Date: 11/1/23
Page:

Database Management System

Data - Known facts that can be recorded

Step 1 :- Gather the requirements coming from the client

DBMS ← company

Requirement — ER Schema designer —maps→ TABLES

Client ↔ ERD/EERD

TABLES ↓ SQL Queries.

Chapter - 3 (Relational Model)

Table name - Relation name
Rows - Tuples
Columns - Attributes

Relation Schema (denoted by capital R)

Relation Schema ~~capital~~ R ($A_1, A_2, \ldots, A_n$) is made up of Relation R and a list of attributes A_1, A_2, A_3 till A_n.

Roll NO	Name	Address

↓ constant

(Just the structure of the table is Relation Schema)

Date:
Page:

Relation State (denoted by small R)
It is the set of values ~~given~~ in a table R given at any point of time.

Roll NO	Name	Address
1	ABC	12 -
2	PQR	- - -
3	LMN	- - -

It can be change or update.

1. Domain of an Attribute

Every attribute ~~of~~ in a Relation R must lies in its domain. E.g like in Name we can't use 123 as name.

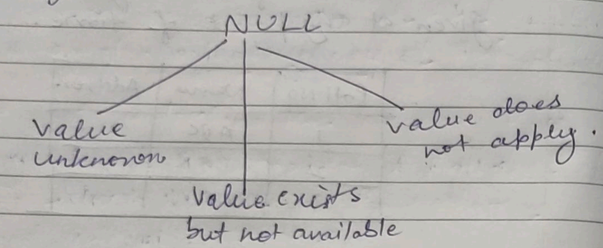
2. Degree of a Relation.
The no. of attributes in a Relation is called Degree of a Relation.

Ordering within a tuples matters & Ordering in a relation (i.e, b/w tuples) doesn't matters.

Missing cell or place can be filled with NULL

2023/03/03 21:47

In Database, NULL can be of three types:



Q What are the four types of constraints that must hold at any given ~~point~~ time?

1. Domain Constraint

2. Key Constraint

Super Key (SK)

A Super Key (SK) specifies a uniqueness constraint that ~~means~~ know ~~to~~ two distinct tuples in any state (or) of relation R can have the same value for SK.

Difference → SK contains the redundant attributes.

Similarity → Unique constraint (i.e, Roll No).

Every Relation has one default Super Key that is the set of all attributes.

STUDENT ⇒ Relation.

Registration	Roll No	Name	Address
101	1	ABC	---
102	2	ABC	---
103	3	PQR	---

Super Key → { Roll No, Name, Address } (Redundant attributes)

→ { Roll No, Address }

→ { Roll No, Name }

Redundant Attributes

Key → { Roll No }.

A Super Key can have redundant attributes.

NOTE

A Key is also a Super Key but not vice-versa.

Candidate Key (C-Key)

A Relation Schema can have more than one Key. Each of the Key is called Candidate key. E.g:- Registration & Roll No are candidate keys.

Primary Key (PK)

We can designate one of the candidate key as the primary key of the relation.

We just underline the primary key.

E.g From Registration & Roll No, if we select Roll No as Primary key, we just underline Roll No.

~~Note~~ Unique Key (UK)

The other ~~do~~ Candidate Keys are designated as Unique Key or Secondary Key

3. Entity Integrity Constraint

It states that primary key can't be null.

4. Referential Integrity Constraint

Social Security Number

EMPLOYEE (R_1) → Foreign Key DEPARTMENT (R_2)

SSN	Name	Address	DNO
11	ABC	123---	1
22	PQR	3----	2
33	Neeraj	----	NULL

Dnumber	Dname
1	Research
2	HR

The Referential Integrity Constraint states that a tuple in one relation that refers to another relation must refer to an existing tuple in that relation.

Foreign Key (FK)

A set of attributes FK in relation schema R_1 is the foreign key of R_1 that references relation R_2 if it satisfies the following two rules-

(i) The attributes in FK must have same domain as the primary key (PK) attributes of R_2

(ii) A value of Foreign Key (FK) in a tuple (t) of the current state of $r_1(R_1)$ even occur as the value of a PK for some tuple t_2 in a current state of $r_2(R_2)$ or NULL.

In Recursive Relation - FK exists too.
(i.e, for Referential Integrity, it applies on a Relation itself).

Primary key can have more than one attributes together.

E.g:- (Essn, Pno) → Primary Key
Together.

- Insertion operation can violate all the four types of constraints.
- Deletion operation can violate only Referential Integrity constraint.
- Updating an attribute that is neither a part of PK nor a FK usually causes no problem except for domain constraint.

Modifying a PK is similar to deleting one tuple and inserting another.

While Updating a FK, make sure that the new value refers to an existing tuple or is said to NULL.

Q. 11

a. No constraint is violated.

b. Dnum = 2. There is no tuple in a Department Relation with Dnumber = 2 exists.

Soln 1. Changing the Dnum to either 5, 4 or 1. or set to NULL.

2. Inserting a new tuple of Dnum = 2 in Department Relation

c. Dnumber = 4 is already exists in the Department Relation

∴ It violates the Key constraint.

Soln:- Just change the Dnumber to 5, 4 or 1.

Mgr_ssn is violating the R.I constraint

Soln:-

(i) Insert new employee who have that Mgr_ssn.

(ii) Assigned an existing supervisor.

R I → See the primary key, comes from or parent class of that relation.

d. Pno which is a Primary key is given NULL which violates E.I constraint.

Also, violates R.I constraint
Essn doesn't exist.

e. No constraint is violated.

f. If we delete child tuple, nothing happens we can delete, it doesn't violate any constraint but if we delete parent, it can be or can not be.

No constraint violated as we can delete from WORK_S_ON (child) but not from EMPLOYEE (PARENT).

g. As we are deleting from PARENT, so it violates RI constraint.
WORKS_ON, EMPLOYEE, DEPENDENT, DEPARTMENT all tables will be affected.

h. Violates RI

i. No violation.

j. R-I violating
What Super-Ssn we are modifying doesn't exist in the table

Soln :- (i) Add new tuple
(ii) Update to NULL.

(k) No violation.

Chapter-4 → SQL

SQL (Structured Query Language)

Create Table Command

It is used to specify a new relation by giving it a name and specifying attribute value.

Syntax:- CREATE Table < TABLE NAME >

```
CREATE TABLE DEPARTMENT
( Dname    VARCHAR(15)    NOT NULL,
  Dnumber  INT            NOT NULL,
  PRIMARY KEY (Dnumber));
```

[Semicolon is used for every statement termination.]

```
CREATE TABLE EMPLOYEE
( SSN    CHAR(10)    NOT NULL,
  NAME   VARCHAR(15) NOT NULL,
  DNo    INT,
  PRIMARY KEY (SSN),
  FOREIGN KEY (DNo) REFERENCES DEPARTMENT (Dnumber))
```

Domains of Attributes -

1. Numeric
   - INTEGER (INT)
   - FLOAT
   - DECIMAL(i, j)
     - i = total no. of digits
     - j = no. of digits after dot (.).

2. Characters
   - CHAR(n) → Allocates the size as given if char(5) means JOHN - space 5 will be allocated
   - VARCHAR(n) → In varchar, char(5) JOHN, no extra space will be allocated.
   - CLOB → (Character Large object)

   It is used to specify columns that have larger text value. such as documents. E.g:- worksheet etc.

3. Bit String
   - BIT(N)
   - BIT VARYING(N)
   - BLOB (Binary Large object Files) → E.g- image, etc. → It is used to specify columns that have large binary values. E.g.- images.

4. Booleans — TRUE / FALSE / NULL

5. Date Type

It has 10 positions & its components are - Year, Month and Date

Format - YYYY - MM - DD (hyphen(-) counted)

To create our own Domain, we will use the command - CREATE <DOMAIN NAME>

[Domain / Datatype - Interchangable.]

```
CREATE DOMAIN SSN-TYPE AS CHAR(9);
```

Specifying constraints in SQL

1. NOT NUL

If NULL is not permitted for a particular attribute.

2. DEFAULT VALUE

It is included in any new tuple if an explicit value is not provided for that value (attribute)

windows → Apache, PHP → WAMP Server (open source)

Linux LAMP → Mysql

Mac MAMP

E.g.:-

```
CREATE TABLE EMPLOYEE
( DNo INT NOT NULL DEFAULT '1',
```

3. CHECK

To restrict an attribute using the CHECK constraint/clause.

E.g.:- Suppose that Dnumber (Department Number) are restrict to integer numbers b/w 1 and 20.

DNumber is the Department Table

```
DEPARTMENT
Dnumber INT NOTNULL CHECK(Dnumber>0
                          AND Dnumber<21);
```

NOTE: We can club the check constraint with the CREATE DOMAIN command.

E.g.:-

```
CREATE DOMAIN D_NUM AS INT
CHECK (D_NUM > 0 AND D_NUM < 21);
```

2023/03/03 21:48

Date: 19/1/23
Page:

Specify Primary Key Constraint

1. If a PK has one attribute, it can be specified as -

In CREATE table command of Department Primary Key can be specified as -

Dnumber INT NOT NULL PRIMARY KEY;

2. If the P.K consist of two or more attributes, then PK can be specified as

Dept location

DNo	Dlocation

```
CREATE TABLE DEPTLOCATION
(DNO INT NOT NULL,
Dlocation VARCHAR NOT NULL,
PRIMARY KEY(Dno, Dlocation),
~~FOREIGN KEY (Dno)~~ ber ),
FOREIGN KEY (Dno) REFERENCES DEPARTMENT(Dnumber));
```

Date:
Page:

To Specify a Secondary Key

The UNIQUE clause ~~spec~~ is used to specify the Secondary Key

Dname	Dnumber

```
CREATE TABLE DEPARTMENT
(Dname VARCHAR(20) NOTNULL UNIQUE,
Dnumber INT NOTNULL PRIMARYKEY,
```

Q Changes are made into the PARENT TABLE, what will be the effect in the CHILD TABLE?

THEME

ThemeID	ThemeName
1	T1
2	T2
3	T3

USER

UserID	UserName	ThemeID
11	ABC	1
22	PQR	3
33	ABC	2

On Delete

1. SET NULL
2. SET DEFAULT
3. CASCADE

① On Delete SET NULL

→ ~~Foreign~~ In User Table (i.e., CREATE TABLE USER
FOREIGN KEY (ThemeID) REFERENCES
THEME (ThemeID) on DELETE SETNULL

Delete Command

DELETE FROM THEME WHERE THEMEID = 2;

Now after deletion in User

UserID	UserName	ThemeID
11	ABC	1
22	PQR	3
33	ABC	NULL

② ON Delete SET DEFAULT

In THEME TABLE

THEMID INT NOTNULL DEFAULT 5,

IN USER TABLE
FOREIGNKEY (ThemID) REFERENCES THEME (ThemeID)
on DELETE SET DEFAULT;

DELETE FROM THEME WHERE THEMID = 2;

Changes

UserId	UserName	ThemeID
11	ABC	1
22	PQR	3
33	ABC	5

③ On Delete - CASCADE

In User Table
FOREIGN KEY (Theme ID) REFERENCES THEME
(ThemeID) on DELETE CASCADE,

Del Command :- DELETE FROM THEME WHERE
THEME ID = 2;

After Deletion →

UserID	UserName	ThemeID
11	ABC	1
22	PQR	3

PTO

Updation

On Update

① SET NULL ② SET DEFAULT ③ CASCADE

~~THEME~~ Theme

ThemeID	ThemeName
2	T1
4	T2
9	T3
10	T11

User

UserID	UserName	ThemeID
11	ABC	2
22	PQR	4
33	PQR	9

① On Update - SET NULL

In User Table

FOREIGN KEY (ThemeID) REFERENCES THEME (ThemeID) on UPDATE SET NULL,

Update Command -

```
UPDATE THEME
SET THEMEID = 3
WHERE THEMID = 4;
```

After Updation - User

UserID	UserName	ThemeID
11	ABC	2
22	PQR	NULL
33	PQR	9

After Updation Theme

ThemeID	ThemeName
2	T1
3	T2
9	T3
10	T11

② On Update - SET DEFAULT

Note:
- SETNULL (ON DELETE OR ON UPDATE) and
- SET DEFAULT (ON DELETE OR OR ON UPDATE) has the same effect as far as the child is concerned.

- ~~On UP~~

③ ON Update - CASCADE.

After applying Update command

11	ABC	2
22	PQR	3
33	PQR	9

INSERT COMMAND

Syntax:- INSERT INTO <TABLE NAME> VALUES( );

E.g.:- INSERT INTO DEPARTMENT VALUES(5, 'HR'); } For insertion of 1 tuple.

INSERT INTO DEPARTMENT VALUES (5, 'HR'), (6, 'FINANCE'); } For insertion of 2 tuples in same command.

DELETE COMMAND.

Syntax:- DELETE FROM <TABLENAME> WHERE <condition>
(where clause is optional).

E.g.:- DELETE FROM DEPARTMENT WHERE Dname = 'FINANCE';

NOTE: If we don't put where clause, all the tuples will be deleted, but the structure of the table remains.

Table will be deleted from the 'DROP' command.

DELETE, UPDATE is a DML (Data Manipulation Language) commands.

CREATE, DROP are the DDL (Data Definition Language) commands.

Retrieval Queries

NOTE:- In case of writing a query, table schema is given and not the values.

Query No. - 1.

Q Retrieve the birth date and address of employee whose name is John B Smith.

SELECT statement

Date:
Page:

NOTE

First find table name, then attribute & then condition

Syntax:

```
SELECT  attribute list
FROM    table name
WHERE   condition ;
```

Ans

```
SELECT  Bdate, Address
FROM    EMPLOYEE
WHERE   Fname = 'John' AND Minit = 'B'
        AND Lname = 'Smith';
```

Bdate	Address
1965-01-09	731 Fondren, Houston, TX

Q Retrieve the name & address of all employees who work for the Research department.

Ans

```
SELECT  Fname, Address
FROM    EMPLOYEE, DEPARTMENT
WHERE   Dname = 'Research' AND Dno = Dnumber;
```

First, we'll see employee PK and Department FK. } we run in order, Employee table ka PK table like in order.

P.K and F.K relation

Date:
Page:

Q For every project located in New Delhi, list the project number, the controlling department number & the department manager's last name, address.

```
SELECT  Pnumber, Dnum, Lname, address
FROM    EMPLOYEE, DEPARTMENT, PROJECT
WHERE   Plocation = 'New Delhi' AND
        SSN = Mgr-SSN AND Dnumber = Dnum;
```

NOTE:

A missing where clause indicates no condition.

Q Select all employee's ssn.

Ans

```
SELECT  Ssn FROM EMPLOYEE;
```

NOTE:

Estric (*) stands for all attributes.

Q Select all the information of employee?

Ans

```
SELECT  *  FROM EMPLOYEE;
```

2023/03/03 21:49

Q Select all information of Employee where Ssn = '1234.

Ans
```
SELECT *
FROM EMPLOYEE
WHERE Ssn='1234';
```

WHERE clause removes duplicacy.

```
SELECT *
FROM EMPLOYEE, DEPARTMENT.
```
} — Wrong Query

Date: 21/1/23

Ambiguous Attribute Name

Employee

SSN	Name	Dno

Department

Dno	Dname

Q Name and address of employees who works in a Research department.

Ans
```
Select Name
FROM Employee, Department
WHERE Dname = 'Research' AND Dno = Dno.
    AND Employee.Dno = Department.Dno
```

In sequel (SQL), the same name can be used for two or more attributes as long as these attributes belongs to different table. We must qualify the attribute name with the relation name to prevent ambiguity, this can be done by using relation name first followed by an attribute name separated by a dot (.)

Creating an object of Relation

Using ~~as~~ AS constraint.

```
Select Name
FROM Employee AS E , Department AS D
WHERE Dname = Research AND E.Dno = D.Dno
```

OR

```
FROM Employee E , Department D
```

It's not important to use AS constraint, a space will also work.

NOTE:

- SQL does not automatically eliminates the duplicate.
  SQL gives the result in a multiset (with duplicate values).

Q Retrieve the salary of every Employee.

```
SELECT Salary
FROM Employee ;
```

Salary
1000
2000
1000
2000

NOTE:- To remove the duplicates, use DISTINCT.

Q Retrieve the unique salary of employees.

```
SELECT DISTINCT Salary
FROM Employee ;
```

Salary
1000
2000.

NOTE:- SQL supports Union except or intersect (−), duplicate tuples are ~~do~~ eliminated from the result.

Q Make the list of all project numbers for projects that involve an employee whose last name is Smith either as a worker or as a manager of the department ~~as~~ that controls the project.

Ans

```
SELECT DISTINCT Pno
FROM Employee , Works_On
WHERE Lname = 'Smith' AND Ssn = Essn.
UNION
SELECT Pnumber
FROM Employee , Department , Project
WHERE Lname = 'Smith' AND Dnumber = Dnum AND
                          SSN = MGRSSN.
```

In case of SET Theory, the results are multisets if we use UNION ALL, INTERSECT ALL, MINUS ALL.

LIKE Comparison operator (used for String Pattern matching).

% replaces an arbitary number of 0 or more characters.

_ (underscore) replaces a single character.

Q Find all employees who are born during the 1950's.

Ans
```
SELECT Fname
FROM Employee
WHERE Bdate LIKE 195-------.
```
or

195 %.

When the value is not fixed, we use %, if it is fixed then (_) can be used.

Join two tables using JOIN keyword

Q Retrieve the name & address of all employees who works in Research department.

```
SELECT Fname, Address
FROM (EMPLOYEE JOIN DEPARTMENT ON Dno = Dnumber)
WHERE Dname = 'Research'.
```

Ans

All Attribs of Employee

NATURAL JOIN

Ssn	Fname	Address	Dno	Dnumber	Dname
-	-	-	1	1	---
-	-	-	2	2	--

After Natural Join,

Ssn	Fname	Address	Dno	Dname

Now, we have to delete the same column which is giving the same value, i.e, to remove redundant columns, or to reduce space or time complexity, we use Natural Join.

- In Natural Join, no join condition is specified.

- The name of the join attribute needs to be same in both the relation.

The AS constraint can be used to rename the relation or all its attributes

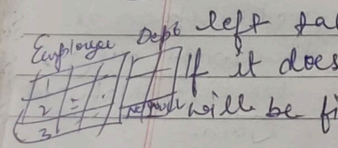
Ans SELECT Fname, Address
FROM (EMPLOYEE NATURAL JOIN
DEPARTMENT AS DEPT (~~Dnumber~~ Dno, Dname))
WHERE Dname = 'Research';

of Natural Join

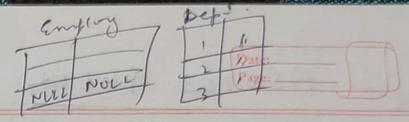
POINTS

① In SQL, joined tables includes only pairs of tuples that match the JOIN condition. also known as INNER JOIN OR JOIN.

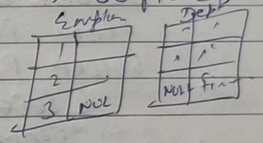
② LEFT OUTER JOIN = Every tuple in the left table will appear in the result. If it does not have a matching tuple, it will be filled with NULL values.



③ Right OUTER JOIN = Every tuple in the right table will appear in the result. If it does not have a matching tuple, it will be filled with NULL values.



④ FULL OUTER JOIN - Every tuple in the left and as well as right table will appear in the result and if does not have a matching tuple, will be filled with NULL values.



IN Clause

It compares the value 'v' with a set or multiset of values 'V' and evaluates to true if 'v' is one of the element in 'V'

Q Retrieve the SSN of all employees who works on the project number 1, 2 & 3.

Ans SELECT Essn
FROM WORKS_ON
WHERE Pno IN (1, 2, 3);

If we don't want to use IN Clause, we can also use DISTINCT Keyword.

Select DISTINCT Essn From Employee
WHERE Pno = '1' or Pno = '2' or Pno = '3';

NESTED QUERRY

Nested Queries are complete select from where block, within the WHERE clause of another querry

Q Make the list of all project - - - - - that controls the project [:: Previous Ques]

```
SELECT DISTINCT Pnumber → (1,2,3)
FROM PROJECT                  ↓
WHERE Pnumber IN            final
    ( SELECT Pno (1,2)
      FROM EMPLOYEE, WORKS_ON
      WHERE Lname='Smith' AND Ssn = Essn)
                                  because of OR
    OR
    Pnumber IN
    ( SELECT Pnumber (1,2,3)
      FROM Employee, Department, Project
      WHERE Lname='Smith' AND Dnumber = Dnum
            AND Ssn = Mgr_Ssn')
```

Date: 28/1/23

Q Select the Essn of all Employees who works the same (project, has) combinations on some project that employee John Smith works on.
Ssn of John is 12345 } Given

```
SELECT Distinct Essn
FROM WORKS_ON
WHERE (Pno, Hours) IN ( SELECT Pno, Hours
                        FROM WORKS_ON
                        WHERE Essn = 12345);
```

Comparision Operators

~~Any~~ With the combination of ANY / SOME, the available comparison operators are
>, <, >=, <=, <> (not =)

NOTE: = ANY OR (= SOME) is equal to IN operator

Correlated Nested Querry

Q Retrieve the name of each employee who has a dependent with same name and same Date of Birth as an employee.

Ans

Simple Query

```
SELECT Fname, Bdate
FROM Employee as E, Department as D
WHERE Fname = Dependent_name AND
      E.Bdate = D.Bdate AND Ssn=Essn;
```

Whenever a condition in a where clause of a NESTED QUERY REFERENCES some attribute of a Relation declared in the outer Queries, the two queries are said to be Co-related nested query.

with Co-Related Query

Ans

```
SELECT Fname
FROM EMPLOYEE AS E
WHERE Ssn IN (SELECT Essn
              FROM DEPENDENT AS D
              WHERE E.Fname = Dependent_name
              AND E.Bdate = D.Bdate);
```

There are three steps in designing of a database :-

I Conceptual Scheme

Once the requirement from the client have been collected & analysed, the next step is to create conceptual design. Conceptual design consists of an ERD model

II Logical Design

The ERD model is mapped into tables.

III Physical Design

The internal storage of the tables.

ERD Model

Entity, Relationship, Diagram

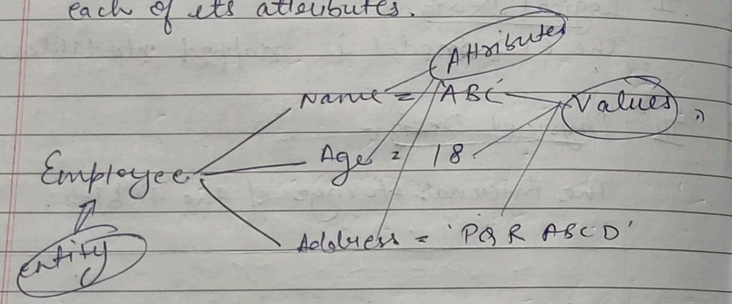
Entity and its Attribute

Entity - An Entity is a thing in the real world

with an independent existence.
An entity may be an object with physical existence (i.e, example car, person) or it may be an object with conceptual existence (e.g- company, job).

Each entity has its own attributes (the particular properties that describe it).

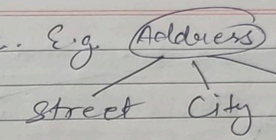
A particular entity will have values for each of its attributes.



Types of Attributes occurs in the ER model.

(1) Composite vs Simple (atomic) Attributes

Composite attributes can be divided into smaller sub parts. Each sub-part represents a simple attribute. E.g. Address



Simple attributes are those attributes that are not divisible.
Atomic. Eg- Age, etc.

(2) Single Valued vs Multi-valued Attribute

Attributes that have a single value for a particular entity is called single-valued Attribute. E.g:- Age, etc.

Attributes can have set of values for the same entity is called multi-value attribute.
Eg:- different people can have different value
Car colour
(1) Blue
(2) blue, black, etc.

(3) Stored vs Derived Attribute

If two or more attributes are related (i.e., age & DOB), then age attribute is called the derived attribute & ~~stored~~ DOB attribute is called the stored attribute.

(4) NULL values

- Not applicable (College_Degree)
  (If a 12th passed is an employee in the company then College_Degree is considered to be NULL)
- attribute value exists but is missing (Height)
- unknown (Home_Phone)

Complex Attributes

We can represent an arbitarary nesting by grouping components of a composite attributes between parenthesis & separating the components with commas and by displaying multi-valued attributes b/w braces.

Q Represent a Complex Attribute if a person can have more than one residence and each residence can have a single address and multiple phone numbers.

Eg. Phone no.
- Area code
- phone no.

Address
- Street address
- city
- state
- pin code

{Address_Phone({Phone (areacode, Phoneno.)},
Address (Street_address, city, State, Pin-code)}

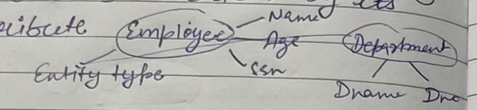
Date: 2/2/23
Page:

ERD Model

Entity

Entity type

It defines the collection of entities that have the common attributes. Each entity type in a database is described by its name and attribute.



Entity

The entity shares the same attribute but each entity have different value for each of the attribute.

11	ABC	18	→ entity
22	PQR	19	→ another entity

Entity set

It is a collection of all entities at any given point

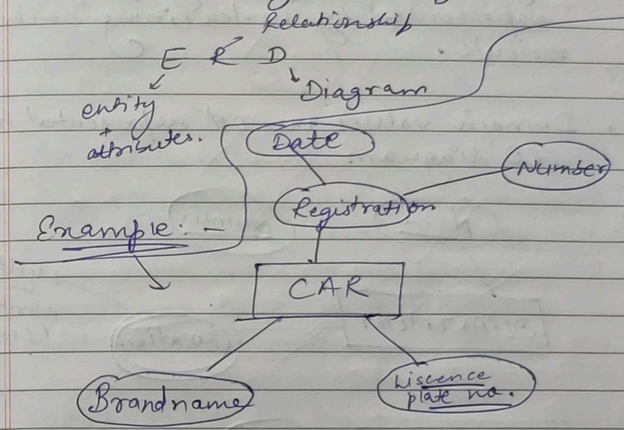
11	ABC	18	
22	PQR	19	→ Entity set.
--	--	--	

- An entity type is represented in the ERD by a rectangular box enclosing the entity type name.

Attribute names are enclosed in ovals & attached to the entity type by using a straight line

Date:
Page:

- Composite attributes are attached to their components using a straight line.



- An entity type has one or more attribute whose values are distinct for each individual entity in the entity set. Such an attributes are called Key Attributes.

Key Attributes are represented by underlining that attribute. E.g- Liscence plate no.

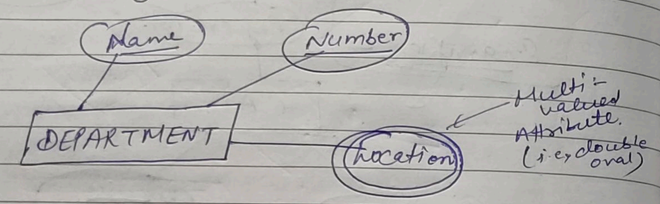
It is only a key attribute not an primary key.

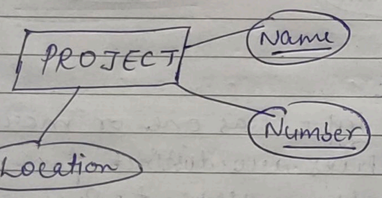
2023/03/03 21:50

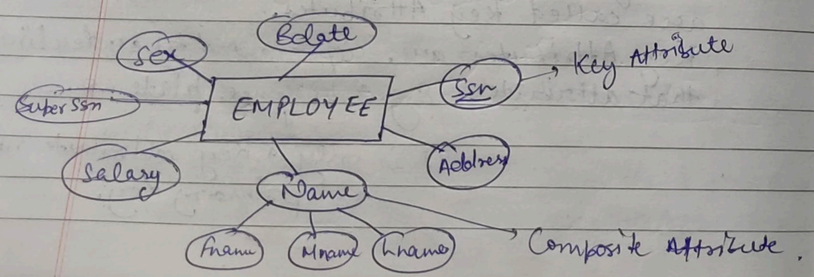
- An entity type can have more than one key attribute. Each key attribute is represented by underlining each key attribute.

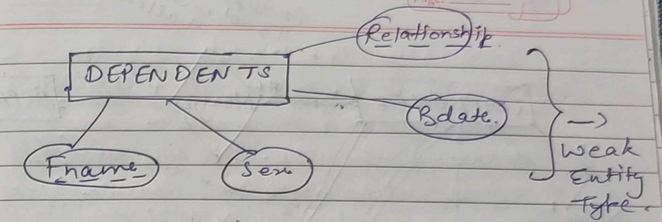
- Domain values are not represented in the ERD diagram

E.g:-









- An entity type that doesn't have a key attribute is called a <u>Weak Entity Type</u>. E.g. Dependents.

- An entity type that does have a key attribute is called a <u>Strong Entity Type</u>.
  E.g:- Department, Project, Employee.

- Weak Entity has partial key attribute.
  i.e, Fname, Relationship → take any one of these
  Partial key attribute
  ~~Weak Entity~~ is represented by a dash (---) line
  Weak Entity is represented by double line rectangular box.
  A weak entity has an identifying relationship with its owner.

Relationship | & its is represented by double line diamond shaped box.

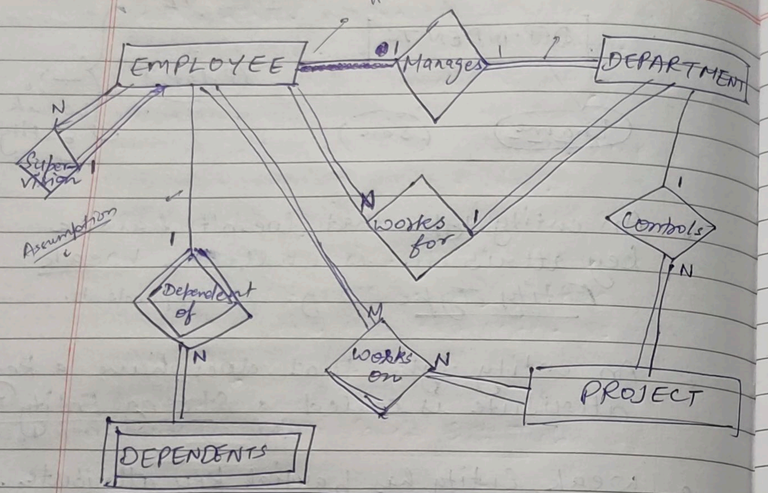
Whenever an attribute of one entity type refers to another entity type, some relationship exists.

NOTE:- In an ERD, relationships are displayed as diamond shaped box (◇) which are connected by the straight line to the rectangular boxes representing the participating entity type.

Convention
~~Right~~ Left → Right
Top → Bottom

Entity Type - Noun
Relationship Type - verb

Assumption
It's not necessary that every employee is allotted to a project department.



Degree of a Relationship (n)

It is the number of participating entity type. E.g:- Manages → Degree is 2.

A relationship type of degree 2 is binary, & relationship for degree 3 is ternary relationship.

Date: 4/2/23
Page:

Constraints on Binary Relationship Type

Cardinality Ratio - is of three types.

① 1:1 ② 1:N or N:1 ③ M:N

M : M, N : N - doesn't exists.

∴ N & M - we are taking two variables having same meaning

- N indicates that there is no maximum number. M also indicates the same → no max. number.
- Cardinality Ratio for the binary relationship specifies the maximum number of relationship instances that an entity can participate in.

For finding cardinality ratio, take one instance and then write the number ~~to~~ at the opposite of the diamond shape relationship. & if assumption is there, do write it. Very imp to write assumptions.

Participation Constraints on Binary Relationship Type

Two types -

① Total Participation
Representation → (|| - Double line)

② Partial Participation.
(| - Single line)

2023/03/03 21:50

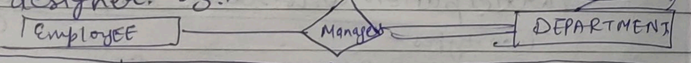
Participation constraints specifies whether the existence of an entity depends on its being related to another entity via the relationship type.
T.P is represented by double line & P.P is represented by single line.

NOTE: Whenever an instance is not clear (means it may be yes may be no), then you have to write the assumptions.

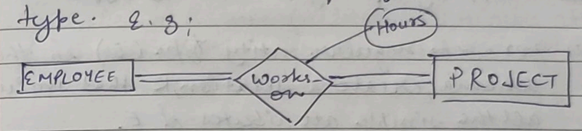
Attributes of a Relationship Type

Relationship types can also have attributes just like the entity type.

In both 1:1 & 1:N relationship type, the decision where to place the relationship attribute (as a relationship type attribute or as an attribute of entity type) is determined subjectively by schema designer. E.g:-



For M:N relationship type, some attributes may be determined by the combination of participating entity type & not by any single entity type. Such attributes must be specified as a relationship type. E.g:



This hrs attributes will be of relationship type.

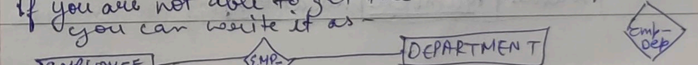
But not necessary

Start date attribute can be of Employee, Manges, Department. but will give it to department or manages because in Employee if we give it to employee, there will be many NULL columns which will take more space because department may not get allot to every employee.

NOTE: The Cardinality and Participation constraints together is called Structural Constraints.

NOTE: If changing the convention, then write arrows.

If you are not able to get the relationship name you can write it as -



Date: 8/2/23
Page:

Mapping of ERD to Relations

Step 1 → Mapping of Regular (means strong Entity) Entity Type

(i) For each regular entity type (E) in the ER schema, create a relation, R that includes all the simple attributes of E.

(ii) Choose one of the Key attribute of E as the primary Key for R. If the chosen key attribute is composite, the set of simple attributes that forms it will together form the P.K of R.

Step 1 → Mapping of Regular Entity Type.

- EMPLOYEE

Ssn	Fname	Minit	Lname	Bdate	Address	Salary	Sex

- DEPARTMENT

Name	Number

- PROJECT

Name	Number	location

Date:
Page:

Step 2 → Mapping of Weak (W) Entity Type

(i) For each weak entity type (W) in the ER schema with owner entity type E, create a relation R and include all simple attributes (components of composite attributes) of W as attributes of R.

(ii) Include as foreign key (F.K) attributes of R, the primary key attribute of the owner entity type.

(iii) The P.K of R is the combination of PK of its owner and the partial key of the weak entity type.

Step 2 → Mapping of Weak Entity Type.

DEPENDENT

Name	Sex	Birth_date	Relationship	Ssn

NOTE: If there is a weak entity (W_2) whose owner is also weak entity type (W_1), then W_1 should be mapped before W_2 to determine its PK first.

2023/03/03 21:50

Step3 → Mapping of Binary 1:1 Relationship Type

(i) Identify the relations S & T that corresponds to entity types participating in R

(ii) In order to map 1:1 relationship type, we have 3 approaches.

- Foreign Key Approach
- Merged Relation Approach
- Cross Reference Approach

Foreign Key Approach

- Choose one of the Relation S and include as a foreign key in S, the primary key of T (the side that has total participation considered it as S). Also, include the simple attribute of Relationship into S.

EMP(T) — PK Ssn

DEP (S) — FK is Ssn of this R

NOTE: If Ssn name attribute is already present, you can rename the attribute as mgr-ssn in Department.

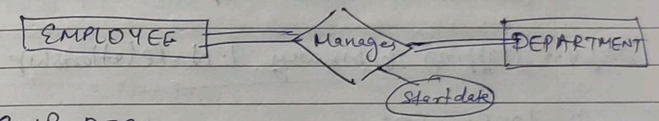
DEPARTMENT

Name	Number	Ssn	Start date

Merged Relation Approach

Merge the two entity type and the relationship into a single relation (this is possible when both sides have total participation).

NOTE: Assuming total participation on both sides.



EMP_DEPT

Ssn	Fname	Minit	Lname	Bdate	Address	Salary	Sex	Name	Number	Start date

Cross Reference Relation

NOTE:- Same as Step 5

Step 4 → Mapping of Binary 1 : N Relationship Type

(i) Identify the relation S that represents the participating entity type at the N side.

Include a foreign key in S, the PK of the relation T that represents the other entity type.

(ii) Include any simple attribute (components of composite attributes) of the 1 : N relationship type as the attribute of S.

Step 4 → Mapping of Binary 1 : N Relationship Type

① WORKS_FOR
1 : N

T — PK DEPARTMENT Number

S — EMPLOYEE

Number → as FK in S

② CONTROLS
1 : N

T — DEPARTMENT PK Number

S — PROJECT

Dnumber → as FK
as Number is already present in Project.

③ SUPERVISION
1 : N

PK EMPLOYEE Ssn → Supervision_ssn → as FK of Employee

④ DEPENDENTS_OF
1 : N

EMPLOYEE Ssn → DEPENDENT

mapping of PK as FK in S
This we have done earlier in Step 2.
So, don't need to repeat it again.

For WORK_FOR & SUPERVISION

EMPLOYEE

Ssn	Bdate	Fname	Minit	Lname	Address	Salary	Sex	Number	Supervision_Ssn

FOR CONTROLS

PROJECT

Name	Number	Location	Dnumber

Step5 → Mapping of Binary M:N Relationship Type

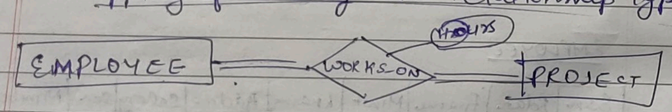
(i) Create a new relation S to represent R.

(ii) Includes as FK attributes in S, the primary key of the relations of the participating entity type.

(iii) Their combinations (the Foreign keys) will form the PK of the newly created relation.

(iv) Include any simple attribute (components of composite attributes) of the relationship type as attribute in the newly created relation

Step5 → Mapping of Binary M:N Relationship Type



WORKS_ON

Ssn	Number	Hours

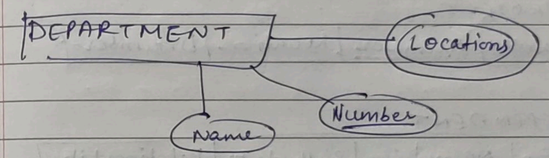
Combination of these two PK will form the PK of works_on.

Step6 → Mapping of Multi-valued attribute

(i) For each multi-valued attribute (A), create a new relation R.

(ii) This relation R will include an attribute corresponding to A + the PK attribute (K) as the FK in R.

(iii) The PK of R is the combination of A & K.



Dept_Location → Newly created table

Location	Number

PK will be the combination.

NOTE:
These 6 tables which is created is sent by mam on group.

Date: _____
Page: _____

The Relation Schemas are – Total-6

① EMPLOYEE

SSn	Fname	Minit	Lname	Bdate	Address	Sex	Salary	Super-SSn	Dnumber

② DEPARTMENT

Name	Dnumber	Mgr-Ssn	Start-date

③ WORKS_ON

Essn	Number	Hours

④ PROJECT

Name	Number	Location	Dnumber

⑤ DEPENDENT

Essn	Name	Sex	Birth-date	Relationship

⑥ DEPT_LOCATIONS

Dnumber	Location

Date: 11/2/23
Page: _____

(min, max) constraints (Alternative of cardinality ratio + participation) ← Structural constraints.

This notation involves associating a pair of integer numbers (min, max) where min values lies b/w 0 & max ($0 \le min \le max$) & max value is greater than 1 ($max \ge 1$).

NOTE: Min means Participation where 0 means partial & 1 means total unless or until specified.
Max represents cardinality.

2023/03/03 21:50

Jis side ki cardinality hogi, usi side likhna hai participation, write this in assumption as (4, N) is not possible there) → In structural constraints.

Assumptions - ① An Employee can have many departments.

② (4, N) - means - 4 employees must be there in each department. (In

Relationship Types of Degree Higher than 2

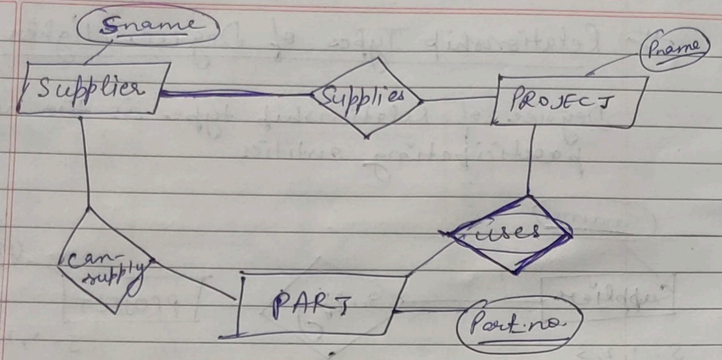
Degree of relationship type is the no. of participating entities

A ternary relationship type represents different information than 3 binary relationship type.

A supplier S1 can supply a part P1 to some project J1.

The binary relationship of the ternary relationship type will be →

P.T.O



Consider the three relationship → can supply, uses & supplies.

- Can_supply
  A supplier S can supply any part P.

- Uses
  A project J can use any part P supplied By any supplier.

- supplies
  A supplier S can supply to any project J.

This relationship does not guarantees an instance S = 1, P = 1 & J = 1.

NOTE: The designer must decide whether to use ternary or binary relationship based on semantics of the question

NOTE: Structural constraint on the ternary relationship type should be (min, max) constraint.

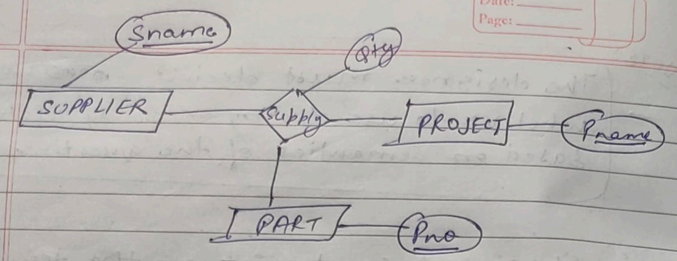
Step 7 → Mapping of Ternary Relationship (N > 2)

(i) For each relationship type where N is greater than 2, create a new relation R.

(ii) Include as FK, the PK of participating relation.

(iii) The combination of the PK (step 2) se jo aayi h will be the the PK of newly created relation.

(iv) Include the simple attribute (components of composite attributes of the relationship).



Step7 → Mapping of Ternary Relationships

SUPPLY

Sname	Pname	Pno	Qty

PK will be the combination of these 3.

EERD Diagrams → Preferable than ERD in paper.
↓
Enhanced Entity Relationship Diagram

An Entity type has sub-types or sub-grouping of its entities that are meaningful and need to be represented explicitly.
E.g:- Employee entity can have sub-groups such as Secretary, Technician & Engineer.
We call the sub-groups as sub-class or sub-type.

An Employee will be the super-class, class or super-type.

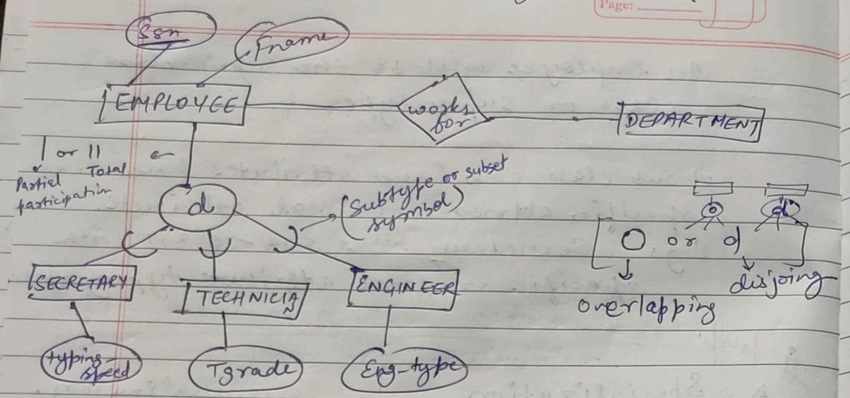
A sub-class can have attributes known as specific attribute or local attribute.
E.g:- Secretary is a sub-class with specific or local attribute typing-speed.

CMA | wtp

Specialization	Generalization.
① Top-down approach	① Bottom-up approach.
② Given a super-class, we will create its sub classes.	② Given a set of sub-classes, we will create a super class.

Similarities-
These two approaches are mathematically equivalent.

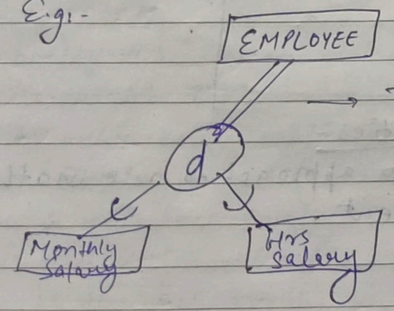
Way of representing a EERD



For overlapping or disjoining, we see sub-classes that all are belongs to that sub-class only or not. If yes, then d. If no, then o.

For total or partial participation - we see through super-class. ~~that all are belongs to~~

E.g.-



→ Total participation.

Every employee will have salary.

Date: 15/2/23

Constraints on Specialization or Generalization.

Two constraints are present -

1. Disjoint 2. Overlapping

Disjoint constraints specifies that the sub-classes of the specialization must be disjoint. This means that an entity can be a member of atmost one of the sub-classes of the specialization. Represented as 'd' in a circle.

Overlapping
The same entity may be a member of more than one sub-classes of the specialization. Represented as 'O' in the circle.

Totalness or Completeness Constraints

1. Total participation 2. Partial participation.

Four types of constraints are possible in case of EERD.

1. Disjoint Total
2. Disjoint Partial
3. Overlapping Total
4. Overlapping Partial

NOTE: In Generalization - Bottom-up approach. Always total participation.

Date: 18/2/23

ERD to tables/relation - Step 1 to 7
EERD to tables - Step 1 to 8

Step 8 → Convert each specialization with m subclasses $\{S_1, S_2, \ldots, S_m\}$ and super class with attributes $\{K, a_1, a_2, \ldots a_n\}$ where K stands for Primary Key using one of the following options.

Step-8
- 8A
- 8B
- 8C
- 8D

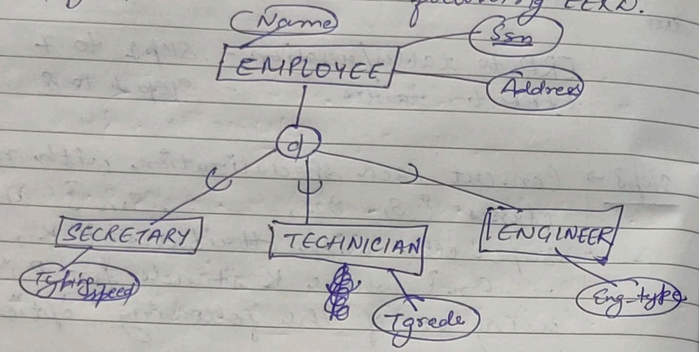
8A → Safest option (if not given the method in the ques, apply 8A)

Option - 8A multiple relation, super class and subclasses.

1. Create a relation for superclass with attributes $\{a_1, a_2, \ldots a_n\}$ + a primary key.

2. Create a relation S_i for each of the subclasses with attributes = attributes of S_i $\cup \{K\}$. The PK of S_i is K.

Q Apply option-8A on the following EERD.



EMPLOYEE

Ssn	Name	Address

SECRETARY

Ssn	Typing_speed

TECHNICIAN

Ssn	Tgrade

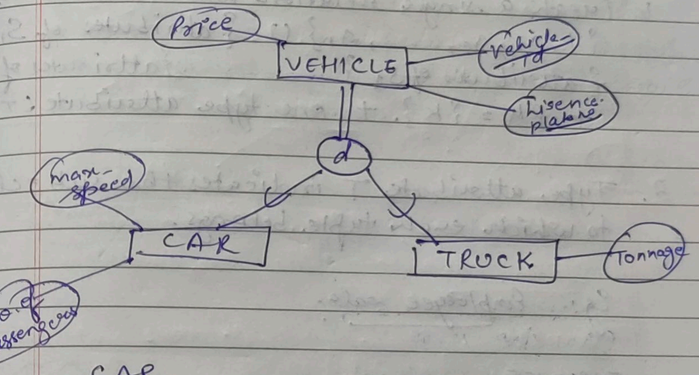
ENGINEER

Ssn	Eng_type

NOTE: 8A option works for any specialization.

Option 8B Multiple relations, sub-class relation only.

1. Create a relation for each of the sub-class (S_i) with attributes = { attributes of S_i } ∪ { $k, a_1, a_2, \ldots, a_n$ } with PK = {k}



CAR

Vehicle_id	Licence plate no.	Price	max speed	no. of passengers

TRUCK

Vehicle_id	Licence_plate no.	Price	Tonnage

NOTE: Choose PK ya toh vehicle_id or licence_plate no. any one only

NOTE:

This option works only for disjoint total.

Option 8C - Single relation with one type attribute

1. Create a single relation with attributes $\{K, a_1, a_2, \ldots, a_n\} \cup \{$attributes of $S_1\} \cup \{$attributes of $S_2\} \ldots \{$attributes of $S_m\}$ + PK = $\{K\}$. + one type attribute (T)

2. Type attribute T indicates the sub-class to which each tuple belongs.

Eg:- Employee wala.

EMPLOYEE

Name	Ssn	Address	Typing-speed	Grade	Eng-type	Job-type

NOTE:

This option works only for the sub-classes that are disjoint.

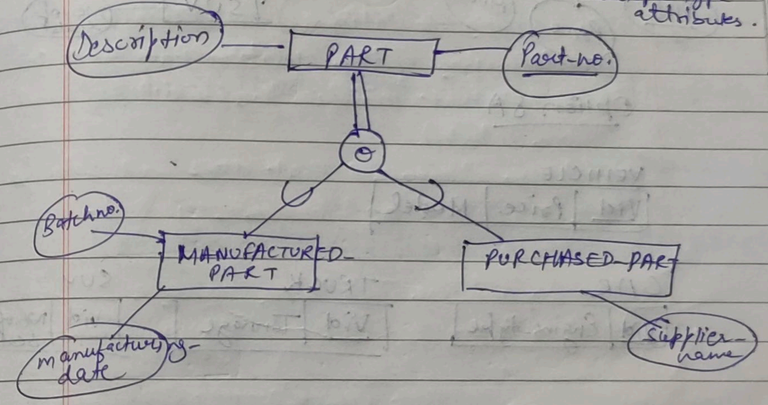
Disadvantage -

If a sub-class have many specific attributes then, it has the potential to generate many NULL values.

Option 8D - Single relation with multiple type attribute.

1. Create a single relation with attributes $\{K, a_1, a_2, \ldots a_n\} \cup \{$attributes of $S_1\} \cup \{$attributes of $S_2\} \ldots \{$attributes of $S_m\}$ + PK $\{K\}$ + $\{t_1, t_2 \ldots t_n\}$

↳ multiple type attributes.



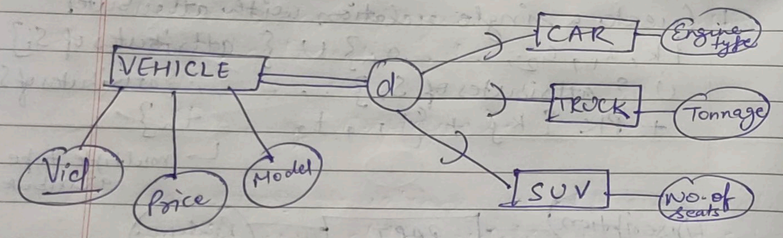
PART

Partno.	Description	Batchno.	manufacturing-date	Supplier-name

NOTE:

This option only works for overlapping

Q. Apply Option 8A, 8B, 8C and 8D to the following EERD.



Option 8A

VEHICLE

Vid	Price	Model

CAR

Vid	Engine-type

TRUCK

Vid	Tonnage

SUV

Vid	No. of seats

Option 8B

CAR

Vid	Price	Model	Engine-type

TRUCK

Vid	Price	Model	Tonnage

SUV

Vid	Price	Model	No. of seats

Option 8C

VEHICLE

Vid	Price	Model	Engine-type	Tonnage	No. of seats	Type

wrong ques

Option 8D (Vaise Not Applicable) but given in ques.

VEHICLE

Vid	Price	Model	Engine-type	Tonnage	No. of seats.

Relational Algebra – new Chapter

Operations

1. Select Operation

1. It is used to choose a subset of the tuples from a relation that satisfies a

Selection condition.

2. It is a filter that keeps only those tuples that satisfy a qualifying condition.

Prototype → $\sigma_{\text{selection condition}}(R)$

Q Retrieve the employees who are working in Department no. 4.

Ans $\sigma_{Dno=4}(\text{Employee})$

3. Sigma (σ) operations creates a horizontal partitioning of the relation into two sets. One set that satisfies the condition and hence selected.
and the other set that does not satisfy the condition and hence rejected.

Q Retrieve all the employees whose salary is greater than 30,000.

Ans $\sigma_{Salary>30,000}(\text{Employee})$

NOTE: Degree of a Relation is the no. of attributes in the relation

After applying SELECT operation, degree of Relation is same as R

~~2. Project Operation~~

NOTE: In this case - 10 is the degree

NOTE: π can't have condition in it. for condition, we have to use σ.

2. Project Operation

1. It selects specific column from the table and discards the other column.

2. It creates the vertical partitioning of the relation.

Prototype → $\pi_{\text{attribute list}}(R)$

Q Retrieve first name of all the employees.

Ans $\pi_{Fname}(\text{Employee})$

Q Retrieve the first name of all employees working in department number 4.

Ans $\pi_{Fname}(\sigma_{Dno=4}(\text{Employee}))$ ← Inline expressions

3. The degree of the relation is equal to the no. of attributes in π.

NOTE: π is the duplicate eliminator operator.

3. Rename Operation

1. we can write the operation as the single relational algebra expression by nesting the operation or we can apply one operation at a time and stores the intermediate results

Q Retrieve the first name of all employees working in department number 4.

$R1 \longleftarrow \sigma_{Dno=4}(Employee)$

$RESULT \longleftarrow \pi_{Fname}(R1)$

To Rename the attribute

$RESULT(FirstName) \longleftarrow \pi_{Fname}(R1)$.

$RESULT(FirstName, Lname) \longleftarrow \pi_{Fname, Lname}(R1)$

(If you just want to rename Fname not Lname, but u have to write for the both.)

Relational Algebra

Date: 25/2/23

4. Union, Intersection and Minus (Set difference/) Operation

Union Compatible or Type Compatible

Two Relations $R(A_1, A_2, \ldots A_n)$ and $S(B_1, B_2, \ldots B_n)$ are said to be union compatible if -

(i) They have the same degree (i.e, no. of columns).

(ii) And the corresponding pair of attributes have the same domain.

Q Retrieve the Ssn of all employees who either works in department no. 5 or directly supervise an employee who works in department no. 5.

Ans

$DEPT5EMP \longleftarrow \sigma_{Dno=5}(Employee)$

$R2 \longleftarrow \pi_{super_ssn}(DEPT5EMP)$

$R1 \longleftarrow \pi_{Ssn}(DEPT5EMP)$

$RESULT \longleftarrow R1 \cup R2$

2023/03/03 21:51

NOTE: Name of the column depends on which we have written first, if we write R1 ∪ R2, the Ssn will be the name of the new column. & if we write R2 ∪ R1, then ~~Sup~~ Super-ssn will be the name of the column.

NOTE: SET operations removes the duplicate tuples.

Using ~~all~~ All clause (union ALL, intersect ALL, minus ALL), the duplicate tuples will appear in the result.

RESULT ← R1 union ALL R2

5. Cartesian Product or Cross Product Operation

Denoted by a 'X' symbol. (Binary operation)

NOTE: The two relations do not have to be union compatible.

Q Retrieve the list of names of each female employees dependent.

Ans R1 ← $\sigma_{Sex='F'}$ (EMPLOYEE)

This we have write just for refinement. Not required necessarily. We'll take just those columns which is necessary not all.

R ← $\pi_{Fname, Ssn}$ (R1)

For cross product, we want two relations, suppose R × S

R is found. Now, for S.

R2 ← $\pi_{Essn, Dependent_name}$ (DEPENDENT)

Now, S is also found.

TEMP ← R × R2 → This will give all the tuples which is not required.

RESULT ← $\sigma_{Ssn=Essn}$ (TEMP)

Drawback - Taking more space which is not needed

6. JOIN Operation (Binary Operation)

It is denoted by ⋈.

Prototype: R ⋈$_{Join\ Condition}$ S

Q Retrieve the name of the manager of each department.

R1 ← EMPLOYEE ⋈$_{Ssn=Mgr_ssn}$ DEPARTMENT

RESULT ← $R \bowtie_{Ssn=Essn} R2$ } Previous ques using JOIN condition

SO, don't have to use TEMP wala part. in previous ques. — This way it will not take more space. if we use JOIN condition in [P.Q.]

NOTE: In JOIN operation, the list of operators that are acceptible are $=, >=, <=, >, <, \neq$

These types of operations are called 'theta' JOIN.

When the operation is using = sign, it is called 'equi' join.

Otherwise, all operations are called 'θ' join. except 'equi' join.

Q Combine each project tuple with the department tuple.

Natural join

rho

RESULT ← PROJECT * $\rho_{(Dname, Dnum, Mgr_ssn, Mgr_start_date)}$ DEPARTMENT

Pname	Pnumber	Plocation	Dnum	Dname	Mgr_ssn	Mgr_start date

→ The new Relation will look like this. using Natural Join.

We have made Dnumber as Dnum using operation ρ then using Natural Join, eliminates the duplicate column Dnum.