Jeonghyeon Woo

CS415

Assignment 1

1. Facebook is one of big data service. It has data of big volume since many people use it globally. As the input data is big, it has big data volume. The data has big variety. For example, there are many hash tags such as #I_am_groot, and the database stores the value. It has big velocity obviously. The user of Facebook frequently upload, remove, and edit their data, so the change of data is big. If I build the database system of Facebook, then I would use XML since it has DTD method and it is web friendly.
2.
   a. Relation schema is name and attributes of tuples. Ex. Airport <ID, Name, …>

      Attribute is the element of tuples. Ex. ID, Name, City …

      Relation database schema is the structure of database. It includes PK underlined.

      Ex.

      Airport

ID	Name	City	Country	…

      Airline

IATA	ICAO

      Route

ICAO	Latitude	Longitude	Altitude	Timezone

      Domain is valid type of data. Ex. Dataset of Airport must include ID and Name.

      Attribute domain is valid type of data for attribute. Ex. IATA must be 3 letter

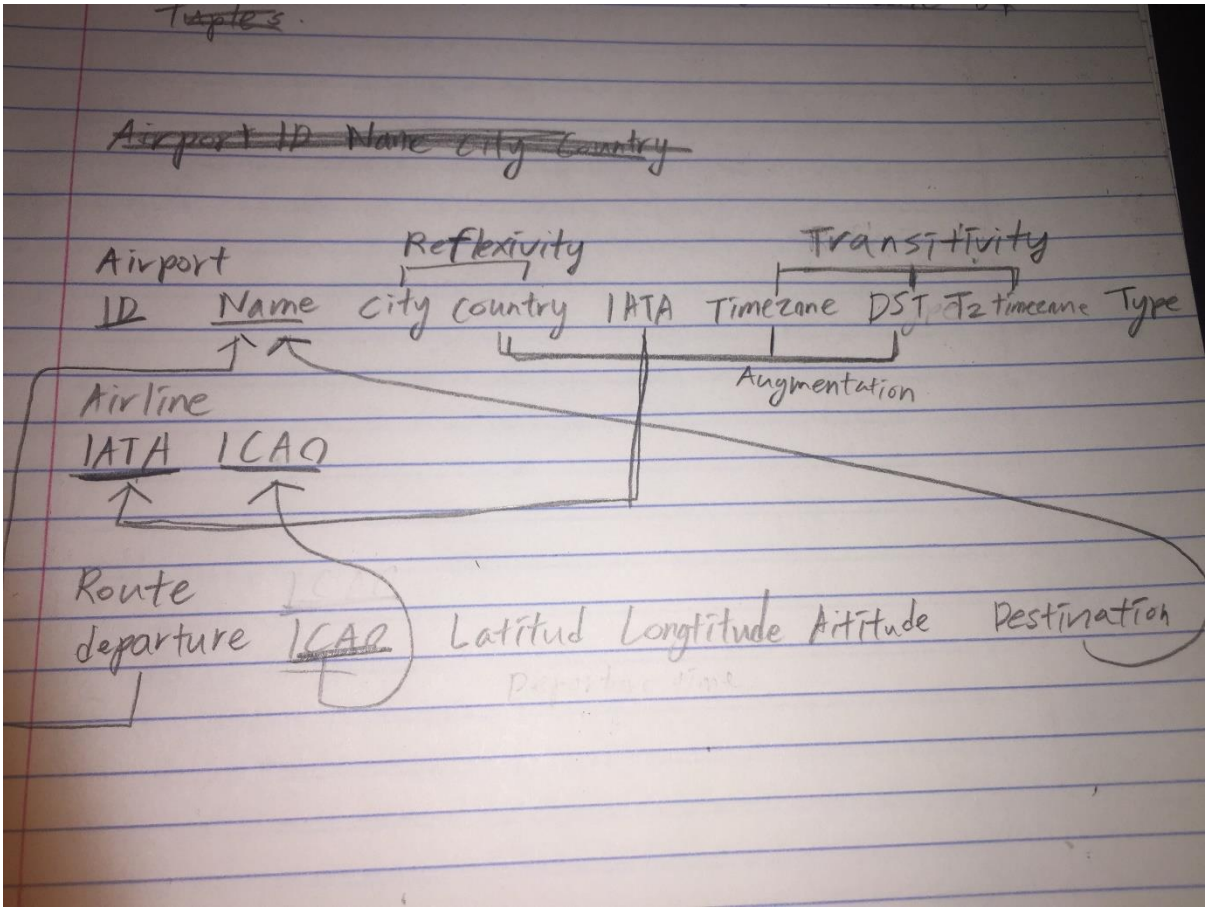
      Relation instance is tuple. Ex. <ID, Name, City…>

      Instance of airport table.

      Airport

ID	Name	City	Country	…
507	London	London	UK	…

	Hathrow			
26	Kugaaruk	Pelly Bay	Canada	…
3127	Pokhara	pokhara	Nepal	…
8810	Hamburg Hbf	Hamburg	Germany	…

b. 

c.

1. $\text{city} \subseteq \text{country}$ so Airport<ID, City, IATA…> for City<city, country>

   Country -> DST so Country*timezone -> DST*timezone, and DST -> Time Zone -> TZ, so Timezone<dst, country, TZ> and TZ<tz, timezone, dst> in Airport<ID, Country, IATA, Timezone>

2. A) $\text{y} \subseteq \text{yz}$ and $\text{z} \subseteq \text{yz}$ by rule of cartesian product

   yz->y and yz->z by reflexivity rule

   x->y and x->z by transitivity rule

   x->yz then x->y and x->z

   B) x->y then xw->yw by Augmentation rule

   xw->yw and yw->z so xw->z by transitivity rule

d. $A_1A_2 \rightarrow A_3$ *and* $A_2A_3 \rightarrow A_4$. As $A_1 \rightarrow A_3$, A_2A_3 are keys of A_4

And $A_2A_3 \rightarrow A_4$ and $A_3A_4 \rightarrow A_1$ As $A_2 \rightarrow A_4$, A_3A_4 are keys of A_1

In schema expression, A4(A_2A3) for A3(A_1, A_2) and A1(A_3,A4) for A4(A_2, A_3).

There are A1 and A4 dataset. Those dataset are normalized in bcnf(well bcnf⊆3nf, so it is also 3nf normalized) since there is no level-2 node in the relation.

3. Q1: $\pi_{\text{Theater}}(\sigma_{\text{title=Zootopia}}\text{Schedule})$

Q2: $\pi_{\text{address,phone number}}(((\sigma_{\text{director=Steven Spielberg}}\text{Movies})\bowtie_{\text{title}}\text{Schedule})\bowtie_{\text{theater}}\text{Location}))$

Q3: $\pi_{\text{address,phone number}}(\sigma_{\text{theater=Le Champo}}\text{Location})$

Q4: $(\rho(C(1 \rightarrow \text{title1},3 \rightarrow \text{title2})\text{Movie}')$-(Movie-$\sigma_{\text{actor}}$(Movie)))$\bowtie_{\text{title}}$

(Movie-σ_{actor}(Movie))

(So in Q4, remove tuples of movie that have same actor as Movie' do, and also vice versa. And then, join them by same title of movie)

4.

<port>

<airport id=507>

<airport name>London Hathrow</name>

<city> London <city>

<country> UK <country>

…

</airport>

<airport id=26>

<airport name> Kugaaruk </airport name>

<city> Pelly Bay </city>

<country> Canada <country>

…

</airport>

…

</port>

b)

```
<Port>
        <airport id=id>
                <name> Name </name>
                <locate> City </locate>
                <IATA> IATA </IATA>
                <tz> TZ timezone </tz>
                <type> Type </type>
        </airport>
        <city>
                <cityname> City </cityname>
                <country> Country </country>
        </city>
        <time>
                <timezone> Timezone </timezone>
                <dst> DST </dst>
                <tz database timezone> TZ timezone </tz databae timezone>
        </time>
        <airline>
                <iata id=IATA> ICAO </iata>
                <icao id=ICAO> ICAO </icao>
        </airline>
        <route id=icao>
                <departure> airport name </departure>
                <latitude> Latitude </latitude>
                <longitude> Longitude </longitude>
                <altitude> Altitude </altitude>
                <destination> airport name </destination>
        <route>
```

<source> Source <source>

</Port>

The key is attribution of element. Ex. Key of route is route.@id

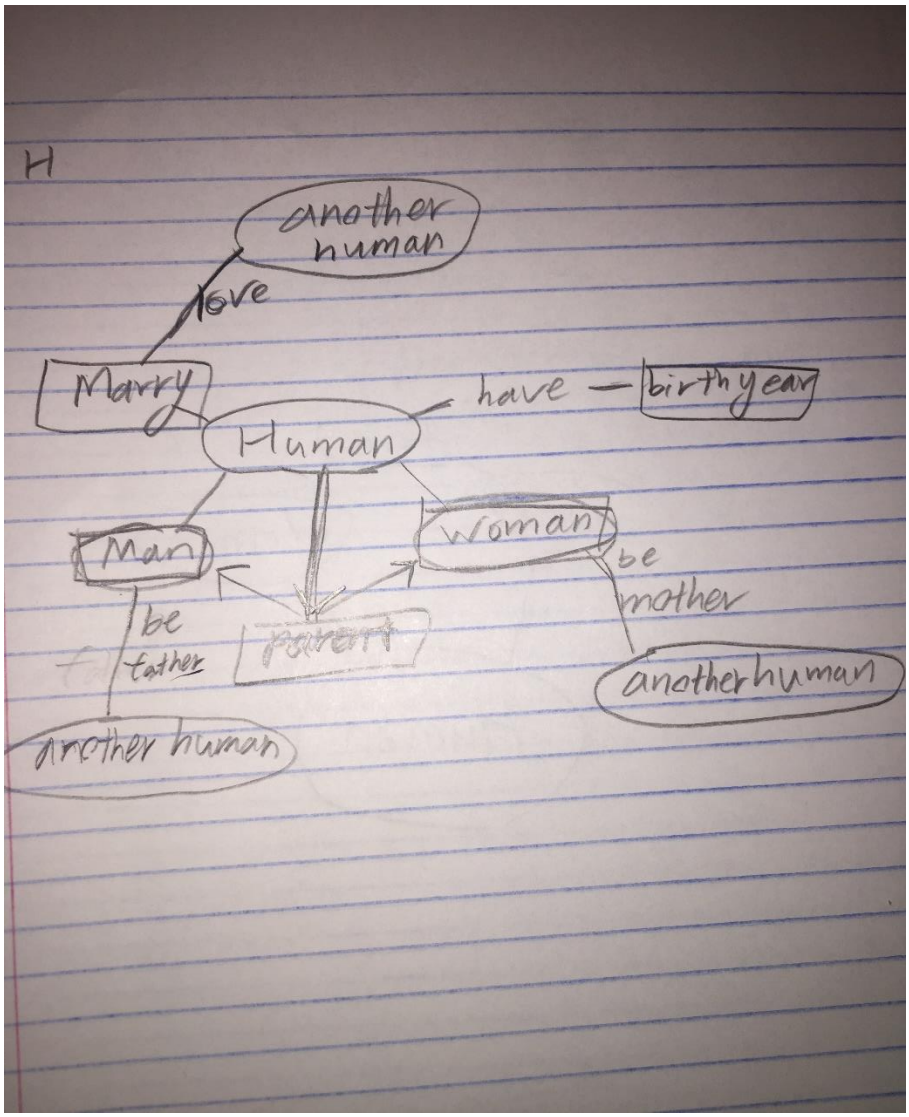
The foreign key is set attribute of element as subset of another element object. Ex. route.@id⊆icao.@id, icao.@id->icao

c)

```
<rdf:RDF>
    <rdf: human>
        <rdf: can>
            <uni: like> another human <uni: like>
            <xs: have birthyear=”xs:Year”/>
            <uni: marry rdf:resource=”human”/>
        </rdf: can>
        <rdf:love rdf:with=marry/>
    </rdf: human>
    <rdf: man>
        <rdfs: subClassOf rdf: resource=”human”/>
        <rdf: can rdf: be=”parent”>
            <uni: father_of rdf:resource=”human”/>
        </rdf: can>
    </rdf: man>
    <rdf: woman>
        <rdfs: subClassOf rdf: resource=”human”/>
        <rdf: can rdf: be=”parent”>
            <uni: mother_of rdf:resource=”human” />
        </rdf:can>
    <rdf:woman>
    <rdf: parent>
        <rdfs: subClassOf rdf: resource=”human”/>
```

<rdf:parent>

</rdf:RDF>



<rdf: RDF>

<rdf:human rdf:id=”mary” rdf:class=”woman” “parent”>

<uni: marry rdf:source=”John”/>

<uni: love rdf:source=”John”/>

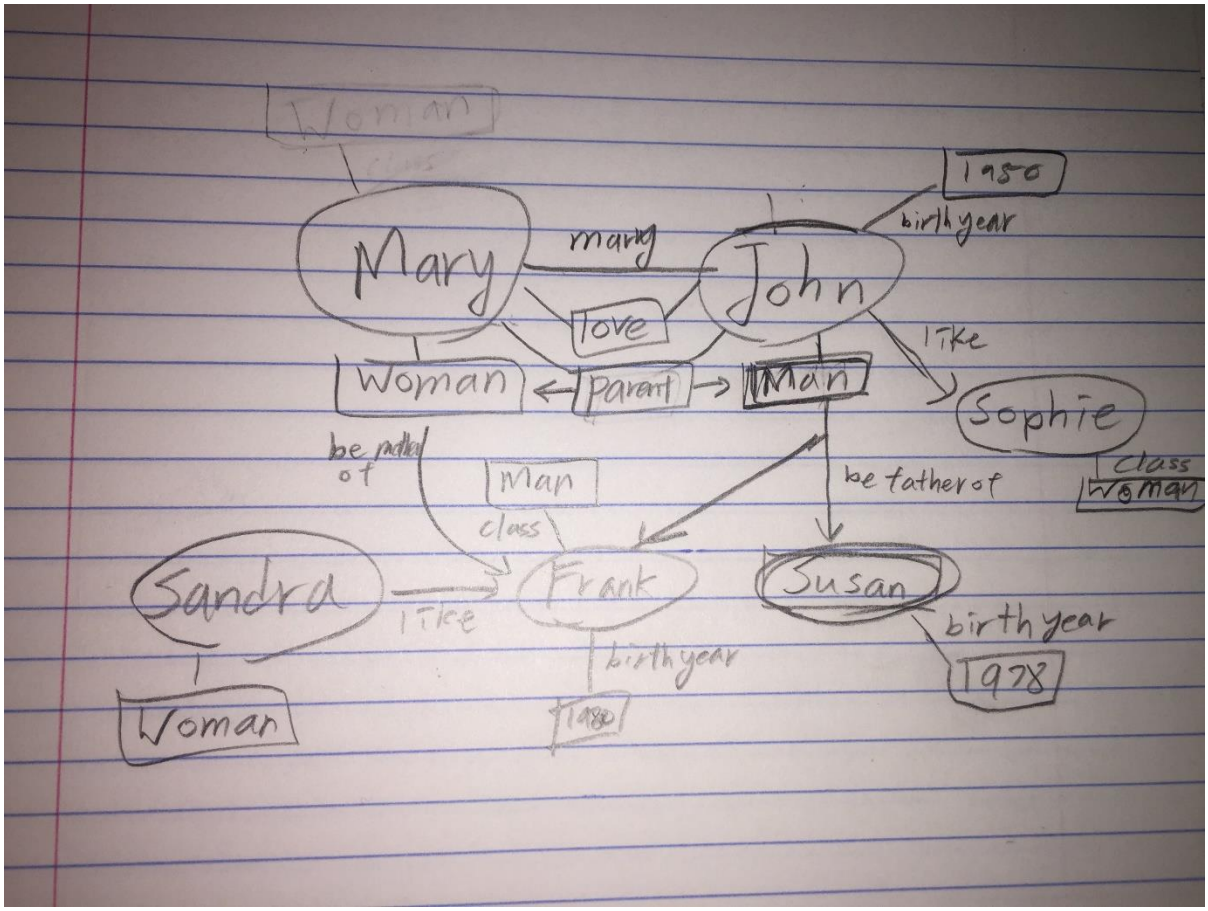
<uni:mother_of rdf:resource=”Frank”/>

</rdf:human>

<rdf:human rdf:id=”John” rdf:class=”man” “parent”>

<uni: marry rdf:source=”Mary”/>

```
        <uni: marry rdf:source=”Mary”/>
        <uni:like> Sophie </uni:like>
        <xs:have birthyear=”xs:1950”/>
        <uni: father_of rdf:source=”Frank” “Susan”/>
      </rdf:human>
      <rdf:human rdf:id=”Sophie” rdf:class=”woman”></rdf:human>
      <rdf:human rdf:id=”Sandra” rdf:class=”woman”>
        <uni:like> Frank </uni:like>
      </rdf:human>
      <rdf:human rdf:id=”Susan” rdf:class=”woman”>
          <xs:have birthyear=”xs:1978”/>
      </rdf:human>
      <rdf: human rdf:id=”Frank” class=”man”>
              <xs:have birthyear=”xs: 1980”/>
      </rdf: human>
</rdf: RDF>
```



With the given schema, we can infer John is man.

father⊆Parent and father->man. John is father, and father is man, so John is man.