Date No

Armstrong Axioms. (functional dependencies)

- Can be used to identify all functional dependencies that exist.

- Armstrong Axiom has 3 rules and 4 Secondery rules.

3 primary rules

- Reflexivity
- Augmentation
- Transitivity.

- Reflexivity.

if AB is a relationship also an subset of of ABC. in that case it can say ABC functionally determing AB.

- AB ⊆ ABC = ABC → AB

(⊆ : Subset)

- Augmentation.

if we have a functional dependency called A → B We can add an some attribute to both side.

A → B = AC → CB

attribute C added to both side.

- Transitivity.

if $A \rightarrow B$ and $B \rightarrow C$, in that case we can say $A \rightarrow C$.

$$A \rightarrow B,\ B \rightarrow C \quad = \quad A \rightarrow C$$

4 Secondary rules.

- Union
- Composition
- Decomposition
- Pseudo Transitivity.

- Union.

if $x \rightarrow y$ and $x \rightarrow z$, then we can say $x \rightarrow yz$.

$$x \rightarrow y,\ x \rightarrow z \quad = \quad x \rightarrow yz$$

- Composition.

if $x \rightarrow Y$ and $A \rightarrow B$ then the combination of xA determine YB

$$x \rightarrow Y,\ A \rightarrow B \quad = \quad xA \rightarrow YB$$

- Decomposition.

Simply the opposite of union.

$$x \rightarrow yz \quad = \quad x \rightarrow y,\ x \rightarrow z$$

only seperate L, H, S

Date No

- Pseudo Transitivity.

if $x \to Y$ and $YZ \to W$. then we can say $xz \to W$.

$$x \to Y \;,\; YZ \to W \quad = \quad xz \to W$$

transitivity

dep. so replace with x

Attribute closure. $[x]^+$

. Set of attributes that could be determined.

notation $[x]^+$.

if $R(A, B, C, D, E)$ and FD $\{A \to B \;,\; B \to C \;,\; C \to D \;,\; D \to E\}$.

find A and AB .

A →

$A \to A$ (reflexivity)
$A \to B$ (given)
$A \to C$ (Transitivity)
$A \to D$ (Transitivity)
$A \to E$ (Trans)

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

Date No

BC →

B → C there for BC → C (reflexivity)

BC → D (Union)

BC → E (Transitivity)

∴ BC+ = {C, D, E}

Activity 2.1

R (A, B, C, D) F = {A → B, B → C, C → D}

A → A (ref)

B → B (ref)

C → C (ref)

D → D (ref)

A → B (given)

B → C (given)

C → D (given)

A → C (trans)

A → D (trans)

A → BC (union)

A → AB (union)

Activity 2.1

R (A, B, C, D) F = {A → B, B → C, B → D}

A → A (ref)
B → B (ref)
C → C (ref)
D → D (ref)
A → B (giv)
B → C (giv)
B → D (giv)
A → C (trans)
A → D (trans)
A → ABCD (union) → Pk
B → BCD (union)
~~C → C~~

Activity 2.2.

R (A, B, C, D, E) F = {A → B, A → C, CD → E, B → D, E → A}

A → A
B → B
C → C
D → D
E → E
A → D (trans)
A → C (trans)
A → CD (union)
A → E (trans)

A → ABCDE (union) → R, k
E → ABCDE (tran union) → R, k
CD → ABCDE (tran union) → k
BC → CD = ABCDE
BC = ABCDE (trans) → k

Lecture 03

Normalization

refining the Schema.

might have,

- Data redundency
- Data Anomalies. (insert / Delet / Update)

to Solve these We can do Decompose the relation.

Decompose - Separating to Smaller relations,

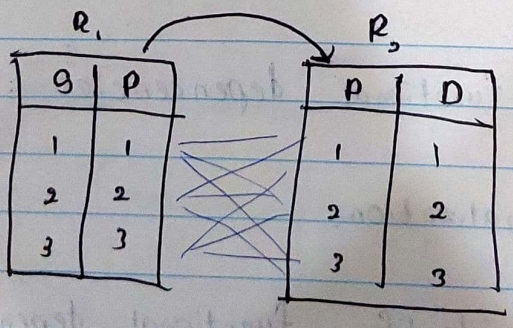
Decompose Methods (2)

- Loss-less join property.
- Dependency preserving property.

Depomposed relation should have Loss-less and Dependency preserving properties.

T_1

S	P	D
1	1	1
2	2	2
3	3	3



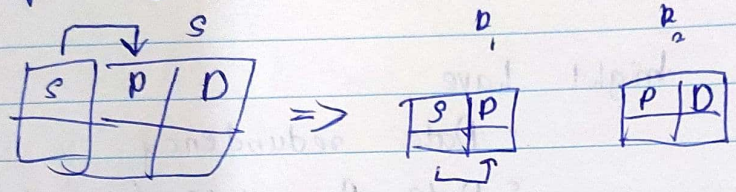
P in R_2 like FK

- T_1 table devided in to R_1 and R_2, by joining R_1 and R_2 you should able to restore (T_1).

S — P — D

Dependency Preserving.

After decompose you should able to find dependency on relational as well.



functional dependency

$x \rightarrow y$

- x functionally determines y
- y functionally dependent on x.

if you know 'x' you can retrive y.

Normalization.

- Way of guid to show how to decomposite a relation.
- fuectional dependencies causes redundency.

Step of Normalization.

- identify functional dependencies given to us.
- identify all keys in relation
- Normaliz the relation

Stages of Normalization

you have to add each and every table to Normalizatine

① Unnormalized
(UDF)

↓ Remow o repeating group: (multivalue, composit Attributes)

② First Normal Form
(1NF)

↓ Remove Partial dependencies

③ Second Normal form
(2NF)

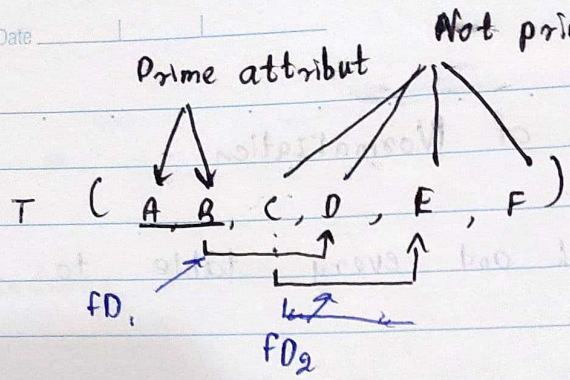
↓ Remove transitive dependencies.

④ Third normal form → Need to do Unlitl. 3NF
(3NF)

↓ Remove a remaining functional dependency anomalies.

⑤ Boyce-codd normal form
(BCNF)

1.26



FD1 = B → D

FD2 = C → E

① Partial functional dependency

Non prim attribute depend on part of the primary key.

ex : FD1 is a partial dependant.

② Default functional dependency

. occurs from the primary key, this not cause redundancy, by Pk.

ex : {A,B} → C, D, E, F.

③ Transitive functional dependency

FD2 = C → E

C → E

A,B → C

A,B → E

Non prime attribute transitivly depends on the A,B prim key

① 1st Normalization form

- remove all repeating, multivalues.

ex : User (Tp_no , ID , name)

- Separate multivalu. to another table.

User (ID , name)

↑

user_Tp (ID , Tp_no)

② 2nd Normalization form

- Remove all partial functional dependencies.

ex : R2 (x, y , z , m , l)

↓

Separate prime attribut and non prim attribut that having Partial dependency

R2 (x, y , z , m , l)

R2A (y , m)

↑

R2B (x, y , z , l)

③ 3rd normalization form

- Remove all transitive functional dependency

ex : R3 (x , y , z , m)

z → m
x → z

↓

Bring ~~PK and~~ non Attributes to Separate table. make appointer PK.

R3A (z , m)

R3B (x , y , z)