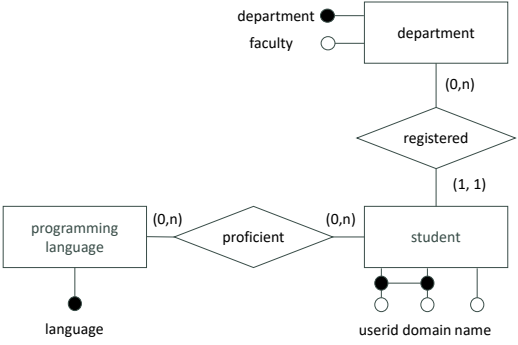
Anomalies and Normal Forms

Stéphane Bressan

Sentosa University of Technology (SUT) records the programming language skills of the students of its different faculties and departments.



We store everything in one table (`proficiency`.) What can go wrong?

proficiency					
name	userid	domain	department	faculty	language
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing	JavaScript
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing	Python
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing	C++
Stanley Georgeau	stan	comp.sut.edu	computer science	computing	Python
Goh Jin Wei	go	comp.sut.edu	information systems and analytics	computing	Python
Tan Hee Wee	tanhw	eng.sut.edu	computer engineering	engineering	C++
Tan Hee Wee	tanhw	eng.sut.edu	computer engineering	engineering	Fortran
Bjorn Sale	bjorn	eng.sut.edu	computer engineering	engineering	C++
Bjorn Sale	bjorn	eng.sut.edu	computer engineering	engineering	Fortran
Tan Hooi Ling	tanh	sci.sut.edu	physics	science	Julia
Tan Hooi Ling	tanh	sci.sut.edu	physics	science	Fortran
Roxana Nassi	rox	sci.sut.edu	mathematics	science	R
Amirah Mokhtar	ami	med.sut.edu	pharmacy	medecine	R

Each faculty is organised in several departments. A department belongs to one faculty only and there is no two different departments with the same name in Sentosa University of Technology.

proficiency					
name	userid	domain	department	faculty	language
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing	JavaScript
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing	Python
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing	C++
Stanley Georgeau	stan	comp.sut.edu	computer science	computing	Python
Goh Jin Wei	go	comp.sut.edu	information systems and analytics	computing	Python
Tan Hee Wee	tanhw	eng.sut.edu	computer engineering	engineering	C++
Tan Hee Wee	tanhw	eng.sut.edu	computer engineering	engineering	Fortran
Bjorn Sale	bjorn	eng.sut.edu	computer engineering	engineering	C++
Bjorn Sale	bjorn	eng.sut.edu	computer engineering	engineering	Fortran
Tan Hooi Ling	tanh	sci.sut.edu	physics	science	Julia
Tan Hooi Ling	tanh	sci.sut.edu	physics	science	Fortran
Roxana Nassi	rox	sci.sut.edu	mathematics	science	R
Amirah Mokhtar	ami	med.sut.edu	pharmacy	medecine	R

$$\{department\} \rightarrow \{faculty\}$$

Students are identified by their email. The email of a student is composed of her userid and domain.

proficiency					
name	userid	domain	department	faculty	language
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing	JavaScript
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing	Python
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing	C++
Stanley Georgeau	stan	comp.sut.edu	computer science	computing	Python
Goh Jin Wei	go	comp.sut.edu	information systems and analytics	computing	Python
Tan Hee Wee	tanhw	eng.sut.edu	computer engineering	engineering	C++
Tan Hee Wee	tanhw	eng.sut.edu	computer engineering	engineering	Fortran
Bjorn Sale	bjorn	eng.sut.edu	computer engineering	engineering	C++
Bjorn Sale	bjorn	eng.sut.edu	computer engineering	engineering	Fortran
Tan Hooi Ling	tanh	sci.sut.edu	physics	science	Julia
Tan Hooi Ling	tanh	sci.sut.edu	physics	science	Fortran
Roxana Nassi	rox	sci.sut.edu	mathematics	science	R
Amirah Mokhtar	ami	med.sut.edu	pharmacy	medecine	R

$$\{userid, domain\} \rightarrow \{name\}$$

We record the primary department of the student.

proficiency					
name	userid	domain	department	faculty	language
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing	JavaScript
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing	Python
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing	C++
Stanley Georgeau	stan	comp.sut.edu	computer science	computing	Python
Goh Jin Wei	go	comp.sut.edu	information systems and analytics	computing	Python
Tan Hee Wee	tanhw	eng.sut.edu	computer engineering	engineering	C++
Tan Hee Wee	tanhw	eng.sut.edu	computer engineering	engineering	Fortran
Bjorn Sale	bjorn	eng.sut.edu	computer engineering	engineering	C++
Bjorn Sale	bjorn	eng.sut.edu	computer engineering	engineering	Fortran
Tan Hooi Ling	tanh	sci.sut.edu	physics	science	Julia
Tan Hooi Ling	tanh	sci.sut.edu	physics	science	Fortran
Roxana Nassi	rox	sci.sut.edu	mathematics	science	R
Amirah Mokhtar	ami	med.sut.edu	pharmacy	medecine	R

$$\{userid, domain\} \rightarrow \{department\}$$

For each student, the database records (only once) each programming language in which the student is proficient.

proficiency					
name	userid	domain	department	faculty	language
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing	JavaScript
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing	Python
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing	C++
Stanley Georgeau	stan	comp.sut.edu	computer science	computing	Python
Goh Jin Wei	go	comp.sut.edu	information systems and analytics	computing	Python
Tan Hee Wee	tanhw	eng.sut.edu	computer engineering	engineering	C++
Tan Hee Wee	tanhw	eng.sut.edu	computer engineering	engineering	Fortran
Bjorn Sale	bjorn	eng.sut.edu	computer engineering	engineering	C++
Bjorn Sale	bjorn	eng.sut.edu	computer engineering	engineering	Fortran
Tan Hooi Ling	tanh	sci.sut.edu	physics	science	Julia
Tan Hooi Ling	tanh	sci.sut.edu	physics	science	Fortran
Roxana Nassi	rox	sci.sut.edu	mathematics	science	R
Amirah Mokhtar	ami	med.sut.edu	pharmacy	medecine	R

$$\{userid, domain, language\} \rightarrow \{name, userid, domain, department, faculty, language\}$$

How can we avoid repeating the same rows?

proficiency					
name	userid	domain	department	faculty	language
...					
Stanley Georgeau	stan	comp.sut.edu	computer science	computing	Python
Stanley Georgeau	stan	comp.sut.edu	computer science	computing	Python
...					

```
1 CREATE TABLE proficiency (
2 ...
3 PRIMARY KEY (name, userid, domain, department, faculty, language));
```

Let us call the table R (`proficiency`) and the columns A (`name`) , B (`userid`), C (`domain`), D (`department`), E (`faculty`), and F (`language`), respectively.

We have the following set of functional dependencies:
$\Sigma = \{\{B, C\} \to \{A\}, \{B, C\} \to \{D\}, \{D\} \to \{E\}, \{B, C, F\} \to \{A, B, C, D, E, F\}\}$

A minimal cover of Σ is:
$\Sigma' = \{\{B, C\} \to \{A\}, \{B, C\} \to \{D\}, \{D\} \to \{E\}\}$

A compact minimal cover of Σ is:
$\Sigma'' = \{\{B, C\} \to \{A, D\}, \{D\} \to \{E\}\}$

A compact minimal cover of Σ is:
$\Sigma'' = \{\{B, C\} \to \{A, D\}, \{D\} \to \{E\}\}$

Calculate the attribute closures to find all the candidate keys.
$\{A\}^+ = \{A\}$.
...
$\{B, C\}^+ = \{A, B, C, D, E\}$.
...
$\{B, C, F\}^+ = \{A, B, C, D, E, F\}$.
...

The candidate key (there is only one) of R with Σ is $\{B, C, F\}$ (`userid`, `domain`, `language`).

We store everything in one table. What can go wrong?

If we do not enforce the functional dependencies, we can experience anomalies:

- redundant storage,
- update anomalies,
- deletion anomalies, and
- insertion anomalies.

Let us illustrate the four different anomalies with this design and the functional dependency:

$$\{department\} \rightarrow \{faculty\}$$

proficiency					
name	userid	domain	department	faculty	language
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing	JavaScript
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing	Python
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing	C++
Stanley Georgeau	stan	comp.sut.edu	computer science	computing	Python
Goh Jin Wei	go	comp.sut.edu	information systems and analytics	computing	Python
Tan Hee Wee	tanhw	eng.sut.edu	computer engineering	engineering	C++
Tan Hee Wee	tanhw	eng.sut.edu	computer engineering	engineering	Fortran
Bjorn Sale	bjorn	eng.sut.edu	computer engineering	engineering	C++
Bjorn Sale	bjorn	eng.sut.edu	computer engineering	engineering	Fortran
Tan Hooi Ling	tanh	sci.sut.edu	physics	science	Julia
Tan Hooi Ling	tanh	sci.sut.edu	physics	science	Fortran
Roxana Nassi	rox	sci.sut.edu	mathematics	science	R
Amirah Mokhtar	ami	med.sut.edu	pharmacy	medecine	R

proficiency					
name	userid	domain	department	faculty	language
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing	JavaScript
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing	Python
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing	C++
Stanley Georgeau	stan	comp.sut.edu	computer science	computing	Python
Goh Jin Wei	go	comp.sut.edu	information systems and analytics	computing	Python
Tan Hee Wee	tanhw	eng.sut.edu	computer engineering	engineering	C++
Tan Hee Wee	tanhw	eng.sut.edu	computer engineering	engineering	Fortran
Bjorn Sale	bjorn	eng.sut.edu	computer engineering	engineering	C++
Bjorn Sale	bjorn	eng.sut.edu	computer engineering	engineering	Fortran
Tan Hooi Ling	tanh	sci.sut.edu	physics	science	Julia
Tan Hooi Ling	tanh	sci.sut.edu	physics	science	Fortran
Roxana Nassi	rox	sci.sut.edu	mathematics	science	R
Amirah Mokhtar	ami	med.sut.edu	pharmacy	medecine	R

Redundant Storage

The faculty of a department is repeated for every student of the department and every time the student is proficient in a programming language.

Is it really an issue?

IBM 726 (rental USD$850 per month in 1952)

Seagate ST12000NE0008 IronWolf Pro Internal Hard Drive, 12TB, 3.5"

S$595.23

Get it **Tue, 29 Mar - Fri, 1 Apr**
FREE Delivery

$$\{department\} \rightarrow \{faculty\}$$

If we want to save space and some query time (or not), we can store the departments and their faculties in a separate table.

```
1 CREATE TABLE department (
2 department VARCHAR(64),
3 faculty VARCHAR(128));
```

department	
department	faculty
computer science	computing
information systems and analytics	computing
computer engineering	engineering
physics	science
mathematics	science
pharmacy	medecine

Even then we still need to indicate the departments in the main table ...

proficiency				
name	userid	domain	department	language
Tan Hee Wee	tanh	comp.sut.edu	computer science	JavaScript
Tan Hee Wee	tanh	comp.sut.edu	computer science	Python
Tan Hee Wee	tanh	comp.sut.edu	computer science	C++
Stanley Georgeau	stan	comp.sut.edu	computer science	Python
Goh Jin Wei	go	comp.sut.edu	information systems and analytics	Python
Tan Hee Wee	tanhw	eng.sut.edu	computer engineering	C++
Tan Hee Wee	tanhw	eng.sut.edu	computer engineering	Fortran
Bjorn Sale	bjorn	eng.sut.edu	computer engineering	C++
Bjorn Sale	bjorn	eng.sut.edu	computer engineering	Fortran
Tan Hooi Ling	tanh	sci.sut.edu	physics	Julia
Tan Hooi Ling	tanh	sci.sut.edu	physics	Fortran
Roxana Nassi	rox	sci.sut.edu	mathematics	R
Amirah Mokhtar	ami	med.sut.edu	pharmacy	R

We join the two tables to recover the original table.

```
1 SELECT p.name, p.userid, d.domain, p.department, p.faculty, p.language
2 FROM proficiency p, department d
3 WHERE p.department = d.faculty;
```

```
1 SELECT p.name, p.userid, d.domain, p.department, p.faculty, p.language
2 FROM proficiency p INNER JOIN department d ON p.department = d.faculty;
```

```
1 SELECT *
2 FROM proficiency p NATURAL JOIN department d
```

proficiency					
name	userid	domain	department	faculty	language
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing	JavaScript
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing	Python
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing	C++
Stanley Georgeau	stan	comp.sut.edu	computer science	informatics	Python
Goh Jin Wei	go	comp.sut.edu	information systems and analytics	computing	Python
Tan Hee Wee	tanhw	eng.sut.edu	computer engineering	engineering	C++
Tan Hee Wee	tanhw	eng.sut.edu	computer engineering	engineering	Fortran
Bjorn Sale	bjorn	eng.sut.edu	computer engineering	engineering	C++
Bjorn Sale	bjorn	eng.sut.edu	computer engineering	engineering	Fortran
Tan Hooi Ling	tanh	sci.sut.edu	physics	science	Julia
Tan Hooi Ling	tanh	sci.sut.edu	physics	science	Fortran
Roxana Nassi	rox	sci.sut.edu	mathematics	science	R
Amirah Mokhtar	ami	med.sut.edu	pharmacy	medecine	R

```
1 UPDATE proficiency SET faculty='informatics' WHERE userid='stan' AND domain='comp.sut.edu' ;
```

Update Anomaly

Two rows of the table have the same value for the column `department` but different values for the column `faculty`. This violates the functional dependency.

If we store the department and the faculty in a separate table and we make the column `department` the primary key of this new table, the functional dependency is enforced by the constraint!

```
1 CREATE TABLE department (
2 department VARCHAR(128) PRIMARY KEY,
3 faculty VARCHAR(128) NOT NULL);
```

$$\{department\} \rightarrow \{faculty\}$$

department	
department	faculty
computer science	computing
information systems and analytics	computing
computer engineering	engineering
physics	science
mathematics	science
pharmacy	medecine

The new `proficiency` table has an `department` column but no `faculty` column. We make the `department` column in the new `proficiency` table a foreign key referencing the primary key of the `department` table.

proficiency				
name	userid	domain	department	language
Tan Hee Wee	tanh	comp.sut.edu	computer science	JavaScript
Tan Hee Wee	tanh	comp.sut.edu	computer science	Python
Tan Hee Wee	tanh	comp.sut.edu	computer science	C++
Stanley Georgeau	stan	comp.sut.edu	computer science	Python
Goh Jin Wei	go	comp.sut.edu	information systems and analytics	Python
Tan Hee Wee	tanhw	eng.sut.edu	computer engineering	C++
...				

```
1 CREATE TABLE proficiency (
2 ...
3 department VARCHAR(128) REFERENCES department(department);
4 ...);
```

proficiency					
name	userid	domain	department	faculty	language
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing	JavaScript
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing	Python
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing	C++
Stanley Georgeau	stan	comp.sut.edu	computer science	computing	Python
Goh Jin Wei	go	comp.sut.edu	information systems and analytics	computing	Python
Tan Hee Wee	tanhw	eng.sut.edu	computer engineering	engineering	C++
Tan Hee Wee	tanhw	eng.sut.edu	computer engineering	engineering	Fortran
Bjorn Sale	bjorn	eng.sut.edu	computer engineering	engineering	C++
Bjorn Sale	bjorn	eng.sut.edu	computer engineering	engineering	Fortran
Tan Hooi Ling	tanh	sci.sut.edu	physics	science	Julia
Tan Hooi Ling	tanh	sci.sut.edu	physics	science	Fortran
Roxana Nassi	rox	sci.sut.edu	mathematics	science	R

```
1 DELETE FROM proficiency WHERE userid='ami' AND domain='med.sut.edu';
```

Deletion Anomaly

When Amirah Moktar (`ami@med.sut.edu`) graduates, we may forget that we have a department of pharmacy and a faculty of medcine.

This cannot happen if we store the departments and their faculties in a separate table.

```
1 CREATE TABLE department (
2 department VARCHAR(128) PRIMARY KEY,
3 faculty VARCHAR(128) NOT NULL);
```

department	
department	faculty
computer science	computing
information systems and analytics	computing
computer engineering	engineering
physics	science
mathematics	science
pharmacy	medecine

proficiency					
name	userid	domain	department	faculty	language
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing	JavaScript
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing	Python
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing	C++
Stanley Georgeau	stan	comp.sut.edu	computer science	computing	Python
Goh Jin Wei	go	comp.sut.edu	information systems and analytics	computing	Python
Tan Hee Wee	tanhw	eng.sut.edu	computer engineering	engineering	C++
Tan Hee Wee	tanhw	eng.sut.edu	computer engineering	engineering	Fortran
Bjorn Sale	bjorn	eng.sut.edu	computer engineering	engineering	C++
Bjorn Sale	bjorn	eng.sut.edu	computer engineering	engineering	Fortran
Tan Hooi Ling	tanh	sci.sut.edu	physics	science	Julia
Tan Hooi Ling	tanh	sci.sut.edu	physics	science	Fortran
Roxana Nassi	rox	sci.sut.edu	mathematics	science	R
null	null	null	pharmacy	medecine	null

Insertion Anomaly

We cannot record the department of pharmacy and the faculty of medicine as there is no student from this department and faculty and we cannot have null values for `name`, `userid`, and `domain`, and `language` because they should be `NOT NULL` (some are prime attributes part of the `PRIMARY KEY`).

The solution , as we have seen, is to modify the `profiiency` table and create a separate `department` table.

department	
department	faculty
computer science	computing
information systems and analytics	computing
computer engineering	engineering
physics	science
mathematics	science
pharmacy	medecine

We could use an outer join[a] to combine the two tables and recreate the original table with null values that we intended.

[a] Do we need a full, left,or right outer join?

```
1 SELECT *
2 FROM proficiency p FULL OUTER JOIN  department d ON p.department = d.department;
```

proficiency					
name	userid	domain	department	faculty	language
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing	JavaScript
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing	Python
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing	C++
Stanley Georgeau	stan	comp.sut.edu	computer science	computing	Python
Goh Jin Wei	go	comp.sut.edu	information systems and analytics	computing	Python
Tan Hee Wee	tanhw	eng.sut.edu	computer engineering	engineering	C++
Tan Hee Wee	tanhw	eng.sut.edu	computer engineering	engineering	Fortran
Bjorn Sale	bjorn	eng.sut.edu	computer engineering	engineering	C++
Bjorn Sale	bjorn	eng.sut.edu	computer engineering	engineering	Fortran
Tan Hooi Ling	tanh	sci.sut.edu	physics	science	Julia
Tan Hooi Ling	tanh	sci.sut.edu	physics	science	Fortran
Roxana Nassi	rox	sci.sut.edu	mathematics	science	R
null	null	null	pharmacy	medecine	null

Verify that we have similar anomalies with the other non-trivial functional dependencies.

The purpose of normal forms is to recognise designs that enforce functional dependencies by means of the main SQL constraints (primary key, unique, not null, and foreign key constraints) and thus protect the data against anomalies.

The purpose of normalisation is to transform (decompose) a poor design into a design that enforces functional dependencies by meansthe main SQL constraints.

The candidate key is $\{userid, domain, language\}$, yet some attributes, like $name$, $department$, and $domain$ depend only on a proper subset of a candidate key:

$$\{userid, domain\} \rightarrow \{name, department, faculty\}$$

These attributes describe the student and do not depend on or inform us about the relationship with the programming language.

Definition

We say that an attribute A is fully dependent on X if and only if there exists a non-trivial (the right-hand side is a subset of the left-hand side) functional dependency $X \rightarrow \{A\}$ such that no proper subset Y of X is such that $Y \rightarrow \{A\} \in \Sigma^+$.

The candidate key is $\{userid, domain, language\}$, therefore $name$ and $department$ are not fully dependent on the candidate key $\{userid, faculty, language\}$ because $\{userid, domain\} \rightarrow \{name\}$ and$\{userid, domain\} \rightarrow \{department\}$.

First Idea

Let us require that every attribute be fully depends on the primary key.

But prime attributes (attributes in a candidate key) may not fully depend on the key (if there is a composite primary key)!

First Idea (refined)

Let us require that every non-prime attribute fully depends on the primary key.

But there could be more than one candidate key!

First Idea (further refined)

Let us require that every non-prime attribute fully depends on each candidate key.

Definition

A relation R with a set of functional dependencies Σ is in Second Normal Form, or 2NF for short, if and and only if every non-prime attribute is fully dependent on each candidate key.

Theorem

A relation R with a set of functional dependencies Σ is in Second Normal Form if and only if for every functional dependency $X \to \{A\} \in \Sigma^+$:

- $X \to \{A\}$ *is trivial or*
- A *is a prime attribute or*
- X *is not a proper subset of a candidate key.*

It is sufficient to look at Σ.

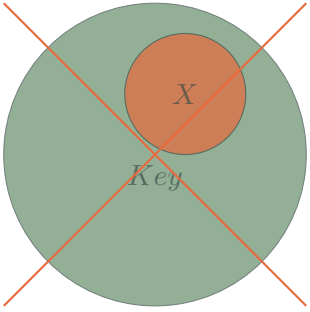
Theorem

A relation R with a set of functional dependencies Σ is in Second Normal Form if and only if for every functional dependency $X \to \{A\} \in \Sigma^+$:

- $X \to \{A\}$ *is trivial or*
- A *is a prime attribute or*
- X *is not a proper subset of a candidate key.*

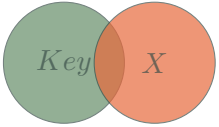
It is sufficient to look at Σ.

This situation where X is a proper subset of a candidate key is forbidden:

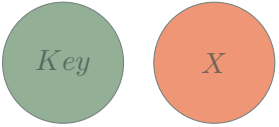


X cannot be proper subset of a candidate key.
A must be fully dependent on each candidate key.

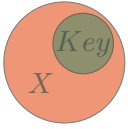
For all candidate keys, we must have one of the following:



X intersect with the candidate key.



X and the candidate key are disjoint.



X is a superset of the candidate key (X is a superkey).

X is a candidate key (X is a superkey).

$\{userid, domain\} \rightarrow \{name\}$ is non-trivial, its left-hand side is a proper subset of a candidate key, and its right-hand side is not a prime attribute. It does violate the three conditions of the theorem.

We have found a culprit. The table is not in 2NF.

A solution is to decompose the table into two tables, `student` and `proficiency`.

student				
name	userid	domain	department	faculty
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing
Stanley Georgeau	stan	comp.sut.edu	computer science	computing
Goh Jin Wei	go	comp.sut.edu	information systems and analytics	computing
...				

proficiency		
userid	domain	language
tanh	comp.sut.edu	JavaScript
tanh	comp.sut.edu	Python
tanh	comp.sut.edu	C++
stan	comp.sut.edu	Python
go	comp.sut.edu	Python
...		

We now need to study the two new tables and their (projected) functional dependencies.

student				
name	userid	domain	department	faculty
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing
Stanley Georgeau	stan	comp.sut.edu	computer science	computing
Goh Jin Wei	go	comp.sut.edu	information systems and analytics	computing
...				

This table has the following non-trivial projected functional dependencies.

$$\{userid, domain\} \rightarrow \{name\}, \{userid, domain\} \rightarrow \{department\},$$
$$\{department\} \rightarrow \{faculty\}$$

Its candidate key is $\{userid, domain\}$. The prime attributes are $userid$ and $domain$.

We check every constraint meets at least one of the conditions of the theorem:

- $\{userid, domain\} \rightarrow \{name\}$ is non-trivial, its left-hand side is not a proper subset of a candidate key (it is a candidate key), and its right-hand side is not a prime attribute. It does not violate all the conditions of the theorem.
- $\{userid, domain\} \rightarrow \{department\}$ is non-trivial, its left-hand side is not a proper subset of a candidate key (it is a candidate key), and its right-hand side is not a prime attribute. It does not violate all the conditions of the theorem.
- $\{department\} \rightarrow \{faculty\}$ is non-trivial, its left-hand side is not a proper subset of a candidate key (it is disjoint), and its right-hand side is not a prime attribute. It does not violate all the conditions of the theorem.

The table is in 2NF.

proficiency		
userid	domain	language
tanh	comp.sut.edu	JavaScript
tanh	comp.sut.edu	Python
tanh	comp.sut.edu	C++
stan	comp.sut.edu	Python
go	comp.sut.edu	Python
...		

This table has no non-trivial projected functional dependencies. Its candidate key is $\{userid, domain, language\}$. It is in 2NF (there is no non-trivial functional dependency to violate the three conditions in the theorem.)

student				
name	userid	domain	department	faculty
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing
Stanley Georgeau	stan	comp.sut.edu	computer science	computing
Goh Jin Wei	go	comp.sut.edu	information systems and analytics	computing
...				

There are still attributes that do not directly depend on the candidate keys.

Indeed, $faculty$ was fully dependent on the candidate key $\{userid, domain, language\}$ and is still fully dependent on the candidate key $\{userid, domain\}$, yet it depends on something on $department$, which is not a candidate key: $\{department\} \rightarrow \{faculty\}$.

student				
name	userid	domain	department	faculty
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing
Stanley Georgeau	stan	comp.sut.edu	computer science	computing
Goh Jin Wei	go	comp.sut.edu	information systems and analytics	computing
...				

$faculty$ is transitively dependent on the primary key $\{userid, domain\}$ because $\{userid, domain\} \rightarrow \{department\}$ and $\{department\} \rightarrow \{faculty\}$ (therefore $\{userid, domain\} \rightarrow \{faculty\}$).

Definition

We say that an attribute A is transitively dependent on a candidate key if and only if there exists a set of attributes S such that S is not a superkey and the non-trivial functional dependency $S \rightarrow \{A\}$ holds.

Second Idea

Let us require that no attribute be transitively dependent on any candidate key.

"A non-key field must provide a fact about the key[s], the whole key[s], and nothing but the key[s], [so help me Codd.]",
W. Kent in "A Simple Guide to Five Normal Forms in Relational Database Theory", Communication of the ACM, Volume 26, Number 2 (1983).

Definition

A relation R with a set of functional dependencies Σ is in Boyce-Codd Normal Form, or BCNF for short, if and only if for every attribute set $S \subset R$, if any attribute of R not in S is functionally dependent on S, then all attributes in R are functionally dependent on S.

Theorem

A relation R with a set of functional dependencies Σ is in BCNF if and only if no attribute is transitively dependent on any key. [David Maier]

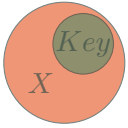
Theorem

A relation R with a set of functional dependencies Σ is in BCNF if and only if for every functional dependency $X \to \{A\} \in \Sigma^+$:

- *$X \to \{A\}$ is trivial or*
- *X is a superkey.*

It is sufficient to look at Σ.

For some candidate key, we must have one of the following:



X is a superset of the candidate key (X is a superkey).

X is the candidate key (X is a superkey).

$\{department\} \rightarrow \{faculty\}$ is non-trivial, its left-hand side is not a superkey, and its right-hand side is not a prime attribute. It does violate the three conditions of the theorem.

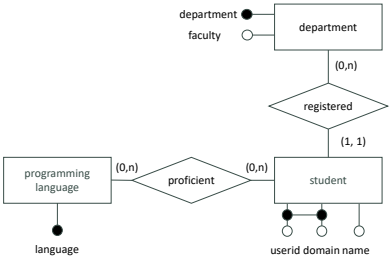
We have found a culprit. The table is not in BCNF.

A solution is again to decompose the table into two.

student			
name	userid	domain	department
Tan Hee Wee	tanh	comp.sut.edu	computer science
Tan Hee Wee	tanh	comp.sut.edu	computer science
Tan Hee Wee	tanh	comp.sut.edu	computer science
Stanley Georgeau	stan	comp.sut.edu	computer science
Goh Jin Wei	go	comp.sut.edu	information systems and analytics
...			

department	
department	faculty
computer science	computing
information systems and analytics	computing
computer engineering	engineering
physics	science
mathematics	science
pharmacy	medecine

We now need to study the two new tables and their (projected) functional dependencies. (Verify that they are in BCNF.)



We have three tables: `student(userid, domain, name, department)`, `department(department, faculty)`, and `proficiency(userid, domain, language)`.

What about this case?

$R = \{A, B, C\}$
$\Sigma = \{\{A, B\} \to \{C\}, \{C\} \to \{B\}\}$
The candidate keys are $\{A, B\}$ and $\{A, C\}$.
$\{C\} \to \{B\}$ is non trivial, yet $\{C\}$ is not a candidate key.
It is not in BCNF! There is nothing that we do about it without losing some functional dependencies.

Third Idea

Let us relax the BCNF requirements for prime attributes.

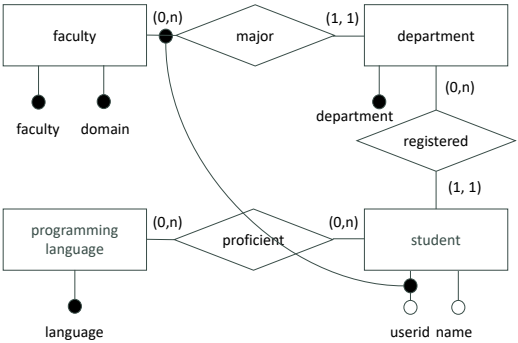
Theorem

A relation R with a set of functional dependencies Σ is in Third Normal Form, or 3NF for short, if and only if for every functional dependency $X \to \{A\} \in \Sigma^+$:

- $X \to \{A\}$ *is trivial or*
- X *is a superkey or*
- A *is a prime attribute.*

It is sufficient to look at Σ.

What if, in our case, there is exactly one domain for each faculty?



What if, in our case, there is exactly one domain for each faculty?

$\Sigma = \{\{userid, domain\} \rightarrow \{name, department\}, \{department\} \rightarrow \{faculty\}, \{faculty\} \rightarrow \{domain\}, \{domain\} \rightarrow \{faculty\}\}$

student				
name	userid	domain	department	faculty
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing
Stanley Georgeau	stan	comp.sut.edu	computer science	computing
Goh Jin Wei	go	comp.sut.edu	information systems and analytics	computing
Tan Hee Wee	tanhw	eng.sut.edu	computer engineering	engineering
Tan Hee Wee	tanhw	eng.sut.edu	computer engineering	engineering
Bjorn Sale	bjorn	eng.sut.edu	computer engineering	engineering
Bjorn Sale	bjorn	eng.sut.edu	computer engineering	engineering
Tan Hooi Ling	tanh	sci.sut.edu	physics	science
Tan Hooi Ling	tanh	sci.sut.edu	physics	science
Roxana Nassi	rox	sci.sut.edu	mathematics	science
Amirah Mokhtar	ami	med.sut.edu	pharmacy	medecine

Let us call the table R (`student`) and the columns A (`name`) , B (`userid`), C (`domain`), D (`department`), and E (`faculty`), respectively.

We have the following compact minimal cover:
$\Sigma'' = \{\{B, C\} \rightarrow \{A, D\}, \{D\} \rightarrow \{E\}, \{C\} \rightarrow \{E\}, \{E\} \rightarrow \{C\}\}$

Calculate the attribute closures to find all the candidate keys.

...
$\{B, C\}^+ = \{A, B, C, D, E\}.$
...
$\{B, D\}^+ = \{A, B, C, D, E\}$
....
$\{B, E, \}^+ = \{A, B, C, D, E\}.$

Let us call the table R (`student`) and the columns A (`name`) , B (`userid`), C (`domain`), D (`department`), and E (`faculty`), respectively.

We have the following compact minimal cover:
$\Sigma'' = \{\{B, C\} \to \{A, D\}, \{D\} \to \{E\}, \{C\} \to \{E\}, \{E\} \to \{C\}\}$

The candidate keys of R with Σ are $\{B, C\}$ (`userid`, `domain`), $\{B, E\}$ (`userid`, `faculty`), and $\{B, D\}$ (`userid`, `department`). B, C, D, and E are prime.

R with Σ is in 2NF, in 3NF, but not in BCNF.

$\Sigma = \{\{userid, domain\} \rightarrow \{name, department\}, \{department\} \rightarrow \{faculty\}, \{faculty\} \rightarrow \{domain\}, \{domain\} \rightarrow \{faculty\}\}$

`student` with Σ is in 2NF, in 3NF, but not in BCNF.

student				
name	userid	domain	department	faculty
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing
Tan Hee Wee	tanh	comp.sut.edu	computer science	computing
Stanley Georgeau	stan	comp.sut.edu	computer science	computing
Goh Jin Wei	go	comp.sut.edu	information systems and analytics	computing
Tan Hee Wee	tanhw	eng.sut.edu	computer engineering	engineering
Tan Hee Wee	tanhw	eng.sut.edu	computer engineering	engineering
Bjorn Sale	bjorn	eng.sut.edu	computer engineering	engineering
Bjorn Sale	bjorn	eng.sut.edu	computer engineering	engineering
Tan Hooi Ling	tanh	sci.sut.edu	physics	science
Tan Hooi Ling	tanh	sci.sut.edu	physics	science
Roxana Nassi	rox	sci.sut.edu	mathematics	science
Amirah Mokhtar	ami	med.sut.edu	pharmacy	medecine

All You Need to Remember

A relation R with a set of functional dependencies Σ is in

- Second Normal Form if and only if for every non-trivial functional dependency $X \to \{A\} \in \Sigma$:
  - X is not a proper subset of a candidate key or
  - A is a prime attribute.
- Third Normal Form, or 3NF for short, if and only if for every non-trivial functional dependency $X \to \{A\} \in \Sigma$:
  - X is a superkey or
  - A is a prime attribute.
- BCNF if and only if for every non-trivial functional dependency $X \to \{A\} \in \Sigma$:
  - X is a superkey.

Theorem

$$BCNF \subset 3NF \subset (2NF) \subset \text{“}1NF\text{”}$$

Theorem

$$1NF \neq 2NF \neq 3NF \neq BCNF$$

There are more normal forms that correspond to functional dependencies as well as other integrity constraints (e.g. multi-valued dependencies).

$R = \{A, B, C, D, E\}$
$\Sigma = \{\{A, B\} \to \{C, D, E\}, \{A, C\} \to \{B, D, E\}, \{B\} \to \{C\}, \{C\} \to \{B\}, \{C\} \to \{D\}, \{B\} \to \{E\}, \{C\} \to \{E\}\}$

$\Sigma'' = \{\{B\} \to \{C\}, \{C\} \to \{B, D, E\}\}$

Is R with Σ in 2NF? 3NF? BCNF?

$R = \{A, B, C, D, E\}$
$\Sigma = \{\{A, B\} \to \{C, D, E\}, \{A, C\} \to \{B, D, E\}, \{B\} \to \{C\}, \{C\} \to \{B\}, \{C\} \to \{D\}, \{B\} \to \{E\}, \{C\} \to \{E\}\}$

The two candidate keys are $\{A, B\}$ and $\{A, C\}$.

$\{C\} \to \{D\}$ is non-trivial, $\{D\}$ is not a prime attribute and $\{c\}$ is a proper subset of a candidate key. Therefore R with Σ in not in 2NF. Therefore it is not in 3NF and not in BCNF.