

Relational Database Design, Tools and Techniques: Hands-On - 4 Days

Course 382 Overview

You Will Learn How To

- Design, build and query a relational database
- Capture the structure of an existing database with a CASE tool
- Develop a data model to describe an application's data
- Apply normalisation to data for effective, stable database design
- Build a relational database from the logical database design
- Access data in a relational database using simple SQL queries

Course Benefits

Relational databases often drive company-critical and Web-enabled applications. Creating a database design that accurately and completely captures user requirements is vital for success. This course provides a comprehensive foundation for designing, building, and working with relational databases, enabling you to participate in the development process and to effectively use relational databases in your environment.

Who Should Attend

Anyone involved in designing, building and using relational databases, implementing database applications, or managing database development projects.

Hands-On Training

A continuing case study provides you with the skills to analyse, design, build and work with a relational database. Exercises include:

- Analysing an existing database with a CASE tool
- Developing data models
- Creating a logical data model that identifies entities, attributes and relationships
- Normalising data to create stable table structures
- Exploiting a CASE tool to generate SQL
- Building a database to correspond to a logical database design
- Constructing simple SQL queries to access the database

Relational Database Design, Tools and Techniques: Hands-On - 4 Days

Course 382 Outline

Introduction

An overview of DBMS technology

- Key concepts and terminology
- How data is accessed, organised and stored
- The importance of business rules
- Uses of databases
- The database development process

DBMS and related user tools

- Query languages
- Query and application development tools
- CASE tools for database analysis and design

How a Relational DBMS Works

Relational technology fundamentals

- The structure of a relational database
- Tables, attributes and relationships
- Primary and foreign keys
- Relational integrity constraints
- Manipulating data: selection, projection, join, union, intersection, difference

Components of a relational DBMS

- An integrated, active data dictionary
- The query optimiser
- An engine that manages the data
- Front-end tools for easy user access

Designing Relational Databases

A step-by-step approach and techniques

- Developing the logical data model
- Mapping the data model to the relational model
- Specifying integrity constraints
- Defining the data in the data dictionary

Entity-relationship modelling

- Capturing entities, attributes and identifiers
- Describing relationships: one-to-one, one-to-many, many-to-many
- Optional and mandatory relationships
- Resolving many-to-many relationships for implementation
- Guidelines for a well-formed E-R diagram

Normalising data to design tables

- Why/why not normalise
- Avoiding update anomalies
- Identifying functional dependencies
- Applying rules for normalisation

Working with a CASE tool

- Database design and documentation
- Generating the SQL to build the database

- Reverse engineering to capture the design of an existing database

Physical database design

- Grouping and assigning tables to disc files for performance and maintenance
- Fragmenting large tables
- Planned denormalisation vs. accidental denormalisation
- Indexing for performance and integrity

Building and Querying a Relational Database

Fundamentals of SQL

- A dynamic and evolving language
- ANSI and ISO standards

Creating a relational database

- Defining the database and its objects: tables, keys, views and indexes
- Declaring integrity constraints
- Altering structures and constraints

Accessing the database with SQL

- Querying the database to retrieve exactly the desired information
- Joining tables to retrieve related data
- Updating data while maintaining database integrity

Constructing and using views

- Defining views for simplicity and security
- Customising windows into the database
- Querying and updating through views

The Future of Database Design

Enforcing business rules for data integrity

- Defining declarative constraints
- Server-side programming in Java or a procedural language

Trends

- Modelling in analysis and design
- Focusing on business rules
- Creating an intelligent server
- Using stored procedures and triggers