

3

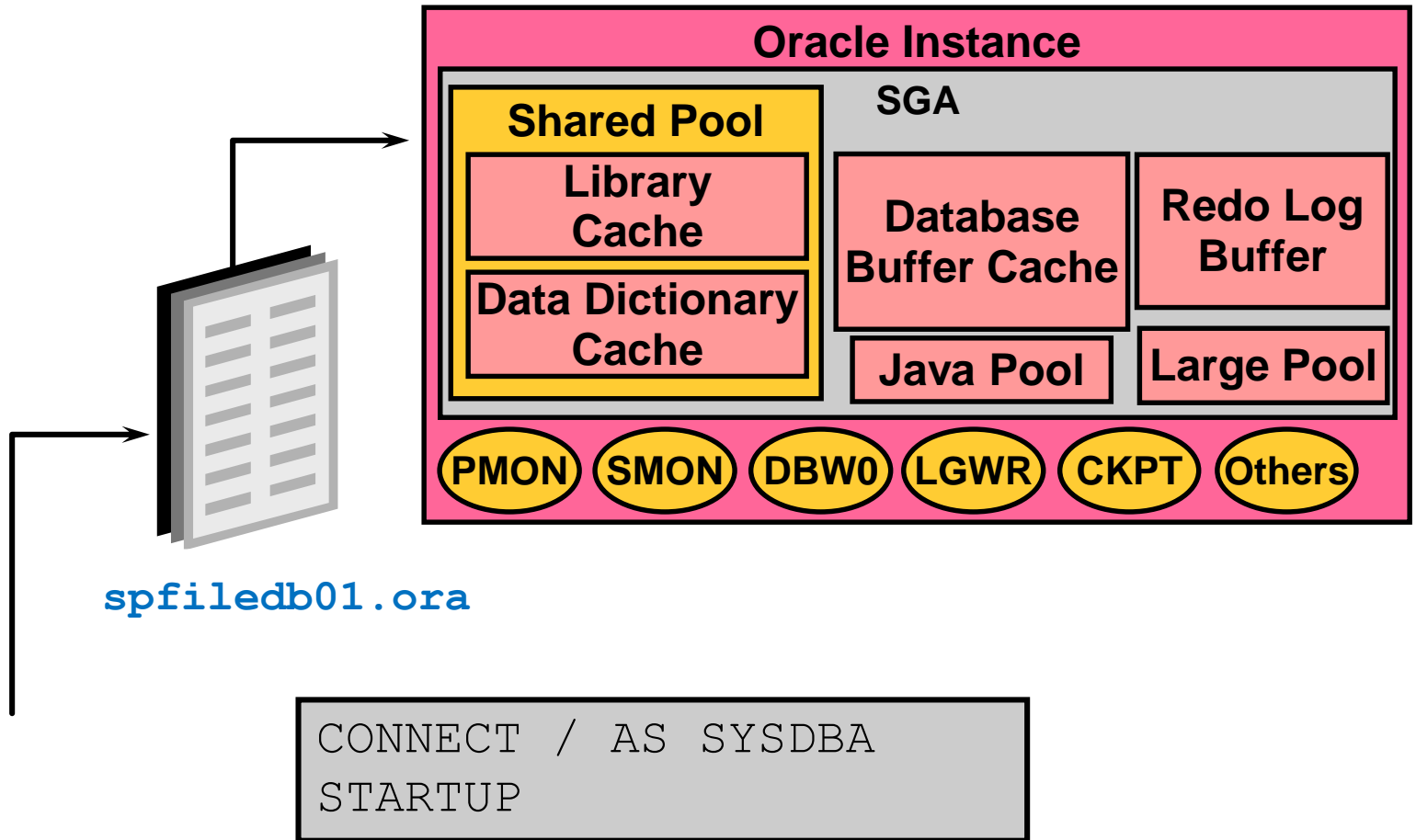
Managing an Oracle Instance

Objectives

After completing this lesson, you should be able to do the following:

- Create and manage initialization parameter files
- Start up and shut down an instance
- Monitor and use diagnostic files

Initialization Parameter Files



Initialization Parameter Files

- Entries are specific to the instance being started
- **Two types of parameters:**
 - **Explicit:** Having an entry in the file
 - **Implicit:** No entry within the file, but assuming the Oracle default values
- Multiple initialization parameter files can exist
- Changes to entries in the file take effect based on the type of initialization parameter file used
 - **Static** parameter file, `PFILE`
 - **Persistent** parameter file, `SPFILE`

PFILE

`initSID.ora`

- Text file
- Modified with an operating system editor
- Modifications made manually
- Changes take effect on the next startup
- Only opened during instance startup (read only)
- Default location is `$ORACLE_HOME/dbs`

Creating a PFILE

- Created from a sample `init.ora` file
 - Sample installed by the Oracle Universal Installer
 - Copy sample using operating system copy command
 - Uniquely identify by database `SID`

```
cp init.ora $ORACLE_HOME/dbs/initdba01.ora
```

- Modify the `initSID.ora`
 - Edit the parameters
 - Specific to database needs

PFILE Example

```
# Initialization Parameter File: initdba01.ora
db_name                = dba01
instance_name          = dba01
control_files          = (
    home/dba01/ORADATA/u01/control01dba01.ctl,
    home/dba01/ORADATA/u02/control01dba02.ctl)
db_block_size          = 4096
db_cache_size          = 4M
shared_pool_size       = 50000000
java_pool_size         = 50000000
max_dump_file_size     = 10240
background_dump_dest   = /home/dba01/ADMIN/BDUMP
user_dump_dest         = /home/dba01/ADMIN/UDUMP
core_dump_dest         = /home/dba01/ADMIN/CDUMP
undo_management        = AUTO
undo_tablespace        = UNDOTBS
```

. . .

SPFILE

`spfileSID.ora`

- Binary file
- Maintained by the Oracle server
- Always resides on the server side
- Ability to make changes persistent across shutdown and startup
- Can self-tune parameter values
- Can have Recovery Manager support backing up to the initialization parameter file

Creating an SPFILE

- Created from a PFILE file

```
CREATE SPFILE = '$ORACLE_HOME/dbs/spfileDBA01.ora'  
FROM PFILE = '$ORACLE_HOME/dbs/initDBA01.ora';
```

where

- SPFILE-NAME: SPFILE to be created
 - PFILE-NAME: PFILE creating the SPFILE
- Can be executed before or after instance startup

SPFILE Example

```
*.background_dump_dest='/home/dba01/ADMIN/BDUMP'  
*.compatible='9.0.0'  
*.control_files='/home/dba01/ORADATA/u01/ctrl01.ctl'  
*.core_dump_dest='/home/dba01/ADMIN/CDUMP'  
*.db_block_size=4096  
*.db_name='dba01 '  
*.db_domain='world'  
*.global_names=TRUE  
*.instance_name='dba01'  
*.remote_login_passwordfile='exclusive '  
*.java_pool_size=50000000'  
*.shared_pool_size=50000000  
*.undo_management='AUTO'  
*.undo_tablespace='UNDOTBS '
```

. . .

STARTUP Command Behavior

- Order of Precedence
 - spfileSID.ora
 - Default SPFILE
 - initSID.ora
 - Default PFILE
- Specified PFILE can override precedence

```
STARTUP PFILE = $ORACLE_HOME/dbs/initDBA1.ora
```

- PFILE can indicate to use SPFILE

```
SPFILE = /database/startup/spfileDBA1.ora
```

Modifying Parameters in SPFILE

- Parameter value changes made by ALTER SYSTEM

```
ALTER SYSTEM SET undo_tablespace = 'UNDO2';
```

- Specify whether the change is temporary or persistent

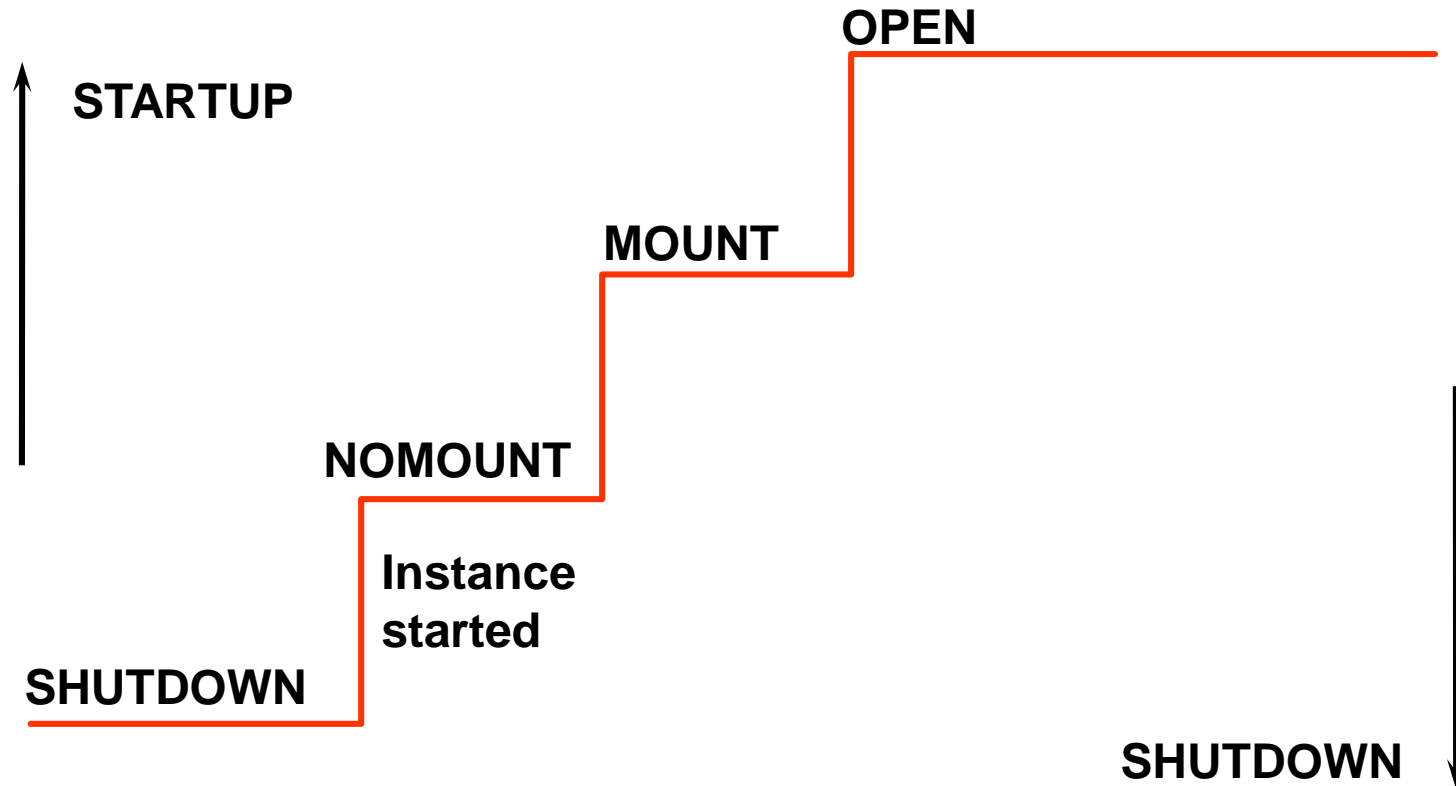
```
ALTER SYSTEM SET undo_tablespace = 'UNDO2'  
SCOPE=BOTH;
```

- Delete or reset values

```
ALTER SYSTEM RESET undo_suppress_errors  
SCOPE=BOTH SID='*';
```

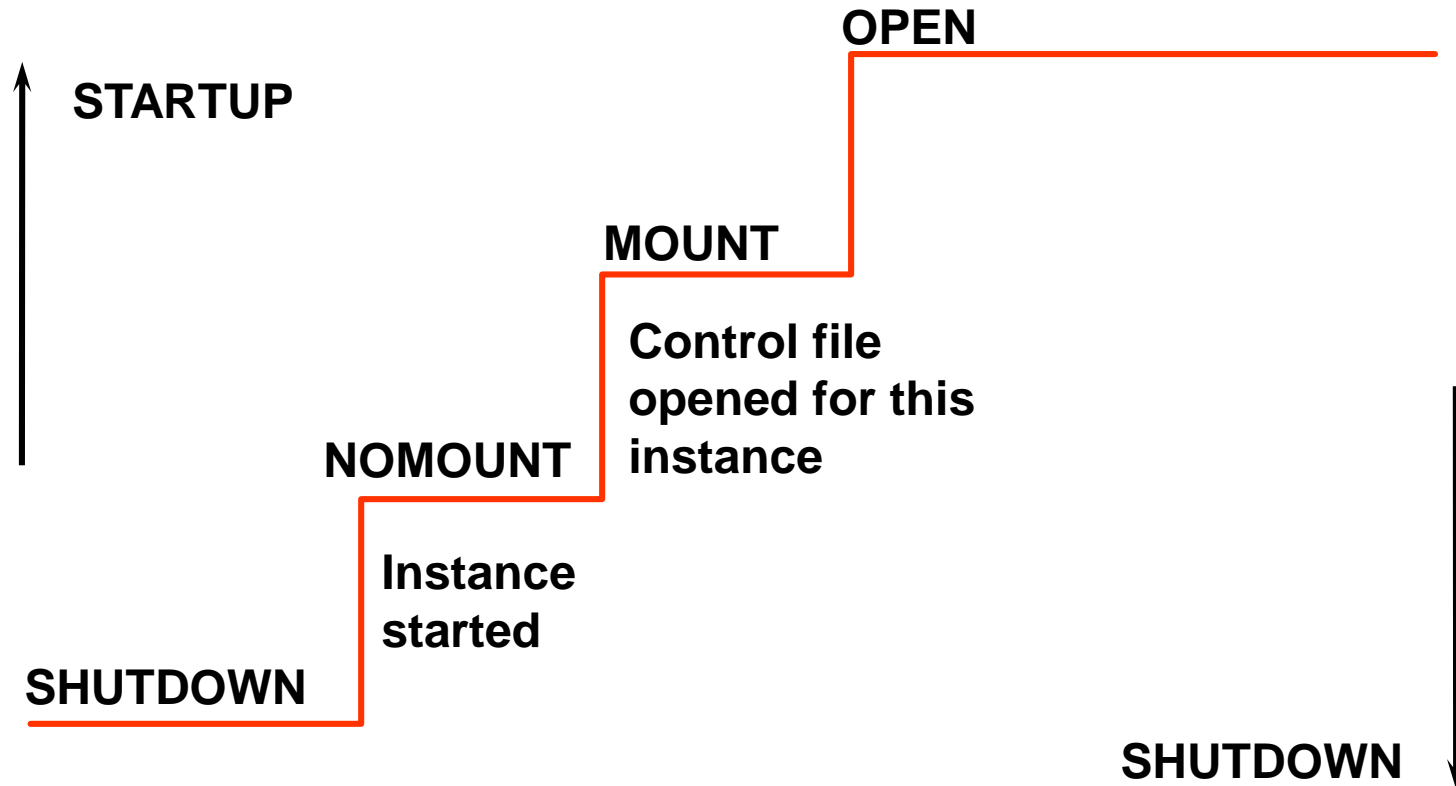
Starting Up a Database

NOMOUNT



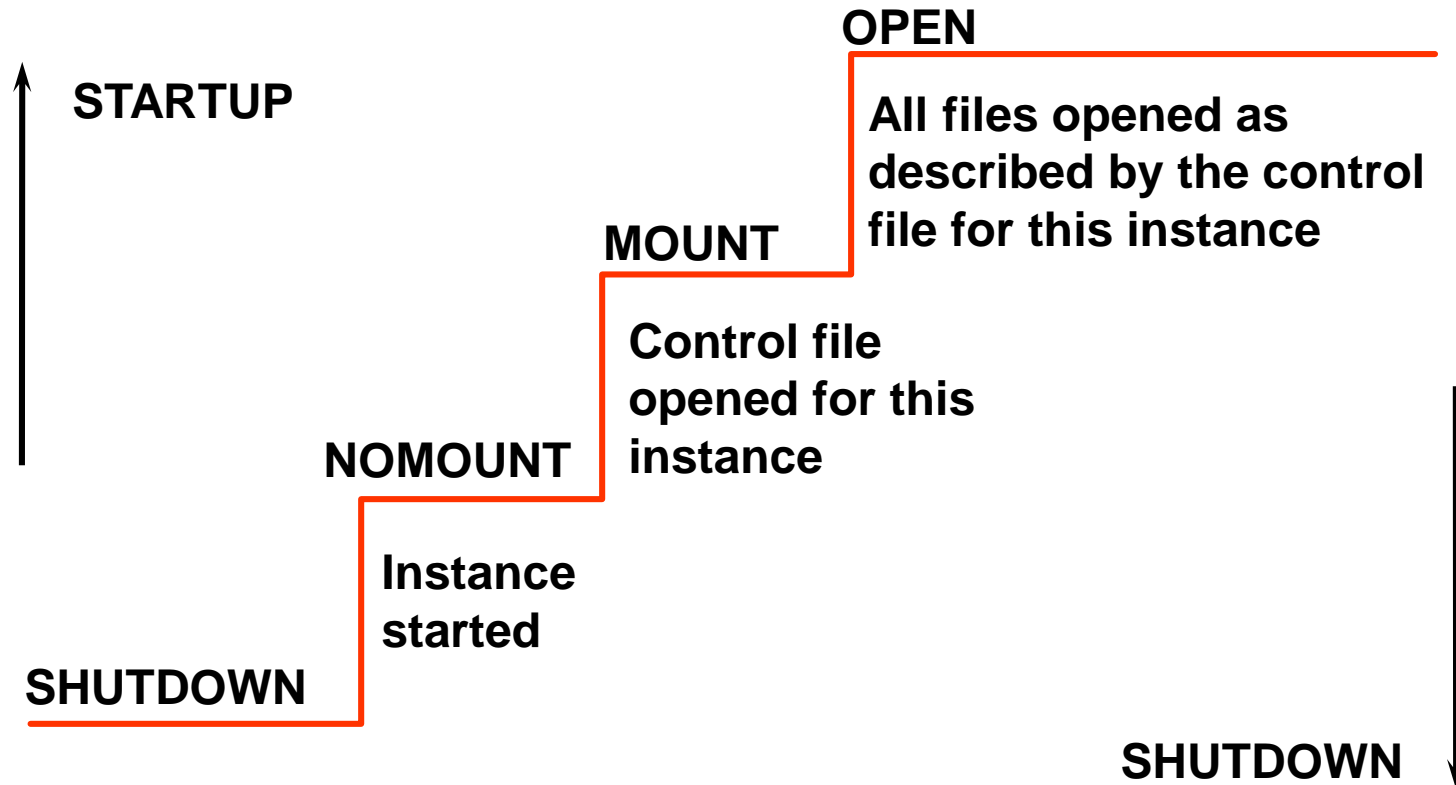
Starting Up a Database

MOUNT



Starting Up a Database

OPEN



STARTUP Command

Start up the instance and open the database:

```
STARTUP
```

```
STARTUP PFILE=$ORACLE_HOME/dbs/initdb01.ora
```


ALTER DATABASE Command

- Change the state of the database from `NOMOUNT` to `MOUNT`:

```
ALTER DATABASE db01 MOUNT;
```

- Open the database as a read-only database:

```
ALTER DATABASE db01 OPEN READ ONLY;
```

Opening a Database in Restricted Mode

- Use the `STARTUP` command to restrict access to a database:

```
STARTUP RESTRICT
```

- Use the `ALTER SYSTEM` command to place an instance in restricted mode:

```
ALTER SYSTEM ENABLE RESTRICTED SESSION;
```

Opening a Database in Read-Only Mode

- Opening a database in read-only mode

```
STARTUP MOUNT  
ALTER DATABASE OPEN READ ONLY;
```

- Can be used to:
 - Execute queries
 - Execute disk sorts using locally managed tablespaces
 - Take datafiles offline and online, but not tablespaces
 - Perform recovery of offline datafiles and tablespaces

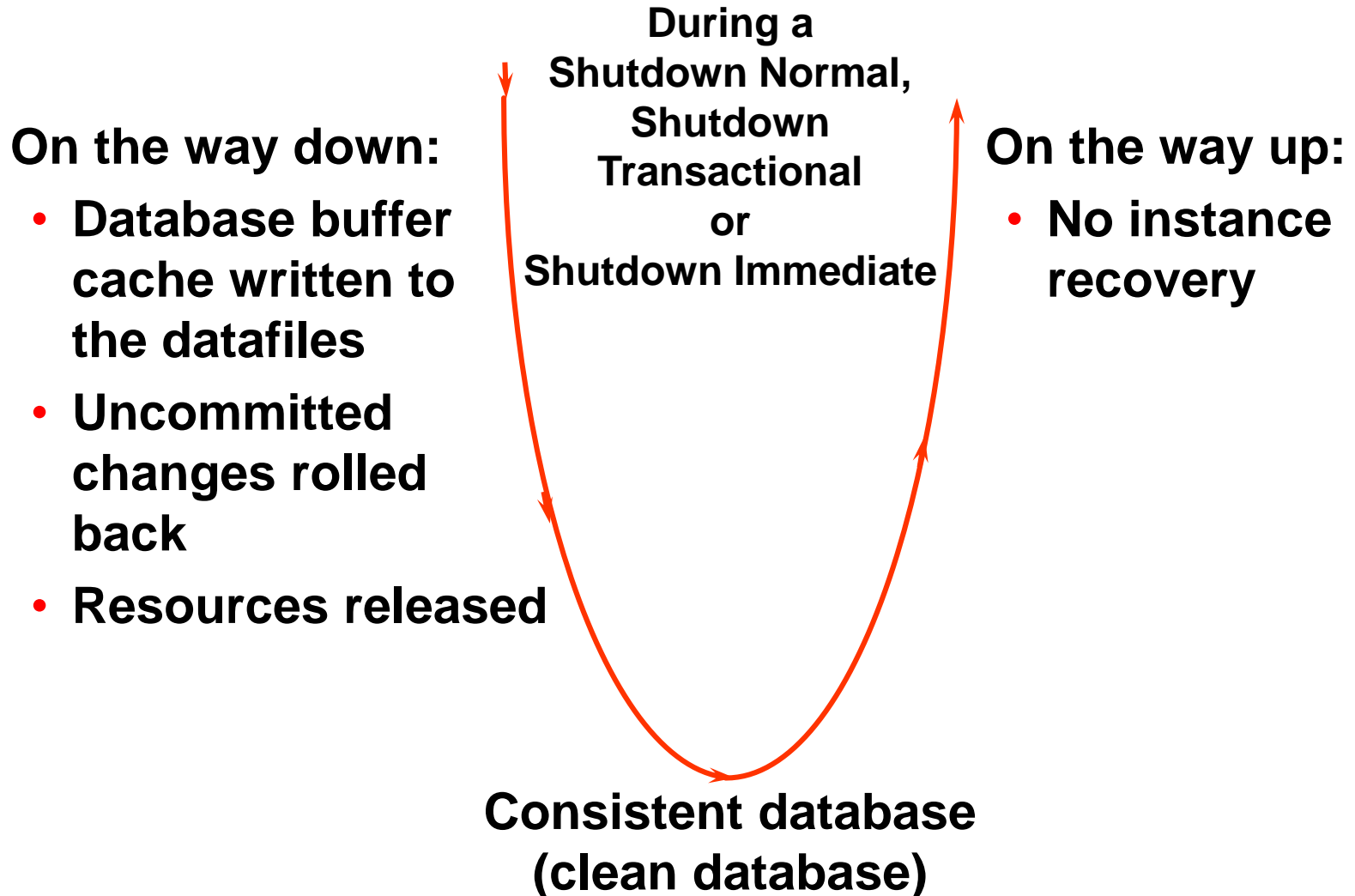
Shutting Down the Database

Shutdown Mode	A	I	T	N
Allow new connections	No	No	No	No
Wait until current sessions end	No	No	No	Yes
Wait until current transactions end	No	No	Yes	Yes
Force a checkpoint and close files	No	Yes	Yes	Yes

Shutdown mode:

- A = ABORT
- I = IMMEDIATE
- T = TRANSACTIONAL
- N = NORMAL

Shutdown Options

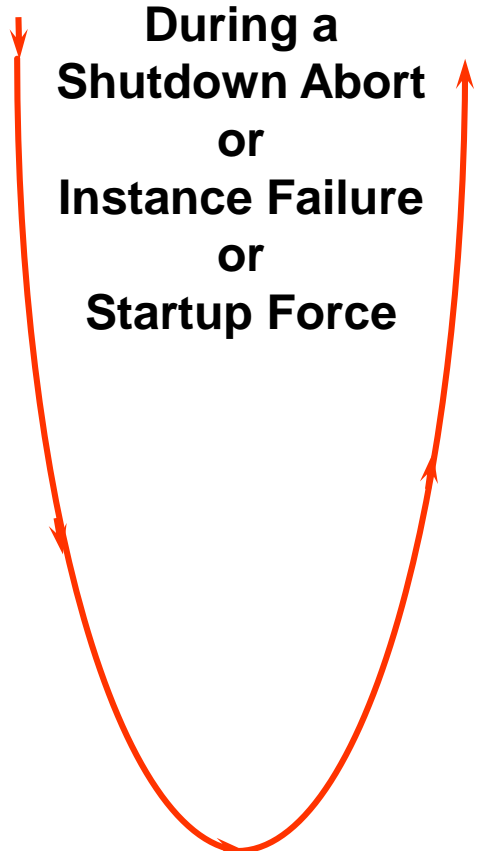


Shutdown Options

On the way down:

- Modified buffers are not written to the datafiles
- Uncommitted changes are not rolled back

During a
Shutdown Abort
or
Instance Failure
or
Startup Force



On the way up:

- Redo logs used to reapply changes
- Undo segments used to roll back uncommitted changes
- Resources released

**Inconsistent database
(dirty database)**

Monitoring an Instance Using Diagnostic Files

- **Diagnostic files**
 - Contain information about significant events encountered
 - Used to resolve problems
 - Used to better manage the database on a day-to-day basis
- **Several types exist:**
 - `alertSID.log` file
 - Background trace files
 - User trace files

Alert Log File

- **alertSID.log file:**
 - Records the commands
 - Records results of major events
 - Used for day-to-day operational information
 - Used for diagnosing database errors
- Each entry has a time stamp associated with it
- Must be managed by DBA
- Location defined by **BACKGROUND_DUMP_DEST**

Background Trace Files

- **Background trace files**
 - Logs errors detected by any background process
 - Used to diagnose and troubleshoot errors
- **Created when a background process encounters an error**
- **Location defined by BACKGROUND_DUMP_DEST**

User Trace File

- **User trace file**
 - Produced by the user process
 - Can be generated by a server process
 - Contains statistics for traced SQL statements
 - Contains user error messages
- Created when a user encounters user session errors
- Location is defined by `USER_DUMP_DEST`
- Size defined by `MAX_DUMP_FILE_SIZE`

Enabling or Disabling User Tracing

- **Session level:**
 - Using the **ALTER SESSION** command:
`ALTER SESSION SET SQL_TRACE = TRUE`
 - Executing DBMS procedure:
`dbms_system.SET_SQL_TRACE_IN_SESSION`
- **Instance level**
 - Setting the initialization parameter:
`SQL_TRACE = TRUE`

Summary

In this lesson, you should have learned how to:

- **Create and manage initialization parameter files**
- **Start up and shut down an instance**
- **Monitor and use diagnostic files**

Practice 3 Overview

This practice covers the following topics:

- **Creating an SPFILE**
- **Starting up and shutting down the database in different modes**

