Canonical Cover

In database management systems (DBMS), a canonical cover is a set of functional dependencies that is equivalent to a given set of functional dependencies but is minimal in terms of the number of dependencies. The process of finding the canonical cover of a set of functional dependencies involves three main steps:

- **Reduction:** The first step is to reduce the original set of functional dependencies to an equivalent set that has the same closure as the original set, but with fewer dependencies. This is done by removing redundant dependencies and combining dependencies that have common attributes on the left-hand side.
- **Elimination:** The second step is to eliminate any extraneous attributes from the left-hand side of the dependencies. An attribute is considered extraneous if it can be removed from the left-hand side without changing the closure of the dependencies.
- **Minimization:** The final step is to minimize the number of dependencies by removing any dependencies that are implied by other dependencies in the set.

Illustrative Example

Consider a set of Functional dependencies: $F=\{A\rightarrow BC, B\rightarrow C, AB\rightarrow C\}$. Here are the steps to find the canonical cover –

Step 1:Decompose FDs to have a single attribute on the right-hand side

- $A\rightarrow BC$ becomes $A\rightarrow B$ and $A\rightarrow C$.
- Therefore, we have $\{A\rightarrow B,\ A\rightarrow C,\ B\rightarrow C,\ AB\rightarrow C\}$.

Step 2:Remove extraneous attributes from the left-hand side of FDs

- Checking $AB\rightarrow C$: First, check if A or B is extraneous.
- We can reach C without using $AB\rightarrow C$ with other functional dependencies; therefore, we remove $AB\rightarrow C$.
- Finally, we have $\{A\rightarrow B,\ A\rightarrow C,\ B\rightarrow C\}$.

Step 3:Remove redundant FDs

- Check each functional dependency to see if it can be reached without using it. For example, $A\rightarrow C$ can be reached with $A\rightarrow B$ and $B\rightarrow C$. Therefore, $A\rightarrow C$ is redundant and can be removed.
- Hence, Canonical Cover = $\{A\rightarrow B,\ B\rightarrow C\}$.

How to Find Canonical Cover

To compute the canonical cover for set F, follow this algorithm –

- Use the union rule to replace any dependencies in $\alpha 1 \rightarrow \beta 1$ and $\alpha 2 \rightarrow \beta 2$ with $\alpha 1 \rightarrow \beta 1\beta 2$.
- Find a functional dependency $\alpha \rightarrow \beta$ with an extraneous attribute either in α or in β.

- If an extraneous attribute is found, delete it from $\alpha \rightarrow \beta$.
- Repeat until F does not change.

Example 1

Given F = { A → BC, B → C, A → B, AB → C }

- **Step 1 Reduction:** There are two functional dependencies with the same attributes on the left: A → BC, A → B are already in their simplest form.
- **Step 2 Elimination:** In A → BC, C is extraneous because A → C can be derived from A → B and B → C. Thus, we reduce it to A → B.
- **Step 3 Minimization:** No redundant dependencies remain.

Hence, the canonical cover is Fc = { A → B, B → C }

Example 2

Given F = { A → BC, CD → E, B → D, E → A }

- **Step 1 Reduction:** Each left-hand side of the functional dependencies is unique and cannot be combined further.
- **Step 2 Elimination:** None of the attributes on the left or right sides of any functional dependency are extraneous.
- **Step 3 Minimization:** No dependencies are redundant.

Hence, the canonical cover is F = { A → BC, CD → E, B → D, E → A }

How to Check Whether a Set of FD's F Canonically Covers Another Set of FD's G?

Consider the following two sets of functional dependencies: F = { A → B, AB → C, D → A, CD → E } G = { A → B, CD → AB } Now, we are required to find out whether one of these f.d.'s canonically covers the other set of f.d.'s. This means, we need to find out whether F canonically covers G, G canonically covers F, or none of the two canonically cover the other. To find out, we follow the following steps:

- Create a singleton right-hand side. This means the attributes to the right side of the f.d. arrow should all be a singleton. All the ride side is single. So, F = { A → B, AB → C, D → A, CD → E }
- Remove all extraneous attributes. Consider any functional dependency XY → Z. If X in itself can determine Z, then the attribute Y is extraneous and can be removed. As we can see, the occurrence of extraneous attributes is possible only in those functional dependencies where there is more than one attribute in the LHS. So, consider the functional dependency AB → C. Now, we must find the closures of A and B to find whether any of these is extraneous. $[A]^+$=ABC, $[B]^+$=B As we can see, C can be determined from A. This means we can remove B from the functional dependency AB → C. F = { A → B, A → C, D → A, CD → E }

- Remove all redundant functional dependencies. Check all f.d.'s one by one, and see if by removing an f.d. X → Y, we can still find out Y from X by some other f.d. A more formal way to state this finds [X]+ without making use of the f.d. we are testing and check whether Y is a part of the closure. If yes, then the f.d. is redundant. No f.d can be removed in this step. So, final F = { A → B, A →C, D → A , CD → E } or, F = { A → BC, D → A , CD → E }.

Now, checking G this time

- Create a singleton right-hand side. This means the attributes to the right side of the f.d. arrow should all be a singleton. G = { A → B, CD → A, CD → B }
- Remove all extraneous attributes. Since the RHS of all f.d.'s contains only 1 attribute, no extraneous attribute is possible.
- Remove all redundant functional dependencies. By looping over all f.d.'s and checking the closure of the LHS in all cases, we observe that the f.d. CD → B is redundant as it can be obtained through a combination of 2 other f.d.'s, CD → A and A → B. So, final G = { A → B, CD → A }

Now, since all f.d.'s of G are not covered in F, we conclude that **F does not cover G**.

Features of the Canonical Cover

- **Minimal:** The canonical cover is the smallest set of dependencies that can be derived from a given set of dependencies, i.e., it has the minimum number of dependencies required to represent the same set of constraints.
- **Lossless:** The canonical cover preserves all the functional dependencies of the original set of dependencies, i.e., it does not lose any information.
- **Deterministic:** The canonical cover is deterministic, i.e., it does not contain any redundant or extraneous dependencies.
- **Reduces Data Redundancy:** The canonical cover helps to reduce data redundancy by eliminating unnecessary dependencies that can be inferred from other dependencies.
- **Improves Query Performance:** The canonical cover helps to improve query performance by reducing the number of joins and redundant data in the database.
- **Facilitates Database Maintenance:** The canonical cover makes it easier to modify, update, and delete data in the database by reducing the number of dependencies that need to be considered.