Week 5 Workshop

Alice: Your model reduces the most interesting information to something flat and boring.

Vittorio: You're right, and this causes a lot of problems.

Sergio: Designing the schema for a complex application is tough, and it is easy to make mistakes when updating a database.

Riccardo: Also, the system knows so little about the data that it is hard to obtain good performance.

Alice: Are you telling me that the model is bad?

Vittorio: No, wait, we are going to fix it!

(Foundations of Databases, S. Abiteboul, R. Hull, V. Vianu, Addison-Wesley, 1995)

Housekeeping

1. **Assignment 1 (SQL)** (due 11:59pm 3 Sep 2021)

Housekeeping

1. **Assignment 1 (SQL)** (due 11:59pm 3 Sep 2021)
   - Which directors have collaborated with at least two different writers? (Clarification: two different writers, not including director themselves)

Housekeeping

1. **Assignment 1 (SQL)** (due 11:59pm, 3 Sep 2021)
   - Which directors have collaborated with at least two different writers? (Clarification: two different writers, not including director themselves)
   - Among those directors who have never won any director award, who directed the largest number of movies? (Clarification: Among . . .)

Housekeeping

1. **Assignment 1 (SQL)** (due 11:59pm 3 Sep 2021)
   - Which directors have collaborated with at least two different writers? (Clarification: two different writers, not including director themselves)
   - Among those directors who have never won any director award, who directed the largest number of movies? (Clarification: Among . . .)
   - Pay attention to which attributes you need to list, whether you need to order the tuples, syntax issues,etc. (Partial marks may be awarded)
   - **Do not wait until the last minute to check/submit your solution.** (Refer to the instructions in the assignment specification.)

Housekeeping

1. **Assignment 1 (SQL)** (due 11:59pm 3 Sep 2021)
   - Which directors have collaborated with at least two different writers? (Clarification: two different writers, not including director themselves)
   - Among those directors who have never won any director award, who directed the largest number of movies? (Clarification: Among . . .)
   - Pay attention to which attributes you need to list, whether you need to order the tuples, syntax issues,etc. (Partial marks may be awarded)
   - **Do not wait until the last minute to check/submit your solution.** (Refer to the instructions in the assignment specification.)

2. **More consultation hours in Week 6**
   - Aug 30 (Mon) 2-3 pm
   - Aug 31 (Tue) 2-3 pm
   - Sep 1 (Wed) 8-9 pm

Housekeeping

1. **Assignment 1 (SQL)** (due 11:59pm 3 Sep 2021)
   - Which directors have collaborated with at least two different writers? (Clarification: two different writers, not including director themselves)
   - Among those directors who have never won any director award, who directed the largest number of movies? (Clarification: Among . . .)
   - Pay attention to which attributes you need to list, whether you need to order the tuples, syntax issues,etc. (Partial marks may be awarded)
   - **Do not wait until the last minute to check/submit your solution.** (Refer to the instructions in the assignment specification.)
2. **More consultation hours in Week 6**
   - Aug 30 (Mon) 2-3 pm
   - Aug 31 (Tue) 2-3 pm
   - Sep 1 (Wed) 8-9 pm

Update Anomalies

- **What could happen to insert, delete and update operations?**

ENROLMENT					
Name	StudentID	DoB	CourseNo	Semester	Unit
Tom	123456	25/01/1989	COMP2400	2010 S2	6
Tom	123456	25/01/1989	COMP8740	2011 S2	12
Michael	123458	21/04/1985	COMP2400	2009 S2	6
Michael	123458	21/04/1985	COMP8740	2011 S2	12
Fran	123457	11/09/1987	COMP2400	2009 S2	6

Update Anomalies

- **What could happen to insert, delete and update operations?**

ENROLMENT					
Name	StudentID	DoB	CourseNo	Semester	Unit
Tom	123456	25/01/1989	COMP2400	2010 S2	6
Tom	123456	25/01/1989	COMP8740	2011 S2	12
Michael	123458	21/04/1985	COMP2400	2009 S2	6
Michael	123458	21/04/1985	COMP8740	2011 S2	12
Fran	123457	11/09/1987	COMP2400	2009 S2	6

- **Insertion anomalies:** If inserting a new course COMP3000, then ...

Update Anomalies

- **What could happen to insert, delete and update operations?**

ENROLMENT					
Name	StudentID	DoB	CourseNo	Semester	Unit
Tom	123456	25/01/1989	COMP2400	2010 S2	6
Tom	123456	25/01/1989	COMP8740	2011 S2	12
Michael	123458	21/04/1985	COMP2400	2009 S2	6
Michael	123458	21/04/1985	COMP8740	2011 S2	12
Fran	123457	11/09/1987	COMP2400	2009 S2	6

- **Insertion anomalies:** If inserting a new course COMP3000, then ... (i.e., cannot insert NULL values into Course because of the entity integrity constraint).

Update Anomalies

- **What could happen to insert, delete and update operations?**

ENROLMENT					
Name	StudentID	DoB	CourseNo	Semester	Unit
Tom	123456	25/01/1989	COMP2400	2010 S2	6
Tom	123456	25/01/1989	COMP8740	2011 S2	12
Michael	123458	21/04/1985	COMP2400	2009 S2	6
Michael	123458	21/04/1985	COMP8740	2011 S2	12
Fran	123457	11/09/1987	COMP2400	2009 S2	6

- **Insertion anomalies**: If inserting a new course COMP3000, then ... (i.e., cannot insert NULL values into Course because of the entity integrity constraint).
- **Deletion anomalies**: If deleting the enrolled course COMP2400 of Fran, then ...

Australian National University

Update Anomalies

- **What could happen to insert, delete and update operations?**

ENROLMENT

Name	StudentID	DoB	CourseNo	Semester	Unit
Tom	123456	25/01/1989	COMP2400	2010 S2	6
Tom	123456	25/01/1989	COMP8740	2011 S2	12
Michael	123458	21/04/1985	COMP2400	2009 S2	6
Michael	123458	21/04/1985	COMP8740	2011 S2	12
Fran	123457	11/09/1987	COMP2400	2009 S2	6

- **Insertion anomalies**: If inserting a new course COMP3000, then ... (i.e., cannot insert NULL values into Course because of the entity integrity constraint).
- **Deletion anomalies**: If deleting the enrolled course COMP2400 of Fran, then ... the personal information of Fran, such as DoB, will be lost as well.

Update Anomalies

- **What could happen to insert, delete and update operations?**

ENROLMENT

Name	StudentID	DoB	CourseNo	Semester	Unit
Tom	123456	25/01/1989	COMP2400	2010 S2	6
Tom	123456	25/01/1989	COMP8740	2011 S2	12
Michael	123458	21/04/1985	COMP2400	2009 S2	6
Michael	123458	21/04/1985	COMP8740	2011 S2	12
Fran	123457	11/09/1987	COMP2400	2009 S2	6

- **Insertion anomalies**: If inserting a new course COMP3000, then ... (i.e., cannot insert NULL values into Course because of the entity integrity constraint).
- **Deletion anomalies**: If deleting the enrolled course COMP2400 of Fran, then ... the personal information of Fran, such as DoB, will be lost as well.
- **Modification anomalies**: If changing the DoB of Michael, then ...

Update Anomalies

- **What could happen to insert, delete and update operations?**

ENROLMENT					
Name	StudentID	DoB	CourseNo	Semester	Unit
Tom	123456	25/01/1989	COMP2400	2010 S2	6
Tom	123456	25/01/1989	COMP8740	2011 S2	12
Michael	123458	21/04/1985	COMP2400	2009 S2	6
Michael	123458	21/04/1985	COMP8740	2011 S2	12
Fran	123457	11/09/1987	COMP2400	2009 S2	6

- **Insertion anomalies**: If inserting a new course COMP3000, then ... (i.e., cannot insert NULL values into Course because of the entity integrity constraint).
- **Deletion anomalies**: If deleting the enrolled course COMP2400 of Fran, then ... the personal information of Fran, such as DoB, will be lost as well.
- **Modification anomalies**: If changing the DoB of Michael, then ... update every tuple that records the DoB of this student.

Update Anomalies?

ENROLMENT

Name	StudentID	DoB	CourseNo	Semester	Unit
Tom	123456	25/01/1988	COMP2400	2010 S2	6
Tom	123456	25/01/1988	COMP8740	2011 S2	12
Michael	123458	21/04/1985	COMP2400	2009 S2	6
Michael	123458	21/04/1985	COMP8740	2011 S2	12
Fran	123457	11/09/1987	COMP2400	2009 S2	6

⇓

STUDENT

Name	StudentID	DoB
Tom	123456	25/01/1988
Michael	123458	21/04/1985
Fran	123457	11/09/1987

COURSE

CourseNo	Unit
COMP2400	6
COMP8740	12

ENROL

StudentID	CourseNo	Semester
123456	COMP2400	2010 S2
123456	COMP8740	2011 S2
123458	COMP2400	2009 S2
123458	COMP8740	2011 S2
123457	COMP2400	2009 S2

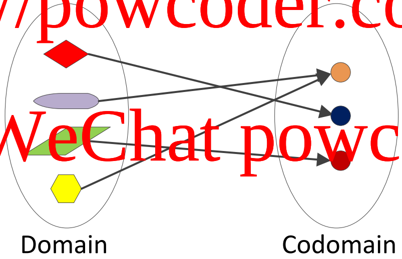
Why Functional Dependencies?

FDs tell us "relationship between and among attributes"!

- FDs are developed to define the **goodness** and **badness** of (relational) database design in a formal way.
  - **Top down**: start with a relation schema and FDs, and produce smaller relation schemas in certain normal form (called *normalisation*).
  - **Bottom up**: start with attributes and FDs, and produce relation schemas (*not popular in practice*).

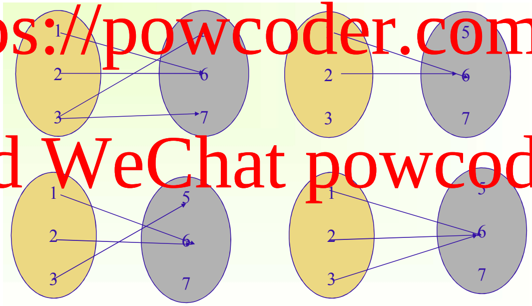
What is "Functional" about Functional Dependencies?

- The notion of functional dependency is very close to the notion of function.
- A (total) **function** $f : X \rightarrow Y$ describes a relationship between two sets X and Y such that each element of X is mapped to a unique element of Y.



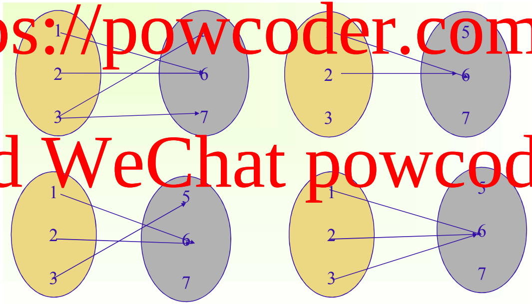
What is "Functional" about Functional Dependencies?

- A (total) **function** $f : X \to Y$ describes a relationship between two sets X and Y such that each element of X is mapped to a unique element of Y.
- **Exercise:** *which of them represent a function?*



What is "Functional" about Functional Dependencies?

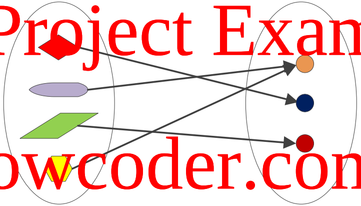
- A (total) **function** $f : X \rightarrow Y$ describes a relationship between two sets X and Y such that each element of X is mapped to a unique element of Y.
- **Exercise:** *which of them represent a function?*



Answer: **The ones at the bottom.**

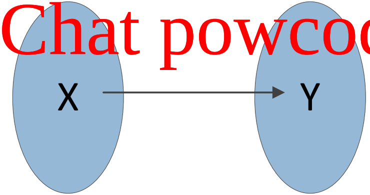
Functions vs Functional Dependencies

Function



Domain Codomain

Functional dependency



Determinant Dependant

Functions vs Functional Dependencies

$$f(x) = x^2$$

Functions vs Functional Dependencies

$$f(x) = x^2$$

x	$f(x)$
1	1
2	4
3	9
4	16
5	25
6	36
...	...

Functions vs Functional Dependencies

$$f(x) = x^2$$

x	$f(x)$
1	1
2	4
3	9
4	16
5	25
6	36
...	...

$$X \rightarrow f(x)$$

Formal Definition

- Let R be a relation schema.
  - A **FD** on R is an expression $X \rightarrow Y$ with attribute sets $X, Y \subseteq R$.
  - A relation $r(R)$ **satisfies $X \rightarrow Y$ on R** if, for any two tuples $t_1, t_2 \in r(R)$, whenever the tuples t_1 and t_2 coincide on values of X, they also coincide on values of Y.

$$t_1[X] = t_2[X]$$
$$\Downarrow$$
$$t_1[Y] = t_2[Y]$$

- A FD is **trivial** if it can *always* be satisfied, e.g.,
  - $\{A, B\} \rightarrow \{A\}$
  - $\{A, B, C\} \rightarrow \{A, B, C\}$

- **Syntactical convention**: (1) Instead of $\{A, B, C\}$, we may use ABC. (2) $A, B, \ldots$ for individual attributes and $X, Y, \ldots$ for sets of attributes.

Exercise - Functional Dependencies

- A functional dependency specifies a constraint on the relation schema that must hold **at all times**.
- Consider the following relation with attributes {A,B,C,D,E}. Do they satisfy the given FDs?

r(*R*)

A	B	C	D	E
1	2	3	4	5
1	2	2	2	2
1	2	3	2	3
2	2	2	4	4

Exercise - Functional Dependencies

- A functional dependency specifies a constraint on the relation schema that must hold **at all times**.
- Consider the following relation with attributes {A,B,C,D,E}. Do they satisfy the given FDs?

$r(R)$

A	B	C	D	E
1	2	3	4	5
1	2	2	2	2
1	2	3	2	3
2	2	2	4	4

1. $ABC \rightarrow AB$

Exercise - Functional Dependencies

- A functional dependency specifies a constraint on the relation schema that must hold **at all times**.
- Consider the following relation with attributes $\{$A,B,C,D,E$\}$. Do they satisfy the given FDs?

$r(R)$

A	B	C	D	E
1	2	3	4	5
1	2	2	2	2
1	2	3	2	3
2	2	2	4	4

1. $ABC \rightarrow AB$ Yes.

Exercise - Functional Dependencies

- A functional dependency specifies a constraint on the relation schema that must hold **at all times**.

- Consider the following relation with attributes $\{$A,B,C,D,E$\}$. Do they satisfy the given FDs?

$r(R)$

A	B	C	D	E
1	2	3	4	5
1	2	2	2	2
1	2	3	2	3
2	2	2	4	4

1. $ABC \rightarrow AB$ Yes.
2. $ABC \rightarrow D$

Exercise - Functional Dependencies

- A functional dependency specifies a constraint on the relation schema that must hold **at all times**.
- Consider the following relation with attributes $\{A,B,C,D,E\}$. Do they satisfy the given FDs?

$r(R)$

A	B	C	D	E
1	2	3	4	5
1	2	2	2	2
1	2	3	2	3
2	2	2	4	4

1. $ABC \rightarrow AB$ Yes.
2. $ABC \rightarrow D$ No.

Exercise - Functional Dependencies

- A functional dependency specifies a constraint on the relation schema that must hold **at all times**.
- Consider the following relation with attributes {A,B,C,D,E}. Do they satisfy the given FDs?

$r(R)$

A	B	C	D	E
1	2	3	4	5
1	2	2	2	2
1	2	3	2	3
2	2	2	4	4

1. $ABC \rightarrow AB$ Yes.
2. $ABC \rightarrow D$ No.
3. $E \rightarrow ABCD$

Exercise - Functional Dependencies

- A functional dependency specifies a constraint on the relation schema that must hold **at all times**.
- Consider the following relation with attributes $\{$A,B,C,D,E$\}$. Do they satisfy the given FDs?

$r(R)$

A	B	C	D	E
1	2	3	4	5
1	2	2	2	2
1	2	3	2	3
2	2	2	4	4

1. $ABC \rightarrow AB$ Yes.
2. $ABC \rightarrow D$ No.
3. $E \rightarrow ABCD$ Yes.

Exercise - Functional Dependencies

- A functional dependency specifies a constraint on the relation schema that must hold **at all times**.
- Consider the following relation with attributes $\{$A,B,C,D,E$\}$. Do they satisfy the given FDs?

$r(R)$

A	B	C	D	E
1	2	3	4	5
1	2	2	2	2
1	2	3	2	3
2	2	2	4	4

1. $ABC \rightarrow AB$ Yes.
2. $ABC \rightarrow D$ No.
3. $E \rightarrow ABCD$ Yes.

How to Identify FDs in General?

- A functional dependency specifies a constraint on the relation schema that must hold **at all times**.

How to Identify FDs in General?

- A functional dependency specifies a constraint on the relation schema that must hold **at all times**.

- In real-life applications, we often use the following approaches:

How to Identify FDs in General?

- A functional dependency specifies a constraint on the relation schema that must hold **at all times**.

- In real-life applications, we often use the following approaches:

  (1) **Analyse data requirements**
  Can be provided in the form of discussion with application users and/or data requirement specifications

How to Identify FDs in General?

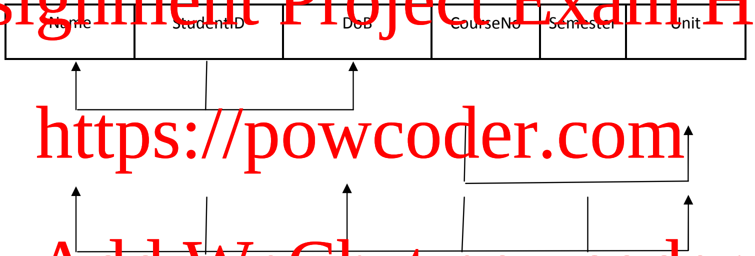
- A functional dependency specifies a constraint on the relation schema that must hold **at all times**.

- In real-life applications, we often use the following approaches:

  (1) **Analyse data requirements**
  Can be provided in the form of discussion with application users and/or data requirement specifications.

  (2) **Analyse sample data**
  Useful when application users are unavailable for consultation and/or the document is incomplete.

(1) Analyse Data Requirements and FD Diagram



- StudentID $\rightarrow$ Name, DoB;
- CourseNo $\rightarrow$ Unit;
- StudentID, CourseNo, Semester $\rightarrow$ Name, DoB, Unit.

(2) Analyse Sample Data

- Can you find some FDs on ENROLMENT based on the sample data?

(2) Analyse Sample Data

- Can you find some FDs on ENROLMENT based on the sample data?

ENROLMENT					
Name	StudentID	DoB	CourseNo	Semester	Unit
Tom	123456	25/01/1988	COMP2400	2010 S2	6
Tom	123456	25/01/1988	COMP8740	2011 S2	12
Michael	123458	21/04/1985	COMP2400	2009 S2	6
Michael	123458	21/04/1985	COMP8740	2011 S2	12
Fran	123457	11/09/1987	COMP2400	2009 S2	6

(2) Analyse Sample Data

- Can you find some FDs on ENROLMENT based on the sample data?

ENROLMENT

Name	StudentID	DoB	CourseNo	Semester	Unit
Tom	123456	25/01/1988	COMP2400	2010 S2	6
Tom	123456	25/01/1988	COMP8740	2011 S2	12
Michael	123458	21/04/1985	COMP2400	2009 S2	6
Michael	123458	21/04/1985	COMP8740	2011 S2	12
Fran	123457	11/09/1987	COMP2400	2009 S2	6

- We may have:

(2) Analyse Sample Data

- Can you find some FDs on ENROLMENT based on the sample data?

ENROLMENT

Name	<u>StudentID</u>	DoB	<u>CourseNo</u>	<u>Semester</u>	Unit
Tom	123456	25/01/1988	COMP2400	2010 S2	6
Tom	123456	25/01/1988	COMP8740	2011 S2	12
Michael	123458	21/04/1985	COMP2400	2009 S2	6
Michael	123458	21/04/1985	COMP8740	2011 S2	12
Fran	123457	11/09/1987	COMP2400	2009 S2	6

- We may have:
  - $\{StudentID\} \rightarrow \{Name, DoB\}$;

(2) Analyse Sample Data

- Can you find some FDs on ENROLMENT based on the sample data?

ENROLMENT					
Name	StudentID	DoB	CourseNo	Semester	Unit
Tom	123456	25/01/1988	COMP2400	2010 S2	6
Tom	123456	25/01/1988	COMP8740	2011 S2	12
Michael	123458	21/04/1985	COMP2400	2009 S2	6
Michael	123458	21/04/1985	COMP8740	2011 S2	12
Fran	123457	11/09/1987	COMP2400	2009 S2	6

- We may have:
  - $\{$StudentID$\} \rightarrow \{$Name, DoB$\}$;
  - $\{$StudentID, Name$\} \rightarrow \{$DoB$\}$;

(2) Analyse Sample Data

- Can you find some FDs on ENROLMENT based on the sample data?

ENROLMENT

Name	StudentID	DoB	CourseNo	Semester	Unit
Tom	123456	25/01/1988	COMP2400	2010 S2	6
Tom	123456	25/01/1988	COMP8740	2011 S2	12
Michael	123458	21/04/1985	COMP2400	2009 S2	6
Michael	123458	21/04/1985	COMP8740	2011 S2	12
Fran	123457	11/09/1987	COMP2400	2009 S2	6

- We may have:
  - {StudentID} → {Name, DoB};
  - {StudentID, Name} → {DoB};
  - {Name} → {StudentID} ×;
  - ......

(2) Analyse Sample Data

- Can you find some FDs on ENROLMENT based on the sample data?

ENROLMENT

Name	StudentID	DoB	CourseNo	Semester	Unit
Tom	123456	25/01/1988	COMP2400	2010 S2	6
Tom	123456	25/01/1988	COMP8740	2011 S2	12
Michael	123458	21/04/1985	COMP2400	2009 S2	6
Michael	123458	21/04/1985	COMP8740	2011 S2	12
Fran	123457	11/09/1987	COMP2400	2009 S2	6

- We may have:
  - {StudentID} $\rightarrow$ {Name, DoB};
  - {StudentID, Name} $\rightarrow$ {DoB};
  - {Name} $\rightarrow$ {StudentID} $\times$;
  - ......

Limitations:
(1) Sample data needs to be a true representation of **all possible values** in the database.
(2) Do we need all FDs?

Inference?

- To design a good database, we need to consider **all possible FDs**.

Example:

If $\{StudentID\} \rightarrow \{ProjectNo\}$ and $\{ProjectNo\} \rightarrow \{Supervisor\}$, we can infer $\{StudentID\} \rightarrow \{Supervisor\}$.

Inference?

- To design a good database, we need to consider **all possible FDs**.

Example:

If $\{StudentID\} \rightarrow \{ProjectNo\}$ and $\{ProjectNo\} \rightarrow \{Supervisor\}$, we can infer $\{StudentID\} \rightarrow \{Supervisor\}$.

If each student works on one project and each project has one supervisor, then each student must have one project supervisor.

Inference?

- To design a good database, we need to consider **all possible FDs**.

  Example:

  If $\{StudentID\} \rightarrow \{ProjectNo\}$ and $\{ProjectNo\} \rightarrow \{Supervisor\}$, we can infer $\{StudentID\} \rightarrow \{Supervisor\}$.

  If each student works on one project and each project has one supervisor, then each student must have one project supervisor.

- **Can we systematically infer all possible FDs?**

Armstrong's Inference Rules (Slides 16-25 will not to be assessed)

- The **Armstrong's inference rules** consist of the following three rules:

  - **Reflexive rule**: $XY \rightarrow Y$
  - **Augmentation rule**: $\{X \rightarrow Y\} \models XZ \rightarrow YZ$

  - **Transitive rule**: $\{X \rightarrow Y, Y \rightarrow Z\} \models X \rightarrow Z$

- We use the notation $\Sigma \models X \rightarrow Y$ to denote that $X \rightarrow Y$ is **inferred** from the set Σ of functional dependencies.

Rule 1 – Reflexive Rule

- $XY \rightarrow Y$.

ENROLMENT					
Name	<u>StudentID</u>	DoB	<u>CourseNo</u>	<u>Semester</u>	Unit
Tom	123456	25/01/1988	COMP2400	2010 S2	6
Tom	123456	25/01/1988	COMP8740	2011 S2	12
Michael	123458	21/04/1985	COMP2400	2009 S2	6
Michael	123458	21/04/1985	COMP8740	2011 S2	12
Fran	123457	11/09/1987	COMP2400	2009 S2	6

- **Example**:
  {StudentID, CourseNo, Semester} → {CourseNo, Semester}, where
    - X={StudentID};
    - Y={CourseNo, Semester}.

Rule 2 – Augmentation Rule

- $\{X \rightarrow Y\} \models XZ \rightarrow YZ$

ENROLMENT					
Name	StudentID	DoB	CourseNo	Semester	Unit
Tom	123456	25/01/1988	COMP2400	2010 S2	6
Tom	123456	25/01/1988	COMP8740	2011 S2	12
Michael	123458	21/04/1985	COMP2400	2009 S2	6
Michael	123458	21/04/1985	COMP8740	2011 S2	12
Fran	123457	11/09/1987	COMP2400	2009 S2	6

- **Example**:
$\{\{\text{CourseNo}\} \rightarrow \{\text{Unit}\}\} \models \{\text{CourseNo, Semester}\} \rightarrow \{\text{Unit, Semester}\}$,
where
  - X={CourseNo};
  - Y={Unit};
  - Z={Semester}.

Rule 3 – Transitive Rule

- $\{X \rightarrow Y, Y \rightarrow Z\} \models X \rightarrow Z$.

ENROLMENT					
Name	StudentID	DoB	CourseNo	Semester	Unit
Tom	123456	25/01/1988	COMP2400	2010 S2	6
Tom	123456	25/01/1988	COMP8740	2011 S2	12
Michael	123458	21/04/1985	COMP2400	2009 S2	6
Michael	123458	21/04/1985	COMP8740	2011 S2	12
Fran	123457	11/09/1987	COMP2400	2009 S2	6

- **Example**: {StudentID, CourseNo} $\rightarrow$ {CourseNo}, {CourseNo} $\rightarrow$ {Unit} $\models$ {StudentID, CourseNo} $\rightarrow$ {Unit}, where
  - X={StudentID, CourseNo};
  - Y={CourseNo};
  - Z={Unit}.

Other Derived Rules

- From Armstrong's axioms (i.e. reflexive, augmentation, transitive rules), we can derive the following rules:

Other Derived Rules

- From Armstrong's axioms (i.e., reflexive, augmentation, transitive rules), we can derive the following rules:
  - **Union rule**: If $X \rightarrow Y$ and $X \rightarrow Z$, then $X \rightarrow YZ$
  - **Example:** If StudentID $\rightarrow$ Name and StudentID $\rightarrow$ DoB hold, then we have StudentID $\rightarrow$ Name, DoB, where

    - X=StudentID;
    - Y=Name;
    - Z=DoB.

Other Derived Rules

- From Armstrong's axioms (i.e., reflexive, augmentation, transitive rules), we can derive the following rules:
  - **Union rule**: If $X \rightarrow Y$ and $X \rightarrow Z$, then $X \rightarrow YZ$
  - **Example:** If StudentID $\rightarrow$ Name and StudentID $\rightarrow$ DoB hold, then we have StudentID $\rightarrow$ Name, DoB, where
    - X=StudentID;
    - Y=Name;
    - Z=DoB.
  - **Decomposition rule**: If $X \rightarrow YZ$, then $X \rightarrow Y$ and $X \rightarrow Z$
  - **Example:** If StudentID $\rightarrow$ Name, DoB holds, then we have StudentID $\rightarrow$ Name and StudentID $\rightarrow$ DoB, where
    - X=StudentID;
    - Y=Name;
    - Z=DoB.

Example on Armstrong's Inference Rules

- If each student works on one project and each project has one supervisor, does each student have one project supervisor?

$$\{\{\text{StudentID}\}\rightarrow\{\text{ProjectNo}\},\ \{\text{ProjectNo}\}\rightarrow\{\text{Supervisor}\}\} \models \{\text{StudentID}\}\rightarrow\{\text{Supervisor}\}$$

Example on Armstrong's Inference Rules

- If each student works on one project and each project has one supervisor, does each student have one project supervisor?

$$\{\{\text{StudentID}\}\rightarrow\{\text{ProjectNo}\}, \{\text{ProjectNo}\}\rightarrow\{\text{Supervisor}\}\} \models \{\text{StudentID}\}\rightarrow\{\text{Supervisor}\}$$

- This can be proven by using the Transitive rule:

$$\{X \rightarrow Y, Y \rightarrow Z\} \models X \rightarrow Z$$

Example on Armstrong's Inference Rules

- Can we use the following rules to infer FDs, i.e., are they correct?

  (1) $\{X \rightarrow Y\} \models XZ \rightarrow YZ$

Example on Armstrong's Inference Rules

- Can we use the following rules to infer FDs, i.e., are they correct?

  (1) $\{X \rightarrow Y\} \models XZ \rightarrow YZ$

  Yes, using the Augmentation rule.

Example on Armstrong's Inference Rules

- Can we use the following rules to infer FDs, i.e., are they correct?

(1) $\{X \rightarrow Y\} \models XZ \rightarrow YZ$

Yes, using the Augmentation rule.

(2) $\{XZ \rightarrow YZ\} \models X \rightarrow Y$

Example on Armstrong's Inference Rules

- Can we use the following rules to infer FDs, i.e., are they correct?

  (1) $\{X \rightarrow Y\} \models XZ \rightarrow YZ$

  Yes, using the Augmentation rule.

  (2) $\{XZ \rightarrow YZ\} \models X \rightarrow Y$

  No. See the counter-example below:

X	Y	Z
a	b	c
a	c	d

Example on Armstrong's Inference Rules

- Can we use the following rules to infer FDs, i.e., are they correct?

(1) $\{X \rightarrow Y\} \models XZ \rightarrow YZ$

Yes, using the Augmentation rule.

(2) $\{XZ \rightarrow YZ\} \models X \rightarrow Y$

No. See the counter-example below:

X	Y	Z
a	b	c
a	c	d

(3) $\{X \rightarrow Y\} \models Y \rightarrow X$

Example on Armstrong's Inference Rules

- Can we use the following rules to infer FDs, i.e., are they correct?

(1) $\{X \rightarrow Y\} \models XZ \rightarrow YZ$

Yes, using the Augmentation rule.

(2) $\{XZ \rightarrow YZ\} \models X \rightarrow Y$

No. See the counter-example below:

X	Y	Z
a	b	c
a	c	d

(3) $\{X \rightarrow Y\} \models Y \rightarrow X$

No. See the counter-example below:

X	Y
0	2
1	2

Armstrong's Inference Rules

- **Two questions:**

[1] William Ward Armstrong: Dependency Structures of Data Base Relationships, page 580-583. IFIP Congress, 1974.

Australian National University

Armstrong's Inference Rules

- **Two questions:**
  - Are all the FDs inferred using the Armstrong's inference rules correct?
    $\rightsquigarrow$ **soundness** (you cannot prove anything that is wrong)

[1] William Ward Armstrong: Dependency Structures of Data Base Relationships, page 580-583. IFIP Congress, 1974.

Armstrong's Inference Rules

- **Two questions:**
  - Are all the FDs inferred using the Armstrong's inference rules correct?
    $\rightsquigarrow$ **soundness** (you cannot prove anything that is wrong)
  - Can we use the Armstrong's inference rules to infer all possible FDs?
    $\rightsquigarrow$ **completeness** (you can prove anything that is right)

[1] William Ward Armstrong: Dependency Structures of Data Base Relationships, page 580-583. IFIP Congress, 1974.

Armstrong's Inference Rules

- **Two questions:**
  - Are all the FDs inferred using the Armstrong's inference rules correct?
    ⇝ **soundness** (you cannot prove anything that is wrong)
  - Can we use the Armstrong's inference rules to infer all possible FDs?
    ⇝ **completeness** (you can prove anything that is right)

- **Theorem** (W. W. Armstrong, 1974[1])
  - The Armstrong's inference rules are both **sound** and **complete**.

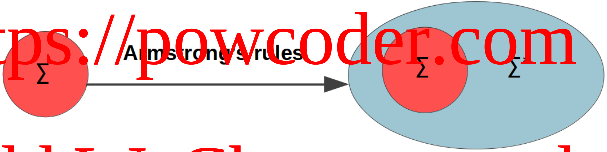
[1] William Ward Armstrong: Dependency Structures of Data Base Relationships, page 580-583. IFIP Congress, 1974.

Implied Functional Dependencies

- We write Σ^* for all possible FDs **implied** by Σ.

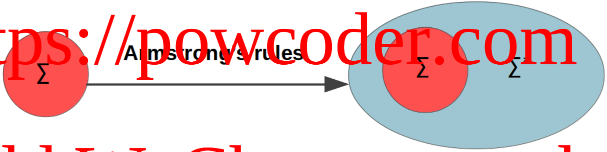
Implied Functional Dependencies

- We write Σ^* for all possible FDs **implied** by Σ.
- Σ^* can be computed using the Armstrong's inference rules.



Implied Functional Dependencies

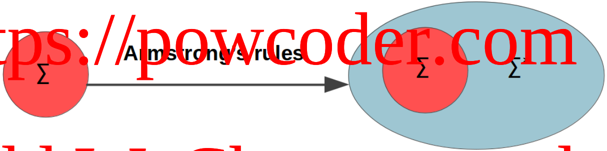
- We write Σ^* for all possible FDs **implied** by Σ.
- Σ^* can be computed using the Armstrong's inference rules.



- **Why can we compute Σ^* using the Armstrong's inference rules?**

Implied Functional Dependencies

- We write Σ^* for all possible FDs **implied** by Σ.
- Σ^* can be computed using the Armstrong's inference rules.

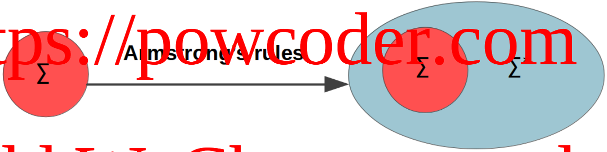


- **Why can we compute Σ^* using the Armstrong's inference rules?**

  Because the Armstrong's inference rules are both **sound** and **complete**.

Implied Functional Dependencies

- We write Σ^* for all possible FDs **implied** by Σ.
- Σ^* can be computed using the Armstrong's inference rules.



- **Why can we compute Σ^* using the Armstrong's inference rules?**

  Because the Armstrong's inference rules are both **sound** and **complete**.
- Nonetheless, computing Σ^* using the Armstrong's inference rules is **not efficient**.

Implied Functional Dependencies

- Computing Σ^* using the Armstrong's inference rules is **not efficient**.

  Example: Consider a relation schema $R = \{A, B, C, D, E\}$ and a set of FDs $\Sigma = \{AB \rightarrow CD, B \rightarrow E, DE \rightarrow A\}$. How can we use the Armstrong rules to show that $DB \rightarrow A \in \Sigma^*$?

Implied Functional Dependencies

- Computing Σ^* using the Armstrong's inference rules is **not efficient**.

  Example: Consider a relation schema $R = \{A, B, C, D, E\}$ and a set of FDs $\Sigma = \{AB \to CD, B \to E, DE \to A\}$. How can we use the Armstrong rules to show that $DB \to A \in \Sigma^*$?

$$\dfrac{\dfrac{\dfrac{\dfrac{DB \to B \qquad \boxed{B \to E}}{DB \to E}}{DB \to BDE} \qquad BDE \to DE}{DB \to DE} \qquad \boxed{DE \to A}}{DB \to A}$$

Implied Functional Dependencies

- Computing Σ^* using the Armstrong's inference rules is **not efficient**.

  Example: Consider a relation schema $R = \{A, B, C, D, E\}$ and a set of FDs $\Sigma = \{AB \rightarrow CD, B \rightarrow E, DE \rightarrow A\}$. How can we use the Armstrong rules to show that $DB \rightarrow A \in \Sigma^*$?

$$\dfrac{\dfrac{\dfrac{\dfrac{DB \rightarrow B \qquad \boxed{B \rightarrow E}}{DB \rightarrow E}}{DB \rightarrow BDE} \qquad BDE \rightarrow DE}{DB \rightarrow DE} \qquad \boxed{DE \rightarrow A}}{DB \rightarrow A}$$

- **How can we derive the proof more efficiently?**

Implied Functional Dependencies

- Let Σ be a set of FDs. Check whether or not $\Sigma \models X \rightarrow W$ holds?

[2] See Algorithm 15.1 on Page 538 in [Elmasri & Navathe, 7th edition] or Algorithm 1 on Page 555 in [Elmasri & Navathe, 6th edition]

Australian National University

Implied Functional Dependencies

- Let Σ be a set of FDs. Check whether or not $\Sigma \models X \rightarrow W$ holds? We need to

[2] See Algorithm 15.1 on Page 538 in [Elmasri & Navathe, 7th edition] or Algorithm 1 on Page 555 in [Elmasri & Navathe, 6th edition]

Australian National University

Implied Functional Dependencies

- Let Σ be a set of FDs. Check whether or not $\Sigma \models X \rightarrow W$ holds? We need to

  1. Compute **the set of all attributes** that are dependent on X, which is called the **closure** of X under Σ and is denoted by X^+.

[2] See Algorithm 15.1 on Page 538 in [Elmasri & Navathe, 7th edition] or Algorithm 1 on Page 555 in [Elmasri & Navathe, 6th edition]

Implied Functional Dependencies

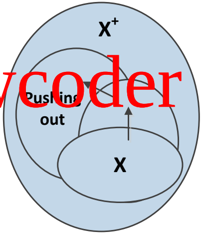
- Let Σ be a set of FDs. Check whether or not $\Sigma \models X \rightarrow W$ holds? We need to
  1. Compute **the set of all attributes** that are dependent on X, which is called the **closure** of X under Σ and is denoted by X^+.
  2. $\Sigma \models X \rightarrow W$ holds iff $W \subseteq X^+$.

[2] See Algorithm 15.1 on Page 538 in [Elmasri & Navathe, 7th edition] or Algorithm 1 on Page 555 in [Elmasri & Navathe, 6th edition]

Implied Functional Dependencies

- Let Σ be a set of FDs. Check whether or not $\Sigma \models X \to W$ holds? We need to
  1. Compute **the set of all attributes** that are dependent on X, which is called the **closure** of X under Σ and is denoted by X^+.
  2. $\Sigma \models X \to W$ holds iff $W \subseteq X^+$.

- **Algorithm**[2]
  - $X^+ := X$;
  - repeat until no more change on X^+
    - for each $Y \to Z \in \Sigma$ with $Y \subseteq X^+$, add all the attributes in Z to X^+, i.e., replace X^+ by $X^+ \cup Z$.



[2] See Algorithm 15.1 on Page 538 in [Elmasri & Navathe, 7th edition] or Algorithm 1 on Page 555 in [Elmasri & Navathe, 6th edition]

Implied Functional Dependencies – Example

- Consider a relation schema $R = \{A, B, C, D, E, F\}$, a set of FDs $\Sigma = \{AC \rightarrow B, B \rightarrow CD, C \rightarrow E, AF \rightarrow B\}$ on R.
- Decide whether or not $\Sigma \models AC \rightarrow DE$ holds .

Implied Functional Dependencies – Example

- Consider a relation schema $R = \{A, B, C, D, E, F\}$, a set of FDs $\Sigma = \{AC \rightarrow B, B \rightarrow CD, C \rightarrow E, AF \rightarrow B\}$ on R.
- Decide whether or not $\Sigma \models AC \rightarrow DE$ holds.

1. We first build the closure of AC:

   $(AC)^+ \quad \supseteq AC$ initialisation

Implied Functional Dependencies – Example

- Consider a relation scheme $R = \{A, B, C, D, E, F\}$, a set of FDs $\Sigma = \{AC \rightarrow B, B \rightarrow CD, C \rightarrow E, AF \rightarrow B\}$ on R.
- Decide whether or not $\Sigma \models AC \rightarrow DE$ holds.

1. We first build the closure of AC:

$(AC)^+$	$\supseteq AC$	initialisation
	$\supseteq ACB$	using $AC \rightarrow B$

Implied Functional Dependencies – Example

- Consider a relation scheme $R = \{A, B, C, D, E, F\}$, a set of FDs $\Sigma = \{AC \to B, B \to CD, C \to E, AF \to B\}$ on R.
- Decide whether or not $\Sigma \models AC \to DE$ holds.

1. We first build the closure of AC:

$(AC)^+$	$\supseteq$	AC	initialisation
	$\supseteq$	ACB	using $AC \to B$
	$\supseteq$	$ACBD$	using $B \to CD$

Implied Functional Dependencies – Example

- Consider a relation schema $R = \{A, B, C, D, E, F\}$, a set of FDs $\Sigma = \{AC \rightarrow B, B \rightarrow CD, C \rightarrow E, AF \rightarrow B\}$ on R.
- Decide whether or not $\Sigma \models AC \rightarrow DE$ holds.

1. We first build the closure of AC:

$$
\begin{array}{lll}
(AC)^+ & \supseteq AC & \text{initialisation} \\
 & \supseteq ACB & \text{using } AC \rightarrow B \\
 & \supseteq ACBD & \text{using } B \rightarrow CD \\
 & \supseteq ACBDE & \text{using } C \rightarrow E
\end{array}
$$

Implied Functional Dependencies – Example

- Consider a relation schema $R = \{A, B, C, D, E, F\}$, a set of FDs $\Sigma = \{AC \rightarrow B, B \rightarrow CD, C \rightarrow E, AF \rightarrow B\}$ on R.
- Decide whether or not $\Sigma \models AC \rightarrow DE$ holds.

1. We first build the closure of AC:

$$
\begin{array}{lll}
(AC)^+ & \supseteq AC & \text{initialisation} \\
 & \supseteq ACB & \text{using } AC \rightarrow B \\
 & \supseteq ACBD & \text{using } B \rightarrow CD \\
 & \supseteq ACBDE & \text{using } C \rightarrow E \\
 & = ACBDE &
\end{array}
$$

Implied Functional Dependencies – Example

- Consider a relation schema $R = \{A, B, C, D, E, F\}$, a set of FDs $\Sigma = \{AC \to B, B \to CD, C \to E, AF \to B\}$ on R.
- Decide whether or not $\Sigma \models AC \to DE$ holds.

1. We first build the closure of AC:

$$\begin{array}{lll} (AC)^+ & \supseteq AC & \text{initialisation} \\ & \supseteq ACB & \text{using } AC \to B \\ & \supseteq ACBD & \text{using } B \to CD \\ & \supseteq ACBDE & \text{using } C \to E \\ & = ACBDE & \end{array}$$

2. Then we check that $DE \subseteq (AC)^+$. Hence $\Sigma \models AC \to DE$.

Implied Functional Dependencies – Example

- Consider a relation schema $R = \{A, B, C, D, E, F\}$, a set of FDs $\Sigma = \{AC \rightarrow B, B \rightarrow CD, C \rightarrow E, AF \rightarrow B\}$ on R.
- Decide whether or not $\Sigma \models AC \rightarrow DE$ holds.

1. We first build the closure of AC:

$$
\begin{array}{lll}
(AC)^+ & \supseteq AC & \text{initialisation} \\
 & \supseteq ACB & \text{using } AC \rightarrow B \\
 & \supseteq ACBD & \text{using } B \rightarrow CD \\
 & \supseteq ACBDE & \text{using } C \rightarrow E \\
 & = ACBDE &
\end{array}
$$

2. Then we check that $DE \subseteq (AC)^+$. Hence $\Sigma \models AC \rightarrow DE$.

- **Can you quickly tell whether or not $\Sigma \models AC \rightarrow EF$ holds?**

Implied Functional Dependencies – Example

- Consider a relation schema $R = \{A, B, C, D, E, F\}$, a set of FDs $\Sigma = \{AC \rightarrow B, B \rightarrow CD, C \rightarrow E, AF \rightarrow B\}$ on R.

- Decide whether or not $\Sigma \models AC \rightarrow DE$ holds.

  1. We first build the closure of AC:

$$\begin{aligned} (AC)^+ &\supseteq AC && \text{initialisation} \\ &\supseteq ACB && \text{using } AC \rightarrow B \\ &\supseteq ACBD && \text{using } B \rightarrow CD \\ &\supseteq ACBDE && \text{using } C \rightarrow E \\ &= ACBDE \end{aligned}$$

  2. Then we check that $DE \subseteq (AC)^+$. Hence $\Sigma \models AC \rightarrow DE$.

- **Can you quickly tell whether or not $\Sigma \models AC \rightarrow EF$ holds?**

  $\Sigma \models AC \rightarrow EF$ does not hold because $EF \not\subseteq (AC)^+$

Exercise – Implied Functional Dependencies

- Consider a relation schema $R = \{A, B, C, D, E\}$ and a set of functional dependencies $\Sigma = \{A \rightarrow C, B \rightarrow C, CD \rightarrow E\}$ on R.
- Decide whether or not

  1. $\Sigma \models AD \rightarrow CE$ holds
  2. $\Sigma \models BD \rightarrow AC$ holds

Exercise – Implied Functional Dependencies

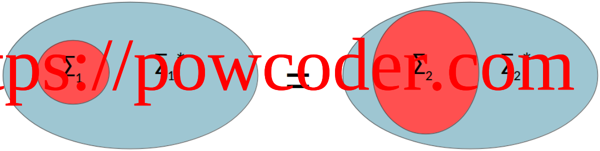
- Consider a relation schema $R = \{A, B, C, D, E\}$ and a set of functional dependencies $\Sigma = \{A \rightarrow C, B \rightarrow C, CD \rightarrow E\}$ on R.
- Decide whether or not

  1. $\Sigma \models AD \rightarrow CE$ holds
  2. $\Sigma \models BD \rightarrow AC$ holds

- We build the closure for the set of attributes and check:
  1. $(AD)^+ = (ACD)^+ = (ACDE)^+ = ACDE$ and $CE \subseteq (AD)^+$, hence $\Sigma \models AD \rightarrow CE$.
  2. $(BD)^+ = (BCD)^+ = (BCDE)^+ = BCDE$ and $AC \not\subseteq (BD)^+$, hence $\Sigma \not\models BD \rightarrow AC$.

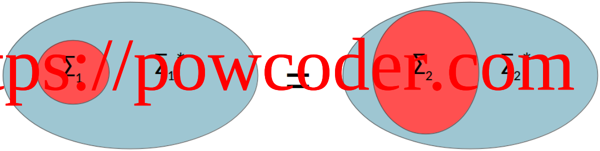
Equivalence of Functional Dependencies

- Σ_1 and Σ_2 are **equivalent** if $\Sigma_1^* = \Sigma_2^*$.



Equivalence of Functional Dependencies

- Σ_1 and Σ_2 are **equivalent** if $\Sigma_1^* = \Sigma_2^*$.

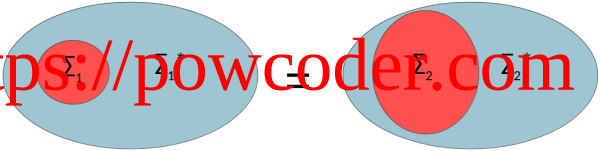


- Let $\Sigma_1 = \{X \rightarrow Y, Y \rightarrow Z\}$ and $\Sigma_2 = \{X \rightarrow Y, Y \rightarrow Z, X \rightarrow Z\}$. Note $\Sigma_1 \neq \Sigma_2$ but $\Sigma_1^* = \Sigma_2^* = \{X \rightarrow Y, Y \rightarrow Z, X \rightarrow Z\}$ (Σ_1 and Σ_2 are equivalent)

Australian National University

Equivalence of Functional Dependencies

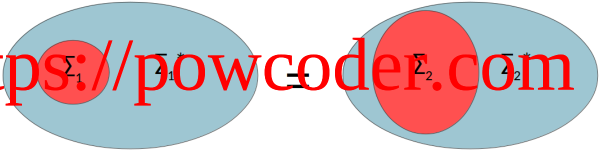
- Σ_1 and Σ_2 are **equivalent** if $\Sigma_1^* = \Sigma_2^*$.



- Let $\Sigma_1 = \{X \rightarrow Y, Y \rightarrow Z\}$ and $\Sigma_2 = \{X \rightarrow Y, Y \rightarrow Z, X \rightarrow Z\}$. Note $\Sigma_1 \neq \Sigma_2$ but $\Sigma_1^* = \Sigma_2^* = \{X \rightarrow Y, Y \rightarrow Z, X \rightarrow Z\}$ (Σ_1 and Σ_2 are equivalent)
- If $\Sigma_1 \models \Sigma_2$ and $\Sigma_2 \models \Sigma_1$, are Σ_1 and Σ_2 equivalent?

Equivalence of Functional Dependencies

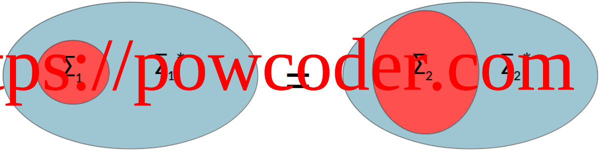
- Σ_1 and Σ_2 are **equivalent** if $\Sigma_1^* = \Sigma_2^*$.



- Let $\Sigma_1 = \{X \rightarrow Y, Y \rightarrow Z\}$ and $\Sigma_2 = \{X \rightarrow Y, Y \rightarrow Z, X \rightarrow Z\}$. Note $\Sigma_1 \neq \Sigma_2$ but $\Sigma_1^* = \Sigma_2^* = \{X \rightarrow Y, Y \rightarrow Z, X \rightarrow Z\}$ (Σ_1 and Σ_2 are equivalent)
- If $\Sigma_1 \models \Sigma_2$ and $\Sigma_2 \models \Sigma_1$, are Σ_1 and Σ_2 equivalent? Yes.

Equivalence of Functional Dependencies

- Σ_1 and Σ_2 are **equivalent** if $\Sigma_1^* = \Sigma_2^*$.



- Let $\Sigma_1 = \{X \to Y, Y \to Z\}$ and $\Sigma_2 = \{X \to Y, Y \to Z, X \to Z\}$. Note $\Sigma_1 \neq \Sigma_2$ but $\Sigma_1^* = \Sigma_2^* = \{X \to Y, Y \to Z, X \to Z\}$ (Σ_1 and Σ_2 are equivalent)
- If $\Sigma_1 \models \Sigma_2$ and $\Sigma_2 \models \Sigma_1$, are Σ_1 and Σ_2 equivalent? Yes.

- **Questions:** Can we find the **minimal** one among equivalent sets of FDs?

Minimal Cover – The Hard Part!

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder

Minimal Cover – The Hard Part!

- Let Σ be a set of FDs. A **minimal cover** Σ_m of Σ is a set of FDs such that

   1. Σ_m is equivalent to Σ, i.e., start with $\Sigma_m = \Sigma$;

Australian National University

Minimal Cover – The Hard Part!

- Let Σ be a set of FDs. A **minimal cover** Σ_m of Σ is a set of FDs such that

  1. Σ_m is equivalent to Σ, i.e., start with $\Sigma_m = \Sigma$;

  2. **Dependent:** each FD in Σ_m has only a single attribute on its right hand side, i.e., replace each FD $X \rightarrow \{A_1, \ldots, A_k\}$ in Σ_m with $X \rightarrow A_1, \ldots, X \rightarrow A_k$;

Minimal Cover – The Hard Part!

- Let Σ be a set of FDs. A **minimal cover** Σ_m of Σ is a set of FDs such that

1. Σ_m is equivalent to Σ, i.e., start with $\Sigma_m = \Sigma$;
2. **Dependent:** each FD in Σ_m has only a single attribute on its right hand side, i.e., replace each FD $X \rightarrow \{A_1, \ldots, A_k\}$ in Σ_m with $X \rightarrow A_1, \ldots, X \rightarrow A_k$;
3. **Determinant:** each FD has as few attributes on the left hand side as possible, i.e., for each FD $X \rightarrow A$ in Σ_m, check each attribute B of X to see if we can replace $X \rightarrow A$ with $(X - B) \rightarrow A$ in Σ_m;

Minimal Cover – The Hard Part!

- Let Σ be a set of FDs. A **minimal cover** Σ_m of Σ is a set of FDs such that

1. Σ_m is equivalent to Σ, i.e., start with $\Sigma_m = \Sigma$;

2. **Dependent:** each FD in Σ_m has only a single attribute on its right hand side, i.e., replace each FD $X \rightarrow \{A_1, \ldots, A_k\}$ in Σ_m with $X \rightarrow A_1, \ldots, X \rightarrow A_k$;

3. **Determinant:** each FD has as few attributes on the left hand side as possible, i.e., for each FD $X \rightarrow A$ in Σ_m, check each attribute B of X to see if we can replace $X \rightarrow A$ with $(X - B) \rightarrow A$ in Σ_m;

4. Remove a FD from Σ_m if it is redundant.

Minimal Cover - Examples

- Given the set of FDs $\Sigma = \{B \rightarrow A, D \rightarrow A, AB \rightarrow D\}$, we can compute the minimal cover of Σ as follows:

Minimal Cover - Examples

- Given the set of FDs $\Sigma = \{B \rightarrow A, D \rightarrow A, AB \rightarrow D\}$, we can compute the minimal cover of Σ as follows:
  1. start from Σ;

Minimal Cover - Examples

- Given the set of FDs $\Sigma = \{B \rightarrow A, D \rightarrow A, AB \rightarrow D\}$, we can compute the minimal cover of Σ as follows:
  1. start from Σ;
  2. check whether all the FDs in Σ have only one attribute on the right hand side (look good);

Minimal Cover - Examples

- Given the set of FDs $\Sigma = \{B \rightarrow A, D \rightarrow A, AB \rightarrow D\}$, we can compute the minimal cover of Σ as follows:
  1. start from Σ;
  2. check whether all the FDs in Σ have only one attribute on the right hand side (look good);
  3. check if $AB \rightarrow D$ can be replaced by $A \rightarrow D$?

Minimal Cover - Examples

- Given the set of FDs $\Sigma = \{B \rightarrow A, D \rightarrow A, AB \rightarrow D\}$, we can compute the minimal cover of Σ as follows:
  1. start from Σ;
  2. check whether all the FDs in Σ have only one attribute on the right hand side (look good);
  3. check if $AB \rightarrow D$ can be replaced by $A \rightarrow D$?
     - $\Sigma = \{B \rightarrow A, D \rightarrow A, \mathbf{AB} \rightarrow \mathbf{D}\}$, $\Sigma_1 = \{B \rightarrow A, D \rightarrow A, \mathbf{A} \rightarrow \mathbf{D}\}$

Minimal Cover - Examples

- Given the set of FDs $\Sigma = \{B \rightarrow A, D \rightarrow A, AB \rightarrow D\}$, we can compute the minimal cover of Σ as follows:
  1. start from Σ;
  2. check whether all the FDs in Σ have only one attribute on the right hand side (look good);
  3. check if $AB \rightarrow D$ can be replaced by $A \rightarrow D$?
     - $\Sigma = \{B \rightarrow A, D \rightarrow A, \mathbf{AB \rightarrow D}\}$, $\Sigma_1 = \{B \rightarrow A, D \rightarrow A, \mathbf{A \rightarrow D}\}$
     - check whether $\Sigma^* = \Sigma_1^*$?

Minimal Cover - Examples

- Given the set of FDs $\Sigma = \{B \to A, D \to A, AB \to D\}$, we can compute the minimal cover of Σ as follows:
  1. start from Σ;
  2. check whether all the FDs in Σ have only one attribute on the right hand side (look good);
  3. check if $AB \to D$ can be replaced by $A \to D$?
     - $\Sigma = \{B \to A, D \to A, \mathbf{AB \to D}\}$, $\Sigma_1 = \{B \to A, D \to A, \mathbf{A \to D}\}$
     - check whether $\Sigma^* = \Sigma_1^*$? (we have $\Sigma_1 \models \Sigma$, but $\Sigma \models \Sigma_1$?)

Minimal Cover - Examples

- Given the set of FDs $\Sigma = \{B \rightarrow A, D \rightarrow A, AB \rightarrow D\}$, we can compute the minimal cover of Σ as follows:
  1. start from Σ;
  2. check whether all the FDs in Σ have only one attribute on the right hand side (look good);
  3. check if $AB \rightarrow D$ can be replaced by $A \rightarrow D$?
     - $\Sigma = \{B \rightarrow A, D \rightarrow A, \mathbf{AB \rightarrow D}\}$, $\Sigma_1 = \{B \rightarrow A, D \rightarrow A, \mathbf{A \rightarrow D}\}$
     - check whether $\Sigma^* = \Sigma_1^*$? (we have $\Sigma_1 \models \Sigma$, but $\Sigma \models \Sigma_1$?)
     - check $\Sigma \models \mathbf{A \rightarrow D}$?

Minimal Cover - Examples

- Given the set of FDs $\Sigma = \{B \rightarrow A, D \rightarrow A, AB \rightarrow D\}$, we can compute the minimal cover of Σ as follows:
  1. start from Σ;
  2. check whether all the FDs in Σ have only one attribute on the right hand side (look good);
  3. check if $AB \rightarrow D$ can be replaced by $A \rightarrow D$?
     - $\Sigma = \{B \rightarrow A, D \rightarrow A, \mathbf{AB} \rightarrow \mathbf{D}\}$, $\Sigma_1 = \{B \rightarrow A, D \rightarrow A, \mathbf{A} \rightarrow \mathbf{D}\}$
     - check whether $\Sigma^* = \Sigma_1^*$? (we have $\Sigma_1 \models \Sigma$, but $\Sigma \models \Sigma_1$?)
     - check $\Sigma \models \mathbf{A} \rightarrow \mathbf{D}$?

       If $\Sigma \models \mathbf{A} \rightarrow \mathbf{D}$, then $\Sigma \models \Sigma_1$ and $\Sigma_1 \models \Sigma$, indicating $\Sigma^* = \Sigma_1^*$.

Minimal Cover - Examples

- Given the set of FDs $\Sigma = \{B \rightarrow A, D \rightarrow A, AB \rightarrow D\}$, we can compute the minimal cover of Σ as follows:
  1. start from Σ;
  2. check whether all the FDs in Σ have only one attribute on the right hand side (look good);
  3. check if $AB \rightarrow D$ can be replaced by $A \rightarrow D$?
     - $\Sigma = \{B \rightarrow A, D \rightarrow A, \mathbf{AB \rightarrow D}\}$, $\Sigma_1 = \{B \rightarrow A, D \rightarrow A, \mathbf{A \rightarrow D}\}$
     - check whether $\Sigma^* = \Sigma_1^*$? (we have $\Sigma_1 \models \Sigma$, but $\Sigma \models \Sigma_1$?)
     - check $\Sigma \models \mathbf{A \rightarrow D}$?

       If $\Sigma \models \mathbf{A \rightarrow D}$, then $\Sigma \models \Sigma_1$ and $\Sigma_1 \models \Sigma$, indicating $\Sigma^* = \Sigma_1^*$.

       If $\Sigma \not\models \mathbf{A \rightarrow D}$, then $\Sigma^* \neq \Sigma_1^*$.

Minimal Cover - Examples

- Given the set of FDs $\Sigma = \{B \to A, D \to A, AB \to D\}$, we can compute the minimal cover of Σ as follows:
  1. start from Σ;
  2. check whether all the FDs in Σ have only one attribute on the right hand side (look good);
  3. check if $AB \to D$ can be replaced by $A \to D$?
     - $\Sigma = \{B \to A, D \to A, \mathbf{AB \to D}\}$, $\Sigma_1 = \{B \to A, D \to A, \mathbf{A \to D}\}$
     - check whether $\Sigma^* = \Sigma_1^*$? (we have $\Sigma_1 \models \Sigma$, but $\Sigma \models \Sigma_1$?)
     - check $\Sigma \models \mathbf{A \to D}$?

       If $\Sigma \models \mathbf{A \to D}$, then $\Sigma \models \Sigma_1$ and $\Sigma_1 \models \Sigma$, indicating $\Sigma^* = \Sigma_1^*$.

       If $\Sigma \not\models \mathbf{A \to D}$, then $\Sigma^* \neq \Sigma_1^*$.
     - $\Sigma \not\models \mathbf{A \to D}$ because $D \not\subseteq (A)^+$.

     No. $AB \to D$ cannot be replaced by $A \to D$.

Minimal Cover - Examples

- Given the set of FDs $\Sigma = \{B \rightarrow A, D \rightarrow A, AB \rightarrow D\}$, we can compute the minimal cover of Σ as follows:
  1. start from Σ;
  2. check whether all the FDs in Σ have only one attribute on the right hand side (look good);
  3. check if $AB \rightarrow D$ can be replaced by $B \rightarrow D$?

Minimal Cover - Examples

- Given the set of FDs $\Sigma = \{B \rightarrow A, D \rightarrow A, AB \rightarrow D\}$, we can compute the minimal cover of Σ as follows:
  1. start from Σ;
  2. check whether all the FDs in Σ have only one attribute on the right hand side (look good);
  3. check if $AB \rightarrow D$ can be replaced by $B \rightarrow D$?
     - $\Sigma = \{B \rightarrow A, D \rightarrow A, \mathbf{AB \rightarrow D}\}$, $\Sigma_2 = \{B \rightarrow A, D \rightarrow A, \mathbf{B \rightarrow D}\}$

Minimal Cover - Examples

- Given the set of FDs $\Sigma = \{B \rightarrow A, D \rightarrow A, AB \rightarrow D\}$, we can compute the minimal cover of Σ as follows:

  1. start from Σ;
  2. check whether all the FDs in Σ have only one attribute on the right hand side (look good);
  3. check if $AB \rightarrow D$ can be replaced by $B \rightarrow D$?
     - $\Sigma = \{B \rightarrow A, D \rightarrow A, \mathbf{AB \rightarrow D}\}$, $\Sigma_2 = \{B \rightarrow A, D \rightarrow A, \mathbf{B \rightarrow D}\}$
     - check whether $\Sigma^* = \Sigma_2^*$?

Minimal Cover - Examples

- Given the set of FDs $\Sigma = \{B \rightarrow A, D \rightarrow A, AB \rightarrow D\}$, we can compute the minimal cover of Σ as follows:
  1. start from Σ;
  2. check whether all the FDs in Σ have only one attribute on the right hand side (look good);
  3. check if $AB \rightarrow D$ can be replaced by $B \rightarrow D$?
     - $\Sigma = \{B \rightarrow A, D \rightarrow A, \mathbf{AB \rightarrow D}\}$, $\Sigma_2 = \{B \rightarrow A, D \rightarrow A, \mathbf{B \rightarrow D}\}$
     - check whether $\Sigma^* = \Sigma_2^*$?

Minimal Cover - Examples

- Given the set of FDs $\Sigma = \{B \rightarrow A, D \rightarrow A, AB \rightarrow D\}$, we can compute the minimal cover of Σ as follows:
  1. start from Σ;
  2. check whether all the FDs in Σ have only one attribute on the right hand side (look good);
  3. check if $AB \rightarrow D$ can be replaced by $B \rightarrow D$?
     - $\Sigma = \{B \rightarrow A, D \rightarrow A, \mathbf{AB \rightarrow D}\}$, $\Sigma_2 = \{B \rightarrow A, D \rightarrow A, \mathbf{B \rightarrow D}\}$
     - check whether $\Sigma^* = \Sigma_2^*$? (we have $\Sigma_2 \models \Sigma$, but $\Sigma \models \Sigma_2$?)

Australian National University

Minimal Cover - Examples

- Given the set of FDs $\Sigma = \{B \rightarrow A, D \rightarrow A, AB \rightarrow D\}$, we can compute the minimal cover of Σ as follows:

  1. start from Σ;
  2. check whether all the FDs in Σ have only one attribute on the right hand side (look good);
  3. check if $AB \rightarrow D$ can be replaced by $B \rightarrow D$?
     - $\Sigma = \{B \rightarrow A, D \rightarrow A, \mathbf{AB \rightarrow D}\}$, $\Sigma_2 = \{B \rightarrow A, D \rightarrow A, \mathbf{B \rightarrow D}\}$
     - check whether $\Sigma^* = \Sigma_2^*$? (we have $\Sigma_2 \models \Sigma$, but $\Sigma \models \Sigma_2$?)
     - check $\Sigma \models \mathbf{B \rightarrow D}$?

Minimal Cover - Examples

- Given the set of FDs $\Sigma = \{B \rightarrow A, D \rightarrow A, AB \rightarrow D\}$, we can compute the minimal cover of Σ as follows:
  1. start from Σ;
  2. check whether all the FDs in Σ have only one attribute on the right hand side (look good);
  3. check if $AB \rightarrow D$ can be replaced by $B \rightarrow D$?
     - $\Sigma = \{B \rightarrow A, D \rightarrow A, \mathbf{AB \rightarrow D}\}$, $\Sigma_2 = \{B \rightarrow A, D \rightarrow A, \mathbf{B \rightarrow D}\}$
     - check whether $\Sigma^* = \Sigma_2^*$? (we have $\Sigma_2 \models \Sigma$, but $\Sigma \models \Sigma_2$?)
     - check $\Sigma \models \mathbf{B \rightarrow D}$?

       If $\Sigma \models \mathbf{B \rightarrow D}$, then $\Sigma \models \Sigma_2$ and $\Sigma_2 \models \Sigma$, indicating $\Sigma^* = \Sigma_2^*$.

Minimal Cover - Examples

- Given the set of FDs $\Sigma = \{B \rightarrow A, D \rightarrow A, AB \rightarrow D\}$, we can compute the minimal cover of Σ as follows:
  1. start from Σ;
  2. check whether all the FDs in Σ have only one attribute on the right hand side (look good);
  3. check if $AB \rightarrow D$ can be replaced by $B \rightarrow D$?
     - $\Sigma = \{B \rightarrow A, D \rightarrow A, \mathbf{AB} \rightarrow \mathbf{D}\}$, $\Sigma_2 = \{B \rightarrow A, D \rightarrow A, \mathbf{B} \rightarrow \mathbf{D}\}$
     - check whether $\Sigma^* = \Sigma_2^*$? (we have $\Sigma_2 \models \Sigma$, but $\Sigma \models \Sigma_2$?)
     - check $\Sigma \models \mathbf{B} \rightarrow \mathbf{D}$?

       If $\Sigma \models \mathbf{B} \rightarrow \mathbf{D}$, then $\Sigma \models \Sigma_2$ and $\Sigma_2 \models \Sigma$, indicating $\Sigma^* = \Sigma_2^*$.

       If $\Sigma \not\models \mathbf{B} \rightarrow \mathbf{D}$, then $\Sigma^* \neq \Sigma_2^*$

Australian National University

Minimal Cover - Examples

- Given the set of FDs $\Sigma = \{B \rightarrow A, D \rightarrow A, AB \rightarrow D\}$, we can compute the minimal cover of Σ as follows:

1. start from Σ;

2. check whether all the FDs in Σ have only one attribute on the right hand side (look good);
3. check if $AB \rightarrow D$ can be replaced by $B \rightarrow D$?
    - $\Sigma = \{B \rightarrow A, D \rightarrow A, \mathbf{AB \rightarrow D}\}$, $\Sigma_2 = \{B \rightarrow A, D \rightarrow A, \mathbf{B \rightarrow D}\}$
    - check whether $\Sigma^* = \Sigma_2^*$? (we have $\Sigma_2 \models \Sigma$, but $\Sigma \models \Sigma_2$?)
    - check $\Sigma \models \mathbf{B \rightarrow D}$?

      If $\Sigma \models \mathbf{B \rightarrow D}$, then $\Sigma \models \Sigma_2$ and $\Sigma_2 \models \Sigma$, indicating $\Sigma^* = \Sigma_2^*$.

      If $\Sigma \not\models \mathbf{B \rightarrow D}$, then $\Sigma^* \neq \Sigma_2^*$
    - $\Sigma \models \mathbf{B \rightarrow D}$ because $D \subseteq (B)^+$.

   Yes. $AB \rightarrow D$ can be replaced by $B \rightarrow D$.

Minimal Cover - Examples

- Given the set of FDs $\Sigma = \{B \rightarrow A, D \rightarrow A, AB \rightarrow D\}$, we can compute the minimal cover of Σ as follows:
  1. start from Σ;
  2. check whether all the FDs in Σ have only one attribute on the right hand side (look good);
  3. $AB \rightarrow D$ can be replaced by $B \rightarrow D$;

Minimal Cover - Examples

- Given the set of FDs $\Sigma = \{B \rightarrow A, D \rightarrow A, AB \rightarrow D\}$, we can compute the minimal cover of Σ as follows:
  1. start from Σ;

  2. check whether all the FDs in Σ have only one attribute on the right hand side (look good);
  3. $AB \rightarrow D$ can be replaced by $B \rightarrow D$;

  4. look for a redundant FD in $\{B \rightarrow A, D \rightarrow A, B \rightarrow D\}$

Minimal Cover - Examples

- Given the set of FDs $\Sigma = \{B \rightarrow A, D \rightarrow A, AB \rightarrow D\}$, we can compute the minimal cover of Σ as follows:

  1. start from Σ;
  2. check whether all the FDs in Σ have only one attribute on the right hand side (look good);
  3. $AB \rightarrow D$ can be replaced by $B \rightarrow D$;
  4. look for a redundant FD in $\{B \rightarrow A, D \rightarrow A, B \rightarrow D\}$
     - check whether $B \rightarrow A$ is redundant?

Australian National University

Minimal Cover - Examples

- Given the set of FDs $\Sigma = \{B \rightarrow A, D \rightarrow A, AB \rightarrow D\}$, we can compute the minimal cover of Σ as follows:
  1. start from Σ;
  2. check whether all the FDs in Σ have only one attribute on the right hand side (look good);
  3. $AB \rightarrow D$ can be replaced by $B \rightarrow D$;
  4. look for a redundant FD in $\{B \rightarrow A, D \rightarrow A, B \rightarrow D\}$
     - check whether $B \rightarrow A$ is redundant?
     - $B \rightarrow A$ is redundant because $\{D \rightarrow A, B \rightarrow D\} \models B \rightarrow A$;

Minimal Cover - Examples

- Given the set of FDs $\Sigma = \{B \rightarrow A, D \rightarrow A, AB \rightarrow D\}$, we can compute the minimal cover of Σ as follows:

1. start from Σ;
2. check whether all the FDs in Σ have only one attribute on the right hand side (look good);
3. $AB \rightarrow D$ can be replaced by $B \rightarrow D$;
4. look for a redundant FD in $\{B \rightarrow A, D \rightarrow A, B \rightarrow D\}$
   - check whether $B \rightarrow A$ is redundant?
   - $B \rightarrow A$ is redundant because $\{D \rightarrow A, B \rightarrow D\} \models B \rightarrow A$;

Therefore, the minimal cover of Σ is $\{D \rightarrow A, B \rightarrow D\}$.

Minimal Cover

- **Theorem:**

The minimal cover of a set of functional dependencies Σ always exists but is not necessarily unique.

Minimal Cover

- **Theorem:**

  The minimal cover of a set of functional dependencies Σ always exists but is not necessarily unique.

- **Examples:** Consider the following set of functional dependencies:

$$\Sigma = \{A \rightarrow BC, B \rightarrow C, B \rightarrow A, C \rightarrow AB\}$$

Minimal Cover

- **Theorem:**

> The minimal cover of a set of functional dependencies Σ always exists but is not necessarily unique.

- **Examples:** Consider the following set of functional dependencies:

$$\Sigma = \{A \rightarrow BC, B \rightarrow C, B \rightarrow A, C \rightarrow AB\}$$

Σ has two different minimal covers:

- $\Sigma_1 = \{A \rightarrow B, B \rightarrow C, C \rightarrow A\}$
- $\Sigma_2 = \{A \rightarrow C, C \rightarrow B, B \rightarrow A\}$

Minimal Cover

- **Theorem:**

  The minimal cover of a set of functional dependencies Σ always exists but is not necessarily unique.

- **Examples:** Consider the following set of functional dependencies:

$$\Sigma = \{A \rightarrow BC, B \rightarrow C, B \rightarrow A, C \rightarrow AB\}$$

  Σ has two different minimal covers:

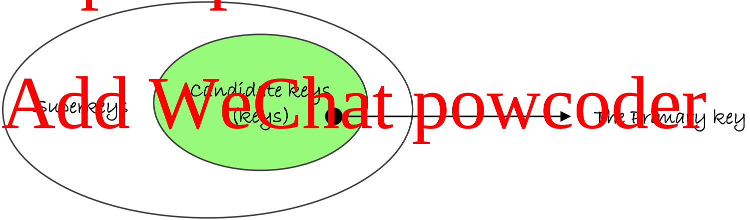
  - $\Sigma_1 = \{A \rightarrow B, B \rightarrow C, C \rightarrow A\}$
  - $\Sigma_2 = \{A \rightarrow C, C \rightarrow B, B \rightarrow A\}$

- The algorithm in the previous slide can find one, but not all minimal covers of a set of functional dependencies Σ.

Finding Keys

- Given a set Σ of FDs on a relation *R*, the question is:

How can we find all the (candidate) keys of *R*?



Finding Keys

- **Fact**: A key K of R always defines a FD $K \rightarrow R$.

[3] It extends Algorithm 15.2(a) in [Elmasri & Navathe, 7th edition, pp. 542], or Algorithm 2(a) or in Algorithm 2(a) in [Elmasri & Navathe, 6th edition pp. 558] to finding all keys of R

Finding Keys

- **Fact**: A key K of R always defines a FD $K \rightarrow R$.
- **Algorithm**[3]:

Input: a set Σ of FDs on R.

Output: the set of all keys of R.

[3] It extends Algorithm 15.2(a) in [Elmasri & Navathe, 7th edition, pp. 542], or Algorithm 2(a) or in Algorithm 2(a) in [Elmasri & Navathe, 6th edition pp. 558] to finding all keys of R

Australian National University

Finding Keys

- **Fact**: A key K of R always defines a FD $K \rightarrow R$.
- **Algorithm**[3]:

Input: a set Σ of FDs on R.

Output: the set of all keys of R.

[3] It extends Algorithm 15.2(a) in [Elmasri & Navathe, 7th edition, pp. 542], or Algorithm 2(a) or in Algorithm 2(a) in [Elmasri & Navathe, 6th edition pp. 558] to finding all keys of R

Finding Keys

- **Fact**: A key K of R always defines a FD $K \rightarrow R$.
- **Algorithm**[3]:

Input: a set Σ of FDs on R.

Output: the set of all keys of R.

- for every subset X of the relation R, compute its closure X^+

[3] It extends Algorithm 15.2(a) in [Elmasri & Navathe, 7th edition, pp. 542], or Algorithm 2(a) or in Algorithm 2(a) in [Elmasri & Navathe, 6th edition pp. 558] to finding all keys of R

Finding Keys

- **Fact**: A key K of R always defines a FD $K \rightarrow R$.
- **Algorithm**[3]:

Input: a set Σ of FDs on R.

Output: the set of all keys of R.

- for every subset X of the relation R, compute its closure X^+
- if $X^+ = R$, then X is a superkey.

[3] It extends Algorithm 15.2(a) in [Elmasri & Navathe, 7th edition, pp. 542], or Algorithm 2(a) or in Algorithm 2(a) in [Elmasri & Navathe, 6th edition pp. 558] to finding all keys of R

Finding Keys

- **Fact**: A key K of R always defines a FD $K \rightarrow R$.
- **Algorithm**[3]:

  Input: a set Σ of FDs on R.

  Output: the set of all keys of R.

  - for every subset X of the relation R, compute its closure X^+
  - if $X^+ = R$, then X is a superkey.
  - if no proper subset Y of X with $Y^+ = R$, then X is a key.

[3] It extends Algorithm 15.2(a) in [Elmasri & Navathe, 7th edition, pp. 542], or Algorithm 2(a) or in Algorithm 2(a) in [Elmasri & Navathe, 6th edition pp. 558] to finding all keys of R

Finding Keys

- **Fact**: A key K of R always defines a FD $K \rightarrow R$.
- **Algorithm**[3]:

  Input: a set Σ of FDs on R.

  Output: the set of all keys of R.
  - for every subset X of the relation R, compute its closure X^+
  - if $X^+ = R$, then X is a superkey.
  - if no proper subset Y of X with $Y^+ = R$, then X is a key.

- A **prime attribute** is an attribute occurring in a key, and a **non-prime attribute** is an attribute that is not a prime attribute.

[3] It extends Algorithm 15.2(a) in [Elmasri & Navathe, 7th edition, pp. 542], or Algorithm 2(a) or in Algorithm 2(a) in [Elmasri & Navathe, 6th edition pp. 558] to finding all keys of R

Exercises - Keys and Minimal Cover

- Consider RENTAL = {CustID, CustName, PropertyNo, DateStart, Owner} and the following set Σ of FDs:
  - $\{\text{CustID}\} \rightarrow \{\text{CustName}\}$
  - $\{\text{PropertyNo, StartDate}\} \rightarrow \{\text{CustID}\}$
  - $\{\text{PropertyNo, CustID}\} \rightarrow \{\text{StartDate}\}$
  - $\{\text{CustID, StartDate}\} \rightarrow \{\text{PropertyNo}\}$
  - $\{\text{Owner}\} \rightarrow \{\text{PropertyNo}\}$

- **Questions:**
  1. What are the keys of RENTAL?
  2. What is a minimal cover of Σ?

Exercises - Keys and Minimal Cover

- Consider RENTAL={CustID, CustName, PropertyNo, DateStart, Owner} and its FDs in the abbreviated form as
  - R={C, N, P, D, O}, and
  - $\Sigma = \{C \rightarrow N, PD \rightarrow C, CP \rightarrow D, OD \rightarrow P, O \rightarrow P\}$

- What are the keys of RENTAL?

Exercises - Keys and Minimal Cover

- Consider RENTAL={CustID, CustName, PropertyNo, DateStart, Owner} and its FDs in the abbreviated form as
  - R={C, N, P, D, O}, and
  - $\Sigma = \{C \to N, PD \to C, CP \to D, CD \to P, O \to P\}$

- What are the keys of RENTAL?
- **Solution**: Check $(X)^+$ for every subset of {C, N, P, D, O}.
  - O never appears in the dependent of any FD, O must be part of each key.

Exercises - Keys and Minimal Cover

- Consider RENTAL={CustID, CustName, PropertyNo, DateStart, Owner} and its FDs in the abbreviated form as
  - R={C, N, P, D, O}, and
  - $\Sigma = \{C \rightarrow N, PD \rightarrow C, CP \rightarrow D, OD \rightarrow P, O \rightarrow P\}$

- What are the keys of RENTAL?
- **Solution**: Check $(X)^+$ for every subset of {C, N, P, D, O}.
  - O never appears in the dependent of any FD, O must be part of each key.
  - $(O)^+ = OP$

Exercises - Keys and Minimal Cover

- Consider RENTAL={CustID, CustName, PropertyNo, DateStart, Owner} and its FDs in the abbreviated form as
  - R={C, N, P, D, O}, and
  - $\Sigma = \{C \rightarrow N, PD \rightarrow C, CP \rightarrow D, OD \rightarrow P, O \rightarrow P\}$

- What are the keys of RENTAL?
- **Solution**: Check $(X)^+$ for every subset of {C, N, P, D, O}.
  - O never appears in the dependent of any FD, O must be part of each key.
  - $(O)^+ = OP$
  - $(CO)^+ = CPNDO$, $(DO)^+ = CPNDO$ ...
  - Thus, {CustID, Owner} and {Owner, DateStart} are the keys.

Exercises - Keys and Minimal Cover

- Consider RENTAL={CustID, CustName, PropertyNo, DateStart, Owner} and its FDs in the abbreviated form as
  - R={C, N, P, D, O}, and
  - $\Sigma = \{C \rightarrow$ N, PD $\rightarrow$ C, CP $\rightarrow$ D, CD $\rightarrow$ P, O $\rightarrow$ P}

- What is a minimal cover of Σ?

Exercises - Keys and Minimal Cover

- Consider RENTAL={CustID, CustName, PropertyNo, DateStart, Owner} and its FDs in the abbreviated form as
  - R={C, N, P, D, O}, and
  - $\Sigma = \{C \rightarrow N, PD \rightarrow C, CP \rightarrow D, CD \rightarrow P, O \rightarrow P\}$

- What is a minimal cover of Σ?
- **Solution**:
  1. start from Σ

Exercises - Keys and Minimal Cover

- Consider RENTAL={CustID, CustName, PropertyNo, DateStart, Owner} and its FDs in the abbreviated form as
  - R={C, N, P, D, O}, and
  - $\Sigma = \{C \rightarrow$ N, PD $\rightarrow$ C, CP $\rightarrow$ D, CD $\rightarrow$ P, O $\rightarrow$ P$\}$

- What is a minimal cover of Σ?
- **Solution**:
  1. start from Σ;
  2. check whether all the FDs in Σ have only one attribute on the right hand side (look good);

Exercises - Keys and Minimal Cover

- Consider RENTAL={CustID, CustName, PropertyNo, DateStart, Owner} and its FDs in the abbreviated form as
  - R={C, N, P, D, O}, and
  - $\Sigma = \{C \rightarrow$ N, PD $\rightarrow$ C, CP $\rightarrow$ D, CD $\rightarrow$ P, O $\rightarrow$ P}

- What is a minimal cover of Σ?
- **Solution**:

1. start from Σ;
2. check whether all the FDs in Σ have only one attribute on the right hand side (look good);
3. determine if PD $\rightarrow$ C, CP $\rightarrow$ D and CD $\rightarrow$ P have any redundant attribute on the left hand side (look good);

Exercises - Keys and Minimal Cover

- Consider RENTAL={CustID, CustName, PropertyNo, DateStart, Owner} and its FDs in the abbreviated form as
  - R={C, N, P, D, O}, and
  - $\Sigma = \{C \rightarrow$ N, PD $\rightarrow$ C, CP $\rightarrow$ D, CD $\rightarrow$ P, O $\rightarrow$ P$\}$

- What is a minimal cover of Σ?

- **Solution**:

  1. start from Σ;
  2. check whether all the FDs in Σ have only one attribute on the right hand side (look good);
  3. determine if PD → C, CP → D and CD → P have any redundant attribute on the left hand side (look good);
  4. look for a redundant FD in Σ (none of FDs in Σ are redundant);

Exercises - Keys and Minimal Cover

- Consider RENTAL={CustID, CustName, PropertyNo, DateStart, Owner} and its FDs in the abbreviated form as
  - R={C, N, P, D, O}, and
  - $\Sigma = \{C \rightarrow N, PD \rightarrow C, CP \rightarrow D, CD \rightarrow P, O \rightarrow P\}$

- What is a minimal cover of Σ?

- **Solution**:
  1. start from Σ;
  2. check whether all the FDs in Σ have only one attribute on the right hand side (look good);
  3. determine if $PD \rightarrow C$, $CP \rightarrow D$ and $CD \rightarrow P$ have any redundant attribute on the left hand side (look good);
  4. look for a redundant FD in Σ (none of FDs in Σ are redundant);

Therefore, Σ is a minimal cover itself.

Accommodation Database

- Consider the following:
  - HOTEL(hotelNo, hotelName, city) with PK {hotelNo}
  - ROOM(roomNo, hotelNo, type, price) with PK {roomNo, hotelNo}
  - GUEST(guestNo, guestName, guestAddress) with PK {guestNo}
  - BOOKING(guestNo, hotelNo, date, roomNo) with PK {??}

Accommodation Database

- Consider the following:
  - HOTEL(hotelNo, hotelName, city) with PK {hotelNo}
  - ROOM(roomNo, hotelNo, type, price) with PK {roomNo, hotelNo}
  - GUEST(guestNo, guestName, guestAddress) with PK {guestNo}
  - BOOKING(guestNo, hotelNo, date, roomNo) with PK {??}
- We have some requirements on BOOKING:

R1 A booking can be made for one day only.
R2 A guest can make several bookings in a hotel for different days.
R3 A guest cannot make two or more bookings in the same hotel for the same day.
R4 A guest can make two or more bookings in different hotels for the same day.
R5 A room in any hotel can only be booked by one guest on the same date, i.e., no *double-booking*.

How to Identify FDs?

- Consider the following:
  - HOTEL(hotelNo, hotelName, city) with PK {hotelNo}
  - ROOM(roomNo, hotelNo, type, price) with PK {roomNo, hotelNo}
  - GUEST(guestNo, guestName, guestAddress) with PK {guestNo}
  - BOOKING(guestNo, hotelNo, date, roomNo) with PK {?}

- Which functional dependency does the following requirement imply?

  R1 A booking can be made for one day only.

How to Identify FDs?

- Consider the following:
  - HOTEL(hotelNo, hotelName, city) with PK {hotelNo}
  - ROOM(roomNo, hotelNo, type, price) with PK {roomNo, hotelNo}
  - GUEST(guestNo, guestName, guestAddress) with PK {guestNo}
  - BOOKING(guestNo, hotelNo, date, roomNo) with PK {?}

- Which functional dependency does the following requirement imply?

R1 A booking can be made for one day only.

$\hookrightarrow$ {guestNo, hotelNo, roomNo} $\rightarrow$ {date}?

How to Identify FDs?

- Consider the following
  - HOTEL(hotelNo, hotelName, city) with PK {hotelNo}
  - ROOM(roomNo, hotelNo, type, price) with PK {roomNo, hotelNo}
  - GUEST(guestNo, guestName, guestAddress) with PK {guestNo}
  - BOOKING(guestNo, hotelNo, date, roomNo) with PK {?}

- Which functional dependency does the following requirement imply?

R1 A booking can be made for one day only.

$\hookrightarrow$ {guestNo, hotelNo, roomNo} $\rightarrow$ {date}? **No**

How to Identify FDs?

- Consider the following
  - HOTEL(hotelNo, hotelName, city) with PK {hotelNo}
  - ROOM(roomNo, hotelNo, type, price) with PK {roomNo, hotelNo}
  - GUEST(guestNo, guestName, guestAddress) with PK {guestNo}
  - BOOKING(guestNo, hotelNo, date, roomNo) with PK {?}

- Which functional dependency does the following requirement imply?

R1 A booking can be made for one day only.

↪ {guestNo, hotelNo, roomNo} → {date}? **No**

guestNo	hotelNo	roomNo	Date
001	H1	R101	28/08/2020
001	H1	R101	29/08/2020

How to Identify FDs?

- Consider the following:
  - HOTEL(hotelNo, hotelName, city) with PK {hotelNo}
  - ROOM(roomNo, hotelNo, type, price) with PK {roomNo, hotelNo}
  - GUEST(guestNo, guestName, guestAddress) with PK {guestNo}
  - BOOKING(guestNo, hotelNo, date, roomNo) with PK {?}

- Which functional dependency does the following requirement imply?

R2 A guest can make several bookings in a hotel for different days.

Australian National University

How to Identify FDs?

- Consider the following:
  - HOTEL(hotelNo, hotelName, city) with PK {hotelNo}
  - ROOM(roomNo, hotelNo, type, price) with PK {roomNo, hotelNo}
  - GUEST(guestNo, guestName, guestAddress) with PK {guestNo}
  - BOOKING(guestNo, hotelNo, date, roomNo) with PK {?}

- Which functional dependency does the following requirement imply?

R2 A guest can make several bookings in a hotel for different days.

None

How to Identify FDs?

- Consider the following:
  - HOTEL(hotelNo, hotelName, city) with PK {hotelNo}
  - ROOM(roomNo, hotelNo, type, price) with PK {roomNo, hotelNo}
  - GUEST(guestNo, guestName, guestAddress) with PK {guestNo}
  - BOOKING(guestNo, hotelNo, date, roomNo) with PK {??}
- Which functional dependency does the following requirement imply?

  R3 A guest cannot make two or more bookings in the same hotel for the same day.

How to Identify FDs?

- Consider the following:
  - HOTEL(hotelNo, hotelName, city) with PK {hotelNo}
  - ROOM(roomNo, hotelNo, type, price) with PK {roomNo, hotelNo}
  - GUEST(guestNo, guestName, guestAddress) with PK {guestNo}
  - BOOKING(guestNo, hotelNo, date, roomNo) with PK {??}

- Which functional dependency does the following requirement imply?

  R3 A guest cannot make two or more bookings in the same hotel for the same day.

  $\hookrightarrow$ {guestNo, hotelNo, date} $\rightarrow$ {roomNo}?

How to Identify FDs?

- Consider the following:
  - HOTEL(hotelNo, hotelName, city) with PK {hotelNo}
  - ROOM(roomNo, hotelNo, type, price) with PK {roomNo, hotelNo}
  - GUEST(guestNo, guestName, guestAddress) with PK {guestNo}
  - BOOKING(guestNo, hotelNo, date, roomNo) with PK {??}
- Which functional dependency does the following requirement imply?

R3 A guest cannot make two or more bookings in the same hotel for the same day.

$\hookrightarrow$ {guestNo, hotelNo, date} $\rightarrow$ {roomNo}? **Yes**

guestNo	hotelNo	roomNo	Date
001	H1	R101	29/08/2020
001	H1	R102 ×	29/08/2020

How to Identify FDs?

- Consider the following:
  - HOTEL(hotelNo, hotelName, city) with PK {hotelNo}
  - ROOM(roomNo, hotelNo, type, price) with PK {roomNo, hotelNo}
  - GUEST(guestNo, guestName, guestAddress) with PK {guestNo}
  - BOOKING(guestNo, hotelNo, date, roomNo) with PK {?}

- Which functional dependency does the following requirement imply?

R4 A guest can make two or more bookings in different hotels for the same day.

How to Identify FDs?

- Consider the following:
  - HOTEL(hotelNo, hotelName, city) with PK {hotelNo}
  - ROOM(roomNo, hotelNo, type, price) with PK {roomNo, hotelNo}
  - GUEST(guestNo, guestName, guestAddress) with PK {guestNo}
  - BOOKING(guestNo, hotelNo, date, roomNo) with PK {?}

- Which functional dependency does the following requirement imply?

R4 A guest can make two or more bookings in different hotels for the same day.

None

How to Identify FDs?

- Consider the following:
  - HOTEL(hotelNo, hotelName, city) with PK {hotelNo}
  - ROOM(roomNo, hotelNo, type, price) with PK {roomNo, hotelNo}
  - GUEST(guestNo, guestName, guestAddress) with PK {guestNo}
  - BOOKING(guestNo, hotelNo, date, roomNo) with PK {??}
- Which functional dependency does the following requirement imply?

R5 A room in any hotel can only be booked by one guest on the same date, i.e., no *double-booking*.

$\hookrightarrow$ {hotelNo, date, roomNo} $\rightarrow$ {guestNo}

How to Identify FDs?

- Consider the following:
  - HOTEL(hotelNo, hotelName, city) with PK {hotelNo}
  - ROOM(roomNo, hotelNo, type, price) with PK {roomNo, hotelNo}
  - GUEST(guestNo, guestName, guestAddress) with PK {guestNo}
  - BOOKING(guestNo, hotelNo, date, roomNo) with PK {?}

- Which functional dependency does the following requirement imply?

R5 A room in any hotel can only be booked by one guest on the same date, i.e., no *double-booking*.

$\hookrightarrow$ {hotelNo, date, roomNo} $\rightarrow$ {guestNo} **Yes**

guestNo	hotelNo	roomNo	Date
001	H1	R101	29/08/2020
002 ×	H1	R101	29/08/2020

How to Find Candidate Keys?

- Consider the following:
  - HOTEL(hotelNo, hotelName, city) with PK {hotelNo}
  - ROOM(roomNo, hotelNo, type, price) with PK {roomNo, hotelNo}
  - GUEST(guestNo, guestName, guestAddress) with PK {guestNo}
  - BOOKING(guestNo, hotelNo, date, roomNo) with PK {?}

- FDs on BOOKING
  - {guestNo, hotelNo, date} $\rightarrow$ {roomNo} by **R3**
  - {hotelNo, date, roomNo} $\rightarrow$ {guestNo} by **R5**

How to Find Candidate Keys?

- Consider the following
  - HOTEL(hotelNo, hotelName, city) with PK {hotelNo}
  - ROOM(roomNo, hotelNo, type, price) with PK {roomNo, hotelNo}
  - GUEST(guestNo, guestName, guestAddress) with PK {guestNo}
  - BOOKING(guestNo, hotelNo, date, roomNo) with PK {?}

- FDs on BOOKING
  - {guestNo, hotelNo, date} $\rightarrow$ {roomNo} by **R3**
  - {hotelNo, date, roomNo} $\rightarrow$ {guestNo} by **R5**

- Candidate keys on BOOKING
  - {guestNo, hotelNo, date}
  - {hotelNo, date, roomNo}

Australian National University

How to Identify FDs?

- Consider BOOKING(guestNo, hotelNo, date, roomNo) and the following changes:

R1 A booking can be made for one day only.
R2 A guest can make several bookings in a hotel for different days.
R3 ~~A guest cannot make two or more bookings in the same hotel for the same day.~~
R4 ~~A guest can make two or more bookings in different hotels for the same day.~~
R5 A room in any hotel can only be booked by one guest on the same date, i.e., no *double-booking*.
R6 A guest is not allowed to make more than one booking for the same day even in the different hotels.

How to Identify FDs?

- Consider the following:
  - HOTEL(hotelNo, hotelName, city) with PK {hotelNo}
  - ROOM(roomNo, hotelNo, type, price) with PK {roomNo, hotelNo}
  - GUEST(guestNo, guestName, guestAddress) with PK {guestNo}
  - BOOKING(guestNo, hotelNo, date, roomNo) with PK {?}

- Which functional dependency does the following requirement imply?

  R6 A guest is not allowed to make more than one booking for the same day even in the different hotels.

How to Identify FDs?

- Consider the following:
  - HOTEL(hotelNo, hotelName, city) with PK {hotelNo}
  - ROOM(roomNo, hotelNo, type, price) with PK {roomNo, hotelNo}
  - GUEST(guestNo, guestName, guestAddress) with PK {guestNo}
  - BOOKING(guestNo, hotelNo, date, roomNo) with PK {?}

- Which functional dependency does the following requirement imply?

R6 A guest is not allowed to make more than one booking for the same day even in the different hotels.

$\hookrightarrow$ {guestNo, date} $\rightarrow$ {hotelNo, roomNo}

How to Find Candidate Keys?

- Consider the following
  - HOTEL(hotelNo, hotelName, city) with PK {hotelNo}
  - ROOM(roomNo, hotelNo, type, price) with PK {roomNo, hotelNo}
  - GUEST(guestNo, guestName, guestAddress) with PK {guestNo}
  - BOOKING(guestNo, hotelNo, date, roomNo) with PK {?}

- FDs on BOOKING
  - {hotelNo, date, roomNo} $\rightarrow$ {guestNo} by **R5**
  - {guestNo, date} $\rightarrow$ {hotelNo, roomNo} by **R6**

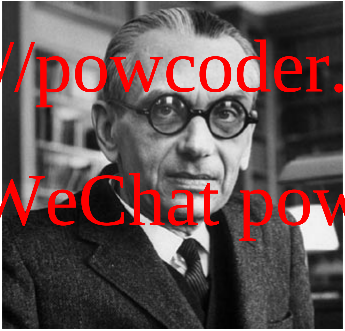
How to Find Candidate Keys?

- Consider the following
  - HOTEL(hotelNo, hotelName, city) with PK {hotelNo}
  - ROOM(roomNo, hotelNo, type, price) with PK {roomNo, hotelNo}
  - GUEST(guestNo, guestName, guestAddress) with PK {guestNo}
  - BOOKING(guestNo, hotelNo, date, roomNo) with PK {?}

- FDs on BOOKING
  - {hotelNo, date, roomNo} → {guestNo} by **R5**
  - {guestNo, date} → {hotelNo, roomNo} by **R6**

- Candidate keys on BOOKING
  - {hotelNo, date, roomNo}
  - {guestNo, date}

(credit cookie) Kurt Gödel and Incompleteness Theorem



Kurt Gödel (1906-1978)

Armstrong's Inference Rules

- **Two questions:**
  - Are all the FDs inferred using the Armstrong's inference rules correct?
    $\rightsquigarrow$ **soundness (you cannot prove anything that is wrong)**

  - Can we use the Armstrong's inference rules to infer all possible FDs?
    $\rightsquigarrow$ **completeness (you can prove anything that is right)**

- **Theorem** (W. W. Armstrong)
  - The Armstrong's inference rules are both **sound** and **complete**.

Hilbert's program (1920s)

- **Formulation of mathematics**: formalize all true mathematical statements
- **Completeness**: all true mathematical statements can be proved
- **Consistency**: no contradiction can be obtained in the formalism
- **Decidability**: decide the truth or falsity of any mathematical statement.

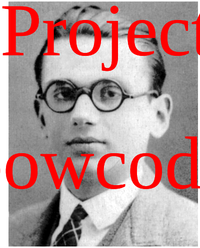
Hilbert's program (1920s)

- **Formulation of mathematics**: formalize all true mathematical statements
- **Completeness**: all true mathematical statements can be proved
- **Consistency**: no contradiction can be obtained in the formalism
- **Decidability**: decide the truth or falsity of any mathematical statement.

David Hilbert (1862-1943)

We must know. We will know.

Kurt Gödel and Incompleteness Theorem



Kurt Gödel
(1906-1978)

- **Theorem** (Kurt Gödel, 1931)
  For any computable axiomatic system that is powerful enough to describe the arithmetic of the natural numbers, **there will always be at least one true but unprovable statement**.

Kurt Gödel and Gödel Prize

Kurt Gödel
(1906-1978)

John von Neumann
(1903-1957)

- Kurt Gödel’s achievement in modern logic is singular and monumental – indeed it is more than a monument, it is a landmark which will remain visible far in space and time. — **John von Neumann**

Kurt Gödel and Gödel Prize

Kurt Gödel
(1906-1978)

John von Neumann
(1903-1957)

- Kurt Gödel's achievement in modern logic is singular and monumental – indeed it is more than a monument, it is a landmark which will remain visible far in space and time. — **John von Neumann**
- The **Gödel prize** became an annual prize for outstanding papers in the area of theoretical computer science since 1993.