Департамент образования Вологодской области

бюджетное профессиональное образовательное учреждение Вологодской области

«ЧХТК»

Специальность 09.02.07

Информационные системы и программирование

**Отчет**

**По дисциплине PL \ SQL**

Проверяющий: Цветков К.Н.;

Разработал: Новиков Кирилл Александрович,

Студент(ка) группы 82/2021

2023

**Содержание**

[**Введение 3**](#_gjdgxs)

[**1. Практическая работа №1 4**](#_30j0zll)

[**2. Практическая работа №2 6**](#_1fob9te)

[**3. Практическая работа №3 9**](#_3znysh7)

[**4. Практическая работа №4 14**](#_2et92p0)

[**5. Практическая работа №5 17**](#_tyjcwt)

[**Вывод 18**](#_3dy6vkm)

# **Введение**

Цель данного отчета заключается в систематизации и анализе результатов выполненных практических заданий в рамках учебной практики. В процессе выполнения этих заданий применялись знания, навыки и методы, полученные в ходе обучения.

Отчет включает в себя описание поставленных задач, использованные методы решения, полученные результаты, а также выводы, основанные на опыте, полученном в процессе выполнения заданий.

Проведенный анализ практических заданий не только позволяет оценить уровень освоения материала, но также выявить области, требующие дополнительного улучшения навыков и знаний.

# **Практическая работа №1**

Описание практической работы:

Выбрать предметную область из предложенного списка. По этой предметной области необходимо продумать базу данных, а именно составить список таблиц с полями и их описать (указать название таблицы, полей, их тип данных). Не менее 5 таблиц.

Условия:

* Обязательное наличие полей, содержащих числовые значения. (например, зарплата, стоимость товара, количество).
* Обязательно наличие полей с наименованиями или ФИО.
* Обязательно наличие полей с датами (например, дата рождения, дата приема на работу, дата выпуска продукции, дата продажи)

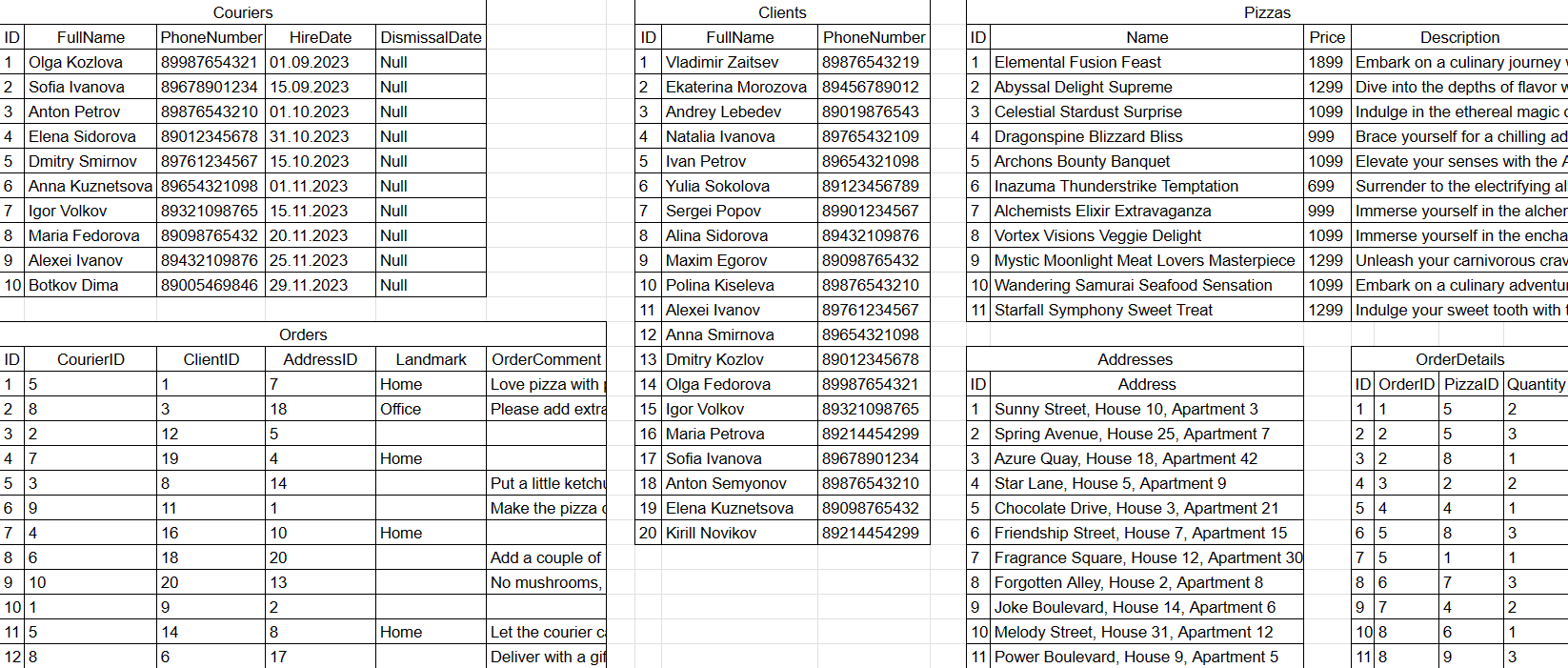
Решение задания:

Прежде чем начать создание таблиц, необходимо провести анализ выбранной предметной области. Определите, какие данные будут храниться, как они связаны между собой и какие операции будут выполняться над этими данными.

Открываем Excel и создайте новый лист, предназначенный для хранения вашей таблицы. Заполняем 7 таблиц по предметной области Пиццерия.

Разберитесь с отношениями между таблицами и проверим правильность введенных данных.

Скриншот выполненного задания:



# **Практическая работа №2**

Описание практической работы:  
 Самостоятельно изучить материалы по теме «Нормализация БД». Привести БД к 3НФ. Создать все таблицы и заполнить их данными.

Условия:  
 По возможности необходимо создать не менее 10 записей в каждой таблице.

Решение задания:

Перед тем как начать нормализацию, необходимо ознакомиться с теорией нормализации баз данных. С теоретической частью можно ознакомиться на [сайте](https://info-comp.ru/database-normalization).

Приведем все к 3 форме нормализаций данных.

Теперь мы можем создать все необходимые таблицы и заполнить их данными. Создаем таблицы в PL / SQL Developer со следующими названиями "Couriers", "Clients", "Pizzas", "Addresses", "Orders" и "OrderDetails".

Далее заполняем таблицы в соответствий с типом данных, предназначенным для столбцов.

Код, необходимый для создания приведённых выше таблиц:

*-- Создание таблицы Курьеры*

CREATE TABLE Couriers (

ID NUMBER PRIMARY KEY,

FullName VARCHAR2(255),

PhoneNumber VARCHAR2(11),

HireDate DATE,

DismissalDate Date

);

*-- Создание таблицы Клиенты*

CREATE TABLE Clients (

ID NUMBER PRIMARY KEY,

FullName VARCHAR2(255),

PhoneNumber VARCHAR2(11)

);

*-- Создание таблицы Пиццы*

CREATE TABLE Pizzas (

ID NUMBER PRIMARY KEY,

Name VARCHAR2(255),

Price NUMBER,

Description VARCHAR2(500)

);

*-- Создание таблицы Адреса*

CREATE TABLE Addresses (

ID NUMBER PRIMARY KEY,

Address VARCHAR2(500)

);

*-- Создание таблицы Заказы*

CREATE TABLE Orders (

ID NUMBER PRIMARY KEY,

CourierID NUMBER,

ClientID NUMBER,

AddressID NUMBER,

Landmark VARCHAR2(255),

OrderComment VARCHAR2(1000),

FOREIGN KEY (ClientID) REFERENCES Clients(ID),

FOREIGN KEY (CourierID) REFERENCES Couriers(ID),

FOREIGN KEY (AddressID) REFERENCES Addresses(ID)

);

*-- Создание таблицы Детали заказа*

CREATE TABLE OrderDetails (

ID NUMBER PRIMARY KEY,

OrderID NUMBER,

PizzaID NUMBER,

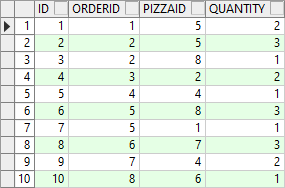
Quantity NUMBER,

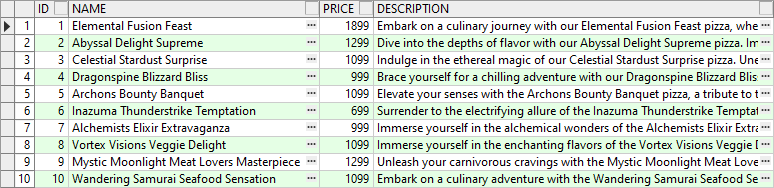
FOREIGN KEY (OrderID) REFERENCES Orders(ID),

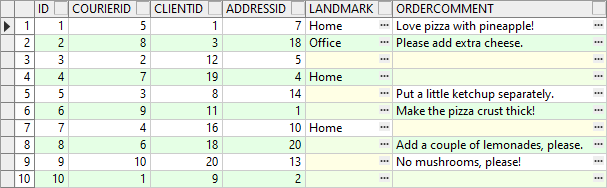
FOREIGN KEY (PizzaID) REFERENCES Pizzas(ID)

);

Скриншоты выполненной работ:







# **Практическая работа №3**

Описание практической работы:

Самостоятельно изучить тему «Соединение таблиц». Написать не менее 5 запросов, которые будут выводить данные из 2-3 таблиц, используя разные виды соединения. Для вычисляемых столбцов рекомендуется использовать синонимы.

Обновить поле таблицы (например, стоимость, зарплата, стипендия), исходя из данных.

Удалить записи в таблице, например, данные в которых не актуальны.

Решение задания:

Перед тем как начать соединение таблиц , необходимо ознакомиться с теорией соединения таблиц. С теоретической частью можно ознакомиться на [сайте](https://sky.pro/media/join-v-sql/). В рамках выполнения задания по запросам к базе данных я разработал 5 SQL-запросов, используя различные типы фильтрации и соединения таблиц. Постепенно осваивая знания, полученные в ходе лекций, я также провел операции обновления данных с использованием оператора UPDATE, внеся изменения в несколько записей таблиц. Кроме того, столкнулся с командой DELETE, удаляя ненужные данные из базы.

Код, полученный в ходе задания:

Select запросы:

**-- Курьер и их доходность в день**

SELECT

C.FullName AS CourierName,

TRUNC(SYSDATE - C.HireDate) AS WorkExperience,

SUM(P.Price \* OD.Quantity) AS TotalAmount,

ROUND(SUM(P.Price \* OD.Quantity) / TRUNC(SYSDATE - C.HireDate), 2) AS AmountPerDay

FROM

Couriers C

JOIN

Orders O ON C.ID = O.CourierID

JOIN

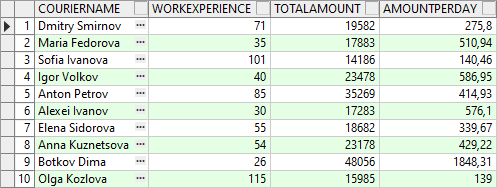
OrderDetails OD ON O.ID = OD.OrderID

JOIN

Pizzas P ON OD.PizzaID = P.ID

GROUP BY

C.ID, C.FullName, C.HireDate;



*-- Номер телефона клиента, количество заказов и их общий чек*

SELECT

C.FullName AS ClientName,

C.PhoneNumber AS ClientPhoneNumber,

COUNT(DISTINCT O.ID) AS TotalOrders,

SUM(P.Price \* OD.Quantity) AS TotalAmount

FROM

Clients C

JOIN

Orders O ON C.ID = O.ClientID

JOIN

OrderDetails OD ON O.ID = OD.OrderID

JOIN

Pizzas P ON OD.PizzaID = P.ID

GROUP BY

C.FullName, C.PhoneNumber;



**-- Сколько раз заказали каждую пиццу и сколько они стоили**

SELECT

P.ID AS PizzaID,

P.Name AS PizzaName,

COUNT(OD.ID) AS TimesOrdered,

SUM(P.Price \* OD.Quantity) AS TotalAmount

FROM

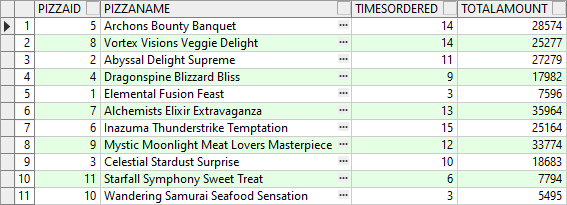
Pizzas P

JOIN

OrderDetails OD ON P.ID = OD.PizzaID

GROUP BY

P.ID, P.Name;



*-- Заказы чек которых больше 6000*

SELECT

O.ID AS OrderID,

C.FullName AS ClientName,

C2.FullName AS CourierName,

SUM(P.Price \* OD.Quantity) AS TotalAmount

FROM

Orders O

JOIN

Clients C ON O.ClientID = C.ID

JOIN

Couriers C2 ON O.CourierID = C2.ID

JOIN

OrderDetails OD ON O.ID = OD.OrderID

JOIN

Pizzas P ON OD.PizzaID = P.ID

GROUP BY

O.ID, C.FullName, C2.FullName

HAVING

SUM(P.Price \* OD.Quantity) > 6000;



*-- Количество доставок конкретным курьером конкретному клиенту*

SELECT

C.FullName AS CourierName,

Cl.FullName AS ClientName,

COUNT(\*) AS DeliveryCount

FROM

Orders O

JOIN

Couriers C ON O.CourierID = C.ID

JOIN

Clients Cl ON O.ClientID = Cl.ID

GROUP BY

C.FullName, Cl.FullName

ORDER BY

COUNT(\*) DESC;



Update запрос:

*-- Добавляет префикс, обозначающий прекращение готовки пиццы,*

*которую заказали меньше всего раз*

UPDATE Pizzas

SET Name = 'NotCooking ' || Name

WHERE ID = (

SELECT PizzaID

FROM (

SELECT PizzaID, COUNT(\*) AS order\_count

FROM OrderDetails OD

JOIN Pizzas P ON OD.PizzaID = P.ID

WHERE NOT P.Name LIKE 'NotCooking%'

GROUP BY PizzaID

ORDER BY order\_count ASC

FETCH FIRST ROW ONLY

) least\_ordered\_pizza

);



Delete запросы:

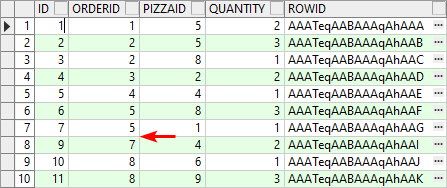
*-- Удаляет информацию про заказ номер 6*

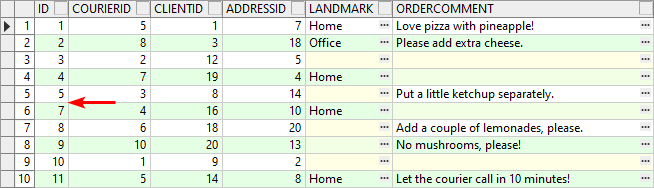
DELETE FROM OrderDetails

WHERE OrderID = 6;

DELETE FROM Orders

WHERE ID = 6;





# **Практическая работа №4**

Описание практической работы:

Написать процедуру вставки данных в таблицу.   
При успешном завершении выдавать сообщение - «запись вставлена», в случае ошибки - вывод сообщения. Использовать exception.

Написать функцию, которая будет возвращать кол-во записей из таблицы с определенным фильтром. Таблицу выбираем самостоятельно.   
У функции должен быть как минимум 1 входной параметр - фильтр.

Решение задания:

Процедура:

Создаем процедуру с именем insert\_in\_table и указываем параметры, которые будут передаваться при вызове процедуры. Параметры: ID, FullName, PhoneNumber, HireDate, и DismissalDate.

Далее вставляем данных в таблицу Couriers с использованием переданных параметров.

После использую блок exception, чтобы обработать исключения, которые могут возникнуть в процессе выполнения. Выводим "Запись вставлена" с помощью DBMS\_OUTPUT.PUT\_LINE, а если произойдет ошибка, выводится сообщение "Ошибка при вставке записи" с использованием DBMS\_OUTPUT.PUT\_LINE.

Таким образом, при вызове процедуры с нужными параметрами, она вставляет данные в таблицу customers и сообщает о результате.

Функция:

Создаем функцию с именем count\_orders\_for\_client, которая возвращает количество записей из таблицы Orders с использованием заданного фильтра. Входным параметром является p\_client\_id типа NUMBER.

Объявляете переменную v\_order\_count, которая будет использоваться для хранения результата запроса.

Дальше используем запрос SELECT COUNT(\*) для подсчета количества записей в таблице orders, где значение в столбце ClientID равно переданному параметру p\_client\_id. Результат помещается в переменную v\_order\_count.

После выполнения запроса, результат (v\_order\_count) возвращается из функции.

Если произойдет ошибка во время выполнения, то выводится сообщение об ошибке с использованием DBMS\_OUTPUT.PUT\_LINE.

Таким образом, при вызове функции с указанным фильтром, она возвращает количество записей из таблицы Orders, соответствующих заданным условиям.

Код, полученный в ходе задания:

Процедуры:

*-- Процедура вставки данных в таблицу*

create or replace procedure insert\_in\_table(

ID2 NUMBER,

FullName2 VARCHAR2,

PhoneNumber2 NUMBER,

HireDate2 DATE,

DismissalDate2 Date

)

is

begin

insert into Couriers (ID, FullName, PhoneNumber, HireDate, DismissalDate)

values (ID2, FullName2, PhoneNumber2, HireDate2, DismissalDate2);

DBMS\_OUTPUT.PUT\_LINE('Запись вставлена');

exception

when others then

DBMS\_OUTPUT.PUT\_LINE('Ошибка при вставке записи: ' || SQLERRM);

end insert\_in\_table;

Функции:

*-- Функция. которая возвращает количество записей*

CREATE OR REPLACE FUNCTION count\_orders\_for\_client(

p\_client\_id NUMBER

) RETURN NUMBER

IS

v\_order\_count NUMBER;

BEGIN

SELECT COUNT(\*)

INTO v\_order\_count

FROM Orders

WHERE ClientID = p\_client\_id;

RETURN v\_order\_count;

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE(SQLERRM);

RETURN NULL;

END count\_orders\_for\_client;

# 

# **Практическая работа №5**

Описание практической работы

Создаем процедуру с именем RandomValuesInCourier и в ней определяем два массива: NameArray для имен и SurnameArray для фамилий. Каждый массив имеет максимальную длину 7 и содержит строки длиной до 50 символов. После инициализируем массивы имен и фамилий соответственно.

Создаем переменную v\_id для хранения количества courier\_id в таблице courier.

Далее вставлям новые записи в таблицу courier, используя случайные значения для имени, фамилии и номера.

Код, полученный в ходе задания:

*-- Заполнение таблицы случайными значениями*

CREATE OR REPLACE PROCEDURE random\_valuesIn\_clients

IS

TYPE NameArray IS VARRAY(7) OF VARCHAR2(50);

TYPE SurnameArray IS VARRAY(7) OF VARCHAR2(50);

fn NameArray := NameArray('Scarlett', 'Lucas', 'Isabella', 'Grace', 'Emma', 'Elijah', 'Sophia');

lkn SurnameArray := SurnameArray('Johnson', 'Smith', 'Anderson', 'Taylor', 'Miller', 'Clark', 'Robinson');

v\_id NUMBER;

v\_full\_name VARCHAR2(255);

v\_phone\_number VARCHAR2(11);

BEGIN

SELECT COUNT(ID) INTO v\_id FROM Clients;

v\_full\_name := fn(DBMS\_RANDOM.VALUE(1, 7)) || ' ' || lkn(DBMS\_RANDOM.VALUE(1, 7));

v\_phone\_number := TO\_CHAR(TRUNC(8 + DBMS\_RANDOM.VALUE(1, 999)), 'FM00000000000');

INSERT INTO Clients (ID, FullName, PhoneNumber)

VALUES (v\_id + 1, v\_full\_name, v\_phone\_number);

END random\_valuesIn\_client

# **Вывод**

В ходе выполнения задач по PL/SQL я накопил ценный опыт и значительно усовершенствовал свои навыки работы с базой данных Oracle. Особенно глубоко я погрузился в изучение операторов SELECT, UPDATE и DELETE, освоив их эффективное применение для извлечения, обновления и удаления данных. Приобретенные навыки создания процедур и функций позволяют мне гибко управлять данными и повышать эффективность обработки запросов. Этот опыт стал не только укреплением моих компетенций в области PL/SQL, но и предоставил ценные инструменты для более продуктивной работы с базами данных в целом.