Database Link User's Manual

Release 6.1.1

ALTIBASE

Contents

Preface

About This Manual

This manual contains information to help you understand Database Link concepts and use Database Link. We trust that this manual will be of great help. This preface contains the following:

- Who This Manual is For

- Targeted Software

- How This Manual is Structured

- Documentation Conventions

- Related Reading

- On-line Manuals

- Altibase Welcomes Your Comments

Who This Manual is For

This manual is intended for the following ALTIBASE HDB users:

- Database Administrators

- Performance Managers

- Database Users

- Application Developers

- Technical Assistance Team

In order to fully understand the contents of this manual, we recommend that you have the following background knowledge:

- A working knowledge of your computer, your operating system, and the utilities provided with your operating system

- Some experience working with relational databases or exposure to database concepts

- Some experience with computer programming

- Some experience with database server administration, operating system administration or network administration

Targeted Software

This manual assumes that your database server is ALTIBASE HDB server, Version 6.

How This Manual is Structured

This manual covers the following topics:

- Chapter1.Introduction to Database Link

  This chapter introduces and provides an overview of Database Link.

- Chapter2.Supported Objects, SQL Statements and Data Types

  This chapter covers the objects, SQL commands and data types supported by Database Link.

- Chapter3.Configuration and Operation of Database Link

  This chapter discusses how to set up the Database Link environment and how to start up Database Link.

- Chapter4.Database Link-Related SQL Statements

  This chapter lists all of the features provided by Database Link, and describes how to create and drop a Database Link instance.

- Appendix A. Property and Data Dictionary

  This appendix lists all of the properties and the data dictionary related to Database Link.

Documentation Conventions

This section explains the following documentation conventions. These conventions make it easier to glean information from the ALTIBASE HDB manuals.

- Command-Line Conventions

- Typographical Conventions

Command-Line Conventions

This section defines and illustrates the format of commands that are available in Altibase products. These commands have their own conventions, which might include alternative forms of a command, required and optional parts of a command, and so forth.

Elements	Meaning
Reserved word	The command starts. A syntax element which is not a complete command starts with an arrow.
⟶	The command continues on the next line. A syntax element which is not a complete command terminates with this symbol.

Elements	Meaning
	The command continues from the previous line. A syntax element which is not a complete command starts with this symbol.
	The end of a statement.
	Mandatory
NOT	Optional
ADD / DROP	A mandatory field comprising multiple options. One, and only one, of the options must be specified.
ASC / DESC	An optional field comprising multiple options. Only one of the options may be specified.
ASC / DESC ,	An optional field comprising multiple options. More than one of the options may be specified. A comma must precede every repetition .
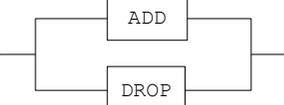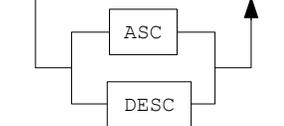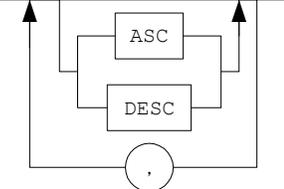
Typographical Convetions

This manual uses the following standard set of conventions to introduce new terms, illustrate screen displays, describe command syntax, and so forth.

Rules	Meaning	Example
[]	Indicates an optional field.	```VARCHAR [(size)] [[FIXED] VARIABLE]```
{}	Indicates a mandatory field for which one or more items must be selected.	```{ ENABLE	DISABLE	COMPILE }```
|	A delimiter between optional or mandatory arguments.	```{ ENABLE	DISABLE	COMPILE } [ENABLE	DISABLE	COMPILE]```
. . .	Repetition of the previous argument. Indicates another iteration of the preceding parameter. Indicates that some example code has been omitted.	```iSQL> select e_lastname from employees; E_LASTNAME ---------------------- Moon Davenport Kobain . . . 20 rows selected.```
Other Symbols	Symbols other than those shown above	```EXEC :p1 := 1; acc NUMBER(11,2);```
Italics	In the main text, new terms and emphasized words appear in italics. In statement definitions, values that you are to specify appear in italics.	```SELECT * FROM table_name; CONNECT userID/password;```
Lower Case Letters	Indicate program elements set by the user, such as table names, column names, file names, etc.	```SELECT e_lastname FROM employees;```
Upper Case Letters	Keywords and all elements provided by the system appear in upper case.	```DESC SYSTEM_.SYS_INDEX_;```

Related Reading

For additional technical information, consult the following manuals.

- Administration Getting Started

- Administration Administrators' Manual

- Administration Replication Manual

- Application Development SQL Reference

- Application Development ODBC Reference

- Application Development Spatial SQL Reference

- Application Development Application Program Interface Users' Manual

- Tools iSQL Users' Manual

- Message Error Message Reference

On-line Manuals

Manuals (PDF and HTML) in Korean and English are available at the Altibase Customer Support (http://support.altibase.com/).

Altibase Welcomes Your Comments

Please let us know what you like or dislike about our manuals. To help us with future versions of our manuals, please tell us about any corrections or clarification that you would find useful. Please include the following information :

- The name and version of the manual that you are using

- Any comments that you have about the manual

- Your name, address, and phone number

For immediate assistance regarding technical issues, please contact the Altibase Customer Support (http://support.altibase.com/).

Thank you. We appreciate your feedback and suggestions.

1 Introduction to Database Link

This chapter explains what Database Link is, explains relevant concepts, and lists the features provided with Database Link. It contains the following sections:

1.1 What is Database Link?

This section introduces some concepts and terminology that will be of help in developing a conceptual understanding of Database Link. This section briefly introduces the following topics:

- 1.1.1 Concepts

- 1.1.2 Terminology

- 1.1.3 How to Implement Database Link

- 1.1.4 System Environment

1.1.1 Concepts

Database Link is a technology for linking to database servers that meet the following criteria so that users can integrate material provided on different servers whenever they wish.

The database servers to be connected should meet the following criteria:

- The database servers should have a logical relationship between them.

- The database servers should be connected to a computer communication network.

- The database servers should be physically separated.

"Logical relationship" means that the data structures on the servers should be consistent, and that the user should have sufficient privileges to perform the desired operations. In the case where it is desired to unite disparate data spread across multiple servers, the user must be able to provide a centralized system for this purpose.

The requirement that the database servers be connected to a computer communication network means that they must be linked to each other via hardware and software. This could range from intranet nodes on the same subnet to being located some distance from one another via a WAN or the Internet.

Being physically separated means that the main purpose of the servers is to provide data service, that is to say, the database servers must be constructed such that they can be operated independently of one another. This includes everything from hardware located in close proximity to completely independent sites.

1.1.2 Terminology

Local Server

This is the database server that creates and uses a Database Link object. The local server reformats a user's queries, sends them to a remote server for execution, and reformats the received results to match the original query. ALTIBASE HDB is the only local server that is capable of running Database Link.

Remote Server

This is the destination database server that receives requests from the local server via Database Link

and sends the results back to the local server. ALTIBASE HDB or any relational database that supports ODBC can be used as the remote server.

AltiLinker

A separate process exists to intermediate the data exchange between Database Link and the remote server. This process is called AltiLinker[1]. When the DBLINK_ENABLE property is set to 1, AltiLinker starts up when the server starts up and shuts down when the server shuts down. However, the user can use the ALTER DATABASE LINKER statement to manually shut down or restart AltiLinker.

Location Descriptor

The location descriptor must be used in order to use Database Link in queries. The location descriptor consists of an "@ " (ampersand character) between the object name and the link.

The Database Link location descriptor is used only in the FROM clause of a SELECT SQL statement.

```
SELECT * FROM emp@link1;
```

1.1.3 How to Implement Database Link

Database Link provides a link between independently operated DBMSs, which can be of different types.

Figure 1-1 How to Implement Database Link

---
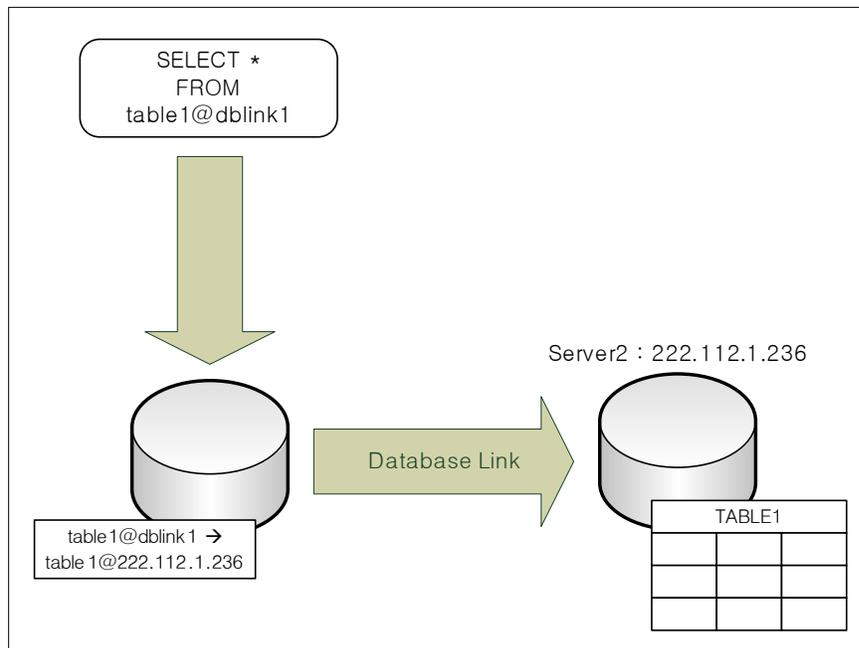
1.   Because the AltiLinker process is created with the same name as the ALTIBASE HDB server process, the number of processes with the name 'altibase' increases by 1 for each Altilinker process that is in operation.

1.1 What is Database Link?

[Figure 1-1] illustrates the overall process of implementing Database Link. When the user executes a query, after checking whether a table corresponding to the specified destination table exists on the remote server, the results are retrieved from the remote server and the final query results are returned.

1.1.4 System Environment

Segregated servers should provide independent operating environments in order to implement Database Link. In other words, the system should be designed so that queries that pertain to data on respective servers and are stored on a local server are capable of being executed even in the absence of Database Link. For example, if information about a manufacturing process or a product is stored on one server, the system should be designed so that processing can take place without involving other servers when a transaction related to product information occurs.

The following figure roughly illustrates the Database Link operating environment. A large number of sites or servers provide independent operating environments and are interconnected through a computer communication network. Each of them has its own database. Queries can be sent to other sites and servers for execution through the computer communication network.

Figure 1-2 Operating Environment of Database Link
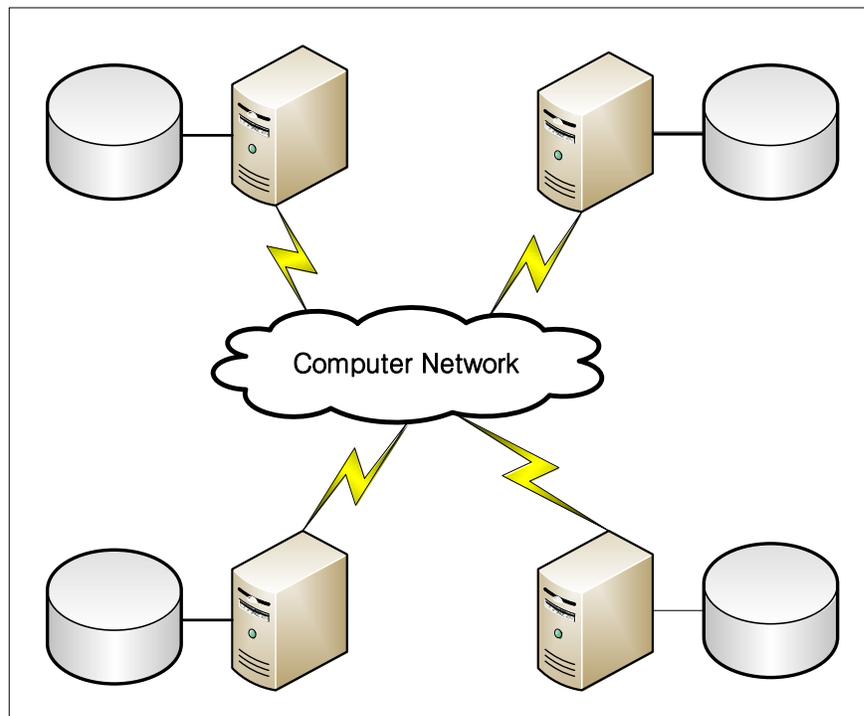
1.2 Benefits of Database Link

Database link realizes the following benefits:

- 1.2.1 Convenience

- 1.2.2 Efficiency

- 1.2.3 Scalability

- 1.2.4 High Availability

1.2.1 Convenience

Database Link realizes transparency with respect to the location of remote servers. In other words, the client application does not need to know the location of the original data used to generate the end result.

Figure 1-3 An Application Accessing Respective Remote Servers
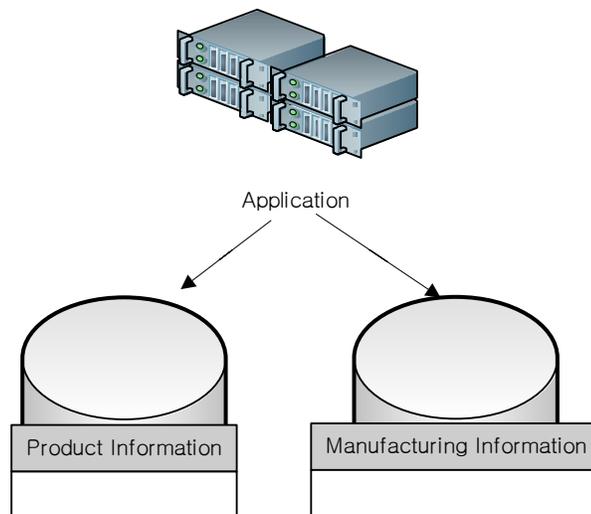
For example, assume that product information is stored in Site1 and manufacturing information is stored separately in Site2, as in Figure 1-4. When information about the manufacture of a certain product is desired, the client application gets product information from Site1 and manufacturing information from Site2, respectively, and then combines the data from Site1 and Site2 to construct the required information.

Figure 1-4 An Application Accessing Servers via Database Link
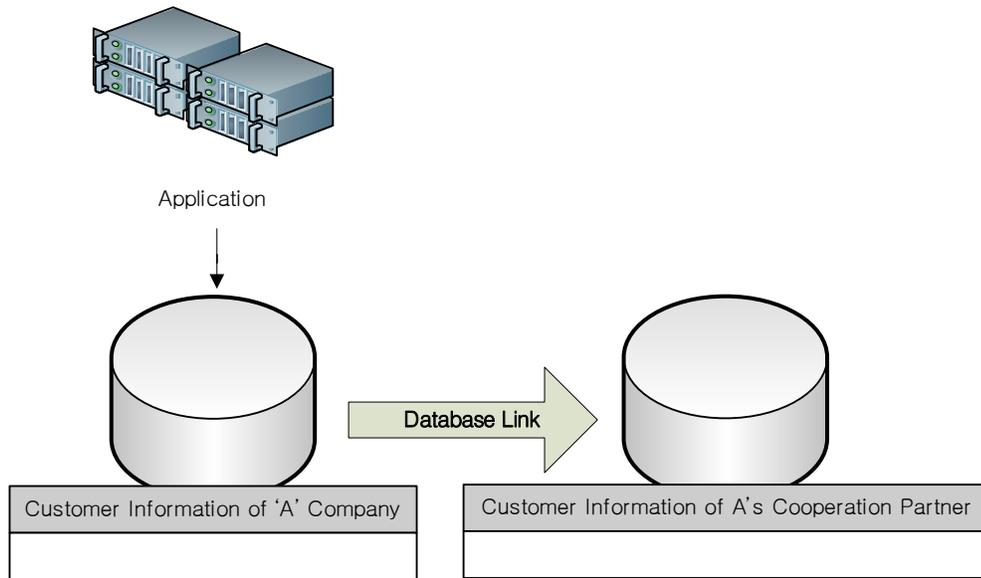
In contrast, when queries are executed on a local server that is part of a system in which Database Link is deployed, the local server displays final results sent from the remote server. In other words, even though the data are physically distributed across multiple servers or sites, they can be accessed as though they were stored in one logical server.

1.2.2 Efficiency

When large amounts of data are centralized in a single location, QoS (Quality of Service) can't be guaranteed because large numbers of simultaneous transactions can have a negative effect on performance.

Figure 1-5 Single-Server Environment
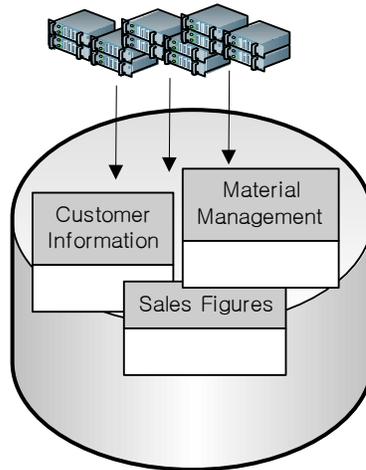
However, when data that are not strongly interrelated are distributed across multiple servers, the workload of serving the data in tables is shared. Furthermore, the data in respective tables can be combined using Database Link. In other words, because largely unrelated transactions take place independently on different servers, any performance degradation is mitigated, and system resources such as CPU, memory and I/O are used more efficiently.

Figure 1-6 Distributed Data Environment
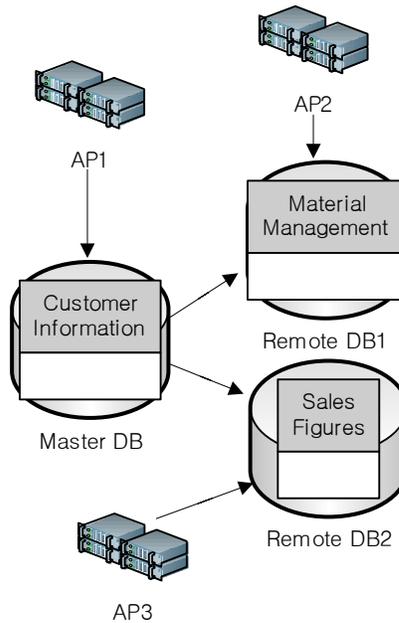
Introduction to Database Link

1.2.3 Scalability

Database Link has more flexible scalability than centralized systems because, when using Database Link, more servers can be flexibly installed when the operating system reaches the limits of its processing capability.

1.2.4 High Availability

Even if a failure occurs in part of a system running Database Link, the rest can continue to function. [Figure 1-7] illustrates a Database Link system in which product information is stored in Site1, manufacturing information is stored in Site2, and shipping information is stored in Site3. In a system structured in this way, if some problem afflicts Site 3, there would still be no problem fulfilling requests that pertain only to manufacturing or product information.

Figure 1-7 Availablity Even in the Event of Partial Failure
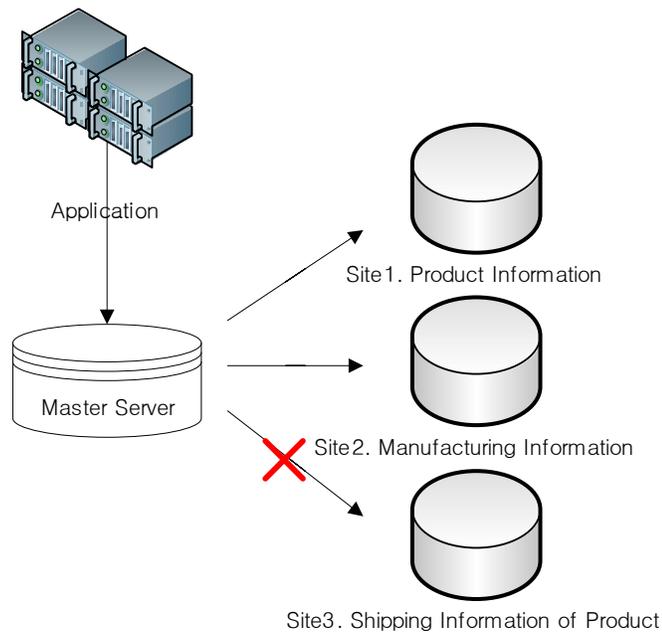
1.3 Restrictions

Database Link can be established for following database systems:

- ALTIBASE HDB version 4.3.9 or higher

- Oracle Database 10g or higher

1.3 Restrictions

2 Supported Objects, SQL Statements and Data Types

This chapter covers the remote schema objects, SQL commands and data types that are supported for use with Database Link. It includes the following sections:

2.1 Accessible Remote Schema Objects

The following table lists the remote schema objects that can be accessed using Database Link.

Table 2-1 Remote Schema Objects Accessible using Database Link

Remote Schema Object	Accessible
Table	O
Index	X
View	O
Stored Procedure	No, but stored procedures on the local server can access the remote server's schema
Sequence	X
Queue	X
Trigger	X
Synonym	X
Constraint	X

The supported objects are described in slightly greater detail below.

2.1.1 Tables

The table is the most fundamental database schema object, and is the place where records are stored. Database Link's essential function is to realize interoperability between disparate tables, so it follows that most Database Link functionality pertains to tables.

2.1.2 Views

Views can join one or more tables so that they appear to users as a single logical table. Although views can be used as structural elements in other views, the actual substructure of any view ultimately comprises a combination of base tables. The basic structural elements of views are tables, which are supported by DB Link, but because views themselves are perceived as logical tables, they are also supported by DB Link.

2.1.3 Stored Procedures

Database Link cannot be used to access stored procedures that are defined on the remote server.

However, stored procedures that are defined on the local server can use Database Link to access tables and views in the remote server's schema. Stored procedures used in this way cannot contain ROWTYPE variable declarations based on tables on the remote server. Additionally, they cannot be used with cursors.

2.2 SQL Statements Supported by Database Link

This section explains which SQL statements are supported by Database Link.

Commands	Supported
DDL	X
DCL	X
SELECT	O
DML (other than SELECT)	X

2.2.1 DDL

Database Link cannot be used to execute DDL (Data Definition Language) on the remote server.

2.2.2 DCL

Database Link cannot be used to execute DCL (Data Control Language) on the remote server. The exception is the COMMIT and ROLLBACK statements, which are useful in the following way:

When a SELECT statement is executed on a remote server using Database Link, a transaction is initiated. This transaction can maintain a lock on objects on the remote server, which can cause other transactions on the remote server (e.g. DROP TABLE) to fail. The COMMIT and ROLLBACK statements can be used to explicitly release such a lock. (The COMMIT and ROLLBACK statements are functionally identical to each other when used in this way.)

2.2.3 DML

ALTIBASE HDB supports the use of Database Link to perform SELECT operations on the remote server, but not INSERT, UPDATE or DELETE operations. Support for SELECT statements extends to everything including selection, join, subquery, aggregation, set and view. In addition, Database Link can be used in a SELECT query nested inside a DDL/DML statement.

SELECT ... FOR UPDATE statements make changes to data, just like INSERT, UPDATE, and DELETE statements, and thus are not supported for use with Database Link.

2.3 Data Types Supported by Database Link

Because Database Link uses the ODBC interface, only the standard data types supported by ODBC are supported for use with Database Link. The following table shows which of the data types defined within ALTIBASE HDB are supported for use with Database Link.

Table 2-2 Data Types for Database Link

ALTIBASE HDB Data Type	Corresponding ODBC Data Type	Supported
CHAR	SQL_CHAR	O
VARCHAR	SQL_VARCHAR	O
NCHAR	SQL_WCHAR	O
NVARCHAR	SQL_WCHAR	O
BIGINT	SQL_BIGINT	O
DECIMAL	SQL_DECIMAL	O
DOUBLE	SQL_DOUBLE	O
FLOAT	SQL_FLOAT	O
INTEGER	SQL_INTEGER	O
NUMBER	SQL_NUMERIC	O
NUMERIC	SQL_NUMERIC	O
REAL	SQL_REAL	O
SMALLINT	SQL_TINYINT	O
DATE	SQL_DATE	O
BLOB/CLOB		X
BYTE		X
NIBBLE		X
BIT		X
GEOMETRY		X

3 Configuration and Operation of Database Link

Database Link uses ODBC to access remote servers. The explanations in this chapter will be made on the basis of an open source Unix-ODBC environment. However, the particulars of each ODBC Driver Manager installation can vary depending on the circumstances and server configuration.

This chapter comprises the following sections:

- 3.1 How to Run Database Link

- 3.2 Configuration

- 3.3 Activating AltiLinker

3.1 How to Run Database Link

3.1.1 Database Link Procedure

Figure 3-1 Database Link Operating Procedure
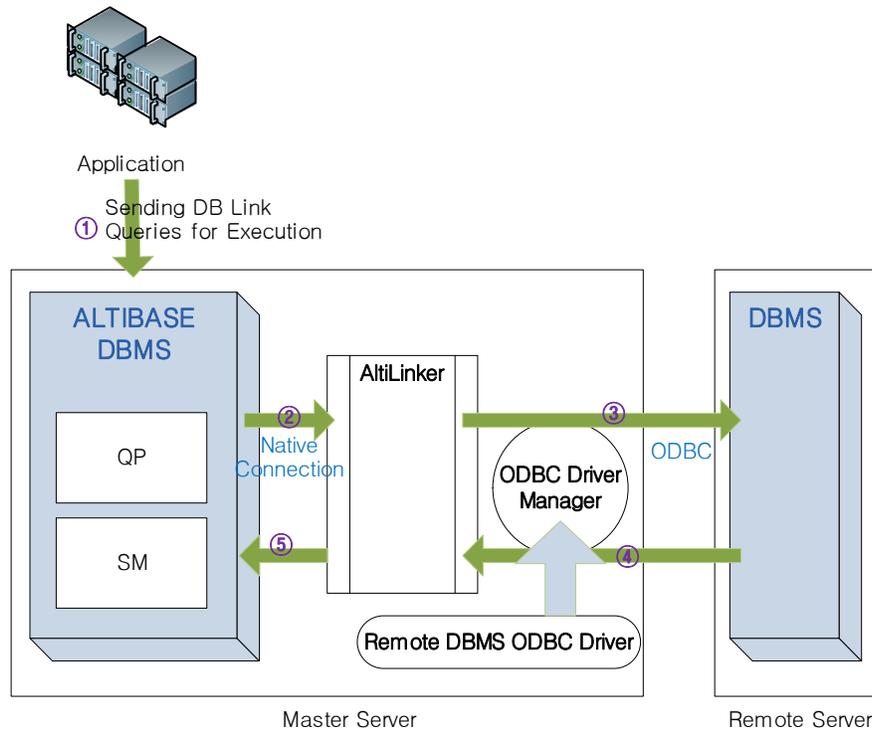
1. The user sends a query containing locator information to the local server.

2. The query processor on the local server parses the query and analyzes the location descriptor. The query to be sent to the remote server designated according to the location descriptor is then rewritten, and an execution plan is made.

   If the storage manager is requested to provide a table scan according to this plan, the storage manager sends a request for a remote server table scan to AltiLinker and waits for the result. Communication with AltiLinker uses a native connection module.

3. AltiLinker passes the query to the relevant remote server for execution. AltiLinker accesses and interacts with the remote server using ODBC. Therefore, an ODBC driver manager must be installed on the local server, and an ODBC driver must be installed on the remote server.

4. The remote server processes the query as requested by AltiLinker and then sends the query results via ODBC. The amount of data that is returned will vary depending on the format of the query, which was rewritten by the query processor on the local server.

5. AltiLinker receives the query results from the remote server and then sends them to the local server one record at a time. The local server converts the data from the ODBC data type into

the ALTIBASE HDB data type and passes them from the storage manager to the query processor. After the query processor collates and filters the data sent from the storage manager, the resultant data corresponding to the original queries are returned.

3.1.2 Supported Communication Protocols

Steps 2 and 5 in the preceding section pertain to the communication between the local server and AltiLinker. The type of native connection to use is chosen by appropriately setting the value of the LINKER_LINK_TYPE property. If LINKER_LINK_TYPE is set to 0, TCP is used for communication, whereas if it is set to 1, a Unix domain socket is used. Note that only TCP is supported for use with Windows.

Setting LINKER_LINK_TYPE to 2 specifies that IPC (Inter-Process Communication) is to be used. However, at present, only TCP and Unix domain are supported, so this setting is reserved for future use.

Only ODBC is currently supported for communication in steps 3 and 4 above. For more detailed information, please refer to 3.2 Configuration.

3.2 Configuration

This section provides instructions on configuration. It covers the following topics:

- 3.2.1 ODBC Driver Manager Installation

- 3.2.2 How to Configure odbc.ini

3.2.1 ODBC Driver Manager Installation

Database Link uses ODBC to enable a local server to access a remote server. The ALTIBASE HDB ODBC driver is included in the installation package. To use it, an ODBC driver manager must additionally be installed in the system. Because ALTIBASE HDB doesn't provide an ODBC driver manager, it must be installed separately.

The Unix ODBC driver manager is one open source product that is available for use in Unix and Linux environments. You can find it at http://www.unixodbc.org/. Other common ODBC driver managers are Data Direct and EasySoft. Windows provides its own ODBC driver manager. The ODBC environment provides a common interface for data access via the ODBC driver manager. However, the system construction can vary depending on the specific driver manager that is used.

Users must add the library path of the installed ODBC driver manager to the environment variable that points at the library paths of the account that is used to run ALTIBASE HDB, and must create an odbc.ini file in the home directory for the ALTIBASE HDB account or in the etc directory in the directory where the ODBC driver manager is installed.

3.2.2 How to Configure odbc.ini

The following DSN must be added to odbc.ini. More than one DSN can be added to odbc.ini.

```
[ODBC Data Sources]
altibase_odbc = Altibase ODBC Driver

[altibase_odbc]
Driver = /home/altibase/altibase_home/lib/libaltibase_odbc-64bit-ul32.so
ServerType = Altibase
Server = 192.168.3.62
Port = 20300
NLS_USE = US7ASCII
Database = mydb
FetchBufferSize = 64
ReadOnly = no
```

In the above example, when the DSN called altibase_odbc is created, the information about the target server to be integrated using Database Link is specified. At this time, either libaltibase_odbc-64bit-ul32.so or libaltibase_odbc-64bit-ul64.so in the $ALTIBASE_HOME/lib/ directory can be chosen as the driver when ALTIBASE HDB is used as the remote database. The reason for this is that the sizes of SQLLEN and SQLULEN vary depending on the kind of 64-bit ODBC manager.

ALTIBASE HDB versions 5.1.5.28 and above support the use of the ALTIBASE HDB 64-bit Unix ODBC driver with both 32-bit and 64-bit SQLLEN. 64-bit server and client packages include the following two drivers.

ALTIBASE HDB 64-bit unix odbc driver	libaltibase_odbc-64bit-ul64.so(SQLLEN is 64bit)
	libaltibase_odbc-64bit-ul32.so(SQLLEN is 32bit)

The driver should be selected depending on the ODBC manager that is installed, according to the following table.

Table 3-1 ODBC Manager

64-bit odbc manager	SQLLEN
unix-odbc 64-bit(~2.2.12)	32-bit
unix-odbc 64-bit (~2.2.12) DBUILD_REAL_64_BIT_MODE	64-bit
unix-odbc 64-bit(2.2.13~)	64-bit
unix-odbc 64-bit (2.2.13~) BUILD_LEGACY_64_BIT_MODE	32-bit
iodbc 64-bit	64-bit
Windows 64-bit	64-bit

3.3 Activating AltiLinker

AltiLinker must be activated in order to use Database Link. To achieve this, the Database Link-related properties in $ALTIBASE_HOME/conf/altibase.properties must be appropriately set.

First, set DBLINK_ENABLE to 1 in order to activate AltiLinker. Then, set LINKER_PORT_NO to the port number to be used when sending or receiving data. LINKER_PORT_NO must be set when LINKER_LINK_TYPE is TCP.

After setting the properties, when ALTIBASE HDB is started up, AltiLinker starts up together with ALTIBASE HDB. For more detailed information about Database Link properties, please refer to the *General Reference*.

4 **Database Link-Related SQL Statements**

This chapter explains how to use the SQL statements that are provided for controlling Database Link, as well as the use of the SELECT statement in detail. It includes explanations of the following statements:

- 4.1 CREATE DATABASE LINK

- 4.2 DROP DATABASE LINK

- 4.4 ALTER SESSION

- 4.5 SELECT

- 4.6 The EXEC_REMOTE Hint

4.1 CREATE DATABASE LINK

To create a Database Link object, use the CREATE DATABASE LINK statement. A Database Link object can have only one remote server as its target.

4.1.1 Syntax

```
CREATE [PUBLIC|PRIVATE] DATABASE LINK dblink_name
WITH ODBC dsn
CONNECT TO user_id IDENTIFIED BY password;
```

4.1.2 Description

Only the SYS user or a user with the CREATE DATABASE LINK system privilege can create a Database Link object.

PUBLIC|PRIVATE

Specifies the Database Link PUBLIC|PRIVATE attribute. Specify PUBLIC to make the created Database Link object available to all users, or PRIVATE to make the Database Link object available only to the creator. If this clause is omitted, the Database Link object defaults to PRIVATE.

dblink_name

This sets the name of the Database Link object to be created.

dsn

This is used to specify the name of the ODBC DSN (Data Source Name).

Because Database Link uses ODBC, in order to access the remote server, an ODBC driver must be installed. Additionally, the details about the DSN on the remote server must be specified in the ODBC environment file. For more information, please refer to 3.2.2 How to Configure odbc.ini.

user_id/password

This is the ID and password of the database user on the remote server. However, If a user ID and password have been set in odbc.ini, those settings will take priority over the ones specified in this statement. Therefore, if you want Database Link to actually use the user ID and password specified in the CREATE DATABASE LINK statement, do not specify a different user ID or password in the odbc.ini file.

Note also that the user specified here must have been granted permission for the relevant target objects on the remote node when accessing them via Database Link. Otherwise, errors related to privileges will occur.

4.1.3 Example

<Query1> *user1/user1* are the ID and password with which to connect to a database on a remote server for which the DSN is altibase_odbc. A user creates a Database Link object with the name *link1*, which only s/he can use.

```
iSQL> CREATE PRIVATE DATABASE LINK link1
```

```
 WITH ODBC altibase_odbc
 CONNECT TO user1 IDENTIFIED BY user1;
```

<Query2> *user1/user1* are the ID and password with which to connect to a database on a remote server for which the DSN is altibase_odbc. A user creates a Database Link object with the name *link2*, which all users can use.

```
iSQL> CREATE PUBLIC DATABASE LINK link2
 WITH ODBC altibase_odbc
 CONNECT TO user1 IDENTIFIED BY user1;
```

4.2 DROP DATABASE LINK

To drop a Database Link object, use the DROP DATABASE LINK statement.

4.2.1 Syntax

```
DROP [PUBLIC|PRIVATE] DATABASE LINK dblink_name;
```

4.2.2 Description

Only the SYS user or a user with the DROP DATABASE LINK system privilege can drop a Database Link object.

dblink_name

This specifies the name of the Database Link object to be dropped.

4.2.3 Restriction

A Database Link object that is currently in use cannot be dropped. A Database Link object can be dropped only if no queries that use it are being executed. If a Database Link object is dropped while queries are being executed, an error will occur.

4.2.4 Example

<Query1> Drop a private Database Link object named *dblink1*.

```
iSQL> DROP DATABASE LINK dblink1;
```

<Query2> Drop a public Database Link object named *dblink1*.

```
iSQL> DROP PUBLIC DATABASE LINK dblink1;
```

4.3 ALTER DATABASE LINKER

To start or stop AltiLinker, use the ALTER DATABASE LINKER statement.

4.3.1 Syntax

```
ALTER DATABASE LINKER START;
ALTER DATABASE LINKER STOP;
```

4.3.2 Prerequisite

Only the sys user can execute this statement, and only when running in sysdba mode.

4.3.3 Description

START

This starts AltiLinker. AltiLinker must not already be running.

STOP

This stops AltiLinker. However, no transaction that uses Database Link objects can exist when Database Link is stopped. If transactions that use Database Link objects exist, execution of this statement will fail.

4.3.4 Example

```
$ isql -u sys -p manager -sysdba
-------------------------------------------------------
 Altibase Client Query utility.
 Copyright 2000, ALTIBASE Corporation or its subsidiaries.
 All Rights Reserved.
-------------------------------------------------------
ISQL_CONNECTION = UNIX, SERVER = 127.0.0.1, PORT_NO = 20300

iSQL(sysdba)> ALTER DATABASE LINKER STOP;
Alter success.
iSQL(sysdba)> ALTER DATABASE LINKER START;
Alter success.
```

4.4 ALTER SESSION

To terminate a Database Link session, use the ALTER SESSION statement.

4.4.1 Syntax

```
ALTER SESSION CLOSE DATABASE LINK;
```

4.4.2 Prerequisite

Only the sys user can execute this statement, and only when running in sysdba mode.

4.4.3 Description
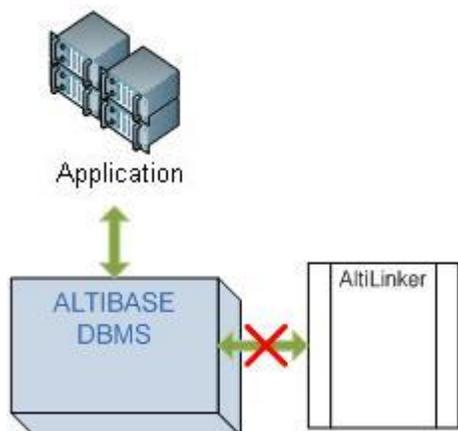
When a user connects to a server, a session is created on the server. If the user access a remote server using Database Link in this session, an additional session is created for use with Database Link and AltiLinker. Then, when the user's session is closed, the Database Link session is closed along with it.

However, after the user finishes working with Database Link, if the user session is not closed but remains open to perform other tasks, the Database Link session will unnecessarily remain open. In such cases, close the Database Link session using the ALTER SESSION CLOSE DATABASE LINK statement. That is, the ALTER SESSION CLOSE DATABASE LINK is used to close a Database Link session that is being kept alive by a user session, not the user session itself.

4.4.4 Example

```
$ isql -u sys -p manager -sysdba
------------------------------------------------
 Altibase Client Query utility.
 Copyright 2000, ALTIBASE Corporation or its subsidiaries.
 All Rights Reserved.
------------------------------------------------
ISQL_CONNECTION = UNIX, SERVER = 127.0.0.1, PORT_NO = 20300
```

```
iSQL(sysdba)> ALTER SESSION CLOSE DATABASE LINK;
Alter success.
```

4.5 SELECT

At present, Database Link supports only SELECT operations, not operations that change data, such as INSERT, UPDATE and DELETE operations. This section discusses the use of the SELECT statement.

4.5.1 FROM Clause

To use Database Link in a SELECT query, use the location descriptor in the FROM clause. The location descriptor comprises the "@" (ampersand) character and the name of the Database Link object.

Specify the name of the object (i.e. table or view) to query on the remote server before the location descriptor. If the name of the owner of the object is different from the name of the user with which the connection to the remote database was established, specify the owner name before the object name, separated by a period (".") character.

4.5.1.1 Example

<Example 1>

```
SELECT * FROM emp@link1;
```

4.5.2 WHERE Clause

The WHERE clause is used in the same way as when using a SELECT statement to query objects on the local server.

If you need to specify an object on the remote server as part of a WHERE condition, do not use the location descriptor directly in the WHERE clause. Instead, specify an alias for the object and location descriptor (e.g. "object@linkname") in the FROM clause, and then use the alias in the WHERE clause.

4.5.2.1 Example

<Example 1> Retrieve rows for which the value I1 is greater than 100 from table T1 on a remote server, which is indicated by link1.

```
SELECT * FROM T1@link1 WHERE I1 > 100;
```

<Example 2> Retrieve the names of all the employees who work in the Quality Assurance Department from table *emp1* on a remote server, indicated by *link1*, and the *employees* table on the local server.

```
SELECT e_firstname, e_lastname
 FROM (SELECT eno, e_firstname, e_lastname
  FROM employees
  UNION ALL
  SELECT eno, e_firstname, e_lastname FROM emp1@link1) v1, departments
  WHERE v1.dno = departments.dno
  AND departments.dname = 'QUALITY ASSURANCE DEPT';
```

4.5.3 Other SELECT Statement Features

Database Link supports the use of joins, subqueries, set operators and aggregate functions when executing SELECT statements. Additionally, Database Link can be used in SELECT subqueries that are part of DDL or DML statements.

4.5.3.1 Examples

<Example 1>Retrieve unique values from the *i1* column of table *T1* on a remote server referenced by *link1*.

```
SELECT DISTINCT i1 FROM T1@link1;
```

<Example 2> Retrieve the number and average age of the employees in each department by joining two tables on the remote server. Disregard any negative department IDs.

```
SELECT t1.dept_id, COUNT(*), AVG(age)
FROM t_member@link1 t1, t_dept@link1 t2
WHERE t1.dept_id=t2.dept_id
GROUP BY t1.dept_id
HAVING t1.dept_id>=0;
```

<Example 3> Join tables *t_member* and *t_dept* on a remote server referred to as *link1* to retrieve the name and age of the three employees under age 30 with the highest IDs, along with the total age of all employees.

```
SELECT t1.name, t1.age,
 (SELECT SUM(age) FROM t_member@link1) sum
FROM t_member@link1 t1,
 (SELECT dept_name, dept_id
 FROM t_dept@link1) t2
WHERE t1.dept_id=t2.dept_id
 AND t1.age<30
ORDER BY t1.member_id DESC LIMIT 3;
```

<Example 4> Retrieve the names and ages from table t2 on a remote server referred to as link1 and insert them into table t1 on the local server.

```
INSERT INTO t1 SELECT name, age FROM t2@link1;
```

4.6 The EXEC_REMOTE Hint

Use the EXEC_REMOTE hint to pass a query or subquery to a remote server so that the query can be processed and executed directly on the remote server. Setting the AUTO_REMOTE_EXEC property (in the altibase.properties file) to 1 achieves the same result.

4.6.1 Restrictions

- All of the objects referenced in the query must reside on the remote server.

- The query can only reference a single Database Link object.

- The query must not contain any references to stored procedures or sequences.

- The query must not contain any references to host variables.

- In the case of a subquery, the subquery must not reference any columns, objects or aliases defined outside of the subquery.

If any of the above conditions is false, the EXEC_REMOTE hint will be ignored.

When a SELECT query contains a subquery, the EXEC_REMOTE hint can be used in either the main query or the subquery. If the EXEC_REMOTE hint is located in the main query, it will apply to the entire query, including the subquery. If the EXEC_REMOTE hint is located in the subquery, it will apply only to the subquery. This means that there is no need to use the EXEC_REMOTE hint in the subquery when using it in the main query. It also means that it is impossible to execute a subquery on the local server when the main query is to be processed on the remote server.

4.6.2 Examples

<Example 1> Retrieve names from a table called "employees" on a remote server referred to as *link1*. All query processing, including the removal of duplicate values, will be performed on the remote server.

```
SELECT /*+EXEC_REMOTE*/ DISTINCT name
FROM employees@link1;
```

<Example 2>Return everything from table *T1* on a remote server referred to as link1, but only if the sum of the values in column *I2* in table *T2* on the server is greater than 5 for all rows in which *I1* equals 3. In this case, the main query and the subquery are processed together on the remote server.

```
SELECT /*+EXEC_REMOTE*/ * FROM T1@link1
WHERE 5 < ( SELECT /*+EXEC_REMOTE*/ SUM(I2)
FROM T2@link1 WHERE I1=3 );
```

<Example 3> The query is the same as in <Example 2>, but the subquery is first passed to the remote server and executed. The condition is then evaluated on the local server, and is used to determine whether to retrieve data from table *T1* on the remote server.

```
SELECT * FROM T1@link1
WHERE 5 < ( SELECT /*+EXEC_REMOTE*/ SUM(I2)
FROM T2@link1 WHERE I1=3 );
```

Appendix A. Property and Data Dictionary

Properties related to Database Link

In order to use Database Link, it will be necessary to set some of the properties in the altibase.properties file appropriately. The properties that pertain to Database Link are listed below. For more details, please refer to the *General Reference*.

- AUTO_REMOTE_EXEC
- DBLINK_ENABLE
- LINKER_CONNECT_TIMEOUT
- LINKER_LINK_TYPE
- LINKER_PORT_NO
- LINKER_RECEIVE_TIMEOUT
- LINKER_THREAD_COUNT
- LINKER_THREAD_SLEEP_TIME
- MAX_DBLINK_COUNT
- REMOTE_SERVER_CONNECT_TIMEOUT

Data Dictionary related to Database Link

You can check the current state of Database Link using the meta table and performance views listed below.

For more detailed information about these and other meta tables and performance views, please refer to the *General Reference*.

Meta Table

- SYS_DATABASE_LINKS_

Data Dictionary related to Database Link

Performance View

- V$DBLINK_REMOTE_STATEMENT_INFO

- V$DBLINK_REMOTE_TRANSACTION_INFO

- V$DBLINK_TRANSACTION_INFO

- V$LINKER_STATUS

Index