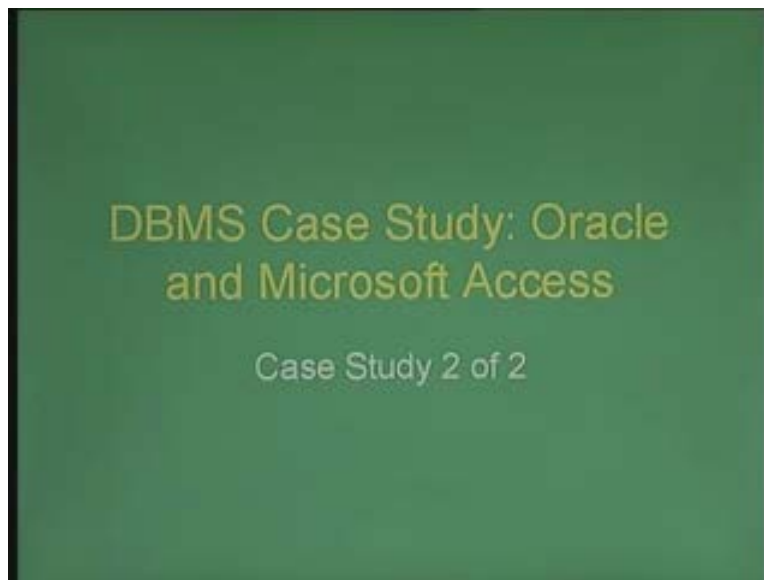


Database Management System
Dr. S. Srinath
Department of Computer Science & Engineering
Indian Institute of Technology, Madras
Lecture No. # 33

Case Study
ORACLE & Microsoft Access

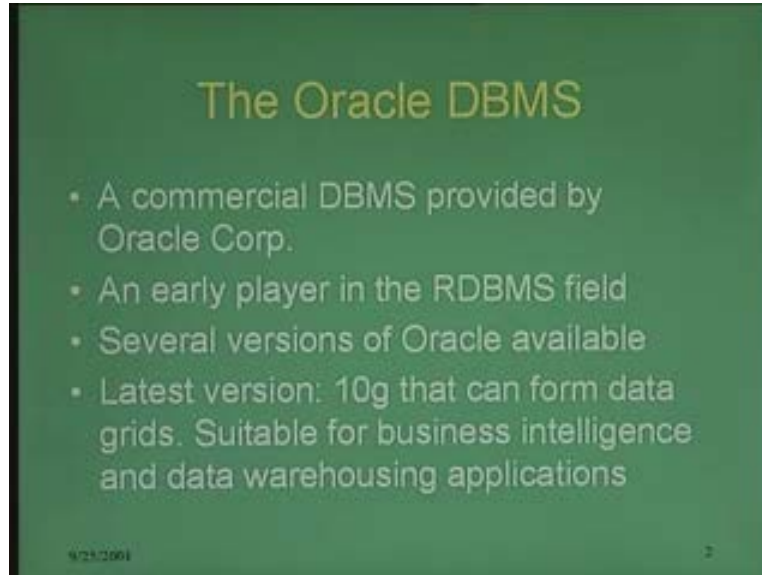
Hello and welcome. In this lecture today let us look at one more case study of a commercial DBMS this time.

(Refer Slide Time: 01:19)



In fact we are going to look at case studies, the well-known oracle DBMS and the Microsoft Access DBMS which is shipped with Microsoft office suite of a product whenever you buy them. And in the previous session, we talked about MySQL which is an open source or free software while the once that we are talking about or that we are reviewing here are both commercial software that are proprietary and you have to buy it off and which can be bought off the self and so on.

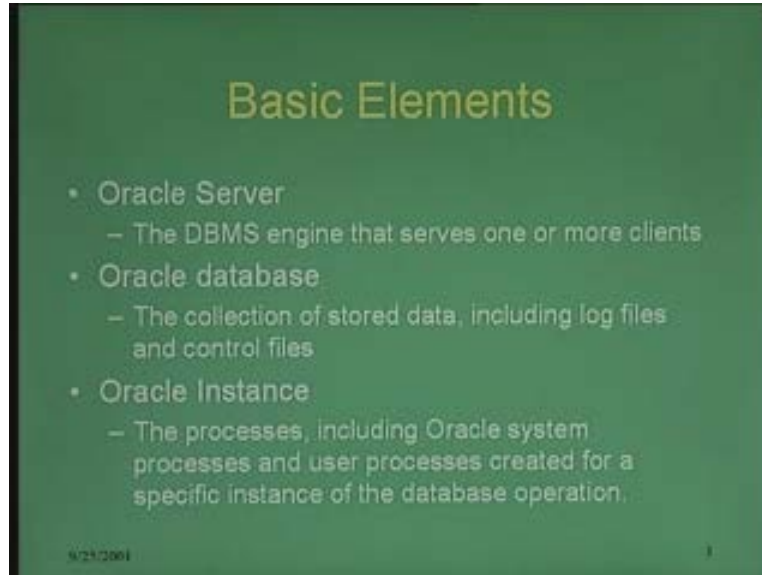
(Refer Slide Time: 02:06)



Now let us have a look at the main structures of this, firstly the oracle DBMS. In this lecture we are not looking at any specific version of oracle. Oracle is, the DBMS oracle is manufactured by Oracle Corporation headed by of course Larry Wall and who has been a pretty early player in the whole field of commercial database management systems. And there have been several versions of oracle that have been evolved over the years and the latest version at least as of today is oracle 10g, the g here stands for data grids where the database can sense that it is part of a larger data grid and operate accordingly. That is a grid is a collection of different databases which could be distributed widely probably across the world and while functioning as a single data source or a database. And it's been touted or it's been kind of targeted towards the data warehousing and business intelligence kinds of applications.

What are the basic elements of or the building blocks of an oracle database? When we are talking about oracle databases, there are some terminologies that we need to be aware of. Many of these terminologies as we noted earlier, oracle has been an early player in the field of commercial DBMS systems and while terminologies have been invented and evolved over time in the academic and research community for database management systems. Similar kinds of terminologies and evolutions have happened in the commercial field as well.

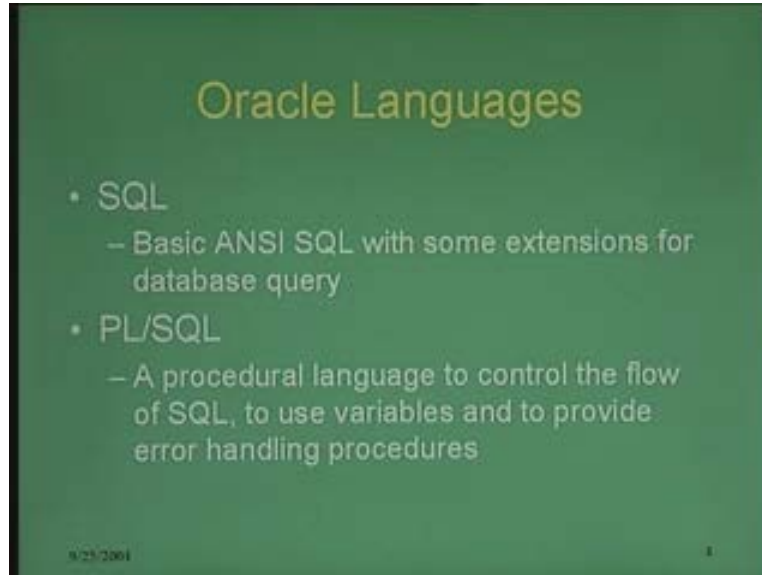
(Refer Slide Time: 03:18)



So sometimes it may so happen that the terminology used, that we have used for example in this class might differ from the terminology used in oracle or let's say db 2 or some other commercial database engine. And so it makes a lot of sense to clear what each of these terminology mean or what kinds of terminology mismatches can we have between, what we have been using and what oracle for example uses. When we talk about an oracle server in the database management system, the oracle server is actually the DBMS server that we know very well. That is it's the DBMS engine that serves one or more clients. And when you talk about the oracle database, it is more than just the set of data files that form the database but oracle considers its database as a collection of all its stored data including log files and control files and other such auxiliary files that make up the database.

And usually in the academic community when you say database, we usually talk about only the data files and the index files, when we talk about for example the storage structure. Then when we say an oracle instance, this refers to the set of processes that is the set of all processes including oracle system processes which with oracle server runs and user processes created for a specific instance of the database operation. What kinds of languages does oracle support? Oracle of course supports the ANSI SQL standard with some extensions of itself that was specific to oracle.

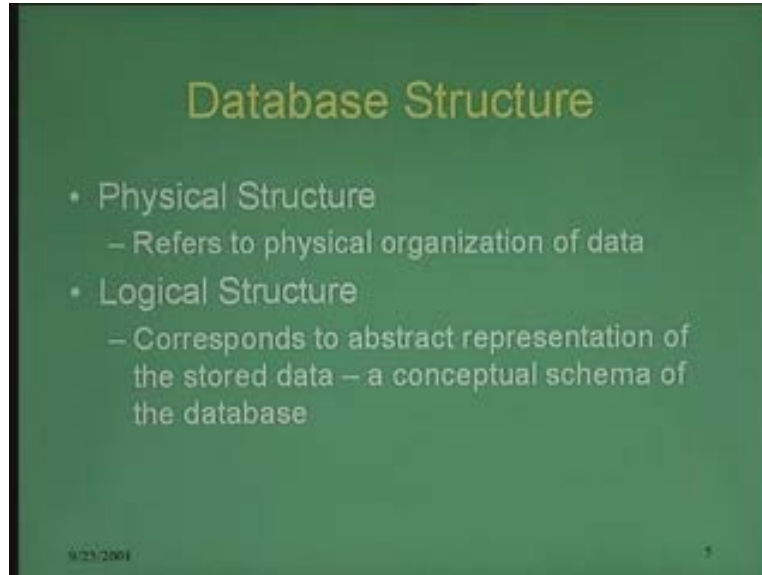
(Refer Slide Time: 05:43)



And in addition to the well-known ANSI SQL, oracle has its own query language called PL/SQL. PL/SQL is a procedural query language, on top of SQL that is say for example SQL language is in some sense a pure query language in the sense that it does not have other programmatic constructs like say using variables or well it has certain, you can use certain kinds of variables by aliasing and so on. But generally it doesn't have the generic structure of any procedural language like C or Perl or some other kinds of language.

PL/SQL on the other hand is a complete procedural language where you can actually define variables, you can define control flow, if then else constructs and looping constructs and so on and so forth and embed SQL statement within these procedural constructs. When we talk about the database structure in oracle, we usually refer to two different kinds of structures what are called as the physical structure and the logical structure of the DBMS. The physical structure refers to the storage structure that is a physical organization of the data on the database. While the logical structure corresponds to what is a conceptual schema of the database that oracle is maintaining, I mean logical structure here is the logical structure of the way oracle manages data not that of a particular database itself.

(Refer Slide Time: 07:13)



So the way oracle manages data itself can be treated as a database which comprises of several other databases and which comprises of a larger logical structure and update rules and associations and so on and so forth.

Let us look at the physical structure in a little bit more detail. We shall also briefly look at the logical structure as well. When it comes to the physical structure of the database, the way oracle stores data on disk, it basically stores data in several different files on disk and very unlike let's say MySQL oracle uses its own buffer management policies for writing data into files. What is meant by buffer management policies for writing data? DBMS like MySQL uses for example, if you are taking a Unix based implementation or Linux based implementation of and we are comparing MySQL and oracle, MySQL uses what are called as high level system calls for writing data into disk. That means high level system call essentially says, gives a block of data to the operating system kernel and lets the kernel take care of writing the data on to the disk by itself.

Now what the kernel actually does is in order to speed up the process, it doesn't immediately write your data block onto disk instead it keeps it in its own memory, in its own cache that is in main memory and then flushes it on to the disk at some point in time. This cache is called the buffer cache. We saw buffer cache and how buffer caches are come into play or affect DBMS operations when we are talking about database recovery processes.

(Refer Slide Time: 08:08)



So when we write something onto disk and we use a high level system call, there is no guarantee that **whatever you have written on to**, whatever you have written has actually been recorded on to disk. Now if there is suddenly a system failure then even though our write has returned us a successful operation and we said we have committed some transaction, it may not actually have been reflected on to disk.

On the other hand oracle uses low level system calls that means oracle manages its own buffer cache and oracle decides for itself when it has to flush the buffer cache and when it has to write to files and so on. However it still uses files that are visible from the file system interface of the operating system. So let us look at the files that oracle uses. So when we are looking at the files here that oracle uses, we have to keep in mind that the way that files are updated in oracle is very different from the way that files are updated in say MySQL. So these are some of the files (Refer Slide Time: 10:45) that oracle uses for maintaining its databases.

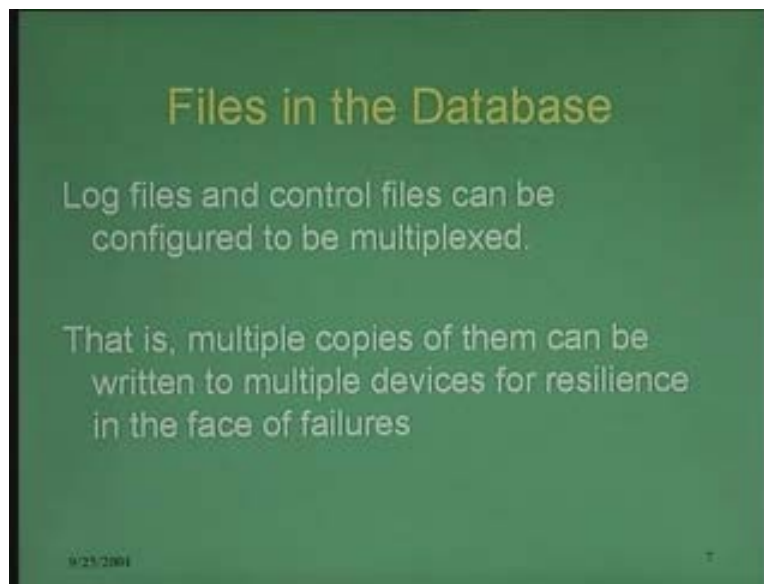
Firstly the data files of course. The data files are the files that contain the actual data and in addition to the data files, a database is associated with a set of redo log files. Remember what is a redo log file, redo log files are the set of log files that are used for redo based recovery. We looked at a set of log based recovery mechanisms where you first write data on to the log file and then after flushing the log file, you write data on to the actual database.

Now whenever there is a system crash, you just have to redo based starting from the last check point in your log file all the operations that have been performed until the system crash happens. So oracle does something like that, so it maintains the set of redo log files as part of database itself. Then there are a set of control files which contain different kinds of control information like the database name or the different file names and the locations of this different files and so on. These control files are also used in the process

of recovery, sometimes they are useful in the process of recovery and in addition to control files there are other files like what are called as trace files and alert logs. Trace files are essentially files that track certain background processes that oracle runs.

Oracle basically runs several different background processes that for example, the server is a background processes and there are several system monitors that, oracle runs to monitor events and call triggers or enable triggers and so on and so forth. Now trace files essentially track this background files and logs them on to files and alert logs maintains a log of the major events that have happened from these background processes and from anywhere else in the system. The log files and control files as you can see are the files that are essential for the process of database recovery. Therefore oracle also supports mechanism by which log files and control files can be multiplexed onto different devices.

(Refer Slide Time: 13:15)



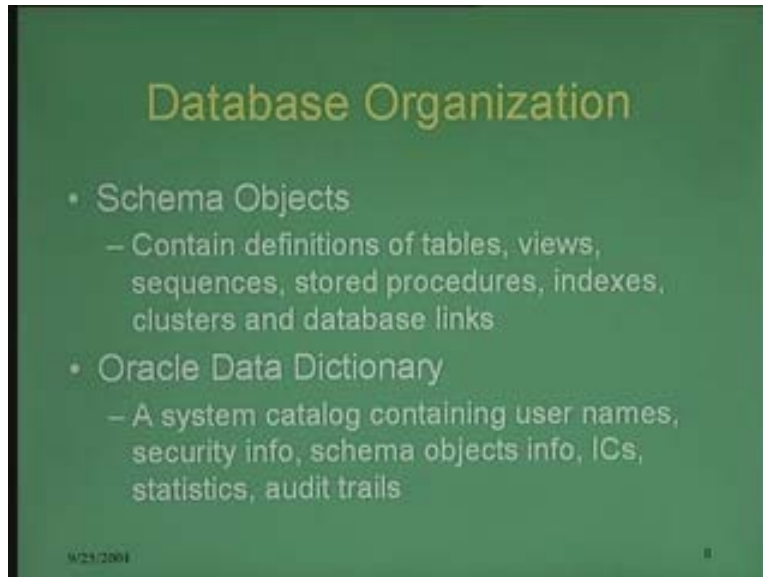
So when you write a log file, you can configure oracle so that it actually writes it in parallel to multiple devices. The advantages of this is even if there is, in a system crash even if there is let us say a media failure and the log file itself is lost, we can use these backup copies of the log files from actual devices, from different devices to recover back. So the resilience of the recovery process is increased as part of this.

So let us look at the logical database organization now. That is how is database organized in oracle, what are the building blocks that make up a database in a logical sense and what are the different roles of these different building blocks and so on.

When we are talking about an oracle database, in the physical sense there are only files, everything is a file and file is managed by low level system call operation. But from a logical sense, each of these files means certain different things and they are looked at as objects rather than files. Now what are the different kinds of objects that an oracle database contains, there are what are called as schema objects.

A schema object contains definitions of some relevant entity in the database. For example it might contain definitions of tables, it can contain definitions of views or sequences or stored procedures, indexes, clusters or links to other database and so on.

(Refer Slide Time: 14:09)



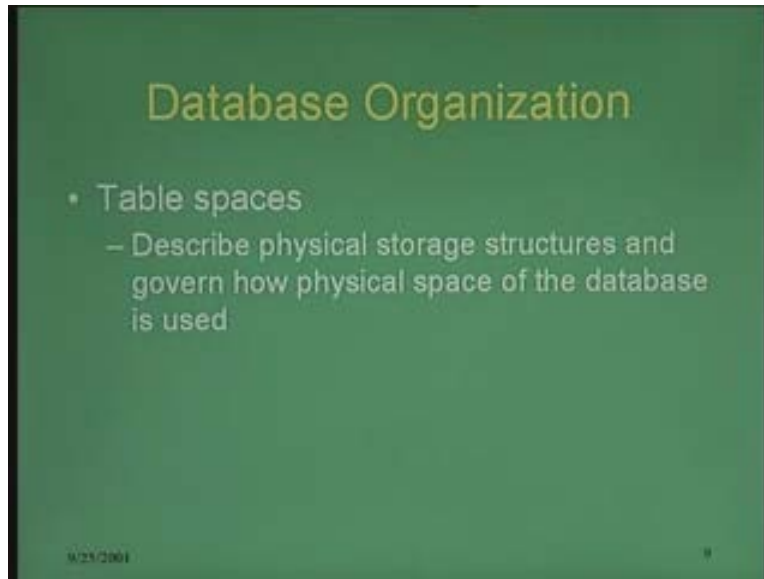
So everything is treated as an object here and the object is essentially in a sense serialized and stored on to disk. So an object for using the object orientation methodology comes with its own set of procedures or methods which handles or which provides a set of services based on the objects and definitions. So when we say a schema object represents a table, it also provides methods by which we can access or perform different kinds of table accesses, access a table row wise, access or perform any perform any index search on table and so on. And in addition to the schema object, there is what is called as a data dictionary which is essentially the system catalog and it maintains different kinds of cataloging information for oracle.

So it contains information like user names and security information that is access privileges and what kinds of users or what kinds of privileges on which schema object and information about schema objects themselves, the modification date and the creation date and any other relevant information for any given schema object. Then integrity constraints which in turn trigger all the stored procedures or which in turn trigger certain schema objects which are stored procedures and then certain kinds of statistics essentially for example that help in performing good query execution plans. We saw the role of cataloging information when we were looking into how do we manage or how do we process a query and optimize a query.

Cataloging information contain certain, I mean the statistics here in the cataloging information contain certain estimates on let us say how many rows are there in a table, how many number of distinct values of are there for a particular column and which rows is accessed how many number of times or whatever.

All of these information go into performing a or forming a good query execution plan while making the query. And then there are audit trails that is auditing information about the different aspects of the database.

(Refer Slide Time: 17:46)



The next aspect of the logical schema or logical organization of the oracle databases is what is termed as the table space. A table space is again an important concept in oracle which basically describes the physical storage structures of a given set of tables in a database and it basically governs how the physical space of a given database is used. We look into how the tables space is organized, they basically contain different segments and which in turn manages the pages or the disk blocks or that make up a particular database.

So earlier we mentioned the concept of an oracle instance. An oracle instance like we said has a specific definition that is in some sense it is a snapshot of the entire oracle system. That is it is the set comprising of all processes that is oracle processes and user processes and all of which comprises one instance of a servers operation.

(Refer Slide Time: 18:41)



Now an instance itself can be logically seen as comprising of several different parts. So an instance is logically divided into what is called as a system global area. What is the system global area? The system global area is essentially the area which in some sense to give an analogy to operating systems, it is the memory area that is used by a kernel for example in contrast to that of areas used by user processes. That is all the resources and memory locations that are used by the kernel is a system wide and globally relevant.

So the system global area is in some sense the kernel of the oracle system. It contains the database buffer cache which we saw is maintained by oracle itself in order to flush buffers on to disk and in order to basically control when buffers are flushed on to disk. Then there is the redo log buffer which is the buffer cache for the redo log. So whenever a redo log is returned, it is actually first return into the redo log buffer and which in turn is flushed on to the disk at periodic intervals. Then there is shared pool of other resources which we will see. Then in addition to the system global area, when we talk about an oracle instance we also talk about the user processes.

An oracle instance again is like I said is something like a snapshot of the entire system. So you might also take an analogy to operating systems where we say an instance of the operating system is the set of all processes comprising of the kernel processes and all user processes at any given instance of time. In addition to user processes there is what is called as a program global area in contrast to the system global area.

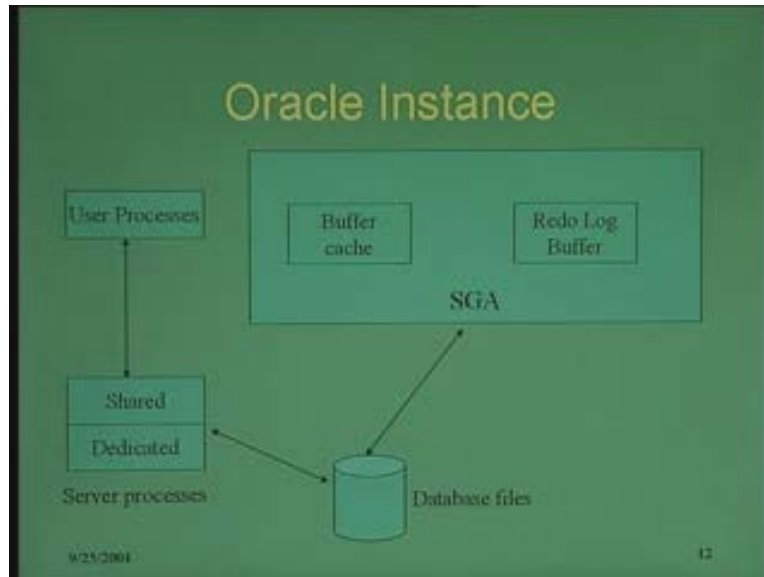
(Refer Slide Time: 21:23)



A program global area is a memory buffer that contains a data and control information for server processes. That is where, this is the global area again there is no specific analogy as such to operating systems here but you can think of it as a global area which is code specific that is for the data and control information for the global processes that are happening in the database. In addition there are certain system processes itself which are the oracle processes. So oracle itself runs several processes in the background which comprises the server instance or the oracle instance at any given point in time.

So oracle processes may comprise of the server process itself or the server process is depending upon whether it is single threaded server or its concurrent server where it can actually serve several different clients concurrently. And then there are other background processes like audit trails and system monitors and so on which belong to oracle. So here is a schematic diagram of the system global area. Of course this of an oracle instance rather and of course this diagram is in some sense leaves out a lot of things that is for example the program global area is not explicitly shown here and the system processes is not explicitly shown but generally all of this form the oracle instance.

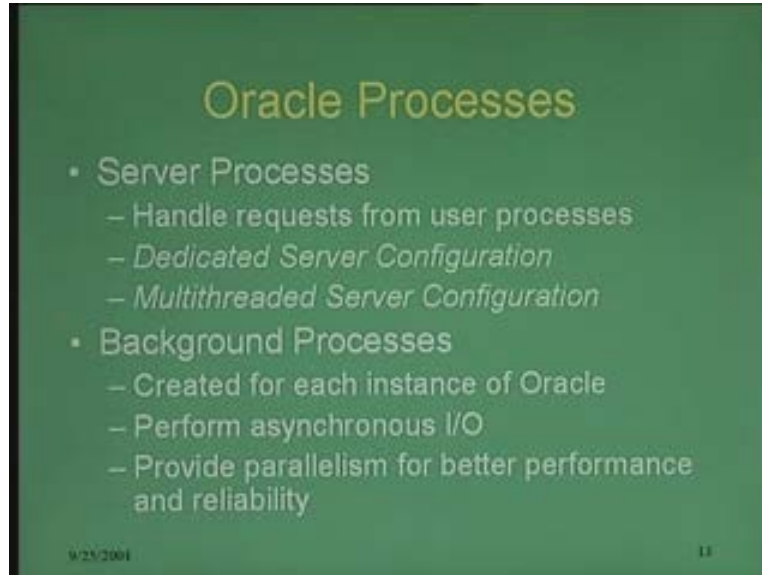
(Refer Slide Time: 22:54)



So the system global area which comprises of the buffer cache and the redo log buffer they interact directly with the database files and user processes and of course the server processes they interact with the database actually through the SGA or through this system global area. So there are user processes which in turn which interact with the oracle server processes and oracle processes which could be server processes and possibly other processes. And these in turn rather than interacting directly with the database file, to be more correct this diagram should be, the arrow here should come through this thing. That is it interacts with the database or with a disk through the system global area.

On the other hand if we are talking about database files, this logical arrow is correct which says that logically the server processes deal with the database files. So let us look at oracle processes again in little bit more detail. Oracle processes can be categorized into different forms namely server processes. Server processes are those which actually handle request from user processes or oracle client or other application programs and so on which send SQL queries to the server processes.

(Refer Slide Time: 24:24)

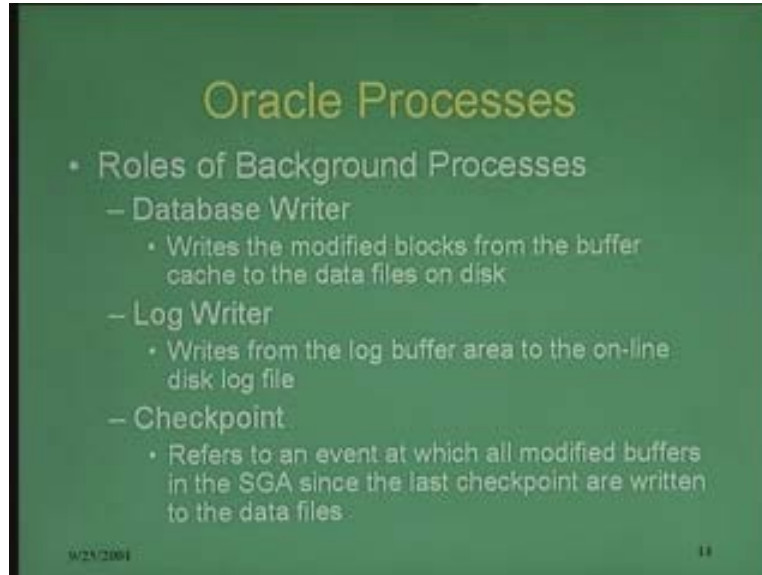


There are two kinds of ways in which you can configure your oracle server, you can either configure it as a dedicated server or as a multithreaded server. And dedicated server essentially is a server that is dedicated to a particular client, so it can take handle one client at a time whereas a multithreaded server is concurrent in the sense that it can handle many client connections at the same time by spawning different threads or processes if your kernel doesn't support threads and so on.

In addition to server processes there are what are called as background processes. And there are different functionalities for each of these background processes and they are created for each instance of oracle and one of the main functionalities of these background processes are to perform these asynchronous I/O. That is there this process is that mediate between the buffer cache in the system global area and the physical disk. So these processes essentially perform what are called as read ahead or delayed write operation. So when you write on to a disk, your data goes into the buffer cache and in some point in time the background processes wakes up and asynchronous late at some point in time the background processes wakes up and then flushes all the buffer cache onto disk.

Similarly when you give a read command, **your data is read** the relevant data blocks are read from the disk but at the same time a background process is awakened which in turn reads several other data blocks also from the disk into the buffer cache, again for performance considerations. So the main functionality or the main requirement of the background process is to provide parallelism for better performance and reliability.

(Refer Slide Time: 27:25)

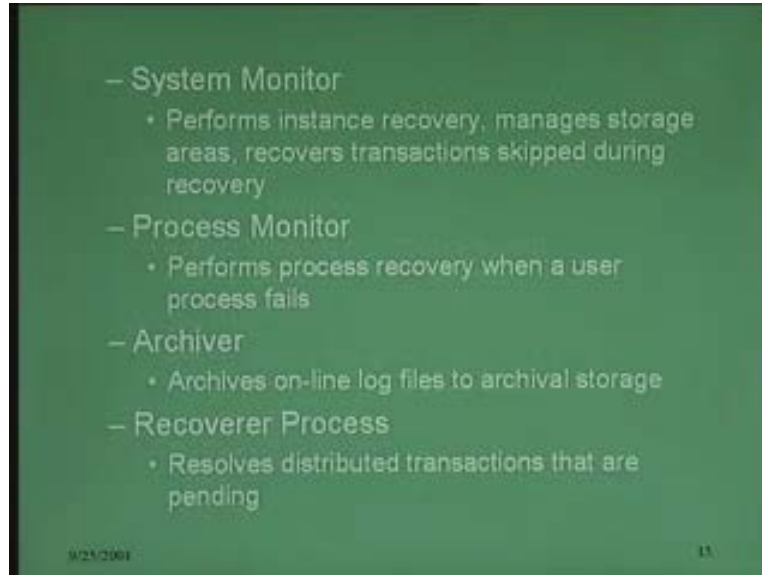


So there are different kinds of background processes which take on different roles that perform these asynchronous operations and usually it is good to distinguish between these kinds of background process. For example the database writer process, so this is the process that writes the buffer cache of the data blocks from the buffer cache to the data files on to the disk. Then there is the log writer background process which mediates between the log or the redo log buffer cache and the log files on the disk.

So data processes are managed separately from the log flushing process and then there is the check point background process. A check point as we saw in sessions on recovery based techniques, this basically refers to a event in which all modified buffers are returned to the data files. That is any log data or any data about transactions that have been committed before the check points can now be safely discarded. So we saw what are check points. So this check points as we saw in the session on database recovery have to be taken at some points in time and there is a tradeoff between the speed or the overhead introduced by the check pointing process versus how much background data or how much historical data that you need to store in order to perform a recovery in the phase of failures.

So this check point background process is a process that runs once in a while to perform this check pointing operation. And as we saw in our sessions on database recovery, check pointing is a costly operation because you need to bring the database onto a quiescent state when you are performing a check point. But there are other techniques by which, there are other check pointing techniques which can obviate this problem so that you don't have to stop all database operation when taking a check point.

(Refer Slide Time: 28:55)

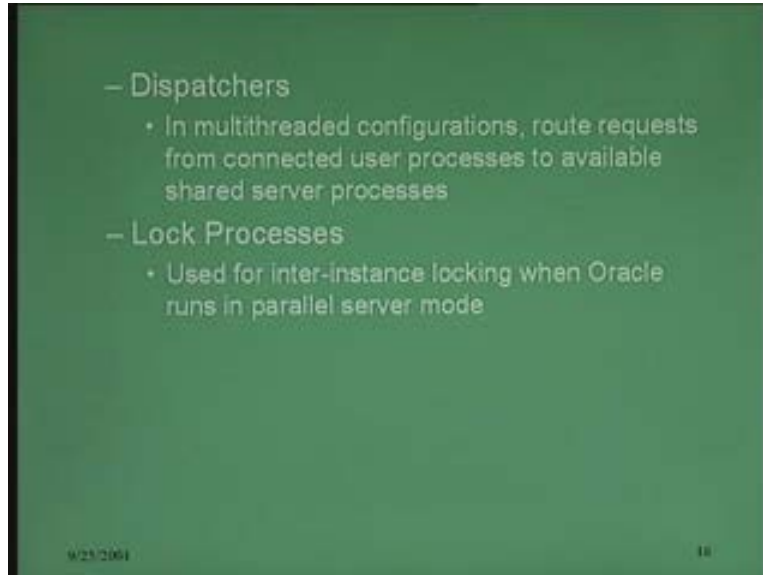


Then there are other process like the system monitor which performs recovery of an instance that is it identifies that when the system comes up, it identifies that the system had crashed and performs redo based log recovery then it manages storage areas and it also manages any kinds of recovers transactions that were skipped during the recovery process. Then there is the process monitor background process, it performs a recovery operations whenever a user process fails.

Note that failure of a user process can also leave the database in an inconsistent state especially when it is in the midst of the transaction and especially even more when transaction data return directly onto disk. So if the user process fails to complete the transaction then the database is still in the inconsistent state and then you need to recover but then this is different from a system monitor in the sense that the entire system has not crashed, so other transactions are running, so this recovery process has to be handled separately.

There is also a process called the archival process which archives the online log files on to archival storage especially, essentially for back up purposes. We also saw some mechanisms of archival when we are talking about the database recovery techniques. Then there are recoverer processes which essentially are useful when oracle is used in a distributed database setting, we have not as yet looked into issues pertaining to the distributed databases. But there are several issues when database is distributed across several different clusters especially to detect that a particular transaction that is spanning across different machines has actually crashed or it is spending or something of that sort.

(Refer Slide Time: 31:38)



Then there are other processes like dispatchers and lock processes where dispatchers in multithreaded configurations, they route request from user processes to the appropriate or the available server processes and then there are locking process which are in some sense again monitors that monitors the locks across different instances when oracle runs in parallel server mode.

(Refer Slide Time: 31:47)

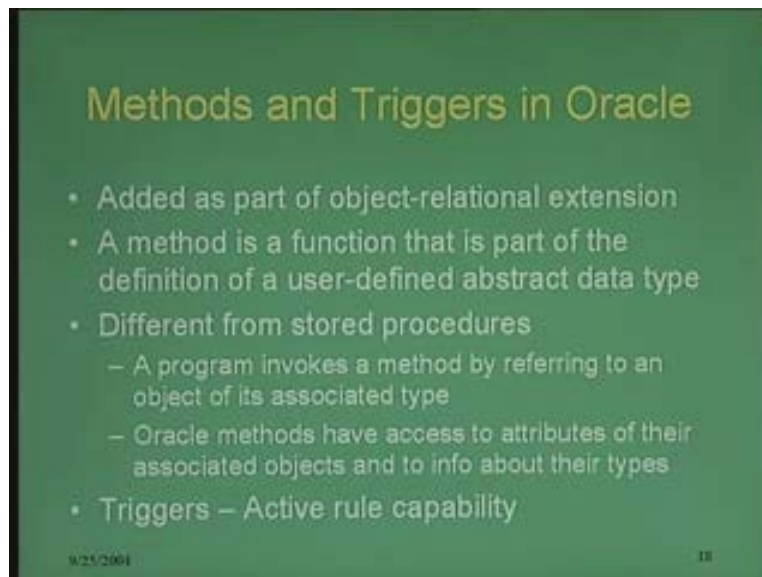


Now let us look at SQL in oracle. What kinds of SQL statements are handled? As we noted earlier oracle supports ANSI SQL. In addition to ANSI SQL, it has its own extension to SQL namely the PL/SQL which helps the user to embed SQL statement into

procedural language but as far as the support for SQL itself is concerned, oracle supports all kinds of standards data definition statements and data manipulation statements and it supports transactional semantics, full acid semantics then other kinds of constructs like session control semantics and even embedded SQL statements.

In addition to support for SQL, oracle comes with an elaborate mechanism for methods and triggers and these utilities were added to oracle as part of its object relational extension. So oracle essentially is an object relational database where we have not again as yet seen in to what constitutes an object relational database but essentially a database that uses object oriented semantics like method calls, triggers and triggers are also some kinds of method calls and inheritances, associations and so on. So a method is actually a function that is part of the definition of a user defined data type and its slightly different from the stored procedure in the sense that program essentially that's a user program invokes the method by referring to an object of its associated type.

(Refer Slide Time: 34:03)



But stored procedures are actually called from SQL statements independent of any particular object as such. It's only the database whose context is necessary for stored procedures. And oracle methods have access to attributes of their associated objects and information about their types. So a method for example, an oracle method for example would be a method in one of the schema object. Let us say schema object represents a table. A method would be some code which says how do we access the next row in a table or how do we get the primary key field of a table and so on. So get primary key or get next row or query or something of that sort which abstracts away the implementation or the storage structure of the table from the database engine as well.

So this is different slightly different from the stored procedure concept because stored procedures are visible to user programs or SQL constructs while methods are essentially, I mean the methods are visible to user programs only in an object relational context and

not in a pure relational context. And triggers are those methods which have active rule capability that is which are called automatically in response to events and conditions.

Let us have some more deeper look into the storage organization of the oracle database. We noted earlier that storage is managed by what is called as the table space. Now table space is the space that manages the physical organization or physical allocation of memory or disk blocks for a given database.

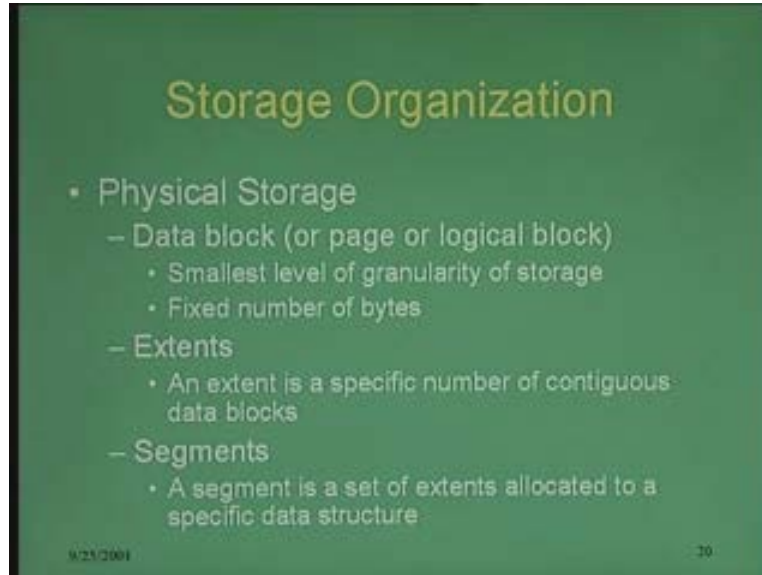
(Refer Slide Time: 35:33)



So a database is divided into several different table spaces and among these we can distinguish between what are called as system table space and the user table spaces. A system table space maintains essentially all data that are visible only to the oracle system while the user table space maintains data that are accessible by user programs. And data files are stored or managed by the table space itself. Every oracle database contains a unique system table space and this is labeled as SYSTEM in caps here. And the data dictionary objects and other system wide or information are stored in this system table space or managed by this system table space.

Now the physical storage itself I mean how does the table space manages the physical storage. The physical storage itself is divided into three different kinds of storage what are called as data block or and extents and segments. A data block corresponds to what is called as a page in or data page in several other database systems. It is a smallest level of granularity in which data is stored.

(Refer Slide Time: 37:22)



Data blocks usually have a one to one correspondence with a disk blocks or the granularity with which your disk accepts data. Most of the or at least all of the disks accept data in terms of one sector size of the disk but there are some disks with enhanced capability that can actually read and write multiple contiguous sectors at the same time.

Now this multiple contiguous sector form one data block. And then extent, the term extents is used to refer to a specific number of contiguous data blocks, so where data block is the minimum unit of storage. Now a set of segments that are allocated to a specific data structure like a table or an index or something like that is called as a segment. So data blocks again each of these, let's look at each of these different elements in a little bit more detail.

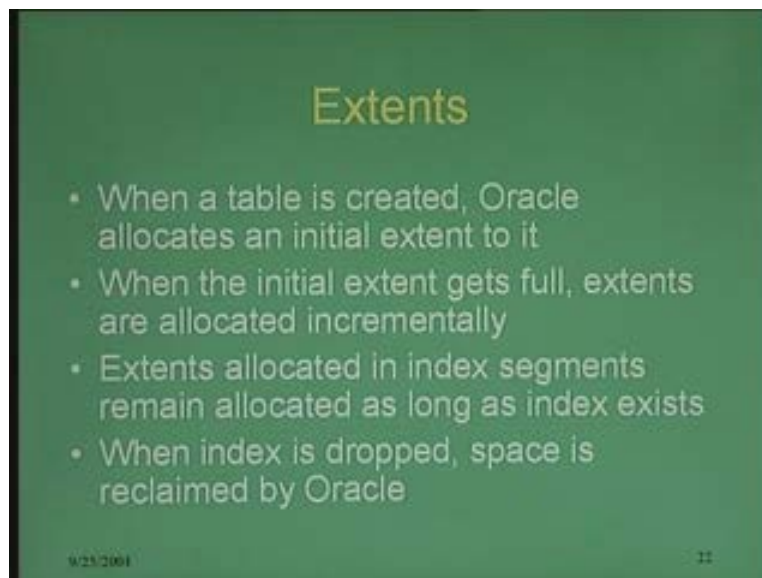
A data block whenever a particular data block is stored, it contains the following information. It contains the some header and table directory information which talks about which table it belongs to and pointers to other tables and so on. And there is a row directory information which talks about which are the specific rows that are stored in this data block and the data that pertains to a row and any free space information, if there is any free space information, if there is any free space left out in these data blocks.

(Refer Slide Time: 38:34)



And an extent as we said earlier is a set of contiguous data blocks. So when whenever you create a table, oracle allocates an extent to it and an extent is extended in a sense that is an extent is, more extents are added to the table as and when extents become full.

(Refer Slide Time: 39:10)



And these extents are managed by segment that is we saw that a segment is a set of different extents. And extents can be allocated for any given data structure like tables or indexes and so on.

So if I allocate an extent to an index, the extent remains allocated as long as the index exists and is automatically freed when the index is deleted. And a collection of extent is what is called as the segments. So there are different kinds of segments, what might be termed as data segments and index segments. And data segments are those which store data or which handle elements that are stored in the data files.

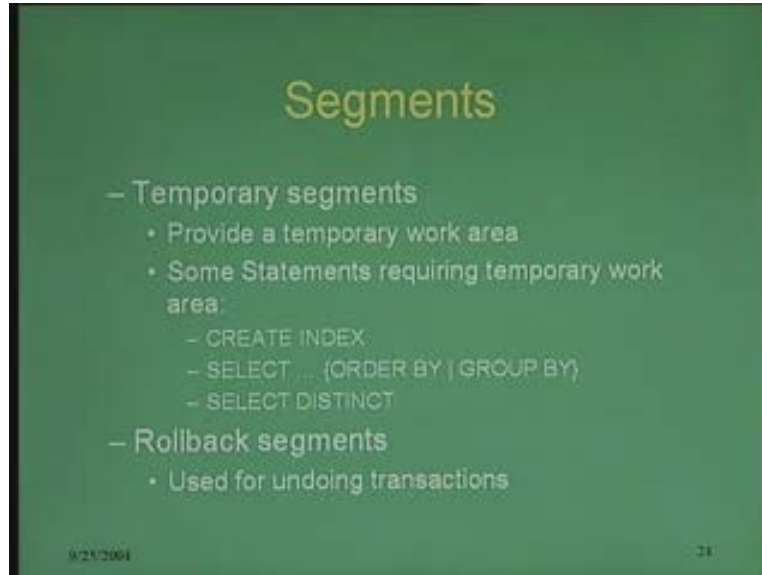
(Refer Slide Time: 40:04)



And data segments belongs to each what is called as a non-cluster table and to each cluster. What is a non-cluster table? As we noted earlier, a cluster is a collection of machines on a lan which gives us the single system interface. Now a cluster table is something that is spread across the cluster that is a part of my table is in one machine and part of it is in another machine and so on and so forth.

Now for every such part in a given machine, a given data segment is allocated. And index segments are allocated for every index structures that's been created on these tables, so every create index command will allocate a index segment. There are also other segments called as temporary segments which are used as temporary work areas especially to store intermediate results or to allocate data for virtual tables or results of an intermediate query and so on.

(Refer Slide Time: 41:17)



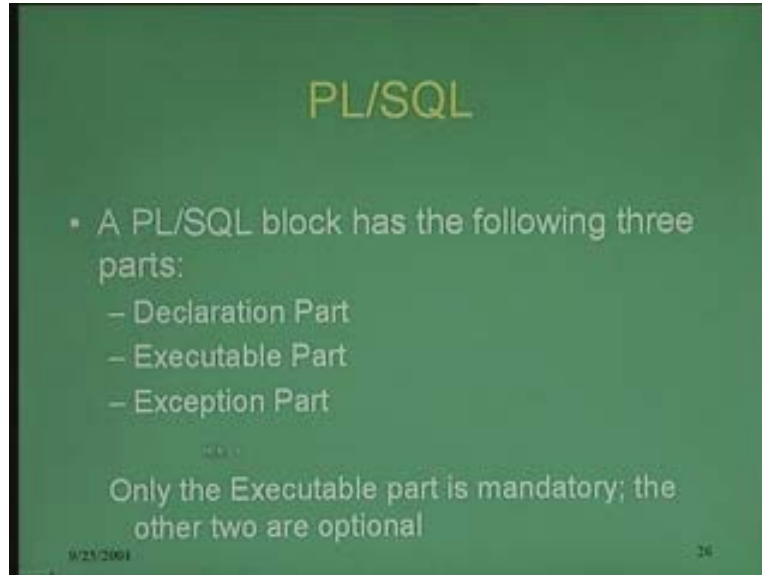
And then there are rollback segments that are used for undoing transactions. So like MySQL or any other DBMS, oracle servers are usually expose several different APIs, so that user processes or application programs can directly talk to the oracle DBMS.

(Refer Slide Time: 41:53)



So the sql support or embedded sql support is provided for different languages like COBOL, C, Pascal and so on and in addition oracle has its own procedural language called the PL/SQL.

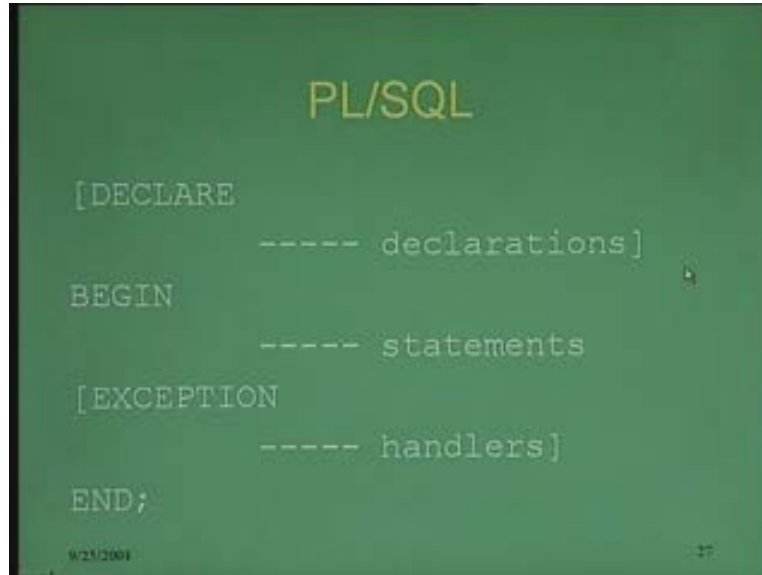
(Refer Slide Time: 42:47)



Let us have a brief look at how PL/SQL looks like and what kinds of operations does it provide over and above plane vennila SQL. PL/SQL as we noted earlier provides procedural constructs within which SQL constructs can be embedded. A PL/SQL block can be divided into three different parts what are called as the declaration part, executable part and the exception part. As you know, if you notice correctly this is very similar to how a typical procedural language is also ordered except for this exception part and out of these three parts it's only the executable part that is mandatory.

So the declaration part is where you declare variables or and any other characteristics or functions that PL/SQL uses and the set of statements or the set of executions is embedded within a begin and a end construct here where you have the set of statements as well as the exception handlers. That is the executable part as well as the exception part written within this block that is within this begin and end construct.

(Refer Slide Time: 43:23)



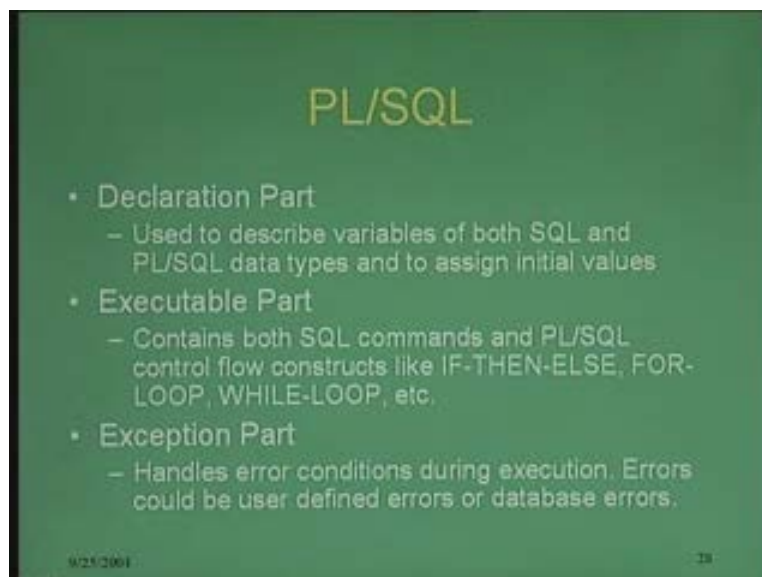
The slide shows the structure of a PL/SQL block. It is titled "PL/SQL" in yellow. The structure is as follows:

```
[DECLARE  
    ----- declarations]  
BEGIN  
    ----- statements  
[EXCEPTION  
    ----- handlers]  
END;
```

At the bottom left, it says "9/23/2004" and at the bottom right, it says "27".

So the declaration part like we noted is used to declare variables and this could be variables that are used in both SQL and PL/SQL data types. And the SQL and the executable part contain both SQL commands and PL/SQL constructs like if then else or for loop and while loop and so on.

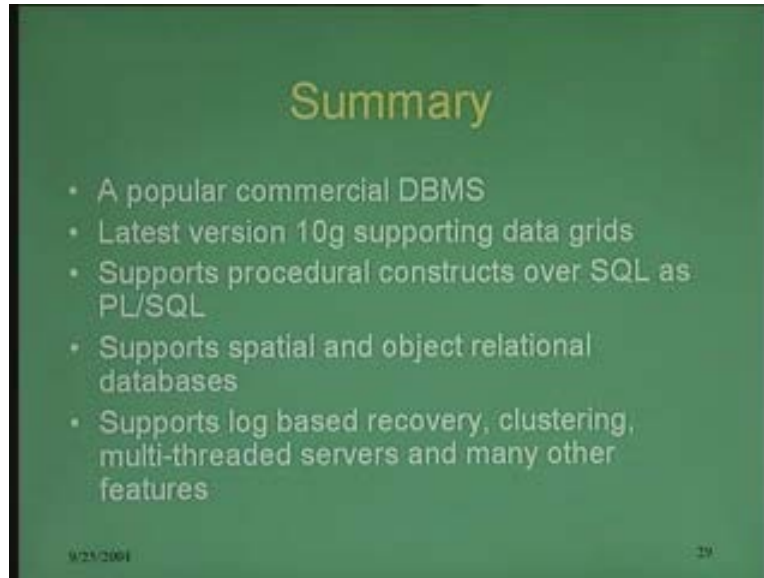
(Refer Slide Time: 43:41)

- 
- The slide is titled "PL/SQL" in yellow. It lists three parts of a PL/SQL block:
- Declaration Part
 - Used to describe variables of both SQL and PL/SQL data types and to assign initial values.
 - Executable Part
 - Contains both SQL commands and PL/SQL control flow constructs like IF-THEN-ELSE, FOR-LOOP, WHILE-LOOP, etc.
 - Exception Part
 - Handles error conditions during execution. Errors could be user defined errors or database errors.
- At the bottom left, it says "9/23/2004" and at the bottom right, it says "28".

So you can say if some condition then send this SQL statement, else send this other SQL statement and so on. And the exception part handles error conditions whenever an exception is raised and errors could be either system errors that are flagged by the

database itself or they could be user defined errors that are flagged by the violation of for example some integrity constraint.

(Refer Slide Time: 44:54)



So let us summarize what we have seen in oracle before having a brief look at MS Access as well. Oracle is a very popular commercial database management systems. It's a commercial one and it's in contrast to the MySQL which is an open source DBMS and the latest version 10g supports data grids as part of oracle. And oracle actually supports not just relational, it also supports object relational databases, spatial databases and database of sequences and so on. So different kinds of data can be stored in oracle rather than the pure relational database and it supports procedural constructs over SQL in the form of PL/SQL where you can write control flow constructs if then else and while loop and for loop and so on.

And oracle supports transactions and log based recovery, it also supports clustering where you can have a data file that is clustered over or a table that is distributed over several machines in a cluster and several other features. Let us now briefly look into the second commercial database whose case study that we are seen which is the Microsoft Access. **Microsoft Access we have** we have included Microsoft Access as part of this case study mainly because it's a commercial database which is kind of tailor made for ease of use for a non-technical user and where everything can be performed using graphical interfaces. So the set up time and the learning curve for this database is much smaller than any other commercial database like oracle or other database like DB 2 and even probably MySQL.

So Microsoft Access provides the database engine and a GUI based that's the graphical user interface based mechanism for almost anything that is for data definition, data manipulation, queries and reporting and almost anything to do with a database can be performed using graphical user interfaces.

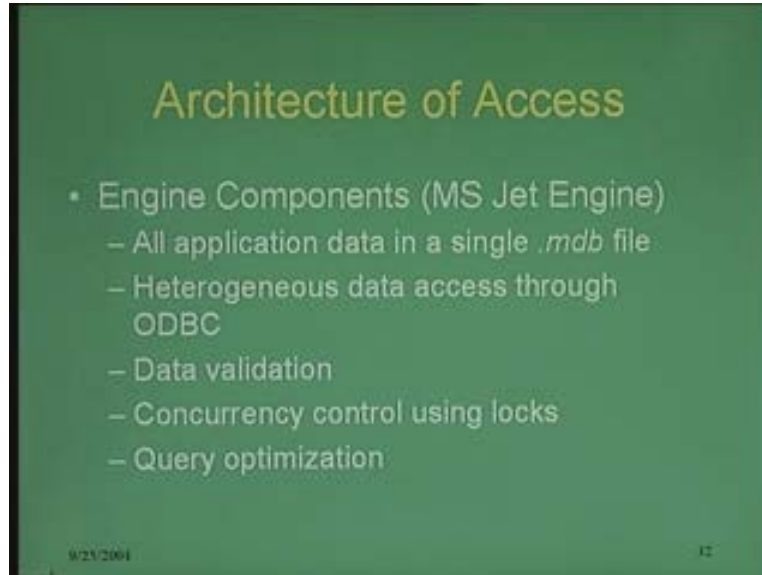
(Refer Slide Time: 47:18)



But it's not that, it's only through the GUI that MS Access can be accessed. You can also access the database using basic that is visual basic and several other macros that you can write to form of your own procedures and so on. And one feature of Ms Access is it provides hyperlinks that is url as a native data type which is generally not supported by other databases where we can treat a url as a data type by itself and not as a string.

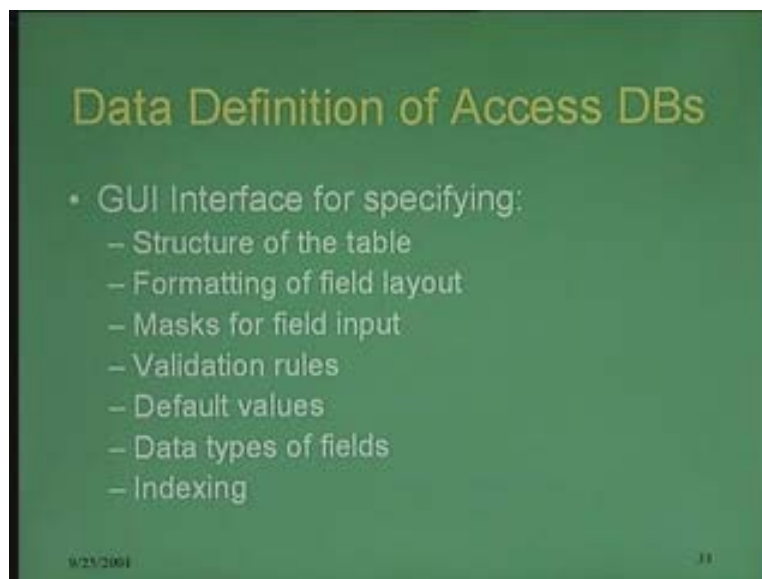
So what are the different components of the Microsoft Access database. It's called the engine of the of the Microsoft Access database. All data for a given application are stored in a single file which has a suffix called dot mdb but even then, you can access these mdb files through different ways using the open database connectivity mechanisms. We have not talked about ODBC as yet but generally ODBC is essentially is a common interface by which you can access several different databases as though they were data sources using a single common interface. And MS Access provides the support for data validation and concurrency control using logs but not full acid semantics and it also provides some amount of query optimization. What kinds of GUIs are available? We have some screen shorts which we will also see of these GUIs later on.

(Refer Slide Time: 47:47)



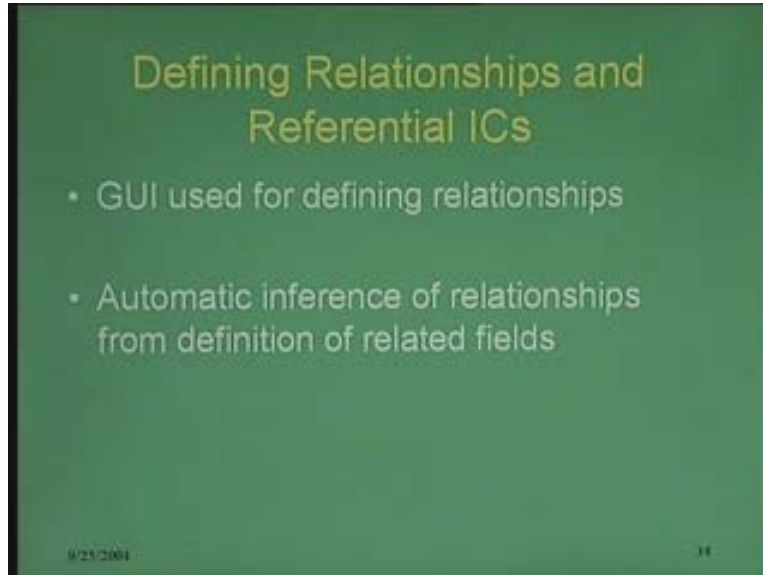
There are GUIs that are available for specifying the structure of a table, formatting of the of the field layout, any kinds of masks which talks about invalid inputs and validation rules, default values, data types, index structures and anything to do with database design can be specified using graphical user interfaces.

(Refer Slide Time: 49:07)



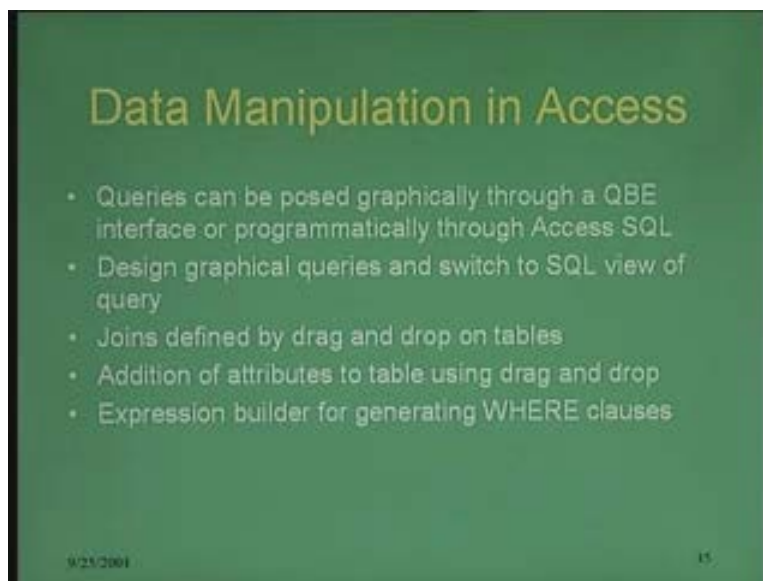
You can also use GUIs for defining foreign key relationships and you can also enable what is called as an automatic inference of relationships that is whenever we use the primary key field of one table as an attribute of another table, there is automatically MS Access would infer that there is a foreign key relationship.

(Refer Slide Time: 49:43)



So it also provides this facility of an automatic inference of relationships between different fields. Then queries can be graphically posed through a what is called as a query by example interface that is you can formulate or you can graphically show how your query should look like, that is I should have these tables and these tables and these tables with these associations and so on as part of my query result and then say now give me the query.

(Refer Slide Time: 50:19)



So there is a QBE interface or query by example interface and as and when you build your query as part of the QBE interface, programmatically there is an SQL statement that

is being created to which also you can switch to and change your query whenever you can. And you can perform joins by drag and drop operations between tables and you can form addition or and deletion of attributes to tables using drag and drop. Then there are what are called as expression builders where in which you can specify that the constructs of the WHERE clause of your SQL query.

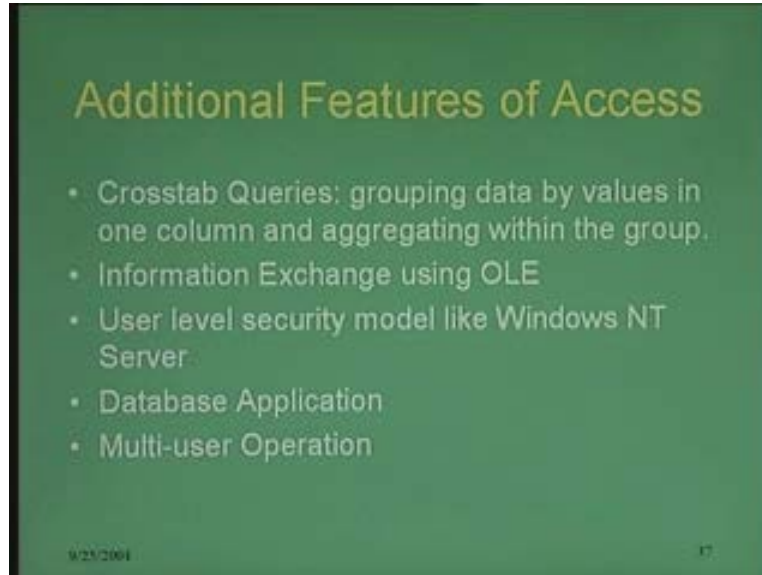
(Refer Slide Time: 51:50)



Reports generation again can be performed in a GUI fashion that is it's an integral part of the access database. The access database doesn't just return a query and leave it at that, you can actually specify how the return data should look like in the form of a report, let us say in a page where different elements of your query go into different parts of the page. So it becomes a reporting mechanism. And the reporting, whatever reporting mechanisms that you have created are tightly bound to the underlying database tables and queries. So there are different report generation wizards which help you in showing how a database report can be created and different styles in which report can be created.

So many other additional features what are called as cross tab queries which wherein you can perform group by on specific values within a column and aggregating within the group that you perform the group by on and the tables that you have generated or your database are available as OLE objects. OLE is the well-known object linking and embedding mechanism that are used across different Microsoft applications, therefore for example you can actually use your database table as an embedded table in a word document for example or in a power point presentation and so on.

(Refer Slide Time: 52:08)

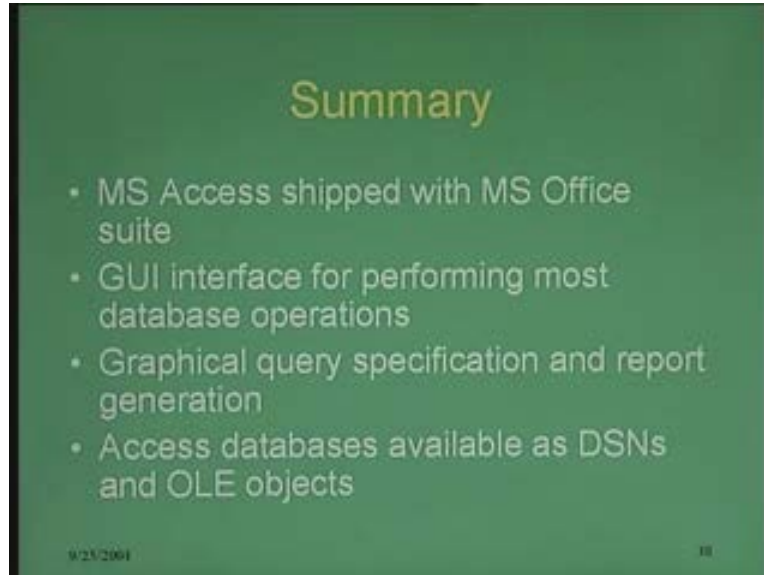


And there is a user level security based on login and password that is based around the NT server model. And it also performs multi user, it also supports multi user operations and concurrent clients and so on. So to summarize what we learnt on MS Access, MS Access is a DBMS system that is meant primarily for the non-technical end user where the learning curve is much smaller than in using say oracle or MySQL or any other larger commercial database system.

It is shipped with the MS Office with, so whenever you buy the MS Office suite of packages MS Access usually comes shipped with it. And the main feature of MS Access is the GUI or the graphical user interface which can help you perform almost any database operation whether it is specifying your database or formulating queries or reporting or any other kind of database operations.

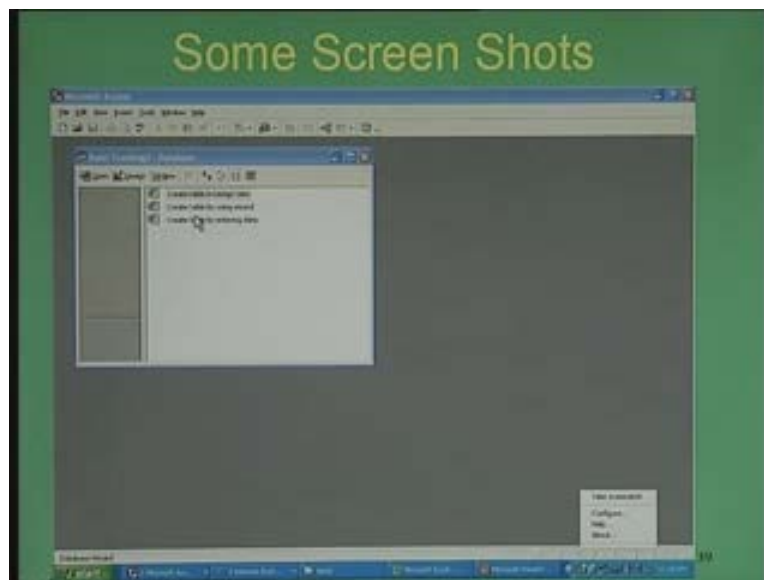
So it gives support for graphical query specification and also report generation and these databases are available as DSNs or data source names through an ODBC interface that is any database client that is compatible with ODBC can access MS Access databases as easily as it can access other ODBC comply databases. Oracle also is also ODBC comply and several most widely available databases are ODBC comply.

(Refer Slide Time: 53:02)



And these tables or database objects in MS Access are available as OLE objects that can be linked or embedded within other MS Office or within other Microsoft applications like say MS Word or EXCEL or POWER POINT or so on.

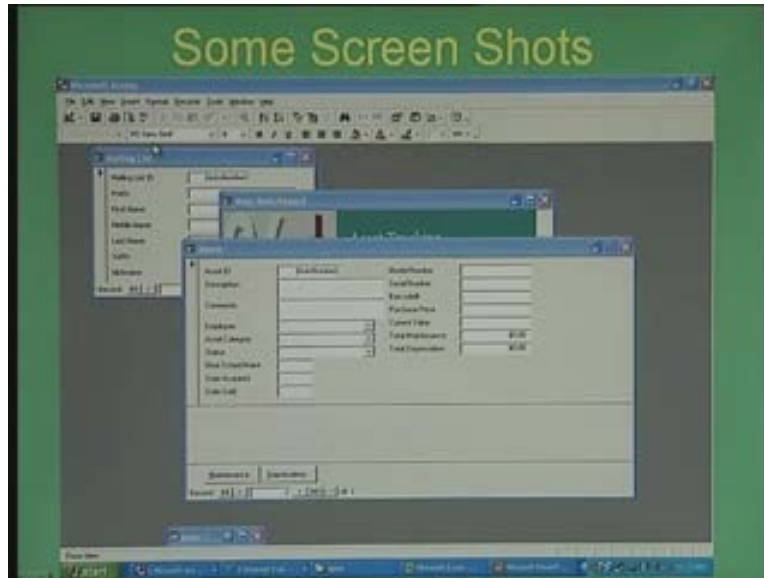
(Refer Slide Time: 55:27)



So let us have a look at some of these screen shots here that show how a typical MS Access or working with MS Access looks like. So when you start a database, this is what you get that is you can start a database where you can specify your database design using different views, there what is called as a design view or you can create tables using a wizard where it will automatically fill up certain well known fields in a table or suggest

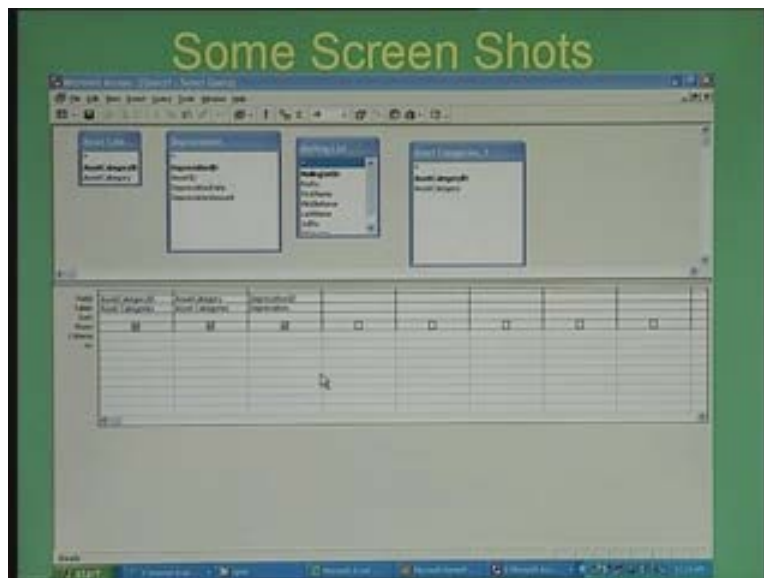
you for certain structures for your database or you can even create tables by actually entering data using a form.

(Refer Slide Time: 56:03)



This screen shot here shows an actual form by which data can be populated. So here there are two different forms for two different tables. There is table called mailing list and there is a table called assets and these are the different fields that go into this list. So you basically fill up this form and the data is actually stored on to the database.

(Refer Slide Time: 56:21)



The last screen shot here shows how you can graphically build a query using the QBE interface or the query by interface. The window here essentially says that or the user who is formulating the query has said that the result of this query should contain these 4 different tables, out of these 4 different tables these are the fields that are required and the user can specify and basically there is a show which can be clicked or unclicked which will say what to show in the table. And the criteria can be specified here which will say how to formulate the query. So that is the beauty of this MS Access databases. So with that we come to the end of the second case study.