COURSE: CS/DSA-4513 -DATABASE MANAGEMENT SECTION: 001

SEMESTER: FALL 2021

INSTRUCTOR: Dr. Le Gruenwald

GRADED HOMEWORK NUMBER: 4

GROUP NUMBER: 18

GROUP MEMBERS: Naeem Shahabi Sani, Ashesh Guar, Jay Rothenberger

SCORE:

Problem 1:

a) $\Pi_{\text{person\_name}}(\sigma_{\text{company}-\text{name} = \text{"BigBank"}}\ (\text{works}))$

b) $\Pi_{\text{person\_name,city}}((\sigma_{\text{company}-\text{name} = \text{"BigBank"}}\ (\text{works})) \bowtie \text{employee})$

c) $\Pi_{\text{person\_name,city,street}}(\text{employee} \bowtie$
$\sigma_{(\text{company}-\text{name} = \text{"BigBank"} \wedge \text{salary} > 10000)}\text{works})$

d) $\Pi_{\text{person\_name}}(\text{works} \bowtie \text{employee} \bowtie \text{company})$

Problem 2.

a)
Attribute set closure –
(classid, id, gender)+ :

Result = classid, id, gender

Result = classid, id, gender, name (id->name && id ⊂ Result)

Result = classid, id, gender, name, age (name -> age, id)

Result = classid, id, gender, name, age, salary, manager (classid, id, gender -> salary, manager)

Thus, (classid, id, gender) is a superkey of the relational schema.
No combination of attributes (or individual attributes) in this superkey is a superkey itself. Thus, (classid, id, gender) is a candidate key.

Considering manager+ :

Result = manager

Result = manager, gender, age, classid, id (manager ⊂ manager && manager -> gender, age, classid, id)

Result = manager, gender, age, classid, id, salary (classid, id, gender -> salary, manager && classid, id, gender ⊂ Result)

Result = manager, gender, age, classid, id, salary, name

Thus, manager is a superkey of the relational schema.
Also, since it is a single attribute, it is a candidate key.

b)
1NF:
Since all attributes are atomic, 1NF is satisfied by all FDs.

2NF:
The non prime attributes are: name, age & salary
For name:
Id -> name means that name is functionally dependent on a subset of the candidate key (classid, id, gender). Thus, 2NF second property is violated here.

For age:
Name -> age, id
Thus, by decomposition rule:
Name -> age
It is not a trivial FD. Also, it is functionally dependent on a non primary attributed. Thus, 2NF is violated here as well.

For Salary:
Classid, id, gender -> salary, manager
By decomposition rule:
Classid, id, gender -> salary

It is not a trivial FD. Also, salary is fully functionally dependent on a candidate key. Thus, 2NF is not violated by this FD.

3NF:
Considering FD classid, id, gender -> salary, manager
Classid, id, gender is a superkey of the relational schema, thus, 3NF is obeyed.

Considering FD name -> age, id
It is not a trivial FD.
Name is not a super key of the schema.
Id is a prime attribute, but age is not, so
Name -> age violates 3NF.

Considering FD id -> name
It is not a trivial FD
Id is not a super key of the schema (it is a prime attribute)
Name is not a prime attribute of the schema. Thus, 3NF is violated by this FD

Considering the FD manager -> gender, age, classid, id
Manager is a superkey of the schema (candidate key).
Thus, this FD obeys 3NF

4NF:

Considering FD classid, id, gender -> salary, manager
Classid, id, gender is a superkey of the relational schema, thus, 3NF is obeyed.

Considering FD name -> age, id
It is not a trivial FD.
Name is not a super key of the schema.
Thus, BCNF is violated

Considering FD id -> name
It is not a trivial FD
Id is not a super key of the schema (it is a prime attribute)
Thus, BCNF is violated.

Considering the FD manager -> gender, age, classid, id
Manager is a superkey of the schema (candidate key).
Thus, BCNF is obeyed by this FD.

c)
The highest Normal Form that the relation schema does not satisfy is 2NF.
Thus, functional decomposition to remove the FD’s violating 2NF:
Id -> name
Name -> age

Thus, R1 (id, classid, gender, manager, salary)
R2(id, name)
R3(name, age)

For R1:
F+:
classid, id, gender -> salary, manager,
Manager -> gender, classid, id

Candidate keys for R1:

Considering manager+ :
Result = manager, gender, classid, id (manager -> gender, classid, id && manager ⊂ manager)
Result = manager, gender, classid, id, salary (classid, id, gender -> salary, manager && classid, id, gender ⊂ Result)

Thus, manager is a candidate key (a single attribute)

Considering (classid, id, gender)+
Result = classid, id, gender, salary, manager (classid, id, gender -> salary manager and classid, id, gender ⊂ Result)
Thus, classid, id, gender is a super key.
It is also a candidate key as none of the constituent attributes is a super key according to the F+ for this schema.
Thus, prime attributes : manager, classid, id, gender
Non prime attributes: salary
For salary: classid, id, gender -> manager, salary
By the decomposition rule : classid, id, gender -> salary.
Thus, salary is fully dependent on the candidate key classid, id, gender

Also, manager -> gender, classid, id
We know that gender, classid, id -> manager, salary
Thus, manager -> manager, salary
And so, manager -> salary
Thus, salary is fully dependent on the second candidate key, manager.
Thus, R1 is in 2NF.

For R2:

F+: id -> name
Candidate key: id
Non prime attribute: name
Name is fully dependent on the candidate key in R2. Thus, R2 obeys 2NF.

For R3:

F+: name -> age
Candidate key: name
Non prime attribute: age
Name is fully dependent on the candidate key in R3. Thus, R3 obeys 2NF.

Problem 3

Description: Gold's Gym maintains a database system to manage customer information across their many franchises. for eahc customer the system stores their name, address, a unique email address, their saved payment information, and the ids of the franchises locations at which they have a membership.

for each franchise the database stores a unique address, contact information consisting of a contact email and phone number, and the name of the franchise owner. Some franchises support the use of fingerprint access to their facility, so the database also stores the fingerprint of customers who choose to have access to this feature.

Each customer has an account that records their balance, payment interval, and optionally the credit card a customer has selected to use for autopay, if they have chosen to do so.

From this description we can derive the following set of functional dependencies:

$$\{email \rightarrow name, address, fingerprint;$$
$$\text{fingerprint} \rightarrow name, address, email;$$
$$\text{franchise\_address, customer\_email} \rightarrow payment\_interval, auto\_pay\_card;$$
$$\text{franchise\_address} \rightarrow contact\_phone, contact\_email, owner\_name;$$
$$\text{referred\_email, referrer\_email} \rightarrow date\_referred;$$
$$\text{card\_number} \rightarrow billing\_address; \}$$

We propose the following schema each of which are in Third Normal Form:

Schema: **Customer(email, name, address, fingerprint)**

From the description we have that email is unique for each customer. Since each customer can have only one fingerprint, and since the database records only one address for each customer this gives us the dependencies:

$$\{email \rightarrow name, address, fingerprint$$
$$\text{fingerprint} \rightarrow name, address, email\}$$

Using Armstrong's Axioms we can compute a non-trivial part of the closure of this set of dependencies:

$$\{email \rightarrow name, address, fingerprint;$$
$$\text{email} \rightarrow name, address;$$
$$\text{email} \rightarrow name, fingerprint;$$
$$\text{email} \rightarrow fingerprint, address;$$
$$\text{email} \rightarrow fingerprint;$$
$$\text{email} \rightarrow address;$$
$$\text{email} \rightarrow name;$$

$$
\begin{gathered}
\text{fingerprint} \rightarrow name, address, email; \\
\text{fingerprint} \rightarrow name, address; \\
\text{fingerprint} \rightarrow email, address; \\
\text{fingerprint} \rightarrow name, email; \\
\text{fingerprint} \rightarrow name; \\
\text{fingerprint} \rightarrow email; \\
\text{fingerprint} \rightarrow address; \}
\end{gathered}
$$

From this set of dependencies we can deduce that both fingerprint and email are minimal superkeys, and thus candidate keys, and thus also they are prime attributes for this schema.
Claim: this schema is in Third Normal Form

proof:
All attributes in Customer are atomic, thus it is in First Normal Form

Above are listed (at least) all of the functional dependencies $X \rightarrow A$ such that X is a candidate key and A is an attribute in Customer. Trivially, any superset U of X for $X \in \{email, fingerprint\}$ must be a superkey of Customer, so any rule in F^+, $U \rightarrow A$, must satisfy the property that U be a superkey of Customer. All other rules derived from armstrong's axioms will be trivial.
□

Schema: **membership(franchise_address, customer_email, auto_pay_card, payment_interval)**

From the description we have that an email is unique for each customer, and each customer can have only one membership at each franchise. Naturally, each franchise can only be at one location and each membership can only have one interval on which it is paid for. From this information we deduce the following set of functional dependencies:

$$\{franchise\_address, customer\_email \rightarrow payment\_interval, auto\_pay\_card\}$$

From this set we can compute all of the rules in F^+, $X- > Y$ such that X is minimal using Armstrong's Axioms:

$$
\begin{gathered}
\{franchise\_address, customer\_email \rightarrow payment\_interval, auto\_pay\_card; \\
\text{franchise\_address} \rightarrow franchise\_address; \\
\text{customer\_email} \rightarrow customer\_email; \\
\text{payment\_interval} \rightarrow payment\_interval; \\
\text{auto\_pay\_card} \rightarrow auto\_pay\_card; \\
\text{franchise\_address, customer\_email} \rightarrow payment\_interval; \\
\text{franchise\_address, customer\_email} \rightarrow auto\_pay\_card; \\
\text{franchise\_address, customer\_email} \rightarrow payment\_interval, franchise\_address; \\
\text{franchise\_address, customer\_email} \rightarrow auto\_pay\_card, franchise\_address; \\
\text{franchise\_address, customer\_email} \rightarrow payment\_interval, customer\_email; \\
\text{franchise\_address, customer\_email} \rightarrow auto\_pay\_card, customer\_email; \\
\text{franchise\_address, customer\_email} \\
\rightarrow auto\_pay\_card, customer\_email, franchise\_address;
\end{gathered}
$$

$$\text{franchise\_address, customer\_email} \\ \rightarrow payment\_interval, customer\_email, franchise\_address; \\ \text{franchise\_address, customer\_email} \\ \rightarrow auto\_pay\_card, payment\_interval, customer\_email, franchise\_address\}$$

Claim: This schema is in Third Normal Form
proof
All of the attributes in the table membership are atomic, thus the table is in First Normal Form

Every rule not included in the set above is of the form $U \rightarrow Y$ where $U \supset X$. Each dependency in the set above is either trivial, or the LHS is a superkey of membership. Since all dependencies not included above must have a LHS that is a superset of these superkeys their LHS must also be a superkey. Thus we have that every dependency $X \rightarrow A \in F^+$ must either have the property that X is a superkey of membership, or the dependency is trivial. $\square$

Schema: **franchise(address, contact_phone, contact_email, owner_name)**

From the description we know that each franchise can exist at only one address. With that information we obtain the functional dependency:

$$\{address \rightarrow contact\_phone, contact\_email, owner\_name\}$$

From this dependency we can use Armstrong's Axioms to compute the rules $X \rightarrow Y$ in F^+ such that X is minimal:

$$\{address \rightarrow address; \\ \text{contact\_phone} \rightarrow contact\_phone; \\ \text{contact\_email} \rightarrow contact\_email; \\ \text{owner\_name} \rightarrow owner\_name; \\ \text{address} \rightarrow address, contact\_phone, contact\_email, owner\_name; \\ \text{address} \rightarrow contact\_phone, contact\_email, owner\_name; \\ \text{address} \rightarrow contact\_phone, contact\_email; \\ \text{address} \rightarrow contact\_phone, owner\_name; \\ \text{address} \rightarrow contact\_email, owner\_name; \\ \text{address} \rightarrow contact\_email; \\ \text{address} \rightarrow owner\_name; \\ \text{address} \rightarrow contact\_phone;$$

$\}$ Claim: This schema is in Third Normal Form

proof
All of the attributes in the franchise schema are atomic, so it is in First Normal Form

The above set contains all dependencies $X \rightarrow Y$ such that X is minimal, so by inspection we can see that any other rules in F^+ must either be trivial or

have the property that their LHS is a superset of *address* which we can see is a candidate key, so those rules have the property that their LHS is a superkey of franchise.
□

Schema: **referrals(referred_email, referrer_email, date_referred)**

From the description we can parse that emails are unique for customers of Gold's Gym, and that referrals last for a month. It is not logical to store multiple referrals for the same pair of users. From this information we derive the following set of functional dependencies:

$$\{referred\_email, referrer\_email \rightarrow date\_referred\}$$

Using Armstrong's Axioms we can compute $F+$:

$$\{referred\_email \rightarrow referred\_email;$$
$$\text{referrer\_email} \rightarrow referrer\_email;$$
$$\text{date\_referred} \rightarrow date\_referred;$$
$$\text{referrer\_email, referred\_email} \rightarrow referrer\_email, referred\_email;$$
$$\text{referred\_email, date\_referred} \rightarrow referred\_email, date\_referred;$$
$$\text{date\_referred, referrer\_email} \rightarrow date\_referred, referrer\_email;$$
$$\text{date\_referred, referrer\_email, referred\_email}$$
$$\rightarrow date\_referred, referrer\_email, referred\_email;$$
$$\text{referred\_email, referrer\_email} \rightarrow date\_referred;$$
$$\text{referred\_email, referrer\_email} \rightarrow referrer\_email;$$
$$\text{referred\_email, referrer\_email} \rightarrow referred\_email;$$
$$\text{referred\_email, referrer\_email} \rightarrow date\_referred;$$
$$\text{referred\_email, referrer\_email, date\_referred} \rightarrow date\_referred;$$
$$\text{referred\_email, referrer\_email, date\_referred} \rightarrow referred\_email;$$
$$\text{referred\_email, referrer\_email, date\_referred} \rightarrow referrer\_email;$$
$$\text{referrer\_email, referred\_email, date\_referred}$$
$$\rightarrow referrer\_email, referred\_email;$$
$$\text{referred\_email, referrer\_email, date\_referred} \rightarrow referred\_email, date\_referred;$$
$$\text{referred\_email, date\_referred, referrer\_email} \rightarrow date\_referred, referrer\_email;$$

} Claim: This schema is in Third Normal Form
proof:
All attributes in the referrals table are atomic, so this schema is in First Normal Form

By inspection we can see that all of the dependencies $X \rightarrow A \in F^+$ where A is a single attribute have the property that either they are trivial or $X \supset \{referred\_email, referrer\_email\}$ which we can see is a candidate key for referrals.
□

Schema: **customer_card(card_number, customer_email)**

From the description we parse that a customer is uniquely identified by their email, but each customer could have multiple credit cards. We cannot derive any functional dependencies from the description, so the F^+ on this schema must only have trivial dependencies, and thus since this schema has only atomic attributes it is trivially in Third Normal Form.

Schema: **card_billing(billing_address, card_number)**

Each credit card can only have one billing address. From this information we can derive the following set of functional dependencies:

$$\{card\_number \rightarrow billing\_address;\}$$

From this set of dependencies we can use Armstrong's Axioms to derive all of the dependencies $X \rightarrow A$ such that A is a single attribute of payment_info in F^+:

$$\{billing\_address \rightarrow billing\_address; card\_number \rightarrow card\_number; card\_number \rightarrow billing\_address; card\_number, billing\_address \rightarrow card\_number; card\_number, billing\_address \rightarrow billing\_address;\}$$

Claim: This schema is in Third Normal Form
proof:
All attributes in the card_billing table are atomic, so this schema is in First Normal Form

By inspection we can see that every rule $X \rightarrow AF^+$ is either trivial, or X is a superkey of card_billing.
□