




# **HFSQL**

## ***UNIVERSAL DATABASE***

*Windows, Linux, Mac, Android, iOS (iPhone, iPad)*

*Client/Server, Cluster, Cloud, Standalone, Mobile, Embedded*

[www.windev.com](http://www.windev.com)



# WELCOME TO A WORLD OF SECURITY AND PERFORMANCE

**A**n enterprise's data is a strategic resource.

The **HFSQL**® Relational Database Management System allows you to manage this data safely.

The performance is remarkable.

Used on several millions computers the world over, the flexibility and the scalability of **HFSQL**® allows real time responses to the most demanding mission critical applications.

You too, choose **HFSQL**®!

## Table of contents

Overview	3
Local	4
Mobile – Embedded	4
Client/Server	4
Cluster - Cloud	5
Types of data and index	5
SQL	6
Features	6
Security	10
Openness	11
The tools	12
Programming	17
List of supported SQL statements	17
List of WLanguage commands	18
Vocabulary	21
Who uses HFSQL?	22
Benefits	22

# HFSQL®

HFSQL was first released in 1998.

**HFSQL Technical-commercial documentation.**  
Some knowledge of WINDEV, WEBDEV or WINDEV Mobile is useful. If you're not familiar with them, don't hesitate to request their complete documentation (free).



# HFSQL®



## HFSQL OVERVIEW

### A UNIVERSAL DATABASE

HFSQL is a powerful RDBMS (Relational Database Management System).

HFSQL exists in four versions:

- mobile version (embedded)
- local version (standalone or network)
- Client/Server version
- cluster version.

HFSQL is suitable for all types of applications: business applications, 24/7 real-time critical applications, software, application servers, Web servers, standalone PC or mobile devices.

Note that HFSQL is the new name of HyperFileSQL.

*HFSQL is fully compatible with HyperFileSQL and Hyper File.*

### PERFORMANCE, SECURITY, OPENNESS, FLEXIBILITY

HFSQL is the ideal choice for a database engine.

**Open:** based on industry standards, HFSQL doesn't lock you up into a proprietary technology.

**Flexible:** support for large volumes of data (tens of billions of rows in a table) is provided.

**Platform independent:** tables can be moved from a Client/Server implementation to a mobile implementation, from a Windows server to a Linux server, etc....

**Scalable:** you can go without constraints from one user to hundreds of users, from a 2-tier architecture to a multi-tier architecture...

**Frugal** in resources: the Client/Server mode takes up less than 40 MB on disk.

HFSQL works in **heterogenous** environments: Windows, Linux, Mac, iOS, Android, TSE, Citrix, ADSL, VPN, Wi-Fi, 3G, 4G, in the cloud...

The forward and backward **compatibility** of tables is ensured.

**Longevity** of the publisher: PC Soft has been around for more than 25 years.

**Performance, scalability:** thanks to an optimize index and cache management, the speed is constant.

**Secure access:** protection against SQL injection is ensured via the automatic creation of secure UI.

### REDUCED TCO

An important characteristic of HFSQL is its unlimited free deployment (see license).

There is no additional cost, not for the number of CPUs on the server, not for the number of client computers, not based on the type of application (commercial,...) etc.

HFSQL comes as a complete product, with all its features. The maintenance costs are very low.

The technical support is also free (as part of a WINDEV, WEBDEV, or WINDEV Mobile license). It is provided via email. The DBA and developers can also access very active professional newsgroups.

Service packs are offered on a regular basis for free.

100% **Windows**

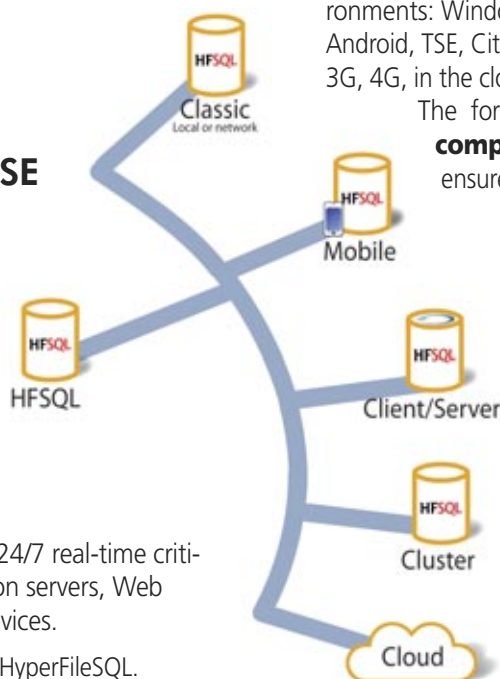
100% **Linux**

100% **Cloud**

100% **Mac**

100% **Android**

100% **iOS**



HFSQL®

# VERSIONS

HFSQL is available in 5 versions. These versions are compatible among themselves.

## LOCAL VERSION ("CLASSIC" VERSION)

The local version (standalone and network) of HFSQL offers performance, ease of deployment, installation and maintenance. This version is also called "Classic" version because it is the first version that came out, back in 1988.

Compatibility with previous versions is complete (tables, index, relationships, constraints).

This version is specifically designed for standalone computers and small networks. A common use for the Classic version is integrated into a software.

The database is created and installed automatically on the end user's machine. The maintenance is automatic.

A HFSQL database can also be installed and used directly on a USB key.

This is useful for mobile applications, or for very sensitive data.

HFSQL Classic installs on machines with Windows (2000, 2008, 2012, Vista, 7, 8, 10 ...), MacOS, iOS, (iPhone and iPad), Android and Linux.



## MOBILE VERSION (EMBEDDED)

HFSQL is totally adapted to mobile devices of all types.

HFSQL only requires small amount of resources, and installs on all mobile devices (terminal, smartphone, tablet) that run on Windows CE, Windows Mobile, Windows 8 RT, iOS (iPhone and iPad), Android.

The installation is very simple, and maintenance, almost zero, is automated.

The performance is amazingly fast.



iOS



**Imagine,** 128 GB on a memory card is now common. Thanks to HFSQL you can now easily and for a low cost embed large size secure databases (**up-to 130 million rows**) on mobiles, tablets, smartphones.

## CLIENT/SERVER VERSION



The Client/Server version of HFSQL is the ideal version for managing large number of users and remote accesses. Local and remote accesses are supported. The installation is extremely simple, and

the administration is easy though very powerful.

HFSQL is not limited in the number of processors used, or memory. Load balancing is supported for better response time.

The engine is auto-restart.

HFSQL works both in 32-bit and 64-bit modes. Servers and clients can be mixed.

*Some of the supported Windows servers:*

- Windows Server 2012, 2008, 2003, 2000, XP, Vista, 7, 8...

in all their versions.

*Some of the supported Linux distributions:*

- RedHat, Debian, OpenSuse, Ubuntu, Fedora, Mandriva, CentOS...

*Among the supported clients:*

- 32-bit and 64-bit Windows
- Linux
- MacOS, iOS
- Windows CE and Mobile
- Android
- ...

## CLUSTER VERSION (SERVER FARMS)



Thanks to the HFSQL cluster feature, a set of physical servers appears as a single server to the clients. The potential failure of a physical server does not prevent access to the database (high availability, fault tolerance). Servers automatically replicate each other in real time. The read load charge is distributed on all the servers (load balancing for rea-

ding). You can add and remove servers on the fly. If a server crashes, it automatically re-synchronizes when restarting. When a user is connected to a server that fails, the application will be automatically reconnected to a valid server (automatic fall over).

## CLOUD



HFSQL Client/Server is available in cloud version, from PCSCloud for instance. open a cloud account, and your server is immediately operational: no installation, no equipment or system management needed,... Billing is done based on actual consumption. Installing a HFSQL database in the cloud frees you from all aspects of management and hardware maintenance.



## DATA AND INDEXES

### DATA TYPE

HFSQL supports all the types of data:

- Text, character
- Numeric (integer, real, decimal with 38 significant digits),
- Date, time, duration
- Boolean
- Array type column
- Blob ("memo", binary format: image, video,...)

Powerful features are available:

- Unicode is supported, with support of linguistic sorts
- The sort order for different character sets is taken into account.
- Default value
- Management of NULL...

### UNICODE

HFSQL supports Unicode type text and blob columns.

Indexes can be sorted by the alphabetical order corresponding to each language: Russian from Russia or Ukraine, Chinese from Singapore, Taiwan, Hong Kong, Macao, ...

主题	日
Technologies	2 octobe
高潮重新加热	2007年10
History of Philosophy	2 octobe
农业地方在我们的世界	2007年10
地球昨天和的气候在明天	2007年10
Painting in Amazonia	2007年10
巨大天气现象	2007年10
所有在大轰隆	2007年10
地球昨天和的气候在明天	2007年10
令人惊讶的植物	2007年10
21世纪的技术	2007年10

*Data in Chinese*

### INDEX AND KEYS

HFSQL manages keys and indexes for any type of column.

In order to insure optimum performance, the server uses an optimization mechanism based on the data distribution, that get automatically activated during idle times.

The following types of indexes can be

created:

- Simple index
- Composite index
- Partial index
- Full text index.

HFSQL insures data integrity by managing:

- Unique constraints
- Cardinality constraints
- Automatic Identifier
- Primary and foreign keys.

### FULL TEXT INDEX

The "full text" search allows for very fast string (words or expressions) searches inside your data. It allows you for instance to find a word among one millions rows in less than 2 ms (average for found occurrence).

This enables you to index, without programming, the texts found in a HFSQL database.

Results are offered according to a relevance order ("ranking").

To perform searches on words stored in RTF or HTML documents, the creation of the index supports these formats by ignoring tags during indexing.



Texts can be contained inside text or blob type controls.

A full text index can index one or more columns, therefore a single search can be done on several columns at the same time.

Hallow words are supported.

## CAPACITY (VOLUMES)

HFSQL Client/Server offers large storage capacity, in line with current and future storage models, as well as the ever increasing needs of enterprises.

During a recent roadshow, in front of more than 10.000,20 professional developers, PC SOFT demonstrated the use of a HFSQL database containing more than 20 billions rows: data searches started instantly!

HFSQL MAXIMUM CAPACITY	
Columns per table	65,535
Index per table	65,535
Rows per table	329 millions of billions
Table size	4 millions of TB
Row size	2 GB
Column size	65,535 bytes
Size of a blob column (text memo, image, video, binary, etc.)	4 GB

## SQL: OVERVIEW

HFSQL supports the ANSI SQL 92 standard. The SQL supported by HFSQL also accepts a large number of additional and specific syntax for SQL Server and Oracle, among others. HFSQL supports subqueries and nested queries.

HFSQL supports union operators: union, cartesian, join, external join, aggregation operators: count, sum, avg, min, max, standard-deviation, variance, sort and group operators: group by, having, order by...

The speed of the SQL engine is optimized by the use of index statistics and the constraints for choosing the most discriminating access indexes in queries.

The advanced management of memory caches also improves performance.

The engine automatically performs load balancing: if a client executes a large number of queries requiring a lot of resources (CPU, ...), the server automatically balances the load in order to not penalize the other clients.

Simultaneously with the SQL code, you can benefit from the functional richness of the WLanguage 5GL.

The direct use of WLanguage functions and the call to stored procedures (developed in WLanguage themselves) are possible in your applications.

You'll find at the end of this document the list of SQL functions supported by HFSQL, as well as other programming information (cursor programming).

```
SELECT
    Product.Name AS Name,
    SUM(OrderLine.Quantity) AS Quantity_Sum,
    Customer.ZipCode AS ZipCode
FROM
    Product
    LEFT OUTER JOIN
    (
        (
            Customer
            INNER JOIN
            Orders
            ON Customer.IDCustomer = Orders.IDCustomer
        )
        INNER JOIN
        OrderLine
        ON Orders.IDOrders = OrderLine.IDOrders
    )
    ON Product.Reference = OrderLine.Reference
WHERE
    Customer.ZipCode LIKE '34%'
GROUP BY
    Product.Name,
    Customer.ZipCode
```

*Example of an SQL query*

## FEATURES

HFSQL offers a large number of features. You'll find the description of the main features below.

The entire online help for HFSQL is available on the Internet at [doc.windev.com](http://doc.windev.com)

### STORED PROCEDURES (UDF)

The stored procedures (sometime called UDF) are used to simplify the development and the maintenance of your applications by factorizing the code.

Indeed, when the method for calculating a result or a business rule found in a stored procedure evolves, all you have to do is modify the stored procedure on the server without performing any modification in the deployed applications

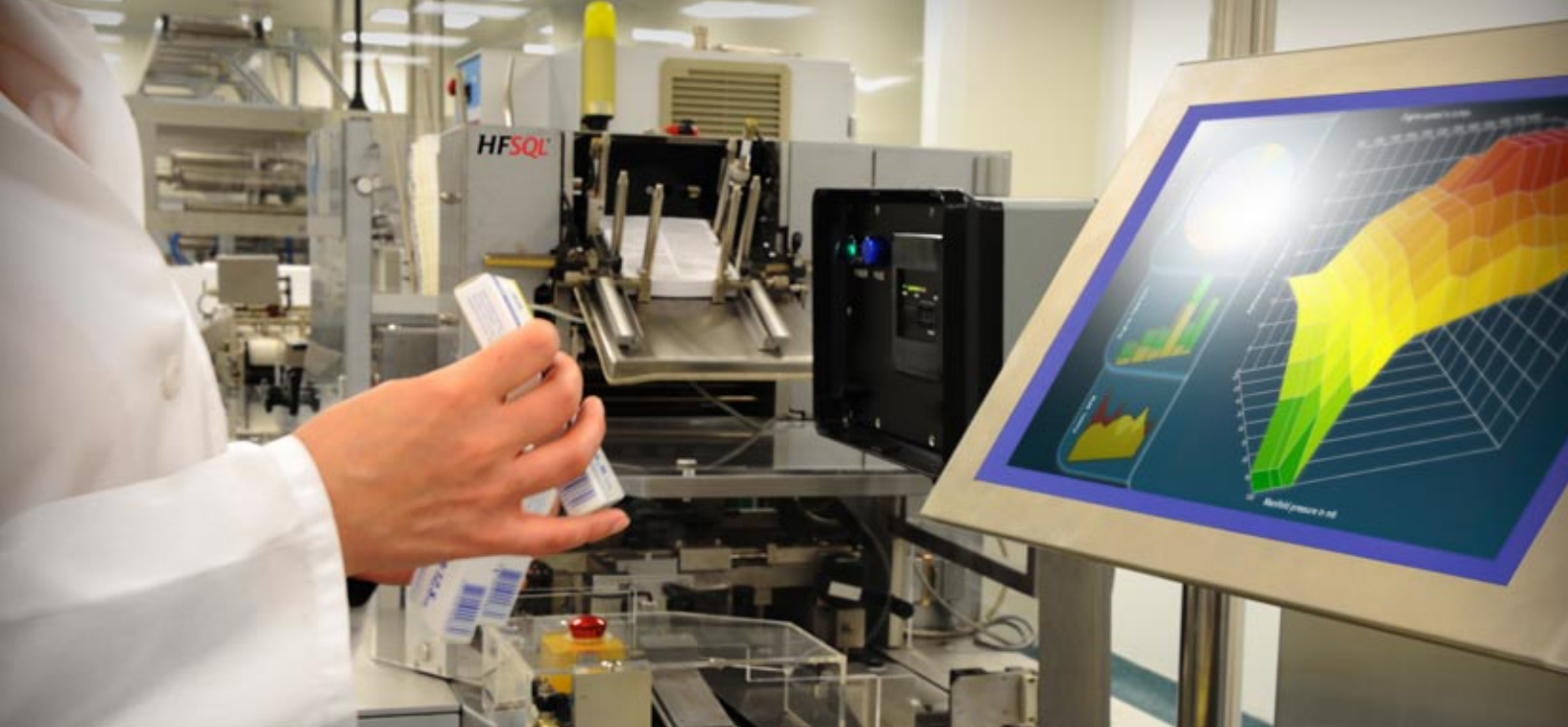
A stored procedure is also used to limit the number of back and forth processes between the client computer and the server, and therefore increases the speed of processes.

The same stored procedure can be shared among several applications.

Stored procedures are programmed using WLanguage, and therefore benefit

# 329.000.000.000.000.000.000

**329 millions of billions... This is the number of rows (records) that can be found in a HFSQL table: you're safe!**



from the feature richness and ease of use of the 5GL. Stored procedures accept parameters and can return integers, strings, dates, etc...

The creation of a stored procedure from the WINDEV or WEBDEV environment is really easy.

## SQL VIEWS

An SQL view is a "virtual data source", defined via an SQL query.

All the SQL views created are kept on the HFSQL server, and they can be reused from the applications accessing the database.

The application can create queries on these SQL views.

An application can use SQL views to not be dependent on the physical organization of data in the database.

"Materialized views" are also available.

The fundamental difference with a simple SQL view is that the result of a "materialized view" is physically stored on the server's drive.

Unlike an "SQL view" which is re-extracted at each call, a "materialized view" stores the data on the drive.

## TRIGGERS

A trigger allows you to trigger a stored procedure before or after an event on a database table: for example when deleting a row, or after modifying a row.

A trigger brings a lot of security.

The trigger will get triggered regardless of the application or component that ac-

cesses the database and that performs the defined operation, without the application's developer having to program anything.

Server triggers, as their name indicate, are run directly on the server.

The right to create a trigger is defined via the database's rights.

The WINDEV environment indicates to the developer the presence of these triggers. Triggers are viewed in the data schema.

## INTEGRITY: CONSTRAINTS, DELETIONS, CASCADING UPDATE

It is easy to define integrity constraints.

The cardinalities can be configured:

(0,n); (0,1); (1,n); (3,n); etc.

Reflexive links are supported.

Constraints examples:

- Referential integrity:

referential integrity will prevent an author from being deleted, as long as the database contains at least one book referring to this author.

You cannot delete a row in a table if this row is linked to other table rows. For example: you cannot delete a customer if there are orders linked to this customer.

Referential integrity can be defined for each link, from the data model editor.

- Cascading deletion:

If a row is deleted in a table, the corres-

ponding rows in the linked tables are also deleted (this constraints can be enable or disable for each relationship)

## LOGS

The log is a special table where all the operations performed on one or more tables from a given time are automatically stored.

The log contains the history of the logged tables: author, date and time, before/after value, application name, IP ...

The following operations can be performed from a log:

- restore the content of a logged table if the data is lost or destroyed
- restore the content of a table up to a given date
- find the author, the date and time of an operation performed on a specific row
- Keep the history of a table use (to calculate statistics for instance)

These operations can be run from a command line, from the WDLLog tool or via programming.



## TRANSACTIONS: HFSQL ACID

A transaction is a set of indissociable operations: either all the operations in the transaction are performed, or none is performed.

Transaction management is the best way to ensure the integrity of a set of indissociable write operations performed on HFSQL tables.

A transaction is used to make sure that the updates performed on one or more tables have successfully completed. HFSQL supports all the types of transactions, and therefore meet the ACID criteria (ACID is the acronym for Atomicity, Consistency, Isolation and Durability). HFSQL Client/Server offers 4 isolation modes for transactions.

- Unvalidated data (READ UNCOMMITTED)
- Validated data (READ COMMITTED)
- Instant transaction photography (REPEATABLE READ)
- Serializable transaction (SERIALIZABLE).

## "BLOB" DATA COMPRESSION

"blob" type data (text and binary memos) can be compressed to optimize the space used on the disk. The space used can be significantly reduced this way.

## FRAME COMPRESSION



A frame is a data packet that flows on the network.

HFSQL, like all the DBMSs, is using frames to establish communication between the server and client computer. A server's data transfer speed depends on the travel speed of the frames through the network and on the size of these frames.

Frame compression allows you to reduce the size of the packets traveling over the network.

In a context of remote communication,

frame compression is very important. The speed for remote connection is improved.

## AUTOMATIC ROW LOCKING

HFSQL supports locks at the table level and at the row level.

Support for locks at the row level ensures better access security.

## SEVERAL DATABASES ON THE SAME SERVER

HFSQL classic supports the presence of several databases on the same server. The databases are isolated. Specific rights can be defined on each database.

This prevents having to use multiple servers.

## AUTOMATIC RECONNECTION

This feature automatically manages disconnections happening between the client and the server.

Usually, this problem occurs with hardware whose connection with the server is not always on: mobile devices (Wi-Fi, 3G, 4G, ...) notably.

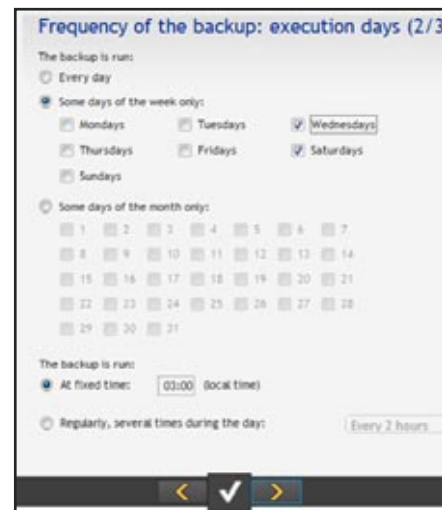
In a Wi-Fi connection for example, the link between a remote device and the server can be interrupted briefly. This can also happen on regular wired networks. When the application is reconnected, the application needs to resume at the point where the connection was interrupted, and make sure the buffers are accurate.

With the automatic resuming

of the connection, all the buffers and positions are stored and reassigned. The application can resume without any error, as if the connection never was interrupted.

You can also manage the connection interruptions by programming, or execute additional processes if you want.

## BACKUP



*Backup configuration*

Backups are important features of a database.

You can save the entire server's content, only the database, or only a selection of tables with or without index.

HFSQL supports hot backups, as well as differential backups.

A backup is portable, for instance from a Windows server to a Linux server, or from a Client/Server version to a Classic version.

A backup can be triggered from the Control Center (instant backup, scheduled backup), or via programming, directly from the application.

The frequency of the full backup and the differential backup can be specified. For example: 1 full backup every month and a differential backup once a week.

The execution of stored procedures before





and/or after the backup lets you perform automated processes. send email, copy the backup to a network location, etc... The number of backups to keep can be specified.

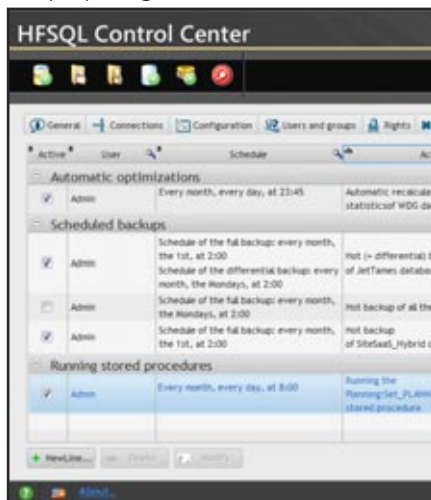
A backup can be performed "hot", without disconnecting users, transparently, without interfering with the applications.

## INTEGRATED SCHEDULER (SCHEDULED TASKS)

HFSQL has an integrated scheduler that lets you define and configure scheduled tasks.

A task consists not only in executing a stored procedure (UDF), but also in triggering a backup, or in forcing a performance optimization request.

The definitions is done in the Control center or via programming you can create, add, modify, enable, disable tasks scheduled via programming, or from the administration tool, as long as you have the proper rights.



*Graphical and user-friendly interface of the scheduler (scheduled tasks)*

The scheduler allows the DBA to program the automated execution of tasks on the server: it also allows you to create batch processes.

Tasks can be run at a set date, and repeated at regular intervals.

## REPLICATION



HFSQL offers 4 types of replication:

- HFSQL server replication
- replication between heterogeneous databases, a HFSQL and Oracle database for example
- replication with mobile devices (iOS, Android, ...)
- offline replication, without permanent link

A replication is easily defined via the replication wizard, or via programming.

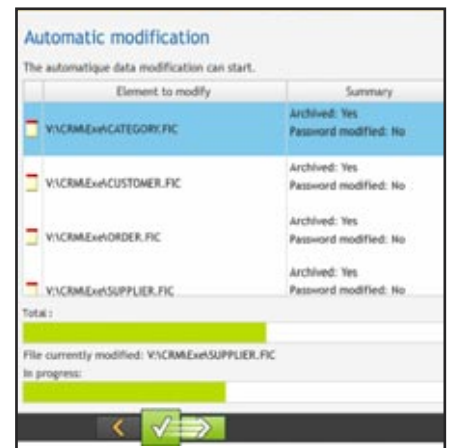
## AUTOMATIC DATA MODIFICATION ON EACH SITE (DSS)

Which developer hasn't complained about having to write some quick and dirty hack to add a column or increase its size, add an index to an existing table, or change the type of data in a column? Writing those scripts is always tricky because they alter the data.

With HFSQL these will be things of the past!

HFSQL manages the evolution of the data schema transparently thanks to the DSS (Data Schema Synchronization) technology.

No more "hack jobs"! No more scripts!  
No more risky "Alter table" commands!



*DSS: Automatic update of the data schema (also called Auto. Modif.)*

DSS automatically performs:

- The comparison and synchronization of the database structure and data against the reference schema
- The addition, deletion or renaming of columns
- The change of type, size
- The addition/deletion of key/index, addition/deletion of constraints
- The addition/deletion of triggers and stored procedures.

DSS can also be started via command line or programming.

This DSS feature can be executed live (hot), without disconnecting the users, transparently, without interfering with the applications running.

## "HOT" ADMINISTRATION

A large number of maintenance tasks can be performed live ("hot"), without needing to disconnect users, and without interfering with their running applications, which continue to read and write data during this time:

- Hot "auto modif" DSS
- Hot reindexing
- Hot automatic optimization of performances
- Hot change of password
- Save.

## LINK WITH OTHER DATABASES

You can of course use HFSQL in parallel with other databases. Actually, most IT departments use several heterogeneous databases.

The same application can use a HFSQL

database simultaneously with another database.  
HFSQL also lets you import and export data with other databases.

## SECURITY

The integration, the automatic lock management, the Control Center ... ensure by their very own existence a strong security.

Security specific features are also available.

### ACCESS RIGHTS: AUTHENTICATION FOR ESTABLISHING THE CONNECTION

The server has a user authentication system.

It checks that a user is authorized to connect, and then that he has sufficient rights to run his queries: for example rights to delete rows when running a deletion query.

You can restrict access for a user based on his IP address or a DNS name.

The tuning of the rights is very granular: at the server level, the database level or the table level.

You can choose to do it by programming or via a user-friendly interface.

You can define an expiration period for password.

You can define groups of users.

#### For the server:

- Rights to delete and add users or groups
- Rights to see the users and the groups
- Rights to create a database
- Rights to change the rights
- Rights to stop the server
- Rights to change your own password
- Rights to disconnect the client computers
- Rights to send messages to the client computers
- Rights to configure the server
- Rights to configure the priority of users
- Rights to perform backups
- Rights to configure the scheduled tasks
- Rights to see the activity statistics and the logs of the server
- Rights to define a server replication.

#### At the database level:

- Rights to add new rows into a table (data file)
- Rights to lock the tables or the table rows
- Rights to change the rights
- Rights to modify the integrity rules on a table
- Rights to modify the owner of an element
- Rights to connect to a server (encrypted and unencrypted connection or encrypted connection only)
- Rights to create a table via programming
- Rights to enable and disable the management of duplicates
- Rights to read a table's rows
- Rights to start a re-index operation or

to calculate statistics

- Rights to perform automatic table modification (DSS)
- Rights to delete a table's rows
- Rights to delete a table's rows
- Rights to delete a database
- Rights to delete a table by programming
- Rights to enable and disable the management of integrity
- Rights to lock access to a database
- Rights to run stored procedures and/or VLanguage commands in the queries
- Rights to configure the stored procedures
- Rights to debug the stored procedures
- Rights to modify the triggers
- Rights to perform backups.

#### At a table's level:

- Rights to add new rows into a table
- Rights to lock the tables or the table rows
- Rights to change the rights
- Rights to modify the integrity rules on a table
- Rights to modify the owner of an element
- Rights to enable and disable the management of duplicates
- Rights to read a table's rows
- Rights to start a re-index operation or to calculate statistics
- Rights to perform automatic table modification (DSS)
- Rights to delete a table's rows
- Rights to delete a table's rows
- Rights to delete a table by programming.

## SQL INJECTION NOT POSSIBLE

The use of the WINDEV window generator and WEBDEV page generator, with their edit controls that are automatically generated based on the data schema, makes attacks via "SQL injection" almost impossible, and this automatically. The use of SQL queries created with the query editor brings the same level of security.

The data that the end user enters are automatically checked, in real time as soon as they're entered, and are not sent to the application if they're unexpected, erroneous or inconsistent.



*HFSQL Control Center: defining the rights*



## ENCRYPTED CONNECTIONS

The connection between the client and the server can be encrypted.

To define a high level of security, you can forbid non-encrypted connections to the server.

## ENCRYPTING THE DATA

Data access can be secured, and data itself can be secured.

We can specify that the opening of the table requires a password.

The data itself can be encrypted.

Several encryption modes are supported:

- Standard on 128 bits
- RC5 12 rounds in 128 bits
- RC5 16 rounds in 128 bits.

If an attacker obtains an encrypted file (theft, copy, recovered from a recycled machine, on a lost computer, ...), he or she won't be able to use it.

## INCIDENT DETECTIONS

When the HFSQL server detects an incident (for example an inaccessible replicated server, or a schedule tasks that triggers an error), the server sends a notification of this incident to a list of specified email addresses.

## OPENNESS

HFSQL is open to all the technologies, and is easily inserted into your existing Information System.

## 32 & 64 BIT ODBC DRIVER

The ODBC driver (32 or 64 bit driver) allows third-party applications to access the data stored on a HFSQL server, such as PHP, Python, Ruby, Access...

## 32 & 34 BIT OLE DB PROVIDER

The OLE DB driver (32 or 64 bit driver) allows third-party applications to access the data stored on a HFSQL server, such as C#, ASP.Net, Crystal Reports, Business Object, PHP, Excel, ...

## DATABASE IMPORT

The WDCONVER tool (provided with your product) lets you import third-party databases: Oracle, SQL Server, MySQL, ... The import of the data schema is automatic.

Data import from text formats (customizable separators: tab, espace...), CSV, XML, is also supported.

## THE ADMINISTRATION TOOL

### ADMINISTRATION: HFSQL CONTROL CENTER

The HFSQL Control Center is an essential management tool with an intuitive and user-friendly graphical interface.

The HFSQL Control Center lets you perform a large number of tasks, from a network computer or from the Internet, such as:

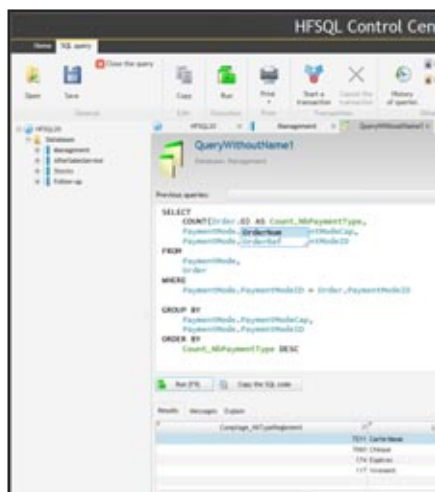
- Database management
- Management of the data size
- Stop/start the various server instances
- View information specific to the server, the database, the tables
- Uninstall the HFSQL server
- Display the list of current connections
- Ability to end/disallow connections.
- Send messages to users
- Configure the location of the databases, activation and location of the

logs and activity statistics,

- Define the connection port to the server,
- Define the port for remote debugging,
- Edit the accounts
- Manage the current transactions, transaction rollback
- Manage scheduled tasks
- Create backups
- Manage the backups
- Hot tuning: cache size, log activation, etc....
- Create, delete, import databases.
- Database explorer • Run the queries
- Save and restore the data,
- View the structure of the tables.
- Automate common functions.
- Monitoring.
- Manage users and groups of users, as well as their rights
- Manage the connected users
- User disconnection
- Gather accurate statistics on the server use: computers, queries, logs, parameters, .....
- View row locks
- ...



The HFSQL Control Center



*The HFSQL Control Center SQL querier*

- control screen (visual warning and/or sound)
  - starting a WLanguage procedure
  - third-party program (this program can for instance, send a configurable message via SMS to chosen numbers).
- Among the monitoring parameters that can be specified, you'll find:

- the frequency: test interval, from 2 minutes to 1 day
- repetition: in case there's no answer from the monitored element, how often to retry and how long before triggering the warning
- text of the message to send
- the message's medium (SMS, e-mail, ...)

## LINK WITH WINDEV, WEBDEV AND WINDEV MOBILE



## WINDEV, WEBDEV AND WINDEV MOBILE NATIVE ACCESS

WINDEV, WEBDEV and WINDEV Mobile are Integrated Development Environments.

The HFSQL access is "native" in WINDEV, WINDEV Mobile and WEBDEV, which means that the access performance (read, write) are optimized. HFSQL data schema are also directly and instantly recognized by the WINDEV, WEBDEV and WINDEV Mobile environments, and therefore benefit from the automation and wizards of these environments. automatic creation of UI, controls, completion in the code editor ...

Data binding is supported, visually in the environment and by programming.

## MODELING A DATABASE

The definition of a database schema is easily done using the provided powerful visual editor.

A large number of wizards are available to help you out.

The visual editing of the data model (creation, deletion, modification of the tables, columns, relationships, constraints, index, triggers,...) enables you to define a database schema without having to write any SQL code. The editor displays graphically the organization of the data and processes.

## THE TOOLS

### WDMAP: DATA VIEWER

The WDMAP tool lets you view, edit and modify data in table.

WDMAP is very useful in the test and debugging phase.

WDMAP lets you filter and sort data, perform instant export (to Words, Excel, OpenOffice, XML, ...)

### WDHFDIFF: DATA COMPARISON TOOL

The WDHFDiff tool lets you compare:

- the structure of 2 tables
- the data of 2 tables.

This can be very useful in the fine tuning phase.

### MONITORING ROBOT

The monitoring engine (which can be re-distributed with your applications) lets you secure your servers.

The monitoring engine always monitors, and instantly detects new unauthorized connections with the server.

The server notifies you by:

- sending a configurable email message to the specified addresses (up to 20 addresses)
- message sent to a specific application (internal messaging,...)
- message sent to the integrated messaging system



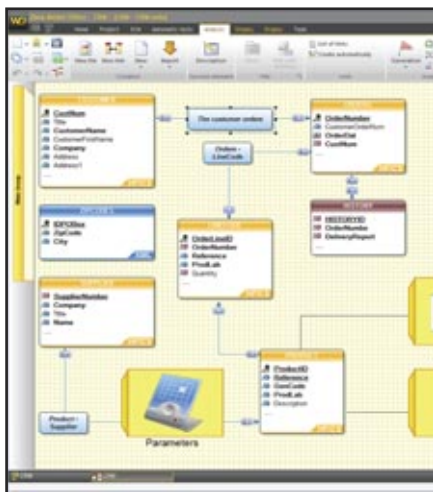
### SIS: SOFTWARE INFRASTRUCTURE SUPERVISOR

SIS lets you identify all the accesses from the WINDEV applications and WEBDEV sites to the network resources, including the HFSQL databases.

SIS lets you analyze the access to databases over time, find out which applications and which computers access a database,...

Accurate statistics are available at any time.





*A partial data schema  
(an analysis)*

An automatic import of existing schemas can be performed.

The editor knows how to import schemas from databases such as HFSQL, SQL Server, Oracle, OLE DB, ...

To create a data description, we start by specifying the type of columns, the type of keys (index) ...

Any new column is stored in the data dictionary.

Then we just need to define the relationships between the tables.

To link tables, simply draw a link with the mouse!

The wizard asks questions in natural language to determine the nature of the relationships. For example "A customer can have several orders: YES or NO",

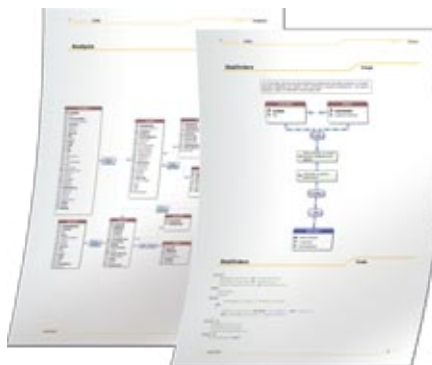
"Each order can have several products: YES or NO", etc.

The wizard also asks whether checking referential integrity must be automatic or not.

The wizard then asks whether to generate any relationship tables necessary, or whether to use existing tables.

Finally, the wizard asks for relationships' name: the schema is defined.

A documentation of the database schema can be printed at any time (hardcopy, HTML, PDF, Word, OpenOffice).



*Pages taken out of a  
documentation*

The visual data model editor also supports:

- Reverse analysis from a server
- Logical or physical modeling
- Connection editing
- Schema comparison
- Schema history

- The generation of DDL scripts
- The export of the data model in a vector image format.

## CURSOR IN WINDEV AND WEBDEV

Native programming in WINDEV and WEBDEV is greatly facilitated by a set of highly advanced automations and wizards.

Relationships between tables are automatically detected.

The access to a database control is easily defined using a clear and intuitive syntax: table name, column name (For example: `customer.name`).

In the environment, the code entry assistance is constant and helps avoid typos and having to look up names.

## PERFORMANCE TUNING, AUDIT

The Profiler and the Dynamic Audit let you analyze an application's performance, and thus verify that the data access is programmed in an optimum way.

Tuning allows you to optimize queries, check index, set up statistics, monitor the server and control the memory, the CPU use, disk space, connections, etc.

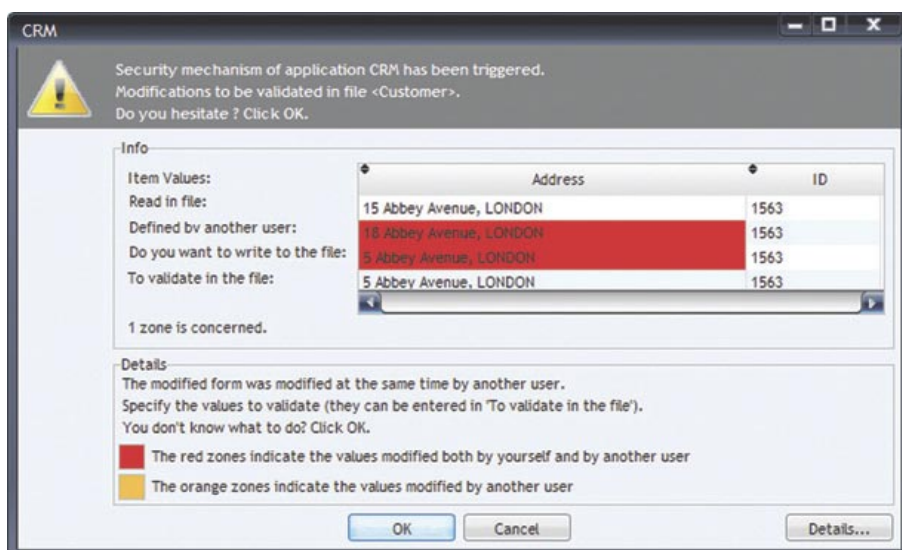
The **Explain** SQL function enables you to analyze in what order a query is executed.

## AUTOMATIC ASSISTANCE FOR THE END USER

In a WINDEV application, the assistance to the end user is automatically provided on HFSQL aspects in the case of the following errors:

- detection of the non-protected concurrent accesses
- duplicates
- non respect of the integrity constraints
- wrong password
- disconnection
- lock.

if one of these errors occurs, the application automatically displays a relevant help window.

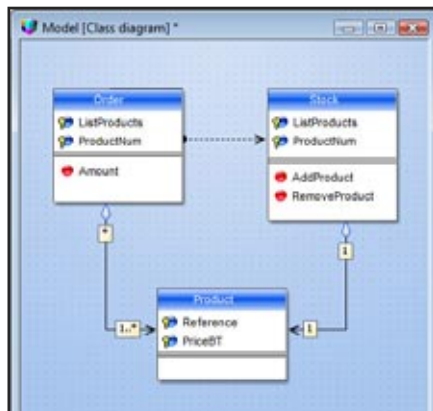


*In this case, 2 users are trying to modify the same control at the same time! A window comes up automatically and asks for the correct value. No programming is required.*

## MULTICONTEXT

You can use several contexts and several different connections on the same database at the same time.

## UML



The UML class diagram

WINDEV and WEBDEV supports the 9 UML models.

Specifically, the class diagram can be automatically generated from the database schema.

3-tier programming is extremely simplified by an automatic object-relation mapping: classes are automatically generated to implement the business logic specific to the application, as well as the update of the object toward the table and vice versa.

## AUTOMATIC WINDOW AND UI GENERATOR FROM TABLE SCHEMA (RAD)

UI, windows, pages, controls,... (as well as the code) can be generated automatically.

The generated UIs take into account the table's definitions.

For example, if a column is a numeric type column, with a maximum length of 8, only data of this type will be authorized in input in the corresponding control.

It will be impossible for the end user to enter a text or a number of greater length. An error message automatically comes up, and the erroneous value entered will not be sent to the application or site.



Example of a window created with WINDEV from the data schema

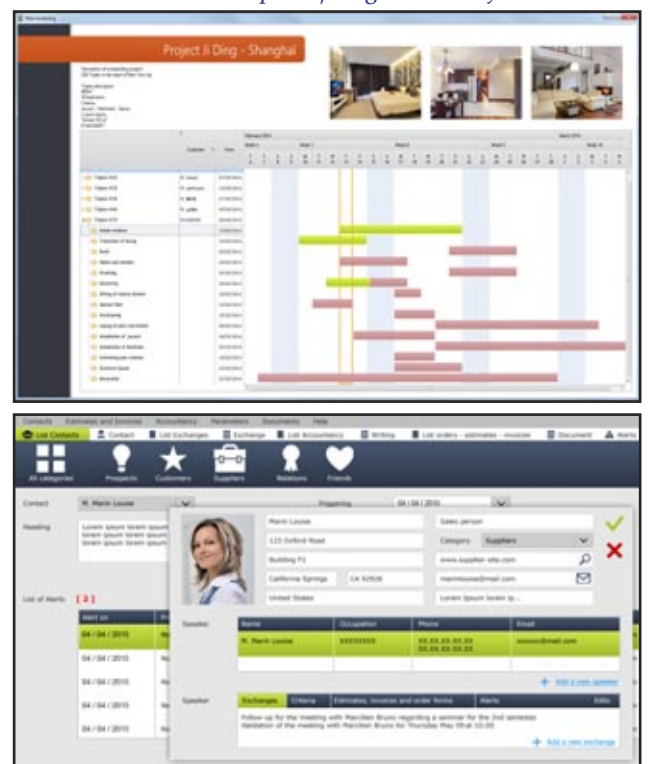
The necessary sophisticated controls are generated via RAD and are of course available to create the UI "by hand". They're available by simple drag/drop.

Here's the list of controls:

- formatted edit controls
- display control (static)
- tabs
- drop-down list box
- combo boxes
- auto-filled combo box
- image
- animated image
- scrollbar
- graphic button (icon)
- animated graphic button
- text buttons
- on/off button
- delay button
- check box
- single or multi-column radio buttons
- array
- listview
- treeview list
- treeview table
- hideshow
- OLE control
- ActiveX control
- click area
- spin buttons
- controls with soft shadow
- sliders
- HTML control
- icon bars
- geometric shapes

- splitter
- status bar
- Web camera
- RTF
- loopers
- progress bar
- sidebar
- chart
- bar codes
- carousel
- calendar
- spreadsheet control
- Gantt
- etc.

Examples of UI generated by WINDEV





## QUERY EDITOR

Query creation is done in SQL or in WLanguage 5GL.

The queries can be directly coded, or generated by the query editor (Reports & Queries).

This editor comes with WINDEV and WEBDEV, and can be freely distributed to the end users of the applications you've created.

The query editor lets you optimize the database description (schema) by detecting and defining the indexes needed for the best runtime performance for the created queries.

The query editor generates the queries both in SQL and in natural language, so you can check they do what you meant them to do! This way there's no risk of error.

The query is also generated in schematic form (animated graphic).

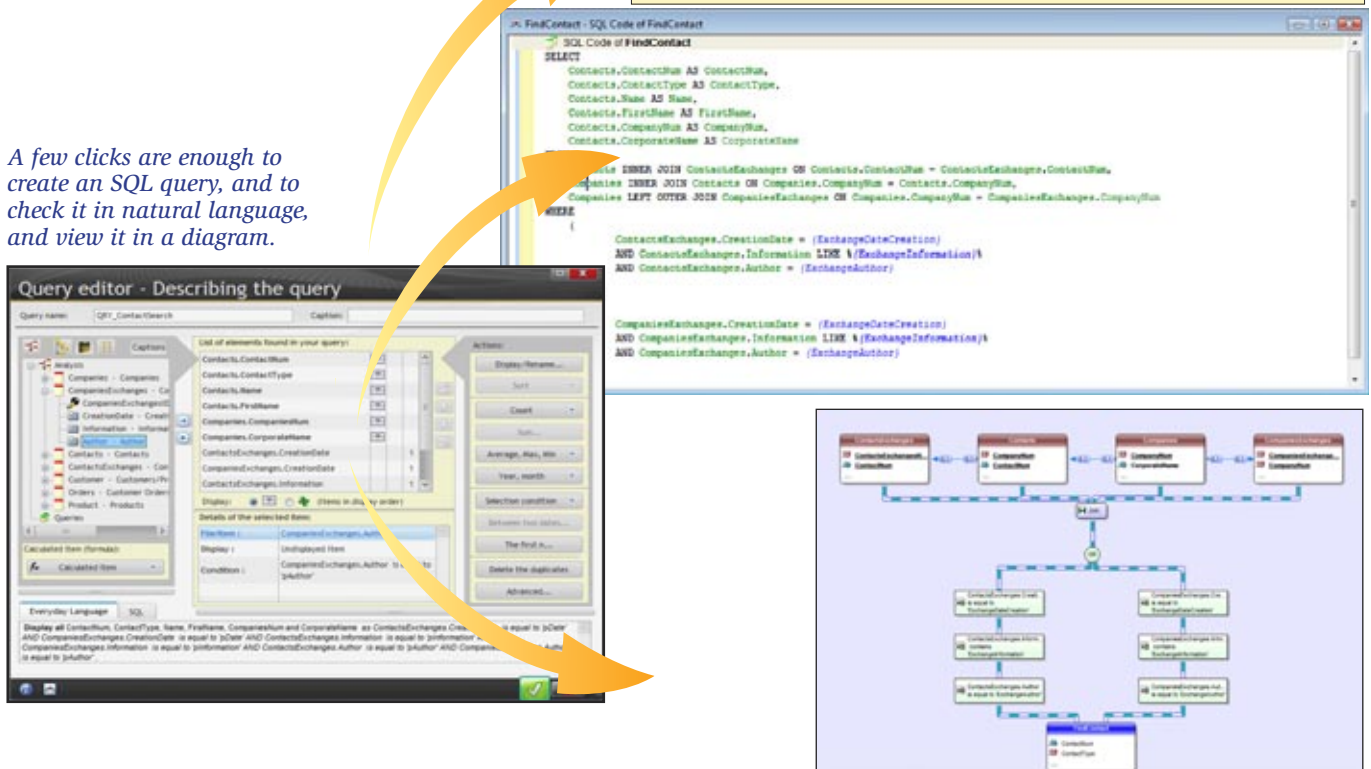
Creating a query is simple: using the wizard, choose the columns to include and the selection conditions, then the query is generated in optimized SQL code.

The editor can also perform a reverse-analysis of existing queries.

A query can use the result of another query as its source.

**Display all** OrderNumber, OrderNumberCustomer, OrderDate, CustomerNum, PaymentMethod and InvoiceDate  
**such as InvoiceDate is equal to 'InvoiceDate'**  
*AND ORDERS.Total is greater than or equal to Amount*  
**AND HYSTORY.AdminAssistant is equal to 'AdminAssistant'**  
*OR PRODUCT.MinQty is equal to MinQuantity*  
**AND PaymentMethod contains 'PaymentMethod'**  
*AND ORDERS.Observations contains 'Observation'.*

*A few clicks are enough to create an SQL query, and to check it in natural language, and view it in a diagram.*





*A report created with Reports & Queries*

## REPORTING TOOL ("REPORTS & QUERIES" TOOL)

The "Reports & Queries" tool is a report editor provided with WINDEV and WEBDEV.

It can be freely distributed to your end users, for any application created with WINDEV or WEBDEV.

This report editor interfaces natively with HFSQL, and allows for easy creation of very sophisticated reports using data stored in HFSQL databases (or other databases).

The PDF standard is supported, as well as page background, bar codes, labels, export to Word and Excel ... and everything you need!

## ROLAP CUBE: PIVOT TABLE

Decision makers love it! The Pivot Table control dynamically displays in n dimensions data coming from the crosscheck of different files found in a database.

For example: the volume of sales according to product families, products, regions, over time, with or without details. The end user can expand information, hide it, ...

The pivot table performs the calculations: everything is automatic, no programming is needed to fill it.

Filter by family:	2010				2011			Total
	Quarter 1	Quarter 2	Quarter 3	Quarter 4	Quarter 1	Quarter 2	Quarter 3	
Germany	Dortmund	\$918,341.23	\$1,775,375.78	\$911,603.02	\$234,524.55	\$4,372,297.55	\$1,888,933.45	\$2,757,865.26
	Hamburg	4,821	8,822	11,858	8,162	18,829	18,829	17,788
	Munich	\$17,344.52	\$95,127.85	2,336	\$765,187.17	\$1,317,413.14	\$1,086,734.57	\$250,042.22
		4,450	\$103,555.66	2,336	4,715	27,762	5,884	4,808
England	Liverpool	\$194,362.94	\$102,755.54	\$88,912.64	\$180,503.86	\$1,570,084.88	\$427,894.58	\$587,168.12
	London	5,289	6,362	5,329	6,379	47,838	16,918	16,988
	Manchester	\$131,845.84	1,820	2,558	\$157,980.40	\$1,536,490.13	\$1,126,847.34	\$1,834,098.39
		3,723	2,776,339.39	4,783	1,317	28,861	9,590	7,791
Australia	Perth	\$181,782.92	\$276,339.39	\$104,842.56	\$804,233.46	\$254,881.89	\$525,000.94	\$97,328.29
		810	4,884	8,440	27,810	4,300	16,883	5,281
Canada	Chicoutimi	\$594,791.50	\$214,302.87	\$1,380,989.98	\$1,380,989.98	\$1,377,762.53	\$1,360,004.16	\$2,147,568.74
		12,228	4,818	18,381	18,379	28,517	28,517	23,158
		\$1,076,066.90	\$90,400.86	\$18,548.96	\$2,915,137.10	\$2,428,565.48	\$941,080.64	\$349,081.55
		6,152	5,331	1,074	25,938	16,488	4,370	12,759
		\$52,854.49	\$211,235.13	\$1,044,427.33	\$794,579.34	\$4,365,616.74	\$1,172,143.39	\$2,495,819.36
		7,495	1,286	3,266	32,388	18,209	9,686	12,853
		\$104,434.72	\$123,848.79	\$95,760.75	\$2,115,767.26	\$671,745.36	\$649,089.90	\$521,233.30
		3,886	4,588	5,852	36,423	6,782	12,884	12,885
		\$44,527.08	\$332,854.24	\$1,393,760.26	\$941,852.66	\$5,468,639.36	\$893,644.00	\$1,524,510.10
		876	9,700	9,777	58,079	9,818	28,026	20,128
		\$17,883.45	\$168,430.72	\$63,816.81	\$91,757.52	\$886,290.55	\$143,147.18	\$593,796.28
		5,978	5,340	3,209	25,462	6,425	29,782	5,382
		\$1,868,347.68	\$95,836.68	\$793,422.81	\$3,606,292.42	\$3,798,431.48	\$480,574.35	\$1,428,233.73
		10,440	9,925	9,327	55,484	21,888	18,278	23,575
		\$67,441.35	\$105,156.20	\$131,641.52	\$44,426.52	\$5,526,096.77	\$194,713.30	\$1,411,534.75
		2,179	2,484	2,708	1,081	37,851	5,123	10,882
		\$13,099,871.88	\$12,443,745.78	\$26,349,536.48	\$10,746,267.24	\$743,508,891.26	\$45,841,626.77	\$48,316,884.43
		153,488	738,478	185,223	150,477	7,582,514	492,512	589,293
								348,239
								4,351,339

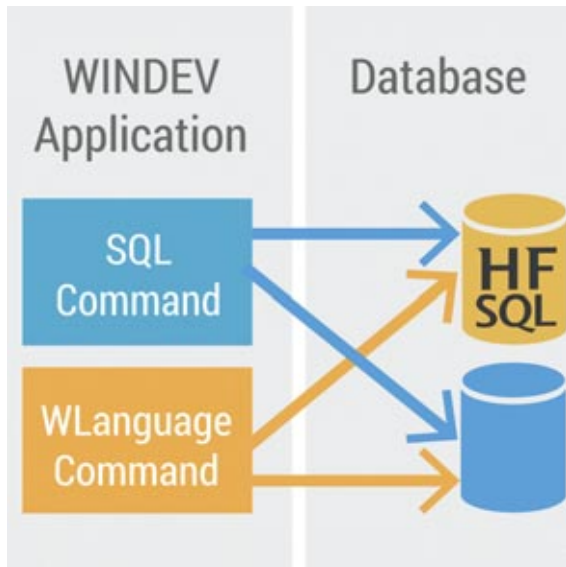
*Example of a cube on HFSQL data*



# HFSQL®



# PROGRAMMING: SQL AND 5GL LANGUAGE



## EASY YET PROWEFUL PROGRAMMING

The programming of the HFSQL database is both powerful and easy.

This programming is done in SQL and/or in WLanguage 5GL.

Programming with the SQL language is universally known.

Programming with the 5<sup>th</sup> generation WLanguage allows for the streamlined and powerful cursor programming..

Automation with the applications and sites developed with WINDEV and WEBDEV are very strong then.

## RAD: TO GENERATE CODE

The code can be generated on demand by WINDEV and WEBDEV by using the RAD functionality, or by using the large number of wizards available for these environments.

The generated code can be modified later.

RAD supports the pattern concept, which lets you define the code to be generated yourself.

## LIST OF SUPPORTED SQL STATEMENTS

Here is the list of supported SQL functions (this list is not exhaustive).

Each SQL function is not examined in details here.

ABS	ACOS	ADD_MONTHS	ALL
ALTER TABLE	AND	ANY	AS
ASC	ASCII	ASIN	ATAN
ATAN2	AUTO_INCREMENT	AVG	BETWEEN
BIGINT	BIN	BINARE	BLOB
BOTH	BOTTOM	BTRIM	CASE
CAST	CBRT	CEIL	CEILING
CHAR_LENGTH	CHARACTER_LENGTH	CI	CLOB
COALESCE	COLLATE	COLUMN	COMMENT
CONCAT	CONSTRAINT	COS	COUNT
CREATE TABLE	CREATE VIEW	CROSS	CURRENT_DATE
CURRENTDATA	DATE	DATETIME	DECIMAL
DECODE	DEFAULT	DEGREES	DELETE
DESC	DISTINCT	DIV	DOUBLE
DROP TABLE	DROP VIEW	ELSE	ELT
END	ESCAPE	EXISTS	EXP
EXPLAIN	EXTRACTVALUE	FIELD	FIND_IN_SET
FLOAT	FLOOR	FOR	FROM
FULL OUTER JOIN	GRANT	GREATEST	

GROUP BY	HAVING	HEX	SI
IFNULL	IN	INDEX	INITCAP
INNER JOIN	INSERT	INSTR	INTO
ISNULL	LAST_DAY	LAST_INSERT_ID	LCASE
LEADING	LEAST	LEFT	LEFT OUTER JOIN
LEN	LENGTH	LIKE	LIMIT
LN	LOG	LOG10	LOWER
LPAD	LTRIM	MATCH AGAINST	MAX
MD5	MID	MIN	MOD
MONEY	MONTHS_BETWEEN	NATURAL	NCLOB
NEW_TIME	NEXT_DAY	NOT	NTEXT
NULL	NUMBER	NUMERIC	NVARCHAR
NVL	OCT	OCTET_LENGTH	Offset
ON	OR	ORDER BY	OVERLAY
PATINDEX	PERCENT	PI	PLACING
POSITION	POWER	PRECISION	RADIANS
RANDOM	REAL	REFRESH VIEW	REGEXP/REGLIKE
RENAME	REPEAT	REPLACE	REVERSE
REVOKE	RIGHT	RIGHT OUTER JOIN	
ROUND	RPAD	RTRIM	SELECT
SET	SHA	SHA1	SIGN
SIN	SMALLINT	SOME	SON
SOUND2	SOUNDEX	SOUNDEX2	SPACE
SPLIT_PART	SQRT	STDDEV	STDDEV_POP
STDDEV_SAMP	STRING_AGG	SUBSTR	SUBSTRING
SUM	SYSDATE	TAN	TEXT
THEN	TIME	TINYINT	TO_CHAR
TO_DATE	TOP	TRAILING	TRANSLATE
TRIM	TRUNC	UCASE	UNHEX
UNICODE	UNION	UNIQUE	UNSIGNED
UPDATE	UPPER	USING	UUID
VALUES	VAR_POP	VAR_SAMP	VARBINARY
VARCHAR	VARIANCE	VARYING	WHEN
WHERE	AVEC	XOR	

The "Select" type SQL queries accept a large number of WLanguage functions as parameters, which lets you easily refine the selection.

The accepted WLanguage functions are:

Abs	Age	CurrentYear	AnsiToOem
AnsiToUnicode	ArcCos	ArcSin	ArcTan
ArcTan2	Round	RoundDown	RoundUp
Asc	BufferToInteger	BufferToHexa	BufferToReal
Charact	CharactTypeOccurrence	CharactType	CharactUnicode
StringStartsWith	StringCompare	StringBuild	StringEndsWith
StringFormat	StringIncrement	StringInsert	StringReverse
StringCount	StringDelete	StringToDate	StringToDuration
StringToUTF8	Complete	CompleteDir	Compress
MakeInteger	Contains	Conversion	Cos
CoTan	ColorLightness	ColorSaturation	ColorHue
Crypt	DateDifference	Today	DateTimeDifference
DateTimeLocalToUTC	SysDateTime	DateTimeUTCtoLocal	DateTimeValid
DateSys	DateValid	DateToString	DateTimeInteger
DateToDay	DateToDayInAlpha	DateToMonthInAlpha	DateToWeekNumber
DecimalToSexagesimal	UncompleteDir	Uncompress	Uncrypt
LastDayOfWeek	LastDayOfMonth	GetGUID	Right
DurationToString	EmailCheckAddress	IntegerToDate	IntegerToTime
IntToHexa	IntegerToDay	IntegerToDayInAlpha	IntegerToMonthInAlpha
IntegerToWeekNumber	IsOdd	IsNumeric	IsEven
BinaryAND	ExeInfo	Exp	ExtractString
ExtractLine	Factorial	fAttributeReadOnly	fLoadBuffer
fLoadText	fShortPath	fLongPath	fCompress
fCompare	fBuildPath	fCopyFile	fCreateLink
fCrypt	fDate	fDateTime	fUncompress
fUncrypt	fMoveFile	fDriveInfo	fExtractPath
fFileExist	fTime	fShortName	fLongName
fDirAttribute	fCopyDir	fMakeDir	fDirAttrib
fDirectoryExist	fTempPath	fParentDir	fRemoveDir
fDirSize	fSaveBuffer	fSaveText	fSep
fSeparator	fDelete	fSize	fSizeUncompressed
Left	GeneratePassword	Random	TimeDifference
TimeSys	TimeValid	TimeToString	TimeToInteger
HexaToBuffer	HexaToInt	HTMLToRGB	HTMLToText
INIWrite	INIRead	InitRandom	Reverse
ExeRun	LineToPosition	Ln	Log
Now	Upper	Max	Middle
Min	Lower	CurrentMonth	WordOccurrence
NetMACAddress	NetIPToMAC	NetMACToIP	NumberInWords
BinaryNOT	NumToString	DayNumberInAlpha	MonthNumberInAlpha
WeekNumber	OemToAnsi	BinaryOR	BinaryXOR
Easter	DecimalPart	IntegerPart	Phonetic
LoWord	HiWord	Position	PositionOccurrence
PositionToLine	FirstDayOfWeek	FirstDayOfMonth	Power
Root	RegistrySeek	RegistryNextKey	RegistryCreateKey
RegistrySetValue	RegistryExist	RegistryListValue	RegistryQueryValue





RegistryFirstSubKey	RegistryDeleteKey	RegistryDeleteValue	RegistryValueType
Replace	RepeatString	NetworkConnect	NetworkDisconnect
NetworkDomainName		NetworkDirName	NetworkUser
RGB			
RGBBlue	RGBRed	RGBToHTML	RGBGreen
NoAccent	NoCharacter	NoRightCharacter	NoLeftCharacter
NoSpace	sComputeCrc16	sComputeCrc32	WeekToDate
SexagesimalToDecimal	Sin	SysColor	SysColorRes
SysEnvironment	SysSpace	SysInstance	SysNameExe
SysDir	SysWindowsVersion	SysXRes	SysYRes
Length	CommonLength	LengthToString	Tangent
TextToRTF	Truncate	HSL	UnicodeToAnsi
URLDecode	URLEncode	URLExtractPath	UTF8ToAnsi
UTF8ToString	UTF8ToUnicode	UUDecode	UUEncode
Val	MatchRegularExpression	WindowsVersion	

## LIST OF WLANGUAGE COMMANDS

The WLanguage commands let you program cursors, or program all the processes you want.

The 5GL WLanguage programming is very powerful and very intuitive.

For example, searching for a row (a record), and all the associated processes (opening the table, assigning data, etc.) is done in one simple, powerful line of code:

**HReadSeek (CUSTOMER, NAME, "DOE")**

The source programs are clear, easy to write and especially easy to maintain; this reduces the chance for errors, and makes the applications you develop more reliable, and faster.

Plain commands makes it easier to use: the code is self-commented!

The **HReadSeek** command can be understood by any developer, even rookies!

Other example, the creation of a table is done in 1 line of code, without any script: **hCreate**.

Non-exhaustive list of WLanguage functions (5GL used by WINDEV, WEBDEV and WINDEV Mobile), with summary of their action.



ScreenToFile	Automatically initializes: - the memory value of the items of a data file with the value of the controls found in the window or in the page. - the value of the WLanguage variables with the value of the controls found in the window or in the page.
ScreenToSource	Automatically initializes: - the memory value of the items of a data file with the value of the controls found in the window or in the page. - the value of the WLanguage variables with the value of the controls found in the window or in the page.
FileToScreen	Automatically initializes the controls found in a window or in a page with: - the values of the associated items in the current record (loaded in memory) of the data file. * the values of the associated WLanguage variables.
SourceToScreen	Automatically initializes the controls found in a window or in a page with: - the values of the associated items in the current record (loaded in memory) of the data file. * the values of the associated WLanguage variables.
WithSpace	Adds or deletes the spaces found on the right of a text item when reading it.
HAccelerateSpeed	Reorganizes the internal structure of the indexes to optimize the speed for accessing the data.
HActivateFilter	Enables the filter that was previously created for the specified data file (view or query).
HActivateAutoFilter	Enables an automatic filter on the linked files when browsing an XML file.
HActivateTrigger	Re-enables a trigger that was disabled by HDeactivateTrigger.
HActivateServerTrigger	Re-enables a server trigger that was previously disabled by HDeactivateServerTrigger.
HAdd	Adds the record found in memory into the data file (query or

	view).
HAddGroup	Adds a group of users.
HAddLink	Adds an integrity rule between two files on the server.
HAddScheduledOptimization	Adds an optimization task of HFSQL Client/Server data files.
HAddScheduledBackup	Adds a scheduling for full backup (with or without differential backup) on the server defined by the connection.
HAddTask	Adds a scheduled task on the server defined by the connection.
HAddUser	Adds a user to a database.
HAlias	Creates a logical alias of a data file (or query) or cancels all the existing aliases.
HCancelAlias	Cancels an alias that was previously declared by HAlias.
HCancelDeclaration	Deletes a declaration performed by HDeclare, HDeclareExternal or HDescribeFile
HCancelSeek	Cancels the current search criterion.
HCancelBackup	Cancels a current backup.
HStopServer	Stops a HFSQL server.
HLinkMemo	Used to associate a file with a binary memo item or to cancel the existing link between a file and a binary item.
HForward	Moves several records forward from the current position in the data file, according to a specified item.
HLockFile	Locks a data file and restricts the access to this data file for all the other sites or applications.
HLockRecNum	Locks a record and restricts the access to this record for all the other applications.
HChangeKey	Changes the search key.
HChangeConnection	Dynamically changes the connection associated with a data file.
HChangeLocation	Modifies the search mode of data files.
HChangePassword	Changes the password of a HFSQL Client/Server data file.
HChangeName	Modifies the physical name of a data file.
HChangeDir	Modifies the access path to a data file (which means the directory in which the file will be handled).
HChangeLogDir	Modifies the location of the log files corresponding to a HFSQL data file.
HLoadParameter	Reads a parameter that was saved from a stored procedure by HSaveParameter.
HClusterAddNode	Enables a node in a HFSQL cluster.
HClusterStop	Suspends the execution of a HFSQL cluster.
HClusterStart	Starts a HFSQL cluster.
HClusterState	Returns the status of a HFSQL cluster by interrogating its coordinator
HClusterIgnoreSynchro	Defines a node of the HFSQL cluster as data source to perform the cluster synchronization.
HClusterNodeInfo	Returns the status of each cluster node by interrogating the coordinator.
HClusterParameter	Reads and modifies the parameters of a HFSQL cluster.
HClusterDeleteNode	Disables a node in a HFSQL cluster.
HConnect	Redefines one or more parameters of a connection to a specific table or to a set of tables.
HConnectRemoteAccess	Opens an analysis in HFSQL Classic format via a remote access
HBuildKeyValue	Builds the value of a composite key to create a filter or to perform a search.
HBuildKeyValueANSI	On a Unicode platform, used to build the value of a composite key.
HConvert	Converts a numeric value into a binary string in order to perform a search on a numeric key.
HCopRecord	Copies the content of the current record (loaded in memory) into the current record of a data file.
HCopFile	Copies a HFSQL file.
HCreation	Creates an empty data file with the index file and the memo file if necessary.
HCreationIfNotFound	Creates an empty data file (if the file does not exist) or opens a data file (if the file exists).
HCreateServerTrigger	Adds or modifies a server trigger on the HFSQL server.
HCreateView	Creates a HFSQL view.
HRecordDate	Returns the date and time of the last write operation performed on a record found in a HFSQL file.
HDBCcreation	Ends the description of the structure of the xBase data file by programming.
HDBDescribeFile	Describes by programming a file in dBase3 format.
HDBDescribeIndex	Describes by programming the different index files that will be created.
HDBDescribeField	Describes by programming each item of the structure of an xBase file.
HDBIndex	Opens an xBase index file.
HDBOpen	Opens the xBase data file and the "memo" file if it exists.
HDBOpenNoLock	In single-user mode, opens an xBase data file without locking it.
HDBSortType	Returns or modifies the sequence of text items in the xBase files.



HUnlockFile	Unlocks the records of a data file.	HSetRemoteAccess	Temporarily disables the remote access in order to access HFSQL Classic data files found locally.
HUnlockRecNum	Unlocks a record.	HSetCache	Allows you to configure the management of caches in the HFSQL Client/Server engine.
HDeclare	Declares a description of data file (found in an analysis) in the current project.	HSetDuplicates	Enables or disables the management of duplicates on a unique key
HDeclareExternal	Temporarily imports into the current analysis the description of a file from an existing HFSQL file.	HSetIntegrity	Enables or disables the management of an integrity constraint on a file link.
HDisconnectClient	Displays a message on the client computers and disconnects the application.	HSetLog	Enables or disables the management of the log for a logged file.
HDescribeConnection	Describes a new connection to an external database.	HSetMemo	Used to modify the management mode of memo items.
HDescribeFile	Describes a data file by programming.	HSetREP	Enables or disables the management of .REP file.
HDescribeFullTextIndex	Describes a full-text index of a data file created by programming.	HSetServer	Used to find out and modify some settings of the HFSQL Client/Server server.
HDescribeLink	Describes a link between two files by programming	HManageTask	Enables or disables a scheduled task on a HFSQL Client/Server server.
HDescribeItem	Describes by programming an item found in a data file.	HSetTransaction	Enables or disables the management of transactions for one or more files.
HDescribeTrigger	Adds or modifies a trigger on a HFSQL data file.	HSetTrigger	Enables or disables the management of triggers.
HDescribeServerTrigger	Adds or modifies a server trigger.	HHistoryModification	Returns the modifications made to one or more items of a given record.
HStartServer	Used to start a HFSQL server (uses MantaManager).	HImportHF55	Imports a Hyper File 5.5 file into a file in HFSQL Classic format.
HLast	Positions on the last record of a data file according to a search item.	HImportText	Imports a Text file into a data file in HFSQL Classic format.
HDeactivateFilter	Temporarily disables the filter on a data file (view or query).	HImportXML	Imports an XML file into a file in HFSQL Classic format
HDeactivateAutoFilter	Disables an automatic filter on the linked files when browsing an XML file.	HInfoAnalysis	Returns information about an analysis (WDD file).
HDeactivateTrigger	Disables a trigger.	HInfoLock	Returns information about the lock performed on a data file, on a record or on all the records found in a data file.
HDeactivateServerTrigger	Disables a HFSQL Client/Server server trigger on a server.	HInfoDatabaseRights	Allows you to find out the rights granted to a user or to a group on a database.
HDeleteTrigger	Destroys a trigger.	HInfoFileRights	Allows you to find out the rights granted to a user or to a group on a HFSQL Client/Server data file.
HDeleteServerTrigger	Destroys a server trigger.	HInfoServerRights	Allows you to find out the rights granted to a user or to a group on a server.
HDeleteView	Destroys a view that was created beforehand.	HInfoFile	Returns the characteristics of a file found on a HFSQL server.
HDuplicateRecord	Duplicates the record read in a data file: the record found in memory is added into the data file (query or view).	HInfoGroup	Returns information about the specified group of users.
HWrite	Writes a record into the data file without updating the corresponding indexes.	HInfoLog	Returns information about the server logs.
HOut	Allows you to find out whether the record on which we want to be positioned is located outside the data file, filter, view or query.	HInfoMemo	Returns the characteristics of binary and text memos.
HRecordToXML	Retrieves the structure and the value of the current record and exports them into a character string in XML format.	HInfoDatabaseProperty	Allows you to find out the properties of a database found on a HFSQL server.
HSendMessageToClient	Displays a message on the client computers.	HInfoFileProperty	Used to find out the properties of a data file found on a HFSQL server.
HError	Returns the number of the last error triggered by the HFSQL engine.	HInfoServerProperty	Allows you to find out the properties of a HFSQL server.
HErrorLock	Allows you to find out whether a lock error occurred.	HInfoBackup	Returns information about one or more backups performed on a HFSQL Client/Server server.
HErrorDuplicates	Allows you to find out whether a duplicate error occurred.	HInfoServer	Returns the specified information about the server.
HErrorStatusModification	Returns the status of a record during a modification conflict	HInfoTask	Returns the characteristics of a scheduled task.
HErrorInfo	Returns a detailed information about the last error triggered by the HFSQL engine.	HInfoUser	Updates the variables for user management with the information about the specified user.
HErrorIntegrity	Used to find out whether an integrity error occurred.	HNoDatabaseAccess	Forbids all the accesses to a database or to all the databases.
HErrorModification	During a modification conflict, returns the value of a record item.	HNoModif	Prevents from modifying a file (for all the programs, including the program that requested the restriction)
HErrorPassword	Allows you to find out whether a password error occurred on this data file.	HLogInfo	Adds comments into the log when saving the logged operation.
HState	Used to find out the status of a record.	HLogRecreate	Used to re-create an empty log.
HServerStatus	Used to find out the status of a HFSQL server.	HLogRestart	Restarts the log process on a file.
HExecuteProcedure	Runs a stored procedure.	HLogStop	Stops the log process of a file.
HExecuteQuery	Declares a query created in the query editor to the HFSQL engine and runs this query.	HFree	Transforms the crossed records of a data file into deleted records.
HExecuteSQLQuery	Initializes a query written in SQL language and declares this query to the HFSQL engine.	HFreePosition	Deletes a position saved by HSavePosition
HExecuteScheduledBackup	Forces the execution of a scheduled backup.	HFreeQuery	Frees the resources of a query.
HExecuteView	Runs a view that was created beforehand.	HListAnalysis	Lists the analyses in HFSQL Classic format available in a given directory.
HExportXML	Exports the records from a file (HFSQL or OLE DB), view or query into an XML file.	HListDatabase	Lists the Client/Server databases associated with a connection.
HExtractMemo	Extracts the content of a blob (binary memo) item from a physical file.	HListKey	Lists the keys of a file (a query or a view) recognized by the HFSQL engine.
HClose	Closes a data file or all the opened data files.	HListConnection	Lists the connections currently described in the application.
HCloseAnalysis	Closes the current analysis.	HListStoredElement	Lists the elements stored on a HFSQL server (sets of procedures, stored procedures or queries).
HCloseConnection	Closes a connection to a database.	HListFile	Lists the files in the current analysis or in a specific analysis recognized by the HFSQL engine.
HFileExist	Allows you to find out whether a file exists, or whether a view or a query has been defined.	HListGroup	Lists the user groups defined for a connection.
HFilter	Defines and enables a filter on a data file, a view or a query.	HListFullTextIndex	Lists the full-text indexes of a file (a query or a view) recognized by the HFSQL engine.
HFilterStartsWith	Defines and enables a "Start With" filter on a file, view or query.	HListLink	Lists the links found in the current analysis or in a specific analysis.
HFilterIncludedBetween	Defines and enables an "Included between" filter on a file, view or query.	HListScheduledOptimization	Lists the scheduled optimization tasks of the HFSQL Client/Server data files for a connection.
HFilterContains	Defines and enables a "Contains" filter on a data file, a view or a query.	HListParameter	Lists the parameters saved from the procedures stored on the server.
HFilterIdentical	Defines and enables a filter used to find the exact value of a string item.	HListQueryParameter	Lists the parameters of a query created in the query editor
HEndNoDatabaseAccess	Re-authorizes the access to one or more databases accessible by a connection.	HListCustomFolder	Lists the custom-folders defined in the analysis.
HEndNoModif	Unlocks a file locked by the same program with HNoModif.	HListProvider	Lists the OLE DB providers and/or Native Access installed on
HFlush	Forces the operating system of the computer where the data files are found to write the data onto the disk.		
HMergeView	Creates a HFSQL view from two views created beforehand		





HListREP	the computer. Lists the assignments for the data files used by the current application.	HSetPosition	Positions on a record from the approximate position of one of its items.
HListItem	List the items in a file (a query or a view) recognized by the HFSQL engine.	HPost	Stores a unique computer number or identifier in order to use the logs and the transactions.
HListScheduledBackup	Lists the full and differential backups that have been scheduled on a HFSQL Client/Server server.	HPrevious	Positions on the previous record found in a file according to a search item.
HListServer	Lists the HFSQL servers installed on a computer.	HFirst	Positions on the first record of a data file according to the specified search item.
HListTask	Returns the list of scheduled tasks found on a HFSQL Client/Server server for a given connection.	HPrepareQuery	Initializes a query and declares this query to the database server in order to optimize the next executions of this query.
HListTrigger	Lists the triggers applied to one or more HFSQL data files.	HPrepareSQLQuery	Initializes a query written in SQL and declares this query to the database server in order to optimize the next executions of this query.
HListServerTrigger	Lists the different triggers available on a connection or on one of the connection files.	HPriority	Allows you to find out and modify the priority of the calling application.
HListUser	Lists the users defined for a connection.	HPriorityClient	Modifies the priority of a client application.
HListConnectedUser	Lists the users currently connected to one or more files handled by a Client/Server connection.	HClearWorkingDir	Clears and destroys the temporary directory previously created during the execution of HServerWorkingDir.
HRead	Reads a record in a file according to a given record number.	HConnectionQuality	Returns the quality level of the connection: the higher the level is, the faster the connection will be.
HReadLast	Positions on the last file record according to a search item.	HCross	Crosses a record in a data file.
HReadPrevious	Positions on the previous file record according to a search item.	HReset	Initializes one or more variables of the items found in a data file with their default values.
HReadFirst	Positions on the first record of a file according to a search item.	HResetClient	Initializes the structure for managing the client computers (HClient structure)
HReadSeek	Positions on the first file record whose value for a specific item greater than or equal to a sought value (generic search).	HResetGroup	Initializes the structure for group management with the default values.
HReadSeekLast	Positions on the last file record whose value for a specific item is less than or equal to a sought value (exact-match search).	HResetUser	Initializes the structure for user management with the default values.
HReadSeekFirst	Positions on the first file record whose value for a specific item is strictly equal to a sought value (exact-match search).	HSeek	Points to the first file record whose value for a specific item is greater than or equal to a sought value (generic search by default).
HReadNext	Positions on the next record of a data file according to a search item.	HSeekLast	Positions on the last file record whose value for a specific item is less than or equal to a sought value.
HMigrateLinkedCompositeKey	Migrates the values of the linked composite keys coming from a file in Hyper File 5.5 format to the HFSQL Classic format.	HSeekFirst	Positions on the first file record whose value for a specific item is greater than or equal to a sought value.
HRefreshSet	Creates or refreshes a set of procedures on a server	HReconnect	Establishes a reconnection to the server for all the interrupted connections.
HRefreshQuery	Creates or refreshes a query on a HFSQL server.	HBackward	Moves backward several records from the current position in the data file, according to a specified item.
HMode	Changes the lock mode for the data files.	HRetrieveRecord	Returns the content of the current record (in a file, a view or a query, ...).
HModify	Modifies the specified record or the record found in memory in the data file (the query or the view).	HRetrieveLog	Creates a text file containing the server logs between two given dates.
HModifyDatabaseRights	Modifies the rights granted to a user or to a group for a HFSQL Client/Server database.	HRetrieveItem	Returns the content of an item found in the current record (in the data file, view, query, ...).
HModifyFileRights	Modifies the rights granted to a user or to a group on a HFSQL Client/Server data file.	HRegenerateFile	Regenerates a data file from its log.
HModifyServerRights	Modifies the rights granted to a user or to a group on a HFSQL server.	HIndexingInProgress	Indicates that a data file is currently re-indexed and returns the percentage of the file already re-indexed.
HModifyGroup	Modifies the group information according to the elements found in the corresponding variables for group management.	HIndex	Rebuilds the index of a data file
HModifyScheduledOptimization	Modifies a scheduled optimization task on the HFSQL server defined by the connection.	HServerWorkingDir	Returns the path of a temporary directory on the server.
HModifyDatabaseProperty	Modifies the properties of a database found on a HFSQL server.	HRestoreBackup	Restores a backup performed by the HBackup function or via the HFSQL Control Center
HModifyFileProperty	Modifies the properties of a HFSQL file found on a server.	HRestorePosition	Restores the context of a previously saved data file.
HModifyServerProperty	Modifies the properties of a HFSQL server.	HRSAddConfig	Adds a replication between two HFSQL server onto the master server.
HModifyScheduledBackup	Modifies a backup scheduling.	HRSExecute	Immediately runs a recurring replication between HFSQL servers: the replication is triggered before the scheduling
HModifyStructure	Updates the structure of a HFSQL data file by performing a data synchronization.	HSRInfo	Allows you to read the configuration of the replication for a HFSQL server taking part in one or more replications.
HModifyTask	Modifies a scheduled task on the HFSQL server defined by the connection.	HSRInit	Configures a HFSQL server in order for this server to be a master server or a subscriber server for a replication between HFSQL servers.
HModifyUser	Modifies the user information according to the elements found in the corresponding variables for user management.	HSRListConfig	Lists the replications available on a master HFSQL server.
HNbRec	Returns the number of records in a file, a query or a HFSQL view: active records, crossed records, deleted records, etc.	HSRModifyConfig	Modifies some parameters of an existing replication between two HFSQL servers.
HNotifAddCCRRecipient	Add recipients for the notifications sent via the Control Centers.	HSRDeleteConfig	Deletes a replication between two HFSQL servers.
HNotifAddEmailRecipient	Adds recipients for the notifications sent by email.	HBackup	Saves the content of a HFSQL server.
HNotifConfigure	Specifies and configures the server used to send notifications by the HFSQL server.	HSaveParameter	Saves a persistent value from a stored procedure.
HNotifListCCRRecipient	Returns the list of recipients for a notification sent via the Control Centers.	HSavePosition	Stores the current context of a data file.
HNotifListEmailRecipient	Returns the list of recipients for a notification by email.	HSecurity	Enables or disables the security mechanism..
HNotifDeleteCCRRecipient	Deletes the recipients of a notification sent via the Control Centers.	HSimulateNetwork	Simulates the operating mode of HFSQL Client/Server on an ADSL or 3G network.
HNotifDeleteEmailRecipient	Deletes the recipients of a notification by email.	HStatCalculate	Performs statistical calculations on the keys of a file.
HRecNum	Returns the number of the current record in the HFSQL data file or in the HFSQL view.	HStatDate	Returns the date of the last update for the index statistics
HOptimize	Uses idle periods (period without processing) to optimize the queries and the read operations that will be run thereafter.	HStatTime	Returns the time of the last update for the index statistics
HOptimizeQuery	Optimizes the selection queries by using idle times (period without processing)	HStatNbDuplicates	Returns the number of duplicates for a given key item.
HOpen	Opens a data file.	HStatNbRec	Returns the number of entries for a given key item.
HOpenAnalysis	Opens an analysis in HFSQL Classic format.	HStatNbRecRange	Returns an estimate regarding the number of entries for a given key item in a given interval of values.
HOpenConnection	Establishes a connection to a specific database.	HSubstDir	Associates the data directory specified in the analysis with a directory found on disk.
HPass	Defines the password used to create or open a data file.	HNext	Positions on the next record of a data file according to a search
HGetCurrentPosition	Returns the approximate position of the current record in the data file.		



# VOCABULARY

The vocabulary varies based on the interlocutors.  
The same concept is often described using different words.  
Each person has his own standard, habits!  
Here's a small glossary of terms used around databases.

<i>PC SOFT vocabulary</i>	<i>Other publishers</i>
Analysis (CDM, LDM)	Schema, relational model, entity/relationship model
Analysis chart	Data schema
File	Table
Item	Column, Field (the control is the intersection of a column and a row)
Record	Line, tuple, row
Link	Relationship
Lock	Lock
Key	Index
Unique key	Primary key
Key with duplicates	Foreign key or key without uniqueness constraint
Link Item	Foreign key
Window	Form
Control	Check
Report	Report
Viewing table	Datagrid, Browse
Scheduled task	Scheduler
Text memo	Lob or clob
Binary memo	Lob or blob
Stored procedure	UDF (User Defined Function)

A **database** is a set of tables (files) linked via relationships (links).

A **table (data files)** is a set of data organized in columns (items), made of rows (records) The intersection of a row and a column is a field (item value).

An **index** is a way to accelerate searches, queries and accesses to a table.

An index can be defined on a column (key item) or on several columns (composite key).

A **primary key** is a **unique** key that can't be null.

A **foreign key** is a key that accepts **duplicates**, used jointly with a primary key to establish a relationship (link) between 2 tables.

	item.
HDelete	Deletes a record from a data file (a query or a view).
HDeleteDatabase	Deletes a database found on a HFSQL server.
HDeleteSet	Deletes a set of stored procedures from a HFSQL server.
HDeleteFile	Deletes the HFSQL data files (.fic, .ndx, .ftx and .mmo files if they exist) from the server.
HDeleteGroup	Deletes (from the server) a group of users associated with a connection.
HDeleteLink	Delete an integrity rule between two data files on the server.
HDeleteScheduledOptimization	Deletes a scheduled optimization task of HFSQL Client/Server data files.
HDeleteParameter	Deletes a parameter that was previously saved by HSaveParameter.
HDeleteDirectory	Deletes a directory found in a HFSQL Client/Server database.
HDeleteQuery	Deletes a query (used by stored procedures) from a HFSQL server.
HDeleteBackup	Deletes a backup that was performed by HBackup.
HDeleteScheduledBackup	Deletes a scheduling for backup from a HFSQL Client/Server server.
HDeleteTask	Deletes a scheduled task from a HFSQL Client/Server server.
HDeleteAll	Deletes all the records from a data file, from a HFSQL view or from a query.
HDeleteUser	Deletes (from the server) a user associated with a connection
HOnServerCall	Customizes the management of message display on the client computer and the management of disconnection from a client computer.
HOnError	Customizes the management of HFSQL errors.
HTransactionCancel	If a transaction is in progress, cancels all the operations performed on the data files in transaction since the beginning of transaction.
HTransactionStart	Starts a transaction on the HFSQL files and creates the transaction file.
HTransactionEnd	Validates the current transaction.
HTransactionInterrupted	Allows you to find out whether a transaction was interrupted (the transaction was neither validated nor canceled).
HTransactionIsolation	Configures the transaction isolation level for a connection to a given HFSQL server.
HTransactionFree	Transforms all the records "in transaction" into "normal" records if these records do not belong to a transaction currently in progress.
HTransactionList	Returns the list of current or interrupted transactions found on the server for the specified connection.
HSortView	Sorts a view by creating an index on a view item.
HFound	Checks whether the current record corresponds to the current filter or to the current search.
HCheckIndex	Checks whether the data found in the index file (.NDX file) properly refers the data found in the data file (.FIC file).
HCheckStructure	Defines the mode for comparing the data files.
HToFile	Copies a data source (file, query, view, ...) to a physical HFSQL file with the same description. This file is neither encrypted nor password protected.
HVersion	Allows you to find out whether the file's content was modified.
HToItem	Assigns the specified value to an item of the current record.
HViewToFile	Saves the modifications made to a view in the corresponding file.

## WHICH COMPANIES USE HFSQL?

Several million copies of HFSQL are deployed in over 100 countries.

HFSQL is deployed on the most demanding web sites (Web, telecoms, enterprises, banks, hospitals, research, software publishers, administrators, government, etc.) that require high availability (24/7) with top performance in real time.

## TESTIMONIALS

Here are some testimonial:

***"HFSQL: light speed!"***

***"HFSQL completely delivers in terms of robustness and flexibility"***

***"HFSQL allows us to save several hundred millions of euros thanks to the fact that we don't need individual licenses for the database."***

***"This represents close to a billion operations hosted and processed by HFSQL corresponding to about 24 billions euros in debit operations"***

***"All the applications rely on the HFSQL database to ensure complete data security"***

***"The data is stored on our dedicated server with an HFSQL database that supports our entire Information Services"***

***"In term of performance, HFSQL delivers. It's always instantaneous"***

***"We're managing more than one TB of data (with HFSQL) and we're thrilled with the database performance".***

You'll also find technical video and testimonial videos on the [www.windev.com](http://www.windev.com) site.

## LIST OF HFSQL BENEFITS

**Here's a summary of the benefits linked to adopting HFSQL:**

- Feature rich
- Free (complete, unlimited version with all the tools)
- Data schema description tool
- Easy to install
- Easy to embed
- Easy administration (auto-administered, auto-optimized)
- Powerful administration tools
- Tight integration with PC SOFT's best sellers: WINDEV, WEBDEV, WINDEV Mobile
- All-in-one solution with WINDEV and WEBDEV: RAD oriented, it generates the tables, processes, windows and reports
- Encryption of the data, tables and indexes
- Encryption at the column, backup, network traffic level
- Compatibility: Windows (10, 8, 7, Vista, XP, Mobile, CE...), Linux, Mac, iOS (iPhone, iPad), Android...
- Binary compatibility of the databases and indexes: local, network, mobile, embedded, client/server, cluster
- Stored procedures
- Hot and incremental backups
- Protection against SQL code injection
- Unicode
- Efficient language and character set management as well as sort order, granularity at the column level
- Easy replication
- Automatic reconnection
- Easy monitoring
- High availability cluster
- Performances
- Sustainability
- Audit, tuning functions
- Optimizing the queries
- Full Text indexing support
- Blob, Lob
- Integrity constraints
- Automatic schema (DDS) maintenance , on an unlimited number of deployed databases
- Robustness for large volumes of data
- Low resource requirements
- Secure access
- Automatic load distribution among clients
- Ease of deployment and use
- Free technical support\*

HFSQL® is included for free with the WINDEV, WEBDEV and WINDEV Mobile IDE. HFSQL is optimized to run with these IDE. HFSQL is the new name of HyperFileSQL. The use and distribution of the HFSQL database is free with applications and sites created using these IDE, regardless of the quantity deployed and the type of applications (educational applications, personal applications and commercial applications). There's no royalty to pay, no reporting to do.

The deployment is free and unlimited. Please refer to the term of the user license agreement for any additional information. The ODBC driver and the OLE DB provider are freely distributable with your applications created with WINDEV, WEBDEV or WINDEV Mobile. The tools mentioned in this documentation come with the product. All trademarks are properties of their respective owners

WINDEV, WEBDEV and WINDEV Mobile are professional software. Despite the care taken in creating this document, it is not contractual. The screen shots and the lists are given for information purposes only. Don't hesitate to contact us if you need any additional information or to get confirmation of a feature. **Environmental policies:** when PC SOFT prints "paper" documents, PC SOFT, the

paper supplier or the printer when it is FSC - Forest Stewardship Council - certified and PEFC - Program for the Endorsement of Forest Certification - certified, replants as many trees as used for the printing. The FSC label was created by the FSC NGO, which includes among others Greenpeace, Friends of the Earth and the WWF. For example printing 100, 000 copies of a 68-page documentation on glossy paper consumes 10 trees: PC SOFT has 10 trees re-

planted immediately. Also, we favor pulp coming mainly from recycled wood (from furniture mills for instance) and from controlled forest clearing.





# **HFSQL<sup>®</sup>**

***HFSQL is the best ally for your data!***

*You too, like millions of users across the world, take advantage of the performance and security of HFSQL.*



# HFSQL<sup>®</sup>

Since 1988

**PERFORMANCE, SECURITY, AVAILABILITY**



## **RDBMS**

Windows, Linux, Mac, Android, iOS (iPhone, iPad)  
Client/Server, Cluster, Cloud, Standalone, Mobile, Embedded



[www.windev.com](http://www.windev.com)