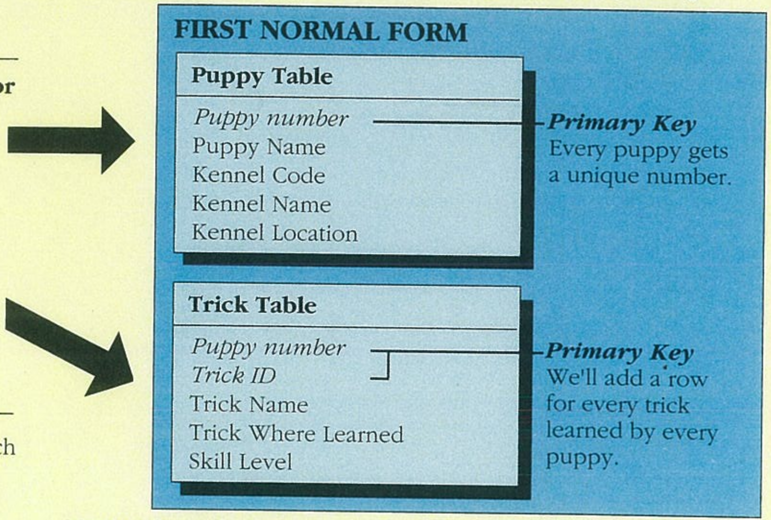
1 ELIMINATE REPEATING GROUPS

Make a separate table for each set of related attributes, and give each table a primary key.

Unnormalized Data Items For Puppies

Puppy Number
Puppy Name
Kennel Code
Kennel Name
Kennel Location
Trick ID 1...n
Trick Name 1...n
Trick Where Learned 1...n
Skill Level 1...n



FIRST NORMAL FORM

Puppy Table

Puppy number
Puppy Name
Kennel Code
Kennel Name
Kennel Location

Primary Key
Every puppy gets a unique number.

Trick Table

Puppy number
Trick ID
Trick Name
Trick Where Learned
Skill Level

Primary Key
We'll add a row for every trick learned by every puppy.

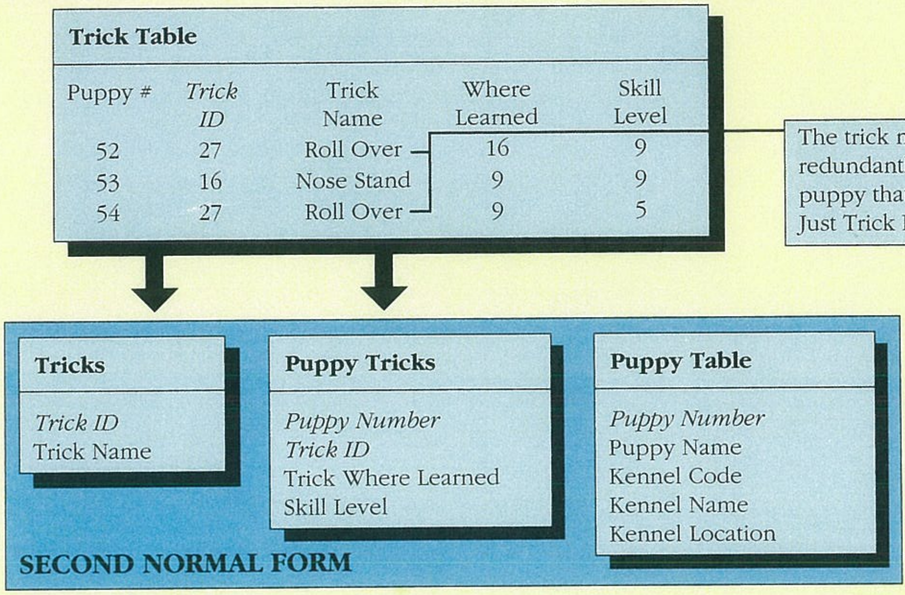
In the original list of data, each puppy's description is followed by a list of tricks the puppy has learned. Some might know ten tricks, some might not know any. To answer the question, "Can Fifi roll over" we need to first find Fifi's puppy record, then scan the list of tricks associated with the record. This is awkward, inefficient, and extremely untidy.

Moving the tricks into a separate table helps considerably. Separating the repeating groups of tricks from the puppy information results in first normal form. The puppy number in the trick table matches the primary key in the puppy table, providing a foreign key for relating the two tables with a join operation. Now we can answer our question with a direct retrieval: look to see if Fifi's puppy number and the trick ID for "roll over" appear together in the trick table.

2 ELIMINATE REDUNDANT DATA

> *"In anything at all, perfection is finally attained not when there is no longer anything to add, but when there is no longer anything to take away."*
>
> Saint-Exupéry,
> Wind, Sand, and Stars.

If an attribute depends on only part of a multi-valued key, remove it to a separate table.



Trick Table

Puppy #	*Trick ID*	Trick Name	Where Learned	Skill Level
52	27	Roll Over	16	9
53	16	Nose Stand	9	9
54	27	Roll Over	9	5

The trick name appears redundantly for every puppy that knows it. Just Trick ID would do.

Tricks
Trick ID
Trick Name

Puppy Tricks
Puppy Number
Trick ID
Trick Where Learned
Skill Level

Puppy Table
Puppy Number
Puppy Name
Kennel Code
Kennel Name
Kennel Location

SECOND NORMAL FORM

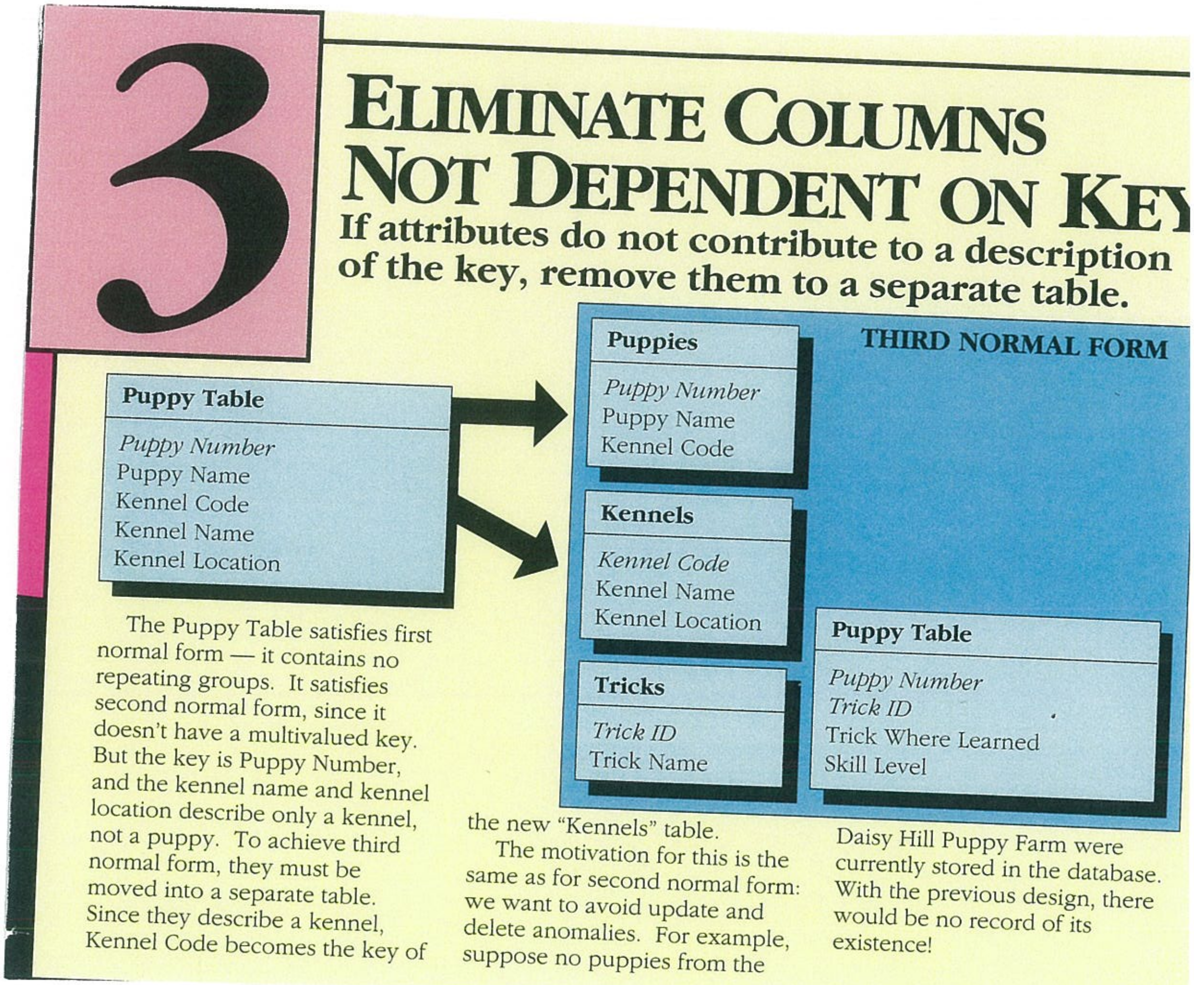
In the Trick Table, the primary key is made up of the puppy number and the trick ID. This makes sense for the "Where Learned" and "Skill Level" attributes, since they will be different for every puppy/trick combination. But the trick name depends only on the Trick ID. The same name will appear redundantly every time its associated ID appears in the Trick Table.

Suppose you want to reclassify a trick—give it a different trick ID. The change has to be made for every puppy that knows the trick! If you miss some, you'll have several puppies with the same trick under different IDs. This is an update anomaly.

Or suppose the last puppy knowing a particular trick gets run over by a car. His records will be removed from the database, and the trick will not be stored anywhere! This is a delete anomaly. To avoid these problems, we need second normal form.

To achieve this, separate the attributes depending on both parts of the key from those depending only on the Trick ID. This results in two tables: "Tricks," which gives the name for each Trick ID, and "Puppy Tricks," which lists the tricks learned by each puppy.

Now we can reclassify a trick in a single operation: look up the Trick ID in the "Tricks" table and change its name. The result will instantly be available throughout the application.



3 ELIMINATE COLUMNS NOT DEPENDENT ON KEY

If attributes do not contribute to a description of the key, remove them to a separate table.

Puppy Table

Puppy Number
Puppy Name
Kennel Code
Kennel Name
Kennel Location

THIRD NORMAL FORM

Puppies

Puppy Number
Puppy Name
Kennel Code

Kennels

Kennel Code
Kennel Name
Kennel Location

Tricks

Trick ID
Trick Name

Puppy Table

Puppy Number
Trick ID
Trick Where Learned
Skill Level

The Puppy Table satisfies first normal form — it contains no repeating groups. It satisfies second normal form, since it doesn't have a multivalued key. But the key is Puppy Number, and the kennel name and kennel location describe only a kennel, not a puppy. To achieve third normal form, they must be moved into a separate table. Since they describe a kennel, Kennel Code becomes the key of the new "Kennels" table.

The motivation for this is the same as for second normal form: we want to avoid update and delete anomalies. For example, suppose no puppies from the Daisy Hill Puppy Farm were currently stored in the database. With the previous design, there would be no record of its existence!