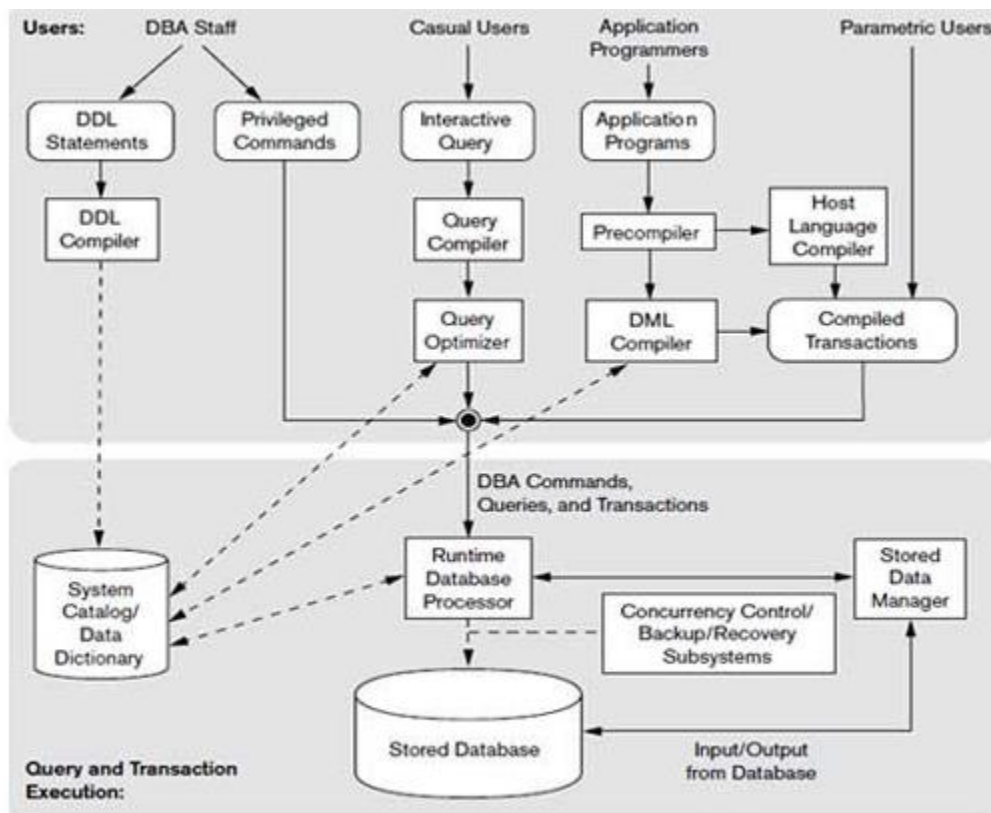


## IAT-1 SOLUTION

### 1. Define database. Elaborate component modules of DBMS and their interaction with diagram.

**Ans:** A database is a collection of related data. A data mean known facts that can be recorded and that have implicit meaning. For eg, consider the names, telephone numbers, and addresses of the people you know. You may have recorded this data in an indexed address book or you may have stored it on a hard drive, using a personal computer and software such as Microsoft Access or Excel. This collection of related data with an implicit meaning is a database.

#### component modules of DBMS:



The figure is divided into two parts. The top part of the figure refers to the various users of the database environment and their interfaces. The lower part shows the internals of the DBMS responsible for storage of data and processing of transactions. The database and the DBMS catalog are usually stored on disk. Access to the disk is controlled primarily by the operating

system (OS), which schedules disk read/write. Many DBMSs have their own buffer management module to schedule disk read/write. Stored data manager controls access to DBMS information that is stored on disk, whether it is part of the database or the catalog.

**DBA staff:** The DDL compiler processes schema definitions, specified in the DDL, and stores descriptions of the schemas (meta-data) in the DBMS catalog. The catalog includes information such as the names and sizes of files, names and data types of data items, storage details of each file, mapping information among schemas, and constraints.

**Casual users:** interact using some form of interface, which we call the interactive query interface. queries are parsed and validated for correctness of the query syntax, the names of files and data elements, and so on by a query compiler that compiles them into an internal form. This internal query is subjected to query optimization. query optimizer is concerned with the rearrangement and possible reordering of operations, elimination of redundancies, and use of correct algorithms and indexes during execution. It consults the system catalog for statistical and other physical information about the stored data and generates executable code that performs the necessary operations for the query and makes calls on the runtime processor.

**Application programmers :** write programs in host languages such as Java, C, or C++ that are submitted to a precompiler. precompiler extracts DML commands from an application program. commands are sent to the DML compiler for compilation. rest of the program is sent to the host language compiler. The object codes for the DML commands and the rest of the program are linked, forming a canned transaction. An example is a bank withdrawal transaction where the account number and the amount may be supplied as parameters.

In the lower part of Figure, the runtime database processor executes

- (1) the privileged commands
- (2) the executable query plans, and
- (3) the canned transactions with runtime parameters.

It works with the system catalog and may update it with statistics. It also works with the stored data manager, which in turn uses basic operating system services for carrying out low-level input/output (read/write).

operations between the disk and main memory. The runtime database processor handles other aspects of data transfer, such as management of buffers, concurrency control, and backup and recovery systems, integrated into the working of the runtime database processor for purposes of transaction management.

**Database System Utilities:** DBMSs have database utilities that help the DBA manage the database system. Common utilities have the following types of functions:

### **Loading**

- - - used to load existing data files
- such as text files or sequential files
- into the database automatically reformats the data and stores it in the database for loading programs, conversion tools are available like IDMS (Computer Associates), SUPRA (Cincom), and IMAGE (HP)

**Backup** - creates a backup copy of the database, by dumping the entire database onto tape - - used to restore the database in case of catastrophic disk failure. Incremental backups are also often used, to save space.

**Database storage reorganization** - used to reorganize a set of database files into different file organizations to improve performance.

**Performance monitoring** - - monitors database usage and provides statistics to the DBA. Statistics used for making decisions. Other utilities may be available for sorting files, handling data compression, monitoring access by users, interfacing with the network, and performing other functions.

**Tools, Application Environments, and Communications Facilities:** CASE tools are used in the design phase of database systems. Data dictionary (or data repository) system for storing catalog information about schemas and constraints. Information repository stores information such as design decisions, usage standards, application program descriptions, and user information. Application development environments provide an environment for developing database applications, including database design, GUI development, querying and updating, and application program development. Communications software, whose function is to allow users at locations remote from the database system site to access the database through computer terminals, workstations, or personal computers. These are connected to the database site through data communications hardware such

as Internet routers, phone lines, long-haul networks, local networks, or satellite communication devices. The integrated DBMS and data communications system is called a DB/DC system.

Top half figure:

it shows interfaces for the DBA staff, casual users who work with interactive interfaces to formulate queries, application programmers who create programs using some host programming languages, parametric users who do data entry work by supplying parameters to predefined transactions. The DBA staff works on defining the database and tuning it by making changes to its definition using the DDL and other privileged commands.

## **2. List and explain the advantages of using the DBMS approach.**

### **Draw and Explain different notations used in ER-Diagram**

**Ans: Advantages of DBMS approach:**

**1. Controlling Redundancy:** Data redundancy (such as tends to occur in the "file processing" approach) leads to wasted storage space, duplication of effort (when multiple copies of a datum need to be updated), and a higher likelihood of the introduction of inconsistency. On the other hand, redundancy can be used to improve performance of queries. Indexes, for example, are entirely redundant, but help the DBMS in processing queries more quickly. A DBMS should provide the capability to automatically enforce the rule that no inconsistencies are introduced when data is updated.

**2. Restricting Unauthorized Access:** A DBMS should provide a security and authorization subsystem, which the DBA uses to create accounts and to specify account restrictions. Then, the DBMS should enforce these restrictions automatically.

**3. Providing Persistent Storage for Program Objects:** Object-oriented database systems make it easier for complex runtime objects (e.g., lists, trees) to be saved in secondary storage so as to survive beyond program termination and to be retrievable at a later time.

**4. Providing Storage Structures and Search Techniques for Efficient Query Processing:** Database systems must provide capabilities for efficiently executing queries and updates. The query processing and optimization module of the DBMS is responsible for choosing an efficient query execution plan for each query based on the existing storage structures.

**5. Providing Backup and Recovery:** A DBMS must provide facilities for recovering from hardware or software failures. The backup and recovery subsystem of the DBMS is responsible for recovery. The recovery subsystem could ensure that the transaction is resumed from the point at which it was interrupted so that its full effect is recorded in the database.


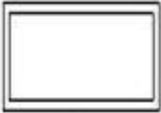





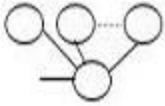

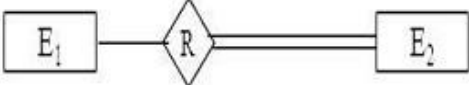

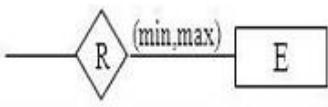
**6. Providing Multiple User Interfaces:** Many types of users with varying levels of technical knowledge use a database, a DBMS should provide a variety of user interfaces. For example, query languages for casual users, programming language interfaces for application programmers, forms and/or command codes for parametric users, menu-driven interfaces for stand alone users.

**7. Representing Complex Relationships Among Data:** A DBMS must have the capability to represent a variety of complex relationships among the data, to define new relationships as they arise, and to retrieve and update related data easily and efficiently.

**8. Enforcing Integrity Constraints:** Most database applications have certain integrity constraints that must hold for the data. The simplest type of integrity constraint involves specifying a data type for each data item. A more complex type of constraint that frequently occurs involves specifying that a record in one file must be related to records in other files. This is known as a referential integrity constraint. Another type of constraint specifies uniqueness on data item values, this is known as a key or uniqueness constraint.

**9. Permitting Inferencing and Actions Via Rules:** In a deductive database system, one may specify declarative rules that allow the database to infer new data! E.g., Figure out which students are on academic probation. Such capabilities would take the place of application programs that would be used to ascertain such information otherwise. Active database systems go one step further by allowing "active rules" that can be used to initiate actions automatically.

**Notations used in ER-Diagram:**

<u>Symbol</u>	<u>Meaning</u>
	ENTITY TYPE
	WEAK ENTITY TYPE
	RELATIONSHIP TYPE
	IDENTIFYING RELATIONSHIP TYPE
	ATTRIBUTE
	KEY ATTRIBUTE
	MULTIVALUED ATTRIBUTE
	COMPOSITE ATTRIBUTE
	DERIVED ATTRIBUTE
	TOTAL PARTICIPATION OF $E_2$ IN R
	CARDINALITY RATIO 1:N FOR $E_1:E_2$ IN R
	STRUCTURAL CONSTRAINT (min, max) ON PARTICIPATION OF E IN R

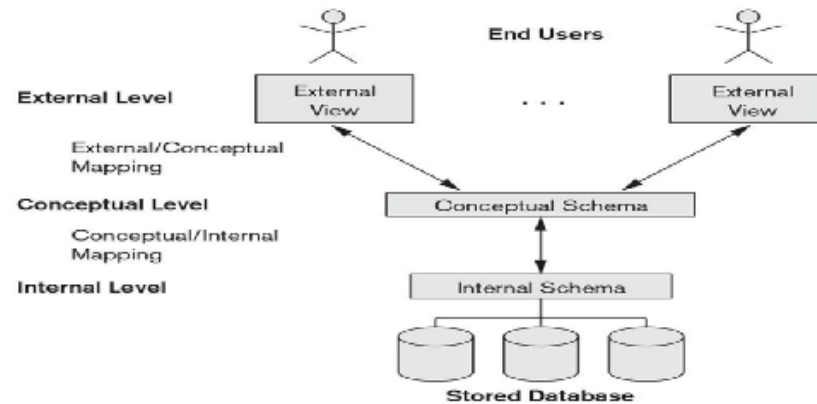
**3. Describe the three-schema architecture. Why do we need mappings**

**between schemas levels? Explain with necessary diagrams.**

**Ans:**

#### **The Three-Schema Architecture**

The goal of the three-schema architecture is to separate the user applications from the physical database.



In this architecture, schemas can be defined at the following three levels:

1. The internal level has an internal schema, -describes the physical storage structure of the database. -uses a physical data model and describes the complete details of data storage and access paths for the database.

2. The conceptual level has a conceptual schema, -describes the structure of the whole database for a community of users -hides the details of physical storage structures and concentrates on describing entities, data types, relationships, user operations, and constraints -representational data model is used to describe the conceptual schema when a database system is implemented -This implementation conceptual schema is often based on a conceptual schema design in a high-level data model.

3. The external or view level includes a number of external schemas or user views, -describes the part of the database that a particular user group is interested in and hides the rest of the database from that user group. -As in the previous level, each external schema is typically implemented using a representational data model, possibly based on an external schema design in a high-level data model.

The DBMS must transform a request specified on an external schema into a request against the conceptual schema, and then into a request on the internal schema for processing over the stored database. If the request is a database retrieval, the data extracted from the stored database must be reformatted to match the user's external view. The processes of transforming

requests and results between levels are called mappings. These mappings may be time-consuming, so some DBMSs—especially those that are meant to support small databases—do not support external views. Even in such systems, however, a certain amount of mapping is necessary to transform requests between the conceptual and internal levels.

#### **4.What are database triggers? Explain with syntax and examples and how do they differ from stored procedures?**

**Ans:**

##### **Database triggers:**

A trigger is a procedure that runs automatically when a certain event occurs in the DBMS. In many cases it is convenient to specify the type of action to be taken when certain events occur and when certain conditions are satisfied. The CREATE TRIGGER statement is used to implement such actions in SQL.

##### **General form:**

```
CREATE TRIGGER <name>
BEFORE | AFTER | <events>
FOR EACH ROW |FOR EACH STATEMENT
WHEN (<condition>)
<action>
```

A trigger has three components

##### **1. Event:** When this event happens, the trigger is activated

- Three event types : Insert, Update, Delete
- Two triggering times: Before the event  
After the event



**2. Condition (optional):** If the condition is true, the trigger executes, otherwise skipped

**3. Action:** The actions performed by the trigger

When the **Event** occurs and **Condition** is true, execute the **Action**

**Create Trigger** ABC  
**Before Insert On**  
Students

This trigger is activated when an insert statement is issued, but before the new record is inserted

**Create Trigger** XYZ  
**After Update On** Students  
....

This trigger is activated when an update statement is issued and after the update is executed

### Example:

1) If the employee salary increased by more than 10%, then increment the rank field by 1.

In the case of **Update** event only, we can specify which columns

```
Create Trigger EmpSal  
Before Update Of salary On Employee  
For Each Row  
Begin  
  IF (:new.salary > (:old.salary * 1.1)) Then  
    :new.rank := :old.rank + 1;  
  End IF;  
End;  
/
```

The assignment operator has " := "

We changed the new value of *rank* field

Trigger	Stored Procedure
A block of SQL code that <b>automatically executes</b> on a specific event.	A block of SQL code that executes <b>only when called/invoked</b> .
<b>Automatically</b> when an event occurs (like INSERT, UPDATE, DELETE).	<b>Manually</b> called by a user, application, or another procedure.
Used to <b>enforce business rules</b> , maintain data integrity, or audit changes.	Used to <b>perform a task or process logic</b> when requested.

## 5.Explain different DDL,DML,TCL,DCL,and DQL statements used in SQL with examples

Ans:

### DDL-Data Definition Language

**1.CREATE:**To create a new database object like table, view, index, etc.

Ex:

```
CREATE TABLE Students (ID INT, Name VARCHAR(50));
```

**2.ALTER:**To modify an existing object (add, delete, or modify columns)

Ex:

```
ALTER TABLE Students ADD Age INT;
```

**3.DROP:**To delete a table completely from the database

Ex:

```
DROP TABLE Students;
```

**4.TRUNCATE:**To delete all data from a table without deleting the structure

Ex:

```
TRUNCATE TABLE Students;
```

**5.RENAME:**To change the name of a database object

Ex:

```
RENAME TABLE Students TO Learners;
```

### DML-Data Manipulation Language

**1.INSERT:**To add new records into a table

Ex:

```
INSERT INTO Students (ID, Name, Age) VALUES (1, 'John', 20);
```

**2.UPDATE:**To modify existing records in a table

Ex:

UPDATE Students SET Age = 21 WHERE ID = 1;

**3.DELETE:**To remove records from a table based on a condition

Ex:

DELETE FROM Students WHERE ID = 1;

### **DCL-Data Control Language**

**1.GRANT:**To give permissions to users on database objects

Ex:

GRANT SELECT, INSERT ON Students TO user1;

**2.REVOKE:**To take back permissions from users

Ex:

REVOKE INSERT ON Students FROM user1;

### **TCL-Transaction Control Language**

**1.COMMIT:**To save all changes made in the current transaction

Ex:

COMMIT;

**2.ROLLBACK:**To undo changes made in the current transaction

Ex:

ROLLBACK;

**3.SAVEPOINT:**To set a temporary point to which you can later rollback

Ex:

SAVEPOINT sp1;

### **DQL-Data Query Language**

**SELECT:**To retrieve data from one or more tables

Ex:

SELECT \* FROM Students;

### **6.Consider the Sailors-Boats-Reserves DB described**

**s (sid, sname, rating, age)**

**b (bid, bname, color)**

**r (sid, bid, date)**

**Write each of the following queries in SQL.**

**1. Find the colors of boats reserved by Alber.**

**2. Find all sailor ids of sailors who have a rating of at least 8 or reserved boat 103.**

**3. Find the names of sailors who have not reserved a boat whose name contains the string “storm”. Order the names in ascending order.**

**4. Find the sailor ids of sailors with age over 20 who have not reserved a boat whose name includes the string “thunder”.**

**Ans:**

```
1.SELECT DISTINCT b.color
FROM sailors s
JOIN reserves r ON s.sid = r.sid
JOIN boats b ON r.bid = b.bid
WHERE s.sname = 'Alber';
```

```
2.SELECT DISTINCT s.sid
FROM sailors s
LEFT JOIN reserves r ON s.sid = r.sid
WHERE s.rating >= 8 OR r.bid = 103;
```

```
3.SELECT DISTINCT s.sname
FROM sailors s
WHERE s.sid NOT IN (
    SELECT r.sid
    FROM reserves r
    JOIN boats b ON r.bid = b.bid
    WHERE b.bname LIKE '%storm%'
)
ORDER BY s.sname ASC;
```

```
4.SELECT DISTINCT s.sid
FROM sailors s
WHERE s.age > 20 AND s.sid NOT IN (
    SELECT r.sid
    FROM reserves r
    JOIN boats b ON r.bid = b.bid
    WHERE b.bname LIKE '%thunder%'
);
```

