**Atomicity** requires that *all* operations (SQL requests) of a transaction be completed; if not, the transaction is aborted.

**Consistency** indicates the permanence of the database’s consistent state.

**Isolation** means that the data used during the execution of a transaction cannot be used by a second transaction

until the first one is completed.

**Durability** ensures that once transaction changes are done and committed, they cannot be undone or lost,

even in the event of a system failure.

**Serializability** ensures that the schedule for the concurrent execution of the transactions yields consistent

results.

Coordinating the simultaneous execution of transactions in a multiuser database system is known as

**concurrency control.**

**Concurrency control** is important because the simultaneous

execution of transactions over a shared database can create several data integrity and consistency problems. The

three main problems are **lost updates,** **uncommitted** data, and **inconsistent** **retrievals**.

**The lost update** problem occurs

when two concurrent transactions, T1 and T2, are updating the same data element

and one of the updates is lost

T1 SELECT data FROM table; T2 SELECT data FROM table; T1 UPDATE data to 25; T2 UPDATE data to 35; T1 COMMIT; T2 COMMIT; T1 is lost

The phenomenon of **uncommitted** **data** occurs when two transactions, T1 and T2, are executed concurrently and

the first transaction (T1) is rolled back after the second transaction (T2) has already accessed the uncommitted data

**Inconsistent retrievals** occur when a transaction accesses data before and after one or more other transactions

finish

working with such data.

**Read uncommited**

START TRANSACTION; UPDATE actor SET first\_name = 'bob' WHERE actor\_id = 25; SELECT SLEEP(15); UPDATE actor SET first\_name = 'frank' WHERE actor\_id = 25; SELECT first\_name FROM actor WHERE actor\_id = 25; COMMIT;

START TRANSACTION; SELECT SLEEP(5); SELECT first\_name FROM actor WHERE actor\_id = 25; COMMIT;

**Dirty Read** a transaction can read data that is not yet committed.

**Phantom Read** a transaction executes a query at time t1, and then it runs the same query at time t2, yielding

additional rows that satisfy the query.

**Unrepeatable read** a transaction reads a given row at time t1, and then it reads the same row at time t2,

yielding different results.

**Read Uncommitted –**Read Uncommitted is the lowest isolation level. In this level, transactions are not isolated from each other.

**Read Committed –**This isolation level guarantees that any data read is committed at the moment it is read. The transaction holds a read or write lock on the current row

**Repeatable Read –**This is the most restrictive isolation level. The transaction holds read locks on all rows it references and writes locks on all rows it inserts, updates, or deletes

**Serializable –**This is the Highest isolation level. Serializable execution is defined to be an execution of operations in which concurrently executing transactions appears to be serially executing.

**Binary lock** Two states: locked (1) and unlocked (0)

**Exclusive lock** (Write Lock) Access is reserved for the transaction that locked the object

**Shared lock** (Read Lock) Concurrent transactions are granted read access on the basis of a common lock

**Deadlocks** Occur when two transactions wait indefinitely for each other to unlock data Also known as deadly embrace Control techniques, prevention, detection, avoidance

**scheduler** is a special DBMS process that establishes the order in which the operations are executed within

concurrent transactions

The exclusive lock is granted

if and only if no other locks are held on the data item **Mutual Exclusive rule**

**2 Phase Lock** A growing phase, in which a transaction acquires all required locks without unlocking any data. 2. A shrinking phase, in

which a transaction releases all locks and cannot obtain a new lock.

**Deadlock prevention.** A transaction requesting a new lock is aborted when there is the possibility that a deadlock

can occur.

**Deadlock detection.** The DBMS periodically tests the database for deadlocks. If a deadlock is found, the

victim transaction is aborted

**Deadlock avoidance.** The transaction must obtain all of the locks it needs before it can be executed.

The choice of which deadlock control method to use depends on the database environment. For example, if the probability of deadlocks is low, deadlock detection is recommended. if the probability of deadlocks is high,

deadlock prevention is recommended. If response time is not high on the system’s priority list, deadlock avoidance might be employed.

**Time stamps** must have two properties: **uniqueness** and **monotonicity**. Uniqueness ensures that no equal time stamp values

can exist, and monotonicity1 ensures that time stamp values always increase.

**disadvantage of the time stamping** approach is that each value stored in the database requires two additional time stamp fields: one for the last time the field was read and one for the last update. Time stamping thus increases memory needs and processing OH

the **wait/die scheme**, the older transaction waits for the younger one to complete and release its locks.

the **wound/wait scheme**, the older transaction rolls back the younger transaction and reschedules it.

Using an **optimistic approach**, each transaction moves through two or three phases,

referred to as read, validation, and write

The **write-ahead-log protocol** ensures that transaction logs are always written *before* any database data are updated

**Redundant transaction logs** ensure that a physical disk failure will not

impair the DBMS’s ability to recover data.

**Database buffers** are temporary storage areas in primary memory used to speed up disk operations

**checkpoints** are operations in which the DBMS writes all of its updated buffers in memory to disk.

Checkpoints that are too frequent would affect transaction performance; too infrequent would affect

database recovery performance

When the recovery procedure uses a **deferred-write technique**, the transaction operations do not immediately update the physical database. Instead, only the transaction log is updated. When the recovery procedure uses a **write-through** technique, the database is

immediately updated even before the transaction reaches its commit point.

**data warehouse** an integrated, subject-oriented, time-variant, nonvolatile collection of data that provides support for decision making. the data warehouse is a read-only database optimized for data analysis and query processing

**star schema** is a data-modeling technique used to map multidimensional decision support data into a relational

database.

Facts are normally stored in a fact table that is the center of the star schema. The fact table contains facts that are linked through their dimensions. The fact table is related

to each dimension table in a many-to-one (M:1) relationship. Because the fact table is related to many dimension tables, the primary key of the fact table is a composite primary key

the ability to focus on slices of the cube to perform a more detailed analysis is known as **slice and dice**

ODBC, OLE-DB, and ADO.NET form the backbone of Microsoft’s **Universal Data Access (UDA)** architecture,

**Data Access Objects (DAO)** is an object-oriented API used to access MS Access, MS FoxPro, and dBase

databases (using the Jet data engine) from Visual Basic programs

**Remote Data Objects (RDO)** is a higher-level, object-oriented application interface used to access remote

database servers.

ODBC, DAO, and RDO are implemented as shared code that is dynamically linked to the Windows operating environment through **dynamic-link libraries (DLLs)**, which are stored as files with a .dll extension

The basic ODBC architecture has three main components:

• A high-level *ODBC API* through which application programs access ODBC functionality

• A *driver manager* that is in charge of managing all database connections

• An *ODBC driver* that communicates directly to the DBMS

Defining a data source is the first step in using ODBC. To define a data source, you must create a **data source name (DSN)** to create one you need to provide an ODBC driver, a name, and ODBC driver parameters

-Java Database Connectivity (JDBC) is an application programming interface that allows a Java program to interact with a wide range of data sources, including relational databases, tabular data sources, spreadsheets, and text files

**-Benefits of internet tech** hardware & software independence. Common and simple UI, location independence, rapid development at manageable costs.

-the web server must include the **database query result on the page *before***it sends that page back to the

client

-A **server-side extension** is a program that interacts directly with the web server to handle specific types of requests

-A database server-side extension program is also known as **web-to-database middleware**

The **web-to-database** middleware compiles the result set, dynamically generates an HTML-formatted page that includes the data retrieved from the database, and sends it to the web server.

-The **Common Gateway Interface (CGI)** uses script files that perform specific functions based on the client’s

parameters that are passed to the web server

-The main **disadvantage of using CGI** scripts is that the script file is an external program that executes separately for each user request and therefore causes a resource bottleneck. Performance also could be degraded by using an interpreted

language or by writing the script inefficiently.

-An **application programming interface (API)** is a newer web server interface standard that is more efficient. APIs are more efficient because they are implemented as shared code or as dynamic-link libraries (DLLs). That means the API is treated as part of the web server program that is dynamically invoked when needed.

**-APIs are faster than CGI scripts** because the code resides in memory, so there is no need to run an external program for each request. Instead, the same API serves all requests. Another advantage is that an API can use a shared connection to the database instead of creating a new one every time, as is the case with CGI scripts.

-Because the APIs share the

same memory space as the web server, an API error can bring down the web server. Another **disadvantage** is that APIs are specific to the web server and to the operating system.

The **web browser**, through its use of HTML, does not have computational abilities beyond formatting output text and accepting form field inputs.

**Client-side extensions** add functionality to the web browser

A **plug-in** is an external application that is automatically invoked by the browser when needed

**JavaScript** is a scripting language (one that enables the execution of a series of commands or macros) that allows web authors to design interactive sites

**ActiveX** is Microsoft’s alternative to Java. ActiveX is a specification for writing programs that run inside the Microsoft client browser, Internet Explorer

**ActiveX controls** can be created in several programming languages; C++ and Visual Basic are most

Commonly used. Microsoft’s .NET framework allows for wider interoperability of ActiveX-based applications (such as ADO.NET) across multiple operating environments.

**VBScript** is another Microsoft product that is used to extend browser functionality

A **web application server** is a middleware application that expands the functionality of web servers by linking them to a wide range of services, such as databases, directory systems, and search engines

**Examples of web application servers** include ColdFusion/JRun by Adobe, WebSphere Application Server by IBM, WebLogic Server by Oracle, Fusion by NetObjects, Visual Studio .NET by Microsoft, and WebObjects by Apple

**Web database development** deals with the process of interfacing databases with the web browser—in short, how to create webpages that access data in a database.

**cloud computing** is a computing model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computer resources (e.g., networks, servers, storage, applications and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction

**Public cloud**. This type of cloud infrastructure is built by a third-party organization to sell cloud services to the general public

**Private cloud** This type of internal cloud is built by an organization for the sole purpose of servicing its own

needs.

**Community cloud** This type of cloud is built by and for a specific group of organizations that share a common

trade, such as agencies of the federal government, the military, or higher education

**Features of clouds** Ubiquitous access via Internet technologies, Shared infrastructure, Lower costs and variable pricing, Flexible and scalable services, Dynamic provisioning, Service orientation, Managed operations

**Cloud service types,** Software as a service, Platform as a service, infrastructure as a service

**Disadvantages of clouds** issues of security, privacy and compliance, hidden costs, data migration is difficult, complex licensing, loss of ownership and control, organization culture resistance, difficult integration with internal IT