ДОДАТОК В

**SQL-код бази даних у MySQL**

CREATE DATABASE Tickets\_booking;

USE Tickets\_booking;

# DROP DATABASE Tickets\_booking;

-- Створення таблиць

-- -----------------------------------------------------

-- Table Type\_transportations

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS Type\_transportations (

transportation\_ID INT AUTO\_INCREMENT NOT NULL,

type\_name VARCHAR(45) NOT NULL,

PRIMARY KEY (transportation\_ID));

-- -----------------------------------------------------

-- Table Firms

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS Firms (

firm\_ID INT NOT NULL AUTO\_INCREMENT,

idTransportation INT NOT NULL,

name VARCHAR(45) NOT NULL,

email VARCHAR(45) NOT NULL,

tel VARCHAR(20) NOT NULL,

regin\_activity VARCHAR(20) NULL,

PRIMARY KEY (firm\_ID),

FOREIGN KEY (idTransportation) REFERENCES Type\_transportations (transportation\_ID)

ON DELETE CASCADE

ON UPDATE CASCADE);

-- -----------------------------------------------------

-- Table Transport

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS Transport (

transport\_ID INT NOT NULL AUTO\_INCREMENT,

name VARCHAR(45) NOT NULL,

number VARCHAR(10) NOT NULL,

countplace INT NOT NULL,

PRIMARY KEY (transport\_ID));

CREATE TABLE Del\_transport LIKE Transport;

ALTER TABLE Del\_transport ADD COLUMN time DATETIME NULL;

-- -----------------------------------------------------

-- Table Flights

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS Flights (

flight\_ID INT NOT NULL AUTO\_INCREMENT,

idFirm INT NOT NULL,

idTransport INT NOT NULL,

f\_from VARCHAR(45) NULL,

f\_to VARCHAR(45) NULL,

f\_distance FLOAT NULL,

starttime DATETIME NOT NULL,

endtime DATETIME NULL,

status ENUM('Скасований', 'Відбувся', 'Очікується') DEFAULT 'Очікується',

PRIMARY KEY (flight\_ID),

FOREIGN KEY (idFirm) REFERENCES Firms(firm\_ID)

ON DELETE CASCADE

ON UPDATE CASCADE,

FOREIGN KEY (idTransport) REFERENCES Transport(transport\_ID)

ON DELETE CASCADE

ON UPDATE CASCADE);

-- -----------------------------------------------------

-- Table Customers

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS Customers (

customer\_ID INT NOT NULL AUTO\_INCREMENT,

c\_name VARCHAR(45) NOT NULL,

c\_birthday DATE NOT NULL,

PRIMARY KEY (customer\_ID));

-- -----------------------------------------------------

-- Table Type\_tickets

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS Type\_tickets (

typeticket\_ID INT NOT NULL AUTO\_INCREMENT,

type\_afterpay ENUM('Купе дорослий', 'Купе дитячий', 'Купе студентський', 'Плацкарт дорослий',

'Плацкарт дитячий', 'Плацкарт студентський', 'Бізнес клас', 'Економ клас', 'Перший клас', 'Автобусний',

'Автобусний студентський', 'Морський', 'Морський VIP', 'Морський економ') NOT NULL,

percent\_afterpay INT NOT NULL,

PRIMARY KEY (typeticket\_ID));

-- -----------------------------------------------------

-- Table Cities

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS Cities (

city\_ID INT NOT NULL AUTO\_INCREMENT,

city\_name VARCHAR(45) NOT NULL,

country VARCHAR(45) NOT NULL,

PRIMARY KEY (city\_ID));

-- -----------------------------------------------------

-- Table Flights\_has\_Cities

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS Flights\_has\_Cities (

idFlight INT NOT NULL,

idCity1 INT NOT NULL,

idCity2 INT NULL,

primaryID INT NOT NULL AUTO\_INCREMENT,

PRIMARY KEY (primaryID),

FOREIGN KEY (idFlight) REFERENCES Flights (flight\_ID)

ON DELETE CASCADE

ON UPDATE CASCADE,

FOREIGN KEY (idCity1) REFERENCES Cities (city\_ID)

ON DELETE CASCADE

ON UPDATE CASCADE,

FOREIGN KEY (idCity2) REFERENCES Cities (city\_ID)

ON DELETE CASCADE

ON UPDATE CASCADE);

-- -----------------------------------------------------

-- Table Airplane\_tickets

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS Airplane\_tickets (

a\_ticket\_ID INT NOT NULL AUTO\_INCREMENT,

at\_price DECIMAL NULL,

idType\_tickets INT NOT NULL,

Flights\_has\_Cities\_primaryID INT NOT NULL,

PRIMARY KEY (a\_ticket\_ID),

FOREIGN KEY (idType\_tickets) REFERENCES Type\_tickets (typeticket\_ID)

ON DELETE CASCADE

ON UPDATE CASCADE,

FOREIGN KEY (Flights\_has\_Cities\_primaryID) REFERENCES Flights\_has\_Cities (primaryID)

ON DELETE CASCADE

ON UPDATE CASCADE);

-- -----------------------------------------------------

-- Table Routes

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS Routes (

route\_ID INT NOT NULL AUTO\_INCREMENT,

Flights\_has\_Cities\_primaryID INT NOT NULL,

idCity INT NOT NULL,

PRIMARY KEY (route\_ID),

FOREIGN KEY (Flights\_has\_Cities\_primaryID) REFERENCES Flights\_has\_Cities (primaryID)

ON DELETE CASCADE

ON UPDATE CASCADE,

FOREIGN KEY (idCity) REFERENCES Cities (city\_ID)

ON DELETE CASCADE

ON UPDATE CASCADE);

-- -----------------------------------------------------

-- Table Train\_tickets

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS Train\_tickets (

t\_ticket\_ID INT NOT NULL AUTO\_INCREMENT,

tt\_price DECIMAL NULL,

idType\_tickets INT NOT NULL,

idRoute INT NOT NULL,

PRIMARY KEY (t\_ticket\_ID),

FOREIGN KEY (idType\_tickets) REFERENCES Type\_tickets (typeticket\_ID)

ON DELETE CASCADE

ON UPDATE CASCADE,

FOREIGN KEY (idRoute) REFERENCES Routes (route\_ID)

ON DELETE CASCADE

ON UPDATE CASCADE);

-- -----------------------------------------------------

-- Table Bus\_tickets

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS Bus\_tickets (

b\_ticket\_ID INT NOT NULL AUTO\_INCREMENT,

bt\_price DECIMAL NULL,

idType\_tickets INT NOT NULL,

idRoute INT NOT NULL,

PRIMARY KEY (b\_ticket\_ID),

FOREIGN KEY (idRoute) REFERENCES Routes (route\_ID)

ON DELETE CASCADE

ON UPDATE CASCADE,

FOREIGN KEY (idType\_tickets) REFERENCES Type\_tickets (typeticket\_ID)

ON DELETE CASCADE

ON UPDATE CASCADE);

-- -----------------------------------------------------

-- Table Ship\_tickets

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS Ship\_tickets (

s\_ticket\_ID INT NOT NULL AUTO\_INCREMENT,

st\_price DECIMAL NULL,

idType\_tickets INT NOT NULL,

Flights\_has\_Cities\_primaryID INT NOT NULL,

PRIMARY KEY (s\_ticket\_ID),

FOREIGN KEY (idType\_tickets) REFERENCES Type\_tickets (typeticket\_ID)

ON DELETE CASCADE

ON UPDATE CASCADE,

FOREIGN KEY (Flights\_has\_Cities\_primaryID) REFERENCES Flights\_has\_Cities (primaryID)

ON DELETE CASCADE

ON UPDATE CASCADE);

-- -----------------------------------------------------

-- Table Train\_tickets\_has\_Customers

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS Train\_tickets\_has\_Customers (

Train\_tickets\_t\_ticket\_ID INT NOT NULL,

Customers\_customer\_ID INT NOT NULL,

t\_booking\_time DATETIME NOT NULL,

t\_numwagon INT NOT NULL,

t\_seat INT NOT NULL,

comments VARCHAR(45) NULL,

PRIMARY KEY (Train\_tickets\_t\_ticket\_ID, Customers\_customer\_ID),

FOREIGN KEY (Train\_tickets\_t\_ticket\_ID) REFERENCES Train\_tickets (t\_ticket\_ID)

ON DELETE CASCADE

ON UPDATE CASCADE,

FOREIGN KEY (Customers\_customer\_ID) REFERENCES Customers (customer\_ID)

ON DELETE CASCADE

ON UPDATE CASCADE);

-- -----------------------------------------------------

-- Table Airplane\_tickets\_has\_Customers

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS Airplane\_tickets\_has\_Customers (

Airplane\_tickets\_a\_ticket\_ID INT NOT NULL,

Customers\_customer\_ID INT NOT NULL,

a\_booking\_time DATETIME NOT NULL,

at\_class VARCHAR(15) NULL,

a\_seat INT NOT NULL,

comments VARCHAR(45) NULL,

PRIMARY KEY (Airplane\_tickets\_a\_ticket\_ID, Customers\_customer\_ID),

FOREIGN KEY (Airplane\_tickets\_a\_ticket\_ID) REFERENCES Airplane\_tickets (a\_ticket\_ID)

ON DELETE CASCADE

ON UPDATE CASCADE,

FOREIGN KEY (Customers\_customer\_ID) REFERENCES Customers (customer\_ID)

ON DELETE CASCADE

ON UPDATE CASCADE);

-- -----------------------------------------------------

-- Table Bus\_tickets\_has\_Customers

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS Bus\_tickets\_has\_Customers (

Bus\_tickets\_b\_ticket\_ID INT NOT NULL,

Customers\_customer\_ID INT NOT NULL,

b\_booking\_time DATETIME NOT NULL,

b\_seat INT NOT NULL,

comments VARCHAR(45) NULL,

PRIMARY KEY (Bus\_tickets\_b\_ticket\_ID, Customers\_customer\_ID),

FOREIGN KEY (Bus\_tickets\_b\_ticket\_ID) REFERENCES Bus\_tickets (b\_ticket\_ID)

ON DELETE CASCADE

ON UPDATE CASCADE,

FOREIGN KEY (Customers\_customer\_ID) REFERENCES Customers (customer\_ID)

ON DELETE CASCADE

ON UPDATE CASCADE);

-- -----------------------------------------------------

-- Table Ship\_tickets\_has\_Customers

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS Ship\_tickets\_has\_Customers (

Ship\_tickets\_s\_ticket\_ID INT NOT NULL,

Customers\_customer\_ID INT NOT NULL,

s\_booking\_time DATETIME NOT NULL,

s\_class VARCHAR(15) NULL,

s\_seat INT NOT NULL,

comments VARCHAR(45) NULL,

PRIMARY KEY (Ship\_tickets\_s\_ticket\_ID, Customers\_customer\_ID),

FOREIGN KEY (Ship\_tickets\_s\_ticket\_ID) REFERENCES Ship\_tickets (s\_ticket\_ID)

ON DELETE CASCADE

ON UPDATE CASCADE,

FOREIGN KEY (Customers\_customer\_ID) REFERENCES Customers (customer\_ID)

ON DELETE CASCADE

ON UPDATE CASCADE);

-- -----------------------------------------------------

-- Table Transport\_has\_Type\_tickets

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS Transport\_has\_Type\_tickets (

Transport\_idTransport INT NOT NULL,

Type\_ticketa\_idType\_tickets INT NOT NULL,

count\_seats\_type INT NOT NULL,

PRIMARY KEY (Transport\_idTransport, Type\_ticketa\_idType\_tickets),

FOREIGN KEY (Transport\_idTransport) REFERENCES Transport (transport\_ID)

ON DELETE CASCADE

ON UPDATE CASCADE,

FOREIGN KEY (Type\_ticketa\_idType\_tickets) REFERENCES Type\_tickets (typeticket\_ID)

ON DELETE CASCADE

ON UPDATE CASCADE);

-- -----------------------------------------------------

-- ЗАПОВНЕННЯ ТАБЛИЦЬ

-- -----------------------------------------------------

BEGIN ;

INSERT INTO Type\_transportations(type\_name)

VALUES ('Повітряне'),('Колійне'),('Водне'),('Наземне');

COMMIT ;

SELECT \* FROM Type\_transportations;

BEGIN ;

INSERT INTO Firms(idTransportation, name, email, tel, regin\_activity)

VALUES (1,'МАУ','uia@flyuia.com','+38 (044) 581-50-50','Україна'),

(1,'KLM','KLM.Ukraine@klm.nl','+31 (0) 20 649 91 23','Нідерланди'),

(1,'Emirates Airlines','pr@emirates.com','600 555 555','ОАЕ'),

(1,'British Airways','ba.ukraine@ba.com','+44 (20) 8738 5050','Англія'),

(1,'Wizz Air','info@wizzair.com','+380 89 320 25 33','Венгрія'),

(1,'Turkish Airlines','ievsales@thy.com','+90 444 0 8491','Туреччина'),

(1,'Lufthansa','info@lufthansa.com','+49 (69) 696 46 01','Німеччина'),

(2,'Укрзалізниця','cgk@uz.gov.ua','0-800-50-311',''),

(2,'Polrail Service','office@polrail.com','+48 52 332 57 81',''),

(2,'CFR Călători','presa@cfrcalatori.ro','0800.88.44.44',''),

(2,'БЖД','ns@rw.by','(+375 17) 225 49 46','Білорусь'),

(3,'Costa Cruises','customercare@us.costa.it','1-800-462-6782',''),

(3,'MSC','UA539-odessa@msc.com','+380 48 784 7272',''),

(3,'Royal Caribbean','cliente@rccl.com.br','(888) 724-7447',''),

(3,'Celebrity Cruises','CelebrityEngagementCenter@celebrity.com','00 1 305-341-0205',''),

(4,'AUTOLUX','help@autolux.ua','+38 044 594 95 00',''),

(4,'Gunsel','info@gunsel.com.ua','+38 (044) 525-45-05',''),

(4,'EuroClub','euroclubbus@gmail.com','+38(044)486 79 00',''),

(4,'Orionbus','rionbilet@gmail.com','050 010 0104',''),

(4,'Ecolines','help@ecolines.ua','+38 044 594 90 10',''),

(4,'EAST WEST EUROLINES','support@ewe.ua','+380988154444',''),

(4,'TransTempo','transtempo@ukr.net','+38 (067) 467-44-77','');

COMMIT ;

# SELECT \* FROM Firms;

BEGIN ;

INSERT INTO Customers(c\_name, c\_birthday)

VALUES ('Тарасович Ратимир Вадимович', '1952-08-10'),

('Ніколенко Ромашка Зорянівна', '1963-05-17'),

('Кириленко Муховіст Полянович', '1987-03-19'),

('Алиськевич Мар''ян Устимович', '1981-07-27'),

('Савка Щедра Фролівна', '1976-05-04'),

('Вітренко Власт Денисович', '1995-11-24'),

('Смолій Ярослав Зорянович', '1997-09-11'),

('Міняйло Йосифата Ярославівна', '1979-11-24'),

('Семків Йоган Артемович', '1991-02-05'),

('Нечитайло Данко Мстиславович', '2000-08-14'),

('Сомко Яромира Остапівна', '1994-09-17'),

('Токарчук Родіон Найденович', '1954-12-21'),

('Соломченко Хотимир Златович', '1996-08-09'),

('Лупійчук Дан Жданович', '1982-07-25'),

('Лазорський Никифор Олександрович', '1978-06-01'),

('Довгань Гаїна Тимурівна', '1977-06-30'),

('Юденко Олесь Денисович', '1967-12-17'),

('Пастушенко Ніна Ростиславівна', '1969-03-26'),

('Пустовойт Колодар Богданович', '1972-05-30'),

('Боєчко Лев Антонович', '1963-10-26'),

('Гриневецький Хотян Сарматович', '1995-07-15'),

('Барвінський Антон Охримович', '1991-04-09'),

('Кущ Югина Арсенівна', '1958-10-21'),

('Замора Єгор Жданович', '1956-07-30'),

('Німчук Злотан Найденович', '1982-03-12'),

('Німченко Йосифата Борисівна', '1956-05-21'),

('Ганущак Фауст Антонович', '1958-10-09'),

('Смішко Іларіон Ігорович', '1979-10-30'),

('Лещинська Жозефіна Августинівна', '1951-12-05'),

('Медяник Йозеф Богуславович', '1990-03-04'),

( 'Тихий Тихон Юхимович', '1977-04-30'),

( 'Мулярчук Хвалимир Остапович', '1994-01-07'),

( 'Мисько Живосил Чеславович', '1951-03-11'),

( 'Дурдинець Триріг Любомирович', '1971-07-10'),

( 'Магура Ізяслав Азарович', '1972-03-22'),

( 'Вінтоняк Альберт Пилипович', '1980-01-11'),

( 'Чалий Йозеф Костянтинович', '1952-10-24'),

( 'Могиленко Йосип Сарматович', '1991-07-10'),

( 'Коструба Щастислав Федорович', '1996-08-08'),

( 'Павлович Гладко Пилипович', '1963-09-12'),

( 'Петрицький Зборислав Давидович', '1950-04-18'),

( 'Сімович Цвітан Августинович', '1957-07-29'),

( 'Мацелюх Євстафій Августинович', '1953-09-01'),

( 'Чубатий Івантослав Давидович', '1960-10-21'),

( 'Сорока Юрій Гордиславович', '1993-06-12'),

( 'Вайда Турбрід Драганович', '1991-09-28'),

( 'Мухопад Царук Адріанович', '1982-04-20'),

( 'Бурбан Вернислав Артемович', '1993-03-05'),

( 'Штинь Атрей Русланович', '1956-11-19'),

( 'Деркач Лютобор Романович', '1966-07-25');

COMMIT ;

# SELECT \* FROM Customers;

BEGIN ;

INSERT INTO Type\_tickets (type\_afterpay, percent\_afterpay)

VALUES ( 'Купе дорослий', 50),

( 'Купе студентський', 30),

( 'Купе дитячий', 10),

( 'Плацкарт дорослий', 0),

( 'Плацкарт дитячий', -30),

( 'Плацкарт студентський', -50),

( 'Бізнес клас', 150),

( 'Економ клас', -10),

( 'Перший клас', 50),

( 'Автобусний', 0),

( 'Автобусний студентський', -20),

( 'Морський', 0),

( 'Морський VIP', 75),

( 'Морський економ',-15);

COMMIT ;

# SELECT \* FROM Type\_tickets;

BEGIN ;

INSERT INTO Cities (country, city\_name)

VALUES ('Молдова', 'Кишинів'),

('Греція', 'Афіни'),

('Хорватія', 'Загреб'),

('Швеція', 'Стокгольм'),

('Португалія', 'Лісабон'),

('Словаччина', 'Братислава'),

('Італія', 'Рим'),

('Македонія', 'Скоп’є'),

('Андорра', 'Андорра-ла-Велья'),

('Бельгія', 'Брюссель'),

('Чорногорія', 'Подгориця'),

('Данія', 'Копенгаген'),

('Ліхтенштейн', 'Вадуц'),

('Люксембург', 'Люксембург'),

('Великобританія', 'Лондон'),

('Словенія', 'Любляна'),

('Мальта', 'Валлетта'),

('Сан-Марино', 'Сан-Марино'),

('Польща', 'Варшава'),

('Німеччина', 'Берлін'),

('Сербія', 'Белград'),

('Швейцарія', 'Берн'),

('Чехія', 'Прага'),

('Литва', 'Вільнюс'),

('Ірландія', 'Дублін'),

('Албанія', 'Тирана'),

('Болгарія', 'Софія'),

('Ісландія', 'Рейк’явік'),

('Франція', 'Париж'),

('Білорусь', 'Мінськ'),

('Боснія і Герцеговина', 'Сараєво'),

('Румунія', 'Бухарест'),

('Фінляндія', 'Гельсінкі'),

('Ватикан', 'Ватикан'),

('Норвегія', 'Осло'),

('Австрія', 'Відень'),

('Латвія', 'Рига'),

('Іспанія', 'Мадрид'),

('Угорщина', 'Будапешт'),

('Монако', 'Монако'),

('Естонія', 'Таллінн'),

('Нідерланди', 'Амстердам'),

('Україна','Вінниця'),

('Україна','Луцьк'),

('Україна','Дніпро'),

('Україна','Донецьк'),

('Україна','Житомир'),

('Україна','Ужгород'),

('Україна','Запоріжжя'),

('Україна','Івано-Франківськ'),

('Україна','Київ'),

('Україна','Кропивницький'),

('Україна','Львів'),

('Україна','Миколаїв'),

('Україна','Луганськ'),

('Україна','Одеса'),

('Україна','Полтава'),

('Україна','Рівне'),

('Україна','Суми'),

('Україна','Тернопіль'),

('Україна','Харків'),

('Україна','Херсон'),

('Україна','Хмельницький'),

('Україна','Черкаси'),

('Україна','Чернівці'),

('Україна','Чернігів');

COMMIT ;

# SELECT \* FROM Cities;

delimiter //

CREATE TRIGGER saveDelTransport BEFORE DELETE ON Transport

FOR EACH ROW

BEGIN

INSERT INTO Del\_transport(transport\_ID, name, number, countplace, time)

VALUES(OLD.transport\_ID, OLD.name, OLD.number, OLD.countplace, NOW());

END;//

delimiter ;

/\*BEGIN ;

DELETE FROM Transport WHERE transport\_ID < 5;

ROLLBACK ;

SELECT \* FROM Transport;

SELECT \* FROM Del\_transport;\*/

BEGIN ;

INSERT INTO Transport(name, number, countplace)

VALUES ( 'Boeing', '777', 250),

( 'Airbus', 'A340', 170),

( 'Airbus', 'A330', 125),

( 'Boeing', '747', 245),

( 'Boeing Next Generation', '737', 290),

( 'Boeing Classic', '737', 230),

( 'AN', '225', 115),

( 'AN', '13', 134),

( 'AN', '74', 100),

( 'AN', '275', 110),

( 'AN', '132G', 265),

( 'AN', '28', 330),

( 'AN', '178', 110),

( 'Douglas', 'DC-1', 285),

( 'Dassault Falcon', '900', 280),

( 'Farman Goliath', 'F.60', 270),

( 'Dassault Falcon', '7X', 260),

( 'Богдан', 'AE5501AA', 25),

( 'Neoplan', 'AE5740KP', 30),

( 'Neoplan', 'AX3116BM', 40),

( 'ЕТАЛОН', 'BM6175CM', 30),

( 'Богдан', 'BC3458MK', 25),

( 'ЕТАЛОН', 'BI7837EA', 28),

( 'Богдан', 'AB1994IA', 25),

( 'ISUZU', 'AE1095AA', 32),

( 'Богдан', 'AE1095AA', 25),

( 'ЕТАЛОН', 'BH4091AA', 28),

( 'ЕТАЛОН', 'BC4780MB', 28),

( 'ЕТАЛОН', 'BO5913CP', 28),

( 'Богдан', 'AE4858OX', 25),

( 'ISUZU', 'BC6097MK', 28),

( 'Neoplan', 'KA2376BT', 30),

( 'ISUZU', 'AT7929EP', 26),

( 'GULERYUZ', 'BK3137HH', 24),

( 'Богдан', 'AC1501EO', 25),

( 'ISUZU', 'AO2753EP', 29),

( 'ISUZU', 'KA1002OP', 34),

( 'Богдан', 'BT1552CC', 25),

( 'ISUZU', 'AB2775HT', 33),

( 'Neoplan', 'BI7798EX', 30),

( 'GULERYUZ', 'BC9664MB', 22),

( 'ЕТАЛОН', 'AM9254EO', 28),

( 'Neoplan', 'AT6426EP', 30),

( 'GULERYUZ', 'BX5339EM', 36),

( 'GULERYUZ', 'BI8403EE', 28),

( 'Neoplan', 'BH2137OH', 32),

( 'ЕТАЛОН', 'AA4243PT', 30),

( 'Neoplan', 'BC5474MK', 32),

( 'ISUZU', 'BM1049CM', 30),

( 'Neoplan', 'AX5109HX', 30),

( 'Південний експрес', '686/685', 686),

( 'Північний експрес','784/785', 841),

( 'Подільський експрес ','103/104', 670),

( 'Приазов`я', '989/988', 898),

( 'Промінь', '421/420', 309),

( 'Рось', '117/118', 602),

( 'Севастополець', '542/543', 634),

( 'Таврія', '835/834', 455),

( 'Хаджибей', '683/682', 643),

( 'Чайка', '093/094', 542),

( 'Enchantment of the Seas', '', 2405),

( 'Symphony of the Seas', '', 1300),

( 'Celebrity Xploration', '', 845),

( 'Celebrity Xpedition', '', 5530),

( 'Forse le Carib', '', 570),

( 'Porco Fugo', '', 240);

COMMIT ;

/\*SELECT \* FROM Transport;

SELECT \* FROM Type\_tickets;

SELECT \* FROM Firms;\*/

# літаки(1-17), поїзда(51-60), кораблі(61-66), автобуси(18-50)

# літаки(1-7), поїзда(8-11), кораблі(12-15), автобуси(16-22)

# BEGIN;

BEGIN ;

INSERT INTO Transport\_has\_Type\_tickets(transport\_idtransport, type\_ticketa\_idtype\_tickets, count\_seats\_type)

VALUES (1, 7, 50),

(1, 8, 75),

(1, 9, 125),

(2, 7, 30),

(2, 8, 100),

(2, 9, 40),

(3, 7, 25),

(3, 8, 75),

(3, 9, 25),

(4, 7, 45),

(4, 8, 150),

(4, 9, 50),

(5, 7, 40),

(5, 8, 200),

(5, 9, 50),

(6, 7, 30),

(6, 8, 150),

(6, 9, 50),

(7, 7, 15),

(7, 8, 80),

(7, 9, 20),

(8, 7, 14),

(8, 8, 100),

(8, 9, 20),

(9, 7, 10),

(9, 8, 50),

(9, 9, 40),

(10, 7, 20),

(10, 8, 60),

(10, 9, 30),

(11, 7, 25),

(11, 8, 200),

(11, 9, 40),

(12, 7, 80),

(12, 8, 150),

(12, 9, 100),

(13, 7, 10),

(13, 8, 75),

(13, 9, 25),

(14, 7, 35),

(14, 8, 200),

(14, 9, 50),

(15, 7, 30),

(15, 8, 200),

(15, 9, 50),

(16, 7, 30),

(16, 8, 200),

(16, 9, 40),

(17, 7, 10),

(17, 8, 150),

(17, 9, 100),

(18, 10, 20),

(18, 11, 5),

(19, 10, 25),

(19, 11, 5),

(20, 10, 30),

(20, 11, 10),

(21, 10, 25),

(21, 11, 5),

(22, 10, 20),

(22, 11, 5),

(23, 10, 25),

(23, 11, 3),

(24, 10, 20),

(24, 11, 5),

(25, 10, 30),

(25, 11, 2),

(26, 10, 20),

(26, 11, 5),

(27, 10, 25),

(27, 11, 3),

(28, 10, 25),

(28, 11, 3),

(29, 10, 25),

(29, 11, 3),

(30, 10, 20),

(30, 11, 5),

(31, 10, 20),

(31, 11, 8),

(32, 10, 25),

(32, 11, 5),

(33, 10, 20),

(33, 11, 6),

(34, 10, 20),

(34, 11, 4),

(35, 10, 20),

(35, 11, 5),

(36, 10, 20),

(36, 11, 9),

(37, 10, 30),

(37, 11, 4),

(38, 10, 20),

(38, 11, 5),

(39, 10, 30),

(39, 11, 3),

(40, 10, 25),

(40, 11, 5),

(41, 10, 20),

(41, 11, 2),

(42, 10, 20),

(42, 11, 8),

(43, 10, 25),

(43, 11, 5),

(44, 10, 30),

(44, 11, 6),

(45, 10, 25),

(45, 11, 3),

(46, 10, 30),

(46, 11, 2),

(47, 10, 30),

(47, 11, 0),

(48, 10, 30),

(48, 11, 2),

(49, 10, 25),

(49, 11, 5),

(50, 10, 25),

(50, 11, 5),

(51, 1, 200),

(51, 2, 36),

(51, 3, 50),

(51, 4, 300),

(51, 5, 50),

(51, 6, 50),

(52, 1, 241),

(52, 2, 50),

(52, 3, 50),

(52, 4, 350),

(52, 5, 75),

(52, 6, 75),

(53, 1, 270),

(53, 2, 25),

(53, 3, 25),

(53, 4, 250),

(53, 5, 50),

(53, 6, 50),

(54, 1, 398),

(54, 2, 50),

(54, 3, 50),

(54, 4, 200),

(54, 5, 50),

(54, 6, 50),

(55, 1, 9),

(55, 2, 5),

(55, 3, 5),

(55, 4, 200),

(55, 5, 50),

(55, 6, 40),

(56, 1, 102),

(56, 2, 75),

(56, 3, 65),

(56, 4, 200),

(56, 5, 80),

(56, 6, 80),

(57, 1, 34),

(57, 2, 95),

(57, 3, 55),

(57, 4, 300),

(57, 5, 75),

(57, 6, 75),

(58, 1, 55),

(58, 2, 50),

(58, 3, 50),

(58, 4, 200),

(58, 5, 50),

(58, 6, 50),

(59, 1, 143),

(59, 2, 50),

(59, 3, 50),

(59, 4, 300),

(59, 5, 50),

(59, 6, 50),

(60, 1, 142),

(60, 2, 50),

(60, 3, 50),

(60, 4, 200),

(60, 5, 50),

(60, 6, 50),

(61,12, 1500),

(61,13, 405),

(61,14, 500),

(62,12, 750),

(62,13, 250),

(62,14, 300),

(63,12, 500),

(63,13, 145),

(63,14, 200),

(64,12, 4530),

(64,13, 300),

(64,14, 700),

(65,12, 370),

(65,13, 100),

(65,14, 100),

(66,12, 140),

(66,13, 50),

(66,14, 50);

COMMIT ;

# SELECT \* FROM Transport\_has\_Type\_tickets;

# 1.

delimiter //

CREATE FUNCTION getHour (inHour FLOAT)

RETURNS TIME

DETERMINISTIC

READS SQL DATA

BEGIN

DECLARE result VARCHAR(15);

IF ~~(ROUND(inHour,2)\*100)%100 >= 60

THEN SET result = CONCAT(FLOOR(inHour) + 1, ':', (~~(ROUND(inHour,2)\*100)%100) - 60, ':00');

ELSE SET result = CONCAT(FLOOR(inHour), ':', ~~(ROUND(inHour,2)\*100)%100, ':00');

END IF;

RETURN result;

END;//

delimiter ;

# 2.

delimiter //

CREATE FUNCTION randDATETIME (from\_date DATETIME, to\_date DATETIME)

RETURNS DATETIME

DETERMINISTIC

READS SQL DATA

BEGIN

DECLARE result DATETIME;

SET result = FROM\_UNIXTIME(RAND()\*(UNIX\_TIMESTAMP(to\_date)-

UNIX\_TIMESTAMP(from\_date))+UNIX\_TIMESTAMP(from\_date));

RETURN result;

END;//

delimiter ;

# DROP FUNCTION randDATETIME;

# 3.

DELIMITER $$

CREATE FUNCTION ExtractNumber(in\_string VARCHAR(50))

RETURNS INT

NO SQL

BEGIN

DECLARE ctrNumber VARCHAR(50);

DECLARE finNumber VARCHAR(50) DEFAULT '';

DECLARE sChar VARCHAR(1);

DECLARE inti INTEGER DEFAULT 1;

IF LENGTH(in\_string) > 0

THEN WHILE(inti <= LENGTH(in\_string)) DO

SET sChar = SUBSTRING(in\_string, inti, 1);

SET ctrNumber = FIND\_IN\_SET(sChar, '0,1,2,3,4,5,6,7,8,9');

IF ctrNumber > 0

THEN SET finNumber = CONCAT(finNumber, sChar);

END IF;

SET inti = inti + 1;

END WHILE;

RETURN CAST(finNumber AS UNSIGNED);

ELSE

RETURN 0;

END IF;

END; $$

DELIMITER ;

# 4. Процедура, яка обчислює вартість авіаквитка

DELIMITER $$

CREATE PROCEDURE `calc\_price\_at`(OUT new\_price DECIMAL, IN new\_FhCpID INTEGER, IN new\_idTt INTEGER)

BEGIN

SET new\_price = 4 \* (SELECT f\_distance FROM Flights INNER JOIN Flights\_has\_Cities FhC

on Flights.flight\_ID = FhC.idFlight WHERE primaryID = new\_FhCpID) +

((SELECT f\_distance FROM Flights INNER JOIN Flights\_has\_Cities FhC

on Flights.flight\_ID = FhC.idFlight WHERE idFlight = new\_FhCpID) \*

(SELECT percent\_afterpay/100 FROM Type\_tickets

WHERE new\_idTt = Type\_tickets.typeticket\_ID));

END; $$

DELIMITER ;

# DROP PROCEDURE calc\_price\_at;

# 5. Процедура, яка обчислює вартість морського квитка

DELIMITER $$

CREATE PROCEDURE `calc\_price\_st`(OUT new\_price DECIMAL, IN new\_FhCpID INTEGER, IN new\_idTt INTEGER)

BEGIN

SET new\_price = 5 \* (SELECT f\_distance FROM Flights INNER JOIN Flights\_has\_Cities FhC

on Flights.flight\_ID = FhC.idFlight WHERE primaryID = new\_FhCpID) +

((SELECT f\_distance FROM Flights INNER JOIN Flights\_has\_Cities FhC

on Flights.flight\_ID = FhC.idFlight WHERE idFlight = new\_FhCpID) \*

(SELECT percent\_afterpay/100 FROM Type\_tickets

WHERE new\_idTt = Type\_tickets.typeticket\_ID));

END; $$

DELIMITER ;

# 6. Процедура, яка обчислює вартість автобусного квитка

DELIMITER $$

CREATE PROCEDURE `calc\_price\_bt`(INOUT new\_price DECIMAL, IN new\_idTt INTEGER)

BEGIN

SET new\_price = new\_price + new\_price \*(SELECT percent\_afterpay/100

FROM Type\_tickets WHERE typeticket\_ID = new\_idTt);

END; $$

DELIMITER ;

/\*CALL calc\_price\_st(@test,32,12);

SELECT ROUND(@test,2) AS "Ціна", f\_distance AS "Відстань", CONCAT('Одеса', ' - ', f\_to) AS "Рейс"

FROM Flights INNER JOIN Flights\_has\_Cities FhC

ON Flights.flight\_ID = FhC.idFlight

WHERE idFlight = 32;\*/

# SELECT \* FROM Ship\_tickets;

# 7. Процедура, яка обчислює вартість автобусного квитка

DELIMITER $$

CREATE PROCEDURE `calc\_price\_tt`(INOUT new\_price DECIMAL, IN new\_idTt INTEGER)

BEGIN

SET new\_price = new\_price/2 + (new\_price/2)\*(SELECT percent\_afterpay/100

FROM Type\_tickets WHERE typeticket\_ID = new\_idTt);

END; $$

DELIMITER ;

/\*SET @test := 394;

CALL calc\_price\_tt(@test,1);

SELECT ROUND(@test,2) AS "Ціна", f\_distance AS "Відстань", CONCAT('Одеса', ' - ', f\_to) AS "Рейс"

FROM Flights INNER JOIN Flights\_has\_Cities FhC

ON Flights.flight\_ID = FhC.idFlight

WHERE idFlight = 31;\*/

# SELECT \* FROM Train\_tickets;

# 8. Процедура для пошуку маршрутів

DELIMITER $$

CREATE PROCEDURE `find\_routes`(IN from\_city VARCHAR(45), IN to\_city VARCHAR(45))

BEGIN

DECLARE flight\_from VARCHAR(45);

DECLARE flight\_to VARCHAR(45);

DECLARE route\_from VARCHAR(45);

DECLARE route\_to VARCHAR(45);

DECLARE flight\_time DATETIME;

DECLARE myCondition BOOLEAN;

DECLARE success BOOLEAN DEFAULT FALSE;

DECLARE myCursor CURSOR FOR SELECT f\_from, f\_to, f\_from, city\_name, starttime

FROM Flights INNER JOIN Flights\_has\_Cities FhC

ON Flights.flight\_ID = FhC.idFlight INNER JOIN Routes R

ON FhC.primaryID = R.Flights\_has\_Cities\_primaryID INNER JOIN Cities C

ON C.city\_ID = R.idCity INNER JOIN Firms F

ON Flights.idFirm = F.firm\_ID INNER JOIN Type\_transportations Tt

ON F.idTransportation = Tt.transportation\_ID;

DECLARE CONTINUE HANDLER FOR NOT FOUND SET myCondition = FALSE;

OPEN myCursor;

SET myCondition = TRUE;

WHILE myCondition DO

FETCH myCursor INTO flight\_from, flight\_to, route\_from, route\_to, flight\_time;

IF(route\_from = from\_city AND route\_to = to\_city) THEN

SELECT flight\_time AS "Час відправлення", CONCAT(flight\_from, ' - ', flight\_to) AS "Рейс",

CONCAT(route\_from, ' - ', route\_to) AS "Маршрут";

SET success = TRUE;

END IF;

END WHILE;

CLOSE myCursor;

IF NOT success

THEN SELECT 'Маршрутів не знайдено!' AS "Повідомлення";

END IF;

END; $$

DELIMITER ;

# DROP PROCEDURE find\_routes;

# CALL find\_routes('Львів', 'Київ');

/\*SELECT type\_name, CONCAT(f\_from, ' - ', f\_to) AS Flight, CONCAT(f\_from, ' - ', city\_name) AS Маршрут

FROM Flights INNER JOIN Flights\_has\_Cities FhC

ON Flights.flight\_ID = FhC.idFlight INNER JOIN Routes R

ON FhC.primaryID = R.Flights\_has\_Cities\_primaryID INNER JOIN Cities C

ON C.city\_ID = R.idCity INNER JOIN Firms F

ON Flights.idFirm = F.firm\_ID INNER JOIN Type\_transportations Tt

ON F.idTransportation = Tt.transportation\_ID;\*/

# 9. Процедура для пошуку рейсів за часом відправлення

DELIMITER $$

CREATE PROCEDURE `find\_flight`(IN flight\_datetime DATETIME)

BEGIN

DECLARE flight\_from VARCHAR(45);

DECLARE flight\_to VARCHAR(45);

DECLARE flight\_distance FLOAT;

DECLARE flight\_startTime DATETIME;

DECLARE flight\_endTime DATETIME;

DECLARE myCondition BOOLEAN;

DECLARE success BOOLEAN DEFAULT FALSE;

DECLARE myCursor CURSOR FOR SELECT f\_from, f\_to, f\_distance, starttime, endtime FROM Flights;

DECLARE CONTINUE HANDLER FOR NOT FOUND SET myCondition = FALSE;

OPEN myCursor;

SET myCondition = TRUE;

WHILE myCondition DO

FETCH myCursor INTO flight\_from, flight\_to, flight\_distance, flight\_startTime, flight\_endTime;

IF(flight\_startTime = flight\_datetime) THEN

SELECT 'Точний збіг' AS ' ', flight\_from, flight\_to, flight\_distance, flight\_startTime, flight\_endTime;

SET success = TRUE;

ELSEIF(DATE(flight\_startTime) = DATE(flight\_datetime)) THEN

SELECT 'Схожий рейс' AS 'Точних збігів не знайдено!', flight\_from, flight\_to, flight\_distance, flight\_startTime, flight\_endTime;

SET success = TRUE;

END IF;

END WHILE;

CLOSE myCursor;

IF NOT success

THEN SELECT 'Рейсів не знайдено!' AS "Повідомлення";

END IF;

END; $$

DELIMITER ;

# DROP PROCEDURE find\_flight;

/\*CALL find\_flight('2021-08-09 20:18:56');

CALL find\_flight('2021-08-09 00:00:00');\*/

# SELECT \* FROM Flights;

# 10. Процедура, яка обчислює час прибуття рейса

DELIMITER $$

CREATE PROCEDURE `calc\_end\_time`(OUT flight\_endTime DATETIME, IN flight\_distance FLOAT, IN flight\_startTime DATETIME,

IN avg\_speed FLOAT)

BEGIN

SET flight\_endTime = ADDTIME(flight\_startTime, getHour(flight\_distance/avg\_speed));

END; $$

DELIMITER ;

# DROP PROCEDURE calc\_end\_time;

# SELECT \* FROM Flights;

/\*CALL calc\_end\_time(@test, 488, '2021-10-24 21:22:46', 50);

SELECT 50 AS avg\_speed, 488 AS distance, '2021-10-24 21:22:46' AS startTime, @test AS endTime;\*/

delimiter //

CREATE TRIGGER setEndTimeFlight BEFORE INSERT ON Flights

FOR EACH ROW

BEGIN

CASE

WHEN NEW.idFirm <= 7

THEN CALL calc\_end\_time(NEW.endtime, NEW.f\_distance,

NEW.starttime, 800);

WHEN NEW.idFirm > 7 AND NEW.idFirm <= 11

THEN CALL calc\_end\_time(NEW.endtime, NEW.f\_distance,

NEW.starttime, 50);

WHEN NEW.idFirm > 11 AND NEW.idFirm <= 15

THEN CALL calc\_end\_time(NEW.endtime, NEW.f\_distance,

NEW.starttime, 22);

WHEN NEW.idFirm > 15

THEN CALL calc\_end\_time(NEW.endtime, NEW.f\_distance,

NEW.starttime, 65);

END CASE;

END;//

delimiter ;

# DROP TRIGGER setEndTimeFlight;

delimiter //

CREATE TRIGGER checkDistance BEFORE INSERT ON Flights

FOR EACH ROW

BEGIN

CASE

WHEN NEW.f\_distance <= 0

THEN SIGNAL sqlstate '45001' SET message\_text = 'Invalid distance!';

WHEN NEW.f\_from REGEXP '[0-9]' OR NEW.f\_to REGEXP '[0-9]'

THEN SIGNAL sqlstate '45001' SET message\_text = 'This string can`t include digits!';

ELSE BEGIN END;

END CASE;

END;//

delimiter ;

# DROP TRIGGER checkDistance;

BEGIN ;

INSERT INTO Flights(idFirm, f\_from, f\_to, f\_distance, idTransport, starttime)

VALUES (8, 'Київ', 'Дніпро', -488, 51, '2021-10-24 21:22:46');

ROLLBACK ;

delimiter //

CREATE TRIGGER checkStatus BEFORE UPDATE ON Flights

FOR EACH ROW

BEGIN

CASE

WHEN CURRENT\_TIMESTAMP < NEW.endtime AND OLD.status = 'Відбувся'

THEN SET NEW.status = 'Очікується';

WHEN NEW.endtime < CURRENT\_TIMESTAMP AND OLD.status = 'Очікується'

THEN SET NEW.status = 'Відбувся';

END CASE;

END;//

delimiter ;

BEGIN ;

INSERT INTO Flights(idFirm, f\_from, f\_to, f\_distance, idTransport, starttime)

VALUES (8, 'Київ', 'Дніпро', 488, 51, '2021-10-24 21:22:46'),

(10, 'Херсон', 'Миколаїв', 80, 57, '2021-05-08 14:01:34'),

(11, 'Київ', 'Луганськ', 922, 6, '2021-08-09 20:18:56'),

(14, 'Одеса', 'Афіни', 1786, 61, '2021-06-15 00:12:53'),

(9, 'Рівне', 'Луганськ', 1252, 56, '2021-09-13 14:36:39'),

(2, 'Київ', 'Тирана', 1300, 5, '2021-01-08 21:21:34'),

(3, 'Рівне', 'Монако', 1631, 15, '2021-04-21 00:32:33'),

(1, 'Житомир', 'Люксембург', 1608, 8, '2021-10-20 17:46:29'),

(1, 'Чернівці', 'Бухарест', 413, 16, '2021-09-15 10:20:25'),

(8, 'Полтава', 'Миколаїв', 463, 52, '2021-09-05 16:22:17'),

(10, 'Львів', 'Харків', 1109, 53, '2021-09-05 01:47:01'),

(11, 'Черкаси', 'Львів', 727, 59, '2021-12-04 13:10:36'),

(8, 'Ужгород', 'Чернівці', 432, 60, '2021-03-02 15:16:33'),

(7, 'Донецьк', 'Мінськ', 945, 1, '2021-11-04 00:17:22'),

(11, 'Хмельницький', 'Луганськ', 1247, 58, '2021-09-15 14:12:39'),

(9, 'Івано-Франківськ', 'Вінниця', 381, 52, '2021-09-12 05:25:12'),

(6, 'Івано-Франківськ', 'Валлетта', 1674, 17, '2021-09-19 12:36:24'),

(1, 'Запоріжжя', 'Івано-Франківськ', 1147, 3, '2021-08-08 13:54:20'),

(5, 'Київ', 'Вільнюс', 584, 9, '2021-11-05 17:59:33'),

(1, 'Львів', 'Одеса', 623, 10, '2021-09-23 10:00:51'),

(1, 'Суми', 'Івано-Франківськ', 1067, 12, '2021-09-12 11:39:37'),

(22, 'Дніпро', 'Харків', 218, 50, '2021-10-24 19:04:23'),

(19, 'Тернопіль', 'Рівне', 153, 48, '2021-07-10 19:58:31'),

(18, 'Одеса', 'Мадрид', 3932, 43, '2021-08-20 16:54:35'),

(17, 'Миколаїв', 'Хмельницький', 556, 31, '2021-06-23 10:10:45'),

(19, 'Чернігів', 'Кропивницький', 426, 22, '2021-07-08 10:02:28'),

(20, 'Вінниця', 'Чернігів', 411, 40, '2021-01-15 00:13:26'),

(6, 'Кропивницький', 'Амстердам', 1985, 16, '2021-03-12 08:58:18'),

(1, 'Луцьк', 'Любляна', 950, 17, '2021-02-23 19:18:07'),

(3, 'Черкаси', 'Брюссель', 1964, 6, '2021-07-30 09:44:29'),

(11, 'Житомир', 'Івано-Франківськ', 466, 55, '2021-07-21 12:52:50'),

(7, 'Донецьк', 'Андорра-ла-Велья', 2845, 2, '2021-07-03 11:36:39'),

(2, 'Хмельницький', 'Париж', 1788, 5, '2021-01-02 08:07:23'),

(1, 'Луганськ', 'Суми', 267, 11, '2021-02-27 13:29:13'),

(6, 'Суми', 'Гельсінкі', 1214, 15, '2021-03-15 02:15:15'),

(1, 'Донецьк', 'Ужгород', 1333, 1, '2021-04-17 00:03:44'),

(2, 'Харків', 'Бухарест', 976, 8, '2021-03-03 03:42:24'),

(9, 'Івано-Франківськ', 'Суми', 1007, 54, '2021-03-16 08:17:25'),

(21, 'Херсон', 'Київ', 545, 49, '2021-01-27 19:51:46'),

(1, 'Ужгород', 'Кишинів', 673, 14, '2021-03-16 12:28:59'),

(8, 'Херсон', 'Київ', 612, 58, '2021-01-12 18:07:00'),

(2, 'Київ', 'Лісабон', 3355, 14, '2021-09-26 09:34:54'),

(17, 'Тернопіль', 'Харків', 897, 34, '2021-02-22 01:45:39'),

(1, 'Полтава', 'Софія', 1101, 7, '2021-01-14 01:33:36'),

(19, 'Миколаїв', 'Київ', 479, 18, '2021-02-04 15:28:19'),

(16, 'Житомир', 'Івано-Франківськ', 419, 23, '2021-03-09 18:59:38'),

(1, 'Миколаїв', 'Сараєво', 1124, 6, '2021-10-06 18:55:15'),

(22, 'Дніпро', 'Луцьк', 881, 39, '2021-10-21 07:47:48'),

(10, 'Кропивницький', 'Миколаїв', 182, 53, '2021-03-10 02:42:48'),

(3, 'Хмельницький', 'Берн', 1444, 4, '2021-03-19 11:52:21'),

(1, 'Київ', 'Осло', 1630, 3, '2021-01-02 18:36:57');

COMMIT ;

SELECT flight\_ID, f\_from, f\_to, starttime, endtime, status FROM Flights WHERE flight\_ID=6;

# UPDATE Flights SET endtime = '2021-01-08 22:00:00' WHERE flight\_ID = 6;

BEGIN ;

INSERT INTO Flights\_has\_Cities(idFlight, idCity1, idCity2)

(SELECT flight\_ID, city\_ID, (SELECT city\_ID FROM Cities WHERE f\_to = city\_name) FROM Flights, Cities

WHERE f\_from = city\_name);

COMMIT ;

# SELECT \* FROM Flights\_has\_Cities;

delimiter //

CREATE TRIGGER setPriceAirTicket BEFORE INSERT ON Airplane\_tickets

FOR EACH ROW

BEGIN

CALL calc\_price\_at( NEW.at\_price, NEW.Flights\_has\_Cities\_primaryID, NEW.idType\_tickets);

END;//

delimiter ;

# Запит для отримання ключів потрібних авірейсів з проміжної таблиці

/\*SELECT idFirm, flight\_ID, primaryID FROM Flights INNER JOIN Flights\_has\_Cities

ON Flights.flight\_ID = Flights\_has\_Cities.idFlight WHERE idFirm BETWEEN 1 AND 7;\*/

BEGIN ;

INSERT INTO Airplane\_tickets(idType\_tickets, Flights\_has\_Cities\_primaryID)

VALUES (7, 10),

(9, 50),

(9, 13),

(8, 25),

(8, 38),

(7, 2),

(8, 30),

(9, 5),

(8, 11),

(7, 33),

(8, 27),

(8, 17),

(8, 20),

(9, 46),

(9, 41),

(7, 18),

(8, 35),

(8, 48),

(9, 45),

(8, 19),

(7, 15),

(8, 24),

(8, 37),

(9, 7),

(8, 6);

COMMIT ;

# SELECT \* FROM Airplane\_tickets;

delimiter //

CREATE TRIGGER setClassAirHasCus BEFORE INSERT ON Airplane\_tickets\_has\_Customers

FOR EACH ROW

BEGIN

CASE

WHEN (SELECT idType\_tickets FROM Airplane\_tickets WHERE NEW.Airplane\_tickets\_a\_ticket\_ID = a\_ticket\_ID) = 7

THEN SET NEW.at\_class = 'Бізнес';

WHEN (SELECT idType\_tickets FROM Airplane\_tickets WHERE NEW.Airplane\_tickets\_a\_ticket\_ID = a\_ticket\_ID) = 8

THEN SET NEW.at\_class = 'Економ';

WHEN (SELECT idType\_tickets FROM Airplane\_tickets WHERE NEW.Airplane\_tickets\_a\_ticket\_ID = a\_ticket\_ID) = 9

THEN SET NEW.at\_class = 'Перший';

END CASE;

END;//

delimiter ;

COMMIT ;

INSERT INTO Airplane\_tickets\_has\_Customers

(Airplane\_tickets\_a\_ticket\_ID, Customers\_customer\_ID, a\_booking\_time, a\_seat, comments)

VALUES ( 1, 1, '2020-10-09 20:21:36', 1, NULL),

( 2, 2, '2020-01-14 09:55:01', 2, NULL),

( 3, 3, '2020-06-23 14:27:41', 3, NULL),

( 4, 4, '2020-05-26 03:05:58', 4, NULL),

( 5, 5, '2020-04-08 16:28:10', 5, NULL),

( 6, 6, '2020-04-14 02:14:58', 6, NULL),

( 7, 7, '2020-08-27 05:01:43', 7, NULL),

( 8, 8, '2020-09-17 00:14:03', 8, NULL),

( 9, 9, '2020-04-08 06:09:20', 9, NULL),

( 10, 10, '2020-04-07 05:52:12', 10, NULL),

( 11, 11, '2020-06-02 16:57:06', 11, NULL),

( 12, 12, '2020-06-01 14:13:55', 12, NULL),

( 13, 13, '2020-12-20 21:12:27', 13, NULL),

( 14, 14, '2020-04-30 03:00:09', 14, NULL),

( 15, 15, '2020-01-10 06:32:32', 15, NULL),

( 16, 16, '2020-10-19 16:27:48', 16, NULL),

( 17, 17, '2020-12-23 04:59:29', 17, NULL),

( 18, 18, '2020-03-08 17:26:36', 18, NULL),

( 19, 19, '2020-06-22 02:36:05', 19, NULL),

( 20, 20, '2020-02-10 15:28:49', 20, NULL),

( 21, 21, '2020-06-22 08:47:22', 21, NULL),

( 22, 22, '2020-12-17 13:17:31', 22, NULL),

( 23, 23, '2020-04-16 16:54:07', 23, NULL),

( 24, 24, '2020-06-25 04:22:31', 24, NULL),

( 25, 25, '2020-10-28 08:31:29', 25, NULL);

COMMIT ;

# SELECT \* FROM Customers;

# SELECT \* FROM Firms;

# SELECT \* FROM Transport;

# SELECT \* FROM Flights;

# SELECT \* FROM Flights\_has\_Cities;

# SELECT \* FROM Type\_tickets;

# SELECT \* FROM Type\_transportations;

# SELECT \* FROM Airplane\_tickets;

# SELECT \* FROM Airplane\_tickets\_has\_Customers;

# SELECT \* FROM Cities;

/\*SELECT c\_name, f\_from, f\_to, f\_distance, at\_price

FROM Customers INNER JOIN Airplane\_tickets\_has\_Customers AthC

ON Customers.customer\_ID = AthC.Customers\_customer\_ID INNER JOIN Airplane\_tickets A

ON AthC.Airplane\_tickets\_a\_ticket\_ID = A.a\_ticket\_ID INNER JOIN Flights\_has\_Cities FhC

ON A.Flights\_has\_Cities\_primaryID = FhC.primaryID INNER JOIN Flights F

ON FhC.idFlight = F.flight\_ID;\*/

delimiter //

CREATE TRIGGER setPriceShipTicket BEFORE INSERT ON Ship\_tickets

FOR EACH ROW

BEGIN

CALL calc\_price\_st( NEW.st\_price, NEW.Flights\_has\_Cities\_primaryID, NEW.idType\_tickets);

END;//

delimiter ;

# DROP TRIGGER setPriceShipTicket;

BEGIN ;

INSERT INTO Ship\_tickets(idType\_tickets, Flights\_has\_Cities\_primaryID)

VALUES ( 12, 32),

( 13, 32),

( 14, 32),

( 12, 32),

( 13, 32),

( 14, 32),

( 12, 32),

( 13, 32),

( 14, 32),

( 12, 32),

( 13, 32),

( 14, 32),

( 12, 32),

( 13, 32),

( 14, 32),

( 12, 32),

( 13, 32),

( 14, 32),

( 12, 32),

( 13, 32),

( 14, 32),

( 12, 32),

( 13, 32),

( 12, 32),

( 12, 32),

( 13, 32),

( 12, 32),

( 12, 32),

( 13, 32),

( 12, 32),

( 12, 32),

( 13, 32),

( 14, 32),

( 12, 32),

( 13, 32),

( 13, 32),

( 13, 32),

( 13, 32),

( 13, 32);

COMMIT ;

# SELECT \* FROM Ship\_tickets;

delimiter //

CREATE TRIGGER setClassShipHasCus BEFORE INSERT ON Ship\_tickets\_has\_Customers

FOR EACH ROW

BEGIN

CASE

WHEN (SELECT idType\_tickets

FROM Ship\_tickets

WHERE NEW.Ship\_tickets\_s\_ticket\_ID = Ship\_tickets.s\_ticket\_ID) = 12

THEN SET NEW.s\_class = 'Стандарт';

WHEN (SELECT idType\_tickets

FROM Ship\_tickets

WHERE NEW.Ship\_tickets\_s\_ticket\_ID = Ship\_tickets.s\_ticket\_ID) = 13

THEN SET NEW.s\_class = 'Преміум';

WHEN (SELECT idType\_tickets

FROM Ship\_tickets

WHERE NEW.Ship\_tickets\_s\_ticket\_ID = Ship\_tickets.s\_ticket\_ID) = 14

THEN SET NEW.s\_class = 'Економ';

END CASE;

END;//

delimiter ;

BEGIN ;

INSERT INTO Ship\_tickets\_has\_Customers(ship\_tickets\_s\_ticket\_id, customers\_customer\_id, s\_booking\_time, s\_seat, comments)

VALUES ( 1, 26, '2020-03-24 05:43:44', 23, NULL),

( 2, 27, '2020-02-25 13:56:49', 97, NULL),

( 3, 28, '2020-09-11 10:54:59', 45, NULL),

( 4, 29, '2020-11-19 02:10:09', 12, NULL),

( 5, 30, '2020-02-14 01:36:50', 88, NULL);

COMMIT ;

SELECT \* FROM Ship\_tickets\_has\_Customers;

BEGIN ;

INSERT INTO Routes(Flights\_has\_Cities\_primaryID, idCity)

VALUES ( 8, 60),

( 8, 63),

( 8, 50),

( 29, 43),

( 29, 63),

( 39, 63),

( 39, 64),

( 39, 57),

( 39, 61),

( 31, 38),

( 31, 65),

( 31, 48),

( 40, 58),

( 51, 51),

( 51, 64),

( 51, 52),

( 28, 64),

( 28, 52),

( 28, 51),

( 1, 51),

( 1, 66),

( 43, 51),

( 43, 54),

( 43, 52),

( 43, 64),

( 4, 61),

( 3, 66),

( 3, 47),

( 3, 58),

( 3, 44),

# маршрути поїздів

( 22, 57),

( 22, 45),

( 34, 52),

( 34, 54),

( 12, 65),

( 42, 51),

( 42, 54),

( 42, 52),

( 42, 64),

( 36, 47),

( 36, 51),

( 36, 57),

( 36, 61),

( 36, 55),

( 16, 43),

( 14, 51),

( 14, 63),

( 14, 59),

( 44, 54),

( 26, 51),

( 26, 57),

( 26, 61),

( 23, 54),

( 21, 57),

( 21, 61),

( 21, 55),

( 49, 60),

( 49, 63),

( 49, 53),

( 47, 64),

( 47, 55),

( 9, 50),

( 9, 60);

COMMIT ;

# SELECT \* FROM Routes;

#

# SELECT \* FROM Flights\_has\_Cities;

# SELECT \* FROM Flights;

# SELECT \* FROM Firms;

# SELECT \* FROM Cities WHERE country = 'Україна';

delimiter //

CREATE TRIGGER setPriceBusTicket BEFORE INSERT ON Bus\_tickets

FOR EACH ROW

BEGIN

CALL calc\_price\_bt(NEW.bt\_price,NEW.idType\_tickets);

END;//

delimiter ;

# DROP TRIGGER setPriceBusTicket;

BEGIN ;

INSERT INTO Bus\_tickets(bt\_price, idType\_tickets, idRoute)

VALUES ( 318, 10, 1),

( 212, 10, 2),

( 448, 10, 3),

( 430, 11, 4),

( 554, 10, 5),

( 112, 11, 6),

( 577, 10, 7),

( 638, 11, 8),

( 953, 10, 9),

( 3754, 10, 10),

( 496, 10, 11),

( 894, 10, 12),

( 153, 10, 13),

( 140, 11, 14),

( 297, 10, 15),

( 423, 10, 16),

( 310, 10, 17),

( 184, 10, 18),

( 384, 10, 19),

( 268, 10, 20),

( 409, 10, 21),

( 545, 10, 22),

( 66, 10, 23),

( 243, 10, 24),

( 368, 11, 25),

( 318, 10, 1),

( 212, 10, 2),

( 448, 10, 3),

( 430, 11, 4),

( 554, 10, 5),

( 112, 11, 6),

( 577, 10, 7),

( 638, 11, 8),

( 953, 10, 9),

( 3754, 10, 10);

COMMIT ;

# SELECT \* FROM Bus\_tickets;

BEGIN ;

INSERT INTO Bus\_tickets\_has\_Customers(bus\_tickets\_b\_ticket\_id, customers\_customer\_id, b\_booking\_time, b\_seat)

VALUES ( 1, 26, '2020-05-08 02:11:23', 1),

( 2, 27, '2020-10-06 21:24:28', 4),

( 3, 28, '2020-05-14 23:59:25', 7),

( 4, 29, '2020-10-27 05:03:21', 19),

( 5, 30, '2020-09-07 08:19:49', 20),

( 6, 31, '2020-05-11 17:41:41', 14),

( 7, 32, '2020-10-30 03:44:05', 13),

( 8, 33, '2020-10-20 15:12:18', 7),

( 9, 34, '2020-02-11 14:04:45', 2),

( 10, 35, '2020-05-12 04:49:42', 10),

( 11, 36, '2020-01-20 08:54:35', 2),

( 12, 37, '2020-11-11 08:28:05', 10),

( 13, 38, '2020-10-21 01:13:59', 11),

( 14, 39, '2020-12-15 00:16:29', 12),

( 15, 40, '2020-07-09 17:08:32', 6),

( 16, 41, '2020-01-05 12:14:06', 17),

( 17, 42, '2020-08-14 14:08:17', 15),

( 18, 43, '2020-01-17 15:10:58', 4),

( 19, 44, '2020-11-29 22:43:53', 20),

( 20, 45, '2020-09-05 11:02:16', 2),

( 21, 46, '2020-02-12 22:26:31', 9),

( 22, 47, '2020-04-10 06:40:08', 8),

( 23, 48, '2020-01-24 19:43:14', 1),

( 24, 49, '2020-09-15 01:05:39', 5),

( 25, 50, '2020-07-22 09:01:15', 3),

( 26, 26, '2020-03-19 20:30:52', 1),

( 27, 26, '2020-03-08 08:22:24', 4),

( 28, 28, '2020-09-20 11:12:35', 7),

( 29, 29, '2020-07-12 20:37:33', 19),

( 30, 26, '2020-12-26 18:52:20', 1),

( 31, 27, '2020-03-11 18:27:57', 4),

( 32, 28, '2020-06-12 02:27:10', 7),

( 33, 29, '2020-08-12 08:28:02', 19),

( 34, 26, '2020-10-04 00:22:49', 1),

( 35, 29, '2020-01-25 05:00:16', 4);

COMMIT ;

# SELECT \* FROM Bus\_tickets\_has\_Customers;

delimiter //

CREATE TRIGGER setPriceTrainTicket BEFORE INSERT ON Train\_tickets

FOR EACH ROW

BEGIN

CALL calc\_price\_tt(NEW.tt\_price,NEW.idType\_tickets);

END;//

delimiter ;

# DROP TRIGGER setPriceTrainTicket;

BEGIN ;

INSERT INTO Train\_tickets( tt\_price, idType\_tickets, idRoute)

VALUES ( 394, 1, 31),

( 554, 1, 32),

( 221, 2, 33),

( 404, 3, 34),

( 400, 4, 35),

( 545, 6, 36),

( 66, 3, 37),

( 243, 5, 38),

( 368, 2, 39),

( 188, 4, 40),

( 327, 4, 41),

( 720, 4, 42),

( 797, 4, 43),

( 1148, 5, 44),

( 364, 1, 45),

( 570, 2, 46),

( 241, 3, 47),

( 906, 4, 48),

( 66, 2, 49),

( 539, 2, 50),

( 931, 6, 51),

( 1009, 6, 52),

( 185, 1, 53),

( 394, 4, 54),

( 472, 5, 55);

COMMIT ;

# SELECT \* FROM Train\_tickets;

BEGIN ;

INSERT INTO Train\_tickets\_has\_Customers(train\_tickets\_t\_ticket\_id, customers\_customer\_id, t\_booking\_time, t\_numwagon, t\_seat)

VALUES ( 1, 2, '2020-03-15 09:59:44', 1, 21),

( 2, 4, '2020-09-01 05:27:15', 6, 13),

( 3, 6, '2020-11-11 16:03:38', 4, 17),

( 4, 8, '2020-10-07 06:42:01', 9, 9),

( 5, 10, '2020-06-04 02:02:28', 10, 60),

( 6, 12, '2020-05-26 09:08:33', 3, 51),

( 7, 14, '2020-11-14 10:25:57', 2, 44),

( 8, 16, '2020-08-09 21:04:42', 7, 28),

( 9, 18, '2020-01-04 23:02:48', 1, 22),

( 10, 20, '2020-01-01 15:59:45', 3, 29),

( 11, 22, '2020-12-31 06:48:37', 6, 27),

( 12, 24, '2020-04-08 05:18:02', 7, 21),

( 13, 26, '2020-12-05 12:04:21', 2, 32),

( 14, 28, '2020-07-27 13:46:32', 8, 6),

( 15, 30, '2020-11-15 01:57:14', 9, 38),

( 16, 32, '2020-09-12 07:16:43', 10, 39),

( 17, 34, '2020-06-22 10:14:16', 8, 26),

( 18, 36, '2020-05-17 00:05:40', 7, 18),

( 19, 38, '2020-11-02 03:13:54', 4, 56),

( 20, 40, '2020-03-03 04:48:08', 2, 49),

( 21, 42, '2020-07-03 21:13:40', 3, 70),

( 22, 44, '2020-12-10 07:47:03', 5, 23),

( 23, 46, '2020-05-06 04:59:54', 7, 42),

( 24, 48, '2020-12-04 22:03:56', 1, 45),

( 25, 50, '2020-05-01 14:58:58', 2, 21);

COMMIT ;

# ЗАПИТИ

# 1. Список пасажирів що придбали плацкартний білет

SELECT c\_name, concat(f\_from, ' - ', f\_to) AS Рейс, CONCAT(f\_from, ' - ', city\_name) AS Маршрут, f\_distance, tt\_price

FROM Customers INNER JOIN Train\_tickets\_has\_Customers TthC

ON Customers.customer\_ID = TthC.Customers\_customer\_ID INNER JOIN Train\_tickets Tt

ON TthC.Train\_tickets\_t\_ticket\_ID = Tt.t\_ticket\_ID INNER JOIN Routes R

ON Tt.idRoute = R.route\_ID INNER JOIN Flights\_has\_Cities FhC

ON R.Flights\_has\_Cities\_primaryID = FhC.primaryID INNER JOIN Flights F

ON FhC.idFlight = F.flight\_ID INNER JOIN Cities C

ON R.idCity = C.city\_ID INNER JOIN Type\_tickets T on Tt.idType\_tickets = T.typeticket\_ID

WHERE type\_afterpay = 'Плацкарт студентський'

ORDER BY tt\_price;

# 2. Рейси за першу половину червня, липня, серпня

SELECT DISTINCT concat(f\_from, ' - ', f\_to) AS Рейс, starttime

FROM Customers INNER JOIN Bus\_tickets\_has\_Customers BthC

ON Customers.customer\_ID = BthC.Customers\_customer\_ID INNER JOIN Bus\_tickets Bt

ON BthC.Bus\_tickets\_b\_ticket\_ID = Bt.b\_ticket\_ID INNER JOIN Routes R

ON Bt.idRoute = R.route\_ID INNER JOIN Flights\_has\_Cities FhC

ON R.Flights\_has\_Cities\_primaryID = FhC.primaryID INNER JOIN Flights F

ON FhC.idFlight = F.flight\_ID INNER JOIN Cities C

ON R.idCity = C.city\_ID

WHERE DAYOFMONTH(starttime) BETWEEN 0 AND 15 AND MONTH(starttime) IN (6,7,8);

# 3. sold tickets

SELECT type\_afterpay, LPAD(CONCAT(COUNT(typeticket\_ID), ' шт.'),10,' ') AS Sold

FROM Type\_tickets INNER JOIN Airplane\_tickets A

ON Type\_tickets.typeticket\_ID = A.idType\_tickets INNER JOIN Airplane\_tickets\_has\_Customers AthC

ON A.a\_ticket\_ID = AthC.Airplane\_tickets\_a\_ticket\_ID GROUP BY 1

UNION

SELECT type\_afterpay, LPAD(CONCAT(COUNT(typeticket\_ID), ' шт.'),10,' ')

FROM Type\_tickets INNER JOIN Train\_tickets T

ON Type\_tickets.typeticket\_ID = T.idType\_tickets INNER JOIN Train\_tickets\_has\_Customers TthC

ON T.t\_ticket\_ID = TthC.Train\_tickets\_t\_ticket\_ID GROUP BY 1

UNION

SELECT type\_afterpay, LPAD(CONCAT(COUNT(typeticket\_ID), ' шт.'),10,' ')

FROM Type\_tickets INNER JOIN Bus\_tickets B

ON Type\_tickets.typeticket\_ID = B.idType\_tickets INNER JOIN Bus\_tickets\_has\_Customers BthC

ON B.b\_ticket\_ID = BthC.Bus\_tickets\_b\_ticket\_ID GROUP BY 1

UNION

SELECT type\_afterpay, LPAD(CONCAT(COUNT(typeticket\_ID), ' шт.'),10,' ')

FROM Type\_tickets INNER JOIN Ship\_tickets S

ON Type\_tickets.typeticket\_ID = S.idType\_tickets INNER JOIN Ship\_tickets\_has\_Customers SthC

ON S.s\_ticket\_ID = SthC.Ship\_tickets\_s\_ticket\_ID GROUP BY 1;

# 4. foreign flights

SELECT T.name AS Transport, CONCAT(f\_from, ' - ', f\_to) AS Flight, F.name AS Firm, type\_name AS Type\_trip

FROM Flights INNER JOIN Firms F

ON Flights.idFirm = F.firm\_ID INNER JOIN Type\_transportations Tt

ON F.idTransportation = Tt.transportation\_ID INNER JOIN Transport T

ON Flights.idTransport = T.transport\_ID INNER JOIN Flights\_has\_Cities FhC

ON Flights.flight\_ID = FhC.idFlight INNER JOIN Cities C

ON FhC.idCity2 = C.city\_ID

WHERE country <> 'Україна';

# 5. Кількість доступних квитків на автобусні рейси

SELECT CONCAT(f\_from, ' - ', f\_to) AS Flight, CONCAT(T.name, '\t', T.number) AS Bus\_and\_number , type\_name,

CONCAT(T.countplace - (SELECT COUNT(Bus\_tickets\_b\_ticket\_ID)

FROM Bus\_tickets\_has\_Customers INNER JOIN Bus\_tickets Bt

ON Bus\_tickets\_has\_Customers.Bus\_tickets\_b\_ticket\_ID = Bt.b\_ticket\_ID

INNER JOIN Routes R ON Bt.idRoute = R.route\_ID INNER JOIN Flights\_has\_Cities FhC

ON R.Flights\_has\_Cities\_primaryID = FhC.primaryID

WHERE flight\_ID = idFlight), '/', T.countplace) AS Tickets

FROM Flights INNER JOIN Firms F

ON Flights.idFirm = F.firm\_ID INNER JOIN Type\_transportations Tt

ON F.idTransportation = Tt.transportation\_ID INNER JOIN Transport T

ON Flights.idTransport = T.transport\_ID

WHERE type\_name = 'Наземне';

# 6. most popular bus travel city

SELECT city\_name, COUNT(idCity2) AS count

FROM Routes INNER JOIN Flights\_has\_Cities FhC

ON Routes.Flights\_has\_Cities\_primaryID = FhC.primaryID INNER JOIN Cities C

ON FhC.idCity2 = C.city\_ID INNER JOIN Flights

ON FhC.idFlight = flight\_ID INNER JOIN Firms F

ON Flights.idFirm = firm\_ID INNER JOIN Type\_transportations Tt

ON F.idTransportation = Tt.transportation\_ID INNER JOIN Transport T

ON Flights.idTransport = T.transport\_ID

WHERE type\_name = 'Колійне'

GROUP BY 1

ORDER BY 2 DESC

LIMIT 1;

# 7. flights with avg price more than 5000 and country not in (ua, fr, bg)

SELECT CONCAT(f\_from, ' - ', f\_to) AS Flight, AVG(at\_price) AS avg\_ticket\_price

FROM Flights INNER JOIN Flights\_has\_Cities FhC

ON Flights.flight\_ID = FhC.idFlight INNER JOIN Airplane\_tickets A

ON FhC.primaryID = A.Flights\_has\_Cities\_primaryID INNER JOIN Cities C

ON FhC.idCity2 = C.city\_ID

WHERE country NOT IN ('Україна', 'Франція', 'Болгарія')

GROUP BY 1

HAVING 5000 < avg\_ticket\_price

ORDER BY 2;

# 8. Пасажири, що купили декілька білетів

SELECT c\_name, COUNT(b\_ticket\_ID) AS tickets, SUM(bt\_price) AS total\_sum

FROM Customers LEFT JOIN Bus\_tickets\_has\_Customers TthC

ON Customers.customer\_ID = TthC.Customers\_customer\_ID RIGHT JOIN Bus\_tickets Tt

ON Tt.b\_ticket\_ID = TthC.Bus\_tickets\_b\_ticket\_ID

GROUP BY 1

HAVING tickets <> 1

ORDER BY 3 DESC;

# 9. Найтриваліший рейс і найкоротший рейс

(SELECT 'Найтриваліший рейс' AS Descrip, CONCAT(f\_from, ' - ', f\_to) AS Flight,

SEC\_TO\_TIME(TIMESTAMPDIFF(SECOND, starttime, endtime)) AS time\_interval

FROM Flights

ORDER BY 3 DESC

LIMIT 1)

UNION

(SELECT 'Найкоротший рейс', CONCAT(f\_from, ' - ', f\_to) AS Flight,

SEC\_TO\_TIME(TIMESTAMPDIFF(SECOND, starttime, endtime))

FROM Flights

ORDER BY 3

LIMIT 1);

# 10. Автобусні рейси до Харкова і Києва дешевше середнього

SELECT b\_ticket\_ID, CONCAT(f\_from, ' - ', city\_name) AS Flight, CONCAT(bt\_price, ' грн.') AS price

FROM Flights INNER JOIN Flights\_has\_Cities FhC

ON Flights.flight\_ID = FhC.idFlight INNER JOIN Routes R

ON FhC.primaryID = R.Flights\_has\_Cities\_primaryID INNER JOIN Bus\_tickets Bt

ON R.route\_ID = Bt.idRoute INNER JOIN Bus\_tickets\_has\_Customers BthC

ON Bt.b\_ticket\_ID = BthC.Bus\_tickets\_b\_ticket\_ID INNER JOIN Cities C

ON C.city\_ID = FhC.idCity2

WHERE bt\_price < (SELECT AVG(bt\_price)

FROM Flights INNER JOIN Flights\_has\_Cities FhC

ON Flights.flight\_ID = FhC.idFlight INNER JOIN Routes R

ON FhC.primaryID = R.Flights\_has\_Cities\_primaryID INNER JOIN Bus\_tickets Bt

ON R.route\_ID = Bt.idRoute INNER JOIN Bus\_tickets\_has\_Customers BthC

ON Bt.b\_ticket\_ID = BthC.Bus\_tickets\_b\_ticket\_ID INNER JOIN Cities C

ON FhC.idCity2 = C.city\_ID)

HAVING Flight LIKE '%Київ' OR Flight LIKE '%Харків'

ORDER BY bt\_price DESC;

# ПРЕДСТАВЛЕННЯ

# 1.

CREATE VIEW sold\_train\_tickets AS SELECT c\_name, concat(f\_from, ' - ', f\_to) AS Рейс,

CONCAT(f\_from, ' - ', city\_name) AS Маршрут, f\_distance, tt\_price

FROM Customers INNER JOIN Train\_tickets\_has\_Customers TthC

ON Customers.customer\_ID = TthC.Customers\_customer\_ID INNER JOIN Train\_tickets Tt

ON TthC.Train\_tickets\_t\_ticket\_ID = Tt.t\_ticket\_ID INNER JOIN Routes R

ON Tt.idRoute = R.route\_ID INNER JOIN Flights\_has\_Cities FhC

ON R.Flights\_has\_Cities\_primaryID = FhC.primaryID INNER JOIN Flights F

ON FhC.idFlight = F.flight\_ID INNER JOIN Cities C

ON R.idCity = C.city\_ID INNER JOIN Type\_tickets T on Tt.idType\_tickets = T.typeticket\_ID

ORDER BY tt\_price;

# 2.

CREATE VIEW summer\_flights AS SELECT DISTINCT concat(f\_from, ' - ', f\_to) AS Рейс, starttime

FROM Customers INNER JOIN Bus\_tickets\_has\_Customers BthC

ON Customers.customer\_ID = BthC.Customers\_customer\_ID INNER JOIN Bus\_tickets Bt

ON BthC.Bus\_tickets\_b\_ticket\_ID = Bt.b\_ticket\_ID INNER JOIN Routes R

ON Bt.idRoute = R.route\_ID INNER JOIN Flights\_has\_Cities FhC

ON R.Flights\_has\_Cities\_primaryID = FhC.primaryID INNER JOIN Flights F

ON FhC.idFlight = F.flight\_ID INNER JOIN Cities C

ON R.idCity = C.city\_ID

WHERE MONTH(starttime) IN (6,7,8);

# 3.

CREATE VIEW unused\_transport AS SELECT \*

FROM Transport

WHERE transport\_ID NOT IN (

SELECT DISTINCT transport\_ID

FROM Transport INNER JOIN Flights F

ON Transport.transport\_ID = F.idTransport);

# 4.

CREATE VIEW foreign\_flights AS SELECT T.name AS Transport, CONCAT(f\_from, ' - ', f\_to) AS Flight, F.name AS Firm, type\_name AS Type\_trip

FROM Flights INNER JOIN Firms F

ON Flights.idFirm = F.firm\_ID INNER JOIN Type\_transportations Tt

ON F.idTransportation = Tt.transportation\_ID INNER JOIN Transport T

ON Flights.idTransport = T.transport\_ID INNER JOIN Flights\_has\_Cities FhC

ON Flights.flight\_ID = FhC.idFlight INNER JOIN Cities C

ON FhC.idCity2 = C.city\_ID

WHERE country <> 'Україна';

# 5.

CREATE VIEW the\_most\_popular\_travel\_city AS SELECT city\_name, COUNT(idCity2)

FROM Routes INNER JOIN Flights\_has\_Cities FhC

ON Routes.Flights\_has\_Cities\_primaryID = FhC.primaryID INNER JOIN Cities C

ON FhC.idCity2 = C.city\_ID INNER JOIN Flights

ON FhC.idFlight = flight\_ID INNER JOIN Firms F

ON Flights.idFirm = firm\_ID INNER JOIN Type\_transportations Tt

ON F.idTransportation = Tt.transportation\_ID INNER JOIN Transport T

ON Flights.idTransport = T.transport\_ID

WHERE type\_name IN ('Наземне', 'Колійне')

GROUP BY 1

ORDER BY 2 DESC

LIMIT 5;

# КОРИСТУВАЧІ

CREATE USER 'Developer'@'localhost:3306' IDENTIFIED BY 'helloWorld';

GRANT ALL PRIVILEGES ON Tickets\_booking.\* TO 'Developer'@'localhost:3306';

FLUSH PRIVILEGES;

# SHOW GRANTS FOR 'Developer'@'localhost:3306';

CREATE USER 'Admin'@'localhost:3306' IDENTIFIED BY 'myAdmin';

GRANT SELECT, INSERT, DELETE, UPDATE, CREATE VIEW, EXECUTE ON Tickets\_booking.\* TO 'Admin'@'localhost:3306';

FLUSH PRIVILEGES;

# SHOW GRANTS FOR 'Admin'@'localhost:3306';

CREATE USER 'Customer'@'localhost:3306' IDENTIFIED BY 'qwerty123';

GRANT SELECT ON Tickets\_booking.\* TO 'Customer'@'localhost:3306';

FLUSH PRIVILEGES;

# SHOW GRANTS FOR 'Customer'@'localhost:3306';

# SELECT User, Host FROM mysql.user WHERE Host LIKE '%3306';

# SHOW TABLES;

/\*SELECT CONCAT(GROUP\_CONCAT(' ',COLUMN\_NAME),';')

FROM INFORMATION\_SCHEMA.COLUMNS

WHERE TABLE\_NAME = 'type\_transportations';\*/