Zac Fotcit
CPSC 321:01
September 23 2021

27/30

HW#2

① a) $\Pi_{maker}(PC \bowtie_{hd=256GB} Product)$

b) $\Pi_{model, price}(\sigma_{maker=B}(Product \bowtie_{model} PC))$ need union -1

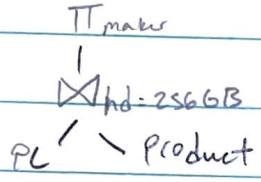
c) $\Pi_{maker}((Product \bowtie_{model} Laptop) \cap (Product \bowtie_{model} PC))$ -1 need set difference

d) $\Pi_{hd}(PC \bowtie_{hd=hd} PC)$ & diff. model -1

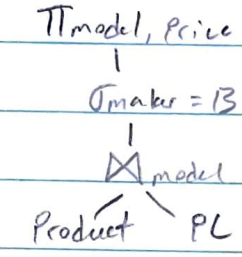
e) $\Pi_{maker}(\sigma_{speed>4.2GHz}(Product \bowtie_{model} PC))$

need set diff -1

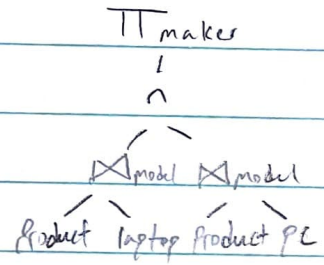
② (a)



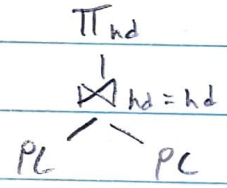
(b)



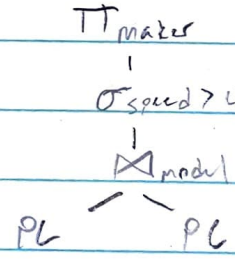
(c)



(d)



(e)



③ a) min = ~~n+m~~ max(n,m) -1, max = n+m

b) min = ~~n~~ 0, max = ~~n+m~~ min(n,m) -1

c) min = ~~n~~ 0, max = n~~+~~*m -1

d) min = ~~n+m~~ 0, max = n~~+~~*m -1

e) min = 0; max = ~~(n - len(Π₁(P)) + m~~ n -1

(4) a) $\Pi_{item\_number,\ name}(\sigma_{part\_list > 10 \wedge price < 25}(Sets))$ join w/ PartList -1

b) $\Pi_{item\_number,\ name}((\sigma_{part\_list.name = 1x2\ brick \wedge color = bright\ blue}(Sets)) \bowtie (\sigma_{part\_list.name = 2x4\ plate \wedge color = bright\ red}(Sets)))$ -1

c) $\Pi\Pi_{item\_number,\ name}(Sets \bowtie_{theme = Disney \wedge category = Building}(Sets))$

d) $\Pi_{item\_number,\ name}((\sigma_{category = Building}(Sets)) \bowtie_{eid} (\sigma_{category = Sports}(Sets)))$ a key? -1

e) $\Pi_{eid,\ name,\ price}(Bricks \bowtie_{design\_id \neq design\_id \wedge elem\_id = elem\_id \wedge price < 0.20} Bricks)$

I needed to structure my Sets table more around the part list attribute to have access to those attributes -1 → need identical bricks table

1) I will be modifying an existing project I have been working on for some time. It will be an expense calculator written in Python that allows a users to track their expenses over a period of time. The current Database component only stores single transactions, tracks a user's current balance, & output a history of the user's transactions. I would like to add more features to flesh out this app. Giving users the ability to tag transactions for easy review would be beneficial. Also providing the users w/ visualizations & trend graphs would be extremely useful. Designing a simple & effective Gui to allow users to navigate the functions of the application is a important feature to add as well. finally, I believe it would be of great benefit to the user to allow entry of multiple transactions at the same time.

2) For this version of the proposal, I decided on 2 new features to describe. First, implementing a GUI is paramount because it should be very clear and easy for a user to interact with the application properly. Additionally, it would be helpful for users to have a feature to enter transactions in bulk in case they are behind on entries.

(3) User tasks in application

- The user will login using their credentials when prompted by the login screen. This will retrieve their associated data (OLTP)
- The user can enter a transaction amount, chose deposit or withdrawl, describe their transaction, and select a transaction tag. Once they click a submit button, the user's transaction will be processed & inserted into the database (OLTP)
- The user can click a history button to have their transactions compiled & displayed to them. This will likely open a new page to the user to allow them to filter out their results (OLAP)
- The user's homepage will display visualizations of their data in the form of trend graphs & other forms (OLAP) details--
- The user will have the capability in the history screen to delete or modify past entries. Ideally this will be accomplished with a button & a pop-up window (OLTP)