File Org. & data base system

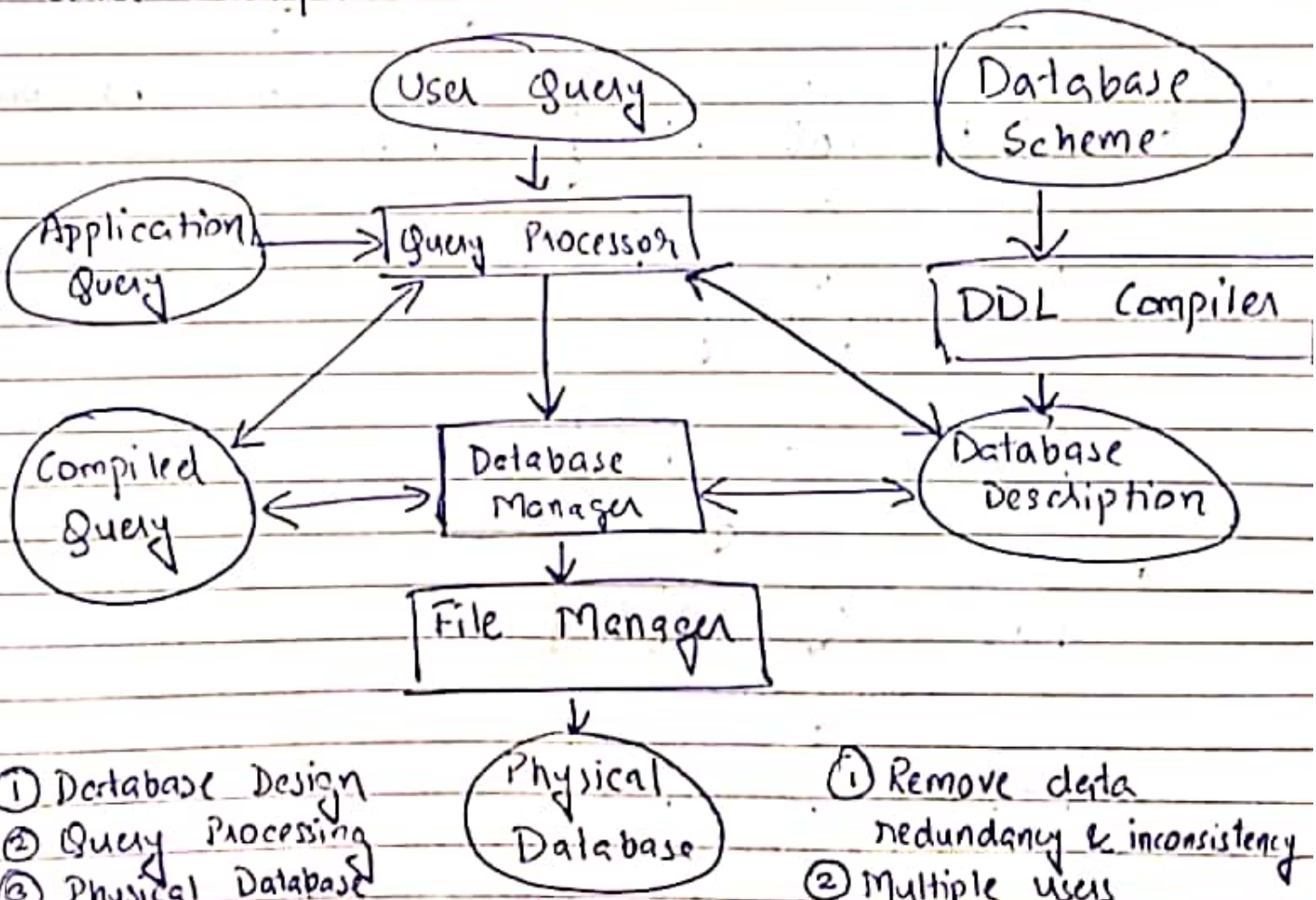
2/01/20

Principles of Database Systems : JD Ulman
Database System Concepts : H.F. Korth & A Silberschatz
Database Management Systems : Raghu Ramakrishna

Raw Data → Field(s) → Records

Data File / Table → Database

DBMS (Database Management System) : Software allows us to create database, datafiles, enter and manipulate data.



① Database Design
② Query Processing
③ Physical Database

① Remove data redundancy & inconsistency
② Multiple users (use same DB simultaneously)
③ Synchronization
④ Security Problems.

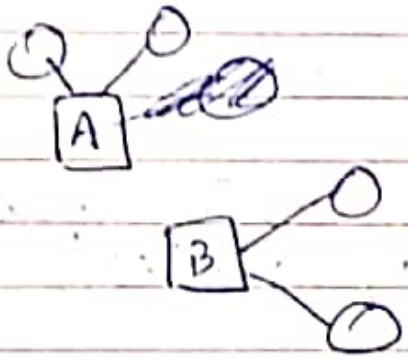
Scanned by CamScanner

23/01/20

Page No.
Date
YOUVA

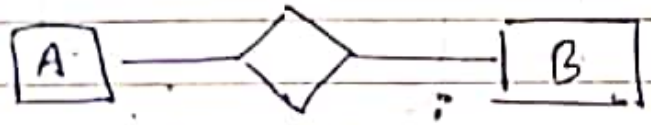
Entity - Relationship Model

- **Entity** :- Entity is an object distinguishible from other objects. Eg. → Roll Number
- **Entity Set** :- {Roll No., Name, Dept., Hall}
  Attributes.

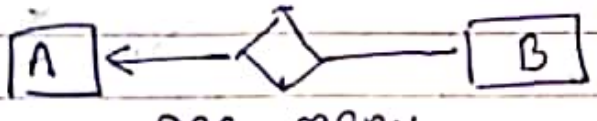


one - one
one - many
many - one
many - many

- **Key** :- It is a set of one or more than one attribute that uniquely identifies a row

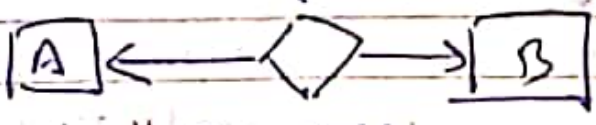


many - many



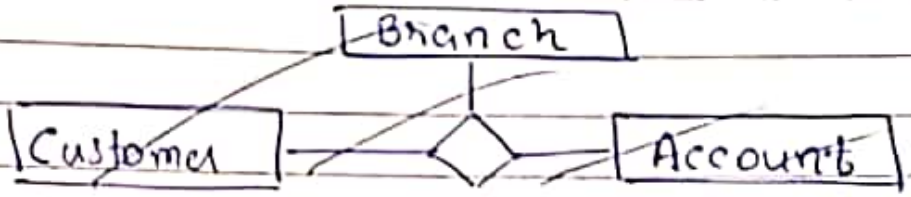
one - many

many - one



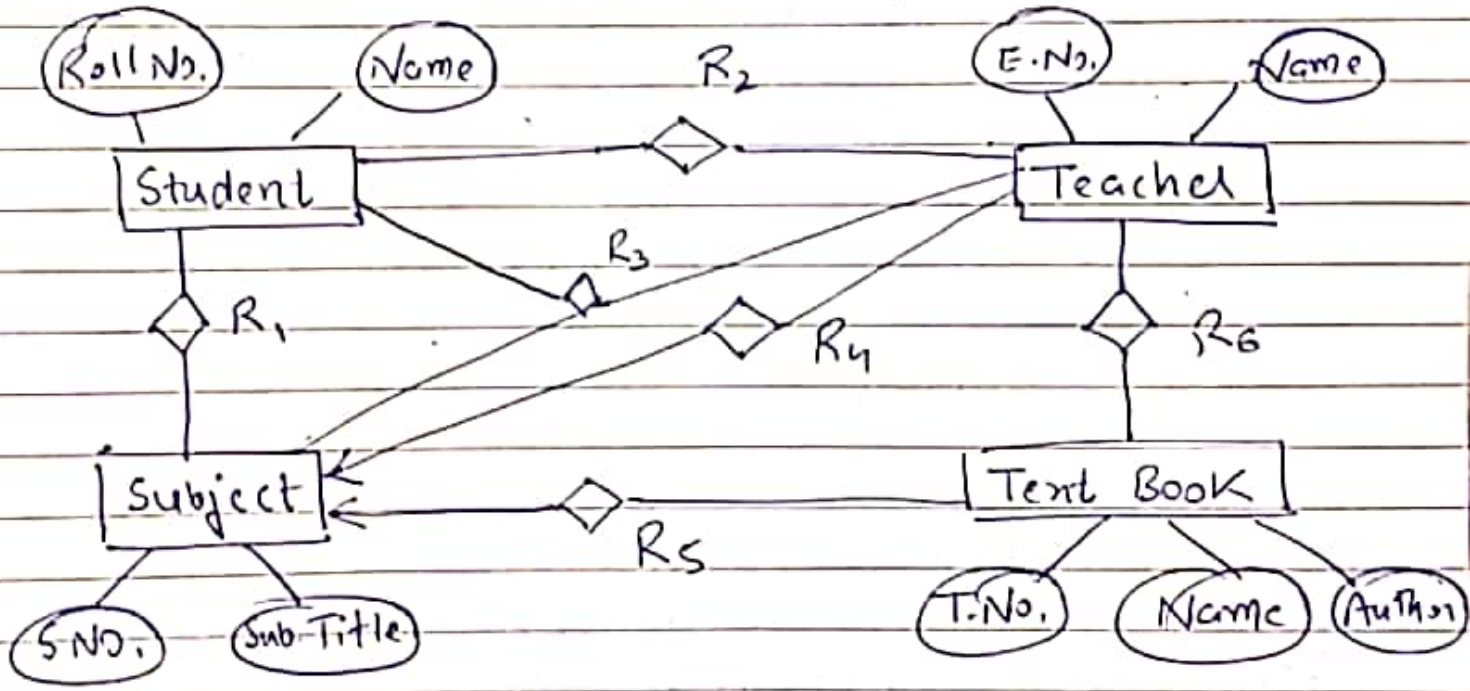
one - one

Scanned by CamScanner



Draw an ER Diagram with entities — Text book, subjects, students, teachers.

1) For each subject each student is taught by single teacher.
2) Each teacher teaches only one subject.
3) Each subject is taught by several teachers.
4) Teachers choose their resp. text book.



Student	Teacher	Subject	Text Book
Roll No. | . . .	E.No. | —	S.No. | — — -	T.No. | — -

R1

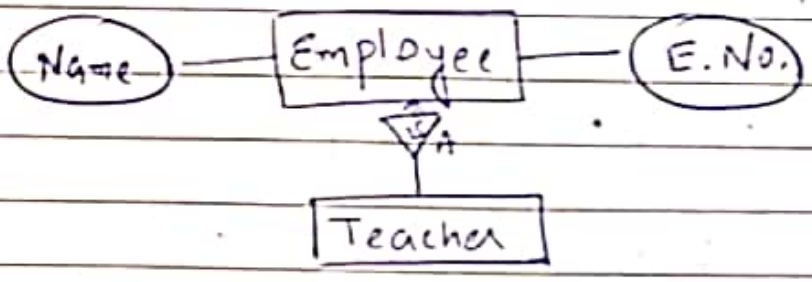
Roll-No.	S.No.

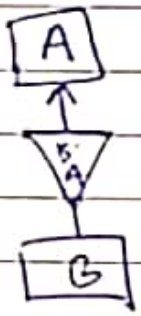
R3

Roll No	S.No.	E.No.

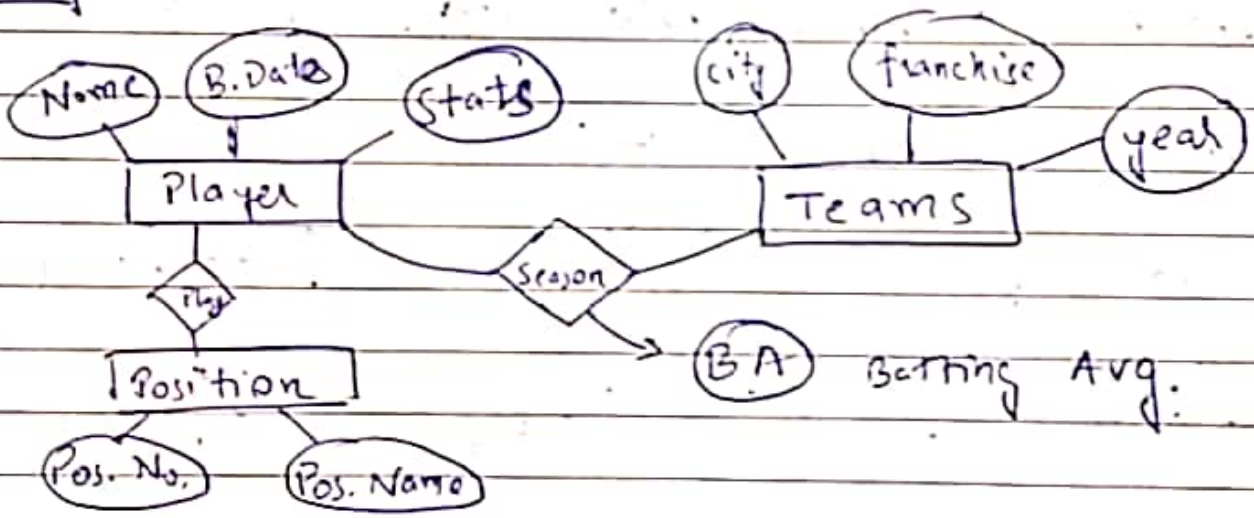
Relational Model

Scanned by CamScanner





⇒ B is a subset of A



Relational Table

Players (Name, B.date, state)
Position (Pos. No., Pos. Name)
Teams (City, Franchise, Year)

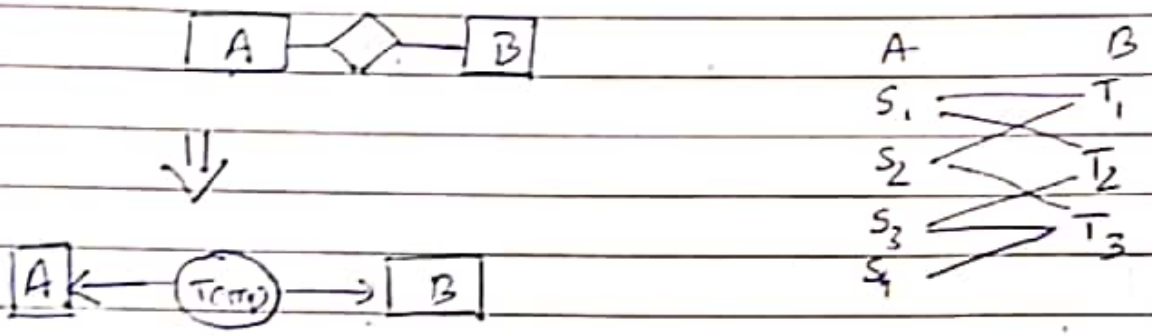
Play (Name, Pos. No.)
Season (Name, city, franchise, B.A.)

Network

Scanned by CamScanner

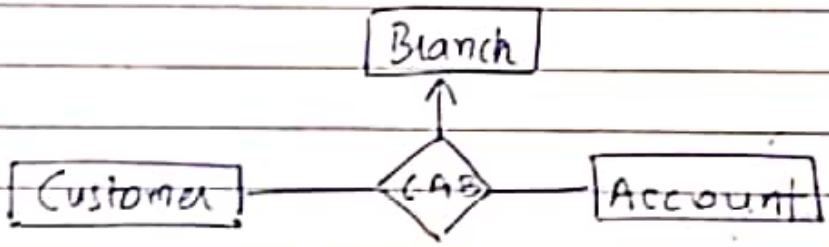
Network Model :- It is ER-Model restricted to binary & many-one relations only.

Drawing Network Model from ER model →

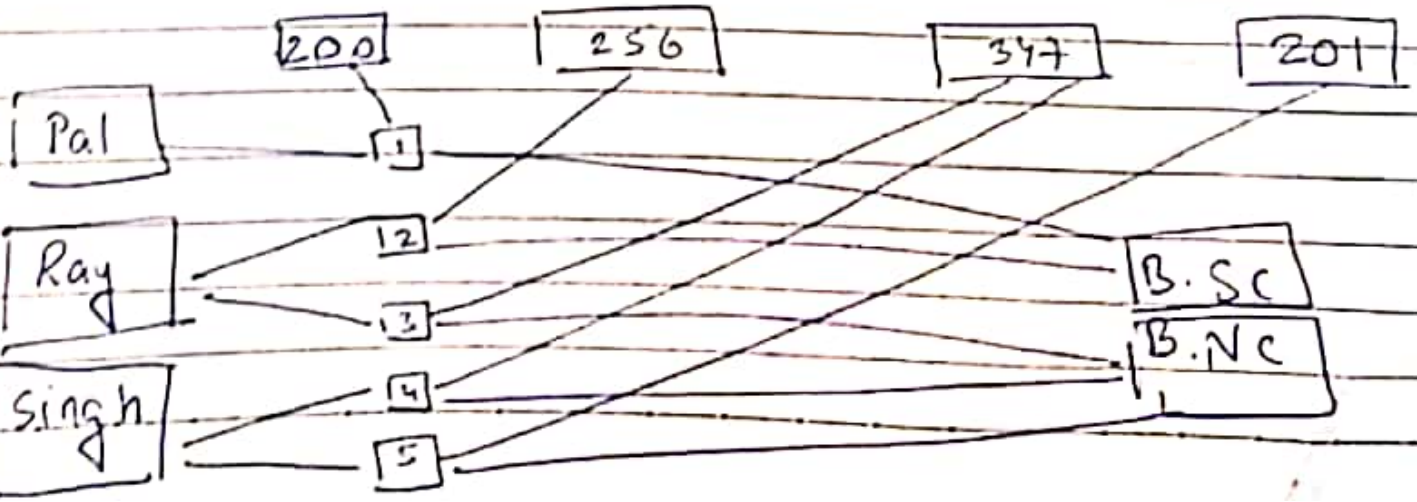


- Many-many relationship is converted to binary many-one relationships.

Eg :-

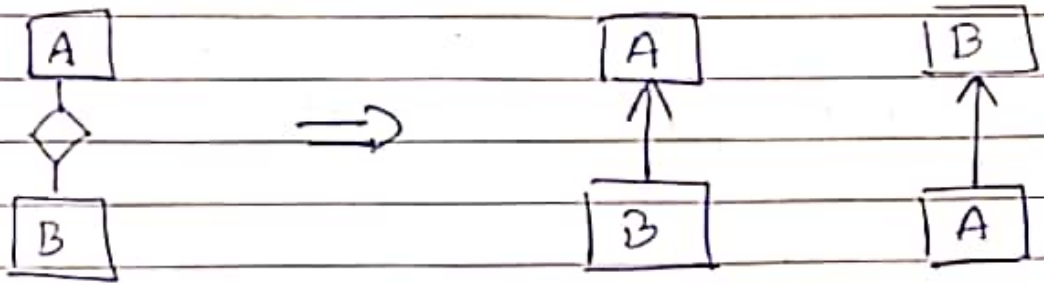


Branch	Customer	Account
B. SC.	Pal	200, 55
B. NC.	Ray	256, 1000
	Singh	347, 667
		201, 500

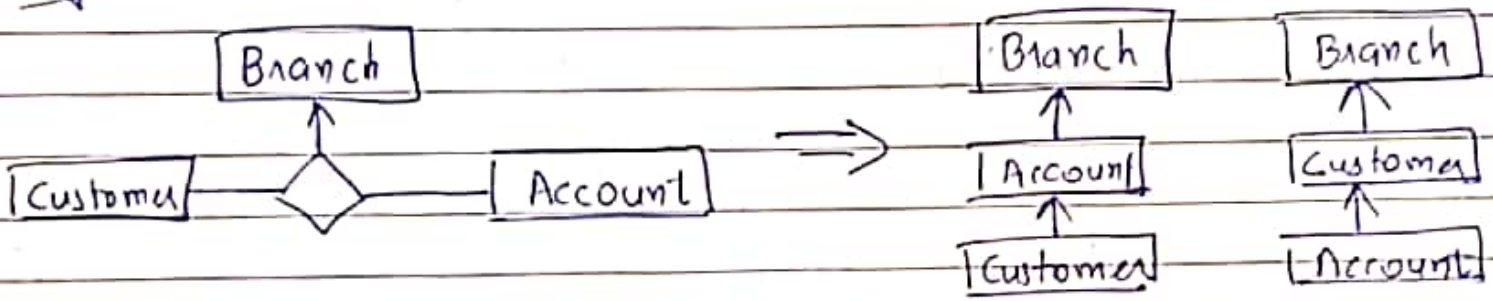


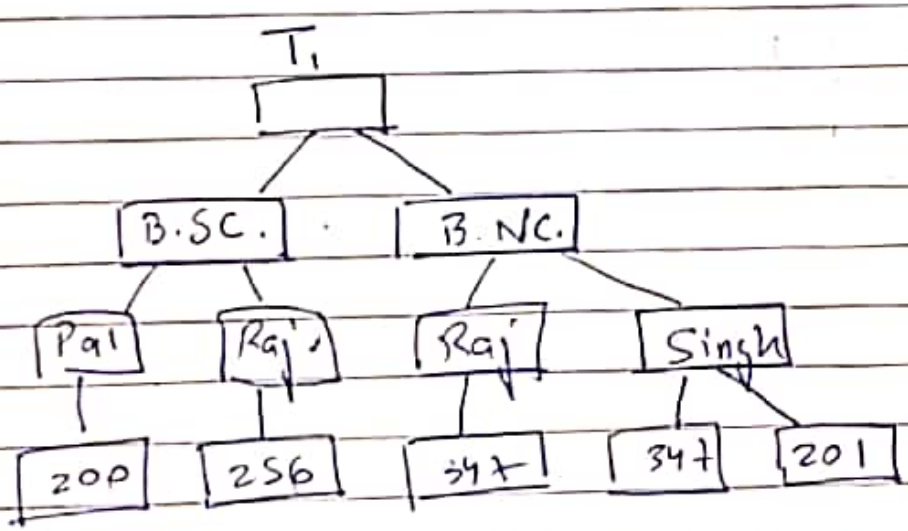
Scanned by CamScanner

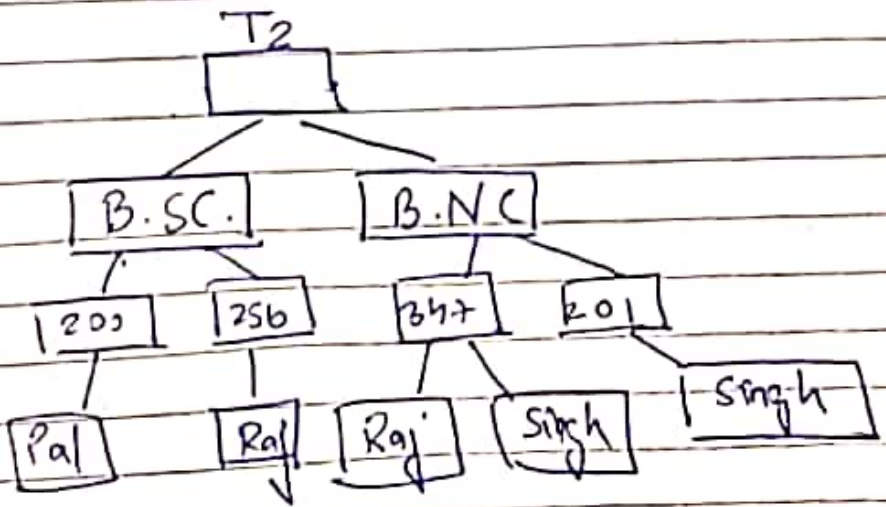
Hierarchial Model



Eg.







Scanned by CamScanner

Relational Database Design

Redundancy Anomaly
Updation Anomaly
Insation Anomaly
Deletion Anomaly

SGP (RN, SNAME, HALL, GAME, FEE)
↑ student game participation

⇒ Break into different tables to avoid the anomalies.

T_1 (RN, SNAME, HALL)
T_2 (GAME, FEE)
T_3 (R.N., GAMA)

Functional Dependancy

U → Universal set of attributes.

$U_i \subseteq U$ is a subset of attributes

$X \to Y$; $r(U_i)$ M_1 & M_2 are two tuples (records)

$$M_1[X] = M_2[X] \Rightarrow M_1[Y] = M_2[Y]$$

R(A, B, C, D)

A	B	C	D
a_1	b_1	c_1	d_1
a_1	b_2	c_1	d_2
a_2	b_2	c_2	d_2
a_2	b_3	c_2	d_3
a_3	b_3	c_2	d_4

$A \to C$
$D \to B$
$A \to A$ Trivial F.D.

Scanned by CamScanner

16/01/20

$A \to B$, $B \to C$ then $A \to C$

Closure of a set of FDs

$F^+ = \{ X \to Y \mid F \models X \to Y \}$

$R(A,B,C)$ $F = \{ A \to B, B \to C \}$

$F^+ = \{ A \to C, AB \to ABC, B \to BC, \ldots \}$

Armstrong's Axioms

1) If $Y \subseteq X \subseteq U$ then $X \to Y$
2) If $X \to Y$ and $Z \subseteq U$

$XZ \to YZ$, $XZ = X \cup Z$

3) If $X \to Y$ and $Y \to Z$ then $X \to Z$

Eg: $R(CITY, ST, PIN)$; CITY ST → PIN
PIN → CITY

→ (CITY ST) ∪ (CITY ST) → PIN ∪ (CITY ST)

⇒ CITY ST → PIN CITY ST

↑ PF
Key

PIN ∪ (PIN ST) ≐ CITY ∪ (PIN ST)

⇒ (PIN ST) = CITY ⊄ PIN ST

↑
Key

Scanned by CamScanner

Inference Rule

1) $\{X \to Y, X \to Z\} \Rightarrow X \to YZ$

2) $\{X \to Y, WY \to Z\} \Rightarrow XW \to Z$

Pseudo Transitivity Rule

3) If $X \to Y$ and $Z \subseteq Y$ Then $X \to Z$

4) $\{X \to YZ\} \Rightarrow X \to Y, X \to Z$

$R(A, B, C, G, H, I)$

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

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

Closure of a set of Attributes

Let F be a set of FDs, The closure of attribute X denoted by X^+ is a set of attributes A, s.t. $X \to A$ can be derived from A using Armstrong's Axioms

Lemma: $X \to Y$ follows from Armstrong's Axiom, if and only if $Y \subseteq X^+$

Th: Armstrong's Axioms are sound and complete

→ Sound: If $X \to Y$ from F, Then it's true in all cases.

Complete: The set of FDs obtained by Armstrong's axioms are all the possible FDs.

Scanned by CamScanner

Page No.:
Date:
YOUVA

X^+ algorithm of Bernstein

1) Let $N = 0$ & $X(N) = X$

2) If $\exists$ a dependancy $A \to B$ in F then left-hand side A is contained in $X(N)$, while right-hand side B is not Then,

$$X(N+1) = X(N) \cup B$$

3) $N = N+1$, go to step 2

Eg: $R(A, B, C, D, E, G)$

$$F = \{AB \to C,\ C \to A,\ BC \to D,\ ACD \to B,\ D \to EG,\ BE \to C,\ CG \to BD,\ CE \to AG\}$$

$(BD)^+ = ?$

→

$X(0) = BD$

$D \to EG$, $X(1) = BD \cup EG = BDEG$

$BE \to C$, $X(2) = BDEGC$

$A \to C$, $X(3) = BDEGCA$

$$\boxed{(BD)^+ = BDEGCA}$$

20/01/20

R{ FI (Familiname), MOC (mode of com), SAL (salary), SS (Spouse salary), ~~FI~~, JOB, COR (city of Residence), LUCK, ED }

effort { EFF, SEN (Sincerity) }

Scanned by CamScanner

{ FI → MOC, SAL , SS → FI , JOB FE → COR ,
LUCK ED → JOB , JOB → ED ,
EFF SEN JOB → SAL }

- To find Key, we have to find min set of attributes whose closure, if it gives you all the attributes we say that it is a key.

{JOB, EF, SEN, LUCK} etc. are not appearing on the RHS at all, so they must be a part of the Key. So, find the closure of these

Simplification of FDs :

- $X \to Y$ is a non trivial FD if $Y \not\subseteq X$
- If ∄ no $X' \subseteq X$ where $X' \to Y$ Then $X \to Y$ is called an elementary FD.
  Eg → $AB \to C$, $B \not\to C$, $A \not\to C$
  i.e. If I supply A & B, I'll get C.

- If ∃ some $X' \subset X$ and $X' \to Y$, Then ∃ partial dependency on that FD.
  Eg → $AB \to C$, $A \to C$, here supplying B was not necessary so its called as "extraneous attribute" while $AB \to C$ is called as partial dependency.

- Let $X \to A$, $B \subseteq X$, if $\{X - B\} \to A$ is in F^+, Then we can say that B is an extraneous attribute of A and is not required.

Scanned by CamScanner

Page No.:
Date:
YOUVA

- Let $F = \{AB \to DEF, AC \to G, A \to C\}$

$AC^+ = ACG$

$A^+ = ACG$

So in the FD $AC \to G$, C is extraneous attr, so you can write it down as

$F = \{AB \to DEF, A \to G, A \to C\}$

- Each FD is represented by a table, so if we remove an attr. without losing the properties, then we are reducing the size of the table and hence, processing time.
- In the prev. case, we tried to find if an attr. is redundant, now we try to ~~prove~~ find out if an FD is redundant.

Let an FD f is redundant in a set of FDs 'F'

$\Rightarrow (F - f)^+ = F^+$

i.e. by removing f^+, we should be able to obtain F^+.

- A set of FDs 'F' is 'minimum' if $\nexists$ no set G with less no of FDs than F s.t. $G^+ = F^+$

$\{A \to B, A \to C\}$ and $\{A \to BC\}$

- A set of FDs 'F' is 'L-minimum' if

1) F is min

2) $\nexists$ no partial dependency in each FD.

- A set of FDs 'F' is 'LR-minimum' if

1) F is L-minimum

2) $\nexists$ no redundancy on RHS

Scanned by CamScanner

21/01/20

G{AB → C, C → A, BC → D, ACD → B, D → EG, BE → C, CG → BD, CE → AG} simplify.

→

AB → C	N.R.	BE → C	N.R.	R : Redundant
AC → A	N.R.	CG → B	N.R.	NR : Non-Redundant
BC → D	N.R.	× CG → D	R	
× ACD → B	R	× CE → A	R	
D → E	N.R.	CE → G	N.R.	
D → G	N.R.			

$(ACD)^+ = ACDEGB$ (All elements)

$D^+ = DE$; $(CG)^+ = CGADE$

Allowed minimum :

{AB → C, C → A, BC → D, D → EG, BE → C, CG → B, CE → G}

×

Decomposition of a Relational scheme

$R(A_1, A_2, \ldots, A_n)$

Obtain, $P = \{R_1, R_2, \ldots, R_n)$ s.t.

$R = R_1 \cup R_2 \cup \ldots \cup R_n$

Lossless Join

R $R_1, R_2, \ldots, R_K$ D

$r = \Pi_{R_1}(r) \bowtie \Pi_{R_2}(r) \bowtie \Pi_{R_3}(r) \bowtie \Pi_{R_4}(r) \bowtie \ldots \bowtie \Pi_{R_K}(r)$

Scanned by CamScanner

If

Th: If $\rho = (R_1, R_2)$ is a decomposition of R then ρ is a lossless join decomposition w.r.t FDs iff

$$(R_1 \cap R_2) \longrightarrow (R_1 - R_2)$$

or

$$(R_1 \cap R_2) \longrightarrow (R_2 - R_1)$$

Eg: $R(A, B, C)$ $\quad F = \{A \to B\}$

(i) $R_1(A, B)$, $R_2(A, C)$

$R_1 \cap R_2 = A$; $R_1 - R_2 = B$

$\Rightarrow$

$R_1 \cap R_2 \to R_1 - R_2$

$\therefore \rho = (R_1 R_2)$ is lossless

(ii) $R_1(A, B)$ $\quad R_2(B, C)$

NOT lossless.

Eg: $r = \{a_1 b_1 c_1, a_2 b_1 c_2\}$

$R_1 = \Pi_{AB}(R) = \{a_1 b_1, a_2 b_1\}$

$R_2 = \Pi_{BC}(R) = \{b_1 c_1, b_1 c_2\}$

$r \neq R_1 \bowtie R_2 = \{a_1 b_1 c_1, a_1 b_1 c_2, a_2 b_1 c_1, a_2 b_1 c_2\}$

Preservation of Dependencies

$R \quad \to \quad R_1, R_2, \ldots, R_K$

$\downarrow F \quad \downarrow F_1 \quad \downarrow F_2 \quad \downarrow F_K$

$F = F_1 \cup F_2 \cup \ldots \cup F_K$

Scanned by CamScanner

Page No.:
Date:
YOUVA

Note:

- Lossless join property must be satisfied.

$R = (C, S, P)$

~~$R_1 (S, P)$~~

$\{CS \rightarrow P \quad , \quad P \rightarrow C\}$

$R_1 (S, P) \quad , \quad R_2 (C, P)$

$F_1 = \phi \quad , \quad F_2 = \{P \rightarrow C\}$

- **Super Key** :- Super key is the combination of fields by which row is uniquely identified. If you add any column or attribute to a primary key, then it becomes a set of super key.

- **Candidate Key** :- Candidate Key is the minimal supa key These are individual columns in a table that qualifies for uniqueness of all rows.

- **Foreign Key** :- Foreign key is a collection of field or fields in one table, that uniquely identifies a row of another table. Thus foreign key is defined in a second table, but it refers to the primary key in the first table.

Eg Employee { E-id, Fullname, SSN, Dept-ID }

{E-id, Fullname}
{E-id, Dept ID}

Scanned by CamScanner

Normal Forms

Boyce Codd Normal Form (BCNF)

- Whenever $X \to Y$, and Y is not contained in X and X is a key.

Eg:- R {PAN, PI, DI, DRUG, QTY, COST}

F = { PAN → PI, PI → DI, PI DRUG → QTY, DRUG QTY → COST }

→ Key : {(PAN, DRUG)}.

PAN → PI

R_1 (PAN, PI) ✓

R_2 (PAN, DI, DRUG, QTY, COST)

DRUG QTY → COST

R_3 (DRUG, QTY, COST) ✓

R_4 (PAN, DI, DRUG, QTY)

PAN → DI

R_5 (PAN, DI) ✓

R_6 (PAN, DRUG, QTY) ✓

PAN DRUG → QTY

23/01/22

- Th: BCNF decomposition always produces lossless joint decomposition.
- After BCNF we might not be able to answer certain queries, while those which we can will have correct information. i.e. it may or may not preserve dependencies.

Scanned by CamScanner

Q. $R(A, B, C, D, E, F)$

$F = \{ C \to A,\ AE \to B,\ BF \to C,\ CD \to EF,\ EF \to AD \}$

→

Key : {CD or EF}

$R_1(A, C)$, $R_3(B, F, C)$, $R_5(B, D, E, F)$

Prime Attribute

- Any attribute A is called as a prime attribute if its a member or subset of a ~~the~~ key. Otherwise its non-prime attribute.
- Key : {AB, BC}. A, B, C are prime attributes.

Third NF (3NF)

- $X \to A$ A is not a subset of X, then X is a key of the relation or A is prime attribute.
- If a relation is in BCNF, it ~~defin~~ certainly is 3NF.
- It removes two things — ① Partial Dependency and Transitive Dependency.

Bernstein's 3NF Algorithm

① Rewrite the FDs s.t. each FD contains only one attribute on the RHS.

② Rewrite the list of FDs s.t. ∃ no redundant FDs.

③ Rewrite FDs s.t. no proper subset of LHS functionally determines RHS. This ensures no partial dependencies.

Scanned by CamScanner

Date: YOUVA

④ Combine dependencies with same LHS using union rule. Then if $X \to Y$ is a FD make a decomposition $\{X, Y\}$.

⑤ If key is not present in any of the subrelation then create a new subrelation with key only.

⑥ If any subrelation is subset of other subrelation remove it.

Eg $R_1(PAN, PI)$, $R_2(PI, DI)$, $R_3(PI, DRUG, QTY)$
$R_4(DRUG, QTY, COST)$, $R_5(PAN, EDRUG)$

Note:- In 3NF the result is unique as it is structured.

- 3NF ensures lossless join as well as preserves dependency

Eg: $R(A, B, C, D, E, F, G, H, I)$

$\{A \to BCD, AE \to FG, F \to AEG, C \to HI\}$

→ Key: AE, F

$R_1(A, B, C, D)$, $R_2(C, H, I)$, $R_3(A, E, F, G)$

27/01/22

Scanned by CamScanner

Date

Scanned by CamScanner

Date

23/01/20

Query Languages

Relational Algebra

① Union (∪) :- R ∪ S , The set of tuples which are in either in R, or in S, or in both of them.

R

A	B	C
a	b	c
d	a	f
c	b	d

S

D	E	F
b	g	a
d	a	f

R ∪ S

a	b	c
d	a	f
c	b	d
b	g	a

- Checks the tuple & not the attribute name.
- R & S must have same number of attributes.

Scanned by CamScanner

② Set Difference :- (−) . R−S.

- The set of tuples which are in R but not in S.

R − S (R & S same as prev.)

a	b	c
c	b	d

- R & S must have same no. of attributes.

⑤ Cartesian Product :- (×) R×S (R,S same as prev)

- Number of attributes is known as arity.
- Let R & and S be tables of arity K_1 & K_2. In R×S, The first K_1 components come from a tuple of R & last K_2 from a table of S.
- Here column names will appear.

A	B	C	D	E	F
a	b	c	b	g	a
a	b	c	d	a	f
d	a	f	b	g	a
d	a	f	d	a	f
c	b	d	b	g	a
c	b	d	d	a	f

④ Projection (Π) :- $\Pi_{i_1, i_2, \ldots, i_K}(R)$

- This will take out 'K' attributs values from each tuple of R.
- Eg:- $\Pi_{C,A}(R)$ =

C	A
c	a
f	d
d	c

$\Pi_{2,3}(S)$ =

E	F
g	a
a	f

Scanned by CamScanner

Date

⑤ Selection (σ) :- Suppose F is a formula.

- $\sigma_F(R)$ means from the relation R we will take only those tuples which will satisfy the formula F.
- Eg :- $\sigma_{B='b'}(R) \equiv$

A	B	C
a	b	c
c	b	d

; $\sigma_{2>3}(R) \equiv$

A	B	C

Empty

$\sigma_{1='a' \vee 1='b'}(R) \equiv$

A	B	C
a	b	c

Either a or b

Additional Operators

① Intersection ($\cap$) :- $R \cap S$:

- Set of tuples that are both in R and S.

$R \cap S = R - (R - S)$

② Quotient ($\div$) :- $R \div S$. Consider arity of R and S as r & s resp. s.t. $r > s$.

$R \div S$ is the tuple set of tuples t with arity r-s s.t. $\forall$ tuples u is S, the tuple tu is in R.

R

a	b	c	d
a	b	e	f
b	c	e	f
e	d	c	d
e	d	e	f
a	b	d	e

S

c	d
e	f

$R \div S$

a	b
eb	d

$$R \div S = \pi_{1,2,\ldots,r-s}(R) - \pi_{1,2,\ldots,r-s}\left((\pi_{1,2,\ldots,r-s}(R) \times S) - R\right)$$

Scanned by CamScanner

③ Θ-Join :- $R \bowtie_{i\,\theta\,j} S$, here i & j are attributes of R & S resp. We first find the cartesian product of R and S, and then takes out those tuples which satisfy $i \theta j$ formula.

Eg → R

A	B	C
1	1	3
4	5	6
7	8	9

S

D	E
3	1
6	2

$R \bowtie_{B<D} S$

A	B	C	D	E
1	1	3	3	1
1	1	3	6	2
4	5	6	6	2

④ Natural Join ($\bowtie$) :- $R \bowtie S$

* • Here R and S must have column/ attr. names.
  • If the attr. name of two columns are same it will merge those two and output only those tuples which have same values in those two attributes.

Eg:- R

A	B	C
a	b	c
d	b	c
b	b	f
c	a	d

S

B	C	D
b	c	d
b	c	e
a	d	b

$R \bowtie S$

A	B	C	D
a	b	c	d
d	b	c	e
a	b	c	e
d	b	c	e
c	a	d	b

Examples :- (Tables to be used in this course)

Customer (C_name, street, c_city)
Branch Deposit (b_name, b_city, asset)
Borrow (b_name, loan_no, c_name, amount)
Client (c_name, emp_name)
Deposit (b_name, ac_no, c_name, balance)

Scanned by CamScanner

Q1 : Find the customer name, where customer name and the employee name who handle the customer are same.

→

$$\Pi_{c\text{-name}}\left(\sigma_{c\text{-name} = emp\text{-name}}(Client)\right)$$

Q2 : Find all clients of employee XYZ, and corresponding cities of those clients

→

$$\Pi_{client.c\text{-name},\ c\text{-city}}\left(\sigma_{emp\_name = 'xyz' \wedge client.c\text{-name} = customer.c\text{-name}}(client \times customer)\right)$$

Q3 : Find all customers who have both account & loan at KGP branch.

→

$$\Pi_{c\text{-name}}\left(\sigma_{b\_name = 'KGP'}(Depsit \bowtie Borrow)\right)$$

Q4 : Find all customers who have account at all branches in the city Kolkata.

→

$$\Pi_{c\text{-name},\ b\_name}(Deposit) \div \Pi_{b\text{-name}}\left(\sigma_{b\_city = 'Kolkata'}(Branch)\right)$$

30/01/20

- Employee (Emp_id, Name, Salary, manager_id)
  - If manager_id = 1 ⇒ employee is a manager otherwise not a manager.

Scanned by CamScanner

Q. Display the name of employees (not manager) whose salary is more than any manager.

→

$$\Pi_2 \left(\sigma_{4 \neq 1 \wedge 5 = 1 \wedge 3 > 7} (\text{Employee} \times \text{Employee}) \right)$$

Here 4 → EmployeeA.manager_id
2 → EmployeeB.manager_id

Lives (p_name, street, city)
Works (p_name, c_name, salary)
Located_in (c_name, c_city)
Manages (p_name, m_name)

Q. Find the name and city of all persons who work for 'XYZ' company

→

$$\Pi_{p\_name} \left(\sigma_{c\_name = 'XYZ'} (\text{Lives} \bowtie \text{Works}) \right)$$

OR.

$$\Pi_{\text{Lives.p-name}} \left(\sigma_{c\_name = 'XYZ' \wedge \text{Lives.p-name} = \text{Works.p\_name}} (\text{Lives} \times \text{Works}) \right)$$

Q. Find all persons who live in the same city and the company they work for.

→

$$\Pi_{1} \sigma_{3=8 \wedge 4=9} \left((\text{Lives} \bowtie \text{Works}) \times (\text{Lives} \bowtie \text{Works}) \right)$$

Q. Find all persons who live in the same city & street, as their manager

→

$$\Pi_{1} \left(\sigma_{1=5 \wedge 2=6 \wedge 4=7 \wedge 5=8} (\text{Manages} \times \text{Lives} \times (\Pi_2 (\text{Manages}) \bowtie \text{Lives})) \right)$$

Scanned by CamScanner

3/02/20

Page No.
Date
Youva

Q Find all persons whose salary is less than every employee of 'xyz' company.

→

$$\Pi_1 (\sigma_{1>2} (P \times P)) = Q$$

$$\boxed{\text{Min} = P - Q}$$

Propositional Logic

$\neg$ (not), $\wedge$ (and), $\vee$ (or), $\Rightarrow$ (implication), $\equiv$ (equivalence)

- Proposition → is a declarative statement that's either true or false., made up of symbols, known as atoms.
- Any expression representing a proposition using atoms is called a formula

Tuple Relational Calculus

- An expression in T.R.C. is written as $\{t \mid \Psi(t)\}$ where t is a tuple variable, $\Psi(t)$ is a formula
- $R(s) \leftarrow s$ is a tuple of relation R
  e.g. → $n[i] \Theta m[j]$
- Universal Quantifier ($\forall x$)
  - $(\forall x)\, G(x)$ means that $G(x)$ is true for all x.
- Existential Quantifier ($\exists x$)
  - $(\exists x)\, G(x)$ then means there exist at least one value of x s.t. $G(x)$ holds.

Scanned by CamScanner

Page No :
Date:

Eg. → $(\exists s)(R(s))$

- $R \cup S$ is T.R.C. → $\{t \mid R(t) \cup S(t)\}$
- $R - S$ is $\{t \mid R(t) \wedge \neg S(t)\}$
- $R \overset{\omega}{\times} S$ is $\{t^{(\lambda+s)} \mid \{(\exists u)(\exists v)(R(u) \wedge S(v) \wedge t[1] = u[1] \wedge \ldots \wedge t[\lambda] = u[\lambda] \wedge t[\lambda+1] = u[1] \wedge \ldots \wedge t[\lambda+s] = u[s]\}$
- $\pi_{i_1, i_2, \ldots, i_\eta}(R)$ is $\{t^{(\kappa)} \mid (\exists x)(R(x)) \wedge t[i_1] = x[i_1] \wedge \ldots \wedge t[i_\kappa] = x[i_\kappa]\}$
- $\sigma_F(R)$ is $\{t \mid R(t) \wedge F'\}$ ← tuple version of F

Th : Every F is relational algebra can be written in tuple algebra.

Eg → $R^{(2)}, S^{(2)}$ $\quad \pi_{1,4}(\sigma_{2=3}(R \times S))$

$\{t \mid (\exists u)(\exists v)(R(u) \wedge S(v) \wedge u[2] = v[1] \wedge t[1] = u[1] \wedge t[2] = u[2])\}$

Scanned by CamScanner

4/02/20

Q1 Find the ~~branch~~ name, ~~total~~ loan no., customer name and amounts for loans more than 10K. (Bank Database)

→ ~~Bank Database.~~

$\{ t \mid t \in Borrow \wedge t[amount] > 10000 \}$

Q2 Find all customers having a loan at iit branch and the cities where they live.

→

$\{ t \mid \exists u \wedge \exists v \ (u \in Borrow \wedge v \in Customer \wedge u[1] = 'IIT' \wedge v[1] = u[3] \wedge t[1] = v[1] \wedge t[2] = v[3]) \}$

$\{ t \mid \exists u (u \in Borrow \wedge u[1] = 'IIT' \wedge \exists v (v \in Customer \wedge u[3] = v[1] \wedge t[1] = v[1] \wedge t[2] = u[3]) \}$

Better Time complexity

Q3 : Find all customers having a loan or account or both at IIT branch.

→

$\{ t \mid \exists u (u \in Borrow \wedge u[1] = "IIT" \wedge t[1] = u[3]) \vee \exists v (v \in Deposit \wedge v[1] = "IIT" \wedge t[1] = v[3]) \}$

Q4 : Find all customers who have account at all branches in the city KGP.

→

$\{ t \mid \nexists u (u \notin Branch \vee u[b\_city] \neq 'KGP' \vee \exists v (v \in Deposit \wedge u[1] = v[1] \wedge t[1] = v[1])) \}$

Scanned by CamScanner