0.31 Canonical Cover Python Code

you-said

use code to find canonical cover of r2(A,B ,C, D,E ,G ,H ,K) ; FD = {{A, B, H} -> {C}, {A} -> {D, E}, {B ,G, H} -> {K}, {K} -> {A, D, H}, {B, H} -> {G, E}. Extraneous Attributes: Given -> : • a is extraneous if (- a)+ under F • a is extraneous if a + under F’= (F – { → }) Union { → (- a)} Canonical Cover: result = FDs Do: prev_result = result Replace all functional dependencies of type -> 1 and -> 2 by -> 1 Union 2 for f in result: for attribute in f: if attribute is extraneous: remove it from f while prev_result <> result Return result

ChatGPT said:

To find the canonical cover of the given functional dependencies (FDs), you can follow the steps you've outlined in the pseudocode. Here's the Python code implementation for this problem:

[] pythonCopy codefrom collections import defaultdict

# Function to compute the closure of a set of attributes given a set of FDs def compute_closure(attributes, fds): closure = set(attributes) while True: updated = False for lhs, rhs in fds: if set(lhs).issubset(closure) and not set(rhs).issubset(closure): closure.update(rhs) updated = True if not updated: break return closure

# Function to remove extraneous attributes def remove_extraneous(fds): # Step 1: Remove extraneous attributes from left-hand side for i, (lhs, rhs) in enumerate(fds): for attr in lhs: reduced_lhs = lhs.replace(attr, '') if set(rhs).issubset(compute_closure(reduced_lhs, fds)): fds[i] = (reduced_lhs, rhs)

# Step 2: Remove extraneous attributes from right-hand side for i, (lhs, rhs) in enumerate(fds): for attr in rhs: reduced_rhs = rhs.replace(attr, '') if attr in compute_closure(lhs, [fd for j, fd in enumerate(fds) if j != i]): fds[i] = (lhs, reduced_rhs)

return fds

# Function to combine FDs with the same left-hand side def combine_fds(fds): fd_map = defaultdict(set) for lhs, rhs in fds: fd_map[lhs].update(rhs)

return [(lhs, ''.join(sorted(rhs))) for lhs, rhs in fd_map.items()]

# Main function to compute canonical cover def canonical_cover(fds): result = fds prev_result = None

while prev_result != result: prev_result = result.copy()

# Step 1: Combine FDs with same left-hand side result = combine_fds(result)

# Step 2: Remove extraneous attributes result = remove_extraneous(result)

return result

# Example input fds = [(”ABH”, ”C”), (”A”, ”DE”), (”BGH”, ”K”), (”K”, ”ADH”), (”BH”, ”GE”)]

# Compute canonical cover canonical_fds = canonical_cover(fds)
# Display the result for lhs, rhs in canonical_fds: print(f"{lhs} -> {rhs}")
steps-followed-in-the-code

Steps followed in the code:

1. **Closure Calculation**: The `compute_closure` function computes the closure of a set of attributes based on the given FDs.
2. **Extraneous Attribute Removal**: The `remove_extraneous` function iterates over each FD to identify and remove extraneous attributes from both the left-hand side and right-hand side.
3. **Combining FDs**: The `combine_fds` function combines functional dependencies that have the same left-hand side, consolidating the right-hand sides.
4. **Iterative Process**: The `canonical_cover` function repeats the process of combining FDs and removing extraneous attributes until no further changes are made.

You can run this code with the provided example FDs to find the canonical cover.
you-said-1

what is the output of this
ChatGPT said:
It seems like I can't do more advanced data analysis right now. Please try again later. If you'd like, you can try running the code locally to see the output, or I can help explain the expected results based on the logic of the code.
prompt-textarea
composer-background