CSC 222

DATABASE DESIGN AND MANAGEMENT I

University of Ilorin
Centre for Open &
Distance Learning

CODL

Published by the Centre for Open and Distance Learning,
University of Ilorin, Nigeria
E-mail: codl@unilorin.edu.ng
Website: https://codl.unilorin.edu.ng

This publication is available in Open Access under the Attribution-ShareAlike-4.0 (CC-BY-SA 4.0) license (https://creativecommons.org/licenses/by-sa/4.0/).

By using the content of this publication, the users accept to be bound by the terms of use of the CODL Unilorin Open Access Repository(OAR).

COURSE DEVELOPMENT TEAM

Content Authoring

Oluwasogo S. Ayodeji (Ph.D)

Language Editor

Bankole O. Ijeoma

Course Editor

Oladele J. Omalewa (Ph.D)

Instructional Designer

Olawale S. Koledafe

Damilola Adesodun

Ismail O. Olanrewaju

Makinde Gbemileke

From the Vice Chancellor

Courseware development for instructional use by the Centre for Open and Distance Learning (CODL) has been achieved through the dedication of authors and the team involved in quality assurance based on the core values of the University of Ilorin. The availability, relevance and use of the courseware cannot be timelier than now that the whole world has to bring online education to the front burner. A necessary equipping for addressing some of the weaknesses of regular classroom teaching and learning has thus been achieved in this effort.

This basic course material is available in different electronic modes to ease access and use for the students. They are available on the University's website for download to students and others who have interest in learning from the contents. This is UNILORIN CODL's way of extending knowledge and promoting skills acquisition as open source to those who are interested. As expected, graduates of the University of Ilorin are equipped with requisite skills and competencies for excellence in life. That same expectation applies to all users of these learning materials.

Needless to say, that availability and delivery of the courseware to achieve expected CODL goals are of essence. Ultimate attention is paid to quality and excellence in these complementary processes of teaching and learning. Students are confident that they have the best available to them in every sense.

It is hoped that students will make the best use of these valuable course materials.

Professor S. A. Abdulkareem

Vice Chancellor

Foreword

Courseware remains the nerve centre of Open and Distance Learning. Whereas some institutions and tutors depend entirely on Open Educational Resources (OER), CODL at the University of Ilorin considers it necessary to develop its own materials. Rich as OERs are and widely as they are deployed for supporting online education, adding to them in content and quality by individuals and institutions guarantees progress. Doing it in-house as we have done at the University of Ilorin has brought the best out of the Course Development Team across Faculties in the University. Credit must be given to the team for prompt completion and delivery of assigned tasks in spite of their very busy schedules.

The development of the courseware is similar in many ways to the experience of a pregnant woman eagerly looking forward to the D-day when she will put to bed. It is customary that families waiting for the arrival of a new baby usually do so with high hopes. This is the apt description of the eagerness of the University of Ilorin in seeing that the centre for open and distance learning [CODL] takes off.

The Vice-Chancellor, Prof. Sulyman Age Abdulkareem, deserves every accolade for committing huge financial and material resources to the centre. This commitment, no doubt, boosted the efforts of the team. Careful attention to quality standards, ODL compliance and UNILORIN CODL House Style brought the best out from the course development team. Responses to quality assurance with respect to writing, subject matter content, language and instructional design by authors, reviewers, editors and designers, though painstaking, have yielded the course materials now made available primarily to CODL students as open resources.

Aiming at a parity of standards and esteem with regular university programmes is usually an expectation from students on open and distance education programmes. The reason being that stakeholders hold the view that graduates of face-to-face teaching and learning are superior to those exposed to online education. CODL has the dual-mode mandate. This implies a combination of face-to-face with open and distance education. It is in the light of this that our centre has developed its courseware to combine the strength of both modes to bring out the best from the students. CODL students, other categories of students of the University of Ilorin and similar institutions will find the courseware to be their most dependable companion for the acquisition of knowledge, skills and competences in their respective courses and programmes.

Activities, assessments, assignments, exercises, reports, discussions and projects amongst others at various points in the courseware are targeted at achieving the objectives of teaching and learning. The courseware is interactive and directly points the attention of students and users to key issues helpful to their particular learning. Students' understanding has been viewed as a necessary ingredient at every point. Each course has also been broken into modules and their component units in sequential order.

At this juncture, I must commend past directors of this great centre for their painstaking efforts at ensuring that it sees the light of the day. Prof. M. O. Yusuf, Prof. A. A. Fajonyomi and Prof. H. O. Owolabi shall always be remembered for doing their best during their respective tenures. May God continually be pleased with them, Aameen.

Bashiru, A. Omipidan
Director, CODL

Course Goal

Database Management has evolved from a specialized computer application to become a central component of modern computing environment and as a result, knowledge about database systems is needed as it has become an essential part of the Computer Science curriculum. The need was determined from the primary goal of database systems, which is to provide way to store and retrieve database information conveniently and efficiently.

Requirements for Success

The CODL Programme is designed for learners who are absent from the lecturer in time and space. Therefore, you should refer to your Student Handbook, available on the website and in hard copy form, to get information on the procedure of distance/ e-learning. You can contact the CODL helpdesk which is available 24/7 for every of your enquiry.

Visit CODL virtual classroom on http://codllms.unilorin.edu.ng. Then, log in with your credentials and click on CSC 222. Download and read through the unit of instruction for each week before the scheduled time of interaction with the course tutor/facilitator. You should also download and watch the relevant video and listen to the podcast so that you will understand and follow the course facilitator.

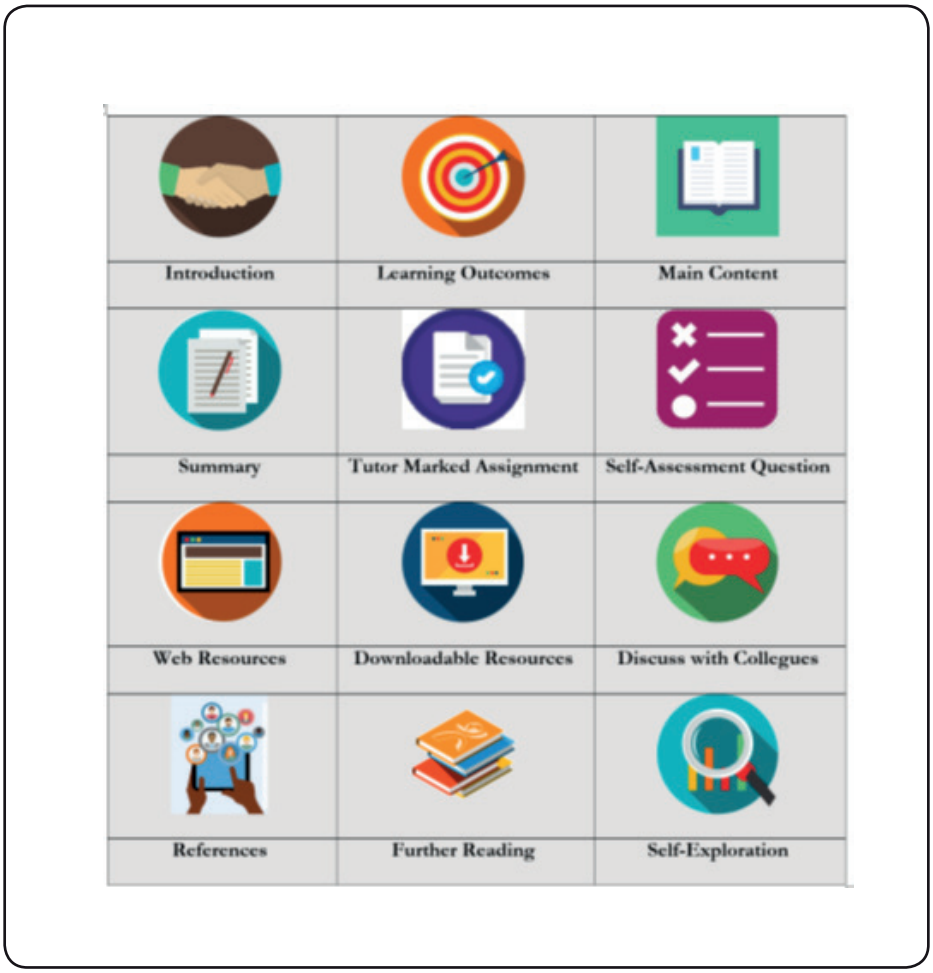
At the scheduled time, you are expected to log in to the classroom for interaction.

Self-assessment component of the courseware is available as exercises to help you learn and master the content you have gone through.

You are to answer the Tutor Marked Assignment (TMA) for each unit and submit for assessment.

Embedded Support Devices

Throughout your interaction with this course material, you will notice some set of icons used for easier navigation of this course materials. We advise that you familiarize yourself with each of these icons as they will help you in no small ways in achieving success and easy completion of this course. Find in the table below, the complete icon set and their meaning.



WORK PLAN

Learning Outcomes

When you have studied this course, you should be able to:

- To organize a File Processing Environment.
- Differentiate between field, record and file.
- Examine the concept of Normalization in Relational Database Systems.
- Manage Database Systems.

COURSE GUIDE

MODULE 1

FILE MANAGEMENT SYSTEMS

SUB UNITS

UNIT 1: File processing systems
UNIT 2: Field, Record and File

MODULE 2

DATABASE AND DATABASE MANAGEMENT SYSTEM

SUB UNITS

UNIT 1: Dataase
UNIT 2: Database Management systems (DBMS)
UNIT 3: Data Users and Administrators

MODULE 3

DATA MODELS

SUB UNITS

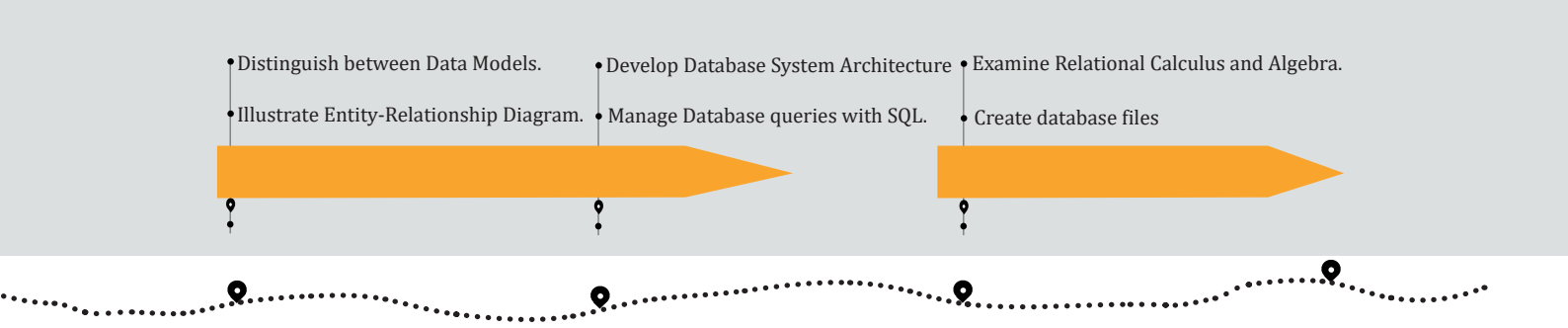
UNIT 1: Database Instances and Schemas
UNIT 2: Netweork,Hierarchical, Relational Models And Object-relational Models.
UNIT 3: Entity-Relationship Models
UNIT 4: Domains, Attributes, Tuples And Relations

Course Guide

Course code: CSC 222
Course Title: Database Design and Management I Credit (C): 3

Prerequisite Course(s)

CSC 112 – Introduction to Computer Science II (C) 2.
Related Courses
CSC 214 – Introduction to File Processing. Credits (C) 2.
CSC 226 – Computer Appreciation III. Credits (E) 2.



MODULE 4

DATABASE DESIGN

SUB UNITS

UNIT 1: Functional Dependencies

UNIT 2: Decomposition

UNIT 3: Normalization

MODULE 5

QUERY LANGUAGES

UNIT 1: Structural Query Language (SQL)

UNIT 2: Relational Algebra And Calculus

UNIT 3: Query Processing

UNIT 4: Query Optimization

MODULE 6

STUDY OF SOME STANDARD DATABASE SYSTEMS

SUB UNITS

UNIT 1: Microsoft Access

UNIT 2: Oracle

UNIT 3: MYSQL

UNIT 4: SQL SERVER

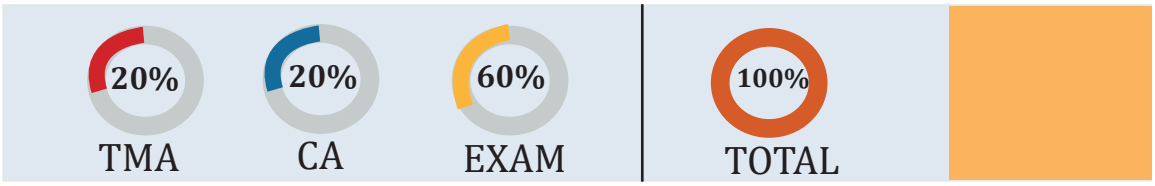
INTRODUCTION

Database Design and Management I is a course that is taken in the second year of the Computer Science programme. CSC 222 (Database Design and Management I) is a three (3) credit unit course. It is a prerequisite to CSC 214 (Introduction to File Processing). The course is compulsory for students offering Bachelor of Science, (B.Sc.), Computer Science, Information Systems and Allied degrees. This implies that it must be taken and passed. The main objective of the course is to deal with the fundamental concepts of Database Design and Management with specific focus on designing, building and utilization of databases. It also provides a general overview of the nature and purpose of Database Systems. The course explain how the concept of Database System was developed, what the common features of Database Systems are, what a Database System does for the user, and how Database Systems are implemented. Various Database Systems are also studied.

The course is organized into six (6) distinct modules with each module addressing a major component of the course and made up of two or more units. In computerized information systems, data is the basic resource of the organization. So, proper organization and management of data is required for organizations to run smoothly. Database management system deals with the knowledge of how data are stored and managed on a computerized information system. In any organization, it requires accurate and reliable data for better decision making, ensuring privacy of data and controlling data efficiently.

Assignments and Grading

Beyond the regular classroom attendance weight will be given to assignments and final examination as follows:



REFERENCES

Silberschatz A., Korth H. F., and Sudarshan S. (2003). Database System Concepts. McGraw-Hill Education.

Begg C. E. and Connolly T. M. (2002). Database Systems: A Practical Approach to Design, Implementation, and Management, Addison-Wesley.

Chao L. (2006). Database Development and Management. Taylor & Francis Group, LLC. ISBN 0-8493-3318-0

Ozzu M. T. (2012). Overview of Database Management. David R. Cheriton School of Computer Science, University of Waterloo.

Kahate A. (20040. Introduction to Database Management Systems. Pearson Education.

ISBN 9788131700785, eISBN 9788131775820

Conger S. (2012). Hands-on Database design and Development. Pearson education. ISBN 13: 978-0-13-610827-6, ISBN 10: 0-13-610827-X

Ramakrishnan R. & Gehrke J. ( ). Database Management Systems. Second edition. McGraw-Hill Higher Education. ISBN 0-07-266535-2

Robinns R. J. (1995). Database Fundamentals.

Frost R., Day J. & Slyke C. V. (2006). Database Design and Development. A Visual Approach. Pearson Prentice Hall™. ISBN 0-13-035122-9.

Radvanyi T. (). Database Management Systems.

Elmasri R. Navathe s. B. (2011). Fundamentals of Database Systems. Pearson. ISBN 10: 0-136-08620-9.

Ponniah P. (2003). Databasr Design and Development. An Essential Guide for IT Professionals. IEEE Press. John Wiley & Sons inc. Publication. ISBN 0-471-21877-4.

Gupta S. B. & Mittal A. (2017). Introduction to Database Management System. Second Edition. University Science Press. ISBN 978-93-81159-31-6.

Rich Maclin, Database Management Systems, Chapter 1. rmaclin@d.umn.edu

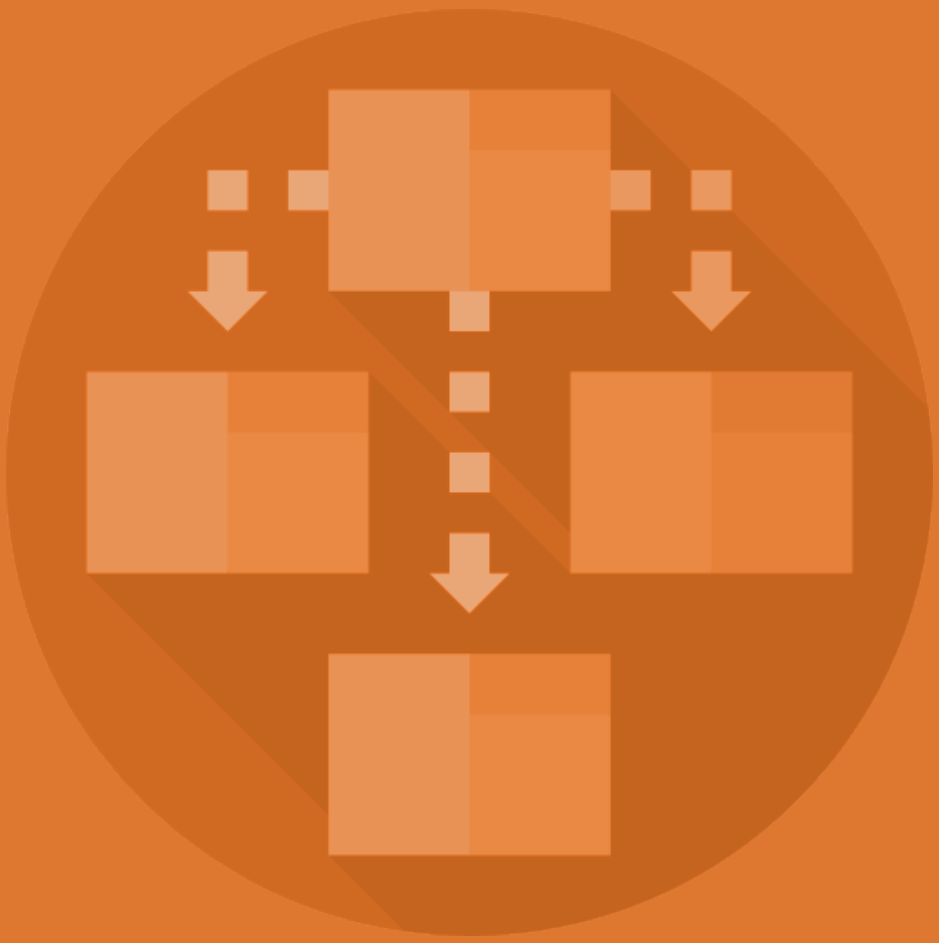
https://Tutorialspoint.com/oracle_sql/index.asp

Data notepads on a table

source: Unsplash by Stil

Module 1

FILE MANAGEMENT SYSTEMS



UNITS

UNIT 1: File processing systems

UNIT 2: Field, Record and File

picture by Pixabay
source: Pexels

UNIT 1

UNIT 1: FILE PROCESSING SYSTEM

Introduction

This section explains the concept of file systems. It explains how the manual filing system was used to initiate the automated file processing system.Learning Outcomes

Learning Outcomes

When you have studied this unit, you should be able to:

- Explain File Systems
- Provide background knowledge of File Processing System (File-Based System)
- Mention the uses of File Processing System
- List the limitations of the file-based syste
- Identify the file processing environment.

picture by Pixabay
source: Pexels

Main Content

File System [SAQ1, 3]

READING TIME: 2 MINS

You should bear in mind that data is the core of any business or organization. Data is transformed to information. We derive information daily transactions in businesses, organizations and banking industries on a daily basis. Information is stored in files for the purpose of record keeping, making updates and for making further analysis that will enhance decision making.

File Systems are similar to the traditional (paper) files. They both allow records to be inserted, updated, deleted, retrieved, searched for, and sorted, and so on. You should take note of the example of a file shelf is shown in figure 1.

Figure 1: File Shelf
(Source: Kahate, 2004)

I want to point out that the manual filing System entails storing records and information in paper files which are stored in File Cabinets and shelves in offices. To access information in manual files, the targeted file has to be located and opened in order to capture the needed information.

The Manual Filing System is known as the traditional way of storing records and files in offices and organizations. Retrieval of information from manual files is quite cumbersome due to the fact that individual files have to be opened one after the other in order to be able to capture vital information about individuals and

entities as the case may be. However, the manual filing system is prone is prone to loss of information, files may be stolen or damaged due to fire outbreak or flood. Some of these limitations brought about the need for file processing system which is an attempt to computerize the manual filing system. Let us take this example. In hospitals, a patient have a unique file number which is used to locate the needed file when a patient visits the hospital. Patients' files are arranged serially in file cabinets for easy access when they are needed. If the file is not placed in the right location based on the serial arrangement, then it cannot be located with ease.

Need for a File

We use a lot of files in carrying out our day-to-day activities. For example, a college student uses notebooks and journals while a workplace contains many files and registers. Why are files needed at all? The simple explanation is that we cannot remember each and every word and number that is important to us. Hence, we store information in the form of files, notebooks and so on as shown in figure 2.

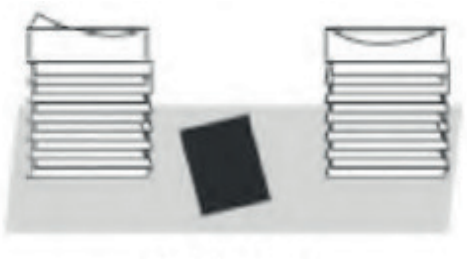


Figure 2: Files (Source: Kahate, 2004)

This then brought about the need for an automated filing system that is based on computer technology. This has led to the advent of the File Processing System.

Background of File Processing System

READING TIME: 2 MINS

I should let you know that the File Processing System is also known as File-based System. It is an attempt to automate the manual filing system. Consider a Banking Enterprise that stores information about all customers and their respective savings and current accounts. The information can be stored in permanent system files. The typical file processing system is supported by a conventional operating system. Permanent records are stored in various files and different application programs are written to extract records from and to add records to the appropriate files.

Before the advent of DBMSs, organizations typically stored information using such systems.

Definition: File Processing System is a collection of application programs that perform services for end users such as the production of reports. Each program defines and manages its own data.

You should be aware that in a typical file processing environment, each user area or unit, such as payroll, personnel and COMSIT has its own collection of files and programs that access these files.

In this type of environment, there is usually overlap of data between units (user areas). Hence, there is redundancy in the system. The address of a personnel can occur in many places, this certainly wastes a lot of space and the problem it causes for updates are potentially much more serious.

Let me add this as well, trying to produce reports or respond to queries that span or cut across units can be extremely difficult. This problem led to the idea of a pool of data, or database, rather that separate collections of individual files.

Definition: A database is a structure that can be house information about multiple types of entities as well as relationships among the entities. This imply that more than one user has access to the dat stored in a database.

Uses of File Processing System [SAQ2]

I computed below some of the uses of File Processing systems are listed as follows:

- Storage of data and records
- Update of records in files
- Retrieval of Information
- Preparation of reports
- Insertion/deletion of records in files

Storage of data and records – You should know that raw data and records are stored in files. The purpose of a database system is to retrieve information from and store new information in the database.

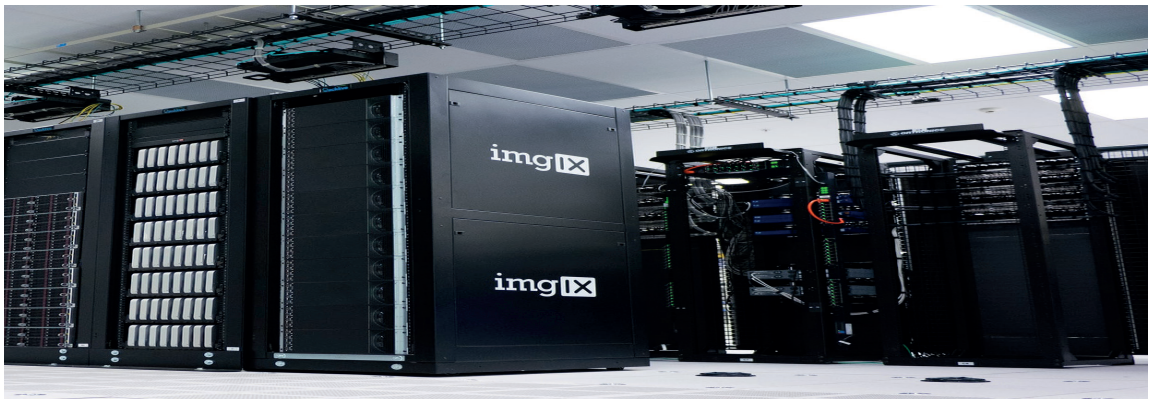
Update of records in files – Records are updated or changed as the status of an entity (i.e a person, place or thing) changes over time.

Retrieval of Information – We retrieve information from the records stored in database files as the need arises. Queries are sent into database files in order to obtain information. Your matriculation number, identity number or account number can be used to retrieve information directly from a database file.

Preparation of reports – I should inform you that reports are generated from records stored in database files. Queries are sent into database files and entities that fall into these categories are listed from the database file in order to generate report for decision making.

Insertion/deletion of records in files – Beware that new records are inserted when additional records of entities or persons are added to a database file. Records are also deleted when they are no longer needed in the database file. This include, deletion of the name of a personnel who has retired from the payroll database.

‘



imgix
source: Pexels

Limitations of the File Processing System

READING TIME: 2 MINS

- Separation and Isolation of data
- Duplication of data
- Data Dependence
- Incompatibility of file formats
- Fixed Queries/Proliferation of application program

Separation and Isolation of data – When data is isolated in separate files, it is more difficult to access data that should be available.

Duplication of data – Bear in mind that due to the decentralization approach taken by each department, the file-based system encourages the uncontrolled duplication of data. Duplication is wasteful. It cost time and money to enter the data more than once. It takes up additional storage space, again with associated cost. Often, the duplication of data can be avoided by sharing data files. Duplication can lead to loss of data integrity. Hence the data is no longer consistent.

Data Dependence – The physical structure and storage of the data files and records are defined in the application code. This means that changes to an existing structure is difficult.

Incompatibility of file formats – As the structure of files are embedded in the application programs, the structure is dependent on the application or programming language. Let us take for an example, the structure of a file generated by a COBOL program will be different from the structure of a file generated by a C program. The direct incompatibility of such files makes them difficult to process jointly. It is also costly and time consuming.

Fixed Queries/Proliferation of application programs – From the end user's point of view, file-based systems proved to be a great improvement over manual systems.

Consequently, the requirement for new or modified queries grew. However, the file-based systems are very dependent upon the Application Programmer. Any queries or reports that are required have to be written by the Application programmer. As a result, two things happened: In some organizations, the type of query or report that could be produced was fixed. There was no facility for asking unplanned queries either about the data itself or about which types of data are available. In others, there was a proliferation of files and application programs. Eventually this reached a point where the work could not be handled by the Data Processing Department

File Processing Environment

READING TIME: 1 MIN

I should let you know as well in a typical file-processing environment, each user area or unit, such as payroll, personnel and COMSIT has its own collection of files and programs that access these files. Let us study the Figure 3 below.

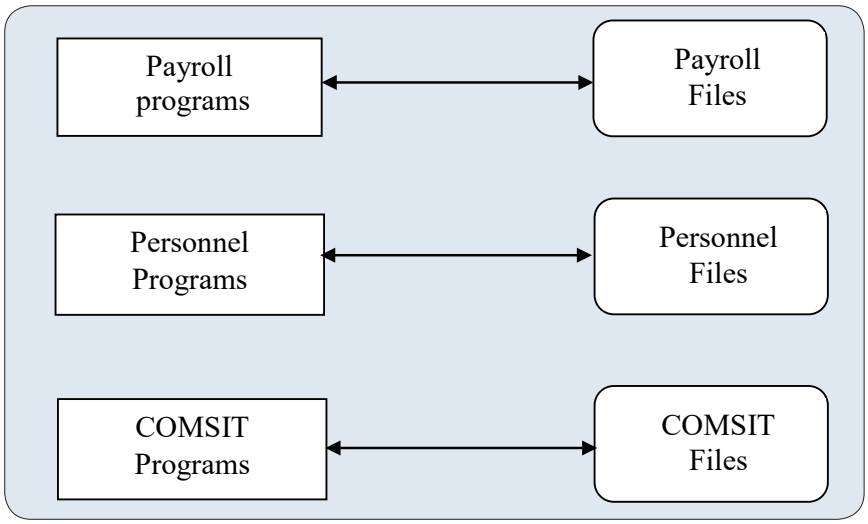


Figure 3: Classical file approach (Source: Silberschatz, Korth & Sudarshan, 2003)

Try to understand that in this type of environment there is usually overlap of data between units (user areas). Hence, there is redundancy in the system. The address of a faculty member can occur in many places. This certainly wastes a lot of space and the problems it causes for updates are potentially much more serious.

I should include in addition that, trying to produce reports or respond to queries that

span or cut across units can be extremely difficult. This problem lead to the idea of a pool of data, or database, rather than separate collection of individual files.

Summary

In this unit, we have examined the classical filing system and we explained the file processing system as a means of computerizing the manual filing system. The uses of file processing system was also discussed. We stated its limitations of the traditional filing system approach. A description of the file processing environment was also explained.

Self-Assessment Questions

1. What is File Processing?
2. Mention five (5) uses of File processing.
3. Distinguish between the Manual Filing System and the File processing System.

Tutor Marked Assignment

What is Record Key?

With the aid of a diagram, explain the differences between the two 92) types of Record Keys.

References

Elmasri R. and Navathe S. B., (2011). Fundamentals of Database Systems.

Gupta S. B. and Mittal A., (2004). Introduction to database management systems. PearsonEducation.

Ramakrishnan R. and Gherke J. (1996). Database Management System, second edition.

McGraw-Hill Higher Education.

Further Reading

en.wikipedia.org/wiki/Database_management_system.

files and records
Source: Pexels(Mike)

UNIT 2

FIELD, RECORD AND FILE

Introduction

Let me inform you that we widely use files in all serious business applications and organizations for the purpose of record keeping and data analysis. Records are stored into files. This can be paper file or computer file. Furthermore, you should note that files are used to store data, records and information. Records and information stored in files can be updated, retrieved, deleted, sorted and archived for future use.

Learning Outcomes

When you have studied this unit, you should be able to:

- Define Fields, Records and Files
- Compare and contrast between Master and Transaction File
- Identify Computer Files

files and records
Source: Pexels(Mike)

Main Content

Fields, Records and Files [SAQ1]

READING TIME: 1 MIN

First, I should let you be informed that in a file processing environment the critical terms are field, record and file. The smallest amount of data that can be stored is the bit. Bits are grouped into bytes or characters).

Collection of characters form a field. A field is an item of information. For example; SURNAME, FIRST_NAME, LAST_NAME, SEX, AGE and so on.

A collection of fields forms a record. Collections of occurrences of a specific type of record form a file. A file is a collection of records. A file is also a collection of logically related data. Figure 4 shows Employee records in a file.

Files

Let us assume that the clerk has decided

to store the information in the form of files.

For understanding the process of filing better, let us discuss it in detail.

notepad
Source: Unsplash

Sample File

sample files
source: Unsplash (Maksim tarasov)

Let us imagine that the clerk is about to start preparing the pay-slips for this month. As we had seen, the clerk now uses a file to store information such as name, basic pay, days-worked etc. Suppose the company has 1000 employees. There will be one page of information for each of the 1000 employees, as shown in figure 4. Thus, there will be 1000 records for the clerk to consult while preparing the pay-slips.

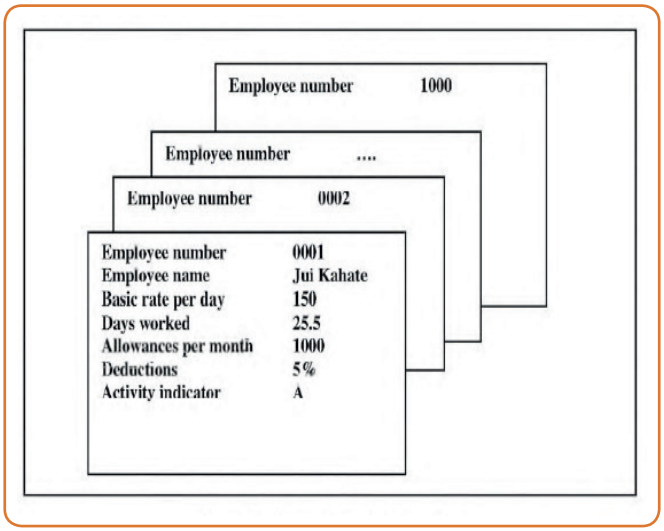


Figure 4: Employee records in a file (Source: Kahate, 2004)

From figure 4, we can deduce that Employee number, Employee name, Basic rater per day, Days worked, Allowances per month, Deductions and Activity indication are fields for capturing information about entities into a file.

The entries 0001, Jui Kahate, 150, 25.5, 1000, 5%, A with respect to each field is the record for the employee with the **Employee number, 0001.**

Employee number, 0002, and Employee number, 1000 are records of employees

with employee number 0002 and employee number 1000 respectively.

All the collection of records 0001, 0002, …., 1000 form a file as shown in figure 4.

As shown in figure 4, you should note that the clerk stores the information related to various employees and their pay details (i.e. employee records) in this file prior to the calculation of final pay. Hence, the clerk calls this file employee file. In other words, this is the file name of this particular file. We need to write an algorithm to prepare the pay-slips in the manual system. The creation of a file would not only simplify the clerk's task, but would also help the person who takes over in his absence. The next few sections build up towards that goal

Records and Fields

We have seen that in the file there is one page or record per employee. Each of these records contains details such as employee number, name etc. Such a detail is called a field or data item of the record. A sample record for Jui lists all the available fields. Table 1 shows the skeleton of a record, and outlines the information that can be stored in it. This skeleton is also called a record layout. Generally, it contains sections such as field name, type of data that can be stored (alphabets, numbers or both) and maximum allowable length of each field.

Table 1: Record layout for the Employee file

Field name	*Type of data*	*Maximum length*
Employee number	N	4
Employee name	X	30
Basic rate per day	N	4
Days worked	N	2 + 1
Allowances per month	N	4
Deductions percentage	N	3
Activity indicator	A	1

Let us note our observations regarding field name, and the type and maximum size of

data that can be stored.

0 We have used some symbols for the type of data allowed, as follows:

N means only numeric data is allowed

A allows only alphabets

X allows both numbers and alphabets

You should bear in mind that in the "maximum length" column, the days worked can contain a fraction. In Table 1, the days worked are 2 + 1. This means that it has two integer positions and one decimal, for example, 23.5 or 12.2. The maximum value in this field can be 99.9. However, in real life, can an employee work for 99.9 days in a month? The specification should, therefore, validate that the entry for this field should not exceed a maximum value of, say 31.0 days, depending on the month for which the record is being created. However, as of now we will not check for this condition so that we keep our observations simple.

500 naira notes
sources: The Africs Report

It is always a good idea to make provisions for the future. So, the basic per day section has been designed to allows up to 4 digits. Let us suppose that the basic pays are currently in the range 100 to 500. But sooner or later, an employee might have a basic pay of 1000. Unless we provide for an extra digit, the bigger figure cannot be stored.

Master and Transaction File (2mins) [SAQ2]

READING TIME: 1 MINS

Let me inform you that we can classify data into two types - master and transaction. Master data does not change with time. Some examples are employee number and employee name. Days worked and allowances are transaction data, which can change from time to time.

Master and transaction data are kept as separate files with reference to Table 1, if some changes have occurred during the month (e.g. recruitment of new employees

resignations or increments) we go through a master maintenance process to keep the master data up-to-date. All the transactions that take place during the month are collected separately in a transaction file. Finally, both files are read and the payroll is processed. In Table 1, we have shown an employee record in which some data is master data and some is transaction data. This kind of mixed data is usually available after both the master and transaction files have been read. The date is extracted after matching the records from both the files for the same employee number, and is further used for the payroll. The process is illustrated in Figure 5.

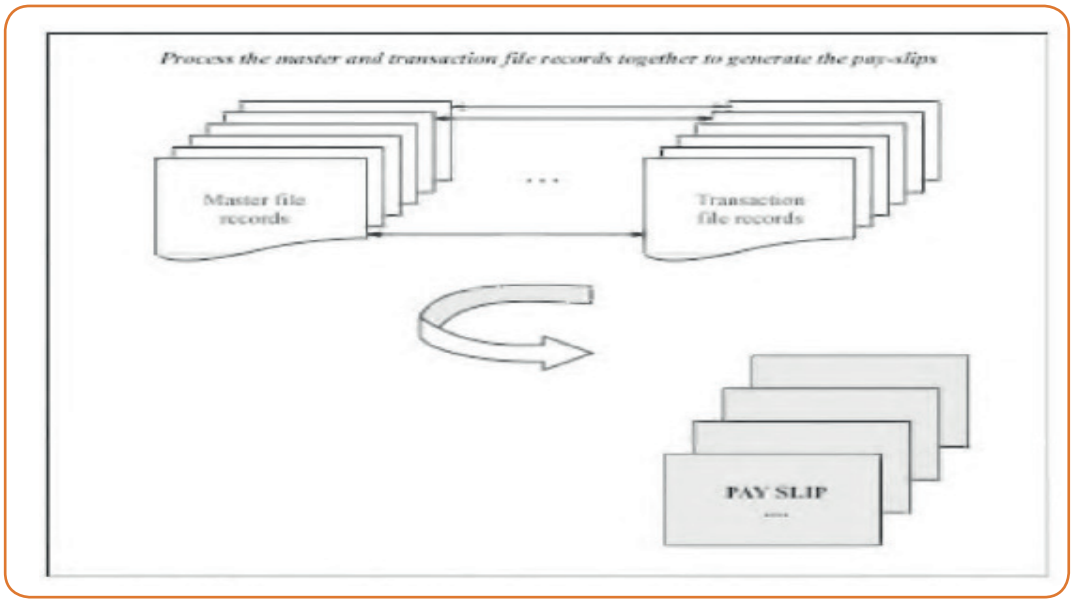


Figure 5: Processing master and transaction files to generate pay-slips
(Source: Kahate, 2004)

Computer Files

READING TIME: 3 MINS

Maintaining records and files in the paper format is convenient. It allows easy access to information and does not require any formal training. However, it is far easier and much more convenient to create and maintain files by using computers. A Computer File contains information arranged in an electronic format.

I should add that in concept, computer files are not significantly different from paper files. Computer files also facilitate easy storage, retrieval, and manipulation of data. I should let you know thus that the major difference between paper files and computer

files, however, is that while the former is stored on paper the latter is stored in the form of bits and bytes. Computer files also contain information similar to paper files. A computer file has a name. Thus, if we store employee information in a computer file, we may call it the employee file. The computer would recognize the file based on this name. A programmer working with this file can give instructions to the computer to open the file, read from it, write to it, modify its contents, close it, and so on.

For example, for payroll processing, the steps canned out by this program (i.e. the algorithm) can be summarised as shown in Figure 6 below. We have assumed that two files are available:

The employee file is the master file, containing records of all the employees. The payroll file is the transaction file, which would contain payroll details such as number of days worked and pay details of all the employees after appropriate calculations.

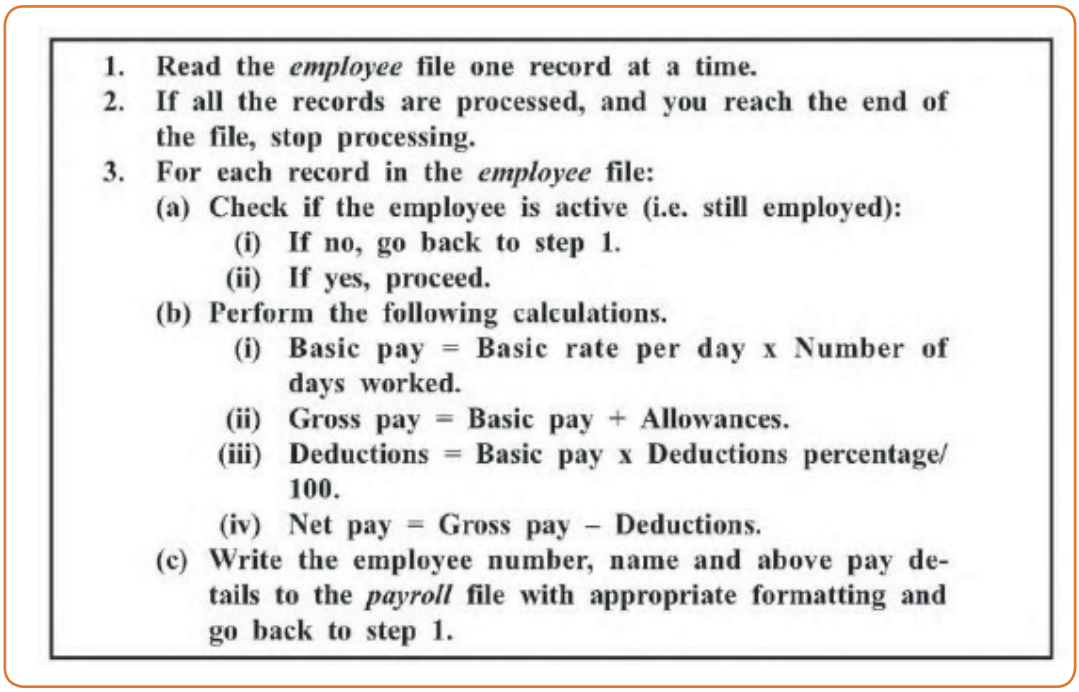

1. Read the *employee* file one record at a time.
2. If all the records are processed, and you reach the end of the file, stop processing.
3. For each record in the *employee* file:
   (a) Check if the employee is active (i.e. still employed):
      (i) If no, go back to step 1.
      (ii) If yes, proceed.
   (b) Perform the following calculations.
      (i) Basic pay = Basic rate per day x Number of days worked.
      (ii) Gross pay = Basic pay + Allowances.
      (iii) Deductions = Basic pay x Deductions percentage/ 100.
      (iv) Net pay = Gross pay – Deductions.
   (c) Write the employee number, name and above pay details to the *payroll* file with appropriate formatting and go back to step 1.

Figure 6: Payroll Processing Algorithm

If we look at the payroll-processing program carefully, we will notice that once it starts executing, the program needs very little interaction on the part of the clerk.

Suppose there is some sort of mechanism by which the following can be achieved: The input records are put (read) into the input area one by one as desired from the external medium (in this case, the employee file stored on the disk).

1. Calculations are done for the record read.
2. The output pay-slips are put (written) into the output area (that is the payroll file).

Then the clerk would virtually do nothing until all the pay-slips were printed and filed for distribution. This is similar to a factory where things keep on moving over a belt. The entire cycle starts with the input of spare parts and ends with the production of the end product. Everything moves over a belt, with processes involved at many intermediate stages. This type of processing is called batch processing. Batch processing involves a batch of goods manufactured and passed along the conveyer belt. Here, the entire batch of input employee records moves over the payroll program. Calculations are performed in

each case. The output of the process is the set of pay-slips. In batch processing, no or minimum human interaction is required. One program passes control to another in a sequence.

However, I should include that in contrast to batch processing, in many situations the program needs to be conversational. That means, it needs to interact with the user of the program. Take the example of a manual telephone directory service. You tell the operator the name of the person and ask for the corresponding telephone number. The operator on the other side searches for this information and comes back with an answer. These days, the computer performs both the searching and the answering operations in an automated manner. A search that can take place at any time is called as an online query. When an instantaneous answer is expected, it is called online processing or real-time processing.

An example of real-time processing is seen in the case of airlines reservations. Note that it is random, which means that there could be any query at any time. In such a case, how do we organize information to enable easy, faster access to it? We shall study this in subsequent sections of the book.

Searching Records

READING TIME: 3 MINS

LIBRARY MANAGEMENT - A CASE STUDY

We shall now use another example to illustrate the more advanced concepts of file processing.

Imagine that we have a library of books and have appointed a librarian to maintain it. The librarian has created one card per book, which contains details such as book number, title, author, price and date of purchase. For this, the librarian has used the conceptual record layout. For simplicity, we have not shown the part of the card containing details on borrowing and returning of books. We shall ignore these details throughout the discussion. As soon as a new book comes in to the library, the librarian creates a new card for it.

Note that a card is similar to a record and, in technical terms, the entire pile of cards is similar to a file. We shall use the terms interchangeably.

Record Keys

One pertinent question is: why do we need additional data elements such as an employee number in the case of an employee record and a book number in a book record? The reason is quite simple. By using an employee number, we can quickly identify an employee; and similarly, by using a book number, we can uniquely identify a book as shown in figure 7.

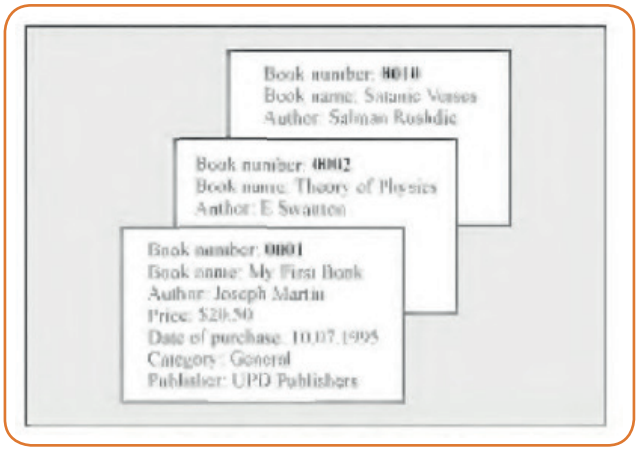


Figure 7: Cards for the books

I should highlight that a field used to identify a record is called as a record key, or just a key. As and when a new book arrives in the library, the librarian creates a new card

for the book and gives it the next number in sequence. Since there can be many books with the same title, author or publisher, none of these can be used to uniquely identify a book. Hence, something additional is needed for this purpose. This is achieved by having book number as an extra field in the record. A field that can identify a record uniquely is called as primary key of the record. Hence, here the book number is the primary key. Given a book number, we are sure that to get only one record. A field that identifies one or more records, not necessarily uniquely, is called a secondary key. Let us take for instance, given an author name, we may or may not be able to identify a book uniquely: the library may have two or more books written by the same author. Hence, the author field identifies a group of records. Hence, it is a secondary key. Other secondary keys in this case can be the book title, publisher and so on.

Thus, I should include that as shown in Figure 8, a record key can be of two types:

1. Primary key: Identifies a record uniquely based on a field value.
2. Secondary key: May or may not identify a record uniquely, but can identify one or more records based on a field value.

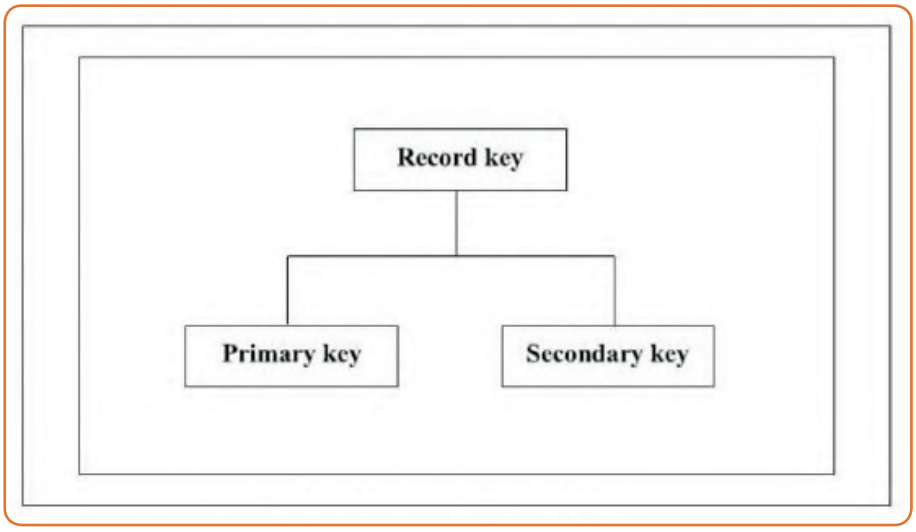


Figure 8: Types of Record Keys

For simplicity, let us assume that there are only 10 books in the library. Suppose a member wants a list of all the books written by Mahatma Gandhi. The librarian can perform a quick search. He would simply look at one card after another and check if the author's name on the card is Gandhi. If they match, the librarian would inform the

member about the book. The algorithm for this process is shown in Figure 9.

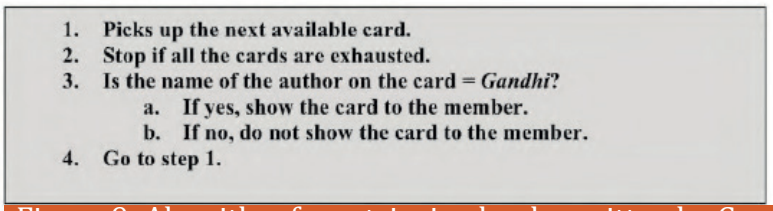

1. Picks up the next available card.
2. Stop if all the cards are exhausted.
3. Is the name of the author on the card = *Gandhi*?
    a. If yes, show the card to the member.
    b. If no, do not show the card to the member.
4. Go to step 1.

Figure 9: Algorithm for retrieving books written by Gandhi

We can carry a similar search can be carried out to find the books published by a particular publisher. As we already know, any such inquiry is called as online query. The process of looking up the cards is called a search. Thus, the records in a file need to be searched in order to respond to an online query.

Summary

So far we have discussed that storing data, records and information into files has become a state-of-the-art. Records or information that need to be archived or are needed for further processing are stored in files and can be accessed readily when they are needed. This unit has taught about master and transaction files, Computer files and hoe to search for records in files.

Self-Assessment Questions

1. Define the following terms:Field
2. Record
3. File
4. Comput Self-Assessment

Tutor Marked Assignment

What is Record Key?

With the aid of a diagram, explain the differences between the two 92) types of Record Keys.

References

Chao L. (2006). Database Development and Management. Taylor & Francis Group, LLC. ISBN 0-8493-3318-0

Ozzu M. T. (2012). Overview of Database Management. David R. Cheriton School of Computer Science, University of Waterloo.

Kahate A. (20040. Introduction to Database Management Systems. Pearson Education. ISBN 9788131700785, eISBN 9788131775820

Further Readings

Conger S. (2012). Hands-on Database design and Development. Pearson education. ISBN 13: 978-0-13-610827-6, ISBN 10:0-13-610827-X.

data flow
source: Unsplash

MODULE 2

DATABASE AND DATABASE MANAGEMENT SYSTEMS

UNITS

imgix database room
Source: unsplash (imgix)

UNIT 1

DATABASES

Introduction

You should note that the success of an organization depends on its ability to acquire accurate and timely data about its operations, to manage this data effectively, and to use it to analyze and guide its activities. The amount of information available to us is literally exploding, and the value of data as an organizational asset is widely recognized. Yet without the ability to manage this vast amount of data, and to quickly find the information that is relevant to a given question, as the amount of information increases, it tends to become a distraction and a liability, rather than an asset. This paradox drives the need for increasingly powerful and flexible data management systems.

Learning Outcomes

When you have studied this unit, you should be able to:

- Identify how the database approach is different and superior to earlier data systems
- Discuss how information demand and technology explosion drive database systems
- Trace the evolution of data systems and note how we have arrived at the database approach
- List the benefits of database systems and perceive the need for them

imgix database room
Source: unsplash (imgix)

Main Content

Overview of Database Concepts [SAQ1, 2]

READING TIME: 2 MINS

Do you know that databases and database technology have a major impact on the growing use of computers as shown in figure 1? It is fair to say that databases play a critical role in almost all areas where computers are used, including business, electronic commerce, engineering, medicine, genetics, law, education, and library science. The word database is so commonly used that we must begin by defining what a database is.

I should let you know a database, at its simplest level, is a collection of related data. It doesn't have to be electronic. The card catalogs that libraries used to have were certainly databases. A scientist's spiral notebook where he or she keeps notes and observations could be considered a database, so too could a phone or address book. When we say "database," though, we usually mean electronic

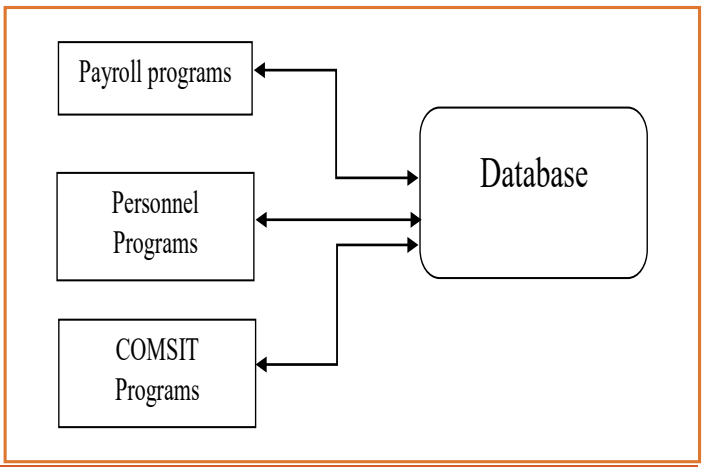


Figure 1: Database (User - maintained)(Source: Silberschatz, Korth & Sudarshan, 2003)

databases, that runs on computer.

Our initial definition is quite general.A database is a collection of related data. By data, we mean known facts that can be recorded and that have implicit meaning. For example, consider the names, telephone numbers, and addresses of the people you know. You may have recorded this data in an indexed address book or you may have stored it on a hard drive, using a personal computer and software such as Microsoft Access or Excel. This collection of related data with an implicit meaning is a database. The preceding definition of database is quite general; for example, we may consider the collection of words that make up this page of text to be related data and hence to constitute a database. However, the common use of the term database is usually more restricted. I should let you know that a database has the following implicit properties:

A database represents some aspect of the real world, sometimes called the miniworld or the universe of discourse (UoD). Changes to the miniworld are reflected in the database.

A database is a logically coherent collection of data with some inherent meaning. A random assortment of data cannot correctly be referred to as a database.

A database is designed, built, and populated with data for a specific purpose. It has an intended group of users and some preconceived applications in which these users are interested.

camera and lenses
Source unsplash by Como ludd

In other words, let me include further that a database has some source from which data is derived, some degree of interaction with events in the real world, and an audience that is actively interested in its contents. The end users of a database may perform business transactions (for example, a customer buys a camera) or events may happen (for example, an employee has a baby) that cause the information in the database to change. In order for a database to be accurate and reliable at all times, it must be a true reflection of the miniworld that it represents;

therefore, changes must be reflected in the database as soon as possible.

I should continue that database can be of any size and complexity. For example, the list of names and addresses referred to earlier may consist of only a few hundred records, each with a simple structure. On the other hand, the computerized catalog of a large library may contain half a million entries organized under different categories—by primary authour's last name, by subject, by book title with each category organized alphabetically.

Purpose of Database Systems

We traced the evolution of data systems. We grasped the essentials of the explosive growth of information technology. We noted the escalating demand of organizations for information. We observed how growth in information technology and the increased demand for information worked hand in hand. Increasing demand for information spurred the growth of information technology. Growth of information technology, in turn, enabled organizations to satisfy the increasing demand for information. Let us summarize the driving forces for organizations to adopt database systems. A major reason is the inadequacy of the earlier file-oriented data systems. We shall review the limitations and see how database systems overcome the limitations and provide significant benefits. In the early days, database applications were built on top of file systems.

Drawbacks of using file systems to store data:

Data redundancy and inconsistency: Multiple file formats, duplication of information in different files.

Difficulty in accessing data: Need to write a new program to carry out each new task.

Data isolation — multiple files and formats.

Integrity problems: Integrity constraints (e.g. account balance > 0) become part of program code, Hard to add new constraints or change existing ones.

Atomicity of updates: Failures may leave database in an inconsistent state with partial updates carried out e.g. transfer of funds from one account to another should

either complete or not happen at all.

Concurrent access by multiple users: Concurrent accessed needed for performance. Uncontrolled concurrent accesses can lead to inconsistencies e.g. two people reading a balance and updating it at the same time

Security problems

Database systems offer solutions to all the above problems

The Database Approach (2mins)

READING TIME: 2 MINS

We have reviewed the benefits of database systems and established how they are superior to the earlier file-oriented data systems. We caught a glimpse of the features of database systems that produce several benefits. Database systems reduce data redundancy, integrate corporate data, and enable information sharing among the various groups in the organization. Now you are ready for an initial, formal definition of a database.

Database: A Formal Definition

Let us examine the following definition:

A database is an ordered collection of related data elements intended to meet the information needs of an organization and designed to be shared by multiple users.

Note the key terms in the definition:

Ordered collection: A database is a collection of data elements. Not just a random assembly of data structures, but a collection of data elements put together deliberately with proper order. The various data elements are linked together in the most logical manner.

Related data elements: The data elements in a database are not disjointed structures without any relationships among them. These are related among themselves and also pertinent to the particular organization.

Information needs: The collection of data elements in a database is there for a specific purpose. That purpose is to satisfy and meet the information needs of the

organization. In a database for a bank, you will find data elements that are pertinent to the bank's business. You will find customer's bank balances and ATM transactions. You will not find data elements relating to a student's major and examination grades that belong in a database for a university. You will not find a patient's medical history that really belongs in a database for a medical center.

Shared: All authorized users in an organization can share the information stored in its database. Integrated information is kept in the database for the purpose of sharing so that all user groups may collaborate and accomplish the organization's objectives.

Data-Driven, Not Process-Driven

I should let you know that when an organization adopts a database approach to managing corporate data, the very method of designing and implementing applications changes. Traditionally, when you design and implement an application with file-oriented data systems, you use a process-driven approach. That method changes with database systems. You shift your design and implementation method to a data-driven approach.

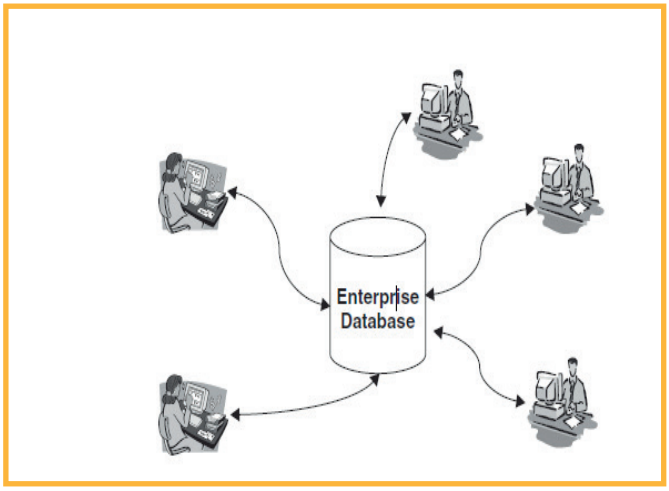


Figure 2 illustrates a centralized database.

Types of Databases

By now you are convinced of the significance of information for an organization. You know that information is a key corporate asset and that it has to be managed, protected, and used like any other major asset. The corporate database that holds an organization's data is the underlying foundation for corporate information. Organizations are also faced with questions regarding how and where to hold the corporate data. Where should an enterprise hold its data? Should all the

corporate data be kept centrally in one place? If so, what are the advantages and disadvantages? Or should the corporate data be divided into suitable fragments and the pieces kept at different locations? What are the implications of this arrangement? Organizations primarily adopt one of two approaches. If the entire database is kept in one centralized location, this type of database is a centralized database. On the other hand, if fragments of the database are physically placed at various locations, this type of database is a distributed database. Each type has its own benefits and shortcomings. Again, whether an enterprise adopts a centralized or a distributed approach depends on the organizational setup and the information requirements.

Let us review the two types.

Centralized Database

Figure 2 illustrates a centralized database. All data are at a single site. Data is accessed from remote sites through communication links. It is easy to administer. Uncertain data availability.

Common Examples are:

1. Personal Database
2. Central Computer Database
3. Client/Server Database

Be informed that personalized databases are always centralized in one location. If your company has a centralized computer system, then the database must reside in that central location. In the client/server architecture, the database resides on a server machine. The entire database may be kept on a single server machine and placed in a central location.

When all corporate data is in one place in a centralized database, companies find it easier to manage and administer the database. You can control concurrent accesses to the same data in the database easily in a centralized database. You can maintain security controls easily. However, if your company's operations are spread

across remote locations, these locations must access the centralized database through communication links. Here, data availability depends on the capacity and dependability of the communication links.

1. Distributed Database

Be informed that figure 3 shows how fragments of a corporate database are spread across remote locations. Global organizations or enterprises with widespread domestic operations can benefit from distributed databases. In such organizations computer processing is also distributed, with processing done locally at each location. A distributed database gets fragmented into smaller data sets. Normally, you would divide the database into data sets on the basis of usage. If a fragment contains data that are most relevant to one location, then that data set is kept at that location. At each location, a fragment of the enterprise data is placed based on the usage. Each fragment of data at every location may be managed with the same type of database management system. For example, you may run Oracle DBMS at every location. In that case, you run your distributed database as a homogenous database. On the other hand, if you elect to manage the data fragments at different locations with different DBMSs, then you run your distributed database as a collection of heterogeneous database systems. Heterogeneous arrangement provides extra flexibility. However, heterogeneous distribution is difficult to coordinate and administer.

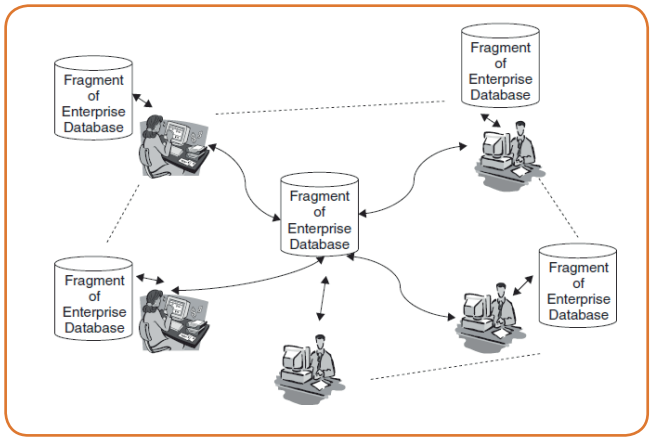


Figure 3: Distributed database (Source: Silberschatz, Korth & Sudarshan, 2003).

It is used for global and spread-out organizations. Enterprise data is distributed

across multiple computer systems.

There are two categories of Distributed Database:

i. Homogeneous databases - consist of elements that are the same or similar kind of nature.

ii. Heterogeneous databases - consisting of elements that are not of the same kind or nature.

Advantages of Database Processing [SAQ3]

READING TIME: 2 MINS

1. Getting more information from same amount of data

2. Sharing of data

3. Balancing conflicting requirements

4. Enforcement of standardsControlled RedundancyConsistency

5.Integrity

6. Security

7. Data Independence

- **Getting more information from same amount of data** – The main goal of a computer is to turn data (recorded facts) into information. If all the data were in a common database, all authorized users will be able to access it easily.

- **Sharing of data** – The data can be shared among authorized users, allowing users access to more data, several users will have access to the same piece of data but use it in a variety of ways. When a Personnel's address changes, the change is immediately available to all users.

- **Balancing conflicting requirements** – For the database approach to function adequately, there must be a person or group within the organization in charge of the database itself. This group is often called Database Administrator (DBA). By keeping this overall needs of the organization in mind, DBA can structure the database to the benefit of the entire organization, not just a single user group.

- **Enforcement of Standards** – With the central controls mentioned in the previous paragraphs, DBA can ensure that standards for such things as data names,

usages and formats are followed uniformly throughout the organization.

- **Controlled Redundancy** – since data that was kept separately in a file-oriented system is now integrated into a single database, we no longer have multiple copies of the same data.
- **Consistency** – Since we have controlled redundancy, data in each record for various entities remain the same throughout the database.
- **Integrity** – An integrity constraint is a rule that data in the database must follow. A database has integrity if data in the database satisfies all integrity constraints that have been established. The DBA can define validation procedures that will ensure the integrity of the database.

- **Security** – is the prevention of access to the database by unauthorized users. Since, DBA has control over the operational data. It can define procedures to ensure that only legitimate users access the data.
- **Data Independence** – occurs when the structure of the database can change without requiring the programs that access the database to change.

Disadvantages of Database Processing Managing Projects [SAQ4]

READING TIME: 2 MINS

1. Size
2. Complexity
3. Cost
4. Additional hardware requirements
5. Higher impact of a failure
6. Recovery more difficult

1. **Size** – I should inform you that to support all the complex functions it must provide to users, a DBMS must by its nature, be a large program occupying megabytes of disk space, as well as a substantial amount of internal memory.
2. **Complexity** – the complexity and breadth of the functions furnished by a DBMS make it a complex product. Programmers and Analysts must understand the features of the system to take full advantage of it.

3. **Cost** – a good mainframe DBMS is an expensive product. By the time all the appropriate components related to the DBMS are purchased for a major mainframe system, the total price can easily run into $100,000 to $400,000 range.

4. **Additional hardware requirements** – because of the size of the size and complexity of a DBMS, greater hardware resources are required than would be necessary without the DBMS.

5. **Higher impact of a failure** – since many of the information systems resources are not concentrated in the databases, a failure of any component has a much more far – reaching effect than a non database environment.

6. **Recovery more difficult** – because of the added complexity, the process of recovering the database in the event of a catastrophe is a more complicated one, particularly if the database is being updated by a large number of users concurrently at the same time).

Summary

It is ideal I ended that for nearly three decades, there has been tremendous growth in every sector of computing technology, especially in data storage devices and in the software to manage enterprise data. Organizations began to need not only more information but also different types of information for newer purposes than merely running day-to-day operations.

Major forces driving the evolution of database systems: information considered as key corporate asset, explosive growth of computer technology, escalating demand for information, and inadequacy of earlier data systems.

Self-Assessment Questions

1. What is Data?
2. What is Database?
3. State five (5) advantages of database Processing.
4. State five (5) disadvantages of database Processing.

Tutor Marked Assignment

State the differences between File-based system and Database Management system.

State four (4) drawbacks of using file Systems.

Explain the Database Approach for Information Processing.

References

Rich Maclin[3], Database Management Systems, Chapter 1. rmaclin@d.umn.edu

Silberschatz[1] A., Korthand H. F., and Sudarshan S., (1999), Database System Concepts (3rd Edition) (pp. 63–72), WCB/McGraw-Hill, USA. ISBN 0-07-031086-6.

Begg C. E. and Connolly T. M. (2002). Database Systems: A Practical Approach to Design, Implementation, and Management, Addison-Wesley.

Further Reading

en.wikipedia.org/wiki/Database_system.

en.wikipedia.org/wiki/Database_management_system Accessed 23rd September 2018.

database files
Source: Pexels by Kevin Ku

UNIT 2

DATABASE MANAGEMENT SYSTEM (DBMS)

Introduction

Database Management System (DBMS) is a piece of software that is designed to make tasks easier. By storing data in DBMS, rather than as a collection of operating system files, the DBMS features can be used to manage the data in a robust and efficient manner. As the volume of data an number of users grow, hundreds of gigabytes of data and thousands of users connected to share DBMS resources.

Learning Outcomes

When you have studied this unit, you should be able to:

- Identify the different Views of data.
- Define Instances and Schema
- List the functions of Database Management System (DBMS)

database files
Source: Pexels by Kevin Ku

Main Content

Database Management System (DBMS) [SAQ1]

READING TIME: 2 MINS

I should inform you that we use software packages called database Management systems (DBMS) for manipulating databases. A Database Management System is a software product through which users interact with a database. The actual manipulation of the underlying database structure is handled by DBMS as shown in figure 1.

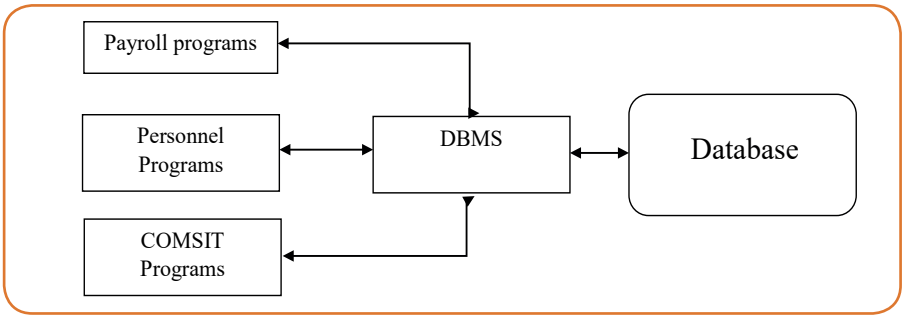


Figure 1: Database (Using a DBMS)

I should let you know that a database management system (DBMS) is a collection of programs that enables users to create and maintain a database. The DBMS is a general-purpose software system that facilitates the processes of defining, constructing, manipulating, and sharing databases among various users and applications. Defining a database involves specifying the data types, structures, and constraints of the data to be stored in the database.

The database definition or descriptive information is also stored by the DBMS in

the form of a database catalog or dictionary; it is called meta-data. Constructing the database is the process of storing the data on some storage medium that is controlled by the DBMS. Manipulating a database includes functions such as querying the database to retrieve specific data, updating the database to reflect changes in the miniworld, and generating reports from the data. Sharing a database allows multiple users and programs to access the database simultaneously. An application program accesses the database by sending queries or requests for data to the DBMS. A query2 typically causes some data to be retrieved; a transaction may cause some data to be read and some data to be written into the database. Let me tell you as well that other important functions provided by the DBMS include protecting the database and maintaining it over a long period of time.

View of Data

READING TIME: 1 MIN

Levels of Abstraction

Physical level: describes how a record (e.g., customer) is stored.

Logical level: describes data stored in database, and the relationships among the data.

```
type customer = record
name : string;
street : string;
city : integer;
end;
```

View level: application programs hide details of data types. Views can also hide information (e.g. salary) for security purposes as shown in figure 2.

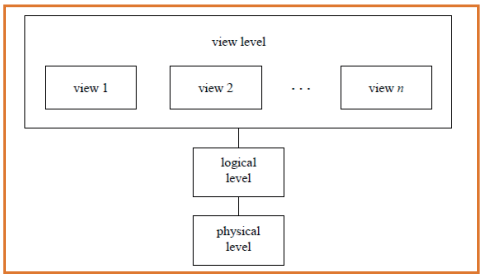


Figure 2: View of Data (Source: Silberschatz, Korth & Sudarshan, 2003).

Instances and Schema

READING TIME: 1 MIN

Similar to types and variables in programming languages

Schema – the logical structure of the database (e.g., the database consists of information about a set of customers and accounts and the relationship between them) analogous to type information of a variable in a program.

Physical schema: database design at the physical level.

Logical schema: database design at the logical level.

Instance – the actual content of the database at a particular point in time analogous to the value of a variable.

Physical Data Independence – the ability to modify the physical schema without changing the logical schema. Applications depend on the logical schema. In general, the interfaces between the various levels and components should be well defined so that changes in some parts do not seriously influence others.

Functions of Database Management System (DBMS)

READING TIME: 1 MIN

The main function of a DBMS is to store, update, and retrieve data in a database.

1. **Data storage, retrieval and update** – A DBMS must provide users with the ability to store, retrieve and update data in the database.

2. **A User-accessible Catalog** - A DBMS must provide a catalog in which descriptions of data items are stored and that is accessible to users.

3. **Transaction support** - A DBMS must provide a mechanism that will ensure that either all the updates corresponding to a given transaction are made or that none of them is made.

4. **Concurrency Control Services** - A DBMS must provide a mechanism to ensure that the database is updated correctly when multiple users are updating the database concurrently.

5.**Recovery Services** - A DBMS ensures the recovering of the database in the event that the database is damaged in any way.

6. **Authorization Services** - A DBMS ensures that only authorized users can access the database.

7. **Support for Data Communication** - A DBMS must be capable of integrating with communication software.

8. **Integrity Services** - A DBMS ensures that both data in the database and changes to the data follow certain rules.

9. **Services to promote Data Independence** - A DBMS must include facilities to support the independence of programs from the actual structure of that database.

10. Utility Services - A DBMS must provide a set of utility services for enhancing the proper functioning of the database.

Summary

In this section, we examined Database Management System. We explained instances, schema, and also the functions of a DBMS.

Self-Assessment Questions

1. What is a Database Management System

Tutor Marked Assignment

State and explain the functions of a DBMS

Explain what you understand by a database schema and an instance.

References

Elmasri R. and Navathe S. B., (2011). Fundamentals of Database Systems. Addison-Wesley.

Gupta S. B. and Mittal A., (2004). Introduction to database management systems. Pearson Education.

Further Reading

Begg C. E. and Connolly T. M. (2002). Database Systems: A Practical Approach to Design, Implementation, and Management, Addison-Wesley.

woman at Data server room
source: Pexels by Christina Morillo

UNIT 3

DATABASE USERS AND ADMINISTRATORS

Introduction

In this unit we will learn about the database users, database administrators and their various functions and roles. It also reflects on the Network database and the servers that allow several users to access the database.

Learning Outcomes

When you have studied this unit, you should be able to:

- Identify the types of database users.
- List the functions of a Database Administrator (DBA).
- Describe the SQL Server Security model.

woman at Data server room
source: Pexels by Christina Morillo

Main Content

Database Users and Administrators

READING TIME: 10 SECS

Please note that a primary goal of a database system is to retrieve information from and store new information into the database. People who work with a database can be categorized as database users or database administrators.

Database Users and User Interfaces [SAQ1]

READING TIME: 2 MINS

There are four different types of database-system users, differentiated by the way they interact with the system. Different types of user interfaces have been designed for the different types of users.

Na¨ıve users are unsophisticated users who interact with the system by invoking one of the application programs that have been written previously. For example, a clerk in the university who needs to add a new instructor to department A invokes a program called new hire. This program asks the clerk for the name of the new instructor, her new ID, the name of the department (that is, A), and the salary.

The typical user interface for na¨ıve users is a forms interface, where the user can fill in appropriate fields of the form. Na¨ıve users may also simply read reports generated from the database.

As another example, consider a student, who during class registration period, wishes to register for a class by using a Web interface. Such a user connects to a Web application program that runs at a Web server. The application first verifies the identity of the user, and allows her to access a form where she enters the desired information. The form information is sent back to the Web application at the server, which then determines if there is room in the class (by retrieving information from the database) and if so adds the student information to the class roster in the database.

Application programmers are computer professionals who write application programs. Application programmers can choose from many tools to develop user interfaces. Rapid application development (RAD) tools are tools that enable an application programmer to construct forms and reports with minimal programming effort.

Sophisticated users interact with the system without writing programs. Instead, they form their requests either using a database query language or by using tools such as data analysis software. They submit such queries to a query processor, whose function is to break down Data Manipulation Language (DML) statements into instructions. Analysts who submit queries to explore data in the database fall in this category.

Specialized users are sophisticated users who write specialized database applications that do not fit into the traditional data-processing framework. Among these applications are computer-aided design systems, knowledgebase and expert systems, systems that store data with complex data types (for example, graphics data and audio data), and environment-modeling systems.

Database Administrator [SAQ2]

One of the main reasons for using DBMSs is to have central control of both the data and the programs that access those data. A person who has such central control over the system is called a Database Administrator (DBA).

Below are the functions of a DBA as includes:

- **Schema definition**. The DBA creates the original database schema by executing a set of data definition statements in the DDL.

- **Storage structure and access-method definition.**

- **Schema and physical-organization modification.** The DBA carries out changes to the schema and physical organization to reflect the changing needs of the organization, or to alter the physical organization to improve performance.

- **Granting of authorization for data access.** By granting different types of authorization, the database administrator can regulate which parts of the database various users can access. The authorization information is kept in a special system structure that the database system consults whenever someone attempts to access the data in the system.

- **Routine maintenance**. Examples of the database administrator's routine maintenance activities are:

a. Periodically backing up the database, either onto tapes or onto remote servers, to prevent loss of data in case of disasters such as flooding.

b. Ensuring that enough free disk space is available for normal operations, and upgrading disk space as required.

c. Monitoring jobs running on the database and ensuring that performance is not degraded by very expensive tasks submitted by some users.

Network Database

READING TIME: 2 MINS

I should inform you that a network database allows a large number of users to use the same database. Any database failure will have a great impact on many users as well as the entire organization or enterprise. This brings up some questions such as

how maintain user accounts, database security, and reliability. Therefore, we need to address issues such as managing user accounts and security, database backup and restoration, and database performance tuning. These tasks are normally handled by Database Administrators (DBAs).

The following are common responsibilities of DBAs in network environments:

Implementing and maintaining databases:

DBAs participate in the database design process. They install and upgrade DBMS systems, perform proper data-transfer processes, schedule database turn-on and turn-off, maintain databases to allow users to access them at any time, and keep database-related documentation in a safe place.

Managing user accounts and database security:

I should let you know as well that DBAs create user accounts on a database and enforce security such as setting up different levels of log-ins, authentications, and permissions for users.

I. Performing database backup and restoration:

DBAs develop plans to recover a database as quickly as possible after a database failure. They set up a proper database backup plan to prevent the loss of data and perform scheduled backups and database recovery.

II. Monitoring and tuning database performance:

DBAs configure and tune database parameters to ensure that a database is running with optimalperformance. They detect problems that cause a slowdown in performance.

III. Setting up replication services:

DBAs design a proper replication topology for reliability and performance in a distributed computing environment.

IV. Providing data analysis services:

DBAs maintain the data warehouse or data mart for enterprises and provide services for data analysis and decision making.

Working with database application developers:

DBAs help developers implement application-related constraints on database objects and upload

server-side stored procedures, triggers, and ASP programs. A qualified DBA should be knowledgeable in many database fields such as having a good understanding of database design, DBMS systems, database applications, and an organization's network architecture. He or she should also have interpersonal skills. In this chapter and the next chapter, we study related topics to accomplish some of the above-mentioned tasks.

Managing Database User Accounts and Security

READING TIME: 1 MIN

You should be informed that when a database is used in a network environment, it requires a variable security model so that users can perform database operations on various levels. Security requirements for data stored in a network database and for operations performed on database objects are different depending on the types of users.

A DBMS system, such as SQL Server, often provides a security model to meet security requirements. The components of the SQL Server security model are described in Figure 1.

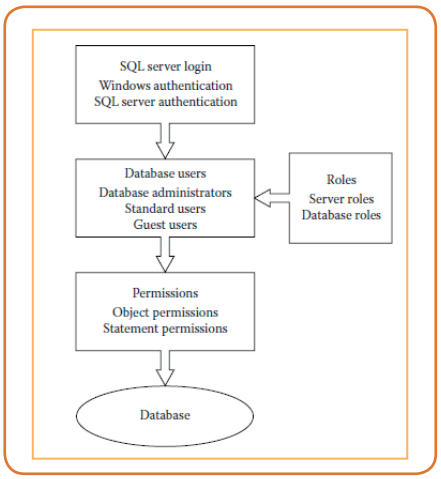


Figure 1: SQL Server security model

1. Server Authentication

Bear in mind that SQL Server log-in is an authentication process, which is the first stage of security measures. The authentication process boils down to two parts: the Windows authentication mode and SQL Server authentication mode.

a. Windows Authentication Mode

By using this authentication mode, you can bypass the SQL Server log-in process by letting the SQL Server share the Windows log-in process. Once a Windows user logs on to the computer with the correct username and password, he or she automatically logs on to the SQL Server database. The benefits of Windows authentication are:

It takes advantage of the Windows user security mechanism such as network encryption for log-in information and automatically logs out repeatedly failed log-in attempts.

For each Windows user, there is no need to create and update another account for a SQL Server database. This can reduce the workload of a system administrator who manages a large number of users. The default log-in mode for SQL Server is set as SQL Server Login. However, you can easily change the authentication mode to Windows authentication if it is required.

b. Mixed Authentication Mode

You should be aware that when using the mixed authentication mode, both the Windows authentication mode and SQL Server authentication mode are used. For users who have no Windows accounts, the SQL Server authentication will allow them to use the SQL Server database. Often in an enterprise network environment, many computers use Linux or UNIX operating systems. In such a case, the SQL

Server authentication mode is the choice. To configure security with the SQL Server authentication mode, select the SQL Server Authentication option in Figure 2 and enter the password for the SQL Server.

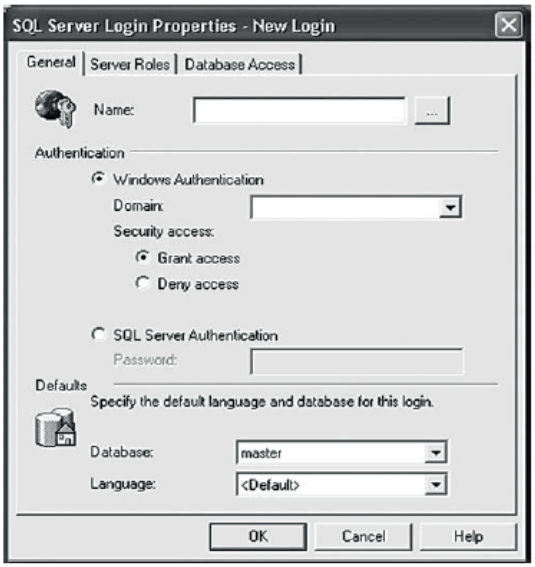


Figure 2: New log-in Dialog

2. Server Authourization

You should be aware that after front-end users successfully log on to a database with the correct username and password, they will try to access some database objects and perform some database-related activities. These users are prevented from executing database administration commands or accessing the database objects without proper permissions. This leads to the authorization stage, which sets up security measures to specify permissions given to different types of users. We first create users and then add different roles to these users based on their job duties. Through the roles, we can specify database-accessing permissions for the users.

3. Database Users

You should note that there are three types of users: database administrators, standard users, and guest users. Each type has different security requirements.

a. Database Administrators:

Beware that DBAs are expected to do many database management jobs such as starting and shutting down database servers, installing and upgrading database server components, managing user accounts, tuning database server processes, creating and updating database objects, replicating databases, and backing up and restoring databases.

For a small company, there may be one DBA who does everything. For a large company, there is a group of DBAs who share the above responsibilities.

b. Standard Users:

These users may be allowed to query only or to query and modify tables created in the database, depending on the permissions assigned to them.

c. Guest users:

You should be informed that there may be public users who need to access a server's default database such as Northwind or pubs in SQL Server through the Internet connection for learning purposes. In such a case, guest users are created. With the correct username and password, a guest user can log on to SQL Server and access the Northwind and pubs databases, but he or she will not be allowed to access any other database objects.

d. Permissions

Basically, a permission gives you the ability to do certain things and controls the access to database objects. There are two kinds of permissions that are supported by SQL Server: the statement permission and the object permission.

he statement permission allows a database user to use database manipulation statements such as
CREATE,ALTER,DELETE and BACKUP to create, change, drop, and back up database objects in a database. The object permission is used to allow a database user to read, write, update, or execute the content in database objects such as tables, views, or stored procedures. to create, change, drop, and back up database objects in a database.

e. Roles

A role allows a database administrator to apply a set of permissions to a group of database users who have similar tasks. When you change permissions to group users

who are assigned the same role, the permissions are changed at the role level instead of at the user level. A database user can be assigned to different roles. For example, an accounting manager can be assigned to an accountant role and a manager role. SQL Server 2000 supports two types of fixedroles, server roles and database roles.

Summary

In this unit we explained the database users and their respective roles. This teaches about authentication and authourization of users and assigning permissions to persons who can access the database.

Self-Assessment Questions

1. List the four types of database users.
2. State five functions of a Database Adminstrator (DBA)

Tutor Marked Assignment

With the aid of a diagram, describe the SQL security model.

References

Chao L. (2006). Database Development and Management. Taylor & Francis Group, LLC. ISBN 0-8493-3318-0.

Silberschatz A., Korth H. F., and Sudarshan S. (2003). Database System Concepts. McGraw-Hill Education.

Further Reading

en.wikipedia.org/wiki/Database_system.

en.wikipedia.org/wiki/Database_management_system.

coding
source: Unsplash

MODULE 3

DATA MODELS

UNITS

UNIT 1: Database Instances and Schemas

UNIT 2: Netweork,Hierarchical, Relational Models And Object-relational Models.

UNIT 3: Entity-Relationship Models

UNIT 4: Domains, Attributes, Tuples And Relations

man hand on screen
source: Pexls by Vitaly Vlasov

UNIT 1

DATABASE INSTANCES AND SCHEMAS

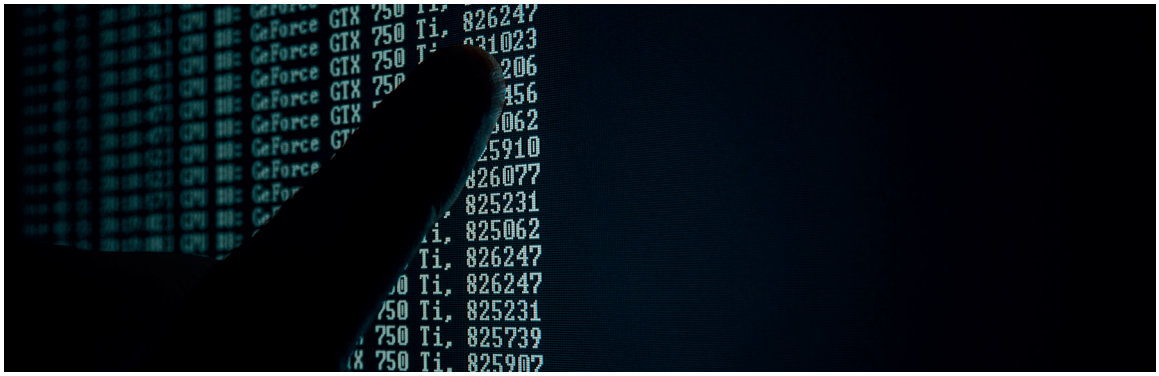
Introduction

In this unit, we will learn about the plan of creating databases and the data to be stored in them. How structure of the database is defined, concepts or Instances and Schemas will be discussed as well.

Learning Outcomes

When you have studied this unit, you should be able to:

- Describe Instances and Schemas.
- Distinguish between Logical and Physical Schema.



man hand on screen
source: Pexls by Vitaly Vlasov

Main Content

Schema [SAQ1, 2, 3, 4]

READING TIME: 1 MIN

This is similar to types and variables in programming languages

Schema – the logical structure of the database (e.g., set of customers and accounts and the relationship between them)

Instance – the actual content of the database at a particular point in time.

Physical schema: database design at the physical level

Logical schema: database design at the logical level

Instance – the actual content of the database at a particular point in time analogous to the value of a variable

Physical Data Independence – the ability to modify the physical schema without changing the logical schema. Applications depend on the logical schema. In general, the interfaces between the various levels and components should be well defined so that changes in some parts do not seriously influence others. It is important I let you know that a schema is plan of the database that give the names of the entities and attributes and the relationship among them.

A schema includes the definition of the database name, the record type and the components that make up the records. Alternatively, it is defined as a frame-work into which the values of the data items are fitted. The values fitted into the frame-work changes regularly but the format of schema remains the same e.g., consider the database consisting of three files ITEM, CUSTOMER and SALES. The data structure diagram for this schema is shown in Figure 1. The schema is shown in database language.

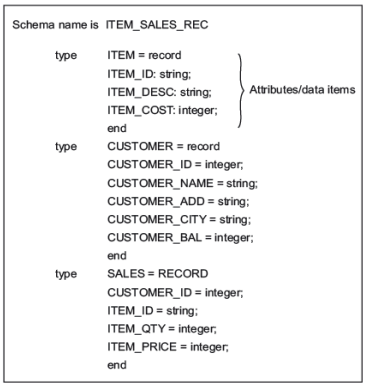


FIGURE 1.5. Data structure diagram for the item sales record.

Figure 1: Data structure diagram for the item sales record (Source: Begg & Connolly, 2002)

Categories of schema

READING TIME: 1 MIN

Generally, a schema can be partitioned into two categories, i.e., (i) Logical schema and (ii) Physical schema.

(i) ***The logical schema*** is concerned with exploiting the data structures offered by the
DBMS so that the schema becomes understandable to the computer. It is important as programs use it to construct applications.

(ii) ***The physical schema*** is concerned with the manner in which the conceptual database get represented in the computer as a stored database. It is hidden behind the logical schema and can usually be modified without affecting the application programs.
The DBMS's provide DDL and DSDL to specify both the logical and physical schema.

Subschema [SAQ7]

READING TIME: 1 MIN

You should note that a subschema is a subset of the schema having the same properties that a schema has. It identifies a subset of areas, sets, records, and data names defined in the database schema available to user sessions. The subschema allows the user to view only that part of the database that is of interest to him. The subschema defines the portion of the database as seen by the application programs and the application programs can have different view of data stored in the database.

The different application programs can change their respective subschema without affecting other's subschema or view.

The Subschema Definition Language (SDL) is used to specify a subschema in the DBMS.

Instances

READING TIME: 1 MIN

It is imperative I let you know that the data in the database at a particular moment of time is called an instance or a database state. In a given instance, each schema construct has its own current set of instances. Many instances or database states can be constructed to correspond to a particular database schema. Every time we update (i.e., insert, delete or modify) the value of a data item in a record, one state of the database changes into another state. The Figure 2 shows an instance of the ITEM relation in a database schema.

ITEM

ITEM-ID	ITEM_DESC	ITEM_COST
1111A	Nutt	3
1112A	Bolt	5
1113A	Belt	100
1144B	Screw	2

FIGURE 1.6. An instance/database state of the ITEM relation.

Figure 2: An instance/database state of the ITEM relation (Begg & Connolly, 2002)

The Three-schema Architecture[SAQ5, 6, 8]

READING TIME: 2 MINS

The goal of the three-schema architecture, illustrated in Figure 3, is to separate the user applications from the physical database. In this architecture, schemas can be defined at the following three levels:

1. The internal level has an internal schema, which describes the physical storage structure of the database. The internal schema uses a physical data model and describes the complete details of data storage and access paths for the database.

2. The conceptual level has a conceptual schema, which describes the structure of the whole database for a community of users. The conceptual schema hides the details of physical storage structures and concentrates on describing entities, data types, relationships, user operations, and constraints. Usually, a representational data model is used to describe the conceptual schema when a database system is implemented. This implementation conceptual schema is often based on a conceptual schema design in a high-level data model.

3. The external or view level includes a number of external schemas or user views. Each external schema describes the part of the database that a particular user group is interested in and hides the rest of the database from that user group.

As in the previous level, each external schema is typically implemented using a representational data model, possibly based on an external schema design in a high-level data model.

I should let you be aware also that the three-schema architecture is a convenient tool with which the user can visualize the schema levels in a database system as shown in figure 4. Most DBMSs do not separate the three levels completely and explicitly, but support the three-schema architecture to some extent. Some older DBMSs may include physical-level details in the conceptual schema. The three-level ANSI architecture has an important place in database technology development because it clearly separates the users' external level, the database's conceptual level, and the internal storage level for designing a database.It is very much applicable in the design of DBMSs, even today. In most DBMSs that support user views, external schemas are specified in the same data model that describes the conceptual-level information (for example, a relational DBMS like Oracle uses SQL for this). Some DBMSs allow different data models to be used at the conceptual and external levels. An example is Universal Data Base (UDB), a DBMS from IBM, which uses the relational model to describe the conceptual schema, but may use an object-oriented model to describe an external schema.

You should be notified that the three schemas are only descriptions of data; the stored data that actually exists is at the physical level only. In a DBMS based on the three-schema architecture, each user group refers to its own external schema. Hence, the DBMS must transform a request specified on an external schema into a request against the conceptual schema, and then into a request on the internal schema for processing over the stored database. If the request is a database retrieval, the data extracted from the stored database must be reformatted to match the user's external view. The processes of transforming requests and results between levels are called mappings. These mappings may be time-consuming, so some DBMSs—especially those that are meant to support small databases—do not support external views. Even in such systems, however, a certain amount of mapping is necessary to transform requests between the conceptual and internal levels.

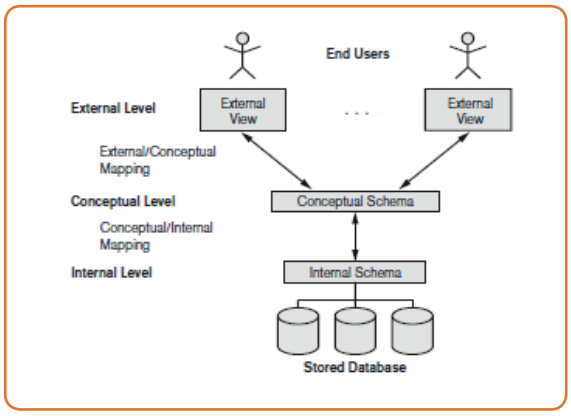


Figure 3: The three-schema Architecture (Source: Begg & Connolly, 2002)

Advantages of three-level Architecture

READING TIME: 1 MIN

Let me inform you the motive behind the three-level architecture is to isolate each user's view of the database from the way the database is physically stored or represented. The advantages of the three-level architecture are as follows:

The changes to physical storage organization does not affect the internal structure of the database. e.g., moving the database to a new storage device.

To use the database, the user is no need to concern about the physical data storage details.

1. The conceptual structure of the database can be changed by the DBA without affecting any user.

2. The database storage structure can be changed by the DBA without affecting the user's view.

Summary

In this chapter we examined schemas, subschemas, instances and also a three- level architecture. Data independence was also examined. Advantages of a three-level architecture was also examined.

Self-Assessment Questions

1. A(n) _____ contains the logical structure for the information.

2. A(n)_____ represents how data is physically stored on a storage device.

3. A(n)_____ represents how knowledge users see information.

4. A data model is a collection of concepts that can be used to describe the _____ of a

database.

5. _____ schema describes physical storage structures and access paths.

6. In three-schema architecture, user views are defined at _____ schema.

7. ____ data model provides concepts that are close to the way many users perceive data.

8. External schema describes the various user _____.

Tutor Marked Assignment

1. Which of the following is not a type of database management system and why?

(a) Hierarchical (b) Network

(c) Relational (d) Sequential.

2. What does a Schema describes?

References

Silberschatz A., Korth H. F., and Sudarshan S. (2003). Database System Concepts. McGraw-Hill Education.

Begg C. E. and Connolly T. M. (2002). Database Systems: A Practical Approach to Design, Implementation, and Management, Addison-Wesley.

Further Reading

en.wikipedia.org/wiki/Database_system.

en.wikipedia.org/wiki/Database_management_system.

Network room
Source: Pexels by Manuel geissinger

UNIT 2

NETWORK, HIERARCHICAL, RELATIONAL MODELS AND OBJECT-RELATIONAL MODELS

Introduction

This unit explains the concept of data models. It explains how data models support communication between the users and the Database Designers.

Learning Outcomes

When you have studied this unit, you should be able to:

- List the various data models.
- Describe operations of data models.
- State the advantages and disadvantages of the models.

Network room
Source: Pexels by Manuel geissinger

Main Content

Types of Data Model (2mins)

READING TIME: 2 MINS

I should let you know that a data model is a collection of concepts that can be used to describe the structure of the database including data types, relationships and the constraints that apply on the data. A data model helps in understanding the meaning of the data. A data model supports communication between the users and database designers. The major use of data model is to understand the meaning of the data and to facilitate communication about the user requirements.

The various data models can be divided into three categories, such as

(i) Record Based Data Models.

(ii) Object Based Data Models.

(iii) Physical Data Models.

(i) Record Based Data Models : These models represent data by using the record structures. These models lie between the object based data models and the physical data models. These data models can be further categorised into three types:

(a) Hierarchical Data Model

(b) Network Data Model

(c) Relational Data Model.

(ii) Object Based Data Models: You should note that these models are used in describing the data at the logical and user view levels. These models allow the users to implicity specify the constraints in the data. I should include that this data models can be further categorised into four types:

(a) Entity Relationship Model (ER-Model)

(b) Object Oriented Model

(c) Semantic Data Model

(d) Functional Data Model.

We will discuss about this models in the coming sections.

(iii) Physical Data Models: These models provide the concepts that describes the details of how the data is stored in the computer along with their record structures, access paths and ordering. Only specialized or professional users can use these models. These data models can be divided into two types:

(a) Unifying Model.

(b) Frame Memory Model.

Record Based Data Models

READING TIME: 4 MINS

I should inform you that record based data models represent data by using the record structures. These are used to describe data at the conceptual view level. These are named because the database is structured in a fixed format records of several types. The use of fixed length records simplify the physical level implementation of the database. These models lie between the object based data models and the physical data models. These models provide the concepts that may be understood by the end users. These data models do not implement the full detail of the data storage on a computer system. Thus, these models are used to specify overall logical structure of the database and to provide high level description of implementation. These are generally used in traditional DBMS's and are also known as 'Representational Data Models'.

The various categories of record based data models are as follows:

(i) Hierarchical Data Model

(ii) Network Data Model

(iii) Relational Data Model.

Hierarchical data model

Hierarchical Data Model: You should be notified that Hierarchical Data Model is one of the oldest database models. The hierarchical model became popular with the introduction of IBM's Information Management System (IMS). The hierarchical data model organizes records in a tree structure i.e., hierarchy of parent and child records relationships. This model employs two main concepts: Record and Parent Child Relationship. A record is a collection of field values that provide information of an entity. A Parent Child Relationship type is a 1:N relationship between two record types. The record type of one side is called the parent record type and the one on the N side is called the child record type. In terms of tree data structure, a record type corresponds to node of a tree and relationship type corresponds to edge of the tree. The model requires that each child record can be linked to only one parent and child can only be reached through its parent.

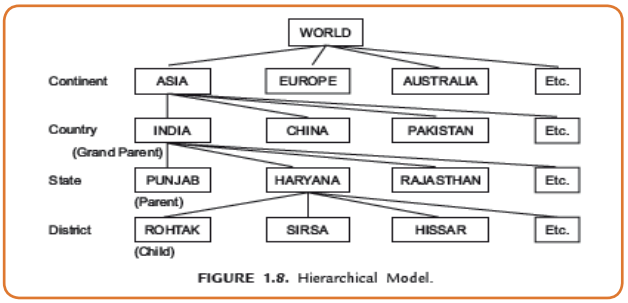


Figure 1: Hierarchical Model (Source: Begg & Connolly, 2002).

In the Figure 1 above, you note that the 'World' acts as a root of the tree structure which has many children's like Asia, Europe, Australia etc. These children can act as a parent for different countries such as ASIA continents acts as a parent for countries like India, China, Pakistan etc. Similarly, these children can act as a parent for different states such as INDIA country acts as a parent for states Punjab, Haryana, Rajasthan etc. Further the same follows. Consider child 'Rohta k' which has a parent 'Haryana' which further has a parent 'India' and so on. Now 'India' will acts a grandparent for the child 'Rohta k'. The major advantages of Hierarchical Model are that it is simple, efficient, maintains data integrity and is the first model that provides the concept of data security. The major disadvantages of Hierarchical model are that it is complex to implement, Lacking of structural independence, operational anomalies and data management problem.

Network Data Model

Network Data Model: I should tell you that as a result of limitations in the hierarchical model, designers developed the Network Model. The ability of this model to handle many to many (N : N) relations between its records is the main distinguishing feature from the hierarchical model. Thus, this model permits a child record to have more than one parent. In this model, directed graphs are used instead of tree structure in which a node can have more than one parent. This model was basically designed to handle non-hierarchical relationships.

The relationships between specific records of 1 : 1 (one to one), 1 : N (one to many) or N : N (many to many) are explicitly defined in database definition of this model. The Network Model was standardized as the Codasyl DBTG (Conference of Data System Languages, Database Task Group) model. There are two basic data structures in this model—Records and Sets. The record contains the detailed information regarding the data which are classified into record types. A set type represents relationship between record types and this model use linked lists to represent these relationships. Each set type definition consists of three basic elements: a name for set type an owner record type (like parent) and a member record type (like child). To represent many to many relationship in this model, the relationship is decomposed into two one to many (1 : N) relationships by introducing an additional record type called an Intersection Record or Connection Record. The major advantages of Network Model are that it is conceptually simple, handles more relationship types, promotes database integrity, data access flexibility and conformance to the standards. The major disadvantages of Network Model are that it is complex and lack of structural independence.

Relational Data Model

Relational data Model: It is important I let you know that The Relational Model was first introduced by Dr. Edgar Frank, an Oxford-trained Mathematician, while working in IBM Research Centre in 1970's. The Relational Model is considered one of the most popular developments in the database technology because it can be used for representing most of the real world objects and the relationships between them. The main significance of the model is the absolute separation of the logical view and the physical view of the data. The physical view in relational model is implementation dependent and not further defined. The logical view of data in relational model is set

oriented. A relational set is an unordered group of items. The field in the items are the columns. The column in a table have names. The rows are unordered and unnamed. A database consists of one or more tables plus a catalogue describing the database.

The relational model consists of three components:

1. A structural component—A set of tables (also called relations) and set of domains that defines the way data can be represented.

2. A set of rules for maintaining the integrity of the database.

3. A manipulative component consisting of a set of high-level operations which act upon and produce whole tables.

Bear in mind that in the relational model the data is represented in the form of tables which is used interchangeably with the word Relation. Each table consists of rows also known as tuples (A tuple represents a collection of information about an item, e.g., student record) and column also known as **attributes.** (An attribute represents the characteristics of an item, e.g., Student's Name and Phone No.). There are relationships existing between different tables. This model doesn't require any information that specifies how the data should be stored physically. The major advantages of Relational Model are that it is structurally independent, improved conceptual simplicity adhoc query capability and powerful DBMS. The major disadvantages of relational model are substantial hardware and software overhead and facilitates poor design and implementation.

Object Relational Model

You should note that Object Based Data Models are also known as conceptual models used for defining concepts including entries, attributes and relationships between them. These models are used in describing data at the logical and user view levels. These models allow the constraints to be specified on the data explicitly by the users. An entity is a distinct object which has existence in real world. . It will be implemented as a table in a database. An attribute is the property of an entity, in other words, attribute is a single atomic unit of information that describes something about its entity. It will be implemented as a column or field in the database. The associations or links between the various entities is known as relationships.

There are 4 types of object based data models. These are:

a. Entity-relationship (E-R) Model

b. Object-Oriented Model

c. Semantic Data Model

d. Functional Data Model

These are discussed as follows:

Entity-Relationship (E-R) Model: You should be aware that the E-R model is a high-level conceptual data model developed by Chen in 1976 to facilitate database design. The E-R model is the generalization of earlier available commercial model like the hierarchical and network model. It also allows the representation of the various constraints as well as their relationships. The relationship between entity sets is represented by a name. E-R relationship is of 1 : 1, 1 : N or N : N type which tells the mapping from one entity set to another. E-R model is shown diagrammatically using entity-relationship (E-R) diagrams which represents the elements of the conceptual model that show the meanings and relationships between those elements independent of any particular DBMS. The various features of E-R model are:

(i) E-R Model can be easily converted into relations (tables).

(ii) E-R Model is used for purpose of good database design by database developer.

(iii) It is helpful as a problem decomposition tool as it shows entities and the relationship between those entities.

(iv) It is an iterative process.

(v) It is very simple and easy to understand by various types of users.

I should also add that the major advantages of E-R model are that it is conceptually simple, have visual representation, an effective communication tool and can be integrated with the relational data model.

The major disadvantages of E-R model are that there is limited constraint representation, limited relationship representation, no data manipulation language and loss of information content.

Object-Oriented Data Model: Have it at the back of your mind that Object-oriented data model is a logical data model that captures the semantics of objects supported in an object-oriented programming. It is based on collection of objects, attributes and relationships which together form the static properties. It also consists of the integrity rules over objects and dynamic properties such as operations or rules defining new database states. An object is a collection of data and methods. When different objects of same type are grouped together they form a class. This model is used basically for multimedia applications as well as data with complex relationships. The object model is represented graphically with object diagrams containing object classes. Classes are arranged into hierarchies sharing common structure and behaviour and are associated with other classes.

Advantages of Object-Oriented Data Models

The various advantages of object-oriented data model are as follows:

(i) **Capability to handle various data types**: The object-oriented databases has the capability to store various types of data such as text, video pictures, voices etc.

(ii) **Improved data access**: Object oriented data models represent relationships explicitly. This improves the data access.

(iii) **Improved productivity**: Object-oriented data models provide various features such as inheritance, polymorphism and dynamic binding that allow the users to compose objects. These features increase the productivity of the database developer.

(iv) **Integrated application development system**: Object-oriented data model is capable of combining object-oriented programming with database technology which provides an integrated application development system.

Disadvantages of Object-Oriented Data Models

The various disadvantages of object-oriented data models are as follows:

(i) **Not suitable for all applications**: Object-oriented data models are used where there is a need to manage complex relationships among data objects. They are generally suited for applications such as e commerce, engineering and science etc. and not for all applications.

(ii) **No precise definition**: It is difficult to define what constitutes an object-oriented DBMS since the name has been applicable to wide variety of products.

(iii) **Difficult to maintain**: The definition of object is required to be changed periodically and migration of existing databases to confirm to the new object definition. It creates problems when changing object definitions and migrating databases.

(c) **Semantic Data Models**: These models are used to express greater interdependencies among entities of interest. These interdependencies enable the models to represent the semantic of the data in the database. This class of data models are influenced by the work done by artificial intelligence researchers. Semantic data models are developed to organize and represent knowledge but not data. This type of data models are able to express greater interdependencies among entities of interest. Mainframe database are increasingly adopting semantic data models. Also, its growth usage is seen in PC's. In coming times database management systems will be partially or fully intelligent.

(d) **Functional Data Model**: The functional data model describes those aspects of a system concerned with transformation of values-functions, mappings, constraints and functional dependencies. The functional data model describes the computations within a system. It shows how output value in computation are derived from input values without regard for the order in which the values are computed. It also includes constraints among values. It consists of multiple data flow diagrams. Data flow diagrams show the dependencies between values and computation of output values from input values and functions, without regard for when the functions are executed. Traditional computing concepts such as expression trees are examples of functional models.

Comparism of various data models [SAQ1, 2]

READING TIME: 1 MIN

The most commonly used data models are compared on the basis of various properties. The comparison table is given table 1.

Property	Hierarchical	Network	Relational	E-R Diagram	Object-oriented
1. Data element organization	Files, records	Files, records	Tables/tuples	Objects, entity sets	Objects
2. Identity	Record based	Record based	Value based	Value based	Record based
3. Data Independence	Yes	Yes	Yes	Yes	Yes
4. Relationship Organization	Logical proximity in a linearised tree.	Intersecting Networks	Identifiers of rows in one table are embedded as attribute values in another table.	Relational extenders that support specialized applications.	Logical containment
5. Access Language	Procedural	Procedural	Non-procedural	Non-procedural	Procedural
6. Structural Independence	No	No	Yes	Yes	Yes

Table 1: Comparison Table

Summary

In this unit we learn about the different categories of data model. We also focused on record based models. The different types of record based models were examined. Their components and features. Analysis of object oriented data models.

Self-Assessment Questions

1. Explain the difference between the types of record based models
2. Explain the difference between the types of data models.

Tutor Marked Assignment

1. Define data model?
2. What are the categories of data models?
3. Explain the different types of record based models.

References

Elmasri R. and Navathe S. B., (2011). Fundamentals of Database Systems. Addison-Wesley.

Gupta S. B. and Mittal A., (2004). Introduction to database management systems. Pearson Education.

Ramakrishnan R. and Gherke J. (1996). Database Management System, second edition. McGraw-Hill Higher Education.

Further Reading

Silberschatz A., Korth H. F., and Sudarshan S. (2003). Database System Concepts. 2

time lapse of blue lights
source: Pexels by Pixabay

UNIT 3

ENTITY-RELATIONSHIP MODEL

Introduction

You should be informed that data model tells how the logical structure of a database is modeled. Data Models are fundamental entities to introduce abstraction in DBMS. Data models define how data is connected to each other and how it will be processed and stored inside the system. The very first data model could be flat data-models where all the data used to be kept in same plane. Because earlier data models were not so scientific they were prone to introduce lots of duplication and update anomalies.

Learning Outcomes

When you have studied this unit, you should be able to:

- Illustrate
- Explain the relational set
- Describe mapping constraints
- Explain role and Recursive Relationship Set

time lapse of blue lights
source: Pexels by Pixabay

Main Content

Entity-Relationship model [SAQ1]

READING TIME: 1 MIN

Entity-Relationship (E-R) Model: You should be of notice that the E-R model is a high level conceptual data model developed by Chen in 1976 to facilitate database design. The E-R model is the generalization of earlier available commercial model like the hierarchical and network model. It also allows the representation of the various constraints as well as their relationships. The relationship between entity sets is represented by a name. E-R relationship is of 1 : 1, 1 : N or N : N type which tells the mapping from one entity set to another. E-R model is shown diagrammatically using entity-relationship (E-R) diagrams which represents the elements of the conceptual model that show the meanings and relationships between those elements independent of any particular DBMS. The various features of E-R models are:

(i) E-R Model can be easily converted into relations (tables).

(ii) E-R Model is used for purpose of good database design by database developer.

(iii) It is helpful as a problem decomposition tool as it shows entities and the relationship between those entities.

(iv) It is an iterative process.

(v) It is very simple and easy to understand by various types of users.

The major advantages of E-R model are that it is conceptually simple, have Vishal representation, an effective communication tool and can be integrated with the relational data model. The major disadvantages of E-R model are that there are limited constraint representation, limited relationship representation, no data manipulation

language and loss of information content.

Relationship Sets (1min)

READING TIME: 1 MIN

1. Relationship: A relationship is the association among several entities. It connects different entities through a meaningful relation.

2. Relationship Set: A relationship set is a set of relationships of the same type. Consider an example, employees work in different departments. Then relationship exists between employees and departments because each employee must belongs to some department. Relation of all employees with department when combined makes the relationship set because each employee has same kind of relation with departments.

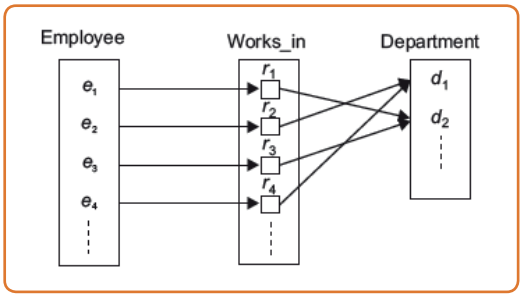


Figure 1: Binary Relationship Set
(Source: Elmasri & Navathe, 2011)

Here, Employee and Department are two entity sets. r stands for relationship between Employee and Department. Works_in is the relationship set as shown in Figure 1.

Descriptive Attributes: Attributes of any relationship set are known as descriptive attributes.

Degree of Relationship Set [SAQ2]

READING TIME: 1 MINS

Total number of entity sets participate in a relationship set is known as degree of that relationship set.

1. Binary Relationship Set

A relationship set in which only two entity sets are involved is known as binary relationship set. Ex. The Figure 13 shows the Binary relationship set.

2. Ternary Relationship Set

A relationship set in which three entity sets are involved is known as ternary relationship set or a relationship set having degree three. Ex. The Figure 2 shows the relationship set works_in, which is a ternary relationship set.

Role and Recursive Relationship Set (1min)

Role: The function of any entity which it plays in relationship set is called that entity's role. e.g., employee plays the role of worker in his department in Figure 2.

Recursive Relationship Set : You should relate that when the same entity sets participate in same relationship set more than once with different roles each time, then this type of recursive relationship set is known as Recursive Relationship set. e.g., consider an example of relationship set works in and two entity set student and college. A student who attends weekend classes in college as student may also be lecturer in that college. Then this person plays two roles (student, faculty) in same relationship set works in.

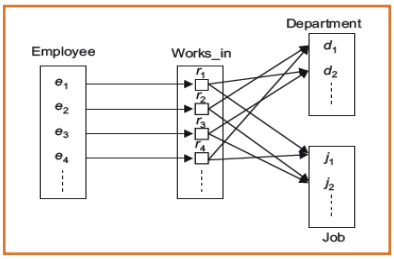


Figure 2: Ternary Relationship Set (Source: Elmasri & Navathe, 2011)

Mapping Constraints

READING TIME: 1 MIN

There are certain constraints in E-R model. Data in the database must follow the constraints. Constraints act as rules to which the contents of database must conform. There are two types of mapping constraints : (a) Mapping cardinalities, (b) Participation constraints.

Mapping Cardinalities (Cardinality Ratios)

It specifies the number of entities of an entity set that are associated with entities of another entity set through a relationship set. Mapping Cardinalities are helpful in describing binary relationship sets. Two entity sets X and Y having binary relationship set R must have one of the following mapping cardinality:

1. One to One (1 : 1) : An entity in X is associated with at most one entity in Y and an entity in Y is associated with at most one entity in X. A country has only one president. Any person may be the president of at most one country.

2. One to Many (1 : N) : An entity in X is associated with any number of entities in Y. An entity in Y is associated with at most one entity in X. manager has many employees under it but an employee works under only one manager.

3. Many to One (N : 1) : An entity in X is associated with at most one entity in Y. An entity in Y is associated with any number of entities in X. A employee can work on single project while any project can be assigned to more than one employee.

4. Many to Many (M : N) : An entity in X is associated with any number (zero or more) of entities in Y and vice versa. A student can have more than one subject and one subject can be given to more than one student.

Summary

In this unit, we have examined the E-R model. The relationship set and degree attributes was also explained. The different degree of relationship set. The binary relationship set and the tertiary relationship set. Role and recursive relationship set was also examined. The different mapping cardinalities was explained.

Self-Assessment Questions

1. Explain what you understand by E-R model.
2. Distinguish between binary relationship set and tertiary relationship set.

Tutor Marked Assignment

Explain the different cardinalities with example.

What do you understand by binary relationship set and tertiary relationship set.

References

Elmasri R. and Navathe S. B., (2011). Fundamentals of Database Systems. Addison-Wesley.

Gupta S. B. and Mittal A., (2004). Introduction to database management systems. Pearson Education.

Ramakrishnan R. and Gherke J. (1996). Database Management System, second edition. McGraw-Hill Higher Education.

Further Reading

Silberschatz A., Korth H. F., and Sudarshan S. (2003). Database System Concepts. McGraw-Hill Education.

Begg C. E. and Connolly T. M. (2002). Database Systems: A Practical Approach to Design, Implementation, and Management, Addison-Wesley.

data server ports
source: Pexels by Panumas Nikhomkai

UNIT 4

DOMAINS, ATTRIBUTES, TUPLES AND RELATIONS

Introduction

This unit introduces you to the relational model. The relational model represents the database as a collection of relations. A relation is made up of attributes and tuples which represent the values of entries made into the database.

Learning Outcomes

When you have studied this unit, you should be able to:

- Distinguish between Domain, Attribute, Tuple and Relation.
- Create Database Tables with respective domain, attribute, tuple and relation.

data server ports
source: Pexels by Panumas Nikhomkai

Main Content

Domain, Attribute, Tuple and Relation [SAQ1, 2]

READING TIME: 4 MINS

I should point out that a **domain** DD is a set of atomic values. By **atomic** we mean that each value in the domain is indivisible as far as the formal relational model is concerned. A common method of specifying a domain is to specify a data type from which the data values forming the domain are drawn. It is also useful to specify a name for the domain, to help in interpreting its values. You should note some examples of domains below:

i. Usa_phone_numbers. The set of ten-digit phone numbers valid in the United States.

ii. Local_phone_numbers. The set of seven-digit phone numbers valid within a particular area code in the United States. The use of local phone numbers is quickly becoming obsolete, being replaced by standard ten-digit numbers.

iii.Social_security_numbers. The set of valid nine-digit Social Security numbers.(This is a unique identifier assigned to each person in the United States for employment, tax, and benefits purposes.)

iv. Names: The set of character strings that represent names of persons.

v. Grade_point_averages. Possible values of computed grade point averages; each must be a real (floating-point) number between 0 and 4.

vi. Employee_ages. Possible ages of employees in a company; each must be an integer value between 15 and 80.

vii. Academic_department_names. The set of academic department names in a university, such as Computer Science, Economics, and Physics.

'PHYS'.

You should know that the proceedings are called logical definitions of domains. A data type or format is also specified for each domain. For example, the data type for the domain Usa_phone_numbers can be declared as a character string of the form (ddd) ddddddd, where each d is a numeric (decimal) digit and the first three digits form a valid telephone area code. The data type for Employee_ages is an integer number between 15 and 80. For Academic_department_names, the data type is the set of all character strings that represent valid department names. A domain is thus given a name, data type, and format. Additional information for interpreting the values of a domain can also be given; for example, a numeric domain such as Person_weights should have the units of measurement, such as pounds or kilograms.

A relation schemaRR, denoted by $R(A1,A2,\ldots,An)R(A1,A2,\ldots,An)$, is made up of a relation name $RR$and a list of attributes, $A_1,A_1,\ldots,A_nA_1,A_1,\ldots,A_n$. Each attribute A_i $A_i$is the name of a role playedby some domain DDin the relation schema RR. $DD$is called the domain of $A_iA_i$and isdenoted by dom(A_1A_1). A relation schema is used to describe a relation; RRiscalled thename of this relation. The degree (or arity) of a relation is the number of attributes$nn$of its relation schema.

A relation of degree seven, which stores information about university students,would contain sevenattributes describing each student. as follows:

STUDENT(Name, Ssn, Home_phone, Address, Office_phone, Age, Gpa).

Using the data type of each attribute, the definition is sometimes written as:

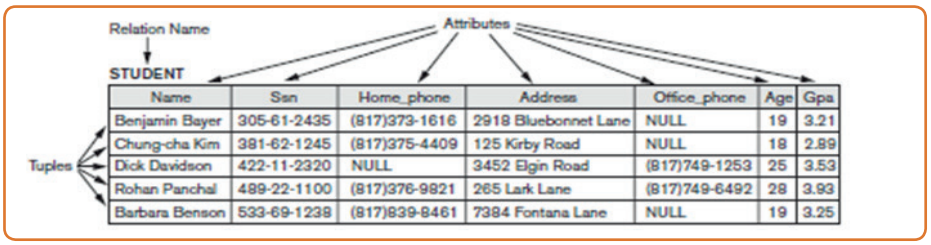
STUDENT(Name: string, Ssn: string, Home_phone: string, Address: string,Office_ phone: string, Age: integer, Gpa: real)

For this relation schema, STUDENT is the name of the relation, which has sevenattributes. In the preceding definition, we showed assignment of generic types suchas string or integer to the attributes. More precisely,we can specify the followingpreviously defined domains for some of the attributes of the STUDENT relation:

dom(Name) = Names; dom(Ssn) = Social_security_numbers; dom(HomePhone) = USA_phone_numbers3, dom(Office_phone) = USA_phone_numbers, and dom(Gpa) = Grade_point_averages. It is also possible to refer to attributes of a relation schema bytheir position within therelation; thus, the second attribute of the STUDENT relationis Ssn, whereas the fourth attribute is Address.

A relation (or relation state) $rr$of the relation schema $R(A_1,A_2,\ldots,A_n)R(A_1,A_2,\ldots,A_n)$, also denoted by $r(R)r(R)$, is a set of n-tuples $r = \{t_1,t_2,\ldots,t_m\}r = \{t_1,t_2,\ldots,t_m\}$. Each n-tuple $tt$is an ordered list of nnvalues $t =< v_1,v_2,\ldots,v_n >t =< v_1,v_2,\ldots,v_n >$, where each value v_iv_i, $1 \leq v_iv_i \leq v_nv_n$, is an element of dom($A_iA_i$) or is a special NULL value. The $ithith$ value in tuple tt, which corresponds to the attribute A_iA_i, is referred to as $t[A_i]t[A_i]$ or $t.A_it.A_i$(or $t[i]t[i]$ if we use the positional notation). The terms relation intension for the schema $RR$and relation extension for a relation state $r(R)r(R)$are also commonly used.

Figure 1 show us an example of a STUDENT relation, which corresponds to the STUDENT schema just specified. Each tuple in the relation represents a particular student entity (or object). We display the relation as a table, where each tuple is shown as a row and each attribute corresponds to a column header indicating a role or interpretation of the values in that column. NULL values represent attributes whose values are unknown or do not exist for some individual STUDENT tuple. See figure 1.



Name	Ssn	Home_phone	Address	Office_phone	Age	Gpa
Benjamin Bayer	305-61-2435	(817)373-1616	2918 Bluebonnet Lane	NULL	19	3.21
Chung-cha Kim	381-62-1245	(817)375-4409	125 Kirby Road	NULL	18	2.89
Dick Davidson	422-11-2320	NULL	3452 Elgin Road	(817)749-1253	25	3.53
Rohan Panchal	489-22-1100	(817)376-9821	265 Lark Lane	(817)749-6492	28	3.93
Barbara Benson	533-69-1238	(817)839-8461	7384 Fontana Lane	NULL	19	3.25

Figure 1: The attributes and tuples of a relation STUDENT (Source: Silberschatz, Korth & Sudarshan, 2003.

The earlier definition of a relation can be restated more formally using set theory concepts as follows. A relation (or relation state) $r(R)r(R)$ is a mathematical relation of degree n on the domains dom(A_1A_1), dom(A_2A_2), ...,dom(A_nA_n), which is a sub-set of the Cartesian product (denoted by *) of the domains that define $R: r(R)R: r(R)$ $\subseteq$(dom(A_1A_1) *dom(A_2A_2) *... *dom(A_nA_n)). This product of cardinalities of all do-mains represents the total number of possible instances or tuples that can ever exist in any relation state $r(R)r(R)$. Of all these possible combinations, a relation state at a given time—the current relation state—reflects only the valid tuples that represent a particular state of the real world. In general, as the state of the real-world changes, so does the relation state, by being transformed into another relation state. However, the schema R is relatively static and changes very infrequently—for example, as a

result of adding an attribute to represent new information that was not originally stored in the relation. It is possible for several attributes to have the same domain. he attribute names indicate different roles, or interpretations, for the domain. For example, in the STUDENT relation, the same domain USA_phone_numbersplays the role of Home_phone, referring to the home phone of a student, and the role of Office_phone,referring to the office phone of the student. A third possible attribute (not shown) with the same domain could be Mobile_phone.

Characteristics of Relation

READING TIME: 4 MINS

The earlier definition of relations implies certain characteristics that make a relation different from a file or a table. We now discuss some of these characteristics.

Ordering of Tuples in a Relation

It is cogent you note that a relation is defined as a set of tuples. Mathematically, elements of a set have no order among them; hence, tuples in a relation do not have any particular order. In other words, a relation is not sensitive to the ordering of tuples. However, in a file, records are physically stored on disk (or in memory), so there always is an order among the records. This ordering indicates first, second, $ithith$, and last records in the file. Similarly, when we display a relation as a table, the rows are displayed in a certain order. Tuple ordering is not part of a relation definition because a relation attempts to represent facts at a logical or abstract level. Many tuple orders can be specified on the same relation. There is no preference for one ordering over another. Hence, the relation displayed in Figure 3 is considered identical to the one shown in Figure 1.When a relation is implemented as a file or displayed as a table, a particular ordering may be specified on the records of the file or the rows of the table.

Ordering of Values within a Tuple and an Alternative Definition of a Relation

Take note of the preceding definition of a relation, an n-tuple is an ordered list of n $n$values, so the ordering of values in a tuple—and hence of attributes in a relation schema—is important. However, at a more abstract level, the order of attributes and their values is not that important as long as the correspondence between attributes and values is maintained. Alternative definition of a relation can be given, making the ordering of values in a tuple unnecessary.

In this definition, a relation schema $R = \{A_1, A_2, \ldots, A_n\}$ is a set of attributes (instead of a list), and a relation state $r(R)r(R)$ is a finite set of mappings $r = \{t_1, t_2, \ldots, t_m\}$ $r = \{t_1, t_2, \ldots, t_m\}$, where each tuple $t_i t_i$is a mapping from $RR$to DD,and D is the union(denoted by ∪) of the attribute domains; that is, D = dom($A_1 A_1$) ∪dom($A_2 A_2$) ∪... ∪dom($A_n A_n$). In this definition, $t[A_i]t[A_i]$ must be in $dom(A_i)\,dom(A_i)$ for 1 ≤i ≤nfor each mappingttin rr. Each mapping $t_i t_i$is called a tuple.

According to this definition of tuple as a mapping, a tuple can be considered as a set of (<attribute>, <value>) pairs, where each pair gives the value of the mapping from an attribute $A_i A_i$to a value $v_i v_i$from $dom(A_i)dom(A_i)$. The ordering of attributes is not important, because the attribute name appears with its value. By this definition, the two tuples shown in Figure 2 are identical. This makes sense at an abstract level, since there really is no reason to prefer having one attribute value appear before another in a tuple.

STUDENT

Name	Ssn	Home_phone	Address	Office_phone	Age	Gpa
Dick Davidson	422-11-2320	NULL	3452 Elgin Road	(817)749-1253	25	3.53
Barbara Benson	533-69-1238	(817)839-8461	7384 Fontana Lane	NULL	19	3.25
Rohan Panchal	489-22-1100	(817)376-9821	265 Lark Lane	(817)749-6492	28	3.93
Chung-cha Kim	381-62-1245	(817)375-4409	125 Kirby Road	NULL	18	2.89
Benjamin Bayer	305-61-2435	(817)373-1616	2918 Bluebonnet Lane	NULL	19	3.21

Figure 2: The relation STUDENT from Figure 1 with a different order of tuples (Source: Silberschatz, Korth & Sudarshan, 2003).

I wish to let you know as well that when a relation is implemented as a file, the attributes are physically ordered as fields within a record. We will generally use the first definition of relation, where the attributes and the values within tuples are ordered, because it simplifies much of the notation. However, the alternative definition given here is more general. Values and NULLs in the Tuples. Each value in a tuple is an atomic value; that is, it is not divisible into components within the framework of the basic relational model. Hence, composite and multivalued attributes are not allowed. This model is sometimes called the flat relational model. Much of the theory behind the relational model was developed with this assumption in mind, which is called the first normal form assumption. Hence, multivalued attributes must be represented by separate relations, and composite attributes are represented only by their simple

component attributes in the basic relational model.

An important concept is that of NULL values, which are used to represent the values of attributes that may be unknown or may not apply to a tuple. A special value, called NULL, is used in these cases. For example, in Figure 1, some STUDENT tuples have NULL for their office phones because they do not have an office (that is, office phone does not apply to these students). Another student has a NULL for home phone, presumably because either he does not have a home phone or he has one but we do not know it (value is unknown). In general, we can have several meanings for NULL values, such as value unknown, value exists but is not available, or attribute does not apply to this tuple (also known as value undefined). An example of the last type of NULL will occur if we add an attribute Visa_status to the STUDENT relation When a relation is implemented as a file, the attributes are physically ordered as fields within a record. We will generally use the first definition of relation, where the attributes and the values within tuples are ordered, because it simplifies much of the notation. However, the alternative definition given here is more general.

Values and NULLs in the Tuples. Each value in a tuple is an atomic value; that is, it is not divisible into components within the framework of the basic relational model. Hence, composite and multivalued attributes are not allowed. This model is sometimes called the flat relational model. Much of the theory behind the relational model was developed with this assumption in mind, which is called the first normal form assumption.

Hence, multivalued attributes must be represented by separate relations, and composite attributes are represented only by their simple component attributes in the basic relational model. An important concept is that of NULL values, which are used to represent the values of attributes that may be unknown or may not apply to a tuple. A special value, called NULL, is used in these cases. Another student has a NULL for home phone, presumably because either he does not have a home phone or he has one but we do not know it (value is unknown). In general, we can have several meanings for NULL values, such as value unknown, value exists but is not available, or attribute does not apply to this tuple (also known as value undefined). An example of the last type of NULL will occur if we add an attribute Visa_status to the STUDENT

relation that applies only to tuples representing foreign students. See figure 3.

The exact meaning of a NULL value governs how it fares during arithmetic aggregations or comparisons with other values. For example, a comparison of two NULLvalues leads to ambiguities—if both Customer A and B have NULL addresses, it does not mean they have the same address. During database design, it is best to avoid NULL values as much as possible. It is possible to devise different codes for different meanings of NULL values. Incorporating different types of NULL values into relational model operations has proven difficult and is outside the scope of our presentation.

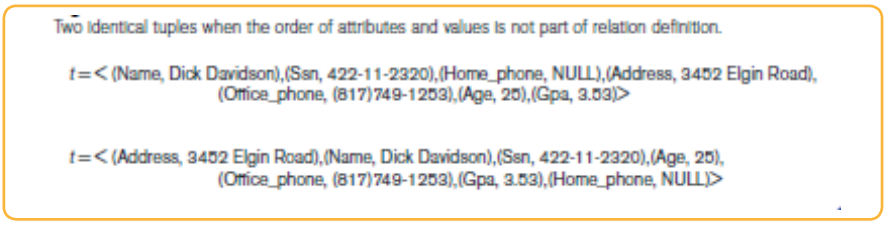


Figure 3: Relation with two NULL values (Source: Silberschatz, Korth & Sudarshan, 2003).

Interpretation (meaning) of a Relation

READING TIME: 1 MIN

Try to understand that the relation schema can be interpreted as a declaration or a type of assertion. Let us take for example, the first tuple in Figure 1 asserts the fact that there is a STUDENT whose Name is Benjamin Bayer, Ssn is 305-61-2435, Age is 19, and so on.

Hope you noticed that that some relations may represent facts about entities, whereas other relations may represent facts about relationships? Let us take another example, a relation schema MAJORS(Student_ssn, Department_code) asserts that students major in academic disciplines.

A tuple in this relation relates a student to his or her major discipline. Hence, the relational model represents facts about both entities and relationships uniformly as relations. This sometimes compromises understandability because one has to guess whether a relation represents an entity type or a relationship type.

You should be aware that an alternative interpretation of a relation schema is as a predicate; in this case, the values in each tuple are interpreted as values that satisfy the predicate. For example, the predicate STUDENT (Name, Ssn, ...) is true for the five tuples in relation STUDENT of Figure 1.

These tuples represent five different propositions or facts in the real world. This interpretation is quite useful in the context of logical programming languages, such as Prolog, because it allows the relational model to be used within these languages. An assumption called the closed world assumption states that the only true facts in the universe are those present within the extension (state) of the relation(s). Any other combination of values makes the predicate false.

Summary

In this unit, I explained the relationship between domain, attribute, tuple and relation.

Self-Assessment Questions

1. What is a Domain?
2. What is an Attribute?

Tutor Marked Assignment

Create a database for Student records using six (6) relevant tuples and Attributes.

References

Chao L. (2006). Database Development and Management. Taylor & Francis Group, LLC. ISBN 0-8493-3318-0.

Ozzu M. T. (2012). Overview of Database Management.

David R. Cheriton School of Computer Science, University of Waterloo.

Further Reading

Kahate A. (20040. Introduction to Database Management Systems. Pearson Education. ISBN 9788131700785, eISBN 9788131775820

Conger S. (2012). Hands-on Database design and Development. Pearson education. ISBN 13: 978-0-13-610827-6, ISBN 10: 0-13-61827-X

laptop on a desk
source: Unsplash by Ivan Shilov

MODULE 4

DATABASE DESIGN

UNITS

Ethernet cables
Source: Pexels by brett Sayles

UNIT 1

FUNCTIONAL DEPENDENCIES

Introduction

In this unit, we will discuss functional dependencies and its types. Thus, I should let you know that normalization is based on the analysis of functional dependencies. A functional dependency is a constraint between two attributes or two sets of attributes. The purpose of the database design is to arrange the various data items into an organized structure so that it generates set of relationships and stores the information without any repetition. A bad database design may result into redundant and spurious data and information. In this section, we will discuss functional dependency and its types.

Learning Outcomes

At the end of this unit, you should be able to:

- Describe functional dependency.
- Draw a functional dependency chart/diagram.
- Distinguish between the different types of functional dependency.

Ethernet cables
Source: Pexels by brett Sayles

Main Content

Functional Dependencies

READING TIME: 1 MIN

Kindly note that functional dependencies are the result of interrelationship between attributes or in between tuples in any relation.

Definition : In relation R, X and Y are the two subsets of the set of attributes, Y is said to be functionally dependent on X if a given value of X (all attributes in X) uniquely determines the value of Y (all attributes in Y).

It is denoted by X → Y(Y depends upon X).

Determinant : Here X is known as determinant of functional dependency.

Consider the example of Employee relation as shown in figure 1:

Employee

EID	Name	Salary
1	Aditya	15,000
2	Manoj	16,000
3	Sandeep	9,000
4	Vikas	10,000
5	Manoj	9,000

Figure 1: Employee Relation (Source: Ramakrishnan & Gherke, 1996)

In Employee relation, EID is primary key. Suppose you want to know the name and salary of any employee. If you have EID of that employee, then you can easily find information of that employee. So, Name and Salary attributes depend upon EID attribute.

Here, X is (EID) and Y is (Name, Salary)
X (EID): Y (Name, Salary)
The determinant is EID

Functional Dependency Chart/Diagram

READING TIME: 1 MIN

It is the graphical representation of function dependencies among attributes in any relation.

The following four steps are followed to draw FD chart.

1. Find out the primary key attributes.

2. Make a rectangle and write all primary key attributes inside it.

3. Write all non-prime key attributes outside the rectangle.

4. Use arrows to show functional dependency among attributes.

Let us consider the example of Figure 1. Its functional dependency chart is shown in Figure 2. It is easier to remember all dependencies by making FD charts.

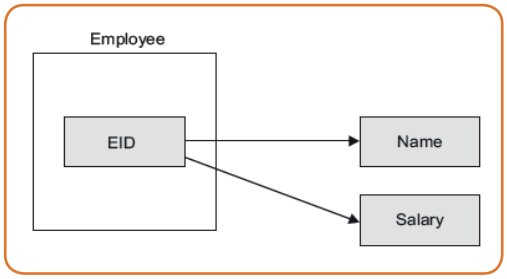


Figure 2: Functional Dependency chart of Employee Relation (Source: Ramakrishnan & Gherke, 1996)

Types of Functional Dependencies [SAQ1, 2]

READING TIME: 3 MINS

You should note that there are four major types of FD's:

1. Partial Dependency and Fully Functional Dependency

Partial dependency: Suppose you have more than one attributes in primary key. Let A be the non-prime key attribute. If A is not dependent upon all prime key attributes then partial dependency exists.
Fully functional dependency : Let A be the non-prime key attribute and value of A is dependent upon all prime key attributes.

Then A is said to be fully functional dependent. Consider a relation student having prime key attributes (RollNo and Game) and non-prime key attributes (Grade, Name and Fee).

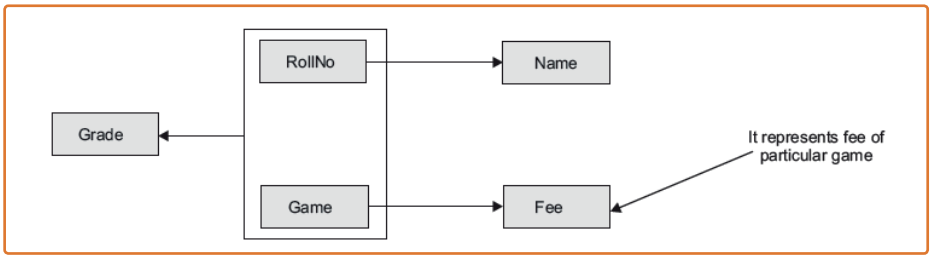


Figure 3: Functional dependency chart showing partial and fully functional dependency of student relation (Source: Ramakrishnan & Gherke, 1996).

As shown in Figure 3, Name and Fee are partially dependent because you can find the name of student by his RollNo. and fee of any game by name of the game. Grade is fully functionally dependent because you can find the grade of any student in a particular game if you know RollNo. and Game of that student. Partial dependency is due to more than one prime key attribute.

2. Transitive Dependency and Non-transitive Dependency

Transitive dependency: Transitive dependency is due to dependency between non-prime key attributes. Suppose in a relation R, X → Y (Y depen upon X), Y → Z (Z depends upon Y), then X → Z (Z depends upon X). Therefore, Z is said to be transitively dependent upon X.

Non-transitive dependency: Any functional dependency which is not transitive is known as Non-transitive dependency. Non-transitive dependency exists if there is no dependency between non-prime key attributes.

Let us consider a relation student (whose functional dependency chart is shown in Figure 4) having prime key attribute (RollNo) and non-prime key attributes (Name, Semester, Hostel).
Try to understand that for each semester there is different hostel Here Hostel is transitively dependent upon RollNo. Semester of any student can be found by his RollNo. Hostel can be found out by semester of student. Here, Name is non-transitively dependent upon RollNo.

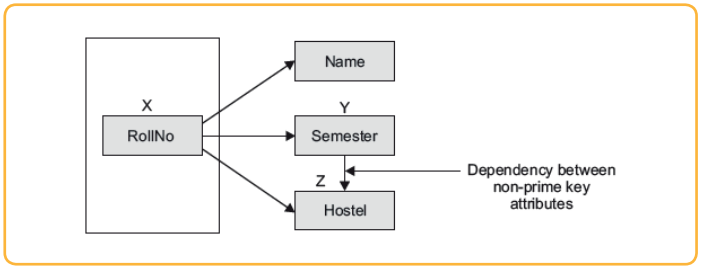


Figure 4: Functional dependency chart showing transitive and non-transitive dependency on relation student (Source: Ramakrishnan & Gherke, 1996).

3. Single Valued Dependency and Multivalued Dependency

Single valued dependency : In any relation R, if for a particular value of X, Y has single value then it is known as single valued dependency.

Multivalued dependency (MVD) : In any relation R, if for a particular value of X, Y has more then one value, then it is known as multivalued dependency. It is denoted by X →→ Y.

Consider the relation Teacher shown in Figure 5(a) and its FD chart shown in Figure 5(b).

ID	Teacher	Class	Days
1Z	Sam	Computer	1
2Z	John	Computer	6
1Z	Sam	Electronics	3
2Z	John	Mechanical	5
3Z	Nick	Mechanical	2

(a)

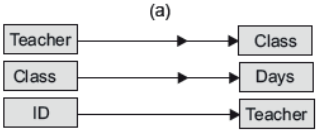


Figure 5: Functional dependency chart showing single valued and multivalued dependency on relation teacher (Source: Ramakrishnan & Gherke, 1996).

You should note that there is MVD between Teacher and Class because a teacher can take more than one class. There is another MVD between Class and Days because a class can be on more than one day. There is single valued dependency between ID and Teacher because each teacher has a unique ID.

4. Trival Dependency and Non-trival Dependency

Trival FD : In any relation R, X → Y is trival if Y ⊆ X (Y is the subset of X).
Non-trival FD : In any relation R, X →Y is non-trival if Y ⊈ X (Y is not the subset of X).
Consider the Supplier-Product relation shown in Figure 6.

S#	City	P#	Quantity
1	Delhi	1P	100
2	Rohtak	8P	200
3	Pune	3P	50
4	Pune	5P	75
5	Rohtak	1P	99
6	Mau	5P	105

Figure 6. Supplier-product relation (Source: Ramakrishnan & Gherke, 1996)

Here, (S#, P#) : S# is trival FD
S# : Supplier ID
P# : Product ID
Example. Consider the given relation R:

O	P	Q
1	2	3
2	1	7
2	1	5
7	3	8

Write all non-trivial functional dependencies satisfied by R. Also give an example of an FD that does not hold on R.
Solution: For functional dependencies look at the data given in the table.
– The FD O → P holds on R, since for each value of O there is a single value of P.

– The FD O → Q does not hold on R, since in the 2nd and 3rd row, O has the same value, but Q has different values.

The non-trivial functional dependencies are as follows:

O → P, Q → O, Q → P, OQ → P, PQ → O, Q → OP, O → OP, OQ → PQ,

OQ → OPQ, Q → OQ, PQ → OPQ, Q → OPQ, P → O,

Summary

In this unit, we discussed about functional dependency explanation. The functional dependency chart/diagram was also examined. The rules for drawing a functional dependency chart was stated and an example of a drawn functional dependency chart was stated. The different types of functional dependency were also examined.

Self-Assessment Questions

1. What do you understand by the term functional dependency?
2. Tutor Marked Assignment
3. State the different types of functional dependencies.
4. Give an example of a functional dependency chart.

Tutor Marked Assignment

State the different types of functional dependencies.

Give an example of a functional dependency chart.

References

Elmasri R. and Navathe S. B., (2011). Fundamentals of Database Systems. Addison-Wesley.

Gupta S. B. and Mittal A., (2004). Introduction to database management systems. Pearson Education.

Ramakrishnan R. and Gherke J. (1996). Database Management System, second edition. McGraw-Hill Higher Education.

Further Reading

Silberschatz A., Korth H. F., and Sudarshan S. (2003). Database System Concepts. McGraw-Hill Education.

en.wikipedia.org/wiki/Database_system Accessed 23rd September 2018.

en.wikipedia.org/wiki/Database_management_system Accessed 23rd September 2018.

Tools on the table
Source:pexels by tima miroshnichenko

UNIT 2

DECOMPOSITION

Introduction

We now present an overview of the problems that schema refinement is intended to address and a refinement approach based on decompositions. Redundant storage of information is the root cause of these problems. Although decomposition can eliminate redundancy, it can lead to problems of its own and should be used with caution.

Learning Outcomes

At the end of this section, the concept of decomposition should be understood. Also, the reason why decomposition is needed should also be addressed.

- Redundant storage: Some information is stored repeatedly.
- Update anomalies: If one copy of such repeated data is updated, an inconsistency is created unless all copies are similarly updated.
- Insertion anomalies: It may not be possible to store some information unless some other information is stored as well.
- Deletion anomalies: It may not be possible to delete some information without losing some other information as well.

Tools on the table
Source:pexels by tima miroshnichenko

Main Content

Redundancy

READING TIME: 1 MIN

Most of you know the literal meaning of redundancy but to refresh your memory, Redundancy means unwanted and uncontrolled duplication of data. Redundancy not only leads to duplication of data, it also has other side effects such as loss of data integrity and data consistency. Storing the same information redundantly, that is, in more than one place within a database, can lead to several problems:

i. Redundant storage: Some information is stored repeatedly.

ii. Update anomalies: If one copy of such repeated data is updated, an inconsistency is created unless all copies are similarly updated.

iii. Insertion anomalies: It may not be possible to store some information unless some other information is stored as well.

iv. Deletion anomalies: It may not be possible to delete some information without losing some other information as well.

Uses of Decomposition [SAQ1]

READING TIME: 1 MIN

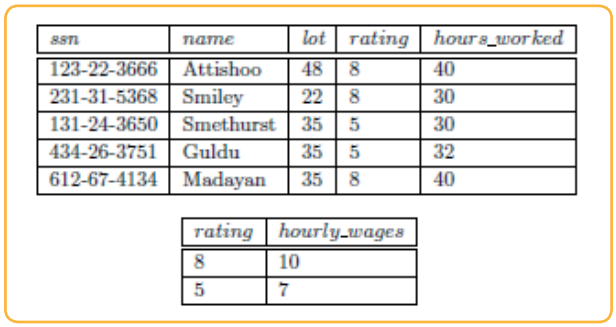
Intuitively, you should note that redundancy arises when a relational schema forces an association between attributes that is not natural. Functional dependencies (and, for that matter, other ICs) can be used to identify such situations and to suggest refinements to the schema. The essential idea is that many problems arising from redundancy can be addressed by replacing a relation with a collection of `smaller' relations. Each of the smaller relations contains a (strict) subset of the attributes of the original relation. We refer to this process as decomposition of the larger relation into the smaller relations.

We can deal with the redundancy in Hourly Emps by decomposing it into two relations:

Hourly Emps2(ssn, name, lot, rating, hours worked)

Wages(rating, hourly wages)

The instances of these relations corresponding to the instance of Hourly Emps is shown in Figure 1.



ssn	*name*	*lot*	*rating*	*hours_worked*
123-22-3666	Attishoo	48	8	40
231-31-5368	Smiley	22	8	30
131-24-3650	Smethurst	35	5	30
434-26-3751	Guldu	35	5	32
612-67-4134	Madayan	35	8	40

rating	*hourly_wages*
8	10
5	7

Figure 1: Instances of Hourly Emps2 and Wages
(Source: Ramakrishnan & Gherke, 1996)

Note that we can easily record the hourly wage for any rating simply by adding a tuple to Wages, even if no employee with that rating appears in the current instance of Hourly Emps. Changing the wage associated with a rating involves updating a single Wages tuple. This is more efficient than updating several tuples (as in the original design), and it also eliminates the potential for inconsistency. Notice that the insertion and deletion anomalies have also been eliminated.

Problems related to Decomposition [SAQ2]

READING TIME: 1 MIN

Unless we are careful, decomposing a relation schema can create more problems than it solves. Two important questions must be asked repeatedly:

1. Do we need to decompose a relation?

2. What problems (if any) does a given decomposition cause?

To help with the first question, several normal forms have been proposed for relations. If a relation schema is in one of these normal forms, we know that certain kinds of problems cannot arise. Considering the normal form of a given relation schema can help us to decide whether or not to decompose it further. If we decide that a relation schema must be decomposed further, we must choose a particular decomposition (i.e., a particular collection of smaller relations to replace the given relation).

With respect to the second question, two properties of decompositions are of particular interest. The lossless-join property enables us to recover any instance of the decomposed relation from corresponding instances of the smaller relations. The dependency preservation property enables us to enforce any constraint on the original relation by simply enforcing some constraints on each of the smaller relations. That is, we need not perform joins of the smaller relations to check whether a constraint on the original relation is violated. A serious drawback of decompositions is that queries over the original relation may require us to join the decomposed relations. If such queries are common, the performance penalty of decomposing the relation may not be acceptable. In this case we may choose to live with some of the problems of redundancy and not decompose the relation. It is important to be aware of the potential problems caused by such residual redundancy in the design and to take steps to avoid them (e.g., by adding some checks to application code).

Functional Dependencies

READING TIME: 1 MIN

A functional dependency (FD) is a kind of IC that generalizes the concept of a key. Let R be a relation schema and let X and Y be nonempty sets of attributes in R. We say that an instance r of R satisfies the FD X→Y if the following holds for every pair of tuples t1 and t2 in r:

If t1:X = t2:X, then t1:Y = t2:Y .

We use the notation t1:X to refer to the projection of tuple t1 onto the attributes in X, in a natural extension of our TRC notation t:a for referring to attribute a of tuple t. An FD X→Y essentially says that if two tuples agree on the values in attributes X, they must also agree on the values in attributes Y. Figure 2 illustrates the meaning of the FD AB→ C by showing an instance that satisfies this dependency. The first two tuples show that an FD is not the same as a key constraint: Although the FD is not violated, AB is clearly not a key for the relation. The third and fourth tuples illustrate that if two tuples di_er in either the A field or the B field, they can di_er in the C _eld without violating the FD. On the other hand, if we add a tuple <a1; b1; c2; d1> to the instance shown in this figure, the resulting instance would violate the FD; to see this violation, compare the first tuple in the figure with the new tuple.

A	B	C	D
a1	b1	c1	d1
a1	b1	c1	d2
a1	b2	c2	d1
a2	b1	c3	d1

Figure 2: An Instance that Satisfies AB→C
(Source: Ramakrishnan & Gherke, 1996)

Summary

In this unit we examined decomposition. We looked at their advantages and challenges, where they are to be used. We also examined redundancy as one of the things that can lead to the need for decomposition. Functional dependency was finally examined.

Self-Assessment Questions

1. What should we look out for when applying decompositions to relational tables?

Tutor Marked Assignment

Why do we need decomposition?

What is the limitation of decomposition?

References

Elmasri R. and Navathe S. B., (2011). Fundamentals of Database Systems. Addi son-Wesley.

Gupta S. B. and Mittal A., (2004). Introduction to database management systems. Pearson Education.

Ramakrishnan R. and Gherke J. (1996). Database Management System, second edition. McGraw-Hill Higher Education.

Further Reading

en.wikipedia.org/wiki/Database_system Accessed 23rd September 2018.

en.wikipedia.org/wiki/Database_management_system Accessed 23rd September 2018.

modern computer in Data center
source: Pexels byBrett Sayles

UNIT 3

NORMALIZATION

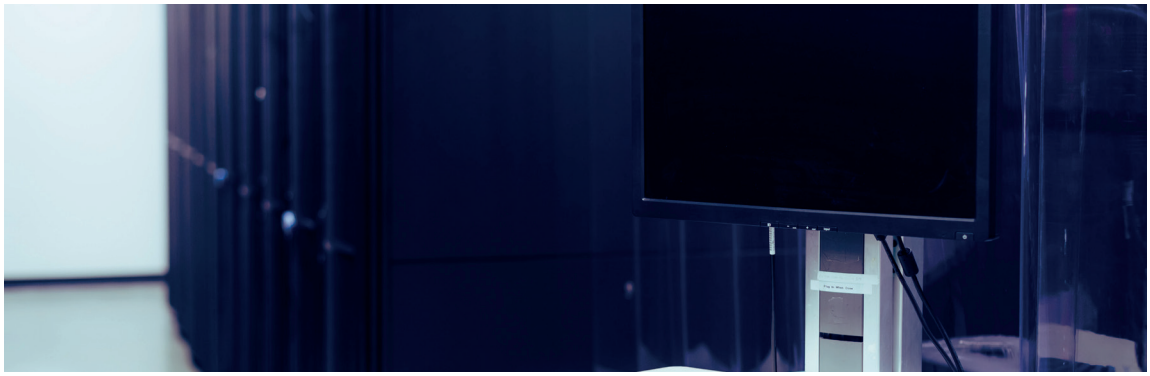
Introduction

In this unit, you will learn about the concept of normalization which is used to remove anomalies and redundancy in databases is being described in this section.

Learning Outcomes

At the end of this, you should be able to :

- Analyse the reason of normalization.
- Highlight various types of normalization.
- State the rules of each type of normalization.



modern computer in Data center
source: Pexels byBrett Sayles

Main Content

Normalization [SAQ1]

READING TIME: 1 MIN

You should learn that normalization is a process for deciding which attributes should be grouped together in a relation. It is a tool to validate and improve a design, so that it satisfies certain constraints that avoid redundancy of data. Furthermore logical, Normalization is defined as the process of decomposing relations with anomalies to produce smaller, well-organized relations. Normalization is based on the analysis of functional dependencies. A functional dependency is a constraint between two attributes or two sets of attributes. The purpose of the database design is to arrange the various data items into an organized structure so that it generates set of relationships and stores the information without any repetition. A bad database design may result into redundant and spurious data and information. In this unit, normal forms are described.

I should let you know thus that, in normalization process, a relation with redundancy can be refined by decomposing it or replacing it with smaller relations that contain the same information, but without redundancy.

Informal Design Guidelines for Relation Schemas [SAQ2]

READING TIME: 2 MINS

Bear in mind that there are four informal measures of quality for relation schema design. These measures are not always independent of one another. These Four informal measures are:

(i) Meaning (semantics) of the relation attributes.

(ii) Reducing the redundant (repetitive) values in tuples.

(iii) Reducing the null values in tuples.

(iv) Not allowing the possibility of generating spurious tuples.

(i) Meaning of the relation attributes: You should be notified that when the attributes are grouped to form a relation schema, it is assumed that attributes belonging to one relation have certain real-word meaning and a proper interpretation associated with them. This meaning (Semantics) specifies how the attribute values in a tuple relate to one another. The conceptual design should have a clear meaning, if it is done carefully, followed by a systematic mapping into relations and most of the semantics will have been accounted for. Thus the easier it is to explain the meaning of the relation, the better the relation schema design will be.

(ii) Redundant information in tuples and update anomalies: I should inform you as well that one major goal of schema design is to minimize the storage space needed by the base relations. A significant effect on storage space occurred, when we group attributes into relation schemas. The second major problem, when we use relations as base relations is the problem of update anomalies. There are mainly three types of update anomalies in a relation i.e., Insertion Anomalies, deletion anomalies and modification Anomalies.

(iii) Null values in tuples: When many attributes are grouped together into a "fat" relation and many of the attributes do not apply to all tuples in the relation, then there exists many NULL'S in those tuples. This wastes a lot of space. It is also not possible to understand the meaning of the attributes having NULL Values. Another problem occur when specifying the join operation. One major and most important problem with Null's is how to account for them when aggregate operation (i.e., COUNT or SUM) are applied. The Null's can have multiple interpretations, like:

(a) The attribute does not apply to this rule.

(b) The attribute is unknown for this tuple.

(c) The value is known but not present i.e., cannot be recorded.

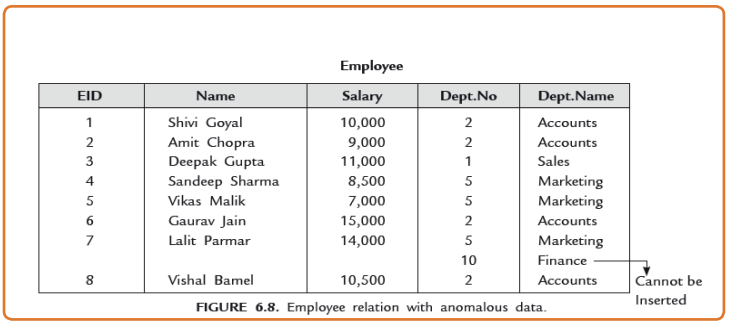
(iv) Generation of spurious tuples: The Decomposition of a relation schema R into two relations R1 and R2 is undesirable, because if we join them back using NATURAL Join, we do not get the correct original information. The join operation generates spurious tuples that represent the invalid information.

Anomalies in Relational Database

READING TIME: 1 MIN

Kindly note that there are various anomalies or pitfalls in relational database. Various dependencies in relational database cause these anomalies.

Anomalies : Anomalies refer to the undesirable results because of modification of data. Consider the relation Employee with attributes EID, Name, Salary, Dept.No, Dept.Name as shown in Figure 1. The various anomalies are as follows:



Employee

EID	Name	Salary	Dept.No	Dept.Name
1	Shivi Goyal	10,000	2	Accounts
2	Amit Chopra	9,000	2	Accounts
3	Deepak Gupta	11,000	1	Sales
4	Sandeep Sharma	8,500	5	Marketing
5	Vikas Malik	7,000	5	Marketing
6	Gaurav Jain	15,000	2	Accounts
7	Lalit Parmar	14,000	5	Marketing
			10	Finance
8	Vishal Bamel	10,500	2	Accounts

Cannot be Inserted

FIGURE 6.8. Employee relation with anomalous data.

Figure 1: Employee relation with anomalous data (Source: Gupta & Mittal, 2004)

1. Insertion Anomaly

Suppose you want to add new information in any relation but cannot enter that data because of some constraints. This is known as Insertion anomaly. In relation Employee, you cannot add new department Finance unless there is an employee in Finance department. Addition of this information violates Entity Integrity Rule 1. (Primary Key cannot be NULL). In other words, when you depend on any other information to add new information then it leads to insertion anomaly.

2. Deletion Anomaly

The deletion anomaly occurs when you try to delete any existing information from any relation and this causes deletion of any other undesirable information. In relation Employee, if you try to delete tuple containg Deepak this leads to the deletion of department “Sales” completely (there is only one employee in sales department).

3. Updation Anomaly

The updation anomaly occurs when you try to update any existing information in any relation and this causes inconsistency of data

Armstrong's Axioms

READING TIME: 2 MINS

The following three rules called inference axioms or Armstrong's Axioms can be used to find all the FDs logically implied by a set of FDs. Let X, Y, Z, and W be subsets of attributes of a relation R. The following axioms hold:

1. **Reflexivity.** If Y is a subset of X, then X → Y. This also implies that X → X always holds. Functional dependencies of this type are called trivial functional dependencies.

2. **Augmentation**. If X → Y holds and Z is a set of attributes, then ZX → ZY.
3. **Transitivity**. If X → Y holds and Y → Z holds, then X → Z holds. These rules are Sound and Complete. They are sound because they do not generate any invalid functional dependencies. They are complete because they allow us to generate F+ (closure of F) from the given set of functional dependencies F. It is very cumbersome and complex to use the Armstrong's Axioms directly for the computation of F+. So some more axioms are added to simplify the process of computation of F+. These additional axioms can be proved correct by using the Armstrong's Axioms. The additional axioms are
4. **Additivity or Union**. If X → Y and X → Z, then X → YZ holds.
5. **Projectivity or Decomposition.** If X → YZ holds, then X → Y and X → Z also holds
6. **Pseudotransitivity**. If X → Y and ZY → W holds, then XZ → W holds.
These additional axioms are also Sound and Complete.
Example: Let R = (A, B, C, D) and F be the set of functional dependencies for R given by {A→B, A→C, BC→D}. Prove A→D.

Solution: Given set of functional dependencies for a relation R is {A→B, A→C, BC→D}
By using Armstrong Axioms, named projectivity, we can show that

A→BC (as A→B, A→C)

Since BC→D, so by transitivity rule,

A→BC and BC→D means A→D. Hence proved.

Example. Let us consider a relation R2(A, B, C, D, E, F) and a set of functional dependencies

FD={AB→C, C→FA, F→E} that hold on R2.

(i) Using Armstrong's axioms show that the functional dependency AB→E also holds on R.

(ii) Does AB→F hold for FD1 = {AB→CD, C→E, DE→F}?

Solution:

(i) AB→C (given).

Apply decomposition to C→FA, get C→F. F→E (given).

Apply transitivity to AB→C, C→F, F→E, get AB→E. Hence proved.

(ii) Yes. We can prove it as follows:

AB→CD (given).

C→E (given). Apply augmentation to C→E, get, CD→DE. DE→F (given).

Apply transitivity to AB→CD, CD→DE and DE→F, get AB→F.

First Normal Form

READING TIME: 1 MIN

Normalization is a method to remove all these anomalies and bring database to consistent state and free from any kinds of anomalies.

First Normal Form: This is defined in the definition of relations (tables) itself as shown in figure 2. This rule defines that all the attributes in a relation must have atomic domains. Values in atomic domain are indivisible units.

Course	Content
Programming	Java, c++
Web	HTML, PHP, ASP

Figure 2: Unorganized Relation (Source: tutorialspoint.com)

We re-arrange the relation in figure 3, to convert it to First Normal Form

Course	Content
Programming	Java
Programming	c++
Web	HTML
Web	PHP
Web	ASP

Figure 3: Relation to 1NF (Source: tutorialspoint.com)

Each attribute must contain only single value from its pre-defined domain

Second Normal Form

READING TIME: 1 MIN

Second Normal Form:

Before we learn about second normal form, we need to understand the following:

Prime attribute: an attribute, which is part of prime-key, is prime attribute.

Non-prime attribute: an attribute, which is not a part of prime-key, is said to be a non-prime attribute.

Second normal form says, that every non-prime attribute should be fully functionally dependent on prime key attribute. That is, if X → A holds, then there should not be any proper subset Y of X, for that Y → A also holds.

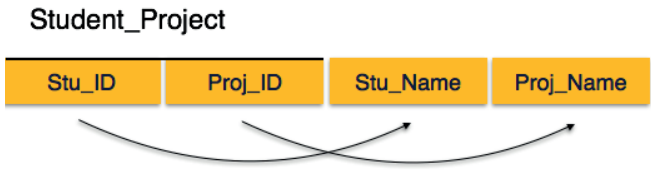


Figure 4: Relation not in 2NF (Source: tutorialspoint.com)

We see here in Student_Project relation that the prime key attributes are Stu_ID and Proj_ID. According to the rule, non-key attributes, i.e. Stu_Name and Proj_Name must be dependent upon both and not on any of the prime key attribute individually. But we find that Stu_Name can be identified by Stu_ID and Proj_Name can be identified by Proj_ID independently. This is called partial dependency, which is not allowed in Second Normal Form. See figure 4.

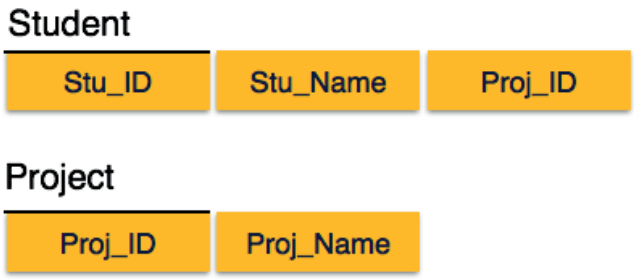


Figure 5: Relation in 2NF (Source: tutorialspoint.com)

We broke the relation in two as depicted in the above picture. So there exists no partial dependency

Third Normal Form

READING TIME: 1 MIN

Third Normal Form:

For a relation to be in Third Normal Form, it must be in Second Normal form and the following must satisfy:

No non-prime attribute is transitively dependent on prime key attribute

For any non-trivial functional dependency, X → A, then either

X is a superkey or,

A is prime attribute.

Figure 6: Relation not in 3NF(Source: tutorialspoint.com)

We find that in above depicted Student_detail relation, Stu_ID is key and only prime key attribute. We find that City can be identified by Stu_ID as well as Zip itself. Neither Zip is a superkey nor City is a prime attribute. Additionally, Stu_ID → Zip → City, so there exists transitive dependency.

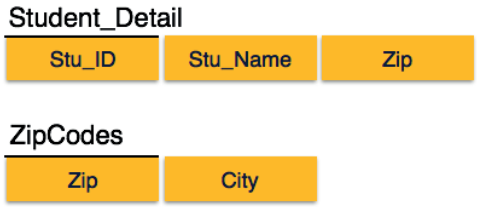


Figure 7: Relation in 3NF (Source: tutorialspoint.com)

we broke the relation as above depicted two relations to bring it into 3NF.

Boyce-Codd Normal Form

READING TIME: 1 MIN

Boyce-Codd Normal Form:

You should note that BCNF is an extension of Third Normal Form in strict way. BCNF states that

For any non-trivial functional dependency, X → A, then X must be a super-key.

In figure 7 depicted picture, Stu_ID is super-key in Student_Detail relation and Zip is super-key in ZipCodes relation. So,

Stu_ID → Stu_Name, Zip

And

Zip → City

Confirms, that both relations are in BCNF.

Summary

In this unit, normalization was examined. The different types of normalization were stated, along with their rules. First normal form, second normal form, third normal form and boyce-codd normal form was examined with an example.

Self-Assessment Questions

1. What is normalization?
2. Explain why Schema normalization often contribute to the enhancement of query processing efficiency.

Tutor Marked Assignment

Why is normalization needed?

What do you understand by lossless decomposition ?

Define and discuss 3NF and BCNF using suitable examples.

Define normalization.

References

Elmasri R. and Navathe S. B., (2011). Fundamentals of Database Systems. Addison-Wesley.

Gupta S. B. and Mittal A., (2004). Introduction to database management systems. Pearson Education.

Further Reading

Ramakrishnan R. and Gherke J. (1996). Database Management System, second edition. McGraw-Hill Higher Education.

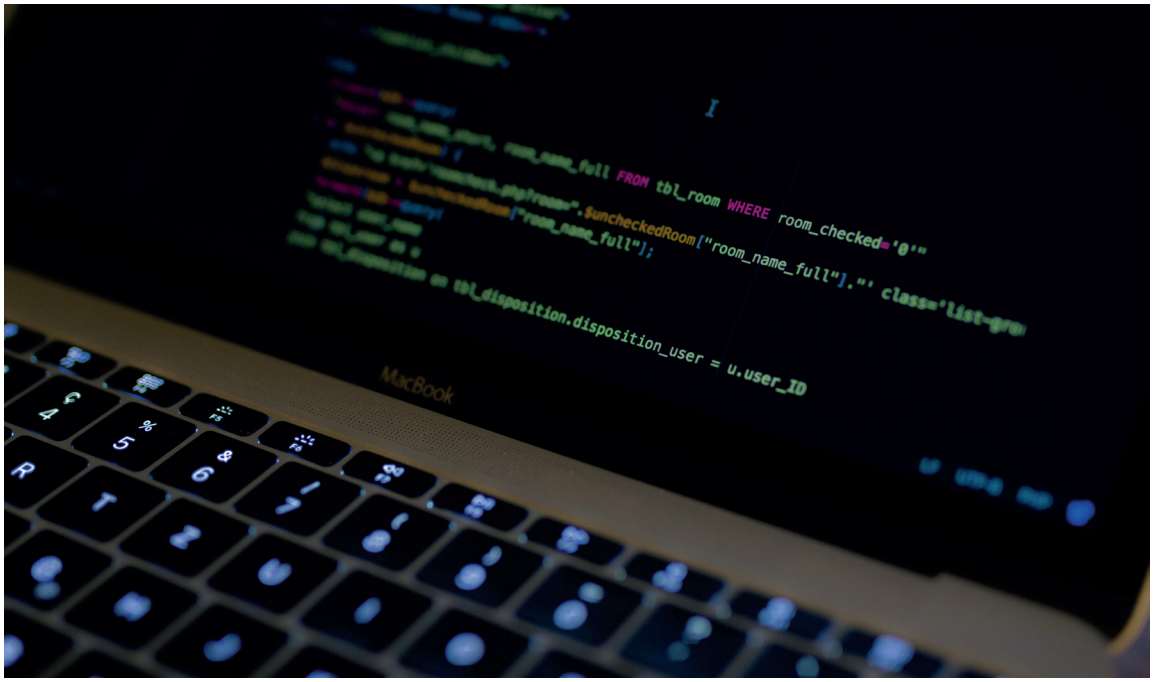
Silberschatz A., Korth H. F., and Sudarshan S. (2003). Database System Concepts. McGraw-Hill Education.

source: Unsplash

MODULE 5

QUERY LANGUAGES

UNITS



Coding SQL query
source: unsplash caspar camille rubin

UNIT 1

STRUCTURED QUERY LANGUAGE (SQL)

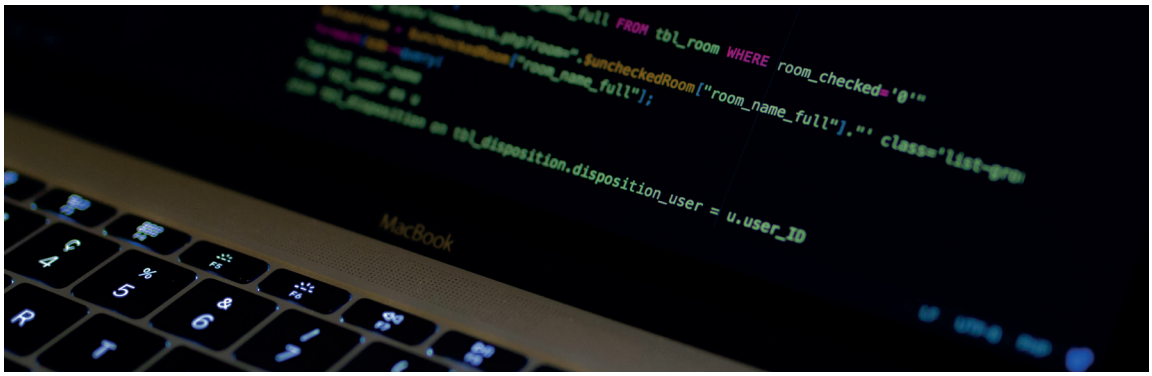
Introduction

It is imperative you know that Structured Query Language (SQL) is used for database manipulation and query generation. We will discuss more about it in this unit.

Learning Outcomes

At the end of this unit, you should be able to:

- Explain what SQL is and what it is used for.
- Create database file and database table.
- State some SQL query statements to populate a table, join tables, update a table and delete the table.



Coding SQL query
source: unsplash caspar camille rubin

Main Content

Structured Query Language (1min) [SAQ1]

Are you aware that the name SQL pronounced as “ess-cue-ell” or ‘sequel’ is the abbreviation for structured query language? The SQL consists of a set of facilities for defining, accessing and managing relational databases. All tasks related to relational data management-creating tables, querying the database, deleting, granting access to users etc., can be done using SQL. It has been accepted as an American standard by American National Standards Institute (ANSI) and is a Federal Information Processing Standard (FIPS). It is also an international standard recognized by the ISO. The first commercial DBMS that supported SQL was Oracle in 1979. SQL statements can be invoked either interactively in a terminal session or by embedding them in application programs.

SQL is a programming language for Relational Databases. It is designed over relational algebra and tuple relational calculus. SQL comes as a package with all major distributions of RDBMS. SQL comprises both data definition and data manipulation languages. Using the data definition properties of SQL, one can design and modify database schema whereas data manipulation properties allows SQL to store and retrieve data from database.

Characteristics of SQL

READING TIME: 1 MIN

The following are the important characteristics of SQL.

1. SQL is extremely flexible.

2. SQL uses a free form syntax that gives the user the ability to structure SQL statements in a way best suited.
3. It is a free formated language, i.e., there is no need to start SQL statements in a particular column or to be finished in a single line.
4. It has relatively few commands.
5. It is a non-procedural language.

Advantages of SQL [SAQ2]

READING TIME: 1 MIN

The advantages of SQL are as follows:

1. SQL is a high-level language that provides a greater degree of abstraction than procedural languages. The programmer has to specify what data is needed but need not to specify, how to retrieve it.

2. SQL is a unified language. The same language can be used to define data structures, querying data, control access to the data, insert, delete and modify occurrences of the data and so on.

3. All the programs written in SQL are portable, thus they can be moved from one database to another with very little modification. Such porting could be required when DBMS needs to be upgraded or changed.

4. The language is simple and easy to learn. It can handle complex situations very efficiently.

5. The language has sound theoretical base and there is no ambiguity about the way a query will interpret the data and produce the results. Thus, the results to be expected are well defined.

6. SQL processes sets-of-records rather than just one record-at-a time. This set-at-a time feature of the SQL makes it more powerful.

7. SQL as a language is independent of the way it is implemented internally. This is because SQL specifies what is required and not how it should be done.

8. SQL enables its users to deal with a number of database management systems where it is available.

Parts (Components) of SQL Language

READING TIME: 3 MINS

Have it at the back of your mind that the SQL language is mainly divided into four major parts. The four parts are further divided into subparts. The major parts and subparts are as follows:

1. Data-Definition Language (DDL)

The SQL DDL provide commands for defining the relations, deleting the relations and modifying the existing relation schemas.

i. View Definition Language (VDL) : The SQL DDL provide commands for defining and

dropping the views.

ii. Integrity: The SQL DDL provide commands for specifying integrity constraints that must be satisfied by the data stored in the database.

iii. Authourization: The SQL DDL provide commands for specifying access rights to the relations and views.

Create and Drop a Database/Table (Data Definition Language)

CREATE - Creates new databases, tables and views from RDBMS.

For example: Create database tutorialspoint;

Create table article;

Create view for_students;

DROP - Drop commands deletes views, tables and databases from RDBMS.

Drop object_type object_name;

Drop database tutorialspoint;

Drop table article;

Drop view for_students;

2. Data Manipulation Language (DML)

You should be informed that the SQL DML provides a query language. This query language is based on relational algebra and tuple relational calculus. This contain commands to insert tuples into the database, to delete tuples from the database and to modify/update tuples in the database. The SQL DCL provide commands that help the DBA to control the database such as commands to grant or revoke privileges to access the database and to store or remove transactions that would affect the database.

Data Manipulation in SQL (Data Manipulation Language)

I should let you know that SQL has one basic statement for retrieving information from the database: The SELECT statement. SQL also provides three other DML statements to modify the database. These statements are: update, delete and insert. The basic form of the SELECT statement is formed of three clauses SELECT, FROM, and WHERE, having the following form:

SELECT < attribute list >

FROM < table list >

WHERE < condition >

In this form,

< attribute list > is the list of attribute names whose values are to be retrieved by the query.

< table list > is the list of relation names required to process the query.

< condition > is a conditional expression that identifies the tuples to be retrieved by the query.

The following examples show the working of SELECT statement with different options.

Elect Statement

Select statement is used to retrieve information from table.

Syntax: select < column list >

from < table name >.

Example 1: To display all department ID and the Department name, the query is

SELECT DID, DName

from Dept;

DID	DName
10	Accounts
20	Sales
30	Research
40	Developing

Example 2: To select all columns use "*".

select *

from Dept;

DID	DName	Loc	Manager-ID
10	Accounts	Bangalore	702
20	Sales	Delhi	705
30	Research	Pune	707
40	Developing	Hyderabad	

(i) Column Alias : You can give name to columns of your choice by using keyword "As"

(gives column name in upper-case letter) or by using " " (gives column name as specified in query).

Example 3: select DID As Department_ID, DName

from Dept;

Department-ID	DName
10	Accounts
20	Sales
30	Research
40	Developing

(ii) Concatenation Operator (II): It is used to concatenate two or more columns.

Example 4 : select DName || Loc As department

from Dept;

DEPARTMENT
AccountsBangalore
SalesDelhi
ResearchPune
DevelopingHyderabad

(iii) Literal Character Strings: A literal is a number, character or date that can be included in columns. Character and date literals must be enclosed with single quotation marks (‘ ‘).

Example 5 : select DName || branch is situated at ’ || Loc As department

from Dept ;

DEPARTMENT
Accounts branch is situated at Bangalore
Sales branch is situated at Delhi
Research branch is situated at Pune
Developing branch is situated at Hyderabad

(v) Arithmetic Operators and NULL Values : SQL provides arithmetic operators to perform calculations. Arithmetic operators with their precedence are

Description	Operator
Multiply	*
Divide	/
Add	+
Subtract	–

A null value is unknown value and it is different from zero. Any arithmetic operation with null value gives null results.

Example 7: Suppose you want to increase salary of each employee by 500.

select EID, salary + 500 “New Salary”

from Emp;

EID	New salary
701	8500
702	9500
703	7500
704	9500
705	7000
706	
707	10000
708	8500

(vi) Where Clause : WHERE clause is used to select particular rows from table.

Syntax : select <column list>

from <table name>

where <condition>.

Example 8 : List the name of employees having salary ` 9000.

select name

from emp

where salary = 9000;

Name
Naresh
Aditya

(a) Comparison or relational operators: The relational operators provided by SQL are as follows.

Operator	Description
=	Equal to
>	Greater than
<	Less than
>=	Greater than equal to
=<	Less than equal to
<>	Not equal to

Example 9: List the name of employees having salary not equal to ` 9000.

select name

from emp

where salary <> 9000;

Name
Deepak
Sumesh
Lalit
Amit
Vishal
Sumit

(b) Special operators: Special operators provided by SQL are as follows:

Operator	Description
IN	Checking the value within a set
BETWEEN	Checking the value within a range
LIKE	Matching the pattern of characters
IS NULL	Checking the null value

Example 10 : List name and EID of employees having salary ` 8000 or ` 9500.

```
select EID, name
from Emp
where salary IN (8000, 9500);
```

EID	Name
701	Deepak
707	Vishal
708	Sumit

Basic Datatypes

READING TIME: 1MIN

The SQL supports a variety of data types as shown in Table 1.

Datatype	Description	Size
Number(p, s)	It is used to store numeric data types. p stands for precision (total number of decimal digits) and s stands for scale (total number of digits after decimal point).	Range of p is from 1 to 38. And s is from -84 to 127.
Date	It is used to store date and time values.	Range of date is from Jan 1, 47 B.C. to Dec. 31, 9999 A.D.
Char(size)	It is used to store fixed size character data.	Range of char is 1 (By default) to 2000 bytes.
Varchar2(size)	It is used to store variable size character data.	Range of varchar2 is 1 (By default) to 4000 bytes.
Long	It is used to store variable size character data.	Range of long is upto 2 GB.
Clob	It is used to store variable size character data.	Range of clob is upto 4 GB
Raw(size)	It is used to store fixed binary data.	Maximum size is upto 2000 bytes.
Long raw	It is used to store variable binary data.	Maximum size is upto 2 GB.

Table 1: Basic data types (Gupta & Mittal, 2004)

Joining of Tables

READING TIME: 2 MINS

If we need information from more than one table then we use joins. To join tables the condition need to be specified.

1. Cartesian Product : In Cartesian product there is no join condition. It returns all possible combinations of rows.

Example 34 : select EID, LOC

from Emp, Dept;

EID	LOC
701	Bangalore
701	Delhi
–	–
–	–
708	Hyderabad

8 × 4 = 32 row

2. Equijoin : When two or more tables are joined by equality of values in one or more columns then it is called Equijoin.

*More than two tables can be joined by using logical operators.

Example 35 : Display EID and DName of all employees by joining over DID.

select Emp.EID, Dept.DName

from Emp, Dept

where Emp.DID = Dept.DID

EID	DName
701	Research
702	Accounts
703	Sales
704	Research
705	Sales
706	Accounts
707	Research
708	Accounts

3. Table Aliases : Alias names can be given to the tables. It is specified in FROM clause. They are helpful to simplify queries.

Example 36 : Repeat Ex. 35 with table alias

select e.EID, d.DName

from Emp e , Dept d

where e.DID = d.DID; // e and d are alias names.

4. Non-Equijoin : When tables are joined by any other operator except the equality operator in condition, then it is known as Non-Equijoin.

Example 37 : Display EID and DName of employees having MID 705 or 707

select e.EID, d.DName

from Emp e, Dept d

where e.MID IN (d.MID = 705, d.MID = 707);

EID	DName
701	Research
702	Accounts
703	Sales
704	Research
705	Sales

5. Outer Join : The outer join operator is '(+)'. During simple joining some rows are missing due to null value. To display these rows use outer join operator towards deficient side.

6. Self Join : A table can be joined to itself by using self joins.

Example 39 : Display the name of employees and name of their managers.

select e.Name "Employee", m.Name "Manager"

from Emp e, Emp m

where e.MID = m.MID;

Employee	Manager
Deepak	Vishal
Naresh	Vishal
Sumesh	Lalit
Aditya	Vishal
Lalit	Vishal
Amit	Naresh
Sumit	Naresh

5. Outer Join : The outer join operator is '(+)'. During simple joining some rows are missing due to null value. To display these rows use outer join operator towards deficient side.

6. Self Join : A table can be joined to itself by using self joins.

Example 39 : Display the name of employees and name of their managers.

select e.Name "Employee", m.Name "Manager"

from Emp e, Emp m

where e.MID = m.MID;

Employee	Manager
Deepak	Vishal
Naresh	Vishal
Sumesh	Lalit
Aditya	Vishal
Lalit	Vishal
Amit	Naresh
Sumit	Naresh

7. Cross Join : It is same as cartesian product

Example 40 : Repeat ex. 35 by cross join

select e.EID, d.DName

from Emp e, Dept d

CROSS JOIN DID;

8. Natural Join : It is used to join two tables having same column names and data types. It select rows from two tables having equal values.

Syntax : select <column names>

from <table_names>

NATURAL JOIN (table name);

9. Left Outer Join : To display all the rows of table left of the join condition, use LEFT OUTER JOIN. This keyword is used instead of outer join operator '(+)'.

10. Right Outer Join : To display all the rows of table right of the join condition, use RIGHT OUTER JOIN. This keyword is used instead of outer join operator '(+)'.

11. Full Outer Join : To display all the rows of both of the tables, use keyword FULL OUTER JOIN.

Update Statement (1min)

Update statement is used to modify the values of existing rows.

Syntax : UPDATE <table name>

SET <(column 1 = value 1), (column 2 = value 2),....., (column n = value n)>

WHERE <condition>;

*All rows in the table satisfies the condition are updated.

Delete Statement (1min)

Delete Statement

Delete statement is used to remove existing rows from table.

Syntax : DELETE FROM <table name>

WHERE <condition>;

Summary

In this unit, we examined SQL, and its different components. We also looked at a few query instructions and their variation. They include examined statements include the select, update, modify and delete statements. The join statement was also examined in details.

Self-Assessment Questions

1. What is SQL?
2. State the uses of SQL?

Tutor Marked Assignment

Write an SQL statement to create a database, and a table.

Write an SQL statement to insert, delete and update the table above.

References

Elmasri R. and Navathe S. B., (2011). Fundamentals of Database Systems. Addison-Wesley.

Gupta S. B. and Mittal A., (2004). Introduction to database management systems. Education.

Further Reading

Ramakrishnan R. and Gherke J. (1996). Database Management System,

seond edition. McGraw-Hill Higher Education.

Silberschatz A., Korth H. F., and Sudarshan S. (2003). Database System Concepts. McGraw-Hill Education.

Crystal cube
Source: unsplash by michael dziedzic

UNIT 2

RELATIONAL ALGEBRA AND CALCULUS

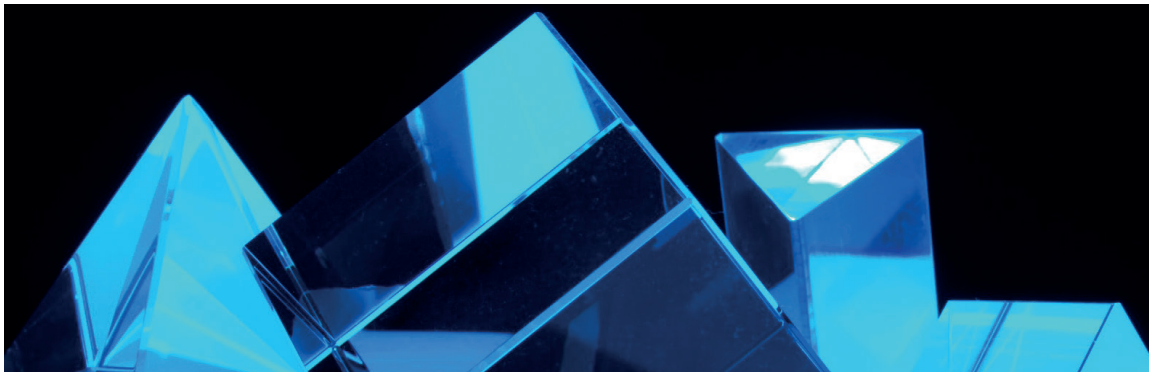
Introduction

In this unit, you will learn about relational Algebra, relational operations and relational calculus. The comparison between Domain Relational calculus and tuple Relational Calculus is being examined in this unit. Relational Calculus and Relational Algebra are also compared.

Learning Outcomes

When you have studies this unit, you should be able to:

- Explain what SQL is and what it is used for.
- Create database file and database table.
- State some SQL query statements to populate a table, join tables, update a table and delete the table.



Crystal cube
Source: unsplash by michael dziedzic

Main Content

Relational Algebra [SAQ1, 2]

READING TIME: 1 MIN

I should start by letting you know that relational algebra is one of the two formal query languages associated with the relational model. It is a procedural language. It specifies the operations to be performed on existing relations to derive result relations. A sequence of relational algebraic operations forms a relational algebraic expression. The result of the relational algebraic expression is also a relation. The relational algebra is very important due to many reasons. Firstly, it provides a basic foundation for relational model operations. Secondly, it is used as a basis for implementing and optimizing queries in RDBMS's. Thirdly, some of the basic concepts of relational algebra are incorporated into the SQL language. Relational calculus is a formal query language where the queries are expressed as variables and formulas on these variables. The formula used in relational calculus, describes the properties of the result relation to be obtained.

calculator on a paper
Source: Pexels Pixabay

Try to understand that relational algebra is a procedural query language. It uses a collection of operators to compose the queries. Every operator in the algebra accepts either one or two relation instances as arguments and output a resultant relation instance. Thus operators can be composed easily to construct complex queries. Each relational algebra query describes a step-by-step procedure for computing the desired answer, based on the order in which the operators are applied in the query.

The relational algebra uses various logical connectives [∧(and), ∨(or), ¬(not)] and comparison operators (<, <=, =, ≠, >=, >) to construct composite and more complex queries. We discuss the different operators (Basic set oriented operators—union, intersection, difference, and cartesian product and relational operators—selection, projection, division and join) in detail, with examples, in subsequent sections. All the examples are based on the EMPLOYEE-STUDENT database shown in Figure 1. This database contain two Tables EMPLOYEE and STUDENT and the relationship is that an employee can also a student and vice versa.

Employee

EID	Name	Salary
1E	John	10,000
2E	Ramesh	5,000
3E	Smith	8,000
4E	Jack	6,000
5E	Nile	15,000

Student

SID	Name	Fees
1S	Smith	1,000
2S	Vijay	950
3S	Gaurav	2,000
4S	Nile	1,500
5S	John	950

Figure 1: Employee and student relations (Source: Gupta & Mittal, 2004).

Operations in Relational Algebra

READING TIME: 1 MIN

It is important I inform you that the relational algebraic operations can be divided into basic set-oriented operations (union, intersection, set difference, and Cartesian product) and relational-oriented operations (selection, projection, division and joins).

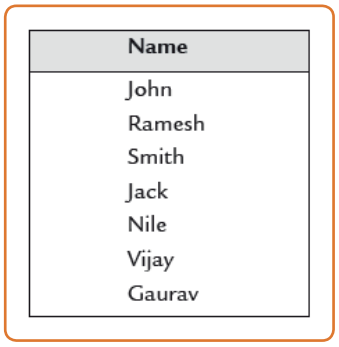
The union operation : The union operation is a binary operation that is used to find union of relations. Here relations are considered as sets. So, duplicate values are eliminated. It is denoted by (∪).

Conditions for union operation : There are two necessary conditions for union operation.

(i) Both the relations have same number of attributes.

(ii) Data types of their corresponding attributes must be same.

Two relations are said to be union compatible if they follow the above two conditions.



Name
John
Ramesh
Smith
Jack
Nile
Vijay
Gaurav

Figure 2: Result of union operation (Source: Gupta & Mittal, 2004).

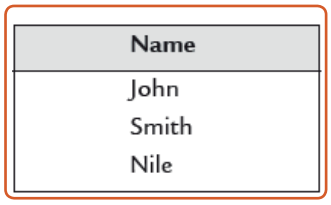
Set Operations

READING TIME: 1 MIN

1. Set intersection operation : Set intersection is used to find common tuples between two relations. It is denoted by (∩). If you want to find all the employees from Relation Employee those are also students. Rules of set union operations are also applicable here. Then the query, is

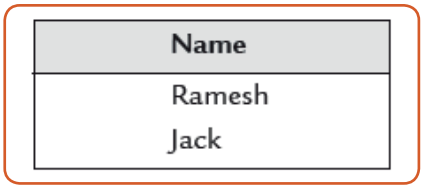
π_{Name} (Employee) ∩ π_{Name} (Student)

The result is shown in Figure 3.



Name
John
Smith
Nile

Figure 3: Result of set intersection operation (Source: Gupta & Mittal, 2004).



Name
Ramesh
Jack

Figure 4: Result of set-difference operation (Source: Gupta & Mittal, 2004).

Cartesian Product

READING TIME: 1 MIN

Cartesian product operation: You should have it at the back of your mind that Cartesian product is a binary operation which is used to combine information of any two relations. Suppose a relation R1 is having m tuples and other relation R2 is having n tuples then R1 × R2 has m × n tuples. It is denoted by (X). Consider the Figure 5. Cartesian product of relation Employee and Job is shown in Figure 6.

Query is → Employee × Job

Employee

EID	Name	JID
1E	Manoj	1J
2E	Deepak	2J
3E	Vinay	1J

Job

JID	Job
1J	Tester
2J	Manager

Figure 5: Employee and Job relation (Source: Gupta & Mittal, 2004).

EID	Name	Employee JID	Job JID	Job
1E	Manoj	1J	1J	Tester
1E	Manoj	1J	2J	Manager
2E	Deepak	2J	1J	Tester
2E	Deepak	2J	2J	Manager
3E	Vinay	1J	1J	Tester
3E	Vinay	1J	2J	Manager

Figure 6: Result of Cartesian product operation (Source: Gupta & Mittal, 2004).

Relational Operations

READING TIME: 2 MINS

(a) Selection or Restriction operation: The selection operation is a unary operation. This is used to find horizontal subset of relation or tuples of relation. It is denoted by sigma (σ).

Ex. *If you want all the employees having salary more than 9,000 from relation Employee. The query, is*

$\sigma_{salary > 9,000}$ *(Employee).* The result is shown in Figure 7.

EID	Name	Salary
1E	John	10,000
5E	Nile	15,000

Figure 7: Result of selection operation (Source: Gupta & Mittal, 2004).

We can also combine these operations. If you want name of all employees having salary less than 7,000. Then the query is $\sigma_{salary < 7,000}$ [π_{Name} (Employee)]

Step 1. First we apply projection operation on relation employee to get name of all employees.

π_{Name} (Employee)

Step 2. Then we apply selection operation to choose employees having salary less than 7,000.

[$\sigma_{salary < 7,000}$ (π_{Name} (Employee))] → Relational algebra expression

Name
Ramesh
Jack

Figure 8: Result of $\sigma_{salary < 7,000}$ [π_{name} (Employee)] (Source: Gupta & Mittal, 2004).

(b) Projection operation : The projection operation is a unary operation which applies only on a single relation at a time. Project operation is used to select vertical subset of relation (i.e., columns of table). It is denoted by pi (π). If you want all the names of employees and their salary from relation employee. Then query, is $\pi_{name, salary}$ (Employee)

The result is shown in Figure 9.

Name	Salary
John	10,000
Ramesh	5,000
Smith	8,000
Jack	6,000
Nile	15,000

Figure 9: Result of projection operation (Source: Gupta & Mittal, 2004).

(c) Division operation : Division operation is useful in special kind of queries that include the phrase "for all". It is denoted by (÷). It is like the inverse of Cartesian product.

For example:

A

X	Y
X1	Y1
X1	Y3
X1	Y2
X4	Y
X5	Y5
X2	Y3
X3	Y4
X4	Y1

B1

Y
Y1
Y3

B2

Y
Y1
Y5

B3

Y
Y5
Y2
Y4

A ÷ B1 gives

X
X1
X4
X2

A ÷ B2 gives

X
X1
X4
X5

A ÷ B3 gives

X
X5
X1
X3

Figure 10: Result of division operation (Source: Gupta & Mittal, 2004).

Let R be the relation with schema r and S be the relation with schema s. Let S ⊆

R. Then tuple t is in r ÷ s if and only if

(i) t is in πR–S (r)

(ii) If tuple is present in the Cartesian product of S and R.

(d) Natural-join operation: Natural join is used to join two relations having any number of attributes. It is denoted by symbol (◊). It also optimizes the query as Cartesian product gives unnecessary results and set-union and set-intersection operations are applicable only on those relations that have equal number of attributes with same data-type. Consider the relations in Figure 11.

Employee

EID	Name	Salary	Dept-ID
1	Amit	5,000	10
2	Sachin	8,000	20
3	Vinay	2,000	20
4	Vivek	6,000	10

Department

Dept_ID	Dept_Name
10	Sales
20	Purchase

Figure 11: Employee and department relation (Source: Gupta & Mittal, 2004).

Name	Dept-Name
Amit	Sales
Sachin	Purchase
Vinay	Purchase
Vivek	Sale

Figure 12: Result of natural join operation (Source: Gupta & Mittal, 2004).

(e) Outer join: Outer Join is an extension of natural join operations. It deals with the missing information caused by natural join operation. Suppose you need all information of all employees and all students in a single relation.

Natural join (◊): The natural join (Employee ◊ Student) gives the result as shown in Figure 13.

EID	SID	Name	Salary	Fees
1E	5S	John	10,000	1,000
3E	1S	Smith	8,000	1,000
5E	4S	Nile	15,000	1,500

Figure 13: Result of natural join (Source: Gupta & Mittal, 2004).

Relational Calculus (1mins)

Are you aware that an alternative to relational algebra is relational calculus? It is a query system where queries are expressed as variables and formulas on these variables. It is a non-procedural or declarative by nature. In this, the formulas describe the properties of the required result relation without describing how to compute it i.e., query represents only results and hides the procedure to find the result. Relational calculus is based on predicate calculus, which is used to symbolize logical arguments in mathematics. Relational calculus has two variants. The first one is called Tuple Relational Calculus in which variables take on tuples as values. The second one is called Domain Relational Calculus in which variables range over the underlying domains. Relational Calculus has strongly influenced the design of many query languages like SQL and QBE.

Comparison of Domain Relational Calculus and Tuple Relational Calculus

READING TIME: 1 MIN

S.No.	Domain Relational Calculus	Tuple Relational Calculus
1.	In domain relational calculus, the variables range over field values.	In tuple relational calculus, the variables take on tuples as values.
2.	It strongly influences SQL.	It strongly influences the QBE.
3.	In domain relational calculus, additional rules are needed to deal with the safety of expressions.	In tuple relational calculus, it is possible to restrict any existentially qualified variable to range over a specific relation for safe expressions.
4.	It is a non-procedural language.	It is also a non-procedural language.

Comparison of Relational Calculus and Relational Algebra (1min)

S.No.	Relational Calculus	Relational Algebra
1.	It is non-procedural or declarative language.	It is a procedural language.
2.	It has a big influence on query languages like SQL and QBE.	It has big influence on almost all query languages based on relations.
3.	It describes the set of answers without being implicit about how they should be computed.	The relational algebra query describes a step by step procedure for computing the desired result.
4.	All relational algebra queries can be expressed in relational calculus.	It is restricted to safe queries on the calculus.
5.	The relational calculus expression can be written in any order and the order does not affect the strategy for evaluating the query.	A certain order among the operations is implicitly specified and the order influences the strategy for evaluating the query.
6.	The relational calculus languages are terse.	The relational algebra is more user friendly.

Summary

In this unit, I educated you about Relational operators. Relational calculus was also examined. Different examples of relational operators in relational algebra were considered and a comparism was made between relational algebra and relational calculus.

Self-Assessment Questions

1. Define Relational Algebra.
2. What is Relational Calculus?

Tutor Marked Assignment

Compare amd contrast between Relational Algebra and Relational Calculus.

Describe 4 (four) different types of Relational Operations, giving relevant examples.

References

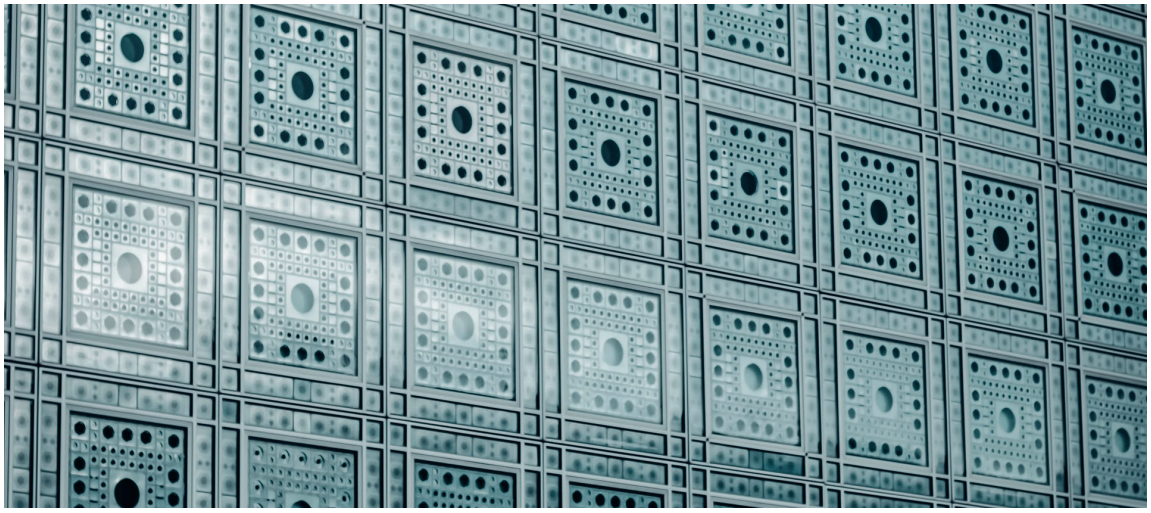
Gupta S. B. and Mittal A., (2004). Introduction to database management systems. Pearson Education.

Ramakrishnan R. and Gherke J. (1996). Database Management System, second edition. McGraw-Hill Higher Education.

Further Reading

Silberschatz A., Korth H. F., and Sudarshan S. (2003). Database System Concepts. McGrawHill Education.

Begg C. E. and Connolly T. M. (2002). Database Systems: A Practical Approach to Design, Implementation, and Management, Addison-Wesley.



glass wall
Source: Unsplash by Mat Reding

UNIT 3

QUERY PROCESSING

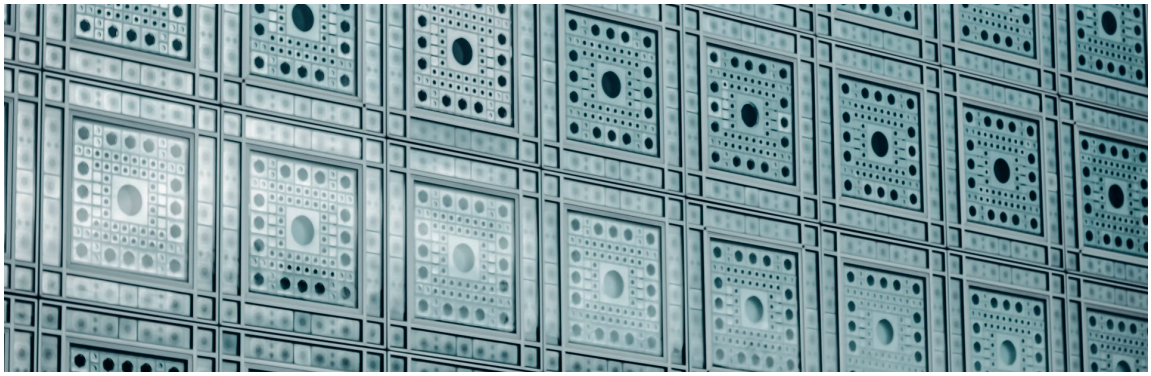
Introduction

In the unit, you will learn about Query processing strategies and various steps in query operations.

Learning Outcomes

When you have studies this unit, you should be able to:

- State the different stages of query processing.
- Describe the strategy for query processing.
- List the steps in query processing.



glass wall
Source: Unsplash by Mat Reding

Main Content

Query Processing

READING TIME: 1 MIN

You need to kindly note that query processing requires that the DBMS identify and execute a strategy for retrieving the results of the query. The query determines what data is to be found, but does not define the method by which the data manager searches the database.

Query Processing: Query Processing is a procedure of converting a query written in high level language (Example, SQL, QBE) into a correct and efficient execution plan expressed in low level language, which is used for data manipulation. The procedure for Query processing is as follows:
Query Processor: Query processor is responsible for generating execution plan.

Execution Plan: Are you aware that query processing is a stepwise process? Before retrieving or updating data in database, a query goes through a series of query compilation steps. These steps are known as execution plan. The success of a query language also depends upon its query processor i.e., how much efficient execution plan it can create? The better execution plan leads to low time and cost. In query processing, the first phase is transformation in which parser first checks the syntax of query and also checks the relations and attributes used in the query that are defined in the database. After checking the syntax and verifying the relations, query is transformed into equivalent expression that are more efficient to execute. Transformation, depends upon various factors like existence of certain database structures, presence of different indexes, file is sorted or not, cost of transformation,

physical characteristics of data etc. After transformation of query, transformed query is evaluated by using number of strategies known as access plans. While generating access plans, factors like physical properties of data and storage are taken into account and the optimal access plan is executed. The next step is to validate the user privileges and ensure that the query does not disobey the relevant integrity constraints. Finally, execution plan is executed to generate the result.

General Strategy for Query Processing [SAQ2]

READING TIME: 1 MIN

You should be informed that the general strategy for query processing is as follows:

(i) Representation of query: Query written by user cannot be processed directly by system. Query processor first checks the syntax and existence of relations and their attributes in database. After validations, query processor transform it into equivalent and more efficient expression for example query will be converted into a standard internal format that parser can manipulate. Parser also adds some additional predicates to the query to enforce security. Internal form may be relational algebra, relational calculus, any low-level language, operator graphs etc.

(ii) Operator graphs : Operator graphs are used to represent query. It gives the sequence of operations that can be performed. It is easy to understand the query represented by operator graphs. It is useful to determine redundancy in query expressions, result of transformation, simplify the view etc.

(iii) Response time and Data characteristics consideration : Data characteristics like length of records, expected sizes of both intermediate and final results, size of relations etc., are also considered for optimizing the query. In addition to this overall response time is also determined.

Steps in Query Processing [SAQ1]

READING TIME: 3 MINS

Beware of Various steps in query processing shown in Figure 1. Suppose that user inputs a query in general query language say QBE, then it is first converted into high-level query language say SQL etc. Other steps in query processing are discussed below in detail:

(i) Syntax Analysis: Query in high-level language is parsed into tokens and tokens are analyzed for any syntax error. Order of tokens are also maintained to make sure that all the rules of language grammars are followed. In case of any error, query is rejected and an error code with explanation for rejection is returned to the user. (Only syntax is checked in this step).

(ii) Query Decomposition: In this step, query is decomposed into query blocks which are the low-level operations. It starts with the high-level query that is transformed into lowlevel operations and checks whether that query is syntactically and semantically correct. For example, a SQL query is decomposed into blocks like Select block, From block, Where block etc. Various stages in query decomposition are shown in Figure 2.

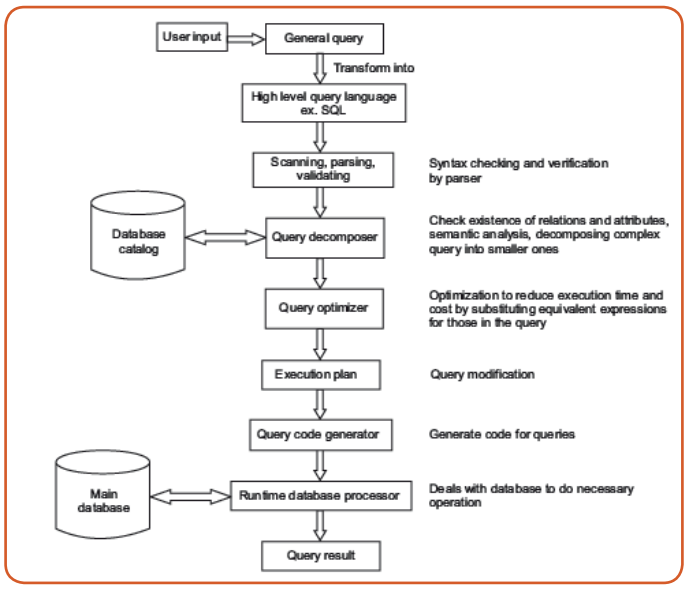


Figure 1: Steps in query processing (Gupta & Mittal, 2004).

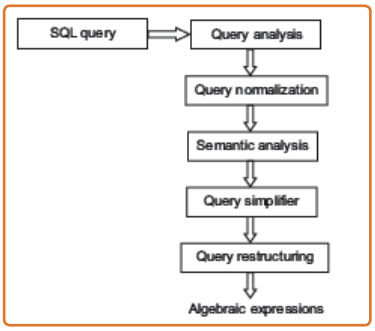


Figure 2: Steps in query decomposition (Gupta & Mittal, 2004).

The steps in query decomposition are listed as follows:

(a) Query Analysis: I should point out that in the query analysis stage, programming language compiler checks that the query is lexically and syntactically correct. A syntactically correct query is analyzed using system catalogues to verify the existence of relations and attributes used in query. After analysis a correct query is converted into some internal representation, which is more efficient for processing. The type specification of the query qualifier and result is also checked at this stage. The internal representation may be, query tree or query graph. See figure 3.

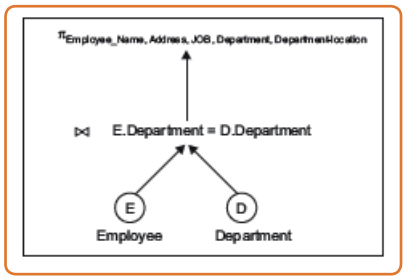


Figure 3: Query tree notation (Source: Gupta & Mittal, 2004).

Query tree notation: Be notified that a typical internal representation of query is query tree. It is also known as relational algebra tree. A query tree is constructed using tree data structure that corresponds to the relational algebra expression. Main components of query tree are:

Root of tree–represents result of query.

Leaf nodes–represent input relations of the query.

Internal nodes–represent intermediate relation that is the output of applying an operation in the algebra.The sequence of operations is directed from leaves to the root node.

For example,

Query graph notation: You should note that graph data structure is also used for internal representation of query. In graphs:

Relation nodes–represent relations by single circle.

Constant nodes–represent constant values by double circle.

Edges–represent relation and join conditions.

Square brackets–represent attributes retrieved from each relation.

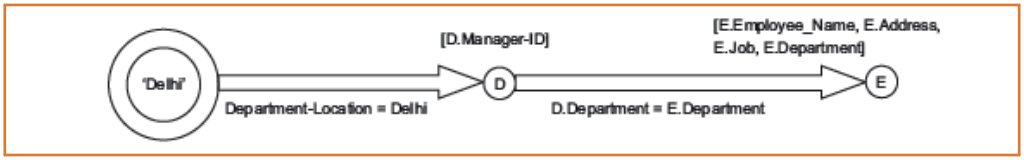


Figure 4: Query graph notation (Source: Gupta & Mittal, 2004).

(b) Query normalization : After query analysis, it is normalized to remove any redundancy. In this phase query is converted into normalized form that can be easily manipulated. A set of equivalency

(c) Semantic analyzer: You should be aware that the semantic analyzer performs the following tasks :

- It helps in reducing the number of predicates.
- It rejects contradictory normalized forms.
- In case of missing join specification, components of query do not contribute to generation of results. It identifies these queries and rejects them.
- It makes sure that each object in query is referenced correctly according to its data type.

(d) Query simplifier: The major tasks of query simplifier are as follows :

- It eliminates common sub-expressions.
- It eliminates redundant qualification.
- It introduces integrity constraints, view definitions into the query graph representation.
- It eliminates query that voids any integrity constraint without accessing the database.
- It transforms sub-graphs into semantically equivalent and more efficient form.
- It deals with user access rights.

Idempotent rules of Boolean Algebra are applied to get final form of simplification.

(e) Query Restructuring : At the final stage of query decomposition, transformation rules are applied to restructure the query to give a more efficient implementation.

(iii) Query Optimization : The aim of the query optimization step is to choose the best possible query execution plan with minimum resources required to execute that plan.

(iv) Execution Plan : Execution plan is the basic algorithm used for each operation in the query. Execution plans are classified into following Four types:

(a) Left-deep tree query execution plane.
(b) Right-deep tree query execution plan.
(c) Linear tree execution plan.
(d) Bushy execution plan.

Summary

This unit examined the various stages of query processing. It presents the general strategy for query processing and explains the steps involved in query processing.

Self-Assessment Questions

1. State and explain the steps involved in query processing.
2. Explain the stages of query processing.

Tutor Marked Assignment

With the aid of a diagram, explain the components of a Query tree.

Describe the Query Graph notation.

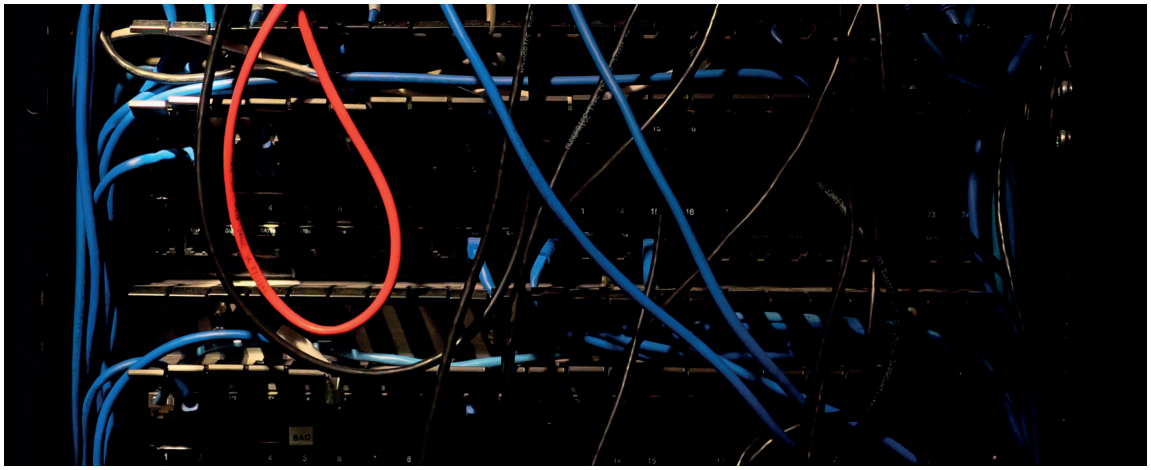
References

Ramakrishnan R. and Gherke J. (1996). Database Management System, second edition. McGraw-Hill Higher Education.

Silberschatz A., Korth H. F., and Sudarshan S. (2003). Database System Concepts. McGraw-Hill Education.

Further Reading

en.wikipedia.org/wiki/Database_management_system Accessed 23rd September 2018.



Electronic device
Source: unsplash byneonbrand

UNIT 4

QUERY OPTIMIZATION

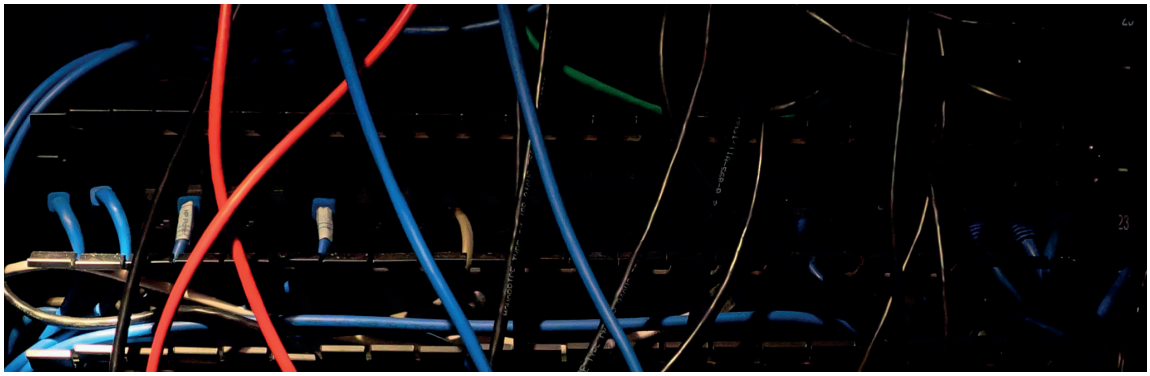
Introduction

In this unit, query optimization issues will be examined and the types of query optimization techniques that exist are described. Also, you should be able to identify heuristic query optimization and cost based query optimization.

Learning Outcomes

When you have studies this unit, you should be able to:

- Define Query Optimization.
- List the types of Query Optimization Techniques.
- Distinguish between heuristic query optimization and cost based query optimization.



Electronic device
Source: unsplash byneonbrand

Main Content

Query Optimization [SAQ1]

READING TIME: 1 MIN

Bear in mind that query optimization is necessary to determine the optimal alternative to process a query. There are two main techniques for query optimization. The first approach is to use a rule based or heuristic method for ordering the operations in a query execution strategy. The second approach estimates the cost of different execution strategies and chooses the best solution. In general, most commercial database systems use a combination of both techniques.

Query performance of a database systems is dependent not only on the database structure, but also on the way in which the query is optimized. Query optimization means converting a query into an equivalent form which is more efficient to execute. It is necessary for high-level relation queries and it provides an opportunity to DBMS to systematically evaluate alterative query execution strategies and to choose an optimal strategy. A typical query optimization process is shown in Figure 1.

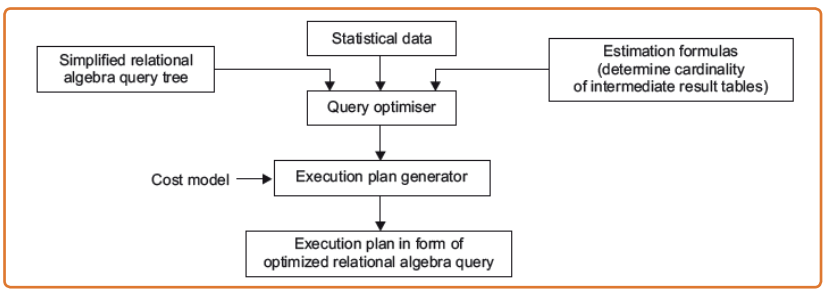


Figure 1: Query optimization process.

The main issues that need to be considered in query optimization are:

1. Reduction of data transfer with database.
2. Use of available indexes for fast searching.
3. Reduction of number of times the database is manipulated.
4. The order in which joins should be performed.
5. How to store intermediate results?

Following are the three relations we used in each example:

Employee (Emp-ID, Emp-Name, Age, Salary, Dept-ID)
Department (Dept-ID, Proj-ID, Dept-Name)
Project (Proj-ID, Name, Location, Duration)

There are two main techniques used to implement query optimization. These are heuristic query optimization and cost based query optimization.

Heuristic Query Optimization

You should be mindful that heuristic query optimization technique is used to modify the internal representation of a query by using heuristic rules and transformation rules. Heuristic rules are used in the form of a query tree or query graph structure. Optimiser starts with initial query tree and transform it into an equivalent and efficient query tree using transformation rules.

Heuristic Optimization Algorithm: It is important I let you know that DBMS use heuristic optimization algorithms to improve the efficiency of query by converting initial query tree into an equivalent and optimized query tree. Optimizers utilize transformation rules to optimize the structure of query tree. Following are the steps of heuristic optimization algorithm.

Step 1. Perform Selection operation as early as possible: By using selection operation at early stages, you can reduce the unwanted number of record or data, to transfer from database to primary memory. Optimizer use transformation rule 1 to divide selection operations with conjunctive conditions into a cascade of selection operations.

Step 2. Perform commutatively of selection operation with other operations as early as possible: Optimizer use transformation rule 2, 4, 6, and 9 to move selection operation as far down the tree as possible and keep selection predicates on the same relation together. By keeping selection operation down at tree reduces the unwanted data transfer and by keeping selection predicates together on same relations reduces the number of times of database manipulation to retrieve records from same database table.

Step 3. Combine the Cartesian Product with subsequent selection operation whose predicates represents a join condition into a JOIN operation : Optimizer uses transformation rule 13 to convert a selection and cartesian product sequence into join. It reduces data transfer. It is always better to transfer only required data from database instead of transferring whole data and then refine it. (Cartesian product combines all data of all the tables mention in query while join operation retrieves only those records from database that satisfy the join condition).

Step 4. Use Commutativ y and Associativity of Binary operations : Optimizer use transformation rules 5, 11, and 12 to execute the most restrictive selection operations first. It rearranges the leaf nodes of query tree. By using the most restrictive selection operations, the number of records fetched from database reduces and also subsequent operations can be performed on less number of records.

Step 5. Perform projection operations as early as possible : After performing selection operations, optimizer use transformation rules 3, 4, 7 and 10 to reduce the number of columns of a relation by moving projection operations as far down the tree as possible and keeping projection predicates on the same relation together.

Step 6. Compute common expressions only once: It is used to identify sub-trees that represent groups of operations that can be executed by a single algorithm.

Cost Based Query Optimization [SAQ2]

READING TIME: 1 MIN

I should include that in cost based query optimization, optimizer estimates the cost of running of all alternatives of a query and choose the optimum alternative. The alternative which uses the minimum resources is having minimum cost. The cost of a query operation is mainly depend on its selectivity i.e., the proportion of the input relations that forms the output. Following are the main components used to determine the cost of execution of a query:

(a) Access cost of secondary storage: Let me add this, beware that access cost to secondary storage consists of cost of database manipulation operations which includes searching, writing, reading of data blocks stored in the secondary memory. The cost of searching depends upon the type of indexes (primary, secondary, hashed), type of file structure, ordering of relation in addition to physical storage location like file blocks are allocated contiguously on the same disk or scattered on the disk.

(b) Storage cost: Storage cost consists of cost of storing intermediate results (tables or files) that are generated by the execution strategy for the query.

(c) Computation cost: Computation cost consists of performing in-memory operations during query execution such as sorting of records in a file, merging of records, performing computations on field values, searching of records. These are mainly performed on data buffers.

(d) Memory usage cost: It consists of cost of pertaining to the number of memory buffers needed during query execution.

(e) Communication cost: It consists of the cost of transferring query and its result from database location to the location of terminal where the query is originated.

Summary

In this unit, I introduced to you Query optimization. The various issues of query optimization were also stated. The different types of query optimization and an explanation of the two types of query optimization.

Self-Assessment Questions

1. What are the two types of query optimization technique?
2. State 4 (four) components of cost based query optimization.

Tutor Marked Assignment

Distinguish between Heuristic Query Optimization and Cost Based Optimization.

References

Ramakrishnan R. and Gherke J. (1996). Database Management System, second edition. McGraw-Hill Higher Education.

Silberschatz A., Korth H. F., and Sudarshan S. (2003). Database System Concepts. McGraw-Hill Education.

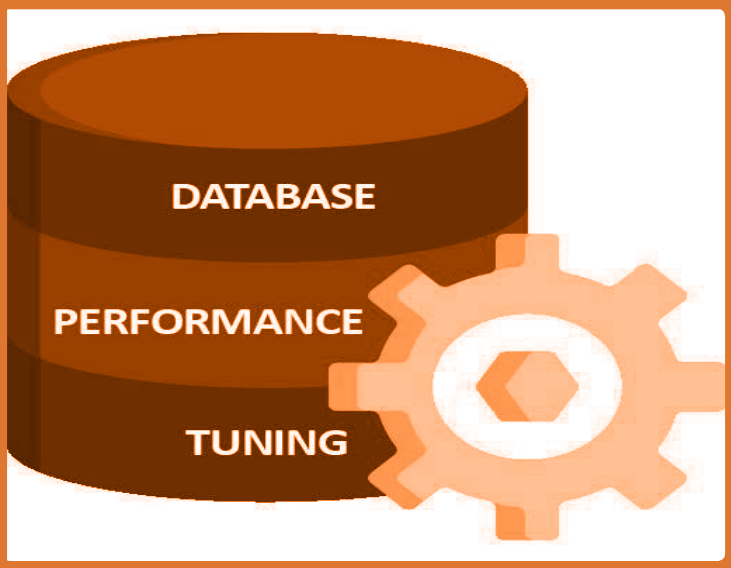
Further Reading

https://www.tutorialspoint.com/sql/index.htm

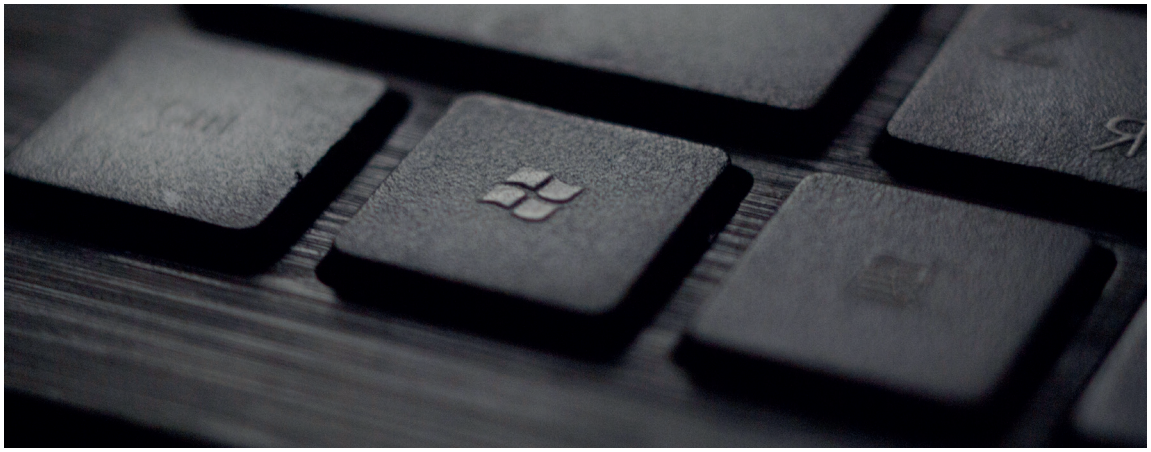
USB Cable
Source: Unsplash

MODULE 6

STUDY OF SOME STANDARD DATABASE SYSTEMS



UNITS



picture of laptop Ke pad
Source: Unsplash by tadas sar

UNIT 1

MICROSOFT ACCESS

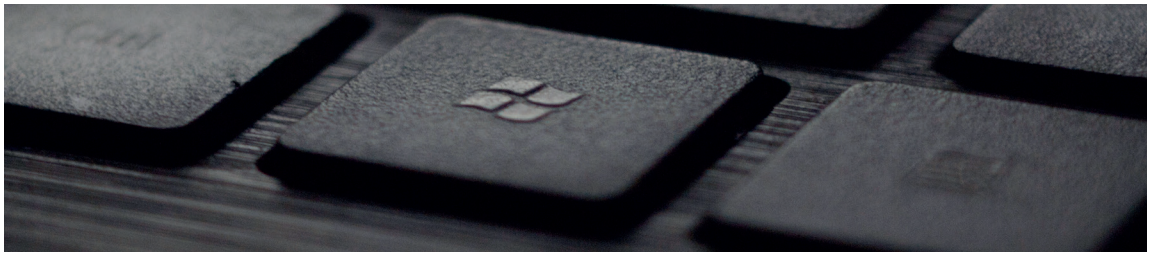
Introduction

In this unit, you will learn about how to create and manipulate databases by using Microsoft Access which is a Database Management System (DBMS) from Microsoft that combines the relational Microsoft Jet Database Engine with a graphical user interface and software development tools. It is a member of the Microsoft Office suite of applications, included in the professional and higher editions.

Learning Outcomes

When you have studies this unit, you should be able to:

- Use Microsoft Access application.
- Create database and tables in Microsoft Access.
- Populate a table with data types acceptable in Microsoft Access.



picture of laptop Ke pad
Source: Unsplash by tadas sar

Main Content

Microsoft Access Objects [SAQ2]

READING TIME: 2 MINS

microsoft access
source: simplilearn

I should start by informing you that Microsoft (MS) Access uses "objects" to help the user list and organize information, as well as prepare specially designed reports. When you create a database, Access offers you Tables, Queries, Forms, Reports, Macros, and Modules. Databases in Access are composed of many objects but the following are the major objects:

i. **Tables**

ii. **Queries**

iii. **Forms**

iv. **Reports**

Together, these objects allow you to enter, store, analyze, and compile your data. Here is a summary of the major objects in an Access database;

Table

Bear in mind table is an object that is used to define and store data. When you create a new table, Access asks you to define fields which is also known as column headings

Each field must have a unique name, and data type.

Tables contain fields or columns that store different kinds of data, such as a name or an address, and records or rows that collect all the information about a particular

instance of the subject, such as all the information about a customer or employee etc. You can define a primary key, one or more fields that have a unique value for each record, and one or more indexes on each table to help retrieve your data more quickly.

Query

An object that provides a custom view of data from one or more tables. Queries are a way of searching for and compiling data from one or more tables.

i. Running a query is like asking a detailed question of your database.

ii. When you build a query in Access, you are defining specific search conditions to find exactly the data you want.

iii. In Access, you can use the graphical query by example facility or you can write Structured Query Language (SQL) statements to create your queries.

iv. You can define queries to Select, Update, Insert, or Delete data.

v. You can also define queries that create new tables from data in one or more existing tables.

Form

Form is an object in a desktop database designed primarily for data input or display or for control of application execution. You use forms to customize the presentation of data that your application extracts from queries or tables.

i. Forms are used for entering, modifying, and viewing records.

ii. The reason forms are used so often is that they are an easy way to guide people toward entering data correctly.

iii. When you enter information into a form in Access, the data goes exactly where the database designer wants it to go in one or more related tables.

Report

Report is an object in desktop databases designed for formatting, calculating, printing, and summarizing selected data.

i. You can view a report on your screen before you print it.

ii. If forms are for input purposes, then reports are for output.

iii. Anything you plan to print deserves a report, whether it is a list of names and addresses, a financial summary for a period, or a set of mailing labels.

iv. Reports are useful because they allow you to present components of your database in an easy-to-read format.

v. You can even customize a report's appearance to make it visually appealing.

vi. Access offers you the ability to create a report from any table or query.

Other Microsoft Access Objects

READING TIME: 1 MIN

Macro

This object is a structured definition of one or more actions that you want Access to perform in response to a defined event. An Access Macro is a script for doing some job. For example, to create a button which opens a report, you could use a macro which will fire OpenReport action.

i. You can include simple conditions in macros to specify when one or more actions in the macro should be performed or skipped.
ii. You can use macros to open and execute queries, to open tables, or to print or view reports.
iii. You can also run other macros or Visual Basic procedures from within a macro.
iv. Data macros can be attached directly to table events such as inserting new records, editing existing records, or deleting records.
v. Data macros in web apps can also be stand-alone objects that can be called from other data macros or macro objects.

Module

Module is an object in desktop databases containing custom procedures that you code using Visual Basic. Modules provide a more discrete flow of actions and allow you to trap errors.

i. Everything that can be done in a macro can also be done in a module, but you don't get the macro interface that prompts you what is needed for each action.

ii. Modules are far more powerful, and are essential if you plan to write code for a multi-user environment, because macros cannot include error handling.

iii. Modules can be standalone objects containing functions that can be called from anywhere in your application, or they can be directly associated with a form or a report to respond to events on the associated form or report.

Microsoft Access – Creating a Database

READING TIME: 1 MIN

To create a database from a template, we first need to open MS Access and you will see the following screen in which different Access database templates are displayed as shown in figure 1.

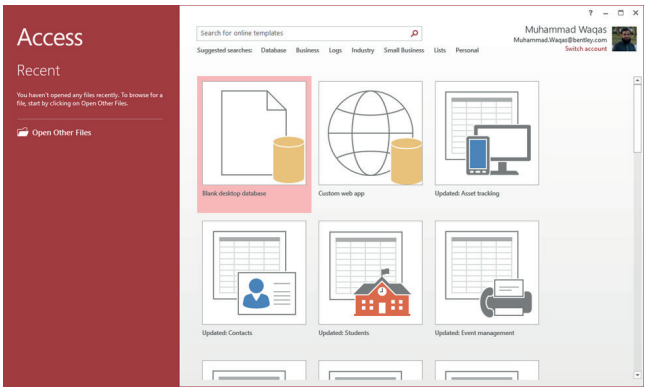


Figure 1: Microsoft Access Templates

To view all the possible databases, you can scroll down or you can also use the search box. Let us enter project in the search box and press Enter. You will see the database templates related to project management.

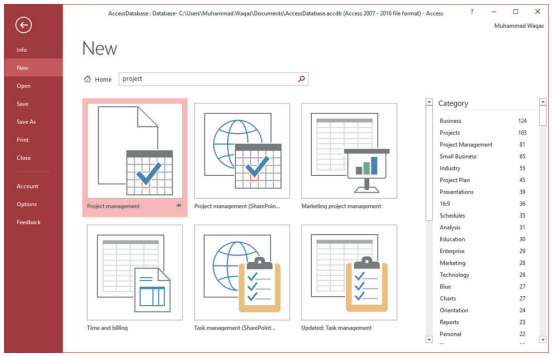


Figure 2: Selecting the first template

Select the Project Management template which is the first template as shown in figure 2. You will see more information related to this template.

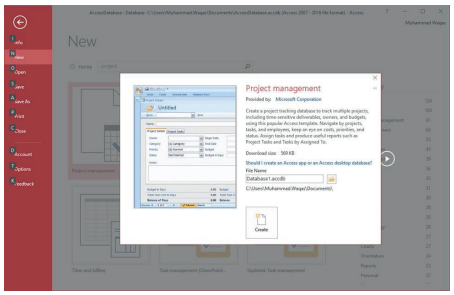


Figure 3. Project Management Template

After selecting a template related to your requirements, enter a name in the File name field and you can also specify another location for your file if you want as shown in figure 3.

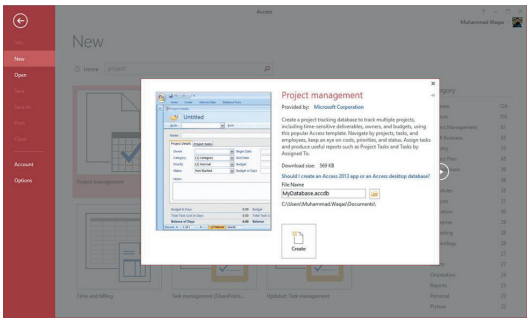


Figure 4: Pressing the Create Option

Now, press the Create option. Access will download that database template and open a new blank database as shown in the following screenshot

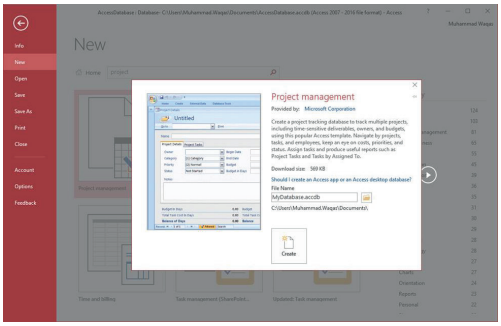


Figure 5: Clicking the Navigation pane

Now, click the Navigation pane on the left side and you will see all the other objects that come with this database.

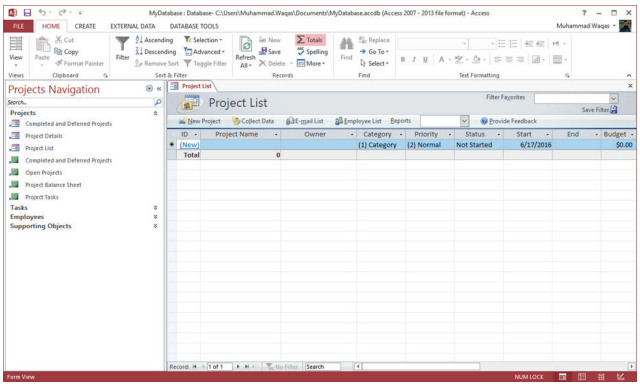


Figure 6: Projects Navigation Screen

Click the Projects Navigation and select the Object Type in the menu.

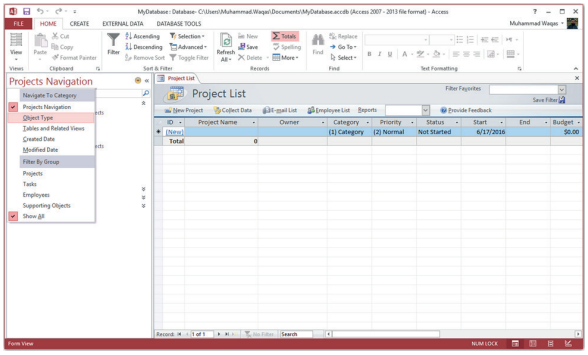


Figure 7: Object type menu

You will now see all the objects types — tables, queries, and so on.

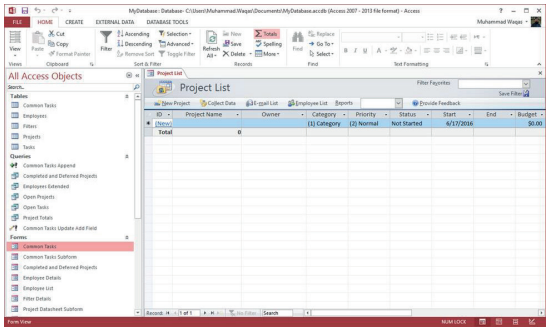


Figure 8: All Access objects

Creating Blank Database

READING TIME: 1 MIN

Are you aware that sometimes database requirements can be so specific that using and modifying the existing templates requires more work than just creating a database from scratch? In such case, we make use of blank database.

Step 1: Let us now start by opening MS Access.

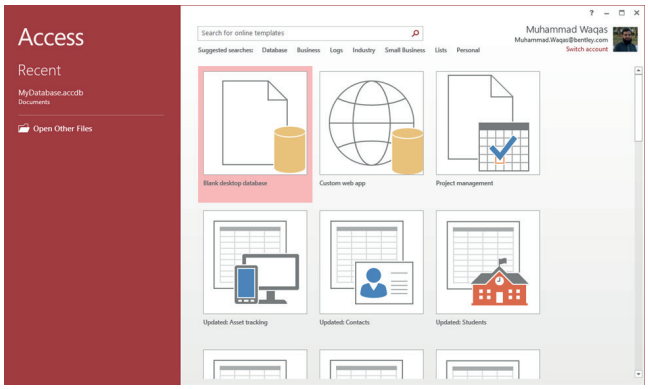


Figure 9: MS Access desktop

Step 2: Select Blank desktop database. Enter the name and click the Create button as shown in figure 9.

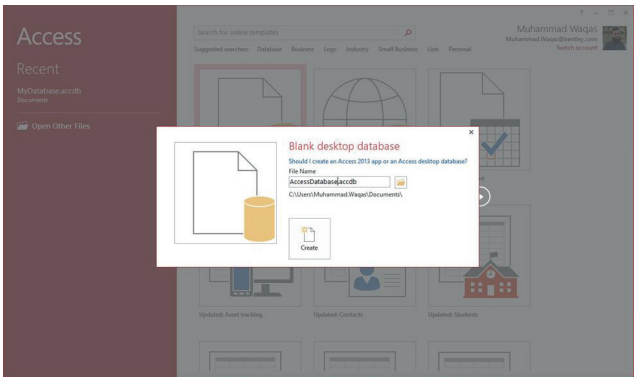


Figure 10: Blank Desktop Database

Step 3: Access will create a new blank database and will open up the table which is also completely blank as shown in figure 10.

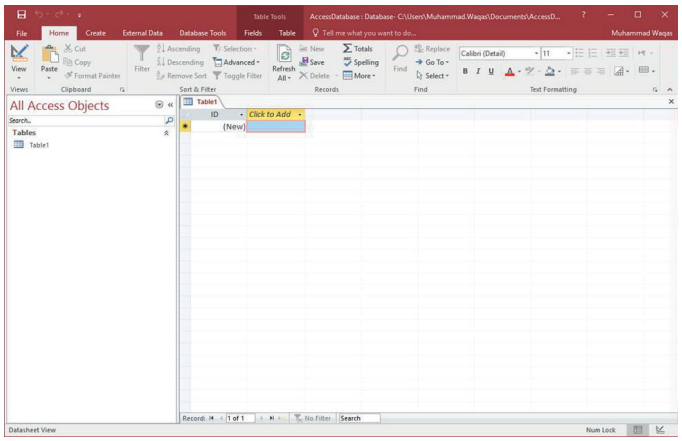


Figure 11: Blank Database

Microsoft Access – Data Types [SAQ1]

READING TIME: 1 MIN

I should let you know that every field in a table has properties and these properties define the field's characteristics and behaviour. The most important property for a field is its data type. A field's data type determines what kind of data it can store. MS Access supports different types of data, each with a specific purpose.

i. The data type determines the kind of the values that users can store in any given field.

ii. Each field can store data consisting of only a single data type.

Table 1 shows some of the most common data types you will find used in a typical Microsoft Access database.

Type of Data	Description	Size
Short Text	Text or combinations of text and numbers, including numbers that do not require calculating (e.g. phone numbers).	Up to 255 characters.
Long Text	Lengthy text or combinations of text and numbers.	Up to 63, 999 characters.
Number	Numeric data used in mathematical calculations.	1, 2, 4, or 8 bytes (16 bytes if set to Replication ID).
Date/Time	Date and time values for the years 100 through 9999.	8 bytes.
Currency	Currency values and numeric data used in mathematical calculations involving data with one to four decimal places.	8 bytes.
AutoNumber	A unique sequential (incremented by 1) number or random number assigned by Microsoft Access whenever a new record is added to a table.	4 bytes (16 bytes if set to Replication ID).
Yes/No	Yes and No values and fields that contain only one of two values (Yes/No, True/False, or On/Off).	1 bit.

Table 1: Microsoft Access Data Types

i. If you use previous versions of Access, you will notice a difference for two of those data types.

ii. In Access 2013, we now have two data types — short text and long text. In previous versions of Access these data types were called text and memo.

iii. The text field is referred to as short text and your memo field is now called long text.

Table 2 shows some of the other more specialized data types, you can choose from in Access.

Data Types	Description	Size
Attachment	Files, such as digital photos. Multiple files can be attached per record. This data type is not available in earlier versions of Access.	Up to about 2 GB.
OLE objects	OLE objects can store pictures, audio, video, or other BLOBs (Binary Large Objects)	Up to about 2 GB.
Hyperlink	Text or combinations of text and numbers stored as text and used as a hyperlink address.	Up to 8,192 (each part of a Hyperlink data type can contain up to 2048 characters).
Lookup Wizard	The Lookup Wizard entry in the Data Type column in the Design view is not actually a data type. When you choose this entry, a wizard starts to help you define either a simple or complex lookup field. A simple lookup field uses the contents of another table or a value list to validate the contents of a single value per row. A complex lookup field allows you to store multiple values of the same data type in each row.	Dependent on the data type of the lookup field.
Calculated	You can create an expression that uses data from one or more fields. You can designate different result data types from the expression.	You can create an expression that uses data from one or more fields. You can designate different result data types from the expression.

Table 2: More specialized data types

These are all the different data types that you can choose from when creating fields in a Microsoft Access table.

Microsoft Access – Create Tables

READING TIME: 2 MINS

When you create a database, you store your data in tables. Because other database objects depend so heavily on tables, you should always start your design of a database by creating all of its tables and then creating any other object. Before you create tables, carefully consider your requirements and determine all the tables that you need.

Let us try and create the first table that will store the basic contact information concerning the employees as shown in table 3:

Field Name	Data Type
EmployeeID	AutoNumber
FirstName	Short Text
LastName	Short Text
Address1	Short Text
Address2	Short Text
City	Short Text
State	Short Text
Zip	Short Text
Phone	Short Text
PhoneType	Short Text

Table 3: Basic contact information about employees

Let us now have short text as the data type for all these fields and open a blank database in Access as shown in figure 12.

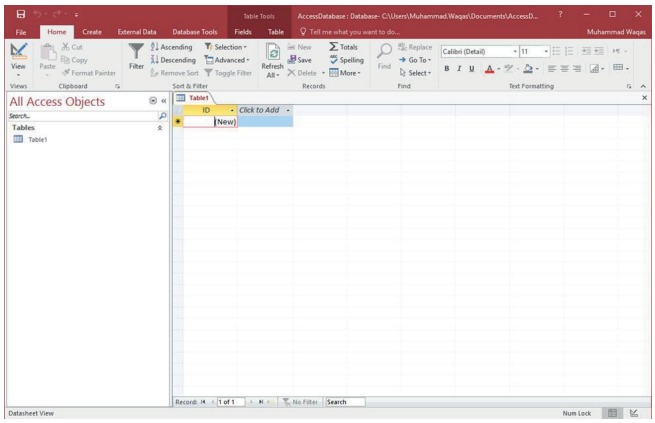


Figure 12: Blank Database

This is where we left things off. We created the database and then Access automatically opened up this table-one-datasheet view for a table.

Let us now go to the Field tab and you will see that it is also automatically created. The ID which is an AutoNumber field acts as our unique identifier and is the primary key for this table.

The ID field has already been created and we now want to rename it to suit our conditions. This is an Employee table and this will be the unique identifier for our employees.

Click on the Name & Caption option in the Ribbon and you will see the following dialog box.

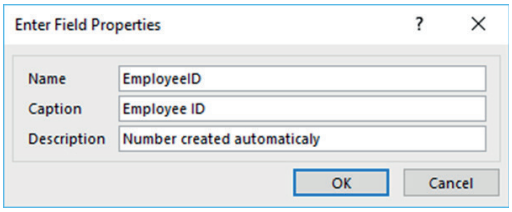


Figure 14: Name and Caption dialog box

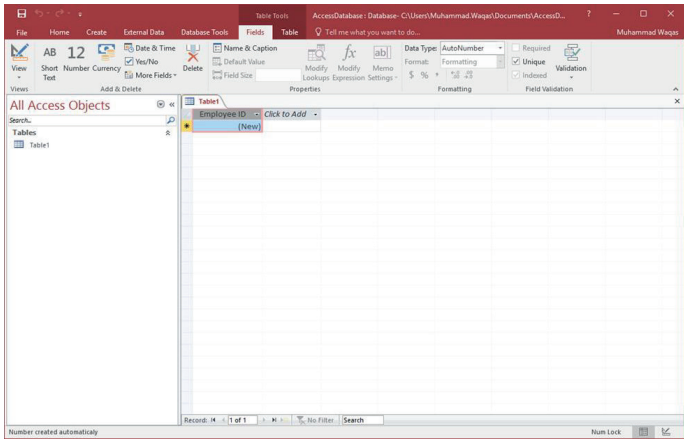


Figure 15: Change the field name

We now have our employee ID field with the caption Employee ID. This is automatically set to auto number so we don't really need to change the data type.

Let us now add some more fields by clicking on click to add.

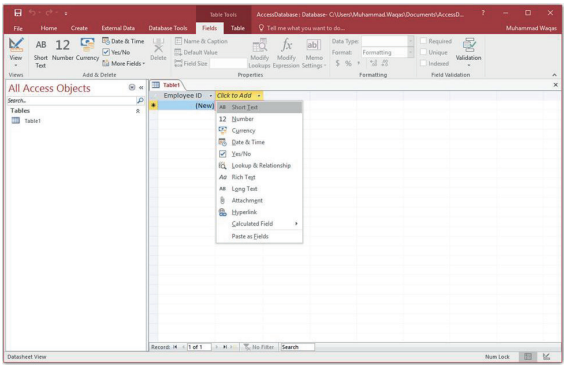


Figure 16: Choose short text field

Type FirstName as the field name. Similarly, add all the required fields as shown in the following screenshot.

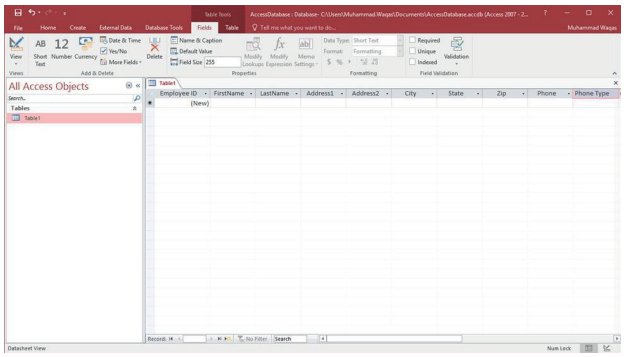


Figure 17: Adding the required fields

Once all the fields are added, click the Save icon.

You will now see the Save As dialog box, where you can enter a table name for the table.

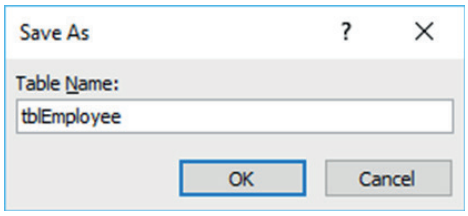


Figure 18: Table name

Enter the name of your table in the Table Name field. Here the tbl prefix stands for table. Let us click Ok and you will see your table in the navigation pane.

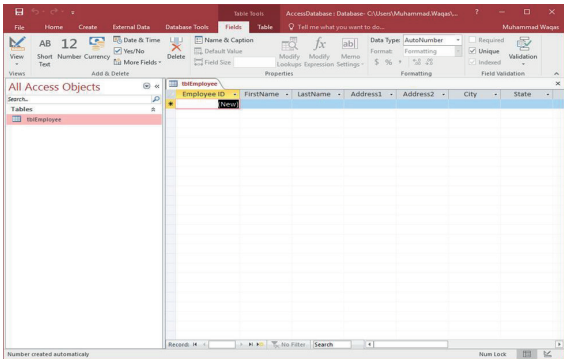


Figure 19: Entering the table name

Microsoft Access – Adding Data

READING TIME: 1 MIN

You should be aware that an Access database is not a file in the same sense as a Microsoft Office Word document or a Microsoft Office PowerPoint are. Instead, an Access database is a collection of objects like tables, forms, reports, queries etc. that must work together for a database to function properly. We have now created two tables with all of the fields and field properties necessary in our database. To view, change, insert, or delete data in a table within Access, you can use the table's Datasheet View.

microsoft spreedsheet
source: xda-developers.com

A datasheet is a simple way to look at your data in rows and columns without any special formatting.

Whenever you create a new web table, Access automatically creates two views that you can start using immediately for data entry.

A table open in Datasheet View resembles an Excel worksheet, and you can type or paste data into one or more fields.

You do not need to explicitly save your data. Access commits your changes to the table when you move the cursor to a new field in the same row, or when you move the cursor to another row.

By default, the fields in an Access database are set to accept a specific type of data, such as text or numbers. You must enter the type of data that the field is set to accept. If you don't, Access displays an error message:

Let us add some data into your tables by opening the Access database we have created.

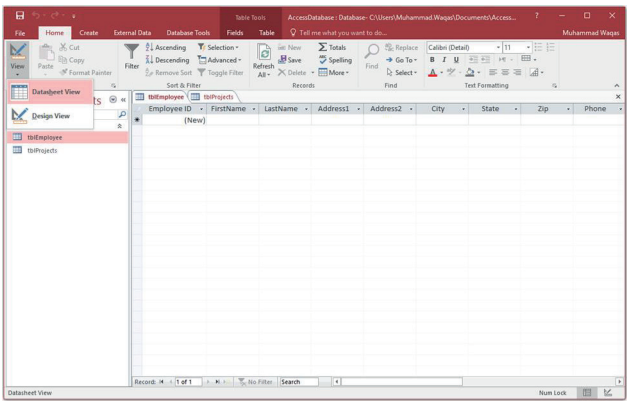


Figure 20: Opening the access database

Select the Views > Datasheet View option in the ribbon and add some data as shown in the following screenshot.

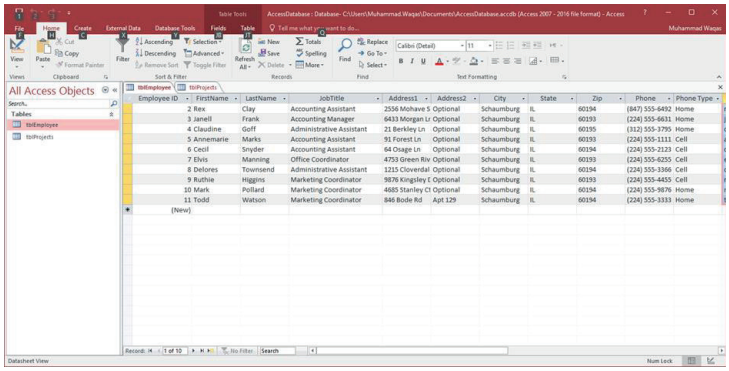


Figure 21: Adding data to a table

Similarly, add some data in the second table as well as shown in the following screenshot.

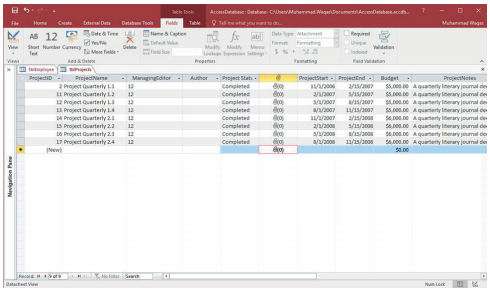


Figure 22: Adding data to a second table

You can now see that inserting a new data and updating the existing data is very simple in Datasheet View as working in spreadsheet. But if you want to delete any data you need to select the entire row first as shown in the following screenshot.

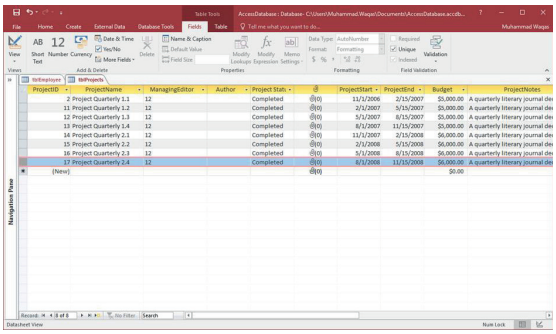


Figure 24: To delete data

Now press the delete button. This will display the confirmation message.

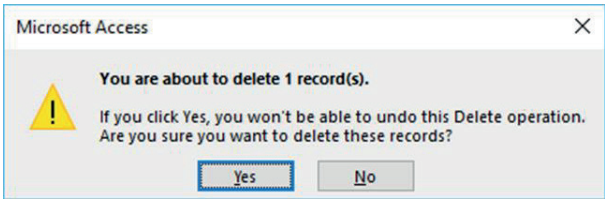


Figure 25: Click the yes button

Click Yes and you will see that the selected record is deleted now.

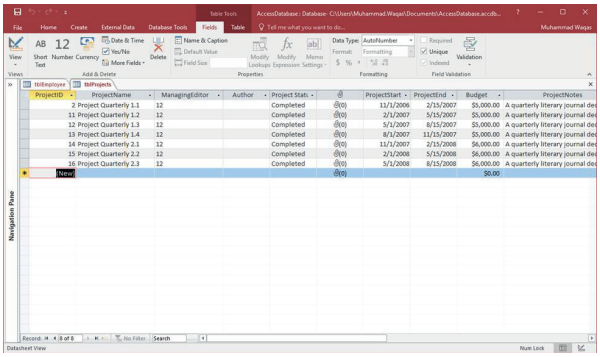


Figure 26: Deleting a record

Summary

In this unit we treated Microsoft access interface, it explains how to create a database and add tables to the database. It states the acceptable data types for attributes in access. Adding data to the created table was also explained.

Self-Assessment Questions

1. State the acceptable data types in Microsoft Access.
2. What is a Database Table?

Tutor Marked Assignment

Create a database in Microsoft access.

Add a table to the created database.

Add data to the table.

References

Gupta S. B. and Mittal A., (2004). Introduction to database management systems. Pearson Education.

Ramakrishnan R. and Gherke J. (1996). Database Management System, second edition. McGraw-Hill Higher Education.

Further Reading

Silberschatz A., Korth H. F., and Sudarshan S. (2003). Database System

Concepts. McGraw-Hill Education.

https://Tutorialspoint.com/Ms-access/index.asp

Oracle sign Photo Illustration by Aleksander Kalka NurPhoto
Source: Getty Images

UNIT 2

ORACLE

Introduction

In this unit, you will be taught how to create database files in Oracle. Oracle is a relational database management system. It is known as Oracle database, OracleDB or simply Oracle. It is produced and marketed by Oracle Corporation. Oracle database is the first database designed for enterprise grid computing. The enterprise grid computing provides the most flexible and cost effective way to manage information and applications.

Learning Outcomes

When you have studies this unit, you should be able to:

- Create a table, drop or delete a table in Oracle.
- Execute some query statement such as INSERT, SELECT, UPDATE and use some clauses to further specify query options.

Oracle sign Photo Illustration by Aleksander Kalka NurPhoto
Source: Getty Images

Main Content

Oracle – Create a Table [SAQ1]

READING TIME: 1 MIN

In Oracle, CREATE TABLE statement is used to create a new table in the database.

To create a table, you have to name that table and define its columns and datatype for each column.

```
Syntax:
CREATE TABLE table_name
 (
  column1 datatype [ NULL | NOT NULL ],
  column2 datatype [ NULL | NOT NULL ],
  ...
  column_n datatype [ NULL | NOT NULL ]
  );
```

Parameters used in syntax

i. **table_name**: It specifies the name of the table which you want to create.

ii. **column1, column2**, ... column n: It specifies the columns which you want to add in the table. Every column must have a datatype. Every column should either be defined as "NULL" or "NOT NULL". In the case, the value is left blank; it is treated as "NULL" as default.

```
Oracle CREATE TABLE Example with primary key
CREATE TABLE customers
( customer_id number(10) NOT NULL,
  customer_name varchar2(50) NOT NULL,
  city varchar2(50),
  CONSTRAINT customers_pk PRIMARY KEY (customer_id)
  );
```

What is Primary key

A primary key is a single field or combination of fields that contains a unique record. It must be filled. None of the field of primary key can contain a null value. A table can have only one primary key.

Oracle – Drop Table

READING TIME: 1 MIN

Oracle DROP TABLE statement is used to remove or delete a table from the Oracle database.

Syntax

```
DROP [schema_name].TABLE table_name
[ CASCADE CONSTRAINTS ]
[ PURGE ];
```

Parameters

schema_name: It specifies the name of the schema that owns the table.

table_name: It specifies the name of the table which you want to remove from the Oracle database.

CASCADE CONSTRAINTS: It is optional. If specified, it will drop all referential integrity constraints as well.

PURGE: It is also optional. If specified, the table and its dependent objects are placed in the recycle bin and can't be recovered.

DROP TABLE Example

```
DROP TABLE customers;
```

This will drop the table named customers.

DROP TABLE Example with PURGE parameter

```
DROP TABLE customers PURGE
```

This statement will drop the table called customers and issue a PURGE so that the space associated with the customers table is released and the customers table is not placed in recycle bin. So, it is not possible to recover that table if required.

Oracle – Select Statement

READING TIME: 1 MIN

Have it at the back of your mind that the Oracle SELECT statement is used to retrieve data from one or more than one tables, object tables, views, object views etc.

Syntax

```
SELECT expressions
FROM tables
WHERE conditions;
```

Parameters

1) expressions: It specifies the columns or calculations that you want to retrieve.

2) tables:This parameter specifies the tables that you want to retrieve records from. There must be at least one table within the FROM clause.

3) conditions: It specifies the conditions that must be followed for selection.

Select Example: select all fields

Let's take an example to select all fields from an already created table named customers

```
SELECT *
FROM customers;
output
```

NAME	AGE	ADDRESS	SALARY
mohan	21	ghaziabad	20000
rohan	22	delhi	22000
sohan	25	noida	24000
alex	28	Paris	40000

4 rows returned in 0.02 seconds

```
Select Example: select specific fields
Example
SELECT age, address, salary
FROM customers
WHERE  age < 25
AND salary > '20000'
ORDER BY age ASC, salary DESC;
```

AGE	ADDRESS	SALARY
22	delhi	22000

1 rows returned in 0.00 seconds

Oracle – Insert Statement

READING TIME: 1 MIN

In Oracle, INSERT statement is used to add a single record or multiple records into the table.

Syntax: (Inserting a single record using the Values keyword):

```
INSERT INTO table
(column1, column2, ... column_n )
VALUES
(expression1, expression2, ... expression_n );
```

Syntax: (Inserting multiple records using a SELECT statement):

```
INSERT INTO table
(column1, column2, ... column_n )
SELECT expression1, expression2, ... expression_n
FROM source_table
WHERE conditions;
```

Parameters:

1) table: The table to insert the records into.

2) column1, column2, ... column_n:

The columns in the table to insert values.

3) expression1, expression2, ... expression_n:

The values to assign to the columns in the table. So column1 would be assigned the value of expression1, column2 would be assigned the value of expression2, and so on.

4) source_table:

The source table when inserting data from another table.

5) conditions:

The conditions that must be met for the records to be inserted.

Oracle Insert Example: By SELECT statement

This method is used for more complicated cases of insertion. In this method insertion is done by SELECT statement. This method is used to insert multiple elements.

See this example:

In this method, we insert values to the "suppliers" table from "customers" table. Both tables are already created with their respective columns. statement. This method is used to insert multiple elements.

See this example:

In this method, we insert values to the "suppliers" table from "customers" table. Both tables are already created with their respective columns.

Execute this query:

```
INSERT INTO suppliers
(supplier_id, supplier_name)
SELECT age, address
FROM customers
WHERE age > 20;
```

Oracle – Update Statement

READING TIME: 1 MIN

In Oracle, UPDATE statement is used to update the existing records in a table. You can update a table in 2 ways.

Traditional Update table method

Syntax:

```
UPDATE table
 SET column1 = expression1,
 column2 = expression2,
  ...
 column_n = expression_n
WHERE conditions;
```

Update Table by selecting rocords from another table

Syntax:

```
UPDATE table1
SET column1 = (SELECT expression1
 FROM table2
 WHERE conditions)
WHERE conditions;
```

Parameters:

1) column1, column2, ... column_n:

It specifies the columns that you want to update.

2) expression1, expression2, ...expression_n:

This specifies the values to assign to the column1, column2, ?. column_n.

3) conditions: It specifies the conditions that must be fulfilled for execution of UPDATE statement.

Oracle Update Example: (Update single column)

```
UPDATE suppliers
SET supplier_name = 'Kingfisher'
WHERE supplier_id = 2;
```

This example will update the supplier_name as "Kingfisher" where "supplier_id" is 2.

Oracle Update Example: (By selecting records from another table)

```
UPDATE customers
SET name = (SELECT supplier_name
FROM suppliers
WHERE suppliers.supplier_name = customers.name)
WHERE age < 25;
Output:
2 row(s) updated.
0.02 seconds
```

Oracle – Delete Statement

READING TIME: 1 MIN

I should include that in Oracle, DELETE statement is used to remove or delete a single record or multiple records from a table.

```
Syntax
DELETE FROM table_name
WHERE conditions;
```

Parameters

1) table_name: It specifies the table which you want to delete.

2) conditions: It specifies the conditions that must met for the records to be deleted.

Oracle Delete Example: On one condition

```
DELETE FROM customers
WHERE name = 'Sohan';
```

This statement will delete all records from the customer table where name is "Sohan".

Oracle Delete Example: On multiple conditions

```
DELETE FROM customers
WHERE last_name = 'Maurya'
AND customer_id > 2;
```

This statement will delete all records from the customers table where the last_name is "Maurya" and the customer_id is greater than 2.

Summary

In this unit so far, we examined Oracle database. We explained the creation of table in oracle, deletion of a table, insert statement, select statement, update statement and delete statement. Examples were also given for easy and practical understanding with the inclusion of clauses.

Self-Assessment

Create a table in Oracle.

Tutor Marked Assignment

Insert values into the table created.

Update the value added.

Delete the table.

References

Elmasri R. and Navathe S. B., (2011). Fundamentals of Database Systems. Addison-Wesley.

Gupta S. B. and Mittal A., (2004). Introduction to database management systems. Pearson Education.

Further Reading

https://www.javapoint.com/oracle-tutorial

https://Tutorialspoint.com/oracle_sql/index.asp

Source: Bleepingcomputer.com

UNIT 3

MYSQL

Introduction

You should cognize that MySQL is the most popular Open Source Relational SQL Database Management System. MySQL is one of the best RDBMS being used for developing various web-based software applications. MySQL is developed, marketed and supported by MySQL AB, which is a Swedish company. This tutorial will give you a quick start to MySQL and make you comfortable with MySQL programming.

Learning Outcomes

When you have studies this unit, you should be able to:

- Create and delete a database in MySQL.
- Add a table to the database.
- Execute some query statements such as INSERT, DELETE, SELECT and WHERE.

Source: Bleepingcomputer.com

Main Content

MySQL – Installation

READING TIME: 1 MINS

I should begin by highlighting that the default installation on any version of Windows is now much easier than it used to be, as MySQL now comes neatly packaged with an installer. Simply download the installer package, unzip it anywhere and run the setup.exe file.

MySQL logo
sorce: cloudooklet

The default installer setup.exe will walk you through the trivial process and by default will install everything under C:\mysql.

Test the server by firing it up from the command prompt the first time. Go to the location of the mysqld server which is probably C:\mysql\bin, and type

mysqld.exe --console

If all went well, you will see some messages about startup and InnoDB. If not, you may have a permissions issue. Make sure that the directory that holds your data is accessible to whatever user (probably MySQL) the database processes run under.

MySQL will not add itself to the start menu, and there is no particularly nice GUI way to stop the server either. Therefore, if you tend to start the server by double clicking the mysqld executable, you should remember to halt the process by hand by using mysqladmin, Task List, Task Manager, or other Windows-specific means.

My SQL – Administration

READING TIME: 1 MIN

First check if your MySQL server is running or not. You can use the following command to check it –

root@host# cd /usr/bin

./safe_mysqld &

Now, if you want to shut down an already running MySQL server, then you can do it by using the following command –

```
root@host# cd /usr/bin
./mysqladmin -u root -p shutdown
Enter password: ******
```

For adding a new user to MySQL, you just need to add a new entry to the user table in the database mysql.

The following program is an example of adding a new user guest with SELECT, INSERT and UPDATE privileges with the password guest123; the

SQL query is –

```
root@host# mysql -u root -p
Enter password:*******
mysql> use mysql;
Database changed

mysql> INSERT INTO user
(host, user, password,
 select_priv, insert_priv, update_priv)
 VALUES ('localhost', 'guest',
 PASSWORD('guest123'), 'Y', 'Y', 'Y');
Query OK, 1 row affected (0.20 sec)
```

```
mysql> FLUSH PRIVILEGES;
Query OK, 1 row affected (0.01 sec)

mysql> SELECT host, user, password FROM user WHERE
user = 'guest';
+-----------+---------+------------------+
|   host    |  user   |    password      |
+-----------+---------+------------------+
| localhost |  guest  | 6f8c114b58f2ce9e |
+-----------+---------+------------------+
1 row in set (0.00 sec)
```

When adding a new user, remember to encrypt the new password using PASSWORD() function provided by MySQL. As you can see in the above example, the password mypass is encrypted to 6f8c114b58f2ce9e.

MySQL – Connection

READING TIME: 1 MIN

You can establish the MySQL database using the mysql binary at the command prompt.

Example

Here is a simple example to connect to the MySQL server from the command prompt –

```
[root@host]# mysql -u root -p
Enter password:******
```

This will give you the mysql> command prompt where you will be able to execute any SQL command. Following is the result of above command:

The following code block shows the result of above code –

```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2854760 to server version: 5.0.9
Type 'help;' or '\h' for help. Type '\c' to clear the buffer.
```

In the above example, we have used root as a user but you can use any other user as well. Any user will be able to perform all the SQL operations, which are allowed to that user.

You can disconnect from the MySQL database any time using the exit command at mysql> prompt.

```
mysql> exit
Bye
```

MySQL – Create a Database

READING TIME: 1 MIN

You would need special privileges to create or to delete a MySQL database. So, assuming you have access to the root user, you can create any database using the mysql mysqladmin binary.

Example: Here is a simple example to create a database called TUTORIALS –

```
[root@host]# mysqladmin -u root -p create TUTORIALS
Enter password:******
```

This will create a MySQL database called TUTORIALS.

You would need special privileges to create or to delete a MySQL database. So, assuming you have access to the root user, you can create any database using the mysql mysqladmin binary.

Be careful while deleting any database because you will lose your all the data available in your database.

Here is an example to delete a database(TUTORIALS) created in the previous chapter –

```
root@host]# mysqladmin -u root -p drop TUTORIALS
Enter password:******
```

This will give you a warning and it will confirm if you really want to delete this database or not.

```
Dropping the database is potentially a very bad thing to do.
Any data stored in the database will be destroyed.

Do you really want to drop the 'TUTORIALS' database [y/N] y
Database "TUTORIALS" dropped
```

It is very simple to select a database from the mysql> prompt. You can use the SQL command use to select a database.

Example: Here is an example to select a database called TUTORIALS –

```
[root@host]# mysql -u root -p
Enter password:******
mysql> use TUTORIALS;
Database changed
mysql>
```

Now, you have selected the TUTORIALS database and all the subsequent operations will be performed on the TUTORIALS database.

MySQL – Data Types

READING TIME: 4 MINS

You should be notified that properly defining the fields in a table is important to the overall optimization of your database. You should use only the type and size of field you really need to use. For example, do not define a field 10 characters wide, if you know you are only going to use 2 characters. These type of fields (or columns) are also referred to as data types, after the type of data you will be storing in those fields.

MySQL uses many different data types broken into three categories –

i. Numeric

ii. Date and Time

iii. String Types.

Let us now discuss them in detail.

Numeric Data Types

MySQL uses all the standard ANSI SQL numeric data types, so if you're coming to MySQL from a different database system, these definitions will look familiar to you.

The following list shows the common numeric data types and their descriptions –

i. ***INT*** – A normal-sized integer that can be signed or unsigned. If signed, the allowable range is from -2147483648 to 2147483647. If unsigned, the allowable range is from 0 to 4294967295. You can specify a width of up to 11 digits.

ii. **TINYINT** – A very small integer that can be signed or unsigned. If signed, the allowable range is from -128 to 127. If unsigned, the allowable range is from 0 to 255. You can specify a width of up to 4 digits.

iii. **SMALLINT** – A small integer that can be signed or unsigned. If signed, the allowable range is from -32768 to 32767. If unsigned, the allowable range is from 0 to 65535. You can specify a width of up to 5 digits.

iv. **MEDIUMINT** – A medium-sized integer that can be signed or unsigned. If signed, the allowable range is from -8388608 to 8388607. If unsigned, the allowable

range is from 0 to 16777215. You can specify a width of up to 9 digits.

v. **BIGINT** – A large integer that can be signed or unsigned. If signed, the allowable range is from -9223372036854775808 to 9223372036854775807. If unsigned, the allowable range is from 0 to 18446744073709551615. You can specify a width of up to 20 digits.

FLOAT(M,D) – A floating-point number that cannot be unsigned. You can define the display length (M) and the number of decimals (D). This is not required and will default to 10,2, where 2 is the number of decimals and 10 is the total number of digits (including decimals). Decimal precision can go to 24 places for a FLOAT.

vi. **DOUBLE(M,D)** – A double precision floating-point number that cannot be unsigned. You can define the display length (M) and the number of decimals (D). This is not required and will default to 16,4, where 4 is the number of decimals. Decimal precision can go to 53 places for a DOUBLE. REAL is a synonym for DOUBLE.

vii. **DECIMAL(M,D)** – An unpacked floating-point number that cannot be unsigned. In the unpacked decimals, each decimal corresponds to one byte. Defining the display length (M) and the number of decimals (D) is required. NUMERIC is a synonym for DECIMAL.

Date and Time Types

The MySQL date and time datatypes are as follows:

i. **DATE** – A date in YYYY-MM-DD format, between 1000-01-01 and 9999-12-31. For example, December 30th, 1973 would be stored as 1973-12-30.

ii. **DATETIME** – A date and time combination in YYYY-MM-DD HH:MM:SS format, between 1000-01-01 00:00:00 and 9999-12-31 23:59:59. For example, 3:30 in the afternoon on December 30th, 1973 would be stored as 1973-12-30 15:30:00.

iii. **TIMESTAMP** – A timestamp between midnight, January 1st, 1970 and sometime in 2037. This looks like the previous DATETIME format, only without the hyphens between numbers; 3:30 in the afternoon on December 30th, 1973 would be stored as 19731230153000 (YYYYMMDDHHMMSS).

iv. **TIME** – Stores the time in a HH:MM:SS format.

YEAR(M) – Stores a year in a 2-digit or a 4-digit format. If the length is specified as 2 (for example YEAR(2)), YEAR can be between 1970 to 2069 (70 to 69). If the length is specified as 4, then YEAR can be 1901 to 2155. The default length is 4.

String Types

Although the numeric and date types are fun, most data you'll store will be in a string format. This list describes the common string datatypes in MySQL.

i. **CHAR(M)** – A fixed-length string between 1 and 255 characters in length (for example CHAR(5)), right-padded with spaces to the specified length when stored. Defining a length is not required, but the default is 1.

ii. **VARCHAR(M)** – A variable-length string between 1 and 255 characters in length. For example, VARCHAR(25). You must define a length when creating a VARCHAR field.

iii. **BLOB or TEXT** – A field with a maximum length of 65535 characters. BLOBs are "Binary Large Objects" and are used to store large amounts of binary data, such as images or other types of files. Fields defined as TEXT also hold large amounts of data. The difference between the two is that the sorts and comparisons on the stored data are case sensitive on BLOBs and are not case sensitive in TEXT fields. You do not specify a length with BLOB or TEXT.

iv. **TINYBLOB or TINYTEXT** – A BLOB or TEXT column with a maximum length of 255 characters. You do not specify a length with TINYBLOB or TINYTEXT.

v. **MEDIUMBLOB or MEDIUMTEXT –** A BLOB or TEXT column with a maximum length of 16777215 characters. You do not specify a length with MEDIUMBLOB or MEDIUMTEXT.

vi. **LONGBLOB or LONGTEX**T – A BLOB or TEXT column with a maximum length of 4294967295 characters. You do not specify a length with LONGBLOB or LONGTEXT.

vii. **ENUM** – An enumeration, which is a fancy term for list. When defining an ENUM, you are creating a list of items from which the value must be selected (or it can be NULL). For example, if you wanted your field to contain "A" or "B" or "C", you would define your ENUM as ENUM ('A', 'B', 'C') and only those values (or NULL) could ever

populate that field.

MySQL – Create Tables [SAQ1]

READING TIME: 1 MIN

To begin with, the table creation command requires the following details –

i. Name of the table

ii. Name of the fields

iii. Definitions for each field

Syntax

Here is a generic SQL syntax to create a MySQL table –

```
CREATE TABLE table_name (column_name column_type);
```

Now, we will create the following table in the TUTORIALS database.

```
create table tutorials_tbl(
   tutorial_id INT NOT NULL AUTO_INCREMENT,
   tutorial_title VARCHAR(100) NOT NULL,
   tutorial_author VARCHAR(40) NOT NULL,
   submission_date DATE,
   PRIMARY KEY ( tutorial_id )
);
```

To drop tables from the command prompt, we need to execute the DROP TABLE SQL command at the mysql> prompt.

Example

The following program is an example which deletes the tutorials_tbl –

```
root@host# mysql -u root -p
Enter password:*******
mysql> use TUTORIALS;
Database changed
mysql> DROP TABLE tutorials_tbl
Query OK, 0 rows affected (0.8 sec)

mysql>
```

MySQL – Insert (1min)

READING TIME: 1 MIN

To insert data into a MySQL table, you would need to use the SQL INSERT INTO command. You can insert data into the MySQL table by using the mysql> prompt or by using any script like PHP.

Syntax

Here is a generic SQL syntax of INSERT INTO command to insert data into the MySQL table –

```
INSERT INTO table_name ( field1, field2,...fieldN )
VALUES
( value1, value2,...valueN );
```

To insert string data types, it is required to keep all the values into double or single quotes. For example “value”.

Inserting Data from the Command Prompt

To insert data from the command prompt, we will use SQL INSERT INTO command to insert data into MySQL table tutorials_tbl.

Example

The following example will create 3 records into tutorials_tbl table –

```
root@host# mysql -u root -p password;
Enter password:*******
mysql> use TUTORIALS;
Database changed

mysql> INSERT INTO tutorials_tbl
->(tutorial_title, tutorial_author, submission_date)
->VALUES
->("Learn PHP", "John Poul", NOW());
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO tutorials_tbl
->(tutorial_title, tutorial_author, submission_date)
->VALUES
->("Learn MySQL", "Abdul S", NOW());
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO tutorials_tbl
->(tutorial_title, tutorial_author, submission_date)
->VALUES
->("JAVA Tutorial", "Sanjay", '2007-05-06');
Query OK, 1 row affected (0.01 sec)
mysql>
```

Summary

In this section we introduced MySQL DBMS. We examined its installation, administration and some of its statements (Query). Creation and removal of database and tables where examined. Query statements such as insert, select, where and delete where also examined.

Self-Assessment Questions

1. State and explain the syntax for an insert, select and delete query statement is MySQL.

Tutor Marked Assignment

Create a database in MySQL.

Create a table in the database above.

Insert data into the table created above.

References

Elmasri R. and Navathe S. B., (2011). Fundamentals of Database Sys tems. Addison-Wesley.

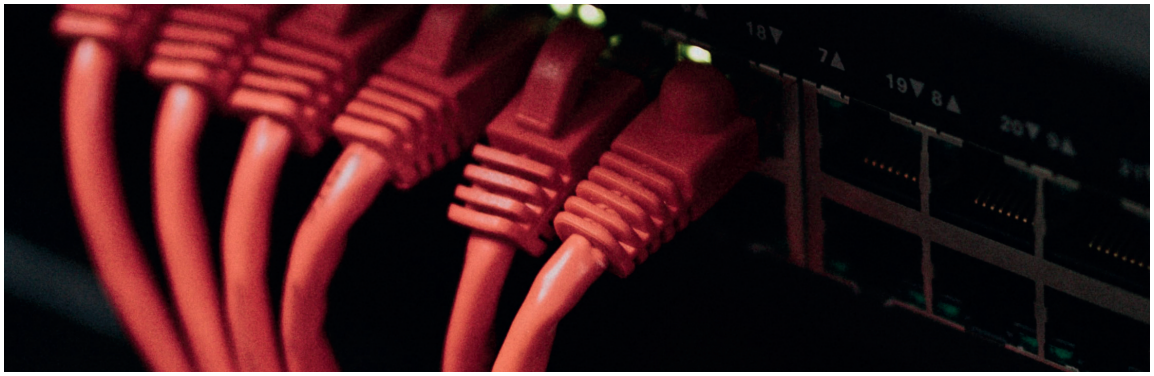
Gupta S. B. and Mittal A., (2004). Introduction to database manage ment systems. Pearson Education.

Ramakrishnan R. and Gherke J. (1996). Database Management Sys tem, second edition. McGraw-Hill Higher Education.

Further Reading

Silberschatz A., Korth H. F., and Sudarshan S. (2003). Database System Concepts. McGraw-Hill Education.

https://www.tutorialspoint.com/ms_sql /index.htm



Red cables in a port
Source: Unsplash by Nathan Dumlao

UNIT 4

SQL SERVER

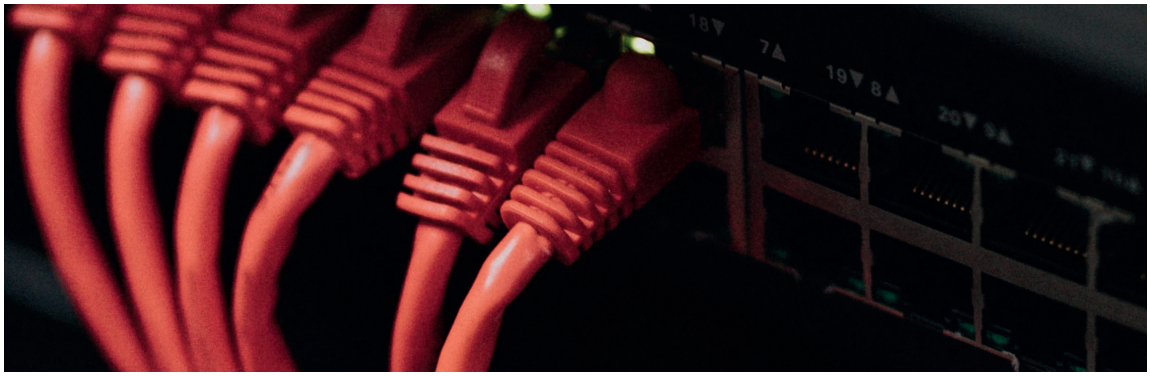
Introduction

Let me start by educating you that Microsoft SQL Server is a relational database management system (RDBMS) developed by Microsoft. This product is built for the basic function of storing retrieving data as required by other applications. It can be run either on the same computer or on another across a network. We will explain some basic and advanced concepts of SQL Server such as how to create and restore data, create login and backup, assign permissions, etc. Each topic is explained using examples for easy understanding.

Learning Outcomes

When you have studies this unit, you should be able to:

- Install SQL Server.
- Configure the server.
- Use the management studio and the command line to create a database, select a database and delete the database.



Red cables in a port
Source: Unsplash by Nathan Dumlao

Main Content

SQL Server – Installation (1min)

READING TIME: 1 MIN

Let me inform you that SQL Server supports two types of installation –

- Standalone
- Cluster based
- Installation Steps

Step1 – Download the Evaluation Edition from http://www.microsoft.com/download/en/details.aspx?id=29066

Step 2 – Double-click the "SQLFULL_x86_ENU_Install.exe" or "SQLFULL_x64_ENU_Install.exe", it will extract the required files for installation in the"SQLFULL_x86_ENU" or "SQLFULL_x86_ENU" folder respectively.

Step 3 – Click the "SQLFULL_x86_ENU" or "SQLFULL_x64_ENU_Install.exe" folder and double-click "SETUP" application.

For understanding, here we have used SQLFULL_x64_ENU_Install.exe software.

Step 4 – Once we click on 'setup' application, the following screen will open.

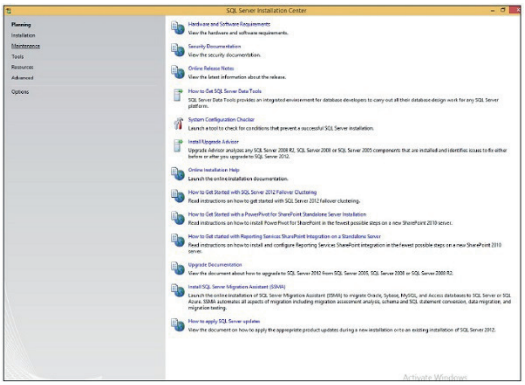


Figure 1: Installation

Step 5 – Click Installation which is on the left side of the above screen.

Figure 2: Server centre

Step 6 – Click the first option of the right side seen on the above screen. The following screen will open and follow the other installation process as you will be guided.

SQL Server - Management Studio [SAQ1]

READING TIME: 1 MIN

You should note that SQL Server Management Studio is a workstation component\ client tool that will be installed if we select workstation component in installation steps. This allows you to connect to and manage your SQL Server from a graphical interface instead of having to use the command line.

In order to connect to a remote instance of an SQL Server, you will need this or similar software. It is used by Administrators, Developers, Testers, etc.

The following methods are used to open SQL Server Management Studio.

First Method

Start → All Programs → MS SQL Server 2012 → SQL Server Management Studio

Second Method

Go to Run and type SQLWB (For 2005 Version) SSMS (For 2008 and Later Versions). Then click Enter.

SQL Server Management Studio will be open up as shown in the following snapshot in either of the above method.

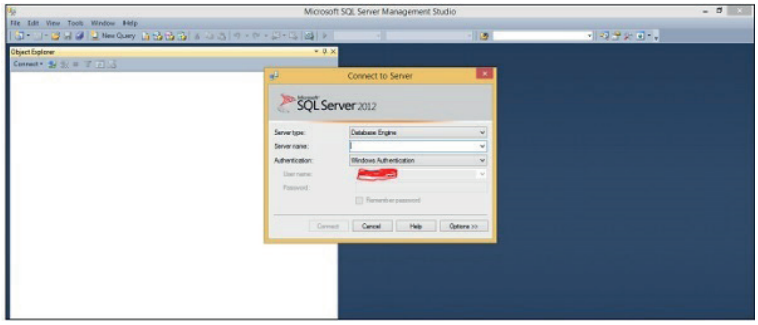


Figure 3: Server Login

SQL Server - Login Database

READING TIME: 1 MIN

Have it at the back of your mind that a login is a simple credential for accessing SQL Server. For example, you provide your username and password when logging on to Windows or even your e-mail account. This username and password builds up the credentials. Therefore, credentials are simply a username and a password.

SQL Server allows four types of logins –

i. A login based on Windows credentials.

ii. A login specific to SQL Server.

iii. A login mapped to a certificate.

iv. A login mapped to asymmetric key.

In this unit, we are interested in logins based on Windows Credentials and logins specific to SQL Server.

Logins based on Windows credentials allow you to log in to SQL Server using a Windows username and password. If you need to create your own credentials (username and password,) you can create a login specific to SQL Server.

To create, alter, or remove a SQL Server login, you can take one of two approaches:

i. Using SQL Server Management Studio.

ii. Using T-SQL statements.

Following methods are used to create Login:

Step 1 – After connecting to SQL Server Instance, expand logins folder as shown in the following snapshot

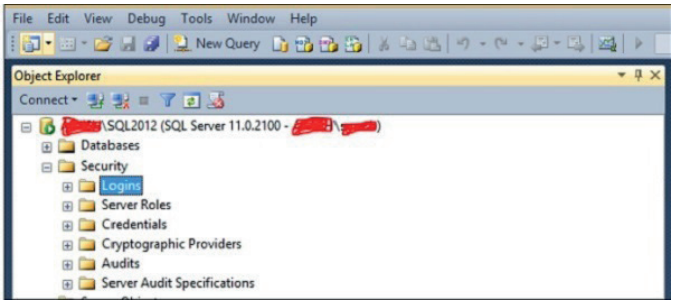


Figure 4: Logins

Step 2 – Right-click on Logins, then click Newlogin and the following screen will open.

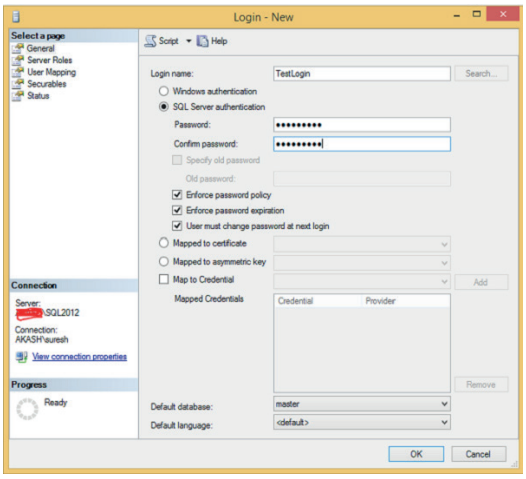


Figure 5: Authentication window

Step 3 – Fill the Login name, Password and Confirm password columns as shown in the above screen and then click OK.

Second Method – Using T-SQL Script

```
Create login yourloginname with password='yourpassword'
```

To create login name with TestLogin and password 'P@ssword' run below the following query.

```
Create login TestLogin with password='P@ssword'
```

SQL Server – Create Database

READING TIME: 1 MIN

It is cogent you cognise that database is a collection of objects such as table, view, stored procedure, function, trigger, etc.

In MS SQL Server, two types of databases are available.

i. System databases

ii. User Databases

System Databases

System databases are created automatically when we install MS SQL Server.

Following is a list of system databases –

- Master
- Model
- MSDB
- Tempdb
- Resource (Introduced in 2005 version)
- Distribution (It's for Replication feature only)

User Databases

User databases are created by users (Administrators, developers, and testers who have access to create databases).

Following methods are used to create user database.

Method 1 – Using T-SQL Script or Restore Database

Following is the basic syntax for creating database in MS SQL Server.

Create database <yourdatabasename>

OR

Restore Database <Your database name> from disk = '<Backup file location + file name>

Example

To create database called 'Testdb', run the following query.

Create database Testdb

OR

Restore database Testdb from disk = 'D:\Backup\Testdb_full_backup.bak'

Method 2 – Using SQL Server Management Studio

Connect to SQL Server instance and right-click on the databases folder. Click on new database and the following screen will appear.

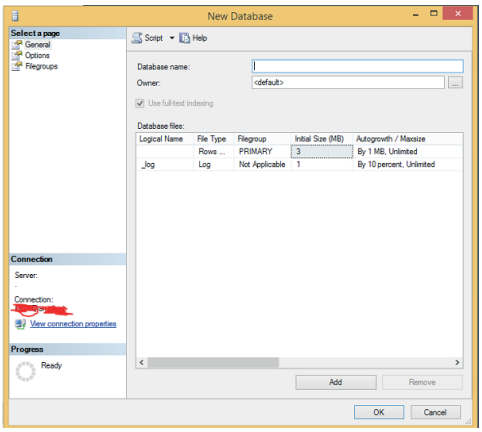


Figure 6: Selecting new database

Enter the database name field with your database name (example: to create database with the name 'Testdb') and click OK. Testdb database will be created as shown in the following snapshot.

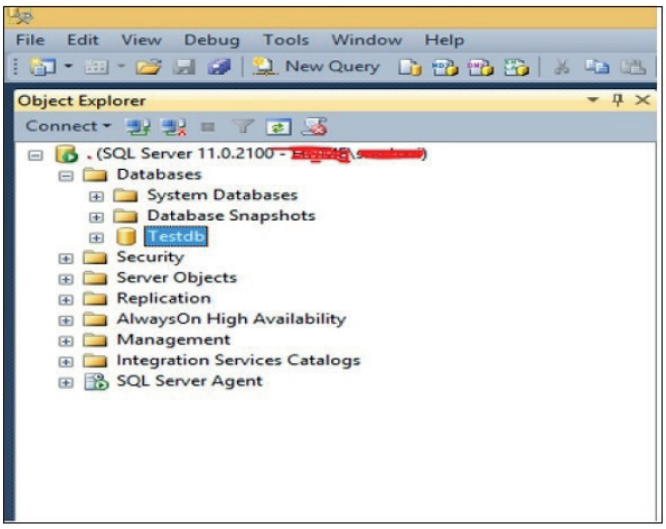


Figure 7: Object Explorer

SQL Server – Select Database

READING TIME: 1 MIN

Select your database based on your action before going ahead with any of the following methods.

Method 1 – Using SQL Server Management Studio

Example

To run a query to select backup history on database called 'msdb', select the msdb database as shown in the following snapshot.

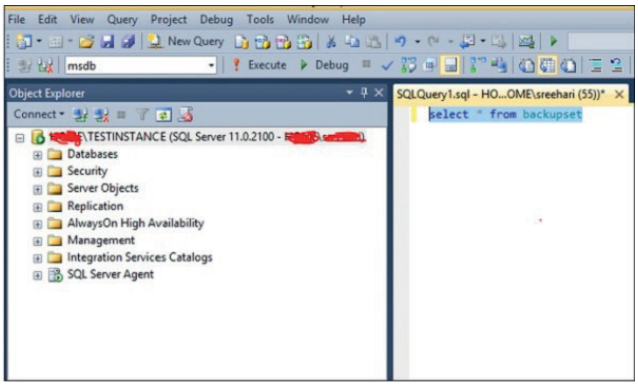


Figure 8: Running SQL Query

Method 2 – Using T-SQL Script

```
Use <your database name>
```

Example

To run your query to select backup history on database called 'msdb', select the msdb database by executing the following query.

```
Exec use msdb
```

The query will open msdb database. You can execute the following query to select backup history.

```
Select * from backupset
```

SQL Server – Drop Database

READING TIME: 1 MIN

To remove your database from MS SQL Server, use drop database command. Following two methods can be used for this purpose.

Method 1 – Using T-SQL Script

Following is the basic syntax for removing database from MS SQL Server.

```
Drop database <your database name>
```

Example

To remove database name 'Testdb', run the following query.

```
Drop database Testdb
```

Method 2 – Using MS SQL Server Management Studio

Connect to SQL Server and right-click the database you want to remove. Click delete command and the following screen will appear.

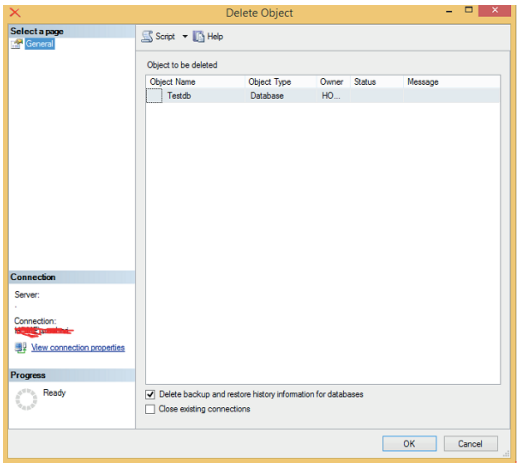


Figure 9: To delete an object

Click OK to remove the database (in this example, the name is Testdb as shown in the above screen) from MS SQL Server.

Summary

This unit introduced you to SQL Server. It examines the creation of database, log into a database, selection of database and the deletion of database using both the command line and management studio.

Self-Assessment Questions

1. What is an SQL Server management studio?

Tutor Marked Assignment

Create a database using the SQL Server management studio.

Login to the created database.

Delete the created database using the command line.

References

Elmasri R. and Navathe S. B., (2011). Fundamentals of Database Systems. Addison-Wesley.

Gupta S. B. and Mittal A., (2004). Introduction to database management systems. Pearson Education.

Ramakrishnan R. and Gherke J. (1996). Database Management System, second edition. McGraw-Hill Higher Education.

Further Reading

Silberschatz A., Korth H. F., and Sudarshan S. (2003). Database System Concepts. McGraw-Hill Education.

https://Tutorialspoint.com/oracle_sql/index.asp