Oracle® Database Lite

Getting Started Guide

10*g* (10.3.0)

B28924-01

April 2007

ORACLE®

Oracle Database Lite Getting Started Guide 10*g* (10.3.0)

B28924-01

Contents

2 Third Party Licensing in Oracle Database Lite

3 Requirements Before Installation or Development

4 Installation of Oracle Database Lite

5 Installing Your Mobile Client

6 Upgrade Oracle Database Lite

7 Quick Start for Oracle Database Lite

Index

Preface

This preface introduces you to the *Oracle Database Lite Getting Started Guide*, discussing the intended audience, documentation accessibility, structure, and conventions of this document.

Intended Audience

This manual is intended for users, who are new to the product, and administrators who are installing or upgrading Oracle Database Lite.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible, with good usability, to the disabled community. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Accessibility standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For more information, visit the Oracle Accessibility Program Web site at

`http://www.oracle.com/accessibility/`

Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

TTY Access to Oracle Support Services

Oracle provides dedicated Text Telephone (TTY) access to Oracle Support Services within the United States of America 24 hours a day, seven days a week. For TTY support, call 800.446.2398.

Send Us Your Comments

Oracle welcomes your comments and suggestions on the quality and usefulness of this publication. Your input is an important part of the information used for revision.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- What features did you like most about this manual?

If you find any errors or have any other suggestions for improvement, please indicate the title and part number of the documentation and the chapter, section, and page number (if available). You can send comments to us in the following ways:

- Electronic mail: olitedoc_us@oracle.com
- FAX: (650) 506-7355. Attn: Oracle Database Lite
- Postal service:

  Oracle Corporation
  Oracle Database Lite Documentation
  500 Oracle Parkway, Mailstop 1op2
  Redwood Shores, CA 94065
  U.S.A.

If you would like a reply, please give your name, address, telephone number, and electronic mail address (optional).

If you have problems with the software, please contact your local Oracle Support Services.

Related Documents

Since both the standalone Mobile Server and the Mobile Server over OracleAS use the Oracle Containers for J2EE (OC4J), the following manuals can be used as reference when configuring your application server:

- *Oracle Containers for J2EE Standalone User's Guide*
- *Oracle Containers for J2EE User's Guide*
- *Oracle Containers for J2EE Security Guide*

Conventions

The following conventions are also used in this manual:

Convention	Meaning
. . .	Vertical ellipsis points in an example mean that information not directly related to the example has been omitted.
. . .	Horizontal ellipsis points in statements or commands mean that parts of the statement or command not directly related to the example have been omitted

Convention	Meaning
boldface text	Boldface type in text indicates a term defined in the text, the glossary, or in both locations.
< >	Angle brackets enclose user-supplied names.
[ ]	Brackets enclose optional clauses from which you can choose one or none.

1

Oracle Database Lite Concepts

The following sections provide an introduction to Oracle Database Lite and its components:

1.1 Overview of Oracle Database Lite

A common misconception of Oracle Database Lite is that it is just a simplified small-scale database designed to run on laptops, handhelds, cell phones, and so on. While you can use it in this manner (see Section 1.2.1, "Embedded Application in Single Process"), this is only one model for using the product. Instead, Oracle Database Lite provides a complete mobile infrastructure designed to run enterprise database applications within a world constantly on the go. Oracle Database Lite provides the infrastructure that makes the enterprise application and data store available even when communications to the enterprise itself are not available or reliable. Oracle Database Lite brings the applications that were once limited to the base office and deploys these applications out into the world where they are most needed.

Oracle Database Lite can remove some of the manual processes performed out in the field. In the past, you may have manually written down the information in the field and then manually entered the data in the enterprise database once you returned to the corporate environment. With Oracle Database Lite, you can manually capture the data once—in the field by entering the data into a client device. Then, this data is synchronized up to the enterprise without returning to the office to manually enter data. This removes the loss of productivity due to manual processes and sends the data immediately to the enterprise where it belongs. In addition, data can flow bi-directionally. If you need information at the remote site that has been updated at the office, this data is brought down to the client device during synchronization.

1.1.1 What Is A Mobile Architecture?

The mobile architecture completes the enterprise system by merging the enterprise infrastructure with every remote aspect of the organization. Previously, the remote location was missing from the enterprise design. A mobile architecture contains the remote application, the remote data store, and the remote rules of the business. The Oracle Database Lite mobile infrastructure is responsible for connecting and synchronizing applications, associated data, and business rules with the applications, data store, and business rules of the enterprise.

There are several ways you can use and implement Oracle Database Lite. See Section 1.2, "Execution Models for Oracle Lite Database" for more details.

1.1.2 What Are The Benefits Of A Mobile Architecture?

A mobile architecture with the proper design, proper security components, and proper implementation saves money. Normally, you manually capturing data on site and then, when you get back to the office, manually enter the data into the office database. With proper design, your mobile application combines these steps into a single step of capturing the data at the remote location, which is then synchronized with the back-end database at the office.

This saves time and enables remote agents the capability of performing more tasks on site without returning to an office for the manual process of entering the captured data.

You can use the mobile architecture in several types of application environments, as follows:

- Mobile option—An application is created, where the user enters data on a client device and the data is synchronized with a back-end Oracle database. For example, if you have a sales force, each sales person retrieves only his/her data on the client device. Any modifications made on either the client device by the sales person in regards to his/her accounts or modified on the server by the office can be synchronized.
- Embedded software option—An application may need an independent small database to exist solely for the application's use. No synchronization of data with a back-end database is necessary. For example, if you have an individual accounting application, it may need a small embedded database to store the data for the individual accounting data.
- Embedded hardware option—A hardware unit may need an embedded database to facilitate gathering information, which is then synchronized with a back-end database for the office to evaluate what is happening with the remote hardware unit.

  For example, a vending machine can use the Oracle Database Lite infrastructure to maintain inventory, control the dispatching of technicians and restock personnel, gather marketing statistics, and so on.

  Another example is a system included in an automobile—known as an in-vehicle system. The mobile infrastructure tracks and communicates maintenance needs for the automobile to the automobile owner and service department. When maintenance needs are proactively found, the customer saves on repair costs, towing and expensive part replacement that may otherwise have occurred.

1.1.3 Why Use Oracle Database Lite?

Oracle Database Lite provides a complete mobile infrastructure suitable for almost any enterprise demands using the following:

- The Mobile repository resides in the back-end enterprise database, which links the enterprise data with the mobile data.
- The Mobile Server is a Web-based tier that integrates with OracleAS and executes on top of the Oracle Containers for J2EE (OC4J). This accesses remote locations through different types of wireless or wired connectivity. It facilitates the major functions for the Mobile option, such as synchronization, application management, device management, and so on.
- The Mobile client uses a client database, called the Oracle Lite database (ODB file), and the means for deploying applications developed using the most popular languages, such as C#, VB, VB.Net, Java, and C/C++. These clients can be executed on most any device from a cell phone, to a personal digital assistant (PDA), Tablet PC, Laptop, and so on.

1.2 Execution Models for Oracle Lite Database

Oracle Database Lite facilitates the development, deployment, and management of Mobile database applications for a large number of mobile users. A Mobile application is an application that can run on mobile devices without requiring constant connectivity to the server. The application requires a small, local database on the mobile device, whose content is a subset of data that is stored in the enterprise data server. This database is known as the Oracle Lite database (ODB file). Modifications made to the local database by the application are occasionally reconciled with the back-end server data. The technology used for reconciling changes between the mobile database and the enterprise database is known as data synchronization.

How your application integrates with the Oracle Lite database depends on the design of your application, as described in the following sections:

1. Do you want to use the small footprint Oracle Lite database to store data embedded in a single application? If you have an application that needs a database with a small footprint for client devices, but you do not need to have the client data synchronized with a back-end Oracle database, then the embedded option is for you. See Section 1.2.1, "Embedded Application in Single Process" for more information.

   Other embedded application models are as follows:

   - Do you want to have multiple applications access the same client database on a single client device? See Section 1.2.2, "Multiple Processes Accessing the Same Database" for more information.
   - Do you want your application to access the client database remotely? See Section 1.2.3, "Multiple Embedded Application Clients Accessing Remote Database" for more information.

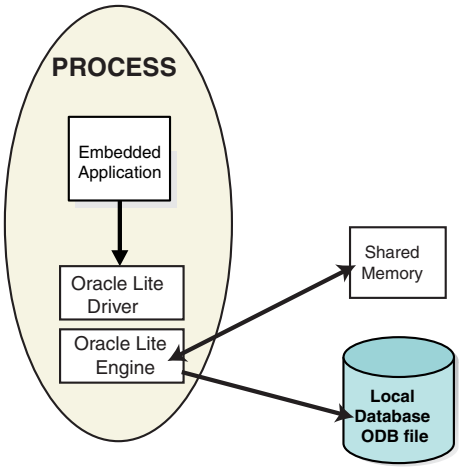
2. Do you want to use the Oracle Lite database to store changes that will be synchronized with a back-end Oracle database? You can design an application where the data is stored in a back-end Oracle database, but only the data that the user needs to see or update is downloaded to the client device. When either side modifies the data, it is synchronized between the client device and the back-end database. This is known as the Mobile client option. See Section 1.2.4, "Mobile Option for a Client in a Single Process" for more details.

Alternatively, you can have a more complex situation where you want your application to access the database remotely and include the ability to synchronize the client data. If so, then see Section 1.2.5, "Mobile Option for Multiple Clients Accessing Remote Database" for more information.

1.2.1 Embedded Application in Single Process

As demonstrated in Figure 1–1, if you chose to build a standalone application with the Oracle Lite database embedded in the application, then when the application is launched, the Oracle Lite database libraries are loaded into the same process as the application.

Figure 1–1 Embedded Application With ODB Libraries in Single Process

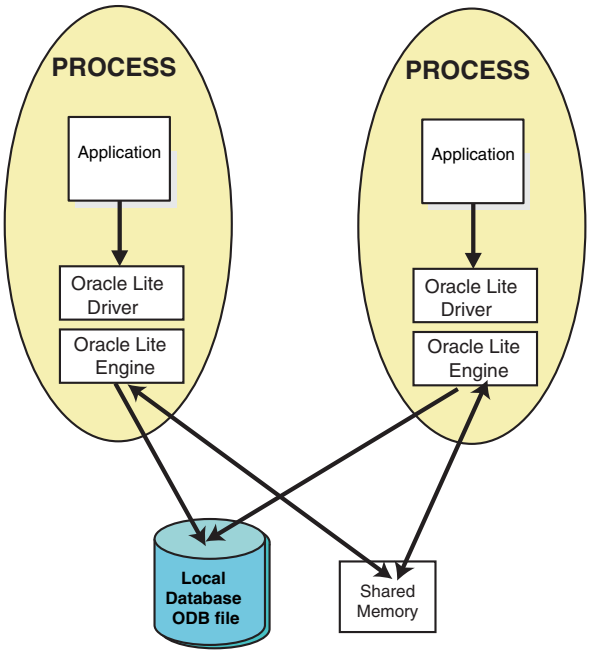


See Section 2.3, "Creating and Managing the Database in an Embedded Application" in the *Oracle Database Lite Developer's Guide* for more information on how to embed an Oracle Lite database into a standalone application.

1.2.2 Multiple Processes Accessing the Same Database

Figure 1–2 shows how you can configure multiple application processes to share the same database on the same machine. Thus, when each application is launched, each application exists in its own process and can access the same database independently. In this scenario, Oracle Database Lite libraries use shared memory to coordinate locking between both processes.

Figure 1–2 Applications in Multiple Processes Accessing Single Database



For details of how to package an embedded Oracle Lite database in your application, see Section 2.3, "Creating and Managing the Database in an Embedded Application" in the *Oracle Database Lite Developer's Guide*.

Note: This scenario is not available on WinCE.

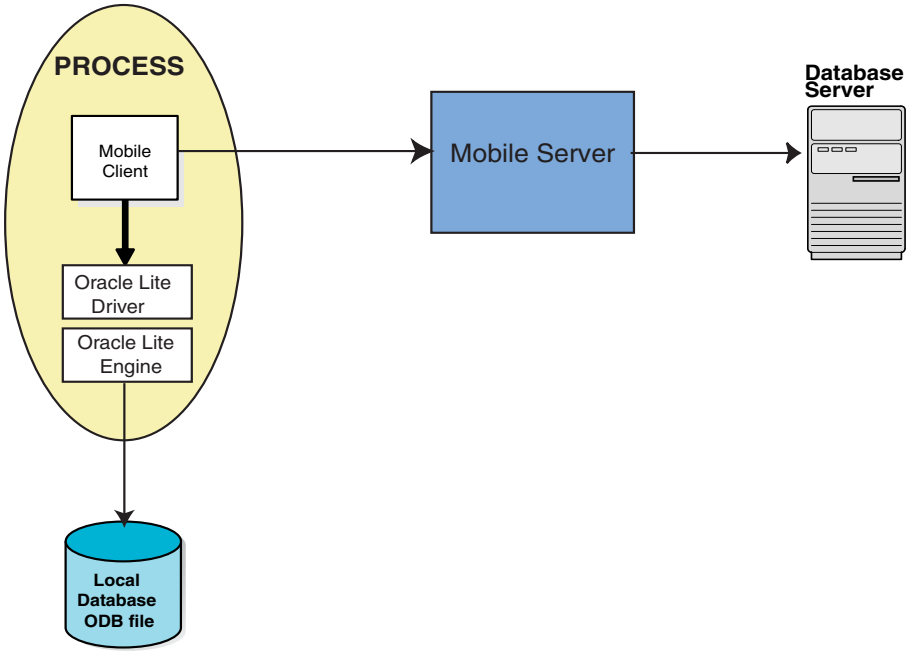
1.2.3 Multiple Embedded Application Clients Accessing Remote Database

If you are embedding a database into your application software, but you want the applications to be located on clients that are remote from the data within the Oracle Lite database, then use the client/server embedded approach with the multi-user service, as described in Section 2.5, "Oracle Database Lite Multi-User Service" in the *Oracle Database Lite Developer's Guide*.

1.2.4 Mobile Option for a Client in a Single Process

If you chose to install the Mobile client and synchronized your user on a single device, then when you launch your application, the Oracle Lite database libraries are loaded into the same process as your application. This scenario is demonstrated in Figure 1–3.

Figure 1–3 Diagram of Mobile Client and ODB Libraries in SIngle Process



For details of how to create a Mobile application using the Oracle Lite database, see Section 2.2, "Creating and Managing the Database for a Mobile Client" in the *Oracle Database Lite Developer's Guide*.

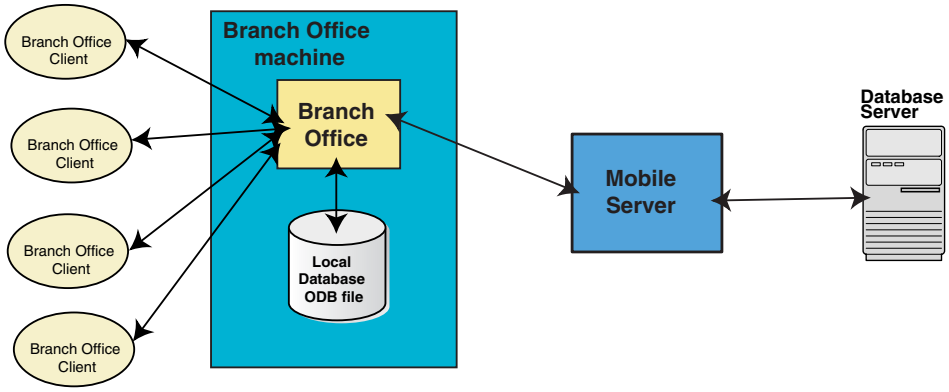
1.2.5 Mobile Option for Multiple Clients Accessing Remote Database

If you have several remote clients accessing the same data, you can use Branch Office to facilitate the remote applications. Figure 1–4 demonstrates how multiple remote Branch Office clients access the data through the Branch Office machine to the Mobile Server and finally accessing the back-end Oracle database.

The Branch Office machine contains the Branch Office executables and the local Oracle Lite database, which all clients access for their information. When a synchronization is requested, information is communicated between the Branch Office and the back-end database through the Mobile Server.

Note: Oracle Database Lite is not identical to the Oracle database; thus, it is not designed for large amounts of transferred data or a large number of concurrent transactions.

Figure 1–4 Using Branch Office for Managing Multiple Clients that Access a Remote Database



See Chapter 10, "Manage Your Branch Office" in the *Oracle Database Lite Administration and Deployment Guide* for more information.

1.3 Oracle Database Lite Application Models

If you are using the embedded model, as described in Section 1.2.1, "Embedded Application in Single Process", then you can create the tables and access the information as you would any RDBMS. However, if you want to use the Mobile option, you need to define the data in the snapshot.

The following sections describe how to define the data and subscribe users to this data.

- Section 1.3.1, "Publish-Subscribe Model for Mobile Users"
- Section 1.3.2, "Client Mobile Database Created on First Synchronization"

1.3.1 Publish-Subscribe Model for Mobile Users

Oracle Database Lite operates within a publish-subscribe model. We use the example of the magazine as an effective way to explain the publish-subscribe model. A magazine is created with specific data that would be of interest to readers, such as sports, hunting, automobiles, and so on. Readers request a subscription for the specific magazine they feel would be in their interest to read. Once this subscription is created only the magazines to which the reader has been subscribed are sent to the reader.

For Oracle Database Lite, the publication is the magazine, the publication items are the specific articles of data and the subscription is the granting of access to the publication for specific users. In the Oracle Database Lite application model, each application defines its data requirements using a publication. Data subsets, known as publications items, are created and added to a publication. Application files are also uploaded to the same publication. Once these publications are deployed to the Mobile Server, any user may be granted a subscription to the publication.

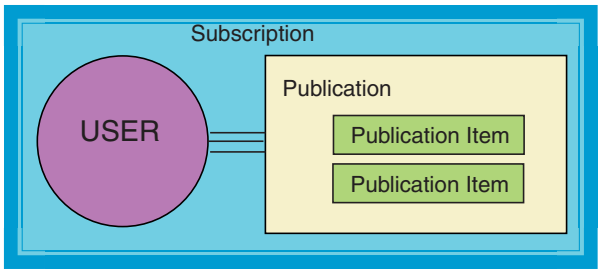
Technically, a publication is akin to a database schema and it contains one or more publication items. A publication item is like a parameterized view definition and defines a subset of data, using a SQL query with bind variables in it. These bind variables are called *subscription parameters* or *template variables*.

As shown in Figure 1–5, a subscription defines the relationship between a user and a publication. Once you subscribe to a particular publication, you begin to receive information associated with that publication. With a newspaper you receive the daily

paper or the Sunday paper, or both. With Oracle Database Lite you receive snapshots, and, depending on your subscription parameter values, those snapshots are partitioned with data tailored for you.

Subscription parameter values can be set by the administrator in order to tailor the snapshot data for each user.

Figure 1–5 Subscription Defines Relationship Between User and Publication



The subscription is the definition of how to retrieve data from the back-end database; the snapshot is the actual data that conforms to the definition within the subscription and which belongs to the user.

This process really forms a simple development cycle for mobile applications, as follows:

1. Create the publication and its publication items that contains the data subset for a particular application.
2. Grant users a subscription to a publication. This forms the specific dataset that is used on a Mobile client.
3. Develop and test the Mobile application to work with the specific data set.
4. Deploy the application to the Mobile Server and install it on the client.

Two of the more common questions and sources of confusion that comes up are what has to be done first:

1. Do you create the publication first or the publication items?

   It does not matter. You can create either the publication or the publication item first. Consider an article for a magazine. That article may have been written by a freelance author. The article exists before it belongs to any publication. The author submits this to two or three magazine publishers since it is relevant to the content they advertise. Two decide it is appropriate for the publication they are distributing currently while one does not include it since the content is not quite what their readers want.

2. Do you have to create a separate publication item for each publication?

   No, you can have one or more publication items in a publication.

> **Note:** You can create publications with the Mobile Database Workbench or the Java APIs.

The following sections describes other pertinent information for publication items:

- Section 1.3.1.1, "Defining the Weight and Conflict Resolution for Publication Items"

- Section 1.3.1.2, "Behavior and Requirements for Primary Keys, Foreign Keys and Not Null Fields in Publication Items"

1.3.1.1 Defining the Weight and Conflict Resolution for Publication Items

The following important aspects of the publication item should be taken into account when you are designing your application:

- Weight—The publication item weight is used to control the order in processing publication items, which avoids conflicts. Changes made on the client are processed according to weight in order to prevent conflicts, such as foreign key violations. The weight determines what tables are applied to the enterprise database first. For example, the `scott.emp` table has a foreign key constraint to the `scott.dept` table. If a new department number is added to the `dept` table and a new record utilizing the new department number were added to the `emp` table, then the transaction would be placed in the error queue if the new record utilizing the new department in the `emp` table was applied to the repository before the new department in the `dept` table was applied. To prevent the violation of the foreign key constraint on the enterprise server, you set the `dept` snapshot to a weight of 1 and the `emp` snapshot to a weight of 2, which applies all updates to the `dept` table prior to any updates to the `emp` table as the lower weight is always processed first.
- Conflict Resolution—In the same scenario, what if someone already updated the enterprise server with the new department number? This causes a conflict when the client attempts to synchronize with the new department that utilizes the same number. To handle this, conflict resolution may be set to either "client wins" or "server wins". If set to "server wins", then the setting on the server takes precedence to the setting on the client. The client transaction is sent to the error queue. However, if "client wins" is set, then the new department number from the client overrides the setting on the server.

1.3.1.2 Behavior and Requirements for Primary Keys, Foreign Keys and Not Null Fields in Publication Items

Only primary keys and not null fields are replicated down to the client. Publication items require a primary key field or, as in the case of a view, primary key hints.

If a foreign key needs to be applied to the client, then the script for the foreign key needs to be added to the publication, so that it will be executed when the client synchronizes the first time. You can set the script for the foreign key within either the MDW scripts section or the API.

Constraints are not the only type of script that may be executed on the client. The script could execute any valid SQL DDL statement on the client.

1.3.2 Client Mobile Database Created on First Synchronization

When a user synchronizes the Mobile client for the first time, the Mobile client creates an Oracle Lite database on the client machine for each subscription that is provisioned to the user. The Mobile client then creates a snapshot in this database for each publication item contained in the subscription, and populates it with data retrieved from the server database by running the SQL query (with all the variables bound) associated with the publication item. Once installed, Oracle Database Lite is transparent to the end user; it requires minimal tuning or administration.

As the user accesses and uses the application, changes made to the data in the Oracle Lite database are captured by the snapshots. When the connection to the Mobile Server is available, the changes can be synchronized with the Mobile Server.

See Section 1.4, "How Oracle Database Lite Synchronizes" for more details.

1.4 How Oracle Database Lite Synchronizes

When most people think of synchronizing data, they think of their Palm Pilot. When you hit the synchronization button for the Palm Pilot, any changes are added to the database of information on the Windows machine immediately. This is not the case for Oracle Database Lite, in that the synchrnization is used for multiple clients—rather than a single user. In order to accomodate a large number of concurrent users, the application tables on the back-end database cannot be locked by a single user. Thus, the synchronization process involves using queues to manage the information between the Mobile clients and the application tables in the database.

Oracle Database Lite uses a synchronization model that maintains data integrity between the Mobile Server and the Mobile client. In addition, the synchronization is asynchronous and that as a result, change propagation is not immediate. The benefit, however, is that the clients do not stay connected for long while the changes are being applied.

You can specify if the synchronization occurs automatically or by manual request. For more details, see Section 1.4.1, "Deciding on Automatic or Manual Synchronization".

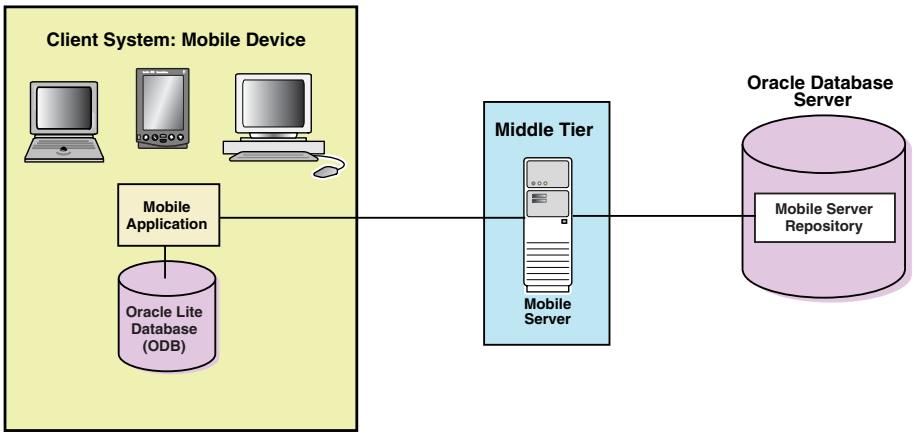
A simplified view of Mobile synchronization is as follows:

- On the client—The Mobile application communicates through the Mobile Sync Server with the Mobile Server and uploads the changes made in the client machine. It then downloads the changes for the client that are already prepared by the Mobile Server.
- On the Mobile Server—A background process called the Message Generator and Processor (MGP), which runs in the same tier as the Mobile Server, periodically collects all the uploaded changes from many Mobile users and then applies them to the server database. Next, MGP prepares changes that need to be sent to each Mobile user. This step is essential because the next time the Mobile user synchronizes with the Mobile Server, these changes can be downloaded to the client and applied to the client database.

Figure 1–6 illustrates the architecture for Oracle Database Lite applications.

Note: This section describes how the synchronization is performed across several components and enterprise tiers to complete successfully. For more details on each component, see Section 1.6, "Oracle Database Lite Components Involved in Synchronization".

Figure 1–6 Oracle Database Lite Architecture

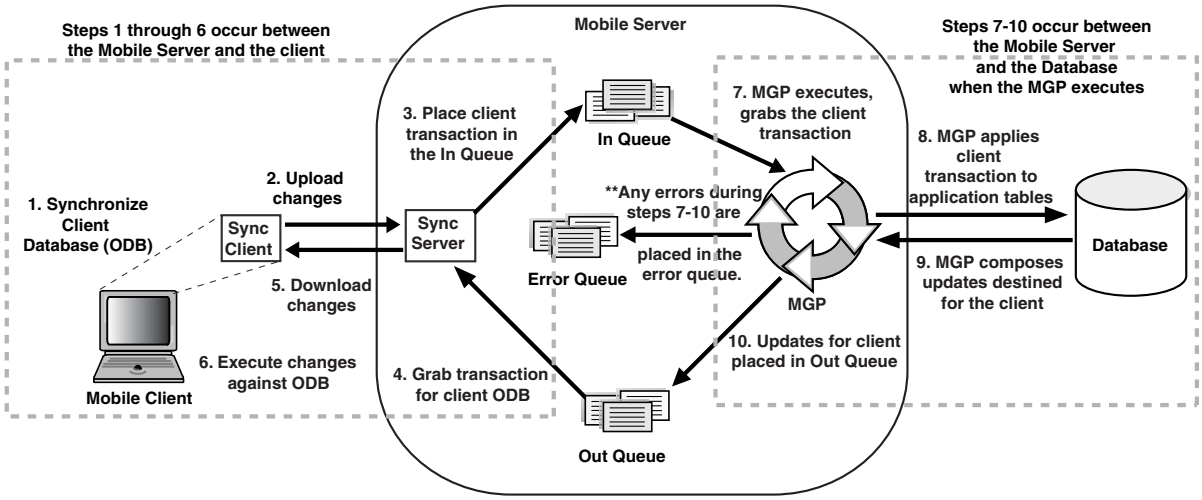


Note: Web-to-Go clients have one additional component, a light weight HTTP listener that is not shown in the diagram.

Oracle Database Lite uses the Mobile Server to replicate data between the Mobile clients with their client Oracle Lite databases (including those for OC4J, Web-to-Go, Win32, Windows CE, Symbian, and Linux platforms) and the application tables, which are stored on a back-end Oracle database.

Thus, the more detailed description of how synchronization is performed within the separate components of Oracle Database Lite is demonstrated by Figure 1–7.

Figure 1–7 Data Synchronization Architecture



1. A synchronization is initiated on the Mobile client either by the user or from automatic synchronization. Note that the Mobile client may be a Windows platform client or a PDA.
2. Mobile client software gathers all of the client changes into a transaction and the Sync Client uploads the transaction to the Sync Server on the Mobile Server.
3. Sync Server places the transaction into the In-Queue.

> **Note:** When packaging your application, you can specify if the transaction is to be applied at the same time as the synchronization. If you set this option, then the transaction is immediately applied to the application tables. However, note that this may not be scaleable and you should only do this if the application of the transaction immediately is important and you have enough resources to handle the load.

4. Sync Server gathers all transactions destined for the Mobile client from the Out-Queue.
5. Sync client downloads all changes for client Oracle Lite database.
6. Mobile client applies all changes for client Oracle Lite database. If this is the first synchronization, the Oracle Lite database is created.

> **Note:** For information on what Oracle Lite database (ODB) files are installed on the client, see Section 2.2, "Synchronizing or Executing Applications on the Mobile Client"in the *Oracle Database Lite Administration and Deployment Guide*.

7. All transactions uploaded by all Mobile clients are gathered by the MGP out of the In-Queue. The MGP executes independently and periodically based upon an interval specified in the Job Scheduler in the Mobile Server.
8. The MGP executes the apply phase by applying all transactions for the Mobile clients to their respective application tables to the back-end Oracle database. The MGP commits after processing each publication. If any conflicts occur during this phase, most are resolved by the MGP or by the conflict resolution rules. If the conflict cannot be resolved, the transaction is moved into the Error Queue. See Section 1.3.1.1, "Defining the Weight and Conflict Resolution for Publication Items" for more information.

> **Note:** The behavior of the apply/compose phase can be modified. See Section 5.1.1, "Defining Behavior of Apply/Compose Phase for Synchronization" in the *Oracle Database Lite Administration and Deployment Guide* for more information.

9. MGP executes the compose phase by gathering the client data into outgoing transactions for Mobile clients.
10. MGP places the composed data for Mobile clients into the Out-Queue, where the Sync Server downloads these updates to the client on the next client synchronization.

Overall, synchronization involves two parties: the Mobile client using the Sync Client/Server to upload and download changes and the MGP process interacting with the queues and the application tables to apply and compose transactions. These are displayed separately in the Data Synchronization section of the Mobile Manager.

1.4.1 Deciding on Automatic or Manual Synchronization

In the past, all that was available was manual synchronization. That is, a client manually requests a synchronization either through an application program executing an API or by a user manually pushing the Sync button.

Now, you can specify that synchronization happens automatically or only when manually requested.

- Manual Synchronization may be initiated, as follows:
  - The user initiates the Oracle Database Lite Mobile Synchronization (msync) application directly.
  - The application programmatically invokes the Mobile Synchronization API.
  - Oracle Database Lite has a Web-based application model, known as Web-to-Go. For this type of application, the synchronization option can be defined from the Web-to-Go workspace to synchronize the data. See Section 1.5.3, "Supported Languages for Building Mobile Applications" for more information.
- Automatic Synchronization—You can configure for synchronization to automatically occur under specific circumstances and conditions. When these conditions are met, then Oracle Database Lite automatically performs the synchronization for you without locking your database, so you can continue to work while the synchronization happens in the background. This way, synchronization can happen seamlessly without the client's knowledge.

For example, you may choose to enable automatic synchronization for the following scenarios:

- If you have a user who changes data on their handheld device, but does not sync as often as you would prefer.
- If you have multiple users who all sync at the same time and overload your system.

These are just a few examples of how automatic synchronization can make managing your data easier, be more timely, and occur at the moment you need it to be uploaded.

Synchronization is closely tied to how you define the snapshot for your application. See Section 1.3.1, "Publish-Subscribe Model for Mobile Users" for a description of a snapshot and its components. One of the components is a publication item. If you want automatic synchronization, you define it at the publication item level.

Note: When a manual synchronization is requested by the client, ALL publication items are synchronized at that time—including those defined as manual and automatic synchronization. However, if an automatic synchronization is currently executing, the manual synchronization request is delayed until the automatic synchronization completes. Alternatively, you can stop the automatic synchronization to allow the manual synchronization to occur. If you choose to do this, then after the manual synchronization is finished, re-start the automatic synchronization.

The differences between the two types of synchronization are as follows:

Table 1–1 Difference Between Automatic and Manual Synchronization

	Manual Synchronization	Automatic Synchronization
Initiation	After the snapshot is set up, you can initiate either by the user initiating mSync or by an application invoking one of the synchronization APIs.	All of the set up for automatic synchronization is configured. Once configured, it happens automatically, so there is no synchronization API. Configuration for automatic synchronization can be defined when you create the publication item, publication or the platform.
Controlling synchronization	Synchronization occurs exactly when the user/application requests it.	Synchronization occurs without the user being aware of it occuring. You may have to manage synchronization through the Sync Control API if you have publications that contain both manual and automatic synchronization publication items.

For more information on Automatic Synchronization, see Section 3.2, "Automatic Synchronization Overview" in the *Oracle Database Lite Developer's Guide*.

1.4.2 Deciding on Synchronization Refresh Option

How or when data changes are applied to either the Mobile Server or the Mobile client depends upon the synchronization refresh option at the publication item level. Synchronization refresh options may ease the cost burden for resources, such as wireless connectivity, bandwidth and network availability, personnel loss of time during the synchronization process, and so on.

Oracle Database Lite employs synchronization refresh options that may be utilized to synchronize data between the Oracle enterprise database and the Mobile client. With the following Oracle Database Lite refresh options, you can maintain data accuracy and integrity between the database and Mobile client:

- Section 1.4.2.1, "Fast Refresh"
- Section 1.4.2.2, "Complete Refresh"
- Section 1.4.2.3, "Queue-Based Refresh"
- Section 1.4.2.4, "Forced Refresh"

1.4.2.1 Fast Refresh

The most common method of synchronization is a fast refresh publication item where changes are uploaded and downloaded by the client. Meanwhile, the MGP periodically collects changes uploaded by all clients and applies them to the back-end Oracle database tables. Then, the MGP composes new data, ready to be downloaded to each client during the next synchronization, based on pre-defined subscriptions.

1.4.2.2 Complete Refresh

During a complete refresh, all data for a publication is downloaded to the client. For example, during the first synchronization session, all data on the client is refreshed from the Oracle database. This form of synchronization takes longer because all rows that qualify for a subscription are transferred to the client device, regardless of existing client data.

The complete refresh model is resource intensive as all aspects of synchronization are performed. This model should only be utilized for snapshots/publication items where it is an absolute requirement.

1.4.2.3 Queue-Based Refresh

The developer creates their own queues to handle the synchronization data transfer. There is no synchronization logic created with a queue-based refresh; instead, the synchronization logic is implemented solely by the developer. A queue-based publication item is ideally suited for scenarios that require synchronization to behave in a different manner than normally executed. For instance, data collection on the client; all data is collected on the client and pushed to the server.

With data collection, there is no need to worry about conflict detection, client state information, or server-side updates. Therefore, there is no need to add the additional overhead normally associated with a fast refresh or complete refresh publication item.

1.4.2.4 Forced Refresh

This is not a refresh option; however, we discuss it here because it is often mistaken for a refresh option—specifically, it is often confused with the complete refresh option. The Forced Refresh is a one-time execution request made from within Mobile Manager, the GUI interface for the Mobile Server. The forced refresh option may result in a loss of critical data on the client.

The forced refresh option is an emergency only synchronization option. This option is used when a client is corrupt or malfunctioning, so that you decide to replace the Mobile client data with a fresh copy of data from the enterprise data store with the forced refresh. When this option is selected, any data transactions that have been made on the client are lost.

When a forced refresh is initiated all data on the client is removed. The client then brings down an accurate copy of the client data from the enterprise database to start fresh with exactly what is currently stored in the enterprise data store.

1.5 Mobile Application Design

Before you start to design your Mobile application, it is important to read the following sections to understand the differences between an enterprise application and the mobile application as well as the choices you have in designing your application:

- Section 1.5.1, "Steps for Designing Your Mobile Application"
- Section 1.5.2, "Application Programming Interfaces"
- Section 1.5.3, "Supported Languages for Building Mobile Applications"
- Section 1.5.4, "Application Deployment into the Mobile Environment"

1.5.1 Steps for Designing Your Mobile Application

With a proper design, you can avoid the most common causes for mobile project failure, not meeting the needs of the business, poor performance, or issues occurring within a production environment. Proper design of the Mobile System includes the infrastructure and the mobile application. Without proper design, a mobile architecture could end up costing more then it saves.

The following assumption is one of the most common misconceptions for taking an enterprise application and incorporating it into a mobile component:

The mobile application is a scaled down version of the enterprise application.

By taking an existing enterprise application, you may intend to provide the same functionality in remote or disconnected locations. Since the enterprise application has already undergone thorough requirements gathering, design, development, testing, and successful implementation, you may assume that it will automatically work seamlessly as a mobile application and so do not test it in this environment. This assumption may lead to project failure.

For example, take an enterprise form-based client/server application. You have a client connecting through a middle-tier connecting to a database on the back-end. Taking this to a mobile infrastructure, such as with Oracle Database Lite, adds a completely new tier that did not exist within the original infrastructure. The Mobile Server tier introduces new concerns, such as the following:

- Security—There is now a system in the infrastructure that potentially gives any outsider access to the entire organization if proper security configuration and implementation is not performed.
- Bandwidth—The Mobile Server may become a bottleneck for all remote locations without the implementation of a Web farm.
- Scalability—The applications are performing synchronization of hundreds to millions of records, which is not the same as providing static Web pages to a large number of users. A system that is fine for serving static Web pages may not be capable of servicing hundreds of users performing synchronization.

A complete redesign of the system specifications may be in order.

You may also need to re-evaluate the original design of the enterprise application. The following lists a few design considerations for the mobile application:

- Memory—An application designed, tested, and implemented on a multiprocessor system with several gigabytes of memory will not perform the same on a standard PC with only a single processor and maybe 512 megabytes of memory.
- Resource Limitations—Several years ago, limitations of available resources made the usage of data types an extreme concern. The storage space saved by using a small integer over an integer was crucial due to limited memory available. With advances in memory and system resources, this has not been a concern to most modern developers. Now, the mobile infrastructure brings resource limitations back to the list of chief concerns for the design and development of mobile applications. One of the most significant of these limitations is the bandwidth available for the mobile client. If a Mobile client is only able to synchronize over a cell phone network, you may not wish to bring a million records down to a client that only needs a few thousand records. This decision impacts the synchronization performance, as well as the costs associated with the synchronization. If the mobile client was only utilized to collect data, then you can create a data collection queue for synchronization and avoid the whole download phase of synchronization.
- Use of Indexes—You use indexes for avoiding full-table scans. So, if you use the same data subset originally designed for a Windows machine down on a client device and do not use an index, then the performance may be adversely effected. Oracle Database Lite uses two types of scans for queries: full table scans and index based scans.

Thus, we recommend the following steps:

- Section 1.5.1.1, "Read the Documentation Before Design"
- Section 1.5.1.2, "Gather Mobile Requirements"

1.5.1.1 Read the Documentation Before Design

To ensure that you develop your applications correctly, we recommend that you start with a light scan of all of the documentation followed by detailed reading of pertinent areas related to the specific project that are to be undertaken.

1.5.1.2 Gather Mobile Requirements

Gathering requirements is the key to successful project development; a mobile project is no exception.

Mobile users have a specific task they perform. Gather the specific requirements of each mobile user type that will use the mobile application. For example, an insurance application may have two types of mobile users:

- The insurance investigator may only drive to locations to photograph an incident for an insurance claim before any required evidence is altered or vanishes. This user needs a mobile application that captures and uploads the images of the incident.
- A claims agent may later be sent to the site to process a claim for the client. This user needs more details for the client, such as their account information and what they are covered for in order to properly process a claim.

Each user requires a different subset of the enterprise data and functionality within the mobile application.

1.5.1.3 Proof of Concept

Proof of concept testing determines if the Oracle Database Lite mobile architecture meets the needs of the business before architectural design starts.

When an aspect of the mobile solution is determined to meet a requirement, a quick proof of concept test ensures that the requirement may be satisfied by the potential solution. There is a difference in what can be done in a mobile environment versus what can be accomplished in the enterprise environment.

1.5.1.4 Prototype

Prototyping may be seen as too costly to utilize for the proper design of a mobile architecture and application. However, consider that a prototype may be anything from sketching out a design on paper to writing a full application prototype.

You may use one of the sample applications included with the Oracle Database Lite product, such as the transport demo, to gain a full understanding of the infrastructure and the design considerations it involves. Take the application through the setup of the environment, the creation of users, the deployment of the mobile application to multiple clients, the synchronization of multiple clients, and the testing of updates

made to the application. This provides a better understanding of the infrastructure as actual hands on experience.

1.5.1.5 Design for Data Subsets

Data subsetting reduces data to be downloaded, which has a direct impact on performance.

What is Data Subsetting? Data subsets are a crucial part of the design for a mobile application. Data sub-setting is accomplished through variables that limit the data to a specific requirement, such as by region or department. An easy way to conceptualize data subsets is to consider a subset that is limited through a `WHERE` clause that uses bind variables—also known as snapshot template variables—to limit the snapshot data.

When you set up your publication item, you may have set up an input parameter that defines what snapshot of data is to be retrieved for this user. For example, if you have an application that retrieves the customer base for each sales manager, the application needs to know the sales manager's identification number to retrieve the data specific to each manager. Thus, if you set up each sales manager as a unique user and set their identification number in the data subsetting screen, then the application is provided that unique information for retrieving data.

1.5.1.6 Design for Indexing

Oracle Database Lite maintains two basic methods for accessing data within a table: full table scan and index-based access. The performance of an application can be affected if the design does not include the appropriate indexing. Use indexes to avoid full table scans.

You can execute the Explain Plan and SQL Trace capabilities within the product to determine the access method used by a statement and to determine if indexes are being utilized. Analyze your statements throughout the design and development cycles to ensure that they are being designed and developed for maximum efficiency.

1.5.1.7 Design for Sequences

Sequences guarantee uniqueness of a value, such as a primary key. Design how the sequences are generated within the mobile infrastructure. For example, if the enterprise database generates a sequence number and the Mobile client generates the same sequence number a conflict with the data occurs and causes an error.

Native sequences may be formed specifically for the Mobile clients. These sequences would never populate on the enterprise database itself, so there is no risk of a conflict occurring. This works well when data updates only occur from the Mobile clients and input to the database does not come from any other source. However, it is often necessary to have the sequences generated by both the database and the clients. To accomplish this, sequences must be designed so the database uses a range separate from the range used by the clients. For example, you could define the sequences where the database uses all odd numbers and the clients uses all even numbers.

You must also design sequences for the Mobile clients, so that each client uses a unique range of values without any two clients using the same range. For this you specify the sequence range for each client, such as sequences 1 through 1000 for client A and sequences 1001 through 2000 for client B. Using these ranges for the sequence numbers prevents each client from using the same sequence number as used by another client.

1.5.1.8 Design for Synchronization

If you are using the Mobile option, synchronization holds the mobile infrastructure together.

Analyze all of the data needed by the mobile user, as follows:

- Most snapshots are created where the data can be modified on either the client or the server, where the modifications are propagated to the other side through synchronization.
- If any snapshots require only the ability to read the data—that is, all modifications to the data are made on the server-level and not by the user—then create read-only snapshots.
- If all or a majority of the users use the same read-only snapshots, then create a cached user that shares the read-only data across multiple clients.

Analyze the type of synchronization that is appopriate for the user's needs, as described below:

- For optimal performance, use fast refresh for all publications, if appropriate.
- Only design publication items for a complete refresh if the following is true:
  - If it is absolutely critical for all changes to be processed and applied to the data store immediately.
  - If it is critical that any enterprise updates are immediately brought down to the client.

> **Note:** The complete refresh is the most resource intensive method and should only be utilized after full consideration of the performance hit is analyzed. Only time critical publication items should be specified for a complete refresh synchronization type.

- If a mobile user is only performing data collection and it is not necessary for server updates to be brought down to the user, then implement a push-only synchronization model for those publication items. For more information, see Section 3.19.2, "Creating Data Collection Queues for Uploading Client Collected Data" in the *Oracle Database Lite Developer's Guide*.
- When a mobile user only requires specific table to be updated or synchronized, perform a selective synchronization methodology limiting the synchronization process to specific tables or specific publications.

1.5.1.9 Design for Administration

When designing for administration, you may create standardized groups and assign users to these groups. It is much easier to administer groups of users then it is to administer several thousand users each as individuals. The situations where groups simplify your administration is when your organizational architecture has data and the environment is structured into a hierarchy, such as by region or department. In this situation, all properties, access settings, and template variables may be set for groups of users and not the users individually.

1.5.1.10 Design for the Language Utilized for Handheld Devices

Any language selected for handheld development has limitations not existing in their fuller counterparts. The mobile application must be designed to meet these limitations. For example, when developing with WinCE utilizing the Microsoft Compact

Framework with C# or VB.Net, forms are not maintained the same way as they are in the full framework when running on a normal PC. Without designing for this limitation, it would be easy to accidentally leave forms sitting in the background that should have been closed when the application exits. This prevents crucial memory from being freed and also prevents any application updates from downloading and installing correctly.

When you properly research and design for the device platform, everything done on the device may be performed on the larger non-mobile applications. Thus, you can reuse and update code simply as it uses the same coding for both environments. Designing in the reverse may result in functionality being utilized in the non-mobile applications that does not have a matching functionality in the mobile version.

1.5.2 Application Programming Interfaces

When you are developing your application, you may decide that you want to control more aspects of Oracle Database Lite within your application—rather than relying on user interaction. In this case, you can use the Oracle Database Lite Application Programming Interfaces (APIs). Almost any task performed by the tools and utilities included with Oracle Database Lite may also be accomplished with the APIs. Some of the more advanced functionality within the product is only available through the use of the APIs. Except for the synchronization APIs which are provided for most languages utilized for application development, most of the APIs are Java interfaces that must be developed with the Java programming language.

The most common APIs utilized and their uses are as follows:

- Synchronization APIs: These APIs provide all of the basic synchronization functionality that is found within the mSync utility. The advantages of using these APIs are that the synchronization process can be fully integrated within the actual Mobile application. The APIs also provide the ability for a push-only synchronization, which allows Mobile clients to only upload data skipping the downloading of new data or applications. The push-only model is useful when bandwidth is limited and when the client just collects data—that is, it is not necessary for a remote client to have updated data from the enterprise.
- Consolidator APIs: The Consolidator APIs provide administrative functionality for creating users, setting the user properties, working with applications, and so on. You can automate common administration tasks and speed up some of the administration tasks required, such as the creation of a large amount of users. The only limitation is that application and user settings are not displayed in the Mobile Manager Web administration tool as these APIs directly access the Mobile Repository.
- Mobile Resource Manager APIs: The Mobile Resource Manager APIs also provides administration functionality for users and applications; however, this API actually updates the Mobile Manager administration tool as well as the repository. This utility may be used to create users, set user access, set the user template variables, and many other tasks.
- Device Manager APIs: The Device Manager APIs provide the ability customize the management of devices. These APIs may be used to gather information on devices, send commands to devices, register devices, and so on.

1.5.3 Supported Languages for Building Mobile Applications

Mobile database applications can be developed using any of the following languages:

- The most common way is to develop native C or C++ applications for specific Mobile platforms. C++ applications can access the Oracle Database Lite database using the Simple Object Data Access API (SODA), an easy-to-use C++ interface that is optimized for the object-oriented and SQL functionality of Oracle Database Lite. For more information about SODA, refer the SODA API documentation, which is installed as part of the Mobile Development Kit.
- Applications that need a standard interface and work with multiple database engines can use either the Open Database Connectivity (ODBC) interface, Active Data Object (ADO) interface, or some other interface built on top of ODBC. ADO.NET can be used on Win32 and Windows CE. Another way to develop a Mobile database application is to use Java and the Java Database Connectivity (JDBC) interface.
- Oracle Database Lite also offers a third way to develop Mobile database applications using the servlet-based Web application model called Web-to-Go. Web-to-Go applications can be built using Web technologies, such as servlet, Java Sever Pages (JSP), applet, HTML, and JDBC.
- Symbian applications that need a standard interface and work with multiple database engines can use either the Open Database Connectivity (ODBC) interface or some other interface built on top of ODBC.

Oracle Database Lite is an integrated framework that simplifies the development, management, and deployment of mobile applications on the following mobile platforms, operating systems, and hardware:

Platform	Programming Languages	Operating System	Hardware
Win32 on a laptop or notebook	When you develop on a laptop, you are using one of the Windows operating systems. You can use any of the languages mentioned in this book. However, C, C++ are better for creating applications with a good user interface. The languages available are C, C++, C#, Java, Visual Basic, JSPs and Servlets. You can use Visual Studio 2003 for development.	Windows XP	Pentium processor
PocketPC	C, C++, Visual Basic, Java applications (no Servlet or JSP support), SODA, ADO.NET. You can use Visual Studio 2005 for development.	Windows CE (WinCE)	ARM
Linux	Java, C, C++	Linux Redhat 3.0	X86
Symbian OS on Nokia and Motorola	C, C++, JDBC	Symbian OS versions 7.0 and 8.0	ARM

Oracle Database Lite provides the Mobile Development Kit, which includes facilities, tools, APIs, and sample code for you to develop your applications. There are three application models:

- Section 1.5.3.1, "Native Applications"
- Section 1.5.3.2, "Standalone Java Applications"

- Section 1.5.3.3, "Web Applications"

1.5.3.1 Native Applications

Native applications are built using C, C++, Visual C++, Visual Basic, Embedded Visual tools, ActiveX Data Objects (ADO), and MetroWerks CodeWarrior. The application must be compiled against the mobile device operating system, such as the Windows CE platform.

Use ODBC to access the Oracle Lite database on the client. See Section 2.4.1, "ODBC" in the *Oracle Database Lite Developer's Guide* for more information.

See Chapter 8, "Native Application Development" in the *Oracle Database Lite Developer's Guide* for more information on C and C++.

1.5.3.2 Standalone Java Applications

Standalone Java applications do not include Web and J2EE technology—such as Servlets and JSPs. Instead, Java applications revolve around using JDBC driver to access the Oracle Lite database on the client platform, and use AWT and SWING classes to build the application UI. In addition, the database supports Java stored procedures and triggers.

Your Java/JDBC application must be compiled for the particular mobile device JVM environment, which can be different across various client devices. Thus, when you are developing your Java application, do the following:

1. Check the environment: Verify that the `olite40.jar`, which is located in `OLITE_HOME/bin`, is in your `CLASSPATH`, which should have been modified during installation.
2. Load the JDBC driver in to your applications. The following is an example:

   ```
   Class.forName("oracle.lite.poljdbc.POLJDBCDriver");
   ```

3. Connect to the Oracle Lite database installed on the client. If your database is on the same machine as the JDBC application, connect using the native driver connection URL syntax, as follows:

   ```
   jdbc:polite:dsn
   ```

   Or if not local, connect as follows:

   ```
   jdbc:polite@[hostname]:[port]:dsn
   ```

See Chapter 9, "Java Application Development" and Chapter 10, "JDBC Programming" in the *Oracle Database Lite Developer's Guide* for more information.

1.5.3.3 Web Applications

You can execute existing Web applications using the J2EE Java technologies, such as servlets and JSPs, in a disconnected mode without modifying the code base. Web-to-Go is a development option for Web applications, and can be executed on laptops. Web-to-Go applications use Java servlets and JSPs that may invoke JDBC to access the database.

For more information, see Chapter 6, "Developing Mobile Web-to-Go Applications" in the *Oracle Database Lite Developer's Guide*.

1.5.4 Application Deployment into the Mobile Environment

Deployment of applications includes setting up the server system so that end users can easily install and use the applications. Mobile applications are deployed to the Mobile Server.

Deployment consists of the following steps:

1. Create the publication with the Mobile Database Workbench (MDW). See Section 1.7.2, "Using the Mobile Database Workbench" for more information.
2. Publishing the application to the server includes installing all the components for an application on the Mobile Server with the Packaging Wizard tool. See Section 1.7.3, "Using the Packaging Wizard" for details.
3. Provisioning the applications to the Mobile users through the Mobile Manager, which is a GUI interface for the Mobile Server. This phase includes determining user accesses to applications with a specified subset of data. The Mobile Manager can create users, grant privileges to execute applications, and define the data subsets for them, among others. You can also use the Java API to provision applications.
4. Testing for functionality and performance in a real deployment environment. A Mobile application system is a complex system involving the following:
   - Multiple Mobile device client technologies—such as, operating systems, form factors, and so on.
   - Multiple connectivity options—such as, LAN, Wireless LAN, cellular, wireless data, and other technologies.
   - Multiple server configuration options.

   When testing, pay particular attention to tuning the performance of the data subsetting queries, as it is the most frequent cause of performance problems.
5. Determining the method of initial installation of applications on Mobile devices (application delivery). Initial installation involves installing the Oracle Database Lite client, the application code, and the initial Oracle Lite database. The volume of data required to install applications on a Mobile device for the first time could be quite high, necessitating the use of either a high-speed reliable connection between the Mobile device and the server, or using a technique known as offline instantiation. In offline instantiation, everything needed to install an application on a Mobile device is put on a CD or a floppy disk and physically given to the user. The user then uses this media to install the application on the device by means of a desktop machine. Oracle Database Lite provides a tool for offline instantiation.

After deployment, both the application and the data schema may change because of enhancements or defect resolution. The Mobile Server manages application updates and data schema evolution. The only requirement is that the administrator must republish the application and the schema. The Mobile Server automatically updates the Mobile clients that have older version of the application or the data.

1.6 Oracle Database Lite Components Involved in Synchronization

The following is a more formal definition of each of the components involved in the synchronization process:

- Section 1.6.1, "Oracle Database Lite RDBMS"
- Section 1.6.2, "Mobile Sync"

1.6.1 Oracle Database Lite RDBMS

The Oracle Database Lite RDBMS is a small footprint, Java-enabled, secure, relational database management system created specifically for laptop computers, handheld computers, PDAs, and information appliances. The Oracle Database Lite RDBMS runs on Windows 2000/XP, Windows CE/Windows Mobile, Linux, and Symbian. Oracle Database Lite RDBMS provides JDBC, ODBC, and SODA interfaces to build database applications from a variety of programming languages such as Java, C/C++, and Visual Basic. These database applications can be used while the user is disconnected from the Oracle database server.

When you install the Mobile Development Kit, the Oracle Database Lite RDBMS and all the utilities listed in Appendix C are installed on your development machine. In a production system, when the Mobile Server installs Oracle Database Lite applications, only the RDBMS, the Mobile Sync, and Mobile SQL applications are installed on the client machine.

1.6.2 Mobile Sync

Mobile Sync (msync) is a small footprint application that resides on the Mobile device. Mobile Sync enables you to synchronize data between handheld devices, desktop and laptop computers and Oracle databases. Mobile Sync authenticates locally, collects changes from the Oracle Lite database and sends them to the server, where the user is authenticated before the changes are uploaded.

Use the `msync` executable for Mobile Sync.

Mobile Sync synchronizes the snapshots in Oracle Database Lite with the data in corresponding Oracle data server. These snapshots are created by the Mobile Server for each user from the publication items associated with a Mobile application. The Mobile Server also coordinates the synchronization process.

The Mobile Sync application communicates with the Mobile Server using any of the supported protocols (that is, HTTP or HTTPS). When called by the Mobile user, the Mobile Sync application first collects the user information and authenticates the users with the Mobile Server. It then collects the changes made to Oracle Database Lite (from the snapshot change logs) and uploads them to the Mobile Server. It then downloads the changes for the user from the Mobile Server and applies them to the Oracle Database Lite.

In addition to this basic function, the Mobile Sync application can also encrypt, decrypt, and compress transmitted data.

When you install the Mobile Development Kit, the Mobile Sync application is also installed on your development machine. The Mobile Server also installs the Mobile Sync on the client machine as part of application installation.

1.6.3 Mobile Server

The installation of the Mobile Server requires an Oracle database instance to be running. You can use an existing test database as well. The Mobile Server stores its metadata in this database.

The Mobile Server is a Web middle-tier server that can exist on Windows, Solaris, HP-UX, AIX, and Linux. The Mobile Server uses the Oracle Containers for Java (OC4J) Web engine and provides the interface between the mobile infrastructure and the enterprise database. Most administration tasks are accomplished through the Mobile Server Web application—the Mobile Manager.

The Mobile Server provides the following features.

- Application Publishing
- Application Provisioning
- Application Installation and Update
- Data Synchronization

The Mobile Manager application provides the capability to manage users, devices, publications and applications. This utility can provides the following:

- Monitors and manages synchronization between the client data store and the enterprise data store.
- Sends administrative commands to the Mobile clients. These commands capture data and logs from the client or instructs the client to carry out necessary tasks. For example, the Mobile Manager could send a command to a client to perform synchronization or to remove the entire client data store, if a device may have been compromised.

> **Note:** If you wish, you can also accomplish the same tasks as the Mobile Manager with the Application Programming Interfaces (APIs).

As with any Web server tier, the Mobile Server may be configured within a Web farm for improved performance within the mobile infrastructure. This enables the use of a load balancer, such as the balancer included with Oracle Internet Application Server (Oracle AS), or with one provided by a 3rd party vendor. The Mobile Server is designed to be fully integrated with Oracle AS to take advantage of the features within Oracle AS, such as the Oracle Internet Directory (OID) capabilities.

> **Note:** As the Mobile Server is a Web-based environment, it is important to design for a proper security environment as would be done with any Web server.

The Mobile Server has two major modules called the Resource Manager and the Consolidator Manager. The Resource Manager is responsible for application publishing, application provisioning, and application installation. The Consolidator Manager is responsible for data and application synchronization.

Application publishing refers to uploading your application to the Mobile Server so that it can be provisioned to the Mobile users. Once you have finished developing your application, you can publish it to the Mobile Server by using the development tool called the Packaging Wizard.

Application provisioning is concerned with creating subscriptions for users and assigning application execution privilege to them. Application provisioning can also be done in one of two ways.

- Using the administration tool called the Mobile Manager, you can create users and groups, create subscriptions for users by assigning values to subscription parameters, and give users or groups privileges to use the application.
- Using the Resource Manager API, you can programmatically perform the above tasks.

End users install Mobile applications in two steps.

1. As the Mobile user, browse the setup page on the Mobile Server and choose the setup program for the platform you want to use.
2. Run the Mobile Sync (mSync) command on your Mobile device, which prompts for the Mobile username and password. The Mobile Sync application communicates with the Consolidator Manager module of the Mobile Server and downloads the applications and the data provisioning for the user.

After the installation of the applications and data, you can start using the application. Periodically, use msync or a custom command to synchronize your local database with the server database. This synchronization updates all applications that have changed.

1.6.4 Message Generator and Processor (MGP)

The Consolidator Manager module of the Mobile Server uploads the changes from the client database to the server, and it downloads the relevant server changes to the client. But it does not reconcile the changes. The reconciliation of changes and the resolution of any conflicts arising from the changes are handled by MGP. MGP runs as a background process which can be controlled to start its cycle at certain intervals.

Note: The mobile infrastructure may allow for multiple Mobile Servers to be configured within a Web farm. However, there may only be one MGP application utilized for the entire Web farm.

Each cycle of MGP consists of two phases: Apply and Compose.

The Apply Phase

In the apply phase, MGP collects the changes that were uploaded by the users since the last apply phase and applies them to the server database. For each user that has uploaded his changes, the MGP applies the changes for each subscription in a single transaction. If the transaction fails, MGP will log the reason in the log file and stores the changes in the error file.

The Compose Phase

When the apply phase is finished, MGP goes into the compose phase, where it starts preparing the changes that need to be downloaded for each client.

Applying Changes to the Server Database

Because of the asynchronous nature of data synchronization, the Mobile user may sometimes get an unexpected result. A typical case is when the user updates a record that is also updated by someone else on the server. After a round of synchronization, the user may not get the server changes.

This happens because the user's changes have not been reconciled with the server database changes yet. In the next cycle of MGP, the changes will be reconciled with the server database, and any conflicts arising from the reconciliation will be resolved. Then a new record will be prepared for downloading the changes to the client. When

the user synchronizes again (the second time), the user will get the record that reflects the server changes. If there is a conflict between the server changes and the client changes, the user will get the record that reflects either the server changes or the client changes, depending on how the conflict resolution policy is defined.

1.6.5 Mobile Server Repository

The Mobile Server Repository contains all the application data as well as all information needed to run the Mobile Server. The Mobile Repository contains the repository schema under which all the data mapping and internal tables utilized to maintain data synchronization exist. This schema also stores the application, application tables and its data published for use with a Mobile client.

The information is normally stored in the same database where the application data resides. The only exception to this is in cases where the application data resides in a remote instance and there is a synonym defined in the Mobile Server to this remote instance.

The Mobile Repository contains some internal tables that the Mobile Server uses to perform its functions. You may query these tables to gain more details about the current state of the environment; however, most of the information needed from these tables is already accessible from the Mobile Manager. You should never alter any of the internal tables and their contents unless explicitly directed to by Oracle Support Services or Oracle Development.

Administration, backup, and recovery of the repository are no different then for any other Oracle database requiring standard Database Administrator (DBA) skills

Changes to the repository should only be made using the Mobile Server Mobile Manager or the Resource Manager API.

1.6.6 Device Manager

The Device Manager manages client devices. On install of the Mobile client, the Device Manager registers a device with the Mobile Server. The Device Manager invokes the update executable after synchronization completes to determine if any mobile application updates are available, then downloads and installs these application updates to a Mobile client. You can request—through the Mobile Manager—that certain commands are invoked on the client. The Device Manager executes these commands. The Device Manager is responsible for most administrative actions between the Mobile Server and the Mobile client.

1.7 Mobile Development Kit (MDK)

On the middle tier, Oracle Database Lite installation provides you with an option to install the Mobile Server and/or the Mobile Development Kit. For application development, you need to install the Mobile Development Kit on your development machine.

Before you develop an application using Oracle Database Lite, you should install the Oracle Database Lite Mobile Development Kit (MDK) on the machine on which you intend to develop your application. The Mobile Development Kit is installed in `<ORACLE_HOME>\Mobile\Sdk`. For instructions on how to install the Mobile Development Kit, see Section 4.3, "Installing Oracle Database Lite".

The Oracle Database Lite Mobile Development Kit includes the following components.

- Oracle Database Lite RDBMS—A lightweight, object-relational database management system, including Mobile SQL (`msql.exe`). Mobile SQL is written in Java. It requires the Java runtime environment JRE 1.4.2 or higher to be installed on your system before you can use it. If you have installed JDK 1.4.2 or higher, the JRE is already installed in your machine
- Mobile Database Workbench (MDW)—A development tool for creating a publication.
- Packaging Wizard—A tool to publish applications to the Mobile Server through executing `runwtgpack.bat`.
- Mobile Sync—A transactional synchronization engine that includes the executable (`msync.exe`) and the Java wrapper for it.
- mSQL—An interactive tool to create, access, and manipulate Oracle Database Lite on laptops and handheld devices

Using any C, C++, or Java development tool in conjunction with the Mobile Development Kit for Windows, you can develop your Mobile applications for Windows against Oracle Database Lite, and then publish the applications to the Mobile Server by using the Packaging Wizard. See Section 4.3, "Installing Oracle Database Lite" for instructions on how to install the Mobile Server.

Once you have published the applications to the Mobile Server, you can use the Mobile Manager to provision the applications to the Mobile users. Provisioning involves specifying the values of the subscription parameters used for subsetting the data needed by the application for a particular user. A user to whom an application has been provisioned can then log in to the Mobile Server and request it to set up everything the user needs to run the applications on the user's device.

The Samples directory <*ORACLE_HOME*>`\Mobile\Sdk\Samples` directory contains some sample applications. Section 2.12, "Using Oracle Database Lite Samples" in the *Oracle Database Lite Developer's Guide* describes the sample programs and explains how to run them. You should familiarize yourself with the various Oracle Database Lite features by perusing the source code and running the samples.

When you install the MDK, it installs a starter database file in the <*ORACLE_HOME*>`\Mobile\Sdk\OLDB40` directory named `polite.odb`.

Note: The `polite.odb` starter database is not the name of the Mobile client database. For information on what Oracle Lite database (ODB) files are installed on the client, see Section 2.2, "Synchronizing or Executing Applications on the Mobile Client" in the *Oracle Database Lite Administration and Deployment Guide*.

When you install the Mobile Development Kit, the installer sets the `PATH` environment variable to include the bin directory of the Mobile Development Kit. You can use the Command Prompt on your Windows 32 machine to do the following quick test.

At the command prompt, enter the following.

```
msql system/manager@jdbc:polite:polite
...
SQL>create table test (c1 int, c2 int);
Table created
SQL>insert into test values(1,2)
1 row(s) created
SQL>select * from test;
```

```
                C1 | C2
               ----+----
                1  |  2
SQL>rollback;
Rollback completed
SQL>exit
```

1.7.1 Mobile SQL (mSQL)

Mobile SQL is an interactive tool that allows you to create, access, and manipulate Oracle Database Lite on laptops and handheld devices. The mSQL installations on laptops cannot be used to create a database, but can create a database on hand-held devices. Using mSQL, you can perform the following actions.

- Create database objects such as tables and views
- View tables
- Execute SQL statements

The mSQL tool is installed with the Mobile Development Kit installation. It is also installed by the Mobile Server as part of application installation. The mSQL tool for the Windows 32 platform is a command line tool that is similar to the Oracle SQL*Plus tool, but does not provide compatibility with SQL*Plus. The mSQL tool for Windows CE supports a graphical user interface.

Note: UTF8 SQL Scripts are not supported in mSQL.

1.7.2 Using the Mobile Database Workbench

The Mobile Database Workbench (MDW) is a new tool that enables you to iteratively create and test publications—testing each object as you add it to a publication. Publications are stored within a project, which can be saved and restored from your file system, so that you can continue to add and modify any of the contained objects within it.

Once you create the project, start creating the publication items, sequences, scripts and resources that are to be associated with the publication. You can create the publication and associated objects in any order, but you always associate an existing object with the publication. Thus, it saves time to start with creating the objects first and associating it with the publication afterwards.

For detailed information on how to use MDW, see Chapter 5, "Using Mobile Database Workbench to Create Publications" in the *Oracle Database Lite Developer's Guide*.

1.7.3 Using the Packaging Wizard

The Packaging Wizard is a graphical tool that enables you to perform the following tasks.

1. Create a new Mobile application.
2. Edit an existing Mobile application.
3. Publish an application to the Mobile Server.

When you create a new Mobile application, you must define its components and files. In some cases, you may want to edit the definition of an existing Mobile application's components. For example, if you develop a new version of your application, you can use the Packaging Wizard to update your application definition. The Packaging

Wizard also enables you to package application components in a .jar file which can be published using the Mobile Manager. The Packaging Wizard also enables you to create SQL scripts which can be used to execute any SQL statements in the Oracle database.

For detailed information on how to use the Packaging Wizard, see Chapter 6, "Using the Packaging Wizard" in the *Oracle Database Lite Developer's Guide*.

1.8 Overview of Performance Tuning

Mobile devices do not have the processing power and memory that standard enterprise systems maintain. If the mobile applications and infrastructure are not tuned appropriately they really are of little benefit to the organization.

The two most important components within the mobile infrastructure to perform tuning on in order to increase the feasibility of a mobile application are as follows:

- The time it takes to enter and retrieve data.
- The time it takes to synchronize data with the enterprise data store.

The type of performance you enjoy can be tuned with SQL query tuning and through the use of the ConsPerf utility provided with Oracle Database Lite, as follows:

- Section 1.8.1, "SQL Query Tuning"
- Section 1.8.2, "Analyzing Performance of Publications With the Consperf Utility"

1.8.1 SQL Query Tuning

To decrease the time it takes to enter and retrieve data, query tuning may be performed through the use of the Explain Plan or through SQL tracing and indexes. Unlike the enterprise database, Oracle Database Lite utilizes two methods for scanning a table and retrieving data: a full table scan and an index based scan.

- Section 1.8.1.1, "Determining Performance of Client SQL Queries With the EXPLAIN PLAN"
- Section 1.8.1.2, "Determining Performance of Client SQL Queries With SQL Tracing"

1.8.1.1 Determining Performance of Client SQL Queries With the EXPLAIN PLAN

If you want to access data on the local client Oracle Lite database, then you can use the EXPLAIN PLAN to determine the performance of your SQL query execution on the Oracle Lite database. To execute a SQL statement, Oracle might need to perform several steps. Each of these steps either physically retrieves rows of data from the database or prepares them in some way for the user issuing the statement. The combination of the steps Oracle uses to execute a statement is called an execution plan, which includes an access path for each table that the statement accesses and an ordering of the tables (the join order) with the appropriate join method. The execution plan shows you exactly how Oracle Database Lite executes your SQL statement.

The components of an execution plan include the following:

- An ordering of the tables referenced by the statement.
- An access method for each table mentioned in the statement.
- A join method for tables affected by join operations in the statement.

The EXPLAIN PLAN command stores the execution plan chosen by the Oracle Database Lite optimizer for SELECT, UPDATE, INSERT, and DELETE statement.

You can generate an Explain Plan using either of the following methods:

- The Consperf tool: The Consperf tool generates the following two primary analysis reports:

  1. Timing statistics for publication items
  2. Explain plans for publications

  For a full description of how to use the Consperf utility, see Section 1.2.1 "Analyzing Performance of Publications With the Consperf Utility" in the *Oracle Database Lite Troubleshooting and Tuning Guide*.

- Manually generrated. See the Section 1.11 "Tuning SQL Statement Execution with the EXPLAIN PLAN" in the *Oracle Database Lite SQL Reference* for full details on how to manually create an EXPLAIN PLAN.

1.8.1.2 Determining Performance of Client SQL Queries With SQL Tracing

By setting the parameter `OLITE_SQL_TRACE = TRUE` in the `polite.ini` or `polite.txt` file on the client device, Oracle Database Lite generates a trace file named `oldb_trc.txt` that shows the following:

- The order tables are accessed by a query.
- The table scan access method used.
- The value of any bind variables utilized by the query.
- The time it takes for the first record to be retrieved.

These are the main items reported that you can use to tune the majority of SQL queries.

If the trace file identifies that a full table scan is occurring, the most common way to get better performance from the query is to add an index that accommodates that query. To do this quickly for testing purposes, you can create the indexes directly on the client until you are satisfied with the performance. Once you identify the structure for the indexes, then they should then be created in the Mobile Database Workbench, or with the APIs with the publication items. Then, brought down to the device through the normal deployment processes. Final testing should validate the deployment and improvements before any application is even moved out of the development stage for testing, quality assurance, or deployment into production.

1.8.2 Analyzing Performance of Publications With the Consperf Utility

The Consperf utility profiles your subscriptions and may modify how the publication item is executed if the utility determines that there is a more performant option. The Consperf tool evaluates how the SQL within the publication item interacts with our Data Synchronization query templates. The first synchronization is always a complete refresh, which is a direct invocation of the query. On subsequent synchronizations, the query templates determine incremental refreshes. This improves your performance from not having to perform a complete refresh each time you synchronize. However, the interaction of our query templates and your SQL may not be optimal, which is discovered by the Consperf tool. We either modify the query template or type of logical delete or insert for you or you can adjust your SQL to be more performant in regards to our templates.

In addition, application developers and administrators use this utility to analyze the performance of subscriptions and identify potential bottlenecks during synchronization.

1.9 Security Considerations

The introduction of handheld devices within the corporate environment can pose a security threat to an organization. Devices are now used to store not only company contacts; but, with external cards, may store up to 60 gigabytes of information or more. Devices also provide a mobile point of entry into the organizational network that is located outside the network security perimeter. It is essential to secure this data if a device is lost or compromised.

Securing a device involves a layered approach. You must secure not only access to the device, but data stored on the device and communications across the network. Most aspects of security for a mobile device must be incorporated before Oracle Database Lite is even involved within the security infrastructure.

1. Security needs to start with the device itself. Authentication on the device must be implemented through pin or password authentication, biometric readers, secure digital media for storage, and even how the device is stored, transported, and accounted for.
2. Once access is gained to the device, further security needs to be implemented within the mobile application to prevent the application from being able to retrieve invalid data. Technologies, such as the Microsoft.Net Compact Framework, incorporate API calls that may be used to encrypt and decrypt any data that will be stored or retrieved from the device.

Oracle Database Lite provides several security features that may be utilized to help in securing data. These features aid in protecting information during both synchronization, and once access to a device has been obtained. The two most important aspects of security provided by Oracle Database Lite for the mobile infrastructure are the following:

1. Use Secure Socket Layer (SSL) to protect the transmission of data during the synchronization process.
2. Use the Oracle Database Lite encryption utility, `ENCRYPT_DB`, to help protect the actual database files.

1.9.1 The Oracle Database Lite Encryption Utility—EncryptDB

EncyrptDB encrypts the Oracle Lite database by using 128 bit Advanced Encryption Standard (AES) encryption. This does not encrypt the data stored within the Oracle Lite database itself; it only encrypts the database as a whole.

2

Third Party Licensing in Oracle Database Lite

This chapter includes third-party license information for all third-party products included with Oracle Database Lite. Oracle acknowledges that that following Third Party and Open Source software are used in the provided programs covered by this documentation.

2.1 Sun Microsystems JRE 1.4.2 and JRE 5.0

This product includes code provided by Sun Microsystems.

2.2 Third Party Licensing for ZLib and JZLib

Oracle gratefully acknowledges the contributions of Jean-loup Gailly and Mark Adler, the authors of zlib.

This product includes JZlib.

Copyright (c) 2000,2001,2002,2003,2004 ymnk, JCraft,Inc. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. The names of the authors may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED ``AS IS'' AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES

OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL JCRAFT, INC. OR ANY CONTRIBUTORS TO THIS SOFTWARE BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

3

Requirements Before Installation or Development

Before you install, you must check to see that you have the correct hardware and software necessary for using Oracle Database Lite on your operating system. The requirements for each type of operating system are detailed in the following sections:

3.1 Include the Latest Patches

For the latest information and patches, refer to Oracle *MetaLink* at the following Web site:

```
http://metalink.oracle.com
```

3.2 Release Notes

We recommend reading *Oracle Database Lite Release Notes* before installing Oracle Database Lite. *Oracle Database Lite Release Notes* are available as part of the documentation shipped with Oracle Database Lite. The most up-to-date version is also available at OTN at the following Web site:

```
http://otn.oracle.com/documentation/index.html
```

3.3 Supported Components and Technologies

The following sections detail the supported components and technologies:

3.3.1 Oracle Universal Installer

The version of the Oracle Universal Installer used is 2.3.0.10.0.

3.3.2 Certified Oracle RDBMS to Use With Oracle Database Lite

Use one of the following database versions with Oracle Database Lite: Oracle 9.2, Oracle 10*g* Release 1 (10.1.0) and Oracle 10*g* Release 2 (10.2.0).

3.3.3 JDK Platform Support

For all Mobile clients, use JRE 5.0. For the Mobile Server and Mobile Development Kit, the version required depends on what version of OracleAS 10*g* you are using.

Table 3–1 JDK Version Supported

OracleAS 10*g* Version Used	JDK Version Supported
Oracle Database Lite Standalone using OC4J Standalone	JDK 5.0
OracleAS 10*g* version 10.1.2	Use either JDK 1.4.2 or 5.0
OracleAS 10*g* version 10.1.3	JDK 5.0

Install the Sun Microsystems JDK before installing the Mobile Server or Mobile Development Kit. If you need to use JDK 5.0 and you already have JDK 1.4.2 installed, then upgrade to version 5.0 as follows:

1. Uninstall the existing JDK version 1.4.2. If you do not remove this version first, then Mobile Server will continue to use version 1.4.2.
2. Install the new JDK version 5.0. After installation, verify that the JDK 5.0 `bin` directory is in the system path. For Linux/UNIX platforms, verify that the `JAVA13_HOME` environment variable points to the JDK 5.0 home directory.

3.3.4 Certified Application Server Configurations and OID Support

The following lists accepted configuration options with a middle-tier application server:

- Standalone Oracle Database Lite, which uses an embedded, standalone OC4J (version 10.1.3) container
- Oracle Application Server 10*g* (10.1.2 or 10.1.3), as long as Oracle Database Lite is installed in the same *`ORACLE_HOME`*

If you want to use OID with your Oracle AS on top of the Mobile Server, then the following shows the support for OID:

- When you use Oracle AS version 10.1.3.1.0, then integration with OID/SSO with AS 10.1.4 infrastructure is supported.
- When you use Oracle AS version 10.1.3.0.x, then integration with OID/SSO is not supported.
- When you use Oracle AS version 10.1.2.0.2 , then integration with OID 10.1.2.0.2 is supported.

3.3.5 Certified Platforms and Technologies for the Mobile Server

You can install the Mobile Server on the following platforms:

- Microsoft Windows 2000 Professional Edition with Service Pack 4
- Microsoft Windows XP Professional Edition with Service Pack 2
- Microsoft Windows 2003 (32 bit)
- Redhat Enterprise Linux Advanced Server 3.0 or 4.0, certified on x86 and AMD64
- SUSE 9, certified on x86 and AMD64
- Sun SPARC Solaris 8.0, 9.0, or 10.0
- HP-UX 11.0 (64-bit)
- HP-UX 11*i* PA-RISC Version 11.11 and 11.23
- AIX 5L with either Version 5.2 and Maintenance Level 4 or Version 5.3

The following are the versions of the supported technologies used with the Mobile Server of Oracle Database Lite. For certified technologies for the Mobile client platforms, see Section 3.8.2, "Software Requirements For Windows Mobile Clients".

Table 3–2 Supported Technologies for the Mobile Server

Technology	Standalone	OracleAS 10*g* (10.1.2)	OracleAS 10*g* (10.1.3)
Java Version (JDK and JRE)	5.0	1.4.2 or 5.0	5.0
OC4J	10.1.3	10.1.2	10.1.3
Servlet	2.4	2.3	2.4
UIX	2.2.15	2.2.15	2.2.15
XML Parser	9.0.3.0.0	9.0.3.0.0	9.0.3.0.0
Oracle JSP	10.1.3.0.0	10.1.2.0.0	10.1.3.0.0
Oracle JDBC Driver	10.1.0.5.0	10.1.0.5.0	10.1.0.5.0

3.3.6 Certified Browsers

Internet Explorer 6.0 and Mozilla 1.7.x are certified as browsers for Oracle Database Lite.

3.4 Oracle Components That Work With Oracle Database Lite

When you synchronize the client, your changes are updated in an Oracle back-end database. Thus, you must have either the Standard or Enterprise Edition Oracle database to use the synchronization ability of Oracle Database Lite.

In addition, Oracle Database Lite uses a middle-tier to communicate between the clients and the back-end database. You use one of the following as the middle-tier:

- Oracle Database Lite in standalone mode, which is automatically installed with Oracle Database Lite—This is the recommended configuration for development environments. Oracle Database Lite in standalone mode uses the standalone version of Oracle Containers for J2EE (OC4J).
- Oracle Application Server 10*g*, which is not installed with Oracle Database Lite—This is the recommended configuration for production environments. If you choose to use the Oracle Application Server 10*g* (OracleAS) as your middle-tier, then you must install it before installing Oracle Database Lite.

When you are installing the Oracle Application Server (OracleAS) in preparation for using Oracle Database Lite, you must choose the Integrated Web and J2EE Server installation option, as follows:

In the OracleAS install, choose the Advanced Installation Mode. In the second screen you will see the following options:

1. J2EE Server and Process Management
2. Web Server and Process Management
3. Integrated Web Server, J2EE Server and Process Management
4. Oracle Toplink

Select ONLY option 3, the Integrated Web Server, J2EE Server and Process Management. This option provides all of the functionality that you need. Do not select any of the other options, as then your OracleAS installation will be missing functionality that is necessary for Oracle Database Lite.

For more information about how Oracle Database Lite works with the middle-tier and the back-end database, see Chapter 1, "Overview" in the *Oracle Database Lite Developer's Guide*.

3.5 System Requirements for Mobile Server on Windows

Before you install the Mobile Server, you must check to see that you have the correct hardware and software necessary for your Windows machines that use Oracle Database Lite. The requirements for both are detailed in the following sections:

- Section 3.5.1, "Hardware Requirements For Windows"
- Section 3.5.2, "Software Requirements For Windows"
- Section 3.5.3, "Defining Password for OracleDatabaseLite User for Branch Office on Windows Machine"

3.5.1 Hardware Requirements For Windows

The hardware requirements for each component of Oracle Database Lite for Windows 2000, Windows XP, or Windows 2003 are described in the following table:

Table 3–3 Hardware Requirements for Windows

Component	Hardware Requirements for this Component
Mobile Server Standalone	CPU: Pentium 4, 3 GHz Disk Space: 1 GB RAM: 1 GB
Mobile Server using OracleAS	See the OracleAS documentation for the OC4J container requirements.
Mobile Development Kit	CPU: Pentium 4, 3 GHz Disk Space: 512 MB RAM: 512 MB

Note: The requirements for Mobile Server is related to the number of users synchronizing and the amount of data transferred.

3.5.2 Software Requirements For Windows

The software requirements for each component of Oracle Database Lite for Windows 2000, Windows XP, or Windows 2003 are described in the following table:

Table 3–4 Software Requirements for Windows

Component	Operating System	Other Software Requirements
Mobile Server	Windows 2000, Windows XP, or Windows 2003	For the appropriate JDK version, see Section 3.3.3, "JDK Platform Support".
Mobile Development Kit	Windows 2000, Windows XP, or Windows 2003	For the appropriate JDK version, see Section 3.3.3, "JDK Platform Support".

You should install all of the patches required for the JDK version you are using on the Windows operating system. This is constantly under review and published on the JDK download page on the Sun Microsystems Web site.

3.5.3 Defining Password for OracleDatabaseLite User for Branch Office on Windows Machine

When you install the Branch Office Manager on the Windows machine, it creates the `OracleDatabaseLite` user account with the minimum set of privileges required to execute the Oracle Database Lite software. This prevents Oracle Database Lite Branch Office executing under the `SYSTEM` account, which has broad privileges within the system and can make the system vulnerable.

Both the 'Oracle Lite Multiuser Service' is created as well as the normal Web-to-Go service executes under the privileges of the `OracleDatabaseLite` user. The Oracle Lite Multiuser Server enables remote clients to connect to the Oracle Lite database.

Normally, when installed, the password for the `OracleDatabaseLite` user is randomly generated during the setup. You can either pre-configure this password before the Branch Office installation or modify it after the configuration using one of the following methods:

- Section 3.5.3.1, "Pre-Configuring OracleDatabaseLite User Password Before Branch Office Installation"
- Section 3.5.3.2, "Modifying Existing OracleDatabaseLite User Password After Branch Office Installation"

3.5.3.1 Pre-Configuring OracleDatabaseLite User Password Before Branch Office Installation

To pre-configure the `OracleDatabaseLite` user password, modify the password attribute in the `BOS.INF` file located in the *<ORACLE_HOME>*`\mobile_oc4j\j2ee\mobileserver\applications\mobileserver\setup\dmc` directory on the Mobile Server, which is as follows:

```
<user name='OracleDatabaseLite' password='aaaa'>
 <file>$OS_DIR$\odbc.ini</file>
 <file>$OS_DIR$\polite.ini</file>
 <registry>MACHINE\SOFTWARE\ODBC\ODBCINST.INI</registry>
</user>
```

3.5.3.2 Modifying Existing OracleDatabaseLite User Password After Branch Office Installation

To modify the password of an existing Branch Office installation, perform the following on the Branch Office Windows machine:

1. Modify the password with the Computer Management MMC console, which you can bring up with Control Panel->Administrative Tools-> Computer Management.
2. Set the new password in the Oracle Lite service using the Windows Service control panel.

3.6 System Requirements For UNIX Systems

Before you install, you must check to see that you have the correct hardware and software necessary for your UNIX machines that use Oracle Database Lite.

The requirements for both are detailed in the following sections:

- Section 3.6.1, "System Requirements For Linux"
- Section 3.6.2, "System Requirements For Sun SPARC-Based Systems"
- Section 3.6.3, "System Requirements For AIX-Based Systems"
- Section 3.6.4, "System Requirements For HP-Based Systems"
- Section 3.6.5, "Software Requirements For All UNIX Systems"

3.6.1 System Requirements For Linux

Table 3–5 lists the minimum hardware requirements for OracleAS Linux-based systems.

Table 3–5 Minimum Hardware Requirements for UNIX Systems

Item	Requirement
Linux	3 GHz, 1 GB RAM
Disk space for Linux	1 GB
display	256 color display

Table 3–5 (Cont.) Minimum Hardware Requirements for UNIX Systems

Item	Requirement
Memory	512 MB
TMP or swap space	1 GB

The following is the minimum hardware requirements for OracleAS Linux-based systems. Use the following operating system requirements for installing Oracle Database Lite as a standalone product:

Table 3–6 Operating System Requirements for Redhat Enterprise Linux AS 3.0

Item	Requirement
Operating System	Redhat Enterprise Linux Advanced Server 3.0 The minimum supported kernel and glibc versions are 2.4.21-4-EL and glibc-2.3.2-95.3
Patches	Apply patch 3006854, which is downloadable from Oracle*MetaLink*.
Software packages	pdksh-5.2.14

Note: There are no additional requirements for Redhat Linux 4.0 or Suse 9.

For the latest information on operating system requirements, refer to Oracle*MetaLink* at the following Web site:

`http://metalink.oracle.com`

3.6.2 System Requirements For Sun SPARC-Based Systems

Table 3–7 lists the minimum hardware requirements for OracleAS Sun SPARC-based systems.

Table 3–7 Minimum Hardware Requirements for UNIX Systems

Item	Requirement
Sun SPARC	Sun SPARC Ultra 1 or higher, 300 MHz or better
Disk space for Sun SPARC	1 GB
Memory	512 MB
TMP or swap space	1 GB
display	256 color display

Use the following operating system requirements for installing Oracle Database Lite as a standalone product:

Table 3–8 Operating System Requirements for Sun SPARC-Based Systems

Item	Requirement
Operating System	Sun Solaris 8.0, 9.0, or 10.0

Table 3–8 (Cont.) Operating System Requirements for Sun SPARC-Based Systems

Item	Requirement
Window Manager	Use any supported Sun Solaris window manager that supports Motif.

For the latest information on operating system requirements, refer to Oracle*MetaLink* at the following Web site:

```
http://metalink.oracle.com
```

3.6.3 System Requirements For AIX-Based Systems

Table 3–9 lists the minimum hardware requirements for OracleAS AIX-based systems.

Table 3–9 Minimum Hardware Requirements for UNIX Systems

Item	Requirement
AIX	All AIX-compatible processors (64-bit)
Disk space for AIX	1 GB
Memory	512 MB
TMP or swap space	1 GB
display	256 color display

Use the following operating system requirements for installing Oracle Database Lite as a standalone product:

Table 3–10 Operating System Requirements for AIX-Based Systems

Item	Requirement
Operating System	AIX 5L with either version 5.2 including Maintenance Level 4 or version 5.3
Window Manager	Use any supported IBM AIX window manager that supports Motif, such as `dtwm`, `twm`, and `olwm`.

For the latest information on operating system requirements, refer to Oracle*MetaLink* at the following Web site:

```
http://metalink.oracle.com
```

3.6.4 System Requirements For HP-Based Systems

Table 3–11 lists the minimum hardware requirements for OracleAS HP-based systems.

Table 3–11 Minimum Hardware Requirements for UNIX Systems

Item	Requirement
HP	HP 9000 Series HP-UX processor for HP-UX 11.0 (64-bit)
Disk space for HP	1 GB
Memory	512 MB
TMP or swap space	1 GB
display	256 color display

Use the following operating system requirements for installing Oracle Database Lite as a standalone product:

Table 3–12 Operating System Requirements for HP-Based Systems

Item	Requirement
Operating System	HP-UX 11.0 (64-bit)
Quality Pack	For HP-UX 11.0 (64 bit), install the Quality Pack Sept 2002 (QPK11000 B.11.00.58.5) or higher.
Patches	PHKL_27813 s700_800 11.00 POSIX AIO;getdirentries;MVFS;rcp;mmap/IDS patch
Window Manager	X Windows must be installed on the system from where the Installer is run. Use any supported X Windows server with support for Motif, such as `dtwm`, `twm`, and `mwm`.

Table 3–13 Operating System Requirements for HP-Based Systems

Item	Requirement
Operating System	HP-UX 11*i* PA-RISC Version 11.11 and 11.23
Quality Pack	Dec 2001 Consolidate Patches (Dec01GQPK11i_Aux_Patch B.03.02.06) or higher.
Patches	PHNE_28089 s700_800 11.11 cumulative ARPA Transport patch
Window Manager	X Windows must be installed on the system from where the Installer is run. Use any supported X Windows server with support for Motif, such as `dtwm`, `twm`, and `mwm`.

For the latest information on operating system requirements, refer to Oracle*MetaLink* at the following Web site:

`http://metalink.oracle.com`

3.6.5 Software Requirements For All UNIX Systems

For all UNIX systems, you must have the JDK installed. See Section 3.3.3, "JDK Platform Support" for the versions supported.

3.7 Recommended System Configuration For UNIX

This section describes the following system configurations for UNIX-based systems:

- Section 3.7.1, "Setting UNIX Environment Variables"
- Section 3.7.2, "Pre-Install Requirements Before Installing the MDK on LINUX"
- Section 3.7.3, "Creating UNIX Accounts and Groups"
- Section 3.7.4, "Configuring Kernel Parameters and Shell Limits for UNIX"

3.7.1 Setting UNIX Environment Variables

The following sections describe the environment variables that must be set before starting Oracle Universal Installer:

- Section 3.7.1.1, "ORACLE_HOME"
- Section 3.7.1.2, "JAVA13_HOME"

- Section 3.7.1.3, "JAVA_HOME and PATH"
- Section 3.7.1.4, "DISPLAY"
- Section 3.7.1.5, "TMP and TMPDIR"

> **Note:** Ensure your PATH, CLASSPATH and library path environment variables do not exceed 1024 characters. Longer values might generate errors such as "Word too long" during installation.
>
> Refer to Table 3–14 for the name of the library path environment variable for your platform.

Table 3–14 lists the names of the library path environment variables for each platform.

Table 3–14 Library Path Environment Variable

Platform	Library Path Environment Variable
Linux	LD_LIBRARY_PATH
HP	SHLIB_PATH and LD_LIBRARY_PATH
AIX	LIBPATH

3.7.1.1 ORACLE_HOME

The Oracle Home directory is the `root` directory in which Oracle software is installed. The `CLASSPATH` contains the entire path you enter for *`ORACLE_HOME`*; thus, the length of *`ORACLE_HOME`* effects the length of the `CLASSPATH`. There are limits to the length of the values of the `CLASSPATH` values with JDK. If the *`ORACLE_HOME`* path is long, this will result in a long `CLASSPATH` and might cause problems running Oracle Database Lite. The workaround is to shorten *`ORACLE_HOME`* path.

The *`ORACLE_HOME`* environment variable must be set before starting the installer, which must be set to the directory where you want to install.

3.7.1.2 JAVA13_HOME

The *`JAVA13_HOME`* environment variable must be set to the directory where the Java Development Kit has been installed. If the JDK has not been installed, please install it before proceeding with the installation. For the appropriate JDK version, see Section 3.3.3, "JDK Platform Support".

Table 3–15 provides examples for the location where the JDK could be installed on the system.

Table 3–15 JAVA13_HOME Environment Variables

Platform	Sample *JAVA13_HOME* Environment Variable
HP	`/opt/java1.5`
AIX	`/usr/java131`

3.7.1.3 JAVA_HOME and PATH

Initialize these environment variables, as follows:

```
export JAVA_HOME=$JAVA13_HOME
export PATH=$JAVA_HOME/bin:$ORACLE_HOME/bin:$PATH
```

3.7.1.4 DISPLAY

Set the *DISPLAY* environment variable to refer to the X Server that will display the installer and Oracle Database Lite. The format of the *DISPLAY* environment variable is:

```
hostname:display_number.screen_number
```

For example, set the *DISPLAY* environment variable, as follows:

```
setenv DISPLAY myhost:0.0
```

> **Note:** In order for the Oracle Database Lite to install correctly, you must add a line to the `opmn.xml` file. This is a post-installation step that is described in Section 4.4.1, "Setup DISPLAY Variable for UNIX Systems in Oracle Application Server".

Oracle Database Lite requires a running X server to properly create graphics for the installer, Web applications, and management tools. The frame buffer X server installed with your operating system requires that you remain logged in and have the frame buffer running at all times. If you do not want to do this, then you must use a virtual frame buffer, such as X Virtual Frame Buffer (XVFB) or Virtual Network Computing (VNC).

See Also:

- Your operating system documentation for more information on the `DISPLAY` environment variable.
- Oracle Technology Network (`http://otn.oracle.com`) for further information about obtaining and installing XVFB or other virtual frame buffer solutions. Search OTN for "frame buffer".

3.7.1.4.1 Installing From a Remote Machine Setting the *DISPLAY* environment variable enables you to run the Oracle Universal Installer remotely from another workstation. On the system where you launch the Oracle Universal Installer, set *DISPLAY* to the system name or IP address of your local workstation.

> **Note:** You can use a PC X emulator to run the installer if it supports a PseudoColor color model or PseudoColor visual. Set the PC X emulator to use a PseudoColor visual, and then start the installer. Refer to the X emulator documentation for instructions on how to change the color model or visual settings.

If you get an Xlib error similar to "Failed to connect to server", "Connection refused by server," or "Can't open display" when starting the installer, then run the commands on your local workstations as listed in the following table.

Shell Types	On the Server Host Machine Where the Installer is Running	In the Session on Your Host
C Shell	prompt> setenv DISPLAY <hostname>:0.0	prompt> xhost +<hostname>
Bourne/Korn Shell	prompt> DISPLAY=<hostname>:0.0;export DISPLAY	prompt> xhost +<hostname>

3.7.1.5 TMP and TMPDIR

During installation, Oracle Universal Installer uses a temporary directory for swap space. This directory must meet the requirements listed in Section 3.6, "System Requirements For UNIX Systems" before installing Oracle Database Lite. The installation may fail if you do not have sufficient space. The installer checks for the `TMP` and `TMPDIR` environment variable to locate the temporary directory. If the `TMP` environment variable is not set, then the installer uses the `/tmp` directory. If the `TMPDIR` environment variable is not set, then the installer uses the `/var/tmp` directory. Set the `TMP` and `TMPDIR` environment variable using the commands in the following table.

C Shell	Bourne/Korn Shell
`prompt> setenv TMP full_path`	`prompt> TMP=full_path;export TMP`
`prompt> setenv TMPDIR full_path`	`prompt> TMPDIR=full_path;export TMPDIR`

3.7.2 Pre-Install Requirements Before Installing the MDK on LINUX

Of all of the UNIX platforms, the MDK is only support on the Linux platform. But before installing the MDK on your Linux system, modify the following environment variables:

Table 3–16 Linux Environment Variable

Linux Environment Variable	Description
`JDKDIR`	Point to the JDK installation directory
`LD_LIBRARY_PATH`	`/usr/lib:$JDKDIR/jre/lib/i386:$JDKDIR/jre/lib/i386/server:$LD_LIBRARY_PATH`
`OLITE_HOME`	`$ORACLE_HOME/mobile/sdk`
`CLASSPATH`	`.:$ORACLE_HOME/mobile/sdk/bin/olite40.jar`
`PATH`	`$JDKDIR/bin:$OLITE_HOME/bin:$PATH`

3.7.3 Creating UNIX Accounts and Groups

The following UNIX account and groups are required for the installation process:

- Section 3.7.3.1, "UNIX Group Name For the Oracle Universal Installer Inventory"
- Section 3.7.3.2, "UNIX Account to Own Oracle Software"

3.7.3.1 UNIX Group Name For the Oracle Universal Installer Inventory

Use the `admintool` or `groupadd` utility to create a group name. In the following text the group name is `oinstall`. The `oinstall` group will own Oracle Universal Installer's `oraInventory` directory. The `oracle` user account that runs the installer must have the `oinstall` group as its primary group and `dba` as its secondary group.

For more information on these utilities, refer to your operating system documentation.

3.7.3.2 UNIX Account to Own Oracle Software

The `oracle` account is the UNIX account that owns Oracle software for your system. You must run Oracle Universal Installer from this account.

Create an `oracle` account with the properties listed in Table 3–17.

Table 3–17 Oracle Account Properties

Variable	Property
Login Name	Select any name to access the account. This document refers to the name as the `oracle` account.
Group Identifier	The `oinstall` group.
Home Directory	Select a home directory consistent with other user home directories.
Login Shell	The default shell can be either the C, Bourne, or Korn shell.

Note: Do not use `root` as the `oracle` account.

3.7.4 Configuring Kernel Parameters and Shell Limits for UNIX

Depending on your operating system, see one of the following sections for information on checking the software requirements:

- Section 3.7.4.1, "Configuring Shell Limits and System Configuration Parameters on AIX"
- Section 3.7.4.2, "Configuring Kernel Parameters on HP-UX"
- Section 3.7.4.3, "Configuring the Kernel Parameters on Linux"
- Section 3.7.4.4, "Set Shell Limits for the User oracle"

3.7.4.1 Configuring Shell Limits and System Configuration Parameters on AIX

On AIX systems, you do not need to configure kernel parameters. However, Oracle recommends that you set shell limits and system configuration parameters as described in this section.

3.7.4.1.1 Configuring Shell Limits for AIX Systems Verify that the shell limits shown in the following table are set to the values shown. The procedure following the table describes how to verify and set the values.

Table 3–18 Shell Limits

Shell Limits as shown in smit	Recommended Value
Soft FILE size	-1 (Unlimited)
Soft CPU time	-1 (Unlimited) -- this is the default value
Soft DATA segment	-1 (Unlimited)
Soft STACK size	-1 (Unlimited)

To view or change the current value specified for these shell limits, follow these steps:

1. Enter the `smit` command: `#smit chuser`
2. In the **User Name** field, enter the user name of the Oracle software owner, such as `oracle`.
3. Scroll down the list and verify that the value shown for the soft limits listed in the previous table is -1. If necessary, modify the existing value to be -1.
4. Press F10 to exit.

3.7.4.1.2 Configure System Configuration Parameters for AIX Verify that the **Maximum number of processes** allowed for each user is set to 2048 or greater. The procedure following the table describes how to verify and set the value.

1. Enter the `smit` command: `#smit chgsys`
2. Verify that the value shown for **Maximum number of PROCESSES** allowed per user is greater than or equal to 2048.
3. Press F10 to exit.

Ensure that the `ARG_MAX` setting is set to the maximum value for AIX 5L:

1. Check the `ARG_MAX` setting, as follows:

```
prompt> getconf ARG_MAX
```

2. If the value is less than 524288, then run the following command as the root `user`:

```
#chdev -l sys0 -a ncargs=128
```

3.7.4.2 Configuring Kernel Parameters on HP-UX

Verify that the kernel parameters shown in the following table are set either to the formula shown or to values greater than or equal to the recommended value shown. The procedure following the table describes how to verify and set the values.

Table 3–19 Kernel Parameters for HP-UX

Parameter	Recommended Formula or Value
nfile	3000
nproc	2048

To view or modify the current value or formula specified for these kernel parameters, do the following:

1. Optionally, set the `DISPLAY` environment variable to specify the display of the local system, as follows:
   - Bourne, Bash, or Korn shell:

```
$ DISPLAY=localhost:0.0 ; export DISPLAY
```

   - C shell:

```
$ setenv DISPLAY localhost:0.0
```

2. Start System Administration Manager (SAM): `#/usr/sbin/sam`
3. Choose the **Kernel Configuration** area, then choose the **Configurable Parameters** area.
4. Check and possibly modify the value or formula specified for each of these parameters.
5. Exit from SAM.
6. If you modified the value specified for any parameter, then reboot the system with the following: `# /sbin/shutdown -r -now`
7. If necessary, when the system restarts, log in and switch the user to `root`.

3.7.4.3 Configuring the Kernel Parameters on Linux

Verify that the kernel parameters shown in the following table are set either to the formula shown, or to values greater than or equal to the recommended value shown. The procedures following the table describe how to verify and set the values.

Table 3–20 Kernel Parameters on Linux

Parameter	Value	File
`file-max`	131072	`/proc/sys/fs/file-max`

To view or modify the current value specified for these kernel parameters, do the following:

1. Enter the following command to view the current value of the `file-max` kernel parameter:

   ```
   # /sbin/sysctl -a | grep file-max
   ```

2. To modify the value, do the following:

   a. Create or edit the `/etc/sysctl.conf` file and add the following.

      ```
      fs.file-max = 131072
      ```

      By specifying the values in the `/etc/sysctl.conf` file, they persist when you reboot the system.

   b. Change the current values of the kernel parameter with the following command:

      ```
      # /sbin/sysctl -p
      ```

      Review the output from this command to verify that the values are correct. If the values are incorrect, then perform these steps again.

   c. On UnitedLinux only, enter the following command to cause the system to read the `/etc/sysctl.conf` file when it reboots:

      ```
      # chkconfig boot.sysctl on
      ```

3.7.4.4 Set Shell Limits for the User oracle

To improve the performance of the software on Linux systems, you must increase the following shell limits for the oracle user, depending on the user's default shell:

Table 3–21 Shell Limits for Linux Systems

Bourne or Bash Shell Limit	Korn Shell Limit	C or tcsh Shell Limit	Hard Limit
nofile	nofile	descriptors	16384
noproc	processes	maxproc	16384

To increase the shell limits, do the following:

1. Add the following lines to the `/etc/security/limits.conf` file, where the arrow (->) represents the tab character:

   ```
   * -> -> soft -> nproc -> -> 2047
   * -> -> hard -> nproc -> -> 16384
   * -> -> soft -> nofile -> -> 2047
   * -> -> hard -> nofile -> -> 16384
   ```

2. Add the following line to the `/etc/pam.d/login` file, if it does not already exist:

```
session required /lib/security/pam_limits.so
```

3. Depending on the `oracle` user's default shell, make the following changes to the default shell start-up file:

   - For the Bourne, Bash, or Korn shell, add the following lines to the `/etc/profile` file:

```
if [ $USER = "oracle" ]; then
   if [ $SHELL = "/bin/ksh" ]; then
     ulimit -p 16384
     ulimit -n 16384
   else
     ulimit -u 16384 -n 16384
   fi
fi
```

   - For the C or `tcsh` shell, add the following lines to the `/etc/csh.login` file:

```
if ( $USER == "oracle" ) then
    limit maxproc 16384
    limit descriptors 16384
endif
```

3.8 System Requirements For Windows Mobile Clients

Before you install, you must check to see that you have the correct hardware and software necessary for Windows Mobile Clients. The requirements for both are detailed in the following sections:

- Section 3.8.1, "Hardware Requirements For Windows Mobile Clients"
- Section 3.8.2, "Software Requirements For Windows Mobile Clients"

3.8.1 Hardware Requirements For Windows Mobile Clients

The hardware requirements for Windows Mobile Clients are described in the following table:

Table 3–22 Hardware Requirements for Mobile Clients

Component	Hardware Requirements for this Component
Mobile Client for Win32	CPU: Pentium 4, 1 GB Disk Space: 30 MB RAM: 256 MB
Mobile Client for Web-to-Go	CPU: Pentium 4, 1 GB Disk Space: 70 MB RAM: 512 MB
Mobile Client for Windows CE/Windows Mobile	CPU: ARM-based processor or Emulator Storage Space: 8 MB Free program memory: 16 MB It matters what external memory media and file system you use. See Section 3.8.1.1, "What File System and External Memory Media Should You Use for Windows CE?" for more information.

Table 3–22 (Cont.) Hardware Requirements for Mobile Clients

Component	Hardware Requirements for this Component
Branch Office	CPU: Pentium 4, 1 GB Disk Space: 70 MB RAM: 512 MB

3.8.1.1 What File System and External Memory Media Should You Use for Windows CE?

When you are using a WinCE device, you will probably use some form of external memory media, such as Compact Flash or SD cards. Of these two, the SD card is more reliable in its method of connecting into the device. We have seen some issues of database corruption when using Compact Flash cards, since the card can be removed in the middle of a write without notice.

When you do choose a media, you can also decide on the type of file system you use. We strongly recommend that you use the Transaction-Safe FAT (TFAT) file system over the FAT system, which is the default and more widely used. FAT has some unreliability that was not noticeable in a laptop or desktop situation. However, this unreliability in its ability to flush its buffers fully when writing out to the removable memory does not handle well when the media is removed in the middle of the transaction. Thus, the shortcomings of the FAT file system is being seen more in the WinCE environment with removable memory. The TFAT design provides transaction-safety for data storage. That is, the data that is being written out to the removable media is either completely committed or rolled-back. Thus, the TFAT file system is highly recommended for any removable data—especially for the Oracle Lite database. There is a performance consideration for using the TFAT file system. It is slower than FAT, but also more reliable.

You need to decide whether performance or reliability is your priority. The following is the order of reliability with 1 being the least reliable and 3 being the most reliable:

1. Compact Flash media with the FAT file system.
2. SD card media with the FAT file system.
3. Compact Flash or SD card media with the TFAT file system.

Note: See the Microsoft Web site for information on how to create the TFAT file system.

3.8.2 Software Requirements For Windows Mobile Clients

The software requirements for Windows Mobile Clients are described in the following sections:

- Section 3.8.2.1, "Certified Operating Systems and Other Software Requirements"
- Section 3.8.2.2, "Supported and Certified Technologies for Windows Mobile Clients"
- Section 3.8.2.3, "Supported Platforms for Oracle Database Lite WinCE"
- Section 3.8.2.4, "Windows Mobile Client Notes"

3.8.2.1 Certified Operating Systems and Other Software Requirements

Table 3–23 Software Requirements for Windows Mobile Clients

Mobile Client	Certified Operating System	Other Software Requirements
Mobile Client for Win32	Windows 2000 Professional Edition with Service Pack 4, Windows XP Professional Edition with Service Pack 2, Windows 2003, or Microsoft Windows XP Tablet PC	If using any Java APIs, including synchronization or JDBC, use JRE 5.0 If implementing any .NET applications, use Compact Framework .NET 1.1 or 2.0
Mobile Client for Web-to-Go	Windows 2000 Professional Edition with Service Pack 4, Windows XP Professional Edition with Service Pack 2, Windows 2003, or Microsoft Windows XP Tablet PC	
Mobile Client for Windows CE	Windows CE 5.0 See Section 3.8.2.3, "Supported Platforms for Oracle Database Lite WinCE" for full details.	If using JDBC, use either IBMJ9 or the CrEme JDK version 4.1 from `NSIcom.com`. ActiveSync version 3.8 or later. Microsoft .NET Compact Framework 1.0
Mobile Client for Windows Mobile	Windows Mobile 2003 Windows Pocket PC 2003 Windows Mobile 2003, 2nd edition Windows Pocket PC 2003, 2nd edition See Section 3.8.2.3, "Supported Platforms for Oracle Database Lite WinCE" for full details.	ActiveSync version 3.8 or higher. Microsoft .NET Compact Framework 1.1 If using JDBC, use either IBMJ9 or the CrEme JDK version 4.1 from `NSIcom.com`.
	Windows Mobile 5.0	ActiveSync version 4.1 or higher. Microsoft .NET Compact Framework 1.1 or 2.0 If using JDBC, use either IBMJ9 or the CrEme JDK version 4.1 from `NSIcom.com`.
OC4J	Windows 2000 Professional Edition with Service Pack 4, Windows XP Professional Edition with Service Pack 2, Windows 2003, or Microsoft Windows XP Tablet PC	
Branch Office	Windows 2000 Professional Edition with Service Pack 4, Windows XP Professional Edition with Service Pack 2, Windows 2003, or Microsoft Windows XP Tablet PC	

You should install all of the patches required for the JDK for the Windows operating system. This is constantly under review and published on the JDK download page on the Sun Microsystems Web site.

3.8.2.2 Supported and Certified Technologies for Windows Mobile Clients

The following are the supported and certified technologies for Mobile clients:

Table 3–24 Supported and Certified Technologies for Windows Mobile Clients

Mobile Client	Supported Technologies	Certified Technologies
Mobile Client for Win32	■ Sun Microsystems Java Runtime Edition 5.0 ■ JDBC 1.2 ■ ADO.Net 1.1 – requires Microsoft .Net Compact Framework 1.1 or 2.0 ■ ODBC 2.0 ■ SQL92	
Mobile Client for Web-to-Go	■ Sun Microsystems Java Runtime Edition 5.0 ■ Java Servlets 2.2 ■ JDBC 1.2 ■ Oracle Java Server Pages Version 9.0.2.0.0 ■ Oracle UIX version 2.1.7 ■ Oracle XML Parser 9.0.3.0.0	Struts version 1.1 is certified for use with Oracle Database Lite Web-to-Go.
Mobile Client for Windows CE	■ ODBC 2.0 ■ JDBC 1.2 ■ ADO.Net 1.1 (Requires Microsoft Compact .Net Framework 1.0 + Service Pack 2) or 2.0 ■ Microsoft ActiveSync version 3.8 or for Windows CE 5.0, use Microsoft ActiveSync version 4.1 or higher.	Oracle Database Lite is certified with the following JVMs on Windows Mobile 2003 Second Edition: ■ IBM J9 Websphere Everyplace Micro Environment for Windows Mobile 2003 ARM Personal Profile ■ Creme JVM 4.1, which can be obtained at `http://www.nsicom.com` NOTE: Java Stored Procedures are not supported on Windows CE.
OC4J	■ Sun Microsystems Java Runtime Edition 5.0 ■ Java Servlets 2.4 ■ JDBC 1.2 ■ Oracle Java Server Pages Version 10.1.3.0.0 ■ Oracle UIX version 2.2.24 ■ Oracle XML Parser 10.1.3.0.0	■ Struts version 1.1 ■ JDeveloper 10.1.3
Branch Office		Struts version 1.1

3.8.2.3 Supported Platforms for Oracle Database Lite WinCE

Table 3–25 provides the full list of supported platforms for Pocket PC and Windows Mobile:

Table 3–25 Pocket PC and Windows Mobile Supported Platforms

Product Name	WinCE Version	Chipsets	Oracle Lite Client CAB file download from Setup page
Pocket PC 2003 Windows Mobile 2003	4.20.1081	ARMV4	Oracle Lite PPC2003 ARMV4, which uses the `<language>\ppc2003\armv4\olite.cab`
Windows Mobile 2003 2nd Edition	4.21.1088	ARMV4	Oracle Lite PPC2003 ARMV4, which uses the `<language>\ppc2003\armv4\olite.cab`
Windows Mobile 5	5.0	ARMV4I	Oracle Lite PPC50 ARMV4I, which uses the `<language>\ppc50\armv4i\olite.cab`

If you wish to use the SDK CAB files, which are included in the MDK, for the JAVA, MSQL and utilties, then you can install the client SDK CAB files, which are listed below. See Section 5.2.2, "Use SDK CAB Files for Client Install for Java, MSQL, and Utility Support" for full details.

Table 3–26 Pocket PC and Windows Mobile Supported Platforms

Product Name	WinCE Version	Chipsets	Client CAB file available from MDK
Windows Mobile 2003	4.20.1081	ARMV4	`olite.<language>.ppc2003.armv4_sdk.cab`
Windows Mobile 2003, 2nd edition	4.21.1088	ARMV4	`olite.<language>.ppc2003.armv4_sdk.cab`
Windows Mobile 5	5.0	ARMV4I	`olite.<language>.ppc50.armv4i_sdk.cab`
Windows CE Standard SDK 5.0 Edition	5.0	ARMV4I	`olite.<language>.stdsdk500.armv4i_sdk.cab`
		X86	`olite.<language>.stdsdk500.x86_sdk.cab`

3.8.2.4 Windows Mobile Client Notes

For Mobile Client Web-to-Go, Win32, Branch Office and BC4J:

- Internet Explorer 6.0 is required when using SSL to synchronize with the Mobile Server.
- The product requires the Microsoft C Runtime Library 7.1 (`msvcrt71.dll`), which you can download off the Microsoft site or other sites on the Web.

3.9 System Requirements for Linux Mobile Clients

The Mobile Server installation includes the following Mobile Clients for Linux:

- Mobile Client for Linux x86
- Mobile Client for Linux Web-to-Go

> **Note:** The Device Manager agent (`DMagent`) must be running to successfully uninstall the Linux Client.

Before you install, you must check to see that you have the requirements necessary for Linux Mobile Clients. The requirements for both are detailed in the following sections:

- Section 3.9.1, "Certified Platforms and Supported Technologies for Linux Mobile Clients"
- Section 3.9.2, "Software and Hardware Requirements for Linux Mobile Clients"

- Section 3.9.3, "Setting Environment Variables Before Installing the Linux Mobile Client"

3.9.1 Certified Platforms and Supported Technologies for Linux Mobile Clients

Table 3–27 provides the full list of certified and supported platforms for Linux Mobile clients:

Table 3–27 Certified Platforms and Supported Technologies for Linux Mobile Clients

Mobile Client	Certified Platforms	Supported Technologies
Oracle Lite Linux Web	Red Hat Enterprise Linux AS release 4	■ JavaSoft Java Runtime Edition 1.4.2 ■ Java Servlets 2.2 ■ JDBC 1.2 ■ Oracle Java Server Pages Version 9.0.2.0.0 ■ Oracle UIX version 2.1.7 ■ Oracle XML Parser 9.0.3.0.0
Oracle Lite Linux x86	Red Hat Enterprise Linux AS release 4	■ JavaSoft Java Runtime Edition 1.4.2 ■ JDBC 1.2 ■ ODBC 2.0 ■ SQL92

Note for Oracle Lite Linux WEB: Mozilla version 1.7.x is the preferred internet browser on Linux.

3.9.2 Software and Hardware Requirements for Linux Mobile Clients

Table 3–28 provides the software and hardware requirements for Linux Mobile clients:

Table 3–28 Software and Hardware Requirements for Linux Mobile Clients

Mobile Client	Hardware Requirement	Software Requirements
Mobile Client for Linux Web-to-Go	CPU: Pentium III 360 MHz Disk Space: 40 MB RAM: 128 MB	Red Hat Enterprise Linux AS R4 JDK 1.4.2
Mobile Client for Linux x86	CPU: Pentium III 360 MHz Disk Space: 30 MB RAM: 128 MB	Red Hat Enterprise Linux AS R4 JDK 1.4.2

3.9.3 Setting Environment Variables Before Installing the Linux Mobile Client

Set the following environment variables:

- Set `OLITE_HOME` to where Oracle Database Lite is installed, such as `/home/<user>/olite`
- Set `JAVA_HOME` to the Java installation directory
- Add the following to the `LD_LIBRARY_PATH`
  `$JAVA_HOME/jre/lib/i386`
  `$JAVA_HOME/jre/lib/i386/server`
  `$OLITE_HOME/bin`

- Add `$OLITE_HOME/bin` to the `PATH`

3.10 System Requirements for Symbian Devices

This section details requirements if you are preparing to use Symbian devices in your Oracle Database Lite solution. Your development environment must include Oracle Database Lite 10*g* as the encompassing platform. For developing native applications with Oracle Database Lite 10*g* on a Symbian platform, the following devices are supported:

The following devices are supported for the Symbian 7.x platform:

- Nokia 6620
- Nokia 9500
- Motorola M1000 / A1000
- Sony Ericsson P910

The following devices are supported for the Symbian 8.x platform:

- Nokia 6630, which is also known in Japan as V702NK
- Nokia 6680

The following are the supported development environments:

- Microsoft Visual Studio 6.0
- Microsoft Visual Studio .Net

3.11 Mounting the Installation CD-ROM For UNIX Systems

Refer to these mounting procedures during installation as necessary:

- Section 3.11.1, "Mounting CD-ROMs For AIX"
- Section 3.11.2, "Mounting CD-ROMs For HP"
- Section 3.11.3, "Mounting CD-ROMs For Linux"

3.11.1 Mounting CD-ROMs For AIX

Mount the disk to begin the installation. Follow these steps to mount the Oracle Database Lite CD-ROM manually:

1. Place the Oracle Database Lite CD-ROM Disk in the CD-ROM drive.
2. Log in as the `root` user and create a CD-ROM mount point directory, if one does not already exist, by using the following commands:

   ```
   $ su root
   # mkdir cdrom_mount_point_directory
   ```

3. Determine the CD-ROM device name by entering the following command:

   ```
   # lsdev -Cc cdrom
   ```

   The output should be similar to the following:

   ```
   cd0 Available 10-60-00-4, 0 SCSI Multimedia CD-ROM Drive
   ```

4. Mount the CD-ROM drive on the mount point directory by entering the following commands:

```
# mount options device_name cdrom_mount_point_directory
```

5. Exit the root account:

```
# exit
```

Example 3–1 shows how to mount the CD-ROM manually for AIX. In the following example, `/dev/cd0` is the CD-ROM device and `/cdrom` is the mount point.

Example 3–1 Mounting the CD-ROM manually for AIX

```
$ su root
# mkdir /cdrom
# mount -rv cdrfs /dev/cd0 /cdrom
# exit
```

Caution: Do not run the Installer while the CD-ROM directory is the current directory or you will be unable to unmount the current CD-ROM when prompted to do so.

3.11.2 Mounting CD-ROMs For HP

Mount the disk to begin the installation. Follow these steps to mount the Oracle Database Lite CD-ROM manually:

1. Place the Oracle Database Lite CD-ROM Disk in the CD-ROM drive.
2. Log in as the `root` user and create a CD-ROM mount point directory, if one does not already exist, by using the following commands:

```
$ su root
# mkdir cdrom_mount_point_directory
```

3. Determine the CD-ROM device name by entering the following command:

```
# ioscan -fun -C disk
```

The output should be similar to the following:

```
disk  10 10/12/5.2.0  sdisk  CLAIMED  DEVICE  TOSHIBA CD-ROM
XM-5701TA /dev/dsk/c4t2d0   /dev/rdsk/c4t2d0
```

4. If there is not already an entry in the `/etc/pfs_fstab` file for your CD-ROM device, you must add one. As the `root` user, use a text editor to add a line, in the following format, to the `/etc/pfs_fstab` file:

```
device_file mount_point filesystem_type translation_method
```

In the preceding format, the first entry is the CD-ROM device, the second entry is the mount point, and the third entry indicates that the CD-ROM to be mounted is in ISO9660 format with Rockridge extensions.

The `device_file` in this example is `/dev/dsk/c4t2d0`. For a CD-ROM device with the path `/dev/dsk/c4t2d0`, you would enter the following:

```
/dev/dsk/c4t2d0 /SD_CDROM pfs-rrip xlat=unix 1 0
```

5. Log in as the `root` user with the following command:

```
$ su root
```

6. Enter the following commands:

```
# nohup /usr/sbin/pfs_mountd &
# nohup /usr/sbin/pfsd &
```

7. Place the Oracle Database Lite CD-ROM Disk in the CD-ROM drive and mount the CD-ROM by entering the following command:

```
# /usr/sbin/pfs_mount /SD_CDROM
```

8. Log out of the `root` account.

```
# exit
```

Follow these steps to unmount the CD-ROM:

1. Change to your system's root directory and log in as the `root` user:

```
$ cd /
$ su root
```

2. To unmount the CD-ROM, enter the following command:

```
# /usr/sbin/pfs_umount /SD_CDROM
```

3. Remove the CD-ROM from the CD-ROM drive:

```
# /usr/sbin/pfs_umount /SD_CDROM
```

3.11.3 Mounting CD-ROMs For Linux

Use the following instructions to mount the CD-ROM for Linux.

- Section 3.11.3.1, "Mounting CD-ROMs For Linux With Auto Mounting Software"
- Section 3.11.3.2, "Mounting CD-ROMs For Linux Manually"

3.11.3.1 Mounting CD-ROMs For Linux With Auto Mounting Software

Mount the disk to begin the installation. If you are using auto mounting software, the CD-ROM is mounted automatically to the directory specified in your auto mount configuration when you insert it into the CD-ROM drive.

To check whether you have auto mounting software, enter the following command:

```
$ ps -aux | grep automount
```

If you have auto mounting software, the output must be similar to the following:

```
root 628 0.0 0.2 1148 588 ? S 17:32 0:00 /usr/sbin/automount /misc file
/etc/auto.misc
```

In the preceding output, the `/etc/auto.misc` section defines the directory under the `/misc` file where the CD-ROM will be mounted.

- If the auto mounting software is running and configured properly, the CD-ROM is mounted automatically.
- If no lines are returned, the auto mounting software is not running, and you will have to mount the CD-ROM manually. Proceed to Section 3.11.3.2.

3.11.3.2 Mounting CD-ROMs For Linux Manually

To mount the Oracle Database Lite CD-ROM manually, use the following steps:

1. Place the Oracle Database Lite CD-ROM Disk in the CD-ROM drive.

2. Log in as the `root` user and, if necessary, create a CD-ROM mount point directory by using the following commands:

```
$ su root
# mkdir cdrom_mount_point_directory
```

3. Mount the CD-ROM drive on the mount point directory by using the following commands:

```
# mount options device_name cdrom_mount_point_directory
```

4. Exit the `root` account.

```
# exit
```

If you are unsure of the correct device name, consult your system administrator. Typically, the device name is `/dev/cdrom`.

Example 3–2 shows how to mount the CD-ROM manually for Linux.

Example 3–2 Mounting the CD-ROM For Linux Manually

```
$ su root
# mkdir /cdrom
# mount -t iso9660 /dev/cdrom /cdrom
# exit
```

Follow these steps to unmount the CD-ROM:

1. Change directory to the root directory of your system and log in as the `root` user by using the following commands:

```
$ cd /
$ su root
```

2. Unmount the CD-ROM by entering the following command:

```
# umount cdrom_mount_point_directory
```

3. Remove the CD-ROM from the CD-ROM drive.

3.12 Setting Up Location of the Datafile on the Server

If you do not want to have the datafile for your Oracle Database Lite applications stored in the default location in the Oracle database, then modify the database configuration file to include the directory where you want your datafile stored. Configure the default directory for new data files in the `db_create_file_dest` parameter. Once updated, restart the Oracle database. This must be done before installing Oracle Database Lite. Refer to your database administration guide for details on how to modify the `db_create_file_dest` parameter.

3.13 Defining Synchronization Tablespace

By default, the synchronization tablespace is `SYNCSERVER`, and is stored in the `mobile`*`xx`*`.dbf` file in the default location for the database instance under *`ORACLE_HOME`*, where *`xx`* is a number between 1 and 25. The tablespace name, filename, and file location for the tablespace is defined in the `$ORACLE_HOME/Mobile/Server/admin/consolidator_o8a.sql` script file, which is executed during the Mobile Server installation process. So, if you want to modify the tablespace, there are a few tasks you need to perform BEFORE you install the Mobile Server.

Tablespace layout across multiple disks can improve the performance of Mobile Server data synchronization, as it reduces movement of the disk heads and improves I/O response time.

For full details on how to alter the synchronization tablespace, see Section 1.2.3, "Synchronization Tablespace Layout" in the *Oracle Database Lite Troubleshooting and Tuning Guide* for more information.

4

Installation of Oracle Database Lite

This chapter describes all of the details for you to install Oracle Database Lite, including the following:

4.1 Starting Oracle Universal Installer

Oracle Database Lite uses Oracle Universal Installer to guide you through each step of the installation process. The Oracle Universal Installer provides the following features:

- Describes installation options for Oracle Database Lite
- Detects pre-set environment variables and configuration settings
- Sets environment variables and configuration during installation
- Offers configuration options for a customized installation of Oracle Database Lite
- Deinstalls products

The Oracle Universal Installer automatically checks your computer prior to installation to verify that your system meets operational requirements. Table 4–1 lists the prerequisite checks that are performed.

Table 4–1 Oracle Universal Installer Automatic Prerequisite Checks

Prerequisite Checks	See Also
Check for enough disk space for Oracle Home installation	Section 3.6, "System Requirements For UNIX Systems"
On UNIX systems, checks for TMP and TMPDIR variable and sufficient swap space	Section 3.6, "System Requirements For UNIX Systems"

You start the Oracle Universal Installer using different methods for each type of operating system, as detailed in the following steps:

4.1.1 Starting the Oracle Universal Installer on Windows

Insert the CD-ROM labeled Oracle Database Lite and double-click `setup.exe`. The Welcome screen appears.

4.1.2 Starting Oracle Universal Installer on UNIX

Follow these steps to start Oracle Universal Installer and install Oracle Database Lite:

1. Insert the CD into the CD-ROM drive.
2. Mount the installation CD-ROM. For information on mounting the installation CD-ROM for your platform, see Section 3.11, "Mounting the Installation CD-ROM For UNIX Systems".
3. Run Oracle Universal Installer from the CD-ROM:

   Note: Ensure you are **not** logged in as the `root` user when you start the Oracle Universal Installer.

   a. Log in as the `oracle` user.

   b. Start the installer by entering the following command:

   ```
   prompt> mount_point/disk1/runInstaller
   ```

   This launches Oracle Universal Installer, which installs Database Lite.

4.1.2.1 Using the oraInventory Directory

The Oracle Universal Installer creates the `oraInventory` directory the first time it is run on a computer. The `oraInventory` directory keeps an inventory of products that the Oracle Universal Installer installs on your computer, as well as other installation information. If you have previously installed Oracle products, then you may already have an `oraInventory` directory.

When a UNIX group name is created and specified, the Oracle Universal Installer grants the specified group the permission to write to the `oraInventory` directory. If another group attempts to run the installer, then they must have permission to write to the `oraInventory` directory. If they do not have permission, then the installation will fail.

The location of `oraInventory` is defined in the `oraInst.loc` file. See Table 4–1 for the location of the `orainst.loc` file for your system.

The latest installation log file is stored in:

```
/your_base_directory/oraInventory/logs/installActiontodays_date_time.log
```

The `your_base_directory` identifier is the location for your installation files and todays_date_time is the date and time of installation. Log file names of previous installation sessions take the form `installActions`*`todays_date_time`*`.log`.

Do not delete or manually alter the `oraInventory` directory or its contents. Doing so can prevent the installer from locating products that you have installed on your system.

4.1.2.2 Location of Files on UNIX

The following table lists the location of the `oratab` and `oraInst.loc` file for each platform:

Table 4–2 File Locations for Each Platform

Platform	oratab and emtab	oraInst.loc
AIX	`/etc`	`/etc`
HP	`/etc`	`/var/opt/oracle`
Linux	`/etc`	`/etc`

4.2 Oracle Database Lite Installation Considerations

Note the following issues before you start your installation:

- Section 4.2.1, "Problems with Multi-Byte Characters in User Names"
- Section 4.2.2, "Installing Multiple Languages on a Single Solaris Machine"
- Section 4.2.3, "National Language Support for Chinese, Japanese, and Korean (CJK)"
- Section 4.2.4, "Mobile Server on a DHCP Server is Not Supported"

4.2.1 Problems with Multi-Byte Characters in User Names

You cannot have a user name with multi-byte characters.

4.2.2 Installing Multiple Languages on a Single Solaris Machine

With Oracle Database Lite installed in a Solaris environment, you cannot install multiple languages on a single Solaris machine. Instead, you must perform a separate installation for each language.

4.2.3 National Language Support for Chinese, Japanese, and Korean (CJK)

Only the Mobile Development Kit has the full National Language Support for (Traditional and Simplified) Chinese, Japanese, and Korean (CJK). All other components, including Mobile clients, support CJK without the Traditional Chinese language. However, the Simplified Chinese language is supported.

4.2.4 Mobile Server on a DHCP Server is Not Supported

Mobile Server can only be installed on a server with a static IP address; thus, the Mobile Server does not function correctly if installed on a DHCP server.

4.3 Installing Oracle Database Lite

Oracle Database Lite consists of two main components: Mobile Server and Mobile Development Kit. Each is installed separately from the Oracle Database Lite CD-ROM. The following sections describe all of your installation options:

- Section 4.3.1, "GUI Install of the Mobile Server and MDK Components"
- Section 4.3.2, "Silent Install of Mobile Server and MDK Components"

- Section 4.3.3, "Install Multiple Mobile Servers Against the Same Mobile Server Repository (Farm)"
- Section 4.3.4, "Install Oracle Database Lite on Linux"
- Section 4.3.5, "Install Oracle Database Lite on Symbian Devices"
- Section 4.3.6, "Client Install"
- Section 4.3.7, "Custom Install"

Note: For instructions on how to install Branch Office, see Chapter 10, "Manage Your Branch Office," in the *Oracle Database Lite Administration and Deployment Guide*. For instructions on how to create a Mobile Client and download an application, see Chapter 7, "Quick Start for Oracle Database Lite".

4.3.1 GUI Install of the Mobile Server and MDK Components

Once the Oracle Universal Installer is initiated, perform the following steps to install Oracle Database Lite:

Note: If you install multiple standalone Mobile Servers on a single machine or if you install Web-to-Go client for OC4J with the standalone Mobile Server on a single machine, change the RMI and JMS ports in the `rmi.xml` and the `jms.xml` files located in the `<ORACLE_HOME>\mobile_oc4j\j2ee\mobileserver\config` directory. The port numbers must not be used by another other process.

---

1. On the Welcome screen, click **Next**.
2. On the File Locations screen, enter the following:
   - In the Source field, either accept or enter a new location and name of the Oracle Database Lite `products.jar` file.
   - In the Destination field, enter the name and path of Oracle Home. The Oracle Home name CANNOT include spaces. **Important: Read the following bulleted list for the appropriate Oracle Home.**

     This is the Oracle Home where you want to install Oracle Database Lite. You may choose to install Oracle Database Lite into a new or existing Oracle Home. Take the following into consideration:

     – **If you use Mobile Server in standalone mode for your middle-tier, do not use an Oracle Home that has any version of OracleAS installed.**

     – **If you use OracleAS as your middle-tier, then you must install Oracle Database Lite into the same Oracle Home where the application server is installed.**

   Click **Next** .
3. On the Installation Types screen, choose the components and the type of installation to execute. Your options are as follows:
   - Mobile Server
   - Mobile Development Kit

- Custom

The first two are the main components of Oracle Database Lite. The custom option is for advanced users only. Each of these Install options are discussed in the following sections:

- Section 4.3.1.1, "Installation of Mobile Development Kit"
- Section 4.3.1.2, "Installation of Mobile Server"

4.3.1.1 Installation of Mobile Development Kit

Install Mobile Development Kit from the Installation Types screen by choosing the Mobile Development Kit and clicking **Next**. On the Summary screen, click **Install**. Click **Exit** to return to the installation screen.

You have now installed the Mobile Development Kit.

4.3.1.2 Installation of Mobile Server

Install the Mobile Server from the Installation Types screen, as follows:

1. Choose Mobile Server and click **Next**.
2. Provide the hostname, port, and SID for the back-end database for the Mobile Server Repository. Mobile Server installs its meta-data in the back-end database. If you are not sure, you can query the data dictionary to obtain these values. Connect as `SYSTEM` and run the following queries.

   To retrieve the SID, execute:

   ```
   select instance_name from v$instance;
   ```

   If you need to retrieve the port number, check the `listener.ora` file on the back-end database.

   Click **Next**.
3. Click **Yes** to install the Mobile Server Repository. For this option, always click **Yes** to install the repository—even if one has already been created. If you are installing a Mobile Server on another host that shares an existing repository, selecting yes updates the repository with the shared Mobile Server information. You only select **No** if instructed to for a certain environment. Click **Next**.
4. On the Mobile Server Configuration Options screen, enter the HTTP listener port for OC4J. Provide a port number for the OC4J HTTP listener if you are executing in standalone mode. The default value is 80. This screen only appears if you are using standalone. If you are using OracleAS, the default ports are used. Click **Next**.

   Note: If, after installation, you have port conflicts and need to change the port number for OC4J standalone, edit the file *ORACLE_HOME*`\mobile_oc4j\j2ee\mobileserver\config\default-web-site.xml` and modify the following with the alternate port number:

   ```
   <web-site port="80" display-name="OracleAS Containers for J2EE HTTP
   Web Site">
   ```

   ---

5. Click **Yes** or **No** indicating if you want the demo applications installed on the server.

6. On the Summary page, click **Install** to start the installation of Mobile Server, which is installed in your Oracle Home.
7. If, previously, you clicked **Yes** to installing the Mobile Server Repository, then the Repository Wizard is now launched to create and populate the repository with the Mobile Server schema.

   The Repository Wizard asks for a user name and password, which will be used to connect to the back-end database, create the schema and assign database privileges for the Mobile Server. When the Mobile Server accesses the Mobile repository, it uses the repository username/password. This defaults to the user MOBILEADMIN and the password is set during install.

   Before you provide the username, this user has to have the following privileges to create the Mobile Server repository in the back-end database:

   - CREATE TABLESPACE, CREATE USER
   - The following privileges are required with the Admin option:

     ALTER ANY TABLE, ALTER SESSION, ALTER SYSTEM, CREATE SESSION, CREATE ANY SEQUENCE, CREATE ANY VIEW, CREATE ANY TRIGGER, CREATE ANY INDEX, CREATE ANY TABLE, CREATE ANY SYNONYM, CREATE ANY PROCEDURE, CREATE PROCEDURE, CREATE SEQUENCE, CREATE SYNONYM, CREATE TABLE, CREATE VIEW, CREATE INDEXTYPE, DELETE ANY TABLE, DROP ANY SEQUENCE, DROP ANY PROCEDURE, DROP ANY VIEW, DROP ANY TABLE, DROP ANY SYNONYM, DROP ANY TRIGGER, DROP ANY INDEX, DROP ANY TABLE, SELECT ANY TABLE , SELECT ANY DICTIONARY, UPDATE ANY TABLE
   - Lastly, the SELECT_CATALOG_ROLE role is required with the Admin option.

   Note: You can set the Mobile Server privileges independently through an API. See Section 3.16, "Set DBA or Operational Privileges for the Mobile Server" in the *Oracle Database Lite Developer's Guide*.

   Click **Next**.
8. If this is a new Repository, the following message is displayed:

   ```
   No existing Repository found. A new Repository will be installed.
   ```

   Click **Next**.
9. Enter the username and password for the Mobile Server Repository. The default name is `MOBILEADMIN`. This schema contains all database objects used by the Mobile Server. Also, check whether you want to install sample applications or not. Click **Next**.

Note: If you choose to install the samples, then the following Mobile Server user accounts will be created: `JOHN`, `JANE`, `JACK`, and `JUNE`. These user accounts have default passwords; thus, you should immediately modify the passwords after the installation complete through Mobile Manager.

You should install the samples only on a development system; you should NEVER install the samples on a production Mobile Server. Doing so creates a security risk on your production environment. The default user accounts that are created as part of the samples may be cause a security risk, as they could provide unauthorized access to your production system.

10. Enter the Mobile Server administrator username and password that you will use to log into the Mobile Manager.
11. Enter the client schema username and password for the schema where the demo applications are installed. When the user accesses the user data in the Mobile Server repository, the Mobile Server authenticates the schema username and password before access is provided to the user data.
12. A summary screen appears informing you if a repository is installed or not. Click **Next**.
13. The installation screen appears. Wait until the install is completed. Click **Next**.
14. Once the repository wizard is finished, click **Finish** to leave the wizard.
15. Click **Exit** to complete the installation.
16. Review the installation log files for any errors, which are located in *`ORACLE_HOME`*`\mobile\server`. The `repository.log` file contains the log file of all general installation errors; the `samples.log` file contains a log of the sample installation errors.
17. If you are installing on top of any version of the application server, then restart the application server.

4.3.2 Silent Install of Mobile Server and MDK Components

You can execute a command-line tool that runs the installer in silent mode. That is, it installs either the Mobile Development Kit or the Mobile Server for you without having to use the Installation GUI. The following sections describe how to silently install these components on either the UNIX or Windows platforms:

- Section 4.3.2.1, "Silent Install Mobile Server or the Mobile Development Kit on UNIX"
- Section 4.3.2.2, "Silent Install for Mobile Server or Mobile Development Kit on Windows"

4.3.2.1 Silent Install Mobile Server or the Mobile Development Kit on UNIX

You can install the Mobile Server and/or the Mobile Development Kit by using the `runInstaller` command, which is on the top level of the CD or the downloaded ZIP file for the product.

There are two response files that contain variables that define how the Mobile Server or the MDK are installed, which are located in the `stage/Response` folder of the CD or ZIP file.

Note: **If you use Mobile Server in standalone mode for your middle-tier, do not use an Oracle Home that has any version of OracleAS installed.**

If you use OracleAS as your middle-tier, then you must install Oracle Database Lite into the same Oracle Home where the application server is installed.

- The response file for installing the Mobile Server is `oracle.lite.install_type_2.rsp`. The variables that you can modify in this file for defining your install are as follows:
  - Define the `ORACLE_HOME` destination directory, as follows:

    ```
    ORACLE_HOME="\private\olite\orahome\10gr2"
    ```

  - Define the name for your Oracle Home, as follows:

    ```
    ORACLE_HOME_NAME="OLITE10g"
    ```

  - Provide the hostname, port, and user for the back-end database, where the repository is to be installed, as follows:

    ```
    db_info={"my-pc.us.oracle.com","1521","myuser"}
    ```

  - Specify true or false if the repository is to be created on the back-end database, as follows:

    ```
    b_repository=false
    ```

  - Provide the Mobile Server port number, where incoming clients connect to it, as follows:

    ```
    s_mobile_port="80"
    ```

  - Provide the database SYSTEM username and password, as follows:

    ```
    s_olite_sysdb_username="myuser"
    s_olite_sysdb_pwd="mypwd"
    ```

  - Provide the Mobile Server schema name and password, as follows:

    ```
    s_olite_mb_user="myschema"
    s_olite_mb_pwd="myschemapwd"
    ```

  - Provide the Mobile Server demo schema user name and password, as follows:

    ```
    s_olite_demoschema_username="demoschema"
    s_olite_demoschema_pwd="demoschemapwd"
    ```

  - Provide the Mobile Server admin user name and password, as follows:

    ```
    s_olite_admin_username="adminuser"
    s_olite_admin_pwd="adminpwd"
    ```

- The response file for installing the MDK is `oracle.lite.install_type_1.rsp`. You can define two variables in this file for your install—the Oracle Home and the name for the Oracle Home—as follows:

  ```
  ORACLE_HOME="c:\olite10g"
  ORACLE_HOME_NAME="OLITE10g"
  ```

If you want to install both the Mobile Server and the MDK, execute the `runInstaller` command twice; each execution is provided a different response file. The following installs the Mobile Server, as follows:

```
./runInstaller  -responseFile
    /full_directory_path_to_response_file/oracle.lite.install_type_2.rsp
    -silent -nowelcome
```

where the options for this command are as follows:

Table 4–3 The runInstaller Command Options

Command Option	Description
`responseFile`	Provide the path and name of the response file for the component that you are installing. There are two response files: one for installing the Mobile Server and one for installing the MDK.
`silent`	Always use this option, as it defines that this is a silent installation.
`nowelcome`	Stops the GUI from displaying.

4.3.2.2 Silent Install for Mobile Server or Mobile Development Kit on Windows

Command to run the installer in silent mode for Windows.

```
setup.exe  -responseFile
/full_directory_path/stage/Response/oracle.lite.install_type_2.rsp
-silent -nowelcome
```

Variables to be filled for silent install in the response file `oracle.lite.install_type_2.rsp` to install Oracle Database Lite are as follows:

- Define the `ORACLE_HOME` destination directory, as follows:

  ```
  ORACLE_HOME="\private\olite\orahome\10gr2"
  ```

- Define the name for your Oracle Home, as follows:

  ```
  ORACLE_HOME_NAME="OLITE10g"
  ```

- Provide the hostname, port, and user for the back-end database, where the repository is to be installed, as follows:

  ```
  db_info={"my-pc.us.oracle.com","1521","myuser"}
  ```

- Specify true or false if the repository is to be created on the back-end database, as follows:

  ```
  b_repository=false
  ```

- Provide the Mobile Server port number, where incoming clients connect to it, as follows:

  ```
  s_mobile_port="80"
  ```

- Provide the database system username and password, as follows:

  ```
  s_olite_sysdb_username="myuser"
  s_olite_sysdb_pwd="mypwd"
  ```

- Provide the Mobile Server schema name and password, as follows:

  ```
  s_olite_mb_user="myschema"
  s_olite_mb_pwd="myschemapwd"
  ```

- Provide the demo schema user name and password, as follows:

```
s_olite_demoschema_username="demoschema"
s_olite_demoschema_pwd="demoschemapwd"
```

- Provide the Mobile Server administrator user name and password, as follows:

```
s_olite_admin_username="adminuser"
s_olite_admin_pwd="adminpwd"
```

Variables in the `oracle.lite.install_type_1.rsp` response file to install Mobile SDK, are as follows:

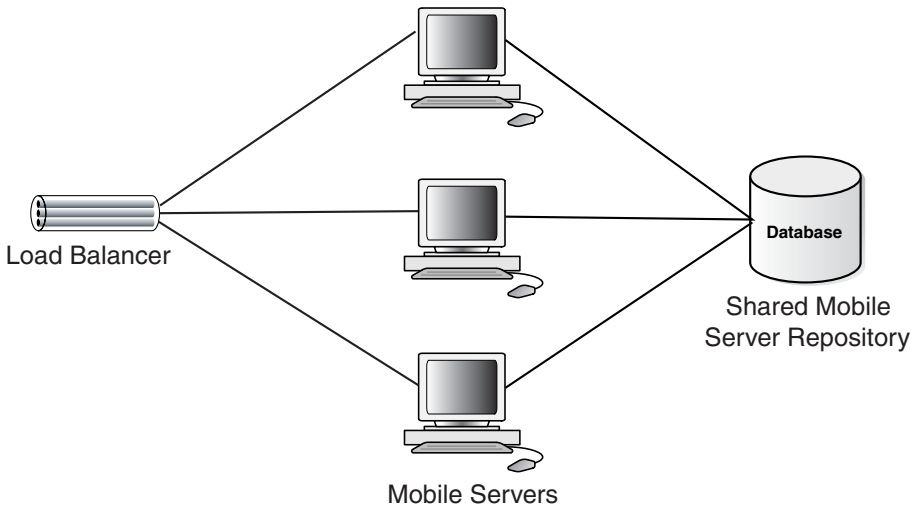
```
ORACLE_HOME="c:\olite10g"
ORACLE_HOME_NAME="OLITE10g"
```

4.3.3 Install Multiple Mobile Servers Against the Same Mobile Server Repository (Farm)

In some cases, you may want to have multiple Mobile Servers using the same Mobile Server repository. For example, as Figure 4–1 shows, if you wanted to load balance your Mobile Clients across multiple Mobile Servers, you could add a load balancer, such as BIG-IP or Oracle WebCache, before the shared Mobile Servers, and then your clients would be balanced across these Mobiles Servers, of which each accesses the same data in the shared repository.

This is the type of configuration that entails an Oracle Database Lite Farm.

Figure 4–1 Multiple Mobile Servers Sharing a Repository



You can install multiple Mobile Servers, each on its own host, that use the same Mobile Server Repository, as follows:

1. Install the first Mobile Server and the Mobile Server Repository following the instructions in Section 4.3.1.2, "Installation of Mobile Server".
2. For each subsequent Mobile Server, follow the instructions in Section 4.3.1.2, "Installation of Mobile Server", supplying the same host name, port and SID where the Mobile Server Repository exists and select **Yes** for creating the repository, which registers the shared Mobile Server with the repository.

4.3.4 Install Oracle Database Lite on Linux

Oracle Database Lite 10*g* includes a Mobile Development Kit for Linux.

The following sections provide considerations when installing Oracle Database Lite on a Linux platform:

- Section 4.3.4.1, "Patch Required if Using Linux Redhat 3.0"
- Section 4.3.4.2, "Providing Enough Swap Space on the Linux Platform"

4.3.4.1 Patch Required if Using Linux Redhat 3.0

If you install Oracle Database Lite 10*g* on Linux Redhat 3.0 and the following error occurs, apply the 3006854 patch and start the installation again.

Error occurred during initialization of VM Unable to load native library: `/tmp/myhost/jre/lib/i386/libjava.so: symbol __libc_wait`, version `GLIBC_2.0` not defined in file `libc.so.6` with link time reference.

4.3.4.2 Providing Enough Swap Space on the Linux Platform

If the swap space is not sufficient on your Linux machine, then modify the value for the `SWAP_SPACE` variable in the `install/linux/oraparam.ini` file. The default value is set to `SWAP_SPACE=1536`. For example, if your machine is not as powerful, decrease the swap space; for example, `SWAP_SPACE=1024`.

4.3.5 Install Oracle Database Lite on Symbian Devices

The following sections describe the installation directions for Symbian devices:

- Section 4.3.5.1, "Prerequisites for Installation"
- Section 4.3.5.2, "Oracle Database Lite 10g Components"
- Section 4.3.5.3, "Installing Oracle Database Lite on Symbian Devices"

4.3.5.1 Prerequisites for Installation

Before installing Oracle Database Lite, perform the following:

- You must install the SDK for the target device on the development machine. For example, if you are using a Motorola device, go to `www.motorola.com` and download the M1000 SDK on the development machine.
- It is recommended, but not mandatory, that for development, you install the target device development suite of either the "PC Suite" or "Desktop Suite".
- In order to use command-line prompts on your device, you need to install `eshell.exe` on the device. In addition, we recommend that you purchase a hardware keyboard to connect to your phone to type in the `eshell.exe` window.

We assume that you have a basic Symbian OS development knowledge to develop your application.

4.3.5.2 Oracle Database Lite 10*g* Components

The following are the components that you will see when you install the Oracle Database Lite 10*g* for Symbian devices:

- `OLAES.DLL` : AES Encryption Module
- `OLOBJ40.DLL` : Core Database Module
- `OLSQL40.DLL` : SQL Parser Module

- OLOD2040.DLL : ODBC 2.0 Driver
- ZLIB.DLL : zlib Compression Module
- OCAPI.DLL : Synchronization Module
- POLITE.INI : System-wide Common Configuration File
- ODBC.INI : Data-source Name Management File
- OLITE40.MSB : Message File (English)
- CREATEDB.EXE : Utility to create a new database
- REMOVEDB.EXE : Utility to remove an existing database
- ENCRYPDB.EXE : Utility to encrypt a database
- DECRYPDB.EXE : Utility to decrypt an encrypted database
- ODBINFO.EXE : Utility to display/modify database setting

4.3.5.3 Installing Oracle Database Lite on Symbian Devices

To install Oracle Database Lite on Symbian devices, perform the following:

1. On the development machine, unzip olite_epoc.zip in the EPOCROOT directory. For example, the EPOCROOT directory could be C:\Symbian\8.0a\S60_2nd_FP2_J.

   When you complete the installation, the following files are unzipped:

   - Header files are placed in the epoc32\include\olite directory.
   - ARMI (urel) .lib files are placed in the epoc32\release\armi\urel directory.
   - THUMB (urel) .lib files are place in the epoc32\release\thumb\urel directory.
   - WINS (udeb) binaries and .lib files are copied into the epoc32\release\wins\udeb directory.
   - Initial configuration files are copied into the epoc32\wins\c\System\Data directory.
   - Samples are copied into the OliteEx directory.

2. On the target device, perform the following:

   a. For most devices, copy the olite_core.sis file to the device using PC Suite, Desktop Suite, or an external memory card. If you are using the Sony Ericsson P910 device, then you must copy the olite_core_uiq2x.sis file to the device.

   b. Install the olite_core.sis file (or the olite_core_uiq2x.sis file, if using the Sony Ericsson P910 device), which causes the following:

      The following files are copied into the target drive, which by default is the !:\System\Libs\ directory: OLAES.DLL, OLOBJ40.DLL, OLSQL40.DLL, OLOD2040.DLL, ZLIB.DLL, and OCAPI.DLL.

      Note: You may chose a different target directory during installation.

If the configuration files do not already exist on the device, then the following files are copied into the `C:\System\Data\` directory: `polite.ini`, `odbc.ini`, and `olite40.msb`.

3. To optionally install the Oracle Database Lite 10*g* Utility Tools, perform the following:

Note: These utility tools are command line based programs; thus, you need to install and use the `eshell.exe` program to execute them.

---

   a. For most devices, copy the `olite_tools.sis` file to the target device using PC Suite, Desktop Suite, or a memory card. If you are using a Sony Ericsson P910, then copy the `olite_tools_uiq2x.sis` file.

   b. Install either the `olite_tools.sis` file or if using the Sony Ericsson P910 device, the `olite_tools_uiq2x.sis` file. This copies the following files into the target directory (which by default is the `!:\System\Programs\` directory): `CREATEDB.EXE`, `REMOVEDB.EXE`, `ENCRYPDB.EXE`, `DECRYPDB.EXE`, and `ODBINFO.EXE`.

Note: You may chose a different target directory during installation.

4.3.6 Client Install

See Chapter 2, "Installing Mobile Clients" in the *Oracle Database Lite Administration and Deployment Guide* for instructions on how to download the Mobile client onto the device.

4.3.7 Custom Install

The custom install option enables a user to install separate components within the Mobile Development Kit and Mobile Server. This is only for the user who is familiar with these components and not for the beginner.

4.4 Post-Installation Configuration Requirements

The following describes any post-installation requirements:

- Section 4.4.1, "Setup DISPLAY Variable for UNIX Systems in Oracle Application Server"
- Section 4.4.2, "Enabling Branch Office on Windows XP Service Pack 2"

4.4.1 Setup DISPLAY Variable for UNIX Systems in Oracle Application Server

If you are using a UNIX system, you may have to configure the `DISPLAY` variable. Mobile Manager uses UIX to create the graphics in its display, which requires access to your XServer. If you have the XServer on the local machine where the Mobile Manager is executing, then you do not have to perform any configuration. However, if the XServer is on another machine, then set the `DISPLAY` variable in the OPMN component of the Oracle Application Server to point to that machine. In the `opmn.xml` file, modify the `<environment>` section and add the following property:

```
<prop name="DISPLAY" value="hostname:display_number.screen_number"
```

where the host is the machine where the XServer is available. For example, if the XServer is on `myhost`, then set the *`DISPLAY`* environment variable, as follows:

```
<environment>
 <prop name="DISPLAY" value="myhost:0.0"
</environment>
```

4.4.2 Enabling Branch Office on Windows XP Service Pack 2

When you install Windows XP Service Pack 2, the Internet Connection Firewall (ICF) defaults to ON. In order for the Branch Office Server to work properly, you either need to turn the ICF OFF or enable port 1160 within the ICF. To enable port 1160, go to the Windows Firewall control on your Windows machine. Select the Exception tab. Click **Add Port**. Add port 1160 with any name.

4.5 Starting Mobile Server

You can start the Mobile Server through the Windows All Programs item or automatically when you start the middle-tier. Both options are described below:

- Select All Programs->Oracle Database Lite 10g->Mobile Server
- Mobile Server is automatically started when you start the middle-tier, as follows:
  - If you are using standalone Mobile Server, which uses OC4J standalone, start the standalone version by executing the following:

    ```
    cd ORACLE_HOME\Mobile\Server\bin
    runmobileserver
    ```

    Note: The `runmobileserver` script contains the default Java flags for starting OC4J. If you want to modify the flags for how Mobile Server is started, you can modify the `runmobileserver` script.

    You should no longer use the `webtogo.exe` to launch the standalone version of the Mobile Server.

    If you want to stop the standalone Mobile Server, you need to stop the OC4J standalone process by executing `admin.jar`. If you use the `-shutdown` option, then it waits for the process to end normally. To end immediately, use the `-shutdown force` option. Both options are as follows:

    ```
    java -jar admin.jar ormi://<mobile_server_host>:<port> oc4jadmin
                        <mobileadmin_password> -shutdown
    ```

    To end immediately, use the following `-shutdown force` option:

    ```
    java -jar admin.jar ormi://<mobile_server_host>:<port> oc4jadmin
                        <mobileadmin_password> -shutdown force
    ```

    Where:

    * The `<mobile_server_host>` is the host where the standalone Mobile Server resides.
    * The `<port>` is the port number on the Mobile Server
    * The `<mobileadmin_password>` is the password you use when you sign in as the mobile administrator for Mobile Server.

    Refer to the OC4J documentation for full details.

OR

- Start OracleAS through the Windows Services panel. Alternatively, you can start the application server from the OracleAS GUI.

Note: If you have installed the Web Cache with your application server installation, then you must always start the Web Cache in order for Mobile Server to execute properly. If Web Cache has not been started, you will receive a `Page Not Found` when directed to the `http://<application server host>:7777/webtogo/index.html` page.

4.6 Testing Your Mobile Server Installation

To test whether your Mobile Server was installed correctly, test it in one of the following ways:

- If you are using the standalone version, test your Mobile Server through a browser with the following URL:

  ```
  http://<machine-name>[:port]/webtogo
  ```

  Note: If, after installation, you have port conflicts and need to change the port number for OC4J standalone, edit the file *ORACLE_HOME*`\mobile_oc4j\j2ee\mobileserver\config\http-web-site.xml` and modify the following with an alternate port number. For all UNIX and Linux platforms, if you are not running as root or superuser when you start the server, then use a port number greater than 1024.

  The following shows the port set to 80:

  ```
  <web-site port="80" display-name="OracleAS Containers for J2EE HTTP
  Web Site">
  ```

- If you are using OracleAS as your middle-tier, test Mobile Server through a browser with the following URL:

  ```
  http://<machine-name>[:port]/webtogo
  ```

For more information on testing your installation using the samples, see Chapter 7, "Quick Start for Oracle Database Lite".

4.7 Removing Demo Applications

Before you use Oracle Database Lite in a production environment, you may wish to remove the demo applications. To remove the demo applications, execute the following batch file:

```
demoinstaller [-i | -u] <sys_user> <sys_pwd> <mobile_user> <mobile_pwd>
```

where the options are as follows:

Option	Description
`-i` or `-u`	`-i`: install the demos `-u`: remove the demos
`<sys_user> <sys_pwd>`	Provide the system username and password that are used to drop the schema in which the demos are installed.
`<mobile_user> <mobile_pwd>`	Provide the Mobile Server log on username and password, which is necessary to remove the demos from the repository.

The removal process enables you to delete the schema into which the applications were installed. If you did not use the schema for any other purpose, then allow the batch file to remove the demo application schema.

4.8 How to Uninstall Oracle Database Lite

After you remove the Oracle Database Lite components through the Oracle Universal Installer, also perform the following tasks:

1. Delete the following directories:
   - *ORACLE_HOME*`\Mobile`
   - If you are using Oracle Database Lite Standalone, then delete the *ORACLE_HOME*`\Mobile_oc4j` directory; if you are using Oracle AS, then delete the *ORACLE_HOME*`\j2ee\home\applications\mobileserver` directory.
2. `Drop TABLESPACE SYNCSERVER INCLUDING CONTENTS`
3. Delete the `Mobile01.dbf` database file from the operating system.

5

Installing Your Mobile Client

One of the benefits of Oracle Database Lite is that you can have an application downloaded onto a device, where data can be synchronized between the device and the back-end Oracle database.

In general, the types of Mobile clients are as follows:

- Windows clients (such as OC4J, Web-to-Go, Branch Office, and BC4J): The application built for these clients uses a Java browser.
- Linux, Win32, and WinCE clients: These applications are client/server applications. Thus, start the application as you would start any application on these platforms.
- Broadbeam-enabled clients, which are used for network routing.

> **Note:** On each client device, only one user can be installed for each device; that is, you cannot have multiple users on a single client device. In addition, you can configure only one device for a particular user/Mobile Client combination. That is, you cannot install the same Mobile Client for the same user on more than one device. For example, it is not possible to have two devices both executing a Mobile Client for Web-to-Go for the user `JOHN`. Server initiated synchronizations, where data changes from the server are automatically synchronized to the device, will not work properly.

The following sections detail how to install Mobile Client software on your client machine:

See Chapter 2, "Managing the Mobile Client" in the *Oracle Database Lite Administration and Deployment Guide* for instructions on how to perform certain functions on the client. See Chapter 1, "Using the Workspace for Managing the Mobile Server" in the *Oracle Database Lite Administration and Deployment Guide* for information on how to manage functionality from the Mobile Server.

5.1 Preparing the Device for a Mobile Application

In order for a device to execute Mobile applications, you must do the following:

> **Note:** Install the Mobile client for any application after the application is published.

1. Install the Oracle Database Lite Mobile client software that is appropriate to the client platform on your client machine. For example, install either the Mobile client for Win32, Mobile client for OC4J or Web-to-Go on a Windows 32 client machine.

   See Section 5.2, "Installing the Mobile Client Software" for a full description.

2. Download the user applications and its associated data.

   Synchronize the Mobile client for the first time. When you sign in with the username/password of the Mobile user who owns the Mobile applications to which this user has access are downloaded. In addition, the data for each application is retrieved.

   > **Notes:** The username and password are limited to a maximum of 28 characters.
   >
   > For more information about synchronization, see Chapter 5, "Managing Synchronization" in the *Oracle Database Lite Administration and Deployment Guide*. .

3. You can now launch your applications from your client machine or from your Mobile device.

5.2 Installing the Mobile Client Software

Before you install the Mobile Client on your device, make sure that there is 1 MB of space available to download the `setup.exe`. Also, we do not support a configuration scenario where, on a single system, both a Mobile client and the Mobile Development Kit (MDK) is installed.

> **Note:** If you are installing a Mobile Client for Linux Web-to-Go, follow the instructions provided in Section 5.2.1, "Installing Web-to-Go on Linux" before downloading the `setup.exe`.

To install the Mobile client software, perform the following tasks.

1. On the client, open a browser to point to the Mobile Server using the following URL.

   ```
   http://<mobile_server>:<port>/webtogo/setup
   ```

   > **Note:** Substitute `https` if using HTTP over SSL.

   Figure 5–1 displays the Mobile client setup page, which contains links to install Mobile client software for multiple platforms and languages. You can select another language than English on the Language pulldown.

For viewing platforms, you can choose to see all available platforms for the indicated language, or only those platforms for Windows or Windows CE with the Platform pull-down menu.

NOTE: All displayed CAB files for Windows and WinCE are optimized for size to minimize the footprint on your device; thus, no Java, `msql` or utility support are provided in these CAB files. If you want Java, `msql`, or utility support; then, before you install the client software, you must replace the lighter CAB file with the full featured SDK CAB file, as described in Section 5.2.2, "Use SDK CAB Files for Client Install for Java, MSQL, and Utility Support".

Note: Only the Mobile Development Kit has the full National Language Support for (Traditional and Simplified) Chinese, Japanese, and Korean (CJK). All other components, including Mobile clients, support CJK without the Traditional Chinese language. However, the Simplified Chinese language is supported.

When you select the language, the collation sequence for the Oracle Lite database is also preconfigured for you. You can only perform a linguistic sort on Oracle Lite databases that have the collation sequence of FRENCH, GERMAN, CZECH, OR XCZECH. You cannot do a linguistic sort on a BINARY collation sequence, which is used with all languages, except the three previously listed.

Figure 5–1 Mobile Client Setup Page

Note: Available clients may differ. See *Oracle Database Lite Administration and Deployment Guide* for more information.

---

2. Click the Mobile client for your language and client platform.

> **Note:** There are two client versions for the Web-to-Go model. The Oracle Lite WEB and the Oracle Lite WEB OC4J. Use Oracle Lite WEB OC4J when you need full J2EE 1.3 compliance. The original Oracle Lite Web client uses the Oracle Lite Servlet engine, which does not support all features for J2EE 1.3. The Oracle Lite WEB OC4J uses the OC4J stack within OracleAS; thus, you have full J2EE 1.3 support. However, you also must create the client according to OC4J/J2EE specifications.

3. The Save As dialog box appears. The file name field displays the setup executable file for the selected platform as an `.exe` file type. Save the executable file to a directory on the client machine.

> **Note:** For WinCE, you install any of the Oracle Lite Windows Mobile platforms to ActiveSync. Then, when the device is put into the cradle, ActiveSync installs the Oracle Database Lite onto the device when it synchronizes.

4. Install the Mobile client. For all platforms, except installing WinCE on ActiveSync, go to the directory where you saved the setup executable file. Double-click the file to execute it.
5. Enter the username and password for the Mobile user. On each client device, only one user can be installed for each device; that is, you cannot have multiple users on a single client device.

> **Note:** The username and password are limited to a maximum of 28 characters long.

6. Provide the client directory name where to install the Mobile client.
7. Once installed, synchronize the Mobile client for the first time. During the first synchronization, all applications and data for this user is brought down and installed on your Mobile client. See Table 5–1 for a description of synchronization for each platform.

> **Note:** See Section 5.3, "Configuring for Default Sync When Installing the Client" for directions on how to enable a default synchronization after any client installation on your device.

Table 5–1 Initializing the First Synchronization for Each Mobile Client Platform

Oracle Mobile Client	Initial Synchronization Details
Oracle Lite WEB or Oracle Lite WEB OC4J, both for Web-to-Go support	The synchronization step takes place when you click **Next**, after executing the `setup.exe`. This prompts you to login to the Mobile client for OC4J or Web-to-Go. If you want to synchronize at another time, do the following: 1. Open a browser to the Mobile client. For example, if you install a Web client with port 8080, point the browser to `http://localhost:8080/webtogo`. 2. Log in with the username/password for the Mobile user. 3. Click **Sync** on the tabs in the upper right corner.
Oracle Lite PocketPC for WinCE devices	If you install the PocketPC platform to ActiveSync, insert the WinCE device in the cradle. ActiveSync performs a synchronization to install Oracle Database Lite on the device. After Oracle Database Lite is installed on the device, then start the Device Manager Agent on the device by executing `dmagent.exe`, which is in the `orace` directory.
All other platforms	Perform the following steps. 1. Locate the directories where you installed the runtime libraries, and launch the Mobile Sync application. 2. The `mSync` dialog appears. Enter the user name and password of the Mobile user. If you do not know your user name and password, ask your system administrator, who creates users and assigns passwords to each user. In the **Server** field, enter the URL for your Mobile Server. Click **Apply** and click **Sync**.

5.2.1 Installing Web-to-Go on Linux

Perform the following to install and run Web-to-Go on Linux.

WARNING: If you are testing the Oracle Database Lite on Suse Linux, you must do the following before installation:

```
ln -s /usr/lib/libssl.so.0.9.7 /usr/lib/libssl.so.4
ln -s /usr/lib/libcrypto.so.0.9.7 /usr/lib/libcrypto.so.4
```

Once the installation is complete, perform your tests and then remove the soft links, as these may cause problems with other programs you have installed on your machine. This instruction is only for testing and should not be a permanent option.

1. Set the following environment variables:
   - Set `OLITE_HOME` to where Oracle Database Lite is installed, such as `/home/<user>/olite`
   - Set `JAVA_HOME` to the Java installation directory
   - Add the following to the `LD_LIBRARY_PATH`
     `$JAVA_HOME/jre/lib/i386`
     `$JAVA_HOME/jre/lib/i386/server`
     `$OLITE_HOME/bin`
   - Add `$OLITE_HOME/bin` to the `PATH`

2. Download the Web-to-Go setup executable by clicking the "Oracle Lite Linux WEB" link on the Mobile Server setup page.
3. After the download is complete, set execution permissions on the setup executable with `chmod 755 setup`.
4. Execute the setup command, as follows:

   ```
   ./setup
   ```

5. To start Web-to-Go in the debug mode, do the following:

   ```
   cd $OLITE_HOME/bin
   ./webtogo -d0
   ```

   To start Web-to-Go in the daemon mode, do the following:

   ```
   cd $OLITE_HOME/bin
   ./webtogo
   ```

   To kill Web-to-Go, which is in the daemon mode, do the following:

   ```
   cd $OLITE_HOME/bin
   ./webtogo -k
   ```

To uninstall Web-to-Go and delete the database files, perform the following:

```
cd $OLITE_HOME
./uninst
```

The `dmagent` is automatically launched in a daemon mode when setup is executed. However if you want to restart it, first kill the current process and then perform the following:

```
cd $OLITE_HOME/bin
./dmagent
```

5.2.2 Use SDK CAB Files for Client Install for Java, MSQL, and Utility Support

The Windows and WinCE CAB files that are included, by default, in the client installation setup directory does not include Java, `msql`, and utility support in order to reduce the footprint on the client devices. If you want to be able to download the full featured SDK CAB files, perform one of the following:

- Section 5.2.2.1, "Downloading SDK CAB Files for PPC50 ARMV4I and PPC2003 ARMV4"
- Section 5.2.2.2, "Downloading SDK CAB Files for WCESTD500 ARMV4I and WCESTD500 X86"

5.2.2.1 Downloading SDK CAB Files for PPC50 ARMV4I and PPC2003 ARMV4

The SDK CAB files are installed with the MDK install.

1. Copy and rename the CAB file. The CAB files are named `olite.<language>.<platform>.<chipset>_sdk.CAB`. Rename the CAB file to `olite.cab`. The SDK CAB files are installed with the MDK in the following directory:

   ```
   <ORACLE_HOME>\Mobile\SDK\wince\<platform>\cabfiles\
   ```

   After renaming to `olite.cab`, copy the SDK CAB file to the appropriate directory depending on the device software and language, as follows:

- Oracle Lite PPC50 ARMV4I: Copy the `olite.cab` file to the *<ORACLE_HOME>*`\mobile_oc4j\j2ee\mobileserver\applications\mobileserver\setup\`*<language>*`\stdsdk500\armv4i\` directory.
- Oracle Lite PPC2003 ARMV4: Copy the `olite.cab` file to the *<ORACLE_HOME>*`\mobile_oc4j\j2ee\mobileserver\applications\mobileserver\setup\`*<language>*`\stdsdk500\armv4\` directory.

You are replacing the smaller CAB files with the SDK CAB files.

2. On the client, open a new browser that points to the setup page to pull down the platform with the SDK CAB file.

5.2.2.2 Downloading SDK CAB Files for WCESTD500 ARMV4I and WCESTD500 X86

The WCESTD500 SDK CAB files will not show up in the setup UI unless you create a platform for them. For a full description, see Section 7.5.4.3.2, "Register the WCESTD500 SDK CAB Files with Java, `msql`, and Utility Support" in the *Oracle Database Lite Administration and Deployment Guide*.

5.3 Configuring for Default Sync When Installing the Client

In the default configuration, all Mobile clients do not automatically synchronize after you install the client. However, you can modify your configuration to automatically sync each client after it is installed, as follows:

1. Logon to the Mobile Server as an Administrator and launch the Mobile Manager tool.
2. Click on Mobile Devices, followed by Administration.
3. Click on Command Management.
4. Edit the Command Device Info (Retrieve device information).
5. Insert 'Synchronize' as a Selected Command and click **Apply** to accept the changes.

See Section 7.7, "Sending Commands to Your Mobile Devices" in the *Oracle Database Lite Administration and Deployment Guide* for more details on sending commands to your Mobile device.

5.4 Configuring the Client for Secure Socket Layer (SSL)

As the end user, you can configure the Mobile client for OC4J or Web-to-Go to establish an SSL connection between the Mobile client and the Mobile Server. A complete description of how to configure your Mobile client to use SSL is described in Section 12.1.4, "Client-Side Configuration for Secure Socket Layer (SSL)" in the *Oracle Database Lite Administration and Deployment Guide*.

5.5 Specifying Whether the Client Uses a Static or Dynamic (DHCP) IP Address

Use the `IP_CONFIG` parameter in the server `webtogo.ora` file to specify the method the client uses to retrieve its IP address. Your client device can use either a static IP address or a dynamic (DHCP) method in retrieving an IP address. If you are using DHCP, then you need to set this parameter to `DYNAMIC`; the default is `STATIC`.

If you are using DHCP, then the underlying code needs to know to not use the IP address that was used for the previous connection/synchronization. If you are using DHCP and have set this parameter to `STATIC`, your synchronization may never occur, since it is probably trying to synchronize to an IP address that is no longer valid for this device.

You set this parameter in the server `webtogo.ora` file, so that the Mobile Server knows if the client is DHCP, then may have a different IP address each time.

For more information, see Section A.1, "WEBTOGO" in the *Oracle Database Lite Administration and Deployment Guide*.

5.6 Using Offline Instantiation to Distribute Multiple Mobile Clients

You can enable your users to install their client using a distribution method, such as a CD, through the network, or email. To install the Mobile client and perform the first synchronization to retrieve the applications (with the initial data) can be a performance issue. In this case, the administrator pre-creates the Mobile binaries with the user ODB files (includes the applications and data for the user) to the client. The download of this package is faster than having each user perform the first synchronization on their device. Thus, this procedure helps users avoid an expensive performance hit when creating and synchronizing the Mobile client for the first time.

Offline instantiation is a tool that enables an administrator to gather and package the Mobile client binaries and the user applications and data into a single directory. Offline instantiation is part of the Mobile Development Kit, which can be installed only on a Windows platform. Thus, you create all of your user distribution files on a Windows machine and you can only create multiple user distribution files for OC4J, Web-to-Go, Branch Office, Win32, and WinCE Mobile clients. We recommend that you use the same Windows environment where a Mobile server exists to create your distribution files.

See Chapter 8, "Offline Instantiation" in the *Oracle Database Lite Administration and Deployment Guide* for full instructions on how to use the Offline Instantiation engine to create and deploy multiple clients.

6

Upgrade Oracle Database Lite

The following sections describe the steps for upgrading your Oracle Database Lite software:

- Section 6.1, "Upgrade Your Oracle Database Lite from 5.0.2.x to 10g Release 3"
- Section 6.2, "Upgrade Your Oracle Database Lite from 10g Release 1 or 2 to 10g Release 3"
- Section 6.3, "Upgrade the Mobile Client"

6.1 Upgrade Your Oracle Database Lite from 5.0.2.x to 10g Release 3

The following sections describe the steps for upgrading your Oracle Database Lite software from 5.0.2.x to 10*g* Release 3:

- Section 6.1.1, "What 5.0.2.x Versions Can I Upgrade to 10g Release 3?"
- Section 6.1.2, "Pre-Upgrade Instructions for the 5.0.2.x Version"
- Section 6.1.3, "Upgrade Your 5.0.2.x Version Mobile Server Repository"
- Section 6.1.4, "Upgrade Your 5.0.2.x Version Existing Applications"
- Section 6.1.5, "Upgrading BC4J Mobile Client 5.0.2.x to Version 10g Release 2"
- Section 6.1.6, "Upgrading 5.0.2.x Version Branch Office"
- Section 6.1.7, "Use the Appropriate Context for Your Mobile Server"
- Section 6.1.8, "Migrate Your 5.0.2.x Version Users From the Mobile Server Repostory to the Oracle Internet Directory"
- Section 6.1.9, "Post-Upgrade Instructions for 5.0.2.x Version"
- Section 6.1.10, "Remove the 5.0.2.x Installation"
- Section 6.1.11, "New Consolidator Sequence Properties Added During Upgrade"
- Section 6.1.12, "Upgrade Oracle Database Lite Release 2 to Release 3"

6.1.1 What 5.0.2.x Versions Can I Upgrade to 10g Release 3?

When you upgrade your Oracle Database Lite 5.0.2.x version to Oracle Database Lite 10*g* Release 3, you need to upgrade, as follows:

1. Upgrade your Oracle Database Lite 5.0.2.x to Oracle Database Lite 10g Release 2. There is no straight upgrade from 5.0.2.x to Oracle Database Lite 10g Release 3.

2. Upgrade the Oracle Database Lite 10g Release 2 to Release 3. See Section 6.2, "Upgrade Your Oracle Database Lite from 10g Release 1 or 2 to 10g Release 3" for more details.

Upgrading Oracle Database Lite from 5.0.2.x to 10*g* Release 2 is only supported in a separate *ORACLE_HOME* from the previous installation; thus, an existing Oracle Database Lite 5.0.2.x installation cannot be upgraded to the Oracle Database Lite 10*g* Release 2 in the same *ORACLE_HOME*.

Table 6–1 details what version of Oracle Database Lite and *i*AS installation can be upgraded to Oracle Database Lite 10*g* Release 2.

Table 6–1 Upgrading 5.0.2.x Oracle Database Lite to 10g Release 2

	10*g* R2 Standalone (different Oracle Home)	**10*g* R2 on *i*AS 10.1.2**	**10*g* R2 on *i*AS 10.1.2 with OID 10.1.2**
5.0.2.x Standalone	Supported	Supported	Not Supported
5.0.2.x and *i*AS 9.0.2	Supported	Supported	Not Supported
5.0.2.x and *i*AS 9.0.2 with OID	Not Supported	Not Supported	Supported
5.0.2.x and *i*AS 9.0.3	Supported	Supported	Not Supported
5.0.2.x and *i*AS 9.0.3 with OID	Not Supported	Not Supported	Supported

6.1.2 Pre-Upgrade Instructions for the 5.0.2.x Version

In order to upgrade your Oracle Database Lite 5.0.2.x installation, perform the following:

1. If you are upgrading a 5.0.2.10 Mobile Server that uses Oracle9*i*AS (version 9.0.4), then remove references to the `wtgias.conf` and `wtgapach.conf` files from the Oracle9*i*AS `oracle_apache.conf` or the HTTP `httpd.conf` configuration files before you perform the upgrade for the Mobile Server. The following line is an example of the reference to the `wtgias.conf` file:

```
include "c:\olite502\Mobile\Server\bin\wtgias.conf"
```

2. If you have multiple Mobile Servers that use the same Mobile Server Repository on a back-end database, you must shut down all of these Mobile Servers before you start the upgrade. Then, upgrade one of these Mobile Servers and the Mobile Server Repository. After this upgrade completes successfully, you can upgrade each additional Mobile Server that uses the Repository. After all of the shared Mobile Servers are upgraded, you can restart them.

3. Install the current version of Oracle Database Lite in a separate Oracle Home than your 5.0.2.x version.

> **WARNING: Upgrading Oracle Database Lite version 5.0.1.x or previous versions is not supported with this process. You must first upgrade to 5.0.2.x before starting this process.**

See Chapter 4, "Installation of Oracle Database Lite" for instructions on how to install Oracle Database Lite. When the Repository Wizard is executed, see Section 6.1.3, "Upgrade Your 5.0.2.x Version Mobile Server Repository" for directions on how to upgrade the repository on the back-end database.

6.1.3 Upgrade Your 5.0.2.x Version Mobile Server Repository

During installation of Oracle Database Lite 10*g* Release 2, the Repository Wizard detects if you have a 5.0.2.x version of Oracle Database Lite installed and starts the upgrade process, as follows:

> **Note:** If you need to start the Repository Wizard outside of the install process, execute
> *ORACLE_HOME*\Mobile\Server\admin\repwizard

1. Enter the SYSTEM password. Click **Next**.
2. Select the schema that you are going to upgrade. Click **Next**.
3. Enter the password for the Mobile Server Repository—MOBILEADMIN—and click **Yes** for installing sample applications. Click **Next**.
4. A summary screen appears. Click **Next**.
5. After the repository is upgraded, click **Finish**.

To upgrade each individual Mobile Server, follow the instructions in Section 6.1.2, "Pre-Upgrade Instructions for the 5.0.2.x Version".

6.1.4 Upgrade Your 5.0.2.x Version Existing Applications

Once you have installed the Oracle Database Lite 10*g* Release 2 in a new Oracle Home and the Repository Wizard has upgraded the Mobile Server Repository successfully, you must execute the Application Upgrade Assistant to upgrade your existing applications.

> **Note:** Before you upgrade the application, you must remove references to the `wtgias.conf` and `wtgapach.conf` files from the Oracle9*i*AS `oracle_apache.conf` or the HTTP `httpd.conf` configuration files before you perform the upgrade for the Mobile Server. The following line is an example of the reference to the `wtgias.conf` file:
>
> ```
> include "c:\olite502\Mobile\Server\bin\wtgias.conf"
> ```

The Application Upgrade Assistant moves the applications from the 5.0.2.x Oracle Home to your new 10*g* Release 2 Oracle Home. In addition, the assistant transforms your Mobile applications to the new application model. In previous versions, you could create a single Mobile application for all platforms with a single dataset. In 10*g*, you must create a single Mobile application for each platform, each with its own dataset. Thus, if you have the `myapp` application, the Application Upgrade Assistant transforms your `myapp` application into the following:

- Web platform: named `myapp` and exists in the `myapp` directory
- Win32 platform: named `myapp_win32` and exists in the `myapp/win32` directory
- WinCE platform: named appropriately by the language and in a directory that is also named according to the language, as follows:
  - English: The application is named `myapp_wce_Pocket_PC_us_arm` and exists in the `/myapp/wce/Pocket_PC/us/arm` directory.

- Chinese: The application is named `myapp_wce_Pocket_PC_cn_arm` and exists in the `/myapp/wce/Pocket_PC/cn/arm` directory.
- Japanese: The application is named `myapp_wce_Pocket_PC_jn_arm` and exists in the `/myapp/wce/Pocket_PC/jn/arm` directory.
- Korean: The application is named `myapp_wce_Pocket_PC_ko_arm` and exists in the `/myapp/wce/Pocket_PC/ko/arm` directory.

Note: Only the Mobile Development Kit has the full National Language Support for (Traditional and Simplified) Chinese, Japanese, and Korean (CJK). All other components, including Mobile clients, support CJK without the Traditional Chinese language. However, the Simplified Chinese language is supported.

Each application is upgraded with the dataset and access rights that was in the original application.

In addition, Oracle Database Lite 10*g* Release 2 uses OC4J for all middle-tier functionality. Thus, all Mobile Web applications must now be contained within a J2EE WAR or JAR file. The Application Upgrade Assistant converts your applications by adding the required XML files and packages your Web applications into a WAR file and publishes these applications. However, for all future applications, you must create the XML files and package them into a WAR or JAR file.

The Application Upgrade Assistant is located in the following locations:

- On a Windows environment:

  ORACLE_HOME\mobile\server\admin\ApplicationUpgradeAssistant.bat

- On a UNIX environment:

  ORACLE_HOME/mobile/server/admin/ApplicationUpgradeAssistant.sh

On the command-line, supply the Mobile Server Repository Name, the Mobile Server Repository Password, and the old Oracle Home. The new Oracle Home destination is taken from the environment variables. For example, on a Windows system, where the repository name is `MOBILEADMIN`, its password is `manager`, the old Oracle Home is `c:\oracle\ora502` and the new Oracle Home is `c:\oracle\ora10g`, execute as follows:

```
ApplicationUpgradeAssistant mobileadmin manager c:\oracle\ora502
```

6.1.4.1 Manual Upgrade Instructions For Web-to-Go Applications in Retrieving the OraUserProfile Class

In previous versions, Web-to-Go applications retrieved the `OraUserProfile` class as follows:

```
OraUserProfile p = ((OraHttpServletRequest)req).getUserProfile();
```

For Oracle Database Lite 10*g*, all existing and future Mobile applications must retrieve the `OraUserProfile` class with the following lines:

```
HttpSession s = request.getSession(true);
OraUserProfile p = (OraUserProfile)s.getAttribute ("x-mobileserver-user");
```

You must manually modify all existing Mobile applications to use the new method of retrieving the `OraUserProfile` class and then republish the application to complete the upgrade process for your 5.0.2.x Mobile applications.

6.1.5 Upgrading BC4J Mobile Client 5.0.2.x to Version 10*g* Release 2

If you have Oracle Database Lite 5.0.2.x, perform the following steps to upgrade your BC4J Mobile client to 10*g* Release 2:

1. Upgrade the existing Oracle Database Lite 5.0.2.x to Oracle Database Lite 10*g* Release 2.
2. If you have any 5.0.2.x Web-to-Go clients, which are upgrading to the 10*g* release 2, then first upgrade all your Web-to-Go clients before continuing to step 3.
3. Add following line in the `[WEBTOGO]` section of server side `webtogo.ora` file and restart Mobile Server:

   ```
   CLIENT_502_UPGRADE_TYPE = BC4J
   ```

4. Login into the BC4J mobile client using the 5.0.2.x BC4J mobile client user.
5. Perform a synchronization.
6. Select **Yes** when prompted for "Do you want to upgrade".
7. The upgrade progran prompts for username and password, use the 5.0.2.x BC4J Mobile client username and password.
8. After all the BC4J mobile clients are upgraded, remove the `CLIENT_502_UPGRADE_TYPE = BC4J` parameter from the server side `webtogo.ora` file and restart the Mobile Server.

6.1.6 Upgrading 5.0.2.x Version Branch Office

To upgrade your 5.0.2.x version Branch Office to the 10*g* version of Oracle Database Lite, perform the following steps:

1. Upgrade the existing Oracle Database Lite 5.0.2.x to Oracle Database Lite 10*g* Release 2.
2. If you have any 5.0.2.x Web-to-Go clients, which are upgrading to version 10*g* , then first upgrade all all of these Web-to-Go clients before continuing to step 3.
3. Add following line in the `[WEBTOGO]` section of server side `webtogo.ora` file and restart Mobile Server:

   ```
   CLIENT_502_UPGRADE_TYPE = BOS
   ```

4. Login into the Branch Office Client using the 5.0.2.x Branch Office user.
5. Perform a synchronization.
6. Select **Yes** when prompted for "Do you want to upgrade".
7. The upgrade program prompts for a username and password, where you will provide the 5.0.2.x Branch Office username and password.
8. After all of the Branch Office clients are upgraded, remove the `CLIENT_502_UPGRADE_TYPE = BOS` parameter from the server side `webtogo.ora` and restart the Mobile Server.

6.1.7 Use the Appropriate Context for Your Mobile Server

The servlet context that you use for all Web-to-Go applications is changed for this version. You must add the `/webtogo/` servlet context to all URLs for your Web-to-Go applications. For example, in the past, you tested your Mobile Server using *<hostname:port>*`/sample1/hello`. For Oracle Database Lite 10*g*, enter *<hostname:port>*`/webtogo/sample1/hello`.

6.1.8 Migrate Your 5.0.2.x Version Users From the Mobile Server Repostory to the Oracle Internet Directory

If you want, you can use the Oracle Internet Directory (OID) for storing and retrieving user information instead of the Mobile Server Repository. To facilitate using OID, you must first migrate all user information from the repository into OID. Once migrated, you can use OID instead of the repository.

OID is part of the Oracle9*i*AS or Oracle Application Server.

When you migrate users from a Mobile Server repository into OID, you cannot have duplicate users in OID. So, if you migrate users from two repositories into a single OID, and you have users with the same name, but different passwords on two separate repositories, the user that is first migrated into OID is the one that is valid. The second attempt to migrate an existing username into OID from a different repository will not migrate and no message is provided. This can be a problem if you have two users in different repositories with different passwords.

Migrate your existing users in the repository to the OID through the `oiduser` tool, which is located in *ORACLE_HOME*`\mobile\server\bin`. The `oiduser` tool migrates your existing users with either randomly-generated passwords or a common password.

1. Set the `SSO_ENABLED` parameter in the `webtogo.ora` file to `YES`.
2. If you are using Oracle9*i*AS, then explicitly grant permission to the `webtogo.jar` file to enable calls originating from this JAR file by adding the following content to the `jazn-data.xml` file, which is located in the *ORACLE_HOME*`/config` directory:

```
<grant>
  <grantee>
    <codesource>
      <url>file:$MIDTIER_ORACLE_HOME/mobile/server/bin/webtogo.jar</url>
    </codesource>
  </grantee>
  <permissions>
    <permission>
     <class>oracle.ias.repository.schemaimpl.CheckRepositoryPermission</class>
      <name>connectAs</name>
    </permission>
  </permissions>
</grant>
```

3. Migrate the user information using the `oiduser` tool with either randomly-generated passwords or a common password, as follows:

> **Note:** The existing password will not be migrated. Instead, either choose to have the new password randomly generated or to use a common password, such as `admin`.

- To use randomly-generated passwords for each user, execute the `oiduser` tool without the `-P` option, as follows:

  ```
  oiduser <Oracle_Home> <Mobile Server Repository username> <Mobile Server
  Repository password> <OID port number> <OID host name> <OID password>
  ```

  For example, the default setting would be:

  ```
  oiduser <Oracle_Home> mobileadmin manager 389 myhost-pc1.com welcome1
  ```

- To use a common password for all users, provide the common password with the `-P` option, as follows:

  ```
  oiduser <Oracle_Home> <Mobile Server Repository username> <Mobile Server
  Repository password> <-P> <common password> <OID port number> <OID host
  name> <OID password>
  ```

  where the common password is specified by you.

Executing `oiduser` generates the `mobile_oid_user.bat` and `oiduserfile.Idif` files.

4. Copy the `mobile_oid_user.bat` and `oiduserfile.Idif` files to the same directory in the application infrastructure machine where OID is installed. You can copy them to any directory, as the `mobile_oid_user.bat` is an executable that uses the `oiduserfile.Idif` file.

   Note: The `mobile_oid_user.bat` executable requires that the `ldapadd` executable is in the `PATH`. The `ldapadd` executable is part of the application server installation.

5. Execute the `mobile_oid_user.bat` file from the command-line on the application server infrastructure machine. This creates the Oracle Database Lite users in the OID.

   All users from the Mobile Server Repository are now migrated to the OID with the passwords, as designated in step 1.

6. Set the `SSO_ENABLED` parameter in the `webtogo.ora` file to `NO`.
7. Login to Mobile Manager as the administrator and select the appropriate server.
8. Click on the Administration tab.
9. Click **Edit Config File** to edit the `webtogo.ora` file for this server.
10. If `SSO_ENABLED` has a hash mark (`#`) before it, eliminate the hash mark and set `SSO_ENABLED` to `YES`.
11. Click **Apply**.
12. Restart both the application server and the Mobile Server.

6.1.9 Post-Upgrade Instructions for 5.0.2.x Version

After upgrading the Mobile Server and before you synchronize, perform the following for your Web-to-Go, BC4J, and Branch Office clients:

1. Modify the `SERVER_URL` parameter in the `webtogo.ora` file on the Mobile client in the `WebToGo` directory and append the `/webtogo` context to the end.
2. Restart the Web-to-Go client and synchronize.

6.1.10 Remove the 5.0.2.x Installation

Once you have completed all of the upgrade exercises, you can now remove the 5.0.2.x version of the Oracle Database Lite from your system. This is not a required step, but is recommended for clarity in the future.

1. Start up the Oracle Universal Installer by double-clicking on `setup.exe`.
2. On the File Locations screen, enter the Oracle Home and path for the 5.0.2.x installation. Click **Installed Products**.
3. On the Inventory screen, select the '+' next to the Oracle Home for your 5.0.2.x installation. Then, click the box next to the 5.0.2.x installation that you are going to remove. Click **Remove**.
4. On the Confirmation screen, click **Yes**.
5. When the removal is complete, click **Close**.
6. To exit the Installer, click **Exit**.
7. Optionally, you can go into your directories and remove the directory where the 5.0.2.x installation existed.

6.1.11 New Consolidator Sequence Properties Added During Upgrade

Any upgrade from a previous version of the Mobile Server upgrades the Consolidator Sequences to a new Sequence model, which contains additional property information. The new Sequence properties have the following default values:

- `MIN_VALUE`: 0
- `WINDOW_SIZE`: 1000
- `THRESHOLD`: 100
- `INCREMENT_BY`: 1
- `SEQUENCE_MODE`: `NA_SEQ`

You can modify any of these properties through the Java `modifySequence` API, which is described fully in the *Consolidator Admin API Specification* JavaDoc.

6.1.12 Upgrade Oracle Database Lite Release 2 to Release 3

Once you have completed the upgrade to Oracle Database Lite Release 2, then upgrade this version to Release 3, as described in Section 6.2, "Upgrade Your Oracle Database Lite from 10g Release 1 or 2 to 10g Release 3".

6.2 Upgrade Your Oracle Database Lite from 10g Release 1 or 2 to 10g Release 3

If you have Oracle Database Lite 10g Release 1 or 2, you must upgrade to the latest version. The following sections describe the steps for upgrading your software:

6.2.1 To What Versions Can I Upgrade Oracle Database Lite?

Upgrading Oracle Database Lite from 10*g* Release 1 or 2 to 10*g* Release 3 is supported in the same or a separate *ORACLE_HOME* from the previous installation.

Table 6–2 details what versions of Oracle Database Lite and Oracle AS installation can be upgraded to Oracle Database Lite 10*g* Release 3.

Table 6–2 Upgrading 10g Release 1 or 2 Oracle Database Lite to 10g Release 3

	10*g* R3 Standalone (diff Oracle Home)	**10*g* R3 and *i*AS 10.1.2**	**10*g* R3 and *i*AS 10.1.2 with OID 10.1.2**	**10*g* R3 and *i*AS 10.1.3.1**	**10*g* R3 and *i*AS 10.1.3.1 with OID 10.1.4**
10*g* R1 or R2 Standalone	Supported	Supported	Not Supported	Supported	Not Supported
10*g* R1 using *i*AS 9.0.2	Supported	Supported	Not Supported	Supported	Not Supported
10*g* R1 using *i*AS 9.0.2 with OID	Not Supported	Not Supported	Supported	Not Supported	Supported
10*g* R1 using *i*AS 9.0.3	Supported	Supported	Not Supported	Supported	Not Supported
10*g* R1 using *i*AS 9.0.3 with OID	Not Supported	Not Supported	Supported	Not Supported	Supported
10*g* R1 or R2 using *i*AS 9.0.4	Supported	Supported	Not Supported	Supported	Not Supported
10*g* R1 or R2 using *i*AS 9.0.4 with OID	Not Supported	Not Supported	Supported	Not Supported	Supported
10*g* R2 using *i*AS 10.1.2	Supported	Supported	Not Supported	Supported	Not Supported
10*g* R2 using *i*AS 10.1.2 with OID	Not Supported	Not Supported	Supported	Not Supported	Supported

6.2.2 Pre-Upgrade Instructions

In order to upgrade your Oracle Database Lite Installation, perform the following:

1. If you have multiple Mobile Servers that use the same Mobile Server Repository on a back-end database, you must shut down all of these Mobile Servers before you start the upgrade. Then, upgrade one of these Mobile Servers and the Mobile Server Repository. After this upgrade completes successfully, you can upgrade each additional Mobile Server that uses the Repository. After all of the shared Mobile Servers are upgraded, you can restart them.
2. You can upgrade your existing Database Lite 10*g* Release 1 or 2 to the Release 3 in the same Oracle Home OR you can install the Release 2 of Oracle Database Lite in a separate Oracle home than your Database Lite 10*g* Release 1 or 2 home.

See Chapter 4, "Installation of Oracle Database Lite" for instructions on how to install Oracle Database Lite. When the Repository Wizard is executed, see Chapter 6.2.3, "Upgrade Your Mobile Server Repository" for directions on how to upgrade the repository on the back-end database.

6.2.3 Upgrade Your Mobile Server Repository

During installation of Oracle Database Lite 10*g* Release 3, the Repository Wizard detects if Release 1 or 2 is installed and initiates the upgrade process, as follows:

Note: If you need to start the Repository Wizard outside of the install process, execute
ORACLE_HOME\Mobile\Server\admin\repwizard

---

1. Enter the SYSTEM password. Click **Next**.
2. Select the schema that you are going to upgrade. Click **Next**.
3. Enter the password for the Mobile Server Repository—MOBILEADMIN—and click **No** for installing the sample applications. Click **Next**.
4. A summary screen appears. Click **Next**.
5. After the repository is upgraded, click **Finish**.

To upgrade each individual Mobile Server, follow the instructions in Section 6.2.2, "Pre-Upgrade Instructions".

6.2.4 Upgrade Your Existing Applications

You have to upgrade existing applications only if you installed Oracle Database Lite 10*g* Release 3 in a separate *ORACLE_HOME* than Oracle Database Lite 10*g* Release 1 or 2. Thus, once you have installed the Oracle Database Lite 10*g* Release 3 in a new *ORACLE_HOME* and the Repository Wizard has upgraded the Mobile Server Repository successfully, execute the Application Upgrade Assistant to upgrade your existing applications.

The Application Upgrade Assistant moves applications from the *ORACLE_HOME* for Oracle Database Lite 10*g* Release 1 or 2 to your the new *ORACLE_HOME* for Oracle Database Lite 10*g* Release 3.

The Application Upgrade Assistant is located in the following locations:

- On a Windows environment:

  ```
  ORACLE_HOME\mobile\server\admin\ApplicationUpgradeAssistant.bat
  ```

- On a UNIX environment:

  ```
  ORACLE_HOME/mobile/server/admin/ApplicationUpgradeAssistant.sh
  ```

On the command-line, supply the following:

- Mobile Server Repository Name
- Mobile Server Repository Password
- The old *ORACLE_HOME*
- The new *ORACLE_HOME*

For example, on a Windows system, where the repository name is `mobileadmin`, password is `manager`, the old *ORACLE_HOME* is `c:\oracle\ora10gR1` and the new *ORACLE_HOME* is `c:\oracle\ora10gR2`, execute the Application Upgrade Assistant, as follows:

```
ApplicationUpgradeAssistant mobileadmin manager
     c:\oracle\ora10gR1 c:\oracle\ora10gR2
```

6.2.5 Installing Sample Applications

For the sample applications, you can either upgrade the existing sample applications, use the existing samples in Release 3, or install new sample applications. The Release 1 or 2 samples will still work in Release 3 without any upgrade.

To install samples use following command:

```
demoinstaller -install
 [Database_Administrator_User_Name] [Database_Administrator_Password]
 [Repository_Owner] [Repository_Password]
 [Demo_user_name ][Demo_password]
```

For Example:

```
demoinstaller -install system manager mobileadmin manager master master
```

The `demoinstaller` utility is located in the following directories:

- On a Windows environment:

  ```
  ORACLE_HOME\mobile\server\demos\demoinstaller.bat
  ```

- On a UNIX environment:

  ```
  ORACLE_HOME/mobile/server/demos/demoinstaller.sh
  ```

6.2.6 Upgrade Branch Office 10*g* Release 1 or 2 to version 10*g* Release 3

If you have Oracle Database Lite 10*g* Release 1 or 2, then upgrade your Branch Office to 10g Release 3, as follows:

> **Note:** These steps assume that you have a 10*g* Release 1 or 2 Branch Office environment already configured and synchronized. If you do not have this environment currently configured, you do not need to upgrade.

1. Install Oracle Lite 10*g* Release 3 Mobile Server in the same *ORACLE_HOME* directory where you installed Release 1 or 2.
2. Stop the Branch Office client executing as a Windows Service.
3. From the Branch Office client machine, which is being migrated to 10*g* Release 3, point your browser to `http://<hostname>:<port>/webtogo/setup` on the Mobile Server 10*g* Release 3.
4. Click on "Oracle Lite Branch Office Client" link from the list of Mobile clients. Download and save the `setup.exe` file into the `BO_CLIENT_HOME\bin` directory. Overwrite the existing `setup.exe` file.
5. Open a command prompt, navigate to the `BO_CLIENT_HOME\bin` directory, and execute the `update.exe` command to start the upgrade process. Alternatively, you can open Programs->Oracle Database Lite->Oracle Lite Update.

6.2.7 Migrate Your Users From the Mobile Server Repository to the Oracle Internet Directory (OID)

You can use the Oracle Internet Directory (OID), which is part of the Oracle application server, for storing and retrieving user information instead of the Mobile Server Repository. To use OID, you must migrate all user information from the existing repository into OID.

When you migrate users from a Mobile Server repository into OID, you cannot have duplicate users in OID. So, if you migrate users from two repositories into a single OID, and you have users with the same name, but different passwords on two separate repositories, the user that is first migrated into OID is the one that is valid. The second attempt to migrate an existing username into OID from a different repository will not migrate and no message is provided. This can be a problem if you have two users in different repositories with different passwords.

Migrate existing users in the repository to OID through the `oiduser` tool, which is located in *`ORACLE_HOME`*`\Mobile\Server\bin`. The `oiduser` tool migrates existing users with either randomly-generated passwords or a common password.

The following sections describe how to migrate your users based on which Oracle Application Server you have installed:

- Section 6.2.7.1, "Mobile Server Installed On Oracle Application Server 10.1.2.0.0"
- Section 6.2.7.2, "Mobile Server Installed on Oracle Application Server 10.1.3.1.0.0 or Higher"

6.2.7.1 Mobile Server Installed On Oracle Application Server 10.1.2.0.0

Perform the following to migrate your users to OID:

1. Set the `IAS_MODE` parameter in the `webtogo.ora` file to `YES`.
2. If you are using Oracle9*i*AS, then explicitly grant permission to the `webtogo.jar` file, which enable calls originating from this JAR file. Grant this permission by adding the following to the `jazn-data.xml` file, which is located in the *`ORACLE_HOME`*`/config` directory:

```
<grant>
  <grantee>
    <codesource>
      <url>file:$MIDTIER_ORACLE_HOME/mobile/server/bin/webtogo.jar</url>
    </codesource>
  </grantee>
  <permissions>
   <permission>
    <class>oracle.ias.repository.schemaimpl.CheckRepositoryPermission</class>
    <name>connectAs</name>
   </permission>
  </permissions>
</grant>
```

3. Migrate the user information using the `oiduser` tool, for either randomly-generated passwords or a common password, as follows:
   - To use randomly-generated passwords for each user, execute the `oiduser` tool without the `-P` option, as follows:

     ```
     oiduser <ORACLE_HOME> <Mobile Server Repository username> <Mobile Server
     Repository password> <OID port number> <OID host name> <OID password> <OID
     admin name> <OID subscriber name>
     ```

For example, the default setting would be:

```
oiduser <ORACLE_HOME> mobileadmin manager 389 myhost-pc1.com welcome1
orcladmin dc=us,dc=oracle,dc=com
```

- To use a common password for all users, provide the common password with the `-P` option, as follows:

  ```
  oiduser <ORACLE_HOME> <Mobile Server Repository username> <Mobile Server
  Repository password> <-P> <common password> <OID port number> <OID host
  name> <OID password> <OID admin name> <OID subscriber name>
  ```

  where the common password is specified by you.

4. The `oiduser` tool generates the `mobile_oid_user` and `oiduserfile.Idif` files. Copy these files to the same directory in the application infrastructure machine where OID is installed. You can copy them to any directory, as the `mobile_oid_user` is an executable that uses the `oiduserfile.Idif` file.

   Note: The `mobile_oid_user` executable requires that the `ldapadd` executable is in the `PATH`. The `ldapadd` executable is part of the application server installation.

5. Execute the `mobile_oid_user` file from the command-line on the application server infrastructure machine. This creates the Oracle Database Lite users in the OID.

All users from the Mobile Server Repository are now migrated to the OID with the passwords, as designated in step 1.

If you want to enable Oracle Single Sign on on the Mobile Server, then perform the following:

1. Login to Mobile Manager as the administrator and select the appropriate server.
2. Click on the Administration tab.
3. Click **Edit Config File** to edit the `webtogo.ora` file for this server.
4. If `SSO_ENABLED` has a hash mark (`#`) before it, then eliminate the hash mark and set `SSO_ENABLED` to `YES`. Click **Apply**.
5. Restart both the application server and the Mobile Server.

6.2.7.2 Mobile Server Installed on Oracle Application Server 10.1.3.1.0.0 or Higher

Perform the following to migrate your users to OID:

1. Set the `IAS_MODE` parameter in the `webtogo.ora` file to `YES`.
2. Migrate the user information using the `oiduser` tool, for either randomly-generated passwords or a common password, as follows:

   - To use randomly-generated passwords for each user, execute the oiduser tool without the `-P` option, as follows:

     ```
     oiduser <ORACLE_HOME> <Mobile Server Repository username> <Mobile Server
     Repository password> <OID port number> <OID host name> <OID password> <OID
     admin name> <OID subscriber name>
     ```

     For example, the default setting would be:

```
oiduser <ORACLE_HOME> mobileadmin manager 389 myhost-pc1.com welcome1
orcladmin dc=us,dc=oracle,dc=com
```

- To use a common password for all users, provide the common password with the `-P` option, as follows:

  ```
  oiduser <ORACLE_HOME> <Mobile Server Repository username> <Mobile Server
  Repository password> <-P> <common password> <OID port number> <OID host
  name> <OID password> <OID admin name> <OID subscriber name>
  ```

  where the common password is specified by you.

All users from the Mobile Server Repository are now migrated to the OID with the required passwords.

If you want to enable Oracle Single Sign on on the Mobile Server then perform the following:

1. Login to Mobile Manager as the administrator and select the appropriate server.
2. Click on the Administration tab.
3. Click **Edit Config File** to edit the `webtogo.ora` file for this server.
4. If `SSO_ENABLED` has a hash mark (#) before it, then eliminate the hash mark and set `SSO_ENABLED` to `YES`. Click **Apply**.
5. Restart both the application server and the Mobile Server.

6.2.8 Remove the 10*g* Release 1 or 2 Installation

Once you have completed all of the upgrade exercises, you can now remove the Oracle Database Lite 10*g* Release 1 or 2 from your system. Only perform these steps if you have installed Oracle Database Lite 10*g* Release 3 in a separate *ORACLE_HOME*. In addition, this is not required, but is recommended for clarity.

1. Start the Oracle Universal Installer by double-clicking on `setup.exe`.
2. On the File Locations screen, enter the *ORACLE_HOME* and `PATH` for the 10*g* Release 1 or 2 installation. Click **Installed Products**.
3. On the Inventory screen, select the '+' next to the *ORACLE_HOME* for your 10*g* Release 1 or 2 installation. Then, click the box next to the installation that you are going to remove. Click **Remove**.
4. On the Confirmation screen, click **Yes**.
5. When the removal is complete, click **Close**.
6. To exit the Installer, click **Exit**.

6.2.9 Remove Duplicate Mobile Server IDs from the Repository

After migrating Oracle Database Lite 10*g* Release 1 or 2 to Release 3, you will see duplicate entries in the Mobile Manager Farms page. To delete these entries from the repository, use the `unregistermobileserver` utility:

```
unregistermobileserver  <MobileServerId>
```

Following is the location of `unregistermobileserver`:

- On a Windows environment:

  ORACLE_HOME`\mobile\server\admin\unregistermobileserver.bat`

- On a UNIX environment:

  ORACLE_HOME/mobile/server/admin/unregistermobileserver.sh

6.3 Upgrade the Mobile Client

The following sections describe how to upgrade each Mobile client platform:

- Section 6.3.1, "Upgrade Mobile Client Version 5.0.2.10 to Version 10g Release 3"
- Section 6.3.2, "Upgrade Mobile Client 10g Release 1 or 2 to 10g Release 3"

6.3.1 Upgrade Mobile Client Version 5.0.2.10 to Version 10*g* Release 3

Table 6–3 shows how each Mobile client platform updates to the latest version.

Table 6–3 Upgrade Mobile Client 5.0.2.10 to 10g Release 3

Mobile Client Platform	Upgrade Instructions
Web-to-Go	Automatic. User is prompted for username/password.
Branch Office	Supported, but manual steps required. See Section 6.1.6, "Upgrading 5.0.2.x Version Branch Office".
BC4J	Supported, but manual steps required. See Section 6.1.5, "Upgrading BC4J Mobile Client 5.0.2.x to Version 10g Release 2".
Win32	Not supported
WinCE	Supported from 5.0.2.9.0 and higher. Run `update.exe` or `msync.exe`, which launches `update.exe` after the synchronization completes.
Linux native clients	Not applicable
Linux Web-to-Go clients	Not applicable

6.3.2 Upgrade Mobile Client 10*g* Release 1 or 2 to 10*g* Release 3

Table 6–4 shows how each Mobile client platform updates to the latest version.

Table 6–4 Upgrade Mobile Client 5.0.2.10 to 10g Release 3

Mobile Client Platform	Upgrade Instructions
Web-to-Go	Automatic.
Branch Office	Supported, but manual steps required. See Section 6.2.6, "Upgrade Branch Office 10g Release 1 or 2 to version 10g Release 3".
BC4J	Automatic.
Win32	Supported. Run `update.exe` or `msync.exe`, which launches `update.exe` after the synchronization completes.
WinCE	Supported. Run `update.exe` or `msync.exe`, which launches `update.exe` after the synchronization completes.
Linux native clients	Not applicable for 10.0; for the 10.2 version, run `update`.
Linux Web-to-Go clients	Not applicable for 10.0; for the 10.2 version, it is automatic.

7

Quick Start for Oracle Database Lite

As a user, there are three tasks that you normally perform—software distribution, application deployment, and data synchronization. This chapter uses the installed samples in Oracle Database Lite to demonstrate—quickly—how to perform these three functions on the Windows 32, Web-based, and Windows Mobile platforms. The following sections describe how to install the sample applications, start Oracle Database Lite, and then distribute, deploy, and sync your application on separate platforms.

7.1 Installing the Sample Applications

As Step 8 in Section 4.3.1.2, "Installation of Mobile Server" notes, you should choose to install the sample applications while you are installing the Mobile Server. However, if you did not install them, start up the Repository Wizard on its own by executing one of the following:

- On Windows:
  ORACLE_HOME\mobile\server\admin\repwizard.bat
- On UNIX:
  ORACLE_HOME/mobile/server/admin/repwizard

Follow Steps 6 through 11 in Section 4.3.1.2, "Installation of Mobile Server" to install the sample applications. The Repository Wizard detects whether the repository is installed or, if it exists, its status. The Repository Wizard takes the appropriate action.

7.2 Starting Oracle Database Lite and Its Job Scheduler

Before you can execute any of the sample applications, verify that the Oracle Database, Oracle Database Lite and its Job Scheduler is started. To start Oracle Database Lite, execute `runmobileserver` on the Mobile Server host. Start the Job Scheduler, as follows:

1. Logon to the Mobile Server.

   Start a browser with the URL `http://<Mobile Server>/webtogo`. Note that the Mobile Server host name that you provide is not the back-end database, but is

the host where the Mobile Server and the middle-tier application server was installed.

Also, the URL assumes that you are using the default port of 80. If there was a port conflict and you had to change the port number, use the URL `http://<Mobile Server>:<port_number>/webtogo`. For more information on modifying port numbers, see Step 4 of Section 4.3.1.2, "Installation of Mobile Server".

2. Logon as the administrator. The default administrator username/password is `administrator/admin`.
3. On the Mobile Workspace screen, click **Mobile Manager**. This brings up the list of Mobile Servers available.
4. Click on the Mobile Server with which you wish to connect. These are listed by host name and port number.
5. On the Mobile Server screen, verify that the Job Scheduler is running. This is indicated by a green check under the Status column in the Components section at the bottom of the page.
6. If the Job Scheduler is not running, select the button next to it and click the **Start** button.
7. Exit the Mobile Manager by closing the window.

7.3 Starting a Windows 32 Application

When you have a Windows 32 (Win32) machine that is set up to interact with Mobile Server, you have an environment similar to Figure 7–1:

Figure 7–1 Windows 32 Interacting With Mobile Server

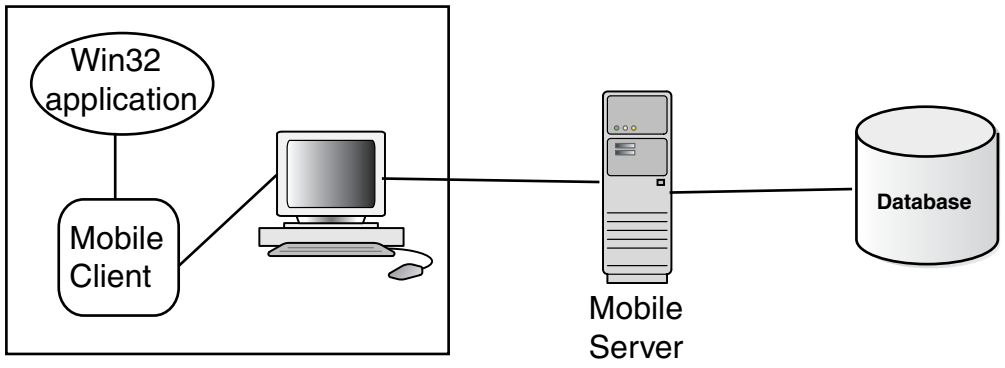


Figure 7–1 demonstrates how the Mobile Server accesses the Mobile Server Repository, which exists in a back-end database. The Mobile Client and client applications are on the Win32 machine. The Mobile Client database synchronizes with the Mobile Server.

To install and use the sample application on your Windows 32 machine, verify that you have the correct environment, retrieve the sample from the Mobile Server and install it on your Win32 machine, as follows:

1. Software Requirements for the Win32 Demo
2. Download the Mobile Client from the Mobile Server
3. Install the Mobile Client on the Win32 Device
4. Start the Win32 Sample Application

5. Enter and View Data in the Win32 Sample Application
6. Synchronize the Win32 Sample Application Data to the Database

The sample application for the Win32 environment is the transportation demo.

> **Note:** This section demonstrates how to install, deploy and use the transport demo. However, if you want to see how to build the transport demo, see Chapter 19, "Tutorial for Building Offline Mobile Applications for Win32" in the *Oracle Database Lite Developer's Guide*.

7.3.1 Software Requirements for the Win32 Demo

The Win32 sample application requires Microsoft Windows 2000/XP and that you install Microsoft .NET Framework 1.1. For directions on how to install the .NET Framework, see the following:

`http://msdn.microsoft.com/netframework/technologyinfo/howtoget/`

7.3.2 Download the Mobile Client from the Mobile Server

To download the Mobile Client, do the following:

1. Open the Mobile Client Setup through the Mobile Server, as follows:
   a. In a browser on your Win32 machine, point the browser to the Mobile Server using the URL `http://<Mobile Server>/webtogo/`.
   b. On the upper right corner, click **Setup**.
2. Click **Oracle Lite Win32**.
3. The Save As dialog box appears. The file name field displays the executable setup file for the selected platform. Save the executable to a local directory on your Win32 machine.

7.3.3 Install the Mobile Client on the Win32 Device

You install the Mobile Client on your Win32 device by performing the following steps:

1. Navigate to the local directory on your Win32 machine and double-click the Mobile Client `setup.exe`.
2. On the Logon to Server screen, enter the username and password `JUNE/JUNE` and click **OK**. The Mobile Client is now installed.
3. Navigate to the `\bin` directory where you installed the Mobile Client—such as `c:\mobileclient\bin`—and launch the Mobile Sync application by double-clicking `msync.exe`.
4. On the Mobile Sync dialog box, verify that the correct information is filled in as follows:
   - Username and Password of `JUNE/JUNE`.
   - Check Save Password.
   - Enter the host name for the Mobile Server.

   Click **Apply**. Click **Sync**.
5. When the sync completes, the Sync Result Dialog appears. Click **OK**.

6. On the Oracle Lite Software Update window, click **Install** to install the Transport Demo (`Transport_WIN32`) on your system.
7. Select the directory where you would like to install the demo and click **OK**.

7.3.4 Start the Win32 Sample Application

The Win32 sample application is the transport application, which tracks delivery of packages for a trucking delivery service. To start the application, perform the following:

1. Launch the transport demo application on your client by navigating to the directory where you installed the demo and double-click on `transport.exe`.
2. Enter username and password of `JUNE/JUNE` when prompted to logon to the sample application. Click **OK**.

7.3.5 Enter and View Data in the Win32 Sample Application

The Win32 sample application—the transport application—enables you to add package delivery information for a truck delivery service.

1. On the Transport Demo screen, you can create or view a package. Click **Create Package**.
2. On the Create Package screen, fill in the following information about the package and the truck on which it is to be delivered:
   - Name of the package: Oracle Lite CD
   - Package weight: 5
   - Truck number: 1
   - The route taken to deliver package: Santa Clara Route
   - Package priority: HIGH

   Click **Next**.

   On the next screen, fill in the package delivery location, as follows:
   - Street: 500 Oracle Parkway
   - City: Redwood Shores
   - State: CA

   Click **Save**.
3. The "Package created successfully" dialog box displays. Click **OK**.
4. Click the **Exit** button, which is the icon with a red sphere, to return to the main screen.
5. On the Transport Demo screen, click **View Packages** to see the data you just entered.
6. Click the **Exit** button to return to the main screen of the transport demo.

7.3.6 Synchronize the Win32 Sample Application Data to the Database

After you have updated all of the records that you want on your client, you can synchronize the application data to the Mobile Server, which updates the database. To

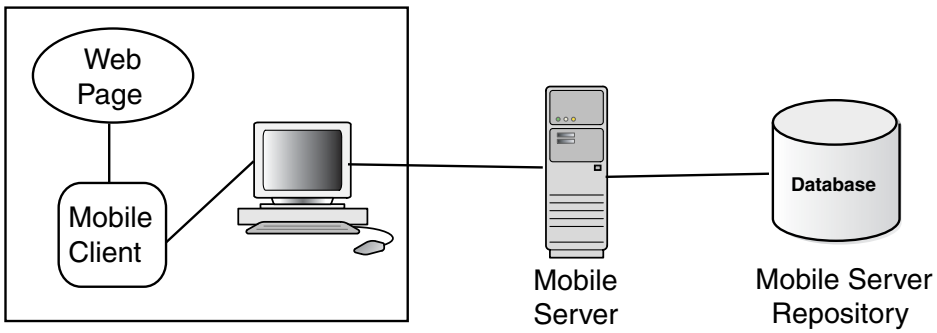
synchronize the data to the database, click the **mSync** button, which is the icon of a database with dual arrows.

You can check to see if the transport applications data was synchronized with the database by viewing the tables used by the sample—the Packages, Trucks and Routes tables—in the repository in the `MASTER` schema with username/password of `MASTER/MASTER`.

7.4 Starting a Web-Based Application

When you have a Web-based application that is set up to interact with Mobile Server, you have an environment similar to Figure 7–2:

Figure 7–2 Web-Based Application Interacting With Mobile Server



To install and use the Web-based Mobile Client, verify that you have the correct environment, retrieve the sample from the Mobile Server and install it on your machine, as follows:

> **Note:** You cannot access the Mobile Server from a Web application that is local to the Mobile Server; that is, you must access the Mobile Server from a machine that is remote to the host where Mobile Server is installed. Direct access is not supported.

1. Download the Mobile Client from the Mobile Server
2. Install the Web Mobile Client
3. Enter and Sync Data in the Web Client Sample

The Web sample application tracks your music. This sample application is also used as the base application for a tutorial on how to build Web-based applications. See Chapter 17, "Tutorial for Building Mobile Web Applications" in the *Oracle Database Lite Developer's Guide* for a full description.

7.4.1 Download the Mobile Client from the Mobile Server

To download the Mobile Client, do the following:

1. Open the Mobile Client Setup through the Mobile Server, as follows:
   a. In a browser on your remote machine, point the browser to the Mobile Server using the URL `http://<Mobile Server>/webtogo/`.
   b. On the upper right corner, click **Setup**.

2. Click **Oracle Lite WEB**.
3. The Save As dialog box appears. The file name field displays the executable setup file for the selected platform. Save the executable to a local directory on your machine—which is remote from the Mobile Server host.

7.4.2 Install the Web Mobile Client

You install the Web Mobile Client by performing the following steps:

1. Navigate to the local directory where you installed the Mobile Client and double-click the Mobile Client `setup.exe`.
2. On the Logon to Server screen, enter the username and password `JOHN/JOHN` and click **OK**.
3. If you are prompted for a port number, there is a conflict with the default port. Enter an unused port number and click **OK**. Use this port number when accessing the Web logon page through a URL.

   The Web Mobile Client Sample is now installed.
4. The Web logon page should appear in your browser. Alternatively, you can launch it through a Web browser with URL `http://localhost/webtogo/index.html`.
5. Enter the username and password of `JOHN/JOHN`. Click **Logon**.
6. A confirmation and client initialization screen appears. Click Next to synchronize the new Web Client with the Mobile Server.
7. After installation, the client is restarted and the Web Workspace appears.

7.4.3 Enter and Sync Data in the Web Client Sample

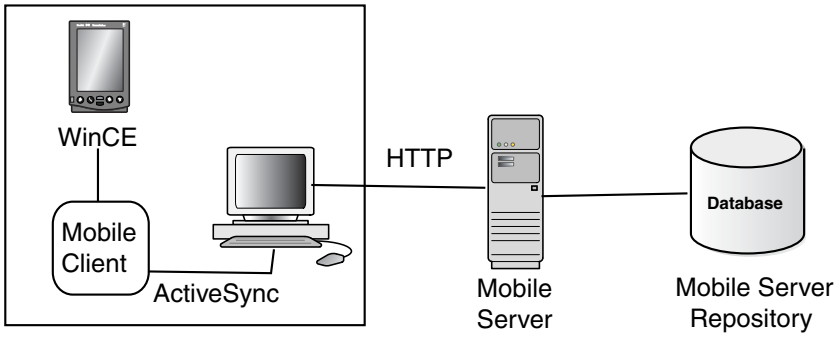
There are several sample applications that you can try; however, this section talks about using Sample3,which tracks music.

1. Select the **Applications** tab. This shows the available samples that you can execute.
2. Click **Sample3**, which launches the application.
3. Add or change data in the Sample3 application by adding musical record details, commit the changes, and close the application window.
4. Select the **Sync** tab to synchronize the modified data with the Mobile Server.
5. You can check to see if the data was synchronized with the database by viewing the Records, Tracks, and Titles tables in the back-end database.

7.5 Starting an Application on Your WinCE Device

When you have a WinCE (Windows Mobile) application that is set up to interact with Mobile Server, you have an environment similar to Figure 7–3:

Figure 7–3 Windows Mobile Application Interacting With Mobile Server



To install and use the Mobile client for your Windows Mobile device, verify that you have the correct environment, retrieve the sample from the Mobile Server and install it on your machine, as follows:

- Section 7.5.1, "Verify the Software and Hardware Requirements for the WinCE Demo"
- Section 7.5.2, "Download the Mobile Client from the Mobile Server"
- Section 7.5.3, "Install the WinCE Mobile Client"
- Section 7.5.4, "Enter and View Data in the WinCE Sample Application"
- Section 7.5.5, "Synchronize the WinCE Application Data to the Database"

The WinCE sample application is the transport application.

Note: This section demonstrates how to install, deploy and use the transport demo. However, if you want to see how to build the transport demo, see Chapter 20, "Tutorial for Building Offline Mobile Applications for Windows CE" in the *Oracle Database Lite Developer's Guide*.

7.5.1 Verify the Software and Hardware Requirements for the WinCE Demo

The WinCE sample application requires the following:

- Software requirements: ActiveSync 3.8 or higher and Microsoft .NET Compact Framework 1.0. For information on how to install Microsoft .NET Compact Framework, see the following:

  `http://msdn.microsoft.com/mobility/downloads/default.aspx/`

- Hardware requirements: Hardware as appropriate for the WinCE device used.

7.5.2 Download the Mobile Client from the Mobile Server

To download the Mobile client, do the following:

1. Open the Mobile client setup through the Mobile Server, as follows:
   a. In a browser on your Windows machine that supports the Windows Mobile device, open and point a browser to the Mobile Server host, which is remote to your machine, using the URL `http://<Mobile Server>/webtogo/`.
   b. On the upper right corner, click **Setup**.

2. Click on the sample that corresponds to the Windows Mobile device that you have. For example, the ARM example is **Oracle Lite PPC50 ARMV4I;US**.
3. The Save As dialog box appears. The file name field displays the executable setup file for the selected platform. Save the executable to a local directory on your machine.

7.5.3 Install the WinCE Mobile Client

You install the Mobile Client on your Windows Mobile (WinCE) device by performing the following steps:

1. Navigate to the local directory on the Windows machine that supports the Windows Mobile device and double-click the Mobile Client `setup.exe`.
2. On the Logon to Server screen, enter the username and password `JUNE/JUNE` and click **OK**.

   The WinCE Mobile Client is now registered with ActiveSync.
3. Launch the ActiveSync Launcher on the Windows machine. You will be asked if you would like to install Oracle Lite using the default application directory. Click **Yes**. this installs the Mobile Client libraries into the `\ORACE` directory on the Windows CE device.
4. On the Windows Mobile device, navigate to the `\ORACE` directory where you installed the Mobile Client and launch the Mobile Sync application by double-clicking `msync.exe`.
5. On the Mobile Sync dialog box, enter the following:
   - Username and Password of `JUNE/JUNE`.
   - Check Save Password.
   - Enter the host name for the Mobile Server.

   Click **Apply**. Click **Sync**.

   Note: The Mobile client device clock must be accurate for the time zone set on the device before attempting to synchronize. An inaccurate time may result in the following exception during synchronization: `CNS: 9026 "Wrong username or password. Please enter correct value and reSync."`

   ---
6. When the synchronization completes, the Sync Result Dialog appears. Click **OK**.
7. On the Oracle Lite Software Update window, click **Install** to install the Transport Demo (`Transport_PPC`) on your system.
8. Select the directory where you would like to install the demo and click **OK**.

7.5.4 Enter and View Data in the WinCE Sample Application

The WinCE sample application—the transport demo—enables you to add package delivery information for a truck delivery service.

1. Launch the transport demo application on your client. Find the Transport demo in the program list under Start->Programs or through Explorer.
2. On the Transport Demo screen, you can create or view a package. Click **Create Package**.

3. On the Create Package screen, fill in the following information about the package and the truck on which it is to be delivered:
   - Name of the package: Oracle Lite CD
   - Package weight: 5
   - Truck number: 1
   - The route taken to deliver package: Santa Clara Route
   - Package priority: HIGH

   Click **Next**.

   On the next screen, fill in the package delivery location, as follows:
   - Street: 500 Oracle Parkway
   - City: Redwood Shores
   - State: CA

   Click **Save**.
4. The "Package created successfully" dialog box displays. Click **OK**.
5. Click the **OK** button in the upper right-hand corner to return to the main screen.
6. On the Transport Demo screen, click **View Packages** to see the data you just entered.
7. Click the **Exit** button—which is the icon with a red sphere—to return to the main screen.

7.5.5 Synchronize the WinCE Application Data to the Database

After you have updated all of the records that you want on your client, you can synchronize the data to the Mobile Server, which updates the database. To synchronize the data to the database, click the **mSync** button, which is the icon of a database with dual arrows, to synchronize the modified data with the Mobile Server.

You can check to see if the transport applications data was synchronized with the database by viewing the tables used by the sample—the Packages, Trucks and Routes tables—in the repository in the `MASTER` schema with username/password of `MASTER/MASTER`.

Index

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W