0.32 Candidate Keys Potential Issues

you-said

rename attributes in this relation and functional dependencies to meaningful words r2 (A,B ,C, D,E ,G ,H ,K) ; F = {{A, B,H} -> { C}, {A} -> {D,E }, {B,G,H} -> {K}, {K} -> {A,D,H}, {B,H} -> {G, E }.

ChatGPT said:

To rename the attributes in this relation and make them more meaningful, you can assign names that reflect their potential roles. Here's an example of how the relation `r2(A, B, C, D, E, G, H, K)` might be renamed with meaningful attribute names, and the functional dependencies (FDs) updated accordingly:

renamed-attributes

Renamed attributes:

- `A` -> `EmployeeID`
- `B` -> `Department`
- `C` -> `Salary`
- `D` -> `HireDate`
- `E` -> `Position`
- `G` -> `Manager`
- `H` -> `Location`
- `K` -> `Project`

updated-functional-dependencies-fds

Updated functional dependencies (FDs):

1. `{EmployeeID, Department, Location} -> {Salary}`
   (The combination of an employee's ID, department, and location determines their salary.)

2. `{EmployeeID} -> {HireDate, Position}`
   (An employee's ID determines their hire date and position.)
3. `{Department, Manager, Location} -> {Project}`
   (The combination of department, manager, and location determines the project an employee is assigned to.)
4. `{Project} -> {EmployeeID, HireDate, Location}`
   (A project determines the employee, their hire date, and location.)
5. `{Department, Location} -> {Manager, Position}`
   (The combination of department and location determines the manager and the employee's position.)

By giving meaningful names, it's easier to understand what each functional dependency represents and how the attributes relate to one another in a real-world scenario. You can adjust the names to match your specific context.

you-said-1

apply this algorithm on the above set of functional dependencies: functional_dependencies of type for ith functional dependency. Initialize case_1, case_2, case_3, case_4 as empty sets. For x in attributes: if x \notin \alpha_i \& x \notin \beta_j \forall i,j: case_1.add(x) elif x \notin \alpha_i \& x \in \beta_j \forall i: case_2.add(x) elif x \in \alpha_i \& \ x notin \beta_j \forall j: case_3.add(x) else: case_4.add(x) return attributes in case_1, case_2, case_3, case_4

ChatGPT said:

To apply the algorithm you provided to the functional dependencies, we'll break it down step by step.

initial-setup

Initial Setup:

We are working with the following renamed functional dependencies:

1. `{EmployeeID, Department, Location} -> {Salary}`
2. `{EmployeeID} -> {HireDate, Position}`
3. `{Department, Manager, Location} -> {Project}`
4. `{Project} -> {EmployeeID, HireDate, Location}`
5. `{Department, Location} -> {Manager, Position}`

step-by-step-breakdown-of-the-algorithm

Step-by-Step Breakdown of the Algorithm:

- **Initialize empty sets for case_1, case_2, case_3, and case_4**.
- The set of all attributes is:

$$\{\text{EmployeeID}, \text{Department}, \text{Salary}, \text{HireDate}, \text{Position}, \text{Manager}, \text{Location}, \text{Project}\}$$

{EmployeeID,Department,Salary,HireDate,Position,Manager,Location,Project}

- Now, we'll check each attribute and determine which case it falls into by iterating through all the attributes (`x`).

case-1-x-notin-alpha$_i - and - x - notin - beta_j - for - all - i - and - j$

Case 1: `x \notin \alpha_i` and `x \notin \beta_j` for all `i` and `j`Case 1: x \notin \alpha_i and x \notin \beta_j for all i and j

This means the attribute is **neither on the left side** (i.e., in α_ii) **nor on the right side** (i.e., in β_jj) in any functional dependency.

After reviewing all the FDs, there are no such attributes in this case.
\text{case_1} = \emptyset
case-2-x-notin-alpha$_i - and - x - in - beta_j - for - all - i$

Case 2: `x \notin \alpha_i` and `x \in \beta_j` for all `i`Case 2: x \notin \alpha_i and x \in \beta_j for all i

This means the attribute is **only on the right-hand side** in some functional dependencies and never appears on the left-hand side of any dependency.

- **Salary**, **HireDate**, and **Project** are only found on the right-hand side of the FDs.

\text{case_2} = \{\text{Salary}, \text{HireDate}, \text{Project}\}
case-3-x-in-alpha$_i - and - x - notin - beta_j - for - all - j$

Case 3: `x \in \alpha_i` and `x \notin \beta_j` for all `j`Case 3: x \in \alpha_i and x \notin \beta_j for all j

This means the attribute is **only on the left-hand side** in some functional dependencies and never appears on the right-hand side of any dependency.

- **EmployeeID** and **Department** are only found on the left-hand side of the FDs.

\text{case_3} = \{\text{EmployeeID}, \text{Department}\}
case-4-all-other-attributes

Case 4: All other attributes

This case includes attributes that appear on both the left-hand side and the right-hand side of the functional dependencies.

- **Location**, **Position**, and **Manager** appear on both the left-hand side and the right-hand side of some FDs.

\text{case_4} = \{\text{Location}, \text{Position}, \text{Manager}\}
final-result

Final Result:

- **case_1**:
- **case_2**: $\{\text{Salary}, \text{HireDate}, \text{Project}\}$\{Salary,HireDate,Project\}
- **case_3**: $\{\text{EmployeeID}, \text{Department}\}$\{EmployeeID,Department\}
- **case_4**: $\{\text{Location}, \text{Position}, \text{Manager}\}$\{Location,Position,Manager\}

These sets represent how each attribute is classified based on its presence in the functional dependencies.
you-said-2

explain how position and project are categorized
ChatGPT said:
Let's explain why **Position** and **Project** are categorized into their respective cases in the algorithm.
position

1. Position:1. Position:

To determine where **Position** falls, we examine whether it appears on the left-hand side (LHS, α_ii) or the right-hand side (RHS, β_jj) of the functional dependencies.

- From FD 2: `{EmployeeID} -> {HireDate, Position}`
  Position is on the **RHS**.
- From FD 5: `{Department, Location} -> {Manager, Position}`
  Position is on the **RHS**.

So far, **Position** has only appeared on the right side.

However, no functional dependency lists **Position** on the left-hand side (i.e., determining other attributes). Therefore, **Position** appears **only on the right side** of some FDs and is never on the left side. Thus, it should belong to **case 2** (attributes that are on the RHS but not on the LHS).

---

project

2. Project:2. Project:

Now, let's evaluate where **Project** fits in:

- From FD 3: {`Department, Manager, Location`} `->` {`Project`}
  Project is on the **RHS**.
- From FD 4: {`Project`} `->` {`EmployeeID, HireDate, Location`}
  Project is on the **LHS**.

Here, **Project** appears on both the **LHS** (determining other attributes in FD 4) and the **RHS** (being determined by other attributes in FD 3).

Therefore, **Project** is categorized into **case 4** (attributes that appear on both the LHS and RHS).

---

summary

Summary:

- **Position** belongs to **case 2** because it **only appears on the RHS** of functional dependencies and never on the LHS.
- **Project** belongs to **case 4** because it **appears on both the LHS and RHS** of different functional dependencies.

you-said-3

what are the updated cases?

ChatGPT said:

Let's revisit the updated cases after clarifying the categorization of **Position** and **Project**:

functional-dependencies-recap

Functional Dependencies Recap:

1. `{EmployeeID, Department, Location} -> {Salary}`
2. `{EmployeeID} -> {HireDate, Position}`
3. `{Department, Manager, Location} -> {Project}`
4. `{Project} -> {EmployeeID, HireDate, Location}`
5. `{Department, Location} -> {Manager, Position}`

case-definitions

Case Definitions:

- **Case 1**: Attributes that are **neither on the left-hand side (LHS)** nor **on the right-hand side (RHS)** of any functional dependency.
- **Case 2**: Attributes that are **only on the RHS** of functional dependencies and never on the LHS.
- **Case 3**: Attributes that are **only on the LHS** of functional dependencies and never on the RHS.
- **Case 4**: Attributes that appear **on both the LHS and RHS** of functional dependencies.

case-by-case-evaluation

Case-by-Case Evaluation:

1. **Case 1**: Attributes that are **neither on the LHS nor the RHS** of any functional dependency.

  - There are no such attributes because all attributes appear either on the LHS or RHS in the given functional dependencies.

  Updated case_1:

2. **Case 2**: Attributes that are **only on the RHS** of functional dependencies.

  - After reviewing, **Salary**, **HireDate**, and **Position** appear only on the right-hand side of functional dependencies.

  Updated case_2: $\{\text{Salary}, \text{HireDate}, \text{Position}\}$\{Salary,HireDate,Position\}

3. **Case 3**: Attributes that are **only on the LHS** of functional dependencies.

  - **EmployeeID** and **Department** only appear on the left-hand side and never on the right.

  Updated case_3: $\{\text{EmployeeID}, \text{Department}\}$\{EmployeeID,Department\}

4. **Case 4**: Attributes that appear **on both the LHS and RHS** of functional dependencies.

  - **Project**, **Manager**, and **Location** appear on both the left and right sides of the FDs.

  Updated case_4: $\{\text{Project}, \text{Manager}, \text{Location}\}$\{Project,Manager,Location\}

final-updated-cases

Final Updated Cases:

- **case_1**:
- **case_2**: $\{\text{Salary}, \text{HireDate}, \text{Position}\}$\{Salary,HireDate,Position\}
- **case_3**: $\{\text{EmployeeID}, \text{Department}\}$\{EmployeeID,Department\}
- **case_4**: $\{\text{Project}, \text{Manager}, \text{Location}\}$\{Project,Manager,Location\}

These are the final updated cases after properly categorizing **Position** and **Project**.
you-said-4

use this formula for closure: Closure of beta under F (beta+) result := beta Do For each gema → delta in F: If (gema result): result := result Union delta While (result does not change) Return result Give closure of case_3 {EmployeeID,Department}

ChatGPT said:

Let's calculate the **closure** of $\{\text{EmployeeID}, \text{Department}\}${EmployeeID,Department} (i.e., **case_3**) using the formula you provided.

functional-dependencies-recap-1

Functional Dependencies Recap:

1. `{EmployeeID, Department, Location} -> {Salary}`
2. `{EmployeeID} -> {HireDate, Position}`
3. `{Department, Manager, Location} -> {Project}`
4. `{Project} -> {EmployeeID, HireDate, Location}`
5. `{Department, Location} -> {Manager, Position}`

closure-calculation-of-textemployeeidtextdepartmentemployeeiddepartment-under-f

Closure Calculation of $\{\textbf{EmployeeID}, \textbf{Department}\}${EmployeeID,Department} under F:Closure Calculation of \{\text{EmployeeID},\text{Department}\}{EmployeeID,Depar under F:

Step 1:

- Initialize:

$$\text{result} = \{\text{EmployeeID}, \text{Department}\}$$

  result={EmployeeID,Department}

Step 2:

- Check each functional dependency to see if the LHS (i.e., γ) is a subset of the current result, and if so, update the result by adding the RHS (i.e., δ).

iteration-1

Iteration 1:

1. **FD1:** $\{\textbf{EmployeeID}, \textbf{Department}, \textbf{Location}\} \rightarrow \{\textbf{Salary}\}$ **{EmployeeID,Department,Location**

   - LHS: $\{\text{EmployeeID}, \text{Department}, \text{Location}\}$ {EmployeeID,Department,Location} is **not a subset** of resultresult, since **Location** is missing.
   - No change to result.

2. **FD2:** $\{\textbf{EmployeeID}\} \rightarrow \{\textbf{HireDate}, \textbf{Position}\}$ **{EmployeeID}→{HireDate,Position}**

   - LHS: $\{\text{EmployeeID}\}$ {EmployeeID} is a subset of resultresult.
   - Add RHS: $\{\text{HireDate}, \text{Position}\}$ {HireDate,Position} to result.
   - Updated result:

   $$\text{result} = \{\text{EmployeeID}, \text{Department}, \text{HireDate}, \text{Position}\}$$

   result={EmployeeID,Department,HireDate,Position}

3. **FD3:** $\{\textbf{Department}, \textbf{Manager}, \textbf{Location}\} \rightarrow \{\textbf{Project}\}$ **{Department,Manager,Location}→{P**

   - LHS: $\{\text{Department}, \text{Manager}, \text{Location}\}$ {Department,Manager,Location} is **not a subset** of resultresult, since **Manager** and **Location** are missing.
   - No change to result.

4. **FD4:** $\{\textbf{Project}\} \rightarrow \{\textbf{EmployeeID}, \textbf{HireDate}, \textbf{Location}\}$ **{Project}→{EmployeeID,HireDate,L**

   - LHS: $\{\text{Project}\}$ {Project} is **not in result**.
   - No change to result.

5. **FD5:** $\{\textbf{Department}, \textbf{Location}\} \rightarrow \{\textbf{Manager}, \textbf{Position}\}$ **{Department,Location}→{Manager,I**

   - LHS: $\{\text{Department}, \text{Location}\}$ {Department,Location} is **not a subset** of resultresult, since **Location** is missing.
   - No change to result.

iteration-2

Iteration 2:

- At this point, the result is $\{\text{EmployeeID}, \text{Department}, \text{HireDate}, \text{Position}\}$ {EmployeeID,Department,Hire
- Let's check the functional dependencies again:

1. **FD1:** $\{\textbf{EmployeeID}, \textbf{Department}, \textbf{Location}\} \rightarrow \{\textbf{Salary}\}$ **{EmployeeID,Department,Location**
   - LHS: $\{\text{EmployeeID}, \text{Department}, \text{Location}\}$ {EmployeeID,Department,Location} is still **not a subset** of result.
   - No change to result.
2. **FD2:** $\{\textbf{EmployeeID}\} \rightarrow \{\textbf{HireDate}, \textbf{Position}\}$ **{EmployeeID}→{HireDate,Position}**
   - LHS: $\{\text{EmployeeID}\}$ {EmployeeID} is already a subset of result, but adding $\{\text{HireDate}, \text{Position}\}$ {HireDate,Position} makes no new additions.
   - No change to result.
3. **FD3:** $\{\textbf{Department}, \textbf{Manager}, \textbf{Location}\} \rightarrow \{\textbf{Project}\}$ **{Department,Manager,Location}→{P**
   - LHS: $\{\text{Department}, \text{Manager}, \text{Location}\}$ {Department,Manager,Location} is still **not a subset** of result.
   - No change to result.
4. **FD4:** $\{\textbf{Project}\} \rightarrow \{\textbf{EmployeeID}, \textbf{HireDate}, \textbf{Location}\}$ **{Project}→{EmployeeID,HireDate,L**
   - LHS: $\{\text{Project}\}$ {Project} is still **not in result**.
   - No change to result.
5. **FD5:** $\{\textbf{Department}, \textbf{Location}\} \rightarrow \{\textbf{Manager}, \textbf{Position}\}$ **{Department,Location}→{Manager,**
   - LHS: $\{\text{Department}, \text{Location}\}$ {Department,Location} is still **not a subset** of result.
   - No change to result.

result

Result:

- The result no longer changes after the second iteration. Therefore, the closure of $\{\text{EmployeeID}, \text{Department}\}$ {EmployeeID,Department} is:

$$\text{Closure}(\{\text{EmployeeID}, \text{Department}\}) = \{\text{EmployeeID}, \text{Department}, \text{HireDate}, \text{Position}\}$$

Closure({EmployeeID,Department})={EmployeeID,Department,HireDate,Position}

This means that from knowing just **EmployeeID** and **Department**, you can also determine **HireDate** and **Position**.

you-said-5

using same algorithm give closure of {EmployeeID,Department, Project}, {EmployeeID,Department, Manager}, {EmployeeID,Department, Location}

2/2

ChatGPT said:

Let's calculate the **closure** of the sets $\{\text{EmployeeID}, \text{Department}, \text{Project}\}$ {EmployeeID,Department,Projec
$\{\text{EmployeeID}, \text{Department}, \text{Manager}\}$ {EmployeeID,Department,Manager}, and $\{\text{EmployeeID}, \text{Department}, \text{Lo}$
using the same algorithm.

functional-dependencies-recap-2

Functional Dependencies Recap:

1. $\{\text{EmployeeID}, \text{Department}, \text{Location}\} \rightarrow \{\text{Salary}\}$ {EmployeeID,Department,Location}→{Salary}
2. $\{\text{EmployeeID}\} \rightarrow \{\text{HireDate}, \text{Position}\}$ {EmployeeID}→{HireDate,Position}
3. $\{\text{Department}, \text{Manager}, \text{Location}\} \rightarrow \{\text{Project}\}$ {Department,Manager,Location}→{Project}
4. $\{\text{Project}\} \rightarrow \{\text{EmployeeID}, \text{HireDate}, \text{Location}\}$ {Project}→{EmployeeID,HireDate,Location}
5. $\{\text{Department}, \text{Location}\} \rightarrow \{\text{Manager}, \text{Position}\}$ {Department,Location}→{Manager,Position}

---

closure-of-textemployeeidtextdepartmenttextprojectemployeeiddepartmentproject

1. Closure of $\{\textbf{EmployeeID}, \textbf{Department}, \textbf{Project}\}$ **{EmployeeID,Department,Project}1. Closure of \{\text{EmployeeID},\text{Department},\text{Project}\}{EmployeeID,Departme**

Step 1:

- Initialize:

$$\text{result} = \{\text{EmployeeID}, \text{Department}, \text{Project}\}$$

  result={EmployeeID,Department,Project}

Step 2:

- Apply the functional dependencies:

1. **FD1:** $\{\textbf{EmployeeID}, \textbf{Department}, \textbf{Location}\} \rightarrow \{\textbf{Salary}\}$ **{EmployeeID,Department,Location**
   - LHS: $\{\text{EmployeeID}, \text{Department}, \text{Location}\}$ {EmployeeID,Department,Location} is **not a subset** of result.
   - No change to result.
2. **FD2:** $\{\textbf{EmployeeID}\} \rightarrow \{\textbf{HireDate}, \textbf{Position}\}$ **{EmployeeID}→{HireDate,Position}**
   - LHS: $\{\text{EmployeeID}\}$ {EmployeeID} is a subset of result.
   - Add RHS: $\{\text{HireDate}, \text{Position}\}$ {HireDate,Position}.
   - Updated result:

$$\text{result} = \{\text{EmployeeID}, \text{Department}, \text{Project}, \text{HireDate}, \text{Position}\}$$

   result={EmployeeID,Department,Project,HireDate,Position}
3. **FD3:** $\{\textbf{Department}, \textbf{Manager}, \textbf{Location}\} \rightarrow \{\textbf{Project}\}$ **{Department,Manager,Location}→{P**
   - LHS: $\{\text{Department}, \text{Manager}, \text{Location}\}$ {Department,Manager,Location} is **not a subset** of result.
   - No change to result.

4. **FD4:** $\{\textbf{Project}\} \rightarrow \{\textbf{EmployeeID}, \textbf{HireDate}, \textbf{Location}\}$ **{Project}→{EmployeeID,HireDate,L**

   - LHS: $\{\text{Project}\}$ {Project} is a subset of result.
   - Add RHS: $\{\text{EmployeeID}, \text{HireDate}, \text{Location}\}$ {EmployeeID,HireDate,Location}.
   - Updated result:

$$\text{result} = \{\text{EmployeeID}, \text{Department}, \text{Project}, \text{HireDate}, \text{Position}, \text{Location}\}$$

result={EmployeeID,Department,Project,HireDate,Position,Location}

5. **FD5:** $\{\textbf{Department}, \textbf{Location}\} \rightarrow \{\textbf{Manager}, \textbf{Position}\}$ **{Department,Location}→{Manager,**

   - LHS: $\{\text{Department}, \text{Location}\}$ {Department,Location} is a subset of result.
   - Add RHS: $\{\text{Manager}, \text{Position}\}$ {Manager,Position}.
   - Updated result:

$$\text{result} = \{\text{EmployeeID}, \text{Department}, \text{Project}, \text{HireDate}, \text{Position}, \text{Location}, \text{Manager}\}$$

result={EmployeeID,Department,Project,HireDate,Position,Location,Manager}

Final result:

$$\text{Closure}(\{\text{EmployeeID}, \text{Department}, \text{Project}\}) = \{\text{EmployeeID}, \text{Department}, \text{Project}, \text{HireDate}, \text{Position}, \text{Locat}$$

Closure({EmployeeID,Department,Project})={EmployeeID,Department,Project,HireDate,Position,Location,

---

closure-of-textemployeeidtextdepartmenttextmanageremployeeiddepartmentmanager

2. Closure of $\{\textbf{EmployeeID}, \textbf{Department}, \textbf{Manager}\}$ {EmployeeID,Department,Manager}2. Closure of \{\text{EmployeeID},\text{Department},\text{Manager}\}{EmployeeID,Departm

Step 1:

- Initialize:

$$\text{result} = \{\text{EmployeeID}, \text{Department}, \text{Manager}\}$$

result={EmployeeID,Department,Manager}

Step 2:

- Apply the functional dependencies:

1. **FD1:** $\{\textbf{EmployeeID}, \textbf{Department}, \textbf{Location}\} \rightarrow \{\textbf{Salary}\}$ **{EmployeeID,Department,Location**
    - LHS: $\{\text{EmployeeID}, \text{Department}, \text{Location}\}$ {EmployeeID,Department,Location} is **not a subset** of result.
    - No change to result.
2. **FD2:** $\{\textbf{EmployeeID}\} \rightarrow \{\textbf{HireDate}, \textbf{Position}\}$ **{EmployeeID}→{HireDate,Position}**
    - LHS: $\{\text{EmployeeID}\}$ {EmployeeID} is a subset of result.
    - Add RHS: $\{\text{HireDate}, \text{Position}\}$ {HireDate,Position}.
    - Updated result:

$$\text{result} = \{\text{EmployeeID}, \text{Department}, \text{Manager}, \text{HireDate}, \text{Position}\}$$

result={EmployeeID,Department,Manager,HireDate,Position}

3. **FD3:** $\{\textbf{Department}, \textbf{Manager}, \textbf{Location}\} \rightarrow \{\textbf{Project}\}$ **{Department,Manager,Location}→{P**
    - LHS: $\{\text{Department}, \text{Manager}, \text{Location}\}$ {Department,Manager,Location} is **not a subset** of result.
    - No change to result.

4. **FD4:** $\{\textbf{Project}\} \rightarrow \{\textbf{EmployeeID}, \textbf{HireDate}, \textbf{Location}\}$ **{Project}→{EmployeeID,HireDate,L**

    - LHS: $\{\text{Project}\}${Project} is **not in result**.
    - No change to result.

5. **FD5:** $\{\textbf{Department}, \textbf{Location}\} \rightarrow \{\textbf{Manager}, \textbf{Position}\}$ **{Department,Location}→{Manager,I**

    - LHS: $\{\text{Department}, \text{Location}\}$ {Department,Location} is **not a subset** of result.
    - No change to result.

Final result:

$$\text{Closure}(\{\text{EmployeeID}, \text{Department}, \text{Manager}\}) = \{\text{EmployeeID}, \text{Department}, \text{Manager}, \text{HireDate}, \text{Position}\}$$

Closure({EmployeeID,Department,Manager})={EmployeeID,Department,Manager,HireDate,Position}

---

closure-of-textemployeeidtextdepartmenttextlocationemployeeiddepartmentlocation

3. Closure of $\{\textbf{EmployeeID}, \textbf{Department}, \textbf{Location}\}${EmployeeID,Department,Location}3. Closure of \{\text{EmployeeID},\text{Department},\text{Location}\}{EmployeeID,Departm

Step 1:

- Initialize:

$$\text{result} = \{\text{EmployeeID}, \text{Department}, \text{Location}\}$$

result={EmployeeID,Department,Location}

Step 2:

- Apply the functional dependencies:

1. **FD1:** $\{\textbf{EmployeeID}, \textbf{Department}, \textbf{Location}\} \rightarrow \{\textbf{Salary}\}$ **{EmployeeID,Department,Location**

- LHS: $\{\text{EmployeeID}, \text{Department}, \text{Location}\}${EmployeeID,Department,Location} is a subset of result.
- Add RHS: $\{\text{Salary}\}${Salary}.
- Updated result:

  $$\text{result} = \{\text{EmployeeID}, \text{Department}, \text{Location}, \text{Salary}\}$$

  result={EmployeeID,Department,Location,Salary}

2. **FD2:** $\{\textbf{EmployeeID}\} \rightarrow \{\textbf{HireDate}, \textbf{Position}\}$ **{EmployeeID}→{HireDate,Position}**

   - LHS: $\{\text{EmployeeID}\}${EmployeeID} is a subset of result.
   - Add RHS: $\{\text{HireDate}, \text{Position}\}${HireDate,Position}.
   - Updated result:

     $$\text{result} = \{\text{EmployeeID}, \text{Department}, \text{Location}, \text{Salary}, \text{HireDate}, \text{Position}\}$$

     result={EmployeeID,Department,Location,Salary,HireDate,Position}

3. **FD3:** $\{\textbf{Department}, \textbf{Manager}, \textbf{Location}\} \rightarrow \{\textbf{Project}\}$ **{Department,Manager,Location}→{P**

   - LHS: $\{\text{Department}, \text{Manager}, \text{Location}\}${Department,Manager,Location} is **not a subset** of result (because **Manager** is missing).
   - No change to result.

4. **FD4:** $\{\textbf{Project}\} \rightarrow \{\textbf{EmployeeID}, \textbf{HireDate}, \textbf{Location}\}$ **{Project}→{EmployeeID,HireDate,L**

   - LHS: $\{\text{Project}\}${Project} is **not in result**.
   - No change to result.

5. **FD5:** $\{\textbf{Department}, \textbf{Location}\} \rightarrow \{\textbf{Manager}, \textbf{Position}\}$ **{Department,Location}→{Manager,I**

   - LHS: $\{\text{Department}, \text{Location}\}${Department,Location} is a subset of result.
   - Add RHS: ({\text{Manager}, \text

Continue generating
prompt-textarea

composer-background