## Explain what is meant by repetition of information and inability to represent information.

**Repetition of information** is a condition in a relational database where the values of one attribute are determined by the values of another attribute in the same relation, and both values are repeated throughout the relation. This is a bad relational database design because it increases the storage required for the relation and it makes updating the relation more difficult.

**Inability to represent information** is a condition where a relationship exists among only a proper subset of the attributes in a relation. This is bad relational database design because all the unrelated attributes must be filled with null values otherwise a tuple without the unrelated information cannot be inserted into the relation.

## How data is represented in object oriented databases? Explain.

1) **Objects:** Basic units of storage that encapsulate both data and behaviour.
2) **Object Identifier (OID):** A unique identifier assigned to each object, ensuring it can be distinctly referenced and retrieved.
3) **Type Constructors:**
   - **Atoms:** Basic data types such as integers, strings, and booleans.
   - **Tuples:** Ordered collections of elements, similar to records or rows in relational databases.
   - **Sets:** Unordered collections of distinct elements.
   - **Lists:** Ordered collections of elements, which can contain duplicates.
   - **Bags:** Unordered collections that allow duplicates.
   - **Arrays:** Indexed collections of elements, typically of a fixed size.

## What are entities? Define various types of mapping between entities.

Entity is a real time object that can be distinguished from other objects.

Various mapping between entities:

- **One-to-One (1:1) Mapping:** Each entity in one entity set is associated with exactly one entity in another entity set, and vice versa.
- **One-to-Many (1:N) Mapping:** Each entity in one entity set is associated with one or more entities in another entity set, but each entity instance in the second entity set is associated with exactly one entity in the first entity set.
- **Many-to-One (N:1) Mapping:** The reverse of one-to-many mapping. Each entity in one entity set is associated with exactly one entity in another entity set, but each entity in the second entity set can be associated with one or more entities in the first entity set.
- **Many-to-Many (N:M) Mapping:** Each entity in one entity set can be associated with one or more entities in another entity set, and vice versa.

Explain how complex data is represented in object oriented databases. How object relational databases differ from object oriented databases.

Representation of complex data:

**Type Constructors:**

- **Atoms:** Basic data types such as integers, strings, and booleans.
- **Tuples:** Ordered collections of elements, similar to records or rows in relational databases.
- **Sets:** Unordered collections of distinct elements.
- **Lists:** Ordered collections of elements, which can contain duplicates.
- **Bags:** Unordered collections that allow duplicates.
- **Arrays:** Indexed collections of elements, typically of a fixed size.

**User defined data type:** They allow developers to define custom data type with multiple attributes.

Example:

```
CREATE TYPE Address AS      (
      Addressline1          Char(20)
      Addressline2          Char(20)
      City                  Char(12)
      State                 Char(15)
      Pincode               Char(6)
      ) ;
```

| **OODBMS** | **ORDBMS** |
|---|---|
| It stands for Object Oriented Database Management System. | It stands for Object Relational Database Management System. |
| Object-oriented databases, like Object Oriented Programming, represent data in the form of objects and classes. | An object-relational database is one that is based on both the relational and object-oriented database models. |
| OODBMSs support ODL/OQL. | ORDBMS adds object-oriented functionalities to SQL. |

| OODBMS | ORDBMS |
|---|---|
| Every object-oriented system has a different set of constraints<br><br>that it can accommodate. | Keys, entity integrity, and referential integrity are constraints of<br><br>an object-oriented database. |
| The efficiency of query processing is low. | Processing of queries is quite effective. |

Explain three phase commit protocols in distributed databases.

**Three phase commit protocol:**

1. **Prepare Phase:** It is same as the prepare phase of Two-phase Commit Protocol. In this phase every site should be prepared or ready to make a commitment.
2. **Prepare to Commit Phase:** In this phase the controlling site issues an "Enter Prepared State" message and sends it to all the slaves. In the response, all the slaves site issues an OK message.
3. **Commit/Abort Phase:** This phase consists of the steps which are same as they were in the two-phase commit. In this phase, no acknowledgement is provided after the process.

Explain the multidimensional data model used in data warehouses.

Working of multidimensional data model:

1. **Assembling data from the client:** It collects correct data from the client.
2. **Grouping different segments of the system:** It recognizes and classifies all the data to their respective sections.
3. **Noticing the different proportions:** The main factors are recognized according to the user's point of view. These factors are also known as "Dimensions".
4. **Preparing the actual-time factors and their respective qualities:** The factors which are recognized in the previous step are used further for identifying the related qualities. These qualities are also known as **"attributes"**.
5. **Finding the actuality of factors which are listed previously and their qualities:** It separates and differentiates the actuality from the factors which are collected by it.
6. **Building the Schema to place the data, with respect to the information collected from the steps above:** A schema is built on the basis of the data which was collected previously.

Explain the decision tree classifiers.

It is a tree-structured classifier, where **internal nodes represent the features of a dataset, branches represent the decision rules** and **each leaf node represents the outcome.**

**Algorithm for decision tree induction:**

**Input:** Set of training data records: $R_1$, $R_2$,...,$R_m$ and set of attributes: $A_1$, $A_2$,...,$A_n$

**Output:** Decision tree

```
procedure Build_tree (records, attributes);
begin
create a node N;
if all records belong to the same class, C then return N as a leaf node with class label C;
if attributes is empty then return N as a leaf node with class label C, such that the majority
of records belong to it;
select attribute $A_i$ (with the highest information gain) from attributes;
label node N with $A_i$;
for each known value, $v_j$ of $A_i$ do
    begin
        add a branch from node N for the condition $A_i = v_j$ ;
        $S_j$ = subset of records where $A_i = v_j$ ;
        if $S_j$ is empty then add a leaf, L, with class label C, such that the majority of
        records belong to it and return L
        else add the node returned by Build_tree($S_j$ , attributes – $A_i$);
    end;
end;
```