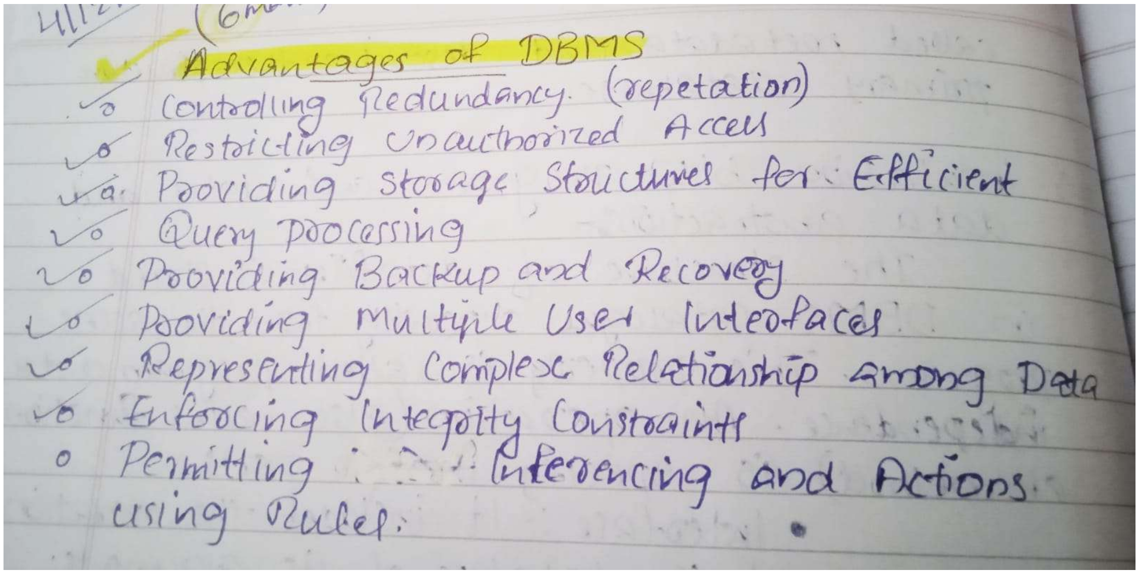
MODULE 1

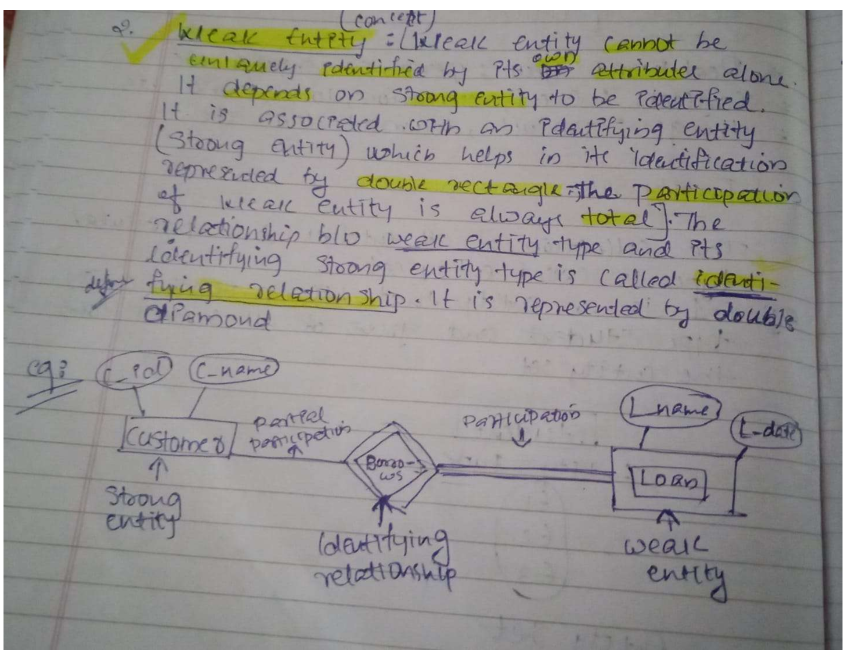
1 List any SIX major advantages of using a DBMS



Advantages of DBMS

- Controlling Redundancy. (repetation)
- Restricting Unauthorized Access
- Providing Storage Structures for Efficient Query processing
- Providing Backup and Recovery
- Providing Multiple User Interfaces
- Representing Complex Relationship among Data
- Enforcing Integrity Constraints
- Permitting Inferencing and Actions using Rules.

2 What is the concept of a weak entity used in data modelling? Define the terms owner entity type, Identifying relationship type.



Q. Weak entity (concept) : [Weak entity cannot be uniquely identified by its own attributes alone. It depends on strong entity to be identified. It is associated with an identifying entity (strong entity) which helps in its identification represented by double rectangle. The participation of weak entity is always total]. The relationship b/w weak entity type and its identifying strong entity type is called identifying relationship. It is represented by double diamond

eg:

Owner entity type:- It is an entity that independently exists and owns or identifies a weak entity in a DB. It depends on the owner entity for its identification. (usually through a primary key).

A bank may store the details of loans but the loan cannot exist without the customer so loan (dependant) will be a weak entity type and customer will be identifying entity type for dependant which means it is a strong entity type.

12 a) What is the difference between logical data independence and physical data independence? Which one is harder to achieve? Why?

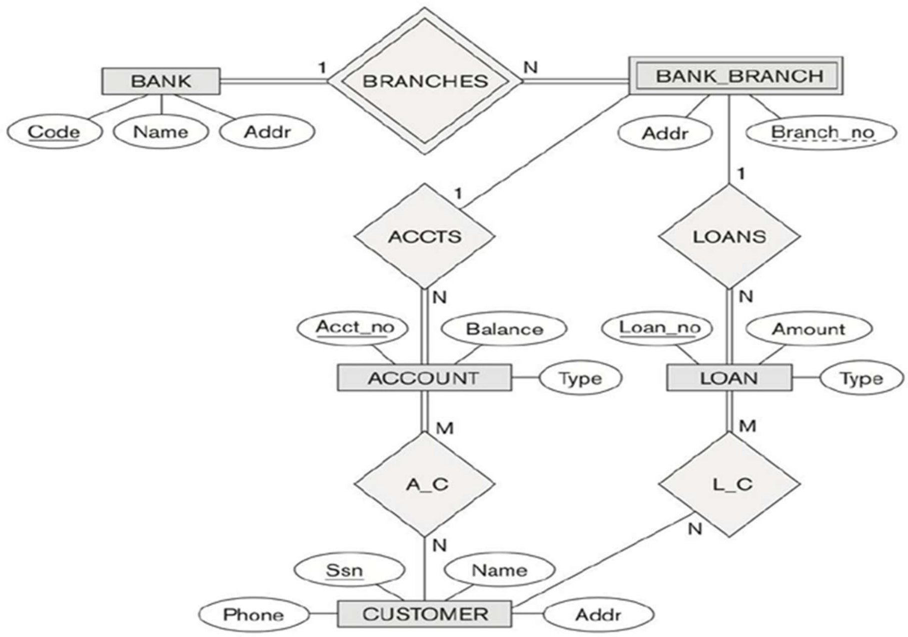
Difference

Physical Data Independence	Logical Data Independence
It mainly concern about how the data is stored into the system.	It mainly concerned about the structure or the changing data definition.
It is easy to retrieve.	It is difficult to retrieve because the data is mainly dependent on the logical structure of data.
As compared to the logical independence it is easy to achieve physical data independence.	As compared to the physical independence it is not easy to achieve logical data independence.
Any change at the physical level, does not require to change at the application level.	The change in the logical level requires a change at the application level.
The modifications made at the internal level may or may not be needed to improve the performance of the structure.	The modifications made at the logical level is significant whenever the logical structure of the database is to be changed.
It is concerned with the internal schema.	It is concerned with the conceptual schema.
Example: Change in compression techniques, Hashing algorithms and storage devices etc.	Example: Add/Modify or Delete a new attribute.

12/8/2025

👉 **Logical data independence is harder**, because changes in logical structure directly affect how data is viewed and used by applications.

b)



Consider the bank database given above and answer the following questions

i. List the strong (nonweak) entity types in the ER diagram.

ii. Is there a weak entity type? If so, give its name, partial key, and identifying relationship.

iii. What constraints do the partial key and the identifying relationship of the weak entity type specify in this diagram?

iv. List the names of all relationship types, and specify the (min, max) constraint on each participation of an entity type in a relationship type.

v. Suppose that every customer must have at least one account but is restricted to at most two loans at a time, and that a bank branch cannot have more than 1,000 loans. How does this show up on the (min, max) constraints?

Ans: i. Strong (non-weak) entity types

A strong entity type has a primary key that uniquely identifies its instances without relying on another entity. In this diagram, they are represented by single-bordered rectangles:

BANK

ACCOUNT

LOAN

CUSTOMER

ii. Weak entity type details

A weak entity is represented by a double-bordered rectangle.

Name: BANK_BRANCH

Partial Key: Branch_no (indicated by the dashed underline).

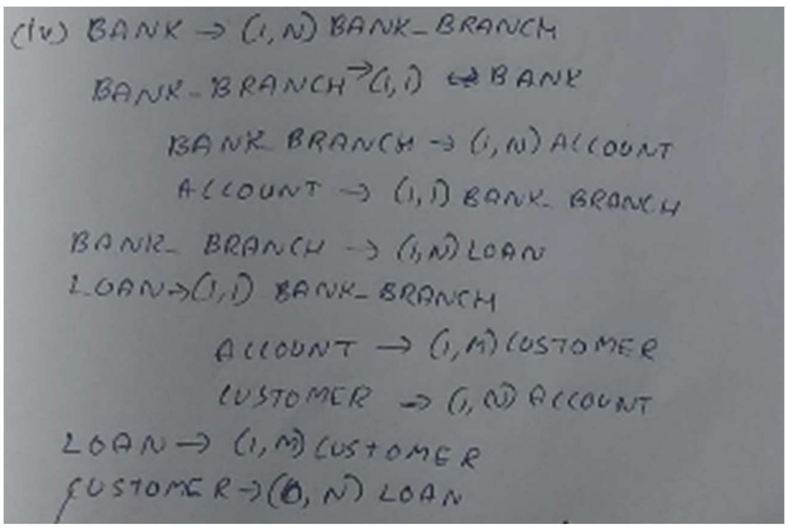
Identifying Relationship: BRANCHES (indicated by the double-bordered diamond).

iii. Constraints of the weak entity

Partial Key (Branch_no): This ensures that branch numbers are unique within a specific bank, but not across different banks.

Identifying Relationship (BRANCHES): This signifies total participation (indicated by the double line) and a dependency on the owner entity (BANK). A branch cannot exist in the system without an associated bank.

4.


(iv) BANK → (1, N) BANK_BRANCH
BANK-BRANCH → (1, 1) BANK
BANK BRANCH → (1, N) ACCOUNT
ACCOUNT → (1, 1) BANK BRANCH
BANK BRANCH → (1, N) LOAN
LOAN → (1, 1) BANK_BRANCH
ACCOUNT → (1, M) CUSTOMER
CUSTOMER → (1, N) ACCOUNT
LOAN → (1, M) CUSTOMER
CUSTOMER → (0, N) LOAN

5. Every customer must have at least one account

CUSTOMER in A_C → (1, N)

Customer can have at most two loans

CUSTOMER in L_C → (0, 2)

A bank branch cannot have more than 1000 loans

BANK_BRANCH in LOANS → (0, 1000)

MODULE 2

3 Define theta join.

Given the two relations R and S:

A	B	C
1	2	3
4	5	6
7	8	9

D	E
3	1
6	2

Find $R\theta_{B<D}S$.

A **theta join (θ-join)** is a join operation between two relations where the join condition is any comparison operator such as $<, >, \leq, \geq, \neq, =$.

It is written as:

$$R \ \theta \ S$$

where θ is the condition.

Find: $\boldsymbol{R \ \theta_{B<D} \ S}$

Join condition: **B < D**

Step-by-step checking

1. **(1, 2, 3)** from R
   - 2 < 3 ✅ → join with (3,1)
   - 2 < 6 ✅ → join with (6,2)
2. **(4, 5, 6)** from R

- 5 < 3 ❌
- 5 < 6 ✅ → join with (6,2)

3. **(7, 8, 9)** from R
    - 8 < 3 ❌
    - 8 < 6 ❌
      → no match

A	**B**	**C**	**D**	**E**
1	2	3	3	1
1	2	3	6	2
4	5	6	6	2

4 Define primary key, candidate key and super key.

Primary Key:
A primary key is a candidate key selected to uniquely identify each record in a table. It does not allow duplicate or NULL values.

Candidate Key:
A candidate key is a minimal set of attributes that can uniquely identify each record in a table. A table can have more than one candidate key.

Super Key:
A super key is a set of one or more attributes that uniquely identify each record in a table. It may contain extra attributes that are not necessary for unique identification.

13 a) Consider the UNIVERSITY database with the following relations: 10

STUDENT (**rollNo**, name, degree, year, sex, deptNo, advisor)

DEPARTMENT (**deptId**, name, hod, phone)

PROFESSOR (**empId**, name, sex, startYear, deptNo, phone)

COURSE (**courseId**, cname, credits, deptNo)

ENROLLMENT (**rollNo, courseId, sem, year**, grade)

TEACHING (**empId, courseId, sem, year**, classRoom)

PREREQUISITE(**preReqCourse, courseID**)

Write relational algebra expressions for the following queries:

i. For each department, find its name and the name, sex and phone number of the head of the department.
ii. Find courses offered by each department.
iii. Find those students who have registered for all courses offered in the department of Computer Science.
iv. Obtain the department Ids for departments with no lady professor.
v. Obtain the rollNo of girl students who have obtained at least one S grade.

i. For each department, find its name and the name, sex and phone number of the head of the department.

We need to join the DEPARTMENT table with the PROFESSOR table where the hod (Head of Department) matches the empId.

ΠDEPARTMENT.name, PROFESSOR.name, sex,

PROFESSOR.phone(DEPARTMENT $\bowtie_{hod=empId}$PROFESSOR)

ii. Find courses offered by each department.

This requires a simple projection of the course names and their associated departments.

$$\Pi_{deptNo,cname}(COURSE)$$

iii. Find those students who have registered for all courses offered in the department of Computer Science.

This is a classic "Division" operator problem.

1. **Step 1:** Get all courses offered by 'Computer Science'.
2. **Step 2:** Get the registration list (student roll numbers and course IDs).
3. **Step 3:** Divide the registrations by the target courses.

$TargetCourses = \Pi_{courseId}(\sigma_{name='ComputerScience'}(DEPARTMENT \bowtie_{deptId=deptNo} COURSE))$ $StudentRegistrations = \Pi_{rollNo,courseId}(ENROLLMENT)$

Result: $StudentRegistrations \div TargetCourses$

iv. Obtain the department Ids for departments with no lady professor.

We use set difference: (All Department IDs) minus (Department IDs that have at least one lady professor).

$AllDepts = \Pi_{deptId}(DEPARTMENT)$ $DeptsWithLadies = \Pi_{deptNo}(\sigma_{sex='F'}(PROFESSOR))$

Result: $AllDepts - \rho_{deptId \leftarrow deptNo}(DeptsWithLadies)$

v. Obtain the rollNo of girl students who have obtained at least one S grade.

We filter the `STUDENT` table for girls and the `ENROLLMENT` table for 'S' grades, then join them on `rollNo`.

$$\Pi_{rollNo}(\sigma_{sex='F'}(STUDENT) \bowtie \sigma_{grade='S'}(ENROLLMENT))$$