# UNIT - V

## FUNCTIONAL DEPENDENCY:

Functional dependencies play a key role in differentiating good database designs from bad database designs.

A functional dependency is a Type of constraints.

Consider a relation schema R and let $\alpha \subseteq R$, and $\beta \subseteq \alpha$, Then the Functional dependency is $\alpha \rightarrow \beta$

Consider the schema.

loan-info = ( loan-no, Branch-name,

We can the reconstruct the Lending Schema using

branch - customer ⋈ customer - loo

| branch-name | branch-city | assets | customer-name | loan-no |
|---|---|---|---|---|
| Downtown | Brooklyn | 90000 | Jones | L-17 |
| Downtown | Brooklyn | 90000 | Jones | L-11 |
| Redwood | Palo alto | 21000 | Smith | L-23 |
| Perryridge | Horseneck | 90000 | Jackson | L-14 |
| Mianus | Horseneck | 80000 | Jones | L-11 |
| Mianus | Horseneck | 80000 | Jones | L-17 |
| Northtown | Rye | 37000 | Hayes | L-16 |

(2)

Decomposition Rule :-

If $\alpha \rightarrow \beta\gamma$ holds then $\alpha \rightarrow \beta$ holds and $\alpha \rightarrow \gamma$ holds.

Pseudo Transitivity Rule :

If $\alpha \rightarrow \beta$ holds and $\gamma\beta \rightarrow \sigma$ holds then $\alpha\gamma \rightarrow \sigma$ holds

The functional dependency $A \rightarrow H$ is logically implied

$A \rightarrow H$

$A \rightarrow B$ & $A \rightarrow H$.

$A \rightarrow H$ using transitivity rule

Union Rule :

$A \rightarrow B$

$A \rightarrow C$

$A \rightarrow BC$ holds

and,

$C G \rightarrow H$

$C G \rightarrow I$

$C G \rightarrow HI$ This is an example of union rule

Because a given loan can be made to more than one customers

| Prof | Course | Roone | Max-enrollment | Day | Time |
|---|---|---|---|---|---|
| Smith | 353 | A532 | 40 | Mon | 11:45 |
| Smith | 354 | A532 | 40 | Wed | 11:45 |
| Clark | 365 | H940 | 300 | Tue | 1:45 |
| Clark | 422 | H940 | 300 | Thurs | 2:10 |
| Turner | 456 | B278 | 45 | Mon | 3:15 |
| Turner | 457 | B278 | 45 | Wed | 4:50 |

Schedule relation.

determinate key
F = Course -> Prof
dependent key

Closure of a set of functional Dependencies:

The set of function dependencie

that is logically implit by called by F

closure of F and it Returns of $F^+$

Set of FD -> F

closure of FD -> $F^+$

Suppose we are given a relational Schema R = (A, B, C, G, H, I) and the set of functional dependencies

A -> B
A -> C
CG -> H
CG -> I
B -> H ①

## Armstrong's Axioms :-

### Reflexivity Rule :

If $\alpha$ is set of attributes & $B \subseteq \alpha$ then $\alpha \rightarrow B$ holds.

### Augmentation Rule :-

If $\alpha \rightarrow B$ holds and $\gamma$ is set of attributes then $\gamma\alpha \rightarrow \gamma B$ holds.

### Transitivity Rule :

If $\alpha \rightarrow B$ holds & $B \rightarrow \gamma$ holds

Then $\alpha \rightarrow \gamma$ holds.

## Additional Rules :-

### Union Rule :-

If $\alpha \rightarrow B$ holds and

Then $\alpha \rightarrow B\gamma$ holds

2) Consider the following query find all bank branches that have made a loan in an amount $\leq 1000$.

The result of this query using the Lending relation is

Mianus.

The result of the same query using the relation branch ⋈ customer ⋈ customer_loan is

| branch_name |
| --- |
| Downtown |
| Mianus |

4) Boyce - codd Normal form (BCNF):

Every determinant is act as a candidate key. It is stronger version of 3NF.

5) Fourth Normal form (4NF):

No multivalued dependency.

6) Fifth Normal form (5NF):

No join dependency. (divide to multiple)

Example:

First, Second & Third Normal forms :- ) ③

Alpha Book house

Pune - 413000 1.

Customer name : Beta School of Computer Science

customer NO : 1052

Address : Sivaji Nagar, Pune - 01.

| ISBN | Book Title | Author's name | Author's country | qty | Unit - Price (RS) | Amount (RS) |
|---|---|---|---|---|---|---|
| 81-203-5 | DOS | P.K. Sinha | India | 5 | 250 | 1250 |
| 0-112-6 | DBMS | Korth | U.S.A | 6 | 300 | 1800 |
| 1-215-9 | Simulation | Gordan | U.S.A | 5 | 100 | 500 |
| | | | Grand Total | | | 355 |

D) Invoice ( cust_no, cust_name, cust_add,

( ISBN, Title, Author_name, Author_country, qty, Unit Price ))

The decomposition of lending relation given below is lossless decomposition.

Branch_Schema = ( Branch_name, Branch_city, Assets)

Loan_Info_Schema = ( Branch_name, customer_name, loan_amount) (2)

(3) Multivalued dependencies:

Consider an unnormalized relation that contains information about Vehicles, dealers, and Models.

Each Tuple contains Vehicle name, dealer's name and Model name.

| Vehicle | Dealer | Model |
|---|---|---|
| Maruti Car | Competent Agarl | Maruti 800 |
| | Link motor | Baleno |
| Scooter | ABC | Bajaj |
| | | RMH. |

**Decomposition :-**

we can avoid update anomalies by decomposition of the original Relation

The relation Schema R is decomposed into following relation Schema's $(R_1, R_2 \ldots R_n)$ in such a way that

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

**Example :**

Consider following Lending Schema

= ( Branch-name, Branch-city, Assets, customer-name, Loan-number, amount )

The Lending Schema is decomposed into following Two relations

Branch-customer-Schema = ( Branch-name, Branch-city, Assets, Customer-name)

Customer-loan-Schema = ( Customer-name, loan-number, amount )

The new relations Branch-customer-Schema and customer-loan-Schema are constructed using the Lending Schema as,

| Vehicle | Dealer | Model |
|---|---|---|
| Maruti car | Competent | Maruti 800 |
| Maruti car | Competent | Baleno |
| Maruti car | Agani | Maruti 800 |
| Maruti car | Agani | Baleno |
| Maruti car | Link motor | Maruti 800 |
| Maruti car | Link motor | Baleno |
| Scooter | ABC | Bajaj |
| Scooter | ABC | LML |

| Vehicle | dealer |
|---|---|
| Maruticar | competent |
| Maruticar | Agani |
| Maruticar | Link motor |
| Maruticar | ABC |
| Scooter | |

| vehicle | model |
|---|---|
| Maruticar | Maruti 800 |
| Maruticar | Baleno |
| Scooter | Bajaj |
| Scooter | LML |

Vehicle - Dealer

Vehicle - model

R.A -> -> R.B | R.C

R -> relation ; A -> vehicle, B -> dealer,
C -> model

Vehicle - dealer - model . vehicle ->
vehicle - dealer - model . dealer / vehicle - dealer - model . model .

21/02/12

## Normalization :

Normalization is an essential pa
of database design. A good understanding of the
symetics of data helps the designer to build
efficient design using the concept of normalizati

### Purpose of Normalization :-

Minimize redudancy in the
data (duplicates)

Remove insert, delete and
update anomaly during database activities

| branch-name | branch-city | Assets | customer-name | loan-number | amount |
|---|---|---|---|---|---|
| Dountown | Brooklyn | 90000 | Jones | L-17 | 1000 |
| Redwood | Palo alto | 21000 | Smith | L-23 | 2000 |
| Perryridge | Horesneck | 90000 | Jackson | L-14 | 1500 |
| Mianus | Horesneck | 80000 | Jones | L-11 | 900 |
| Northdown | Rye | 37000 | Hayes | L-16 | 1300 |

Branch-customer

| branch-name | branch-city | assets | customer-name |
|---|---|---|---|
| Dountown | Brooklyn | 90000 | Jones |
| Redwood | Palo alto | 21000 | Smith |
| Perryridge | Horesneck | 90000 | Jackson |
| Mianus | Horesneck | 80000 | Jones |
| Northdown | Rye | 37000 | Hayes |

1NF :

3) customer ( cust_no, cust_name, cust_add )

3) customer_Book ( cust_no, ISBN, Title, Author_name, Author_country, qty, Unit, Price )

2NF :

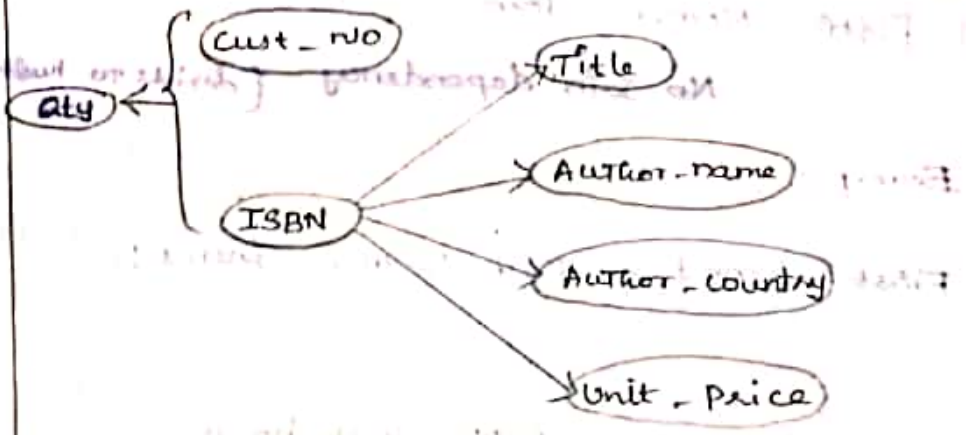


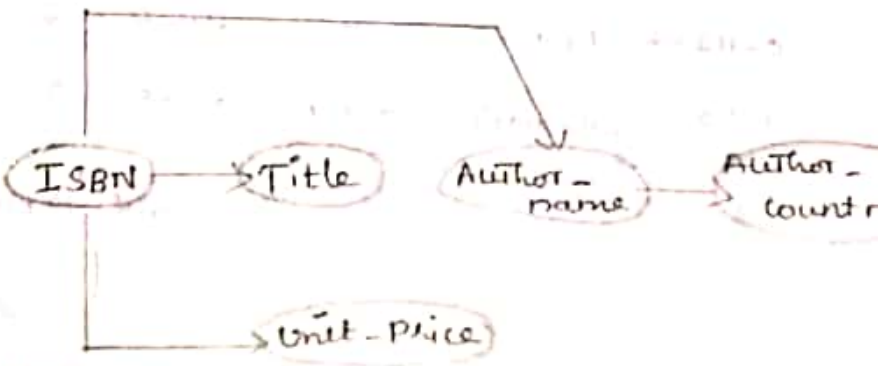

fig :- Depending diagram of relation-3

4) Sales ( cust_no, ISBN, Qty )

5) Book_Author ( ISBN, Title, Author_name, Author_country, Unit_Price,

3NF :-

Dependency diagram relation-5




6) Book ( ISBN, Title, Unit_price, Author_name