Progress®



Overview of OpenEdge RDBMS Architecture

OpenEdge®

Table of Contents

Preface....7

OpenEdge RDBMS architecture components....9

Disk-resident database....11
Logical database....11
Physical database....12
Shared memory....13
Shared memory buffers....14
Database buffers....14
BI buffers....15
AI buffers....15
Shared memory tables....16
Brokers....17
Servers....19
Clients....20
Background processes....21
Database licenses for background processes....22

How the architecture components work together....25

At database startup....26
While the database is running....27
At database shutdown....28

Single-user and multi-user environments....31

Single-user environment....32
Multi-user environment....34

Single-database and multi-database environments....37

Single-database environment....38
Multi-database environment....39

Copyright

© 2021 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

These materials and all Progress® software products are copyrighted and all rights are reserved by Progress Software Corporation. The information in these materials is subject to change without notice, and Progress Software Corporation assumes no responsibility for any errors that may appear therein. The references in these materials to specific platforms supported are subject to change.

DataDirect Cloud, DataDirect Connect, DataDirect Connect64, DataDirect XML Converters, DataDirect XQuery, DataRPM, Defrag This, Deliver More Than Expected, DevReach (and design), Icenium, Inspec, Ipswitch, iMacros, Kendo UI, Kinvey, MessageWay, MOVEit, NativeChat, NativeScript, OpenEdge, Powered by Chef, Powered by Progress, Progress, Progress Software Developers Network, SequeLink, Sitefinity (and Design), Sitefinity, Sitefinity (and design), SpeedScript, Stylus Studio, Stylized Design (Arrow/3D Box logo), Styleized Design (C Chef logo), Stylized Design of Samurai, TeamPulse, Telerik, Telerik (and design), Test Studio, WebSpeed, WhatsConfigured, WhatsConnected, WhatsUp, and WS_FTP are registered trademarks of Progress Software Corporation or one of its affiliates or subsidiaries in the U.S. and/or other countries.

Analytics360, AppServer, BusinessEdge, Chef Automate, Chef Compliance, Chef Desktop, Chef Habitat, Chef WorkStation, Corticon.js, Corticon Rules, Data Access, DataDirect Autonomous REST Connector, DataDirect Spy, DevCraft, Fiddler, Fiddler Everywhere, FiddlerCap, FiddlerCore, FiddlerScript, Hybrid Data Pipeline, iMail, JustAssembly, JustDecompile, JustMock, KendoReact, NativeScript Sidekick, OpenAccess, PASOE, Pro2, ProDataSet, Progress Results, Progress Software, ProVision, PSE Pro, Push Jobs, SafeSpaceVR, Sitefinity Cloud, Sitefinity CMS, Sitefinity Digital Experience Cloud, Sitefinity Feather, Sitefinity Insight, Sitefinity Thunder, SmartBrowser, SmartComponent, SmartDataBrowser, SmartDataObjects, SmartDataView, SmartDialog, SmartFolder, SmartFrame, SmartObjects, SmartPanel, SmartQuery, SmartViewer, SmartWindow, Supermarket, SupportLink, Unite UX, and WebClient are trademarks or service marks of Progress Software Corporation and/or its subsidiaries or affiliates in the U.S. and other countries. Java is a registered trademark of Oracle and/or its affiliates. Any other marks contained herein may be trademarks of their respective owners.

July 2021

Product version: Progress OpenEdge 12.4

Preface

Purpose

The *Overview of OpenEdge RDBMS Architecture* guide describes <placeholder>

Audience

This guide is aimed at <placeholder>.

Organization

This book is organized into the following chapters::

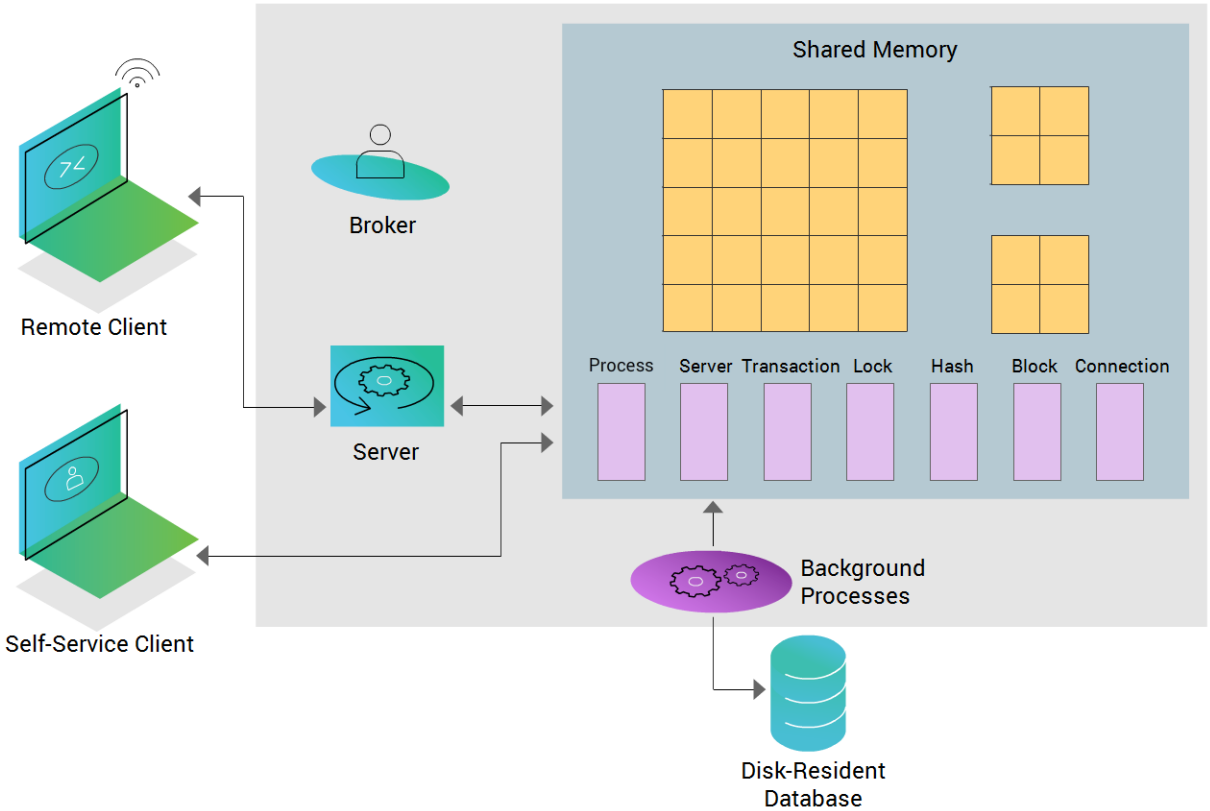
- **OpenEdge RDBMS architecture components**—
- **How the architecture components work together**—
- **Single-user and multi-user environments**—
- **Single-database and multi-database environments**—

Documentation conventions

See Documentation Conventions for an explanation of the terminology, format, and typographical conventions used throughout the OpenEdge content library.

1

OpenEdge RDBMS architecture components



To effectively administer your databases, you must understand the OpenEdge RDBMS architecture. The OpenEdge RDBMS architecture consists of a number of key components.

The table below lists each OpenEdge RDBMS architecture component with its description:

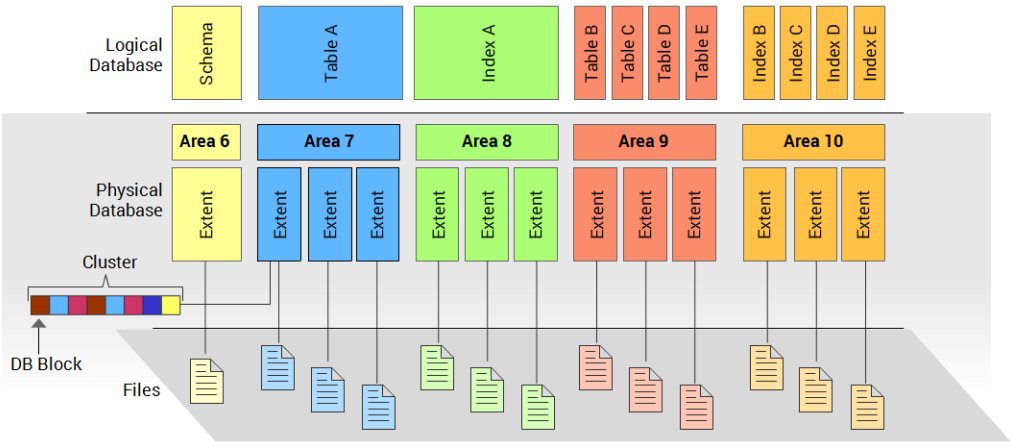
Component	Description
Disk-resident database	A collection of physical operating system files and logical storage structures that contains permanent data on disk.
Shared memory	An area in system memory that OpenEdge RDBMS uses to enable multiple users to access a database at the same time.
Brokers	A broker is a database process that creates and manages shared memory. It also launches servers and assigns remote clients to those servers.
Servers	A server is a database process that manages the interaction between a remote client and a database. It is launched as needed by the broker to handle incoming remote clients.
Clients	A client is a database process that accesses a database through a server (remote client) or directly through shared memory (self-service client).
Background processes	A background process is a database process running in the background that helps maintain data integrity and enhance database performance.

Note: In your application environment, in addition to OpenEdge ABL (Advanced Business Language) application clients, you may also have other clients such as Progress Application Servers for OpenEdge (PAS for OpenEdge), classic OpenEdge Application Servers, NameServers, and WebSpeed Agents.

For details, see the following topics:

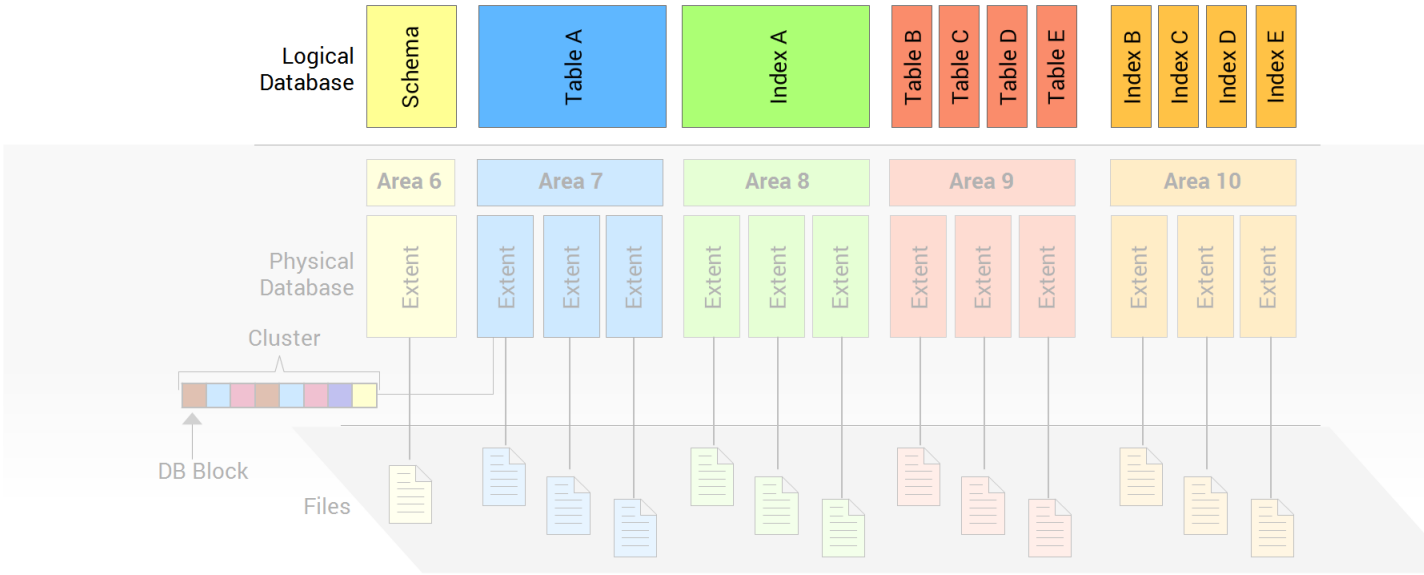
- Disk-resident database
- Shared memory
- Brokers
- Servers
- Clients
- Background processes

Disk-resident database



A disk-resident database is a collection of logical storage structures and operating system files that contains permanent data on disk. A disk-resident database is organized into a logical database and a physical database.

Logical database

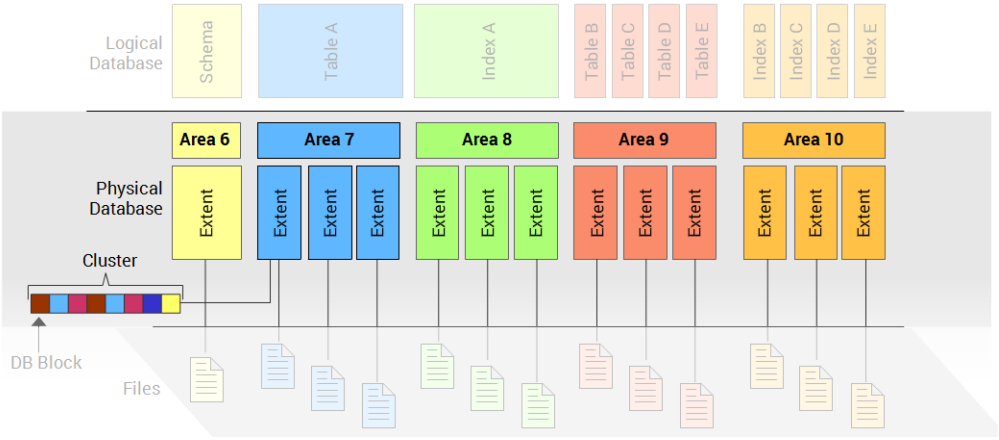


A logical database comprises the following types of database objects:

- Tables—a set of rows and columns
- Indexes—a database object used to uniquely identify rows in a table
- Sequences—a number generator used by triggers to create sequential values

Typically, application developers work with logical databases—they design and build tables, indexes, and sequences to support their applications. As a database administrator (DBA), you are primarily responsible for physical databases.

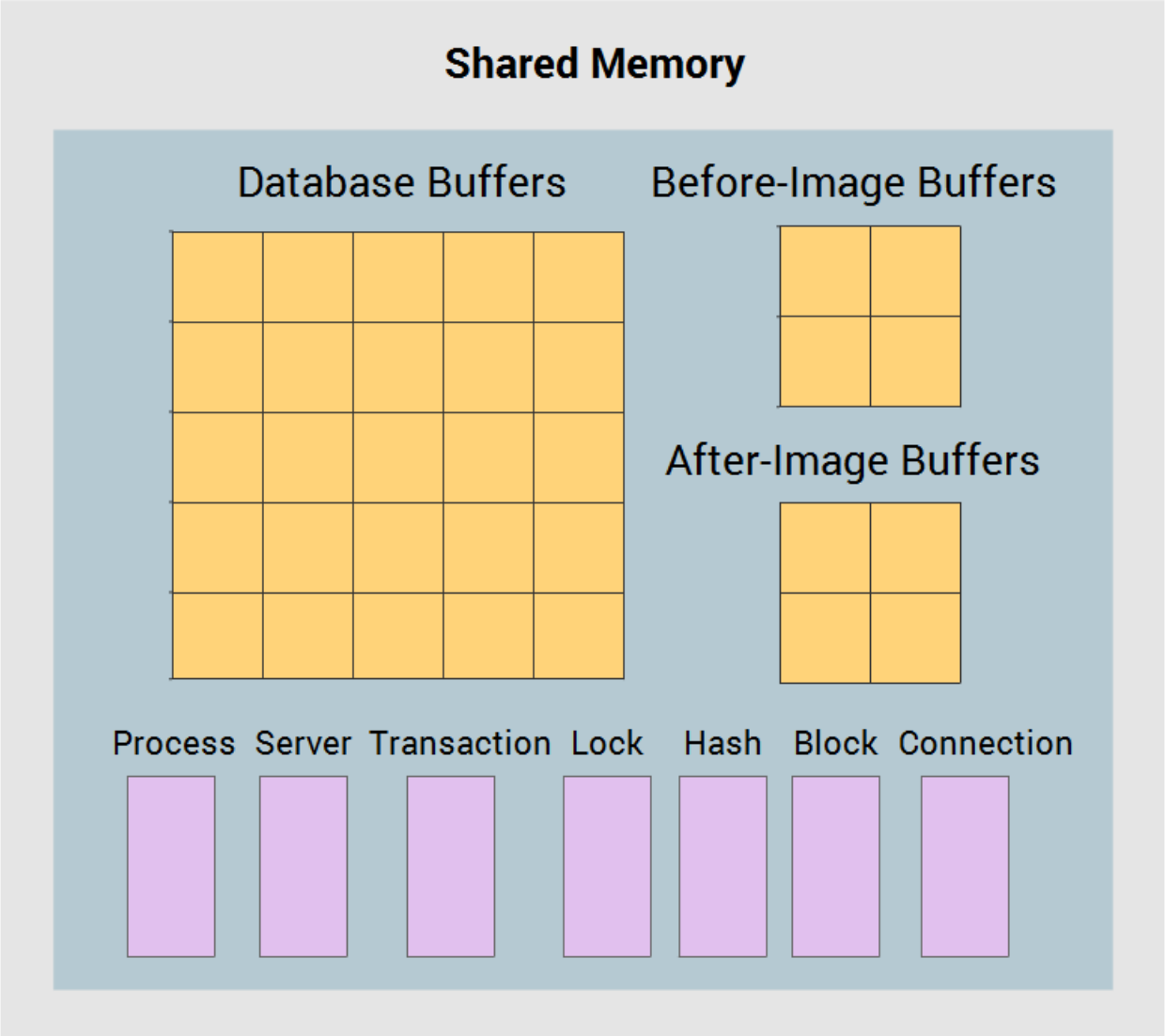
Physical database



A physical database comprises the following types of database objects:

- Storage areas—the largest storage unit of a physical database that enables you to control the types, sizes, and locations of your database files.
- Extents—an extent corresponds to a physical file on disk. A storage area is made up of one or more extents.
- Clusters—a database structure that contains a range of contiguous database blocks that belong to the same object. Each extent consists of one or more clusters.
- Database blocks—the smallest storage unit of a physical database. Each cluster consists of multiple database blocks.

Shared memory



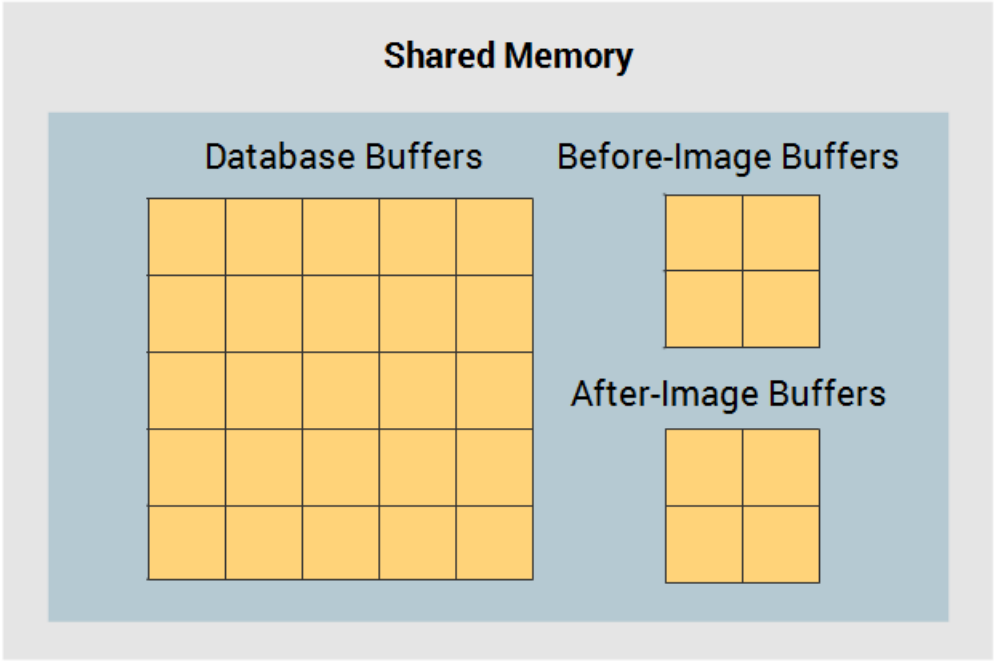
Shared memory is an area in system memory of the computer on which your database resides. OpenEdge RDBMS uses shared memory to enable multiple users to simultaneously access a running database.

Note: Shared memory is available only in a multi-user environment.A multi-user environment is one in which multiple users access a database simultaneously. Most of the time, a production database runs in the multi-user environment.

OpenEdge RDBMS uses shared memory to:

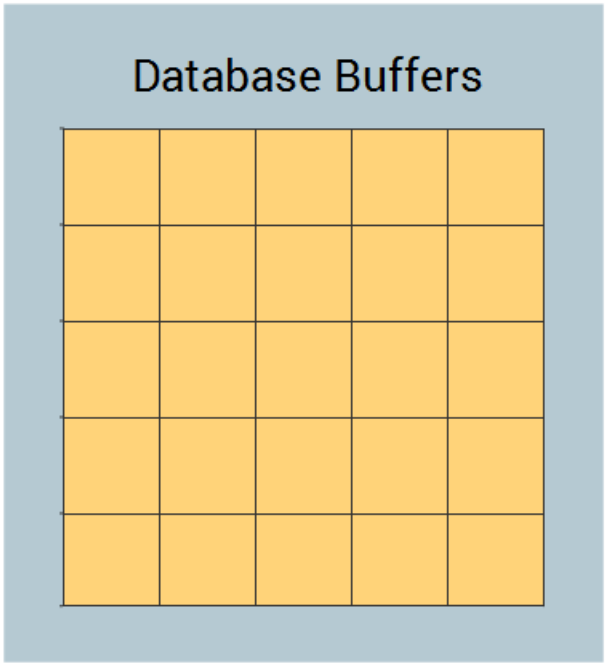
- Store user-requested data that it fetches from disk.
- Allow users to change data in memory.
- Collect notes about database transactions.
- Write changed data to disk.

Shared memory buffers



Shared memory contains three types of buffers used by OpenEdge RDBMS to support a running database: database buffers, before-image (BI) buffers, and after-image (AI) buffers.

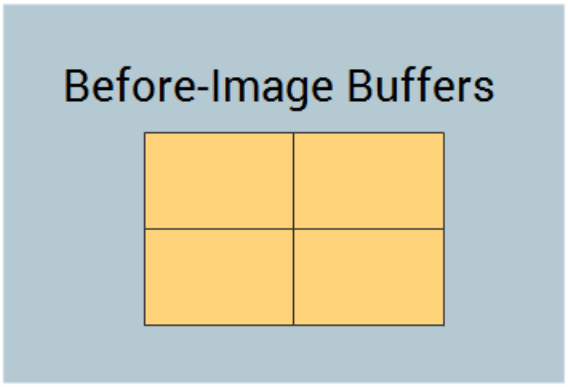
Database buffers



A database buffer is temporary in-memory storage. Each database buffer holds a copy of a database block that contains, for example, a database record. All database buffers are located in an area of shared memory called the buffer pool.

Database buffers enhance database performance and reduce disk I/O by serving as a staging area for all data manipulation operations, such as creating, updating, and deleting data.

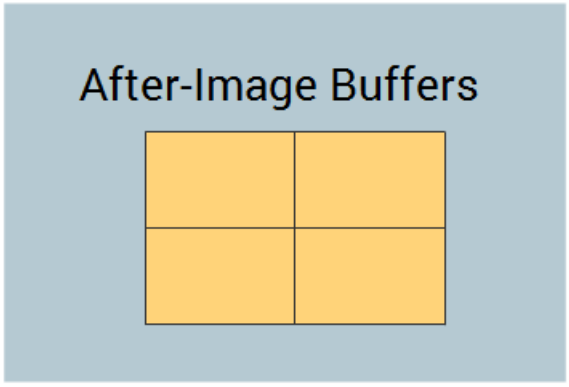
BI buffers



A BI buffer is temporary in-memory storage. OpenEdge RDBMS uses BI buffers to hold BI notes describing database transactions before writing them to the BI files on disk.

OpenEdge RDBMS uses the BI files to recover a database if it crashes (for example, if a power failure unexpectedly shuts down a database).

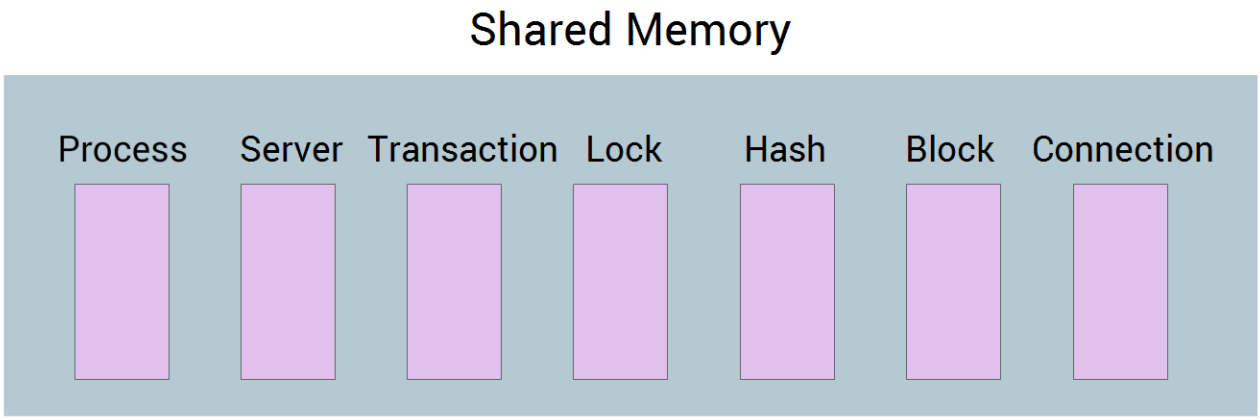
AI buffers



An AI buffer is temporary in-memory storage. OpenEdge RDBMS uses AI buffers to hold AI notes about database transactions before writing them to the AI files on disk.

AI files are used to perform roll-forward recovery of a database in the event of media failure (for example, disk damage).

Shared memory tables



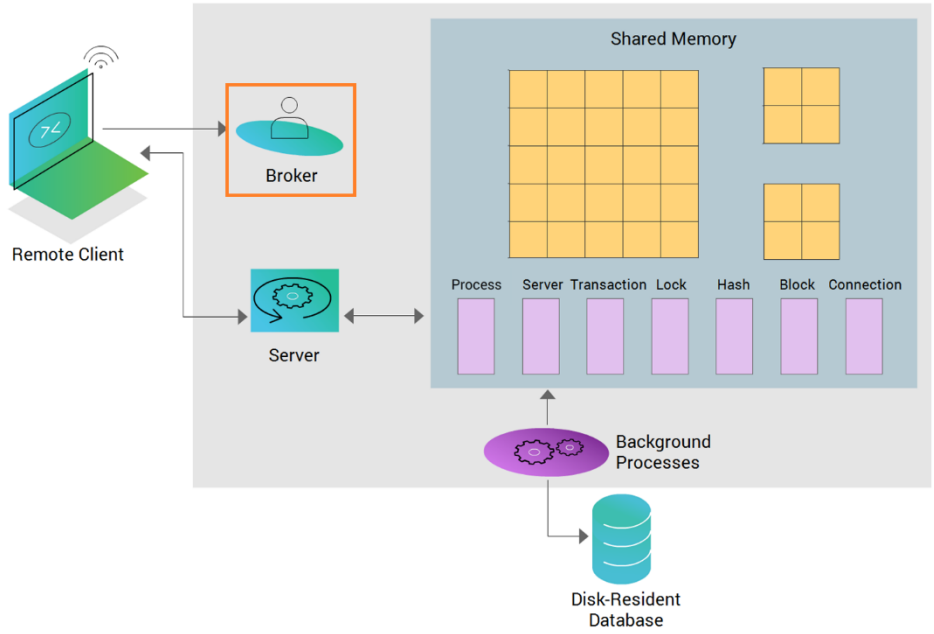
OpenEdge RDBMS also uses a number of shared memory tables to support a running database.

The table below lists each shared memory table with its description:

Table	Description
Process table	Lists all database processes that are currently accessing a database. Key database processes are brokers, servers, remote clients, and self-service clients.
Server table	Contains information about the state of the servers that provide database access to remote clients.
Transaction table	Contains information about active transactions in the database.
Lock table	Contains an entry for each record currently locked by clients. Information for each entry includes user, ROWID, lock type, and lock status. There are three kinds of locks: • NO-LOCK—You can read a record, while others can use EXCLUSIVE-LOCK, SHARE-LOCK, or NO-LOCK on the same record. • SHARE-LOCK—You can read a record and acquire an upgrade to EXCLUSIVE-LOCK to update or delete the record, while others can use SHARE-LOCK or NO-LOCK on the same record. • EXCLUSIVE-LOCK—You can read, update, or delete a record, while others can use NO-LOCK on the same record.
Hash table	Enables OpenEdge RDBMS to quickly locate database buffers that contain requested records.

Table	Description
Block table	Holds status information about database buffers.
Connection table	Contains one entry for each process that is currently connected to shared memory.

Brokers

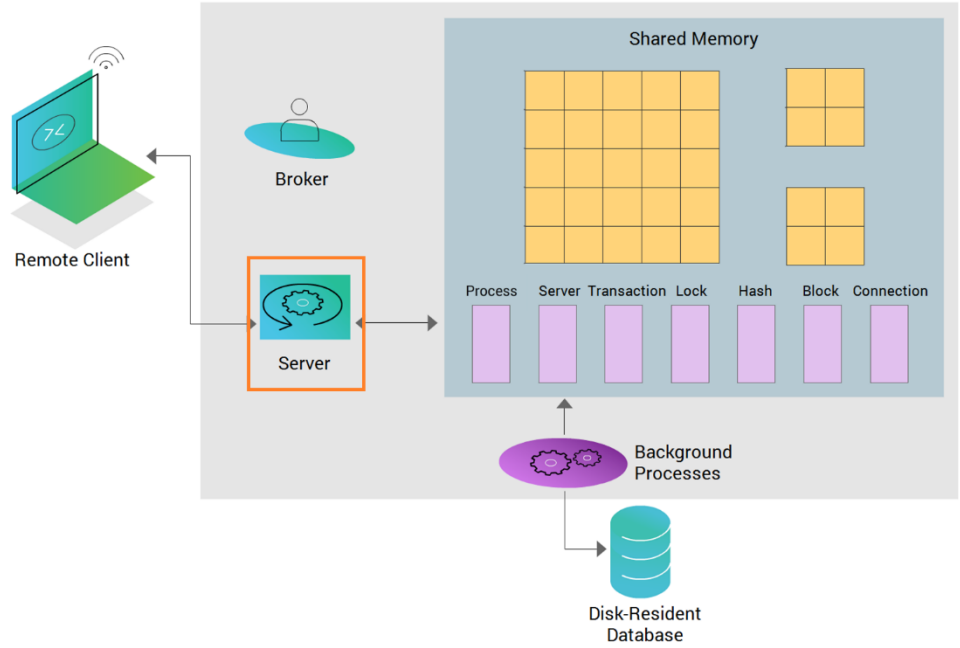


A broker is the database process that instantiates and manages the shared memory area of a running multi-user database. It performs various tasks in the four phases of a running database.

The table below lists the four phases of a running database and the tasks the broker performs in each phase:

Phase	Tasks the broker performs
At database startup	After the broker launches, it: Instantiates the shared memory area of the database based on the database startup parameters, and creates the lock (.lk) file. The .lk file is an indicator to other database processes that the database is in use. Automatically performs crash recovery to ensure database consistency and integrity. Spawns servers, allowing remote clients to connect to the database. Makes an entry in the database log (.lg) file, indicating that the broker has started. Listens for connection requests from remote clients as long as the database is running.
When remote clients connect to a database	The broker checks if there is a server available: • If a server is available, the broker assigns the remote client to the server. • If no server is available, the broker spawns a new server, and assigns the remote client to that server. Note: Whether the broker uses an existing server or spawns a new server for client connections depends on specific database startup parameters.
While the database is running	The broker periodically performs the watchdog function; that is, checking for and removing improperly terminated self-service clients. Note: The broker performs the watchdog function only if there is no dedicated watchdog running.
At database shutdown	The broker stops all remote and self-service clients, frees all shared memory structures, closes the database files, and removes the lock (.lk) file.

Servers



A server is a database process that accesses a running database through shared memory on behalf of one or more remote clients. The broker launches servers, as needed, to handle incoming remote clients.

There are two types of servers:

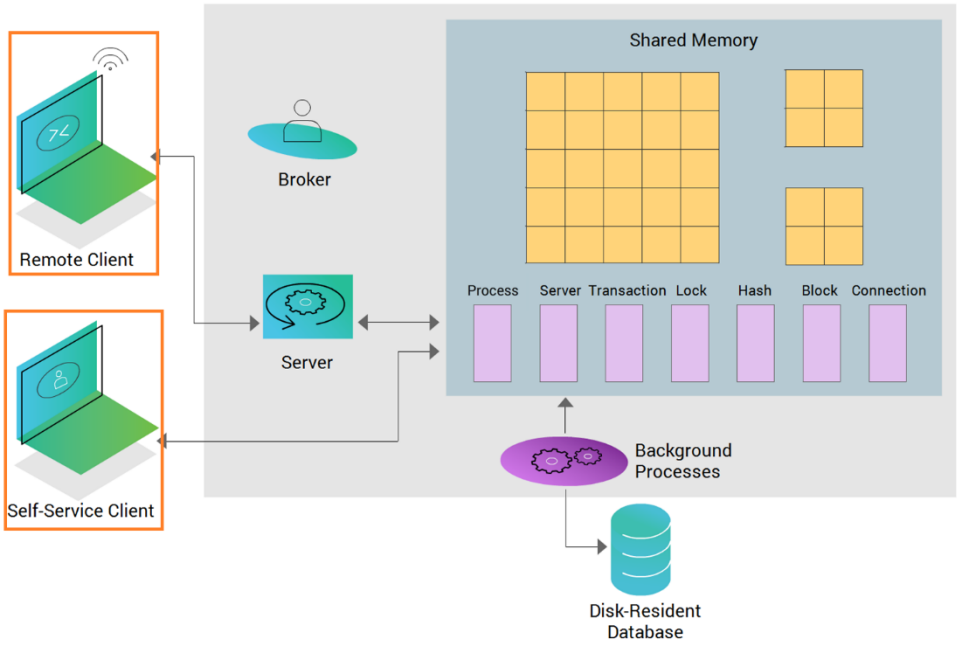
- ABL servers, which support remote OpenEdge ABL clients only
- SQL servers, which support OpenEdge SQL clients only

Note: The same server cannot support both ABL and SQL clients.

The following table lists the steps a server performs when it handles a request from a remote client:

Steps	Description
1	A remote client requests data in the database from the server.
2	The server submits the client's request to the database through shared memory.
3	OpenEdge RDBMS processes the request and returns the results to the server.
4	The server returns the results to the remote client.

Clients



A client is a database process that is initiated by a user. Clients can query, add, delete, or modify data in a database. There are two types of clients—remote and self-service.

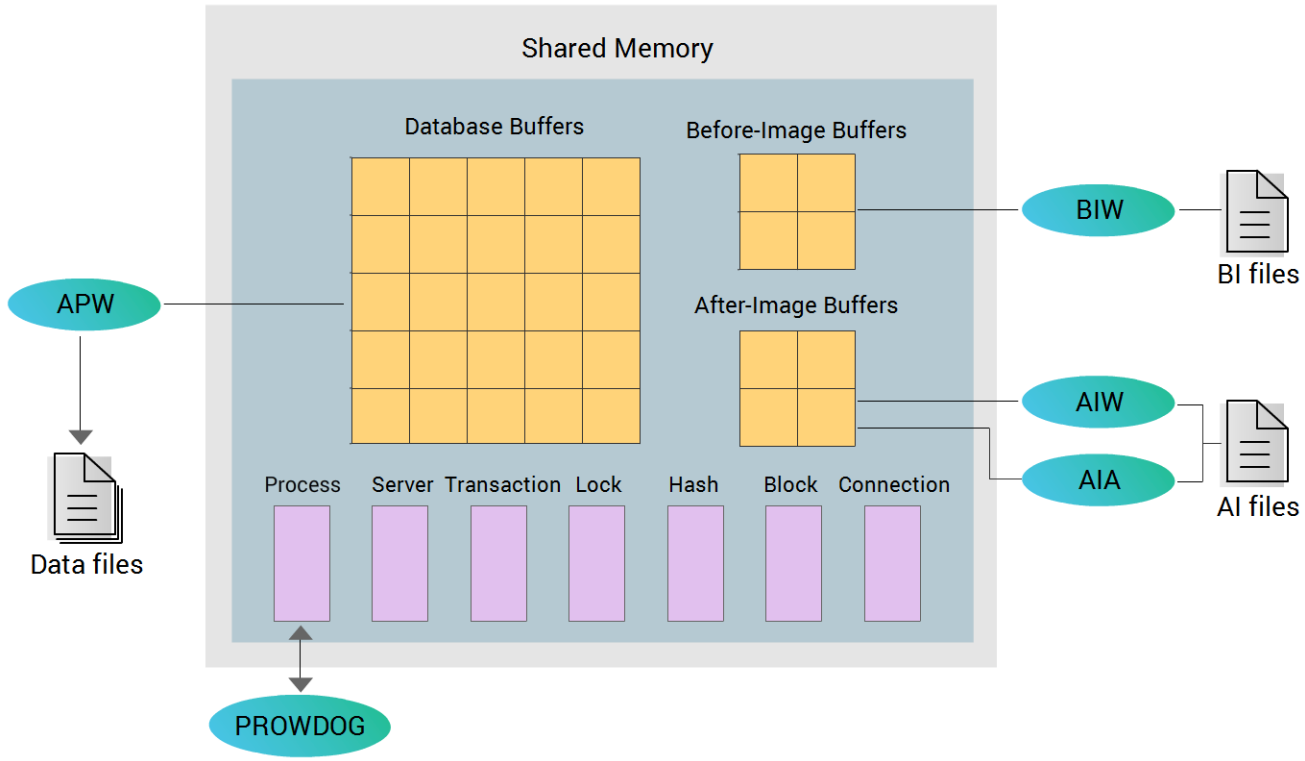
Within remote clients, there are two subtypes of clients: OpenEdge ABL and OpenEdge SQL.

- OpenEdge ABL clients include character and GUI ABL clients, Progress Application Servers for OpenEdge (PAS for OpenEdge), classic OpenEdge Application Servers, NameServers, and WebSpeed Agents.
- OpenEdge SQL clients are SQL applications that access an OpenEdge database through OBDC or JDBC.

The table below lists each type of client with its description:

Client	Description
Remote	A remote client accesses the database only through a server spawned by the broker, and never directly. Note: OpenEdge RDBMS treats SQL clients as remote clients even if they reside on the same computer as the database.
Self-service	A self-service client: • Resides on the same computer as the database. • Accesses the database directly through shared memory. • Performs both client and server functions in one process. • Executes application logic directly.

Background processes



A background process is a database process that runs in the background. Background processes enhance database performance and maintain data integrity in the event of a system crash or media failure. There are five types of background processes.

The following table lists each background process with its description:

Background process	Description
Before-Image Writer (BIW)	The process that continually writes filled BI buffers to the BI files on disk. The BIW enhances database performance because it ensures a steady supply of empty BI buffers.
After-Image Writer (AIW)	The process that continually writes full AI buffers to the AI files on disk. The AIW improves database performance because it ensures a steady supply of empty AI buffers.
After-Image Archiver (AIA)	The AI file management utility that automatically archives full AI files to disk and maintains a log file to help you with roll-forward recovery if required. The AIA frees you to perform other database administration tasks.

Background process	Description
Asynchronous Page Writer (APW)	A process that continually writes unlocked, modified database buffers to the data files on disk. An APW enhances database performance because it: • Ensures that a steady supply of empty database buffers is available. • Reduces the number of database buffers that OpenEdge RDBMS must examine before writing an unlocked, modified database buffer to disk. • Reduces the overhead associated with checkpointingFor information on how to monitor and manage checkpointing because fewer modified database buffers have to be written to disk at every checkpoint.
Watchdog (PROWDOG)	The process that: Checks for the existence of the lock (.lk) file and ensures that the .lk file is valid. If the file is lost or corrupted, the database is automatically shutdown. Cleans up after improperly terminated self-service clients as well as the remote clients of lost servers. Checks if the broker is still alive. If for any reason the broker is not alive, the database is automatically shut down. If the watchdog is not running, the broker automatically performs the first two tasks.

Database licenses for background processes

Depending on your database license, you may or may not have certain background processes. The following table lists the database licenses available with each background process:

Background process	Database license		
		Workgroup	Personal
Before-Image Writer (BIW)	Yes	No	No
After-Image Writer (AIW)	Yes	No	No
After-Image Archiver (AIA)	Yes	No	No
Asynchronous Page Writer (APW)	Yes	No	No
Watchdog (PROWDOG)	Yes	Yes	Yes

Note: The writer and AI archiver background processes are not available with the Workgroup and Personal database licenses. The OpenEdge RDBMS database engine performs the tasks of these background processes. The Enterprise license comes with all the writers and the AI archiver. This means once started they take over the work from the database engine, thus improving scalability and enhancing performance.

2

How the architecture components work together

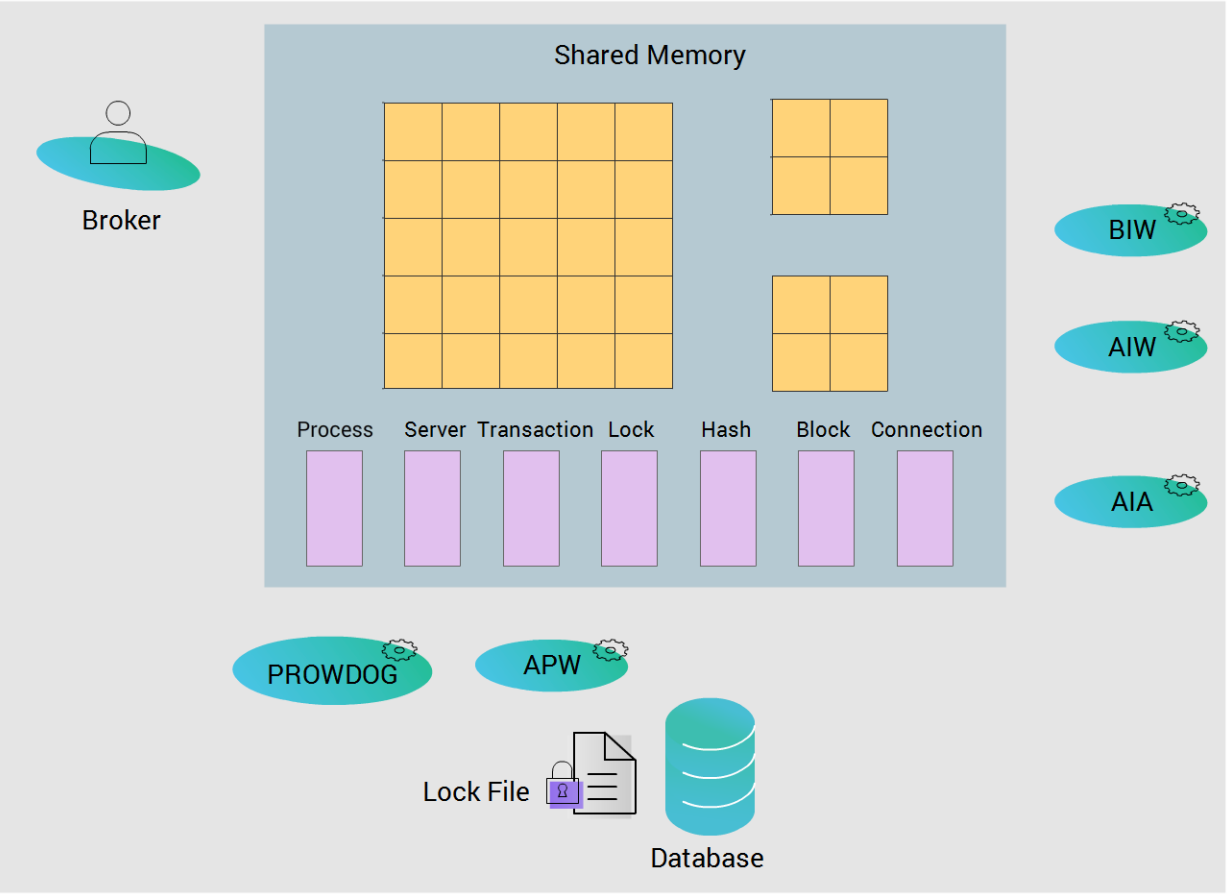
So far you have learned about the key components of the OpenEdge RDBMS architecture. To effectively administer your databases, you need to understand how these components work together.

Next, you will learn how the components of OpenEdge RDBMS architecture work together at database startup, while the database is running, and at database shutdown.

For details, see the following topics:

- At database startup
- While the database is running
- At database shutdown

At database startup

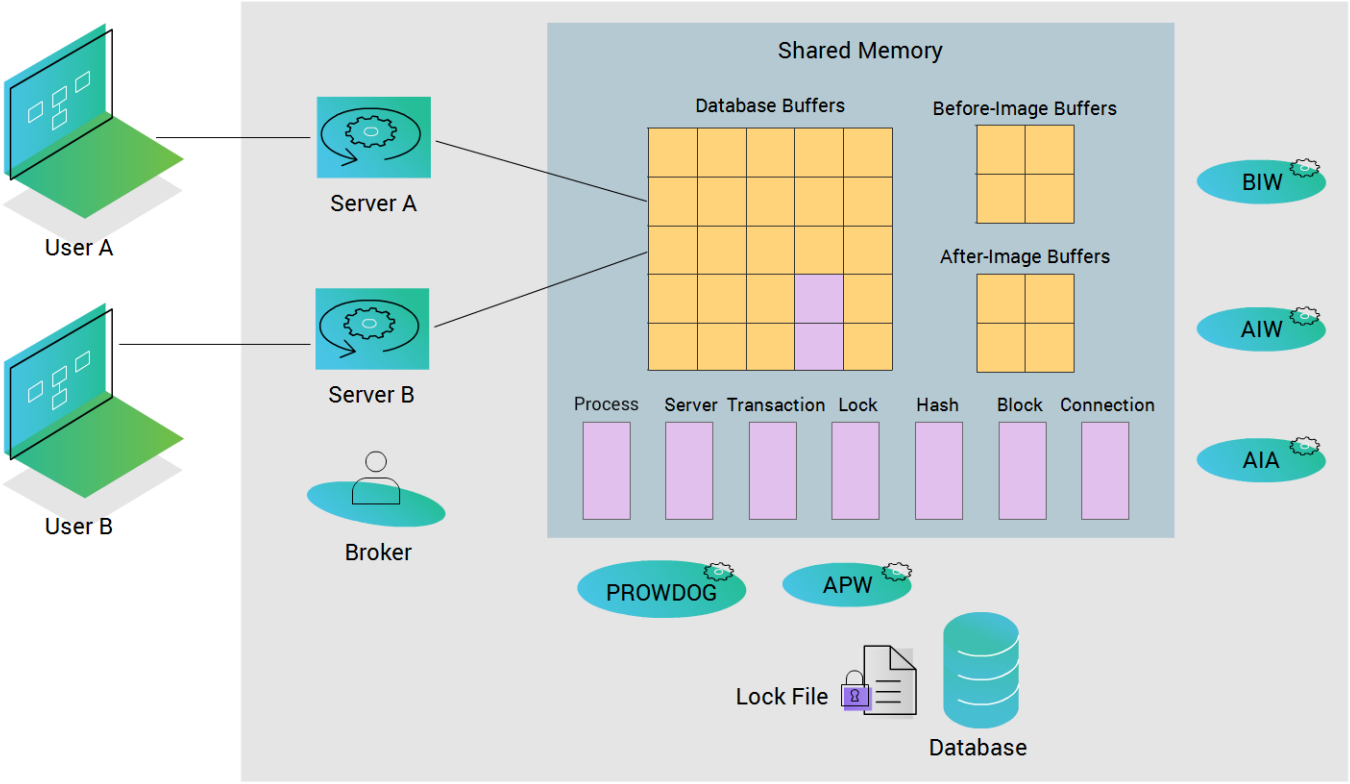


When you start up a database in multi-user mode, different OpenEdge RDBMS architecture components start. The database startup process works as follows:

Step	Description
1	When you start up a database in multi-user mode, OpenEdge RDBMS launches the broker.
2	The broker instantiates the shared memory area of the database, creates the lock (.lk) file, performs automatic crash recovery, and starts listening for connection requests from remote clients.
3	Depending on your database license, as a best practice you should start the available background processes for optimal performance.

Note: The database is only one part of your application environment. You may also have to start Progress Application Servers for OpenEdge (PAS for OpenEdge), classic OpenEdge Application Servers, NameServers, or WebSpeed Agents.

While the database is running



While the database and background processes are running, users can connect to the database and access data. A running database uses all the OpenEdge RDBMS architecture components (disk-resident database, shared memory, brokers, servers, and background processes) to process data requests from clients. The process works as follows:

Step	Description
1	When users connect to a database, the broker launches new servers as necessary to support the users.
2	As users requests data, the associated server first determines if the requested data is already in the database buffers in shared memory. • If the data is in memory, the server returns the data to the user. • If the data is not in memory, the server fetches the data from disk, stores it in database buffers, and returns the data to the user.
3	When users modify data, the data is modified in the database buffers.

Step	Description
4	Transactions are committed or rolled back as required.
5	While the database is running: • The BIW writes BI notes about database transactions first to the BI buffers, and then to the BI files on disk. • The AIW writes AI notes about database transactions first to the AI buffers, and then to the AI files on disk. • The APW writes unlocked, modified data from the database buffers to disk.

- To maintain data integrity, OpenEdge RDBMS ensures that completed transactions are always committed and incomplete or failed transactions are always rolled back. OpenEdge RDBMS adheres to the ACID (Atomicity, Consistency, Isolation, and Durability) properties by default:
- Atomicity—treats a transaction as a logical unit of work that must be performed together or not at all.
- Consistency—requires that the database state after the execution of a transaction is consistent.
- Isolation—provides record locking to isolate a transaction from other transactions.
- Durability—ensures that after a transaction is committed, its effects on the database are permanent.

At database shutdown

Depending on your environment, you may choose to schedule regular downtime (for example, weekly, monthly, or quarterly) to perform routine database maintenance tasks. You should choose a time when there is little or no database activity and inform your users ahead of time.

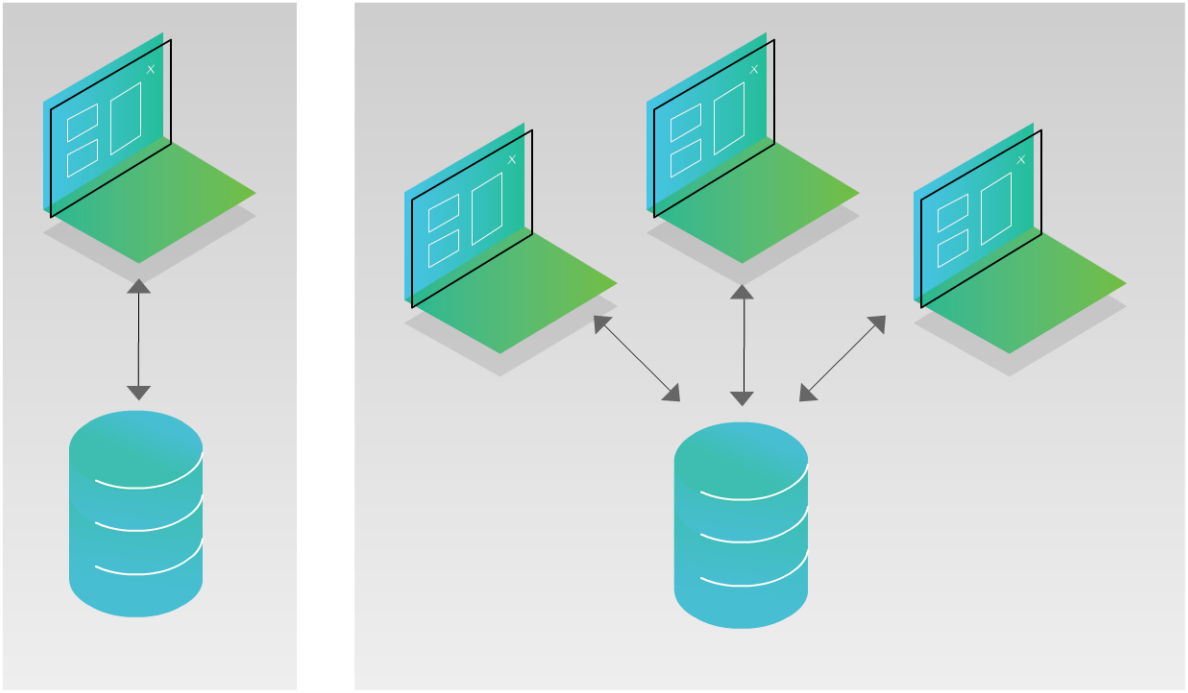
Here is what happens when you shut down a database in normal mode:

Step	Description
1	The broker is informed that the shutdown has been requested and it initiates the shutdown.
2	Any uncommitted transactions are rolled back.
3	The BIW writes any remaining BI transaction notes from the BI buffer to the BI files on disk.
4	The AIW writes any remaining AI transaction notes from the AI buffer to the AI files on disk.

Step	Description
5	The APW writes all modified database buffers to the data files on disk.
6	The broker: • Disconnects all remote and self-service clients connected to the database. • Frees shared memory structures (BI buffers, AI buffers, database buffers, and shared memory tables). • Closes the database files. Removes the lock (.lk) file. • Makes an entry in the database log (.lg) file, indicating that the database is shutdown.

3

Single-user and multi-user environments



Every database can run in two distinct user environments: single-user and multi-user.

For details, see the following topics:

- Single-user environment
- Multi-user environment

Single-user environment

In a single-user environment, only one user can access a database at a time; all other users are locked out of the database. For example, if you connect to a database in single-user mode to perform schema changes, all other users would be unable to connect to the database that you are accessing.

There are a number of administration tasks that require exclusive access to a database, such as:

- Schema changes
- Roll-forward recovery of a database
- Rebuilding of indexes
- Truncation of the BI files
- Converting a database from an older version of OpenEdge to a newer version

In addition, if your environment allows you to have regular downtime for database maintenance tasks, there are a number of tasks you may want to perform in single-user mode, such as:

- Offline dumping and loading of data
- Offline bulk loading of a database
- Offline backup and restoring of a database
- Offline adding files to the database
- Offline enablement of after-imaging
- Offline enablement of automatic AI file management

Multi-user environment

In a multi-user environment, multiple users can access a database at the same time. For example, User A, a business analyst, can connect to a database and generate a report; User B, an application developer, can connect to the same database and test a new application that accesses the database; and User C, a DBA, can perform an online backup of the database.

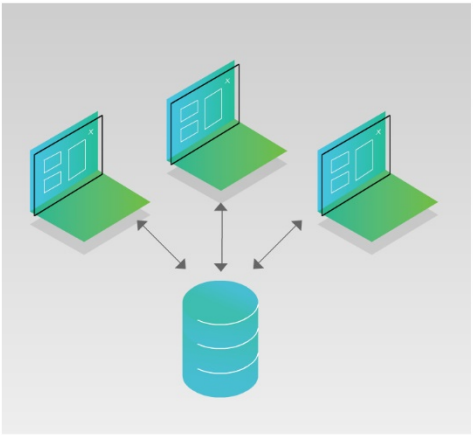
In the multi-user environment, databases can be accessed simultaneously by many users employing a variety of applications, such as OpenEdge ABL, Java, ODBC, and Web-based applications.

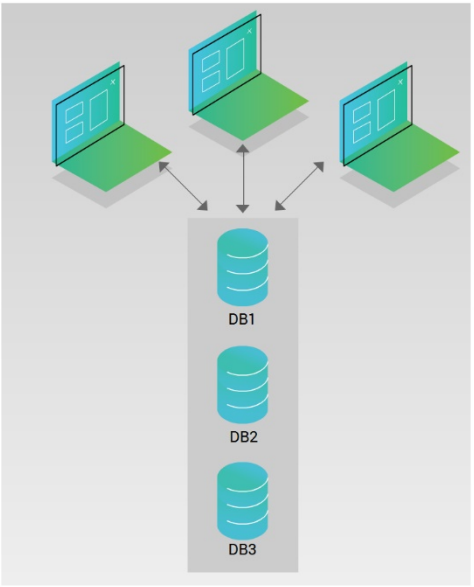
As a DBA, you use the multi-user environment when you need to perform tasks that must be done while other users are still accessing the database, such as:

- Online backup of a database
- Online dumping and loading of data
- Online adding files to the database
- Online enablement of after-imaging
- Online enablement of automatic AI file management
- Online index activation for a specific index
- Online index activation or index rebuild for a specific tenant or tenant group, if you have a multi-tenant database
- Online moving of a table or index for a specific tenant or tenant group
- Online index activation or index rebuild for a specific table partition, if you have a table-partitioned database
- Online moving of a table or index for a specific table partition

4

Single-database and multi-database environments





Each database can run either alone (single-database environment) or along with other databases (multi-database environment). This gives you great flexibility in creating and managing your business data.

For details, see the following topics:

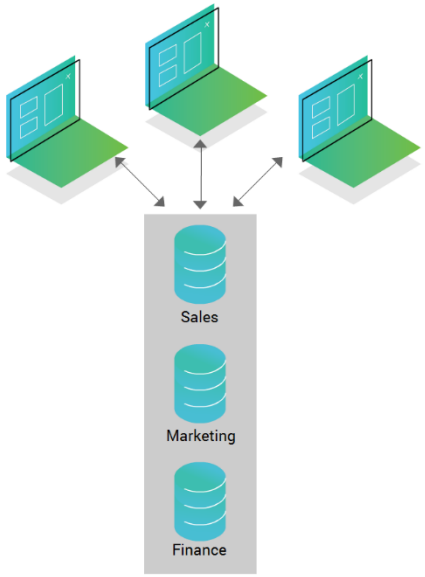
- Single-database environment
- Multi-database environment

Single-database environment

In a single-database environment, you need to access just one database at a time. In this environment, a single database contains all the structures and data your application requires.

For example, you might have a sales application that accesses only a sales database. When you launch the application, it automatically connects to the database, and when you exit the application, it automatically disconnects from the database.

Multi-database environment



In a multi-database environment, you need to access two or more databases at the same time. This environment requires that the databases you access do not share the same table names or that the table names are qualified with the database name.

For example, you have an OpenEdge ABL application that accesses data from a sales database, a marketing database, and a finance database. Every time you launch the application, it automatically connects to all three databases at the same time. This allows you to query, add, modify, or delete data from all three databases.

OpenEdge RDBMS implements two-phase commit to ensure consistency in a multi-database environment. Two-phase commit does this by making sure that all databases involved in a distributed transaction commit the transaction or none commit the transaction.