

## 0.10 Closure Given Algorithm

you-said

For this algorithm: Closure of  $\beta$  under  $F$  ( $\beta^+$ ) result :=  $\beta$  Do For each  $\gamma \rightarrow \delta$  in  $F$ : If ( $\gamma \subseteq \text{result}$ ): result := result  $\cup$   $\delta$  While (result does not change) Return result Show if  $F = \{A \rightarrow B, A \rightarrow C, CG \rightarrow H, CG \rightarrow I, B \rightarrow H\}$ ,  $\beta = \{A, G\}$  satisfies closure for  $R(A, B, C, G, H, I)$

ChatGPT said:

To determine whether  $\beta = \{A, G\} = \{A, G\}$  satisfies closure for  $R(A, B, C, G, H, I)$  under the given set of functional dependencies  $F = \{A \rightarrow B, A \rightarrow C, CG \rightarrow H, CG \rightarrow I, B \rightarrow H\}$  we will compute the closure  $\beta^+$  of  $\beta$  using the provided algorithm step-by-step.

steps-to-calculate-the-closure-betaux3b2

**Steps to calculate the closure  $\beta^+$ :**

### 1. Initialization:

- Start with  $\beta = \{A, G\}$ .
- Initialize result =  $\{A, G\}$ .

### 2. Apply the algorithm:

- Loop through each functional dependency  $\gamma \rightarrow \delta$  in  $F$ .
- If  $\gamma \subseteq \text{result}$ , add  $\delta$  to result.

first-iteration

- Current result =  $\{A, G\}$
- **Consider the dependency  $A \rightarrow B$ :**

– Since  $A \subseteq \{A, G\}$ , we can apply this dependency.

- Update  $\text{result} := \text{result} \cup \{B\} = \{A, G, B\}$   $\text{result} := \text{result}\{B\} = \{A, G, B\}$ .

- **Consider the dependency  $A \rightarrow C$   $\mathbf{A} \rightarrow \mathbf{C}$ :**

- Since  $A \subseteq \{A, G\}$   $\mathbf{A} \subseteq \mathbf{A}\{A, G\}$ , we can apply this dependency.
- Update  $\text{result} := \text{result} \cup \{C\} = \{A, G, B, C\}$   $\text{result} := \text{result}\{C\} = \{A, G, B, C\}$ .

- **Consider the dependency  $CG \rightarrow H$   $\mathbf{CG} \rightarrow \mathbf{H}$ :**

- Since  $\{C, G\} \subseteq \{A, G, B, C\}$   $\{C, G\} \subseteq \mathbf{A}\{A, G, B, C\}$ , we can apply this dependency.
- Update  $\text{result} := \text{result} \cup \{H\} = \{A, G, B, C, H\}$   $\text{result} := \text{result}\{H\} = \{A, G, B, C, H\}$ .

- **Consider the dependency  $CG \rightarrow I$   $\mathbf{CG} \rightarrow \mathbf{I}$ :**

- Since  $\{C, G\} \subseteq \{A, G, B, C, H\}$   $\{C, G\} \subseteq \mathbf{A}\{A, G, B, C, H\}$ , we can apply this dependency.
- Update  $\text{result} := \text{result} \cup \{I\} = \{A, G, B, C, H, I\}$   $\text{result} := \text{result}\{I\} = \{A, G, B, C, H, I\}$ .

- **Consider the dependency  $B \rightarrow H$   $\mathbf{B} \rightarrow \mathbf{H}$ :**

- Since  $B \subseteq \{A, G, B, C, H, I\}$   $\mathbf{B} \subseteq \mathbf{A}\{A, G, B, C, H, I\}$ , this dependency holds, but  $HH$  is already in  $\text{result}$ , so no change is made.

second-iteration

- Current  $\text{result} = \{A, G, B, C, H, I\}$   $\text{result} = \{A, G, B, C, H, I\}$
- Recheck all dependencies, but none of them lead to any new attributes being added to  $\text{result}$ , so the closure process terminates.

conclusion

**Conclusion:**

The closure of  $\beta = \{A, G\} = \{A, G\}$  under  $FF$  is  $\{A, G, B, C, H, I\} = \{A, G, B, C, H, I\}$ , which includes all the attributes of  $R(A, B, C, G, H, I)$ .

Therefore,  $\beta = \{A, G\} = \{A, G\}$  satisfies the closure for  $R(A, B, C, G, H, I)$ .

Run the same algorithm for  $F = \{A \rightarrow B, C \rightarrow A, E \rightarrow C, CF \rightarrow A, E \rightarrow F\}$ ,  $\beta = \{E\}$  and  $R(A, B, C, D, E, F)$

ChatGPT said:

Let's run the closure algorithm for  $\beta = \{E\} = \{E\}$  and the functional dependencies  $F = \{A \rightarrow B, C \rightarrow A, E \rightarrow C, CF \rightarrow A, E \rightarrow F\}$  over the relation schema  $R(A, B, C, D, E, F)$ .

steps-to-calculate-the-closure-betaux3b2-1

**Steps to calculate the closure  $\beta^+$ :****1. Initialization:**

- Start with  $\beta = \{E\} = \{E\}$ .
- Initialize  $\text{result} = \{E\}$ .

**2. Apply the algorithm:**

- Loop through each functional dependency  $\gamma \rightarrow \delta$  in  $FF$ .
- If  $\gamma \subseteq \text{result}$ , add  $\delta$  to  $\text{result}$ .

first-iteration-1

- Current  $\text{result} = \{E\}$
- Consider the dependency  $A \rightarrow B$ :

– Since  $A \not\subseteq \{E\}$ , this dependency cannot be applied.

- Consider the dependency  $C \rightarrow A$ :

- Since  $C \subseteq \not\prec E \} C \{ E$ , this dependency cannot be applied.

- **Consider the dependency  $E \rightarrow C \mathbf{E} \rightarrow \mathbf{C}$ :**

- Since  $E \subseteq \{E\}E\{E\}$ , we can apply this dependency.
- Update  $\text{result} := \text{result} \cup \{C\} = \{E, C\}$   $\text{result} := \text{result}\{C\} = \{E, C\}$ .

- **Consider the dependency  $CF \rightarrow A \mathbf{CF} \rightarrow \mathbf{A}$ :**

- Since  $\{C, F\} \subseteq \not\prec E, C \} \{ C, F \} \{ E, C \}$ , this dependency cannot be applied yet.

- **Consider the dependency  $E \rightarrow F \mathbf{E} \rightarrow \mathbf{F}$ :**

- Since  $E \subseteq \{E\}E\{E\}$ , we can apply this dependency.
- Update  $\text{result} := \text{result} \cup \{F\} = \{E, C, F\}$   $\text{result} := \text{result}\{F\} = \{E, C, F\}$ .

second-iteration-1

- Current  $\text{result} = \{E, C, F\}$   $\text{result} = \{E, C, F\}$

- **Consider the dependency  $A \rightarrow B \mathbf{A} \rightarrow \mathbf{B}$ :**

- Since  $A \subseteq \not\prec E, C, F \} A \{ E, C, F \}$ , this dependency cannot be applied.

- **Consider the dependency  $C \rightarrow A \mathbf{C} \rightarrow \mathbf{A}$ :**

- Since  $C \subseteq \{E, C, F\}C\{E, C, F\}$ , we can apply this dependency.
- Update  $\text{result} := \text{result} \cup \{A\} = \{E, C, F, A\}$   $\text{result} := \text{result}\{A\} = \{E, C, F, A\}$ .

- **Consider the dependency  $E \rightarrow C \mathbf{E} \rightarrow \mathbf{C}$ :**

- Since  $CC$  is already in  $\text{result}$ , no change is made.

- **Consider the dependency  $CF \rightarrow A \mathbf{CF} \rightarrow \mathbf{A}$ :**

- Since  $\{C, F\} \subseteq \{E, C, F, A\}$ , we can apply this dependency.
- Update  $\text{result} := \text{result} \cup \{A\} = \{E, C, F, A\}$  (but  $\{A\}$  is already in  $\text{result}$ ).

• **Consider the dependency  $E \rightarrow F$ :**

- Since  $FF$  is already in  $\text{result}$ , no change is made.

third-iteration

• Current  $\text{result} = \{E, C, F, A\}$

• **Consider the dependency  $A \rightarrow B$ :**

- Since  $A \subseteq \{E, C, F, A\}$ , we can apply this dependency.
- Update  $\text{result} := \text{result} \cup \{B\} = \{E, C, F, A, B\}$ .

• **Consider the dependency  $C \rightarrow A$ :**

- Since  $AA$  is already in  $\text{result}$ , no change is made.

• **Consider the dependency  $E \rightarrow C$ :**

- No change as  $CC$  is already in  $\text{result}$ .

• **Consider the dependency  $CF \rightarrow A$ :**

- No change as  $AA$  is already in  $\text{result}$ .

• **Consider the dependency  $E \rightarrow F$ :**

- No change as  $FF$  is already in  $\text{result}$ .

final-iteration

- Current result =  $\{E, C, F, A, B\}$  result =  $\{E, C, F, A, B\}$
- No further dependencies can be applied as all relevant attributes are already in result result.

conclusion-1

### Conclusion:

The closure of  $\beta = \{E\} = \{E\}$  under  $FF$  is  $\{E, C, F, A, B\}$ . Since this does **not include** attribute  $D$ , the closure of  $\beta$  does **not** satisfy the full relation  $R(A, B, C, D, E, F)$ . Thus,  $\beta = \{E\} = \{E\}$  does **not** satisfy closure for  $R(A, B, C, D, E, F)$ .

you-said-2

For relation  $r_1(E, D, C, B, A)$ ; And functional dependencies  $F = \{\{A, B, C\} \Rightarrow \{D\}, \{A, B, C\} \Rightarrow \{D, E\}, \{D\} \Rightarrow \{A, B\}, \{E\} \Rightarrow \{A, C\}\}$ . Use this algorithm to find candidate keys: Given a set of functional dependencies  $F$  over a relation  $r$ : 1. If an attribute  $a$  is not in  $F$ , then  $a$  is in every key 2. If an attribute  $a$  occurs on the right-hand side of a functional dependency in  $F$  but never occurs on the left-hand side, then  $a$  is never in a key 3. If an attribute  $a$  occurs on the left-hand side of a functional dependency in  $F$  but never occurs on the right-hand side, then  $a$  is in every key 4. If an attribute  $a$  occurs on both sides of functional dependencies in  $F$ , then one cannot say anything about such attribute 1. Compute the core (cases 1 and 3 before) 2. If the closure of the core contains all attributes, break (the core is the single candidate key) 3. Let's  $\gamma$  be all attributes that are not exterior (case 4) 1. Compute all possible combinations of  $\gamma$  in ascending order by size (one attribute, two attributes, three attributes, ...) 2. For each combination of size  $x$ : add the core to the current combination, named  $\delta$ , and check  $\delta^+$ ;  $\delta$  is candidate if it fulfills the definition 3. If all combinations of size  $x$  are superkeys; break •  $\beta$  is minimal if there does not exist another  $\beta'$  such that  $\beta' \subset \beta$

ChatGPT said:

step-by-step approach to finding candidate keys for the given relation  $r_1(E, D, C, B, A)$  and the functional dependencies  $FF$

**Step-by-step approach to finding candidate keys for the given relation  $r_1(E, D, C, B, A)$  and the functional dependencies  $FF$ :**

functional-dependencies-ff

1.  $\{A, B, C\} \rightarrow \{D\}$

2.  $\{A, B, C\} \rightarrow \{D, E\} \{A,B,C\} \rightarrow \{D,E\}$
3.  $\{D\} \rightarrow \{A, B\} \{D\} \rightarrow \{A,B\}$
4.  $\{E\} \rightarrow \{A, C\} \{E\} \rightarrow \{A,C\}$

attributes-in-the-relation

- $r_1(E, D, C, B, A) r_1(E, D, C, B, A)$

step-1-compute-the-core

### Step 1: Compute the core

Based on the rules mentioned:

1. **If an attribute occurs on the right-hand side of a functional dependency but never on the left-hand side, it is not in any key:**
  - In this case, **there is no attribute** that is only on the right-hand side and never on the left-hand side.
2. **If an attribute occurs on the left-hand side but never on the right-hand side, it must be in every key:**
  - In this case, **no attribute occurs only on the left-hand side** of functional dependencies.
3. **If an attribute is not part of the functional dependencies (F), it is in every key:**
  - **All attributes** in  $r_1(E, D, C, B, A) r_1(E, D, C, B, A)$  are part of the functional dependencies, so no attribute is automatically in the key due to this rule.
4. **If an attribute appears on both sides of functional dependencies, nothing can be concluded about it directly:**

- Attributes  $AA$ ,  $BB$ ,  $CC$ ,  $DD$ , and  $EE$  appear on both sides of the functional dependencies, so we need to further explore combinations to determine whether they are part of any candidate keys.

**Conclusion from step 1:** There is **no attribute that can be immediately identified** as being in every key (i.e., no "core"). Therefore, we proceed to the next step to find the candidate keys using the closure method.

step-2-identify-attributes-involved-in-functional-dependencies

## Step 2: Identify attributes involved in functional dependencies

We know from step 1 that **no attribute is excluded or automatically included in the key**, so we need to evaluate combinations of attributes from  $\{A, B, C, D, E\}$ .  
 $\gamma$  is the set of all attributes that are not exterior

- Since no attribute was eliminated by the previous rules,  $\gamma = \{A, B, C, D, E\} = \{A, B, C, D, E\}$ .

step-3-compute-all-combinations-of- $\gamma$ -and-their-closures

## Step 3: Compute all combinations of $\gamma$ and their closures

Now, we compute the closure of different combinations of attributes in  $\gamma$ . The goal is to find the minimal combination (candidate key) whose closure includes all attributes in  $\{A, B, C, D, E\}$ .

closure-of-a-set-of-attributes

For a set of attributes  $\beta$ , compute  $\beta^+$  by applying functional dependencies repeatedly until no new attributes can be added.

check-combinations-in-ascending-order-by-size

### 1. Size 3: Try $\{A, B, C\}$ :

- Initial closure:  $\{A, B, C\}$
- Apply  $\{A, B, C\} \rightarrow D$ :  $\{A, B, C, D\}$
- Apply  $\{A, B, C\} \rightarrow D, E$ :  $\{A, B, C, D, E\}$
- Closure of  $\{A, B, C\} = \{A, B, C, D, E\}$
- Since this closure contains all attributes in  $r_1$ ,  $\{A, B, C\}$  is a **candidate key**.



## 2. Size 2: Try smaller subsets (e.g., $\{A, B\}$ , $\{A, C\}$ , etc.):

- Any combination of size 2 will not generate the closure of all attributes because none of these can derive both  $DD$  and  $EE$  without including  $CC$  in the combination. So, there are no candidate keys of size 2.

## 3. Size 4 and 5:

- Since we already found  $\{A, B, C\}$  to be a candidate key, there is no need to explore larger combinations, as they would only lead to superkeys (supersets of the candidate key).

conclusion-2

## Conclusion:

- **Candidate Key:**  $\{A, B, C\}$

This is the minimal set of attributes whose closure contains all attributes of the relation  $r_1(E, D, C, B, A)$ . Therefore,  $\{A, B, C\}$  is the **only candidate key** for this relation and set of functional dependencies.

you-said-3

use this formula: 1. Compute all possible combinations of attributes in ascending order by size (one attribute, two attributes, three attributes, ...) 2. For each combination of size  $x$ : add the core to the current combination, named  $\Delta$ , and check  $\Delta^+$ ;  $\Delta$  is candidate if it fulfills the definition 3. If all combinations of size  $x$  are superkeys; break

ChatGPT said:

Let's apply the formula you provided to find the candidate keys for the relation  $r_1(E, D, C, B, A)$  and the functional dependencies  $FF$ :

## 1. Functional Dependencies:

- $\{A, B, C\} \rightarrow \{D\}$
- $\{A, B, C\} \rightarrow \{D, E\}$
- $\{D\} \rightarrow \{A, B\}$
- $\{E\} \rightarrow \{A, C\}$

## 2. Relation $r_1$ : $(E, D, C, B, A)$

step-1-compute-the-core-1

### Step 1: Compute the Core

- From earlier analysis, **no attribute can immediately be considered part of every key** (no core exists).
- So, we proceed without a core and use all attributes in  $\gamma = \{A, B, C, D, E\}$  as candidates.

step-2-compute-all-possible-combinations-of- $\gamma$ -in-ascending-order-by-size

### Step 2: Compute all possible combinations of $\gamma$ in ascending order by size

We need to try all combinations of attributes from  $\gamma = \{A, B, C, D, E\}$  in ascending order by size (one attribute, two attributes, etc.).

We'll compute the closure for each combination and check if it's a superkey (i.e., if its closure contains all attributes of  $r_1$ ).

combinations-of-size-1

- **Try  $\{A\}$ :**

- $\{A\}^+ = \{A\}$
- No functional dependencies apply to  $\{A\}$ , so the closure remains  $\{A\}$ , which is not a superkey.

- **Try  $\{B\}$ :**

- $\{B\}^+ = \{B\}$
- No functional dependencies apply to  $\{B\}$ , so the closure remains  $\{B\}$ , which is not a superkey.

- **Try  $\{C\}$ :**

- $\{C\}^+ = \{C\}$

- No functional dependencies apply to  $\{C\}\{C\}$ , so the closure remains  $\{C\}\{C\}$ , which is not a superkey.

- **Try  $\{D\}\{D\}$ :**

- $\{D\}^+ = \{D, A, B\}\{D\}^+ = \{D, A, B\}$  (from  $D \rightarrow A, B$  and  $D \rightarrow A, B$ )
- Closure is  $\{D, A, B\}\{D, A, B\}$ , which is not a superkey (it does not contain all attributes).

- **Try  $\{E\}\{E\}$ :**

- $\{E\}^+ = \{E, A, C\}\{E\}^+ = \{E, A, C\}$  (from  $E \rightarrow A, C$  and  $E \rightarrow A, C$ )
- Closure is  $\{E, A, C\}\{E, A, C\}$ , which is not a superkey.

combinations-of-size-2

- **Try  $\{A, B\}\{A, B\}$ :**

- $\{A, B\}^+ = \{A, B\}\{A, B\}^+ = \{A, B\}$
- No functional dependencies apply to  $\{A, B\}\{A, B\}$ , so the closure remains  $\{A, B\}\{A, B\}$ , which is not a superkey.

- **Try  $\{A, C\}\{A, C\}$ :**

- $\{A, C\}^+ = \{A, C\}\{A, C\}^+ = \{A, C\}$
- No functional dependencies apply to  $\{A, C\}\{A, C\}$ , so the closure remains  $\{A, C\}\{A, C\}$ , which is not a superkey.

- **Try  $\{A, D\}\{A, D\}$ :**

- $\{A, D\}^+ = \{A, B, D\}\{A, D\}^+ = \{A, B, D\}$
- Closure is  $\{A, B, D\}\{A, B, D\}$ , which is not a superkey.

- **Try  $\{A, E\}\{A, E\}$ :**

- $\{A, E\}^+ = \{A, C, E\}\{A, E\}^+ = \{A, C, E\}$
- Closure is  $\{A, C, E\}\{A, C, E\}$ , which is not a superkey.
- **Try  $\{B, C\}\{B, C\}$ :**
  - $\{B, C\}^+ = \{B, C\}\{B, C\}^+ = \{B, C\}$
  - No functional dependencies apply to  $\{B, C\}\{B, C\}$ , so the closure remains  $\{B, C\}\{B, C\}$ , which is not a superkey.
- **Try  $\{B, D\}\{B, D\}$ :**
  - $\{B, D\}^+ = \{A, B, D\}\{B, D\}^+ = \{A, B, D\}$  (from  $D \rightarrow A, BD \rightarrow A, B$ )
  - Closure is  $\{A, B, D\}\{A, B, D\}$ , which is not a superkey.
- **Try  $\{B, E\}\{B, E\}$ :**
  - $\{B, E\}^+ = \{A, B, C, E\}\{B, E\}^+ = \{A, B, C, E\}$
  - Closure is  $\{A, B, C, E\}\{A, B, C, E\}$ , which is not a superkey.
- **Try  $\{C, D\}\{C, D\}$ :**
  - $\{C, D\}^+ = \{A, B, C, D\}\{C, D\}^+ = \{A, B, C, D\}$
  - Closure is  $\{A, B, C, D\}\{A, B, C, D\}$ , which is not a superkey.
- **Try  $\{C, E\}\{C, E\}$ :**
  - $\{C, E\}^+ = \{A, B, C, D, E\}\{C, E\}^+ = \{A, B, C, D, E\}$
  - **This closure contains all attributes**, so  $\{C, E\}\{C, E\}$  is a superkey (and potentially a candidate key).
- **Try  $\{D, E\}\{D, E\}$ :**
  - $\{D, E\}^+ = \{A, B, C, D, E\}\{D, E\}^+ = \{A, B, C, D, E\}$
  - **This closure contains all attributes**, so  $\{D, E\}\{D, E\}$  is a superkey (and potentially a candidate key).

combinations-of-size-3

- Since we already found two superkeys of size 2 ( $\{C, E\}$  and  $\{D, E\}$ ), there is no need to proceed to combinations of size 3.

step-3-check-if-cece-and-dede-are-minimal

**Step 3: Check if  $\{C, E\}$  and  $\{D, E\}$  are minimal**

- $\{C, E\}$  is minimal: There is no subset of  $\{C, E\}$  that is a superkey.
- $\{D, E\}$  is minimal: There is no subset of  $\{D, E\}$  that is a superkey.

conclusion-3

## Conclusion

The **candidate keys** for the relation  $r_1(E, D, C, B, A)$  and functional dependencies  $FF$  are:

- $\{C, E\}$
- $\{D, E\}$

you-said-4

use this formula for closure: Closure of beta under F (beta+) result := beta Do For each  $\alpha \rightarrow \delta$  in F: If ( $\alpha \subseteq \text{result}$ ): result := result  $\cup \delta$  While (result does not change) Return result

ChatGPT said:

Let's use the closure formula you provided to compute the closures of different attribute sets for finding candidate keys. Here's the step-by-step process:

given-relation

## Given Relation:

$r_1(E, D, C, B, A)$

functional-dependencies

## Functional Dependencies:

1.  $\{A, B, C\} \rightarrow D$

2.  $\{A, B, C\} \rightarrow D, E \{A, B, C\} \rightarrow D, E$
3.  $\{D\} \rightarrow A, B \{D\} \rightarrow A, B$
4.  $\{E\} \rightarrow A, C \{E\} \rightarrow A, C$

step-by-step-explanation-of-the-closure-formula

### Step-by-step explanation of the closure formula:

We will compute the closure of different sets using the formula:

1. **Initialize** the result set to the initial attribute set  $\beta$ .
2. **Iterate** over the functional dependencies. If the left-hand side (LHS) of a functional dependency is a subset of the current result set, **add** the right-hand side (RHS) to the result set.
3. **Repeat** until no further attributes can be added.
4. **Return** the final result (closure of  $\beta$ ).

step-1-compute-closures-of-various-sets

### Step 1: Compute closures of various sets

closure-of-aa

- Start with  $result = \{A\}$  result={A}
- Iterate over each dependency:
  - $\{A, B, C\} \rightarrow D \{A, B, C\} \rightarrow D$ : LHS is not a subset of  $\{A\}\{A\}$
  - $\{A, B, C\} \rightarrow D, E \{A, B, C\} \rightarrow D, E$ : LHS is not a subset of  $\{A\}\{A\}$
  - $\{D\} \rightarrow A, B \{D\} \rightarrow A, B$ : LHS is not a subset of  $\{A\}\{A\}$
  - $\{E\} \rightarrow A, C \{E\} \rightarrow A, C$ : LHS is not a subset of  $\{A\}\{A\}$
- **Result:** Closure of  $\{A\} = \{A\}\{A\} = \{A\}$

closure-of-bb

- Start with  $result = \{B\}$  result={B}
- Iterate over each dependency:
  - $\{A, B, C\} \rightarrow D$  {A,B,C}→D: LHS is not a subset of  $\{B\}\{B\}$
  - $\{A, B, C\} \rightarrow D, E$  {A,B,C}→D,E: LHS is not a subset of  $\{B\}\{B\}$
  - $\{D\} \rightarrow A, B$  {D}→A,B: LHS is not a subset of  $\{B\}\{B\}$
  - $\{E\} \rightarrow A, C$  {E}→A,C: LHS is not a subset of  $\{B\}\{B\}$
- **Result:** Closure of  $\{B\} = \{B\}\{B\}=\{B\}$

closure-of-cc

- Start with  $result = \{C\}$  result={C}
- Iterate over each dependency:
  - $\{A, B, C\} \rightarrow D$  {A,B,C}→D: LHS is not a subset of  $\{C\}\{C\}$
  - $\{A, B, C\} \rightarrow D, E$  {A,B,C}→D,E: LHS is not a subset of  $\{C\}\{C\}$
  - $\{D\} \rightarrow A, B$  {D}→A,B: LHS is not a subset of  $\{C\}\{C\}$
  - $\{E\} \rightarrow A, C$  {E}→A,C: LHS is not a subset of  $\{C\}\{C\}$
- **Result:** Closure of  $\{C\} = \{C\}\{C\}=\{C\}$

closure-of-dd

- Start with  $result = \{D\}$  result={D}
- Iterate over each dependency:
  - $\{A, B, C\} \rightarrow D$  {A,B,C}→D: LHS is not a subset of  $\{D\}\{D\}$
  - $\{A, B, C\} \rightarrow D, E$  {A,B,C}→D,E: LHS is not a subset of  $\{D\}\{D\}$
  - $\{D\} \rightarrow A, B$  {D}→A,B: LHS  $\{D\}\{D\}$  is a subset of  $result$  result, so  $result = \{D, A, B\}$  result={D,A,B}

- $\{E\} \rightarrow A, C \{E\} \rightarrow A, C$ : LHS is not a subset of  $\{D, A, B\}$

- **Result:** Closure of  $\{D\} = \{D, A, B\}$

closure-of-ee

- Start with  $result = \{E\}$

- Iterate over each dependency:

- $\{A, B, C\} \rightarrow D \{A, B, C\} \rightarrow D$ : LHS is not a subset of  $\{E\}$
- $\{A, B, C\} \rightarrow D, E \{A, B, C\} \rightarrow D, E$ : LHS is not a subset of  $\{E\}$
- $\{D\} \rightarrow A, B \{D\} \rightarrow A, B$ : LHS is not a subset of  $\{E\}$
- $\{E\} \rightarrow A, C \{E\} \rightarrow A, C$ : LHS  $\{E\}$  is a subset of  $result$ , so  $result = \{E, A, C\}$

- **Result:** Closure of  $\{E\} = \{E, A, C\}$

closure-of-abcabc

- Start with  $result = \{A, B, C\}$

- Iterate over each dependency:

- $\{A, B, C\} \rightarrow D \{A, B, C\} \rightarrow D$ : LHS  $\{A, B, C\}$  is a subset of  $result$ , so  $result = \{A, B, C, D\}$
- $\{A, B, C\} \rightarrow D, E \{A, B, C\} \rightarrow D, E$ : LHS  $\{A, B, C\}$  is a subset of  $result$ , so  $result = \{A, B, C, D, E\}$
- $\{D\} \rightarrow A, B \{D\} \rightarrow A, B$ : LHS  $\{D\}$  is a subset of  $result$ , but all attributes  $\{A, B\}$  are already in the result.
- $\{E\} \rightarrow A, C \{E\} \rightarrow A, C$ : LHS  $\{E\}$  is a subset of  $result$ , but all attributes  $\{A, C\}$  are already in the result.

- **Result:** Closure of  $\{A, B, C\} = \{A, B, C, D, E\}$

step-2-identify-candidate-keys



## Step 2: Identify Candidate Keys

Now that we have computed the closures for different sets, we can identify the candidate keys by checking if the closure of a set contains all attributes of the relation  $r_1(E, D, C, B, A)$ .

- **Closure of  $\{A, B, C\} = \{A, B, C, D, E\}$** : Contains all attributes, so  $\{A, B, C\}$  is a **superkey** and a **candidate key**.
- **Closure of  $\{C, E\} = \{C, E, A\}$** : Does not contain all attributes.
- **Closure of  $\{D, E\} = \{D, E, A, B, C\}$** : Contains all attributes, so  $\{D, E\}$  is also a **superkey** and a **candidate key**.

final-result

## Final Result:

- **Candidate keys for  $r_1(E, D, C, B, A)$** :
  - $\{A, B, C\}$
  - $\{D, E\}$

you-said-5

For relation  $r_1(A, B, C, D, E)$ ; And functional dependencies  $F = \{C \rightarrow A, C \rightarrow D, AE \rightarrow D, BC \rightarrow E, BDE \rightarrow A\}$ . Use this algorithm to find candidate keys: Given a set of functional dependencies  $F$  over a relation  $r$ : 1. If an attribute  $a$  is not in  $F$ , then  $a$  is in every key 2. If an attribute  $a$  occurs on the right-hand side of a functional dependency in  $F$  but never occurs on the left-hand side, then  $a$  is never in a key 3. If an attribute  $a$  occurs on the left-hand side of a functional dependency in  $F$  but never occurs on the right-hand side, then  $a$  is in every key 4. If an attribute  $a$  occurs on both sides of functional dependencies in  $F$ , then one cannot say anything about such attribute 1. Compute the core (cases 1 and 3 before) 2. If the closure of the core contains all attributes, break (the core is the single candidate key) 3. Let's  $\gamma$  be all attributes that are not exterior (case 4) 1. Compute all possible combinations of  $\gamma$  in ascending order by size (one attribute, two attributes, three attributes, ...) 2. For each combination of size  $x$ : add the core to the current combination, named  $\delta$ , and check  $\delta^+$ ;  $\delta$  is candidate if it fulfills the definition 3. If all combinations of size  $x$  are superkeys; break •  $\beta$  is minimal if there does not exist another  $\beta'$  such that  $\beta' \subset \beta$

ChatGPT said:

To find the candidate keys for relation  $r_1(A, B, C, D, E)$  with functional dependencies  $F = \{C \rightarrow A, C \rightarrow D, AE \rightarrow D, BC \rightarrow E, BDE \rightarrow A\}$  we will follow the algorithm provided. Let's go step by step.

step-1-analyze-each-attribute-based-on-the-rules

### Step 1: Analyze Each Attribute Based on the Rules

rule-1-if-an-attribute-is-not-in-any-functional-dependency-ff-then-it-must-be-in-every-key.

- All attributes  $A, B, C, D, E$  are part of at least one functional dependency. So no attribute is required in every key just by this rule.

rule-2-if-an-attribute-appears-only-on-the-right-hand-side-rhs-of-a-functional-dependency-and-never-on-the-left-hand-side-lhs-it-is-never-in-a-key.

- Attributes on the RHS:
  - $A$  appears on the RHS of  $C \rightarrow A$  and  $BDE \rightarrow A$ .
  - $D$  appears on the RHS of  $C \rightarrow D$  and  $AE \rightarrow D$ .
  - $E$  appears on the RHS of  $BC \rightarrow E$ .
- **Conclusion:** All attributes appear on the LHS at least once, so none are excluded by Rule 2.

rule-3-if-an-attribute-appears-only-on-the-left-hand-side-lhs-of-a-functional-dependency-and-never-on-the-right-hand-side-rhs-it-must-be-in-every-key.

- All attributes appear on both sides, so this rule does not apply.

rule-4-if-an-attribute-appears-on-both-the-left-hand-and-right-hand-sides-of-functional-dependencies-we-cannot-say-anything-definitive-about-whether-it-is-in-a-key.

- Attributes  $A, C, D, E$  appear on both sides, so we cannot infer their presence or absence in candidate keys by this rule.

step-2-compute-the-core-minimal-set

## Step 2: Compute the Core (Minimal Set)

To compute the core, we first apply Rules 1 and 3, but neither rule gives us any fixed attributes to include in every key.

Thus, the **core** is initially **empty**.

step-3-identify-gamma-uncertain-attributes

## Step 3: Identify Gamma (Uncertain Attributes)

Since none of the attributes were directly included in the core, all the attributes  $A, B, C, D, E$  are uncertain. We will need to check combinations of these attributes to determine candidate keys.

step-4-compute-all-possible-combinations-of-gamma-in-ascending-order-by-size

## Step 4: Compute All Possible Combinations of Gamma in Ascending Order by Size

We'll compute the closures for different sets in ascending order by size, starting with individual attributes, then pairs, etc.

closure-of-cc-1

- Start with  $result = \{C\}$
- Apply functional dependencies:
  - $C \rightarrow A$ , so  $result = \{C, A\}$
  - $C \rightarrow D$ , so  $result = \{C, A, D\}$
- The closure of  $\{C\} = \{C, A, D\}$ , which does not include all attributes. So,  $\{C\}$  is **not a candidate key**.

closure-of-bb-1

- Start with  $result = \{B\}$
- No functional dependency applies to  $\{B\}$ , so  $\{B\}$  cannot generate any new attributes.
- The closure of  $\{B\} = \{B\}$ , which does not include all attributes. So,  $\{B\}$  is **not a candidate key**.

closure-of-aa-1

- Start with  $result = \{A\}$  result= $\{A\}$
- No functional dependency applies to  $\{A\}\{A\}$ , so  $\{A\}\{A\}$  cannot generate any new attributes.
- The closure of  $\{A\} = \{A\}\{A\}=\{A\}$ , which does not include all attributes. So,  $\{A\}\{A\}$  is **not a candidate key**.

closure-of-dd-1

- Start with  $result = \{D\}$  result= $\{D\}$
- $D \rightarrow A, B$   $D \rightarrow A, B$ , so  $result = \{D, A, B\}$  result= $\{D, A, B\}$
- No more functional dependencies apply.
- The closure of  $\{D\} = \{D, A, B\}\{D\}=\{D, A, B\}$ , which does not include all attributes. So,  $\{D\}\{D\}$  is **not a candidate key**.

closure-of-ee-1

- Start with  $result = \{E\}$  result= $\{E\}$
- $E \rightarrow A, C$   $E \rightarrow A, C$ , so  $result = \{E, A, C\}$  result= $\{E, A, C\}$
- No more functional dependencies apply.
- The closure of  $\{E\} = \{E, A, C\}\{E\}=\{E, A, C\}$ , which does not include all attributes. So,  $\{E\}\{E\}$  is **not a candidate key**.

step-5-check-combinations-of-size-2

## Step 5: Check Combinations of Size 2

Now, we check pairs of attributes.

closure-of-bcbc

- Start with  $result = \{B, C\}$  result= $\{B, C\}$
- $C \rightarrow A$   $C \rightarrow A$ , so  $result = \{B, C, A\}$  result= $\{B, C, A\}$

- $C \rightarrow D$   $C \rightarrow D$ , so  $result = \{B, C, A, D\}$   $result = \{B, C, A, D\}$
- $BC \rightarrow E$   $BC \rightarrow E$ , so  $result = \{B, C, A, D, E\}$   $result = \{B, C, A, D, E\}$
- The closure of  $\{B, C\} = \{A, B, C, D, E\}$   $\{B, C\} = \{A, B, C, D, E\}$ , which includes all attributes. Therefore,  $\{B, C\}$  is a **candidate key**.

closure-of-bdbd

- Start with  $result = \{B, D\}$   $result = \{B, D\}$
- $D \rightarrow A$   $D \rightarrow A$ , so  $result = \{B, D, A\}$   $result = \{B, D, A\}$
- No more functional dependencies apply.
- The closure of  $\{B, D\} = \{B, D, A\}$   $\{B, D\} = \{B, D, A\}$ , which does not include all attributes. So,  $\{B, D\}$  is **not a candidate key**.

closure-of-bebe

- Start with  $result = \{B, E\}$   $result = \{B, E\}$
- $E \rightarrow A$   $E \rightarrow A$ , so  $result = \{B, E, A\}$   $result = \{B, E, A, C\}$
- $BC \rightarrow E$   $BC \rightarrow E$ , but  $EE$  is already in the result.
- The closure of  $\{B, E\} = \{B, E, A, C\}$   $\{B, E\} = \{B, E, A, C\}$ , which does not include all attributes. So,  $\{B, E\}$  is **not a candidate key**.

conclusion-4

## Conclusion:

- The only **candidate key** we found is  $\{B, C\}$ .

you-said-6

define Rule 3 again for these functional dependencies

2/2

ChatGPT said:

For the given functional dependencies  $F = \{C \rightarrow A, C \rightarrow D, AE \rightarrow D, BC \rightarrow E, BDE \rightarrow A\}$   $F = \{C \rightarrow A, C \rightarrow D, AE \rightarrow D, BC \rightarrow E, BDE \rightarrow A\}$  let's clarify **Rule 3** again in this context.

rule-3

**Rule 3:Rule 3:**

If an attribute appears **only on the left-hand side (LHS)** of a functional dependency and **never on the right-hand side (RHS)**, then that attribute **must be in every key**.

This rule implies that if an attribute is necessary to determine other attributes but is not determined by any other attribute, it is essential for identifying the relation and must be part of any candidate key.

analysis-of-attributes-based-on-rule-3

**Analysis of Attributes Based on Rule 3:**

- **Attribute CC:**

- It appears on the left-hand side in:  $C \rightarrow AC \rightarrow A$ ,  $C \rightarrow DC \rightarrow D$ , and  $BC \rightarrow EBC \rightarrow E$ .
- It never appears on the right-hand side.
- **Conclusion: CC must be in every candidate key.**

- **Attribute BB:**

- It appears on the left-hand side in:  $BC \rightarrow EBC \rightarrow E$ ,  $BDE \rightarrow ABDE \rightarrow A$ .
- It never appears on the right-hand side.
- **Conclusion: BB must be in every candidate key.**

- **Attribute AA:**

- It appears on the right-hand side in:  $C \rightarrow AC \rightarrow A$ ,  $BDE \rightarrow ABDE \rightarrow A$ .
- It also appears on the left-hand side in:  $AE \rightarrow DAE \rightarrow D$ .
- Since it appears on both sides, **Rule 3 does not apply** to AA.

- **Attribute DD:**

- It appears on the right-hand side in:  $C \rightarrow DC \rightarrow D$ ,  $AE \rightarrow DAE \rightarrow D$ .
- It also appears on the left-hand side in:  $BDE \rightarrow ABDE \rightarrow A$ .
- Since it appears on both sides, **Rule 3 does not apply** to DD.

- **Attribute EE:**

- It appears on the right-hand side in:  $BC \rightarrow E$ ,  $BC \rightarrow E$ .
- It also appears on the left-hand side in:  $AE \rightarrow D$ ,  $AE \rightarrow D$ ,  $BDE \rightarrow A$ ,  $BDE \rightarrow A$ .
- Since it appears on both sides, **Rule 3 does not apply** to  $EE$ .

conclusion-5

### Conclusion:

From **Rule 3**, we conclude that both  $CC$  and  $BB$  must be part of every candidate key.

you-said-7

$r(A, B, C, D, E, G)$  FD =  $\{A \rightarrow B, A \rightarrow C, CD \rightarrow E, CD \rightarrow G, B \rightarrow E\}$  Use this algorithm to find candidate keys: Given a set of functional dependencies  $F$  over a relation  $r$ : 1. If an attribute  $a$  is not in  $F$ , then  $a$  is in every key 2. If an attribute  $a$  occurs on the right-hand side of a functional dependency in  $F$  but never occurs on the left-hand side, then  $a$  is never in a key 3. If an attribute  $a$  occurs on the left-hand side of a functional dependency in  $F$  but never occurs on the right-hand side, then  $a$  is in every key { 1. Compute the core (cases 1 and 3 before) 2. If the closure of the core contains all attributes, break (the core is the single candidate key) } 4. If an attribute  $a$  occurs on both sides of functional dependencies in  $F$ , then { 1. Compute all possible combinations of  $\gamma$  in ascending order by size (one attribute, two attributes, three attributes, ...) 2. For each combination of size  $x$ : add the core to the current combination, named  $\delta$ , and check  $\delta^+$ ;  $\delta$  is candidate if it fulfills the definition 3. If all combinations of size  $x$  are superkeys; break }

2/2

ChatGPT said:

To find the candidate keys for relation  $r(A, B, C, D, E, G)$  with functional dependencies  $F = \{A \rightarrow B, A \rightarrow C, CD \rightarrow E, CD \rightarrow G, B \rightarrow E\}$  we will follow the steps in the algorithm.

step-1-analyze-each-attribute-based-on-the-rules-1

### Step 1: Analyze Each Attribute Based on the Rules

rule-1-if-an-attribute-is-not-in-any-functional-dependency-ff-then-it-must-be-in-every-key.-

1

- All attributes  $A, B, C, D, E, G$  are part of at least one functional dependency, so none are required in every key based on this rule.

rule-2-if-an-attribute-appears-only-on-the-right-hand-side-rhs-of-a-functional-dependency-and-never-on-the-left-hand-side-lhs-it-is-never-in-a-key.-1

- Attributes on the RHS:

- $BB$  appears on the RHS of  $A \rightarrow BA \rightarrow B$ .
- $CC$  appears on the RHS of  $A \rightarrow CA \rightarrow C$ .
- $EE$  appears on the RHS of  $CD \rightarrow ECD \rightarrow E$  and  $B \rightarrow EB \rightarrow E$ .
- $GG$  appears on the RHS of  $CD \rightarrow GCD \rightarrow G$ .

- **Conclusion:** Attributes  $BB$ ,  $CC$ ,  $EE$ , and  $GG$  appear only on the RHS. Therefore,  $B, C, E, GB, C, E, G$  are **never in a key**.

rule-3-if-an-attribute-appears-only-on-the-left-hand-side-lhs-of-a-functional-dependency-and-never-on-the-right-hand-side-rhs-it-must-be-in-every-key.-1

- The attribute  $AA$  appears on the LHS in:

- $A \rightarrow BA \rightarrow B$
- $A \rightarrow CA \rightarrow C$
- It does not appear on the RHS.

Therefore,  **$AA$  must be in every key**.

- The attributes  $C, DC, D$  appear on the LHS in  $CD \rightarrow ECD \rightarrow E$ ,  $CD \rightarrow GCD \rightarrow G$ , and are on the RHS in some cases, so Rule 3 doesn't apply to them.

step-2-compute-the-core-minimal-set-1

## Step 2: Compute the Core (Minimal Set)

From Rule 3, we know that  $AA$  is in every key. So the **core** is  $\{A\}\{A\}$ .

step-3-compute-the-closure-of-the-core

## Step 3: Compute the Closure of the Core

Let's compute the closure of the core  $\{A\}\{A\}$  under the functional dependencies.  
closure-of-aa-under-ff

1. Start with  $result = \{A\}$  result={A}.



2. Apply functional dependencies:

- $A \rightarrow B$   $A \rightarrow B$ , so  $result = \{A, B\}$   $result = \{A, B\}$ .
- $A \rightarrow C$   $A \rightarrow C$ , so  $result = \{A, B, C\}$   $result = \{A, B, C\}$ .

3. No other functional dependencies apply to  $\{A, B, C\}$   $\{A, B, C\}$ .

The closure of  $\{A\}$   $\{A\}$  is  $\{A, B, C\}$   $\{A, B, C\}$ , which does **not** include all attributes  $\{A, B, C, D, E, G\}$   $\{A, B, C, D, E, G\}$ . Therefore,  $\{A\}$   $\{A\}$  alone is **not a candidate key**.

step-4-identify-gamma-uncertain-attributes

#### Step 4: Identify Gamma (Uncertain Attributes)

The remaining attributes that are not in the core (i.e., not included by Rule 2 or Rule 3) are  $DD$ .

We now need to check combinations involving  $AA$  and  $DD$ .

step-5-compute-the-closure-of-adad

#### Step 5: Compute the Closure of $\{A, D\}$ $\{A, D\}$ Step 5: Compute the Closure of $\{A, D\}$ $\{A, D\}$

closure-of-adad-under-ff

1. Start with  $result = \{A, D\}$   $result = \{A, D\}$ .

2. Apply functional dependencies:

- $A \rightarrow B$   $A \rightarrow B$ , so  $result = \{A, B, D\}$   $result = \{A, B, D\}$ .
- $A \rightarrow C$   $A \rightarrow C$ , so  $result = \{A, B, C, D\}$   $result = \{A, B, C, D\}$ .
- $CD \rightarrow E$   $CD \rightarrow E$ , so  $result = \{A, B, C, D, E\}$   $result = \{A, B, C, D, E\}$ .
- $CD \rightarrow G$   $CD \rightarrow G$ , so  $result = \{A, B, C, D, E, G\}$   $result = \{A, B, C, D, E, G\}$ .

The closure of  $\{A, D\} = \{A, B, C, D, E, G\}$   $\{A, D\} = \{A, B, C, D, E, G\}$ , which includes **all** attributes. Therefore,  $\{A, D\}$   $\{A, D\}$  is a **candidate key**.

step-6-check-for-minimality

### Step 6: Check for Minimality

- The set  $\{A, D\}$  is minimal because:
  - Removing  $A$  leaves  $\{D\}$ , whose closure does not include all attributes.
  - Removing  $D$  leaves  $\{A\}$ , whose closure does not include all attributes either.

conclusion-6

### Conclusion:

The **candidate key** for the relation  $r(A, B, C, D, E, G)$  with functional dependencies  $F$  is  $\{A, D\}$ .

you-said-8

write down the functional dependencies

ChatGPT said:

Here are the functional dependencies you provided for the relation  $r(A, B, C, D, E, G)$ :

1.  $A \rightarrow B$
2.  $A \rightarrow C$
3.  $CD \rightarrow E$
4.  $CD \rightarrow G$
5.  $B \rightarrow E$

These are the functional dependencies we used to find the candidate key.

you-said-9

divide them in 4 cases

ChatGPT said:

Let's divide the functional dependencies into four cases based on the roles of attributes (left-hand side and right-hand side) according to the rules you've specified:

case-1-attributes-that-are-not-in-any-functional-dependency

### Case 1: Attributes that are not in any functional dependency

- These attributes do not appear in any functional dependency, either on the left-hand side (LHS) or right-hand side (RHS), and **must be in every key**.

In our set of functional dependencies:

- **No attributes** are missing from the functional dependencies, so **no attribute** falls into this category.

case-2-attributes-that-appear-only-on-the-right-hand-side-rhs-of-a-functional-dependency

### Case 2: Attributes that appear only on the right-hand side (RHS) of a functional dependency

- Attributes that appear only on the RHS of a functional dependency and **never on the LHS** cannot be part of a key.

From the given functional dependencies:

- $BB$  appears on the RHS of  $A \rightarrow B$  and  $B \rightarrow E$ .
- $CC$  appears on the RHS of  $A \rightarrow C$ .
- $EE$  appears on the RHS of  $CD \rightarrow E$  and  $B \rightarrow E$ .
- $GG$  appears on the RHS of  $CD \rightarrow G$ .

**Conclusion:** The attributes  $BB$ ,  $CC$ ,  $EE$ , and  $GG$  cannot be part of a candidate key.

case-3-attributes-that-appear-only-on-the-left-hand-side-lhs-of-a-functional-dependency

### Case 3: Attributes that appear only on the left-hand side (LHS) of a functional dependency

- Attributes that appear only on the LHS of functional dependencies and **never on the RHS** must be part of every key.

From the given functional dependencies:

- **AA** appears only on the LHS in  $A \rightarrow B$  and  $A \rightarrow C$ .

– Therefore, **AA must be in every candidate key**.

case-4-attributes-that-appear-on-both-sides-of-functional-dependencies

#### Case 4: Attributes that appear on both sides of functional dependencies

- Attributes that appear on both the LHS and RHS are **uncertain** (neither guaranteed nor excluded from being part of a key).

From the given functional dependencies:

- **DD** appears on the LHS of  $CD \rightarrow E$  and  $CD \rightarrow G$ , and on the RHS in no dependency.

Thus, **DD** is an uncertain attribute and needs to be considered when checking combinations for candidate keys.

summary-of-divided-functional-dependencies

#### Summary of Divided Functional Dependencies:

- **Case 1** (Not in any FD): None.
- **Case 2** (Only on RHS): **BB**, **CC**, **EE**, **GG**.
- **Case 3** (Only on LHS): **AA**.
- **Case 4** (Appears on both sides): **DD**.

You've hit the Free plan limit for GPT-4o.

Responses will use another model until your limit resets after 11:08 PM.

Get Plus  
prompt-textarea

composer-background