

Database System Architecture & System Catalog

Instructor: Mourad Benchikh

Text Books:

Elmasri & Navathe Chap. 17

Silberschatz & Korth Chap. 1

Oracle9i Documentation

First-Semester 1427-1428

Definitions

- DB (Database): collection of related data.
- DBMS (Database Management System):
 - A collection of programs that enables users to create and maintain a database.
 - Commercial systems: Oracle, MS SQL-server, IBM/DB2, Sybase, Informix, MS-Access, Ingres, etc.
 - Among existing DBMS classifications the one based on the model supported
 - Traditional: relational DBMS (RDBMS), hierarchic DBMS and network DBMS
 - Emerging: Object-Oriented DBMS (OODBMS) and Object-Relational DBMS (ORDBMS)
 - Oracle9i supports the Object-Relational model to store complex business models in a relational DB. It enables users to define and to use object types, different from native SQL datatypes, specifying both structure of the data and the methods of operating on the data.
- DBS (Database System)
 - The DBMS system software with the database itself. Sometimes, the applications are also included

Purpose of a DS

- In the past, DB applications were built on top of file systems
- Some drawbacks: file systems (FS) use vs. DBMS use:
 - Self-describing nature.
 - FS: redundancy in defining and storing data by different users' applications. Redundancy leads to duplicate effort, waste of storage space, and inconsistency.
 - DBMS: A single data repository (i.e. DB) is used. So, redundancy is reduced.
 - FS: Data definition part of the application pgm : only the DB declared in the pgm could be used.
 - DBMS: DB structure defined in the DBMS catalog. The DBMS can access divers DBs defined in the catalog.
 - Storage structures for efficient query processing
 - FS: the DB is stored in disk files and accessed according to access methods (sequential, direct, etc.) provided by the FS and chosen by the application programmers.
 - DBMS: the *query processing and optimization subsystem* is responsible for choosing an efficient query execution plan for each query based on the existing storage structure. Indexes are among storage structures used to speed up disk search. DBMS also provides *buffering module* to maintains part of the DB in main memory buffers.

Purpose of a DS

–Program-data independence and handling data integrity constraints

- FS: data file structure (and constraints) embedded in the application pgms: changes in the file structure -add a new field- (or add new constraint (Balance>0)) implies changes in the pgms.
- DBMS: data file structure (and constraint) stored in the catalog independently from the access pgms: change in file structure (or constraints) does necessitates access pgms changes. DBMS provides capability for defining and enforcing these constraints.

–Restricting unauthorized access and support of multiple views of data

- FS: all users could see all the data.
- DBMS : *security and authorization subsystem* controls DBMS accesses by allowing only authorized access. Also, the DBMS may allow different users to see different "views" of the DB, according to the perspective each one requires.

–Data sharing and multi-user transaction processing

- FS: inconsistency could result due to uncontrolled concurrent multiuser access.
 - Several reservation clerks try to book the unique remaining seat on an airline flight
- DBMS: *concurrency control subsystem* ensures that multiuser access the DB in a correct way using the concept of transaction (i.e. isolation property).

–Providing backup and recovery

- FS: data could be lost in case of hardware or software failure.
- DBMS: the *backup and recovery subsystem* is used to deal with failure. Example, if a failure happens in the middle of an update transaction, this subsystem ensure that the DB is restored to the state it was in before the transaction started executing (i.e. atomicity property) based on log file.

Main DS actors

–Database Administrator (DBA)

- All the activities around the DS are coordinated by the DBA: deciding for the DB storage structure, describing recovery & backup strategy, authorizing DB access, etc.

–Database designers

- Responsible for identifying the data to be stored in the DB (from requirements analysis) and choosing the appropriate structures to represent and to store this data.

–End users

- Are people whose the DB primarily exists for their use and whose jobs require access to the DB for querying, updating, and generating reports.

–Application programmers

- Implement applications on top of the DBMS to provide convenient access to end users.

When do not use DBMS

– Overhead costs of using DBMS

–Initial investment in hardware, software, and training

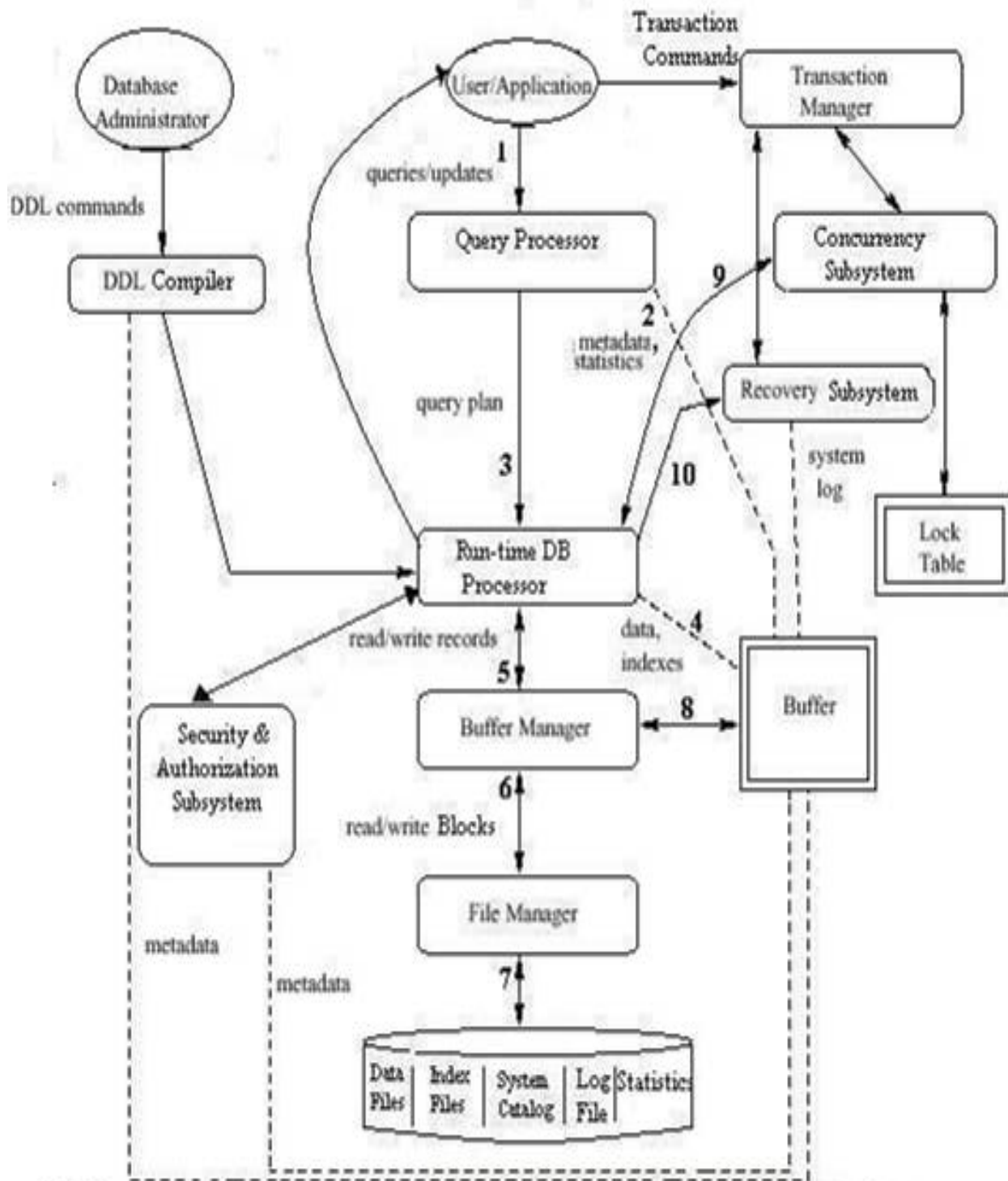
–In these circumstances, it may be desirable to use FS rather than DBMS

–DB and applications simples, well defined, not expected to change

–Stringent real-time pgm requirements may be not met due to DBMS overhead.

–Multiple-user access to data not required.

DBMS Component Modules



Solid lines ctrl & data flow-----Dashed lines: data flow-----Double boxes:in memory data
Single boxes: DBMS components

DBMS's Components

Note that many details are missed from this figure: Buffer manager call the concurrency subsystem, subsystem, etc.

DBMS Component Modules cont'd

- Query processor:
 - handles high-level queries. It parses, validates, optimizes, and compiles or interprets a query which results in the query plan.
- Run-time DB processor:
 - handles database accesses at run-time by receiving retrieval or update operations and carries them out on the database.
- DDL compiler:
 - processes schema definitions (DDL statements) and stores descriptions of the schemas (metadata) on the DBMS catalog.
- Transaction Manager:
 - With the cooperation of the concurrency and recovery subsystems, this module ensures that the transactions -which are constituted of queries and other actions- are executed atomically, in consistency, in isolation and in durability.
- Concurrency subsystem:
 - assures that individual actions of multiple transactions are executed in such an order that the result is the same as if the transactions had executed entirely at one time.

DBMS Component Modules cont'd

- Recovery subsystem:
 - responsible to store every change to the database separately on disk (I.e. log file) . Such information will be used to restore the system to a consistent state after a any failure has occurred.
- Security & authorization subsystem:
 - responsible to protect the database against unauthorized accesses.
- File manager:
 - File manager controls accesses to DBMS information that is stored on the disk and can use OS's File manager services.
- Buffer manager:
 - manages the main memory buffers which store data, metadata, indexes, statistics, and the log. Recall that all this information must be in main memory (i.e. in buffers) before it can be used.
- Note: Buffer Manager & File Manager together are often called Stored Data Manager.

DBMS Component Modules cont'd

- Examples of DBMS modules interactions
 - Suppose a user requests to update a record of a relation (see Figure).
 - the DML statement is issued to the query processor (1) which parses it, checks if the user has the privileges to do such operation using the metadata provided from the catalog information found in the main memory buffers (2) and finally issues the query plan which is the optimized way to execute the statement (3).
 - the run-time DB processor takes the query plan and executes it by writing the update in the buffer if the record is there (4). Otherwise, it requests the buffer manager to get the record (5). The buffer Manager requests the corresponding block from the File manager (6) who gets it from the disk (7), gives it back to the buffer manager who puts it in the buffer (8). The run-time DB processor can then execute the statement.
 - Before executing the statement, the run-time DB processors informs the concurrency subsystem (9) in order to check concurrent accesses for no inconsistency and the recovery subsystem to take the necessary actions to allow a recovery in case of failure (10)

System Architectures for DBMS

- We distinguish between two different DBMS architectures:
 - Centralized DBMS architecture –for earlier systems-
 - Client-Server DBMS architectures –for current systems-
- DBMS architecture followed the trends similar to those of general computer systems architectures.

Centralized DBMS Architecture

- Earlier computer system architectures were based on mainframe computers.
 - Mainframe computers provide the main processing of all system functions.
 - Most users accessed such systems via computer terminals that provide display capabilities with no processing power.
- Hence, in the centralized DBMS architecture, all functions of the system, including user application programs, user interface programs, as well as all the DBMS functionality are executed in the mainframe computer.

Client-Server Architecture

- Client-server computer systems architectures emerged with new computing environments.
 - Large number of PCs, workstations, specialized servers and another equipments are connected together via a network.
- A client-Server architecture contains:
 - Specialized servers with specific functionalities providing resources
 - File server, printer server, web server, E-mail server, etc.
 - Client machines provide users with appropriate interface to utilize these servers as well as with local processing power to run local applications

Two-Tier Client-Server DBMS Architecture

- Client-Server architecture is increasingly being incorporated into commercial DBMSs.
- Two-tier architecture distributes the components between the client and the server to split the burden of processing.
- Two most basic approaches to this two-tiers architecture
- First approach: used in RDBMS.
 - Query and transaction functionality remained at the server side. The server is often called *query server* or *transaction server*.
 - Since RDBMSs are based on the SQL languages, the servers are often called *SQL servers*.
 - The user interface programs and application programs moved to the client side.
 - Standards provide an Application Programming Interface (API) which allows client-side application programs to communicate with a database server.
 - Open Database Connectivity (ODBC) standard.
 - JDBC standard -for JAVA client programs-

Two-Tier Client-Server DBMS Architecture

–Basic API functions:

- Open a connection with a database, send queries and updates, and get back results.

–When a user program requests for a DBMS access, the client establishes a connection to the DBMS then send query and transaction requests using API calls, which are processed at the server sites and the query results are sent back to the client program.

–Most DBMS vendors provide ODBC and JDBC driver libraries for their systems, hence a client program can actually connect to several RDBMS –by simply linking the corresponding library with his program-.

•Second approach: taken by some OODBMS.

–Divide the software modules of the DBMS between client and server in a more integrated way –exact functionality division varies from system to system-. Example:

- Server side, called data server, includes these parts of the DBMS : data storage on disk, local concurrency control and recovery, buffering and caching of disk blocks, etc.
- Client side handles: user interface, data dictionary functions, global query optimization/concurrency control/recovery, etc.

Three-Tier Client-Server DBMS Architecture

- Three-tiers architecture is an extension of the client-server model. The tiers are usually as follows:
 - A client, used for the presentation of the application.
 - An application server (AS), used for the application's business logic processing.
 - A DB server, used for storage and retrieval of data.
- Therefore, the application's processing requirements are moved from the client to the application tier.
- Performances may improve by reducing the amount of traffic between the DB server and the clients since more of the interactions may be performed between the application server and the DB server.
- Application server could also check for security like client's credentials.
- A simplified client maintenance is achieved since application upgrades will reside only on the application server.

Catalog for Relational DBMS

- In general, a catalog is a “minidatabase” itself that stores special data often called metadata.
- This metadata contains:
 - Information on the schemas, or descriptions, of the database that the DBMS maintains such conceptual database schema, internal database schema, etc and the mapping between schemas.
 - Information needed by specific DBMS modules like the query optimization module, etc.
- Precisely, a relational DBMS catalog includes :
 - Relation names, attribute names, attribute domains (data types), constraint descriptions (primary key, etc.).
 - Description of views, storage structures and indexes.
 - Security and authorization information that describes each user’s privileges to access specific database relations and views and the owner of each relation.
 - Etc.

Catalog for Relational DBMS cont'd

- In a relational DBMS, the catalog is stored as relations which allows the DBMS software (as well as users when they are authorized to do so) to access and manipulate the catalog using a language such SQL.
- The choice of catalog relations and their schemas is not unique and vary from a DBMS to another.

Catalog for Relational DBMS cont'd

- Some examples of the catalog's relations:
 - REL_AND_ATTR_CATALOG: possible catalog structure for base relations which stores relation names, attribute names, attribute data types, and primary key information.
 - RELATION_INDEXES: possible catalog structure for indexes which stores relation names, index names, index attributes, index types, the order of attributes within indexes and the type of sort (asc/desc) in the index.
 - VIEW_QUERIES & VIEW_ATTRIBUTES: two possible catalog structures for views. The first to store the text of the query corresponding to the view and the second to store the names of the attributes of the view.

REL_AND_ATTR_CATALOG

REL_NAME	ATTR_NAME	ATTR_TYPE	MEMBER_OF_PK	MEMBER_OF_FK	FK_RELATION
----------	-----------	-----------	--------------	--------------	-------------

RELATION_INDEXES

REL_NAME	INDEX_NAME	MEMBER_ATTR	INDEX_TYPE	ATTR_NO	ASC_DESC
----------	------------	-------------	------------	---------	----------

VIEW_QUERIES

VIEW_NAME	QUERY
-----------	-------

VIEW_ATTRIBUTES

VIEW_NAME	ATTR_NAME	ATTR_NUM
-----------	-----------	----------

System Catalog Information in Oracle9i

- Oracle catalog is called Data Dictionary.
- Data dictionary contains information about *schema objects* such as tables, indexes, views, etc.
- Access to the data dictionary done through views divided into 3 categories:
 - **USER**: the views with prefix **USER** contain schema information for objects owned by a given user.
 - **USER_TABLES**: gives information on all base tables owned (created) by the user.
 - **ALL**: the views with prefix **ALL** contain information for objects owned by a user as well as objects that the user has been granted access to.
 - **ALL_TABLES**: gives information on all base tables to which the user has access.
 - The views with a prefix of **DBA** are for the database administrator and contain information about all database objects.
 - **DBA_TABLES**: displays the base tables information for the whole database.

System Catalog Information in Oracle9i

(cont'd)

- To see the data dictionary views available to you, query the view **DICTIONARY**.
 - Select `table_name`, `comments` from `dictionary`;
 - `COMMENTS` field gives a little definition of the view.
- Some examples:
 - **USER_ROLE_PRIVS**: This view lists roles granted to the user.
 - **ALL_VIEWS**: This view lists the text of views accessible to the user.
 - **DBA_INDEXES**: This view contains descriptions for all indexes in the database
 - **USER_OBJECTS** : This view lists objects (tables, indexes, etc.) owned by the user.
 - **USER_TAB_COLUMNS**: This view contains information about columns of user's tables, views, and clusters
 - **ALL_TAB_PRIVS**: This view lists the grants on objects for which the user or **PUBLIC** is the grantee
 - **DBA_SEGMENTS**: This view contains information about storage allocated for all database segments.

Catalog Information accessed by DBMS

Modules

- DBMS modules use and access the catalog very frequently.
- Query Processor Module
 - Parses queries and database update statements and checks the catalog to verify whether everything is valid.
 - For example, in relational system this module checks that all the relation names specified in the query exist in the catalog.
 - Converts the queries and update statements into low-level file access commands by accessing the catalog to know the mapping between the conceptual and the internal schemas – correspondence Relations-Files, etc-.
 - Access the catalog information (data statistics, etc.), to know the best way to execute the query.

Catalog Information accessed by DBMS

Modules cont'd

- DDL Compiler Module.
 - Parses & checks the specification of a database schema in the DDL statements and stores (populates) such description in the catalog???
- Authorization and Security module
 - Ensures that only the DBA can update the authorization & security portion of the catalog.
 - Stores information in the catalog that will ensure that a user can do only what he has been authorized to do.
- Etc.....

Data Dictionary & Data Repository Systems

- Data Dictionary and Data Repository are used to indicate a more general software utility than a catalog.
- While a catalog is mainly accessed by the various software modules of the DBMS itself, a data dictionary or data repository are mainly used by the designers, users, and administrators of a computer system for information resource management and also by the DBMS modules.
- Used also to document the database design process itself
 - Storing documentation on the results of every design phase and the design decisions
- We distinguish:
 - Passive Data Dictionary or Data repository: used only by the designer, users administrators and not the DBMS software.
 - Active Data Dictionary or Data repository: otherwise.

Overview of Oracle9i architecture

- Oracle9i server is an object-relational DBMS. It consists of: Oracle DB and Oracle server instance
- Oracle DB is a collection of data. It has logical and physical structures.
 - Logical DB structures: DB is divided in {tablespace}, tablespace={segments}, segment={extents}, and extent={contiguous blocks}. DB table or index is allocated a segment.
 - Physical DB structure: DB includes datafiles, redo log file, and control files.
- Oracle server instance comprises the memory structure and the background processes.

Overview of Oracle9i architecture

- Memory structures are divided in:
 - SGA (System Global Area): is a memory area shared by all the DB users. It contains:
 - DB buffer cache to hold blocks from the data files that have been read recently.
 - Redo log buffer is a circular buffer to hold the information about changes made to the DB.
 - Shared pool holds the library cache (shared SQL area, locks, etc.), data dictionary cache, and control structure.
 - PGA (Program Global Area): is a non-shared memory area. It contains:
 - Sort area used to sort data.
 - Software code area to store the code that is executed.
- Background processes perform maintenance work for Oracle server by executing a series of tasks
 - Database Writer (DBWn), with $n=0, \dots, 9$, to write the modified buffer to the data file.
 - Log Writer (LGWR) to write the redo log buffer to the online redo log files.
 - Checkpoint (CKPT) responsible on signaling checkpoints.
 - Others background processes: SMON, PMON, ARCn, RECO, LCKn, QMNn, Dnnn, Snnn.

Overview of Oracle9i architecture

- There are also user processes and server processes.
 - The run of an application (ex. Oracle Forms) starts a user process. The user process may be on the same machine where the instance/DB resides or may be in a client machine in Client/Server architecture.
 - Server process performs works for user processes. They gets requests from user process and interacts with Oracle instance to carry out the requests.

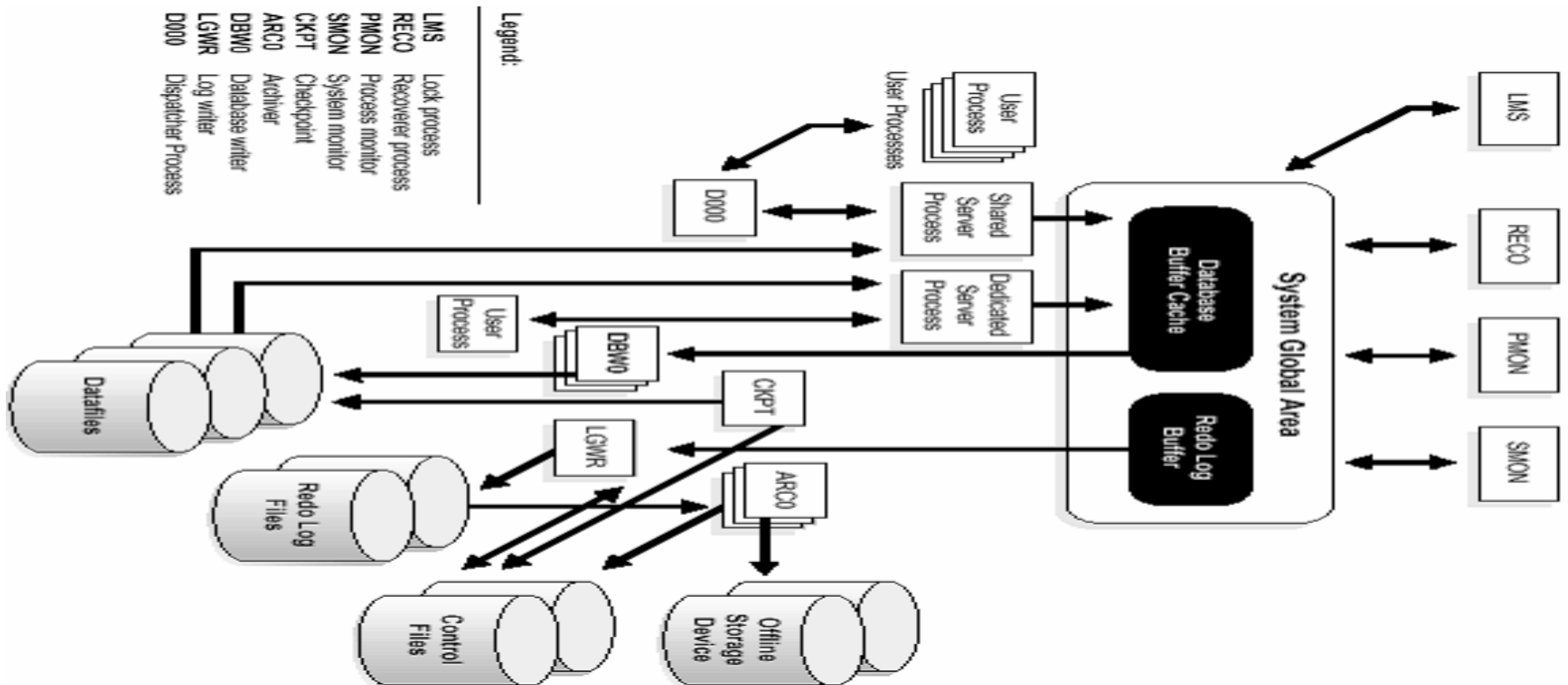


Figure 1-2 Memory Structures and Processes of Oracle