Introducing OpenLink Virtuoso™

Universal Data Access Without Boundaries™

Prepared By: Kingsley Idehen

President & CEO, OpenLink Software

Table of Contents

Table of Figures

The Need For Virtual Database Engines

Situation Analysis
As computer hardware, network protocols, database engines, applications, application servers, and desktop productivity tools, proliferate the enterprise, integration of disparate applications from disparate vendors is becoming an all too common problem.

Add the emergence of standards based Distributed Computing galvanized by the Internet infrastructure and associated Internet protocols to this picture, and the need for Integration is even higher.

Increasingly, the industry at large is looking to a new technology deliverable known as Universal Data Access Middleware to address these systems integration pains.

"With Universal Data Access (UDA), customers receive all of the benefits of a high-level and consistent Application Programming Interface (API) that abstracts all the database complexities while providing a capability that can be specified, controlled, and managed on its own to optimize the near universal need of programs for data access".

- Source IDC, 1998 Middleware Markets & Trends

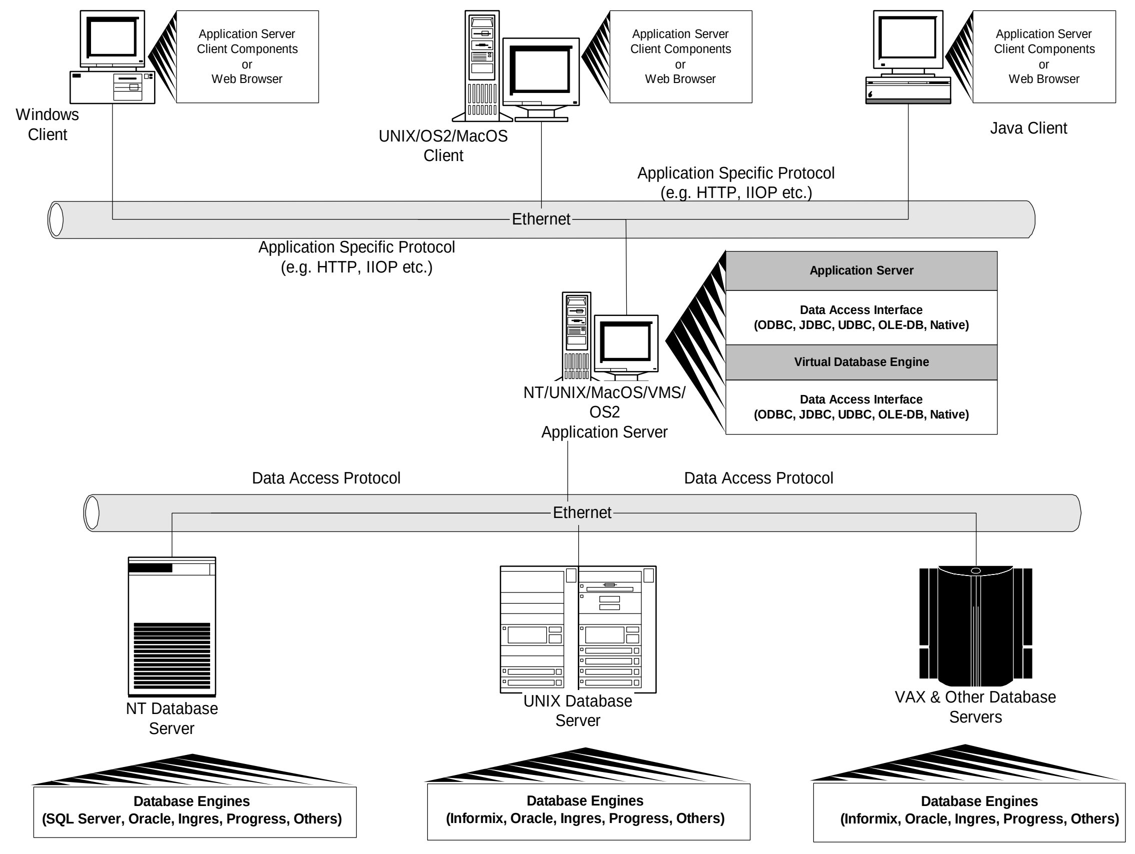
At OpenLink Software, it is our opinion that a new genre of UDA middleware called the "Virtual Database", is set to emerge as the dominant UDA middleware solution for addressing the integration challenges as they exist today, and tomorrow. This new UDA middleware format plays the role of a Universal Data Access manager, fusing traditional database functionality and traditional data access middleware functionality into a single independent packaged software solution.

Virtual Database Engines Defined
A Virtual Database (VDB) Engine is a UDA middleware format that transparently brings local and/or remote heterogeneous databases together using logical database references called Data Source Names (DSNs). A VDB Engine exposes Metadata and Data held within these heterogeneous DSNs to clients applications and services homogeneously.

VDB Engines presume the existence of a number of Database Engines and Data Access Drivers provided by a variety of database vendors within an organization. VDB Engines provide transparent access to these heterogeneous databases via DSNs associated with the relevant data access drivers without exposing end-users or developers to the intricacies of heterogeneous data access.

Data Source Names (DSNs)
A Data Source Name is a logical reference that exposes database to standards compliant or native data access drivers. DSNs provide a flexible naming and binding service for database driven applications developers and end-users alike. Applications no longer need to be inextricably linked to specific database names or specific database engines.

Figure 1 – Distributed Computing Infrastructure Incorporating A Virtual Database Engine



Application Server Client Components or Web Browser
Windows Client
Application Server Client Components or Web Browser
UNIX/OS2/MacOS Client
Application Server Client Components or Web Browser
Java Client
Application Specific Protocol (e.g. HTTP, IIOP etc.)
Ethernet
Application Specific Protocol (e.g. HTTP, IIOP etc.)
Application Server
Data Access Interface (ODBC, JDBC, UDBC, OLE-DB, Native)
Virtual Database Engine
Data Access Interface (ODBC, JDBC, UDBC, OLE-DB, Native)
NT/UNIX/MacOS/VMS/ OS2 Application Server
Data Access Protocol
Data Access Protocol
Ethernet
NT Database Server
UNIX Database Server
VAX & Other Database Servers
Database Engines (SQL Server, Oracle, Ingres, Progress, Others)
Database Engines (Informix, Oracle, Ingres, Progress, Others)
Database Engines (Informix, Oracle, Ingres, Progress, Others)

First Generation Virtual Database Products

Although the strict VDB definition may be new, there are a number of products that have been around for a while that attempt to address VDB issues. The list of such products includes The Microsoft JET Engine, Borland Database Engine (BDE), and IBM DataJoiner.

Microsoft JET

The Microsoft JET Engine lies at the heart of Microsoft Access, it is the piece of technology that allows you to link external and typically remote database tables into your local Access space via ODBC Data Sources. Once this link process has been completed, Access allows you to build Queries, Reports, Forms etc. using these external database tables as though they were Local Access tables. JET can also link to external tables hosted within desktop database engines via native interfaces.

The Microsoft JET Engine services are exposed via Microsoft provided data access interfaces such as: DAO, ADO, and OLE-DB. These interfaces are integral parts of most Microsoft applications, thereby exposing the benefits of the JET VDB transparently.

Borland Database Engine

The Borland Database Engine (BDE) from Inprise like the Microsoft JET Engine also facilitates external table linkage via ODBC Data Sources. The BDE also lets you link to external database tables via native database interfaces and there is no restriction to desktop database engines when you adopt this approach.

Although the BDE has a published set of APIs, it is predominantly used by Inprise applications in very much the same way JET is used by Microsoft applications.

IBM DataJoiner

DataJoiner from IBM provides the ability to access heterogeneous data sources via IBM DB/2 Client Application Enablers. It does support ODBC and JDBC as client interfaces and makes use of Native or ODBC based data access for external Data I/O.

VDB Implementation Issues

The essential components that affect the implementation of VDB Engines are, High-Level Data Access Interfaces, Low-Level Data Access Interfaces and Traditional Database Functionality.

High-Level Data Access Interfaces

A VDB Engine's capabilities are exposed via High Level Data Access interfaces. For the purpose of this document, a high level data access interface is an interface utilized predominantly by applications, as opposed to middleware developers for achieving application database independence. A high level data access interface sits atop Low-Level data access interfaces, providing an abstraction layer that serves to simplifying the process of database independent application development.

A number of High Level Data Access standards exist today, the more prevalent being: Data Access Objects (DAO), Remote Data Objects (RDO), ActiveX Data Objects (ADO), JavaBlend, InfoBus.

It is important to note that low-level Data access interfaces such as ODBC, UDBC, JDBC and OLE-DB transparently serve the high-level interfaces mentioned in the section above. Thus, in most cases VDB vendors will treat ODBC, UDBC, JDBC, and OLE-DB as high-level interfaces by providing VDB data access drivers conforming to these standards as part of the VDB deliverable.

Low-Level Data Access Interfaces

A VDB Engine's data I/O occurs via low-level data access interfaces to underlying database engines or data sources. In recent times the Open Database Connectivity (ODBC) API and the X/Open SQL Call Level Interface (CLI) have emerged as the dominant industry wide Low-Level Data Access Standards. OLE-DB from Microsoft is also emerging as a new low-level data access standard for relational and non-relational data in the Microsoft Component Object Model (COM) world. While JDBC is emerging like wise as the low-level data access standard for the burgeoning Java world.

A VDB may also be a Native Database Interface Client, making use of database engine vendor provided data access interfaces. Native interfaces are based upon Embedded SQL, an older format Low-Level data access interface that preceded the X/Open SQL CLI. It is important to note that ODBC from Microsoft, JDBC from JavaSoft, and UDBC from OpenLink Software are all derived from the X/Open SQL CLI.

Traditional Database Functionality

The degree to which a VDB implements a traditional database engine's functionality has a direct bearing on the intrinsic value of a VDB engine. Traditional database functionality is extensive, but for the purposes of this document, a core set of functionality common to all commercial database engines has been assembled. The functionality list includes:

Query Language Support - standard syntax for interrogating, manipulating, describing, and securing data contained within a database. Examples include the Structured Query Language (SQL) for relational databases and the Object Query Language (OQL) for Object and Object-Relational Databases.

Query Processor – the mechanism used by a database engine to convert Query Language Statements into actual data retrieval instructions. In addition, this database component is responsible for ensuring Query Language syntax conformance, Query Execution Plan Assembly and Query Fulfillment.

Standard Data Types Support – data contained within a database must be describable using standard data types e.g. Character, Number, Date, etc.

VIEW Support – pre constructed query statements stored within a database, for the purpose of query simplification, or content and structural security.

Stored Procedure Support – Stored Procedures facilitate the embedding of application programming logic within a database. Their pre-compiled nature enhances data access performance by reducing message hops between database servers and database clients.

Scrollable Cursor Support – the process by which the result of a database query (known as a result-set) is traversed. Traversal occurs in either direction, backwards or forwards, using result-set chunks (known as row-sets). Resultset scrolling occurs when database engines exchange data with database clients.

Concurrency Control – the process through which a database engine supports multiple sessions running concurrently, across multiple database users and database client applications without compromising underlying data integrity or introducing quantum increases in application response times.

Transaction Support - ensures that database instructions can be grouped into logical units of execution that are Atomic, Consistent, Isolated from the effect of other units of execution affecting the same underlying data, and Durable.

Transaction Isolation - describes the ability of a database engine to provide transaction process partitioning options called Isolation Levels, that offer different ways of managing the effects of multiple and concurrent transactions affecting the same underlying data.

Distributed Transaction Support – describes the ability to preserve transaction atomicity, consistency, integrity, and durability across database servers hosted on the same or different database server machines within a networked environment. This involves supporting transaction Commits and Rollbacks using a 2-phase commit protocol.

User Definable Type Support – this is how a database engine allows end-users to extend its base functionality. This is achieved by providing interfaces that allow end-users to create new ways in which a database engine's data is described and manipulated.

Federated Database Support –data access and manipulation across database servers resident on the same machine.

Distributed Database Support - data access, and manipulation across database servers resident on the different machines within a networked environment.

Security – the process by which data, and data transmission is protected using a combination of database and operating system privileges, roles and roles hierarchies. It also includes the ability of a database engine to protect data transmitted to its clients using data encryption.

Virtual Database Engine Components

The prior section outlined the critical implementation issues that affect the development and implementation of VDB Engines. These issues form the basis around which a component based framework for depicting VDB architectures has been derived.

The components that comprise a VDB Engine framework are as follows:

Data Access Drivers

The VDB component that forms the entry point to the VDB Engine's services, these drivers may or may not conform to industry standards. Applications and Services that sit atop a VDB Engine must have their data access layers written to the same Application Programming Interfaces (APIs) implemented by the Data Access Drivers provided by a VDB engine.

Security Manager

The VDB component that is responsible for protecting data and data transmission (using encryption) within the VDB Engine's domain. It is also responsible for managing Application, User, Group, Role and Domain privileges as they relate to the creation, manipulation and destruction of VDB data and metadata.

Query Manager

The VDB component that handles queries presented to it by the VDB Engine's data access drivers. It provides query syntax checking, query execution plan compilation, and query fulfillment services. A query processor is built in conformance to one or more query language specifications, the most notable being the Structured Query Language (SQL) for relational database engines, and the Object Query Language (OQL) for Object-Relational and Object Database engines.

Meta Data Manager

The VDB component that provides the Query Processor with information about the data entities from which the Query Processor's execution plan is derived. Metadata managers are also the components responsible for linking external data sources into the VDB domain and directing the Query Processor to the appropriate Data I/O manager.

Transaction Manager

The Transaction Manager component ensures that transactions are Atomic (clearly distinguishable units), Consistent (thereby preserving integrity of data), Isolated from the effect of other transactions, and Durable (such that the effects of committed transactions survive failure). The Transaction Manager ensures VDB Engines are capable of supporting Online Transaction Processing (OLTP) and Distributed Transaction oriented applications and services. Transaction Managers may be standards based implementing X/Open's XA Resource Manager Specifications. Distributed transaction support is implemented by using a two-phase commit protocol.

Concurrency Manager

The VDB component that ensures client applications and services are capable of opening multiple concurrent sessions that execute data INSERTS, UPDATES and DELETIONS, without implicitly reducing application response times or compromising data integrity. Concurrency control is delivered in one of two formats, Optimistic or Pessimistic depending on the response times desired by VDB client applications or services.

Local I/O Manager

VDB Engine's that provide local data storage uses this component for reading and writing data to disk. This is how a VDB provides traditional database engine data storage services.

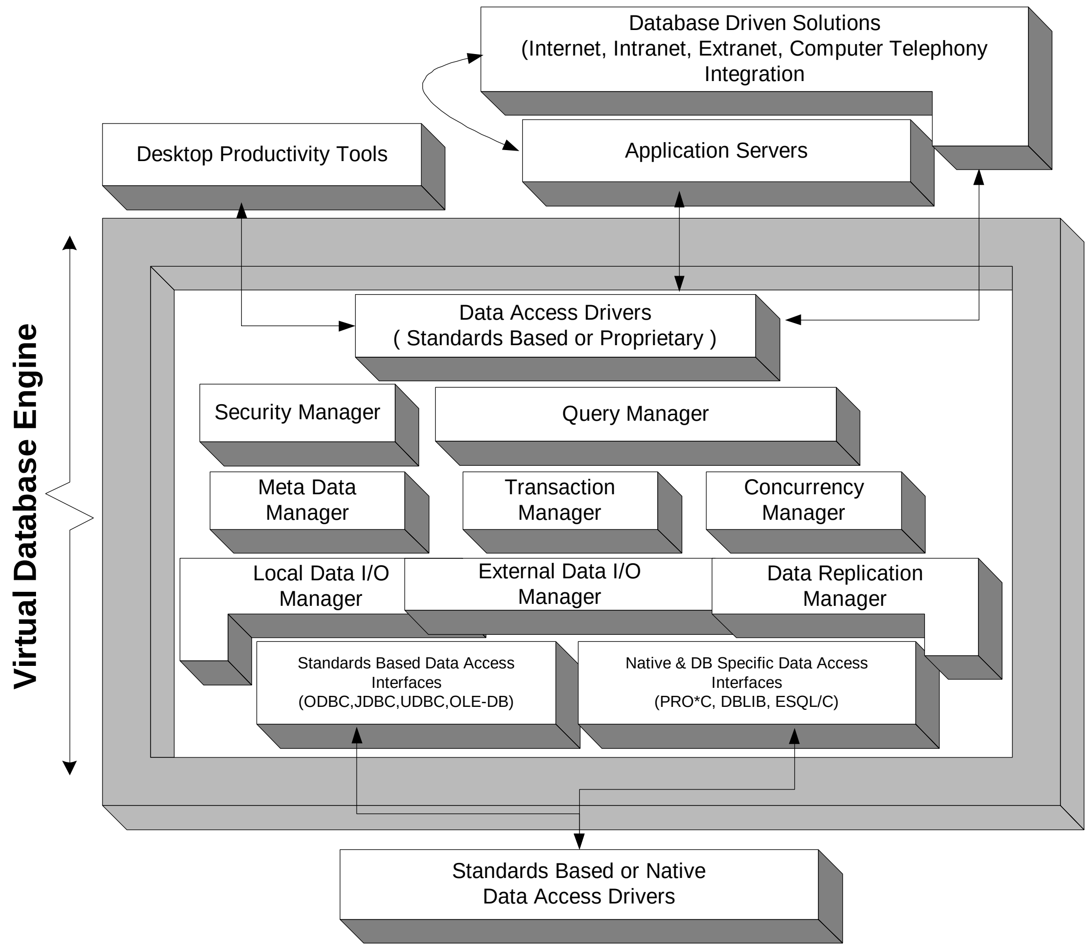
External Data I/O Manager

VDB component that handles data reads and writes to external data sources. The External Data I/O Manager be implemented using standard data access interfaces such as ODBC, JDBC, UDBC, OLE-DB or Native data source interfaces.

Replication Manager

Component that manages data migration and synchronization across two or more VDB servers within a distributed computing environment. This component acts as a data coordinator between the activities of Local Data I/O and External Data I/O Managers across VDB servers. The Replication Manager enables a VDB Engine to offer automated bi-directional data, and metadata transformation services across heterogeneous data sources without end-user or developer intervention.

Figure 2 - Virtual Database Engine Architecture & Components



Database Driven Solutions (Internet, Intranet, Extranet, Computer Telephony Integration
Desktop Productivity Tools
Application Servers
Virtual Database Engine
Data Access Drivers (Standards Based or Proprietary)
Security Manager
Query Manager
Meta Data Manager
Transaction Manager
Concurrency Manager
Local Data I/O Manager
External Data I/O Manager
Data Replication Manager
Standards Based Data Access Interfaces (ODBC,JDBC,UDBC,OLE-DB)
Native & DB Specific Data Access Interfaces (PRO*C, DBLIB, ESQL/C)
Standards Based or Native Data Access Drivers

VDB Implementation Approaches

There are no golden VDB implementation specifications, but the implementation of a VDB has a direct impact the degree to which you realize desired value from the VDB concept as a whole.

The VDB value proposition is simply stated as follows:

> "To provide transparent access to heterogeneous data sources, independent of host operating system and underlying database engines ".

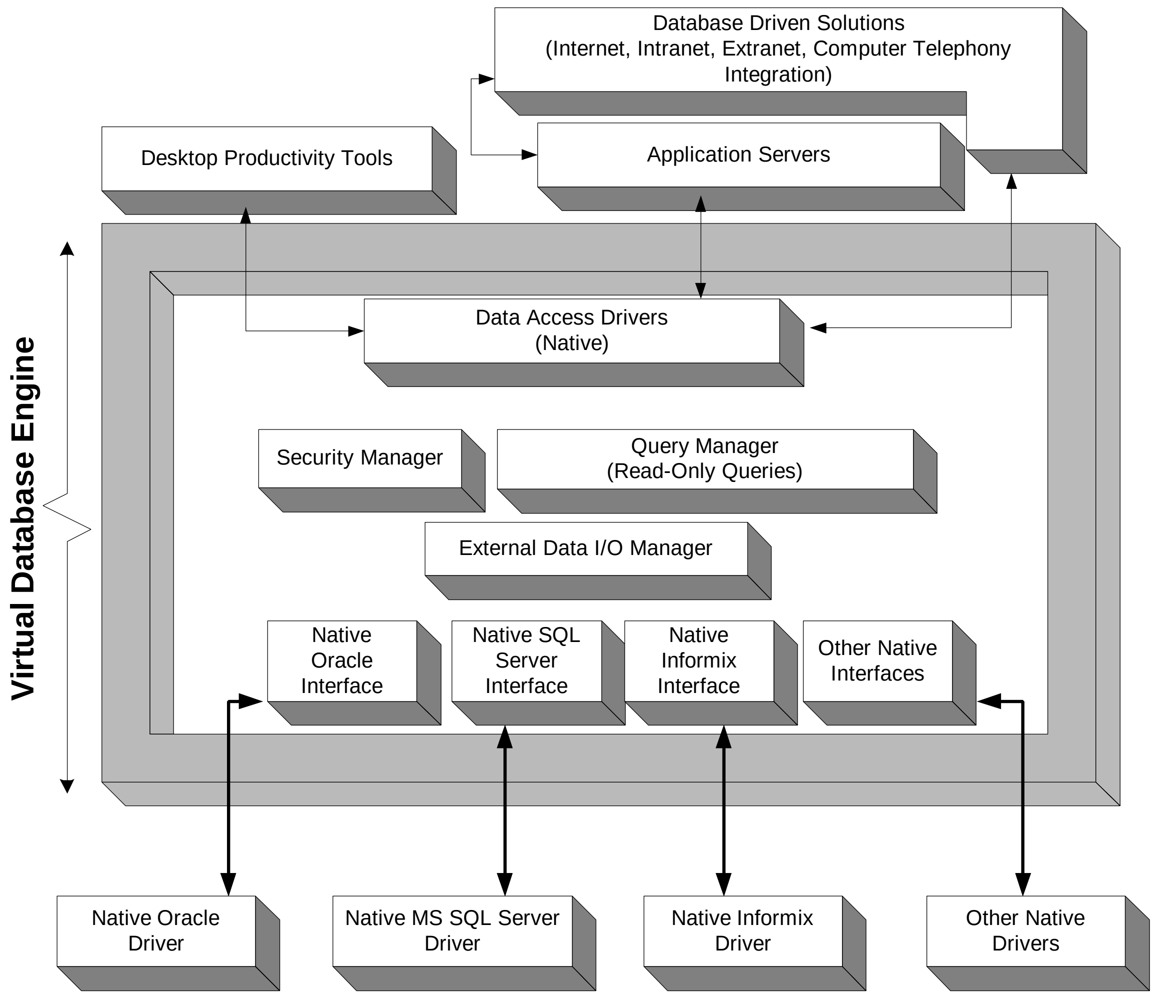
VDB implementations can be categorized as follows:

	VDB Data Access Interface	**VDB External Data I/O**	**Traditional Database Functionality**
Type 1	Native	Native	Partial
Type 2	Native	Native	Full
Type 3	Native	Standards Based	Partial
Type 4	Native	Standards Based	Full
Type 5	Standards Based	Native	Partial
Type 6	Standards Based	Native	Full
Type 7	Standards Based	Standards Based	Partial
Type 8	Standards Based	Standards Based	Full
Type 9	Standards Based	Standards Based or Native	Partial
Type 10	Standards Based	Standards Based or Native	Full

The sections that follow provide illustrations of the different VDB formats, depicting the components that provide the basis for the categorization used in the table above.

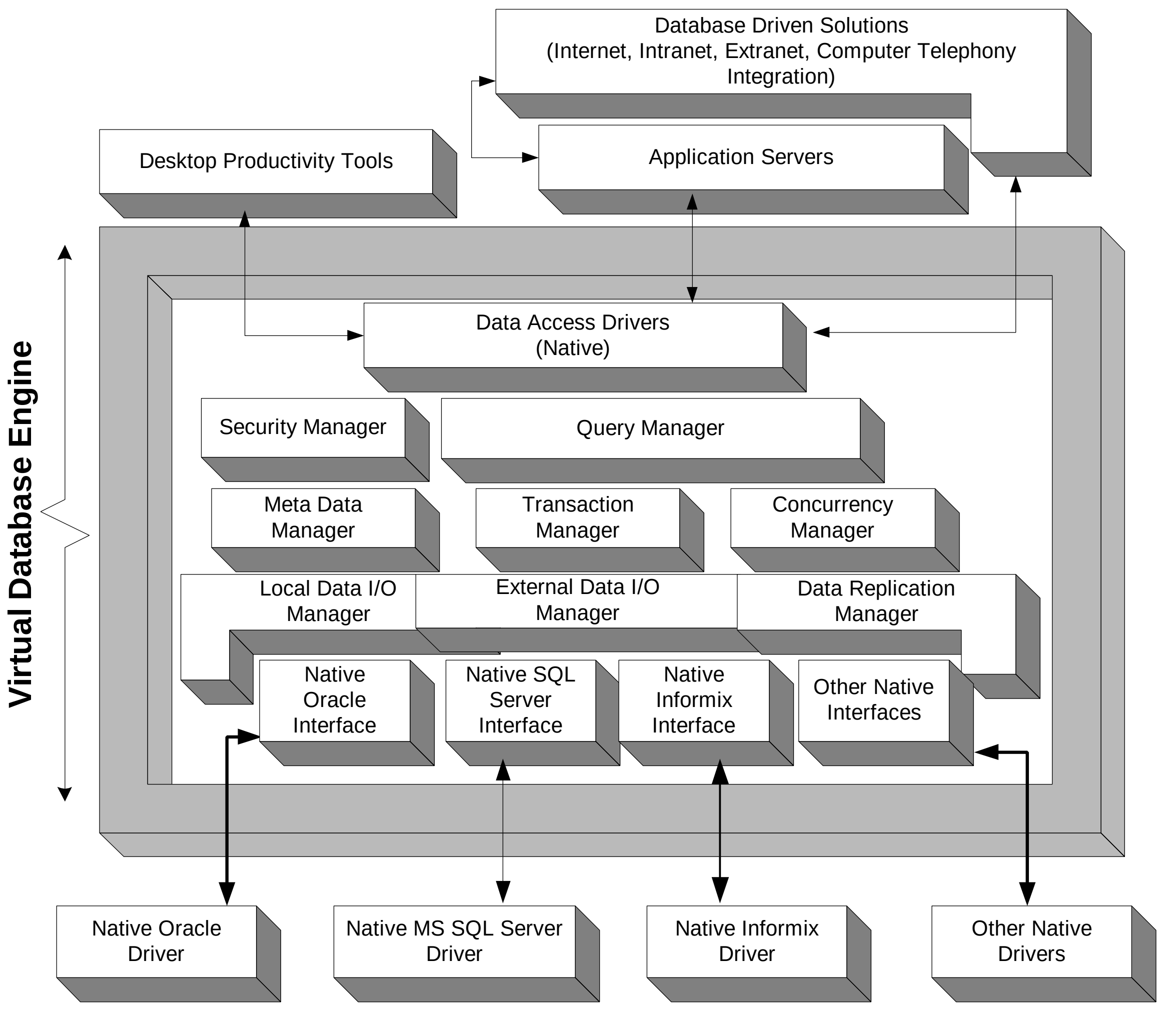
Type 1 VDB Engine
This category of VDB exposes its services to clients via Native and Proprietary high-level data access interfaces. Data I/O is achieved via native, proprietary, and data source specific low-level data access interfaces. This category of VDB does not possess a complete set of traditional database engine components.

Figure 3 - Type 1 VDB Engine Architecture


Database Driven Solutions (Internet, Intranet, Extranet, Computer Telephony Integration)
Desktop Productivity Tools
Application Servers
Virtual Database Engine
Data Access Drivers (Native)
Security Manager
Query Manager (Read-Only Queries)
External Data I/O Manager
Native Oracle Interface
Native SQL Server Interface
Native Informix Interface
Other Native Interfaces
Native Oracle Driver
Native MS SQL Server Driver
Native Informix Driver
Other Native Drivers

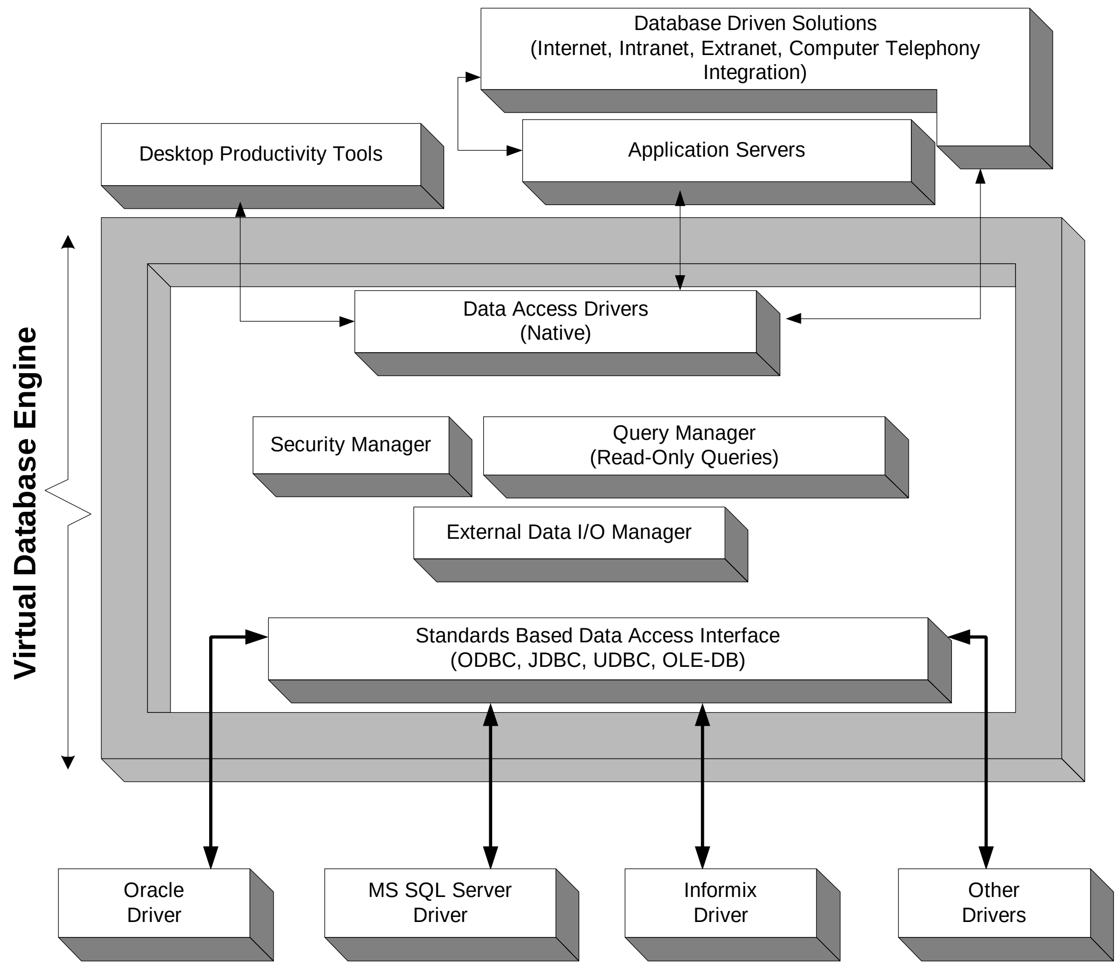
Type 2 VDB Engine
This category of VDB exposes its services to clients via Native and Proprietary high-level data access interfaces. External data I/O is achieved via native, proprietary, and data source specific low-level data access interfaces. This category of VDB possesses a complete set of traditional database engine components.

Figure 4 - Type 2 VDB Engine Architecture



Type 3 VDB Engine
This category of VDB exposes its services to clients via Native and Proprietary high-level data access interfaces. Data I/O is achieved via Open, Standards based, and Database Independent low-level data access interfaces. This category of VDB does not possess a complete set of traditional database engine components.

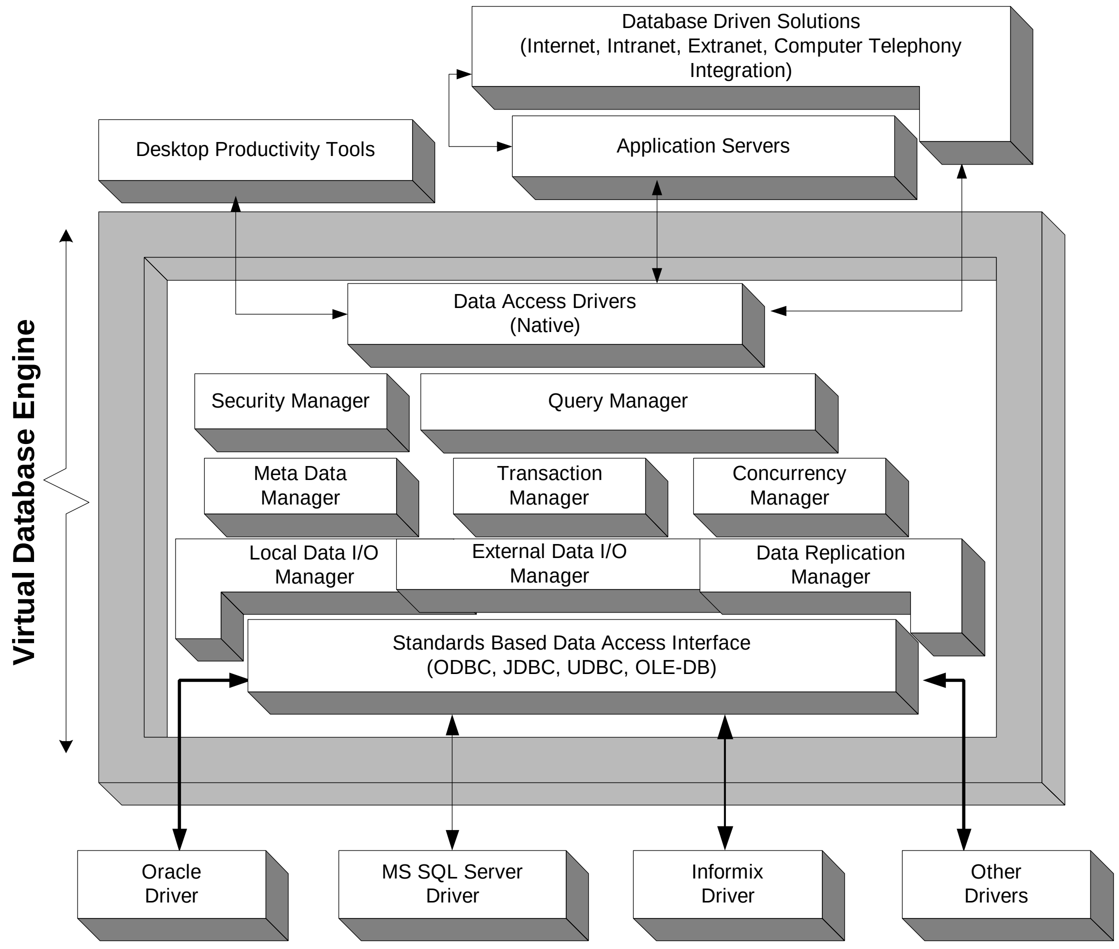
Figure 5 - Type 3 VDB Engine Architecture



Database Driven Solutions (Internet, Intranet, Extranet, Computer Telephony Integration)
Desktop Productivity Tools
Application Servers
Virtual Database Engine
Data Access Drivers (Native)
Security Manager
Query Manager (Read-Only Queries)
External Data I/O Manager
Standards Based Data Access Interface (ODBC, JDBC, UDBC, OLE-DB)
Oracle Driver
MS SQL Server Driver
Informix Driver
Other Drivers

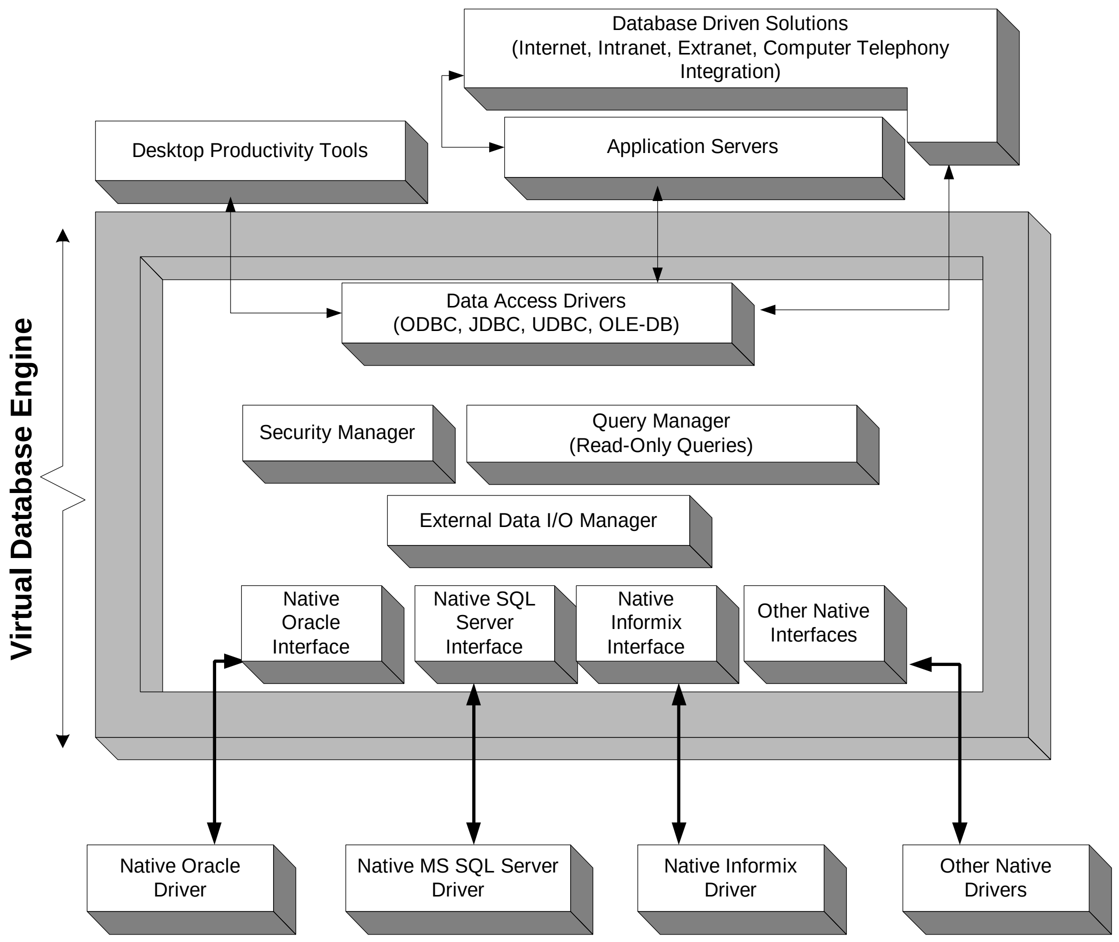
Type 4 VDB Engine
This category of VDB exposes its services to clients via Native and Proprietary high-level data access interfaces. External data I/O is achieved via Open, Standards based, and Database Independent low-level data access interfaces. This category of VDB possesses a complete set of traditional database engine components.

Figure 6 - Type 4 VDB Architecture



Type 5 VDB Engine
This category of VDB exposes its services to clients via open and standards based high-level data access Interfaces. Data I/O is achieved via native, proprietary, and data source specific low-level interfaces. This category of VDB does not possess a complete set of traditional database engine components.

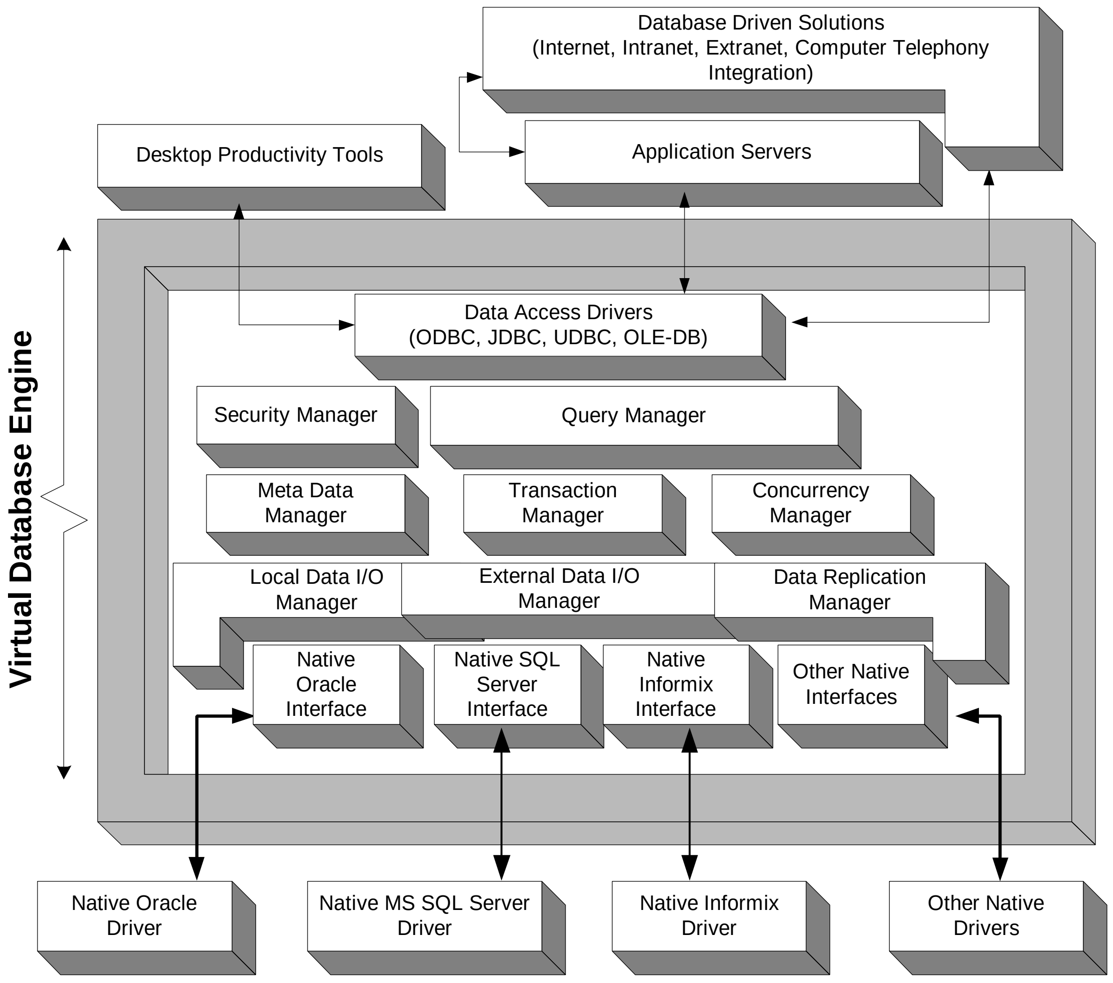
Figure 7 - Type 5 VDB Engine Architecture



Database Driven Solutions (Internet, Intranet, Extranet, Computer Telephony Integration)
Desktop Productivity Tools
Application Servers
Virtual Database Engine
Data Access Drivers (ODBC, JDBC, UDBC, OLE-DB)
Security Manager
Query Manager (Read-Only Queries)
External Data I/O Manager
Native Oracle Interface
Native SQL Server Interface
Native Informix Interface
Other Native Interfaces
Native Oracle Driver
Native MS SQL Server Driver
Native Informix Driver
Other Native Drivers

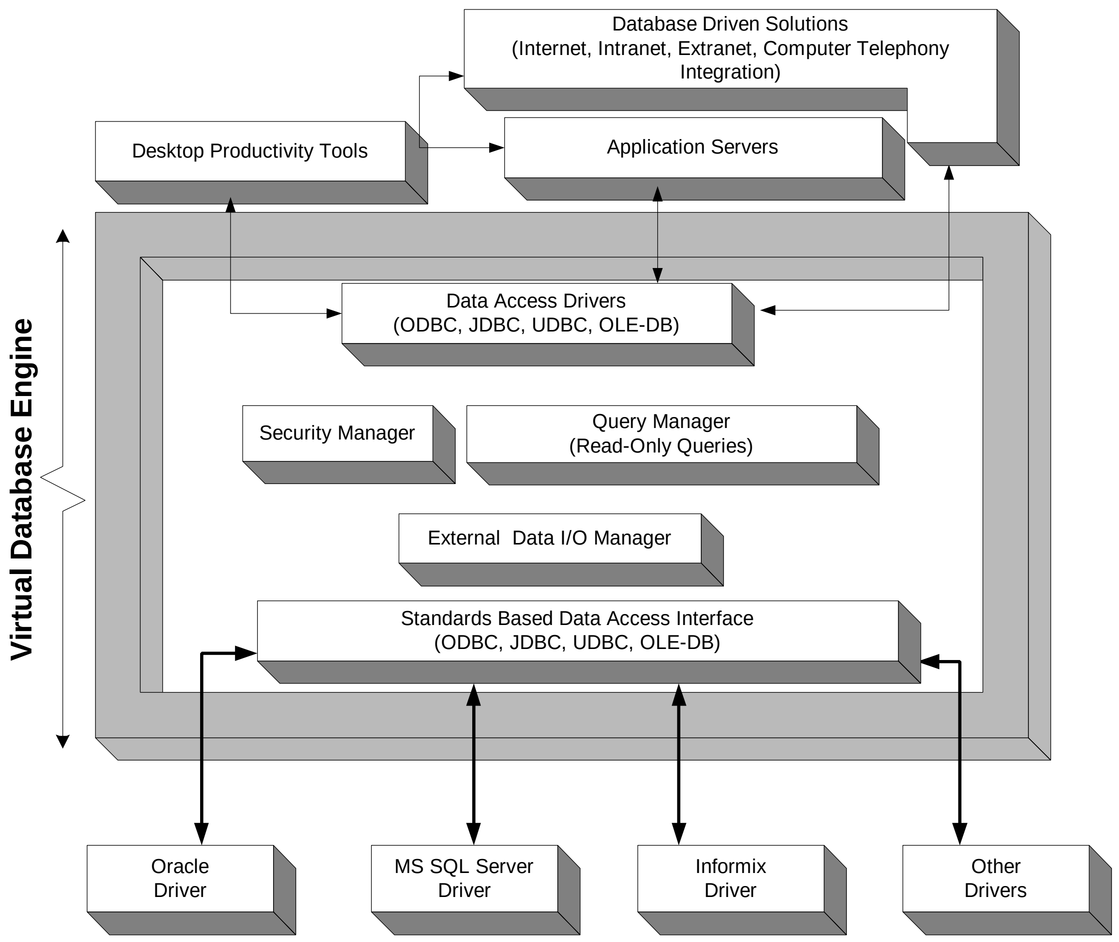
Type 6 VDB Engine
This category of VDB exposes its services to clients via open, standards based, high and low-level Interfaces. External data I/O is achieved via native, proprietary, and data source specific low-level interfaces. This category of VDB possesses a complete set of traditional database engine components.

Figure 8 - Type 6 - VDB Engine Architecture



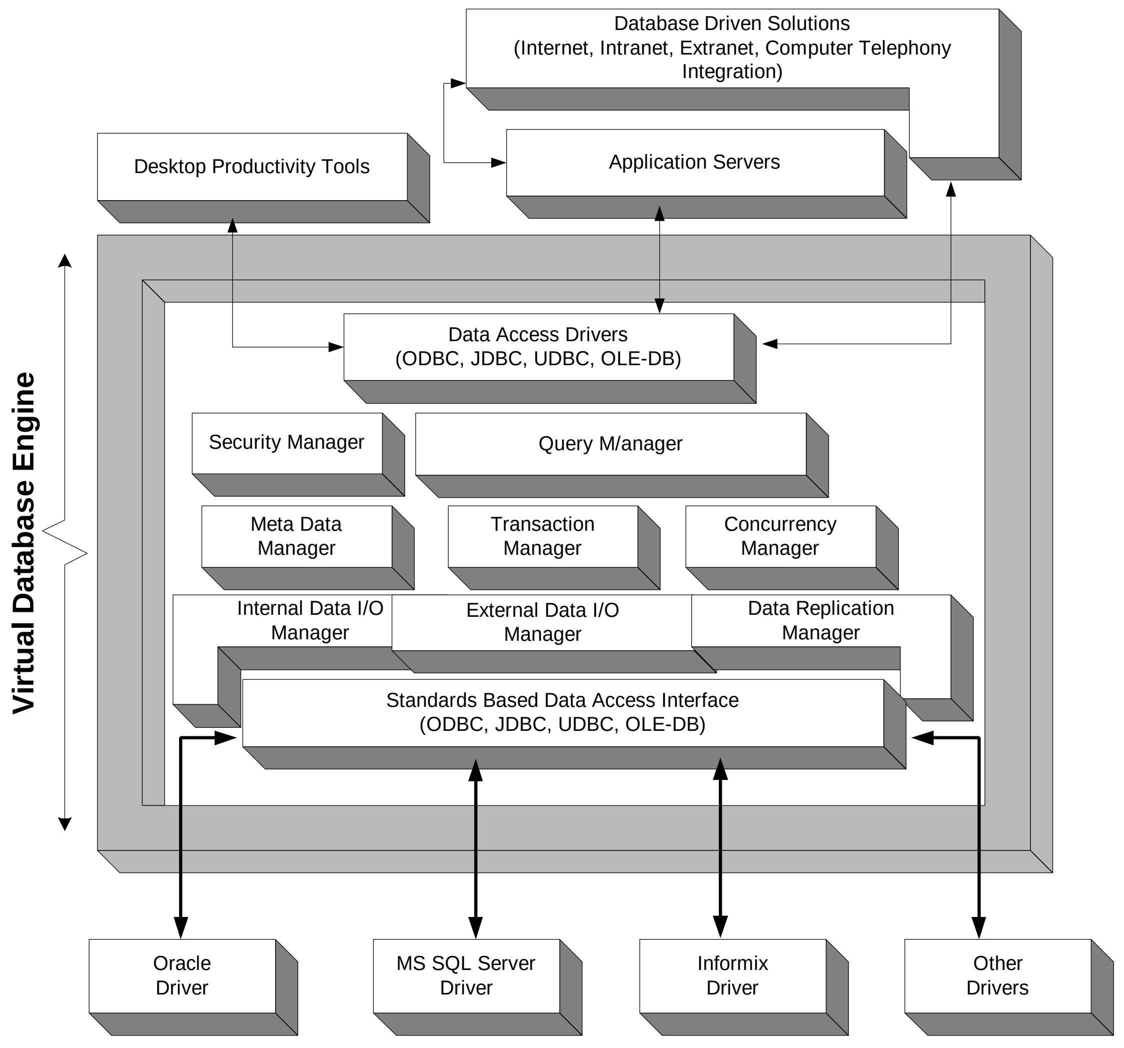
Type 7 VDB Engine
This category of VDB exposes its services via open, standards based high-level data access interfaces. Data I/O is achieved via Open, Standards based, and Database Independent low-level data access interfaces. This category of VDB does not possess a complete set of traditional database engine components.

Figure 9 - Type 7 VDB Engine Architecture


Database Driven Solutions (Internet, Intranet, Extranet, Computer Telephony Integration)
Desktop Productivity Tools
Application Servers
Virtual Database Engine
Data Access Drivers (ODBC, JDBC, UDBC, OLE-DB)
Security Manager
Query Manager (Read-Only Queries)
External Data I/O Manager
Standards Based Data Access Interface (ODBC, JDBC, UDBC, OLE-DB)
Oracle Driver
MS SQL Server Driver
Informix Driver
Other Drivers

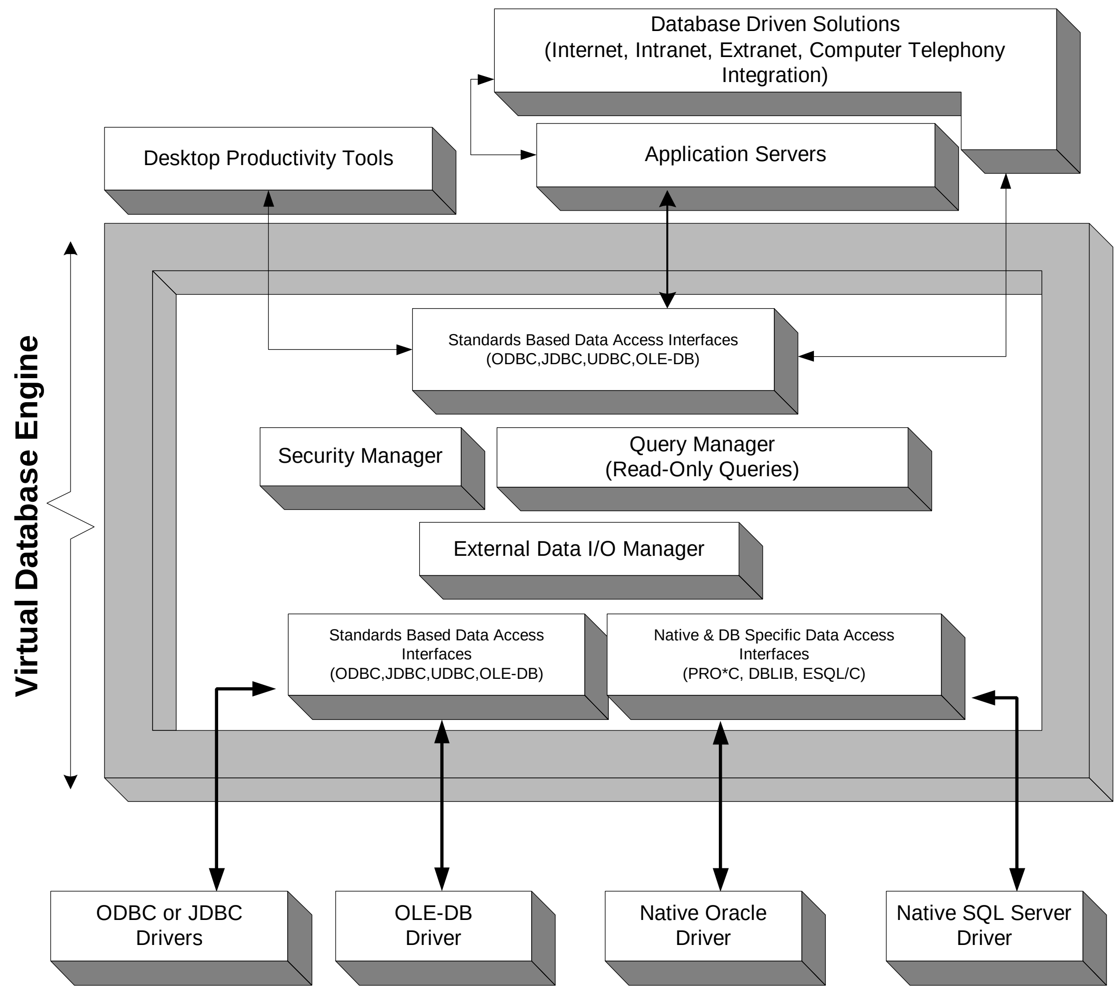
Type 8 VDB Engine
This category of VDB exposes its services via open, standards based, high and low-level interfaces. External data I/O is achieved via Open, Standards based, and Database Independent low-level data access interfaces. This category of VDB does possess a complete set of traditional database engine components.

Figure 10 - Type 8 VDB Engine Architecture


Database Driven Solutions (Internet, Intranet, Extranet, Computer Telephony Integration)
Desktop Productivity Tools
Application Servers
Virtual Database Engine
Data Access Drivers (ODBC, JDBC, UDBC, OLE-DB)
Security Manager
Query M/anager
Meta Data Manager
Transaction Manager
Concurrency Manager
Internal Data I/O Manager
External Data I/O Manager
Data Replication Manager
Standards Based Data Access Interface (ODBC, JDBC, UDBC, OLE-DB)
Oracle Driver
MS SQL Server Driver
Informix Driver
Other Drivers

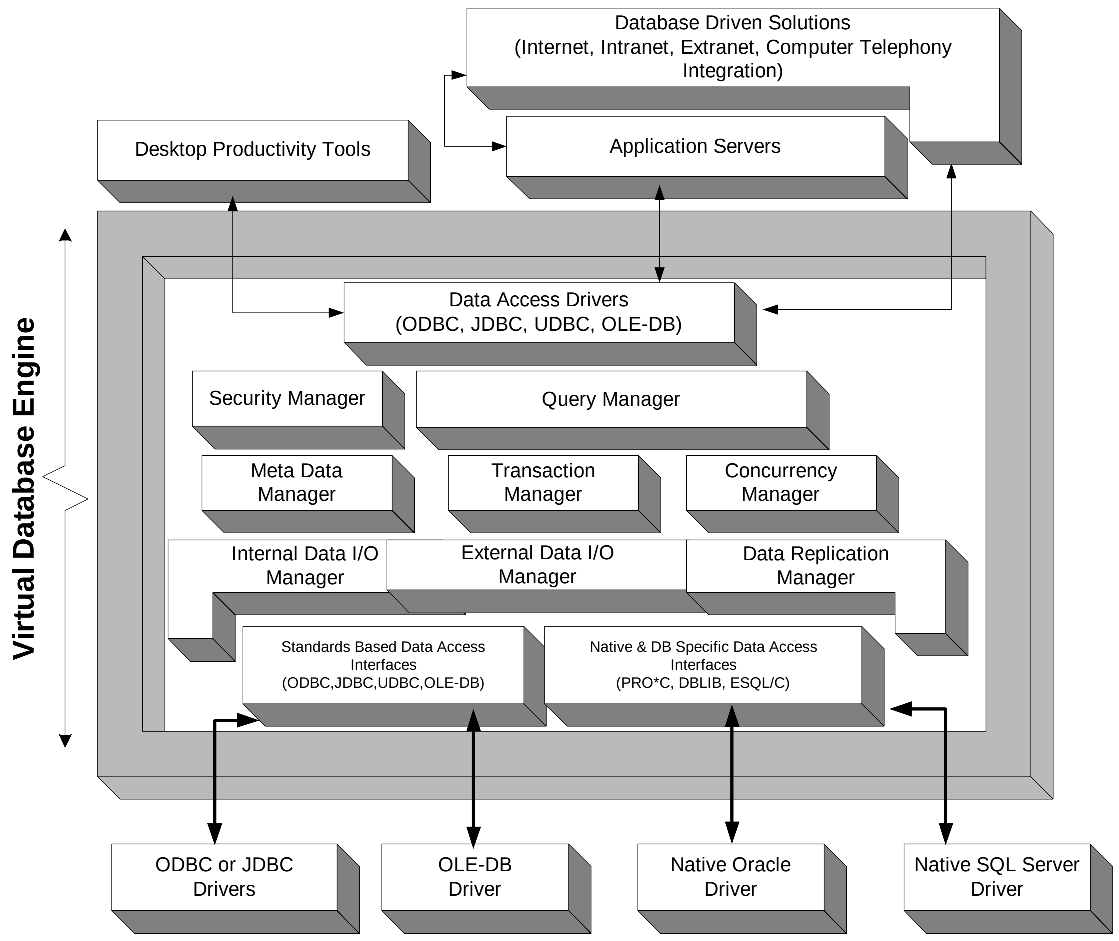
Type 9 VDB Engine
This category of VDB exposes its services via Open, Standards based, high-level data access interfaces. Data I/O is achieved by using either Open, Standards based, and Database Independent low-level data access interfaces or Native, Proprietary, and Database Specific low-level data access interfaces. This category of VDB does not possess a complete set of traditional database engine components.

Figure 11 - Type 9 VDB Architecture


Database Driven Solutions (Internet, Intranet, Extranet, Computer Telephony Integration)
Desktop Productivity Tools
Application Servers
Virtual Database Engine
Standards Based Data Access Interfaces (ODBC,JDBC,UDBC,OLE-DB)
Security Manager
Query Manager (Read-Only Queries)
External Data I/O Manager
Standards Based Data Access Interfaces (ODBC,JDBC,UDBC,OLE-DB)
Native & DB Specific Data Access Interfaces (PRO*C, DBLIB, ESQL/C)
ODBC or JDBC Drivers
OLE-DB Driver
Native Oracle Driver
Native SQL Server Driver

Type 10 VDB Engine
This category of VDB exposes its services via Open, Standards based, high-level data access interfaces. External data I/O is achieved by using either Open, Standards based, and Database Independent low-level data access interfaces or Native, Proprietary, and Database Specific low-level data access interfaces. This category of VDB possesses a complete set of traditional database engine components.

Figure 12 - Type 10 VDB Architecture


Database Driven Solutions (Internet, Intranet, Extranet, Computer Telephony Integration)
Desktop Productivity Tools
Application Servers
Virtual Database Engine
Data Access Drivers (ODBC, JDBC, UDBC, OLE-DB)
Security Manager
Query Manager
Meta Data Manager
Transaction Manager
Concurrency Manager
Internal Data I/O Manager
External Data I/O Manager
Data Replication Manager
Standards Based Data Access Interfaces (ODBC,JDBC,UDBC,OLE-DB)
Native & DB Specific Data Access Interfaces (PRO*C, DBLIB, ESQL/C)
ODBC or JDBC Drivers
OLE-DB Driver
Native Oracle Driver
Native SQL Server Driver

OpenLink Virtuoso™ - Next Generation Virtual Database Engine

Virtuoso is a revolutionary, next generation, high-performance virtual database engine for the Distributed Computing Age. It is an essential universal data access middleware technology set to accelerate our advancement towards the emerging Information Age.

Virtuoso provides transparent access to your existing data sources, which are logical references to databases from different database vendors, exposed by data access drivers also provided by different vendors.

Through a single connection, Virtuoso will simultaneously connect your ODBC, JDBC, UDBC, OLE-DB client applications and services to data within Oracle, Microsoft SQL Server, DB/2, Informix, Progress, CA-Ingres, Sybase, PostgresSQL, Solid, Velocis and other ODBC compliant database engines.

Virtuoso provides the end-user or applications developer with a one application or development environment to many database engines relationship abstracting either party from the complexities of heterogeneous data access.

Virtuoso allows end-users and application developers to retain a single data source or database focus, at the same time exposing either party to the benefits of heterogeneous data access, without introducing proportional increases in complexity.

Virtuoso is a type 10 VDB Engine, exposing its services via standards based data access interfaces such as ODBC, JDBC, UDBC, and OLE-DB and performing external Data I/O occurs via ODBC, UDBC, or Native data access interfaces.

Design Goals

Virtuoso has been developed with the following goals in mind:

1. Operating System Independence – support for all main-stream operating systems

2. Small Memory Footprint – no more than 2MB of memory for basic operations

3. Small Binary Distribution – maximum of 10MB of disk space for base product installation

4. High-Performance – deliver performance levels required by enterprise-wide solutions

5. Standards Based Data I/O – use standard such as ODBC from Microsoft and UDBC from OpenLink Software for low-level data access

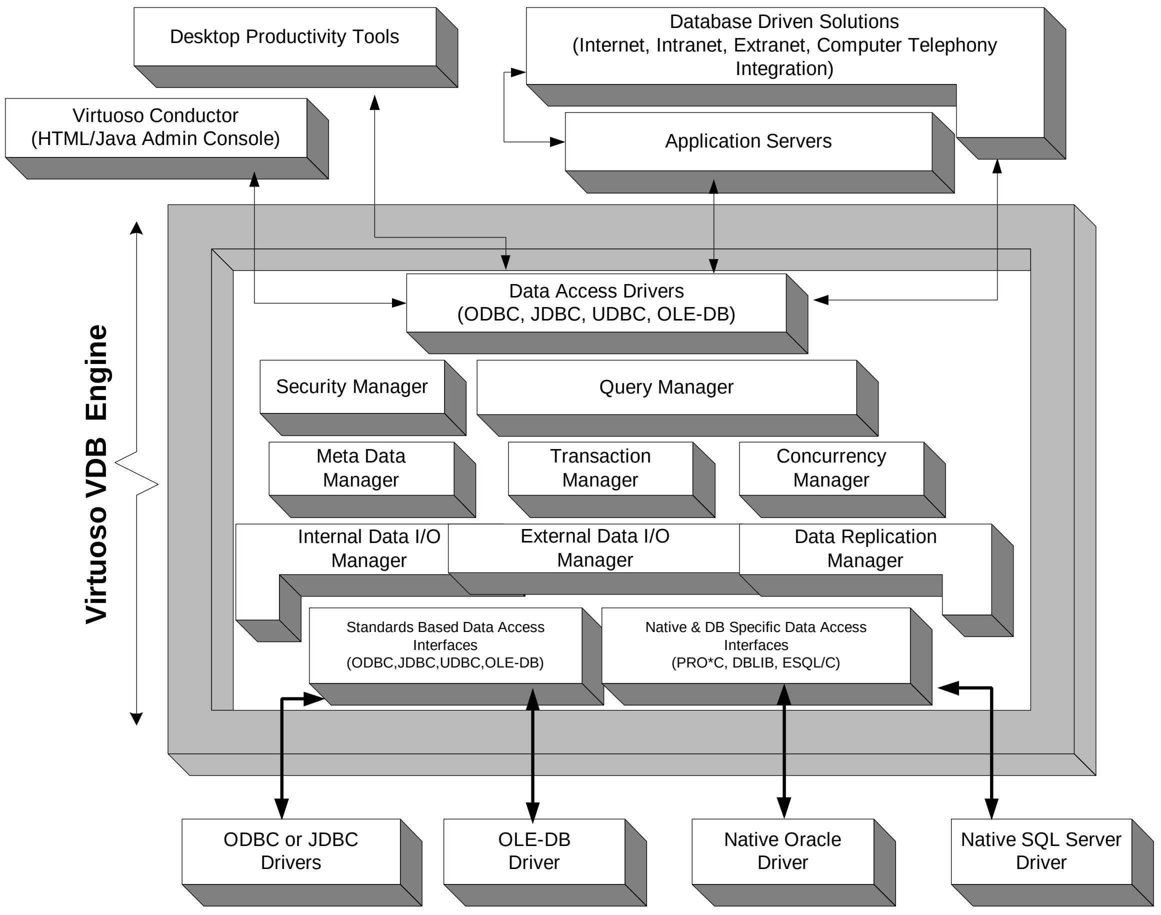
6. Standards Based Interfaces to Services – expose Virtual Database functionality via ODBC, UDBC, JDBC, and OLE-DB data access drivers

7. Web Based Configuration & Management – a Web Browser as the standard administration and configuration interface.

Architecture

Virtuoso's VDB type 10 architecture is depicted below.

Figure 13 - OpenLink Virtuoso™ VDB Architecture


Desktop Productivity Tools
Database Driven Solutions (Internet, Intranet, Extranet, Computer Telephony Integration)
Virtuoso Conductor (HTML/Java Admin Console)
Application Servers
Virtuoso VDB Engine
Data Access Drivers (ODBC, JDBC, UDBC, OLE-DB)
Security Manager
Query Manager
Meta Data Manager
Transaction Manager
Concurrency Manager
Internal Data I/O Manager
External Data I/O Manager
Data Replication Manager
Standards Based Data Access Interfaces (ODBC,JDBC,UDBC,OLE-DB)
Native & DB Specific Data Access Interfaces (PRO*C, DBLIB, ESQL/C)
ODBC or JDBC Drivers
OLE-DB Driver
Native Oracle Driver
Native SQL Server Driver

Note: support for JDBC and OLE-DB based external data I/O is still being developed.

Virtuoso Components

Data Access Drivers

ODBC, JDBC, UDBC, and OLE-DB Drivers for Virtuoso are an integral part of the OpenLink Virtuoso™ product set. Client applications and services consume the virtual database services provided by Virtuoso using one or more of these drivers.

The features of these drivers are listed below:

- Blistering Performance
- ODBC Drivers are ODBC v2.5 compliant
- ODBC Drivers implement driver based Scrollable Cursors
- Drivers for JDBC are JDBC v1.02, v1.1, and v2.0 compliant
- Drivers for JDBC implement driver based Scrollable Cursors
- OLE-DB Data Providers are OLE-DB 2.0 compliant.

Security Manager

Virtuoso's security features are listed below:

- User Login and Password Verification
- Role Definition and Privilege Control
- Table and Column Level Privilege Control
- Logical "Rules Based" Security - enabling security to be devised around custom rules that derive security conditions from a combination of user, client applications, client operating system, and network address parameters, if required
- Data Encryption – data transmission between Virtuoso and its clients can be secured via encryption.

Query Manager

Virtuoso's query manager features are as follows:

- Supports ANSI SQL 92
- Parallel Query processing – enabling parallel execution plan assembly, query execution, and query fulfillment.
- Asynchronous Operation – all activities occur asynchronously except for transaction commits and rollbacks which occur synchronously.
- Distributed Join Optimizer - enabling high-performance heterogeneous joins across disparate database engines.
- Scrollable Cursor Engine - facilitates scrollable cursor support across heterogeneous database engines. This mechanism allows the manipulation of ResultSet subsets call RowSets that span multiple data sources and database engines. This sophisticated featured reduces record fetch

overhead by migrating chunks of data in parallel from external data sources into Virtuoso and then out again to client applications and services. Virtuoso supports Static, Keyset, Mixed, and Dynamic Cursors.

- Heterogeneous VIEWS support - enabling the development of logical data snapshots that transcend disparate database engines. These VIEWS aren't updateable.

- Generic Stored Procedure Language – enables the development of stored procedures that allow application logic to be stored and implemented at the VDB level, these stored procedures can reference tables hosted in disparate database engines. Virtuoso Stored Procedures are sensitive to metadata changes that occur within externally linked tables.

Metadata Manager
The Virtuoso Metadata Manger acts as the repository of all Virtuoso's database objects, this includes Metadata from external tables linked into Virtuoso, Native Tables, Stored Procedures, Views, Object Privileges, Indexes, Referential Integrity constraints, and Database Statistics.

The Metadata Manager does not affect any of the external tables linked into Virtuoso. Thus, an external object dropped by the Metadata Manager simply equates to a reference drop within Virtuoso as opposed to an actual object drop within an external database linked into Virtuoso.

Transaction Manager
The Virtuoso transaction manager possesses the following capabilities:

- Commits & Rollbacks – Transactions are either fully completed or not at all, when dealing with externally linked tables this behavior is preserved, but not to the point of a full 2-phase commit, this functionality will be available in a subsequent product release

- Supports read-only transactions, which provide no lock overhead and add flexibility to the VDB transaction model.

- Before & After Image Transaction Logging – Virtuoso maintains database state logs before and after transactions. Thus, in time of failure, it is possible to Roll the database back to a prior state preceding VDB server initialization. Virtuoso also allows you to roll forward your database by committing all uncommitted transactions held in After Image transaction logs when recovering from system failure.

- Supports Dirty Read, Read Committed, Repeatable Read, and Serializable transaction isolation levels

Concurrency Manager
The concurrency manager ensures that Virtuoso is capable of supporting multiple concurrent database sessions that execute data INSERTS, UPDATES and DELETIONS access without compromising data integrity or implicitly increasing application or service latency.

The Virtuoso Concurrency Manager supports Optimistic and Pessimistic (Locking) concurrency.

Database Rows, Pages, or Tables aren't locked when Optmisitic concurrency is in use, rather UPDATES or DELETIONS are simply rejected on the bases of other user process modifying the same record(s) that you are about to UPDATE or DELETE. Optimistic concurrency presumes low record UPDATE and DELETE frequency.

Pessimistic concurrency on the other hand presumes a high level frequency of record UPDATE and DELETION activity. The Virtuoso concurrency manager implements Page Level locking, and these locks have configurable timeout settings, which reduce the probability of record deadlock which inadvertently introduce application latency. Row level locking is currently in development.

Local I/O Manager
Virtuoso posses full native database engine capabilities, the local I/O Manager is responsible for writing data to local storage when Virtuoso is being used as an Embedded Database Engine. This feature of Virtuoso enables it to serve Application Server solutions as an embedded database engine outside the corporate firewall, providing these solutions with local storage at the same time reducing workload on the corporate databases behind the company firewall.

External Data I/O Manager
Virtuoso reads and writes data to external database engines using ODBC or UDBC as its external data I/O interface (supported for JDBC and OLE-DB based I/O is also planned).

ODBC support enables Virtuoso to link and manipulate data held in ODBC compliant database engines, implying any database with an ODBC driver is accessible from Virtuoso.

UDBC support enables Virtuoso to be hosted on operating system where ODBC isn't supported, it also enables Virtuoso to communicate with external databases using their native interfaces.

Data Replication Manager
The Virtuoso Replication Manager enables the automatic synchronization of data across Virtuoso Servers within your computing infrastructure. The replication process is bi-directional and is capable of replicating data across externally linked tables.

Synchronization of replicated databases is automatic should one of the database within a replica group encounter system failure.

Virtuoso Conductor
This is a HTML based Administrative console that provides a graphical user interface for attaching and detaching external tables associated with your Virtuoso database engine. It also includes configuration wizards and a query editor that enables intuitive configuration and testing of the ODBC and UDBC data sources names that you plan to use with Virtuoso.

Implementation Issues Checklist

Key

Implemented – I	Partially Implemented - PI	Planned - P

VDB Implementation Issue	**Status**
ODBC, UDBC, JDBC, OLE-DB based interface to VDB	I
ODBC & UDBC based external data I/O	I
JDBC based external data I/O	P
OLE-DB based external data I/O	P
SQL Support	I
OQL Support	P

Query Processor	I
Heterogeneous Scrollable Cursor Support	I
Standard Data Types Support	I
VIEW Support	I
Stored Procedure Support	I
Concurrency Control	I
Transaction Isolation	I
Distributed Transaction Support	PI
User Definable Type Support	I
Federated Database Support	I
Distributed Database Support	I
Security	I
Replication Manager	PI

Feature & Benefits Analysis (Virtual Database Functionality)

Feature	**Benefit**
Transparent concurrent access to heterogeneous data sources and backend database engines.	Cost effective mechanism for harnessing information from your existing databases and data sources in real-time.
ODBC, JDBC, UDBC, and OLE-DB Driver availability.	Provides exposure of virtual database services to the plethora of commercial ODBC, JDBC, UDBC, and OLE-DB based solutions currently in existence.
ODBC, UDBC, or Native external Data I/O.	Freedom to Mix & Match the best combination of database engines and data access types for your organization
Cross platform support (95/98/NT, Linux, Solaris, HP-UX, AIX, Digital UNIX, IRIX, SCO Unix, SCO Unixware, DG_UX, Dynix/PTX, BSDI, with VMS, MacOS, OS2, OS400, OS390 planned).	Freedom to Mix & Match the best combination of Application and Database server operating systems for your organization.
Multiple database and ODBC data source name support: Virtuoso Native DBMS Server, Oracle(6/7/8),	Sustains or bolsters database independence within your organization. This also ensures that new database driven

MS SQL Server(4/6.x/7.x), Informix(4/5/6/7), Progress (6/7/8/9), CA-Ingres (6.4), CA-OpenIngres (1.1/2.x), DB2 (2.12 & 5.x), Sybase SQL Server (4.x/10.x/11.x), PostgresSQL, Solid, Velocis, and third party ODBC Drivers.	technologies are usable with your existing database engines and data sources, thereby preserving database investments made in prior years.
Multi Threaded Virtual Database Server.	Parallel execution of data access instructions with minimal system overhead resulting in high-performance data throughput even when concurrent user counts are high.
Asynchronous Data I/O.	All data I/O except transaction commits and rollbacks are Asynchronous. Thus, reducing the probability of data I/O blocks that increase VDB server latency.
Scrollable Cursors Support.	Enables the use of High-Level data access interfaces such as RDO, ADO, OLE-DB, JDBC 2.0 against your existing databases and data sources.
Sophisticated Concurrency Control.	Provides scalability to your VDB based solutions.

Stored Procedure Support.	You can develop database independent stored procedures that can traverse all the databases in your organization using common stored procedure language.
Sophisticated rules based security.	You can control user privileges, define roles, and generally tighten security by adopting a "Rule Book" approach. This allows your security infrastructure be both logical and physical in nature, and driven by rules that you define.
Centralized or Decentralized infrastructure management via Web Browser	You can manage one or more geographically dispersed Virtuoso Servers from a single location or multiple locations through your Web Browser. Zero Admin solution.
Bi-Directional Heterogeneous Replication	Allows numerous Virtuoso and non-Virtuoso database tables to be kept in sync throughout your distributed computing infrastructure without user intervention. Reduces the cost typically associated with data transformation and migration. It also enables the development of sophisticated data caching for store-and-forward solutions.
Full Relational and Object-Relational Database functionality	Flexibility to build database solutions using relational or object-relational models. Future proofs your applications development at

	the data access level.
Small footprint	Can operate within 2MB of RAM and consumes no more than 10MB of disk space for entire product installation. This makes Virtuoso embeddable with solutions that are hosted on devices that have minimal memory and persistent storage capacity.

Product Packaging

Virtuoso is available in 3 distinct product formats:

- OpenLink Virtuoso™ Lite Edition
- OpenLink Virtuoso™ Workgroup Edition
- OpenLink Virtuoso™ Enterprise Edition

OpenLink Virtuoso™ Lite Edition

This is an entry-level product comprising the following components:

- HTML based Interactive ODBC compliant Query Console
- Virtual Database Server
- Virtuoso Conductor – HTML based remote database management console
- OpenLink Virtuoso™ Single Tier ODBC & UDBC Drivers
- OpenLink Virtuoso™ Type 4 Drivers for JDBC.
- Lightweight Thread Manager.

OpenLink Virtuoso™ Lite presumes the existence of ODBC or UDBC drivers within your existing IS infrastructure. It only includes ODBC and JDBC drivers for accessing the Virtual Database Server. It is important to note that prior to using Virtuoso, you must install and configure database specific network communications software and ODBC drivers provided by each of the database vendors whose products you will be exposing to Virtuoso.

Examples of such database vendor provided products include: Oracle SQL*Net, Oracle Net8, Informix I-Connect, Ingres NET, Microsoft NETLIB, Sybase Open Client, Progress Client Networking, DB2 client enablers etc. Each of these products typically includes a free ODBC Driver.

This product set supports the following operating systems:
Windows 95/98/NT, Linux, Solaris (Sparc & x86), AIX, HP-UX, IRIX, Dynix/PTX, DG-UX, Digital UNIX. MacOS, OS/2, and VMS ports are planned.

OpenLink Virtuoso™ Workgroup Edition

This is a departmental or small Internet/Intranet/Extranet server solution comprising the following components:

- HTML based Interactive ODBC compliant Query Console
- Virtual Database Server

- Virtuoso Conductor – HTML based remote database management console
- OpenLink Virtuoso™ Single Tier ODBC & UDBC Drivers
- OpenLink Virtuoso™ Type 1, 2, and 4 Drivers for JDBC.
- OpenLink Data Access Driver Suite (Lite Edition) supporting: Oracle, Microsoft SQL Server, Informix, Sybase, CA-Ingres, Progress, DB2, PostgresSQL, Velocis, and Solid
- Lightweight Thread Manager.

OpenLink Virtuoso™ Workgroup Edition includes OpenLink Software's high-performance Single Tier ODBC & JDBC Drivers (a.k.a. OpenLink Lite). These drivers enable you to expose your existing database engines to Virtuoso right out of the box. It is important to note that prior to using Virtuoso, you must install and configure database specific network communications software and ODBC drivers provided by each of the database vendors whose products you will be exposing to Virtuoso.

Examples of such database vendor provided products include: Oracle SQL*Net, Oracle Net8, Informix I-Connect, Ingres NET, Microsoft NETLIB, Sybase Open Client, Progress Client Networking, DB2 client enablers etc. Each of these products typically includes a free ODBC Driver.

This product set supports the following operating systems:
Windows NT, Linux, Solaris (Sparc & x86), AIX, HP-UX, IRIX, Dynix/PTX, DG-UX, Digital UNIX. MacOS, OS/2, and VMS ports are planned.

OpenLink Virtuoso™ Enterprise Edition

This is an enterprise wide or large Internet/Intranet/Extranet server solution comprising the following components:

- HTML based Interactive ODBC compliant Query Console
- Virtual Database Server – Implemented as an OpenLink Request Broker service
- Virtuoso Conductor – HTML based remote database management console
- OpenLink Virtuoso™ Multi-Tier ODBC & UDBC Drivers
- OpenLink Virtuoso™ Type 1,2,3 & 4 Drivers for JDBC
- OpenLink Data Access Drivers Suite (Multi-Tier Edition) supporting: Oracle, Microsoft SQL Server, Informix, Sybase, CA-Ingres, Progress, DB2, PostgresSQL, Velocis, Solid, and 3rd Party ODBC Drivers
- Heavyweight POSIX or Native Thread Manager.

OpenLink Virtuoso™ Enterprise Edition includes OpenLink Software's high-performance Multi-Tier ODBC & JDBC Drivers. These drivers enable you to expose your existing database engines to Virtuoso right out of the box, without inherent dependencies on database vendor provided network software for remote database connections.

Examples of such database vendor provided products include: Oracle SQL*Net, Oracle Net8, Informix I-Connect, Ingres NET, Microsoft NETLIB, Sybase Open Client, Progress Client Networking, DB2 client enablers etc.

This product set supports the following operating systems:
Windows NT, Linux, Solaris (Sparc & x86), AIX, HP-UX, IRIX, Dynix/PTX, DG-UX, Digital UNIX. MacOS, OS/2, and VMS ports are planned.

Galvanizing Your Enterprise with OpenLink Virtuoso™

Virtuoso enables you to maximize the benefits of new distributed computing technologies and paradigms with minimum disruption, if any, to your existing computing infrastructure. The situation examples in the sections that follow demonstrate how Virtuoso fulfills this value proposition.

Empowering Knowledge Workers

Your organization seeks to empower its knowledge workers by providing transparent access to corporate information so that they can make timely and accurate decisions.

Infrastructure Challenges:

- Your organizations core business solutions are all driven by database engines provided by different database vendors
- The disparate application databases are hosted on a range of different database server machines running different operating systems

Critical Success Factors:

- To maximize the uptake of this initiative, the knowledge workers must be abstracted from the intricacies of heterogeneous data access.
- Information must be available in real-time
- Information must be accurate
- Knowledge workers must be able to manipulate information and make decisions using their existing "off the shelf" desktop productivity tools.

Technology Challenges:

- The data residing in disparate data sources needs to be pulled together in real time to produce information
- High throughput transmission of information culled from disparate application databases to desktop productivity tools being used by the knowledge workers.

Technology Solution:
Implement a 3-tier distributed computing architecture utilizing ODBC, JDBC, or OLE-DB for data access and OpenLink Virtuoso™ as your "Data Junction Box".

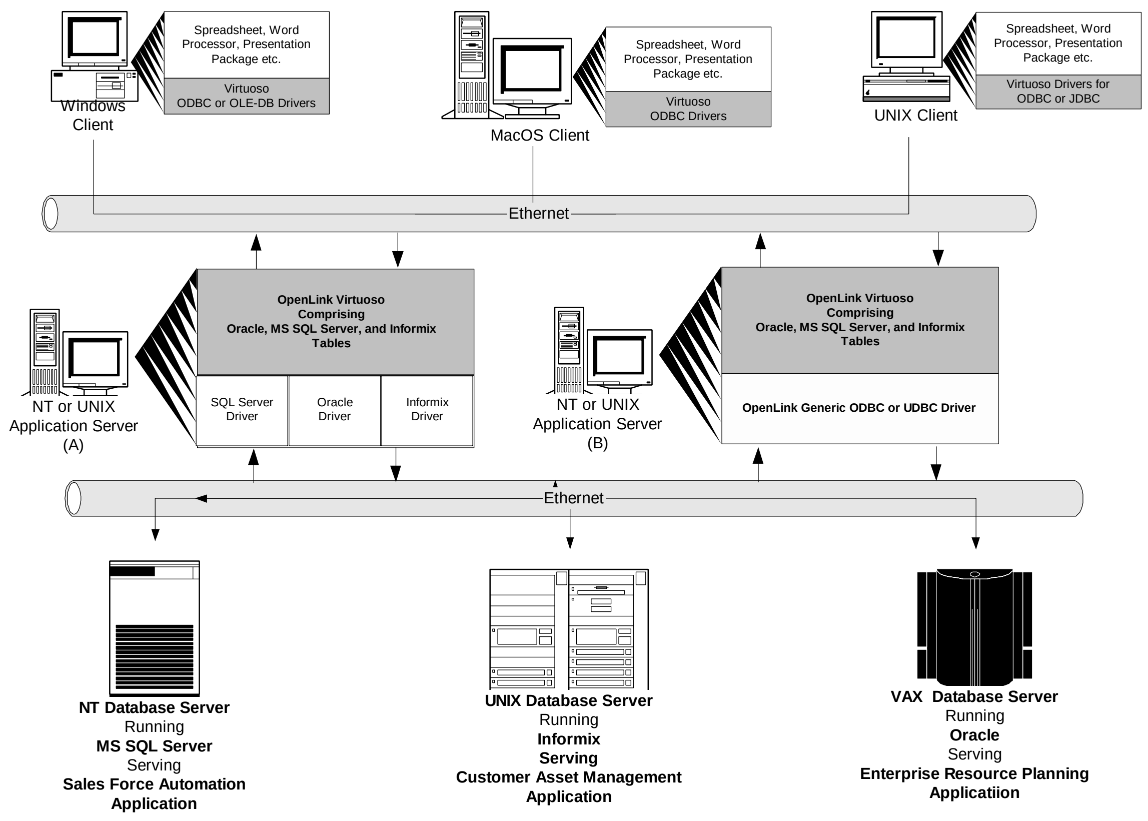
Solution Implementation:
The infrastructure required in order to fulfill the objectives set would be assembled as follows:

- Creation of "Data Junction Box" using a OpenLink Virtuoso™, that holds references to tables hosted by your application database engines
- Hosting one or more of your OpenLink Virtuoso™ servers on one or more designated server machines. These machines could be dedicated application servers or one of the many database servers currently in use
- Creation of ODBC or UDBC data source names on the same machine as the OpenLink Virtuoso™ server. These data sources names will connect OpenLink Virtuoso™ to the different application databases within your computing infrastructure
- Creation of ODBC, JDBC, or OLE-DB data source names that reside on the desktop machines used by your knowledge workers. The data source names created will be used to connect ODBC, JDBC, and OLE-DB based desktop productivity tools to your OpenLink Virtuoso™ servers.

Once the infrastructure has been assembled in the manner described in the previous section, the stage is set for your knowledge workers to commence exploiting the benefits of transparent access to information culled from heterogeneous data sources. The entire process occurs via the use of their chosen "off the shelf" desktop productivity tools.

The entire solution, its infrastructure, and constituent components are depicted below.

Figure 14 - Virtuoso Empowering Knowledge Workers



Spreadsheet, Word Processor, Presentation Package etc.
Virtuoso ODBC or OLE-DB Drivers
Windows Client
Spreadsheet, Word Processor, Presentation Package etc.
Virtuoso ODBC Drivers
MacOS Client
Spreadsheet, Word Processor, Presentation Package etc.
Virtuoso Drivers for ODBC or JDBC
UNIX Client
Ethernet
OpenLink Virtuoso Comprising Oracle, MS SQL Server, and Informix Tables
SQL Server Driver
Oracle Driver
Informix Driver
NT or UNIX Application Server (A)
OpenLink Virtuoso Comprising Oracle, MS SQL Server, and Informix Tables
OpenLink Generic ODBC or UDBC Driver
NT or UNIX Application Server (B)
Ethernet
NT Database Server Running MS SQL Server Serving Sales Force Automation Application
UNIX Database Server Running Informix Serving Customer Asset Management Application
VAX Database Server Running Oracle Serving Enterprise Resource Planning Applicatiion

Note: Databases depicted could be replaced with any ODBC compatible database engine.

Improving Customer & Prospect Interaction via Your Corporate Web Site

Your organization seeks to increase customer and prospect intimacy by implementing a database driven web site. The site offers customers and prospects online support, quotations, order processing, and product information.

Infrastructure Challenges:

- Your organizations core business solutions are all driven by database engines provided by different database vendors
- The disparate application databases are hosted on a range of different database server machines running different operating systems
- Security implications of corporate database exposure to the Internet
- Availability of "Web to Database" middleware for your Web Server.

Critical Success Factors:

- Data transmission throughput between internal databases and web pages
- Scalability of Web Servers, underlying "Web to Database" middleware, and participating database engines
- Information accuracy
- Data updates by site visitors must be routed to the appropriate internal database(s).

Technology Challenges:

- The data residing in disparate data sources needs to be pulled together in real time to produce information
- Delivery of high-performance information access to numerous concurrent site visitors
- Availability of operating system independent "Web to Database" middleware, providing you with the freedom to chose the most suitable operating system for the job at hand.

Technology Solution:
Implement a 3-tier distributed computing architecture utilizing ODBC, JDBC, or OLE-DB for data access and OpenLink Virtuoso™ as your "Data Junction Box".

Solution Implementation:
The infrastructure required in order to fulfill the objectives set would be assembled as follows:

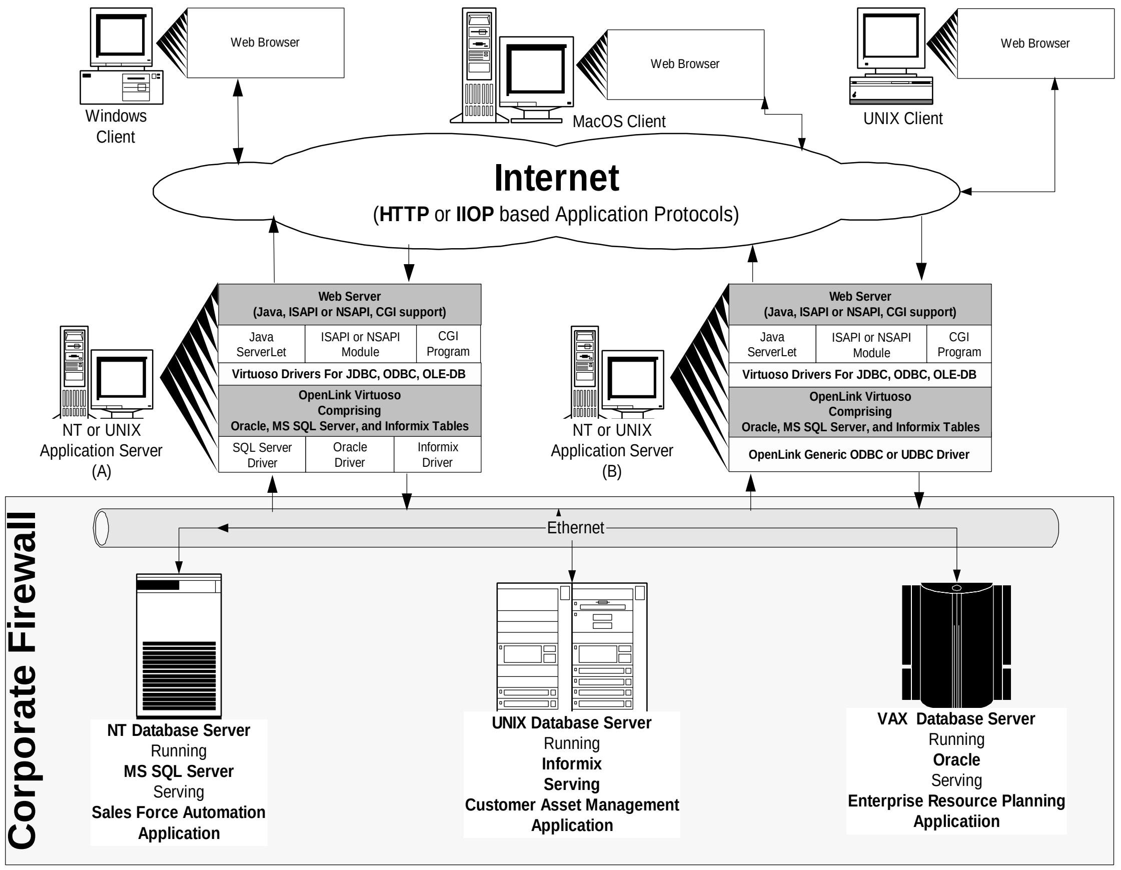
- Creation of "Data Junction Box" using a OpenLink Virtuoso™, that holds references to tables hosted by your application database engines
- Hosting one or more of your OpenLink Virtuoso™ servers on one or more designated server machines. These machines could be dedicated application servers, the same machine hosting your Web Server, or one of the many database servers residing behind your corporate firewall
- Creation of ODBC or UDBC data source names on the same machine as the OpenLink Virtuoso™ server. These data sources names should point to the different application databases within your computing infrastructure.

- "Web to Database" middleware and Application Server Development kit selection.
- Creation of ODBC, JDBC, or OLE-DB data source names on your Web Server for use by your "Web to Database" middleware for connectivity to OpenLink Virtuoso™
- Use the OpenLink Virtuoso™ Stored Procedure language to implement the data access and manipulation aspects of your application logic.

Once the infrastructure has been assembled in the manner described above, the stage is set for you to develop and implement your database driven web. The application development process retains a single as opposed to multiple database focus due to the sophisticated VDB Engine functionality provided by OpenLink Virtuoso™ without compromises in performance, security, or scalability.

The entire solution, its infrastructure, and constituent components are depicted in the illustration that follows.

Figure 15 - Internet Application Server Solution



Windows Client
Web Browser
MacOS Client
Web Browser
UNIX Client
Web Browser
Internet
(HTTP or IIOP based Application Protocols)
Web Server (Java, ISAPI or NSAPI, CGI support)
Java ServerLet
ISAPI or NSAPI Module
CGI Program
Virtuoso Drivers For JDBC, ODBC, OLE-DB
OpenLink Virtuoso Comprising Oracle, MS SQL Server, and Informix Tables
SQL Server Driver
Oracle Driver
Informix Driver
NT or UNIX Application Server (A)
Web Server (Java, ISAPI or NSAPI, CGI support)
Java ServerLet
ISAPI or NSAPI Module
CGI Program
Virtuoso Drivers For JDBC, ODBC, OLE-DB
OpenLink Virtuoso Comprising Oracle, MS SQL Server, and Informix Tables
OpenLink Generic ODBC or UDBC Driver
NT or UNIX Application Server (B)
Corporate Firewall
Ethernet
NT Database Server Running MS SQL Server Serving Sales Force Automation Application
UNIX Database Server Running Informix Serving Customer Asset Management Application
VAX Database Server Running Oracle Serving Enterprise Resource Planning Applicatiion

Product Features & Functionality Comparisons

There are a number of competing products that exist today, they may not be explicitly identified as Virtual Database products by their vendors but certainly, offer some of the capabilities described in this document.

Features Comparison Key:

1. Heterogeneous Joins	2. ODBC Based Ext. data I/O	3. Native data I/O	4. ODBC Driver availability	5. JDBC Driver availability
6. OLE-DB Provider availability	7. Web Browser based Admin	8. Cross Platform Support	9. Full Database Functionality	10. Multi-threaded

	VDB Type	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
OpenLink Virtuoso™	10	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Microsoft SQL Server 7	8	Y	Y	N	Y	N	Y	N	N	Y	Y
IBM DataJoiner	10	Y	Y	Y	Y	Y	N	N	P	P	N
Inprise BDE	3	N	Y	N	N	N	N	N	N	N	N
Microsoft JET	4	Y	Y	N	N	N	Y	N	N	P	N
Symantec DBAnywhere	5	N	Y	N	N	Y	N	N	N	N	N

Y - Yes, N – No, P - Partial

Conclusion

Virtual Database technology is clearly critical technology with inherent impact on Universal Data Access, Distributed Computing and the emerging Information Age.

The Internet as the distributed computing medium of choice will continue to create insatiable demand for information. This simply increases the pressure on information producers and consumers to retrieve data and then rapidly convert into information at ever increasing rates.

Internet/Intranet/Extranet sites will emerge as the dominant medium through which information is exchanged. Information once obtained would then be rapidly converted in to knowledge the ultimate basis of competitive advantage and power.

If the Information Age is the next point of call, and we accept that all Information comes from data? Then it is rational to conclude that Virtual Databases will lie at the heart of the Information Age. In short, there is a limited Information Age at best without the prevalence of Internet/Intranet/Extranet sites driven by Virtual Database Engines.