MANCHESTER
1824
The University of Manchester

FUNCTIONAL DEPENDENCIES

COMP23111 – Database Systems

OUTLINE

Functional Dependency – Definition

Functional Dependency – Advantages

Rules of Functional Dependencies

Types of Functional Dependencies

Functional Dependencies Closure – F^+

Attribute Closure – X^+

FUNCTIONAL DEPENDENCY
- DEFINITION

A Functional Dependency (FD) is a **constraint** between two sets of attributes in a relation from a database.

> FD is when one attribute (or set of attributes) precisely **determines** the value of another attribute in a database.

Given a relation R, where X,Y $\subseteq$ R: → subset

$X \rightarrow Y$

Determinant

Dependent

FUNCTIONAL DEPENDENCY
- ADVANTAGES

- Prevents data redundancy (data stored multiple times).
- Maintains quality of data (e.g. easy to update/delete data).
- Helps <u>determine</u> relations and constraints.

So, it helps avoid ~~**bad database designs**~~!!!

RULES OF FUNCTIONAL DEPENDENCIES *(ARMSTRONG'S AXIOMS)*

Reflexive: If Y is a subset of X, then X→Y.

Example: {seat_no, name} → name.

Augmentation: If X → Y is a valid dependency, then XZ → YZ is also valid.

Example: {seat_no, name} → movie_name, then {seat_no, name, movie_theatre} → {movie_name, movie_theatre}.

Transitivity: If X → Y and Y → Z are both valid dependencies, then X→Z is also valid.

Example: seat_no → movie_name & movie_name → movie__theatre, then seat_no → movie_theatre.

RULES OF FUNCTIONAL DEPENDENCY *(SECONDARY RULES)*

Union: If $X \rightarrow Y$ and $X \rightarrow Z$, then $X \rightarrow YZ$.

Decomposition: If $X \rightarrow YZ$, then $X \rightarrow Y$ and $X \rightarrow Z$.

Pseudo-transitivity: If $X \rightarrow Y$ and $EY \rightarrow Z$, then $XE \rightarrow Z$.

$X \rightarrow \cancel{Y}$

$E\cancel{Y} \rightarrow Z$

$= XE \rightarrow Z$

TYPES OF FUNCTIONAL DEPENDENCIES

Trivial FD: If X → Y and Y **is a subset** of X.

Example: {seat_no, name} → name.

Non-trivial FD: If X → Y and Y is **not a subset** of X (opposite of Trivial FD).

Example: {seat_no, name} → movie_theatre.

Multivalued FD: If X → {Y,Z} and Y and Z are **not dependent** on each other.

Example: seat_no → {name, age}, name → age and age → name not holding.

Transitive FD: If X → Y and Y → Z, then X → Z (as per the **transitivity** rule/axiom).

Example: seat_no → movie_name and movie_name → movie_theatre,
then seat_no → movie_theatre.

FUNCTIONAL DEPENDENCIES CLOSURE – F^+

F^+ is all the FDs that can be **derived**, given a set of FDs F.

By determining the closure of F, we can design normalised databases.

good database design!!!

ATTRIBUTE CLOSURE – X^+

X^+ of an attribute set X is a set of these attributes that can be functionally determined from X.

By determining the closure of X we can determine the keys for the relation.

1. **All attributes** must be functionally dependent on the **key**.
2. **Candidate** key should be **minimal** (primary key will be chosen from candidate keys).

ATTRIBUTE CLOSURE – *X^+ (EXAMPLE)*

R = {X, Y, Z, E, N}

$X \rightarrow Y$
$E \rightarrow Z$
$N \rightarrow X$
$N \rightarrow E$

$X^+ = \{X, Y\}$

$Y^+ = \{Y\}$

$Z^+ = \{Z\}$

$E^+ = \{E, Z\}$

$N^+ = \{N, X, E, Y, Z\}$

FD AND ATTRIBUTE CLOSURE *(EXAMPLE)*

Product_ID	Type	Name	Country_Of_Origin	Factory_ZIP_Code
1	Guitar	Fender Classic V1	US	99876
2	Guitar	Gibson M236	US	72331
3	Suitcase	Cardinal No25	Italy	04100
4	Hairdryer	Philips 1200	Netherlands	1095 MS

How do we find the FDs in this relation?

{Product_ID → Type, Product_ID → Name, Product_ID → Country_Of_Origin, Product_ID → Factory_ZIP_Code}

{Factory_ZIP_Code → Country_Of_Origin}

So, the FDs in this relation are: {Product_ID → Type, Product_ID → Name, Product_ID → Country_Of_Origin, Product_ID → Factory_ZIP_Code, Factory_ZIP_Code → Country_Of_Origin}

FD AND ATTRIBUTE CLOSURE *(EXAMPLE)*

Product _ID	Type	Name	Country_Of_Origin	Factory_ZIP_Code
1	Guitar	Fender Classic V1	US	99876
2	Guitar	Gibson M236	US	72331
3	Suitcase	Cardinal No25	Italy	04100
4	Hairdryer	Philips 1200	Netherlands	1095 MS

How do we find the PK in this relation?

PK $Product\_ID^+=\{Product\_ID, Type, Name, Country\_Of\_Origin, Factory\_ZIP\_Code\}$

$Factory\_ZIP\_Code^+=\{Factory\_ZIP\_Code, Country\_Of\_Origin\}$

Super Key

$(Product\_ID, Type, Name)^+=\{Product\_ID, Type, Name, Country\_Of\_Origin, Factory\_ZIP\_Code\}$

CK $(Product\_ID, Name)^+=\{Product\_ID, Type, Name, Country\_Of\_Origin, Factory\_ZIP\_Code\}$

CK $(Product\_ID, Type)^+=\{Product\_ID, Type, Name, Country\_Of\_Origin, Factory\_ZIP\_Code\}$