Софийски университет
„Св. Климент Охридски“,
гр. София
Факултет по математика и информатика

ДОКУМЕНТАЦИЯ

*към проект по „**СУБД - практикум**“ на тема:*
*„**Онлайн Книжарница**“*

Изготвен от:

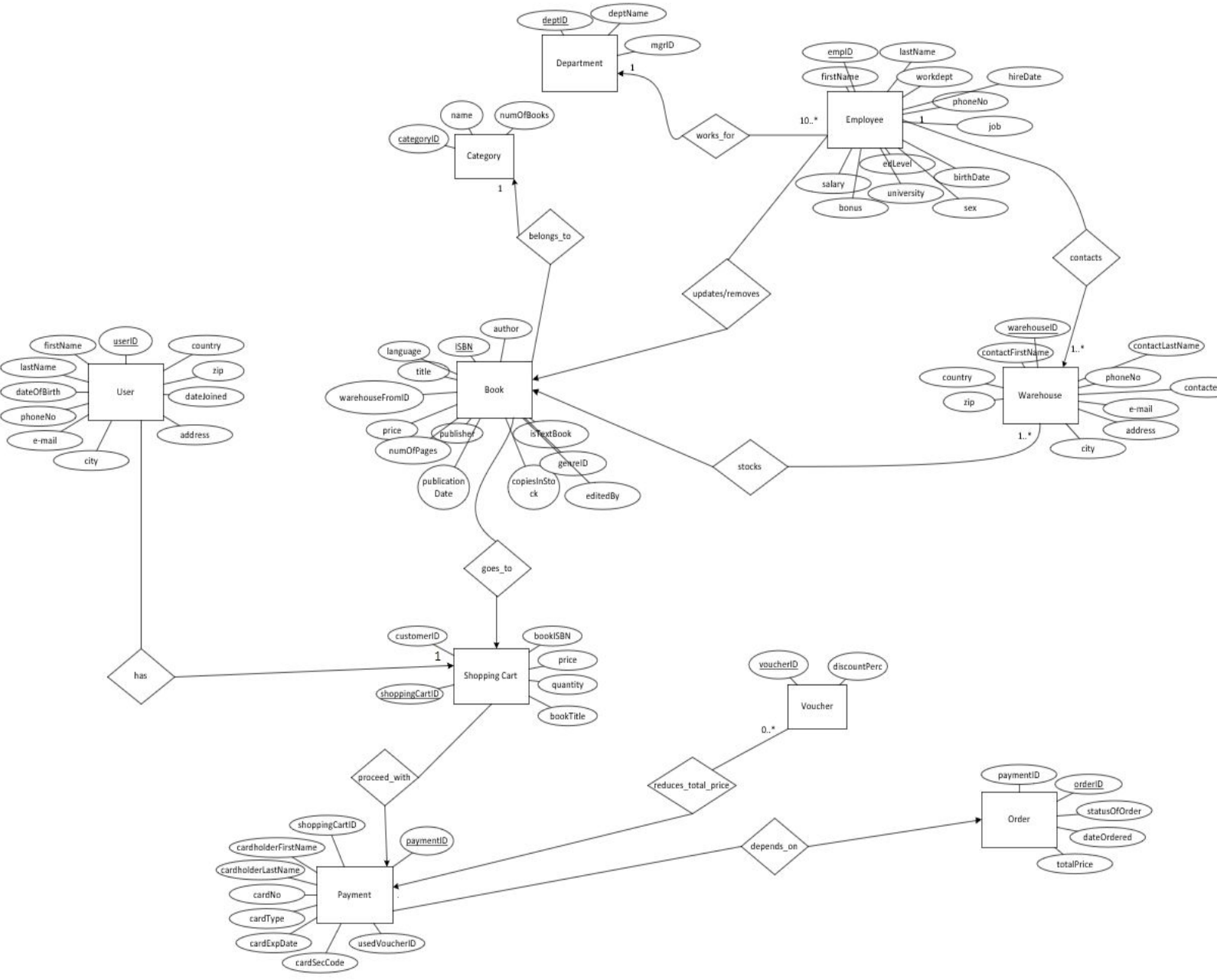
Никол Георгиева Ангелов, ФН:71871, ИС 3 курс

1. Множества от същности и техните атрибути

Базата от данни се фокусира върху различните служители, които спомагат работата на една онлайн книжарница; върху книгите, които са на разположение във въпросната книжарница и върху процеса на закупуване на книга от конкретен потребител.
По-конкретно:

- Всеки Отдел се характеризира с идентификационен номер, име, и идентификационен номер на мениджъра.
- Всеки Служител се определя от идентификационен номер, първо име, фамилия, идентификационен номер на отдел, дата на назначаване, дата на раждане, телефонен номер, професия, ниво на образование, университет, пол, заплата и бонус.
- Всеки Склад има идентификационен номер, ид на на служител, който комуникира със склада, име на служител от склада, фамилия на служител от склада, телефонен номер, имейл, адрес, град, държава, zip код.
- Всяка Категория се характеризира с идентификационен номер, име, брой книги принадлежащи на тази категория
- Всяка Книга се дефинира с ISBN, заглавие, автор, публицист, идентификационен номер на категория, език, брой страници, дата на публикуване, копия в наличност, проверка дали е тип учебник, цена, идентификационен номер на склада, който я доставя, идентификационен номер на служителя, който я редактира.
- Всеки Потребител има идентификационен номер, първо име, фамилия, дата на раждане, телефонен номер, имейл, адрес, град, държава, зип код, дата на създаване на акаунта.
- Всяка Кошница се определя от идентификационен номер, идентификационен номер на потребител, заглавие на книга, ISBN, цена, количество.
- Всеки Ваучер се дефинира с идентификационен номер и процент и отстъпка.
- Всяко Плащане се характеризира с идентификационен номер, идентификационен номер на кошницата, първо име на картодържателя, фамилия на картодържателя, номер на картата, тип на картата, срок на валидност на картата, security код на картата, идентификационен номер на използван ваучер.
- Всяка Поръчка има идентификационен номер, идентификационен номер за плащането, статус на поръчката, дата на създаване на поръчката, обща цена.

2. E/R диаграма на модела на базата от данни



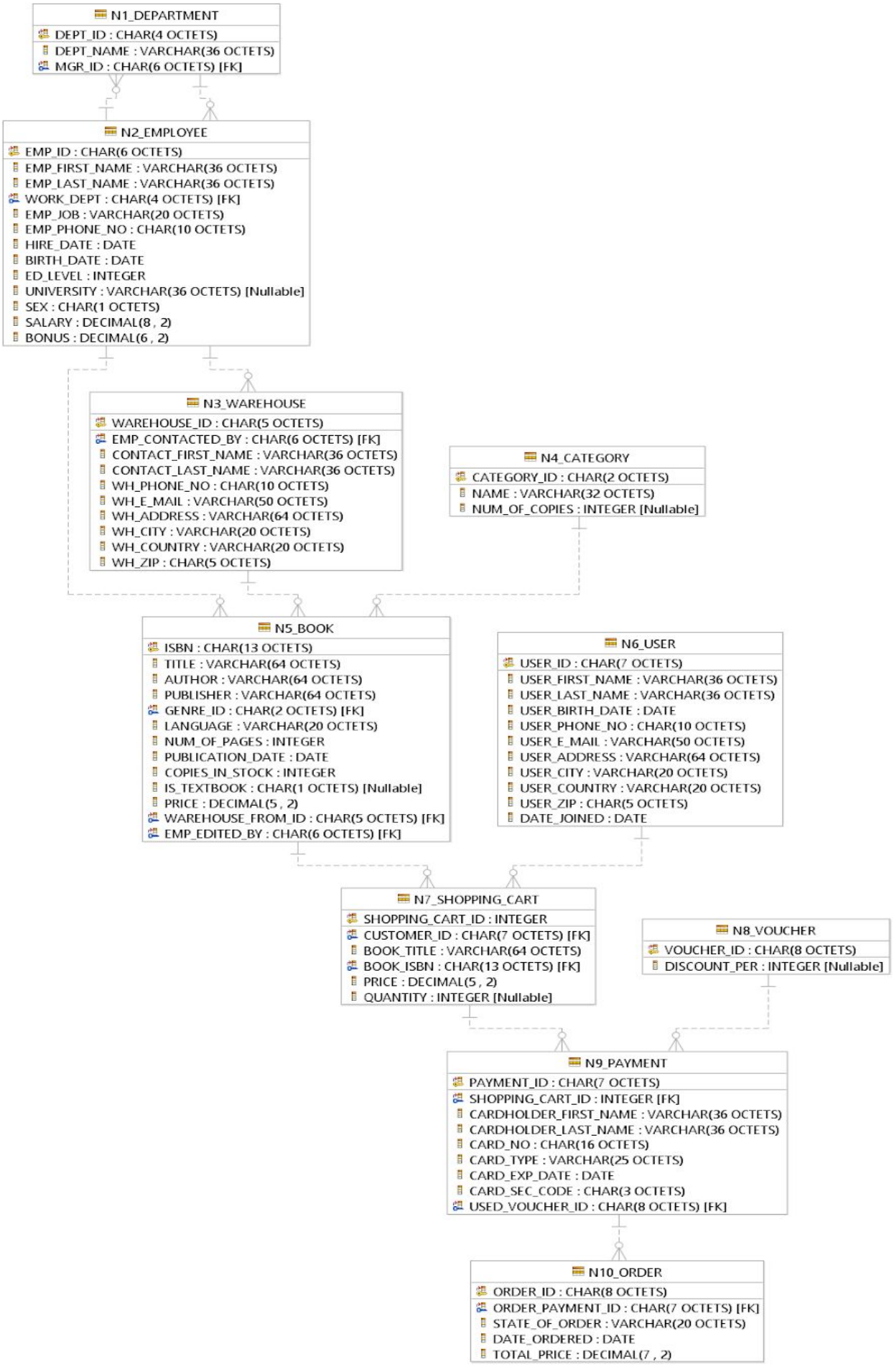
3. Преобразуване от E/R модел към релационен модел

Department (deptID, deptName, mngrID)
Employee (emplID, firstName, lastName, workDept, hireDate, birthDate, phoneNo, job, edLevel, university, salary, bonus, sex)
Warehouse (warehouseID, contactedBy, contactFirstName, contactLastName, phoneNo, email, address, city, country, zip)
Category (categoryID, name, numOfCopies)
Book (ISBN, title, author, publisher, genreID, language, numOfPages, publicationDate, copiesInStock, isTextbook, price, warehouseFromID, empEditedBy)
User (userID, firstName, lastName, birthDate, phoneNo, email, address, city, country, zip, dateJoined)
ShoppingCart (shoppingCartID, customerID, bookTitle, bookISBN, price, quantity)
Voucher (voucherID, discountPerc)
Payment (paymentID, shoppingCartID, cardholderFirstName, cardholderLastName, cardNo, cardType, cardExpDate, cardSecCode, usedVoucherID)
Order (orderID, paymentID, orderStatus, dateOrdered, totalPrice)

Преобразуване на връзки

WorksFor (deptID, emplID)
Contacts (emplID, warehouseID)
Updates/Removes (emplID, ISBN)
BelongsTo (ISBN, categoryID)
Stocks (warehouseID, ISBN)
GoesTo (ISBN, shoppingCartID)
Has (userID, shoppingCartID)
ProceedWith (shoppingCardID, paymentID)
ReducesTotalPrice (voucherID, paymentID)
DependsOn (paymentID, orderID)

4.Картинка на релационния модел от Data Studio



5.Ограничения към БД

- Един служител може да работи само в един отдел
- Една книга може да принадлежи само към една категория
- Само един служител at a time може да се свързва със склад

6.Описание на функциите

```
--1st
CREATE FUNCTION search_books_by_category (in_cat_name VARCHAR(32))
RETURNS TABLE(book_title VARCHAR(64), book_author VARCHAR(64))
RETURN SELECT TITLE AS BookTitle, AUTHOR AS BookAuthor
        FROM N5_BOOK AS BOOKS
        WHERE BOOKS.GENRE_ID IN (SELECT CATEGORY_ID
                                 FROM N4_CATEGORY
                                 WHERE NAME = in_cat_name);

SELECT *
FROM TABLE(FN71871.search_books_by_category('Arts')) as T;

--2nd
CREATE FUNCTION get_manager_of_department (in_dept_name VARCHAR(36))
RETURNS VARCHAR(72)
RETURN SELECT EMP_FIRST_NAME || ' ' || EMP_LAST_NAME
        FROM N2_EMPLOYEE AS EMPL
        WHERE EMP_JOB = 'MANAGER' AND WORK_DEPT IN (SELECT DEPT_ID
                                                    FROM N1_DEPARTMENT
                                                    WHERE DEPT_NAME = in_dept_name);

VALUES FN71871.get_manager_of_department('DISTRIBUTION');
```

Има два вида функции скаларни и таблични.
Първата функция е таблична и приема като входен параметър името на избраната категория и връща като резултат таблица с имената и авторите на книгите, които принадлежат към тази категория.
Втората функция е скаларна. При подаване на име на определен отдел връща като резултат името на мениджъра на този отдел.

7.Описание на тригерите

```
--1st
CREATE TRIGGER UPDATE_COPIES_IN_STOCK
    AFTER INSERT ON FN71871.N7_SHOPPING_CART
    REFERENCING NEW AS N
    FOR EACH ROW
    CALL FN71871.COURSE_PROJECT.UPDATE_BOOK_STOCK(N.BOOK_ISBN, N.QUANTITY);

INSERT INTO FN71871.N7_SHOPPING_CART (CUSTOMER_ID, BOOK_TITLE, BOOK_ISBN, PRICE, QUANTITY)
VALUES ('9875851', 'It', '9781982127794', 17.99, 2);

--2nd
CREATE TRIGGER UPDATE_BOOKS_IN_A_CAT
    AFTER INSERT ON FN71871.N5_BOOK
    REFERENCING NEW AS N
    FOR EACH ROW
    CALL FN71871.COURSE_PROJECT.UPDATE_BOOKS_IN_A_CATEGORY(N.GENRE_ID);


INSERT INTO N5_BOOK
VALUES ('9780063072565', 'Just As I Am', 'Cicely Tyson', 'Barnes & Noble', '25', 'Bulgarian', 432,
                                    DATE '2021-01-26', 0, 'N', 28.99, '00007', '000050');

--3rd
CREATE TRIGGER UPDATE_BONUS_ACCORDINGLY
    AFTER UPDATE ON FN71871.N2_EMPLOYEE
    REFERENCING NEW AS N OLD AS O
    FOR EACH ROW
        WHEN (N.SALARY <> O.SALARY)
            UPDATE FN71871.N2_EMPLOYEE
            SET BONUS = BONUS + (N.BONUS / 4.0)
            WHERE N.EMP_ID = EMP_ID;
```

Тригерите са само от тип AFTER.
Първият тригер се изпълнява след като е въведен нов ред в таблицата N7_SHOPPING_CART и съответното действие е да се извика процедурата UPDATE_BOOK_STOCK (действието е описано в *т.8*), въвеждайки нужните входни параметри.
Вторият тригер се изпълнява след като е въведен нов ред в таблицата N5_BOOK и съответно се извиква процедурата UPDATE_BOOKS_IN_CATEGORY (действието е описано в *т.8*), въвеждайки нужните входни параметри.
Третият тригер има за цел да обнови бонуса в съответствие с обновяването на нечия заплата. Следователно той се изпълнява, когато се извърши промяна върху таблицата N2_EMPLOYEE, но при условие, че има промяна при SALARY атрибута, и тогава към стария BONUS се прибавя и ¼ от него.

8.Описание на изгледите

```
--1st
CREATE VIEW list_of_all_books_in_stock
AS
SELECT TITLE, AUTHOR
FROM N5_BOOK
WHERE COPIES_IN_STOCK > 0;

SELECT * from list_of_all_books_in_stock;

--2nd
CREATE VIEW purchase
AS
SELECT CARDHOLDER_FIRST_NAME || ' ' || CARDHOLDER_LAST_NAME AS NAME, TOTAL_PRICE
FROM N9_PAYMENT, N10_ORDER
WHERE PAYMENT_ID = ORDER_PAYMENT_ID;

SELECT * FROM purchase;

--3rd
CREATE VIEW company_payments
AS
SELECT EMP_FIRST_NAME || ' ' || EMP_LAST_NAME AS NAME, WORK_DEPT, EMP_JOB, SALARY, BONUS
FROM FN71871.N2_EMPLOYEE
WHERE ED_LEVEL = 12;

SELECT * FROM company_payments;
```

Първият изглед се състои от резултата на заявка към таблицата N5_BOOK. Атрибутите на този изглед са заглавието и автора на книги, които са в наличност.
Вторият изглед се състои от резултата на заявка, която прави съединение на таблиците N9_PAYMENT и N10_ORDER. Атрибутите на този изглед са имената на картодържателя (човека, който трябва да плати) и общата цена (съответно, дължимата сума).
Третият изглед се състои от резултата на заявка към таблицата N2_EMPLOYEE. Атрибутите на този изглед са имената на служителя, неговият отдел, професията, заплатата и бонуса, който получава. Заявката извежда горната информация само за служителите, които имат най-ниското ниво на образование(EDLEVEL = 12).

9.Описание на процедурите

```
ALTER MODULE FN71871.COURSE_PROJECT PUBLISH PROCEDURE CALCULATE_TOTAL_PRICE(IN V_CUSTOMER_ID CHAR(7))
BEGIN
    DECLARE SQLCODE INT;
    DECLARE TEMP_TOTAL_PRICE DECIMAL(7,2) DEFAULT 0;
    DECLARE USED_ORDER_ID CHAR(8);
    DECLARE USED_VOUCHER CHAR(8);
    DECLARE DISCOUNT_PERC INTEGER;

    DECLARE CUSTOMER ANCHOR ROW FN71871.N7_SHOPPING_CART;

    DECLARE CURR CURSOR WITH RETURN FOR SELECT * FROM FN71871.N7_SHOPPING_CART WHERE FN71871.N7_SHOPPING_CART.CUSTOMER_ID = V_CUSTOM

    SET USED_ORDER_ID = (SELECT ORDER_ID
                         FROM FN71871.N10_ORDER
                         WHERE ORDER_PAYMENT_ID IN
                                        (SELECT PAYMENT_ID
                                         FROM FN71871.N9_PAYMENT
                                         WHERE SHOPPING_CART_ID IN
                                                        (SELECT SHOPPING_CART_ID
                                                        FROM FN71871.N7_SHOPPING_CART
                                                        WHERE CUSTOMER_ID = V_CUSTOMER_ID)));

    SET USED_VOUCHER = (SELECT USED_VOUCHER_ID
                        FROM FN71871.N9_PAYMENT
                        WHERE PAYMENT_ID IN
                                       (SELECT ORDER_PAYMENT_ID
                                        FROM FN71871.N10_ORDER
                                        WHERE ORDER_ID = USED_ORDER_ID));

    OPEN CURR;
    FETCH CURR INTO CUSTOMER;
    WHILE SQLCODE = 0 DO
        SET TEMP_TOTAL_PRICE = TEMP_TOTAL_PRICE + (CUSTOMER.PRICE * CUSTOMER.QUANTITY);
        FETCH CURR INTO CUSTOMER;
    END WHILE;

    SET DISCOUNT_PERC = (SELECT DISCOUNT_PER FROM FN71871.N8_VOUCHER WHERE USED_VOUCHER = FN71871.N8_VOUCHER.VOUCHER_ID);

    UPDATE FN71871.N10_ORDER
    SET TOTAL_PRICE = TEMP_TOTAL_PRICE - ((DISCOUNT_PERC / 100.0) * TEMP_TOTAL_PRICE)
    WHERE FN71871.N10_ORDER.ORDER_ID = USED_ORDER_ID;

    CLOSE CURR;
END
```

Първата процедура има за цел да изчислява общата цена на избраните книги (поставени в кошницата) от всеки потребител. При подаване на идентификационен номер на купувач, се стартира процедурата, в която са декларирани няколко променливи, които спомагат извършването на процедурата и се създава един курсор, който ми дава информация, ред по ред, за това какво има в кошницата на определеният потребител (избран при подаването на ид номер). Задава се стойност на USER_ORDER_ID и на USED_VOUCHER, за да мога да обновя правилният ред в таблицата N10_ORDER и за да използвам правилната отстъпка за всеки потребител. А самото изчисление на общата цена (взимаме цената на всяка избрана книга и количеството и ги умножаваме) се извършва с помощта на по-горе споменатият курсор и while цикъл, който обхожда таблицата докато не бъде достигнат последния ред.

```
ALTER MODULE FN71871.COURSE_PROJECT PUBLISH PROCEDURE UPDATE_ORDER_STATUS(IN V_CUSTOMER_ID CHAR(7))
BEGIN
    DECLARE AT_END INTEGER DEFAULT 0;
    DECLARE ACC_ORDER_ID CHAR(8);
    DECLARE ORD ANCHOR ROW FN71871.N10_ORDER;

    DECLARE NOT_FOUND CONDITION FOR SQLSTATE '02000';
    DECLARE INVALID_CUSTOMER_ID CONDITION FOR SQLSTATE '42884';
    DECLARE WRONG_ORDER CONDITION FOR SQLSTATE '22003';

    DECLARE CURR_ORDER CURSOR WITH RETURN FOR SELECT * FROM FN71871.N10_ORDER WHERE FN71871.N10_ORDER.ORDER_ID = ACC_ORDER_ID;


    DECLARE CONTINUE HANDLER FOR INVALID_CUSTOMER_ID
        RESIGNAL SET MESSAGE_TEXT = 'This user ID does not exist!';
    DECLARE CONTINUE HANDLER FOR WRONG_ORDER
        RESIGNAL SET MESSAGE_TEXT = 'Something went wrong with the delivery!';
    DECLARE CONTINUE HANDLER FOR NOT_FOUND
        SET AT_END = 1;

    SET ACC_ORDER_ID = (SELECT ORDER_ID
                        FROM FN71871.N10_ORDER
                        WHERE ORDER_PAYMENT_ID IN
                                            (SELECT PAYMENT_ID
                                             FROM FN71871.N9_PAYMENT
                                             WHERE SHOPPING_CART_ID IN
                                                                 (SELECT SHOPPING_CART_ID
                                                                 FROM FN71871.N7_SHOPPING_CART
                                                                 WHERE CUSTOMER_ID = V_CUSTOMER_ID)));


    OPEN CURR_ORDER;
    IF V_CUSTOMER_ID NOT IN(SELECT USER_ID FROM FN71871.N6_USER) THEN SIGNAL INVALID_CUSTOMER_ID;
    ELSE
        L1: LOOP
            FETCH CURR_ORDER INTO ORD;
            IF AT_END = 1 THEN LEAVE L1;
            ELSEIF (ORD.STATE_OF_ORDER IN ('ERROR', 'IN PROCESS', 'SENT')) THEN
                IF (ORD.DATE_ORDERED = CURRENT_DATE OR CURRENT_DATE - ORD.DATE_ORDERED < 3 ) THEN
                    UPDATE FN71871.N10_ORDER
                    SET STATE_OF_ORDER = 'IN PROCESS'
                    WHERE ACC_ORDER_ID = FN71871.N10_ORDER.ORDER_ID;
                ELSEIF (CURRENT_DATE - ORD.DATE_ORDERED > 2  AND  CURRENT_DATE - ORD.DATE_ORDERED < 6)  THEN

                  UPDATE FN71871.N10_ORDER
                  SET STATE_OF_ORDER = 'SENT'
                  WHERE ACC_ORDER_ID = FN71871.N10_ORDER.ORDER_ID;
              ELSEIF (CURRENT_DATE - ORD.DATE_ORDERED > 5  AND  CURRENT_DATE - ORD.DATE_ORDERED < 10)  THEN
                  UPDATE FN71871.N10_ORDER
                  SET STATE_OF_ORDER = 'DELIVERED'
                  WHERE ACC_ORDER_ID = FN71871.N10_ORDER.ORDER_ID;
              ELSE SIGNAL WRONG_ORDER;
              END IF;
            END IF;
        END LOOP;
END IF;

CLOSE CURR_ORDER;

$
```

Втората процедура има за цел да обновява статуса на поръчките. Приема като входен параметър идентификационния номер на потребителя, който е извършил поръчката. Създадени са няколко прихващания на изключения, които имат за цел да сигнализират 1) ако този ид номер не съществува, 2) ако са минали 10 дена или повече от направата на поръчката, но не е със статус ‘DELIVERED’ и 3) в случай, че е достигнат края на таблицата.

Процедурата има създаден курсор, който осигурява информация от N10_ORDER таблицата. Преди да започне да се проверява статуса на поръчката се прави проверка дали номерът на този потребител съществува и ако го няма се сигнализира за грешка. Чрез loop се проверяват всички условия, взимайки информация от курсора, ред по ред, и в случай, че статуса на поръчката е error, in process или sent, се обновява таблицата

според това колко дена са изминали откакто е била направена поръчката, но в случай, че са повече от 10, се сигнализира за грешка и процедурата бива прекъсната.

```
ALTER MODULE FN71871.COURSE_PROJECT PUBLISH PROCEDURE UPDATE_MONTHLY_SALARY(IN V_EMP_ID CHAR(6), OUT V_HIRE_DATE DATE,
                                        OUT V_JOB VARCHAR(20), OUT OLD_SALARY DECIMAL(8, 2), OUT NEW_SALARY DECIMAL(8, 2))
RESULT SETS 1
BEGIN
    DECLARE EMP_INFO ANCHOR ROW FN71871.N2_EMPLOYEE;

    DECLARE CURR_EMPL CURSOR WITH RETURN FOR SELECT * FROM FN71871.N2_EMPLOYEE WHERE FN71871.N2_EMPLOYEE.EMP_ID = V_EMP_ID;

    OPEN CURR_EMPL;

    FETCH CURR_EMPL INTO EMP_INFO;
    SET V_HIRE_DATE = EMP_INFO.HIRE_DATE;
    SET V_JOB = EMP_INFO.EMP_JOB;
    SET OLD_SALARY = EMP_INFO.SALARY;

    IF V_JOB = 'MANAGER' THEN
        UPDATE FN71871.N2_EMPLOYEE
        SET SALARY = SALARY + 300 WHERE FN71871.N2_EMPLOYEE.EMP_ID = V_EMP_ID;
        SET NEW_SALARY = EMP_INFO.SALARY + 300;
    ELSEIF V_JOB IN ('ACCOUNTANT', 'LAWYER')  THEN
        UPDATE FN71871.N2_EMPLOYEE
        SET SALARY = SALARY + 220 WHERE FN71871.N2_EMPLOYEE.EMP_ID = V_EMP_ID;
        SET NEW_SALARY = EMP_INFO.SALARY + 220;
    ELSEIF V_JOB IN ('ANALYST', 'CLERK', 'FIELDREP') THEN
        UPDATE FN71871.N2_EMPLOYEE
        SET SALARY = SALARY + 130 WHERE FN71871.N2_EMPLOYEE.EMP_ID = V_EMP_ID;
        SET NEW_SALARY = EMP_INFO.SALARY + 130;
    ELSE
        UPDATE FN71871.N2_EMPLOYEE
        SET SALARY = SALARY + 170 WHERE FN71871.N2_EMPLOYEE.EMP_ID = V_EMP_ID;
        SET NEW_SALARY = EMP_INFO.SALARY + 170;
    END IF;


    CLOSE CURR_EMPL;
END @
```

Третата процедура има за цел да обновява заплатата на конкретен служител и да извежда определена информация, след извършване на повишението.
Деклариран е един курсор, който дава информация N2_EMPLOYEE и една променлива, която съдържа информация за ред от тази таблица.
Задава се стойност на изходните параметри V_HIRE_DATE, V_JOB, OLD_SALARY с помощта на курсора и променливта, споменати по-горе.
След това се проверява професията на избрания служител и според нея се извършва съответното увеличение на заплатата.
Последното, което се извършва е да се зададе стойност на изходния параметър NEW_SALARY.

```
--++++ procedure
ALTER MODULE FN71871.COURSE_PROJECT PUBLISH PROCEDURE UPDATE_BOOK_STOCK(IN V_ISBN CHAR(13), IN V_QUANTITY INTEGER)
BEGIN
    DECLARE SQLCODE INT;
    DECLARE V_BOOK ANCHOR ROW FN71871.N5_BOOK;

    DECLARE CURR_STOCK CURSOR WITH RETURN FOR SELECT * FROM FN71871.N5_BOOK WHERE V_ISBN = FN71871.N5_BOOK.ISBN;

    OPEN CURR_STOCK;
    FETCH CURR_STOCK INTO V_BOOK;
    WHILE SQLCODE = 0 DO
        UPDATE FN71871.N5_BOOK
        SET COPIES_IN_STOCK = COPIES_IN_STOCK - V_QUANTITY
        WHERE FN71871.N5_BOOK.ISBN = V_ISBN;
        FETCH CURR_STOCK INTO V_BOOK;
    END WHILE;
    CLOSE CURR_STOCK;

END $

CALL FN71871.COURSE_PROJECT.UPDATE_BOOK_STOCK('9780062976581', 2)$

--++++ PROCEDURE
ALTER MODULE FN71871.COURSE_PROJECT PUBLISH PROCEDURE UPDATE_BOOKS_IN_A_CATEGORY(IN V_GENRE_ID CHAR(2))
BEGIN
    DECLARE SQLCODE INT;
    DECLARE V_CATEGORY ANCHOR ROW FN71871.N4_CATEGORY;

    DECLARE C1 CURSOR WITH RETURN FOR SELECT * FROM FN71871.N4_CATEGORY WHERE V_GENRE_ID = FN71871.N4_CATEGORY.CATEGORY_ID;

    OPEN C1;
    FETCH C1 INTO V_CATEGORY;
    WHILE SQLCODE = 0 DO
        UPDATE FN71871.N4_CATEGORY
        SET NUM_OF_COPIES = NUM_OF_COPIES + 1
        WHERE FN71871.N4_CATEGORY.CATEGORY_ID = V_GENRE_ID;
        FETCH C1 INTO V_CATEGORY;
    END WHILE;
    CLOSE C1;

END $
```

Четвъртата процедура има за цел да обнови таблицата N5_BOOK, като премахне определено количество от наличното количество на конкретна книга. Използвам курсор за да се сдобия с информацията от въпросната таблица и с помощта на while цикъл, минавайки през таблицата ред по ред, намирам нужната книга и редуцирам количеството на книгите със зададеното количество.

Последната процедура има за цел да обнови количеството книги от определена категория, поради тази причина се подава идентификационен номер на категорията. Процедурата функционира по същия начин, като четвъртата, но се извършва увеличение само с 1.