Тема 25

Автогара

Проект по Системи за управление на бази от данни-практикум

Изготвила: Берна Метин Сали

Факултетен номер: 72008

Специалност: Информационни системи

Курс : 3

Група:1

1. Обхват на модела. Дефиниране на задачата.

Тази база от данни ще съдържа информация за дадена автогара в съответния град. Разработената база от данни ще обслужва вътрешна система на автогарата, няма да обслужва сайт на автогарата. В територията на един град може да има повече от една автогара. В автогара има каси на фирмите, които отговарят за различните автобусни линии. От автогарата тръгват автобуси, както по направление вътрешни линии, така и по направление международни линии. От касата на дадена фирма, по направление вътрешни линии, може да се получи информация за броя на свободните места, часа на тръгване, населеното място до където ще пътува автобуса.

* по направление международни линии може да се получи информация за броя на свободните места, часа на тръгване, населеното място до където ще пътува автобуса, необходимите документи, при преминаване през границите.

Всяка каса има номер. Касата може да бъде определена еднозначно с името на фирмата. Определена фирма може да осъществява автобусни превози до много населени места. Фирмата се определя еднозначно с името си. Всеки автобус има определен час на тръгване, за да не се съвпада с друг автобус за същото населено място. И автобусите тръгват



само от определен сектор. Според разпреление от един сектор може да тръгват много автобуси за различни населени места в различно време.

Билети могат да се купуват онлайн или на място от касата на дадена фирма. Не може да се направи резервация на места по телефона. При наличие на клиентска карта се прави отстъпка. Клиентската карта се определя еднозначно чрез номера си, която се ползва при закупуване на билета. След закупуването на билета за пътниците се съхранява информация като имена на пътника, номера на клиентската карта(ако има такава), часа на пътуване, датата на пътуване, мястото в автобуса.

Автогарата има едно информационно гише. Където може да се получи информация за всичките превози тръгващи от автогарата. Съхранява се информация за часа на тръгване, за фирмата, чийто автобус ще пътува, маршрута, по който ще пътува, населеното място до където ще пътува, сектора от където ще тръгва. Съхранява се телефоните номера на другите автогари за връзка.

* + автогарата работят служители. Всеки служител отговаря за една каса. За тях се пази информация като имена на служителя, телефонен номер, информаци я за фирмата, за която работи. Всяка каса има определено работно време, телефон.

1. Множества от същности и техните атрибути
   * + Фирми за автобусни превози- име, телефонен номер
     + Пътници – име, място в автобуса, номер на клиентска карта, час на

тръгване, дата

* Служители – име, телефонен номер,служебен номер, информация за фирмата, за която работи
* Каси – номер на касата, име на транспортната фирма
* Автобуси - име на фирмата, сектор, регистрационен номер
* Сектори - номер

1. Домейн на атрибутите
   * Фирми за автобусни превози– име: низ, телефонен номер:низ
   * Пътници – име:низ, място в автобуса: положително цяло число,

номер на клиетксата карта: положително цяло число или NULL, час

на тръгване:низ, дата:низ

* + - * Служители- име:низ, телефонен номер:низ, информация за

фирмата, за която работи:низ

* + - * Каси- номер: положително цяло число или NULL, името на

транспортната фирма: низ

* + - * Автобуси - име на фирмата: низ, сектор: цяло положително число,

регистрационен номер: низ

* + - * Сектор – номер:цяло положително число

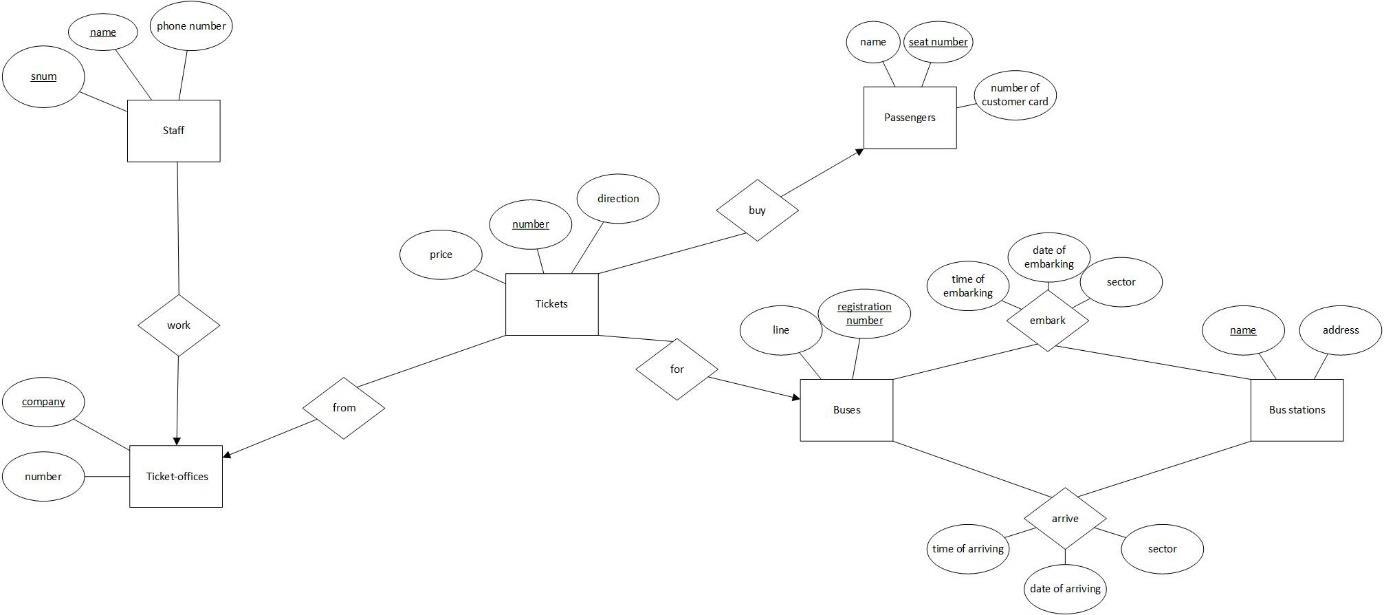
1. Връзки
   * + В една каса работи един служител.
     + Служителя може да продава много билети, но всеки пътник може да купува един билет за определен час и дата, до определеното населено място.
     + Автобус, за опрелено населено място може да тръгва само от един сектор. А от същия сектор може да тръгват много автобуси за други населени места.
     + Пътника пътува с един автобус, но със същия автобус пътуват много пътници
   1. Ограничения по единствена стойност, референтна цялостност и друг тип ограничения
      * + Фирми за автобусни превози- име:еднозначно определя фирмата
        + Пътници – място в автобуса: еднозначно определя пътника
        + Служители – служебен номер: еднозначно определя служителя
        + Каси – името на фирмата за автобусни превози: еднозначно определя касата
        + Автобуси - регистрационен номер: еднозначно определя автобуса
        + Сектори – номер: еднозначно определя сектора



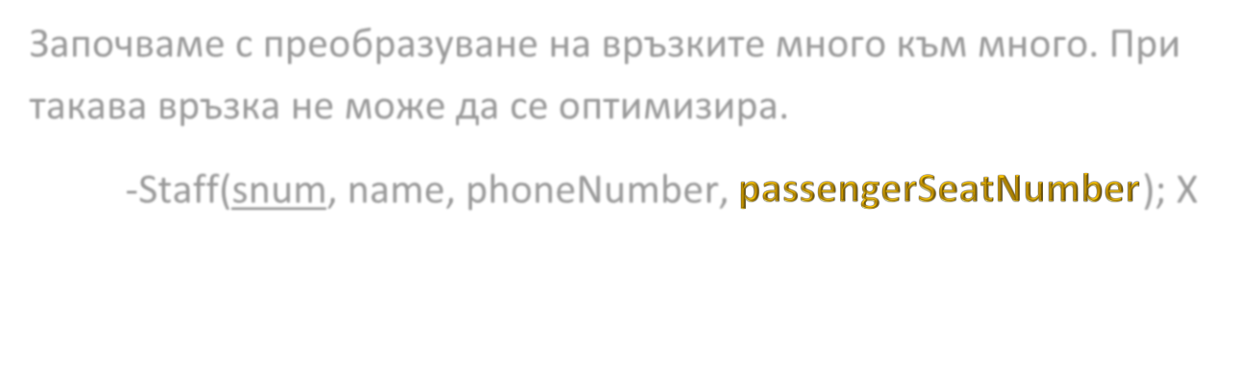
1. Източници

https://www.centralnaavtogara.bg/#b

1. E/R модел на данни

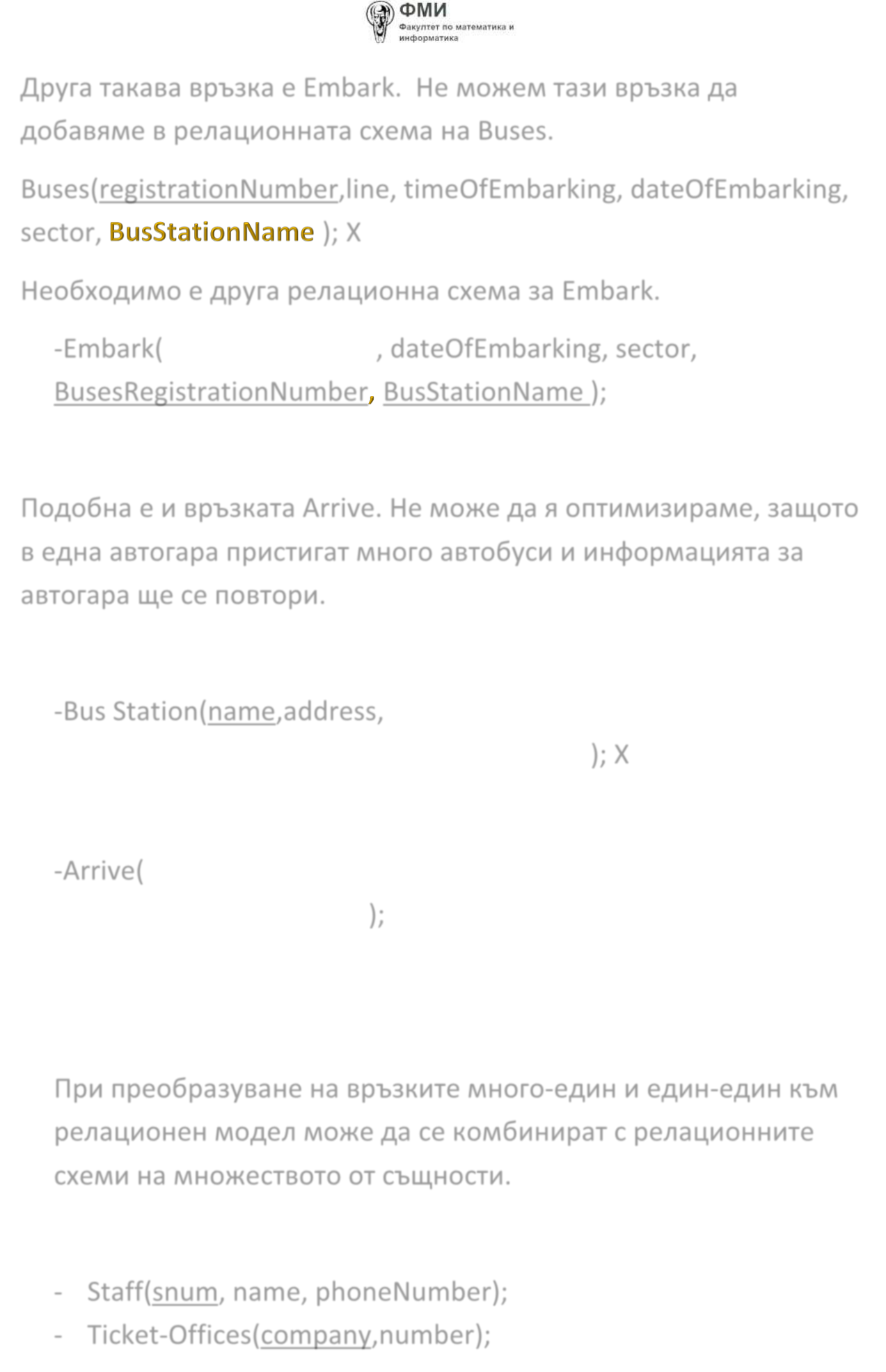


1. Релационен модел на данни



Започваме с преобразуване на връзките много към много. При такава връзка не може да се оптимизира.

-Stаff(snum, name, phoneNumber, ); Х



Друга такава връзка е Embark. Не можем тази връзка да добавяме в релационната схема на Buses.

Buses(registrationNumber,line, timeOfEmbarking, dateOfEmbarking,

sector, ); X

Необходимо е друга релационна схема за Embark.

-Embark(timeOfEmbarking, dateOfEmbarking, sector,

BusesRegistrationNumber BusStationName );

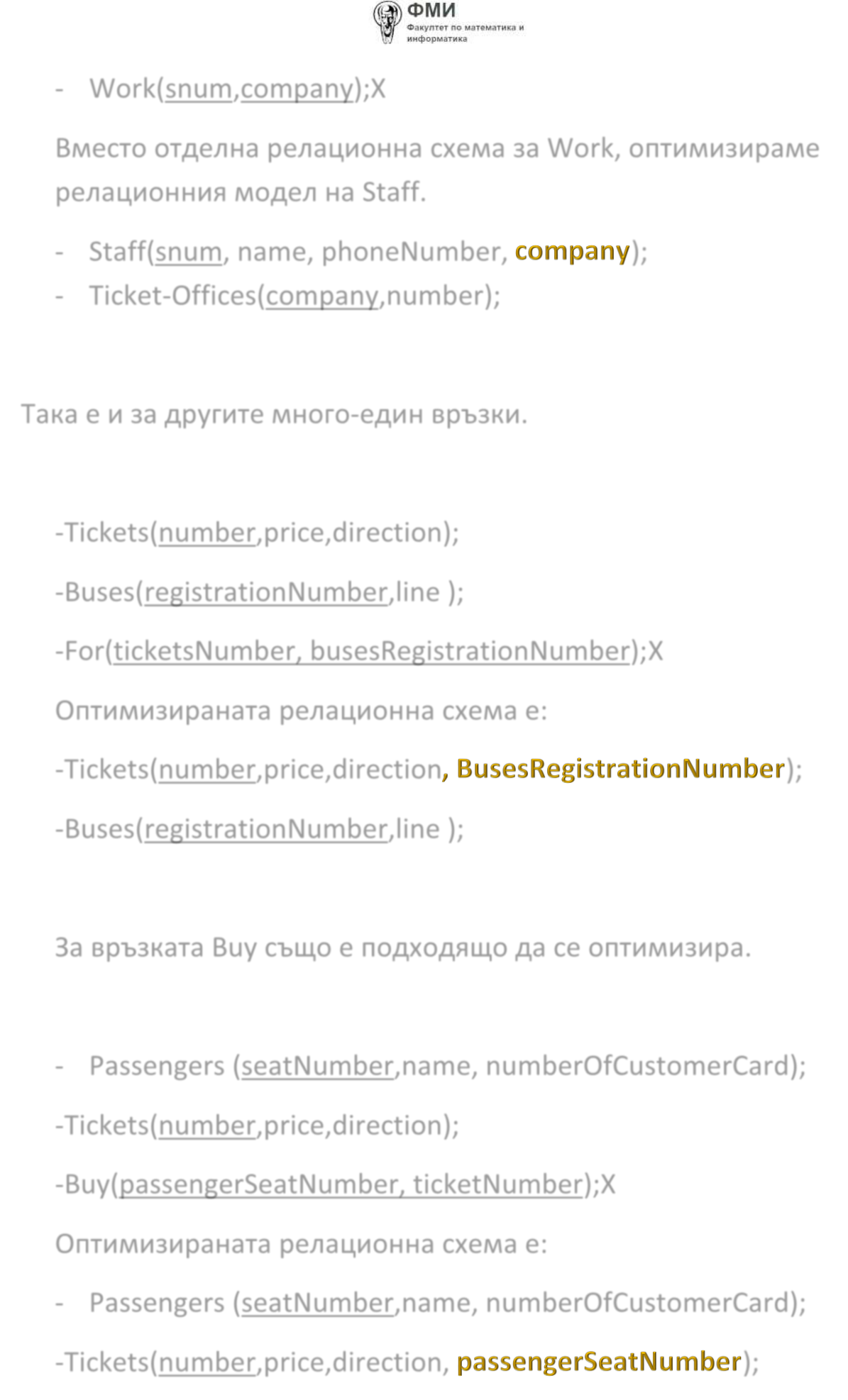
Подобна е и връзката Arrive. Не може да я оптимизираме, защото

* една автогара пристигат много автобуси и информацията за автогара ще се повтори.

-Bus Station(name,address, timeOfArrive, dateOfArrive, sector, BusStationName , BusesRegistrationNumber); X

-Arrive(timeOfArrive, dateOfArrive, sector, BusStationName , BusesRegistrationNumber);

При преобразуване на връзките много-един и един-един към релационен модел може да се комбинират с релационните схеми на множеството от същности.

* Stаff(snum, name, phoneNumber);
* Ticket-Offices(company,number);
* Work(snum,company);X

Вместо отделна релационна схема за Work, оптимизираме релационния модел на Staff.

- Stаff(snum, name, phoneNumber, ); - Ticket-Offices(company,number);

Така е и за другите много-един връзки.

-Tickets(number,price,direction);

-Buses(registrationNumber,line );

-For(ticketsNumber, busesRegistrationNumber);X

Оптимизираната релационна схема е:

-Tickets(number,price,direction ); -Buses(registrationNumber,line );

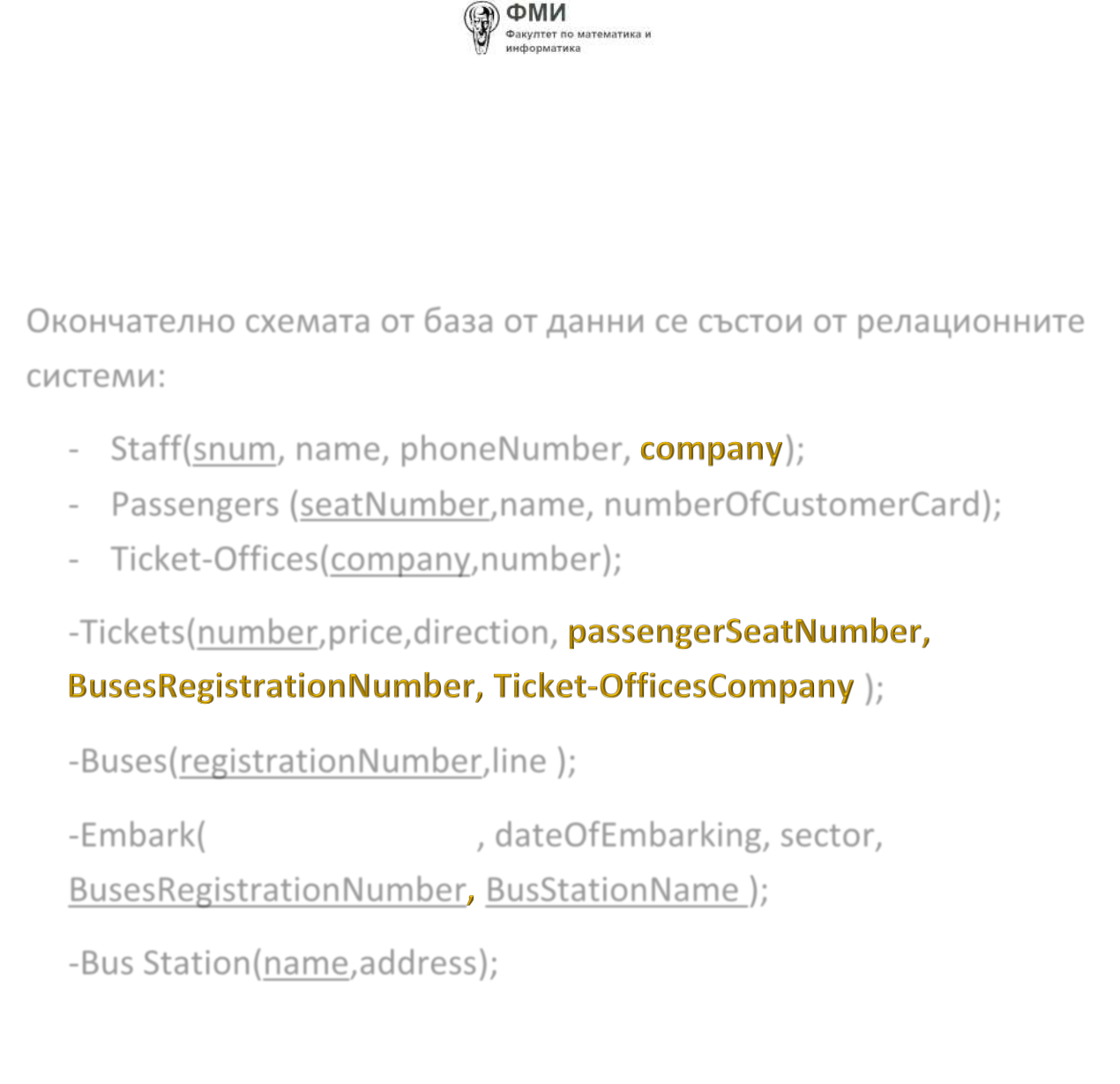
За връзката Buy също е подходящо да се оптимизира.

* Passengers (seatNumber,name, numberOfCustomerCard); -Tickets(number,price,direction); -Buy(passengerSeatNumber, ticketNumber);X

Оптимизираната релационна схема е:

- Passengers (seatNumber,name, numberOfCustomerCard);

-Tickets(number,price,direction, );



1. Релационен модел на данни (релационни схеми)

Окончателно схемата от база от данни се състои от релационните системи:

- Stаff(snum, name, phoneNumber, );

* Passengers (seatNumber,name, numberOfCustomerCard);
* Ticket-Offices(company,number);

-Tickets(number,price,direction,

);

-Buses(registrationNumber,line );

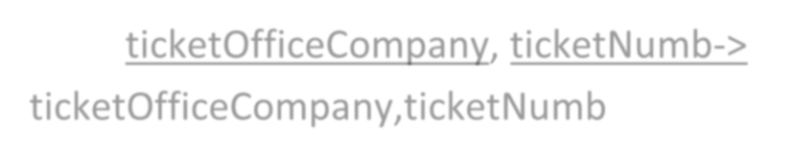
-Embark(timeOfEmbarking, dateOfEmbarking, sector, BusesRegistrationNumber BusStationName );

-Bus Station(name,address);

10.ФЗ и Нормализация

За схемата на БД по-горе са в сила следните ФЗ:

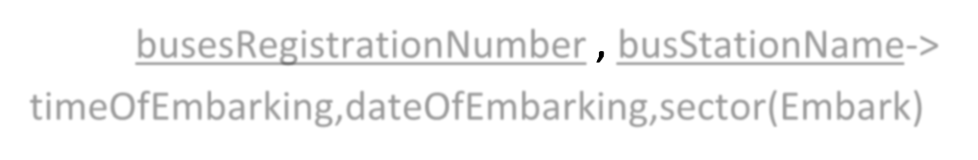
* ФЗ-1: snum,name -> phoneNumber,company(Staff)
* ФЗ-2:seatNumber->name,numberOfCustomerCard(Passenger)
* ФЗ-3:company->number(Ticket-office)
* ФЗ-4:ticketOfficeCompany, ticketNumb-> ticketOfficeCompany,ticketNumb
* ФЗ-5: number-> price,direction,



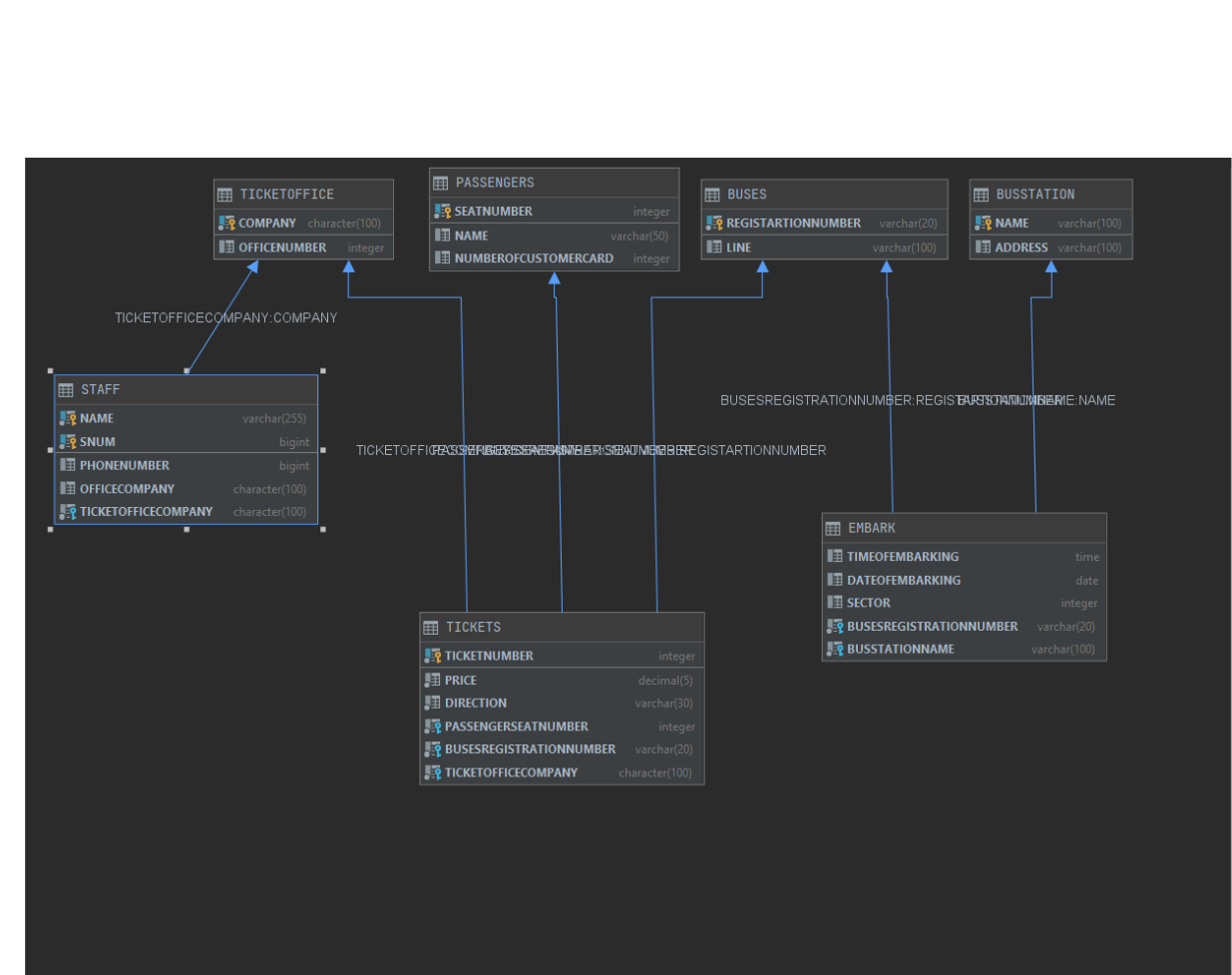


passengerSeatNumber,BusesRegistrationNumber,  (Tickets)

* ФЗ-6: registrationNumber-> line(Buses)
* ФЗ-7: busesRegistrationNumber busStationName-> timeOfEmbarking,dateOfEmbarking,sector(Embark)
* ФЗ-8: name->address(Bus Station)



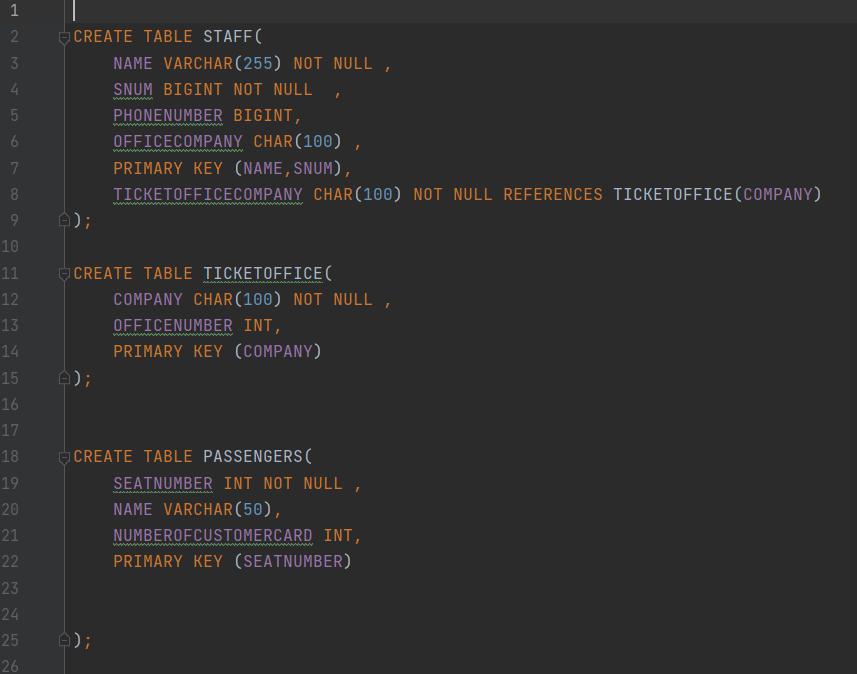
За всички ФЗ на релациите е в сила, че в лявата част се намира суперключ за релацията, следователно всички релации са в НФБК.

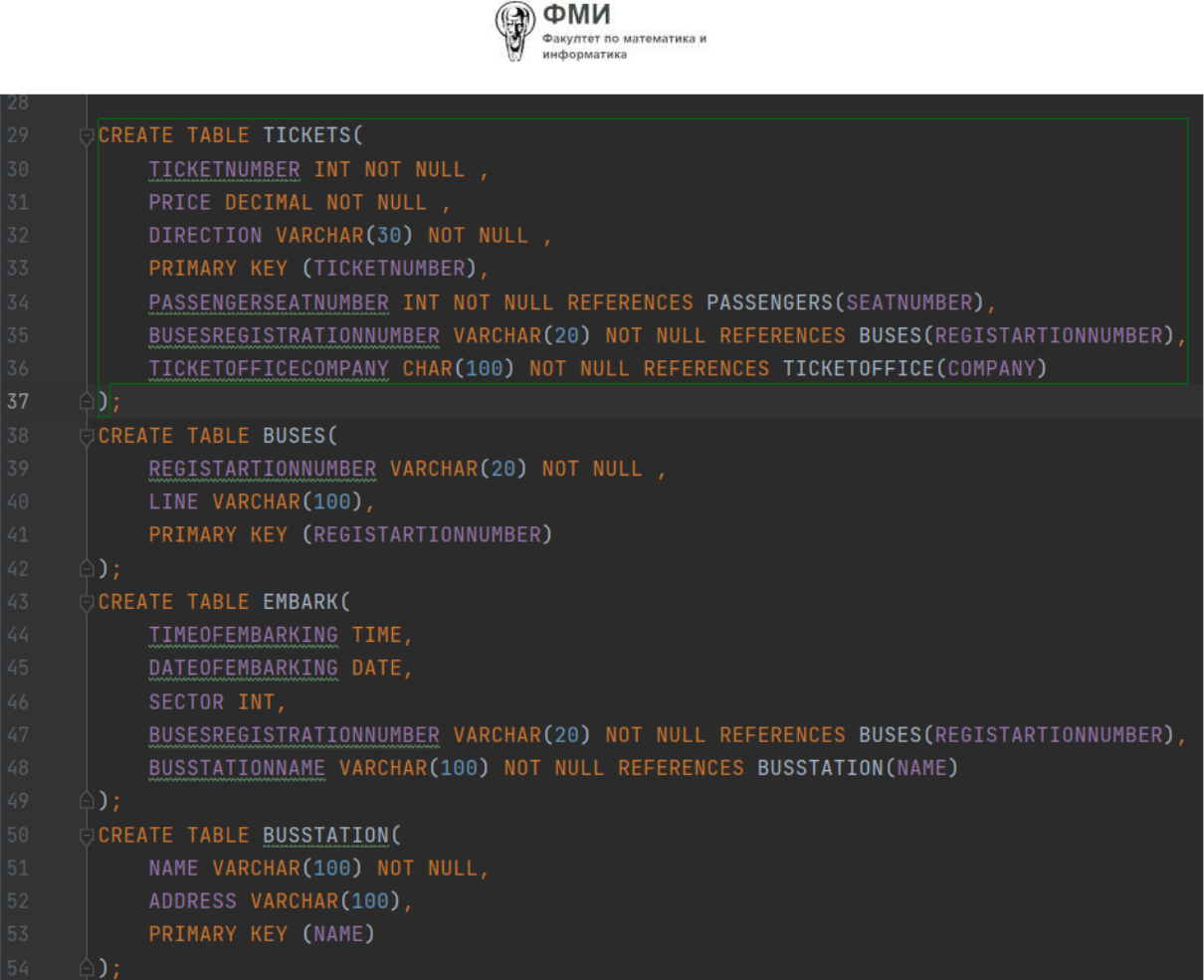


11.Схема на базата от данни



1. Създаване на таблиците в Datagrip.





ИЗГЛЕДИ

CREATE VIEW BUS\_INFO  
AS  
SELECT E.SECTOR,E.BUSSTATIONNAME,B.LINE  
FROM FN72008.EMBARK E, FN72008.BUSES B  
WHERE B.REGISTARTIONNUMBER=E.BUSESREGISTRATIONNUMBER;

SELECT \* FROM FN72008.BUS\_INFO@

CREATE OR REPLACE VIEW K\_BUSSTATION  
AS  
SELECT TIMEOFEMBARKING,SECTOR, BUSSTATIONNAME  
FROM FN72008.EMBARK  
WHERE BUSSTATIONNAME ='АВТОГАРА 21-ВИ ВЕК';

SELECT \* FROM FN72008.K\_BUSSTATION@

Функцията STAFF\_INFO по номер на служител, връща компанията за която работи

CREATE OR REPLACE FUNCTION STAFF\_INFO(S\_SNUM INT)

RETURNS VARCHAR(50)

RETURN

SELECT TICKETOFFICECOMPANY AS STAFF\_COMPANY

FROM FN72008.STAFF

WHERE SNUM=S\_SNUM@

VALUES FN72008.STAFF\_INFO(36167707)@

Функцията TICKET\_INFO по номер на билет връща таблица с цената на билета, посоката, мястото в автобуса и автобусната компания, с която ще пътува

CREATE OR REPLACE FUNCTION TICKET\_INFO(P\_TICKETNO INT)

RETURNS TABLE (PRICE DECIMAL(9,2),DIRECTION VARCHAR(30),PASSENGERSEATNUMBER INT,TICKETOFFICECOMPANY CHAR (100))

SPECIFIC TICKET\_INFO

RETURN

SELECT PRICE,DIRECTION,PASSENGERSEATNUMBER,TICKETOFFICECOMPANY

FROM FN72008.TICKETS

WHERE TICKETNUMBER=P\_TICKETNO@

SELECT \* FROM TABLE (FN72008.TICKET\_INFO(28976)) @

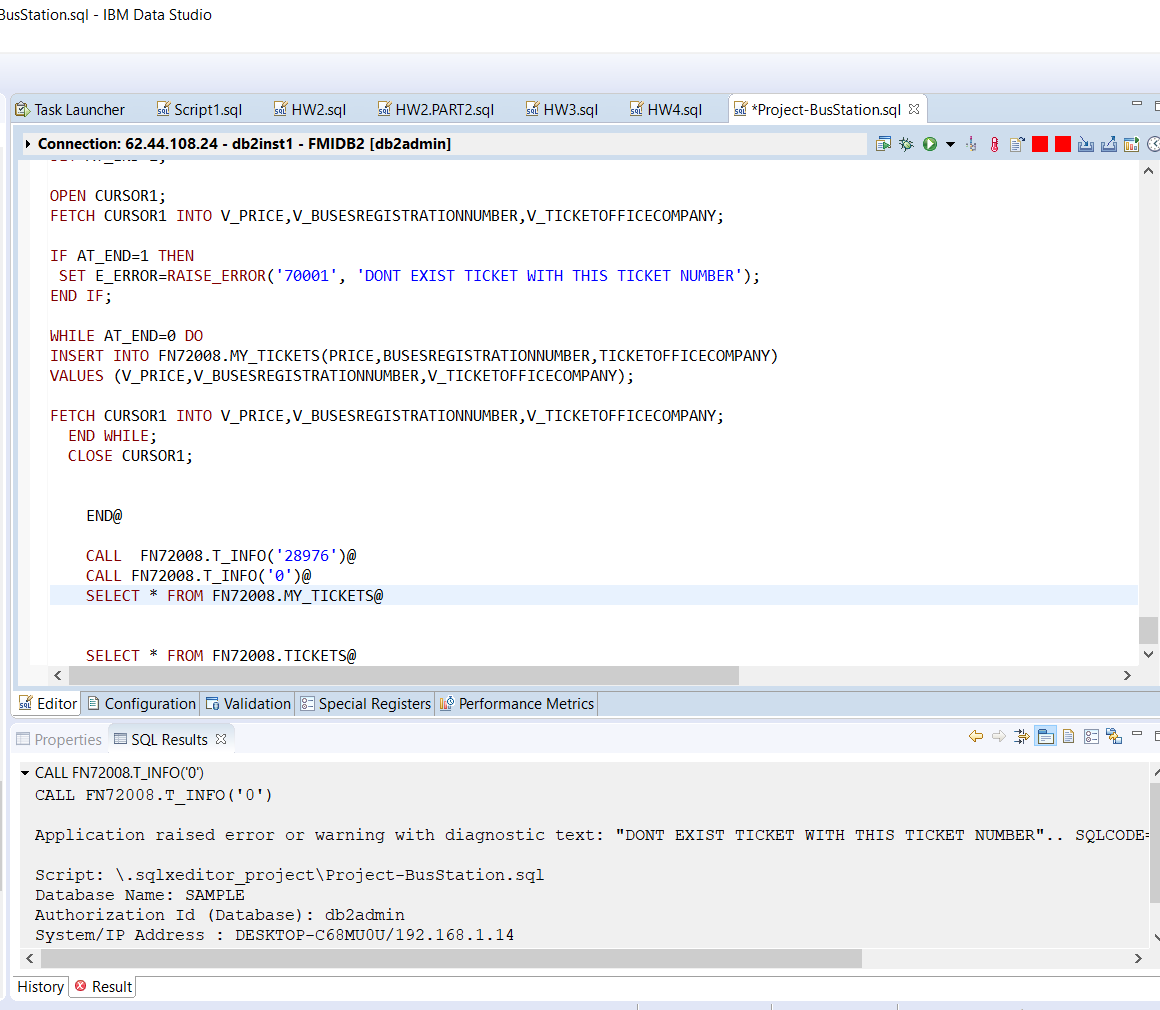
SELECT \* FROM TABLE (FN72008.TICKET\_INFO(28977)) @

SELECT \* FROM TABLE (FN72008.TICKET\_INFO(28974)) @

* процедура с курсор и while цикъл;

При въведено правилен номер на билет връща цената, регистрационния номер на автобуса и компания, от която е купен билета. И тези данни се добавят в таблицата FN72008.MY\_TICKETS

При неправилно въведен номер на билет изкарва грешка 'DONT EXIST TICKET WITH THIS TICKET NUMBER'



CREATE TABLE FN72008.MY\_TICKETS(

PRICE DECIMAL(9,2) ,

BUSESREGISTRATIONNUMBER VARCHAR(20),

TICKETOFFICECOMPANY CHAR(100)

)@

CREATE OR REPLACE PROCEDURE FN72008.T\_INFO(IN T\_TICKETNO INT)

LANGUAGE SQL

BEGIN

DECLARE E\_ERROR INT;

DECLARE V\_PRICE DECIMAL(9,2);

DECLARE V\_BUSESREGISTRATIONNUMBER VARCHAR (20);

DECLARE V\_TICKETOFFICECOMPANY CHAR(100);

DECLARE AT\_END INTEGER DEFAULT 0;

DECLARE CURSOR1 CURSOR FOR SELECT PRICE,BUSESREGISTRATIONNUMBER,TICKETOFFICECOMPANY FROM FN72008.TICKETS WHERE T\_TICKETNO=TICKETNUMBER;

DECLARE CONTINUE HANDLER FOR SQLSTATE '02000'

SET AT\_END=1;

OPEN CURSOR1;

FETCH CURSOR1 INTO V\_PRICE,V\_BUSESREGISTRATIONNUMBER,V\_TICKETOFFICECOMPANY;

IF AT\_END=1 THEN

SET E\_ERROR=RAISE\_ERROR('70001', 'DONT EXIST TICKET WITH THIS TICKET NUMBER');

END IF;

WHILE AT\_END=0 DO

INSERT INTO FN72008.MY\_TICKETS(PRICE,BUSESREGISTRATIONNUMBER,TICKETOFFICECOMPANY)

VALUES (V\_PRICE,V\_BUSESREGISTRATIONNUMBER,V\_TICKETOFFICECOMPANY);

FETCH CURSOR1 INTO V\_PRICE,V\_BUSESREGISTRATIONNUMBER,V\_TICKETOFFICECOMPANY;

END WHILE;

CLOSE CURSOR1;

END@

CALL FN72008.T\_INFO('28976')@

CALL FN72008.T\_INFO('0')@

SELECT \* FROM FN72008.MY\_TICKETS@

* процедура с прихващане на изключение;

CREATE OR REPLACE PROCEDURE FN72008.P\_INFO(IN P\_SEATNO INT, OUT P\_SQLSTATE CHAR(5),OUT P\_SQLCODE INTEGER)

LANGUAGE SQL

BEGIN

DECLARE E\_ERROR INT;

DECLARE V\_NAME VARCHAR(50);

DECLARE V\_NUM\_CARD INT;

DECLARE AT\_END INTEGER DEFAULT 0;

DECLARE SQLCODE INTEGER DEFAULT 0;

DECLARE SQLSTATE CHAR (5) DEFAULT '00000';

DECLARE CURSOR1 CURSOR FOR SELECT NAME,NUMBEROFCUSTOMERCARD FROM FN72008.PASSENGERS WHERE SEATNUMBER=P\_SEATNO;

DECLARE CONTINUE HANDLER FOR SQLSTATE '02000'

SET AT\_END=1;

DECLARE EXIT HANDLER FOR SQLEXCEPTION SELECT SQLCODE,SQLSTATE INTO P\_SQLCODE,P\_SQLSTATE

FROM SYSIBM.SYSDUMMY1;

OPEN CURSOR1;

FETCH CURSOR1 INTO V\_NAME,V\_NUM\_CARD;

IF AT\_END=1 THEN

SET E\_ERROR=RAISE\_ERROR('70001', 'NO PASSENGER WITH THIS SEATNO');

END IF;

WHILE AT\_END=0 DO

INSERT INTO FN72008.P\_PASSENGER(NAME ,NUMBEROFCUSTOMERCARD )

VALUES (V\_NAME,V\_NUM\_CARD);

FETCH CURSOR1 INTO V\_NAME,V\_NUM\_CARD;

END WHILE;

CLOSE CURSOR1;

END@

SELECT \* FROM FN72008.P\_PASSENGER@

Процедурата по място в автобуса връща името на пътника

* процедура с курсор и входни и изходни параметри;

CREATE OR REPLACE PROCEDURE FN72008.PASSENGER\_INFO(IN P\_SEATNO INT, OUT P\_NAME VARCHAR(50))

LANGUAGE SQL

BEGIN

DECLARE C1 CURSOR FOR SELECT NAME FROM FN72008.PASSENGERS WHERE SEATNUMBER=P\_SEATNO;

OPEN C1;

FETCH C1 INTO P\_NAME;

END@

SELECT \* FROM FN72008.PASSENGERS

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

Тригера CHANGE\_PRICE при всяка промяна на посоката от 'ONE-WAY' на 'TWO-WAY' автоматично увеличава цената двойно

SELECT \* FROM FN72008.TICKETS@

CREATE TRIGGER CHANGE\_PRICE

AFTER UPDATE OF DIRECTION ON FN72008.TICKETS

REFERENCING OLD AS O NEW AS N

FOR EACH ROW

WHEN(O. DIRECTION = 'ONE-WAY')

UPDATE FN72008.TICKETS

SET PRICE=PRICE\*2@

UPDATE FN72008.TICKETS

SET DIRECTION = 'TWO-WAY'

WHERE TICKETNUMBER='28976'@

--Тригера задейства преди вмъкване на нов персонал. Тригера ще провери дали персонала има телефонен номер,

--и ако няма стойност тригера ще даде стойност(телефонен номер на автогарата от където може да се получи също информация)

CREATE TRIGGER ADD\_PHONE\_NUMBER

NO CASCADE BEFORE INSERT ON FN72008.STAFF

REFERENCING NEW AS N

FOR EACH ROW

MODE DB2SQL

WHEN (N.PHONENUMBER IS NULL)

SET N.PHONENUMBER=088888888@

INSERT INTO FN72008.STAFF(NAME, SNUM, PHONENUMBER, OFFICECOMPANY, TICKETOFFICECOMPANY) VALUES ('АНИ БОЯДЖИЕВА','036676709',NULL,NULL, 'КРИСТАЛ')@

SELECT \* FROM STAFF@

