**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 1.2. **Приемы моделирования Перечислений**

Студенты:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Группа | ФИО | Подпись исп. | Оценка | Подпись преп. |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

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

Санкт-Петербург

2025

**ДОМАШНЕЕ ЗАДАНИЕ**

|  |
| --- |
| **Тема работы** : Задание 1.2.Приемы моделирования *Перечислений*  **Требования** |

1.1. Разработать проект каркаса для работы с *Перечислениями.*

Функциональные требования:

1. Схема классификатора должна сохраняться в базе данных

2. Обеспечить

* формирование классификатора перечислений
* создание нового перечисления заданного класса;
* редактирование списка значений перечисления;
* Изменение порядка расположения позиций в списке значений;
* Вывод значений перечисления;
* Вывод перечислений;
* Выбор значения перечисления.

1.2. Содержание работы

1.  Провести анализ  фрагментов предметной области выбранного примера, связанной с поддержанием сервисов работы со справочниками изделий с представлением примеров атрибутов изделий с типами значений типа *Перечисление*.

2.Разработать необходимые модели классов этапа проектирования с пояснениями принятых решений.

3.Разработать модели хранения данных с представлением диаграмм Сущность-Связь и описанием введенных сущностей, атрибутов и связей. 4.Расширить БД из задания 1.1 и занести необходимые метаданные

5.Разработать и описать необходимый набор SQL-процедур, обеспечивающий поддержку описанных выше требований.

6.Выбрать представительный пример классов перечислений и описать их в БД.

​​​​​​​7.Выполнить тестирование  работоспособности реализованного фрагмента  проекта моделирования перечислений с протоколированием результатов.

8.Оформить все материалы в виде отчета. Отчет представить в виде файла \*.docx

9.Продемонстрировать работоспособность разработанного фрагмента приложения.

|  |  |  |
| --- | --- | --- |
| № | Этап выполнения | Результат |
|  | Анализ исходных данных | Содержательное описание исходных с примерами атрибутов типа Перечисление. Модель классов предметной области. |
|  | Разработать функциональные требования к проектируемой подсистемы | Диаграммы вариантов использования. Описание компонентов диаграмм |
|  | Разработать модель классов этапа проектирования для моделирования Перечислений различных типов. | Диаграммы классов с операциями и атрибутами. Описание компонентов диаграмм. |
|  | Разработать модель хранения в среде СУБД данных о Перечислениях. | Диаграммы ER. Скрипты метаданных с комментариями. |
|  | Разработать основные SQL-процедуры для работы с Перечислениями | Скрипты SQL-процедур с комментариями. |
|  | Тестирование процедур | Описание тестов. Скрипты исходных данных для тестов.  Скриншоты результатов тестирования. |

1.3. Исходные данные для задачи

Примеры перечислений согласовать с преподавателем (Файл *Бланки Заказа.rar*).

1.4. Рекомендуемые инструменты

1. Редактор StarUML
2. Редактор Visio. Модель Database.Entity relationship
3. DB-редактор IBExpert
4. СУБД Firebird 2.5.3.
5. Текстовый редактор Notepad
6. Текстовый редактор Word
7. Самостоятельный выбор инструментов

1.5. Рекомендуемые источники

1. Data Definition Guide : файл DataDef.pdf

2.Language Reference Guide: файл LangRef.pdf

3. Приемы объектно-ориентированного проектирования. Паттерны проектирования: Изд-во «ПИТЕР», 2008

4. Проектирование корпоративных информационных систем.: СПб, Изд-во СПбГЭТУ «ЛЭТИ», 2013

5. В.А.Дубенецкий, А.Г. Кузнецов Использование объектных моделей при создании информационных систем: учебно-методическое пособие. СПбГЭТУ «ЛЭТИ», 2016. 220 с.

6. В.А.Дубенецкий, А.Г. Кузнецов Проектирование информационных систем с использованием UML: учебно-методическое пособие. СПбГЭТУ «ЛЭТИ», 2016. 62 с.

|  |  |  |
| --- | --- | --- |
| **Содержание пояснительной записки**:  «Введение», «Анализ предметной обрасти с примерами перечислений», «Разработка объектной модели этапа проектирования», «Разработка модели хранения», «Разработка основных процедур», поддерживающих работы с перечислениями, «Результаты тестирования», «Заключение», «Список использованных источников») | | |
| Предполагаемый объем пояснительной записки:  Не менее 20 страниц. | | |
| Дата выдачи задания: | | |
| Дата сдачи отчета: | | |
| Дата защиты работы: | | |
| Студенты:   |  |  |  | | --- | --- | --- | | Группа | ФИО | Подпись исп. | |  |  |  | |  |  |  | |  |  |  | | | |
|  |  |  |
| Преподаватель |  | Дубенецкий В.А. |

**АННОТАЦИЯ**

Приводятся результаты проектирования и фрагменты реализации каркаса, поддерживающего работу с перечислениями на различных примерах изделий. Представлены результаты тестирования разработанных процедур с примерами различных перечислений.

**SUMMARY**

The results of the design and fragments of the implementation of the framework, which supports the work with enumerations on various examples of products, are given. The results of testing the developed procedures with examples of various enumerations are presented.

Оглавление

[1. Разработка функциональных требований к подсистеме 5](#_Toc158039154)

[2. Разработка модели классов этапа анализа 5](#_Toc158039155)

[3. Разработка модели хранения 9](#_Toc158039156)

[4. Реализация 11](#_Toc158039157)

[5. Тестирование 17](#_Toc158039158)

## 1. Разработка функциональных требований к подсистеме

Для разработки концептуального проекта будем использовать редактор «StarUML».

Описание продолжим в рамках проекта с именем *Лаб*. Все компоненты, разработанные в рамках Задания 1.1, доступны для использования. База данных с аллиесом LAB.

Создадим диаграмму вариантов использования с именем *Работа с перечислениями.*

Создадим следующие варианты использования.

На рис. 1 приведена диаграмма ВИ *Работа с перечислениями*.



Рис. 1. Диаграмма ВИ для задачи *Работа с перечислениями*

## 2. Разработка модели классов этапа анализа

В качестве примера выберем бланк заказа воздуховода.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **БЛАНК-ЗАКАЗ** | | | | | | | | | | | | |
| **на переход с диаметра на прямоугольное сечение** | | | | | | | | | | | | |
| Заказ №\_\_\_\_\_\_\_\_ от \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | | | **Организация** | | |  | | | |  | |  |
| **Контактное лицо** | | |  | | | |  | |  |
| **Контактный телефон (e-mail)** | | |  | | | |  | |  |
| **Сроки изготовления** | | |  | | | |  | |  |
|  | | | | | | **Тип соединения** | | | | | | |
| **A** | | | **шино-рейка** | | | |
| **B** | | | **соединение на фланцах** | | | |
| **C** | | | **без фланцев** | | | |
| **D** | | |  | | | |
| **E** | | |  | | | |
|  | | |  | | | |
|  | | | | | | |  | | | | | |
|  | **А х В** | **L** | **D** | **Исполнение** | **Соединение** | | | **Кол-во** | | | **Примечание** | |
| **№** | **мм** | **мм** | **мм** |  |  | | | **шт.** | | |  | |
|  |  |  |  |  |  | | |  | | |  | |
|  |  |  |  |  |  | | |  | | |  | |
|  |  |  |  |  |  | | |  | | |  | |
|  |  |  |  |  |  | | |  | | |  | |
|  |  |  |  |  |  | | |  | | |  | |
|  |  |  |  |  |  | | |  | | |  | |
|  |  |  |  |  |  | | |  | | |  | |

Рис. 2. Пример исходных данных для формирования модели классов с перечислениями

На рис. 3 представлена диаграмма классов, поддерживающая ведение данных о воздуховодах.



Рис. 3. Фрагмент диаграммы классов с использованием перечислений

**Требования к расширению модели перечислений**

1. Множество перечислений в приложении может быть велико (100 и больше).
2. Добавление в приложение нового перечисления потребует внесение изменений в метаданные и код.
3. Требуется расширить множество типов значений перечислений.
4. Для численных перечислений желательно указывать единицу измерения значения перечисления.

**Разработка модели классов этапа проектирования**

Классификатор перечислений представлен на рис. 4.



Рис. 4. Расширение классов перечислений

Так как терминальный класс перечисления известен, объединим данные о строках различных подклассов в единый класс *Позиция перечисления*. Соответствующая диаграмма классов приведена на рис. 5.



Рис. 5. Модель перечислений с единым списком позиций перечисления

*Классификатор перечислений* аналогичен *Классификатору изделий*. Создадим *Классификатор объектов*, в который включим *Классификатор изделий* и *Классификатор перечислений*. Результат представлен на рис. 6.



Рис. 6. Обобщенная модель для работы с перечислениями

Введем метакласс *Классификатор перечислений*, который будет выступать в качестве «фабрики ».

Операции *Создать(), Удалить(), Показать список(), Редактировать список значений(), Выбрать позицию()* должны обеспечить необходимые сервисы для работы с перечислениями.

Класс *Позиция перечисления* и ассоциация *Множество значений* введены для описания списков допустимых значений перечислений.

**Особенности модели (рис. 6**):

1. В модели, представленной на рис. 6, удалось повторно использовать все решения, полученные при работе с классификатором изделий.
2. Появилась возможность создавать новые классы перечислений и списки их значений.
3. Работа со значениями различных перечислений унифицирована.

## 3. Разработка модели хранения

Для разработки диаграммы ER воспользуемся редактором «VISIO». На рис. 7 представлена диаграмма, описывающая сущности и связи модели хранения.



Рис. 7. Локальная ERD для моделирования перечислений (вариант 1)

Независимая сущность *CHEM\_CLASS* позволяет вести классификатор перечислений. Ассоциативная сущность *POS\_ENUM* позволяет вести списки допустимых значений перечислений. Модель соответствует всем 4 формам нормализации. Однако ссылаться на значение приходится через составной ключ. Искусственно можно преобразовать исходную схему к виду, представленному на рис. 8.



Рис. 8. Локальная ERD для моделирования перечислений с уникальным простым ключем.

При таком преобразовании потеряна информация о зависимости. Для поддержания зависимости необходимо ввести уникальный индекс (*ID\_ENUM, NUM*).

## Реализация

Расширим базу из задания 1.1. Создадим дополнительные таблицы:

/\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Дополнительные таблицы\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

CREATE TABLE POS\_ENUM (

ID\_POS INTEGER NOT NULL, /\*Код значения\*/

ID\_ENUM INTEGER NOT NULL, /\*Ид. перечисления\*/

NUM INTEGER NOT NULL, /\*Порядковый номер значения\*/

SHORT\_NAME VARCHAR(15), /\*Обозначение значения\*/

NAME VARCHAR(150), /\*Имя значения\*/

RVAL DOUBLE PRECISION, /\*вещественное значение\*/

INT\_VAL INTEGER, /\*целое значение\*/

PIC\_VAL VARCHAR(50) /\*путь к файлу картинки\*/

);

ALTER TABLE POS\_ENUM ADD PRIMARY KEY (ID\_POS);

/\*\*\* Foreign keys \*\*\*/

ALTER TABLE POS\_ENUM ADD FOREIGN KEY (ID\_ENUM) REFERENCES CHEM\_CLASS (ID\_CLASS);

/\*\*\* Indices \*\*\*/

CREATE UNIQUE INDEX POS\_ENUM\_I ON POS\_ENUM (ID\_ENUM, NUM);

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

/\*\*\*\*\*\*\*\*\*\* Создание генератора \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

CREATE GENERATOR POS\_ENUM\_GEN;

/\*\*\*\*\*\*\*\*\*\* Создание процедуры для вызова генератора \*\*\*\*\*\*\*\*\*\*\*\*\*/

create procedure GEN\_POS\_ENUM

returns ( ONEW integer)

as

begin

/\*функция: возвращает новый уникальный идентификатор\*/

oNew=GEN\_ID(POS\_ENUM\_GEN,1); /\*вызов генератора\*/

suspend;

end

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

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ТРИГГЕРЫ \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

CREATE TRIGGER TD\_POS\_ENUM FOR CHEM\_CLASS BEFORE DELETE AS

BEGIN

delete from POS\_ENUM

where ID\_POS= OLD.ID\_CLASS;

end

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

Разработаем необходимые процедуры для создания и работы с перечислениями:

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Процедуры \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

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

create procedure DEL\_ENUM (pIdEnum integer)

returns(oRes integer)

as

DECLARE VARIABLE vYes INTEGER;

begin

/\*функция: Удаление перечисления

вход: pIdEnum - ид. перечисления

выход:oRes - 0 - ошибка, 1 - ок

эффекты: 1. если есть параметр типа удаляемого перечисления, то удаление не происходит

\*/

select count(\*) from PARAMETR

where TYPE\_PAR=:pIdEnum

into :vYes;

if(:vYes=0) then

begin

delete from ENUM

where ID\_ENUM=:pIdEnum;

oRes=1;

end

else

oRes=0;

suspend;

end

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

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

create procedure DEL\_ENUM\_VAL (pIdPos integer)

returns(oRes integer)

as

DECLARE VARIABLE vYes INTEGER;

DECLARE VARIABLE vIdEnum INTEGER;

begin

/\*функция: Удаление значения перечисления\*/

вход: pIdPos - ид. значения

выход:oRes - 0 - ошибка, 1 - ок

эффекты: 1. если есть параметр типа удаляемого перечисления, то удаление не происходит

\*/

select ID\_ENUM from POS\_ENUM

where ID\_POS=:pIdVal

into :vIdEnum;

select count(\*) from PARAMETR

where TYPE\_PAR=:vIdEnum

into :vYes;

if(:vYes=0) then

begin

delete from POS\_ENUM

where ID\_ENUM=:vIdEnum

and ID\_VAL=:pIdPos;

oRes=1;

end

else

oRes=0;

suspend;

end

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

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

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

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

/\*\*\*\*\*\*\*\*\*\*\*\* Значение в список Перечисления \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

create or alter procedure INS\_VAL\_ENUM(pIdEnum integer,pShName varchar(15),pName varchar(150), pRVal double precision, pIntVal integer,pPicVal varchar(50))

returns(oNew integer,oNum integer,oRes integer)

as

DECLARE VARIABLE vYes INTEGER;

DECLARE VARIABLE vIdClass INTEGER;

DECLARE VARIABLE vYesStr INTEGER;

DECLARE VARIABLE vYesInt INTEGER;

DECLARE VARIABLE vYesPic INTEGER;

DECLARE VARIABLE vYesR INTEGER;

begin

/\*функция: Добавляет новое значение pIdEnum

вход:pIdENum - ид. перечисления

pShName - обозначение

pName - имя

pRVal - веществ значение

pIntVal – целочисл. значение

pPicVal – ссылка на рисунок

выход: oNum - порядковый номер значения в списке перечисления

oRes - 0- операция не выполнена, 1 - операция выполнена успешно

эффекты:

\*/

oRes=1;

vYes=0;

vYesStr=0;

vYesPic=0;

vYesR=0;

vYesInt=0;

select OYES from IN\_GR(106,:pIdEnum)

into :vYesStr;

select OYES from IN\_GR(105,:pIdEnum)

into :vYesPic;

if ((:vYesStr >0) or (:vYesPic >0)) then

begin

select count(\*) from POS\_ENUM

where ID\_ENUM=:pIdEnum and ((SHORT\_NAME=:pShName or NAME=:pName)

or (:pShName is null) or (:pName is null))

into :vYes;

if ((:pShName is null) or (:pName is null)) then vYes=1;

end

select OYES from IN\_GR(108,:pIdEnum)

into :vYesR;

if((:vYesR >0) and (:pRVal is null)) then vYes=1;

select OYES from IN\_GR(109,:pIdEnum)

into :vYesInt;

if((:vYesInt >0) and (:pIntVal is null)) then vYes=1;

if (:vYes=0) then

begin

select MAX(NUM )from POS\_ENUM

where ID\_ENUM=:pIdEnum

into :oNum;

if(:oNum is null) then oNum=0;

oNum=oNum+1;

oRes=1;

select oNew from GEN\_POS\_ENUM

into :oNew;

insert into POS\_ENUM(ID\_ENUM,ID\_POS,NUM,SHORT\_NAME,NAME,RVAL,INT\_VAL,PIC\_VAL)

values(:pIdEnum,:oNew,:oNum,:pShName,:pName,:pRVal,:pIntVal,:pPicVal);

end

else

oRes=0;

suspend;

end

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

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

create or alter procedure UP\_VAL (

PIDVAL integer)

returns (

ONUM integer,

OPREDNUM integer,

ORES integer)

as

declare variable VIDENUM integer;

declare variable VIDPRED integer;

declare variable VMAXNUM integer;

declare variable VMINNUM integer;

begin

/\*функция: Переместить значение вверх

вход: pIdVal - ид. значения\*/

oPredNum=0;

oRes=0;

select ID\_ENUM, NUM from POS\_ENUM

where ID\_POS=:pIdVal

into :vIdEnum,:oNum;

select MAX(NUM) from POS\_ENUM

where ID\_ENUM=:vIdEnum

into :vMaxNum;

select MIN(NUM) from POS\_ENUM

where ID\_ENUM=:vIdEnum

into :vMinNum;

if(:oNum>:vMinNum) then

begin

oRes=1;

select MAX(NUM) from POS\_ENUM

where ID\_ENUM=:vIdEnum

and NUM < :oNum

into :oPredNum;

select ID\_POS from POS\_ENUM

where ID\_ENUM=:vIdEnum

and NUM = :oPredNum

into :vIdPred;

update POS\_ENUM

set NUM=:vMaxNum+1

where ID\_POS=:vIdPred;

update POS\_ENUM

set NUM=:oPredNum

where ID\_POS=:pIdVal;

update POS\_ENUM

set NUM=:oNum

where ID\_POS=:vIdPred;

end

suspend;

end

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

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

create or alter procedure DOWN\_VAL (pIdVal integer)

returns(oNum integer, oPostNum integer,oRes integer)

as

declare variable vIdEnum integer;

declare variable vIdPost integer;

declare variable vMaxNum integer;

declare variable vMinNum integer;

begin

/\*функция: Переместить значение вниз

вход: pIdVal – ид. значения\*/

oPostNum=0;

oRes=0;

select ID\_ENUM, NUM from POS\_ENUM

where ID\_POS=:pIdVal

into :vIdEnum,:oNum;

select MAX(NUM) from POS\_ENUM

where ID\_ENUM=:vIdEnum

into :vMaxNum;

select MIN(NUM) from POS\_ENUM

where ID\_ENUM=:vIdEnum

into :vMinNum;

if(:oNum<:vMaxNum) then

begin

oRes=1;

select MIN(NUM) from POS\_ENUM

where ID\_ENUM=:vIdEnum

and NUM > :oNum

into :oPostNum;

select ID\_POS from POS\_ENUM

where ID\_ENUM=:vIdEnum

and NUM = :oPostNum

into :vIdPost;

update POS\_ENUM

set NUM=:vMaxNum+1

where ID\_POS =:vIdPost;

update POS\_ENUM

set NUM=:oPostNum

where ID\_POS=:pIdVal;

update POS\_ENUM

set NUM=:oNum

where ID\_POS=:vIdPost;

end

suspend;

end

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

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

/\*\*\*\*\*\*\*\*\*\*\*\*\*\* Найти список значений \*\*\*\*\*\*\*\*\*\*\*\*/

create or alter procedure FIND\_LIST\_ENUM(pIdEnum integer)

returns(oIdVal integer,oNum integer,oShName varchar(15), oName varchar(100), oRVal double precision,oIntVal integer, oPic blob)

as

declare variable vIdEnum integer;

begin

/\*функция: Находит список значений перечисления

вход: pIdEnum - ид. перечисления

выход:oIdVal - ид. значения

oNum - порядковый номер значения

oShName - обозначение

oName - имя

oRVal – веществ. значение

oIntVal – целочисл. значение

oPic – рисунок

эффекты:

\*/

for select ID\_POS, NUM, SHORT\_NAME, NAME, RVAL,INT\_VAL, PIC

from POS\_ENUM

where ID\_ENUM=:pIdEnum

order by NUM

into :oIdVal,:oNum,:oShName,:oName,:oRVal,:oIntVal, :oPic

do

suspend;

end

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

## 5. Тестирование

Для тестирования необходимо расширить классификатор объектов. Заполним данные классификатора перечислений. Для того воспользуемся уже разработанной процедурой *INS\_CLASS*:

select \* from INS\_CLASS ( null,"КлПО","Класс ПО",13) /\*32\*/

select \* from INS\_CLASS ( 32,"Изделие","Изделие",13) /\*102\*/

select \* from INS\_CLASS ( 32,"Enum","Перечисление",13) /\*103\*/

select \* from INS\_CLASS ( 103,"ПерСтр","Перечисление строк",13) /\*106\*/

select \* from INS\_CLASS ( 103,"ПерИзобр","Перечисление изображений",13)

/\*105\*/

select \* from INS\_CLASS ( 103,"ЧислПер","Численное перечисление",13) /\*107\*/

select \* from INS\_CLASS ( 107,"ПерВещ","Перечисление вещ",13) /\*108\*/

select \* from INS\_CLASS ( 107,"ПерЦел","Перечисление целых",13) /\*109\*/.

Далее необходимо объявить конкретные перечисления. Для тестирования выберем следующие перечисления.

Перечисление 1:

Тип соединения воздуховода прямоугольного сечения, прямой участок:

«А шино-рейка»;

«В соединение на фланцах»;

«С заглушка с двух сторон»;

«D заглушка с одной стороны».

Данное перечисление относится к классу *106.Перечисление строк* :

select \* from INS\_CLASS ( 106,"ТипСоед","Тип соединения воздуховода",13) /\* 114\*/.

Второе перечисление *Вариант исполнения* относится к классу

*105. Перечисление изображений*:

select \* from INS\_CLASS ( 105,"ВарИспВр","Вариант исполнения ",13) /\*115\*/.

Заполнение списков значений для объявленных терминальных классов перечислений. Для заполнения используем процедуру

/\*procedure INS\_VAL\_ENUM(pIdEnum integer,pShName varchar(15),pName varchar(150), pRVal double precision, pIntVal integer,pPicVal varchar(50))

returns(oNew integer,oNum integer,oRes integer) \*/

select \* from INS\_VAL\_ENUM(114,’A’,’шино-рейка’,null,null,null); /\*1\*/

select \* from INS\_VAL\_ENUM(114,’B’,’соединение на фланцах’, null,null,null); /\*2\*/

select \* from INS\_VAL\_ENUM(114,’C’,’без фланцев’, null,null,null); /\*3\*/

*Перечсление 115.Вариант исполнения* – родительский класс *105.Перечисление изображений*.

Формируем файлы изображений

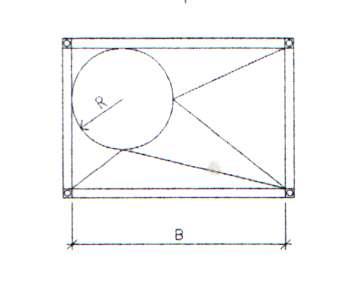
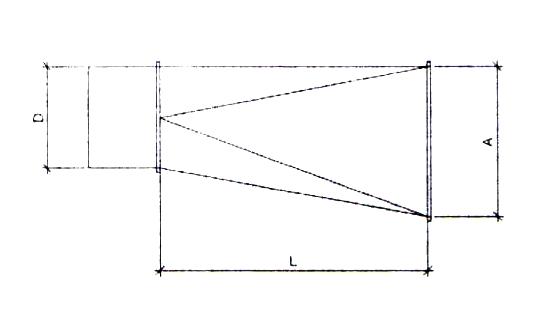
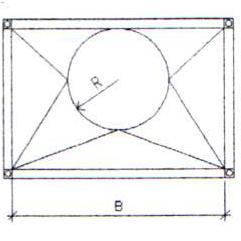
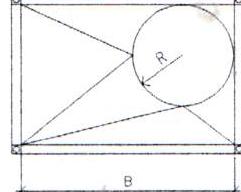
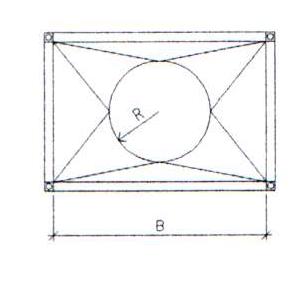
    

Рис. 9. Изображения для примера перечисления

select \* from INS\_VAL\_ENUM(115,’1’,’Вариант исполнения 1’, null,null,null); /\*5\*/

select \* from INS\_VAL\_ENUM(115,’2’,’Вариант исполнения 2’, null,null,null); /\*6\*/

select \* from INS\_VAL\_ENUM(115,'3','Вариант исполнения 3',null, null, null); /\*7\*/

select \* from INS\_VAL\_ENUM(115,'4','Вариант исполнения 4',null, null, null); /\*8\*/

select \* from INS\_VAL\_ENUM(115,'4','Вариант исполнения 4',null, null, null); /\*9\*/

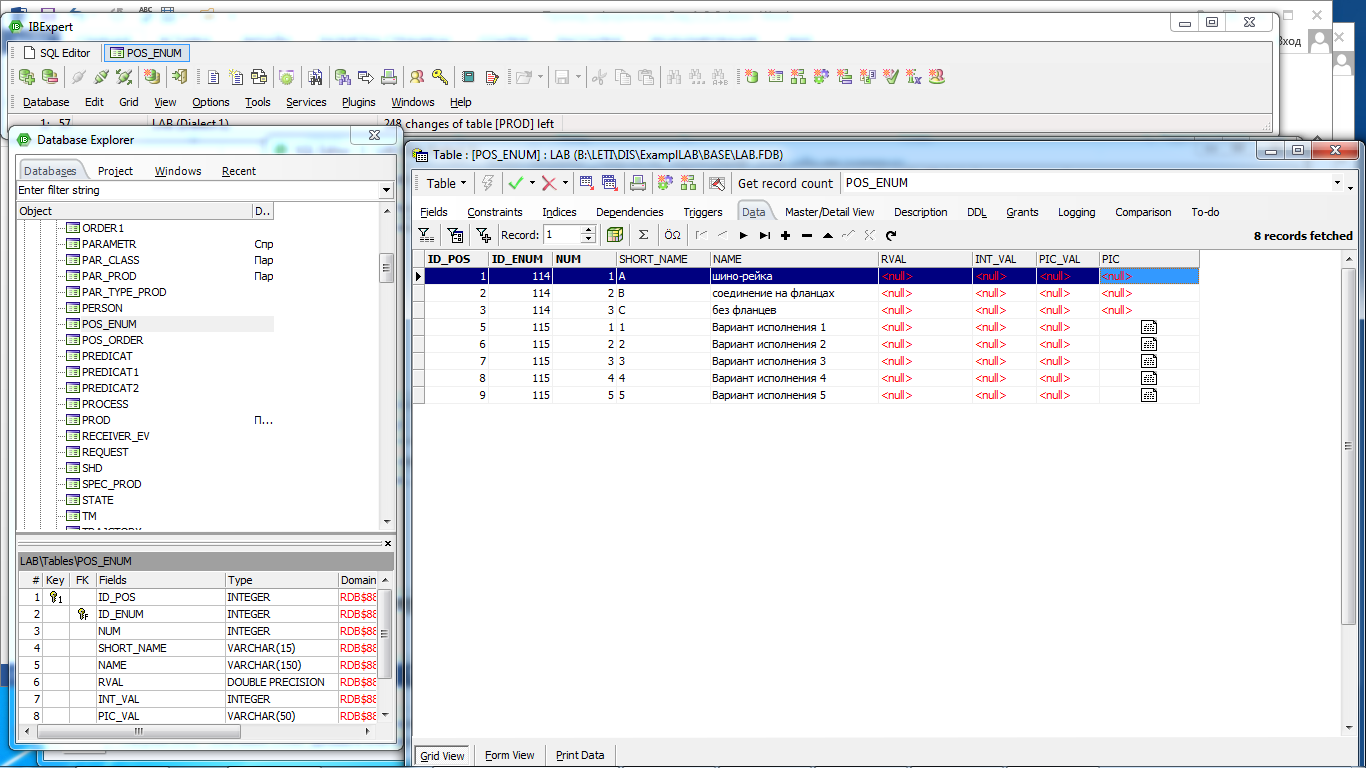


Рис. 10. Результаты заполнения данных для примеров перечислений

**Результаты тестирования.**

Изменение порядка.

Проверка процедуры *UP\_VAL*

Запрос: select \* from UP\_VAL (3)

Ответ представлен на рис. 11.

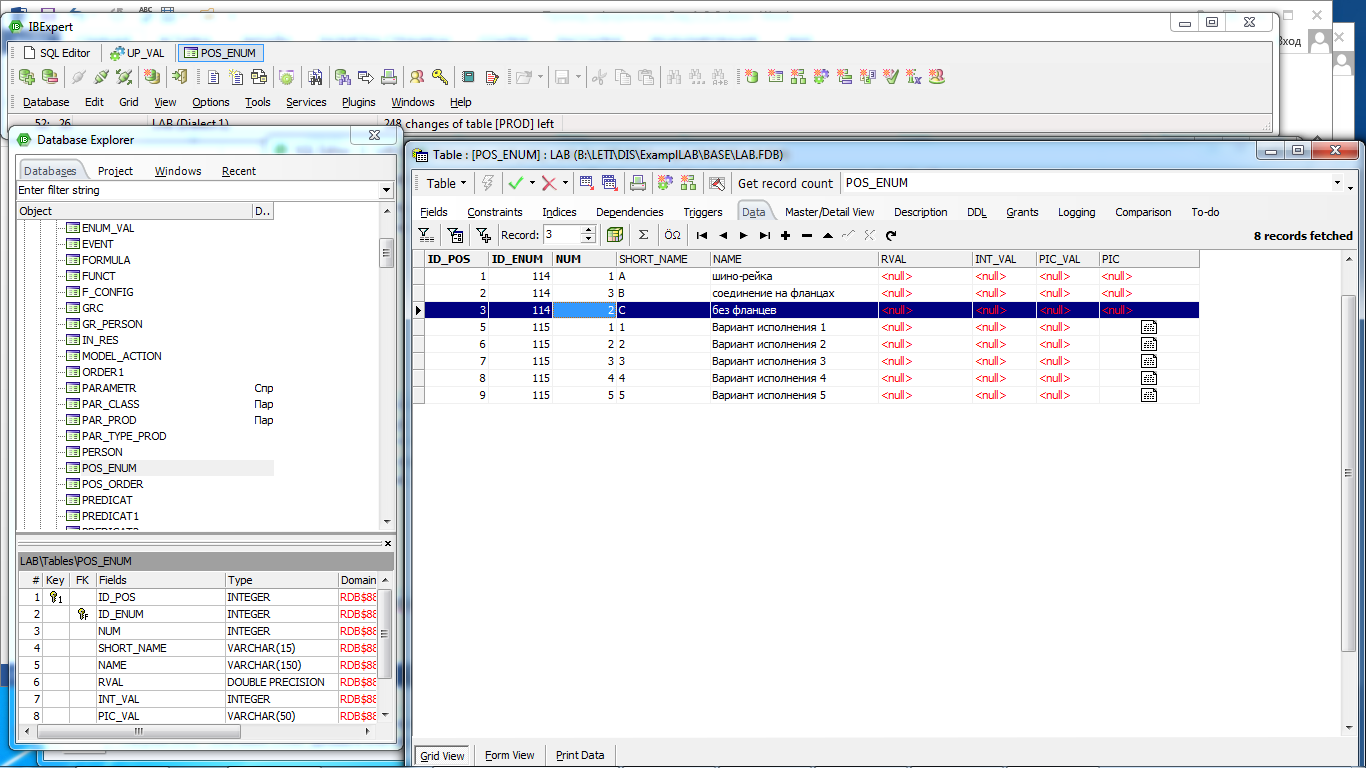


Рис. 11. Результат тестирования процедуры *UP\_VAL*

Проверка процедуры *DOWN\_VAL.*

Запрос:

select \* from DOWN\_VAL (3)

Ответ представлен на рис. 12.

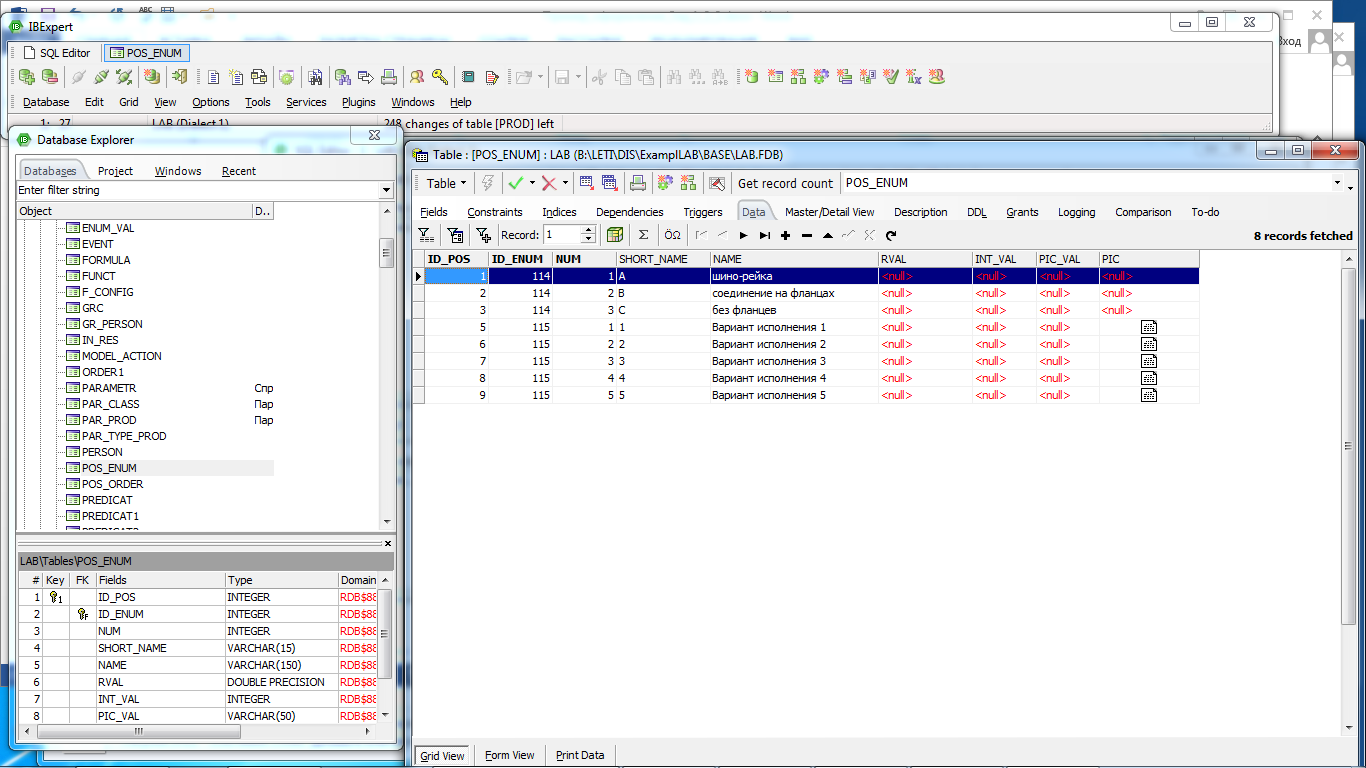


Рис. 12. Результат тестирования процедуры DOWN\_VAL (3)

Проверка процедуры *FIND\_LIST\_ENUM.*

Запрос:

select \* from FIND\_LIST\_ENUM(115)

Ответ представлен на рис. 13.

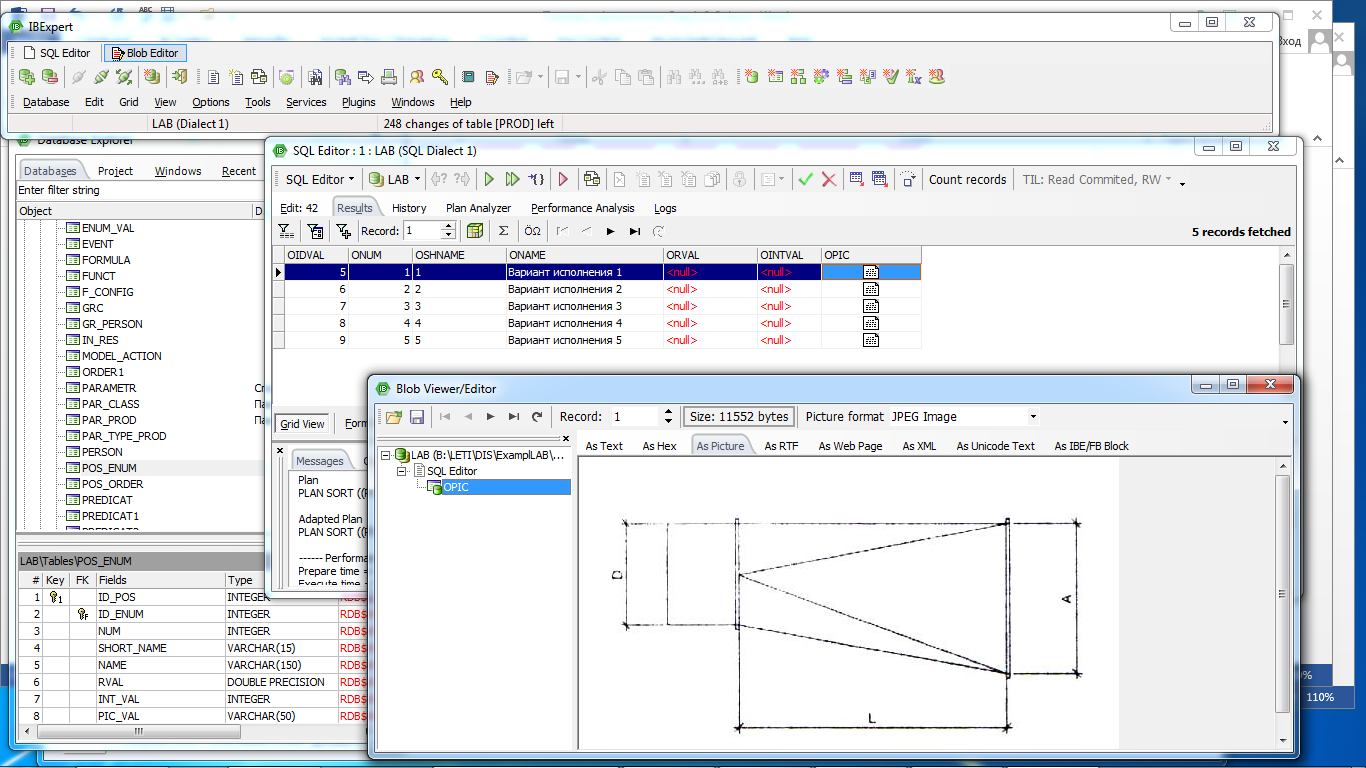


Рис. 13. Результат тестирования процедуры FIND\_LIST\_ENUM(115)

Запрос:

select \* from FIND\_LIST\_ENUM(114).

Ответ представлен на рис. 14.

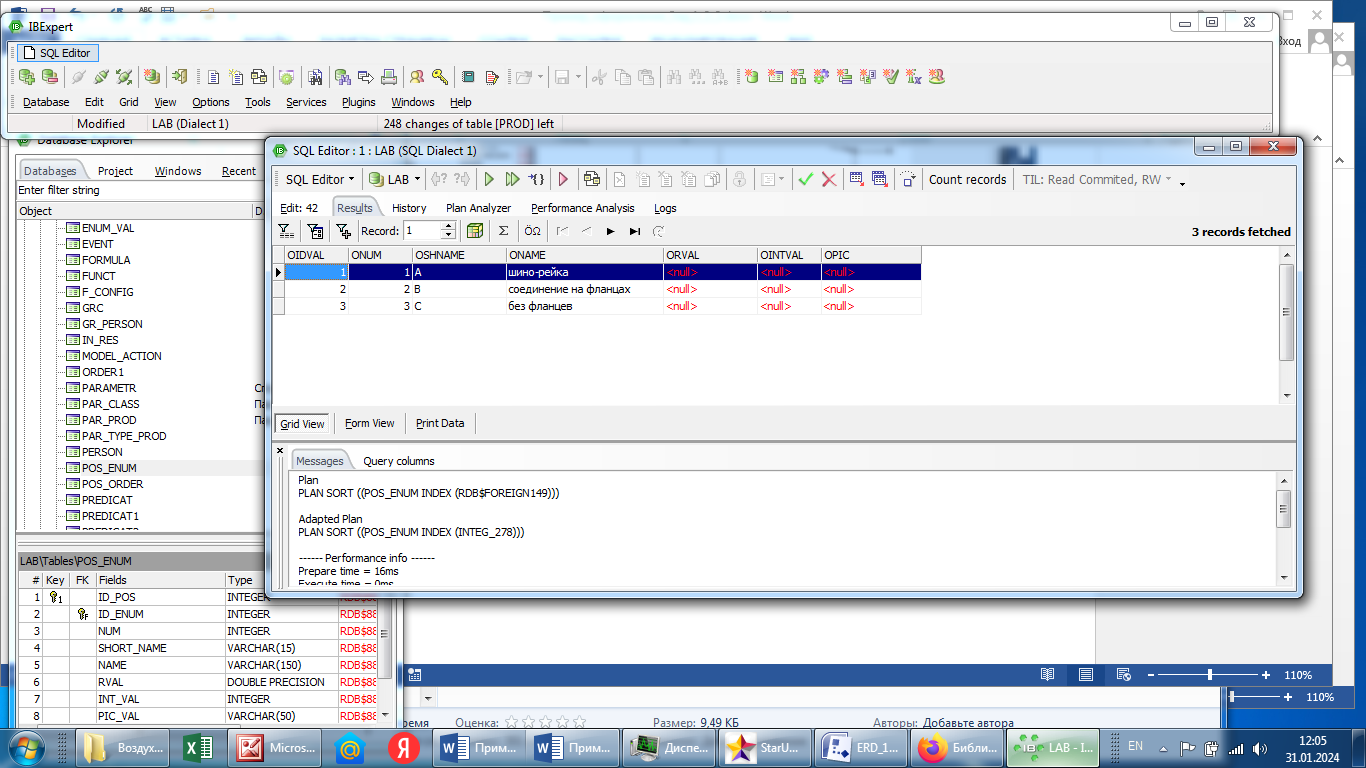


Рис. 14. Результат тестирования процедуры FIND\_LIST\_ENUM(114)

**Выводы**

1. Сервисы для работы с перечислениями удалось реализовать с использованием решений по работе с классификатором объектов.

2. Конструктор перечислений различных типов работает корректно.

3. Основные сервисы по работе с перечислениями работают корректно.