Databases Project – Spring 2021

Team No: 34

Members: Siran Li, Ke Wang, Ningwei Ma

Contents

Contents 1

Deliverable 1 2

Assumptions 2

Entity Relationship Schema 3

Schema 3

Description 4

Relational Schema 6

ER schema to Relational schema 6

DDL 7

General Comments 11

# Deliverable 1

## Assumptions

Each collision case has its unique case\_id.

Each case has a primary collision factor (pcf); Different cases could have the same pcf.

Each case has several parties.

Each case collides in exactly one location; Different cases could collide in same location.

Each collision happened under exactly one condition, among which there could be several weather and road conditions; Different cases could collide under same condition.

Each party has a unique party\_id.

Each party is involved in exactly one case; A case could involve several parties.

Each party may take a vehicle; Vehicles with same attributes are recognized as the same vehicle, and under this condition different parties can take the same vehicle.

“Party\_number” refers to the specific party of a particular case, so “party\_number + case\_id” is unique for each party, playing the same role as party\_id.

Each victim has a unique vic\_id.

Each victim is associated with exactly a party in a case, by “party\_number + case\_id”. A party could be associated with 0 or several victims.

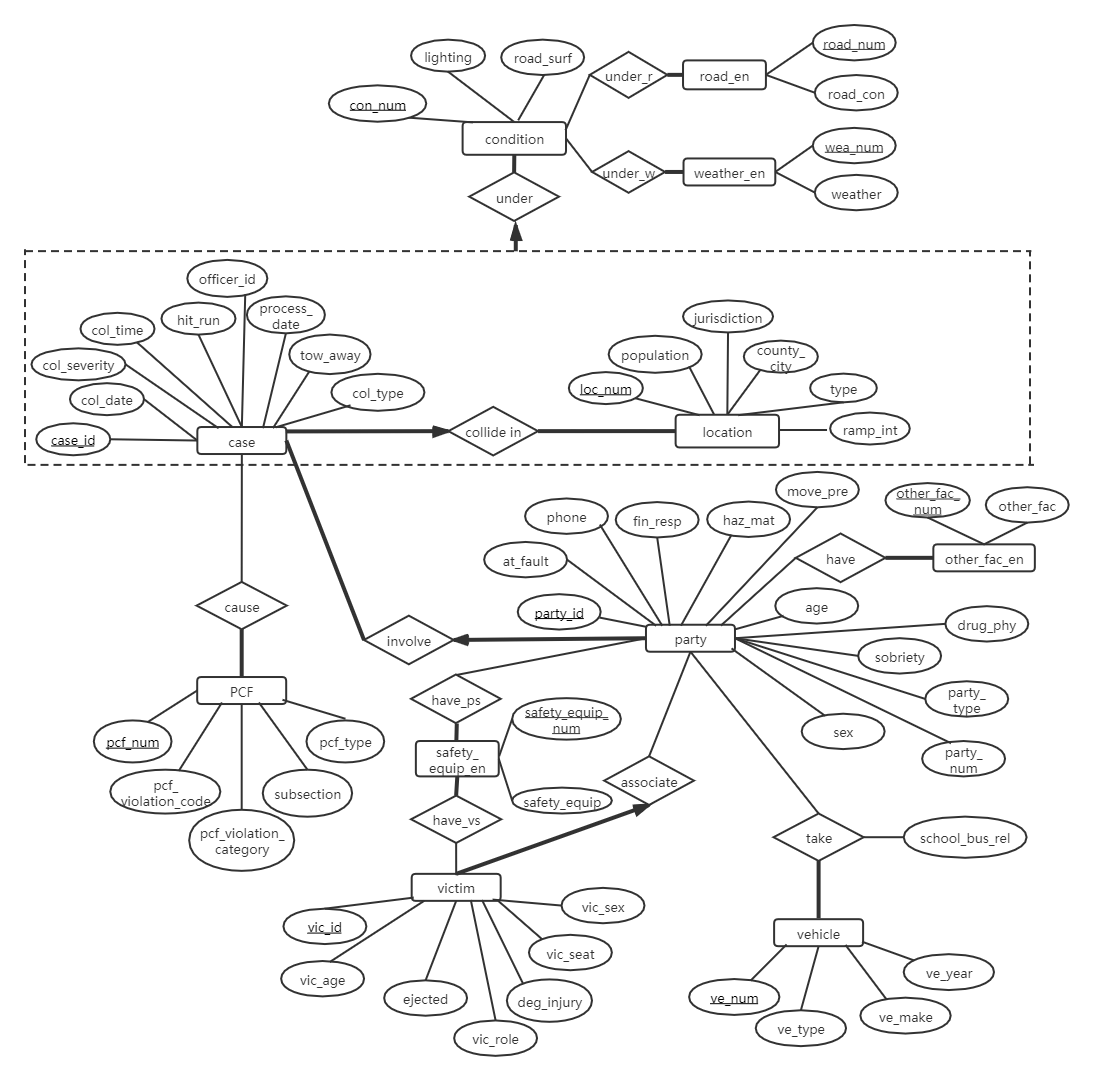
Each party of the case may have some other factors for the collision. Different parties may have same other factors.

Each party and victim may have their different safety\_equipment. Different parties and victims may have same safety\_equipment.

More detailed reasons will be given in the description.

## Entity Relationship Schema

### Schema



### Description

**Entities:** case (collision case), location, PCF (primary collision factor), party, victim, vehicle, condition, weather\_en, road\_en (road\_condition), safety\_equip\_en (safety\_equipment), other\_fac\_en (other\_associated\_factor)

Each case has some information: collision\_date, collision\_severity, collision\_time, hit\_and\_run, officier\_id, process\_date, tow\_away (if towed away or not) and col\_type (type\_of\_collision). We set them as the attributes of the “case” entity, because they are some basic information miscellaneous of the case itself, which are easy to be analyzed as attributes. Each case has a unique case\_id, so we set it as the primary key.

Each case collides in some location, including the following information: county\_city (county\_city\_location), type (location\_type), jurisdiction, population, ramp\_int (ramp\_intersection). We set these as the attributes of the entity “location”, because creating an entity to integrate them together, we may obtain clearer information about the “location information”. For each location, we add a primary key “loc\_num”.

Each collision (a case collide in a location) happens under certain condition: lighting, road\_surf (road\_surface), different weather and different road\_condition. For “weather” and “road\_con”, we set them as another two entities and give them unique keys, because it is troublesome to have weather\_1, weather\_2 as different attributes of condition, and now we can easily add new instances of this kind of entities through a ‘one-to-many’ relationship. Then, “condition” keeps an “under\_r” relationship with “road\_en” and an “under\_w” relationship with “weathe\_enr”, “road\_en” and “weather\_en” have their own primary key “road\_num” and “wea\_num”. For each condition, we add a primary key “con\_num”.

We use aggregation to aggregate case entity, collide\_in relationship, location entity, so that we can treat the relationship as an entity and implement the constraint that “case collide in location” relationship is under exactly one “condition”.

Each case has a PCF (primary collision factor), which includes: pcf\_type (primary\_collision\_factor), subsection (pcf\_violation\_subsection), pcf\_violation\_category, pcf\_violation\_code. For each PCF, we add a primary key “pcf\_num”.

Each case involves parties. Party is an entity, with primary key “party\_id”. Each party can only be involved with one case, because there are no unique person identifier, and so even though one person is involved in different cases, due to the different attributes of the party entity, he/she is regarded as different parties in the data. Each party has: at\_fault, phone (cellphone\_use), fin\_resp (financial\_responsibility), haz\_mat (hazardous\_materials), move\_pre (movement\_preceding\_collision), age (party\_age), drug\_phy (party\_drug\_physical), sobriety, party\_type, party\_num (party\_number), sex (party\_sex) as its attributes. For “other\_fac\_en (other\_associated\_factor)”, we set it as an entity for the same reason with “weather\_en” entity. “Party” and “other\_fac\_en” have a “have” relationship. For “party\_safety\_equipment”, we also set a “safety\_equip\_en” entity with primary key “safety\_equip\_num”, which has a “have\_ps” relationship with “party”.

Each party may take a vehicle. Vehicle is an entity with a created primary key “ve\_num”. Each vehicle has: ve\_year (vehicle\_year), ve\_make (vehicle\_make), ve\_type (statewide\_vehicle\_type) as its attributes. “Party” and “vehicle” has a “take” relationship, which has an attribute “school\_bus\_rel”, indicating whether the vehicle is used as a school bus in this relationship. Vehicles with same “type, make and year” are recognized as a same vehicle in our assumption, so it is not an weak entity of party.

Each party may be associated with some victims. Victim is an entity, with primary key “vic\_id”. As stated before, victim is associated with a party by “case\_id + party\_num” (which is the same as “party\_id”), so we will use ‘party\_id’ as a foreign key to reference party. Victim has: vic\_age (victim\_age), ejected (victim\_ejected), vic\_role (victim\_role), deg\_injury (victim\_degree\_of\_injury), vic\_seat (victim\_seating\_position), vic\_sex (victim\_sex) as its attributes. For “victim\_safety\_equipment”, we use “safety\_equip\_en” entity and give it a “have\_vs” relationship with victim.

**Constraints:**

A case collides in exactly one location, and a collision happens under exactly one condition. So, “case – collide in” and “collide in – under” relationship both have a “exactly one” constraint. On the contrary, different cases can have same “location” and “condition”. And each instance of “location” and “condition” have to appear in one case, otherwise they won’t be recorded and appear in the entity set. Therefore, there is total participation but no “at most one” constraint for “condition – under” and “location – collide in”.

For the same reason, “road\_con\_en”, “weather\_en”, “other\_fac\_en”, “vehicle”, “safety\_equip\_en” and “PCF” all have a total participation in the relationship, otherwise they won’t appear in the set. However, the entities that have such relationships with these entities may have “null” value in the raw data, so there is no constraint on the other side of the relationship.

A case at least involves one party. From the raw data we could find some case\_ids in “collision.csv” that do not appear in the “parties.csv”, but we believe it is due to missing data but not the real situation, so we add this total participation constraint. Each party is involved in exactly one case. From the raw data, we could find that almost every party has its unique ‘case\_id’, ‘party\_num’ pair, and only 8 parties have duplicated data pairs. Compared to the large database,we think that these duplicates are due to error logging. The forum also tells us that there is no “person” entity to correspond to different cases. So “party” has exactly one corresponding case in this “involve” relationship.

For the same reason, a victim\_id is generated if there are some victims associated with a party. We can also tell from the raw data that “case\_id + party\_num” of a victim is unique as “party\_id”. So a victim is associated with exactly one party with “case\_id + party\_num” or “party\_id”.

## Relational Schema

### ER schema to Relational schema

1. We start from the entity “case”. Create a table “Case”. It has primary key “case\_id”. Domain of case\_id, officer\_id and tow\_away is integer, domain of col\_date is date, and for others is variable character string.

2. From the analysis above, we know a party is involved in exactly one case. So we merge the relationship “involve” with party to implement this constraint, and create a table “Party\_involve”. “Party\_id” is the primary key. Meanwhile, we need a foreign key to refer to the “case\_id” of TABLE Case, and “case\_id” cannot be null. Party\_id, case\_id, at\_fault and party\_num should be integer, and others are variable character string.

3. For the same reason, we also merge victim with “associate” relationship into one table. Create a table “Associate\_victim”, with the primary key “vic\_id” and foreign key “party\_id”. By our assumption, party\_id uniquely corresponds to “party\_num+case\_id”, so here we directly use party\_id as foreign key, and “party\_id” cannot be null.

4. For “other\_fac\_en” entity, we create an independent table “Other\_fac\_en” for it, with key “other\_fac\_num” and a description string. We also create table “Have” for “party have other\_fac\_en” relationship. It has unique (other\_fac\_num, party\_id) pairs, and has two foreign keys that refers to “other\_fac\_num” of “Other\_fac\_en” table and “party\_id” of “Party\_involve”.

5. We also create an independent table for “Safety\_equip\_en” for safety equipment of parties and victims, with key “safety\_equip\_num” and a description string. We create table “Have\_ps” and table “Have\_vs” to demonstrate the relationship “party\_have\_safety\_equip” and “victim\_have\_safety\_equip”. They have integer party\_id/vic\_id and safety\_equip\_num as foreign key, the (party\_id/vic\_id, safety\_equip\_num) pairs as primary keys.

6. We create an independent table for “Vehicle”. With a primary key “ve\_num” to denote vehicle with different “type, make, yead”. “Party take vehicle” relationship is signified by table “Take”. This table has “ve\_num”, “party\_id” as foreign keys and the tuples of these two are unique.

7. We create an independent table for “PCF”, which has the attributes of PCF entity, with a primary key “pcf\_num”. “PCF causes case” relationship is signified by table “Cause”. Each entry has “pcf\_num”, “case\_id” as foreign keys. (pcf\_num, case\_id) pairs are unique.

.

8. Location is signified by the table “Location”. It has the attributes of location entity, and uses integer “loc\_num” as its primary key.

9. Condition is signified by the table “Condition”. It has the attributes of condition entity, and uses integer “con\_num” as its primary key.

10. Location, case and condition can be connected by a table “Under”. We use integer “case\_id” as the primary key, to enforce the constraint that each case must collide in exactly one location, and meanwhile enforces that the “case collides in location” relationship is under exactly one condition (case\_id is enough to uniquely identify the collide\_in relationship). The table also has “loc\_num” and “con\_num” as its foreign key, which cannot be null because of the total participation of “case”.

11. We create an independent table “Road\_en”, with a primary key “road\_num” and “road\_con” description. “condition under road\_en” relationship is denoted by table “Under\_r”. Each entry has “road\_num”, “con\_num” as foreign keys. (road\_num, con\_num) pairs are unique.

12. We create an independent table “Weather\_en”, which has the attributes of “weather\_en” entity, with a primary key “wea\_num” and “weather” description. “condition under weather\_en” relationship is denoted by table “Under\_w”. Each entry has “wea\_num”, “con\_num” as foreign keys. (wea\_num, con\_num) pairs are unique.

### DDL

CREATE TABLE Case(

case\_id INTEGER

loc\_num INTEGER

col\_date DATE

col\_severity VARCHAR2(30)

col\_time VARCHAR2(10)

hit\_run VARCHAR2(30)

officer\_id INTEGER

process\_date DATE

tow\_away INTEGER

col\_type VARCHAR2(30)

PRIMARY KEY (case\_id)

);

CREATE TABLE Party\_involve(

party\_id INTEGER

case\_id INTEGER NOT NULL

at\_fault INTEGER

phone VARCHAR2(3)

fin\_resp VARCHAR2(3)

haz\_mat VARCHAR2(3)

move\_pre VARCHAR2(3)

age INTEGER

drug\_phy VARCHAR2(3)

sobriety VARCHAR2(3)

party\_type VARCHAR2(15)

party\_num INTEGER

sex VARCHAR2(6)

PRIMARY KEY (party\_id)

FOREIGN KEY (case\_id) REFERENCES (Case)

);

CREATE TABLE Associate\_victim(

vic\_id INTEGER

party\_id INTEGER NOT NULL

vic\_age INTEGER

ejected INTEGER

vic\_role INTEGER

deg\_injury VARCHAR2(50)

vic\_seat INTEGER

vic\_sex VARCHAR2(6)

PRIMARY KEY (vic\_id)

FOREIGN KEY (party\_id) REFERENCES (Party\_involve)

);

CREATE TABLE Other\_fac\_en(

other\_fac\_num INTEGER

other\_fac VARCHAR2(3)

PRIMARY KEY (other\_fac\_num)

);

CREATE TABLE Have (

other\_fac\_num INTEGER

party\_id INTEGER

PRIMARY KEY (other\_fac\_num, party\_id)

FOREIGN KEY (party\_id) REFERENCES (Party\_involve)

FOREIGN KEY (other\_fac\_num) REFERENCES (Other\_fac\_en)

);

CREATE TABLE Safety\_equip\_en(

safety\_equip\_num INTEGER

safety\_equip VARCHAR2(3)

PRIMARY KEY (safety\_equip\_num)

);

CREATE TABLE Have\_ps(

party\_id INTEGER

safety\_equip\_num INTEGER

PRIMARY KEY (party\_id, safety\_equip\_num)

FOREIGN KEY (party\_id) REFERENCES (Party\_involve)

FOREIGN KEY (safety\_equip\_num) REFERENCES (Safety\_equip\_en)

);

CREATE TABLE Have\_vs(

vic\_id INTEGER

safety\_equip\_num INTEGER

PRIMARY KEY (vic\_id, safety\_equip\_num)

FOREIGN KEY (vic\_id) REFERENCES (Associate\_victim)

FOREIGN KEY (safety\_equip\_num) REFERENCES (Safety\_equip\_en)

);

CREATE TABLE Vehicle(

ve\_num INTEGER

ve\_type VARCHAR2(50)

ve\_make VARCHAR2(20)

ve\_year INTEGER

PRIMARY KEY (ve\_num)

);

CREATE TABLE Take(

ve\_num INTEGER

party\_id INTEGER

school\_bus\_rel VARCHAR2(5)

PRIMARY KEY (ve\_num, party\_id)

FOREIGN KEY (party\_id) REFERENCES (Party\_involve)

FOREIGN KEY (ve\_num) REFERENCES (Vehicle)

);

CREATE TABLE PCF(

pcf\_num INTEGER

pcf\_violation\_code INTEGER

pcf\_violation\_category VARCHAR2(50)

subsection VARCHAR2(3)

pcf\_type VARCHAR2(50)

PRIMARY KEY (pcf\_num)

);

CREATE TABLE Cause(

pcf\_num INTEGER

case\_id INTEGER

PRIMARY KEY (pcf\_num, case\_id)

FOREIGN KEY (pcf\_num) REFERENCES (PCF)

FOREIGN KEY (case\_id) REFERENCES (Case)

);

CREATE TABLE Location(

loc\_num INTEGER

population INTEGER

jurisdiction INTEGER

county\_city INTEGER

type VARCHAR2(20)

ramp\_int VARCHAR2(10)

PRIMARY KEY (loc\_num)

);

CREATE TABLE Condition(

con\_num INTEGER

lighting VARCHAR2(50)

road\_surf VARCHAR2(10)

PRIMARY KEY (con\_num)

);

CREATE TABLE Under(

case\_id INTEGER

loc\_num INTEGER NOT NULL

con\_num INTEGER NOT NULL

PRIMARY KEY (case\_id)

FOREIGN KEY (case\_id) REFERENCES Case

FOREIGN KEY(loc\_num) REFERENCES Location

FOREIGN KEY(con\_num) REFERENCES Condition

);

CREATE TABLE Road\_en(

road\_num INTEGER

road\_con VARCHAR2(20)

PRIMARY KEY (road\_num)

);

CREATE TABLE Under\_r(

con\_num INTEGER

road\_num INTEGER

PRIMARY KEY (road\_num, con\_num)

FOREIGN KEY (road\_num) REFERENCES (Road\_en)

FOREIGN KEY (con\_num) REFERENCES (Condition)

);

CREATE TABLE Weather\_en(

wea\_num INTEGER

weather\_con VARCHAR2(20)

PRIMARY KEY (wea\_num)

);

CREATE TABLE Under\_w(

con\_num INTEGER

wea\_num INTEGER

PRIMARY KEY (wea\_num, con\_num)

FOREIGN KEY (wea\_num) REFERENCES (Weather\_en)

FOREIGN KEY (con\_num) REFERENCES (Condition)

);

## General Comments

From the 3 raw tables, we derived 11 entities by learning the relationships between different fields, some of which are of the similar type and some are to deal with multiple values. We have tried our best to create a rigorous, elegant and efficient model, but we also have some confusion about some constraints and a better way to integrate attributes, entities and relationships.

Rough work allocation:

Siran Li: ER model

Ke Wang: Relation schema & DDL

Ningwei Ma: Report