7-1-2020

COURSE LEARNING OBJECTIVES

1. To discuss and ~~reality~~ realise the importance of database architecture, design notation, ER-modelling mapping and schema design.
2. To gain the knowledge of relational algebra and learn the use SQL and PL/SQL
3. To introduce formal database design approach through normalisation and discuss various normal forms.
4. To understand the importance of concurrent transactions and discuss issues and transaction control algorithms

UNIT - 1

Types of database's

1 traditional - if db store numeric and textual data
2 multimedia - audio, video etc
3 GIS - weather data, maps, satellite images
4 Real-Time/Active - manufacturing processes
5 Datawave house and OLAP systems.
6 No SQL db's
7 Graph db's
8. Object Oriented ~~bb~~ db
9. Object - Relational db

Defn : Collection of related data

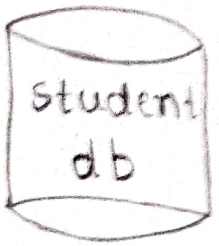
data - known facts < recorded / have implicit meaning

9/1/2020

Properties of database :

A database can be defined as a logical collection of data with an implicit meaning

1. A db represent some aspects of the real world called the miniworld / UOD (universe of discourse). changes to the miniworld are reflected in the database.



IVC - VC
accurate / consistent

↑ source of data (aspects of real world)
(entity)

2. A database is a logically coherent collection of data with some inherent meaning. A random collection of data cannot correctly be referred to as a database.

3. A database is designed, built and populated with data for a specific purpose. It has an intended group of users and application in which these users are interested

4. A database can be of any size and complexity

5. A database may be generated and maintained manually / it may be computerised.

DBMS

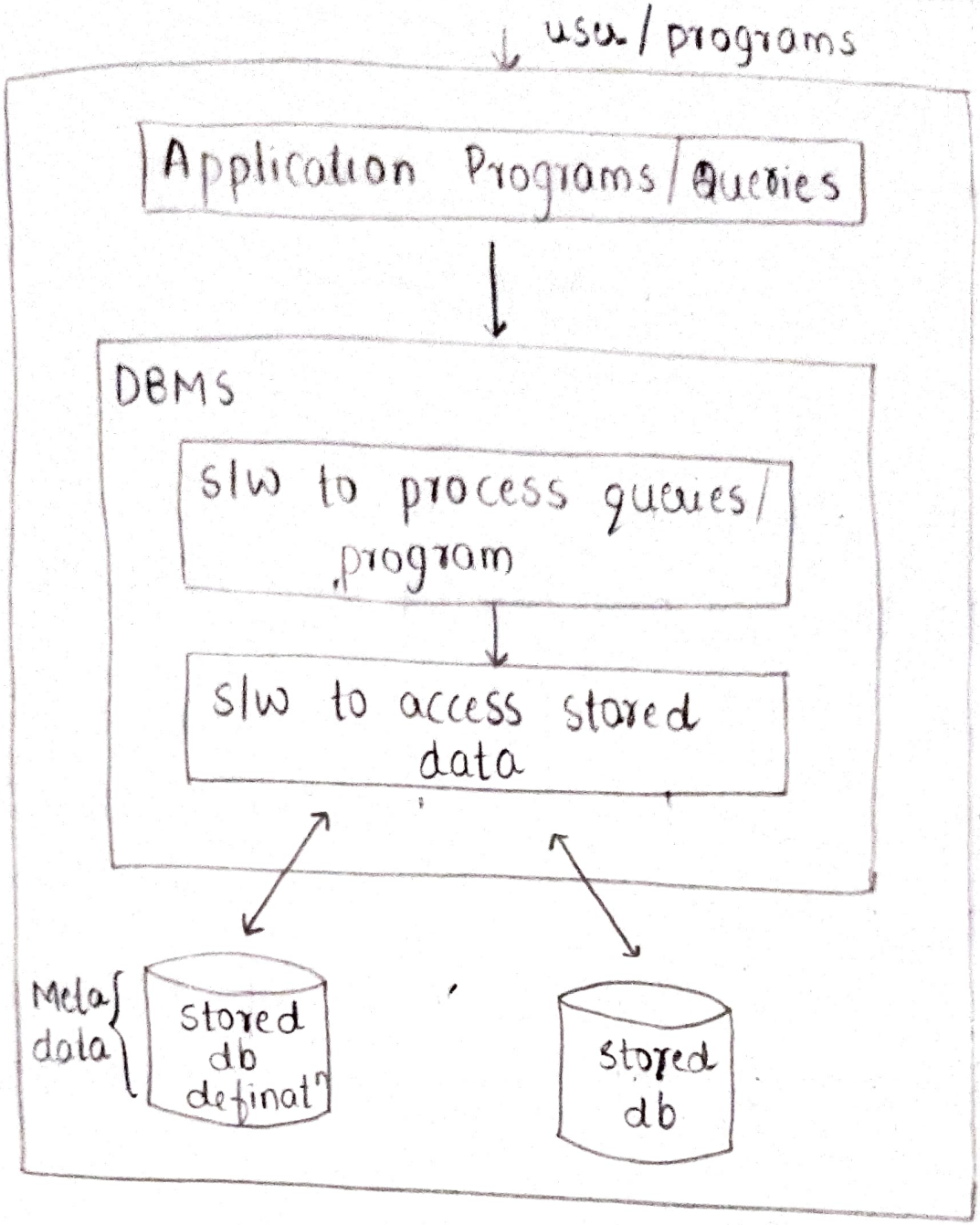
It is a collection of programs that enables users to create and maintain a database.

Functions :

1. Defining : includes specifying the data types, structures, constraints of the data to be stored in the database
2. Construction : involves creating a database and : populating the database and storing on to a physical storage
3. Manipulating : Querying the database (to retrieve the data), updating the existing the data, deleting .
4. Sharing : allowing access to multiple users and application programs simultaneously.
5. Protection : Two forms - i. System protection
   ii. Security protection

i. System : protecting database against h/w and s/w crashes

ii. Security : preventing unauthorized access to database



Student - Course database :

Entities

STUDENT - USN | Name | DOB | Section | Major

COURSE - CID | CName | Credits | Type of Course | Branch

SECTION - SID | ROOMNO | STRENGTH

FACULTY - FID | FName | Branch | Major | Branch

GRADE REPORT - USN | Subject name | score

PRE-REQUISITES - CID | name of preq

10/1/2020

Approaches to info system design

1 Traditional approach
2 Database approach

Main characteristics of the database approach v/s the file processing approach are

1 Self-describing nature of a database system
2 Insulation between programs and data, data abstraction
3. Support of multiple views of the data
4. Sharing of data and multi user transaction proces

13/1/2020.

1 Self-describing nature of a database system.

Database — Db itself
\ Db catalog : data defn / description of the db (metadata)

Any changes made in the structure of database need not to change the application program.

File system - 'File' . } applicatn pgm.

structure of database is in application pgm.

The description of a database is stored in a catalog and is called meta data, which describes the structure of the primary db

Relations

Relation name	No of Columns
Student	5
Course	5
Section	3
Faculty	5
Grade Report	3
Prereq	2

Columns:

Column name	datatype	belongs-to
Ssn	character(10)	Student
Name	Character (30)	Student
Bdate	Character(12)	Student
:		

Translation bet' pgms and data, data abstraction

- program data independence
- program data operation independence

The structure of the data files is stored in the library catalog separately from the access programs.

- changes to the structure of a file will not require changing programs that accesses that file.

Class
- data members - "state"
- member function - "behaviour"

member function
- signature (return type, name, parameters)
- body (implementation/body)

- Signature will act as interface to class. by which application program access.
- User application programs can operate on the data by invoking these operations through their names and argument (signature of a method). regardless of how the operations are implemented. This is called program-operation independence.

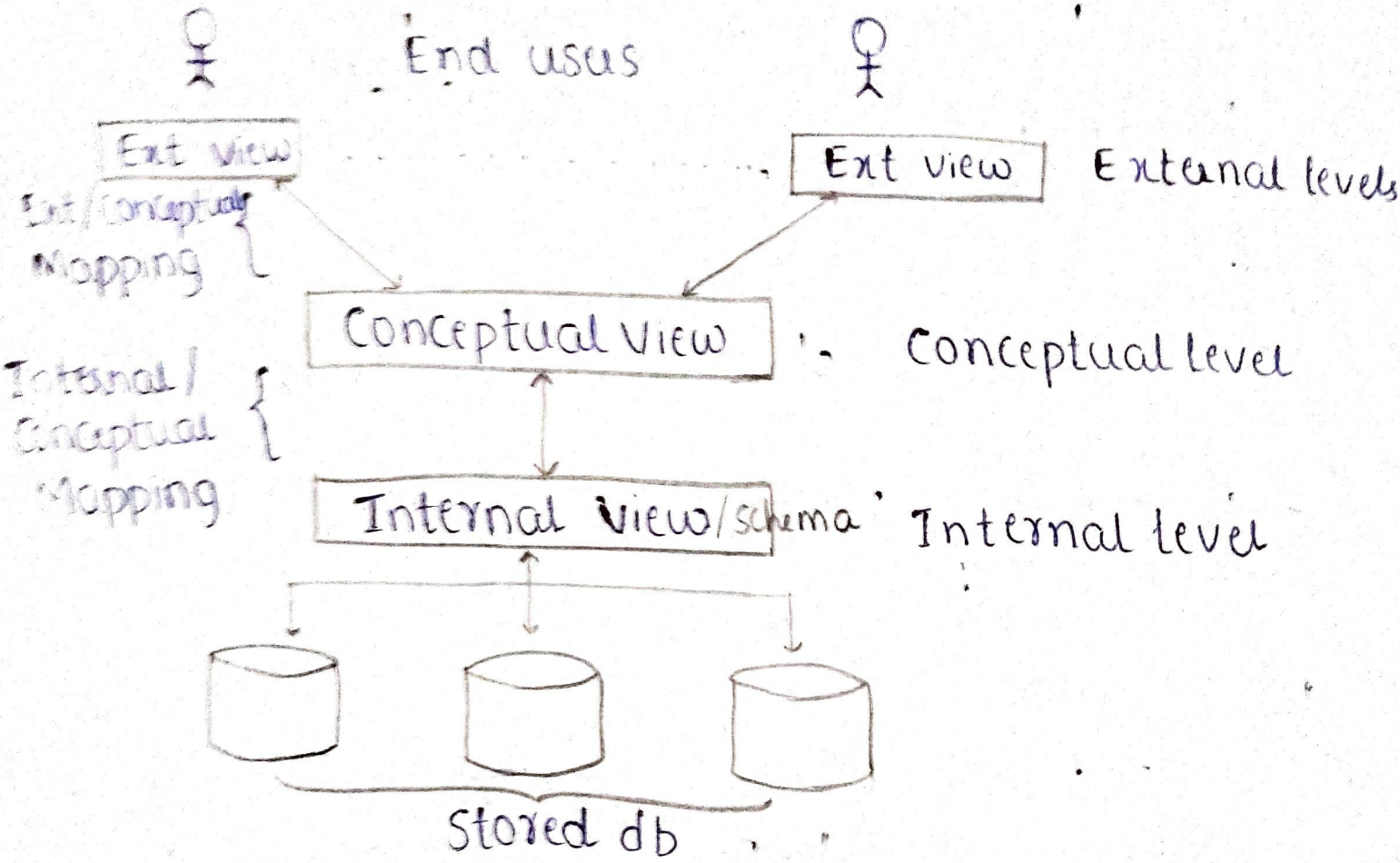
3 Support of multiple view of data

data

Stored	derived
physically stored on to a storage medium "table"	- not stored (generated on the fly) - derived from stored data Eg: age from DOB "views"

4. Sharing of data and multi user transaction processing
- Concurrency control programs.

14/1/2020



Three Schema Architecture

Transaction properties -: ACID

Data Abstraction : Hiding unnecessary info about the system and making visible only what is required

Data Model :

- Helps in achieving data abstraction
- Data model are a collection of concepts that help in describing a database
- used to describe a database interms of relations, attributes, datatypes of attributes, constraints and relationships.

* Schema

Description of the database is called as a schema.

- diagramatic represent^{n} of schema is called schema diagram

1. Internal level
   - defines internal schema
   - describes ⊢ physical storage
     ⊢ access paths.
   - physical data model is used.

2. Conceptual level
   - defines conceptual schema
   - describes db interms of entities, attributes relationships and constraints.

- representational data model is used to define conceptual schema (ER-diagram)

3 External Level

- defines external schema (views)
- representational data model is used

· Each external schema describes the part of the database, that the particular user group is interested in and hides the rest of the database from that user.

Mappings:

The processes of transforming requests and results between levels are called mappings

Data-independence:

Ability to change the schema at one level of a db system without having to change the schema at the next higher level.

1 Logical data independence

Internal schema ⟶ Conceptual schema
Changes does not changes

- means either physical storage or access path changes

2. Physical Data independence

Conceptual schema ⟶ External schema
Changes — does not change
- means entities, constraints etc. changes

* Schemas are designed even before database created / build

ER diagram :
- Entity Relationship diagram
- describe the conceptual schema of db

1. Entity : 
- basic components of ER diagram.
- represent a real world object
- objects have independent existence

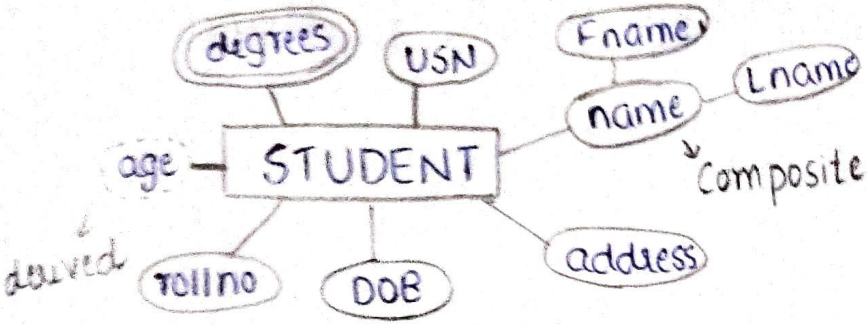
physical / conceptual

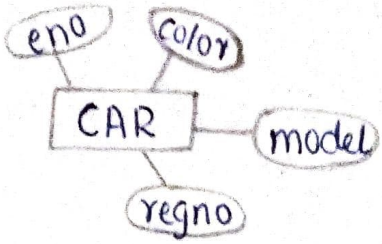
Eg: STUDENT, CAR, BOOK — Physical Existence

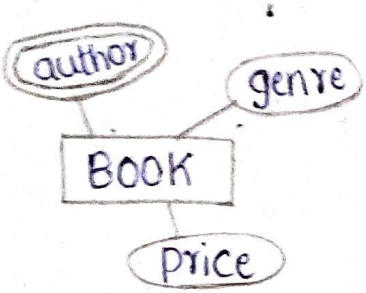
BANK - conceptual existence

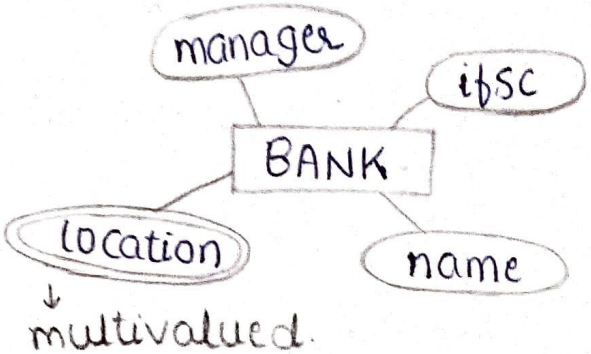
2 Attribute(s) : ⬭

- represents properties of an entity









Types :

1. Simple v/s composite
2. Single v/s multivalued
3. Stored v/s derived

1 Simple : are indivisible / cannot split into subpart

Composite : can be divided into smaller sub parts, which represent more basic attributes with independent meaning

2. Single attributes are the one that have a single value for a particular entity. and in some cases, attribute can have a set of values for the same entity such attributes are called multivalued.

3. Stored v/s multivalued:

Attributes can be assigned NULL values when the value for a particular attribute does not exist

- When value is assigned as null one way it is interpreted as NA.
- Address

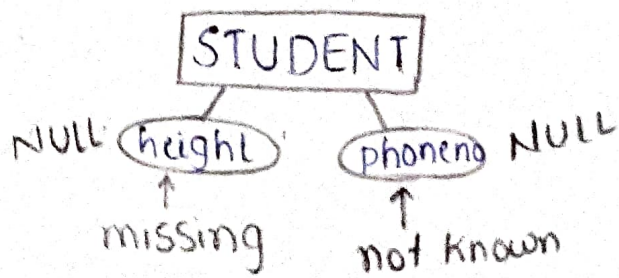
Street	Apartment No	City	ZipCode
10	25	B'glore	16
20	NULL	Mumbai	20

- The second possible interpretation of NULL is unknown

missing — not known

↑ it exists but not recorded

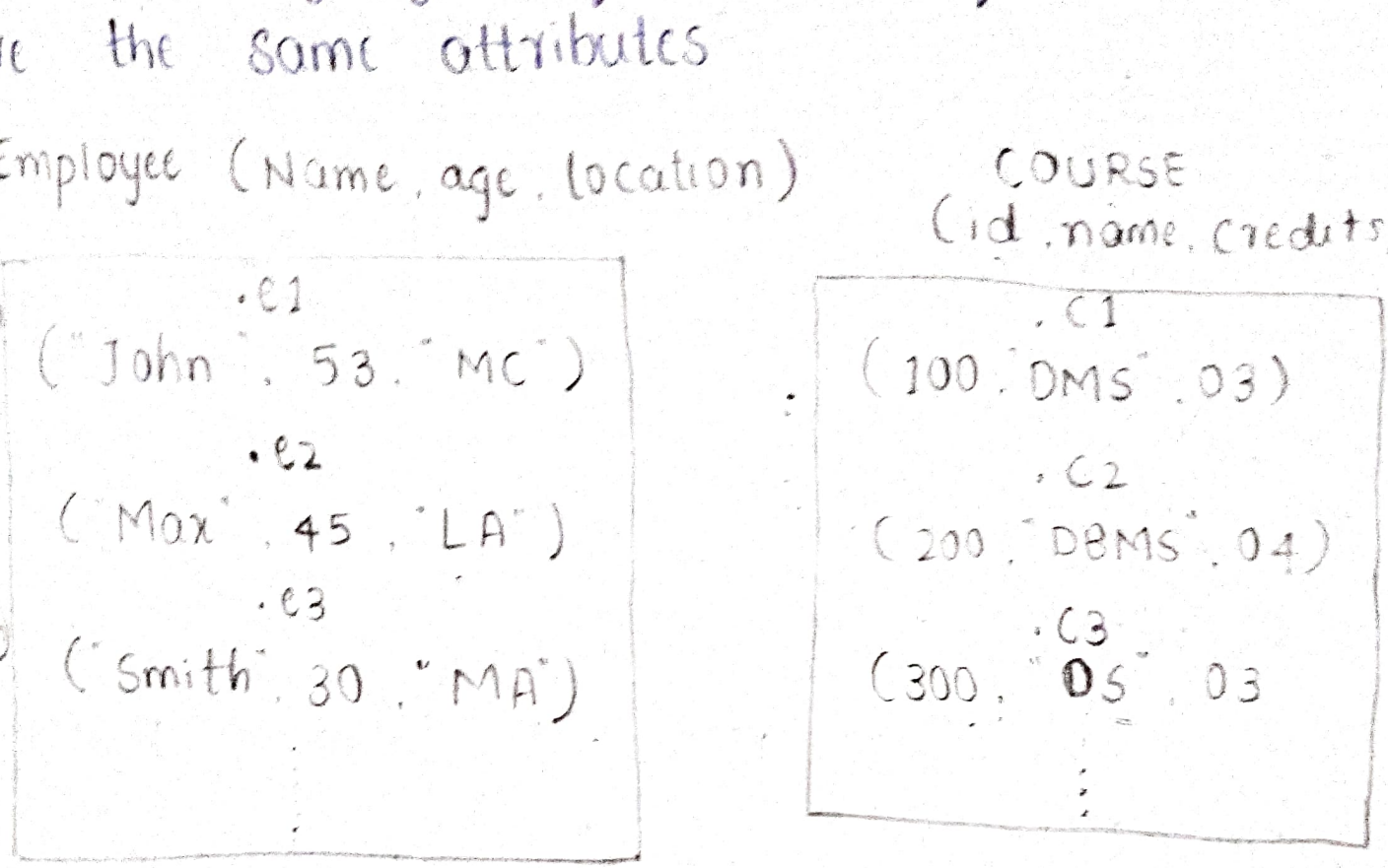
don't know whether it exist



17/1/2020

Entity Types

An entity type defines a set of entity that have the same attributes



Entity Set

An entity set is collection of all entities of a particular entity type in the database at any point in time

* Keys : used to enforce uniqueness constraints on the values of entity in entity set

A key is an attribute or a combination of attributes whose values are used to uniquely identify each individual entity in entity set.

- Key attributes are underlined inside the oval in ER diagram (usn)

- The entities that do not have a key is weak entity

Key — Simple Key : single value
Key — Composite Key : more than 1 is made as key

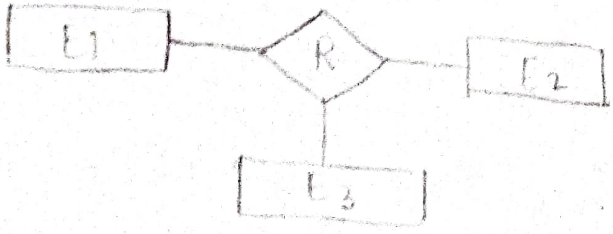
Strong entity : 
Weak entity : 

Relationship :

It is an association between 1 or more entities 

- Recursive - 1 entity
- Binary - 2 entities are associated / participating
- Ternary - 3 " " " " "

- Binary



- Ternary

- Recursive

The degree of a relationship is the no of participating entities

Weak Relation - If one of the participating entity is weak entity it forms weak relationship.

 - weak relationship

Draw an ER-diagram for a movie database

Movie - name, director, producer

Cast - lead actors, supporting actor.

Music - composer, singer, lyricist

Technical team -

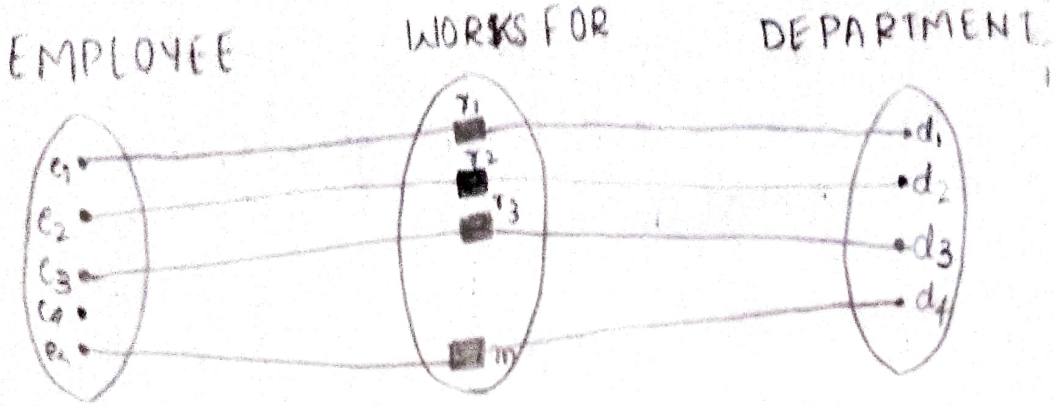
18/1/2020

* Relationship Type :

A relationship type 'R' among 'n' entity types $E_1, E_2 \ldots E_n$ defines a set of associations among entities from these entity type.

- A relationship set is the set of associations among entities from entity type
- The relationship set R is a set of instances r_i where each r_i associates n individual entities

$(e_1, e_2 \ldots e_n)$. and



Degree of works for = 2

* Constraints on relationships.

Relationship types usually have certain constraints that limit the possible combination of entities that may participate in the corresponding relationship sets

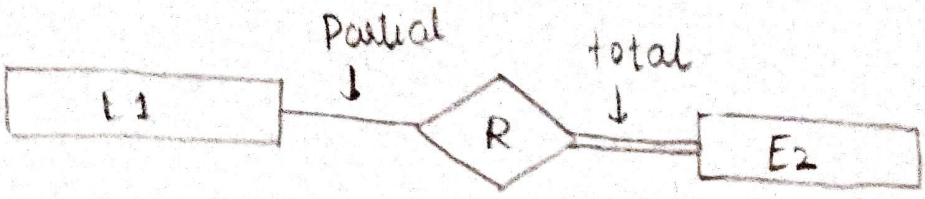
i. Cardinality constraints / ratios :

- Specifies the max. no. of relationship instances that an entity can participate

ii Participation

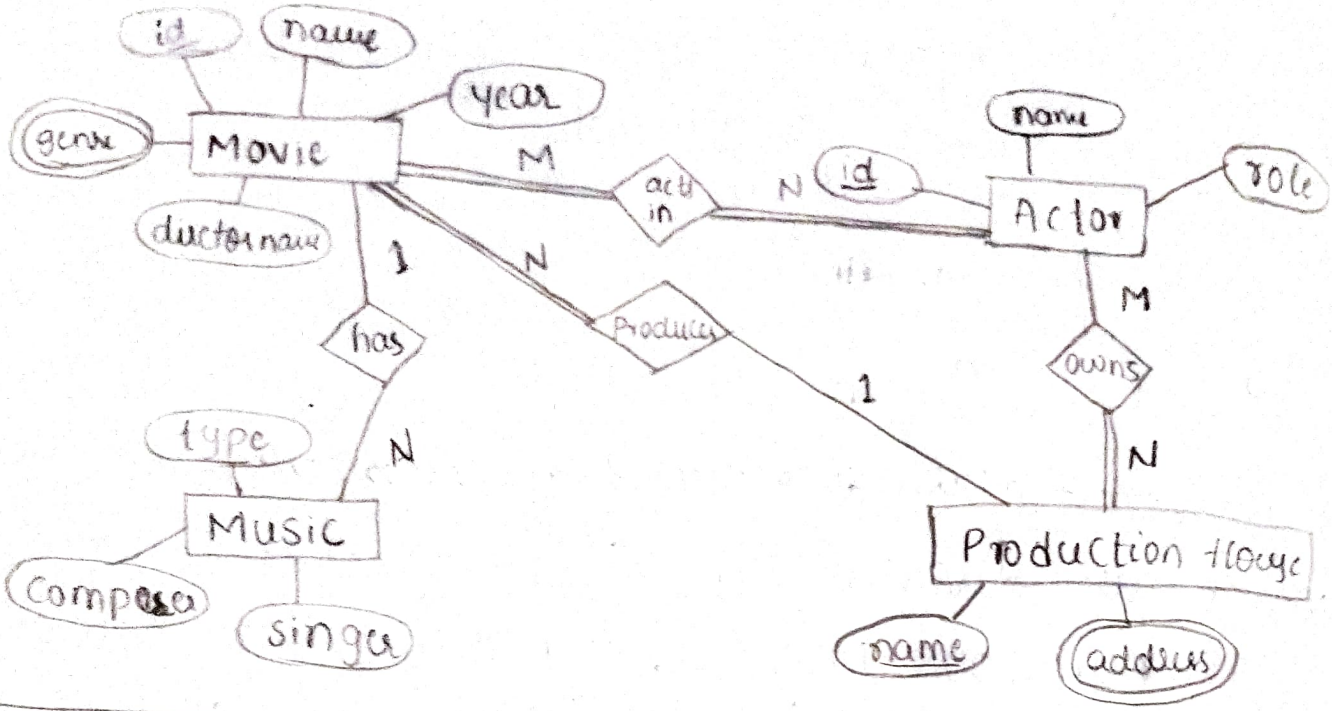
Specifies whether the existence of an entity depends on its been related to another entity via relationship type.

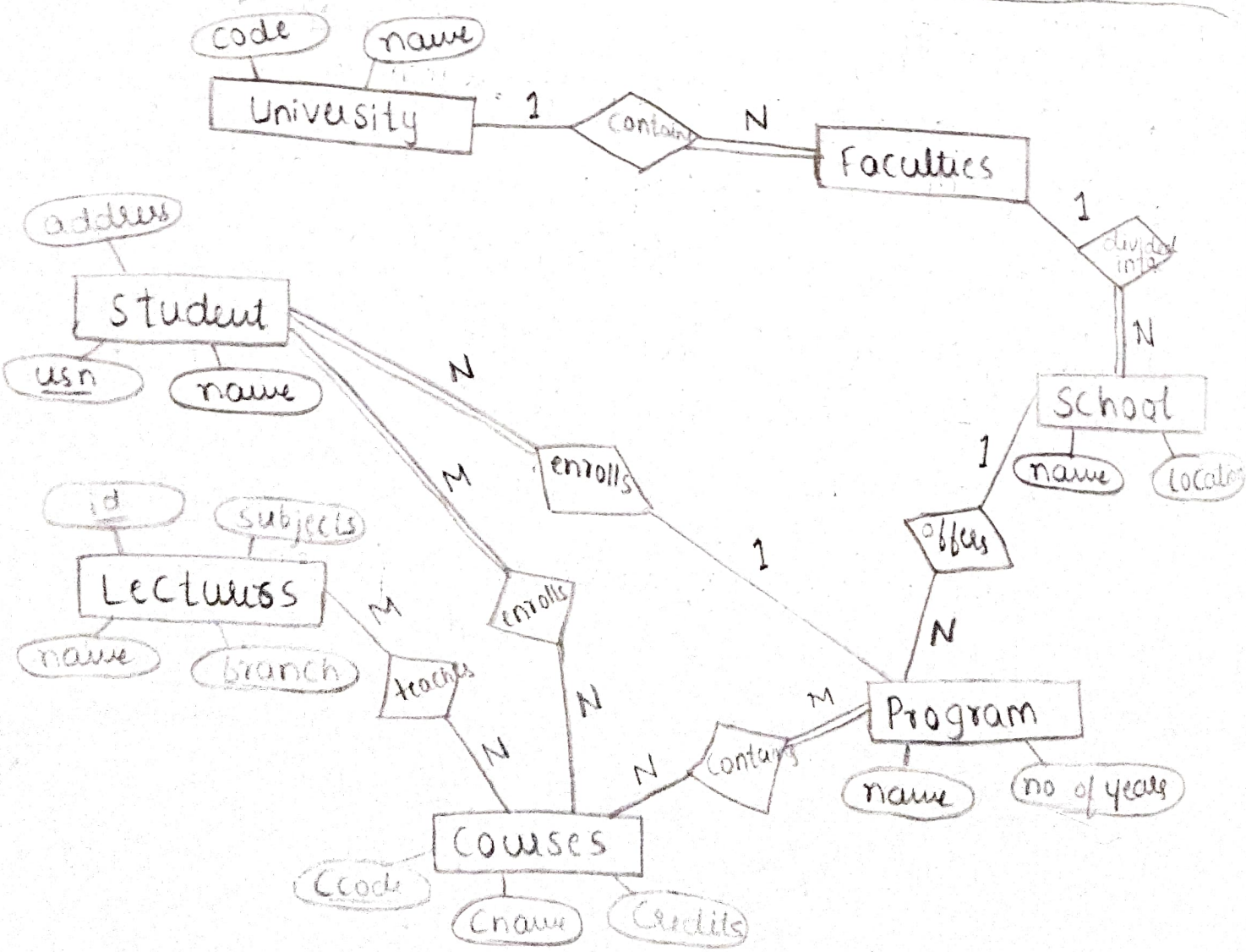
- There are 2 types of participation constraints namely i. Total ii. partial.



A university contains many faculties. The faculties intun are divided into several school. Each school offers numerous program, and each program contain many courses. Lectures can teach many diff. courses and even the same course several time. Courses can also be taught by many lectures. A student is enrolled in only one program but a program can contain many students. Students can be enrolled in many courses at the same the courses ~~had~~ have many students enrolled. Draw the ER-diagram for the above scenario.

20/1/2020


id
name
year
genre
Movie
M
acted in
N
id
name
Actor
role
director name
1
N
has
Produces
M
owns
1
type
N
N
Music
Production House
composer
singer
name
address


code
name
University
1
Contains
N
Faculties
address
1
divided into
Student
usn
name
N
N
School
enrolls
M
1
name
location
id
subjects
1
offers
Lecturers
M
enrolls
N
name
branch
teaches
N
N
M
Program
N
N
Contains
name
no of years
Courses
Ccode
Cname
Credits

Draw an ER diagram for an online bookstore.

Book : name, id, price, genre

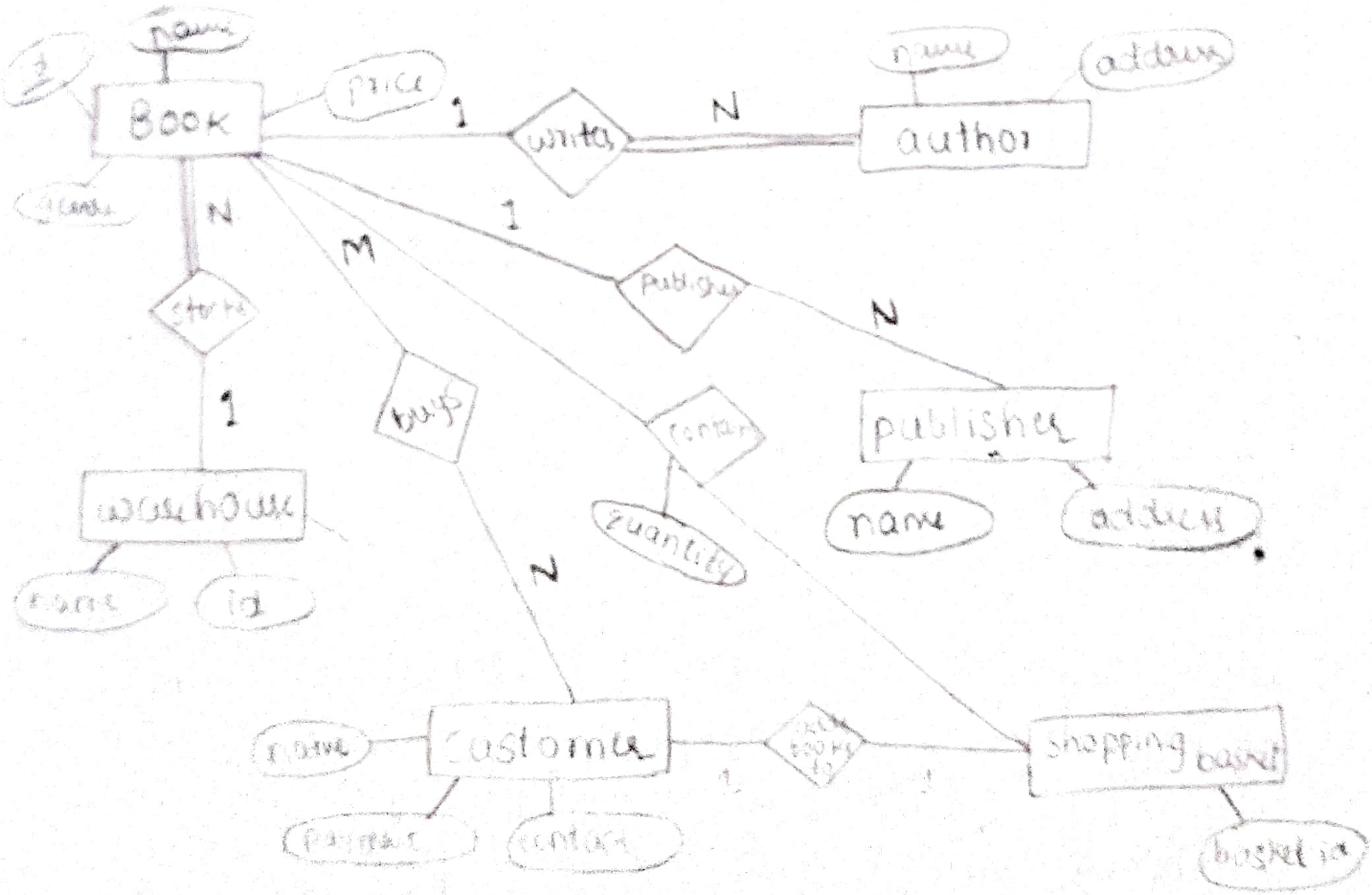
author : name, address

publisher : name, address

Customer : name, payment details, contact details

Shopping bskt : no of books, total price

warehouse



~~A large organis~~

21/1/2020

* Attributes of Relationship Types.

- Attributes describing properties can also be associated with relationship types.

Consider the following info about a university database.

1. Professors have an SSN, name, age, branch and a research speciality
2. Projects have a project no. a sponsor name, a starting date and ending date and a budget
3. Graduate students have USN, name, age and degree program
4. Each project is managed by 1 professor known as the projects principal investigator
5. Each project is worked on by 1 or more professors known as the project's co-investigator
6. Professors can manage and/or work on multiple projects

7 Each project is worked on by 1 or more graduate students known as the project's reasearch asst.s.

8 when graduate students work on ~~you~~ a project, a professor must supervise their work on "

9 Graduate students can work on multiple ~~stu~~ projects in which case they vil have a potentially diff. supervisor for each one

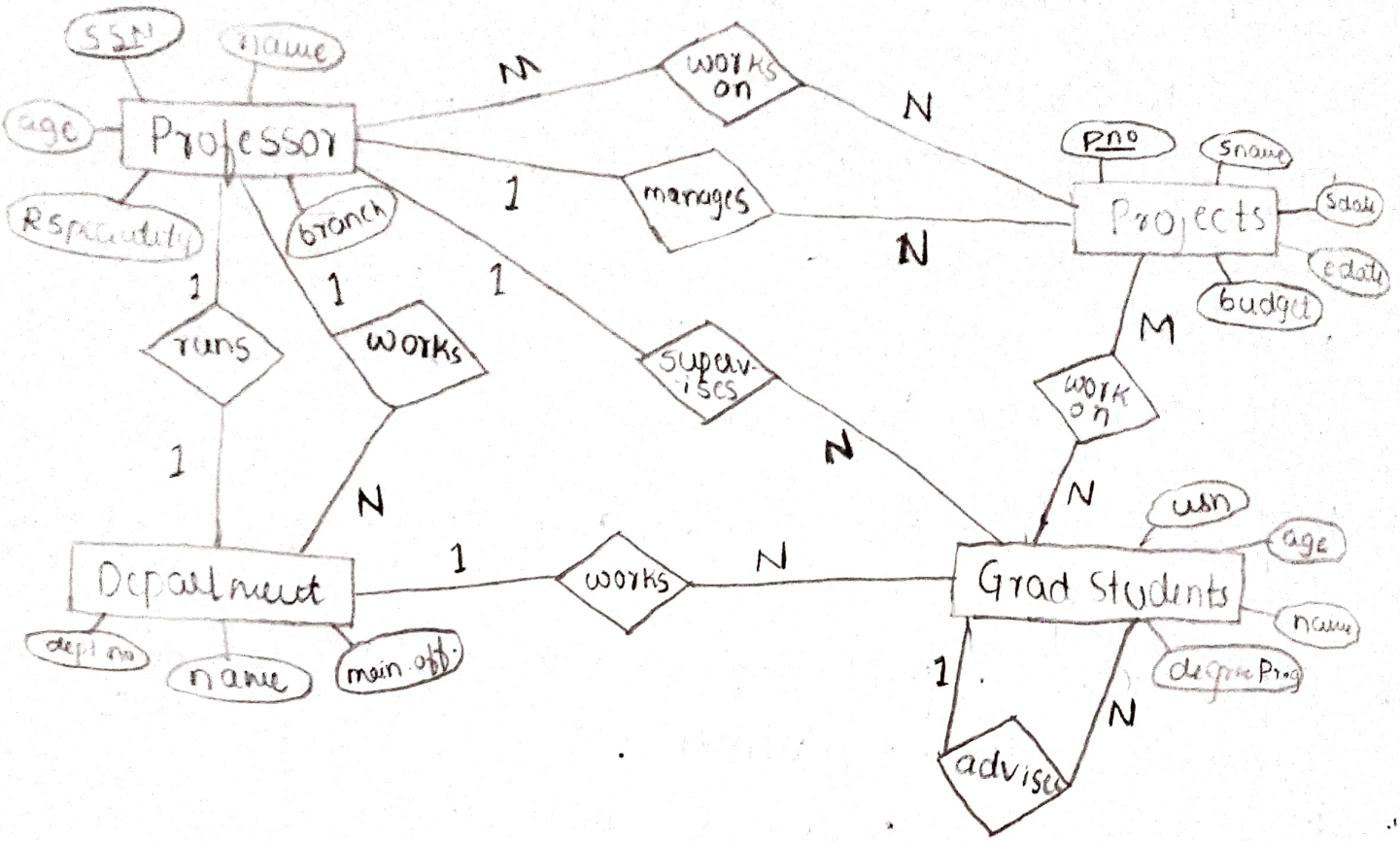
10. Dept. has dept no. name and a main office

11 10. Depts have a professor known as a chairman who runs the department.

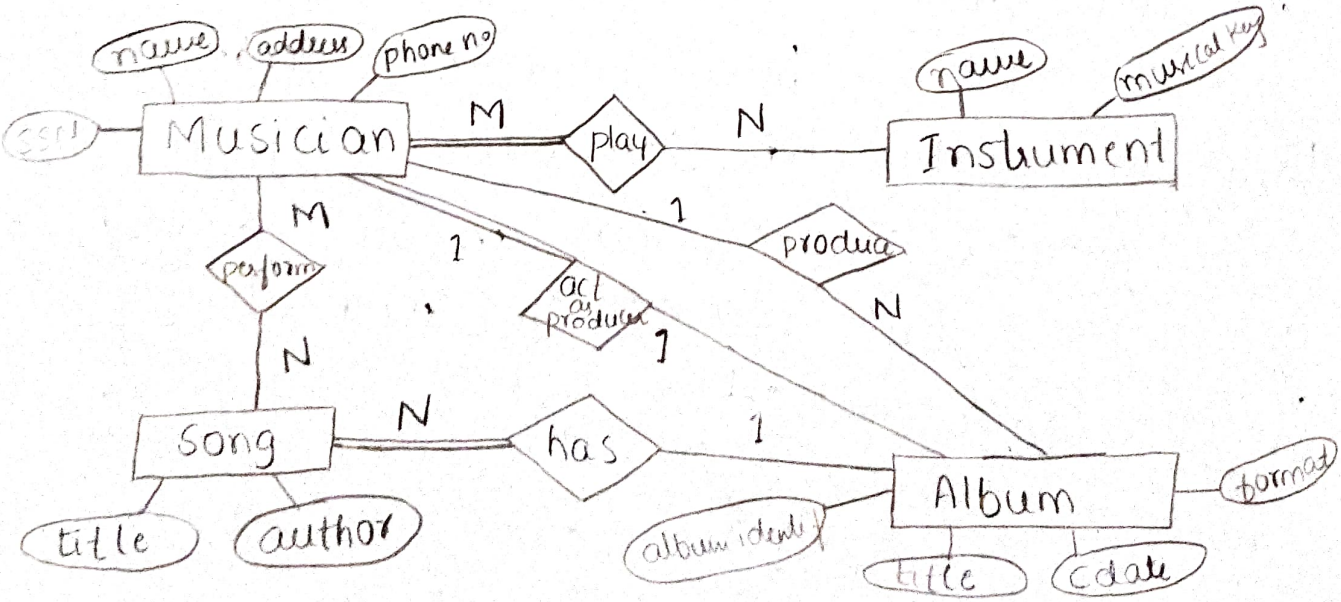
12. Professors work in one or more dept. and for each dept. that they work in a time % is associated with their job.

13. Graduate students have 1 major dept. in which they are working on their degree

14. Each graduate student has an other more graduate student (stu. advisor) who advises him/her on what course to take.



- University info database
- Music store



24-1-2020

UNIT-4

SQL

- It has come to become the standard, for use with Relational-DBMS

- Advantages:

1. Migration from RDBMS to other is easy.
2. Db Application pgms are written in SQL. if the db is changed from one RDBMS to other there is not need to change applicat^{n} pgm, iff the both RDBMS follow same standard language. SQL is the language of db's

History:

SQL is currently expanded as structured Query Language. Before/Originally it was called as SEQUEL - Structured English Query Language

- It was introduced by IBM research

- ANSI + ISO → SQL1/SQL86
  ↓
  SQL99 ← SQL2/SQL92

* Type of Language

- It is a comprehensive DB language.
- It is both DDL and DML
- DDL : allows for definitions of tables, views, domains etc.

  DML : allows to manipulate and also update
- It allows for specifying ~~and~~ security and authorization
- For ~~allows~~ defining integrety constraints.
- Specifying transaction controls
- It also has rules for embedding SQL statem to into a general purpose programming langu C, C++, java.

* Datatypes in SQL

1 NUMERIC :

- integer - INT, INTEGER, SHORTINT
- real no.s - FLOAT / REAL

formatted - DECIMAL (i, j) (j → scale, i → precision)
DEC (i, j)

2 Char - Strings :

- fixed length → CHAR(n) n - no. of chars.
  CHARACTER(n)
- varying length → VARCHAR(n)
  VARYING CHARACTER(n)

3. Bit - Strings :

- fixed length → BIT(n)
- varying length → BITVARYING(n)

} preceded with 'B'

4. Boolean : three logics

1. True 2. False 3. Unknown

In SQL a boolean data type is considered a 3 valued logic.

'06 - Feb - 2020'

5. Date and Time

Date : 10 digits YYYY : MM : DD

Time : 6 digits HH : MM : SS

6. Time Stamp : includes time and date fields plus a min. of 6 positions for decimal fraction of second and an optional time zone qualified

25-1-2020

Create Table Statement

- DDL statement (data defⁿ)
- Create used to create
  i. a new relation (table)

Relational Model		SQL
Relation	-	Table
Attributes	-	Column
Row	-	Tuple

  - the name of relation / table
  - name of the attributes + datatypes
  - initial constraints — column level
    \ tabel level

  ii. a view
  iii. a domain
  iv. triggers
  v. asseations etc

syntax of create statement

```
CREATE TABLE tablename (col^n name1, datatype, constraint
                        :
                        :
                        col^n name n,
                        CONSTRAINT 1,
                        CONSTRAINT 2);
```

1 Constraints :

1. Primary key constraint : used to uniquely identify each record in the table

Keyword for such a key is

PRIMARY KEY — NOT NULL constraint
UNIQUE constraint

Simple
- Single attribute
- can be specified at column level/ table level constraints

Composite
- more than 1 attribute
- table level constraints only.

To view the table created :

describe / desc tablename ;

* Insert Stmt

used to add new row or tuple to existing tables

Optional

Syntax: INSERT INTO Tablename (col1, col2, ... coln) VALUES (val1, value2, ... valuen);

* SELECT Stmt

Select Stmt is used to retrieve the contents of the table

31/1/2020:

2 Consider the following tables

WORKS (Pname , Cname , Salary)

LIVES (Pname , Street , City)

LOCATED_IN (Cname , City)

MANAGER (Pname , Mgrname)

where Pname = Person

Cname = Company

Mgrname = Manager

1. Draw a relational schema diagram for the above table
2. Write create table statements for each relation in the given schema

WORKS

Pname	Cname	Salary

LIVES

Pname	Street	City

LOCATED_IN

Cname	City

MANAGER

* DELETE Stmt :

- The delete command removes tuples from a relation.
- It includes a where clause similar to that used in an SQL query, to select the tuples to be deleted
- Tuples are explicitly deleted from 1 table at a time.

Syntax : DELETE FROM tablename WHERE <condition>; (↓ optional)

Eg: Delete from emp ; deletes all records

Delete from emp where age = 45 ;

Delete from emp where Ename = 'Max'
OR / AND age = 50 ;

* DROP Stmt :

Syntax : DROP table tablename ;

it deletes both tuples + description also.

- UPDATE Stmt

```
UPDATE tablename
    SET col-name = newvalue
    WHERE <condition> ;
```

- Update command is used to modify attribute values of 1 or more selected tuples.

1/2/2020

Write Create table stmts for the tables depicted in the schema diagram below.

```
CREATE table salesman
(
    Salesman_id integer PRIMARY KEY,
    name varchar(30),
    City varchar(20),
    Commission decimal(5,2)
);
```

```
CREATE table Customer
(
    customer_id integer PRIMARY KEY,
    Cust_name varchar(30),
    city varchar(20),
    Grade integer,
    Salesman_id integer REFERENCES Salesman(salesman_id)
);

CREATE table Order
(
    Ord_no integer PRIMARYKEY,
    Purch_Amt Decimal (8,2),
    Ord_date DATE,
    Salesman_id integer REFERENCES Salesman(salesman_id),
    Customer_id integer REFERENCES Customer(customer_id)
);
```

CId	CName
200	Mary
201	Nancy
202	John
203	Smith
204	Max

Write SQL Queries to

1 retrieve customer details whose cid is 203

- SELECT * from customer where cid = 203 ;

2 retrieve the city and grade of the customer whose name is John

- SELECT city, grade from customer where name='John';

3 retrieve name and city of customer whose grade is 1 or 2

- SELECT name, city from Customer where grade = 1 OR grade = 2;

4 retrieve cid, sid and cname for all those customers whose grade is 1

- SELECT cid, sid, cname from customer where grade = 1 ;

5. retrieve details of c.name and city of those customer serviced by salesman whose id is 100

- SELECT c.name, city from customer where sid = 100 ;

6 ~~Selec~~ retrieve details of Salesman whose id is bet^n 100 and 102

- SELECT * from Salesman where id >= 100 AND id <= 102;

7 name, city and Sid of Salesman whose name starts with J

- SELECT name, city, Sid from Salesman where Sname LIKE 'J%';

8 details of salesman whose commission is greater than 12.00

- SELECT * from salesman where commission > 12.0

9 Sid and name of Salesman whose cityname ends with n

- SELECT Sid, name from Salesman where Cityname LIKE '%n';

10 name, city of salesman whose commission lies bet^n 25.00 and 35.00

- SELECT name, city from Salesman where Commission >= 2500 AND commission <= 35.00;

* BETWEEN Operator

Syntax

SELECT * from tablename where col.name BETWEEN 100 AND 102

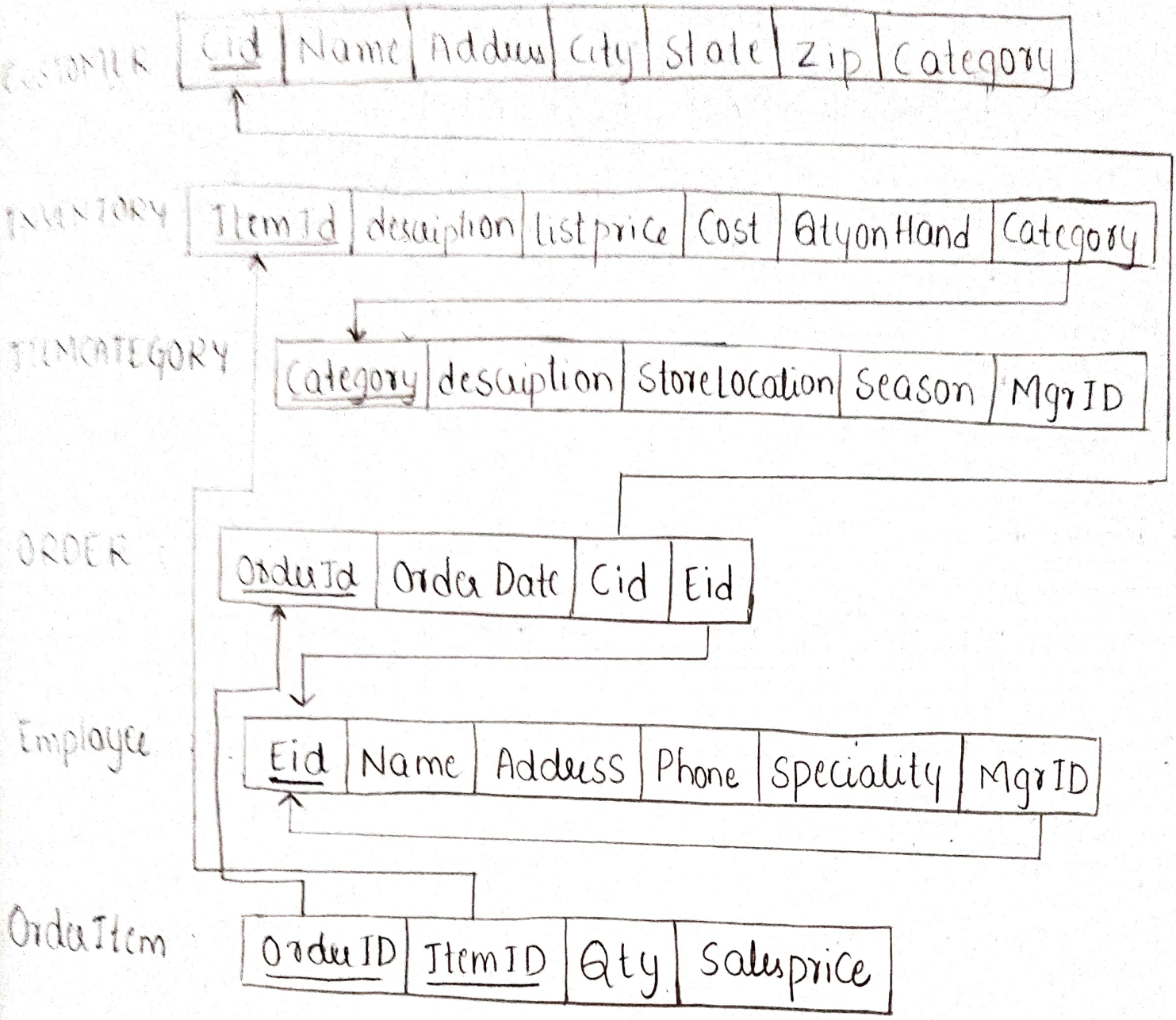
- The between keyword allows u to verify the value of an attribute to be lying betn a specified range

* LIKE operator

- Substring matching
- wildcard symbols,
  - % - matches 0 or more char's
  - _ - matches exactly 1 character

For the set of relations given below,

1 draw the relational schema diagram
2 write create table stmts. for each of the relations involved in the schema



Aggregate functions in SQL

5 funct^{n}s . COUNT(); SUM(); MAX(); MIN(); AVG()

Aggregate funct^n is a funct^n where the values of multiple rows are grouped together as input on certain criteria to form a single value of more significant meaning.

1. Count( ) : Returns count of the no. of rows
Count(attribute name) : Returns count of all the no. of "non-NULL" values over the given column

2. SUM( ) :

3/1/2020

Salesman (salesman_id, name, city, commission)

1. SELECT * from salesman where city = 'Paris' OR City = 'Rome';

2. SELECT *
from salesman
where city IN ('Paris', 'Rome');

3. SELECT * from salesman where city <> 'Paris' OR city <> 'Rome'; / where City NOT IN PARIS /

4. SELECT * from salesman where salesman_id BETWEEN 3007 AND 3009 ; / IN (3007, 3008, 30
NOT city IN PARIS

5. SELECT * from salesman where commission BETWEEN 0.12 AND 0.14;

6 7. SELECT * from salesman where name BETWEEN 'A' AND 'L'

8 SELECT * from salesman where name NOT BETWEEN 'A' AND 'L'

9. SELECT * from salesman where name LIKE 'B%';

10 SELECT * from salesman where name Like '%n';

11 SELECT * from salesman where name LIKE 'N__L%';

IN Operator :

- It is a special operator supported in SQL and helps to check the value of an attribute from among a list of values specified in a comma separated list immediately following IN operator.

NOT IN : It negates the effect of IN operator

ALTER TABLE Statement :

This command is used to change the structure of an existing table

- It helps to add or delete columns, create or destroy indices, change the type of existing ~~table~~ columns or rename the column or the table itself

i. Adding a column :

```
ALTER TABLE tablename
ADD Col_name datatype,
ADD Col_name datatype
  :
  ;
```

ii. droping column

```
ALTER TABLE tablename
DROP Column Column_name ;
```

not a PRIMARY KEY

iii. Adding / Dropping constraints.

ALTER TABLE tablename
ADD CONSTRAINT <constraint_name> constraint ;
type of constraint

Eg: ALTER TABLE salesman
ADD CONSTRAINT salesman_PK PRIMARY KEY
(sid);

- Assigning names to constraints makes it easier to specify the constraint name when the constraint is to be dropped.

ALTER TABLE tablename
DROP constraint_name;

iv. Change type of column

ALTER TABLE tablename
MODIFY (coln_name newType);

4/2/2020

1. The update stmt without the where clause updates all rows in the table
2. The new value for an attribute can be specified using arithmetic expression

3 The 'where' clause when used with the update statement restricts the no. of updates to those rows that satisfy the condition.

4. The update stmt can be used to update multiple attribute values at a time.

1. The delete stmt when used causes 0 or more rows to be deleted from the table

Eg:

COLOR

id	color
1	Red
2	Blue
3	Orange
4	Green
5	Pink
6	Black

SHAPES

id	color_id	shape
1	1	'square'
2	1	'star'
3	2	'Triangle'
4	4	'Circle'

```
CREATE table color
(
    id integer PRIMARY KEY,
    color varchar(10),
);
```

```
CREATE table shapes
(
    id integer PRIMARY KEY,
    Color_id integer REFERENCES color(id),
    Shape varchar(10),
);
```

1. Add a column called dimension to the shapes table

- ALTER table shapes
  ADD dimension integer;

2. ALTER table shapes
   DROP column dimension;

3. ADD foreign key constraint to shapes table.

   ALTER table shapes
   ADD CONSTRAINT colorid-FK FOREIGN KEY;
   REFERENCES Color (id);

7/2/2020

The output of select stmt is in no particular order. However you may want to organise or sort the output of a query in a particular order. The ORDER BY clause helps you to exactly do this.

Syntax

```
Select <column_list>
from tablename
where <condition>
ORDER BY <column> [ASC|DESC]
```

Table Agents

A_Code	A_name	Working_Area	Commission	Phone no	Co

- Select A_name, Working_Area, Commission from Agents ORDER BY A_Code ;
- Select A_name, Working_Area, Commission from Agents ORDER BY A_Code [DESC] ;
- Select A_name, working_Area, Commission from Agents ORDER BY Working_Area, A_Code ;

- Retrieve shape_id, shape and color_id for shapes whose color_id is betⁿ 1 and 4, ordered by "shape" in descending order.

```
SELECT shape_id, shape, color_id from shapes
where color_id BETWEEN 1 AND 4 ORDER BY SHAPE DESC;
```

- Retrieve shape_id, shape where the output is sorted based on the type of shape.

```
SELECT shape_id, shape from shapes
ORDER BY shape;
```

-

GROUP BY :

```
SELECT column1, colum 2,
FROM table_name
WHERE <condition>
GROUP BY column1, column 2   grouping attributes
[HAVING] <condition>;
```

- Grouping attributes have to be in select attributes