

# Normal Forms

---

- Normalisation
- Normal Forms
- Boyce-Codd Normal Form
- Third Normal Form

## ❖ Normalisation

---

**Normalisation:** branch of relational theory providing design insights.

The goals of normalisation:

- be able to characterise the level of redundancy in a relational schema
- provide mechanisms for transforming schemas to remove redundancy

Normalisation draws heavily on the theory of functional dependencies.

Normalisation algorithms reduce the amount of redundancy in a schema

- by decomposition (break schema into connected pieces)

## ❖ Normal Forms

---

Normalisation theory defines six **normal forms** (NFs).

- First, Second, Third Normal Forms (1NF, 2NF, 3NF) (Codd 1972)
- Boyce-Codd Normal Form (BCNF) (1974)
- Fourth Normal Form (4NF) (Zaniolo 1976, Fagin 1977)
- Fifth Normal Form (5NF) (Fagin 1979)

We say that "a schema is in xNF", which ...

- tells us something about the level of redundancy in the schema

1NF allows most redundancy; 5NF allows least redundancy.

For most practical purposes, BCNF (or 3NF) are acceptable NFs.

## ❖ Normal Forms (cont)

---

- 1NF     all attributes have atomic values  
we assume this as part of relational model,  
so *every* relation schema is in 1NF
- 2NF     all non-key attributes fully depend on key  
(i.e. no partial dependencies)  
avoids much redundancy
- 3NF     no attributes dependent on non-key attrs  
BCNF    (i.e. no transitive dependencies)  
avoids most remaining redundancy

## ❖ Normal Forms (cont)

---

In practice, BCNF and 3NF are the most important.

Boyce-Codd Normal Form (BCNF):

- eliminates all redundancy due to functional dependencies
- but may not preserve original functional dependencies

Third Normal Form (3NF):

- eliminates most (but not all) redundancy due to *fds*
- guaranteed to preserve all functional dependencies

## ❖ Boyce-Codd Normal Form

A relation schema  $R$  is in BCNF w.r.t a set  $F$  of functional dependencies iff:

for all  $fds X \rightarrow Y$  in  $F^+$

- either  $X \rightarrow Y$  is trivial (i.e.  $Y \subset X$ )
- or  $X$  is a superkey (i.e. non-strict superset of attributes in key)

A DB schema is in BCNF if all of its relation schemas are in BCNF.

Observations:

- any two-attribute relation is in BCNF
- any relation with key  $K$ , other attributes  $Y$ , and  $K \rightarrow Y$  is in BCNF

## ❖ Boyce-Codd Normal Form (cont)

If we transform a schema into BCNF, we are guaranteed:

- no update anomalies due to *fd*-based redundancy
- lossless join decomposition

However, we are **not** guaranteed:

- the new schema preserves all *fds* from the original schema

This may be a problem if the *fds* contain significant semantic information about the problem domain (use 3NF to preserve dependencies)

A dependency  $A \rightarrow C$  is not preserved if, e.g.

- $X = ABC$  and  $ABC$  are all in relation  $R$
- after decomposition into  $S$  and  $T$ ,  $AB$  is in  $S$  and  $BC$  is in  $T$

## ❖ Boyce-Codd Normal Form (cont)

Example: schema in BCNF

$$R = ABCD, F = \{A \rightarrow B, A \rightarrow C, A \rightarrow D\}$$

$key(R) = A$ , all fds have key on RHS

Example: schema *not* in BCNF

$$R = ABCD, F = \{A \rightarrow BCD, D \rightarrow B, BC \rightarrow AD\}$$

if  $key(R) = A$ ,  $D \rightarrow B$  does not have key on LHS

if  $key(R) = BC$ ,  $D \rightarrow B$  does not have key on LHS



## ❖ Third Normal Form

---

A relation schema  $R$  is in 3NF w.r.t a set  $F$  of functional dependencies iff:

for all  $fds X \rightarrow Y$  in  $F^+$

- either  $X \rightarrow Y$  is trivial (i.e.  $Y \subset X$ )
- or  $X$  is a superkey
- or  $Y$  is a single attribute from a key

A DB schema is in 3NF if all relation schemas are in 3NF.

The extra condition represents a slight weakening of BCNF requirements.

## ❖ Third Normal Form (cont)

---

If we transform a schema into 3NF, we are guaranteed:

- lossless join decomposition
- the new schema preserves all of the *fd*s from the original schema

However, we are **not** guaranteed:

- no update anomalies due to *fd*-based redundancy

Whether to use BCNF or 3NF depends on overall design considerations.

## ❖ Third Normal Form (cont)

---

Example: schema in 3NF

$$R = ABCDE, F = \{ B \rightarrow ACDE, E \rightarrow B \}$$

$key(R) = B$ , in  $E \rightarrow B$ ,  $E$  is not a key, but  $B$  is

Example: schema *not* in 3NF

$$R = ABCDE, F = \{ B \rightarrow ACDE, E \rightarrow D \}$$

$key(R) = B$ , in  $E \rightarrow D$ ,  $E$  is not a key, neither is  $D$

Produced: 25 Mar 2021