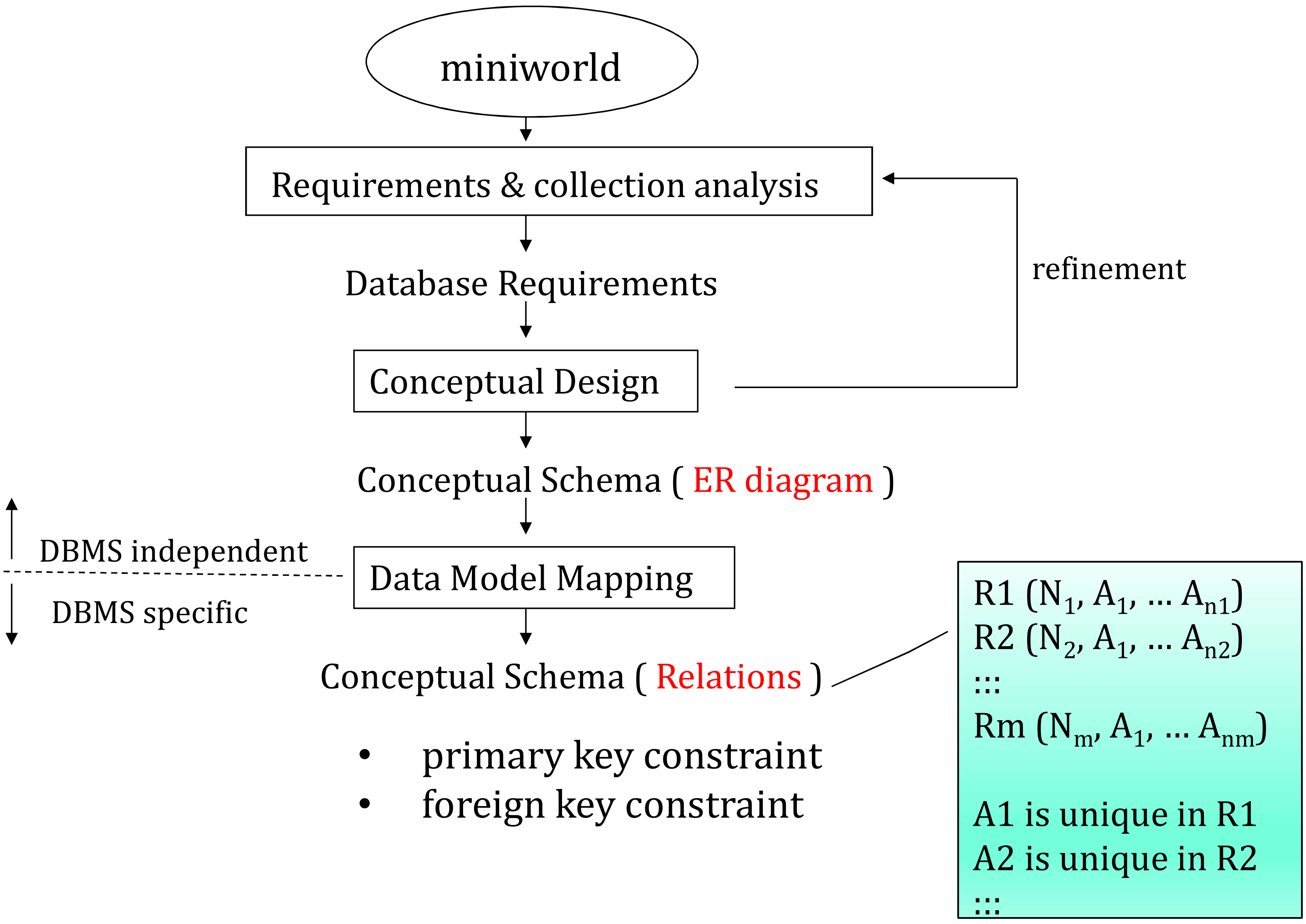
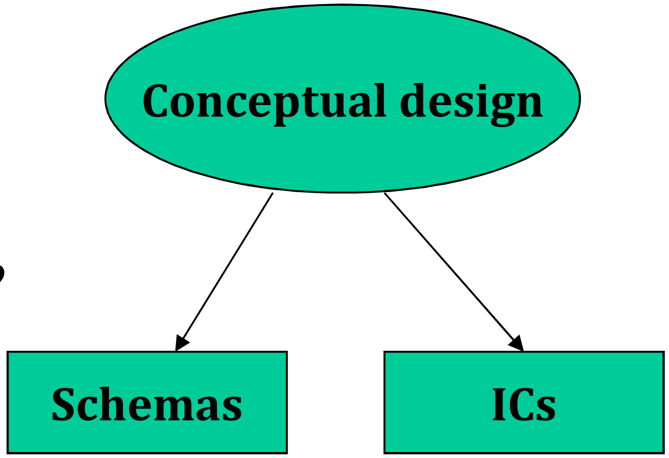

miniworld
Requirements & collection analysis
Database Requirements
refinement
Conceptual Design
Conceptual Schema (ER diagram)
DBMS independent
DBMS specific
Data Model Mapping
Conceptual Schema (Relations)
• primary key constraint
• foreign key constraint
R1 (N_1, A_1, ... A_{n1})
R2 (N_2, A_1, ... A_{n2})
:::
Rm (N_m, A_1, ... A_{nm})
A1 is unique in R1
A2 is unique in R2
:::

Schema Refinement and Normal Forms

- Conceptual database design gives us a set of relation schemas and integrity constraints
- Given a design, can we have a machine tell us if it is a good design? And if not, can the machine make it a good design?
    - A design can be evaluated from various perspectives, here our focus is on data redundancy



Redundancy is at the root of several problems associated with relational schemas:

- redundant storage
- Insertion/update/deletion anomalies

Example

- Schema: Hourly_Emps (*<u>ssn</u>, name, lot, rating, hrly_wages, hrs_worked*)
- Constraints:
  1. *ssn* is the primary key
  2. If two tuples have the same value on *rating*, they have the same value on *hrly_wages*

SSN	Name	Lot	Rd d	W	H
123-22-3666	Attishoo	48	8	10	40
231-31-5368	Smiley	22	8	10	30
131-24-3650	Smethurst	35	5	7	30
434-26-3751	Guldu	35	5	7	32
612-67-4134	Madayan	35	8	10	40

Problems

- Redundant storage: (rating value 8, hourly wage 10) is repeated three times
- Update anomaly: The hourly_wages in the first tuple could be updated without making a similar change in the second tuple
- Insertion anomaly: We cannot insert a tuple for an employee unless we know the hourly wage for the employee's rating value
- Deletion anomaly: If we delete all tuples with a given rating value, we lose the associateion between the rating value and its hourly_wage value

Solution: Decomposition

If we break Hourly_Emps into Hourly_Emps2 and Wages, then we don't have updates, insertion, deletion anomalies.

Hourly_Emps2

S	N	L	R	H
123-22-3666	Attishoo	48	8	40
231-31-5368	Smiley	22	8	30
131-24-3650	Smethurst	35	5	30
434-26-3751	Guldu	35	5	32
612-67-4134	Madayan	35	8	40

Wages

R	W
8	10
5	7

Decomposition Concerns

- Should a relation be decomposed?
  - If a relation is not in certain form, some problems (e.g., redundancy) will arise, are these problems tolerable?
    - Aforementioned anomalies
    - Potential performance loss: Queries over the original relation may required to join the decomposed relations
- How to decompose a relation? Two properties must be preserved:
  - lossless-join: the data in the original relation can be recovered from the smaller relations
  - dependency-preservation: all constraints on the original relation must still hold by enforcing some constraints on each of the small relations

Functional Dependencies (FDs)

In a relation schema R, a set of attributes X functionally determines another set of attributes Y if and only if whenever two tuples of R agree on X value, they must necessarily agree on the Y value.

$$X \rightarrow Y \quad \Leftrightarrow \quad \forall t_1, t_2 \in r(R),$$

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

where X and Y are R's attributes, r(R) is an instance of R, t1 and t2 are two tuples in r(R)

How to read $X \rightarrow Y$:

- Y is functionally dependent on X, or
- X uniquely determines Y or
- X functionally determines Y, or
- X determines Y

Suppose we have X->Y. Does this data set violate this dependency?

X	Y	Z
X1	Y2	Z1
X1	Y2	Z2
X2	Y2	Z3

Does this data set violate Z->Y?

Does this data set violate X->Y?
Does this data set violate XY->Z?
Does this data set violate Z->X?

X	Y	Z
X1	Y1	Z1
X1	Y1	Z2
X1	Y2	Z1

Dependency Reasoning

The challenge of checking dependency preservation stems from the fact that a set of dependencies may imply some additional dependencies.

EMP_DEPT(ENAME,<u>SSN</u>,BDATE,ADDRESS,DNUMBER,DNAME,DMGRSSN)

F={SSN->{ENAME,BDATE,ADDRESS,DNUMBER},
DNUMBER->{DNAME,DMGRSSN} }

F infers the following additional functional dependencies:

$F \models$ {SSN}->{DNAME,DMGRSSN}

$F \models$ {SSN}->{SSN}

$F \models$ {DNUMBER}->{DNAME}

Some important questions

1. Given a set of attributes X, what attributes can be determined by X
2. Given an FD set, what other dependencies are implied
3. Given an FD set F, what is the minimum set of dependencies that is equivalent to F

Armstrong's Axiom 1: Reflexivity

Let X and Y be two sets of attributes in R.

If $X \supseteq Y$, then $X \rightarrow Y$.

PROOF

Let $\{t_1, t_2\} \subseteq r(R)$ such that $t_1[X]=t_2[X]$

Since $X \supseteq Y$, $t_1[X]=t_2[X] \Rightarrow t_1[Y]=t_2[Y]$

$$\Rightarrow X \rightarrow Y.$$

TIP: When proving things, the best way is always going back to the basic definition, function dependency

Armstrong's Axiom 2: Augmentation

Let X, Y, and Z be three sets of attributes in R.

If $X \rightarrow Y$, then $XZ \rightarrow YZ$ for any Z.

PROOF

Assume that the Augmentation rule is not true.

$$\Rightarrow \exists t_1, t_2 \in r(R)$$

$t_1[X] = t_2[X]$ (1)
$t_1[Y] = t_2[Y]$ (2)
$t_1[XZ] = t_2[XZ]$ (3)
$t_1[YZ] \mathrel{!}= t_2[YZ]$ (4)

(1)&(3) $\Rightarrow$ $t_1[Z]=t_2[Z]$ (5)

(2)&(5) $\Rightarrow$ $t_1[YZ]=t_2[YZ]$ (6)

(6) Contradicts (4)

Armstrong's Axiom 3: Transitivity

Let X, Y, and Z be three sets of attributes in R.

If $X \rightarrow Y$ and $Y \rightarrow Z$, then $X \rightarrow Z$

PROOF

Let $X \rightarrow Y$ and (1)

$Y \rightarrow Z$ (2)

$\forall t_1, t_2 \in r(R)$ such that $t_1[X]=t_2[X]$, (3)
we have:

(1) $t_1[Y]=t_2[Y]$ (4)

(2)&(4) $t_1[Z]=t_2[Z]$ (5)

(3)&(5) $X \rightarrow Z$

Properties of Armstrong's Axioms

- Soundness
  - All dependencies generated by the Axioms are correct
- Completeness
  - Repeatedly applying these rules can generate all correct dependency (i.e., any FDs in F^+ be generated)

Question: Other than Armstrong's axioms, do there exist other axioms which also have these two properties?

Armstrong's Axioms

- If $X \supseteq Y$, then $X \rightarrow Y$.
- If $X \rightarrow Y$, then $XZ \rightarrow YZ$ for any Z.
- If $X \rightarrow Y$ and $Y \rightarrow Z$, then $X \rightarrow Z$

Use Armstrong axioms to derive some other useful inference rules

- Union

  $X \rightarrow XZ \quad XZ \rightarrow YZ$

  If $X \rightarrow Y$ and $X \rightarrow Z$,
  then $X \rightarrow Y\boxed{Z}$.

- Decomposition

  If $X \rightarrow YZ$,
  then $X \rightarrow Y$ and $X \rightarrow Z$.

- Pseudotransitive Rule

  If $X \rightarrow Y$ and $WY \rightarrow Z$
  then $WX \rightarrow Z$.

Use Armstrong axioms to derive some other useful inference rules

- **Union rule**: If $X \rightarrow Y$ and $X \rightarrow Z$, then $X \rightarrow YZ$.

Given $X \rightarrow Y$ and (1)

$X \rightarrow Z$. (2)

Applying Augmentation rule on (1), we have

$XX \rightarrow XY \Longrightarrow X \rightarrow XY$. (3)

Applying Augmentation rule on (2), we have

$XY \rightarrow ZY \Longrightarrow XY \rightarrow YZ$. (4)

Applying Transitive rule on (3) and (4), we have

$X \rightarrow YZ$.

Use Armstrong axioms to derive some other useful inference rules

- Decomposition rule: If $X \rightarrow YZ$ then $X \rightarrow Y$ and $X \rightarrow Z$.

$X \rightarrow YZ$

Given $X \rightarrow YZ$. (1)

Since $Y \subseteq YZ$, reflexive rule gives

$YZ \rightarrow Y$. (2)

$YZ \rightarrow Z$

Applying Transitive rule on (1) and (2), we have $X \rightarrow Y$.

$X \rightarrow Z$ is derived in a similar way.

Use Armstrong axioms to derive some other useful inference rules

- Pseudotransitive rule: If $X \rightarrow Y$ and $WY \rightarrow Z$, then $WX \rightarrow Z$.

Given $X \rightarrow Y$ (1)
and $WY \rightarrow Z$. (2)

Applying Augmentation rule on (1), we have
$WX \rightarrow WY$. (3)

Applying Transitive rule on (3)&(2), we have
$WX \rightarrow Z$.

Exercise

$X \to Z$
$\Uparrow$
$X \to Y,\ Y \to Z$

- Prove or disprove the following inference rules

1. $\{W \to Y,\ X \to Z\} \Rightarrow \{WX \to YZ\}$ ✓

2. $\{X \to Y,\ X \to W,\ WY \to Z\} \Rightarrow \{X \to Z\}$

   $X \to YW$ $X \to Z$

3. $\{X \to Y\} \Rightarrow \{X \to YZ\}$

   ✓ $ZX \to YZ \Rightarrow XZ \to Y$

   $X_1\ Y_1\ Z_1$
   $X_1\ Y_1\ Z_2$

4. $\{X \to Y,\ Z \to Y\} \Rightarrow \{XZ \to Y\}$ ✓

   $YZ \to Y$
   $\boxed{X} \supseteq Y$
   $X \to Y$

- Prove using inference rules
- Disprove by showing a counter example

Solutions

- $\{W \rightarrow Y, X \rightarrow Z\} \Rightarrow \{WX \rightarrow YZ\}$
  - Proof:
    - $WX \rightarrow YX$
    - $YX \rightarrow YZ$
- $\{X \rightarrow Y, X \rightarrow W, WY \rightarrow Z\} \Rightarrow \{X \rightarrow Z\}$
  - Proof
    - $X \rightarrow YW \rightarrow Z$
    - $X \rightarrow Z$
- $\{X \rightarrow Y\} \Rightarrow \{X \rightarrow YZ\}$
  - Counter example
    - X1 Y1 Z1
    - X1 Y1 Z2
- $\{X \rightarrow Y\} \Rightarrow \{XZ \rightarrow Y\}$
    - $XZ \rightarrow YZ \rightarrow Y$
- $\{X \rightarrow Y, Z \rightarrow Y\} \Rightarrow \{XZ \rightarrow Y\}$
  - Proof
    - $X \rightarrow Y \Rightarrow XZ \rightarrow YZ \rightarrow Y \Rightarrow XZ \rightarrow Y$

X^+: Closure of Attribute Set X

Let F be a set of functional dependencies on a set of attributes U and let $X \subseteq U$. We define X^+ to be the set of all attributes that are dependent on X (under F).

$$X^+ = \{A \mid X \to A\}$$

X^+ enables us to tell at a glance whether a dependency $X \to A$ follows from F.

For example, $X^+=\{ABC\}$, then we have $X \to ABC \to A$, so $X \to A$

Algorithm to determine X^+ under F

$X^+ = X$;

Repeat until there is no change: {

if there is an FD $A \rightarrow B$ in F such that $A \subseteq X^+$

then $X^+ = X^+ \cup B$

}

Example 1

EMP_PROJ(SSN, PNUMBER, HOURS, ENAME, PNAME, PLOCATION)

F={ {SSN}->{ENAME},
{PNUMBER}->{PNAME, PLOCATION},
{SSN, PNUMBER}→{HOURS} }

(a) Compute $\{SSN\}^+$

Initialization: $\{SSN\}^+ = \{SSN\}$
1st iteration: NEW={ENAME}
$\{SSN\}^+ = \{SSN, ENAME\}$
2nd iteration: NEW={}
$\{SSN\}^+ = \{SSN, ENAME\}$

EMP_PROJ(<u>SSN,PNUMBER</u>, HOURS, ENAME, PNAME, PLOCATION)

F={ {SSN}->{ENAME},
{PNUMBER}->{PNAME,PLOCATION},
{SSN,PNUMBER}→{HOURS} }

(b) Compute $\{PNUMBER\}^+$

Initialization: $\{PNUMBER\}^+$={PNUMBER}
1st iteration: NEW={PNAME, PLOCATION}
$\{PNUMBER\}^+$={PNUMBER, PNAME, PLOCATION}
2nd iteration: NEW={}
$\{PNUMBER\}^+$={PNUMBER, PNAME, PLOCATION}

(c) Compute $\{SSN, PNUMBER\}^+$

Initialization: $\{SSN, PNUMBER\}^+$={SSN, PNUMBER}
1. NEW={ENAME, PNAME,PLOCATION}
$\{SSN, PNUMBER\}^+$={SSN, PNUMBER, ENAME, PNAME, PLOCATION}
2. NEW={HOURS}
$\{SSN, PNUMBER\}^+$={SSN, PNUMBER, ENAME, PNAME, PLOCATION, HOURS}

Algorithm to determine X^+ under F

$X^+ = X$;

Repeat until there is no change: {
if there is an FD $A \rightarrow B$ in F such that $A \subseteq X^+$
then $X^+ = X^+ \cup B$

Example 2

R(<u>A,B</u>,C,D,E,F), FD={ A->D, B->E, D$\rightarrow$B, C->F }

1. What is $\{A\}^+$ = {A, D, B, E}
2. What is $\{B\}^+$ = {B, E}
3. What is $\{E\}^+$ = {E}

F^+: Closure of Functional Dependency Set F

Given a set of functional dependencies F, we define F^+ to be the set of all functional dependencies that can be inferred from F.

Algorithm for computing F^+

1. $F^+ = \{\}$;
2. For each attribute set A in R, computing A^+
3. For each $X \rightarrow Y$ implied by A^+, add $X \rightarrow Y$ to F^+

Computing F^+

1.	$F^+ = \{\}$;
2.	For each attribute set A in R, computing A^+
3.	For each $X \rightarrow Y$ implied by A^+, add $X \rightarrow Y$ to F^+

Example. Consider R(A, B, C, D) and $F = \{A \rightarrow B, B \rightarrow C\}$.

$2^4 - 1$

1. To compute F^+, we enumerate all attribute sets and computes their closure
   - $\{A\}^+ = \{AB\}^+ = \{AC\}^+ = \{ABC\}^+ = \{A, B, C\}$
   - $\{B\}^+ = \{BC\}^+ = \{B, C\}$
   - $\{C\}^+ = \{C\}$
   - $\{D\}^+ = \{D\}$
   - $\{AD\}^+ = \{A, D\}$
   - $\{BD\}^+ = \{CD\}^+ = \{BCD\}^+ = \{B, C, D\}$
   - $\{ABD\}^+ = \{ABCD\}^+ = \{A, B, C, D\}$
   - $\{ACD\}^+ = \{A, C, D\}$
2. For each closure, generate all of its FDs and add to F^+

Equivalence of Sets of Functional Dependencies

Let E and F be two sets of functional dependencies.

- F covers E if $E \subseteq F^+$.
- E and F are equivalent if $E^+ = F^+$.
- $E^+ = F^+$ if and only if E covers F and F covers E.

Note: Equivalence means that every FD in E can be inferred from F, and every FD in F can be inferred from E.

Determine whether F covers E:
For each FD $X \rightarrow Y$ in E, calculate X^+ with respect to F, then check whether $X^+ \supseteq Y$.

EXAMPLE:

Check whether or not F is equivalent to G.

F={A→C, AC→D, E→AD,E→H}

G={A→CD, E→AH}

To show if G is covered by F, we need to prove that every FD in G can be implied by F

1. Does F imply A→CD? T
   - Compute A^+ wrt F
2. Does F imply E→AH? T
   - Compute E^+ wrt F

To show if F is covered by G, we need to prove that every FD in F can be implied by G

1. Does G imply A→C? T
2. Does G imply AC→D? T
3. Does G imply E→AD? T
4. Does G imply E→H? T

$A^+_G =$

$AC^+_G =$

$E^+_G =$

App1: Checking if $X \rightarrow Y$

- Steps of checking if an FD $X \rightarrow Y$ is in the closure of a set of FDs F :

  1. Compute X^+ wrt *F*.
  2. Check if Y is in X^+.
  3. $Y \in X^+ \Leftrightarrow X \rightarrow Y$ is in F^+

- Does $F = \{A \rightarrow B,\ B \rightarrow C,\ CD \rightarrow E\}$ imply $A \rightarrow E$?
  - i.e, is $A \rightarrow E$ in F^+? Equivalently, is E in A^+?
  - A^+ (w.r.t. F)=$\{A, B, C\}$
  - E is not in A^+, thus, $A \rightarrow E$ is not in F^+.

Minimal Cover of Functional Dependencies

A set of functional dependencies F is minimal if it satisfies the following three conditions:

- Every FD in F has a single attribute for its right-hand side. (This is a standard form, not a requirement.)
- We cannot replace any dependency $X \rightarrow A$ in F with a dependency $Y \rightarrow A$, where Y is a proper subset of X, and still have a set of dependencies that is equivalent to F.
- We cannot remove any dependency from F and still have a set of dependencies that is equivalent to F.

There can be several minimal covers for a set of functional dependencies!

Minimal Cover

F={X1→Y1, X2→Y2, …, Xn→Yn} is a minimum cover

1) Any Yi is a single attribute
2) For any Xi→Yi, it is impossible that X'→Y and X' is a subset of X
3) No Xi→Yi can be taken out
   - F' = F - {Xi→Yi} is not equivalent to F

Minimal Cover

<u>Definition</u>: A minimal cover of a set of FDs F is a minimal set of functional dependencies F_{min} that is equivalent to F.

<u>Procedure</u>: Find a minimal cover F_{min} for F.

1. Set F_{min}=F

/*put every FD in a standard form, i.e., it has a single attribute as its right-hand side*/

2. Replace each FD $X \rightarrow A_1, A_2, \ldots, A_n$ in F_{min} by the *n* FDs $X \rightarrow A_1, \ldots, X \rightarrow A_n$.

/* minimize the left side of each FD, i.e., every attribute is needed */

3. For each FD $X \rightarrow A$ in F_{min}
   - For each $B \in X$,
     - Let $T=(F_{min}- \{X \rightarrow A\}) \cup \{(X-\{B\}) \rightarrow A\}$
     - Check whether T is equivalent to F_{min} (1)
     - If (1) is true, then set F_{min} = T.

/* delete redundant FDs, i.e., No redundant FDs remain in Fmin. */

4. For each FD $X \rightarrow A$ in F_{min}

   Let $T=F_{min}-\{X \rightarrow A\}$

   Check whether T is equivalent to F_{min}. (2)

   If (2) is true, set F_{min} = T.

Example:

Find the minimal cover of the set

$F=\{ABCD \rightarrow E, E \rightarrow D, A \rightarrow B, AC \rightarrow D\}$

1) Make sure the right hand side is minimum
2) Make sure the left hand side is minimum
3) Make sure the whole set of dependencies is minimum

Example:

Find the minimal cover of the set
$F=\{ABCD\rightarrow E, E\rightarrow D, A\rightarrow B, AC\rightarrow D\}$.

Step 2: $F_{min}=\{ABCD\rightarrow E, E\rightarrow D, A\rightarrow B, AC\rightarrow D\}$

Step 3:

Replace $AC\rightarrow D$ with $A\rightarrow D$;

$T=\{ABCD\rightarrow E, E\rightarrow D, A\rightarrow B, A\rightarrow D\}$

Compute $\{A\}^+$ wrt to F, $\{A\}^+=\{A,B\}$

Compute $\{A\}^+$ wrt to T, $\{A\}^+=\{A,B,D\}$

Cannot replace F_{min} with T.

Is T equivalent to F_{min}?

Can Fmin Cover T?

No, keep $AC\rightarrow D$

Replace $ABCD\rightarrow E$ with $ACD\rightarrow E$,

$T=\{ACD\rightarrow E, E\rightarrow D, A\rightarrow B, AC\rightarrow D\}$

Compute $\{ACD\}^+$ wrt to F, $\{ACD\}^+=\{A,C,D,B,E\}$

Compute $\{ACD\}^+$ wrt to T,$\{ACD\}^+=\{A,C,D,E,B\}$

Replace F_{min} with T.

Step 3: (cont'd)
Replace $ACD \rightarrow E$ with $AC \rightarrow E$,
$T=\{AC \rightarrow E, E \rightarrow D, A \rightarrow B, AC \rightarrow D\}$
Compute $\{AC\}^+$ wrt to F, $\{AC\}^+=\{ACDBE\}$
Compute $\{AC\}^+$ wrt to T, $\{AC\}^+=\{ACEDB\}$
Replace F_{min} with T.

Step 4:
Consider $T=\{AC \rightarrow E, E \rightarrow D, AC \rightarrow D, A \rightarrow B\}$ (take out $AC \rightarrow D$)
$\{AC\}^+=\{A,C,E,D,B\}$ with respect to T;
$\{D\} \subseteq \{AC\}^+$; we don't have to include $AC \rightarrow D$

The minimal cover of F is:
$\{AC \rightarrow E, E \rightarrow D, A \rightarrow B\}$

App2: Finding a key K for relational schema R based on a set F of FDs

```
Set K=R
For each attribute A in K
      Compute (K-A)+ w.r.t. F
      If (K-A)+ contains all the attributes in R then set
      K= K-A
```

Examples

1. R={A, B, C, D}   F={A→B, B→C, AB→D}; find a key of R.

1. R={A, B, C, D, E, F} F={A->C, A->D, B->C, E->F}; find a key of R

- The algorithm returns only one key out of the possible candidate keys for R.
- The key returned depends on the order in which attributes are removed from R.

Quick Review: Important Concepts

- X^+ : Closure of an attribute set X
  - The set of all attributes that are determined by X
- F^+ : Closure of a dependency set F
  - The set of all dependencies that are implied from F
- F_{min}: a minimum cover of a dependency set F
  - A minimum set of FDs that is equivalent to F
- K: a key
  - A minimum set of attributes that determines all attributes

Questions

- $AB \to C \ \Rightarrow ? \, A \to C$
- $A \to C \ \Rightarrow ? \, AB \to C$

Reading

- https://en.wikipedia.org/wiki/Functional_dependency