CS/IT 315

Hall Ticket Number:

III/IV B.Tech (Supplementary) DEGREE EXAMINATION

November, 2017 **Common for CSE & IT**

Fifth Semester **Database Management Systems**

Time: Three Hours **Maximum :** 60 Marks

Answer Question No.1 compulsorily. (1X12 = 12 Marks)

Answer ONE question from each unit. (4X12=48 Marks)

1. Answer all questions (1X12=12 Marks)

a) What are the responsibilities of a DBA?

The DBA is responsible for authorizing access to the database, coordinating and monitoring its use, and acquiring software and hardware resources as needed.

b) Differentiate between database schema and database state.

Database Schema is the overall Design of the Database. It is the skeleton structure that represents the logical view of the entire database. It tells how the data is organized and how the relations among them are associated.

Database State: Refers to the content of a database at a moment in time. The data in the database at a particular moment in time is called a database state or snapshot.

c) What is a weak entity type?

The entity set which does not have sufficient attributes to form a primary key is called as Weak entity set.

d) What is a foreign key?

A foreign key is a field (or collection of fields) in one table that uniquely identifies a row of another table or the same table.[1][2][3] In simpler words, the foreign key is defined in a second table, but it refers to the primary key or a unique key in the first table.

e) What is outer join?

A set of operations, called outer joins, were developed for the case where the user wants to keep all the tuples in R, or all those in S, or all those in both relations in the result of the JOIN, regardless of whether or not they have matching tuples in the other relation.

f) What is a view?

A view is a virtual table based on the result-set of an SQL statement. A view contains rows and columns, just like a real table. The fields in a view are fields from one or more real tables in the database.

g) Differentiate between B-tree and B+-tree.

1.In a B tree search keys and data stored in internal or leaf nodes. But in B+-tree data store only leaf nodes.
2.Searching any data in a B+ tree is very easy because all data are found in leaf nodes. In a B tree, data cannot be found in leaf nodes.
3.In a B tree, data may be found in leaf nodes or internal nodes. Deletion of internal nodes is very complicated. In a B+ tree, data is only found in leaf nodes. Deletion of leaf nodes is easy.
4. Insertion in B tree is more complicated than B+ tree.
5. B+ trees store redundant search key but B tree has no redundant value .

h) What is Multivalued Dependency?

Definition. A multivalued dependency *X*→→*Y* specified on relation schema *R*, where *X* and *Y* are both subsets of *R*, specifies the following constraint on any relation state *r* of *R*: If two tuples *t*1 and *t*2 exist in *r* such that *t*1[X] = *t*2[X], then two tuples *t*3 and *t*4 should also exist in *r* with the following properties,15 where we use *Z* to denote (*R* – (*X* ∪Y)).

■ *t*3[*X*] = *t*4[*X*] = *t*1[*X*] = *t*2[*X*].
■ *t*3[*Y*] = *t*1[*Y*] and *t*4[*Y*] = *t*2[*Y*].
■ *t*3[*Z*] = *t*2[*Z*] and *t*4[*Z*] = *t*1[*Z*].

Whenever *X*→→*Y* holds, we say that *X* **multidetermines** *Y*.

i) What is lossless join?

A decomposition D = {R1, R2, ..., Rm} of R has the lossless (nonadditive) join property with respect to the set of dependencies F on R if, for every relation state r of R that satisfies F, the following holds, where * is the NATURAL JOIN of all the relations in D: *(πR1(r), ..., πRm(r)) = r.

j) What is a strict schedule?

Strict schedule is a schedule in which transactions can neither read nor write an item X until the last transaction that wrote X has committed (or aborted).

k) What is a timestamp?

Timestamp is a unique identifier created by the DBMS to identify a transaction.

l) What is a transaction?

A transaction is an executing program that forms a logical unit of database processing. A transaction includes one or more database access operations—these can include insertion, deletion, modification, or retrieval operations.

UNIT I

2.a Discuss the advantages of using database approach. 6 M

Advantages of DBMS Any 6 advantages-----6M

 Controlling Data Redundancy

In non-database systems each application program has its own private files. In this case, the duplicated copies of the same data is created in many places. In DBMS, all data of an organization is integrated into a single database file. The data is recorded in only one place in the database and it is not duplicated.

 Sharing of Data

In DBMS, data can be shared by authorized users of the organization. The database administrator manages the data and gives rights to users to access the data. Many users can be authorized to access the same piece of information simultaneously. The remote users can also share same data. Similarly, the data of same database can be shared between different application programs.

 Data Consistency

By controlling the data redundancy, the data consistency is obtained. If a data item appears only once, any update to its value has to be performed only once and the updated value is immediately available to all users. If the DBMS has controlled redundancy, the database system enforces consistency.

 Integration of Data

In Database management system, data in database is stored in tables. A single database contains multiple tables and relationships can be created between tables (or associated data entities). This makes easy to retrieve and update data.

 Integration Constraints

Integrity constraints or consistency rules can be applied to database so that the correct data can be entered into database. The constraints may be applied to data item within a single record or the may be applied to relationships between records.

 Data Security

Form is very important object of DBMS. You can create forms very easily and quickly in DBMS. Once a form is created, it can be used many times and it can be modified very easily. The created forms are also saved along with database and behave like a software component. A form provides very easy way (user-friendly) to enter data into database, edit data and display data from database. The non-technical users can also perform various operations on database through forms without going into technical details of a database.

 Control Over Concurrency

In a computer file-based system, if two users are allowed to access data simultaneously, it is possible that they will interfere with each other. For example, if both users attempt to perform update operation on the same record, then one may overwrite the values recorded by the other. Most database management systems have sub-systems to control the concurrency so that transactions are always recorded with accuracy.

 Backup and Recovery Procedures

In a computer file-based system, the user creates the backup of data regularly to protect the valuable data from damage due to failures to the computer system or application program. It is very time consuming method, if amount of data is large. Most of the DBMSs provide the 'backup and recovery' sub-systems that automatically create the backup of data and restore data if required.

2.b Draw and explain the three schema architecture of a database system. What is data independence? Explain it. 6 M

Explanation of 3 Levels--2M

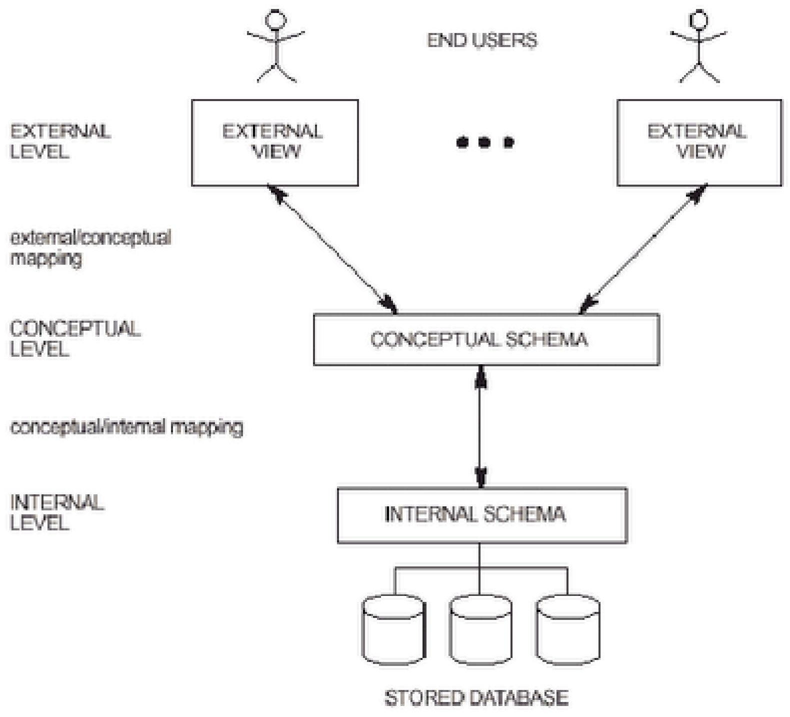
The goal of the three-schema architecture is to separate the user applications and the physical database. In this architecture, schemas can be defined at 3 levels :

1. Internal level or Internal schema : Describes the physical storage structure of the database. The internal schema uses a physical data model and describes the complete details of data storage and access paths for the database

2. Conceptual level or Conceptual schema : Describes the structure of the whole database for a community of users. It hides the details of physical storage structures and concentrates on describing entities, data types, relationships, user operations, and constraints. Implementation data model can be used at this level.

3. External level or External schema : It includes a number of external schemas or user views. Each external schema describes the part of the database that a particular user is interested in and hides the rest of the database from user. Implementation data model can be used at this level.

Diagram--1M



Data independence

Definition--1M
Explanation--2M

The three-schema architecture can be used to further explain the concept of data independence, which can be defined as the capacity to change the schema at one level of a
database system without having to change the schema at the next higher level. We can define
two types of data independence:

1. **Logical data independence** is the capacity to change the conceptual schema without having to change external schemas or application programs. We may change the conceptual schema to expand the database (by adding a record type or data item), to change constraints, or to reduce the database (by removing a record type or data item).
2**. Physical data independence** is the capacity to change the internal schema without having to change the conceptual schema. Hence, the external schemas need not be changed as well. Changes to the internal schema may be needed because some physical files were reorganized—for example, by creating additional access structures—to improve the performance of retrieval or update. If the same data as before remains in the database, we should not have to change the conceptual schema.

OR

3.a Write about the centralized and client/server architectures for DBMSs. 6 M

Centralized DBMSs Architecture ---3M

Earlier architectures used mainframe computers to provide the main processing for all system functions, including user application programs and user interface programs, as well as all the DBMS functionality. The reason was that most users accessed such systems via computer terminals that did not have processing power and only provided display capabilities. Therefore, all processing was performed remotely on the computer system, and only display information and controls were sent from the computer to the display terminals, which were connected to the central computer via various types of communications networks.

At first, database systems used these computers similarly to how they had used display terminals, so that the DBMS itself was still a centralized DBMS in which all the DBMS functionality, application program execution, and user inter-face processing were carried out on one machine. Figure illustrates the physical components in a centralized architecture. Gradually, DBMS systems started to exploit the available processing power at the user side, which led to client/server DBMS architectures.

Diagram--1M

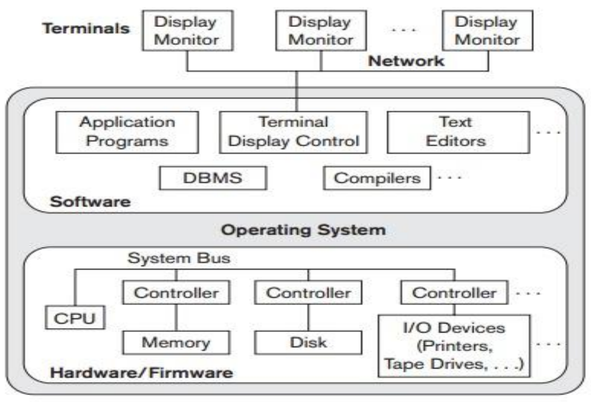


Figure 2.4
A physical centralized architecture.

2. Basic Client/Server Architectures-----3M

The client/server architecture was developed to deal with computing environments in which a large number of PCs, workstations, file servers, printers, database servers, Web servers, e-mail servers, and other software and equipment are connected via a network. The idea is to define specialized servers with specific functionalities. For example, it is possible to connect a number of PCs or small workstations as clients to a file server that maintains the files of the client machines. Another machine can be designated as a printer server by being connected to various printers; all print requests by the clients are forwarded to this machine. Web servers or e-mail servers also fall into the specialized server category. The resources provided by specialized servers can be accessed by many client machines. Theclient machines provide the user with the appropriate interfaces to utilize these servers, as well as with local processing power to run local applications.

Diagram--1M

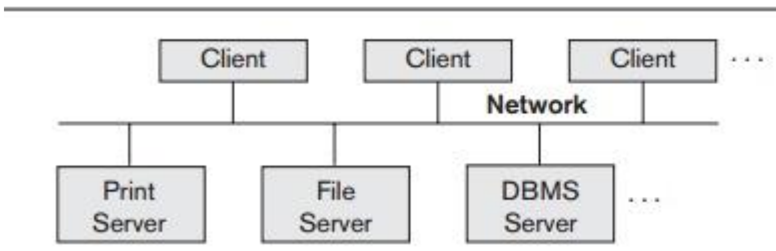
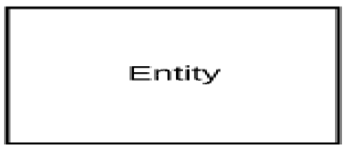


Figure 2.5 Logical two-tier client/server architecture.

3.b Discuss the concepts used in ER data modeling. 6 M

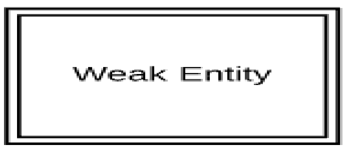
An entity-relationship diagram (ERD) is a data modeling technique that graphically illustrates an information system's entities and the relationships between those entities. An ERD is a conceptual and representational model of data used to represent the entity framework infrastructure

Strong Entity Symbol



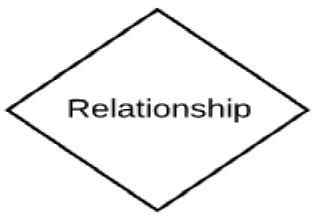
These shapes are independent from other entities, and are often called parent entities, since they will often have weak entities that depend on them. They will also have a primary key, distinguishing each occurrence of the entity.

Weak Entity Symbol



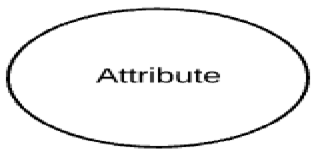
Weak entities depend on some other entity type. They don't have primary keys, and have no meaning in the diagram without their parent entity.

Relationship Symbol



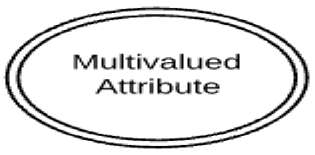
Relationships are associations between or among entities.

Attributes Symbols

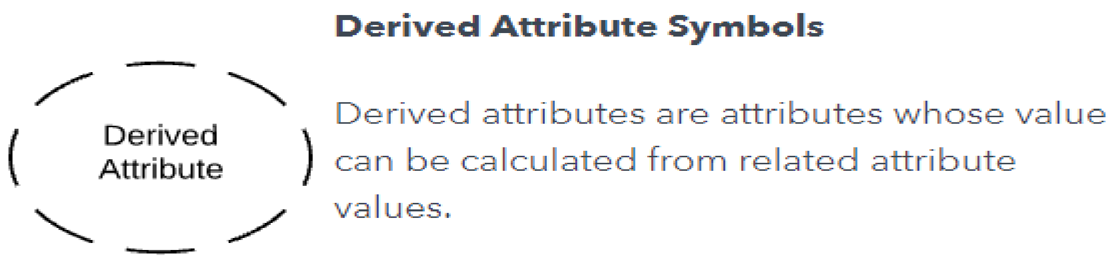


Attributes are characteristics of an entity, a many-to-many relationship, or a one-to-one relationship.

Multivalued Attribute Symbols



Multivalued attributes are those that are can take on more than one value.


Derived Attribute Symbols

Derived attributes are attributes whose value can be calculated from related attribute values.

UNIT-II

4.a Write notes on relational model constraints. 6 M

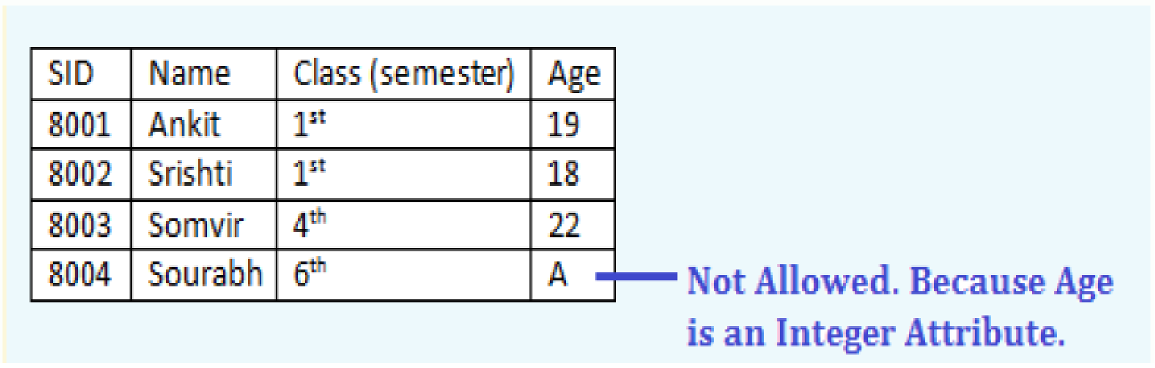
[Examples --2 M]
[Any 4 Constraints explanation ---4M]

- **Domain Constraints**
- **Key Constraints**
- **Single Value Constraints**
- **Entity Integrity Constraint**
- **Referential Integrity Constraint**

Domain Constraints

Domain Constraints specifies that what set of values an attribute can take. Value of each attribute X must be an atomic value from the domain of X. The data type associated with domains include integer, character, string, date, time, currency etc. An attribute value must be available in the corresponding domain. Consider the example below

[Any relevant example can be considered].



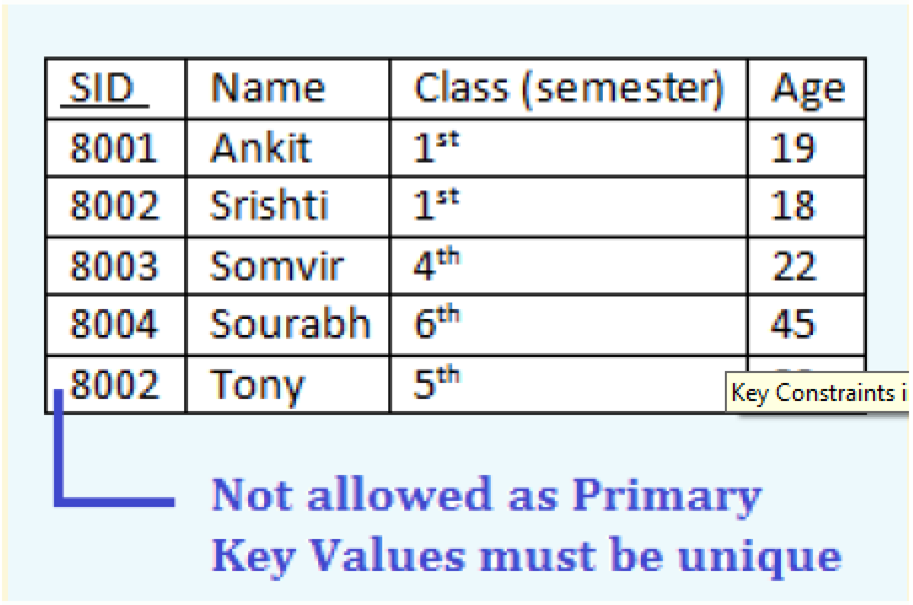
SID	Name	Class (semester)	Age
8001	Ankit	1st	19
8002	Srishti	1st	18
8003	Somvir	4th	22
8004	Sourabh	6th	A

Not Allowed. Because Age is an Integer Attribute.

Key Constraints

Keys are attributes or sets of attributes that uniquely identify an entity within its entity set. An Entity set E can have multiple keys out of which one key will be designated as the primary key. Primary Key must have unique and not null values in the relational table. In an subclass hierarchy, only the root entity set has a key or primary key and that primary key must serve as the key for all entities in the hierarchy.

[Any relevant example can be considered]



SID	Name	Class (semester)	Age
8001	Ankit	1st	19
8002	Srishti	1st	18
8003	Somvir	4th	22
8004	Sourabh	6th	45
8002	Tony	5th	[illegible]

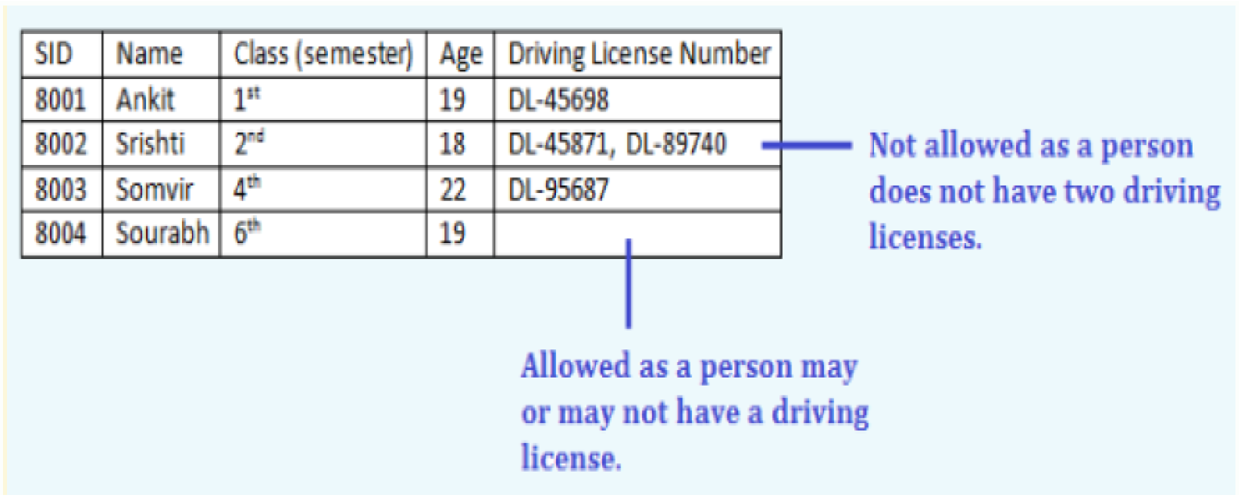
Key Constraints i

Not allowed as Primary Key Values must be unique

Single Value Constraints

Single value constraints refers that each attribute of an entity set has a single value. If the value of an attribute is missing in a tuple, then we cal fill it with a “null” value. The null value for a attribute will specify that either the value is not known or the value is not applicable. Consider the below example

[Any relevant example can be considered]



SID	Name	Class (semester)	Age	Driving License Number
8001	Ankit	1st	19	DL-45698
8002	Srishti	2nd	18	DL-45871, DL-89740
8003	Somvir	4th	22	DL-95687
8004	Sourabh	6th	19	

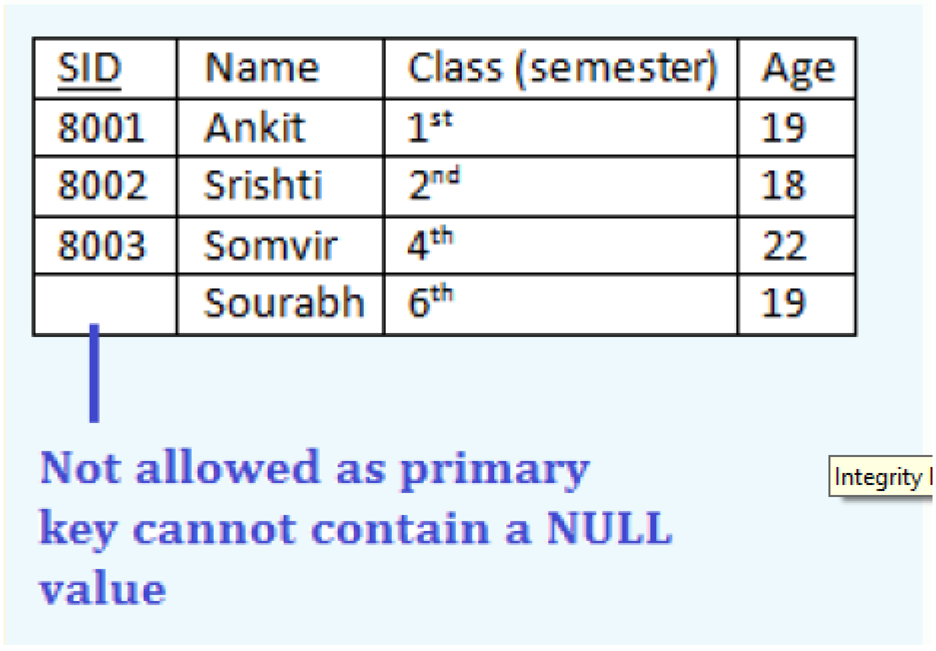
Not allowed as a person does not have two driving licenses.

Allowed as a person may or may not have a driving license.

Entity Integrity Constraint

The Integrity Rule 1 is also called Entity Integrity Rule or Constraint. This rule states that no attribute of primary key will contain a null value. If a relation have a null value in the primary key attribute, then uniqueness property of the primary key cannot be maintained. Consider the example below

[Any relevant example can be considered]



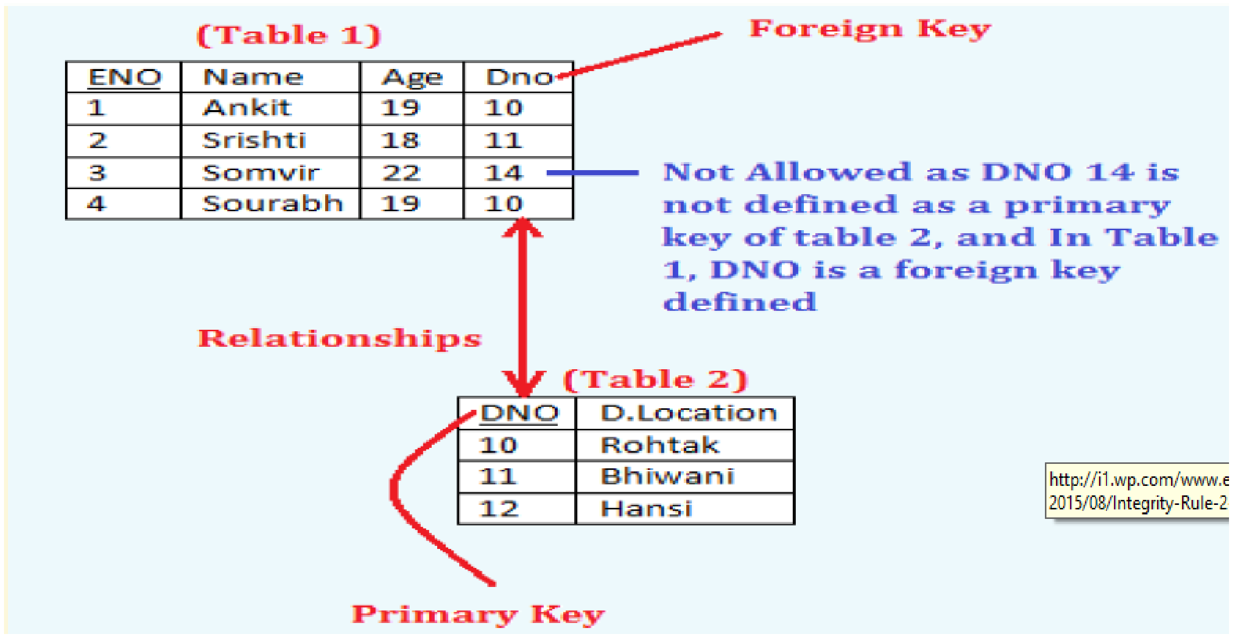
SID	Name	Class (semester)	Age
8001	Ankit	1st	19
8002	Srishti	2nd	18
8003	Somvir	4th	22
	Sourabh	6th	19

Not allowed as primary key cannot contain a NULL value
Integrity I

Referential Integrity Constraint

The integrity Rule 2 is also called the Referential Integrity Constraints. This rule states that if a foreign key in Table 1 refers to the Primary Key of Table 2, then every value of the Foreign Key in Table 1 must be null or be available in Table 2. For example,

[Any relevant example can be considered]



(Table 1)

ENO	Name	Age	Dno
1	Ankit	19	10
2	Srishti	18	11
3	Somvir	22	14
4	Sourabh	19	10

(Table 2)

DNO	D.Location
10	Rohtak
11	Bhiwani
12	Hansi

Foreign Key
Not Allowed as DNO 14 is not defined as a primary key of table 2, and In Table 1, DNO is a foreign key defined
Relationships
Primary Key
http://i1.wp.com/www.e
2015/08/Integrity-Rule-2

4.b Illustrate the basic set of relational algebra operators. **6 M**

Relational algebra is a procedural query language, which takes instances of relations as input and yields instances of relations as output. It uses operators to perform queries. An operator can be either unary or binary. They accept relations as their input and yield relations as their output. Relational algebra is performed recursively on a relation and intermediate results are also considered relations.

The fundamental operations of relational algebra are as follows −

- Select
- Project

- Union
- Set different
- Cartesian product
- Rename

Select Operation (σ)

It selects tuples that satisfy the given predicate from a relation.

Notation − σp(r)

Where σ stands for selection predicate and r stands for relation. p is prepositional logic formula which may use connectors like and, or, and not. These terms may use relational operators like − =, ≠, ≥, < , >, ≤.

For example **[Any relevant example]**

σsubject = "database"(Books)

σsubject = "database" and price = "450"(Books)

Project Operation (Π)

It projects column(s) that satisfy a given predicate.

Notation − ΠA1, A2, An (r)

Where A1, A2 , An are attribute names of relation r. Duplicate rows are automatically eliminated, as relation is a set.

For example – **[Any relevant example]**

Πsubject, author (Books)

Selects and projects columns named as subject and author from the relation Books.

Union Operation (∪) **[Any relevant example]**

It performs binary union between two given relations and is defined as −

r ∪ s = { t | t ∈ r or t ∈ s}

Notation − r U s

Where r and s are either database relations or relation result set (temporary relation).

For a union operation to be valid, the following conditions must hold −

- **r**, and **s** must have the same number of attributes.
- Attribute domains must be compatible.
- Duplicate tuples are automatically eliminated.

Π author (Books) ∪ Π author (Articles)

Output − Projects the names of the authors who have either written a book or an article or both**.**

Set Difference (−) **[Any relevant example]**

The result of set difference operation is tuples, which are present in one relation but are not in the second relation.

Notation − (r − s)

Finds all the tuples that are present in r but not in s.

Π author (Books) − Π author (Articles)

Output

Provides the name of authors who have written books but not articles.

Rename Operation (ρ)
The results of relational algebra are also relations but without any name. The rename operation allows us to rename the output relation. 'rename' operation is denoted with small Greek letter **rho** *ρ*.
Notation − ρ x (E)

5. **Employee(Fname,Minit,Lname,<u>Ssn</u>,Bdate,Address,Sex,Salary,SuperSSN,Dno)** **12M**
Department(Dname, <u>Dnumber</u>,MgrSSN, Mgr_Strt_Date)
Dept_Locations(<u>Dnumber, Dlocation</u>)
Project(Pname, <u>Pnumber</u>, Plocation, Dnum)
Works_On(<u>Essn,Pno</u>,Hours)
Dependent(Essn, <u>Dependent_name</u>, Sex, Bdate, Relationship)
Specify the following queries in SQL on the relational database schema shown in the above figure.

a) **Retrieve the names of all employees in department 5 who work more than 10 hours per week on the ProductX project.**

Select fname from employee, project, works_on where project.pno=works_on.pno and works_on.essn=employee.ssn and employee.dno=5 and project.pname='ProductX'

b) **List the names of all employees who have a dependent with the same first name as themselves.**
Select fname from employee, dependent where employee.ssn=dependent.essn and Dependent.dependent_name=employee.fname

c) **Find the names of all employees who are directly supervised by 'Franklin Wong'.**
Select e1.fname,e2.fname from employee e1, employee e2 where e1.ssn=e2.superssn and e2.fname=' Franklin Wong'.

d) **For each department whose average employee salary is more than $30,000, retrieve the department name and the number of employees working for that department.**
Select avg(salary),count(ssn) from employee group by department having avg(salary)> $30,000

e) **Suppose that we want the number of *male* employees in each department making more than $30,000, rather than all employees**
Select count(ssn) from employees group by deptno where salary>$30,000 and sex='F'

UNIT III

6.a Discuss in detail about the various types of index structures used in database management systems. **6M**

Indexing is a data structure technique to efficiently retrieve records from the database files based on some attributes on which the indexing has been done. Indexing in database systems is similar to what we see in books.

Indexing is defined based on its indexing attributes. Indexing can be of the following types −

 Primary Index − Primary index is defined on an ordered data file. The data file is ordered on a key field. The key field is generally the primary key of the relation.

 Secondary Index − Secondary index may be generated from a field which is a candidate key and has a unique value in every record, or a non-key with duplicate values.

 Clustering Index − Clustering index is defined on an ordered data file. The data file is ordered on a non-key field.

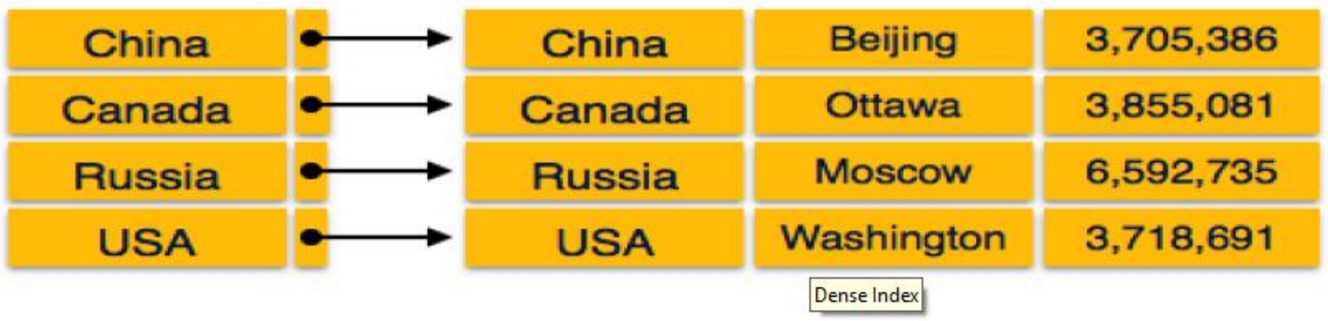
Ordered Indexing is of two types −

 Dense Index

 Sparse Index

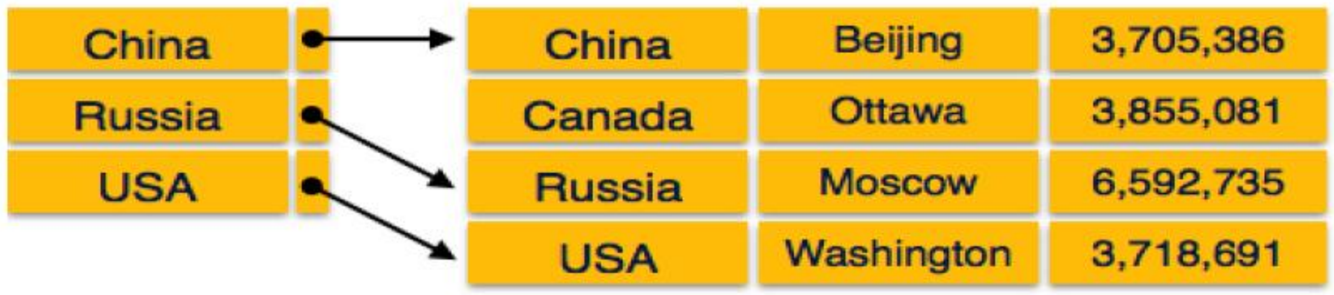
Dense Index

In dense index, there is an index record for every search key value in the database. This makes searching faster but requires more space to store index records itself. Index records contain search key value and a pointer to the actual record on the disk.



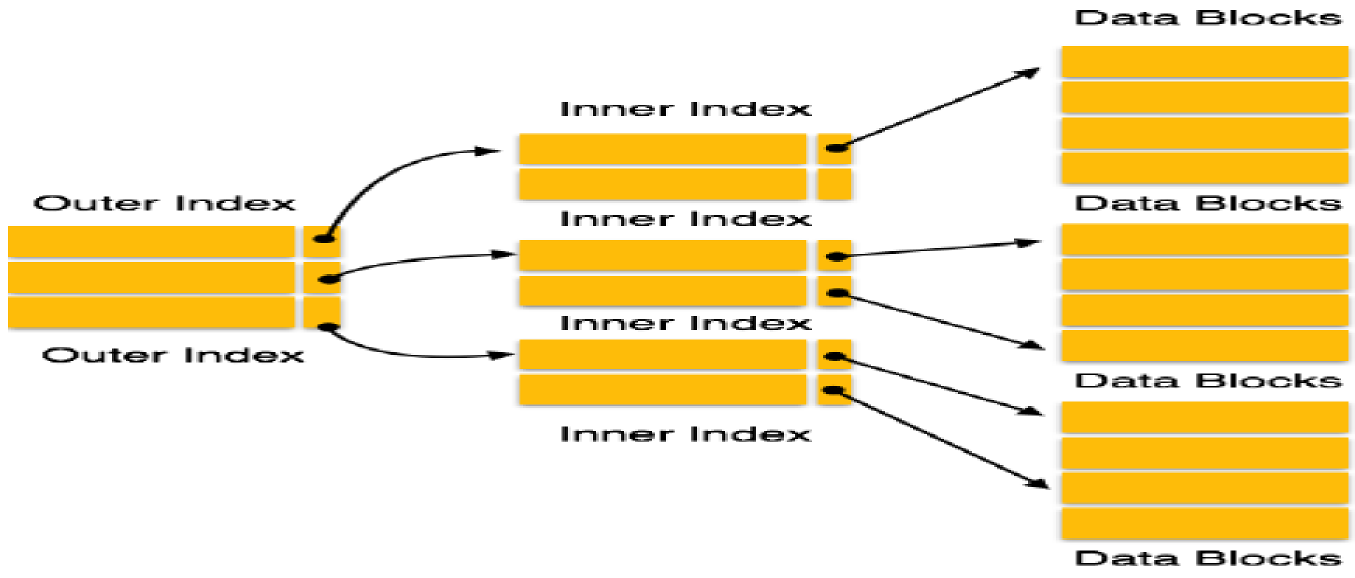
Sparse Index

In sparse index, index records are not created for every search key. An index record here contains a search key and an actual pointer to the data on the disk. To search a record, we first proceed by index record and reach at the actual location of the data. If the data we are looking for is not where we directly reach by following the index, then the system starts sequential search until the desired data is found.



Multilevel Index

Index records comprise search-key values and data pointers. Multilevel index is stored on the disk along with the actual database files. As the size of the database grows, so does the size of the indices. There is an immense need to keep the index records in the main memory so as to speed up the search operations. If single-level index is used, then a large size index cannot be kept in memory which leads to multiple disk accesses.



Multi-level Index helps in breaking down the index into several smaller indices in order to make the outermost level so small that it can be saved in a single disk block, which can easily be accommodated anywhere in the main memory.

6.b Construct a B+ tree of order 5 by inserting the following keys :
12, 34,28, 43.27, 87,39,48, 6 M

OR

7.a What is normalization? Explain different types of normal forms from second normal form to BCNF. **7 M**

Normalization Definition---2M

Normalization is the process of efficiently organizing data in a database with two goals in mind
First goal: eliminate redundant data for example, storing the same data in more than one table
Second Goal: ensure data dependencies make sense for example, only storing related data in a table .

Advantages of Normal forms

Less storage space
Quicker updates
Less data inconsistency
Clearer data relationships
Easier to add data
Flexible Structure

Types of Normal Forms--5M

First Normal Form

‘A relation R is in first normal form (1NF) if and only if all underlying domains contain atomic values only.’

Second Normal Form

‘A relation R is in second normal form (2NF) if and only if it is in 1NF and every nonkey attribute is fully functionally dependent on the primary key.

Third Normal Form

‘A relation R is in third normal form (3NF) if and only if it is in 2NF and every nonkey attribute is nontransitively dependent on the primary key.
OR

A relation schema R is in third normal form (3NF) if, whenever a nontrivial functional dependency X->A holds in R, either (a) X is a superkey of R, or (b) A is a prime attribute of R.

BCNF

A relation schema R is in BCNF if whenever a nontrivial functional dependency X→A holds in R, then X is a super key of R.

7.b What is closure of Functional Dependency? Explain how to find it . 5 M

For each such set of attributes X, we determine the set X+ of attributes that are functionally determined by X based on F; X+ is called the closure of X under F. Algorithm can be used to calculate X+.

Algorithm for Determining X+, the Closure of X under F

Input: A set F of FDs on a relation schema R, and a set of attributes X, which is a subset of R.

```
X+ := X;
repeat
oldX+ := X+;
for each functional dependency Y→Z in F do
if X+ ⊇ Y then X+ := X+ ∪ Z;
until (X+ = oldX+);
```

UNIT IV

8.a What are the ACID properties of a transaction? Describe each. 6 M

Transactions should possess several properties, often called the **ACID** properties; they should be enforced by the concurrency control and recovery methods of the DBMS. The following are the ACID properties:

■ **Atomicity.** A transaction is an atomic unit of processing; it should either be performed in its entirety or not performed at all.

■ **Consistency preservation.** A transaction should be consistency preserving, meaning that if it is completely executed from beginning to end without interference from other transactions, it should take the database from one consistent state to another.

■ **Isolation.** A transaction should appear as though it is being executed in isolation from other transactions, even though many transactions are executing concurrently. That is, the execution of a transaction should not be interfered with by any other transactions executing concurrently.

■ **Durability or permanency.** The changes applied to the database by a committed transaction must persist in the database. These changes must not be lost because of any failure.

8.b What is a schedule? Explain different types of schedules based on recoverability. 6 M

Schedule Definition---1M
Recoverability Explanation--5M

Schedule Definition

When transactions are executing concurrently in an interleaved fashion, then the order of execution of operations from all the various transactions is known as a schedule (or history).

Characterizing schedules based on recoverability

For some schedules it is easy to recover from transaction and system failures, whereas for other schedules the recovery process can be quite involved. In some cases, it is even not possible to recover correctly after a failure. Hence, it is important to characterize the types of schedules for which recovery is possible, as well as those for which recovery is relatively simple.

First, we would like to ensure that, once a transaction T is committed, it should never be necessary to roll back T. This ensures that the durability property of transactions is not violated. The schedules that theoretically meet this criterion are called recoverable schedules; those that do not are called non recoverable and hence should not be permitted by the DBMS.

The definition of **recoverable schedule** is as follows: A schedule S is recoverable if no transaction T in S commits until all transactions T_ that have written some item X that T reads have committed. A transaction T reads from transaction T_ in a schedule S if some item X is first written by T_ and later read by T. In addition, T_ should not have been aborted before T reads item X, and there should be no transactions that write X after T_ writes it and before T reads it (unless those transactions, if any, have aborted\ before T reads X).
In a recoverable schedule, no committed transaction ever needs to be rolled back, and so the definition of committed transaction as durable is not violated. However, it is possible for a phenomenon known as **cascading rollback** (or cascading abort) to occur in some recoverable schedules, where an uncommitted transaction has to be rolled back because it read an item from a transaction that failed.
A schedule is said to be **cascadeless, or to avoid cascading rollback**, if every transaction in the schedule reads only items that were written by committed transactions. In this case, all items read will not be discarded, so no cascading rollback will occur.
Finally, there is a third, more restrictive type of schedule, called a **strict schedule,** in which transactions can neither read nor write an item X until the last transaction that wrote X has committed (or aborted).

(OR)

9.a Describe how to control concurrency using locking. 6 M

In a multiprogramming environment where multiple transactions can be executed simultaneously, it is highly important to control the concurrency of transactions. We have concurrency control protocols to ensure atomicity, isolation, and serializability of concurrent transactions. Concurrency control protocols can be broadly divided into two categories.

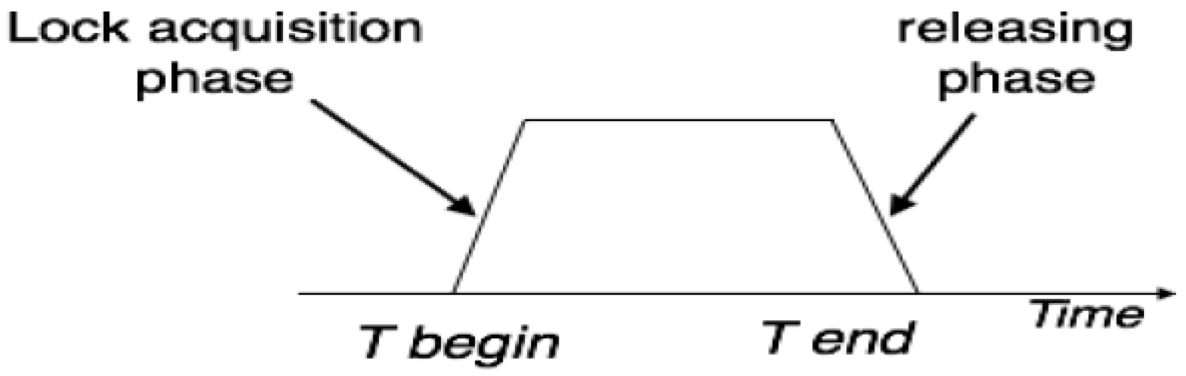
.

Lock-based Protocols
Database systems equipped with lock-based protocols use a mechanism by which any transaction cannot read or write data until it acquires an appropriate lock on it.
Locks are of two kinds .

- **Binary Locks** − A lock on a data item can be in two states; it is either locked or unlocked.
- **Shared/exclusive** − This type of locking mechanism differentiates the locks based on their uses. If a lock is acquired on a data item to perform a write operation, it is an exclusive lock. Allowing more than one transaction to write on the same data item would lead the database into an inconsistent state. Read locks are shared because no data value is being changed.

Two-Phase Locking 2PL

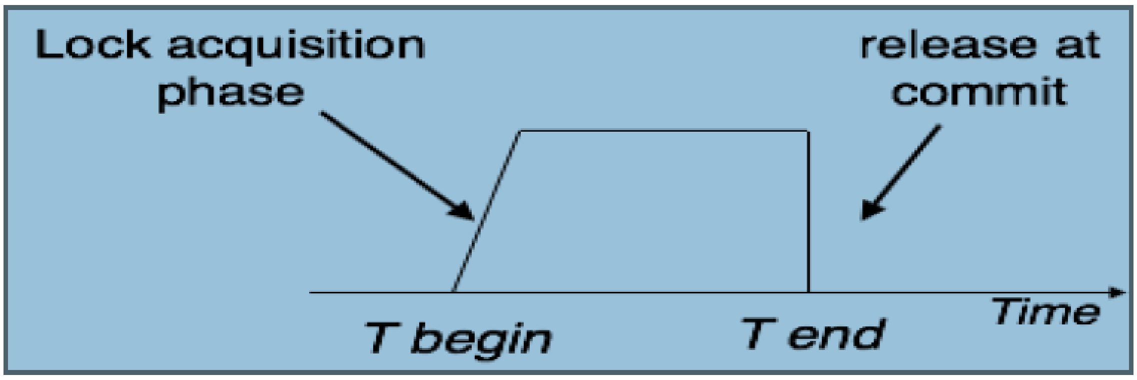
This locking protocol divides the excution phase of a transaction into three parts. In the first part, when the transaction starts executing, it seeks permission for the locks it requires. The second part is where the transaction acquires all the locks. As soon as the transaction releases its first lock, the third phase starts. In this phase, the transaction cannot demand any new locks; it only releases the acquired locks.



- Two-phase locking has two phases, one is growing, where all the locks are being acquired by the transaction; and the second phase is shrinking, where the locks held by the transaction are being released.
- To claim an exclusive (write) lock, a transaction must first acquire a shared (read) lock and then upgrade it to an exclusive lock.

Strict Two-Phase Locking

The first phase of Strict-2PL is same as 2PL. After acquiring all the locks in the first phase, the transaction continues to execute normally. But in contrast to 2PL, Strict-2PL does not release a lock after using it. Strict-2PL holds all the locks until the commit point and releases all the locks at a time.



9.b. Write in detail about mandatory access control. 6M

Mandatory Access Control (MAC) is is a set of security policies constrained according to system classification, configuration and authentication. MAC policy management and settings are established in one secure network and limited to system administrators.
MAC advantages and disadvantages depend on organizational requirements, as follows:

- MAC provides tighter security because only a system administrator may access or alter controls.
- MAC policies reduce security errors.
- MAC enforced operating systems (OS) delineate and label incoming application data, which creates a specialized external application access control policy.

Scheme prepared by **Signature of HOD, IT Dept**

Paper Evaluators:

Sno	Name of the College	Name of the Examiner	Signature