## NORMALIZATION
## normal forms

**Introduction**

Normalization process takes a relation schema through a series of tests to “certify” whether or not it belongs to a certain normal form.

**Normalization of data** can be looked on as a process during which unsatisfactory relation schemas are decomposed by breaking up their attributes into smaller relation schemas that posses desirable properties.

One objective of the normalization process is to ensure that the anomalies do not occur.

Normal forms, when considered in isolation from other factors, do not guarantee a good database design.

The normalization process must also confirm the existence of additional properties that the relational schemas should possess:

• **attribute preservation property** - which guarantee that no attribute will be lost during the normalization process;
• **lossless join property** - which guarantee that the spurious tuple problem does not occur;
• **dependency preservation property** - which ensures that all functional dependencies are represented in some of the individual resulting relation schema

## First Normal Form (1NF)

Relation schema *R* is in *first normal form* (1NF) if attribute values are single atomic (indivisible) values.

*Sex*

| **Name** | **Sex** |
|---|---|
| Peter, John, Tom, | male |
| Anna, Caroline | female |

Relation *Sex* in 1NF

| **Name** | **Sex** |
|---|---|
| Peter | male |
| John | male |
| Tom | male |
| Anna | female |
| Caroline | female |

The first normal form disallows composite attribute that are multivalued. The relations with composite attributes are called **nested relations**. In nested relation each tuple can have another relation within it.

EMP_PROJ(SSN, ENAME, {PROJS(PNUMBER, HOURS)})
EMP_PROJ(SSN, ENAME, {PNUMBER, HOURS})

EMP_PROJ

| | | PROJS | |
|---|---|---|---|
| SSN | ENAME | PNUMBER | HOURS |
| 12345678 | Smith | 1 | 32,5 |
| | | 2 | 7,5 |
| 66543723 | Narayan | 3 | 40,5 |
| 43432266 | English | 1 | 20,0 |
| | | 2 | 20,0 |
| 33333333 | Wong | 2 | 10,0 |
| | | 3 | 10,0 |
| | | 10 | 10,0 |
| | | 20 | 10,0 |

Notice that SSN is the primary key of the EMP_PROJ relation, and PNUMBER is the partial primary key of each nested relation, that is, within each tuple, the nested relation must have unique values of PNUMBER.

To normalize this relation into 1NF, we remove the nested relation attributes into a new relation and propagate the primary key into it.

The primary key of the new relation will combine the partial key with the primary key of the original relation.

EMP_PROJ1

| *SSN* | ENAME |
|---|---|

EMP_PROJ22

| *SSN* | *PNUMBER* | HOURS |
|---|---|---|

This procedure can be applied recursively to a relation with multiple-level nesting to unnest the relation into a set of 1NF relations.
This is useful in converting hierarchical schemas into 1NF relations.

## Second Normal Form (2NF)

**Full functional dependency**

A functional dependency $X \rightarrow Y$ is a **full functional dependency** in schema *R*, if removal of any attribute A from X means that the dependency does not hold any more. In other words, for any attribute $A \in X$, $(X - \{A\}) \nrightarrow Y$.

**Partial functional dependency**

A functional dependency $X \rightarrow Y$ is a **partial functional dependency** in schema *R*, if some attribute $A \in X$ can be removed from X and the dependency still holds. In other words, for some attribute $A \in X$, $(X - \{A\}) \rightarrow Y$.

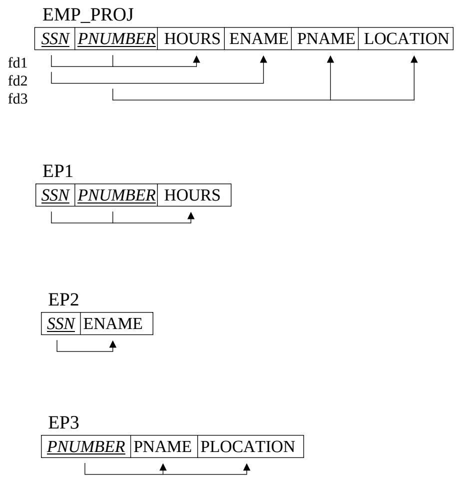
{SSN, PNUMBER} $\rightarrow$ HOURS
full functional dependency
{SSN, PNUMBER} $\rightarrow$ ENAME
partial functional dependency

## General Definition of Second Normal Form (2NF)

A relation schema *R* is 2NF if every nonprime attribute A in R is fully functionally dependent on every key of R

## Simplified Definition of Second Normal Form (2NF)

A relation schema *R* is in second normal form (2NF) if every nonprime attribute A is fully functionally dependent on the primary key of relation *schema R*.




## Third Normal Form (3NF)

Employees_PP

| *Name* | Institute | Faculty |
|---|---|---|
| Brzeziński | C. Sc. | El. Eng. |
| Morzy | C. Sc. | El. Eng. |
| Koszlajda | C. Sc. | El. Eng. |
| Królikowski | C. Sc. | El. Eng. |
| ... | ... | ... |
| Babij | Power | El. Eng. |
| Kordus | Power | El. Eng. |
| Sroczan | Power | El. Eng. |

Key: `Name`

Functional dependencies:

`Name` $\rightarrow$ `Institute`
`Name` $\rightarrow$ `Faculty`
`Institute` $\rightarrow$ `Faculty`

## Transitive functional dependency

A functional dependency X → Y in a relation schema R is a **transitive dependency**, if there is a set of attributes Z that is not a subset of any key of R, and both $X \rightarrow Z$ and $Z \rightarrow Y$ hold.

Employees_PP_1

| *Name* | Institute |
|---|---|
| Brzeziński | C. Sc. |
| Morzy | C. Sc. |
| Koszlajda | C. Sc. |
| Królikowski | C. Sc. |
| ... | ... |
| Babij | Power |
| Kordus | Power |
| Sroczan | Power |

Employees_PP_2

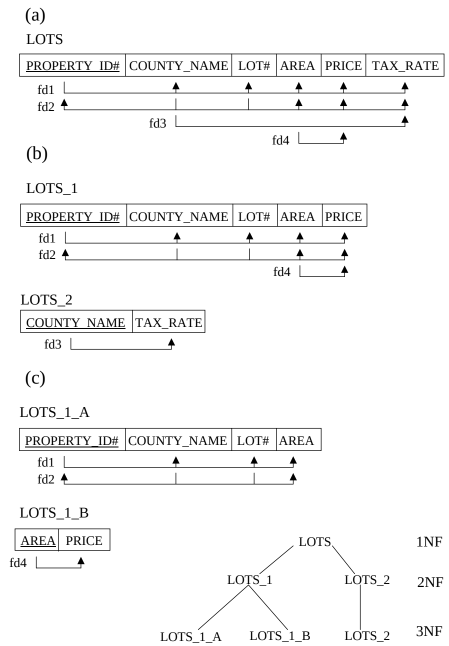
| *Institute* | Faculty |
|---|---|
| C. Sc. | El. Eng. |
| ... | ... |
| Power | El. Eng. |

## General Definition of Third Normal Form (3NF)

The relation *r* with schema *R* is in third normal form (3NF) if, whenever a functional dependency $X \rightarrow A$ hold in *R*, either (a) *X* is superkey of *R*, or (b) *A* is prime attribute of *R*.

## Simplified Definition of Third Normal Form (3NF)

The relation schema *R* is in 3NF if it is in 2NF and no nonprime attributes of R is transitively dependent on the primary key of schema *R*.



## Boyce-Codd Normal Form (BCNF)

Boyce-Codd Normal Form (BCNF) is a sufficient condition but not necessary.

LOTS

| PROPERTY_ID# | COUNTY_NAME | LOT# | AREA | PRICE | TAX_RATE |
|---|---|---|---|---|---|

Suppose that we have thousands of lots in the relation but the lots are from only two counties. Suppose also that lot sizes in the first country are only 0.5, 0.6, 0.7 acres, whereas lot sizes in the second county are restricted to 1, 1.2, 1.4 acres. This information will be duplicated in the thousands of tuples of the relation LOTS and after decomposition in the relation LOTS_1_A.

The relation LOTS is still in the Third Normal Form (County_name is a prime attribute).

LOTS_1_A

| PROPERTY_ID# | COUNTY_NAME | LOT# | AREA |
|---|---|---|---|

LOTS_1_A1

| PROPERTY_ID# | LOT# | AREA |
|---|---|---|

LOTS_1_A2

| AREA | COUNTY_NAME |
|---|---|

**Boyce-Codd Normal Form (BCNF)**

A relation schema $R$ is in BCNF if whenever a functional dependency $X \rightarrow A$ holds in $R$, then $X$ is a superkey of $R$.

## MULTIVALUED DEPENDENCY

**Flight**

| Flight | Day_of_week | Type_of_plane |
|---|---|---|
| 106 | Monday | 134 |
| 106 | Thursday | 154 |
| 106 | Monday | 154 |
| 106 | Thursday | 134 |
| 206 | Wednesday | 747 |
| 206 | Friday | 767 |
| 206 | Wednesday | 767 |
| 206 | Friday | 747 |

**Languages**

| Name | Foreign_language | Program_language |
|---|---|---|
| Smith | English | Basic |
| Smith | Italian | Fortran |
| Smith | English | Fortran |
| Smith | Italian | Basic |
| Smith | Czech | Basic |
| Smith | Czech | Fortran |

**Modification:**

Let us insert a new flight 106, additionally, in Wednesday, and a new type of airplane - TU104.

Flight

| Flight | Day_of_week | Type_of_airplane |
|---|---|---|
| 106 | Monday | 134 |
| 106 | Thursday | 154 |
| 106 | Monday | 154 |
| 106 | Thursday | 134 |
| *106* | *Monday* | *104* |
| *106* | *Thursday* | *104* |
| *106* | *Wednesday* | *134* |
| *106* | *Wednesday* | *154* |
| *106* | *Wednesday* | *104* |

**Flight_1**

| Flight | Day_of_week |
|---|---|
| 106 | Monday |
| 106 | Thursday |
| 206 | Wednesday |
| 206 | Friday |
| ... | ... |
| *106* | *Wednesday* |

**Flight_2**

| Flight | Type_of_plane |
|---|---|
| 106 | 134 |
| 106 | 154 |
| 206 | 747 |
| 206 | 767 |
| ... | ... |
| *106* | *104* |

**Language_1**

| Name | Foreign_language |
|---|---|
| Smith | English |
| Smith | Italian |
| Smith | Czech |

**Language_1**

| Name | Program_language |
|---|---|
| Smith | Basic |
| Smith | Fortran |

Multivalued dependencies are a consequence of first normal form, which disallowed an attribute in a tuple to have a set of values.

If we have two or more multivalued independent attributes in the same relation schema, we get into a problem of having to repeat every value of one of the attributes with every value of the other attribute to keep the relation instances consistent.

Flight $\rightarrow\rightarrow$ Day_of_week
Flight $\rightarrow\rightarrow$ Type_of_airplane

Name $\rightarrow\rightarrow$ Foreign_language
Name $\rightarrow\rightarrow$ Program_language

Appearance of a multivalued dependency $X \rightarrow\rightarrow Y$ in a relation schema $R = XYZ$ expresses two facts:
- relationship between the sets of attributes $X$, $Y$;
- independence of the sets of attributes $Y$, $Z$ - these sets are related indirectly through the set of attribute $X$.

## Multivalued Dependency

A multivalued dependency(MVD) X $\rightarrow\rightarrow$ Y specified on relation schema R, where X and Y are both subsets of R, specifies the following constraint on any relation r of the schema R:
If two any tuples $t_1$ and $t_2$ exist in *r* such that $t_1[X] = t_2[X]$, then two tuples $t_3$ and $t_4$ should also exist in *r* with the following properties:
- $t_1[X] = t_2[X] = t_3[X] = t_4[X]$
- $t_3[Y] = t_1[Y]$ and $t_4[Y] = t_2[Y]$
- $t_3[R - (XY)] = t_2[R - (XY)]$ and $t_4[R - (XY)] = t_1[R - (XY)]$

Whenever X $\rightarrow\rightarrow$ Y holds, we say that X multi-determines Y. Because of the symmetry of the definition, whenever X $\rightarrow\rightarrow$ Y holds in R, so does X $\rightarrow\rightarrow$ (R-(XY))

**Flight_3**

| Flight | Day_week | Type_of_airplane |
|---|---|---|
| 106 | Monday | 134 |
| 106 | Thursday | 154 |
| 106 | Thursday | 134 |
| 206 | Wednesday | 747 |
| 206 | Friday | 767 |

**The relation Flight_3 cannot be decomposed.**

**Trival Multivalued Dependency**

An multivalued dependency $X \rightarrow\rightarrow Y$ in relation *r(R)* is called trival MVD if:
- *Y* is a subset of *X*, or
- $X \cup Y = R$

A dependency is called trival, because it does not specify any constraint on the schema *R* and it will hold in any relation instance r of schema *R*.

## Fourth Normal Form (4NF)

A relation schema *R* is in fourth normal form (4NF) with respect to a set of multivalued dependencies MVD, if it is in 3NF and for every multivalued dependency $X \rightarrow\rightarrow Y \in MVD$, either it is trivial multivalued dependency or *X* is a superkey for *R*.

## Lossless join decomposition

**1. Lossless join decomposition into 3NF relations**

The relation schemas $R_1$ and $R_2$ form a lossless join decomposition of R with respect to a set of functional dependencies F on R if and only if either:
- $R_1 \cap R_2 \rightarrow R_1$ or
- $R_1 \cap R_2 \rightarrow R_2$

**2. Lossless join decomposition into 4NF relations**

The relation schemas $R_1$ and $R_2$ form a lossless join decomposition of R if and only if either:
- $R_1 \cap R_2 \rightarrow\rightarrow (R_1 - R_2)$, or
- $R_1 \cap R_2 \rightarrow\rightarrow (R_2 - R_1)$

**Join Dependencies and Fifth Normal Form**

Agents

| Agent | Company | Product |
|---|---|---|
| Kulczyk | Volkswagen | Car |
| Kulczyk | Volkswagen | Truck |
| Kulczyk | Audi | Car |
| Kulczyk | Audi | Truck |
| Nowak | Ford | Car |
| Nowak | Ford | Truck |
| Misiek | Nissan | Car |

R1

| Agent | Company |
|---|---|
| Kulczyk | Volkswagen |
| Kulczyk | Audi |
| Nowak | Ford |
| Misiek | Nissan |
| Morzy | Skoda |

R2

| Agent | Product |
|---|---|
| Kulczyk | Car |
| Kulczyk | Truck |
| Nowak | Car |
| Nowak | Truck |
| Misiek | Car |

R3

| Company | Product |
|---|---|
| Volkswagen | Car |
| Volkswagen | Truck |
| Audi | Car |
| Audi | Truck |
| Ford | Car |
| Ford | Track |
| Nissan | Car |
| Nissan | Truck |
| Skoda | Car |

Let $R = \{R_1, R_2, ..., R_p\}$ denotes a set of relation schemas defined on a set of attributes $U = \{A_1, A_2, ...A_n\}$, such that $R_1 \cup R_2 \cup ... \cup R_p = U$.
A **<u>join dependency</u>** (JD), denoted by *JD[R1, R2, ..., Rp]*, specified on relation schema R, specifies a constraint on instances of r of R. The constraint states that every legal instance r of R should have a lossless join decomposition into relations $r_1(R_1)$, $r_2(R_2)$, ..., $r_p(R_p)$, that is:

$$r(U) = r_1(R_1) \bowtie r_2(R_2) \bowtie ... \bowtie r_p(R_p).$$

A join dependency $JD[R_1, R_2, ..., R_p]$ specified on relation schema *R* is a trivial JD if one of the relation schemas $R_i$, $i = 1, 2, ...., p$ is equal to *R*.

## Fifth Normal Form (5NF)

A relation schema *R* is in **fifth normal form** (5NF or project-join normal form (PJNF))) if for every join dependency *JD* in the relation schema *R*:

- this dependency is a trivial
- every subschema $R_i$, $i = 1, 2, ..., p$ is a superkey of *R*.