

***Федеральное агентство по рыболовству***

***Федеральное государственное бюджетное образовательное***

***учреждение высшего образования***

***«Астраханский государственный технический университет»***

**Система менеджмента качества в области образования, воспитания, науки и инноваций сертифицирована**

**ООО «ДКС РУС» по международному стандарту ISO 9001:2015**

|  |  |
| --- | --- |
| Институт | Информационных технологий и коммуникаций |
| Направление | 09.03.01 Информатика и вычислительная техника |
| Профиль | «Автоматизированные системы обработки информации и управления» |
| Кафедра | «Автоматизированные системы обработки информации и управления» |

**Лабораторная работа № 14**

**«Функции в PostgreSQL. Проектирование структуры БД**»

по дисциплине «СУБД PostgreSQL»

|  |  |  |
| --- | --- | --- |
|  |  | Работа выполнена студентом группы ДИНРБ-31  Кузургалиев Р.А. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (Фамилия И.О.) подпись |
|  |  | Проверил работу:  ст. преподаватель Мамлеева А.Р,\_\_\_\_\_\_\_\_\_\_\_\_  (ученая степень, ученое звание, Фамилия И.О.) |

Работа защищена

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2024 г.

**АСТРАХАНЬ – 2024**

**СОДЕРЖАНИЕ ОТЧЕТА**

*Цель работы:*Изучить правила создания функций. Приобрести практические навыки создания функций в среде PostgreSQL. Научиться проектировать БД в целом на основе поставленного задания с использованием всех полученных ранее базовых знаний и навыков.

*Задание. Общая часть:*Ознакомиться с теоретическими сведениями о возможностях создания пользовательских функций в PostgreSQL. Разработать БД в соответствии с индивидуальным заданием. Создать функции, реализующие интерфейс для работы с базой данных. Проверить работоспособность функций путем выполнения этих функций с параметрами, обеспечивающими как успешное выполнение функции, так и невыполнение функции.

Обязательные требования к БД:

1) Наличие таблиц-справочников и таблиц, использующих справочники. Предусмотреть сохранение ссылочной целостности для таблиц, использующих таблицы-справочники.

2) Предусмотреть следующие роли:

а) оператор БД (пополнение справочников)

б) пользователь БД (основная работа с БД, с ограничениями для некоторого вида операций)

в) аналитик (разрешено выполнение запросов и функций, не изменяющих данные в БД)

г) администратор БД (просмотр протокола операций, любые изменения БД)

3) Действия, изменяющие БД пользователем с любой ролью протоколируются в таблице-журнале операций.

4) Для всех запросов необходимо создать индексы (для гарантированного использования индексов можно использовать отключение параметра **enable\_seqscan** в текущей сессии)

Создать функции, реализующие интерфейс для работы с базой данных. Проверить работоспособность функций путем выполнения этих функций с параметрами, обеспечивающими как успешное выполнение функции, так и невыполнение функции.

**Вариант 9:** База данных гарантийного ремонта. Должна содержать следующие данные: информацию о выпускаемых производителем товарах, гарантийных мастерских в разных городах и товарах, ремонт которых они могут производить, данные о выполненных ремонтах. Предусмотреть анализ следующих показателей: рейтинг убыточности товаров, обеспеченность каждого города мастерскими по разным группам товаров и всему ассортименту в целом.

1. Таблица Products для хранения информации о выпускаемых производителем товарах:

CREATE TABLE Products (

ProductId SERIAL PRIMARY KEY,

ProductName VARCHAR NOT NULL,

Manufacturer VARCHAR NOT NULL,

Rating DECIMAL

);

2. Таблица ServiceCenters для хранения информации о гарантийных мастерских в разных городах:

CREATE TABLE ServiceCenters (

CenterId SERIAL PRIMARY KEY,

City VARCHAR NOT NULL,

CenterName VARCHAR NOT NULL

);

3. Таблица RepairServices для связи между гарантийными мастерскими и товарами, ремонт которых они могут производить:

CREATE TABLE RepairServices (

CenterId INTEGER REFERENCES ServiceCenters(CenterId),

ProductId INTEGER REFERENCES Products(ProductId),

PRIMARY KEY (CenterId, ProductId)

);

4. Таблица Repairs для хранения данных о выполненных ремонтах:

CREATE TABLE Repairs (

RepairId SERIAL PRIMARY KEY,

CenterId INTEGER REFERENCES ServiceCenters(CenterId),

ProductId INTEGER REFERENCES Products(ProductId),

RepairDate DATE,

RepairCost DECIMAL

);

5. Таблица OperationLog для журналирования операций пользователя с базой данных:

CREATE TABLE OperationLog (

LogId SERIAL PRIMARY KEY,

UserName VARCHAR,

OperationDate TIMESTAMP,

OperationDescription VARCHAR

);

После создания таблиц необходимо настроить роли и функции для работы с базой данных. Вот примеры функций, которые могут быть полезны для работы с базой данных гарантийного ремонта:  
  
1. Функция для добавления нового товара в таблицу Products:

CREATE OR REPLACE FUNCTION addProduct(product\_name VARCHAR, manufacturer VARCHAR, rating DECIMAL) RETURNS VOID AS $$

BEGIN

INSERT INTO Products (ProductName, Manufacturer, Rating) VALUES (product\_name, manufacturer, rating);

INSERT INTO OperationLog (UserName, OperationDate, OperationDescription) VALUES ('admin', NOW(), 'Added new product: ' || product\_name);

END;

$$ LANGUAGE plpgsql;

2. Функция для удаления товара из таблицы Products:

CREATE OR REPLACE FUNCTION deleteProduct(product\_id INTEGER) RETURNS VOID AS $$

BEGIN

DELETE FROM Products WHERE ProductId = product\_id;

INSERT INTO OperationLog (UserName, OperationDate, OperationDescription) VALUES ('admin', NOW(), 'Deleted product with ID: ' || product\_id);

END;

$$ LANGUAGE plpgsql;

3. Функция для анализа рейтинга убыточности товаров:

CREATE OR REPLACE FUNCTION analyzeProductRating() RETURNS TABLE (

ProductName VARCHAR,

Manufacturer VARCHAR,

Rating DECIMAL

) AS $$

BEGIN

RETURN QUERY

SELECT ProductName, Manufacturer, Rating

FROM Products

ORDER BY Rating ASC;

END;

$$ LANGUAGE plpgsql;

1. Индекс для таблицы Products по полю ProductName:

CREATE INDEX idx\_Products\_ProductName ON Products (ProductName);

2. Индекс для таблицы ServiceCenters по полю City:

CREATE INDEX idx\_ServiceCenters\_City ON ServiceCenters (City);

3. Индексы для таблицы RepairServices по полям CenterId и ProductId (так как это составной ключ):

CREATE INDEX idx\_RepairServices\_CenterId ON RepairServices (CenterId);

CREATE INDEX idx\_RepairServices\_ProductId ON RepairServices (ProductId);

4. Индексы для таблицы Repairs по полям CenterId, ProductId и RepairDate:

CREATE INDEX idx\_Repairs\_CenterId ON Repairs (CenterId);

CREATE INDEX idx\_Repairs\_ProductId ON Repairs (ProductId);

CREATE INDEX idx\_Repairs\_RepairDate ON Repairs (RepairDate);

5. Индексы для таблицы OperationLog по полям UserName и OperationDate:

CREATE INDEX idx\_OperationLog\_UserName ON OperationLog (UserName);

CREATE INDEX idx\_OperationLog\_OperationDate ON OperationLog (OperationDate);

После создания индексов, чтобы гарантировать использование индексов при выполнении запросов, можно отключить параметр enable\_seqscan в текущей сессии. Например, таким образом:

SET enable\_seqscan = off;

1. Создание роли оператора БД (пополнение справочников):  
  
CREATE ROLE operator\_bd1 LOGIN;  
GRANT USAGE ON SCHEMA public TO operator\_bd;  
GRANT SELECT, INSERT, UPDATE, DELETE ON ALL TABLES IN SCHEMA public TO operator\_bd;  
  
2. Создание роли пользователя БД (основная работа с БД, с ограничениями для некоторых операций):  
  
CREATE ROLE user\_bd1 LOGIN;  
GRANT USAGE ON SCHEMA public TO user\_bd;  
GRANT SELECT, INSERT, UPDATE ON ALL TABLES IN SCHEMA public TO user\_bd;  
REVOKE DELETE ON ALL TABLES IN SCHEMA public FROM user\_bd;  
  
3. Создание роли аналитика (разрешено выполнение запросов и функций, не изменяющих данные в БД):  
  
CREATE ROLE analyst1 LOGIN;  
GRANT USAGE ON SCHEMA public TO analyst;  
GRANT SELECT ON ALL TABLES IN SCHEMA public TO analyst;  
  
4. Создание роли администратора БД (просмотр протокола операций, любые изменения БД):  
  
CREATE ROLE db\_admin1 LOGIN;  
GRANT ALL PRIVILEGES ON ALL TABLES IN SCHEMA public TO db\_admin;  
GRANT ALL PRIVILEGES ON ALL SEQUENCES IN SCHEMA public TO db\_admin;

**Вывод:** *:*в ходе лабораторной работы я изучил правила создания функций, приобрести практические навыки создания функций в среде PostgreSQL, а также научился проектировать БД в целом на основе поставленного задания с использованием всех полученных ранее базовых знаний и навыков.