

Microsoft



# Improving the Reach and Manageability of Microsoft® Access 2010 Database Applications with Microsoft® Access Services

January, 2010

**Microsoft**

# Contents

Introduction .....	1
Empowering End Users with Access .....	1
Benefits .....	1
Manageability Challenges .....	2
Meeting Manageability Challenges by Centralizing Storage .....	3
Managing Split Access Applications .....	3
Moving Data to SQL Server .....	3
Using Terminal Services to Deploy Access Applications.....	4
Increasing Manageability with SharePoint .....	4
Publishing Access 2010 Databases to Access Services .....	6
Web Access, Better Performance and Enhanced Manageability .....	6
Web Databases, Web Objects, and Client Objects.....	7
Deploying Access databases in SharePoint .....	9
Storing databases into SharePoint Document Libraries.....	9
Publishing an Access Services application .....	9
Publishing Client only Applications .....	10
Hosted SharePoint Options .....	10
Migrating Legacy Data to Web Tables .....	11
Using the Web Compatibility Checker.....	11
Handling Compatibility Issues .....	11
Creating New Compatible Tables and Importing Legacy Data .....	13
Synchronizing data between web tables and external sources.....	13
Migrating Legacy Application Objects .....	15
Handling Publishing, Compilation, and Runtime Errors.....	15
Publishing Issues .....	15
Compilation Issues .....	16
Runtime Issues .....	16
Upgrading Databases to Access 2010 .....	17
64-bit VBA Issues.....	17

Summary .....	17
Appendix .....	18
Access 2010 Features by Object Type .....	18
Tables.....	18
Forms .....	19
Reports.....	21
Queries.....	22
Macros.....	23
Expressions .....	26

## Introduction

Microsoft Access empowers end users in business units to develop applications quickly and at low cost. This agility is valued by large organizations around the world. However, some of these organizations struggle with managing the myriad Access databases in use. With Microsoft Access 2010 and Microsoft SharePoint 2010 Products working together, the best of both worlds is possible: you can satisfy the need for agile development from your business units and still rest assured that the data is secured, backed up and managed appropriately.

Access Services is a new feature of Microsoft® SharePoint Server 2010 Enterprise Edition that supports close integration with Microsoft® Access 2010. This integration enables users to extend the ease of Access application development to the creation of Web forms and reports, and it enables IT managers to extend the ease of SharePoint -2010 Products administration to the management of Access data, application objects, and application behavior. This paper explains the benefits and architecture of this new level of integration, and it provides technical details that will be helpful in implementing successful migration of existing Access applications to this new architecture.

## Empowering End Users with Access

Access has long been one of the most popular desktop tools for building database applications, because it empowers end users to accomplish tasks that would otherwise require the services of IT professionals.

The easy-to-use graphical interfaces and wizards in Access enable users to create related tables for data storage, as well as links to external data, and to build forms and reports for entering, manipulating, and analyzing data. Unlike Microsoft Excel, Access is built on a relational engine and is therefore inherently optimized for data validation, referential integrity, and quick, versatile queries.

### Benefits

The ever-evolving data management needs of an organization cannot all be met by trained programmers and database administrators, whose time is costly and limited. End users frequently can't wait for these resources to become available or can't justify the expense. So they turn to alternatives, from manual notes to spreadsheets, often limiting their productivity and effectiveness. When users discover Access and invest just a short time learning to use it, they are usually delighted to see how much they can accomplish.

Access empowers users to gather information from many disparate sources, including spreadsheets, HTML tables, SharePoint lists, structured text files, XML, Web services, or any ODBC data source such as Microsoft SQL Server, Oracle, or MySQL. Depending on the source and the user's permissions, this data can even be updated from Access, and can be combined in distributed queries.

As project requirements and user skills evolve, Access applications can become increasingly complex and full-featured. Users go online and easily find a rich ecosystem of help and

other resources available, including tips and samples that have accumulated since Access was first released in 1992.

By satisfying user needs without burdening IT resources, Access often plays a valued role in meeting an organization's data management needs.

### **Manageability Challenges**

The vast majority of Access applications never come to the attention of the IT department. However, a percentage of Access databases do create problems that draw the attention of IT managers. Access data and application files may be lost or corrupted. Long-running queries can burden network or server resources. Sensitive data can inadvertently be left unprotected. Performance can degrade as more data and users are added. And departments can come to depend on applications developed by people who are no longer available or whose skills are inadequate to meet current requirements.

IT managers sometimes take the position that Access should be prohibited to avoid these issues. However, users either defy the ban or revert to alternatives such as spreadsheets that are no more manageable or secure. The needs that drive Access adoption don't go away and can't all be met by IT-mediated application development.

Most IT departments eventually conclude that the best approach is to provide guidance and management that encourages users to leverage the capabilities of Access safely and productively. This management and guidance can take several forms, including distribution of templates and samples that encourage best practices.

In addition, centralizing storage of Access data and application files can help improve reliability, security, and manageability, without unduly inconveniencing users. Centralizing data storage can also enable Access applications to scale to serve many users. The remainder of this article discusses several options for centralizing storage of Access data and applications, with an emphasis on the new capabilities provided by integrating Access 2010 and SharePoint Server 2010.

## Meeting Manageability Challenges by Centralizing Storage

As summarized below, a range of options have emerged over the years for centralizing the storage of Access data and application objects, including storage of files on managed network shares or on terminal servers, migration of data to database servers such as SQL Server, migration of data to SharePoint lists, and storage of applications in SharePoint document libraries. Access Services introduces a new set of options that not only extend Access application to the Web, but also significantly enhance manageability.

### Managing Split Access Applications

Access is distinctive in its ability to combine a query processor and storage engine with an application development environment and an application runtime framework. A single Access database file can contain both data tables and application objects stored in system tables. However, Access users commonly split the storage of application objects, such as forms and reports, from the storage of user data.

Especially in applications that are shared by multiple users, the most manageable pattern is to place only the user data tables in one Access database, usually called the back end, on a file share. All the other objects, including forms, reports, VBA code modules, Access macros, and saved queries, are stored in a separate database file, which also contains links to the data tables. Multiple copies of this front-end file are distributed to users for storage on their local drives. This allows front-end objects to be updated without disturbing the data, and local storage of these objects provides better performance than sharing a single remote copy of the front end.

Microsoft has encouraged this pattern of deployment by providing a wizard that automatically creates a second database file, exports all local tables to the new database, and creates links to the exported tables. Any application object that had worked with the local tables automatically then work with the new links to exported tables.

However, the manageability of these database files is limited. Users must have full permissions to create and delete files on the data share, and database files can proliferate uncontrollably. It is not uncommon for enterprises to discover that tens of thousands of Access database files are scattered across their networks.

In addition, collaborative design work is difficult for users to manage. When multiple users have their own copies of a front-end application file, it is difficult for them to receive a new version without losing any customizations they may have made.

### Moving Data to SQL Server

Once data tables have been separated from application objects, it requires little work to migrate those tables to a SQL Server database and change the links to ones that use ODBC to access the data. Access includes an "upsizing" wizard for exporting tables to SQL Server and linking to them, and SQL Server also provides a tool call the SQL Server Migration Assistant for Access

(<http://www.microsoft.com/downloads/details.aspx?FamilyID=d842f8b4-c914-4ac7-b2f3-d25fff4e24fb&DisplayLang=en>).

This improves reliability, scalability, and security. However, users require privileges and training to create, maintain, and administer SQL Server databases. Some IT organizations restrict the number of users who have such privileges.

### **Using Terminal Services to Deploy Access Applications**

Another option is to centralize Access applications on terminal servers. This provides the significant benefit of allowing users to access their applications across a wide area network or the Internet, while maintaining good performance. IT managers have better control over backup, reliability, and security for those applications.

Users can't get to their applications from any browser on any device. Using Terminal Services works well for intranet deployments of canned Access applications, but is less useful for supporting ad hoc user-generated solutions.

## **Increasing Manageability with SharePoint 2010 Products**

SharePoint 2010 Products are architected to support safe and scalable creation of thousands of sites and lists by minimally privileged and minimally trained users. In addition, SharePoint Server is highly manageable. It has a security model that is tightly integrated with Active Directory. Data backups are assured, and a multi-level recycle bin provides easy recovery of deleted data items. A highly scalable architecture supports handling increased load by adding servers. Plus, SharePoint 2010 Products are engineered to provide highly configurable activity throttles to protect server and network resources, while still supporting end-user creation of new applications and new content.

As a Web-based platform that employs standard Internet protocols, SharePoint 2010 Products enable users to access their applications from any browser on any device. Users are often delighted to learn how easy it is for them to collaborate over the Web with SharePoint 2010 Products. This ease of use, combined with IT-friendly manageability has made it the fastest growing server product in the history of Microsoft.

For all these reasons, the case for integrating Access and SharePoint 2010 Products is strong, and with each of the last three versions of the products that integration has deepened.

### **A Brief History of Access/SharePoint Products and Technologies Integration**

Access 2003 introduced integration with Microsoft SharePoint Portal Server 2003 and Windows SharePoint Services by adding an Installable ISAM driver that enabled the Jet database engine, the engine used by Access 2003, to connect to SharePoint lists. This allowed Access users to view and edit SharePoint data and to create queries that join SharePoint data to data from other sources.

Access 2007 added significant new support for Microsoft Office SharePoint Server 2007 and Windows SharePoint Services by enabling users to take list data offline and then synchronize with the server when they reconnect. To accomplish this, the Access team branched off a proprietary version of the Jet database engine, renamed ACE. They added new data engine features to provide parity with SharePoint data types, including support for file attachments and complex multi-valued columns. New Access UI made it easier to move data to SharePoint lists, and SharePoint Products and Technologies also added UI features to support working with Access applications stored in document libraries.

With Access 2007, users had a more seamless experience of working with SharePoint lists, but those lists still lacked full suitability for use in most Access applications. Performance was often slow and important database features, such as referential integrity and enforcement of validation rules, couldn't be implemented without resorting to complex Workflow programming on the computer that is running Office SharePoint Server or Windows SharePoint Services.

Access 2010 and SharePoint 2010 Products address these shortcomings. Performance issues have been eliminated through server-side and client-side data caching, as explained below. Referential integrity and the basic expected forms of data validation are now enabled on the server without requiring any custom programming. For more advanced validation, users can easily use Access macros to create server-based Workflow customizations.

In addition, Access 2010 offers exciting new ways of integrating with SharePoint 2010 Products that allow users to run Access applications using only a Web browser. These new capabilities, based on Access Services running on SharePoint Server 2010, require Enterprise CAL licensing, but economical hosted solutions are or will soon be made available from Microsoft and third parties for organizations that don't have in-house installations.

## Publishing Access 2010 Databases to Access Services

Access 2010 introduces the ability to publish a database to Access Services, which creates a SharePoint site for the application. Any local tables are moved to SharePoint lists. If any of the data can't be moved into a list, then publishing cannot happen until the data is exported to a separate database or changed to be compatible. A compatibility checker supports this by listing any problems that would prevent publishing.

After a database is published, it becomes a Web database, meaning that users can add Web forms, reports, queries and macros that can execute on the computer that is running SharePoint Server when the application runs in a browser or on the client when the application runs in Access. Users can browse to Web forms and reports over the Internet, or they can run them in the Access client.

Published databases can also contain objects, with a fuller feature set, that only run in the Access client. Tables linked to external data, such as data in other Access databases, Excel spreadsheets, SQL Server tables, or even in SharePoint lists on other sites, are only available in the Access client, not to Web forms and reports. All design work occurs in the Access client.

Publishing Access databases to Access Services on SharePoint Server, rather than simply saving them in SharePoint document libraries, provides three key advantages:

- Published applications can contain forms and reports that are enabled to run in a browser as well as in the Access client
- Published applications are stored and synchronized with greater granularity and efficiency than applications in document libraries
- Published applications are more manageable than applications stored in document libraries

### **Web Access, Better Performance and Enhanced Manageability**

Making forms and reports available over the Web provides a key advantage. In today's distributed workforce, being able to collaborate with colleagues all around the world is critical. Increasingly, users are looking for a no-install solution for collaboration that can work with varied bandwidth and varied proximity to data. Web applications also enable users to work with Access applications without being distracted by tools for customizing the application, a benefit that has in the past often required professional programming services.

In published databases, individual Access objects are serialized and saved in a hidden SharePoint list. This is similar to the way that programming objects are saved in source control systems. When users choose to open published applications in the Access client, rather than in a browser, the local version of the database is synchronized with the version on the server. Synchronization affects all the objects in the database, not just the data. Because objects are stored as individual data items, SharePoint Server maintains the user identity and date for each modification, as it does for all data changes.

As in source control, the Access client downloads the entire database only when a user doesn't already have a local copy. Subsequently, Access fetches only objects and data items

that have changed. This arrangement is much more efficient than working with applications saved monolithically in document libraries and provides noticeably faster performance. Different users can make changes to different objects or different data items in a list without causing conflicts. When data conflicts do occur, a conflict resolution wizard enables the user to choose which version of data items to preserve. For object conflicts, Access provides the name of the user who made the saved change and creates a renamed copy of the local object before downloading the other user's changes. This fosters collaboration on resolving any design conflicts and ensures that no design work is lost.

Publishing also enables greater administrative control. Permissions can limit some users from being able to modify, delete, or create objects in a site, while still allowing them to run the application. In addition, because application objects are stored individually as list items rather than monolithically in document libraries, the throttles in SharePoint Server for limiting list traffic and user activity can apply to individual types of application objects.

Because of the many advantages of using published databases rather than document libraries to centralize application storage on a computer that is running SharePoint Server, document libraries should only be considered for storing legacy Access databases that cannot be upgraded to Access 2010, or when a server supporting Access Services isn't available.

### **Web Databases, Web Objects, and Client Objects**

To create an Access 2010 Web database, you can choose Blank Web Database when creating a new one or you can publish an Access 2010 database that wasn't originally created as a Web database. If it publishes successfully, it automatically becomes a Web database.

In a Web database, you can create two types of objects: Web objects that can run either in a browser or in the Access client, and non-Web objects that can only run in the Access client. All design changes for all types of objects must be made in the Access client.

When you are working on a Web database in the Access client, the Create ribbon refers to non-Web objects as Client objects, specifically Client Forms, Client Reports, Client Queries, and Client Macros. That terminology might be confusing, because these so-called "client" objects actually do get published to the server along with your Web objects. Any design changes you make to them are propagated to the server when you synchronize, and you also receive any synchronized changes made to them by other users, just like with Web objects. What makes non-Web objects special is that they can only run in the Access client, not in a browser. They use the Web for synchronizing design changes but not for execution. All linked tables are client tables, and only client objects can see them, but the definitions of the links get synchronized with the server like other client objects. Synchronization provides a useful way to collaborate and deploy applications, even if all the objects in the database are client objects and all the tables are client linked tables.

To create Web objects, you must be working in a Web database, where you can create Web objects even before you've published. To add Web objects to a non-Web database, you must first publish it. However, you can create a Web database from scratch and add Web objects even before you've published. When you create local tables in a Web database that you haven't yet published, the table schema is guaranteed to be compatible with SharePoint

lists. So it is very useful to create a Web database when you start a new application, even if you have no immediate plans to publish it.

The only data available to Web objects is the data contained in the application's Web tables. Only client objects can work with linked tables. In the Access client, when connected to the computer that is running SharePoint Server, data and design changes to Web tables automatically synchronize with the server, and Access works against local copies of the tables. When disconnected, Access seamlessly continues to work against the local copy, and doesn't allow design changes to the tables. When reconnected, Access notifies users that they can now synchronize and resolve any conflicts. Design changes to objects other than Web tables synchronize only when users explicitly request synchronization by clicking the Sync All button in the Info section of Backstage view. Backstage view is the name for the view displayed in the File menu.

Web objects support the same feature set whether running in a browser or in the Access client, but some features of client objects are not supported by the corresponding Web object types. For example, VBA code only runs in client forms, not Web forms. Web forms rely on macros for programmability, but this is less of a restriction than you might expect, because Access 2010 macro capabilities are significantly enhanced. Separate sections below, under the heading Access 2010 Features by Object Type, provide more details on how the different types of Web objects differ from their client counterparts.

# Deploying Access databases in SharePoint Technologies

The following sections provide an overview of several different topologies that are supported for integrating Access and SharePoint Technologies.

## Storing databases into SharePoint Document Libraries

Access 2007 supports the use of SharePoint document libraries for centralizing storage and deployment of Access applications. This includes support for Access forms and reports in databases stored in document libraries. When users open these, the databases automatically open in the Access client. Users can move entire applications to SharePoint libraries. The applications always run in the Access client, and they are downloaded only when a user first opens the application or when the server version is updated. One restriction is that the design objects in these applications are read-only. To make design changes, a user must work on another local copy and upload the new version, replacing the old one on the server. These applications can work with local Access tables, data in SharePoint lists via linked tables, or any other supported external data source.

Moving forward, document libraries should only be used as legacy support for users who have not upgraded to Access 2010 or for SharePoint Technology installations that do not support Access Services. When Access Services are available, they provide many advantages over using traditional Access applications stored in documents libraries. Access 2010 applications published to SharePoint Server using Access Services support design changes and allow multiple users to collaborate on design. Design changes are tracked per object rather than per project, resulting in fewer conflicts. In addition, Access Services supports the addition of Web objects that users can access through a browser, without depending exclusively on the Access client.

## Publishing an Access Services application

Access 2010 combined with Access Services introduces the ability to publish a database to SharePoint Server. A site is created for the database and tables are stored as SharePoint lists. Web forms will be available from within a browser and the site and data will be backed-up and permission levels can be maintained by SharePoint Server. The publish process moved the data from the database to SharePoint Server and converts the tables to linked tables. Users will have the option to use the Web objects in the browser or open the database from the website in their client, enabling access to the client objects that are also stored inside the site.

Users also can link to SharePoint lists from databases that are not Web databases and that will never be published to SharePoint Server. For example, a user might create a simple Web database to collect data from other users over the Web. In a separate application, the user can link to that data and create reports that combine the data with other data sources.

Linked SharePoint lists in Access 2010 also have the same support as Web tables for offline work. Disconnected users can view or modify the data offline and then synchronize with the server when they reconnect. In addition, users can work through the standard Web interfaces in SharePoint Server to work with list data, even if the data isn't in a published Access Web database.

### **Publishing Client-only Applications**

Even with Access 2010 applications rely exclusively on linked data from external Access databases, spreadsheets, database servers, web services, or from linked SharePoint lists, there are advantages to publishing the applications. For the users, published applications support convenient deployment, versioning and collaboration. For IT managers, published applications benefit from the backup, security, and manageability features in SharePoint Server.

By adding Web tables to these applications, users have the ability to extend their applications to include some forms and reports that run on the Web.

### **Hosted SharePoint Server Options**

For organizations or users that do not have Enterprise CAL SharePoint Server licenses or do not want to maintain their own installation of SharePoint Server, Microsoft and third parties provide hosted options with economical monthly per-user rates. These options include multi-tenant hosting where data from multiple organizations is segregated on one server, or dedicated options that provide the added assurance of complete segregation on dedicated servers maintained by the service provider.

## Migrating Legacy Data to Web Tables

Web objects in Access 2010 can only work with data in an application's Web tables, which are implemented on the server as SharePoint lists. To create Web forms and reports that work with legacy data, users must import their data into local Access tables and publish the database, or import the data into existing Web tables in the database. Publishing succeeds only when the table schema and the data itself in local tables are compatible with SharePoint lists.

Some or all of the legacy data in an application can remain in external data sources that appear in Access as linked tables, but this data isn't available to Web objects. Linked table data is only available to client forms, reports, queries, and macros running in the Access client.

### Using the Web Compatibility Checker

The Web Compatibility Checker inspects table designs and logs each incompatibility that it finds in a table named Web Compatibility Issues. You can run the Web Compatibility Checker by right-clicking on a table and selecting Check Web Compatibility, or click the Run Compatibility Checker button that appears when you choose to Publish to Access Services in the Save and Send section of Backstage.

### Handling Compatibility Issues

The most common compatibility issues found by the Web Compatibility Checker involve invalid names of tables or columns, compound multi-column indexes, incompatible lookup definitions, composite and text-based keys, and the use of table relationships to enforce referential integrity.

#### Invalid Names

Table and column naming restrictions are described in the previous section on tables. You must ensure that your names do not conflict with SharePoint reserved words and do not contain illegal characters. The Access Name AutoCorrect feature will propagate changes to dependent objects, such as queries and bound controls, but you should thoroughly inspect and test the application to ensure that no required changes were missed. VBA code and values in all expressions, for example, are not automatically corrected.

#### Compound Indexes

Indexes based on multiple columns are not supported in Web applications, as explained in the section on tables above.

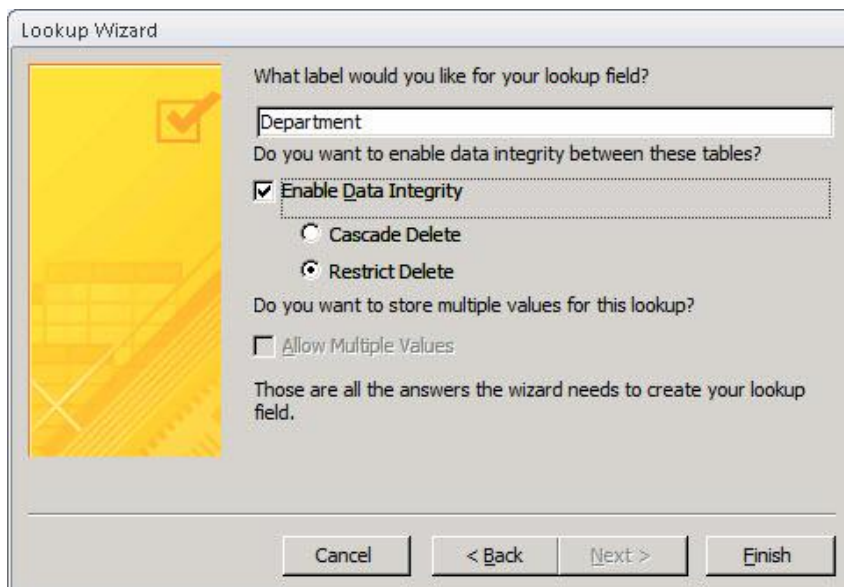
#### Lookup Definitions

Access tables support queries in lookup definitions that are not supported in SharePoint lists. SharePoint Server requires the input source to be a single table with a Long Integer primary key. SQL queries in lookup definitions also must not contain the DISTINCT or DISTINCTROW keywords. When lookups use value lists, the bound column must be the first column.

## Referential Integrity

Declarative referential integrity is not supported for SharePoint lists. Instead properties have been added to SharePoint Server 2010 lookups to enforce data restrictions. Users can opt to prohibit insertions, deletions, and updates that would create "orphaned" rows in "child" lists.

For example, suppose you have a list of employees with a column showing the department of each employee. The Department column is a lookup to a separate Departments list. Using the Lookup Wizard in Access to configure the column, you can select Enable Data Integrity, as shown in the following figure. This prevents an employee being assigned to a department that doesn't appear in the Departments list.



In some cases, you want to restrict deletions in parent tables to avoid creating orphans, as with Departments and Employees, but in other cases you want those deletions to propagate, or "cascade," to the child list. For example, with Orders in one list and Order Items in another, you may want to allow users to delete an order and automatically delete the related line items for that order. In that case, you choose Cascade Delete rather than Restrict Delete in the Lookup Wizard.

These lookup properties are also supported in unpublished Access 2010 databases for local Access tables, but they are separate from the Relationships window that users are familiar with for configuring referential integrity in previous versions. Users must configure a lookup and set Enable Data Integrity before publishing an Access table to SharePoint Server. If referential integrity has already been configured using the Relationships window in Access, then users will have to delete the relationship before they can use the Lookup Wizard, which is invoked from the list of field types in the table designer. Tables with relationships that aren't implemented in lookups can't be published, and those lookups must be based on columns that have a numeric data type of Long Integer.

## Primary and Foreign Keys

In non-Web local Access tables, Access supports the use of composite primary and foreign keys, which combine the values in two or more columns to create the key. Access also supports the use of a variety of data types for primary and foreign keys, including text and dates. These are not supported for published applications, because composite and string values cannot be used to create SharePoint Server lookups. String values can be displayed in lookups but the underlying relationship is always based on a numeric ID.

If users have composite primary and foreign keys based on multiple columns or even on text columns, which is quite common, they will need to change to using a Long Integer numeric key before they publish, or they will receive a compatibility error.

The easiest way to achieve compatibility is to add an autonumber column to the parent table, such as Department, and to add a corresponding Long Integer column in the child table as the foreign key. You can then use an update query to place the correct foreign key values in the child table.

For example, if you have a Departments table with a text primary key named Department and you've used these department names as foreign keys in the Employees table, add an autonumber DepartmentID column to the Department table and a Long Integer DepartmentID column to the Employees table. Then run this Access query:

```
UPDATE Departments INNER JOIN Employees ON Departments.Department = Employees.Department SET Employees.DepartmentID = Departments.DepartmentID;
```

You would also need to delete the old relationship and create a new one using the Lookup Wizard. Additional work would be required to change existing queries, forms, reports, and VBA code to use the new DepartmentID column in the Employees table, allowing you to delete the old text-based foreign key.

### **Creating New Compatible Tables and Importing Legacy Data**

In a Web database, the table designer restricts the available options to ones that are Web compatible. It is often much easier to use this designer to create new tables than to work through all the issues raised when attempting to publish an incompatible legacy table. Create Lookup columns to enforce referential integrity, as explained in the previous section covering referential integrity.

You can then create a linked table pointing to your legacy data and create a (client only) append query to append data from the legacy tables to the new Web-compatible tables.

One disadvantage of this technique is that you cannot use Name AutoCorrect to fix up names used in legacy dependent objects such as queries and bound controls. You will need to do this manually and carefully test for errors. An alternative is to create client queries with columns aliased to the original names, which can simplify the changes required in dependent objects.

### **Synchronizing data between web tables and external sources**

Web forms, web reports, and web queries cannot work with data from external data sources, such as SQL Server tables or SharePoint lists outside the current application site. To work around this limitation, you may want to build administrative applications that

regularly copy data from external sources into the SharePoint lists of Web applications. This enables you to include the data in Web reports or display it read-only in Web forms. Several different strategies can support this.

One option is to create Access linked tables that connect to the external data, and to execute client queries that move the data into your Web tables. These queries can exist in the Web database you want to update or in another database that has tables linked to the SharePoint lists corresponding to your Web tables.

If the Web data that you want to maintain is not used in any lookups, then you can execute queries that first delete all the old data and then append the current data. If lookups require you to preserve existing key values in the data, then you can use a more complex process that updates existing values and handles deletions. In some cases, related child rows may also need to be deleted.

Another option is to perform the data maintenance on a local disconnected copy of your Web database and to rely on Access/SharePoint Server synchronization to propagate the changes automatically when the database is reconnected to the server.

To ensure that data maintenance runs automatically on a defined schedule, you could create a SQL Server Integration Services package. Alternatively, you can use a Windows scheduled task to open an administrative Access application with an autoexec macro or a startup option that executes the data maintenance and closes the application. You can also execute an Access macro from a command line using the /x command-line switch.

## Migrating Legacy Application Objects

You must recreate as Web objects any form and reports that you want users to run in a browser. You must also recreate as Web objects any supporting queries and macros. You need to create Web-compatible macros to replace any VBA code. Controls from legacy forms and reports cannot be copied and pasted into Web forms and reports, but control formatting can be copied and pasted.

Legacy application objects can remain in the database without interfering in any way with publishing, and design changes that you make can be synchronized with the server version of the database, enabling easy versioning and deployment. However, these objects can run on the Access client only. Centralizing storage of application objects on a computer that is running SharePoint Server by publishing the application to Access Services improves manageability even for databases that don't contain any Web objects and always run in the rich Access client.

## Handling Publishing, Compilation, and Runtime Errors

The Web Compatibility Checker inspects table designs and logs each incompatibility that it finds to a local table named Web Compatibility Issues. The most common incompatibilities relate to primary keys and lookups, which were discussed in the previous section on handling compatibility issues.

However, the Web Compatibility Checker doesn't detect certain types of incompatibilities, which either cause publishing errors when Access attempts to publish incompatible schema that the Web Compatibility Checker overlooked, or when Access attempts to populate tables with incompatible data.

After publishing succeeds, Web objects compile asynchronously and can generate compile errors. Even after successful compilation, runtime errors can result from invalid object definitions that don't interfere with compilation or from logic errors.

### Publishing Issues

Access logs publishing issues in a local table named Move to SharePoint Site Issues, and the message informing you that publishing failed provides a link to the table.

Most schema issues that cause publishing to fail after the Web Compatibility Checker reports success are related to expressions. The Expression Builder and IntelliSense guide users toward creation of valid expressions in Access, but users can easily enter invalid expressions and the Web Compatibility Checker does not evaluate them. As explained in the previous section on expressions, the expression services used on the client and on the server are different, and some expressions that are valid on the client are not valid on the server.

Another common cause of publishing errors is incompatible data, because the Web Compatibility Checker does not check data values, only data schema. Data values that are valid in Access but not in SharePoint Server will generate errors that are also logged to the local Move to SharePoint Site Issues table.

The following sections discuss several types of data incompatibility.

## URLs

The Hyperlink data type in Access uses an underlying memo column to store display text and URLs. SharePoint Server also supports Hyperlink columns. However, it performs validation on Hyperlink URLs that may reject data contained in Access Hyperlink columns. For example, relative URLs are incompatible and must be replaced with ones that are fully qualified. In addition, SharePoint Server rejects URLs with more than 255 characters.

## Dates

Access and SharePoint Server both store date/time values using the Double data type. The integer portion of the number represents the number of days since day 0, and the fractional part of the number represents time as a fraction of a full day. However, the two systems use different timelines, and SharePoint Server does not support dates with underlying numeric values less than 1. In Access, day 1 is December 30, 1899, and prior dates are stored as negative numbers. In SharePoint Server, day1 is January 1, 1900, and prior dates are not supported.

Many legacy Access applications contain dates that cannot convert to SharePoint Server. A common practice in Access is to use day-0 dates in columns designed to show time-only values. In addition, data entry errors in Access applications frequently result in dates prior to 1900, unless such errors have been prevented by validation rules.

To check for Access date values that are not Web compatible, you can create a query that filters for dates prior to January 1, 1900, for example:

```
SELECT InvoiceID, InvoiceDate FROM Invoices WHERE InvoiceDate < #1/1/1900#;
```

## Compilation Issues

Invalid expressions in data schema definitions, such as in validation rules and calculated columns, can cause publishing errors. However, invalid expressions in Web forms, reports, and queries surface only when the objects compile, which occurs asynchronously after publishing has succeeded.

Access Services logs compilation errors to the USysApplicationLog table, which is accessible through the View Application Log Table button in the Info section of Backstage view. Access uses the status bar to cue the user when issues are pending in the application log.

## Runtime Issues

Even after publishing and compilation succeed, invalid expressions can still cause runtime errors. For example, invalid expressions in form or report control sources don't surface only when the object executes, causing the control to display #Error.

When a macro fails at runtime, Access 2010 records the error in the application log.

Another type of runtime error relates to images. All the images in a Web application are available through a single image gallery and must be uniquely named. When synchronizing design changes, image naming conflicts may be resolved by appending "\_username" to the name of a new image. This won't generate an error, but the new or modified form or report

may unexpectedly display the wrong image, because the reference is to the original name. The affected image names and control properties must be modified to correct this.

## Upgrading Databases to Access 2010

Access 2010 supports the mdb file format for backward compatibility, but to use the new features, including support for Web databases, you must use the accdb format that was introduced in Access 2007. If you have a legacy database that you want to upgrade to take advantage of Access Services, you must convert it to an accdb file. You can expect a smooth upgrade for Access 2007 databases that are already in the accdb format.

For guidance on upgrading an mdb to an accdb, see the white paper, *Transitioning Your Existing Access Applications to Access 2007* (<http://msdn.microsoft.com/en-us/library/bb203849.aspx>).

### 64-bit VBA Issues

Office 2010 provides 64-bit support primarily to enable Excel and Project users to work with a much larger address space. There are no advantages to running the 64-bit version of Access 2010. However, users who need 64-bit support for Excel or Project may try to run Access and could encounter some incompatibilities in applications that run fine in 32-bit mode. When 64-bit Office is installed on a machine, the user is required to uninstall any 32-bit versions of Office applications, including prior versions. 32-bit versions can be installed after 64-bit is installed, but Microsoft has not thoroughly tested these scenarios. The best practice is to run any 64-bit Office instance on a machine dedicated to that version only. 64-bit Office does not support any ActiveX controls or any COM add-ins.

All compiled VBA must be recompiled in 64-bit instances, meaning that mde and accde Access applications will not run. VBA code containing Declare statements must be rewritten before being recompiled, because pointer and handle values can no longer be contained in variables using the Long data type. Instead they must use the new LongLong or LongPtr data types. In addition, the new PtrSafe indicator must be added after the Declare keyword (Declare PtrSafe...) for the code to compile successfully. Conditional compilation, using #If, must be used in code that needs to compile under both legacy 32-bit and new 64-bit versions of Office. A convenient workaround is to give the few users who really need 64-bit support separate machines for running the applications that need the added memory space.

## Summary

Access provides compelling benefits to end users, who love being able to create their own data tracking and reporting applications, and to IT departments that cannot otherwise fulfill all the application building requirements of their organizations. Access 2010 significantly extends its value proposition for users by integrating with SharePoint Server 2010 to support convenient creation of full-featured Web applications with broad reach, and client applications that are easily shared, revised, and deployed. Of equal significance are the manageability improvements provided through this deep integration with SharePoint Server.

## Appendix

### Access 2010 Features by Object Type

The following sections discuss the technologies used for implementing the various types of Web objects, the differences between Web and client objects, and many of the new features introduced in Access 2010.

#### Tables

Web databases do not support local tables. They must be compatible with SharePoint lists or publishing will fail, and they are always converted to SharePoint lists when you successfully publish or synchronize the database. This is enforced in the table view, which is used to modify tables in 2010. It only allows you to create schema that is compatible with SharePoint Server when you are working in a Web database. Not all system tables are moved to SharePoint Server. System tables other than log tables are not stored as tables on SharePoint Server. Any tables in a Web database that are linked to SharePoint lists outside the application's site, or to any other type of external data, can only run in the Access client. Linked tables can't be seen by Web objects in the database, even when running in the Access client. Web objects only use Web tables, which transparently link to SharePoint lists in the application site.

A configurable administrative setting determines the maximum size of attachments in Web tables, which may interfere with publishing or synchronizing if you have an attachment that is too big. The default limit is 50 MB. Web tables don't support multiple attachment columns in one record, and when Web tables are taken off line, you can't add an attachment.

SharePoint Server doesn't support certain table names that are allowed in Access client applications, and tables with illegal names will prevent publishing. The following illegal names conflict with reserved SharePoint list names: Lists, Docs, WebParts, ComMd, Webs, Workflow, WFTemp, Solutions, and ReportDefinitions. In addition, the following illegal names conflict with tables created during publishing: MSysASO, MSsLocalStatus, UserInfo, and USysApplicationLog.

To be publishable, table column names, as well as all Access objects, cannot contain the following characters and character pairs: /, \, :, \*, ?, \", <, >, |, #, {, }, %, ~, &, \t, ;.

SharePoint lists implement a form of indexing to speed filtering performance, and single-field indexes in Access tables propagate to the server. However, SharePoint Server doesn't support indexes composed of multiple columns, and multi-column Access indexes prevent tables from being Web compatible. This also means that composite keys and uniqueness constraints are not Web compatible. A workaround for enforcing uniqueness based on multiple column values is to use a BeforeChange data macro, discussed in the section on data macros below.

Native Access tables limit the total number of columns (to 255) and the total number of characters in a row (4000 with Unicode compression set), but they do not limit the number of columns of a particular type. To be Web compatible, however, tables must also conform to the SharePoint Server 2010 limits for each data type, which are 5 times greater than the limits enforced in previous versions. Here are default limits for how many columns of a

type you can have in a SharePoint list: date/time 40, bit 30, uniqueidentifier 5, number 60, note 160, and text 320.

The maximum number of lookup columns and multi-value columns is also limited. Published memo column values are truncated if they contain more than 8,192 characters.

Both native and Web tables support calculated columns based on expressions. This is a new feature in Access 2010 that can provide significant performance improvements compared to calculating values when queries execute. Centralizing the calculation definition in the table also improves reliability and consistency. If the application logic changes, you make the change in one place, rather than attempting to find every place the calculation was used. Because the calculation is maintained by the database engine, this storage of a calculated value does not violate normalization rules. Expressions that are not Web-compatible, which might also appear in column and table validation rules, will prevent publishing, even in Web databases that appear to meet the requirements of the compatibility checker. The section below on expressions provides more details on expression compatibility.

## Forms

Web forms in Access 2010 enable users with no Web development experience to create full-featured Web pages. The design experience is familiar to anyone who has worked with Access client forms, and easy to learn for new users. On the server, Access Services does the work of translating the user's design into ASP.NET pages that run in any standard browser (IE7, IE8, Firefox, and Safari are all explicitly supported). These pages do not use any ActiveX controls. Macros that users attach to form and control events are implemented as JavaScript code. Popup forms are implemented as floating divs. Design themes are implemented as CSS style sheets. The resulting pages are highly responsive and frequently employ asynchronous JavaScript with XML (AJAX) to refreshing views rather than performing full postbacks, which provides snappy performance. The browser Back and Forward buttons are fully supported.

Like all Access objects in Web applications, forms are serialized using the open Access Application Transfer Protocol (MS-AXL), which is documented at <http://msdn.microsoft.com/en-us/library/dd927584.aspx>. Forms also make use of the open standard XAML protocol used by Windows Presentation Foundation (WPF). Access uses these protocols to synchronize design changes with the server and to generate ASP.NET pages.

In the Access designer, Layout view for creating Web forms and reports is enhanced to make it much easier to control the exact positioning of controls by splitting and combining columns and cells. On the server, these layouts get implemented as hidden HTML tables. Design view is not available for Web forms.

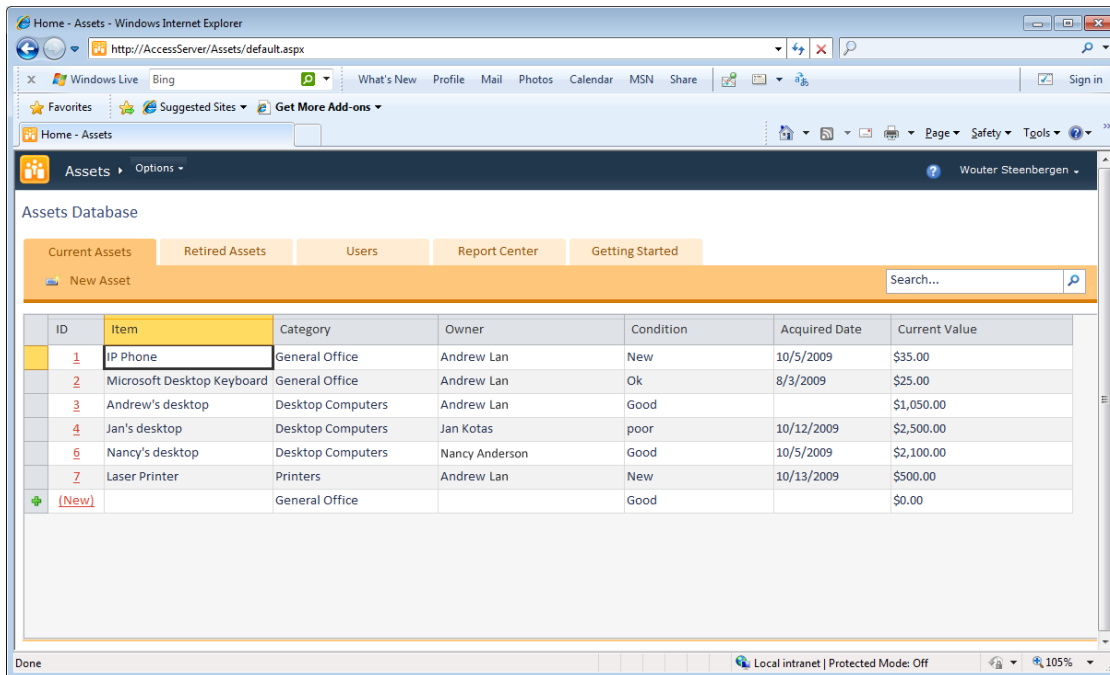
Experienced Access users with little or no Web development experience are likely to be delighted at their ability to create attractive, highly functional pages using only the skills they've developed in Access. Several new form features are geared specifically toward creating "Web-like" interfaces.

For example, the new Navigation form creates Web forms (or client forms) with versatile hierarchical menus than can display other forms and reports embedded in the resulting page. Parameters can filter the record source query of the displayed object. A new Browser

control supports parameterized URLs based on form control values, enabling easy creation of "mashup" interfaces that embed maps or other context-specific external content.

Web forms provide parity in the feature sets available in the browser and in the Access client, as do all Web objects. However, an application may need to behave differently based on the runtime environment. Web applications support separate Web and Client startup form properties, available in Current Database settings in Backstage view. In addition, the IsServer and IsClient expressions return True or False, allowing macros to branch based on the runtime environment. Web forms "just work" in the client with no special programming required. The figures that follow show the same form, first in the Access client and then in a browser.

ID	Item	Category	Owner	Condition	Acquired Date	Current Value
1	IP Phone	General Office	Andrew Lan	New	05/10/2009	\$35.00
2	Microsoft Desktop Keyboard	General Office	Andrew Lan	Ok	03/08/2009	\$25.00
3	Andrew's desktop	Desktop Computers	Andrew Lan	Good		\$1,050.00
4	Jan's desktop	Desktop Computers	Jan Kotas	poor	12/10/2009	\$2,500.00
6	Nancy's desktop	Desktop Computers	Nancy Anderson	Good	05/10/2009	\$2,100.00
7	Laser Printer	Printers	Andrew Lan	New	13/10/2009	\$500.00
* (New)						



One difference between Web and client forms could occasionally cause confusion: client forms allow expressions to reference all columns that appear in the record source of the form, even if those columns are not bound to controls on the form. However Web forms only support references to columns that are bound to controls on the form. Users may need to add invisible controls to work around this limitation. The use of such hidden controls is a common practice in Access client reports, which have always enforced a similar restriction.

## Reports

Web reports use SQL Server Reporting Services. They are deployed to the server using AXL and Report Definition Language (RDL). As described below, the Access Database Service mediates all data access to the SharePoint lists of a Web database, providing the same caching behavior and performance benefits available to forms.

Web reports support exporting to PDF from the browser, which provides a great printing experience, and users can also export to Word or Excel document formats. A handy new feature in Access 2010 enables subforms to host reports, and Navigation forms display reports by using this feature.

Both forms and reports support standard and custom Office Themes for configuring display appearance. Organizations can specify preferred themes as a way of encouraging design consistency. When a user changes a database theme, all the forms and reports that use the theme are affected. To propagate theme changes in Web objects to the server, you must open the objects in layout view, save them, and then synchronize. This pattern also applies to Name AutoCorrect, an Access client feature that propagates name changes to dependent objects. The dependent objects aren't renamed until you open and save them, and you can then synchronize to rename the corresponding objects on the server.

To publish and synchronize successfully, Web forms, reports, and controls, as well as all other Access objects, cannot contain the following characters and character pairs: /, \\, :, \*, ?, \", <, >, |, #, {, }, %, ~, &, \t, ;. In addition, Web form and report controls must have names that begin with an upper or lower case letter or an underscore (no numbers), and they must contain only upper or lower case letters, underscores, or numbers (the naming rules for C# variables). This is more restrictive than the list of allowed characters in names for client forms and reports.

In Web forms, record source queries that include lookup columns automatically join to the related tables to display the text values for lookups. However, in Web reports this automatic joining behavior for lookups doesn't occur — you must explicitly add the related tables to your report queries. Design view is not supported for Web reports.

## Queries

Web queries are stored and implemented by Access Services using AXL and Collaborative Application Markup Language (CAML). The query processor in Access Services caches execution plans as well as data, and pushes as much filtering to the server as possible to improve performance. On disconnected clients, Access uses the client-side ACE query processor to execute Web queries against locally cached data. Client queries can work with Web tables or linked tables from other data sources and can join the two.

SQL View is not available for Web queries in the query designer. Users must employ the Access query design grid, which enforces Web query restrictions. In the Access client, you can still use VBA to get a Web query's SQL representation by retrieving the SQL property of a DAO QueryDef object. In the Immediate Window of the Visual Basic Editor, enter the following, replacing "QueryName" with the actual name of your query:

```
?CurrentDb.QueryDefs!QueryName.SQL
```

Web queries support projection of selected columns from multiple joined data sources, including both inner and outer joins. The data sources for a Web query can include other saved Web queries in addition to tables. Web queries also support filtering based on multiple criteria, including expressions, sorting based on multiple columns, and calculated columns based on expressions.

However, Web queries do not support the full range of features available in client queries. They do not support aggregates, crosstabs, unions, Cartesian products (cross joins), subqueries, or any actions that modify data or create new objects.

Several client query properties are unavailable in Web queries, including Output All Fields ("SELECT \*" in SQL), Top Values (TOP), Unique Values (DISTINCT), UniqueRecords (DISTINCTROW), Max Records, and Subdatasheet properties.

Although Web queries cannot calculate aggregate values, aggregation is fully supported on Web reports. In addition, as explained below, data macros can maintain aggregate values in tables.

Web queries support the use of parameters. Macros that open forms, reports, and query datasheets, and ones that set subform source object property values with the new BrowseTo macro action, all allow you to specify parameter values for the record source queries of these objects. You can use any valid expression to set the parameter, including expressions that refer to data values in form controls.

Traditional Access client queries are very flexible about how users can define parameters. A Parameters dialog allows users to specify the name and data type of each parameter, but for most queries this is optional. Users can simply enclose any text in square brackets and the Access client query processor treats the value as a parameter name unless it is the name of a column in the query. In Web queries, however, all parameters must be explicitly defined with the Parameters dialog.

Expressions that define calculated columns or criteria in Web queries must be compatible with the Excel-based expression library that Access Services uses. For more information on this, see the Expressions section below. In addition, expressions in Web queries cannot reference controls on forms or macro variables.

Configurable administrative settings limit many aspects of Web query design to protect resource usage. For example, you can limit the number of outer joins, which are resource intensive, in addition to the number of output columns, data sources, etc. See the section covering administration below for more details. Also, note that each lookup column a query uses adds an extra data source, because of the hidden join needed to retrieve the displayed value.

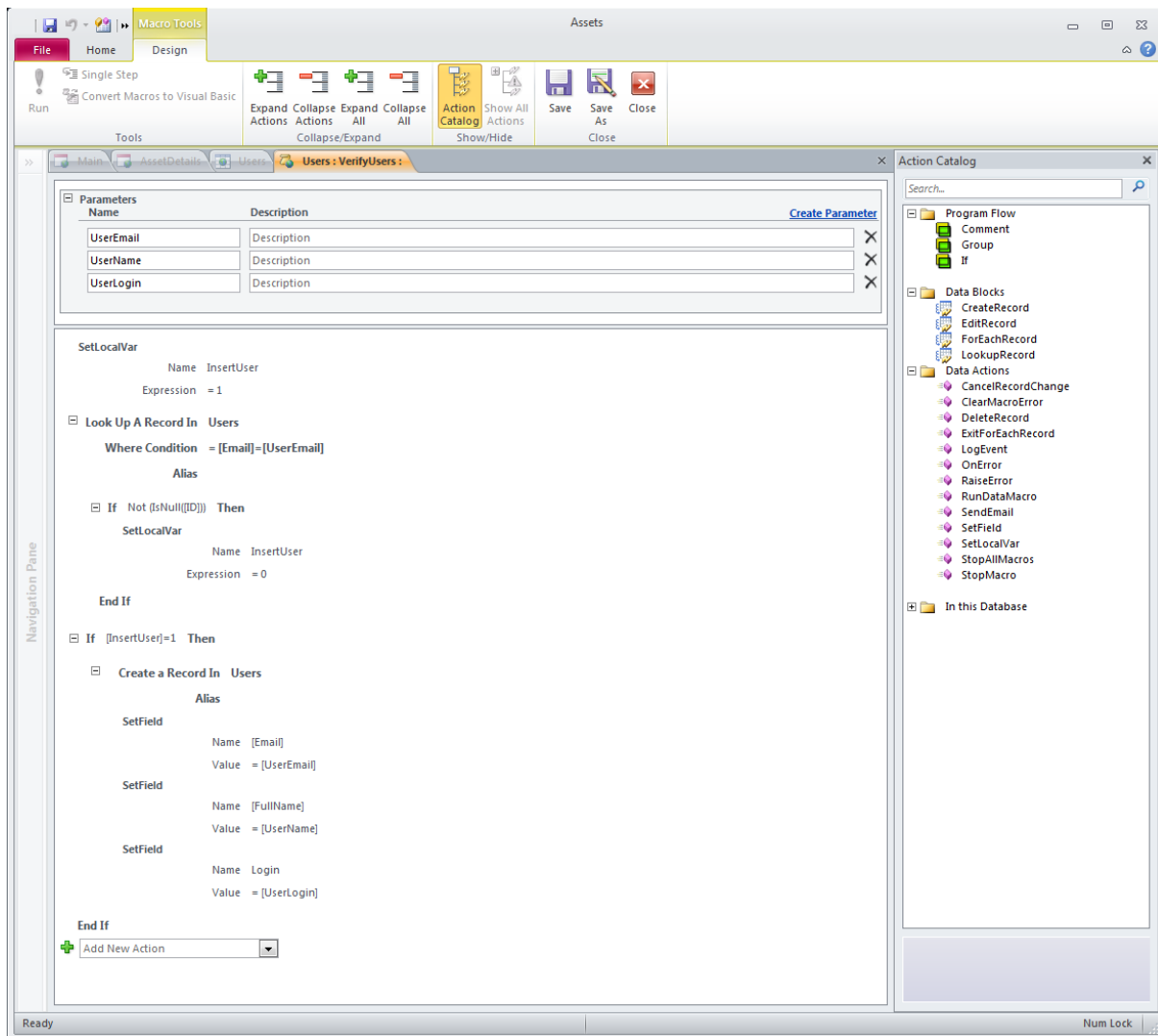
## Macros

Web objects do not support VBA code. Programmability for Web objects relies instead on Access macros, which have been significantly enhanced to provide greater ease of use, security, resilience, and manageability.

The macro designer was completely recreated in Access 2010 to improve ease of use and to support more complex logic. Users select from context sensitive options to generate readable, structured, collapsible blocks of code. Full IntelliSense guides users to appropriate syntax and argument values.

Access 2010 supports two different types of macros. UI macros, which are simply called Macros in the Access user interface, extend the capabilities of traditional Access macros to respond to user actions and to control application flow and navigation. In Web forms running in a browser, these macros are implemented as JavaScript. Data macros, which are new in Access 2010, are similar to SQL triggers. They run in response to data modifications. On the server, data macros are implemented as SharePoint Workflow actions.

These two types of macros create a clean separation between presentation tier and data tier code in Access applications. An architectural shortcoming of many traditional Access applications that rely heavily on VBA code is that they often muddy the distinction between these logical tiers. UI macros can only perform data-related actions by calling named macros, which are saved data macros that aren't attached to specific table events. Data macros can also call named macros, supporting code reuse and maintainability. Named macros support parameters, as shown in the following figure.



In addition to parameters, data macros support the use of local variables, and UI macros support both local and global variables. Originally added in Access 2007, macros support robust error handling, and for debugging, the MacroError object provides properties, such as Number, Description, and ActionName, which you can record in the application log with the new LogEvent action. You can also nest If/Then/ElseIf/Else blocks to create complex conditional logic. Macros can serialize to XML or load from XML, to support sharing and reuse.

All macros that run in Web objects on the server are mediated by Access Services with configurable throttles to ensure safe execution. Web macros support a subset of the macro actions that client macros support, and client macros can use a sandboxed subset of actions to create applications that don't require full trust.

## Data Macros

Data Macros, introduced in Access 2010, are available for Web tables in Web databases or native tables in unpublished databases. Even tables linked to Access data in other databases support data macros, but the data macros must be defined in the database containing the native tables, not the links. Data macros use an event model similar to that of triggers in SQL Server, to enable reliable enforcement of data rules.

Once you define a data macro for a table event, that macro will run no matter how the data is accessed. This provides a significant new capability for Access that enables much more application reliability than was previously available for Access tables.

In past versions, the Access database engine was able to enforce referential integrity defined in relationships between tables, and domain integrity enforced by table-level and column-level validation rules. Users could also configure columns to enforce rules for unique and required values. (These constraints are all still supported in Web databases.) Any other rules for data, however, relied on logic in data entry forms for enforcement. Developers attempted to prevent users from circumventing the rules by hiding tables from view, but this was never foolproof. In addition, having data rules enforced in multiple forms and possibly in multiple applications invited inconsistency.

In tables that have been published to SharePoint lists, the rules are enforced on the server by SharePoint Workflow actions. In native Access tables, data macro execution is enforced locally by the Access database engine, which provides parity with the actions available to server-side data macros.

Here are a few examples of data macro scenarios you might find in a Donations Management database:

- Validate that a contributor doesn't have outstanding donations before accepting a new donation.
- Keep a history of any changes to a donation record.
- Send a "Thank You" email when a contributor makes a donation greater than \$1,000.
- Maintain a total of all donations and the last donation date in summary columns of the contributors table. (Although Web reports support aggregates, Web queries do not. So using data macros to maintain aggregate values in tables can be useful.)

You can attach data macros to the BeforeChange, BeforeDelete, AfterInsert, AfterUpdate, and AfterDelete events of tables. Data macros attached to After events, and all UI macros, can call named data macros associated with a table. The calling macros can pass in parameter values and can get back a collection of return values as well as errors. Errors are also logged to the USysApplicationLog table, which is easily discovered in Backstage view and is maintained in both Web and non-Web databases.

The BeforeChange and BeforeDelete events are designed to support fast, light-weight operations. Data macros attached to these events can inspect the old and new values in the current record, and can compare them with a record in the current table or another table by using LookupRecord. They can also use SetField to alter data in the row being changed or prevent the change from occurring. To assure that the operations remain lightweight,

however, they cannot iterate over a collection of records. The BeforeChange event fires for both inserts and updates, but data macros can use the IsInsert expression to distinguish the type of operation.

The AfterUpdate, AfterInsert, and AfterDelete events can support more long-running operations that require iteration. The old and updated values are available, and macros invoked from these events can inspect and modify other records in the table or in other tables. Typically, users should not use these events to modify the current record; the BeforeChange and BeforeDelete events are more appropriate.

Occasionally, a user may need a data macro to modify the current record, potentially causing the macro to be called again recursively. Data macros are limited to 10 levels of recursion, but they can call the Updated ("FieldName") function, which returns True or False, to determine which column or columns were affected by the current change. Judicious use of this feature can usually prevent cyclical recursion.

A few cautions concerning data macros:

- In some instances, when SharePoint lists are taken offline in disconnected Access applications, data macro execution is delayed until the user reconnects. Data changes made on the disconnected client are automatically propagated to the server when the connection is restored, and the data macros run on the server.
- Unlike SQL Server triggers, data macros are not performed within a transactional context. Access 2010 does not provide transactional contexts for any serial data operations, which are all atomic.
- Data macros cannot process data from multi-valued or attachment columns.
- Access 2007 SP1 can read but not write data in linked Access 2010 tables with data macros, because the Access 2007 data engine can't execute them.

## Expressions

Access supports the use of expressions in form and report control sources and events, query criteria, calculated columns in queries and tables, validation rules for tables, columns, and controls, default values for columns or controls, and macro arguments.

Access expressions are similar to Excel formulas, and Access Services uses a modified version of the Excel Calculation Services library. One important modification is to support the use of database nulls. However, this library does not provide complete parity with the Access client expression service.

The Expression Builder is significantly improved in Access 2010 to show a context-sensitive list of available options and to provide IntelliSense support, which is available anywhere that users can enter expressions.

Incompatible expressions used in the validation rules or calculated columns of Access tables in unpublished client databases may not be detected by the compatibility checker and could cause compile errors when the database is published.

Access could also fail to detect an incompatible expression in the design of a Web object, causing a runtime error when the object executes. For example, a form control that displays "#Error" could indicate that its control source uses an invalid expression. A Web query

containing an incompatible expression returns a runtime error indicating an invalid expression.

Access 2010 adds support for expression keywords targeted at Web applications. For example, you can use CurrentWebUser to get the email, display name or network name of the current user when IsServer is true.

Here are a few expression issues that can cause errors when executed on the server:

- You must fully qualify control references: Use Forms!MyForm!MySubform.Form!MyControl, not MySubform.Form!MyControl
- Don't rely on type coercion. For example, If conditions must return Boolean values. Use If (15<>0), not If (15). When possible, use the Format function to convert expressions to the proper type.
- Dates do coerce to doubles, but they use a different numbering system on the server. SharePoint Server does not recognize dates prior to 1/1/1900. Use FormatDateTime if you need to convert to strings.
- For Booleans, use True/False, not -1/0.
- Access Services doesn't support the DateAdd, DatePart, and Date Diff functions. Instead, use DateSerial, Day, Month, Year
- Field references in expressions in forms must refer to fields used in bound controls.
- You can't use the Between operator, which is commonly used in expressions in query criteria. Use >= and <= instead.

Expressions in legacy databases require scrutiny to assure successful publishing, but the new server-based expression service is very full-featured and supports almost all the tasks that Access users have come to expect. In addition, the new Expression Builder makes it much easier to create compliant expressions in Web objects.

This is a preliminary document and may be changed substantially prior to final commercial release of the software described herein.

The information contained in this document represents the current view of Microsoft Corporation on the issues discussed as of the date of publication. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication.

This White Paper is for informational purposes only. MICROSOFT MAKES NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, AS TO THE INFORMATION IN THIS DOCUMENT.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

Unless otherwise noted, the example companies, organizations, products, domain names, e-mail addresses, logos, people, places and events depicted herein are fictitious, and no association with any real company, organization, product, domain name, email address, logo, person, place or event is intended or should be inferred.

© 2009 Microsoft Corporation. All rights reserved.

Microsoft, Excel, InfoPath, MSDN, the Office logo, Outlook, PowerPoint, SharePoint, Visual Basic, Visual Studio, Win32, and Windows are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

The names of actual companies and products mentioned herein may be the trademarks of their respective owners.