Configuring JDBC

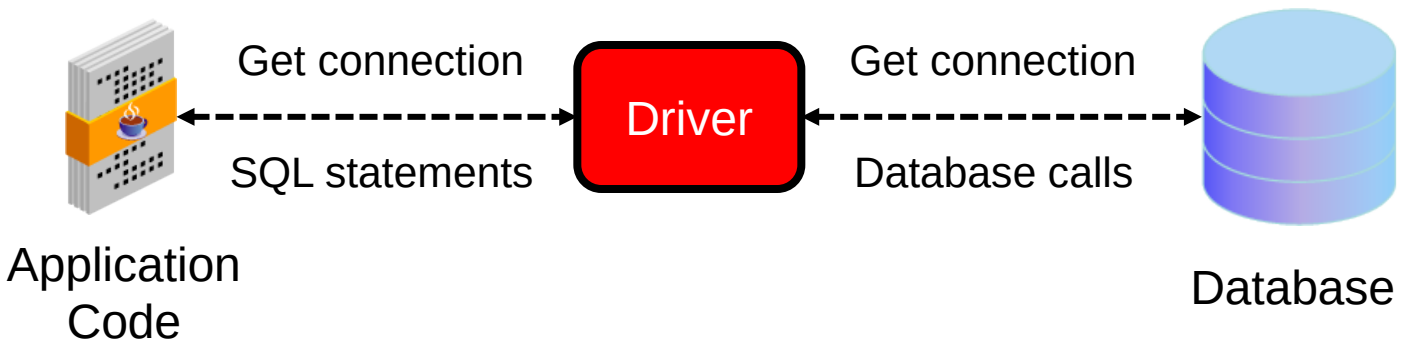
Objectives

After completing this lesson, you should be able to configure:

- A generic data source
- A GridLink data source

JDBC: Overview

- The Java Database Connectivity (JDBC) API:
  - Is a platform and vendor-independent mechanism for accessing and using a database
  - Provides transparency from proprietary vendor issues
  - Requires the use of a *driver* (a Java class)
- JDBC drivers are supplied with the WebLogic Server installation or by your database vendor.

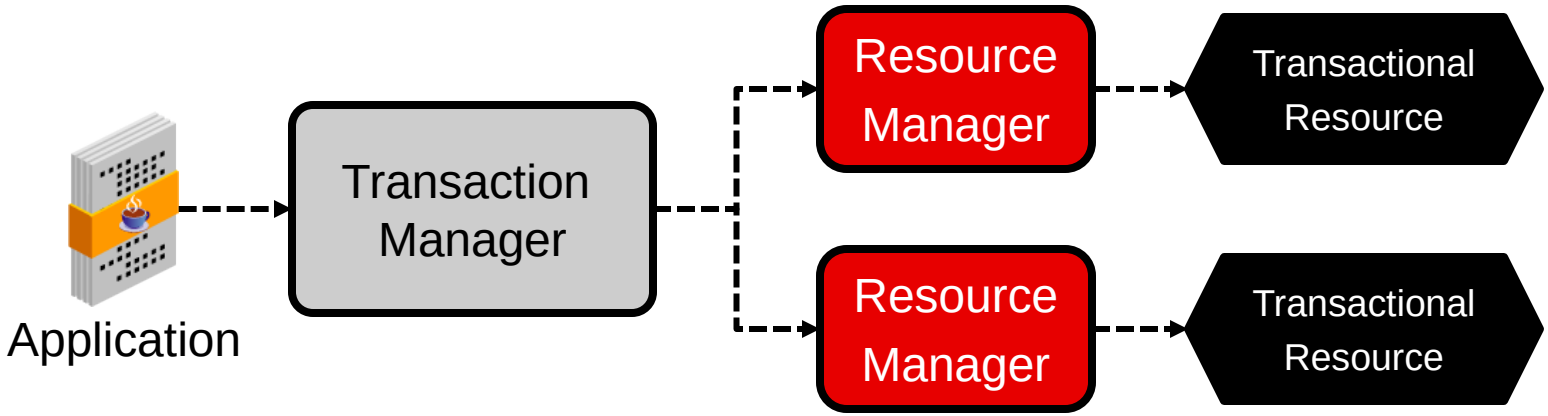


WebLogic JDBC Drivers

- Oracle and third-party drivers for many databases are included in the WebLogic Server installation:
  - Oracle 11*g* and 12*c*
  - Sybase
  - Microsoft SQL Server
  - IBM DB2
  - Informix
  - MySQL
- By default, these drivers are added to the server's CLASSPATH. To use other drivers, you must update the server's CLASSPATH.
- XA (Extended Architecture) drivers provide support for global transactions.

Global Transactions: Overview

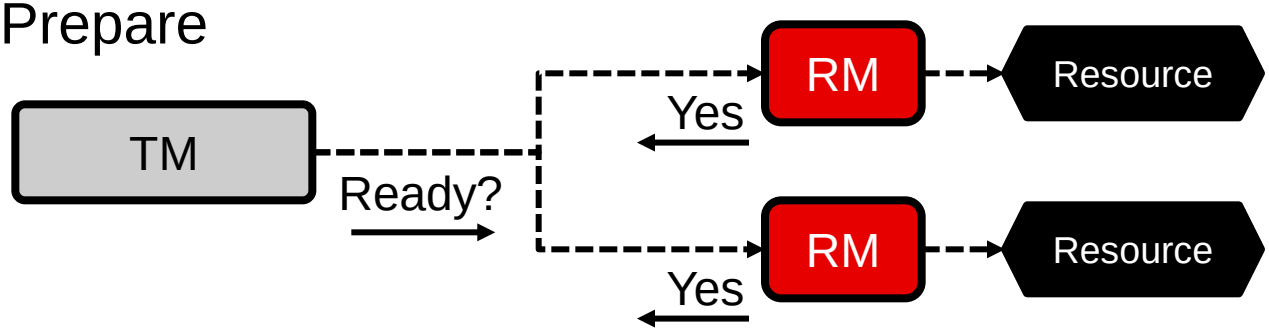
- A global (distributed) transaction involves more than one transactional resource.
  - A transaction manager (TM) deals with each resource manager (RM). WebLogic Server can act as a TM.



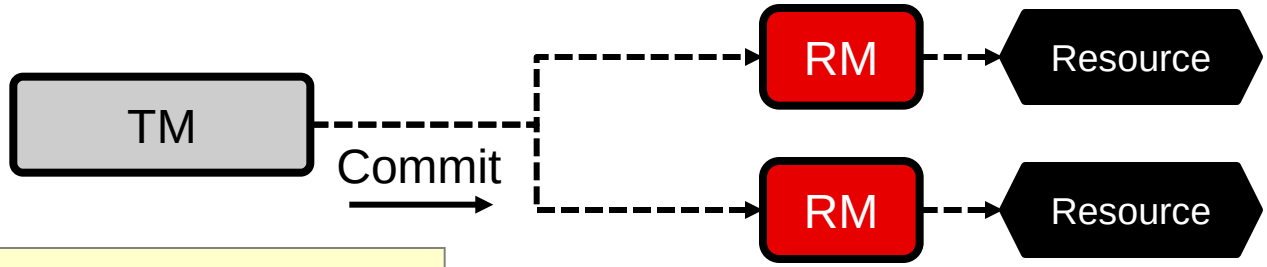
Two-Phase Commit

The Two-Phase Commit (2PC) protocol uses two steps to commit changes within a global transaction:

- Phase I: Prepare



- Phase II: If all resources are ready, the TM tells them to commit. Otherwise, the TM tells them to roll back.



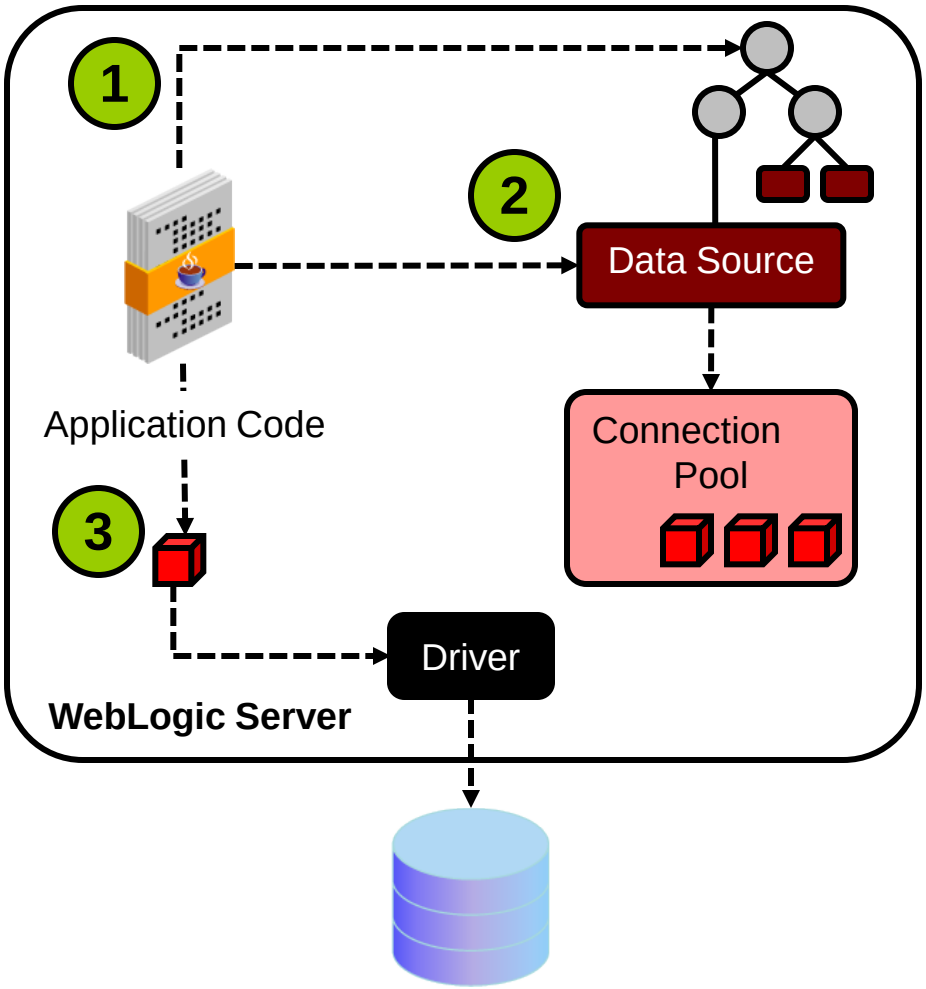
Extended Architecture (XA) implements 2PC.

JDBC Data Source

- A data source is a Java object targeted to and managed by one or more instances of WebLogic Server.
- A deployed data source has connections to a particular database in its connection pool ready-to-go for applications running on those servers.
  - The connection pool configuration determines which database is used and how many connections are in the pool.
  - Applications locate a data source in a server's tree of resources by using the Java Naming and Directory Interface (JNDI) API.
    - After the application has a reference to the data source, it can request a database connection from the data source.

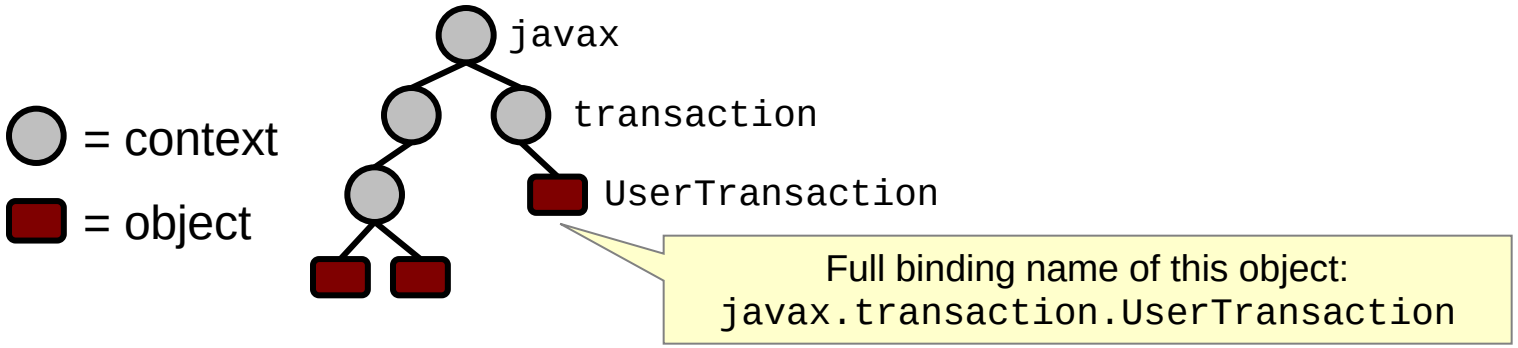
JDBC Data Source

1. An application looks up the data source in a server's resource tree by using the JNDI API.
2. It asks the data source for a connection.
3. It uses the connection (which uses a driver) to access the database.
4. When finished, it closes the connection (which returns it to the pool).



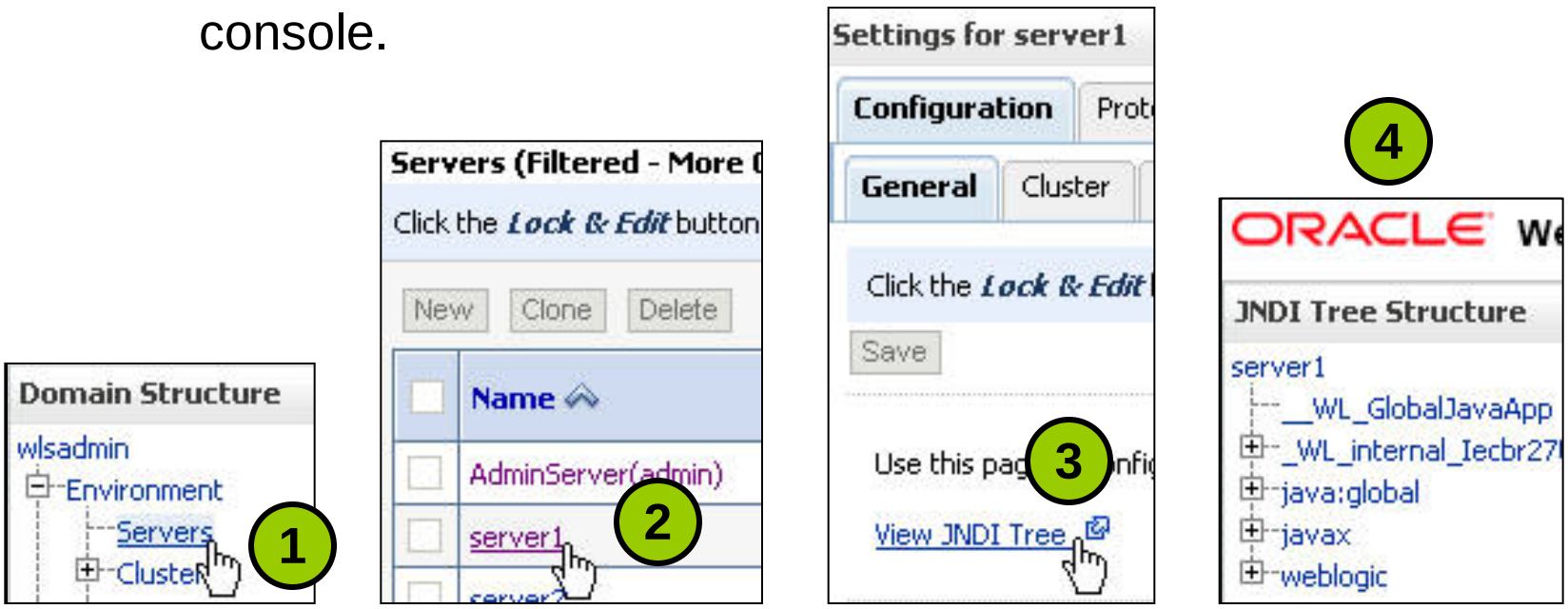
Java Naming and Directory Interface (JNDI)

- An API for accessing directory or naming services.
- WebLogic Server keeps a tree of its resources in memory that can be accessed by using JNDI.
- JNDI terms:
  - *Context*: A “container” node in the JNDI tree that can contain other contexts and objects
  - *Object*: A leaf node in the JNDI tree. Resources are objects.
  - *Binding*: Associating an object with a name and a context



JNDI Duties of an Administrator

- Report to developers the JNDI names of resources you create, so they can look them up in their code.
- Check whether resources are in the JNDI tree of an instance of WebLogic Server.
  - You can view the JNDI tree of a server by using the admin console.



Deployment of a Data Source

- The configuration of a WebLogic Server data source is stored in an XML document called a *JDBC module*:
  - A WLS-specific extension of Java EE modules
  - Targeted to certain instances of WebLogic Server
- A JDBC module is either a *system module* or an *application module*.
  - A JDBC system module is created by using the administration console or WLST. It resides in the domain configuration here:

    domainname`/config/jdbc/`*dsname*`-jdbc.xml`

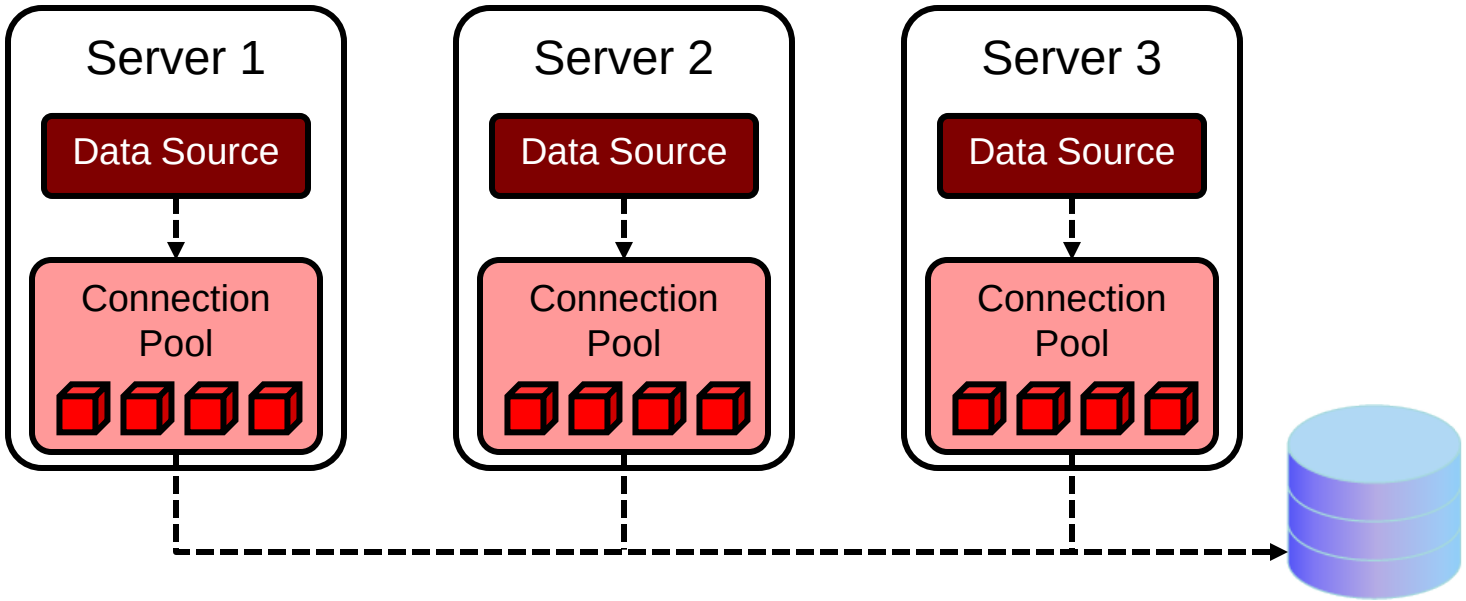
    - *domainname*: The main domain directory
    - *dsname*: The name of the data source

  - A JDBC application module can be deployed either within a Java EE application or stand-alone.

Targeting of a Data Source

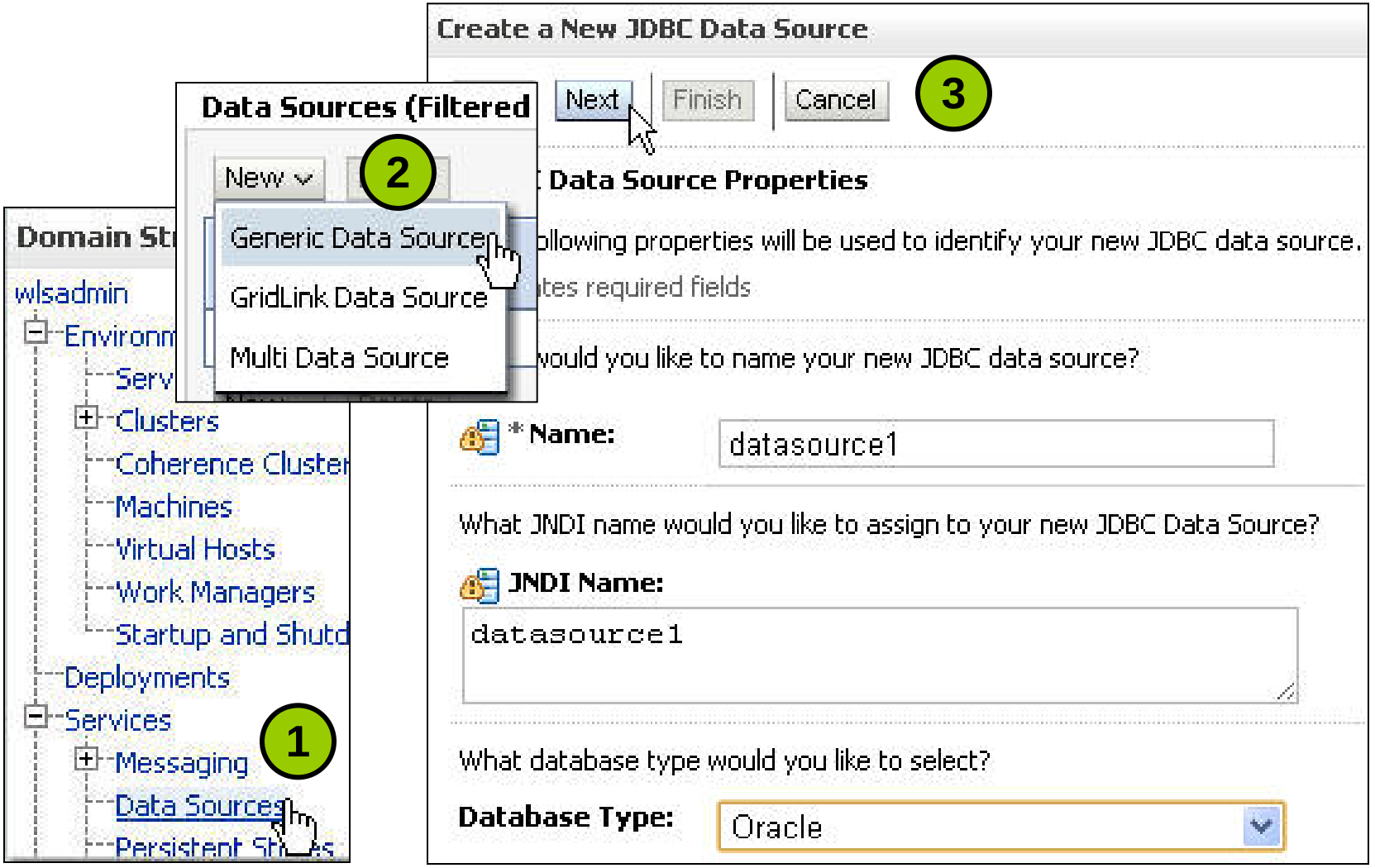
- A data source is targeted to one or more instances of WebLogic Server.
  - Each targeted server has its own instance of that data source.
  - Each data source has its own connection pool.



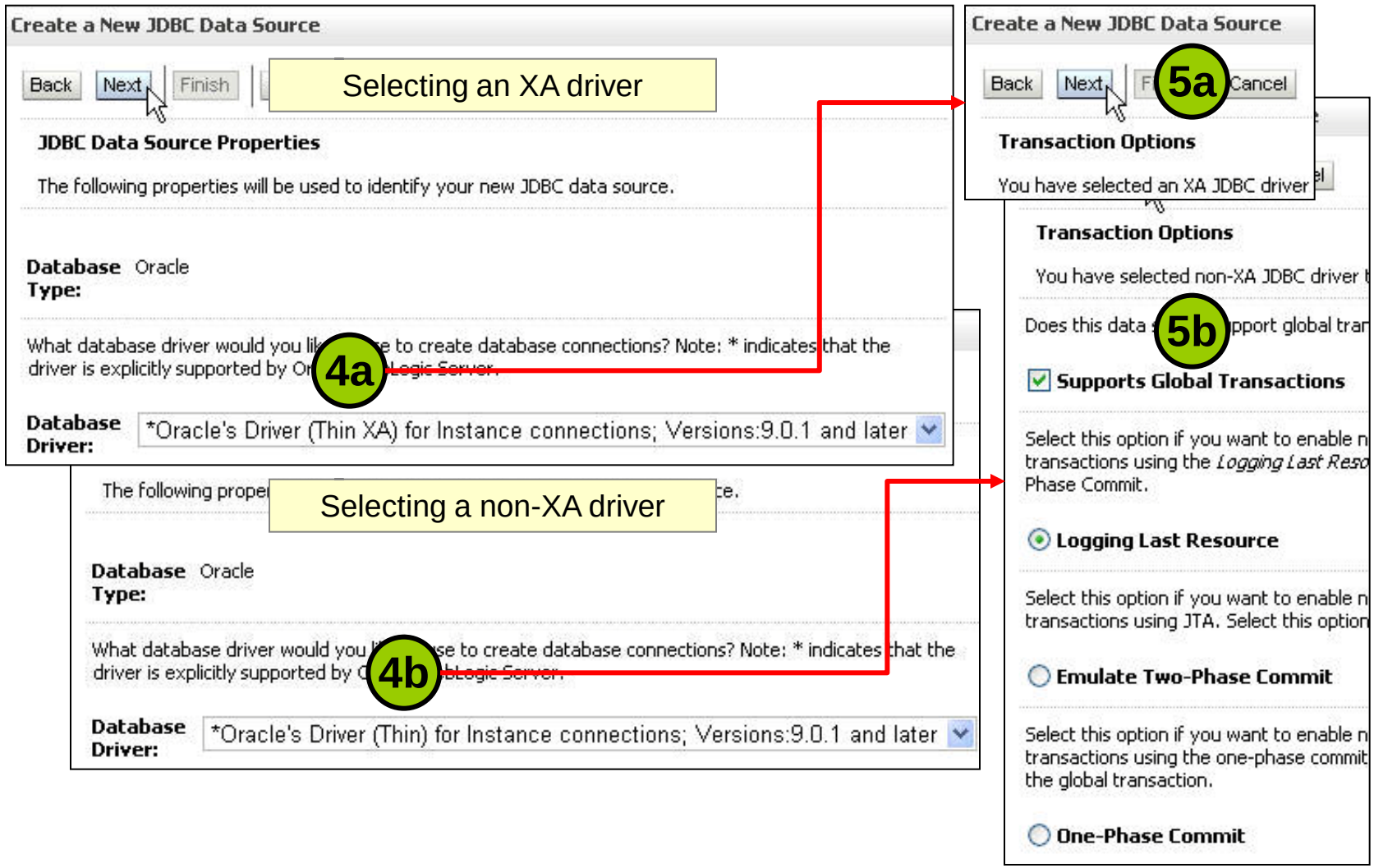
Types of Data Sources

- Generic data source
  - Is a standard data source with a connection pool tied to a particular database
- Multi data source
  - Is a collection of generic data sources tied to multiple database servers
  - Is looked up and used by applications like a generic data source
  - Transparently provides load balancing or failover across its member generic data sources
- GridLink data source
  - Provides connectivity between WebLogic Server and the Oracle Real Application Clusters (RAC) Database

Creating a Generic Data Source



Creating a Generic Data Source

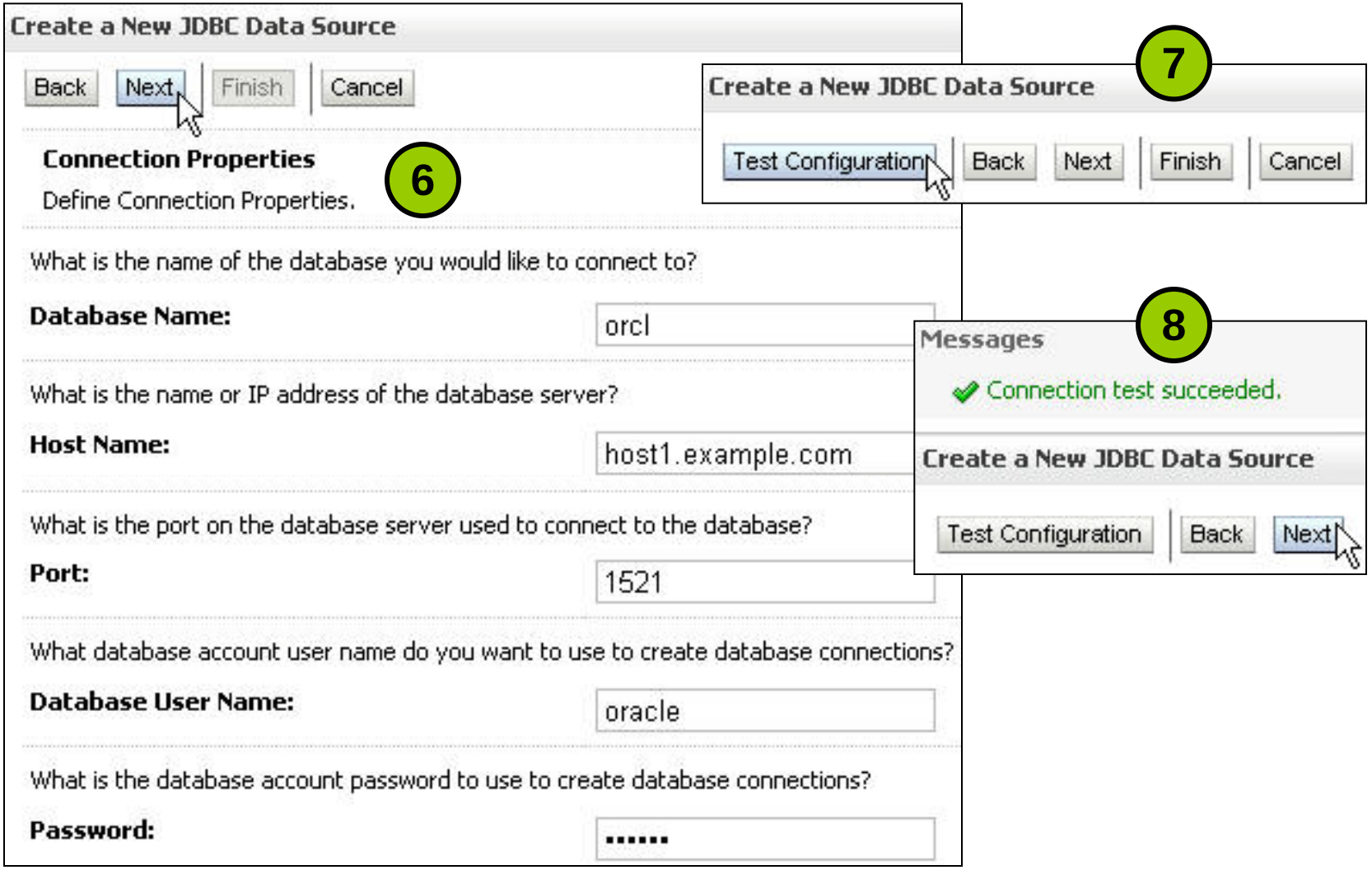


Non-XA Driver Transaction Options

If a non-XA driver is selected, you can still choose for the data source to support global transactions. If you enable this option, you must specify *how* the data source will participate in those transactions:

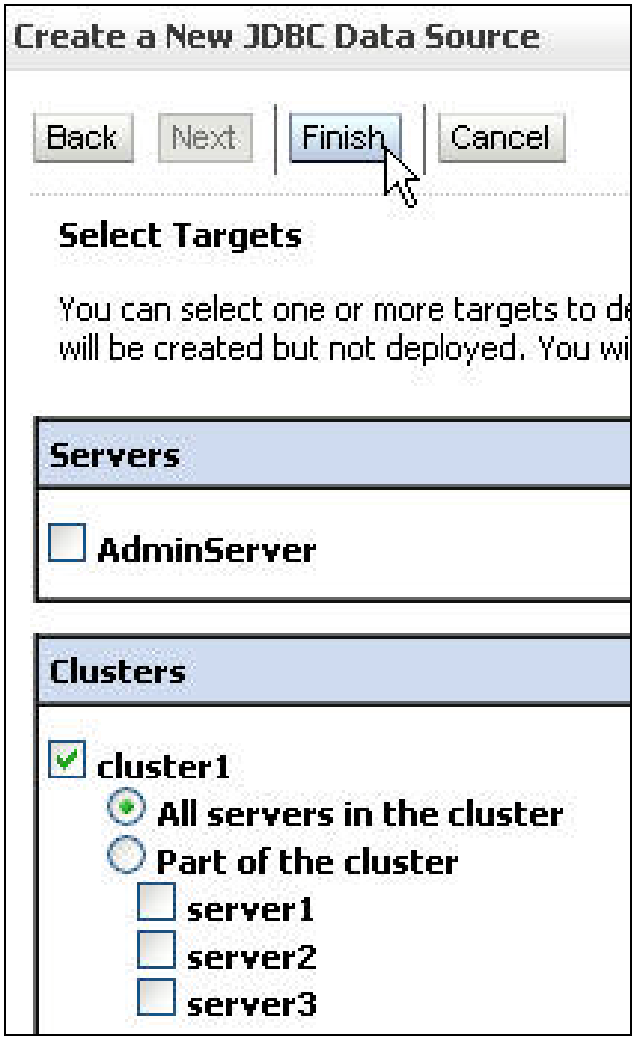
- Logging Last Resource: This resource is processed last. If it succeeds, the other resources are told to commit; if it fails, they are told to roll back.
- Emulate Two-Phase Commit: This resource always returns “ready” during phase one of 2PC. This can possibly cause heuristic conditions.
- One-Phase Commit: Only this resource can participate in the global transaction.

Creating a Generic Data Source

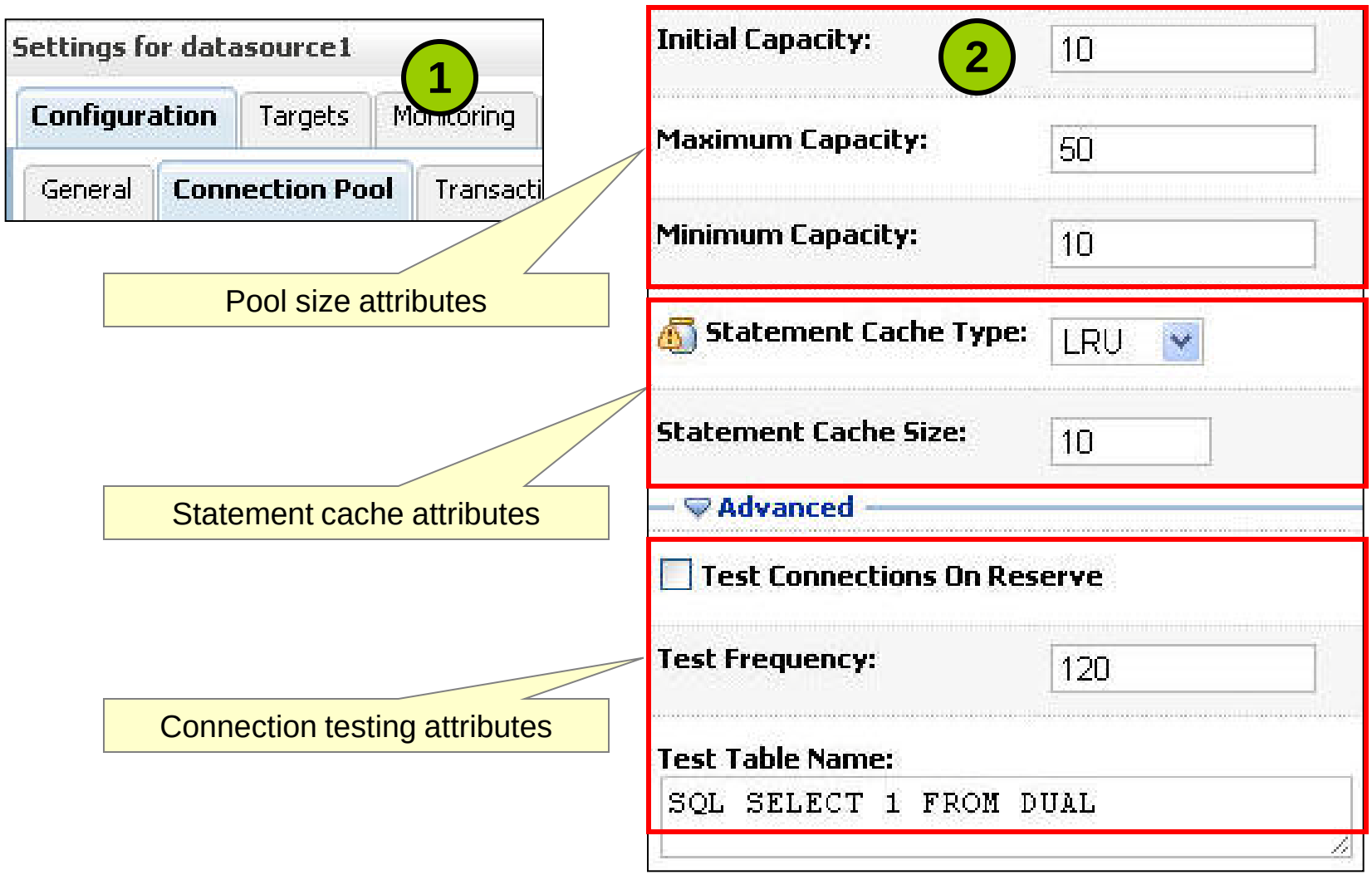


Creating a Generic Data Source

9



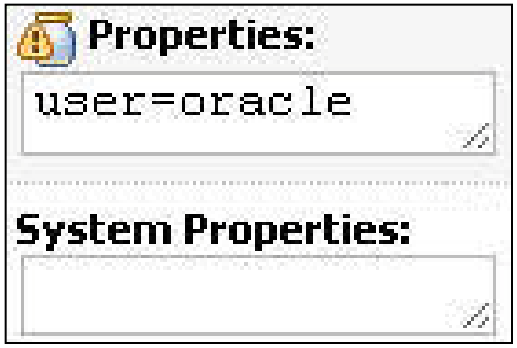
Connection Pool Configuration



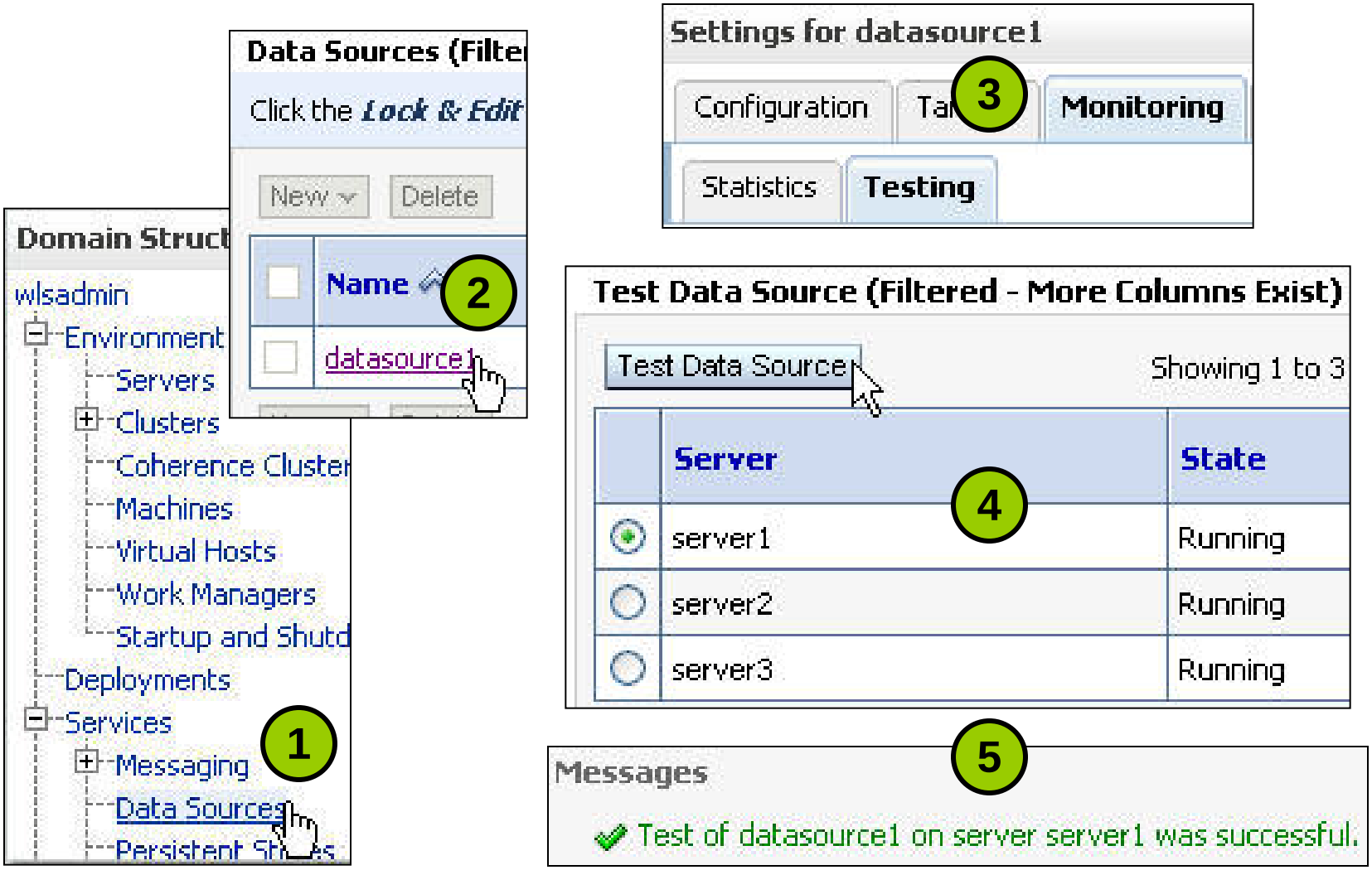
Connection Properties

Connection properties are:

- Found in the configuration of the data source under **Configuration** > **Connection Pool**
- Property/value pairs entered in:
  - **Properties** as: *property=value*
  - **System Properties** as: *property=systemProperty*
- Used to configure JDBC connections
- Passed to the driver during connection setup

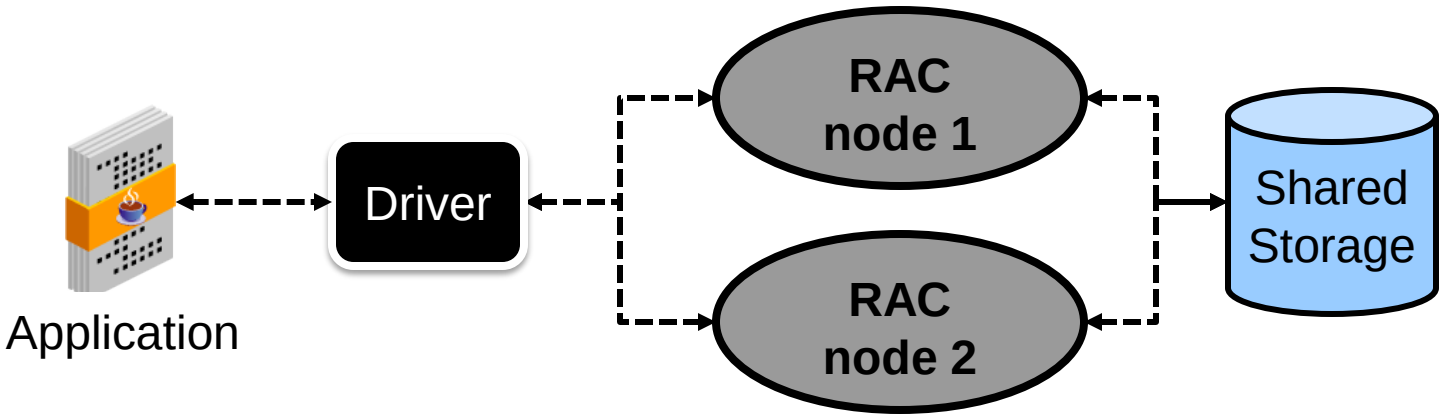


Testing a Generic Data Source



Oracle Real Application Clusters: Overview

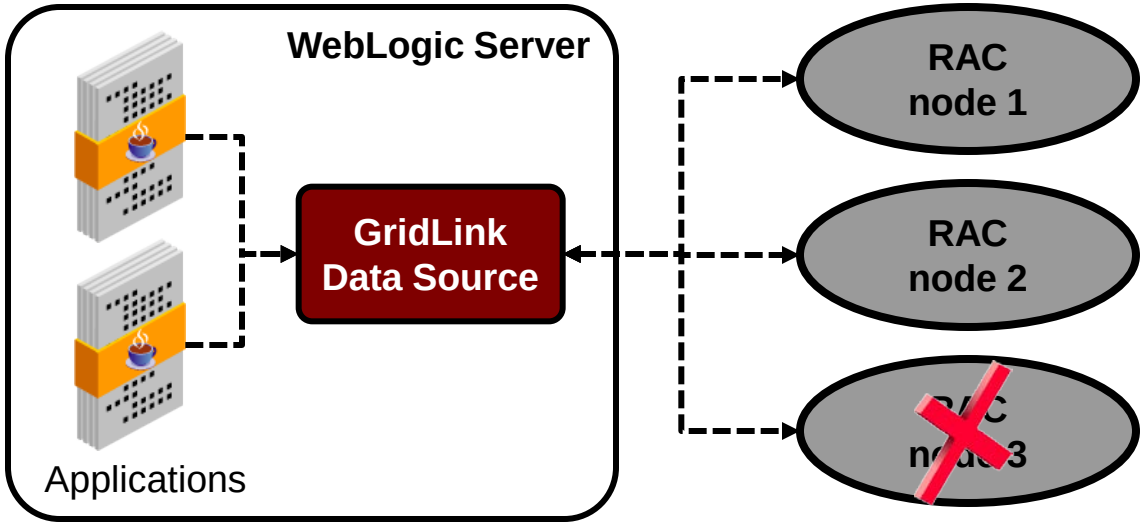
- Oracle Real Application Clusters (RAC):
  - Supports multiple Oracle database servers for greater scalability and reliability
  - Relies on database servers having access to a shared and highly available storage device



GridLink Data Source for RAC

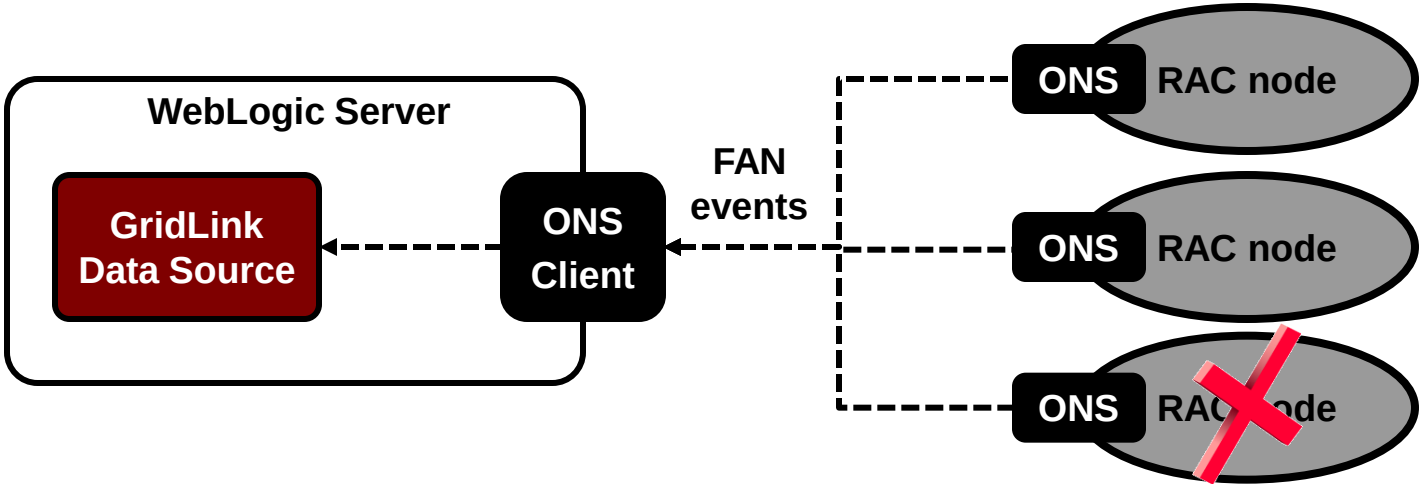
WebLogic Server's *GridLink* data source is "RAC-aware." It:

- Performs intelligent load balancing based on the current RAC workload
- Implements RAC's Fast Connection Failover (FCF) pattern
- Ensures that all database operations within a global transaction are routed to the same RAC node ("XA affinity")



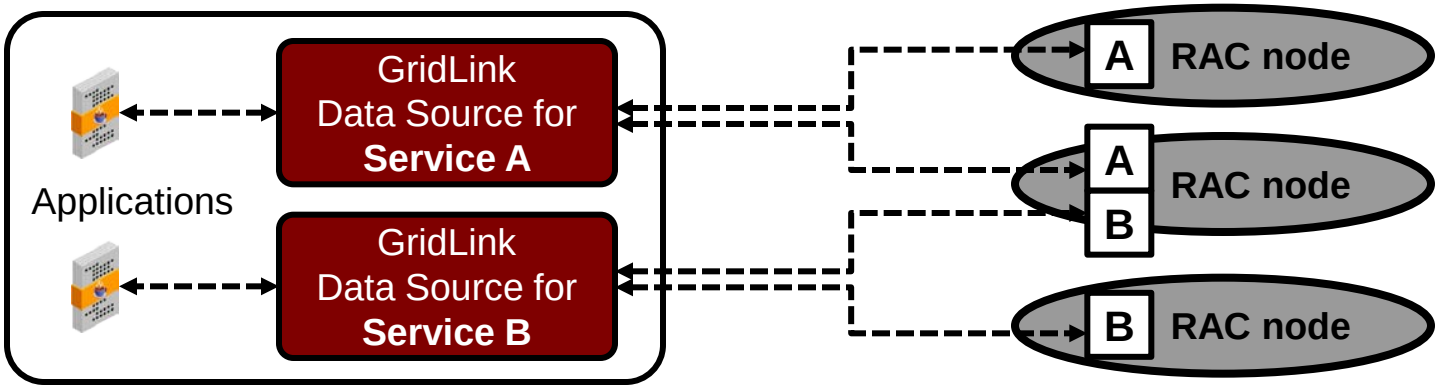
GridLink, FCF, and ONS

- Typically, when a system goes down, applications must wait for the network to time out (perhaps minutes).
- FCF pattern: The Oracle Notification Service (ONS) delivers Fast Application Notification (FAN) events about RAC availability and workload to registered subscribers.
- The GridLink data source can subscribe to ONS and immediately respond to nodes that are added or removed.



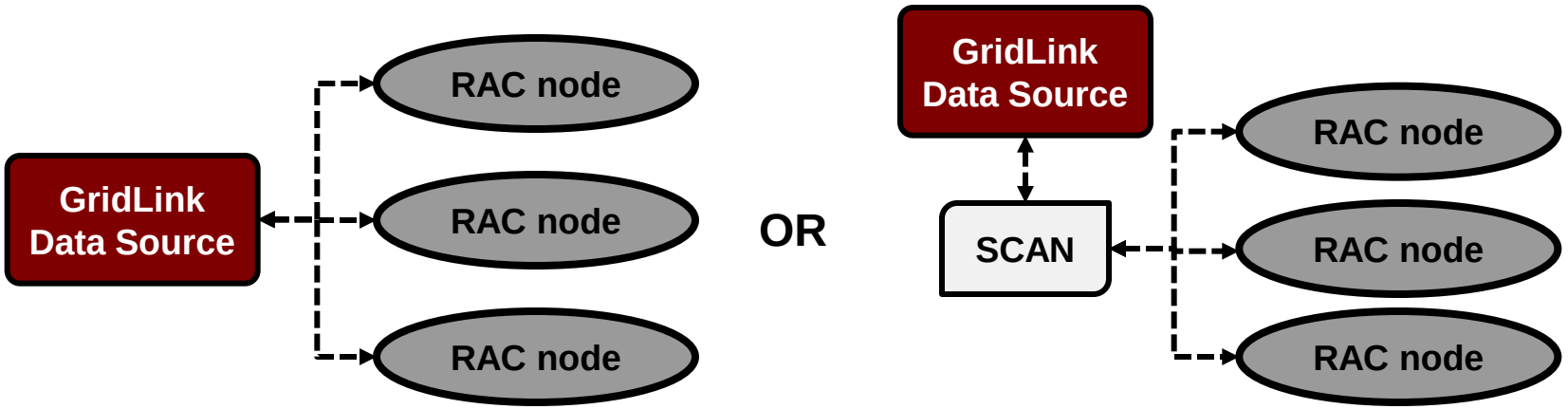
GridLink and Services

- Oracle Database supports services that:
  - Act as gateways to a subset of RAC nodes
  - Automatically start on another node if the current one fails
  - Are accessed by applications by using the service name
  - Enable you to control and prioritize the available capacity on your grid for different clients
- FCF requires the use of database services.
- Create a separate GridLink data source for each service.

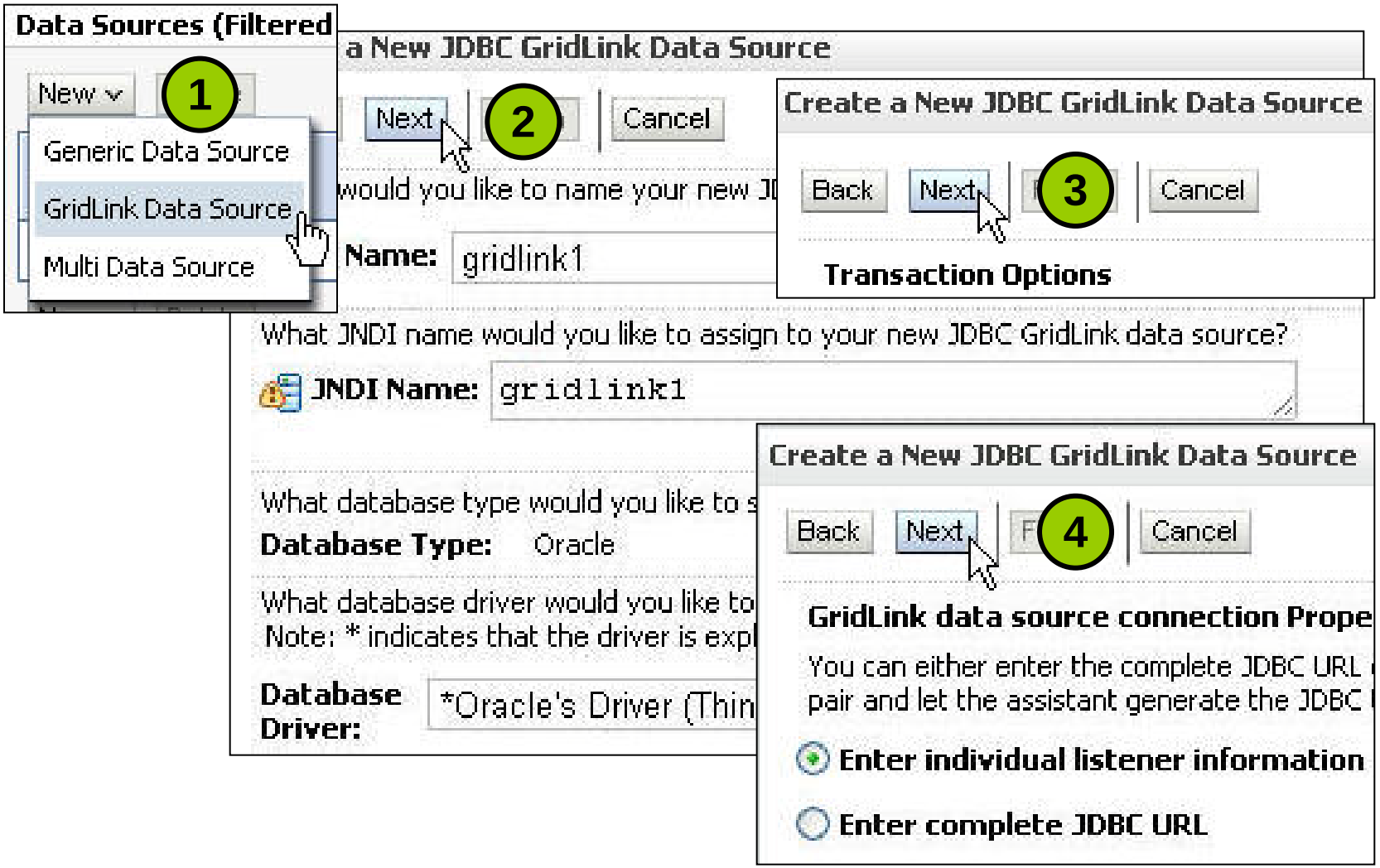


GridLink and Single Client Access Name (SCAN)

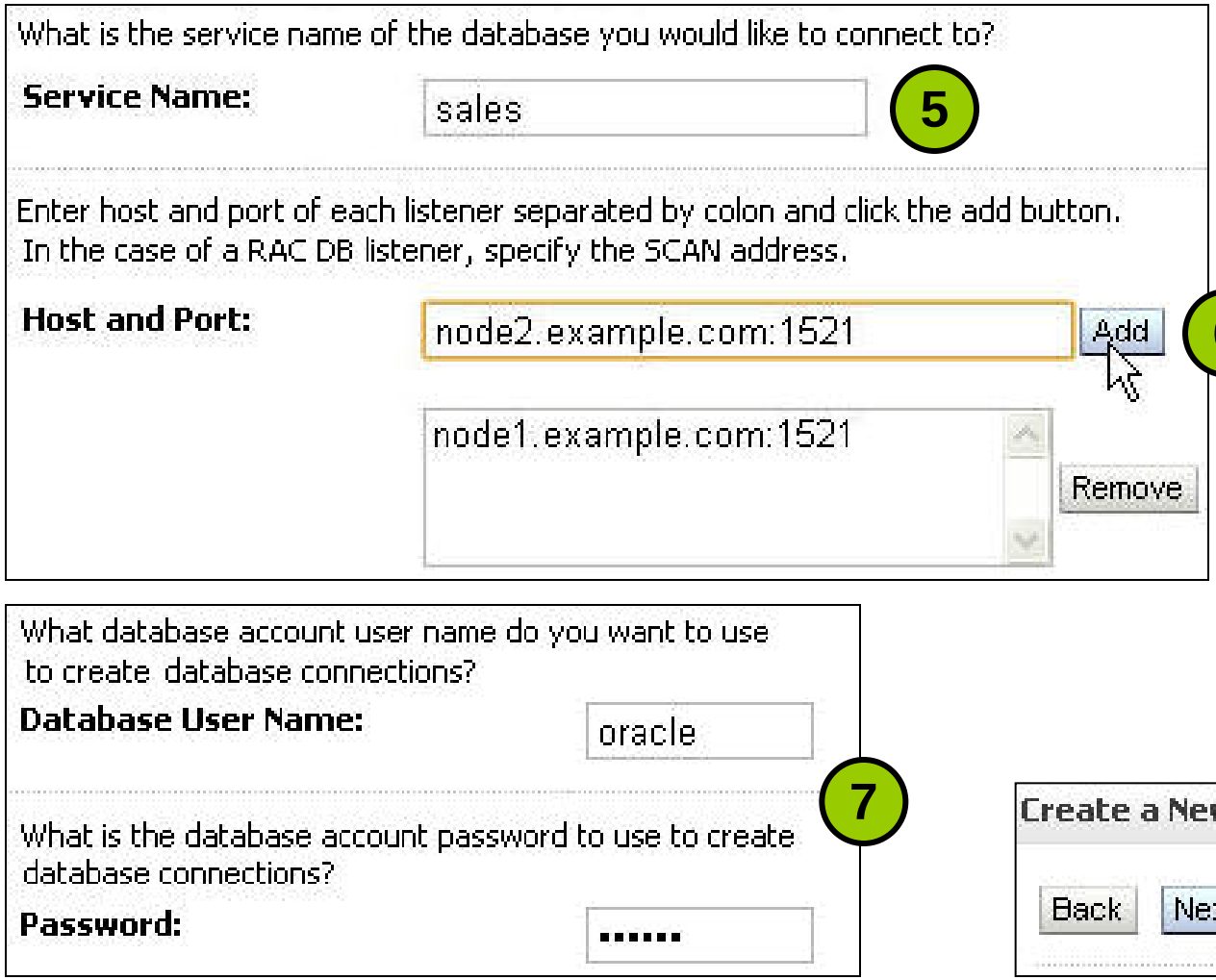
- Starting with Oracle RAC version 11*g*R2, a SCAN service is provided that:
  - Accepts a database cluster alias
  - Returns the locations of cluster members
  - Can run independently or integrate with your corporate DNS
- GridLink data sources can either use a list of database node locations or a single SCAN address.



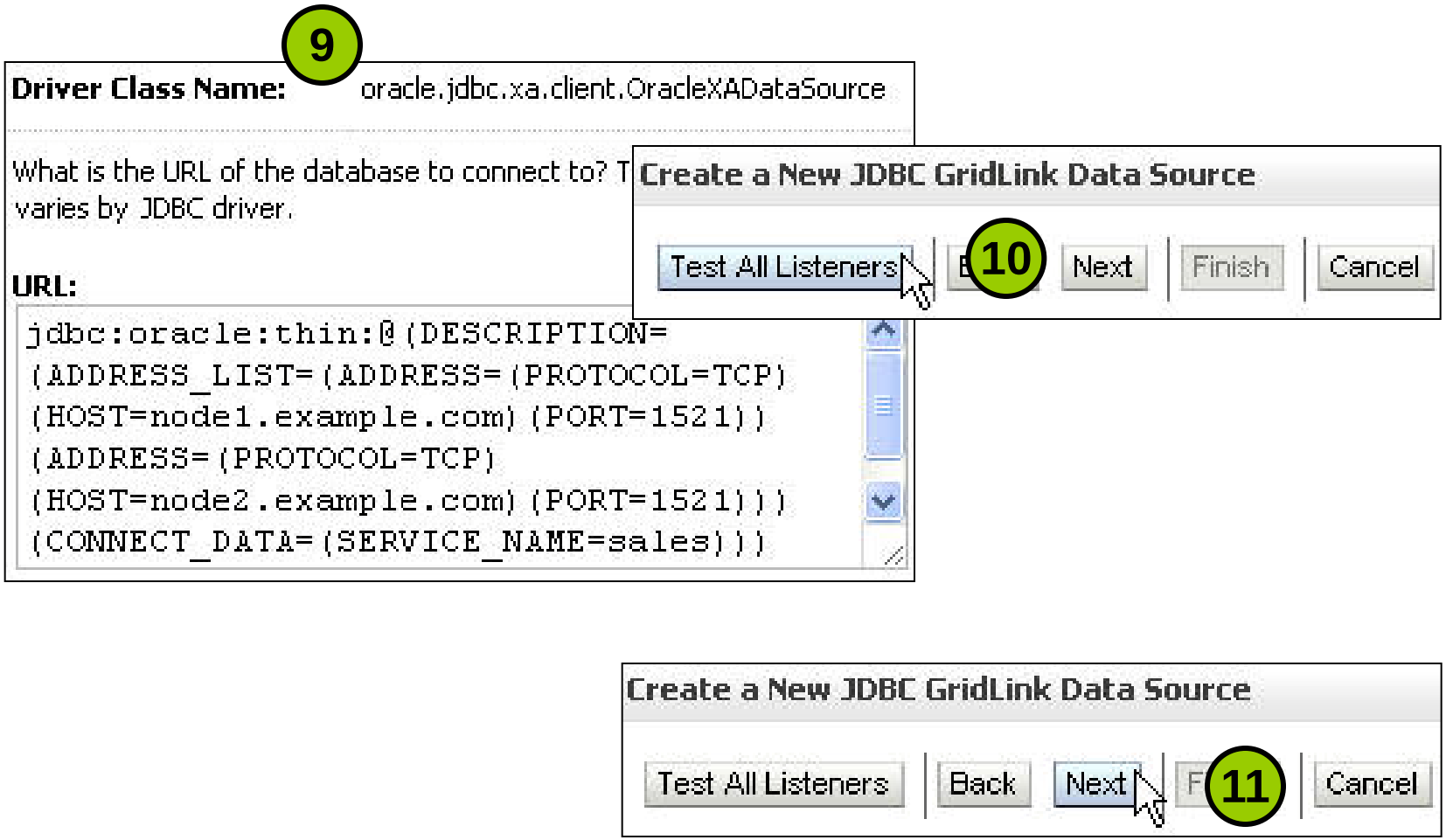
Creating a GridLink Data Source



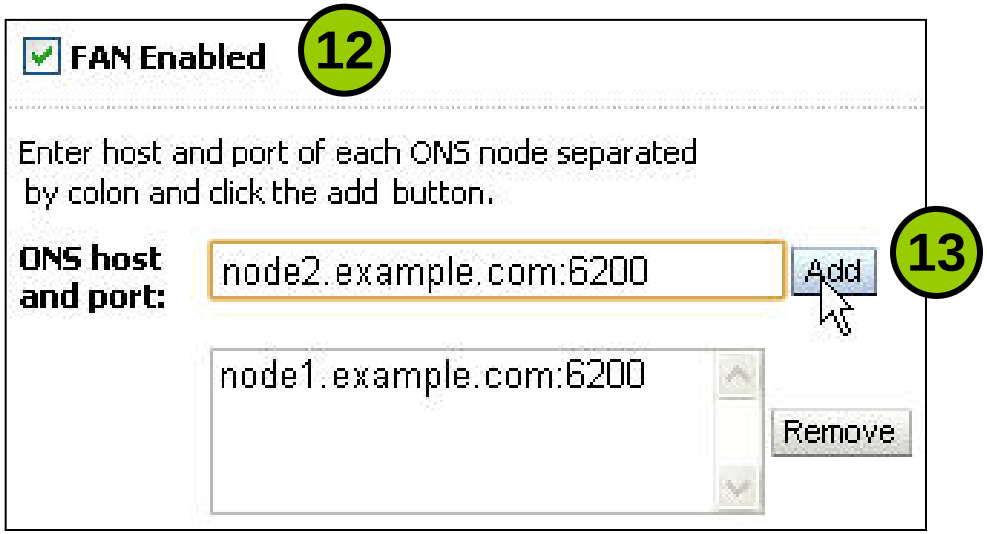
Creating a GridLink Data Source



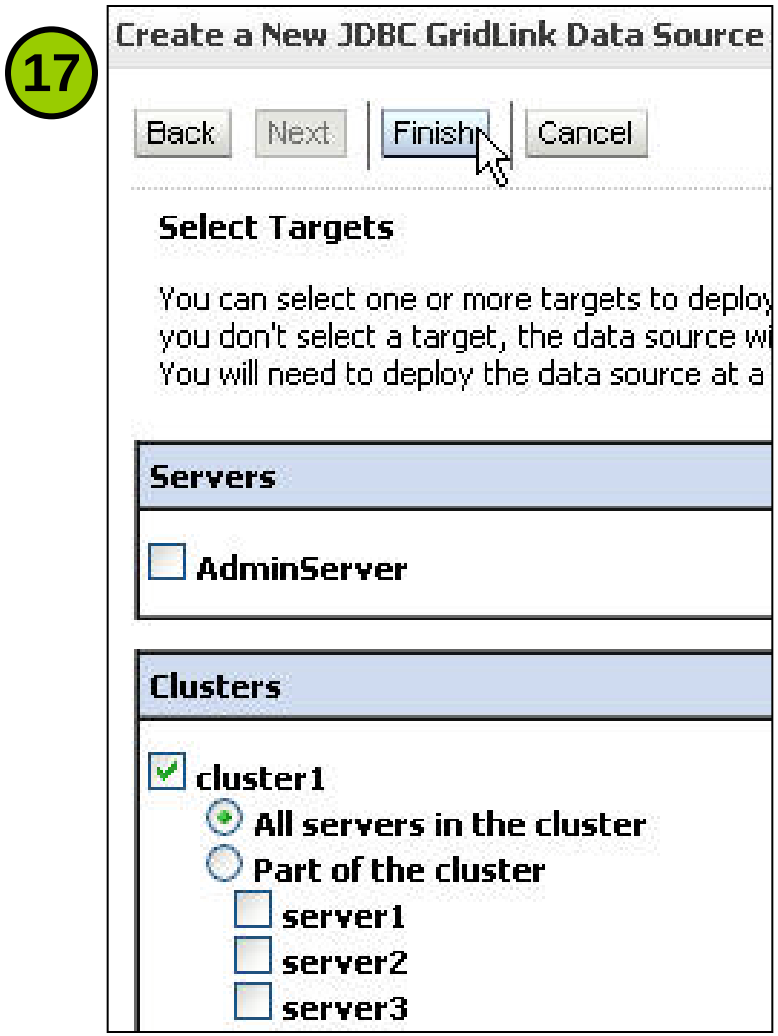
Creating a GridLink Data Source



Creating a GridLink Data Source



Creating a GridLink Data Source

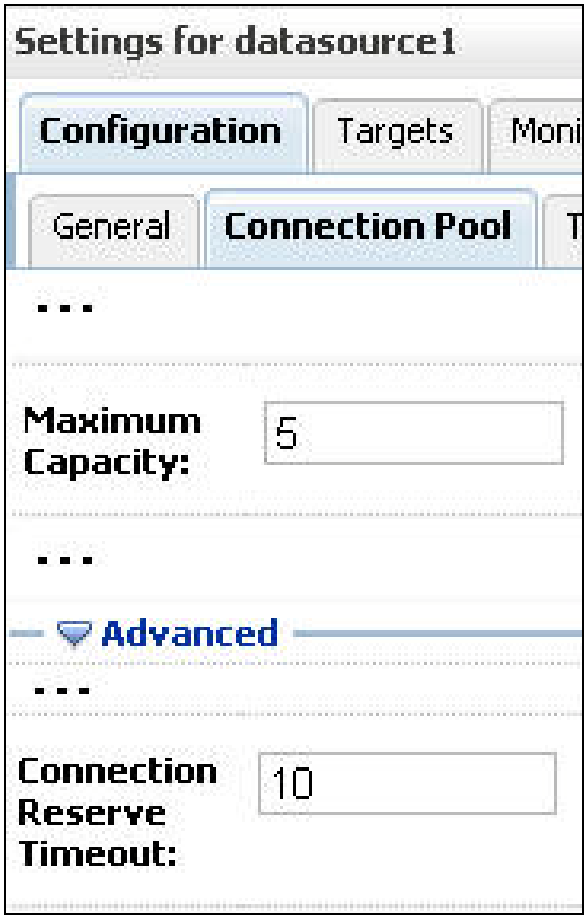


Common Data Source Problems

- Data source configuration errors:
  - Invalid JDBC URL
    - When the data source is created or deployed it will fail. The underlying exception is `java.net.ConnectException`.
  - Invalid credentials
    - When the data source is created or deployed, it will also fail. Not because a network connection could not be established, but because the database rejects the connection attempt.
  - Wrong driver version in the `CLASSPATH`
    - Errors can be subtle and hard to catch if the version of the driver is not the one expected. Ensure that the drivers you want to use are in the `CLASSPATH` before the standard ones. For example, add this line in `startWebLogic.sh` after the `CLASSPATH` has been set:

```
CLASSPATH="/path/customdriver.jar:${CLASSPATH}"
```

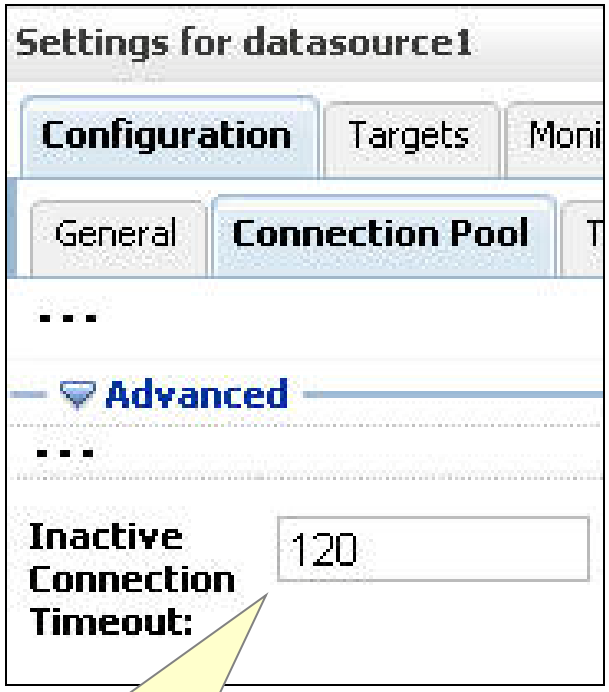
Common Data Source Problems



- Insufficient connections
  - When the maximum capacity is reached, the next application to request a connection will wait Connection Reserve Timeout seconds for a free one.
    - If none are free at that time, an exception is returned.
  - If this happens often, see whether more connections from the database are available, and increase the Maximum Capacity.
  - If the applications can wait longer, perhaps increase the Connection Reserve Timeout seconds.

Common Data Source Problems

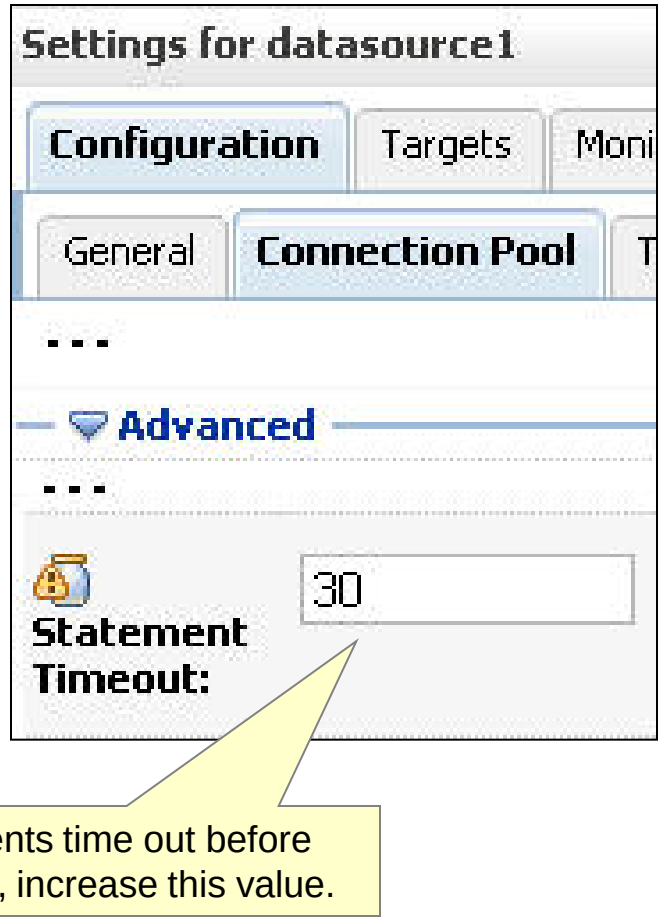
- Connection leaks
  - Poorly implemented applications can starve the data source of connections.
    - Hold a connection too long, even when not currently using it
    - Forget to close a connection (so it can return to the pool)
  - To provide a failsafe, WebLogic Server can automatically reclaim a connection after a certain amount of inactivity.



Reclaim a reserved connection after two minutes of inactivity.

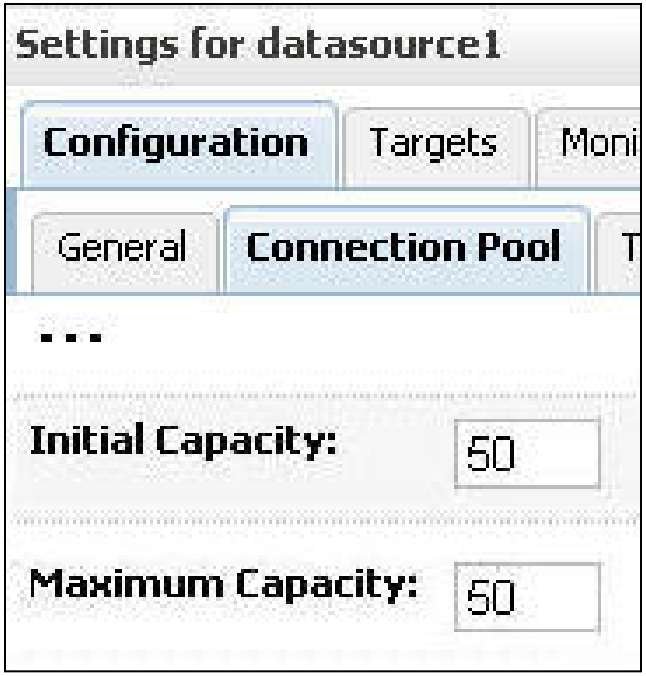
Common Data Source Problems

- Statement timeout
  - Most JDBC drivers support a maximum time limit for SQL statements that can be set:
    - By using the Statement Timeout attribute
    - Programmatically by the application
  - Increase this value if your applications require complex, long-running database operations.



Basic Connection Pool Tuning

- Pool Capacity
  - Connection creation is expensive.
  - For applications with consistent, heavy database use:
    - Determine Maximum Capacity experimentally
    - Set Initial Capacity equal to Maximum Capacity
  - For applications with intermittent peak database use:
    - Use different values for Initial and Maximum Capacity
    - Tune the Shrink Frequency (seconds) based on the speed of the load changes

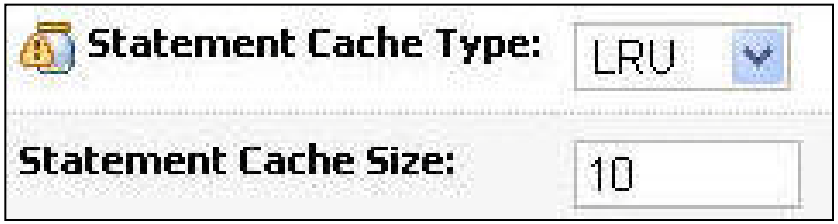


Basic Connection Pool Tuning

- Connection testing helps to ensure that connections are viable, but can degrade performance.
  - WebLogic Server tests a connection before giving it to an application when you enable Test Connections On Reserve.
  - WebLogic Server tests connections periodically based on the Test Frequency value.
    - If you use Test Frequency, do not test too often.
  - To help minimize the performance impact of testing, use Seconds to Trust an Idle Pool Connection
    - It is the number of seconds after a connection has been proven viable that WebLogic Server trusts it and skips testing it again.

Basic Connection Pool Tuning

- Statement caching
  - Prepared and callable statements can be cached to improve overall performance through reuse.
  - The Statement Cache Type determines the algorithm.
    - LRU: Once the cache is full, the least recently prepared/callable statement is replaced by a new one.
    - FIXED: Once the cache is full, *no* new prepared/callable statement is cached.
  - Determine the size of the cache through experimentation
    - Warning: Some databases maintain an open cursor for each open statement, so if the cache is too large, you could exceed the open cursor limit.



Quiz

Developers look up a server's data source by using which API?

a. JDBC
b. EJB
c. SQL
d. JNDI

Quiz

The connection pool of a generic data source has its Initial Capacity set to 2 and its Maximum Capacity set to 10. The data source is targeted to three managed servers. What is the most number of database connections that this can cause to be used at one time?

a. 10
b. 12
c. 30
d. 36

Summary

In this lesson, you should have learned how to configure:

- A generic data source
- A GridLink data source

Practice 7-1 Overview: Configuring a JDBC Data Source

This practice covers creating and configuring a generic data source.