

MS Access 2003 Migration Guidelines & Procedures



Authored by

© Practical Computer Applications, Inc.
2000 Commonwealth Avenue
Newton, MA 02466
Phone 617.527.4722
www.pcapps.com

Table of Content

INTRODUCTION	4
<i>Reader Assumptions</i>	<i>4</i>
<i>Document Organization</i>	<i>4</i>
<i>About PCA</i>	<i>5</i>
<i>The Authors</i>	<i>5</i>
MS ACCESS MIGRATION OVERVIEW	6
<i>Migration Defined</i>	<i>6</i>
MIGRATION OPTIONS	7
<i>Standalone MS Access Application</i>	<i>8</i>
<i>Split MS Access Application</i>	<i>9</i>
<i>MS Access Data Store Application</i>	<i>10</i>
<i>MS Access Data Project</i>	<i>11</i>
<i>MS Access Client/Server Application</i>	<i>12</i>
<i>Client/Server Application</i>	<i>13</i>
<i>Web Browser Application</i>	<i>14</i>
<i>Smart Client Application</i>	<i>15</i>
<i>Note on Access Components</i>	<i>16</i>
MIGRATION PROCESS AND TOOLS	17
<i>Migration Preparation</i>	<i>18</i>
<i>Pre-Migration Testing</i>	<i>19</i>
<i>Optimize For Client/Server</i>	<i>19</i>
<i>Post-Migration Testing</i>	<i>19</i>
<i>Deployment</i>	<i>19</i>
MS ACCESS MIGRATION CRITERIA	20
<i>Migration Decision Criteria</i>	<i>20</i>
<i>SQL Server vs. Oracle</i>	<i>21</i>
PREPARING MS ACCESS FOR MIGRATION	22
DATA PREPARATION GUIDELINES	23
<i>Data Validation</i>	<i>23</i>
<i>Date Ranges</i>	<i>23</i>
<i>Naming Conventions</i>	<i>23</i>
<i>Keywords</i>	<i>23</i>
<i>Keys</i>	<i>23</i>
<i>Columns/Fields</i>	<i>23</i>
<i>Indexes</i>	<i>24</i>
<i>Unique Columns</i>	<i>24</i>
<i>NULL Values</i>	<i>24</i>
<i>Referential Integrity</i>	<i>24</i>

FRONT-END PREPARATION GUIDELINES	25
<i>Dynamic SQL</i>	25
<i>MS Access Merge Files</i>	25
<i>Forms</i>	25
<i>Controls</i>	25
<i>Data Access Controls</i>	25
<i>ActiveX Controls</i>	25
<i>3rd-Party Controls</i>	25
<i>Reports</i>	25
<i>Queries</i>	26
<i>Cross Tab</i>	26
<i>Upsize to Views</i>	26
<i>Parameters</i>	26
<i>Macros</i>	26
<i>Programming with VBA</i>	27
<i>Library References</i>	27
SQL SERVER MIGRATION PROCEDURES	28
<i>Overview</i>	28
<i>MS SQL Server Migration Procedures</i>	28
ORACLE MIGRATION PROCEDURES	30
<i>Overview</i>	30
<i>Oracle Migration Workbench Preparation</i>	31
<i>Oracle Workbench Migration Procedures</i>	32
DEPLOYING A MIGRATED MS ACCESS APPLICATION	34
<i>MDE</i>	34
<i>Security</i>	35
<i>Other Database Connectivity</i>	35
<i>Performance Enhancement Techniques</i>	36
<i>Suboptimal Access Migration Techniques</i>	36
<i>Data Caching</i>	36
<i>Forms</i>	36
<i>SQL Pass Through</i>	36
<i>Converting Select Queries to Views and Stored Procedures</i>	36
ADDENDUM	37
<i>Other Migration Tools</i>	37
<i>MS Access Developer Edition</i>	37
<i>MS Access Data Type Conversion Tables</i>	38
<i>MS Access Data Column Limitations</i>	39
<i>MS Access to Oracle Functions Conversion</i>	40
<i>List of Figures</i>	44

Introduction

This document is written by experts at Practical Computer Applications, Inc. (PCA), a custom software design and engineering firm located in Newton, MA. For over 15 years PCA has been migrating “office grown” MS Access applications to commercial database platforms for clients in every industry. This document captures our best migration techniques and procedures — and is designed to assist the experienced MS Access 2003 developer with migrating an MS Access application to the MS SQL Server and Oracle Client/Server database platforms.

Business requirements criteria is provided to assist with the decision *whether to migrate*, guidance is provided on *which migration path* makes the most sense, and specific MS Access *migration procedures* are provided for both the SQL Server and Oracle platforms.

Reader Assumptions


This document assumes that the reader is an experienced MS Access Developer comfortable with Visual Basic for Applications, MS Access Forms, Reports, and Queries — with a strong knowledge of either SQL Server or Oracle SQL syntax. Additionally, the reader is expected to understand how to normalize a database to the 3rd Normal Form (3F). More detail on 3F is available in the Addendum.

The reader is also assumed to have a working knowledge of database schema design and constructs, a functional knowledge of database referential integrity (RI), and a thorough understanding of table design including Primary and Foreign Keys, Indexes, Constraints, Views and Stored Procedures. Lastly, the reader is assumed to have the necessary security privileges that are required to add, edit and delete tables, views and stored procedures within the target database server. It is acceptable that a Database administrator creates the actual database and the reader functions with these security privileges.

Document Organization

These MS Access Migration Guidelines & Procedures contain concise step-by-step procedures, practical tips and techniques, useful examples, cautionary notes, and pointers to handy tools and resources.

Icon Key

 Tip or Technique

 *Cautionary Note*

 Reference Resource

About PCA

PCA designs and builds Custom Distributed Database applications for LAN and Internet deployment. We are recognized for our ability to understand unique and oftentimes complex line-of-business requirements — and effectively translate Clients' needs into well-structured database information models with intuitive, streamlined, and simple-to-use application interfaces. Origins at MIT and 15 years of experience that spans many different Industries and custom application segments allows our team to anticipate many of our clients needs in advance to provide practical, reliable and cost-effective database applications.

We frequently work with medium to large businesses in many Industries. Many of our Clients have developed a custom "office grown" database system with desktop productivity tools like MS Excel, MS Access and MS SQL Server — have outgrown the capabilities these tools can provide — and need database and business process design experts who can help understand their unique business needs, vet the options that are available, and develop a cost-effective & reliable database business solution.

The Authors

Pat Tormey (Author) is a Board Certified Professional Engineer (Massachusetts), Microsoft Certified Professional (Visual Basic & C# Web), and Board Certified Navy Engineering Duty Officer (Reserve). Pat earned a BS in Industrial Engineering/Operations Research (1981) and an MS in Engineering Management from the University of Massachusetts (1987). Mr. Tormey currently serves as the President of the New Hampshire .Net Users Group, the President of the Boston User Groups Inc., and is a Founding member of the Visual Basic Pro User Group.

Tripp Micou (Author) is CEO and Founder of Practical Computer Applications, Inc., and serves as the principal application design architect for many of PCA's larger and more challenging projects. Prior to PCA, Mr. Micou was a partner at Cardwell-Micou, a database consulting firm. Tripp holds a BS from The University of Michigan with Honors and a Masters of Science in Engineering from MIT. Tripp was a Graduate Fellow at the Hughes Aircraft Company from 1984 - 1987.

Kent Summers (Editor) is Executive Vice President of Practical Computer Applications, Inc., where he oversees sales, marketing and new business development planning and management activities. Prior to PCA, Kent served as CEO of Collego Corporation and MyHelpdesk, Inc. and VP of marketing at Electronic Book Technologies. Kent is a co-founding member of the W3C Technical Advisory Board (MIT 1994), and served as a past Director of the X Consortium and OASIS. Kent is an active mentor with the MIT Venture Mentoring Services program.

MS Access Migration Overview

Migration Defined

Typical MS Access applications are constructed around a single “MDB” file that contains the front-end User Interface Forms, Queries, code, etc., *and* the back-end data store component.



Figure 1: Standard MS Access Application Configuration

It’s quite common for an MS Access application to begin life as a single User productivity tool, and, as more and more people see the benefit of using the application, morph over time into a multi-user application. Real problems start to occur when an MS Access application that was originally designed for a single User grows in use, complexity (ad hoc features), and size of the data store. In addition, new multi-user class capabilities become necessary e.g. remote access, enhanced security, improved data integrity and application performance — capabilities that were unnecessary or completely irrelevant to the original MS Access application use and design intent.

“Migration” is the process of swapping out the back-end MS Access data store to either SQL Server or Oracle, and reconnecting the front-end MS Access Interface to the new back-end data store. Properly done, a migrated MS Access application will support more demanding business requirements. These requirements may include more users, more data, increased application reliability and performance, and enhanced application security.

Several different options are available for migrating MS Access applications. Each approach requires a different level of technical expertise and development effort — each approach also results in different application capabilities. The primary goal in any MS Access migration project is to select the migration option that is best suited to your business needs, and requires the least amount of effort and technical risk.

The various advantages that can be achieved by swapping out the back-end MS Access data store to either SQL Server or Oracle are limited. In some cases, a complete migration of the MS Access application may be necessary to meet more rigorous business requirements. Under this scenario, a well-designed MS Access Application serves as a Functional Prototype for Front-End migration as well.

Migration Options

The diagram below shows the various MS Access Front-end and Back-end migration configuration options that are available. As a general rule, the migration development effort and resulting application capabilities increase as you go down this diagram — from Tier 1 application configurations to Tier 3 solutions.

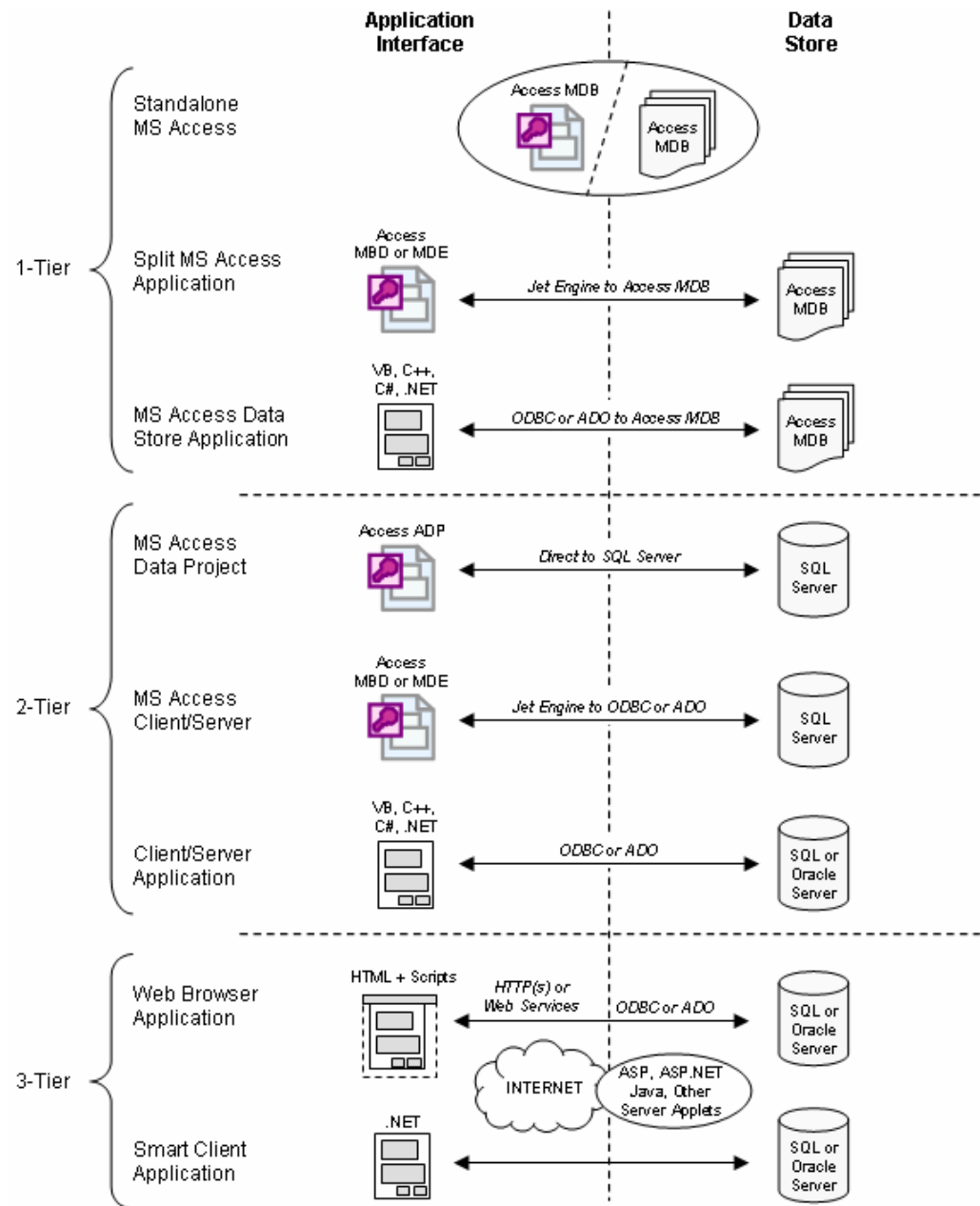


Figure 2: MS Access Migration Configurations

M I G R A T I O N O V E R V I E W

The table below provides a more detailed overview of the various MS Access Front-end and Back-end configuration options that are available:

Configuration Name	Architecture	Application Component	Data Store Component	Connection Method	Application Location	Data Store Location
Standalone MS Access Application	1 Tier	MS Access MDB		Jet Engine to Self	1 Desktop or File Server	
Split MS Access Application	1 Tier	MS Access MDB or MDE	MS Access MDB	Jet Engine to Data MDB	1 or more LAN Desktops	LAN File Server
MS Access Data Store Application	1 Tier	VB, C++, C#, .NET	MS Access	ODBC or ADO to MS Access	1 or more LAN Desktops	LAN File Server
MS Access Data Project	2 Tier	MS Access ADP	SQL Server	Direct to SQL Server	Multiple LAN Desktops	LAN SQL Server
MS Access Client/Server Application	2 Tier	MS Access MDB or MDE	MSDE, SQL Express, SQL Server or Oracle	Jet Engine to ODBC or ADO	Multiple LAN Desktops	LAN Database Server
Client/Server Application	2 Tier	VB, C++, C#, .NET	SQL Server or Oracle database	ODBC or ADO	Multiple Internet Desktops	Internet Database Server
Web Browser Application	3 Tier	ASP, ASP.NET, Java	SQL Server or Oracle database	HTTP(s) or Web Services	Web Server	Internet Database Server
Smart Client Application	3 Tier	.NET Smart Client	SQL Server or Oracle database	HTTP(s) or Web Services	Multiple Internet Desktops	Internet Database Server

Figure 3: MS Application Migration Table

Standalone MS Access Application

Typical MS Access applications are constructed around a single MS Access MDB file. This is the “default” MS Access configuration (absent a design decision, this is what you will get). While a Standalone MS Access Application is quite suitable to a single User’s business needs, it falls far short of meeting multi-user needs. This approach is also particularly difficult to maintain in a multi-user environment. For example, data is frequently lost or overwritten, application updates are difficult to deploy, and security is limited to only protecting well-intentioned End Users from harming the application.



A very common multi-user workaround for a Standalone MS Access application is to provide remote User access to the application via Citrix, Terminal Services, etc. While this does solve the immediate multi-user deployment need, it does not address problems inherent to sharing a single-user designed application. Over time, this common workaround will only exacerbate problems, and force a decision to properly migrate the MS Access application to a more appropriate multi-user environment.

Split MS Access Application

A superior method for deploying MS Access applications within a LAN environment among a relatively small group of End Users is to “split” the MS Access application into 2 MDB files: one which contains the Front-End Application components; and another MDB file that contains the Back-end Data Store component. This approach is generally useful for small workgroup environments e.g. an HR Department personnel management.

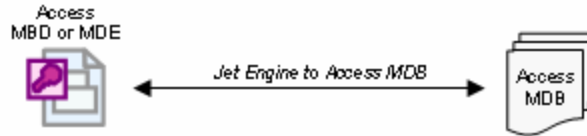


Figure 4: Split MS Access Application

It is also fairly common to use the Split Access method to support different MS Access Front-End applications that are connected to a single Back-End MS Access data store. For example, a single MS Access data store that contains Customer Contact data that is connected to MS Access applications in the Marketing and Customer Support.

Limitations to this approach generally include the ability to share across the Wide Area Network (WAN) or scale up to dozens of simultaneous users.

SCORECARD		
END USERS	< 5	LAN Only
PERFORMANCE	Slow	Depends on size of the database
RELIABILITY	Medium	Not for high availability
DATA INTEGRITY	Low	Reduced as the data file gets larger
SECURITY	Very Low	Access Internal only
MAINTENANCE BURDEN	High	Manual only Periodic Repair and Compact

MIGRATION OVERVIEW

MS Access Data Store Application

The MS Access data store design uses a single Back-End MS Access data store connected to a Windows application Client — typically written in Java or VB or one of the Dot Net languages. This approach is generally useful for applications that need to be deployed to a disconnected computer like a laptop or notebook PC for reference information (primarily “read” more than “write”), or data collection that is synchronized periodically to update and share data.

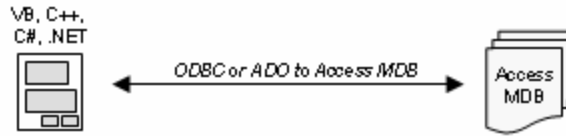


Figure 5: MS Access Data Store

The MS Access Data Store method can be used to house data on a network file share, however this approach forces MS Access to handle multi-user data concurrency issues, which are always best handled by a database server. Limitations to this approach generally include data integrity and security issues inherent with all MS Access file based solutions. As a desktop product, this model does not lend itself to centralized maintenance; MS Access Repair and Compact must be run by the end user. In the end, this design is as costly to develop and deploy as a full Client/Server application, but lacks the full Client/Server benefits of centralized Security, better performance, and ease of maintenance.

SCORECARD		
END USERS	1	Desktop Only
PERFORMANCE	Medium	Application design dependent
RELIABILITY	Medium	Application design dependent
DATA INTEGRITY	Medium	Application design dependent
SECURITY	Very Low	Access internal security
MAINTENANCE BURDEN	High	Manual only - Requires user to maintain

**MS Access
Data Project**

An MS Access Data Project (ADP) replaces the native Jet Engine data store with a SQL Server data store. All Queries are replaced with SQL Views, Functions and Stored Procedures. The design interface supports editing the SQL content in the MS Access ADP project file. No need to migrate MS Access ADP applications to SQL Server or Oracle because — they already are Client/Server Applications.

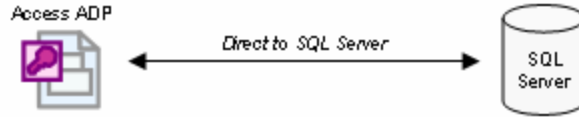


Figure 6: MS Access Data Project

Developing MS Access Data Projects does make the MS Access application “available on the Internet,” however this approach does require that all Users run the application through the MS Internet Explorer browser, and each User must have a locally installed copy of MS Access (or the appropriate MS Access Runtime engine) for the application to function.

ADP applications are limited primarily due to more complex deployment issues e.g. MS Access Runtime and/or IE Browser availability and version issues. While overcoming some of the inherent MS Access multi-user deployment limitations, this approach can also suffer from poor performance (vs. standard web-based applications) due primarily to the start up of the local instance of MS Access. Data security, reliability and integrity are entirely dependent on the SQL Server design.

SCORECARD		
END USERS	Any	IE Browser
PERFORMANCE	Medium	MS Runtime Start-up Expense
RELIABILITY	Medium	Application design dependent
DATA INTEGRITY	High	Application design dependent
SECURITY	High	Uses SQL Server or Oracle Security
MAINTENANCE BURDEN	Low	Standard SQL Server or Oracle maintenance

MS Access Client/Server Application

The MS Access Client/Server application architecture uses MS Access converted to an MDE format for deployment that is connected to any ODBC-compliant data store. The MS Access Client/Server design represents the most common migration approach.

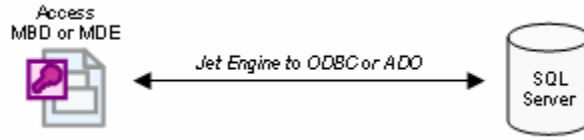


Figure 7: MS Access Client/Server Application

Properly designed to use the Database Server performance and security, the MS Access Client/Server approach supports rapid development, ease of maintenance, data transactions and fairly robust performance. MS Access or its runtime engine is required for deployment, and the End User can easily get direct access to the data tables and queries.

Typically deployed to a network share, MS Access Client/Server applications are recommended for non-regulated, shared data across a LAN, for example, inter-department management of statistics and production analysis.

MS Access Client/Server applications are limited to LAN deployment. Large Front End applications can be slow to load over a WAN. The MS Access Client/Server does expose the data tables directly to the End User, which may represent a security or data integrity issue.

SCORECARD		
END USERS	Any	LAN only. MS Access is Required
PERFORMANCE	High (following Access Startup)	Application design and local PC memory dependent
RELIABILITY	High	Application design dependent
DATA INTEGRITY	High	Application design dependent
SECURITY	Medium	Uses SQL Server or Oracle Security, but exposes data tables to End Users
MAINTENANCE BURDEN	Low	Standard SQL Server or Oracle maintenance

Client/Server Application

Client /Server applications are typically written in Java or one of the .Net products and are designed to run against and a Data Server, resulting is high performance, tight security and the most robust user experience. Deployment can be automated and maybe restricted to MS Windows, True Client/Server applications are designed for computers that are constantly connected the to network.

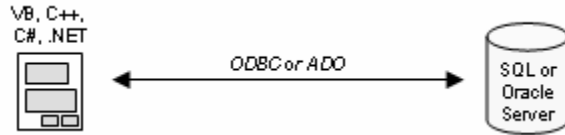


Figure 8: Client/Server Application

As a true Client/Server application, with a relatively small memory requirement, this approach enjoys standard Windows deployment and security coupled with high-end database server performance.

The architecture is recommended for any application where security and performance are important. Examples include data entry into regulated systems for accounting, medical research and personnel management.

Client/Server applications require a full time reliable connection to the data store through the Local, wide area connection or secure Internet connection.

SCORECARD		
END USERS	Any	Standard Desktop Application
PERFORMANCE	High	Application Design Dependent
RELIABILITY	High	Application Design Dependent
DATA INTEGRITY	High	Application Design Dependent
SECURITY	High	SQL Server or Oracle Security
MAINTENANCE BURDEN	Low	Standard SQL Server or Oracle maintenance

Web Browser Application

Web Browser based applications hosted either internally on intranet server or externally as internet server typically require no installation on the client computer, but are restricted to the tools a browser can support. Typically Web Browser applications host their data in a database server with provides the scalability, performance and the database level security.

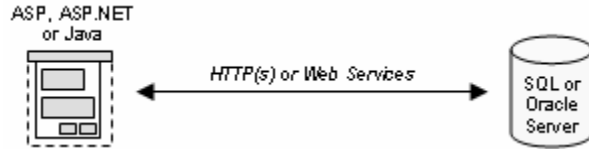


Figure 9: Web Browser Application

Web based application are generally easy to deploy and update and require no installation in the client computer. Well designed Web Applications run under any browser and may deploy disparate operating systems, to handheld computers and even to web enabled cell phones.

Web applications are typically less secure than desktop applications because the interface is asynchronous and which can be exploited by malignant individuals. The complexity of Web applications security and combined with inherent database security issues limits the Web browser Application to non critical data. Intranet (private) applications are inherently much more secure than internet (public facing) applications, simply due to the limited accessibility of the user.

SCORECARD		
END USERS	Any	Browser Deployment
USABILITY	Medium	Limited to simple forms and linear workflow
PERFORMANCE	High	Usability Design Dependent
RELIABILITY	High	Application Design Dependent
DATA INTEGRITY	High	Application Design Dependent
SECURITY	High	Uses Database Server Security and Web Server Security
MAINTENANCE BURDEN	Low	Standard Database Server Maintenance

Smart Client Application

The Smart Client architecture approach enjoys the same advantages of the Client/Server method and supports the full rich User Interface of MS Windows applications. Smart Clients applications can be designed to operate in connected and “occasionally connected” models.

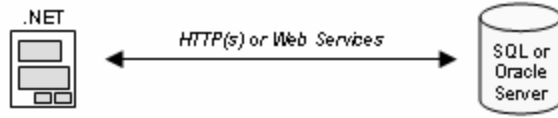


Figure 10: Smart Client Application

Occasionally connected designs support the user when off the local area network, caching the data locally and synchronizing data including update automatically when the connection is restored making them ideal of portable devices like Tablet PCs and the new range of handheld devices.

SCORECARD		
END USERS	Any	One-click deployment thru web or email link
USABILITY	High	Full Windows GUI
PERFORMANCE	High	Application Design Dependent
RELIABILITY	High	Application Design Dependent
DATA INTEGRITY	High	Application Design Dependent
SECURITY	High	Uses Database Server Security and Desktop Security
MAINTENANCE BURDEN	Low	Database Server Standard

Note on Access Components

There is no compiled “executable” application for any configuration that uses MS Access. MS Access itself INTERPRETS the code, reports and form data objects in the front-end to access the back-end database.

While the Microsoft Jet Engine is continually updated with many quality, functional, and performance improvements, at the end of the day it is still a file-based, desktop-class data store (versus a more robust Server-Based data store). The Jet Engine is not intended (or architected) to support more demanding business application requirements. These can include more than several End Users, usage from multiple locations, large sets of data, more complex functionality, and the need for enhanced application security. Common indicators that you are pushing the envelop on the native MS Access capabilities include: slow performance, poor application reliability, a security breach, data corruption, and the like.

To overcome the limitations inherent to the MS Access data store, MS Access can be configured to work with most 3rd-party back-end SQL data sources e.g. Oracle, SQL Server (or any other ODBC compliant or OLEDB compliant 3rd-party back-end data source). MS Access Migration refers to the procedure of transitioning the data store/management processing from the native MS Access Jet Engine to a 3rd-party server.

The front-end MS Access Application (the User Interface component) remains — and is linked to the 3rd-party data source. This process is also commonly called “Upsizing” the MS Access application. Depending upon the business requirements, it might be necessary to re-write the Front End Application to meet application security, performance, and other more demanding business application requirements.

When the migration is complete, all the data will be stored/hosted on the database server, and the MS Access application will continue to operate smoothly as the desktop application. It is important to start your migration with a well-designed MS Access application to avoid “Garbage In, Garbage Out.”

Migration Process and Tools

There are two ways to migrate an MS Access application: manual migration, or using the built-in Upsizing Wizard that is part of MS Access. The manual process requires that you create the necessary tables in the target DB Server through a simple export and then link the new tables back to the MS Access front-end application. The alternative is to use the built-in Access Upsizing Wizard (AUW) or the Oracle Migration Workbench (OMW). We recommend that beginners start with the appropriate Wizard, and complete whatever outstanding tasks are required by hand. Once you get the hang of it, it is more reliable and quite straightforward to follow the manual process.

The flow diagram below covers the recommended steps to migrate MS Access to SQL Server and Oracle. Start with preparing and testing the MS Access data, security and application front-end, running the respective migration tool, retesting the migrated application against the new data source, then deploying the migrated application.

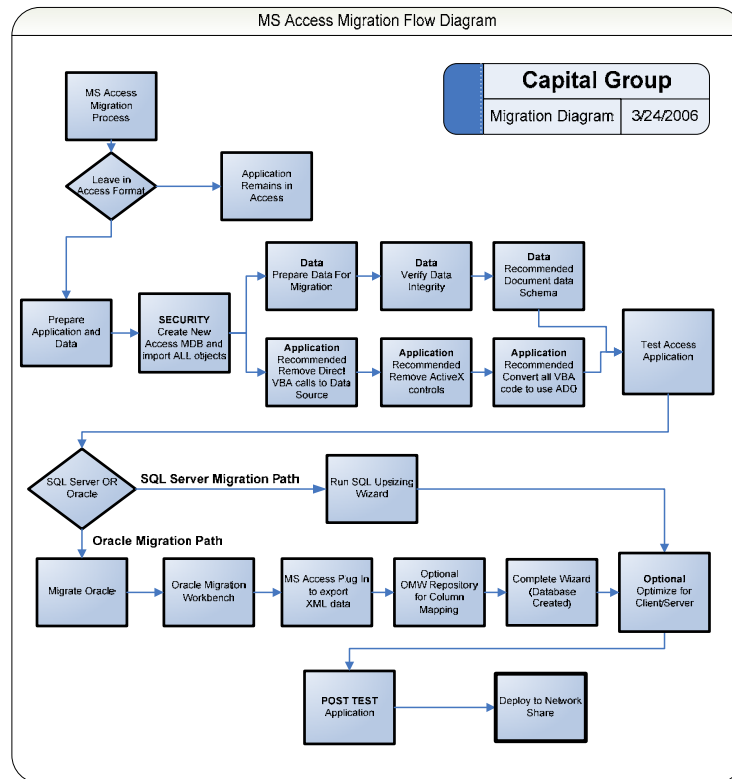


Figure 11: MS Access Migration Flow Diagram

Migration Preparation

Security

SECURITY
Create New Access MDB and import ALL objects

MS Access Security must be removed for data migration. The Server cannot use MS Access security, nor will the Access Front End application need it following the migration. Security preparation requires that you create a new MS Access MDB file and Import all the objects into the new MS Access MDB.

Data

Data
Prepare Data For Migration

Check the Data schema and table structures to insure that a primary key exists for each table. Any Table without a Primary Key will be treated by the Server as read-only. Also Set the Referential Integrity and Verify database normalization

Data
Verify Data Integrity

It is recommended to document the MS Access Database Schema for comparison after the migration is complete.

Data
Document data Schema

Application

Application
Recommended
Remove Direct VBA calls to Data Source

This is the process to prepare the MS Access Forms, Reports, and Query objects. Many MS Applications will run reasonably well if one or all of these steps are ignored. This step avoids performance and deployment issues that typically occur when these steps are ignored.

Application
Recommended
Remove ActiveX controls

Remove direct VBA calls to the data tables (direct VBA Calls to the data tables probably use a separate security model with its own connection string), remove all ActiveX controls (ActiveX controls frequently do not deploy very well), and convert all VBA code to use ADO (ADO is the Microsoft Data Access tool that superseded DAO the default data tool in MS Access).

Application
Recommended
Convert all VBA code to use ADO

Pre-Migration Testing



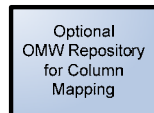
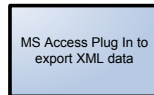
Test the MS Access application before proceeding. Both Oracle’s and Microsoft’s data Migration wizards are a one- way trip. It is much easier to remedy potential migration problems in MS Access than to deal with them in Oracle or SQL Server.

SQL Server Track



Access provides a built in SQL Server Upsizing Wizard that quickly handles the migration of the data without necessarily modifying any component. For SQL Server migration simply run the Upsizing Wizard and run the application unchanged. Let testing determine if any performance or other modifications are needed.

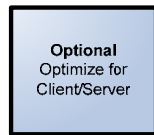
Oracle Track



MS Access to Oracle migration requires a two-step process handled by the Oracle Migration Workbench and the MS Access Plug in. The first step is to extract the MS Access data and schema into a set of text (XML) files. The second step is to import these text files into the OMW for Oracle to use in the actual Migration.

Optionally, the Workbench tool supports mapping the structure during data import and scripting the import for re-use with production data after testing.

Optimize For Client/Server



MS Access is not natively Client/Server-aware. With most applications MS Access does a fair job of handling the data. True Client/Server performance requires MS Access to use the full data handling of the Server data source. Converting MS Access Queries to Views and Stored Procedures effectively uses the Database Server.

Post-Migration Testing



Also known as full system integration testing, the Post-migration Test should exercise every aspect of the Migrated application to be sure there are no performance or data handling anomalies.

Deployment



Deploying the MS Access Front-End application requires converting the fully-tested MS Access MDB file to a “complied” MDE state, copying the MDE file to a restricted access network share, setting the End User security on the database server, and installing any supporting applications on the End User’s computers if necessary (any ODBC drivers, or other third-party software must be installed on each end user’s computer).

MS Access Migration Criteria

The decision to migrate from the native MS Access back-end to a Client/Server configuration is driven by a variety of existing and/or future anticipated business requirements. Topping the list is the need to concurrently share business information in a high performance and secure environment.

Migration Decision Criteria

The table below provides a high-level overview of the different business requirements that influence a decision to migrate, the limitations of MS Access relative to these requirements, and suitability of the MS SQL Server and Oracle platforms for meeting these more rigorous business requirements.

BUSINESS REQUIREMENT	MS ACCESS LIMITATIONS	SQL SERVER CAPABILITIES	ORACLE CAPABILITIES
End Users	LAN: < 10 Concurrent End Users WAN: Not Recommended INTERNET: Not Recommended	> 10 Concurrent End Users WAN: OK INTERNET: Not Recommended	> 10 Concurrent End Users WAN: OK INTERNET: Not Recommended
Front End	MS Access databases are appropriate for MS Access Front-Ends and Visual Basic Front-Ends only. Web Front-Ends should never be used with MS Access back-ends	SQL Databases are appropriate for all Front-End applications including MS Access, VB, C++, C#, .NET, ASP, ASP.NET, Java, and any other Web or Desktop Front-End	Oracle Databases are appropriate for all Front-End applications including MS Access, VB, C++, C#, .NET, ASP, ASP.NET, Java, and any other Web or Desktop Front-End
Security	MS Access should not be used to manage any data that is sensitive, company confidential, or regulated	Appropriate for sensitive, company confidential, and regulated data	Appropriate for sensitive, company confidential, and regulated data
Data Size	< 200 MBs. MS Access inherits a 2 GB file limit from Windows. In practice, MS Access "self limits" on performance at .5 GB.	> 100 MBs	> 100 MBs
Application Complexity	MS Access should not be used where the number of tables or forms or reports exceeds 100 objects, or requires intense graphical objects. Such objects tend to corrupt MS Access applications.	Appropriate for any level of complexity or graphics intensity.	Appropriate for any level of complexity or graphics intensity.

MIGRATION CRITERIA

BUSINESS REQUIREMENT	MS ACCESS LIMITATIONS	SQL SERVER CAPABILITIES	ORACLE CAPABILITIES
Application Availability	MS Access should not be used for availability needs under 8 hours. MS Access does not support transactional or log level operations.	Appropriate for sub-8 hour application availability requirement	Appropriate for sub-8 hour application availability requirement
Data Recovery	It is not possible to back-up MS Access while anyone is using the data. All changes to data between periodic back-ups are therefore at risk. MS Access does not support Fail-Over or transactional log-based back up or recovery.	Appropriate for most data availability and recovery requirements	Appropriate for most data availability and recovery requirements
MS Desktop Integration	Seamless (built-in) integration with all MS Office applications	Configurable integration with all MS Office applications	Data export/import work-arounds required
Performance	Poor quality application design, Number of End Users, Size of database, and Network traffic each negatively impact MS Access application performance	SQL Server is designed to support a large number of users across both the Wide Area and Local Area Networks. The greatest performance limit is poor Database Schema design followed by ineffective application design. Occasionally network throughout can also be detrimental to performance.	Oracle is designed to support a large number of users across both the Wide Area and Local Area Networks. The greatest performance limit is poor Database Schema design followed by ineffective application design. Occasionally network throughout can also be detrimental to performance.
Regulatory Compliance	MS Access does not meet Sarbanes Oxley, HIPPA requirements or SAS 70 Audit standards	Appropriate for Sarbanes Oxley, HIPPA and SAS 70 requirements	Appropriate for Sarbanes Oxley, HIPPA and SAS 70 requirements

SQL Server vs. Oracle

An important but often overlooked business objective is to add new application capabilities over time to improve the overall business process — while keeping the institutional data as centralized as possible to promote ease of data reuse and maintenance. From a purely technical perspective, neither Oracle nor SQL Server has any decided or demonstrable advantages as a target database migration platform. The decision to migrate to one database platform vs. the other should be based primarily on the business objective above: where the application data is already stored or usefully available.

For example, if the MS Access is being used to monitor, analyze and report against a SQL Server-based accounting system, or if integration with MS Office (Excel, Word, InfoPath, Outlook or PowerPoint) is a primary consideration, then SQL Server is the most suitable target platform.

Preparing MS Access for Migration

This section is directed at preparing the MS Access data for migration to the SQL Server or Oracle database platforms. MS Access applications typically migrate to SQL or Oracle data sources without any modifications to the MS Access application itself. However, most migration problems only become apparent *after* the application is deployed (when it's too late). If not properly prepared, MS Access applications can be difficult to install or will exhibit unacceptable performance limitations.

This section presents the recommended guidelines for preparing the MS Access application for migration — to avoid post-deployment issues — and provides some guidance on performance enhancement techniques.

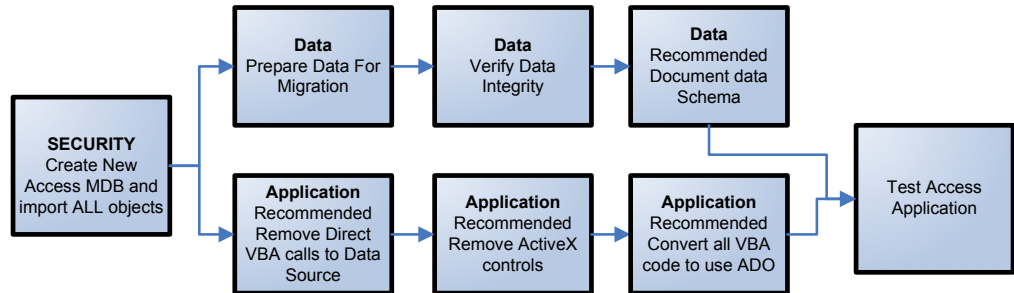


Figure 12: Migration Preparation Procedures

Data Preparation Guidelines

Data Validation

It is highly recommended that you validate the MS Access data, relational integrity and data structures *before* migration. It is far easier and faster to remedy data problems in the source MS Access application than deal with dirty integrity issues in the target database platform.

Data validation requires that, among other things listed below, you insure that the Data Schema makes sense; all Primary and Foreign Keys exist, and are of the same data type; and Referential Integrity is working properly. Included below are several additional data validation tips that will make your migration to SQL Server or Oracle much smoother.

Date Ranges

MS Access supports extreme date ranges that can cause problems with both Oracle and SQL Server. Any rows of data that contain extreme dates will not migrate, and will cause an error in the target database platform.

- Oracle: verify that date ranges are between 1/1/4712 BC and 12/31/4712 AD
- SQL Server: verify that all Dates/Time are in the 1/1/1753 to 1/31/9999 range. SmallDateTime must be between 1/1/1900 and 6/6/2079

Tables

All tables — with the exception of System Tables and any Tables marked as hidden — are routinely migrated to SQL Server or Oracle. All MS Access Objects that are stored within internal System Tables are not useful to either target Database Server.

Naming Conventions

Check your Table names to insure they do not contain spaces, punctuation, or symbols (including the common use of “#” or “\$” or “-“).

Keywords

Avoid “SQL Server” and “Oracle” Keyword for names of Tables and columns. Also avoid using generic keywords such as “Name”, “Property”, “Date”, “Comment”, “Desc”, “Text”, “Table”, “Index” etc.

Keys

Each and every table should always have a Primary Key, whether the table is stored in MS Access, SQL Server or Oracle. Any Table without a Primary Key will not be editable when the Migration is complete (it will be interpreted as “read only” by both Oracle and SQL Server). Note that Oracle does not support Cascading Delete or Cascading updates to Primary Keys.

Columns/Fields

Normally the OMW and AUW Wizards do an acceptable job of converting the data to a reasonable data type. In addition, the OWM process allows you to fine-tune the candidate Data Schema, prior to committing the data upload.

Indexes

Both AUW and OMW translate indexes as part of the migration process. Insure that indexes represent the actual business process being modeled by the MS Access application. After migration, check that the Primary (unique) and Foreign Keys (Duplicate) are indexed appropriately.

Unique Columns

Not all unique columns need to be in part of the Primary Key. If Employee number must be unique, then index the column uniquely. Note that data marked as Unique cannot be NULL — and the End User simply may not have the data available – you therefore might need to set a default value such as “N/A” or blank, or use sequential or random values.

NULL Values

We recommend that most fields should be marked as “Required” in MS Access or “Not Null” in SQL Server and Oracle. Unless the NULL value can be justified by a business requirement (as often occurs in Date fields), NULLs should be avoided.

Referential Integrity

MS Access optionally migrates Referential Integrity as part of the Upsizing Wizard. The Oracle Migration Workbench does not migrate Referential Integrity. After Migration to Oracle, be sure to set up Referential Integrity. Set all relationships using Access Declarative Referential Integrity design tool Tools → Relationships.

Beware: The MS Access Upsizing Wizard often migrates Referential Integrity (especially Cascading) as Triggers, and not as Declarative Referential Integrity (DRI). This behavior is sub-optimal, and requires substantial knowledge of T-SQL and trigger behavior to properly understand and manage.

It is important that Referentially Integrity (RI) exactly match the source MS Access RI.

Front-End Preparation Guidelines

Dynamic SQL

MS Access SQL is not entirely ANSI standard. MS Access supports the use of Dynamic SQL in VBA code modules. This must be converted to comply with the target Back-End data format.

MS Access Merge Files

MS Access supports its own unique brand of replication. MS Access Replication is not supported by either SQL Server or Oracle, and must be resolved before migration. Merge All Replica data prior to migration.

Forms

Ordinarily MS Access Forms require no modification prior to migration. However, Forms that return particularly large data sets e.g. a complete customer list, may cause performance issues, and therefore should be redesigned to return the least amount of data needed to accomplish the intended function. Slow Forms can be the result of a Combination Box loading all records unnecessarily.



Advanced Data Caching techniques can be employed to solve many performance issues associated with moving from a desktop application to a client-server application. Discussion of such techniques is beyond the scope of this document.

Typically MS Access Forms are built directly on Tables or Queries that return all rows of data. If the data is large enough to merit migration to a Client-Server system, then select Forms should be converted to use Parameters. Forms should avoid requesting more rows of data than are required to present to the End User.

Controls

Data Access Controls

Native MS Access Controls typically migrate without any problems. If you anticipate that performance may be an issue, look into the data access controls (i.e. ComboBox) to insure that the data source retrieves the least amount of data required to present to the End User, and that the SQL syntax is compatible with the new data source.

ActiveX Controls

Generally speaking, non-native ActiveX Controls are difficult to deploy. MS Access standard deployment techniques do not recognize missing or non-current versions of ActiveX Controls, and present obtuse error messages that are difficult to trap and debug.

3rd-Party Controls

3rd Party Controls, including the MSCalendar control, are ActiveX controls and should be avoided whenever possible. MS Access deployment doesn't have the ability to detect the correct version of third party controls.

Reports

Reports are also built on Tables or Queries but typically less sensitive to performance problems. Normally, no changes are required to Reports after migration. If you anticipate that performance may be an issue, read-only "Pass-Through Queries" are substantially faster at retrieving data especially for joins and sorting. Alternatively, manually converting Form and Report Record Sources to SQL Views and Procedures

can insure optimal performance, usually generating significant performance improvements over native MS Access.

Queries

Queries are not migrated by either wizard (OMW attempts to migrate Queries, but it is successful only on the simplest Queries). You can achieve a substantial improvement in performance by converting all existing MS Access Queries to Client-Server compatible “Views” and “Stored Procedures.”

Generally speaking MS Access handles Queries to Upsized data sources without any problems. As MS Access is not designed for Client-Server optimization, performance issues can be resolved by converting the Query to a Server View or Stored Procedure and linking it back to MS Access or calling it directly from an MS Access Form or Report.

Cross Tab

Cross Tab Queries do not convert readily to SQL Server or Oracle. Re-design these Queries the appropriate back-end SQL format. Note that SQL Server 2005 supports Cross Tabs natively.

Upsize to Views

Converting Queries to Views and Stored Procedures is a recommended practice. Views and Stored Procedures are optimized by the Client-Server system and prevent the MS Access application user from editing the design of the View (this assumes that SQL /Oracle’s security is set correctly).

Parameters

Where possible, Queries that return large data sets should be converted to take Parameters. This will avoid unacceptable delays opening Forms.

Macros

Never deploy any Application that uses MS Access Marcos. Convert all MS Access Marcos to their VBA equivalent. Macros do not support parameters. To open ten Forms or Reports with Macros requires ten Macros seriously bloating the application. More importantly, MS Access Macros do not support standard forms or error handling. Macro errors display an MS Access System error that is typically both incomprehensible and unrecoverable.

**Programming
with VBA**

Proper preparation of the application for upsizing the data to any Client/Server application may require converting the existing Visual Basic for Applications (VBA) code to support the new connection. VBA code that explicitly calls a direct link to the data will include its own connection string *including the security context*.

It is probably unnecessary to edit the code *unless* the VBA code calls the data directly. Only recode the program if there is a performance issue. Note that any changes to code will necessitate retesting of the application.

The MS Access Data interface called DAO and RDO are older technologies that have been superseded by a new data interface called ADO.

Library References

No Unused Library References should remain in MS Access 2003.

To be certain, remove the Microsoft DAO Object Library and the Microsoft Jet and Replication Library. Open the VBA editing Window, select Tools → References, and uncheck all references as shown.

SQL Server Migration Procedures

Overview

MS Access contains the “SQL Server Upsizing Wizard” expressly for the purpose of porting the MS Access data to SQL Server. Depending upon the amount of data, the Wizard is both quick and efficient. The Wizard creates the SQL Server Database and all the tables and column definitions dynamically.

MS SQL Server Migration Procedures

MS Access to SQL Server migration is accomplished using the built-in MS Access Upsizing Wizard (AUW).

Before Migration, make sure to follow the Front-End Preparation Guidelines and Data Preparation Guidelines presented in this document. Additionally the operator must be a SQL Server Administrator or the Database Owner for the Target SQL Database.

The time to complete this operation is dependent on the number of tables, the amount of data in the tables, and the SQL Server connection speed.

Migration Steps

- 1** Before running the Wizard set the ODBC timeout to 0 to avoid timing out during the Upsize Wizard’s run time
- 2** Create a New Access MDB called <ApplicationName>_FE.mdb, and import all the objects from the MS Access application. Appending “_FE” (Front End) to the MS Access Manager application component is a recommended practice. If the application uses an external Access Data MDB then Import all of its objects too.
- 3** Repair and Compaq the new MS Access file you just created to avoid compilation errors during the migration.
- 4** Unhide all Tables except System Tables (Msys)
- 5** Insure that all tables have a Primary Key, or they will be interpreted as read-only
- 6** Run the Access Upsizing Wizard: Tools → Database → Upsizing Wizard

7 On the Wizard menu, allow the Wizard to create a database for this application, or select the existing SQL Server blank database, and Click NEXT. Do not upsize any MS Access tables to existing SQL Server Databases with data in them.

8 Select All Tables from the Wizard menu and click NEXT

9 Select all options including Declarative Referential Integrity DRI. If you do not elect DRI the Wizard will generate SQL Triggers to support Referential Integrity. Triggers are Code placed on Database Events e.g. Add, Delete and Edit. Optionally, allow SQL Server to Add timestamps to each and click NEXT. Timestamps are a special purpose column that increments distinctly whenever the data row has a changed value, providing a great tool for determining concurrency.



Notwithstanding its name, a TimeStamp data type has NO relation to dates or time, and cannot be used to “stamp” a record with a particular date or time, or retrieve the date or time a record was updated. At most one TimeStamp column can be added to any table.

10 Tell the Upsizing Wizard to Link the tables back to the Existing Application. This will cause the original tables to be renamed <TableName>_Local, and the Wizard will name the new tables the same as Access tables. Link SQL Server is the instruction to link the tables back. This can be done manually but is not recommended. Allow Access to save the password (The final Password and credentials are handled under Deployment Security). Click NEXT

11 On completion the Wizard will allow you to preview the Upsizing Parameters. Check to see that the Wizard completed the migration as planned. Print the report.

12 Delete LOCAL tables. The Upsized Access Application has all the original tables still contained in the MDB. The New Tables are linked to SQL Server and the link Icon is a small green globe.

13 Repair and Compact the MS Access Upsized MDB

On successful completion the MS Access application is ready for testing. See Access Deployment Guidelines in the Addendum when testing is complete

Oracle Migration Procedures

Overview

The recommended tool for migrating MS Access data to the Oracle Server is the Oracle Migration Workbench (OMW). To accomplish this you will need Oracle DBA credentials or have table create permissions. You will also need a current copy of the OWM and the MS Access Plug in from Oracle's Website.

OMW converts any ODBC-compliant data source to Oracle by parsing the original Data source into a XML file, allowing the user to modify some of the parameters in the OMW's design tools, and importing the data. To accomplish this in MS Access, Oracle provides the MS Access Migration Plug In.

The MS Access Plug In extracts the MS Access data and structures, and stores them into a directory in XML format. OMW recognizes these files and imports the data and structures into a staging database, where you can edit or map the MS Access data to other Oracle data types or columns. OMW creates all the tables and optionally imports all the data based upon the structures in the staging database. When the Plug In completes, OMW creates an Oracle Repository to hold the new Data and Structures. OMW smoothly handles tables and keys and can create Oracle Views for each Query.

OMW links the tables back to MS Access by renaming the local tables TableName_L and the remote tables TableName_R, and then creates an MS Access Query with the same name as the original table, and aliases the Column names back to the original names used in MS Access. As the only object in MS Access with the original table's name, the new query becomes the base for all report, forms, and other queries. OMW converts all queries to Oracle and makes a "best guess" to replace the MS Access-specific functions with the PL/SQL equivalent.



Views sometimes fail to convert to a usable format. OMW does not notify the user that the resultant views do not function.



The reader should also be warned that OMW quietly removes the Referential Integrity, the column captions, all default values and pushes the Column name to all capital letters.

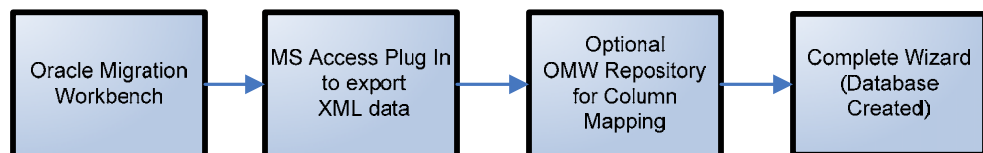


Figure 13: Oracle Migration Process

**Oracle Migration
Workbench
Preparation**

Before Migration, make sure to follow the Front-End Preparation Guidelines and Data Preparation Guidelines presented in this document. Additionally the operator must be a SQL Server Administrator or the Database Owner for the Target Oracle Database.

Follow these 5 steps before running Oracle Migration Workbench:

- 1** Create a New Access MDB called <ApplicationName>_FE.mdb (“FE” for Front End) and import all the objects from the MS Access application. If the application uses an external Access Data MDB then Import all of its objects too.
- 2** Repair and Compaq the new MS Access file you just created
- 3** Unhide all Tables except System Tables (Msys).
- 4** Insure that all tables have a Primary Key, or they will be interpreted as read-only
- 5** Per the ReadMe file, copy the MSAccess.Jar file to the OMW Plug Ins directory

**Oracle Workbench
Migration
Procedures**

Once you have completed the preparation steps above, follow the procedures below to run the Oracle Migration Workbench process:

- 1** Run the appropriate MS Access Exporter to Capture the Application. It should be in the Oracle\omwb\MSAccess_Exporter directory by default. The MS Access Exporter is MS Access version-dependent, and can take a few minutes to get started.
- 2** On the Plug In screen notice the Source and Target locations. Export BOTH the Schema and the Table data in two separate steps from this same screen. Click each Export then Exit.
- 3** The first time through the migration process, another Wizard will appear. This Wizard sets up the Repository Schema using a 4-step process to define the table spaces. In Step 1, define where to locate the XML Target Data.
- 4** Step 2: edit the Column Type Conversions, if needed. Click Next
- 5** Step 3: create the Oracle Model for Data Import. Click Next
- 6** Step 4: specify the Oracle Destination Database Schema. You should see a banner announcing successful completion. The XML Structures are complete. Click Next
- 7** OMW now creates the Oracle Repository. Click “Close” and resume the Oracle Migration Workbench Wizard
- 8** The Wizard starts with a screen to set the security context of the new Database. Enter the appropriate data and click Next
- 9** Select the XML file from the repository. Make sure to select the same one you just created. Click “OK” and OMW will automatically start the process of migration.
- 10** OMW creates the Schema form and the Model in the Oracle Repository. Click “Next” when complete.
- 11** Optionally you can import the data now. An important part of this step is that OMW also allows importing the MS Access data at a later time. The option to import the data at a time later is very handy for importing production data right before the final testing begins.
- 12** When done choose All options to Migrate the actual data.

- 13** Review the Oracle Migration Summary on the final Screen and click “Next”
- 14** All Identified errors are reported on a short list data screen. Click “Done”
- 15** OMW returns to the Main screen on Completion

The default behavior for OMW is to leave the Local data connected as “_L” but map the Remote data to Oracle using the “_R” suffix. OMW creates a Query in MS Access with the original name of the MS Access Table, and Aliases all the fields back to their Pascal Cased names. This is the only object in MS Access with the original Table name. All other MS Access reports, Queries and Forms will use the Query instead of the Table.

You can now view the data in MS Access. Linked Tables appear with the green globe Icon, which indicates an ODBC-connected Table.

- 16** Thoroughly test the migrated MS Access Application against the new back-end database.

Deploying a Migrated MS Access Application

As a multi-user class business application with a database Server back-end data source, the migrated MS Access Application needs to be deployed and made available to the target End Users.

The performance of a migrated and deployed MS Access application can be dramatically improved by moving all the data manipulation methods to the back-end data source. This is accomplished using Views and Stored Procedures.

A recommended deployment technique is to compile the MS Access front-end Application into its MDE format, and deploy the MDE file to a secure network share that limits access to the MDE file only to authorized End Users.

Note that each End User will require the appropriate version of any third-party tools, controls and ODBC drivers, as well as the current version of ADO if used.

MDE

Convert all MS Access Front End applications to the MDE format prior to deployment. Tools → Database Utilities → Make MDE. The MDE Wizard removes all ability to view or edit VBA code, and compiles a final pseudo-code (p-code) version of the Application for shared deployment. The p-code is not binary, and is therefore still interpreted by MS Access. Therefore, you should not expect any speed improvement at the code level. Its primary purpose is to irreversibly hide the source code from the End Users to prevent unmanaged changes to the application.



DO NOT DELETE YOUR ORIGINAL MDB FILE, AS ONLY THE MDB FILE CONTAINS THE ORIGINAL SOURCE CODE!

Migrated and compiled MS Access MDE applications have no data and require a much smaller footprint. Consequently, they take up less disk space, are easier to manage, and are less prone to corruption. Typically, there is an improvement in data access speed due to the inherent speed over the network of client-server data access versus file server data access. Putting the increase in speed of the data access aside, there is generally no improvement in loading or running a migrated or compiled MS Access application, nor any increased speed in opening a form or loading a report. The central network deployment method is not only much more secure, it also eases the developer's maintenance burden considerably, as all future updates to the application require no further deployment procedures, and are seamless to the End Users.

Security

MS Access security is only sufficient to prevent well-intentioned End Users from inadvertently harming the data. As such, the recommended technique is to deploy only the MDE version of the migrated MS Access Application to a network shared resource which is dedicated explicitly to target End User's work group, and linked to the desktop and/or menu with a shortcut. This deployment method insures that all application security responsibilities are under the aegis of the centralized network share security model, which is much more rigorous than MS Access. Care must be taken to ensure the deployed MS Access Application does not incidentally expose the End User passwords. Note that any End User that is authorized to edit the Database server's tables that are linked to MS Access also has direct access to the SQL/Oracle Tables.

Other Database Connectivity

MS Access uses the Other Database Connectivity (ODBC) interface through the Microsoft System Data Source Name (DSN). DSN is configured by a system Administrator via Start → Administrator Tools → Data Sources.

ODBC is a Microsoft-published open specification that allows disparate data management systems to talk to and share data with each other. This thin layer contains all the connection information, including End User names and passwords. ODBC is installed on the Client Computer, and is a deployment concern for all migrated MS Access applications.

Not all ODBC drivers are alike. Oracle and Microsoft provide the drivers for free and Microsoft provides an Oracle driver at no cost. Sometime third-party drivers can be easier to maintain and deploy.

MS Access uses ODBC to make the default table connections for linked Tables and Views. Working with ODBC, MS Access will cache the user credentials according the installation preferences. This option is specified when the MS Access application Tables are linked back to the Database Server.

VBA makes it possible to bypass ODBC and use the newer more robust ADO (Microsoft Data Access Components or MDAC). MS Access applications that use ADO also need the current approved ADO standard product and version installed as a prerequisite to deployment.

ADO under VBA, and DAO under VBA typically embed (hard code) the connection strings throughout the VBA Code, effectively hiding it from the User, but making it difficult to change security permissions following deployment. Connection strings that depend on the Windows Security are centrally maintained and automatically updated with the users Windows profile.

There is some argument that Windows Integrated Security is less scaleable as the Data Server cannot optimize connection pooling. The point is well taken in a Web Server environment. Highly scaleable applications are not good candidates for MS Access.

**Performance
Enhancement
Techniques**

MS Access does not natively optimize for Client/Server performance. Below are several time-tested techniques that will — individually and collectively — substantially improve the performance of a migrated and deployed MS Access application.

Suboptimal Access Migration Techniques

Queries that employ heterogeneous joins (joins from more than one type of database), queries that contain functions, overly complex joins, use of the UNION clause and WHERE clauses that refer to objects in MS Access should all be avoided or rewritten.

Data Caching

Fortunately MS Access caches data locally and consequently does not normally need to complete a round-trip data transfer with the source database for Sorts and Filters contained in Views.

Stored Procedures

MS Access benefits substantially from the appropriate use of Stored Procedures. For example, a Combo Box that returns all records in MS Access is an ideal candidate to migrate to a Stored Procedure using a Parameter as the data source.

Forms

Opening a Form will, by default, fetch *all* records, and substantially slow the application response time and clog the network with unnecessary data traffic. For performance reasons, any Form that directly accesses a large data Table, or a Query that returns all rows on a large data Table, needs to have the data source restricted. This performance issue is further aggravated when a Form requires multiple Table joins.

Combo-boxes that are driven from remote Tables are a very common cause of poor application performance. If the data is static i.e. State Name abbreviations, consider using local Access Tables to populate the Combo-boxes. If performance is still an issue it will be necessary to re-design the form to restrict the range of data it requests.

SQL Pass Through

Setting a Query to SQL Pass Through will dramatically improve performance. Remember, if the SQL is completely back-end compatible, then it should stand as a SQL View for better security and performance.

Converting Select Queries to Views and Stored Procedures

Converting MS Access Queries to Client/Server “Views” will result in performance gains and enhanced security. This technique requires that you create a View on the server and link it back to the MS Access application. Any Query that uses an MS Access Internal or User-Defined Function will need to be re-written as a Server-based function if the Query is moved to the server as a View.

A well-designed View will support reading and writing of data, and the database will optimize the performance. This technique has the additional advantage of inheriting the back-end security model. Note that Views cannot be updated unless all Primary Keys in each table are included in the “Select” statement. Once the View is created on the Server, link it back to MS Access using the Linked Table Manager Wizard, temporarily rename the Old Query to XX_QueryName, and name the new linked View to match the original MS Access Query Name.

Addendum

Other Migration Tools

SSW Upsizing Pro

While only designed for Upsizing to SQL Server, SSW's Access Upsizing Pro is helpful for Oracle Migration too.



<http://www.ssw.com.au/ssw/UpsizingPRO/Default.aspx>

Unlike either Microsoft's Wizard or Oracle's Workbench Wizard, the Upsizing Pro identifies the remedial actions that need to be addressed in the source MS Access application prior to the new database server migration.

SSW Upsizing Pro is based upon a rules-based engine, and includes analysis of performance issues. The Upsizing Pro carefully documents exactly what should be corrected while still in MS Access — where these types of changes are much more easily implemented.

Data Direct ODBC Driver

MS Access uses Microsoft's ODBC drivers to link the data from MS Access to other data sources. Unfortunately, the current drivers shipping with MS Access do not support Windows Authentication (Oracle's External User) for Oracle Database Servers.

There is a wide variety of ODBC data connection tools available, we recommend the Data Direct driver:



<http://www.datadirect.com/products/odbc/index.ssp>

MS Access Developer Edition

MS Access Developer Edition (MSDE) allows deployment of an MS Access application to computers that do not have MS Access pre-installed. MSDE is a recommended deployment practice, as it eliminates one more opportunity for User modifications to a tested application.

ADDENDUM

MS Access Data Type Conversion Tables

MS Access		SQL Server
Text	→	Nvarchar
Number	→	Int or Decimal
Currency	→	Money
AutoNumber	→	int (identity)
Boolean	→	Bit
OLE Object	→	Image
Hyperlink	→	Text
Memo	→	Text
DateTime	→	Datetime
Field Size	→	Fieldsize
Description	→	Description

MS Access		Oracle
Boolean	→	NUMBER(1, 0)
Byte	→	NUMBER(3, 0)
Currency	→	NUMBER(15, 4)
Date	→	DATE
Double	→	FLOAT(126)
Integer	→	NUMBER(5, 0)
Long	→	NUMBER(11, 0)
LongBinary	→	BLOB
Memo	→	CLOB
Single	→	FLOAT(126)
Text	→	VARCHAR2

A D D E N D U M

**MS Access Data
Column
Limitations**

Data Type	Description	Minimum Value	Maximum Value
Text	Stores variable length text	1	255
Memo	Large variable length text	1	64,000 bytes
Number Byte	1 byte storage	0	255
Number Integer	2 bytes storage	-32,768	32,767
Number Long Integer	4 bytes storage	-2,147,483,648	2,147,483,647
Number Single	4 bytes storage	-3.4 x 10 ³⁸	3.4 x 10 ³⁸
Number Double	8 bytes storage	-1.8 x 10 ³⁰⁸	1.8 x 10 ³⁰⁸
Currency	8 bytes storage - monetary values	-922337203685477.5808	922337203685477.5808
Counter Yes/No	4 bytes - AutoIncrement Field 1 bit storage - Boolean Value	0	2,147,483,647
Date/Time	8 bytes storage		
OLE Object	OLE, graphics other complex data	1	1.2 gigabytes

A D D E N D U M

**MS Access to
Oracle Functions
Conversion**

Microsoft Access	Oracle	Conversion Action
Abs	Abs	OK
Asc	Ascii	Handled automatically by OMW
Atn	-	DO NOT USE
CCur	CCUR	Handled automatically by OMW
Cdbl	CDBL	Handled automatically by OMW
Chr	Chr	OK
Chr\$	Chr	OK
CInt	CINT	Handled automatically by OMW
CLng	CLNG	Handled automatically by OMW
Command	-	DO NOT USE
Command\$	-	DO NOT USE
Cos	COS	OK
CSng	To_Number	Handled automatically by OMW
CStr	To_Char	Handled automatically by OMW
CVar	To_Char	Handled automatically by OMW
CVDate	-	DO NOT USE
Date	SYSDATE	Handled automatically by OMW
Date\$	SYSDATE	Handled automatically by OMW
DateAdd	DATEADD	Handled automatically by OMW
DateDiff	-	DO NOT USE
DatePart	-	DO NOT USE
DateSerial	-	DO NOT USE
DateValue	-	DO NOT USE
To_Date	-	DO NOT USE
Day	-	DO NOT USE
Environ	-	DO NOT USE

ADDENDUM

Microsoft Access	Oracle	Conversion Action
Environ\$	-	DO NOT USE
Exp	EXP	OK
Fix	Trunc	Handled automatically by OMW
Format	-	DO NOT USE
Format\$	-	DO NOT USE
Hex	-	DO NOT USE
Hex\$	-	DO NOT USE
Hour	-	DO NOT USE
In	-	CODE (not supported in def)
InStr	InStr	OK
Int	INTN	Handled automatically by OMW
Is Not Null		CODE (not supported in def)
Is Null		CODE (not supported in def)
IsDate	-	DO NOT USE
LCase	LOWER	Handled automatically by OMW
LCase\$	LOWER	Handled automatically by OMW
Left	LEFT	Handled automatically by OMW
Left\$	SUBSTR	Handled automatically by OMW
Len	LENGTH	Handled automatically by OMW
Like	*	
Log	LOG	OK
LTrim	LTRIM	OK
LTrim\$	LTRIM	OK
Mid	SUBSTR	Handled automatically by OMW
Mid\$	MID	Handled automatically by OMW
Minute	-	DO NOT USE

ADDENDUM

Microsoft Access	Oracle	Conversion Action
Month	-	DO NOT USE
Now	SYSDATE	Handled automatically by OMW
Oct	-	DO NOT USE
Oct\$	-	DO NOT USE
RGB	-	DO NOT USE
Right	RIGHT	Handled automatically by OMW
Right\$	SUBSTR	Handled automatically by OMW
Rnd	RND	Handled automatically by OMW
RTrim	RTRIM	OK
RTrim\$	RTRIM	OK
Second	-	DO NOT USE
Sgn	SIGN	Handled automatically by OMW
Sin	SIN	OK
Space	SPACE	Handled automatically by OMW
Space\$	LPAD/RPAD	Handled automatically by OMW
Sqr	SQRT	Handled automatically by OMW
Str	TO_Char	Handled automatically by OMW
Str\$	TO_Char	Handled automatically by OMW
StrComp	-	DO NOT USE
String	LPAD/RPAD	Handled automatically by OMW
String\$	LPAD/RPAD	Handled automatically by OMW
Tan	TAN	OK
Time	TIME_1	Handled automatically by OMW
Time	TIME_2	Handled automatically by OMW
Time\$	SYSDATE	Handled automatically by OMW
Timer	-	DO NOT USE

ADDENDUM

Microsoft Access	Oracle	Conversion Action
TimeSerial	-	DO NOT USE
TimeValue	-	DO NOT USE
Trim	TRIM	Handled automatically by OMW
Trim\$	TRIM	OK
UCase	UPPER	Handled automatically by OMW
UCase\$	UPPER	Handled automatically by OMW
Val	TO_NUMBER	Handled automatically by OMW
Weekday	-	DO NOT USE
Year	-	DO NOT USE
^	Power(m,n)	Handled automatically by OMW
<	LTN	Handled automatically by OMW
>	GTN	Handled automatically by OMW
<=	LTEN	Handled automatically by OMW
>=	GTEN	Handled automatically by OMW
<>	NOTEQN	Handled automatically by OMW
=	EQN	Handled automatically by OMW
and	ANDN	Handled automatically by OMW
or	ORN	Handled automatically by OMW
not	NOTN	Handled automatically by OMW
divide operator (/)	INTDIV	Handled automatically by OMW
Eqv	-	DO NOT USE
Imp	-	DO NOT USE
Mod	MOD(m,n)	Handled automatically by OMW
Xor	-	DO NOT USE

A D D E N D U M

List of Figures

FIGURE 1: STANDARD MS ACCESS APPLICATION CONFIGURATION 6

FIGURE 2: MS ACCESS MIGRATION CONFIGURATIONS 7

FIGURE 3: MS APPLICATION MIGRATION TABLE 8

FIGURE 4: SPLIT MS ACCESS APPLICATION 9

FIGURE 5: MS ACCESS DATA STORE..... 10

FIGURE 6: MS ACCESS DATA PROJECT 11

FIGURE 7: MS ACCESS CLIENT/SERVER APPLICATION 12

FIGURE 8: CLIENT/SERVER APPLICATION..... 13

FIGURE 9: WEB BROWSER APPLICATION..... 14

FIGURE 10: SMART CLIENT APPLICATION 15

FIGURE 11: MS ACCESS MIGRATION FLOW DIAGRAM..... 17

FIGURE 12: MIGRATION PREPARATION PROCEDURES..... 22

FIGURE 13: ORACLE MIGRATION PROCESS 30