CENG352 Written Assignment 1 Solutions

Seda Civelek-2237147

April 2021

1 XML and JSON

1.1 XML

a.
```
<X>
    <A>
        <A>one</A>
        <B>
            <B>two</B>
            <B>three</B>
        </B>
        <C>four</C>
    </A>
    <A>
        <B>
            <A>five</A>
            <A>six</A>
        </B>
        <C>seven</C>
    </A>
</X>
```

b.
i. four, seven
ii. one, four, seven
iii. seven
iv. two, three
v. two, three, five, six
vi. two, three

1.2 JSON

I added the parts that aren't sold by suppliers also in self info but, I made their prices null. This representation doesn't avoid redundancy since all parts are

repeated for each supplier.

```
{
  "suppliers": [
    {
      "sid": 101,
      "sname": "Acme",
      "address": "123 Main",
      "sell_info" : [
        {
          "part": {
            "pid": 90,
            "pname": "bumper",
            "color": "Red"
          },
          "price": null
        },
        {
          "part": {
            "pid": 91,
            "pname": "caliper",
            "color": "Blue"
          },
          "price": null
        },
        {
          "part": {
            "pid":92,
            "pname":"handle",
            "color":"green"
          },
          "price": 5.21
        },
        {
          "part": {
            "pid":93,
            "pname":"gasket",
            "color":"red"
          },
          "price": null
        }
      ]
    },
    {
      "sid": 102,
```

```
  "sname": "Ace",
  "address": "456 Lake",
  "sell_info" : [
    {
      "part": {
        "pid": 90,
        "pname": "bumper",
        "color": "Red"
      },
      "price": null
    },
    {
      "part": {
        "pid": 91,
        "pname": "caliper",
        "color": "Blue"
      },
      "price": null
    },
    {
      "part": {
        "pid":92,
        "pname":"handle",
        "color":"green"
      },
      "price": 6.5
    },
    {
      "part": {
        "pid":93,
        "pname":"gasket",
        "color":"red"
      },
      "price": 65.99
    }
  ]
},
{
  "sid": 103,
  "sname": "Figaro",
  "address": "678 First",
  "sell_info" : [
    {
      "part": {
        "pid": 90,
        "pname": "bumper",
```

```
            "color": "Red"
          },
          "price": null
        },
        {
          "part": {
            "pid": 91,
            "pname": "caliper",
            "color": "Blue"
          },
          "price": null
        },
        {
          "part": {
            "pid":92,
            "pname":"handle",
            "color":"green"
          },
          "price": null
        },
        {
          "part": {
            "pid":93,
            "pname":"gasket",
            "color":"red"
          },
          "price": null
        }
      ]
    }
  ]
}
```


2 Database Design

2.1 BCNF Decomposition

a.

Merge some FDs:
AuthorNo → AuthorName,AuthorEmail,AuthorAdress
AuthorEmail → AuthorNo
PaperNo → FirstAuthorNo,PaperTitle,PaperAbstract,PaperStatus
ReviewerNo → ReviewerName,ReviewerEmail,ReviewerAddress
ReviewerNo,PaperNo → Comments,ProgramComm,ReviewDate,Rating

ReviewerEmail → ReviewerNo

Check AuthorNo → AuthorName,AuthorEmail,AuthorAdress
$(AuthorNo)^+ = AuthorNo, AuthorName, AuthorEmail, AuthorAdress$
LHS is not a superkey, split
$R_1 = (AuthorNo, AuthorName, AuthorEmail, AuthorAdress)$
$R_2 = (PaperNo, FirstAuthorNo, AuthorNo, PaperTitle, PaperAbstract, PaperStatus, ReviewerNo, ReviewerName, ReviewerEmail, Comments, ProgramComm, ReviewDate, Rating, ReviewerAddress)$

Check ReviewerNo → ReviewerName,ReviewerEmail,ReviewerAddress
$(ReviewerNo)^+ = ReviewerNo, ReviewerName, ReviewerEmail, ReviewerAddress)$
LHS is not a superkey, split
$R_{21} = (ReviewerNo, ReviewerName, ReviewerEmail, ReviewerAddress)$
$R_{22} = (PaperNo, FirstAuthorNo, AuthorNo, PaperTitle, PaperAbstract, PaperStatus, ReviewerNo, Comments, ProgramComm, ReviewDate, Rating)$

Check PaperNo → FirstAuthorNo,PaperTitle,PaperAbstract,PaperStatus
$(PaperNo)^+ = PaperNo, FirstAuthorNo, PaperTitle, PaperAbstract, PaperStatus$
LHS is not a superkey, split
$R_{221} = (PaperNo, FirstAuthorNo, PaperTitle, PaperAbstract, PaperStatus)$
$R_{222} = (PaperNo, AuthorNo, ReviewerNo, Comments, ProgramComm, ReviewDate, Rating)$

Check ReviewerNo,PaperNo → Comments,ProgramComm,ReviewDate,Rating
$(ReviewerNo, PaperNo)^+ = PaperNo, ReviewerNo, Comments, ProgramComm, ReviewDate, Rating$
LHS is not a superkey, split
$R_{2221} = (PaperNo, ReviewerNo, Comments, ProgramComm, ReviewDate, Rating)$
$R_{2222} = (PaperNo, AuthorNo, ReviewerNo)$

b.

Since the relations are all in BCNF, decomposition is lossless.
$AuthorNo \rightarrow AuthorName, AuthorEmail, AuthorAdress$ and
$AuthorEmail \rightarrow AuthorNo$ are FDs' of R_1.
$PaperNo \rightarrow FirstAuthorNo, PaperTitle, PaperAbstract, PaperStatus$ is FD's
of R_{221}.
$ReviewerNo \rightarrow ReviewerName, ReviewerEmail, ReviewerAddress$ and $ReviewerEmail \rightarrow ReviewerNo$ are FDs' of R_{21}.
$ReviewerNo, PaperNo \rightarrow Comments, ProgramComm, ReviewDate, Rating$
is FD's of R_{2221}.
All FDs are preserved.

Author			
AuthorNo(PK)	AuthorName	AuthorEmail	AuthorAdress

Reviewer			
ReviewerNo(PK)	ReviewerName	ReviewerEmail	ReviewerAddress

Paper				
PaperNo(PK)	FirstAuthorNo	PaperTitle	PaperAbstract	PaperStatus

Review					
ReviewerNo(PK,FK)	PaperNo(PK,FK)	Comments	ProgramComm	ReviewDate	Rating

Conference		
PaperNo(PK,FK)	AuthorNo(PK,FK)	ReviewerNo(PK,FK)

Figure 1: New Database Schema(PK:primary key, FK:foreign key)

2.2 3NF Decomposition

a.

Step 1: AC $\rightarrow$ BGH turns into: AC $\rightarrow$ B , AC $\rightarrow$ G , AC $\rightarrow$ H
E $\rightarrow$ FK turns into: E $\rightarrow$ F , E $\rightarrow$ K
H $\rightarrow$ BGH turns into: H $\rightarrow$ B , H $\rightarrow$ G , H $\rightarrow$ H
D $\rightarrow$ E
G $\rightarrow$ B
FD $\rightarrow$ K
ADF $\rightarrow$ C

Step 2: For AC $\rightarrow$ BGH:
$(C)^+ = C$, C can't be removed
$(A)^+ = A$, A can't be removed
For FD $\rightarrow$ K:
$(F)^+ = F$, D can't be removed
$(D)^+ = DEFK$, F can be removed
For ADF $\rightarrow$ C:
$(DF)^+ = DEFK$, A can't be removed
$(AF)^+ = AF$, D can't be removed
$(AD)^+ = ADEFKC$, F can be removed
New FDs:

AC → B
AC → G
AC → H
E → F
E → K
H → B
H → G
H → H
D → E
G → B
D → K
AD → C

Step 3: For AC → B:
$(AC)^+ = ACGHB$, B can be removed
For AC → G:
$(AC)^+ = ACGHB$, G can be removed
For AC → H:
$(AC)^+ = AC$, H can't be removed
For D → E:
$(D)^+ = DK$, E can't be removed
For G → B:
$(G)^+ = G$, B can't be removed
For E → F:
$(E)^+ = EK$, F can't be removed
For E → K:
$(E)^+ = EF$, K can't be removed
For D → K:
$(D)^+ = DEFK$, K can be removed
For AD → C:
$(AD)^+ = ADEFDK$, C can't be removed
For H → B:
$(H)^+ = HGB$, B can be removed
For H → G:
$(H)^+ = H$, G can't be removed
For H → H:
$(H)^+ = HGB$, H can be removed

Minimal Cover is:
min_cover = {D → E , AC → H , E → F , E → K , H → G , AD → C , G → B }

b.

Minimal cover is:
U = {D → E , AC → H , E → F , E → K , H → G , AD → C , G → B }

Partition U into sets:
$U_1 = \{D \rightarrow E\}, U_2 = \{AC \rightarrow H\}, U_3 = \{E \rightarrow F, E \rightarrow K\}, U_4 = \{H \rightarrow G\}, U_5 = \{AD \rightarrow C\}, U_6 = \{G \rightarrow B\}$

For each U_i, schema $R_i = (R_i, U_i)$:
$R_1 = (DE : D \rightarrow E)$
$R_2 = (ACH : AC \rightarrow H)$
$R_3 = (EFK : E \rightarrow F, E \rightarrow K)$
$R_4 = (HG : H \rightarrow G)$
$R_5 = (ADC : AD \rightarrow C)$
$R_6 = (GB : G \rightarrow B)$
R_5 is a superkey for R. Decomposition is lossless.

2.3 Finding Dependencies

a.

Bad FDs I found to use them in BCNF decompositions are:
$B \rightarrow A$
$D \rightarrow C$
$G \rightarrow F$

```
select count(distinct f)
from sample
group by g;

select count(distinct c)
from sample
group by d;

select count(distinct a)
from sample
group by b;
```

Since, B, D and G are not a key, we can use them to do BCNF decomposition.

b.

$R_1 = (A, B)$
$R_{21} = (C, D)$
$R_{221} = (F, G)$

$R_{222} = (B, D, E, G)$

```
create table if not exists BA (
        B varchar,
        A varchar,
        primary key(B)
);

create table if not exists DC (
        D varchar,
        C int,
        primary key(D)
);
create table if not exists GF(
        G varchar,
        F int,
        primary key(G)
);
create table if not exists BDEG(
        B varchar,
        D varchar,
        E int,
        G varchar,
        primary key(B,D,E,G),
        foreign key (B) references BA(B),
        foreign key (D) references DC(D),
        foreign key (G) references GF(G)
);
```

c.

```
insert into BA
select B,A from sample group by B,A;

insert into DC
select D,C from sample group by D,C;

insert into GF
select  G,F from sample group by G,F;

insert into BDEG
select B,D,E,G from sample group by B,D,E,G;
```

3 SQL DDL

3.1 Create Table

```
create table Product(
        ProdNo int not null,
        ProdName varchar,
        ProdPrice int,
        ProdShipDate timestamp,
        primary key(ProdNo)
);
create table Customer (
        CustNo int not null,
        CustFirstName varchar,
        CustLastName varchar,
        CustCity varchar,
        CustState varchar,
        CustZip varchar,
        CustBal varchar,
        primary key (CustNo)
);
create table Employee (
        EmpNo int not null default 007,
        EmpFirstName varchar,
        EmpLastName varchar,
        EmpPhone varchar,
        EmpEmail varchar,
        EmpDeptName varchar,
        EmpStatus varchar,
        EmpSalary int,
        supervisor int,
        primary key (EmpNo),
        foreign key (supervisor) references Employee (EmpNo)
                                    on delete set default
);
create table Order (
        OrdNo int not null,
        CustNo int not null,
        EmpNo int,
        OrdDate timestamp,
        OrdName varchar,
        OrdCity varchar,
        OrdZip varchar,
        primary key(OrdNo),
        foreign key(CustNo) references Customer(CustNo)
                            on delete cascade,
```

```
        foreign key(EmpNo) references Employee(EmpNo)
                           on delete set null
);

create table Contains(
        OrdNo int not null,
        ProdNo int not null,
        Qty int,
        primary key(OrdNo,ProdNo),
        foreign key(OrdNo) references "Order"(OrdNo)
                           on delete cascade,
        foreign key(ProdNo) references Product(ProdNo)
                           on delete cascade
);
```

3.2 SQL Check

```
alter table Contains add constraint prodCheck check (Qty>=3);

alter table Order add constraint cityCheck check (OrdName like
concat('%',OrdCity,'%'));


alter table Employee add constraint emailCheck check(EmpEmail not like
concat('%',EmpFirstName,'%') and EmpEmail not like
concat('%',EmpLastName,'%'));
```

3.3 Assertion

```
Create assertion NoOrderProductLess check(not exists(
        select sum(c.Qty)
        from Contains c,Order o
        where c.OrdNo = o.OrdNo
        group by c.OrdNo
        having sum(c.Qty) < 30
    )
);
```

3.4 Trigger

```
create trigger UpdateStatus
    after update of EmpSalary on Employee
```

```
referencing old row as o
            new row as n
for each row
when(n.EmpSalary-o.EmpSalary > 0.15*o.EmpSalary)
    update Employee
    set EmpStatus = 'Successful'
    where EmpNo = o.EmpNo
```