Q.2 a. Give the classifications of Data Models. (4)

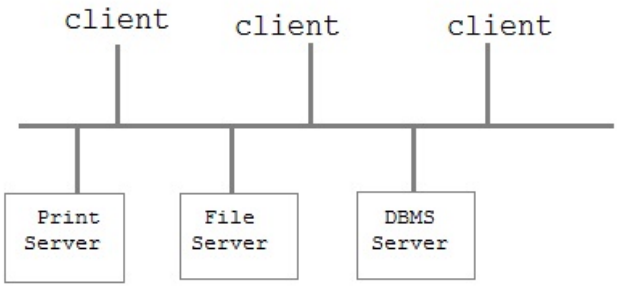
Answer:

Data models can be classified as follows:

1) Record-based Data Model
   a. Hierarchical data model
   b. Network data model
   c. Relational data model
2) Object-based Data Model
   a. Entity Relationship data model
   b. Object-oriented data model
   c. Semantic data model
   d. Functional data model
3) Physical Data Model
   a. Unifying data model
   b. Frame memory model

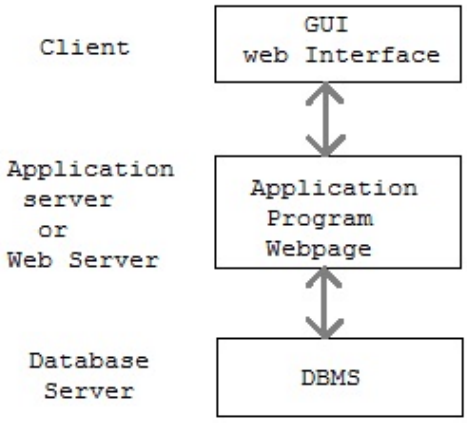
b. Explain Client/Server architecture of DBMS. (8)

Answer: Client / Server Architecture (Two-tier)



Two-tier Client / Server architecture is used for User Interface program and Application Programs that runs on client side. An interface called ODBC(Open Database Connectivity) provides an API that allow client side program to call the DBMS. Such clients are called **Data server**.

Client / Server Architecture (Three-tier)



Three-tier Client / Server database architecture is commonly used architecture for web applications. Intermediate layer called **Application server** or Web Server stores the web connectivity software and the business logic(constraints) part of application used to access the right amount of data from the database server.

c. What is schema? Explain with example. (4)

Answer:
Overall design of database is called database schema. It is a logical structure of database.
Example: The database consists of information about a set of customer and accounts and the relationship between them is an example of schema.

Q.3 a. Define following attributes. (4)
(i) Multi-valued attribute
(ii) Derived attribute

Answer:

(i). **Multi-valued attribute**

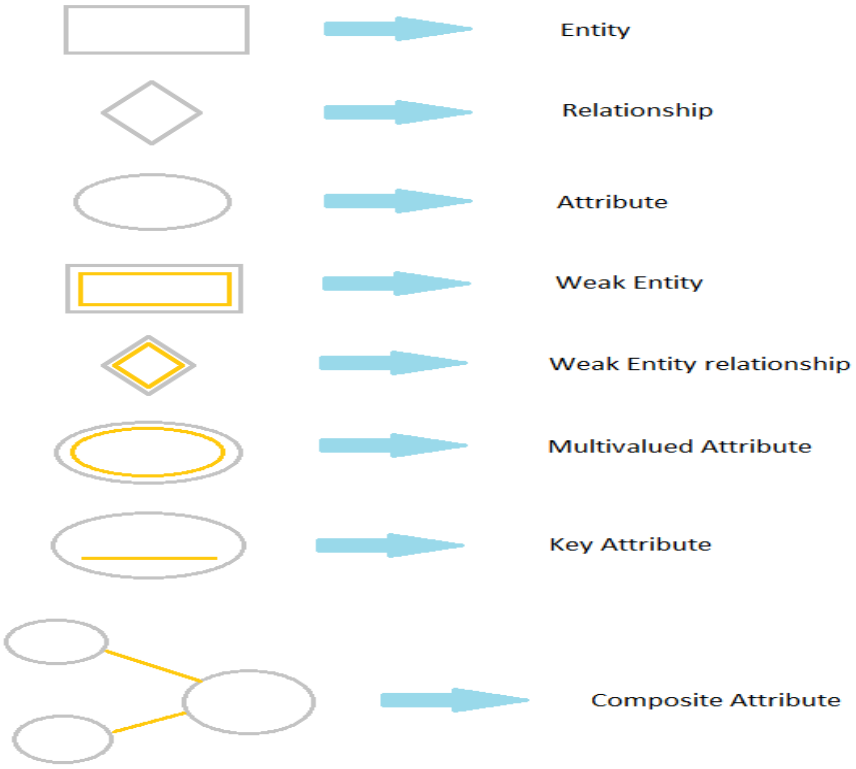
Multi-valued attributes may contain more than one values. For example, a person can have more than one phone number, email_address, etc.

(ii). **Derived attribute**

Derived attributes are the attributes that do not exist in the physical database, but their values are derived from other attributes present in the database. For example, age can be derived from data_of_birth.

b. List out E-R diagram symbols. (8)

Answer:



c. Define following. **(4)**
(i) Weak Entity set
(ii) Primary Key

Answer:

(i) **Weak Entity set**

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

(ii) **Primary Key**

A primary key is a special relational database table column (or combination of columns) designated to uniquely identify all table records.

Q.4 a. Define following. **(4)**
(i) Super Key
(ii) Candidate Key
(iii) Relation Key
(iv) Alternate Key

Answer:

(i). **SuperKey**

An attribute, or a set of attributes, that uniquely identifies a tuple within a relation is called SuperKey.

(ii). **Candidate Key**

If a relation contains more than one relation keys, then they each are called candidate key.

(iii). **Relation Key**

A Superkey for which no subset is a super key is called a relation key. It is a minimal super key.

(iv). **Alternate Key**

Candidate keys that are not selected to be a primary key are called Alternate Key.

b. What is transaction? Explain Commit and Rollback commands. **(4)**

Answer: A transaction is a set of database operations that performs a particular task.

Commit:

Commit can be used to reflect the changes in the database. Until you commit, you can only see how your work affects the tables.

Rollback:

The transactions like update, insert or delete can be undone with the help of rollback command.

c. Explain how insert, update and delete operations violate constraints? Discuss how to deal with these violations? (8)

Answer: The INSERT Operation:

The insert operation allows us to insert a new tuple in a relation. When we try to insert a new record, then any of the following four types of constraints can be violated:

- **Domain Constraint:**
  If the value given to an attribute lies outside the domain of that attribute.
- **Key Constraint:**
  If the value of the key attribute in the new tuple t is the same as in the existing tuple in relation R.
- **Entity Integrity Constraint:**
  If the primary key attribute value of new tuple is given as null.
- **Referential Integrity Constraint:**
  If the value of the foreign key in t refers to a tuple that does not appear in the referenced relation.

If the insertion violates one or more constraints then

- Insertion can be rejected and the reason for rejection can also be explained to the user by DBMS

or

- Ask the user to correct the data and resubmit, also give the reason for rejecting the insertion.

The Deletion Operation:

Using the delete operation some existing records can be deleted from a relation. To delete some specific records from the database a condition is also specified based on which records can be selected for deletion.

Constraints that can be violated during deletion:

Only one type of constraint can be violated during deletion, it is referential integrity constraint. It can occur when delete a record in the table where it is referenced by the foreign key of another table.

If the deletion violates referential integrity constraint, then

- Reject the deletion
- Attempt to cascade the deletion by deleting tuples that references the tuple that is being deleted.
- Change the value of referencing attribute that causes the violation.

The Update Operation:

Update operations are used for modifying database values.

The constraints violations faced by this operation are logically the same as the problem faced by Insertion and Deletion operations.

Q.5 a. Explain SELECT and PROJECT unary relation operations. (8)

Answer:

Select Operation (σ) : It selects tuples that satisfy the given predicate from a relation.

Notation – $\sigma_p(r)$
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 – $=, \neq, \geq, <, >, \leq$.

For example –

1) Selects tuples from books relation where subject is 'database'.

$\sigma_{subject = "database"}(Books)$

2) Selects tuples from books relation where subject is 'database' and 'price' is 450.

$\sigma_{subject = "database" \text{ and } price = "450"}(Books)$

Project Operation (∏): It projects column(s) that satisfy a given predicate.

Notation – $\prod_{A1, A2, An}(r)$
Where A_1, A_2 , A_n are attribute names of relation **r**.

For example –

1) List out subject and author from the relation Books.

$\prod_{subject, author}(Books)$

b. List out different JOIN operations. Explain Outer Join. (8)

Answer:

The following are the types of JOIN

(1) Inner Join or Equi Join
(2) Outer Join
  a. Left Outer
  b. Right Outer
  c. Full Outer
(3) Cross Join or Cartesian Product

Outer Join:

- Three types of outer joins:
  1. Left Outer Join ⟕ includes all tuples in the left hand relation and includes only those matching tuples from the right hand relation.

2. Right Outer Join ⟖ includes all tuples in the right hand relation and includes only those matching tuples from the left hand relation.
3. Full Outer Join ⟗ includes all tuples in the left hand relation and from the right hand relation.

- Examples:

Assume we have two relations: PEOPLE and MENU:

PEOPLE:

Name	Age	Food
Alice	21	Hamburger
Bill	24	Pizza
Carl	23	Beer
Dina	19	Shrimp

MENU:

Food	Day
Pizza	Monday
Hamburger	Tuesday
Chicken	Wednesday
Pasta	Thursday
Tacos	Friday

- PEOPLE $\bowtie_{people.food = menu.food}$ MENU

Name	Age	people.Food	menu.Food	Day
Alice	21	Hamburger	Hamburger	Tuesday
Bill	24	Pizza	Pizza	Monday
Carl	23	Beer	*NULL*	*NULL*
Dina	19	Shrimp	*NULL*	*NULL*

- PEOPLE $⟕_{people.food = menu.food}$ MENU

Name	Age	people.Food	menu.Food	Day
Bill	24	Pizza	Pizza	Monday
Alice	21	Hamburger	Hamburger	Tuesday
NULL	*NULL*	*NULL*	Chicken	Wednesday
NULL	*NULL*	*NULL*	Pasta	Thursday
NULL	*NULL*	*NULL*	Tacos	Friday

- PEOPLE $\bowtie_{\text{people.food = menu.food}}$ MENU

Name	Age	people.Food	menu.Food	Day
Alice	21	Hamburger	Hamburger	Tuesday
Bill	24	Pizza	Pizza	Monday
Carl	23	Beer	*NULL*	*NULL*
Dina	19	Shrimp	*NULL*	*NULL*
NULL	*NULL*	*NULL*	Chicken	Wednesday
NULL	*NULL*	*NULL*	Pasta	Thursday

Q.6 a. What is constraint? Explain referential integrity constraint with example (8)

Answer:

A Constraint is a rule that restricts the values that may be present in the database.

Referential integrity is a relational database concept in which multiple tables share a relationship based on the data stored in the tables, and that relationship must remain consistent. A foreign key is a way to enforce referential integrity within your database.

Syntax:

The syntax for creating a foreign key using a CREATE TABLE statement is:

Master Table:

```
CREATE TABLE supplier
( supplier_id numeric(10) Primary Key,
  supplier_name varchar2(50) not null,
  contact_name varchar2(50),
 );
```

Detail Table:

```
CREATE TABLE products
( product_id numeric(10) not null,
  supplier_id numeric(10) not null,
  FOREIGN KEY (supplier_id)REFERENCES supplier(supplier_id)
);
```

b. What is VIEW? Write syntax of create view command. (4)

Answer:

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.

Syntax:

```
CREATE VIEW view_name AS
SELECT column1, column2.....
FROM table_name
WHERE [condition];
```

c. List out domain integrity constraint and give syntax of creating check constraint. (4)

Answer:

Domain Integrity constraints:
(1) Not Null (2) Check

Syntax of creating check constraint:

Column Level: columnname datatype(size) check (condition)

Table Level:
```
Create Table Tablename
(Column1 datatype(size),
Column2 datatype(size),
---------------------------
Check (columnname condition);
```

Q.7 a. What is functional dependency? Explain Full Functional dependency with Diagram. (4)

Answer:

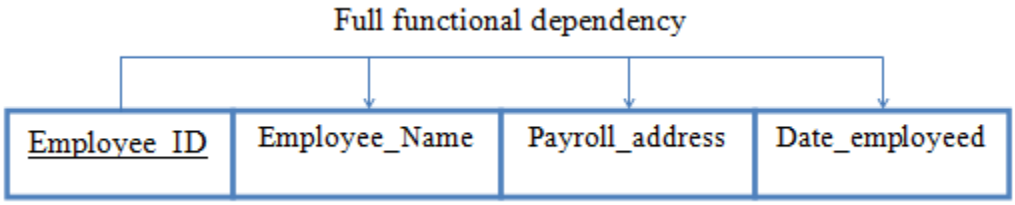
Functional dependency (FD): A functional dependency occurs when one attribute in a relation uniquely determines another attribute. This can be written A -> B which would be the same as stating "B is functionally dependent upon A."

Fully Functional Dependence (FFD) is defined, as Attribute Y is FFD on attribute" X, if it is FD on X and not FD on any proper subset of X. For example, in relation Supplier, different cities may have the same status.

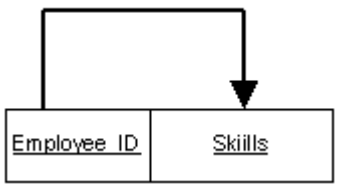
Given a relation R and Functional Dependency X→Y

Y is fully functionally dependent on X and there should not be any Z→Y, Where Z is a proper subset of X.

The employee table has following dependencies which are shown in diagram.



The EmpSkills table has following dependencies which are also shown here.



b. What is Normalization? State the criteria for the 1NF. (4)

Answer:

Database normalization (or normalisation) is the process of organizing the columns (attributes) and tables (relations) of a relational database to minimize data redundancy.

Normalization is a systematic approach of decomposing tables to eliminate data redundancy and undesirable characteristics like Insertion, Update and Deletion Anomalies.

Normalization is used for mainly two purpose,

1. Eliminating reduntant (useless) data.
2. Ensuring data dependencies make sense i.e data is logically stored.

Criteria for the 1NF

- No two Rows of data must contain repeating group of information i.e each set of column must have a unique value, such that multiple columns cannot be used to fetch the same row.
- Each table should be organized into rows, and each row should have a primary key that distinguishes it as unique.

c. Explain 2NF and 3NF with example. (8)

Answer:

2NF

There must not be any partial dependency of any column on primary key. It means that for a table that has concatenated primary key, each column in the table that is not part of the primary key must depend upon the entire concatenated key for its existence.
If any column depends only on one part of the concatenated key, then the table fails Second normal form.
Criteria for 2NF

- It must be in the first normal form
- Every non-prime attribute of the relation is fully functionally dependent on primary key.

In following example there are two rows for Adam, to include multiple subjects that he has opted for. In this the candidate key is {Student, Subject}, Age of Student only depends on Student column, which is incorrect as per Second Normal Form.

Student	Age	Subject
Adam	15	Biology
Adam	15	Maths
Alex	14	Maths
Stuart	17	Maths

To achieve second normal form, it would be helpful to split out the subjects into an independent table, and match them up using the student names as foreign keys.

New Student Table following 2NF will be :

Student	Age
Adam	15
Alex	14
Stuart	17

In Student Table the candidate key will be **Student** column, because all other column i.e **Age** is dependent on it.

New Subject Table introduced for 2NF will be:

Student	Subject
Adam	Biology
Adam	Maths
Alex	Maths
Stuart	Maths

In Subject Table the candidate key will be {Student, Subject} column. Now, both the above tables qualifies for Second Normal Form and will never suffer from Update Anomalies.

3NF

A database is in third normal form if it satisfies the following conditions:

- It is in second normal form
- There is no transitive functional dependency

By transitive functional dependency, we mean we have the following relationships in the table: A is functionally dependent on B, and B is functionally dependent on C. In this case, C is transitively dependent on A via B.

For example, consider a table with following fields.

Student_Detail Table :

Student_id	Student_name	DOB	Street	city	State	Zip

In this table Student_id is Primary key, but street, city and state depends upon Zip. The dependency between zip and other fields is called **transitive dependency**. Hence to apply **3NF**, we need to move the street, city and state to new table, with **Zip** as primary key.

New Student_Detail Table :

Student_id	Student_name	DOB	Zip

Address Table :

Zip	Street	city	state

Q.8 a. What is decomposition? Explain lossy and loseless properties of Decomposition. (8)

Answer:

Decomposition is the process of breaking down given relation into two or more relations.

Consider following table:

Account:

ano	balance	bname
A01	5000	vvn
A02	5000	ksad

Lossy Decomposition: The decomposition of relation R into R1 and R2 is lossy when join of R1 and R2 does not yield the same relation as in R.

Acc_bal :

ano	balance
A01	5000
A02	5000

Bal_branch:

balance	bname
5000	vvn
5000	kasd

Acc_Joined

ano	balance	bname
A01	5000	vvn
A01	5000	kasd
A02	5000	vvn
A02	5000	kasd

Lossless Decomposition: The decomposition of relationR into R1 and R2 is lossless when the join of R1 and R2 yields the same relation as in R.

Acc_bal :

ano	balance
A01	5000
A02	5000

Bal_branch:

balance	bname
A01	vvn
A02	kasd

Acc_Joined:

ano	balance	bname
A01	5000	vvn
A02	5000	kasd

b. What is Multi Valued Dependency? Explain 4NF with example. (8)

Answer:

In database theory, a multivalued dependency is a full constraint between two sets of attributes in a relation. In contrast to the functional dependency, the multivalued dependency requires that certain tuples be present in a relation.

4NF:

Eliminate redundancy due to multiplicative effect of MVD's.
Roughly: treat MVD's as FD's for decomposition, but not for finding keys.
Formally: R is in Fourth Normal Form if whenever MVD $X \rightarrow\rightarrow Y$ is *nontrivial*(Y is not a subset of X, and $X \cup Y$ is not all attributes), then X is a superkey.
Decompose R, using
4NF violation $X \rightarrow\rightarrow Y$,
into
XY and $X \cup (R—Y)$.

Example:

Drinkers(name, addr, phones, beersLiked)

- FD: name → addr
- Nontrivial MVD's: name →→ phones andname →→ beersLiked.
- Only key: {name, phones, beersLiked}
- All three dependencies above violate 4NF.
- Successive decomposition yields 4NF relations:

D1(name, addr)
D2(name, phones)
D3(name, beersLiked)

Q.9 a. Explain ACID properties of transactions. (8)

Answer:

Atomicity

The atomicity property identifies that the transaction is atomic. An atomic transaction is either fully completed, or is not begun at all.

Consistency

The database must remain in a consistent state after any transaction. No transaction should have any adverse effect on the data residing in the database. If the database was in a consistent state before the execution of a transaction, it must remain consistent after the execution of the transaction as well.

Isolation

In a database system where more than one transaction are being executed simultaneously and in parallel, the property of isolation states that all the transactions will be carried out and executed as if it is the only transaction in the system. No transaction will affect the existence of any other transaction.

Durability

A transaction is durable in that once it has been successfully completed, all of the changes it made to the system are permanent. There are safeguards that will prevent the loss of information, even in the case of system failure.

b. Explain conflict Serializability. (8)

Answer:

Conflict Serializability

Instructions Ii and Ij, of transactions Ti and Tj respectively, conflict if and only if there exists some item Q accessed by both Ii and Ij, and at least one of these instructions wrote Q.

1. Ii=read(Q),Ij=read(Q).Ii and Ij don't conflict.
2. Ii=read(Q),Ij=write(Q). They conflict.
3. Ii=write(Q),Ij=read(Q). They conflict.
4. Ii=write(Q),Ij=write(Q). They conflict.

Intuitively, a conflict between Ii and Ij forces a (logical) temporal order between them. If Ii and Ij are consecutive in a schedule and they do not conflict, their results would remain the same even if they had been interchanged in the schedule.

Example of a schedule that is not conflict serializable :

T3	T4
read(Q)	
	write(Q)
write(Q)	

We are unable to swap instructions in the above schedule toobtain either the serial schedule <T3,T4>, or the serialschedule <T4,T3>.

TEXT-BOOK

I. Fundamentals of Database Systems, Ramez Elmasri, Shamkant B Navathe, 5th edition, Pearson Eductiaon, 2008.