KAPLAN HIGHER EDUCATION ACADEMY

KAPLAN HIGHER EDUCATION INSTITUTE

Study Guide

Diploma Programme

Introduction to Database Design and Development

v2.0

Message to Student

"I encourage you to be an active learner, questioning and extending what you know in order to become a contributing participant in the modern global economy."

Dear Kaplan Student,

Thank you for choosing Kaplan Singapore and entrusting your educational investment with us.

It is Kaplan Singapore's and the School of Diploma Studies' mission as your 'private education institution of choice' to commit to providing a fulfilling student journey and learning experience for your development. The underlying goal of Kaplan Singapore is to ensure that our students are able to respond the demands of the globalised economy set against a highly uncertain and complex environment.

In the 21st Century it is important that your educational experience provides with you with the skills and competencies on HOW to think and not just WHAT to think.

As such, the Kaplan Diplomas, which have a long history of being recognised as equivalent to the first year or more of undergraduate studies in high-quality international universities, aim to prepare you for work, further study, and lifelong learning.

Your role as a Kaplan student is to participate, think deeply on a topic and work with other students and your Lecturers to broaden your knowledge. I encourage you to be an active learner, questioning and extending what you know in order to become a contributing participant in the modern global economy.

However, only you can embody these attributes. You are the major determinant of your own academic success and as adult learners you will receive both the freedom and responsibility this entails. With that, I wish you well in your learning and look forward to seeing you succeed.

Krishna Rajulu
Academic Dean, Kaplan Singapore

Kaplan Desired Graduate Attributes

Through the reading of this module, Kaplan Singapore intends to:

- Instill in students the value of lifelong and self-directed learning by stimulating intellectual curiosity, creative and critical thinking and an awareness of cultural diversity;
- Assist students in developing professional attributes, ethical values, social skills and strategies that will nurture success in both their professional and personal lives;
- Foster integrity, commitment, responsibility and a sense of service to the community;
- Prepare students to meet the ever-changing needs of their communities both now and in the future; and
- Promote innovative and effective teaching.

Culminating from these institutional values and educational goals, Kaplan Singapore's Desired Graduate Attributes are:

Inquiry and criticality: Graduates will be able to critically collect, evaluate and apply information and data in order to make decisions in a wide variety of professional situations. This attribute is demonstrated when students:

- Undertake, evaluate and apply appropriate research, theories, concepts and tools to investigate problems and find solutions;
- Exercise critical thinking and independent judgement to assess situations and determine solutions; and
- Have an informed respect for the principles, methods, values and boundaries of their profession and the capacity to question these.

Ethicality and discernment: Graduates will be able to assess situations and respond in an ethically, socially and professionally responsible manner. This attributed is demonstrated when students:

- Act responsibly, ethically and with integrity in their profession;
- Hold personal values and beliefs and participate in the broad discussion of these values and beliefs while respecting the views of others;
- Understand the broad local and global economic, political, social and environmental systems and their impact as appropriate to their discipline and profession; and
- Acknowledge personal responsibility for their own judgments and behaviour

Ability to communicate well: Graduates will recognise the importance and value of communication in the learning and professional environment. This attributed is demonstrated when students:

- Create and present knowledge, arguments and ideas confidently and effectively using a variety of methods and technologies;
- Recognise the wide range of possible audiences for information and respond with communication strategies appropriate to those audiences; and
- Work collaboratively with people from diverse backgrounds and be aware of the different roles of team members and to function within that team.

Independent and reflective practitioner

- Graduates will be able to work independently and be self-directed learners with the capacity and motivation for continued professional learning and development; and
- They will be able to critically reflect on their own practice and evaluate and understand current capacity and further development needs

Embedded within the desired graduate attributes are the following skills:

- Conduct research.
- Analyse, organise and present data and information.
- Think and read critically.
- Make an oral presentation.
- Intellectual curiosity and awareness of culture and diversity.
- Develop professional ethos and practice that will foster success in career and life.
- Meet the ever changing needs of communities now and in the future.

Table of Contents

About this module

Modern organisations transact, manipulate, investigate and store increasingly large amounts of data. This also means that the database management system is arguably the most important solution for these organisations.

Database management systems are the tools used to manipulate raw data into meaningful information and store in a robust manner.

As the importance of database management system increased with significant developments in hardware capability, hardware capacity, and communications, including the emergence of the Internet, electronic commerce, business intelligence, mobile communications, grid computing, understanding and using database management systems is an essential skill required of anyone seeking to work in modern organisation with large quantities of data.

This module will introduce students to the basic concepts of databases, database modelling, the relational model, logical database design, Structured Query Language (SQL) programming language, transaction management, database security and database administration.

Students will demonstrate practical skills in data modelling, database design, and using SQL to create databases.

On completion of this module, students will be knowledgeable in database design and development to perform basic functional roles in their respective professions.

Module Learning Outcomes

Upon successful completion of this module, the student should be able to:

- Explain the importance of databases in information management;
- Describe the theoretical underpinning of the relational model and modern database design;
- Describe the relationship between database management systems and database applications used in modern organisations;
- Examine the concepts of database security, transaction management and database integrity;
- Create and manage data in a Database Management System using Structured Query Language;
- Explain the issues and approaches involved in the processing of concurrent database transactions;
- Explain the use of database management systems, and their administration in organisations and how these help archive strategic goals.

Overview of Learning Resources

Recommended reading:

Connelly, T., Begg, C., Database Systems:A Practical Approach to Design, Implementation, and Management, 6thedition, Pearson

Jeffrey A Hoffer;V Ramesh; Heikk:i Topi(2007), Modern databasemanagement, 8th edition, Pearson

Other sources:

See Proquest and Newslink databases linked to your Elearn LMS homepage. The National Library Board on North Bridge Road (databases are for Singaporean/PR only)

Scheme of Work

LESSON	TOPICS
1	01 Introduction to Databases • Introduction • Traditional File-Based Systems • Database Approach • Roles in the Database Environment • History of Database Management Systems • Advantages and Disadvantages of DBMSs
2	02 Database Modelling • Types of Data Models • Comparison of Data Models • The Three-Level ANSI-SPARC Architecture • Terminology of Relational Model • Properties of Valid Relation • Types of Keys • Integrity Constraints
3	03 Conceptual Modelling – Entity Relationship Model • Entity Types • Relationship Types • Attributes • Strong and Weak Entity Types • Attributes on Relationships • Problems with ER Model
4	04 Normalization • Anomalies • First Normal Form • Second Normal Form • Third Normal Form • Denormalization
5	05 Structured Query Language (SQL) Part I • Introduction to SQL • Writing SQL Commands • The ISO SQL Data Types • Integrity Enhancement Feature • DDL Commands
6	06 Structured Query Language (SQL) Part II • Create Simple Database • DML Commands • Aggregate functions
7	Assignment Consultation
8	07 Structured Query Language (SQL) Part III • Types of Join • Views • Discretionary Access Control

9	08 Transaction Management • Properties of Transactions • Locks • Two phase locking • Concurrency control
10	09 Database Administration and Security • DBA Roles and Responsibilities • Database Security • Countermeasures – Computer-Based Controls • DBMSs and Web Security
11	Assignment Consultation
12	10 Distributed DBMS • Centralized vs Distributed DBMS • Advantages and Disadvantages of Distributed DBMS • Fragmentation
13	Assignment Consultation (PT) Assignment Presentation (FT)
14	

Assessment Matters

Assessment Overview

Assessment 1: CA Quiz

Weightage: 20% (20 marks)
Date: To be confirmeed
Duration: 10 minutes per quiz
Test Format: 5 MCQs per topic

Assessment 2: Individual Assignment

Weightage: 80% (100 marks)
Date: Lesson Session 12 (FT) / Session 7 (PT)
Word Limit: Module Specific
Citation Format: APA
References: Module Specific

Important Policies

Penalties for Plagiarism

Plagiarism in any form is not tolerated by Kaplan Singapore. That said, direct quotations and general similarities of common terms and language mean the E-Learn LMS will often pick up every small similarity so the likelihood of a Turnitin Similarity report recording a result of 0% is unrealistic. After all, no technology is perfect and there is the need for some direct quotation (provided you reference using APA guidelines, of course) and to use commonly accepted terms and language.

By way of quality control, Kaplan Singapore has imposed an absolute cut-off of 25% in proprietary programmes (a zero grade for the assessment). In all cases, the lecturer is the true determinant of any result. To be certain you should always reference according to the APA style required by the institution.

TOP TIP:
The surest way to succeed is to ensure all work is correctly referenced. Keep a copy of the Kaplan Singapore Academic Works and APA Guide handy when you are typing your assignments and use it to guide you as to correct referencing, citation and other aspects of academic writing.

APA GUIDE: http://bit.ly/igd-apa

Penalties for late submissions

Kaplan Singapore prepares students for the realities of the workforce and further education by requiring students to meet deadlines and submit all work on time. As such, students are required to seek approval and penalties will be imposed on late assignment submissions in accordance with the table below and cited in the Programme Handbook:

No of days late	Penalty
1 – 5 days	10% deduction per day from the marks attained by students.
After 5 days	Assignments that are submitted more than 5 days after the due date will not be accepted and it will be deemed as "No Submission". Student will be required to re-module.

Source: Kaplan School of Diploma Studies Student Handbook (V4.5, 2017), Section 9

Assignments and Kaplan Learning Management System

Kaplan Singapore School of Diploma Studies requires you to submit Assignments through the Learning Management System (E-Learn LMS). When submitted, your assignment is checked for plagiarism by software called Turnitin linked to the E-Learn LMS. The software is intended to provide one more tool to improve the quality of academic writing and as such will be compulsory for use. It is important to note that this is merely one of many tools available to you and that final decisions about the quality of your work rest with your lecturer.

Assigment Submission: How to Use E-Learn LMS for Assignment Submission

1. You will be enrolled by the School of Diploma Studies Programme Management into the E-Learn LMS system only after your fee payment is confirmed.
2. You will be sent your USER NAME and PASSWORD via email.
3. Reset your password as prompted.
4. Enter the site at the following address: https://elearn-diploma.kaplan.com.sg
5. To submit assignment please refer to the LMS Manual

Please refer to your Student Handbook for more details on Penalties for Plagiarism, Misconduct, Examinations Rules and Regulations. Should you have any queries, please contact diploma.sg@kaplan.com

This page intentionally left blank

Study Guide

Topic 01: Introduction to Databases

Lesson Learning Outcomes

- Describe the difference between data and information.
- Explain the concept of a database and the different types of databases.
- Explain the importance of database design.
- Describe how modern database systems evolved.
- Explain the differences between a database system and a file system.
- Describe how a DBMS functions within the database system.

1.1 Data and Information

Data can be classified as raw facts or the building block of information. These are usually unprocessed information. Data can exist in a variety of forms such as numbers or text on pieces of paper and bytes stored in electronic memory, or as facts stored in a person's mind.

Metadata describes how and when and by whom a particular set of data was collected, and how the data is formatted.

Information can be classified as processed data. Information should be meaningful, relevant, accurate and timely. These are essential ingredients to good decision-making, which is the key to organisational success.

Data Management is a discipline that focuses on the proper generation, storage and retrieval of data.

1.2 What is a Database?

A collection of information organised in such a way that a computer programme can quickly select desired pieces of data
Traditional databases are organised as ***field****s*, ***records*****,** and ***file****s*.

A field is a single piece of information that has a specific meaning. A field is used to define and store data. A record is one complete set of fields describing a person, place or thing. A file is a collection of related records. To access information from a database, you need a ***database management system*** *(DBMS)*.

A good example would be telephone book, which is analogous to a file. It contains a list of records and each record consists of three fields namely name, address and telephone number

1.3 What is a Database Management System (DBMS)?

A ***DBMS*** is a collection of programmes (software) that enables you to store, modify, and extract information from a database.

There are many different types of DBMSs, ranging from small systems that run on personal computers to huge systems that run on mainframes. DBMSs' makes data management more effective. The end-user has as better access to more and better-managed data. It also allows integration view of organisation's data leading. Another advantage is its ability to

produce quick responses to ad hoc queries. The DBMS software interacts with the users' application programmes and the database. Typically, a DBMS provides the following facilities:

- It allows users to define the database, usually through a ***Data Definition Language*** (DDL). The DDL allows users to specify the data-type and structures, and the constraints on that data to be stored in the database.

- It allows users to insert, update, delete and retrieve data from the database, usually through a ***Data Manipulation Language*** (DML). Having a central repository for all data and data descriptions allows the DML to provide inquiry facilities to this data called a query language.

1.4 Types of Databases

There are several classifications of databases. The ***single-user* database** is the one, which supports one user at a time. The ***desktop* database** that is also a single-user database that runs on the desktop of a personal computer. The multi-user database is one that supports multiple users at the same time. The ***workgroup database*** is one that supports a small group of users or a single department. The ***enterprise database*** is a multi-user database that supports a large group of users or an entire organisation.

Another classification possible is by location. The ***centralised database*** is one that supports data located at a single site. The advantage of centralised database is the ability to exercise greater control over accessing and updating data. However, the disadvantage is its vulnerability to failure. A ***distributed database*** is a single logical database that is spread physically across computers in multiple locations.

Yet another type of classification is by its use. The ***transactional database*** is used to process transactions such as product or service sales, payments and supply purchases, which are routine and often, daily operational activities of an organisation. In contrast, a ***data-warehouse database*** focuses on storage of data that will be used for information generation for the purpose of decision-making

1.5 Database Design

Database design refers to the design of the database structure that will be used to store and manage data. It does NOT refer to the designing the DBMS software. A well- designed database will minimize the problem of ***data-redundancy***. A poorly designed database tends to generate errors that will ultimately lead to bad decisions, which is the main cause of organisation failure.

1.5.1 The File-based Systems

This is the traditional approach to information system design. It focuses on the data processing needs of individual departments in the organisation. Managing data through file systems is largely obsolete.

1.5.2 Manual Filing Systems and Computerization

Manual filing systems are usually made up of a collection of file folders kept in filing cabinet. The use of folders was based on data's expected use. It was suitable when the amounts of data to be managed were small and with few reporting requirements. It was very hard to manage in cases with large amounts of data that require lots of interaction and manipulation.

The conversion from a manual filing system to a computerized file system was technically challenging. Technical specialists were needed to perform these tasks. These staff members were called ***data processing specialists***. These specialists created file structures, wrote software, and designed application programmes. This resulted in a large variation of "customized" systems being created. The design of these computer files, were similar in design to manual files. These specialists created programmes such as, monthly summaries, analysis reports, etc. There were separate files for each department.

As systems grew larger and more complex, demand for programming skills also grew. The main activity of the DP department remained programming. The problem with file-based programmes was that they were written in third generation programming languages (3GL). These types of languages required the programmer to specify task and *how* it must be done

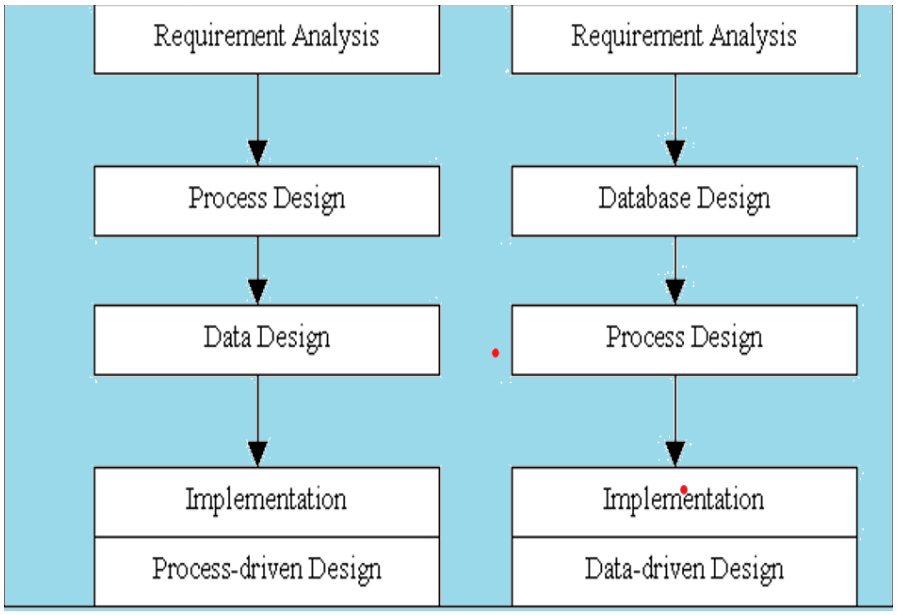
This type of programming was time-consuming that required high skill levels. The programmer must be familiar with physical file structure. New programmes had to be written every time a new type of query was needed. Any changes made to the file would require modifications to the programmes that use these files. This is known as ***structural dependence***.

Modifications often led to errors requiring additional time and effort for debugging. ***Data-redundancy*** was a common phenomenon in file-based systems. If not controlled, this can lead to ***data inconsistency***, lack of ***data integrity*** and ***data anomalies***.

1.6 Database vs. File System

The problems inherent in file-based systems make using a database system desirable. The file-based systems use many separate and unrelated files but a database system stores all data in a central repository. The ***repository*** is "single logical unit". This means that the actual physical data can be physically distributed among multiple data storage facilities but it appears as a single unit to the end-user. File Systems focuses on the data processing needs of individual departments in the organisation.

Database Systems emphasizes the integration and sharing of data across the organisation. The file-based system is known as ***process-driven system*** and a database system is known as ***data-driven system.***



1.7 Database Approach

This approach emphasizes the integration and sharing of data across the organisation. The data is independent of programmes. It supports multiple user views. It makes use of catalogue to store the database description (schema).

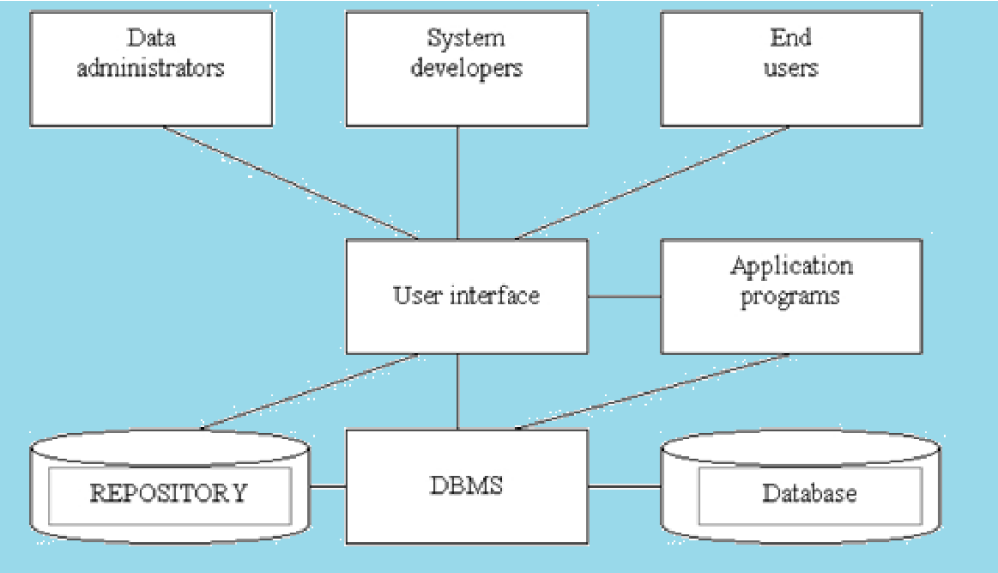
COMPONENTS OF THE DATABASE ENVIRONMENT

Repository

A centralised knowledge base containing all data definitions, screen and report formats and definitions of other organisations and system components

Database management system (DBMS)

Commercial software system used to create, maintain and provide controlled access to the database and repository.



Database

A shared collection of logically related data, designed to meet the information needs of multiple users in an organisation.

Application programmes

Computer programmes that are used to create and maintain the database and provide information to users.

User interface

Languages, menu and other facilities by which users interact with various system components.

Data administrators

Persons who are responsible for the overall information resources of an organisation.

System developers

Persons such as system analysts and programmers who design new application programmes.

End users

Persons throughout the organisation who add, delete and modify data in the database and who request or receive information from it.

1.8 DBMS FUNCTIONS

Data dictionary management – The DBMS stores the definitions of all data elements and their relationships in the data dictionary. The DBMS uses the dictionary to look up the required data component structure and relationships.

Data Storage Management – The DBMS creates and manages the complex structures required. It is also important database performance tuning.

Data transformation and presentation – The DBMS transforms the input data to conform to the data structures that are required to store the data. This maintains data independence.

Security Management – The DMBS creates a security system that enforces user security and data privacy within the database. Rules determine which users can access the database, which data items each user may access and which data operations the user may perform.

Multi-user access control – The DBMS manages and provides the environment so that multiple users can access the data.

Back-up recovery and management – The DBMS provides back-up and recovery procedures to ensure data safety and integrity.

Data Integrity management – The DBMS promotes and enforces rules to eliminate data-integrity problems, thus minimizing data-redundancy and maintaining data consistency.

Database access languages– The DBMS provides data access through a query language. The Query language has two components: the DDL and the DML. These have been explained above.

Database communications interfaces – Modern DBMS allow databases to accept end-user requests within a computer network (includes WANs such as the Internet.

References

Hoffer J.A., Ramesh V., and Topi H., *Essentials of Database Management (1st ed.)* Pearson

Database Solutions: A Step-by-step guide to building databases, by Thomas Connolly & Carolyn Begg. Pearson Addison Wesley

Topic 02: Database Modelling

Lesson Learning Outcomes

- Explain the purpose of data modelling
- Describe the approaches to Database Design
- Describe the Three-Level ANSI-SPARC Architecture
- Explain the terminology and structural concepts of the relational model
- Demonstrate an understanding of relational database principles and theory
- Explain the Relational Integrity Constraints

2.1 Types of Data Models

Data Model

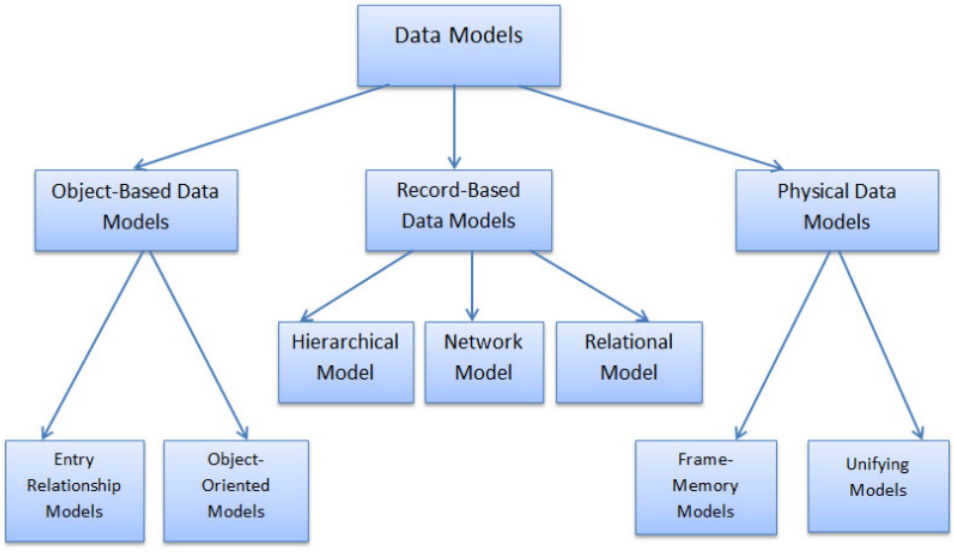
"A **data model** is an abstract model that organizes elements of data and standardizes how they relate to one another and to the properties of real-world entities." ("Data model", 2020)

It is an integrated collection of data concepts for describing, manipulating, the relationships between, and constraints of the data in an organization.

They fall into 3 broad categories:

- object-based,
- record-based, and
- physical data models.

Overview of a Data Models



Object-Based Data Models

Object-based data models use entities in the real world, attributes of each entity and their relationships.

The object-oriented data model not only include the state but also the actions that are associated with the objects; together known as behaviour. It encapsulates both state and behaviour.

- An entity is a distinct object. It may be a person, place, thing, concept, event in an organization that is to be represented in the database.
- An attribute describes the object such as Name, Address, Age and Phone number.
- And a relationship is an association between entities.

Record-Based Data Models

Principally, there are 3 types of record-based logical data model:

- the Relational Data model,
- the Network Data model, and
- the Hierarchical Data model.

(A) Relational Data Model

Data and relationships are represented as tables, each has a number of columns with a unique name. Figure 2.1 is a sample instance of a relational schema for part of the DreamHome case study.

Branch

branchNo	street	city	postCode
B005	22 Deer Rd	London	SW1 4EH
B007	16 Argyll St	Aberdeen	AB2 3SU
B003	163 Main St	Glasgow	G11 9QX
B004	32 Manse Rd	Bristol	BS99 1NZ
B002	56 Clover Dr	London	NW10 6EU

Staff

staffNo	fName	lName	position	sex	DOB	salary	branchNo
SL21	John	White	Manager	M	1-Oct-45	30000	B005
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000	B003
SG14	David	Ford	Supervisor	M	24-Mar-58	18000	B003
SA9	Mary	Howe	Assistant	F	19-Feb-70	9000	B007
SG5	Susan	Brand	Manager	F	3-Jun-40	24000	B003
SL41	Julie	Lee	Assistant	F	13-Jun-65	9000	B005

Figure 2.1 A sample instance of a Relational Schema (Connolly T., Begg C., 2015, p. 95)

Users state what information they want and let the database management system software store and retrieve the answering queries as most relational databases use the SQL data definition and query language.

(B) Network data model

In Network model, data is represented as collections of records where objects and relationships are presented graphically. Object types are nodes and relationship types are arcs. Figure 2.2 shows an instance of a network schema.

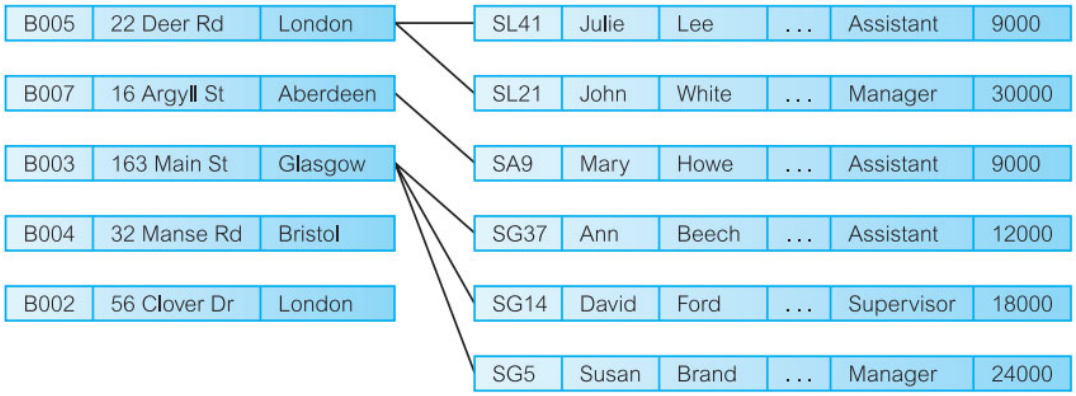
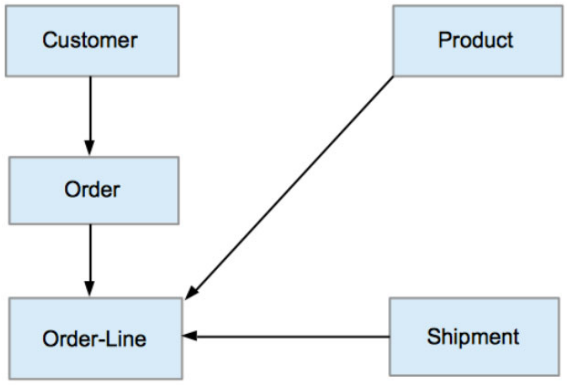


Figure 2.2 A sample instance of a Network Schema (Connolly T., Begg C., 2015, p. 96)



Bachman diagram of a simple Network Database
("Network model", 2020)

(C) Hierarchical data model

A Hierarchical database model is a data model in which the data are organized into a tree-like structure. The data are stored as records which are connected to one another through links. Hierarchical model allows a node to have only one parent. Figure 2.3 shows an instance of a hierarchical schema.

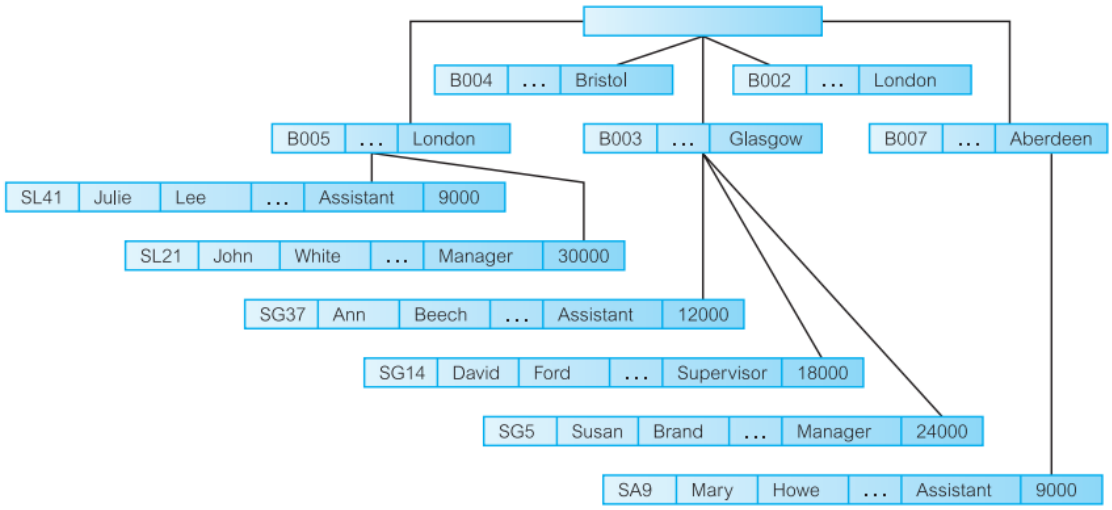


Figure 2.3 A sample instance of a Hierarchical Schema

In summary, Record-based (logical) data models are used to specify the overall structure of the database and a higher-level description of the implementation. Their main drawback is that they do not provide adequate facilities for explicitly specifying constraints on the data, whereas the object-based data models lack the means of logical structure specification but provide more semantic substance by allowing the user to specify constraints on the data.

The hierarchical database model requires each child record has only one parent, whereas each parent record can have one or more child records.

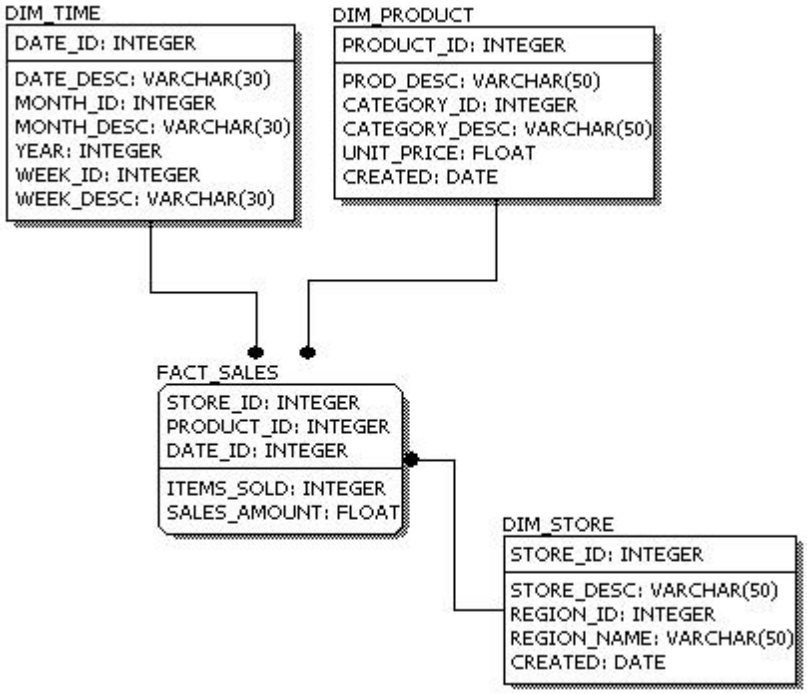
Physical Data Models

Physical data models describe how data is stored in the computer such as record structures, record orderings, and access paths.

Features of a physical data model include:

- Specification all tables and columns.
- Foreign keys are used to identify relationships between tables.
- De-normalization may occur based on user requirements.
- Physical considerations may cause the physical data model to be quite different from the logical data model.
- Physical data model will be different for different RDBMS. For example, data type for a column may be different between MySQL and SQL Server.
- The steps for physical data model design are as follows:
- Convert entities into tables.
- Convert relationships into foreign keys.
- Convert attributes into columns.
- Modify the physical data model based on physical constraints / requirements.
- The figure below is an example of a physical data model.

Physical Data Model



("Physical Data Model", 2020)

2.2 Comparison of Data Models

During the life-cycle of a model, you may make changes to the database and may want to compare the differences in the objects and properties, logical and physical level.

Comparison between Hierarchical model, Network model and Relational model.

Hierarchical Data Model	Network Data Model	Relational Data Model
1. Relationship between records is of the parent child type.	1. Relationship between records is expressed in the form of pointers or links.	1.Relationship between records is represented by a relation that contains a key for each record involved in the relationship.
2. Many to many relationship cannot be expressed in this model	2. Many to many relationship can also be implemented in this model	2. Many to many relationship can be easily implemented.
3. It is a simple, straightforward and natural method of implementing record relationships.	3. Record relationship implementation is quite complex due to the use of pointers.	3. Relationship implementation is very easy through the use of a key or composite key field.
4. This type of model is useful only when there is some hierarchical character in the database.	4. Network model is useful for representing such records which have many to many relationships.	4. Relational model is useful for representing most of the real world objects and relationships among them.
5. Searching for a record is very difficult since one can retrieve a child only after going through its parent record.	5. Searching for a record is easy since there are multiple access paths to a data element.	5. A unique, indexed key field is used to search for a data element.
6. In Hierarchical model record relations are physical.	6.In Network model record relations are physical.	6. Relational model does not maintain physical connection among records, data is organized logically in the form of rows and columns and stored in table.
7. During updation or deletion process, chances of data inconsistency is involved.	7. No problem of inconsistency exists in Network model.	7. Data integrity maintaining methods like Normalization process are adopted for consistency.

("Comparison Between Hierarchical model Network model Relational model", 2020)

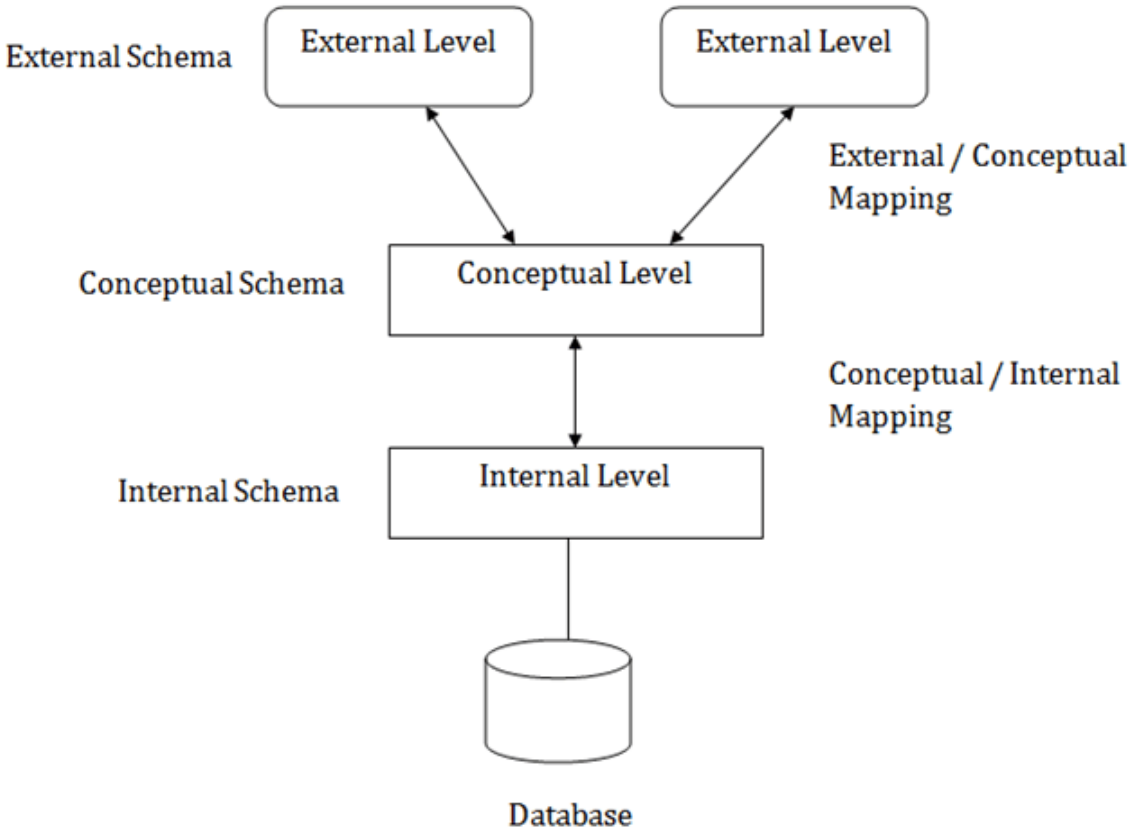
2.3 The Three-Level ANSI-SPARC Architecture

The three schema architecture is also called ANSI/SPARC architecture or three-level architecture comprising of:

- an external,
- a conceptual, and
- an internal level.

It describes the structure of a specific database system and the main objective is to separate the user applications and physical database. So that Users perceive the data at *external* level. DBMS and the operating system perceive the data at *internal* level, and data is stored using the data structures and file organizations. Thus, user's interaction with the database should be independent of storage considerations.

The Three-schema architecture



("DBMS Three schema Architecture - javatpoint", 2020)

1. External Level

- At the external level, a database contains several schemas that sometimes called as subschema. It is used to describe the different view of the database.
- It is also known as view external schema.
- Each view schema describes the database part that a particular user group view or is interested in and hides the remaining database from that user group.
- The view schema describes the end user interaction with database systems.
- It includes entities, attributes, and relationships in the “real world”.

2. Conceptual Level

- The conceptual schema describes the design of a database and is also known as logical level.
- It describes the structure of the whole database.
- It describes what data are to be stored in the database and what relationship exists among those data.
- Internal details such as an implementation of the data structure are hidden. Programmers and database administrators work at this level.
- It is a complete view of the data which includes:
    - all entities, their attributes, and their relationships;
    - the constraints on the data;
    - semantic information about the data;
    - security and integrity information

3. Internal Level

- The internal level has an internal schema which describes the physical storage structure of the database.
- It is also known as a physical schema.
- It uses the physical data model. It define how data will be stored in a block.
- The physical level is used to describe complex low-level data structures in detail.
- Is concerned with:
    - storage space allocation for data and indexes;
    - record descriptions for storage;
    - data compression and data encryption techniques

2.4 Terminology of Relational Model

In this section we explain some of the most important relational database terminology and structural concepts of the relational model.

Tables / Relation
A table has rows and columns. Rows represent records and columns represent attributes.

Attribute
Attributes are the named column, the properties that define a relation. e.g., Empl**_No.**

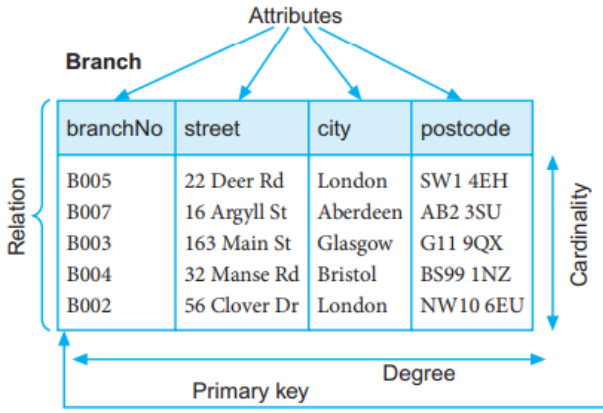
Domain
A domain is the set of allowable values for an attribute. For example, an integer number data type field can only allow whole numbers to be entered and not others.

Tuple
A single row of a table is known as tuple which contains a single record.

Degree
The number of attributes in the table

Cardinality
The total number of rows (tuple) present in the table.

Relational database
It consists of relations that are normalized and structured. See Figure 2.4



Staff

staffNo	fName	lName	position	sex	DOB	salary	branchNo
SL21	John	White	Manager	M	1-Oct-45	30000	B005
SG37	Ann	Beech	Assistant	F	10-Nov-60	12000	B003
SG14	David	Ford	Supervisor	M	24-Mar-58	18000	B003
SA9	Mary	Howe	Assistant	F	19-Feb-70	9000	B007
SG5	Susan	Brand	Manager	F	3-Jun-40	24000	B003
SL41	Julie	Lee	Assistant	F	13-Jun-65	9000	B005

Relation

Attribute	Domain Name	Meaning	Domain Definition
branchNo	BranchNumbers	The set of all possible branch numbers	character: size 4, range B001–B999
street	StreetNames	The set of all street names in Britain	character: size 25
city	CityNames	The set of all city names in Britain	character: size 15
postcode	Postcodes	The set of all postcodes in Britain	character: size 8
sex	Sex	The sex of a person	character: size 1, value M or F
DOB	DatesOfBirth	Possible values of staff birth dates	date, range from 1-Jan-20, format dd-mmm-yy
salary	Salaries	Possible values of staff salaries	monetary: 7 digits, range 6000.00–40000.00

Figure 2.4 Branch and Staff (Connolly T., Begg C., 2015, p. 153)

2.5 Properties of Valid Relation

A relation need to have the following properties:

- the relation has a name that is distinct from all other relation names in the relational schema;
- each cell of the relation contains exactly one atomic (single) value;
- each attribute has a distinct name;
- the values of an attribute are all from the same domain;
- each tuple is distinct; there are no duplicate tuples;
- the order of attributes has no significance;
- the order of tuples has no significance, theoretically.
  (However, in practice, the order may affect the efficiency of accessing tuples.)

(Connolly T., Begg C., 2015, p. 156)

2.6 Types of Keys

Since no duplicate tuples are in a relation. To identify one or more attributes, we need to use relational keys which many include:

Superkey

- The set of attributes that uniquely identifies a tuple within a relation is known as Super Key. For Example, STUD_NO, STUD_NAME.

Candidate key

- The minimal set of attribute which can uniquely identify a tuple is known as candidate key and there may be several candidate keys for a relation.

Primary key

- There may be more than one candidate key in a relation. Of which, one can be chosen as the primary key. The candidate keys that are not selected to be the primary key are called alternate keys.

Alternate Key

- The candidate key other than the primary key is called an alternate key.

Foreign key

- If an attribute can only take the values which are present as values of some other attribute, it will refer to as a foreign key.
- A foreign key is a field that matches the primary key column of another table. Foreign keys need not have unique values in the referencing relation.

2.7 Integrity Constraints

Integrity constraints refer to conditions which must be present for a valid relation. Any violation to the constraints and the operation will fail.

Constraints on the Relational Database Management System (RDMS) can be divided into 3 main categories:

- Domain constraints
- Key constraints
- Referential integrity constraints

Domain Constraints

Domain constraints can be violated if an attribute value is not of the appropriate data type.

Standard data types include integers, real numbers, characters, Booleans, variable length strings, etc.

Key Constraints

An attribute that can uniquely identify a tuple in a relation is called the key and in a database relation, no attribute of a primary key can be null.

A key has two properties:

- It should be unique for all tuples.
- It can't have NULL values.

Referential integrity constraints

Referential integrity constraints work on the concept of Foreign Keys where it is referred to in other relationships. It happens where relation refers to a key attribute of a different or same relation.

Review Questions

1. Explain the concept of database schema and discuss the three types of schema in a database.

2. What is a data model? Discuss the main types of data model.

3. Discuss the function and importance of conceptual modelling.

4. Define the term “database integrity”.

5. Discuss each of the following concepts in the context of the relational data model:
   a. relation
   b. attribute
   c. domain

6. Discuss the properties of a relation.

7. Discuss the differences between the candidate keys and the primary key of a relation. Explain what is meant by a foreign key. How do foreign keys of relations relate to candidate keys? Give examples to illustrate your answer.

Exercises

1. A database approach uses different data models. Common database models include the relational model, the network model and the hierarchical model. Which data model should be chosen under which circumstances and why?

2. Identify the foreign keys in this schema. Explain how the entity and referential integrity rules apply to these relations.
   The following tables form part of a database held in a relational DBMS:
   a. Hotel (hotelNo, hotelName, city)
   b. Room (roomNo, hotelNo, type, price)
   c. Booking (hotelNo, guestNo, dateFrom, dateTo, roomNo)
   d. Guest (guestNo, guestName, guestAddress)
      - Where Hotel contains hotel details and hotelNo is the primary key;
      - Room contains room details for each hotel and (roomNo, hoteINo) forms the primary key;
      - Booking contains details of bookings and (hoteINo, guestNo, dateFrom) forms the primary key;
      - Guest contains guest details and guestNo is the primary key.

3. Analyze the RDBMSs that you are currently using. Determine the support the system provides for primary keys, alternate keys, foreign keys, relational integrity, and views.

(Connolly T., Begg C., 2015, p. 104, 166)

Reference List

Comparison Between Hierarchical model Network model Relational model. (2020). Retrieved 19 October 2020, from https://webeduclick.com/comparison-between-hierarchical-model-network-model-and-relational-model/

Data model. (2020). Retrieved 19 October 2020, from https://en.wikipedia.org/wiki/Data_model

DBMS Three schema Architecture - javatpoint. (2020). Retrieved 19 October 2020, from https://www.javatpoint.com/dbms-three-schema-architecture

Network model. (2020). Retrieved 19 October 2020, from https://en.wikipedia.org/wiki/Network_model

Physical Data Model. (2020). Retrieved 19 October 2020, from https://www.1keydata.com/datawarehousing/physical-data-model.html

Topic 03: Conceptual Modelling – Entity Relationship Model

Lesson Learning Outcomes

- Explain the purpose and importance of conceptual modelling
- Describe how Entity-Relationship modelling is use in database design
- Demonstrate practical skills in data modelling using entity-relationship modelling (ER-Model)

"A conceptual model's primary objective is to convey the fundamental principles and basic functionality of the system which it represents. Also, a conceptual model must be developed in such a way as to provide an easily understood system interpretation for the model's users." (Conceptual model, 2020)

In an **Entity Relationship Model (ER) diagram**, it has three main components:

1. Entity
2. Relationship
3. Attribute

3.1 Entity Types

An *entity* type represents a group of "objects" of same properties having independent existence.

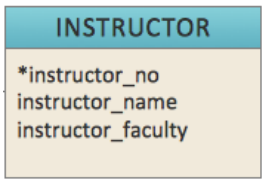
An entity can be:
- An object with physical/real existence (e.g., table, student, house)
- An object with conceptual/abstract existence (e.g., job, course, position)

Entities are nouns and can be person, place, object, event or a concept which must have attribute(s), and a unique key. Every entity is made up of some 'attributes' which represent that entity.

Examples of entities:
- Person: Employee, Student, Patient
- Place: Store, Building
- Object: Machine, product, and Car
- Event: Sale, Registration, Renewal
- Concept: Account, Course

(ER Diagram Tutorial in DBMS (with Example), 2020)

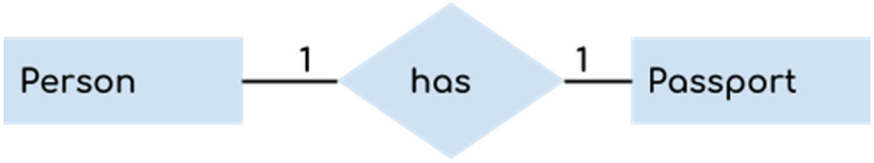
An entity is shown as a rectangle, with its name on top and its attributes listed in the body of the entity shape.


INSTRUCTOR
*instructor_no
instructor_name
instructor_faculty

3.2 Relationship Types

Relationship is an association among two or more entities. It is a connection that holds the tables together.

For example, a person has only one passport and a passport is given to one person.



(Entity Relationship Diagram - ER Diagram in DBMS, 2020)

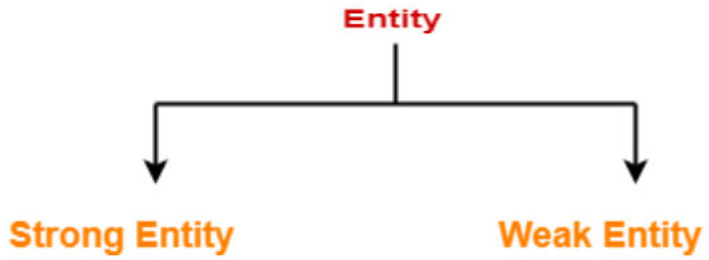
3.3 Attributes

Attributes are description of the entities. For example, for entity student, the attributes can be first name, last name, email, address and phone numbers.

Table below shows the different type of attributes.

Types of Attributes	Description
Simple attribute	Simple attributes can't be divided any further. For example, a student's contact number. It is also called an atomic value.
Composite attribute	It is possible to break down composite attribute. For example, a student's full name may be further divided into first name, second name, and last name.
Derived attribute	This type of attribute does not include in the physical database. However, their values are derived from other attributes present in the database. For example, age should not be stored directly. Instead, it should be derived from the DOB of that employee.
Multivalued attribute	Multivalued attributes can have more than one values. For example, a student can have more than one mobile number, email address, etc.

3.4 Strong and Weak Entity Types



Within the entity type, in can be class as *strong* and *weak*.

A **strong entity type** if its existence does not depend upon the existence of another entity type. It is independent to any other entity in the schema and strong entity always has a primary key. Relationship between two strong entities is represented by a diamond.

For example, we can identify each staff using the Staff_No attribute, which is also the primary key.

Strong entity

Weak entity is dependent on strong entity. It cannot be identified by its own attributes alone. It needs a foreign key to relate it to a strong entity. A weak entity is represented by double rectangle. Relationship between a strong entity and a weak entity is represented by double diamond.
For example, a voucher which may be complemented by a set of support documents such as bills, detail and date of expenses, issuing parties and others. Such supporting documents are weak entity type.

Weak entity

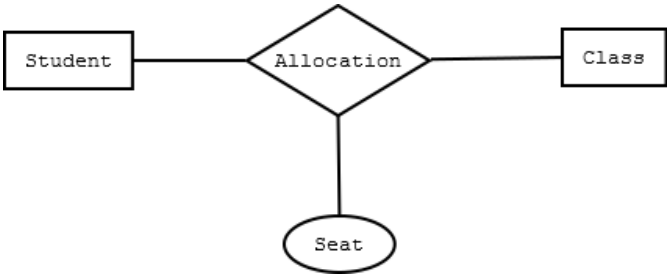
A Comparison of Strong and Weak Entity Set

Strong Entity Set	Weak Entity Set
Strong entity set always has a primary key.	It does not have enough attributes to build a primary key.
It is represented by a rectangle symbol.	It is represented by a double rectangle symbol.
It contains a Primary key represented by the underline symbol.	It contains a Partial Key which is represented by a dashed underline symbol.
The member of a strong entity set is called as dominant entity set.	The member of a weak entity set called as a subordinate entity set.
Primary Key is one of its attributes which helps to identify its member.	In a weak entity set, it is a combination of primary key and partial key of the strong entity set.
In the ER diagram the relationship between two strong entity set shown by using a diamond symbol.	The relationship between one strong and a weak entity set shown by using the double diamond symbol.
The connecting line of the strong entity set with the relationship is single.	The line connecting the weak entity set for identifying relationship is double.

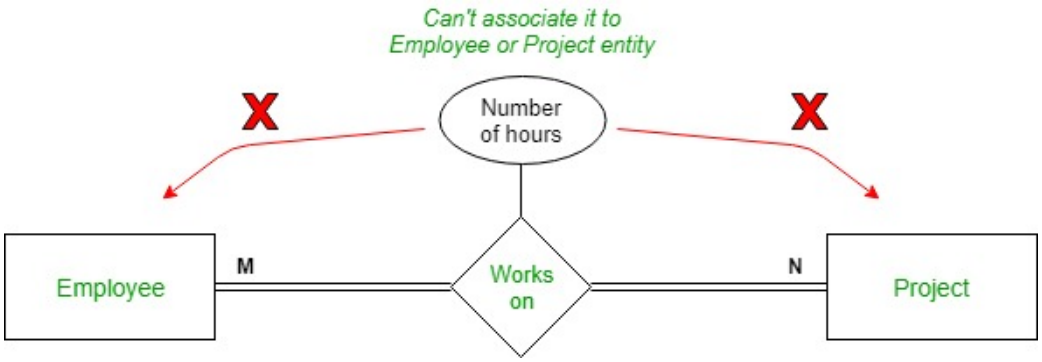
3.5 Attributes on Relationships

In many ER diagrams, attributes are usually assigned to the relations. However, attributes can also be assigned to relationships. Attributes on relationships allow you to record facts about the relationship as opposed to one of the entities that make up the relationship.

Examples 1:
A student's class allocation may have an assigned seat



Example 2:
An employee can work on many projects simultaneously and each project can have many employees working on it. So assigning the *Number_of_Working_hours* to the employee will not work as to which project's working hours because a single employee can work on multiple projects. Similarily, same with the *project* entity. Hence, we are forced to assign the *Number_of_Working_hours* attribute to the relationship.



(Attributes to Relationships in ER Model - GeeksforGeeks, 2020)

3.6 Problems with ER Models

E-R model can result in some problems, called connection traps, these may be due to limitations in the way they are related and a misinterpretation of the meaning of certain relationships. There are mainly two types of connection traps:

Fan Traps
These occur when a model represents a relationship between entity types but the pathway between certain entity occurrences is ambiguous. It may exist where two or more one-to-many (1:N) relationships fan out from the same entity

Chasm Traps
These occur when a model suggests the existence of a relationship between entity types, but the pathway does not exist between certain entity occurrences.
It occurs when a relationship set with partial participation, which forms part of the pathway between entities that are related.

References

beginnersbook.com. 2020. Entity Relationship Diagram - ER Diagram In DBMS. [online] Available at: <https://beginnersbook.com/2015/04/e-r-model-in-dbms/> [Accessed 19 October 2020].

diagrams, W. and Prasanna, G., 2020. Why Attributes Are In Relation In ER Diagrams. [online] Stack Overflow. Available at: <https://stackoverflow.com/questions/36245153/why-attributes-are-in-relation-in-er-diagrams> [Accessed 19 October 2020].

Topic 04: Normalization

Lesson Learning Outcomes

- Describe the importance of Normalization and understand normalization rule
- Explain different types of Normalization
- Explain the concept of Denormalization
- List the advantages and disadvantages of Denormalization

The word “normalization” used in statistics would mean the process of separating relations into well-structured relations or a process of reorganizing data in a database that allow users to insert, delete, and update tuples without introducing unwanted database. Fundamentally, normalization has to meet two basic requirements:

1. No redundancy of data as all data is stored in only one place.
2. Data dependencies are logical as all related data items are stored together.

(What is Normalization? - Definition from Techopedia, 2020)

4.1 Anomalies

If a database is poorly planned and designed, de-normalized and inconsistent, it will create problems when trying to insert, delete or modify/update the tables in the database.
This causes **anomalies**, which make handling the data increasingly difficult as the database grows. To make the data consistent once an anomaly occurs can become a costly and challenging task.
Basically, there are 3 types of problems that can occur in databases:

- **Insertion anomaly:** The database has been created in such a way that required data cannot be added unless another piece of unavailable data is also added. For example, a hospital database that cannot store the details of a new member until that member has been seen by a doctor.

- **Deletion anomaly:** The legitimate deletion of a record of data can cause the deletion of some required data. For example, deleting some of the patient’s details can remove all the details of the patient from the hospital database.

- **Modification/Update anomaly:** Incorrect data may have to be changed, which could involve many records having to be changed, leading to the possibility of some changes being made incorrectly.
  (How to get rid of anomalies | Database design concepts | Siyavula, 2020)

Normalization

In database, problems might occur due to anomalies, as such, normalization is significant and need to be addressed out front.

Normalisation is a systematic approach of decomposing tables to eliminate data redundancy; eliminates unnecessary duplication(s). Without normalisation, database systems can be inaccurate, slow, and inefficient.
Firstly, we need to understand normalization rules. They are divided into the following forms:

- First Normal Form
- Second Normal Form
- Third Normal Form

In summary of the rules required in each normal form (1st, 2nd & 3rd), we shall addressed them in brief before looking into an example.

4.2 First Normal Form (1NF)

1. It should only have single (atomic) valued attributes/columns.
2. Values stored in a column should be of the same domain
3. All the columns in a table should have unique names.
4. And the order in which data is stored, does not matter.

4.3 Second Normal Form (2NF)

- It should be in the First Normal form.
- And, it should not have Partial Dependency.

4.4 Third Normal Form (3NF)

1. It is in the Second Normal form.
2. And, it doesn't have Transitive Dependency.

(1NF, 2NF, 3NF and BCNF in Database Normalization | Studytonight, 2020)

An Example:

Without any normalization, all information is stored in one table as shown below.

FULL NAMES	PHYSICAL ADDRESS	MOVIES RENTED	SALUTATION
Janet Jones	First Street Plot No 4	Pirates of the Caribbean, Clash of the Titans	Ms.
Robert Phil	3rd Street 34	Forgetting Sarah Marshal, Daddy's Little Girls	Mr.
Robert Phil	5th Avenue	Clash of the Titans	Mr.

First Normal Form (1NF)

- Each table cell should contain a single value.
- Each record needs to be unique.

FULL NAMES	PHYSICAL ADDRESS	MOVIES RENTED	SALUTATION
Janet Jones	First Street Plot No 4	Pirates of the Caribbean	Ms.
Janet Jones	First Street Plot No 4	Clash of the Titans	Ms.
Robert Phil	3rd Street 34	Forgetting Sarah Marshal	Mr.
Robert Phil	3rd Street 34	Daddy's Little Girls	Mr.
Robert Phil	5th Avenue	Clash of the Titans	Mr.

Second Normal Form (2NF)

- Be in 1NF
- Single Column Primary Key

MEMBERSHIP ID	FULL NAMES	PHYSICAL ADDRESS	SALUTATION
1	Janet Jones	First Street Plot No 4	Ms.
2	Robert Phil	3rd Street 34	Mr.
3	Robert Phil	5th Avenue	Mr.

Table 1

MEMBERSHIP ID	MOVIES RENTED
1	Pirates of the Caribbean
1	Clash of the Titans
2	Forgetting Sarah Marshal
2	Daddy's Little Girls
3	Clash of the Titans

Table2

What happen next?

- 1NF table is divided into 2 tables; Table 1 and Table2.
- Table 1 contains member information.
- Table 2 contains information on movies rented.

- A new column called Membership_ID is being introduced which is the primary key for Table 1.

- In Table 2, Membership_ID is the Foreign Key.

Third Normal Form (3NF)

- Be in 2NF
- Has no transitive functional dependencies

MEMBERSHIP ID	FULL NAMES	PHYSICAL ADDRESS	SALUTATION ID
1	Janet Jones	First Street Plot No 4	2
2	Robert Phil	3rd Street 34	1
3	Robert Phil	5th Avenue	1

Table 1

MEMBERSHIP ID	MOVIES RENTED
1	Pirates of the Caribbean
1	Clash of the Titans
2	Forgetting Sarah Marshal
2	Daddy's Little Girls
3	Clash of the Titans

Table 2

SALUTATION ID	SALUTATION
1	Mr.
2	Ms.
3	Mrs.
4	Dr.

Table 3

What happen next?

- A new table was created to store Salutations.
- There are no transitive functional dependencies, and hence the table is in 3NF.
- In Table 3 Salutation ID is primary key, and in Table 1 Salutation ID is foreign to primary key in Table 3.

(What is Normalization? 1NF, 2NF, 3NF, BCNF Database Example, 2020)

4.5 Denormalization

Denormalization is a database optimization technique in which we deliberately add redundant data to one or more tables. This process will help to avoid multiple table joins. *Denormalization is an optimization technique that is applied after doing normalization.*
(S, 2020)

Advantages of Denormalization

- Data retrieval will be faster
- Avoid multiple table joins
- Query will be easy to read as it will refer fewer tables

Disadvantages of Denormalization

- Extra storage space
- Update and insert operations are more expensive
- Data redundancy
- Potential data anomalies

Review Questions

1. Describe the purpose of normalizing data and its importance.

2. How would you fix the below table to reach 1N?

ACCOUNTNO	NAME	SUBJECT
542	Tafadzwa	Eng, Maths
543	Sipho	Science, ICT
544	Gift	ICT, Maths
545	Naledi	English, Science

(How to get rid of anomalies | Database design concepts | Siyavula, 2020)

3. The second normal form (2NF) is realized by removing partial dependencies from 1NF relations. Briefly describe the term “partial dependency.”

4. Describe the concept of transitive dependency and describe how this concept relates to 3NF. Provide an example to illustrate your answer

Exercises

The table shown in Figure 14.19 lists sample dentist/patient appointment data. A patient is given an appointment at a specific time and date with a dentist located at a particular surgery. On each day of patient appointments, a dentist is allocated to a specific surgery for that day.

a. The table shown in Figure 14.19 is susceptible to update anomalies. Provide examples of insertion, deletion, and update anomalies.

b. Identify the functional dependencies represented by the attributes shown in the table of Figure 14.19. State any assumptions you make about the data and the attributes shown in this table.

c. Describe and illustrate the process of normalizing the table shown in Figure 14.19 to 3NF relations. Identify the primary, alternate, and foreign keys in your 3NF relations.

staffNo	dentistName	patNo	patName	appointment date	time	surgeryNo
S1011	Tony Smith	P100	Gillian White	12-Sep-13	10.00	S15
S1011	Tony Smith	P105	Jill Bell	12-Sep-13	12.00	S15
S1024	Helen Pearson	P108	Ian MacKay	12-Sep-13	10.00	S10
S1024	Helen Pearson	P108	Ian MacKay	14-Sep-13	14.00	S10
S1032	Robin Plevin	P105	Jill Bell	14-Sep-13	16.30	S15
S1032	Robin Plevin	P110	John Walker	15-Sep-13	18.00	S13

Figure 14.19 Table displaying sample dentist/patient appointment data.

(Connolly T., Begg C., 2015, p. 476, 477)

References

Guru99.com. 2020. What Is Normalization? 1NF, 2NF, 3NF, BCNF Database Example. [online] Available at: <https://www.guru99.com/database-normalization.html> [Accessed 19 October 2020].

Intl.siyavula.com. 2020. How To Get Rid Of Anomalies | Database Design Concepts | Siyavula. [online] Available at: <https://intl.siyavula.com/read/it/grade-12-it/database-design-concepts/02-database-design-concepts?id=sec2-3> [Accessed 19 October 2020].

Studytonight.com. 2020. 1NF, 2NF, 3NF And BCNF In Database Normalization | Studytonight. [online] Available at: <https://www.studytonight.com/dbms/database-normalization.php> [Accessed 19 October 2020].

S, V., 2020. Database Table Denormalization Example - Dwgeek.Com. [online] DWgeek.com. Available at: <https://dwgeek.com/database-table-denormalization-example.html/> [Accessed 19 October 2020].

Techopedia.com. 2020. What Is Normalization? - Definition From Techopedia. [online] Available at: <https://www.techopedia.com/definition/1221/normalization> [Accessed 19 October 2020].

Topic 05: Structured Query Language (SQL) Part I

Lesson Learning Outcomes

- Explain the purpose and importance of the Structured Query Language (SQL)
- List the data types supported by the SQL Standard
- Describe the facilities provided by SQL Standard for integrity control.
- Describe the purpose of Data Definition Language (DDL)
- Demonstrate practical skills in using SQL

5.1 Introduction to SQL

SQL stands for Structured Query Language, which is a standardised language for interacting (e.g. storing and managing) with RDBMS (Relational Database Management System). Almost all RDBMS (e.g. MySql, Oracle, Infomix, Sybase, MS Access) use **SQL** as the standard database query language.

5.2 Writing SQL Commands

SQL statement consists of reserved words and user-defined words.

Reserved words
- They are a fixed part and have a fixed meaning. Must be spelled exactly and cannot be split across lines.

User-defined words
- They are created by the user which denotes the names of various database objects such as tables, columns, views, indexes, and others. The statement used to be built according to a set of syntax rules although the standard does not require it.

Before writing SQL commands, we need to understand the structure of an SQL statement and the notation to use to define the format of the various SQL constructs.
As well, SQL statements will be more readable if indentation and lineation are used.

For example:

- each clause in a statement should begin on a new line;
- the beginning of each clause should line up with the beginning of other clauses;
- if a clause has several parts, they should each appear on a separate line and be indented under the start of the clause to show the relationship.
- uppercase letters are used to represent reserved words and must be spelled exactly as shown;
- lowercase letters are used to represent user-defined words;
- a vertical bar (|) indicates a choice among alternatives; for example, a | b | c;
- curly braces indicate a required element; for example, {a};

- square brackets indicate an optional element; for example, [a];
- an ellipsis (. . .) is used to indicate optional repetition of an item zero or more times.

(Connolly T., Begg C., 2015, p. 196)

SQL is basically a combination of four different languages, they are –

DQL (Data Query Language)

- DQL is used to fetch the information from the database which is already stored there.

DDL (Data Definition Language)

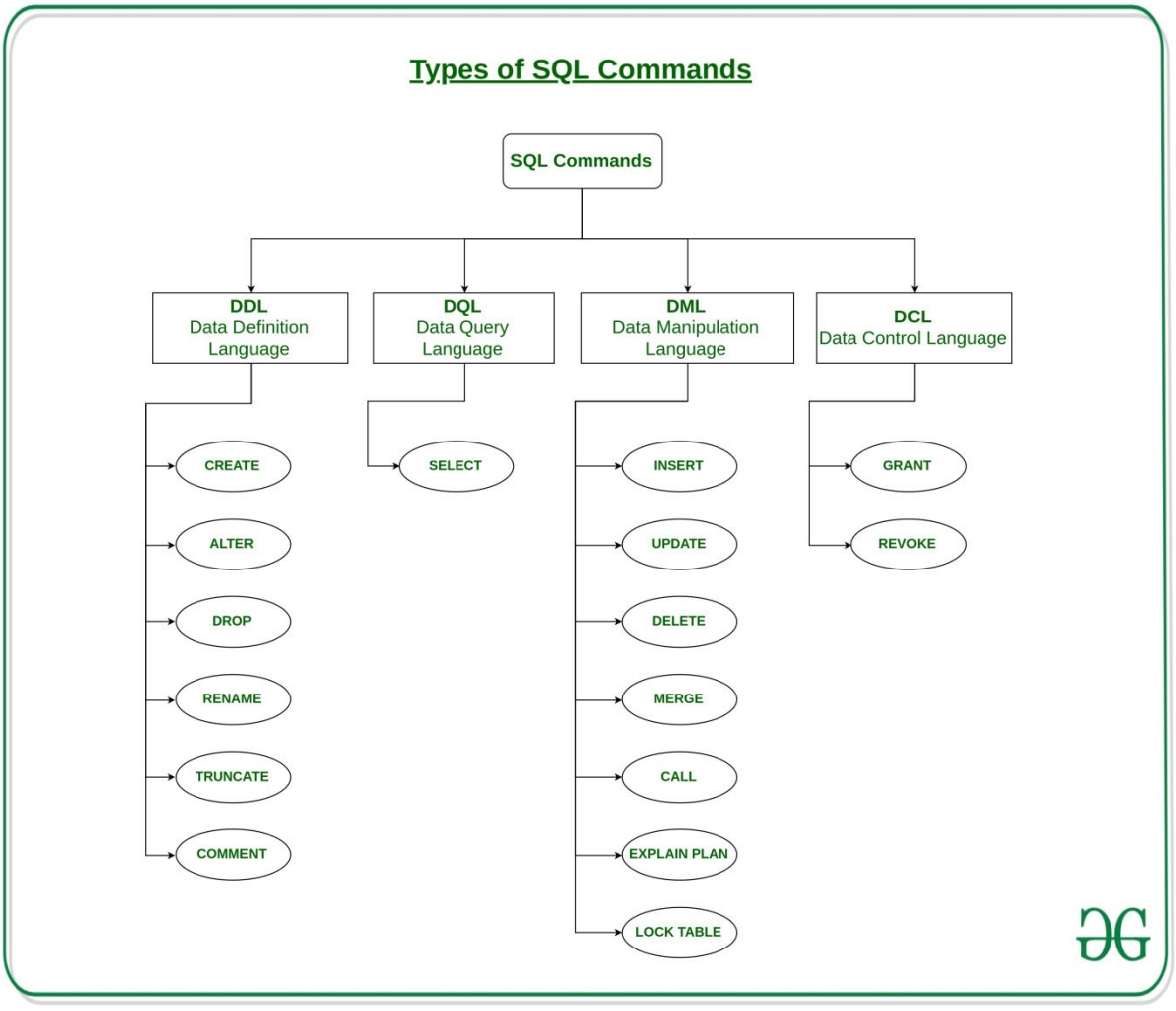
- DDL is used to define table schemas.

DCL (Data Control Language)

- DCL is used for user & permission management. It controls the access to the database.

DML (Data Manipulation Language)

- DML is used for inserting, updating and deleting data from the database.



(Introduction to SQL, 2020)

5.3 The ISO SQL Data Types

In this section, we have to define what would constitute to a valid identifier in SQL.

SQL Identifiers

They are to identify objects in the database, such as *table* names, *view* names, and *columns*.

The default character sets consists of the upper-case letters **A…Z**, the lower-case letters **a…z**, the digits **0…9**, and the underscore character (**_**).

Some restrictions are imposed on an identifier:

- an identifier can be no longer than 128 characters (most dialects have a much lower limit than this);
- an identifier must start with a letter;
- an identifier cannot contain spaces.

ISO SQL data types

DATA TYPE	DECLARATIONS				
boolean	BOOLEAN				
character	CHAR	VARCHAR			
bit†	BIT	BIT VARYING			
exact numeric	NUMERIC	DECIMAL	INTEGER	SMALLINT	BIGINT
approximate numeric	FLOAT	REAL	DOUBLE PRECISION		
datetime	DATE	TIME	TIMESTAMP		
interval	INTERVAL				
large objects	CHARACTER LARGE OBJECT		BINARY LARGE OBJECT		

(Connolly T., Begg C., 2015, p. 235)

SQL Scalar Data Types

The character and bit are collectively referred to as string data types and numeric data types.

The scalar functions require either one or two single-value arguments. Scalar functions can be nested to any level.

The types of scalar functions are as follows:

- Data type conversion
- Numeric
- String
- Date
- Hash
- Random number

(OpenROAD, 2020)

5.4 Integrity Enhancement Feature

Integrity control consists of constraints that need to be enforced in order to protect the database from becoming inconsistent. There are 5 types of integrity constraint

- Required data;
- Domain constraints;
- Entity integrity;
- Referential integrity;
- General / Enterprise constraints.

Required Data

Some columns must contain some valid value, they are not allowed to contain nulls. SQL provides the **NOT NULL** column specifier in the **CREATE TABLE** statement to enforce the required data constraint. For example every employee must have a position. The column position of the Employee table cannot be null:

```
Position VARCHAR (15) NOT NULL
```

Domain constraints

Every column will have a set of allowable values. For example, gender of the employee is either male (M) or female (F). SQL uses **CHECK** clause to enforce this domain constraint on a column.

```
Sex CHAR NOT NULL CHECK (sex IN ('M','F'))
```

Entity integrity

SQL supports entity integrity with **PRIMARY KEY** clause as primary key value of a table. It must be unique and cannot be null.

```
PRIMARY KEY (EmpNo)
```

To define a composite primary key, we specify multiple column names

```
PRIMARY KEY (clientNo,propertyNo)
```

Referential Integrity

Since a **foreign key** links each row in the child table to a parent table containing the matching candidate key value, it means a foreign key must contains an existing value or valid row in the parent table.

```
FOREIGN KEY (branchNo) REFERENCES Branch
```

Thus, SQL rejects any INSERT or UPDATE operation that attempts to create a foreign key value in a child table without a matching candidate key value in the parent table. The action SQL takes for any UPDATE or DELETE operation of a candidate key value in the parent

table must have some matching rows in the child table is dependent on the **referential action** specified using the ON UPDATE and ON DELETE sub-clauses.

SQL supports 4 options regarding the actions to be taken:

- **CASCADE:** Delete the row from the parent table will automatically delete the matching rows in the child table since the candidate key is used as a foreign key in another table.

- **SET NULL:** Delete the row from the parent table will set the foreign key value(s) in the child table to NULL. Valid only if the foreign key columns do not have the NOT NULL qualifier specified.

- **SET DEFAULT:** Delete the row from the parent table will need to set each component in the child table to the specified default value provided the foreign key columns have a DEFAULT value specified.

- **NO ACTION:** Reject the delete operation from the parent table. This is the default setting if the ON DELETE rule is omitted.

  FOREIGN KEY (staffNo) REFERENCES Staff ON DELETE SET NULL

Enterprise Constraints;

Updates to tables may be constrained by enterprise rules governing real-world transactions that are represented by the updates. The CREATE ASSERTION statement is an integrity constraint that is not directly linked with a table definition;

```
CREATE ASSERTION AssertionName
CHECK (searchCondition)
```

When a general constraint involves more than one table, it may be preferable to use an ASSERTION rather than duplicate the check in each table.

```
CREATE ASSERTION StaffNotHandlingTooMuch
CHECK(NOT EXISTS(SELECT staffNo
        FROM PropertyForRent
        GROUP BY staffNo
        HAVING COUNT(*)>100));
```

("ISO SQL data types," n.d.)

5.4 DDL Commands

SQL DDL commands can be used to define the database schema. It allows database objects to be *created, modify* the structure and *destroyed.*

Examples of DDL commands:

- **CREATE** – is used to create the database or its objects (like table, index, function, views, store procedure and triggers).
- **DROP** – is used to delete objects from the database.
- **ALTER**-is used to alter the structure of the database.
- **TRUNCATE**–is used to remove all records from a table, including all spaces allocated for the records are removed.
- **COMMENT** –is used to add comments to the data dictionary.
- **RENAME** –is used to rename an object existing in the database.

Some Examples:

CREATE
The very first step to store the data is to create a database in a well-structured manner. The **CREATE DATABASE** statement is used to create a new database in SQL.

```
CREATE DATABASE database_name;
```

Next, store the data in a table using **CREATE TABLE** which comprises of rows and columns.

```
CREATE TABLE Students
(
ROLL_NO int(3),
NAME varchar(20),
SUBJECT varchar(20),
);
```

DROP
The command is use to remove a user account along with its privileges from the MySQL completely.

```
DROP USER 'user'@'host';
```

Example:
From this:

```
DROP USER 'gfguser1'@'localhost';
```

```
+---------------+-------------------+
| user          | host              |
+---------------+-------------------+
| gfguser1      | localhost         |
| gfguser2      | localhost         |
| gfguser3      | localhost         |
| gfguser4      | localhost         |
+---------------+-------------------+
```

To this:

```
+---------------+-------------------+
| user          | host              |
+---------------+-------------------+
| gfguser2      | localhost         |
| gfguser3      | localhost         |
| gfguser4      | localhost         |
+---------------+-------------------+
```

ALTER

It is used to *add, delete/drop* or *modify* columns in the existing table. It is also used to *add* and *drop* various constraints on the existing table.

ADD columns into the existing table:

```
ALTER TABLE table_name
        ADD (Columnname_1  datatype,
        Columnname_2  datatype,
        …
        Columnname_n  datatype);
```

DROP COLUMN to delete unwanted columns from the table.

```
ALTER TABLE table_name
DROP COLUMN column_name;
```

Multiple columns can be modified using MODIFY command.

```
ALTER TABLE table_name
MODIFY column_name column_type;
```

References

beginnersbook.com. 2020. Introduction To SQL. [online] Available at: <https://beginnersbook.com/2018/11/introduction-to-sql/> [Accessed 19 October 2020].

Docs.actian.com. 2020. Openroad. [online] Available at: <https://docs.actian.com/openroad/6.0/index.html#page/LangRef/Scalar_Functions.htm> [Accessed 19 October 2020].

The ISO SQL data types. (n.d.). Angelfire: Welcome to Angelfire. https://www.angelfire.com/al3/nop/sql_ddl.htm

Topic 06: Structured Query Language (SQL) Part II

Lesson Learning Outcomes

- Explain the purpose of creating simple database.
- Describe the purpose of Data Manipulation Language (DML).
- List SQL aggregate functions and its usage.

6.1 Create Simple Database

There are two CREATE statements that need to be addressed:
- CREATE DATABASE
- CREATE TABLE

Create Database

The very first step to store the data in a structured manner is to create a database. In SQL, Data Definition Language (DDL) commands are used to create and modify the structure of a database and objects.

Syntax:
CREATE DATABASE Database_Name

Try:
Created a database ' **testDB1** '

Create Table

Next, we need to create tables (comprising of rows and columns). We have to provide names of the columns, type of data to be stored in columns, size of the data etc.

Syntax:
```
CREATE TABLE table_name (
   column1 datatype,
   column2 datatype,
   column3 datatype,
  ....
);
```

Example:

```
CREATE TABLE Students
(
ROLL_NO int(3),
NAME varchar(20),
SUBJECT varchar(20),
```

);

Try:
Created a table of an ' **Employee** '

6.2 DML commands

Data Manipulation Language (DML) commands are used for managing data within tables.

Examples of DML commands:

SELECT – To select records or data from a table
INSERT – Insert data into a database table.
UPDATE – Update existing records within a table
DELETE – Delete unwanted records from a table

Syntax : SELECT
SELECT *column1*, *column2, ...*
FROM *table_name*;

Example:
SELECT CustomerName, City FROM Customers;

Syntax : INSERT INTO SELECT
INSERT INTO *table2* (*column1*, *column2*, *column3*, ...)
SELECT *column1*, *column2*, *column3*, ...
FROM *table1*
WHERE *condition*;

Example:
INSERT INTO Customers (CustomerName, City, Country)
SELECT SupplierName, City, Country FROM Suppliers
WHERE Country='Germany';

Syntax : UPDATE
UPDATE *table_name*
SET *column1* = *value1*, *column2* = *value2*, ...
WHERE *condition*;

Example:
UPDATE Customers

```
SET ContactName = 'Alfred Schmidt', City= 'Frankfurt'
WHERE CustomerID = 1;
```

Syntax : DELETE
DELETE FROM *table_name* WHERE *condition*;

Example:
DELETE FROM Customers WHERE CustomerName='Alfreds Futterkiste';

6.3 Aggregate functions

Aggregate functions are used for summarizing results and obtaining useful data through SQL commands (e.g. reports in the form of tables using simple calculations).

Some commonly used SQL aggregate functions include:

- **AVG** – calculates the average of a set of values.
- **COUNT** – counts rows in a specified table or view.
- **MIN** – gets the minimum value in a set of values.
- **MAX** – gets the maximum value in a set of values.
- **SUM** – calculates the sum of values.

Syntax : AVG() SELECT AVG(*column_name*) FROM *table_name* WHERE *condition*;	**Syntax : COUNT()** SELECT COUNT(*column_name*) FROM *table_name* WHERE *condition*;
Syntax : MIN() SELECT MIN(*column_name*) FROM *table_name* WHERE *condition*;	**Syntax : MAX()** SELECT MAX(*column_name*) FROM *table_name* WHERE *condition*;
Syntax : SUM() SELECT SUM(*column_name*) FROM *table_name* WHERE *condition*;	

References

W3Schools Online Web Tutorials. https://www.w3schools.com/sql/sql_create_db.asp.

Topic 07 : Structured Query Language (SQL) Part III

Lesson Learning Outcomes

- Describe different types of Join used in SQL.
- Explain the purpose of Views in database.
- Explain the mechanism of SQL Discretionary Access Control (DAC).

7.1 Types of Join

JOIN clause is used to combine rows (combine data) from two or more tables using the SQL based on a related column between them.
There are basically few different types of JOINs in SQL:

- **(INNER) JOIN**: Returns records that have matching values in both tables.
- **LEFT (OUTER) JOIN**: Returns all records from the left table, and the matched records from the right table.
- **RIGHT (OUTER) JOIN**: Returns all records from the right table, and the matched records from the left table.
- **FULL (OUTER) JOIN**: Returns all records when there is a match in either left or right table.

A. Syntax : INNER JOIN

```
SELECT column_name(s)
FROM table1
INNER JOIN table2
ON table1.column_name = table2.column_name;
```

Example:

```
SELECT Orders.OrderID, Customers.CustomerName
FROM Orders
INNER JOIN Customers
ON Orders.CustomerID = Customers.CustomerID;
```

B. Syntax : LEFT JOIN

```
SELECT column_name(s)
FROM table1
LEFT JOIN table2
ON table1.column_name = table2.column_name;
```

Example:

```
SELECT Customers.CustomerName, Orders.OrderID
FROM Customers
LEFT JOIN Orders
ON Customers.CustomerID = Orders.CustomerID
ORDER BY Customers.CustomerName;
```

C. Syntax : RIGHT JOIN

SELECT *column_name(s)*
FROM *table1*
RIGHT JOIN *table2*
ON *table1.column_name* = *table2.column_name*;

Example:
SELECT Orders.OrderID, Employees.LastName, Employees.FirstName
FROM Orders
RIGHT JOIN Employees
ON Orders.EmployeeID = Employees.EmployeeID
ORDER BY Orders.OrderID;

D. Syntax : FULL OUTER JOIN
SELECT *column_name(s)*
FROM *table1*
FULL OUTER JOIN *table2*
ON *table1.column_name* = *table2.column_name*
WHERE *condition*;

Example:
SELECT Customers.CustomerName, Orders.OrderID
FROM Customers
FULL OUTER JOIN Orders
ON Customers.CustomerID=Orders.CustomerID
ORDER BY Customers.CustomerName;
Note: FULL OUTER JOIN keyword returns all matching records from both tables whether the other table matches or not.
Test out the SQL commands form source:
https://www.w3schools.com/sql/sql_join_inner.asp

7.2 Views

Views are the virtual relations, which consist of columns, rows from the referenced table. They do not exist in the database but can be produced upon request at the time of request.

To the user, a view appears just like a real table, however, it does not necessarily exist in the database. A view is defined as a query where the DBMS stores the definition of the view in the database. Upon request, one has to look up this definition or SQL statement and perform the corresponding request. Thus, View always shows up-to-date data.

Syntax : CREATE VIEW
CREATE VIEW *view_name* AS
SELECT *column1*, *column2*, ...
FROM *table_name*
WHERE *condition*;

Example :

```
CREATE VIEW Manager3Staff
AS SELECT *
FROM Staff
WHERE branchNo = 'B003';
```

A view called Manager3Staff is created with the same column names as the Staff table but containing only those rows where the branch number is B003.

(Connolly T., Begg C., 2015, p. 251)

A view is deleted with the DROP VIEW command.

Syntax : SQL DROP VIEW

DROP VIEW *view_name*;

Example

DROP VIEW [Brazil Customers];

7.3 Discretionary Access Control

Discretionary Access Control (DAC) is a mechanism to ensure that only authorized users can access the intended database with certain mode (e.g. read or write). It enforces database security in a multiple user database environment. DAC allows the owner to determine who will have accessed to objects they control.

SQL supports discretionary access control through GRANT and REVOKE statements.

With **GRANT** command: it allows users to grant access rights to other users.

With **REVOKE** command: it removes user access rights to the database objects.

Syntax : GRANT

```
GRANT privilege_name
ON object_name
TO {user_name |PUBLIC |role_name}
[WITH GRANT OPTION];
```

- ***privilege_name*** is the access right or privilege granted to the user. Some of the access rights are ALL, EXECUTE, and SELECT.
- ***object_name*** is the name of an database object like TABLE, VIEW, STORED PROC and SEQUENCE.
- ***user_name*** is the name of the user to whom an access right is being granted.
- ***PUBLIC*** is used to grant access rights to all users.
- ***ROLES*** are a set of privileges grouped together.
- ***WITH GRANT OPTION*** - allows a user to grant access rights to other users.

Example

```
GRANT UPDATE (EMPNO,DEPT)
ON TABLE EMP
TO NATZ;
```

- This statement grants UPDATE privileges on columns EMPNO and DEPT in the EMP table to user NATZ.

Syntax : REVOKE

```
REVOKE privilege_name
ON object_name
FROM {user_name |PUBLIC |role_name}
```

Example

```
REVOKE UPDATE
ON EMP
FROM NATZ;
```

- This statement revokes the UPDATE privilege on the EMP table from NATZ.
("SQL GRANT, REVOKE, Privileges and Roles", 2020)
("IBM Knowledge Center", 2020)

Review Questions

1. Discuss the differences between the Join operations.
2. Discuss the advantages and disadvantages of views.
3. Describe how the access control mechanisms of SQL work.

Source: Connolly T., Begg C., 2015, p. 180, 268

References

IBM Knowledge Center. (2020). Retrieved 26 October 2020, from https://www.ibm.com/support/knowledgecenter/en/SSEPEK_12.0.0/intro/src/tpc/db2z_.

SQL GRANT, REVOKE, Privileges and Roles. (2020). Retrieved 26 October 2020, from https://beginner-sql-tutorial.com/sql-grant-revoke-privileges-roles.htm.

Topic 08: Transaction Management

Lesson Learning Outcomes

- Describe the properties of Transactions.
- List different types of lock protocols.
- Explain the issues and approaches involved in the processing of concurrent database transactions.

Transaction management refers to the tasks of processing multiple transactions on a database while maintaining **A**tomicity, **C**onsistency, **I**solation, and **D**urability; known commonly as **ACID** properties. This is to ensure accuracy, completeness, and data integrity.

8.1 Properties of Transactions

ACID Properties

A transaction must maintain ACID properties:

- **Atomicity** − A transaction must be treated as a single operation. Either all of its operations are executed or none. No transaction should be left partially completed.
- **Consistency** – Data should be in a consistent state when a transaction starts and when it ends. For example, funds being transferred from one account to another. Under consistency rule, total value of funds in both the accounts must still be the same before and after transaction.
- **Isolation** − Where more than one transaction are being executed in parallel, For example, in an application that transfers funds from one account to another, the transactions that run concurrently appear to be serialized. Property of isolation states that all the transactions will be carried out as if it is the only transaction in the system. It will not affect the existence of any other transaction.
- **Durability** − Database holds all its latest updates even if the system fails. For example, a transaction commits to transfer funds to another account, but the system fails before the data could be written, that data will be updated once the system comes back into action.

8.2 Locks

The importance of control in multiple transactions in multiprogramming environment is vital. Concurrency control protocols like *Lock-based Protocols* can lock data variable and help synchronize access to the database items.

In the database systems, any transaction cannot read or write data until it acquires an appropriate lock on it. Basically, there are 2 kinds of locks:

- **Binary Locks** − A lock on a data item can be in two states; it is either locked or unlocked.

- **Shared/exclusive** – It differentiates the locks based on their uses. To perform a **Write** operation; also known as exclusive lock. It allows more than one transaction to read and write on the same data item. **Read** locks are shared because no data value is being changed.

There are **4 types of lock protocols** available:

1. **Simplistic Lock Protocol**
It allows transactions to obtain a lock on every object before a 'write' operation is executed. Transactions may unlock the data item after ending the ‘write’ operation.

2. **Pre-claiming Lock Protocol**
The protocols evaluate its operations and create a list of required data items on which they need. Before execution, the system requests all the locks needed. If all locks are granted, the transaction will execute otherwise the transaction will rolls back and waits until all the locks are granted. It releases all locks when all its operations are over.

8.3 Two phase locking

3. **Two-Phase Locking (2PL)**

This locking protocol divides the execution phase of a transaction into three parts.

- In the first part, the transaction starts executing, it seeks permission for the locks it requires.
- The second part, the transaction acquires all the locks. Third phase starts as soon as the transaction releases its first lock.
- The third part, the transaction cannot demand any new locks; it only releases the acquired locks.

There are 2 phases of 2PL:

- **Growing phase:** In the growing phase, all locks are being acquired by the transaction and none can be released.
- **Shrinking phase:** In the shrinking phase, locks held by the transaction are being released and no new locks can be acquired.

4. **Strict Two-Phase Locking (Strict-2PL)**
The first phase of Strict-2PL is identical to 2PL. The transaction continues to execute normally after acquiring all the locks. In contrast to 2PL, Strict-2PL does not release a lock after using it. It holds all the locks from transaction to commit, only then it releases all locks at a time.

8.4 Concurrency control

Concurrency control is an **important** concept for managing simultaneous operations without conflicting with each another or violating the data integrity of databases. It avoids any loss of data and to ensure proper updating of data.

Concurrency control is provided in a database to:

- enforce isolation among transactions.
- preserve database consistency through consistency preserving execution of transactions.
- resolve read-write and write-read conflicts.

Deadlocks

In concurrent computing, a deadlock is a situation where two (or more) transactions are blocked because each process is holding a resource and waiting for another resource held by the other. Thus, neither transaction can continue because each transaction is in the waiting queue

Various concurrency control techniques are: (as seen below). Lock-Based Protocols & Two Phase were covered earlier.

- Lock-Based Protocols
- Two Phase locking Protocol
- Timestamp-Based Protocols
- Multi version concurrency control
- Validation Concurrency Control

Timestamp-based Protocols

It uses algorithm timestamp to serialize the execution of concurrent transactions. It ensures every read and write operations are executed in timestamp order.
Every transaction has a timestamp, and the order is determined by the age of the transaction. Thus, the older transaction is always given priority and is the most commonly used concurrency protocol.

Advantages:

- Schedules are serializable just like 2PL protocols
- No waiting for the transaction and eliminates the possibility of deadlocks!

Disadvantages:

- Starvation is possible if the same transaction is restarted and continually aborted

Multiversion Concurrency Control:

The schemes keep old versions of data item to increase concurrency. Timestamps are used to label the new version of the data item written.

Validation Concurrency Control:

This is an optimistic approach based on the assumption that the majority of the database operations do not conflict. It requires neither locking nor time stamping techniques. Transaction is executed without restrictions until it is committed. Each transaction moves through read, validation and write.

Review Questions

1. The consistency and reliability aspects of transactions are due to the “ACIDity” properties of transactions. Discuss each of these properties and how they relate to the concurrency control and recovery mechanisms. Give examples to illustrate your answer.

2. Describe, with examples, the types of problem that can occur in a multi-user environment when concurrent access to the database is allowed.

Exercises

1. Write an algorithm that checks whether the concurrently executing transactions are in deadlock.

Source: Connolly T., Begg C., 2015, p. 724, 725

References

Concurrency control techniques. (2019, August 26). GeeksforGeeks. https://www.geeksforgeeks.org/concurrency-control-techniques/

DBMS Lock based Protocol - javatpoint. (2020). Retrieved 26 October 2020, from https://www.javatpoint.com/dbms-lock-based-protocol

DBMS Concurrency Control: Two Phase, Timestamp, Lock-Based Protocol. (2020). Retrieved 26 October 2020, from https://www.guru99.com/dbms-concurrency-control.html

DBMS - Concurrency Control - Tutorialspoint. (2020). Retrieved 26 October 2020, from https://www.tutorialspoint.com/dbms/dbms_concurrency_control.htm

IBM Knowledge Center. (2020). Retrieved 26 October 2020, from https://www.ibm.com/support/knowledgecenter/en/SSGMCP_5.4.0/product-overview/acid.html

Rungta, K. (n.d.). DBMS concurrency control: Two phase, Timestamp, lock-based protocol. Meet Guru99 - Free Training Tutorials & Video for IT Courses. https://www.guru99.com/dbms-concurrency-control.html

Two-phase locking. (2020). Retrieved 26 October 2020, from https://en.wikipedia.org/wiki/Two-phase_locking

(2020). Retrieved 26 October 2020, from https://www.tutorialspoint.com/dbms/dbms_transaction.htm

Topic 09: Database Administration and Security

Lesson Learning Outcomes

- Describe the Database Administrator roles and responsibilities.
- Discuss the role of the database administrator in database security and recovery.
- Explain the importance of database security and types of security approaches used in the organizations.
- Describe the approaches for securing DBMSs on the Web.

9.1 DBA Roles and Responsibilities

Database Administrator (DBA) is responsible for the physical realization of the database. The role includes capacity planning, installation, configuration, database design, migration, performance monitoring, security, troubleshooting, backup and data recovery.

Briefly, they can be summaries as:

- Administer and maintain all activities associate with database and ensure optimal performance.
- Provide support to SQL Server through various activities.
- Maintain all Parallax operations and provide support to database operations.
- Monitor database and application levels and ensure work according to user privileges.
- Coordinate with various teams and monitor all replication and recovery procedures for database.
- Design physical database and recommend optimizations of existing databases.
- Determine development and test for various systems.
- Prepare appropriate documents for all tests procedures for databases.
- Manage database and validate it to ensure integrity of data.
- Monitor database performance and monitor efficient working.
- Install various patches and releases on various software.
- Provide support for resource utilization and prepare appropriate reports for same.
- Coordinate with security administrators and participate in various user access processes.
- Manage all recertification process for connectivity.
- Prepare schedule for backup for various operating systems.
- Coordinate with development team and manage all technical specifications.
- Provide an estimate of various programs and projects.

9.2 Database Security

Database security encompasses hardware, software, people, and data.

The amounts of crucial corporate data being stored and the loss or unavailability of this data could prove to be disastrous.

Database security need to be addresses urgently:

- theft and fraud;
- loss of confidentiality (secrecy);
- loss of privacy;
- loss of integrity;
- loss of availability

To date, many articles have been written pertaining to useful database security best practices.

Database Hardening Best Practices

This checklist was developed by IST system administrators to provide guidance for securing databases storing sensitive or protected data. Implementing these security controls will help to prevent data loss, leakage, or unauthorized access to your databases.

Physical Database Server Security

- The physical machine hosting a database is housed in a secured, locked and monitored environment to prevent unauthorized entry, access or theft.
- Application and web servers are not hosted on the same machine as the database server.

Firewalls for Database Servers

- The database server is located behind a firewall with default rules to deny all traffic.
- The database server firewall is opened only to specific application or web servers, and firewall rules do not allow direct client access. If the development environment cannot meet this requirement, then protected data is not stored in the development database server and mock data is made up for development. Data obfuscation of production data is not sufficient.
- Firewall rule change control procedures are in place and notification of rule changes are distributed to System Administrators (SAs) and Database Administrators (DBAs).
- Firewall rules for database servers are maintained and reviewed on a regular basis by SAs and DBAs. If using the IST provided firewall service, the rules are also regularly reviewed by the Information Security Office (ISO).
- Regularly test machine hardening and firewall rules via network scans, or by allowing ISO scans through the firewall.

Database Software

- The database software version is currently supported by the vendor or open source project, as required by the campus minimum security standards.
- All unused or unnecessary services or functions of the database are removed or turned off.

- Unneeded default accounts are removed, or else passwords are changed from defaults.
- Null passwords are not used, and temporary files from the install process that may contain passwords are removed.
- Database software is patched to include all current security patches. Provisions are made to maintain security patch levels in a timely fashion.

Application / Web Servers / Application Code

- Destination systems (application/web servers) receiving protected data are secured in a manner commensurate with the security measures on the originating system. All servers and clients meet minimum security standards.
- All servers, applications and tools that access the database are documented.
- Configuration files and source code are locked down and only accessible to required OS accounts.
- Application code is reviewed for SQL injection vulnerabilities.
- No "Spyware" is allowed on the application, web or database servers.

User/Client Workstations

- If users are allowed protected data on their workstations, then client workstations meet the minimum security standards.
- If users are allowed protected data on their workstations, then the workstation is protected against unauthorized access to a session by deploying screen savers. Users understand the requirement to lock their workstations when leaving the station.
- If users are allowed protected data on their workstations, then the workstation should require an individual login and password.
- If users are allowed protected data on their workstations, then protected data on the client workstation is encrypted by the workstation's operating system.
- Protected data is not stored on transportable devices.
- Protected data is never sent via email, either in the body or as an attachment, by either users or as an automated part of the system.
- Protected data that is no longer needed is routinely deleted.
- If users are allowed protected data on their workstations, then no "Spyware" is allowed on the client workstations.

Administrator Accounts / Permissions / Passwords

- DBAs understand their responsibility for reviewing all requested script and database changes to ensure the security of the system is not compromised.
- Accounts with system administration capabilities are provided to as few individuals as is practical, and only as needed to support the application.
- All Developers, Vendors, SAs, DBAs & Contractors have signed a non-disclosure agreement.
- All developers, SAs, DBAs and contractors have passed a criminal background check if required by the background check policy. The background check policy may be found at http://campuspol.chance.berkeley.edu/Policies/BackgroundChecks.htm(link is external)
- Operating system accounts used by DBA staff to login to dataserver machines for administrative duties are individual accounts, and not a shared group account.
    - When possible, the daemon OS account that is required to run the dataserver process does not allow a direct login.

    - Instead, individual OS accounts are used to login, then sudo or su to the daemon account (for UNIX) or disallow desktop login (Windows).
    - Database accounts used by DBA staff for administrative duties are individual accounts, and not a shared group account.
    - A group account is permitted for running automated DBA maintenance and monitoring jobs, such as backups.
    - This group account is not used for daily interactive tasks by the DBA group, except when required to troubleshoot maintenance and monitoring jobs.
- Passwords for all DBA operating system accounts and database accounts are strong passwords, and are changed when administrators/contractors leave positions. See: Password complexity guidelines
- If the DBA and developer roles are being filled by a single person, changes are approved by the Data Proprietor.

User Database Roles / Permissions / Passwords / Management & Reporting

- Secure authentication to the database is used.
- The procedure for provisioning and reviewing access to the database is documented. The data proprietor has signed the procedures document.
- Only authorized users have access to the database.
- Users are granted the minimal permissions necessary for their job function in the database. Permissions are managed through roles or groups, and not by direct grants to user IDs where possible.
- Strong passwords in the database are enforced when technically possible, and database passwords are encrypted when stored in the database or transmitted over the network.
- Applications require individual database login/password and roles/grants when possible. When not possible, application accounts may be utilized. However, the login ID and password must be secured in this case, and this information does not exist on the client workstation.
- Applications should manage user permissions and auditing to meet the Data Proprietors requirements.
- User database objects with protected data do not have public grants when possible. Document any public grants if needed in databases with protected data.
- Non-DBA accounts do not allow the granting of roles or permissions in any environment with protected data (QA, Production, Dev).
- Database accounts are locked after at most six failed logins.
- Procedure to address inactive users are documented and approved by the Data Proprietor.
- A report of elevated database permissions is provided to the data proprietor by the DBAs on a quarterly basis.
- A report of all access rights for users is provided to the data proprietor by the DBAs on a regular basis. Twice a year is the recommended interval.

Protected Data

- Only the protected data required for the business function is kept within the database. When possible, historical information is purged when no longer required.
- Redundancy of protected data is eliminated throughout the system, and shadowing of protected data outside the system of record is avoided wherever possible. Hashing functions are applied to protected data elements before storing if the data is only required for matching purposes. If possible, disassociate protected data from personally identifiable

information and keep offline until needed. If data transfers are required for other applications, notify them of protected data and its security requirements.
- Protected data in non-production environments is held to the same security standards as production systems. In cases where non-production environments are not held to the same security standard as required in production, data in these non-production environments must either be encrypted using industry-standard algorithms, or else test data must be made up for these systems. Data obfuscation is not sufficient.
- The protected data elements within the database are documented.
- Protected data is never used as a key in a table.

Change Management
- Change management procedures are documented and meet the data proprietor's requirements.
- Change management controls are in place to log all changes to the production database.
- All programs scheduled to run against the database which read or modify production data are documented.

Database Auditing
- All logins to operating system and database servers, successful or unsuccessful, are logged. These logs are retained for at least one year.
- Database objects with protected data have auditing turned on where technically possible.
- Audit logs are regularly reviewed by knowledgeable and independent individuals appointed by the data proprietor to meet the data proprietor's requirements. These requirements and the review process are documented.
- Accounts that are locked due to maximum database login failures trigger an automatic notification of the security administrator(s) responsible for this system.

Database Backup & Recovery
- The backup and recovery procedures are documented and meet data proprietor's requirements.
- Backup and recovery procedures are periodically tested.
- Backup retention intervals are documented and sufficient to meet the business resumption requirements and expectations of the data proprietor.

Database Encryption & Key Management
- Protected data is encrypted during transmission over the network using encryption measures strong enough to minimize the risk of the data's exposure if intercepted or misrouted from database to client workstation.
- If database-level encryption for protected data is implemented, procedures for secure key management are documented. (Check National Institute of Standards and Technology (NIST) for current recommendations.) *Note: It is recommended that all application layers (network, application, client workstation) are already encrypted before encrypting the database. Database encryption is not a substitute for any of the above requirements. Database encryption of protected data is not mandatory to meet this standards document.*
- For data subject to disclosure that is encrypted at storage, the means to decrypt must be available to more than one person and approved by the data proprietor.
- Backup tapes store backups of the database in an encrypted format, and the tapes do not store the plain text encryption keys necessary to decrypt the backups.

- Key management procedures for decrypting backups are documented, available to more than one person and approved by the data proprietor.

9.3 Countermeasures—Computer-Based Controls

There are many different forms of countermeasure to threats on computer systems ranging from physical controls to administrative procedures. Followings are some computer-based security controls:

- authorization,
- access controls,
- views,
- backup and recovery,
- integrity,
- encryption, and
- RAID technology.

9.3.1 Authorization

The granting of a right or privilege that enables a subject to have legitimate access to a system or a system's object.

System administrator is usually the person responsible for allowing users to have access to a computer system (e.g. biometrics, keycards or by creating individual user accounts and password, etc). However, it does not necessarily authorize him/her the rights to access the DBMS. The Database Administrator (DBA) is the person whom set up individual user accounts and passwords to access the DBMS itself. As such, DBMSs will usually maintain a list of valid user identifiers and associated passwords.

Besides, many are confuse with the terms authentication and authorization. So what is the difference?
Authentication is the security practice to confirm someone is who they claim to be whereas authorization is concerned with the level of access each user is granted.
For example, think of a traveller checking into a hotel. When they register at the front desk, they are asked to provide a passport to prove that they are indeed the person whose name is on the reservation. This is an example of authentication.
("Attention required!," n.d.)

9.3.2 Access Controls

The normal way of permitting access controls to a database system is based on granting and revoking of privileges within the database. A privilege allows a user to create or access (e.g. read, write, or modify) some database object (relation, view, or index) or to run some specific DBMS utilities. Privileges are granted to users to accomplish the tasks required for their jobs. The database provides various types of access controls:

- Discretionary Access Control (DAC)
- Mandatory Access Control (MAC)

Discretionary Access Control (DAC)
In discretionary access control (DAC), the owner specifies or allows which end user to access their objects/data. This is typically the default access control mechanism because control of access is based on the discretion of the owner.

The operating systems (e.g. Windows, Linux, and Macintosh, etc) will make the access control decision based on the access privileges you created earlier.

Mandatory Access Control (MAC)
Whereas in mandatory access control (MAC), the system (not the users) specifies which subjects can access specific data objects. It is the strictest of all levels of control.

With MAC, the process of gaining access:

- The administrator configures access policies and defines security attributes: confidentiality levels, clearances for accessing different projects and types of resources.
- The administrator assigns each subject (user or resource that accesses data) and object (file, database, port, etc.) a set of attributes.
- When a subject attempts to access an object, the operating system examines the subject's security attributes and decides whether access can be granted. ("Mandatory access control vs discretionary access control: Which to choose?," 2020)

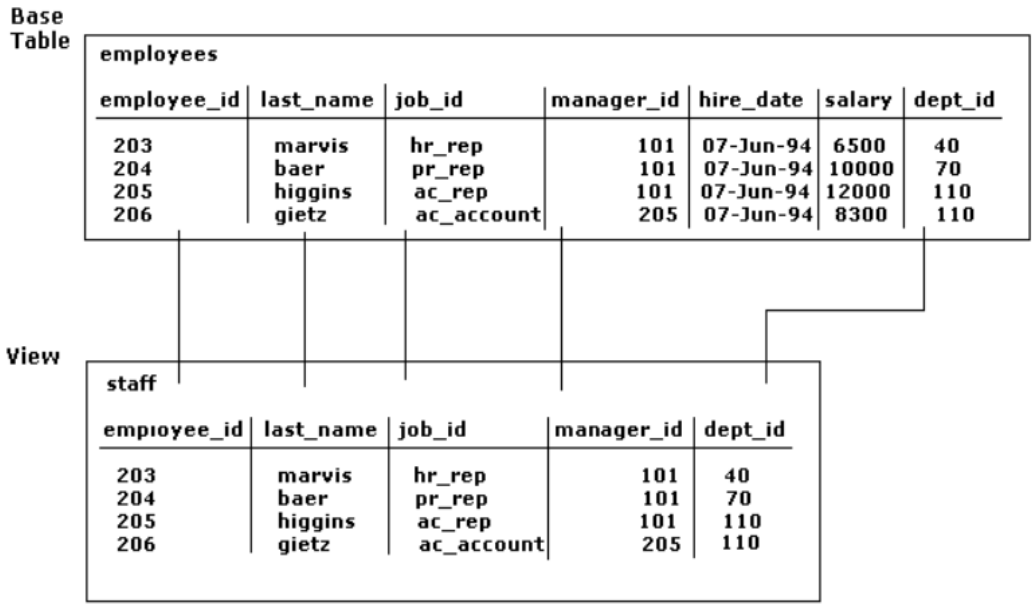
9.3.3 Views

A view is presentation of data selected from one or more relational operations operating on the base relations to produce another relation. It shows the structure of the underlying tables but is a virtual relation that does not actually exist in the database, and is produced upon request by user demand at the time of request.

A view contains no actual data, therefore, it does not necessarily exist in the storage. It creates and shows the tables on which it is based on. It can be manipulated as if it were a base relation.

The contents of a view are derived from query on one or more base relations. Views are dynamic as the data in a view can be updated or deleted, and new data inserted. Such operations will make changes to the base relations. As the operations will directly alter the based tables, integrity constraints on the subject are critical.

Shown below an example of a view called staff derived from the base table employees. Notice the View shows only five of the columns in the base table.



https://docs.oracle.com/cd/B12037_01/network.101/b10773/accessre.htm

(Connolly T., Begg C., 2015, Ch20)

9.3.4 Backup and Recovery

It is a process of periodically copying of the database and log file to offline secured storage media. DBMS should provide these facilities (automatically backed up on regular basis) to restore the database to the latest possible state in the event of system failure. Backup can be in the form full run, incremental, or differential.

The pros and cons to each backup plan type, primarily circle around the performance of obtaining and restoring backup data. A full backup will take longer time to create, however, is much quicker than restoring from incremental or differential backups. A good option maybe a mix of backup plans; full backup once a week and daily incremental backups.

Other question such as how often you should the run this backup may depends on the regulatory frameworks, balance performance and storage costs at a level of risk that is acceptable for the company business.

9.3.5 Integrity

As discussed in our earlier chapter, integrity constraints refer to conditions which must be present for a valid relation. And under RDMS, it can divided into 3 main categories are such as Domain constraints, Key constraints and Referential integrity constraints

There are many types of integrity constraints and it is necessary to understand some of these concepts pertaining to this before we move further.

Null
It represents a value of an attribute that is currently unknown or is not applicable for this row. Nulls can cause implementation problems if not defined.

Entity Integrity
In a base relation, entity integrity constraint states that a primary key value cannot be null because primary key value is used to identify individual rows in relation and if it has a null value, then we cannot identify those rows. A table can contain a null value but not the primary key field.

General Constraints
Lastly, user or database administrators can also specify additional constraints that the data must satisfy that maybe due to the constrains imposed or required by the enterprise.

Examples:

- A class can have a maximum of 30 students.
- A teacher can teach a maximum of four classes per semester.
- An employee cannot take part in more than five projects.
- The salary of an employee cannot exceed the salary of the employee's manager.

("Chapter 9 integrity rules and constraints – Database design – 2nd edition," 2014)

9.3.6 Encryption

Encryption is the method by which data/information is converted into secret code using a special algorithm that renders the data unreadable or hides the information true meaning.
Thus, “to transmit data securely over insecure networks requires the use of a cryptosystem, which includes:

- an encryption key to encrypt the data (plaintext);
- an encryption algorithm that with the encryption key transforms the plaintext into ciphertext;
- a decryption key to decrypt the ciphertext;
- a decryption algorithm that with the decryption key transforms the ciphertext back into plaintext.”

(Connolly T., Begg C., 2015, p. 617)

Basically, there are two main kinds of encryption:

- Symmetric encryption and
- Asymmetric encryption. Also known as public key encryption.

In symmetric encryption, there is only one key, and all communication lines use the same key for encryption and decryption. Data Encryption Standard (DES) is a standard encryption algorithm developed by IBM which uses one key for both encryption and decryption
In asymmetric encryption, there are two keys: one key (public key), shared publicly, is used for encryption, and a different key is used for decryption (private key).
The formulas to encode and decode messages are called *encryption algorithms,* or *ciphers* and the science of encrypting and decrypting the information is called *cryptography*.

9.3.7 RAID (Redundant Array of Independent Disks)

RAID (Redundant Array of Independent Disks or Redundant Array of Inexpensive Disks) is a way of storing the same data in different places on multiple hard disks to protect data in the case of a drive failure and to improve system performance.
This would mean DBMS should continue to operate if one of the hardware components fails. Thus, having redundant components working seamlessly is vital whenever there is one or more component failures.
There are different RAID levels configurations. A brief description of each RAID level is given here:

• **RAID 0**—Nonredundant. This level maintains no redundant data and therefore has the best write performance, as updates do not have to be replicated. Data striping is performed at the level of blocks. A diagrammatic representation of RAID 0 is shown in Figure 20.4(a).

• **RAID 1**—Mirrored. This level maintains (mirrors) two identical copies of the data across different disks. To maintain consistency in the presence of disk failure, writes may not be performed simultaneously. This is the most expensive storage solution. A diagrammatic representation of RAID 1 is shown in Figure 20.4(b).

• **RAID 0+1**—Nonredundant and Mirrored. This level combines striping and mirroring.

• **RAID 2**—Memory-Style Error-Correcting Codes. With this level, the striping unit is a single bit and Hamming codes are used as the redundancy scheme. A diagrammatic representation of RAID 2 is shown in Figure 20.4(c).

• **RAID 3**—Bit-Interleaved Parity. This level provides redundancy by storing parity information on

a single disk in the array. This parity information can be used to recover the data on other disks should they fail. This level uses less storage space than RAID 1, but the parity disk can become a bottleneck. A diagrammatic representation of RAID 3 is shown in Figure 20.4(d).

• **RAID 4**—Block-Interleaved Parity. With this level, the striping unit is a disk block—a parity block is maintained on a separate disk for corresponding blocks from a number of other disks. If one of the disks fails, the parity block can be used with the corresponding blocks from the other disks to restore the blocks of the failed disk. A diagrammatic representation of RAID 4 is shown in Figure 20.4(e).

• **RAID 5**—Block-Interleaved Distributed Parity. This level uses parity data for redundancy in a similar way to RAID 3 but stripes the parity data across all the disks, similar to the way in which the source data is striped. This alleviates the bottleneck on the parity disk. A diagrammatic representation of RAID 5 is shown in Figure 20.4(f).

• **RAID 6**—P+Q Redundancy. This level is similar to RAID 5, but additional redundant data is maintained to protect against multiple disk failures. Errorcorrecting codes are used instead of parity.

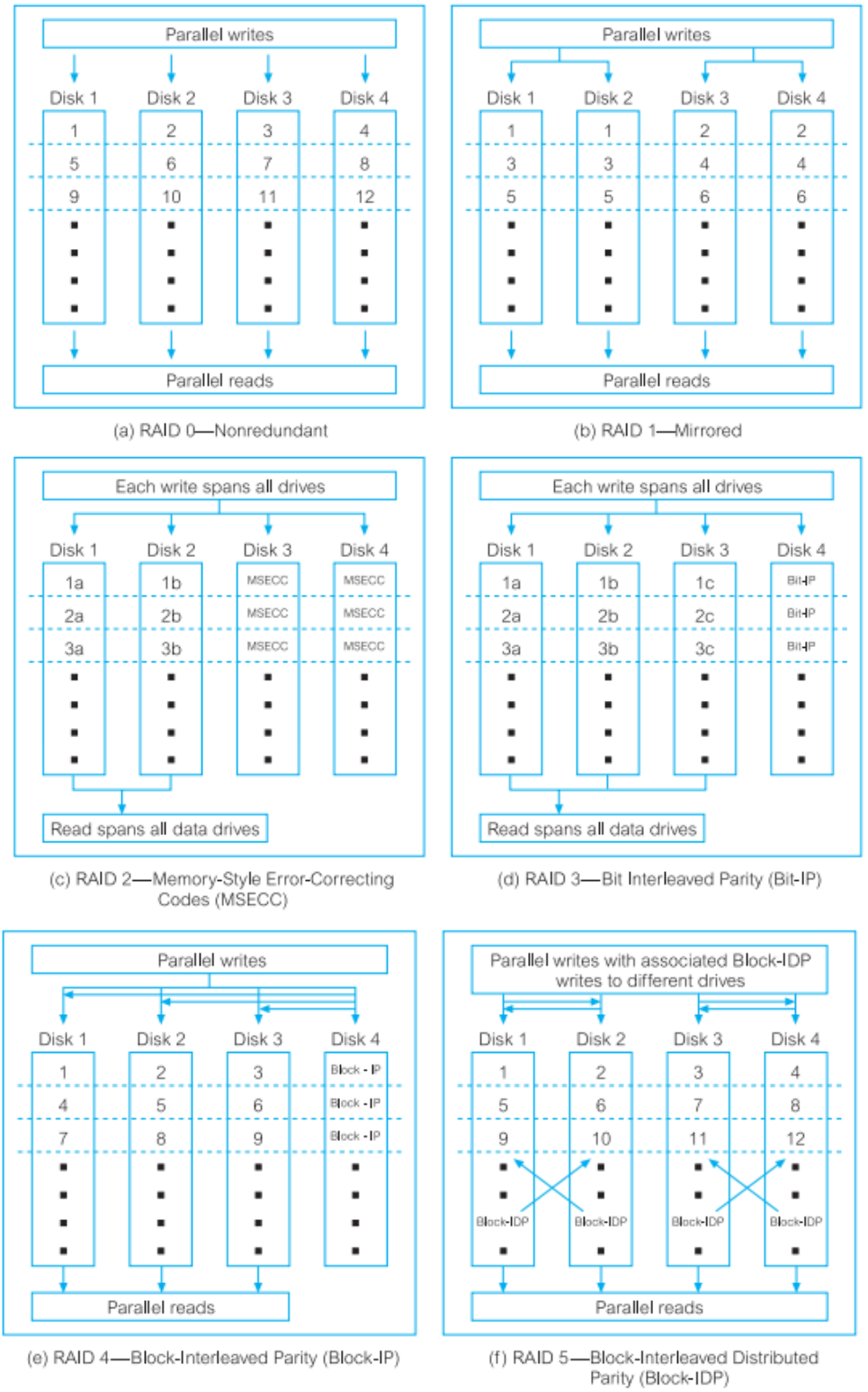


Figure 20.4 RAID levels. The numbers represent sequential data blocks and the letters indicate segments of a data block.

(Connolly T., Begg C., 2015, p. 619,620)

9. 4 DBMSs and Web Security

DBMSs and Web Security focus on how to make a DBMS secure on the Web. As Internet is an open network and communication relies basically on TCP/IP and HTTP as the underlying protocol, it is not designed with security in mind.

Anyone who monitors the traffic can use a “packet sniffing” software to read those messages. However, protecting the transaction solves only part of the problem as once the information reached the Web server, it must also be protected as well.

Besides, information transmitted to the client’s machine may contain executable content such as HTML pages may contain ActiveX controls, JavaScript/VBScript, and/or Java applets. These executable contents can perform malicious actions.

Thus, the need to address database security in these environments is critical.

Proxy Servers

“A proxy server is basically another computer which serves as a hub through which internet requests are processed. By connecting through one of these servers, your computer sends your requests to the server which then processes your request and returns what you were wanting. Moreover, in this way it serves as an intermediary between your home machine and the rest of the computers on the internet. Proxies are used for a number of reasons such as to filter web content, to go around restrictions such as parental blocks, to screen downloads and uploads and to provide anonymity when surfing the internet.”

("What is a proxy or proxy server," 2020)

Firewalls

“A firewall is a system designed to prevent unauthorized access to or from a private network. Firewalls can be implemented as both hardware and software or a combination of both. They are frequently used to prevent unauthorized Internet users from accessing private networks connected to the Internet, especially intranets. All messages entering or leaving the intranet pass through the firewall, which examines each message and blocks those that do not meet the specified security criteria”

(Connolly T., Begg C., 2015, p. 619,620)

Firewall types can be divided into several different categories based on their general structure and method of operation. Here are eight types of firewalls:

- Packet-filtering firewalls
- Circuit-level gateways
- Stateful inspection firewalls
- Application-level gateways (a.k.a. proxy firewalls)
- Next-gen firewalls
- Software firewalls
- Hardware firewalls

- Cloud firewalls

When deciding on which firewall to choose, most organizations would consider the following factors:

- The size of the organization.
- The resources available.
- The level of protection required.

Message Digest Algorithms and Digital Signatures

"A message digest is a fixed size numeric representation of the contents of a message, computed by a hash function. A message digest can be encrypted, forming a digital signature.
Messages are inherently variable in size. A message digest is a fixed size numeric representation of the contents of a message. A message digest is computed by a hash function, which is a transformation that meets two criteria:

- The hash function must be one way. It must not be possible to reverse the function to find the message corresponding to a particular message digest, other than by testing all possible messages.
- It must be computationally infeasible to find two messages that hash to the same digest.

The message digest is sent with the message itself. The receiver can generate a digest for the message and compare it with the digest of the sender. The integrity of the message is verified when the two message digests are the same. Any tampering with the message during transmission almost certainly results in a different message digest.
….
….
The sender can also generate a message digest and then encrypt the digest using the private key of an asymmetric key pair, forming a digital signature. The signature must then be decrypted by the receiver, before comparing it with a locally generated digest."
("IBM knowledge center," n.d.)

Digital Certificates

"A digital certificate is an attachment to an electronic message used for security purposes, most commonly to verify that a user sending a message is who he or she claims to be and to provide the receiver with the means to encode a reply."

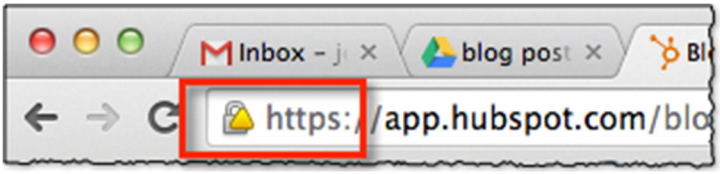
(Connolly T., Begg C., 2015, p. 631)
User gets a digital certificate from a recognized Certificate Authority (CA) such as VeriSign, GeoTrust and others. CA then is a trusted third party that is relied upon to verify the matching of public keys to identity such information.

Digital Certificates is used for electronic transactions such as e-mail, e-commerce and electronic funds transfers.

The most widely accepted standard for Digital Certificates is X.509.

As some website put it simply, "A digital certificate primarily acts like an identification card; something like a driver's license, a passport, a company ID, or a school ID. It basically tells other people who you are."
(Villanueva, n.d.)

Today, websites protected by certs usually display "https" on the site's URL when viewed on your browser's URL bar. This assured user that interaction with the Web site has no eavesdroppers



Secure Sockets Layer and Secure HTTP 630

In the beginning, data was transmitted in plaintext on the Web whom anyone could read if they intercepted. For example, consumer credit card number. Consequently, Secure Sockets Layer (SSL) was created to protect user privacy by encrypting any data that goes between a user and a web server. As a result, intruders could only see scrambled mess if they intercepted.

SSL is able to provide privacy, authentication, and integrity to Internet communications which eventually evolved into Transport Layer Security (TLS).

Any website that implements SSL/TLS has "HTTPS" in its URL. This website includes an SSL/TLS certificate that is signed by a publicly trusted CA. Data is encrypted and transmitted across the web. Because of these, SSL/TLS and HTTPS allow users to securely transmit confidential information (e.g. credit card numbers, social security numbers, etc.) over the internet. This means that anyone who intercepted this data will only see a mangled mix of characters that is almost impossible to decrypt.

Review Questions

1. Explain the purpose and scope of database security.

2. Discuss the role of the database administrator in database security and recovery.

3. Explain the following in terms of providing security for a database:
    a. authorization;
    b. authetication;
    c. access controls;
    d. views;
    e. backup and recovery;
    f. integrity;
    g. encryption;
    h. RAID technology.

4. Describe the approaches for securing DBMSs on the Web.

Exercises

1. Identify the types of security approach that are used by your organization to secure any DBMSs that are accessible over the Web.

2. You are contracted to deploy database security for the university accommodation system. Describe how you will approach the project.

(Connolly T., Begg C., 2015, p. 640, 641)

References

Attention required! (n.d.). Cloudflare - The Web Performance & Security Company | Cloudflare. https://www.cloudflare.com/learning/access-management/what-is-access-control/

Chapter 9 integrity rules and constraints – Database design – 2nd edition. (2014, October 24). BCcampus Open Textbooks – Open Textbooks Adapted and Created by BC Faculty. https://opentextbc.ca/dbdesign01/chapter/chapter-9-integrity-rules-and-constraints/

Database hardening best practices. (n.d.). https://security.berkeley.edu/education-awareness/best-practices-how-tos/system-application-security/database-hardening-best

IBM knowledge center. (n.d.). IBM - United States. https://www.ibm.com/support/knowledgecenter/SSFKSJ_7.5.0/com.ibm.mq.sec.doc/q009810_.htm

List of database administrator DBA responsibilities and duties. (2016, June 29). Great Sample Resume. https://www.greatsampleresume.com/job-responsibilities/data-systems-administration/database-administrator

Mandatory access control vs discretionary access control: Which to choose? (2020, April 10). Ekran System — Insider Threat Protection Software. https://www.ekransystem.com/en/blog/mac-vs-dac

Villanueva, J. C. (n.d.). What is a digital certificate? | Examples of digital certificates. Managed File Transfer | Secure FTP Server | Reverse Proxy | Enterprise File Transfer. https://www.jscape.com/blog/what-is-a-digital-certificate

What is a proxy or proxy server. (2020, October 22). WhatIsMyIP.com®. https://www.whatismyip.com/what-is-a-proxy/)

Topic 10: Distributed DBMS

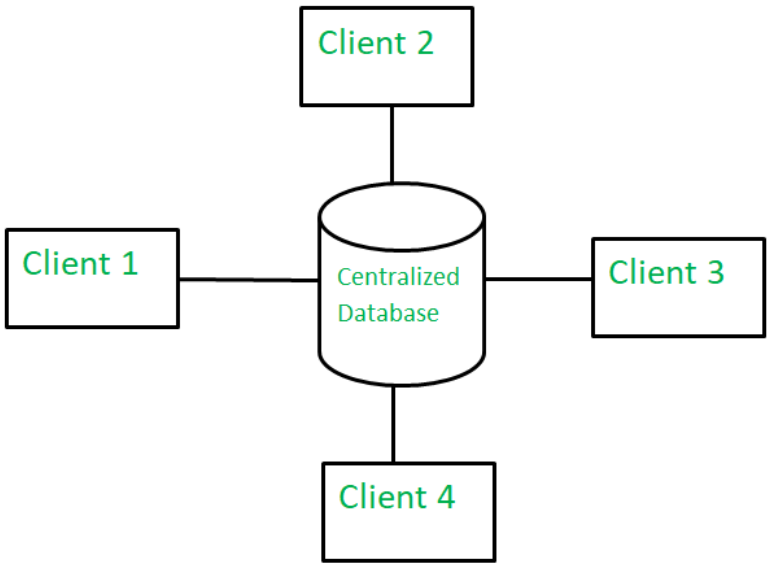
Lesson Learning Outcomes

- Describe the purpose of using Centralised relational database and Distributed relational database.
- List the advantages and disadvantages of Distributed DBMS.
- Explain the difference between Centralised and Distributed relational database.
- Explain the differences between horizontal and vertical fragmentation schemes

A database is a collection of related data where organizations used to store, manage and retrieve data. There are fundamentally two types of databases; centralized and distributed. A centralized database is a single database where multiple users can access it. On the other hand, the distributed database separates the database into multiple files at various locations in the network. In a nutshell, centralized database works with a single database file while a distributed database works with multiple database files.

10.1 Centralized vs Distributed DBMS

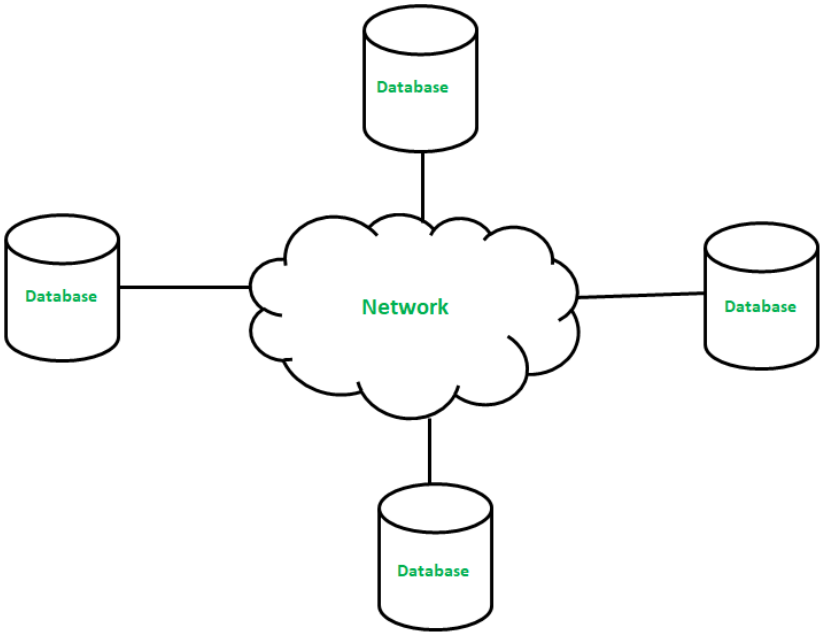
Centralized DBMS
A centralized database is basically a single database file stored and maintained at one location in the network. Multiple users can access this single database via an internet connection (LAN, WAN, etc). As there is only a single database file, it is easier to get a complete view of the data, and mainly used by institutions or organizations. It has a higher usage, easier to manage and control, update and take backups of data.



("Difference between centralized database and distributed database," 2020)

Distributed DBMS

A distributed database consists of two or more database files located at different sites in the network; database is split into multiple files. Thus, users can access the nearest database file without interfering with one another. This increase the speed of retrieving data as different users can access and manipulate data relevant to them at the nearest locations. Besides, different users can access the database without interfering with one another. To ensure data consistency, DBMS must periodically synchronise the scattered databases. Added advantage is that if one database fails, the system still runs as user can access other database files.



("Difference between centralized database and distributed database," 2020)

("Difference Between Centralized and Distributed Database - Pediaa.Com", 2020)

10.2 Advantages and Disadvantages of Distributed DBMS

The distribution DBMS has many conceivable advantages over traditional centralized database systems. Regrettably, there are also disadvantages which we will address here.

In brief, some advantages and disadvantages are:

Advantages

- This database can be easily expanded as data is already spread across different physical locations.
- The distributed database can easily be accessed from different networks.
- This database is more secure in comparison to centralized database.

Disadvantages

- This database is very costly and it is difficult to maintain because of its complexity.
- In this database, it is difficult to provide a uniform view to user since it is spread across different physical locations.

A more detailed discussion, in summary, would include:

TABLE 24.1 Summary of advantages and disadvantages of DDBMSs.

ADVANTAGES	DISADVANTAGES
Reflects organizational structure	Complexity
Improved shareability and local autonomy	Cost
Improved availability	Security
Improved reliability	Integrity control more difficult
Improved performance	Lack of standards
Economics	Lack of experience
Modular growth	Database design more complex
Integration	
Remaining competitive	

(Connolly T., Begg C., 2015, p. 796)

10.3 Fragmentation

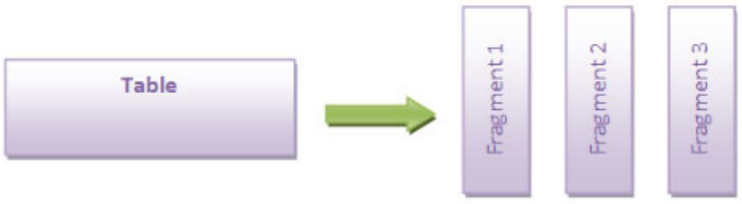
What is fragmentation?
The process of dividing the whole database into a smaller multiple chunks/ tables and storing them in different locations is called as fragmentation.

Fragmentation process should be carrried out in such a way that the reconstruction from the fragments whenever required to the original database is possible.

Fragmentation can be of three types: horizontal, vertical, and hybrid (combination of horizontal and vertical).

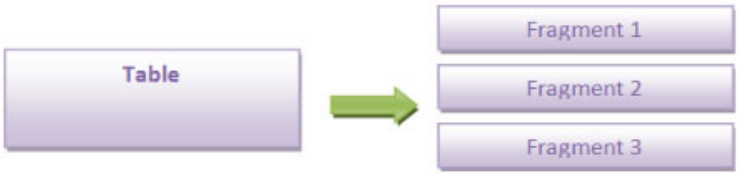
Vertical Fragmentation

Vertical fragmentation divides a relation (table) vertically into groups of columns i.e. subsets of tables. And in order to preserve reconstructiveness, each fragment should retain the primary key field(s). In addition, vertical fragmentation can enforce data privacy.



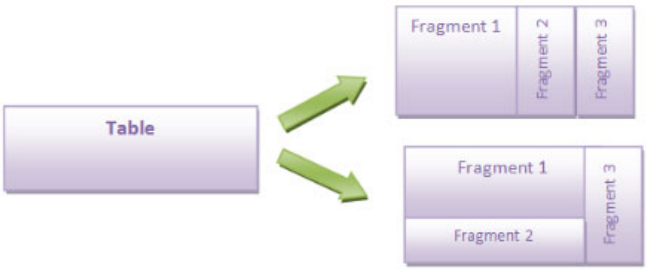
Horizontal Fragmentation

Horizontal fragmentation divides a relation (table) into the group of rows in accordance to values or user requirements to create subsets of tables. To maintain reconstructiveness, each horizontal fragment must have all columns of the original table.



Hybrid Fragmentation

Hybrid fragmentation can be achieved by performing horizontal and vertical partition together. This technique is flexible, however, reconstruction of the original table is often a challenging and expensive task.



There are some useful benefits to fragmentation:

- **Easy usage of Data:** It makes most frequently accessed set of data near to the user. Hence these data can be accessed easily as and when required by them.
- **Efficiency :** It in turn increases the efficiency of the query by reducing the size of the table to smaller subset and making them available with less network access time.
- **Security :** It provides security to the data. That means only valid and useful records will be available to the actual user. The DB near to the user will not have any unwanted data in their DB. It will contain only those informations, which are necessary for them.
- **Parallelism :** Fragmentation allows user to access the same table at the same time from different locations. Users at different locations will be accessing the same table in the DB at their location, seeing the data that are meant for them. If they are accessing the table at one location, then they have to wait for the locks to perform their transactions.
- **Reliability :** It increases the reliability of fetching the data. If the users are located at different locations accessing the single DB, then there will be huge network load. This will not guarantee that correct records are fetched and returned to the user. Accessing the fragment of data in the nearest DB will reduce the risk of data loss and correctness of data.
- **Balanced Storage :** Data will be distributed evenly among the databases in DDB.

Review Questions

1. Discuss why processes used to design a centralized relational database are not the same as those for a distributed relational database.

2. Discuss the advantages and disadvantages of a DDBMS.

3. One problem area with DDBMSs is that of distributed database design. Discuss the issues that have to be addressed with distributed database design.

4. What are the differences between horizontal and vertical fragmentation schemes?

(Connolly T., Begg C., 2015, p. 830)

References

Data fragmentation in DBMS. (2020, February 3). TutorialCup. https://www.tutorialcup.com/dbms/data-fragmentation.htm

Difference Between Centralized and Distributed Database - Pediaa.Com. (2020). Retrieved 26 October 2020, from https://pediaa.com/difference-between-centralized-and-distributed-database/#Centralized Database

Difference between Centralized Database and Distributed Database - GeeksforGeeks. (2020). Retrieved 26 October 2020, from https://www.geeksforgeeks.org/difference-between-centralized-database-and-distributed-database/

Thomas Connolly, Carolyn Begg (Sixth edition, Global edition) 2015, Database Systems. A Practical Approach to Design, Implementation, and Management , Pearson, USA.

Kaplan City Campus @ Wilkie Edge | 8 Wilkie Road, Level 2, Singapore 228095

Kaplan City Campus @ PoMo | 1 Selegie Rd, Level 7, Singapore 188306

6733 1877 info.sg@kaplan.com

KaplanSingapore

Introduction to Database Design and Development

Liew You Jie

CT0357623

Department of Information Technology, Kaplan

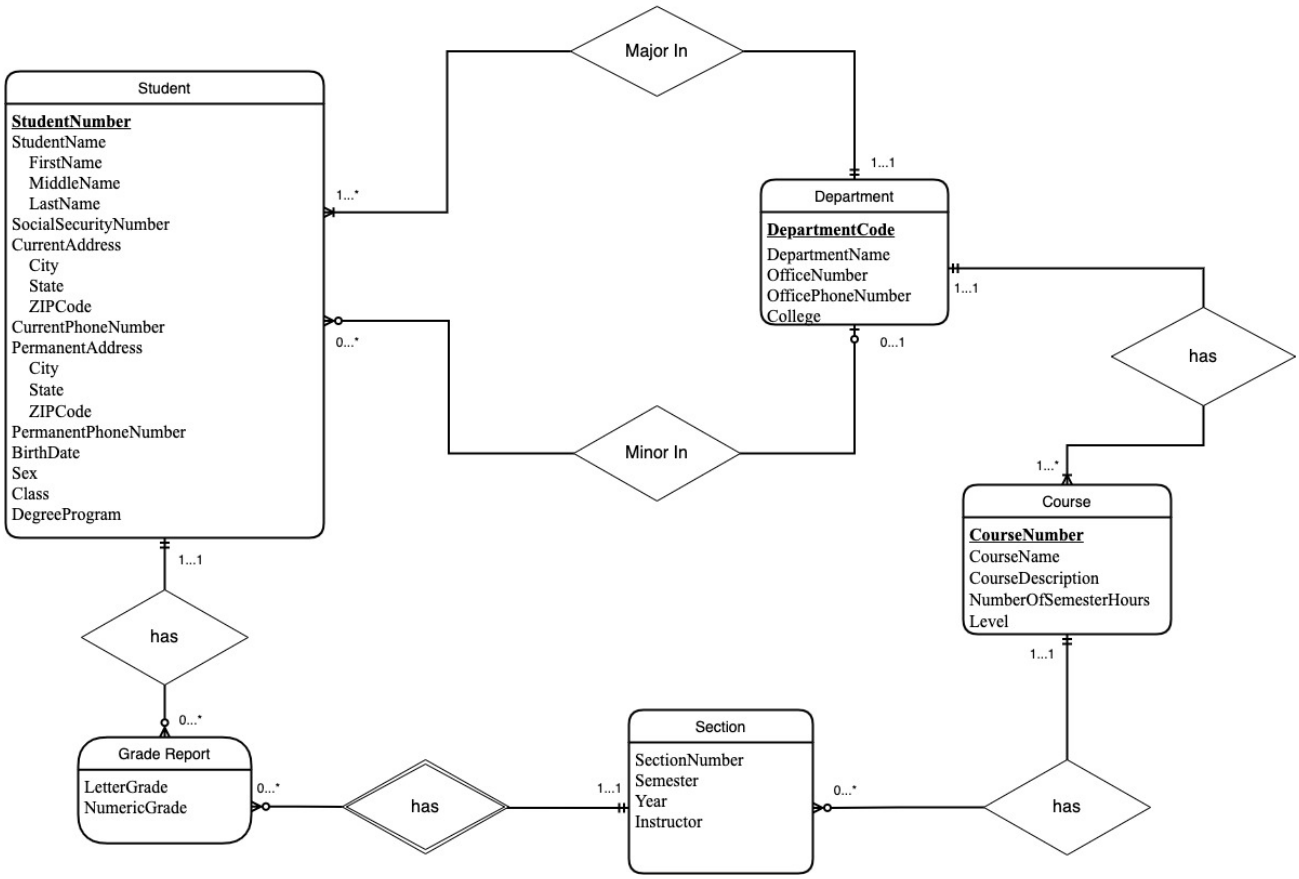
FTDip2106_IT71

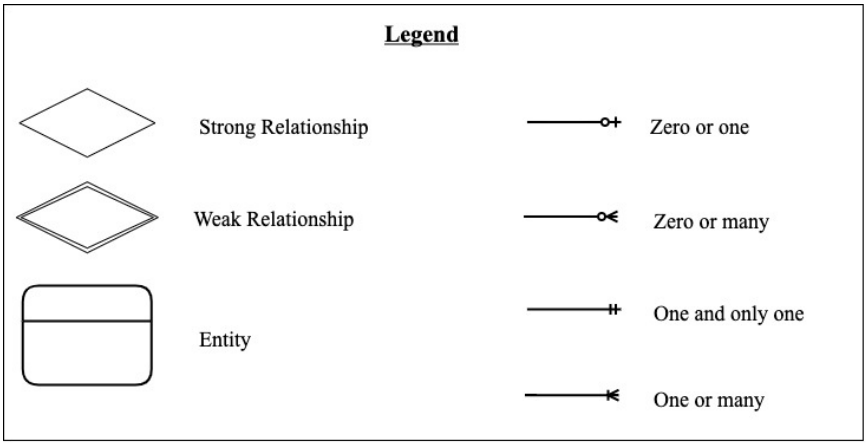
Mrs. Tan Mei Ling

Assignment Due Date 7 Dec 2021

Question 1

ER Diagram





Assumption

- StudentNumber uniquely identifies each student
- DepartmentCode uniquely identifies each department
- CourseNumber uniquely identifies each course
- Each student must have a major department
- Each student does not necessarily have a suitable minor department
- A student may not have a grade report (e.g. a student who has contracted an illness and drops the course for the semester)

Question 2

(i) This table is the order form in UNF. prodNumber, prodDesscription, prodQuantity and unitPrice have multi-value attributes in one column. They should be distinguished and stored as separate attributes.

UNF

orderNumber	orderDate	cusNumber	cusName	cusAddress	city	country	prodNumber	prodDesscription	prodQuantity	unitPrice
1234	11-4-98	9876	Billy	456 HighTower Street	Hong Kong	China	A123 B234 C345	Pencil Eraser Sharpener	100 200 5	$3.00 $1.50 $8.00

(ii) Table in 1NF:

Convert UNF to 1NF - Eliminate repeating group, define PK

Explain:

cusNumber can't be primary key because they are not bought only once.

cusName can't be primary key because sometime they can be duplicated.

OrderForm (orderNumber, cusNumber, cusName, cusAddress, city, country, orderDate, prodNumber, prodDesscription, prodQuantity, unitPrice)

Primary Key is (orderNumber, prodNumber) as it uniquely identifies the order form.

1NF

orderNumber	orderDate	cusNumber	cusName	cusAddress	city	country	prodNumber	prodDesscription	prodQuantity	unitPrice
1234	11-4-98	9876	Billy	456 HighTower Street	Hong Kong	China	A123	Pencil	100	$3.00
1234	11-4-98	9876	Billy	456 HighTower Street	Hong Kong	China	B234	Eraser	200	$1.50
1234	11-4-98	9876	Billy	456 HighTower Street	Hong Kong	China	C345	Sharpener	5	$8.00

(iii) Table in 2NF:

Convert 1NF to 2NF – Remove partial dependencies

Partial Dependencies:

orderNumber —> orderDate, cusNumber, cusName, cusAddress, city, country

prodNumber —> prodDesscription, unitPrice

Explain:

Partial dependency is non-key column that depends on part of the primary key.

orderNumber uniquely identifies an Order.

prodNumber uniquely identifies a Product.

prodDesscription can't be primary key because sometime they can be duplicated.

Order (orderNumber, orderDate, cusNumber, cusName, cusAddress, city, country)

Product (prodNumber, prodDesscription, unitPrice)

OrderForm (orderNumber, prodNumber, prodQuantity)

Foreign Key: orderNumber references Order(orderNumber)

Foreign Key: prodNumber references Product(prodNumber)

Order						
orderNumber	orderDate	cusNumber	cusName	cusAddress	City	Country
1234	11-Apr-98	9876	Billy	456 HighTower Street	Hong Kong	China

Product		
productNumber	prodDesscription	unitPrice
A123	Pencil	$3.00
B234	Eraser	$1.50
C345	Sharpener	$8.00

OrderForm		
orderNumber	prodNumber	prodQuantity
1234	A123	100
1234	B234	200
1234	C345	5

(iv) Table in 3NF:

Convert 2NF to 3NF – Remove transitive dependencies

Transitive Dependencies:

cusNumber—> cusName, cusAddress, city, country

A new table is create to stored Customer. cusNumber is introduced as PK for this table.

Explain:
Transitive Dependencies is non key column that depends on another non-key column.
cusNumber unique identifies a Customer.

Order (orderNumber, orderDate, cusNumber)

Foreign Key: cusNumber references Customer(cusNumber)

OrderForm (orderNumber, prodNumber, prodQuatity)

Foreign Key: orderNumber references Order(orderNumber)

Foreign Key: prodNumber references Product(prodNumber)

Customer (cusNumber, cusName, cusAddress, city, country)

Product (prodNumber, prodDesscription, unitPrice)

Order		
orderNumber	orderDate	cusNumber
1234	11-4-98	9876

OrderForm		
orderNumber	prodNumber	prodQuantity
1234	A123	100
1234	B234	200
1234	C345	5

Customer				
cusNumber	cusName	cusAddress	city	country
9876	Billy	456 HighTower Street	Hong Kong	China

Product		
prodNumber	prodDesscription	unitPrice
A123	Pencil	$3.00
B234	Eraser	$1.50
C345	Sharpener	$8.00

Question 3

i) –SQL statements to create the table

Order_Form

```
1  create database Stationery_Store;
2
3  use Stationery_Store;
4
5  create table Order_Form (
6      OrderNo int,
7      ProdNumber varchar(4),
8      ProdQuantity int CHECK (ProdQuantity between 1 and 1000),
9      Primary Key (OrderNo, ProdNumber),
10     Foreign Key (OrderNo) references Orders(OrderNo),
11     Foreign Key (ProdNumber) references Product(ProdNumber)
12 );
13
14 desc Order_Form;
15
```

SQL command

100% 5:15

Result Grid Filter Rows: Search Export:

Field	Type	Null	Key	Default	Extra
OrderNo	int	NO	PRI	NULL	
ProdNumber	varchar(4)	NO	PRI	NULL	
ProdQuantity	int	YES		NULL	

Result of running the SQL command

Orders

```
16 create table Orders (
17     OrderNo int,
18     OrderDate date NOT NULL,
19     CusNumber int,
20     Primary Key (OrderNo),
21     Foreign Key (CusNumber) references Customer(CusNumber)
22 );
23
24 desc Orders;
25
```

SQL command

100% 1:25

Result Grid Filter Rows: Search Export:

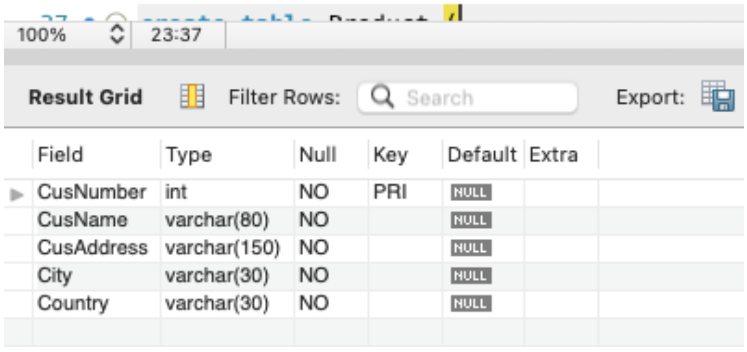
Field	Type	Null	Key	Default	Extra
OrderNo	int	NO	PRI	NULL	
OrderDate	date	NO		NULL	
CusNumber	int	YES	MUL	NULL	

Result of running the SQL command

Customer

```
26  create table Customer (
27      CusNumber int,
28      CusName varchar(80) NOT NULL,
29      CusAddress varchar(150) NOT NULL,
30      City varchar (30) NOT NULL,
31      Country varchar (30) NOT NULL,
32      Primary Key (CusNumber)
33  );
34
35  desc Customer;
36
```

SQL command



Result Grid | Filter Rows: Search | Export:

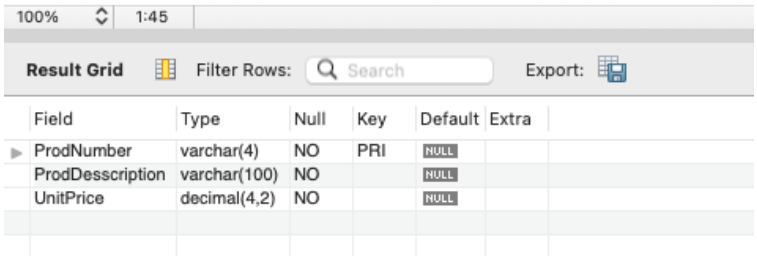
Field	Type	Null	Key	Default	Extra
CusNumber	int	NO	PRI	NULL	
CusName	varchar(80)	NO		NULL	
CusAddress	varchar(150)	NO		NULL	
City	varchar(30)	NO		NULL	
Country	varchar(30)	NO		NULL	

Result of running the SQL command

Product

```
37  create table Product (
38      ProdNumber varchar(4),
39      ProdDesscription varchar(100) NOT NULL,
40      UnitPrice decimal(4,2) NOT NULL,
41      Primary Key (ProdNumber)
42  );
43
44  desc Product;
45
```

SQL command



Result Grid | Filter Rows: Search | Export:

Field	Type	Null	Key	Default	Extra
ProdNumber	varchar(4)	NO	PRI	NULL	
ProdDesscription	varchar(100)	NO		NULL	
UnitPrice	decimal(4,2)	NO		NULL	

Result of running the SQL command

ii) –SQL statements to insert rows for each table

Order_Form

```
44 •  insert into Order_Form
45    values (1234, 'A123', 100),
46           (1234, 'B234', 200),
47           (1234, 'C345', 5),
48           (1357, 'G789', 150),
49           (2345, 'D456', 200),
50           (2345, 'H890', 10),
51           (2468, 'A123', 250),
52           (2468, 'F678', 200),
53           (2468, 'G789', NULL),
54           (3456, 'D456', 80),
55           (3456, 'D458', 140),
56           (3579, 'E567', 350),
57           (3579, 'G789', 180),
58           (4567, 'A123', 500),
59           (4567, 'C345', 90),
60           (4567, 'D465', 110),
61           (5678, 'E567', 780),
62           (5885, 'D465', 75),
63           (6789, 'F678', 20),
64           (6789, 'H890', NULL),
65           (7890, 'B234', 230),
66           (7890, 'F678', 100);
67
68 •  select * from Order_Form;
```

SQL command

```
68 •  select * from Order_Form;
69
```

100% 28:60

Result Grid Filter Rows: Search

OrderNo	ProdNumber	ProdQuantity
1234	A123	100
1234	B234	200
1234	C345	5
1357	G789	150
2345	D456	200
2345	H890	10
2468	A123	250
2468	F678	200
2468	G789	NULL
3456	D456	80
3456	D458	140
3579	E567	350
3579	G789	180
4567	A123	500
4567	C345	90
4567	D465	110
5678	E567	780
5885	D465	75
6789	F678	20
6789	H890	NULL
7890	B234	230
7890	F678	100
NULL	NULL	NULL

Result of running the SQL command

Orders

```
29 ●  insert into Orders
30    values (1234, '1998-04-11', '9876'),
31           (1357, '1998-07-28', '8921'),
32           (2345, '1999-11-12', '7689'),
33           (2468, '1999-12-06', '5321'),
34           (3456, '2000-01-08', '9898'),
35           (3579, '2001-07-23', '8921'),
36           (4567, '2002-05-09', '9899'),
37           (5678, '2009-05-09', '4523'),
38           (5885, '2006-06-13', '7890'),
39           (6789, '2008-11-22', '6534'),
40           (7890, '2021-11-21', '8878');
41
42 ●  select * from Orders;
```

SQL command

75% 1:43

Result Grid Filter Rows: Search

OrderNo	OrderDate	CusNumber
1234	1998-04-11	9876
1357	1998-07-28	8921
2345	1999-11-12	7689
2468	1999-12-06	5321
3456	2000-01-08	9898
3579	2001-07-23	8921
4567	2002-05-09	9899
5678	2009-05-09	4523
5885	2006-06-13	7890
6789	2008-11-22	6534
7890	2021-11-21	8878
NULL	NULL	NULL

Result of running the SQL command

Customer

```
15 ●  insert into Customer
16    values ('9876', 'Billy', '456 HighTower Street', 'Hong Kong', 'China'),
17           ('9899', 'James', '16 Sandiland Road', 'Sandiland', 'Singapore'),
18           ('8921', 'Alvin', '55 Merdeka Road', 'Kuala Lumpur', 'Malaysia'),
19           ('6534', 'Peter', '47 Ciburial Indah', 'Bandung', 'Indonesia'),
20           ('8878', 'Kelvin', '11 Phra Khanong District', 'Bangkok', 'Thailand'),
21           ('7689', 'Alvin', '258 Kim Keat Ave', 'Toa Payoh', 'Singapore'),
22           ('9898', 'Jackson', '92 Jalan Indahpura', 'Johor', 'Malaysia'),
23           ('7890', 'Joanne', '109 Wanquanhe Road', 'Beijing', 'China'),
24           ('4523', 'Mia', '51 Weijin South Road', 'Tianjin', 'China'),
25           ('5321', 'Olivia', '78 Yio Chun Kang Road', 'Serangoon', 'Singapore');
26
27 ●  select * from Customer;
```

SQL command

19:29

Result Grid Filter Rows: Search Edit: Export/Import:

CusNumber	CusName	CusAddress	City	Country
4523	Mia	51 Weijin South Road	Tianjin	China
5321	Olivia	78 Yio Chun Kang Road	Serangoon	Singapore
6534	Peter	47 Ciburial Indah	Bandung	Indonesia
7689	Alvin	258 Kim Keat Ave	Toa Payoh	Singapore
7890	Joanne	109 Wanquanhe Road	Beijing	China
8878	Kelvin	11 Phra Khanong District	Bangkok	Thailand
8921	Alvin	55 Merdeka Road	Kuala Lumpur	Malaysia
9876	Billy	456 HighTower Street	Hong Kong	China
9898	Jackson	92 Jalan Indahpura	Johor	Malaysia
9899	James	16 Sandiland Road	Sandiland	Singapore
NULL	NULL	NULL	NULL	NULL

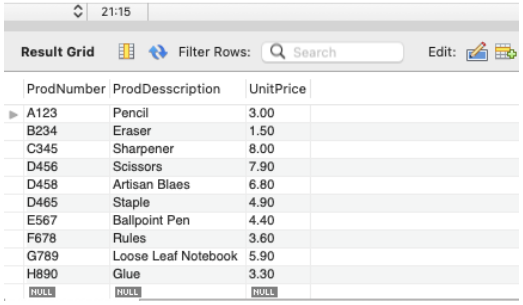
Result of running the SQL command

Product

```sql
insert into Product
values ('A123', 'Pencil',3.00),
       ('B234', 'Eraser', 1.50),
       ('C345', 'Sharpener', 8.00),
       ('D456', 'Scissors', 7.90),
       ('D458', 'Artisan Blaes', 6.80),
       ('D465', 'Staple', 4.90),
       ('E567', 'Ballpoint Pen', 4.40),
       ('F678', 'Rules', 3.60),
       ('G789', 'Loose Leaf Notebook', 5.90),
       ('H890', 'Glue', 3.30);

select * from Product;
```

SQL command



Result Grid Filter Rows: Search Edit:

ProdNumber	ProdDesscription	UnitPrice
A123	Pencil	3.00
B234	Eraser	1.50
C345	Sharpener	8.00
D456	Scissors	7.90
D458	Artisan Blaes	6.80
D465	Staple	4.90
E567	Ballpoint Pen	4.40
F678	Rules	3.60
G789	Loose Leaf Notebook	5.90
H890	Glue	3.30
NULL	NULL	NULL

Result of running the SQL command

iii) SQL Queries using SELECT

a) Restriction and projection:

The Restriction constraint is used to specify a set of conditions for the data in a table.
The Projection function is to select the specific attribute from table.
Display CusNumber and CusName, where the Customer is stay in Singapore.

```
1    -- Restriction and Projection
2 •  select CusNumber, CusName from Customer
3    where Country = 'Singapore';
4
5
```

100% 32:6

Result Grid Filter Rows: Search Edit:

CusNumber	CusName
5321	Olivia
7689	Alvin
9899	James
NULL	NULL

SQL command

Result of running the SQL command

b) Aliases:

This function is used to temporarily rename the attribute when it is displayed.
Display CusNumber, alias to Customer_ID from the table in Customer.

```
5    -- Aliases
6 •  select CusNumber as Customer_ID
7    from Customer;
8
```

100% 21:11

Result Grid Filter Rows: Search

Customer_ID
4523
5321
6534
7689
7890
8878
8921
9876
9898
9899

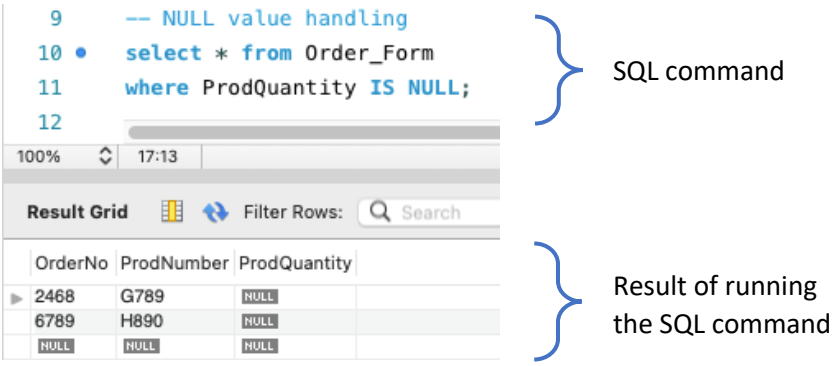
SQL command

Result of running the SQL command

c) NULL value handling:

This function is to replace missing data in the table.

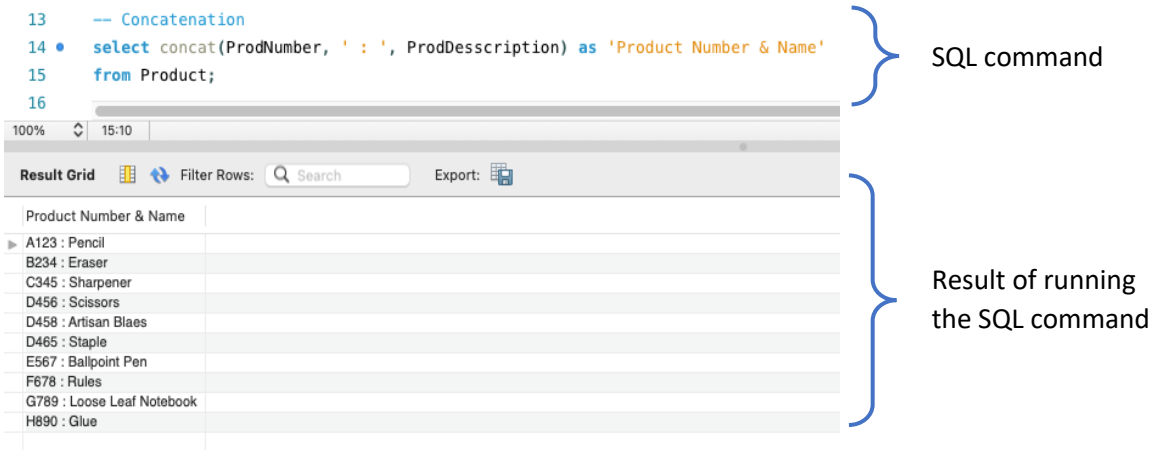
Display all column, when ProdQuantity is null in the table Order_Form.



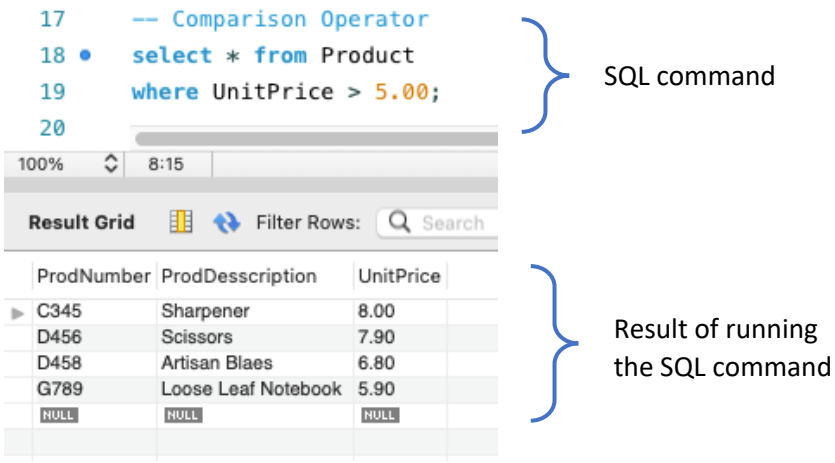
d) Concatenation:

This function combines two different attributes together.

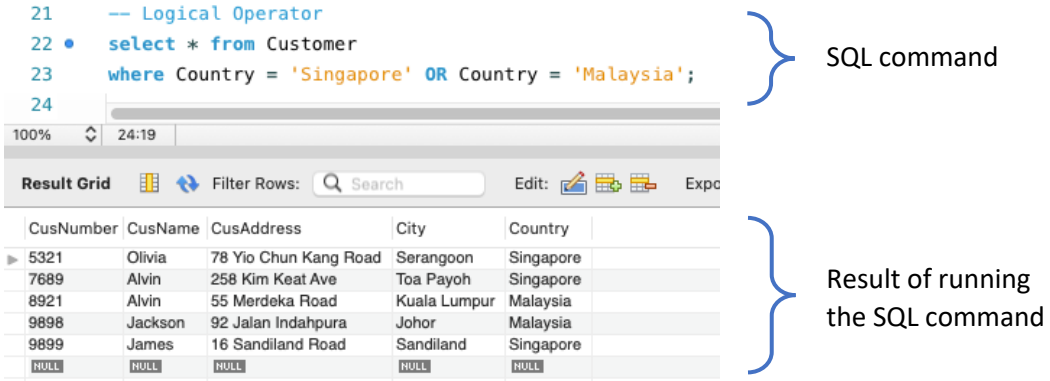
My select statement picks out ProdNumber and ProdDescription from the Product table.



e) Comparison Operator:
This function is a mathematical function used to compare 2 or more separate values.
Display all column, specifying UnitPrice over 5.00 from the Product table.



f) Logical Operator:
This function determines the truth or falsity of an expression.
Display all column, find Country is Singapore OR Malaysia from Customer table.



g) Sorting:
This function is to sort a set of data to a specified order.
Display all column, sorted by CusNumber from Customer table.

```
25    -- sorting
26 •  select * from Customer
27    order by CusNumber;
28
```

SQL command

100% 28:23

Result Grid | Filter Rows: Search | Edit:

CusNumber	CusName	CusAddress	City	Country
4523	Mia	51 Weijin South Road	Tianjin	China
5321	Olivia	78 Yio Chun Kang Road	Serangoon	Singapore
6534	Peter	47 Ciburial Indah	Bandung	Indonesia
7689	Alvin	258 Kim Keat Ave	Toa Payoh	Singapore
7890	Joanne	109 Wanquanhe Road	Beijing	China
8878	Kelvin	11 Phra Khanong District	Bangkok	Thailand
8921	Alvin	55 Merdeka Road	Kuala Lumpur	Malaysia
9876	Billy	456 HighTower Street	Hong Kong	China
9898	Jackson	92 Jalan Indahpura	Johor	Malaysia
9899	James	16 Sandiland Road	Sandiland	Singapore
NULL	NULL	NULL	NULL	NULL

Result of running the SQL command

h) Function()Group function & Single Row function:
This function groups one or more columns against the result set and uses the aggregate function to calculate the result.
Display Country (upper) and CusNumber (count), with group by Country from Customer table.

```
29    -- Function()Group function & Single Row Function
30 •  select upper(Country), count(CusNumber)
31    from Customer
32    group by Country;
33
```

SQL command

100% 12:25

Result Grid | Filter Rows: Search | Export:

upper(Count...	count(CusNumb...
CHINA	3
SINGAPORE	3
INDONESIA	1
THAILAND	1
MALAYSIA	2

Result of running the SQL command

i) Joins:

Inner Join the attributes of two different tables into a single table if there is a match between the columns.

Display OrderNo and CusName from Orders table, CusName will be gotten from Customer table by refering CusNumber.

```
34    -- Joins
35 •  select Orders.OrderNo, Customer.CusName
36    from Orders
37    inner join Customer
38    On Orders.CusNumber = Customer.CusNumber;
39
```

SQL command

100% 14:40

Result Grid | Filter Rows: Search | Export:

OrderNo	CusName
5678	Mia
2468	Olivia
6789	Peter
2345	Alvin
5885	Joanne
7890	Kelvin
1357	Alvin
3579	Alvin
1234	Billy
3456	Jackson
4567	James

Result of running the SQL command

j) Subqueries:

Subqueries are used to return the required data for the main query, or to further restrict the retrieved data.

Display all column, the highest UnitPrice from the table of Product.

```
40    -- Subqueries
41 •  select *
42    from Product
43    where UnitPrice = (select max(UnitPrice)
44                       from Product);
```

SQL command

100% 20:37

Result Grid | Filter Rows: Search | Edit:

ProdNumber	ProdDesscripti...	UnitPrice
C345	Sharpener	8.00
NULL	NULL	NULL

Result of running the SQL command

KAPLAN

Introduction to Database Design and Development

Topic 01
Introduction to Databases

Learning Outcomes

- Describe the difference between data and information.
- Explain the concept of a database and the different types of databases.
- Explain the importance of database design.
- Describe how modern database systems evolved.
- Explain the differences between a database system and a file system.
- Describe how a DBMS functions within the database system.

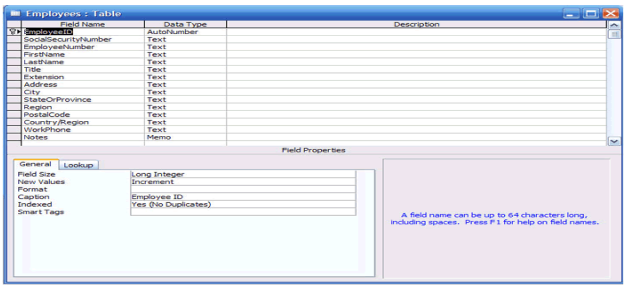
Data and Information

- ***Data*** can be classified as raw facts or the building block of information.
- These are usually unprocessed information.
- Data can exist in a variety of forms such as numbers or text on pieces of paper and bytes stored in electronic memory, or as facts stored in a person's mind.

Data and Information

- ***Metadata***

Describes how and when and by whom a particular set of data was collected, and how the data is formatted.



Data and Information

- ***Information*** can be classified as processed data.
- Information should be meaningful, relevant, accurate and timely.
- These are essential ingredients to good decision-making, which is the key to organizational success.

Figure 1-1a Data in Context

Class Roster

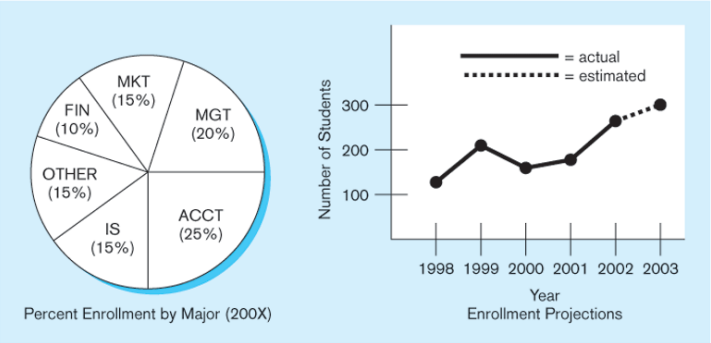
Course: MGT 500 Business Policy Semester: Spring 200X

Section: 2

Name	ID	Major	GPA
Baker, Kenneth D.	324917628	MGT	2.9
Doyle, Joan E.	476193248	MKT	3.4
Finkle, Clive R.	548429344	PRM	2.8
Lewis, John C.	551742186	MGT	3.7
McFerran, Debra R.	409723145	IS	2.9
Sisneros, Michael	392416582	ACCT	3.3

Context helps users understand data

Figure 1-1b Converting data to information - Summarized data



Graphical displays turn data into useful information that managers can use for decision making and interpretation

7
22/2/2021
KAPLAN
7

Table 1-1 Example Metadata for Class Roster

Data Item			*Value*			
Name	**Type**	**Length**	**Min**	**Max**	**Description**	**Source**
Course	Alphanumeric	30			Course ID and name	Academic Unit
Section	Integer	1	1	9	Section number	Registrar
Semester	Alphanumeric	10			Semester and year	Registrar
Name	Alphanumeric	30			Student name	Student IS
ID	Integer	9			Student ID (SSN)	Student IS
Major	Alphanumeric	4			Student major	Student IS
GPA	Decimal	3	0.0	4.0	Student grade point average	Academic Unit

Descriptions of the properties or characteristics of the data, including data types, field sizes, allowable values, and data context

8
22/2/2021
KAPLAN
8

WHAT IS A DATABASE?

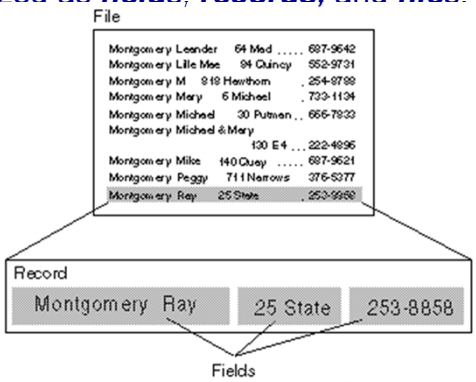
- A collection of information organised in such a way that a computer program can quickly select desired pieces of data.
- Databases are organised as ***fields***, ***records***, and ***files***.

9

KAPLAN

DATABASE ORGANIZATION

- A collection of information organized in such a way that a computer program can quickly select desired pieces of data.
- Databases are organized as ***fields***, ***records***, and ***files***.



10

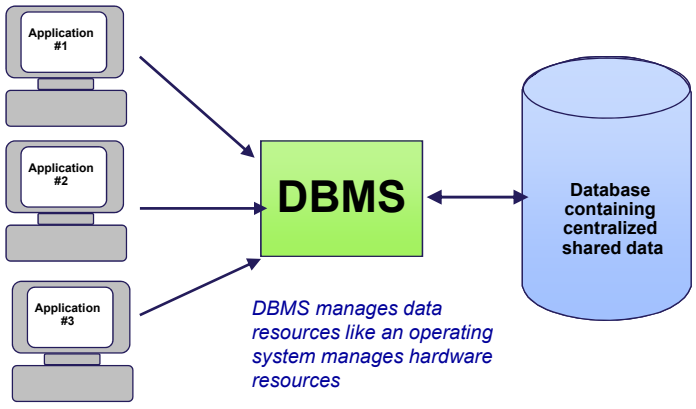
KAPLAN

DATABASE MANAGEMENT SYSTEM

- To access information from a database, you need a ***database management system***
- A ***DBMS*** is a collection of programs that enables you to store, modify, and extract information from a database.
- DBMSs' makes data management more effective

KAPLAN

DATABASE MANAGEMENT SYSTEM



KAPLAN

DATABASE DESIGN

- Database design refers to the design of the database structure that will be used to store and manage data.
- It does NOT refer to the designing the DBMS software.
- A well- designed database will minimize the problem *of data-redundancy*.
- A poorly designed database tends to generate errors.

HISTORY OF DATABASES

Early Manual System { Before-1950s}

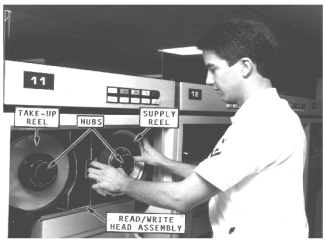
- Data was stored as paper records.
- Lot of manpower involved.
- Lot of time was wasted. e.g. when searching
- Therefore inefficient.

MANUAL FILING SYSTEMS

- Manual filing systems are usually made up of a collection of file folders kept in filing cabinet.
- Suitable when the amounts of data to be managed were small and with few reporting requirements.
- Very hard to manage in cases with large amounts of data
- Manual filing systems.

REVOLUTION BEGAN

1950s and early 1960s:
- Data processing using magnetic tapes for storage
- Tapes provide only sequential access
- Punched cards for input



HISTORY OF DATABASES

Late 1960s and 1970s:

-Hard disks allow direct access to data

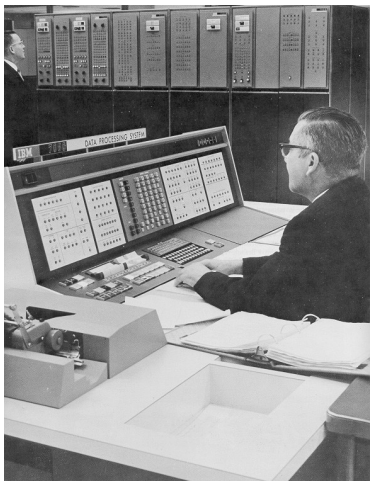
-Data stored in files

-Known as File Processing System

COMPUTERISATION

- Conversion from a manual system to a computerised file system challenging.
- Data processing specialists were needed to perform these tasks.
- Created file structures, wrote software, and designed application programs.
- There were separate files for each department

DATABASE SYSTEMS: THEN



COMPUTERISATION

- As systems grew larger and more complex, demand for programming skills also grew.
- File-based programs were written in third generation programming languages.
- These required the programmer to specify task and *how* it must be done.
- This was time-consuming that required high skill levels.

COMPUTERISATION

- New programs had to be written every time a new type of query was needed.
- This is known as ***structural dependence***.
- Modifications often led to errors requiring additional time and effort for debugging.
- ***Data-redundancy*** was a common phenomenon in file-based systems.
- May cause ***data inconsistency***, lack of ***data integrity*** and ***data anomalies***

DISADVANTAGES OF FILE PROCESSING

- Program-Data Dependence
  -All programs maintain metadata for each file they use
- Duplication of Data
  -Different systems/programmes have separate copies of the same data
- Limited Data Sharing
  -No centralized control of data
- Lengthy Development Times
  -Programmers must design their own file formats
- Excessive Programme Maintenance
  -80% of information systems budget

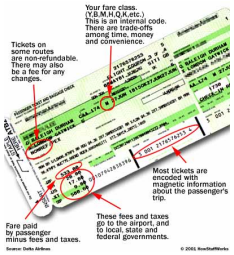
DATABASE SYSTEMS: TODAY

From Friendster.com on-line tour

OTHER DATABASES YOU MAY USE

amazon.com

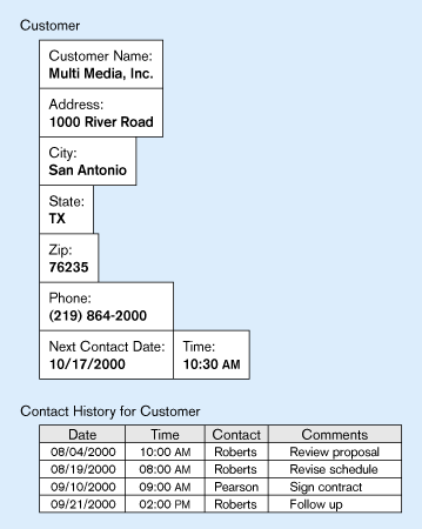
ebay



CLASSIFICATION OF DATABASE

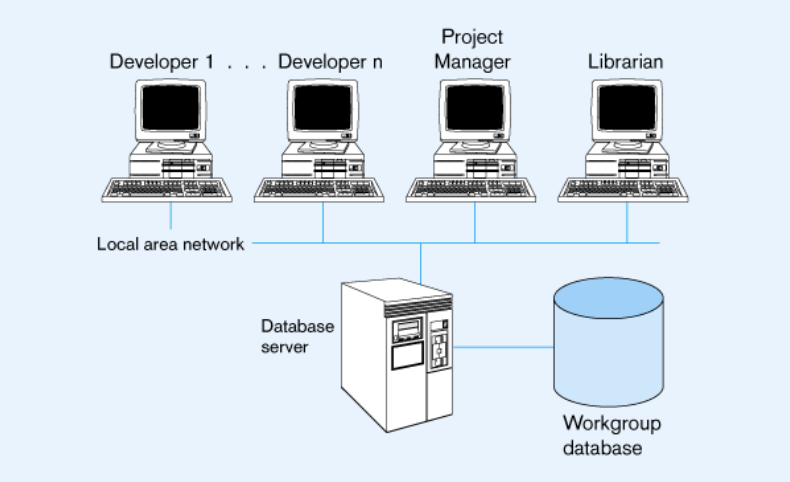
- Personal Database – standalone desktop database
- Workgroup Database – local area network (<25 users)
- Department Database – local area network (25-100 users)
- Enterprise Database – wide-area network (hundreds or thousands of users)

Figure 1-7 Typical data from a personal computer database



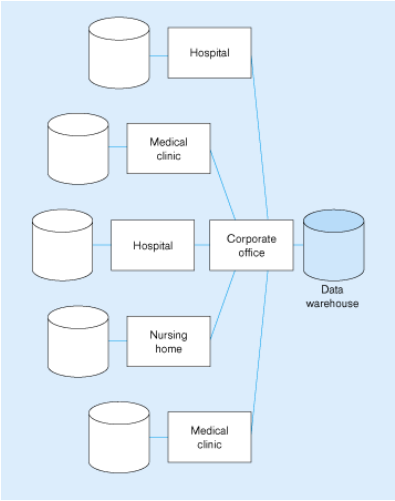
Date	Time	Contact	Comments
08/04/2000	10:00 AM	Roberts	Review proposal
08/19/2000	08:00 AM	Roberts	Revise schedule
09/10/2000	09:00 AM	Pearson	Sign contract
09/21/2000	02:00 PM	Roberts	Follow up

Figure 1-8 Workgroup database with local area network



27
22/2/2021
KAPLAN
27

Figure 1-9 An enterprise data warehouse



28
22/2/2021
KAPLAN
28

OTHER CLASSIFICATIONS OF DATABASES

- ***A centralised database*** is one that supports data located at a single site.
- A ***distributed database*** is a single logical database that is spread physically across computers in multiple locations
- The ***transactional database*** is used to process transactions
- ***A data-warehouse database*** focuses on storage of data that will be used for information generation for the purpose of decision-making.

WHY STUDY DATABASES?

- Databases are useful
  - -Many computing applications deal with large amounts of information
  - -Database systems give a set of tools for storing, searching and managing this information
- Databases in IT
  - -Databases are a 'core topic' in computer science and IT
  - -Basic concepts and skills with database systems are part of the skill set you will be assumed to have as an IT graduate

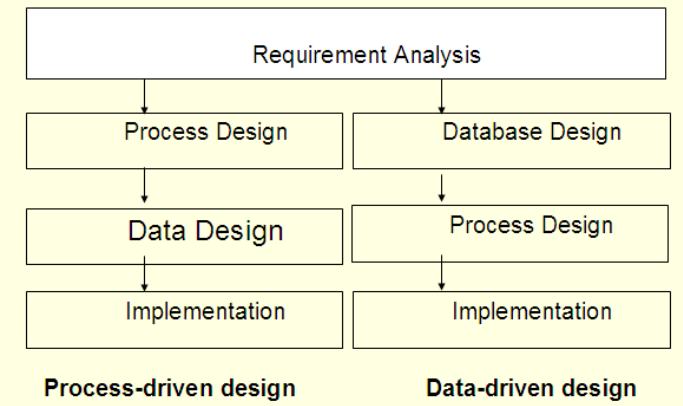
DATABASE vs. FILE SYSTEM

- The file-based systems use many separate files but a database system stores all data in a single logical unit called the repository.
- The physical data can be physically distributed among multiple data storage facilities but it acts as a single unit.
- File Systems focuses on the data processing needs of individual departments.
- Database Systems emphasizes the integration and sharing of data.
- File-based system is known as ***process-driven system*** and a database system is known ***data-driven system.***

31

KAPLAN

FILE SYSTEM vs. DATABASE



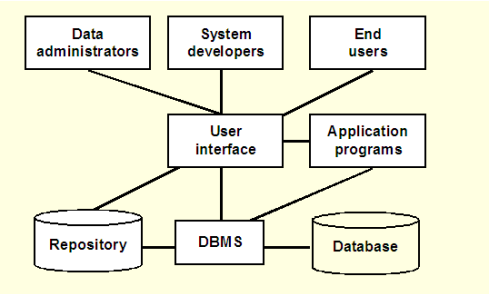
32

KAPLAN

COMPONENTS OF DATABASE ENVIRONMENT

- **Repository**
  - A centralised knowledge base containing all data definitions, screen and report formats and definitions of other organisations and system components

- **Database management system (DBMS) -** Commercial software system used to create, maintain and provide controlled access to the database and repository.



COMPONENTS OF DATABASE ENVIRONMENT

- **Database**

  A shared collection of logically related data, designed to meet the information needs of multiple users in an organisation.

- **Application programs**

  Computer programs that are used to create and maintain the database and provide information to users.

- **User interface**

  Languages, menu and other facilities by which users interact with various system components.

- **Data administrators**

  Persons who are responsible for the overall information resources of an organisation.

COMPONENTS OF DATABASE ENVIRONMENT

- **System developers**
  Persons such as system analysts and programmers who design new application programs.

- **End users**
  Persons throughout the organisation who add, delete and modify data in the database and who request or receive information from it.

35

KAPLAN

DBMS FUNCTIONS

- **Data dictionary management** – The DBMS stores the definitions of all data elements and their relationships in the data dictionary. The DBMS uses the dictionary to look up the required data component structure and relationships.
- **Data Storage Management** – The DBMS creates and manages the complex structures required. It is also important database performance tuning.
- **Data transformation and presentation** – The DBMS transforms the input data to conform to the data structures that are required to store the data. This maintains data independence.

36

KAPLAN

DBMS FUNCTIONS

- **Security Management** – The DMBS creates a security system that enforces user security and data privacy within the database. Rules determine which users can access the database, which data items each user may access and which data operations the user may perform.
- **Multi-user access control** – The DBMS manages and provides the environment so that multiple users can access the data.
- **Back-up recovery and management** – The DBMS provides back-up and recovery procedures to ensure data safety and integrity.

37 22/2/2021 KAPLAN

DBMS FUNCTIONS

- **Data Integrity management** – The DBMS promotes and enforces rules to eliminate data- integrity problems, thus minimizing data-redundancy and maintaining data consistency.
- **Database access languages**– The DBMS provides data access through a query language. The Query language has two components: the DDL and the DML. These have been explained above.
- **Database communications interfaces** – Modern DBMS allow databases to accept end-user requests within a computer network (includes WANs such as the Internet).

38 22/2/2021 KAPLAN

MCQ's

Database is collection of ____.

A Modules
B None of these
C Programs
D Data

_____ is collection of interrelated data and set of program to access them.

A Database
B Data Structure
C Programming language
D Database Management System

39

KAPLAN

DBMS should provide following feature(s) _____.

A Authorized access
B Safety of the information stored
C Protect data from system crash
D All of these

Which of the following is considered as DBMS ?

A Access
B Foxpro
C Oracle
D All of these

Before use of DBMS information was stored using _____.

A Cloud Storage
B None of these
C Data System
D File Management System

40

KAPLAN

The DBMS acts as an interface between what two components of an database system ?

A Database and SQL
B Database Application and Database
C Data and Database
D Database and User

Long form of DBA is _____.

A Database Administrator
B Database Application
C None of these
D Database Admin

A database is a complex type of _____.

A Manager
B None of these
C Data Structure
D Application

41

KAPLAN

Duplication of data at several places is called as _____.

A Data Inconsistency
B Data Redundancy
C Atomicity Problem
D Data Isolation

If in redundant file common fields are not matching then it results in _____.

A Data Isolation
B Data Integrity Problem
C Data Inconsistency
D Data Redundancy

Data Isolation caused due to _____ in traditional file system.

A Complex Data
B Duplicate Data
C Atomic Data
D Scattering of Data

42

22/2/2021

KAPLAN

If DBA modify the structure of the data record then this modification do not affect other application is called as _____.

A Data Integrity
B Data Independance
C Data Security
D Data Isolation

A main purpose of DBMS is to provide _____ view of data to user.

A None of these
B Partial
C Complete
D Abstract

Which of the following is not a level of data abstraction?

A Physical Level
B Critical Level
C Logical Level
D External Level

KAPLAN

Introduction to Database Design and Development

Topic 02
Database Modelling

LEARNING OUTCOMES

Upon successful completion of this module, the student should be able to:

1. Explain the purpose of data modelling
2. Describe the approaches to Database Design
3. Describe the Three-Level ANSI-SPARC Architecture
4. Explain the terminology and structural concepts of the relational model
5. Demonstrate an understanding of relational database principles and theory
6. Explain the Relational Integrity Constraints

DATA AND INFORMATION

- ***Data*** can be classified as raw facts or the building block of information.
- These are usually unprocessed information.
- Data can exist in a variety of forms such as numbers or text on pieces of paper and bytes stored in electronic memory, or as facts stored in a person's mind.

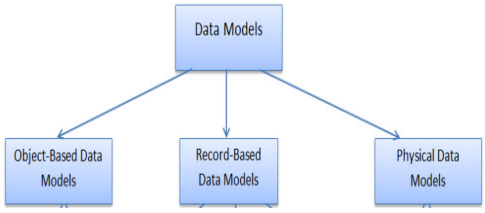
DATA MODEL

"A **data model** is an abstract model that organizes elements of data and standardizes how they relate to one another and to the properties of real-world entities."
https://en.wikipedia.org/wiki/Data_model

2.1 Types of Data Models

They fall into 3 broad categories:

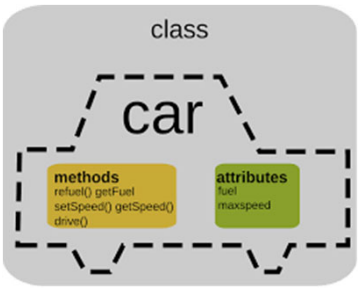
- object-based,
- record-based, and
- physical data models.



OBJECT-BASED DATA MODELS

- Encapsulates both state and behaviour.

- Object - a person, place, thing, concept,
- Attribute - Name, Address, Age and Phone number.
- Relationship - association between entities.



5 24/2/2021 KAPLAN 5

RECORD-BASED DATA MODELS

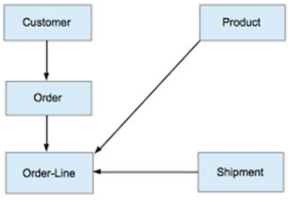
3 types of record-based logical data model:

the Relational Data model (tables)

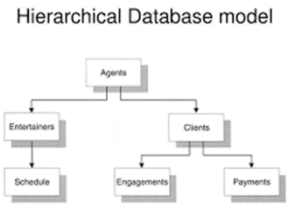
Branch

branchNo	street	city	postCode
B005	22 Deer Rd	London	SW1 4EH
B007	16 Argyll St	Aberdeen	AB2 3SU
B003	163 Main St	Glasgow	G11 9QX
B004	32 Manse Rd	Bristol	BS99 1NZ
B002	56 Clover Dr	London	NW10 6EU

the Network Data model (objects and relationships)



the Hierarchical Data model (data are organized into a tree-like structure)



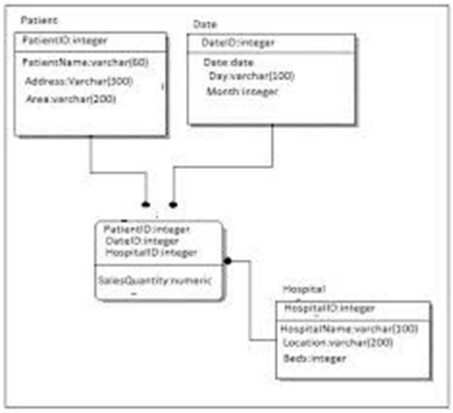
6 24/2/2021 KAPLAN 6

PHYSICAL DATA MODELS

Physical data model design

Convert :

- Entities into tables and columns.
- Relationships into foreign keys.
- Attributes into columns.



7 24/2/2021 KAPLAN 7

2.2 COMPARISON OF DATA MODELS

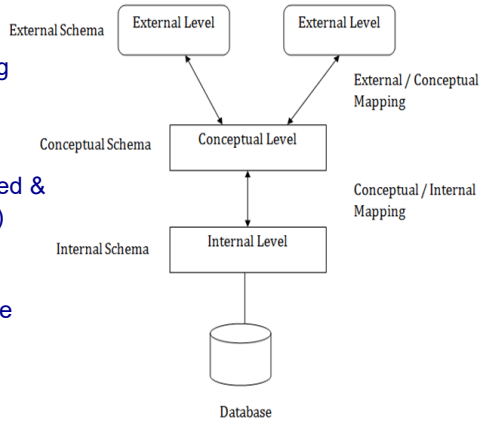
Hierarchical Data Model	Network Data Model	Relational Data Model
1. Relationship between records is of the parent child type.	1. Relationship between records is expressed in the form of pointers or links.	1.Relationship between records is represented by a relation that contains a key for each record involved in the relationship.
2. Many to many relationship cannot be expressed in this model	2. Many to many relationship can also be implemented in this model	2. Many to many relationship can be easily implemented.
3. It is a simple, straightforward and natural method of implementing record relationships.	3. Record relationship implementation is quite complex due to the use of pointers.	3. Relationship implementation is very easy through the use of a key or composite key field.
4. This type of model is useful only when there is some hierarchical character in the database.	4. Network model is useful for representing such records which have many to many relationships.	4. Relational model is useful for representing most of the real world objects and relationships among them.
5. Searching for a record is very difficult since one can retrieve a child only after going through its parent record.	5. Searching for a record is easy since there are multiple access paths to a data element.	5. A unique, indexed key field is used to search for a data element.

8 KAPLAN

2.3 THREE-LEVEL ANSI-SPARC ARCHITECTURE

- External Schema - describes the database part that a particular user group view is interested in and hides the remaining database
- Conceptual Schema – describes what data are to be stored & relationship among those data (Entities, Attributes, Relationships)

-Internal level - describes the physical storage structure of the database. (data compression and data encryption techniques)



9

KAPLAN

2.4 TERMINOLOGY OF RELATIONAL MODEL

- Tables / Relation (A table has rows and columns. Rows represent records and columns represent attributes)
- Attribute (named column, the properties that define a relation. e.g., Empl**_No.**)
- Domain (A domain is the set of allowable values for an attribute. E.g , an integer number data type field can only allow whole numbers to be entered and not others)
- Tuple (A single row of a table is known as tuple which contains a single record)

10

24/2/2021

KAPLAN

10

2.4 TERMINOLOGY OF RELATIONAL MODEL

- Degree (The number of attributes in the table)
- Cardinality (The total number of rows (tuple) present in the table.)
- Relational database (It consists of relations that are normalized and structured)

2.5 PROPERTIES OF VALID REASON

- the relation has a name that is distinct from all other relation names in the relational schema;
- each cell of the relation contains exactly one atomic (single) value;
- each attribute has a distinct name;
- the values of an attribute are all from the same domain;
- each tuple is distinct; there are no duplicate tuples;
- the order of attributes has no significance;
- the order of tuples has no significance, theoretically.

2.6 TYPES OF KEYS

- **Superkey** (set of attributes uniquely identifies a tuple. E.g STUD_NO, STUD_NAME)
- **Candidate key** (The minimal set of attribute which can uniquely identify a tuple. May be several candidate keys for a relation)
- **Primary key** (More than one candidate key but one can be chosen as the primary key. The candidate keys that are not selected are called alternate keys)

2.6 TYPES OF KEYS

- **Alternate Key** (Candidate key other than the primary key is called an alternate key)
- **Foreign key**
  - -If an attribute can only take the values which are present as values of some other attribute, it will refer to as a foreign key.
  - -A foreign key is a field that matches the primary key column of another table.

2.7 INTEGRITY CONSTRAINTS

- Must be present for a valid relation
- 3 main categories:
  - Domain constraints
  - Key constraints
  - Referential integrity constraints

Domain Constraints

- Violated if an attribute value is not of the appropriate data type.
- Standard data types include integers, real numbers, characters, Booleans, variable length strings, etc.

2.7 INTEGRITY CONSTRAINTS

Key Constraints

- Uniquely identify a tuple in a relation. Primary key.
- A key has two properties:
  - It should be unique for all tuples.
  - It can't have NULL values.

Referential integrity constraints

- Work on the concept of Foreign Keys where it is referred to in other relationships.

Review Questions

1. Explain the concept of database schema and discuss the three types of schema in a database.
2. What is a data model? Discuss the main types of data model.
3. Discuss the function and importance of conceptual modelling.
4. Define the term “database integrity”.

Review Questions

1. Explain the concept of database schema and discuss the three types of schema in a database.
2. What is a data model? Discuss the main types of data model.
3. Discuss the function and importance of conceptual modelling.
4. Define the term “database integrity”.

Review Questions

5. Discuss each of the following concepts in the context of the relational data model:
   (a) relation
   (b) attribute
   (c) domain
6. Discuss the properties of a relation.
7. Discuss the differences between the candidate keys and the primary key of a relation. Explain what is meant by a foreign key. How do foreign keys of relations relate to candidate keys? Give examples to illustrate your answer.

Exercises

1. A database approach uses different data models. Common database models include the relational model, the network model and the hierarchical model. Which data model should be chosen under which circumstances and why?

REFERENCES

Thomas Connolly, Carolyn Begg (Sixth edition, Global edition) 2015, Database Systems. A Practical Approach to Design, Implementation, and Management , Pearson, USA.

KAPLAN

Introduction to Database Design and Development

Topic 03

Conceptual Modelling – Entity Relationship Model

LEARNING OUTCOMES

Upon successful completion of this module, the student should be able to:

1. Explain the purpose and importance of conceptual modelling
2. Describe how Entity-Relationship modelling is use in database design
3. Demonstrate practical skills in data modelling using entity-relationship modelling (ER-Model).

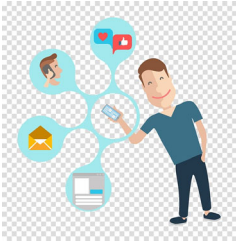
2

KAPLAN

DATA MODEL

"A conceptual model's primary objective is to convey the fundamental principles and basic functionality of the system which it represents. Also, a conceptual model must be developed in such a way as to provide an easily understood system interpretation for the model's users."

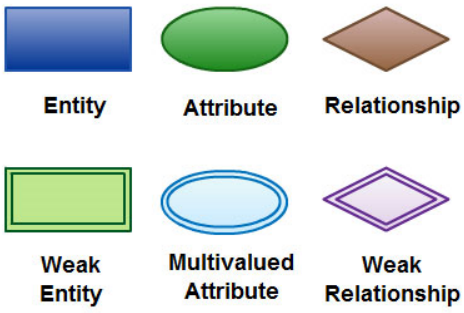
Source: https://en.wikipedia.org/wiki/Conceptual_model



ENTITY RELATIONSHIP MODEL

Entity Relationship Model (ER) diagram has three main components:

1. Entity
2. Relationship
3. Attribute

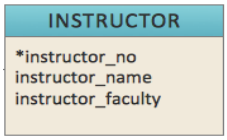


3.1 ENTITY TYPES

An *entity* type represents a group of "objects" of same properties having independent existence.

An entity can be:
- An object with physical/real existence (e.g., table, student, house)
- An object with conceptual/abstract existence (e.g., job, course, position)

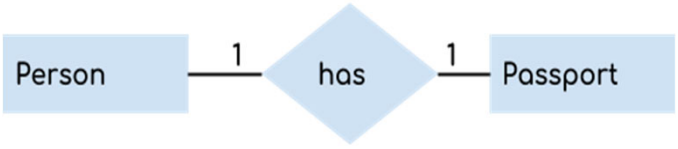
Entity : shown as Rectangle. Name on top, attributes listed in the body.



3.2 RELATIONSHIP TYPES

Relationship is an association among two or more entities. It is a connection that holds the tables together.

E.g., a person has only one passport and a passport is given to one person.

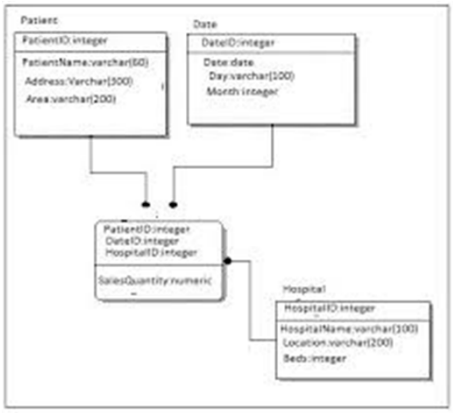


PHYSICAL DATA MODELS

Physical data model design

Convert :

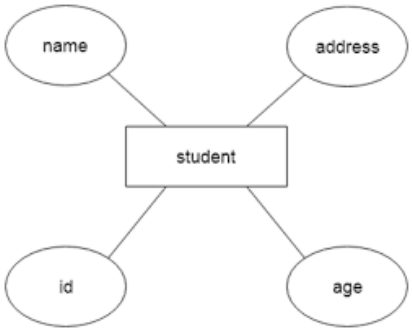
- Entities into tables and columns.
- Relationships into foreign keys.
- Attributes into columns.



3.3 Attributes

Attributes are description of the entities. E.g entity student attributes can be

- first name,
- last name,
- email,
- address and
- phone numbers.



DIFFERENT TYPES OF ATTRIBUTES

Types of Attributes	Description
Simple attribute	Simple attributes can't be divided any further. For example, a student's contact number. It is also called an atomic value.
Composite attribute	It is possible to break down composite attribute. For example, a student's full name may be further divided into first name, second name, and last name.
Derived attribute	This type of attribute does not include in the physical database. However, their values are derived from other attributes present in the database. For example, age should not be stored directly. Instead, it should be derived from the DOB of that employee.
Multivalued attribute	Multivalued attributes can have more than one values. For example, a student can have more than one mobile number, email address, etc.

https://www.guru99.com/er-diagram-tutorial-dbms.html

9

KAPLAN

3.4 STRONG & WEAK ENTITY TYPES

A **strong entity type** if its existence does not depend upon the existence of another entity type.

Weak entity is dependent on strong entity. It cannot be identified by its own attributes alone. It needs a foreign key to relate it to a strong entity.

10

24/2/2021

KAPLAN

10

COMPARISON OF STRONG & WEAK ENTITY SET

Strong Entity Set	Weak Entity Set
Strong entity set always has a primary key.	It does not have enough attributes to build a primary key.
It is represented by a rectangle symbol.	It is represented by a double rectangle symbol.
It contains a Primary key represented by the underline symbol.	It contains a Partial Key which is represented by a dashed underline symbol.
The member of a strong entity set is called as dominant entity set.	The member of a weak entity set called as a subordinate entity set.
Primary Key is one of its attributes which helps to identify its member.	In a weak entity set, it is a combination of primary key and partial key of the strong entity set.
In the ER diagram the relationship between two strong entity set shown by using a diamond symbol.	The relationship between one strong and a weak entity set shown by using the double diamond symbol.
The connecting line of the strong entity set with the relationship is single.	The line connecting the weak entity set for identifying relationship is double.

https://www.guru99.com/er-diagram-tutorial-dbms.html

11

24/2/2021

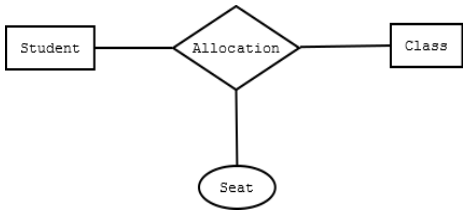
KAPLAN

3.5 ATTRIBUTES ON RELATIONSHIPS

- Attributes can also be assigned to relationships.
- Allow you to record facts about the relationship.

Example 1:

A student's class allocation may have an assigned seat



https://stackoverflow.com/questions/36245153/why-attributes-are-in-relation-in-er-diagrams

12

24/2/2021

KAPLAN

Review Questions

1. The ER model uses a number of notations and tags to represent different concepts. Outline how the basic ER components are represented in an ER diagram.
2. Provide an example of a relationship type with attributes.
3. Distinguish between the Entity—Relationship model and the Entity—Relationship diagram.
4. Describe how fan and chasm traps can occur in an ER model and how they can be resolved.

Exercises

1. Create an ER model for each of the following descriptions:
   a. A large organization has several parking lots, which are used by staff.
   b. Each parking lot has a unique name, location, capacity, and number of floors (where appropriate).
   c. Each parking lot has parking spaces, which are uniquely identified using a space number.
   d. Members of staff can request the sole use of a single parking space. Each member of staff has a unique number, name, telephone extension number, and vehicle license number.
   e. Represent all the ER models described in parts (a), (b), (c), and (d) as a single ER model. Provide any assumptions necessary to support your model.

Exercises

2. Create an ER model to represent the data used by the library.

The library provides books to borrowers. Each book is described by title, edition, and year of publication, and is uniquely identified using the ISBN. Each borrower is described by his or her name and address and is uniquely identified using a borrower number. The library provides one or more copies of each book and each copy is uniquely identified using a copy number, status indicating if the book is available for loan, and the allowable loan period for a given copy. A borrower may loan one or many books, and the date each book is loaned out and is returned is recorded. Loan number uniquely identifies each book loan.

REFERENCES

Thomas Connolly, Carolyn Begg (Sixth edition, Global edition) 2015, Database Systems. A Practical Approach to Design, Implementation, and Management , Pearson, USA.

KAPLAN

Introduction to Database Design and Development

Topic 04
Normalization

LEARNING OUTCOMES

Upon successful completion of this module, the student should be able to:

1. Describe the importance of Normalization and understand normalization rule
2. Explain different types of Normalization
3. Explain the concept of Denormalization
4. List the advantages and disadvantages of Denormalization

2

KAPLAN

NORMALIZATION

The word “normalization” used in statistics would mean the process of separating relations into well-structured relations or a process of reorganizing data in a database that allow users to insert, delete, and update tuples without introducing unwanted database. Fundamentally, normalization has to meet two basic requirements:

-No redundancy of data as all data is stored in only one place.
-Data dependencies are logical as all related data items are stored together.

4.1 ANOMALIES

If a database is poorly planned and designed, it will create problems when trying to insert, delete or modify/update the tables in the database.

Anomalies - once anomaly occurs, can become a costly and challenging task as the database grows.

3 types of problems that can occur in databases:

- Insertion anomaly
- Deletion anomaly
- Modification/Update anomaly

INSERTION ANOMALY

Required data cannot be added unless another piece of unavailable data is also added.

For example, a hospital database that cannot store the details of a new member until that member has been seen by a doctor.

5

24/2/2021

KAPLAN

DELETION ANOMALY

Deletion of a record of data can cause the deletion of some required data.

For example, deleting some of the patient's details can remove all the details of the patient from the hospital database.

6

24/2/2021

KAPLAN

6

MODIFICATION/UPDATE ANOMALY

Incorrect data may have to be changed, which could involve many records having to be changed, leading to the possibility of some changes being made incorrectly.

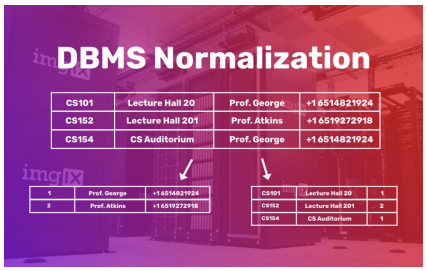
NORMALIZATION

Due to anomalies, normalization needs to be addressed

Normalisation is a systematic approach of decomposing tables to eliminate data redundancy; eliminates unnecessary duplication(s). Without normalisation, database systems can be inaccurate, slow, and inefficient.

Normalization rules. 3 forms:

- First Normal Form
- Second Normal Form
- Third Normal Form



Without any normalization, all information is stored in one table as shown below.

FULL NAMES	PHYSICAL ADDRESS	MOVIES RENTED	SALUTATION
Janet Jones	First Street Plot No 4	Pirates of the Caribbean, Clash of the Titans	Ms.
Robert Phil	3rd Street 34	Forgetting Sarah Marshal, Daddy's Little Girls	Mr.
Robert Phil	5th Avenue	Clash of the Titans	Mr.

4.2 FIRST NORMAL FORM (1NF)

- It should only have single (atomic) valued attributes/columns.
- Values stored in a column should be of the same domain
- All the columns in a table should have unique names.
- And the order in which data is stored, does not matter.

FULL NAMES	PHYSICAL ADDRESS	MOVIES RENTED	SALUTATION
Janet Jones	First Street Plot No 4	Pirates of the Caribbean	Ms.
Janet Jones	First Street Plot No 4	Clash of the Titans	Ms.
Robert Phil	3rd Street 34	Forgetting Sarah Marshal	Mr.
Robert Phil	3rd Street 34	Daddy's Little Girls	Mr.
Robert Phil	5th Avenue	Clash of the Titans	Mr.

4.3 SECOND NORMAL FORM (2NF)

- It should be in the First Normal form.
- And, it should not have Partial Dependency.

Comments :

- 1NF table is divided into 2 tables; Table 1 and Table2.
- Table 1 contains member information.
- Table 2 contains information on movies rented.
- A new column (Membership_ID) is added which is the primary key for Table 1.
- In Table 2, Membership_ID is the Foreign Key

MEMBERSHIP ID	FULL NAMES	PHYSICAL ADDRESS	SALUTATION
1	Janet Jones	First Street Plot No 4	Ms.
2	Robert Phil	3rd Street 34	Mr.
3	Robert Phil	5th Avenue	Mr.

Table 1

MEMBERSHIP ID	MOVIES RENTED
1	Pirates of the Caribbean
1	Clash of the Titans
2	Forgetting Sarah Marshal
2	Daddy's Little Girls
3	Clash of the Titans

Table2

4.4 THIRD NORMAL FORM (3NF)

- It is in the Second Normal form.
- Doesn't have Transitive Dependency.

MEMBERSHIP ID	FULL NAMES	PHYSICAL ADDRESS	SALUTATION ID
1	Janet Jones	First Street Plot No 4	2
2	Robert Phil	3rd Street 34	1
3	Robert Phil	5th Avenue	1

Table 1

MEMBERSHIP ID	MOVIES RENTED
1	Pirates of the Caribbean
1	Clash of the Titans
2	Forgetting Sarah Marshal
2	Daddy's Little Girls
3	Clash of the Titans

Table 2

SALUTATION ID	SALUTATION
1	Mr.
2	Ms.
3	Mrs.
4	Dr.

Table 3

Comments :

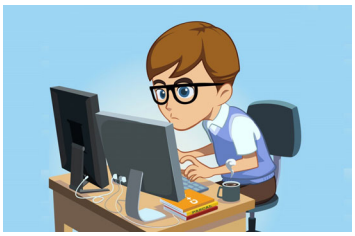
- 1NF table is divided into 2 tables; Table 1 and Table2.
- New table was created to store Salutations.
- There are no transitive functional dependencies, and hence the table is in 3NF.
- In Table 3 **Salutation ID** is *primary key*, and in Table 1 **Salutation ID** is *foreign* to primary *key* in Table 3.

4.5 DENORMALIZATION

Denormalization is a database optimization technique in which we deliberately add redundant data to one or more tables. This process will help to avoid multiple table joins. *Denormalization is an optimization technique that is applied after doing normalization.*

Advantages of Denormalization
- Data retrieval will be faster
- Avoid multiple table joins
- Query will be easy to read as it will refer fewer tables

Disadvantages of Denormalization
- Extra storage space
- Update and insert operations are more expensive
- Data redundancy
- Potential data anomalies



13

24/2/2021

KAPLAN

Review Questions

1. Describe the purpose of normalizing data and its importance.
2. 2. How would you fix the below table to reach 1N?

ACCOUNTNO	NAME	SUBJECT
542	Tafadzwa	Eng, Maths
543	Sipho	Science, ICT
544	Gift	ICT, Maths
545	Naledi	English, Science

14

24/2/2021

KAPLAN

Review Questions

3. The second normal form (2NF) is realized by removing partial dependencies from 1NF relations. Briefly describe the term “partial dependency.”

4. Describe the concept of transitive dependency and describe how this concept relates to 3NF. Provide an example to illustrate your answer.

Exercises

The table shown in Figure 14.19 lists sample dentist/patient appointment data. A patient is given an appointment at a specific time and date with a dentist located at a particular surgery. On each day of patient appointments, a dentist is allocated to a specific surgery for that day.

(a) The table shown in Figure 14.19 is susceptible to update anomalies. Provide examples of insertion, deletion, and update anomalies.

(b) Identify the functional dependencies represented by the attributes shown in the table of Figure 14.19. State any assumptions you make about the data and the attributes shown in this table.

(c) Describe and illustrate the process of normalizing the table shown in Figure 14.19 to 3NF relations. Identify the primary, alternate, and foreign keys in your 3NF relations.

Exercises

staffNo	dentistName	patNo	patName	appointment date	time	surgeryNo
S1011	Tony Smith	P100	Gillian White	12-Sep-13	10.00	S15
S1011	Tony Smith	P105	Jill Bell	12-Sep-13	12.00	S15
S1024	Helen Pearson	P108	Ian MacKay	12-Sep-13	10.00	S10
S1024	Helen Pearson	P108	Ian MacKay	14-Sep-13	14.00	S10
S1032	Robin Plevin	P105	Jill Bell	14-Sep-13	16.30	S15
S1032	Robin Plevin	P110	John Walker	15-Sep-13	18.00	S13

Figure 14.19 Table displaying sample dentist/patient appointment data.

REFERENCES

Thomas Connolly, Carolyn Begg (Sixth edition, Global edition) 2015, Database Systems. A Practical Approach to Design, Implementation, and Management , Pearson, USA.

KAPLAN

Introduction to Database Design and Development

Topic 05
Structured Query Language (SQL) Part I

LEARNING OUTCOMES

Upon successful completion of this module, the student should be able to:

1. Explain the purpose and importance of the Structured Query Language (SQL)
2. List the data types supported by the SQL Standard
3. Describe the facilities provided by SQL Standard for integrity control.
4. Describe the purpose of Data Definition Language (DDL)
5. Demonstrate practical skills in using SQL

5.1 INTRODUCTION TO SQL

SQL stands for Structured Query Language, which is a standardised language for interacting (e.g. storing and managing) with RDBMS (Relational Database Management System). Almost all RDBMS (e.g. MySql, Oracle, Infomix, Sybase, MS Access) use **SQL** as the standard database query language.

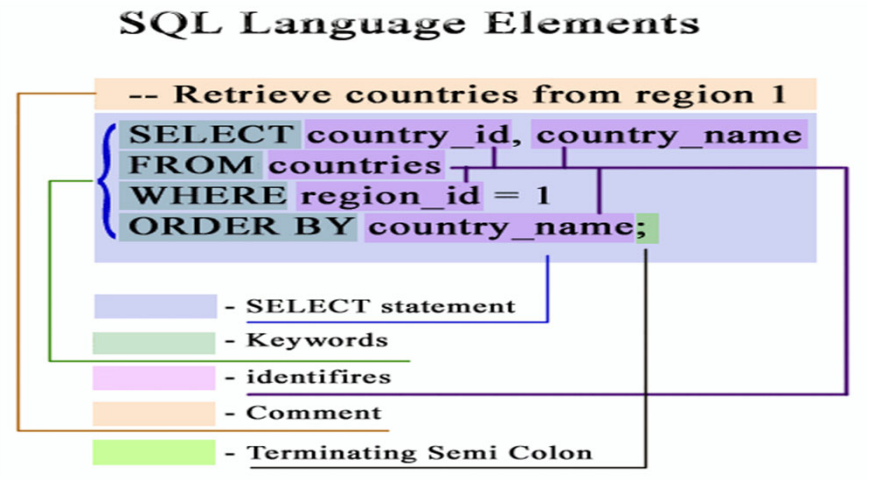
5.2 WRITING SQL COMMANDS

Reserved words

-They are a fixed part and have a fixed meaning. Must be spelled exactly and cannot be split across lines.

User-defined words

-They are created by the user which denotes the names of various database objects such as tables, columns, views, indexes, and others.



SQL is basically a combination of four different languages, they are –

DQL (Data Query Language)

DQL is used to fetch the information from the database which is already stored there.

DDL (Data Definition Language)

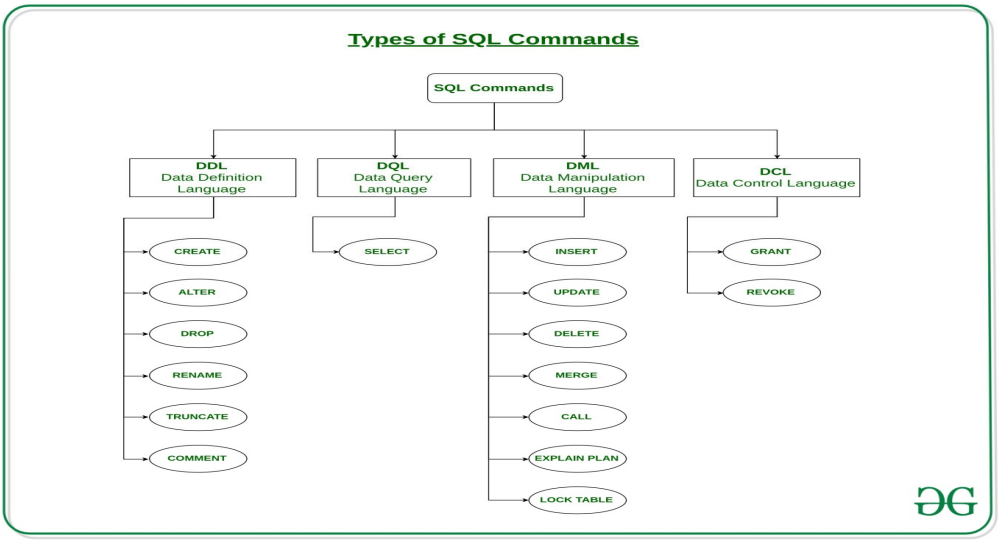
DDL is used to define table schemas.

DCL (Data Control Language)

DCL is used for user & permission management. It controls the access to the database.

DML (Data Manipulation Language)

DML is used for inserting, updating and deleting data from the database.



Source: https://beginnersbook.com/2018/11/introduction-to-sql/

ISO SQL DATA TYPES

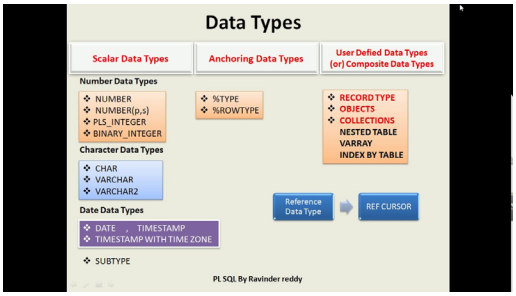
DATA TYPE	DECLARATIONS				
boolean	BOOLEAN				
character	CHAR	VARCHAR			
bit†	BIT	BIT VARYING			
exact numeric	NUMERIC	DECIMAL	INTEGER	SMALLINT	BIGINT
approximate numeric	FLOAT	REAL	DOUBLE PRECISION		
datetime	DATE	TIME	TIMESTAMP		
interval	INTERVAL				
large objects	CHARACTER LARGE OBJECT		BINARY LARGE OBJECT		

ISO SQL DATA TYPES

The character and bit are collectively referred to as string data types and numeric data types.

The types of scalar functions are as follows:

- Data type conversion
- Numeric
- String
- Date
- Hash
- Random number



5.4 INTEGRITY ENHANCEMENT FEATURE

Integrity control consists of constraints that need to be enforced in order to protect the database from becoming inconsistent. There are 5 types of integrity constraint

- Required data;
- Domain constraints;
- Entity integrity;
- Referential integrity;
- General / Enterprise constraints.

5.4 INTEGRITY ENHANCEMENT FEATURE

Required Data

Some columns must contain some valid value, they are not allowed to contain nulls. SQL provides the **NOT NULL** column specifier in the **CREATE TABLE** statement to enforce the required data constraint.

```
Position VARCHAR (15) NOT NULL
```

Domain constraints

Every column will have a set of allowable values. For example, gender of the employee is either male (M) or female (F). SQL uses **CHECK** clause to enforce this domain constraint on a column.

```
Sex CHAR NOT NULL CHECK (sex IN ('M','F'))
```

11

24/2/2021

KAPLAN

5.4 INTEGRITY ENHANCEMENT FEATURE

Entity integrity

SQL supports entity integrity with **PRIMARY KEY** clause as primary key value of a table. It must be unique and cannot be null.

```
PRIMARY KEY (EmpNo)
```

To define a composite primary key, we specify multiple column name

```
PRIMARY KEY (clientNo,propertyNo)
```

12

24/2/2021

KAPLAN

5.4 INTEGRITY ENHANCEMENT FEATURE

Referential Integrity

Since a **foreign key** links each row in the child table to a parent table containing the matching candidate key value, it means a foreign key must contains an existing value or valid row in the parent table.

FOREIGN KEY (branchNo) REFERENCES Branch

SQL rejects any INSERT or UPDATE operation that attempts to create a foreign key value in a child table without a matching candidate key value in the parent table. The action SQL takes for any UPDATE or DELETE operation of a candidate key value in the parent table must have some matching rows in the child table is dependent on.

13

24/2/2021

KAPLAN

5.4 INTEGRITY ENHANCEMENT FEATURE

Enterprise Constraints;

Updates to tables may be constrained by enterprise rules governing real-world transactions that are represented by the updates.

14

24/2/2021

KAPLAN

5.4 DDL COMMANDS

SQL DDL commands can be used to define the database schema. It allows database objects to be *created, modify* the structure and *destroyed.*

Examples of DDL commands:

CREATE – is used to create the database or its objects (like table, index, function, views, store procedure and triggers).
DROP – is used to delete objects from the database.
ALTER-is used to alter the structure of the database.
TRUNCATE–is used to remove all records from a table, including all spaces allocated for the records are removed.
COMMENT –is used to add comments to the data dictionary.
RENAME –is used to rename an object existing in the database.

Types of SQL Commands

DDL	DML	DCL	TCL
CREATE ALTER DROP TRUNCATE RENAME	SELECT INSERT UPDATE DELETE MERGE	GRANT REVOKE	COMMIT ROLLBACK SAVEPOINT

15
24/2/2021
KAPLAN

CREATE

The very first step to store the data is to create a database in a well-structured manner. The **CREATE DATABASE** statement is used to create a new database in SQL.

```
CREATE DATABASE database_name;
```

Next, store the data in a table using **CREATE TABLE** which comprises of rows and columns.

```
CREATE TABLE Students
(
ROLL_NO int(3),
NAME varchar(20),
SUBJECT varchar(20),
);
```

16
24/2/2021
KAPLAN

DROP

The command is use to remove a user account along with its privileges from the MySQL completely.

DROP USER 'user'@'host';

Example:

From this:

```
DROP USER 'gfguser1'@'localhost';
```

```
+---------------+--------------------+
| user          | host               |
+---------------+--------------------+
| gfguser1      | localhost          |
| gfguser2      | localhost          |
| gfguser3      | localhost          |
| gfguser4      | localhost          |
+---------------+--------------------+
```

To this:

```
+---------------+--------------------+
| user          | host               |
+---------------+--------------------+
| gfguser2      | localhost          |
| gfguser3      | localhost          |
| gfguser4      | localhost          |
+---------------+--------------------+
```

17

24/2/2021

KAPLAN

ALTER

It is used to *add, delete/drop* or *modify* columns in the existing table. It is also used to *add* and *drop* various constraints on the existing table.

ADD columns into the existing table:

```
ALTER TABLE table_name
        ADD (Columnname_1  datatype,
        Columnname_2  datatype,
        …
        Columnname_n  datatype);
```

DROP COLUMN to delete unwanted columns from the table.

```
ALTER TABLE table_name
DROP COLUMN column_name;
```

18

24/2/2021

KAPLAN

Multiple columns can be modified using MODIFY command.

ALTER TABLE table_name
MODIFY column_name column_type;

Types of SQL Commands			
DDL	**DML**	**DCL**	**TCL**
CREATE ALTER DROP TRUNCATE RENAME	SELECT INSERT UPDATE DELETE MERGE	GRANT REVOKE	COMMIT ROLLBACK SAVEPOINT

19 24/2/2021 KAPLAN

Review Questions

1. What are the main SQL DDL statements?
2. Discuss the functionality and importance of the Integrity Enhancement Feature (IFF).

20 24/2/2021 KAPLAN

Exercises

1. Create the Hotel table using the integrity enhancement features of SQL.

2. Now create the Room, Booking, and Guest tables using the integrity enhancement features of SQL with the following constraints:
   a. Type must be one of Single, Double, or Family.
   b. Price must be between £10 and £100.
   c. Room No must be between 1 and 100.
   d. Date From and date To must be greater than today's date.
   e. The same room cannot be double-booked.
   f. The same guest cannot have overlapping bookings.

REFERENCES

Thomas Connolly, Carolyn Begg (Sixth edition, Global edition) 2015, Database Systems. A Practical Approach to Design, Implementation, and Management , Pearson, USA.

Introduction to Database Design and Development

Topic 06

Structured Query Language (SQL) Part II

LEARNING OUTCOMES

Upon successful completion of this module, the student should be able to:

1. Explain the purpose of creating simple database.
2. Describe the purpose of Data Manipulation Language (DML).
3. List SQL aggregate functions and its usage.

6.1 CREATE SIMPLE DATABASE

There are two CREATE statements that need to be addressed:

- CREATE DATABASE
- CREATE TABLE

Create Database

The very first step to store the data in a structured manner is to create a database. In SQL, Data Definition Language (DDL) commands are used to create and modify the structure of a database and objects.

Syntax:

CREATE DATABASE Database_Name

Try:

Created a database ‘ **testDB1** ’

6.1 CREATE SIMPLE DATABASE

Create Table

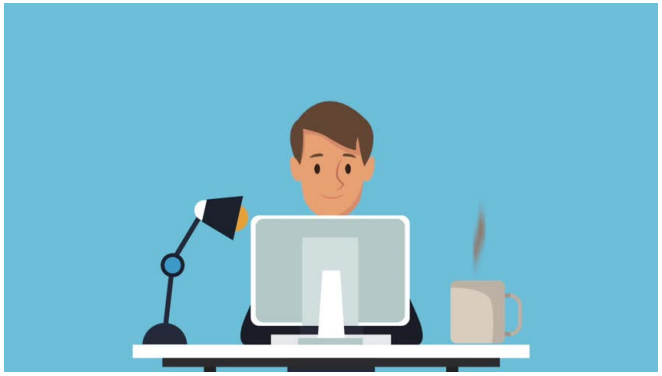
Next, we need to create tables (comprising of rows and columns). We have to provide names of the columns, type of data to be stored in columns, size of the data etc.

Syntax:

CREATE TABLE *table_name* (
    column1 datatype,
    column2 datatype,
    column3 datatype,
   ....
);

Example:

```
CREATE TABLE Students
(
ROLL_NO int(3),
NAME varchar(20),
SUBJECT varchar(20),
);
```



Try:

Created a table of an ‘ **Employee** ‘

6.2 DML commands

Data Manipulation Language (DML) commands are used for managing data within tables.

Examples of DML commands:

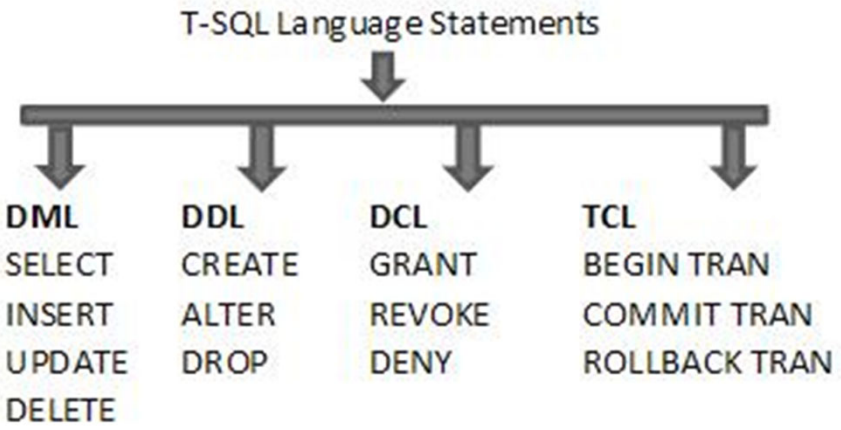
SELECT	To select records or data from a table
INSERT	Insert data into a database table
UPDATE	Update existing records within a table
DELETE	Delete unwanted records from a table

Syntax : SELECT

```
SELECT column1, column2, ...
FROM table_name;
```

Example:

```
SELECT CustomerName, City FROM Customers;
```



Syntax : INSERT INTO SELECT

```
INSERT INTO table2 (column1, column2, column3, ...)
SELECT column1, column2, column3, ...
FROM table1
WHERE condition;
```

Example:

```
INSERT INTO Customers (CustomerName, City, Country)
SELECT SupplierName, City, Country FROM Suppliers
WHERE Country='Germany';
```

Syntax : UPDATE

```
UPDATE table_name
SET column1 = value1, column2 = value2, ...
WHERE condition;
```

Example:

```
UPDATE Customers
SET ContactName = 'Alfred Schmidt', City= 'Frankfurt'
WHERE CustomerID = 1;
```

ISO SQL DATA TYPES

Syntax : UPDATE

```
UPDATE table_name
SET column1 = value1, column2 = value2, ...
WHERE condition;
```

Example:

```
UPDATE Customers
SET ContactName = 'Alfred Schmidt', City= 'Frankfurt'
WHERE CustomerID = 1;
```

Syntax : DELETE

DELETE FROM table_name WHERE condition;

Example:

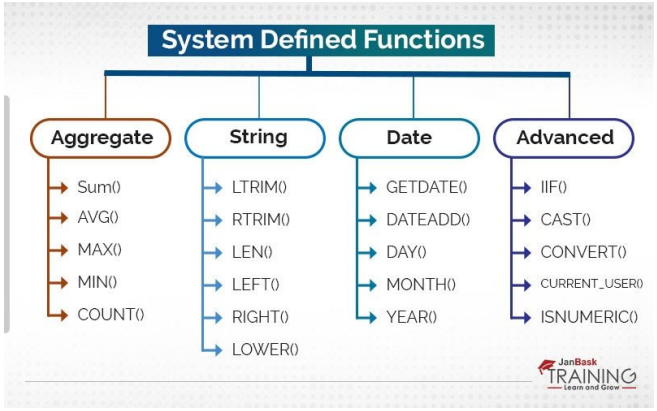
DELETE FROM Customers WHERE CustomerName='Alfreds Futterkiste';

6.3 AGGREGATE FUNCTIONS

Aggregate functions are used for summarizing results and obtaining useful data through SQL commands (e.g. reports in the form of tables using simple calculations).

Some commonly used SQL aggregate functions include:

- **AVG** – calculates the average of a set of values.
- **COUNT** – counts rows in a specified table or view.
- **MIN** – gets the minimum value in a set of values.
- **MAX** – gets the maximum value in a set of values.
- **SUM** – calculates the sum of values.



Syntax : AVG() **SELECT** **AVG(column_name)** **FROM table_name** **WHERE condition;**	**Syntax : COUNT()** **SELECT** **COUNT(column_name)** **FROM table_name** **WHERE condition;**
Syntax : MIN() SELECT MIN(column_name) FROM table_name WHERE condition;	Syntax : MAX() SELECT MAX(column_name) FROM table_name WHERE condition;
Syntax : SUM() SELECT SUM(column_name) FROM table_name WHERE condition;	

Trys:

To calculate the **AVERAGE** of total selected records or rows, look at example in:

https://www.tutorialgateway.org/sql-avg-function/

To **COUNT** the total number of rows (records) selected by the SELECT Statement, look at example in:

https://www.tutorialgateway.org/sql-count-function/

To find the **MAXIMUM** value from the total rows or records selected by the SELECT Statement, look at example in:

https://www.tutorialgateway.org/sql-max-function/

Trys:

To find the **MINIMUM** value from the total records (or rows) selected by the SELECT Statement, look at example in:

https://www.tutorialgateway.org/sql-min-function/

To calculate the total or **SUM** of records (or rows) selected, look at example in:

https://www.tutorialgateway.org/sql-sum-function/

Review Questions

1. Briefly describe the four basic SQL DML statements, Aggregate functions and explain their usage.

REFERENCES

Thomas Connolly, Carolyn Begg (Sixth edition, Global edition) 2015, Database Systems. A Practical Approach to Design, Implementation, and Management , Pearson, USA.

KAPLAN

Introduction to Database Design and Development

Topic 07
Structured Query Language (SQL) Part III

LEARNING OUTCOMES

Upon successful completion of this module, the student should be able to:

1. Describe different types of Join used in SQL.
2. Explain the purpose of Views in database.
3. Explain the mechanism of SQL Discretionary Access Control (DAC).

7.1 TYPES OF JOIN

Few different types of JOINs in SQL:

- **(INNER) JOIN**: Returns records that have matching values in both tables
- **LEFT (OUTER) JOIN**: Returns all records from the left table, and the matched records from the right table
- **RIGHT (OUTER) JOIN**: Returns all records from the right table, and the matched records from the left table
- **FULL (OUTER) JOIN**: Returns all records when there is a match in either left or right table

KAPLAN

24/2/2021

A. Syntax : INNER JOIN

SELECT *column_name(s)*
FROM *table1*
INNER JOIN *table2*
ON *table1.column_name* = *table2.column_name*;

Example:

```
SELECT Orders.OrderID,
Customers.CustomerName
FROM Orders
INNER JOIN Customers
ON Orders.CustomerID = Customers.CustomerID;
```

B. Syntax : LEFT JOIN

SELECT *column_name(s)*
FROM *table1*
LEFT JOIN *table2*
ON *table1.column_name* = *table2.column_name*;

Example:

```
SELECT Customers.CustomerName, Orders.OrderID
FROM Customers
LEFT JOIN Orders
ON Customers.CustomerID = Orders.CustomerID
ORDER BY Customers.CustomerName;
```

4

KAPLAN

24/2/2021

C. Syntax : RIGHT JOIN

SELECT *column_name(s)*
FROM *table1*
RIGHT JOIN *table2*
ON *table1.column_name = table2.column_name*;

Example:

```
SELECT Orders.OrderID, Employees.LastName,
Employees.FirstName
FROM Orders
RIGHT JOIN Employees
ON Orders.EmployeeID = Employees.EmployeeID
ORDER BY Orders.OrderID;
```

D. Syntax : FULL OUTER JOIN

SELECT *column_name(s)*
FROM *table1*
FULL OUTER JOIN *table2*
ON *table1.column_name = table2.column_name*
WHERE *condition*;

Example:

```
SELECT Customers.CustomerName, Orders.OrderID
FROM Customers
FULL OUTER JOIN Orders
ON Customers.CustomerID=Orders.CustomerID
ORDER BY Customers.CustomerName;
```

5
24/2/2021
KAPLAN

7.2 VIEWS

Views are the virtual relations, which consist of columns, rows from the referenced table.

It does not necessarily exist in the database.

DBMS stores the definition of the view in the database.

View always shows up-to-date data.

6
24/2/2021
KAPLAN
6

Syntax : CREATE VIEW

CREATE VIEW *view_name* AS
SELECT *column1*, *column2*, ...
FROM *table_name*
WHERE *condition*;

Example :

```
CREATE VIEW Manager3Staff
AS SELECT *
FROM Staff
WHERE branchNo = 'B003';
```

A view called Manager3Staff is created with the same column names as the Staff table but containing only those rows where the branch number is B003.

A view is deleted with the DROP VIEW command.

Syntax : SQL DROP VIEW

DROP VIEW *view_name*;

Example

DROP VIEW [Brazil Customers];

7.3 DISCRETIONARY ACCESS CONTROL

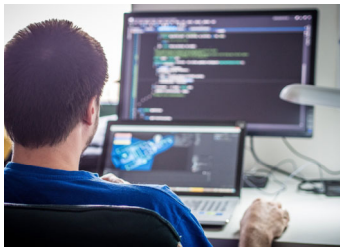
Discretionary Access Control (DAC) is a mechanism to ensure that only authorized users can access the intended database with certain mode (e.g. read or write).

SQL supports discretionary access control through GRANT and REVOKE statements.

With **GRANT** command: it allows users to grant access rights to other users.

With **REVOKE** command: it removes user access rights to the database objects.

Syntax : GRANT



```
GRANT privilege_name
ON object_name
TO {user_name |PUBLIC |role_name}
[WITH GRANT OPTION];
```

- ***privilege_name*** is the access right or privilege granted to the user. Some of the access rights are ALL, EXECUTE, and SELECT.
- ***object_name*** is the name of an database object like TABLE, VIEW, STORED PROC and SEQUENCE.
- ***user_name*** is the name of the user to whom an access right is being granted.
- ***PUBLIC*** is used to grant access rights to all users.
- ***ROLES*** are a set of privileges grouped together.
- ***WITH GRANT OPTION*** - allows a user to grant access rights to other users.

Example

```
GRANT UPDATE (EMPNO,DEPT)
ON TABLE EMP
TO NATZ;
```

- This statement grants UPDATE privileges on columns EMPNO and DEPT in the EMP table to user NATZ.

Syntax : REVOKE

```
REVOKE privilege_name
ON object_name
FROM {user_name |PUBLIC |role_name}
```

Example

```
REVOKE UPDATE
ON EMP
FROM NATZ;
```

This statement revokes the UPDATE privilege on the EMP table from NATZ.

Review Questions

1. Discuss the differences between the Join operations.
2. Discuss the advantages and disadvantages of views.
3. Describe how the access control mechanisms of SQL work.

REFERENCES

Thomas Connolly, Carolyn Begg (Sixth edition, Global edition) 2015, Database Systems. A Practical Approach to Design, Implementation, and Management , Pearson, USA.

KAPLAN

Introduction to Database Design and Development

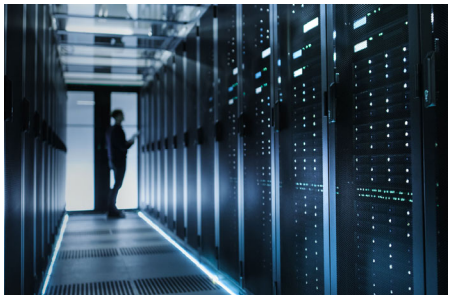
Topic 08
Transaction Management

LEARNING OUTCOMES

Upon successful completion of this module, the student should be able to:

1. Describe the properties of Transactions.
2. List different types of lock protocols.
3. Explain the issues and approaches involved in the processing of concurrent database transactions.

The tasks of processing multiple transactions on a database while maintaining **A**tomicity, **C**onsistency, **I**solation, and **D**urability; known commonly as **ACID** properties. This is to ensure accuracy, completeness, and data integrity.



KAPLAN

24/2/2021

8.1 PROPERTIES OF TRANSACTION

ACID Properties

A transaction must maintain ACID properties:

Atomicity − A transaction must be treated as a single operation. Either all of its operations are executed or none. No transaction should be left partially completed.

Consistency – Data should be in a consistent state when a transaction starts and when it ends. For example, funds being transferred from one account to another. Under consistency rule, total value of funds in both the accounts must still be the same before and after transaction.

4

KAPLAN

24/2/2021

8.1 PROPERTIES OF TRANSACTION

ACID Properties

Isolation − Where more than one transaction are being executed in parallel. For example, in an application that transfers funds from one account to another, the transactions that run concurrently appear to be serialized. Property of isolation states that all the transactions will be carried out as if it is the only transaction in the system. It will not affect the existence of any other transaction.

Durability − Database holds all its latest updates even if the system fails. For example, a transaction commits to transfer funds to another account, but the system fails before the data could be written, that data will be updated once the system comes back into action.

5

24/2/2021

KAPLAN

8.2 LOCKS

The importance of control in multiple transactions in multiprogramming environment is vital.

Concurrency control protocols like *Lock-based Protocols* can lock data variable and help synchronize access to the database items.

In the database systems, any transaction cannot *read* or *write* data until it acquires an appropriate lock on it.

Basically, there are 2 kinds of locks:

6

24/2/2021

KAPLAN

Binary Locks − A lock on a data item can be in two states; it is either locked or unlocked.

Shared/exclusive – It differentiates the locks based on their uses. To perform a **Write** operation; also known as exclusive lock. It allows more than one transaction to read and write on the same data item. **Read** locks are shared because no data value is being changed.

There are **4 types of lock protocols** available:

1. Simplistic Lock Protocol

It allows transactions to obtain a lock on every object before a 'write' operation is executed. Transactions may unlock the data item after ending the ‘write’ operation.

2. Pre-claiming Lock Protocol

The protocols evaluate its operations and create a list of required data items on which they need. Before execution, the system requests all the locks needed.

If all locks are granted, the transaction will execute otherwise the transaction will rolls back and waits until all the locks are granted. It releases all locks when all its operations are over.

8.3 TWO PHASE LOCKING

3. Two-Phase Locking (2PL)

This locking protocol divides the execution phase of a transaction into three parts.

- In the first part, the transaction starts executing, it seeks permission for the locks it requires.
- The second part, the transaction acquires all the locks. Third phase starts as soon as the transaction releases its first lock.
- The third part, the transaction cannot demand any new locks; it only releases the acquired locks.

8.3 TWO PHASE LOCKING

There are 2 phases of 2PL:

- **Growing phase:** In the growing phase, all locks are being acquired by the transaction and none can be released.
- **Shrinking phase:** In the shrinking phase, locks held by the transaction are being released and no new locks can be acquired.

8.3 TWO PHASE LOCKING

4. Strict Two-Phase Locking (Strict-2PL)

The first phase of Strict-2PL is identical to 2PL. The transaction continues to execute normally after acquiring all the locks.

In contrast to 2PL, Strict-2PL does not release a lock after using it. It holds all the locks from transaction to commit, only then it releases all locks at a time.

11 24/2/2021 KAPLAN 11

8.4 CONCURRENCY CONTROL

Is an **important** concept for managing simultaneous operations without conflicting with each another or violating the data integrity of databases. It avoids any loss of data and to ensure proper updating of data.

Concurrency control is provided in a database to:

- enforce isolation among transactions.
- preserve database consistency through consistency preserving execution of transactions.
- resolve read-write and write-read conflicts.

12 24/2/2021 KAPLAN 12

Deadlocks

In concurrent computing, a deadlock is a situation where two (or more) transactions are blocked because each process is holding a resource and waiting for another resource held by the other. Thus, neither transaction can continue because each transaction is in the waiting queue

Various concurrency control techniques are: (as seen below). Lock-Based Protocols & Two Phase were covered earlier.

- Lock-Based Protocols
- Two Phase locking Protocol
- Timestamp-Based Protocols
- Multi version concurrency control
- Validation Concurrency Control

13 24/2/2021 KAPLAN 13

Timestamp-based Protocols

It uses algorithm timestamp to serialize the execution of concurrent transactions. It ensures every read and write operations are executed in timestamp order.

Every transaction has a timestamp, and the order is determined by the age of the transaction. Thus, the older transaction is always given priority and is the most commonly used concurrency protocol.

Advantages:

- Schedules are serializable just like 2PL protocols
- No waiting for the transaction and eliminates the possibility of deadlocks!

Disadvantages:

- Starvation is possible if the same transaction is restarted and continually aborted

14 24/2/2021 KAPLAN 14

Multiversion Concurrency Control:

The schemes keep old versions of data item to increase concurrency.

Timestamps are used to label the new version of the data item written.

Validation Concurrency Control:

This is an optimistic approach based on the assumption that the majority of the database operations do not conflict.

It requires neither locking nor time stamping techniques. Transaction is executed without restrictions until it is committed.

Each transaction moves through read, validation and write.

Review Questions

1. The consistency and reliability aspects of transactions are due to the "ACIDity" properties of transactions. Discuss each of these properties and how they relate to the concurrency control and recovery mechanisms. Give examples to illustrate your answer.

2. Describe, with examples, the types of problem that can occur in a multi-user environment when concurrent access to the database is allowed.

Exercises

1. Write an algorithm that checks whether the concurrently executing transactions are in deadlock.

REFERENCES

Thomas Connolly, Carolyn Begg (Sixth edition, Global edition) 2015, Database Systems. A Practical Approach to Design, Implementation, and Management , Pearson, USA.

KAPLAN

Introduction to Database Design and Development

Topic 09

Database Administration and Security

LEARNING OUTCOMES

Upon successful completion of this module, the student should be able to:

1. Describe the Database Administrator roles and responsibilities.
2. Discuss the role of the database administrator in database security and recovery.
3. Explain the importance of database security and types of security approaches used in the organizations.
4. Describe the approaches for securing DBMSs on the Web.

2

KAPLAN

9.1 DBA ROLES & RESPONSIBILITIES

Database Administrator (DBA) is responsible for the physical realization of the database. The role includes capacity planning, installation, configuration, database design, migration, performance monitoring, security, troubleshooting, backup and data recovery.

Briefly, they can be summaries as:

- Administer and maintain all activities associate with database and ensure optimal performance.
- Provide support to SQL Server through various activities.
- Maintain all Parallax operations and provide support to database operations.
- Monitor database and application levels and ensure work according to user privileges.
- Coordinate with various teams and monitor all replication and recovery procedures for database.

……..

3

24/2/2021

KAPLAN

9.2 DATABASE SECURITY

Database security encompasses hardware, software, people, and data.

The amounts of crucial corporate data being stored and the loss or unavailability of this data could prove to be disastrous.

Database security need to be addresses urgently:

- theft and fraud;
- loss of confidentiality (secrecy);
- loss of privacy;
- loss of integrity;
- loss of availability

4

24/2/2021

KAPLAN

9.3 COUNTERMEASURES – COMPUTER-BASED CONTROLS

There are many different forms of countermeasure to threats on computer systems ranging from physical controls to administrative procedures. Followings are some computer-based security controls:

- authorization,
- access controls,
- views,
- backup and recovery,
- integrity,
- encryption, and
- RAID technology.

5 24/2/2021 KAPLAN 5

9.3.1 Authorization

The granting of a right or privilege that enables a subject to have legitimate access to a system or a system's object.

9.3.2 Access Controls

The normal way of permitting access controls to a database system is based on granting and revoking of privileges within the database.

The database provides various types of access controls:

- **Discretionary Access Control (DAC)**
- **Mandatory Access Control (MAC)**

6 24/2/2021 KAPLAN 6

Discretionary Access Control (DAC)

In discretionary access control (DAC), the owner specifies or allows which end user to access their objects/data. This is typically the default access control mechanism because control of access is based on the discretion of the owner.

Mandatory Access Control (MAC)

Whereas in mandatory access control (MAC), the system (not the users) specifies which subjects can access specific data objects. It is the strictest of all levels of control.

7

24/2/2021

KAPLAN

7

9.3.3 Views

A view is presentation of data selected from one or more relational operations operating on the base relations to produce another relation.

It shows the structure of the underlying tables but is a virtual relation that does not actually exist in the database, and is produced upon request by user demand at the time of request.

A view contains no actual data, therefore, it does not necessarily exist in the storage. It creates and shows the tables on which it is based on. It can be manipulated as if it were a base relation.

8

24/2/2021

KAPLAN

8

The contents of a view are derived from query on one or more base relations.

Views are dynamic as the data in a view can be updated or deleted, and new data inserted.

As the operations will directly alter the based tables, integrity constraints on the subject are critical.

Base Table

employees

employee_id	last_name	job_id	manager_id	hire_date	salary	dept_id
203	marvis	hr_rep	101	07-Jun-94	6500	40
204	baer	pr_rep	101	07-Jun-94	10000	70
205	higgins	ac_rep	101	07-Jun-94	12000	110
206	gietz	ac_account	205	07-Jun-94	8300	110

View

staff

employee_id	last_name	job_id	manager_id	dept_id
203	marvis	hr_rep	101	40
204	baer	pr_rep	101	70
205	higgins	ac_rep	101	110
206	gietz	ac_account	205	110

9.3.4 Backup and Recovery

It is a process of periodically copying of the database and log file to offline secured storage media.

DBMS should provide these facilities (automatically backed up on regular basis) to restore the database to the latest possible state in the event of system failure. Backup can be in the form *full run, incremental, or differential*.

A full backup will take longer time to create, however, is much quicker than restoring from incremental or differential backups. A good option maybe a mix of backup plans; full backup once a week and daily incremental backups.

11 24/2/2021 KAPLAN 11

9.3.5 Integrity

Integrity constraints refer to conditions which must be present for a valid relation.
There are many types of integrity constraints and it is necessary to understand some of these concepts

Null

It represents a value of an attribute that is currently unknown or is not applicable for this row. Nulls can cause implementation problems if not defined.

. . . . .

12 24/2/2021 KAPLAN 12

Entity Integrity

In a base relation, entity integrity constraint states that a primary key value cannot be null because primary key value is used to identify individual rows in relation and if it has a null value, then we cannot identify those rows. A table can contain a null value but not the primary key field.

General Constraints

Lastly, user or database administrators can also specify additional constraints that the data must satisfy that maybe due to the constrains imposed or required by the enterprise.

Examples:
A class can have a maximum of 30 students.

13
24/2/2021
KAPLAN
13

9.3.6 Encryption

Encryption is the method by which data/information is converted into secret code using a special algorithm that renders the data unreadable or hides the information true meaning.

Thus, “to transmit data securely over insecure networks requires the use of a cryptosystem, which includes:

- an encryption key to encrypt the data (plaintext);
- an encryption algorithm that with the encryption key transforms the plaintext into ciphertext;
- a decryption key to decrypt the ciphertext;
- a decryption algorithm that with the decryption key transforms the ciphertext back into plaintext.”

14
24/2/2021
KAPLAN
14

Basically, there are two main kinds of encryption:

In **symmetric encryption,** there is only one key, and all communication lines use the same key for encryption and decryption. Data Encryption Standard (DES) is a standard encryption algorithm developed by IBM which uses one key for both encryption and decryption

In **asymmetric encryption**, there are two keys: one key (public key), shared publicly, is used for encryption, and a different key is used for decryption (private key).

The formulas to encode and decode messages are called *encryption algorithms,* or *ciphers* and the science of encrypting and decrypting the information is called *cryptography*.

15 24/2/2021 KAPLAN 15

9.3.7 RAID (Redundant Array of Independent Disks)

RAID (Redundant Array of Independent Disks or Redundant Array of Inexpensive Disks) is a way of storing the same data in different places on multiple hard disks to protect data in the case of a drive failure and to improve system performance.

This would mean DBMS should continue to operate if one of the hardware components fails. Thus, having redundant components working seamlessly is vital whenever there is one or more component failures.

There are different RAID levels configurations.

RAID 0 to RAID 6

16 24/2/2021 KAPLAN 16

9. 4 DBMSs AND WEB SECURITY

DBMSs and Web Security focus on how to make a DBMS secure on the Web. As Internet is an open network and communication relies basically on TCP/IP and HTTP as the underlying protocol, it is not designed with security in mind.

Anyone who monitors the traffic can use a “packet sniffing” software to read those messages.

Besides, information transmitted to the client’s machine may contain executable content such as HTML pages may contain ActiveX controls, JavaScript/VBScript, and/or Java applets. These executable contents can perform malicious actions.

Thus, the need to address database security in these environments is critical.

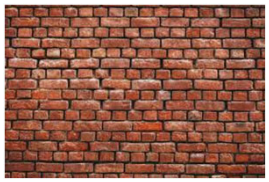
Proxy Servers

“A proxy server is basically another computer which serves as a hub through which internet requests are processed. By connecting through one of these servers, your computer sends your requests to the server which then processes your request and returns what you were wanting.

Firewalls

"A firewall is a system designed to prevent unauthorized access to or from a private network. Firewalls can be implemented as both hardware and software or a combination of both.

Used to prevent unauthorized Internet users from accessing private networks connected to the Internet, especially intranets. All messages entering or leaving the intranet pass through the firewall, which examines each message and blocks those that do not meet the specified security criteria".



19

24/2/2021

KAPLAN

19

Here are eight types of firewalls:

- Packet-filtering firewalls
- Circuit-level gateways
- Stateful inspection firewalls
- Application-level gateways (a.k.a. proxy firewalls)
- Next-gen firewalls
- Software firewalls
- Hardware firewalls
- Cloud firewalls

20

24/2/2021

KAPLAN

20

When deciding on which firewall to choose, most organizations would consider the following factors:

- The size of the organization.
- The resources available.
- The level of protection required.

21 24/2/2021 KAPLAN 21

Digital Certificates

"A digital certificate is an attachment to an electronic message used for security purposes, most commonly to verify that a user sending a message is who he or she claims to be and to provide the receiver with the means to encode a reply."

User gets a digital certificate from a recognized Certificate Authority (CA) such as VeriSign, GeoTrust and others. CA then is a trusted third party that is relied upon to verify the matching of public keys to identity such information.

22 24/2/2021 KAPLAN 22

Digital Certificates is used for electronic transactions such as e-mail, e-commerce and electronic funds transfers.

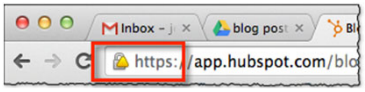
The most widely accepted standard for Digital Certificates is X.509.

“A digital certificate primarily acts like an identification card; something like a driver's license, a passport, a company ID, or a school ID. It basically tells other people who you are.”

Secure Sockets Layer and Secure HTTP 630

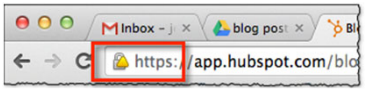
In the beginning, data was transmitted in plaintext on the Web whom anyone could read if they intercepted. For example, consumer credit card number. Consequently, Secure Sockets Layer (SSL) was created to protect user privacy by encrypting any data that goes between a user and a web server. As a result, intruders could only see scrambled mess if they intercepted.

SSL is able to provide privacy, authentication, and integrity to Internet communications which eventually evolved into Transport Layer Security (TLS).



Any website that implements SSL/TLS has "HTTPS" in its URL. This website includes an SSL/TLS certificate that is signed by a publicly trusted CA. Data is encrypted and transmitted across the web. Because of these, SSL/TLS and HTTPS allow users to securely transmit confidential information (e.g. credit card numbers, social security numbers, etc.) over the internet.

This means that anyone who intercepted this data will only see a mangled mix of characters that is almost impossible to decrypt.



25
24/2/2021
KAPLAN
25

Review Questions

1. Explain the purpose and scope of database security.
2. Discuss the role of the database administrator in database security and recovery.
3. Explain the following in terms of providing security for a database:
   a. authorization;
   b. authentication;
   c. access controls;
   d. views;
   e. backup and recovery;
   f. integrity;
   g. encryption;
   h. RAID technology.
4. Describe the approaches for securing DBMSs on the Web.

26
24/2/2021
KAPLAN

Exercises

1. Identify the types of security approach that are used by your organization to secure any DBMSs that are accessible over the Web.
2. You are contracted to deploy database security for the university accommodation system. Describe how you will approach the project.

REFERENCES

Thomas Connolly, Carolyn Begg (Sixth edition, Global edition) 2015, Database Systems. A Practical Approach to Design, Implementation, and Management , Pearson, USA.

KAPLAN

Introduction to Database Design and Development

Topic 10
Distributed DBMS

LEARNING OUTCOMES

Upon successful completion of this module, the student should be able to:

1. Describe the purpose of using Centralised relational database and Distributed relational database.
2. List the advantages and disadvantages of Distributed DBMS.
3. Explain the difference between Centralised and Distributed relational database.
4. Explain the differences between horizontal and vertical fragmentation schemes

2

KAPLAN

A database is a collection of related data where organizations used to store, manage and retrieve data.

There are fundamentally two types of databases; centralized and distributed.

A centralized database is a single database where multiple users can access it.

On the other hand, the distributed database separates the database into multiple files at various locations in the network.

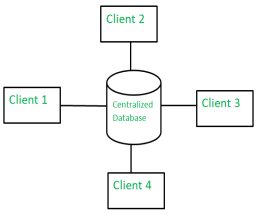
3

24/2/2021

KAPLAN

10.1 CENTRALIZED vs DISTRIBUTED DBMS

Centralized DBMS



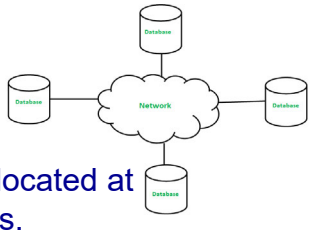
- A centralized database is basically a single database file stored and maintained at one location in the network.
- Multiple users can access this single database via an internet connection (LAN, WAN, etc).
- As there is only a single database file, it is easier to get a complete view of the data
- It has a higher usage, easier to manage and control, update and take backups of data.

4

24/2/2021

KAPLAN

10.1 CENTRALIZED vs DISTRIBUTED DBMS



Distributed DBMS

- A distributed database consists of two or more database files located at different sites in the network; database is split into multiple files.
- Users can access the nearest database file without interfering with one another.
- This increase the speed of retrieving data as different users can access and manipulate data relevant to them at the nearest locations.
- To ensure data consistency, DBMS must periodically synchronise the scattered databases.
- If one database fails, the system still runs as user can access other database files.

5

24/2/2021

KAPLAN

CENTRALIZED DATABASE
VERSUS
DISTRIBUTED DATABASE

CENTRALIZED DATABASE	DISTRIBUTED DATABASE
A type of database that contains a single database located at one location in the network	A type of database that contains two or more database files located at different locations in the network
Managing, updating and taking in backups of data is easier because there is only one database file	As there are multiple database files in a distributed database, it requires time to synchronize data
Requires time for accessing data because multiple users access the database file	Speed in accessing the data is higher because the data is retrieved from the nearest database file
If the database fails, the users do not have access to a database	If one database fails, the users can still access other database files
Has more data consistency and it provides the complete view to the user	Can have data replications, and there can be some data inconsistency

Visit www.PEDIAA.com

6

24/2/2021

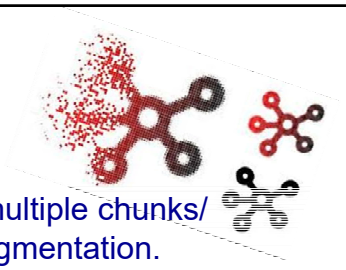
KAPLAN

10.2 ADVANTAGES & DISADVANTAGES OF DISTRIBUTED DBMS

TABLE 24.1 Summary of advantages and disadvantages of DDBMSs.

ADVANTAGES	DISADVANTAGES
Reflects organizational structure	Complexity
Improved shareability and local autonomy	Cost
Improved availability	Security
Improved reliability	Integrity control more difficult
Improved performance	Lack of standards
Economics	Lack of experience
Modular growth	Database design more complex
Integration	
Remaining competitive	

10.3 FRAGMENTATION



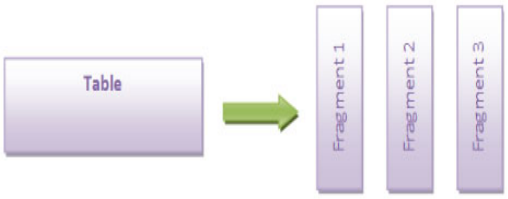
The process of dividing the whole database into a smaller multiple chunks/ tables and storing them in different locations is called as fragmentation.

Fragmentation process should be carrried out in such a way that the reconstruction from the fragments whenever required to the original database is possible.

Fragmentation can be of three types: *horizontal, vertical*, and *hybrid* (combination of *horizontal and vertical*).

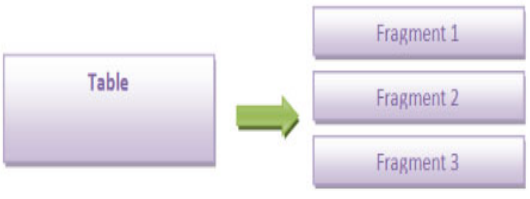
Vertical Fragmentation

Vertical fragmentation divides a relation (table) vertically into groups of columns i.e. subsets of tables. And in order to preserve reconstructiveness, each fragment should retain the primary key field(s). In addition, vertical fragmentation can enforce data privacy.



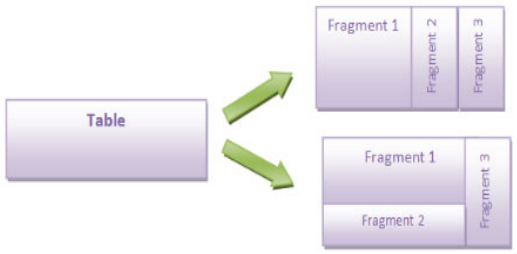
Horizontal Fragmentation

Horizontal fragmentation divides a relation (table) into the group of rows in accordance to values or user requirements to create subsets of tables. To maintain reconstructiveness, each horizontal fragment must have all columns of the original table.



Hybrid Fragmentation

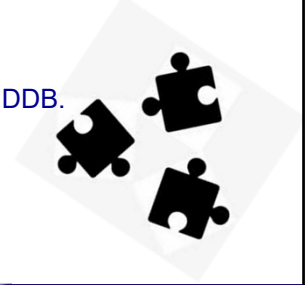
Hybrid fragmentation can be achieved by performing horizontal and vertical partition together. This technique is flexible, however, reconstruction of the original table is often a challenging and expensive task.



There are some useful benefits to fragmentation:

- **Easy usage of Data:** It makes most frequently accessed set of data near to the user. Hence these data can be accessed easily as and when required by them.
- **Efficiency :** It in turn increases the efficiency of the query by reducing the size of the table to smaller subset and making them available with less network access time.
- **Security :** It provides security to the data. That means only valid and useful records will be available to the actual user. The DB near to the user will not have any unwanted data in their DB. It will contain only those informations, which are necessary for them.

- **Parallelism :** Fragmentation allows user to access the same table at the same time from different locations. Users at different locations will be accessing the same table in the DB at their location, seeing the data that are meant for them. If they are accessing the table at one location, then they have to wait for the locks to perform their transactions.
- **Reliability :** It increases the reliability of fetching the data. If the users are located at different locations accessing the single DB, then there will be huge network load. This will not guarantee that correct records are fetched and returned to the user. Accessing the fragment of data in the nearest DB will reduce the risk of data loss and correctness of data.
- **Balanced Storage :** Data will be distributed evenly among the databases in DDB.



Review Questions

1. Discuss why processes used to design a centralized relational database are not the same as those for a distributed relational database.
2. Discuss the advantages and disadvantages of a DDBMS.
3. One problem area with DDBMSs is that of distributed database design. Discuss the issues that have to be addressed with distributed database design.
4. What are the differences between horizontal and vertical fragmentation schemes?

REFERENCES

Thomas Connolly, Carolyn Begg (Sixth edition, Global edition) 2015, Database Systems. A Practical Approach to Design, Implementation, and Management , Pearson, USA.