Homework 10 Solution

Imagine that you are the inexperienced designer we've been talking about, and that you've designed the multi-company database of SQL Chapter as A SINGLE TABLE DATABASE.

-Write up the design as such, i.e., as a one-table design, changing attribute names where necessary. (Be sure to write up the meaning of each attribute, being careful to use "a" "the" "that" and other words to indicate constraints.)

-Write the set of functional dependencies that goes with the one-table design. They must represent exactly the same constraints as apply to the original design of the multi-company database. You're not allowed to drop any and you're not allowed to add any.

-Normalize the table, and discuss whether your normalization is the same as or different from the original design – and why

The original database schema is

company(co_name, govt_id, ceo_ssn, hq_loc)
division(co_id, div_name, subdiv_of, dir_ssn, div_hq)
site(co_id, div_name, loc)
product(prod_id, manuf_co, manuf_div, loc, prod_descr)
person(ssn, name, address)
works_for(ssn, co_id, div_name, salary, emp_id, sup_ssn)
skill(ssn, prod_id, manuf_co)

•Each row of the *multiple-company-database* table will hold (colors shows boundaries of original table schemas that attributes come from):

- •co_name: the name of a company
- •govt_id: the tax identification number assigned to the company by the federal government
- •ceo_ssn: the social security number of the company's chief executive officer
- •hq_loc: the name of the city in which the company's headquarters are located
- •div_name: the name of a division of the company
- •subdiv_of: the name of that division of the company of which this division is a direct subdivision.
- •dir_ssn: the social security number of the division's director
- •div_hq: the city in which the division's headquarters are located
- •div_site: the name of a city in which a site (installation) of the division of the company is located.
- •prod_id: the company-assigned identifier of a product that the division of the company manufactures
- •prod_loc: the location (site) of the division of the company at which the product is manufactured
- •prod_descr: the company-assigned description of the product
- •ssn: a social security number
- •name: the name of the person with that social security number
- •address: the address of the person with that social security number
- •emp_ssn: the social security number of a person who works for the division of the company
- •emp_salary: the person's (emp_ssn's) yearly salary paid by the company
- •emp_id: the identification number assigned to the person (emp_ssn) by the company
- •sup_ssn: the social security number of the person's (emp_ssn's) direct supervisor in the company
- •skill_ssn: the social security number of a person trained in the manufacture of the product for the company

The "ringers" in the above are the attributes from the PERSON table, that have no clear connection to any of the others, and probably would have been modeled as a separate table even by a very inexperienced designer.

govt_id -> co_name, ceo_ssn, hq_loc
govt_id, div_name -> subdiv_of, dir_ssn, div_hq
prod_id, govt_id -> manuf_div, manuf_loc, prod_descr
ssn -> name, address
emp_ssn, govt_id -> div_name, emp_salary, emp_id, sup_ssn

div_site is an orphan attribute
skill_ssn is an orphan attribute

There will also be recursive foreign key references:
-from ceo_ssn to ssn
-from dir_ssn to ssn
-from emp_ssn to ssn
-from sup_ssn to ssn
-from skill_ssn to ssn

The rules for normalization would give us the original tables