Лабораторная работа №3

**РЕАЛИЗАЦИЯ СИСТЕМЫ ОБМЕНА ДАННЫМИ С БД**

**Цель работы:** научиться разрабатывать систему обмена данными с БД.

**Выполнение работы**

1. Концептуальная модель:

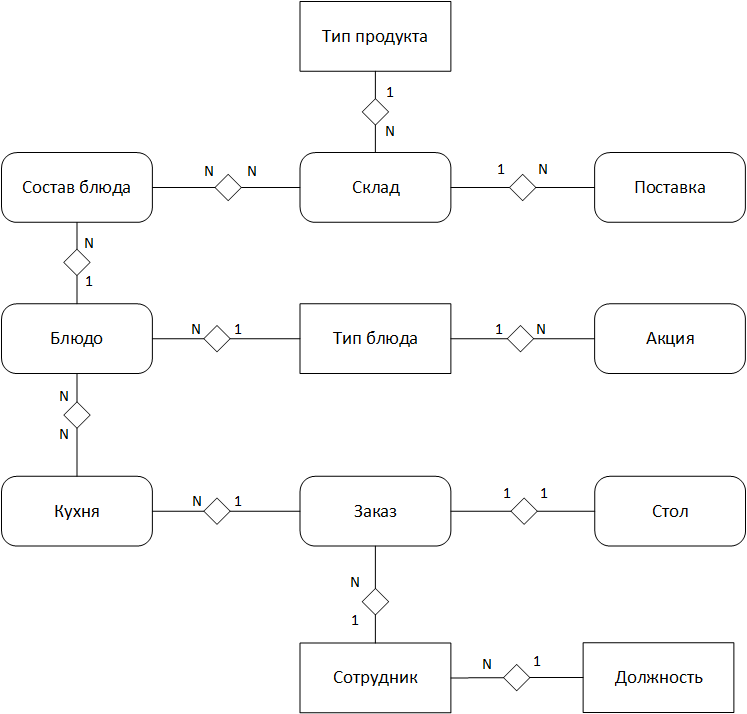


Рисунок 1 – Концептуальная модель

1. Логическая модель:

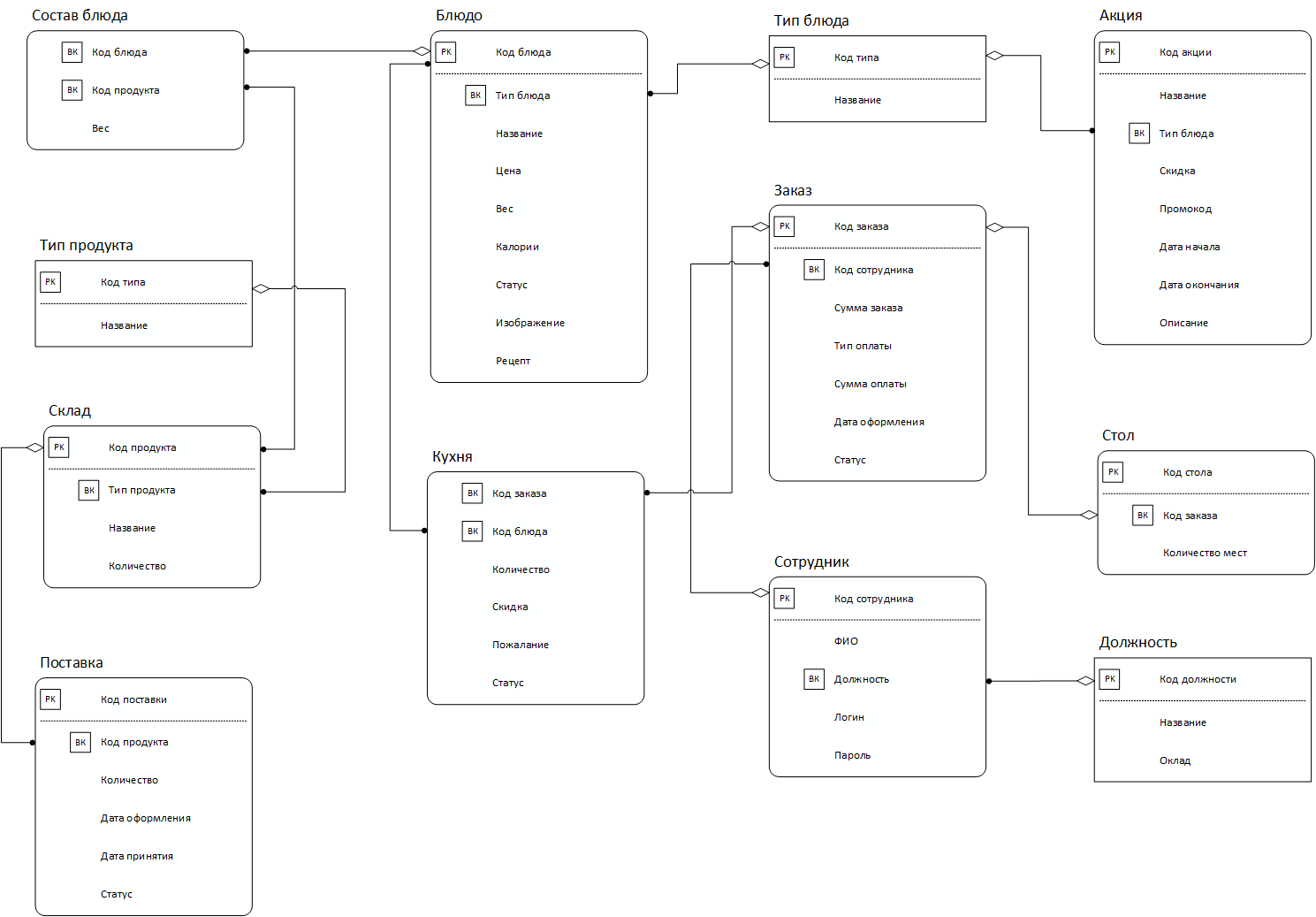


Рисунок 2 – Логическая модель

**Характеристика мощностей связей между сущностями**

Максимальная мощность между сущностями «Блюдо» – «Состав блюда» –1:*N*: один экземпляр сущности «Блюдо» может относиться ко многим экземплярам сущности «Состав блюда», и один экземпляр сущности «Состав блюда» может относиться только к одному экземпляру сущности «Блюдо».

Максимальная мощность между сущностями «Блюдо» – «Тип блюда» –*N*:1: один экземпляр сущности «Блюдо» может относиться только к одному экземпляру сущности «Тип блюда», и один экземпляр сущности «Тип блюда» может относиться ко многим экземплярам сущности «Блюдо».

Максимальная мощность между сущностями «Блюдо» – «Кухня» – *N*:*N*: один экземпляр сущности «Блюдо» может относиться ко многим экземплярам сущности «Кухня», и один экземпляр сущности «Кухня» может относиться только ко многим экземплярам сущности «Блюдо».

Максимальная мощность между сущностями «Состав блюда» – «Склад» – *N*:*N*: один экземпляр сущности «Состав блюда» может относиться ко многим экземплярам сущности «Склад», и один экземпляр сущности «Склад» может относиться ко многим экземплярам сущности «Состав блюда».

Максимальная мощность между сущностями «Склад» – «Тип продукта» –*N*:1: один экземпляр сущности «Склад» может относиться только к одному экземпляру сущности «Тип продукта», и один экземпляр сущности «Тип продукта» может относиться ко многим экземплярам сущности «Склад»

Максимальная мощность между сущностями «Склад» – «Поставка» - 1:*N*: один экземпляр сущности «Склад» может относиться ко многим экземплярам сущности «Поставка», и один экземпляр сущности «Поставка» может относиться только к одному экземпляру сущности «Склад»

Максимальная мощность между сущностями «Тип блюда» – «Акция» 1:*N*: один экземпляр сущности «Тип блюда» может относиться ко многим экземплярам сущности «Акция», и один экземпляр сущности «Акция» может относиться только к одному экземпляру сущности «Тип блюда»

Максимальная мощность между сущностями «Сотрудник» – «Должность» –*N*:1: один экземпляр сущности «Сотрудник» может относиться только к одному экземпляру сущности «Должность», и один экземпляр сущности «Должность» может относиться ко многим экземплярам сущности «Сотрудник»

Максимальная мощность между сущностями «Сотрудник» – «Заказ» –1:*N*: один экземпляр сущности «Сотрудник» может относиться ко многим экземплярам сущности «Заказ», и один экземпляр сущности «Заказ» может относиться только к одному экземпляру сущности «Сотрудник».

Максимальная мощность между сущностями «Заказ» – «Кухня» –*N*:*N*: один экземпляр сущности «Заказ» может относиться ко многим экземплярам сущности «Кухня», и один экземпляр сущности «Кухня» может относиться ко многим экземплярам сущности «Заказ».

Максимальная мощность между сущностями «Заказ» – «Стол» –1:1: один экземпляр сущности «Заказ» может относиться только к одному экземпляру сущности «Стол», и один экземпляр сущности «Стол» может относиться только к одному экземпляру сущности «Заказ».

**Описание связей модели**

ЗАКАЗ <выполняется> одним СОТРУДНИКОМ;

СОТРУДНИК <имеет> одну ДОЛЖНОСТЬ;

СТОЛ <резервируется> под один ЗАКАЗ;

ПРОДУКТ <относится> к одному ТИПУ ПРОДУКТА;

ПОСТАВКА <оформляется> на один ПРОДУКТ;

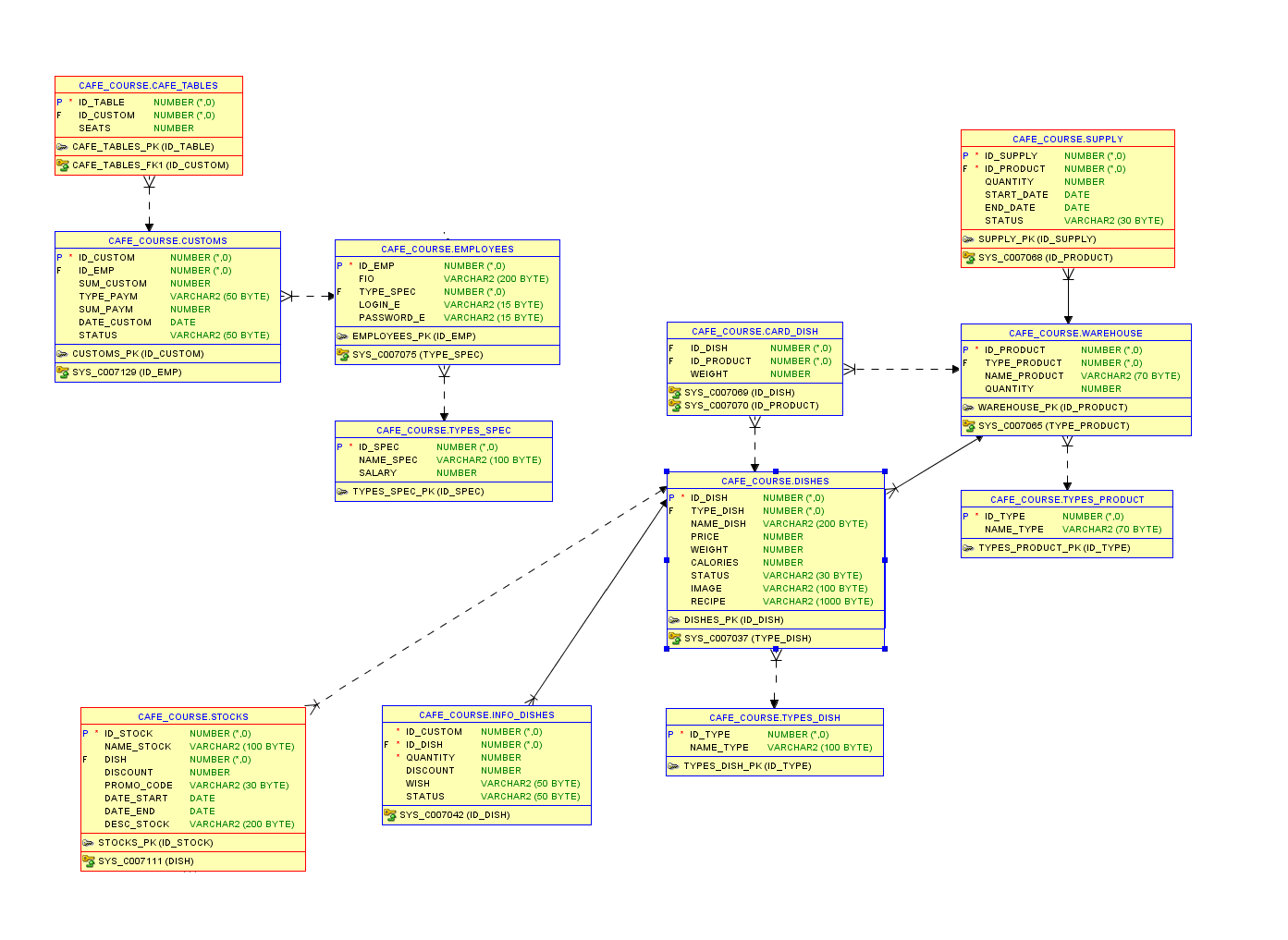
АКЦИЯ <оформляется> на один ТИП БЛЮДА;

БЛЮДО <относится> к одному ТИПУ БЛЮДА;

СОСТАВ БЛЮДА <относится> к одному БЛЮДУ и <включает в себя> много ПРОДУКТОВ;

КУХНЯ <выполняет> много ЗАКАЗОВ и <включает в себя> много БЛЮД.

1. Физическая модель:



**Описание таблиц БД**

1. Таблица *cafe\_tables* хранит информацию о столах, а именно: номер стола (*id\_table* типа *integer*, первичный ключ), номер заказа (*id\_custom* типа *integer*, внешний) и количество место (*seats* типа *number*);
2. Таблица *customs* хранит информацию о заказах, а именно: номер заказа (*id\_custom* типа *integer*, первичный ключ), код сотрудника (*id\_emp* типа *integer*, внешний ключ), сумма заказа (*sum\_custom* типа *number*), тип (*type\_paym* типа *varchar2*) и сумма оплаты (*sum\_paym* типа *number*), дата заказа (*date\_custom* типа *date*) и статус (*status* типа *varchar2*);
3. Таблица *employees* хранит информацию о сотрудника, а именно: код сотрудника (*id\_emp* типа *integer*, первичный ключ), ФИО (*fio* типа *varchar2*), должность (*type\_spec* типа *integer*, внешний ключ), логин (*login\_e* типа *varchar2*) и пароль (*password\_e* типа *varchar2*) для входа в систему;
4. Таблица *types\_spec* хранит информацию о должностях, а именно: код должности (*id\_spec* типа *integer*, первичный ключ), название (*name\_spec* типа *varchar2*) и оклад (*salary* типа *number*);
5. Таблица *supply* хранит информацию о поставках, а именно: код поставки (*id\_supply* типа *integer*, первичный ключ), код продукта (*id\_product* типа *integer*, внешний ключ), количество (*quantity* типа *number*), дата оформления (*start\_date* типа *date*) и принятия (*end\_date* типа *date*) и статус (*status* типа *varchar2*);
6. Таблица *warehouse* хранит информацию о продуктах, а именно: код продукта (*id\_product* типа *integer*, первичный ключ), тип (*type\_product* типа *integer*, внешний ключ), название (*name\_product* типа *varchar2*) и количество (*quantity* типа *number*);
7. Таблица *types\_product* хранит информацию о типах продуктов, а именно: код типа (*id\_type* типа *integer*, первичный ключ) и название (*name\_type* типа *varchar2*);
8. Таблица *card\_dish* хранит информацию о составе блюд, а именно: код блюда (*id\_dish* типа *integer*, первичный ключ), код продукта (*id\_product* типа *integer*, внешний ключ) и вес (*weight* типа *number*);
9. Таблица *dishes* хранит информацию о блюдах, а именно: код блюда (*id\_dish* типа *integer*, первичный ключ), тип (*type\_dish* типа *integer*, внешний ключ), название (*name\_dish* типа *varchar2*), цена (*price* типа *number*), вес (*weight* типа *number*), калории (*calories* типа *number*), статус (*status* типа *varchar2*), изображение (*image* типа *varchar2*) и описание (*recipe* типа *varchar2*);
10. Таблица *types\_dish* хранит информацию о видах блюд, а именно: код типа (*id\_type* типа *integer*, первичный ключ) и название (*name\_type* типа *varchar2*);
11. Таблица *info\_dishes* хранит информацию о заказанных блюдах, а именно: номер заказа (*id\_custom* типа *integer*, внешний ключ), код блюда (*id\_dish* типа *integer*, внешний ключ), количество (*quantity* типа *number*), скидка (*discount* типа *number*), пожелание (*wish* типа *varchar2*) и статус (*status* типа *varchar2*);
12. Таблица *stocks* хранит информацию об акциях, а именно: код акции (*id\_stock* типа *integer*, первичный ключ), название (*name\_stock* типа *varchar2*), код блюда (*dish* типа *integer*, внешний ключ), скидка (*discount* типа *number*), промокод (*promo\_code* типа *varchar2*), дата начала (*date\_start* типа *date*) и окончания (*date\_end* типа *date*), описание акции (*desc\_stock* типа *varchar2*).

**Функциональные запросы**

* Добавить блюдо в заказ:

INSERT INTO Info\_dishes VALUES(@id\_custom, @id\_dish, @quantity, @discount, @wish,'');

где *@id\_custom, @id\_dish, @quantity, @discount, @wish -* условные обозначения подставляемых значений из формы приложения для внесения в таблицу данных о номере заказа, номере блюда, количестве блюда, скидке и пожелании гостя соответственно;

* Изменить количество блюда в заказе:

UPDATE Info\_dishes

SET Quantity = Quantity + @quantity

WHERE Id\_dish = @id\_dish AND Id\_custom = @id\_custom

где *Quantity –* имя поля таблицы, в котором хранится информация о количестве блюда;

*Id\_dish –* имя поля таблицы, в котором хранится информация о номере блюда;

*Id\_custom –* имя поля таблицы, в котором хранится информация о номере заказа;

*@quantity, @id\_dish, @id\_custom –* условные обозначения подставляемых значений из формы приложения;

* Проверка наличия скидки на блюдо

CREATE OR REPLACE FUNCTION Dish\_discount (@Id\_d INTEGER)

RETURN NUMBER IS

Count\_d NUMBER;

Disc\_d NUMBER;

BEGIN

SELECT COUNT(\*) INTO Count\_d

FROM Stocks

WHERE Date\_start < sysdate AND Date\_end > sysdate AND Dish = @Id\_d;

IF Count\_d != 0 THEN

SELECT Discount INTO Disc\_d

FROM Stocks

WHERE Date\_start < sysdate AND Date\_end > sysdate AND Dish = @Id\_d;

ELSE

Disc\_d := 0;

END IF;

RETURN Disc\_d;

END;

где *Disc\_d –* переменная, хранящая информацию о скидке блюда;

*@Id\_d -* условное обозначение подставляемого значения из формы приложения;

* Оформить заказ

UPDATE Customs

SET Sum\_custom = @price\_c, Status ='Ожидается оплата'

WHERE Id\_custom = @id\_custom;

где *Sum\_custom –* имя поля таблицы, в котором хранится информация о сумме заказа;

*Status –* имя поля таблицы, в котором хранится информация о статусе заказа;

*@price\_c, @id\_custom –* условные обозначения подставляемых значений из формы приложения;

* Завершить заказ

UPDATE Customs

SET Status = 'Завершен', Sum\_paym = @sum\_paym, Type\_paym = @type\_paym

WHERE Id\_custom = @id\_custom

где *Status –* имя поля таблицы, в котором хранится информация о статусе заказа;

*Sum\_paym –* имя поля таблицы, в котором хранится информация о сумме оплаты;

*Type\_paym –* имя поля таблицы, в котором хранится информация о типе оплаты;

*Id\_custom –* имя поля таблицы, в котором хранится информация о номере заказа;

*@sum\_paym, @type\_paym, @id\_custom –* условные обозначения подставляемых значений из формы приложения;

* Отменить заказ

UPDATE Customs

SET Status = 'Отменен'

WHERE Id\_custom = @id\_custom

где *Status –* имя поля таблицы, в котором хранится информация о статусе заказа;

*Id\_custom –* имя поля таблицы, в котором хранится информация о номере заказа;

*@id\_custom –* условное обозначение подставляемого значения из формы приложения;

UPDATE Info\_dishes

SET Status = 'Отменено'

WHERE Id\_custom = @id\_custom

где *Status –* имя поля таблицы, в котором хранится информация о статусе блюда;

*Id\_custom –* имя поля таблицы, в котором хранится информация о номере заказа;

*@id\_custom –* условное обозначение подставляемого значения из формы приложения;

* Приготовить блюдо:

UPDATE Info\_dishes

SET Status = 'Завершено'

WHERE Id\_dish = @id\_dish AND Id\_custom = @id\_custom

где *Status –* имя поля таблицы, в котором хранится информация о статусе блюда;

*Id\_dish –* имя поля таблицы, в котором хранится информация о номере блюда;

*Id\_custom –* имя поля таблицы, в котором хранится информация о номере заказа;

*@id\_custom, @id\_dish –* условные обозначения подставляемых значений из формы приложения;

CREATE OR REPLACE PROCEDURE Complete\_dish

IS

Id\_d INTEGER;

Quant NUMBER;

W NUMBER;

TYPE GenericCursor IS REF CURSOR;

c1 GenericCursor;

c2 GenericCursor;

c3 GenericCursor;

TYPE TableIdC IS TABLE OF Info\_dishes.Id\_custom%TYPE;

Id\_customs TableIdC;

TYPE TableIdD IS TABLE OF Info\_dishes.Id\_dish%TYPE;

Id\_dishes TableIdD;

TYPE TableIdP IS TABLE OF Warehouse.Id\_product%TYPE;

Id\_products TableIdP;

BEGIN

OPEN c1 FOR SELECT DISTINCT Id\_custom

FROM Info\_dishes

WHERE Status = 'Завершено';

FETCH c1 BULK COLLECT INTO Id\_customs;

CLOSE c1;

IF Id\_customs IS NOT NULL THEN

FOR i IN Id\_customs.first..Id\_customs.last

LOOP

OPEN c2 FOR SELECT Id\_dish

FROM Info\_dishes

WHERE Id\_custom = Id\_customs(i) AND Status = 'Завершено';

FETCH c2 BULK COLLECT INTO Id\_dishes;

CLOSE c2;

IF Id\_dishes IS NOT NULL THEN

FOR j IN Id\_dishes.first..Id\_dishes.last

LOOP

SELECT Quantity INTO Quant

FROM Info\_dishes

WHERE Id\_dish = Id\_dishes(j) AND Id\_custom = Id\_customs(i);

OPEN c3 FOR SELECT Id\_product

FROM Card\_dish

WHERE Id\_dish = Id\_dishes(j);

FETCH c3 BULK COLLECT INTO Id\_products;

CLOSE c3;

IF Id\_products IS NOT NULL THEN

FOR k IN Id\_products.first..Id\_products.last

LOOP

SELECT Weight/1000\*Quant INTO W

FROM Card\_dish

WHERE Id\_dish = Id\_dishes(j) AND Id\_product = Id\_products(k);

UPDATE Warehouse

SET Quantity = Quantity - W

WHERE Id\_product = Id\_products(k);

END LOOP;

END IF;

UPDATE Info\_dishes

SET Status = 'Готово'

WHERE Id\_custom = Id\_customs(i) AND Id\_dish = Id\_dishes(j);

END LOOP;

END IF;

END LOOP;

END IF;

END;

* Оформить поставку

INSERT INTO Supply VALUES ('', @id\_product, @quantity, sysdate, '' ,'Ожидается');

где *@id\_product, @quantity -* условные обозначения подставляемых значений из формы приложения для внесения в таблицу данных о номере продукта и его количестве соответственно;

* Списать продукты

UPDATE Warehouse

SET Quantity = Quantity – @quantity

WHERE Id\_product = @id\_product

где *Quantity –* имя поля таблицы, в котором хранится информация о количестве продукта;

*Id\_product –* имя поля таблицы, в котором хранится информация о номере продукта;

*@id\_product –* условное обозначение подставляемого значения из формы приложения;

* Сформировать/Обновить стоп-лист

CREATE OR REPLACE PROCEDURE Stop\_list

IS

Count\_pr NUMBER;

Count\_dish NUMBER;

TYPE GenericCursor IS REF CURSOR;

c1 GenericCursor;

c2 GenericCursor;

TYPE TablePr IS TABLE OF Warehouse.Id\_product%TYPE;

Table\_product TablePr;

TYPE TableDish IS TABLE OF Card\_dish.Id\_dish%TYPE;

Table\_dish TableDish;

BEGIN

UPDATE Dishes

SET Status = 'В наличии';

SELECT Count(\*) INTO Count\_pr

FROM Warehouse

WHERE Quantity <= 1;

IF Count\_pr !=0 THEN

OPEN c1 FOR SELECT Id\_product

FROM Warehouse

WHERE Quantity <= 1;

FETCH c1 BULK COLLECT INTO Table\_product;

CLOSE c1;

FOR i IN Table\_product.first..Table\_product.last

LOOP

SELECT Count(\*) INTO Count\_dish

FROM Card\_dish

WHERE Id\_product = Table\_product(i);

IF Count\_dish !=0 THEN

OPEN c2 FOR SELECT Id\_dish

FROM Card\_dish

WHERE Id\_product = Table\_product(i);

FETCH c2 BULK COLLECT INTO Table\_dish;

CLOSE c2;

FOR j IN Table\_dish.first..Table\_dish.last

LOOP

UPDATE Dishes

SET Status = 'Стоп-лист'

WHERE Id\_dish = Table\_dish(j);

END LOOP;

END IF;

END LOOP;

END IF;

END;

**Информационно-поисковые запросы**

* Поиск информации о продуктах, входящих в состав блюда:

SELECT Id\_product

FROM Card\_dish

WHERE Id\_dish = @id\_dish

где *Id\_product –* имя поля таблицы, в котором хранится информация о номере продукта;

*Id\_dish –* имя поля таблицы, в котором хранится информация о номере блюда;

*@id\_ dish –* условное обозначение подставляемого значения из формы приложения;

* Поиск информации о сотрудниках, ответственных за оформление заказа:

SELECT FIO

FROM Employees

WHERE Type\_spec = (SELECT Id\_spec

FROM types\_spec

WHERE Name\_spec = 'Официант');

* Поиск информации о блюдах, ожидающих приготовления:

SELECT Id\_dish

FROM Info\_dishes

WHERE Id\_custom = @id\_custom AND Status = 'Ожидает приготовления'

где *Id\_dish –* имя поля таблицы, в котором хранится информация о номере блюда;

*Id\_custom –* имя поля таблицы, в котором хранится информация о номере заказа;

*Status –* имя поля таблицы, в котором хранится информация о статусе блюда;

*@id\_custom –* условное обозначение подставляемого значения из формы приложения;

**Информационно-агрегативные запросы**

* Определить количество заказов, находящихся на стадии оформления:

SELECT Count(\*)

FROM Customs

WHERE Status = 'Оформляется'

* Определить количество актуальных акций, действующих на блюдо:

SELECT COUNT(\*) INTO Count\_d

FROM Stocks

WHERE Date\_start < sysdate AND Date\_end > sysdate AND Dish = @Id\_d;

где *Count\_d –* переменная, хранящая количество акций;

*Status* – имя поля таблицы, в котором хранится информация о статусе акции;

*@Id\_d -* условное обозначение подставляемого значения из формы приложения;

* Определить номер последнего оформленного заказа:

SELECT MAX(Id\_custom)

FROM Customs;

**Вывод:** в ходе выполнения лабораторной работы были получены навыки реализации системы обмена данными с БД.