Fourth normal form: 4NF

Motivation - 4NF

- courses - tutors - reference books
- assumptions
  - for a course - any number of tutors and any numbers of books
  - a book can be used for any number of courses
  - a tutor can teach any number of courses
  - no link between tutors and books (!!!)

Un-normalised relation (CTB)

CTB

Course	Tutors	Books
Databases	M. Ursu	Introduction to DB
	M. Ursu	OO and Java
	M. Harman	DB Design
Languages	J. Kuljis	Pascal - How to ...
	M. Ursu	Programming in Prolog
		OO and Java
...	...	...

1N relation

Course	Tutor	Book
Databases	M. Ursu	Introduction to DB
Databases	M. Ursu	DB Design
Databases	M. Ursu	OO and Java
Databases	M. Harman	Introduction to DB
…	…	…
Languages	J. Kuljis	Pascal - How to ...
Languages	J. Kuljis	Logic Programming
Languages	J. Kuljis	OO and Java
...	...	...

Note

IF (c1, t1, b1) and (c1, t2, b2) exist
as tuples in the relation

THEN (c1, t1, b2) and (c1, t2, b1)
also exist in the relation

Question

- what normal forms is this relation in?
- helping question
  - what functional dependencies can you identify?

Shortcomings

- redundancy
- therefore: update anomalies
  - e.g. add a new tutor, C. Fox, for the Databases course: both tuples, (Databases, C. Fox, Introduction to DB) and (Databases, C. Fox, DB Design), have to be added
  - could you think of a better solution?
- this is a “problem” BCNF relation
- *what do you conclude about the studied normal forms*?

Common-sense solution

- separate *independent repeating groups*
  - decompose the un-normalised relation
  - bring it to 1NF
  - the solution:
    - shortcomings - removed
    - non-loss decomposition

Decompose the un-normalised relation

Course	Tutors
Databases	M. Ursu
	M. Harman
Languages	J. Kuljis
	M. Ursu

Course	Books
Databases	Introduction to DB
	DB Design
Languages	Pascal - How to ...
	Programming in Prolog OO and Java

Bring to 1NF

Course	Tutor
Databases	M. Ursu
Databases	M. Harman
Languages	J. Kuljis
Languages	M. Ursu

Course	Book
Databases	Introduction to DB
Databases	DB Design
Languages	Pascal - How to ...
Languages	Programming in Prolog
Languages	OO and Java

Solution

- what is the theoretical basis?
  - not FDs (there aren’t any)
  - multi-valued dependencies (MVDs)
    - generalisation of FDs
    - e.g. Course $\twoheadrightarrow$ Tutor
    - each course has a well defined **set** of tutors
    - in the relation CTB, Tutor depends on the value of Course **alone** (Book does not influence)

Multi-valued dependency - simple

- Let R be a relation and A and B arbitrary sets of attributes of R.

  There is a multi-valued dependency from A to B

  $A \twoheadrightarrow B$

  if and only each A value **exactly** determines a set of B values, independently of the other attributes

Multi-valued dependency - Date

- Let R be a relation; A, B and C are arbitrary sets of attributes of R

  A multi-determines B (B is multi-dependent on A)
  $A \twoheadrightarrow B$
  if and only if the set of B values matching an (A, C) pair depends only on the A value.

Multi-valued dependency
(Elmasri and Navathe, 2000, p. 514)

- Let R be a relation and X and Y arbitrary sets of attributes of R; if for any 2 tuples (in any extension) t_1 and t_2 with the property
  - $t_1[X] = t_2[X]$

  there exists 2 tuples t_3 and t_4 such that $(Z = R - (X \cup Y))$
  - $t_1[X] = t_3[X] = t_4[X]$
  - $t_1[Y] = t_3[Y]$ and $t_2[Y] = t_4[Y]$
  - $t_1[Z] = t_2[Z]$ and $t_3[Z] = t_4[Z]$

  then

  $X \twoheadrightarrow Y$

Trivial MVDs

- R, is a relation; X and Y arbitrary sets of attributes of R

  $X \twoheadrightarrow Y$ is trivial if and only if

  - $Y \subseteq X$ or
  - $X \cup Y = R$
- a trivial MVD does not represent a constraint

Questions

- is every FD a MVD?
- is every MVD a FD?

Example

Course	Tutor
Databases	M. Ursu
Databases	M. Harman
Languages	J. Kuljis
Languages	M. Ursu

Course	Book
Databases	Introduction to DB
Databases	DB Design
Languages	Pascal - How to ...
Languages	Programming in Prolog
Languages	OO and Java

Discussion

- R = (Patient, Disease, Doctor)
  - a patient has a number of diseases and, independently of them is assigned a number of doctors; therefore
    Patient ->> Disease | Doctor
  - each disease is treated by a certain number of doctors and, independently of this, there is a certain number of patients that suffer from each disease; therefore:
    Disease ->> Doctor | Patient
  - a patient, for a certain disease, is assigned only one doctor; therefore:
    (Patient, Disease) -> Doctor
    and **NO** MVDs exist

Discussion

- R = (Patient, Disease, Doctor)
  - each patient suffers from a number of diseases (independently of doctors) and each disease is treated by a number of doctors (independently of the patients that suffer from it); can we say that
    Patient ->> Disease and Disease ->> Doctor ?
    have we not already discussed the above case?

Discussion

- what about
  R = (Patient, Disease, Treatment, Nurse)
  where
  - a patient can have a number of diseases (independently of treatments and nurses)
  - a disease has a number of treatments (independently of patients and nurses);
  - a treatment is administered by a number of nurses (independently of patients and diseases).

Discussion

- what about
  R = (Patient, Disease, Doctor, Nurse)
  where
  - a patient has a number of diseases independently of doctors and nurses
  - a patient has a unique doctor for a given disease
  - a doctor has (works with) a number of nurses, independently of patients and diseases

Conclusion

- 2NF, 3NF and BCNF
    - based on FDs
- there are other constraints that can be expressed through projection
- multi-valued dependency
- 4NF

AWARD

UNIVERSITY	DISCIPLINE	DEGREE
Old Town	Computing	BSc
Old Town	Mathematics	PhD
New City	Computing	PhD
Old Town	Computing	PhD

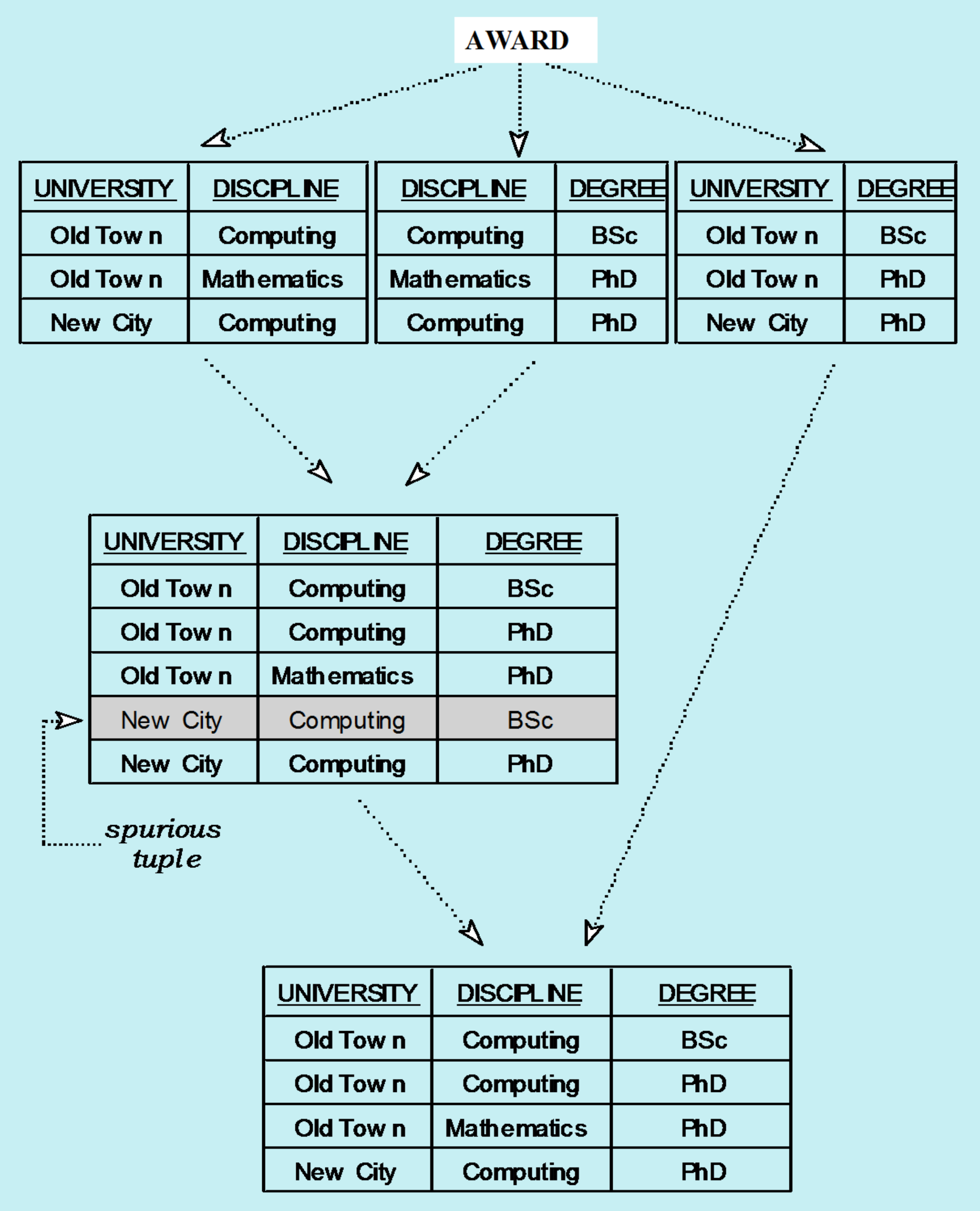
teaches (UNIVERSITY, DISCIPLINE)
is_read_for (DISCIPLINE, DEGREE)
awards (UNIVERSITY, DEGREE)

teaches (NewCity, Computing) = **true**
awards (NewCity, PhD) = **true**
is_read_for (Computing, BSc) = **true**

FROM
(NewCity *teaches* Computing) **and** (Computing *is_read_for* BSc)

IT DOES NOT FOLLOW
NewCity *awards* BSc *for_reading* Computing

: 4NF



AWARD

UNIVERSITY	DISCIPLINE
Old Town	Computing
Old Town	Mathematics
New City	Computing

DISCIPLINE	DEGREE
Computing	BSc
Mathematics	PhD
Computing	PhD

UNIVERSITY	DEGREE
Old Town	BSc
Old Town	PhD
New City	PhD

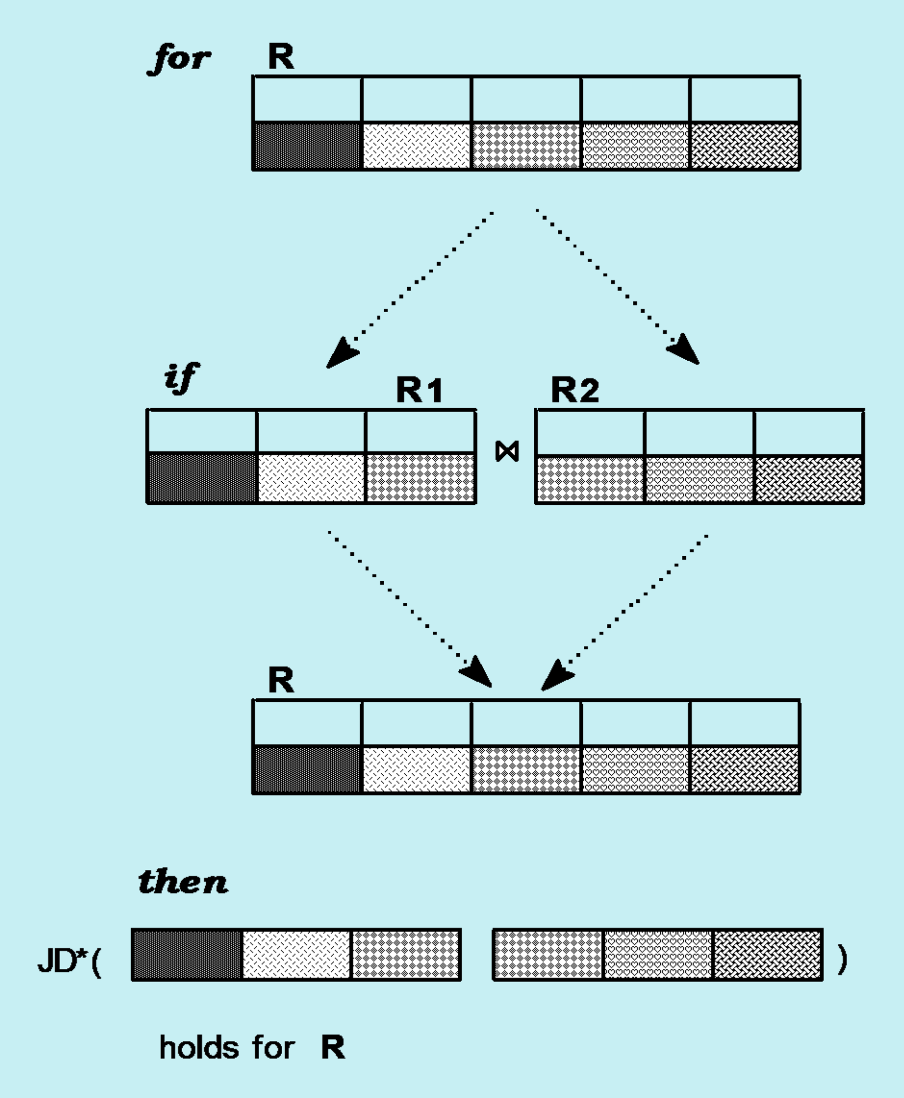
UNIVERSITY	DISCIPLINE	DEGREE
Old Town	Computing	BSc
Old Town	Computing	PhD
Old Town	Mathematics	PhD
New City	Computing	BSc
New City	Computing	PhD

spurious tuple

UNIVERSITY	DISCIPLINE	DEGREE
Old Town	Computing	BSc
Old Town	Computing	PhD
Old Town	Mathematics	PhD
New City	Computing	PhD

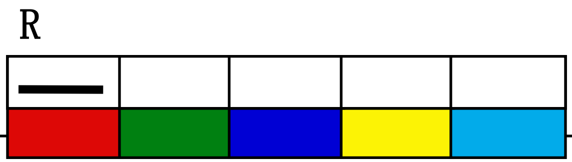
Join Dependency

JD * $(R_1, R_2, R_3, ..., R_m)$ **holds in R iff R** = join $(R_1, R_2, R_3, ..., R_m)$, R_i - a projection of **R**

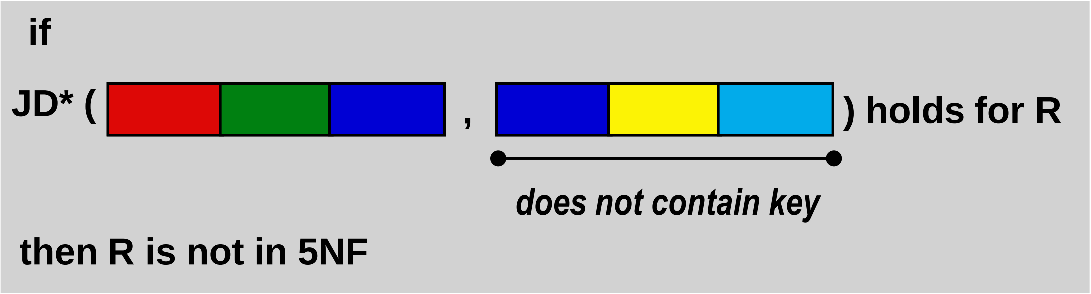


Fifth Normal Form

preventing illogical conjunction of facts



A relation R is in 5NF iff
for all JD * (R_1, R_2, R_3, ..., R_m) in R,
every R_i is a superkey for R.

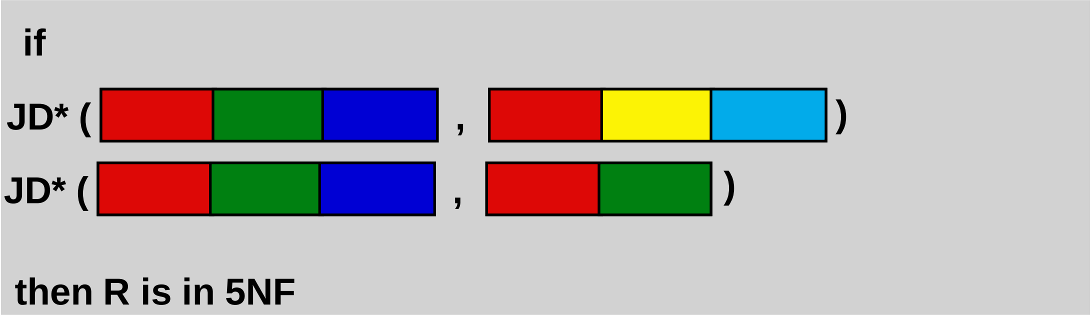


if

JD* (,) holds for R

does not contain key

then R is not in 5NF

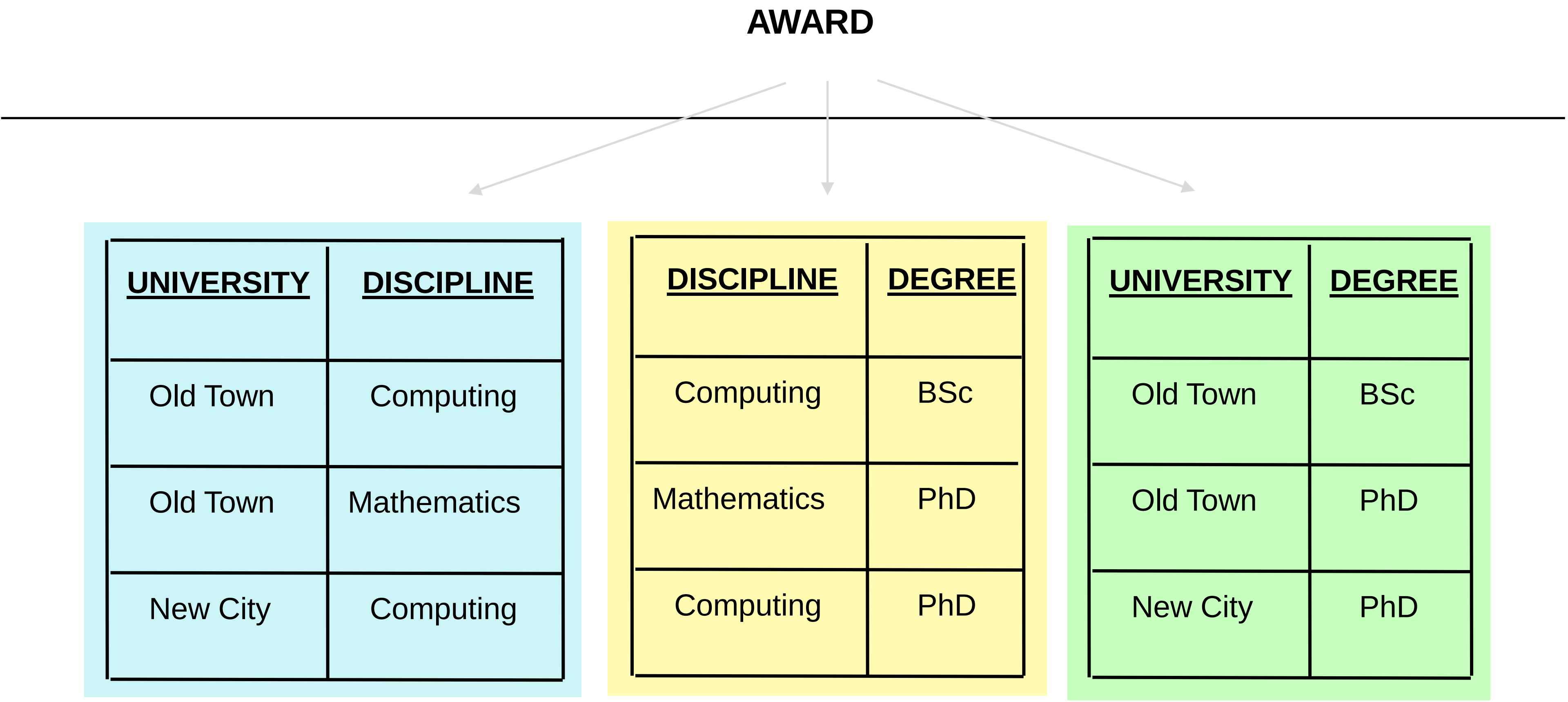


if

JD* (,)

JD* (,)

then R is in 5NF



AWARD

UNIVERSITY	DISCIPLINE
Old Town	Computing
Old Town	Mathematics
New City	Computing

DISCIPLINE	DEGREE
Computing	BSc
Mathematics	PhD
Computing	PhD

UNIVERSITY	DEGREE
Old Town	BSc
Old Town	PhD
New City	PhD

RANKING

NAME	CODE	TEACHING	RESEARCH
Old Town	P05	21	4
New City	C01	23	3

candidate keys - NAME or CODE

join dependencies

JD1 * ((NAME, CODE, TEACHING), (NAME, RESEARCH))
JD2 * ((NAME, CODE, RESEARCH), (NAME, TEACHING))
JD3 * ((NAME, CODE, TEACHING), (CODE, RESEARCH))
JD4 * ((NAME, CODE, RESEARCH), (CODE, TEACHING))
JD5 * ((NAME, CODE), (NAME, TEACHING), (CODE,RESEARCH))
....

all projections in JD1 - to JD5 are superkeys for **RANKING** → **5NF**