

### 3/ Database Users:

- \* Users are differentiated by the way they expect to interact with the system.
- \* Application programmers: interact with system through DML calls.
- \* Sophisticated users: write specialized database applications that do not fit into the traditional data processing framework.
- \* Naive users: invoke one of the permanent applications programs that have been written previously.  
Ex: People accessing database over the web, bank-tellers, classical staff.

### 2/ Database Administrator:

Coordinates all the activities of the database system, the DBA has a good understanding of the enterprise's information resources & needs. DBA duties include:

- \* Schema definition.
- \* Storage structure & access method definition.
- \* Schema & physical organization modification.
- \* Granting users authority to access the database.
- \* Specifying integrity constraints
- \* Acting as liaison with users.
- \* Monitoring performance & responding to change in requirements.

### 3/ Transaction management:

A transaction is a collection of operation that perform a single logical function in a database application.

Ex: Transfer funds from one account to another.

- Transaction management component ensures that the DB remains in a consistent state despite system failures.
  - Concurrency control management controls the interaction among the concurrent transactions to ensure the consistency of the database
- Ex: Simultaneous withdrawls.

#### 4. Storage management:

Storage manager is a program module that provides the interface b/w low level data stored in the DB & the applications programs & queries submitted to the system.

- \* Storage manager is responsible for the following tasks:
  - Interaction with the file manager.
  - Efficient storing, retrieving & updating of data

#### Overall System Structure of DBMS:

A DBMS is divided into two modules:

#### → Query Processor:

QE's Components are

DDL interpreter: Interprets DDL statements & records the definition in data dictionary.

DML compiler: Converts DML statements into low level instruction  
Query evaluation: Executes low level instructions generated by DML compiler.

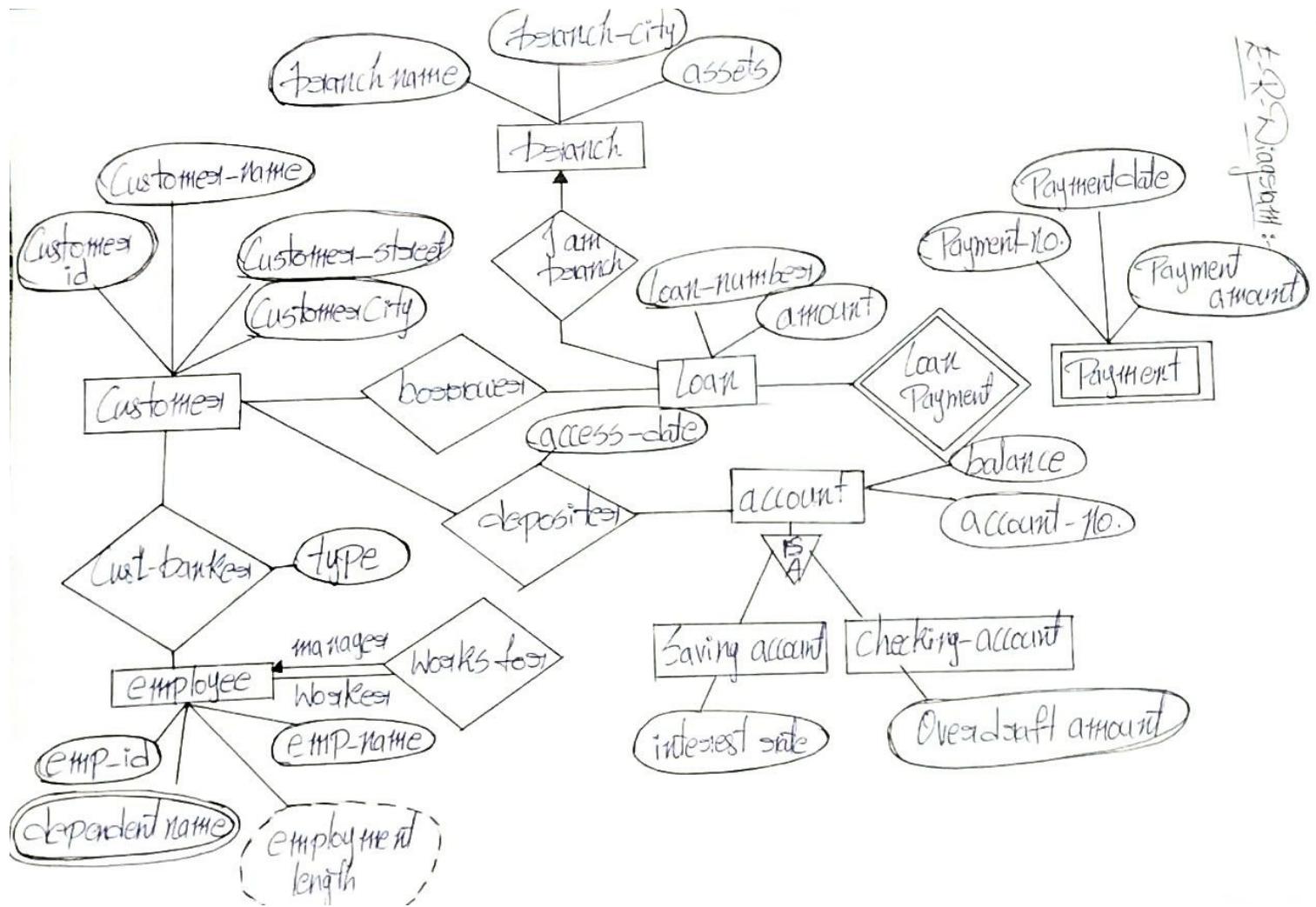
## → Storage manager:

It is a program module that provides the interface b/w low level data structure in the database & application programs & queries submitted to the system.

It is responsible for storing, retrieving & updating data in database.

Its components include authorization & integrity manager, transaction manager, file manager, buffer manager.

It implements multiple data structures like data files, dictionaries & indexes.



### 37. ① Key Constraints:

- \* Keys are the entity set used to identify an entity within the entity set uniquely.
- \* A primary key can contain a unique & null value in the relational table.

### ② Participation Constraints:

If specifies the existence of an entity when it is related to another entity in a relationship type

There are two types of Participation Constraints

\* Total Participation.

\* Partial Participation.

### ③ Class Hierarchies:

If is of two types - specialization & generalization.

Specialization is a process of identifying subsets of an entity that shares different characteristics. It breaks an entity into multiple entities from highest level to lower level.

Generalization is a process of extracting common properties from a set of entities & create a generalized entity from it.

### ④ Aggregation:

The relation b/w two entities is treated as a single entity relationship with it to corresponding entities is aggregated into a higher level entity.

4). Tuple relational calculus is a formal query language where variables represent tuples of a relation. The query specifies a set of conditions, & the result is a set of all tuples that satisfy those conditions.

It's often written in the form  $\{t / P(t)\}$ , where  $t$  is a tuple variable &  $P(t)$  is a predicate condition.

Ex: Find all customers in city "Delhi" in Customers table.

$\rightarrow \{t / \text{Customers}(t) \wedge \text{city} = "Delhi"\}$

- Domain relational calculus represents domain of a relation. The query specifies the attributions to be retrieved & the condition on those attributes.

It's written in the form  $\{x_1, x_2, \dots / P(x_1, x_2, \dots, x_n)\}$  where  $x_1, x_2$  are domain variables &  $P$  is the predicate.

Ex: Find the name & age of all employees in the "Sales" department.

$\rightarrow \{ \text{name}, \text{age} / \exists \text{empID}, \text{Salary} (\text{empID}, \text{name}, \text{age}) \in \text{Employee}$   
 $\quad \quad \quad \text{dept} = "Sales"\}$

5. Enforcing integrity constraints in RDBMS is usually for maintaining accuracy, consistency & reliability of data.

- Integrity constraints are rules that restrict the data that can be inserted, updated or deleted from a database.

## Primary Key Constraint:

Primary key uniquely identifies each record in a table.

Ex: Create table student (student-ID int Primary key, name varchar(255));

## Foreign Key Constraint:

A foreign key is a column or a set of columns in one table that refers to the primary key of another table.

It establishes a link or relationship between tables & enforces referential integrity.

Ex: Create table student (student-ID int Primary key, course name varchar(255), foreign key (course-ID) references course (course-ID)).

## Unique Constraint:

It ensures that all values in a column are unique.

It can also contain NULL values.

Ex: Create table employee (emp-ID int Primary key, email varchar(255) unique);

## Check Constraint:-

It ensures all values in a column satisfy a specific condition.

Ex: Create table products (product-ID int Primary key, price int(20), check (price > 0));