2η Προγραμματιστική Εργασία στα Συστήματα Διαχείρισης και Ανάλυσης Δεδομένων

Νικόλαος Χριστοδούλου, ΑΜ:3190223

Ζήτημα Πρώτο

1. Παρακάτω παρατίθενται οι εντολές που εκτελέστηκαν στον SQL Server για την δημιουργία του πίνακα accdata και την φόρτωση των δεδομένων από το αρχείο ACCDATA.txt:

```
create table accdata (
	accident_id varchar(15),
	severity_id integer,
	severity varchar(10),
	road_surface_conditions_id integer,
	road_surface_conditions varchar(50),
	accident_date date,
	number_of_vehicles integer,
	vehicle_type_id integer,
	vehicle_type varchar(50),
	driver_class_id integer,
	sex_of_driver varchar(6),
	age_of_driver integer,
	sex_of_casualty varchar(6),
	age_of_casualty integer
);

BULK INSERT accdata
FROM 'C:\Users\Nikos\Desktop\DBMS\Project_2\ACCDATA.TXT'
WITH (FIRSTROW =2, FIELDTERMINATOR='|', ROWTERMINATOR = '\n');
```

2. Εν συνεχεία, δημιουργούμε το λογικό σχήμα της αποθήκης δεδομένων, το οποίο έχει τη μορφή αστέρα.

```
--We start with the dw's dimensions.

create table vehicles (
	vehicle_type_id int primary key,
	vehicle_type varchar(50)
);

create table drivers (
	driver_class_id int primary key,
	sex_of_driver varchar(6),
	age_of_driver integer
);

create table severities (
	severity_id integer primary key,
	severity varchar(10)
);

create table road_conditions(
	road_surface_conditions_id integer primary key,
	road_surface_conditions varchar(50)
);


create table dateinfo (
	acc_date date primary key,
	acc_year int,
	acc_month int,
	acc_quarter int,
);


--Procede with the fact table

create table accidents (
	accident_id varchar(15),
	driver_class_id int,
	vehicle_type_id int,
	road_surface_conditions_id int,
	severity_id integer,
	accident_date date,
	number_of_vehicles int,
	number_of_casualties integer,

	primary key (accident_id, driver_class_id, vehicle_type_id,
	road_surface_conditions_id, severity_id, accident_date),
	foreign key (driver_class_id) references drivers(driver_class_id),
	foreign key (vehicle_type_id) references vehicles(vehicle_type_id),
	foreign key (road_surface_conditions_id) references
road_conditions(road_surface_conditions_id),
	foreign key (severity_id) references severities (severity_id),
	foreign key (accident_date) references dateinfo (acc_date)

);
```

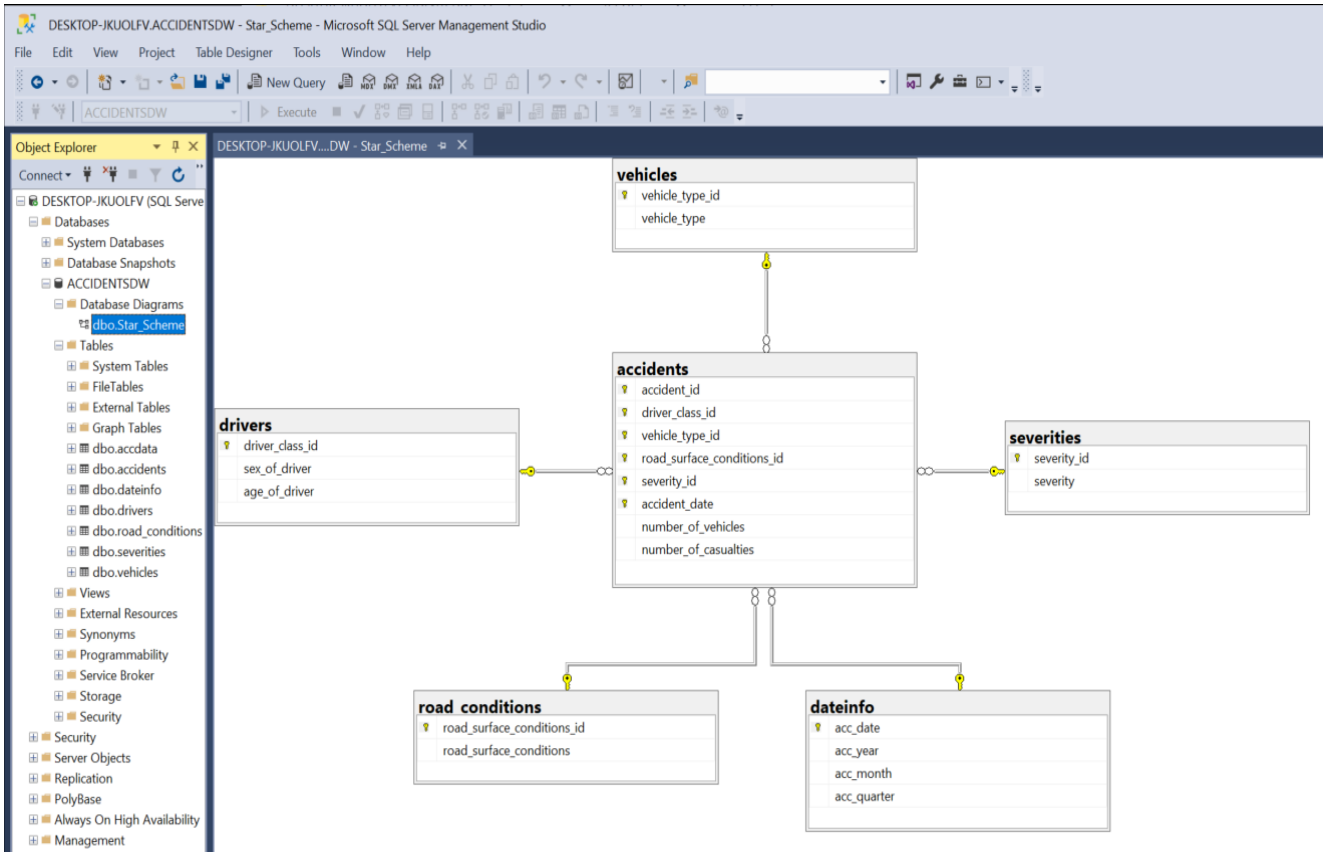
Όπως βλέπουμε επιλέξαμε τη δημιουργία 5 πινάκων διαστάσεων, ανάλογα με τα πεδία που μας δίνονταν στην αποθήκη δεδομένων. Σημαντικό είναι να αναφέρουμε ότι τα πεδία sex_of_casualty και age_of_casualty δεν τα συμπεριλαμβάνουμε σε κάποια διάσταση, διότι δεν έχουν κάποιο μοναδικό id που να μπορέσει να τα συσχετίσει με το fact table των accidents. Επιπλέον, δημιουργήθηκε μία διάσταση χρόνου (πίνακας dateinfo), στην οποία συμπεριλήφθηκε το κλειδί accident_date μαζί με τον χρόνο, το τρίμηνο και τον μήνα του ατυχήματος.

Όσον αφορά στο fact table, συμπεριλάβαμε όλα τα πεδία των id που συσχετίζονται με τις διαστάσεις των πινάκων και προσθέσαμε τα μετρήσιμα πεδία της αποθήκης δεδομένων. Βέβαια, επειδή θέλουμε να έχουμε μία μετρική του αριθμού των θυμάτων (casualties), η οποία δε δίνεται από την αποθήκη δεδομένων, χρησιμοποιούμε όπως θα δούμε και παρακάτω το count (age_of_casualty). Με αυτόν τον τρόπο, λοιπόν, έχουμε εξασφαλίσει και τον υπολογισμό των θυμάτων.

3. Παρακάτω παρουσιάζονται οι εντολές για την τροφοδότηση των διαστάσεων και του fact table με δεδομένα:

```
insert into vehicles
       select distinct vehicle_type_id, vehicle_type
       from accdata;


insert into drivers
       select distinct driver_class_id, sex_of_driver, age_of_driver
       from accdata;

insert into severities
       select distinct severity_id, severity
       from accdata;

insert into road_conditions
       select distinct road_surface_conditions_id, road_surface_conditions
       from accdata


insert into dateinfo
       select distinct accident_date, datepart(year, accident_date),
                                    datepart(month, accident_date),
datepart(quarter, accident_date)
       from accdata

insert into accidents
       select accident_id,
       driver_class_id,
       vehicle_type_id,
       road_surface_conditions_id,
       severity_id,
       accident_date,
       number_of_vehicles,
       count(age_of_casualty)

  from accdata group by
  accident_id,driver_class_id,vehicle_type_id,road_surface_conditions_id,
       severity_id,accident_date,number_of_vehicles;
```

4. Στο σημείο αυτό παρουσιάζεται διαγραμματικά το λογικό σχήμα της αποθήκης δεδομένων:



Ζήτημα Δεύτερο

Παρακάτω παρατίθενται οι εντολές σε SQL για την απάντηση του αντίστοιχου ερωτήματος της εκφώνησης.

1.
```
select count(accident_id) as number_of_accidents, severity, acc_year
      from accidents, severities, dateinfo
            where accidents.severity_id = severities.severity_id and
                  dateinfo.acc_date = accidents.accident_date
            group by acc_year, severity
            order by acc_year desc
```

2.
```
select count(accident_id) as num_of_fatal_accidents, sex_of_driver,
age_of_driver, sum(number_of_casualties) as sum_of_casualties
       from accidents, drivers, severities
              where drivers.driver_class_id = accidents.driver_class_id and
                     accidents.severity_id = severities.severity_id and
                            severity = 'Fatal'
              group by sex_of_driver, age_of_driver
```

3.
```
select count(accident_id) as num_of_accidents, road_surface_conditions,
severity
       from accidents, road_conditions, severities
              where accidents.road_surface_conditions_id =
road_conditions.road_surface_conditions_id and
                     severities.severity_id = accidents.severity_id
              group by road_surface_conditions, severity
```

4.
```
select count(accident_id) as num_of_accidents, sum(number_of_casualties) as
sum_of_casualties, vehicle_type, acc_year
       from accidents, vehicles, dateinfo
              where accidents.vehicle_type_id = vehicles.vehicle_type_id and
                     accidents.accident_date = dateinfo.acc_date and
                            number_of_vehicles > 2
       group by vehicle_type, acc_year
```

5. Στο συγκεκριμένο ερώτημα η λύση θα δοθεί με τη βοήθεια του group by rollup. Στο α. μέρος της άσκησης στην ουσία ζητείται το group by none, το οποίο αθροίζει όλα τα πεδία που βρίσκονται στο group by και επιστρέφει το συνολικό αριθμό των ατυχημάτων, οχημάτων και θυμάτων που καταγράφηκαν όλη τη δεκαετία. Στο b. μέρος ζητείται το group by acc_year, ενώ στο c. ζητείται το group by acc_quatert, acc_month. Όλα τα παραπάνω, λοιπόν, μπορούμε να τα απαντήσουμε με το rollup με την χρήση μόνο μίας επερώτησης.

```
select count(accident_id) as num_of_accidents, sum(number_of_vehicles) as
sum_of__vehicles, sum(number_of_casualties) as sum_of_casualties, acc_year,
acc_quarter, acc_month
       from accidents, dateinfo where accidents.accident_date =
dateinfo.acc_date
              group by rollup(acc_year, acc_quarter, acc_month)
```

Ζήτημα Τρίτο

1. Το συγκεκριμένο ερώτημα απαντάται πολύ απλά με την χρήση του group by cube, όπως φαίνεται και στο παρακάτω query:

```
select count(accident_id), severity, road_surface_conditions, vehicle_type
       from accidents, severities, road_conditions, vehicles
              where accidents.severity_id = severities.severity_id and
                     accidents.road_surface_conditions_id =
road_conditions.road_surface_conditions_id and
                           accidents.vehicle_type_id =
vehicles.vehicle_type_id
       group by cube(severity, road_surface_conditions, vehicle_type);
```

2. Για τη δημιουργία materialized view στον SQL Server απαιτείται η δημιουργία μίας απλής όψης με την χρήση **with schemabinding** και στην συνέχεια η δημιουργία unique clustered index πάνω στην όψη αυτή. Βέβαια, προτού φτιάξουμε την όψη πρέπει πρώτα να διαμορφώσουμε απαραίτητα τις ρυθμίσεις του SQL Server για να επιτρέψει την δημιουργία του indexed view. Άρα, πρώτα τρέχουμε τις εντολές

```
SET NUMERIC_ROUNDABORT OFF;
SET ANSI_PADDING, ANSI_WARNINGS, CONCAT_NULL_YIELDS_NULL, ARITHABORT,
   QUOTED_IDENTIFIER, ANSI_NULLS ON;
--Create view with schemabinding.
IF OBJECT_ID ('v1 ', 'view') IS NOT NULL
   DROP VIEW v1;
```

και τώρα φτιάχνουμε την όψη ως εξής:

```
create view v1 with schemabinding
       as select count_big(*) as num_of_accidents, severity,
road_surface_conditions, vehicle_type
       from dbo.accidents, dbo.severities, dbo.road_conditions, dbo.vehicles
              where accidents.severity_id = severities.severity_id and
                     accidents.road_surface_conditions_id =
road_conditions.road_surface_conditions_id and
                           accidents.vehicle_type_id =
vehicles.vehicle_type_id
       group by severity, road_surface_conditions, vehicle_type;
```

Τώρα είμαστε στην θέση να δημιουργήσουμε και το ευρετήριο.

```
create unique clustered index idx_accidents on v1(severity,
road_surface_conditions, vehicle_type)
```

Όπως φαίνεται από τις παραπάνω εντολές, η όψη που δημιουργήθηκε περιέχει την εντολή group by severity, road_surface_conditions, vehicle type, οπότε με το `select * from v1` παίρνουμε τα κατάλληλα δεδομένα. Όσον αφορά στα υπόλοιπα αποτελέσματα που λαμβάνουμε από το group by cube πρέπει να υπολογιστούν άλλα $2^3 - 1$ (αφού έχουμε ήδη υπολογίσει το ένα group by και το group by cube

περιέχει 3 πεδία). Παρακάτω παρουσιάζονται οι υπόλοιπες 7 επερωτήσεις με την χρήση του indexed view για την απάντηση της άσκησης.

```
select sum(num_of_accidents), severity from dbo.v1
       group by severity

select sum(num_of_accidents), road_surface_conditions from dbo.v1
       group by road_surface_conditions

select sum(num_of_accidents), vehicle_type from dbo.v1
       group by vehicle_type

select sum(num_of_accidents), severity, road_surface_conditions from dbo.v1
       group by severity, road_surface_conditions

select sum(num_of_accidents), severity, vehicle_type from dbo.v1
       group by severity, vehicle_type

select sum(num_of_accidents), road_surface_conditions, vehicle_type from dbo.v1
       group by road_surface_conditions, vehicle_type

select sum(num_of_accidents) from dbo.v1
```