Lecture 6

Chapter 3: Functional Dependencies, Section 3.2.5 — 3.5.1

John Connor

March 7, 2018

Definition: Trivial Functional Dependency

A functional dependency $A_1 \cdots A_n \to B_1 \cdots B_m$ is said to be trivial if $\{B_1, \cdots, B_m\} \subseteq \{A_1, \cdots, A_n\}$.

Definition: Minimal Basis

Given two sets F, G of functional dependencies, if F is equivalent to G then F is said to be a *basis* of G.

Definition: Minimal Basis

Given two sets F, G of functional dependencies, if F is equivalent to G then F is said to be a *basis* of G.
A basis F of G is a *minimal basis* if

Definition: Minimal Basis

Given two sets F, G of functional dependencies, if F is equivalent to G then F is said to be a *basis* of G.
A basis F of G is a *minimal basis* if

1. Every functional dependency in F has one attribute on the right hand side.

Definition: Minimal Basis

Given two sets F, G of functional dependencies, if F is equivalent to G then F is said to be a *basis* of G.
A basis F of G is a *minimal basis* if

1. Every functional dependency in F has one attribute on the right hand side.
2. If we remove any functional dependency from F, then the result is not a basis.

Definition: Minimal Basis

Given two sets F, G of functional dependencies, if F is equivalent to G then F is said to be a *basis* of G.
A basis F of G is a *minimal basis* if

1. Every functional dependency in F has one attribute on the right hand side.
2. If we remove any functional dependency from F, then the result is not a basis.
3. If for any functional dependency in F we remove one or more attributes from the left hand side of the FD, then the result is not a basis.

Question: Minimal Basis

What is a minimal basis for $A \to B, B \to C, A \to C$?

Question: Minimal Basis

What is a minimal basis for $A \rightarrow B, B \rightarrow C, A \rightarrow C$?

$$A \rightarrow B$$
$$B \rightarrow C$$

Anomalies (3.3.1)

Table: A poorly designed relation "Movies1".

title	year	length	genre	studio	star
Star Wars	1977	124	sciFi	Fox	Carrie Fisher
Star Wars	1977	124	sciFi	Fox	Mark Hamill
Gone With the Wind	1939	231	drama	MGM	Vivien Leigh
Wayne's World	1992	95	comedy	Paramnt	Dana Carvey
Wayne's World	1992	95	comedy	Paramnt	Mike Meyers

Assume there are no other relations in the database pertaining to movies. What is wrong with this relation?

Anomalies (3.3.1)

Table: A poorly designed relation "Movies1".

title	year	length	genre	studio	star
Star Wars	1977	124	sciFi	Fox	Carrie Fisher
Star Wars	1977	124	sciFi	Fox	Mark Hamill
Gone With the Wind	1939	231	drama	MGM	Vivien Leigh
Wayne's World	1992	95	comedy	Paramnt	Dana Carvey
Wayne's World	1992	95	comedy	Paramnt	Mike Meyers

Assume there are no other relations in the database pertaining to movies.
What is wrong with this relation?

1. Redundancy. Information, such as genre, is repeated unnecessarily.

Anomalies (3.3.1)

Table: A poorly designed relation "Movies1".

title	year	length	genre	studio	star
Star Wars	1977	124	sciFi	Fox	Carrie Fisher
Star Wars	1977	124	sciFi	Fox	Mark Hamill
Gone With the Wind	1939	231	drama	MGM	Vivien Leigh
Wayne's World	1992	95	comedy	Paramnt	Dana Carvey
Wayne's World	1992	95	comedy	Paramnt	Mike Meyers

Assume there are no other relations in the database pertaining to movies. What is wrong with this relation?

1. Redundancy. Information, such as genre, is repeated unnecessarily.
2. Update Anomalies. We might update genre, length, studio, etc. for some tuples (say with the title "Star Wars") but not all.

Anomalies (3.3.1)

Table: A poorly designed relation "Movies1".

title	year	length	genre	studio	star
Star Wars	1977	124	sciFi	Fox	Carrie Fisher
Star Wars	1977	124	sciFi	Fox	Mark Hamill
Gone With the Wind	1939	231	drama	MGM	Vivien Leigh
Wayne's World	1992	95	comedy	Paramnt	Dana Carvey
Wayne's World	1992	95	comedy	Paramnt	Mike Meyers

Assume there are no other relations in the database pertaining to movies. What is wrong with this relation?

1. Redundancy. Information, such as genre, is repeated unnecessarily.
2. Update Anomalies. We might update genre, length, studio, etc. for some tuples (say with the title "Star Wars") but not all.
3. Deletion Anomalies. If we delete tuples containing a star of "Vivien Leigh" then we lose all information about "Gone With the Wind."

Decomposing Relations (3.3.2)

Given a relation $R(A_1, \cdots, A_n)$ we can *decompose* it into two relations $S(B_1, \cdots, B_m)$ and $T(C_1, \cdots, C_k)$ such that

1. $\{A_1, \cdots, A_n\} = \{B_1, \cdots, B_m\} \cup \{C_1, \cdots, C_k\}$

Decomposing Relations (3.3.2)

Given a relation $R(A_1, \cdots, A_n)$ we can *decompose* it into two relations $S(B_1, \cdots, B_m)$ and $T(C_1, \cdots, C_k)$ such that

1. $\{A_1, \cdots, A_n\} = \{B_1, \cdots, B_m\} \cup \{C_1, \cdots, C_k\}$
2. $S = \pi_{B_1, \cdots, B_m} R$

Decomposing Relations (3.3.2)

Given a relation $R(A_1, \cdots, A_n)$ we can *decompose* it into two relations $S(B_1, \cdots, B_m)$ and $T(C_1, \cdots, C_k)$ such that

1. $\{A_1, \cdots, A_n\} = \{B_1, \cdots, B_m\} \cup \{C_1, \cdots, C_k\}$
2. $S = \pi_{B_1, \cdots, B_m} R$
3. $T = \pi_{C_1, \cdots, C_k} R$

Decomposing Relations (3.3.2)

Given a relation $R(A_1, \cdots, A_n)$ we can *decompose* it into two relations $S(B_1, \cdots, B_m)$ and $T(C_1, \cdots, C_k)$ such that

1. $\{A_1, \cdots, A_n\} = \{B_1, \cdots, B_m\} \cup \{C_1, \cdots, C_k\}$
2. $S = \pi_{B_1, \cdots, B_m} R$
3. $T = \pi_{C_1, \cdots, C_k} R$

Decomposing Relations Example (3.14)

Table: "Movies2".

title	year	length	studio	genre
Star Wars	1977	124	Fox	sciFi
Gone With the Wind	1939	231	drama	MGM
Wayne's World	1992	95	comedy	Paramnt

Table: "Movies3".

title	year	star
Star Wars	1977	Carrie Fisher
Star Wars	1977	Mark Hamill
Gone With the Wind	1939	Vivien Leigh
Wayne's World	1992	Dana Carvey
Wayne's World	1992	Mike Meyers

Decomposing Relations Example (3.14)

We can check to make sure that these relations don't have any anomalies, but how can we be sure we didn't lose anything by decomposing things in this way?

Decomposing Relations Example (3.14)

We can check to make sure that these relations don't have any anomalies, but how can we be sure we didn't lose anything by decomposing things in this way?

We can reconstruct the original relation with the relational algebra query

Decomposing Relations Example (3.14)

We can check to make sure that these relations don't have any anomalies, but how can we be sure we didn't lose anything by decomposing things in this way?

We can reconstruct the original relation with the relational algebra query

$$\texttt{Movies2} \bowtie \texttt{Movies3}$$

Decomposing Relations Example (3.14)

We can check to make sure that these relations don't have any anomalies, but how can we be sure we didn't lose anything by decomposing things in this way?

We can reconstruct the original relation with the relational algebra query

$$\texttt{Movies2} \bowtie \texttt{Movies3}$$

or the corresponding SQL query

Decomposing Relations Example (3.14)

We can check to make sure that these relations don't have any anomalies, but how can we be sure we didn't lose anything by decomposing things in this way?

We can reconstruct the original relation with the relational algebra query

$$\texttt{Movies2} \bowtie \texttt{Movies3}$$

or the corresponding SQL query

```
SELECT * FROM Movies2 NATURAL JOIN Movies3;
```

or

```
SELECT * FROM Movies2
JOIN Movies3
  ON Movies2.year  = Movies3.year
 AND Movies2.title = Movies3.title;
```

Decomposing Relations Example (3.14)

Because we can recover the original relation, we say this is a *loseless decomposition*.

Boyce-Codd Normal Form (3.3.3)

Is it always possible to replace a relation containing anomalies with relations that do not exhibit anomalies?

Boyce-Codd Normal Form (3.3.3)

Is it always possible to replace a relation containing anomalies with relations that do not exhibit anomalies?

Yes. There is a simple condition on relations called Boyce-Codd Normal Form (BCNF) which guarantees that anomalies do not exist.

Definition: Boyce-Codd Normal Form (3.3.3)

Let $A = \{A_1, A_2, \ldots, A_n\}, B = \{B_1, B_2, \ldots, B_m\}$.

A relation R is in BCNF if whenever R satisfies a non-trivial functional dependency $A \rightarrow B$, then A is a superkey for R.

Example (3.15)

Table: A poorly designed relation "Movies1".

title	year	length	genre	studio	star
Star Wars	1977	124	sciFi	Fox	Carrie Fisher
Star Wars	1977	124	sciFi	Fox	Mark Hamill
Gone With the Wind	1939	231	drama	MGM	Vivien Leigh
Wayne's World	1992	95	comedy	Paramnt	Dana Carvey
Wayne's World	1992	95	comedy	Paramnt	Mike Meyers

What is a key for Movies1?

Example (3.15)

Table: A poorly designed relation "Movies1".

title	year	length	genre	studio	star
Star Wars	1977	124	sciFi	Fox	Carrie Fisher
Star Wars	1977	124	sciFi	Fox	Mark Hamill
Gone With the Wind	1939	231	drama	MGM	Vivien Leigh
Wayne's World	1992	95	comedy	Paramnt	Dana Carvey
Wayne's World	1992	95	comedy	Paramnt	Mike Meyers

What is a key for Movies1? $\{\texttt{title}, \texttt{year}, \texttt{star}\}$.

Example (3.15)

Table: A poorly designed relation "Movies1".

title	year	length	genre	studio	star
Star Wars	1977	124	sciFi	Fox	Carrie Fisher
Star Wars	1977	124	sciFi	Fox	Mark Hamill
Gone With the Wind	1939	231	drama	MGM	Vivien Leigh
Wayne's World	1992	95	comedy	Paramnt	Dana Carvey
Wayne's World	1992	95	comedy	Paramnt	Mike Meyers

What is a key for Movies1? $\{\mathtt{title}, \mathtt{year}, \mathtt{star}\}$.

However, $\{\mathtt{title}, \mathtt{year}\} \to \{\mathtt{length}, \mathtt{genre}, \mathtt{studio}\}$, and $\{\mathtt{title}, \mathtt{year}\}$ is not a super key.

Example (3.15)

Table: A poorly designed relation "Movies1".

title	**year**	**length**	**genre**	**studio**	**star**
Star Wars	1977	124	sciFi	Fox	Carrie Fisher
Star Wars	1977	124	sciFi	Fox	Mark Hamill
Gone With the Wind	1939	231	drama	MGM	Vivien Leigh
Wayne's World	1992	95	comedy	Paramnt	Dana Carvey
Wayne's World	1992	95	comedy	Paramnt	Mike Meyers

What is a key for Movies1? $\{\texttt{title}, \texttt{year}, \texttt{star}\}$.

However, $\{\texttt{title}, \texttt{year}\} \rightarrow \{\texttt{length}, \texttt{genre}, \texttt{studio}\}$, and $\{\texttt{title}, \texttt{year}\}$ is not a super key. Therefore the relation is not in BCNF.

A Problem With BCNF (3.4.4)

There are situations in which BCNF preserve functional dependencies while being a loseless decomposition.
(See section 3.4.4 for details.)

A Problem With BCNF (3.4.4)

There are situations in which BCNF preserve functional dependencies while being a loseless decomposition.
(See section 3.4.4 for details.)
The solution?

Definition: Third Normal Form (3.5.1)

Let $A = \{A_1, A_2, \ldots, A_n\}, B = \{B_1, B_2, \ldots, B_m\}$.

A relation R is in *third normal form* (3NF) if whenever $A \rightarrow B$ is nontrivial, either A is a superkey, or each $B_i \in B \backslash A$ is a member of some key.

Aside: What Are the Other Normal Forms?

Algorithm: Third Normal Form (3.5.2 *modified*)

`INPUT:` A relation R and a set F of functional dependencies for R.

Algorithm: Third Normal Form (3.5.2 *modified*)

`INPUT`: A relation R and a set F of functional dependencies for R.
`OUTPUT`: A decomposition of R into a collection of relations in 3NF.

Algorithm: Third Normal Form (3.5.2 *modified*)

`INPUT`: A relation R and a set F of functional dependencies for R.
`OUTPUT`: A decomposition of R into a collection of relations in 3NF.
`METHOD`:

1. Find a minimal basis for F, say G.

Algorithm: Third Normal Form (3.5.2 *modified*)

`INPUT`: A relation R and a set F of functional dependencies for R.
`OUTPUT`: A decomposition of R into a collection of relations in 3NF.
`METHOD`:

1. Find a minimal basis for F, say G.
2. For each $X \to A \in G$, XA is the schema of one of the relations in the decomposition.

Algorithm: Third Normal Form (3.5.2 *modified*)

`INPUT:` A relation R and a set F of functional dependencies for R.
`OUTPUT:` A decomposition of R into a collection of relations in 3NF.
`METHOD:`

1. Find a minimal basis for F, say G.
2. For each $X \to A \in G$, XA is the schema of one of the relations in the decomposition.
3. Remove redundant schemas.

Algorithm: Third Normal Form (3.5.2 *modified*)

`INPUT`: A relation R and a set F of functional dependencies for R.
`OUTPUT`: A decomposition of R into a collection of relations in 3NF.
`METHOD`:

1. Find a minimal basis for F, say G.
2. For each $X \to A \in G$, XA is the schema of one of the relations in the decomposition.
3. Remove redundant schemas.
4. If none of the relations generated in step 2 is a superkey for R, add another relation whose schema is a key for R.

Algorithm (Example): Third Normal Form (3.27)

`INPUT:` $R(A, B, C, D, E),\ AB \to C,\ C \to B,\ A \to D.$

Algorithm (Example): Third Normal Form (3.27)

`INPUT:` $R(A, B, C, D, E)$, $AB \to C$, $C \to B$, $A \to D$.

1. The functional dependencies are already a minimal basis.

Algorithm (Example): Third Normal Form (3.27)

`INPUT:` $R(A, B, C, D, E),\ AB \to C,\ C \to B,\ A \to D.$

1. The functional dependencies are already a minimal basis.
2. $S_1(A, B, C)$

Algorithm (Example): Third Normal Form (3.27)

`INPUT:` $R(A, B, C, D, E)$, $AB \to C$, $C \to B$, $A \to D$.

1. The functional dependencies are already a minimal basis.
2. $S_1(A, B, C)$ $S_2(B, C)$

Algorithm (Example): Third Normal Form (3.27)

`INPUT:` $R(A, B, C, D, E)$, $AB \to C$, $C \to B$, $A \to D$.

1. The functional dependencies are already a minimal basis.
2. $S_1(A, B, C)$ $S_2(B, C)$ $S_3(A, D)$.

Algorithm (Example): Third Normal Form (3.27)

`INPUT:` $R(A, B, C, D, E)$, $AB \to C$, $C \to B$, $A \to D$.

1. The functional dependencies are already a minimal basis.
2. $S_1(A, B, C)$ $S_2(B, C)$ $S_3(A, D)$.
3. Remove S_2 since $\{B, C\} \subset \{A, B, C\}$.

Algorithm (Example): Third Normal Form (3.27)

`INPUT:` $R(A, B, C, D, E)$, $AB \to C$, $C \to B$, $A \to D$.

1. The functional dependencies are already a minimal basis.
2. $S_1(A, B, C)$ $S_2(B, C)$ $S_3(A, D)$.
3. Remove S_2 since $\{B, C\} \subset \{A, B, C\}$.
4. Are either S_1 or S_3 superkeys?

Algorithm (Example): Third Normal Form (3.27)

INPUT: $R(A, B, C, D, E)$, $AB \to C$, $C \to B$, $A \to D$.

1. The functional dependencies are already a minimal basis.
2. $S_1(A, B, C)$ $S_2(B, C)$ $S_3(A, D)$.
3. Remove S_2 since $\{B, C\} \subset \{A, B, C\}$.
4. Are either S_1 or S_3 superkeys? Add $S_4(A, B, E)$.
5. The solution is $S_1(A, B, C), S_3(A, D), S_4(A, B, E)$.