Normalization Preview

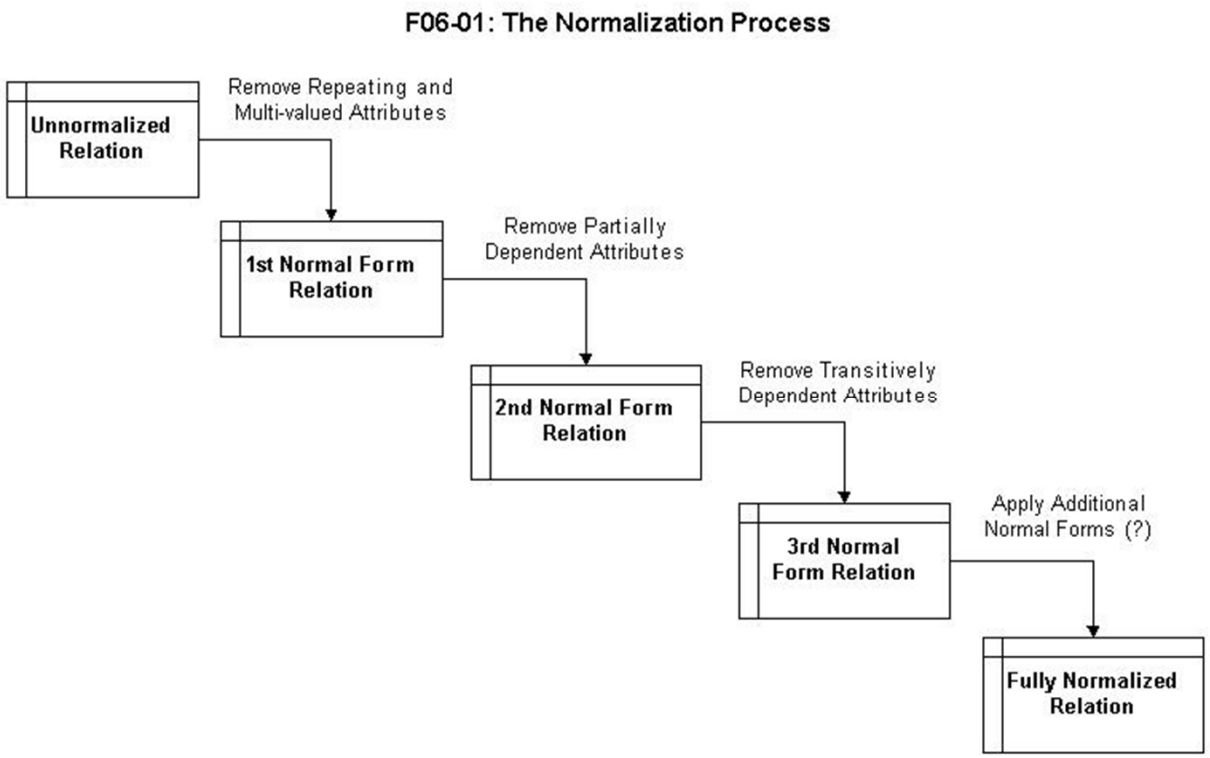
(Review if you have seen it before)

Normalization

Normalization: A technique for producing a set of relations with desirable properties, given the data requirements of an enterprise

- A bottom-up approach to database design
- First developed by E.F. Codd around 1972, with 3 normal forms
- Additional normal forms subsequently developed by Boyce-Codd and Fagin

The Normalization Process



Data Anomalies

Insertion Anomaly: Insert operation blocked by an artificial dependency (e.g. cannot insert a new project without an employee to assign to it)

Deletion Anomaly: Delete operation destroys unintended information (e.g. delete of last employee on project destroys project information)

Modification (Update) Anomaly: Changing a single data value requires changes to multiple tuples (e.g. changing a project due date requires a change to the record of every person assigned to the project)

A primary purpose of normalization is to remove these anomalies

Functional Dependency

Functional Dependency: "B" is functionally dependent on "A" if for every value of "A", there is exactly one value of "B"

Mathematically, we say that "A" ***determines*** "B"

Physically, we might say that "A" is a unique identifier for "B"

Full Functional Dependency: "B" is functionally dependent on "A" but not on any subset of "A"

Transitive Dependency: If "B" and "C" are both dependent on "A", but "C" is also dependent on "B", we say that "C" is ***transitively dependent*** on "B"

Functional Dependency

Full Functional Dependency: "B" is functionally dependent on "A" but not on any subset of "A"

Transitive Dependency: If "B" and "C" are both dependent on "A", but "C" is also dependent on "B", we say that "C" is *transitively dependent* on "B"

Normalization Process

Formal technique for analyzing relations based on their primary key and functional dependencies

Often executed as a number of steps, each step corresponding to a specific normal form

Normalization: Choosing a Unique Identifier

Normalization requires that we choose a *unique identifier* for each relation

Unique Identifier: a collection of one or more attributes that uniquely identifies each occurrence (each *tuple*) of a relation

Natural identifiers have real-world meaning

Surrogate (artificial) identifiers are meaningless replacements for real-world identifers

Criteria for Choosing a Unique Identifier

If there is only one candidate, choose it

Choose the candidate least like to have its value changed

Choose the simplest candidate

Choose the shortest candidate

Invent a unique identifer

First Normal Form (1NF)

Unnormalized Relation: a relation that contains repeating groups (or that has not been tested for 1NF)

First Normal Form: A relation where the intersection of each row and column contains only one value (i.e. a relation with no repeating groups of attributes and no multi-valued attributes)

Transformation to 1NF

Assign a unique identifier to each relation

Move repeating group to a new relation, copying the original unique identifier and adding to it so that it is unique.

For a multi-valued attribute (a repeating group of only one attribute), the attribute itself may be added to the original primary key to achieve uniqueness.

Example: Unnormalized Relation

INVOICE NUMBER, customer number, customer name, street address, city, state, zip, telephone, terms, ship via, order date, (product number, description, quantity, unit price, extended amount), total order amount

Example: 1NF

INVOICE NUMBER, customer number, customer name, street address, city, state, zip, telephone, terms, ship via, order date, total order amount

INVOICE NUMBER, PRODUCT NUMBER, description, quantity, unit price, extended amount

Second Normal Form (2NF)

Second Normal Form: A relation in 1NF where every non-key attribute is fully functionally dependent on the entire key

A 1NF relation with a single attribute primary key is automatically in 2NF

Example: 2NF

INVOICE NUMBER, customer number, customer name, street address, city, state, zip, telephone, terms, ship via, order date, total order amount

INVOICE NUMBER, PRODUCT NUMBER, quantity, sale price, extended amount

PRODUCT NUMBER, description, unit price

Third Normal Form (3NF)

Third Normal Form: a relation that is in 1NF and 2NF and in which no non-primary-key attributes are transitively dependent

Calculated attributes are transitively dependent since they are determined by other attributes

Transformation to 3NF

Calculated attributes may simply be removed (document the algorithm or formula)

Non-calculated attributes that are transitively dependent are placed in a relation (new or existing) where the primary key is their principal determinant

Example: 3NF

INVOICE NUMBER, customer number, terms, ship via, order date

INVOICE NUMBER, PRODUCT NUMBER, quantity, sale price

PRODUCT NUMBER, description, unit price

CUSTOMER NUMBER, name, address, city, state, zip, telephone

Normalization Summary

In a Third Normal Form relation,

Every non-key attribute depends on the key, the whole key and nothing but the key,

So help me Codd

Class Exercises