

**Name** create-jdbc-connection-pool – registers a JDBC connection pool

**Synopsis** create-jdbc-connection-pool [--help]  
 [--datasourceclassname=*datasourceclassname*]  
 [--restype=*resourcetype*]  
 [--steadypoolsize=*poolsize*]  
 [--maxpoolsize=*maxpoolsize*]  
 [--maxwait=*maxwaittime*]  
 [--poolresize=*poolresizelimit*]  
 [--idletimeout=*idletimeout*]  
 [--initsql=*initsqlstring*]  
 [--isolationlevel=*isolationlevel*]  
 [--isolationguaranteed={true|false}]  
 [--isconnectvalidatereq={false|true}]  
 [--validationmethod=*validationmethod*]  
 [--validationtable=*validationtable*]  
 [--failconnection={false|true}]  
 [--allownoncomponentcallers={false|true}]  
 [--nontransactionalconnections={false|true}]  
 [--validateatmostonceperiod=*validationinterval*]  
 [--leaktimeout=*leaktimeout*]  
 [--leakreclaim={false|true}]  
 [--statementleaktimeout=*statementleaktimeout*]  
 [--statementleakreclaim={false|true}]  
 [--creationretryattempts=*creationretryattempts*]  
 [--creationretryinterval=*creationretryinterval*]  
 [--sqltracelisteners=*sqltracelisteners*[, *sqltracelisteners*] \  
 [time-to-keep-queries-in-minutes=*minutes*] [number-of-top-queries-to-report=*number*]]  
 [--statementtimeout=*statementtimeout*]  
 [--lazyconnectionenlistment={false|true}]  
 [--lazyconnectionassociation={false|true}]  
 [--associatewiththread={false|true}]  
 [--driverclassname=*jdbcdriverclassname*]  
 [--matchconnections={false|true}]  
 [--maxconnectionusagecount=*maxconnectionusagecount*]  
 [--ping={false|true}]  
 [--pooling={false|true}]  
 [--statementcachesize=*statementcachesize*]  
 [--validationclassname=*validationclassname*]  
 [--wrapjdbcobjects={false|true}]  
 [--description *description*]  
 [--property *name=value*][:*name=value*]\*]  
 [--target=*target*]  
*connectionpoolid*

**Description** The create-jdbc-connection-pool subcommand registers a new Java Database Connectivity (“JDBC”) software connection pool with the specified JDBC connection pool name.

A JDBC connection pool with authentication can be created either by using a `--property` option to specify user, password, or other connection information, or by specifying the connection information in the XML descriptor file.

This subcommand is supported in remote mode only.

- Options**
- `--help`  
 `-?`  
Displays the help text for the subcommand.
  - `--datasourceclassname`  
The name of the vendor-supplied JDBC datasource resource manager. An XA or global transactions capable datasource class will implement the `javax.sql.XADataSource` interface. Non-XA or exclusively local transaction datasources will implement the `javax.sql.DataSource` interface.
  - `--restype`  
Required when a datasource class implements two or more interfaces (`javax.sql.DataSource`, `javax.sql.XADataSource`, or `javax.sql.ConnectionPoolDataSource`), or when a driver classname must be provided.
    - If `--restype = java.sql.Driver`, then the `--driverclassname` option is required. You can also specify the `--datasourceclassname` option.
    - If `--restype = javax.sql.DataSource`, `javax.sql.XADataSource` or `javax.sql.ConnectionPoolDataSource`, then the `--datasourceclassname` option is required. You can also specify the `--driverclassname` option.
    - If `--restype` is not specified, then either the `--driverclassname` or `--datasourceclassname` option must be specified, but not both.
  - `--steadypoolsize`  
The minimum and initial number of connections maintained in the pool. The default value is 8.
  - `--maxpoolsize`  
The maximum number of connections that can be created. The default value is 32.
  - `--maxwait`  
The amount of time, in milliseconds, that a caller will wait before a connection timeout is sent. The default is 60000 (60 seconds). A value of 0 forces the caller to wait indefinitely.
  - `--poolresize`  
Number of connections to be removed when `idle-timeout-in-seconds` timer expires. This is the quantity by which the pool will scale up or scale down the number of connections. Scale up: When the pool has no free connections, pool will scale up by this quantity. Scale down: All the invalid and idle connections are removed, sometimes resulting in removing connections of quantity greater than this value. Connections that have been idle for longer than the timeout are candidates for removal. Steadypoolsize will be ensured. Possible values are from 0 to `MAX_INTEGER`. The default value is 2.

**--idletimeout**

The maximum time, in seconds, that a connection can remain idle in the pool. After this time, the implementation can close this connection. This timeout value must be kept shorter than the database server side timeout value to prevent the accumulation of unusable connections in the application. The default value is 300.

**--initsql**

An SQL string that is executed whenever a connection is created from the pool. If an existing connection is reused, this string is not executed. Connections that have idled for longer than the timeout are candidates for removal. This option has no default value.

**--isolationlevel**

The transaction-isolation-level on the pooled database connections. This option does not have a default value. If not specified, the pool operates with the default isolation level that the JDBC driver provides. You can set a desired isolation level using one of the standard transaction isolation levels: read-uncommitted, read-committed, repeatable-read, serializable. Applications that change the isolation level on a pooled connection programmatically risk polluting the pool. This could lead to program errors.

**--isolationguaranteed**

This is applicable only when a particular isolation level is specified for transaction-isolation-level. The default value is true.

This option assures that every time a connection is obtained from the pool, isolation level is set to the desired value. This could have some performance impact on some JDBC drivers. Administrators can set this to false when the application does not change --isolationlevel before returning the connection.

**--isconnectvalidatereq**

If set to true, connections are validated or checked to see if they are usable before giving out to the application. The default value is false.

**--validationmethod**

Type of validation to be performed when is-connection-validation-required is true. Valid settings are: auto-commit, meta-data, table, or custom-validation. The default value is table.

**--validationtable**

The name of the validation table used to perform a query to validate a connection. If is-connection-validation-required is set to true and connection-validation-type set to table, this option is mandatory.

**--failconnection**

If set to true, all connections in the pool must be closed when a single validation check fails. The default value is false. One attempt is made to reestablish failed connections.

**--allownoncomponentcallers**

A pool with this property set to true can be used by non-Java EE components, that is, components other than EJBs or Servlets. The returned connection is enlisted automatically

with the transaction context obtained from the transaction manager. Connections obtained by non-component callers are not automatically cleaned by the container at the end of a transaction. These connections need to be explicitly closed by the caller.

--nontransactionalconnections

A pool with this property set to true returns non-transactional connections. This connection does not get automatically enlisted with the transaction manager.

--validateatmostonceperiod

Specifies the time interval in seconds between successive requests to validate a connection at most once. Setting this attribute to an appropriate value minimizes the number of validation requests by a connection. Default value is 0, which means that the attribute is not enabled.

--leaktimeout

Specifies the amount of time, in seconds, for which connection leaks in a connection pool are to be traced. When a connection is not returned to the pool by the application within the specified period, it is assumed to be a potential leak, and stack trace of the caller will be logged. This option only detects if there is a connection leak. The connection can be reclaimed only if `connection-leak-reclaim` is set to true.

If connection leak tracing is enabled, you can use the Administration Console to enable monitoring of the JDBC connection pool to get statistics on the number of connection leaks. The default value is 0, which disables connection leak tracing.

--leakreclaim

Specifies whether leaked connections are restored to the connection pool after leak connection tracing is complete. Possible values are as follows:

false

Leaked connections are *not* restored to the connection pool (default).

true

Leaked connections are restored to the connection pool.

--statementleaktimeout

Specifies the amount of time, in seconds, after which any statements that have not been closed by an application are to be detected. Applications can run out of cursors if statement objects are not properly closed. This option only detects if there is a statement leak. The statement can be reclaimed only if `statement-leak-reclaim` is set to true. The leaked statement is closed when it is reclaimed.

The stack trace of the caller that creates the statement will be logged when a statement leak is detected. If statement leak tracing is enabled, you can use the Administration Console to enable monitoring of the JDBC connection pool to get statistics on the number of statement leaks. The default value is 0, which disables statement leak tracing.

The following limitations apply to the statement leak timeout value:

- The value must be less than the value set for the connection leak-timeout.
- The value must be greater than the value set for statement-timeout.

--statementleakreclaim

Specifies whether leaked statements are closed and reclaimed after statement leak tracing is complete. Possible values are as follows:

false

Leaked statements are *not* closed and reclaimed (default).

true

Leaked statements are closed and reclaimed.

--creationretryattempts

Specifies the maximum number of times that GlassFish Server retries to create a connection if the initial attempt fails. The default value is 0, which specifies that GlassFish Server does not retry to create the connection.

--creationretryinterval

Specifies the interval, in seconds, between successive attempts to create a connection.

If --creationretryattempts is 0, the --creationretryinterval option is ignored. The default value is 10.

--sqltracelisteners

A list of one or more custom modules that provide custom logging of database activities. Each module must implement the `org.glassfish.api.jdbc.SQLTraceListener` public interface. When set to an appropriate value, SQL statements executed by applications are traced. This option has no default value.

The following properties can be used when SQL tracing is enabled:

`time-to-keep-queries-in-minutes` Specifies the number of minutes will be cached for use in calculating frequently used queries. The default value is 5 minutes.

`number-of-top-queries-to-report` Specifies the number of queries to list when reporting the top and most frequently used queries. The default value is 10 queries.

--statementtimeout

Specifies the length of time in seconds after which a query that is not completed is terminated.

A query that remains incomplete for a long period of time might cause the application that submitted the query to hang. To prevent this occurrence, use this option set a timeout for all statements that will be created from the connection pool that you are creating. When

creating a statement, GlassFish Server sets the `QueryTimeout` property on the statement to the length of time that is specified. The default value is -1, which specifies that incomplete queries are never terminated.

--lazyconnectionenlistment

Specifies whether a resource to a transaction is enlisted only when a method actually uses the resource. Possible values are as follows:

false

Resources to a transaction are always enlisted and *not* only when a method actually uses the resource (default).

true

Resources to a transaction are enlisted *only* when a method actually uses the resource.

--lazyconnectionassociation

Specifies whether a physical connection should be associated with the logical connection only when the physical connection is used, and disassociated when the transaction is completed. Such association and dissociation enable the reuse of physical connections. Possible values are as follows:

false

A physical connection is associated with the logical connection even before the physical connection is used, and is *not* disassociated when the transaction is completed (default).

true

A physical connection is associated with the logical connection only when the physical connection is used, and disassociated when the transaction is completed. The

--lazyconnectionenlistment option must also be set to true.

--associatewiththread

Specifies whether a connection is associated with the thread to enable the thread to reuse the connection. If a connection is not associated with the thread, the thread must obtain a connection from the pool each time that the thread requires a connection. Possible values are as follows:

false

A connection is *not* associated with the thread (default).

true

A connection is associated with the thread.

--driverclassname

The name of the vendor-supplied JDBC driver class. This driver should implement the `java.sql.Driver` interface.

--matchconnections

Specifies whether a connection that is selected from the pool should be matched by the resource adaptor. If all the connections in the pool are homogenous, a connection picked

from the pool need not be matched by the resource adapter, which means that this option can be set to false. Possible values are as follows:

`false`

A connection should *not* be matched by the resource adaptor (default).

`true`

A connection should be matched by the resource adaptor.

--maxconnectionusagecount

Specifies the maximum number of times that a connection can be reused. When this limit is reached, the connection is closed. By limiting the maximum number of times that a connection can be reused, you can avoid statement leaks.

The default value is 0, which specifies no limit on the number of times that a connection can be reused.

--ping

Specifies if the pool is pinged during pool creation or reconfiguration to identify and warn of any erroneous values for its attributes. Default value is false.

--pooling

Specifies if connection pooling is enabled for the pool. The default value is true.

--statementcachesize

The number of SQL statements to be cached using the default caching mechanism (Least Recently Used). The default value is 0, which indicates that statement caching is not enabled.

--validationclassname

The name of the class that provides custom validation when the value of `validationmethod` is `custom-validation`. This class must implement the `org.glassfish.api.jdbc.ConnectionValidation` interface, and it must be accessible to GlassFish Server. This option is mandatory if the connection validation type is set to custom validation.

--wrapjdbcobjects

Specifies whether the pooling infrastructure provides wrapped JDBC objects to applications. By providing wrapped JDBC objects, the pooling infrastructure prevents connection leaks by ensuring that applications use logical connections from the connection pool, not physical connections. The use of logical connections ensures that the connections are returned to the connection pool when they are closed. However, the provision of wrapped JDBC objects can impair the performance of applications. The default value is true.

The pooling infrastructure provides wrapped objects for implementations of the following interfaces in the JDBC API:

- `java.sql.CallableStatement`
- `java.sql.DatabaseMetaData`

- `java.sql.PreparedStatement`
- `java.sql.ResultSet`
- `java.sql.Statement`

Possible values of `--wrapjdbcobjects` are as follows:

`false`

The pooling infrastructure does *not* provide wrapped JDBC objects to applications. (default).

`true`

The pooling infrastructure provides wrapped JDBC objects to applications.

`--description`

Text providing details about the specified JDBC connection pool.

`--property`

Optional attribute name/value pairs for configuring the pool. The following properties are available:

`user`

Specifies the user name for connecting to the database.

`password`

Specifies the password for connecting to the database.

`databaseName`

Specifies the database for this connection pool.

`serverName`

Specifies the database server for this connection pool.

`port`

Specifies the port on which the database server listens for requests.

`networkProtocol`

Specifies the communication protocol.

`roleName`

Specifies the initial SQL role name.

`datasourceName`

Specifies an underlying `XADatasource`, or a `ConnectionPoolDataSource` if connection pooling is done.

`description`

Specifies a text description.

`url`

Specifies the URL for this connection pool. Although this is not a standard property, it is commonly used.



**dynamic-reconfiguration-wait-timeout-in-seconds**

Specifies the time after which in-use connections and in-flight transactions must be retried after a an attribute or property change to the JDBC connection pool. Existing requests that occur within this timeout period will be transparently rolled over to the new pool configuration. New requests will wait for the pool reconfiguration to complete and connections will be acquired using the modified pool configuration.

**LazyConnectionEnlistment**

Deprecated. Use the equivalent attribute. The default value is false.

**LazyConnectionAssociation**

Deprecated. Use the equivalent attribute. The default value is false.

**AssociateWithThread**

Deprecated. Use the equivalent attribute. The default value is false.

**MatchConnections**

Deprecated. Use the equivalent attribute. The default value is true.

**Prefer-Validate-Over-Recreate**

Specifies whether pool resizer should validate idle connections before destroying and recreating them. The default value is true.

**Note** – If an attribute name or attribute value contains a colon, the backslash (\) must be used to escape the colon in the name or value. Other characters might also require an escape character. For more information about escape characters in command options, see the [asadmin\(1M\)](#) man page.

**--target**

Do not specify this option. This option is retained for compatibility with earlier releases. If you specify this option, a syntax error does not occur. Instead, the subcommand runs successfully and displays a warning message that the option is ignored.

**Operands** *connectionpoolid*

The name of the JDBC connection pool to be created.

**Examples** **EXAMPLE 1** Creating a JDBC Connection Pool

This example creates a JDBC connection pool named `sample_derby_pool`.

```
asadmin> create-jdbc-connection-pool
--datasourceclassname org.apache.derby.jdbc.ClientDataSource
--restype javax.sql.XDataSource
--property portNumber=1527:password=APP:user=APP:serverName=
localhost:databaseName=sun-appserv-samples:connectionAttributes=\;
create\=true sample_derby_pool
Command create-jdbc-connection-pool executed successfully
```

The escape character backslash (\) is used in the --property option to distinguish the semicolon (;). Two backslashes (\\) are used to distinguish the equal sign (=).

**Exit Status** 0 subcommand executed successfully  
1 error in executing the subcommand

**See Also** [delete-jdbc-connection-pool\(1\)](#), [list-jdbc-connection-pools\(1\)](#)  
[asadmin\(1M\)](#)