ICCS240 Database Management

Integrity Constraints:
Functional Dependency (FD)

Many slides in this lecture are either from or adapted from slides provided by

Jeff Ullman, Stanford U

Werner Nutt, Free U of Bozen-Bolzano

Example of a bad relation

`Students(Id, Name, AdvisorId, AdvisorName, FavouriteAdvisorId)`

- If you know a student's `Id`, can you determine the values of any other attributes?

  `Id` → `Name`

  `Id` → `FavouriteAdvisorId`

  How about `AdvisorId` → `AdvisorName`?

- Suppose a student `Id` can have/be associated with multiple advisors `AdvisorID`.
- What is the key for the `Sudents` then?

  {`Id`, `AdvsiorId`}

Why is this relation bad?

Parts of the key determine other attributes.

Motivation for Functional Dependencies

- Reason about constraints on attributes in a relation
- Procedurally determine the keys of a relation
- Detect when a relation has redundant information
- Improve database designs systematically using normalization

Functional Dependencies (FD)

A functional dependency (or FD) is written

$$\boldsymbol{X} \rightarrow A \text{ or } \boldsymbol{X} \rightarrow \boldsymbol{Y}$$

The attribute on the left side of the FD is called **determinant**.

Notation: $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}$ represents sets of attributes

$A, B, C, \ldots$ represent single attributes

$\boldsymbol{X} \rightarrow A$ ("$\boldsymbol{X}$ functionally determines A") is an assertion about a relation R:

Whenever two tuples of R agree on all the attributes of $\boldsymbol{X}$,

then they must also agree on the attribute A

$$t_1[X] = t_2[X] \text{ implies } t_1[A] = t_2[A], \text{ for all } t_1, t_2 \in R$$

Convention: We say "$\boldsymbol{X} \rightarrow A$ holds in (relation) R"

Notation: No set brackets in sets of attributes: just ABC rather than $\{A, B, C\}$.

Example: FD's to asserts over Drinkers

Drinkers(name, addr, beers_liked, manf, fav_beer)

name	addr	beers_liked	manf	fav_beer
Alice	Voyager	Bud	A.B.	WickedAle
Alice	Voyager	WickedAle	Pete's	WickedAle
Bob	Enterprise	Bud	A.B.	Bud

Reasonable FD's to assert:

name → addr

beers_liked → manf

name → fav_beer

FDs with multiple attributes

FDs with **more than one** attribute on the right don't increase expressivity …

… but allow for convenient shorthands that combine FDs

Example: name → addr and name → fav_beer

become name → addr fav_beer

More than one attribute on the left, however, may be essential

Example: bar beer → price

Keys of Relations

Given relation R and a set K of attributes of R,

K is a **superkey** for relation R if K *functionally determines* all of R

K is a key for R if K is a superkey,

but there is no proper subset of K that is a super key.

(That is, K is a *minimal* super key.)

Sometimes we call "keys" also "candidate keys", to indicate they are candidates for choosing the primary key

Example

name	addr	beers_liked	manf	fav_beer
Alice	Voyager	Bud	A.B.	WickedAle
Alice	Voyager	WickedAle	Pete's	WickedAle
Bob	Enterprise	Bud	A.B.	Bud

Drinkers(name, addr, beers_liked, manf, fav_beer)

We have

$$\text{name} \rightarrow \text{addr fav\_beer}$$

$$\text{beers\_liked} \rightarrow \text{manf}$$

Therefore, {name, beers_like} determine all the other attributes

Hence, {name, beers_like} is a *superkey*

Example (cont.)

name	addr	beers_liked	manf	fav_beer
Alice	Voyager	Bud	A.B.	WickedAle
Alice	Voyager	WickedAle	Pete’s	WickedAle
Bob	Enterprise	Bud	A.B.	Bud

Neither {name} nor {beers_liked} is a superkey:

name → manf doesn’t hold

beers_liked → addr doesn’t hold

Hence: **{name, beers_liked}** is a key

There are no other keys, but many other superkeys.

In fact, any superset of {name, beers_liked} is a superkey.

ER and Relational Keys

- Keys in ER concern **entities**
- Keys in relations concern **tuples**
- Usually, one tuple corresponds to one entity, so the ideas are similar
- But – in poor relational designs,
  one tuple may represent several entities
  … so ER keys and relational keys are different

Example Data: Drinkers

name	addr	beers_liked	manf	fav_beer
Alice	Voyager	Bud	A.B.	WickedAle
Alice	Voyager	WickedAle	Pete's	WickedAle
Bob	Enterprise	Bud	A.B.	Bud

Relational key = {name, beers_liked}

But in E/R,

name is a key for Drinkers

beers_liked is a key for Beers

Where do keys come from?

1. We could simply assert a key K.
   Then the only FD's are $K \rightarrow A$ for all attributes A,
   and K turns out to be the only key obtainable from the FD's.

2. We could assert FD's and deduce the keys by systematic exploration.
   - E/R model gives us FDs from entity-set keys and from man-one relationships

FDs from “Physics”

- While most FDs come from E/R keyness and many-one relationships, some are simply from physical laws (or our domain knowledge)

- Example: “no two courses can meet in the same room at the same time” tells us:

$$\text{hour room} \rightarrow \text{course}$$

Inferring FDs – Motivation

We are given FDs

$$X_1 \to A_1, X_2 \to A_2, \ldots, X_n \to A_n$$

and we want to know whether an FD

$$Y \to B$$

must hold in any relation that satisfies the given FDs.

Example: If $A \to B$ and $B \to C$ hold, then surely $A \to C$ holds

Important for design of good relation schemas

Inference Test

```
    <--Y-->abc...d
T1: 000...0000...0
T2: 000...0???...?
```

To test if $Y \rightarrow B$, …

- Start assuming two tuples agree in all attributes of Y.
- Use the given FDs to infer that these tuples must also agree in certain other attributes.
  - If B is eventually found to be one of these attributes, then $Y \rightarrow B$ is true.
  - Otherwise, the two tuples, with any forced equalities form a two-tuple relation that proves $Y \rightarrow B$ does not follow from the given FDs.

Closure Test – an easier way

- An easier test is based on the concept of attribute closure.
- Let R be a relation, $\mathcal{F}$ be a set of FDs over R, and Y be a set of attributes of R
- The **closure** of Y with respect to $\mathcal{F}$, denoted Y^+,
  consists of all attributes that are determined by Y given $\mathcal{F}$.
- Observation: $Y \rightarrow B$ follows from $\mathcal{F}$ **iff** $B \in Y^+$

- **Basic**: $Y^+ \coloneqq Y$.
- **Induction**:

  Look for an FD's whose left side X is a *subset* of the current Y^+.

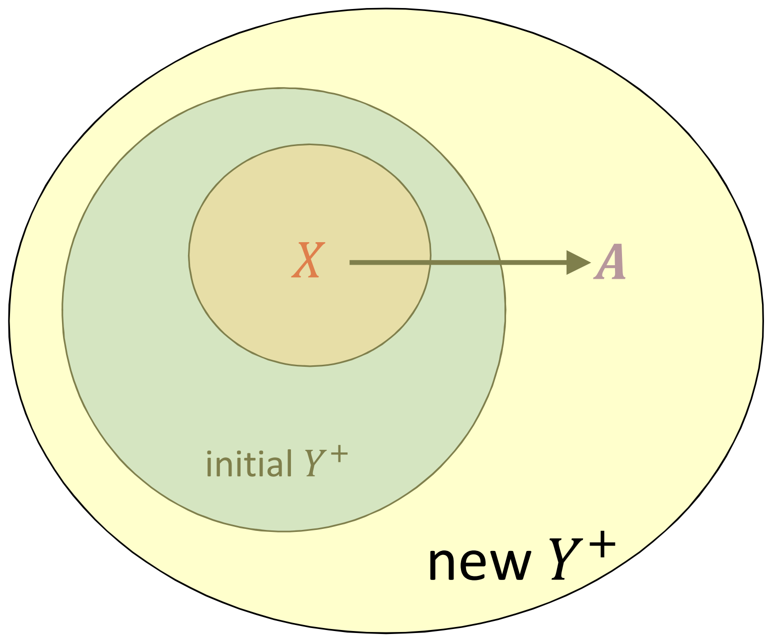
  If the FD is $X \rightarrow A$, add A to Y^+.

Example

Given a relation $R = \{A, B, C, D, E, F, G\}$
having a set of FDs $\mathcal{F} = \{A \to BC, B \to DF, E \to F\}$,
what is the closure of A?

- Basic: $A^+ = \{A\}$
- Induction step1: $A \to BC \in \mathcal{F}$, then $A^+ = \{A, B, C\}$
- Induction step2: $B \to DF \in \mathcal{F}$, then $A^+ = \{A, B, C, D, F\}$
- …
- $A^+ = R \setminus \{E, G\}$

Closure Test – Idea



Finding all implied FDs

We know how we can determine whether one FD follows from a set of FDs $\mathcal{F} = \{X_1 \rightarrow A_1, \dots, X_n \rightarrow A_n\}$

Question: How can we find **all such FDs**?

Motivation: To get a better schema, we "normalize", i.e.,

we break one relation schema

into two or more schema

Example: Finding all implied FDs

- Relation R with attributes $ABCD$ with FDs
$$AB \to C, C \to D, D \to A$$

- Decompose R into ABC, AD

- *Question*: What FDs hold in ABC
- *Answer*: Not only $AB \to C$, but also $C \to A$!

Why $C \rightarrow A$?

ABCD	$a_1b_1cd_1$	$a_2b_2cd_2$
comes from	$\downarrow$	$\downarrow$
ABC	a_1b_1c	a_2b_2c

$d_1=d_2$ because $C \rightarrow D$

$a_1=a_2$ because $D \rightarrow A$

Thus, tuples in the projection with equal *C*s have equal *A*s: $C \rightarrow A$.

Projecting FDs

How can we find the FDs that hold on the projection of R?

Basic Idea:

Attributes in left and right sides are disjoint

1. Start with the given FDs
2. Find all **non-trivial** FDs that follow from the given FDs
3. Restrict to those FDs that involve **only attributes** of the **projected** schema

An (exponential) algorithm to find projecting FDs

1. For each set of attribute X, compute X^+
2. Add $X \to A$ for all $A \in X^+ \setminus X$
3. However, drop $XY \to A$ whenever we discover $X \to A$

   (Because $XY \to A$ follows from $X \to A$ in any projection)

4. Finally, return only FDs involving projected attributes.

Optimization – a few tricks

Suppose that Z is the set of all attributes of R. Then

- $\emptyset^+ = \emptyset$
- $Z^+ = Z$
- If $X \subseteq Y$ and $X^+ = Z$, then $Y^+ = Z$.

Example

Relation ABC with FDs $A \to B$ and $B \to C$

Project onto AC

- $A^+ = \{A, B, C\}$ → $A \to B, A \to C$
  (optimization: we do not need to compute AB^+ or AC^+)
- $B^+ = \{B, C\}$ → $B \to C$
- $C^+ = \{C\}$ → nothing
- $BC^+ = \{B, C\}$ → nothing

Resulting FDs: $A \to B$, $A \to C$ and $B \to C$.

Projection onto AC: $A \to C$. (The only FD that involves a subset of AC.)

A Geometric View of FDs

- Imagine the set of all **instances** of a particular relation.
- That is, all finite sets of tuples that have the proper number of components.
- Each instance is a point in this space.

Example: $R(A, B)$

{(1,2), (3,4)}

{}

{(5,1)}

{(1,2), (3,4), (1,3)}

An FD is a subset of instances

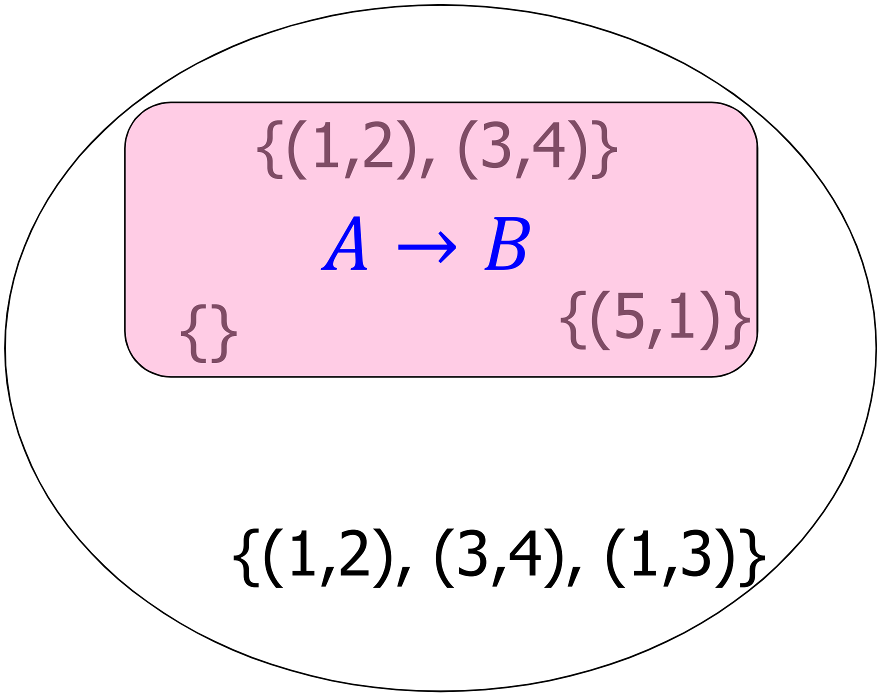
- For each FD $X \rightarrow A$, there is a subset of all instances that satisfy the FD.

- We can represent an FD by a region in the space

- ***Trivial FD***: an FD that is represented by the entire space

  Example: $A \rightarrow A$

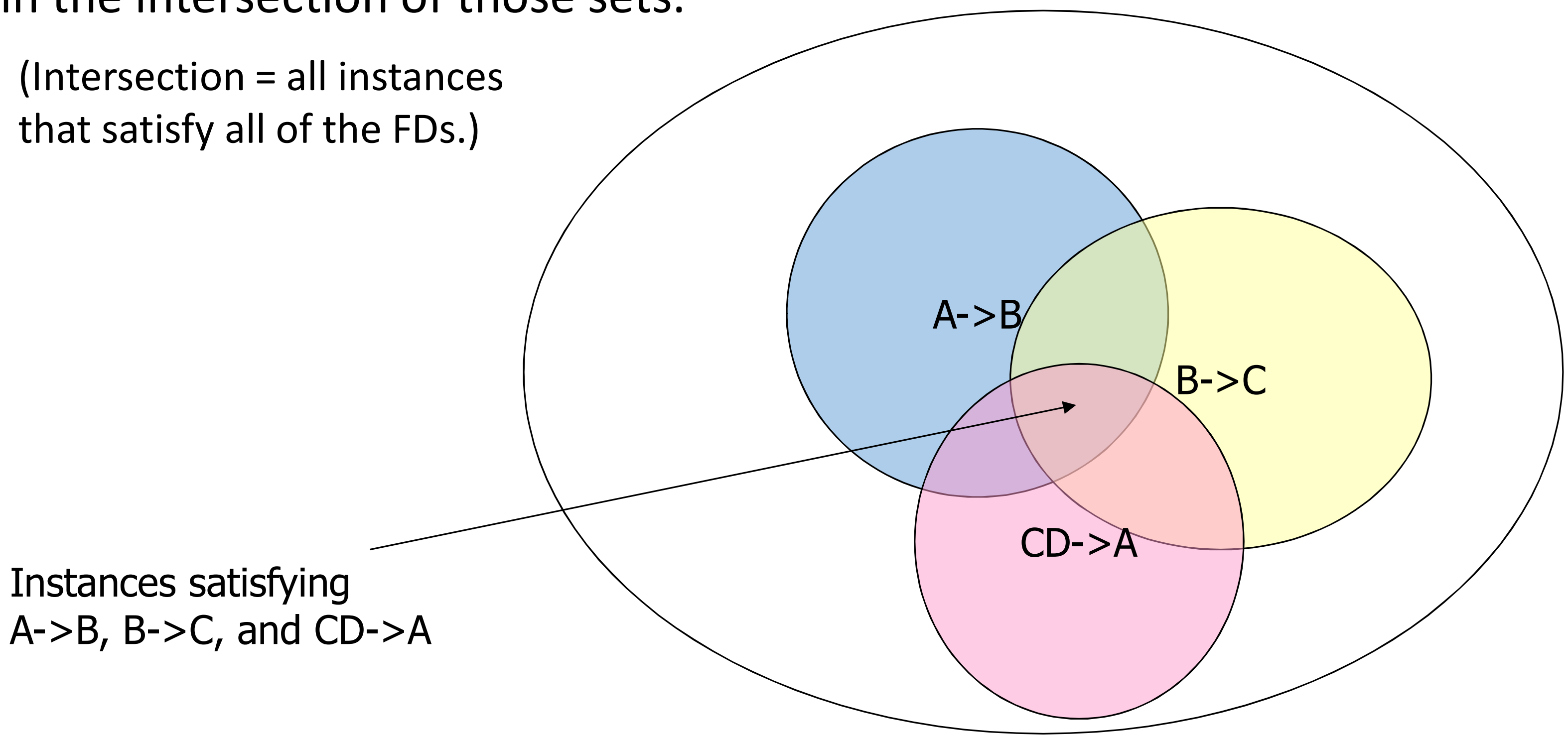
Example: $A \rightarrow B$ for $R(A, B)$



Representing Sets of FDs

If each FD is a set of relation instances, then a collection of FDs corresponds to the in the intersection of those sets.

(Intersection = all instances that satisfy all of the FDs.)



(Geometrically) Implication of FDs

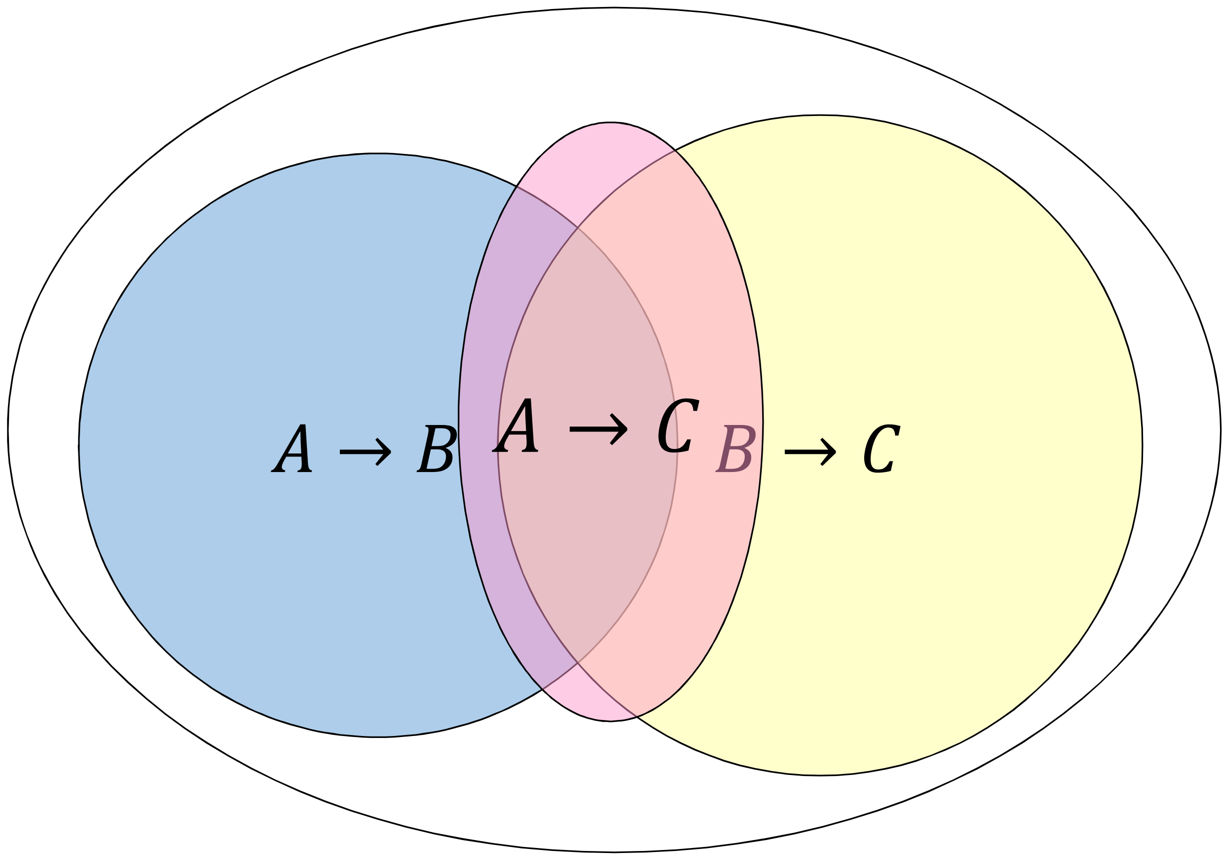
$\mathcal{F} = \{X_1 \to A_1, \ldots, X_n \to A_n\}$ set of FDs

- An FD $Y \to B$ follows from $\mathcal{F}$ or is implied by $\mathcal{F}$
  if every instance that satisfies all FDs in $\mathcal{F}$ also satisfies $Y \to B$

This can be visualized:

- If $Y \to B$ follows from the set $\mathcal{F} = \{X_1 \to A_1, \ldots, X_n \to A_n\}$,
  then in the space of instances the **region for** $Y \to B$ must **include** the **intersection of the regions for the** FDs $X_i \to A_i$.
- That is,
  - Every instance satisfying all the $X_i \to A_i$ surely satisfies $Y \to B$.
  - But an instance could satisfy $Y \to B$, yet not be in this intersection.

Example



Finding Keys from FDs

An (exponential) algorithm for computing all keys

Given a relation $R(A_1, \dots, A_n)$ and the set $\mathcal{F}$ of all FDs that hold in R, we can use the following algorithm to compute all possible keys for R.

1. For every subset $K \subseteq \{A_1, \dots, A_n\}$ compute K^+.
2. If $K^+ = \{A_1, \dots, A_n\}$
   and for every attribute A, $(K - \{A\})^+ \neq \{A_1, \dots, A_n\}$,
   then output K as a key.

Source: T.M. Murali

More on Inference Rules

Inference Rules

- The Armstrong’s axioms are the basic inference rule.
- Armstrong’s axioms are used to conclude functional dependencies on a relational database.
- The inference rule is a type of assertion. It can apply to a set of FDs.
- Using the inference rule, we can derive additional FDs from the initial set.

6 types of Inference Rule:

1. Reflexive Rule: If $X \supseteq Y$, then $X \rightarrow Y$
2. Augmentation Rule: If $X \rightarrow Y$, then $XZ \rightarrow YZ$
3. Transitive Rule: If $X \rightarrow Y$ and $Y \rightarrow Z$, then $X \rightarrow Z$
4. Union Rule: If $X \rightarrow Y$ and $X \rightarrow Z$, then $X \rightarrow YZ$
5. Decomposition Rule: If $X \rightarrow YZ$, then $X \rightarrow Y$ and $X \rightarrow Z$
6. Pseudo Transitive Rule: If $X \rightarrow Y$ and $YZ \rightarrow W$, then $XZ \rightarrow W$

Minimal Basis

Minimal Basis

- A relation may have a large set of equivalent sets of FDs.
- If we are given a set $\mathcal{F}$ of FDs, then any set of FDs that is equivalent to $\mathcal{F}$ is called a **basis** of $\mathcal{F}$.

A set $\mathcal{B}$ of FDs is a **minimal basis** for a relation R if

1. Every FD in $\mathcal{B}$ has one attribute on the right hand side.
2. If we remove any FD from $\mathcal{B}$,
   then the result is not a basis.
3. For any FD in $\mathcal{B}$,
   if we remove one or more attribute from the *left hand side* of the FD,
   then the result is not a basis.

Example of Minimal Basis

- $R(A, B, C)$ is a relation such that each attribute functionally determines the other two attributes

- FDs : $A \to BC, B \to AC, C \to AB,$

  $A \to B, A \to C, B \to A, B \to C, C \to A, C \to B,$

  $AB \to C, BC \to A, AC \to B, \ldots$

- Minimal bases :

  $\{A \to B, B \to A, B \to C, C \to B\},$
  $\{A \to B, B \to C, C \to A\}, \ldots$

Adapt from source: T.M. Murali

Exercise

Given a relation $R(A, B, C, D, E, F, G)$ with the following FDs $\mathcal{F}$:

(1) $A \rightarrow BC$, (2) $E \rightarrow CF$, (3) $B \rightarrow E$, (4) $CD \rightarrow EF$, (5) $A \rightarrow G$

Answer the following questions:

1. Find the closure of A
2. Find the closure of G
3. Find a candidate key for R

More exercise (or HW?)

`Contracts(cno, suppNo, projNo, deptNo, partNo, qty, value)`

Short: $C\ S\ Pr\ D\ Pa\ Q\ V$

A designer has found the following set of FDs:

- C is a key, i.e., $C \rightarrow SPrDPaQV$
- A project purchases each part using a single contract, $PrPa \rightarrow C$
- A department purchases at most on part from a supplier, $SD \rightarrow Pa$

His colleague has come up with a slightly different set:

- A project purchases each part using a single contract. $PrPa \rightarrow C$
- A contract determines project, supplier and department. $C \rightarrow PrSD$
- $SPrD$ is a key, $SDPr \rightarrow CPaQV$

Are the findings of the second designer different from those of the first?