Class-1

Database Management System

Syllabus

Unit-1

Introduction :-

1. Introduction to db
2. Characteristics of db
3. File system v/s. db system
4. Users of db system
5. Approaches of building a db
6. data models
7. dbms
8. data Independence
9. DBMS architecture
10. challenges in building a DBMS
11. Various Components of a DBMS

Unit-2

ER-model :- (5 classes)

1. Conceptual data modeling - motivation
2. Entity, Entity types, various types of attributes, relationships, relationship types, Entity set types, participation constraints
3. ER notation
4. Extended ER model

Unit-3

Relational data model :- (4 classes)

1. Concepts of relations
2. Schema - instance distinction
3. keys

④ referential integrity and foreign keys
⑤ Converting the db specification in ER notation to the relational schema.

Unit-4

Relational Algebra & Calculus :- Operators - Selection, projection, cross product, various types of joins, division, set operations, example queries, tuple relational calculus, domain relational calculus. (8 days)

Fundamentals of SQL (PLSQL) in lab.
↳ Standard can be executed on the provided by different vendors.

Unit-5

Relational db design :- ① Importance of a good schema design.
② Problems encountered with bad schema designs
③ motivation for normal forms
④ dependency theory - functional dependencies
⑤ Armstrongs axioms for FD's
⑥ closure of a set of FD'S
⑦ minimal cover
⑧ Normalization - 1NF, 2NF, 3NF, 4NF, 5NF, BCNF
⑨ decomposition and desirable properties of them.
⑩ multi-valued dependencies, join dependency, concept of denormalization. (10 days)

Unit-6

Transaction Processing :- and Error Recovery

(7 classes)

1. Concepts of TP
2. ACID properties
3. Concurrency control
4. Serialization
5. Locking based protocols
6. Timestamp based protocols
7. Recovery and logging methods.

Unit-7

Data Storage & Indexing :-

(8 classes)

1. File Organisations
2. Primary, secondary Index structure.
3. Various Index structures - hash based, dynamic hashing techniques, multi level indexes, B & B^+ trees.

Books :- Navathe, Silberschatz.

Marks Distribution

Mid Sem :- 20 marks.

Activity :- (5 components)

10 marks → test 1 (before mid) and test 2 (after completing Portions).

10 marks → Assignment 1 (→ mid sem paper), Assignment 2 (→ End sem paper)

10 marks → 5 marks (attendance + notes), 5 marks (presentation → group activity)

Class-2

Introduction

Basic Concepts of db systems.

What is data?

A→ It is a fact that can be recorded and has some implicit meaning. Eg:- 15 (i know the meaning) → someone else cannot.

What is database?

A→ collection of related data *[individually each one is a fact]

Ex:- 1. A Btech Kiit (real fact about a real world obj)

What is DBMS? → (software)

A→ Set of programs → written in any programming lang, which helps us to create databases, manipulate data in an efficient way.

What is db System?

A→ (database) + DBMS + storage
application programs. (actual data stored on the disk)

(1)

Now a days Information technology + Computing techniques ⇒ cater to the need of various fields like Engineering, economics, science etc.

Data

1
A
Btech
Kiit

105686
M
Mtech
Kiit

Data-base

1 A Btech Kiit → student db.

105686 M Mtech Kiit → faculty db.

(Data is always stored as relation (table)) → So, the name relational db.

DBMS.

provides software to create db, manipulate.

DB system : Database + DBMS. + stored data
(↳ application programs)

↓

data stored can be electronically accessed through a computer.

DBMS examples :- mysql, oracle, dBase, Microsoft Access, Sql Server.

What all can be done with DBMS?

① Defining the database

② Constructing the database

③ Manipulating db for various applications

Defining means :- what should be there in the table?, how many columns, data-type? constraints..

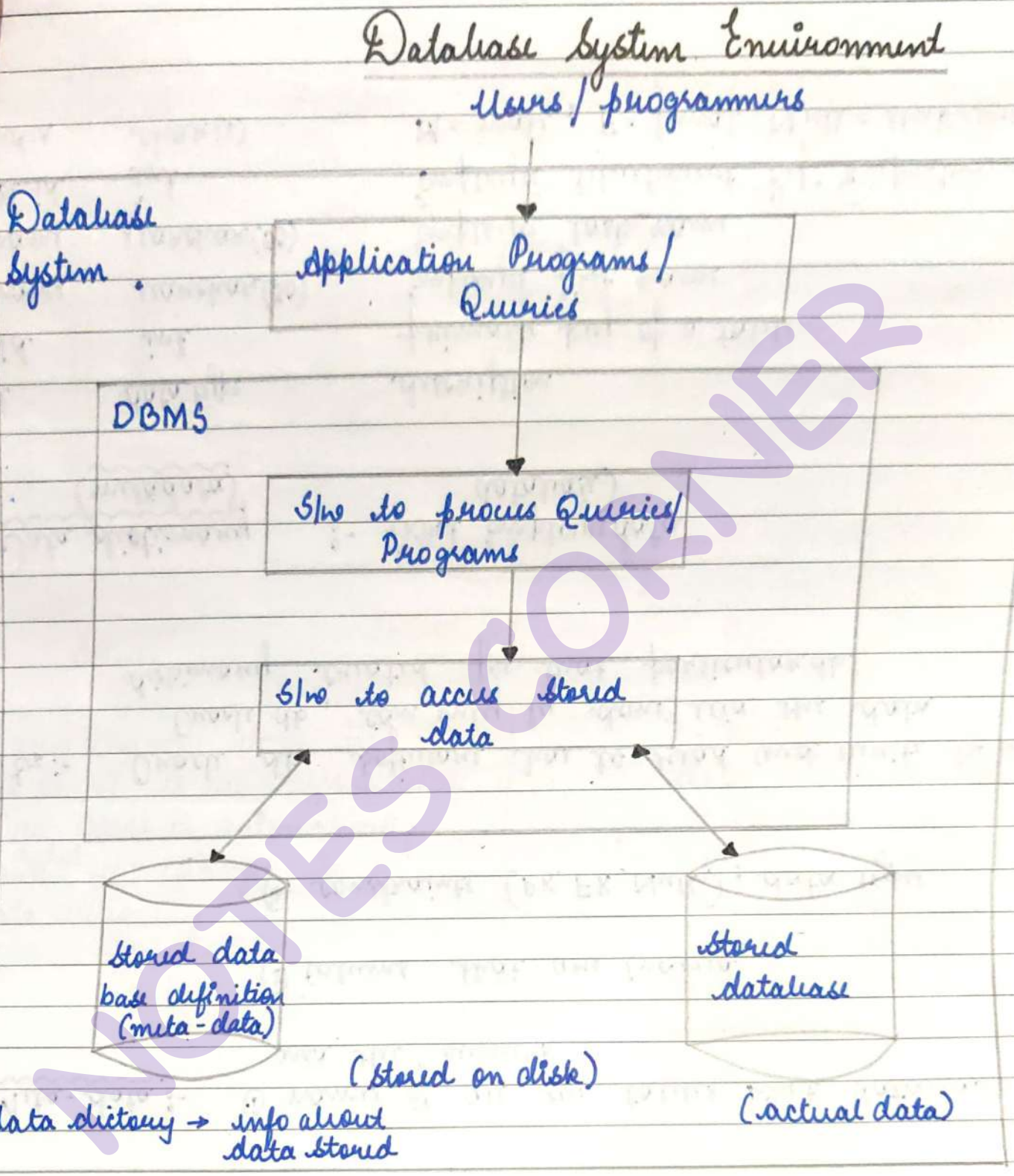
Constructing :- create table, insert data

manipulating :- with the requirement of application, we will have to select a part of data, delete, alter.

We have applications like chartrip, MMT

(we will be storing basic data) remaining is done by the application.

Database System Environment



Data dictionary and meta-data :-

data dictionary :- is set of files / files that contains database's metadata.

② data dictionary is a crucial component of any relational database, as it is invisible to database users. Typically only the database administrator interacts with the data dictionary. ※

Meta-data :- ① names of all the tables wrt databases and the owners

② Colums that are indexed

③ constraints (PK, FK, Null), data types

Ex :- Oracle db software has to read and write to an Oracle db, can only be done via the data dictionary created for that particular db.

Data dictionary (metadata) :- (wrt Employeedata database)

col	data type	discription
E_id	int	primary key of a table
F_name	varchar(50)	Employee first name
L_name	varchar(50)	Employee last name
dept_id	int	Employee department Ref: Department (table)
gender	char(1)	M = male, F = femal, Null = Unknown

actual data

e_id	Fname	L_name	dept_id	gender
1	A	X	1	M
2	B	M	1	F
3	C	O	2	F
4	D	F	3	F
5	Q	T	4	M

Data dictionary

Active

if any changes done in the db, it is the responsibility of the DBMS to automatically updated the changes in the data dictionary.

Passive

Contrarily here the data dictionary has to be manually updated by the administrator. Should be careful otherwise db and data dictionary are out of sync.

Database Objects :- a db is made up of various objects each having specific responsibility for managing data.

Ex:- a) table object → stores records.

b) view object → creates a virtual table for a specific user.

Data Dictionary and System Catalog

① We create a lot of tables, views, constraints, indexes in a database, difficult for the developer / user to remember all of them.

② Since every database stores every information about structure, definition, storage, no of columns, records, dependencies, access rights, owner will be stored. → this is called meta-data.

③ Meta-data is also stored in a table.

④ Collection of all these metadata is stored in data dictionary or System catalog.

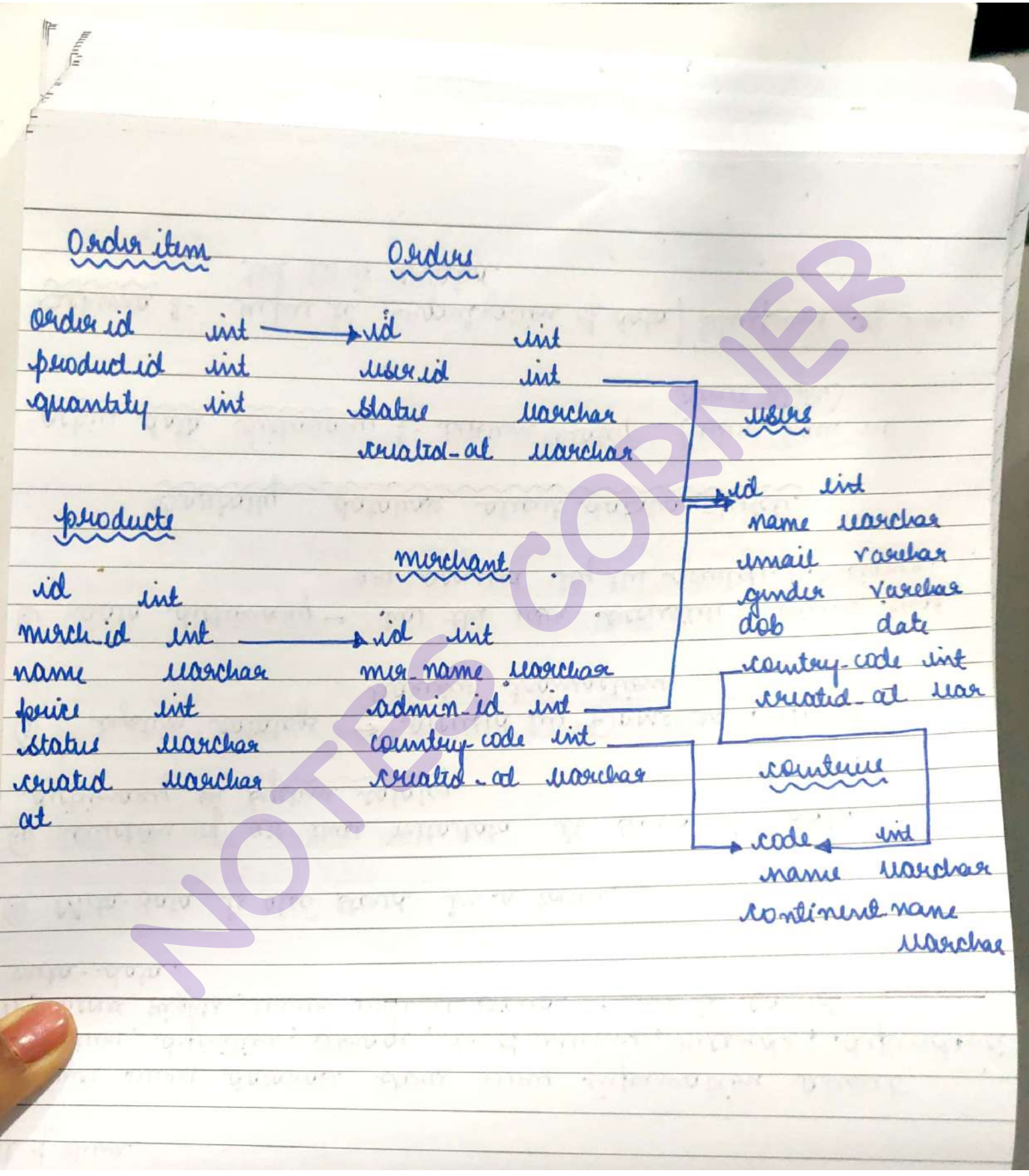
⑤ System catalogs → accessed by DBMS to perform various transactions

⑥ data dictionary → has the user accessible views that are accessed by the developer / designer.

Basically database about database objects.

active data dictionary :- System Catalog (users have no access rights)

Schema :- refers to organization of data / blueprint of how the db is created.



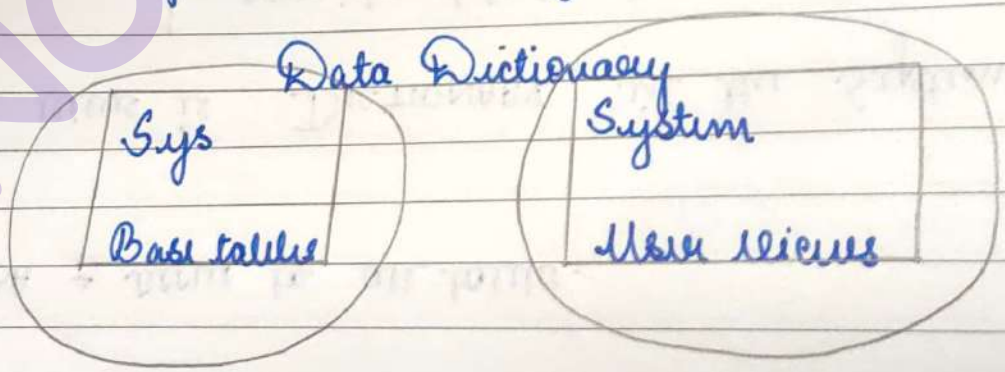
Sys Schema :- is a read only database that contains all system objects.

tables, views, procedures, functions, constraints, log, default, → One row for each.

(7) In order to know info about table, data dictionary / system catalog has to be viewed. for this reason Select rights are given to the users.

(8) DD tables are created in the Sys Schema no access privileges are given to the users. Only RDBMS access it.

(9) Views of these tables are given/created in the System Schema where users only have select rights.



(10) depending on access rights we have 3 users.

① User → tables / views created by that user.

② dba → all tables / views owned by the users.

③ DBA → access to all tables.

One of the views is DICTIONARY in the System schema

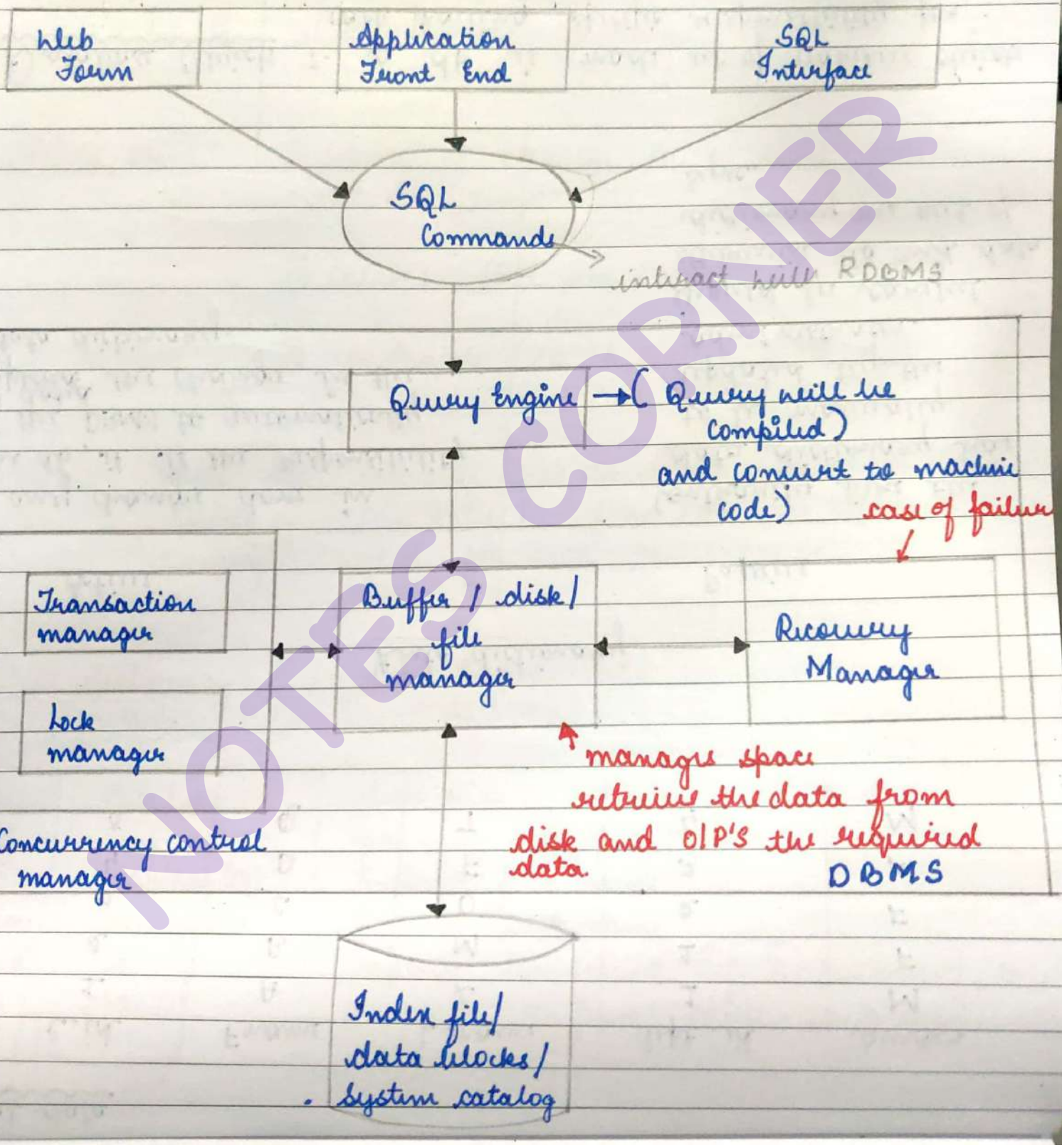
↳ provides details of all the objects in the db System.

table_name	comments.

Select * from DICTIONARY;

Class 3 :-

DBMS Modules (Architecture)



Steps.

Compiled
↳ check for syntax.
↳ generate a query tree
↳ plans (ordered steps used to access the data of db)
↳ select the best plan (query optimization)
↓
convert to machine code.
↓
to access data.

File System

The first model that was suggested for storing the data was 'Flat File Model'.

Here all the data was just dumped into a single file.

Table : (Car Insurance Company)

Name	contact	income	car reg no	model	name of car	place of accident	date of accident	insure
S	.....	10,000	MH-05-1318	2004	...	Bandra	31-5-2005	500
S	.....	10,000	MH-05-1318	2004	...	Borivelli	30-8-200x	500
S	.....	10,000	MH-05-1318	2004	....	Null	Null	500
R	.....	5000	KA-05-1515	2009	....	Null	Null	20
R	.....	5000	KA-05-1515	2009	....	Null	Null	20

Disadvantages :-

① **Data Redundancy :-** repetation of the data
(we can see tuples / rows having same data or just minimal difference)

② **Data Inconsistency :-** change done to 1 row, should be replicated to other rows which has same data. Otherwise the data would loose its intigrity.

③ **Data Insecurity :-** either the entire file is given access or the entire file is not given access. There is no tuple / attribute level access.

④ **Concurrent access :-** not allowed.
(multiple users trying to access the file)

⑤ **Memory Utilization :-** Since few tuples will have null values wrt attributes, wastage of memory.

⑥ **Flexibility :-** concurrent access not provided, memory utilization is poor. so, its not flexible.

hence 'flat file model' was discarded and RDBMS came to use.

(2marks) Characteristics of DataBase Managment System

① Removes redundancy and provides security.

② Self describing nature

③ Supports multiple views of the same data.

④ Sharing data and multiuser transaction processing.

⑤ DBMS follows Acid properties.

Difference between DBMS and Flat File

DBMS	Flat File
① multi user access.	does not support.
② design to fulfill the needs of small & large businesses	② only small businesses.
③ Removes redundancy preserves integrity	③ redundancy and integrity issues
④ Expensive	④ cheap
⑤ handles complicated transactions	⑤ do not support

Database Users

① actually use & control the db content (actors on the scene)
→ db designers, db administrators
→ define the content, structure, constraints, functions, transactions

① design, develop & maintain (actors behind the scene)
→ communicate & understand the needs

Name	task
① Application Programmers (writing a 'C' code to fetch the details of employee working in a particular department)	write programs in various languages to interact with databases. Embedded sql query in C program.
② Database Administrators (authorize access, coordinate & monitor its use, acquire h/w, s/w resources)	helps to manage the entire DBMS.
③ End-Users → casual (access the db occasionally) → naive	interact with dbms. Perform various operations on db like retrieving, updating, deleting.
④ Sophisticated Users (designers & developers of DBMS) business analyst, scientist, engineers know all system. work with stored db.	a) write sql queries (to select/ insert / delete / update data). b) do not use any application program to access the db. Ex: scientists, engineers, analyst.
⑤ Specialized Users	developers who develop the complex programs to the requirement.

⑥ Stand-alone users Scientist creating db for his own experiments. User maintaining address list	use stand-alone db for personal use. Bastially these dbs will have menues and GI.
⑦ Native users	users who use the existing application to interact with the db. Ex:- online library system, ticket booking use them to interact with the database to fulfill their requests.

Database Administrators :-

① The life cycle of db starts from designing, implimenting to administration of it.

② The db grows as the data grows in the database. due to which its performance comes down, also accessing the data becomes challenging.

③ The administration and maintenance of db is taken care by database administrator - DBA.

A good performing db is in the hands of a DBA.

① Installing and upgrading the DBMS Servers :-

a) DBA is responsible for installing a new DBMS Server for the new projects.

b) As the new versions come up, upgrading of the servers should take place.

c) Also should update the service packs, patches to the DBMS servers

② Design and Implimentation :-

decision about proper memory management, file organization error handling, log maintenance etc for the db.

③ Performance tuning :- Since data in database is quite voluminous, there would be variation in the performance. DBA should insure that all the queries, programs work in fraction amount of time.

④ Migrate db servers :- Sometimes users using Oracle would like to shift to SQL Server. It is the responsibility of DBA to make sure that the migration happens without any failure and there is no data loss.

⑤ Backup and recovery :- proper recovery and back programs needs to be developed by DBA. This is an important responsibility. Data / db objects should be backed up regularly so that if there is any crash, it should be recovered without much effort and data loss.

⑥ Security :- DBA is responsible for creating various users and giving them different levels of access rights.

⑦ Documentation :- DBA should properly daument all his activities so that if he quits or any new DBA comes in, he should be able to understand the db without any effort. He should basically maintain all :-
a) instalations, b) recovery and backup updates
c) security methods followed d) reports about db performance

Class 4

Approaches of building a database

In order to remove all limitations of the file based approach, a new approach was required that must be more effective known as 'Database Approach'.

① The db is a shared collection of logically related data, designed to meet the information needs of an organization.

② A db is a computer based record keeping system whose purpose is to record and maintain information.

③ A db is a single, large repository of data, which can be used simultaneously by many departments and users. Instead of disconnected files with redundant data, all data items are integrated.

④ The db not only holds organizations operational data but also a description of this data (metadata), for this reason db is also defined as self describing collection of integrated records.

In the DBMS approach, application program written in some programming language can be Java etc uses database connectivity to access the db stored in the disk with the help of OS file management system.

End User

↓

Application program

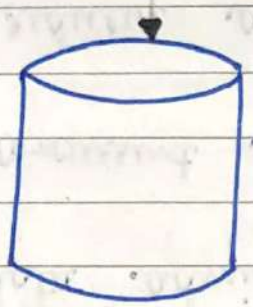
↓

Interface designed in lang like C, C++, Java

↓

OS

file managment system



File - System Interface

End user

↓

Application prog

↓

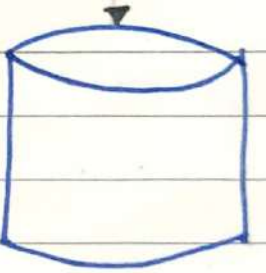
Interface designed in lang like C/C++/Jav

↓

DBMS

↓

OS file managment system



DBMS Interface

Building blocks of a db :-

1. columns / attributes
2. Rows / tuples
3. tables :- logical collection of columns

   ※ there should be no duplicate data.

Characteristics of db (4 marks)

The data in a db should have

Advantages of using DBMS

1. Data Independence
2. Efficient data access
3. Data integrity and security (accuracy and consistency of data)
4. data administration
5. concurrent access and crash recovery
6. reduced application development time

Data Model (used to represent the structure of data in the db)

A data model is a collection of concepts that can be used to describe the structure of the database.

Structure means :-

a) data types
b) relationships
c) constraints etc.

DBMS allows a user to define data to be stored in terms of data model.

Data Model

- high-level data models (entities, objects, attributes, relationship) to describe db, so very close to user understanding
- low level data models (describe db based on storage → record, hash, index) for db.
- Representational / Implementation data models (also the type models) describe the db depending on implementation of data model.

Relational Model (Implimentation data model)

The central data description construct in this model is a relation.

relation = set of records.

Schema :- description of data in terms of a data model is called schema.

A relation schema specifies :-

a) the name of the relation
b) fields
c) type etc (data type)

Ex :- Student (sid : String ; name : String ; age : integer)

※ each row follows the schema of the relation.

High level data model	Representational data model	low level data model
→ ER model → EER model	→ N/w model → Hierarchial model → Relational model → Object data model → Object relational model	(how the data is stored) [disk, file, record]

Description of Data models

① Relational model :- model sorts the data into table. (also called as relation)

(Ex:- Oracle, Mysql, Sql Server)

a) column → attributes
rows → records / tuples.

b) Unique attribute helps to identify the row - Primary Key. this PK is refered by another table it is called Foreign key.

Ex:-

Stud_id	Fname	Lname

N/w

Provider_id	Provider name

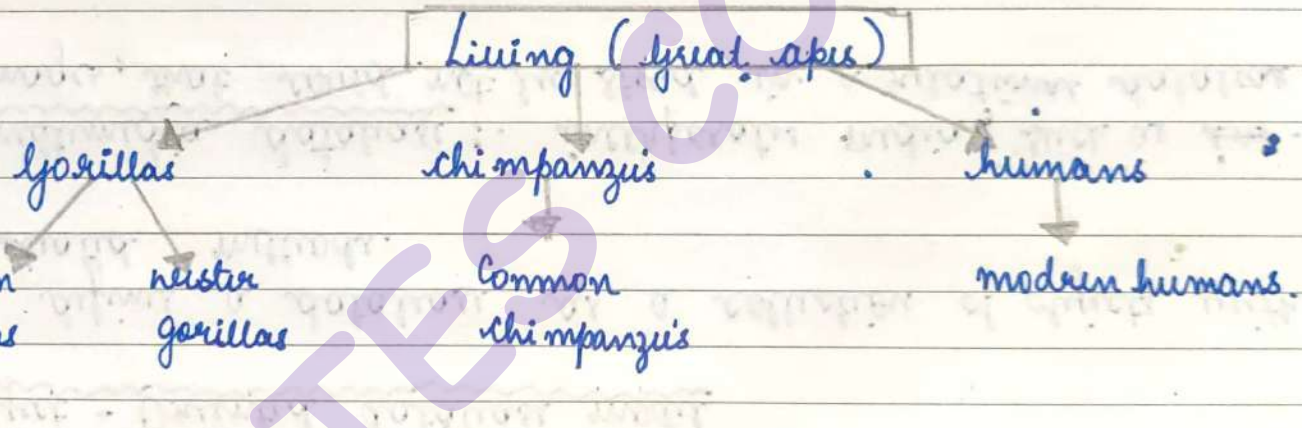
Stud_id	N/w. provider id	type of plan	start date

Relational databases are typically written in SQL. The model was introduced by E.F. Codd in 1970.

② Hierarchical Model :- Organizes data into a tree like structure, each record has a single parent or root. Sibling records are sorted in a particular order.

Good for describing many real world relationships.

primarily used by IBM's → Information management System in 60's and 70's. (Ex :- IBM's IMS in late 60's)
(Example :- Student and course, a Student can have multiple courses)

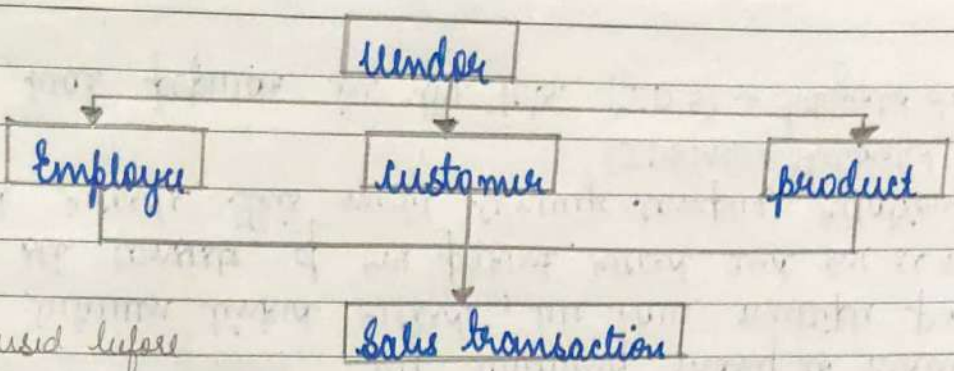


③ Network model :- (data organised as graph) builds on hierarchial model by allowing many to many relationships between linked records, we have multiple parent record. Each set consists of one parent record and one or more child records. This model conveys complex relationships.

(IDBMS) → Integrated
(IDS) → Integrated data Store.

It was popular in the 70's.

[can have more than 1 parent] [more relationships are established, so data is more related]
accessing data is easier and fast



widely used before relational data model

(Ex: GE's integrated data store (IDS) early 60's

④ Object - Oriented database model

It defines a database as a collection of objects with associated methods.

A multimedia database :- incorporates media, such as images, that could not be stored in a relational database.

A hypertext database :- allows any object to link to any other object.

It is ideal for organising data, but not ideal for numerical analysis.

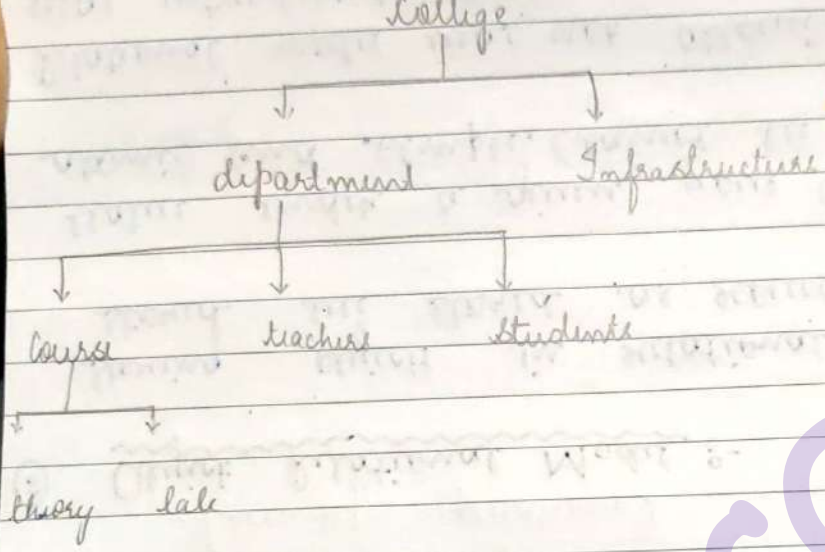
Object 1: Sales report

month	01-15-16
Product code	54
Vendor	154-234
Revenue	$887

Object 2: Sales activity

Customer
Product code
Product name
Sales associate
Date of sale
Price

Hierarchial model → Organise data in the form of a tree



Data Model :-
Collection of concepts used to describe the database.

⑤ Object Data Model :-

We create objects using some programming language (C/C++/Java) with its actual complex structure, we try to store in the database.

'Object Query language' → is used to query ODM.

Ex:- Object Store, Ontos, Poet etc.

(Scientific applications)

⑥ Object Relational Model :- (Values should be simple, atomic)

Storing objects in relational data model, constraint is stored we stored as rows and columns.

Value under a given row and column should be atomic and simple. (cannot be complex / multivalued)

Relational model was not allowing that, so this model was introduced ORM.

Ex:- Oracle (8i and onwards), Informix..

.

.

.

.

Relational, Network and hierarchial are all based on records, how we maintain the relationship between the records is different.

Customer records

101	Gopi	hyd
103	mohan	che
108	Kiran	che
112	John	Del

account records

10	10,000
20	2000
30	17000
40	5000

[1 customer can have multiple accounts, and 1 account can have multiple customer (joint account)

a) customer (1:K) account.

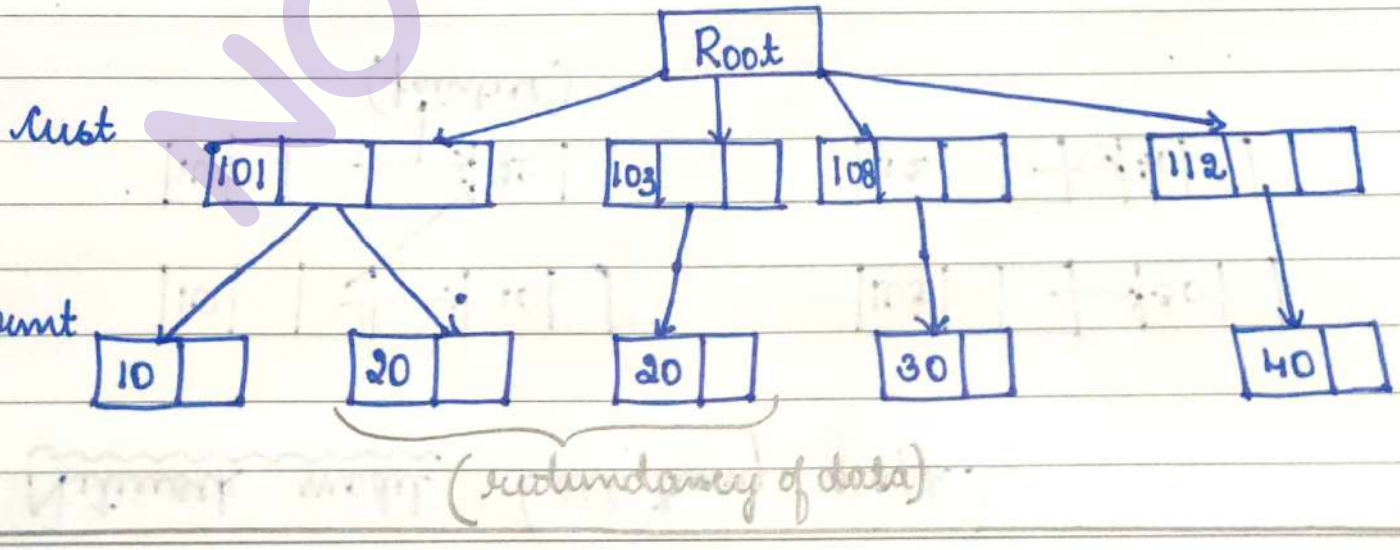
(arranged as table)

Cust#	Name	City	Ac#
101			10
101			20
103			20
108			30
112			40

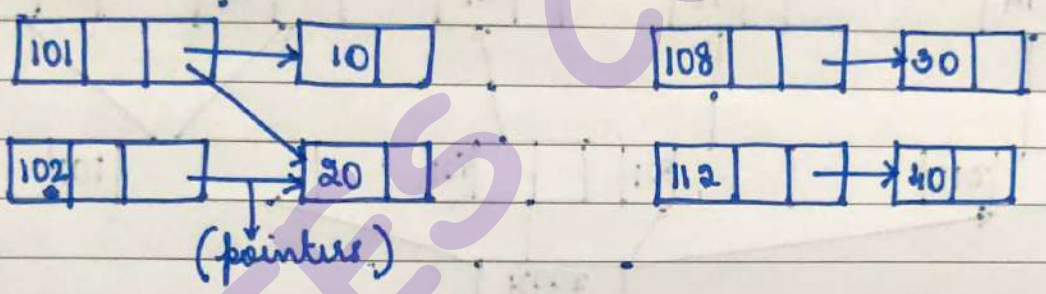
Ac#	balance
10	10K
20	2K
30	17K
40	5K

Relational model.

b) hierarchial data model :-



Network model (arbitrary graph)



Database Schema

Schema :- plan to describe something.

DB Schema :- description of the database.

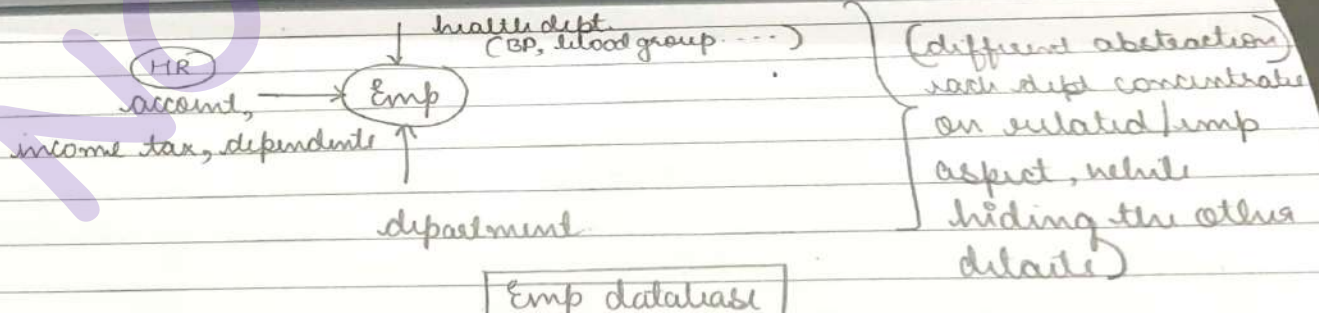
Three Schema Architecture

A db can be described using three different levels of abstractions.

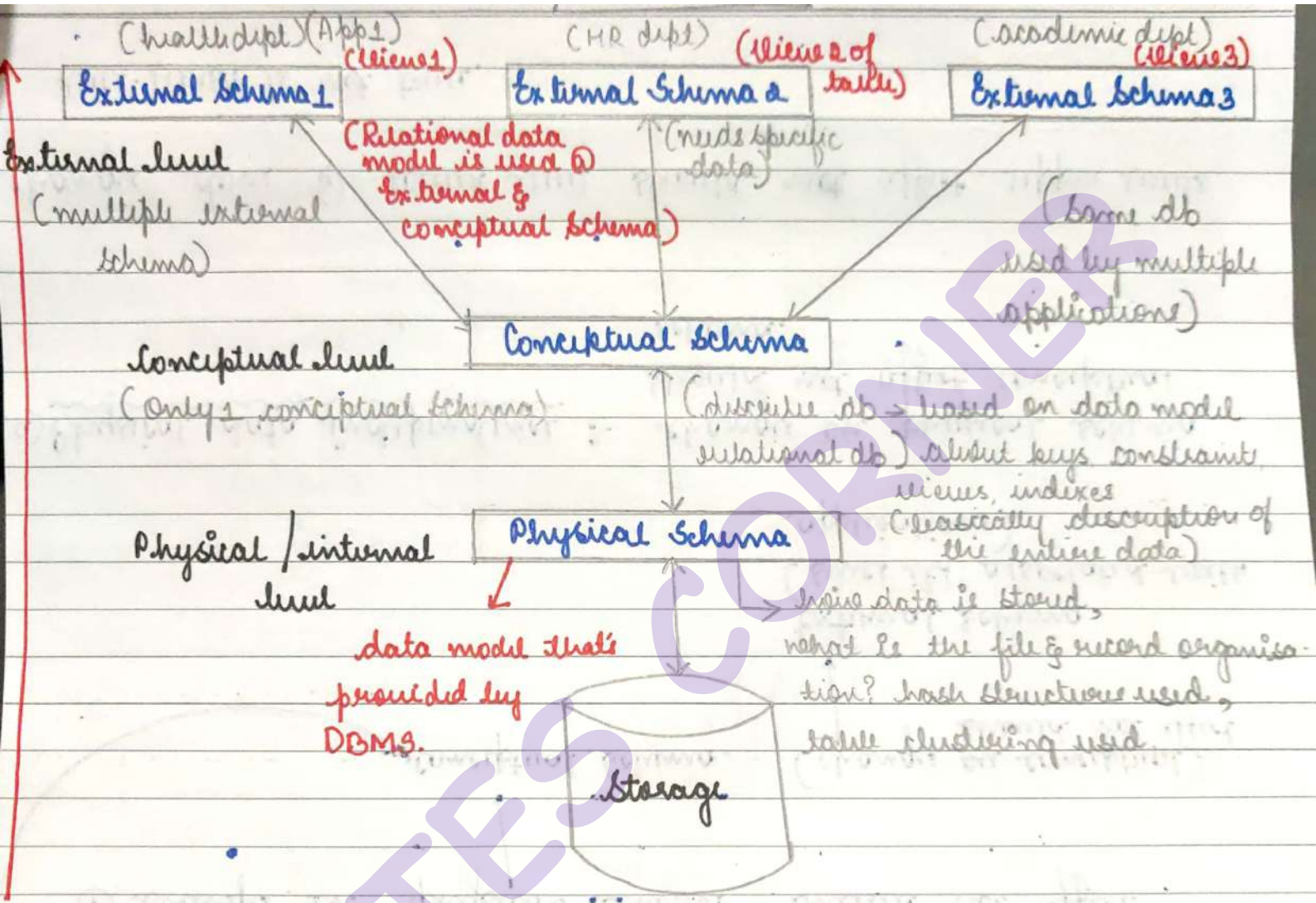
Description at each level can be defined by a Schema. For each abstraction we focus on one of the spicific issues such as user views, concepts, storage etc.

1. External Schema
2. Conceptual Schema (logical Schema)
3. Physical Schema.

} viewing db @ different levels/ angles.



(different abstraction each dept concentrate on related/emp aspect, while hiding the other details)



Three Schema architecture of DBMS.

Advantage of Three Schema Architecture

Logical data Independence :-

① changes in physical schema (stored on tape or disk / spanned or unspanned record organisation) should not affect conceptual schema. (changes on conceptual should not affect external schema (since it's associated with applications)

a) Physical data independence :- changes on physical schema should not affect conceptual schema.

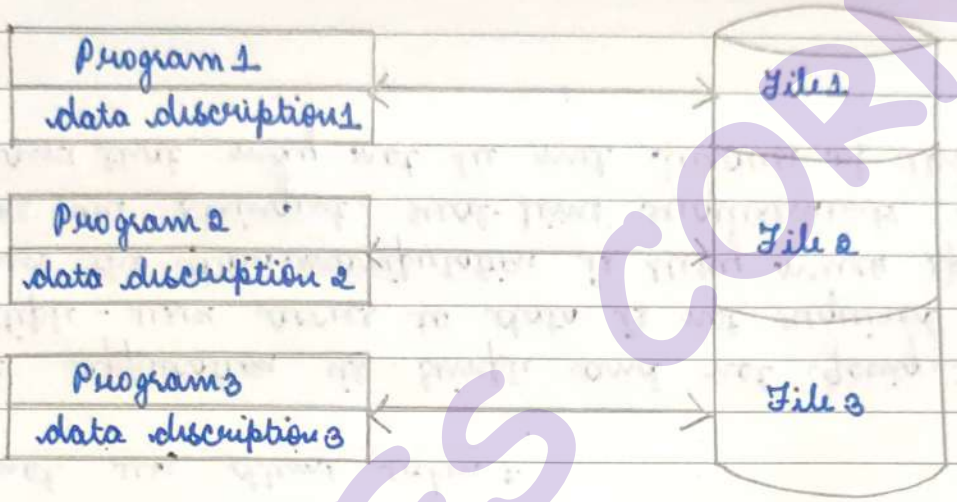
changes done @ lower level should not affect upper levels

vice-versa is not true.

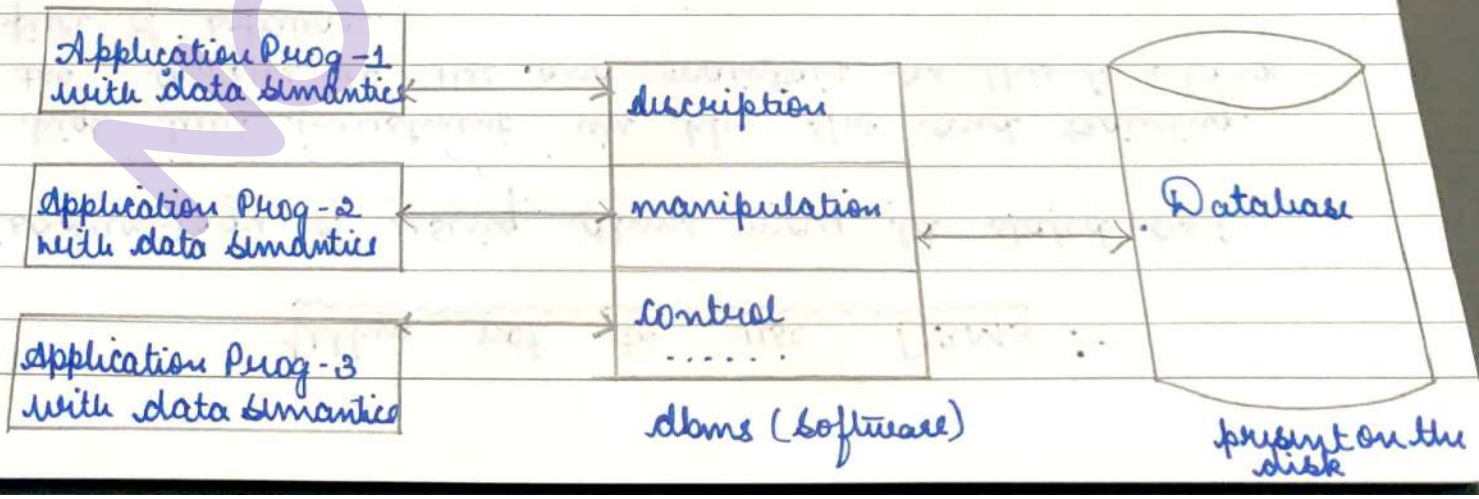
Characteristics of the database Approach.

① Individual programs, have their own files (where they can open, save and close). So each program will have individual files.

② Suppose a program wants to search a piece of information need to open and search the entire file. (too time consuming)



Database Approach :- Independent of the system dependent code and can achieve program-data independence.



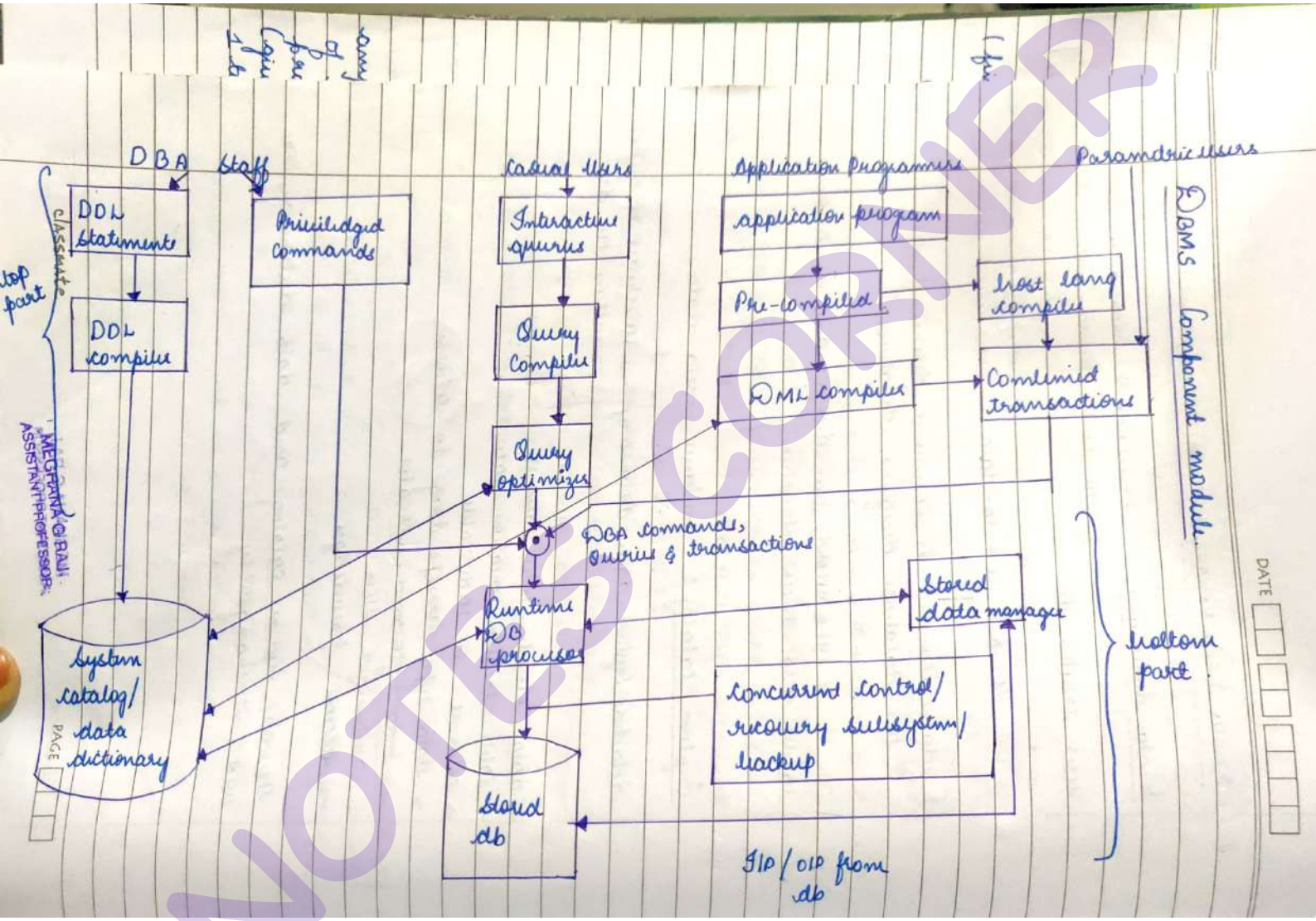
When not to use DBMS :-

Disadvantages of using dbms may be stated as :-

a) high level investment in h/w, s/w and training.
b) too complex to use and maintain as this is a large piece of software.
c) Overhead for providing security, concurrency control, recovery and integrity functions.

Do not use dbms when :-

a) The application is simple and not going to change often.
b) multiple - user access to data is not required
c) when the data manipulation is very much specialized.
d) There are stringent real-time requirements for some programs that may not be met because of dbms overhead.



DATE

① Fig divided into two parts.

Top part :- various users of the db sys and their interfaces.

lower part :- internals of the DBMS responsible for storage of data and processing of transaction.

② Database + DBMS → stored on disk catalog

③ access to the disk controlled by OS, which schedules read/write.

④ many DBMS have their own "buffer management" module to schedule disk read/write, because this has a considerate effect on performance.

⑤ Stored data manager :- controls access to DBMS info that is stored on the disk, whether part of db or the catalog.

Top part discription :-

① DBA staff :- works on defining the db and tuning it by making changes to its definition using ddl / other privileged commands.

② casual users :- work with interactive interface to formulate queries

③ application programmers :- create some programs using some host programing lan

PAGE

classmate

MEGHANA G RAJ
ASSISTANT PROFESSOR

④ Parametric users :- do the data entry work by supplying parameters to predifined transactions.

Working :-

① DDL compiler — proces schema difinition specified by DDL

↓

Store the description of schema (meta-data) in the DBMS catalog.

② catalog includes a) info such as name & size of files
b) data types
c) storage details of files
d) constraints
e) also contains info needed by the DBMS module.

Causual user ⟶ interactive query interface

↓

query are parsed and validated for correctness of query syntax.
(by query compiler)

↓ compiles to

internal query ⟶ subject to Query optimization

DATE

④ Query optimizer —concerned with→ a) reordering of operations
b) elimination of redundancies.
c) use of correct algo during execution.

a) consults the system catalog → statistical / physical info about stored data

↓

generate executable code.

↓

performs necessary operations for the query, makes calls on runtime processor.

⑤ App programmers → write code in C/C++/Java

↓

submitted to a pre-compiler

⑥ Pre-compiler → extracts the DML command from an application prog

↓

DML commands sent to DML compiler.

↓

compilation into object code for data access.

↓

rest of the prog sent to lang compiler

DATE

⑦ Object code for the (linked) DML commands + rest of the program

↓

canned transaction → executable code includes calls to the runtime db processor.

canned inst → executed repeatedly by Parametric users

Eg:- Bank withdrawal transaction where the account number and the amount may be supplied as parameter.

Lower part description :-

⑧ Runtime db processor executes a) privileged instructions
b) executable query plans
c) canned transaction with runtime parameters.

↓

works with the System Catalog.
also works with Stored data manager

↓

uses basic operating System services for carrying out low-level I/P / O/P (read/write) operations between disk and main memory.

DATE

(9) Runtime db processor handles data transfer, management of buffers in the mm.

Some DBMS's have their own buffer management module while others depend on the OS buffer management.

(10) Concurrency control + backup recovery work on runtime db processor for purpose of transaction management.

ass-1)

Unit-2

E-R and EER models

Question :- Is logical and conceptual schema same?

Conceptual schema

- Semantic level CS
- Implementational CS / logical.

* In 3 schema architecture we meant (logical) Implementation specific conceptual schema.

* Semantic level (high level representation) → ER / EER

% ER / EER fall into the category of semantic level (high level conceptual schema)

constraints are nothing but business rules. ∴
Ex:- While designing a db for a project
constraints can be:-

1) Skill set should know Java.
2) age should be less than 40 years.

} examples

Major Steps in database design process.

① Step 1 :- Requirement Analysis

with respect to the application
- a) Understanding the domain
- b) Identifying the data to be stored
- c) Identifying the constraints. (as explained in the previous page)

② Step 2 :- Conceptual database design

Semantic (high-level) nothing to do with data model

ER modeling / UML.

③ Step 3 :- (3 tier architecture starts here) Logical database design

(here we decide whether to go with relational data model or object data model) according define the

SCHEMA

※ (we decide the table structure and nothing to do with the data)

Step 4 :- Refinement of schema

(here we concentrate on normalising the table)

⑤ Step 5 : Physical database design

(Even though the db designer has no much role since this is taken care by the DBMS) to some extent changes can be done.

taken care by the dbms software we choose.

a) Indexing (externally can be specified)
b) clustering (placing related tables in the same file which would help easy retrieval of data)
c) Storage formats.

What is E-R modeling?

① E-R model is a popular high-level (conceptual data model)

② It is an approach to designing Semantic conceptual schema of a database.

③ ER model allows us to describe the data involved in a real-world environment in terms of objects and

their relationships, which are widely used in design of database.

③ ER model provides preliminary concepts or idea about the data representation which is later modified to achieve final detailed design.

DATE

① ER model (graphical representation of db)
② Steps in database design process
③ Entities, attributes and associations
④ ER notations
⑤ Enhanced ER modeling
⑥ Super class and sub class representation.

ER-model (Step 2) :-

① Is a popular high-level (conceptual) data model.
② It is an approach to designing Semantic conceptual schema of a database.
③ ER model allows us to describe the data involved in a real-world environment in terms of objects and their relationships, which are widely used in design of database.
④ ER model provides preliminary concepts or idea about the data representation which is later modified to achieve final detailed design.

Notions

→ nouns (thing in a real world)

Entity :- Any object in the real world can be seen as entity.

attributes :- (nouns which describe nouns) reflect the property of an entity

entity type :- Strong and weak entity types.

Relationships :- (Verbs) (works for) association b/w relation entity.

Constraints :- cardinality, degree, participation, min-max

hierarchies (super/sub)

Entity Set.

elements form entity → { ⋮ } set → name of set → entity type

DATE

Entity types and entity sets:

① dbs contain group of entities that are similar. ①

Eg: company hires group of employees ②
↓
all employee details are stored together.

employee → share the same attributes ③
└ values for each entity is different.

④ Person (age, name, add)

describes schema or intension → Entity type: collection of entities that have same attributes. ⑤

described by its name and attributes.

⑥ Entity type name — employee (name, age, salary)
company (name, headquaters, president)

Entity Set: the collection of all entities of a particular entity type in the db at any point in time is called an entity set.
↓
extension of the entity type.

Employee (name, age, salary) → entity type

name	age	salary	
A	20	10,000	} entity set.
B	25	20,000	

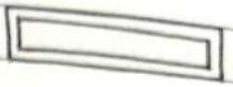
Entity: instance (extension) (A, 25, Bang) (elements in entity type → called state of entity)

DATE

Proper naming of Schema Constructs

1. Entity

2. weak entity

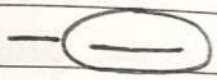


3. relationship

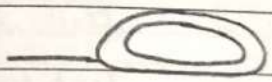
4. Identifying relationship

5. attribute

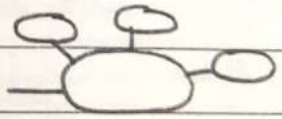
6. key attribute



7. multivalued attribute



8. composite attribute



9. derived attribute.

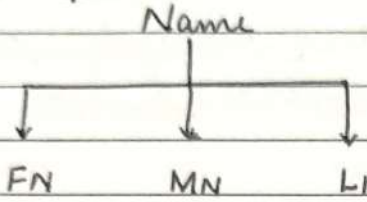
Attributes : describe entity.

Classification :-

Eg :-
Person (Name, age, add, phone)

Simple and composite : a) Simple (atomic) cannot be divided.

b) Composite : can be divided further.



If value does not exist → no phone (so no number)
or
if attribute not applicable → no middle name

in such case insert null value.

Composite → collection of simple value attribute.

Singe Valued and multi valued :-

multi valued attribute : will have more than 1 value.

Eg : (a person can have multiple phone numbers)

Singe valued : will have singe value.

Eg :- age.

DATE

③ Stored and derived attributes :-

Store :- Date of birth stored

derive :- derive age from DOB.

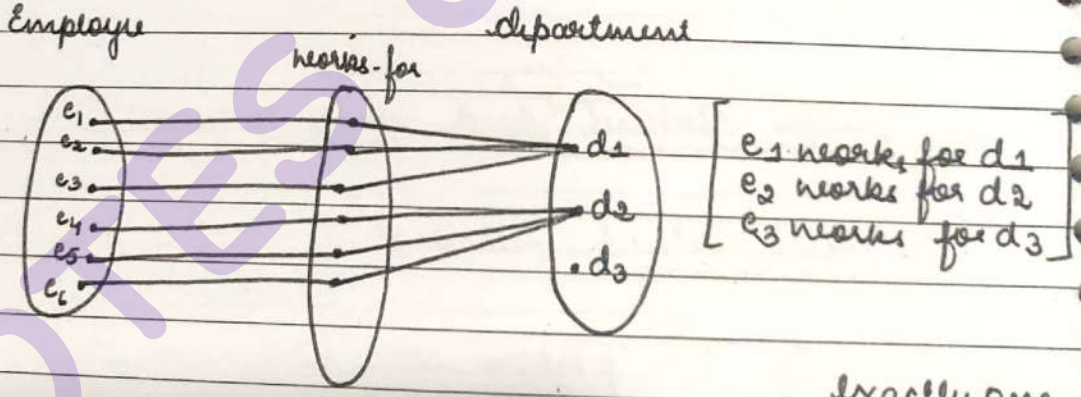
Eg: Name :- composite attribute and singe value
age :- singe value and derived.
add :- multi valued attribute (defined)
(more than 1 add) +
composite.

complex attribute :- composite + multi-valued.

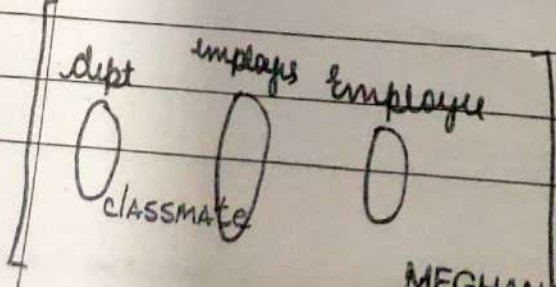
Relationship

1 : M

① Association among entities
② Eg :- Employe works for department.



Requirement analysis :- a) Every employe works for exactly one a department
b) dept can have many employe.
c) new dept need not have any employe.



depending on the arrangement of entity, we decide the relationship.

MEGHANA G RAJ
ASSISTANT PROFESSOR

PAGE

Characteristic of relationship :-

① degree :- how many entities (entity types) are participating.

show example in binary relationship. (2 tables)

② cardinality ratio :- what is the max no of relationships in which an entity can participate.

Eg :- employee can work only in 1 depart (1)
department can have any no of employee (n)

③ Participation / existence constraint : (min no of relationship an entity can participate)
↳ minimum cardinality / existance constraint)
↳ total participation)

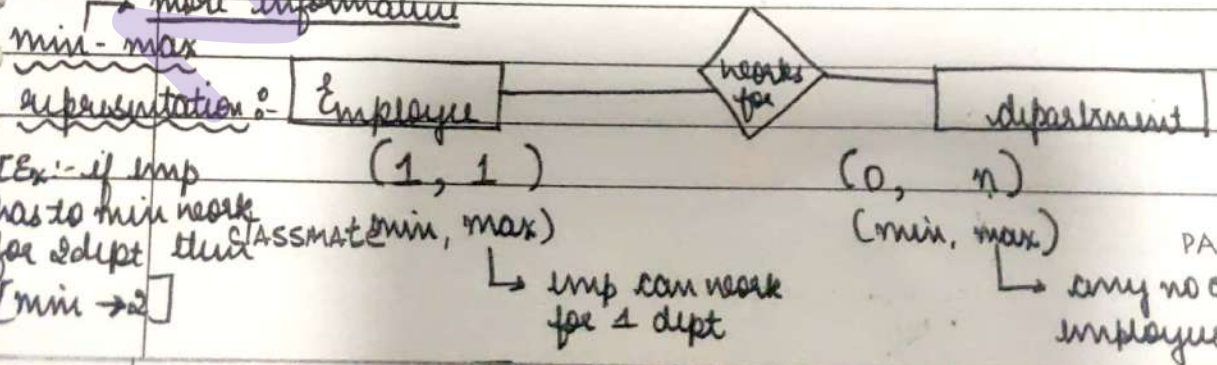
Eg :- min (employee) = 1 (employee has to work for 1 dept)
min (department) = 0. (not necessary for department to have employees).

structural constraints

[depending on verbal discription]

Some of the entities not participating in relation.

min-max representation :- more information



[Ex :- if emp has to min work for 2 dept then [min → 2]]

[ASSMATE]

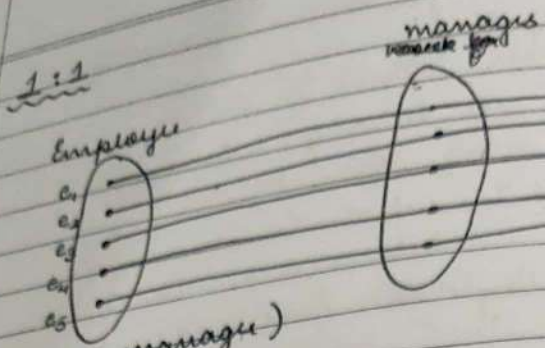
↳ emp can work for 1 dept

↳ any no of employees.

PAGE

DATE

1:1

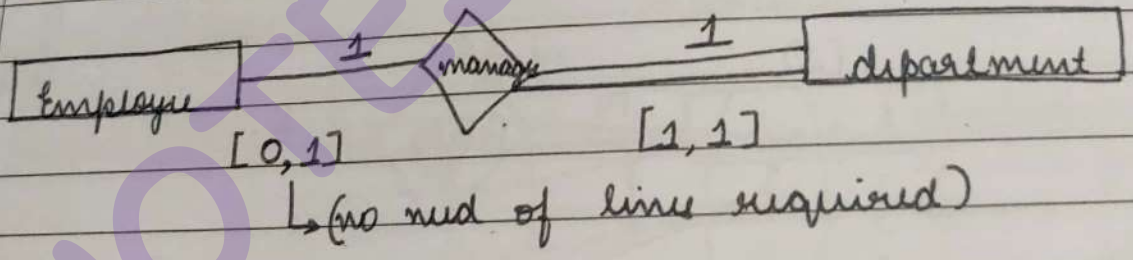


Verbal description :- Every dept should have a manager and only 1 employee manage the department an employee can manage only 1 dept

Employee & dept :- ① degree : 2 (2 entities are participating)

② cardinality ration (max) :- 1

③ Participation :- total on dept side (can have employee are not managers) (min)

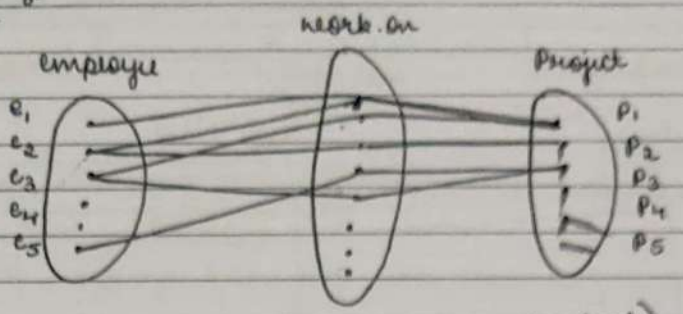


↳ (no need of line required)

DATE

M:N :-

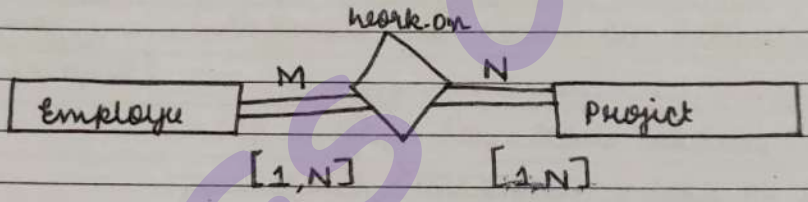
working
$P_1 = e_1, e_2, e_3$
$P_2 = e_2, e_3$
$P_3 = e_5$
$P_5 = e_3, e_4, e_5$



(relation ⊆ cross product)

discription :- Every employe should work on 1 project or many project.
Every project should have atleast 1 or more employe.

Degree :- 2
(max) cardinality :- many.
(min) participation :- total.

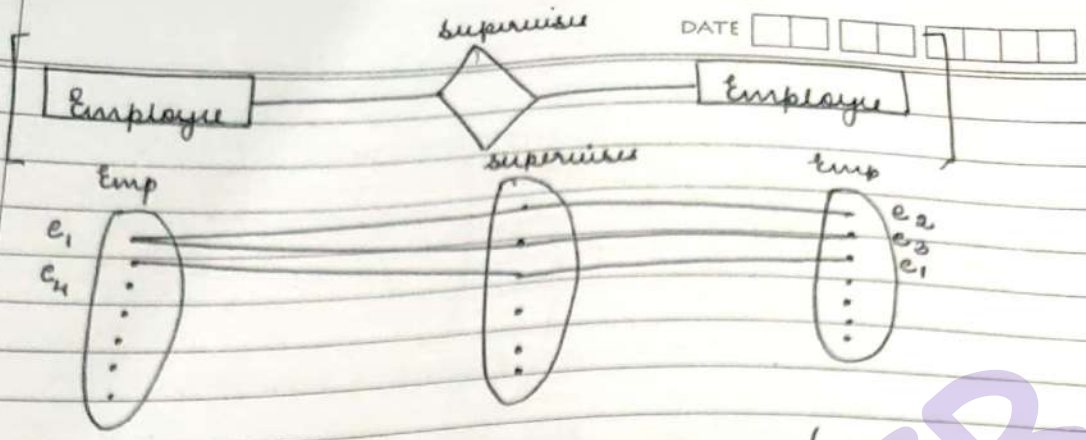


(role: project workes) (employer)

Recursive :-

An employe reports to supervisor and employe must have a supervisor.
[CEO does not report to anyone, can have employe who do not report to any]

(Same table we consider)



(role: supervisor)

(role: supervisee)

Also

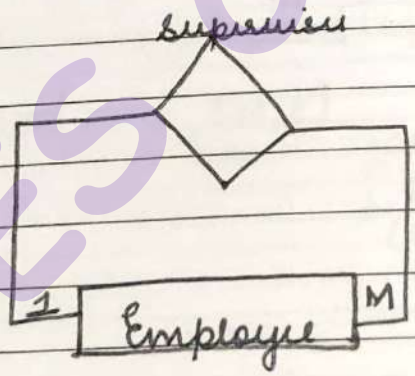
e_2 supervises e_3.

e_4 supervises e_1.

Degree :- 1

cardinality (max) :- supervisor (many)
supervisee (1)

participation (min) :- supervisor (0)
supervisee (0)



role
(supervisor)

(0, m)

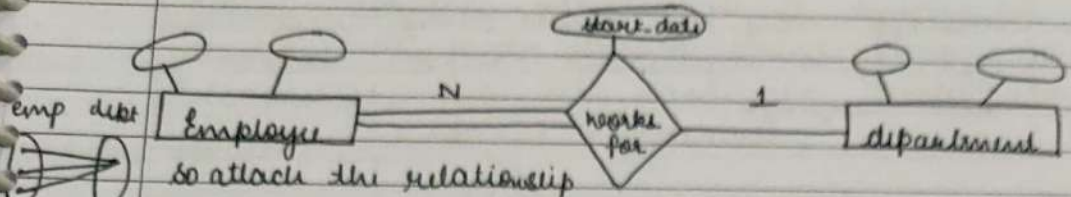
role
(supervisee)

(0, 1)

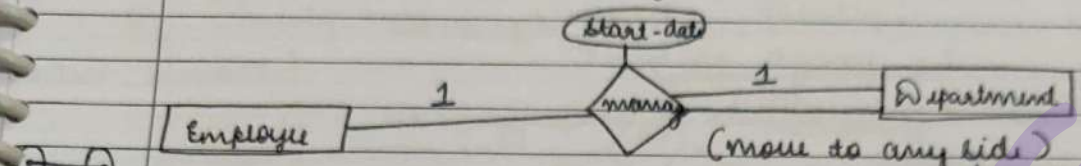
Employee can have only 1 supervisor
a supervisor supervises many employee

DATE

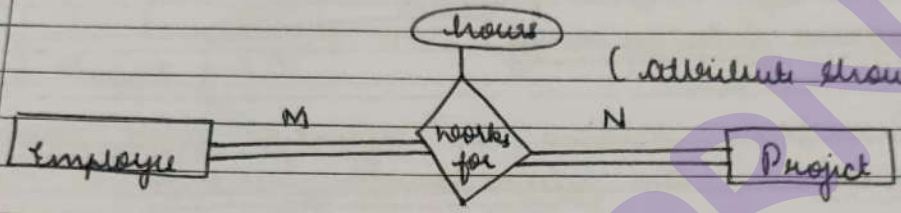
Attributes for relationship :-



so attach the relationship attribute to employee (always add toward N side)



(move to any side)
↳ entity.



(attribute should be under relationship)

(employee works in many project)

(difficult to move on either attributes)

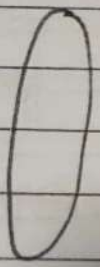
if info like how many hours did employee work on project)

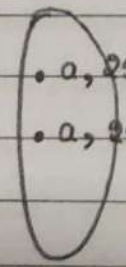
Weak Entity

→ Entity which do not have a key to uniquely identify the tuple (row).

emp (Strong entity)

dependants (weak entity)





- a, 25, son
- a, 25, son

} can have same name and same age of some other employee.

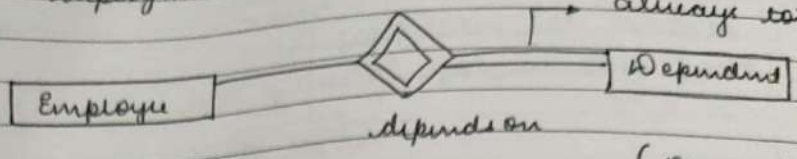
this is the prob if there is no key of its own.

classmate weak entity related to Strong entity

→ relationship :- identifying relation.

DATE

Employee may not have any dependents. But, for a dependent he/she should be associated with employee.



(Owning entity) (Owned entity)

Take key attribute of employee to dependent.

Problem Solving on ER.

Description related to research and sponsored projects in an educational institute.

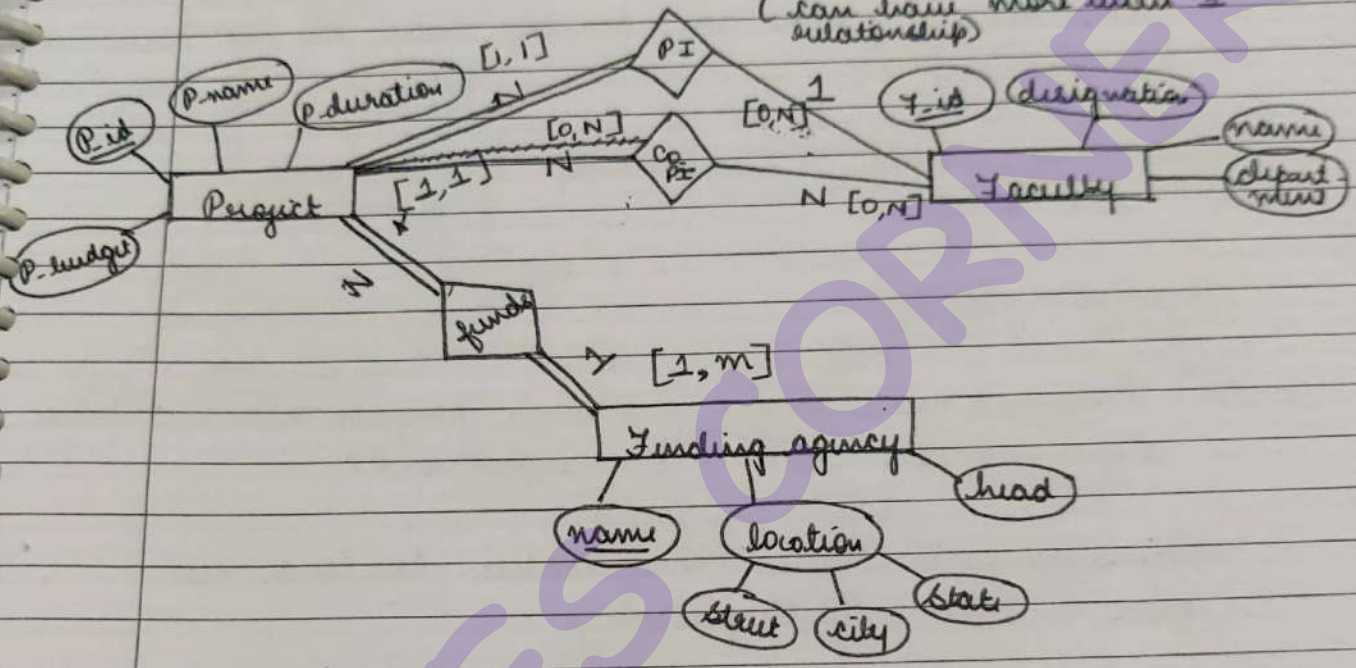
a) Institute has some sponsored projects.
b) Projects will have one faculty as Principal Investigator (PI), and can have one or more faculty members as Co-PI's
c) A project has Project ID (unique), Project name, Budget, duration as attributes.
d) Faculty are identified by Faculty ID (FID) (unique) and have name, dept, designation as other attributes.
e) A project is funded by only one funding agency (like UGC, DIT, DST) which has - agency name (unique).
f) Head location (with street, city and state as sub components)
g) A faculty, as a PI can have zero to any number of projects.
h) Similarly a faculty as a Co-PI can have zero to any number of projects.
i) A funding agency might have funded one or more projects.

V. * [Here PK or FK does not come into picture, that is only in relational model. Here entity type can have any no of keys.]

DATE

Draw an ER diagram for the above requirement. Assume necessary data which is missing in the question. The model should include entity types, relationships, min-max, cardinality, participation and other relevant constraints.

① Entity type :- project, Faculty, Funding agency.
(can have more than 1 relationship)



① all are strong entities.

② Project & funding agency relation :-

(Only if it funds project [1 1] we include FA.)

→ [1 N] 1 FA can fund N projects
1 Project can be funded by only 1 FA.

FA [n] : P [1]

③ Project ——(PI)—— Faculty.
&
(Co-PI)

DATE

1 project can have 1 PI.
1 project can have 1 or more Co-PI.

Faculty can be PI/Co-PI for many projects.

Faculty
[N : 1]
project.

considering : there are faculty who do not have any project.

Project [min, max]
(PI) [1 , 1]
(1 pi)

Faculty [min, max]
[0 , 1]

(Co-PI) Project [min, max]
[0 , N]

some projects may not have any Co-PI.

whereas we have min value to be 1. we give a total participation.

Participation :-

a) Every project should have a funding agency and every funding agency will fund a project.
So → total participation.

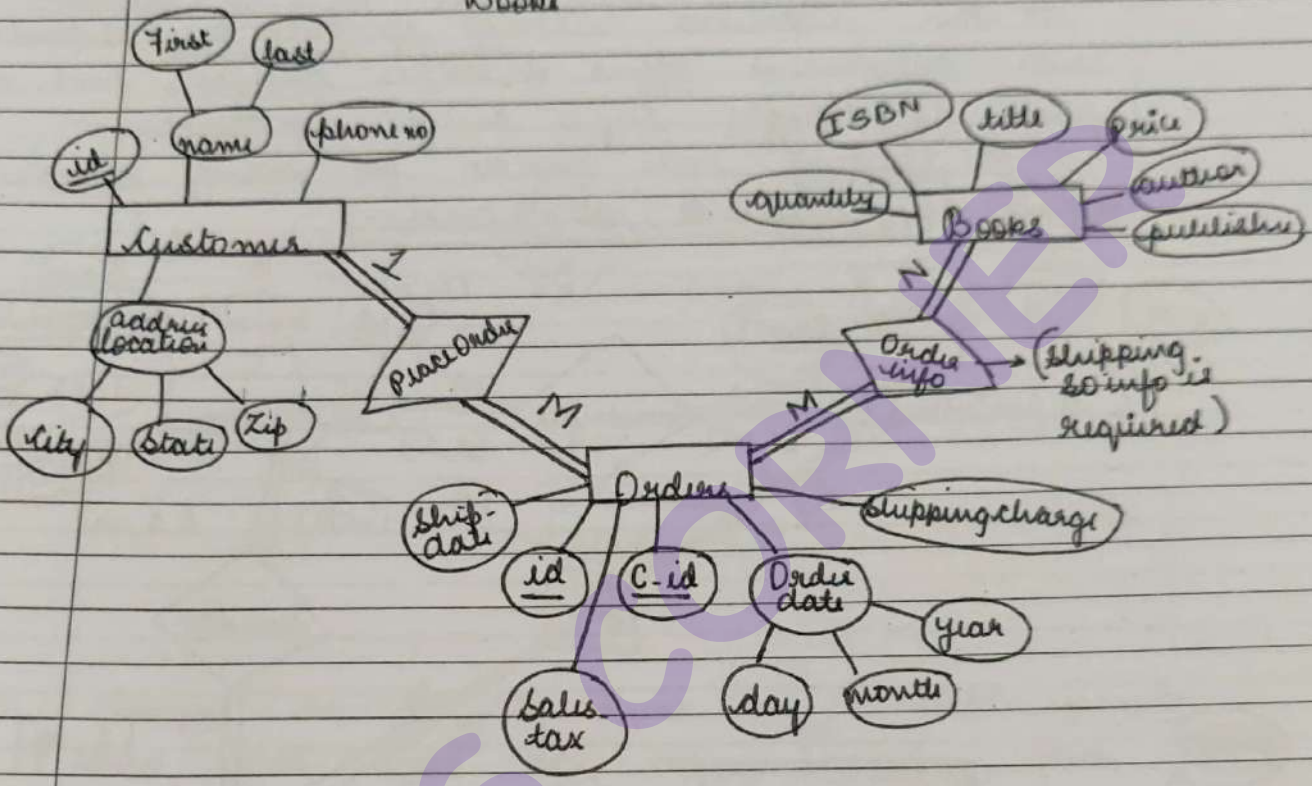
b) Every project must have a PI and not necessary that all Faculties are PI. partial.

c) Every project its not compulsory to have a Co-P
(so partial dependency.)
Participation)

2. Eurist book database

Eurist books business revolves around customers placing orders for books and Eurist books shipping the books specified in the order.

Entity types - Customers,
Orders,
Books



cust [1, m] order [1, 1]

Rules assumed.

① A customer can place multiple orders, but an order can be associated with only one customer.

② An order can have multiple books, and a book can have multiple orders.

cannot lead to unique solution; can change with different designer.

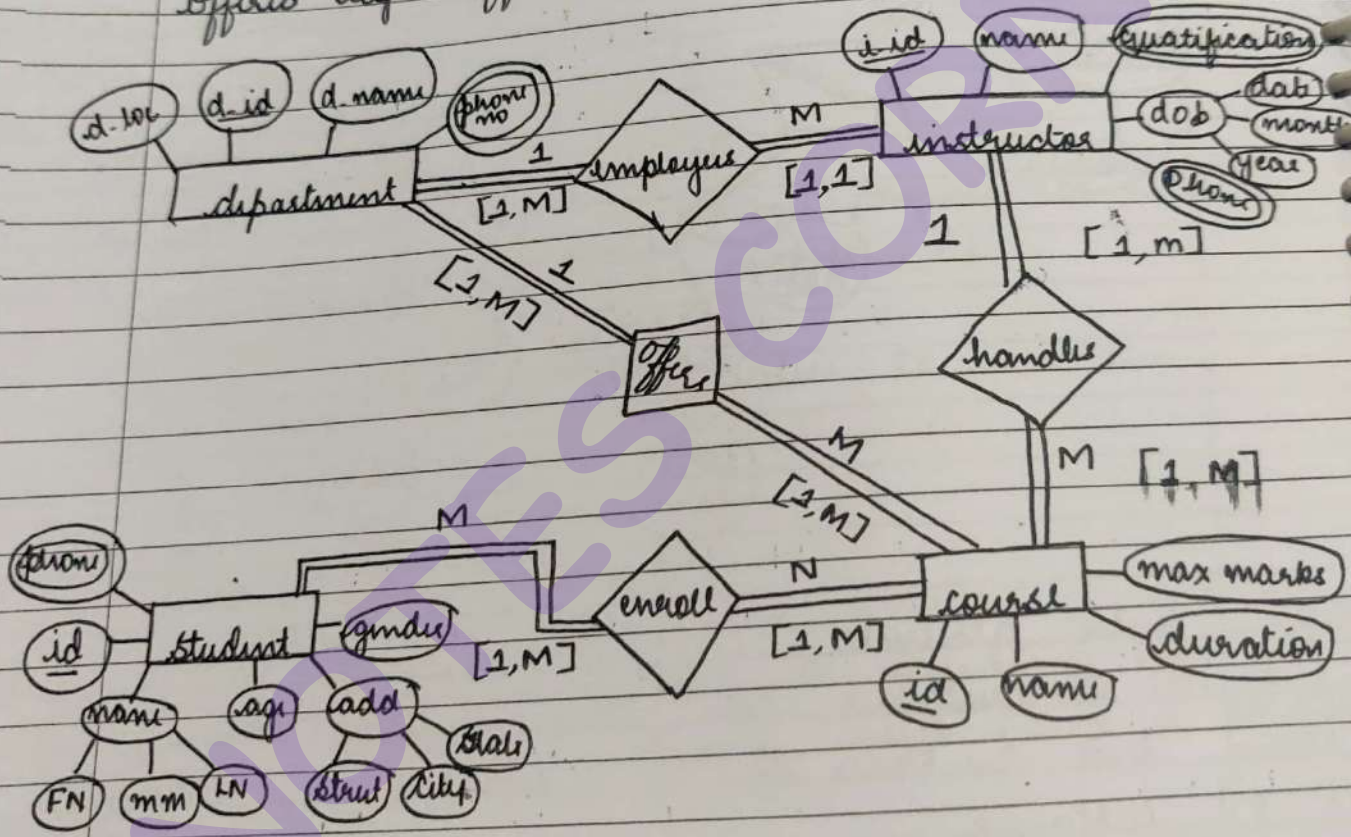
classmate

PAGE

MEGHANA G RAJ
ASSISTANT PROFESSOR

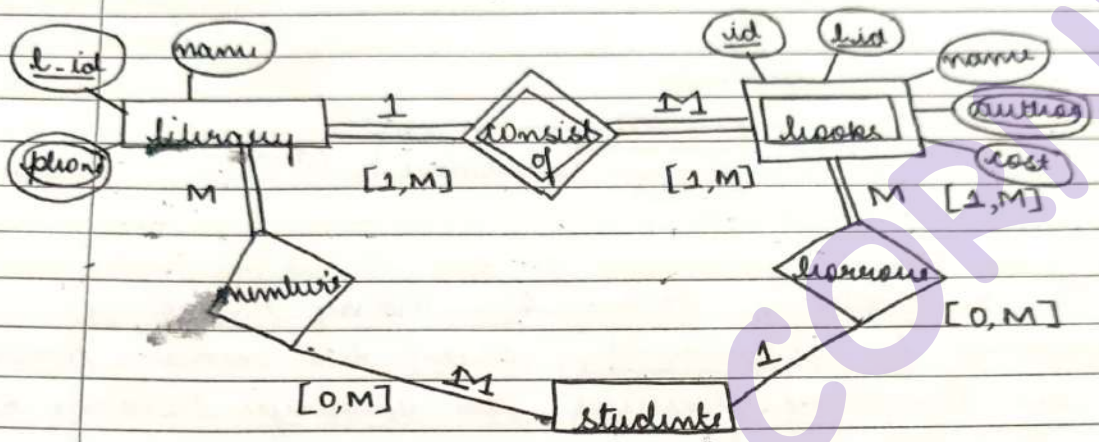
Condition:- that all entity types must participate (to total participation.

3. University data base has the following requirement specification.
   a) a university has many departments
   b) each department has multiple instructors
   c) an instructor belongs to only one department each department offers multiple courses, each of which is taught by a single instructor
   d) a student can enroll for many courses offered by different departments.

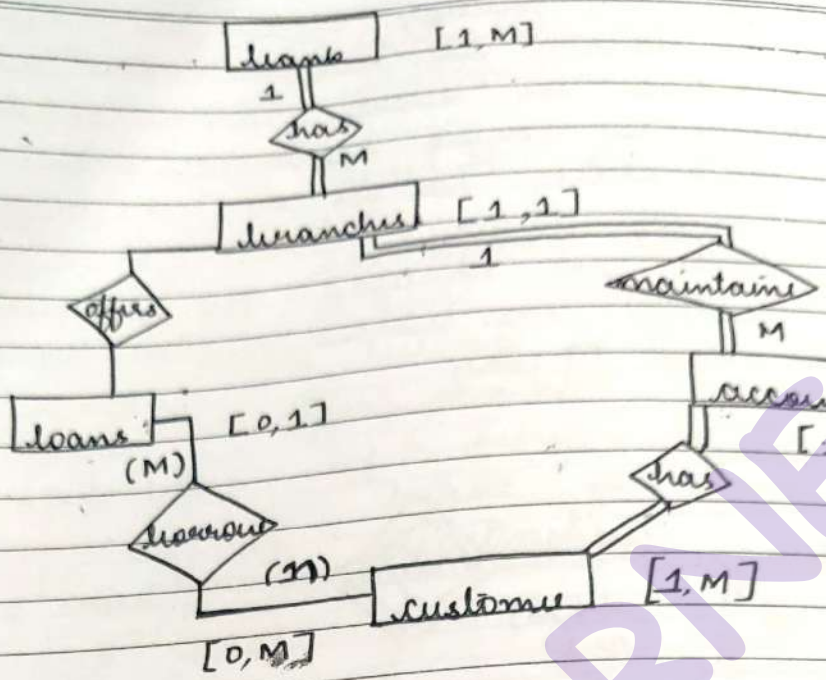


DATE

4. In the university
a) there are multiple libraries
b) each library has multiple student members.
c) students can become members to multiple libraries by paying appropriate membership fee.
d) each library has its own set of books within the library.
e) books are identified by unique no.
f) students can borrow multiple books from the subscribed library.



5. Assume in a city there are multiple banks and each bank has many branches, each branch has multiple customers.
a) Customer has various types of account
b) Some customer's also have different types of loans from the bank branches.
c) One customer can have multiple accounts and loans.



6. In a company there are many employees working for departments, there are many departments in the company however an employee belongs to only-one department.

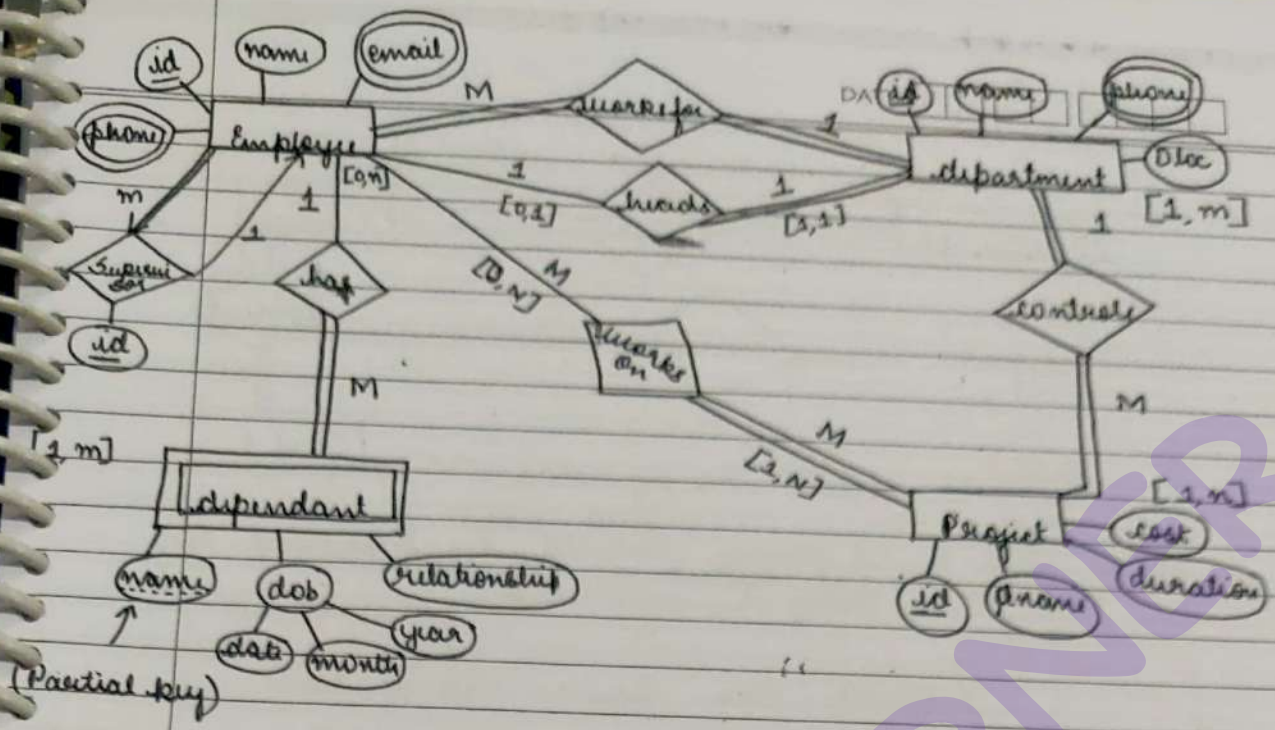
b) each department controls many projects, however every project belongs to only one department, many employees work on a project.

c) the employees dependent details also must be present.

d) employee can work on more than 1 project.

e) One among the employees would be a supervisor.

f) every department is managed by a head.

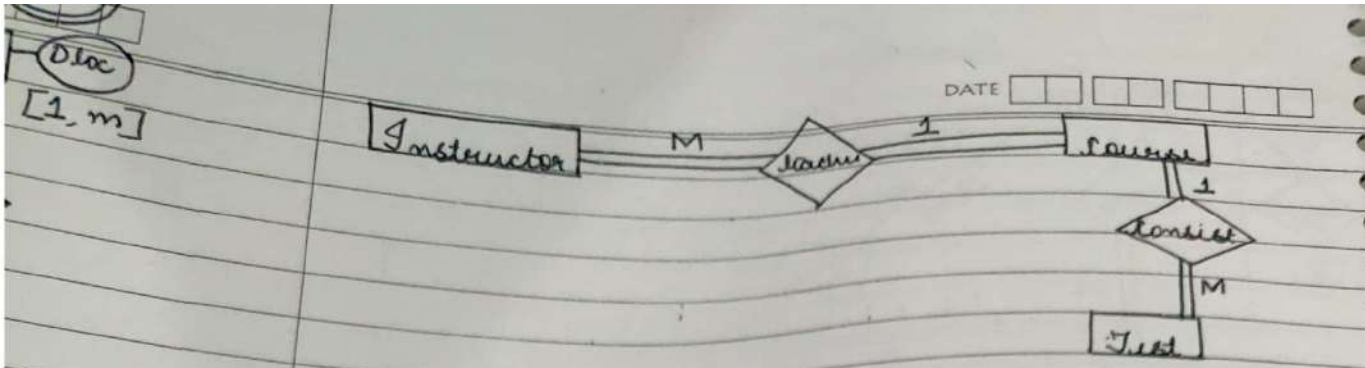


Assumption :- not a necessary condⁿ that employee must have dipendants, and a dipendant must be associated with an employee.

'm' → total participation.
1 → partial participation.

Various assumption are made while drawing the ER model.

(7) Establish cardinality and participation for the following section of the ER diagram. Mention the assumptions made explicitly.

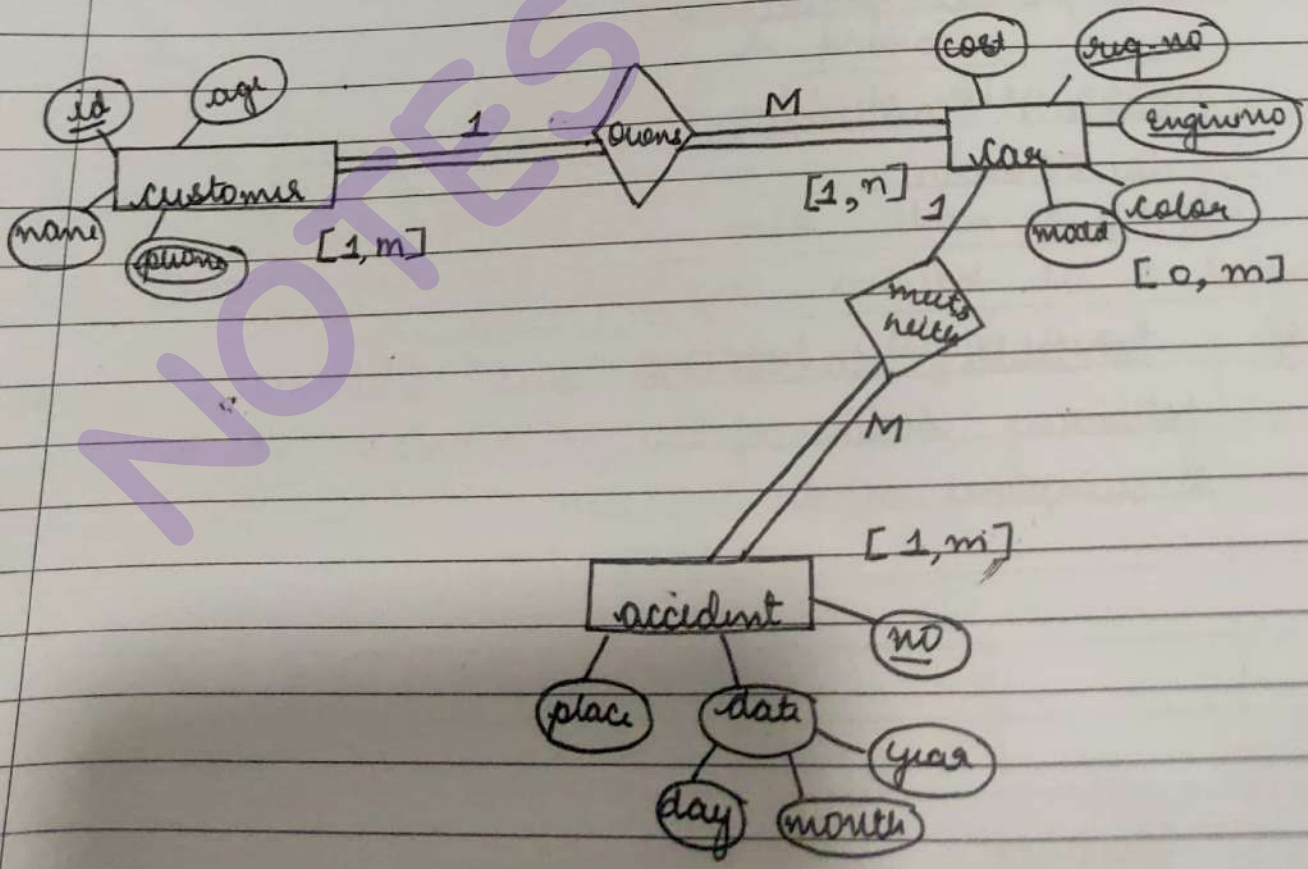


assumption :- 1 course should be handled by only 1 instructor.

1 course can have many tests.

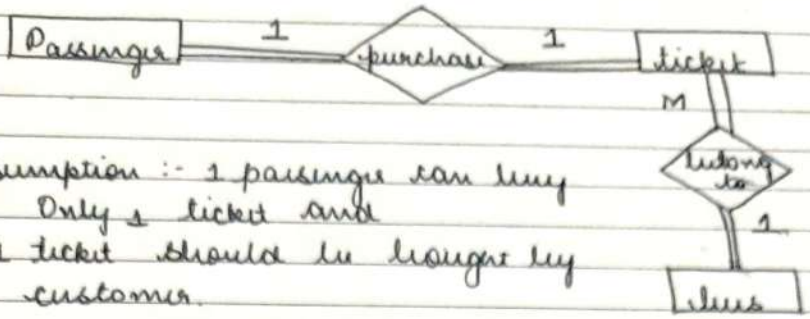
Each instructor must teach a course.

⑧ Car insurance company with a set of customers each of whom owns a number of cars, each car has a number of recorded accidents associated with it, mention any assumption made explicitly.



DATE

⑨ ER for bus reservation system.



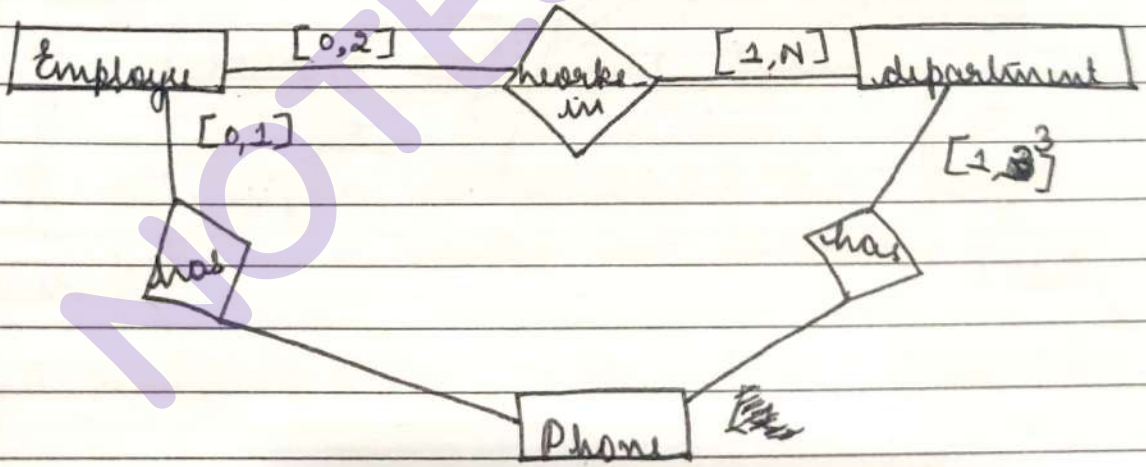
assumption :- 1 passenger can buy only 1 ticket and 1 ticket should be bought by a customer.

⑩ ① Assume that an employ may work in upto two departments or may not be assigned to any department.

② Assume that each department must have one and may have one and upto three phone number.

Supply min, max constraints.

Any assumptions made explain explicitly.



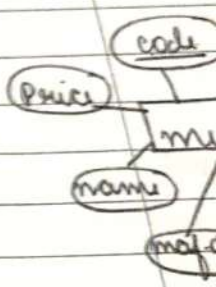
⑪ Mod

Entities

M:M same and it will n

M:M is as a industry

1:M is industry



DATE

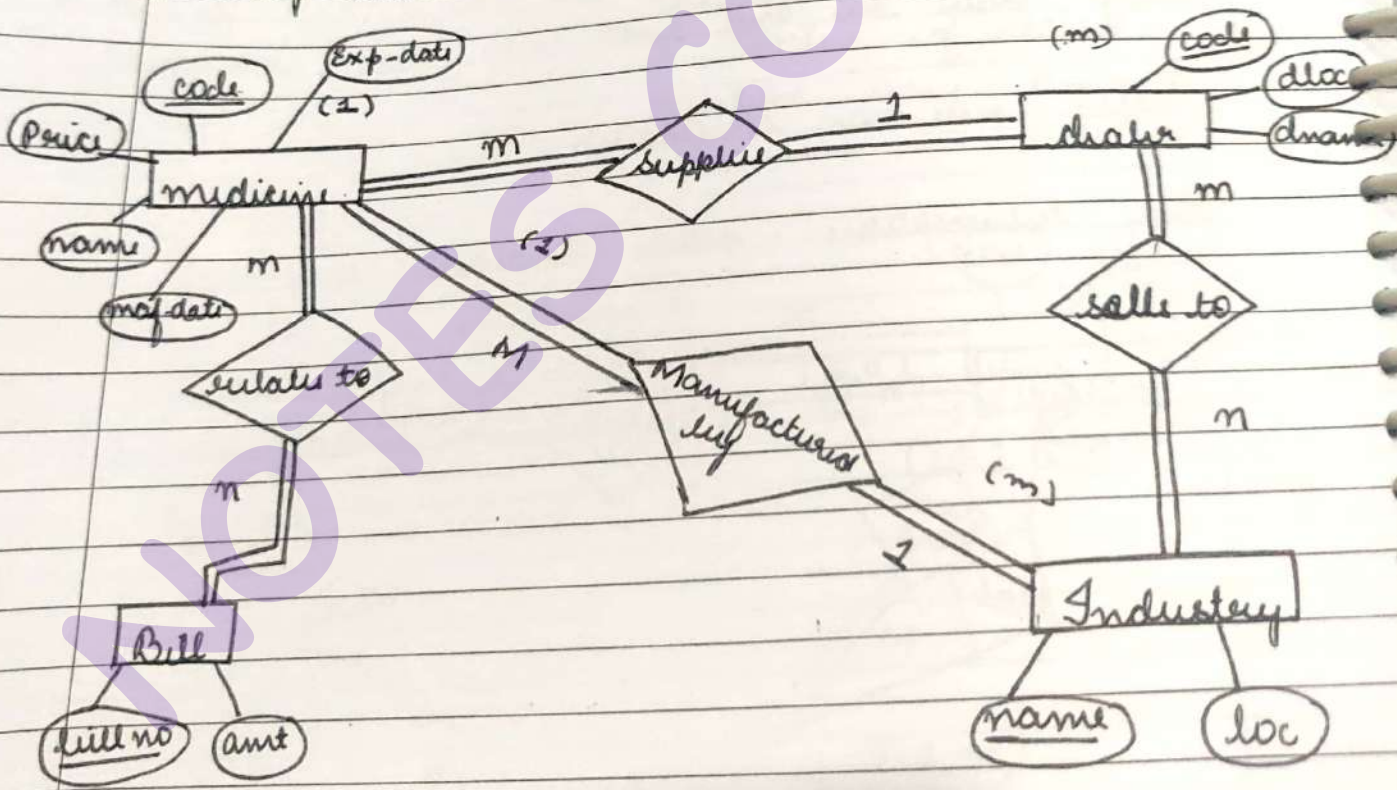
(11) Medicine transaction System.

Entities :- medicine, dealer, industry, bill.

M:M exists between medicine and bill, as in the same bill no. more than 1 medicine can be sold and the same medicine can be sold in the more than one bill in the same date or different dates.

M:M relationship also exists type Dealer and industry as a dealer keeps medicine of more than one manufacturer industry and the medicine of one industry can be sold by more than one dealer.

1:M relationship exists between dealer and medicine, industry and medicine.



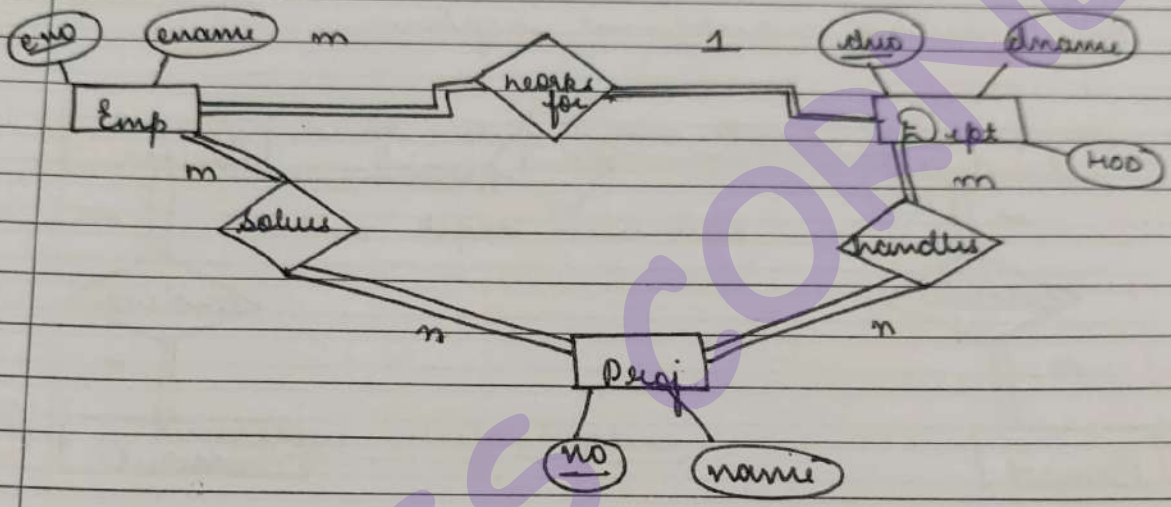
DATE

(12) Company E.R

Problem :- m:m relationship betn dept and Proj as a dept can handle more than one project and same project can involve more than 1 department.

Emp and Proj as an employee can handle more than one project and a project can be executed by more than one employee.

1 to m relationship exist betn dept and emp.

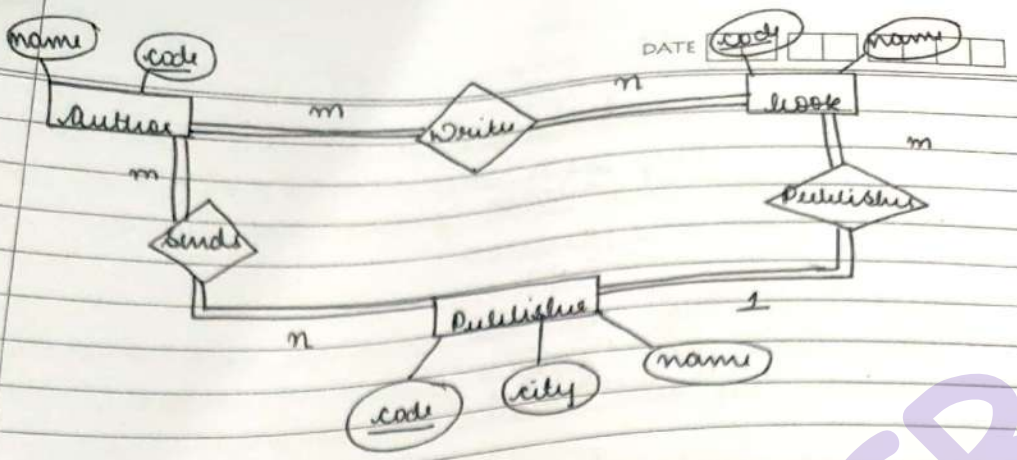


(13) Publication managment System

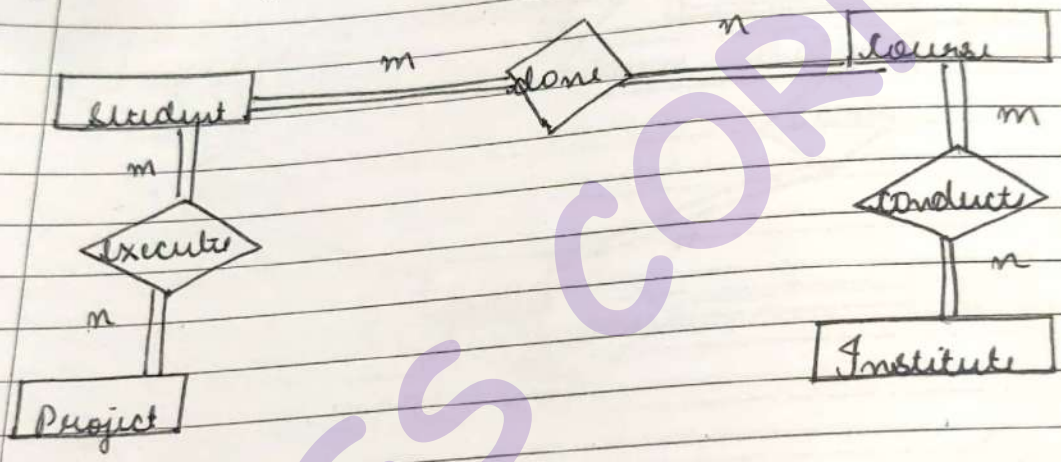
m:m between (author and book)
(author and publisher)

1:m relationship exist between (publisher and book)

(14) Tra



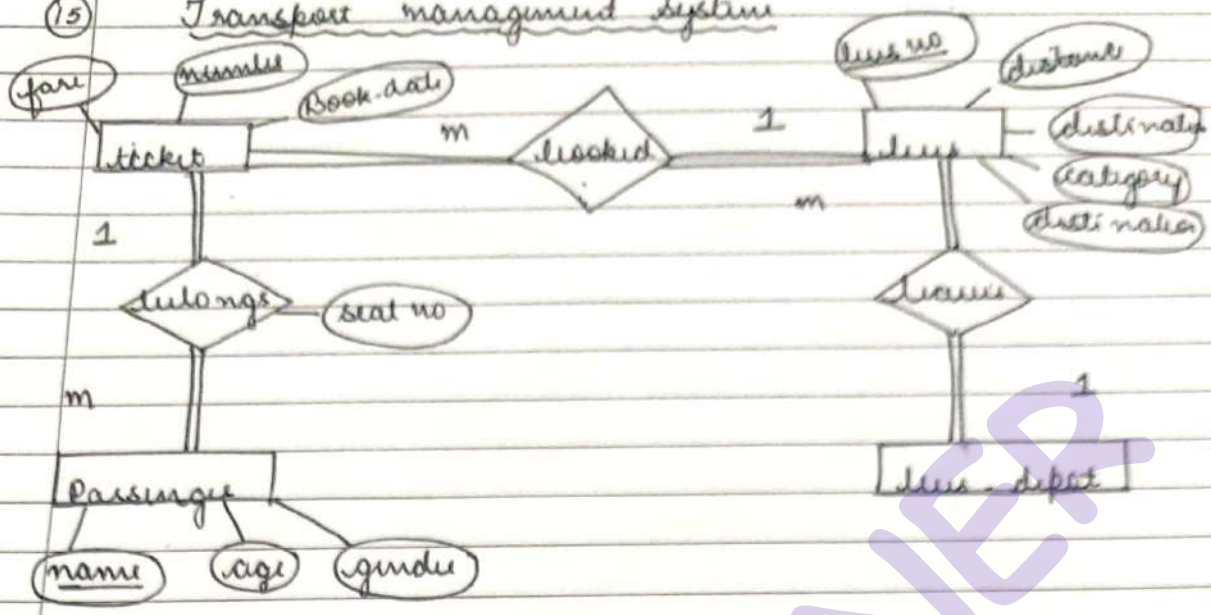
(14) Training managment system.



m:n between student and course
student and project
course and institute

DATE

(15) Transport managment system



1:m relationship b/w ticket and passenger
bus & ticket
bus depot and bus

(16) Employee info system

1:m relationship exist between department and employee (dept has more than 1 employee.

Employee can posses more than 1 skill and also more than 1 employee can have the same skill resulting in m:n relationship b/w employee and skill.

An employee can have more than 1 qualification and the same qualification can be acquired by more than 1 employee.

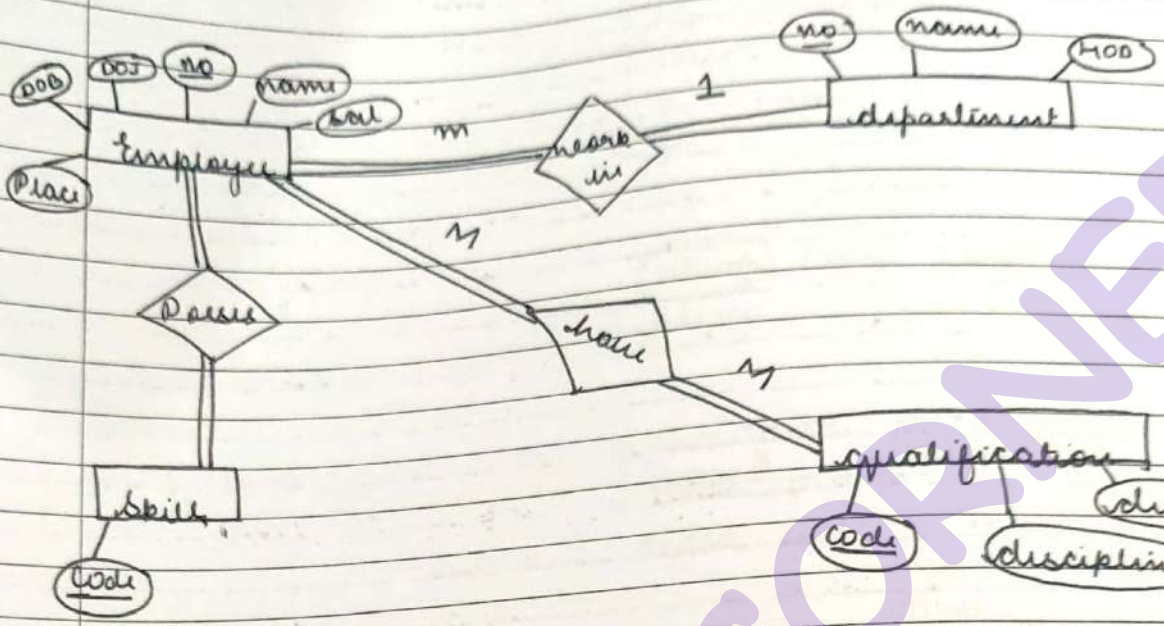
classmate

PAGE

DATE

resulting in m : m relationship between emp and qualification.

1 : m exists between department and employee.

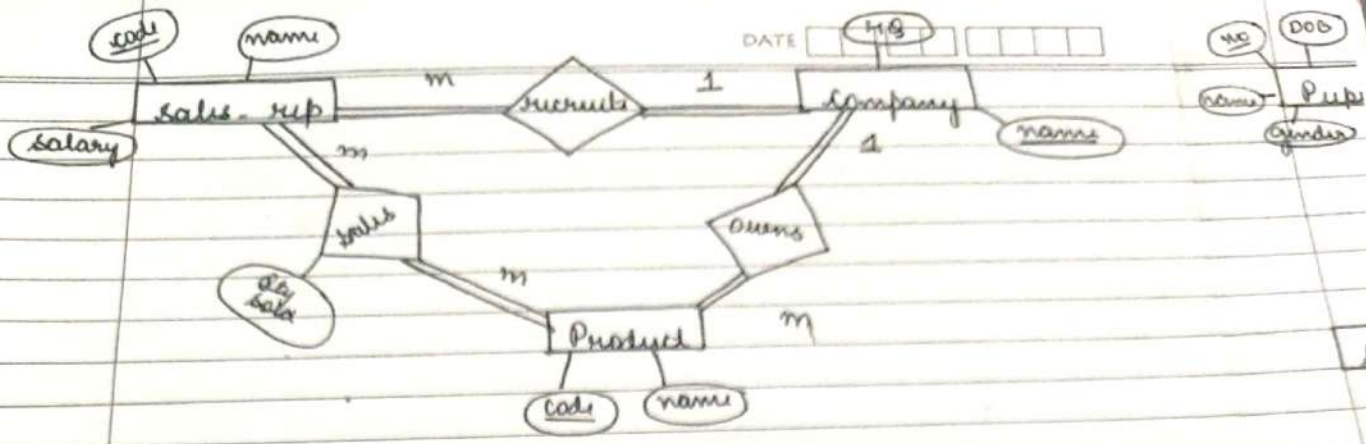


cond^{n} imposed : every employee should work in any one of the departments.

(17) Sales tracking system

Problem :- A company can have more than 1 type of products and in the same fashion same type of products can be launched by more than one company, resulting in m : m relationship btwn company and product.

A product is marketed by more than one sales representative and in the similar fashion a sales representative can market more than 1 product resulting in (m to m) relationship btwn product and sales. rep.



m : m b/w sales rep and product
Product and company

1 : m b/w company and sales rep.

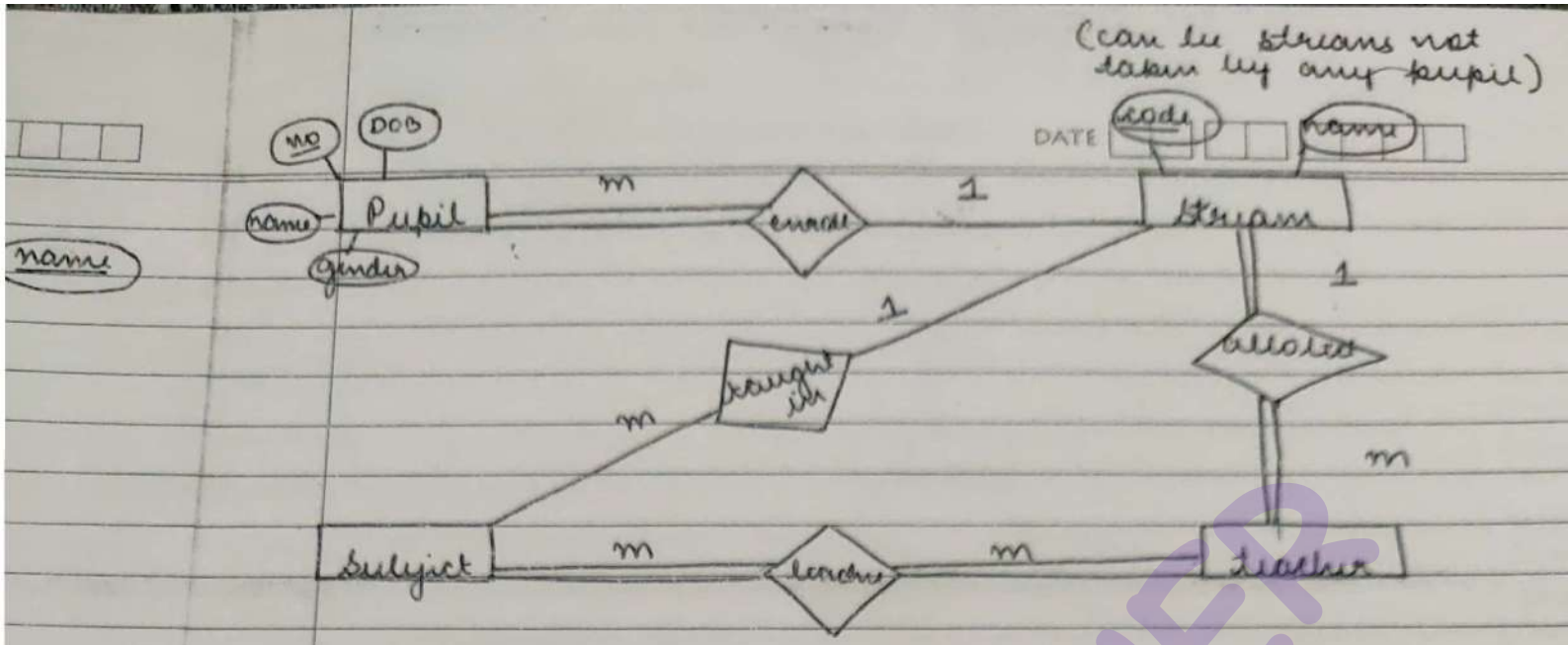
(18) Teaching activity information system.

1 : m b/w Stream and pupil
(can have many pupil)

1 : m b/w Stream and teacher
(1 teacher belongs to only 1 stream)

1 : m b/w Stream and subject

A teacher can teach more than 1 or more subjects and different subjects topics can be taught by more than 1 teacher creating many to many relationship b/w teacher and subject.



(19) Patient diagnostic info system

1 : m line outdoor and patient

↑

(more than 1 patient is treated)

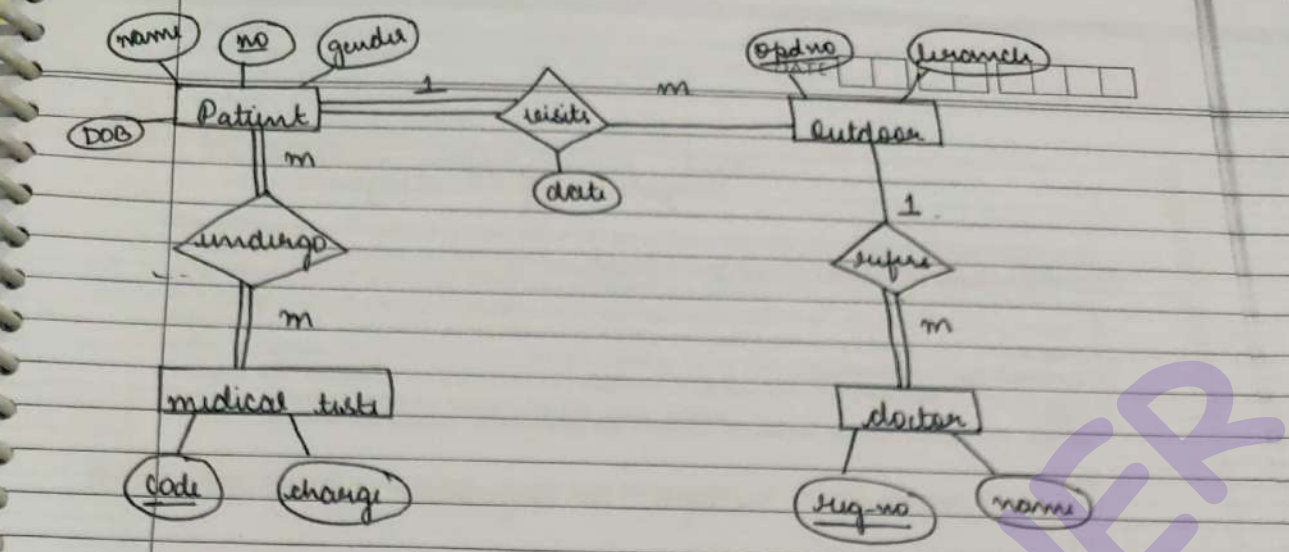
1 : m line outdoor and doctor.

↑

(more than 1 doctor registered to treat patient)

m : m line patient and medical test.

a patient can undergo more than 1 te and same test can be done by more th patient resulting in m : m line patient and medical test.



① Mid

Step 1
Read the

Step 2
Identify

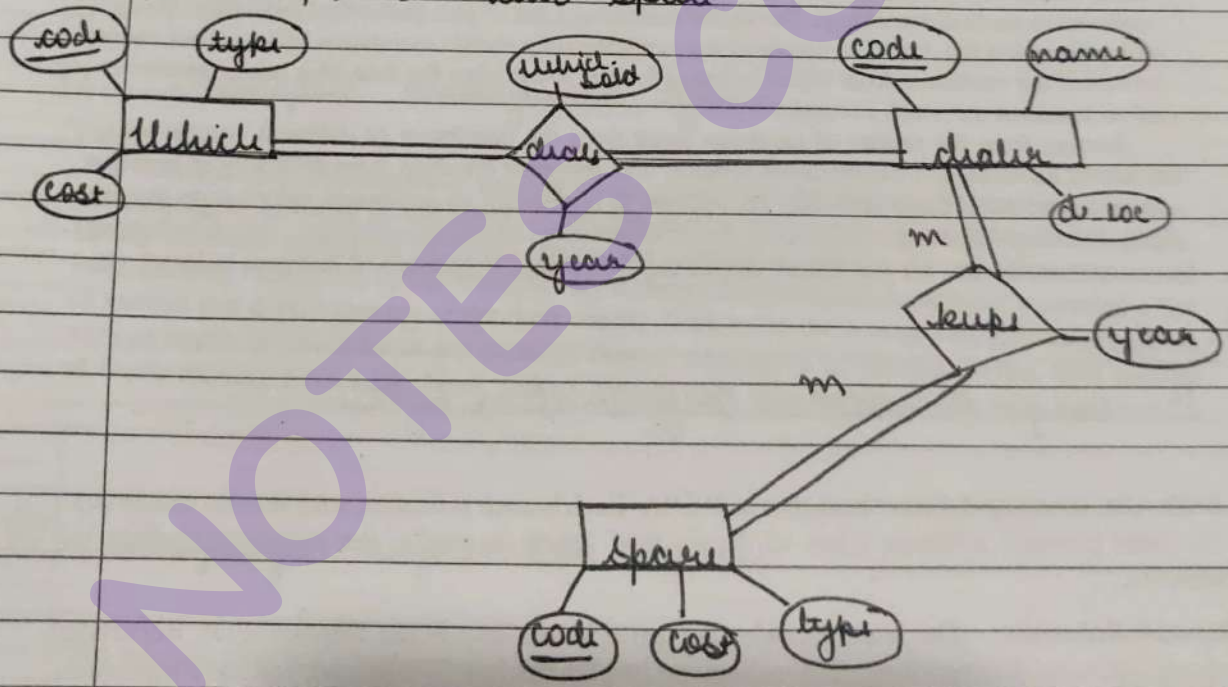
Step 3
Draw th

Step 4
Create

Step
Solv

(20) Automobile sale tracking system

m:m relationship exists between vehicle and dealer, dealer and spare



m:n between dealer and vehicle.
dealer and spare.

(designing for complex domains)

DATE

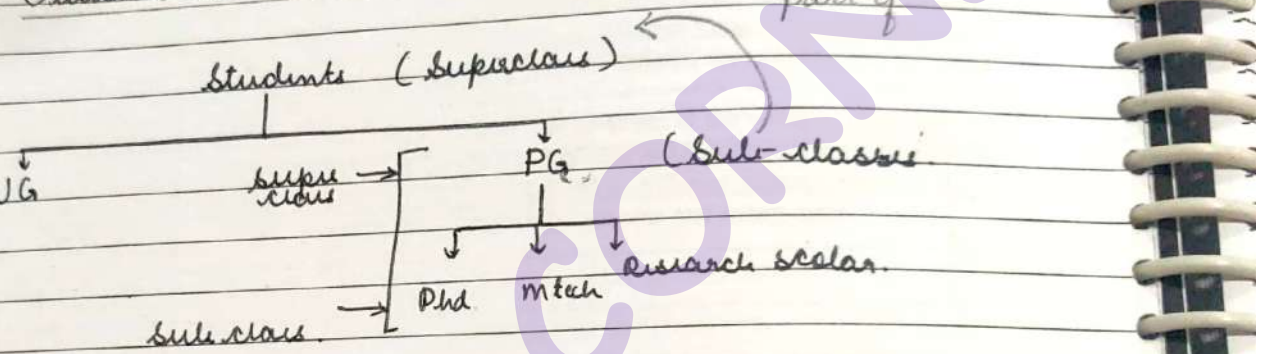
(Enhanced)
Extended Entity Relationship diagram.

Why EER? → used to design for complex applications like telecom, scientific applications.

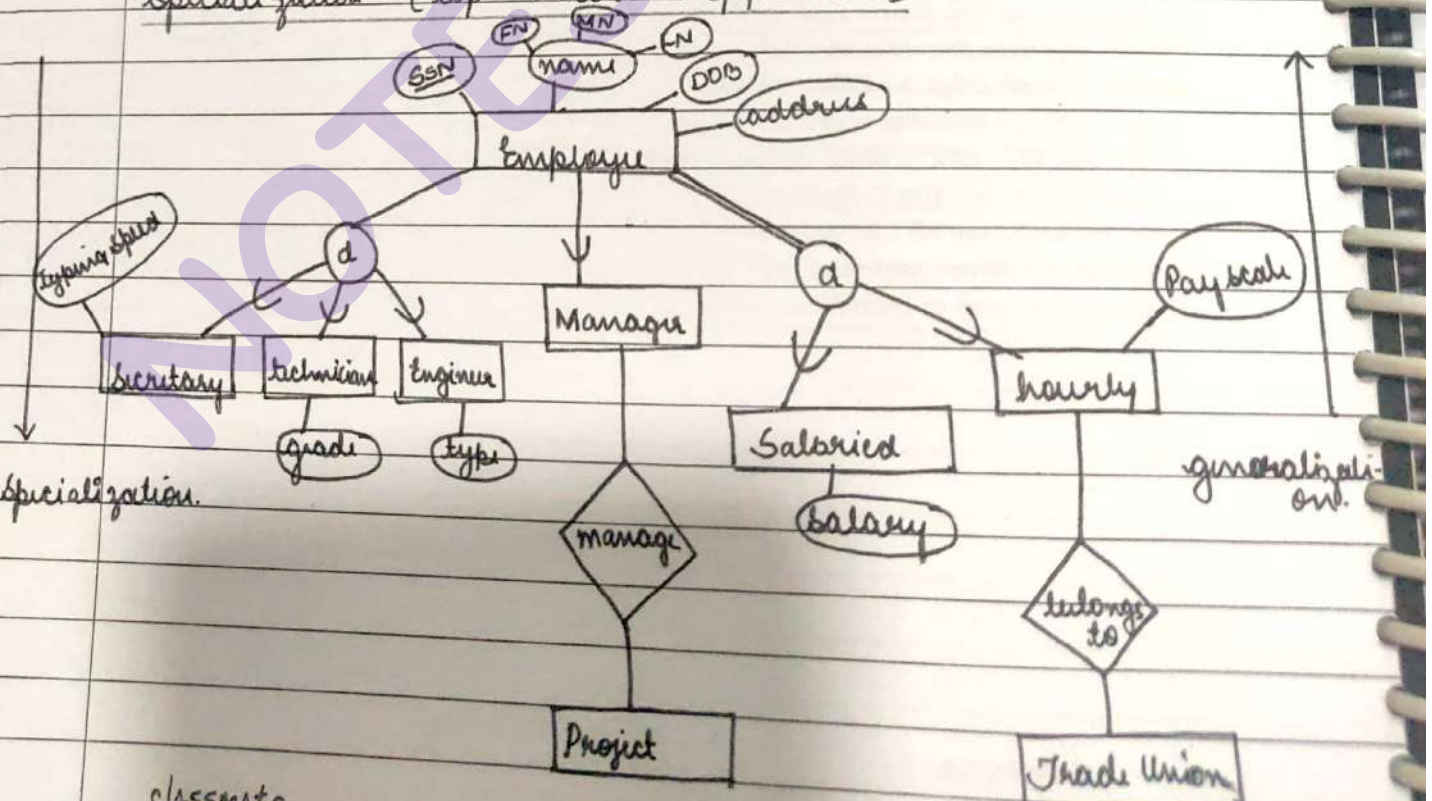
Additional concepts used in EER (designing class hierarchy)
a) Subclass / Super class
b) Specialization / generalization
c) Attribute and relationship
a) Inheritance

} fundamentals of conceptual modeling.

Subclass and Superclass.



Specialization (top down approach)



DATE

Definition : [By arrow position/direction]

① **Specialization :-** is the process of defining a set of subclasses of a superclass.

② **Generalization :-** is the reverse of the specialization process. Several classes with common features are generalized into a superclass. Original classes become its subclasses (bottom up approach).

Basic

The to

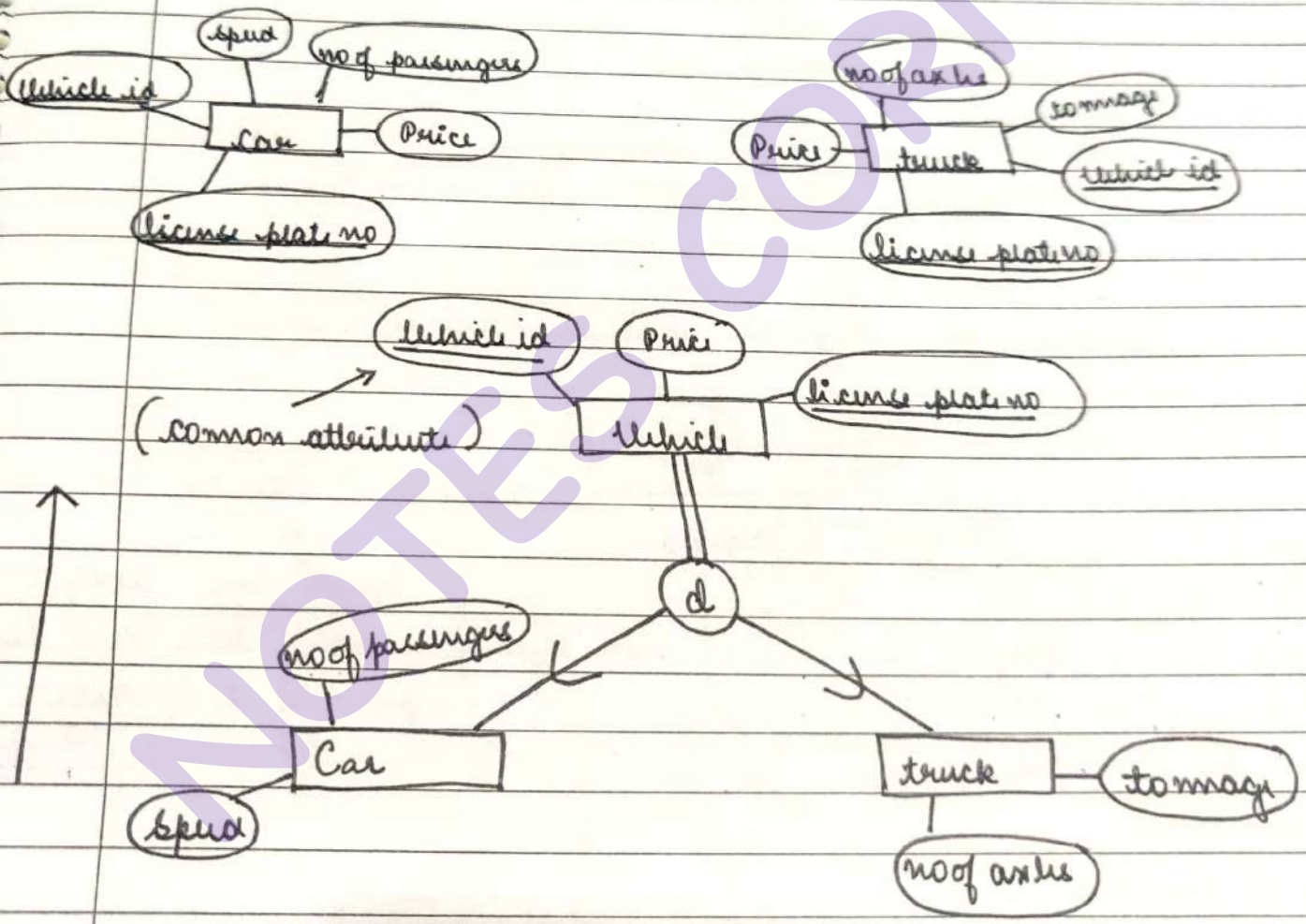
a) D is
b) Comp

Schemes

a) d
b) c
c)
d)

N

Generalization (bottom - up approach)



DATE

Basic Constraints.

The two basic constraints that can be applied to specialization / generalization:-

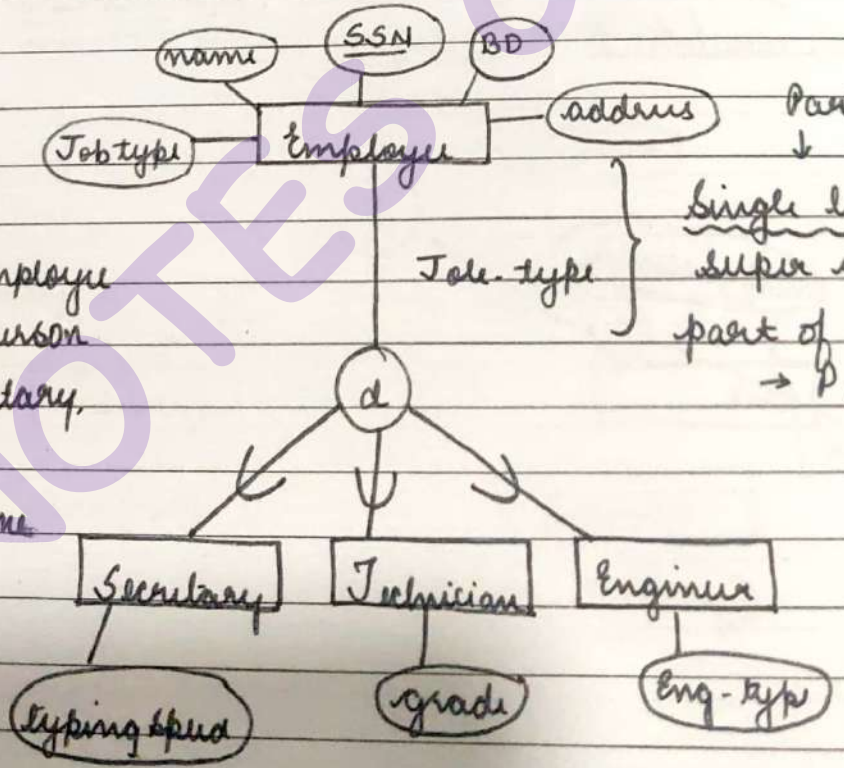
a) Disjointness constraint
b) Completeness Constraint.

Hence we have 4 types of specialization / generalization

a) disjoint, total
b) disjoint, partial
c) Overlapping, total
d) Overlapping partial.

Note:- Generalization usually is total because the superclass is derived from the subclasses.

Example for disjoint partial Specialization.



(So disjoint)
(if we take Employee instance, a person cannot be Secretary, technician and engineer @ the same time)

Job-type } Partial.
↓
Single line: every super class need be part of subclasses → Partial.

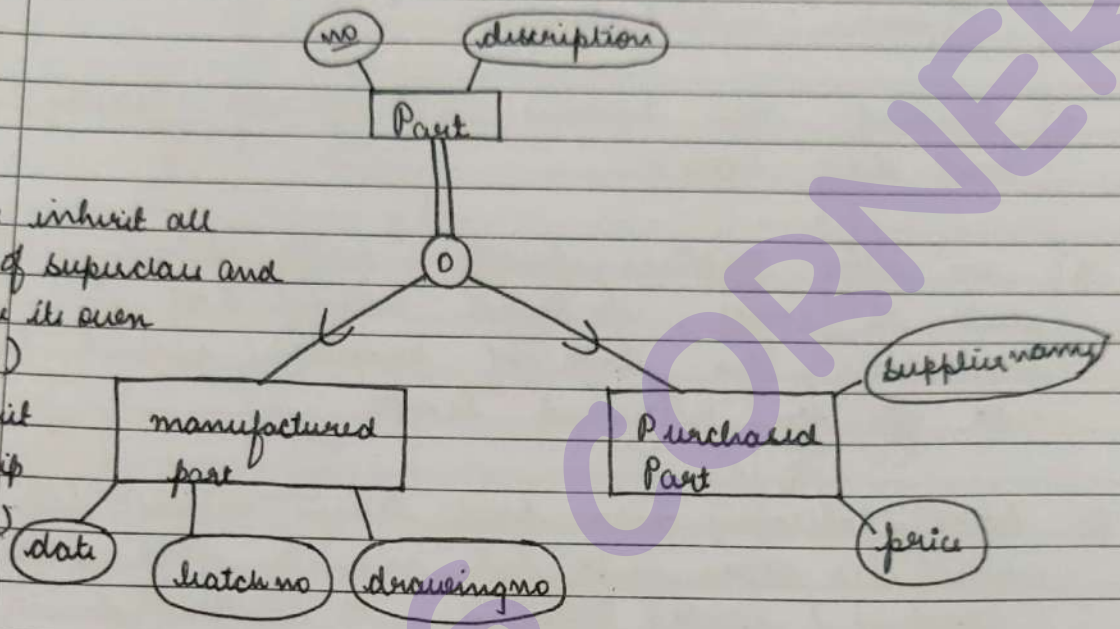
total participation :- that means any one instance of employee have to belong to either of 3 ~~that~~ categories. (anyone, not all)
group

DATE

Overlapping total specialization

total :- there is no instance in the superclass that does not belong to any of the categories.

overlap :- a part can be manufactured and purchased.



※ (subclass inherit all feature of superclass and can have its own property)

※ (also inherit relationship from superclass)

Example :-

For a library of a college, we need to design a database. The business rules are as follows:

a) library users can be student / faculty / staf (non-teaching). A user can borrow books from the library. User can belong to atleast one category and some times more than one.

b) Each book belongs to a category like Eng / managment / arts / science (only one category)

DATE

c) Each book has title, bookid, isbn #, publisher, editor, no of pages, price, no volumes, copies available for issue etc.

d) books are available in racks and each rack has a unique ID and location like :- ground floor / first floor / second floor and rack type (wooden / metal / synthetic)

e) Students can borrow upto 3 books, faculty upto 5 and staff upto 2.

f) We also capture books issued details with date of issue, expected return date etc. We store only the info of currently issued books, not for the returned books.

g) Each library user will have a user id, for users we store info like name, add, category (student / staff / faculty). A student will have branch, faculty will have department and designation, non-teaching staff will have pno ~~fno~~ and division as specific attributes.

Draw EER diag as appropriate for the above discription. Indicate - cardinality, keys, attributes min max and participation constraints for entity types involved in the relationships

Assume necessary data.

name → entity type.

※ if no attributes have to be captured for subclass, no use in having them as an entity.

classmate

MEGHANA G RAJ
ASSISTANT PROFESSOR

PAGE

Date : Page :

Topic :

(d)

disjoint :- Instance of superclass, may not be part of multiple groups. (or may not be part of any subclass)

participation :- partial (Instance of employe need not belong to any of subclasses)

total (Every instance in superclass belongs to atleast and atmost one subclass)

Employe		total participation	partial participation
101	A	Secretary	Secretary
102	B	Engineer	-
103	C	Technician	-
104	D	Engineer	-

Overlapping :- (o) Superclass instance may belong to more than 1 subclass.

partial :- not all instances belong to two or more subclasses.

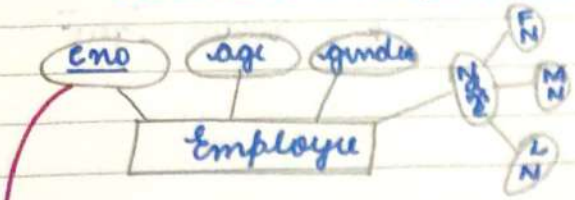
total :- all instance in superclass belong to two or more subclasses.

Employe.		partial	total
101	A	Engineer, Secretary	Engineer, Sec
102	B		Sec, Technician
103	C	Secretary, Technician	Sec, Technician
104	D		Eng, Technician

attributes :- inherit from superclass & can also have their own attributes.

relationship :- if superclass has relationship that is inherited by the subclass as well.

Conversion of ER diagram to table

① 

→ 3 simple attributes
→ 1 composite attribute

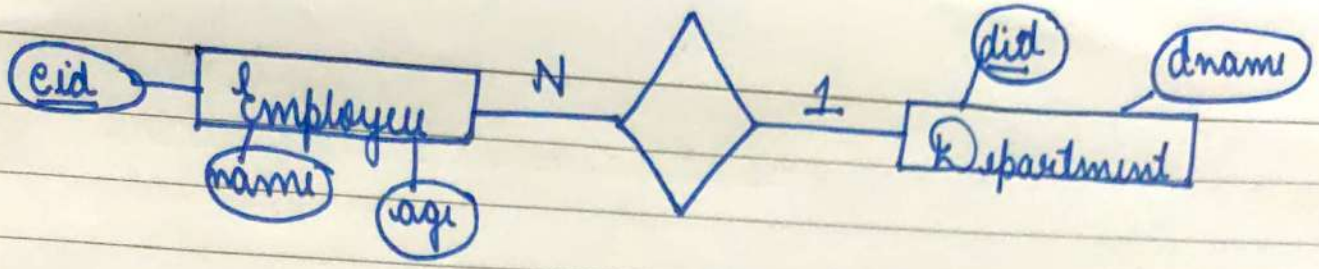
→ prime attribute (similar to primary key)

Employee (relation)

Eno	age	gender	first name	middlename	L

Rules :-
① Convert every entity will be made a relation
② the prime attribute is made as - primary key
③ Simple attributes be made as individual attributes in the table.
④ Composite attributes → broken down into sim attributes and must be included in the ta

② Conversion of 1 : M cardinality ratio into relational model.



Note :- always move the prime attribute (where the side is 1 side to N side.

Employee

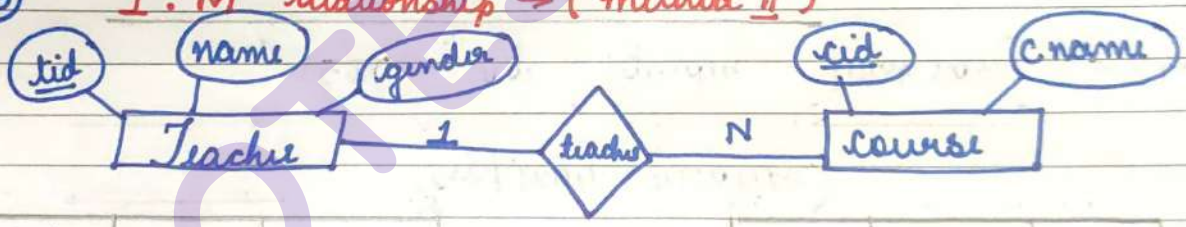
eid	name	age	did

Department

did	dname.

FK

(3) 1 : M relationship → (method II)



a)

tid	name	gender

cid	cname	tid

→ min no of table required is '2'.

b)

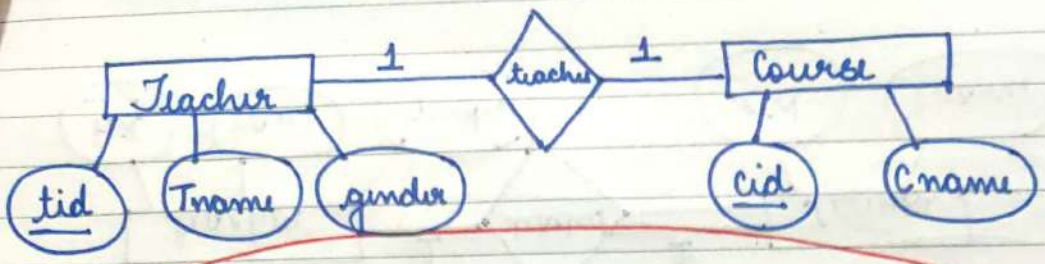
tid	cid

← max no of table required is 3.

tid	name	gender

cid	cname

Converting 1 : 1 relationship



a)

tid	tname	gender

cid	cname	tid

b)

tid	tname	gender	cid

cid	cname

min no of tables 2.

Converting 1:1 relationship (method II)

maximum no of tables 3

tid	tname	gender

cid	cname

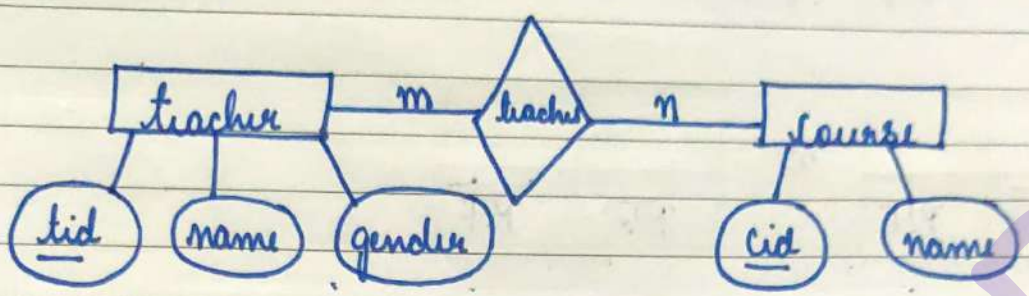
tid	cid

or

tid	cid

4 ways to represent 1:1 relationship

Converting m:n relationship



tid	name	gender

cid	name

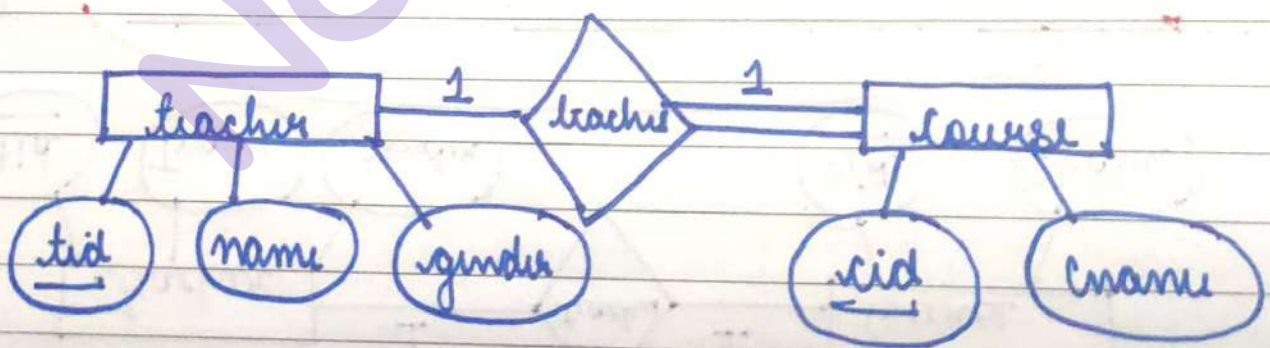
tid	cid

(references tid)

references (cid)

(min no of tables required is 3)

Converting 1:1 relationship with total participation on one side (applicable only to 1:1)



tid	name	gender

cid	cname	tid

(good approach and not a rule)

Note :- We always give importance to cardinality, participation does not really matter.
(Even (m:1) or (m : n) follow the same procedure, participation is not considered)

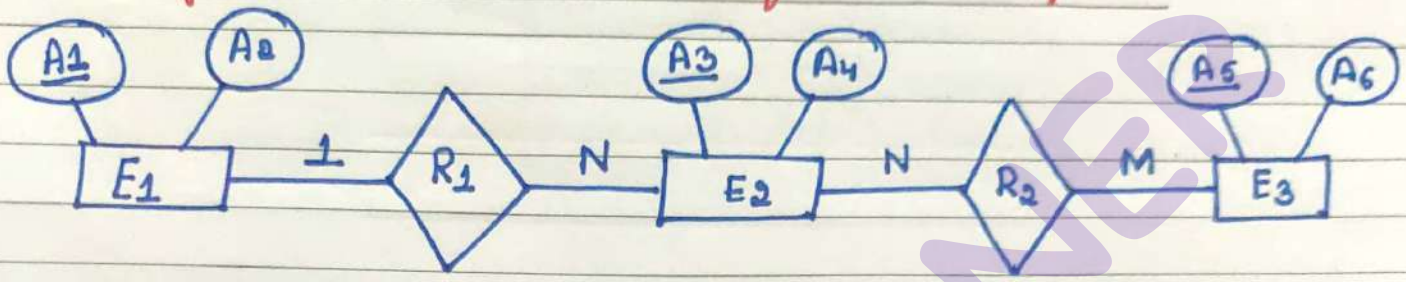
Multivalued attribute

① Multivalued attributes are allowed in entity relationship diagram.

② Multivalued attributes are not allowed when we convert to relational model (✗)

Problems

Finding minimum number of tables required.



E1

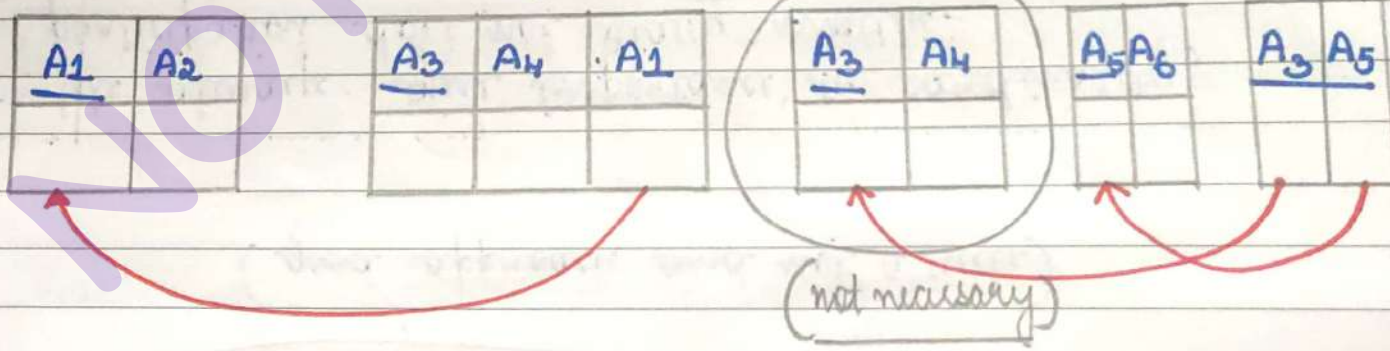
A1	A2

E2

A3	A4

E3

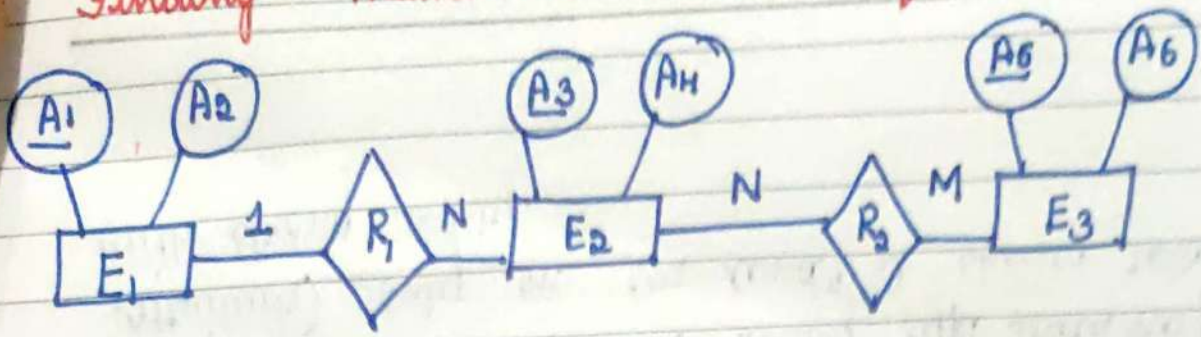
A5	A6



minimum no of tables required are

4

Finding maximum number of tables required



E1

A1	A2

E2

A3	A4

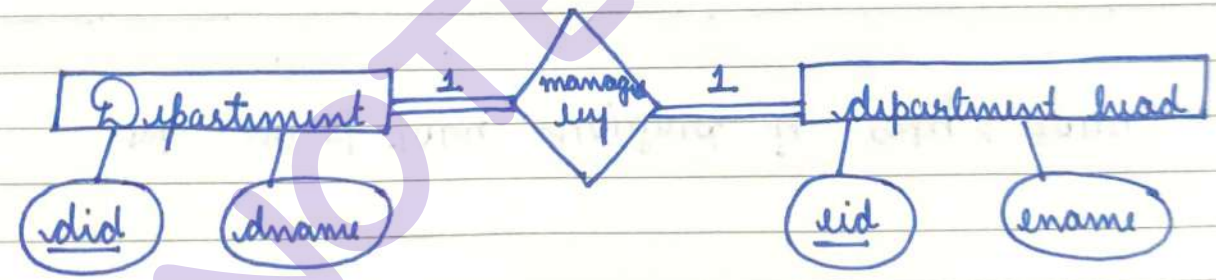
E3

A1	A3	A5	A6

A3	A5

maximum no of table required is 5.

1:1 relationship with total participation on both the side



minimum no of tables should be considered.

→ no of tuples in department = no of tuples in department head table.

did	dname

eid	ename

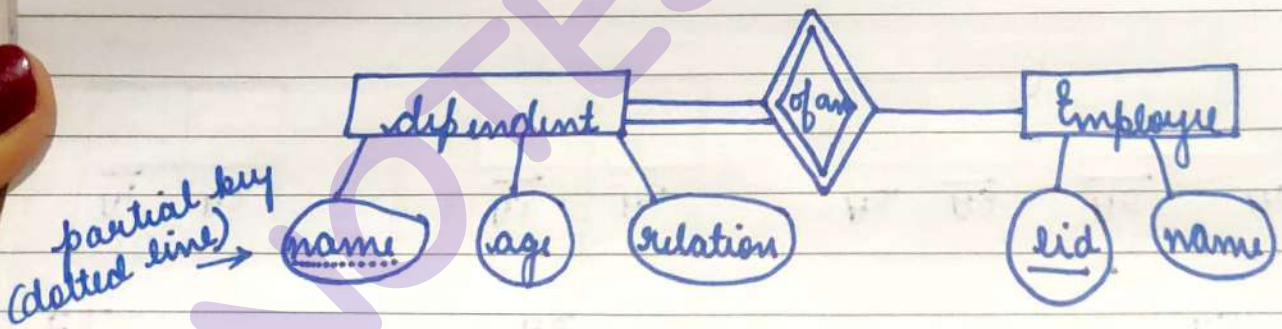
did	dname	eid	ename

(just one table)

min no of table required is only 1 table.

Weak v/s Strong Entity - diffrence

An entity with no prime attribute is called a weak entity



(Er model we can have entity type with no prime attribute) only on one cond" it should totally participate with Strong entity.

(cannot be PK, an employee can have more than 1 relation)

dependent

name	age	relation	eid

employee

eid	name

((eid, name) → key)

partial key :- (will not be able to single handedly identify a relation, given an emp id he cannot have more than 1 relation with the same name (this is the reason name is called a partial key)

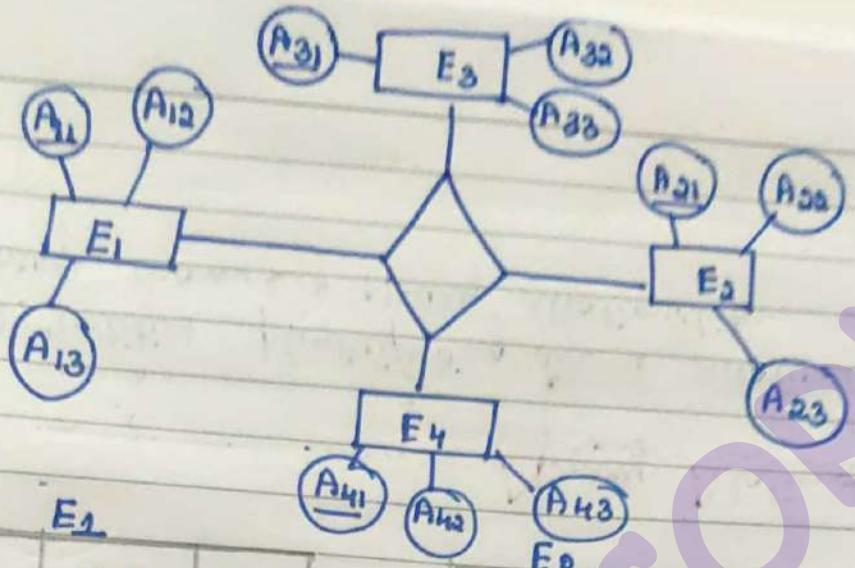
※ A weak entity should have partial key.

example :- 1 sam } not allowed ✗
1 sam

1 ram } allowed ✓
1 sam

N-ary Relationship

2 entities participating in a relationship → binary.
more than 2 → N-ary relationship.



(more than 2 entity types)

E1

A11	A12	A13

E2

A21	A22	A23

E3

E31	E32	E33

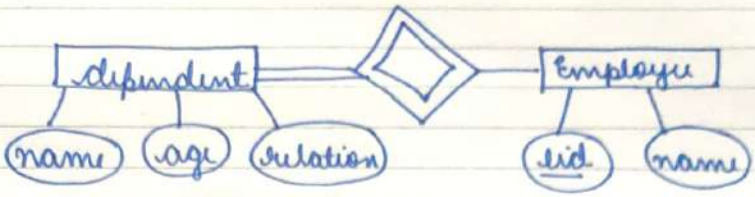
A11	A21	A31	A41

(to establish a relationship

E4

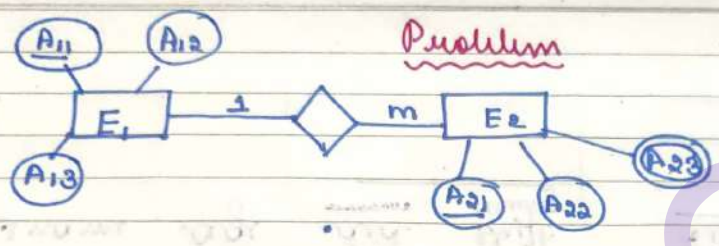
A41	A42	A43

Identifying relationship



Problem

(i)



E1

A11	A12	A13

E2

A11	A21	A22

A21	A23

(minimum no of tables)
= 3

A11	A12	A13

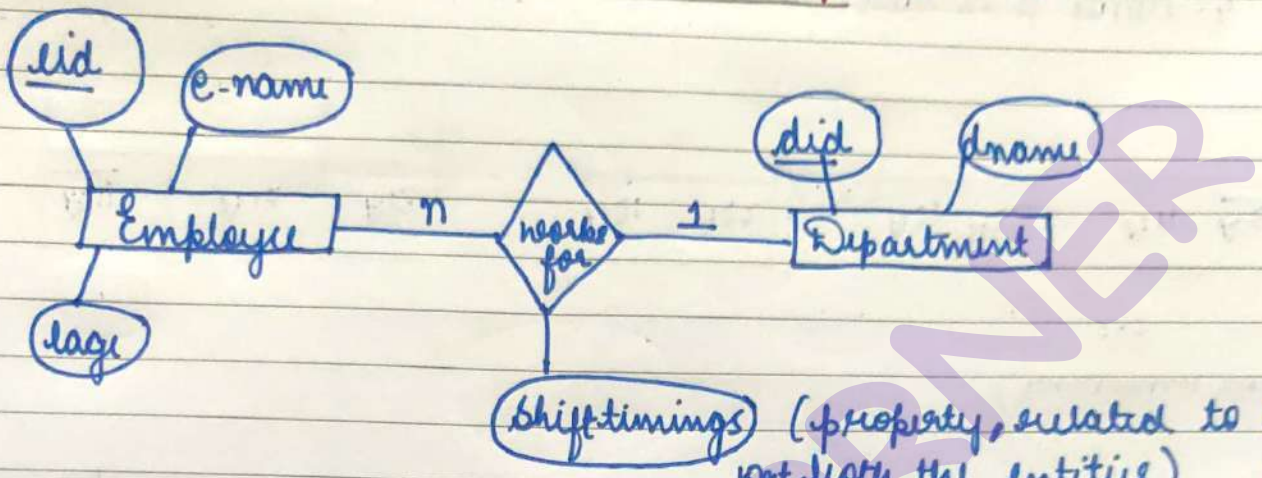
A21	A22

A21	A23

A11	A21

max no of tables = 4

Attributes to relationship (1:m)



Employee

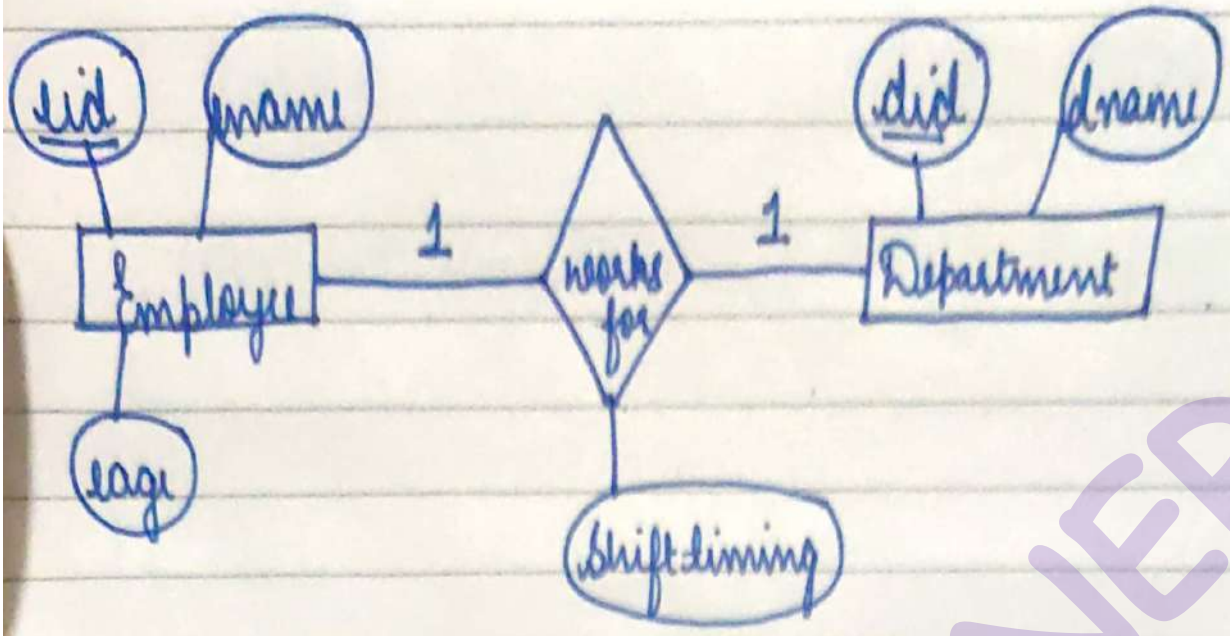
eid	ename	age	did	Shift timing

department

did	dname

(Note :- move the attribute of the relationship towards the 'N' side) (To avoid multi valued entries)

Attribute to relationship (1:1)



sid	sname	sage	did	Shift timing	did	dname

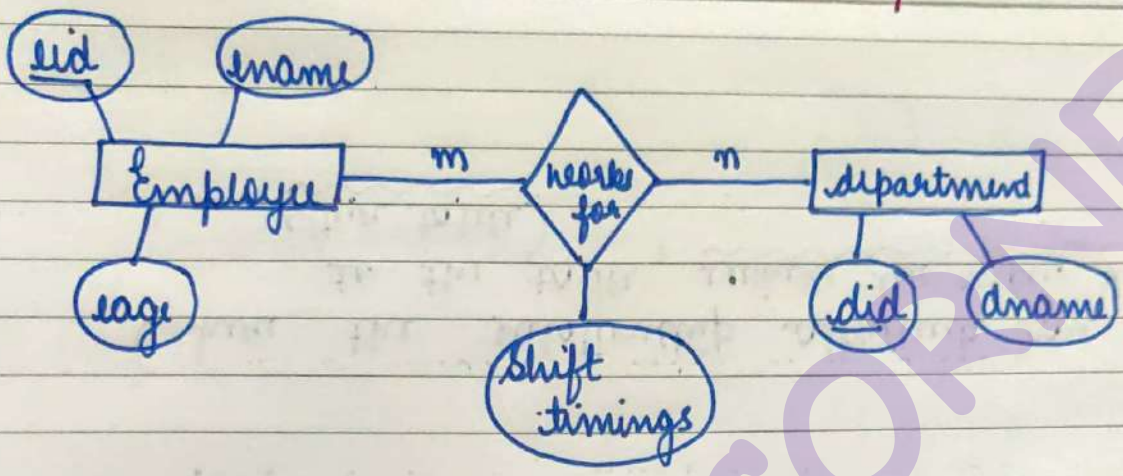
or

sid	sname	sage

sid	did	dname	Shift timing

(here the relationship attribute can be added to the table, where we store the PK of the other table)

Attributes to relationship (m:m)

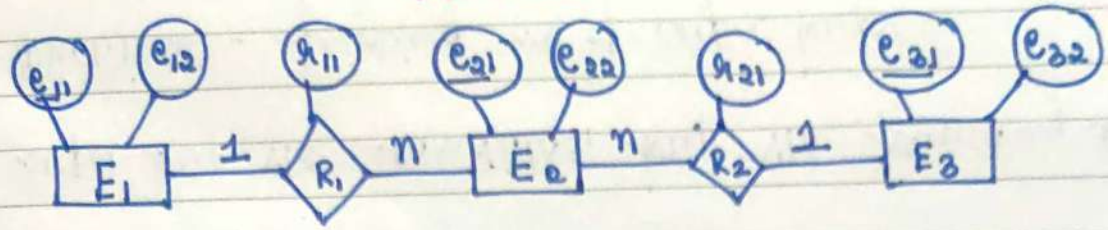


eid	ename	eage

did	dname

eid	did	Shift timing

Problem



E_1

e_{11}	e_{12}

E_2

e_{31}	e_{11}	e_{21}	e_{22}	R_{11}	R_{21}

E_3

e_{31}	e_{32}

minimum no of table = 3.

※ Always move the relationship attribute to the table, where we move the PK.

Unit - 4 :- **Relational Algebra**

Introduction to RA :-

A language (or) set of operations that can be used to manipulate data in the relational model.
↳ (insert, delete, modify)

Employee

Eid	name	dept
1	ABC	35
2	XYZ	36
3	MNO	25
4	ABC	36

1) RA is more a theoritical language.
2) SQL is a practical language.

Select distinct Name → (to avoid duplicates)
From employee;

Π_{name} (employee)
(RA operators remove duplicate values)

Select name from employee;

Name
ABC
XYZ
MNO
ABC

(Sql will not remove duplicates automatically)

Name
ABC
XYZ
MNO

Selection Operation (horizontal partition)

① Used to select a subset of tuple from a relation.

② 'σ' Symbol.

③ We select subset of rows from relation R by applying a condition

Employee

Eid	Eage	Mid	Mage
1000	35	1001	45
1001	45	1002	25
1002	25	1003	65
1003	65	Null	Null

$$[\sigma_{\text{(Selection condition)}} (R)]$$

⇓

(Boolean expression)

displays Empage > 35

$$\sigma_{Eage > 35} (Employee)$$

	Eid	Eage	Mid	Mage
please don't include → 35.	1000	35	1001	45
	1001	45	1002	25
	1002	65	Null	Null

→ 35>35 (False) → 45>35 (true)

1002	65	Null	Null

→ 35>35 (False) → 45>35 (true)

'σ' → no of tuples can decrease depending on condition.

Selection Condition :-

a) can be done attribute name and constant (eage>35)

b) attribute name and attribute name.

Ex:- $\sigma_{(eage > Mage)}$ (Employee)

Eid	Eage	Mid	Mage
1001	45	1002	25

※ No of attributes in the O/P always all the attributes will be displayed.

$$\sigma_{eage>25}\left(\sigma_{mage>25}(\text{Employee})\right)$$

① ②

$\sigma_{eage>25}$ (

Eid	eage	Mid	Mage
1000	35	1001	45
1002	25	1003	65

)

⇓

Eid	eage	Mid	Mage
1000	35	1001	45

* Selection Condition is Commutative

$\sigma_{mage>25}(\sigma_{eage>25}(employee))$

* $\sigma_{eage>25 \text{ AND } mage>25}(employee)$

(∧)

* minimum cardinality of $\sigma_C(R)$ is 0.

(no of tuples)

Ex:- $\sigma_{eage>100}(employee)$;

Eid eage Mid Mage

* maximum cardinality of $\sigma_c(R)$ is "t"

. t → no of tuples in relation 'R'.

Projection Operation

① Used to select subset of the columns from the relation.

② also called Vertical partitioning.

$\Pi_{sid, age}$

sid	age
1	25
2	45
3	35
4	25

$\Pi_{attributes}(R)$

※ Π → always eliminates duplicates.

$\Pi_{name, age}(employee)$

name	age
ABC	25
XYZ	45
MNO	35

Note :-

1) Let 't' be the no of tuples in relation R.

$t \geq$ no of tuples in $\Pi_{attribute}(R)$

(t: no of tuples in the relation)

(no of tuples in the projected relation)

Ex :- Emp

tid	age
1	23
2	25
3	23
4	23
5	23

Π_{tid}

tid
1
2
3
4
5

Π_{age}

age
23
25

1) If the attribute is a superkey.

t = no of tuples in $\Pi_{attribute}(R)$

(no of tuples in original relation) = (no of tuples in the projected relation.)

2) If attribute list is not a superkey,

$t >$ no of tuples in $\Pi_{attribute}(R)$

3) projection operation is not commutative.

$$\Pi_A(\Pi_{A,B}(R)) \neq \Pi_{A,B}(\Pi_A(R))$$

A is a subset of AB.

Rename

① Used to rename a relation (or) attribute of a relation.

② Represented as 'ρ'

* $\rho_{(newname)}(Oldname)$

$\rho_{emp\text{-}data}(Employee)$ → table name will be changed to emp-data.

* to change attribute name.

$\rho_{(id, age, gender)}(Emp\text{-}data)$

→ instead of a, b, c the attributes would be renamed.

* to change the table name and attribute together.

$\rho_{employee\text{-}data(id, age, gender)}(Emp\text{-}data)$

Union Operator

① Union of 2 relations $R_1 \cup R_2$ will contain the tuples which are present in either R_1 or R_2 or both.

R_1

Id	Age	Gender
1	45	M
2	46	F
3	47	M

R_2

Id	Age	Gender
1	45	M
4	46	M
5	48	F

$R_1 \cup R_2$

Id	Age	Gender
1	45	M
2	46	F
3	47	M
4	46	M
5	48	F

(by convension we usually take the attribute name from R_1)

(no duplicate values)

Two relations should be union compatable :-

① no of attributes in both relation (R_1 & R_2) should be same.

② the domain of corresponding attributes should be same.

Note :- corresponding attributes need not have the same name.

Intersection

① Intersection of 2 relations $R_1 \cap R_2$ will contain the tuples which are present in both R_1 & R_2.

R_1

Id	Age	Gender
1	45	M
2	46	F
3	47	M

R_2

Eid	Age	Gender
1	45	M
4	46	M
5	48	F

$R_1 \cap R_2$

Id	Age	Gender
1	45	M

Intersection Compatable (same as union compatability)

Set Difference

Minus between two relations, R1 and R2 $(R_1 - R_2)$ is nothing but the tuples which are present in R1 and not R2.

R_1

Id	Age	gender
1	45	M
2	65	M
3	36	F

R_2

Eid	Eage	gender
1	45	M
2	65	F
4	45	M

$R_1 - R_2$

Id	Age	gender
3	36	F

here also compatability is checked.

$R_1 \cup R_2 = R_2 \cup R_1$ } commutative property.
$R_1 \cap R_2 = R_2 \cap R_1$ }

$R_1 - R_2 \neq R_2 - R_1$

Cartisian Product (Cross product)

① Denoted by X.

R_1

A_1	A_2	A_3
1	ABC	55
2	MNO	53
3	XYZ	21

R_2

A_4	A_5
100	AXY
101	MBH

$R_1 \times R_2$

attributes 8-
$R_1(A_1, A_2, A_3) +$
$R_2(A_4, A_5)$
We add the attributes.

A_1	A_2	A_3	A_4	A_5
1	ABC	55	100	AXY
1	ABC	55	101	MGH
2	MNO	53	100	AXY
2	MNO	53	101	MGH
3	XYX	21	100	AXY
3	XYX	21	101	MGH

} multiply the tables tuples.

② Generally, cross product is not used alone, we mostly use selection (σ) along with cross product.

Cross product ($\times$) + Selection (σ)	= Join

Join Operation

Syntax :- $R_1 \bowtie_{(\text{Join condition})} R_2$

Employee

Employeeid	Age	Gender
1	45	M
2	56	M
3	60	F

dependent

Id	Fname
1	Ram
2	Sam

Employee ⋈ Dependent

employeeid = id (domain should be same)

(attributes are same we do not delete)

Eid	Eage	Egender	Id	Fname	
1	45	M	1	Ram	
1	45	M	2	Sam	X
2	56	M	1	Ram	X
2	56	M	2	Sam	
3	60	F	1	Ram	X
3	60	F	2	Sam	X

a) find father name of all Male employees ?

Important Points of Join

① If relation R_1 has 't_1' tuples and relation R_2 has 't_2' tuples, then no of tuples in :-

a) $R_1 \times R_2 = t_1 \times t_2$

b) $R_1 \bowtie_{(\text{Join cond})} R_2 = 0$ (min tuples) / $(t_1 \times t_2)$ (max tuples)

② If relation R_1 has a_1 attributes and relation R_2 has a_2 attributes, then no of attributes in :-

a) $R_1 \times R_2 = a_1 + a_2$

b) $R_1 \underset{(\text{Join Cond})}{\bowtie} R_2 = a_1 + a_2$

Natural Join

Employee

tid	tage	tgender
1	45	M
2	56	M
3	60	F

Employee father

Id	name
1	Ram
2	Sam

Employee $\underset{tid = Id}{\bowtie}$ Employee's father

(Join we do not delete the attribute since attribute name is different)

Natural Join :- here the join condition will not be mentioned.

→ Condition :- We need to have the name of the attribute we are comparing to have the same name.

Employee

tid	age	gender
1	45	M
2	56	M
3	60	F

Employee-father

tid	fname
1	Ram
2	Sam

Employee ⋈ Employee-father

tid	age	gender	tid	fname
1	45	M	1	Ram
1	45	M	2	Sam
2	56	M	1	Ram
2	56	M	2	Sam
3	60	F	1	Ram
3	60	F	2	Sam

delete

filter out attributes which have the same name.

(delete the common attributes)

Join (Even if we do not have attribute with same name, we can go ahead)

Natural Join :- (two attributes should have the same name)

O/P of natural Join would be ≃ cartesian product

(If attribute with same name does n

O/P of natural Join would be ≃ cartesian product
(If attribute with same name does not exist)

Cartesian product vs Join vs Natural Join

Employee

Eid	Ename	Eage
1	Ram	45
2	Bob	43

Department

Eid	Department
1	Accounts
2	HR

Cartesian product :- Employee × department.

Eid	Ename	Eage	Eid	Department
1	Ram	45	1	Accounts
1	Ram	45	2	hr
2	Bob	43	1	Accounts
2	Bob	43	1	hr

Join : Employee ⋈ Department

Employee.eid = department.Eid (both have the same name, so we use table-name)

Eid	Ename	Eage	Eid	Department
1	Ram	45	1	Accounts
2	Bob	43	2	HR

Natural Join :- Employee ⋈ department

('Eid' is not repeated)

Eid	Ename	Eage	Department
1	Ram	45	Accounts
2	Bob	43	HR

Left Outer Join and Right Outer Join

Employee

Eid	Ename	Eage
1	A	45
2	B	43
3	C	46

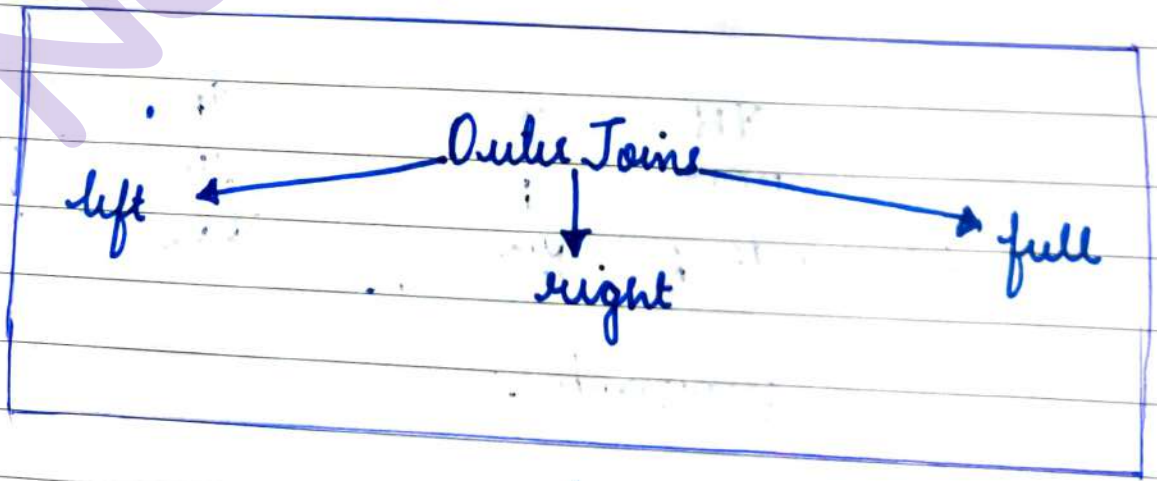
Projects

ProjID	ManagerID
1000	1
1001	2
1002	Null

Employee ⋈ Projects
Eid = ManagerID

Eid	Ename	Eage	ProjID	ManagerID
1	A	45	1000	1
2	B	43	1001	2
3	C	46	Null	Null

(Join → means inner joins)



Steps :- ① Include the symbol (⟗) along with cond^{n}.
② Perform cartesian product
③ make the selection based on condition.
④ Include remaining tuples of left most table. the matching right table will have null values.

• Right Outer Join

Employee ⟖ Project
tid = managid

tid	tname	tage	ProjectID	managid
1	A	45	1000	1
2	B	43	1001	2
Null	Null	Null	1002	Null

Full Outer Join

Employee ⟗ Project
tid = managid

tid	tname	tage	Project id	managid
1	A	45	1000	1
2	B	43	1001	2
3	C	46	Null	Null
Null	Null	Null	1002	Null

Inner Join V/S Left Outer Join V/S Right Outer Join V/S Full Outer Join

R_1

A_{11}	A_{12}
1	3
2	4
3	5

R_2

A_{21}	A_{22}
1	5
2	6
4	8

Inner Join.

$R_1 \bowtie R_2$

$A_{11} = A_{21}$

A_{11}	A_{12}	A_{21}	A_{22}
1	3	1	5
2	4	2	6

Left Outer Join

$R_1 \ \text{⟕} \ R_2$

$A_{11} = A_{21}$

A_{11}	A_{12}	A_{21}	A_{22}
1	3	1	5
2	4	2	6
3	5	Null	Null

Right Outer Join

A_{11}	A_{12}	A_{21}	A_{22}
1	3	1	5
2	4	2	6
Null	Null	4	8

Full Outer Join

A_{11}	A_{12}	A_{21}	A_{22}
1	3	1	5
2	4	2	6
3	5	Null	Null
Null	Null	4	8

Minimum and Maximum Number of tuples possible

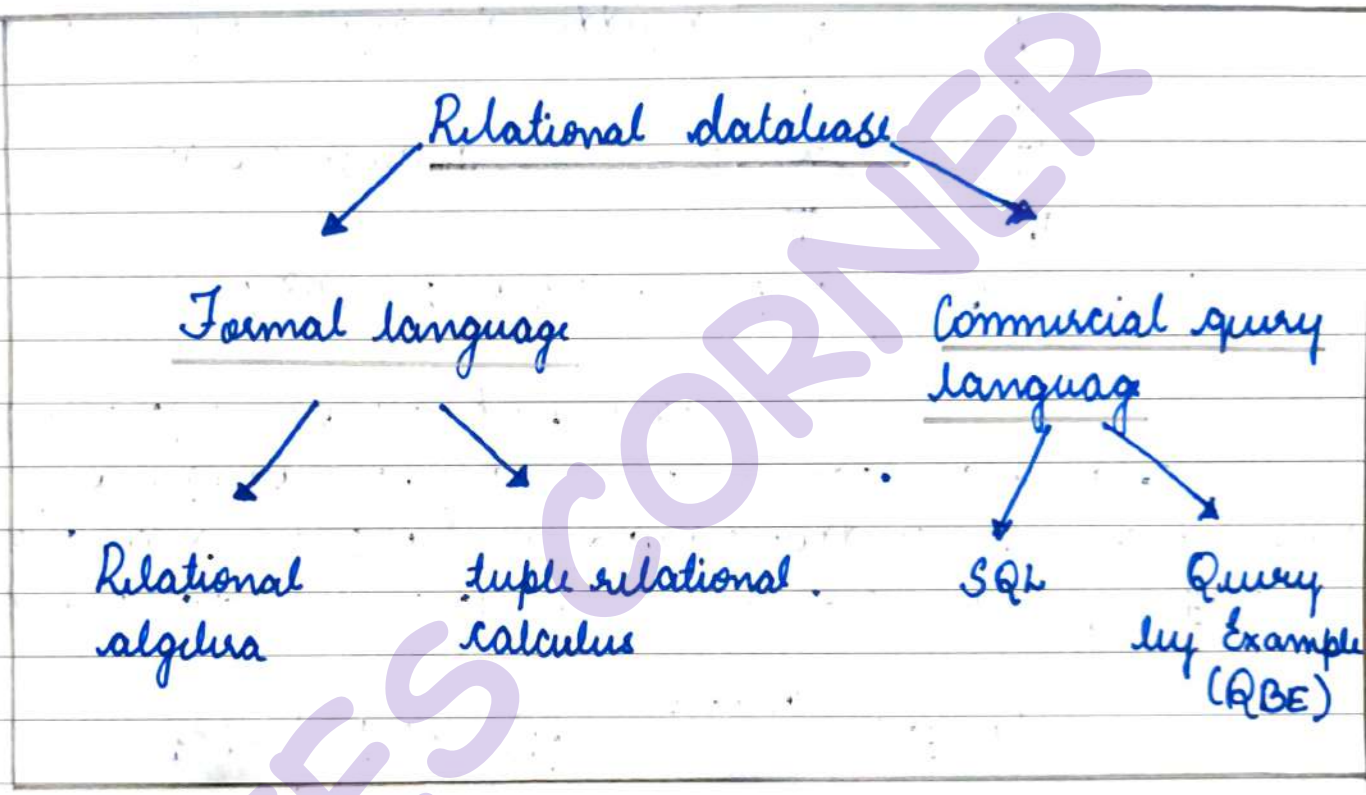
$R_1 \rightarrow$ 'm' tuples $\quad R_2 \rightarrow$ n tuples

	Minimum no of tuples	Maximum no of tuples
$\times$	$n \times m$	$n \times m$
$\bowtie_{(\text{Join cond})}$	0	$n \times m$
$\bowtie$	0	$n \times m$
$\overset{\text{Join cond}}{\bowtie}$	$R_1 \bowtie R_2$ m	$n \times m$
$\overset{\text{Join cond}}{\bowtie}$	$R_1 \bowtie R_2$ n	$n \times m$
$\overset{\text{Join cond}}{\bowtie}$	$R_1 = 3$ tuples, $R_2 = 5$ tuples. $\max(m, n)$	$n \times m$
$\cup$	$R_1 = 3$ tuples $\quad R_2 = 6$ tuples suppose 3 is common, add remaining 2 tuples from R_2. $\max(R_1, R_2) = \max(m, n)$	$m + n$
$\cap$	0	$\min(m, n)$
$-$	$(R_1 - R_2) \Rightarrow$ 0 $\quad (R_1 - R_2) \leq R_1$ $(R_2 - R_1) \Rightarrow$ 0	$(R_1 - R_2) \Rightarrow m$ tuples $(R_2 - R_1) \Rightarrow n$ tuples

Problems on Relational Algebra.

Set Operations :- Union, Intersection, difference, cartesian product.

Relational Operations :- Select, project, join, division.



Division :- ($\div$)

R

A	B
a_1	b_1
a_1	b_2
a_1	b_3
a_2	b_2
a_2	b_3
a_3	b_1
a_3	b_2
a_3	b_3
a_4	b_2

S

B
b_1
b_2
b_3

A
a_1
a_3

satisfy all properties (b_1, b_2, b_3)

$T = R \div S$.

$(R-S)$ = 'A' column.

$(AB)-B = A$

Additional Relational Operations

Aggregate functions :- Sum, Average, Max, Min, count.

Grouping function :- $\mathcal{F}$ (script F)

the tuples of a relation are first grouped by the value of some attributes and then aggregate functions are applied on individual groups.

Eg :- a) $Dno\ \mathcal{F}_{count(SSN)}\ (Emp)$

the above expression first groups the tuples in Emp table based on Dno, and then applies count function on individual groups this will output no of employees in each department.

Result Relation → (10 departments)

Dno	count (SSN)
1	10
2	10
3	5
4	6

[Aggregate function will be applied after grouping]

b) $Dno, Age\ \mathcal{F}_{count(SSN)}\ (Emp)$

(more than 10 groups)

Dno	Age	Count (SSN)
1	34	5
1	36	6
2	35	7
3	41	10
3	45	4

Multiple Tables :- Joins and Set Operations

1. Design queries based on multiple tables.
2. Retrieve related data from various tables in a database.
3. Various types of table joins.
4. Set operations :- Union, Union all, Intersect and Minus.

Join

1. When the required data are in more than one table, related tables are joined using a join condition.

2. The join condition combines a row in one table with a row in another table based on the same values in the common columns.

3. In most cases (but not always), the common columns are the primary key in one table and a foreign key in another.

Cartesian product

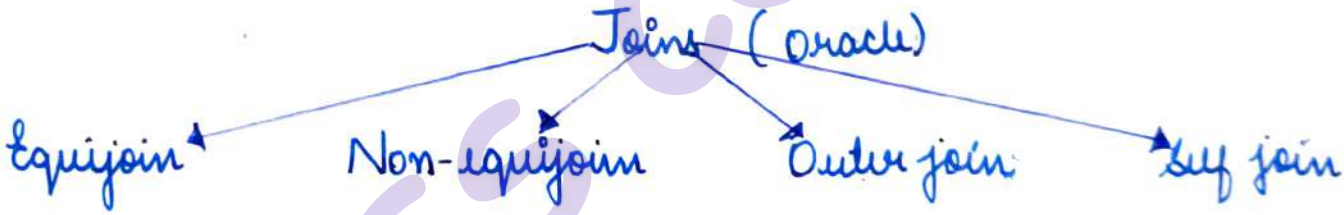
A cartesian product results from a multitable query that does not have a where clause. The product operation joins each row in the first table with each row in the second table. The product normally results in an

output with a large number of rows and is not very useful.

table 1	table 2	total no of rows
m	n	m x n rows.

Student (6 rows) x Faculty (8 rows) = 48 rows.

Remember :- that the number of joins conditions is one less than the number of table names used in the From clause.



Equijoin :- is a join with a join condition involving common columns from two tables.

Ex :- Select Student. last || ', ' || Student. first

Student, faculty. Name Faculty,
faculty. phone Telephone from
Student, faculty
where
Student. facultyid = faculty. facultyid
(same attribute name)