Unit-1

Parallel & Distributed Database

Database System Architectures

- Database systems can be centralized, where one server machine executes operations on the database.
- Database systems can also be designed to exploit parallel computer architectures.
- Distributed databases span multiple geographically separated machines.

Centralized Database system

- Centralized database systems are those that run on single computer system.
- Such database systems span a range from single user database systems running on mobile devices or personal computers to high performance database systems running on a server with multiple CPU cores & disks & large amount of main memory that can be accessed by any of the CPU cores.
- It widely used for enterprise-scale applicati
- We distinguish two ways in which computers are used as - single user systems
  - multiuser systems

- single user systems
  - smartphones & personal computers are example of single user systems.
  - A typical single user system is a system used by a single person, usually with only one processor with multiple cores & one or two disks

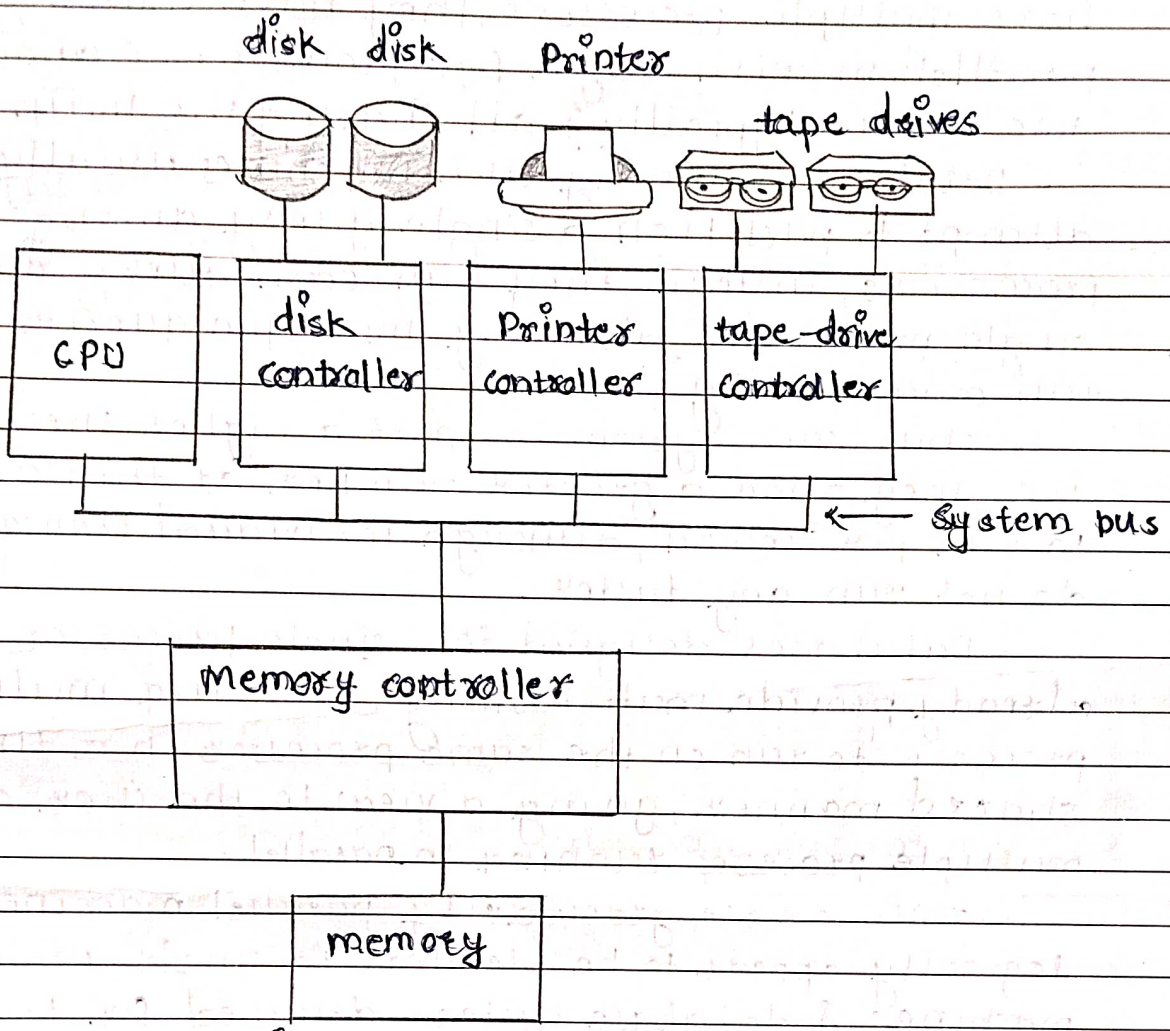
Multiuser systems

- A typical multiuser system on the other hand, has multiple disks, a large amount of memory & multiple processors.
- Such systems serve a large number of users who are connected to the system remotely & they are called server systems.

- Database system designed for single user sys usually do not provide many of the facilities th multiuser database provides.
- In particular, they may support very simple concurrency control schemes, since highly concurrent access to the database is very unlikely.
- Provisions for crash recovery in such system may also be either very basic or even absent in some cases. such systems may not support SQL & may instead provide an API for data access., such database systems are referred to as embedded databases, they are designed to be linked to a single appln program & they accessible only from that applicatio
- multiuser database systems support full transactional features that are us databases are usually designed as servers, which service requests received from application programs, the request could be in the form of SQL queries or they could be requests for retrieving, storing or updating data specified using API

- centralized database systems are those that run on a single computer system & do not interact with other computer system. such database system running on personal computers to high performance database systems running on high end server systems.
- We distinguish two ways in which computers are used as single user systems & multiuser system. Personal computers & workstations fall into the first category. A typical single user system is a desktop unit used by single person, usually with only one person using the machine at a time.
- A typical multiuser system on other hand has a more disks & more memory, may have multiple CPUs & has a multiuser operating system. It serves a large number of users who are connected to the

system via terminals.



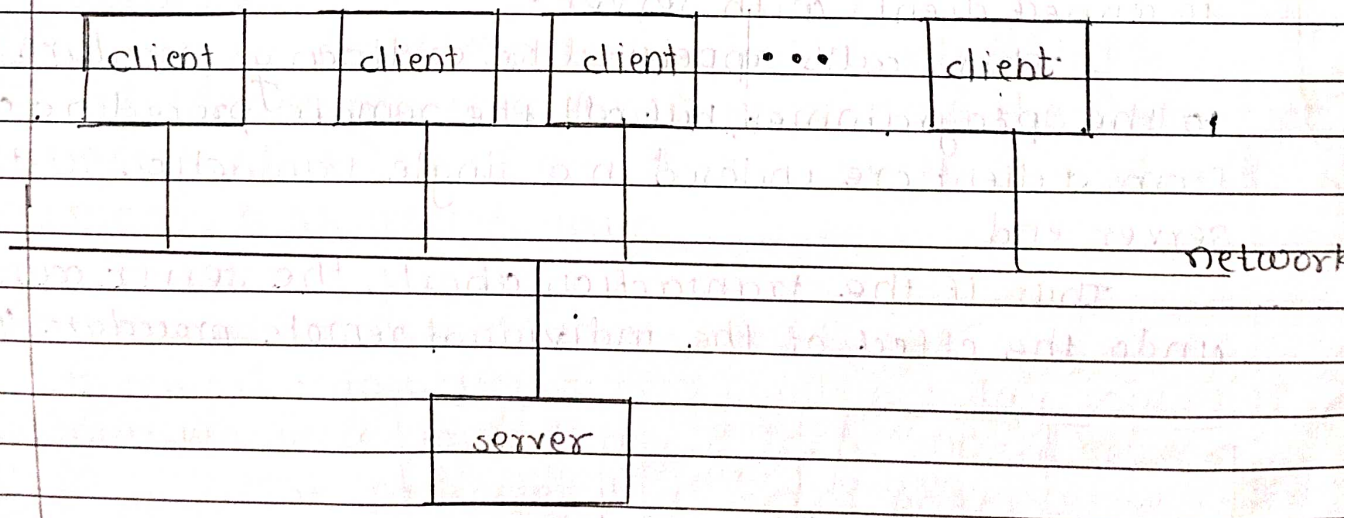
centralized computer system

- Database systems designed for use by single users usually do not provide many of the facilities that a multiuser database provides.
- In particular, they may not support concurrency control, which is not required when only single user can generate update.
- Provisions for crash recovery in such system are either absent or primitive example, they may consist of simply making a backup of the database before any update.
- Many such systems do not support SQL & provide a simpler query language such as variant of QBE.
- In contrast, database systems designed for multiple user system support the full transactional features.

- Although general purpose computer systems have multiple processors, they have coarse-grain parallelism, with only a few processors (about two to four typically) all sharing the main mem
- Databases running such machines usually do not attempt to partition a single query among the processors, instead, they run each query on a single processors, allowing multiple queries to run concurrently.
- Thus, such systems support a higher throughput, i.e, they allow a greater number of transaction to run per second, although individual transactions do not run any faster.
- Databases designed for single processor machin already provide multitasking, allowing multiple processes to run on the same processor in a time shared manner, giving a view to the user of multiple processes running in parallel.
- Thus coarse granularity parallel machines logically appear to be identical to single-processor machines & database systems designed for time shared machines can be easily adapted to run on them.
- In contrast, machines with fine granularity parallelism have a large no. of processors & DB systems running on such machines attempt to parallelize single tasks submitted by users.

Client server system

- As personal computers became faster, more powerful & cheaper there was a shift away from the centralized system architecture.
- Personal computers supplanted terminals connected to centralized systems. Correspondingly, personal computers assumed the user interface functionality that used to be handled directly by the centralized systems. As a result, centralized systems today act as server systems that satisfy requests generated by client system.
- Database functionality can be broadly divided into two parts: front end & back end as in figure.
- The back end manages access structures, query evaluation & optimization, concurrency control & recovery.
- The front end of database system consists of tools such as forms, reports writers & graphical user interface facilities.
- The interface between the front end & back end is through SQL or through an application program.



Client-server system

- standards such as ODBC & JDBC, were developed to interface clients with servers.
- Any client that uses the ODBC or JDBC interfaces can connect to any server that provides the interface.
- In earlier generation database systems, the lack of such standards necessitated that the front end & back end be provided by the same software vendor. With the growth of interface standards, the front end user interface & the back end server are often provided by different vendors.
- Application development tools are used to contruct user interfaces. they provide graphical tools that can be used to construct interfaces without any programming.
- Some of the popular application development tools are power-Builder, Magic & Borland Delphi, visual Basic is also widely used for application development.
- Further certain application programs, such as spreadsheets & statistical-analysis packages use the client-server interface directly to access data from a back end server.
- In effect they provide front ends specialized for particular tasks. Some transaction processing systems provide a transactional remote procedure call interface to connect clients with server.
- These calls appear like ordinary procedure calls to the programmer, but all the remote procedure calls from a client are enclosed in a single transaction at the server end.

Thus, if the transaction aborts, the server can undo the effect of the individual remote procedure calls

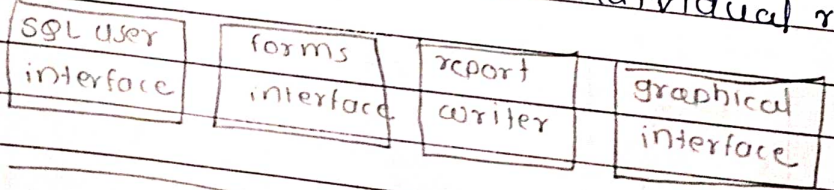


Fig → front end & back end

Server system Architectures

Server systems can be broadly categorized as transaction servers & data servers.

i) Transaction server system also called query server system provide an interface to which clients can send requests to perform an action in response to which key execute the action & send back results to the client.

- Usually client machine ship transactions to the server systems where those transactions are executed & results are shipped back to clients that are in charge of displaying the data. Requests may be specified by using SQL or through a specialized application program interface.

ii) Data server systems allow client to interact with the servers by making requests to read or update data, in units such as files or pages.
e.g, file servers provide a file system interface where clients can create, update, read & delete files.

- Data servers for database systems offer much more functionality, they support units of data such as pages, tuples or objects that are smaller than a file.

- They provide indexing facilities for data, & provide transaction facilities so that the data are never left in an inconsistent state if a client machine or process fails.

i) Transaction server process structure

A typical transaction server system today consist of multiple processes accessing data in shared memory.

The processes that form part of the database system include.

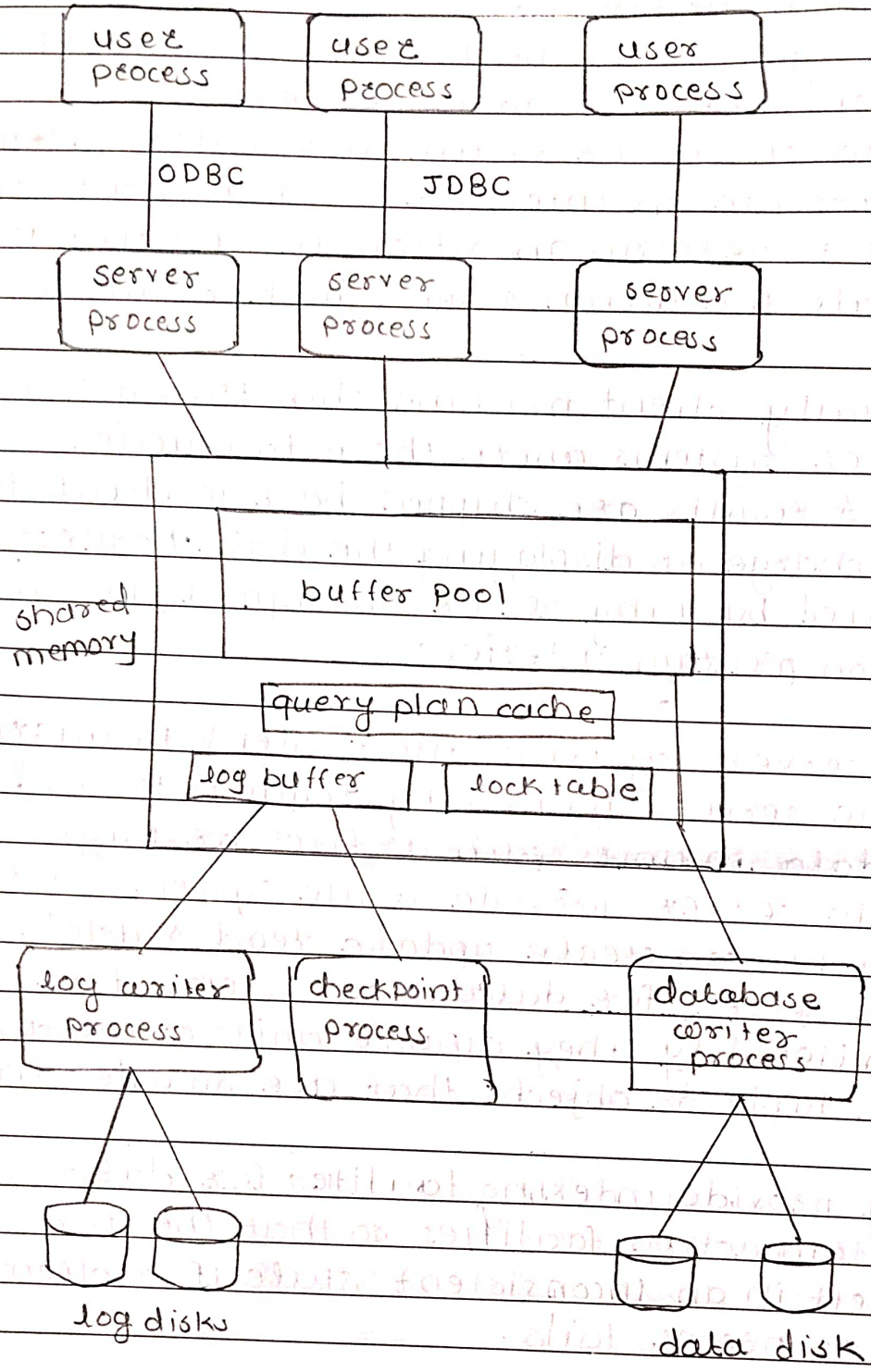


Figure → shared memory & process structure

s The processes that form part of the database system include.

i) server processes

These are processes that receive user queries (transactions), execute them & send the results back.

- The queries may be submitted to the server processes from a user interface or from a user process running embedded SQL or via JDBC, ODBC or similar protocols.

- Some database systems use a separate process for each user session & few use a single database process for all user sessions but with multiple thread so that multiple queries can execute concurrently. (A thread is like a process, but multiple thread execute as part of the same process, run in the same virtual memory space. Multiple threads within process can execute concurrently.)

- many database systems use a hybrid architecture with multiple processeces, each one running multiple threads.

ii) lock manager process

This process implements lock manager functionality which includes lock grant, lock release & deadlock detection.

iii) Database writer process

There are one or more processes that output modified buffer blocks back to disk on a continuous basis.

iv) log writer process

This process outputs log records from the log record buffer to stable storage. server processes simply add log records to the log record buffer in shared memory & if a log force is required, they request the log writer process to output log records

v) checkpoint process
This process performs periodic checkpoints.

vi) Process monitor process
This process monitors other processes & if any of them fails, it takes recovery actions for the process such as aborting any transaction being executed by the failed process & then restarting the process.

The shared memory contains all shared data such as
- Buffer pool
- Lock table
- Log buffer, containing log records waiting to be output to the log on stable storage.
- Cached query plans, which can be reused if the same query is submitted again all database processes can access the data in shared memory. Since multiple process may read or perform updates on data structures in shared memory, they must be a mechanism to ensure that only one of them is modifying any data structure at a time & no process is reading a data structure while it is being written by others.

Data server

- Data server system are used in local area networ where there is a high speed connection between the clients & the server, the client machines are comparabl in processing power to the server machines are & the tasks to be executed are computation intensive.
- In such an env., it makes sense to ship data to client machines, to perform all processing at the clien machine & then to ship the data back to the server machine. & the tasks to be executed are computatio intensive. Note that this architecture requires full back end functionality at the clients. Data server architectures have been particularly popular in object oriented database system.
- Interesting issues arise in such an architecture since the time cost of communication between client & server is high compared to that of local memory reference.

• Page shipping versus item shipping.

- The unit of communication for data can be of coarse granularity such as a page or fine granulem such as a tuple.
- We use the term item to refer to both tuples & objec If the unit of communication is a single item, the overhead of message passing is high compared to the amount of data transmitted. Instead when an item is requested, it makes sense also to send back other items that are likely to be used in the near future.
- fetching items even before they are requested is called prefetching. Page shipping can be considered a form of prefetching if multiple items reside on a page, since all the items in the page are shipped when a process desires to access a single item in the page.

Locking

- Locks are usually granted by the server for the data that it ships to the client machines.
- Disadvantage of page shipping is that client machines may granted locks of too coarse a granularity a lock on page implicitly locks all items contained in the page.
- Even if the client is not accessing some items in the page it has implicitly acquired locks on all prefetched items.
- other client machines that require locks on those items may be blocked unnecessarily.
- Techniques for lock descalation, have been proposed where the server can requests it's clients to transfer back locks on prefetched items. If the client machine does not need the prefetched items, it can transfer locks on the item back to the server & locks can then be allocated to other clients

Data caching

- Data that are shipped to a client on behalf of a transaction can be cached at the client, even after the transaction completes, if sufficient storage space is available.
- successive transaction completes, if sufficient storage space is available at the same client may be able to make use of the cached data.
- However, cache coherency is an issue: Even if a transaction finds cached data, it must make sure that those data are up to date, since they may have been updated by a different client after they were cached.
- Thus a message must still be exchanged with the server to check validity of the data & to acquire a lock on the data.

Look caching

If the use of data is mostly partitioned among the clients with clients rarely requesting data that are also requested by other clients, locks can also be cached at the client machine.

Parallel System

* Parallel database systems consist of multiple processing & multiple disks connected by a fast interconnection network.
- A coarse grain powerful ||el machine consists of a small no. of powerful processors
- A massively parallel or fine grain parallel machine utilizes thousands of smaller processors.
- In ||el processing many operations are performed simultaneously
- ||el computers with hundreds of processors & disks are available commercially.
- There are 2 main measures of performance of DB syst
  1) Throughput → No. of tasks that can be completed in given time interval.
  2) Response time → The amount of time it takes to complete single task from the time it is submitted.

Speedup Scaleup

- A fixed sized problem executing on small system is given to a system which is N-times larger. measured by:

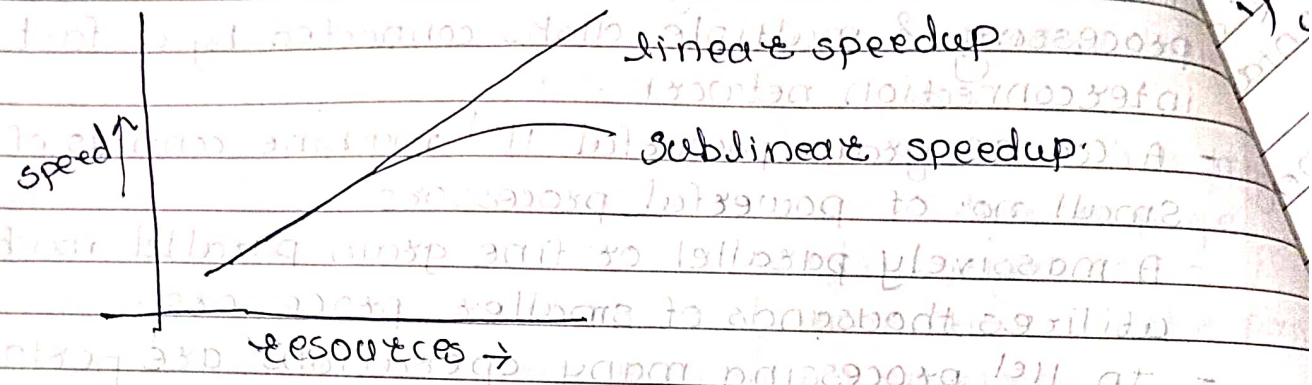
$$\text{Speedup} = \frac{\text{small system elapsed time}}{\text{large system elapsed time}}$$

- Speedup is linear if equation equals N.

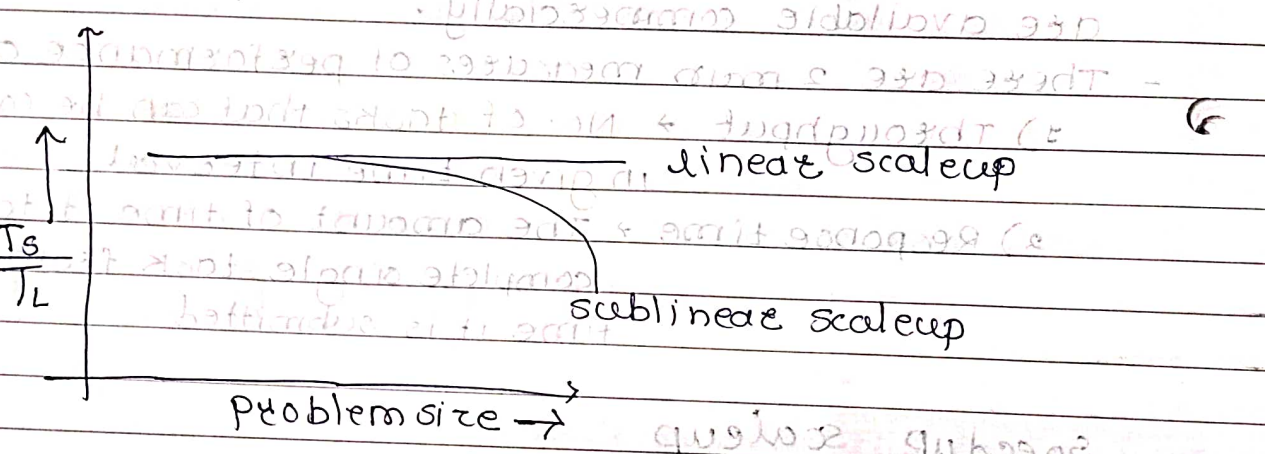
*
- Increase the size of both the problem & the system
- It is N times larger system used to perform N-times larger job.
- measured by

$$\text{Scale up} = \frac{\text{small system small problem elapsed time}}{\text{big system big problem elapsed time}}$$

Scaleup is linear if equation equals 1



(1) fig: speedup



Batch scaleup.
- A single large job, typical of most decision support queries & scientific simulation
- Use an N times larger computer on N-times larger problem.

Transaction scaleup
- Numerous small queries submitted by independent user to shared database, typical transaction processing & time sharing systems.
- N times as many users submitting requests to an N-times larger db on an N times larger computer.
- Suited to rel DB.

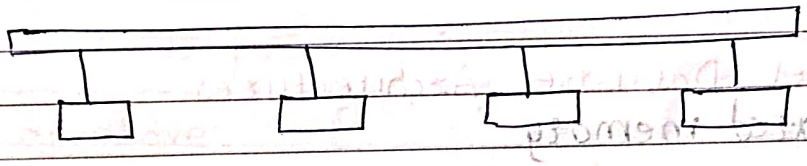
Factors limiting speedup & scaleup

i) startup costs
cost of starting up multiple processes may dominate computation time, if the degree of parallelism is high.

ii) Interference:
Processes accessing shared resources compete with each other, thus spending time waiting on other processes, rather than performing useful work.

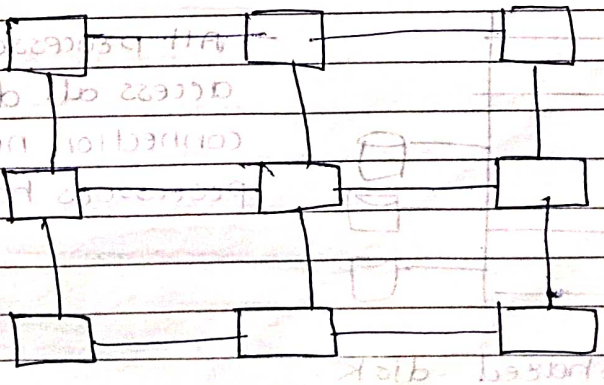
Interconnection Network Architecture

1) Bus
   i) System components, send data on & receive data from a single communication bus.
   ii) It does not scale well with increasing parallelism



a) bus.

2) Mesh
   i) components are arranged as nodes in a grid & each components is connected to all adjacent component
   ii) communication link grow with growing no. of components & so scales better
   iii) But may require √n hops to send message to a node

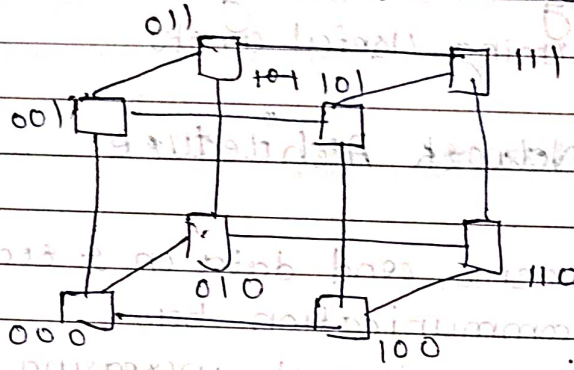


b) mesh

Hypercube

i) components are numbered in binary, components connected to one another if their binary representa differ in exactly one bit.

ii) n components are connected to log(n) other compone & can reach each other via at most log(n) links, reduces communication delays.

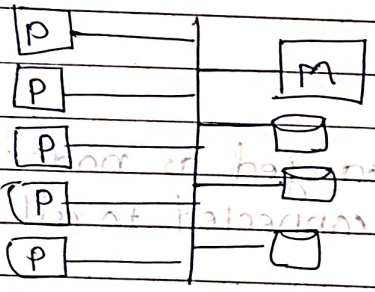


c) hypercube

Parallel Database Architectures

1) Shared memory

- Processors share a common memory

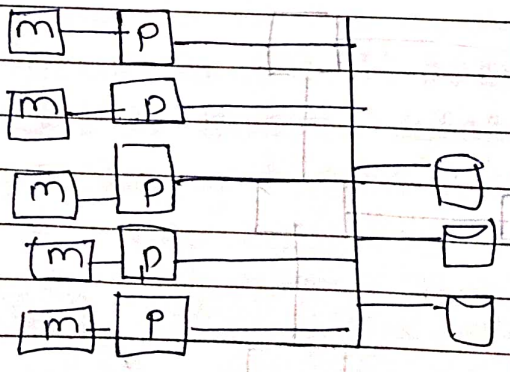


a) shared memory.

- Process & disks have access to a common memory typically via a bus or through an interconnection network

b) shared disk

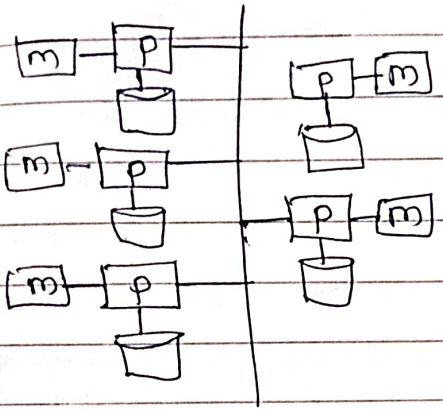
Processors share a common disk.



b) shared disk

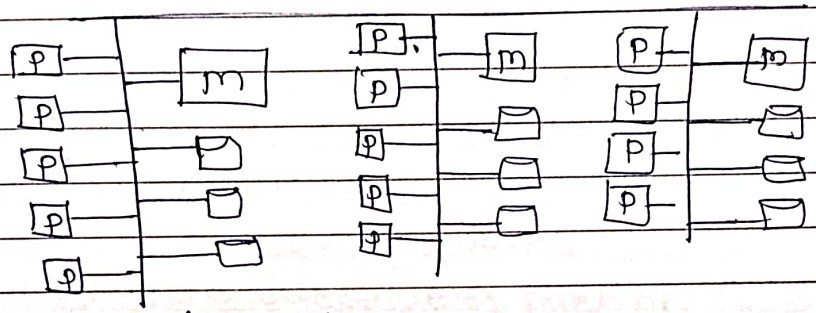
- All processors can directly access all disks via an interconnection network, but the processors have private memory

3) shared nothing



- Processors share neither a common memory nor common di

4) Hierarchical



- Hybrid of above
- It combines characteristics of shared memory, shared disk & shared nothing arch.

Distributed DB system

i) It is logically related collection of data that is shared which is physically distributed over computer network on different sites.

ii) It consists of loosely coupled sites that share no physical components.

iii) DB systems that run on each site are independent of each other

iv) Transactions may access data at one/more sites

Homogeneous Distributed DB

In an homogeneous distributed dB

i) All sites have identical software.

ii) Are aware of each other & agree to cooperate in processing user requests.

iii) Each site surrenders part of its autonomy in terms of right to change schemas or s/w

iv) Appears to user as single system.

Heterogeneous distributed DB

i) Different sites may use different schemas & s/w

ii) Difference in schema is major problem for query processing

iii) Difference in s/w is a major problem for transaction processing.

iv) Sites may not be aware of each other & may provide only limited facilities for cooperation in transaction processing

* Distributed data storage

1) Replication

i) System maintains multiple copies of data * stored in different sites for faster retrieval & fa tolerance.

ii) A relation or fragment of relation is replicated it is stored redundantly in two or more

iii) Fully replication of r is the case where r relation is stored at all sites.

* Data fragmentation

Division of relation r into fragments $r_1, r_2 \cdots r_n$ which contain sufficient info. to reconstruct r.

i) Horizontal fragmentation

Each tuple of r is assigned to one / more fragm

E.g

branch-name	accnt-no.	bal
Hillside	A-305	500
Hillside	A-226	800
Hillside	A-155	62

$account = \sigma_{branch\text{-}name = \text{"Hillside"}}(account)$

branch-name	accnt-no.	bal
valleyview	A-177	205
—"—	A-402	10000
—"—	A-408	1123
—"—	A-639	750

$account_2 = \sigma_{branch\text{-}name = \text{"valleyview"}}(account)$

ii) vertical fragmentation

The schema for relation r is split into several smaller schemas.

a) All schemas must contain a common candidate key or super key to ensure lossless join property

b) A special attribute, the tuple-id attribute may be added to each schema to serve as a candidate key.

accnt-no	bal	tuple-id
A-305	500	1
A-226	836	2
A-177	205	3
A-402	10000	4
A-155	62	5
A-304	1123	6
A-302	750	7

$deposit_2 = \Pi_{account\_no,\ balance,\ tuple\_id}(employee)$

Advantages of Fragmentation

1) Horizontal

i) It allows parallel processing on fragments of relation

ii) It allows a relation to be split so that tuples are located where they are most frequently accessed.

2) vertical

i) Allows tuples to be split so that each part of the tuple is stored where it is most frequently accessed.

ii) tuple-id attribute allows efficient joining of vertical fragments.
iii) Allow parallel processing on a relation.
iv).
3) vertical & horizontal fragmentation can be mixed.
- Fragments may be successively fragmented to an arbitrary depth.

* Data Transparency

i) It is a degree to which system user may remain unaware of the details of how & where the data items are stored in a distributed system.

Transparency issues in relation to
i) Fragmentation transparency
ii) Replication transparency
iii) location transparency.

* Naming of data Items - criteria
1) Every data item must have a system wide unique name.
2) It should be possible to find the location of data items efficiently.
3) It should be possible to change the location of data items transparently.
4) Each site should be able to create new data items autonomously.

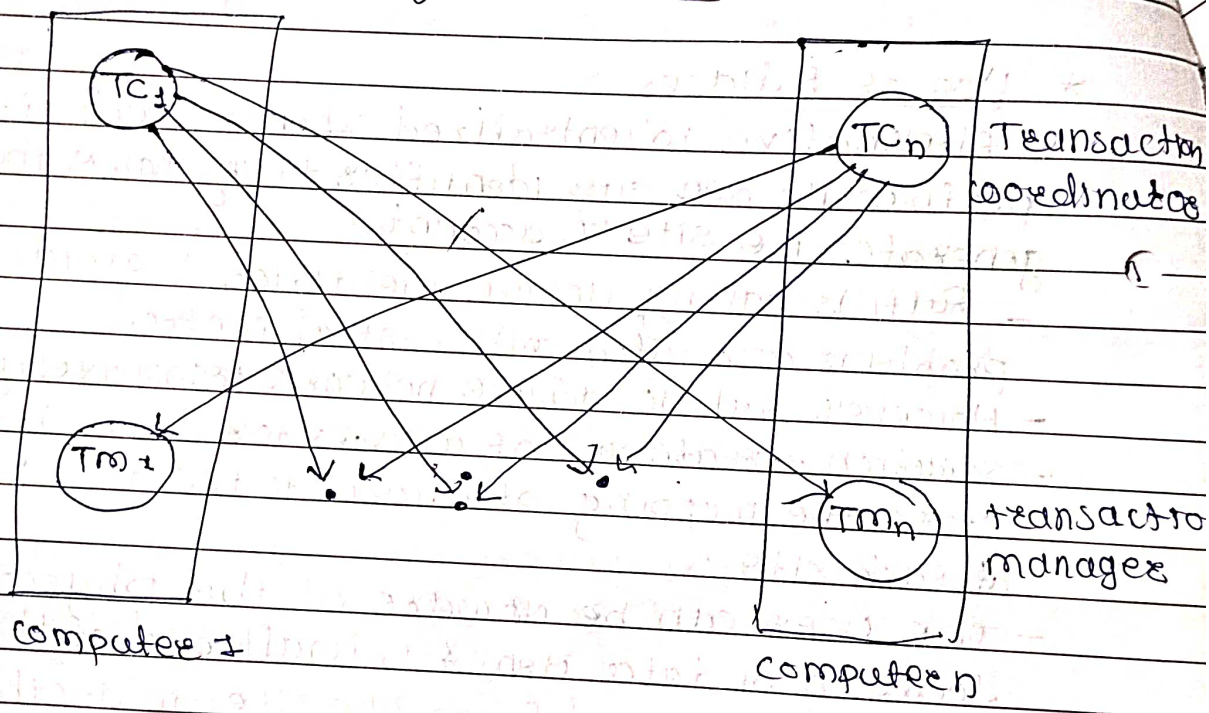
* centralized scheme - Name server
1) Structure
i) name server assigns all names.
ii) Each site maintain record of local data items.
iii) Sites ask name server to locate non local data items.

* Use of Aliases
- Alternative to centralized scheme → each site prefixes its own site identifier to any name that it generates i.e site IT. account.
- Fulfills having unique identifiers & avoids problems associated with central control.
- However, fails to achieve network transparency.
- Solution - create set of aliases for data items, store the mapping of aliases to the real names at each site.
- The user can be unaware of the physical location of data item & is unaffected if the data item is moved from one site to another

Distributed Transaction
- Transaction may access data at several sites.
- Each site has a local transaction manager responsible for i) maintaining a log for recovery purposes.
ii) Participating in coordinating the concurrent execution of the transactions executing at that site.
- Each site has transaction coordinator, which is responsible for
i) starting the execution of transactions that originate at the site
ii) Distributing subtransactions at appropriate sites for execution

iii) Coordinating the termination of each ~~appropriate~~ transaction that originates at the site, which may result in the transaction being committed at all sites or aborted at all sites.

Transaction system Architecture



* System failures modes
  - Failures unique to distributed systems
    a) Failure of a site
    b) Loss of messages
       - Handled by n/w transmission control protocols such as TCP-IP
  - Failure of communication link.
    - Handled by n/w protocols by routing msg. via alternative links.

commit protocol

- commit protocols are used to ensure atomicity across sites
  i) A transaction which executes at multiple sites must either be committed at all the sites or aborted at all the sites.
  ii) Not acceptable to have a transaction committed at one site & aborted at another.
- The two phase commit (2PC) protocol is widely used
- The three phase commit (3PC) protocol is more complicated & more expensive but avoids some drawbacks of two phase commit protocol. This protocol is not used in practice.

i) Two phase commit Protocol (2PC)

i) Assumes fail stop model - failed sites simply stop working & do not cause any other harm such as sending incorrect msg. to other sites.
ii) Execution of the protocol is initiated by the co-ordinator after the last step of the transaction has been reached.
iii) The protocol involves all the local sites at which the transaction executed.
iv) Let T be the transaction initiated at site S_i & let the transaction coordinator at S_i be C_i

phase 1 - obtaining a Decision

i) coordinator asks to all participants to prepare to commit transaction T.
   - C_i adds the records <prepare T> to the log & forces log to stable storage
   - sends prepare T msg. to all sites at which T executed.

ii) Upon receiving msg, transaction manager at determines if it can commit the transaction
- if not, add record <no T> to the log & send abort T msg. to Ci
- if the transaction can be committed, then
- add the records <ready T> to th log
- force all records for T to stable storage
- send ready T msg. to Ci

Phase 2 → Recording the Decision
- T can be committed of Ci received a ready T message from all the participating sites otherwise T must be aborted.
- coordinator adds a decision record <commit T> or <abort T>, to the log & forces record onto stable storage. once the record stable storage it is irrevocable (even if failure occur)
- coordinator sends a msg. to each participant informing it of the decision (commit / abort)
- Participants take appropriate action locally

* Handling of failures - site failure

- when site Si recovers, it examines it's log to determine the fate of transactions active at the time of the failure.
i) Log contain <commit T> record : txn had completed, nothing to be done.
ii) Log contains <abort T) record : txn had completed, nothing to be done.
iii) Log contains <ready T> record : site must consult Ci to determine the fate of T
- If T committed, redo (T), write <commit T> record
- If T aborted, undo (T)

iv) The log contains no log records concerning T:
- Implies that S_k failed before responding to the prepare T message from C_i.
- Since the failure of S_k precludes the sending of such response coordinator C_i must abort T.
- S_k must execute undo(T)

Recovery & concurrency control
- In doubt transactions have a <ready T> but neither a <commit> nor an <abort T> log record.
- The recovering site must determine the commit abort status of such transactions by contacting other sites this can slow & potentially block recovery.
- Recovery algo. can note lock info. in the log.
- Instead of <ready T> write out <ready T, L>
L = list of locks held

3PC
- i) No n/w partitioning
ii) At any point at least one site must be up.
iii) At most K sites can fail

Phase 1 - Obtaining preliminary Decision: Identical to 2PC phase 1
- every site is ready to commit if instructed to do so.

Phase 2 of 2PC is split into 2 phases, Phase 2 & phase 3 of 3PC
- In phase 2 coordinator makes a decision as in 2PC & records it in multiple sites.
In phase 3 coordinator sends commit msg. to all participants.

concurrency control

i) Modify concurrency control schemes for use in distributed env.
ii) we assume that each site participates in the execution of commit protocol to ensure global transaction automicity.
iii) we assume all replicas of any item are upd

1) Single lock manager Approach

- System maintains single lock manager that resides in single chosen site say Si
- when transaction needs to lock data item it sends a lock request to Si & lock manager determines whether the lock can be granted immediately,
  - i) if Yes lock manager sends a msg to the site which initiated the request
  - ii) If No request is delayed until it can be granted, at which time a msg is sent to the initiating site.

2) Di

2) Distributed lock manager

- i) In this approach functionality of locking is implemented by lock managers at each site
  ii) lock managers control access to local data items but special protocols may be used for replicas.
  iii) work is distributed & can be made robust to failures.
  iv) several variants of this approach
    a) primary copy
    b) majority protocol
    c) Biased protocol
    d) quorum consensus

Distributed Query Processing

i) For centralized systems, the primary criteria for measuring the cost of a particular strategy is the no. of disk accesses.

ii) In a distributed system, other issues must be taken into account

iii) The cost of data transmission over the network.

iv) The potential gain in performance from having several sites process parts of the query in parallel.

- consider algebraic queries on fragments.
- it must be possible to construct relation r from its fragments
- Replace relation r by the expression to construct relation r from its fragments

- consider horizontal fragmentation of account relation into.

account 1 = $\sigma_{branch\_name = \text{"Baroda"}}(account)$
account 2 = $\sigma_{branch\_name = \text{"Kodoli"}}(account)$

The query $\sigma_{branch\_name = \text{"Baroda"}}(account)$
becomes $\sigma_{branch\_name = \text{"Baroda"}}(account_1 \cup account_2)$
which is optimized into
$\sigma_{branch\_name = \text{"Baroda"}}(account_1) \cup \sigma_{branch\_name = \text{"Baroda"}}(account_2)$

Database Link.

i) Use the CREATE DATABASE LINK statement to create a database link. is a schema object in one DB that enables you to access objects on another DB.

ii) The other DB need not be an oracle DB system. However to access non-oracle systems you must use oracle Heterogeneous service.

iii) After you have created a DB link you can use it to refer to tables & views on the other DB.

iv) In SQL statements you can refer to a table or view on the other DB by appending @dblink to the table or view name. You can query a table or view on the other DB with SELECT statement.

v) You can also access remote tables & views using any INSERT, UPDATE, DELETE or LOCK TABLE statement.

vi) To create a private DB link, you must have the CREATE DATABASE LINK system privilege. To create public DB, you must have the CREATE TABLE DATABASE LINK system privilege. Also you must have the CREATE SESSION system privilege. ~~Also~~ on the remote oracle DB.

syntax

create - database - link

→ create → shareed / public → Database → Link - dblink →

Updatable view

i) It is a special case of deletable view.

ii) A deletable view becomes an updatable view when at least one of its columns is updatable.

iii) A column of view is updatable when all of the following rules are true.

- The view is deletable
- The column resolves to column of a table & READ ONLY option is not specified.
- All the corresponding columns of the operand of a UNION ALL have exactly matching data types & matching default values if the full select of the view include a UNION ALL

iv) The following examples uses constant values that can't be updated. However the view is a deletable view & at least one of its columns is updatable. There it is an updatable view.

```
CREATE VIEW updatable-view
    (no., current_date, current
AS
SELECT NO., (current Date, (
from weather. forecast
where no = 300 ;)
```

* when can a view be updated ?

i) The view is defined based on one & only one table
ii) view must include primary of table based upon which view has been created.
iii) View should not have any field made out of agg. fun.
iv) The view must not have any DISTINCT clause in it's def.
v) Not groupby, having by clause, view not have subquer
vi) Any of the selected output fields must not use constants, strings or value expressions.

Syntax

Update <view-name> SET <column1 ...> where <cond>

2. Advanced SQL

Relational Set Operators

SQL provides relational set operators to combine the output of two queries to generate a new relation.

1) Union
2) Union All
3) Intersect
4) Minus.

1) Union

i) The union statements combines rows from two or more queries without including duplicate rows.

ii) The syntax of UNION statement is

query UNION query.

iii) UNION statement combines output of two SELECT queries. SELECT statement must be union-compatible.

iv) That is, they must return same no. of attributes & similar data types.

Example 1)

customer 1

Cus-LName	F-Name	Phone
More	Swati	99294
Patil	Seema	22244
Shinde	Ashu	22245
Yadav	Trupti	22246

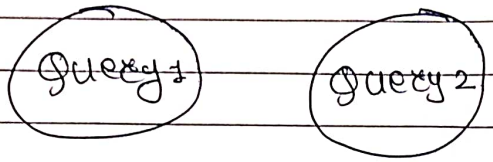
customer 2

Cus-LName	FName	Phone	Hometown
More	Swati	99294	Kodoli
Patil	Seema	22246	Warana
Shinde	Ashu	22245	Talsande
Yadav	Trupti	22246	Pargaon
Powar	Aarav	22247	Kodoli
Kumbhar	Shreya	22248	Satara

```
SELECT CUS_LNAME, CUS-FName, CUS-PHONE
From customer1
UNION
select, cus-LName, Cus-FName, cus-Phone
from customer 2;
```

2) Union All

Similar to Union just that union all returns also the duplicate values



Syntax

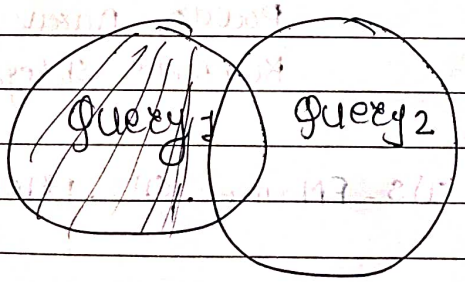
```
SELECT * from table 1
 UNION ALL
SELECT * from table 2;
```

when using UNION & UNION ALL column in SELECT statements need to match ~~word~~. This would return an error.

```
SELECT column 1 from table 1
UNION ALL
SELECT * from table 2;
```

3) MINUS

- Minus returns the difference between the first & second SELECT statements.
- It is one where we need to be careful which statement will put 1st cause we will get only those results that are in the 1st SELECT statement & not in the second



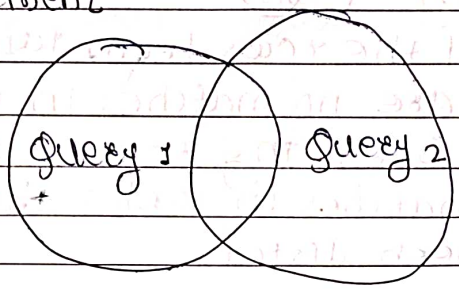
Syntax

```
SELECT * from table 1
MINUS
SELECT * from table 2;
```

(4)

INTERSECT

i) It is opposite from MINUS as it returns us the results that are both to be found in 1st & 2nd select statement -

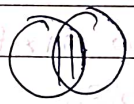


Syntax

```
SELECT * From table.1
INTERSECT
Select * from table 2
```

* SQL Join Queries

1) Inner Join



- It returns all rows from both tables where there is match - If there are rows in employees that don't matches in orders those rows will not be listed

Syntax

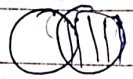
```
Select field1, field2 from 1st table
Inner join second table
ON first table.keyfield = 2nd table. foreign keyfield
```

2) Left outer join

- It returns all the rows from 1st table, even if there are no matches in the 2nd table (orders)
- If there are rows in employees that do not have matches in orders, those rows also will be listed.

Syntax

```
select field1, field2 from table1
Left join table2
ON table1.keyfield = table2.foreign.key field
```

5) Right outer join 

- It returns all the rows from table 2 (orders) even if there are no matches in the 1st table
- If there had been any rows in orders that didn't have matches in tables those rows also would have been listed.

Syntax

```
select field1, field2 from table1
Right join table 2
ON table1.keyfield = table2.foreign.keyfield
```

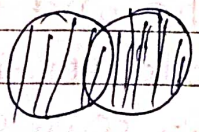
4) cross join

- It is also known as cartesian join, which return the cartesian product of two or more joined tables.
- It produces table that merges each row from the 1st table with each 2nd row.
- It is not required to include any condition in cross join

Syntax

```
select * from table1
cross join
table2;
```

6) Full outer Join



SQL Functions

i) SQL functions are very useful tools.
ii) when you want to list all employees ordered by year of birth or when your marketing dept wants you to generate list of all cust. ordered by zip code & 1st 8 digits of their telephone no., in both cases you will need SQL functions that can be derived from an existing attributes.
iii) Functions always use numerical, date or string value
iv) The value may be part of the command itself or it may be an attribute located in table.
v) Therefore fun. may appear anywhere in an SQL statement where value or an attribute can be used.
vi) There are many types of SQL-fun. such as arithmetic trignometric, string, date & time fun.

1) Date & time function
i) All date fun. take one parameter & return value,

a) YEAR → It returns four digit year
Syntax
YEAR (date-value)
e.g → list all employees born in 1996

b) MONTH → It Returns two digit month code
Syntax → MONTH (date-value)

c) DAY → It returns no. of day
Syntax → DAY (date-value)

d) DATE() or GETDATE → It returns todays date

Numeric function

i) Numeric functions can be grouped in many diffe ways, such as algebraic, trignometric & logarithmi

ii) **ABS** → It returns absolute value of no.
syntax → ABS (numeric_value)

iii) **ROUND** → It a value to specified precision no. of digits.
Syntax → ROUND (numeric_value, p)
P = precision

iv) **CEIL/CEILING/FLOOR**
It returns the smallest integer greater than or equal to no. to returns the largest integ equal to or less than no. respectively,
syntax → CEIL (numeric_value) — oracle
CEILING (—"—) — SQL server
FLOOR (—"—)

String function

i) **Concatenation**
|| → oracle
+ - ms Access / SQL server
→ concatenation data from two different character columns & return a single column
→ Syntax.
streg_value || streg_value
streg_value + streg_value

e.g - List all employees names.
SELECT EMP_LNAME || ', ' || EMP_FNAME AS NAME
FROM EMPLOYEE

ii) **UPPER/LOWER**
It returns a string in all capital or all lowercase letters
syntax → UPPER (streg_value)
LOWER (streg_value)

e.g - List all employee name in all capital letters

```
SELECT UPPER(EMP_LNAME) || ', ' || UPPER
(EMP_FNAME) AS NAME
FROM EMPLOYEE;
```

iii) SUBSTRING

It returns substring or part of given string parameters.

syntax → SUBTRING (string-value, P, l)

SUBSTR (|| —— || ——);

P = start position

l = length of character

e.g → List all 1st three characters of all employee phone no.

```
SELECT EMP_PHONE, SUBSTR(EMP_PHONE, 1, 3)
AS PREFIX
FROM EMPLOYEE
```

iv) LENGTH

It returns no. of characters in string value

syntax → LENGTH (string-value)

e.g → Lists all employee last names & length of their names; ordered descended by last name length.

```
SELECT EMP_LNAME, LENGTH(EMP_LNAME) AS
NAMESIZE
FROM EMPLOYEE;
```

conversion function

i) It allow you to take value of given datatype & convert it to the equivalent value in another data type.

ii) TO_CHAR & TO_DATE, TO_CHAR function takes date value & returns a character string representing a day, month or year.

iii) TO_DATE fun. takes character string representing data & returns an actual date.

Numeric to character

TO_CHAR

i) It returns character string from numeric value

Syntax → TO_CHAR (numeric_value, fmt) → Oracle
CAST (numeric AS varchar(length)) } SQL
CONVERT (varchar(length), numeric) } Server

Date to character

TO_CHAR - Oracle
CAST - SQL Server
CONVERT - SQL Server

i) It returns character string or a formatted character string from date value

ii) Syntax

Oracle → TO_CHAR (date_value, fmt)
SQL Server → CAST (date AS varchar(length))
CONVERT (varchar(length), date)

String to Number

TO_NUMBER

i) It returns formatted no: from character string using given format

ii) Syntax

Oracle - TO_NUMBER(char_value, fmt)

fmt = format used; can be
9 = display digit
0 = " " leading zero
, = " " the comma
. = " " decimal point
$ = " " dollar sign
B = leading blank
S = leading sign
MI = trailing minus sign

PL/SQL
- Procedural language using SQL
- By using SQL we can ;)
- PL/SQL is block structured language
- SQL limitation →
  does not support checking, looping, branching so that we use PL/SQL
- All statements of block are passed to oracle engine all at once which increases processing speed & decreases traffic
- PL/SQL supports functions, looping, branching & it able to handle exceptions.
-

PL/SQL blocks.

- Declare
- Begin
- Exception
- End

PL/SQL have some parts.
i) stored procedure
ii) Triggers
iii) cursors

i) Stored Procedure

```
CREATE [OR REPLACE] PROCEDURE proc_name
[list of parameters]
[IS / AS]
[variable_name data type [:= initial_value]]

BEGIN
    PL/SQL or SQL statement;
END;
```

i) In above syntax argument specifies the param that are passed to the stored procedure, A sto procedure could have been zero or more argu or parameters.

ii) IN/OUT indicates whether the parameter is for input, output or both.

iii) Data type is one of the procedural SQL data types used in RDBMS., the data types normall match those used in the RDBMS table creation statement.

iv) variables can be declared bet. keywords IS & BEGIN, you must specify variable name, its data type & initial value.

Example

```
CREATE OR REPLACE PROCEDURE PRC_PROD_DISCOUNT
AS BEGIN
    UPDATE PRODUCT
    SET P_DISCOUNT = P_DISCOUNT + .05
    WHERE P_QOH >= P_MIN * 2;
DBMS_OUTPUT.PUT_LINE ('** Update finished **');
END;
```

Here procedure is created.

- In above example PRC_PROD_DISCOUNT stored procedure uses the DBMS_OUTPUT.PUT_LINE fun. to display a message when procedure executes.
- To execute stored procedure, you must use following syntax:

```
EXEC procedure_name [(parameter_list)];
```

* PL/SQL processing with CURSORS

i) In SQL statement you have used inside a PL/SQL block have returned a single value.
ii) If the SQL statement returns more than one value you will generate an error.
iii) By using cursor, you use an SQL statement that returns more than one value inside PL/SQL code.
iv) Cursor is a special construct used in procedural SQL to hold the data rows returned by SQL query
v) Cursor as a reserved area of memory in which the output of the query is stored
vi) Cursors are held in reserved memory area in the DBMS server, not in client computer

There are two types of cursors.
i) Implicit
ii) Explicit

i) Implicit cursor

Implicit cursor is automatically created in procedural SQL when the SQL statement return only one value.

ii) Explicit cursor

- Explicit cursor is created to hold the output of an SQL statement that may return two or more rows. It returns 0 or only one row

Syntax

```
CURSOR cursor_name IS select_query;
```

* Declaring cursor

```
CURSOR C_customers IS
select id, name, address FROM customer;
```

opening cursor

opening cursor allocates memory for cursor & makes it ready for fetching rows returned by SQL statement into it

e.g `OPEN C_customers.`

*Fetching the cursor

It involves accessing one row at a time for example, we will fetch rows from above opened cursor as follows.

```
FETCH c_customers INTO c_id, c_addr;
```

* closing the cursor

closing cursor means releasing allocated memory

```
CLOSE c_customers;
```

~~Example~~ :

~~c_id customers~~ ..

cursors command

i) Open :- Opening the cursor executes the SQL command & populates the cursor with data, opening the cursor for processing

- The cursor declaration command only reserves a named memory area for the cursor, it doesn't populate the cursor with data before you can use cursor, you need to open it

e.g → OPEN cursor-name

ii) FETCH

- once the cursor is opened, you can use FETCH command to retrieve data from the cursor & copy it to the PL/SQL variable for processing.

syntax

```
FETCH cursor-name INTO var1 [, var2, --]
```

- The PL/SQL variables used to hold the data must be declared in the DECLARE section & must have data types compatible with the columns retrieved by the SQL command. If the cursor SQL statement return 5 columns, there must be 5 PL/SQL var. to receive the data from cursor

iii) CLOSE

The CLOSE command closes the cursor for processing

Cursor Attributes

1) %ROWCOUNT

It returns the no. of rows fetched so far
If the cursor is not OPEN, it returns an error, if no FETCH has been done but cursor is OPEN it returns 0

2) %FOUND

- It returns TRUE if the last FETCH returned a row & FALSE if not.
- If the cursor is not OPEN, it returns an error.
- If no FETCH has been done, it contains NULL.

3) %ISOPEN

- It returns TRUE if the cursor is open (ready for processing) or FALSE if the cursor is closed.

* PL/SQL Stored functions →

i) Stored procedures & functions are very similar.
ii) Stored function is basically named group of procedural & SQL statement that returns a value i.e (RETURN)

iii) Syntax:

```
CREATE FUNCTION fun_name (argument IN datatype,
RETURN data_type [IS]
BEGIN
        PL/SQL Statement
        ----
        RETURN (value or expression);
END;
```

iv) Stored fun. can be invoked only from within stored procedures / triggers & can't be invoked from SQL statements.

PL/SQL Stored Functions

i) A stored function is basically named group of procedural & SQL statements that returns a value i.e indicated by a RETURN statement in it's program code.

ii) Syntax

```
CREATE FUNCTION fun.-name (argument IN data_type,--)
RETURN data.types [IS]
BEGIN
        PL/SQL statements;
        :
        RETURN (value/expression);
END;
```

iii) Stored functions can be invoked only from within stored procedures or triggers & cannot be invoked from SQL statements.

Example

```
DECLARE
   a number;
   b number;
   c number;
FUNCTION findmax (x IN number, y IN no.
RETURN number
IS
   z number;
BEGIN
   IF x > y THEN
      z := x;
   ELSE
      z := y;
   END IF;
   RETURN z;
END;
BEGINE
   a = 23;
   b = 45;
   c = findmax (a, b);
   dbms_output.Put.line ('maximum of (23,45): ' ) ) c );
END;
```

Embedded SQL

i) Embedded SQL is a term used to refer to SQL statements that are contained within an application programming lang. such as visual Basic, .Net, C#, or java.

ii) static/embedded sql are SQL statements in an application that do not change at runtime & therefore can be hard coded into the appln

* Dynamic SQL

- It is a programming technique that enables you to build SQL statements dynamically at runtime.
- you can create more general purpose, flexible appln by using dynamic SQL because the full text of SQL statement may be unknown. at compilation.

	Embedded SQL	Dynamic SQL
i)	DB access procedure in pre-determined in the statement	DB will be accessed can be determine only at run time
ii)	Statements are compiled at compile time	compiled at run time
iii)	Stat: Execute, prepare, immediate not used	Used

when cond^n is true then trigger is automatically occured

when event is occured & condition is true, Execute the action

Trigger –

i) It is a stored procedure in DB which automatically invokes whenever a special event e in the DB occurs

ii) e.g → A trigger can be invoked when a row is inserted into a specified table or when certain table columns are being updated.

iii) **Syntax**

```
create trigger [trigger-name]
[before | after]
{insert | update | delete} – types of events
on [table-name]
[for each row]
[trigger-body]
```

i) create trigger [trigger-name] → creates / replace an existing trigger with the trigger-name

ii) [Before | after] :

This specifies when the trigger will be executed

iii) {insert / update / delete} : This specifies the DML opern.

iv) on [table-name] :

This specifies the name of the table associated with the trigger.

v) [for each row] :

This specifies row level trigger i.e trigger will be executed for each row being affected

vi) [trigger-body] :

This provides the opern to be performed as trigger is fired

Example → create table empsal

before|after insert|update on employee

Trigger

i) It cannot execute manually it is big difference between stored procedure & trigger

ii) They only execute in response to user action like INSERT

iii) It is used to check the validity of data that are inserted into table

iv) If you need a certain piece of code to always be executed in response to an event, the best option is to use triggers.

Example →

table (1)
```
create table Employee
( id int primary key, name varchar(45),
Sal int, gender varchar(12), deptId int)
Insert into employee values (1, 'ABC', 10000, Mal
```

table (2)
```
create table empl-AuditTest
( id int identity, Audit_Action text)
```

Now, we create trigger that stores trans. record of each insert operation on emp. table into the Employee_Audit-Test table.

```
- create trigger trigInsertEmployee
  on Employee
  for Insert
  As
  BEGIN
    Declare @Id int
    select @Id = Id from inserted
    Insert into Emp-AuditTest
values('New empl with Id = ' + CAST(@Id As varchar(10)) +
' is added at' + CAST(Getdate() As varchar(22)))
End
```

if you could create sequence to automatically assign values to the customer code each time new customer is added & create another sequence

To check sequence :-

→ SELECT * FROM USER_SEQUENCES;

After creating trigger, we will add following

Insert into employee values (6, 'Swati', 36000, 'female', 13)

Delete

```
create Trigger Deleteemp
on Employee
For delete
As
BEGIN
Declare @ Id int
Select @ Id = Id from deleted
Insert into emp_AuditTest
values ('An existing empl with Id = ' + CAST
        (@Id As varchar(10)) + ' is deleted at ' +
        CAST(Getdate() As varchar(22)))
End
```

After creating trigger

Delete from employee where Id = 2;

Difference bet. fun stored function & stored procedures

	Function	Procedure
i)	It must return a value	It is optional, it return zero.
ii)	Functions can have only input parameters	It can have input or output parameters
iii)	Functions can be called from procedure	Procedures cannot be called from function
iv	It allows only select statement in it	It allows select as well as DML statements
v	It can be used in join clause as a result set	It can't used in join clause
vi	Used in SQL statements anywhere in WHERE/ HAVING/SELECT section.	It can't used in SQL statements in where/ having/select clause
vi	Try catch block can't used	An exception can be handled by try catch block in this
vii	We use select command to execute a function	We use EXECUTE keyword to execute stored procedure

Oracle SEQUENCE

i) Oracle does not support Autonumber data type or the identity column property, instead you can use a 'sequence' to assign values to column on a table but an oracle sequence is very different from Access AutoNumber data type & deserves close security

→ oracle sequences are an independent object in the DB sequence are not data type
→ oracle sequences are not tied to table or column
→ oracle sequence have name & can be used anywhere value is expected.
→ oracle sequences generate numeric value that can be assigned to any column in any table.
→ The table attribute to which you assigned a value based on sequence can be edited & modified
→ oracle sequence can be created & deleted anytime

Syntax

CREATE SEQUENCE name [START WITH n] [INCREMENT BY n] [CACHE | NOCACHE]

where

- name is the name of sequence
- n is the integer value that can be +ve or -ve
- START WITH specifies initial sequence value
- INCREMENT BY determines the value by which sequence is incremented
- The CACHE/NOCACHE clause indicates whether oracle will preallocate sequence no. in memory (by default 20 values)

e.g → CREATE SEQUENCE CUS_SEQ START WITH 2010 NOCACHE;

if you could create sequence to automatically assign values to the customer code each time new customer is added & create another sequence

To check sequence :-

→ SELECT * FROM USER_SEQUENCES;

Subquery

i) A subquery is best defined as query within query. Subquery enable you to write queries that select rows for criteria that are actually developed while the query is executing at run time.

ii) It is the use of SELECT statement inside one of the clauses of another SELECT statement.

iii) In fact subquery can be contained inside another subquery, this is inside another subquery & so forth.

iv) A subquery can also be nested inside INSERT, UPDATE & DELETE statement. Subqueries must be enclosed within parentheses.

v) A subquery can be used any place where an expression is allowed providing it returns a single value, this means that a subquery that returns a single value can also be listed as an object in a FROM clause listing.

vi) This is termed inline view because when subquery is used as part of FROM clause, it is treated like a virtual table or view.

vii) Subquery can be placed either in FROM clause, WHERE clause or HAVING clause of main query.

Types of subqueries

i) Single row sub query
ii) Multiple row sub query
iii) Correlated sub query

1) Single Row sub query
   i) Sub query which returns single row output.
   ii) They mark the usage of single row comparison operators when used in WHERE conditions.

2) Multiple row sub query
   i) Sub query returning multiple row output.
   ii) They make use of multiple row comparison operators like IN, ANY, ALL.
   iii) There can be sub queries returning multiple columns also.

Correlated sub query

i) It depend on data provided by the outer query.
ii) This type of subquery also includes subqueries that use the EXIST operator to test the existence of data rows satisfying specified criteria.
iii) Correlated subquery, the inner query is executed repeatedly once for each row that might be selected by outer query.
iv) It produce result tables that answer complex mgmt. questions.

e.g →
```
SELECT EMPLOYEE_ID, salary, Dept-id
From employees E
where salary > (SELECT AVG(salary)
                From EMP I
                where E.dept-id = I.dept-id)
```

Consider above query, unlike the subqueries previously considered, the subquery in this statement can't be resolved independently of the main query. Notice that the outer query specifies that rows are selected from the employees table with an alias name of e1. The inner query compares the employee dept. no. column of the employee table with alias e2 to the same column for the alias table name e1

* PostgreSQL

i) It is an enterprise class open source database mgmt. system.
ii) It supports both SQL & JSON for relational & non relational queries for extensibility & SQL compliance.
iii) It supports advanced data types & performance optimization features, which are only available in expensive commercial db. like Oracle, SQL SQL it is also knows as postgres.
iv) It is backed by an experienced community of developers who have made tremendous contributions to make it a highly reliable DBMS system

Advantages

i) It can run dynamic websites & web apps as LAMP stack option
ii) It's write-ahead logging makes it a highly fault tolerant DB
iii) Source code is freely available under an open source license. This allows you freedom to use, modify & implement it as per your business needs
iv) It supports geographic objects so you can use it for location based service & geographic info. systems.
v) To learn postgres you don't need much training as it's easy to use.
vi) Low maintenance & administration for both embedded & enterprise use of PostgreSQL

Disadvantages

i) It is not owned by one organization, so it has had trouble getting its name out there despite being fully featured & comparable to other DBMS systems.
ii) many open source apps support MYSQL, but may not support postgreSQL

3. NoSQL database Management

NOSQL.

i) NoSQL database is a non-relational data management system, that does not require a fixed schema

ii) It avoids joins & is easy to scale

iii) The major purpose of using NoSQL database is for distributed data stores with humongous data storage needs.

iv) NoSQL is used for Big data & real time web apps. e.g → Twitter, Facebook & Google collect terabytes of user data every single day.

v) It stands for "Not Only SQL" or Not SQL"

vi) RDBMS uses SQL syntax to store & retrieve data for further insights.

vii) NoSQL database system encompasses a wide range of database technologies that can store structured, semi structured, unstructured & polymorphic data.

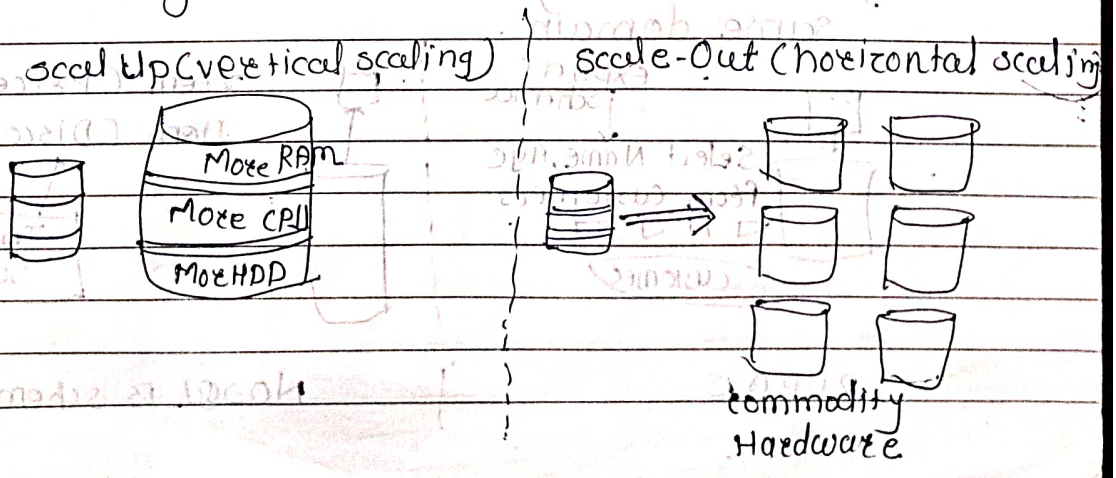
Why NoSQL

i) The concept of NoSQL database became popular with internet giants like Google, facebook, Amazon etc. who deal with huge volumes of data.

ii) The system response time becomes slow when you use RDBMS for massive volumes of data.

iii) To resolve this problem we could "scal up" our systems by upgrading our existing hardware so this process is expensive.

iv) The alternative for this issue is to distribute database load on multiple hosts whenever the load increases, this method is known is as "scaling out".



v) NoSQL database is non-relational, so it scales out better than relational databases as they are design with web applications in mind.

History of NoSQL DB

- 1998 - Carlo Strozzi use the term NoSQL for his lightweigh
- Open source relational database
- 2000 - Graph database
- 2004 - Google BigTable is launched
- 2005 - CouchDB is launched
- 2007 - The research paper on Amazon Dynamo is release
- 2008 - Facebook open sources the cassandra project
- 2009 - the term NoSQL was reintroduced.

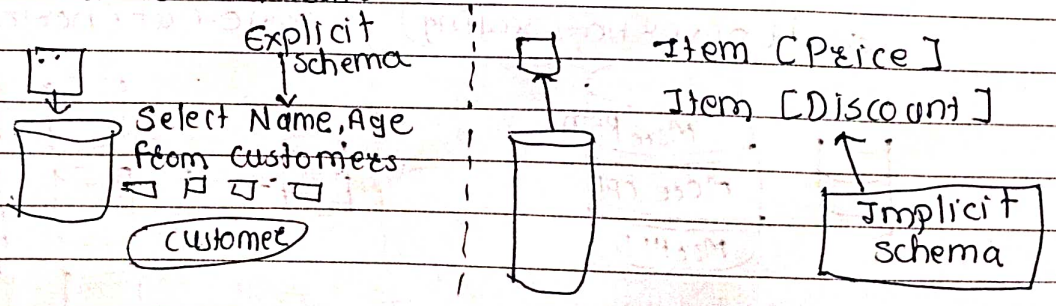
Features of NoSQL

1) Non-relational

i) NoSQL database never follow the relational model.
iii) Never provide tables with flat fixed-column records
iv) Work with self contained aggregates or BLOBs
v) Doesn't require object relational mapping & data normalization.
vi) No complex features like query language, query planners, referential integrity, joins, ACID.

2) schema free

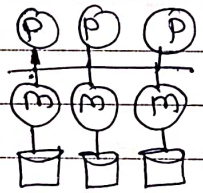
i) NoSQL databases are either schema free or have relaxed schemas.
ii) Do not require any sort of definition of the schema of the data.
iii) offers heterogeneous structures of data in the same domain.



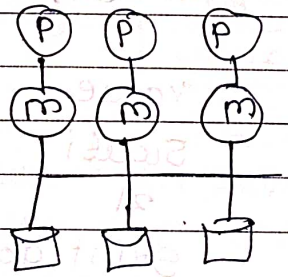
RDBMS | NOSQL is schema free

3) Simple API
i) Offers easy to use interfaces for storage & querying data provided.
ii) APIs allow low level data manipulation & selection methods.
iii) Text based protocols mostly used with HTTP REST) with JSON
iv) Mostly used no standard based NoSQL query lang.
v) Web-enabled databases running as internet facing services.

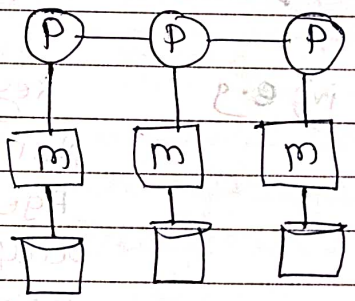
4) Distributed
i) Multiple NoSQL databases can be executed in a distributed fashion
ii) Offers auto scaling & fail over capabilities.
iii) Often ACID concept can be sacrificed for scalability & throughput
iv) Mostly no synchronous replication between distributed nodes Asynchronous Multi-Master Replication, peer-to-peer, HDFS Replication.
v) Only providing eventual consistency
vi) Shared Nothing Architecture. This enables less co-ordinations & higher distribution.



Shared mem.
e.g - oracle 11g



Shared disk
e.g oracle RAC



Shared Nothing
e.g NoSQL

Types of NoSQL Database

i) NoSQL database are mainly categorized into,
1) Key value pair
2) Column oriented
3) Graph based
4) Document oriented.

ii) Every category has it's unique attributes & limitations.

iii) None of the above specified database is better to solve all problems, User should select the DB based on their product needs.

1) Key value pair

i) Data is stored in key/value pairs. it is designed in such a way to handle lots of data & heavy load.

ii) Key value pair storage database store data as hash table where each key is unique & the value can be JSON, BLOB (Binary Large Objects) string etc.

iii) e.g → key value pair may contain a key like "Website" associated with a value like "Guregg"

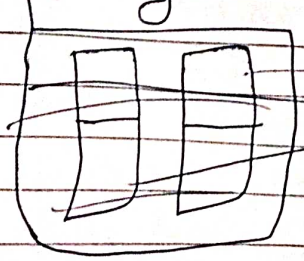
Database → project voldemart, riak. redis.

iv) e.g

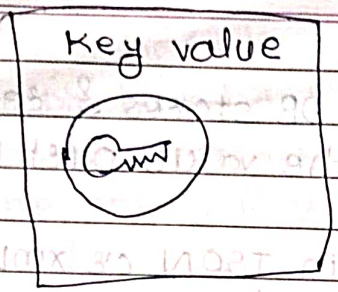
Key	value
Name	Scarlett
Age	21
Occupation	Stunt double
Height	175cm
Weight	47 kg

v) This kind of NoSQL database is used as collection, dictionaries, associative arrays etc.

vi) They work best for shopping cart contents.



example → Big table, ~~cassandra~~, Hbase, ~~Hypertable~~.



Example → Riak, Tokyo.cabinet, Redis seever, memcached scalaris.

② column Based

i) column oriented database work on docume column & are based on Big Table paper by Google.
ii) Every column is treated separately
iii) values of single column database are stored contiguously.
iv) e.g →

columnFamily	
Row Key	column Name Key Key Key value value value column Name Key Key Key value value value.

v) The deliver high performance on aggregation queries like SUM, COUNT, AVG, MIN etc. as the data is readily available in a column.
vi) column based NoSQL database are widely used to manage data warehouse, business intelligience CRM, Library card catalogs.
vii) HBase, Cassandra, Hypertable are NoSQL query example of column based database.

② Document oriented

i) Document oriented NoSQL DB stores & retrieve as a key value pair but the value part is store as document.

ii) The document is stored in JSON or XML formats

iii) The values is understood by DB & can be queri

col 1	col 2
Data	Data
Data	Data
Data	Data
Data	Data

```
Document 1
{ "prop 1" : data,
  "Prop2" : data,
  "prop3" : data,
}
```

Redational Vs. Document

iv) In above fig. on your left you can see we have rows & columns & in the right we have a document ⇒ a database which has similar structure to JSON

v) In relational db, you have to know what columns you have & so on, however for document database you have data store like JSON object, you do not require to define which make it flexible.

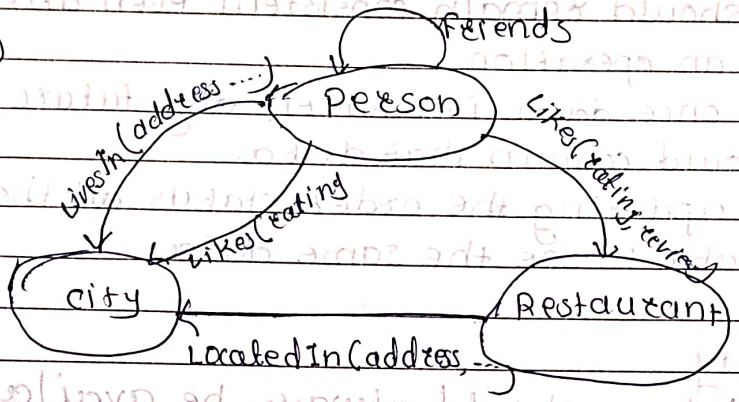
vi) The document type is mostly used for cms system blogging platforms, real time analytics & e-commerce appln.

vii) It should not use for complex transactions which require multiple operations or queries against varying aggregate structures.

ix) Amazon simple DB, couch DB, MongoDB, Riak, Lotus Notes are popular document originated DBms system.

(4) Graph based

- i) Graph based database stores entities as well as the relations amongst those entities.
- ii) The entity is stored as a node with the relationship as edges.
- iii) An edge gives relationship between nodes.
- iv) Every node & edge has a unique identifier.
- v)



- vi) Compared to a relational db where tables are loosely connected, a Graph database is a multi relational in nature
- vii) Traversing relationship is fast as they are already captured into the DB & there is no need to calculate them.
- viii) Graph based database mostly used for social networks, logistics, spatial data.
- ix) Neo4j, Infinite graph, Orient DB, FlockDB are some popular graph based database.

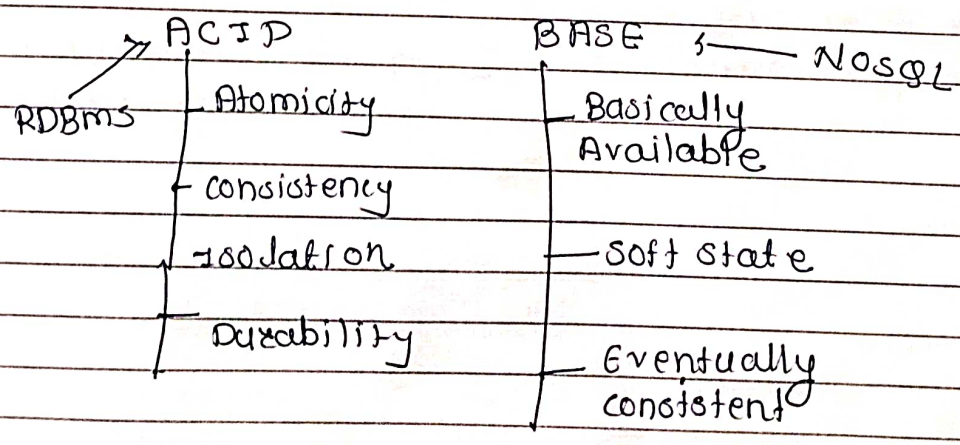
CAP Theorem (Brewer's theorem)

1) Consistency
2) Availability
3) Partition Tolerance

1) Consistency
- The data should remain consistent even after the execution of an operation
- This means once data is written any future read request should contain that data
- e.g after updating the order status, all the clients should be able to see the same data.

2) Availability
- The database should always be available & responsive
- It should not have any downtime.

3) Partition Tolerance
~~The database should always be available & responsive.~~
- The system should continue to run even if the communication among the servers is not stable
- e.g → servers can be partitioned into multiple groups which may not communicate with each other. Here if part of the db is unavailable other parts are always unaffected.



1) Basically Available
Database is available all the time as per CAP theorem

2) Soft state
Means even without an input, the system state may change.

3) Eventually consistency
- Means that the system will become consistent over time
- Eventually consistency means to have copies of data on multiple machines to get high availability & scalability. Thus changes made to any data item on one machine has to be propagated to other replicas.
- Data replication may not be instantaneous as some copies will be updated immediately while others in due course of time, these copies may be mutually but in due course of time they become consistent. Hence the name eventual consistency.

Database in MongoDB

i) MongoDB is an open source, cross platform, docum oriented database that provides high performance high availability & easy scalability
ii) It works on concept of collection & documents.

Database in MongoDB.

i) Database is physical container for collections.
ii) Each database gets it's own set of files on the file system.
iii) A single MongoDB server typically has multiple db

collection

i) collection is a group of MongoDB documents.
ii) collection exists within a single database.
iii) It do not enforce a schema.
iv) Documents within collection can have different fields.
v) All documents in a collection are of similar or related purpose.

Document.

i) A document is a set of key-value pairs.
ii) It have dynamic schema means that documents in the same collection do not need to have same set of fields or structure & column fields in collections documents may hold different types of data.

RDBMS	mongoDB
Database	Database
Table	collection
Tuple / Row	Document.
column	Field
Table join	Embedded documents
Primary key	Primary key (by default gener
Mysql / oracle	mongod (db server &
mysql / sqlplus	Mongo (client)

Advantages of MongoDB over RDBMS:

i) Schemaless
MongoDB is document database in which one collection holds different documents. No. of fields, content & size of the document can differ from one document to another

ii) Structure of single object is clear

iii) No complex join

iv) Deep quality ability
It support dynamic queries on documents using a document based query language that's nearly as powerful as SQL tuning.

v) Ease-of scale out
MongoDB is easy to scale.

vi) Conversion/mapping of appln objects to db objects not needed.

vii) Uses internal memory for storing the working set;

Installation of MongoDB

1) Pre-Requisite information on windows
1st check your windows 32/64 bit
go to cmd
→ C:\ wmic os get osarchitecture
above command shows the window is 32/64

2) Download mongoDB on windows from website
select community server
then select
windows [64-bit]
windows [32-bit]
then download

3) Open exe file, click next & accept license agreement & goes to the next & choose setup type, there are two setup type
i) complete & ii) custom
you may select complete & click on install.
& click on finish.

④ Go to the c disk & get into program files & th you will find "mongoDB" folder.

lock disk C → Prgm file → mongoDB → server → bin

copy above path.

⑤ Go to the computer preops & Advanced then select env. variable & paste above path. & edit & then ok, ok, ok.

⑥ Download Robo 3T setup, & open exe file, then go to next button, agree license & then browse destination folder.

e.g C:/program files\ Robo 3T 1.2.1.

& click on next. & select Robo 3T 1.2.1 & install then click on finish button.

⑦ mongoDB connection setting

click on create conn. then one windows is display

Type - Direct connection

Name - New conn.

Address - localhost : 27017

& save then connect.

Accessing MongoDB

① open cmd, copy path upto mongoDB server folder

C → Prgfile → mongodb - server - bin

& paste above path on cmd.

"Here paste above path /mongo.exe" - - version

it directly goes to the mongod,

otherwise you can directly & open the mongoDB cmd by using below steps

→ select C → Prgfile - mongodb - server - bin - mongod

click on mongod, it directly open mongoDB cmd.

CRUD operations

① Create Database

i) MongoDB use Database_Name is used to create a database, The command will create a new database, if it doesn't exist otherwise it will return existing db.

syntax use DATABASE_NAME

example → > use swati
switched to db mydb

ii) To check your currently selected database use command db

syntax → db database name

example → > db swati

iii) If you want to check your database list, then use command show dbs

> show dbs
local
swati

② Drop database

syntax → dropDatabase()
db.dropdatabase()

i) This will delete the selected database, if you have not selected any database, it will delete default test database.

ii) In mongoDB default database is test if you didn't create any database then collection will be stored in test database.

(I) Create documents.

i) To create database

→ Use Database name

e.g → Use Swati

switched to db swati

ii) To show database

> show dbs

It shows all database that are present in our MongoDB.

→ show dbs

↓

admin 0.0006B
config 0.0006B
local 0.0006B
Springboot 0.0006B
swati 0.0006B

If database does not contain any document then it doesn't show db, so u can insert atleast one documents.

iii) Insert Single Record (single ~~doc~~)

> db.employee.insert({name: "swati", emeil: "abc@gmail.com", age: 28})

↓

{ ack = true
Inserted id : objectId("")
}

iv) Insert many Record (capital)

db.employee.insertMany([{},{},{}]);

* Difference betn couchDB & MongoDB

Apache couch DB

i) It is developed by Apache software foundation & initially released in 2005.
ii) It is written in Erlang
iii) It is open source DB that uses different formats & protocols to store, transfer & process it's data.
iv) It uses JSON to store data, javascript as it's query lang. using Map Reduce.
v) Documents r primary unit of data in couch DB & they also include metadata.
vi) Document fields are uniquely named & contain values of varying types & there is no set limit to text size or element count.

	Couch DB	Mongo DB
1)	Data store in JSON format	Datastores in BSON format
2)	The DB contains documents	The DB contains collections
3)	It favors availability	It favors consistency
4)	It is written in Erlang	It is written in C++
5)	It is eventually consistent	It is strongly consistent.
6)	MongoDB is faster than couchDB	Mongo DB provides faster read speeds.
7)	It follows Map/Reduce query method	Map/Reduce creating a collection & object-based query langu
8)	CouchDB uses map reduce functions & it will difficult for users with traditional SQL learning experience	Mongo DB is easier to learns as it is closest in syntax to SQL

Sulekha

4. Database Security & Authorization

Introduction

There are 3 main objectives when designing secure database application:

1) Secrecy → Information should not be disclosed to unauthorized users.
   e.g → student should not be allowed to examine other students grades.
2) Integrity → only authorized users should be allowed to modify data.
   e.g → student may be allowed to see their grades, yet not allowed to modify them.
3) Availability → Authorized user should not be denied access.
   e.g → An instructor who wishes to change a grade should be allowed to do so

To achieve these objectives, a clear & consistent security policy should be developed to describe what security measures must be enforced.

Access control

- A database for an enterprise contains a great deal of info. & usually has several groups of users.
- Most users need to access only small part of the DB to carry out their tasks.
- User's unrestricted access to all the data can be undesirable & DBMS should provide mechanisms to control access to data.
- A DBMS offers two main approaches to access control
  i) Discretionary access control
  ii) Mandatory access control

1) Discretionary Access control
- It is based on the concept of access rights, or privileges & mechanisms for giving users such privileges
- A privilege allows a user to access some data object
- A user, who creates a database object such as a table or views automatically, gets all applicable privileges on that object.
- SQL supports discretionary access control through the GRANT & REVOKE commands.

GRANT
- The GRANT command gives users privileges to base tables & views.
- The syntax of this command is as follows

GRANT privilege ON object TO users [WITH GRANT OPTION]

- For our purposes object is either a base table or a views. Several privileges can be specified including these

i) SELECT
- the right to access (read) all columns of the table specified as the object including columns added later through ALTER TABLE commands.

ii) INSERT (column name)
- The right to insert rows with non null values in the column of the table named as object. If this right is to be granted with respect to all columns, including columns that might be added later, simply use INSERT
- The privileges UPDATE (column name) & UPDATE are similar.

iii) DELETE -
The right to delete rows from the table named as object.

REFERENCES (column-name):

- The right to define foreign keys that refer to the specified column of the table object.
- REFERENCES without column name specified denotes this right w.r.t all columns, including any that are added later.

GRANT & REVOKE ON VIEWS & INTEGRITY constraints

- The privileges held by the creator of view change over time as he/she gains or loses privileges on the underlying tables.
- If the creator loses a privilege held with the grant option, users who where given that privilege on the view lose it as well.
- There are some subtle aspects to the GRANT & REVOKE commands when they involve views or integrity constraints
- We consider some e.g that highlight the foll. imp. points

i) A view may be dropped because a SELECT privilege is revoked from the user who created the view.

ii) If the creator of a view gains additional privileges on the underlying tables, he or she automatically gains additional privileges on the view.

The need for & Role of database in organization

1) Data are used by different people in different dept. for different reasons.

DBMS facilitates

a) interpretation & presentation of data in useful formats by transforming raw data into info.

b) Distribution of data & info. to the right people at right time

c) Data presentation & monitoring the data asage for adequate periods of time.

d) control over data duplication & use both internally & externally

- whatever the type of org. the db predominant role is to support managerial decision making at all levels in the org. while preserving data privacy & security

- An org. managerial structure might be divided into 3 level

1) Top → Strategic decision
2) middle → tactical decision
3) operational → daily operational decision

1) Top management derived

i) Provide the info. necessary for strategic decision making strategic planning, policy formulation & goal defn.

ii) Provide access to external & internal data to identify growth opportunities & to chart the direction of such growth.

iii) Provide a framework for defining & enforcing organizational policies.

iv) Improve the like hood of tve return on investment for the company by searching for new ways to reduce costs & By boosting productivity

v) Provide feedbacks to monitor whether the company is achieving it's goals.

2) Middle management level
i) Deliver the data necessary for tactical decision & plan
ii) Monitor & control the allocation & use of company resources & evaluate the performance of various dept.
iii) Provide framework for enforcing & ensuring the security & privacy of data in db

Privacy → It deals with the rights of individuals & the org. to determine the "who, what, when, where & how" of data usage.

Security → It protecting the data against accidental / ~~de~~ intentional use by unauthorized users.

3) operational Mgmt level.

i) Represent & support the company opern as closely as possible. The data model must be flexible enough to incorporate all required present & expected data.
ii) Produce query results within specified performance levels. The performance requirements increase for lower levels of mgmt & opern
iii) Enhance the company's short term operational ability by providing timely info. for customer support & for appln development mgmt & computer opern.

* Evolution of DB Adminstration function
- Data adminstn has it's roots in odd, decentralized world of the file system
- The cost of data & managerial duplication in such file systems gave rise to centralized data admnstn fun. known as the electronic Data Processing (EDP) or data processing (DP) dept.
- The dP dept. to evolve into an Information system (IS) dept.
- Responsibility of IS
i) A service fun. to provide end users with active data rngmt. support

ii) A production fan. to provide end users with specific soln for their info. needs through integrated appln or mgmt info. system.

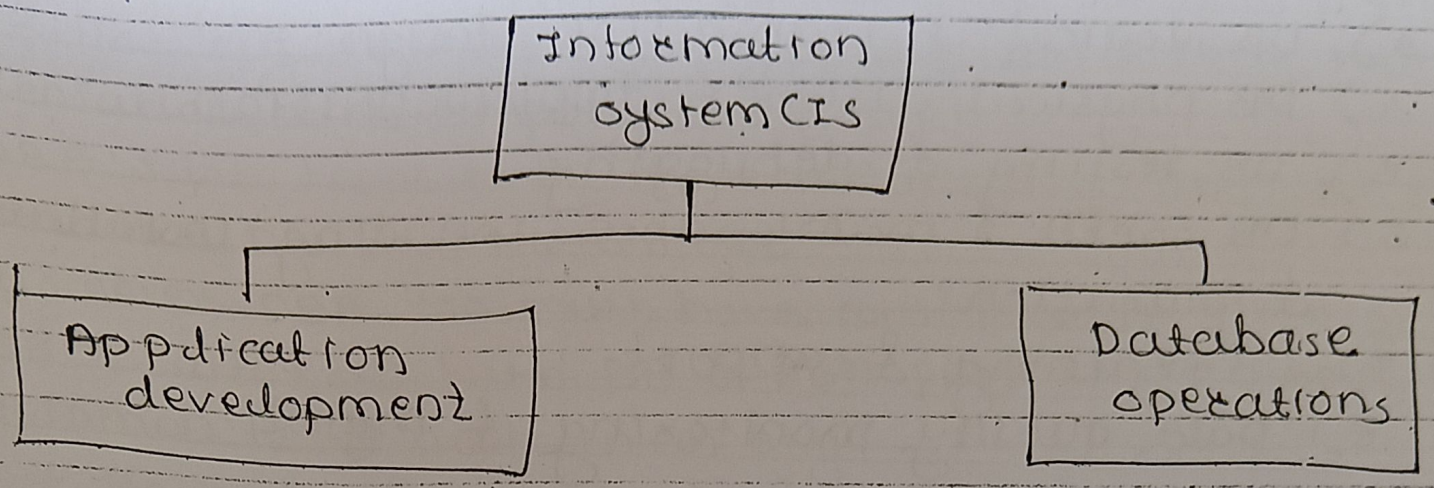
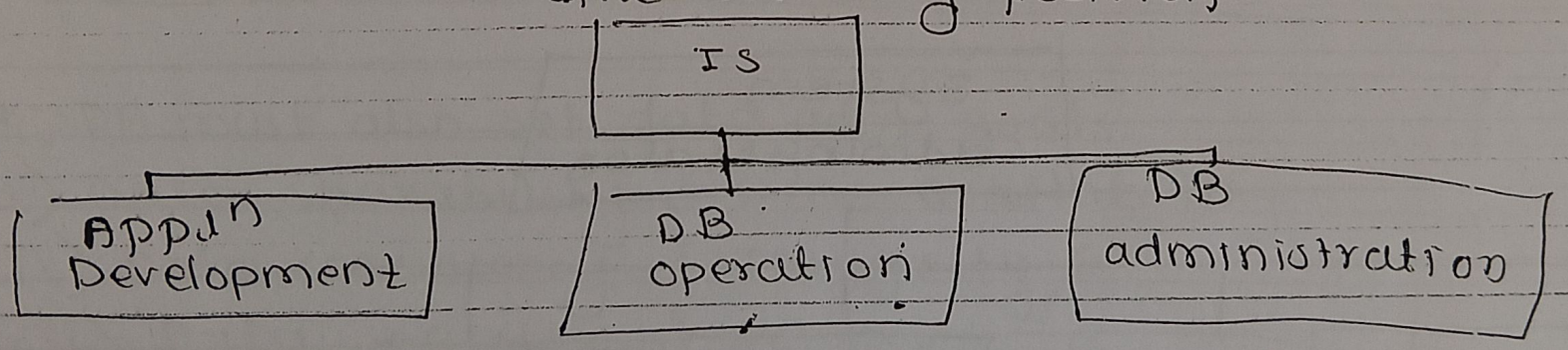


fig → IS dept. intenal org.

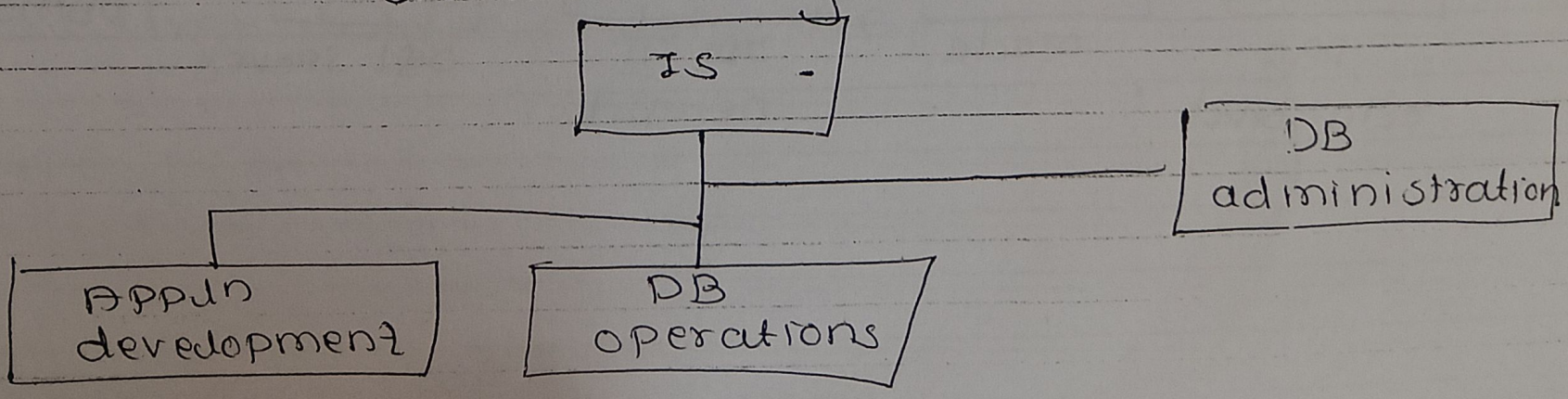
In above fig. as the demand for appln development grew the IS appln development segment was subdivided by the type of supported system: accounting, inventory, marketing & so on. This development meant that DB admin responsibilities were divided. The appln development segment was in charge of gathering DB requirements & logical DB design whereas the DB oper. segment took charge of implementing, monitoring & controlling the DBMS opern.

* Placement of DBA fun:

Line authority position



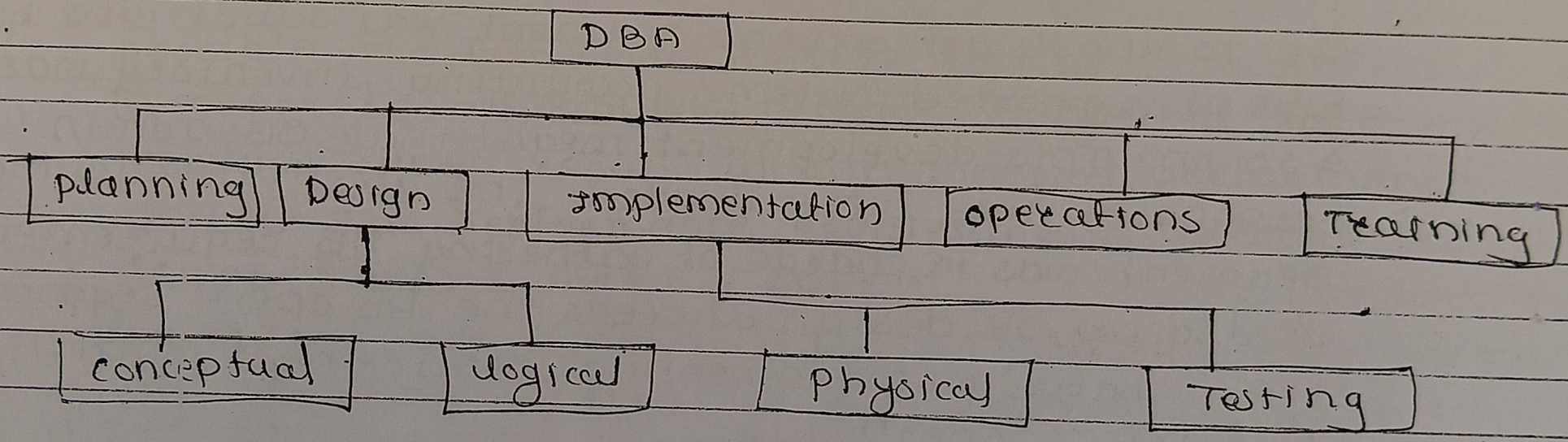
staff consulting position



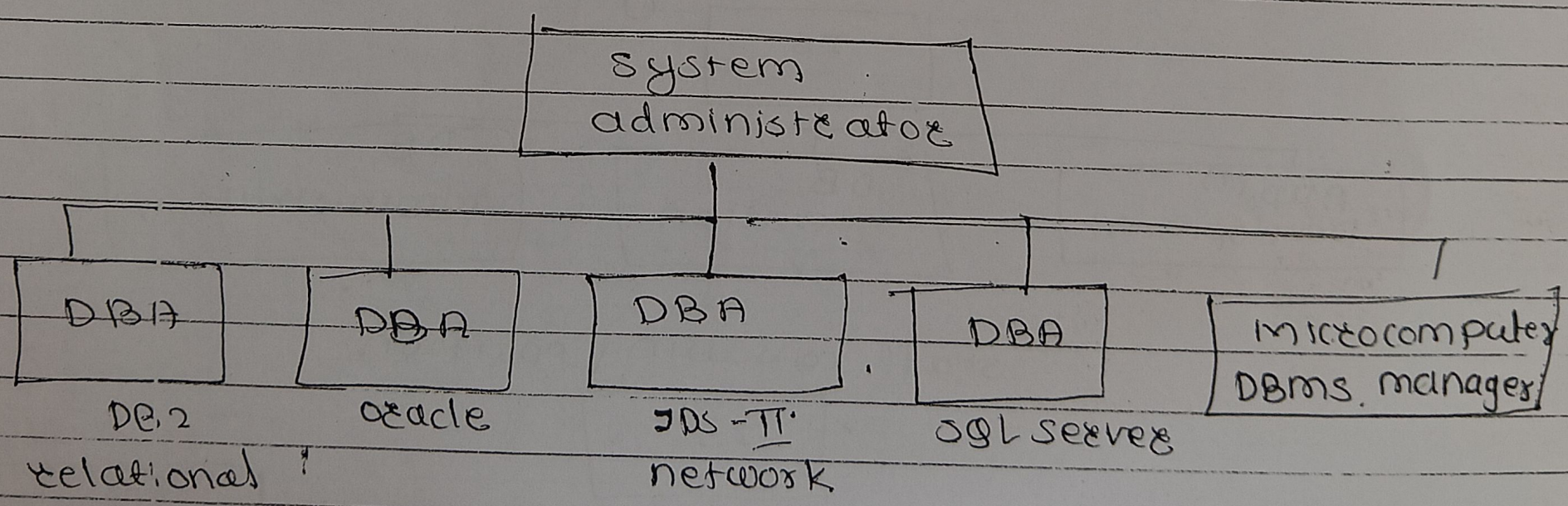
DBE requires personnel to cover the foll. activities

1) DB planning, including the defn of standards, procedures & enforcement.
2) DB requirements gathering & conceptual design.
3) DB logical & transaction design.
4) DB physical design & implementation
5) DB testing & debugging
6) DB opern & maintenance, including installation, conversion & migration.
7) DB training & support
8) Data quality monitoring & mgmt.

A DBA functional organization.



Multiple DB admin. in an organization



The DB environments Human components

i) Effective data admin. requires both technical & managerial skills. e.g the DA's job typically has strong managerial orientation with company wide scope.

ii) DBA's job tends to be more technically oriented & has a narrower DBMS specific scope.

iii) However DBA, too must have considerable store of people skills after all both DA & DBA perform "people" fun.. common to all dept. in an org.

E.g → Both DA & DBA direct & control personnel staffing & training within their respective dept.

DA	DBA
i) Does strategic planning	controls & supervises
ii) sets long term goals	Executes plans to reach goals
iii) Sets policies & standards	Enforces policies & procedures
	Enforces programming std
iv) Is broad in scope	Is narrow in scope
v) Focused on the long term	Focuses on short term
vi) Has managerial orientation	Has a technical orientation
vii) Is DBMS independent	Is DBMS specific

* DA must also set data adm. goals

i) Data sharability & time availability
ii) Data consistency & Integrity
iii) Data security & privacy
iv) Data quality std.
v) Extent & type of data use

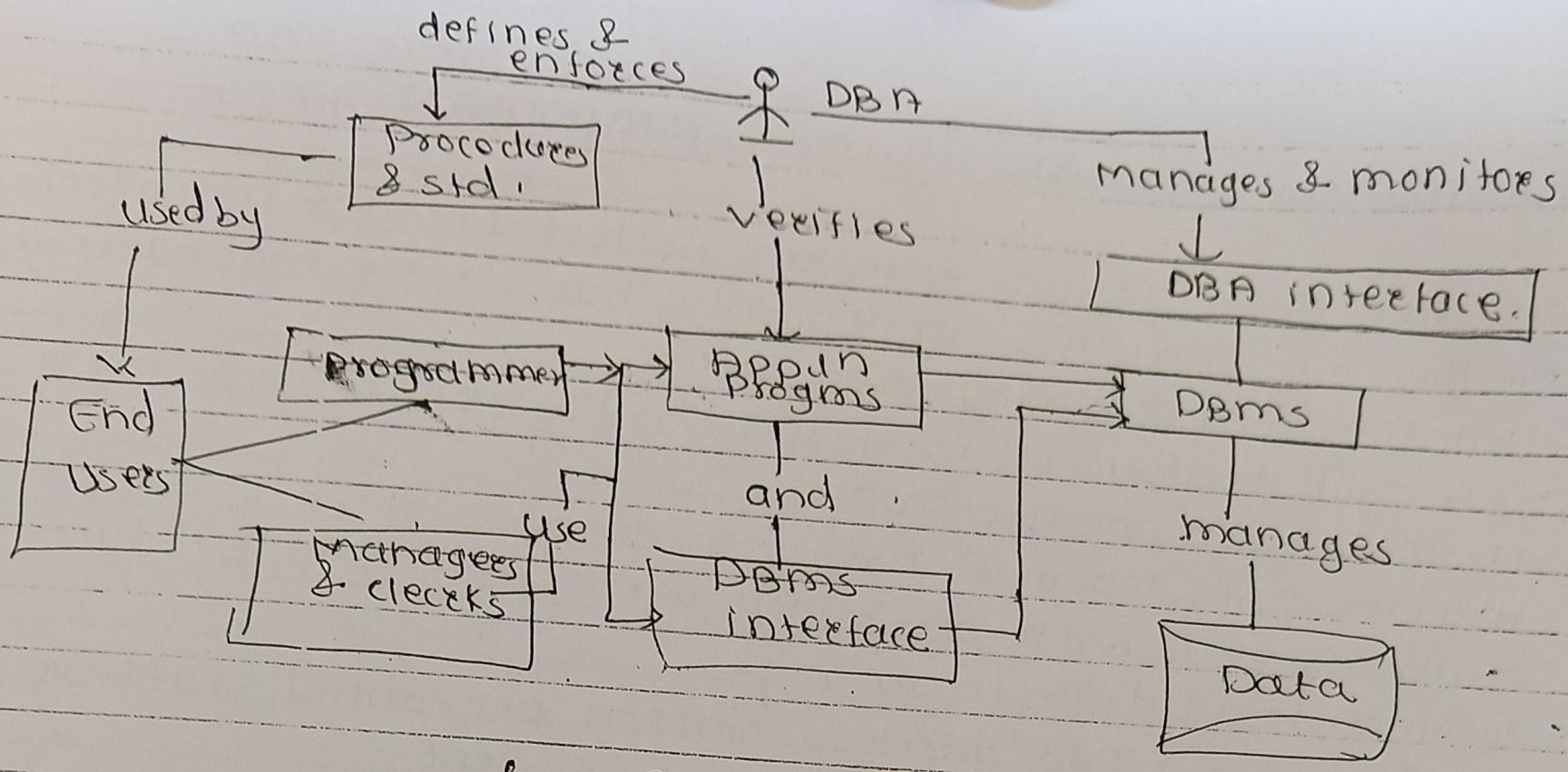


fig. DBA activities.

✱ DBA activities & services

planning
organizing
Testing
Monitoring
Delivering
} ← of →
{ End user support
Policies, procedures & std.
Data security, privacy & integrity
Data backup & recovery
Data distribution & use.

1) End user support

DBA interact with the end user by providing data & info. support services to the org. dept. because end users usually have dissimilar computer backgrounds, end user support services include.

i) Gathering user requirements
ii) Building end user confidence
iii) Resolving conflicts & problems.
iv) Finding soln to info. needs.
v) Ensuring quality & integrity of data & appln
vi) Managing & training & support DBMS users

2) Policies, procedures & standards.

i) Policies are general statements of direction or action that communicate & support DBA goals

iii) Standeds → It describes minimum requirements of give DBA activity they are more detailed & specific than podicies.

e.g. standards define the structure of appln progms & naming conventions programmers must use.

iii) Procedures

It written instructions that describe series of steps to be followed during performance of given activity

3) Data security, privacy & integrity.

The security, privacy & integrity of data in DB are great concern to DBAs who manage current DBMs installation

4) Data backup & Recovery

- when data are not readily available, companies face potentially ruinous losses, therfore, data backup & recovery procedures are critical in all DB installation
- The DBA must also ensure the data in DB can be fully recovered in case of physical data loss of DB integrity

5) Data distribution & Use

i) Data are useful only when they reach the right users in timely fashion

ii) The DBA is responsible for ensuring that the data are distributed to the right people at right time & in right formal

Database administration Tool:

The data dictionary as a data administration tool as well as the DBMS use of computer Aided Software Engineering (CASE) tools to support database analysis & design.

1) Data Dictonary

i) The DBMS data dictionary provides the DBMS with its self describing characteristics. In effect data dictionary resembles an X-ray of the company's entire data set & it is crucial element in data administration

ii) Two main types of data dictionaries
   a) Integrated
   b) standalone

iii) An Integrated data dictionary is included with DBMS. e.g. all relational DBMS include built in data dictionary or system catlog i.e frequently accessed & updated by RDBMS.

iv) other DBMS older type do not have built in data dictonary instead the DBA may use 3rd party standalone data dictionary system.

Data dictionaries can also classified as active or passive

1) Active data dictionary
   It is automatically updated by the DBMS with every DB access, thereby keeping it's access info. up to date.

2) Passive data dictionary
   It is not updated automatically & usually requires running a batch process.

There is no standard format for the info. stored in the data dictionary, several common features are

1) Data elements that are defined in all tables of all database. It stores the names, data types, display formats, internal storage format & validation rules.

2) Tables defined in all databases, it store the name of table creator, date of creation, access authorization & no. of columns.
3) Indexes defined for each database table. for each index DBMS stores at least index name, attribute used location, specific authorizations & no. of columns. index characteristics & creation date.
4) Defined DB. This include who created each DB, when & where DB is located, who the DBA.
5) End users & admin. of the DB
6) Programs that access the DB. This include screen formats, report formats, appln programs & sql queries
7) Access authorization of for all users of all DB
8) Relationship among data elements.

Examples of Data Dictionary Usage

1) List the names & creation dates of all table created by user Swati in the current DB.

→

```
SELECT  NAME, CTIME
FROM    SYSTABLES
WHERE   CREATOR = 'SWATI';
```

(2) Case Tools

i) CASE is the acronym for Computer Aided systems Engg.
ii) It provides an automated framework for the system development Life cycle (SDLC).
iii) CASE uses structured methodologies & powerful graphical interfaces
iv) It play an increasingly important role in Info. system development.
v) CASE tools are usually classified according to the extent of support they provide for SDLC
e.g → front end CASE tools that provide support for the planning & design phases, back end case tools provide support for the coding & implementation phases

Sulekha

CASE Tools include

i) A reduction in development time & costs
ii) Automation of the SDLC
iii) standardization of systems development methodologies
iv) Easier maintenance of appln systems developed with CASE tools.

- One of the CASE tools most important components is an extensive data dictionary which keeps track of all objects created by the systems designer.
- e.g. CASE data dictionary stores data flow diagrams, structure charts, descriptions of all external & internal entities, data store, data items, report formats & screen formats.

A typical CASE tool provides 5 components

1) Graphics designed to produce structured diagrams such as data flow diagrams, ER diagrams, class diagrams & object diagrams.
2) Screen painters & report generators to provide the info. system i/p & o/p format.
3) An integrated repository for storing & cross referencing the system design data. This repository include a comprehensive data dictionary.
4) An analysis segment to provide a fully automated checks on system consistency, syntax & completeness
5) A program documentation generator.

CASE Tools

Casewise	Corporate Modeler suite
Computer Associates	ERwin
Embarcadero Technologies	ER/Studio
Microsoft	Visio
Oracle	Designer
IBM	System Architect
Sybase	Power Designer
Visible	Visible Analyst

* Developing a Data Administration Strategy.

i) The database administration strategy must not conflict with IS plans. After all, the IS plans are derived from detailed analysis of company's goals, it's condition & situation & business needs.

ii) Several methodologies are available to ensure the compatibility of data admin & IS plans & to guide the strategic plan development.

iii) Information Engg. (IE) allows for the translation of company's strategic goals into data & app. that will help company achieve those goals.

iv) The output of the IE process is an IS architecture (ISA) that serves as the basis for planning development & control of future IS.

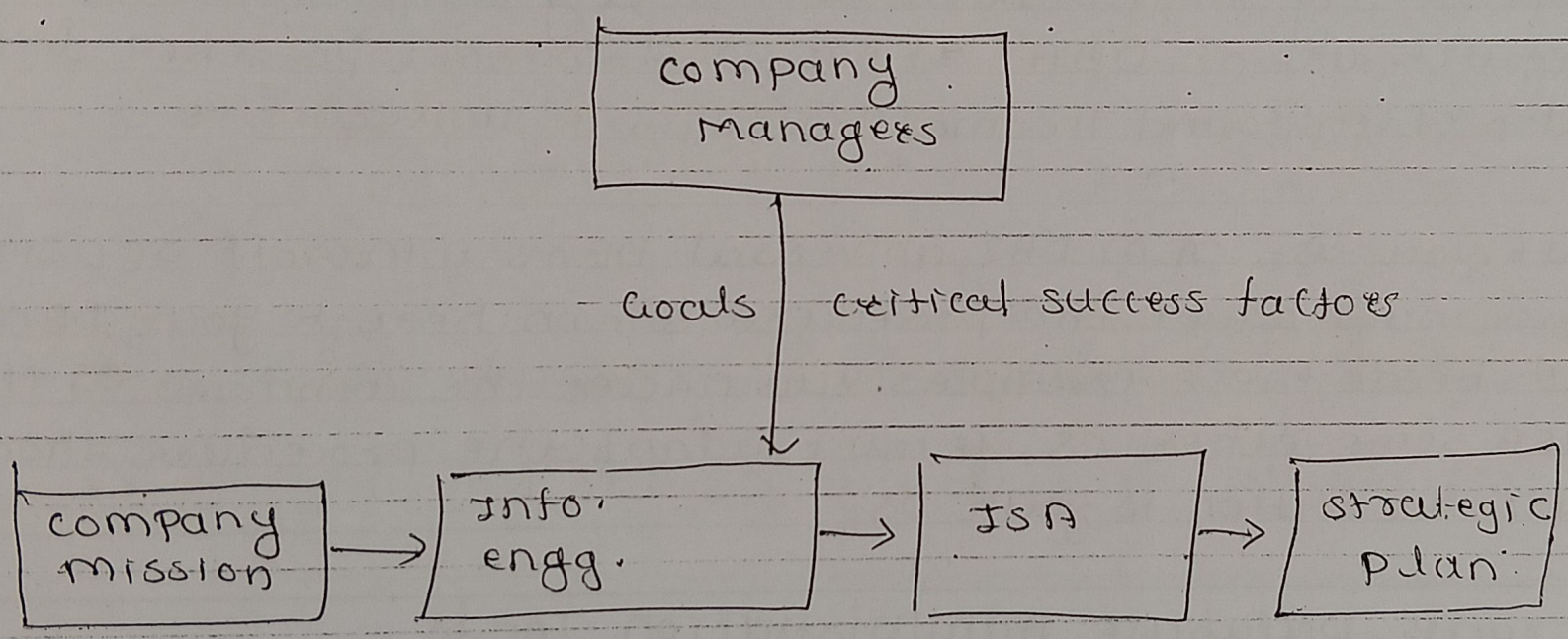


fig. Forces affecting development of the ISA

- Implementing IE methodologies is an org. is a costly process that involes planning, commitment of resources, mgmt. stability, well defined objectivies, indentification of critical factors & control.
- ISA provides framework that includes the use of computerized, automated & integrated tools such as DBMS & CASE tools.

The DBA at work: Using Oracle for DB administration
How DBA might handle the following technical tasks in specific DBMS

1) creating & expanding database storage structures
2) Managing DB object such as tables, indexes, triggers & procedures
3) Managing end user DB env. including type & extent of DB access
4) customizing DB initialization parameters

- All DBMS vendors provide set of programs to interface with DB & to perform wide range of DB admin^{n} tasks

- We choose oracle 11g for windows to illustrate the selected DBA tasks becor it is typically found in org. that are sufficiently large & have sufficiently complex db env. to require use of DBA. It has good market presence & it is also often found in small colleges & universities

- If you use IBM DB2 universal DB or Microsoft SQL Server, you must adapt the procedures shown here to your DBMS & becor these examples run under the windows OS if you use some other OS, u must adapt the procedures shown in this section to your OS.

Oracle Database Administration Tools.

1) All DB vendors supply a set of DBA tools. In oracle you perform most DBA tasks via oracle enterprise Manager Interface
2) The default login → To perform any administrative task you must connect to the DB using username with DBA privileges. By default oracle automatically creates SYSTEM & SYS user ids that have administrative privileges with every new DB you create
3) Ensuring an automatic RDBMS start
One of the basic DBA tasks is to ensure that your DB access is automatically started when you turn on the computer

& tablespace & Datafield

- A database is logically composed of one or more tablespace. A tablespace is logical storage space. & are used primarily to group related data logically.

5) - oracle automatically creates the tablespaces & datafiles
- i) The SYSTEM tablespace is used to store data dictonary data
2) USERS → Is used to store table data created by end users
3) TEMP → to store temporary tables & indexes created during the execution of SQL statements
4) UNDO TBS 1 - used to store DB transaction recovery info.

(5) Managing the DB objects, tables, views, triggers & procedures
- Oracle Enterprise Manager gives the DBA graphical user interface to create, edit, view & delete DB objects in DB.

6) Managing Users & Establishing security

7) Customizing the DB initialization parameters

5. Business Intelligence & Data Warehouse

ability to apply/acquire Knowledge

The Need for Data Analysis

- Manager must be able to track daily transaction to evaluate how the business is performing.
- Strategies should be developed to meet organizational goals using operational databases.
- Data analysis provides info. about short term tactical evaluations & strategies.

Business Intelligence

Defination

Comprehensive, cohesive, integrated tool & process to capture, collect, integrate, store & analyze data & the generate info. to support business decision making.

= It includes BI tools & tech. to transform raw data into meaningful & actionable info.

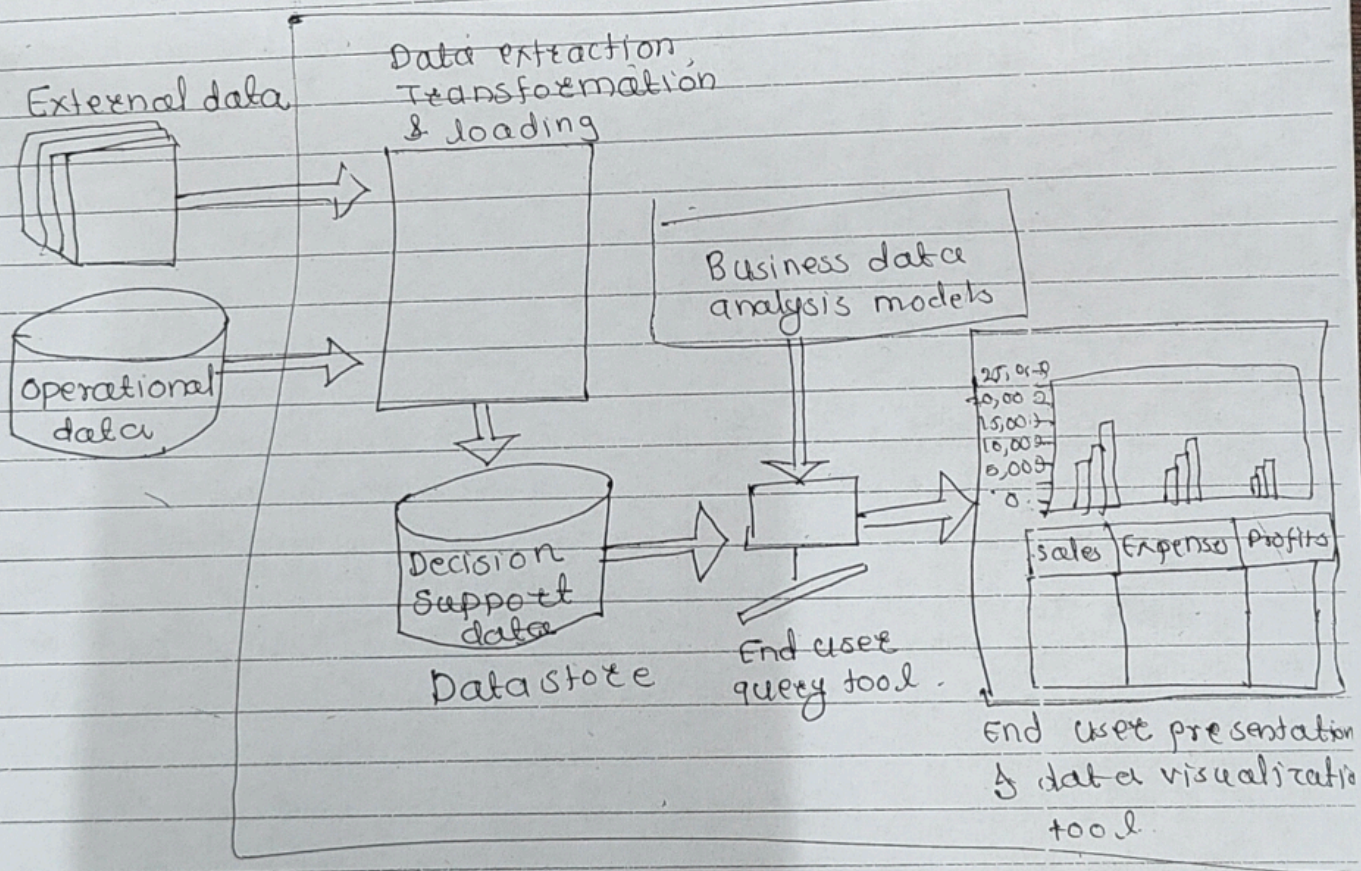
- Framework that allows a business to transform
  - Data into info.
  - info into knowledge
  - knowledge into wisdom.

- Implementing BI captors not only business data (internal & external) but also metadata.
- BI involves the following general steps.

1) Collecting & storing operational data
2) Aggregating the operational data into decision support data.
3) Analyzing decision support data to generate information
4) Presenting such info. to the end user to support business decision.
5) Making business decisions which in turn generate information. more data is collected, stored etc. (restarting the process)

6) Monitoring result to evaluate outcomes of the business decisions (Providing data to be collected, stored etc)

i) composed of data, people, processes, technology & management of components.
ii) Focuses on strategic & tactical use of information
iii) Multiple tools from different vendors can be integrated into a single BI framework.
iv) KPI (Key Performance indicators) measurements that assess companys effectiveness & or success in reaching goals.
v) Master Data management
A collection of concepts, techniques & processes for the proper identification, definition & management of data elements within an organi"



(1) Decision support Data

i) Operational data
- Mostly stored in relational database
- Optimized to support transactions representing daily operations.

ii) Decision support data
It differes from operational data in three main areas
- Time span
- Granularity
  - drill down & roll up to different levels of aggregation.
- Dimensionality

iii) Operational data have narrow timespan, low granularity & single focus, such data are usually presented in tabular format in which each row represents a single transaction. This format often makes it difficult to derive useful information

iv) DSS data focus on broader timespan tend to have high level of granularity & can be examined in multiple dimensions, for example
- sales by product, region, agent etc.
- sales for all years or only few selected years.
- sales for all products or only few selected products.

characteristics	Operation Data	DSS Data
Data currency	current operations Real time data	Historic data, snapshot of company data, Time component (week / month / year)
Granularity	Atomic detailed data	summarized data
summarization level	low, some aggregate yields	High, many aggregation levels
Data model	Highly normalized mostly relational DBMS	Non-normalized, complex structures some relational but mostly multidimensional DBMS
Transaction Type	Mostly updates	Mostly query
Transaction volumes	High update volumes	Periodic loads & summary calculations
Transaction speed	Updates are critical	Retrievals are critical
Query activity	Low to medium	High
Query scope	Narrow range	Broad range
Query complexity	simple to medium	very complex.
Data volumes	Hundreds of megabytes up to gigabytes.	Hundreds of gigabytes up to terabytes.

Decision support DB requirements
- Specialized DBMS tailored to provide fast answers to complex queries.

i) For main requirement
- Database schema
- Data extraction & loading
- End user analytical interface
- Database size.

ii) Database schema
- must support complex data representations.
- Bitmap indexes, data partioning, non-normalized
- Must contain aggregated & summarized data
- Queries must be able to extract multidimensional time slices.

iii) Data extraction & filtering
- should allow batch & scheduled data extraction
- Support different data sources.
- Flat files, Hierarchical, network & relational databases, multiple vendors.

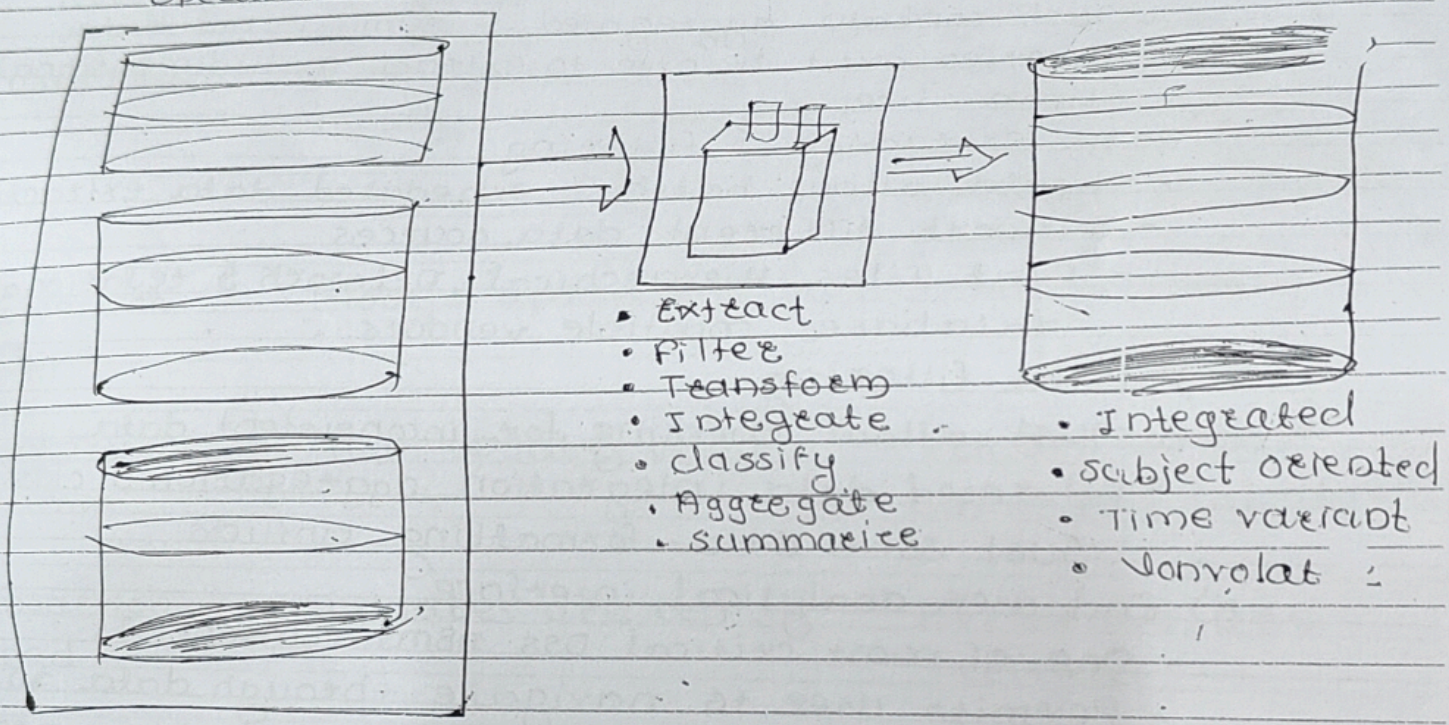
iv) Data filtering.
- Must allow checking for inconsistent data.
- Advanced data integration, aggregation & classificat
- Must solve data-formatting conflicts.

v) End user analytical interface.
- One of most critical DSS DBMS components
- Permits user to navigate through data to simplify & accelerate decision making process.

vi) Database size.
- DBMS must support very large databases
- Might be required to use advanced hardware such as disk arrays, symmetric multiprocessor or a massively parallel processor.

Data warehouse
- Integrated, subject oriented, time variant & nonvolati collection of data
- It provides support for decision making
- Usually a read only database optimized for data analysis & query processing.
- Requires time money & considerable managerial effort to create.

creating data warehouse

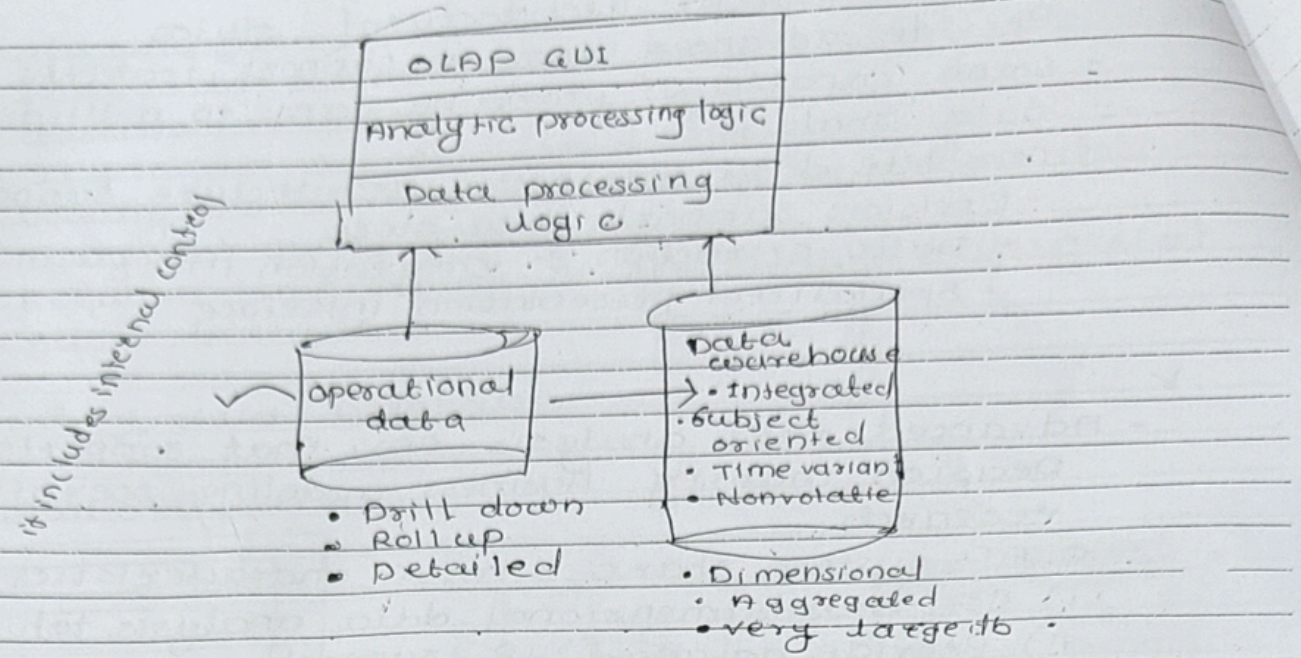


· Data mart
- Small, single subject, data warehouse subset
- More manageable data set than data warehouse
- Provides decision support to small group of people
- Typically lower cost & lower implementation time than data warehouse

Decision support architectural styles
- Provide advanced decision support features.
- Some capable of providing access to multidimensional data analysis
- Complete data warehouse architecture supports
  - Decision support data store
  - Data extraction & integration filter
  - Specialized presentation interface

*
- Advanced data analysis env. that supports Decision making, Business modeling, operations research.
- OLAP system share 4 main characteristics.
  i) Use multidimensional data analysis tech.
  ii) Provide advanced dB support
  iii) Provide easy to use end user interface
  iv) Support client/server architecture
- Easy to use end user interface
- Advanced OLAP features more useful when access is simple
- Many interface features are borrowed from previous generations of data analysis tools.

OLAP Architecture
- Operational characterestics 3 main modules.
  - Graphical User Interface (GUI)
  - Analytical processing logic
  - Data processing logic
- Designed to use both operational & data warehouse data.
- In most implementations, data warehouse & OLAP are interrelated & complementary
- OLAP systems merge data warehouse & datamart approaches.



1) Multidimensional

→ i) In multidimensional analysis, data are processed & viewed as part of multidimensional structure.

ii) This type of analysis is particularly attractive to business decision makers because they tend to view business data as data that are related to other business data

iii) multidimensional data analysis tech. are augumented by the following fun.

a) Advanced data presentation functions.
- 3-D graphics, pivot tables, crosstabs, data rotation & 3 dimensional cubes. such facilities are compatible with desktop spreadsheets, statistical packages & query & report packages.

b) Advanced data aggregation, consodidation & classification fun.

→ These allow the data analyst to create multiple data aggregation levels, slice & dice data. & drill down & roll up data across different dimension & aggregation levels.

e.g → Aggregating data across the time dimensions by week, month, quarter & year allow data analyst to drill down & rollup across time dimensions.

c) Advanced computational functions.
- These include business oriented variable, financial & accounting ratio & statistical & forecasting function. These fun- are automatically & end user does not need to redefine their components each time they are accessed

d) Advanced data modeling functions.
These provide support for what if scenarios, variable assessment, variable contributions to outcome, ~~their~~ linear programming & other modelling tools.

② Advanced Database support.
i) Access to many different kinds of DBMS, flat files & internal & external data sources.
ii) Access to aggregated data warehouse data as well as to detail data found in operational databases.
iii) Advanced data navigation features such as drill down & roll up.
iv) Rapid & consistent query response times.
v) The ability to map end user requests, expressed in either business or model terms to the appropriate data source & then to the proper data access language
vi) It support for very large databases.

③ Easy to use End user Interface.
i) OLAP tool vendors learned this lesson early & have equipped their sophisticated data extraction & analysis tools with easy to use graphical interface
ii) Many of the interface features are "borrowed" from previous generation of data analysis tools that are already familiar to end user. This familiarity makes OLAP easily accepted & readily used.

Sulekha

OLAP Architecture

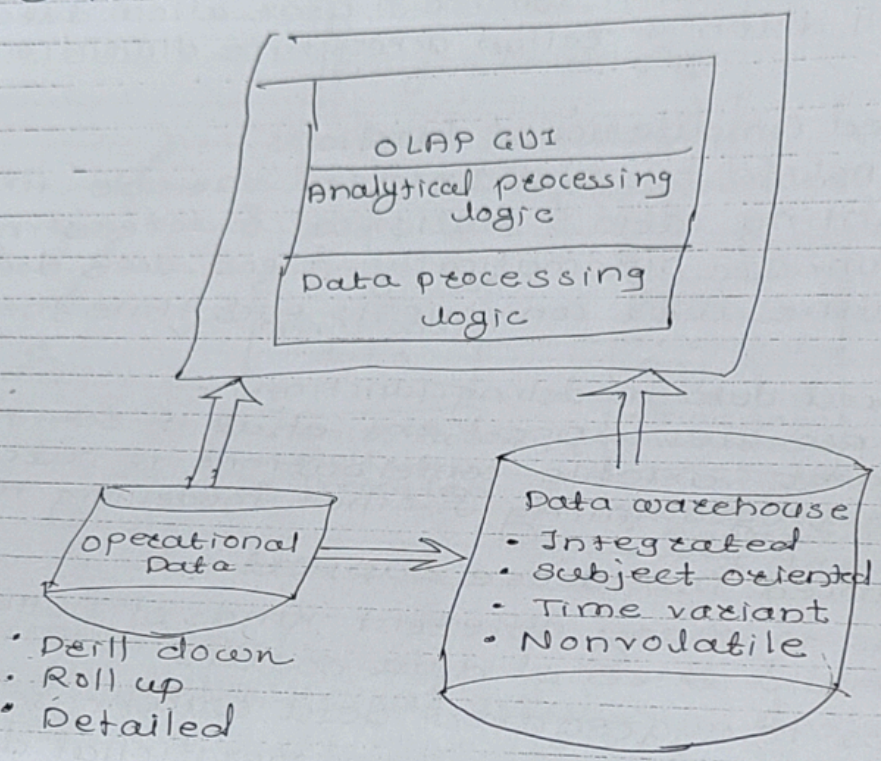


Fig.

OLAP can be divided into three main modules
i) GUI (Graphical User Interface)
ii) Analytical processing logic
iii) Data processing logic

* OLAP systems are designed to use both operational & data warehouse data.
* Fig. shows OLAP system components located on single computer but this single user scenario is only one of many

Data warehousing

i) Data warehouse is the foremost repository for the data available for developing BI architectures & decision support systems.

ii) The data warehousing indicates the whole set of interrelated activities involved in designing, implementing & using a data warehouse.

iii) It is possible to identify 3 main categories of data feeding into a data warehouse
   a) Internal data
   b) external data
   c) Personal data

a) Internal data

i) Internal data are stored for the most part in db, referred to as transactional systems or operational systems, that are backbone of an enterprise info. system.

ii) Internal data are gathered through transactional appln that routinely preside over the operation of a company such as administration, accounting, production & logistics

iii) This collection of transactional biz appln is termed enterprise resource planning (ERP)

iv) The data stored in the operational systems usually deal with main entities involved in a company processes namely customers, products, sales, employees & suppliers These data usually come from different components of the info. system

   a) back office system

      That collect basic transactional records such as orders, invoice, inventories, production & logistic data.

   b) front office systems

      -hat contain data originating from call center activities, customer assistance, execution of marketing compaigns

c) web based systems
That gather sales transactions on e-commerce, web sites, visits to websites, data available on forms filled out by existing & prospective customers.

2) External data
i) There r several sources of external data that may be used to extend the wealth of info. stored in the internal DB.
ii) e.g - some agencies gather & make available data relative to sales, market share & future trend predictions for specific business industries as well as economic & financial indicators.
iii) source of external data is provided by geographic information system (GIS) which represent a set of appln for acquiring organizing & storing & presenting territorial data.

3) Personal data
In most cases, decision makers performing B.I analysis also rely on info. & personal assessments stored inside worksheets or local db located in their computers. The retrieval of such info. & its integration with structured data from internal & external sources is one of the objectives of knowledge mgmt. system.

* Data Warehouse architecture

Figure includes the following major components:
1) The data warehouse itself, together with additional data marts, that contains the data & fun. that allow the data to be accessed, visualized & perhaps modified.
2) Data acquisition appln also known as extract, transform & load (ETL) or back end tools which allow the data to be extracted, transformed & loaded into the data warehouse.

3) Business intelligence & decision support appln which represent the front end & allow the knowledge workers to carry out the analyses & visualize the results.

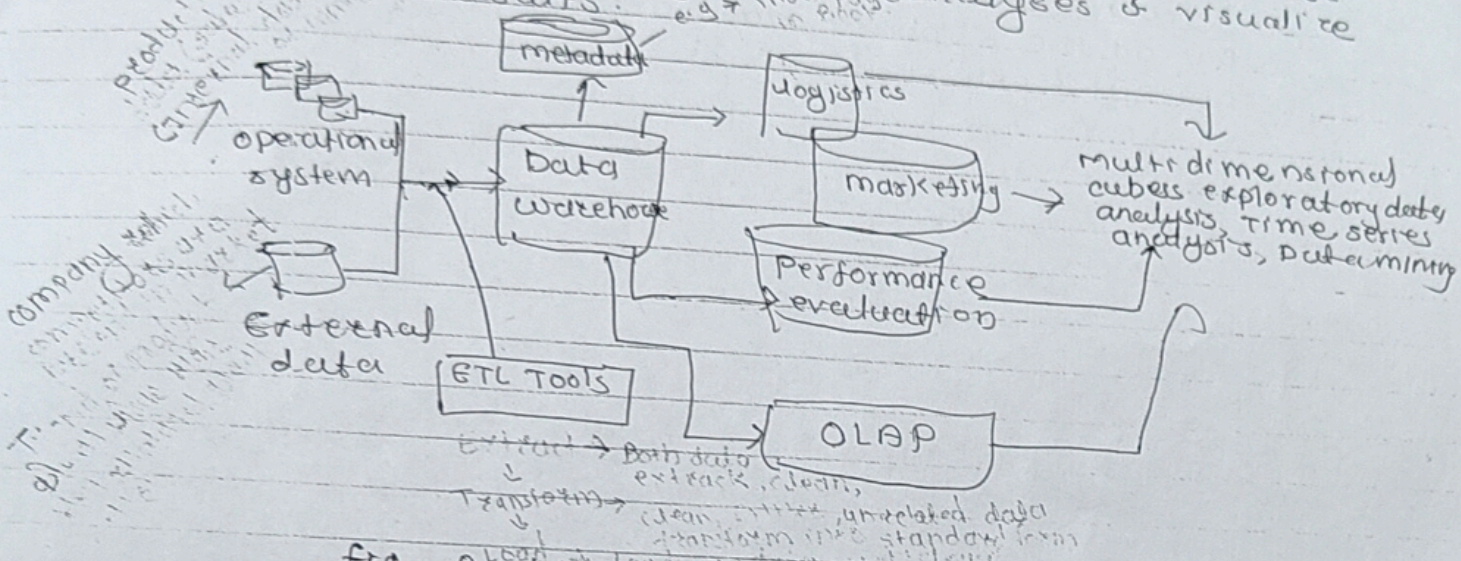


fig. Architecture & fun. of data warehouse

Three level distinction applies to the architecture shown in above fig.

1) The level of the data sources & related ETL tools that are usually installed on one or more servers.

2) The level of data warehouse & any data mart possibly available on one or more servers as well & separated from those containing the data sources. This second level also includes the metadata documenting the origin & meaning of the records stored in the data warehouse

3) The level of the analyses that increase the value of info. contained in data warehouse through query, reporting & possibly sophisticated decision support tools

The data warehouse may be implemented according to different design approaches

a) Top down → It is based on the overall design of the data warehouse & is therefore more systematic

b) Bottom up → It is based on the use of prototypes & therefore system extensions are made according to step by step.

Sulekha

ETL Tools

1) Extraction

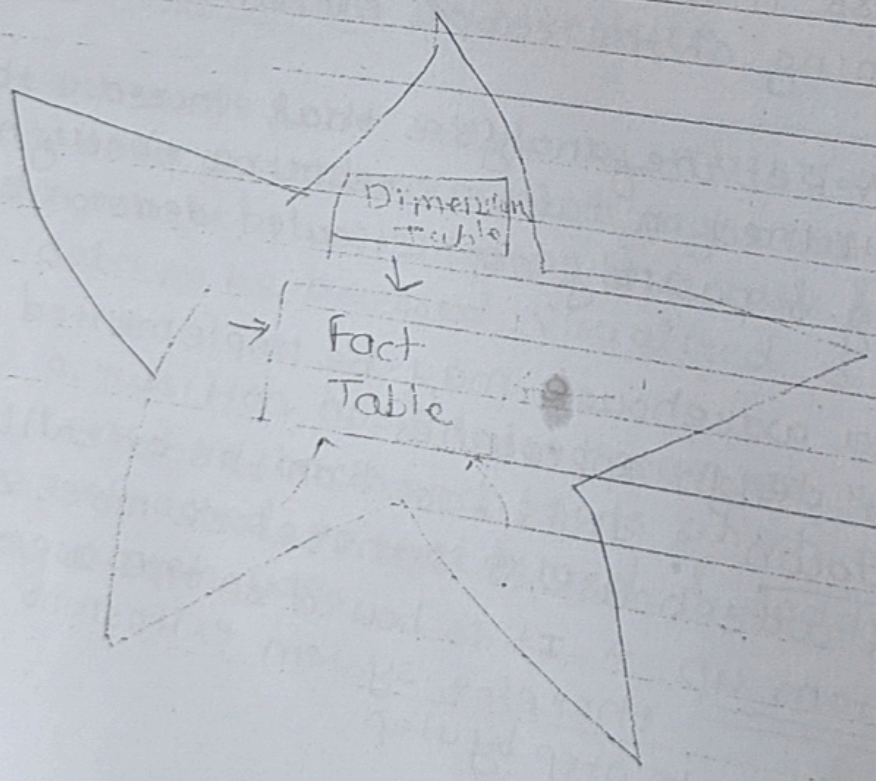
During the 1st phase, data are extracted from the orig
internal & external sources

2) Transform

After extracting the data, then clean &
filter data needs unrelated data or in
Ex: make standard format of data

3) Load :

Store & load into data warehouse



Star schema

i) Star schema is the fundamental schema among the data mart schema & it is simplest.
ii) This schema is widely used to develop or build a data warehouse & dimensional data marts.
iii) It includes one or more fact tables indexing any no. of dimensional tables.
iv) It is necessary cause of snowflake schema.
v) In star schema, business process data, that holds the quantitative data about business is distributed in fact tables & dimensions which are descriptive characteristics related to fact data.
vi) Sales price, sale quantity, distant, speed, weight, & weight measurements are few e.g of fact data in star schema.
vii) It having multiple dimensions is termed as centipede schema.

Advantages
1) simpler queries
2) simplified Business Reporting logic
3) Feeding cubes

* SQL extension for OLAP

i) OLAP requires almost invariably data aggregations & SQL does support such aggregations through it's Group-by instruction.
ii) However during OLAP, each data analysis session usually requires repeated aggregations of similar type over the same data.
iii) To facilitate the execution of OLAP queries & data aggregation, SQL introduced three extensions to the GROUPBY, CUBE & ROLLUP.

iv) The CUBE operator computes a union of GROUP BY's on every subset of the specified attribute types. Its result set represents a multidimensional cube based upon the source table.

v) The grouping() function returns 1 in case NULL value is generated during the aggregation & 0 otherwise. This distinguishes generated NULLs & possible real NULLs stemming from the data.

vi) - The ROLLUP operator computes the union on every prefix of the list of specified attribute types from most detailed up to the grand total.
- It is especially useful to generate reports containing both subtotals & totals.
- The key difference bet'n the ROLLUP & CUBE operators is that the former generates a result set showing the aggregates for hierarchy of values of the specified attribute types.

* Materialized view

i) When the results of view expression are stored in DB system they are called materialized views.
ii) SQL does not provides any standard way of defining materialized view, however some DBMS provides custom extension to use materialized views.
iii) The process of keeping the materialized views updated is known as view maintenance.

* DB system uses one of the 2 way to keep the materialize view updated.
1) Update the materialized view as soon as the relation on which it is defined is updated
2) Update the materialized view every time the view is accessed

3) Update the materialized view periodically.

- Materialized view is useful when view is accessed frequently as it saves the computation time as the result are stored in DB before hand.

'6: Data Analysis & Exploration

PAGE:
DATE:

*Data Mining
i) Data mining refers to the overall process consisting of data gathering & analysis, development of inductive learning models & adoption of practical models & methods. decision & consequent actions based on the knowledge acquired.
ii) Data mining process is based on inductive learning methods, whose main purpose is to derive general rules starting from a set of available examples consisting of past observations recorded in one or more dB.
iii) The purpose of data mining analysis is to draw some conclusions starting from a sample of past observations & to generalize these conclusions with reference to the entire population in such way that they are as accurate as possible.
iv) A further characteristic of data mining depends on the procedure for collecting past observations & inserting them into a DB. Indeed these records are usually stored for purposes that are not primarily driven by data mining analysis.
v) Data mining activities can be subdivided into two major investigation streams according to the main purpose of the analysis:
1) Interpretation
2) Prediction

i) The purpose of interpretation is to identify regular patterns in the data & to express them through rules & criteria that can be easily understood by experts in the appln domain.
ii) The rules generated must be original & non-trivial in order to actually increase the level of knowledge & understanding of the system of interest.

Subitha

i) The purpose of prediction is to anticipate the value that random variable will assume in the future or to estimate the likehood of future events.
e.g → Mobile phone provider may develop a data mining analysis to estimate for it's customers the probability of churning in favor of some competitor.

ii) In different context, retail company might predict the sales of given product during subsequent weeks.
iii) Most data mining tech. derive their predictions from the value of set of variables associated with the entities in a database.

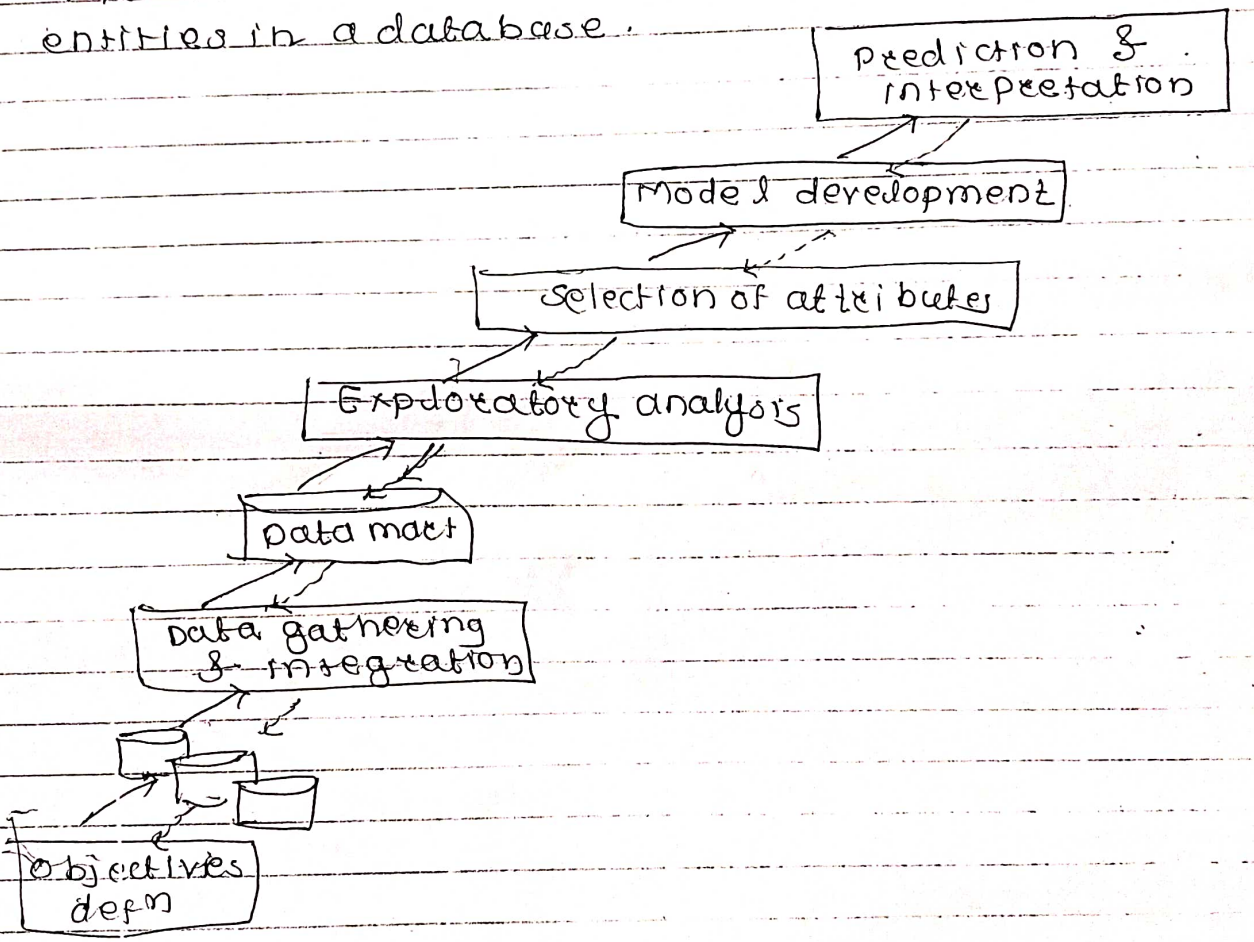


fig. Data mining process.

* Models & methods for data mining

i) There are several learning methods that are available to perform a different data mining tasks.

ii) A no. of techniques originated in the field of computer science such as classification trees or association rules & are referred to as machine learning or knowledge discovery in db.

iii) In most cases an empirically based approach tends to prevail within this class of techniques & other methods belong to multivariate statistics such as regression or Bayesian classifiers & are often parametric in nature but appear more theoretically grounded

iv) statistical learning theory which are based on solid theoretical foundations & place themselves at the crossroads of various disciplines among which probability theory, optimization theory & statistics.

* Applications of data mining

1) Relational marketing

- Identification of customer segments that are most likely to respond to targeted marketing campaigns such as cross selling & up selling
- Identification of target customer segments for retention campaigns
- Prediction of the rate of +ve responses to marketing campaigns
- Interpretation & understanding of the buying behavior of the customers.
- analysis of the products jointly purchased by customers known as market basket analysis

2) Fraud detection

Fraud may affect different industries such as telephony, insurance & banking.

Sulakshi

3) Risk evaluation
The purpose of risk analysis is to estimate the risk connected with future decisions, which often assume a dichotomous form.
e.g. using the past observations available, bank may develop a predictive model to establish if it is appropriate to grant a monetary loan or home loan based on characteristics of the applicant.

4) Text Mining
Data mining can be applied to different kinds of texts which represent unstructured data, in order to classify articles, books, documents, emails & web pages
e.g. web search engines or the automatic classifn of press releases for storing purpose.

5) Image Recognition
The treatment & classification of digital images both static & dynamic is an exciting subject both for its theoretical interest & great no. of applns it offers.

6) web mining
They may prove useful for the analysis of e-commerce sites in offering flexible & customized pages to surfers in caching most popular pages or in evaluating the effectiveness of an e-learning training course.

7) Medical diagnosis
- Learning models are an invaluable tool within the medical field for the early detection of diseases using clinical test results.
- Image analysis for diagnostic purpose is another field of investigation i.e currently burgeoning.

PAGE 5
DATE / /

Data preparation

i) Data preparation is used for proper business data analysis.
ii) The data preparation process involves collecting, cleaning & consolidating data into a file that can be further used for analysis.

why data preparation is necessary

i) To filter unstructured, inconsistent & disordered data
ii) Connecting data from real time multiple data sources
iii) For quick reporting of data.
iv) To handle data collected from scraped file like pdf doc.

Steps of data preparation

- Gather
  - Discover
    - Cleanse
      - Transform
        - develop
          - Store

① **Gather data**
This is initial process for each business, in this phase it is necessary to collect data from various sources, the source can be of any type such as from catalogs or ad hoc can be added.

2) **Discover data**
The next step is discovering the data, here it is imp. to understand the data & categorize it into different dataset. this step might take long time to filter becoz of the huge coll^n of datasets.

3.) cleaning & validating data

i) Removing unnecessary data & outliers
ii) Use the appropriate patterns for refining all the data
iii) Use the dock to protect your sensitive data.
iv) Fill the empty space for data flow.

4) Transforming the data
It defines maintaining the format or value enteres inorder to meet well define o/p & can clearly understand the wider audience.

5.) storing data
This is final step after going through all the above processes. once data is cleaned, it is ready to offer 3rd party tools such as BI tools for analysis.

Data exploration / Exploratory data analysis (EDA)

i) It provides simple set of exploration tools that bring out the basic understanding of real time data into data analytics.
ii) The outcomes of data exploration can be powerful factor in understanding the structure of data, values distributions & interrelationships
iii) It is the 1st data analytic.

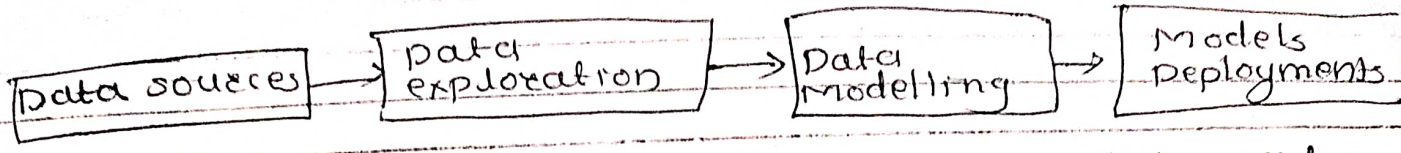
Data exploration Method

There are two formats of data exploration.
1) automatic
2) Manual.
Mostly analysts preferred automated methods such as data visualization tools becoz of their accuracy & quick response.

Manual data exploration on other hand, methods include filtering & drilling down into data in Excel spreadsheets or writing scripts to analyze raw data sets.

PAGE: 7
DATE: / /

stages of data mining



There are several tech. for analyzing data such as

1) Univariate analysis
It is the simplest form of analyzing data. Univariate means that there is only one variable in your data.

2) Bivariate analysis
It is simplest form of quantitative analysis. It includes analysis of two variables. It includes analysis of 2 variables (x, y) used for calculating the empirical rel'n beth two variables.

3) Principlae components analysis
The analysis & conversion of possibly correedated variables into smaller no. of uncorreedated var.

4) Multivariate analysis
It can be used to refer to any analysis that involve more than one variable.

The next step after data exploration is data discovery In this phase BI tools are used to inspect trends sequences & events are create visualization to present to business managers.