Normal Forms - Reference

First Normal Form (1NF): Attributes should be atomic and tables should have no repeating groups

Second Normal Form: For every **X** → **A** that holds over relationship schema **R**, where A is a non-prime attribute

1. either **A ∈ X** (it is trivial), or
2. **X** is a superkey for **R**, or
3. **X** is transitively dependent on a super key R

Easier to think of the opposite: **X** cannot be a partial candidate key for R

– Says nothing about non-prime to non-prime dependencies!

Third Normal Form (3NF): For every **X** → **A** that holds over relationship schema **R**,

1. either **A ∈ X** (it is trivial), or
2. **X** is a superkey for **R**, or
3. **A** is a member of some candidate key for R

Easier to think of: **X** must be a full candidate key, unless A itself is a part of a candidate key

"Every non-key attribute must provide a fact about the Key, the whole Key, and nothing but the Key… so help me Codd"

Boyce-Codd Normal Form (BCNF): For every **X** → **A** that holds over relationship schema **R**,

1. either **A ∈ X** (it is trivial), or
2. **X** is a superkey for **R**

Functional Dependencies (FD)

X → Y : "X determines Y" / "Y is dependent on X"
where X and Y are sets of attributes

The **Closure** of F (denoted **F+**) is the set of all FDs:
{X → Y | X → Y is derivable from F by Armstrong's Axioms}

Two sets of dependencies F and G are equivalent if F+=G+

Armstrong's Axioms: where A, B, C are sets of attributes
Reflexive rule: if B ⊆ A, then A → B
Augmentation rule: if A → B, then C A → C B
Transitivity rule: if A → B, and B → C, then A → C

Union rule: If A → B and A → C, then A → B C
Decomposition rule: If A → B C, then A → B and A → C
Pseudotransitivity rule:If A → B and C B → D, then A C → D

Superkey of R: A (*possibly larger than necessary*) set of attributes that is sufficient to uniquely identify each tuple in r(R)

Candidate Key of R: A "minimal" superkey.

Primary Key: A specific Candidate Key chosen to represent a relation/table.

Non-prime: An attribute that is not part of any candidate key

Lossless Decomposition Test: R1, **R2** is a lossless join decomposition of **R** with respect to **F if and only if** at least one of the following dependencies is in **F+**

1. (R1 ∩ R2) → R1 – R2
2. (R1 ∩ R2) → R2 – R1

Dependency Preservation: After decomposition from **R** to **R1 … Rn**, the closure of FDs of all **R1…Rn** must be equivalent to that of **R**

Normal Forms - Practice

Q1: Consider a Relation **R3 = (A, B, C, D, E, F)** which has the following functional dependencies **F**:

A -> BC
CD -> E
CD -> F
B -> F

Which of the following must also hold:

1. A -> B
2. A -> C
3. A -> E
4. A -> F
5. C -> E
6. AD -> E

Q2: What Attributes can be used to define a Candidate Key for **R3** (above)?

Q3: Consider the Relation **R4 = (A,C,B,D,E)**, with Functional Dependencies:

A -> B
C -> D

What is the Candidate Key for R4? Which normal forms does this satisfy?

Q4: Consider the Relation **R5 = (V, W, X, Y, Z)**, with the following functional dependencies:

V -> X
WY -> X
VWY -> Z

In this relation, **(V, W, Y)** is the Candidate Key. What normal form does **R5** satisfy? You may assume that all tuples are unique and attributes are atomic.

Q5: Consider the relation **R6 = (A, B, C, D)**, with the following functional dependencies:

AB -> C
C -> D

What is the Candidate Key for this relation?

What normal forms does **R6** satisfy? You may assume that all tuples are unique and attributes are atomic.

If *instead* we had functional dependencies:

AB -> CD
D -> A

Which normal forms does **R6** now satisfy?