ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА»  
(РУТ (МИИТ))

Институт транспортной техники и систем управления

Кафедра «Управление и защита информации»

ОТЧЁТ  
О ЛАБОРАТОРНОЙ РАБОТЕ №3

По дисциплине «Основы построения защищенных баз данных»

ВАРИАНТ 13

Выполнил: ст. гр. ТКИ-441

Фамилия Порхун Д.Д.

Проверил: к.т.н., доц. Васильева М. А.

Москва 2024

## Цель работы

Изучить операторы SQL диалекта PostgreSQL, необходимые для работы с данными, находящимися в разных таблицах. Научиться создавать запросы на соединение данных из множества таблиц. Отчет по выполненной работе должен быть выполнен **с соблюдением ГОСТ по НИР 7.32**

## РЕЗУЛЬТАТЫ ВЫПОЛНЕННОЙ РАБОТЫ

Задание 1

1. Формулировка задания:

Показать бонус (Bonus) и поля FirstName, LastName из таблиц Person.Person, Sales.SalesPerson.

1. Текст скрипта:

**SELECT**

*sp*."Bonus",

*p*."FirstName",

*p*."LastName"

**FROM**

"Person"."Person" *p*

**JOIN**

"Sales"."SalesPerson" *sp* **ON** *p*."BusinessEntityID" = *sp*."BusinessEntityID";

1. Результат выполнения скрипта:

Результат выполнения скрипта представлен ниже (Рисунок 1).

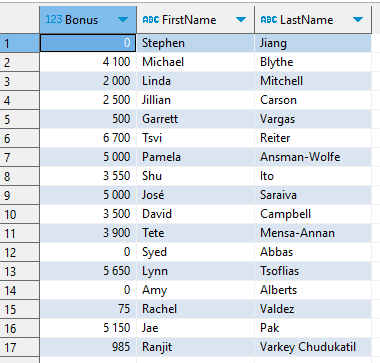


Рисунок 1 − Результат выполнения первого задания

Задание 2

1. Формулировка задания:

Показать список языков (поле Name), в котором указано, есть ли описание товаров на данном языке или нет, из таблиц Production.Culture, Production.ProductModelProductDescriptionCulture, используя LEFT OUTER JOIN.

1. Текст скрипта:

**SELECT**

*c*."Name" **AS** *"Language"*,

**CASE**

**WHEN** *pd*."CultureID" **IS** **NOT** **NULL** **THEN** **'Есть описание'**

**ELSE** **'Нет описания'**

**END** **AS** *"DescriptionStatus"*

**FROM**

"Production"."Culture" *c*

**LEFT** **OUTER** **JOIN**

"Production"."ProductModelProductDescriptionCulture" *pd* **ON** *c*."CultureID" = *pd*."CultureID";

1. Результат выполнения скрипта:

Результат выполнения скрипта представлен ниже (Рисунок 2).

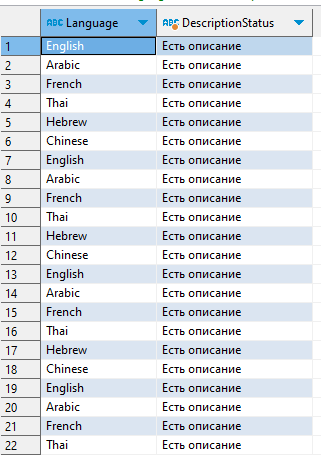


Рисунок 2 − Результат выполнения второго задания

Задание 3

1. Формулировка задания:

Показать список людей (поля FirstName, LastName), в котором указано, есть ли у человека часы отпуска (VacationHours) или нет, из таблиц HumanResources.Employee, Person.Person, используя RIGHT OUTER JOIN.

1. Текст скрипта:

**SELECT**

*p*."FirstName",

*p*."LastName",

**CASE**

**WHEN** *e*."VacationHours" **IS** **NOT** **NULL** **THEN** **'Есть часы отпуска'**

**ELSE** **'Нет часов отпуска'**

**END** **AS** *"VacationStatus"*

**FROM**

"HumanResources"."Employee" *e*

**RIGHT** **OUTER** **JOIN**

"Person"."Person" *p* **ON** *e*."BusinessEntityID" = *p*."BusinessEntityID";

1. Результат выполнения скрипта:

Результат выполнения скрипта представлен ниже (Рисунок 3).

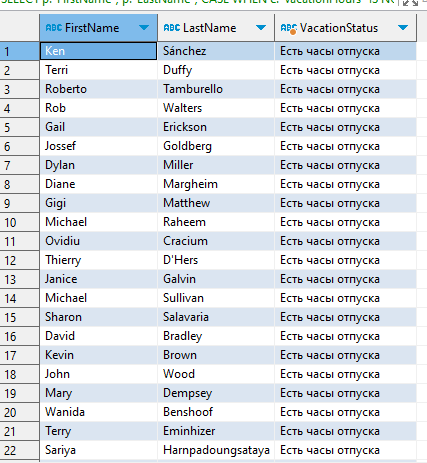


Рисунок 3 − Результат выполнения второго задания

Задание 4

1. Формулировка задания:

Показать список покупателей (CustomerID), к которым относятся несколько сроков выполнения заказа (DueDate), из таблицы Sales.SalesOrderHeader, используя SELF JOIN.

1. Текст скрипта:

**SELECT**

*so1*."CustomerID",

*so1*."DueDate"

**FROM**

"Sales"."SalesOrderHeader" *so1*

**JOIN**

"Sales"."SalesOrderHeader" *so2* **ON** *so1*."CustomerID" = *so2*."CustomerID"

**WHERE**

*so1*."DueDate" <> *so2*."DueDate"

**ORDER** **BY**

*so1*."CustomerID", *so1*."DueDate";

1. Результат выполнения скрипта:

Результат выполнения скрипта представлен ниже (Рисунок 4).

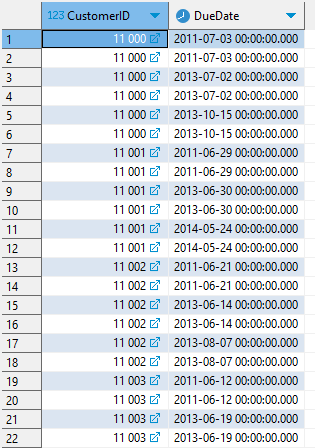


Рисунок 4 − Результат выполнения второго задания

Задание 5

1. Формулировка задания:

Показать список сроков выполнения заказа (DueDate), которые относятся к нескольким покупателям (CustomerID), из таблицы Sales.SalesOrderHeader, используя SELF JOIN.

1. Текст скрипта:

**SELECT**

*so1*."DueDate",

**COUNT**(**DISTINCT** *so1*."CustomerID") **AS** *"CustomerCount"*

**FROM**

"Sales"."SalesOrderHeader" *so1*

**JOIN**

"Sales"."SalesOrderHeader" *so2* **ON** *so1*."DueDate" = *so2*."DueDate" **AND** *so1*."CustomerID" <> *so2*."CustomerID"

**GROUP** **BY**

*so1*."DueDate"

**HAVING**

**COUNT**(**DISTINCT** *so1*."CustomerID") > 1

**ORDER** **BY**

*so1*."DueDate";

1. Результат выполнения скрипта:

Результат выполнения скрипта представлен ниже (Рисунок 5).

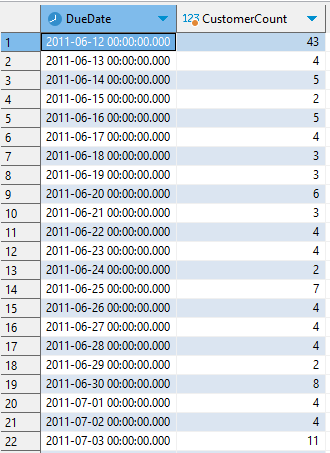


Рисунок 5 − Результат выполнения второго задания

Задание 6

1. Формулировка задания:

Показать подкатегории продуктов (ProductSubcategoryID из таблицы Production.Product), которые имеют несколько цветов, содержащих в названии букву “e”, используя SELF JOIN.

1. Текст скрипта:

**SELECT**

p1.**"ProductSubcategoryID"**,

p1.**"Name"**

**FROM**

**"Production"**.**"Product"** p1

**JOIN**

**"Production"**.**"Product"** p2 **ON** p1.**"ProductSubcategoryID"** = p2.**"ProductSubcategoryID"**

**WHERE**

p1.**"Name"** **LIKE** **'%e%'** **AND**

p1.**"Color"** <> p2.**"Color"**

1. Результат выполнения скрипта:

Результат выполнения скрипта представлен ниже (Рисунок 6).

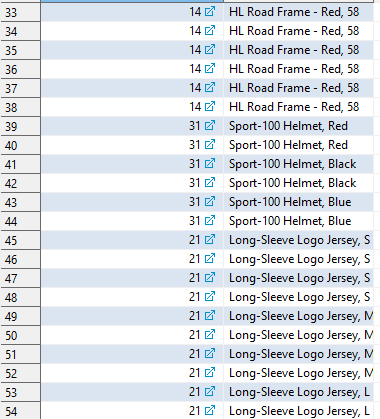


Рисунок 6 − Результат выполнения второго задания

Задание 7

1. Формулировка задания:

Показать комбинированный список таблиц Production.Product по полям ProductID, StandardCost, Purchasing.ProductVendor по полям ProductID, StandardPrice, используя UNION.

1. Текст скрипта:

**SELECT**

"ProductID",

"StandardCost" **AS** *Cost*

**FROM**

"Production"."Product"

**UNION**

**SELECT**

*pv*."ProductID",

"StandardPrice" **AS** *Cost*

**FROM**

"Purchasing"."ProductVendor" *pv*;

1. Результат выполнения скрипта:

Результат выполнения скрипта представлен ниже (Рисунок 7).

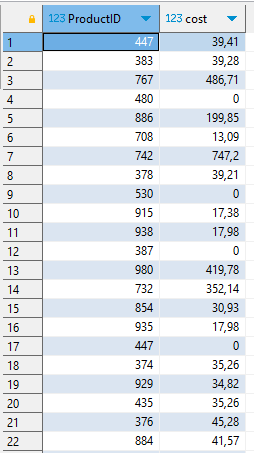


Рисунок 7 − Результат выполнения второго задания

Задание 8

1. Формулировка задания:

Показать список ProductID, которые содержатся в таблице Production.Product, но не содержатся в таблице Production.TransactionHistoryArchive.

1. Текст скрипта:

**SELECT** "ProductID"

**FROM** "Production"."Product"

**EXCEPT**

**SELECT** "ProductID"

**FROM** "Production"."TransactionHistoryArchive";

1. Результат выполнения скрипта:

Результат выполнения скрипта представлен ниже (Рисунок 8).

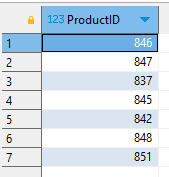


Рисунок 8 − Результат выполнения второго задания

Задание 9

1. Формулировка задания:

Ограничить результирующий набор, полученный в п.8.

1. Текст скрипта:

**SELECT** "ProductID"

**FROM** "Production"."Product"

**EXCEPT**

**SELECT** "ProductID"

**FROM** "Production"."TransactionHistoryArchive"

**limit** 5;

1. Результат выполнения скрипта:

Результат выполнения скрипта представлен ниже (Рисунок 9).

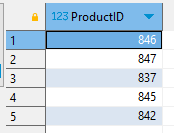


Рисунок 9 − Результат выполнения второго задания

Задание 10

1. Формулировка задания:

Пропустить 15 строк из результирующего набора, полученного в п.9.

1. Текст скрипта:

**SELECT** "ProductID"

**FROM** "Production"."Product"

**EXCEPT**

**SELECT** "ProductID"

**FROM** "Production"."TransactionHistoryArchive"

**limit** 5 **offset** 2;

1. Результат выполнения скрипта:

Результат выполнения скрипта представлен ниже (Рисунок 10).

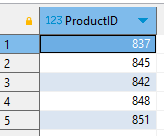


Рисунок 10 − Результат выполнения второго задания

## Отметка о выполнении лабораторной работы в веб-хостинге GitHub

Подтверждение о правильно выполненной работе представлено ниже (Рисунок 11).



Рисунок 11 − Подтверждение о правильно выполненной работе