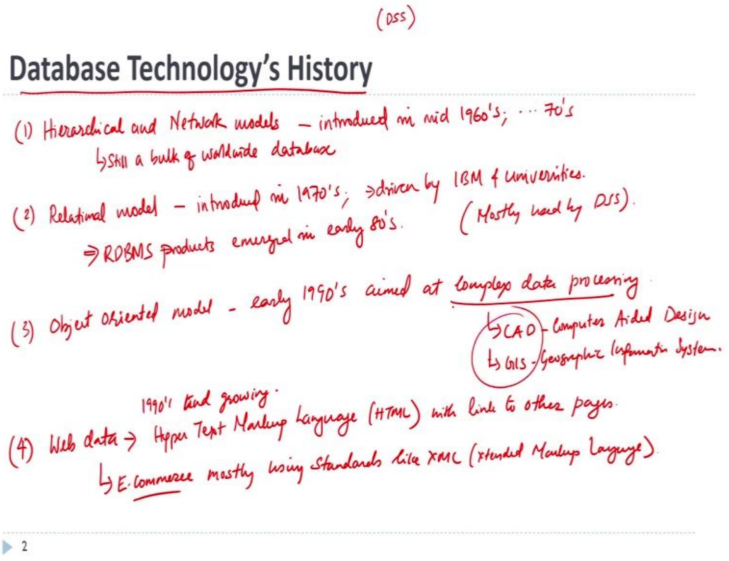
Computer Aided Decision Systems - Industrial practices using Big Analytics

Professor Deepu Philip
Professor Amandeep Singh
Department of Industrial & Management Engineering
Indian Institute of Technology Kanpur

Lecture - 9
Introduction to Database

Good evening. Welcome to yet another lecture of the ‘Web Based Decision Support System’ course and as promised in the previous lecture, we are now moving towards understanding the concept of database, the major aspects of database, how to create a database, how to design a database and how to manipulate the database? And also optimize the database.

(Refer Slide Time: 0:43)



(DSS)

Database Technology’s History

(1) Hierarchical and Network models – introduced in mid 1960's; ··· 70's
↳ Still a bulk of worldwide database

(2) Relational model – introduced in 1970's; ⇒ driven by IBM & universities.
⇒ RDBMS products emerged in early 80's. (Mostly used by DSS).

(3) Object oriented model - early 1990's aimed at complex data processing.
↳ CAD - Computer Aided Design
↳ GIS - Geographic Information System.

(4) Web data → Hyper Text Markup Language (HTML) with link to other pages.
1990's and growing.
↳ E-commerce mostly using Standards like XML (Extended Markup Language).

2

So, without much delay let us get into the major thing that is ‘Database Technologies History’ and this is important, especially when you are looking at DSS, because in a Corporate DSS this is very important aspect. So, some of the models are:

i) Hierarchical and Network Models- it was introduced in mid-1960’s until mid-1970’s. So, this is like mostly COBOL database etc.

- Still a bulk of worldwide database occurs in this model. Majority chunk is still part of this.

ii) Relational Model- introduced in 1970s, driven by IBM and universities and the RDBMS (Relational Database Management System) software products emerged in early 80s. So, the Relational model, which is the most popular one this is (mostly used by DSS).

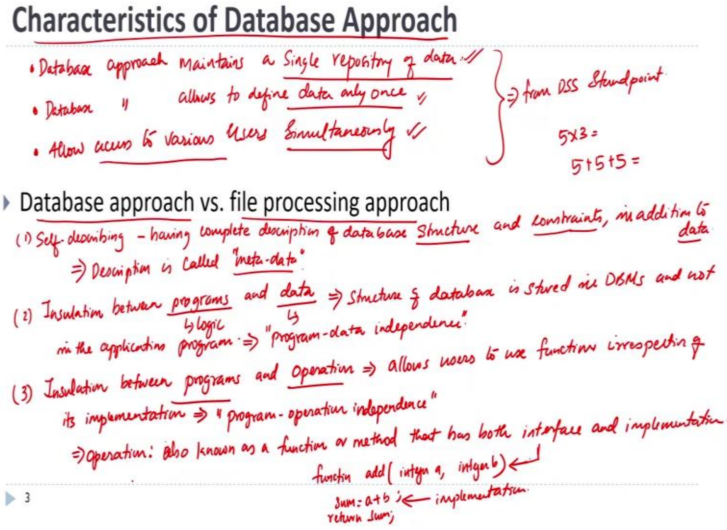
iii) Object-Oriented Model- You can call it as early 90's aimed at Complex Data Processing, like example of this is:

- CAD 'Computer Aided Design',
- GIS 'Geographic Information System'.

So, the Object-Oriented Model came in 1990s and now a lot of these specifics are coming as part of it and still gaining popularity.

iv) Web Data- Mostly we can think about as 'Hyper Text Markup Language' (HTML) with link to other pages. So, the entire website in the link to websites another type of a data, then you have something like also this contains something called E-commerce website is another example. Mostly using standards like XML (Extended Markup Language). So, these are the major technology. So, this Web Data again became popular in the 90's and growing. So, websites like E-commerce, eBay actually created a lot of these aspects related to and again remember this object oriented model of use also becoming popular because especially CAD, GIS systems are gaining popularity as part of this.

(Refer Slide Time: 5:27)



Characteristics of Database Approach

- Database approach maintains a Single repository of data. ✓✓
- Database " allows to define data only once ✓
- Allow access to various Users Simultaneously ✓✓

⇒ from DSS Standpoint

5x3 =

5+5+5 =

- Database approach vs. file processing approach

(1) Self-describing – having complete description of database Structure and constraints, in addition to data

⇒ Description is called "meta-data"

(2) Insulation between programs and data ⇒ Structure of database is stored in DBMS and not in the application program ⇒ "program-data independence"

↳ logic

↳

(3) Insulation between programs and Operation ⇒ allows users to use function irrespective of its implementation ⇒ "program-operation independence"

⇒ operation: also known as a function or method that has both interface and implementation

function add(integer a, integer b) ←

sum:=a+b; ← implementation

return sum;

3

Now, let us see why you need the ‘Characteristics of a Database Approach’. So, this is another aspect that we need to do. The main reason is the Database Approach. Why do we do database approach?

- It maintains a single repository of data. So, the main reason is you have a single location where it is and single repository.
- Database approach allows to define data only once. So, you do not need to keep on. You define it once for your application and that is it.
- Allow access to various users simultaneously. So, the major aspect is, you maintain a single repository of data rather than every decision making getting one individual copy of the data, you have a single repository of the data in which the entire data whatever is contained in that repository is defined only once and uniformly define and allow access to various users simultaneously at the same time. People can access this single repository of data.

So, that is the main characteristics of a database approach and these are from DSS standpoint. What is the major difference between why cannot we just use files, instead of why do we need to go with the database approach instead of the file processing approach? So, the main advantages of it is, the first advantage is

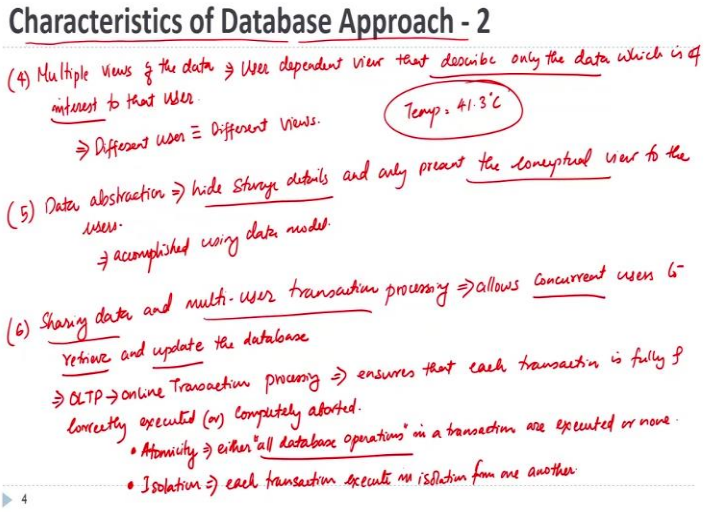
i) Self-Describing- having complete description of database structure, and constraints in addition to data. In addition to just defining the data, you have a complete description of the structure of the entire database and the constraints or the restrictions that are associated with the data. So, the complete description is called as… the name for it is called ‘Metadata’. When somebody says ‘Metadata’ we are talking about is a complete description of the data. That is the first advantage of using the database approach over file approach.

ii) Insulation between programs and data- What is program? Program in another way is logic written in a computer language. So, that the computer can do and the data is something upon which the program acts upon. So, the ‘Implication’ is structure of database is stored in DBMS and not in the application program. So, this term is ‘program-data independence’. So, it insulates programs and data that separates the logic and the data.

iii) Insulation between programs and operation or data operation- So, operation is the sequence the way in which you are operating the program so, what it does is it allows users to use functions irrespective of its implementation. So, it does not matter if you say 5 multiplied by 3 whether this multiplied in this way or this 5 plus 5 plus 5 (as shown in slide), whichever ways you have implemented it does not matter, you do not care how the operation is implemented. So, what is this is known as ‘Program Operation Independence’. Database allows you to do that.

❖ Sometimes people are always asked what is ‘Operation’? And for this practical purpose in this DSS course, we call it as also known as a function or method that has both interface and implementation. If you write something like function add integer ‘a’ integer ‘b’’ it is a function and what happens is sum is equal to ‘a’ plus ‘b’ and you say a return some if you do something like this (as shown in the slide). So, what actually happens is this is the interface of the function and this is the implementation of the function (as shown in the slide). Both ‘Interface’ and ‘Implementation’ that is known as an ‘Operation’.

(Refer Slide Time: 12:46)



Characteristics of Database Approach - 2

(4) Multiple views of the data ⇒ User dependent view that describe only the data which is of interest to that user.

⇒ Different user ≡ Different views.

Temp = 41.3°C

(5) Data abstraction ⇒ hide Storage details and only present the conceptual view to the users.

⇒ accomplished using data model.

(6) Sharing data and multi-user transaction processing ⇒ allows Concurrent users to retrieve and update the database

⇒ OLTP → online Transaction processing ⇒ ensures that each transaction is fully & correctly executed (or) completely aborted.

- Atomicity ⇒ either "all database operations" in a transaction are executed or none.
- Isolation ⇒ each transaction executes in isolation from one another.

4

So, now, we get into the we continue on why we need to use database approach over the file approach and one of the major. So, we already seen the three.

iv) Multiple Views of the Data- It means is User Dependent view that describe only the data which is of interest to that user. Critically whatever the user wants, whatever it is of interest or importance to that user only describe and display that data.

Which implies different user translates to different views. So, the fourth part is the multiple view of the data.

v) Data Abstraction- What it implies is hide storage details and only present the conceptual view to the user. What happens here is you do not care in which track, which sector of the hard disk this data is stored. It is hidden from you and you are basically saying that here is. So, if you say temperature is 41.3 degrees Celsius. How this data is stored is immaterial to you but you just remember that or you just told that the value of the variable temperature is our 41.3 degree Celsius. And, you can remember this as it is accomplished using data model.

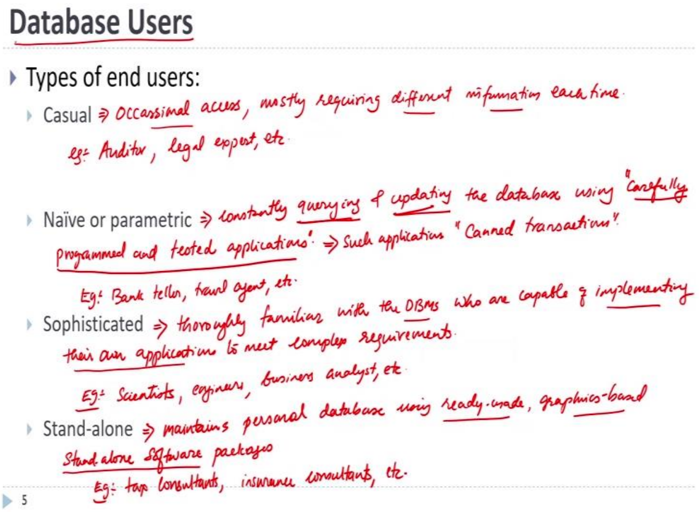
vi) Sharing Data and Multi User Transaction Processing- It allows for the sharing of the data and it also allows for multi user to process a transaction. So, which implies that it allows concurrent users to retrieve and update the database. You can retrieve or ask query withdraw information and you can also update the information that is available in the database. How concurrently not one at a time simultaneously.

❖ One of the key words you can heard about it does OLTP. It stands for (Online Transaction Processing). So, what it does? It ensures that each transaction is fully and correctly executed or completely aborted. Either you completely execute or you do not execute at all. So, there is two parts to this.

- Atomicity- implies either all database operations in a transaction are executed or not. Atomicity implies all operations in the database operations in a transaction are executed otherwise none of them are executed.

- Isolation- Which means each transaction execute in isolation from one to another. So, when one transaction is done, it isolates itself from the other transaction that is going to happen.

(Refer Slide Time: 18:51)



Database Users

- Types of end users:
  - Casual ⇒ Occassimal access, mostly requiring different information each time.
    eg: Auditor, legal expert, etc.
  - Naïve or parametric ⇒ constantly querying & updating the database using "carefully programmed and tested applications". ⇒ Such applications "Canned transactions".
    Eg: Bank teller, travel agent, etc.
  - Sophisticated ⇒ thoroughly familiar with the DBMS who are capable of implementing their own applications to meet complex requirements.
    Eg: Scientists, engineers, business analyst, etc.
  - Stand-alone ⇒ maintains personal database using ready-made, graphics-based Stand alone Software packages
    Eg: tax consultants, insurance consultants, etc.

5

So, then we get into what we call us Database Uses. So, for us there are so many types of database users. But at this point we try to discuss only four types of database users for induces as part of the database.

i) Casual User- Occasional access, mostly requiring different information each time. So, these users they do not access every day, they access it occasionally and each time they access they look for different information. Let us take few examples of this. Like an auditor of a company, legal expert, etc. are examples of such casual uses.

ii) Naive or Parametric User- They are users that are constantly querying and updating (Querying also means retrieving) the database using carefully programmed and tested applications. So, the they consistently or constantly query and update the database and they use carefully programmed and tested applications. What are this one is known as? Such apps or applications are called ‘Canned Transactions’. It is like canned beans. So, everything is in one canned. Example of this is a bank teller, travel agent etc.

iii) Sophisticated User- Who is thoroughly familiar with the DBMS (Database Management System), who are capable of implementing their own applications. They can program and implement applications to meet complex requirements. So, these are people who are capable. They are very familiar with the DBMS. And, they can implement, they can code their own application to meet whatever the sophisticated details. Example of this is scientists, engineers, business analyst, etc.

iv) Standalone- They maintain personal database using a readymade, graphics-based standalone software packages. The examples of this are things like tax consultants, insurance, consultants, etc. So, they have a very specific database. That is readymade for a very specific application mostly graphics-based or GUI driven and these are standalone software packages.

(Refer Slide Time: 23:55)

Why Use DBMS?

Now, we will get into what we called as when to use a Database Management System and when not to use a Database Management System. And then we will start with an example. So, the coming lecture what we will do is we will quickly spend when to use a database and when not to use a database and we will start with an example and then we move into something called an ER diagram and how to use an ER diagram to conceptualize the database and then we go from there. Thank you very much.