DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

Databases Project – Spring 2021

Team No:

Members:

Contents

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

Deliverable 1

Assumptions

On Identification:

Every party number should be unique within a collision. Every party_id, victim_id, case_id should be unique by its own within the corresponding .csv files.

On data:

We assumed that in the .csv files every field would be represented by its key or that we would make it so during the data cleaning phase. We assumed that every description could fit in 150 char. We assumed based on data that party_id, victim_id and case_id can be typed as integer.

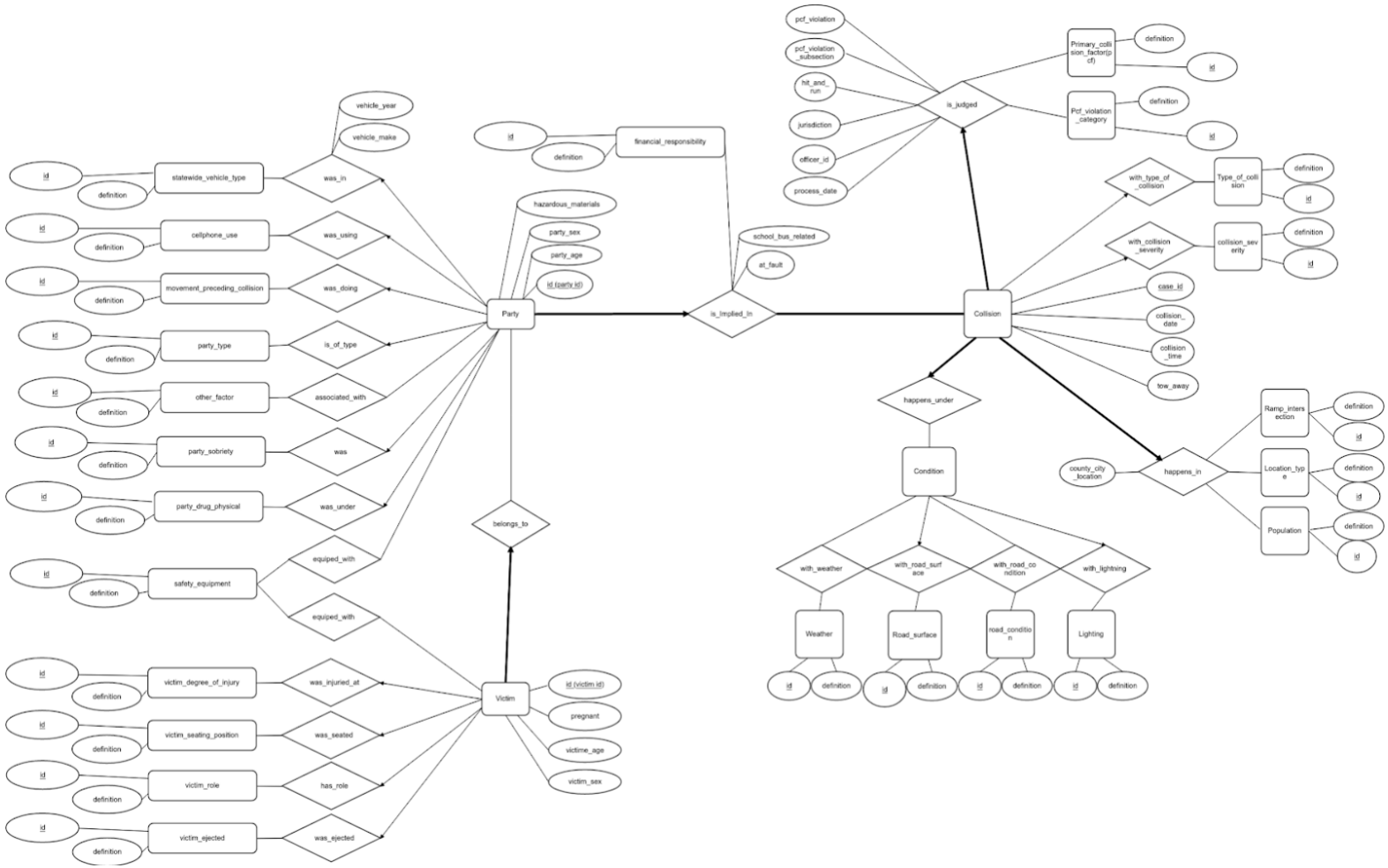
On integrity:

Every victim should be associated with an unique party. Every party should be implicated in a unique collision.

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

Entity Relationship Schema

Schema



DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

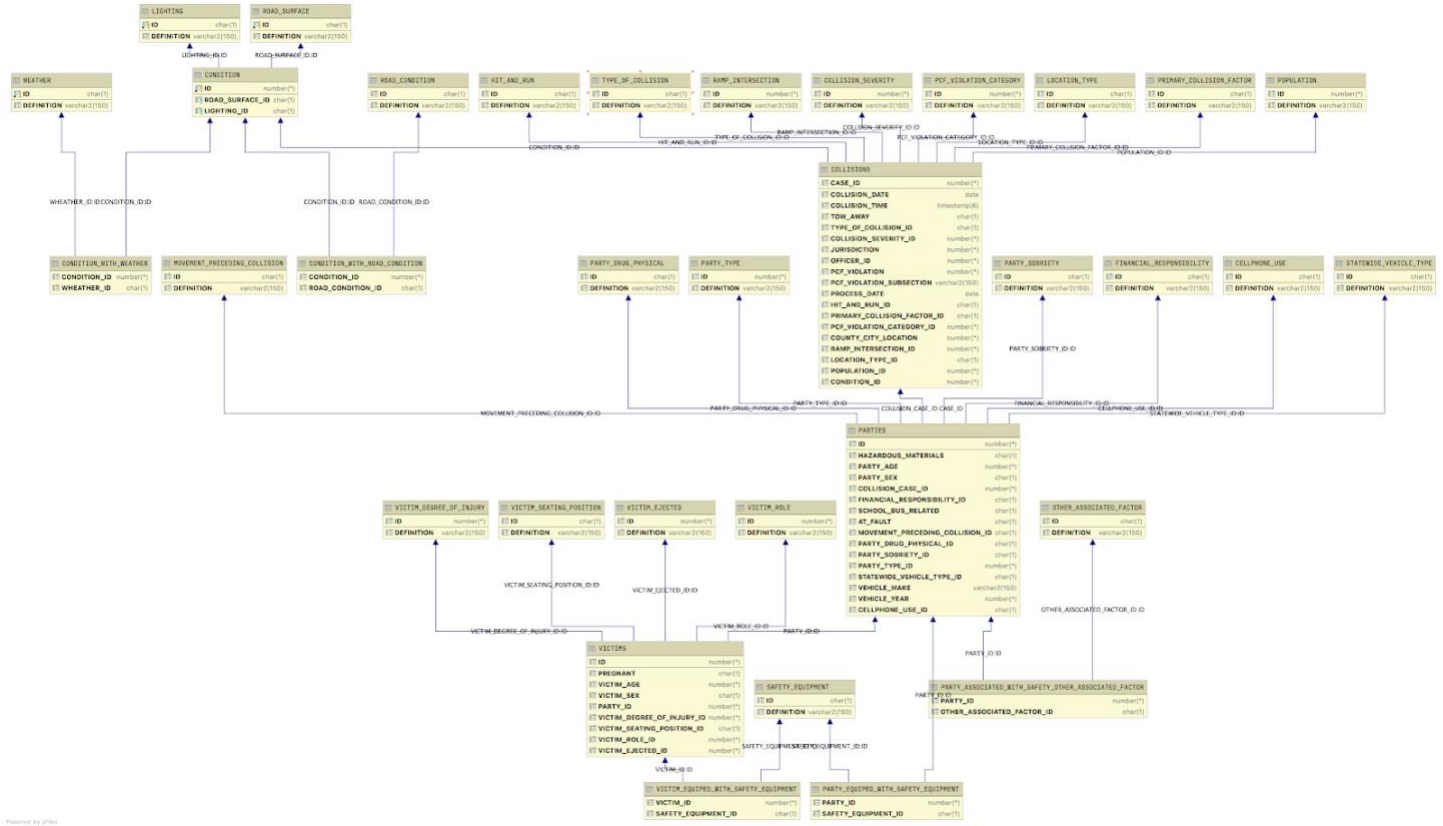
Description

For the ER diagram, we first decided to divide the attributes into 3 main entities called Victim, Party and Collision, because it seemed to us that they were the main actors in the model.
Then, we saw that it didn't make much sense to have only these 3 entities, because some attributes wouldn't be logically attributed to them. For example, it wouldn't make sense that a collision has an attribute population, because they are not directly correlated. Therefore, we tried to group attributes that logically belonged to a common idea together (star schema). For the collisions, we saw that there were many attributes related to the location of the collision, the conditions under which the collision happened and the legal part related to the collision. For the parties, many attributes were related to the vehicle. Hence, we wanted to add these 4 entities to our diagram (but finally modified it slightly, see below).
Also, after we spoke with some assistants, we realised that it would be a good idea to create entities for attributes that are lists with some finite non-logically predefined values (A:..., B:...). The reasons are the following: it would be easier to enforce the data we store to be cleaned and in the same format (it avoids to have one time 'a' and one time 'A' referencing to the same value) and it would make it more modulable and easier to change (if we realize that we would like to add/remove an option, we could simply add/remove one row in the table of the entity and add/invalidate these entries in the other table).
When there were many times the same attribute in the csv files (..._1 and ..._2), we also decided to create an entity. This has the advantage to be more modulable, since we could decide to add a third (..._3) attribute or even more of them in the future if we would like to slightly change the model. For that, we simply allowed the relation to have many of these new entities.
Finally, when we wanted to merge all our previous ideas together to construct the diagram, we found that creating the 4 entities mentioned above was not really practical because we would have to create these entities which now have no (or not many) attributes (since their corresponding attributes were often lists which we now model with an entity and bind through a relation), which makes them almost useless and increases the complexity of the diagram. Therefore, we decided to create N-ary relations directly to group the collision and all the attributes related to a given theme. This seems easier to understand and will create the same result in the database (since every attribute will finally be stored in the Collision table after the merging due to the many-to-one relation) when we translate it from the ER model to the SQL DDL commands.

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

Relational Schema

ER schema to Relational schema



DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

DDL

```
------------------------Design implementations--------------------
-- Boolean => char(1)
-- definition => varchar(150)
-- Table_name (First letter upper case then underscores)
-- One-to-Many (Store key in one)
-- No state is null, set key to null
-- In an entity: id is id of current entity, create new attribute
table_id for referenced id

--Questions

--victim age/ pregnancy: age of 999 implies that person is not yet
born, so that we don't lose information about the age of the
mother
-- there would be 2 distinct victims (mother normal age, and yet
to be born child age 99)

--in Victims: attribute victim_seating_position_id ||
seating_position_id
-- merge state: Unknown with blank ? => key == null ?

-- Used in both victims and parties

--Check for line between collisions and is_implied in

--Check if attributes of is_implied_in are done correctly

--Pcf_violation_subsection: which type?

--County_city_location: which type?
```

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

```
------------------------Conditions start-------------------------
CREATE TABLE Weather
(
   id          char(1), -- check if if is one of letter
   definition varchar(150),
   PRIMARY KEY (id)
);

CREATE TABLE Road_surface
(
   id          char(1), -- check if if is one of letter
   definition varchar(150),
   PRIMARY KEY (id)
);

CREATE TABLE Road_condition
(
   id          char(1), -- check if if is one of letter
   definition varchar(150),
   PRIMARY KEY (id)
);

CREATE TABLE Lighting
(
   id          char(1), -- check if if is one of letter
   definition varchar(150),
   PRIMARY KEY (id)
);

CREATE TABLE Condition
(
   id                int,
   road_surface_id   char(1) references Road_surface (id),
   lighting_id       char(1) references Lighting (id),
   PRIMARY KEY (id)
);

CREATE TABLE Condition_with_weather
(
   condition_id           int     references Condition (id),
   wheather_id            char(1) references Weather (id),
   PRIMARY KEY (condition_id, wheather_id)
);
```

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

```
CREATE TABLE Condition_with_road_condition
(
   condition_id           int     references Condition (id),
   road_condition_id      char(1) references Road_condition (id),
   PRIMARY KEY (condition_id, road_condition_id)
);
------------------------Conditions end---------------------------


------------------------Collisions start-------------------------

CREATE TABLE Type_of_collision
(
   id          char(1), --check char between a & h
   definition varchar(150),
   PRIMARY KEY (id)
);

CREATE TABLE Collision_severity
(
   id          int CHECK (0 <= id and id <= 4),
   definition varchar(150),
   PRIMARY KEY (id)
);

CREATE TABLE Hit_and_run
(
   id          char(1),
   definition varchar(150),
   PRIMARY KEY (id)
);

CREATE TABLE Primary_collision_factor
(
   id          char(1),
   definition varchar(150),
   PRIMARY KEY (id)
);

CREATE TABLE Pcf_violation_category
(
   id          int CHECK ((0 <= id and id <= 24)),
   definition varchar(150),
   PRIMARY KEY (id)
```

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

```
);

CREATE TABLE Ramp_intersection
(
   id          int CHECK (1 <= id and id <= 8),
   definition varchar(150),
   PRIMARY KEY (id)
);

CREATE TABLE Location_type
(
   id          char(1),
   definition varchar(150),
   PRIMARY KEY (id)
);

CREATE TABLE Population
(
   id          int CHECK (0 <= id and id <= 9),
   definition varchar(150),
   PRIMARY KEY (id)
);

CREATE TABLE Collisions
(
   case_id                     int,
   collision_date              date,
   collision_time              timestamp,
   tow_away                    char(1) CHECK (tow_away = 'Y' or
tow_away = 'N'),
   type_of_collision_id        char(1) references
Type_of_collision (id),
   collision_severity_id       int not null references
Collision_severity (id),
   -- Relations is_judged
   jurisdiction                int CHECK (0<=jurisdiction and
jurisdiction <= 9999),
   officer_id                  int,
   pcf_violation               int,
   pcf_violation_subsection    varchar(150),
   process_date                date,
   hit_and_run_id              char(1) references Hit_and_run
(id),
```

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

```
   primary_collision_factor_id char(1) references
Primary_collision_factor (id),
   pcf_violation_category_id   int references
Pcf_violation_category (id),
   -- Relations happens_in
   county_city_location        int,
   ramp_intersection_id        int references Ramp_intersection
(id),
   location_type_id            char(1) references Location_type
(id),
   population_id               int references Population (id),
   -- Relations happens_under
   condition_id                int references Condition (id),
   PRIMARY KEY (case_id)
);

-------------------------Collisions end---------------------------

CREATE TABLE Safety_equipment
(
   id          char(1),
   definition varchar(150),
   PRIMARY KEY (id)
);


----------------------------Victims start-------------------------

CREATE TABLE Victim_degree_of_injury
(
   id          int CHECK (0 <= id and id <= 7), -- can we make sure
id and def are consistent
   definition varchar(150),
   PRIMARY KEY (id)
);

CREATE TABLE Victim_seating_position
(
   id          char(1), --can we check if id is number or char?
   definition varchar(150),
   PRIMARY KEY (id)
);

CREATE TABLE Victim_role
```

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

```
(
   id          int CHECK (1 <= id and id <= 6),
   definition varchar(150),
   PRIMARY KEY (id)
);

CREATE TABLE Victim_ejected
(
   id          int CHECK (0 <= id and id <= 3), --make sure entity
is still created if id is null
   definition varchar(150),
   PRIMARY KEY (id)
);

CREATE TABLE Victims
(
   id                          int,
   pregnant                    char(1) not null,
   victim_age                  int,
   victim_sex                  char(1),
--- referenced ids--
   party_id                    int     not null,
   victim_degree_of_injury_id int      not null references
Victim_degree_of_injury (id),
   victim_seating_position_id char(1) references
Victim_seating_position (id),
   victim_role_id              int     not null references
Victim_role (id),
   victim_ejected_id           int references Victim_ejected (id),
--     party_id int not null REFERENCES PARTICIPANT (party_id),
   PRIMARY KEY (id)
);

CREATE TABLE Victim_equiped_with_safety_equipment
(
   victim_id           int     references Victims (id),
   safety_equipment_id char(1) not null references
Safety_equipment (id),
   PRIMARY KEY (victim_id, safety_equipment_id)
);

----------------------------Victims end----------------------------
```

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

```
----------------------------Parties start-------------------------

-- Related entities with party: one to many
CREATE TABLE Movement_preceding_collision
(
   id          char(1),
   definition varchar(150),
   PRIMARY KEY (id)
);

CREATE TABLE Party_drug_physical
(
   id          char(1),
   definition varchar(150),
   PRIMARY KEY (id)
);

CREATE TABLE Party_sobriety
(
   id          char(1),
   definition varchar(150),
   PRIMARY KEY (id)
);

CREATE TABLE Party_type
(
   id          int,
   definition varchar(150),
   PRIMARY KEY (id)
);

CREATE TABLE Statewide_vehicle_type
(
   id          char(1),
   definition varchar(150),
   PRIMARY KEY (id)
);

CREATE TABLE Cellphone_use
(
   id          char(1),
   definition varchar(150),
   PRIMARY KEY (id)
);
```

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

```
-- Relations with party: Many to many
CREATE TABLE Other_associated_factor
(
   id          char(1),
   definition varchar(150),
   PRIMARY KEY (id)
);

CREATE TABLE Financial_responsibility
(
   id          char(1),
   definition varchar(150),
   PRIMARY KEY (id)
);

-- Parties
CREATE TABLE Parties
(
   id                                int,
   -- Atributes
   hazardous_materials               char(1),
   party_age                         int,
   party_sex                         char(1),
   -- relation to collision
   collision_case_id                 int     not null references
Collisions (case_id),
   financial_responsibility_id       char(1) references
Financial_responsibility (id),
   school_bus_related                char(1) not null,
   at_fault                          char(1) not null,
   -- referenced ids
   movement_preceding_collision_id char(1) references
Movement_preceding_collision (id),
   party_drug_physical_id            char(1) references
Party_drug_physical (id),
   party_sobriety_id                 char(1) references
Party_sobriety (id),
   party_type_id                     int references Party_type (id),
   statewide_vehicle_type_id         char(1) references
Statewide_vehicle_type (id),
   vehicle_make                      varchar(150),
   vehicle_year                      int,
```

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

```
   cellphone_use_id                char(1) references
Cellphone_use (id),
   -- key
   PRIMARY KEY (id)
);

CREATE TABLE Party_equiped_with_safety_equipment
(
   party_id             int     not null references Parties (id),
   safety_equipment_id char(1) not null references
Safety_equipment (id),
   PRIMARY KEY (party_id, safety_equipment_id)
);

CREATE TABLE Party_associated_with_safety_other_associated_factor
(
   party_id                    int     not null references Parties
(id),
   other_associated_factor_id char(1) not null references
Other_associated_factor (id),
   PRIMARY KEY (party_id, other_associated_factor_id)
);

------------------------------Parties end-------------------------
```

General Comments

In general, we found it pretty hard to create the ER diagram at first because there were a lot of attributes to proceed and understand and also because we didn't have much experience with this kind of work. But after having spent some time, we think that our implementation is now logical and should allow us to retrieve the information without having too many problems.
The allocation between the members was good, since we almost always worked together as a team. We first all took part in the elaboration of the ER diagram by concentrating us each on a CSV file and then talking with each other to see which attributes could belong together.
We then all wrote some of the SQL DDL commands to create the tables and wrote the report together.

Deliverable 2

Assumptions

<In this section write down the assumptions you made about the data. Write a sentence for each assumption you made>

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

Data Loading/Cleaning

Query Implementation

<For each query>
Query a:
Description of logic:
<What does the query do and how do I decide to solve it>
SQL statement
<The SQL statement>
Query result (if the result is big, just a snippet)
<The SQL statement result>

General Comments

<In this section write general comments about your deliverable (comments and work allocation between team members>

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

Deliverable 3

Assumptions

<In this section write down the assumptions you made about the data. Write a sentence for each assumption you made>

Query Implementation

<For each query>
Query a:
Description of logic:
<What does the query do and how do I decide to solve it>
SQL statement
<The SQL statement>
Query result (if the result is big, just a snippet)
<The SQL statement result>

Query Performance Analysis – Indexing

<In this section, for 6 selected queries explain in detail why do you see given improvements (or not). For example, why building an index on certain field changed the plan and IO.>
Query 1
<Initial Running time/IO:
Optimized Running time/IO:
Explain the improvement:
Initial plan
Improved plan>

Query 2
<Initial Running time/IO:
Optimized Running time/IO:
Explain the improvement:
Initial plan
Improved plan>

General Comments

<In this section write general comments about your deliverable (comments and work allocation between team members>