5

Creating an Oracle Database by Using DBCA

ORACLE

Objectives

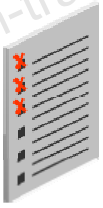
After completing this lesson, you should be able to do the following:

- Create a database by using the Database Configuration Assistant (DBCA)
- Generate database creation scripts by using DBCA
- Manage database design templates by using DBCA
- Configure database options by using DBCA

Planning the Database

As a DBA, you must plan:

- The logical storage structure of the database and its physical implementation:
  - How many disk drives do you have? What type of storage is being used?
  - How many data files will you need? (Plan for growth.)
  - How many tablespaces will you use?
  - What types of information will be stored?
  - Are there any special storage requirements due to type or size?
- Overall database design
- Database backup strategy



It is important to plan how the logical storage structure of the database will affect system performance and various database management operations. For example, before creating any tablespaces for your database, you should know how many data files will make up the tablespace, what type of information will be stored in each tablespace, and on which disk drives the data files will be physically stored. Information such as the availability of network attached storage (NAS) and the bandwidth for the private storage network are important. If storage area networks (SAN) are going to be used, knowing how the logical volumes are configured and the stripe size is useful.

When planning the overall logical storage of the database structure, take into account the effects that this structure will have when the database is actually created and running. You may have database objects that have special storage requirements due to type or size.

In distributed database environments, this planning stage is extremely important. The physical location of frequently accessed data dramatically affects application performance.

During the planning stage, develop a backup strategy for the database. You can alter the logical storage structure or design of the database to improve backup efficiency. Backup strategies are introduced in a later lesson.

Types of Databases

- General purpose or transaction processing:
  - Online transaction processing (OLTP) system, for example a retail billing system for a software house or a nursery
- Custom:
  - Multipurpose database (perhaps combined OLTP and data warehouse functionality)
- Data warehouse:
  - Research and marketing data
  - State or federal tax payments
  - Professional licensing (doctors, nurses, and so on)

Different types of databases have their own specific instance and storage requirements. Your Oracle Database software includes templates for the creation of these different types of databases. Characteristics of these examples are the following:

- **General purpose:** For general purpose or transaction processing usage such as working with transactions and storing them for a medium length of time
- **Custom:** For customized databases that do not fit into the general purpose or data warehouse template
- **Data warehouse:** For storing data for long periods and retrieving them in read operations

Choosing the Appropriate Character Set

- Oracle Database supports different classes of character-encoding schemes:
  - Single-byte character sets
    - 7-bit
    - 8-bit
  - Multibyte character sets, including Unicode
- The character set is chosen at the time of database creation. Choose the character set that best meets your business requirements now and in the future because it can be difficult to change character sets later on.
- In general, Unicode is recommended because it is the most flexible character set.

When computer systems process characters, they use numeric codes instead of the graphical representation of the character. An *encoded character set* maps numeric codes to characters that a computer or terminal can display and receive. Different character sets support different character repertoires. Because character sets are typically based on a particular writing script, they can support more than one language. However, script-based character sets are restricted in the sense that they are limited to groups of languages based on similar scripts. Universal character sets encompass most major scripts of the modern world and provide a more useful solution to multilingual support. For information about the Unicode standards, see the website at http:www.unicode.org.

Oracle Database supports three classes of encoding schemes: Single-byte, Varying-width multibyte, and Universal. Choose the correct character set that best meets your business requirements now and in the future because it can be difficult to change character sets later on. For best performance, choose a character set that avoids character set conversion and uses the most efficient encoding for the languages desired. Single-byte character sets result in better performance than multibyte character sets, and they also are the most efficient in terms of space requirements. However, single-byte character sets limit how many languages you can support.

To choose your correct database character set, evaluate your current and future business requirements, as well as technical requirements (for example, the XML and Java standards require Unicode). In general, Oracle recommends the use of Unicode for all new databases, because it is the most flexible character set and avoids future conversions.

Single-Byte Character Sets

In a single-byte character set, each character occupies one byte. Single-byte 7-bit encoding schemes can define up to 128 (2^7) characters; single-byte 8-bit encoding schemes can define up to 256 (2^8) characters.

Examples of Single-Byte Schemes

7-bit character set:

- American Standard Code for Information Interchange (ASCII) 7-bit American (`US7ASCII`)

8-bit character set:

- International Organization for Standards (ISO) 8859-1 West European (`WE8ISO8859P1`)
- DEC 8-bit West European (`WE8DEC`)
- Extended Binary Coded Decimal Interchange Code (EBCDIC) Code Page 1144 8-bit Italian (`I8EBCDIC1144`)

Multibyte Character Sets

A varying-width multibyte character set is represented by one or more bytes per character. Multibyte character sets are commonly used for Asian language support. Some multibyte encoding schemes use the value of the most significant bit to indicate whether a byte represents a single byte or is part of a series of bytes representing a character. However, other character-encoding schemes differentiate single-byte from multibyte characters. A shift-out control code, sent by a device, indicates that any successive bytes are double-byte characters until a shift-in code is encountered. Shift-sensitive encoding schemes are used primarily on IBM platforms.

Unicode is a universal encoded character set that enables information from any language to be stored using a single character set. Unicode provides a unique code value for every character, regardless of the platform, program, or language.

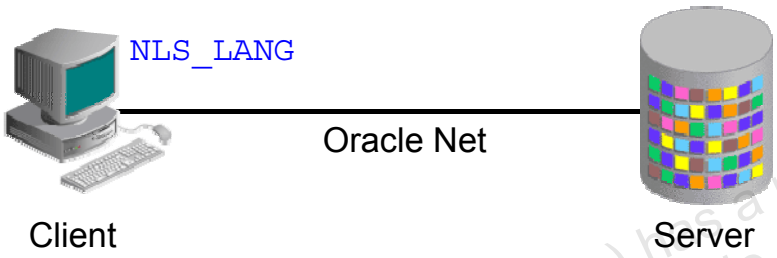
The Unicode standard has been adopted by many software and hardware vendors. Many operating systems and browsers now support Unicode. Unicode is required by standards such as XML, Java, JavaScript, LDAP, and WML. It is also synchronized with the ISO/IEC 10646 standard.

Examples of Varying-Width Multibyte Schemes

- Shift-JIS 16-bit Japanese (`JA16SJIS`)
- MS Windows Code Page 950 with Hong Kong Supplementary Character Set HKSCS-2001 (`ZHT16HKSCS`)
- Unicode 4.0 UTF-8 Universal character set (`AL32UTF8`): A variable-width type of encoding and also a strict superset of ASCII.

Understanding How Character Sets Are Used

- Oracle Net compares the client NLS_LANG setting to the character set on the server.
- If needed, conversion occurs automatically and transparently.



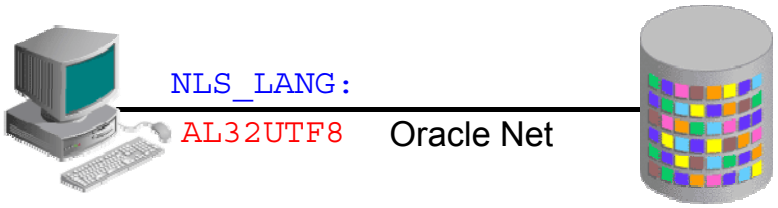
The NLS_LANG parameter defines a client terminal's character-encoding scheme. Different clients can use different encoding schemes. Data passed between the client and the server is converted automatically between the two encoding schemes. The database's encoding scheme should be a superset, or equivalent, of all the clients' encoding schemes. The conversion is transparent to the client application.

When the database character set and the client character set are the same, the database assumes that the data being sent or received is of the same character set, so no validations or conversions are performed.

Character set conversion may be required in a client/server environment, if a client application resides on a different platform than the server and if the platforms do not use the same character-encoding schemes. Character data passed between the client and the server must be converted between the two encoding schemes. Character conversion occurs automatically and transparently through Oracle Net.

Setting the NLS_LANG Initialization Parameter

Example:



Client
Windows English
Code page: WE8MSWIN1252

Server
Database character set:
AL32UTF8

No conversion occurs, because it does not seem to be required.

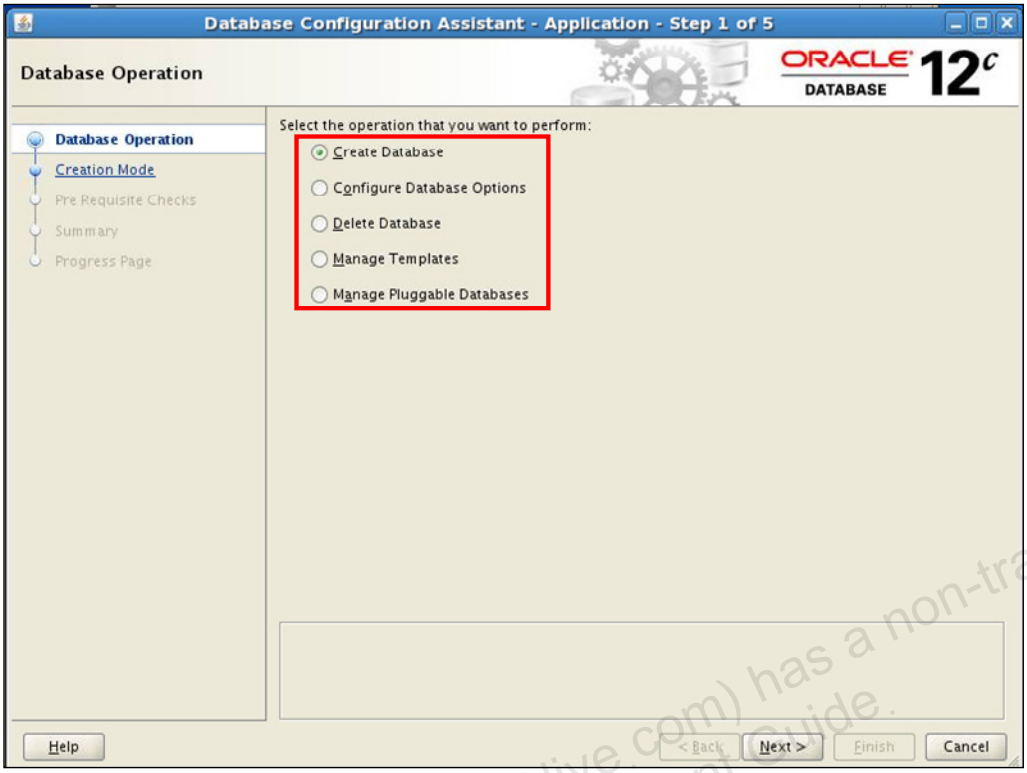
Issue: Invalid data are entered into the database.

Invalid data usually enters a database when the NLS_LANG parameter is not set properly on the client. The NLS_LANG value should reflect the encoding of the incoming data.

- When the NLS_LANG parameter is set properly, the database can automatically convert incoming data from the client operating system.
- When the NLS_LANG parameter is not set properly, the data entering the database is not converted properly.

For example, suppose that the database character set is AL32UTF8, the client is an English Windows operating system (code page: WE8MSWIN1252), and the NLS_LANG setting on the client is AL32UTF8. Data entering the database is encoded in WE8MSWIN1252 and is not converted to AL32UTF8 data because the NLS_LANG setting on the client matches the database character set. Thus the Oracle Database server assumes that no conversion is necessary, and invalid data is entered into the database.

Using the DBCA to Create a Database



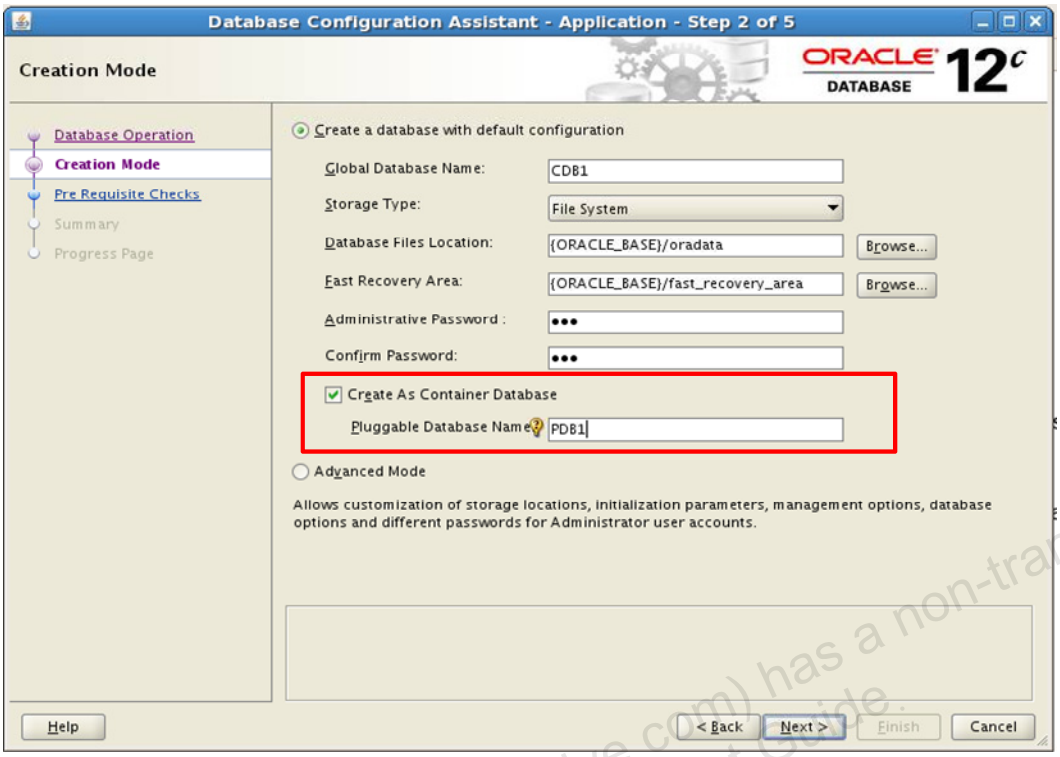
You can use the DBCA to create, change the configuration of, or delete a database. You can also create a database from a list of predefined templates or use an existing database as a sample to create a new database or template. The “Manage Pluggable Databases” option enables you to create, configure, unplug, and delete PDBs.

The DBCA provides several options to allow you to create a database to meet your needs. The DBCA provides a series of pages where you enter configuration information. On most pages, the DBCA provides default settings that you can accept if they apply.

Invoke DBCA to create a database as follows:

1. Log on to your computer as a member of the OS DBA group that is authorized to install the Oracle software. If required, set environment variables and enter `dbca` to invoke the DBCA.
2. Select “Create Database” and click Next.

Creating a Container Database by Using DBCA

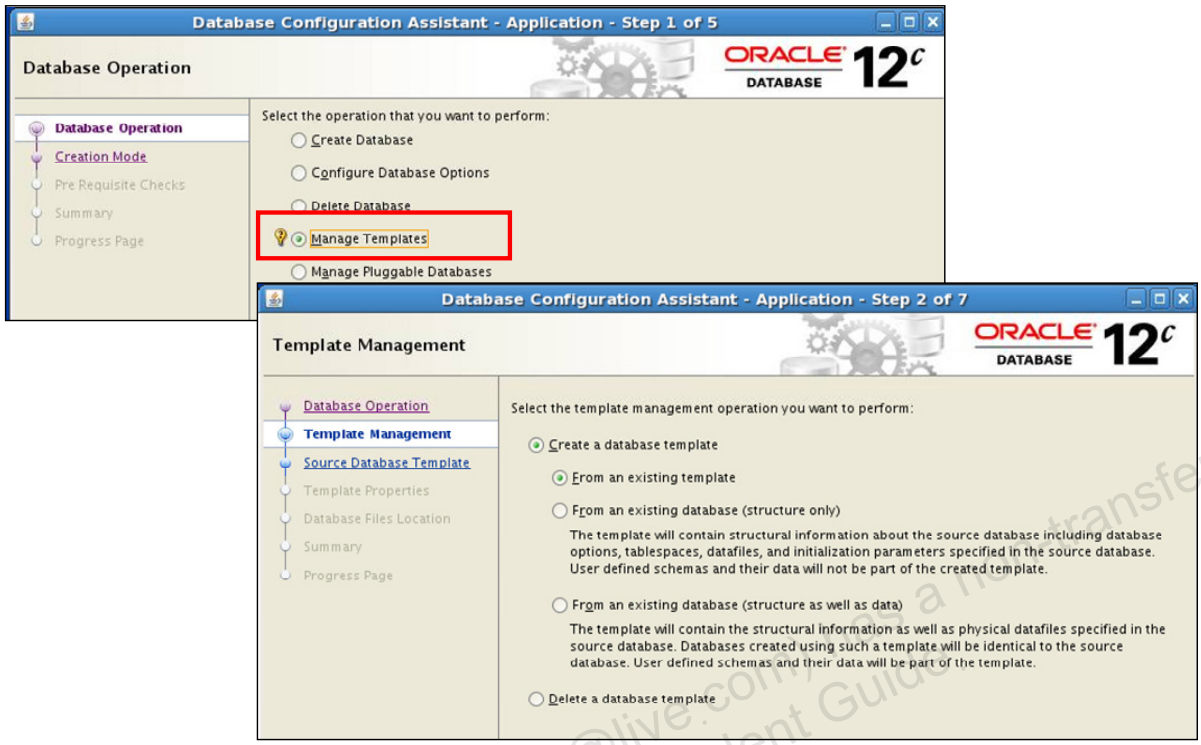


To create a container database by using DBCA, select “Create As Container Database.”

You also have to provide a pluggable database name when you select the “Create a Database with Default Configuration” check box. If you select “Advanced Mode,” you can create an empty multitenant container database with only the root and seed containers.

In Advanced Mode, you can register the CDB with Enterprise Manager Cloud Control and set passwords for the `SYS` and `SYSTEM` users.

Creating a Database Design Template



A template is a predefined database definition that you use as a starting point for a new database. If you do not create a template as part of the database creation process, you can do it at any time by invoking the DBCA and choosing the Manage Templates operation.

There are three ways to create a template:

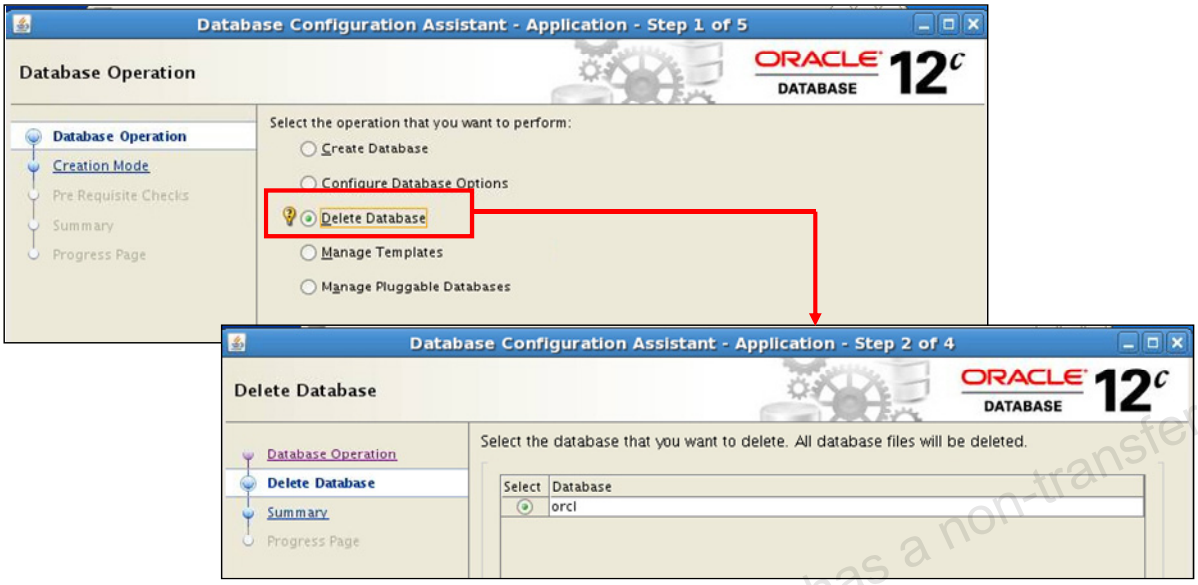
- From an existing template
- From an existing database (structure only)
- From an existing database (structure as well as data)

The DBCA guides you through the steps to create a database design template.

If you no longer need a specific template use the “Delete a database template” option on the Template Management page of the DBCA.

Note: Templates you created will appear in the Database Templates list when you create a new database using the DBCA.

Using the DBCA to Delete a Database



Start the DBCA by entering `dbca` in a terminal window. To delete a database, perform the following steps:

1. On the “Database Operation” page, select “Delete a Database.” Then click Next.
2. Select the database that you want to delete and click Next.
3. Click Finish on the Summary page.
4. Click Yes to confirm your deletion.
5. When the deletion is completed, you will be asked whether you want to perform another operation. Answer accordingly.

Note: The database being deleted must be opened so that DBCA can connect to the database to determine file location information.

Dropping a database involves removing its data files, redo log files, control files, and initialization parameter files. You can drop a database manually using the `DROP DATABASE` SQL statement. The `DROP DATABASE` statement deletes all control files and all other database files listed in the control file. To use the `DROP DATABASE` statement successfully, all of the following conditions must apply:

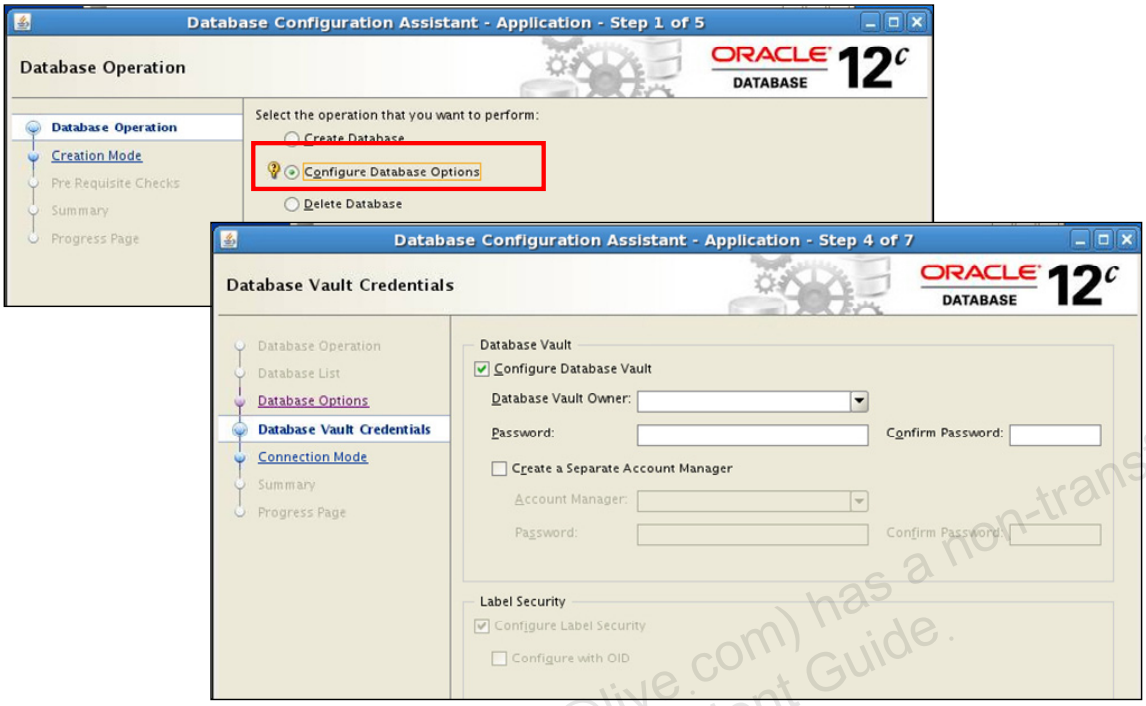
- The database must be mounted and closed.
- The database must be mounted exclusively (not in shared mode).
- The database must have been started in `RESTRICT` mode.

An example of these statements is:

```
STARTUP RESTRICT FORCE MOUNT;
DROP DATABASE;
```

The `DROP DATABASE` statement has no effect on archived log files, nor does it have any effect on copies or backups of the database. It is best to use Recovery Manager (RMAN) to delete such files. If the database is on raw disks, the actual raw disk special files are not deleted.

Using the DBCA for Additional Tasks



You can use the DBCA to configure database options such as Oracle Database Vault and Oracle Label Security. Not all options are installed by default during database software installation, so it may be necessary to install additional software prior to attempting to configure a database to use certain options.

Notes

- For more information about Oracle Database Vault, see the *Oracle Database Vault Administrator's Guide*.
- For more information about Oracle Label Security, see the *Oracle Label Security Administrator's Guide*.

Summary

In this lesson, you should have learned how to:

- Create a database by using the Database Configuration Assistant (DBCA)
- Generate database creation scripts by using DBCA
- Manage database design templates by using DBCA
- Configure database options by using DBCA

Practice 5

- 5-1: Creating a Non-CDB
- 5-2: Creating a CDB