ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

***«*САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ПЕТРА ВЕЛИКОГО»**

Высшая школа программной инженерии

Изображение выглядит как текст

Автоматически созданное описание

**ОТЧЕТ ПО ПРАКТИЧЕСКИМ ЗАДАНИЯМ**  
по дисциплине «Базы данных»

Изображение выглядит как стрела

Автоматически созданное описание

Студент

3530202/90202 Потапова А. М.

Преподаватель Прокофьев О. В.

­­

Санкт-Петербург

2021 г.

**Содержание**

[Практическое задание №1.1 Проектирование схемы базы данных 3](#_Toc102574592)

[Практическое задание №1.2 Создание и заполнение таблиц 4](#_Toc102574593)

[Практическое задание №1.3 Операторы манипулирования 10](#_Toc102574594)

[Практическое задание №1.4 Контроль целостности данных 14](#_Toc102574595)

[Практическое задание №2.1 Управление доступом 19](#_Toc102574596)

[Практическое задание №2.2 Функции и язык PL/pgSQL 22](#_Toc102574597)

# **Практическое задание №1.1 Проектирование схемы базы данных**

Постановка задачи

Практическое задание связано с проектированием схемы базы данных для работы приложения. Результатом данного практического задания является схема базы данных.

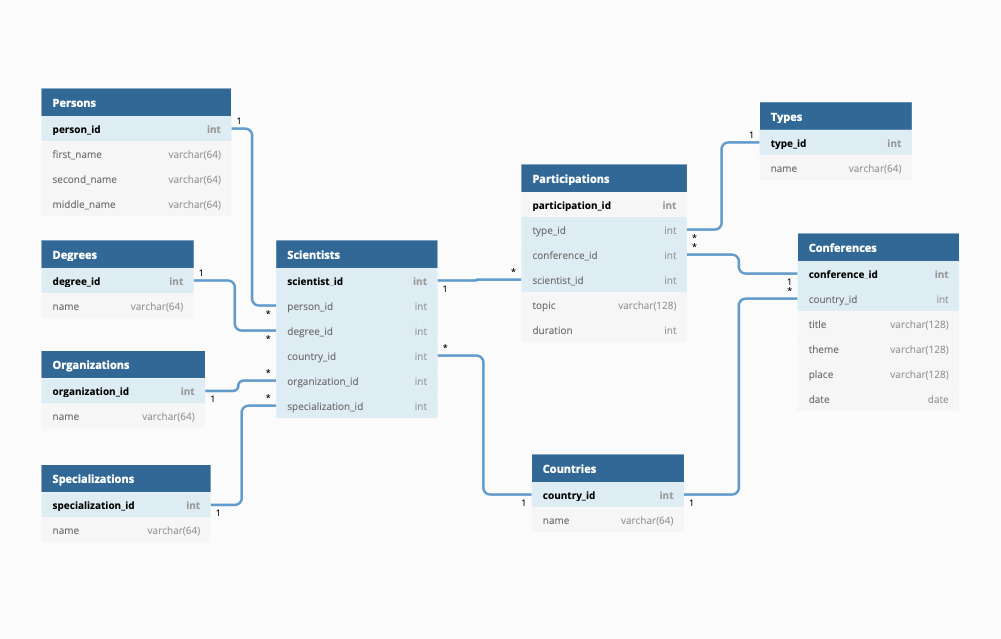
**Вариант 22**

Научная конференция



Ход работы

Следуя предметной области, была разработана следующая схема базы данных:



# **Практическое задание №1.2 Создание и заполнение таблиц**

Постановка задачи

Практическое задание заключается в подготовке SQL-скрипта для создания таблиц согласно схеме, полученной в предыдущем задании (с уточнением типов столбцов). Необходимо определить первичные и внешние ключи, а также декларативные ограничения целостности (возможность принимать неопределенное значение, уникальные ключи, проверочные ограничения и т. д.). Таблицы следует создавать в отдельной базе данных. Кроме того, нужно подготовить данные для заполнения созданных таблиц. Объем подготовленных данных должен составлять не менее 10 экземпляров для каждой из стержневых (неподчиняющиеся основные сущности) сущностей и 15 экземпляров для каждой из ассоциативных (подчиненные многие-ко-

многим). На основе этих данных необходимо создать SQL-скрипт для вставки соответствующих строк в таблицы БД.

Ход работы

1. Создание таблиц:

CREATE TABLE "Persons" (

"person\_id" SERIAL PRIMARY KEY,

"first\_name" varchar(64) NOT NULL,

"second\_name" varchar(64) NOT NULL,

"middle\_name" varchar(64)

);

CREATE TABLE "Countries" (

"country\_id" SERIAL PRIMARY KEY,

"name" varchar(64) NOT NULL,

UNIQUE(name)

);

CREATE TABLE "Degrees" (

"degree\_id" SERIAL PRIMARY KEY,

"name" varchar(64) NOT NULL,

UNIQUE(name)

);

CREATE TABLE "Specializations" (

"specialization\_id" SERIAL PRIMARY KEY,

"name" varchar(64) NOT NULL,

UNIQUE(name)

);

CREATE TABLE "Organizations" (

"organization\_id" SERIAL PRIMARY KEY,

"name" varchar(128) NOT NULL,

UNIQUE(name)

);

CREATE TABLE "Types" (

"type\_id" SERIAL PRIMARY KEY,

"name" varchar(64) NOT NULL,

UNIQUE(name)

);

CREATE TABLE "Scientists" (

"scientist\_id" SERIAL PRIMARY KEY,

"person\_id" int NOT NULL,

"degree\_id" int,

"country\_id" int,

"organization\_id" int,

"specialization\_id" int

);

CREATE TABLE "Participations" (

"participation\_id" SERIAL PRIMARY KEY,

"scientist\_id" int NOT NULL,

"conference\_id" int NOT NULL,

"type\_id" int,

"topic" varchar(128),

"duration" time

);

CREATE TABLE "Conferences" (

"conference\_id" SERIAL PRIMARY KEY,

"country\_id" int NOT NULL,

"title" varchar(128) NOT NULL,

"theme" varchar(128) NOT NULL,

"place" varchar(128),

"date" timestamp

);

2. Связи таблиц:

ALTER TABLE "Scientists" ADD FOREIGN KEY ("person\_id") REFERENCES "Persons" ("person\_id") ON DELETE CASCADE ON UPDATE CASCADE;

ALTER TABLE "Scientists" ADD FOREIGN KEY ("degree\_id") REFERENCES "Degrees" ("degree\_id") ON DELETE SET NULL ON UPDATE CASCADE;

ALTER TABLE "Scientists" ADD FOREIGN KEY ("country\_id") REFERENCES "Countries" ("country\_id") ON DELETE SET NULL ON UPDATE CASCADE;

ALTER TABLE "Scientists" ADD FOREIGN KEY ("organization\_id") REFERENCES "Organizations" ("organization\_id") ON DELETE SET NULL ON UPDATE CASCADE;

ALTER TABLE "Scientists" ADD FOREIGN KEY ("specialization\_id") REFERENCES "Specializations" ("specialization\_id") ON DELETE SET NULL ON UPDATE CASCADE;

ALTER TABLE "Participations" ADD FOREIGN KEY ("scientist\_id") REFERENCES "Scientists" ("scientist\_id") ON DELETE CASCADE ON UPDATE CASCADE;

ALTER TABLE "Participations" ADD FOREIGN KEY ("conference\_id") REFERENCES "Conferences" ("conference\_id") ON DELETE CASCADE ON UPDATE CASCADE;

ALTER TABLE "Participations" ADD FOREIGN KEY ("type\_id") REFERENCES "Types" ("type\_id") ON DELETE SET NULL ON UPDATE CASCADE;

ALTER TABLE "Conferences" ADD FOREIGN KEY ("country\_id") REFERENCES "Countries" ("country\_id") ON DELETE CASCADE ON UPDATE CASCADE;

3. Заполнение таблиц:

INSERT INTO "Countries" (name) VALUES

('Россия'),

('Франция'),

('Китай'),

('США'),

('Германия'),

('Великобритания'),

('Япония'),

('Индия'),

('Канада'),

('Бразилия'),

('Италия'),

('Испания'),

('Австралия'),

('Швейцария'),

('Израиль')

INSERT INTO "Persons" (first\_name, second\_name, middle\_name) VALUES

('Виктор','Попов','Сергеевич'),

('Джон','Гилберт', NULL),

('Кур','Дина', NULL),

('Луи','Де ла Круа', NULL),

('Владислав','Карташов','Константинович'),

('Виктория','Гукова','Олеговна'),

('Кристиан','Кане', NULL),

('Елена','Вальверде', NULL),

('Стефано','Волонте', NULL),

('Робберт','Эллис', NULL),

('Джуди','Далтон', NULL),

('Глория','Антонелли', NULL),

('Бруно','Перес', NULL),

('Рю','Ким', NULL),

('Ритик','Хан', NULL)

INSERT INTO "Degrees" (name) VALUES

('Доктор архитектурных наук'),

('Доктор биологических наук'),

('Доктор географических наук'),

('Доктор филологических наук'),

('Доктор технических наук'),

('Доктор медицинских наук'),

('Доктор физико-математических наук'),

('Доктор политологических наук'),

('Кандидат архитектурных наук'),

('Кандидат биологических наук'),

('Кандидат географических наук'),

('Кандидат филологических наук'),

('Кандидат технических наук'),

('Кандидат медицинских наук'),

('Кандидат физико-математических наук')

INSERT INTO "Specializations" (name) VALUES

('Градостроительство'),

('Политтехнолог'),

('Биохимия'),

('Системная аналитика'),

('Терапевт'),

('Космическая физика'),

('Зарубежная филология'),

('Информационная безопасность'),

('Реставрация архитектурных объектов'),

('Научная и преподавательская деятельность'),

('Аллергология и иммунология'),

('Молекулярная биология'),

('Биоинженерия'),

('Математические методы в современной физике'),

('Промышленная разработка')

INSERT INTO "Organizations" (name) VALUES

('Федерация европейских биохимических обществ'),

('Ученые будущего'),

('Яндекс'),

('Р-Фарм'),

('Группа 77'),

('Филологический факультет МГУ имени М.В.Ломоносова'),

('Международный математический союз'),

('Международная ассоциация академий наук'),

('Европейская ассоциация географов'),

('Международный исследовательский центр по сохранению и реставрации культурных ценностей'),

('Ассоциация вычислительной техники'),

('Институт высших научных исследований'),

('Международная академия астронавтики'),

('Международный совет по науке'),

('Союз социальной политики')

INSERT INTO "Types" (name) VALUES

('Участие с докладом'),

('Участие с публикацией, но без доклада'),

('Участие без публикации и доклада')

INSERT INTO "Scientists" (person\_id, degree\_id, country\_id, organization\_id, specialization\_id) VALUES

('1','7','1','13','6'),

('2','8','4','15','2'),

('3','6','15','1','3'),

('4','1','2','9','10'),

('5','5','1','2','4'),

('6','4','1','6','10'),

('10','14','6','1','3'),

('11','10','13','1','11'),

('12','13','10','11','4'),

('14','15','3','14','14'),

('7','5','2','12','8'),

('8','10','12','1','13'),

('9','3','11','9','10'),

('13','9','14','10','1'),

('15','1','7','10','9')

INSERT INTO "Conferences" (country\_id, title, theme, place, date) VALUES

('1','Международная научно-практическая конференция','Advances in Science and Technology','МГУ','2022-05-16 14:00:00'),

('1','Международная междисциплинарная конференция','Развитие науки в XXI веке','Научно-издательский центр «Толмачево»','2022-06-01 12:00:00'),

('6','4-я Международная конференция','Информационные технологии и безопасные трансакции','Университет Ист-Лондона','2022-04-25 15:00:00'),

('4','10-я Международная научно-практическая конференция','Вопросы трансформации образования','SCOPUS','2022-08-03 10:00:00'),

('4','10-я Ежегодная международная конференция','Наука и технологии','SCOPUS','2022-05-30 13:30:00'),

('8','VII Международная научно-практическая конференция','Экономические и социальные тренды устойчивого развития современного общества','ГУАП','2022-05-30 13:30:00'),

('1','10-я Ежегодная международная конференция','AGRITECH-VII – 2022: Агротехнологии, экологический инжиниринг и устойчивое развитие','Красноярский краевой Дом науки и техники','2022-07-30 10:00:00')

('1','III Всероссийская научно-практическая конференция с международным участием','Социальные и гуманитарные науки в XXI веке: итоги, вызовы, перспективы','Научно-исследовательский центр «Пересвет»','2022-09-30 16:00:00'),

('4','I Международная научно-практическая конференция-семинар','«Актуальные вопросы современного здравоохранения. Электронная медицина»','Университет им. Д. Бен гуриона','2022-10-30 12:00:00')

INSERT INTO "Participations" (scientist\_id, conference\_id, type\_id, topic, duration) VALUES

('8','9','1','Вакцинация населения. Значение вакцинации для жизни отдельного человека и человечества в целом','00:10:00'),

('7','9','1','Микроэлементы в человеческой жизни, их основная роль','00:10:00'),

('3','9','2','Бесплодные пары: ЭКО как решение проблемы на современном этапе жизни','00:10:00'),

('12','9','3',NULL,'00:10:00'),

('13','4','1','Современное географическое образование','00:10:00'),

('4','4','1','История региональной архитектуры','00:10:00'),

('6','4','1','Языковое сознание: формирование и функционирование','00:10:00'),

('1','2','1','Перспективы освоения космического пространства','00:10:00'),

('5','2','2','Средства защиты отчуждаемых носителей информации','00:10:00'),

('9','1','1','Методики системного анализа и бизнес-аналитики','00:10:00'),

('10','1','2','Современные физические методы применяемые в онкологии','00:10:00'),

('11','3','3',NULL,'00:10:00')

('8','9','1','Анализ частоты и причин отказа родителей от проведения вакцинации','00:12:00'),

('9','1','1','Система как объект системного анализа','00:10:00'),

('5','2','1','Источники возникновения и последствия реализации угроз информационной безопасности','00:08:00'),

('5','2','1','Защита информации от несанкционированного доступа методом криптопреобразования','00:12:00'),

('5','1','1','Защита информации от несанкционированного доступа методом криптопреобразования','00:12:00')

Вывод

В ходе выполнения практического задания №1.2 в среде pgAmdin:

* разработана база данных «Научная конференция»,
* созданы таблицы согласно схеме, полученной в предыдущем задании,
* созданы скрипты для заполнения таблиц приготовленными данными.

# **Практическое задание №1.3 Операторы манипулирования**

Постановка задачи

Практическое задание посвящено манипулированию данными с помощью

операторов SQL. Нужно подготовить 3-4 выборки, которые имеют осмысленное значение для предметной области, и также составить для них SQL-скрипты. Сформулировать 3-4 запроса на изменение и удаление из базы данных. Запросы должны быть сформулированы в терминах предметной области. Составить SQL-скрипты для выполнения этих запросов.

Ход работы

1. Были составлены следующие выборки с соответствующими SQL-скриптами:

* Список конференций в хронологическом порядке:

SELECT

TO\_CHAR("Conferences".date, 'DD-MM-YYYY') AS date,

"Conferences".title,

"Conferences".theme,

"Conferences".place,

"Countries".name AS "country"

FROM "Conferences", "Countries" WHERE "Countries".country\_id = "Conferences".country\_id

ORDER BY "Conferences".date;

Результат:

Изображение выглядит как текст

Автоматически созданное описание

* Список всех ученых:

SELECT

"Persons".first\_name,

"Persons".second\_name,

"Degrees".name AS degree,

"Specializations".name AS "specialization",

"Countries".name AS "country"

FROM "Scientists"

JOIN "Persons" ON "Persons".person\_id = "Scientists".person\_id

JOIN "Degrees" ON "Degrees".degree\_id = "Scientists".degree\_id

JOIN "Countries" ON "Countries".country\_id = "Scientists".country\_id

JOIN "Specializations" ON "Specializations".specialization\_id = "Scientists".specialization\_id

Результат:

Изображение выглядит как текст

Автоматически созданное описание

* Список участников выбранной научной конференций:

SELECT "Persons".first\_name,

"Persons".second\_name,

"Types".name AS Type,

"Participations".topic,

TO\_CHAR("Participations".duration, 'MI:SS')

FROM "Participations"

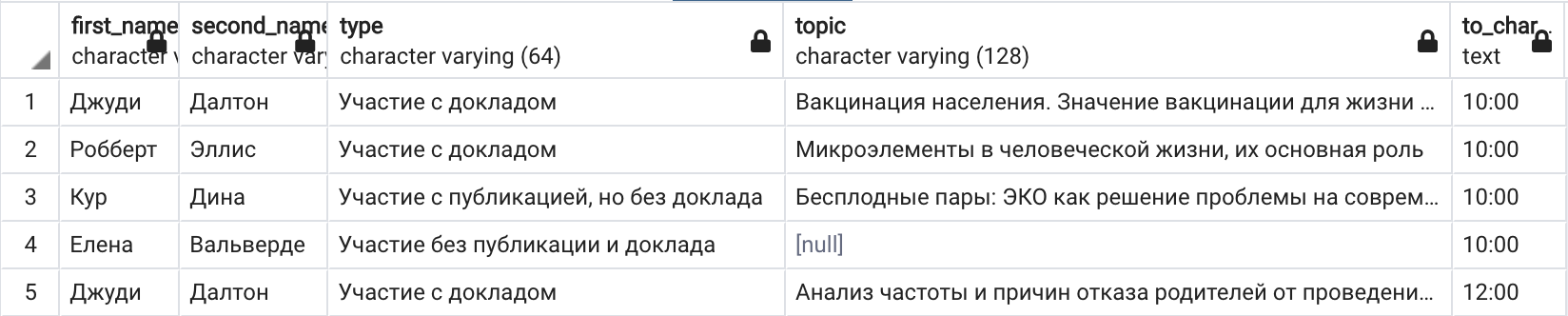
JOIN "Conferences" ON "Conferences".conference\_id = "Participations".conference\_id AND "Conferences".conference\_id = 9

JOIN "Scientists" ON "Scientists".scientist\_id = "Participations".scientist\_id

JOIN "Persons" ON "Persons".person\_id = "Scientists".person\_id

JOIN "Types" ON "Types".type\_id = "Participations".type\_id

Результат:



* Список научных конференций с количеством участников по убыванию:

(SELECT

"Conferences".title,

COUNT("Participations".scientist\_id) as number\_of\_scientists

FROM "Participations"

JOIN "Conferences" ON "Conferences".conference\_id = "Participations".conference\_id

GROUP BY "Conferences".title ORDER BY COUNT("Participations".scientist\_id) DESC)

UNION ALL

(SELECT

"Conferences".title, 0 AS "number\_of\_scientists"

FROM "Conferences"

WHERE NOT EXISTS (SELECT \* FROM "Participations" WHERE "Conferences".conference\_id = "Participations".conference\_id))

Результат:

Изображение выглядит как текст, стол

Автоматически созданное описание

* Ученые, которые докладывали в одной конференции более 1 раза:

SELECT

"Persons".second\_name,

COUNT("Participations".conference\_id),

"Conferences".title

FROM "Participations"

JOIN "Scientists" ON "Scientists".scientist\_id = "Participations".scientist\_id

JOIN "Persons" ON "Persons".person\_id = "Scientists".person\_id

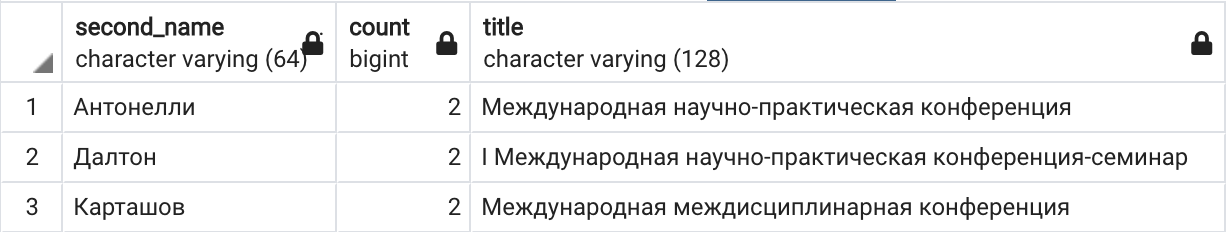
JOIN "Conferences" ON "Conferences".conference\_id = "Participations".conference\_id

JOIN "Types" ON "Types".type\_id = "Participations".type\_id AND "Types".name = 'Участие с докладом'

GROUP BY "Persons".second\_name,"Conferences".title

HAVING COUNT("Participations".conference\_id) > 1

Результат:



* Фамилии ученых за этот год, которые приняли участие в 2 международных конференциях:

SELECT

"Persons".second\_name,

COUNT(DISTINCT "Conferences".conference\_id)

FROM "Scientists"

JOIN "Persons" ON "Persons".person\_id = "Scientists".person\_id

JOIN "Participations" ON "Participations".scientist\_id = "Scientists".scientist\_id

JOIN "Conferences" ON "Conferences".conference\_id = "Participations".conference\_id

AND TO\_CHAR("Conferences".date, 'YYYY') = '2022' AND "Conferences".title ~ '(международная|Международная)'

GROUP BY "Persons".second\_name

HAVING COUNT(DISTINCT "Conferences".conference\_id) = 2

Результат:

Изображение выглядит как текст

Автоматически созданное описание

* Количество конференций по странам (оконная функция):

SELECT

"Conferences".title,

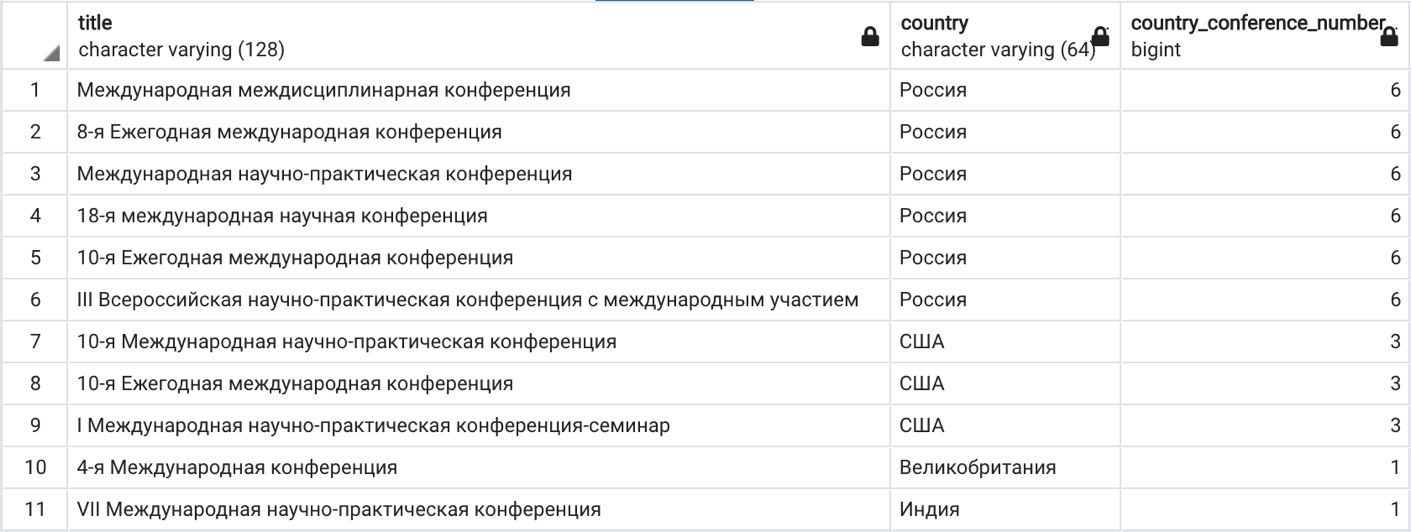
"Countries".name AS country,

COUNT(\*) OVER (PARTITION BY "Conferences".country\_id) AS country\_conference\_number

FROM "Conferences"

JOIN "Countries" ON "Countries".country\_id = "Conferences".country\_id

Результат:



* Нумерованный список конференций по странам по дате (оконная функция):

SELECT

ROW\_NUMBER() OVER (PARTITION BY "Conferences".country\_id ORDER BY "Conferences".date ASC) AS row\_number,

"Conferences".date,

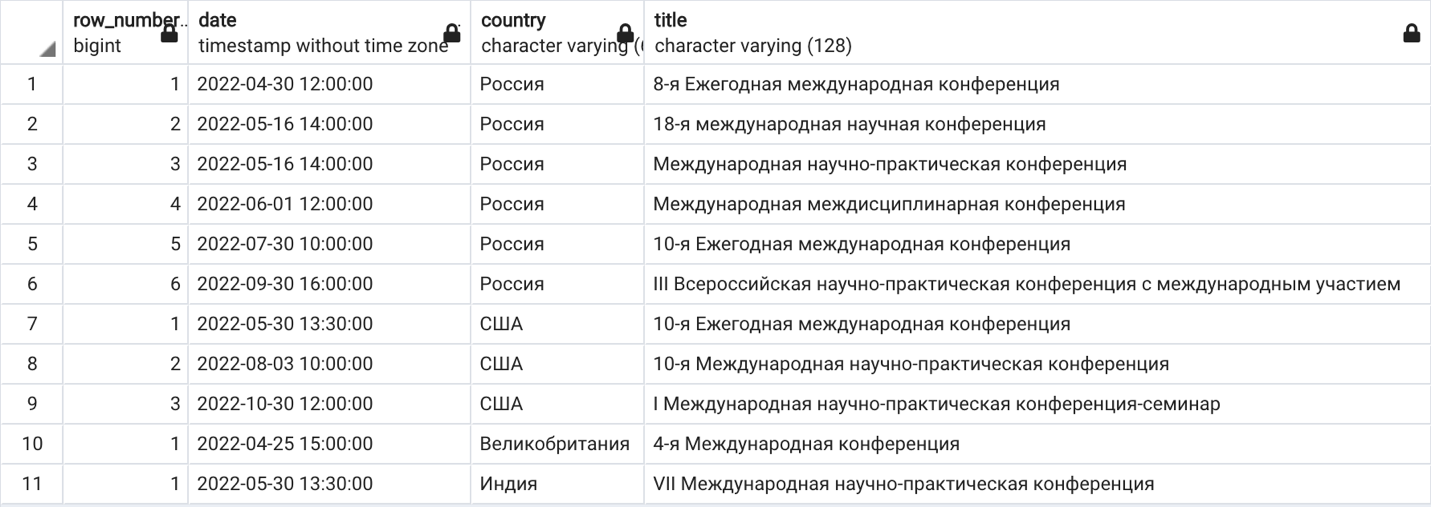
"Countries".name AS country,

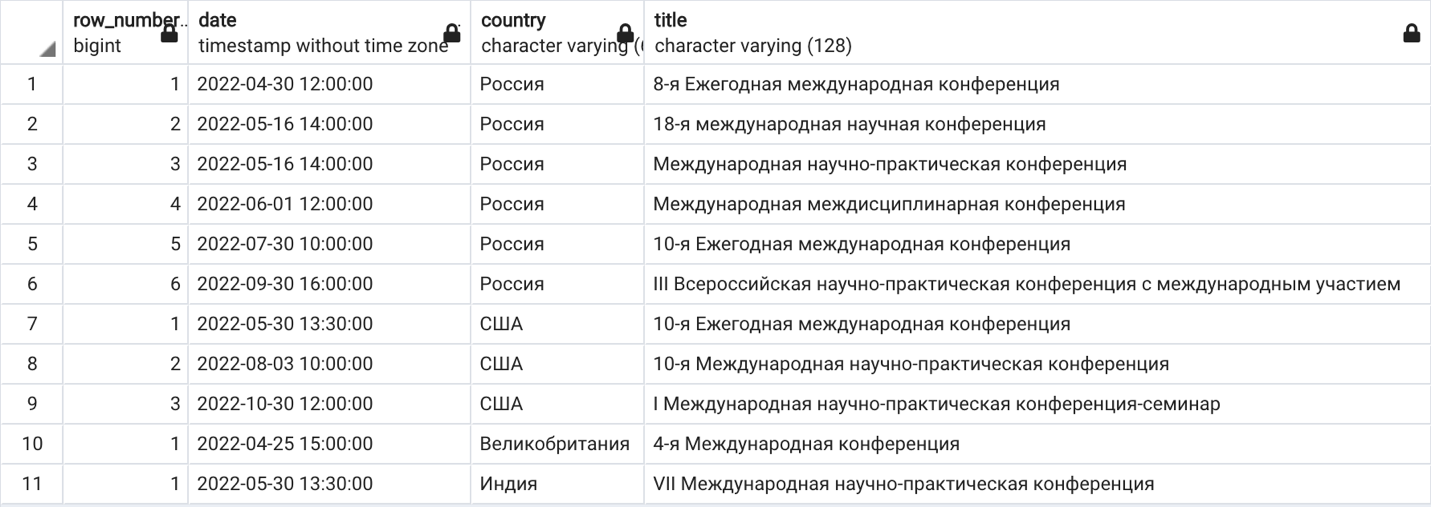
"Conferences".title

FROM "Conferences"

JOIN "Countries" ON "Countries".country\_id = "Conferences".country\_id

Результат:





2. Запросы на удаление/изменение с соответствующими SQL-скриптами:

* Изменение даты проведения научной конференции:

UPDATE "Conferences"

SET date = '2022-11-05 10:00:00'

WHERE "Conferences".title = 'I Международная научно-практическая конференция-семинар'

AND "Conferences".theme = '«Актуальные вопросы современного здравоохранения. Электронная медицина»'

* Изменение научной степени выбранного ученого:

UPDATE "Scientists"

SET degree\_id = (SELECT "Degrees".degree\_id FROM "Degrees" WHERE "Degrees".name = 'Кандидат архитектурных наук')

WHERE "Scientists".person\_id = (SELECT "Persons".person\_id FROM "Persons" WHERE "Persons".first\_name = 'Бруно' AND "Persons".second\_name = 'Перес')

* Удаление конференции:

DELETE FROM "Conferences"

WHERE title = 'Тест'

AND theme = 'Big Data Days 2022'

# **Практическое задание №1.4 Контроль целостности данных**

Постановка задачи

Практическое задание посвящено контролю целостности данных, который

производится с помощью механизма транзакций и триггеров.

Необходимо подготовить SQL-скрипты для проверки наличия аномалий (потерянных изменений, грязных чтений, неповторяющихся чтений, фантомов) при параллельном исполнении транзакций на различных уровнях изолированности SQL/92 (READ UNCOMMITTED, READ COMMITTED, REPEATABLE READ, SERIALIZABLE). Для проверки наличия аномалий потребуются два параллельных сеанса, операторы в которых выполняются пошагово. Необходимо составить скрипт для создания триггера, а также подготовить несколько запросов для проверки и демонстрации его полезных свойств.

Ход работы

**Уровни изоляции и аномалии**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Уровень изоляции | Потерянные изменения | Грязное чтение | Неповторяемое чтение | Фантомное чтение | Аномалия сериализации |
| Read uncommitted |  |  |  |  |  |
| Read committed |  |  |  |  |  |
| Repeatable read |  |  |  |  |  |
| Serializable |  |  |  |  |  |

Возможно –

Невозможно –

Допускается, но не в PG –­

**Скрипты для проверки**

1. Запрос на выборку. Дата конференции с id = 1:

SELECT "Conferences".date

FROM "Conferences"

WHERE conference\_id = 1

2. Запрос на изменение даты конференции с id = 1:

UPDATE "Conferences" SET date = date + interval '1 hour'

WHERE conference\_id = 1

**Проверка наличия аномалий**

1. Установить в обоих сеансах уровень изоляции READ UNCOMMITTED. Выполнить сценарии проверки наличия аномалий потерянных изменений и грязных чтений.

|  |  |  |  |
| --- | --- | --- | --- |
| **Потерянных изменений нет** | | **Date** | |
| BEGIN TRANSACTION ISOLATION LEVEL  READ UNCOMMITTED; | BEGIN TRANSACTION ISOLATION LEVEL  READ UNCOMMITTED; |  |  |
| SELECT "Conferences".date  FROM "Conferences"  WHERE conference\_id = 1 |  | 14:00:00 |  |
|  | SELECT "Conferences".date  FROM "Conferences"  WHERE conference\_id = 1 |  | 14:00:00 |
| UPDATE "Conferences" SET date = date + interval '1 hour'  WHERE conference\_id = 1 |  | 15:00:00 |  |
| COMMIT |  |  |  |
|  | UPDATE "Conferences" SET date = date + interval '1 hour'  WHERE conference\_id = 1 |  | 16:00:00 |
|  | COMMIT |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Грязных чтений нет** | | **Date** | |
| BEGIN TRANSACTION ISOLATION LEVEL  READ UNCOMMITTED; | BEGIN TRANSACTION ISOLATION LEVEL  READ UNCOMMITTED; |  |  |
| SELECT "Conferences".date  FROM "Conferences"  WHERE conference\_id = 1 |  | 16:00:00 |  |
|  | UPDATE "Conferences" SET date = date + interval '1 hour'  WHERE conference\_id = 1 |  | 17:00:00 |
| SELECT "Conferences".date  FROM "Conferences"  WHERE conference\_id = 1 |  | 16:00:00 |  |
| COMMIT | COMMIT |  |  |

1. Установить в обоих сеансах уровень изоляции READ COMMITTED. Выполнить сценарии проверки наличия аномалий грязных чтений и неповторяющихся чтений.

|  |  |  |  |
| --- | --- | --- | --- |
| **Грязных чтений нет** | | **Date** | |
| BEGIN TRANSACTION ISOLATION LEVEL  READ COMMITTED; | BEGIN TRANSACTION ISOLATION LEVEL  READ COMMITTED; |  |  |
|  | SELECT "Conferences".date  FROM "Conferences"  WHERE conference\_id = 1 |  | 17:00:00 |
| UPDATE "Conferences" SET date = date + interval '1 hour'  WHERE conference\_id = 1 |  | 18:00:00 |  |
|  | SELECT "Conferences".date  FROM "Conferences"  WHERE conference\_id = 1 |  | 17:00:00 |
| COMMIT | COMMIT |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Неповторяющиеся чтения есть** | | **Date** | |
| BEGIN TRANSACTION ISOLATION LEVEL  READ COMMITTED; | BEGIN TRANSACTION ISOLATION LEVEL  READ COMMITTED; |  |  |
|  | SELECT "Conferences".date  FROM "Conferences"  WHERE conference\_id = 1 |  | 18:00:00 |
| SELECT "Conferences".date  FROM "Conferences"  WHERE conference\_id = 1 |  | 18:00:00 |  |
| UPDATE "Conferences" SET date = date + interval '1 hour'  WHERE conference\_id = 1 |  | 19:00:00 |  |
| COMMIT |  |  |  |
|  | SELECT "Conferences".date  FROM "Conferences"  WHERE conference\_id = 1 |  | 19:00:00 |
|  | COMMIT |  |  |

1. Установить в обоих сеансах уровень изоляции REPEATABLE READ. Выполнить сценарии проверки наличия аномалий неповторяющихся чтений и фантомов.

|  |  |  |  |
| --- | --- | --- | --- |
| **Неповторяющихся чтений нет** | | **Date** | |
| BEGIN TRANSACTION ISOLATION LEVEL  REPEATABLE READ; | BEGIN TRANSACTION ISOLATION LEVEL  REPEATABLE READ; |  |  |
|  | SELECT "Conferences".date  FROM "Conferences"  WHERE conference\_id = 1 |  | 19:00:00 |
| UPDATE "Conferences" SET date = date + interval '1 hour'  WHERE conference\_id = 1 |  | 20:00:00 |  |
| COMMIT |  |  |  |
|  | SELECT "Conferences".date  FROM "Conferences"  WHERE conference\_id = 1 |  | 19:00:00 |
|  | COMMIT |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Фантомного чтения нет** | | **Date** | |
| BEGIN TRANSACTION ISOLATION LEVEL  REPEATABLE READ; | BEGIN TRANSACTION ISOLATION LEVEL  REPEATABLE READ; |  |  |
|  | SELECT "Conferences".date  FROM "Conferences"  WHERE conference\_id = 1 |  | 19:00:00 |
| UPDATE "Conferences" SET date = date + interval '1 hour'  WHERE conference\_id = 1 |  |  |  |
| COMMIT |  |  |  |
|  | SELECT "Conferences".date  FROM "Conferences"  WHERE conference\_id = 1 |  | 19:00:00 |
|  | COMMIT |  |  |

1. Установить в обоих сеансах уровень изоляции SERIALIZABLE. Выполнить сценарий проверки наличия фантомов.

|  |  |  |  |
| --- | --- | --- | --- |
| **Фантомного чтения нет** | | **Date** | |
| BEGIN TRANSACTION ISOLATION LEVEL  SERIALIZABLE; | BEGIN TRANSACTION ISOLATION LEVEL  SERIALIZABLE; |  |  |
|  | SELECT "Conferences".date  FROM "Conferences"  WHERE conference\_id = 1 |  | 19:00:00 |
| UPDATE "Conferences" SET date = date + interval '1 hour'  WHERE conference\_id = 1 |  |  |  |
| COMMIT |  |  |  |
|  | SELECT "Conferences".date  FROM "Conferences"  WHERE conference\_id = 1 |  | 19:00:00 |
|  | COMMIT |  |  |

|  |  |
| --- | --- |
| **Аномалия сериализации** | |
| BEGIN TRANSACTION ISOLATION LEVEL  SERIALIZABLE; | BEGIN TRANSACTION ISOLATION LEVEL  SERIALIZABLE; |
| UPDATE "Conferences" SET date = date + interval '1 hour'  WHERE conference\_id = 1 |  |
|  | UPDATE "Conferences" SET date = date + interval '1 hour'  WHERE conference\_id = 2 |
| COMMIT |  |
|  | COMMIT |
|  | [40001] ОШИБКА: не удалось сериализовать доступ из-за зависимостей чтения/записи между транзакциями |

**Триггер**

* Скрипт:

CREATE OR REPLACE FUNCTION check\_conference\_date()

RETURNS TRIGGER AS $$

BEGIN

IF NEW.date <= now()

THEN

RAISE EXCEPTION 'The date of the conference must be greater than the current';

END IF;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER check\_conference\_date\_trigger

AFTER INSERT ON "Conferences"

FOR EACH ROW EXECUTE PROCEDURE check\_conference\_date()

* Запросы для проверки и демонстрации полезных свойств:

-- ОШИБКА (‘The date of the conference must be greater than the current’)

INSERT INTO "Conferences" (country\_id, title, theme, place, date)

VALUES ('1','18-я международная научная конференция','Advances in Biology','СПБГУ','2022-03-16 14:00:00')

-- ВЫПОЛНИЛСЯ (в таблицу ‘Conferences’ добавлена новая конференция)

INSERT INTO "Conferences" (country\_id, title, theme, place, date)

VALUES ('1','18-я международная научная конференция','Advances in Biology','СПБГУ','2022-05-16 14:00:00')

# **Практическое задание №2.1 Управление доступом**

Постановка задачи

Целью практического задания является освоение работы с представлениями и другими способами управления доступом.

Ход работы

1. Создать пользователя test и выдать ему доступ к базе данных.

CREATE USER test WITH PASSWORD '123';

GRANT ALL PRIVILEGES ON DATABASE scientific\_conferences TO test;

1. По крайней мере, для одной таблицы новому пользователю присваиваются права SELECT, INSERT, UPDATE в полном объеме.

GRANT SELECT, INSERT, UPDATE ON TABLE "Conferences" TO test;

1. По крайней мере, для одной таблицы новому пользователю присваиваются права SELECT и UPDATE только избранных столбцов.

GRANT SELECT(scientist\_id, person\_id, country\_id),

UPDATE(organization\_id, specialization\_id, degree\_id)

ON TABLE "Scientists" TO test;

1. По крайней мере, для одной таблицы новому пользователю присваивается только право SELECT.

GRANT SELECT ON TABLE "Persons" TO test;

GRANT SELECT ON TABLE "Participations" TO test;

GRANT SELECT ON TABLE "Countries" TO test;

1. Составить SQL-скрипты для создания нескольких представлений.

* Представление с учеными и количеством выступлений(участий) на конференциях

CREATE VIEW Scientists\_Number\_Of\_Participations AS (

SELECT

"Persons".first\_name,

"Persons".second\_name,

"Countries".name AS country,

COUNT(distinct "Participations".participation\_id) AS paricipation\_number

FROM "Participations"

JOIN "Conferences" ON "Conferences".conference\_id = "Participations".conference\_id

JOIN "Scientists" ON "Scientists".scientist\_id = "Participations".scientist\_id

JOIN "Persons" ON "Persons".person\_id = "Scientists".person\_id

JOIN "Countries" ON "Countries".country\_id = "Scientists".country\_id

GROUP BY "Persons".first\_name, "Persons".second\_name, "Countries".name

ORDER BY paricipation\_number desc

);

* Представление со странами по убыванию количества проводимых конференций

CREATE VIEW Сountries\_By\_Number\_Of\_Conferences AS (

(SELECT

"Countries".name AS country,

COUNT("Conferences".country\_id) AS conferences\_num

FROM "Conferences"

JOIN "Countries" ON "Countries".country\_id = "Conferences".country\_id

GROUP BY "Countries".name ORDER BY COUNT("Conferences".country\_id) DESC)

UNION ALL

(SELECT

"Countries".name AS country, 0

FROM "Countries"

WHERE NOT EXISTS (SELECT \* FROM "Conferences"

WHERE "Countries".country\_id = "Conferences".country\_id))

);

1. Присвоить новому пользователю право доступа (SELECT) к одному из представлений

GRANT SELECT ON Scientists\_Number\_Of\_Participations TO test;

1. Создать стандартную роль уровня базы данных.

CREATE ROLE test\_role;

1. Присвоить ей право доступа (UPDATE на некоторые столбцы) к одному из представлений.

GRANT UPDATE(country) ON Сountries\_By\_Number\_Of\_Conferences TO test\_role;

1. Назначить новому пользователю созданную роль.

GRANT test\_role TO test;

**Проверка доступа**

|  |  |  |  |
| --- | --- | --- | --- |
| **Таблица** | **Разрешен доступ** | **Запрос** | **Результат** |
| Conferences | SELECT  UPDATE  INSERT | SELECT \* FROM "Conferences" WHERE "Conferences".conference\_id = 11 |  |
| UPDATE "Conferences"  SET date = date + interval '1 hour' WHERE "Conferences".conference\_id = 11 |  |
| INSERT INTO "Conferences" (country\_id, title, theme, place, date) VALUES ('1','14-я Международная научная конференция','Advances in Biology','СПБГУ','2022-08-04 10:00:00') |  |
| Scientists | SELECT  (scientist\_id, person\_id, country\_id)  UPDATE  (organization\_id, specialization\_id, degree\_id) | SELECT scientist\_id, person\_id, country\_id FROM "Scientists"; |  |
| UPDATE "Scientists" SET organization\_id = 1 WHERE scientist\_id = 1; |  |
| SELECT \* FROM "Scientists"  UPDATE "Scientists" SET country\_id = 2 WHERE scientist\_id = 1; |  |
| Persons | SELECT | SELECT \* FROM "Persons" |  |
| UPDATE "Persons" SET first\_name = 'Иван' WHERE "Persons".person\_id = 1 |  |
| INSERT INTO "Persons" (first\_name, second\_name, middle\_name) VALUES ('Евгений','Панарин','Владимирович') |  |
| Scientists\_  Number\_  Of\_  Participations | SELECT | SELECT \* FROM Scientists\_Number\_Of\_Participations |  |
| Сountries\_  By\_  Number\_  Of\_  Conferences | − | SELECT \* FROM Сountries\_By\_Number\_Of\_Conferences |  |

Ошибка доступа –

Запрос выполняется –

**Вывод макрокоманды \dp**

# **Практическое задание №2.2 Функции и язык PL/pgSQL**

Постановка задачи

Практическое задание посвящено упрощению работы с помощью создания и

использования функций. При выполнении задания необходимо:

* Составить SQL-скрипты для создания нескольких (2–3) функций, упрощающих работу с данными.
* Продемонстрировать полученные знания о возможностях языка PL/pgSQL. В скриптах должны использоваться:
* Циклы.
* Переменные.
* Курсоры.
* Ветвления.
* Исключения.
* Обосновать преимущества механизма функций перед механизмом представлений.

Ход работы

1. Скрипт на добавление конференции с использованием *ветвления* и *исключений*:

CREATE OR REPLACE FUNCTION add\_new\_сonference(country\_id INT, title VARCHAR(128), theme VARCHAR(128), place VARCHAR(128), date TIMESTAMP)

RETURNS VOID AS $$

BEGIN

IF (date <= now())

THEN

RAISE EXCEPTION 'The date of the conference must be greater than the current';

ELSIF (SELECT EXISTS(SELECT \* FROM "Countries" WHERE "Countries".country\_id = add\_new\_сonference.country\_id))

THEN

INSERT INTO "Conferences" (country\_id, title, theme, place, date)

VALUES (add\_new\_сonference.country\_id, add\_new\_сonference.title, add\_new\_сonference.theme, add\_new\_сonference.place, add\_new\_сonference.date);

ELSE

RAISE EXCEPTION 'Country pk does not exist';

END IF;

END;

$$ LANGUAGE plpgsql;

1. Скрипт на поиск конференций по странам:

CREATE OR REPLACE FUNCTION search\_country(country\_name VARCHAR(64))

RETURNS TABLE(

conference\_title VARCHAR(128),

conference\_theme VARCHAR(128),

conference\_place VARCHAR(128),

conference\_date TIMESTAMP) AS

$$

BEGIN

RETURN QUERY(SELECT

"Conferences".title,

"Conferences".theme,

"Conferences".place,

"Conferences".date

FROM "Conferences" JOIN "Countries" ON "Countries".country\_id = "Conferences".country\_id

AND "Countries".name ILIKE '%' || country\_name ||'%');

END;

$$ LANGUAGE plpgsql;

1. Скрипт на вывод количества конференций за определенный период с использованием *курсоров, переменных и циклов*:

CREATE OR REPLACE FUNCTION conferences\_num\_for\_selected\_period (start\_period TIMESTAMP, end\_period TIMESTAMP)

RETURNS INT AS

$$

DECLARE

cur CURSOR FOR SELECT "Conferences".date FROM "Conferences";

result\_conferences\_num INT;

BEGIN

result\_conferences\_num = 0;

FOR rec IN cur

LOOP

IF ((rec.date >= start\_period) AND (rec.date <= end\_period))

THEN

result\_conferences\_num = result\_conferences\_num + 1;

END IF;

END LOOP;

RETURN result\_conferences\_num;

END;

$$ LANGUAGE plpgsql;