same database.
Consider the following example for declaring a record data type in Pascal Language
type customer = record
name: string;
street: string;
city: string;
end;
This defines a new record called customer with three fields. Each field has a name and
a type associated with it. A banking enterprise may have several such record types, including:
• account, with fields number and balance.
• employee, with fields name and salary.
 At the physical level, a customer, account, or employee record can be
described as a block of consecutive storage locations (for example, words or
bytes).
 At the logical level, each such record is described by a type definition,
illustrated above, and the interrelationship among these record types is
defined.
 Finally, at the view level, several views of the database are defined. For
example, tellers in a bank see only that part of the database that has
information on customer accounts. They cannot access information concerning
salaries of employees........EXPLAINEASILY WITH SCENARIES

In database systems, **data abstraction** is used to simplify user interaction by hiding the complexity of data storage. There are three levels of abstraction: **physical**, **conceptual (logical)**, and **view levels**. Here's an explanation of these with real-life scenarios:

1. Physical Level (How Data is Stored)

- **What it is**: This is the lowest level of data abstraction. It describes how the data is physically stored, like the bytes, blocks, and memory locations used.
- **Scenario**: Imagine a library's catalog. At the physical level, the data about books (titles, authors, etc.) is stored in the library's server on a hard disk as sequences of bytes. You don't need to know how the data is exactly arranged in terms of bytes or blocks to find a book.

2. Conceptual (Logical) Level (What Data is Stored and How it Relates)

- **What it is**: This is the next higher level of abstraction. It focuses on what data is stored in the database and the relationships between them, without worrying about how it's physically stored.
- **Scenario**: In the library, the conceptual level defines that each book has a title, author, publication date, and category. The system stores books and also relationships like which books are currently checked out. For example, if a librarian needs to track who borrowed which book, this relationship is described at the conceptual level.

3. View Level (How Users See the Data)

- **What it is**: The highest level of abstraction. It focuses on presenting a specific part of the data to the user depending on their needs. Different users might only see a portion of the overall data.
- **Scenario**: In the library, users (readers) don't need to see how the data is stored or the

book is available or not. However, a librarian may also want to see which member has borrowed which book and when it’s due back, which is a different view of the data.

Example: Bank Database

Consider a **banking system** as an example of these levels:

- **Physical Level**: At the physical level, customer details (like name, account balance) and employee details (like salary) are stored in memory as blocks of data (bits and bytes). This level handles how this data is stored in the bank's server.
- **Logical Level**: At the logical level, we define what data is stored. For example, we store customer records (name, address, account number), account records (account number, balance), and employee records (name, salary). We also define relationships like which customer owns which account.
- **View Level**: The bank tellers, for example, only need to see customer information to manage transactions, so they only have access to customer accounts. On the other hand, the bank manager can also view employee salary details.

In short, each level of abstraction hides unnecessary details from the user to make interaction with the system simpler and more focused on what they need to see.