	`VARCHAR`: 8000 bytes `CHAR`: 8000 bytes `LONG VARCHAR`: 2 GB	`VARCHAR`: 255 bytes `CHAR`: 255 bytes `LONGTEXT`: 4 GB	`VARCHAR, CHAR, TEXT`: 1 GB

	4 KB ⇒ 4 GB 8 KB ⇒ 32 GB 16 KB ⇒ 256 GB				
Unicode Support	Supported (UTF-8, UTF-16)	Supported (UCS-2)	Supported (UTF-16/UCS-2)	Supported (UTF-8)	Supported (UTF-8)
Complex Query Limits					
Maximum SQL Statement Size	64 KB	Configurable	≥ 16 KB	≤ 1 GB	No information available
Maximum IN List Members	1499	≥ 2676	2041 using the default PACKET_SIZE parameter value	≥ 27042	10917 using the default "max_stack_depth" parameter value
Maximum Number of Joined Tables	255	126	64	61	152 (test run stopped due to slow query execution)
Maximum Logical Operators in WHERE clause	AND: 255 OR: ≥ 1593 RANDOM: ≥ 1593	AND: 1023 OR: 1617 (server crashed) RANDOM: ≥ 1708	AND: 511 OR: 511 RANDOM: 511	AND: ≥ 5354 OR: ≥ 5354 RANDOM: ≥ 5354	AND: ≥ 1006 OR: ≥ 5008 RANDOM: ≥ 5008
Database Interfaces					
SQL-Standards	SQL-92 at entry level. Parts of the SQL-99 standard. INNER/OUTER join SQL-92 syntax supported	SQL-92 at entry level. INNER/OUTER join SQL-92 syntax supported.	Several SQL modes: INTERNAL, ANSI X3.135-1992 Entry SQL, DB2 v4, ORACLE v7 INNER/OUTER join SQL-92 syntax not supported	Several differences between MySQL and standard SQL. INNER/OUTER join SQL-92 syntax supported. No full outer join.	Subset of both the SQL-92 and SQL-99 standards. INNER/OUTER join SQL-92 syntax supported.
Programming Language Interfaces	C/C++, JDBC, ODBC, .NET, OLEDB, OO frameworks for Borland Delphi/Kylix/C++ Builder, PHP, Python and Perl	C/C++, JDBC, ODBC, .NET, Ada, Cobol, Fortran, Pascal and PL1	JDBC, ODBC, SQLDBC, PHP, Python and Perl. OLEDB, .NET via ODBC.	APIs, JDBC, ODBC, .NET, ….	ODBC, JDBC, ECPG, Tcl .Net Data Provider in development
Database Links	Not supported	Not supported	Not supported	Not supported	Supported
Gateways	Not supported	Not supported	Not supported	Not supported	Not supported
Programming Features					
Transactions	ACID-compliant Support for savepoints	ACID-compliant Support for savepoints	ACID-compliant Support for sub-transactions and/or savepoints	ACID-compliant in InnoDB Support for savepoints	ACID-compliant Support for savepoints
Transaction Isolation Levels	Supported: READ COMMITTED SNAPSHOT SNAPSHOT TABLE STABILITY Not supported: READ UNCOMMITTED REPEATABLE READ SERIALIZABLE	Supported: READ UNCOMMITTED READ COMMITTED REPEATABLE READ SERIALIZABLE	Supported: READ UNCOMMITTED READ COMMITTED REPEATABLE READ SERIALIZABLE	Supported: READ UNCOMMITTED READ COMMITTED REPEATABLE READ SERIALIZABLE	Supported: READ COMMITTED SERIALIZABLE Not supported: READ UNCOMMITTED REPEATABLE READ
Prepared Statements	Supported	Supported	Supported	Supported	Supported
Bind Variables	Supported	Supported	Supported	Supported	Supported
Stored Procedures with Bind Variables	Supported	Not supported	Supported	Not supported	Supported
Indicator Host Variables	Supported in ESQL	Supported in ESQL	Supported in ESQL	Not supported	Supported in ESQL
Array Inserts / Array Fetches	Not supported	Not supported	Supported	Not supported	Not supported
Row-Level Locks	Supported MVCC: Supported	Supported MVCC: Not supported	Supported MVCC: Not supported	Supported in InnoDB MVCC: Supported in InnoDB	Supported MVCC: Supported

Timeout Functionality	Not supported	Supported and customizable	Not supported	Not supported	Supported
Error Code Expressiveness	Broad range of error codes/messages available. Does not follow the SQL standard's convention for `SQLSTATES` codes.	Broad range of error codes/messages available. Follows the SQL standard's convention for `SQLSTATES` codes.	Broad range of error codes/messages available. Does not follow the SQL standard's convention for `SQLSTATES` codes.	Broad range of error codes/messages available. Does not follow the SQL standard's convention for `SQLSTATES` codes.	Broad range of error codes/messages available. Follows the SQL standard's convention for `SQLSTATES` codes.
Date/Time Literals	Broad range of date/time literal formats available	Broad range of date/time literal formats available	Support for EUR | INTERNAL | ISO | JIS | USA date/time literals	Supported	Broad range of date/time literal formats available
Conversion from String to Date/Time	Implicit/explicit `CAST` to date/time type supported. No Oracle like `TO_DATE` function.	Implicit/explicit `CAST` to date/time type supported. No Oracle like `TO_DATE` function.	Implicit/explicit `CAST` to date/time type supported. No Oracle like `TO_DATE` function.	Implicit/explicit `CAST` to date/time type supported. No Oracle like `TO_DATE` function.	Implicit/explicit `CAST` to date/time type supported. Oracle like `TO_DATE` function available.
Conversion from String to Numeric	Implicit/explicit `CAST` to numerical data type supported. No Oracle like `TO_NUMBER` function.	Implicit/explicit `CAST` to numerical data type supported. No Oracle like `TO_NUMBER` function.	`NUM(a)` conversion function No Oracle like `TO_NUMBER` function.	Implicit/explicit `CAST` to numerical data type supported. No Oracle like `TO_NUMBER` function.	Implicit/explicit `CAST` to date/time type supported. Oracle like `TO_NUMBER` function available.
Stored Procedure Support	Supported	Supported	Supported	Not supported	Supported
Trigger Support	Supported	Supported	Supported	Not supported	Supported
Text Searching Capabilities					
Full-Text Search	Not supported. Available via third-party solutions	Not supported	Not supported	Only on non-transactional table types	Not supported. Available through add-on module on special data type.
Soundex Search	Not supported. Available via third-party UDFs.	Supported	Supported	Supported	Not supported. Available via add-on module.
Performance of the Database					
Table Structures	No tree data structure	Heap, Hash, Isam and B-Tree	B* tree	Only index clustering	Heap
Index Structures	B-Tree	B-Tree, ISAM, Hash and R-Tree	B* tree	No expression index	B-Tree, R-Tree, GiST, and Hash.
Scalability / SMP Support	SMP support with Classic Server Parts of SQL statements can't be run in parallel	SMP support, Grid Option, Clustering Parts of SQL statements can be run in parallel	SMP support Parts of SQL statements can't be run in parallel	SMP support Parts of SQL statements can't be run in parallel	No SMP support Parts of SQL statements can't be run in parallel
Load Balancing	Not supported	Supported	Not supported	No parallel query, partial table scans in clusters	Supported `SELECT` load balancing between two nodes through add-on module
Multi-volume Support	Supported. Data files and temporary areas. Due to the multi-generational architecture there are no transaction log files	Supported. Transaction logs and the database are separated	Supported. Data, log volumes and swap, paging areas.	Supported. Tablespaces, datafiles, binary log	Supported. Tablespaces, datafiles, binary log
Large Memory Support	More than 2 GB on 32-bit platforms	More than 2 GB on 32-bit platforms	More than 2GB on 32-bit platforms	No information available	No information available
Distributed Transactions					
Two-Phase Commit	Supported	Supported	Not supported	Not supported	Not supported
XA-Protocol Support	Not supported	Partly supported	Not supported	Not supported	Not supported
Consistent Multi-Database-	Supported	Not supported	Not supported	Not supported	Not supported

Recovery					
Replication					
Replication Support	Not supported. Available via third-party solutions	Supported	Partly supported via (hot) standby system	Supported	Not supported. Available via third-party solutions
High Availability Features					
Online Backup	Supported	Supported	Supported	Limited or commercial	Supported
Online Reorganization	No index activating / deactivating, dropping when index is in use by a concurrent transaction	Indexes can be created, dropped and re-built	No index enabling / disabling, dropping when index is in use by a concurrent transaction	No index rebuild. Analyze can cause instabilities	Indexes can be added to and removed from tables at any time.
Cluster Support	Not supported	Supported Transparent to applications, several Ingres features are not available in Cluster mode	Support for clustered Hot-Standby System	Supported	Supported via add-on module
Ease of Administration					
System Management	No visual tools in the core package. Command-line tools to query database health information. No native support for SNMP or any other management protocol.	There is a tool called “Ingres Visual Manager” available. Native support for SNMP, DME and CMIP.	There is a tool called “Database Manager” available. No native support for SNMP or any other management protocol.	There is a tool called “MySQL Administrator” available. No native support for SNMP or any other management protocol.	There is a tool called “pgAdmin III” available. No native support for SNMP or any other management protocol.
Database Management	Set of command-line utilities in the Firebird core package. Several GUI driven third-party tools available (freeware, commercial)	Set of command-line utilities. Several GUI driven tools	GUI tool available	GUI tool available	GUI tool available
Performance Tuning / Profiling Tools	Configuration parameters. Inspection of the optimizer’s plan. Profiling/monitoring tables aren’t available	Fine grained auditing, session- and IO-tracing, query execution plan, Lock monitoring tools	Configuration parameters. Inspection of the optimizer’s plan. Profiling is supported	Configuration parameters and explain plan Profiling is supported.	Configuration parameters. Performance statistics can be written to the server log. Inspection of the optimizer’s plan.
Security	Proprietary authentication protocol using user/password combination. No support for Kerberos, LDAP, ... No built-in SSL support	Kerberos for authentication No built-in SSL support	Proprietary authentication protocol using user/password combination. No support for Kerberos, LDAP, ... No built-in SSL support	Proprietary authentication protocol using user/password combination. No support for Kerberos, LDAP, ... Built-in SSL support	Kerberos for authentication Built-in SSL, PAM, MD5 and SSH support
Technical Appraisal					
Project Activity / Release Cycle	Very active	Activity hard to determine, because Ingres went open-source recently	Very active	Very active	Very active
Size of Development Team	~ 40	No official numbers available	No official numbers available	~ 60	~ 30 – 50
Support Availability	Free and commercial support available.	Free and commercial support available.	Free and commercial support available.	Free and commercial support available.	Free and commercial support available.
Future Trends	Available in the official available Firebird roadmap	HP-UX und Open VMS ports	Planned features for version 7.6 are officially available	Many new features in release 5.0.x	Available in the official available todo list

4 Limit Tests

To determine limits in respect to complex queries, three tests have been implemented and executed against each candidate. The tests are called:

- MAXIN
- MAXOP
- MAXJOIN

4.1 MAXIN Test

The MAXIN test determines the maximum number of possible values in an `IN` <member_list> clause. A MAXIN table is created:

```
CREATE TABLE MAXIN1 (
  MAXIN1_ID INTEGER NOT NULL,
  F1 INTEGER NOT NULL,
  CONSTRAINT PK_MAXIN1 PRIMARY KEY (MAXIN1_ID)
);
```

Sample 1: Limit Tests – MAXIN Table DDL

The following `SELECT` statements get executed against that table with a number of constants in the <member_list> starting with 1 and increasing by 1. For example:

- Iteration 1: `SELECT 1 FROM MAXIN1 WHERE F1 IN (1);`
- Iteration 2: `SELECT 1 FROM MAXIN1 WHERE F1 IN (1, 2);`
- and so on ...

The test run either reached the maximum number of members in an `IN` clause or was stopped due to peformance reasons – a long running query.

4.2 MAXOP Test

The MAXOP test determines the maximum number of possible logical operators `AND` and `OR` in a `WHERE` clause. A MAXOP table is created:

```
CREATE TABLE LOGOP1 (
  LOGOP1_ID INTEGER NOT NULL,
  F1 INTEGER NOT NULL,
  CONSTRAINT PK_LOGOP1 PRIMARY KEY (LOGOP1_ID)
);
```

Sample 2: Limit Tests – MAXOP Table DDL

The following `SELECT` statements get executed against that table with a number of boolean expressions concatenated with `AND`, `OR` and random occurences of `AND`/`OR` starting with 1 and increasing by 1. For example:

- Iteration 1 (`AND`): `SELECT 1 FROM LOGOP1 WHERE F1=1;`
- Iteration 2 (`AND`): `SELECT 1 FROM LOGOP1 WHERE F1=1 AND F1=2;`
- and so on ...
- Iteration 1 (`OR`): `SELECT 1 FROM LOGOP1 WHERE F1=1;`

- Iteration 2 (OR): `SELECT 1 FROM LOGOP1 WHERE F1=1 OR F1=2;`
- and so on ...
- Iteration 1 (RANDOM): `SELECT 1 FROM LOGOP1 WHERE F1=1;`
- Iteration 2 (RANDOM): `SELECT 1 FROM LOGOP1 WHERE F1=1 OR F1=2;`
- Iteration 2 (RANDOM): `SELECT 1 FROM LOGOP1 WHERE F1=1 OR F1=2 AND F1=3;`
- and so on ...

The test run either reached the maximum number of boolean expressions or was stopped due to peformance reasons – a long running query.

4.3 MAXJOIN Test

The MAXJOIN test determines the maximum number of joinable tables. 1000 JOIN<nnnn> tables are created:

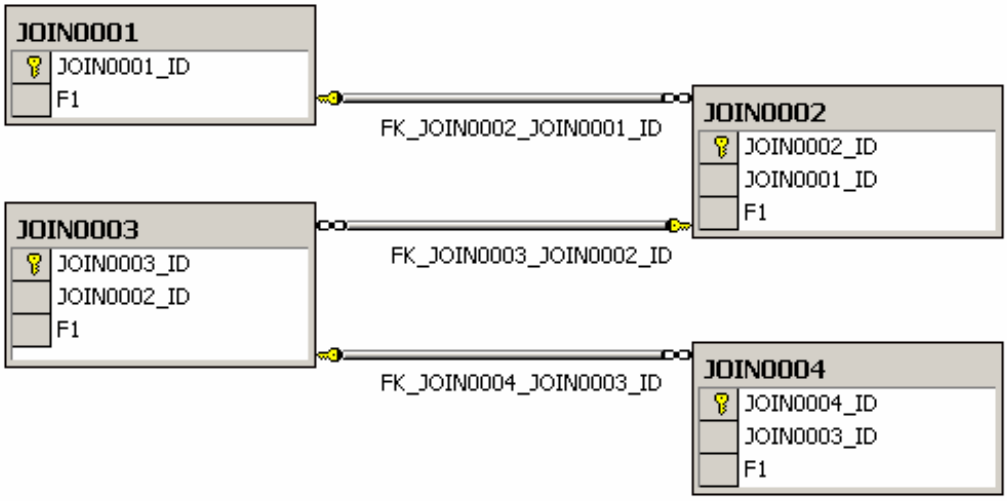
```
CREATE TABLE JOIN0001 (
  JOIN0001_ID INTEGER NOT NULL,
  F1 INTEGER NOT NULL,
  CONSTRAINT PK_JOIN0001 PRIMARY KEY (JOIN0001_ID)
);

CREATE TABLE JOIN0002 (
  JOIN0002_ID INTEGER NOT NULL,
  JOIN0001_ID INTEGER NOT NULL,
  F1 INTEGER NOT NULL,
  CONSTRAINT PK_JOIN0002 PRIMARY KEY (JOIN0002_ID),
  CONSTRAINT FK_JOIN0002_JOIN0001_ID FOREIGN KEY (JOIN0001_ID) REFERENCES JOIN0001
(JOIN0001_ID)
);

CREATE TABLE JOIN0003 (
  JOIN0003_ID INTEGER NOT NULL,
  JOIN0002_ID INTEGER NOT NULL,
  F1 INTEGER NOT NULL,
  CONSTRAINT PK_JOIN0003 PRIMARY KEY (JOIN0003_ID),
  CONSTRAINT FK_JOIN0003_JOIN0002_ID FOREIGN KEY (JOIN0002_ID) REFERENCES JOIN0002
(JOIN0002_ID)
);
CREATE TABLE JOIN0004 (
  JOIN0004_ID INTEGER NOT NULL,
  JOIN0003_ID INTEGER NOT NULL,
  F1 INTEGER NOT NULL,
  CONSTRAINT PK_JOIN0004 PRIMARY KEY (JOIN0004_ID),
  CONSTRAINT FK_JOIN0004_JOIN0003_ID FOREIGN KEY (JOIN0003_ID) REFERENCES JOIN0003
(JOIN0003_ID)
);
...
CREATE TABLE JOIN1000 (
  JOIN1000_ID INTEGER NOT NULL,
  JOIN0999_ID INTEGER NOT NULL,
  F1 INTEGER NOT NULL,
  CONSTRAINT PK_JOIN1000 PRIMARY KEY (JOIN1000_ID),
  CONSTRAINT FK_JOIN1000_JOIN0999_ID FOREIGN KEY (JOIN0999_ID) REFERENCES JOIN0999
(JOIN0999_ID)
);
```

Sample 3: Limit Tests – MAXJOIN Tables DDL

For a better understanding of the relationship between the `JOIN<nnnn>` tables through the `FOREIGN KEY` constraints, Sample 4 illustrates the first four tables in an Entity Relationship Diagram (ERD).



Sample 4: Limit Tests – MAXJOIN Tables – ER diagram

A series of `SELECT` statements is executed against these tables joining them via the fields used in the `FOREIGN KEY` constraint `FK_JOIN<nnnn>` using the SQL-89 join syntax. A test run starts with joining two tables. For example:

```
SELECT 1 FROM JOIN0001,JOIN0002
WHERE
JOIN0001.JOIN0001_ID=JOIN0002.JOIN0001_ID;
```

Sample 5: Limit Tests – MAXIN – Join of two tables

Continuing joining three tables:

```
SELECT 1 FROM JOIN0001,JOIN0002,JOIN0003
WHERE
JOIN0001.JOIN0001_ID=JOIN0002.JOIN0001_ID
AND JOIN0002.JOIN0002_ID=JOIN0003.JOIN0002_ID;
```

Sample 6: Limit Tests – MAXIN – Join of three tables

The test run either reached the maximum number of joinable tables or was stopped due to peformance reasons – a long running query.

5 Conclusion

Firebird 1.5.2	Firebird supports ACID-compliant transactions (incl. 2PC protocol support), stored procedures, triggers, views, cascading referential constraints, generators (aka sequences), online backups, several character sets / collations, multi-processor support in the Classic Server architecture and many other features mentioned in this document or in [FbBo04] [FbWs05]. It lacks in some areas like no built-in full-text search and replication facility. No incremental backup (will be available in Firebird 2.0), built-in clustering support (load balancing and failover), other authentication mechanism and a database profiling facility (planned for Firebird 2.0 as well).
Ingres r3 3.0.1	Ingres shines with its support of XA-compliant distributed transactions, asynchronous master-master replication, cluster and SMP support, parallel query execution, support for standard monitoring protocols and C2 security, Kerberos, stored procedures and triggers. It has a long history as being used inside of CA.com and is now a new competitor in the market of open source database management systems. Ingres could benefit from full-text search and a consistent multi-database-recovery tool. Regrettably Ingres can only be used either in Two Phase Commit or in Cluster mode, not both.
MaxDB 7.5.0.23	MaxDB is an SAP certified, mature relational database management system promising ease of administration, high performance and availability, SMP support, online (incremental) data/log backups and much more. [MaWs05] Nevertheless, it also lacks in some areas. MaxDB doesn't have a built-in full-text search facility. It doesn't support the two-phase commit protocol, which might be an important requirement for enterprise-level applications. Replication will be improved in MaxDB 7.6 with a utility called "Synchronization Manager". Allowing only three triggers (one per phase/event) per table might be a restriction for certain situations as well.
MySQL 4.1.10	With cluster support, replication, SMP support and query caching, MySQL has several features available that make it good for read-intensive tasks with simple SQL statements for a good price. MySQL has set a world record for speed & price/performance in 2004 [MyPe04] [MySpec04] and is known to be used in a large number of successfully projects involving large amounts of data [MyCbI05]. It lacks of the more complex SQL features like full-text search in transaction enabled tables, views, stored procedures, triggers and distributed transactions – most of these issues will be adressed in the next major release however (currently available as a BETA release). Looking at available enterprise features, MySQL seems to be optimized for read intensive tasks like being the backend of a website. [MyRM05]

	[MyCL05]
PostgreSQL 8.0.1	PostgreSQL has very good support for high precision numeric and large binary / character data types among others. No other evaluated database management system offers comparable support for different date and number formats. PostgreSQL supports very large databases, even on operating systems with limited capabilities. Replication, load balancing, multi-processor and cluster support is available. Several authentication and encryption methods are available as well as database links. Its full-text search facility requires the use of a special data type and online backup is a very new feature of PostgreSQL. A significant feature missing in PostgreSQL is distributed transaction support. [PGMAN05] [PGPOOL05] [PGCL05] [PG800RN05] [PG801RN05]

6 Bibliography

6.1 Firebird 1.5.2

[CJDBC04] *C-JDBC: Flexible Database Clustering Middleware, 2005*
http://c-jdbc.objectweb.org/current/doc/C-JDBC_Flexible_Database_Clustering_Middleware.pdf

[FbBo04] *The Firebird Book: A Reference for Database Developers; Helen Borrie; ISBN 1-59059-279-4, 2004*

[FbCi05] *Firebird – CVS check-ins, 2005*
news://news.atkin.com/sourceforge.firebird-checkins

[FbDn04] Thread "Some inside Firebird related questions" in Firebird-devel newsgroup, *2004*
news://news.atkin.com/sourceforge.firebird-devel

[FbId04] *IBPhoenix Publications - Inside Firebird Databases, 2004*
PDF document on the quarterly published IBPhoenix CD.
http://www.ibphoenix.com/main.nfs?a=ibphoenix&page=ibp_cd

[FbRm05] *Firebird – Roadmap, 2005*
http://firebird.sourceforge.net/index.php?op=devel&sub=engine&id=roadmap

[FbWs05] *Firebird Project – Website, 2005*
http://www.firebirdsql.org

[FIDPL04] *Initial Developer's Public License, 2004*
http://www.firebirdsql.org/index.php?op=doc&id=idpl

[FIPl00] *InterBase Public License, 2000*
http://www.firebirdsql.org/index.php?op=doc&id=ipl

[IbEs00] *InterBase 6.0 - Embedded SQL Guide, 2000*
http://www.ibphoenix.com/downloads/60EmbedSQL.zip

[IpAt05] *IBPhoenix Contributed Downloads – Administration Tools, 2005*
http://www.ibphoenix.com/main.nfs?a=ibphoenix&s=1103107587:483980&page=ibp_admin_tools

[IpCg05] *IBPhoenix Community – Community Groups, Lists and Forums, 2005*
http://www.ibphoenix.com/main.nfs?a=ibphoenix&page=ibp_groups

[IpFd05] *IBPhoenix Download – Firebird V1.5.x Downloads, 2005*
http://www.ibphoenix.com/main.nfs?a=ibphoenix&page=ibp_download_15

[IbPh05] *IBPhoenix – Website, 2005*
http://www.ibphoenix.com

[IpRt05] *IBPhoenix Contributed Downloads – Replication Tools, 2005*
http://www.ibphoenix.com/main.nfs?a=ibphoenix&s=1101199972:345924&page=ibp_repl_tools

[IpSf05] IBPhoenix Research – Implementing a Soundex Founction, *2005*
http://www.ibphoenix.com/main.nfs?a=ibphoenix&s=1101957447:392398&page=ibp_howto4

[LiTe05] *Win32 Limittest Utility developed for that study, 2005*

[MoPl05] *Mozilla Public License v.1.1, 2005*
http://www.opensource.org/licenses/mozilla1.1.php

6.2 Ingres r3 3.0.1

[In64OS05] *64-Bit Operating Systems and RDBMS: Optimizing Performance*, *2005*
http://opensource.ca.com/projects/ingres/documents/whitepapers/ingresr364bitwp.pdf

[InCAH05] *CA Homepage*, *2005*
http://www.ca.com/

[InCG05] *Connectivity Guide*, *2005*
http://opensource.ca.com/projects/ingres/documents/product/ingresr3/A001821E.pdf

[InCPl05] *Course Path Ingres*, *2005*
http://ca.com/education/paths/tables/lp_advantage_ingres.htm

[InCS05] *Ingres Customer Successes*, *2005*
http://www3.ca.com/Solutions/CollateralList.asp?CCT=19526&ID=1013

[InDAG05] *Database Administrator Guide*, *2005*
http://opensource.ca.com/projects/ingres/documents/product/ingresr3/A001841E.pdf

[InDS05] *Ingres r3 Data Sheet*, *2005*
http://opensource.ca.com/projects/ingres/documents/whitepapers/ingresr3ds.pdf

[InDTPUG05] *Distributed Transaction Processing User Guide*, *2005*
http://opensource.ca.com/projects/ingres/documents/product/ingresr3/A001951E.pdf

[INFaq05] *Ingres for Linux FAQ*, *2005*
http://opensource.ca.com/projects/ingres/documents/technical/FAQ_Questions_About_Ingres_for_Linux.pdf

[InGSfL05] *Getting Started for Linux*, *2005*
http://opensource.ca.com/projects/ingres/documents/product/ingresr3/A001921E.pdf

[InLIC05] *Computer Associates Trusted Open Source License*, *2005*
http://www3.ca.com/Files/Licensing/ca_trusted_open_source.pdf

[InMG05] *Migration Guide*, *2005*
http://opensource.ca.com/projects/ingres/documents/product/ingresr3/A001931E.pdf

[InROUG05] *Replicator Option User Guide*, *2005*
http://opensource.ca.com/projects/ingres/documents/product/ingresr3/A001881E.pdf

[InRS05] *Release Summary*, *2005*
http://opensource.ca.com/projects/ingres/documents/product/ingresr3/A001811E.pdf

[InSAG05] *System Administrator Guide*, *2005*
http://opensource.ca.com/projects/ingres/documents/product/ingresr3/A001831E.pdf

[InSC05] *Ingres Support Connect*, *2005*
http://supportconnectw.ca.com/public/ingres/ingressupp.asp

[InSRG05] *SQL Reference Guide*, *2005*
http://opensource.ca.com/projects/ingres/documents/product/ingresr3/A001851E.pdf

[OSPSI05] *CA Open Source Project Site - Ingres*, *2005*
http://opensource.ca.com/projects/ingres

6.3 MaxDB 7.5.0.23

[MaCh05] *MaxDB Change History, 2005*
http://dev.mysql.com/doc/maxdb/changes/

[MaCl04] *MySQL Commercial License, 2004*
http://www.mysql.com/company/legal/licensing/commercial-license.html

[MaDc05] *MaxDB – SQL Database Connectivity (SQLDBC), 2005*
http://dev.mysql.com/doc/maxdb/sqldbc/en/index.html

[MaDl05] *MaxDB Development – Developers' Mailing List, 2005*
http://lists.mysql.com/maxdb

[MaDo05] *Download MaxDB Standard Version, 2005*
http://dev.mysql.com/downloads/maxdb/7.5.00.html

[MaLi05] *MySQL MaxDB Library, 2005*
http://dev.mysql.com/doc/maxdb/en/default.htm

[MaMq05] *MySQL Lists: maxdb - Thread: "MaxDB 7.5 - Some more questions", 2005*
http://lists.mysql.com/maxdb/24462

[MaOl04] *MySQL Open Source LIcense, 2004*
http://www.mysql.com/company/legal/licensing/opensource-license.html

[MaPr05] *MySQL Commercial License – Pricing, 2005*
https://shop.mysql.com/

[MaRm04] *MaxDB Presentations and Whitepapers – The MaxDB Roadmap to Technology and Low TCO, 2004*
http://dev.mysql.com/doc/maxdb/pdf/teched04_CI103.pdf

[MaTf05] *MaxDB Features – Technical Specification, 2005*
http://www.mysql.com/products/maxdb/features.html#Specification

[MaTq05] *MySQL Lists: maxdb - Thread: "MaxDB 7.5 - Some technical questions", 2005*
http://lists.mysql.com/maxdb/24403

[MaWs05] *MaxDB – Website, 2005*
http://www.mysql.com/products/maxdb/

6.4 MySQL 4.1.10

[MyBugs05] *MySQL Bugs: Statistics, 2005*
http://bugs.mysql.com/bugstats.php

[MyCbI05] *MySQL Customers by Industry, 2005*
http://www.mysql.com/customers/

[MyDL05] *MySQL Downloads, 2005*
http://dev.mysql.com/downloads/mysql/4.1.html

[MyCL05] *MySQL Cluster, 2005*
http://www.mysql.com/products/cluster/

[MyNews05] *MySQL in the News, 2005*
http://www.mysql.com/news-and-events/in-the-news/

[MyPe04] *MySQL Helps Set New World Records for Speed & Price/Performance in Independent Benchmarks, 2004*
http://www.mysql.com/news-and-events/press-release/release_2004_27.html

[MyRM05] *MySQL Reference Manual, 2005*
http://dev.mysql.com/doc/mysql/en/index.html

[MySNMP05] *Micro Agent for MySQL, 2005*
http://solutions.mysql.com/software/?item=468

[MySp05] *MySQL Support, 2005*
http://dev.mysql.com/support/

[MySpec04] *SPECjAppServer2002 Result,* Aug-*2004*
http://www.spec.org/osg/jAppServer2002/results/res2004q3/jAppServer2002-20040817-00028.html

[MySSP05] *MySQL Server Supported Platforms, 2005*
http://www.mysql.com/support/supportedplatforms.html

6.5 PostgreSQL 8.0.1

[PG800RN05]	*Release Notes Release 8.0.0, 2005* http://www.postgresql.org/docs/8.0/static/release-8-0.html
[PG801RN05]	*Release Notes Release 8.0.1, 2005* http://www.postgresql.org/docs/8.0/static/release.html #RELEASE-8-0-1
[PGCGS05]	*Cybertec Geschwinde & Schoenig, 2005* http://www.postgresql.at/
[PGCL05]	*PGCluster, 2005* http://www.csra.co.jp/~mitani/jpug/pgcluster/en/
[PGCON05]	*PostgreSQL Consulting Companies, 2005* http://techdocs.postgresql.org/companies.php
[PGCON05]	*Solutions based on the PostgreSQL database, 2005* http://gborg.postgresql.org/
[PGDBL05]	*dblink, 2005* http://developer.postgresql.org/docs/pgsql/contrib/dblink/doc/
[PGDL05]	*Download Site, 2005* ftp://ftp.postgresql.org/pub
[PGERSRV05]	*The erserver Project, 2005* http://gborg.postgresql.org/project/erserver/projdisplay.php
[PGFAQ05]	*PostgreSQL – FAQ, 2005* http://www.postgresql.org/docs/faq/
[PGHP05]	*PostgreSQL – Homepage, 2005* http://www.postgresql.org/
[PGHPGE05]	*PostgreSQL – Homepage German, 2005* http://www.postgres.de/
[PGJDBC05]	*PostgreSQL JDBC Driver for 8.0.1, 2005* http://jdbc.postgresql.org
[PGJXA05]	*PostgreSQL JDBC Driver, Frequently Asked Questions, 2005* http://jdbc.postgresql.org/documentation/faq.html
[PGLC05]	*PostgreSQL – Licence, 2005* http://www.postgresql.org/about/licence
[PGLIM05]	*Limitations of PostgreSQL, 2005*

	http://www.at.postgresql.org/users-lounge/limitations.html
[PGMAN05]	*PostgreSQL – Manuals*, *2005* http://www.postgresql.org/docs/
[PGNEWS05]	*PostgreSQL News Archive*, *2005* http://www.postgresql.org/about/newsarchive
[PGNPG05]	*The npgsql Project -- .Net Data Provider for PostgreSQL*, *2005* http://gborg.postgresql.org/project/npgsql/projdisplay.php
[PGPOOL05]	*pgpool page*, *2005* http://www2b.biglobe.ne.jp/~caco/pgpool/index-e.html
[PGPRESS05]	*PostgreSQL 7.4 Press Kit*, *2005* http://www.postgresql.org/presskit/en/presskit74.html
[PGSLONY05]	*The slony1 Project*, *2005* http://gborg.postgresql.org/project/slony1/projdisplay.php
[PGSP05]	*PostgreSQL – Support Platforms, 2005* http://www.postgresql.org/about/licence
[PGTD05]	*TODO list for PostgreSQL*, *2005* http://developer.postgresql.org/todo.php
[PGTS2G05]	*Tsearch2 - full text extension for PostgreSQL*, *2005* http://www.sai.msu.su/~megera/postgres/gist/tsearch/V2/

6.6 General

[IsLe95]	A Critique of ANSI SQL Isolation Levels, Berenson et al., *1995* http://www.cs.duke.edu/~junyang/courses/cps216-2003-spring/papers/berenson-etal-1995.pdf
[TpCp00]	The Two-Phase Commit Protocol, *2000* http://www.vermicelli.pasta.cs.uit.no/ipv6/students/andrer/doc/html/node18.html
[UnCo05]	Unicode Home Page, *2005* http://www.unicode.org/
[UnIs05]	*UTF-8 and Unicode FAQ for Unix/Linux, 2005* http://www.cl.cam.ac.uk/~mgk25/unicode.html
[WiDt05]	*Wikipedia – Distributed Transaction, 2005* http://en.wikipedia.org/wiki/Distributed_transaction
[WiSc05]	*Wikipedia – Scalability, 2005* http://en.wikipedia.org/wiki/Scalability

This research paper has been financed by Fabalabs Software GmbH. This is the result of a research by the "Institut Für Anwendungsorientierte Wissensverarbeitung (FAW)" and Fabalabs Software GmbH.

7 Information / Allgemeines

7.1 Es wird darauf verwiesen, dass alle Angaben in dieser Studie trotz sorgfältiger Bearbeitung ohne Gewähr erfolgen und eine Haftung der Bearbeiter oder des Herausgebers und Urhebers Fabalabs Software GmbH ausgeschlossen ist.

7.2 Alle Rechte der Verbreitung, auch durch fotomechanische Wiedergabe, Tonträger jeder Art, auszugsweisen Nachdruck oder Einspeicherung und Rückgewinnung in Datenverarbeitungsanlagen aller Art, sowie die Nutzung oder Veränderung sind vorbehalten.

Der Leser der Studie und der dort aufgeführten Inhalte anerkennt, dass der jeweils in der Fußzeile stehende Copyrightvermerk im Sinne des Welturheberrechtsabkommens an der vor Urheber festgelegten Stelle bleibt, um den Vorbehalt des Urheberrechtes genügend zum Ausdruck zu bringen.

Dieser Urheberrechtsvermerk darf werder vom Leser und/oder von Dritten entfernt, verändert oder disloziert werden.

Alle verwendeten Hard- und Softwarenamen sind Handelsnamen und/oder Marken der jeweiligen Hersteller.