

# **МИНОБРНАУКИ РОССИИ**

**Федеральное государственное бюджетное образовательное  
учреждение высшего образования  
«Ярославский государственный университет им. П.Г. Демидова»**

Отчет по лабораторной работе на тему

## **Музыкальный магазин**

Дисциплина «Объектные базы данных»

Направление подготовки 02.04.02 Фундаментальная информатика и  
информационные технологии

Студентка группы ИТ-21МО

\_\_\_\_\_ Д.В. Грушевская

«06» декабря 2020 г.

Ярославль 2020 г.

## СОДЕРЖАНИЕ

Задание .....	3
Описание БД «Музыкальный магазин» .....	5
Репозиторий .....	5
Структура БД .....	5
Методы .....	9

# ЗАДАНИЕ

## ВАРИАНТ 2 (Грушевская Дарья)

**Предметная область:** «Музыкальный магазин».

**Объекты:**

- запись (идентификатор, название, время звучания, стиль);
- альбом (идентификатор, название, стоимость, количество на складе, количество проданных экземпляров);
- исполнитель (имя, псевдоним или название группы; страна).

Для каждого альбома должен быть указан список треков с указанием порядкового номера для каждой записи; одна запись может входить в разные альбомы; один трек также может быть записан сразу несколькими исполнителями. В каждом альбоме не более 30 записей.

**Методы:**

Минимальный функционал:

- 1) добавить запись (изначально указывается один исполнитель);
- 2) добавить исполнителя для записи (если указанная запись не добавлена ни в один альбом);
- 3) добавить исполнителя;
- 4) добавить альбом (изначально указывается один трек или ни одного);
- 5) добавить трек в альбом (если не продано ни одного экземпляра);
- 6) список альбомов в продаже (количество на складе больше 0);
- 7) список исполнителей;
- 8) поставка альбома (количество на складе увеличивается на указанное значение);
- 9) продать альбом (количество на складе уменьшается, проданных – увеличивается; продать можно только альбомы, в которых есть хотя бы один трек).
- 10) удалить исполнителей, у которых нет ни одной записи.

Основной функционал:

- 11) трек-лист указанного альбома с указанием суммарного времени звучания альбома;
- 12) выручка магазина (суммарная стоимость проданных альбомов по каждому в отдельности и по магазину в целом);
- 13) удалить трек с указанным номером из альбома с пересчётом остальных номеров (если не продано ни одного экземпляра альбома);
- 14) удалить исполнителя из записи (если запись не входит ни в один альбом и если этот исполнитель не единственный);
- 15) определить предпочитаемый музыкальный стиль указанного исполнителя (стиль, в котором записано большинство его треков).

16) определить предпочитаемый музыкальный стиль по каждой стране происхождения исполнителей;

17) определить авторство альбомов (для каждого альбома выводится исполнитель или список исполнителей, если все треки этого альбома записаны одним множеством исполнителей; в противном случае выводится «Коллективный сборник»).

# ОПИСАНИЕ БД «МУЗЫКАЛЬНЫЙ МАГАЗИН»

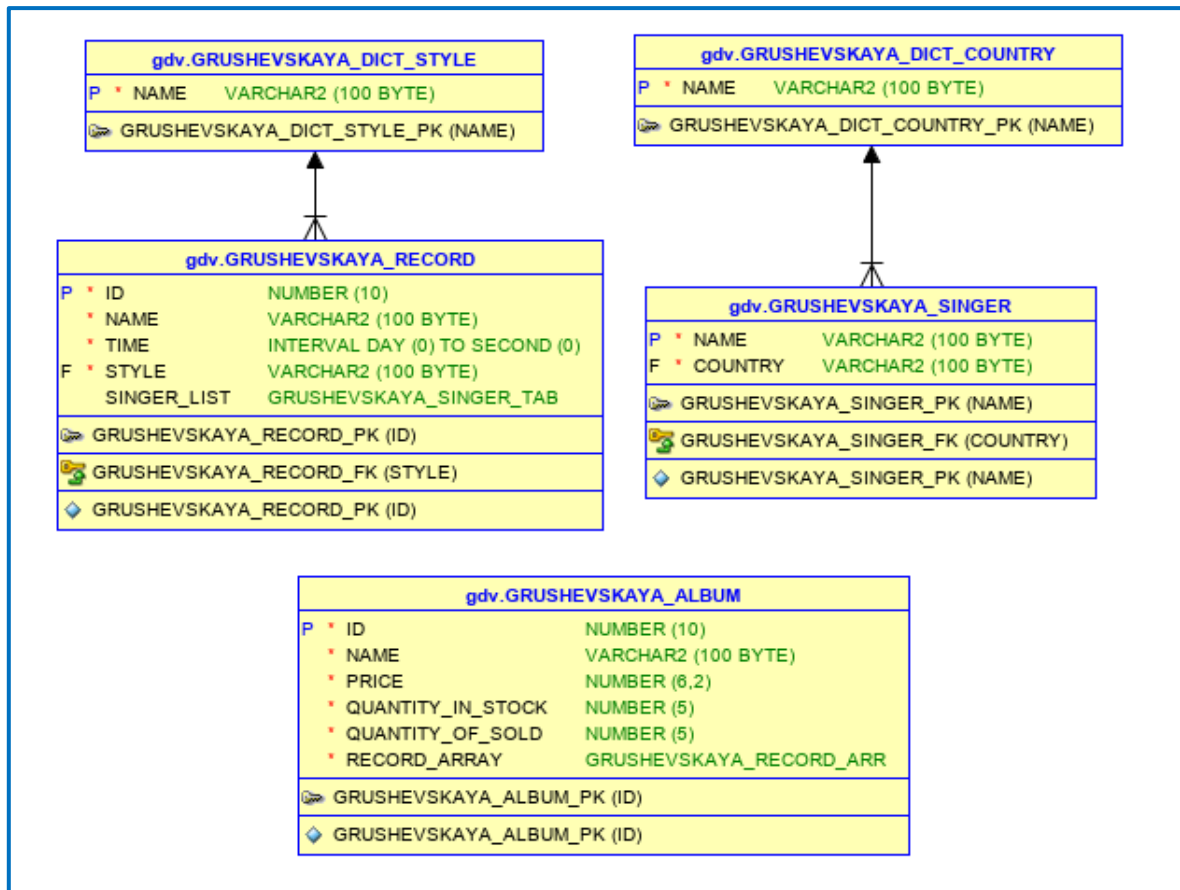


Рис. 1 Схема БД

## РЕПОЗИТОРИЙ

[https://github.com/ggeraldina/sql\\_2020](https://github.com/ggeraldina/sql_2020)

## СТРУКТУРА БД

Помимо таблиц, требующихся по заданию, были созданы дополнительно 2 таблицы – словарь стран GRUSHEVSKAYA\_DICT\_COUNTRY и словарь стилей GRUSHEVSKAYA\_DICT\_STYLE. Данные таблицы помогают отслеживать уникальность стран и стилей. В результате не будет ситуации, когда один исполнитель из РФ, а другой из России, хотя это по сути одно и то же.

```
CREATE TABLE Grushevskaya_dict_country(  
  -- название страны  
  name VARCHAR2(100 BYTE) PRIMARY KEY NOT null  
);
```

```
CREATE TABLE Grushevskaya_dict_style(
  -- название стиля
  name VARCHAR2(100 BYTE) PRIMARY KEY NOT null
);
```

В таблице GRUSHEVSKAYA\_SINGER содержится информация об исполнителях (имя, псевдоним или название группы; страна из GRUSHEVSKAYA\_DICT\_COUNTRY).

```
CREATE TABLE Grushevskaya_singer(
  -- имя, псевдоним или название группы
  name VARCHAR2(100 BYTE),
  -- страна
  country VARCHAR2(100 BYTE)
);
```

В таблице GRUSHEVSKAYA\_RECORD содержится информация о записях (идентификатор; название; время звучания; стиль из таблицы GRUSHEVSKAYA\_DICT\_STYLE; список исполнителей). Трек может быть написан одним исполнителем или несколькими. Информация об исполнителях хранится в поле исполнители (SINGER\_LIST), представляющим собой вложенную таблицу типа GRUSHEVSKAYA\_SINGER\_TAB. GRUSHEVSKAYA\_SINGER\_TAB – тип коллекции, таблица из имен исполнителей.

```
CREATE TYPE Grushevskaya_singer_tab AS TABLE OF VARCHAR2(100 BYTE);
/
CREATE TABLE Grushevskaya_record(
  -- идентификатор
  id NUMBER(10,0),
  -- название
  name VARCHAR2(100 BYTE),
  -- время звучания
  time INTERVAL DAY (0) TO SECOND (0),
  -- стиль
  style VARCHAR2(100 BYTE),
  -- список исполнителей
  singer_list Grushevskaya_singer_tab
) NESTED TABLE Singer_list
  STORE AS Grushevskaya_singer_list;
```

В таблице GRUSHEVSKAYA\_ALBUM содержится информация об альбомах (идентификатор; название; стоимость; количество на складе; количество проданных экземпляров; список записей). Информация о записях, входящих в альбом, хранится в поле RECORD\_ARRAY типа GRUSHEVSKAYA\_RECORD\_ARR. GRUSHEVSKAYA\_RECORD\_ARR – массив длины 30, состоящий из идентификаторов записей. Для хранения был выбран массив, так как в нем треки автоматически получают порядковый номер звучания.

```

CREATE TYPE Grushevskaya_record_arr AS Varray(30) OF NUMBER(10,0);
/
CREATE TABLE Grushevskaya_album (
  -- идентификатор
  id NUMBER(10, 0),
  -- название
  name VARCHAR2(100 BYTE),
  -- стоимость
  price NUMBER(6,2),
  -- количество на складе
  quantity_in_stock NUMBER(5, 0),
  -- количество проданных экземпляров
  quantity_of_sold NUMBER(5, 0),
  -- список (массив) записей
  record_array Grushevskaya_record_arr
);

```

Все поля в таблицах являются обязательными (NOT NULL) за исключением SINGER\_LIST, так как на вложенные таблицы нельзя наложить данное ограничение. Поэтому данное ограничение было реализовано с помощью триггера GRUSHEVSKAYA\_TR\_ON\_RECORDS. Об устройстве триггера будет написано чуть позднее.

На поля таблиц были наложены естественные ограничения. С помощью CHECK: цена альбома, количество альбомов на складе, количество проданных альбомов не могут быть меньше нуля. Другие ограничения были связаны с полями типа вложенная таблица или тапа массив, поэтому ограничения были реализованы с помощью триггеров.

Внешние (F) и первичные (P) ключи изображены на схеме (см. рис 1). Помимо изображенных на схеме связей дополнительно была реализована связь «многие-ко-многим» таблиц SINGER-RECORD и RECORD-ALBUM. Эти связи были реализованы с помощью триггеров.

В БД было создано всего 6 триггеров:

- GRUSHEVSKAYA\_TR\_ON\_RECORDS,
- GRUSHEVSKAYA\_TR\_ON\_SINGERS\_DEL,
- GRUSHEVSKAYA\_TR\_ON\_SINGERS\_UDP,
- GRUSHEVSKAYA\_TR\_ON\_ALBUM,
- GRUSHEVSKAYA\_TR\_ON\_RECORD\_DEL,
- GRUSHEVSKAYA\_TR\_ON\_RECORD\_UDP.

GRUSHEVSKAYA\_TR\_ON\_RECORDS – триггер уровня записи для таблицы GRUSHEVSKAYA\_RECORD, срабатывающий до вставки или обновления строки.

1) Проверяет поле SINGER\_LIST на NULL – имитирует работу ограничения NOT NULL для поля типа вложенная таблица.

2) Удаляет NULL значения из SINGER\_LIST.

3) Проверяет на пустоту SINGER\_LIST. Список исполнителей не должен быть пуст. Хотя бы один исполнитель должен быть.

4) В случае обновления исполнителей записи проверяет: не содержится ли данная запись в одном из альбомов. Исполнителей нельзя добавлять, удалять, если запись присутствует в одном из альбомов.

5) Частично имитирует работу внешнего ключа для SINGER-RECORD. Если подмножество исполнителей не соответствует таблице исполнителей, то отменяет вставку или "откатывает" обновление до прежней версии полей.

GRUSHEVSKAYA\_TR\_ON\_SINGERS\_DEL – триггер уровня записи для таблицы GRUSHEVSKAYA\_SINGER, срабатывающий до удаления строки.

1) Частично имитирует работу внешнего ключа для SINGER-RECORD. Перед удалением исполнителя проверяет: нет ли у исполнителя записей. Если есть, то удалять исполнителя нельзя.

GRUSHEVSKAYA\_TR\_ON\_SINGERS\_UDP – составной триггер уровня команды для таблицы GRUSHEVSKAYA\_SINGER, срабатывающий при обновлении поля NAME.

1) Частично имитирует работу внешнего ключа для SINGER-RECORD. После обновления имени исполнителя обновляет его имя для всех записей:

- В блоке AFTER уровня записи сохраняет значения изменяющихся имен в ассоциативном массиве, определяемом как глобальная переменная составного триггера.
- В блоке AFTER уровня команды перебираются все списки исполнителей в треках и вносятся все необходимые изменения с помощью сохранённого ассоциативного массива и булевого флага.

GRUSHEVSKAYA\_TR\_ON\_ALBUM – триггер уровня записи для таблицы GRUSHEVSKAYA\_ALBUM, срабатывающий до вставки или обновления строки.

1) Удаляет дубликаты из массива с помощью ассоциативного массива.

2) Проверяет был ли продан альбом. Если да, то обновлять, добавлять, удалять новые треки нельзя. Случаи на равенство старых, новых значений NULL-ю пришлось разграничить, так как возникало исключение.

3) Частично имитирует работу внешнего ключа для RECORD-ALBUM. Если подмножество записей не соответствует таблице записей, то отменяет вставку или "откатывает" обновление до прежней версии полей.



GRUSHEVSKAYA\_TR\_ON\_RECORD\_DEL – триггер уровня записи для таблицы GRUSHEVSKAYA\_RECORD, срабатывающий до удаления строки.

1) Частично имитирует работу внешнего ключа для RECORD-ALBUM. Перед удалением трека проверяет: нет ли его в одном из альбомов. Если есть, то удалять запись нельзя.

GRUSHEVSKAYA\_TR\_ON\_RECORD\_UDP – составной триггер уровня команды для таблицы GRUSHEVSKAYA\_RECORD, срабатывающий при обновлении поля ID.

1) Частично имитирует работу внешнего ключа для RECORD-ALBUM. После обновления идентификатора записи обновляет его идентификатор для всех альбомов:

- В блоке AFTER уровня записи сохраняет значения изменяющихся идентификаторов в ассоциативном массиве, определяемом как глобальная переменная составного триггера.
- В блоке AFTER уровня команды перебираются все списки записей в альбомах и вносятся все необходимые изменения с помощью сохранённого ассоциативного массива и булевого флага.

Все триггеры при нарушении какого-либо из условий выбрасывают исключения. Исключения перечислены в отдельном пакете GRUSHEVSKAYA\_EXCEPTIONS.

Для того чтобы не вводить руками ID записей и альбомов, были созданы две последовательности (SEQUENCE). GRUSHEVSKAYA\_NUM\_RECORD и GRUSHEVSKAYA\_NUM\_ALBUM соответственно.

## Методы

Все методы реализованы в пакете GRUSHEVSKAYA\_PACKAGE.

Методы и параметры прокомментированы. Что есть что можно понять из комментариев. Все методы являются процедурами и выводят сообщения на экран. Ниже приведена соответствующая часть скрипта.

Дополнительно были реализованы общедоступные методы добавления стран и стилей, а так же приватный метод PRINT\_MSG\_EX, печатающий код и сообщение исключения. В идеале вызываться не должен.

```
-- Пакет grushevskaya_package с реализованным функционалом
CREATE OR REPLACE
PACKAGE grushevskaya_package AS
-- Добавить страну в словарь.
```

```

PROCEDURE add_in_dict_country(
    -- Название страны
    name VARCHAR2
);
-- Добавить стиль в словарь.
PROCEDURE add_in_dict_style(
    -- Название стиля
    name VARCHAR2
);

-- Минимальный функционал

-- 1) Добавить запись (изначально указывается один исполнитель).
PROCEDURE add_record(
    -- Название
    name VARCHAR2,
    -- Количество часов звучания
    hours NUMBER,
    -- Количество минут звучания
    minutes NUMBER,
    -- Количество секунд звучания
    seconds NUMBER,
    -- Стиль из словаря
    style VARCHAR2,
    -- Имя исполнителя
    singer VARCHAR2
);
-- 2) Добавить исполнителя для записи
-- (если указанная запись не добавлена ни в один альбом
-- - Условие проверяется на уровне триггера).
PROCEDURE add_singer_in_record(
    -- id записи
    record_id NUMBER,
    -- Имя исполнителя
    singer_name VARCHAR2
);
-- 3) Добавить исполнителя.
PROCEDURE add_singer(
    -- Имя (ФИО)
    name VARCHAR2,
    -- Страна из словаря
    country VARCHAR2
);
-- 4) Добавить альбом (изначально указывается один трек или ни одного).
-- Реализация для добавления альбома с одной записью.
PROCEDURE add_album(
    -- Название
    name VARCHAR2,
    -- Цена (>= 0)
    price NUMBER,
    -- Количество на складе (>= 0)
    quantity_in_stock NUMBER,
    -- id добавляемой записи
    record_id NUMBER

```

```

);
-- 4) Добавить альбом (изначально указывается один трек или ни одного).
-- Реализация для добавления альбома без записей.
PROCEDURE add_album(
    -- Название
    name VARCHAR2,
    -- Цена (>= 0)
    price NUMBER,
    -- Количество на складе (>= 0)
    quantity_in_stock NUMBER
);
-- 5) Добавить трек в альбом
-- (если не продано ни одного экземпляра
-- - Условие проверяется на уровне триггера).
PROCEDURE add_record_in_album(
    -- id альбома
    album_id NUMBER,
    -- id добавляемой записи
    record_id NUMBER
);
-- 6) Список альбомов в продаже (количество на складе больше 0).
PROCEDURE print_albums_in_stock;
-- 7) Список исполнителей.
PROCEDURE print_singers;
-- 8) Поставка альбома
-- (количество на складе увеличивается на указанное значение).
PROCEDURE add_albums_in_stock(
    -- id альбома
    album_id NUMBER,
    -- Количество
    quantity NUMBER
);
-- 9) Продать альбом
-- (количество на складе уменьшается, проданных – увеличивается;
-- продать можно только альбомы, в которых есть хотя бы один трек
-- - Условие проверяется в самой функции).
PROCEDURE sell_albums(
    -- id альбома
    album_id NUMBER,
    -- Количество
    quantity NUMBER
);
-- 10) Удалить исполнителей, у которых нет ни одной записи.
PROCEDURE delete_singers_without_records;

-- Основной функционал

-- 11) Трек-лист указанного альбома
-- с указанием суммарного времени звучания альбома.
PROCEDURE print_album_records(album_id NUMBER);
-- 12) Выручка магазина
-- (суммарная стоимость проданных альбомов
-- по каждому в отдельности
-- и по магазину в целом).

```

```

PROCEDURE print_income;
-- 13) Удалить трек с указанным номером из альбома
-- с пересчётом остальных номеров
-- (если не продано ни одного экземпляра альбома
-- - Условие проверяется на уровне триггера).
PROCEDURE delete_record_from_album(
    -- id альбома
    album_id NUMBER,
    -- Номер звучания записи в альбоме
    record_number NUMBER
);
-- 14) Удалить исполнителя из записи
-- (если запись не входит ни в один альбом
-- и если этот исполнитель не единственный
-- - Условия проверяются на уровне триггера).
PROCEDURE delete_singer_from_record(
    -- id записи
    record_id NUMBER,
    -- Имя исполнителя
    singer_name VARCHAR2
);
-- 15) Определить предпочитаемый музыкальный стиль указанного исполнителя
-- (стиль, в котором записано большинство его треков).
PROCEDURE print_singer_style(
    -- Имя исполнителя
    singer_name VARCHAR2
);
-- 16) Определить предпочитаемый музыкальный стиль
-- по каждой стране происхождения исполнителей.
PROCEDURE print_country_style;
-- 17) Определить авторство альбомов
-- (для каждого альбома выводится
-- исполнитель или список исполнителей,
-- если все треки этого альбома записаны
-- одним множеством исполнителей;
-- в противном случае выводится «Коллективный сборник»).
PROCEDURE print_album_author;
END;
```