# [6-3 : Defining Data Definition Language (DDL)](https://dl.spbstu.ru/mod/resource/view.php?id=44757" \o "6-3 : Defining Data Definition Language (DDL))

**Exercise 1: Creating Tables Using Oracle Application Express**

Tasks 1. Create the DDL Statements for creating the tables for the Academic Database listed above – include NOT NULL constraints where necessary. (Other constraints will be added later) 2. Run/execute these commands in Oracle Application Express

Задачи

**1. Создайте инструкции DDL для создания таблиц для академической базы данных, перечисленных выше – при необходимости включите ограничения NOT NULL. (Другие ограничения будут добавлены позже)**

2. Запустите/исполните эти команды в Oracle Application Express

**РЕЗУЛЬТАТ**

CREATE TABLE PARENT\_INFORMATION(

ID VARCHAR2(11) NOT NULL,

PARENT\_1\_FIRST\_NAME VARCHAR2(11) NOT NULL,

PARENT\_1\_LAST\_NAME VARCHAR2(11) NOT NULL,

PARENT\_2\_FIRST\_NAME VARCHAR2(11) NOT NULL,

PARENT\_2\_LAST\_NAME VARCHAR2(11) NOT NULL

);

CREATE TABLE STUDENT\_COURSE\_DETAIL (

GRADE VARCHAR2(11) NOT NULL,

STUDENTS\_ID VARCHAR2(10) NOT NULL,

COURSES\_ID VARCHAR2(10) NOT NULL

);

CREATE TABLE DEPARTMENT (

ID VARCHAR2(11) NOT NULL,

NAME VARCHAR2(11) NOT NULL,

HEAD VARCHAR2(11) NOT NULL

);

CREATE TABLE FACULTY (

ID VARCHAR2(11) NOT NULL,

FIRST\_NAME VARCHAR2(11) NOT NULL,

LAST\_NAME VARCHAR2(11) NOT NULL,

EMAIL VARCHAR2(11) NOT NULL

);

CREATE TABLE STUDENTS (

ID VARCHAR2(11) NOT NULL,

FIRST\_NAME VARCHAR2(11) NOT NULL,

LAST\_NAME VARCHAR2(11) NOT NULL,

REGISTRATION\_YEAR NUMBER(4) NOT NULL,

EMAIL VARCHAR2(11) NOT NULL

);

CREATE TABLE COURSES (

ID VARCHAR2(11) NOT NULL,

NAME VARCHAR2(11) NOT NULL

);

CREATE TABLE ON\_LINE (

LOGIN\_ID VARCHAR2(11) NOT NULL,

PASSWORD VARCHAR2(11) NOT NULL

);

CREATE TABLE SEATED (

BUILDING VARCHAR2(11) NOT NULL,

ROOM VARCHAR2(11) NOT NULL,

DATE\_TIME DATE NOT NULL

);

CREATE TABLE FACULTY\_COURSE\_DETAIL (

CONTACTHOURS NUMBER(2) NOT NULL

);

CREATE TABLE FACULTY\_LOGIN\_DETAIL (

LOGIN\_DATETIME DATE NOT NULL

);

CREATE TABLE STUDENT\_ATTENDANCE (

NUMBER\_OF\_WORKING\_DAYS NUMBER(4) NOT NULL,

NUMBER\_OF\_DAYS\_OFF NUMBER(4) NOT NULL,

ELIGIBILITY\_FOR\_EXAM VARCHAR2(11)

);

CREATE TABLE ACADEMIC\_SESSION (

ID VARCHAR2(11) NOT NULL,

NAME VARCHAR2(11) NOT NULL

);

CREATE TABLE EXAM\_RESULTS (

GRADE VARCHAR2(11) NOT NULL

);

CREATE TABLE EXAM (

ID VARCHAR2(11) NOT NULL,

START\_DATE VARCHAR2(11) NOT NULL

);

CREATE TABLE EXAM\_TYPE (

TYPE VARCHAR2(11) NOT NULL,

NAME VARCHAR2(11) NOT NULL,

DESCRIPTION VARCHAR2(11)

);

CREATE TABLE ACADEMIC\_SESSION (

ID VARCHAR2(11) NOT NULL,

NAME VARCHAR2(11) NOT NULL

);

Изображение выглядит как текст

Автоматически созданное описание

**Exercise 2: Altering the Tables**

Tasks 1. Alter the tables in the Academic Database to define the primary key, foreign key and unique constraints. 2. Alter the table AD\_FACULTY\_LOGIN\_DETAILS and specify a default value for the column LOGIN\_DATE\_TIME of SYSDATE. 3. Set the AD\_PARENT\_INFORMATION table to a read-only status. NOTE: You can execute the INSERT / ALTER TABLE statements in Oracle Application Express in one of the two ways: Method 1: a. Open Oracle Application Express and paste the commands into the SQL Commands screen one at a time and run. Method 2: a. Open Oracle Application Express and use the same script upload method as you did with the DDL commands above.

Задачи

1. **Измените таблицы в академической базе данных, чтобы определить первичный ключ, внешний ключ и уникальные ограничения.**

Решение

-- ALTER TABLE ACADEMIC\_SESSION ADD CONSTRAINT ACADEMIC\_SESSION\_PK PRIMARY KEY ( id );

-- ALTER TABLE COURSES ADD CONSTRAINT COURSES\_PK PRIMARY KEY ( id );

ALTER TABLE COURSES ADD ACADEMIC\_SESSION\_ID VARCHAR2(10) NOT NULL;

ALTER TABLE COURSES ADD CONSTRAINT "COURSES\_ACADEMIC\_SESSION\_FK" FOREIGN KEY ( ACADEMIC\_SESSION\_ID ) REFERENCES ACADEMIC\_SESSION ( id );

ALTER TABLE COURSES ADD DEPARTMENTS\_ID VARCHAR2(10) NOT NULL;

ALTER TABLE COURSES ADD CONSTRAINT COURSES\_DEPARTMENTS\_FK FOREIGN KEY ( DEPARTMENTS\_ID ) REFERENCES DEPARTMENT ( id );

ALTER TABLE COURSES ADD ON\_LINE\_LOGIN\_ID VARCHAR2(10);

ALTER TABLE COURSES ADD CONSTRAINT "ON\_LINE\_LOGIN\_ID\_FK" FOREIGN KEY ( ON\_LINE\_LOGIN\_ID ) REFERENCES ON\_LINE ( LOGIN\_ID );

ALTER TABLE COURSES ADD SEATED\_BUILDING VARCHAR2(11);

ALTER TABLE COURSES ADD SEATED\_ROOM VARCHAR2(11);

ALTER TABLE COURSES ADD SEATED\_DATE\_TIME DATE;

ALTER TABLE COURSES ADD CONSTRAINT COURSES\_SEATED\_FK FOREIGN KEY ( SEATED\_BUILDING, SEATED\_ROOM, SEATED\_DATE\_TIME) REFERENCES SEATED ( BUILDING, ROOM, DATE\_TIME );

-- ALTER TABLE DEPARTMENT ADD CONSTRAINT DEPARTMENT\_PK PRIMARY KEY ( id );

-- ALTER TABLE EXAM ADD CONSTRAINT EXAM\_PK PRIMARY KEY ( id );

ALTER TABLE EXAM ADD EXAM\_TYPE\_TYPE VARCHAR2(15) NOT NULL;

ALTER TABLE EXAM ADD CONSTRAINT EXAM\_TYPE\_TYPE\_FK FOREIGN KEY ( EXAM\_TYPE\_TYPE ) REFERENCES EXAM\_TYPE ( TYPE );

ALTER TABLE EXAM\_RESULTS ADD COURSES\_ID VARCHAR2(10);

ALTER TABLE EXAM\_RESULTS ADD EXAM\_ID VARCHAR2(10);

ALTER TABLE EXAM\_RESULTS ADD STUDENT\_ID VARCHAR2(10);

ALTER TABLE EXAM\_RESULTS ADD CONSTRAINT EXAM\_RESULTS\_COURSE\_FK FOREIGN KEY ( COURSES\_ID ) REFERENCES COURSES ( id );

ALTER TABLE EXAM\_RESULTS ADD CONSTRAINT EXAM\_RESULTS\_EXAM\_FK FOREIGN KEY ( EXAM\_ID ) REFERENCES EXAM ( id );

ALTER TABLE EXAM\_RESULTS ADD CONSTRAINT EXAM\_RESULTS\_STUDENT\_FK FOREIGN KEY ( STUDENT\_ID ) REFERENCES STUDENTS ( id );

-- ALTER TABLE EXAM\_TYPE ADD CONSTRAINT EXAM\_TYPE\_PK PRIMARY KEY ( type );

-- ALTER TABLE FACULTY ADD CONSTRAINT FACULTY\_PK PRIMARY KEY ( id );

ALTER TABLE FACULTY ADD FACULTY\_COURSE\_DETAIL\_LDT DATE NOT NULL;

ALTER TABLE FACULTY ADD SALARY VARCHAR2(10);

ALTER TABLE FACULTY ADD INSURANCE VARCHAR2(6);

ALTER TABLE FACULTY ADD HOURLY\_RATE VARCHAR2(4);

ALTER TABLE FACULTY ADD DEPARTMENT\_ID VARCHAR2(10);

ALTER TABLE FACULTY ADD CONSTRAINT FACULTY\_DEPARTMENT\_FK FOREIGN KEY ( DEPARTMENT\_ID ) REFERENCES DEPARTMENT ( id );

-- ALTER TABLE FACULTY\_LOGIN\_DETAIL ADD CONSTRAINT FACULTY\_LOGIN\_DETAIL\_PK PRIMARY KEY ( LOGIN\_DATETIME );

ALTER TABLE FACULTY\_LOGIN\_DETAIL ADD FACULTY\_ID VARCHAR2(10) NOT NULL;

ALTER TABLE FACULTY\_LOGIN\_DETAIL ADD CONSTRAINT FACULTY\_LOGIN\_DETAIL\_FAC\_FK FOREIGN KEY ( FACULTY\_ID ) REFERENCES FACULTY ( id );

ALTER TABLE FACULTY\_COURSE\_DETAIL ADD FACULTY\_ID VARCHAR2(10) NOT NULL;

ALTER TABLE FACULTY\_COURSE\_DETAIL ADD COURSES\_ID VARCHAR2(10) NOT NULL;

ALTER TABLE FACULTY\_COURSE\_DETAIL ADD CONSTRAINT FACULTY\_COURSE\_DETAIL\_FAC\_FK FOREIGN KEY ( FACULTY\_ID ) REFERENCES FACULTY ( id );

ALTER TABLE FACULTY\_COURSE\_DETAIL ADD CONSTRAINT FACULTY\_COURSE\_DETAIL\_COUR\_FK FOREIGN KEY ( COURSES\_ID ) REFERENCES COURSES ( id );

-- ALTER TABLE ON\_LINE ADD CONSTRAINT ON\_LINE\_PK PRIMARY KEY ( LOGIN\_ID );

-- ALTER TABLE PARENT\_INFORMATION ADD CONSTRAINT PARENT\_INFORMATION\_PK PRIMARY KEY ( ID );

-- ALTER TABLE SEATED ADD CONSTRAINT SEATED\_PK PRIMARY KEY ( BUILDING, ROOM, DATE\_TIME );

-- ALTER TABLE STUDENTS ADD CONSTRAINT STUDENTS\_PK PRIMARY KEY ( ID );

ALTER TABLE STUDENTS ADD PARENT\_INFORMATION\_ID VARCHAR2(10) NOT NULL;

ALTER TABLE STUDENTS ADD CONSTRAINT STUDENTS\_PARENT\_INFORMATION\_FK FOREIGN KEY ( PARENT\_INFORMATION\_ID ) REFERENCES PARENT\_INFORMATION ( id );

ALTER TABLE STUDENT\_ATTENDANCE ADD STUDENTS\_ID VARCHAR2(10) NOT NULL;

ALTER TABLE STUDENT\_ATTENDANCE ADD ACADEMIC\_SESSION\_ID VARCHAR2(10) NOT NULL;

ALTER TABLE STUDENT\_ATTENDANCE ADD CONSTRAINT STUDENT\_ATTENDANCE\_ACADEMIC\_SESSION\_FK FOREIGN KEY ( ACADEMIC\_SESSION\_ID ) REFERENCES ACADEMIC\_SESSION ( id );

ALTER TABLE STUDENT\_ATTENDANCE ADD CONSTRAINT STUDENT\_ATTENDANCE\_STUDENTS\_FK FOREIGN KEY ( STUDENTS\_ID ) REFERENCES STUDENTS ( id );

ALTER TABLE STUDENT\_COURSE\_DETAIL ADD STUDENTS\_ID VARCHAR2(10) NOT NULL;

ALTER TABLE STUDENT\_COURSE\_DETAIL ADD COURSES\_ID VARCHAR2(10) NOT NULL;

ALTER TABLE STUDENT\_COURSE\_DETAIL ADD CONSTRAINT STUDENT\_COURSE\_DETAIL\_COURSES\_FK FOREIGN KEY ( COURSES\_ID ) REFERENCES COURSES ( id );

ALTER TABLE STUDENT\_COURSE\_DETAIL ADD CONSTRAINT STUDENT\_COURSE\_DETAIL\_STUDENTS\_FK FOREIGN KEY ( STUDENTS\_ID ) REFERENCES STUDENTS ( id );

1. **Измените таблицу AD\_FACULTY\_LOGIN\_DETAILS и укажите значение по умолчанию для столбца LOGIN\_DATE\_TIME SYSDATE.**

ALTER TABLE FACULTY\_LOGIN\_DETAIL MODIFY LOGIN\_DATETIME DEFAULT SYSDATE;

Изображение выглядит как текст, снимок экрана, монитор, экран

Автоматически созданное описание

1. **Установите для таблицы AD\_PARENT\_INFORMATION статус "только для чтения".**

Изображение выглядит как текст, снимок экрана, монитор, экран

Автоматически созданное описание

ПРИМЕЧАНИЕ: Вы можете выполнить инструкции INSERT / ALTER TABLE в Oracle Application Express одним из двух способов:

Способ 1:

Откройте Oracle Application Express и вставьте команды на экран SQL Commands по одной за раз и запустите.

Способ 2:

Откройте Oracle Application Express и используйте тот же метод загрузки скрипта, что и с помощью приведенных выше команд DDL.

**Exercise 3: Creating Composite Primary, Foreign and Unique Keys**

Tasks 1. Create the DEPT table with the following structure

The primary key for this table needs to be defined as a composite comprising of the dept\_id and loc\_id

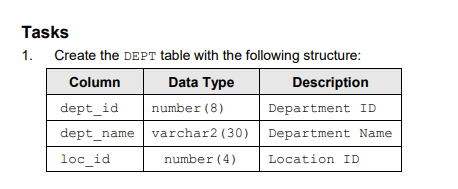
2. Create the SUPPLIERS and PRODUCTS table with the following structure

The primary key for this table needs to be defined as a composite comprising of the sup\_id and sup\_name.

The primary key for this table is product\_id. The foreign key for this table needs to be defined as a composite comprising of the sup\_id and sup\_name

The UNIQUE key for this table needs to be defined as a composite comprising of the dept\_id and dept\_name.

Задачи 1. **Создайте таблицу DEPT со следующей структурой**



Первичный ключ для этой таблицы должен быть определен как составной, состоящий из dept\_id и loc\_id

2. **Создайте таблицу ПОСТАВЩИКОВ и ПРОДУКТОВ со следующей структурой**

Первичный ключ для этой таблицы должен быть определен как составной, состоящий из sup\_id и sup\_name.

Первичным ключом для этой таблицы является product\_id. Внешний ключ для этой таблицы должен быть определен как составной, состоящий из sup\_id и sup\_name

УНИКАЛЬНЫЙ ключ для этой таблицы должен быть определен как составной, состоящий из dept\_id и dept\_name.

CREATE TABLE DEPT(

DEPT\_ID NUMBER(8),

DEPT\_NAME VARCHAR2(30),

LOC\_ID NUMBER(4),

CONSTRAINT PK\_DEPT PRIMARY KEY(DEPT\_ID,LOC\_ID));

CREATE TABLE SUPPLIERS(

SUP\_ID NUMBER(15),

SUP\_NAME VARCHAR2(30),

CONTACT\_NAME NUMBER(4),

CONSTRAINT PK\_SUPPLIERS PRIMARY KEY(SUP\_ID,SUP\_NAME));

CREATE TABLE PRODUCTS(

PRODUCT\_ID NUMBER(10),

SUP\_ID NUMBER(15),

SUP\_NAME VARCHAR2(30),

CONSTRAINT PRODUCTS\_PK PRIMARY KEY ( PRODUCT\_ID ),

CONSTRAINT PRODUCTS\_FK FOREIGN KEY ( SUP\_ID, SUP\_NAME ) REFERENCES SUPPLIERS ( SUP\_ID, SUP\_NAME ));

CREATE TABLE DEPT\_SAMPLE(

DEPT\_ID NUMBER(8),

DEPT\_NAME VARCHAR2(30),

LOC\_ID NUMBER(4),

CONSTRAINT UNQ\_DEPT\_DET UNIQUE (DEPT\_ID, DEPT\_NAME)); Изображение выглядит как текст

Автоматически созданное описание

# [6\_3\_Project](https://dl.spbstu.ru/mod/resource/view.php?id=44758)

Use DDL to build and maintain database tables (S6L3 Objective 3)

Part 1: Reading information from a script In this exercise you will use the “obl Sports.ddl” file to consolidate your knowledge of DDL. Open the “obl Sports.ddl” in a text editor. 1. How many tables have been created using the CREATE TABLE statement? 2. How many columns are created for the price history table? 3. What statement is used to enforce the constraint that the category column of the items table must have a value? 4. What is the name of the foreign key constraint between the customers and customer addresses tables? 5. What are the lowest and highest values that can be stored in the commission\_rate column for the sales\_representatives table? 6. What are the lowest and highest values that can be stored in the price column for the price\_history table? 7. What are the 3 columns that make up the primary key for the price\_history table?

Используйте DDL для создания и обслуживания таблиц базы данных (S6L3 Objective 3)

**Часть 1: Считывание информации из скрипта**

В этом упражнении вы будете использовать файл “obl Sports.ddl”, чтобы закрепить свои знания о DDL.

Откройте “obl Sports.ddl” в текстовом редакторе.

1. Сколько таблиц было создано с помощью инструкции CREATE TABLE?

**ОТВЕТ - 10**

1. Сколько столбцов создано для таблицы истории цен?

**ОТВЕТ - 6**

1. Какой оператор используется для обеспечения соблюдения ограничения, согласно которому столбец категории таблицы items должен иметь значение?

**ОТВЕТ - NOT NULL**

1. Как называется ограничение внешнего ключа между таблицами customers и customer addresses?

**ОТВЕТ - customer\_address\_customer\_fk**

1. Какие минимальные и максимальные значения могут быть сохранены в столбце commission\_rate для таблицы sales\_representatives?

**-99 и 99**

6.Каковы самые низкие и самые высокие значения, которые могут быть сохранены в столбце цена для таблицы price\_history?

**Столбец с типом данных NUMBER(7,2) может хранить числа с плавающей точкой с общей длиной 7 и 2 знаками после запятой. Минимальное значение, которое может быть сохранено в этом столбце, зависит от знаковости числа и выражается формулой:**

**Минимальное значение**

**-999999.99**

**Максимальное значение**

**999999.99**

7. Какие 3 столбца составляют первичный ключ для таблицы price\_history?

**CONSTRAINT price\_history\_pk PRIMARY KEY ( itm\_number, start\_date, start\_time )**

**Часть 2: Обновление ограничений**

Войдите в APEX и перейдите в среду команд SQL

**Part 2 : Updating Constraints**

**Modifying a column**

ALTER TABLE orders

MODIFY (odr\_date DATE DEFAULT SYSDATE);



**Adding a check constraint**

ALTER TABLE customers

ADD CONSTRAINT cust\_bal\_check

CHECK (current\_balance >= 0);



**Adding a column**

ALTER TABLE customers

ADD mobile\_number VARCHAR2(11);

**Dropping a column**

ALTER TABLE customers

DROP COLUMN mobile\_number;

# [6-4 : Defining Data Manipulation](https://dl.spbstu.ru/mod/resource/view.php?id=44833)

Exercise 1: Inserting Rows in Tables

Tasks 1. Insert rows into the tables created for the Academic Database based on the following tables

Задачи

1. **Вставьте строки в таблицы, созданные для академической базы данных на основе следующих таблиц**

-- INSERT INTO ACADEMIC\_SESSION

-- VALUES (100, 'SPRING SES');

-- INSERT INTO ACADEMIC\_SESSION

-- VALUES (200, 'FALL SES');

-- INSERT INTO ACADEMIC\_SESSION

-- VALUES (300, 'SUMMER SES');

-- INSERT INTO DEPARTMENT

-- VALUES (10, 'ACCOUNTING', 'MARK SMITH');

-- INSERT INTO DEPARTMENT

-- VALUES (20, 'BIOLOGY', 'DAVE GOLD');

-- INSERT INTO DEPARTMENT

-- VALUES (30, 'COMP SCI', 'LINDA BROWN');

-- INSERT INTO DEPARTMENT

-- VALUES (40, 'LITER', 'ANITA TAYLO');

-- ALTER TABLE PARENT\_INFORMATION READ WRITE;

-- INSERT INTO PARENT\_INFORMATION

-- VALUES (600, 'NEIL', 'SMITH', 'DORIS', 'SMITH');

-- INSERT INTO PARENT\_INFORMATION

-- VALUES (610, 'WILLIAM', 'BEN', 'NITA', 'BEN');

-- INSERT INTO PARENT\_INFORMATION

-- VALUES (620, 'SEAN', 'TAYLOR', 'RHEA', 'TAYLOR');

-- INSERT INTO PARENT\_INFORMATION

-- VALUES (630, 'DAVE', 'CARMEN', 'CATHY', 'CARMEN');

-- INSERT INTO PARENT\_INFORMATION

-- VALUES (640, 'JOHN', 'AUDRY', 'JANE', 'AUDRY');

INSERT INTO STUDENTS

VALUES (720, 'JACK', 'SMITH', 2012, 'JSTH@SH.EDU', 600);

INSERT INTO STUDENTS

VALUES (730, 'NOAH', 'AUDRY', 2012, 'NAY@SC.EDU', 640);

INSERT INTO STUDENTS

VALUES (740, 'RHONDA', 'TAYLOR', 2012, 'RTA@SCH.EU', 620);

INSERT INTO STUDENTS

VALUES (750, 'ROBERT', 'BEN', 2012, 'RN@SCH.ED', 610);

INSERT INTO STUDENTS

VALUES (760, 'JEANNE', 'BEN', 2012, 'JEN@SCH.ED', 610);

INSERT INTO STUDENTS

VALUES (770, 'MILLS', 'CARMEN', 2013, 'MAN@SCH.ED', 630);

-- INSERT INTO FACULTY

-- VALUES (800, 'JILL', 'MILLER', 'JMILL@SCHOOL.EDU', NULL, 10000, 'HEALTH', NULL, 20);

-- INSERT INTO FACULTY

-- VALUES (810, 'JAMES', 'BORG', 'JJBORG@SCHOOL.EDU', NULL, 30000, 'HEALTH, DENTAL', NULL, 10);

-- INSERT INTO FACULTY

-- VALUES (820, 'LYNN', 'BROWN', 'LBROWN@SCHOOL.EDU', NULL, NULL, NULL, 50, 30);

-- INSERT INTO FACULTY

-- VALUES (830, 'ARTHUR', 'SMITH', 'ASMITH@SCHOOL.EDU', NULL, NULL, NULL, 40, 10);

-- INSERT INTO FACULTY

-- VALUES (840, 'SALLY', 'JONES', 'SJONES@SCHOOL.EDU', NULL, 50000, 'HEALTH,DENTAL,VISION', NULL, 40);

-- INSERT INTO FACULTY\_LOGIN\_DETAIL

-- VALUES ('01-JUN-17 05.10.39.000000 PM', 800);

-- INSERT INTO FACULTY\_LOGIN\_DETAIL

-- VALUES ('01-JUN-17 05.13.15.000000 PM', 800);

-- INSERT INTO FACULTY\_LOGIN\_DETAIL

-- VALUES ('01-JUN-17 05.13.26.000000 PM', 840);

-- INSERT INTO FACULTY\_LOGIN\_DETAIL

-- VALUES ('01-JUN-17 05.13.31.000000 PM', 820);

-- INSERT INTO FACULTY\_LOGIN\_DETAIL

-- VALUES ('01-JUN-17 05.13.36.000000 PM', 830);

-- INSERT INTO EXAM\_TYPE

-- VALUES ('MCE', 'Multiple Choice Exams', 'CHOOSE MORE THAN ONE ANSWER');

-- INSERT INTO EXAM\_TYPE

-- VALUES ('TF', 'TRUE AND FALSE Exams', 'CHOOSE EITHER TRUE OR FALSE');

-- INSERT INTO EXAM\_TYPE

-- VALUES ('ESS', 'ESSAY Exams', 'WRITE PARAGRAPHS');

-- INSERT INTO EXAM\_TYPE

-- VALUES ('SA', 'SHORT ANSWER Exams', 'WRITE SHORT ANSWERS');

-- INSERT INTO EXAM\_TYPE

-- VALUES ('FIB', 'FILL IN THE BLANKS Exams', 'TYPE IN THE CORRECT ANSWER');

-- INSERT INTO EXAM

-- VALUES (500, '12-Sep-2013', 'MCE');

-- INSERT INTO EXAM

-- VALUES (510, '15-Sep-2013', 'SA');

-- INSERT INTO EXAM

-- VALUES (520, '18-Sep-2013', 'FIB');

-- INSERT INTO EXAM

-- VALUES (530, '21-Mar-2014', 'ESS');

-- INSERT INTO EXAM

-- VALUES (540, '02-Apr-2014', 'TF');

-- INSERT INTO STUDENT\_ATTENDANCE

-- VALUES (180, 11, 'Y', 730, 200);

-- INSERT INTO STUDENT\_ATTENDANCE

-- VALUES (180, 12, 'Y', 740, 300);

-- INSERT INTO STUDENT\_ATTENDANCE

-- VALUES (180, 13, 'Y', 770, 300);

-- INSERT INTO STUDENT\_ATTENDANCE

-- VALUES (180, 21, 'Y', 720, 100);

-- INSERT INTO STUDENT\_ATTENDANCE

-- VALUES (180, 14, 'Y', 750, 100);

-- INSERT INTO STUDENT\_ATTENDANCE

-- VALUES (180, 15, 'Y', 760, 200);

INSERT INTO SEATED

VALUES ('BUILDING D', 401, '09/10/2022');

INSERT INTO SEATED

VALUES ('BUILDING a', 101, '12/01/2022');

INSERT INTO SEATED

VALUES ('BUILDING b', 201, '02/04/2022');

INSERT INTO SEATED

VALUES ('BUILDING c', 301, '05/07/2022');

INSERT INTO ON\_LINE

VALUES ('TAX123', 'PASSWORD');

INSERT INTO ON\_LINE

VALUES ('BIO123', 'PASSWORD');

INSERT INTO COURSES

VALUES (195, 'CELL BIOLOGY', 200, 20, NULL, 'BUILDING D', 401, '09/10/2022', NULL);

INSERT INTO COURSES

VALUES (190, 'PRINCIPLES OF ACCOUNTING', 100, 10, NULL, 'BUILDING a', 101, '12/01/2022', NULL);

INSERT INTO COURSES

VALUES (191, 'INTRODUCTION TO BUSINESS LAW', 100, 10, NULL, 'BUILDING b', 201, '02/04/2022', NULL);

INSERT INTO COURSES

VALUES (192, 'COST ACCOUNTING', 100, 10, NULL, 'BUILDING c', 301, '05/07/2022', NULL);

INSERT INTO COURSES

VALUES (193, 'STRATEGIC TAX PLANNING ', 100, 10, 'TAX123', NULL, NULL, NULL, 'PASSWORD');

INSERT INTO COURSES

VALUES (194, 'GENERAL BIOLOGY', 200, 20, 'BIO123', NULL, NULL, NULL, 'PASSWORD');

INSERT INTO STUDENT\_COURSE\_DETAIL

VALUES ('A', 720, 190);

INSERT INTO STUDENT\_COURSE\_DETAIL

VALUES ('A', 750, 192);

INSERT INTO STUDENT\_COURSE\_DETAIL

VALUES ('B', 760, 190);

INSERT INTO STUDENT\_COURSE\_DETAIL

VALUES ('A', 770, 194);

INSERT INTO STUDENT\_COURSE\_DETAIL

VALUES ('B', 720, 193);

INSERT INTO STUDENT\_COURSE\_DETAIL

VALUES ('C', 730, 191);

INSERT INTO STUDENT\_COURSE\_DETAIL

VALUES ('F', 740, 195);

INSERT INTO STUDENT\_COURSE\_DETAIL

VALUES ('C', 760, 192);

INSERT INTO STUDENT\_COURSE\_DETAIL

VALUES ('D', 770, 192);

INSERT INTO STUDENT\_COURSE\_DETAIL

VALUES ('F', 770, 193);

INSERT INTO FACULTY\_COURSE\_DETAIL

VALUES (3, 800, 192);

INSERT INTO FACULTY\_COURSE\_DETAIL

VALUES (4, 800, 193);

INSERT INTO FACULTY\_COURSE\_DETAIL

VALUES (5, 800, 190);

INSERT INTO FACULTY\_COURSE\_DETAIL

VALUES (3, 800, 191);

INSERT INTO FACULTY\_COURSE\_DETAIL

VALUES (4, 810, 194);

INSERT INTO FACULTY\_COURSE\_DETAIL

VALUES (5, 810, 195);

INSERT INTO EXAM\_RESULTS

VALUES (91, 190, 500, 720);

INSERT INTO EXAM\_RESULTS

VALUES (87, 195, 540, 730);

INSERT INTO EXAM\_RESULTS

VALUES (85, 194, 530, 730);

INSERT INTO EXAM\_RESULTS

VALUES (97, 195, 510, 750);

INSERT INTO EXAM\_RESULTS

VALUES (78, 191, 520, 750);

INSERT INTO EXAM\_RESULTS

VALUES (70, 192, 510, 760);

INSERT INTO EXAM\_RESULTS

VALUES (97, 193, 520, 720);

INSERT INTO EXAM\_RESULTS

VALUES (60, 192, 500, 750);

INSERT INTO EXAM\_RESULTS

VALUES (65, 192, 540, 760);

INSERT INTO EXAM\_RESULTS

VALUES (60, 191, 530, 760);

**Exercise 2: Updating Rows in the Tables**

Tasks 1. Alter the AD\_FACULTY\_LOGIN\_DETAILS table to add a field called DETAILS make it a VARCHAR2(50) character field – it can have null values. 2. Update at least 2 records in the DETAILS column in the faculty login details table. \*\*Note: You will have to look up the LOGIN\_DATE\_TIME values for the records being updated since it is part of the primary key. Verify that the DETAILS column has been updated with the values:

Задачи

1. **Измените таблицу AD\_FACULTY\_LOGIN\_DETAILS, чтобы добавить поле с именем DETAILS, сделайте его полем VARCHAR2(50) символов – оно может иметь нулевые значения.**

ALTER TABLE FACULTY\_LOGIN\_DTA ADD DETAILS VARCHAR2(50);

Изображение выглядит как текст, снимок экрана, монитор, экран

Автоматически созданное описание

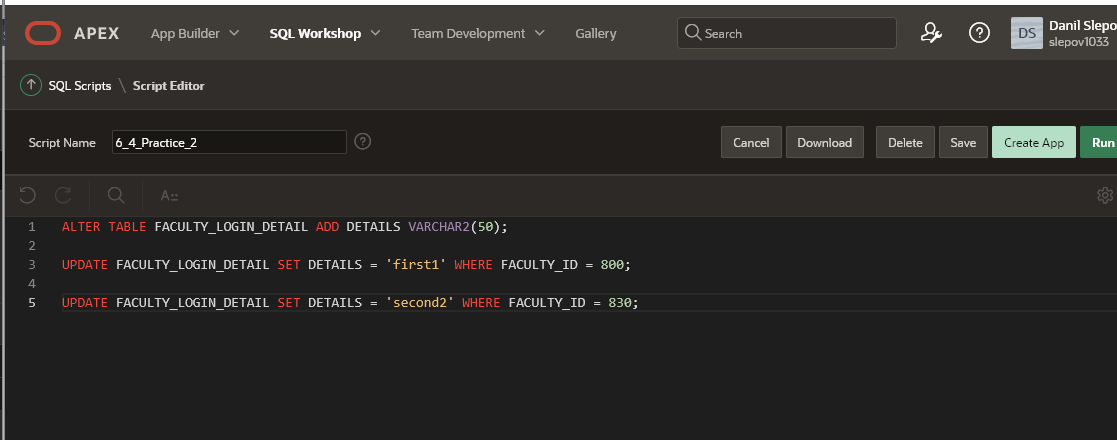
2. **Обновите по крайней мере 2 записи в столбце DETAILS в таблице учетных данных факультета**.

\*\*Примечание: Вам придется искать значения LOGIN\_DATE\_TIME для обновляемых записей, поскольку это часть первичного ключа.

ALTER TABLE FACULTY\_LOGIN\_DETAIL ADD DETAILS VARCHAR2(50);

UPDATE FACULTY\_LOGIN\_DETAIL SET DETAILS = 'first1' WHERE FACULTY\_ID = 800;

UPDATE FACULTY\_LOGIN\_DETAIL SET DETAILS = 'second2' WHERE FACULTY\_ID = 830;



Убедитесь, что столбец DETAILS был обновлен значениями

# [6\_4\_1\_Project](https://dl.spbstu.ru/mod/resource/view.php?id=44835)

Use DML operations to manage database tables (S6L4 Objective 2) In this exercise you will populate and work with the data that is stored in the database system tables. Part 1 : Running a script to populate the tables. You have to consider the order of the tables when populating them. A table that has a foreign key field cannot be populated before the related table with the primary key. 1. Use the table mapping document and list the order that you would use to populate the tables. 2. Open the “sports data.sql” and look at the order the data is being added there, does your list match? This file can be found in the Section 6 Lesson 4 interaction (sports data.zip) and must first be extracted. 3. Run the “sports data.sql” script in APEX to populate your tables 4. Check that no errors occurred when you ran the script.

Используйте операции DML для управления таблицами базы данных (S6 L4 Objective 2)

В этом упражнении вы будете заполнять данные, хранящиеся в системных таблицах базы данных, и работать с ними.

**Часть 1: Запуск скрипта для заполнения таблиц.**

Вы должны учитывать порядок расположения таблиц при их заполнении. Таблица, имеющая поле внешнего ключа, не может быть заполнена перед связанной таблицей с первичным ключом.

1. Используйте документ сопоставления таблиц и укажите порядок, который вы бы использовали для заполнения таблиц.

2. Откройте “sports data.sql” и посмотрите, в каком порядке туда добавляются данные, соответствует ли ваш список совпадение? Этот файл можно найти в разделе 6 Урок 4 взаимодействие (спорт data.zip ) и сначала должен быть извлечен.

3. Запустите скрипт “sports data.sql” в APEX для заполнения ваших таблиц

4. Убедитесь, что при запуске скрипта не возникло ошибок.

**“sports data.sql”**

Изображение выглядит как текст, монитор, черный, снимок экрана

Автоматически созданное описание

**Part 2- Inserting rows to the system**

1. Add a new team to the system

1. Add a new Customer with the following details to the system
2. This information violates the check constraint that the current balance must not be less than zero. Change the current balance to 50 and rerun the query.

Часть 2 - **Вставка строк в систему**

1. **Добавьте новую команду в систему**

INSERT INTO teams (id, name, number\_of\_players, discount)

VALUES('t004', 'Jets', 10, 5);

Изображение выглядит как текст, снимок экрана, монитор, черный

Автоматически созданное описание

1. **Добавьте в систему нового клиента со следующими реквизитами**

INSERT INTO customers (ctr\_number, email, first\_name, last\_name, phone\_number, current\_balance, loyalty\_card\_number)

VALUES('c02001', 'brianrog@hootech.com', 'Brian', 'Rogers', '01654564898', -5, 'lc4587');

Изображение выглядит как текст

Автоматически созданное описание

1. Эта информация нарушает ограничение проверки, согласно которому текущий баланс не должен быть меньше нуля. Измените текущий баланс на 50 и повторите запрос.

**не выполнено условие положительности баланса**

Изображение выглядит как текст, снимок экрана, монитор, экран

Автоматически созданное описание

# [6\_4\_2\_Project](https://dl.spbstu.ru/mod/resource/view.php?id=44836" \o "6_4_2_Project)

Part 1- Updating rows to the system 1. Run the following query to view the content of the price\_history table:

**Часть 1 - Обновление строк в системе**

1. Запустите следующий запрос, чтобы просмотреть содержимое таблицы price\_history:

SELECT start\_date, TO\_CHAR (start\_time, 'HH24:MI:SS'), price, end\_date, TO\_CHAR (end\_time, 'HH24:MI') FROM price\_history;

Изображение выглядит как текст, снимок экрана, монитор, экран

Автоматически созданное описание

1. Obl собирается обновить цену premium bat, поэтому вам нужно будет написать запрос, который закроет текущую цену, добавив значения системной даты в поля end\_date и end\_time. Чтобы выполнить этот запрос, вам нужно будет как сопоставить номер товара, так и указать, что конечная дата равна нулю. Это гарантирует, что вы обновляете последнюю цену.

UPDATE price\_history  
SET end\_date = SYSDATE, end\_time = SYSDATE  
WHERE itm\_number = 'im01101045' AND end\_date is null;

Изображение выглядит как текст, снимок экрана, монитор, экран

Автоматически созданное описание

1. Повторно запустите инструкцию select в таблице price\_history, чтобы убедиться, что инструкция была выполнена.

SELECT start\_date, TO\_CHAR (start\_time, 'HH24:MI:SS'), price, end\_date, TO\_CHAR (end\_time, 'HH24:MI') FROM price\_history;

Изображение выглядит как текст, снимок экрана, монитор, стена

Автоматически созданное описание

1. **Вставьте новую строку, в которой будут использоваться текущие дата и время, чтобы установить новую цену premium bat равной 99,99.**

INSERT INTO price\_history (start\_date, start\_time, price, itm\_number) VALUES(SYSDATE, SYSDATE, 99.99, 'im01101048');

Изображение выглядит как текст, снимок экрана, монитор, черный

Автоматически созданное описание

1. Повторно запустите инструкцию select в таблице price\_history, чтобы убедиться, что инструкция была выполнена.

SELECT start\_date, TO\_CHAR (start\_time, 'HH24:MI:SS'), price, end\_date, TO\_CHAR (end\_time, 'HH24:MI') FROM price\_history;

Изображение выглядит как текст, снимок экрана, монитор, черный

Автоматически созданное описание

**Part 2: Deleting rows from the system**

1. Bob Thornberry has contacted Obl to ask that the 83 Barrhill Drive address be removed from the system as he can longer receive parcels at this address. Write a SQL statement that will remove this address from the system. 2. Run a select statement on the customers\_addresses table to ensure that the statement has been executed

Часть 2: Удаление строк из системы

1. Боб Торнберри связался с Obl, чтобы попросить удалить адрес Barrhill Drive, 83 из системы, поскольку он больше не может получать посылки по этому адресу. **Напишите инструкцию SQL, которая удалит этот адрес из системы.**

DELETE FROM customers\_addresses  
WHERE id = 'ca0101';

Изображение выглядит как текст, монитор, снимок экрана, черный

Автоматически созданное описание

1. Запустите инструкцию select в таблице customers\_addresses, чтобы убедиться, что инструкция была выполнена

SELECT \* FROM customers\_addresses;

Изображение выглядит как текст, монитор, снимок экрана, экран

Автоматически созданное описание

**ЛАБОРАТОРНАЯ РАБОТА № 1 (SQL SERVER) ДЕСЯТОЙ НЕДЕЛИ КУРСА «УПРАВЛЕНИЕ ДАННЫМИ». ДОБАВЛЕНИЕ, ИЗМЕНЕНИЕ, УДАЛЕНИЕ ДАННЫХ**

**Задание. По аналогии добавьте в таблицу новую запись о книге Карповой Т.С. «Базы данных», вышедшей в издательстве «Питер» в 2001 году.**

INSERT INTO Book (BookId,Author, Title, Publisher, BookYear)

VALUES (4,'Карпова Т.С.','Базы данных','Питер',2001);

**Изображение выглядит как текст

Автоматически созданное описание**

**Задание. Используя базу из файлов к лабораторной работе, напишите запрос, добавляющий новый статус «устарела» в таблицу BookStatus.**

Use [MyLib];

GO

INSERT INTO BookStatus (StatusName, StatusID)

VALUES ('в библиотеке', 1), ('в учебном классе', 2),('выдана', 3);

INSERT INTO BookStatus (StatusName, StatusID)

VALUES ('устарела', 4);

**Изображение выглядит как текст

Автоматически созданное описание**

**Задание. Посмотрите имеющиеся данные в таблицах Book и BookStatus (например, выполнив Select \* From Book; и аналогично – для второй таблицы).**

Use [MyLib];

GO

SELECT \* FROM Book;

**Изображение выглядит как текст

Автоматически созданное описание**

Use [MyLib];

GO

SELECT \* FROM BookStatus;

**Изображение выглядит как текст

Автоматически созданное описание**

**Напишите и выполните скрипт, вносящий 3-5 строк в таблицу BookInLib, обратите внимание, чтобы не нарушались ограничения первичного и внешних ключей. Кроме того, не забудьте, что ключевое поле LibID не определялось как счетчик и его значение надо явно указывать.**

Use [MyLib];

GO

SELECT \* FROM Book;

SELECT \* FROM BookStatus;

INSERT INTO BookInLib

VALUES (1,1,1),(2,4,3),(3,4,3);

SELECT FROM BookInLib;

Изображение выглядит как стол

Автоматически созданное описание

**Задание. По аналогии с рассмотренным примером создайте временную таблицу #Book и внесите в нее записи из таблицы Book о книгах, выпущенных после 2000 года.**

Use [MyLib];

GO

CREATE TABLE Book2(

BookId int PRIMARY KEY,

Author varchar(100),

Title varchar(255) NOT NULL,

Publisher varchar(50),

BookYear smallint );

INSERT INTO Book2

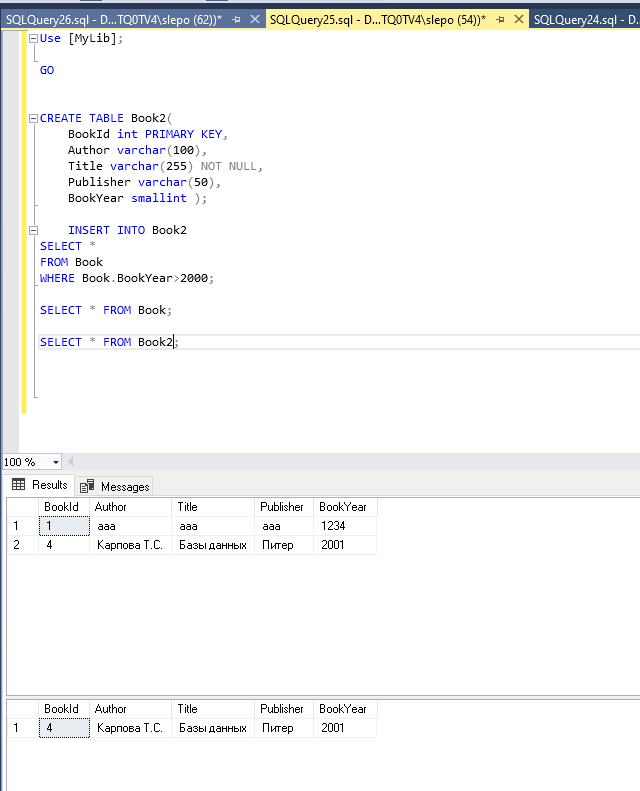
SELECT \*

FROM Book

WHERE Book.BookYear>2000;

SELECT \* FROM Book;

SELECT \* FROM Book2;

****

**Задание. Напишите два запроса, первый из которых увеличивает год издания всех книг в таблице #Book на 2, а второй изменяет в таблице #BookStatus название статуса c «устарела» на «обветшала».**

Use [MyLib];

GO

UPDATE Book2 SET BookYear=BookYear+2;

SELECT \* FROM Book2;

CREATE TABLE BookStatus2(

StatusID int PRIMARY KEY,

StatusName varchar(50) NOT NULL UNIQUE);

INSERT INTO BookStatus2 SELECT \* FROM BookStatus;

UPDATE BookStatus2

SET StatusName='обветшала'

WHERE StatusName = 'устарела';

SELECT \* FROM BookStatus2;

**Изображение выглядит как текст

Автоматически созданное описание**

**Задание. Напишите запрос, удаляющий статус «обветшала» из таблицы #BookStatus.**

DELETE

FROM BookStatus2

WHERE

StatusName='обветшала';

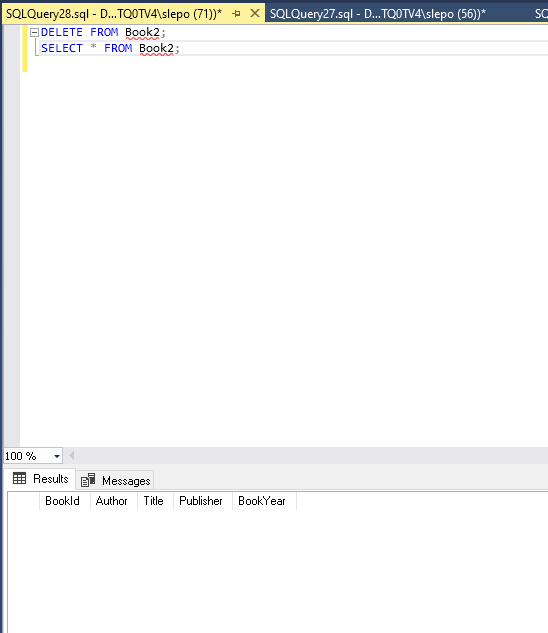
**Изображение выглядит как текст

Автоматически созданное описание**

**Задание. Проделайте этот эксперимент и проанализируйте результат. Почему не было проблем при удалении записей из таблицы #Book?**

DELETE FROM Book2;

SELECT \* FROM Book2;



Временная таблица #Book хранит в себе данные и не имеет связей с другими таблицами, поэтому не было никаких проблем с удалением записей из нее.