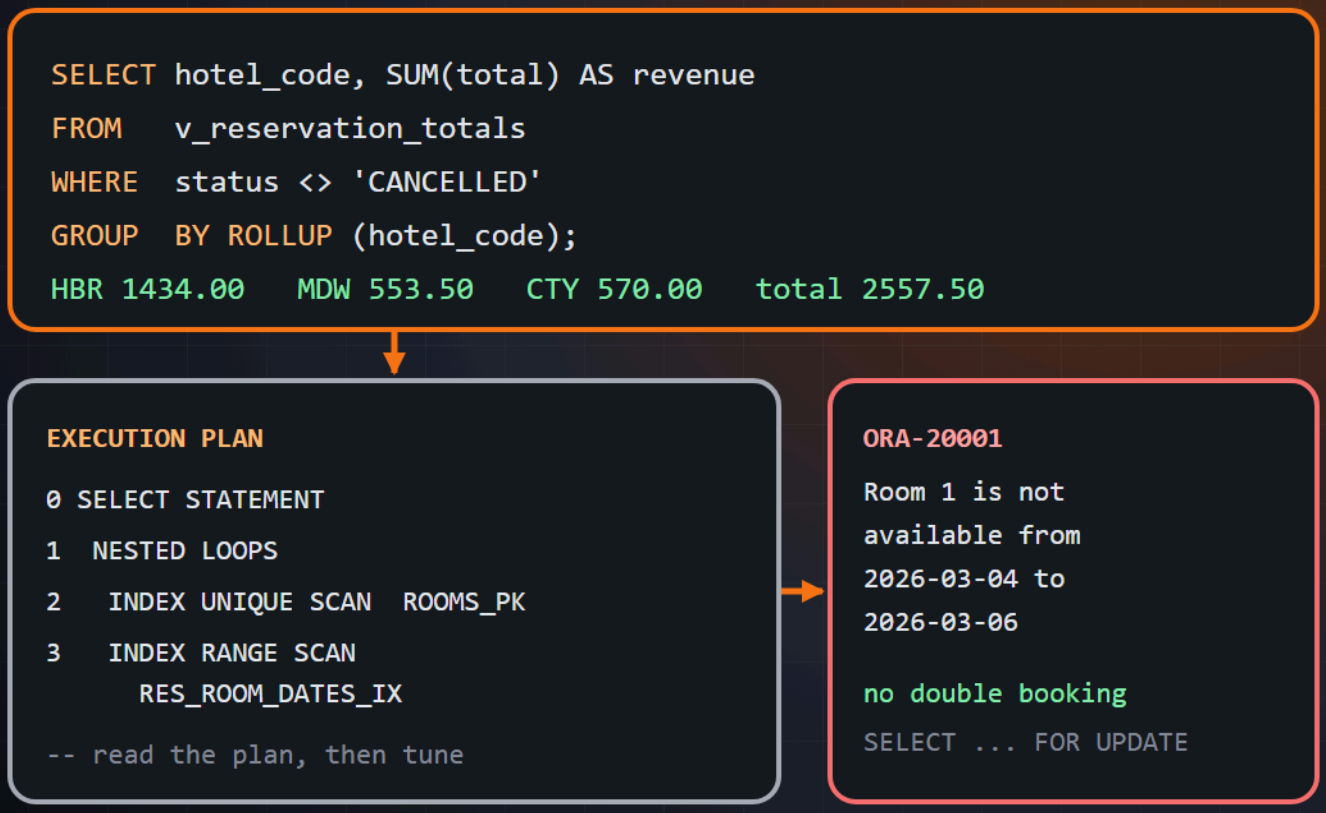


SQL · INDEXES · UNDO · OPTIMIZER · RMAN · SECURITY

FOR DEVELOPERS AND NEW DBAS

Oracle Database Essentials

SQL, architecture, performance, and administration — explained clearly

From your first query to execution plans, concurrency, backup and recovery — 19c and 23ai

SQLCL · SQL DEVELOPER · PL/SQL · DATA PUMP · RMAN

THENEXBOOKS

Oracle Database Essentials

SQL, Architecture, Performance, and Administration for Developers and New DBAs

First Edition

THENEXBOOKS

Oracle Database Essentials
SQL, Architecture, Performance, and Administration for Developers and New DBAs

First Edition, 2026

Copyright © 2026 thenexbooks (thenexbooks.gumroad.com). All rights reserved.

No part of this publication may be reproduced, distributed, or transmitted in any form or by any means without the prior written permission of the publisher, except for brief quotations in reviews and certain other non-commercial uses permitted by copyright law. Code examples may be used freely in your own projects.

The information in this book is provided "as is" without warranty of any kind. While every precaution has been taken in its preparation, the publisher assumes no responsibility for errors or omissions, or for damages resulting from the use of the information contained herein. Oracle Database features, licensing terms, and tool output change between releases; always check the documentation and license terms that apply to your installation, and test backup and recovery procedures before you rely on them.

Oracle, Java, and MySQL are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners. This book is independent and is not affiliated with, sponsored by, or endorsed by Oracle Corporation.

Sablewood Hotels, its staff, guests, and hotels are fictional. Any resemblance to real companies or people is coincidental.

Preface

Oracle Database runs a remarkable share of the world's business systems: banking, billing, airline reservations, government records, hospital systems, and countless internal applications. Sooner or later, many developers find themselves connected to an Oracle database, and many teams discover that someone has to look after one. If that is you, this book is for you.

Oracle can feel intimidating at first. It has its own vocabulary — instances and PDBs, undo and redo, tablespaces and segments — its own SQL extensions, and a reputation for complexity. But underneath, its design is consistent and well reasoned. Once you understand how Oracle stores data, keeps queries consistent while others change the same rows, decides how to run a statement, and protects committed work, the rest falls into place. This book builds that understanding step by step, always connecting concepts to code you can run.

We start by installing the free edition of Oracle Database in a container and connecting to it. Then we learn how an Oracle database is put together, write Oracle SQL from simple queries to analytic functions, design tables, constraints, indexes, views, and partitions, and study concurrency and read consistency. We write just enough PL/SQL to build reliable database logic, use the newest SQL and JSON features, and connect applications in Java, Python, and Node.js. Finally we learn how the optimizer works, how to read execution plans and tune queries, how to monitor a database, and how to manage storage, backups, and recovery.

Who This Book Is For

- **Application developers** who build software on Oracle and want to write correct, fast SQL and understand what the database is doing.
- **Developers and DBAs coming from PostgreSQL, MySQL, or SQL Server** who need to learn Oracle's concepts and differences quickly.
- **New and "accidental" DBAs** who have been handed an Oracle database and need a solid foundation in architecture, security, performance, and backups.
- **Students** in database courses who want a practical, modern introduction to Oracle.

What You Will Learn

- Running Oracle Database Free in a container, connecting with SQL Developer and SQLcl, and how instances, CDBs, PDBs, memory, processes, and files fit together.
- Oracle SQL: queries, functions, data types, joins, subqueries, aggregation, analytic functions, hierarchical queries, PIVOT, MERGE, and transactions.
- Schema design: tables, constraints, identity columns, indexes, views, materialized views, sequences, synonyms, and partitioning.

- Concurrency: read consistency, undo, locking, isolation levels, and how to prevent double bookings.
- PL/SQL essentials, packages and triggers, JSON and the newest SQL features, and connecting applications safely with bind variables.
- Security with users, roles, privileges, and least privilege.
- How the optimizer works, reading execution plans, tuning queries, statistics, and monitoring.
- Tablespaces and redo, backup and recovery with RMAN, Data Pump, and Flashback, and an overview of high availability and the cloud.

How This Book Is Organized

Part I — Getting Started (Chapters 1–5) installs Oracle Database Free, explains the architecture, creates the Sablewood schema, and covers SQL fundamentals. **Part II — Oracle SQL in Depth** (Chapters 6–10) covers data types, joins, aggregation and analytic functions, advanced queries, and changing data. **Part III — Schema Design and Database Objects** (Chapters 11–15) covers constraints, indexes, views, partitioning, and concurrency. **Part IV — Programming and Connecting** (Chapters 16–20) covers PL/SQL essentials, packages and triggers, JSON and modern SQL, application connectivity, and security. **Part V — Performance** (Chapters 21–24) covers the optimizer, execution plans, statistics, and monitoring. **Part VI — Operations and Next Steps** (Chapters 25–28) covers storage, backup and recovery, high availability, and a production checklist. The appendices provide an Oracle SQL cheat sheet, a data dictionary reference, an ORA- error field guide, and a glossary.

Each chapter includes practice exercises and ends with a **Key Takeaways** box for quick review.

The Running Example: Sablewood Hotels

Throughout the book we build and run the database for **Sablewood Hotels**, a small group of three hotels — Sablewood Harbor, Sablewood Meadow, and Sablewood City. Iris, the operations director, asks the business questions; Tom, the application developer, builds the booking system; and Priya, newly appointed as the group's DBA, keeps the database healthy. You will create their schema of hotels, rooms, guests, reservations, and payments, query it in every way a business needs, protect it from double bookings, connect applications to it, tune it, secure it, and back it up.

Versions and Conventions

Examples work on Oracle Database 19c and later; features that need Oracle Database 23ai are clearly marked. The practice environment is Oracle Database Free. Query results are shown as tables, and execution plans are shown for their shape only, because exact costs and timings depend on your system. Some features discussed in Parts V and VI require Enterprise Edition options or management packs; these are pointed out where they appear.

> **TIP**
>
> A recommended practice or a technique that saves you time.

> **NOTE**
>
> Background information, version or licensing details, or a useful aside.

> **WARNING**
>
> A common mistake, a risky command, or a habit to avoid.

If you want to go deeper into PL/SQL after this book, the companion titles *PL/SQL Programming from Zero* and *Oracle PL/SQL Programming* pick up where Chapters 16 and 17 leave off.

Contents

Preface

PART I — GETTING STARTED

1 Oracle Database Today: Editions, Releases, and Where It Fits

2 Installing Oracle Database Free and Connecting: Containers, SQL Developer, and SQLcl

3 Oracle Architecture for Developers: Instances, CDBs and PDBs, Memory, Processes, and Files

4 Users, Schemas, and the Sablewood Database

5 Oracle SQL Fundamentals: SELECT, Filtering, Sorting, and Single-Row Functions

PART II — ORACLE SQL IN DEPTH

6 Data Types: NUMBER, VARCHAR2, DATE, TIMESTAMP, LOBs, BOOLEAN, and JSON

7 Joins, Subqueries, and Set Operators

8 Aggregation and Analytic Functions

9 Advanced Queries: WITH, Hierarchical Queries, PIVOT, and Row Limiting

10 Changing Data: INSERT, UPDATE, DELETE, MERGE, and Transactions

PART III — SCHEMA DESIGN AND DATABASE OBJECTS

11 Tables and Constraints: Keys, Checks, Identity Columns, and Data Integrity

12 Indexes: B-Tree, Bitmap, Function-Based, and Composite Indexes

13 Views, Materialized Views, Sequences, and Synonyms

14 Partitioning Fundamentals

15 Concurrency and Read Consistency: Undo, Locks, and Isolation

PART IV — PROGRAMMING AND CONNECTING

16 PL/SQL Essentials for Database Developers

17 Packages and Triggers: Building the Sablewood Booking Logic

18 JSON and Modern SQL Features in Oracle Database 23ai

19 Connecting Applications: Java, Python, and Node.js with Bind Variables and Connection Pools

20 Security: Users, Roles, Privileges, and Least Privilege

PART V — PERFORMANCE

21 How Oracle Executes SQL: Parsing, the Optimizer, and Execution Plans

22 Reading Execution Plans and Tuning Queries

23 Optimizer Statistics, Bind Variables, and Cursor Sharing

24 Monitoring and Diagnostics: Dynamic Performance Views, the Alert Log, and AWR

PART VI — OPERATIONS AND NEXT STEPS

25 Storage Management: Tablespaces, Datafiles, Redo, and Archiving

26 Backup and Recovery: RMAN, Data Pump, and Flashback

27 High Availability and the Cloud: Data Guard, RAC, and Autonomous Database in Overview

28 Project: Running Sablewood End to End — A Production Checklist and Where to Go Next

APPENDICES

A Oracle SQL Cheat Sheet

B Data Dictionary and V$ Views Quick Reference

C ORA- Error Field Guide

D Glossary

PART I

Getting Started

What Oracle Database is and where it fits, running the free edition in a container, how an Oracle database is put together, creating the Sablewood schema, and writing your first Oracle SQL.

1 · Oracle Database Today
2 · Installing and Connecting
3 · Oracle Architecture
4 · Users, Schemas, and Sablewood
5 · Oracle SQL Fundamentals

CHAPTER 1 · PART I — GETTING STARTED

Oracle Database Today: Editions, Releases, and Where It Fits

Before you install anything, it helps to know what Oracle Database actually is, which edition and release you are likely to meet at work, and how it compares with the databases you may already know. This chapter gives you that map, introduces the Sablewood Hotels team you will work with for the rest of the book, and lays out the route we will take together.

What Oracle Database Is

Oracle Database is a relational database management system: software that stores data in tables, enforces rules about that data, lets many users read and change it at the same time, and protects it against crashes and mistakes. You talk to it in **SQL** (Structured Query Language) and, when you need procedural logic close to the data, in **PL/SQL**, Oracle's procedural extension to SQL. Around that core sits a very large set of features: partitioning, parallel execution, advanced security, replication, high-availability clustering, JSON and spatial data, and in recent releases vector search for AI workloads.

Oracle released the first commercially available SQL relational database in 1979, and the product has been in continuous development ever since. That long history explains two things you will notice quickly. First, the database is extremely mature: features that are new or still maturing in other products, such as read-consistent queries that never block writers, online index rebuilds, or point-in-time recovery, have been part of Oracle for decades. Second, it carries some historical habits that surprise newcomers, such as treating the empty string as `NULL` or having a one-row table called `DUAL`. We will point those habits out as we meet them so they never catch you off guard.

Oracle Database is used heavily in banking, telecommunications, government, retail, healthcare, and logistics, often for the systems that must not lose a single committed transaction. The practical consequence for you is that Oracle skills tend to be valuable and long-lived: the SQL you write today against a 19c database will, with very few exceptions, run unchanged on the newest release.

Database, instance, and server: a first distinction

People use the word "database" loosely. In Oracle's vocabulary the **database** is the set of files on disk that hold your data, and the **instance** is the set of memory structures and background processes that open those files and serve requests. When someone says "the database is down", they usually mean the instance is not running; the files are still there. Chapter 3 explains the architecture properly. For now, remember that the two words mean different things in Oracle, and that both live on a **database server**, the machine (physical, virtual, or a container) that runs the software.

The Editions

Oracle Database is sold and distributed in several **editions**. The SQL language is essentially the same in all of them; what differs is the set of optional features, the hardware limits, and the license terms. Knowing which edition you are on matters because a feature that works in your practice environment may not be licensed in production.

Edition	Typical use	Key points
Oracle Database Free	Learning, development, small apps	No cost; resource limits apply
Standard Edition 2 (SE2)	Small and medium production	Lower price; hardware limits; fewer options
Enterprise Edition (EE)	Large and critical production	All core features; extra-cost options and packs
Cloud services	Managed databases on OCI and partners	Oracle runs infrastructure; you pay by use

Oracle Database Free

Oracle Database Free is the edition we use throughout this book. It replaced the older Express Edition (XE) and is free to download, use, and even deploy for small workloads. At the time of writing it is limited to 2 CPU threads, 2 GB of RAM for the database, and 12 GB of user data. It is distributed as a container image, as Linux RPM packages, and as a Windows installer, and in Chapter 2 we run it in a container.

Free is built from the same code base as the commercial editions, so most features you will learn here, including some that are extra-cost options elsewhere, are present. That is wonderful for learning, but it creates a trap we will warn about repeatedly: being able to *use* a feature in Free tells you nothing about whether your employer is licensed to use it in production.

Standard Edition 2

Standard Edition 2 (SE2) is the lower-priced commercial edition. It is restricted to smaller servers (at the time of writing, servers with at most two processor sockets, and a cap on the number of CPU threads each database may use) and it omits many Enterprise features, for example parallel query execution, the Partitioning option, and most of the advanced performance and security packs. Plenty of businesses run SE2 happily for years. If your workplace runs SE2, the SQL, schema design, and PL/SQL in this book apply fully; some performance and operations chapters will mention features you do not have, and we flag them.

Enterprise Edition and its options

Enterprise Edition (EE) includes the full core feature set: parallel execution, online operations, Data Guard physical standby databases, flashback features, and more. On top of EE, Oracle sells **options** and **management packs**, each licensed separately. Examples you will meet in this book:

- **Partitioning**, a separately licensed EE option, covered in Chapter 14.
- **Diagnostics Pack**, required for AWR, ASH, and ADDM, covered in Chapter 24 (Statspack is a free alternative).
- **Tuning Pack**, required for the SQL Tuning Advisor and SQL Monitor, mentioned in Chapter 22.
- **Real Application Clusters (RAC)** and **Active Data Guard**, discussed in overview in Chapter 27.

> **WARNING**
> Oracle licensing is contractual, not technical. Many extra-cost features are installed and can be switched on even where they are not licensed, and the database records feature usage. Before you use a pack or option on a production system, check your organization's license with whoever owns it. This book will remind you at each point where it matters, but your own contract is the only authority.

Cloud: Base Database, Exadata, and Autonomous Database

Oracle also delivers the database as a service. On Oracle Cloud Infrastructure (OCI) you can rent a database that Oracle installs and patches while you keep administrative control (the Base Database Service), run on Oracle's engineered Exadata platform, or use **Autonomous Database**, where Oracle automates provisioning, patching, backups, scaling, and much of the tuning. Autonomous Database comes in workload-oriented flavors, such as transaction processing and data warehousing. Oracle has also made its database services available inside other public clouds through partnership offerings; the details change often, so check the current Oracle documentation if that is relevant to you.

For learners, OCI has offered an Always Free tier that has included small Autonomous Database instances; availability and limits can change, so check before relying on it. Autonomous Database gives no operating-system or SYSDBA access, so some administration exercises can only be observed there. Chapter 2 covers these alternatives.

The Release Model

Oracle numbers releases by year. Since 2018 the version number reflects the year of release (18c, 19c, 21c, 23ai), and the first two digits of the internal version string follow suit: an Oracle Database 19c system reports version 19, a 23ai system version 23. Within a release, Oracle ships quarterly **Release Updates** (RUs) that bundle security and bug fixes; a full version such as 19.x tells you which update level is installed.

Releases come in two kinds:

- **Long Term Releases** receive several years of Premier Support followed by extended support. They are what most production systems run. **Oracle Database 19c** is the widely deployed long-term release, and you will meet it at most employers for years to come.

- **Innovation Releases** (21c was one) deliver new features earlier, with a shorter support window. They are useful for trying features but are rarely chosen for long-lived production systems.

Oracle Database 23ai is the newest long-term release line covered in this book. It was first previewed under the name 23c and renamed 23ai, reflecting new AI-related capabilities such as the `VECTOR` data type and vector search. For everyday developers it also brings many quality-of-life improvements to SQL, several of which we use in later chapters and always mark as 23ai-only: a native `BOOLEAN` type in SQL, `IF [NOT] EXISTS` in DDL statements, `SELECT` without `FROM DUAL`, grouping by column alias or position, the `DB_DEVELOPER_ROLE` role, schema-level privileges, SQL domains, and JSON Relational Duality Views. Oracle Database Free is currently based on the 23ai line.

> **NOTE**
>
> Oracle has announced successor branding for its next long-term line: **Oracle AI Database 26ai**. This book does not make claims about its features. If your environment reports 26ai, the fundamentals here (SQL, architecture, transactions, indexing, backup) still apply; check Oracle's documentation for what is new or changed in your exact release.

Nearly everything in this book works on both 19c and 23ai; where a statement needs 23ai we say so.

Finding out what you are connected to

When you connect to an unfamiliar database, the first useful question is "which release and edition is this?" Two standard queries answer it. The `V$VERSION` view returns one or more banner lines that name the product, edition, and release, and the `PRODUCT_COMPONENT_VERSION` view returns the version numbers in separate columns:

version.sql

```
SELECT banner_full FROM v$version;

SELECT product, version, version_full
FROM   product_component_version;
```

The `BANNER_FULL` column exists from 18c onward; on older systems use `BANNER`. We do not reproduce the output here because the exact wording and update number depend on the image you download and the Release Update level, but you will see the edition name ("Free", "Enterprise Edition", and so on) and a version such as 19.x or 23.x. Run these queries in Chapter 2 once you are connected, and note the answer: it tells you which "23ai only" markers in this book apply to you.

Where Oracle Fits: A Fair Comparison

Many readers arrive from PostgreSQL, MySQL, or Microsoft SQL Server. All four are excellent relational databases, all speak SQL, and all can run serious production workloads. The differences lie in licensing, ecosystem, administration model, and a handful of behaviors that change how you write

code. The table below summarizes what matters to a developer; it is deliberately neutral, because the right choice depends on the organization, not on a feature checklist.

Topic	Oracle	PostgreSQL	MySQL	SQL Server
License	Commercial; Free edition	Open source	Open source and commercial	Commercial; free editions
Procedural language	PL/SQL	PL/pgSQL and others	Stored routines	T-SQL
Read consistency	Undo-based multiversioning	Row versions in table	Undo-based (InnoDB)	Locking by default; optional row versions
Schema vs user	A schema is a user's objects	Schemas separate from roles	Schema means database	Schemas separate from logins
Empty string	Treated as NULL	Distinct from NULL	Distinct from NULL	Distinct from NULL
DDL in transactions	Implicit commit	Transactional	Implicit commit	Mostly transactional

Where Oracle tends to be chosen

Oracle is commonly chosen when an organization needs a long-supported commercial product with a single vendor accountable for the whole stack; when it needs specific high-availability features such as Real Application Clusters or Data Guard; when it runs packaged applications certified on Oracle; or when it already has Oracle skills and tooling in place.

Where other databases tend to be chosen

PostgreSQL is often chosen for new projects that value an open-source license, a rich extension ecosystem, and low entry cost. MySQL is very common for web applications and has a large hosting ecosystem. SQL Server is a natural fit in organizations built around Microsoft tooling. Cost is a real factor: Enterprise Edition and its options are expensive, and teams sometimes choose another database simply because the license budget is not there.

Behaviors that change how you write code

If you are moving from another database, a few Oracle behaviors are worth learning on day one. Each is explained fully later; this list is a heads-up.

- **The empty string is NULL.** Inserting `''` into a `VARCHAR2` column stores `NULL`, so a `NOT NULL` column rejects it (Chapter 6).
- **DDL commits.** `CREATE`, `ALTER`, and `DROP` statements implicitly commit any open transaction before and after they run (Chapter 10).

- **No automatic commit in the database.** Oracle itself never commits on your behalf after a DML statement; some client tools and drivers do. SQL*Plus and SQLcl do not autocommit by default (Chapter 10).
- **Readers never block writers.** A query sees a consistent snapshot as of its start, reconstructed from undo, and does not lock rows it reads (Chapter 15).
- **Users own schemas.** Creating a user creates a schema of the same name; there is no separate `CREATE SCHEMA` to make a namespace (Chapter 4).
- **Identifiers are uppercase by default.** Unquoted names are stored in uppercase, so `hotels` becomes `HOTELS` in the data dictionary (Chapter 4).

> **TIP**
> If you are coming from another database, keep a short "Oracle differences" note while you read. Writing down each surprise in your own words, with the chapter number, gives you a personal cheat sheet you will reuse for years.

Meet Sablewood Hotels

Abstract examples are forgettable, so this book follows one small business from start to finish. **Sablewood Hotels** is a group of three hotels in three fictional towns. It is moving its bookings out of spreadsheets and into an Oracle database, and you will build, query, secure, tune, and back up that database alongside the three people who run the project.

- **Iris** is the operations director. She asks business questions: how much revenue did each hotel book in March, which rooms are idle, who are our repeat guests? Her questions drive the SQL chapters.
- **Tom** is the application developer building the booking system. He needs a clean schema, safe concurrent bookings, and fast queries from his application code. His needs drive the schema design, PL/SQL, and application chapters.
- **Priya** has just been appointed DBA, somewhat by accident, because she "knows databases". She must keep the system running, secure, and recoverable. Her questions drive the architecture, security, performance, and operations chapters.

Iris put the goal plainly at the kickoff meeting: "I want one place where a booking lives, and I want to trust the numbers." Tom added: "And two receptionists must never be able to sell the same room for the same night." Priya, looking at the list of things she now owned, asked the question every new DBA asks: "If the server dies at 3 a.m., what do we lose?" Each of those three sentences maps to a part of this book.

The data at a glance

The Sablewood database is deliberately small, so you can check every result by hand. It has six tables: `hotels`, `room_types`, `rooms`, `guests`, `reservations`, and `payments`. Chapter 4 creates them with a single script, `sablewood_setup.sql`. Here are the three hotels:

HOTEL_ID	CODE	NAME	CITY
1	HBR	Sablewood Harbor	Port Ellis
2	MDW	Sablewood Meadow	Ashby
3	CTY	Sablewood City	Dunmore

Each hotel has four rooms: two Standard rooms, one Deluxe room, and one Suite, twelve rooms in all. The room types carry the base rates the hotels charge:

TYPE_CODE	DESCRIPTION	BASE_RATE	MAX_GUESTS
STD	Standard	120.00	2
DLX	Deluxe	165.00	3
STE	Suite	240.00	4

One room, the Suite at Sablewood Meadow (room 8), is out of order, so eleven rooms are active. There are five guests from five countries, seven reservations in March 2026, and four payments. Three guests belong to the loyalty program: Amelia and Lena at the SILVER tier and Sofia at GOLD; loyalty members receive a discount (5% for SILVER, 10% for GOLD). When every non-cancelled reservation is totaled after discounts, the group has booked 2557.50 in March, and it has received 1634.00 in payments, leaving 923.50 outstanding. By the end of Chapter 8 you will compute all of those numbers yourself.

The rules that make it interesting

A hotel database looks simple until you write down the rules. Sablewood has three that recur throughout the book:

1. A room can be booked only if it is `ACTIVE`, and the check-out date must be after the check-in date.
2. A booking must not overlap another non-cancelled reservation for the same room. Checking out on the day another guest checks in is allowed. Formally, two stays overlap when `new.check_in < existing.check_out AND new.check_out > existing.check_in`.
3. Only reservations with status `BOOKED` can be cancelled.

The first rule can be expressed with ordinary constraints, which Chapter 11 discusses. The second cannot: no simple constraint can compare a row with other rows of the same table in this way. Enforcing it correctly when two receptionists click "Book" at the same moment requires

understanding Oracle's concurrency model (Chapter 15) and writing a small PL/SQL package that locks the room row while it checks (Chapter 17). That package, `sw_booking`, is the centerpiece of Part IV.

Users in the Sablewood database

The design uses three accounts: `SABLEWOOD` owns all objects and is not used for daily logins (Chapter 4), `SW_APP` is Tom's application login that may call the booking package and read views, and `SW_REPORT` may only read views. Chapter 20 creates the last two. This separation is called **least privilege**.

Reading the Error: Your First ORA- Message

Oracle reports problems with codes of the form **ORA-nnnnn**, followed by a short message. Learning to read them calmly is a core skill, so most chapters in this book include a walkthrough like this one. Suppose Tom, on his first day, connects to the database as his own login and runs a query against the hotels table before anyone has created it, or while connected as the wrong user:

first_query.sql

```
SELECT code, name FROM hotels;
```

Output

```
ORA-00942: table or view does not exist
```

- **What it says:** the name `HOTELS` could not be resolved to a table or view that this session can see.
- **Common causes:** the table really does not exist yet; it exists in another schema (for example it belongs to `SABLEWOOD` but Tom is connected as someone else); or it exists but the current user has no privilege on it. Oracle deliberately gives the same error in the last case, so it does not reveal the existence of objects you cannot access.
- **How to fix it:** check which user you are connected as, check whether the table exists in that schema (Chapter 4 shows the `USER_TABLES` view), and if it lives elsewhere either qualify the name (`sablewood.hotels`), create a synonym (Chapter 13), or have the owner grant access (Chapter 20).

The message says *what* failed, not *why*: list the possible causes and check them one by one.

Recent releases sometimes add help text or a documentation link after the classic message (Chapter 5 shows this); the code and first line are what you search for. Appendix C collects the errors used in this book.

How This Book Is Organized

The book has six parts. You can read it front to back, and that is what we recommend the first time, because each chapter builds on the Sablewood database as left by the previous ones.

- **Part I — Getting Started (Chapters 1–5).** Install Oracle Database Free in a container and connect with SQL Developer, SQLcl, and SQL*Plus (Chapter 2); understand the architecture well enough to reason about it (Chapter 3); create the `SABLEWOOD` user and load the data (Chapter 4); and refresh SQL the Oracle way (Chapter 5).
- **Part II — Oracle SQL in Depth (Chapters 6–10).** Data types, joins, aggregation and analytic functions, advanced queries, and changing data with transactions. This is where Iris's questions get answered.
- **Part III — Schema Design and Database Objects (Chapters 11–15).** Constraints, indexes, views, materialized views, sequences, synonyms, partitioning, and the concurrency model that makes Oracle distinctive.
- **Part IV — Programming and Connecting (Chapters 16–20).** PL/SQL essentials, the `sw_booking` package and triggers, JSON and modern SQL features, connecting from Java, Python, and Node.js, and security.
- **Part V — Performance (Chapters 21–24).** How Oracle executes SQL, reading execution plans, statistics and bind variables, and monitoring.
- **Part VI — Operations and Next Steps (Chapters 25–28).** Storage, backup and recovery, high availability and the cloud, and a final end-to-end production checklist for Sablewood.

PL/SQL appears in two chapters only, and only to the depth a database developer or DBA needs. thenexbooks publishes two dedicated PL/SQL books if you want to go further; we point to them where relevant rather than repeating their material. Appendices give you an SQL cheat sheet, a quick reference to data dictionary and `V$` views, a field guide to ORA- errors, and a glossary.

How each chapter works

Every chapter explains a concept with small examples, applies it to Sablewood, walks through a likely error, lists common mistakes, and ends with practice exercises and key takeaways. Query results appear as tables derived from the canonical Sablewood data, so you can check them against your own output. Execution plans show only the *shape* of a plan; cost and time figures depend on your system and statistics, so we replace them with `...`.

Conventions

SQL keywords are uppercase and identifiers lowercase. Dates in code use ANSI literals such as `DATE '2026-03-05'` (Chapter 6 explains why). File names above code blocks are scripts you can save and run; blocks captioned "Output" show what the tool prints. 23ai-only features and separately licensed features are marked in the text.

Applying It to Sablewood: Choosing a Platform

At the end of the kickoff, Iris asked the question that every project faces: "Which Oracle do we buy?" It is a good exercise in applying this chapter, so let us follow Priya's reasoning.

Sablewood is small: three hotels, a few dozen rooms in total once the real inventory is loaded, a handful of concurrent users, and a data volume measured in megabytes, not terabytes. It does not need clustering across several servers, and its reporting needs are modest. On paper, Oracle Database Free would hold the entire business within its limits. Priya's notes read:

- **Development and learning:** Oracle Database Free in a container on each developer laptop. No cost, identical SQL, easy to reset. Tom and Priya can break things freely.
- **Production, option A:** Standard Edition 2 on a modest server, with a standby copy maintained by backups and scripted recovery. Lower license cost; no Partitioning, no Diagnostics Pack.
- **Production, option B:** a managed cloud database, where patching and backups are handled by the provider and the team focuses on the application.
- **Not needed now:** Enterprise Edition with RAC. It solves problems Sablewood does not have.

Priya did not decide on the spot, and neither should you in a real project; cost, support, skills, and the hosting strategy all matter. What matters for us is that every option on her list runs the same SQL. The schema and code we build in this book move between Free, SE2, EE, and the cloud services with little or no change, provided we stay away from extra-cost features in production, or license them deliberately. That portability is one of the strongest arguments for learning Oracle on the Free edition.

Common Mistakes

- **Assuming Free means licensed everywhere.** Practicing with Partitioning or AWR in Free is fine; using them on a production Enterprise Edition system without the license is not. Always check before enabling a feature at work.
- **Confusing "database" and "instance".** Saying "the database crashed" when the instance stopped leads to wrong conclusions about data loss. The files persist; Chapter 3 explains how Oracle recovers.
- **Writing 23ai-only SQL for a 19c target.** `SELECT 1;` without `FROM DUAL` works on 23ai and fails on 19c. Know which release your code will run on.
- **Treating other databases' habits as universal.** Expecting the empty string to differ from `NULL`, or expecting DDL to roll back, causes real bugs in Oracle.
- **Choosing a platform by feature list alone.** Edition and deployment decisions are about cost, skills, support, and fit, not about which product has the longest list of capabilities.

Practice

1. **Edition fit.** A small clinic needs a database for appointment scheduling with about 10 GB of data, five users, and no clustering. List two editions or services that could fit and one question you would ask before deciding.

Solution: Standard Edition 2 on a small server, or a managed cloud database; Oracle Database Free could fit the data volume at the time of writing but leaves little headroom. Ask about growth, support requirements, and budget.

2. **Release markers.** Which of these need Oracle Database 23ai: (a) `CREATE TABLE IF NOT EXISTS`, (b) `SELECT SYSDATE FROM DUAL`, (c) a SQL column of type `BOOLEAN`, (d) `GROUP BY` a column alias?
   Solution: (a), (c), and (d) are 23ai features. (b) works in every release.
3. **Licensing.** Priya wants to use AWR reports on the production Enterprise Edition system. What must be true first, and what free alternative exists?
   Solution: The Diagnostics Pack must be licensed for that system; Statspack is the free alternative. Chapter 24 covers both.
4. **Sablewood data.** Using only the facts in this chapter, how many rooms does Sablewood have, how many are active, and how much revenue was outstanding at the end of March?
   Solution: 12 rooms (four per hotel), 11 active (room 8 is out of order), 923.50 outstanding.
5. **Reading an error.** Kenji's booking report fails with ORA-00942 although Tom is certain the `reservations` table exists. List three things to check.
   Solution: Which user the session is connected as; whether the table lives in another schema (try `sablewood.reservations`); whether the current user has been granted access to it.

Key Takeaways

- Oracle Database is a mature relational database you use through SQL and PL/SQL; the "database" is the files and the "instance" is the memory and processes that serve them.
- Editions differ in features, limits, and price: Free for learning and small use, Standard Edition 2 for smaller production, Enterprise Edition with separately licensed options and packs, plus managed cloud services including Autonomous Database.
- Oracle Database Free is limited at the time of writing to 2 CPU threads, 2 GB of RAM, and 12 GB of user data, and it includes features that are extra-cost elsewhere.
- 19c is the widely deployed long-term release; 23ai is the newest long-term line covered here, and 23ai-only features are marked throughout the book. Oracle has announced 26ai as successor branding.
- Oracle, PostgreSQL, MySQL, and SQL Server are all capable; choose by fit, cost, and skills. Learn Oracle's distinctive behaviors early: empty string is NULL, DDL commits, readers never block writers, users own schemas.
- Sablewood Hotels (Iris, Tom, and Priya) is the running example: three hotels, twelve rooms, five guests, seven reservations, four payments, and a no-overlap booking rule enforced in Chapter 17.
- Licensing is contractual: always check your own license before using a pack or option in production.