# DQpowersuite®

DECISION SUPPORT INC
Intelligent Reporting Solutions

## Superior Architecture

Since its first release in 1995, DQ*powersuite* has made it easy to access and join distributed enterprise data. DQ*powersuite* provides an easy-to-implement architecture that makes all of a company's business data look and act like it is contained in a single relational database — including non-relational legacy data.

DQ*powersuite* frees organizations from the time consuming and resource intensive task of bringing data together from many different sources to answer important business questions. Instead, users can quickly and easily view, modify and join data regardless of location or platform. DQ*broker* avoids a significant pitfall encountered with other data integration solutions by accessing most major RDBMS using their native interfaces.

DQ*powersuite* is built from the ground up to distribute query processing as close to source data as possible, to share the burden of processing queries between multiple servers, and to significantly reduce the amount of data that crosses the network. Other solutions use a three-tiered approach where a hub server processes all query logic and large quantities of unnecessary data cross the network.

### A Complete Data Integration Package

DQ*powersuite* simplifies data access, extraction, transformation, and loading in mixed data environments. DQ*powersuite* is ideal for meeting the challenges of a wide variety of data integration projects, including e-commerce applications, enterprise reporting, data warehouses, data marts, and integration necessitated by mergers & acquisitions. DQ*powersuite* consists of the following fully integrated functional components that together provide unprecedented database-level integration:

**DQ*broker*®** — a distributed data access server that makes an entire mixed data environment look and act like a single relational database. DQ*broker* efficiently and effectively accesses, joins, and updates distributed data in real time.

**DQ*transform*®**— using DQ*broker* as a foundation, DQ*transform* allows the automated extraction, transformation, and loading (ETL) of data from multiple sources simultaneously.

**DQ*powerlibs*®** — a software development kit (SDK), which includes APIs and libraries that allow the distributed data access and extraction, transformation and loading (ETL) capabilities of DQ*broker* and DQ*transform* to be embedded into custom applications.

**DQ*view*®** — an add-on product that uses DQ*broker's* access engine to provide enterprise access within Microsoft Excel®. Working as an Excel plug-in, DQ*view* allows queries to be created and embedded within the spreadsheet.

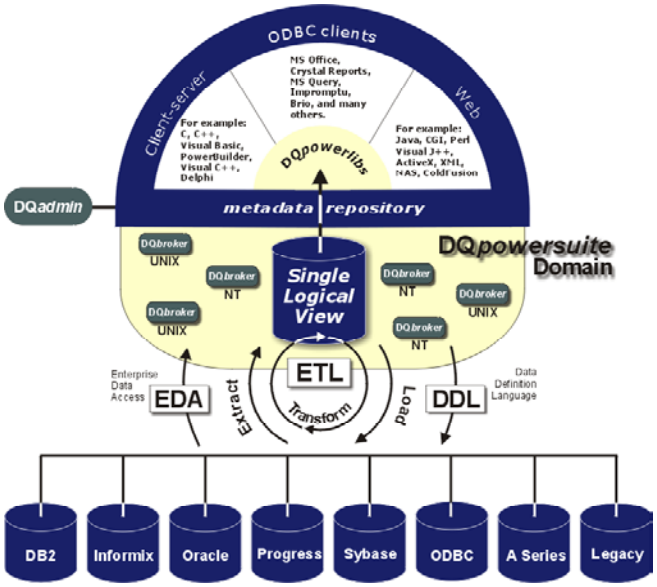


Figure 1: **DQpowersuite Components, Features, and Capabilities**

## Superior Architecture

DQ*powersuite* allows users, developers, and applications to connect to any DQ*broker* server in an installation to obtain the same universal view of distributed relational and non-relational data. This is accomplished by the effective use of metadata and the maintenance of a *global* configuration repository.

DQ*powersuite* is engineered from the ground up to process queries that join data from various data sources, on different platforms, across geographic locations, as efficiently and effectively as possible.

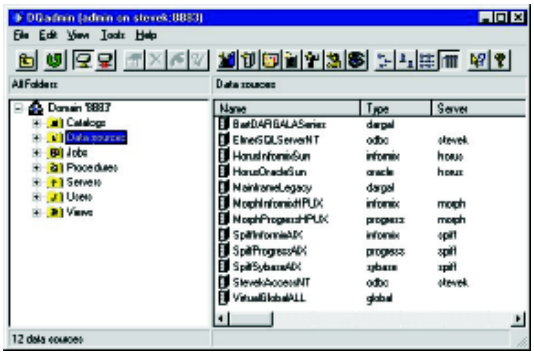

**Figure 2:** DQbroker's Administration GUI presents all data sources to users, developers, and applications as a single logical database — regardless of hardware or software vendor.

## Global Configuration Repository

The global repository is a logical container that holds *all* of the distributed DQ*powersuite* configuration information. It includes information about *all* of the servers and databases in the installation, access permissions for users, the definitions of global views, and the configuration of automated ETL processes.

Each DQ*broker* data access server in an installation of DQ*powersuite* manages an instance of the global repository. These instances are aware of each other and cooperate to maintain complete consistency between each instance on each DQ*broker* server.

This global repository of configuration information allows the entire DQ*powersuite* domain to look exactly the same to users, developers, and administrators no matter where they connect to it.

## Effective Metadata Management

All DQ*broker* servers in a DQ*powersuite* installation work together to maintain a global cache of information about the data that is stored in all available databases. This data about data is called metadata, and consists of information about what tables are in a database, what fields are in the tables, what type of data the fields store, and what indices exist, for example.

A global metadata cache is valuable because it enables every DQ*broker* server to know the current state of *all* data in a distributed DQ*powersuite* domain. Caching metadata also makes the retrieval of database properties faster. This accelerates query processing in a distributed environment because DQ*powersuite* has all the information it needs to validate a query.

If required metadata is not already available in the cache or the existing metadata has expired, the DQ*broker* server obtains it dynamically. This is important, because before a query can be performed, the server must know where the data needed to resolve the query resides and that it refers to valid information.

Metadata caching can be enabled or disabled for all DQ*broker* servers in the DQ*powersuite* domain. Metadata can be configured to expire periodically. Expired metadata is refreshed on a data source-by-data source basis the next time it is requested. Metadata can also be periodically refreshed on a daily, weekly or monthly basis. Periodically refreshed metadata is updated regardless of whether or not it has expired. Metadata can also be manually purged or refreshed for the entire system or for an individual DQ*powersuite* data source.

Effective configuration and administration of the metadata cache is integral to the performance and accuracy of a DQ*powersuite* installation.

## n-tiered and Peer-To-Peer

The global configuration repository and global metadata cache work well because the relationship between DQ*broker* servers is *peer-to-peer*. This n-tiered architecture is more scalable and flexible than the rigid, three-tiered, client-server approaches to distributed data access common in the marketplace today.

In a DQ*powersuite* installation, each DQ*broker* server communicates with every other DQ*broker* server in an installation as an *equal*. Changes to the metadata cache or global configuration repository on one DQ*broker* server are propagated automatically to all DQ*broker* servers in an installation. This provides greater flexibility to users. A connection to any one DQ*broker* server can interact transparently with all other DQ*broker* servers in the installation. End-users get access to all the data they need without having to change the way they work or learn new tools.

While DQ*powersuite* is engineered to be capable of distributing query processes to n-tiers (unlimited), most frequently a distributed query results in processing across four-tiers. In this scenario, the fourth tier often consists of multiple DQ*broker* servers.

**For example, a client machine (1st tier) connects to a DQbroker server (2nd tier) and submits an SQL query that needs data from three different data sources. One of these is local to the DQbroker server that received the request, but two are not. In this case, the DQbroker server at the 2nd tier breaks the query up, processes the part that needs local data, and sends each of the other two DQbroker servers (3rd tiers) the parts of the query that require data local to them. Each 3rd tier DQbroker then sends as much of the SQL as possible to their local database server (4th tier) and processes the rest. The results of the query pieces are piped back up the chain and reassembled and the result set is streamed to the requesting client.**

## True Distributed Query Processing

DQ*powersuite's* n-tier, thin-client architecture distributes the processing of queries as close to source data as possible. Queries that access and join data from multiple heterogeneous data sources are processed on multiple servers simultaneously.

DQ*powersuite* dynamically adapts to the capabilities of each different data source. If a database server is capable of performing an operation required by a query, the query is passed to the database to be processed. A database server knows best how to process its own data. However, DQ*broker* handles any SQL constructs that can't be processed by a particular DBMS.

DQ*powersuite* passes only the data *necessary* to resolve a query across the network, reducing network traffic. Other three-tier solutions can be inefficient and ineffective because they require entire data sets to move across the network to be processed by a single hub server.

## Easy Installation and Configuration

The initial installation and configuration of DQ*powersuite* can be accomplished in an hour or less. It's that easy! Other software that tries to handle distributed queries is so complicated it requires a week of training just to learn how to install and configure it.

Basic installation and configuration consists of installing the software servers and configuring the data sources. With DQ*powersuite*, adding data sources to the environment is easy. It is a simple matter of telling DQ*powersuite* what kind of data source it is (e.g., Oracle, Sybase, DB2), where it is located, and providing some basic configuration information. Once defined, DQ*powersuite* automatically obtains the data sources metadata and it *immediately* becomes a part of the single virtual database provided by DQ*powersuite* to users and developers.

A DQ*powersuite* installation is logically configured as a *single* distributed unit that can consist of any number of DQ*broker* data access servers. Configuration information is automatically propagated to each server in the installation once it has been properly installed. The global configuration repository makes it very easy to add more servers, data sources, and users to an installation of DQ*powersuite*.

Because of the global configuration repository, configuration information can be entered or edited via any DQ*broker* server in an installation. Users, developers, and administrators will always see the same environment regardless of their point of access.

## Try Before You Buy

DQ*powersuite* includes a simple GUI interface — the SQL Builder — that makes building global views easy. Global views can include columns of data from any available data source.

## Native Access

DQ*powersuite* talks to Oracle, Informix, Sybase, Progress, and DB2 databases in their native languages. This allows processing to happen much more efficiently than ODBC-based solutions, because *native is faster*.

## Flexible and Scalable Architecture

DQ*broker* was engineered using object-oriented design principles to have a flexible and scaleable architecture. Two important pieces of this architecture are DQ*broker's* multi-processing and multi-threading capabilities.

Each query submitted to a DQ*broker* server by a user, developer, application or another DQ*broker* server is managed by a separate instance of the query processing application. This means that a DQ*broker* server can process multiple queries simultaneously.

Each instance of the query processing application has a variety of jobs it can perform, and each of those jobs is managed as a separate thread in the application. Multi-threading permits DQ*broker* to process queries asynchronously.

## Local Control of Access and Security

Each DBMS controls access to its own data. DQ*broker* provides a layer of security in addition to that already provided by the DBMS. DQ*broker* can only grant access or update privileges granted to DQ*broker* by the DBMS. With DQ*broker* the goals of local control and universal views are both achieved.

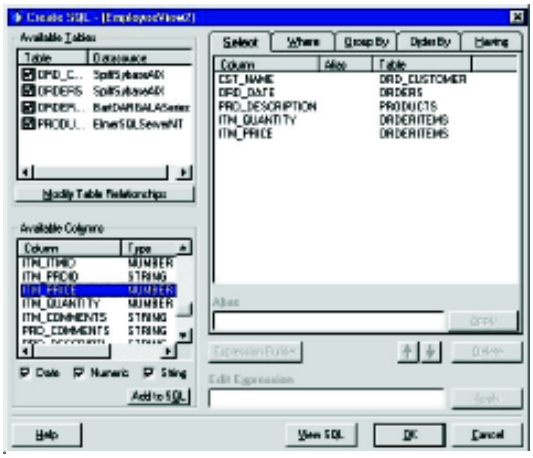

**Figure 3:** The SQL Builder GUI allows global views to be created quickly and easily.

Design, view, and test virtual tables step-by-step with DQ*powersuite*'s global (or enterprise) views, without ever executing a single CREATE TABLE command. For example, create a data mart fact table that consists of information from throughout the enterprise as a view (virtual table). This can also be accomplished with the data mart's dimension tables. Virtual tables look and act like any other table in the enterprise. End-users can test the virtual tables before they are created as physical tables. Once the users are satisfied with the content of the virtual tables, creating them as physical tables in any major RDBMS is quick and easy with Decision Support's one-step, table-copying GUI.

In essence, DQ*powersuite* capabilities provide true "try before you buy" options.

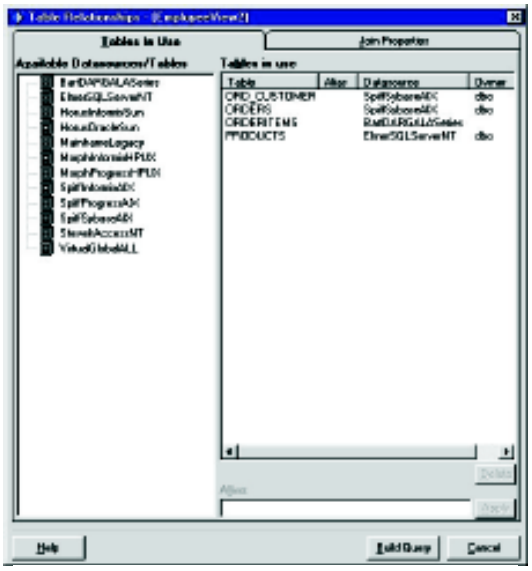

**Figure 4:** The Table Replicator GUI allows any table to be quickly and easily copied from any data source to any data source — including virtual tables.

## Get To Market Faster With Decision Support's SDK

Decision Support's software development kit (SDK) consists of a simple, easy-to-use, object-oriented interface for connecting to DQ*broker* servers, obtaining metadata, and submitting queries for processing.

Called DQ*powerlibs*, the SDK allows the enterprise data access and enterprise extraction, transformation, and loading (ETL) capabilities of DQ*broker* and DQ*transform* to be embedded into any custom application.

DQ*powerlibs* supports development of web-based and client/server applications in C++, Java, and COM/ActiveX.

## More Than Just ETL

Extraction, Transformation, and Loading (ETL) operations done with DQ*transform* have significant advantages over other tools. It can extract and transform data from multiple data sources in a single step.

DQ*transform* offers other significant advantages when developing warehouses and marts. First, with DQ*transform*, no intermediate data staging is required. DQ*transform* allows data to be staged as virtual tables. Second, DQ*transform* makes your ETL independent from vendor specific SQL and DDL (data definition language). DQ*transform* allows you to construct your source tables with generic DDL queries. One set of code constructs a data mart or data warehouse in any RDBMS (e.g., Oracle, Informix, Sybase, DB2, SQL Server or Progress). This is especially advantageous to consultants and integrators who value code reusability and to organizations experiencing uncertainty about their database architecture (e.g., during mergers & acquisitions).

## Robust Data Transformation

DQ*transform* can create and build data warehouses and data marts that include columns of transformed data. DQ*transform* quickly and easily handles any transformation of data that can be done with SQL. In addition, Decision Support's SDK extends DQ*transform* to provide unlimited flexibility in customizing data transformation to the needs of your business.

Users can write custom transformation programs using C++, Java or COM/ActiveX using the libraries and interfaces included in the DQ*powerlibs* SDK. These custom programs can then be automatically called as part of a DQ*transform* ETL Procedure.

A query (in the form of standard SQL) is submitted by a client application to an instance of the query processing application.

## Flexible Connectivity

DQ*broker* is typically installed within a TCP/IP network. This is a standard for the open-systems marketplace. However, DQ*broker* can also interface with network bridge software. This allows it to talk to databases using non-TCP/IP type network protocols.

For example, DQ*broker* can communicate with an IBM mainframe DB2 database from a UNIX or Windows NT platform via a network bridge. In this instance, a DQ*broker* server does not reside on the IBM mainframe, but the DQ*powersuite* domain can still include the database in the single logical view it provides users and developers.

## Update Capable

DQ*broker* can add, delete or modify rows in any table available to it in an entire DQ*powersuite* domain. The administrator controls update privileges at the user and data source level.

## DDL Support

DQ*powersuite* supports Data Definition Language (DDL) syntax (as defined by the ODBC standard), and can create, delete or modify tables in any available data sources that support DDL.

## The Life of a Query

An application on a client machine connects to a DQ*powersuite* listener program using a network socket connection. Shrink-wrapped applications connect to the listener program through Decision Support's ODBC or JDBC driver. Custom applications connect directly using native

calls from the DQ*powerlibs* SDK. DQ*powersuite's* GUI administration tool (DQ*admin*) also connects directly. (DQ*admin* was developed in Microsoft Visual Basic with the DQ*powerlibs* COM/ActiveX interface.)

The listener program establishes the identity of the connecting user and starts an instance of the query processing application to handle the connection. The listener program then goes back to listening for requests from other clients.

A request to the listener program for a connection to a DQ*broker* server can come from a client application submitting a query, another DQ*broker* server that needs information from the global repository or the metadata cache, or from another DQ*broker* server that has a piece of a query it needs processed.

A query (in the form of standard SQL) is submitted by a client application to an instance of the query processing application. During analysis the query processing application:

- *Validates the syntax of the query.*
- *Communicates with its local listener program to find out where all the necessary data resides. The listener program determines this based on the metadata cache and global repository.*
- *Breaks the query into multiple queries based on the location of the needed data. The query pieces contain as much selection, filtering, joining, and sorting as possible. This allows DQbroker to leverage the capabilities of each database management system (DBMS), minimizing the amount of data returned and the time needed to return it.*
- *Sends the pieces of the query to the other DQbroker servers closest to the data required by the query pieces.* (DQ*broker servers process query pieces in the same way as any other query.)*
- *Processes the query pieces for local data.*

Each instance of the DQ*broker* query-processing application gets the result set for its query and streams the result to its point of origin. When all components begin streaming back to the first DQ*broker* server, the result set is streamed to the client application.

The multi-processing and multi-threading capabilities of DQ*powersuite* allow all of the steps required to process a distributed query to happen simultaneously (and asynchronously). Multiple queries are also processed simultaneously.

## The Bottom Line

DQ*powersuite* is easy to implement and provides unprecedented integration of distributed business information at the database level. DQ*powersuite* lets users, developers, shrink wrapped, and custom applications access and use all business data as if it were in a single relational database. Most importantly, DQ*powersuite* is engineered to more efficiently and effectively process queries that join data from different databases and platforms — whether for data access or data movement and transformation.

Decision makers and knowledge workers get the information they need to make the best decisions possible at a much lower cost — lower cost of implementation, lower cost of maintenance, lower cost of hardware, lower cost of bandwidth, lower cost of training.

## System Requirements

DQ*powersuite* supports most major UNIX platforms, as well as Windows NT server and workstation. Whenever possible, DQ*broker* uses native database access routines.

### Operating System Specifications

- UNIX
- Windows NT Server or Workstation (Intel)
- Windows NT Server or Workstation (Alpha)
- AIX
- DG/UX
- HP-UX
- Linux
- Solaris

### Operating Environments

The DQ*broker* server supports data access on most hardware platforms, including:

- Data General
- DEC
- HP
- IBM
- Intel
- NCR
- Sun
- Unisys

### Database Access

DQ*broker* can access the following data source types:

- DB2
- Informix
- ODBC (32-bit)
- Oracle
- Progress
- SQL Server
- Sybase
- ADABAS
- CA-IDMS
- CA-IDMS
- DMS
- DMSII
- IMS-DL/I
- KEYEDIO
- NEON
- Sequential
- VSAM

### Licensing Model

DQ*powersuite* is priced for scalability. The license fees are based on the following factors:

- Concurrent User Connections
- Different data source types
- Number of hardware servers
- Desired Additional Functional Components

*For more information, contact:*

**Decision Support Inc.**

PO Box 1058, Matthews, NC 28106
Tel: 704-845-1000
Fax: 704-847-4875
e-mail: info@decisionsupport.com
web site: www.DecisionSupport.com