Date
17/08/23

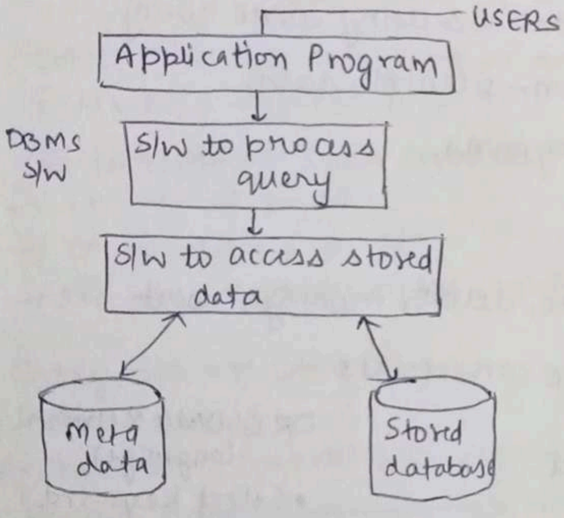
UNIT-1 INTRODUCTION

Data - raw facts or information (processed data)
known facts that can be recorded and that have implicit meaning

Database - collection of data with coherent relations and inherent meaning.
Generally, a collection of related data. It is designed, built and populated with data for a specific purpose.
Intended group of users.

Database Management System (DBMS)
A collection of programs that enables users to create and maintain a database.
- software tool
- maintains the data inside database
- data consistency
- access restriction

Architecture of DBMS Software



DBMS Catalog
→ rules (constraints): Set of rules and restrictions
→ storage structure of data - method to store data
→ Meta data

Meta data : data about data

Difference between traditional file system and DBMS
(i) Data inconsistency and redundancy
(ii) Difficulty in accessing data - different files design in different file formats
(iii) Data Isolation - difficult to isolate a particular data
(iv) Integrity - In case of merging two files
(v) Atomicity - Either transaction is fully completed or return to its previous state which is not possible in traditional system.
(vi) Concurrent Access - parallely access
In DBMS → Scheduling system

In File System: 9500 9900
10,000
500 A
100 B

(vii) Security - data abstraction - hiding some data from users
not possible in file system as different application programs for different queries.

Schema

Overall design of the database

State - at a particular moment the content of database is called state or instant of the database

Database users or Actors

1- Database Administrator (DBA) : topmost actor of the database
overall design is decided by DBA.

→ specifying integrity constraints
→ access restriction
→ backup (data recovery)
→ monitoring performance

2- Designers

3- Application Programmers

4- End Users

Application Programmers : DML calls

Sophisticated users: who can access the database using some query

Specialized users : for particular application - scientific data

Naive users: access using application programs.

Database language

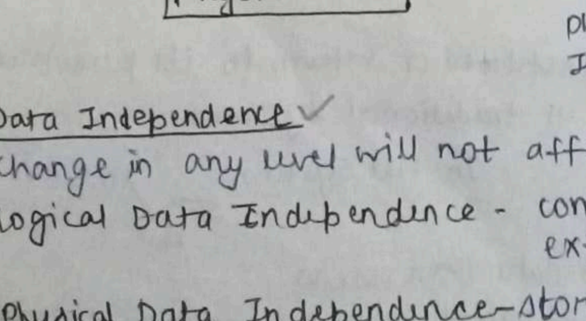
Data Manipulation Language (DML) - update, delete, modify, insert

Data Definition Language (DDL) - defining the constraints

DRL (Data Retrieval language)
- Select keyword

Data abstraction : hiding some portion of data from users

Three Level Architecture



External Schema
show some portion of data according to requirement.

Conceptual Schema {describes content of the data entity, values, constraints}

Storage
physical or
Internal schema

Data Independence

Change in any level will not affect the upper level

→ logical Data Independence - conceptual change will not affect the external level

→ Physical Data Independence - storage structure changes.

23/08/23

External Schema — External Schema — External Schema

Conceptual Schema — Logical Data Independence

Physical schema — Physical data Independence

DATABASE LANGUAGES

- Data Definition Language (DDL)
  Allows DBA to create entities
- Data Manipulation Language (DML)
  tell system exactly how to manipulate data

→ Procedural DML

→ Non-Procedural DML → allows user to state what data is needed rather than how it is to be retrieved.

DATA MODELS → conceptual tool
real world objects and events

Components of Data models
→ Structural part
→ Manipulative part
→ Set of Integrity rules, which ensures that the data is accurate

Categories of Data Models — Entities → real world objects

1- Object Based
2- Physical → how data is stored in computer
3- Record Based →
- Hierarchical (Tree) (1 to n) — Insert, Update, Delete } operations
- Network model (N to M) (Graph)
- Relational

OBJECT BASED
- Entity-Relationship
- Object Oriented
  ↳ actions and behaviours of entities

Advantages of Hierarchical Model
- Simplicity
- Data Security
- Data Integrity
- Efficiency

Disadvantages
- Implementation Complexity
- Database Management Problems
- lack of Structural Independence
- Operational Anomalies
  ↳ Insertion, Deletion, Updation
- Implementation Limitation
  (doesn't support N:N relationship)

Network Model
Advantages
- Conceptual simplicity
- Capability to handle more relationship types
- Easier data access

Highly Complex

Relational Model (Table)

Cardinality of a relation → no. of tuples

Degree of relation - no. of attributes

(Each column in the tuple is called an attribute)

<u>Disadvantages</u>

Hardware overheads

Ease of design can lead to bad design

'Information island' problem

Two database
- Centralised
- Client-server

<u>24/08/23</u>

<u>E-R Model</u>

Entity — Attribute - property of entity

student
e_1
e_2
e_3
⋮
e_n

→ simple vs composite

→ single valued vs Multi-valued (more than one value)

(Ex - height, age) — Ex - degree

→ Stored vs Derived — Ex - stored (D.O.B) → Age can be derived

→ Null Attribute
- doesn't exist
- exist but unknown to designer

Name → First, Middle, Last

Address → Flat no., Rer, City, District

→ complex Attribute

composite as well as multi-valued

address (flat no., Phone no. (Ph1, Ph2, Ph3), Street No., City)

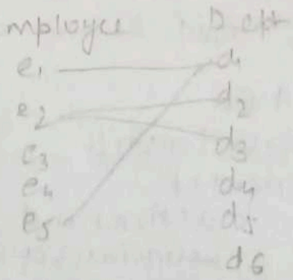
$\underbrace{\text{address}}_{\text{composite}}$ — $\underbrace{\text{Phone no.}}_{\text{multi-valued}}$ — complex

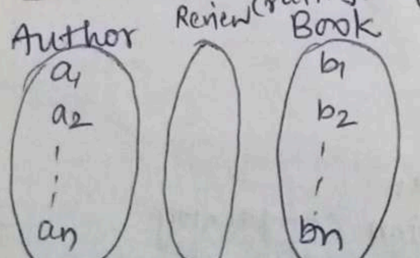
<u>Domain</u> - set of attribute values / Range of attribute values

<u>RELATIONSHIP</u>

Association among several entities

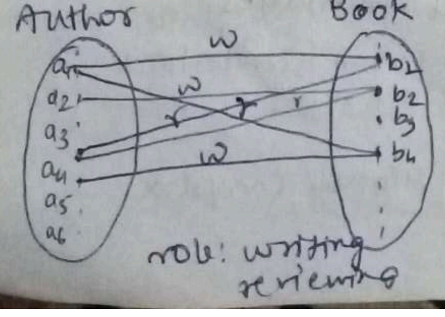
<u>relationship set</u> - (attribute of review)





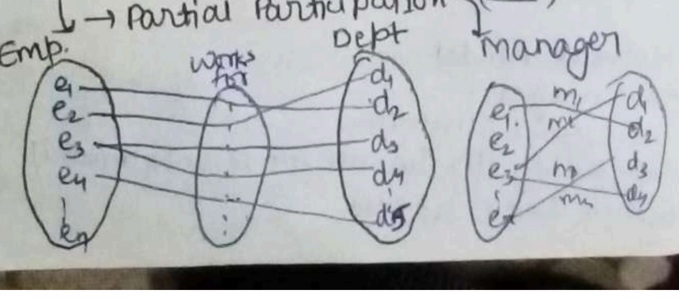
<u>Recursive Relationship</u>

(more than one roles)

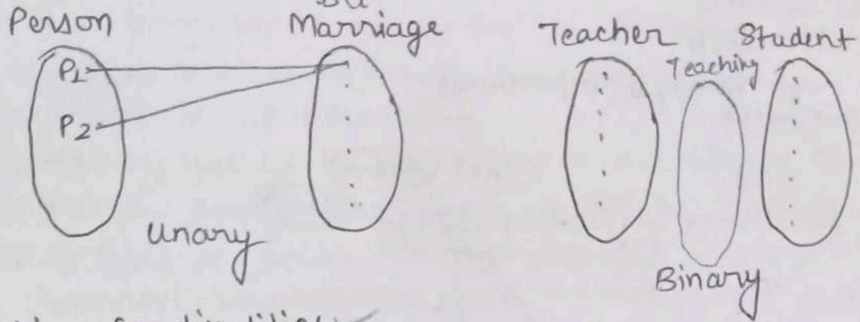


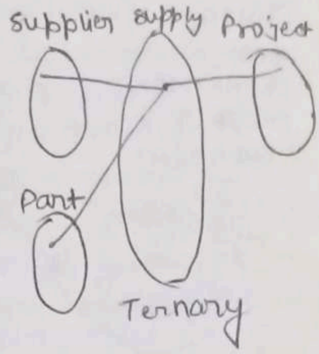
→ Total Participation (——) single link

→ Partial Participation (═══) double link



Relationship → Unary, Binary, Ternary, ...
degree of relationship → how many entity sets are connected to a particular set

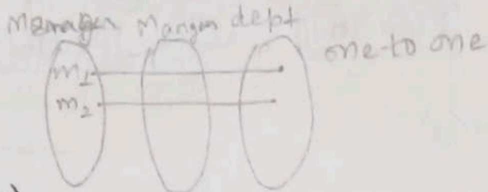




Mapping Cardinalities

Express the no. of entities to which another entity can be associated via a relationship set.

for binary relationship-
- one to one (1 to m)
- one to many
- many to one
- many to many (m_1 to m_2)



ER diagram- cardinality ratio (show)

Key - uniquely identified the entity
- super key → set of attributes through which we can uniquely identify an entity from entity set.
- Candidate key
- (const) Primary Key

sup (sno., sname, city, address, dob)

Primary key: should not be null and should be unique.

Candidate key: minimal subset of superkey is call the candidate key

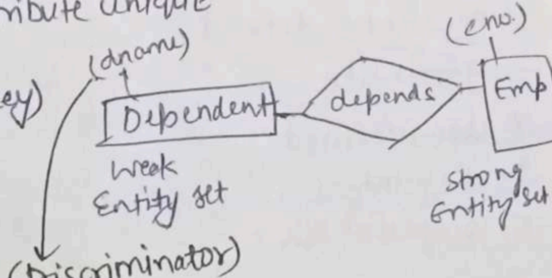
(sno.)
(sno., sname) ×
(sname, address)
(sno, city) ×
(sname, dob)
address ×

candidate key chosen by database designer is primary key.

(sno.) and (sname, address)
↳ If not chosen then Alternate key.

Unique key- task is to, if want to make attribute unique, doesn't allow duplicate value, but can take null value

- Weak Entity set (doesn't have primary key)
- Strong Entity set (have primary key)



ER Diagram

□ Rectangle: Entity sets
◇ Diamond: Relationship sets
○ Ellipse: attributes → ◎ → multi-valued attributes
eno. → primary key
dname → discriminator in weak entity
⧈ - Weak Entity
◈ → Relationship among weak entity (rows of relationship should be specified)

30/08/23

E-R Diagram Assumptions

Entities with attribute

EMP (Name...)

DEPARTMENT

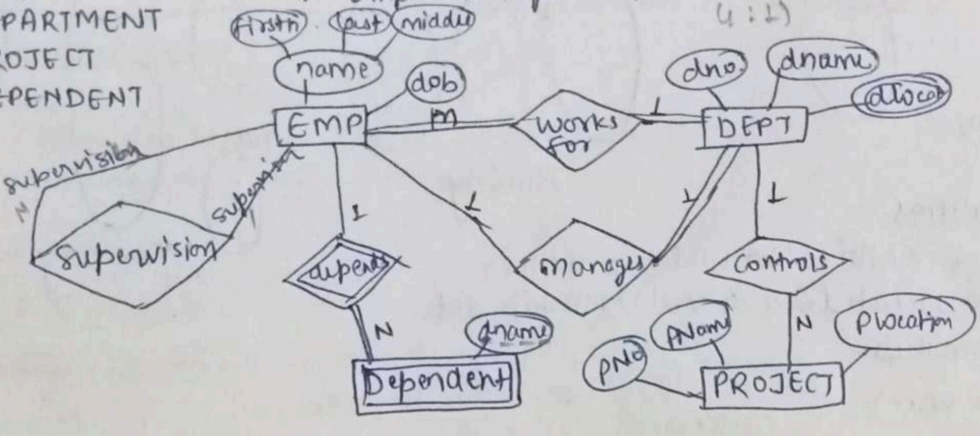
PROJECT

DEPENDENT

Relationship

1- Emp manages Department. (1:1)

Mention Primary key and assumptions, cardinality, participation



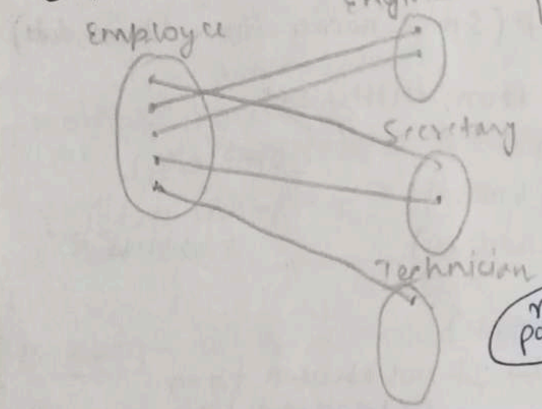
Extended E-R Diagram

→ Specialization

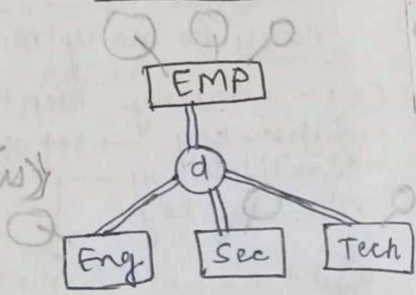
→ Aggregation

→ Generalisation → combining

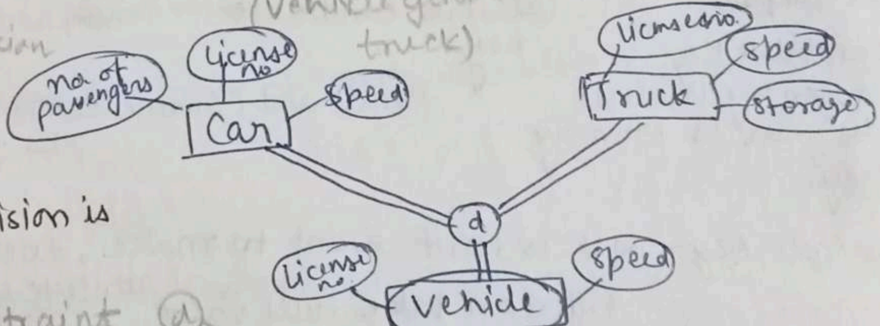
{Special attribute through which they differ from eachother (subclass)}



Specialization



(Vehicle generalized form of Car & truck)



superclass - subclass division is based on -

1- Predicate defined constraint (d Jobtypes)

2- Attribute (no. of passengers & storage)

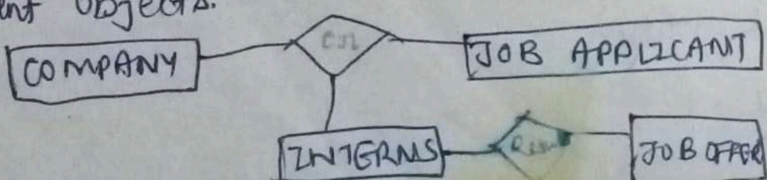
3- Userdefined

4- disjointers (division in such a way that all subentities are disjoint of other)

5- completeness

Aggregation Aggregate object communicates with manager through another relationship. (another entity)

It is an abstraction concept for building composite objects from their component objects.

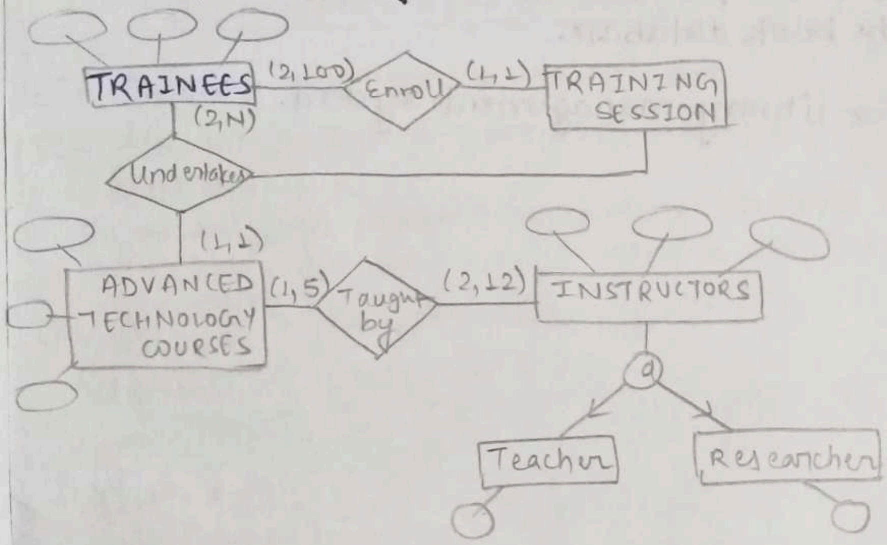


06/09/23

Assignment on Thursday

Design an ER diagram for an IT training group database that will meet the information needs for its training program. clearly indicate the entities, relationship and the key constraint. The description of the environment is as follows —
The company has 12 instructors and can handle upto 100 trainees for each training session. The company offers 5 advanced technology courses, each of which is taught by a team of two or more instructors. Each instructor is assigned to a maximum of two teaching team or may be assigned to do research. Each trainee undertakes one advanced technology courses for training session.

(Primary key in each identity)



Cardinality Ratio (constant)
(1,5)
(2,12)

Assumptions mentioned at below of ER digram

Assume:
2 & N for Trainee in Advanced course

Design an ER diagram for a company database as per the requirement given below, make appropriate assumptions to complete the specification
(a) The company stores the information about the currently working employee. The information includes emp.no., name, gender, salary, dob, doj, address and Phone no.. Each employee works for a department for a particular project for a particular no. of hours.
(on)
(b) The information about department includes dept no. and dept. name. Each department controls some projects currently running in the company. Also each department is managed by a particular employee who becomes the manager for the department. This employee is also supervises all the other employees in that dept.
(c) The project information includes Project no., Project name and its description.
(d) An employee can work for only one department. However, a dept. can have any no. of employees.
A department is managed by only one manager and a manager can manage only one dept.. A dept. can control any no. of projects. However, one project can be handled by only one dept. Any no. of emp. can work on any no of projects.

Draw the ER diagram specifying the key attributes of each entity type, you name and mapping cardinality.

② Consider a bank database having customer, loan, account, branch, employee as entity type.

Each branch of bank allows customer to open accounts and borrow loan. A customer can open more than one account and one account may also belong to one or more customer. Similarly a customer can take out more than one loan and a loan may be held by more than one customer. The bank has a no of employees working in different branches of bank. Add appropriate attributes for each entity type. Represent the key attribute, weak entity type and cardinality ratio. Make appropriate assumptions to complete the specifications. Design the ER diagram for the bank database.

③ Draw the ER diagram for library management system.

Referential Integrity Constraint

Primary & Foreign key
↓
refers one table to another
↳ should be consistent (consistency maintained)

13/0
Co
I- Fi
ER
DE
PR
In
II-
D
III-
IV-
8
IV:
S
E
D
P
V-
W
VI
R
→
U
si

Conversion of E-R model to Relational Model

Relational table of drawn E-R diagram

I- Finding the entities, Put the attributes.

Separate table for multivalued attributes

EMP

SD	Fname	mname	lname	add	sal

DEPT

name	dno.	mgrid	mgr start date

(mapping 1 to 2)

mgr start date → attribute of manages relationship

PROJECT

name	no.	location

In composite attribute - make it in simple form

II- Representing weak entity (connected to strong entity) ↳ primary key of that strong entity (ID)

DEPENDENT

eID	name	sex	dob	relation

III- mapping with one to one relation

(full participation → add primary key of other table as foreign key)

IV- full participation from both sides of relationship

single relation created by joining EMP and DEPARTMENT and primary key of both is combined Primary Key

IV: One to N relationship (1:N)

(SSN: supervision supervisor No)

Supervision (self-relation)

EMP

ID	Fname	mname	lname	add	sal	ESSN

EID	SSN
1A	1D
1B	1D
1C	1D
1D	NULL

DEPT

name	dno.	mgr id	mgr start date

PROJECT

~~id~~ name	no.	location	dno.

(Prime N same foreign key add)

(dept 1 project N)

V- M to N relationship

separate table of relationship contains Primary key of Project & Emp

works on

id.	Pno	hours

(hours → attribute)

VI- Mapping with multivalued attributes

make the different table for this attribute named as (entity-attribute). Primary key of dept. table acts as primary key of dept-location table

Dept-location

dno.	location

RELATIONAL ALGEBRA

Formal language method to represent a query.

→ Unary and Binary Operators

Unary ↳ used on single table

Binary ↳ more than one table (relation)

AND OR ($\wedge$ $\vee$)

Sigma Operator (σ): used for selection

$\sigma_{condition}$ (Table name)

$\sigma_{sal > 10000 \wedge location = 'patna'}$ (emp) ($\wedge$: and operator)

Select details of employee who lives in Patna or Delhi

$\sigma_{location = 'Patna' \vee location = 'Delhi'}$ (emp)

$\sigma^{(emp)}_{loc='Patna' \vee 'Delhi'}$

For column Selection (Π)

→ Select Emp Name and Emp Address from Employee Table

$\Pi^{(table)}_{columnname1, columnname2}$ $\Pi^{(Emp)}_{Empname, empAddress}$

→ Select Name and addr of all employees whose salary > 10,000.

$\Pi_{name, address}(\sigma^{(emp)}_{sal>10000})$

14/09/23

$\sigma^{(tablename)}_{(condition)}$ $\sigma^{emp}_{sal>10000}$ $\sigma^{(emp)}_{sal>10000 \wedge address='Patna'}$

$\Pi^{(emp)}_{name, sal}$ $\Pi^{(R) \to \text{Relation Table}}_{columnname1, columnname2}$ $\sigma^{(emp)}_{address='Patna' \vee Address='Delhi'}$

emp

name	address	sal	dob	doj

$R_1 \leftarrow \sigma^{(emp)}_{sal>20000 \wedge address='Patna'}$

$Result \leftarrow \Pi^{(R_1)}_{name, address}$

R_1

$\Pi_{name, address}(\sigma^{(emp)}_{sal>20000 \wedge address='Patna'})$

Binary Operator

Union (U)

Arity → column
↓
same (condition for union)

emp1

name	address
aaa	Kol
bbb	Kol
ccc	Patna
ddd	Delhi

emp2

name	address
aaa	Kol
kkk	Mumbai
mmm	Bangalore
ddd	Delhi

emp1 ∪ emp2

aaa	Kol
bbb	Kol
ccc	Patna
ddd	Delhi
kkk	Mumbai
mmm	Bangalore

emp1 ∩ emp2 → common tuple

aaa	Kol
ddd	Delhi

emp1 − emp2

bbb	Kol
ccc	Patna

Cartesian Product

emp

Id	name	address	dno.
1A	aaa	Kol	1A
2B	bbb	Kol	1A
3C	ccc	Patna	1B
4D	ddd	Delhi	1C
5	xxx	Patna	1G

dept

dno.	dname
1A	MTech
1B	MCA
1C	BTech
1D	BCA

Arity of new table = arity of table1 + arity of table2

emp × dept

no. of tuples = tuple of table1 × tuple of table2

tuple = 4×3 = 12

Arity = 4+2 = 6

Id	name	address	dno.	dno.	dname
1A	aaa	Kol	1A	1A	MTech
1A	aaa	Kol	1A	1B	MCA
1A	aaa	Kol	1A	1C	BTech
2	bbb	Kol	1A	1A	MTech
2	bbb	Kol	1A	1B	MCA
2	bbb	Kol	1A	1C	BTech
3	ccc	Patna	1B	1A	MTech
3	ccc	Patna	1B	1B	MCA
3	ccc	Patna	1B	1C	BTech
4	ddd	Delhi	1C	1A	MTech
4	ddd	Delhi	1C	1B	MCA
4	ddd	Delhi	1C	1C	BTech

Cartesian Product is the first step of joining

Joining

Joining on the basis of some conditions.

1. Inner Join – those tuples that do not match in two tables will not be included in joining. (⋈)

for e.g. 1 pin dept table or 1 in emp table

- Theta Join – on the basis of some condition ($\bowtie_\theta$).

emp $\bowtie_{dname='MTech'}$ dept

- Equi Join: emp $\bowtie_{emp.dno.=dept.dno.}$ dept

(same/equal values)

First table common attribute value = Second table common attribute value

1	aaa	Kol	1A	1A	MTech
2	bbb	Kol	1A	1A	MTech
3	ccc	Patna	1B	1B	MCA
4	ddd	Delhi	1C	1C	BTech

- Natural Join | common attribute which is basis of equi joint should be represented once then it is called Natural Join. (reducing redundance)

emp ⋈ dept

2– Outer Join
- Left ⟕
- Right ⟖
- Full ⟗

emp ⟖ dept (Right Outer Join)

Right side row not matching with left then row is putted on

1	aaa	Kol	1A	1A	MTech
1	aaa	Kol	1A	1A	MTech
2	bbb	Kol	1B	1B	MCA
3	ccc	Patna	1C	1C	BTech
4	ddd	Delhi	1D	1D	BCA
NULL	NULL	NULL			

Left Outer Join (Left side Row)

emp ⟕ dept

Full (all rows and columns) (Both side Row)

⟗

1 aaa	Kol	1A	1A	MTech	
2 bbb	Kol	1A	1A	MTech	
3. ccc	Patna	1B	1B	MCA	
4 ddd	delhi	1C	1C	B.Tech	
5 - xxx	Patna	NULL 1G	NULL	NULL	
NULL NULL	NULL	1D	1D	BCA	

Rename Operator (unary)

$\rho(R, E)$ — R → new name, E → prev name

$\rho(e, emp)$

Division Operator $(R_1 \div R_2)$

'for all objects having the specified property'.

If this type of phrases are present in our query, then we use division operator.

R_1

A	B
a_1	b_1
a_1	b_2
a_1	b_3
a_1	b_4
a_2	b_1
a_2	b_2
a_3	b_2
a_4	b_2
a_4	b_4

R_2

B
b_2
b_4

$R_1 \div R_2$

a_1
a_4

$\Big\}$ output

Aggregation Operation

$\mathcal{G}_{max(sal)}$ (tablename) $\quad \mathcal{G}_{min(sal)}$ (tablename) $\quad \mathcal{G}_{avg(sal)}$ (emp)

max(Column name)

min(Column name)

Avg(   )

count(   ) (no. of tuples it counts) for that column

Date
21/09/23

Book (<u>ISBN</u>, Book-title, Category, Price, Copyright date, year, Pagecount, PID)
Publisher (<u>PID</u>, Pname, Address, State, Phone, emailID)
Author (<u>A-ID</u>, Aname, City, State, Zip, Phone, URL)
Author_Book (<u>A-ID</u>, <u>ISBN</u>)
Review (<u>R-ID</u>, ISBN, Rating)

1- Write the queries in relational algebra

(i) Retrieve the city, Phone and URL of the Author whose name is RAM

$$\Pi_{city, Phone, URL}(\sigma_{Aname = 'Ram'}(Author))$$

(ii) Retrieve the name, address and phone of all the publishers located in Patna (Bihar).

$$\Pi_{name, address, phone}(\sigma_{\cancel{Address}\ State = 'Bihar'}(Publisher))$$

(iii) Retrieve the title and price of all the text book with a pagecount greater than 600.
↳ category

$$\Pi_{Booktitle, price}(\sigma_{pagecount > 600 \wedge category = 'Textbook'}(Book))$$

(iv) Retrieve the ISBN, title and price of the books belonging to either novel or language book category.

$$\Pi_{ISBN, title, price}(\sigma_{category = 'novel' \vee Category = 'language'}(Book))$$

(v) Retrieve the ID, name, address and phone of the publishers publishing novels.

$$\Pi_{Publisher.PID, name, address, phone}(\sigma_{category = 'Novel'}(Book \bowtie_{Publisher.PID = Book.PID} Publisher))$$

↳ mentioned as present in both table

(vi) Retrieve the title and price of all the books published by Hills Publication.

$$\Pi_{book.title, book.price}(\sigma_{Pname = 'Hills'}(Book \bowtie_{Publisher.PID = Book.PID} Publisher))$$

can write Pname or publisher.pname also

(vii) Retrieve the book_title, Reviewers ID and rating of all the text books

$$\Pi_{Book.book\_title,\ Review.R\_ID,\ Review.Rating}\left(\sigma_{Book.category = 'Textbook'}\left(Book \bowtie_{Book.ISBN = Review.ISBN} Review\right)\right)$$

(viii) Retrieve the title category and price of all the books written by Charles Smith.

$$\sigma\ \cancel{Author} \bowtie_{Author.A\_ID = Author\_Book.A\_ID} Author\_Book \bowtie_{Author\_Book.ISBN = Book.ISBN} Book$$

$$R_1 \leftarrow Author \bowtie_{Author.A\_ID = Author.Book.A\_ID} Author\_Book$$

$$\Pi_{Book.Category,\ Book.Book\_title,\ Book.price}\left(\sigma_{R_1.Aname = 'Charles\ Smith'}\left(R_1 \bowtie_{R_1.ISBN = Book.ISBN} Book\right)\right)$$

(ix) Retrieve the ID, name, URL of the Author and category of the book C++.

$$\Pi_{Author.A\_ID,\ Author.Aname,\ Author.URL}\left(\sigma_{Category = 'C++'}\left(Author \bowtie_{Author.A\_ID = Author\_Book.A\_ID} Author\_Book \bowtie_{Author\_Book.ISBN = Book.ISBN} Book\right)\right)$$

(x) Retrieve the Book_title, price, URL, Author name for the publishers Hills publication.

$$\left(Book \bowtie_{Book.PID = Publisher.PID} Publisher\right) \bowtie_{Book.ISBN = Author\_Book.ISBN} \left(Author \bowtie_{Author.A\_ID = Author\_Book.A\_ID} Author\_Book\right)$$

$$\Pi_{Book.Book\_title,\ Book.Price,\ Author.URL,\ Author.Aname}\left(\sigma_{Publisher.Pname = 'Hills'}\left(\left(\left(Book \bowtie_{Book.PID = Publisher.PID} Publisher\right) \bowtie_{Book.ISBN = Author\_Book.ISBN} Author\_Book\right) \bowtie_{Author.AID = Author\_Book.A\_ID} Author\right)\right)$$

(xi) Retrieve and name and address of Publishers who have not published any book.

$$R_1 \leftarrow \Pi_{PID}(Publisher)$$

$$R_2 \leftarrow \Pi_{PID}(Book)$$

$$R_3 \leftarrow R_1 - R_2$$

$$\Pi_{Pname,\ address}\left(R_3 \bowtie_{R_3.PID = Publisher.PID} Publisher\right)$$

In SQL –

PID not in

(xii) Retrieve the name of all the publishers, ~~on~~ the ISBN and the title of the book published by them.

$$\Pi_{\text{Publisher.Pname}, \text{Book.ISBN}, \text{Book.title}} \left(\text{Book} \bowtie_{\text{Book.PID} = \text{Publisher.PID}} \text{Publisher} \right)$$

(xiii) Retrieve the ID and the no. of books written by each other.

$$_{\text{A\_ID}} \mathcal{G}_{\text{count (ISBN)}} \text{(Author-Book)}$$

Author ID

(group by) A.ID

SBN

(xiv) Retrieve the ISBN and Average rating given to each book.

$$_{\text{ISBN}} \mathcal{G}_{\text{avg (Rating)}} \text{(Review)}$$

(xv) Retrieve the name of the publishers who have published all categories ~~of~~ books. (Division Operator)

$$R_1 \leftarrow \Pi_{\text{PID, Category}} \left(\text{Book} \bowtie_{\text{Book.PID} = \text{Publisher.PID}} \text{Publisher} \right)$$

$$R_2 \leftarrow \Pi_{\text{Category}} \text{Book}$$

$$R_1 \div R_2$$

Date 27/09/23 RELATIONAL CALCULUS (non-procedural process to write query)

Class Test next Thursday — Till Relational Calculus

(1) Tuple Relational Calculus

(2) Domain Relational Calculus

Relational Calculus is a non-procedural or declarative query language as it specifies what to retrieve rather than how to retrieve. In the relational calculus, queries are expressed in the form of variables and formulas consisting of these variables.

The tuple relational calculus consisting of three components:

- R → specify the relation for which the tuple variable T is defined.
- A condition P(T) - On the basis of which set of tuple is to be retrieved
- An attribute list specifying the required attributes to be retrieved for the tuples satisfying the given condition

$\{T \mid P(T)\}$

Select the ~~autho~~ bookname for the book where price is greater than 600

$\{T.\text{Bookname} \mid \text{BOOK}(T) \wedge T.\text{Price} > 600\}$
(iii) (ii)

Select the bo

$T_1.A$ opr $T_2.B$

$\wedge \ \vee \ < \ > \ \leq \ \geq$

Write down a general expression of tuple relational calculus consist of a tuple variable T and a formula P(T). A formula is made up of atoms and can be of any of the forms.

1. $P(T)$
2. $T_1.A$ opr $T_2.B$
3. $T.A$ opr $T.B$
4. $T.A$ opr C

$\Big\}$ ways to represent the formula

$\neg$(Not)

$F_1 \vee F_2$, $F_1 \wedge F_2$

$(\neg(F_1) \vee F_2)$

Quantifier (use only when confident about it and to join table)

Quantifiers are special symbols that are used to quantify a tuple variable T.

→ Universal $\forall$ (for all)

→ Existential $\exists$ (for some)

The expression $\forall T$ means for every occurence of T and the expression $\exists T$ means for some occurence of T.

A tuple variable T is said to be bound if it is quantifieq ie it appeared either in $\exists T$ or $\forall T$ clause, otherwise it is free. not free

Write down a safe expression (is an expression that results with a finite no. of output).

$\{T.Bookname \mid \neg(Book(T))\}$ → unsafe expression

1. $\neg f$ (true when f is false)
2. $F_1 \wedge F_2$ (true if both are true)
3. $F_1 \vee F_2$
4. If $F_1 \Rightarrow F_2$ (F2 is true whenever F1 is true)
5. $(\exists T)(F)$ True if any
6. $(\forall T)(F)$ (T) true if F is true for every tuple

(i) Retrive the city, phone and URL of the author whose name is RAM

$\{T.city, T.phone, T.URL \mid Author(T) \wedge T.Aname = \text{"Ram"}\}$

(i) Retrieve the name, address and phone of all the publishers located in Patna, Bihar.

{T.name, T.address, T.phone | Publisher(T) ∧ T.state = "Bihar"}

(ii) Retrieve the title and price of all the textbook with a pagecount greater than 600.

{T.title, T.price | Book(T) ∧ T.pagecount > 600 ∧ T.category = "Text"}

(iv) Retrieve the ISBN, title, price of the books belonging to either novel or language book category.

{Book.ISBN, Book.Title, Book.price | Book(T) ∧ (T.category = "Language" ∨ T.category = "Novel")}

(v) Retrieve ID, name, address and phone of the publishers publishing novel category

{T.PID, T.Name, T.address, T.phone | Publisher(T) ∧ (∃S) (Book(S) ∧ S.PID = T.PID ∧ S.category = "Novel")}

(vi) Retrieve the title and price of all books published by Hills Publication

{T.book_title, T.price | Book(T) ∧ (∃S)(Publisher(S) ∧ S.name = "Hills" ∧ S.PID = T.PID)}

(vii) Retrieve the book title, Reviewers ID and rating of all the text book

04/10/23

Domain Relational Calculus

$A(a_1, a_2, a_3, a_4)$

$\{T \mid P(T)\}$

$\{a_4 \mid A(a_1, a_2, a_3, a_4)\}$

$B(b_1, b_2, --)$ $a_1 = b_1$

Incomplete Notes

The domain relational calculus is based on domain variables which range over the values from the domain of an attribute rather than values for an entire tuple.

Formula in domain relational calculus are built-up from atoms where atom can be represented in the following way -

① $R(x_1, x_2, ---, x_n)$ $x_1, x_2, --, x_n \rightarrow$ domain variable name
$R \rightarrow$ Relation

② x_1 opr x_2 opr is the comparison operator
x_1 and x_2 are domain variables

OR

③ x_1 opr c $\wedge$ c opr x_2 opr is an operator and it is used for the comparison purpose and c is the constant i.e assignedent

④ F and !F $F \Rightarrow$ Formula

⑤ If F_1 and F_2 are formula

⑥ $F_1 \wedge F_2$, $F_1 \vee F_2$

If F is a formula then there exist x $\exists x(F)$
for all x $\forall x(F)$

① Retrieve the city, phone and URL of the author whose name is 'Ram'

Author $(a_1, a_2, a_3, a_4, a_5, a_6, a_7)$

$\{a_3, a_6, a_7 \mid (\exists a_2)$ Author $(a_1, a_2, a_3, a_4, a_5, a_6, a_7) \wedge a_2 = \text{'Ram'})\}$

$(\exists a_2)($ $)$

② Retrieve the name, address and phone of all the publishers located in Bihar.

Publisher $(b_1, b_2, b_3, b_4, b_5, b_6)$

$\{b_2, b_3, b_5 \mid \exists (b_4)($ Publisher $(b_1, b_2, b_3, b_4, b_5, b_6) \wedge b_4 = \text{'Bihar'})\}$

③ Retrieve the title and price of all the textbook with a pagecount greater than 600.

Book $(c_1, c_2, c_3, c_4, c_5, c_6, c_7, c_8)$

$\{c_2, c_4 \mid (\exists c_3)(\exists c_7)$ Book $(c_1, c_2, c_3, c_4, c_5, c_6, c_7, c_8) \wedge c_3 = \text{'Textbook'} \wedge c_7 > 600)\}$

④ Retrieve the ID, name, address and phone of the publishers publishing novels.

$\{b_1, b_2, b_3, b5 \mid (\exists c_3)(\exists c_8)$ Publisher $(b_1, b_2, b_3, b_4, b_5, b_6) \wedge$ Book $(c_1, c_2, c_3, c_4, c_5, c_6, c_7, c_8)$
$\wedge\ b_1 = c_8 \wedge c_3 = \text{'Novel'})\}$

(5) Retrieve the name and address of publisher who have not published any book.

In sql:

Select name, address from Publisher, Book
where published.PID = book.PID;
} For the publisher who published book.

Select name, address from Publisher
where PID not in (Select PID from publisher, books
where publisher.PID = Book.PID)

In domain calculus: Not symbol

$\{b_2, b_3 \mid (\neg)(\exists c_8)$ Publisher $(b_1, b_2, b_3, b_4, b_5, b_6) \wedge$ Book $(c_1, c_2, c_3, c_4, c_5, c_6, c_7, c_8)$
$\wedge\ b_1 = c_8)$

Till (ix) → Tuple and Domain
And all in sql and relational algebra

Constraints

Student, Teacher

Implicit constraints: Constraints that are inherent in the data model are called implicit constraints.

Domain
Integrity constraints
Entity
Referential
constraints

Explicit constraints: Constraints that can be directly expressed in schemas of the data model typically by specifying in the DDL
and hence must be expressed and enforced by the application program
These constraints are called application-based or semantic constraint.