(SG)

21/08/24

DBMS

(interrelated)

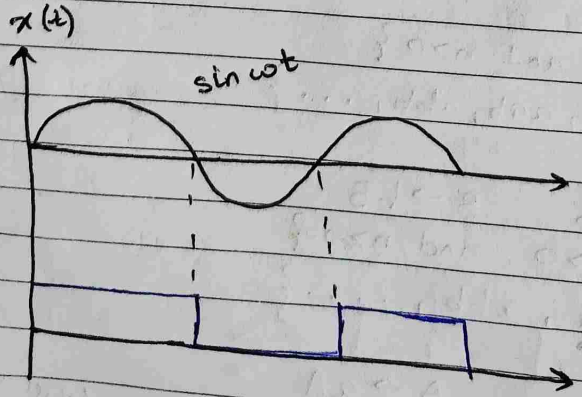
- collection of data
- set of programs to access those data
- provide an environment that is convenient and efficient to access those data from the database and to store the data in the storage.

Drawbacks of file Management Process :

1. Data redundancy and inconsistency
2. Difficulty in accessing data
3. Data isolation
4. Integretiby problem (including "constraints")
5. Atomicity problem
6. Concurrent access anamolies
7. Security problems

"Whenever a transaction is being performed, either it is performed in its entirety or not at all."

performed



u(sin ωt) = 1 for sin ωt ≥ 0

u(sin ωt) = 0 for sin ωt < 0

27 MARCH Wednesday

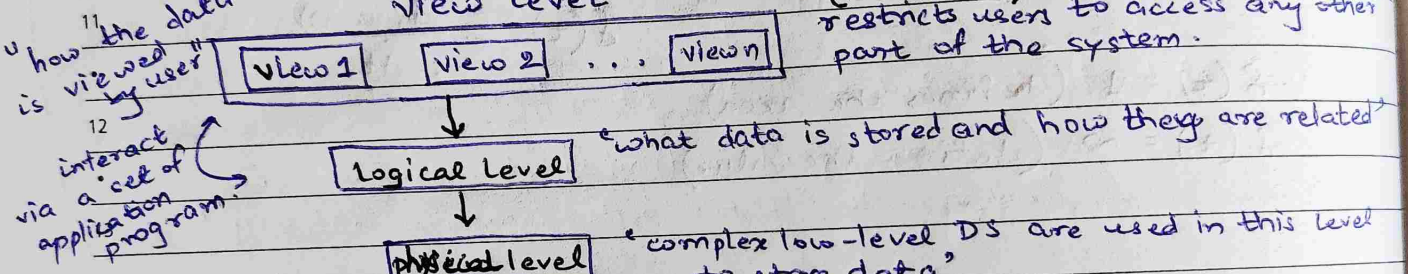
87/279 28/08/24 (SG)

→ View of Data :

- Provide an abstraction (abstract view) to the user.
  ↳ hides details from user

3 abstractions –

- Physical Level
- Logical Level
- View Level



→ Instances and Schemas :

Collection of information stored at a particular time instance is known as instance of a database.

- Schema is the "overall design of the database".
  - Physical Level Schema
  - Logical Level Schema } Single schema
  - View Level Schema (refered to as sub-schemas)

→ Data Independance : [Physical level, Logical level]

- ability to modify schema at a particular level without affecting the next higher level schema

*At logical level when schema is changed, not only application programs needs to be changed but as well as the data part. This is the reason for the difficulty in changing schema at Logical Level.

People are lonely because they build walls instead of bridges. – Joseph Newton

→ **Data Models :**

- underlying structure of the database.
- consists of no. of conceptual tools used to describe -
  (i) data (ii) relationship among data
  (iii) semantics (iv) consistency constraints over data

Different data models -

(i) Object based logical model
  (a) E-R model (b) object oriented model
  (c) the semantic data model
  (d) the functional data model
(ii) Record based logical model
  (a) relational model
  (b) network model
  (c) hierarchical data model
(iii) Physical model

02/09/24

- E-R model : describes the system in terms of objects (entities)
- Network model : records are described by links (pointer) given by arbitary graph
- hierarchical data model : described by trees (hierarchy).

* Physical model → Unifying model
                 → Frame memory model

29 MARCH Friday

89/277

→ **Database Languages :**

(i) Specify the schema
(ii) Queries and Updates

- **Data Definition Language : (DDL)**
- (set of definitions to describe the schema)
- after compilation, we will get a set of tables as output stored in special files → Data Dictionary / Directory
  ↓
  "meta-data" - data about data
- Data storage and definition language
  ↳ "gives the implementation details of schema"

- **Data Manipulation language : (DML)**

(more difficult to implement)
(i) Procedural DML - 'what data and how to get that data from the database, is specified by user explicitly'

(ii) Non-procedural DML - 'only what data is specified'

09/09/24

→ **Database Administrator :**
- The person having central control of both data and program accessing that data over the system.
- Coordinates all activities of the database.

(i) Schema definition
(ii) Storage structure and access method definition
(iii) Schema and physical organization modification
(iv) granting of authorization for data access
(v) Intregrity constraint specification

*

To be what we are and to become what we are capable of becoming, is the only end of life. – Baruch Spinoza

4/2024

Monday	1	8	15	22	29
Tuesday	2	9	16	23	30
Wednesday	3	10	17	24	
Thursday	4	11	18	25	
Friday	5	12	19	26	
Saturday	6	13	20	27	
Sunday	7	14	21	28	

MARCH
Saturday
30

13.week

DML preprocessor { a special character is ~~90/prede~~ for each DML call.

→ Database Users :

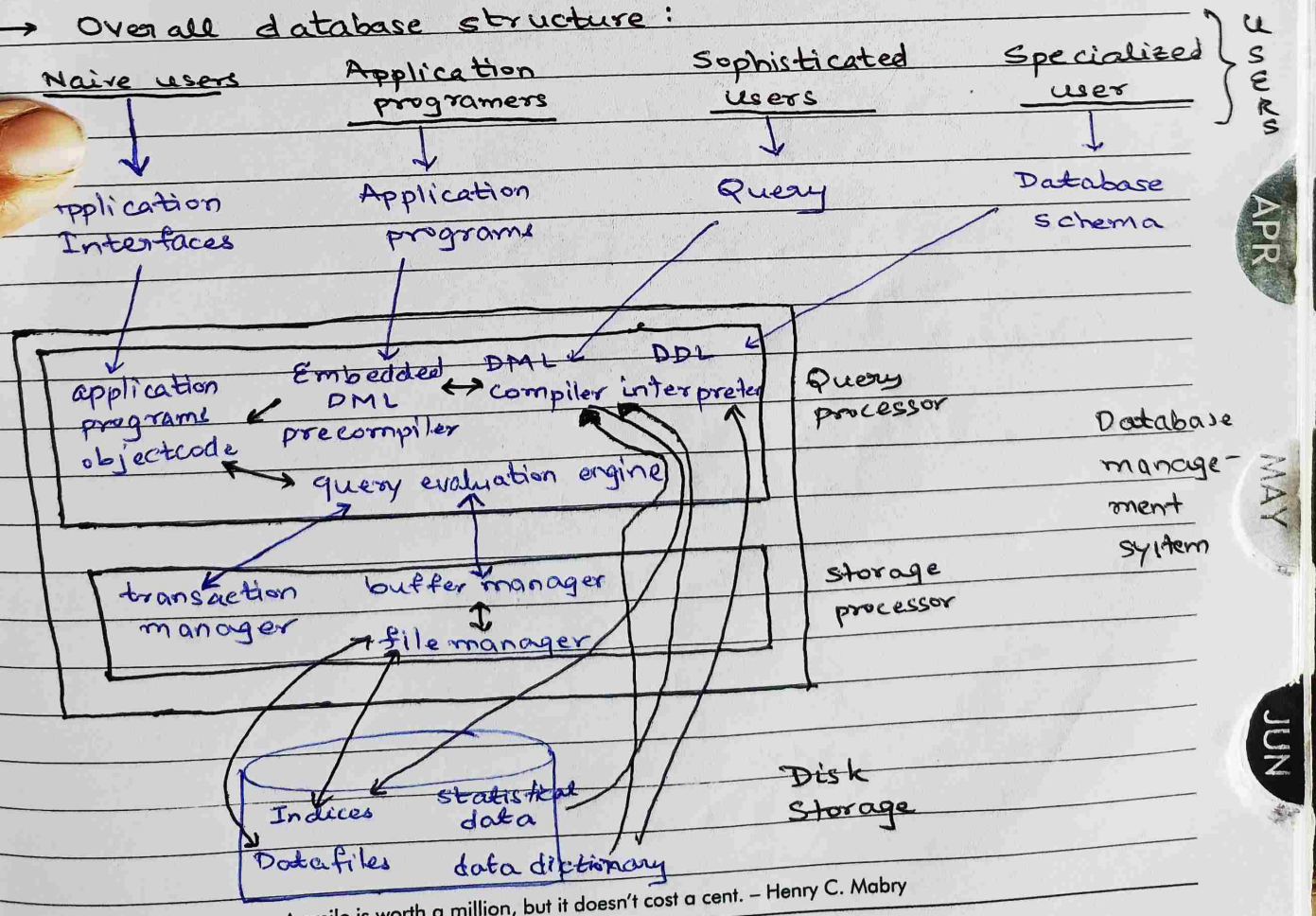
i) Application programmers → interact by DML calls : "Embedded in host language"
   ↳ - compiled by DML precompiler
   - converted into normal procedural form after compilation.
   - low level instructions generated (output)

ii) Sophisticated users
   - Interact without any application program
   - Interact with query language

iii) Specialized users
   - Interact with some specialized methods

iv) Naive users
   - interact with existing application programs.

→ Overall database structure :



A smile is worth a million, but it doesn't cost a cent. – Henry C. Mabry

31

MARCH
Sunday

3/2024

Monday	–	4	11	18	25
Tuesday	–	5	12	19	26
Wednesday	–	6	13	20	27
Thursday	–	7	14	21	28
Friday	1	8	15	22	29
Saturday	2	9	16	23	30
Sunday	3	10	17	24	31

91/275

13.week

Transaction Manager :

It creates consistency in the system and ensures that the concurrent operations must be performed.

8

Buffer Manager : Allocates the disk storage and the structure of the storage.

9

File Manager : It fetches data from the storage.

10

11 Indices : For faster access of data

12

13

14

15

19 APRIL Friday

Monday	1	8	15	22
Tuesday	2	9	16	23
Wednesday	3	10	17	24
Thursday	4	11	18	25
Friday	5	12	19	26
Saturday	6	13	20	27
Sunday	7	14	21	28

16. week

110/256

30/09/24 (SG)

→ **Entity-Relationship Model :**

- collection of entities
- relationship among entities

"entity sets" ← { It is an object that exists and distinguishable from other objects.

"attribute" → (describing property)

collection of entities of same type and also share the same properties

relationship : association among entities , min. 2 entity sets needed. (binary relationship)

Relationship sets : $\{ e_1, e_2, \ldots, e_n \mid e_1 \in E_1, e_2 \in E_2, \ldots, e_n \in E_n \}$

↓ relationship.

NOTE :

- There may exist an entity which is not associated with relationship set.
- Relationship sets can also have attributes.

Degree of Relationship sets :

Number of entities in the relationship sets associated via a relationship

"Domain" - set of permitted values for each attribute

- Types of Attributes :-

i) Simple ii) Composite iii) Single valued

iv) Multivalued v) Component vi) Derived

- Mapping Cardinalities :

i) one to one ii) one to Many

iii) Many to one iv) Many to many

Experience is what you get when you don't get what you want. – Don Stanford

8	15	22	29		
9	16	23	30		
10	17	24	-		
11	18	25	-		
12	19	26	-		
13	20	27	-		
14	21	28	-		

bjects.

roperties

ed.

r)

En}

ip set.

Monday	-	6	13	20	27
Tuesday	-	7	14	21	28
Wednesday	1	8	15	22	29
Thursday	2	9	16	23	30
Friday	3	10	17	24	31
Saturday	4	11	18	25	-
Sunday	5	12	19	26	-

- **KEYS**

entity sets → 1. Super Key - one or more attribute jointly identifies the values of entities in the entity sets uniquely.

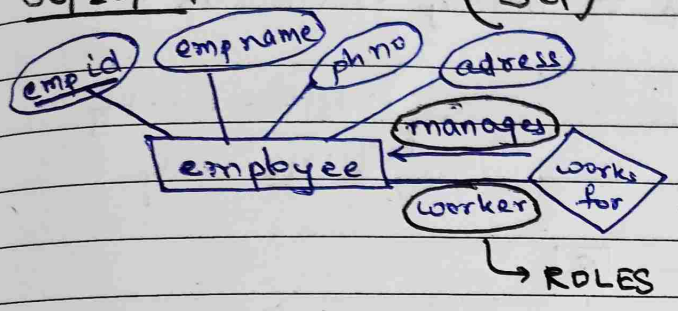
relationship set → 2. Candidate Key - "minimal super key"

3. Primary Key - one of the candidate key which can uniquely identify each of the entities in the entity set.

"of entitysets jointly describes the primary key of relationship set."

- When there is more than one candidate key, we have to consider "semantics".

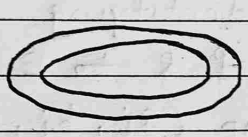
03/10/24 (SG)



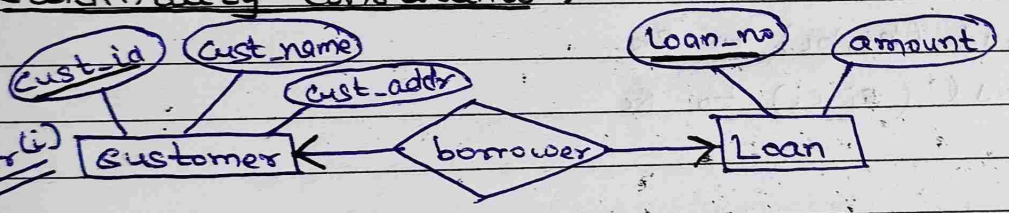
→ attributes
→ entity
→ relationship

for representing primary key
↳ underline.


→ derived attribute


→ multivalued attribute

→ Cardinality Constraints :



For (i)

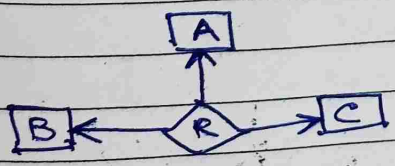
(i) One to one

(ii) One to many : A customer can have several loans. Multiple loans can have a single customer.

(iii) Many to One : Many customers can have a single loan. A single loan can be associated with multiple customer via the relationship model.

(iv) Many to many : Number of customers can have multiple loans. Many loans can be associated with multiple customer.

→ Ternary Relationship :

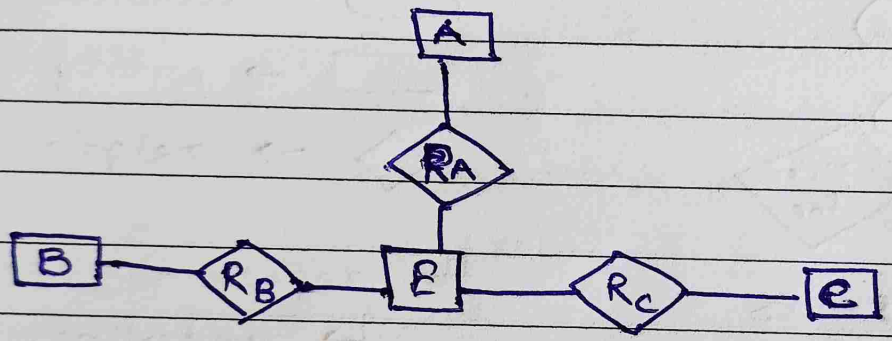


- Only one arrow is possible from relationship to a entity.

If allowing multiple arrows —

- Each entity 'A' is associate uniquely with entity 'B' and 'C'.

Converting the Ternary Relationship to Binary Relationship -



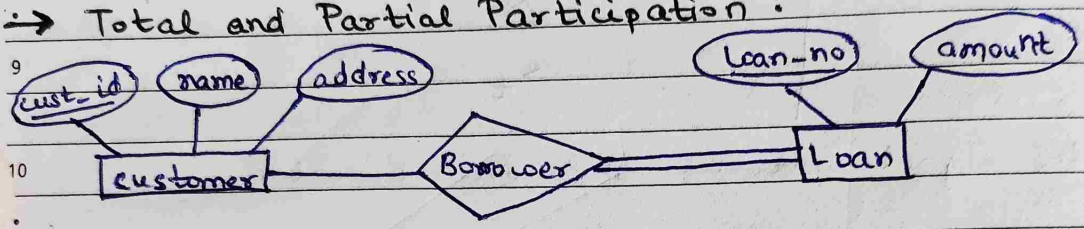
↳ Additional entity
↳ "bridge" entity

i) Create a special identifying attribute for E.
ii) Add any attribute of R to E
iii) for each relationship (a_i, b_i, c_i) in R, create -

- create a new entity e_i in E
- add (e_i, a_i) to R_A
- add (e_i, b_i) to R_B
- add (e_i, c_i) to R_C

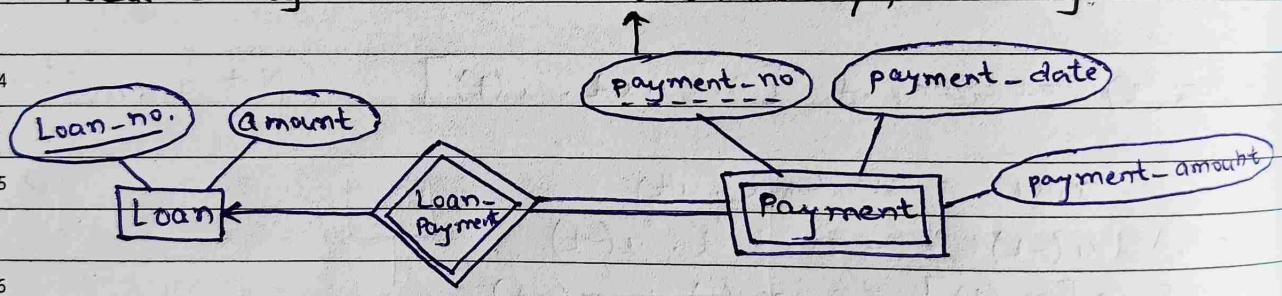
23/10/24 (SG)

→ Total and Partial Participation :



• Every loan in the loan entity set must have a customer.
  Loan is totally participating with relationship borrower.
• A customer may not have a loan.

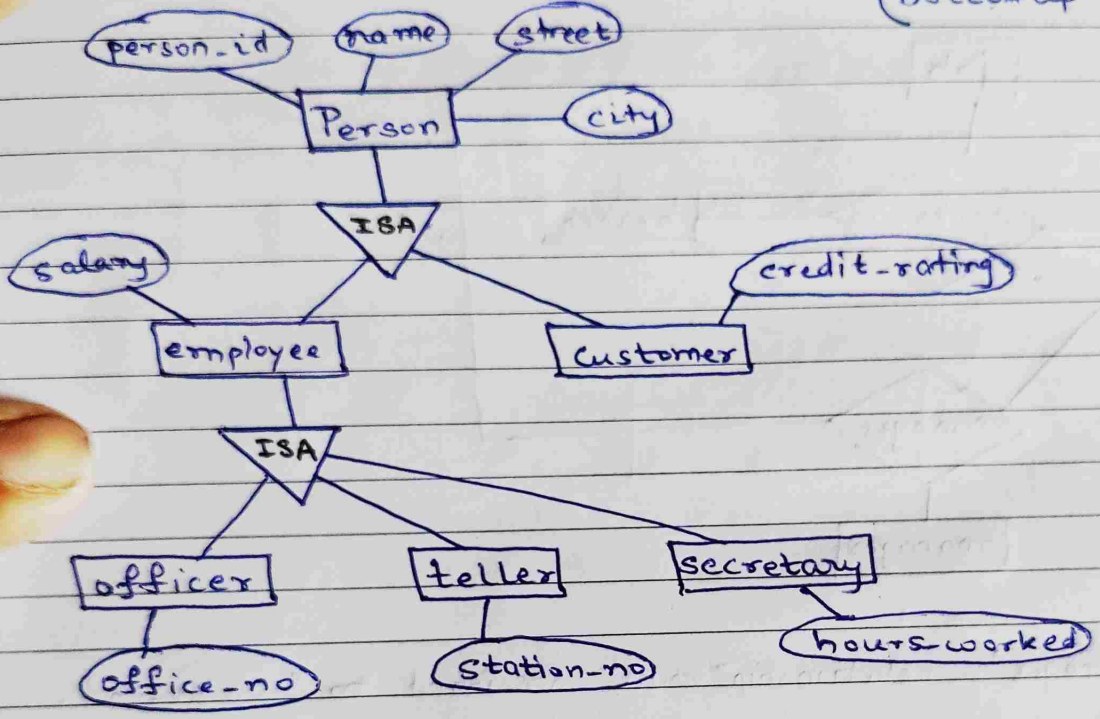
→ Weak Entity Sets : discriminator / partial key



- An entity that doesn't have any primary key is weak entity.
- Dependent upon strong entity sets.

Payment is associated with loan (the identifying entity set) via the relationship loan-payment.
Loan_no. and payment_no together makes up the primary key for the weak entity set - Payment.
But loan_no is not explicitly mention in the entity Payment as it is already implicitly defined in the relationship - Loan-Payment.

Specialization / Generalization : → (Top-down approach) / (Bottom-up approach)



ISA → superclass - subclass relationship

Attribute inheritance - every subgroup can inherit all the attributes and relationship from the superclass.

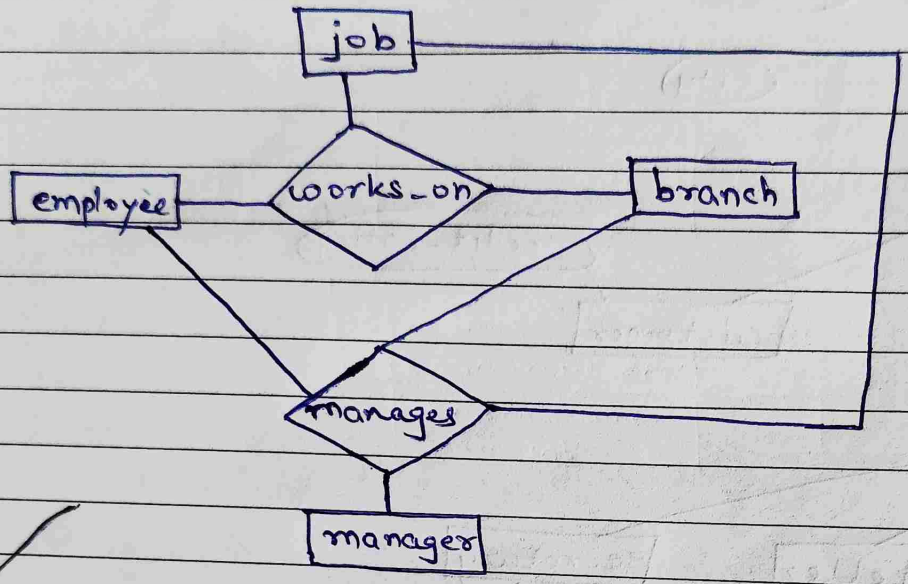
→ Design Constraints for Specialization / Generalization :

i) Condition defined

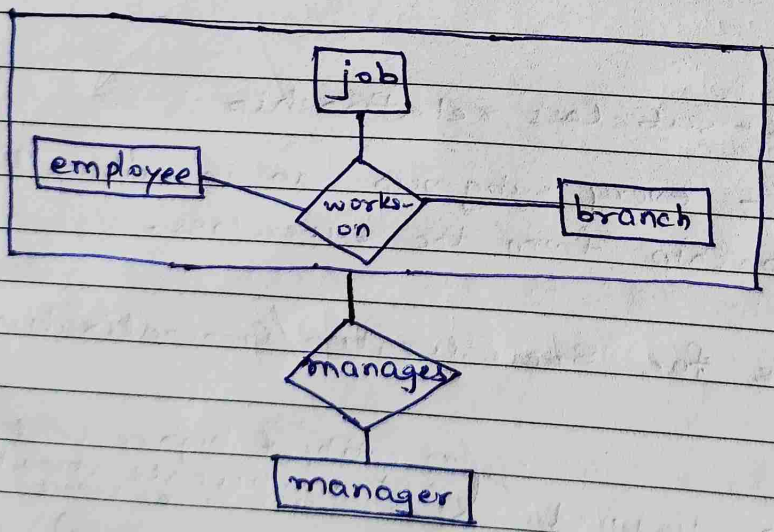
ii) user - defined → disjoint (an entity of higher level must participate atmost one with lower level)
→ overlapping - (more than one)

iii) completeness constraint → total - (atleast one)
→ partial

→ Aggregation : (relationship b/w relationship.)



• Every 'manages' relationship must correspond to 'works_on'.

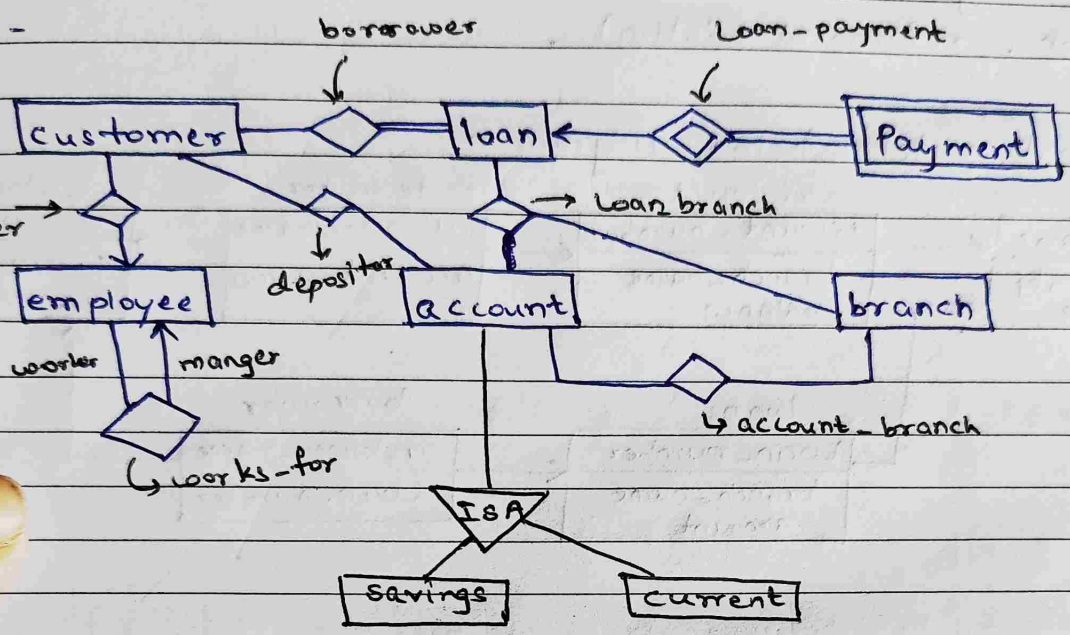


← abstract entity

Q : customer, loan, payment, employee, account, branch.
↑ entities.

Draw a E-R diagram.

Ans:-



06/11/24

→ Relational Model :

Given set $D_1, D_2, \ldots, D_n$ a relation 'r' is a subset of -

$r \subseteq D_1 \times D_2 \times \ldots \times D_n$

Thus, a relation (r) is a set of n-tuples $(a_1, a_2, \ldots a_n)$ where each $a_i \in D_i$.

If $A_1, A_2, \ldots A_n$ are attributes
$R = (A_1, A_2, \ldots A_n)$ is a relation schema
$r(R)$ denotes the relation 'r' on the relation schema 'R'

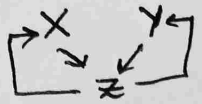
Relation Schema

Keys

Let $K \subseteq R$.

1. Superkey - if values of K are sufficient to identify a unique tuple of each possible relation $r(R)$.
2. Candidate Key - if K is minimal
3. Primary Key - a candidate key chosen as the principal means of identifying tuples within a relation.

The winds blow strongest against those who stand tallest. – FC Hayes



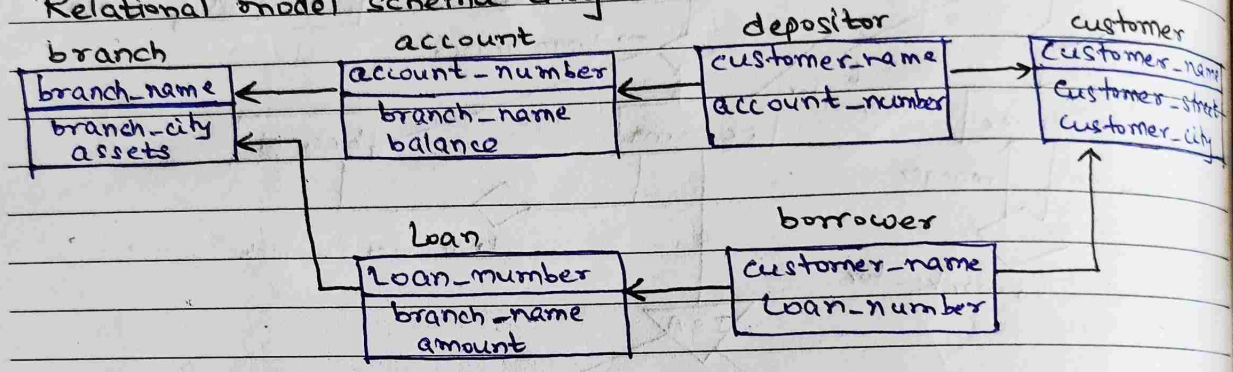
Z → referencing relation

X, Y → referenced relation.

NOTES

4. Foreign Key – attribute that corresponds to primary key of another relation. (referenced relation) ← referencing relation.

Relational model schema diagram –



→ Query Language :

Pure language { i) Relational Algebra → select (σ) project (π) union (∪) set diff (−) cartesian product (×) rename (ρ)

ii) Relational Calculus

Tuple relational Calculus

Domain relational calculus

$\sigma_p(r) = \{ t \mid t \in r \text{ and } p(t) \}$

↓ select predicate

- formula in propositional calculus consisting of terms where each term consists of – <attribute> or <constant> and also connected by set of connectives ($\wedge, \vee, \neg$).

- set of comparison operators ($<, \leq, =, \neq, >, \geq$)

Q: Select name of customer who lives in burdwan.

Ans:- $\sigma_{\text{customer\_city} = \text{'burdwan'}}$ (Customer)

11/10/24 (SG)

i) Project (π)

ii) Union ($\cup$) - $r \cup s = \{t \mid t \in R \text{ or } t \in S\}$

Conditions -

i) Same parity

ii) Domain must be same

iii) Set Difference

$r - s = \{t \mid t \in R \text{ and } t \notin S\}$

iv) Cartesian Product

$r \times s = \{tq \mid t \in R \text{ and } q \in S\}$

A	B
α	1
β	2

C	D	E
α	10	a
β	10	a
β	20	b
γ	10	b

A	B	C	D	E
α	1	α	10	a
α	1	β	10	a
α	1	β	20	b
α	1	γ	10	b
β	2	α	10	a
β	2	β	10	a
β	2	β	20	b
β	2	γ	10	b

v) Rename (ρ)

Q:i) Find the loan number for each loan of an amount > 1200.

ii) Find the name of all customers who have a loan and account or both from the bank.

Ans:-

i) $\Pi_{\text{loan\_number}}\left(\sigma_{\text{amount} > 1200}(\text{loan})\right)$

ii) $\Pi_{\text{customer\_name}}(\text{depositor}) \cup \Pi_{\text{customer\_name}}(\text{borrower})$

(iii) Find the names of all customer who have a loan at the 'burdwan' branch but do not have any account in any branch of the bank.

Ans:-

$$\Pi_{\text{customer\_name}}\left(\sigma_{\text{branch\_name} = \text{'Burdwan'}}\left(\sigma_{\text{borrower.loan\_number} = \text{loan.loan\_number}}(\text{borrower} \times \text{loan})\right)\right)$$

$$- \Pi_{\text{customer\_name}}(\text{depositor})$$

13/01/24 (SG)

(vi) Set Intersection –

$$r \cap S = \{ t \mid t \in r \text{ and } t \in S \}$$

(vii) Natural Join ($\bowtie$) –

A natural join $r \bowtie S$ is a relation on schema R U S obtained by –

1. considered each pair of tuple tr from R and ts from S.
2. If tr and ts have the same value on each of the attributes in $r \cap S$, add a tuple (t) to the result where 't' has the same value as the tr on R and ts on S.

$R = (A, B, C, D)$ $S = (E, B, D)$ $\Rightarrow$ (A, B, C, D, E)

$$\left(\sigma_{r.B = s.B \ \wedge \ r.D = s.D} (r \times s) \right)$$

R ⋈ S

A	B	C	D	E
α	1	α	a	α
α	1	α	a	γ
α	1	γ	a	α
α	1	γ	a	γ
δ	2	β	b	δ

(viii) Division ($\div$)

$A = \{ A_1, A_2, \dots A_m, B_1, B_2, \dots B_n \}$

$B = \{ B_1, B_2, \dots, B_n \}$

$A \div B = \{ A_1, A_2, \dots, A_m \}$

$$r \div s = \{ t \mid t \in \pi_{R-S}(r) \wedge \forall u \in s (t \cup \in r) \}$$

Everybody thinks of changing the world; but no one thinks of changing himself. – Leo Tolstoy

Q: Find the name of all customers who have a loan at the bank and the loan amount.

Ans:-

$\Pi_{customer\_name, loan\_number, amount}$ (loan ⋈ borrower)

→ Aggregate operations (sum, max, min, avg, count)

$G_1, G_2, \ldots, G_n$ g $F_1(A_1), F_2(A_2), \ldots, F_n(A_n)$ (E)

$G_1, G_2, \ldots, G_n$: list of attributes on which to group

$A_1, A_2, \ldots, A_n$: attributes name

$F_1, F_2, \ldots, F_n$: aggregate function

→ Outer Join

LOAN ⇒

loan_number	branch_name	amount
L-170	A	3000
L-230	B	4000
L-260	C	1700

BORROWER ⇒

customer_name	loan_number
X	L-170
Y	L-230
Z	L-155

Left outer join (loan ⋈ borrower)

loan_number	branch_name	amount	customer_name
L-170	A	3000	X
L-230	B	4000	Y
L-260	C	1700	Null

11 MAY Saturday

					5/2024
Monday	–	6	13	20	27
Tuesday	–	7	14	21	28
Wednesday	1	8	15	22	29
Thursday	2	9	16	23	30
Friday	3	10	17	24	31
Saturday	4	11	18	25	–
Sunday	5	12	19	26	–

132/234

19.week

(Loan ⟗ borrower) Right Outer Join

loan_number	branch_name	amount	customer_name
L-170	A	3000	X
L-230	B	4000	Y
L-155	NULL	NULL	Z

(Loan ⟗ borrower) full outer Join

Loan_number	branch_name	amount	customer_name
L-170	A	3000	X
L-230	B	4000	Y
L-260	C	7000	NULL
L-155	NULL	NULL	Z

18/011/24

→ Relational Calculus :

Tuple variable defined → ranges of the relation.

↓ over rows → Tuple Relational Calculus

→ over columns → Domain Relational Calculus.

- sequence may or may not be maitained.

- Tuple Relational Calculus

$\{t \mid COND(t)\}$

Q: Find the first and last name of all employees whose salary is above Rs. 5000.

Ans:- $\{t.FNAME, t.LNAME \mid Employee(t) \text{ AND } t.salary > 5000\}$

$\Rightarrow \pi_{FNAME, LNAME}\left(\sigma_{salary > 5000}(Employee)\right)$

Justice being violated, destroys justice being preserved, preserves. – Manusmriti

Existential and Universal Quantifier

"∃" "∀"

- "bounded" b/w values if they are "quantified" otherwise they are "free".

- if F is formula then ∃t(F) is also a formula.
  - there must exist one tuple that satisfies it ⇐ true
  - false
  - ↔ atleast one tuple must be assigned to free occurence of tuple variable.

- if F is a formula then ∀t(F) is also a formula.
  - all tuples of the relation must satisfy the condition. ⇐ true
  - false
  - ↕ all tuples must be assigned to the free occurence of the tuple variable.

#

① Employee (FNAME, LNAME, MINIT, SSN, BDATE, ADDRESS, SEX, SALARY, SUPERSSN, DNO)

② Department (DNAME, DNUMBER, MGRSSN, MGRSTARTDATE)

③ DEPT_LOCATIONS (DNUMBER, DLOCATION)

④ PROJECT (PNAME, PNUMBER, PLOCATION, DNUM)

⑤ WORKS_ON (ESSN, PNO, HOURS)

⑥ DEPENDENT (ESSN, DEPENDENT_NAME, SEX, BDATE, RELATIONSHIP)

Q: Retriere the name and address of all employees who work for the research department.

Ans:- { t.FNAME, t.LNAME, t.ADDRESS | Employee(t) AND ∃d Department(d) AND d.DNAME = "Research" AND t.DNO = d.DNUMBER }

Q: For every project located in Kolkata, list the project number, the controlling dept. no and dept manager LNAME, BDATE, ADDRESS.

Ans:-

Hypocrisy may deceive the cleverest man, but the least wide-awake of children recognizes it. – Leo Tolstoy

JUN

19 MAY Sunday

Monday – 6 13 20
Tuesday – 7 14 21
Wednesday 1 8 15 22
Thursday 2 9 16 23
Friday 3 10 17 24
Saturday 4 11 18 25
Sunday 5 12 19 26

140/226 20 week

27/11/24 (SG)

→ Domain Relational Calculus:

$\{x_1, x_2, \ldots, x_n \mid COND(x_1, x_2, \ldots, x_n, x_{n+1}, \ldots x_{n+m})\}$

↓

domain variables that range over the domain of attributes

Q: Retrieve the birthdate and address of the employee whose name is 'John. B. Smith'.

Ans:- $\{uv \mid (\exists q)(\exists r)(\exists s)(\exists t)(\exists w)(\exists x)(\exists y)(\exists z)(EMPLOYEE(qrstuvwxyz)$

and q = 'John' and r = 'B' and s = 'Smith' $\}$

Q: Retrieve the name and address of all employees who works for research department.

Ans:- $qsv \quad (\exists l)(\exists m)(\exists z)$

$\{$ ~~qsv~~ $\mid$ ~~$(\exists x)(\exists t)(\exists u)(\exists w)(\exists x)(\exists y)(\exists z)$~~ $(EMPLOYEE(qrstuvwxyz)$

and $DEPARTMENT(lmno)$ and l = 'Research'

and $z = m\}$

Q: For every project located in Kolkata list the project no, the controlling department no. and dept. manager's last name, birth date and address.

Ans:- $(\exists k)$

$\{suvik \mid (\exists j)(\exists z)(\exists m)(\exists t)$ $EMPLOYEE(qrstuvwxyz)$

~~and~~ ~~$DEPARTMENT(lmno)$~~ ~~and~~ $PROJECT(hijk)$ and

j = 'Kolkata' and $k = z\}$

and $n = t$

Reason can answer questions, but imagination has to ask them. – Ralph Gerard

→ Database Normalization :

Process of efficient of organization of data in the database

↓

"Relations"

Goal - i) Eliminate redundant data

ii) Ensure data dependancies

↓

Normal Form ⇒ 1NF, 2NF, 3N, BCNF

↓

series of test over the relation to check whether it satisfies or not.

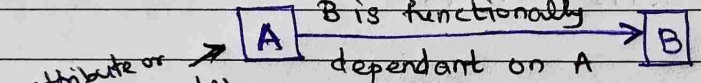
Reason : A relation can be normalized to a specific form to prevent the possible occurences of update anomalies (insertion, deletion and modification).

① Deletion Anomaly - ② Update Anamoly ③ Insertion Anamoly

02/12/24 (SG)

- Normalization – formal method that identifies relations based on their primary keys and functional dependencies among their attributes.
  ↓
  constraint b/w the attributes and describes the relationship b/w the attributes in a relation.

- If A and B are two attributes of a relation, the B is functionally dependant on A, if each value of A is associated with every single value of B.



- Every non-key attribute should be dependant on key-attribute.

- Types of FDs –

i) Fully Functional Dependency $A \rightarrow B$
   - B is functionally dependent on A
   - but no proper subset of A

ii) Partial Dependency

iii) Transitive Dependency
   - If $A \rightarrow B$, $B \rightarrow C$ then $A \rightarrow C$, C is functionally dependent on A via B.
   - Also, A should not be dependent on B or C.

① 1NF → 1NF (condition) → should not exist "repeating groups"
for every single occurrence of key attribute there exists multiple attributes/values.

Customer Rental Table

Customer_No	Cname	Property_No ① ②	PAddress ① ⑤ ③	Rent start	Rent finish	Rent	Owner No	OName

Customer

Customer_No	Cname

Property_Owner

Customer_No	Property_No	. . . .

② **2NF**

- Relation should be in 1NF
- There will be no partial dependency.

Primary Key → Customer_no + Property_No

Full F.D → Customer_no + Property_No → Rent start, Rent finish

Partial → Customer_no + Property_No → Cname

Customer_no + Property_No → PAddress, Rent, Owner_No, ONam

Rental

Customer_no	Property_No	Rent start	Rent finish

Property owner

Property_No	PAddress	Rent	owner_No	Oname

The one thing more difficult than following a regimen is not imposing it on others. – Marcel Proust

JUN

27 MAY Monday

Monday	–	6	13	20	27
Tuesday	–	7	14	21	28
Wednesday	1	8	15	22	29
Thursday	2	9	16	23	30
Friday	3	10	17	24	31
Saturday	4	11	18	25	–
Sunday	5	12	19	26	–

5/2024

148/218 04/12/24 (SG) 22.week

③ 3NF

- relation should be in 2NF
- there should not be any transitive dependency

In 'Property_owner' relation transitive dependency exist.

↓

Property_no → Owner_no

Owner_no → Oname

↓

Property For Rent

Property No	PAddress	Rent	Owner_no

Owner

owner_no	O name

④ BCNF

3NF ⇏ BCNF
BCNF ⇒ 3NF
} If a relation is in 3NF, it does not means that it is in BCNF.

↳ If a relation is in BCNF, it means it is in 3NF.

Client_No	Interview_Date	Interview_Time	Staff_No	Room_No

FDs

Client_No, Interview_Date → Interview_Time, Staff_No, Room_No

staff_No, Interview_Date, Interview_Time → Client_No

Staff_No, Interview_Date → Room_No

→ Closure of a set of FDs:

$F \rightarrow F^+$

⁂ Armstrong Axioms :

i) Reflexity – $\beta \subseteq \alpha$; $\alpha \rightarrow \beta$

ii) Augmentation – $\alpha \rightarrow \beta$; $\gamma\alpha \rightarrow \gamma\beta$

iii) Transitivity – $\alpha \rightarrow \beta$, $\beta \rightarrow \gamma$; $\alpha \rightarrow \gamma$

→ Additional Rules :

i) Union : if $\alpha \to \beta$ and $\alpha \to \gamma$ then $\alpha \to \beta\gamma$

ii) Decomposition : if $\alpha \to \beta\gamma$ then $\alpha \to \beta$ and $\alpha \to \gamma$

iii) Pseudotransitivity : if $\alpha \to \beta$ and $\gamma\beta \to \delta$ then $\alpha\gamma \to \delta$

Q : R = (A, B, C, G, H, I)

F = $A \to B$, $A \to C$, $CG \to H$, $CG \to I$, $B \to H$. find F^+.

Ans:- $F^+ = \{A, B, C, H\}$

$F^+ = \{ A \to H, AG \to I, AG \to H, CG \to HI \}$

$AG \to CG$

$CG \to I$

$AG \to I$

(Augmentation) $CG \to CGI$

$CGI \to HI$

→ Closure of Attribute Sets :

result := α

while (changes to result) do

for each $\beta \to \gamma$ in F do

begin

if $\beta \subseteq$ result then

result := result $\cup \gamma$

end

Q: R = (A, B, C, G, H, I)

F = { $A \to B$, $B \to C$, $CG \to H$, $CG \to I$, $B \to H$ }

Ans:- $AG^+ = \{B, C, H, A, G, I\}$

result = AG

result = ABCG

result = ABCGHI

Real success is finding your lifework in the work that you love. – David McCullough

JUN

29 MAY Wednesday

					5/2024
Monday	–	6	13	20	27
Tuesday	–	7	14	21	28
Wednesday	1	8	15	22	29
Thursday	2	9	16	23	30
Friday	3	10	17	24	31
Saturday	4	11	18	25	
Sunday	5	12	19	26	

150/216

22 week

1. Testing for SuperKey
   if α^+ contains all the attributes of R
2. Testing FDs
   if $\beta \subseteq \alpha^+$
   then $\alpha \rightarrow \beta$ exists
3. Computing closure of F
   for each $\gamma \subseteq R$ find γ^+.
   and for each $S \subseteq \gamma^+$ output a FD $\gamma \rightarrow S$.

7/2024

Monday	1	8	15	22	29
Tuesday	2	9	16	23	30
Wednesday	3	10	17	24	31
Thursday	4	11	18	25	-
Friday	5	12	19	26	-
Saturday	6	13	20	27	-
Sunday	7	14	21	28	-

JUNE
Sunday
02

22.week

154/212

09/12/24 (SG)

→ Finding Extraneous Attributes :

Consider a set 'F' of functional dependancies and a F.D $\alpha \to \beta$ in F.

① attribute 'A' is extraneous in 'α' if $A \in \alpha$ and F logically implies $(F - \{\alpha \to \beta\}) \cup \{(\alpha - A) \to \beta\}$

② attribute 'A' is extraneous in 'β' if $A \in \beta$ and set of FDs $(F - \{\alpha \to \beta\}) \cup \{\alpha \to (\beta - A)\}$ logically implies F.

Testing if an attribute is extraneous :-

① Consider a set of FDs 'F' and a FD $\alpha \to \beta$ in F.

② To test if an attribute '$A \in \alpha$' is extraneous in 'α' -

i) Compute $(\{\alpha\} - A)^+$ using FDs in F.

ii) Check that $(\{\alpha\} - A)^+$ contains 'β', if it does then 'A' is extraneous in 'α'

③ To test if attribute '$A \in \beta$' is extraneous in 'β' -

i) Compute α^+ using only the dependancies in $F' = (F - \{\alpha \to \beta\}) \cup \{\alpha \to (\beta - A)\}$

ii) Check that α^+ contains 'A', if it does then 'A' is extraneous in β.

Q : $F = \{A \to C, AB \to C\}$

Ans :-

Here $\alpha = AB$, $\beta = C$

Now, $AB - A = B$ OR $AB - B = A$

$\therefore B^+ = \{B\}$ $A^+ = \{A, C\}$

A is not extraneous. $\therefore$ B is extraneous.

Q: $F = \{A \to C, AB \to CD\}$

Ans:- $\alpha = AB, \quad \beta = CD$

$CD - C$

$F' = \{A \to C, AB \to D\}$

$\therefore$ C is extraneous

$AB^+ = \{A, B, C, D\}$

$CD - D$

Again, $F' = \{A \to C, AB \to C\}$

$AB^+ = \{A, B, C\}$ $\therefore$ D is not extraneous.

→ Canonical Cover :

A canonical cover of F is a set of FDs F_c such that –

① F → all the functional dependencies in F

② F_c → all the FDs in F.

③ No functional dependency in F_c contains an extraneous attribute.

④ Each left side of the FDs in F_c is unique

Procedure:

repeat

use union rule to replace any FD in F

$\alpha_1 \to \beta_1$ and $\alpha_2 \to \beta_2$ with $\alpha_1 \to \beta_1\beta_2$

find a FD $\alpha \to \beta$ with an extraneous attribute either in 'α' or in 'β'

If an extraneous attribute is found, delete it from $\alpha \to \beta$ until 'F' does not change.

Q: $R = (A, B, C)$ $F = \{A \to BC, B \to C, A \to B, AB \to C\}$

Ans:- (∵ A is extraneous) \ Union / ①

② $F_c = \{A \to BC, B \to C\}$ ⇒ $F_c = \{A \to BC, B \to C, AB \to C\}$

③ $F_c = \{A \to B, B \to C\}$

$F' = \{A \to B, B \to C\}$ $A^+ = \{A, B, C\}$

07 JUNE Friday

159/207

23.week

16/12/24 (SG)

→ Lossless Join Decomposition :

A decomposition of R into R_1 and R_2 is lossless if and only if at least one of the following dependencies is in F^+.

$R_1 \cap R_2 \rightarrow R_1$

$R_1 \cap R_2 \rightarrow R_2$

→ Dependency Preservation :

A decomposition is dependency preserving if -

$(F_1 \cup F_2 \cup \ldots F_n)^+ = F^+$

If not then check for violation of FDs.

• Testing for dependency preservation :

To check if a FD $\alpha \rightarrow \beta$ is preserved in a decomposition of R into $R_1, R_2, \ldots, R_n$ apply the following test :

(with attribute closure done w.r.t F)

result = α

while (changes to result) do

for each R_i in the decomposition

$t = (result \cup R_i)^+ \cap R_i$

result = result $\cup$ t

if result contains all attributes in 'β', then FD $\alpha \rightarrow \beta$ is preserved.

Q : R = (A, B, C) F = {A → B, B → C}

Case I : R_1 = (A, B) ; R_2 = (B, C)

Case II : R_1 = (A, B) ; R_2 = (A, C)

Ans :- Case I

$A \rightarrow B \Rightarrow \alpha \rightarrow \beta$

For R_1 ↳ result = A

$t = (A \cap AB)^+ \cap AB = A^+ \cap AB$

$= AB$

We are what we repeatedly do. Excellence, then, is not an act, but a habit. – Aristotle

result = A ∪ AB = AB

For R_2 → $t = (AB \cap BC)^+ \cap BC$

$= B^+ \cap BC = BC$

result = AB ∪ BC

= ABC

The result contains B,

B → C

For R_1 → result = B, $t = (B \cap AB)^+ \cap AB$

$= B^+ \cap AB = B$

result = B ∪ B = B

For R_2 → $t = (B \cap BC)^+ \cap BC$

$= B^+ \cap BC = BC$

result = B ∪ BC

= BC. The result contains C.

Hence, the decomposition is preserved.

Case II

B → C

For R_1 → result = B, $t = (B \cap AB)^+ \cap AB$

$= B^+ \cap AB = B$

result = B ∪ B = B

For R_2 → $t = (B \cap AC)^+ \cap AC = \phi$

result = φ ∪ B = B.

∴ Decomposition is not preserved.

09 JUNE Sunday

161/205

23.week

→ BCNF decomposition Algorithm :

8- Testing BCNF decomposition

- to check if a Relation R_i in a decomposition of R is in BCNF.

9 i) either test R_i for BCNF w.r.t restriction of F to R_i

ii) or use the original set of FDs in F that hold on

10 R, but with the following test :

- a) for every set of attribute $\alpha \subseteq R_i$, check that either α^+

11 includes no attributes of $R_i - \alpha$ or includes all attributes

- of R_i.

12 b) if the code is violated by some $\alpha \rightarrow \beta$ in F, the

- dependency $\alpha \rightarrow (\alpha^+ - \alpha) \cap R_i$ hold on R_i and

13 R_i violates BCNF.

- Use the above dependency to decompose 'R'.

14

- Algo :

```
result = {R}
15  done := false
    compute F+.
16  while (not done) do
        if there is a schema Ri in result that is not in BCNF
17      then begin
            let α → β be a FD that holds Ri
18          such that α → Ri is not in F+.
            and α ∩ β = ϕ
19          result = (result - Ri) ∪ (Ri - β) ∪ (α, β)
        end
20  else done := true
```

*Q : R = (A, B, C) F = {A → B, B → C}

Key = {A}

18/12/24

(SG)

Ans:-

$A^+ = \{ABC$ $\quad B^+ = BC$

Testing for $B \to C$

$\alpha \to (\alpha^+ - \alpha)$ A Ri

$B \to (BC - B) \cap ABC$

$B \to C \cap ABC$

$B \to C$ violating the condition for BCNF.

Decomposition

result = ABC

$F^+ = \{A \to B, B \to C, A \to C\}$

$B \to ABC$ and $B \cap C = \phi$

(not in F^+)

result = $(ABC - ABC) \cup (ABC - C) \cup (B, C)$

$= \phi \cup \boxed{(A, B)} \cup \boxed{(B, C)}$

most people don't recognize them. – Ann Landers

11 JUNE Tuesday

163/203

18/12/24 (SG)

→ Transaction Processing:

- logical unit of database processing that includes one or more access operations (read_retrieval, write_insert or update, delete)

- read_item (x) "database item"
- write_item (x)

'x' → disk storage → frame buffer

x ← 'x'

program variable with same name.

- Problems of Concurrent Transactions:

① The Lost Update Problem

T_1	T_2
read_item(x)	
x := x-N;	
	read_item(x);
	x := x+M;
write_item(x);	
read_item(x);	
	write_item(x);
y := y+N;	
write_item(y);	

② Temporary Update or Dirty Read

T_1	T_2
read_item(x);	
x := x-N;	
write_item(x);	
	read_item(x);
	x := x+M;
	write_item(x);
read_item(y);	

③ Incorrect Summary Problem

T_1	T_3
	sum := 0
	read_item(A);
	sum := sum+A;
read_item(x);	⋮
x := x-N;	⋮
write_item(x);	⋮
	read_item(x);
	sum := sum+x;
	read_item(y);
	sum := sum+y;
read_item(y);	
y := y+N;	
write_item(y);	

Believe that life is worth living and your belief will create the fact. – William James