Relational Database Architecture
3

Terminal Learning Objective

Action: Maintain an Oracle Database.

Condition: Given a student handout and the Oracle DBA Handbook.

***Standard:** Students must pass a multiple choice written examination with a minimum score of 70% and successfully create an Oracle Storage Structure during a scenario-based performance evaluation.

*See Student Evaluation Plan (SEP) for details.

Oracle Products and Services

- Oracle databases
- Oracle Application Server
- Oracle applications
- Oracle Collaboration Suite
- Oracle Developer Suite
- Oracle services

ORACLE®

3-3

Oracle Products

- **Oracle databases:** The Oracle database is the first database that is designed for enterprise grid computing (the most flexible and cost-effective way to manage information and applications).
- **Oracle Application Server:** Oracle's Java 2 Platform, Enterprise Edition Cloud Control (J2EE)–certified server integrates everything that is needed to develop and deploy Web-based applications. The application server deploys e-business portals, Web services, and transactional applications, including PL/SQL, Oracle Forms, and J2EE-based applications.
- **Oracle applications:** Oracle E-Business Suite is a complete set of business applications for managing and automating processes across your organization.
- **Oracle Collaboration Suite:** Oracle Collaboration Suite is a single, integrated system for all your organization's communications data: voice, e-mail, fax, wireless, calendar information, and files.
- **Oracle Developer Suite:** Oracle Developer Suite is a complete, integrated environment that combines application development and business intelligence tools.
- **Oracle services:** Services such as Oracle Consulting and Oracle University provide you with the necessary expertise for your Oracle projects. For useful links to a variety of resources, see the appendix titled "Next Steps, Continuing Your Education."

Overview

- **This course focuses on the features of Oracle Database 19*c* that are applicable to database administration.**
- **Previous experience with Oracle databases is required for a full understanding of many of the new features.**
- **Hands-on practices emphasize functionality rather than test knowledge.**

3-4

This course is designed to introduce you to the new features of Oracle Database 12*c* that are applicable to the work usually performed by database administrators and related personnel. The course does not attempt to provide every detail about a feature or cover aspects of a feature that were available in previous releases (except when defining the context for a new feature or comparing past behavior with current behavior). Consequently, the course is most useful to you if you have administered other versions of Oracle databases, particularly Oracle Database 11*g*. Even with this background, you should not expect to be able to implement all of the features discussed in the course without supplemental reading, especially the Oracle Database 12*c* documentation.

The course consists of instructor-led lessons and demonstrations, plus many hands-on practices that enable you to see for yourself how certain new features behave. As with the course content in general, these practices are designed to introduce you to the fundamental aspects of a feature. They are not intended to test your knowledge of unfamiliar syntax or to provide an opportunity for you to examine every nuance of a new feature. The length of this course precludes such activity. Consequently, you are strongly encouraged to use the provided scripts to complete the practices rather than struggle with unfamiliar syntax.

Oracle Database 19c: “c” Stands for Cloud (continued)

Automatic Storage Management spreads database data across all disks, creates and maintains a storage grid, and provides the highest input/output (I/O) throughput with minimal management costs. As disks are added or dropped, ASM redistributes the data automatically. (There is no need for a logical volume manager to manage the file system.) Data availability increases with optional mirroring, and you can add or drop disks online. For more information, see the lesson titled “Managing Database Storage Structures.”

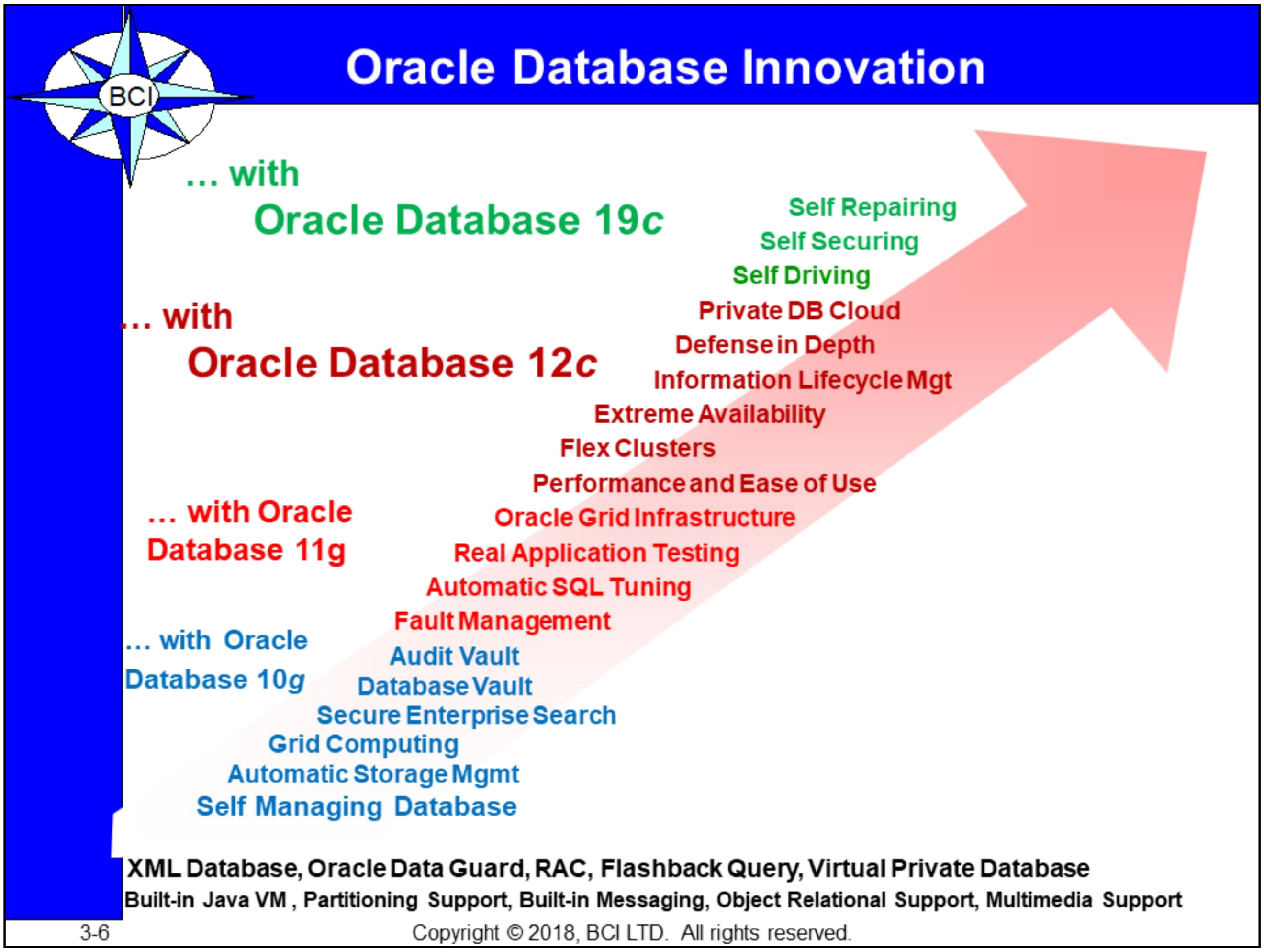
Oracle’s **Real Application Clusters** runs and scales all application workloads on a cluster of servers and offers the following features:

- **Integrated clusterware:** This includes functionality for cluster connectivity, messaging and locking, cluster control, and recovery. It is available on all platforms that are supported by Oracle Database 12c.
- **Automatic workload management:** Rules can be defined to automatically allocate processing resources to each service both during normal operations and in response to failures. These rules can be dynamically modified to meet the changing business needs. This dynamic resource allocation within a database grid is unique to Oracle RAC.
- **Automatic event notification to the mid-tier:** When a cluster configuration changes, the mid-tier can immediately adapt to instance failover or availability of a new instance. This enables end users to continue working in the event of instance failover without the delays typically caused by network timeouts. In the event of new instance availability, the mid-tier can immediately start load balancing connections to that instance. Oracle Database 12c Java Database Connectivity (JDBC) drivers have the “fast connection failover” functionality that can be automatically enabled to handle these events.

Golden Gate provides a unified framework for information sharing, combining message queuing, data replication, event notification, data warehouse loading, and publishing and subscribing functionality into a single technology. Oracle Streams can keep two or more data source copies synchronized when updates are applied at either site. It can automatically capture database changes, propagate the changes to subscribing nodes, apply changes, and detect and resolve data update conflicts. Golden Gate can be used directly by applications as a message-queuing or workflow feature, enabling communications between applications in the grid.

Enterprise Manager Grid Control manages grid wide operations that include managing the entire stack of software, provisioning users, cloning databases, and managing patches. It can monitor the performance of all applications from the point of view of your end users. Grid Control views the performance and availability of the grid infrastructure as a unified whole rather than as isolated storage units, databases, and application servers. You can group hardware nodes, databases, and application servers into single logical entities and manage a group of targets as one unit.

Note: In this course, you use Enterprise Manager Database Console to manage one database at a time.



As a result of its early focus on innovation, Oracle has maintained the lead in the industry with a large number of trend-setting products.

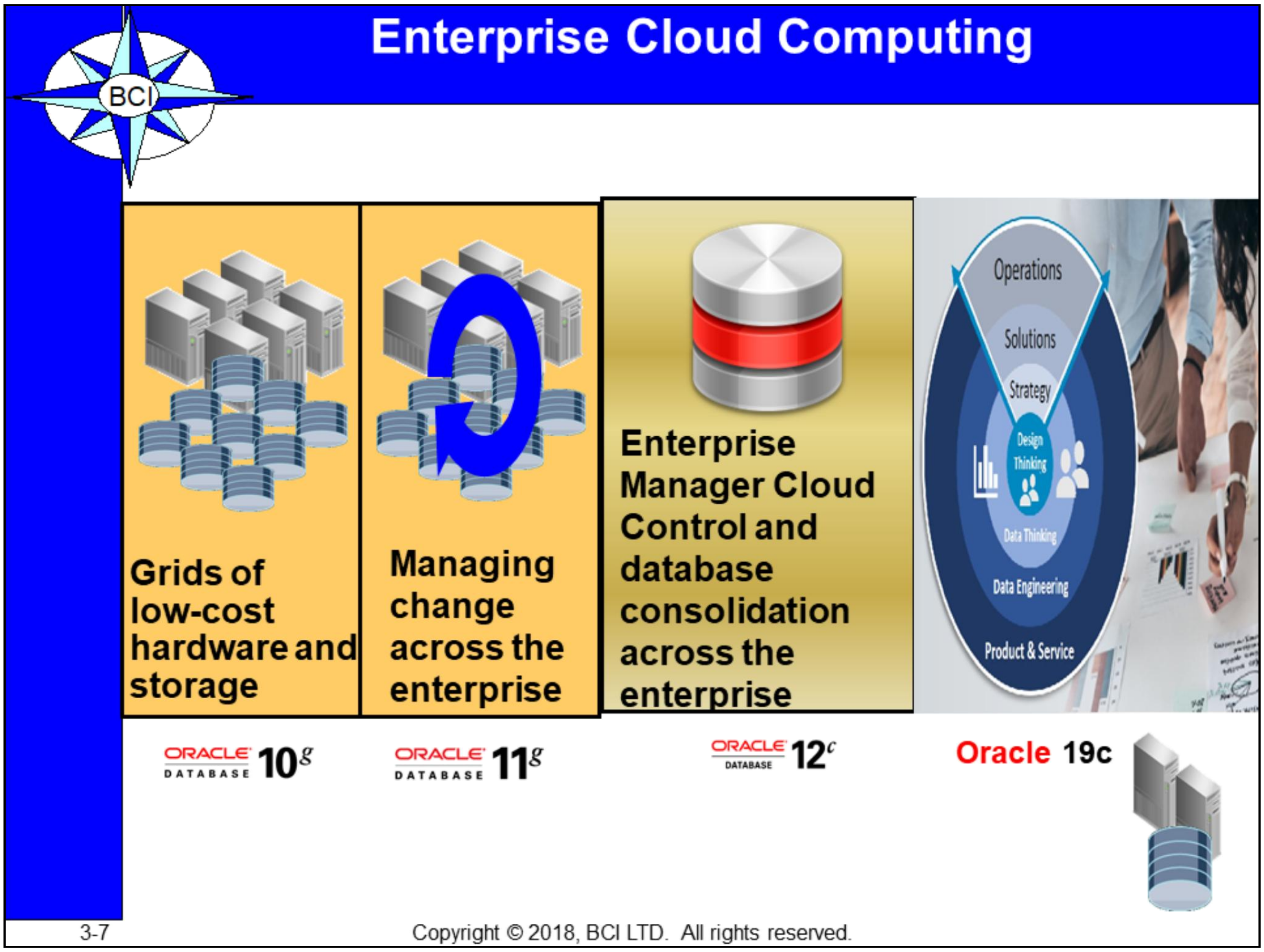
Some of the marquee areas in the Oracle Database 12*c* and 19c release are the following:

• Self-Driving. With easy to define workloads and policies, automation then makes them happen.

• Self-Securing. With automated protection from both external attacks and internal users.

• Self-Repairing. With automated protection from all downtime.

- Private Database Cloud
- Defense in Depth included Oracle Data Redaction, Real Application Security
- Information Lifecycle Management (ILM), which includes hot/cold data classification, declarative compression and tiering, In-database Archiving and Valid-Time Temporal
- Flex Clusters
- Extreme Availability, which includes Data Guard Far-Sync and Application Continuity
- Lower Cost Migrations
- Performance and Ease of Use, which includes "just in time" optimizations, attribute clustering, and zone maps for Exadata only



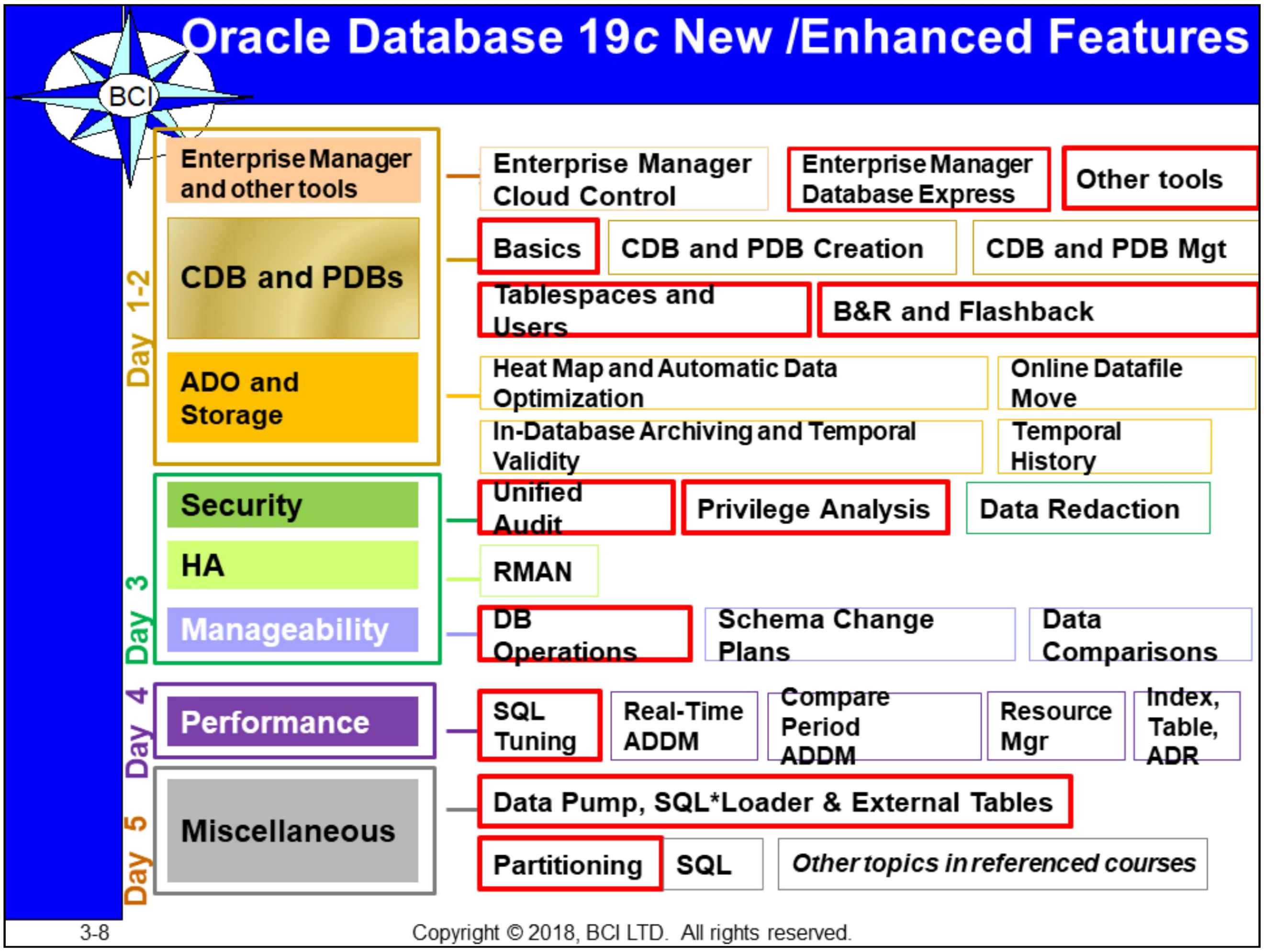
Oracle Database 10*g* was the first database management system designed for grid computing.

Oracle Database 11*g* consolidates and extends Oracle's unique ability to deliver the benefits of grid computing, transforming data centers from silos of isolated system resources to shared pools of servers and storage.

Oracle Database 12*c* and Enterprise Manager Cloud Control are designed for cloud computing. Cloud computing creates a complete, pre-integrated, off-the-shelf private cloud solution that allows you to quickly transform the enterprise data center into a private cloud.

The key benefits are the following:

- Reduce servers sprawl and improve CPU utilization by consolidating on fewer servers
- Reduce the amount of time a DBA spends installing and configuring databases, by automating deployment of standard database configurations
- Single console to manage entire Cloud life cycle – plan, set up, deliver, operate
- Prevent resource hogging by setting quotas for individual users
- Forecast future resource needs by analyzing trending reports
- Compute chargeback based on performance and configuration metrics



Features, Options and Schedule

Enterprise Manager Cloud Control: Oracle Enterprise Manager is Oracle's integrated enterprise IT management product line, and provides the industry's first complete cloud lifecycle management solution.

Oracle Multitenant: Oracle Multitenant is a new option in Oracle Database 12*c*. The multitenant architecture enables an Oracle database to contain a portable set of schemas, objects, and related structures that appears logically to an application as a separate database. This self-contained collection is called a pluggable database (PDB). A multitenant container database (CDB) contains one or several PDBs.

Information Lifecycle Management: One challenge facing each organization is to understand how its data evolves and grows, monitor how its usage changes over time, and decide how long it should survive, while adhering to all the rules and regulations that now apply to that data. Information Lifecycle Management (ILM) is designed to address these issues, with a combination of processes, policies, software, and hardware so that the appropriate technology can be used for each stage in the life cycle of the data. ILM offers new features including heat Map, Automatic Data Optimization and enhancements to In-Database Archiving, including the new Temporal Validity feature.

Defense in Depth

- Unified auditing makes audit information available in a uniform format and centralized in a single place accessible only by privileged users.
- New administrative privileges help separation of duties.
- Privilege Analysis contributes to identify privilege usage and to decide which privileges must be revoked.
- Oracle Data Redaction prevents the display of sensitive data by performing masking in each application, defining redaction at the database level and displaying redacted data regardless of the access method, even to DBAs.

High Availability

RMAN enhancements include recovering a table or table partitions from existing backups, using backup sets for cross-platform data transport, connecting with the new `SYSBACKUP` privilege, and many more.

Manageability

- Database Operation monitoring extends and generalizes Real Time SQL Monitoring.
- Database Change Management Pack is all about making sure that enterprises can manage all changes, either proactively as they promote a database from development test to production, or reactively when any unwanted changes are done in a production environment that are causing problems. Enterprise Manager with Schema Change Plans and Data Comparisons provides this capability to automatically identify and detect these changes so that organizations can take immediate actions and automate the process of applying the corrective actions without scripts and manual intervention.

Performance

- SQL tuning is essentially based on performance automation with Adaptive SQL Plan Management, Adaptive Execution Plans and Optimizer Statistics Management.
- Real-Time ADDM and Emergency Monitoring detect that the system is sick and analyzes the root causes of the hanging situation to help you find the appropriate action.
- In a situation where the performance has decreased or increased, Compare Period Advisor identifies what has changed and why and which changes may have impacted the performance between two periods of time. This helps you take appropriate actions upon relevant analysis.
- Resource Manager is now adapted to multitenant container databases and pluggable databases to share CPU and other resources among the different containers in a database.

There are also many other enhancements about Oracle Data Pump, SQL*Loader, partitioning and online operations, and SQL covered in the course.

Many other subjects like Automatic Storage Management, Real Application Cluster, Data Guard, Real Application Testing and many other subjects of expertise cannot be presented in this course. You can find references to these courses, demonstrations, and OBEs in the last lesson.

Oracle Database Architecture

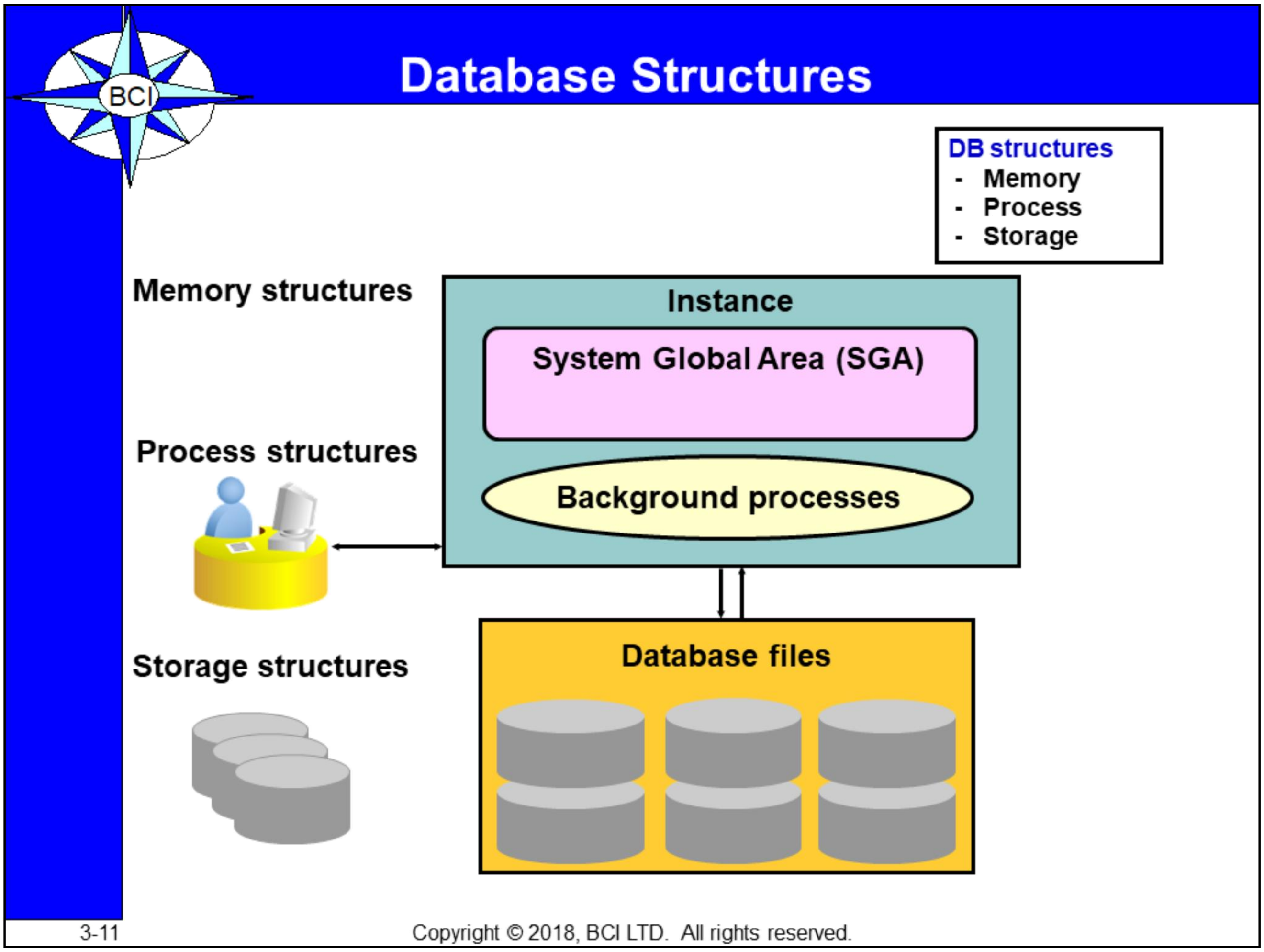
An Oracle server:

- Is a database management system that provides an open, comprehensive, integrated approach to information management
- Consists of an Oracle instance and one or more Oracle databases
- SQL> show sga
- SQL> show parameter sga

3-10

Oracle Database Architecture

The Oracle server is the key to information management. In general, an Oracle server must reliably manage a large amount of data in a multiuser environment so that many users can concurrently access the same data. All this must be accomplished while delivering high performance. An Oracle server must also prevent unauthorized access and provide efficient solutions for failure recovery.



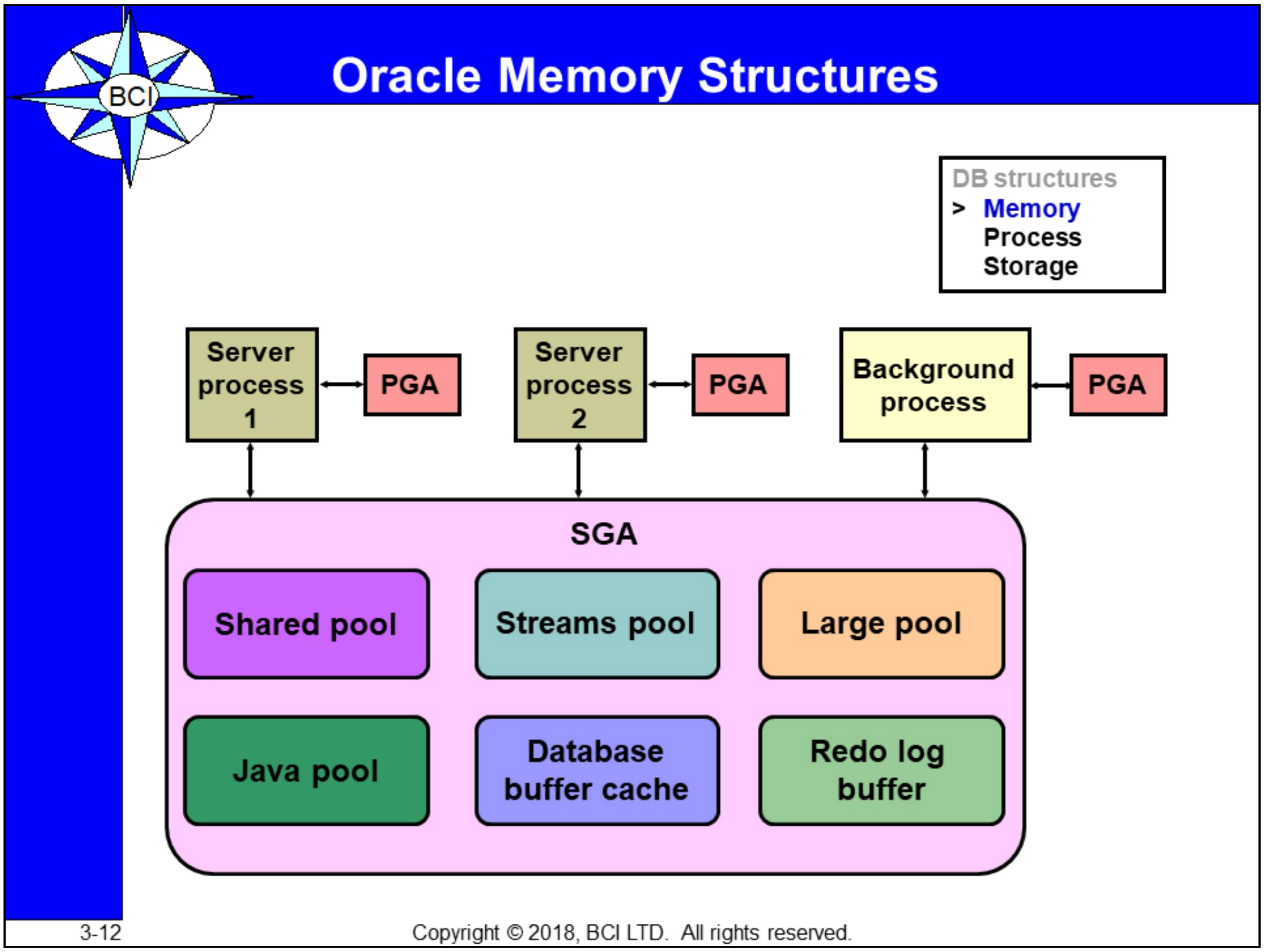
Database Structures

Each running Oracle database is associated with an Oracle instance. When a database is started on a database server, the Oracle software allocates a shared memory area called the System Global Area (SGA) and starts several Oracle background processes. This combination of the SGA and the Oracle processes is called an Oracle instance.

After starting an instance, the Oracle software associates the instance with a specific database. This is called mounting the database. The database is then ready to be opened, which makes it accessible to authorized users. Multiple instances can execute concurrently on the same computer, each accessing its own physical database or databases.

You can look at the Oracle database architecture as various interrelated structural components.

An Oracle database uses memory structures and processes to manage and access the database. All memory structures exist in the main memory of the computers that constitute the database server. Processes are jobs that work in the memory of these computers. A process is defined as a “thread of control” or a mechanism in an operating system that can run a series of steps.



Oracle Memory Structures

The basic memory structures associated with an Oracle instance include the following:

- **System Global Area (SGA):** Shared by all server and background processes
- **Program Global Area (PGA):** Private to each server and background process. There is one PGA for each process.

The SGA is a memory area that contains data and control information for the instance.

The SGA includes the following data structures:

- **Database buffer cache:** Caches blocks of data retrieved from the database
- **Redo log buffer:** Caches redo information (used for instance recovery) until it can be written to the physical redo log files stored on the disk
- **Shared pool:** Caches various constructs that can be shared among users
- **Large pool:** Is an optional area that provides large memory allocations for certain large processes, such as Oracle backup and recovery operations, and I/O server processes
- **Java pool:** Is used for all session-specific Java code and data within the Java Virtual Machine (JVM)
- **Streams pool:** Is used by Oracle GoldenGate for Replication.

When you start the instance by using Enterprise Manager or SQL*Plus, the amount of memory allocated for the SGA is displayed.

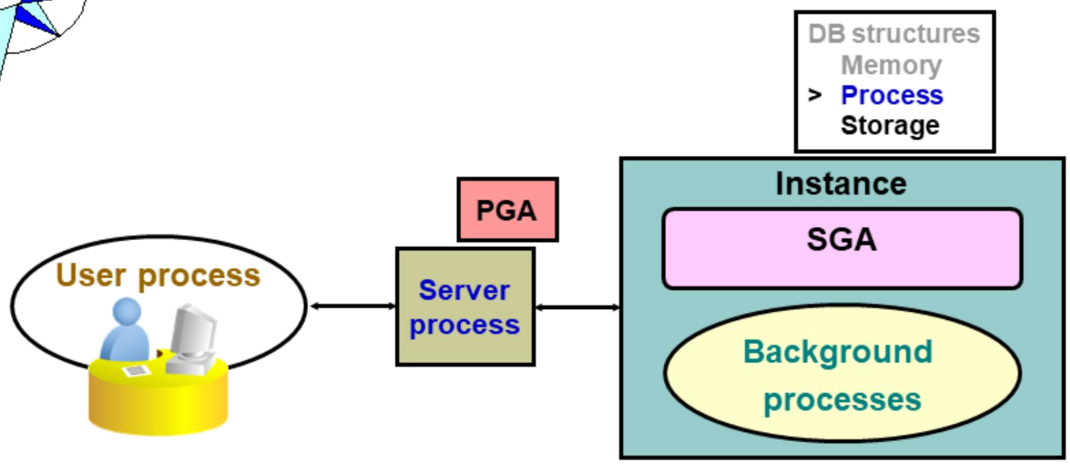
Oracle Memory Structures (continued)

A Program Global Area (PGA) is a memory region that contains data and control information for each server process. An Oracle server process services a client's requests. Each server process has its own private PGA that is created when the server process is started. Access to the PGA is exclusive to that server process, and the PGA is read and written only by the Oracle code acting on its behalf.

With the dynamic SGA infrastructure, the size of the database buffer cache, the shared pool, the large pool, the Java pool, and the Streams pool changes without shutting down the instance.

The Oracle database uses initialization parameters to create and configure memory structures. For example, the `SGA_TARGET` parameter specifies the total amount of space available to the SGA. If you set `SGA_TARGET` to 0, Automatic Shared Memory Management is disabled.

Process Structures

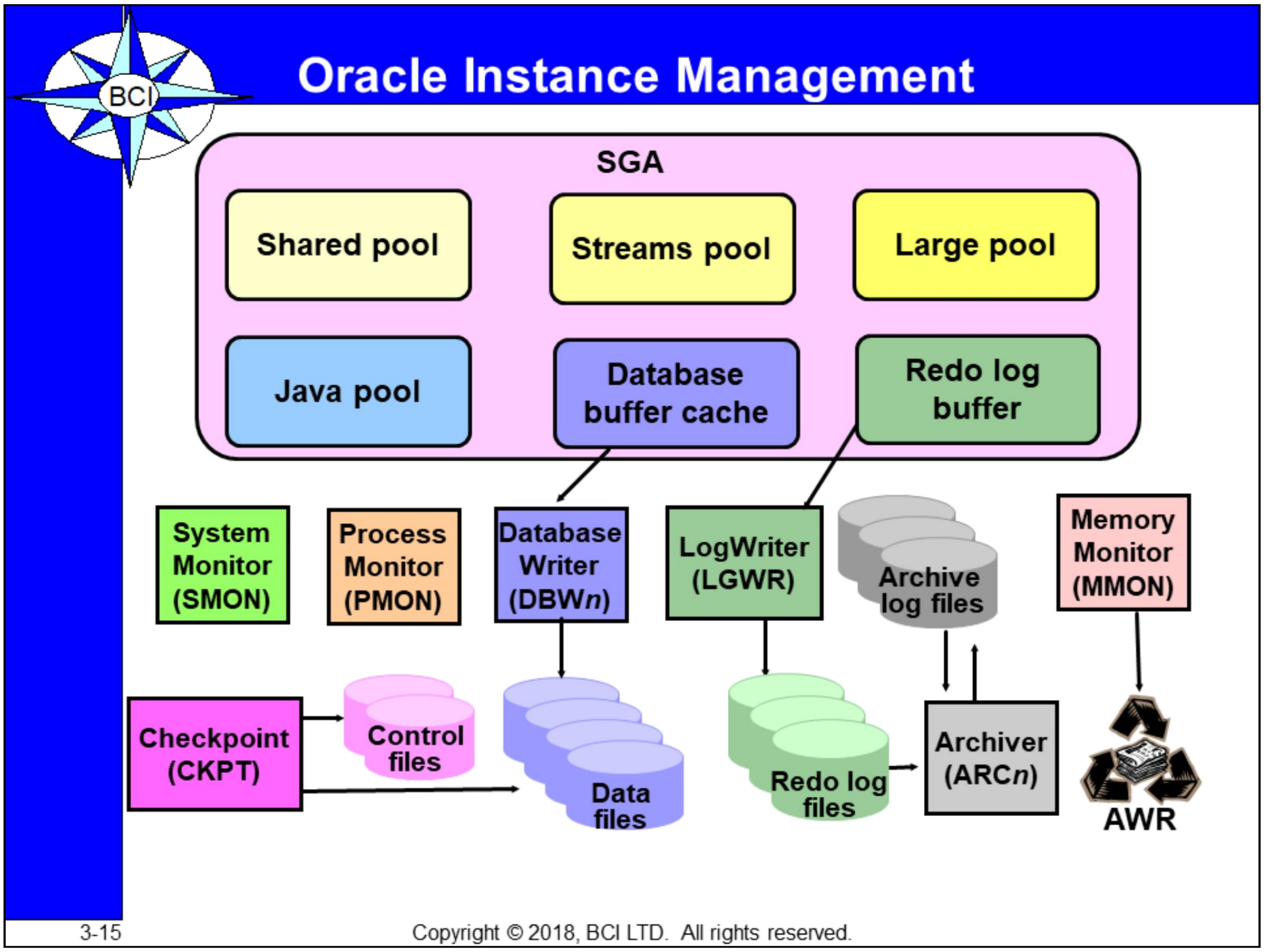


- **User process:** Is started at the time a database user requests a connection to the Oracle server
- **Server process:** Connects to the Oracle instance and is started when a user establishes a session
- **Background processes:** Are started when an Oracle instance is started

3-14

Process Structures

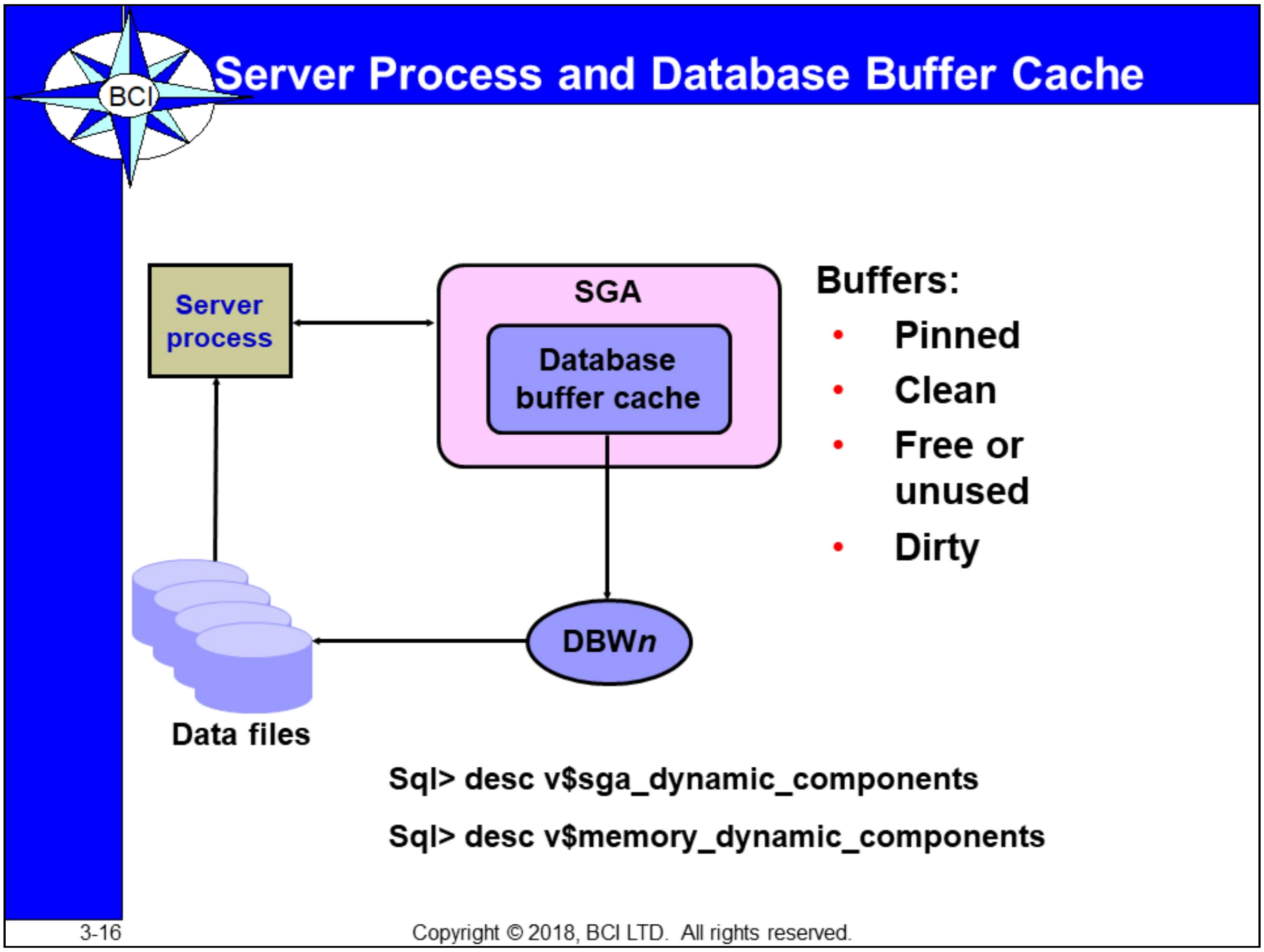
When you invoke an application program or an Oracle tool, such as Enterprise Manager, the Oracle server creates a server process to execute the commands issued by the application. The Oracle server also creates a set of background processes for an instance that interact with each other and with the operating system to manage the memory structures, asynchronously perform I/O to write data to disk, and perform other required tasks. Which background processes are present depends on the features that are being used in the database.



Oracle Instance Management

An Oracle database server consists of an Oracle database(s) and an Oracle instance. An Oracle instance is made up of memory structures, known as the System Global Area (SGA), and background processes that handle much of the behind-the-scenes work involved in running an instance. The most common background processes are the following:

- **System Monitor (SMON):** Performs crash recovery when the instance is started following a failure
- **Process Monitor (PMON):** Performs process cleanup when a user process fails
- **Memory Monitor (MMON):** Performs memory management across background processes
- **Database Writer (DBW*n*):** Writes modified blocks from the database buffer cache to the data files on the disk
- **Checkpoint (CKPT):** Updates all the data files and control files of the database to indicate the most recent checkpoint
- **LogWriter (LGWR):** Writes redo log entries to the disk
- **Archiver (ARC*n*):** Copies redo log files to the archival storage when a log switch occurs

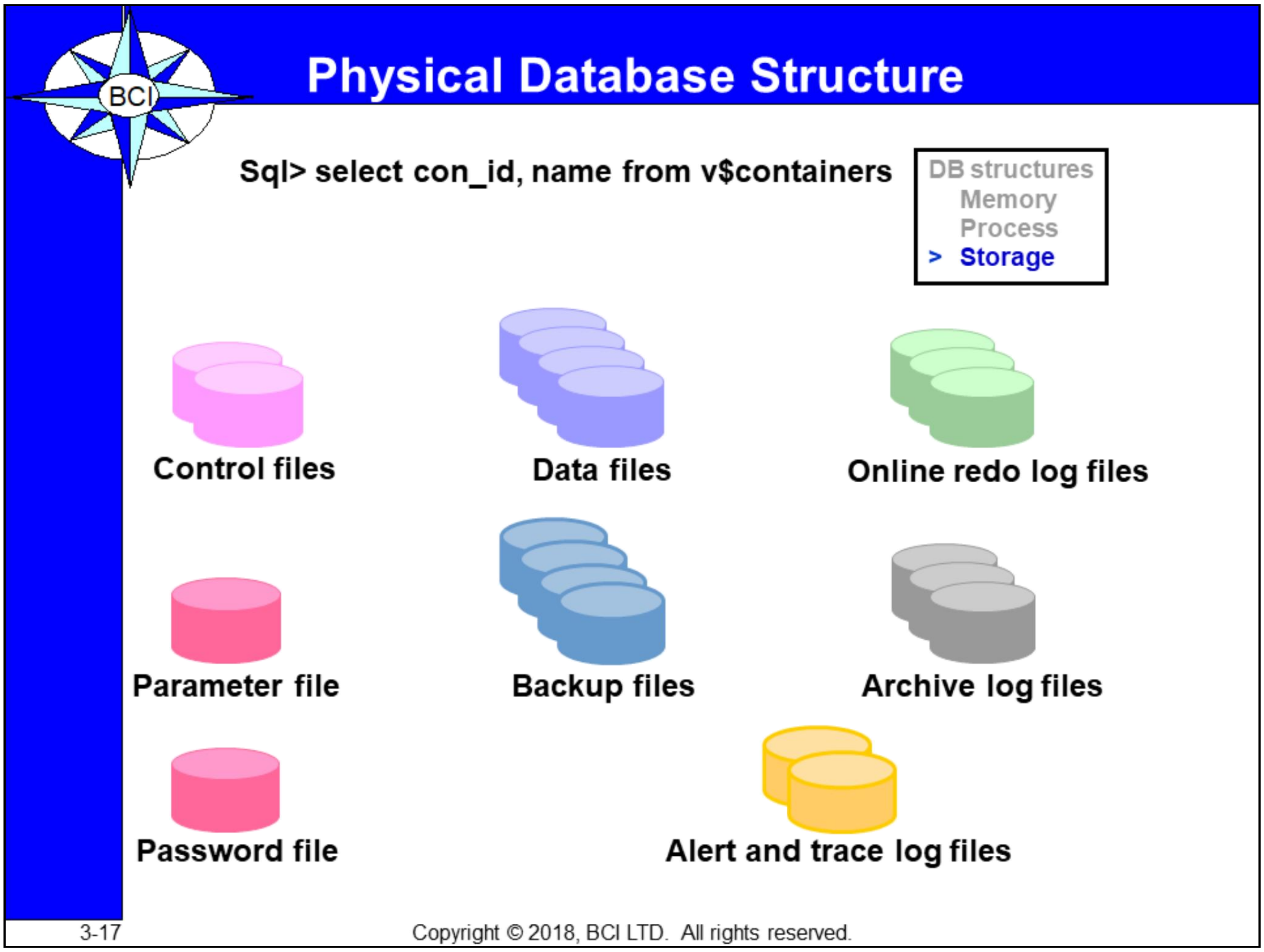


Server Process and Database Buffer Cache

When a query is processed, the Oracle server process looks in the database buffer cache for any blocks that it needs. If the block is not found in the database buffer cache, the server process reads the block from the data file and places a copy in the database buffer cache. Because subsequent requests for the same block may find the block in memory, the requests may not require physical reads. The Oracle server uses the least recently used algorithm to age out buffers that have not been accessed recently to make room for new blocks in the database buffer cache.

Buffers in the buffer cache can be in one of the following four states:

- **Pinned:** Multiple sessions are kept from writing to the same block at the same time. Other sessions wait to access the block.
- **Clean:** The buffer is now unpinned and is a candidate for immediate aging out, if the current contents (data block) are not referenced again. Either the contents are in sync with the block contents stored on the disk or the buffer contains a consistent read (CR) snapshot of a block.
- **Free or unused:** The buffer is empty because the instance has just started. This state is very similar to the clean state, except that the buffer has not been used.
- **Dirty:** The buffer is no longer pinned but the contents (data block) have changed and must be flushed to the disk by DBW*n* before it can be aged out.



Physical Database Structure

The files that constitute an Oracle database are organized into the following:

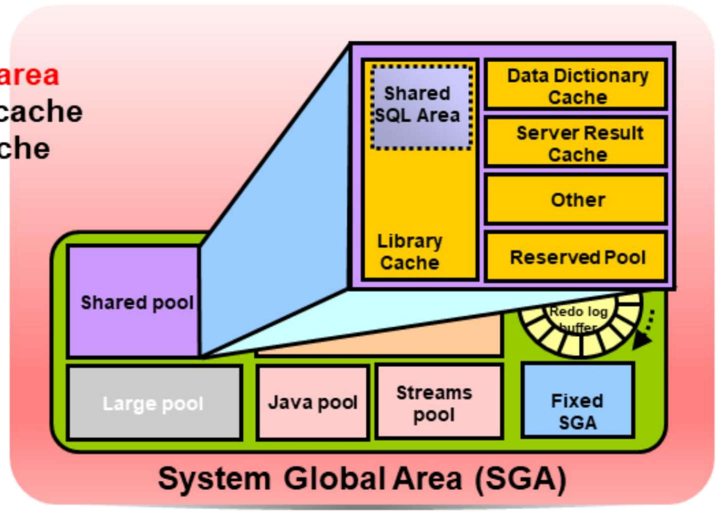
- **Control files:** Contain data about the database itself (that is, physical database structure information). These files are critical to the database. Without them, you cannot open data files to access the data within the database.
- **Data files:** Contain the user or application data of the database
- **Online redo log files:** Allow for instance recovery of the database. If the database crashes and does not lose any data files, then the instance can recover the database with the information in these files.

The following additional files are important to the successful running of the database:

- **Parameter file:** Is used to define how the instance is configured when it starts up
- **Password file:** Allows users to connect remotely to the database and perform administrative tasks
- **Backup files:** Are used for database recovery. You typically restore a backup file when a media failure or user error has damaged or deleted the original file.
- **Archive log files:** Contain an ongoing history of the data changes (redo) that are generated by the instance. Using these files and a backup of the database, you can recover a lost data file. That is, archive logs enable the recovery of restored data files.

Shared Pool

- Is a portion of the SGA
- Contains:
  - Library cache
    - Shared SQL area
  - Data dictionary cache
  - Server result cache



3-18

The shared pool portion of the SGA contains the library cache, the data dictionary cache, the server result cache containing the SQL query result cache and the PL/SQL function result cache, buffers for parallel execution messages, and control structures.

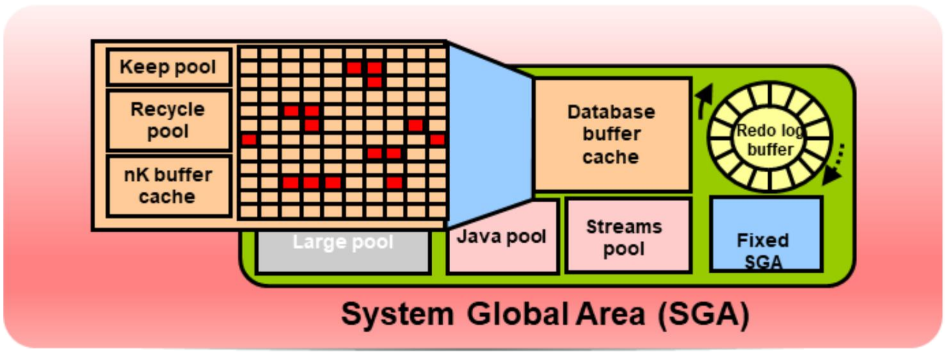
The *data dictionary* is a collection of database tables and views containing reference information about the database, its structures, and its users. Oracle Database accesses the data dictionary frequently during SQL statement parsing. This access is essential to the continuing operation of Oracle Database.

The data dictionary is accessed so often by Oracle Database that two special locations in memory are designated to hold dictionary data. One area is called the *data dictionary cache*, also known as the row cache because it holds data as rows instead of buffers (buffers hold entire blocks of data). The other area in memory that holds dictionary data is the *library cache*. All Oracle Database user processes share these two caches for access to data dictionary information.

Oracle Database represents each SQL statement that it runs with a shared SQL area (as well as a private SQL area kept in the PGA). Oracle Database recognizes when two users are executing the same SQL statement and reuses the shared SQL area for those users.

Database Buffer Cache

- Is part of the SGA
- Holds copies of data blocks that are read from data files
- Is shared by all concurrent users



3-19

The database buffer cache is the portion of the SGA that holds block images read from data files or constructed dynamically to satisfy the read consistency model. All users who are concurrently connected to the instance share access to the database buffer cache.

The first time an Oracle Database user process requires a particular piece of data, it searches for the data in the database buffer cache. If the process finds the data already in the cache (a cache hit), it can read the data directly from memory. If the process cannot find the data in the cache (a cache miss), it must copy the data block from a data file on disk into a buffer in the cache before accessing the data. Accessing data through a cache hit is faster than accessing data through a cache miss.

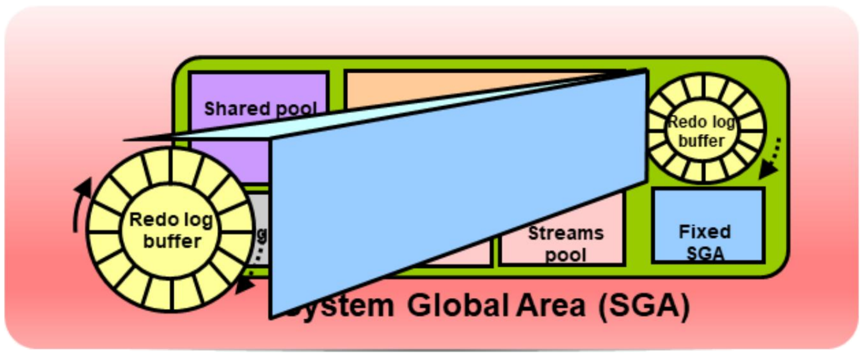
The buffers in the cache are managed by a complex algorithm that uses a combination of least recently used (LRU) lists and touch count. The LRU helps to ensure that the most recently used blocks tend to stay in memory to minimize disk access.

The keep buffer pool and the recycle buffer pool are used for specialized buffer pool tuning. The keep buffer pool is designed to retain buffers in memory longer than the LRU would normally retain them. The recycle buffer pool is designed to flush buffers from memory faster than the LRU normally would.

Additional buffer caches can be configured to hold blocks of a size that is different from the default block size.

Redo Log Buffer

- Is a circular buffer in the SGA
- Holds information about changes made to the database
- Contains redo entries that have the information to redo changes made by operations such as DML and DDL



3-20

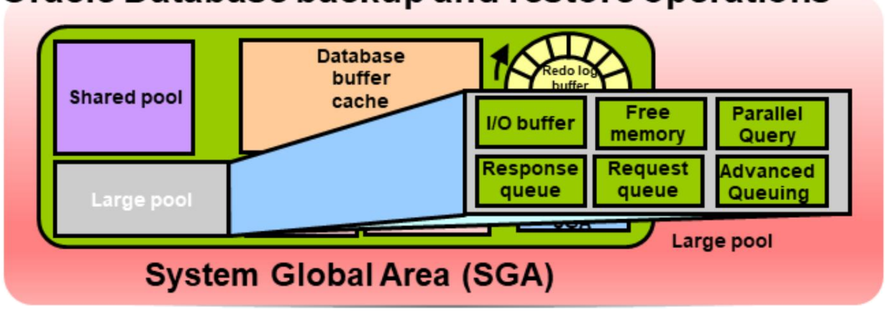
The redo log buffer is a circular buffer in the SGA that holds information about changes made to the database. This information is stored in redo entries. Redo entries contain the information necessary to reconstruct (or redo) changes that are made to the database by DML, DDL, or internal operations. Redo entries are used for database recovery, if necessary.

As the server process makes changes to the buffer cache, redo entries are generated and written to the redo log buffer in the SGA. The redo entries take up continuous, sequential space in the buffer. The log writer background process writes the redo log buffer to the active redo log file (or group of files) on disk.

Large Pool

Provides large memory allocations for:

- Session memory for the shared server and the Oracle XA interface
- I/O server processes
- Oracle Database backup and restore operations



The database administrator can configure an optional memory area called the *large pool* to provide large memory allocations for:

- Session memory for the shared server and the Oracle XA interface (used where transactions interact with multiple databases)
- I/O server processes
- Oracle Database backup and restore operations
- Parallel Query operations
- Advanced Queuing memory table storage

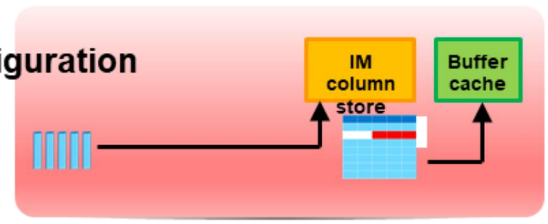
By allocating session memory from the large pool for shared server, Oracle XA, or parallel query buffers, Oracle Database can use the shared pool primarily for caching shared SQL and avoid the performance overhead that is caused by shrinking the shared SQL cache.

In addition, the memory for Oracle Database backup and restore operations, for I/O server processes and for parallel buffers, is allocated in buffers of a few hundred kilobytes. The large pool is better able to satisfy such large memory requests than the shared pool.

The large pool is not managed by a least recently used (LRU) list.

In-Memory Column Store: Introduction

- Instant query response:
  - Faster queries on very large tables on any columns (100x)
  - Use of scans, joins, and aggregates
  - Without indexes
  - Best suited for analytics: few columns, many rows
- Faster DML: Removal of most analytics indexes (3 to 4x)
- Full application transparency
- Easy setup:
  - In-memory column store configuration
  - Segment attributes



3-22

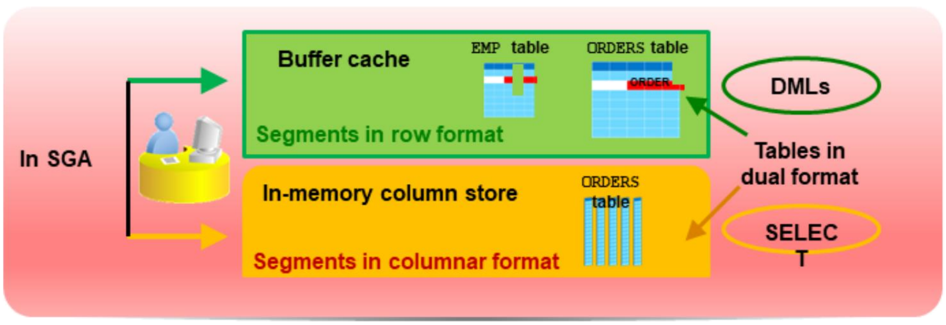
The In-Memory Column Store feature enables objects (tables, partitions, and other types) to be stored in memory in a new format known as the *columnar format*. This format enables scans, joins, and aggregates to perform much faster than the traditional on-disk format, thus providing fast reporting and DML performance for both OLTP and DW environments. This is particularly useful for analytic applications that operate on few columns returning many rows rather than for OLTP that operates on few rows returning many columns. The DBA must define the segments that are to be populated into the in-memory column store (IM column store), such as hot tables, partitions, and, more precisely, the more frequently accessed columns.

The in-memory columnar format does not replace the on-disk or buffer cache format. It is a consistent copy of a table or of some columns of a table converted to the new columnar format that is independent of the disk format and only available in memory. Because of this independence, applications are able to transparently use this option without any changes. For the data to be converted into the new columnar format, a new pool is requested in the SGA. The pool is the IM column store.

If sufficient space is allocated for the IM column store, a query that accesses objects that are candidates to be populated into the IM column store performs much faster. The improved performance allows ad hoc analytic queries to be executed directly on the real-time transaction data without impacting the existing workload.

In-Memory Column Store: Overview

- A pool in the SGA: In-Memory column store
  – Segments populated into the IM column store are converted into a columnar format.
  – In-Memory segments are transactionally consistent with the buffer cache.
- Only one segment on disk and in row format



3-23

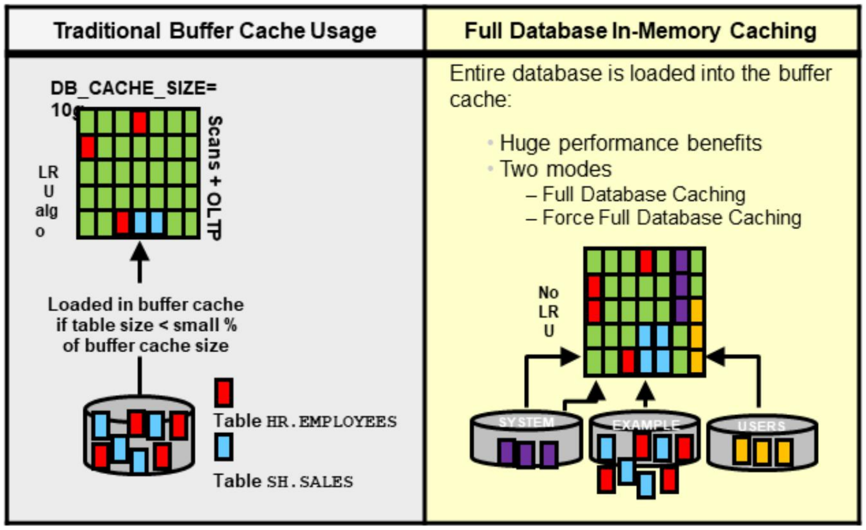
The in-memory columnar format does not replace the on-disk or buffer cache format. This means that when a segment, such as a table or a partition, is populated into the IM column store, the on-disk format segment is automatically converted into a columnar format and optionally compressed. The columnar format is a pure in-memory format. There is no columnar format storage on disk. It never causes additional writes to disk and therefore does not require any logging or undo space.

All data is stored on disk in the traditional row format.

Moreover, the columnar format of a segment is a transaction-consistent copy of the segment either on disk or in the buffer cache. Transaction consistency between the two pools is maintained.

If sufficient space is allocated to the IM column store in SGA, a query that accesses objects that are populated into the IM column store performs much faster. The improved performance allows more ad hoc analytic queries to be executed directly on real-time transaction data without impacting the existing workload. A lack of IM column store space does not prevent statements from executing against tables that could have been populated into the IM column store.

Full Database In-Memory Caching



3-24

The current algorithm for table scans loads a table into the buffer cache only when the table size is less than a small percent of the buffer cache size. For very large tables, the database uses a direct path read, which loads blocks directly into the PGA and bypasses the SGA, to avoid flooding the buffer cache. The DBA must explicitly declare small lookup tables, which are accessed frequently, as `CACHE` to load data into memory and avoid bypassing the SGA. This clause indicates that the blocks retrieved for these tables are placed at the most recently used end of the least recently used (LRU) list in the buffer cache when a full table scan is performed.

The Full Database In-memory Caching feature enables an entire database to be cached in memory when the database size (sum of all data files, `SYSTEM` tablespace, LOB CACHE files minus `SYSAUX`, `TEMP`) is smaller than the buffer cache size. Caching and running a database from memory leads to huge performance benefits. Two modes can be used:

- **Full Database Caching:** Implicit default and automatic mode in which an internal calculation determines if the database can be fully cached for an instance. `NOCACHE` LOBs are not cached in Full Database Caching. But in Force Full Database Caching mode, even `NOCACHE` LOBs are cached.
- **Force Full Database Caching:** Neither Full Database Caching nor Force Full Database Caching forces or prefetches data into memory. Workload must access the data first for them to be cached. It considers the entire database as eligible to be completely cached in the buffer cache. This mode requires the DBA to execute the `ALTER DATABASE FORCE`

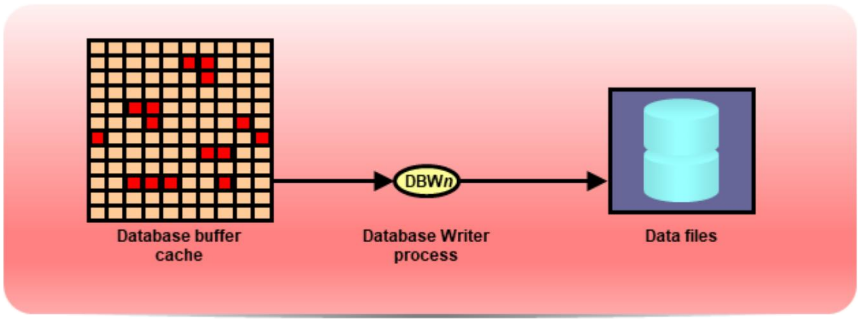
`FULL DATABASE CACHING` command. This mode takes precedence over Full Database Caching mode. To revert to traditional caching, use the `ALTER DATABASE NO FORCE FULL DATABASE CACHING` command.

Refer to *Oracle Database Administrators Guide* and *Oracle Database Performance Tuning Guide* for detailed information about this feature.

Database Writer Process (DBWn)

Writes modified (dirty) buffers in the database buffer cache to disk:

- Asynchronously while performing other processing
- To advance the checkpoint



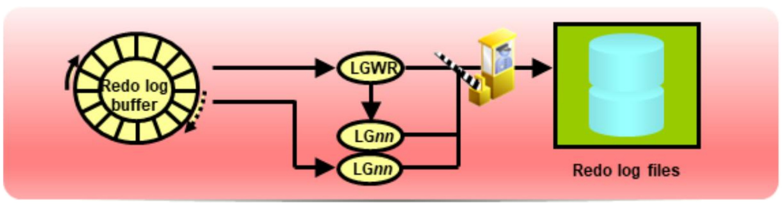
3-25

The Database Writer process (DBW*n*) writes the contents of buffers to data files. The DBW*n* processes are responsible for writing modified (dirty) buffers in the database buffer cache to disk. Although one Database Writer process (DBW*0*) is adequate for most systems, you can configure additional processes to improve write performance if your system modifies data heavily. The additional processes are named DBW1 through DBW9, DBWa through DBWz, and BW36-BW99. These additional DBW*n* processes are not useful on uniprocessor systems.

When a buffer in the database buffer cache is modified, it is marked dirty and added to the head of the checkpoint queue that is kept in system change number (SCN) order. This order, therefore, matches the order of redo that is written to the redo logs for these changed buffers. When the number of available buffers in the buffer cache falls below an internal threshold (to the extent that server processes find it difficult to obtain available buffers), DBW*n* writes nonfrequently used buffers to the data files from the tail of the LRU list so that processes can replace buffers when they need them. DBW*n* also writes from the tail of the checkpoint queue to keep the checkpoint advancing.

Log Writer Process (LGWR)

- Writes the redo log buffer to a redo log file on disk:
  - When a user process commits a transaction
  - When an online redo log switch occurs
  - When the redo log buffer is one-third full or contains 1 MB of buffered data
  - Before a DBW*n* process writes modified buffers to disk
  - When three seconds have passed since the last write
- Serves as coordinator of LG*nn* processes and ensures correct order for operations that must be ordered

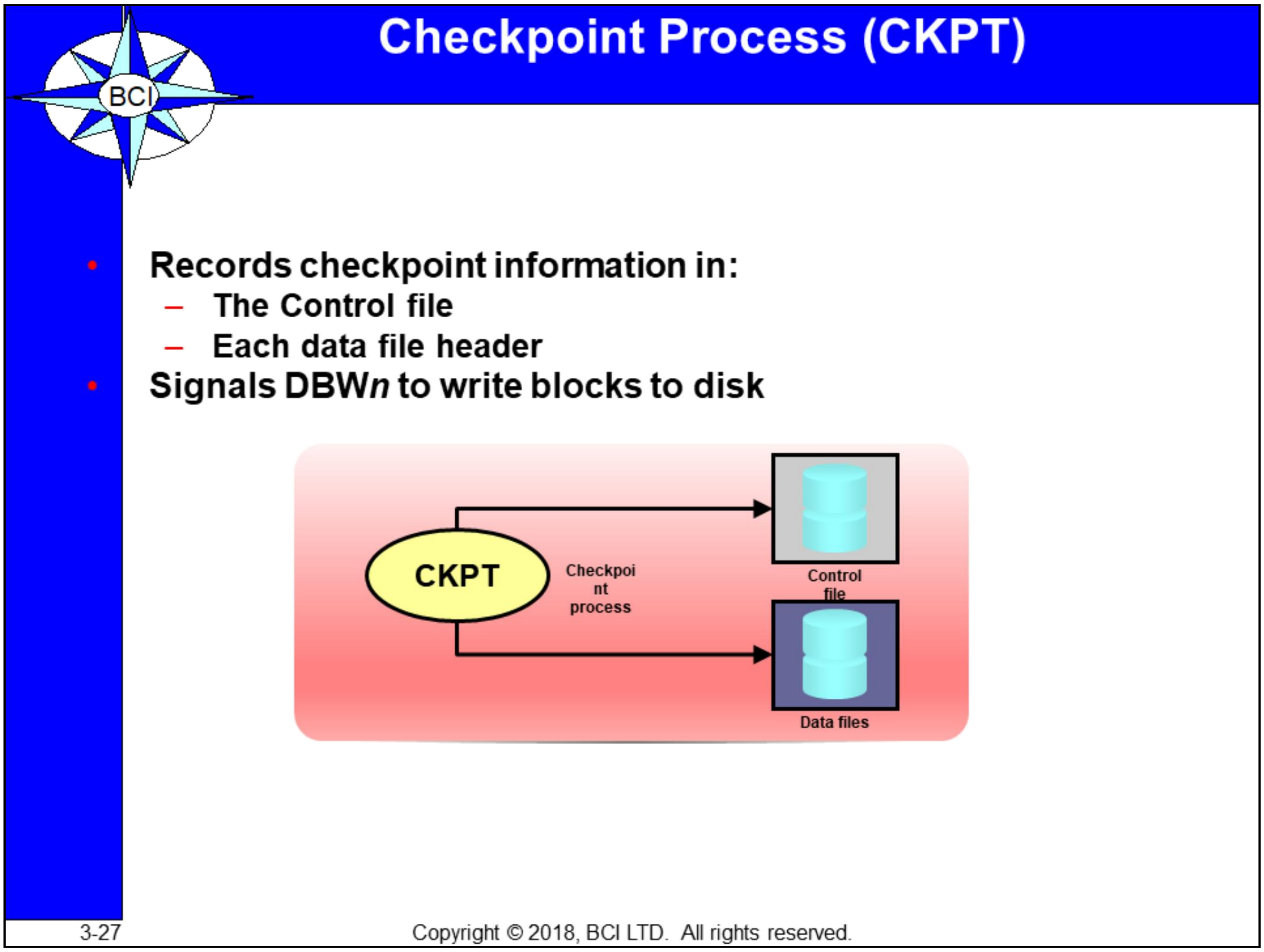


The Log Writer process (LGWR) is responsible for redo log buffer management by writing the redo log buffer entries to a redo log file on disk. LGWR writes all redo entries that have been copied into the buffer since the last time it wrote.

LGWR starts and coordinates multiple helper processes that concurrently perform some of the work. LGWR handles the operations that are very fast or must be coordinated and delegates operations to the LG*nn* that could benefit from concurrent operations, primarily writing the redo from the log buffer to the redo log file and posting the completed write to the foreground process that is waiting.

Because LG*nn* processes work concurrently and certain operations must be performed in order, LGWR forces ordering so that even if the writes complete out of order, the posting to the foreground processes will be in the correct order.

The redo log buffer is a circular buffer. When LGWR writes redo entries from the redo log buffer to a redo log file, server processes can then copy new entries over the entries in the redo log buffer that have been written to disk. LGWR normally writes fast enough to ensure that space is always available in the buffer for new entries, even when access to the redo log is heavy. LGWR writes one contiguous portion of the buffer to disk.

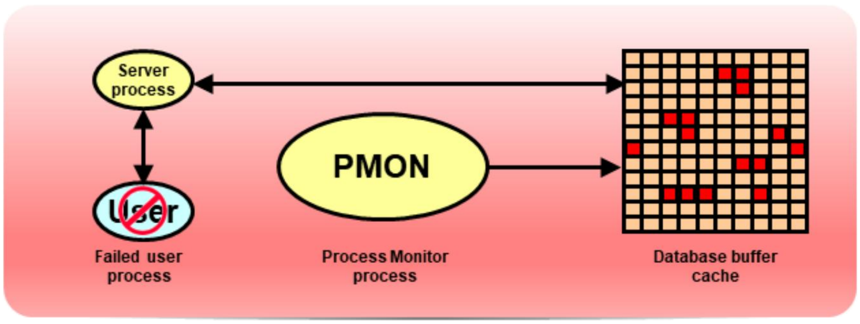


A *checkpoint* is a data structure that defines a system change number (SCN) in the redo thread of a database. Checkpoints are recorded in the control file and in each data file header. They are a crucial element of recovery.

When a checkpoint occurs, Oracle Database must update the headers of all data files to record the details of the checkpoint. This is done by the CKPT process. The CKPT process does not write blocks to disk; DBW*n* always performs that work. The SCNs recorded in the file headers guarantee that all changes made to database blocks before that SCN have been written to disk.

Process Monitor Process (PMON)

- Performs process recovery when a user process fails
  - Cleans up the database buffer cache
  - Frees resources that are used by the user process
- Monitors sessions for idle session timeout



3-28

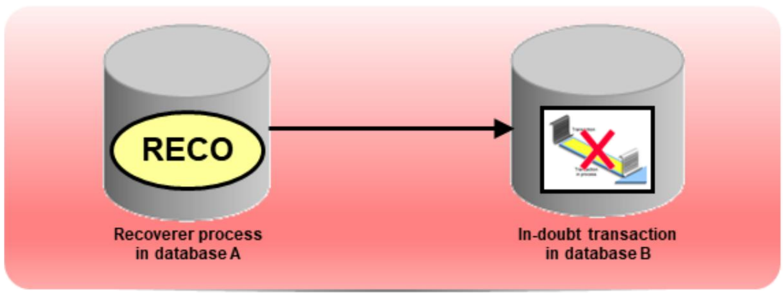
The Process Monitor process (PMON) performs process recovery when a user process fails. PMON is responsible for cleaning up the database buffer cache and freeing resources that the user process was using. For example, it resets the status of the active transaction table, releases locks, and removes the process ID from the list of active processes.

PMON periodically checks the status of dispatcher and server processes and restarts any that have stopped running (but not any that Oracle Database has terminated intentionally).

Like SMON, PMON checks regularly to see whether it is needed; it can be called if another process detects the need for it.

Recoverer Process (RECO)

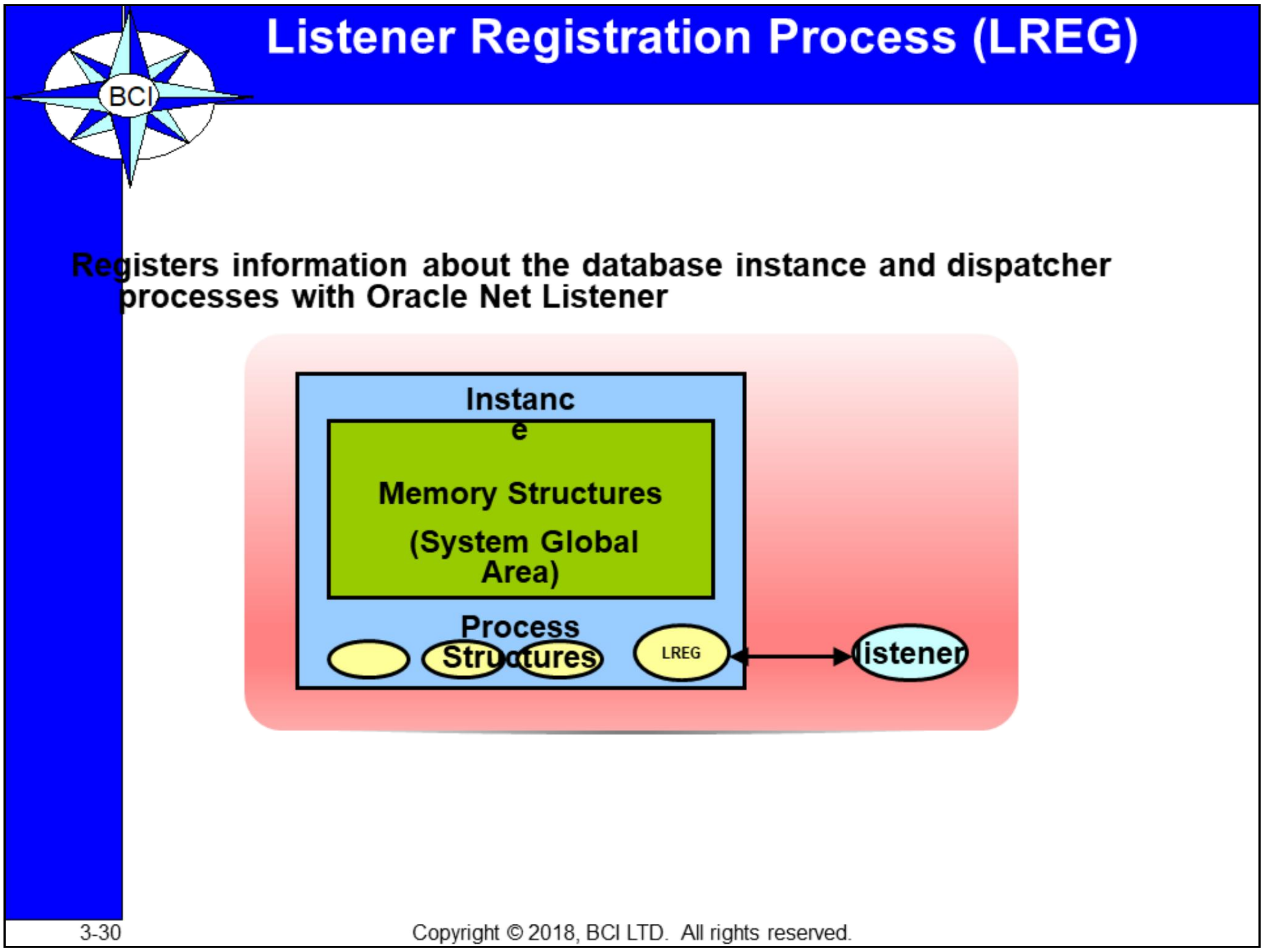
- Used with the distributed database configuration
- Automatically connects to other databases involved in in-doubt distributed transactions
- Automatically resolves all in-doubt transactions
- Removes any rows that correspond to in-doubt transactions



3-29

The Recoverer process (RECO) is a background process used with the distributed database configuration that automatically resolves failures involving distributed transactions. The RECO process of an instance automatically connects to other databases involved in an in-doubt distributed transaction. When the RECO process reestablishes a connection between involved database servers, it automatically resolves all in-doubt transactions, removing from each database's pending transaction table any rows that correspond to the resolved in-doubt transactions.

If the RECO process fails to connect with a remote server, RECO automatically tries to connect again after a timed interval. However, RECO waits an increasing amount of time (growing exponentially) before it attempts another connection.



The Listener Registration process (LREG) registers information about the database instance and dispatcher processes with the Oracle Net Listener. LREG provides the listener with the following information:

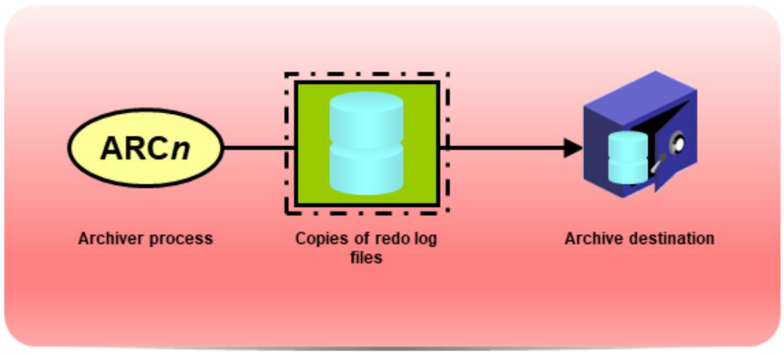
- Names of the database services
- Name of the database instance associated with the services, and its current and maximum load
- Service handlers (dispatchers and dedicated servers) available for the instance, including their type, protocol addresses, and current and maximum load

When the instance starts, LREG attempts to connect to the listener. If the listener is running, LREG passes information to it. If the listener is not running, LREG periodically attempts to connect to it. It may take up to 60 seconds for LREG to register the database instance with the listener after the listener has started.

You can use the `ALTER SYSTEM REGISTER` command to immediately initiate service registration after starting the listener.

Archiver Processes (ARC*n*)

- Copy redo log files to a designated storage device after a log switch has occurred
- Can collect transaction redo data and transmit that data to standby destinations



The Archiver processes (ARC*n*) copy redo log files to a designated storage device after a log switch has occurred. ARC*n* processes are present only when the database is in `ARCHIVELOG` mode and automatic archiving is enabled.

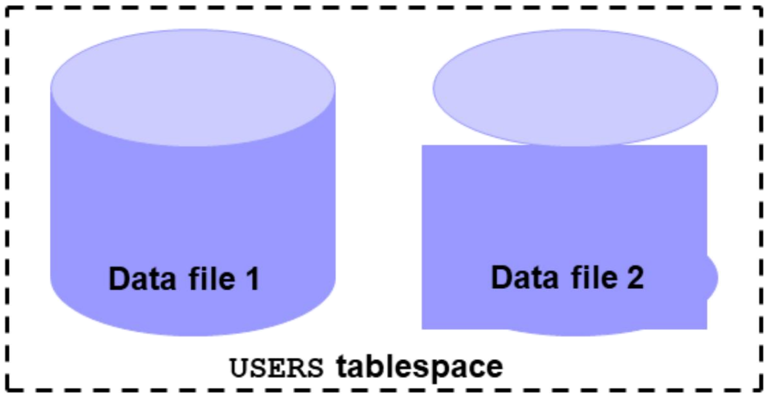
If you anticipate a heavy workload for archiving (such as during bulk loading of data), you can increase the maximum number of Archiver processes. There can also be multiple archive log destinations. It is recommended that there be at least one Archiver process for each destination. The default is to have four Archiver processes.

Physical Database Structure (continued)

- **Trace files:** Each server and background process can write to an associated trace file. When an internal error is detected by a process, the process dumps information about the error to its trace file. Some of the information written to a trace file is intended for the database administrator, whereas other information is for Oracle Support Services.
- **Alert log files:** These are special trace files. They are also known as alert logs. The alert log of a database is a chronological log of messages and errors. Oracle recommends that you review these files.

Tablespaces and Data Files

- Tablespaces consist of one or more data files.
- Data files belong to only one tablespace.



Sql> desc dba_data_files

3-33

Tablespaces and Data Files

A database is divided into logical storage units called tablespaces, which can be used to group related logical structures together. Each database is logically divided into one or more tablespaces. One or more data files are explicitly created for each tablespace to physically store the data of all logical structures in a tablespace.

Note: You can also create the bigfile tablespaces, which are tablespaces with a single but very large (up to 4 billion data blocks or 8 exabytes) data file. The traditional smallfile tablespaces (which are the default) can contain multiple data files, but the files cannot be as large – up to 32 gb. For more information about the bigfile tablespaces, see the *Database Administrator's Guide*.

SYSTEM and SYSAUX Tablespaces

- The SYSTEM and SYSAUX tablespaces are mandatory tablespaces.
- They are created at the time of database creation.
- They must be online.
- The SYSTEM tablespace is used for core functionality (for example, data dictionary tables).
- The auxiliary SYSAUX tablespace is used for additional database components
- MGMT_ECM_DEPOT_TS, MGMT_TABLESPACE, and MGMT_AD$J_TS are used when Enterprise Manager is created.

Sql> select tablespace_name from dba_tablespaces

3-34

SYSTEM and SYSAUX Tablespaces

Each Oracle database contains a `SYSTEM` tablespace and a `SYSAUX` tablespace. They are automatically created when the database is created. The system default is to create a smallfile tablespace. You can also create bigfile tablespaces, which enable the Oracle database to manage ultra large files (up to 8 exabytes).

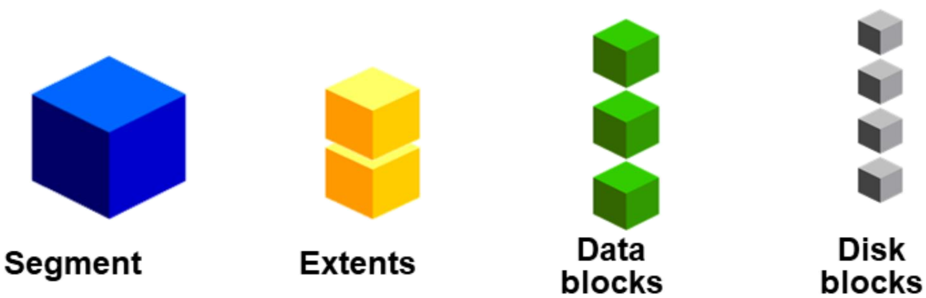
A tablespace can be online (accessible) or offline (not accessible). The `SYSTEM` tablespace is always online when the database is open. It stores tables that support the core functionality of the database, such as the data dictionary tables.

The `SYSAUX` tablespace is an auxiliary tablespace to the `SYSTEM` tablespace. The `SYSAUX` tablespace stores many database components, and it must be online for the correct functioning of all database components.

Segments, Extents, and Blocks

- Segments exist within a tablespace.
- Segments are made up of a collection of extents.
- Extents are a collection of data blocks.
- Data blocks are mapped to disk blocks.

```
Sql> desc dba_segments
Sql> desc dba_extents
```



3-35

Segments, Extents, and Blocks

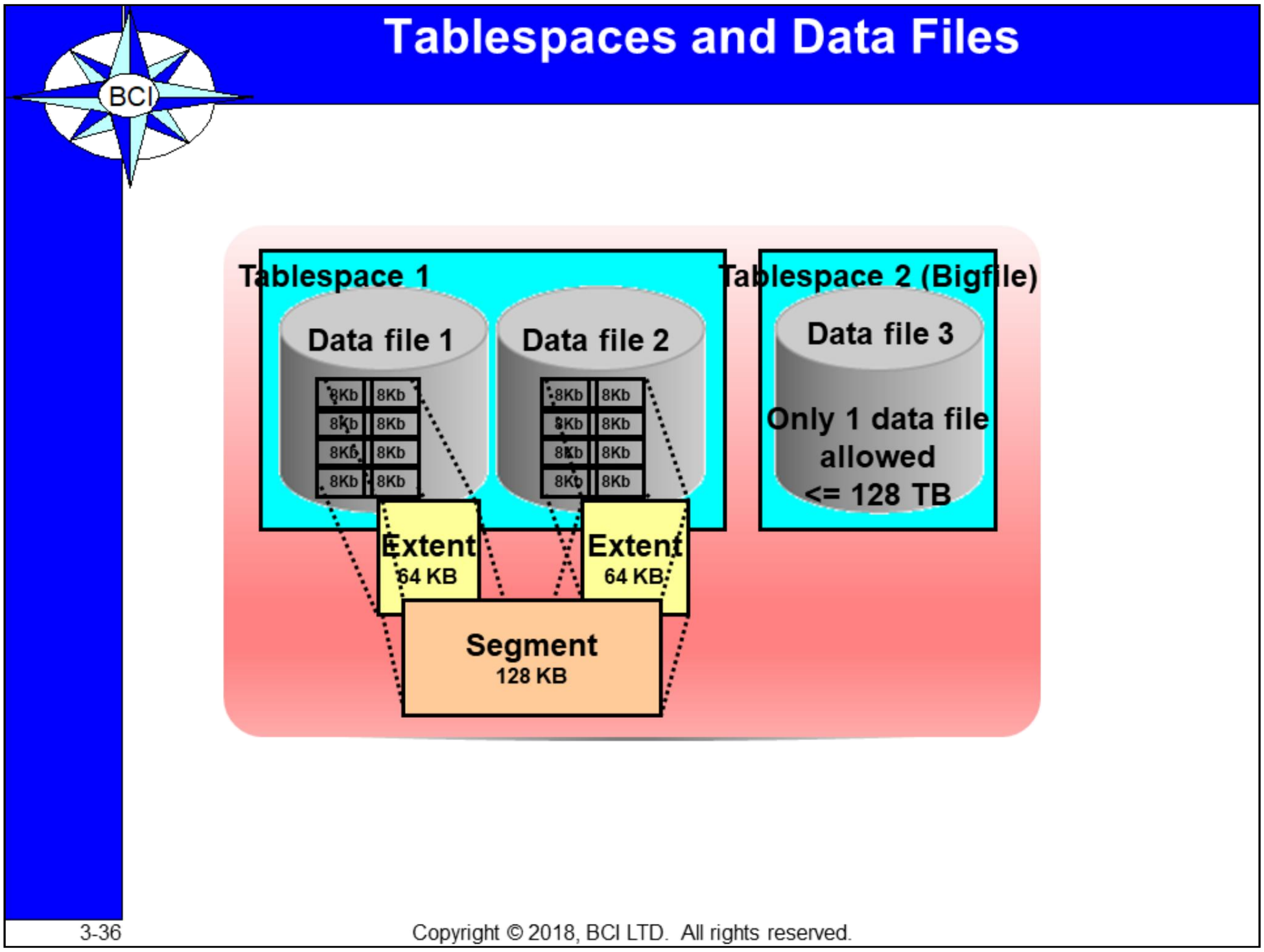
Database objects, such as tables and indexes, are stored as segments in tablespaces. Each segment contains one or more extents. An extent consists of contiguous data blocks, which means that each extent can exist only in one data file. Data blocks are the smallest unit of I/O in the database.

When the database requests a set of data blocks from the operating system (OS), the OS maps this to an actual file system or disk block on the storage device. Because of this, you need not know the physical address of any of the data in your database. This also means that a data file can be striped or mirrored on several disks.

The size of the data block can be set at the time of the creation of the database. The default size of 8 KB is adequate for most databases. If your database supports a data warehouse application that has large tables and indexes, then a larger block size may be beneficial.

If your database supports a transactional application where reads and writes are random, then specifying a smaller block size may be beneficial. The maximum block size depends on your OS. The minimum Oracle block size is 2 KB and should rarely (if ever) be used.

You can have tablespaces with different block sizes. However, this should be used only for transportable tablespaces. For details, see the *Database Administrator's Guide*.



A database is divided into *tablespaces*, which are logical storage units that can be used to group related logical structures. One or more data files are explicitly created for each tablespace to physically store the data of all logical structures in a tablespace.

The graphic in the slide illustrates tablespace 1, composed of two data files. A segment of 128 KB size, composed of two extents, is spanning the two data files. The first extent of size 64 KB is in the first data file and the second extent, also of size 64 KB, is in the second data file. Both extents are formed from contiguous 8 KB Oracle blocks.

Note: You can also create bigfile tablespaces, which have only one file that is often very large. The file may be of any size up to the maximum that the row ID architecture permits. The maximum size of the single data file or temp file is 128 terabytes (TB) for a tablespace with 32 K blocks and 32 TB for a tablespace with 8 K blocks.

Traditional smallfile tablespaces (which are the default) may contain multiple data files, but the files cannot be as large. For more information about bigfile tablespaces, see *Oracle Database Administrator's Guide*.

SYSTEM and UNDOTBS Tablespaces

- The SYSTEM and UNDOTBS tablespaces are mandatory tablespaces that are created at the time of database creation. They must be online.
- The SYSTEM tablespace is used for core functionality (for example, data dictionary tables).
- The UNDOTBS tablespace is used for additional database components.
- The SYSTEM and SYSAUX tablespaces should not be used for application data.

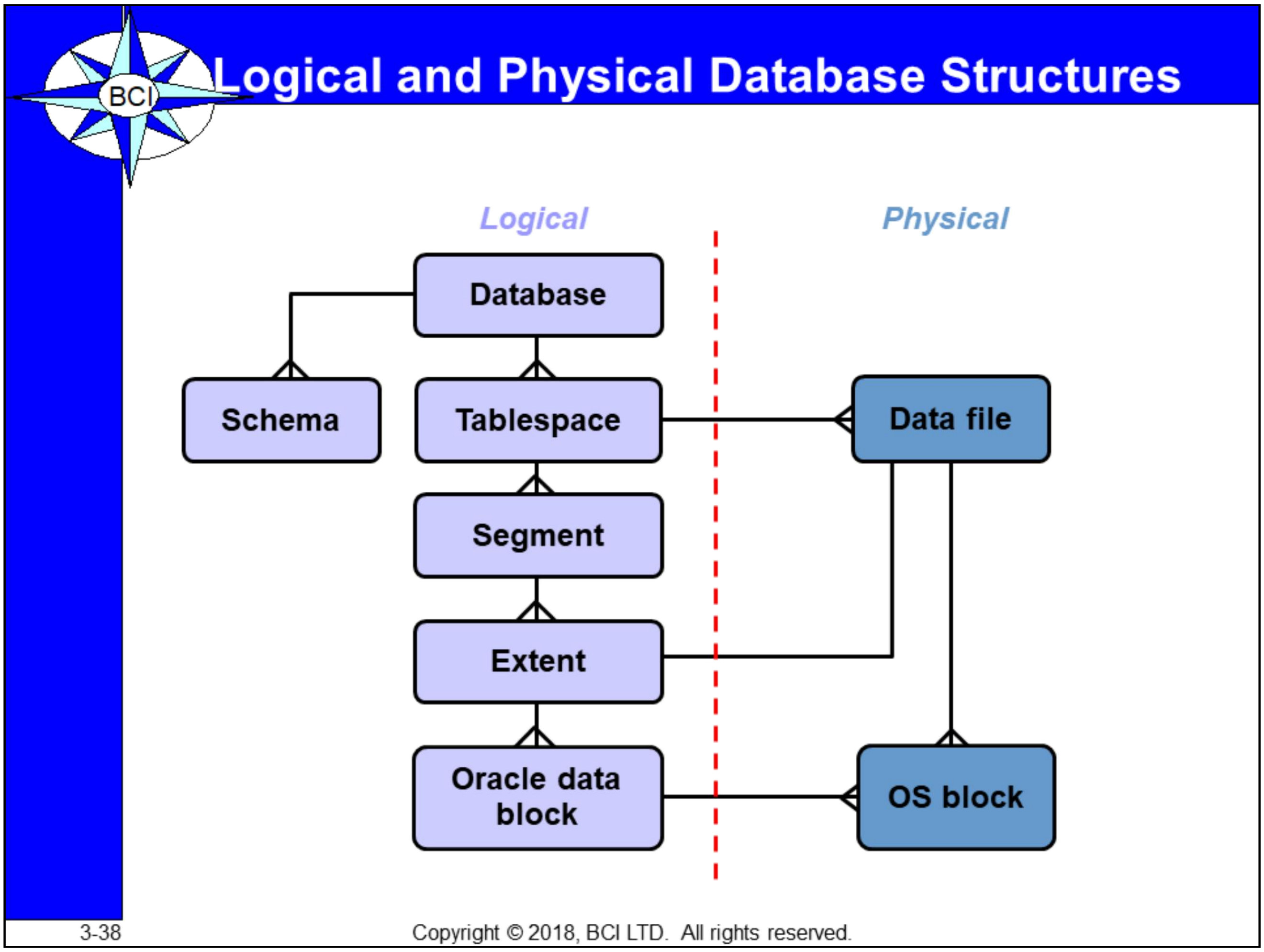
3-37

Each Oracle database must contain a SYSTEM tablespace and a SYSAUX tablespace. They are automatically created when the database is created. The system default is to create a smallfile tablespace. You can also create bigfile tablespaces, which enable the Oracle database to manage ultralarge files.

A tablespace can be online (accessible) or offline (not accessible). The SYSTEM tablespace is always online when the database is open. It stores tables that support the core functionality of the database, such as the data dictionary tables.

The SYSAUX tablespace is an auxiliary tablespace to the SYSTEM tablespace. The SYSAUX tablespace stores many database components and must be online for the correct functioning of all database components. The SYSTEM and SYSAUX tablespaces are not recommended for storing an application's data. Additional tablespaces can be created for this purpose.

Note: The SYSAUX tablespace may be taken offline to perform tablespace recovery, whereas this is not possible for the SYSTEM tablespace. Neither of them may be made read-only.



Logical and Physical Database Structures

An Oracle database is a collection of data that is treated as a unit. The general purpose of a database is to store and retrieve related information. The database has logical structures and physical structures.

Tablespaces

A database is divided into logical storage units called tablespaces, which group related logical structures together. For example, tablespaces commonly group all of an application's objects to simplify some administrative operations. You may have a tablespace for application data and an additional one for application indexes.

Databases, Tablespaces, and Data Files

The relationship among databases, tablespaces, and data files is illustrated in the slide. Each database is logically divided into one or more tablespaces. One or more data files are explicitly created for each tablespace to physically store the data of all logical structures in a tablespace. If it is a `TEMPORARY` tablespace, instead of a data file, then the tablespace has a temporary file.

Logical and Physical Database Structures (continued)

Schemas

A schema is a collection of database objects that are owned by a database user. Schema objects are the logical structures that directly refer to the database's data. Schema objects include such structures as tables, views, sequences, stored procedures, synonyms, indexes, clusters, and database links. In general, schema objects include everything that your application creates in the database.

Data Blocks

At the finest level of granularity, an Oracle database's data is stored in data blocks. One data block corresponds to a specific number of bytes of physical database space on the disk. A data block size is specified for each tablespace when it is created. A database uses and allocates free database space in Oracle data blocks.

Extents

The next level of logical database space is called an extent. An extent is a specific number of contiguous data blocks (obtained in a single allocation) that are used to store a specific type of information.

Segments

The level of logical database storage above an extent is called a segment. A segment is a set of extents allocated for a certain logical structure. For example, the different types of segments include:

- **Data segments:** Each nonclustered, non-indexed-organized table has a data segment. All of the table's data is stored in the extents of its data segment. For a partitioned table, each partition has a data segment. Each cluster has a data segment. The data of every table in the cluster is stored in the cluster's data segment.
- **Index segments:** Each index has an index segment that stores all of its data. For a partitioned index, each partition has an index segment.
- **Undo segments:** One `UNDO` tablespace is created by the database administrator to temporarily store *undo* information. The information in an undo segment is used to generate read-consistent database information and, during database recovery, to roll back uncommitted transactions for users.
- **Temporary segments:** Temporary segments are created by the Oracle database when a SQL statement needs a temporary work area to complete execution. When the statement finishes execution, the temporary segment's extents are returned to the instance for future use. Specify a default temporary tablespace for every user or a default temporary tablespace, which is used database wide.

The Oracle database dynamically allocates space. When the existing extents of a segment are full, additional extents are added. Because extents are allocated as needed, the extents of a segment may or may not be contiguous on the disk.

Database Architecture: Summary of Structural Components

- **Memory structures:**
  - System Global Area (SGA): Database buffer cache, redo buffer, and various pools
  - Program Global Area (PGA)
- **Process structures:**
  - User process and Server process
  - Background processes: SMON, PMON, DBW*n*, CKPT, LGWR, ARC*n*, and so on
- **Storage structures:**
  - Logical: Database, schema, tablespace, segment, extent, and Oracle block
  - Physical: Files for data, parameters, redo, and OS block

3-40

Database Architecture: Summary of Structural Components

In this lesson, you learned at a high level about the structural components of the Oracle database: memory, process, and storage structures. More details are covered in the following lessons.

Relational Database Architecture

3-41

Terminal Learning Objective

Action: Maintain an Oracle Database.

Condition: Given a student handout and the Oracle DBA Handbook.

***Standard: Students must pass a multiple choice written examination with a minimum score of 70% and successfully create an Oracle Storage Structure during a scenario-based performance evaluation.**

***See Student Evaluation Plan (SEP) for details.**