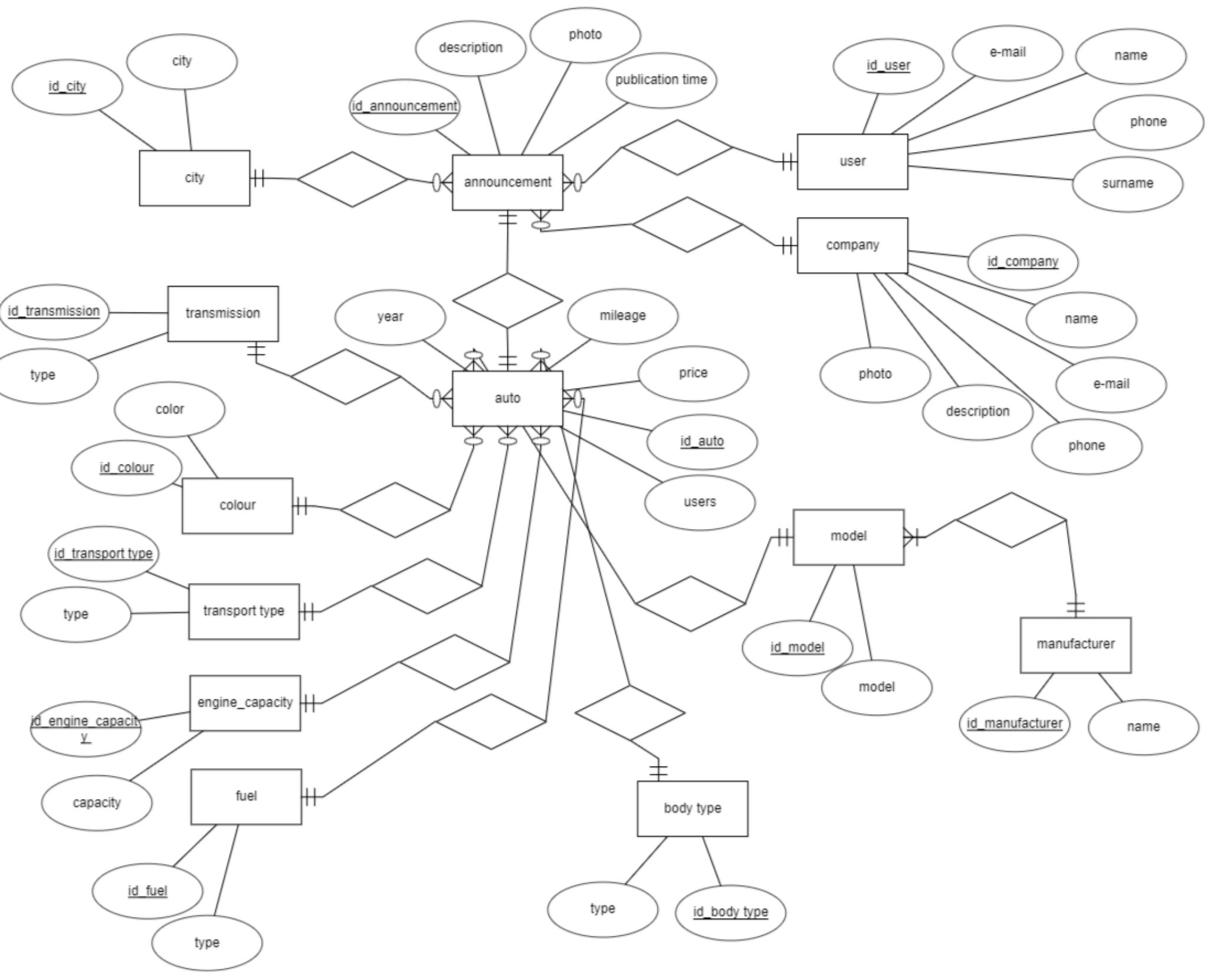
DATABÁZA AUTO-BAZÁR

Ruslan Diakov

Entitno-relačný model

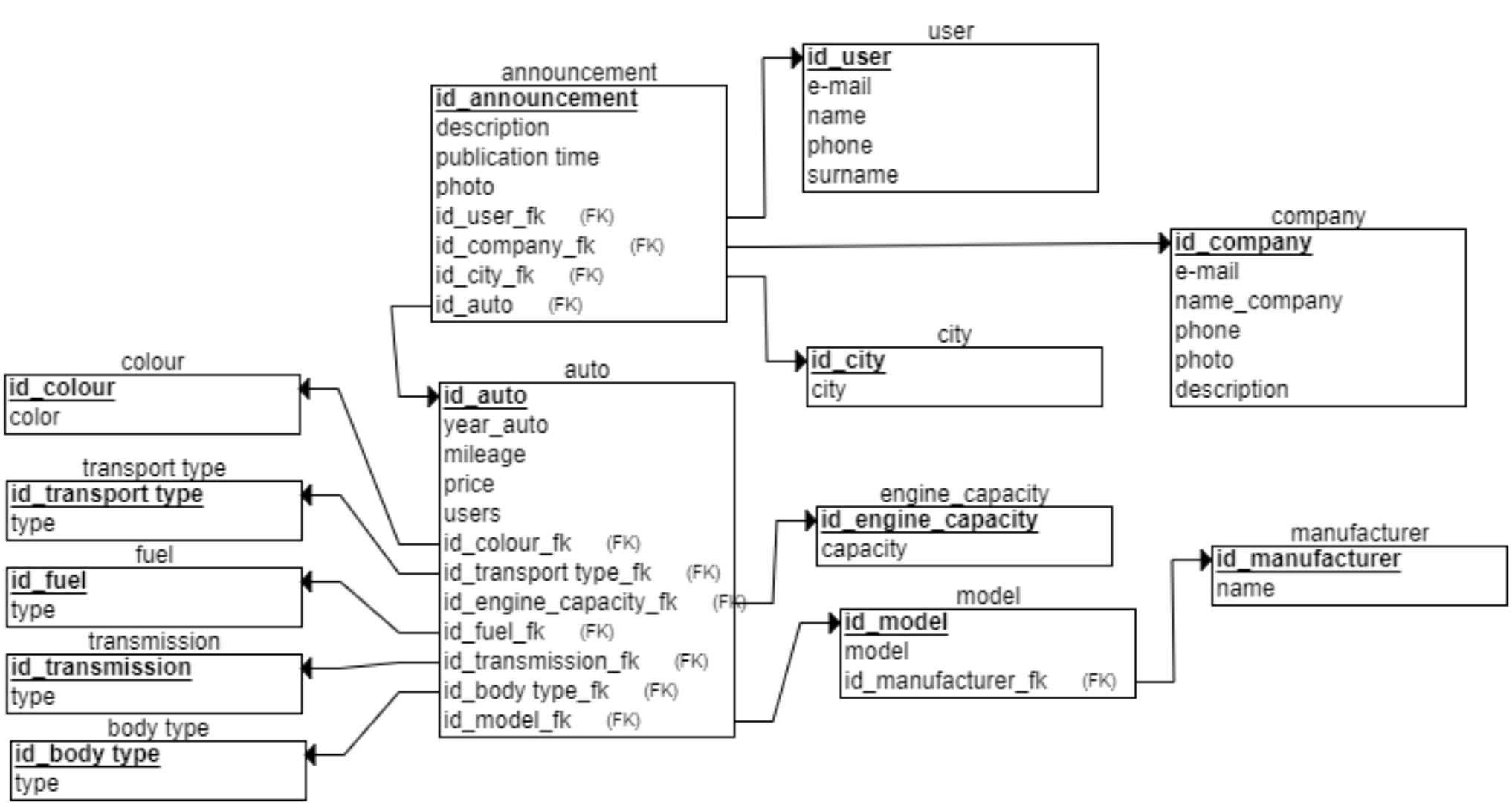
Entities	Link type	Link content
announcement-user	M-1	jeden používateľ môže mať veľa reklám
announcement-company	M-1	jedna spoločnosť môže mať veľa reklám
announcement-city	M-1	jedno mesto môže odkazovať na mnoho reklám
announcement-auto	1-1	jedna reklama môže mať iba 1 auto
auto-transmission	M-1	auto môže mať iba jeden typ prenosu
auto-colour	M-1	auto môže mať iba jednu farbu karosérie
auto-transport_type	M-1	vozidlo môže byť iba v 1 forme (auto ,motocykel)
auto-engine_capacity	M-1	auto môže byť iba s jedným objemom motora
auto-fuel	M-1	auto môže mať iba 1 palivo (tiež hybrid)
auto-body_type	M-1	auto môže mať iba 1 typ karosérie (kombi, SUV, sedan, kupé)
auto-model	M-1	vozidlo môže byť len 1 model
model-manufacturer	M-1	model automobilu môže patriť iba 1 spoločnosti



Logický relačný model

Entity	Attribute
user	id_user e-mail name surname phone
company	id_company e-mail name_company phone photo description
announcement	id_announcement description publication_time photo id_user_fk id_company_fk id_city_fk
auto	id_auto year_auto mileage price users id_announcement_fk id_colour_fk id_transport type_fk id_engine_capacity_fk id_fuel_fk id_transmission_fk id_body type_fk id_model_fk

Entity	Attribute
city	id_city city
colour	id_colour color
transport_type	id_transport type type
fuel	id_fuel type
transmission	id_transmission type
body_type	id_body type type
engine_capacity	id_engine_capacity capacity
model	id_model model id_manufacturer_fk
manufacturer	id_manufacturer name



SQL skript

```
--manufacturer
 create table manufacturer
(
    id_manufacturer serial
        constraint manufacturer_pk
            primary key,
    name            varchar(50) not null
);

create unique index manufacturer_id_manufacturer_uindex
    on manufacturer (id_manufacturer);

create unique index manufacturer_name_uindex
    on manufacturer (name);

--model
create table model
(
    id_model          serial
        constraint model_pk
            primary key,
    model             varchar(50) not null,
    id_manufacturer_fk int         not null
        constraint model_manufacturer_id_manufacturer_fk
            references manufacturer
);

create unique index model_id_model_uindex
    on model (id_model);

create unique index model_model_uindex
    on model (model);

--city
create table city
(
    id_city serial
        constraint city_pk
            primary key,
    city    varchar(50) not null
);

create unique index city_city_uindex
    on city (city);

create unique index city_id_city_uindex
    on city (id_city);

--body_type
create table body_type
(
    id_body_type serial
        constraint body_type_pk
            primary key,
    type         varchar(20) not null
);

create unique index body_type_id_body_type_uindex
    on body_type (id_body_type);

create unique index body_type_type_uindex
    on body_type (type);

--colour
create table colour
(
    id_colour serial
        constraint colour_pk
            primary key,
    color     varchar(15) not null
);

create unique index colour_color_uindex
    on colour (color);

create unique index colour_id_colour_uindex
    on colour (id_colour);
```

SQL skript

```
--engine_capacity
create table engine_capacity
(
    id_engine_capacity serial
        constraint engine_capacity_pk
            primary key,
    capacity           float not null
);

create unique index engine_capacity_capacity_uindex
    on engine_capacity (capacity);

create unique index engine_capacity_id_engine_capacity_uindex
    on engine_capacity (id_engine_capacity);

--fuel
create table fuel
(
    id_fuel serial
        constraint fuel_pk
            primary key,
    type    varchar(20) not null
);

create unique index fuel_id_fuel_uindex
    on fuel (id_fuel);

create unique index fuel_type_uindex
    on fuel (type);

--transport_type
create table transport_type
(
    "id_transport type" serial
        constraint transport_type_pk
            primary key,
    type                varchar(20) not null
);

create unique index "transport_type_id_transport type_uindex"
    on transport_type ("id_transport type");

create unique index transport_type_type_uindex
    on transport_type (type);

--transmission
create table transmission
(
    id_transmission serial
        constraint transmission_pk
            primary key,
    type            varchar(20) not null
);

create unique index transmission_id_transmission_uindex
    on transmission (id_transmission);

create unique index transmission_type_uindex
    on transmission (type);

--user
create table "user"
(
    id_user      serial
        constraint user_pk
            primary key,
    "e-mail"     varchar(255) not null,
    name_user    varchar(50)  not null,
    surname_user varchar(50)  not null,
    phone        int          not null
);

create unique index user_id_user_uindex
    on "user" (id_user);
```

SQL skript

```
--company
create table company
(
    id_company   serial
        constraint company_pk
            primary key,
    "e-mail"     varchar(255) not null,
    name_company varchar(50)  not null,
    phone        int          not null,
    photo        varchar(255)         not null,
    description  varchar(255)
);

create unique index company_id_company_uindex
    on company (id_company);

--announcement
create table announcement
(
    id_announcement  serial
        constraint announcement_pk
            primary key,
    description      varchar(255),
    publication_time date not null,
    photo            varchar(255)  not null,
    id_user_fk       int
        constraint announcement_user_id_user_fk
            references "user",
    id_city_fk       int  not null
        constraint announcement_city_id_city_fk
            references city,
    id_company_fk    int
        constraint announcement_city_id_company_fk
            references company,
    id_auto_fk       integer      not null
       constraint announcement_auto_id_auto_fk
            references schema_autobazar.auto
);

create unique index announcement_id_announcement_uindex
    on announcement (id_announcement);

--auto
create table auto
(
    id_auto              serial
        constraint auto_pk
            primary key,
    year_auto            date not null,
    mileage              int  not null,
    price                int  not null,
    users                int  not null,
    id_colour_fk         int  not null
        constraint auto_colour_id_colour_fk
            references colour,
    id_transport_type_fk  int  not null
        constraint "auto_transport_type_id_transport type_fk"
            references transport_type,
    id_engine_capacity_fk int  not null
        constraint auto_engine_capacity_id_engine_capacity_fk
            references engine_capacity,
    id_fuel_fk           int  not null
        constraint auto_fuel_id_fuel_fk
            references fuel,
    id_transmission_fk    int  not null
        constraint auto_transmission_id_transmission_fk
            references transmission,
    id_body_type_fk       int  not null
        constraint auto_body_type_id_body_type_fk
            references body_type,
    id_model_fk          int  not null
        constraint auto_model_id_model_fk
            references model (id_model)
);

create unique index auto_id_auto_uindex
    on auto (id_auto);
```

SQL skript na naplnenie databázy

```
--colour
--INSERT INTO colour (color)
--VALUES ('White'), ('Black'), ('Silver'), ('Red'), ('Green'), ('Yellow'), ('Blue');

--city
--INSERT INTO city (city)
--VALUES ('Bratislava'), ('Kosice'), ('Preshov'), ('Zhilin'), ('Banska Bystrica'), ('Nitra'), ('Trnava'), ('Trenchin'), ('Martin');

--fuel
--INSERT INTO fuel (type)
--VALUES ('Gasoline'), ('Diesel'), ('Gas'), ('Hybrid'), ('Electo');

--transport type
--INSERT INTO transport_type (type)
--VALUES ('Passenger Car'), ('Truck'), ('Moto'), ('Water Transport'), ('Air Transport'), ('Motorhome');

--transmission
--INSERT INTO transmission (type)
--VALUES ('Manual'), ('Automatic'), ('Tiptronic'), ('Variator');

--body type
--INSERT INTO body_type (type)
--VALUES ('Station Wagon'), ('Sedan'), ('Compartment'), ('Hatchback'), ('Cabriolet'), ('Limousine'), ('Minivan'), ('Pickup'),
('Crossover');
```

```
--engine_capacity
--INSERT INTO engine_capacity (capacity)
--VALUES
--    ('1.0'), ('1.1'), ('1.2'), ('1.3'), ('1.4'), ('1.5'), ('1.6'), ('1.7'), ('1.8'), ('1.9')
--,   ('2.0'), ('2.1'), ('2.2'), ('2.3'), ('2.4'), ('2.5'), ('2.6'), ('2.7'), ('2.8'), ('2.9'),
--    ('3.0'), ('3.1'), ('3.2'), ('3.3'), ('3.4'), ('3.5'), ('3.6'), ('3.7'), ('3.8'), ('3.9')
--,   ('4.0'), ('4.1'), ('4.2'), ('4.3'), ('4.4'), ('4.5'), ('4.6'), ('4.7'), ('4.8'), ('4.9'),
--    ('5.0'), ('5.1'), ('5.2'), ('5.3'), ('5.4'), ('5.5'), ('5.6'), ('5.7'), ('5.8'), ('5.9')
--,   ('6.0'), ('6.1'), ('6.2'), ('6.3'), ('6.4'), ('6.5'), ('6.6'), ('6.7'), ('6.8'), ('6.9'), ('7.0');

-- manufacturer
--INSERT INTO manufacturer (name)
--VALUES ('Audi'), ('Bentley'), ('Porsche'), ('SEAT'), ('Skoda'),
--       ('Volkswagen'), ('Toyota'), ('Lexus'), ('Renault'),
--       ('Nissan'), ('Mitsibishi'), ('Hyundai'), ('Kia'), ('Ford'),
--       ('Honda'), ('Fiat'), ('Peugeot'), ('Citroen'), ('Opel'),
--       ('Suzuki'), ('Mercedes-Benz'), ('BMW'), ('Mini'), ('Rolls-Royce'),
--       ('Kawasaki'),('Yamaha');

--model
--INSERT INTO model (model, id_manufacturer_fk)
--VALUES ('Q2',1), ('Q3',1), ('Q4',1), ('Q5',1), ('Q6',1), ('Q7',1), ('Q8',1),
--       ('Mulsanne',2), ('Eight',2), ('Continental',2),
--       ('Taycan',3), ('Cayenne',3), ('911',3),
--       ('Alhambra',4),
--       ('Octavia',5), ('Rapid',5), ('Yeti',5), ('Fabia',5),
--       ('Amarok',6), ('Passat',6),('Polo',6),
--       ('GSX-R750',20), ('GSX-S1000GT',20), ('KINGQUAD',20),
--       ('Jet Ski Ultra 310R',25), ('Jet Ski STX 160LX',25),
--       ('212SS',26);

--user
--INSERT INTO "user" ("e-mail", name_user, surname_user, phone)
--VALUES ('ruslan.diakov@mail.com','Ruslan','Diakov',951300300),
--       ('leonardo.dicaprio@mail.com','Leonardo','Dicaprio',951300777),
--       ('brad.pitt@mail.com','Brad','Pitt',951390712),
--       ('angelina.jolie@mail.com','Angelina','Jolie',951780712),
--       ('johnny.depp@mail.com','Johnny','Depp',951300777),
--       ('marcel.vološin@mail.com','Marcel','Vološin',951303030);

--company
--INSERT INTO company ("e-mail", name_company, phone, photo, description)
--VALUES ('autopodium@mail.com','Autopodium',951100100,'C:\photo\company\Autopodium.png','Ahoj predávame autá'),
--       ('autogalaxy@mail.com','Autogalaxy',951202200,'C:\photo\company\Autogalaxy.png','Kúpte si auto svojich snov'),
--       ('autoworld@mail.com','Autoworld',951199900,'C:\photo\company\Autoworld.png','Najlepšie autá nášho mesta'),
--       ('freshauto@mail.com','Freshauto',951123450,'C:\photo\company\Freshauto.png','');

--announcement (user auto)
/*INSERT INTO announcement (description, publication_time, photo, id_user_fk, id_city_fk, id_auto_fk)
VALUES ('dobré, krásne, nie rozbité auto', '2022-03-24', 'C:\photo\announcement\1\foto1.png',1,2,1),
      ('auto the best', '2022-02-21', 'C:\photo\announcement\2\foto1.png',2,2,8),
      ('Dobre auto', '2022-03-12', 'C:\photo\announcement\4\foto1.png',2,1,3),
      ('Auto po taxíku', '2022-02-17', 'C:\photo\announcement\5\foto1.png',3,3,4),
      ('Auto je lepšie ako bicykel', '2022-03-01', 'C:\photo\announcement\6\foto1.png',4,3,5),
      ('', '2022-02-19', 'C:\photo\announcement\7\foto1.png',5,6,6),
      ('Super auto', '2022-03-20', 'C:\photo\announcement\8\foto1.png',6,2,7);
*/
```

SQL skript na naplnenie databázy

```
--announcement (company auto)
--INSERT INTO announcement (description, publication_time, photo, id_company_fk, id_city_fk, id_auto_fk)
--VALUES ('opravené auto, môže riadiť', '2022-01-13', 'C:\photo\announcement\2\foto1.png',1,1,2);

--auto
--INSERT INTO auto (year_auto, mileage, price, users, id_colour_fk, id_transport_type_fk, id_engine_capacity_fk, id_fuel_fk,
id_transmission_fk, id_body_type_fk, id_model_fk)
--VALUES (2010,166000,40500,1,1,1,21,2,2,9,7),
-- (2013,150000,13000,2,2,1,11,1,2,2,21),
-- (2008,190000,4500,4,4,4,5,1,2,10,25),
-- (2020,30000,76000,1,1,1,62,5,3,2,11),
-- (2019,87000,60300,2,7,1,39,2,3,9,12),
-- (2011,170000,9000,3,5,1,3,1,1,9,17),
-- (2021,21000,89000,1,3,1,31,2,2,5,13),
-- (2004,180000,23450,5,2,4,9,1,2,10,27);
```

SQL skript na na vytvorenie pohľadov

```
-- 1) zobraziť všetky autá, Zoradiť podľa ceny
/*
select a.price as price, m2.name || ' '|| m.model as auto
from auto a
left join model m on m.id_model = a.id_model_fk
left join manufacturer m2 on m2.id_manufacturer = m.id_manufacturer_fk
order by a.price
*/

-- 2) zobraziť všetky vozidlá, ktoré majú špecifickú farbu (biela)
/*
select c.color, m2.name ||' '|| m.model as Auto, auto.year_auto
from auto
left join colour c on c.id_colour = auto.id_colour_fk
left join model m on m.id_model = auto.id_model_fk
left join manufacturer m2 on m2.id_manufacturer = m.id_manufacturer_fk
where c.color= 'White'
*/

-- 3) zobrazenie typov prepravy a ich počtu
/*
select tt.type, count(*)
from announcement a
left join auto a2 on a2.id_auto = a.id_auto_fk
left join transport_type tt on tt."id_transport type" = a2.id_transport_type_fk
group by tt.type;
*/

-- 4) zobrazenie autá, ktoré majú motorovú naftu typu a najazdených kilometrov pod 100k
/*
select f.type as fuel, m2.name || ' ' || m.model as auto, a.mileage
from auto a
left join fuel f on f.id_fuel = a.id_fuel_fk
left join model m on m.id_model = a.id_model_fk
left join manufacturer m2 on m2.id_manufacturer = m.id_manufacturer_fk
where f.type = 'Diesel' and a.mileage < 100000
*/

-- 5) zobraziť úplné informácie o autách, ktoré používateľ predáva (vyhľadávanie podľa mena a priezviska)
/*
select Uuser.name_user as "user", Mmanufacturer.name ||' '|| Mmodel.model ||' '|| round(cast(EC.capacity as numeric)
 as Auto, Aauto.price ||' €' as price, c.city ||' '|| Aannouncement.publication_time  as City_Date,
       Aauto.year_auto ||' - '|| Aauto.mileage as year_mileage, Ccolour.color,
       tt.type ||' - '|| BT.type ||' - '|| t.type AS transport_type, f.type as Fuel_Type
from announcement Aannouncement
left join "user" Uuser  on Uuser.id_user = Aannouncement.id_user_fk
left join auto Aauto  on Aannouncement.id_auto_fk = Aauto.id_auto
left join model Mmodel on Aauto.id_model_fk = Mmodel.id_model
left join manufacturer Mmanufacturer on Mmodel.id_manufacturer_fk = Mmanufacturer.id_manufacturer
left outer join colour Ccolour on Aauto.id_colour_fk = Ccolour.id_colour
left outer join body_type BT on Aauto.id_body_type_fk = BT.id_body_type
left outer join engine_capacity EC on Aauto.id_engine_capacity_fk = EC.id_engine_capacity
left outer join transport_type tt on Aauto.id_transport_type_fk = tt."id_transport type"
left outer join fuel f on Aauto.id_fuel_fk = f.id_fuel
left outer join city c on Aannouncement.id_city_fk = c.id_city
left outer join transmission t on Aauto.id_transmission_fk = t.id_transmission

where Uuser.name_user ='Ruslan' and Uuser.surname_user = 'Diakov'
*/
```

SQL skript na na vytvorenie pohľadov

```
-- 6) zobrazte priemernú cenu automobilu a počet reklám v konkrétnom meste
/*
select c.city, round(AVG(price), 0) || ' €' as avg_price, Count(*) as count_announcement
from auto
left join announcement a on auto.id_auto = a.id_auto_fk
left join city c on c.id_city = a.id_city_fk
where city = 'Kosice'
group by c.city
*/

-- 7) priemerné ročné auto a počet publikácií za určitý mesiac
/*
select round(AVG(a.year_auto), 0) as avg_year_auto, Count(*) as count_announcement
from auto a
left join announcement aa on a.id_auto = aa.id_auto_fk
where extract(month from aa.publication_time) = 2;
*/
--8) zobrazenie počtu reklám v každom mesiaci
/*select extract(years from a.publication_time) ||'-'|| extract(months from a.publication_time) as daate,
       count(extract(day from a.publication_time)) as count_announcement
from announcement a
group by rollup (daate)
order by daate;*/
```

```
--9)zobraziť všetky reklamy z mesta Košice, kde je druh dopravy "osobný automobil",
-- a tiež zobraziť všetky reklamy z mesta Bratislava s akýmkoľvek druhom dopravy
/*
select c.city, m2.name ||' - '|| m.model as auto, tt.type
from city c
join announcement a on c.id_city = a.id_city_fk
join auto a2 on a2.id_auto = a.id_auto_fk
join model m on m.id_model = a2.id_model_fk
join manufacturer m2 on m2.id_manufacturer = m.id_manufacturer_fk
join transport_type tt on tt."id_transport type" = a2.id_transport_type_fk
where c.city='Kosice' and tt.type = 'Passenger Car'
union
select c.city, m2.name ||' - '|| m.model as auto, tt.type
from city c
join announcement a on c.id_city = a.id_city_fk
join auto a2 on a2.id_auto = a.id_auto_fk
join model m on m.id_model = a2.id_model_fk
join manufacturer m2 on m2.id_manufacturer = m.id_manufacturer_fk
join transport_type tt on tt."id_transport type" = a2.id_transport_type_fk
where c.city='Bratislava';
*/

--10) pomocou "right join" zobraziť všetkých výrobcov a počet modelov
-- zoskupte podľa názvu výrobcu a vyberte iba tie, ktoré majú menej ako 3 modely
/*
select m.name, count(m2.model) as count
from model m2
right join manufacturer m on m.id_manufacturer = m2.id_manufacturer_fk
group by m.name
having count(m2.model)<3
order by count desc;
*/
```

SQL skript odovzdanie 3

```
----------Finálne odovzdávanie zadania----------
----1 pohľad s použitím množinových operácií
--zobraziť všetky autá s čiernym lakom a benzínom, modrý lak a nafta, a všetky motocykle
create or replace view multiselect as
select tt.type as transport_type, m.model as auto, c.color, f.type as fuel
from auto a
full outer join colour c on c.id_colour = a.id_colour_fk
full outer join fuel f on f.id_fuel = a.id_fuel_fk
full outer join model m on m.id_model = a.id_model_fk
full outer join transport_type tt on a.id_transport_type_fk = tt.id_transport_type
where c.color = 'Black' and f.type ='Gasoline'
union
select  tt.type as transport_type, m.model as auto, c.color, f.type as fuel
from auto a
full outer join colour c on c.id_colour = a.id_colour_fk
full outer join fuel f on f.id_fuel = a.id_fuel_fk
full outer join model m on m.id_model = a.id_model_fk
full outer join transport_type tt on a.id_transport_type_fk = tt.id_transport_type
where c.color = 'Blue' and f.type ='Diesel'
union
select  tt.type as transport_type, m.model as auto, c.color, f.type as fuel
from auto a
full outer join colour c on c.id_colour = a.id_colour_fk
full outer join fuel f on f.id_fuel = a.id_fuel_fk
full outer join model m on m.id_model = a.id_model_fk
full outer join transport_type tt on a.id_transport_type_fk = tt.id_transport_type
where tt.type = 'Moto'
order by  transport_type;

----2 pohľady s použitím vnorených poddopytov.
--zobraziť všetky autá používateľa podľa telefónneho čísla
create or replace view model_userPhone as
select m.model
from auto a
left join model m on m.id_model = a.id_model_fk
where a.id_auto in (
    select a2.id_auto_fk
    from announcement a2
    join "user" u on a2.id_user_fk = u.id_user
    where u.phone = '951300300'
    );

--zobraziť všetky autá a používateľov v meste s najväčším počtom reklám
create or replace view Usermodel_cityMaxAnnouncement as
select "user".name_user, m.model
from "user"
left join announcement a on "user".id_user = a.id_user_fk
left join auto a2 on a2.id_auto = a.id_auto_fk
left join model m on m.id_model = a2.id_model_fk
where id_city_fk = (
    select t1.id_city_fk
    from (
        select count(*) as ccount, a2.id_city_fk
        from city c
        join announcement a2 on c.id_city = a2.id_city_fk
        group by a2.id_city_fk
        order by ccount desc
        limit 1
        )as t1
);

----skript na vytvorenie triggeru/triggerov, ktoré budú implementovať autoinkrementáciu umelých kľúčov
--trigger pre tabuľky city z autoinkrementácim id_city
drop trigger autoIncCity on city;
create trigger autoIncCity before insert on city
    for each row
    execute procedure func_autoinc();

create or replace function func_autoinc() returns trigger as
$$
declare
        max_id_city int;
    begin
        select max(id_city) into max_id_city from city;

        --skontrolujme, či je tabuľka prázdna
        if (max_id_city) is null and new.id_city is null then
            new.id_city = 1;
        end if;
```

SQL skript odovzdanie 3

```
 --skontrolujme, či 'new.id_city' prázdny
 if new.id_city is null then
 new.id_city = (max_id_city)+1;
 end if;
 --skontrolujme, či 'new.id_city' uz je v tabulke
 if( select id_city from city
 where new.id_city in(id_city)) is not null then
 new.id_city = (max_id_city)+1;
 end if;
 return new;
 end;
 $$
language plpgsql;

insert into city(city)
values ('Lviv');
insert into city(id_city, city)
values (99, 'Odesa');

----skript na aspoň dva zmysluplné triggre (okrem triggerov na autoinkrementáciu)
--trigger pre view manufacturer_models ktorý pridá nový model do tabuľky modelov
create or replace view manufacturer_models as
select m.name as Mmanufacturer, m2.model as Mmodel
from model m2
right join manufacturer m on m.id_manufacturer = m2.id_manufacturer_fk;

create trigger add_manufacturer_models instead of update on manufacturer_models
   for each row
   execute procedure add_auto();

drop function ret_id_manufacturer;
CREATE or replace FUNCTION ret_id_manufacturer(nname varchar(50)) RETURNS integer AS $$
   select id_manufacturer from manufacturer where name = nname;
$$ LANGUAGE SQL;

create or replace function add_auto() returns trigger as
   $$
   begin
      insert into model(model, id_manufacturer_fk)
      VALUES (new.Mmodel ,ret_id_manufacturer(new.Mmanufacturer));
      return new;
   end;
   $$
language plpgsql;

update manufacturer_models set mmodel = 'PICANTO' where mmanufacturer = 'Kia';

--trigger pre  view auto_price2 čo zmení cenu auta
drop view auto_price2;
create or replace view auto_price2 as
select auto.id_auto as id, auto.price as pprice, m.model as mmodel
from auto
inner join model m on m.id_model = auto.id_model_fk
order by pprice;

CREATE or replace FUNCTION ret_id_model(nname varchar(50)) RETURNS integer AS $$
   select id_model from model where model = nname;
$$ LANGUAGE SQL;

create or replace function udpPrice()returns trigger as
   $$
   begin
      update auto set price = new.pprice where id_model_fk = ret_id_model(new.mmodel) and id_auto = new.id;
      return new;
   end;
   $$
language plpgsql;

create trigger udp_price instead of update on auto_price2
   for each row
   execute procedure udpPrice();

update auto_price2 set pprice = 13000 where id = 9 and mmodel = 'Q5';
```

SQL skript odovzdanie 3

```
----skript na aspoň jednu storovanú procedúru a jednu funkciu
----------  1  --------------------
--trigger pridá do tabuľky user_log nového používateľa
create table user_log(
    id_user_log serial unique not null primary key ,
    id_user      int not null unique ,
    "e-mail"     varchar(255) not null,
    name_user    varchar(50)  not null,
    surname_user varchar(50)  not null,
    phone        int          not null,
    date_time timestamp not null
);

create or replace function add_new_user_log()
returns trigger as
    $$
    begin
    insert into user_log(id_user, "e-mail", name_user, surname_user, phone, date_time)
    values (new.id_user, new."e-mail", new.name_user, new.surname_user, new.phone, current_timestamp);
    return new;
    end;
    $$
language 'plpgsql';

create trigger add_new_user after insert on
    "user"
    for each row
    execute procedure add_new_user_log();

INSERT INTO "user" ("e-mail", name_user, surname_user, phone)
VALUES ('Anton2.Tun@mail.com','Anton2','Tun2',951393865);

----------  2  --------------------
-- procedure pre pridanie mesta
create or replace procedure add_city(new_city varchar(50))
LANGUAGE plpgsql
AS $$ begin
        INSERT INTO city(city) values (new_city);
        end
$$;
call add_city('Kyiv');

----------  3  --------------------
--pridanie nového auta podľa mena, nie cez foreign key
CREATE or replace FUNCTION return_id_model(new_model varchar(50)) RETURNS integer AS $$
    select id_model from model where model.model= new_model
$$ LANGUAGE SQL;
CREATE or replace FUNCTION return_id_colour(new_color varchar(15)) RETURNS integer AS $$
    select id_colour from colour where color= new_color
$$ LANGUAGE SQL;
CREATE or replace FUNCTION return_id_transport_type(new_transport_type varchar(20)) RETURNS integer AS $$
    select id_transport_type from transport_type where type = new_transport_type
$$ LANGUAGE SQL;
CREATE or replace FUNCTION return_id_engine_capacity(new_engine_capacity float) RETURNS integer AS $$
    select id_engine_capacity from engine_capacity where capacity = new_engine_capacity
$$ LANGUAGE SQL;
CREATE or replace FUNCTION return_id_fuel(new_fuel varchar(20)) RETURNS integer AS $$
    select id_fuel from fuel where type = new_fuel
$$ LANGUAGE SQL;
CREATE or replace FUNCTION return_id_transmission(new_transmission varchar(20)) RETURNS integer AS $$
    select id_transmission from transmission where type = new_transmission;
$$ LANGUAGE SQL;
CREATE or replace FUNCTION return_id_body_type(new_body_type varchar(20)) RETURNS integer AS $$
    select id_body_type from body_type where type = new_body_type
$$ LANGUAGE SQL;
CREATE or replace FUNCTION return_id_user(new_user varchar(50)) RETURNS integer AS $$
    select id_user from "user" where name_user = new_user
$$ LANGUAGE SQL;
----
create or replace procedure add_auto(new_mileage int, new_price int, new_users int, new_colour
varchar(15),new_transport_type varchar(20),
 new_engine_capacity float, new_fuel varchar(20),new_transmission varchar(20),new_body_type varchar(20), new_model
varchar(50), new_year int)
LANGUAGE plpgsql
AS $$ begin
        INSERT INTO auto(mileage, price, users, id_colour_fk, id_transport_type_fk, id_engine_capacity_fk, id_fuel_fk,
id_transmission_fk, id_body_type_fk, id_model_fk, year_auto)
        values (new_mileage,new_price, new_users,return_id_colour(new_colour),return_id_transport_type(new_transport_type),
                return_id_engine_capacity(new_engine_capacity),return_id_fuel(new_fuel),return_id_transmission(new_transmission),
                return_id_body_type(new_body_type),return_id_model(new_model),new_year);
        end
$$;
```

SQL skript odovzdanie 3

```
create or replace function add_auto_log()
returns trigger as
    $$
    begin
    insert into auto_log(id_auto, date_time)
    values (new.id_auto, current_timestamp);
    return new;
    end;
    $$
language 'plpgsql';

create trigger add_new_auto after insert on
    auto
    for each row
    execute procedure add_auto_log();

create table auto_log(
    id_auto_log serial unique not null primary key ,
    id_auto      int not null unique ,
    date_time timestamp not null
);

call add_auto(180000, 13000, 3, 'Green', 'Moto', 1.1,
    'Gasoline', 'Automatic', 'Other', 'GSX-R750', 2019);

---------------  4  --------------------
--pri aktualizácii inzerátu, status = sold. reklamy sú odstránené z tabuľky "announcement" a pridal sa do tabuľky
"archive_announcement"
alter table  announcement
add status varchar(10) not null default ('not sold');

create table archive_announcement
(
    id_archive_announcement  serial unique not null primary key,
    description      varchar(255),
    publication_time date not null,
    photo            varchar(255) not null,
    id_user_fk       int,
    foreign key (id_user_fk) references "user"(id_user),
    id_city_fk       int  not null,
    foreign key (id_city_fk) references city(id_city),
    id_company_fk    int,
    foreign key (id_company_fk) references company(id_company),
    status varchar(10) not null default ('sold'),
    date_time_sold timestamp not null
);
--
create or replace function sold_announcement() returns trigger as
    $$
    begin
        insert into archive_announcement(description, publication_time, photo, id_user_fk, id_city_fk, id_company_fk, status,
date_time_sold)
        values (old.description, old.publication_time, old.photo, old.id_user_fk, old.id_city_fk, old.id_company_fk, 'sold',
current_timestamp);
        delete from announcement where status = 'sold';
        return old;
    end;
    $$
language 'plpgsql';
--
drop trigger sold_announcement on announcement;
create trigger sold_announcement after update of status
    on announcement
    for each row
    execute procedure sold_announcement();

INSERT INTO announcement (id_announcement, description, publication_time, photo, id_user_fk, id_city_fk, id_auto_fk, status)
VALUES (99, 'auto', '2022-04-08', 'C:\photo\announcement\99\foto1.png',1,2,18, 'not sold');

--update announcement set status='sold' where id_announcement = 99;
create or replace procedure set_sold_announcement(new_user_name varchar(50), new_id_announcement int )
language plpgsql
AS $$
    begin
        update announcement set status='sold' where id_announcement = new_id_announcement
                                        and id_user_fk = return_id_user(new_user_name);
    end
    $$;
call set_sold_announcement('Ruslan', 99);
```