Relational Database Management System

1.1

Introduction:

1) Definitions:

a) Database: A collection of inter-related data which is used to retrieve, insert and delete the data efficiently. It is also used to organize the data in the form of a table, schema, views and reports, etc.

b) Database Management System: DBMS is a software which is used # to manage the database. It provides an interface to perform various operations like database creation, storing data in it, updating data, creating a table in the database, etc. It also provides protection and security to the database.

c) Relational Database management System: RDBMS is a DBMS based on the relational model (will learn subsequently).

2) Uses: DBMS are used to manage data collections that are:

a) Highly Valuable.
b) Relatively large
c) Accessed by multiple users and applications, often at the same time.

3) Applications of DBMS:

a) Enterprise Information
   i) Sales: For customer, product, and purchase information.
   ii) Accounting: For payments, receipts, account balances, assets, and other accounting information.
   iii) Human Resources: For information about employees, salaries, payroll taxes, and benefits, and for generation of paychecks.

b) Universities:
   i) Courses: Course identifiers, title, department, course number, etc.
   ii) Students: Student-Identifier, name, address, phone, etc.

c) Etc.

4) When we access a website, information about us may be retrieved from a database to select which advertisements we should see. Almost every interaction with a smartphone results in some sort of database access. Data bout our web access may be stored in a database. Although most people are not even aware that they are dealing with a database, accessing databases forms an essential part of almost everyone's life today.

There are two modes in which databases are used:

a) Online transaction Processing: A large number of users use the database, with each user retrieving relatively small amounts of data, and performing small updates.

b) Data Analytics: Processing of data to draw conclusions, and infer rules or decision procedures, which are then used to drive business decisions.
Eg: Bank needs to decide whether to give loan to a loan applicant based on his history and bank account information.

Characteristics of Databases:
(Topic explored further in the end)

1) It uses a digital repository established on a server to store and management the information.
(Digital repository is an instrument for managing and storing digital content).

2) It can provide a clear and logical view of the process that manipulates data.

3) DBMS contains automatic backup and recovery procedures.

4) It contains ACID properties which maintain data in a healthy state in case of failure.
(ACID will learn soon).

5) It can reduce the complex relationship between data.

6) It is used to support manipulation and processing of data.

7) It is used to provide security of data.

8) It can view the database from different viewpoints according to the requirements of the user. (Will learn soon).

Comparison of File system and Database approach:

1) Definitions:

a) File System: A decentralized approach. Each department stores and controls its own data.

b) Database approch: Well organized collection of data that are related in a meaningful way which can be stored only once in a system.

2) Basis	DBMS Approach	File-System Approach
a) Meaning	The user is not required to write the data.	The user has to write the procedures for managing the data.

Basis	DBMS Approach	File-System Approach
b) Sharing of data	Due to centralized approach, data sharing is easy.	Data is distributed in many files, & it may be of different formats, so it isn't easy to share the data.
c) Data Abstraction	DBMS gives an abstract view of data that hides the details.	Provides the detail of the data representation & storage of data.
d) Security and Protection	DBMS provides a good protection mechanism.	It isn't easy to protect a file under the file system.
e) Recovery Mechanism	DBMS provides a crash recovery mechanism, i.e., DBMS protects the user from system failure.	The file system doesn't have a crash mechanism The content of a file can be lost forever.
f) Manipulation Techniques	DBMS provides a wide variety of sophisticated techniques to store and retrieve the data.	Can't efficiently store and retrieve the data.
g) Concurrency Problems	Takes care of concurrent access of data using some form of locking.	Concurrent access can cause incorrect updates.

h)	Cost	The database system is expensive to design.	Cheaper to design.
i)	Data redundancy & inconsistency	Due to centralization, redundancy & inconsistency are controlled.	There exists a lot of duplication of data which may lead to inconsistency.
j)	Structure	Complex to design.	Simple Structure.
k)	Data independence	Data independence is there	No data independence
l)	Integrity Constraints	Integrity constraints are easy to apply.	Integrity constraints are difficult to apply.
m)	Data Model	There are 3 types of data models	No concept of data models
n)	Flexibility	Making changes is more easy.	Flexibility is less compared to DBMS.
o)	Eg:	Oracle, SQL, etc.	Cobol, C++, etc.

{ Some points will be understood later }

3) Disadvantages of keeping organizational information in a file-processing system.

a) Data Redundancy and Inconsistency.
Since data is not centralized, i.e., different programmers create the files and application programs over a long period, the various files are likely to have different structure and programming languages. The same information can be duplicated in several files.

Student Info → Robocon files
Student Info → Onyx files

Student address changed.
Robocon updated the address.
Onyx hasn't yet updated.

Student info → Robocon with new address
Student info → Onyx with old address

Data is inconsitent since both data are different.

Since student's information was stored in multiple files (Data redundancy), it lead to inconsistent data (this is data inconsitency).

b) Difficulty in accessing data.
Conventional file-processing environments do not allow needed data to be retrieved in a convenient and

efficient manner.
Example, File systems do not have application programs to do every task that could potentially be needed to do already, if some tasks would have to be done manually or a programmer would have to write the necessary application code each time a new task comes in hand.
Eg: Sorting student list according to marks.

c) Data Isolation: Since data are scattered in various files, and files may be in different formats, writing new application programs to retreive the appropriate data is difficult.

d) Integrity Problems: Arises when the database fails to satisfy certain integrity constraints.
Eg: Bank balance should not go below 1000, but this is called consitency constraint.
But now the bank wants to allow till 100. This new consistency constraint should be updated in the existing files. It is difficult to make these changes

e) Atomicity Problems: If money is debited from your account but not credited to the account you were sending money to, it is an example of atomicity problem. The money should either be debited & credited both or it should not be debited at all.

That is, the funds transfer should be atomic, it must happen in its entirety or not at all. It is difficult to ensure atomicity in a conventional file based system.

d) Concurrent - access anamolies:
Suppose A debits 100 from an account & at the exact time B debits 50 from that account, it is posible that instead of 150, only 100 or 50 is debited from the account due to the concurrent access.
To guard against this possibility the system must maintain some form of supervision. This supervision is difficult with file based system.

e) Security Problems: Not every user of the database system should be able to access all the data. But since application programs are added to the file-processing system whenever necessary or needed, enforcing such security constraints is difficult.

These difficulties prompted the need for DBMS. DBMS solves the problems with file processing systems.

{ACID = Atomicity, consistency, isolation & durability}

Users of database System

1) There are 4 different types of database system users, differentiated by the way they interact with the system. Different types of UIs have been designed for the different types of users:

a) Naive Users: Unsophisticated users who interact with the ~~interface~~ system using predefined user interfaces, such as web or moblie applicabons. The typical user interface for naive users is a forms interface, where the user can fill in appropriate fields of the form.

b) Application Programmers: Computer professionals who write application programs. Application programmer can choose from many tools to develop user interfaces. (~~Frontend~~ Full Stack Programmers basically).

c) Sophisticated Users: Users who have their own way of accessing the database. They don't interact with the user interface and they don't write the program code. They interact with the database by writing SQL qureries, directly through the query processor.

d) Database Administrator: The DBA has central control of both the data and the programs that access those data. DBA is also responsible for providing security to the database

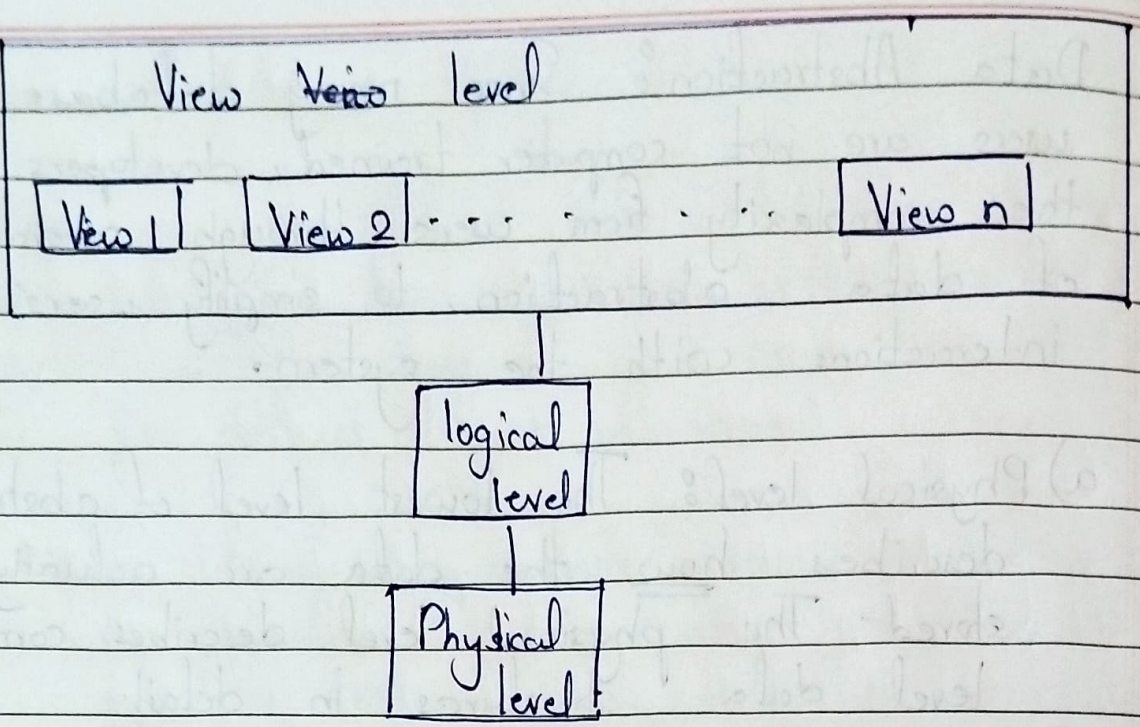
2) Data Abstraction: Since many database-system users are not computer trained, developers hide the complexity from users through several levels of data abstraction, to simplify users' interactions with the system.

a) Physical level: The lowest level of abstraction describes how the data are actually stored. The physical level describes complex low level data structures in detail.

b) Logical level: The next-higher level of abstraction describes ~~to~~ what data are stored in the database, and what relationships exist among those data.
Database administrators, who must decide what information to keep in the database, use the logical level of abstraction.

c) View level: Logical level uses simpler structures, but the complexity remains because of the variety of information stored in a large database.
The view level (highest level) of abstraction exists to simplify user interaction with the system further.
The system may provide many views for the same database.

The database system allows application developers to store and retrieve data using the abstractions of the data model and converts the abstract operations into operations on the low-level implementation.



The 3 levels of data abstraction.

3) Instances and Schema:

a) Instance: Databases change over time as information is inserted and deleted. The collection of information stored in the database at a particular moment is called an instance of the database.

b) Schema: The overall design of the database is called the schema.

Example: A database schema corresponds to the variable declarations and type definitions in a program. Each variable has a particular value at a given instant. The values of the variable in a program at a point of time correspond to an instance of the database schema.

Database systems have several schemas, partitioned according to the levels of abstraction:

i) Physical Schema: Describes the database design at the physical level

ii) Logical Schema: Describes the database design at the logical level.

Eg: The database consists of information about a set of customers and accounts in a bank and the relationship between them. The type of information is not mentioned in the above example.

Concerns when using an enterprise database.
(Topic explored further in the end)

1) Definition: An enterprise database is used by enterprises and large organizations to manage their huge collection of data.
An enterprise DBMS is such in which 100 to 10,000 individuals can access simultaneously.

2) A few concerns are as follows:

a) Deployment failures: Causes the wrong set of files to get deployed to site folder.
Most pre-deployment tests for databases just check the functionality whether the database is doing what it's supposed to do, but they don't check if the database is doing something its not supposed to do.

b) Privilege - based issues:
It is very important to give only the required privileges or access to the users or applications of the database.

c) SQL injections:
SQL injections is a common web and data hacking technique, which involves placing malicious code in the database through vulnerable SQL data input channels.

d) ~~Broken Databases~~:

d) Denial of Services: A form of cyber attack wherein a hacker or an intruder prevents legitimate users from accessing certain resources or a service by temporarily or indefinitely disrupting the server host.

1.2

Data Independence:
The capacity to change the schema at one level of a database system without having to change the schema at the next higher level.
There are 2 types of data independence:
a) Logical data independence:
Logical data independence is defined as the ability to modify the logical level without affecting the view levels or the application program.
Example: Changes in the logical level: adding new

attributes to a relation, deleting existing attributes of the relation, etc. We do not want to change any application or programs that do not require to use the modified attribute.

b) Physical data independence:
Defined as the ability to make changes in the structure of the lowest level of the database management system without affecting the higher-level schemas. Hence, modification in the physical level should not result in any changes in the logical or view levels.
Example of changes in physical level: creating a new file, storing the new files in the system, creating a new index, etc.

When a schema at a lower level is changed, only the mappings between this schema and higher level schemas need to be changed in a DBMS that fully supports data independence.
The higher level schemas themselves are unchanged.

Database System Architecture:

1) Database Languages:
A database system provides a data-definition language (DDL) to specify the database schema and a data-definition manipulation language (DML) to express database queries and updates.
DDL and DML, in practice, constitute to form one language, eg: SQL.

a) Data- Definition Language :

Definition: We specify a database schema by a set of definitions expressed by a special language called a data definition language (DDL).

b) Data- Manipulation Language:
DML is a language that enables users to access or manipulate data as organised by the appropriate data model.
The types of access are:
i) Retrieval of information stored in the database.
ii) Insertion of new information into the database.
iii) Deletion of information from the database.
iv) Modification of information stored in the database.

2) Database Engine

a) ~~Storage~~ A database system is partitioned into 3 functional components:

a) Storage Manager
b) Query Processor
c) Transaction Manager.

a) Storage Manager: Large enterprises could have databases that could reach into the multi-petabyte range. (petabyte = 1024 terabytes) Since the main memory cannot store this much information, and since the disks contents

of main memory are lost in a system crash, the information is ~~used~~ stored on disks.

The storage manager is the component of a database system that provides the interface between the low-level data stored in the database and the application programs and queries submitted to the system.

The storage manager is responsible for storing, retriving and updating data in the database.

The components of a storage manager include:

i) Authorization and integrity manager: ~~Theat~~ Tests for the satisfaction of integrity constraints and checks the authority of users to access data.

ii) Transaction Manager: Ensures that the database remains in a consistent state despite system failures, and the concurrent transaction executions proceed without conflicts.

iii) File manager: Manages the allocation of space on disk storage and the data structures used to represent information stored on disk.

iv) Buffer manager: Responsible for fetching data from disk storage into main memory,

and deciding what data to cache in main memory. Enables the database to handle data sizes that are much bigger than the size of main memory.

b) Query Processor: Helps the database system to simplify and facilitate access to data. Allows users to obtain good performance while being able to work at the view level and not be burdened with understanding the physical level details of the implementation of the system.

The database system itself translates updates and queries written in non-procedural language into a sequence of operations at the physical level.

The query processor components include:

i) DDL interpreter: Interprets DDL machine statements and records the definitions in data dictionary. (will define data dictionary later).

ii) DML compiler: Translates DML statements in a query language into low level instruction (machine language) so that the query-evaluation engine can understand.

iii) Query evaluation engine: Executes low level instructions generated by the DML compiler.

iv) Application program object code:
Contains the code sent by a compiler, which are machine readable instructions that is processed by the eq query evaluation engine.

c) Transaction Management: (not shown in detail in diagram)

A transaction should ~~be~~:

i) ~~Atomic~~ be atomic

ii) ~~Consistent~~ ensure database consistency

iii) Durable (The new values of data should persist after updates)

(Points (i) and (ii) are explored before)

A transaction manager consists of:

i) Concurrency-control manager: The responsibility of concurrency-control manager is to control the interaction among the concurrent transactions, to ensure the consistency of the database.

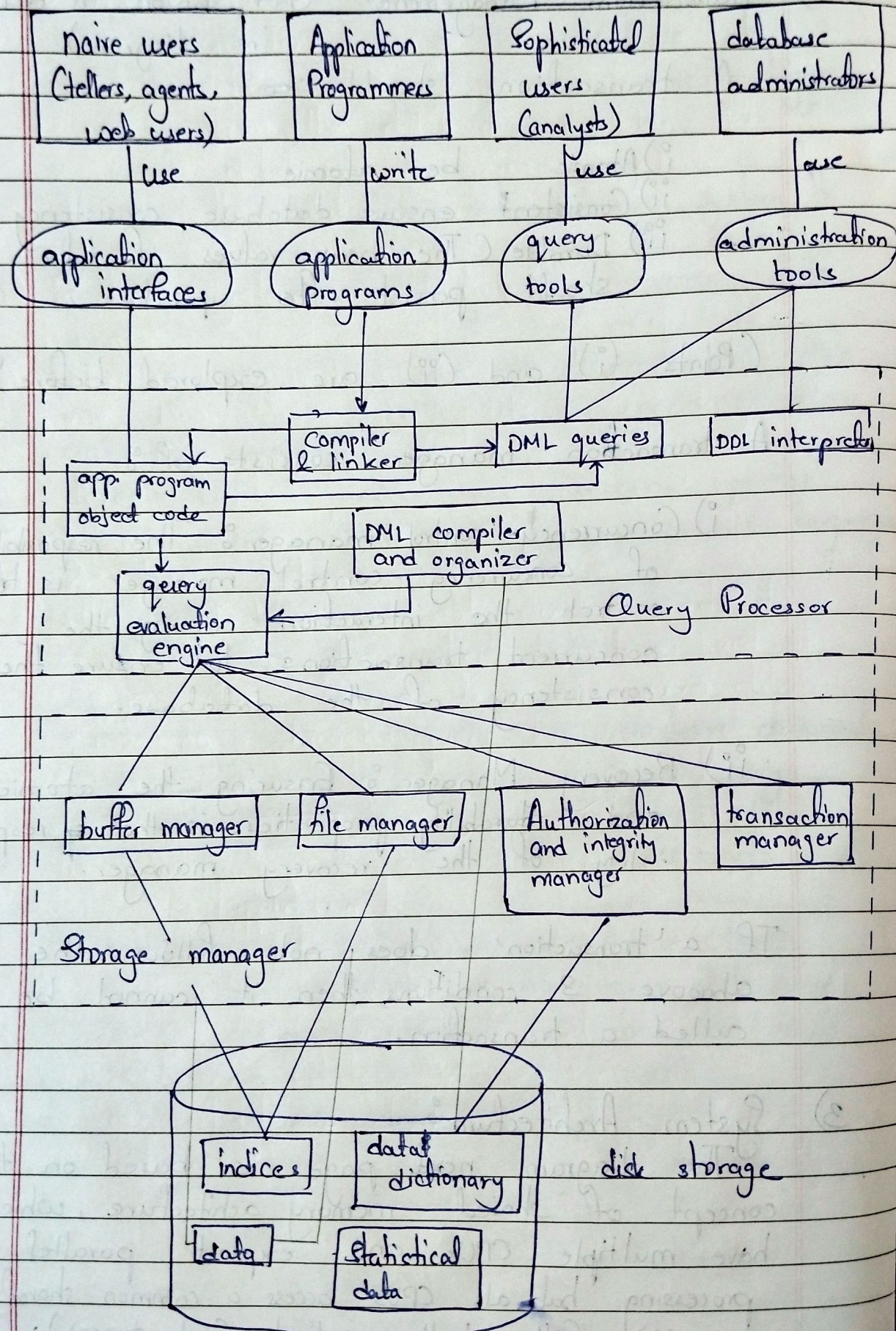
ii) Recovery Manager: Ensuring the atomicity and durability properties is the ~~of~~ responsib-ility of the recovery manager.

If a 'transaction' does not follow the abooove 3 conditions, then it cannot be called a transaction.

3) System Architecture:

The diagram next page is based on the concept of shared memory achitecture, which have multiple CPUs and exploit parallel processing but all CPUs access a common shared memory (the bottom cylinder (disk storage)).

(lol diagram ppt se dekh)



1) Disk Storage:

a) Data files: Stores the database itself.

b) Data dictionary: It contains the information about the structure of the database or database schema.

c) ~~Schema~~: Indices: Can provide fast access to the ~~database~~ data items.
Eg: Page no.s in a textbook

d) Statistical data: Statistical database contains details of huge populations, it is used mainly to produce statistics on various populations.

2) 2 and 3 tier architectures:

a) ~~Two - tier architecture~~:

a) Definitions:

i) Client: A client is a program that uses services that other programs ~~(or~~ (namely servers) provide. The client makes a request for a service, and the server performs that service.
The client is isolated from the need to know anything about the actual resource manager.

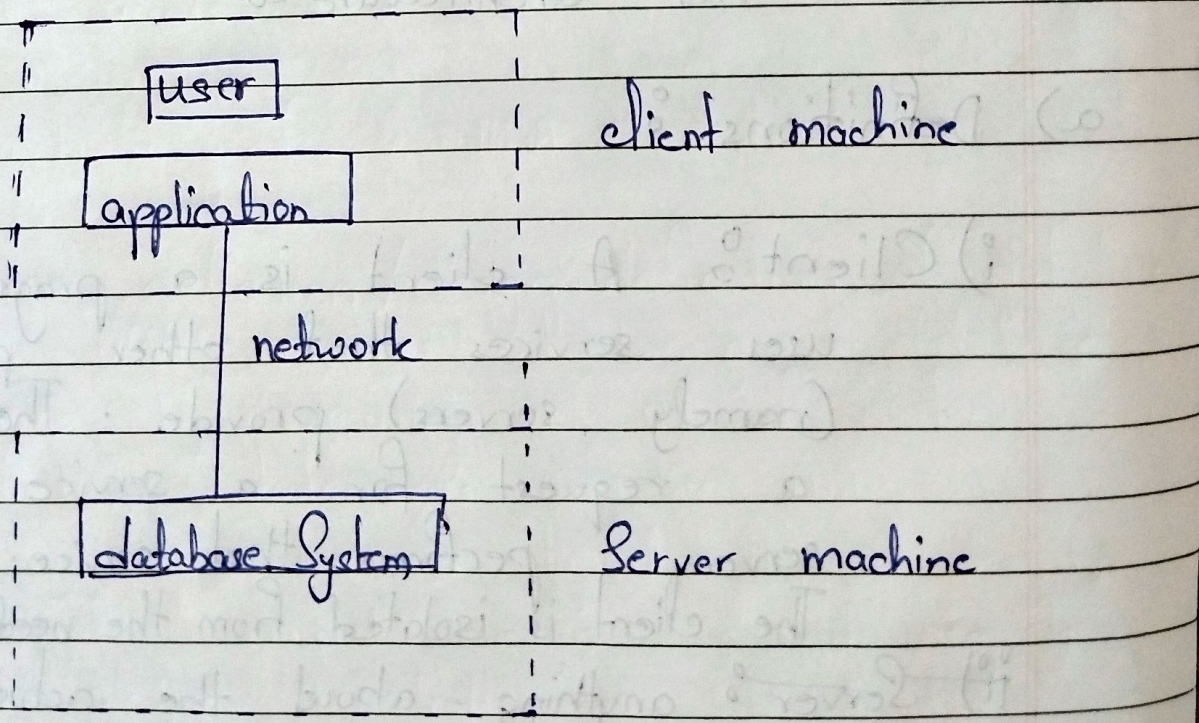
ii) ~~Server~~:

ii) Server: Database servers are networked computers on a network dedicated to database storage and data retrieval from the database. It holds the database management system and the databases.

b) 2 tier system architecture:

* Database applications are usually partitioned into two or three parts, depending of on the partion we have two or three tier system architectures.

b) 2- tier system architecture:
The application is not divided and it resides in the client machine and it calls for data access service from the server database System machine through query language. system



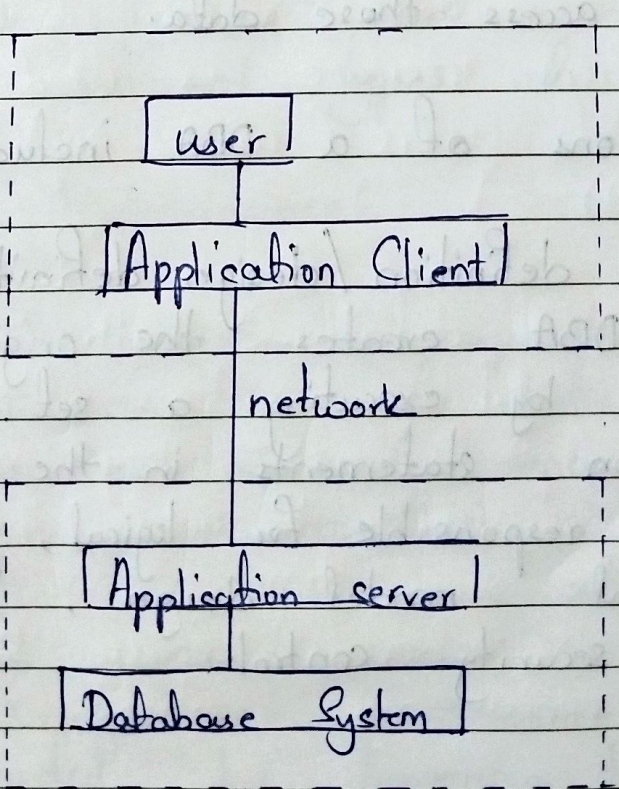
c) 3 - tier system architecture:

The client machine only acts like a frontend and, unlike in 2-tier system architecture, does not call for data access service from the database system using query language. Instead, it communicates with an application server, usually through forms interface. The application server in turn communicates with a database system to access data.

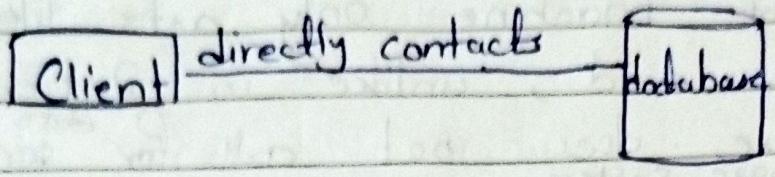
{ Here application is divided into two parts: application client and application server }

* The interface mentioned above can come in the forms of ODBC, JDBC, etc.

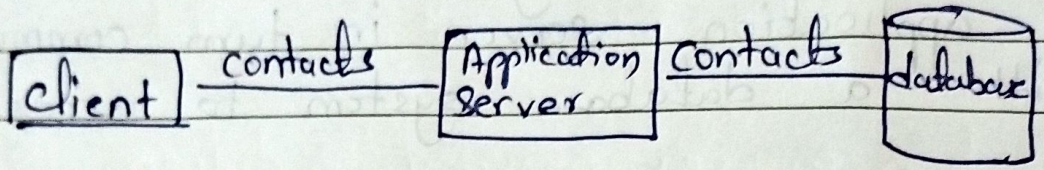
Three tier applications are more appropriate for large applications and for applications that run on the world wide web.



In short, in two tier architecture



In three tier architecture



(above 2 diagrams only for understanding)

Database Administrator:

1) Definition: The database administrator (DBA) ~~controls~~ has a ~~cenel~~ central control over over the system. He has control over both, the data and the programs that access those data.

2) The functions of a DBA includes:

a) Schema definition / design definition of database: The DBA creates the original database schema by executing a set of data definition statements in the DDL. He is responsible for logical, physical design, external model design, and integrity and security control.

b) Storage structure and access method definition:

Access method - The portion of a computer's operating system responsible for formatting datasets.

Storage structure - Defines storage device hierarchy, etc.

DBA specifies some parameters that define the physical organisation of the data and the indices to be created.

c) Schema and physical-organization modification:

The DBA can change the schema or the physical-organisation to reflect the changing needs of the organization or to improve performance.

d) Granting of authorization for data access:

DBA can regulate which parts of the database various users can access, by granting different types of authorization.

e) Routine Maintanance:

The DBA has to:-

i) Periodically back up the database to prevent loss of data.

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

iii) Ensuring that performance is not degraded by expensive tasks from users.

Extention of Previous Topics:

Characteristics of database:
(Some more important points)

1) Self-describing nature of database system:
DBMS not only contains the database itself, but also metadata which defines and describes the data and relationships between tables in the database.

2) Insulation between program and data:
As we saw before, any change in the physical structure of the database does not impact the application program (physical data independence).

3) Data abstraction:
Does not include details of how data is stored or how operations are implemented.

4) Supports multiple views of data:
A different interface is used for different types of users (naive, sophisticated, programmers, DBA).

5) Sharing of data & multiuser system:
Current database systems are designed for multiple users. They may allow many users to access the same database at the same time.
(This is achieved because DBMS has concurrency control)

6) Access flexibility and security:
Unauthorized access is restricted.
(eg: naive users can't access DBA's interface)

7) Transaction Processing:
A transaction is a program including a collection of
The operations performed in a transaction include operations like, insert, delete, update or retriere data.
It is an atomic process.

8) Backup & recovery facilities:
To protect from data loss the database system provides a separate process for backing up & recovering data.

Concerns when using an enterprise database
(Some more important points)

1) Enterprise Vulnerability:
Database has to be kept safe against destruction, unauthorized modification, etc.

2) Confidentiality, privacy and security:
Centralized database must be protected from various security threats.

3) Data Quality: Should control data quality & maintain integrity.

4) Cost of building & using a DBMS.
5) Should be able to easily change the database when necessary.