**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра Информационных систем**

**ДОМАШНЯЯ РАБОТА**

**по дисциплине «Методы и средства проектирования информационных систем»**

**Тема: Задание 1.1. Приемы моделирования Классификаторов**

Студенты:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Группа | ФИО | Подпись исп. | Оценка | Подпись преп. |
| 1373 | Новикова А.С. |  |  |  |
| 1373 |  |  |  |  |
| 1373 |  |  |  |  |

|  |  |  |
| --- | --- | --- |
| Преподаватель |  | Дубенецкий В.А. |

Санкт-Петербург

2024

|  |  |  |
| --- | --- | --- |
| **Содержание пояснительной записки**:  «Анализ исходных данных», «Разработка функциональных требований к подсистеме», «Разработка модели классов», «Разработка модели хранения в среде СУБД данных классификатора изделия», «Создание базы данных», «Разработка основных процедур», «Тестирование разработки», «Заключение») | | |
| Предполагаемый объем пояснительной записки:  Не менее 15 страниц. | | |
| Дата выдачи задания: 05.02.2024 | | |
| Дата сдачи отчета: 28.03.2024 | | |
| Дата защиты работы: 28.03.2024 | | |
| Студенты:   |  |  |  | | --- | --- | --- | | Группа | ФИО | Подпись исп. | | 1373 | Новикова А.С. |  | | 1373 |  |  | | 1373 |  |  | | | |
|  |  |  |
| Преподаватель |  | Дубенецкий В.А. |

СОДЕРЖАНИЕ

[1. ОПИСАНИЕ ВЫПОЛНЕННОЙ РАБОТЫ 4](#_Toc162478066)

[1.1. Анализ исходных данных 4](#_Toc162478067)

[1.2. Разработка функциональных требований к подсистеме 5](#_Toc162478068)

[1.3. Разработка модели классов 6](#_Toc162478069)

[1.4. Разработка модели хранения в среде СУБД данных классификатора изделия 6](#_Toc162478070)

[2. СОЗДАНИЕ БАЗЫ ДАННЫХ 7](#_Toc162478071)

[3. ПОДГОТОВКА СКРИПТОВ МЕТАДАННЫХ 8](#_Toc162478072)

[3.1. Разработка основных процедур 9](#_Toc162478073)

[3.2. Тестирование разработки 16](#_Toc162478074)

[ЗАКЛЮЧЕНИЕ 25](#_Toc162478075)

1. **ОПИСАНИЕ ВЫПОЛНЕННОЙ РАБОТЫ**
   1. **Анализ исходных данных**

Для разработки концептуального проекта будем использовать редактор StarUML.

В качестве исходных данных используется источник

|  |  |
| --- | --- |
| Инструменты музыкальные | https://classinform.ru/ok-eskd/kod-331.html |

Фрагмент классификатора представлен ниже:

Изделия 1

Инструменты музыкальные 2

Инструменты струнные и язычковые 3

Струнные клавишные и смычковые 4

Клавишные рояли 5

Клавишные пианино 6

Смычковые скрипки 7

Смычковые альты 8

Струнные щипковые 9

Балалайки 10

Гитары 11

Инструменты духовые 12

Амбушюрные с вентильным механизмом и цуговые 13

Корнеты 14

Трубы 15

Диаграмма классов для фрагмента предметной области представлена на рис. 1.

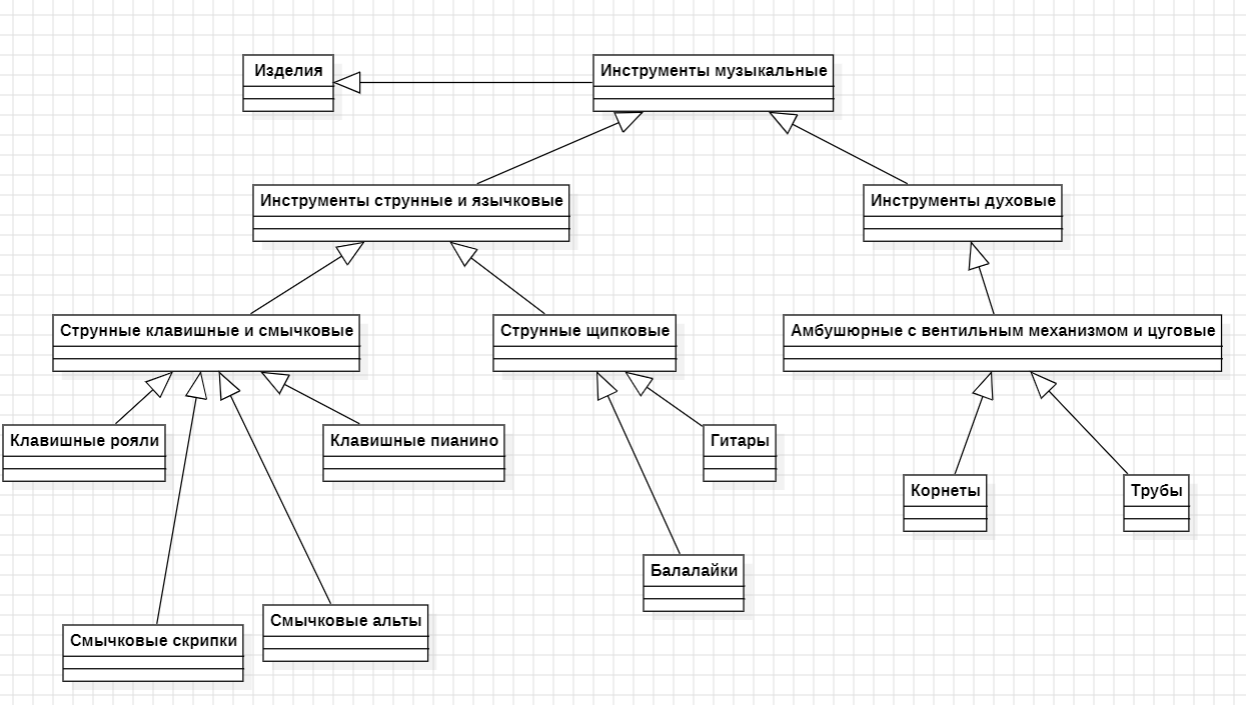


Рисунок 1. – Модель классов для объекта *Инструменты музыкальные*

* 1. **Разработка функциональных требований к подсистеме**

Диаграмма вариантов использования представлена на рис. 2.

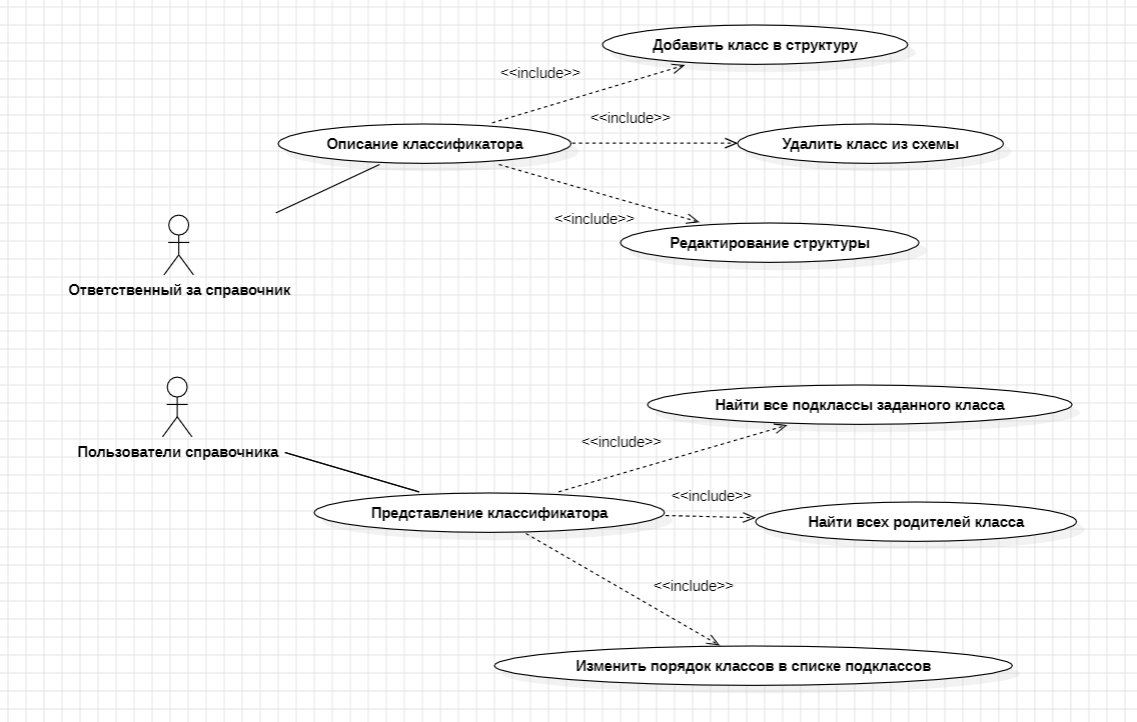


Рисунок 2. – Диаграмма вариантов использования для задачи *Работа с классификатором*

* 1. **Разработка модели классов**

Модель классов для ВИ Работа с классификатором представлена на рис. 3.

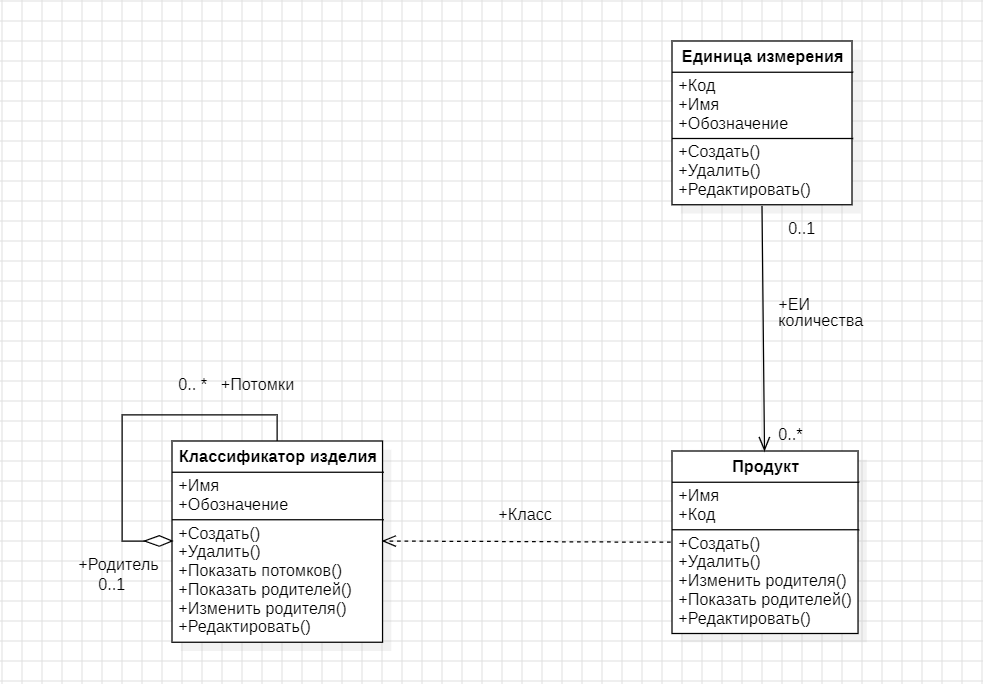


Рисунок 3. – Модель классов для ВИ *Ведение классификатора*

* 1. **Разработка модели хранения в среде СУБД данных классификатора изделия**

Первоначально опишем модель хранения в виде ERD. ER-диаграмма предтавлена на рис. 4.

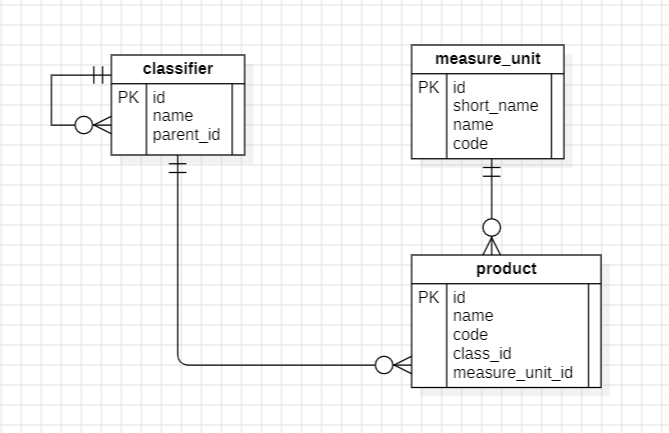


Рисунок 4. ERD для задачи Работа с классификатором изделий

1. **СОЗДАНИЕ БАЗЫ ДАННЫХ**

Локально поднимаем на сервере базу данных lab1.

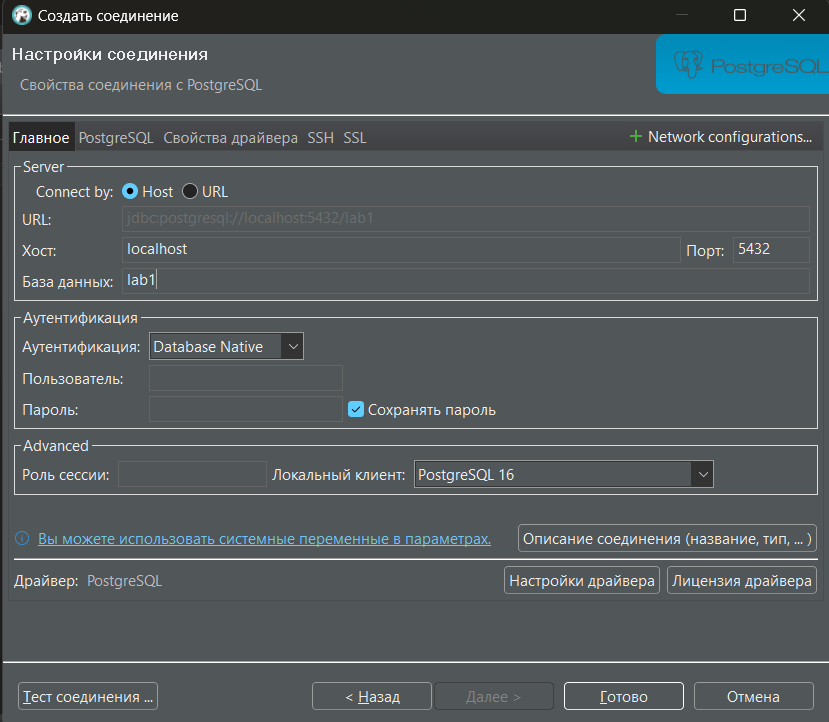


Рисунок 5. Создание базы данных в DBeaver

1. **ПОДГОТОВКА СКРИПТОВ МЕТАДАННЫХ**

/\* создание таблицы единица измерения \*/

**create** **table** measure\_unit (

id serial **not** **null** **primary** **key**,

short\_name **varchar**(5),

**name** **varchar**(50) **unique**,

code **int** **unique**

);

Спецификация таблицы *measure\_unit* в «DBeaver» приведена на рис. 6.

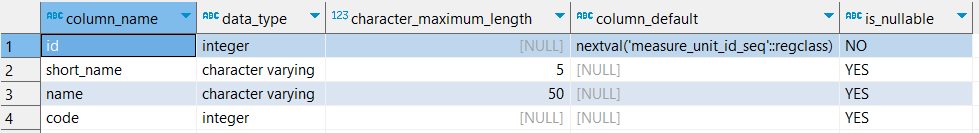


Рисунок 6. Пример просмотра таблицы *measure\_unit*

/\* создание таблицы классификатор \*/

**create** **table** classifier (

id serial **not** **null** **primary** **key**,

**name** **varchar**(96) **unique**,

parent\_id **int** **references** classifier (id) **on** **delete** **restrict**,

measure\_unit\_id **int** **references** measure\_unit **on** **delete** **restrict**

);

Спецификация таблицы *classifier* в «DBeaver» приведена на рис. 7.

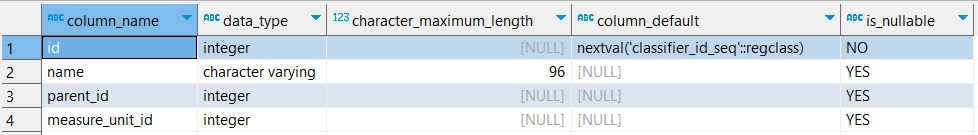


Рисунок 7. Пример просмотра таблицы *classifier*

/\* создание таблицы продукт \*/

**create** **table** product (

id serial **not** **null** **primary** **key**,

**name** **varchar** (96),

code **int** **unique**,

class\_id **int** **references** classifier (id) **on** **delete** **restrict**,

measure\_unit\_id **int** **references** measure\_unit **on** **delete** **restrict**

);

Спецификация таблицы *product* в «DBeaver» приведена на рис. 8.

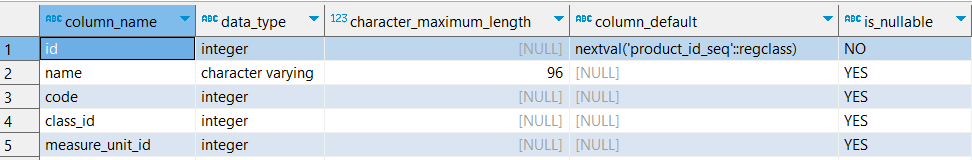
****

Рисунок 8. Пример просмотра таблицы *product*

* 1. **Разработка основных процедур**

/\* ФУНКЦИЯ: добавление единицы измерения \*/

/\* ВХОД: code\_ - международный код ЕИ, short\_name\_ - обозначение, name\_ - имя ЕИ\*/

/\* ВЫХОД: 0 - ошибка, 1 - успешно\*/

**create** **function** insert\_measure\_unit (

code\_ **int**,

short\_name\_ **varchar**(5),

name\_ **varchar**(50)

) **returns** **int** **language** plpgsql **as** **$$**

**declare**

id\_ **int**;

**begin**

**select** id **from** measure\_unit **where** code = code\_ **into** id\_;

**if** (id\_ **is** **null**) **and** ((**select** **count**(\*) **from** measure\_unit **where** **name** = name\_) = 0) **then**

**insert** **into** measure\_unit (short\_name, **name**, code)

**values** (short\_name\_, name\_, code\_);

**return** 1;

**else**

**return** 0;

**end** **if**;

**end**;

**$$**;

/\* ФУНКЦИЯ: удаление единицы измерения \*/

/\* ВХОД: id\_ - идентификатор ЕИ\*/

/\* ВЫХОД: 0 - ошибка, 1 - успешно, 2 - успешно, но в таблице classifier и/или product ячейка ЕИ стала пустой \*/

**create** **function** delete\_measure\_unit(

id\_ **int**

) **returns** **int** **language** plpgsql **as** **$$**

**declare**

product\_id\_ **int**;

**begin**

**select** id **from** product **where** measure\_unit\_id = id\_ **into** product\_id\_;

**if** (product\_id\_ **is** **null**) **then**

**delete** **from** measure\_unit **where** id = id\_;

**return** 1;

**else**

**update** product **set** measure\_unit\_id = **null** **where** measure\_unit\_id = id\_;

**delete** **from** measure\_unit **where** id = id\_;

**return** 2;

**return** 0;

**end** **if**;

**end**;

**$$**;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* ФУНКЦИЯ: создание класса\*/

/\* ВХОД: name\_ - имя класса, parent\_id\_ - идентификатор родителя\*/

/\* ВЫХОД: 0 - ошибка

\* -1 - ошибка, добавление в непустую таблицу класса без родителя ИЛИ добавление класса с родителем в пустую таблицу

\* 1 - успешно\*/

**create** **function** create\_class(

name\_ **varchar**(96),

parent\_id\_ **int**

) **returns** **int** **language** plpgsql **as** **$$**

**declare**

id\_ **int**;

**begin**

**if** ((parent\_id\_ **is** **null**) **and** ((**select** **count**(\*) **from** classifier) > 0) **or**

(parent\_id\_ **is** **not** **null**) **and** ((**select** **count**(\*) **from** classifier) = 0)) **then**

**return** -1;

**end** **if**;

**select** id **from** classifier **where** **name** = name\_ **into** id\_;

**if** ((parent\_id\_ **is** **null**) **and** ((**select** **count**(\*) **from** classifier) = 0)) **or**

((parent\_id\_ **is** **not** **null**) **and** ((**select** **count**(\*) **from** classifier) > 0) **and** (id\_ **is** **null**)) **then**

**insert** **into** classifier (**name**, parent\_id) **values** (name\_, parent\_id\_);

**return** 1;

**else**

**return** 0;

**end** **if**;

**end**;

**$$**;

/\* ФУНКЦИЯ: удаление класса \*/

/\* ВХОД: id\_ - идентификатор класса\*/

/\* ВЫХОД: 0 - ошибка, -1 - ошибка, такого класса нет, 1 - успешно\*/

**create** **function** delete\_class (

id\_ **int**

) **returns** **int** **language** plpgsql **as** **$$**

**begin**

**if** ((**select** **count**(\*) **from** classifier **where** id = id\_) = 0) **then**

**return** -1;

**end** **if**;

**if** ((**select** **count**(\*) **from** classifier **where** parent\_id = id\_) = 0) **and**

((**select** **count**(\*) **from** product **where** class\_id = id\_) = 0)

**then**

**delete** **from** classifier **where** id = id\_;

**return** 1;

**end** **if**;

**return** 0;

**end**;

**$$**;

/\* ФУНКЦИЯ: замена родителя у класса \*/

/\* ВХОД: id\_ - идентификатор класса, parent\_id\_ - идентификатор класса родителя \*/

/\* ВЫХОД: -1 - ошибка, такого класса не существует, -2 - ошибка, такого родителя не существует

\* -3 - ошибка, класс не может быть своим же родителем, 1 - успешно\*/

**create** **function** change\_class\_parent (

id\_ **int**,

parent\_id\_ **int**

) **returns** **int** **language** plpgsql **as** **$$**

**declare**

p\_id\_ **int**;

**begin**

**if** ((**select** **count**(\*) **from** classifier **where** id = id\_) = 0) **then**

**return** -1;

**end** **if**;

**if** ((**select** **count**(\*) **from** classifier **where** id = parent\_id\_) = 0) **then**

**return** -2;

**end** **if**;

p\_id\_ = parent\_id\_;

**loop**

**if** p\_id\_ = id\_

**then**

**return** 0;

**end** **if**;

**if** p\_id\_ **is** **null**

**then**

**exit**;

**end** **if**;

p\_id\_ = (**select** parent\_id **from** classifier **where** id = p\_id\_);

**end** **loop**;

**update** classifier **set** parent\_id = parent\_id\_ **where** id = id\_;

**return** 1;

**end**;

**$$**;

/\* ФУНКЦИЯ: поиск родителей класса\*/

/\* ВХОД: id\_ - идентификатор класса\*/

/\* ВЫХОД: cl\_id - идентификатор класса, cl\_name - имя класса, p\_id - идентификатор класса-родителя\*/

**create** **function** find\_class\_parents (

id\_ **int**

) **returns** **table**(cl\_id **int**, cl\_name **varchar**(96), p\_id **int**) **language** plpgsql **as** **$$**

**begin**

**return** query

**select** classifier.id, classifier.**name**, classifier.parent\_id

**from** classifier **where** id = id\_

**union**

(**with** **recursive** r **as** (

**with** t **as** (**select** parent\_id **as** parent **from** classifier **where** id = id\_)

**select** classifier.id, classifier.**name**, classifier.parent\_id

**from** classifier, t

**where** id = t.parent

**union**

**select** classifier.id, classifier.**name**, classifier.parent\_id

**from** classifier **join** r **on** classifier.id = r.parent\_id

)

**select** \* **from** r)

**order** **by** id **desc**;

**end**;

**$$**;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* ФУНКЦИЯ: создание продукта \*/

/\* ВХОД: name\_ - имя продукта, code\_ - код продукта, measure\_unit\_id\_ - идентификатор ЕИ, class\_id\_ - идентификатор класса \*/

/\* ВЫХОД: -1 - ошибка, продукт с таким кодом уже есть, -2 - ошибка, такой ЕИ не существует

\* -3 - ошибка, такого класса не существует, -4 - ошибка, продукт с таким именем уже есть

\* 1 - успешно\*/

**create** **function** create\_product(

name\_ **varchar**(96),

code\_ **int**,

measure\_unit\_id\_ **int**,

class\_id\_ **int**

) **returns** **int** **language** plpgsql **as** **$$**

**begin**

**if** ((**select** **count**(\*) **from** product **where** code = code\_) > 0) **then**

**return** -1;

**end** **if**;

**if** ((**select** **count**(\*) **from** measure\_unit **where** id = measure\_unit\_id\_) = 0) **and** (measure\_unit\_id\_ **is** **not** **null**) **then**

**return** -2;

**end** **if**;

**if** ((**select** **count**(\*) **from** classifier **where** id = class\_id\_) = 0) **then**

**return** -3;

**end** **if**;

**if** ((**select** **count**(\*) **from** product **where** **name** = name\_) > 0) **then**

**return** -4;

**end** **if**;

**insert** **into** product (**name**, code, class\_id, measure\_unit\_id)

**values** (name\_, code\_, class\_id\_, measure\_unit\_id\_);

**return** 1;

**end**;

**$$**;

/\* ФУНКЦИЯ: удаление продукта \*/

/\* ВХОД: id\_ - идентификатор продукта\*/

/\* ВЫХОД: -1 - ошибка, такого продукта нет, 1 - успешно\*/

**create** **function** delete\_product(

id\_ **int**

) **returns** **int** **language** plpgsql **as** **$$**

**begin**

**if** ((**select** **count**(\*) **from** product **where** id = id\_) = 0) **then**

**return** -1;

**end** **if**;

**delete** **from** product **where** id = id\_;

**return** 1;

**end**;

**$$**;

/\* ФУНКЦИЯ: изменение родителя у продукта \*/

/\* ВХОД: id\_ - идентификатор продукта, class\_id\_ - идентификатор класса родителя \*/

/\* ВЫХОД: -1 - ошибка, такого продукта не существует, -2 - ошибка, такого класса не существует, 1 - успешно\*/

**create** **function** change\_product\_parent(

id\_ **int**,

class\_id\_ **int**

) **returns** **int** **language** plpgsql **as** **$$**

**begin**

**if** ((**select** **count**(\*) **from** product **where** id = id\_) = 0) **then**

**return** -1;

**end** **if**;

**if** ((**select** **count**(\*) **from** classifier **where** id = class\_id\_) = 0) **then**

**return** -2;

**end** **if**;

**update** product **set** class\_id = class\_id\_ **where** id = id\_;

**return** 1;

**end**;

**$$**;

/\* ФУНКЦИЯ: поиск родителей продукта\*/

/\* ВХОД: id\_ - идентификатор продукта\*/

/\* ВЫХОД: cl\_id - идентификатор класса, cl\_name - имя класса, p\_id - идентификатор класса-родителя\*/

**create** **function** find\_product\_parents(

id\_ **int**

) **returns** **table** (cl\_id **int**, cl\_name **varchar**(96), p\_id **int**) **language** plpgsql **as** **$$**

**declare**

class\_id\_ **int**;

**begin**

**select** class\_id **from** product **where** id = id\_ **into** class\_id\_;

**return** query

**select** \* **from** find\_class\_parents(class\_id\_);

**end**;

**$$**;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\* ФУНКЦИЯ: Поиск всех подклассов класса\*/

/\* ВХОД: id\_ - идентификатор класса, mode\_ - выбор вывода по возрастанию или убыванию\*/

/\* ВЫХОД: cl\_id - идентификатор класса, cl\_name - название класса, p\_id - идентификатор класса-родителя\*/

/\* ТРЕБОВАНИЯ: 1. mode\_ должен быть равен 'desc' для вывода по убыванию, либо 'asc'/''/null для вывода по возрастанию

\* 2. Элемент должен быть в дереве \*/

/\* ЭФФЕКТЫ: 1. Для терминального элемента выдает исходный элемент

\* 2. Для элемента, которого нет в дереве, выдаёт пустую строку

\* 3. Если mode\_ введён неверно, то выдаёт пустую строку\*/

**create** **function** find\_cl\_cl (

id\_ **int**,

mode\_ **varchar**(10)

) **returns** **table** (cl\_id **int**, cl\_name **varchar**(96), p\_id **int**) **language** plpgsql **as** **$$**

**begin**

**if** mode\_ = 'desc' **then**

**return** query

**select** classifier.id, classifier.**name**, classifier.parent\_id

**from** classifier **where** id = id\_

**union**

(**with** **recursive** r **as** (

**select** classifier.id, classifier.**name**, classifier.parent\_id

**from** classifier **where** parent\_id = id\_

**union**

**select** classifier.id, classifier.**name**, classifier.parent\_id

**from** classifier **join** r **on** classifier.parent\_id = r.id

)

**select** \* **from** r)

**order** **by** id **desc**;

**else**

**if** mode\_ = 'asc' **or** mode\_ **is** **null** **or** mode\_ = '' **then**

**return** query

**select** classifier.id, classifier.**name**, classifier.parent\_id

**from** classifier **where** id = id\_

**union**

(**with** **recursive** r **as** (

**select** classifier.id, classifier.**name**, classifier.parent\_id

**from** classifier **where** parent\_id = id\_

**union**

**select** classifier.id, classifier.**name**, classifier.parent\_id

**from** classifier **join** r **on** classifier.parent\_id = r.id

)

**select** \* **from** r)

**order** **by** id;

**else**

**return** query

**select** \* **from** classifier **where** id = 0;

**end** **if**;

**end** **if**;

**end**;

**$$**;

/\* ФУНКЦИЯ: Поиск всех продуктов класса\*/

/\* ВХОД: id\_ - идентификатор класса, mode\_ - выбор вывода по возрастанию или убыванию\*/

/\* ВЫХОД: pr\_id - идентификатор продукта, pr\_name - название продукта, pr\_code - код продукта

\* cl\_id - идентификатор класса, m\_u\_id - идентификатор ЕИ \*/

/\* ТРЕБОВАНИЯ: 1. mode\_ должен быть равен 'desc' для вывода по убыванию, либо 'asc'/''/null для вывода по возрастанию

\* 2. Элемент должен быть в дереве\*/

/\* ЭФФЕКТЫ: 1. Для элемента, которого нет в дереве, выдаёт пустую строку

\* 2. Если mode\_ введён неверно, то выдаёт пустую строку\*/

**create** **function** find\_pr\_cl(

id\_ **int**,

mode\_ **varchar**(5)

) **returns** **table** (pr\_id **int**, pr\_name **varchar**(96), pr\_code **int**, cl\_id **int**, m\_u\_id **int**) **language** plpgsql **as** **$$**

**begin**

**if** mode\_ = 'desc' **then**

**return** query

**select** \* **from** product **where** class\_id = id\_

**order** **by** id **desc**;

**else**

**if** mode\_ = 'asc' **or** mode\_ **is** **null** **or** mode\_ = '' **then**

**return** query

**select** \* **from** product **where** class\_id = id\_;

**else**

**return** query

**select** \* **from** product **where** id = 0;

**end** **if**;

**end** **if**;

**end**;

**$$**;

/\* ФУНКЦИЯ: Вывод всех продуктов выбранного класса \*/

/\* ВХОД: id\_ - идентификатор класса\*/

/\* ВЫХОД: pr\_id - идентификатор продукта, pr\_name - название продукта, pr\_code - код продукта

\* m\_u\_name - обозначение ЕИ, cl\_id - индентификатор класса, cl\_name - название класса\*/

/\* ЭФФЕКТЫ: 1. Для элемента, которого нет в дереве, выдаёт пустую строку\*/

**create** **function** find\_list(

id\_ **int**

) **returns** **table** (pr\_id **int**, pr\_name **varchar**(96), pr\_code **int**, m\_u\_name **varchar**(5), cl\_id **int**, cl\_name **varchar**(96))

**language** plpgsql **as** **$$**

**begin**

**return** query

**with** t **as** (**select** \* **from** find\_cl\_cl(id\_, 'asc'))

**select** product.id, product.**name**, product.code **int**, measure\_unit.short\_name, product.class\_id, classifier.**name**

**from** product

**inner** **join** classifier **on** classifier.id = product.class\_id

**inner** **join** measure\_unit **on** product.measure\_unit\_id = measure\_unit.id

**where** product.class\_id = **any**(**select** t.cl\_id **from** t);

**end**;

**$$**;

* 1. **Тестирование разработки**

Единицы измерения берем из стандарта:

003 мм Миллиметр

006 м Метр

055 м2 Квадратный метр

166 кг Килограмм

168 т Тонна

Скрипты для заполнения справочника ЕИ для тестирования:

**select** \* **from** insert\_measure\_unit('003', 'мм', 'Миллиметр');

**select** \* **from** insert\_measure\_unit('006','м','Метр');

**select** \* **from** insert\_measure\_unit('055','м2','Квадратный метр');

**select** \* **from** insert\_measure\_unit('166','кг','Килограмм');

**select** \* **from** insert\_measure\_unit('168','т','Тонна');

Пример заполнения таблицы measure\_unit представлен на рис. 9.

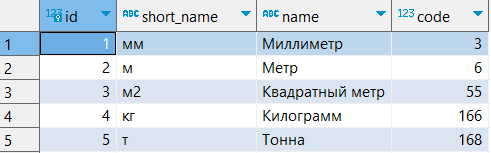


Рисунок 9. Пример заполнения таблицы measure\_unit

Процедура **insert\_measure\_unit** работает корректно.

Данные для заполнения фрагмента классификатора изделий берем из исходных данных.

Изделия 1

Инструменты музыкальные 2

Инструменты струнные и язычковые 3

Струнные клавишные и смычковые 4

Клавишные рояли 5

Клавишные пианино 6

Смычковые скрипки 7

Смычковые альты 8

Струнные щипковые 9

Балалайки 10

Гитары 11

Инструменты духовые 12

Амбушюрные с вентильным механизмом и цуговые 13

Корнеты 14

Трубы 15

Скрипты для заполнения классификатора приведены ниже:

**select** \* **from** create\_class('Изделия', **null**, **null**);

**select** \* **from** create\_class('Инструменты музыкальные', 4, 1);

**select** \* **from** create\_class('Инструменты струнные и язычковые', 4, 2);

**select** \* **from** create\_class('Струнные клавишные и смычковые', 4, 3);

**select** \* **from** create\_class('Струнные щипковые', 4, 3);

**select** \* **from** create\_class('Инструменты духовые', 4, 2);

**select** \* **from** create\_class('Амбушюрные с вентильным механизмом и цуговые', 4, 6);

Пример заполнения таблицы classifier представлен на рис. 10.

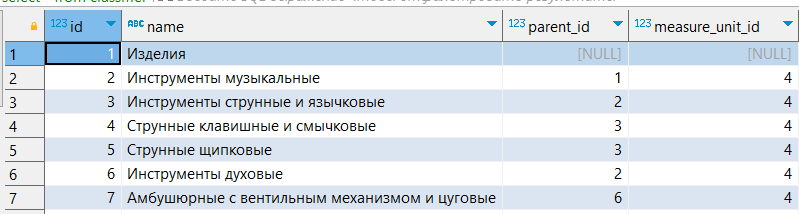


Рисунок 10. Пример заполнения таблицы classifier

**select** \* **from** create\_product('Клавишные рояли', 331111, 4, 4);

**select** \* **from** create\_product('Клавишные пианино', 331112, 4, 4);

**select** \* **from** create\_product('Смычковые скрипки', 331114, 4, 4);

**select** \* **from** create\_product('Смычковые альты', 331115, 4, 4);

**select** \* **from** create\_product('Балалайки', 331121, 4, 5);

**select** \* **from** create\_product('Гитары', 331122, 4, 5);

**select** \* **from** create\_product('Корнеты', 331211, 4, 7);

**select** \* **from** create\_product('Трубы', 331212, 4, 7);

Пример заполнения таблицы product представлен на рис. 11.

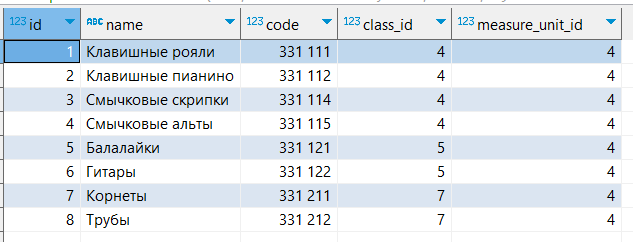


Рисунок 11. Пример заполнения таблицы product

Видно, что процедуры **create\_class** и **create\_product** работают корректно.

**Тестирование процедуры find\_cl\_cl.**

Запрос с вершины дерева.

**select** \* **from** find\_cl\_cl(1, 'asc'); /\* Изделия \*/

Ответ представлен на рис. 12.

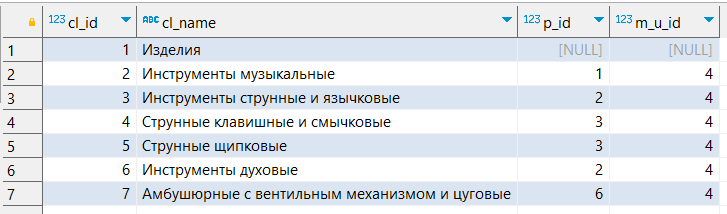


Рисунок 12. Результат поиска потомков

Запрос с узла дерева.

**select** \* **from** find\_cl\_cl(3, 'desc'); /\* Инструменты струнные и язычковые \*/

Ответ представлен на рис. 13.

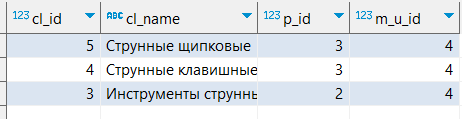


Рисунок 13. Результат поиска потомков Инструменты струнные и язычковые

Запрос с терминального узла.

**select** \* **from** find\_cl\_cl(7, 'asc'); /\* Амбушюрные с вентильным механизмом и цуговые \*/

Ответ представлен на рис. 14.

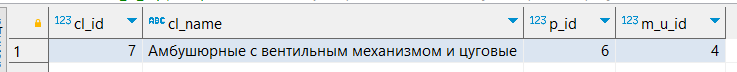


Рисунок 14. Результат поиска потомков Амбушюрные с вентильным механизмом и цуговые

**Тестирование процедуры find\_pr\_cl**

Запрос с вершины дерева.

**select** \* **from** find\_pr\_cl(1, 'asc'); /\* Изделия \*/

Ответ представлен на рис. 15.

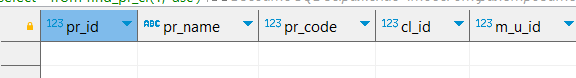


Рисунок 15. Результат поиска продуктов Изделия

Запрос с узла дерева.

**select** \* **from** find\_pr\_cl(4, 'asc'); /\* Струнные клавишные и смычковые \*/

Ответ представлен на рис. 16.

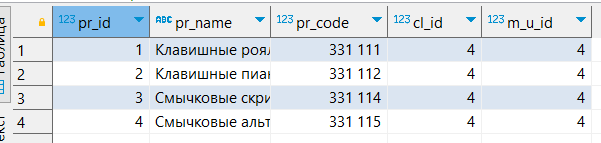


Рисунок 16. Результат поиска продуктов Струнные клавишные и смычковые

**Тестирование функции find\_list**

Запрос с вершины дерева

**select** \* **from** find\_list(1); /\* Изделия \*/

Ответ представлен на рис. 17.

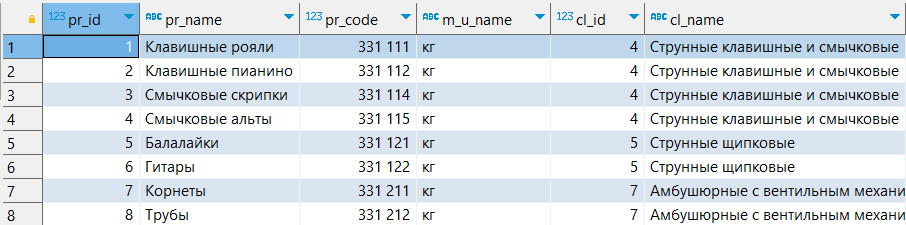


Рисунок 17. Результат поиска продуктов класса Изделия

Запрос с узла дерева

**select** \* **from** find\_list(4); /\* Струнные клавишные и смычковые \*/

Ответ представлен на рис. 18.

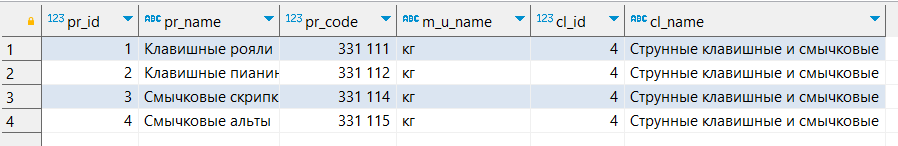


Рисунок 18. Результат поиска продуктов класса Струнные клавишные и смычковые

Процедуры find\_cl\_cl, find\_pr\_cl, find\_list работают корректно.

**Тестирование функции delete\_measure\_unit**

Запрос 1:

**select** \* **from** delete\_measure\_unit(5); /\* Тонна \*/

Ответ и изменённая таблица представлены на рис. 19-20

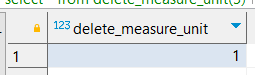


Рисунок 19. Результат удаления ЕИ Тонна

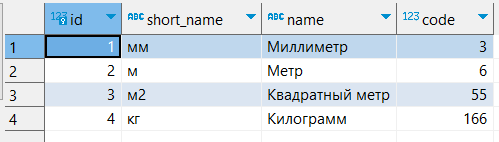


Рисунок 20. Таблица после удаления ЕИ Тонна

Запрос 2:

**select** \* **from** delete\_measure\_unit(4); /\* Килограмм \*/

Ответ и изменённая таблица представлены на рис. 21-22

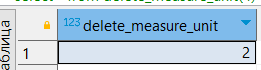


Рисунок 21. Результат удаления ЕИ Килограмм

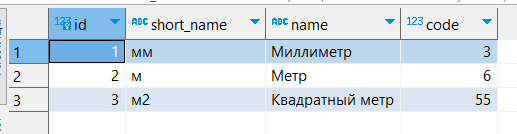


Рисунок 22. Таблица после удаления ЕИ Килограмм

После выполнения этого запроса у нас должны были измениться ячейки ЕИ в таблицах classifier и product. Таблицы после удаления ЕИ Килограмм представлены на рис. 23-24

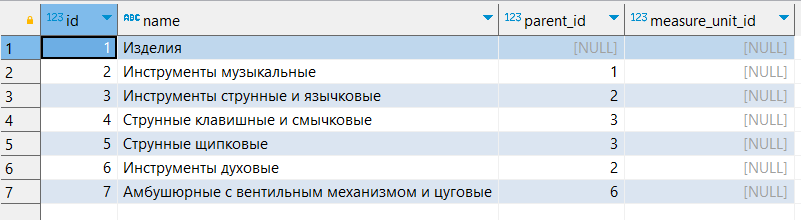


Рисунок 23. Таблица classifier после удаления ЕИ Килограмм

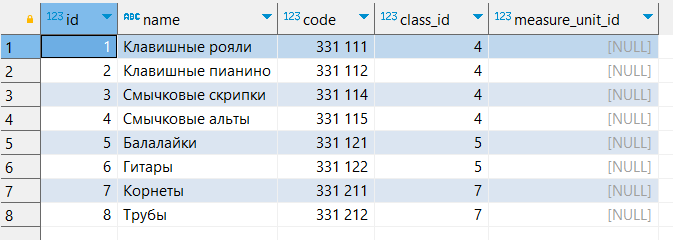


Рисунок 24. Таблица product после удаления ЕИ Килограмм

**Тестирование функции delete\_product**

Запрос 1:

select \* from delete\_product(5); /\* Балалайки \*/

Ответ и изменённая таблица представлены на рис. 25-26

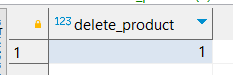


Рисунок 25. Результат удаления продукта Балалайки

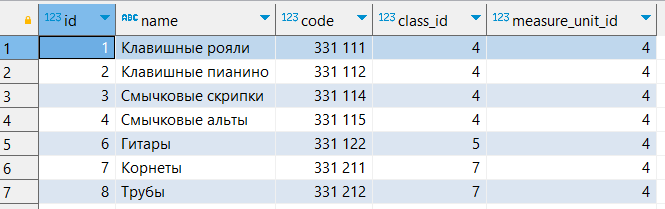


Рисунок 26. Таблица product после удаления продукта Балалайки

**Тестирование функции delete\_class**

Запрос 1.

**select** \* **from** delete\_class(2); /\* Инструменты музыкальные \*/

Ответ представлен на рис. 27.

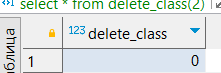


Рисунок 27. Результат удаления класса Инструменты музыкальные

Удалить класс не получилось, так как у него есть потомки.

Запрос 2.

Для этого запроса были удалены продукты Балалайки и Гитары.

**select** \* **from** delete\_class(5); /\* Струнные щипковые \*/

Ответ и изменённая таблица представлены на рис. 28-29.

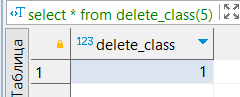


Рисунок 28. Результат удаления класса Струнные щипковые

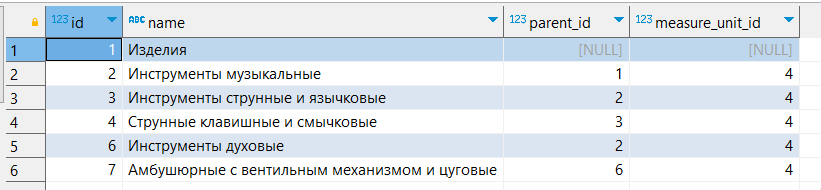


Рисунок 29. Таблица classifier после удаления класса Струнные щипковые

Класс удалить удалось, так как у него нет потомков

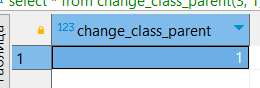
Процедуры **delete\_measure\_unit**, **delete\_product**, **delete\_class** работают корректно.

**Тестирование функции change\_class\_parent**

Запрос 1.

**select** \* **from** change\_class\_parent(3, 1); /\* Инструменты струнные и язычковые \*/

Ответ и изменённая таблица представлены на рис. 30-31



. Рисунок 30. Результат смены родителя класса Инструменты струнные и язычковые

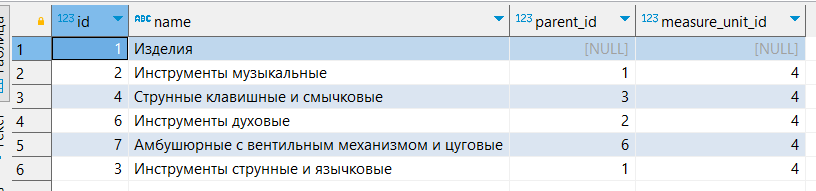


Рисунок 31. Таблица classifier после смены родителя класса Инструменты струнные и язычковые

**Тестирование функции change\_product\_parent**

Запрос 1.

**select** \* **from** change\_product\_parent(7, 3); /\* Корнеты \*/

Ответ и изменённая таблица представлены на рис. 32-33

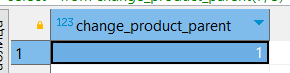


Рисунок 32. Результат смены родителя продукта Корнеты

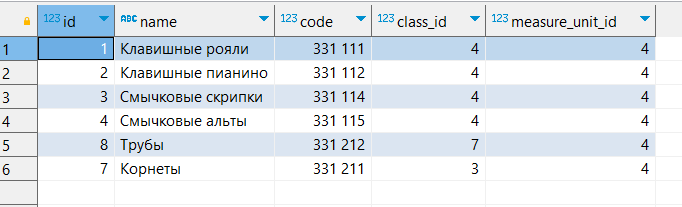


Рисунок 33. Таблица product после смены родителя продукта Корнеты

Процедуры **change\_class\_parent**, **change\_product\_parent** работают корректно.

**Тестирование функции find\_class\_parents**

Запрос 1.

**select** \* **from** find\_class\_parents(5); /\* Струнные щипковые \*/

Ответ представлен на рис. 34.

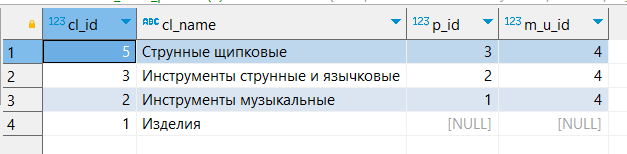


Рисунок 34. Результат поиска родителей класса Струнные щипковые

Запрос 2.

**select** \* **from** find\_class\_parents(1); /\* Изделия \*/

Ответ представлен на рис. 35.

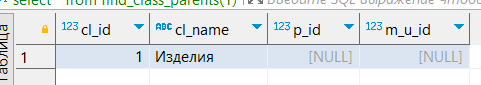


Рисунок 35. Результат поиска родителей класса Изделие

**Тестирование функции find\_product\_parents**

Запрос 1.

**select** \* **from** find\_product\_parents(1); /\* Клавишные рояли \*/

Ответ представлен на рис. 36.

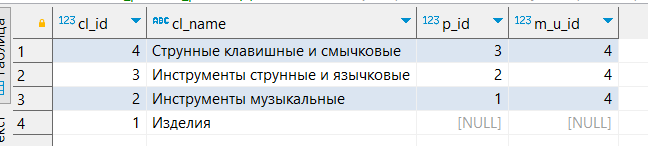


Рисунок 36. Результат поиска родителей продукта Клавишные рояли

Процедуры **find\_class\_parents**, **find\_product\_parents** работают корректно.

# ЗАКЛЮЧЕНИЕ

В ходе выполнения данной работы была создана модель предметной области «Магазин музыкальных инструментов». Были созданы необходимые функции для работы с классами и продуктами в базе данных PostgreSQL.

Задание выполнено в полном объёме, все функции работают корректно.