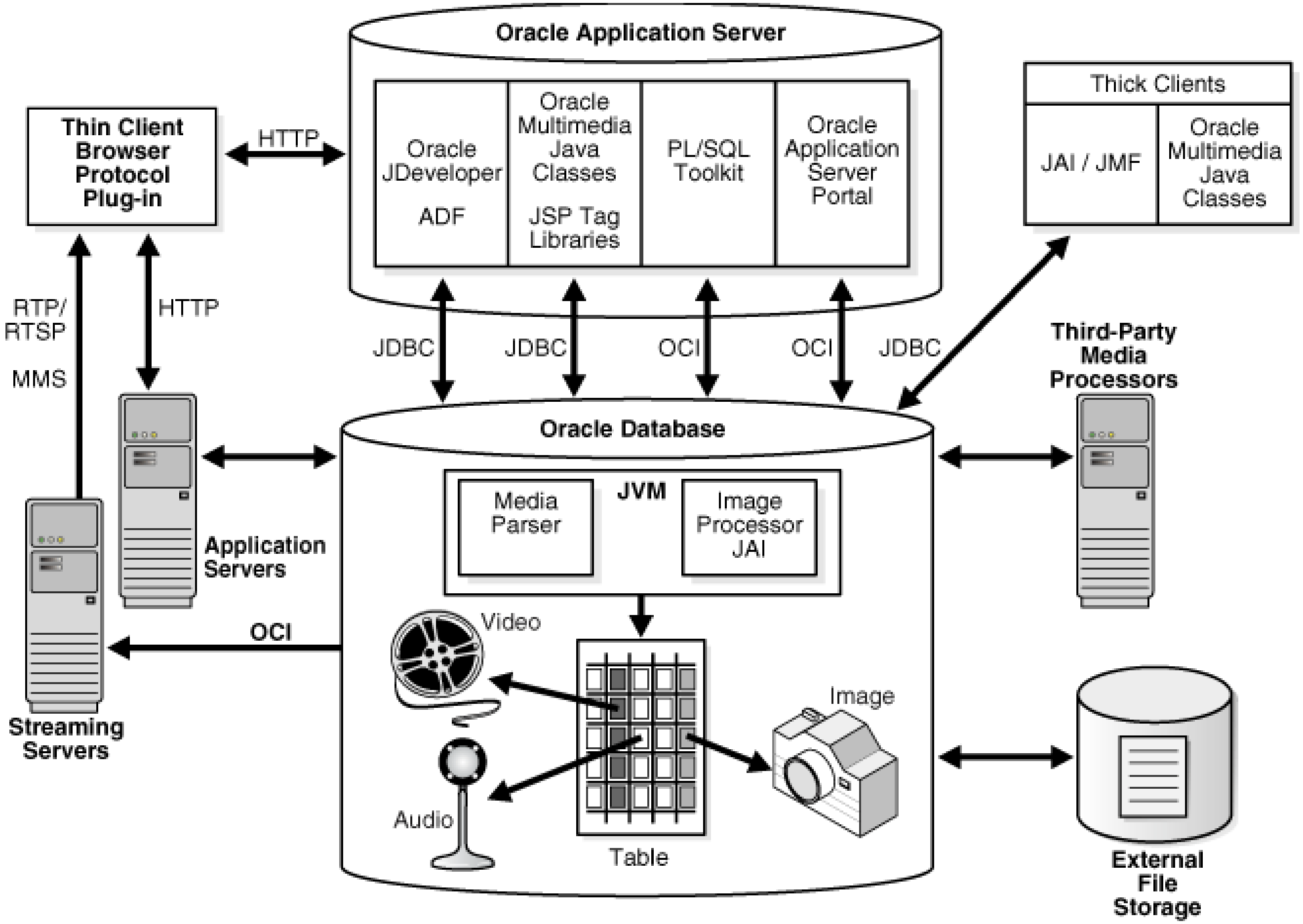
● Database architecture is a structure that facilitates the database to complete a transaction.



Tier architecture

- The rules surrounding technology are constantly changing.
- Decisions and architectures based on current technology might easily become out of date with hardware changes.
- To best understand how multimedia and unstructured data fit and can adapt to the changing technology, it's important to understand how and why we arrived at our different current architectural positions.
- In some cases we have come full circle and reinvented concepts that were in use 20 years ago.
- Only by learning from the lessons of the past can we see how to move forward to deal with this complex environment.
- In the past 20 years a variety of architectures have come about in an attempt

to satisfy some core requirements:

- Allow as many users as possible to access the system
- Ensure those users had good performance for accessing the data
- Enable those users to perform DML (insert/update/delete) safely and

securely (safely implies ability to restore data in the event of failure)

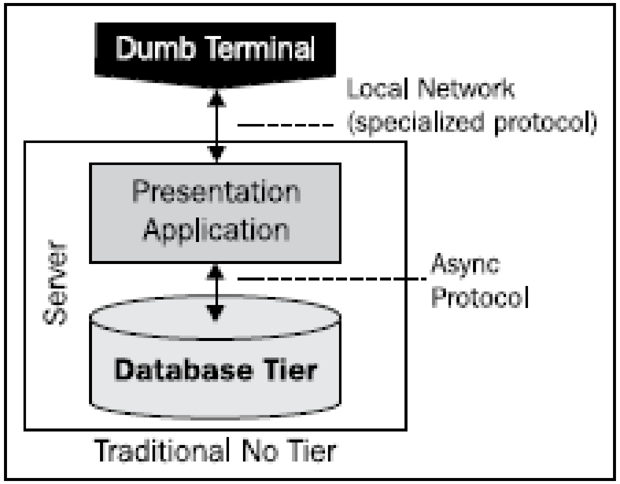
The scenario of early databases:

The goal of a database management system was to provide an environment where the requirements could be met.

The scenario of early databases:

- The first databases were not relational.
- They were heavily I/O focused as the computers did not have much memory and the idea of caching was deemed to be too expensive.
- The servers had kilobytes and then eventually, megabytes of memory. This memory was required foremost by the programs to run in them.
- The most efficient architecture was to use pointers to link the data together.
- The architecture that emerged naturally was hierarchical and a program
- would navigate the hierarchy to find rows related to each other.
- Users connected in via a dumb terminal. This was a monitor with a keyboard that could process input and output from a basic protocol and display it on the screen.
- All the processing of information, including how the screen should display it (using simple escape sequence commands), was controlled in the server.

Traditional r



- The mainframes used a block mode structure, where the user would enter a screen full of data and press the Enter key.
- After doing this the whole screen of information was sent to the server for processing.
- Other servers used asynchronous protocols, where each letter, as it was typed, was sent to the server for processing.
- This method was not as efficient as block mode because it required more server processing power to handle the data coming in.
- It did provide a friendlier interface for data entry as mistakes made could be relayed immediately back to the user.

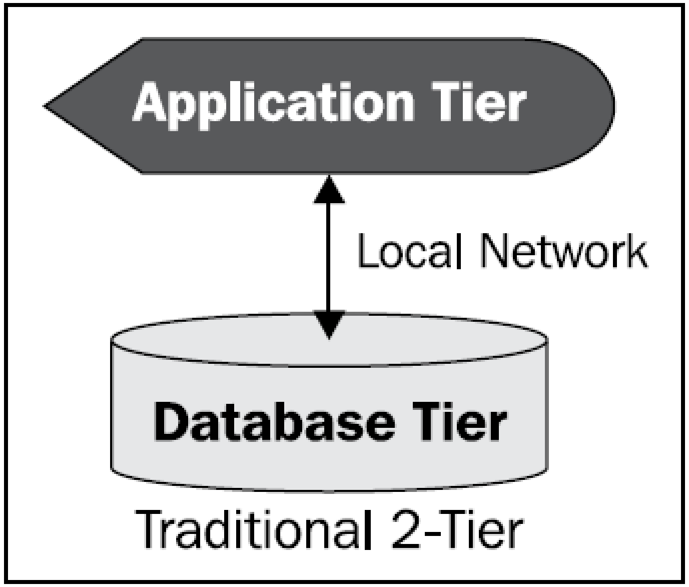
Two tier

- The GUI interface had one major drawback; it was expensive to run on the CPU.
- Some vendors experimented with running the GUI directly on the server (the Solaris operating system offered this capability), but it become obvious that this solution would not scale.
- To address this, the **two-tier architecture** was born.
- This involved using the GUI, which was running on an Apple Macintosh or Microsoft Windows or other

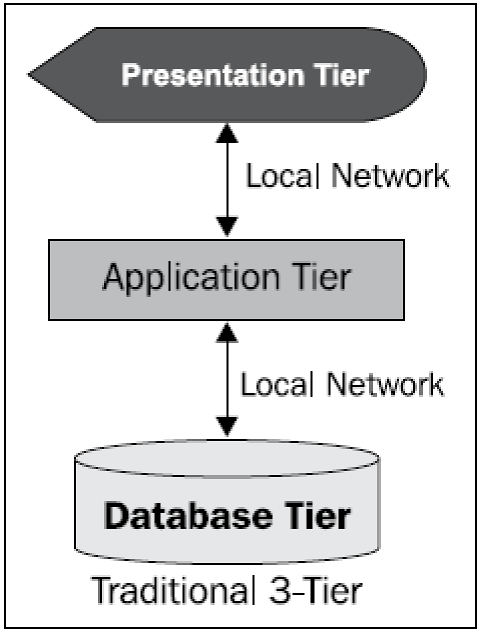
  Windows environment (Microsoft Windows wasn't

  the only GUI to run on Intel platforms) to handle

  the display processing.
- This was achieved by moving the application displayed to the computer that the user was using.
- Thus splitting the GUI presentation layer and application from the database.



Three tier



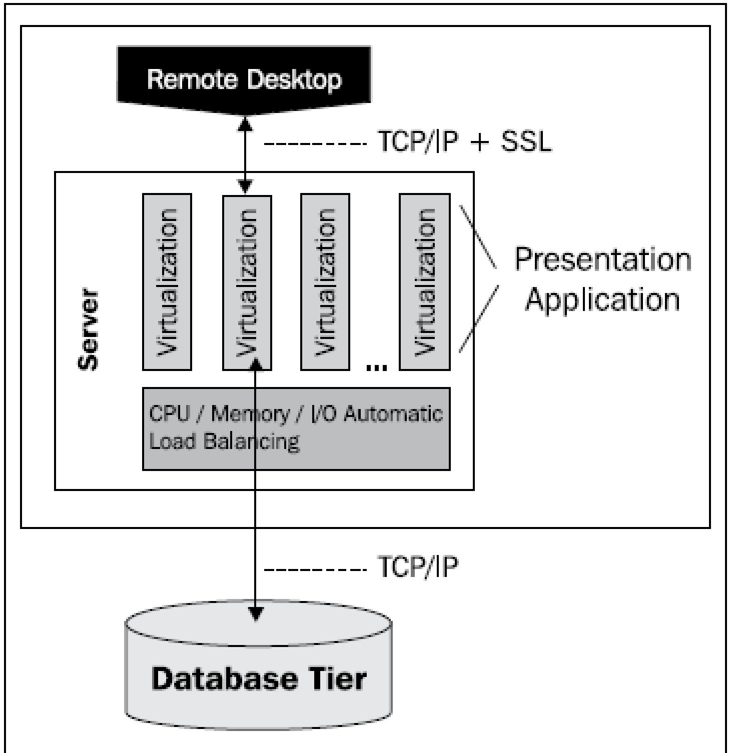
- Specialized software vendors tried to come to the rescue by offering the ability to lock down a client computer from being modified and allowing remote access to the computer to perform remote updates.

- Even then, the maintenance side proved very difficult to deal with and when the idea of a three tier architecture was pushed by vendors, it was very quickly adopted as the ideal solution to move towards because it critically addressed the maintenance issue.

- In the mid 1990s the idea of splitting the presentation layer from the application became a reality as more applications appeared in the browser.
- The web browser was not an ideal platform for data entry as the HTTP protocol was stateless making it very hard to perform transactions in it.
- The HTTP protocol could scale. The actual usage involved the exact same concepts as block mode data entry performed on mainframe computers.
- In a web browser all the data is entered on the screen, and then sent in one go to the application handling the data.

- the three tier environment has a major flaw preventing it from truly scaling.
- The flaw is the bottleneck between the application layer and the database.
- The three tier environment also is designed for relational databases.
- It is not designed for multimedia databases.
- In the architecture if the digital objects are stored in the database, then to be delivered to the customer they need to pass through the application-database network (exaggerating the bottleneck capacity issues), and from there passed to the presentation layer.

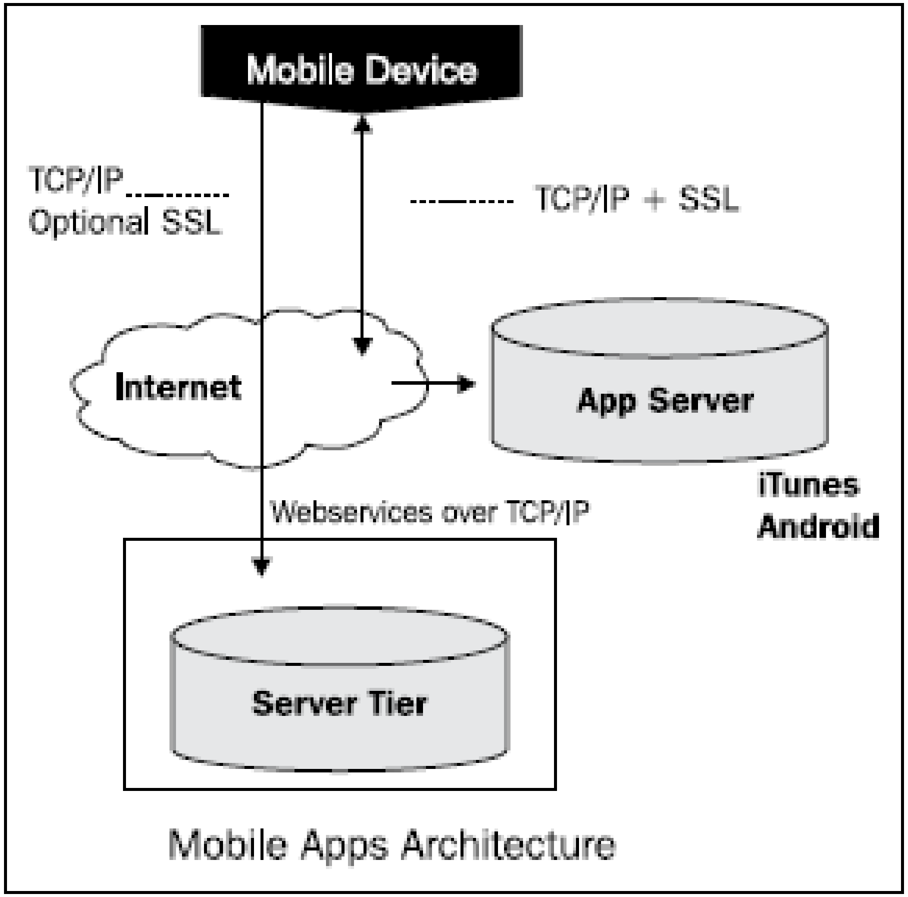
Virtualized architecture



Virtualized Architecture

- In the mid 2000s the idea of a virtualization began to appear in the marketplace.
- A virtualization was not really a new idea and the concept has existed on the IBM MVS environment since the late 1980s.
- What made this virtualization concept powerful was that it could run Windows, Linux, Solaris, and Mac environments within them.
- A virtualized environment was basically the ability to run a complete operating system within another operating system.
- If the computer server had sufficient power and memory, it could run **multiple virtualizations** (**VMs**).
- We can take the snapshot of a VM, which involves taking a view of the disk and memory and storing it.
- It then became possible to rollback to the snapshot.

Mobile applications architecture

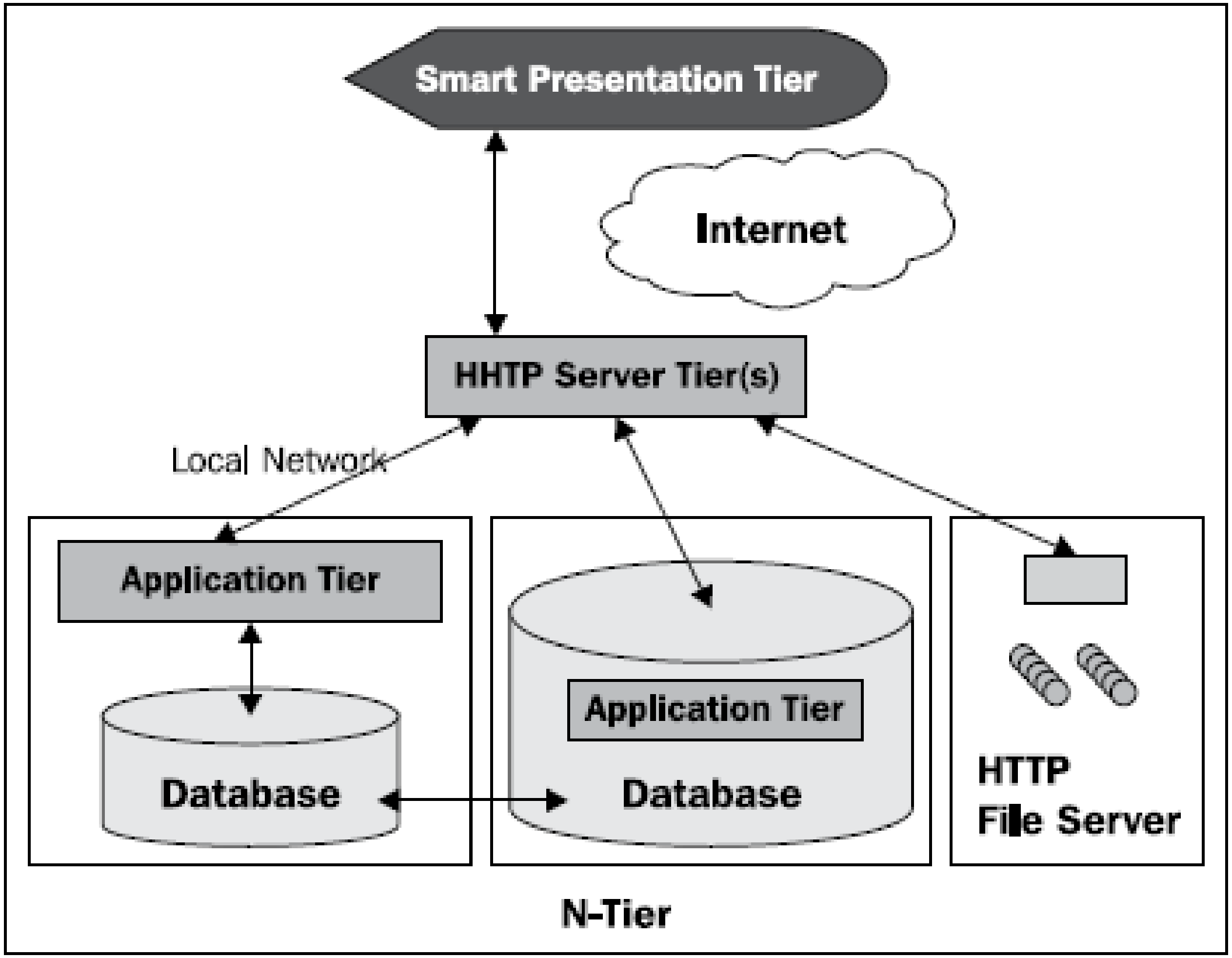


Mobile Apps Architecture

- The iPhone, iPad, Android, Samsung, and other devices have caused a disruption in the marketplace as to how the relationship between the user and the application is perceived and managed.
- These devices are simpler and on the face of it employ a variety of architectures including two tier and three tier.
- Quality control of the application is managed by having an independent and separate environment, where the user can obtain their application for the mobile device.

- Though the interface is not ideal for heavy data entry, the applications are naturally designed to be very friendly and use touch screen controls.
- The low cost combined with their simple interface has made them an ideal product for most people and are replacing the need for a laptop in a number of cases.

N-Tier


Smart Presentation Tier
Internet
HHTP Server Tier(s)
Local Network
Application Tier
Database
Application Tier
Database
HTTP
File Server
N-Tier

- For a multimedia environment the ideal solution to implement the application is based on the Web.
- This is because the web environment over the last 15 years has evolved into one which is very flexible and adaptable for dealing with the display of those objects.
- From the display of digital images to streaming video, the web browser (with sometimes plugins to improve the display) is ideal.
- This includes the display of documents.

- The browser environment though is not strong for the editing of these digital objects.
- Adobe Photoshop, Gimp, Garage Band, Office, and a whole suite of other products are available that are designed to edit each type of digital object perfectly.
- This means that currently the editing of those digital objects requires a different solution to the loading, viewing and delivery of those digital objects.

- There is no right solution for the tier architecture to manage digital objects.
- The N-Tier model moves the application and database back into the database tier.
- An HTTP server can also be located in this tier or for higher availability it can be located externally.

- Optimal performance is achieved by locating the application as close to the database as possible.
- This reduces the network bottleneck. By locating the application within the database (in Oracle this is done by using PL/SQL or Java) an ideal environment is configured where there is no overhead between the application and database.

- The N-Tier model also supports the concept of having the digital objects stored outside the environment and delivered using other methods.
- This could include a streaming server.
- The N-Tier model also supports the concept of transformation servers.
- Scalability is achieved by adding more tiers and spreading the database between them. The model also deals with the issue of the connection to the Internet becoming a bottleneck.
- A database server in the tier is moved to another network

to help balance the load.

Transaction Properties

4 basic transaction properties that all transactions should possess is called ACID test to determine transaction reliability:

1) Atomicity: the all-or-nothing property. Transaction is an undivisible unit that is either performed in its entirety or it is not performed at all.

2) Consistency: a transaction must transform a database from one consistent state to another consistent state.

3) Independence: transactions execute independently of one another.

4) Durability: the effects of successfully committed transaction should be permanently recorded in the database and must not be lost because subsequent failure.

- ACID test may appear irrelevant when we are using a single user PC database, with only one person carrying out transaction.
- For multi-user environments (in a commercialise database), large number of users accessing the same data at the same time, ACID test is relevant to ensure that a database is reliable and consistent.

- a transaction can only be achieved by locking other rows involved to stop other users changing the data while the transaction is running.
- in a replicated database there may be more than one copy of data need to be updated at the same time.
- the property of atomicity means that the whole transaction must be completed or it will have to be rolled back.

- a transaction involving multimedia data will be expected to take longer because of the size of the data involved.
- this means locks will have to be maintained for longer periods.
- we may be concerned to avoid transmission of media data across networks during transaction.

Access methods are significantly different for different media for example:

- one-dimensional objects : text and audio data are accessed in a contiguous manner (e.g. ASCII ans signal waves) which is applied to text and speech.
- two-dimensional objects: image, refers to spatial locations in two directions
- three-dimensional objects: video, refers to two spatial directions and time.

- the architecture of the database system will be strongly influenced by the underlying computer and network system.
- the alternative solutions for database architecture are:

1) *centralised database systems*: that run on a single computer system that does not interact with other computer systems.
2) *client-server systems*: networking computers allows a division of work so that some tasks relating to database structure are executed on server systems and others relating presentations to user on client systems.
3) *distributed database systems*: developed to handle geographically and administratively distributed data spread over multiple computer systems.

Multimedia server requirements

- multimedia databases are expensive to established, are often large-scale applications, supporting thousand of users and media objects.
- the design of the media server must take into account:
  1) user access behaviour
  2) bandwidth
  3) storage requirements

- centralised database supported the first multi-user databases, located in a single organisation.
- large storage and memory capabilities needed; multiple CPUs, and multiple operating systems.
- users connected to the system via terminals (e.g. PCs or mobile devices).
-parallel processing is utilised to enable faster response to queries and transactions.

- the requirements for any centralised system to support are:

a) **High volumes of information and real time requirements:**

- high volumes of information need large amounts of disk space.

- real time requirements imply bandwidth and need for policies for scheduling disk access request. This requires the application of

special multimedia database servers and data compression.

b) **Temporal requirements of video:**

- add considerably to the storage and requires compression

techniques (with acceptable quality)

Among the requirements of any multimedia server system:

1) ***minimal response time*** : server must minimise response times to live up to the expectations of the user.

2) ***fast processing capability and low data access rate*:** to ensure fast response times, client requests should be processed rapidly and data access rates should be minimised.

3) ***reliability and availability*:** large number of users and volume of data handled requires special software and hardware for fault tolerance; minimise the time sever is unavailable.

4) ***ability to sustain guaranteed number of streams***: maximum number of data streams the server can handle simultaneously.

5) ***real-time delivery:*** accurate real-time operating systems have to be developed.

Performance Issues in Specific Implementations

- normal database functions such as parsing and optimising

queries can reduce performance of multimedia systems.

- The way Oracle fetches data from disk depends on whether it

uses an index.

- With no index specified, Oracle processes the table one row at a time, called as a full table scan.
- With index, I/O processes can be reduced. A row is located by

reading **rowid** and retrieve the correct block from this address into memory.

- Tuning involves managing the allocation of memory and the I/O stream.

- One objective of database tuning is to minimize the amount of I/O processing for multimedia data. If full table scan can be avoided it will be a great benefit.

- In an Oracle system, a query will be processed by the DBMS in three steps:

1) parsing: checking if the query is a valid SQL statement.

2) planning: producing processing plan that specifies how a statement should be executed in terms of a sequence of basic table operations, such as restrict and project.

3) execution: performs the reads, locks and writes required by the plan.

- In planning step, a DBMS uses system tables to obtain details of how data is stored and accessed, and chooses one plan as the “best”. This evaluation is called to as **optimization.**
- an action path is the method used by Oracle to determine how to retrieve data.
- the path depends on the locks in use and the session environment.
- the optimizer is a set of internal routines that decides on the most efficient path to the data.