

Microsoft® Virtual Labs

**Developing a SQL Mobile
Application with Visual Studio
2005 and SQL Server 2005**

Microsoft®

Table of Contents

Developing a SQL Mobile Application with Visual Studio 2005 and SQL Server 2005	1
Exercise 1 Creating and Using a SQL Mobile Database in a Windows Mobile 5.0 Application	2
Exercise 2 Synchronizing Data Between SQL Server 2005 and SQL Mobile	24
Exercise 3 Synchronizing Data Between Any Backend Database and SQL Mobile Database Using a Web Service.	59
Summary.....	65

Developing a SQL Mobile Application with Visual Studio 2005 and SQL Server 2005

Objectives

After completing this lab, you will be better able to:

- Create and Use a SQL Mobile Database in a Windows Mobile 5.0 Application
- Synchronize Data Between SQL Server 2005 and SQL Mobile
- Synchronize Data Between Any Backend Database and SQL Mobile Database Using a Web Service

Scenario

Learn how to use SQL Server 2005 Mobile Edition (SQL Mobile) to synchronize data between a Windows Mobile-based device and a SQL Server 2005 backend database. You will use SQL Server Management Studio to create a SQL Mobile database and to set up and configure SQL Server 2005 and IIS for merge replication. You will use Visual Studio 2005 to create a .NET Compact Framework application to maintain the SQL Mobile data and synchronize it with SQL Server data. You will also learn how to synchronize SQL Mobile data with any backend data store using a Web service. When you have completed this lab, you will know how to easily build a mobile application that keeps mission-critical data in synchronization with a backend database.

Because completing all exercises in the lab may take more time than you have available, you may select only those exercises that are most useful for you. Each exercise provides you with files that offer a starting point as if you had completed the previous exercise. While there is no explicit order in which you have to perform the exercises, each exercise builds on the activities performed in the previous ones.

Estimated Time to Complete This Lab

90 Minutes

Computer used in this Lab



MEDC

Exercise 1

Creating and Using a SQL Mobile Database in a Windows Mobile 5.0 Application

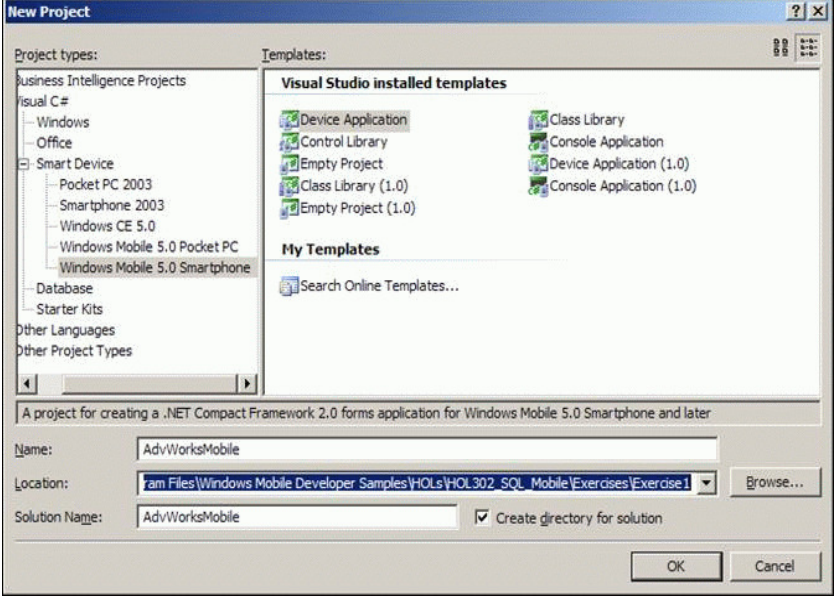
Scenario

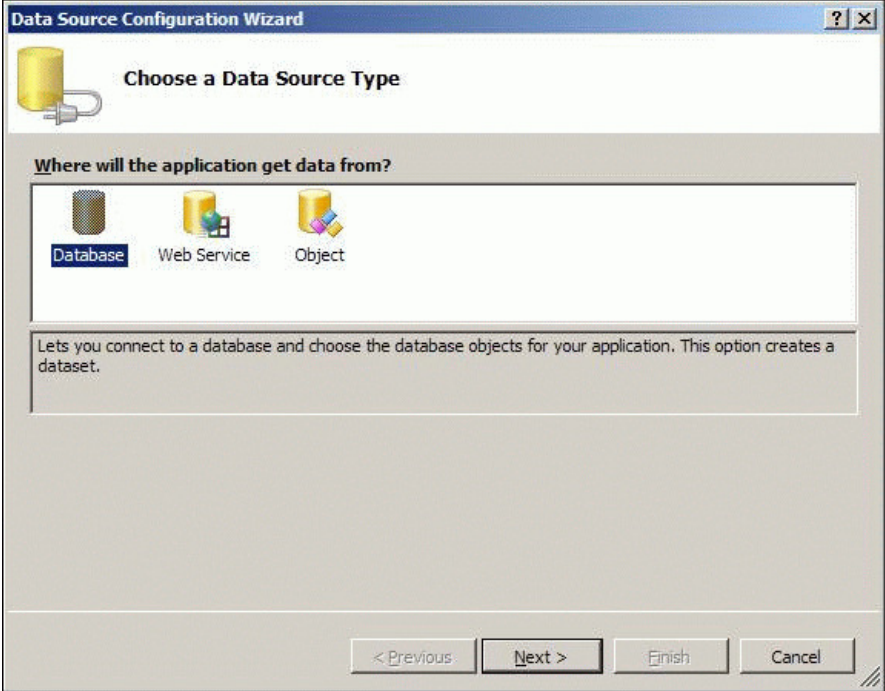
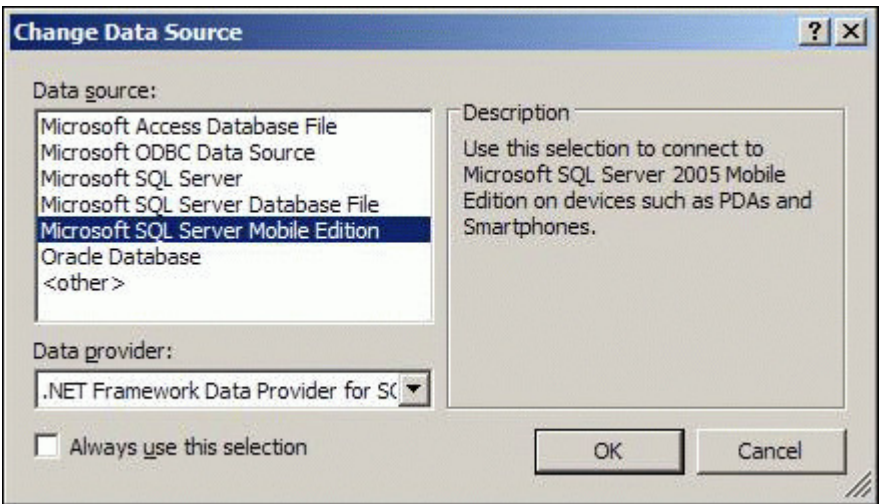
In this exercise, you will use SQL Server Management Studio to create a stand-alone SQL Mobile vendor database. You will then create a Windows Mobile 5.0 application to view and maintain the vendor data within the SQL Mobile database. The vendor database will be based on the AdventureWorks sample database that ships with SQL Server 2005.

Tasks	Detailed Steps
1. Create a SQL Mobile database	a. Start SQL Server Management Studio by clicking Start All Programs Microsoft SQL Server 2005 SQL Server Management Studio. b. In the Connect to Server dialog box, select SQL Server Mobile in the Server type box, as shown in Figure 1.  Figure 1. Selecting SQL Server Mobile as the server type c. In the Database file box, select <New Database...>. The Create New SQL Server 2005 Mobile Edition Database dialog box appears, as shown in Figure 2.

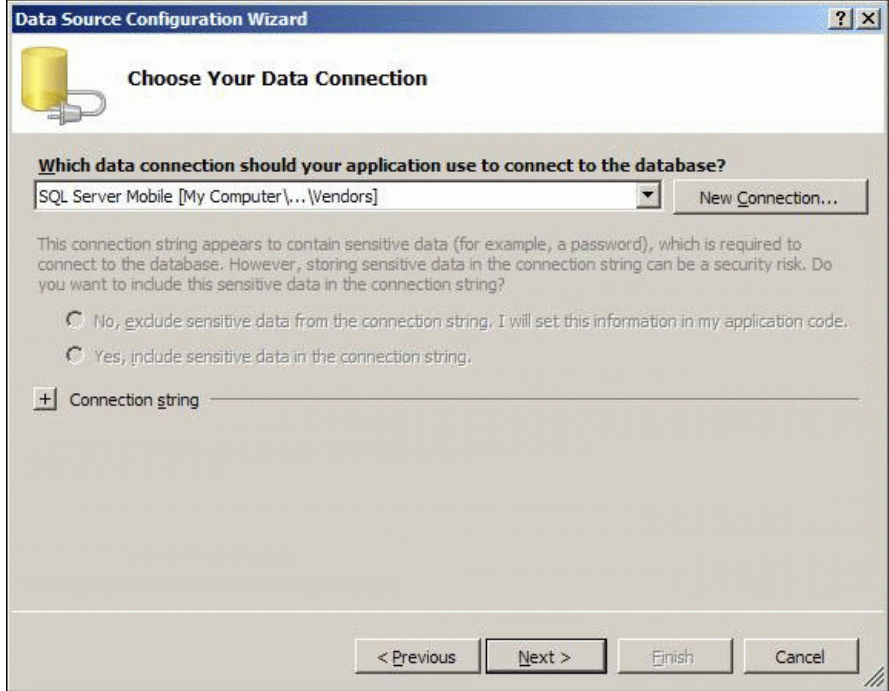
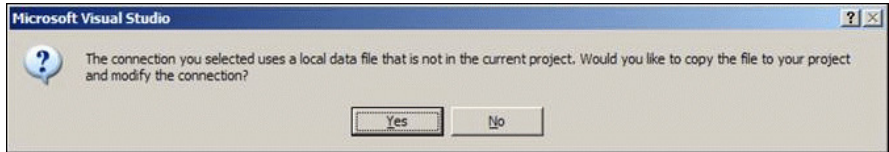
Tasks	Detailed Steps
	Figure 2. Creating a new SQL Server 2005 Mobile Edition database d. Under Enter the new SQL Server 2005 Mobile Edition database filename, type C:\Program Files\Windows Mobile Developer Samples\HOLs\HOL302_SQL_Mobile\Exercises\Exercise1\Vendors.sdf. e. Leave the other values as their defaults, and then click OK. f. If you are prompted with a message that says, Password is either not provided or blank, click Yes to continue with the blank password. <i>Note: Step f is not a recommended security practice, but for the purpose of simplicity in this lab, you will use this setting.</i> g. Click Connect to connect to the new SQL Mobile database, as shown in Figure 3. Figure 3. Connecting to the new SQL Server 2005 Mobile Edition

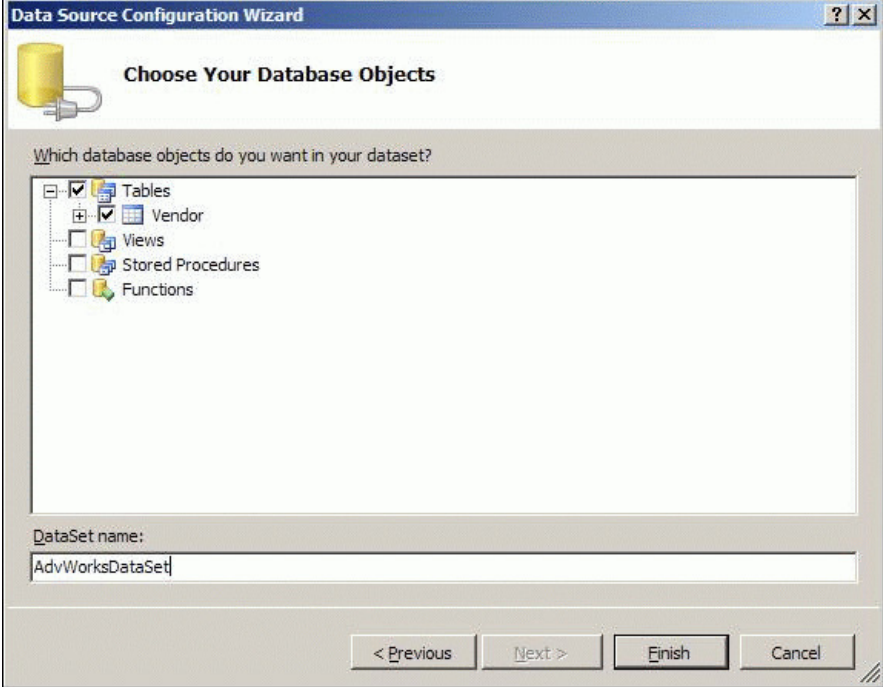
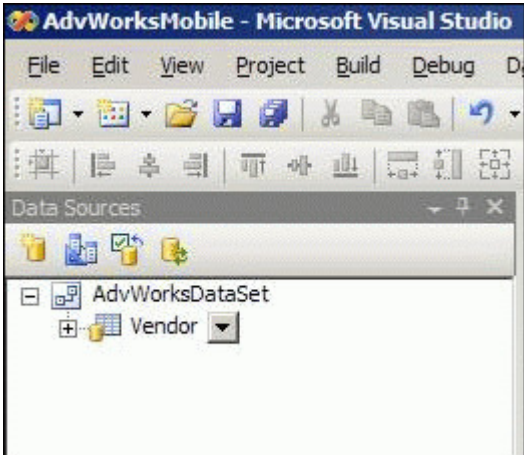
Tasks	Detailed Steps
	database <i>Note: Although you can create tables and other database objects interactively, just as you can when connected to a SQL Server 2005 database, a script is provided for convenience to create the Vendor table in your new SQL Mobile database in the next procedure.</i>
2. Create a table for the new SQL Mobile database	a. In SQL Server Management Studio, click File Open File. b. Browse to C:\Program Files\Windows Mobile Developer Samples\HOLs\HOL302_SQL_Mobile\Setup, and then select CreateVendor.sqlce. Click Open. c. Verify that the full path of your new Vendors.sdf database file is selected in the Database file box, and then click Connect. d. Click Query Execute to create the Vendor table. The Message pane appears with the message (1 row(s) affected) repeated four times. e. In the Object Explorer pane, right-click the Tables folder, and then click Refresh. f. Expand the Tables folder, and then verify that there is now a Vendor table. g. Expand the Vendor table and its Columns folder to familiarize yourself with the table's structure, which happens to conform to the schema of the Vendors table in the AdventureWorks sample database. <i>Note: In the next exercise, you will see why having the table's structure conform to the schema is desirable. The table contains four records that you can browse by right-clicking the SQL Server Mobile database node, and then by clicking New Query. In the query editor, type SELECT * FROM Vendor, and then click Query Execute. When you are finished browsing, close the Query Editor window. There is no need to save your changes when prompted.</i> h. Close SQL Server Management Studio.
3. Create a new Windows Mobile 5.0 application for Pocket PC	a. Start Visual Studio 2005 by clicking Start All Programs Microsoft Visual Studio 2005 Microsoft Visual Studio 2005. b. Click File New Project to create a new Windows Mobile application. c. In the New Project dialog box under Project types, browse to Visual C# Smart Device Windows Mobile 5.0 Pocket PC. <i>Note: Depending on your Visual Studio configuration, Visual C# may appear under Other Languages.</i> d. In the Templates box, select Device Application. e. Change the Name to AdvWorksMobile. f. Change the Location to C:\Program Files\Windows Mobile Developer Samples\HOLs\HOL302_SQL_Mobile\Exercises\Exercise1. g. Be sure Create directory for solution is selected, and then click OK, as shown in Figure 4.

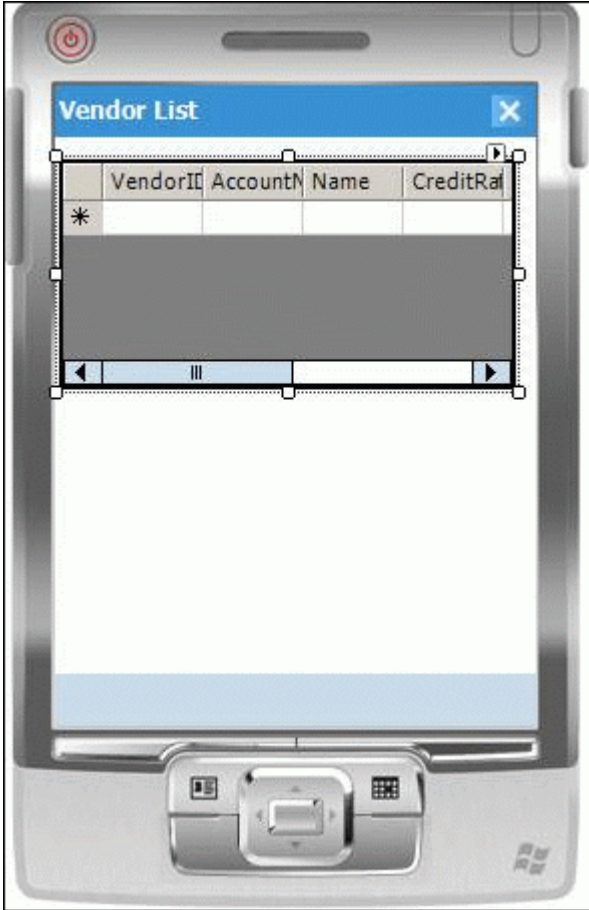
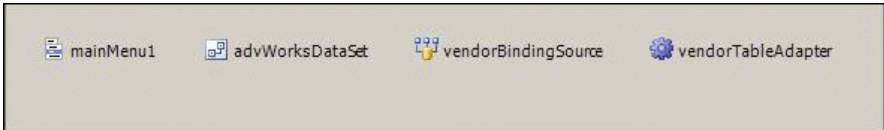
Tasks	Detailed Steps
	 Figure 4. The New Project dialog box <i>Note: Visual Studio creates a new project and opens Form1 in the form designer.</i>
4. Create the Vendor List form	- In Solution Explorer in Visual Studio, right-click Form1.cs, and then click Rename. - Type VendorList.cs as the new file name, and then press ENTER. - Click Yes in response to the prompt to rename all references of the Form1 code element in the project. - If the VendorList form is not already open in the form designer, double-click VendorList.cs in the Solution Explorer pane to open it in the form designer. - In the form designer, right-click the blank area of the form, and then click Properties. - In the Properties pane, change the Text property value to Vendor List. - Change the MinimizeBox property value to False. <i>Note: This step causes an ok button to appear at the top right corner of the form when it is displayed on the device instead of the default smart minimize button (which looks like an x). Normally, when the x is clicked, Windows Mobile applications are minimized to the background where they are still running. Changing this property value to False causes the form to actually be closed when clicking ok, making debugging easier because the application needs to be deployed multiple times, which would not be possible if the application were still running in the background.</i>
5. Add the SQL Mobile Vendors database as a project data source	- In Visual Studio, click Data Add New Data Source. <i>Note: The Data Source Configuration Wizard appears.</i> - Select Database as the data source type, and then click Next, as shown in Figure 5.

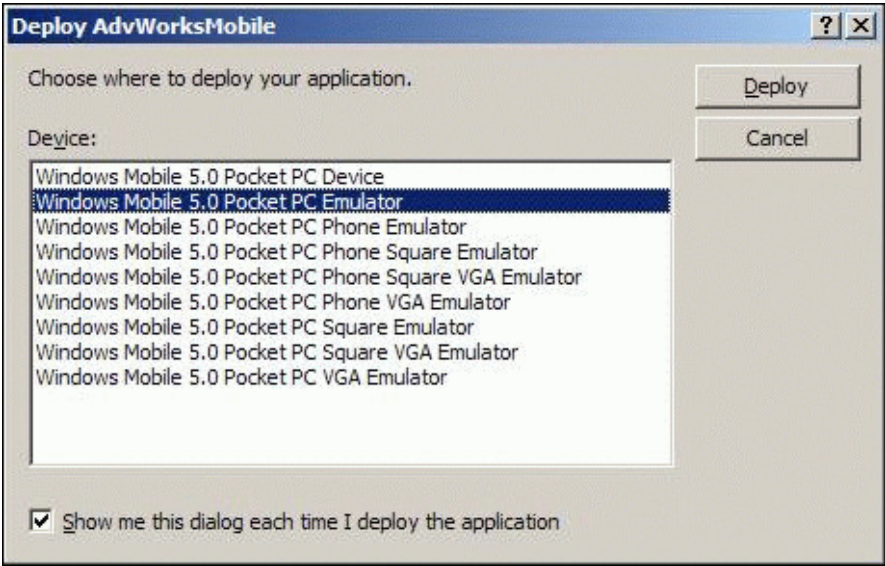
Tasks	Detailed Steps
	 The screenshot shows the 'Data Source Configuration Wizard' window. The title bar says 'Data Source Configuration Wizard'. Below the title bar is a yellow folder icon and the text 'Choose a Data Source Type'. The main area is titled 'Where will the application get data from?' and contains three icons: 'Database' (a cylinder), 'Web Service' (a cube with a network icon), and 'Object' (a cube with a document icon). The 'Database' option is selected. Below the icons is a text box that says: 'Lets you connect to a database and choose the database objects for your application. This option creates a dataset.' At the bottom are four buttons: '< Previous', 'Next >', 'Finish', and 'Cancel'. Figure 5. Choosing a data source type in the Data Source Configuration Wizard c. On the Choose Your Data Connection page, click New Connection. d. In the Add Connection dialog box, click Change. e. Select Microsoft SQL Server Mobile Edition, and then click OK, as shown in Figure 6.  The screenshot shows the 'Change Data Source' dialog box. The title bar says 'Change Data Source'. It has two main sections. The left section is titled 'Data source:' and contains a list box with the following items: 'Microsoft Access Database File', 'Microsoft ODBC Data Source', 'Microsoft SQL Server', 'Microsoft SQL Server Database File', 'Microsoft SQL Server Mobile Edition' (which is highlighted), 'Oracle Database', and '<other>'. The right section is titled 'Description:' and contains the text: 'Use this selection to connect to Microsoft SQL Server 2005 Mobile Edition on devices such as PDAs and Smartphones.' Below the list box is a 'Data provider:' label and a dropdown menu showing '.NET Framework Data Provider for SQL'. At the bottom left is a checkbox labeled 'Always use this selection'. At the bottom right are 'OK' and 'Cancel' buttons. Figure 6. Changing the data source f. In the Add Connection dialog box, be sure the My Computer option is selected as the Data Source. g. Under Connection Properties, click Browse to select the database. h. Browse to C:\Program Files\Windows Mobile Developer Samples\HOLs\HOL302_SQL_Mobile\Exercises\Exercise1, click Vendors.sdf,

Tasks	Detailed Steps
	and then click Open. i. Click OK to add the connection, as shown in Figure 7. Figure 7. The Add Connection dialog box j. On the Choose Your Data Connection page, after the new connection appears in the Which data connection should your application use to connect to the database box, click Next, as shown in Figure 8.

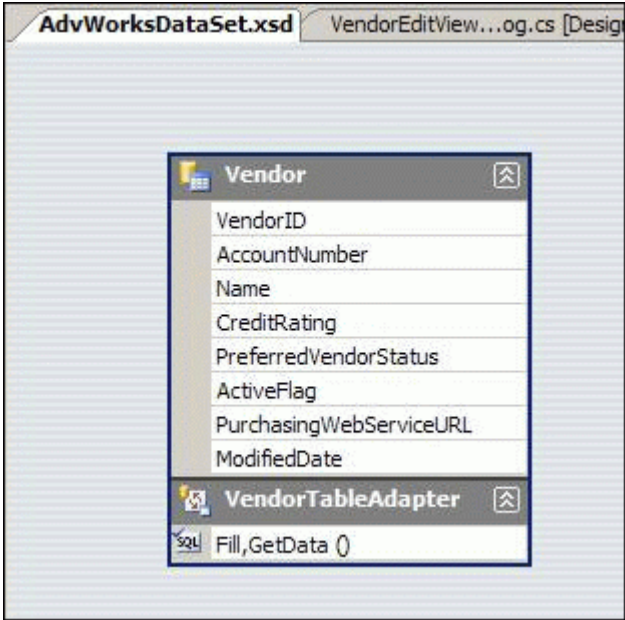
Tasks	Detailed Steps
	 Figure 8. Choosing Your Data Connection page with the new connection selected k. When Visual Studio prompts you to copy the local data file to your current project and modify the connection, click Yes, as shown in Figure 9.  Figure 9. Message to copy the local data file to your current project <i>Note: Visual Studio copies the Vendors.sdf file into your project folder and sets its properties such that the file will also be copied to your device when the application is deployed.</i> l. On the Choose Your Database Objects page, expand Tables, and then select Vendor. m. Change the DataSet name to AdvWorksDataSet, and then click Finish, as shown in Figure 10.

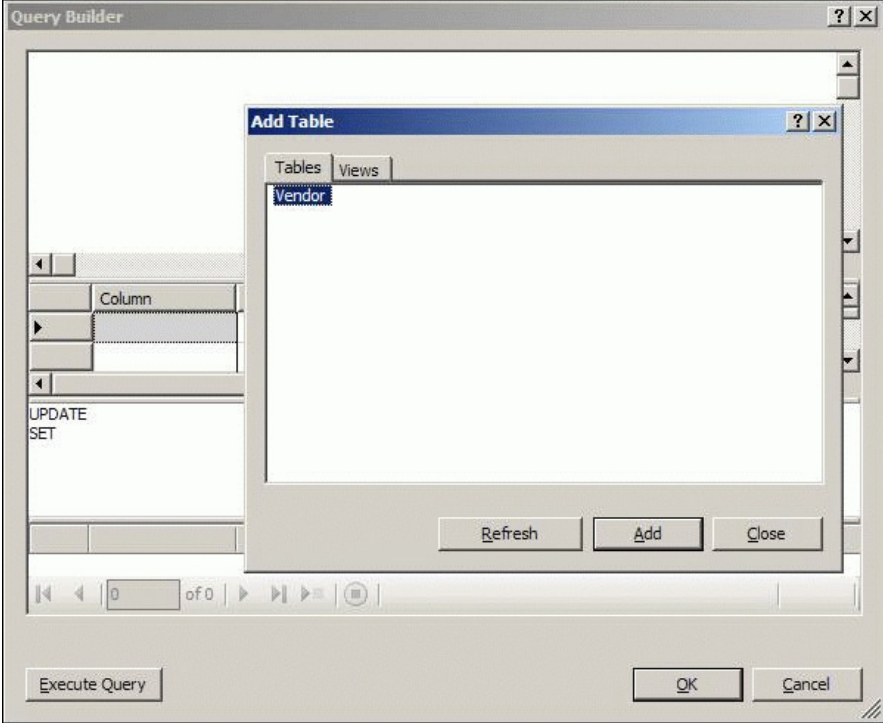
Tasks	Detailed Steps
	 The screenshot shows the 'Data Source Configuration Wizard' window with the title 'Choose Your Database Objects'. It asks 'Which database objects do you want in your dataset?'. A tree view on the left shows 'Tables' and 'Vendor' selected with checkboxes. Other options like 'Views', 'Stored Procedures', and 'Functions' are unchecked. Below the tree is a text box for 'DataSet name:' containing 'AdvWorksDataSet'. At the bottom are buttons for '< Previous', 'Next >', 'Finish', and 'Cancel'. Figure 10. Choose Your Database Objects page <i>Note: Visual Studio adds the AdvWorksDataSet.xsd schema and other related files to your project.</i>
6. Add content to the Vendor List screen	a. The VendorList.cs Form Designer should still be visible in Visual Studio, but if it is not, double-click VendorList.cs in Solution Explorer to open it in the form designer. b. Click Data Show Data Sources to show the Data Sources pane if it is not already visible, as shown in Figure 11.  The screenshot shows the Visual Studio interface with the 'Data Sources' pane open. The pane title is 'Data Sources'. It contains a tree view with 'AdvWorksDataSet' expanded, showing a sub-item 'Vendor' with a dropdown arrow. Figure 11. The Data Sources pane <i>Note: You can drag and drop tables and individual columns to the form designer to create controls that are bound to the data source. By default, a DataGrid control is created when dragging an entire table to a form designer.</i> c. In the Data Sources pane, expand AdvWorksDataSet. d. Click and drag the Vendor table, and then drop it anywhere in the blank area on

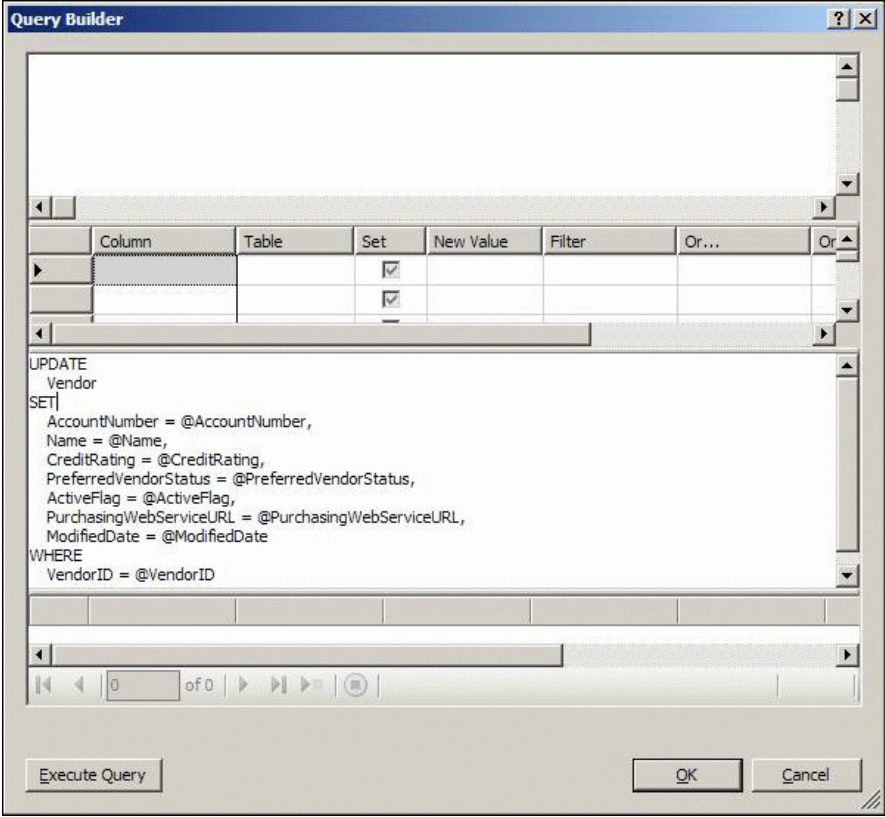
Tasks	Detailed Steps
	the VendorList.cs form designer. <i>Note: Visual Studio creates a DataGrid control for the Vendor table, as shown in Figure 12.</i>  Figure 12. The DataGrid control for the Vendor table in the form designer <i>Note: Notice that in the component tray (located at the bottom of the form designer) Visual Studio has also created the advWorksDataSet DataSet, the vendorBindingSource BindingSource, and the vendorTableAdapter TableAdapter, as shown in Figure 13. Visual Studio uses these controls that are invisible at run time to bind the vendor data to the DataGrid.</i>  Figure13. The component tray in Visual Studio e. Right-click the DataGrid control, and then click Properties. f. In the Properties pane, set the Location property to 0, 0, and then set the Size property to 240, 268, effectively filling the whole screen. <i>Note: You are now ready to test the application's functionality.</i>
7. Test the application	a. In Visual Studio , click Debug Start Debugging .

Tasks	Detailed Steps
	b. In the Deploy AdvWorksMobile dialog box that appears, be sure Windows Mobile 5.0 Pocket PC Emulator is selected, and then click Deploy, as shown in Figure 14.  Figure 14. The Deploy AdvWorksMobile dialog box Note: If you receive an error message that there was an error deploying the application any time that you deploy the application during this lab, be sure the application is not already running, and then try to start debugging again. Visual Studio launches the Pocket PC - WM 5.0 device emulator and begins to deploy the application to the device. Deploying the first Windows Mobile 5.0 application may take some time because the .NET Compact Framework version 2.0 and your application's files—including the SQL Mobile database in this case—need to be copied to the device. Note: When the application appears in the device emulator, as shown in Figure 15, you should see the list of four vendors that were inserted into the table when you created the database. You can scroll right and left and resize the columns with the column dividers on the header row to see all of the fields in the table.

Tasks	Detailed Steps
	Figure 15. The emulator running the application with the Vendor table c. Click ok to dismiss the form, and then close the application. <i>Note: Besides simply viewing a list of vendors, your application needs to allow the user to edit the vendor data. You will use the DataGrid's smart tag tasks to generate additional data forms, but before you do that, you need to set the default values for some data columns that do not allow null values.</i>
8. Generate add and edit functionality	a. In Visual Studio, open the VendorList.cs form in the form designer if it is not open already. b. In the component tray, click the advWorksDataSet component, and then click the smart tags button at the top-right corner of the control. The smart tag menu appears, as shown in Figure 16. Figure 16. The smart tag menu for the advWorksDataSet

Tasks	Detailed Steps
	component c. In the smart tag menu for AdvWorksDataSetTasks, click Edit in DataSet Designer. <i>Note: The AdvWorksDataSet DataSet designer opens, displaying the Vendor table, as shown in Figure 17.</i>  Figure 17. The AdvWorksDataSet DataSet designer d. Click the row in the Vendor table that contains the PreferredVendorStatus column name. In the Properties pane, locate the DefaultValue property, and then change its value to True. e. In the same way as Step d, click the ActiveFlag column name in the Vendor table, and then change its DefaultValue property to True. <i>Note: Now you need to add the ability to write data changes from the DataSet to the SQL Mobile database.</i> f. In the DataSet Designer, click VendorTableAdapter. g. In the Properties pane, locate the UpdateCommand property. h. In the Update Command property box, select (New). i. Expand the UpdateCommand property, click the CommandText property, and then click its ellipsis button. <i>Note: The Query Builder dialog box appears with the Add Table dialog box, as shown in Figure 18.</i>

Tasks	Detailed Steps
	 Figure 18. The Query Builder and Add Table dialog boxes j. On the Add Table dialog box, click Close. k. In the Query Builder dialog box, type or paste the following SQL statement, as shown in Figure 19, which follows the code. <pre> UPDATE Vendor SET AccountNumber = @AccountNumber, Name = @Name, CreditRating = @CreditRating, PreferredVendorStatus = @PreferredVendorStatus, ActiveFlag = @ActiveFlag, PurchasingWebServiceURL = @PurchasingWebServiceURL, ModifiedDate = @ModifiedDate WHERE VendorID = @VendorID </pre>

Tasks	Detailed Steps
	 Figure 19. The Query Builder dialog box with a SQL statement l. Click OK. m. Close the AdvWorksDataSet.xsd DataSet designer. When you are prompted to save your changes, click Yes. n. In the VendorList.cs form designer, click the DataGrid control, and then click the smart tags button at the top-right corner of the control. The smart tag menu appears, as shown in Figure 20.

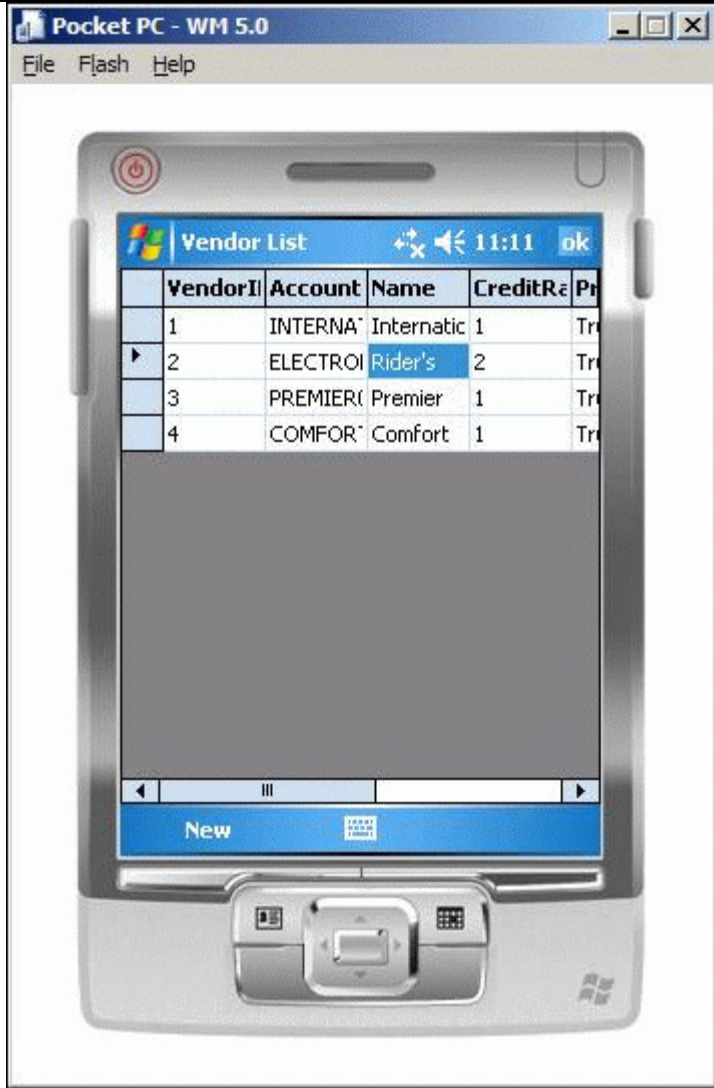
Tasks	Detailed Steps
	Figure 20. The smart tag menu for the DataGrid control o. In the smart tag menu for DataGrid Tasks, click Generate Data Forms. Note: Visual Studio generates two new forms in Solution Explorer: VendorEditViewDialog.cs and VendorSummaryViewDialog.cs, as shown in Figure 21. Figure 21. Two new forms in Solution Explorer Note: Visual Studio has also added a New menu to the VendorList main menu, as shown in Figure 22.

Tasks	Detailed Steps
	Figure 22. The New menu in the VendorList main menu <i>Note: In addition to these readily-visible changes Visual Studio has made to your project, Visual Studio has also added code behind the VendorList form and the new forms it has generated.</i> p. In Solution Explorer, right-click VendorList.cs, and then click View Code. q. In Code view, locate the newMenuItemMenuItem_Click method. This new method that has been added to the form's code responds to the user clicking the New menu. The method begins by adding a new record to the vendorBindingSource data binding source; next, it creates an instance of the VendorEditViewDialog form, passing it the vendorBindingSource variable; finally, it displays the form instance that it just created. r. Locate the vendorDataGrid_Click method. This method has been added to the form's code, and it responds to the user clicking the DataGrid control. The method creates an instance of the VendorSummaryViewDialog form, passing it the vendorBindingSource variable, and then it shows the form. s. In Solution Explorer, double-click VendorEditViewDialog.cs to open the form designer. <i>Note: Notice that the form has an individual data-bound control for each column in the Vendor table.</i> t. Scroll to and right-click the text box under Modified Date, and then click Properties. u. Double-click the Locked property to change its value to True, and then change the BorderStyle property to None. In a moment, you will add code to modify this control's value before the record is saved, so the control's text should not be edited by the user. v. Click View Code to view the form's code. w. Notice that in the VendorEditViewDialog_Closing method, it calls the EndEdit method of the vendorBindingSource data binding source that it received when it was instantiated, thus saving any data that has been added or edited since the form was opened. x. Create a new line before the call to EndEdit and assign the current date and time to the ModifiedDate column. You can easily access the data binding source as shown in the following code example. <pre data-bbox="602 1776 1321 1894"> DataRowView dataRowView = (DataRowView)this.vendorBindingSource.Current; AdvWorksDataSet.VendorRow vendorRow = (AdvWorksDataSet.VendorRow)dataRowView.Row; </pre>

Tasks	Detailed Steps
	<pre data-bbox="544 195 1398 226">vendorRow.ModifiedDate = DateTime.Now;</pre> Note: A more full-featured implementation of this form would be to provide functionality that would allow the user to cancel their changes, but to save time, that functionality is left as an optional exercise. y. In Solution Explorer, double-click VendorSummaryViewDialog.cs to open the form designer. Note: Notice that this form also has an individual data-bound control for each column in the Vendor table, but the controls are read-only. Also notice that the form has an Edit command. z. View the form's code and locate the editMenuItemMenuItem_Click method, which responds to the Edit command being clicked. It behaves much like the newMenuItemMenuItem_Click method on the VendorList form; it simply relies on the fact that the vendorBindingSource variable's current record will be edited when the VendorEditViewDialog form is opened. This command works because the controls on this form are databound to the same vendorBindingSource instance as the DataGrid on the VendorList form. Note: You need to add the ability to add any data changes made in your application to the SQL Mobile database file when the user closes the application.
9. Add functionality to add modified data to the SQL Mobile database	a. In Solution Explorer, double-click VendorList.cs to return to the VendorList form designer. b. Click the form's title bar. In the Properties pane, click the Events button (with a lightning bolt symbol on it). c. Locate and double-click the Closing event to generate an event handler and open the event in Code view. d. Add the following code to the VendorList_Closing event handler. <pre data-bbox="544 1150 1398 1182">this.vendorTableAdapter.Update(advWorksDataSet);</pre> Note: This code uses the vendorTableAdapter table adapter to update the SQL Mobile database file with the data changes that have been made to the advWorksDataSet DataSet. Note: While the generated forms and code provide add and edit functionalities with the single click of a command, a more full-featured implementation would be to provide functionality that would allow the user to delete a record, but to save time, that functionality is left as an optional exercise.
10. Test application features	a. In Visual Studio, click Debug Start Debugging. b. In the Deploy AdvWorksMobile dialog box, be sure Windows Mobile 5.0 Pocket PC Emulator is selected, and then click Deploy. c. When the application appears in the device emulator, click a row in the DataGrid. Note: You may need to click the device emulator's button on the Windows taskbar to bring the emulator to the foreground. d. Notice that the VendorSummaryViewDialog form appears with the details of the row that you clicked, as shown in Figure 23.

Tasks	Detailed Steps
	The screenshot shows a Pocket PC interface with a window titled 'Pocket PC - WM 5.0'. Inside the window is a form titled 'VendorSummary'. The form contains the following fields and values: Vendor ID: 2 Account Number: ELECTRON0002 Name: Electronic Bike Repair _Supplies Credit Rating: 1 Preferred Vendor Status: True Active Flag: True Modified Date: (field is present but the value is not clearly visible) At the bottom of the form is a blue button labeled 'Edit'. Figure 23. Details about a row that was clicked in the Vendor table e. Click Edit to open the data form that you can edit, as shown in Figure 24.

Tasks	Detailed Steps
	The image shows a screenshot of a Pocket PC emulator window titled 'Pocket PC - WM 5.0'. Inside the emulator, a form titled 'VendorEditViewDialog' is displayed. The form has a blue header bar with the Windows logo, the title 'VendorEditViewDialog', and icons for zooming and volume. The time '11:08' and an 'OK' button are also visible. The form contains several input fields and checkboxes: 'Account Number' with the value 'ELECTRON0002', 'Name' with 'Electronic Bike Repair & Supplies', 'Credit Rating' with '1', 'Preferred Vendor Status' with a checked checkbox, 'Active Flag' with a checked checkbox, 'Purchasing Web Service URL' (empty), and 'Modified Date' (empty). The form is presented on a virtual device with a screen and a physical keypad. Figure 24. Editing a data form in the emulator f. After the VendorEditViewDialog form appears, modify some of the data, and then click OK to save it. Notice that the Vendor List form's DataGrid is updated to reflect the data changes you made, as shown in Figure 25.

Tasks	Detailed Steps																									
	 The screenshot shows a Pocket PC interface with a 'Vendor List' application. The application window has a title bar 'Pocket PC - WM 5.0' and a menu bar with 'File', 'Flash', and 'Help'. The main display area shows a table with the following data: <table><thead><tr><th>VendorID</th><th>Account</th><th>Name</th><th>CreditRate</th><th>Priority</th></tr></thead><tbody><tr><td>1</td><td>INTERNA</td><td>Internatic</td><td>1</td><td>Tr</td></tr><tr><td>2</td><td>ELECTROI</td><td>Rider's</td><td>2</td><td>Tr</td></tr><tr><td>3</td><td>PREMIER</td><td>Premier</td><td>1</td><td>Tr</td></tr><tr><td>4</td><td>COMFOR</td><td>Comfort</td><td>1</td><td>Tr</td></tr></tbody></table> Below the table is a large grey rectangular area. At the bottom of the screen is a blue bar with a 'New' button. The device has a physical keypad and a screen with a power button at the top. Figure 25. The updated Vendor List g. Click New. h. After the VendorEditViewDialog form appears, enter data for a new record, as shown in Figure 26.	VendorID	Account	Name	CreditRate	Priority	1	INTERNA	Internatic	1	Tr	2	ELECTROI	Rider's	2	Tr	3	PREMIER	Premier	1	Tr	4	COMFOR	Comfort	1	Tr
VendorID	Account	Name	CreditRate	Priority																						
1	INTERNA	Internatic	1	Tr																						
2	ELECTROI	Rider's	2	Tr																						
3	PREMIER	Premier	1	Tr																						
4	COMFOR	Comfort	1	Tr																						

Tasks	Detailed Steps
	 Figure 26. Entering a new record i. Click ok to save the data. Notice that the Vendor List form's DataGrid is updated to reflect the new record that you entered, as shown in Figure 27.

Tasks	Detailed Steps
	 Figure 27. The updated Vendor List with the new record j. Click ok to dismiss the form, and then close the application. k. Close Visual Studio 2005. Note: In this exercise, you have learned how to create a stand-alone SQL Mobile database and how to create a Windows Mobile application that can be used to maintain this stand-alone data. In the next exercise, you will learn how to synchronize data between your mobile database and a backend SQL Server database.

Exercise 2

Synchronizing Data Between SQL Server 2005 and SQL Mobile

Scenario

In this exercise, you will enhance the Windows Mobile 5.0 application that you created in Exercise 1 to synchronize data between the SQL Server 2005 AdventureWorks sample database and the SQL Mobile database that your application uses. You will use merge replication to accomplish the synchronization so your device does not always need to be connected to a network or backend database to be fully functional.

Tasks	Detailed Steps
1. Create a merge replication publication	- Start SQL Server Management Studio. Click Start All Programs Microsoft SQL Server 2005 SQL Server Management Studio to open it. - In the Connect to Server dialog box, click the Server type box, and then click Database Engine if it is not already selected. - In the Server name box, select the entry with your computer name (VSTOWB-4051400). <i>Note: Be sure that you select the entry with just your computer name, and not the entry with your computer's name followed by \SQLEXPRESS.</i> - In the Authentication box, be sure Windows Authentication is selected. - Click Connect. - In the Object Explorer pane, expand Replication, right-click Local Publications, and then click New Publication. <i>Note: The New Publication Wizard appears, as shown in Figure 28.</i>

Tasks	Detailed Steps
	Figure 28. The New Publication Wizard g. Click Next. h. On the Distributor page, be sure 'VSTOWB-4051400' will act as its own Distributor; SQL Server will create a distribution database and log is selected, as shown in Figure 29.

Tasks	Detailed Steps
	Figure 29. The Distributor page i. Click Next. <i>Note: If the Snapshot Folder page does not appear, your computer has already been configured for merge distribution. If you are sure that your Windows user account has permissions to read the network snapshot share, you may skip to Step s; however, it is highly recommended for the purposes of this lab to configure merge distribution as described in the following steps.</i> j. On the Snapshot Folder page, make a note of the Snapshot folder value or copy it to Clipboard (by pressing CTRL+C), as shown in Figure 30.

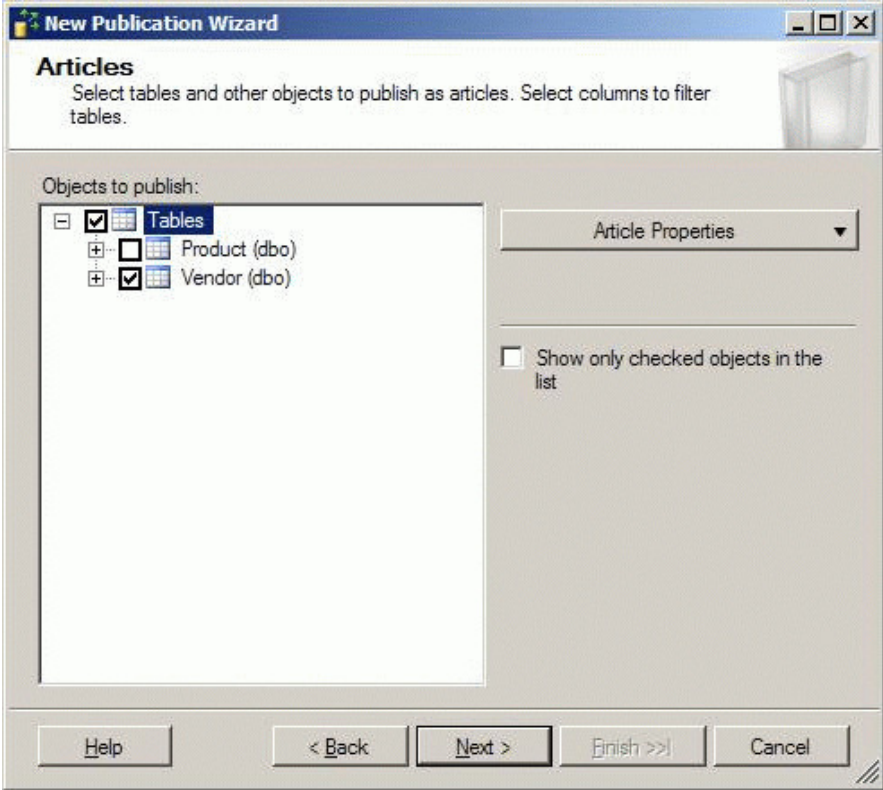
Tasks	Detailed Steps
	Figure 30. The Snapshot Folder page k. On the desktop computer, click Start Run. l. On the Snapshot folder page of the wizard, copy the path from Snapshot folder box. m. In the Run dialog box, paste the path into the Open box, and then click OK, as shown in Figure 31. Figure 31. The Run dialog box n. In the Windows Explorer window, click Tools Folder Options. o. In the Folder Options dialog box, click the View tab, and then scroll to the bottom of the Advanced settings list. p. Be sure to clear the Use simple file sharing (Recommended) option, and then click OK, as shown in Figure 32.

Tasks	Detailed Steps
	Figure 32. The Folder Options dialog box q. Right-click anywhere in the blank area of the Windows Explorer file and folder area (it is probably empty), and then click Properties. r. In the Properties dialog box, click the Sharing tab, and then select Share this folder, as shown in Figure 33.

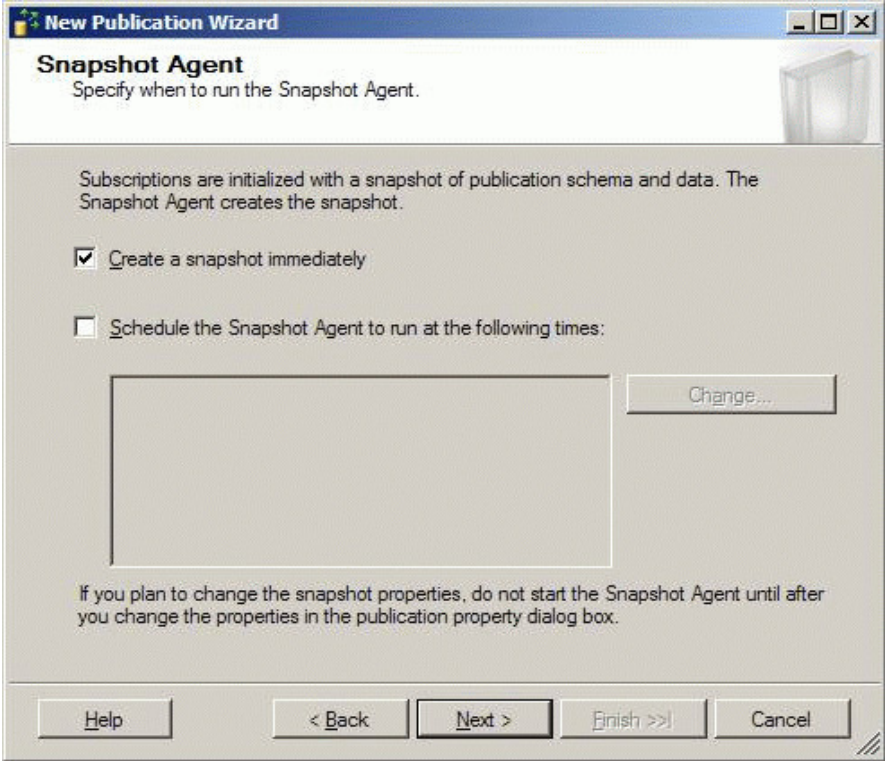
Tasks	Detailed Steps
	The screenshot shows the 'ReplData Properties' dialog box with the 'Sharing' tab selected. It contains instructions on how to share the folder, radio buttons for 'Do not share this folder' and 'Share this folder' (which is selected), a text field for 'Share name' containing 'repldata', an empty 'Comment' field, and radio buttons for 'Maximum allowed' and 'Allow this number of users'. There are buttons for 'Permissions' and 'Caching'. At the bottom, it states 'Windows Firewall is configured to allow this folder to be shared' with a link to 'View your Windows Firewall settings'. The 'OK', 'Cancel', and 'Apply' buttons are at the bottom right. Figure 33. Sharing a folder s. Click the Permissions button. t. In the Permissions for Everyone list, check the Allow checkbox next to Full Control, and then click OK. u. Click OK, and then close the Windows Explorer window. v. In the New Publication Wizard, click Next on the Snapshot Folder page. w. On the Publication Database page, be sure AdventureWorks is selected as the publication database, as shown in Figure 34, and then click Next.

Tasks	Detailed Steps
	Figure 34. The Publication Database page x. On the Publication Type page, select Merge publication as the Publication type, and then click Next, as shown in Figure 35. Figure 35. Selecting a publication type

Tasks	Detailed Steps
	y. On the Subscriber Types page, clear the SQL Server 2005 option, and then select SQL Server 2005 Mobile Edition, as shown in Figure 36. Click Next. Figure 36. The Subscriber Types page z. On the Articles page under Objects to publish, expand Tables, select Vendor, and then click Next, as shown in Figure 37.

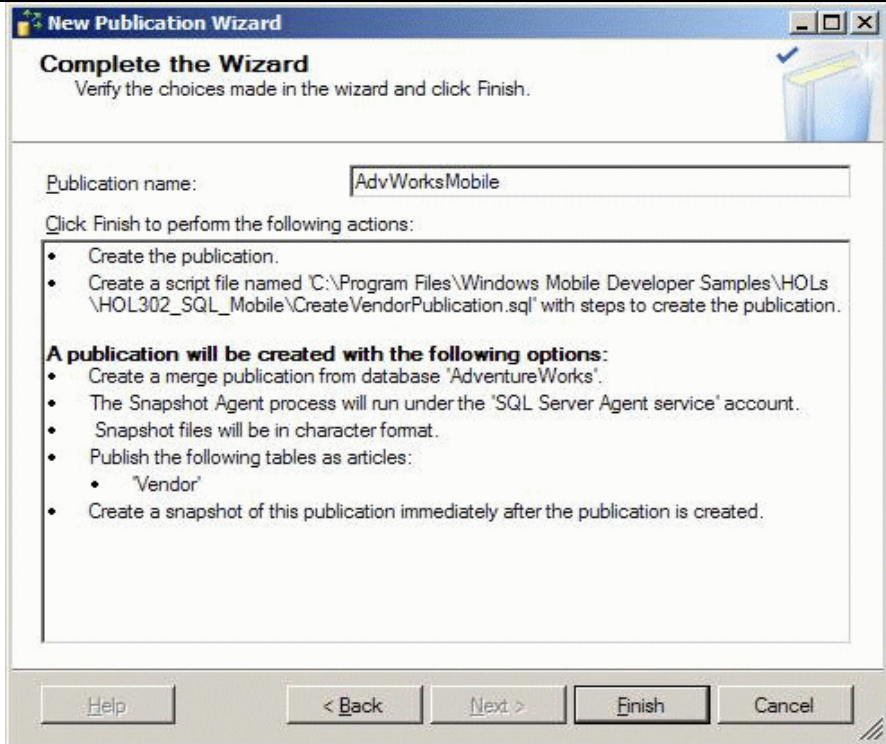
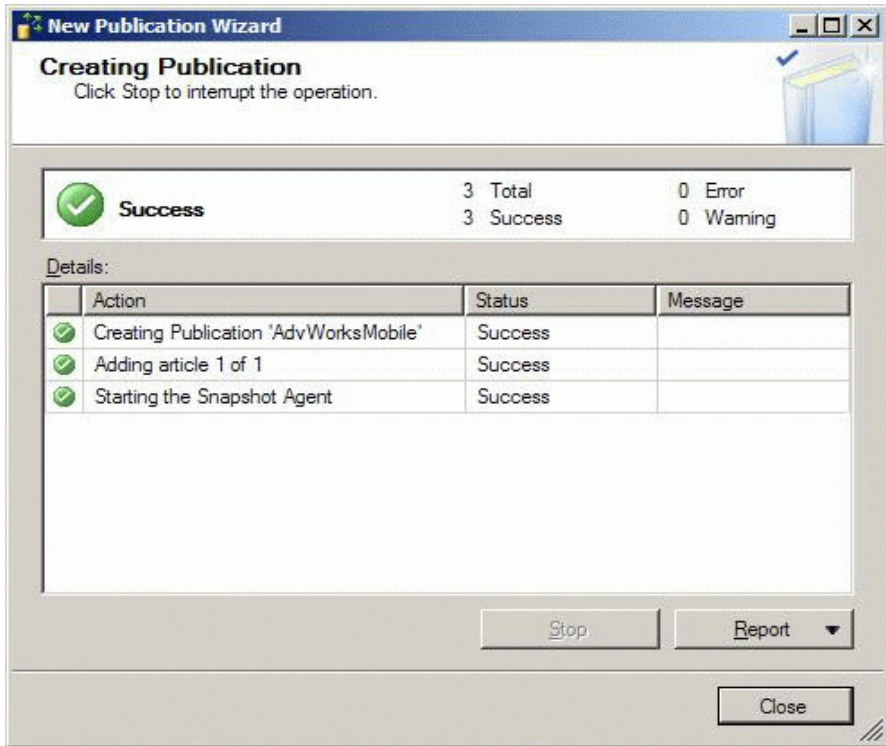
Tasks	Detailed Steps
	 Figure 37. The Articles page <i>Note: Additional tables may appear in the Objects to publish list than those in Figure 37.</i> aa. On the Article Issues page, click Next when the Uniqueidentifier columns will be added to tables issue appears in the list, as shown in Figure 38.

Tasks	Detailed Steps
	Figure 38. The Article Issues page bb. On the Filter Table Rows page, click Next to indicate that no row filtering is necessary, as shown in Figure 39. Figure 39. The Filter Table Rows page

Tasks	Detailed Steps
	cc. On the Snapshot Agent page, clear the Schedule the Snapshot Agent to run at the following times option, and be sure that Create a snapshot immediately is selected, as shown in Figure 40. Click Next.  Figure 40. The Snapshot Agent page dd. Click the Security Settings button to display the Snapshot Agent Security dialog box. ee. Select the Run under the SQL Server Agent service account option, and then click OK, as shown in Figure 41. <i>Note: Step cc is not a recommended security practice, but for the purpose of simplicity in this lab, use this setting.</i>

Tasks	Detailed Steps
	Figure 41. The Snapshot Agent Security dialog box ff. On the Agent Security page, click Next, as shown in Figure 42. Figure 42. The Agent Security page

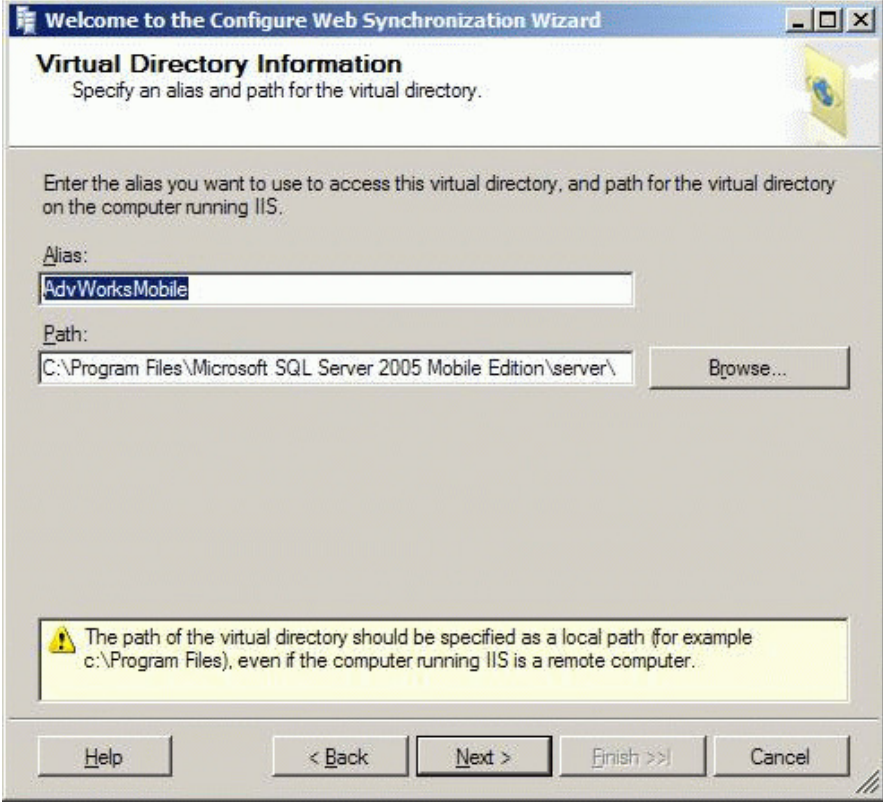
Tasks	Detailed Steps
	gg. On the Wizard Actions page, be sure that Create the publication is selected, and be sure that the Generate a script file with steps to create the publication option is clear, and then click Next, as shown in Figure 43. Figure 43. The Wizard Actions page hh. On the Complete the Wizard page, type AdvWorksMobile as the Publication name, and then click Finish, as shown in Figure 44.

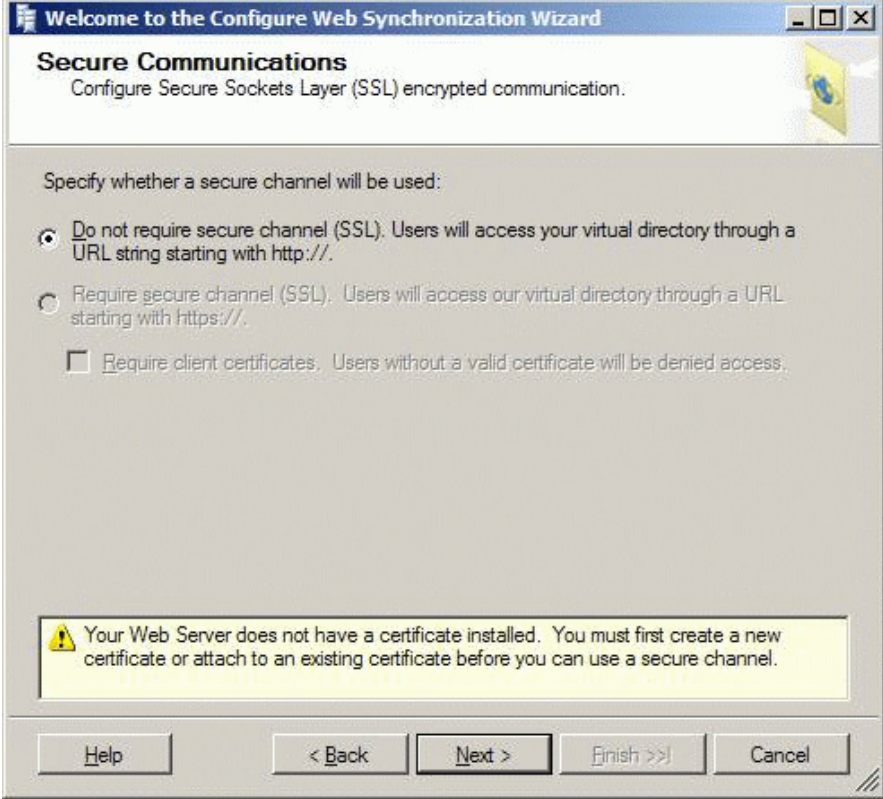
Tasks	Detailed Steps
	 Figure 44. The Complete the Wizard page ii. Wait for the publication wizard to finish successfully, and then click Close, as shown in Figure 45.  Figure 45. The New Publication Wizard successfully finishes

Tasks	Detailed Steps
2. Configure merge replication Web synchronization for the publication	a. In the SQL Server Management Studio Object Explorer pane, expand Local Publications to reveal the newly created publication. b. Right-click the new [AdventureWorks]: AdvWorksMobile publication, and then click Configure Web Synchronization. <i>Note: The Configure Web Synchronization Wizard appears, as shown in Figure 46.</i>  Figure 46. The Welcome page for the Configure Web Synchronization Wizard c. Click Next. d. On the Subscriber Type page, select SQL Server Mobile Edition, and then click Next, as shown in Figure 47.

Tasks	Detailed Steps
	Figure 47. The Subscriber Type page e. On the Web Server page, be sure your computer name appears in the Enter the name of the computer running IIS box. f. Select Create a new virtual directory. g. Under Select the Web site in which to create the new virtual directory, expand the local computer, expand Web Sites, and then select Default Web Site, as shown in Figure 48. Click Next.

Tasks	Detailed Steps
	Figure 48. The Web Server page h. On the Virtual Directory Information page in the Alias box, type AdvWorksMobile for the virtual directory that will be created, and then click Next, as shown in Figure 49.

Tasks	Detailed Steps
	 Figure 49. The Virtual Directory Information page - If you are prompted with a message that says The folder does not exist, click Yes to create the folder. - If you are prompted with a message that says This folder does not contain a copy of the SQL Mobile Server Agent, click Yes to copy and register the agent. - On the Secure Communications page, leave the Do not require secure channel (SSL) option selected, and then click Next, as shown in Figure 50.

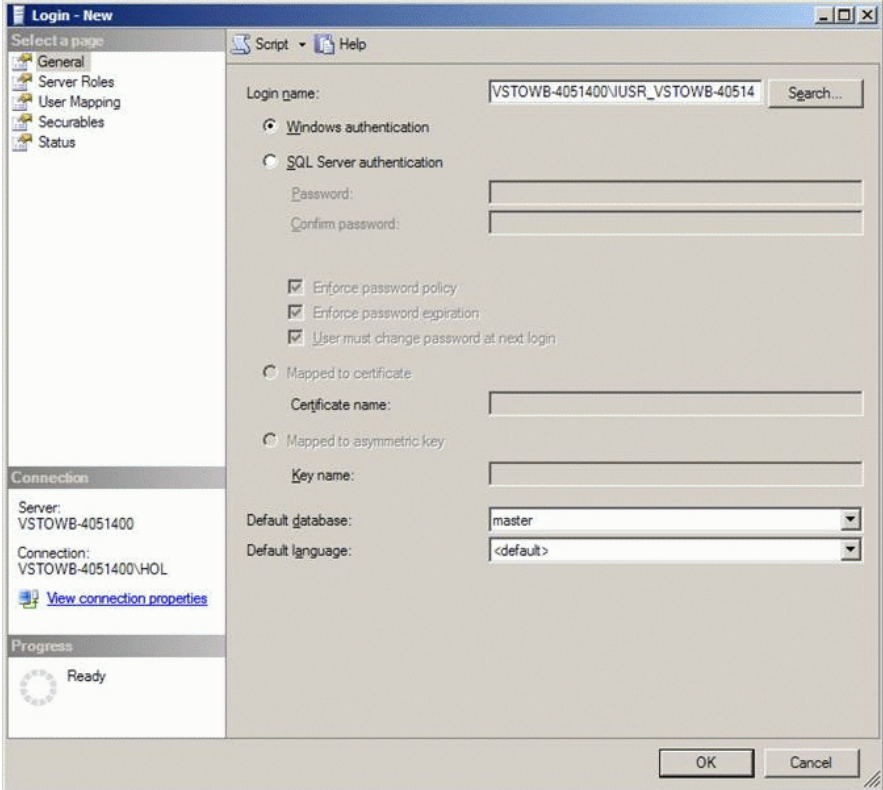
Tasks	Detailed Steps
	 Figure 50. The Secure Communications page I. On the Client Authentication page, select the Clients will connect anonymously option, and then click Next, as shown in Figure 51.

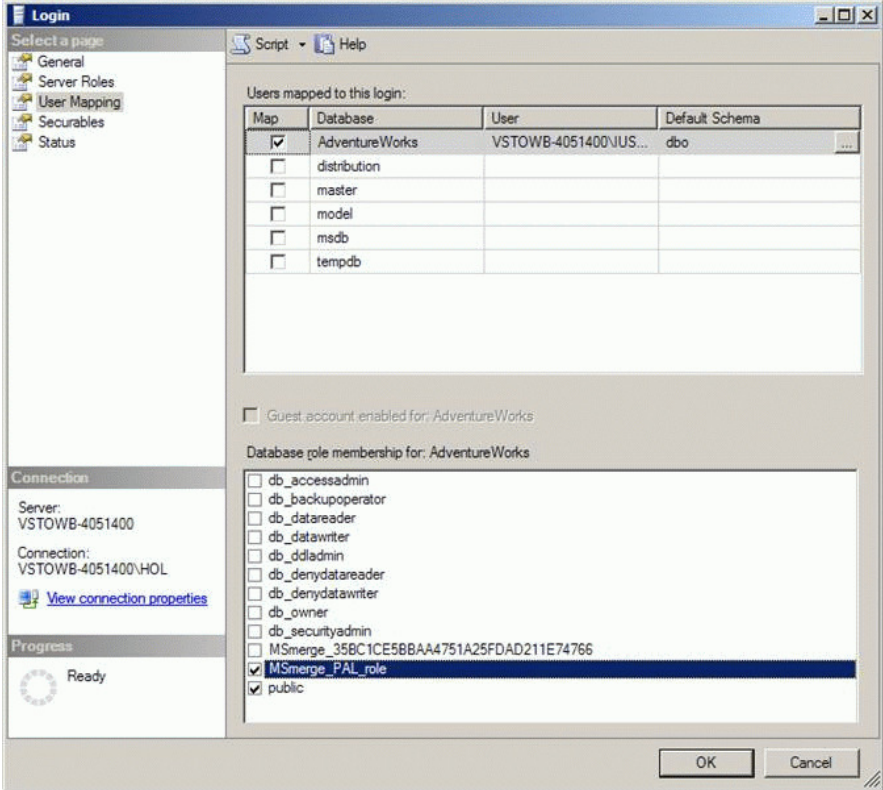
Tasks	Detailed Steps
	Figure 51. The Client Authentication page m. On the Anonymous Access page, leave the default values, and then click Next, as shown in Figure 52.

Tasks	Detailed Steps
	Figure 52. The Anonymous Access page n. On the Snapshot Share Access page in the Share box, type \\VSTOWB-4051400\\repldata. The share name, repldata, corresponds to the merge distribution share that you created in the previous task. Click Next.

Tasks	Detailed Steps
	Figure 53. The Snapshot Share Access page o. On the Complete the Wizard page, click Finish to complete the configuration. p. After the Configure Web Synchronization Wizard completes successfully, click Close, as shown in Figure 54. Figure 54. The Configure Web Synchronization Wizard successfully finishes

Tasks	Detailed Steps
	Note: Because the publication's virtual directory will be accessed anonymously, you need to allow your computer's anonymous Internet user account to access the SQL Server publication database. Note: The following task is not necessarily a recommended security practice, but it meets the demonstration purposes of this lab. In a production system, you would use some form of authentication to identify the users who access the publication database based on the security requirements of your system.
3. Permit anonymous Internet user to access the SQL Server publication	a. In the SQL Server Management Studio Object Explorer pane, right-click the Security folder, and then click New Login. b. In the Login - New window, click the Search button. c. In the Select User or Group dialog box, click the Advanced button. d. Click the Find Now button. Note: Scroll through the list of names, and then select the name that starts with IUSR_ and contains VSTOWB-4051400, as shown in Figure 55. The screenshot shows the 'Select User or Group' dialog box. At the top, there are two dropdown menus: 'Select this object type:' with 'User or Built-in security principal' selected, and 'From this location:' with 'VSTOWB-4051400' selected. Below these are 'Object Types...' and 'Locations...' buttons. The 'Common Queries' section has two dropdown menus: 'Name:' with 'Starts with' selected, and 'Description:' with 'Starts with' selected. There are checkboxes for 'Disabled accounts' and 'Non expiring password', both unchecked. A 'Days since last logon:' dropdown is set to 'All'. On the right side, there are 'Columns...', 'Find Now', and 'Stop' buttons. At the bottom, there are 'OK' and 'Cancel' buttons. A list of users is displayed at the bottom, with 'IUSR_VSTOWB-4051400' selected. The list includes: INTERACTIVE, IUSR_VSTOWB-4051400, IWAM_VSTOWB-4051400, LOCAL SERVICE, NETWORK, NETWORK SERVICE, REMOTE INTERACTIVE LOGON, SERVICE, SUPPORT_388945a0, SYSTEM, and TERMINAL SERVER USER. Figure 55. The Select User or Group dialog box e. Click OK. f. Click OK. The name of the user that you selected in Step d is displayed in the Login name box, as shown in Figure 56.

Tasks	Detailed Steps
	 Figure 56. The Login name box is populated with your selection g. Under Select a page, select User Mapping. h. Under Users mapped to this login, select the Map column to the left of AdventureWorks. i. In the Database role membership box, be sure MSmerge_PAL_role and public are selected, as shown in Figure 57.

Tasks	Detailed Steps
	 Figure 57. The user mapping settings in the Login window j. Click OK. Note: Drag and move the Login window up if the OK button is not visible. k. In the Object Explorer pane, expand Replication Local Publications if it is not already expanded. l. Right-click the [AdventureWorks]: AdvWorksMobile publication, and then click Properties. m. Select the Publication Access List page, and then click the Add button. n. In the Add Publication Access dialog box, select the anonymous Internet user (IUSR_*), and then click OK, as shown in Figure 58.

Tasks	Detailed Steps
	Figure 58. The Add Publication Access dialog box o. Confirm that the user has been added to the Publication access list, and then click OK, as shown in Figure 59. <i>Note: Drag and move the window up if the OK button is not visible.</i> Figure 59. Verifying that a user has been added to the publication

Tasks	Detailed Steps
	access list <i>Note: Now that you have created a publication and configured the Web server and database for replication, you can programmatically subscribe to the publication and synchronize data in your application. Instead of connecting your application's data source to the SQL Mobile database file that you created manually in Exercise 1, you will now connect to a SQL Mobile database that will be created and populated from the AdvWorksMobile publication. Because the schema of the Vendor table that you created in the previous exercise is compatible with that of the Vendor table in the AdventureWorks database, you can simply swap the new database for the old one.</i>
4. Add synchronization features to your application	a. Start Visual Studio by clicking Start All Programs Microsoft Visual Studio 2005 Microsoft Visual Studio 2005. b. Click File Open Project/Solution. c. Browse to C:\Program Files\Windows Mobile Developer Samples\HOLs\HOL302_SQL_Mobile\Exercises\Exercise2\AdvWorksMobile, and then select AdvWorksMobile.sln. Click Open. This is a copy of the solution that you created in Exercise 1. d. In Solution Explorer, double-click VendorList.cs to open the form designer. e. Click the light blue area above the right soft key. The area changes to gray and contains the text Type Here. f. Click the same area again to turn it dark blue, so you can replace the Type Here text, as shown in Figure 60. Figure 60. Preparing to change the soft key's text g. Type the word Menu, and then press ENTER. This menu will provide multiple synchronization functions. h. Type Init Sync, and then press ENTER. This command will be used to create a synchronized copy of the vendor data and display it to the user. i. Type Sync, and then press ENTER. This command will be used to synchronize the database that has already been created via the Init Sync command. j. Verify that the menu editor now looks like Figure 61.

Tasks	Detailed Steps
	Figure 61. The commands for the right soft key k. Right-click the Init Sync command, and then click Properties. l. Change the (Name) property to menuInitSync. m. In the same way, change the Sync command's (Name) property to menuSync. n. Double-click the Init Sync command to create a click event handler for the command. The Code view opens. o. Return to the form designer, and then double-click the Sync command to create a click event handler for it. You will add code to the event handlers that you have just created in a few steps. p. Locate the using statements at the beginning of the file, and then add the following three using statements. <pre data-bbox="544 646 1398 737">using System.Data.SqlServerCe; using System.IO; using System.Reflection;</pre> q. Declare a class-level variable of type SqlCeReplication that is named _repl. A good place to put the following line of code is near the beginning of the VendorList class. <pre data-bbox="544 940 1398 968">SqlCeReplication _repl = new SqlCeReplication();</pre> r. Create a private read-only string property named DataFileName, as shown in the following code example. <pre data-bbox="544 1136 1398 1352">private string DataFileName { get { return Path.ChangeExtension(Assembly.GetExecutingAssembly().GetName().CodeBase, ".sdf"); } }</pre> <i>Note: This property generates and returns the name of the SQL Mobile database file that will be created and updated with subscription data. It uses the System.IO.Path.ChangeExtension() method and System.Reflection.Assembly.GetExecutingAssembly().GetName().CodeBase property to generate the path and name for the database file. The database file is stored in the same folder and has the same base file name as the application's assembly with an .sdf extension added.</i> s. Create another private read-only string property named DataSourceConnectionString to generate and return the data source connection string for the SQL Mobile database, as shown in the following code example. <pre data-bbox="544 1776 1398 1894">private string DataSourceConnectionString { get { return string.Format("Data Source = {0};", this.DataFileName); }</pre>

Tasks	Detailed Steps
	<pre data-bbox="544 195 1398 226">}</pre> Note: This code uses the DataFileName property that you just created and returns a string in the format expected for a connection string to a SQL Mobile database. t. Create a private bool method named Connect that conditionally connects to the SQL Mobile database, as shown in the following code example. <pre data-bbox="544 415 1398 793">private bool Connect() { if (File.Exists(this.DataFileName)) { this.vendorTableAdapter.Connection = new SqlConnection(this.DataSourceConnectionString); return true; } else return false; }</pre> Note: This code uses the DataFileName property that you just created and the System.IO.File class's Exists method to check if the SQL Mobile database file is available for connection. If it is available, the connection is made between the SQL Mobile database and the vendorTableAdapter table adapter and a value of true is returned; otherwise, false is returned. u. Create a private method named GetData that uses the vendorTableAdapter table adapter to fill the advWorksDataSet data set from the SQL Mobile database, as shown in the following code example. <pre data-bbox="544 1150 1398 1308">private void GetData() { this.vendorTableAdapter.Fill(this.advWorksDataSet.Vendor); }</pre> v. Likewise, create a private method named SaveData that uses the vendorTableAdapter table adapter to save the advWorksDataSet data set contents to the SQL Mobile database, as shown in the following code example. <pre data-bbox="544 1505 1398 1633">private void SaveData() { this.vendorTableAdapter.Update(advWorksDataSet); }</pre> w. Create a private method named InitReplication that sets up the _repl SqlCeReplication object, as shown in the following code example. <pre data-bbox="544 1799 1398 1890">private void InitReplication() { _repl.InternetUrl = @"http://vstowb-</pre>

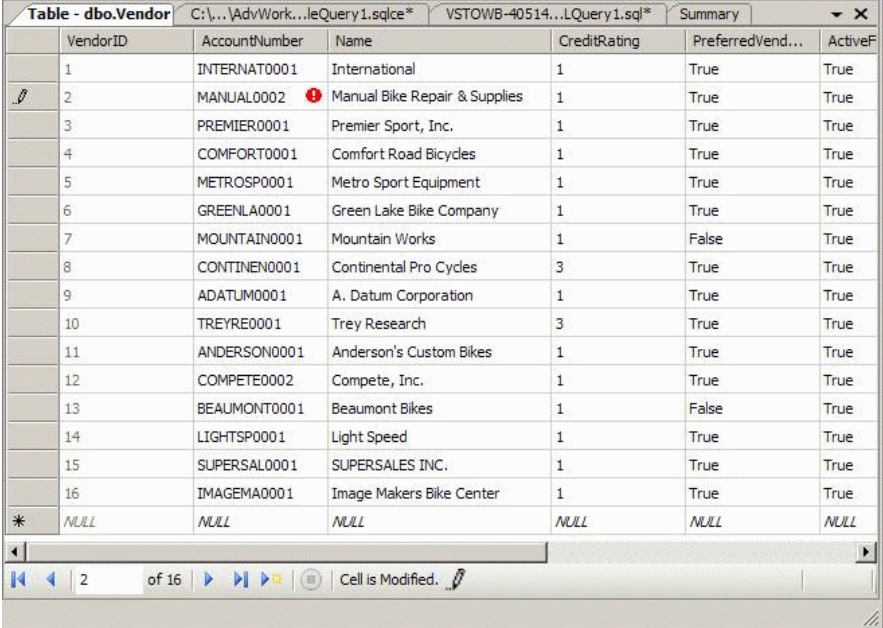
Tasks	Detailed Steps
	<pre data-bbox="545 197 1235 506"> 4051400/AdvWorksMobile/sqlcesa30.dll"; _repl.Publisher = @"VSTOWB-4051400"; _repl.PublisherDatabase = @"AdventureWorks"; _repl.PublisherSecurityMode = SecurityType.NTAuthentication; _repl.Publication = @"AdvWorksMobile"; _repl.Subscriber = @"AdvWorksMobile"; _repl.SubscriberConnectionString = this.DataSourceConnectionString; } </pre> Note: This code applies several property values to the _repl object to set it up for proper merge replication. The InternetUrl property should contain the name of the computer that has been set up as the merge publication Web share, and it needs to end with <i>sqlcesa30.dll</i>. Be sure to replace the italicized <i>your_computer_name</i> with your computer's actual name or IP address. Be aware that because this code will run on a device or emulator—and not your computer—you cannot use localhost here. Set the Publisher property to be the name of the database server that has been set up as the publisher, again replacing the italicized <i>your_computer's_name</i> with your computer's name or IP address. The name of the PublisherDatabase is the AdventureWorks database that was set up for replication. The PublisherSecurityMode is set to SecurityType.NTAuthentication, so the IUSR_* user is automatically logged in to the publishing database server. The Publication property is set to the name of the publication that you previously created: AdvWorksMobile. The Subscriber is named the same as the publication for simplicity. The SubscriberConnectionString uses the DataSourceConnectionString property that you previously created to connect to the device's SQL Mobile database. It is no longer desirable to load the locally stored data that you created manually in Exercise 1 when the application starts. x. Locate the VendorList_Load method, and then comment out or remove the line of code that fills the vendorTableAdapter table adapter, as shown in the following code example. <pre data-bbox="545 1415 1382 1476"> //this.vendorTableAdapter.Fill(this.advWorksDataSet.Vendor); </pre> y. In its place, add the following code that calls the InitReplication, Connect, and GetData methods that you previously created to retrieve any data that may exist. <pre data-bbox="602 1644 932 1734"> this.InitReplication(); if (this.Connect()) this.GetData(); </pre> z. Create a private method named SyncVendors that commits any pending data changes in the advWorksDataSet DataSet to the SQL Mobile database, synchronizes the data via merge replication, and rebinds the synchronized data to the advWorksDataSet DataSet, as shown in the following code example. This

Tasks	Detailed Steps
	should all be conditional upon the existence of the SQL Mobile database file. <pre data-bbox="544 268 1388 682"> private void SyncVendors() { if (File.Exists(this.DataFileName)) { this.SaveData(); _repl.Synchronize(); this.GetData(); } else MessageBox.Show("The subscription database has not yet been created. You must first select the Init Sync menu item."); } </pre> <i>Note: This code uses the DataFileName property that you previously created and the System.IO.File class's Exists method to determine if synchronization is valid at this point based on the existence of the SQL Mobile database file. Assuming the file exists, the code saves the data to the SQL Mobile database file by using the SaveData method that you created. Next, it calls the _repl object's Synchronize method to make changes in both the publishing SQL Server database and the subscribing SQL Mobile database. All modifications made in the local SQL Mobile database are applied to the SQL Server database, and any modifications that have been made to the SQL Server data are applied to the SQL Mobile database. Finally, it retrieves the updated data by using the GetData method that you created.</i> <i>If the database file does not yet exist, the code gives the user feedback and instructions to first initialize the synchronization.</i> aa. Locate the empty menuInitSync_Click method. Add the following code to create a subscription to the merge publication and synchronize the data for the first time. <pre data-bbox="544 1228 1388 1795"> bool okToContinue = !File.Exists(this.DataFileName); if (!okToContinue) okToContinue = MessageBox.Show("The SQL Mobile database already exists on this device and will be overwritten. Continue?", "Warning", MessageBoxButtons.YesNo, MessageBoxIcon.Exclamation, MessageBoxDefaultButton.Button1) == DialogResult.Yes; if (okToContinue) { File.Delete(this.DataFileName); _repl.AddSubscription(AddOption.CreateDatabase); this.Connect(); this.SyncVendors(); } </pre> <i>Note: Before the subscription is synchronized, this code checks for and deletes any</i>

Tasks	Detailed Steps
	existing <i>SQL Mobile</i> database file using the System.IO.File class's FileExists and Delete methods, first giving the user a chance to cancel the initialization if the user doesn't want to delete the existing subscription data. Next, the _repl.AddSubscription method is called with the CreateDatabaseAddOption enumeration value to create the <i>SQL Mobile</i> database file and set it up for synchronization. Now that the <i>SQL Mobile</i> database file exists, the Connect method that you previously created is used to connect to it. Finally, the code calls the SyncVendors method that you created to perform the first actual synchronization. During this first synchronization, the table is created in the <i>SQL Mobile</i> database with the same schema it has on the server. The data from the <i>SQL Server</i> table is downloaded into the table that is contained in the <i>SQL Mobile</i> database. At the end of the synchronization, the table in the <i>SQL Server</i> database and the table in the <i>SQL Mobile</i> database have the same data. bb. Locate the VendorList_Closing event handler, and then replace the call to the vendorTableAdapter.Update method with the following code. <pre> this.SaveData(); _repl.Dispose(); </pre> Note: This code uses the SaveData method that you created, which now has the code to update the <i>SQL Mobile</i> database. The code also properly disposes of the _repl object's resources after they are no longer needed. cc. Locate the menuSync_Click method, and then add a line of code that calls the SyncVendors method. This option allows the user to initiate synchronization. <pre> this.SyncVendors(); </pre> Note: To enable the emulator to connect to the merge publication Web site that is running on your local computer, you need to provide the emulator with network connectivity. The easiest way to provide network connectivity is to use ActiveSync 4.0 or later to connect to the emulator. This is a newly provided feature of Visual Studio 2005 and ActiveSync 4.0.
5. Establish connectivity between the emulator and the desktop computer	a. On the desktop computer, click Start All Programs Microsoft ActiveSync. b. In ActiveSync, click File Connection Settings. c. In the Connection Settings dialog box, select Allow connections to one of the following, if it is not already checked, and then select DMA in the corresponding list. d. Click OK. Note: ActiveSync is now able to connect to the emulators and physical devices. The next step is to connect and cradle the emulator by using the new Visual Studio 2005 Device Emulator Manager. e. In Visual Studio, click Tools Device Emulator Manager. f. In Device Emulator Manager, scroll to and select Windows Mobile 5.0 Pocket PC Emulator. You may have to expand the Windows Mobile 5.0 Pocket PC SDK node to locate Windows Mobile 5.0 Pocket PC Emulator. g. If there is not already a green arrow next to Windows Mobile 5.0 Pocket PC Emulator, click Actions Connect in Device Emulator Manager. A green arrow should appear next to Windows Mobile 5.0 Pocket PC Emulator after the

Tasks	Detailed Steps
	emulator starts up. h. With Windows Mobile 5.0 Pocket PC Emulator still selected, click Actions Cradle in Device Emulator Manager. <i>Note: ActiveSync should begin the connection process, which may take a minute or two.</i> i. If the Microsoft Office Outlook dialog box appears, click OK. j. If the Microsoft ActiveSync dialog box appears, click OK. k. On the Synchronizing Setup Wizard page, click Cancel. <i>Note: Clicking Cancel establishes connectivity between the emulator and the desktop computer without requiring that ActiveSync be configured to keep files and Microsoft Office Outlook® information on the emulator and desktop computer synchronized.</i> <i>Note: You are now ready to test the application.</i>
6. Test application features	a. In Visual Studio, click Debug Start Debugging. b. In the Deploy AdvWorksMobile dialog box that appears, be sure Windows Mobile 5.0 Pocket PC Emulator is selected, and then click Deploy. c. When the application appears in the emulator, click Menu. d. Click the Init Sync command. After going through the synchronization setup and actual synchronization, the emulator displays the updated vendor list, as shown in Figure 62.



Tasks	Detailed Steps
	Figure 62. The updated vendor list in the emulator - Click the first vendor row (with a Vendor ID of 1) to display the summary screen. - Click the Edit command, and then wait for the Edit screen to appear. - Change some of the data values, and then click ok to save your data changes. - Verify that your changes are reflected in the DataGrid. - Click New. - When the Edit screen appears, fill in the fields with new data, and then click ok. - Verify that your new record appears in the DataGrid. - Return to SQL Server Management Studio, and then connect to your computer's SQL Server database engine if it is not already connected. - In the Object Explorer pane, expand Databases AdventureWorks Tables. - Right-click the Vendor table (it may have a prefix), and then click Open Table. - Browse to the second vendor (with a VendorID of 2), and then change the AccountNumber and Name to new values, like those in Figure 63.  Figure 63. Changing the second vendor's account number and name - Use the mouse or arrow keys to browse to another row in the table to save your changes. The editing pencil icon disappears from the left column. - Browse to the new row (with an asterisk in the left column), and then add a new record. You don't need to add values for the VendorID column or any columns beyond and including the ModifiedDate column. <i>Note: The merge replication procedure created the additional rowguid column to identify records for replication. The column's value is generated automatically.</i> - Browse to another row to save your new record. - In the application on the emulator, click Menu Sync to synchronize the data in the SQL Server and SQL Mobile Vendor tables. Be sure you click Sync this time, and not Init Sync. Selecting Init Sync would delete and recreate the SQL Mobile database rather than synchronizing it with the SQL Server database.

Tasks	Detailed Steps
	t. After the synchronization is complete, view the list of records in the device application's DataGrid. Verify that the changes you made on the device and on the server by using SQL Server Management Studio are reflected on the device. u. On the vendor table in SQL Server Management Studio, click Query Designer Execute SQL to refresh the data, and then verify that the changes you made in both places are reflected on the server as well. v. On the emulator, click ok to close the application. w. Close Visual Studio 2005. <i>Note: In this exercise, you have learned how to use merge replication to synchronize data between a SQL Mobile database and a backend SQL Server database, retaining a fully functional mobile application that only needs to connect to a network for synchronization.</i>

Exercise 3

Synchronizing Data Between Any Backend Database and SQL Mobile Database Using a Web Service

Scenario

In this exercise, you will modify your application to synchronize its SQL Mobile data with any backend data store using a Web service.

Note: For the purposes of this exercise, the Web service synchronization will only be one way—from the SQL Mobile database to the SQL Server database. A more full-featured implementation would synchronize data both ways, but the purpose of this exercise is to demonstrate how simple it is to use a Web service as the mechanism for transmitting data from a mobile device to a backend data store.

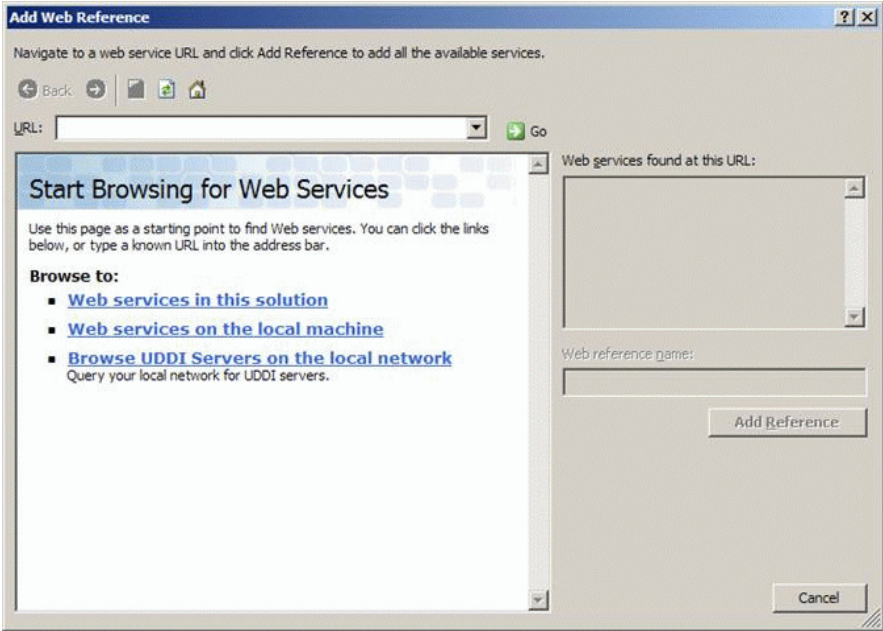
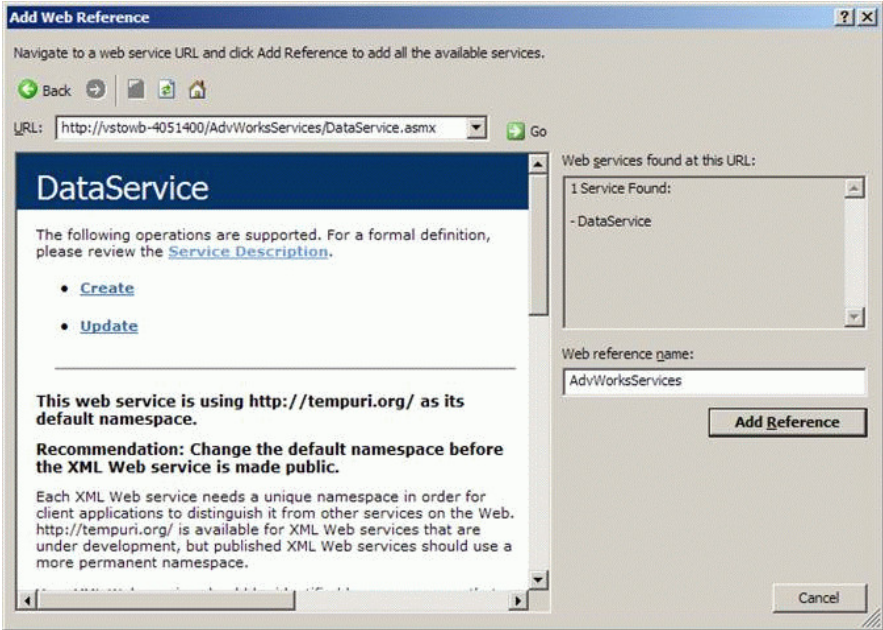
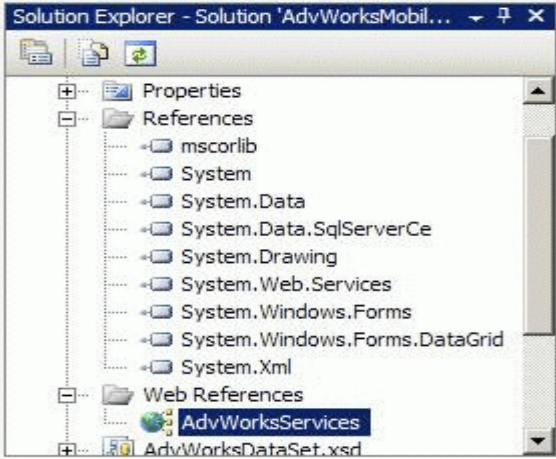
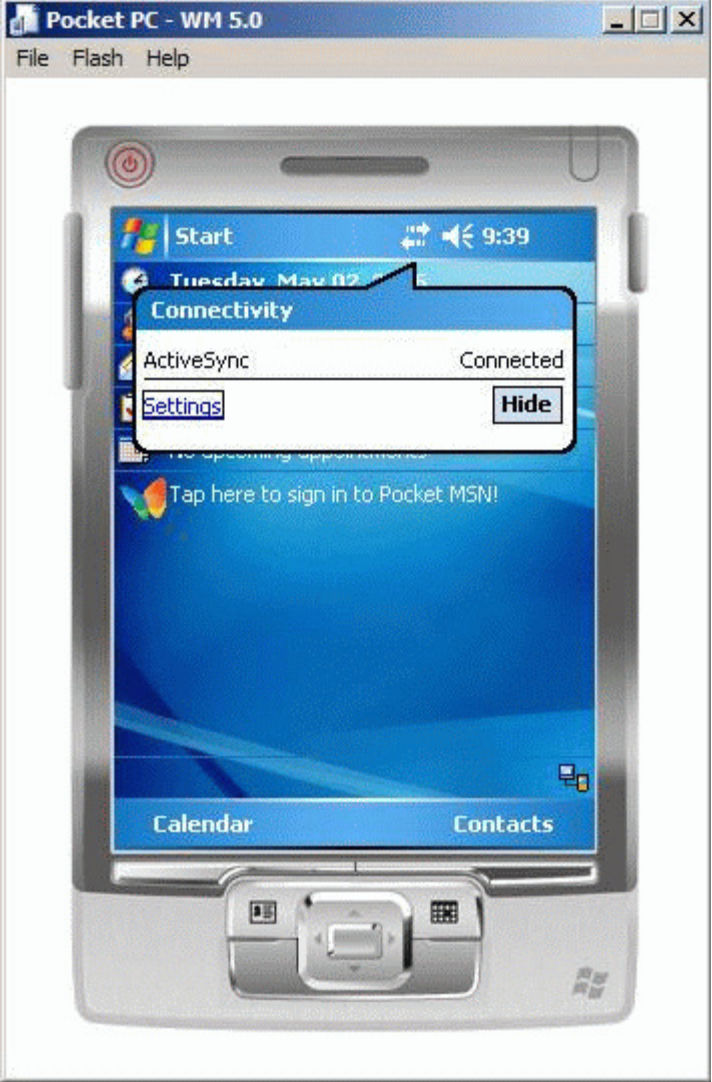
Tasks	Detailed Steps
1. Add Web service synchronization features to your application	a. Start Visual Studio by clicking Start All Programs Microsoft Visual Studio 2005 Microsoft Visual Studio 2005. b. Click File Open Project/Solution. c. Browse to C:\Program Files\Windows Mobile Developer Samples\HOLs\HOL302_SQL_Mobile\Exercises\Exercise3\AdvWorksMobile, and then select AdvWorksMobile.sln. Click Open. This is a copy of the solution that you created in Exercise 2. d. In Solution Explorer, right-click References, and then click Add Web Reference. Note: The <i>Add Web Reference</i> dialog box appears, as shown in Figure 64. 

Figure 64. The Add Web Reference dialog box

Tasks	Detailed Steps
	e. In the URL box, type http://VSTOWB-4051400/AdvWorksServices/DataService.asmx, and then click Go. <i>Note: The Web service that appears is a simple service that was installed with the files for this HOL. As you will see, the Web service provides Create and Update methods that accept an AdvWorksDataSet.VendorDataTable DataTable whose schema is compatible with the AdvWorksDataSet.VendorDataTable in your mobile application. The Create method expects to receive any new records that the caller has added since the last synchronization, and the Update method expects to receive any records the caller has updated since the last synchronization.</i> f. In the Web reference name box, type AdvWorksServices, and then click Add Reference, as shown in Figure 65.  Figure 65. Providing a Web reference name g. Verify that the AdvWorksServices Web service appears in Solution Explorer under Web References, as shown in Figure 66.  Figure 66. Verifying the new Web reference is in Solution

Tasks	Detailed Steps
	Explorer h. In Solution Explorer, double-click VendorList.cs to open the form designer. i. Click Menu above the right soft key to reveal the commands. j. Click the gray Type Here text below the Sync command to turn it dark blue, so you can replace the default text, as shown in Figure 67. Figure 67. Replacing the default text k. Type Sync with WS, and then press ENTER. This command will be used to synchronize your data with a Web service. l. Verify that the menu editor now looks like Figure 68. Figure 68. The Sync with WS command is added to the menu m. Right-click the Sync with WS command, and then click Properties. n. Change the (Name) property to menuSyncWS. o. Double-click the Sync with WS command to create a click event handler. The Code view opens. p. In the menuSyncWS_Click event handler, add the following code. <pre>AdvWorkServices.DataService advWorksDataService = new AdvWorksMobile.AdvWorkServices.DataService(); DataTable addedRows = advWorksDataSet.Vendor.GetChanges(DataRowState.Added); if (addedRows.Rows.Count > 0) {</pre>

Tasks	Detailed Steps
	<pre> AdvWorksServices.AdvWorksDataSet.VendorDataTable addedRowsForService = new AdvWorksServices.AdvWorksDataSet.VendorDataTable(); addedRowsForService.Merge(addedRows); advWorksDataService.Create(addedRowsForService); } DataTable modifiedRows = advWorksDataSet.Vendor.GetChanges(DataRowState.Modified); if (modifiedRows.Rows.Count > 0) { AdvWorksServices.AdvWorksDataSet.VendorDataTable modifiedRowsForService = new AdvWorksServices.AdvWorksDataSet.VendorDataTable(); modifiedRowsForService.Merge(modifiedRows); advWorksDataService.Update(modifiedRowsForService); } this.SaveData(); </pre> Note: This code begins by creating an instance of the AdvWorksMobile.AdvWorksServices.DataService Web service proxy class that was generated when you referenced the Web service. Next, the code creates a DataTable that contains any newly added rows from the advWorksDataSet Vendor DataTable by calling the GetChanges method with the DataRowState.Added enumeration value. If any new rows exist, they're merged into an AdvWorksServices.AdvWorksDataSet.VendorDataTable DataTable and passed to the Web service's Create method. The code then similarly retrieves any modified records using the DataRowState.Modified enumeration value, and then passes them to the Web service's Update method. Finally, the code saves the additions and modifications to the SQL Mobile database file by using the SaveData method that you created. Note: Before you can test the application, you need to be sure the emulator can connect to the network in such a way that it can call the Web service.
2. Allow network connectivity to call the Web service	a. If the emulator is not already running, follow the steps in the "Establish connectivity between the emulator and the desktop computer" task in Exercise 2. b. On the emulator, click the Connectivity icon at the top of the screen. The Connectivity balloon appears. Figure 69 shows the Connectivity balloon pointing to the Connectivity icon.

Tasks	Detailed Steps
	 Figure 69. The Connectivity balloon points to the Connectivity icon - Click Settings, and then click the Connections icon. - Click the Advanced tab, and then click the Select Networks button. - Be sure that My ISP is selected in both drop-down lists, and then click ok. - Click ok twice, and then click the close button at the top right of the Settings screen. <i>Note: You are now ready to test the new application feature.</i>
3. Test application features	- In Visual Studio, click Debug Start Debugging. - In the Deploy AdvWorksMobile dialog box that appears, be sure Windows Mobile 5.0 Pocket PC Emulator is selected, and then click Deploy. <i>Note: If you skipped Exercise 2, you will need to be sure the emulator is connected and cradled as described in the "Establish connectivity between the emulator and the desktop computer" task.</i> - Use the application's user interface to edit and create new data in the SQL Mobile database as you did in the "Test application features" task in Exercise 2. <i>Note: If you skipped Exercise 2, you will need to use ActiveSync's Explore button to</i>

Tasks	Detailed Steps
	<i>locate \Program Files\AdvWorksMobile\Vendors.sdf on the emulator, and then rename it to AdvWorksMobile.sdf. After doing so, click ok to close the application on the emulator, and then restart this procedure at Step a.</i> d. Click Menu Sync with WS to synchronize your local changes with the backend data store via the Web service. e. Because the backend database in this exercise is the same SQL Server AdventureWorks database that you have been using, use the same SQL Server Management Studio techniques that you used in the "Test application features" task in Exercise 2 to verify that the data changes you made on the device are now present on the server. Note: <i>In this exercise, you modified your application to synchronize its SQL Mobile data with any backend data store using a Web service.</i>

Summary

In this lab, you performed the following exercises:

- Creating and using a SQL Mobile database in a Windows Mobile 5.0 application
- Synchronizing data between a SQL Server 2005 and SQL Mobile database
- Synchronizing data between any backend database and SQL Mobile database using a Web service

In this lab, you learned how to use SQL Mobile to synchronize data between a Windows Mobile–based device and a SQL Server 2005 backend database using merge replication. You set up and configured SQL Server 2005, IIS, and a .NET Compact Framework application. You also learned how to synchronize SQL Mobile data with any backend data store using a Web service. These techniques have prepared you to easily build a mobile application that keeps mission-critical data in synchronization with a backend database.