Name | CWID

Final Exam

December 5th, 2016
2:00-4:00

CS425 - Database Organization
Results

Please leave this empty!

1.1	1.2	1.3	1.4	1.5	1.6	Sum

Instructions

- Try to answer all the questions using what you have learned in class. Keep hard questions until the end.
- **When writing a query, write the query in a way that it would work over all possible database instances and not just for the given example instance!**
- The exam is closed book and closed notes!

Consider the following laptop database schema and example instance:

laptop

manufacturer	model	modelNr	price	releaseYear
Apple	Macbook Pro	Retina, 15-inch, Mid 2014	2500	2014
Apple	Macbook Air	11-inch, Mid 2015	1300	2015
Dell	Inspire	17 - 5000	700	2011

manufacturer

name	yearsGuarantee	country
Apple	1	USA
Dell	2	USA
Lenovo	3	China
Intel	0	USA
AMD	0	USA

part

partId	manufacturer	type	description
Intel Core m3-7Y30	Intel	processor	2.6 GHz
Sempron Huron	AMD	processor	1.5 Ghz
Apple Retina Display 15 inch	Apple	display	

partOf

manufacturer	model	modelNr	partId	quantity
Apple	Macbook Pro	Retina, 15-inch, Mid 2014	Apple Retina Display 15 inch	1
Apple	Macbook Pro	Retina, 15-inch, Mid 2014	Intel Core m3-7Y30	2
Dell	Inspire	17 - 5000	Sempron Huron	1

Hints:

- Attributes with black background form the primary key of a relation (e.g., *name* for relation *manufacturer*)
- The attribute *manufacturer* of relation *laptop* is a foreign key to relation *manufacturer*
- The attribute *manufacturer* of relation *part* is a foreign key to relation *manufacturer*
- The attribute *partId* of relation *partOf* is a foreign key to relation *part*.
- The attribute *manufacturer*, *model*, *modelNr* of relation *partOf* form a foreign key to relation *laptop*
- All foreign keys have been created with the **CASCADE** option.

Part 1.1 Relational Algebra (Total: 20 Points)

The queries in this part should be written using the **set semantics** version of relational algebra.

Question 1.1.1 (4 Points)

Write a **relational algebra** expression that returns pairs of laptops and parts (return *manufacturer*, *model*, *modelNr*, *price*, and *partId*) such that the part is built into the laptop (this information is stored in `partOf`) and the type of the part is either `display` or `processor`.

Solution

$$\pi_{man,model,modelNr,price,partId}(\rho_{man \leftarrow manufacturer}(laptop \bowtie partOf) \bowtie \sigma_{type=processor \vee type=display}(part))$$

Question 1.1.2 (4 Points)

Write a **relational algebra** expression that returns the *name* of manufacturers that give at least 2 years of guarantee and produce at least one laptop with an intel processor (a part used in this laptop is of type `processor` and is produced by `Intel`).

Solution

$$\pi_{name}(\rho_{man \leftarrow manufacturer}(\sigma_{guarantee \geq 2}(manufacturer) \bowtie_{name=manufacturer} laptop \bowtie partOf)$$
$$\bowtie \sigma_{manufacturer=Intel \wedge type=processor}(part))$$

Question 1.1.3 (5 Points)

Write a **relational algebra** expression that returns the average years of guarantee for laptops (the guarantee for a laptop is determined by the manufacturer that produces the laptop).

Solution

$$\mathcal{G}_{avg(yearsGuarantee)}(laptop \bowtie_{manufacturer=name} manufacturer)$$

Question 1.1.4 (7 Points)

Write a **relational algebra** expression that returns laptops (their *model* and *modelNr*) that only contain parts produced by the manufacturer that produces the laptop.

Solution

$$\begin{aligned} laptopPart &\leftarrow laptop \bowtie_{manufacturer=name} manufacturer \bowtie partOf \bowtie \rho_{pman \leftarrow manufacturer}(part) \\ laptopWithOther &\leftarrow \pi_{manufacturer,model,modelNr}(\sigma_{manufacturer \neq pman}(laptopPart) \\ q &\leftarrow \pi_{model,modelNr}(\pi_{manufacturer,model,modelNr}(laptop) - laptopWithOther) \end{aligned}$$

Part 1.2 SQL - DDL (Total: 7 Points)

Question 1.2.1 (7 Points)

Write an **SQL statement** that creates a new relation *desktop* that records information about desktop computers. Desktop computers are identified by a combination of *manufacturer*, *model*, and *modelNr*. Furthermore, we store a *price* and *format*. Attribute *format* can only take one of the following values: `nettop`, `mini`, or `tower`. Furthermore, create a relation *desktopParts* that stores which part is built into which desktop. Parts can be built into multiple desktops and multiple desktops can contain the same part.

Solution

```
CREATE TABLE desktop (
    manufacturer VARCHAR(100),
    model VARCHAR(200),
    modelNr VARCHAR(200),
    price NUMERIC,
    format VARCHAR(6),
    PRIMARY KEY (manufacturer, model, modelNr),
    FOREIGN KEY (manufacturer) REFERENCES manufacturer,
    CHECK(format = 'nettop' OR format = 'mini' OR format = 'tower')
);

CREATE TABLE desktopPart (
    manufacturer VARCHAR(100),
    model VARCHAR(200),
    modelNr VARCHAR(200),
    partId VARCHAR(200),
    PRIMARY KEY (manufacturer, model, modelNr, partId),
    FOREIGN KEY (manufacturer, model, modelNr) REFERENCES desktop,
    FOREIGN KEY (partId) REFERENCES part
);
```

Part 1.3 SQL - Queries (Total: 20 Points)

Question 1.3.1 (3 Points)

Write an **SQL query** that returns the number of laptop models per manufacturer.

Solution

```
SELECT count(*) AS numLap, manufacturer
FROM laptop
GROUP BY manufacturer;
```

Question 1.3.2 (4 Points)

Write an **SQL query** that returns the price of laptops if their *model* attribute contains the string “Macbook”. Make sure each price is only returned once.

Solution

```
SELECT DISTINCT price
FROM laptop
WHERE model LIKE '%Macbook%'
```

Question 1.3.3 (6 Points)

Write an **SQL query** that returns the name(s) of the manufacturer(s) with the longest guarantee (*yearsGuarantee*).

Solution

```
SELECT name
FROM manufacturer
WHERE yearsGuarantee = (SELECT max(yearsGuarantee) FROM manufacturer)
```

Question 1.3.4 (7 Points)

Write an **SQL query** that returns the *partId* of parts that are not built into any laptop.

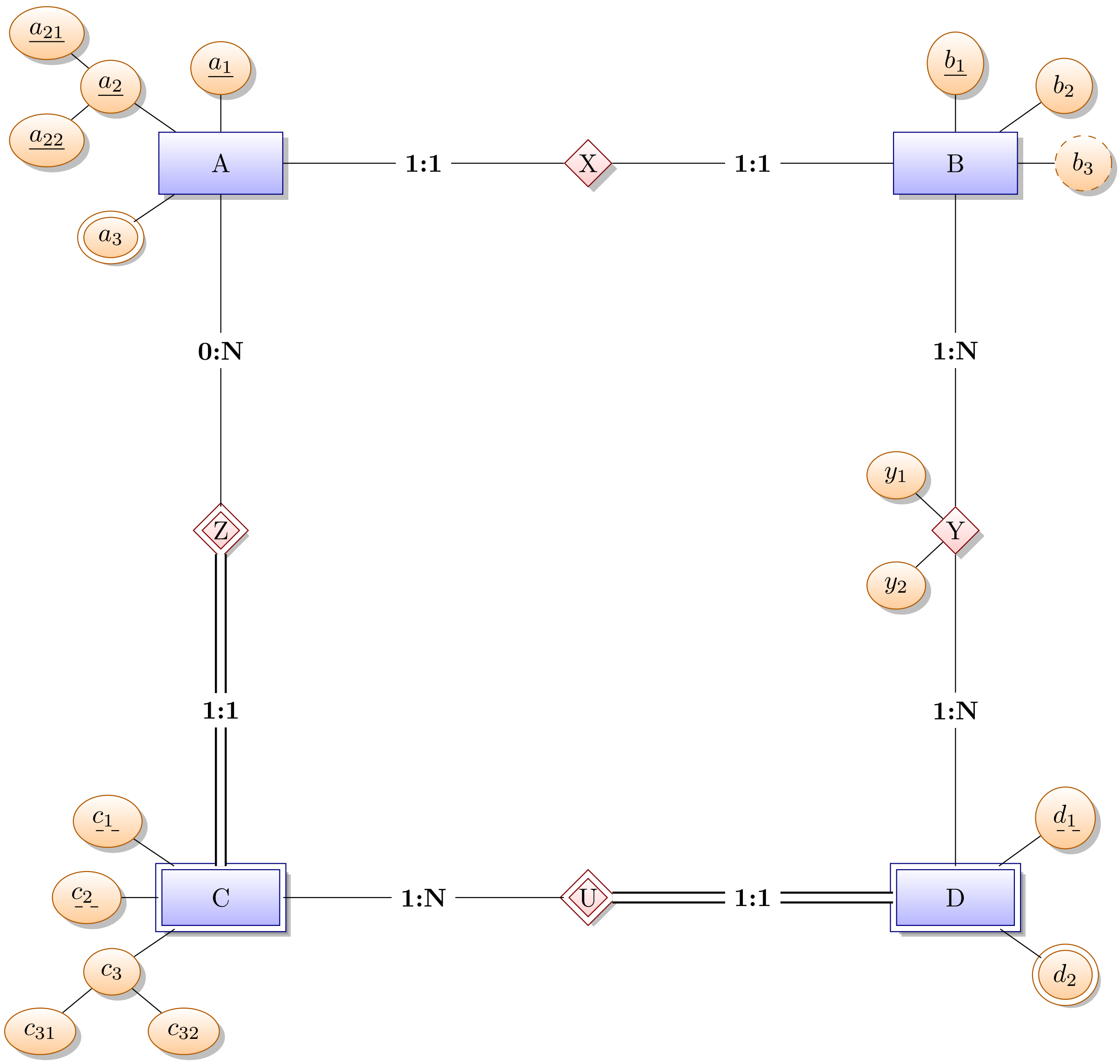
Solution

```
SELECT partId
FROM part
WHERE partId NOT IN (SELECT partId
                     FROM partOf o)
```

Part 1.4 Translation of ER into Relational Model (Total: 22 Points)

Question 1.4.1 (22 Points)

Take the following ER-model and translate it into a relational schema using the rules presented in class. Present the relational schema using the notation from the slides. For example, a relation R with attributes a_1 and a_2 where a_2 is the primary key is written as $R(a_1, \underline{a_2})$. You do not need to specify foreign key constraints. You do not have to show intermediate results.



Solution

1st Step: strong entities

$A(\underline{a_1}, \underline{a_{21}}, \underline{a_{22}})$
$B(\underline{b_1}, b_2)$

2nd Step: weak entities

$A(\underline{a_1}, \underline{a_{21}}, \underline{a_{22}})$
$B(\underline{b_1}, b_2)$
$C(\underline{a_1}, \underline{a_{21}}, \underline{a_{22}}, \underline{c_1}, \underline{c_2}, c_{31}, c_{32})$
$D(\underline{a_1}, \underline{a_{21}}, \underline{a_{22}}, \underline{c_1}, \underline{c_2}, \underline{d_1})$

3rd Step: relationships

$A(\underline{a_1}, \underline{a_{21}}, \underline{a_{22}}, b_1)$
$B(\underline{b_1}, b_2)$
$C(\underline{a_1}, \underline{a_{21}}, \underline{a_{22}}, \underline{c_1}, \underline{c_2}, c_{31}, c_{32})$
$D(\underline{a_1}, \underline{a_{21}}, \underline{a_{22}}, \underline{c_1}, \underline{c_2}, \underline{d_1})$
$Y(\underline{b_1}, \underline{a_1}, \underline{a_{21}}, \underline{a_{22}}, \underline{c_1}, \underline{c_2}, \underline{d_1}, y_1, y_2)$

4th Step: multi-valued attribues

$A(\underline{a_1}, \underline{a_{21}}, \underline{a_{22}}, b_1)$
$A3(\underline{a_1}, \underline{a_{21}}, \underline{a_{22}}, \underline{a_3}$
$B(\underline{b_1}, b_2)$
$C(\underline{a_1}, \underline{a_{21}}, \underline{a_{22}}, \underline{c_1}, \underline{c_2}, c_{31}, c_{32})$
$D(\underline{a_1}, \underline{a_{21}}, \underline{a_{22}}, \underline{c_1}, \underline{c_2}, \underline{d_1})$
$D2(\underline{a_1}, \underline{a_{21}}, \underline{a_{22}}, \underline{c_1}, \underline{c_2}, \underline{d_1}, \underline{d_2})$
$Y(\underline{b_1}, \underline{a_1}, \underline{a_{21}}, \underline{a_{22}}, \underline{c_1}, \underline{c_2}, \underline{d_1}, y_1, y_2)$

Part 1.5 Normalization and Functional Dependencies (Total: 22 Points)

Consider the following relation $R(A, B, C, D, E, F, G)$ and functional dependencies $\mathcal{F}$ that hold over this relation.

$$\begin{aligned}\mathcal{F} = & A \rightarrow C, F \\ & B \rightarrow E, C, D \\ & D \rightarrow E, C, B \\ & F \rightarrow G, A\end{aligned}$$

Question 1.5.1 (4 Points)

Determine all candidate keys of R.

Solution

$\{A, B\}$ $\{A, D\}$ $\{F, B\}$ $\{F, D\}$

Question 1.5.2 (8 Points)

Compute the canonical cover of F. Show each step of the generation according to the algorithm shown in class.

Solution

1th iteration: 1) Apply union rule to combine right-hand sides:
no changes
1th iteration: 2) Find extraneous attribute:
E is extraneous in $B \to E, C, D$

$$\begin{aligned}\mathcal{F}_1 = & A \to C, F \\ & B \to C, D \\ & D \to E, C, B \\ & F \to G, A\end{aligned}$$

2nd iteration: 1) Apply union rule to combine right-hand sides:
no changes
2nd iteration: 2) Find extraneous attribute:

C is extraneous in $B \to C, D$

$$\begin{aligned}\mathcal{F}_2 = & A \to C, F \\ & B \to D \\ & D \to E, C, B \\ & F \to G, A\end{aligned}$$

3nd iteration: 1) Apply union rule to combine right-hand sides:
no changes
3nd iteration: 2) Find extraneous attribute:
no changes

Question 1.5.3 (6 Points)

Apply the 3NF decomposition algorithm to relation R.

Solution

1st step: Create one relation for each FD in F_c:

$$R_1(A, C, F)$$
$$R_2(B, D)$$
$$R_3(D, E, C, B)$$
$$R_4(F, G, A)$$

2st step: Remove redundant relations
R_2 is redundant, remove it

$$R_1(A, C, F)$$
$$R_3(D, E, C, B)$$
$$R_4(F, G, A)$$

3st step: If no relation contains all attributes of a candidate key, then create a relation with all attributes of a candidate key
None contain a candidate key, so, e.g., create
R_5 contains (A, B)

Question 1.5.4 (6 Points)

In what normal forms is relation R?

Solution

1NF, not in 2NF because E depends only on part of the key.

Part 1.6 Concurrency Control (Total: 9 Points)

Question 1.6.1 (9 Points)

For each of the following schedules determine which properties this schedule has. E.g., a schedule may be *recoverable* and *cascade-less*. Consider the following notation for operations of transactions:

$w_1(A)$	transaction 1 wrote item A
$r_1(A)$	transaction 1 read item A
c_1	transaction 1 commits
a_1	transaction 1 aborts

$$S_1 = r_2(B), w_2(C), w_1(C), w_2(A), c_2, r_3(A), c_1, c_3$$

$$S_2 = r_2(A), r_1(B), w_2(A), r_2(B), r_3(A), w_1(B), c_1, w_3(A), c_3, w_2(B), c_2$$

$$S_3 = r_1(A), r_1(B), r_2(C), w_2(C), w_1(B), r_2(A), w_2(A), c_1, w_2(B), c_2$$

- [x] S_1 is recoverable
- [x] S_1 is cascade-less
- [x] S_1 is conflict-serializable
- [x] S_1 is 2PL
- [ ] S_1 is S2PL
- [ ] S_1 is SS2PL

- [ ] S_2 is recoverable
- [ ] S_2 is cascade-less
- [ ] S_2 is conflict-serializable
- [ ] S_2 is 2PL
- [ ] S_2 is S2PL
- [ ] S_2 is SS2PL

- [x] S_3 is recoverable
- [x] S_3 is cascade-less
- [x] S_3 is conflict-serializable
- [x] S_3 is 2PL
- [x] S_3 is S2PL
- [ ] S_3 is SS2PL