

# Oracle Database Lite: Synchronizing Data Between a Device and an Oracle Database

*An Oracle White Paper  
August 2007*

# Oracle Database Lite: Synchronizing Data Between a Device and an Oracle Database

## Contents

Challenges Solved By Synchronizing Remote Mobile Data.....	3
What is Synchronization of Data? .....	3
Overview of Oracle Database Lite Synchronization.....	4
Oracle Database Lite Architecture.....	5
An In-Depth View of the Oracle Database Lite Synchronization Process.....	6
Increasing Performance With Queues .....	7
Avoiding Record Contamination With Conflict Resolution.....	7
Components Work Together To Ensure Seamless Synchronization7	
Propagating Client Updates to the Server Database .....	8
Methods for Initiating Synchronization In Your Application.....	10
Specifying Events or Conditions for Automatic Synchronization .....	10
Automatic Synchronization Platform Rules.....	11
Automatic Synchronization Publication Rules.....	13
Customizing the Synchronization Process for your Enterprise .....	13
Deciding on Synchronization Refresh Option .....	14
Fast Refresh.....	14
Complete Refresh .....	14
Queue-Based Refresh.....	15
Forced Refresh.....	15
Using Sequences over one or Multiple Clients .....	15
Data Subsetting for each Client.....	17
Create Indexes or Views for a Client.....	17
Using Indexes .....	17
Using Views .....	18

# Oracle Database Lite: Synchronizing Data Between a Device and an Oracle Database

## **CHALLENGES SOLVED BY SYNCHRONIZING REMOTE MOBILE DATA**

In designing an enterprise system that includes the mobile capability, you may face the following business challenges:

- How do I provide my mobile workers with the latest data?
- How do I take the work my mobile workers have done and upload it to the main office enterprise system?

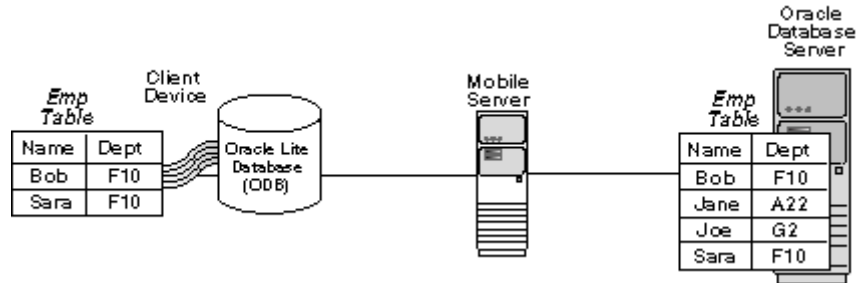
Oracle Database Lite can provide solutions for these challenges and remove some of the manual processes performed out in the field. In the past, you may have manually entered information in the field and then re-entered the data in the enterprise database once you returned to the corporate environment. With Oracle Database Lite, you can manually capture the data once—in the field by collecting data in 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 can be brought down to the client device.

Remote synchronization 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.

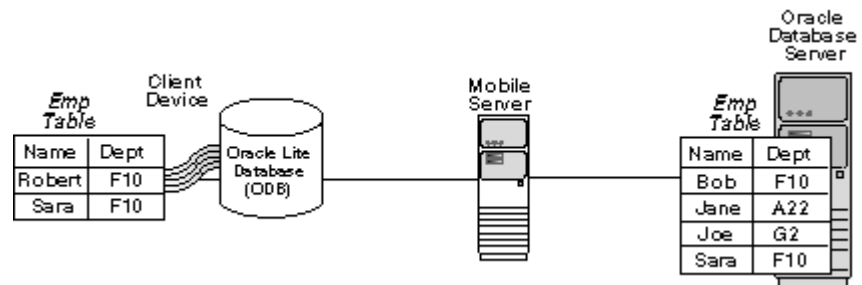
## **WHAT IS SYNCHRONIZATION OF DATA?**

This section describes synchronization and how remote data modified in database tables on a client can be uploaded to a server database.

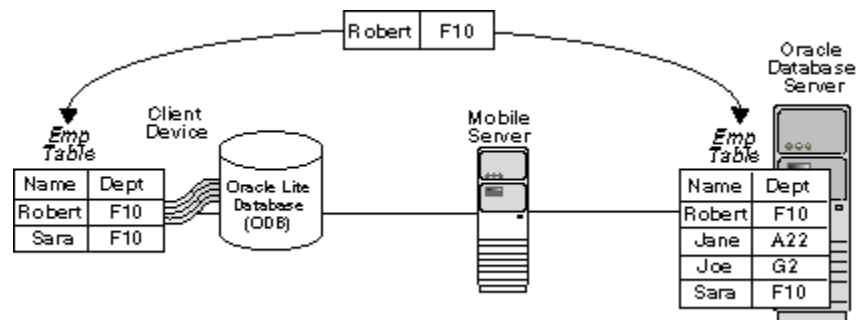
This example uses a customer table, where the customer name and territory number are the columns. The server database contains all customers and their territory numbers; however, the client user—who is the sales manager—only downloads customers from territory F10, as shown below:



The sales manager discovers that Bob would rather be called Robert. So, the sales manager updates the customer name field to Robert on the client device to ensure that everyone who calls on this customer uses the correct name. Notice that Bob's record is now different on the client than on the server.



In order to update the server database with his/her modifications, the sales manager synchronizes with the back-end server database, which means that the change to the customer record is now updated on the Oracle database server. This is shown below:



## OVERVIEW OF ORACLE DATABASE LITE SYNCHRONIZATION

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. However, this only works for a single user interacting with the Palm Pilot back-end. The main difference between the Palm Pilot and Oracle Database Lite synchronization is that

Oracle Database Lite synchronization is used and streamlined for multiple remote clients—rather than a single user.

With Oracle Database Lite, you can create an application where multiple users enter data on their client devices and the data is synchronized with a back-end Oracle database. In addition, only the data designated for each user is downloaded from the server to the client's device. 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. You can even create applications for hardware that collect statistics and can be either synchronized automatically, when an event occurs, at defined intervals, or when polled. This type of implementation could be used on a vending machine, where remote synchronization can keep the office continually informed of how many items are left within the machine.

In order to accommodate 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 uses queues to manage the information between the Mobile clients and the application tables in the database. In addition, the synchronization is asynchronous, which means that change propagation is not immediate. The benefit, however, is that the clients do not stay connected for long while the changes are applied.

## **ORACLE DATABASE LITE ARCHITECTURE**

The Oracle Database Lite 10g architecture consists of the Mobile client, the Mobile Server as the middle tier, and the Mobile repository in the back-end Oracle database server. The Mobile client contains the Oracle Lite database and client applications, which reside on the Mobile device. The Mobile Server acts as the middle tier to coordinate the synchronization process and provide management tools for the administrator. The Mobile Server repository resides in the back-end Oracle database server and is where all data from all users is stored. The Mobile Server uses synchronization to replicate data between the Mobile clients with their client Oracle Lite databases and the application tables, which are stored on a back-end Oracle database.

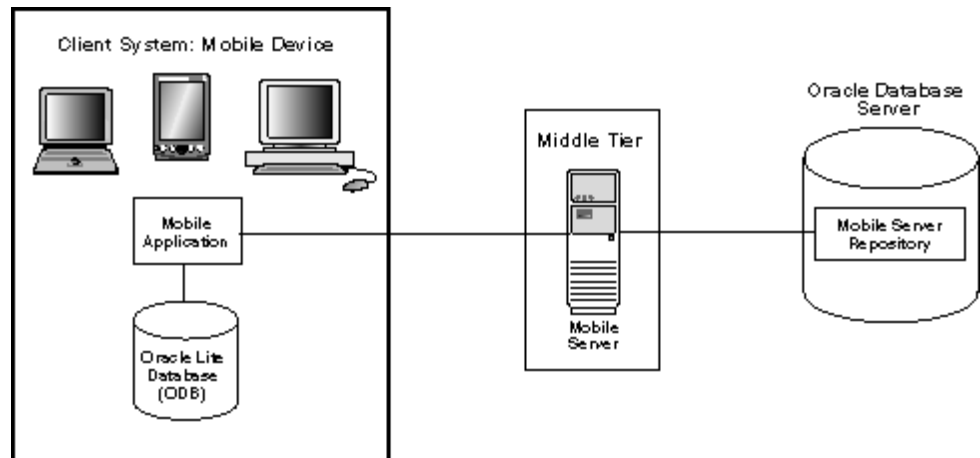
Oracle Database Lite uses a synchronization model that is customizable, maintains data integrity and provides high security between the Mobile Server and the Mobile client.

A simplified view of Mobile synchronization is as follows:

- On the client—The Mobile application communicates with the Mobile Server and uploads the changes made on the client to the Mobile Server. The Mobile client then downloads any new changes for the client from the Mobile Server.

- On the Mobile Server— Oracle Database Lite uses the Mobile Server to replicate data between the Mobile client databases and the application tables, which are stored on a back-end Oracle database.

This description of how synchronization is performed within the main components of Oracle Database Lite as demonstrated in the following graphic:



On the client device, data is stored in a special type of relational table, called a *snapshot table*. A snapshot table behaves identical to a regular relational table, but has functionality that enables tracking changes made to the table.

A SQL query, called the *publication item*, can determine the record set that is downloaded to the snapshot table. The publication item is executed against the server database. The result set of the query defines the structure (columns) of the snapshot table on the client device as well as its rows.

For the sales manager example used earlier in this paper, the SQL query would be as follows:

```
select name, dept from emp where dept = 'F10'
```

A collection of publication items is called a *publication*, which corresponds to a single Oracle Lite database on a client device. All snapshot tables that are based on publication items part of a single publication are stored in the same Oracle Lite database.

### **An In-Depth View of the Oracle Database Lite Synchronization Process**

In reality, our synchronization process uses asynchronous communication and queues to ensure that multiple users can be synchronizing at the same time as well as ensuring data integrity and performance.

### **Increasing Performance With Queues**

When multiple users concurrently synchronize data changes for the same records, then system performance may degrade as all users must wait on each other to obtain and release locks on those records. By using queues and asynchronous communication, Oracle Database Lite enables the user to enqueue the updates for processing and then to download any updates from the server without locking the records. Thus, performance is optimized and all clients can work simultaneously. The downside to using queues is that the changes are not immediately updated. Instead, the updates occur when the queue is read and processed on the server-side.

### **Avoiding Record Contamination With Conflict Resolution**

When both the server and the client can modify the same record, then you could experience record contamination without some form of conflict resolution. How do you decide what changes are valid if multiple users have the same record and update them simultaneously? Which is the correct update if both the server and the client modify the same record? You solve conflicts by configuring in the Mobile Manager who wins if there is a conflict: the server or the client.

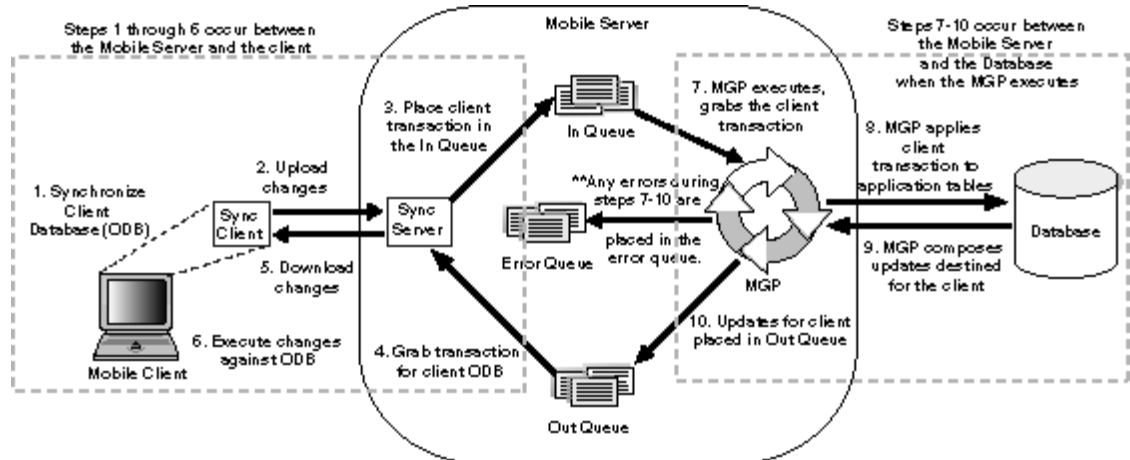
### **Components Work Together To Ensure Seamless Synchronization**

The following graphic and steps describes how all components work in conjunction to create a seamless synchronization from multiple clients. This graphic introduces the following new minor components:

- **Sync Client and Sync Server:** The Sync Client and Sync Server work in conjunction to upload Mobile client changes to the Mobile Server In Queue. The Sync Client resides on the Mobile client; the Sync Server resides on the Mobile Server. The Sync Server places the client modifications into the In Queue and downloads any new data from the server for the client from the Out Queue.
- **Message Generator and Processor:** A background process called the Message Generator and Processor (MGP) runs in the same tier as the Mobile Server, periodically collects all the uploaded changes from the multiple users out of the In-Queue and applies these changes to the repository in the back-end server database. The MGP executes independently and periodically based upon an interval specified in the Job Scheduler in the Mobile Server.

The MGP prepares any modifications that are sent to each Mobile user and places them into the Out Queue. The next time that the Mobile user synchronizes with the Mobile Server, these changes can be downloaded to the client and applied to the client database.

The following steps and graphic demonstrate how Oracle Database Lite accomplishes a successful synchronization:



1. Synchronization is initiated on the Mobile client either by the user or from automatic synchronization.
2. Mobile client software gathers all of the client modifications and the Sync Client uploads them to the Sync Server on the Mobile Server.
3. Sync Server places the modifications into the In-Queue.
4. Sync Server gathers all modifications destined for the Mobile client from the Out-Queue.
5. Sync client downloads all modifications for the client Oracle Lite database (also known as the ODB file).
6. Mobile client applies all changes for client Oracle Lite database. If this is the first synchronization, the Oracle Lite database is created.
7. All modifications uploaded by all Mobile clients are gathered by the MGP out of the In-Queue.
8. The MGP executes the apply phase by applying all modifications for the Mobile clients to their respective application tables to the back-end Oracle database.
9. MGP executes the compose phase by gathering the client data for each Mobile client.
10. MGP places the composed data for Mobile clients into the Out-Queue, waiting for the next client synchronization for the Sync Server to gather the updates to the client.

### PROPAGATING CLIENT UPDATES TO THE SERVER DATABASE

Oracle Database Lite does not keep a record of the SQL operations executed against the client database; instead, all modifications are merged so that only the

final changes are saved and synchronized to the back-end database. For example, if the following was executed on the Oracle Lite database:

1. Insert an employee record 4 with name of Joe Judson.
2. Update employee record 4 with address.
3. Update employee record 4 with salary.
4. Update employee record 4 with office number
5. Update employee record 4 with work email address.

When synchronization occurs, all modifications are captured and only a single insert with the primary key, name, address, salary, office number and email address is performed on the back-end database. Even though the data was created with multiple updates, the Sync Server only takes the final result and makes a single insert.

Once the modifications are synchronized to the back-end database, then the client modifications are applied to the server tables, as follows:

The operations for each publication item are processed according to the specified table weight. The publication creator assigns a “table weight” to each publication item within a specific publication, which defines in what order to process each publication item within the publication. For example, if three publication items exist, where one publication item is based on the `emp` table, one that is based on the `dept` table, and one that is based on the `mgr` table, then you can define the order in which the modifications associated with each publication item are executed. In our example, assign the weight of 1 to the publication item for `dept`, weight of 2 to the publication item for `mgr`, and weight of 3 to the publication item for `emp`. In doing this, you ensure that the publication item that is based on the master table `dept` is always processed first, followed by the publication item for `mgr`, and lastly by the publication item for `emp`.

1. Within each publication item being processed, the SQL operations are processed as follows:
  - a. Client INSERT operations are executed first, from lowest to highest table weight order.
  - b. Client DELETE operations are executed next, from highest to lowest table weight order.
  - c. Client UPDATE operations are executed last, from highest to lowest table weight order.

## **METHODS FOR INITIATING SYNCHRONIZATION IN YOUR APPLICATION**

When creating the publication for your application, you can configure whether you want the updates to be synchronized when the user manually requests it or automatically when certain events or conditions occur.

You can specify if the synchronization occurs automatically or by manual request, as follows:

- **Manual synchronization**—A client manually requests a synchronization either directly through the Oracle Database Lite Mobile Synchronization (msync) application or by the application invoking the Synchronization API.
- **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 the records, so you can continue to work while the synchronization happens in the background. Thus, synchronization occurs 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 timely, and occur at the moment you need it to be uploaded.

### **Specifying Events or Conditions for Automatic Synchronization**

You enable automatic synchronization at the publication item level. Once you have enabled a publication item to use automatic synchronization, you must define the rules under which the automatic synchronization executes. The circumstance under which an automatic synchronization occurs is defined within the synchronization rules.

You can define the rules for automatic synchronization within certain parts of the normal snapshot setup and platform configuration, as follows:

- **Publication level:** Within the publication, you specify the rules under which the synchronization occurs for all publication items enabled for automatic synchronization in that publication.
- **Platform level:** Some of the rules are specific to the platform of the client, such as battery life, network bandwidth, and so on. These rules apply to all

enabled publication items that exist on this particular platform, such as WinCE.

There are two general types of rules: events and conditions.

- Events—An event is variable, as follows:
  - Data events: For example, you can specify that a synchronization occurs when there are a certain number of modified records in the client database.
  - System events: For example, you can specify that if the battery drops below a predefined minimum, you want to synchronize before the battery is depleted.
- Conditions—A condition is an aspect of the client that needs to be present for a synchronization to occur. This includes conditions such as battery life or network availability.

If an event is true, it starts synchronization; however, the synchronization cannot occur unless all conditions are true, as well. This evaluates as follows:

```
When EVENT and if (CONDITIONS) then SYNC;
```

If any one event is true, then the automatic synchronization is initiated the first time the event occurs. For example, if the battery runs below the percentage you specified, the automatic synchronization occurs. As the battery continues to deplete, you will not trigger another synchronization event.

Secondly, if an Automatic Synchronization is about to start, Oracle Database Lite evaluates the conditions to determine if the synchronization can continue. If the condition is not true, the synchronization cannot proceed. For example, if you configured a condition that synchronization can only occur if the battery level is greater than 30% and an automatic synchronization is about to start, but the battery level is at 20%, then this synchronization is queued until the conditions are met.

You define automatic synchronization rules for one or more publication items within a publication. However, you can have some publication items that are enabled to use automatic synchronization and others that only use manual synchronization within the same publication. However, for those that have automatic synchronization enabled, then an automatic synchronization may be occurring when the client requests a manual synchronization. In this case, the manual synchronization stops the automatic synchronization so that all snapshots for all publication items are synchronized.

#### **Automatic Synchronization Platform Rules**

You can specify rules that apply to publications that are enabled for automatic synchronization for a given platform. As described above, there are two types of rules for the platform: system events and conditions. In addition, you can configure the network to be used and in what order to use different networks, if more than one is defined.

These are described in the following sections:

#### ***Platform System Events Defined for Automatic Synchronization***

The following system events will trigger an automatic synchronization if true. You enter the value to define the rule specified by <number>.

- Synchronize when the network bandwidth is greater than <number> Kb/second. Where <number> is an integer that indicates the bandwidth in KB/seconds. When the bandwidth becomes available, the synchronization occurs.
- Synchronize when the battery level drops to <number>%, where <number> is a percentage. Often you may wish to synchronize before you lose battery power. Set this to the percentage of battery left, when you want the synchronization to automatically occur.
- Synchronize when the AC power is detected. Select this checkbox if you want the synchronization to occur when the device is plugged in.
- Synchronize at a specific time or time interval. You can configure an automatic synchronization to occur at a specific time each day or as an interval.

#### ***Platform Conditions for Automatic Synchronization***

The following conditions must be true for this platform for any automatic synchronization to occur:

- Synchronize only if the battery level is greater than <number>%, where <number> is the percentage of battery level left. Sometimes you may not want synchronization to occur and use up what battery you may have left. Thus, you can specify a minimum at which point you do not want this feature to occur. This condition must be true in order for an automatic synchronization to occur.
- Data Conditions: You could have defined records in your snapshot with a data priority of HIGH (value of 0) or LOW (value of 1). Use this condition to specify under what conditions the different priority records are synchronized. By default, the value is LOW, which is synchronized last. If you have a very low network bandwidth and a high ping delay, you may only want to synchronize your HIGH priority data.

#### ***Platform Network for Automatic Synchronization***

The network configuration for the automatic synchronization enables you to provide any proxy server configuration; that is, if your network provider requires that you specify a proxy server to access the internet.

You could have two types of networks: always-on, which may include a proxy server, and dial-up. You can specify in what order these networks should be tried

when automatic synchronization is initiated. For dial-up, Oracle Database Lite can automatically establish the network connection before initiating the synchronization.

#### **Automatic Synchronization Publication Rules**

It is within the publication item that you enable automatic synchronization. Once enabled, all rules defined within the publication and within the platform are evaluated before an automatic synchronization is initiated for this publication item.

Within the publication, you may define the following rules that apply to all publication items enabled for automatic synchronization:

- **Client commit**—Upon commit to the Oracle Lite database, the Mobile client detects the total number of record changes in the automatic synchronization log. If the number of modifications is equal to or greater than the pre-defined number, automatic synchronization occurs.
- **Server MGP compose**—If after the MGP compose cycle, the number of modified records for a user is equal to or greater than your pre-defined number, then an automatic synchronization occurs. Thus, if there are a certain number of records contained in an Out Queue destined for a client on the server, these modifications are synchronized to the client.

### **CUSTOMIZING THE SYNCHRONIZATION PROCESS FOR YOUR ENTERPRISE**

The sections above describe how the synchronization process occurs with the Sync Client/Sync Server, MGP and the Queues. By default, Oracle Database Lite has provided a simple option to initiate synchronization for your application. If you use all of the components that are available, then any data that you can describe in a query can be turned into a snapshot on the client automatically. Depending on the complexity of the query, the snapshot will either be updateable or read-only. For updateable snapshots, all updates to and from the clients are automatically applied to the server with no additional programming required.

However, not all enterprises have the same requirements. So, the following development options are provided for you to customize how synchronization occurs:

- **Data Collection Queues**: If you have an environment where only data is uploaded from the client and never downloaded from the server, then you can optimize your synchronization performance by customizing the application to use data collection queues.
- **Customize Methods Around MGP Functionality**: For your environment, you may want to have certain tasks performed before and after each phase of synchronization as the MGP is performing its function. We provide callback functions for you in the CALLBACK PL/SQL class to provide your own implementation.

- **Queue-Based Synchronization:** For your environment, you may want to control the full spectrum of the apply/compose phases, which the MGP component normally performs for you. You can customize the apply/compose phases for your application using the queue-based synchronization method.
- **MyCompose Class:** The compose phase takes a query for one or more server-side base tables and puts the generated DML operations for the publication item into the Out Queue to be downloaded into the client. All DML operations using the physical DML logs on the server-side base tables are managed. This can be resource intensive if the DML operations are complex—for example, if there are complex data-subsetting queries being used. You can customize this process with the extendable `MyCompose` class, which contains methods which can be overridden.

## **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:

### **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.

### **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.

## Queue-Based Refresh

Developers create 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.

## Forced Refresh

This is not a refresh option; however, 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.

## USING SEQUENCES OVER ONE OR MULTIPLE CLIENTS

A sequence is a database object, from which you can generate unique integers. You can use sequences to automatically generate primary key values. However, when you have multiple clients accessing a single server, you need a method to guarantee unique identifying numbers for new records from multiple clients.

You can create a sequence for generating unique sequence numbers for client and server records. For Oracle Database Lite, you can add a sequence to a publication; then, the sequence is created on all subscribed clients during the initial synchronization. On each client Oracle Lite database, the sequence can be used independently. However, since the sequences are used to generate primary key and unique key values for snapshot tables, it is important to ensure that different clients do not generate the same sequence values. If they did, then conflicts may occur when the clients synchronize their changes to the server tables.

The Sync Server guarantees sequence number uniqueness across all clients. During synchronization, the Sync Server assigns separate sequence ranges, known as a

window, to each client when necessary. A client cannot increment a sequence beyond its current window. Once a client exhausts its window, the Sync Server assigns a new window on the next synchronization.

Since the sequence windows are obtained from the Sync Server during synchronization only, there is a chance that the client could exhaust all available sequence numbers in its window in between synchronization events. The administrator can configure a threshold value, which tells the Sync Server to provide the client a new window during the next synchronization before the current one is exhausted. For example, you set the window to 200 and the threshold to 25. During a synchronization event, if the Sync Server notices that the current sequence value is greater than or equal to 175, then it allocates the next window for the client.

The following describes how to use Oracle Database Lite sequences:

- Only clients use the sequence: If you have more than a single client, you want each client to use a specific range of unique sequence numbers, so that none of the records have duplicate sequence numbers.

Specify the start value, the window size, and select the offline only checkbox. When you define the size of the window, you provide the Sync Server the number of assigned identifiers for each client and the range of values never overlaps with those of other clients.

- Server and clients use the same sequence: If you want the server and one or more clients to share a sequence, then the server and the client use every other number—the identifiers are generated where the server uses all even numbers and the clients use odd numbers—or vice versa, if the start value of the sequence is an even number. Specify the start value, window size and de-select the Offline Only checkbox, which tells Oracle Database Lite to generate a server-side sequence. The increment value always defaults to 2 for this case, even if you specify another number.

A sequence of the supplied name is created automatically on the server in the `mobileadmin` schema to use all even numbers. The clients use odd values within the window assigned to each client. Thus, the server and client sequence values are unique.

For example, the sequence number for the first client starts at 1 and increments by 2 for all of its sequence numbers. The first client still has a window size, which in this example is 100, but it starts with an odd number within that window and always increments by 2 to avoid any positive numbers. Thus, client A has the window of 1 to 100, but the sequence numbers would be 1, 3, 5, and so on up to 99.

Oracle Database Lite creates and maintains the sequence based on the sequence definition in the publication. Once you create a sequence in the project, you can associate it with a publication.

## DATA SUBSETTING FOR EACH CLIENT

What is Data Subsetting? When you set up your publication item, you may have set up an input parameter in the SQL query that defines what snapshot of data is to be retrieved for each user. Then, for each user, a value is provided for the input parameter that is used to retrieve a separate result set for each specific 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 for each manager. The following SQL query retrieves all customers from the CUST table, where the sales manager (salesmgr) identifier matches the input parameter designated by my\_mgrid.

```
Select * from cust where salesmgr = :my_mgrid
```

The my\_mgrid is an input parameter that uniquely identifies each sales manager. This input parameter is known as the data subsetting parameter. Then, by providing the unique identifier for the sales manager, then the application retrieves data solely for that user. In this way, the user does not receive any information during synchronization that he/she should not be using, but does retrieve all information that is relevant to him/her.

The administrator configures this information within the user information on the Mobile Server. The query recognizes the data subsetting parameter as an input parameter when you precede the parameter with the colon (:). For each sales manager, the administrator configures within Mobile Manager the correct manager id (my\_mgrid) for each sales manager.

## CREATE INDEXES OR VIEWS FOR A CLIENT

You can use both indexes and views on a client, as described in the following sections.

### Using Indexes

The Mobile Server supports automatic deployment of indexes in Oracle Database Lite on clients. The Mobile Server automatically replicates primary key indexes from the server database.

The following is a list of the types of indexes you can create or deploy:

- You can create indexes on any columns in your application table.
- You can create an index with multiple columns.
- You can automatically deploy regular and unique indexes to clients.
- You can create secondary indexes on a publication item. If you do not want the primary index, you must explicitly drop it from the publication items.

## Using Views

Publication items can be defined for both tables and views. When publishing updatable multi-table views, the following restrictions apply:

- The view must contain a parent table with a primary key defined.
- `INSTEAD OF` triggers must be defined for data manipulation language (DML) operations on the view.
- All base tables of the view must be published.



Oracle Database Lite: Synchronizing Data Between Device and Oracle Database  
August 2007

Oracle Corporation  
World Headquarters  
500 Oracle Parkway  
Redwood Shores, CA 94065  
U.S.A.

Worldwide Inquiries:  
Phone: +1.650.506.7000  
Fax: +1.650.506.7200  
[oracle.com](http://oracle.com)

Copyright © 2007, Oracle. All rights reserved.

This document is provided for information purposes only and the contents hereof are subject to change without notice.

This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. We specifically disclaim any liability with respect to this document and no contractual obligations are formed either directly or indirectly by this document. This document may not be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without our prior written permission. Oracle, JD Edwards, PeopleSoft, and Siebel are registered trademarks of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.