

Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное образовательное учреждение высшего
образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Отчет

по лабораторной работе №5 «процедуры, функции, триггеры в PostgreSQL»

по дисциплине «Проектирование и реализация баз данных»

Автор: Кузнецов А. Г.

Факультет: ИКТ

Группа: К3240

Преподаватель: Говорова М.М.



Санкт-Петербург 2023

Оглавление

Цель работы	3
Практическое задание	3
Выполнение	4
Создание хранимых процедур:	4
1. Для повышения стипендии отличникам на 10%.	4
2. Для перевода студентов на следующий курс.	5
3. Для изменения оценки при успешной пересдаче экзамена.....	6
Модифицировать триггер.....	7
Вывод.....	8

Цель работы

Овладеть практическими навыками создания представлений и запросов на выборку данных к базе данных PostgreSQL, использования подзапросов при модификации данных и индексов.

Практическое задание

1. Создать процедуры/функции согласно индивидуальному заданию (часть 4).
2. Модифицировать триггер (триггерную функцию) на проверку корректности входа и выхода сотрудника (см. Практическое задание 1 Лабораторного практикума (Приложение)) с максимальным учетом «узких» мест некорректных данных по входу и выходу).

Выполнение

Вариант 12. БД «Сессия»

Создание хранимых процедур:

1. Для повышения стипендии отличникам на 10%.

```
postgres=# CREATE OR REPLACE PROCEDURE increase_scholarship_for_excellent_students()
postgres=# LANGUAGE plpgsql
postgres=# AS $$
postgres=# BEGIN
postgres=#     UPDATE "Session".scholarship
postgres=#     SET sum_of_scholarship = sum_of_scholarship * 1.1
postgres=#     WHERE scholarship_id IN (
postgres=#         SELECT scholarship_id
postgres=#         FROM "Session".scholarship
postgres=#         JOIN "Session".attestation ON "Session".scholarship.studying_student_id = "Session".attestation.studying_student_id
postgres=#         JOIN "Session"."studying student" ON "Session".scholarship.studying_student_id = "Session"."studying student".studying_student_id
postgres=#         WHERE "Session".attestation.mark = 'A'
postgres=#         AND (
postgres=#             (
postgres=#                 EXTRACT(MONTH FROM CURRENT_DATE) = 12
postgres=#                 AND (
postgres=#                     (EXTRACT(MONTH FROM "Session".attestation.attestation_date) = 1 AND EXTRACT(YEAR FROM "Session".attestation.attestation_date) = EXTRACT(YEAR FROM CURRENT_DATE) + 1)
postgres=#                     OR
postgres=#                     (EXTRACT(MONTH FROM "Session".attestation.attestation_date) = 12 AND EXTRACT(YEAR FROM "Session".attestation.attestation_date) = EXTRACT(YEAR FROM CURRENT_DATE))
postgres=#                 )
postgres=#             )
postgres=#             OR (
postgres=#                 EXTRACT(MONTH FROM CURRENT_DATE) BETWEEN 1 AND 4
postgres=#                 AND (
postgres=#                     (EXTRACT(MONTH FROM "Session".attestation.attestation_date) = 12 AND EXTRACT(YEAR FROM "Session".attestation.attestation_date) = EXTRACT(YEAR FROM CURRENT_DATE) - 1)
postgres=#                     OR
postgres=#                     (EXTRACT(MONTH FROM "Session".attestation.attestation_date) = 1 AND EXTRACT(YEAR FROM "Session".attestation.attestation_date) = EXTRACT(YEAR FROM CURRENT_DATE))
postgres=#                 )
postgres=#             )
postgres=#             OR (
postgres=#                 EXTRACT(MONTH FROM CURRENT_DATE) BETWEEN 5 AND 11
postgres=#                 AND EXTRACT(MONTH FROM "Session".attestation.attestation_date) BETWEEN 5 AND 6
postgres=#                 AND EXTRACT(YEAR FROM "Session".attestation.attestation_date) = EXTRACT(YEAR FROM CURRENT_DATE)
postgres=#             )
postgres=#         )
postgres=#     );
postgres=# END;
postgres=# $$;
CREATE PROCEDURE
```

```
postgres=# SELECT * FROM "Session".scholarship;
scholarship_id | scholarship_type_id | payment_start | payment_end | studying_student_id | sum_of_scholarship
-----|-----|-----|-----|-----|-----
141 | 453 | 2023-01-01 | 2024-06-01 | 648116 | 15000
142 | 453 | 2023-01-01 | 2024-06-01 | 977888 | 15000
179 | 94 | 2023-01-01 | 2024-06-01 | 254999 | 16500
180 | 94 | 2023-01-01 | 2024-06-01 | 825344 | 16500
337 | 108 | 2023-01-01 | 2024-06-01 | 460221 | 13500
338 | 108 | 2023-01-01 | 2024-06-01 | 887161 | 13500
413 | 349 | 2023-01-01 | 2024-06-01 | 582963 | 4000
414 | 349 | 2023-01-01 | 2024-06-01 | 916338 | 4000
439 | 625 | 2023-01-01 | 2024-06-01 | 749222 | 25000
440 | 625 | 2023-01-01 | 2024-06-01 | 977891 | 25000
482 | 96 | 2023-01-01 | 2024-06-01 | 256067 | 7500
483 | 96 | 2023-01-01 | 2024-06-01 | 828854 | 7500
587 | 447 | 2023-01-01 | 2024-06-01 | 633101 | 20000
588 | 447 | 2023-01-01 | 2024-06-01 | 931089 | 20000
621 | 66 | 2023-01-01 | 2024-06-01 | 106586 | 7000
622 | 66 | 2023-01-01 | 2024-06-01 | 820857 | 7000
678 | 541 | 2023-01-01 | 2024-06-01 | 674839 | 18000
679 | 541 | 2023-01-01 | 2024-06-01 | 977890 | 18000
(18 строк)
```

```
postgres=# CALL increase_scholarship_for_excellent_students();
CALL
postgres=# SELECT * FROM "Session".scholarship;
scholarship_id | scholarship_type_id | payment_start | payment_end | studying_student_id | sum_of_scholarship
-----|-----|-----|-----|-----|-----
141 | 453 | 2023-01-01 | 2024-06-01 | 648116 | 16500
142 | 453 | 2023-01-01 | 2024-06-01 | 977888 | 16500
179 | 94 | 2023-01-01 | 2024-06-01 | 254999 | 18150
180 | 94 | 2023-01-01 | 2024-06-01 | 825344 | 18150
337 | 108 | 2023-01-01 | 2024-06-01 | 460221 | 14850
338 | 108 | 2023-01-01 | 2024-06-01 | 887161 | 14850
413 | 349 | 2023-01-01 | 2024-06-01 | 582963 | 4400
414 | 349 | 2023-01-01 | 2024-06-01 | 916338 | 4400
439 | 625 | 2023-01-01 | 2024-06-01 | 749222 | 27500
440 | 625 | 2023-01-01 | 2024-06-01 | 977891 | 27500
482 | 96 | 2023-01-01 | 2024-06-01 | 256067 | 8250
483 | 96 | 2023-01-01 | 2024-06-01 | 828854 | 8250
587 | 447 | 2023-01-01 | 2024-06-01 | 633101 | 22000
588 | 447 | 2023-01-01 | 2024-06-01 | 931089 | 22000
621 | 66 | 2023-01-01 | 2024-06-01 | 106586 | 7700
622 | 66 | 2023-01-01 | 2024-06-01 | 820857 | 7700
678 | 541 | 2023-01-01 | 2024-06-01 | 674839 | 19800
679 | 541 | 2023-01-01 | 2024-06-01 | 977890 | 19800
(18 строк)
```

2. Для перевода студентов на следующий курс.

```

postgres=# CREATE OR REPLACE PROCEDURE update_students_courses()
postgres=# LANGUAGE plpgsql
postgres=# AS $$
postgres$# DECLARE
postgres$#     student_record RECORD;
postgres$#     max_bachelor_course INTEGER := 4;
postgres$#     max_specialist_course INTEGER := 5;
postgres$# BEGIN
postgres$#     FOR student_record IN
postgres$#         SELECT ss.studying_student_id, s.course, s.grade_book_id, d.level_of_edu
postgres$#         FROM "Session"."studying student" ss
postgres$#         JOIN "Session"."student s ON ss.grade_book_id = s.grade_book_id
postgres$#         JOIN "Session"."group" g ON ss.group_id = g.group_id
postgres$#         JOIN "Session".syllabus sy ON g.syllabus_id = sy.syllabus_id
postgres$#         JOIN "Session"."educational program" ep ON sy.educational_program_id = ep.educational_program_id
postgres$#         JOIN "Session".direction d ON ep.direction_id = d.direction_id
postgres$#     LOOP
postgres$#         IF student_record.level_of_edu = 'Бакалавриат' THEN
postgres$#             IF student_record.course = max_bachelor_course THEN
postgres$#                 UPDATE "Session"."studying student"
postgres$#                 SET status = 'Выпускник'
postgres$#                 WHERE studying_student_id = student_record.studying_student_id;
postgres$#             ELSE
postgres$#                 UPDATE "Session".student
postgres$#                 SET course = student_record.course + 1
postgres$#                 WHERE grade_book_id = student_record.grade_book_id;
postgres$#             END IF;
postgres$#         ELSIF student_record.level_of_edu = 'Специалитет' THEN
postgres$#             IF student_record.course = max_specialist_course THEN
postgres$#                 UPDATE "Session"."studying student"
postgres$#                 SET status = 'Выпускник'
postgres$#                 WHERE studying_student_id = student_record.studying_student_id;
postgres$#             ELSE
postgres$#                 UPDATE "Session".student
postgres$#                 SET course = student_record.course + 1
postgres$#                 WHERE grade_book_id = student_record.grade_book_id;
postgres$#             END IF;
postgres$#         END IF;
postgres$#     END LOOP;
postgres$# END;
postgres$# $$;
CREATE PROCEDURE

```

grade_book_id	course	status	level_of_edu
678309	4	Обучается	Бакалавриат
440852	4	Обучается	Бакалавриат
491383	4	Обучается	Бакалавриат
669646	4	Обучается	Бакалавриат
678308	4	Обучается	Бакалавриат
788348	4	Обучается	Бакалавриат
264242	4	Обучается	Бакалавриат
689103	4	Обучается	Бакалавриат
706941	4	Обучается	Бакалавриат
734628	4	Обучается	Бакалавриат
894009	4	Обучается	Бакалавриат
164932	4	Обучается	Бакалавриат
264241	4	Обучается	Бакалавриат
536283	4	Обучается	Бакалавриат
593772	4	Обучается	Бакалавриат
793979	4	Обучается	Бакалавриат
985250	4	Обучается	Бакалавриат
440854	4	Обучается	Бакалавриат
491385	4	Обучается	Бакалавриат
494417	4	Обучается	Бакалавриат
536284	4	Обучается	Бакалавриат
604512	4	Обучается	Бакалавриат
985251	4	Обучается	Бакалавриат
604510	4	Обучается	Бакалавриат
793978	4	Обучается	Бакалавриат
823779	4	Обучается	Бакалавриат
440853	4	Обучается	Бакалавриат
734627	4	Обучается	Бакалавриат
491384	4	Обучается	Бакалавриат
604511	4	Обучается	Бакалавриат
689102	4	Обучается	Бакалавриат
823780	4	Обучается	Бакалавриат
164935	4	Обучается	Бакалавриат
182994	4	Обучается	Бакалавриат
593773	4	Обучается	Бакалавриат
602823	4	Обучается	Бакалавриат
793980	4	Обучается	Бакалавриат
440851	4	Обучается	Бакалавриат
536281	4	Обучается	Бакалавриат
678306	4	Обучается	Бакалавриат
793977	4	Обучается	Бакалавриат
264239	4	Обучается	Бакалавриат
491382	4	Обучается	Бакалавриат
602820	4	Обучается	Бакалавриат
604509	4	Обучается	Бакалавриат
669644	4	Обучается	Бакалавриат
689100	4	Обучается	Бакалавриат
706938	4	Обучается	Бакалавриат
985248	4	Обучается	Бакалавриат
602821	4	Обучается	Бакалавриат
669645	4	Обучается	Бакалавриат
678307	4	Обучается	Бакалавриат
706939	4	Обучается	Бакалавриат
788347	4	Обучается	Бакалавриат
985249	4	Обучается	Бакалавриат
823778	4	Обучается	Бакалавриат
894006	4	Обучается	Бакалавриат
164933	4	Обучается	Бакалавриат
264240	4	Обучается	Специалитет
494415	4	Обучается	Специалитет
689101	4	Обучается	Специалитет
164934	4	Обучается	Специалитет
494416	4	Обучается	Специалитет
706940	4	Обучается	Специалитет
734626	4	Обучается	Специалитет
182993	4	Обучается	Специалитет
536282	4	Обучается	Специалитет
894007	4	Обучается	Специалитет
602822	4	Обучается	Специалитет
894008	4	Обучается	Специалитет
182991	4	Обучается	Специалитет
593770	4	Обучается	Специалитет
734625	4	Обучается	Специалитет
788346	4	Обучается	Специалитет
593771	4	Обучается	Специалитет
823781	4	Обучается	Специалитет
669647	4	Обучается	Специалитет
788349	4	Обучается	Специалитет
182992	4	Обучается	Специалитет
494414	4	Обучается	Специалитет

(80 строк)

grade_book_id	course	status	level_of_edu
678309	4	Выпускник	Бакалавриат
440852	4	Выпускник	Бакалавриат
491383	4	Выпускник	Бакалавриат
669646	4	Выпускник	Бакалавриат
678308	4	Выпускник	Бакалавриат
788348	4	Выпускник	Бакалавриат
264242	4	Выпускник	Бакалавриат
689103	4	Выпускник	Бакалавриат
706941	4	Выпускник	Бакалавриат
734628	4	Выпускник	Бакалавриат
894009	4	Выпускник	Бакалавриат
164932	4	Выпускник	Бакалавриат
264241	4	Выпускник	Бакалавриат
536283	4	Выпускник	Бакалавриат
593772	4	Выпускник	Бакалавриат
793979	4	Выпускник	Бакалавриат
985250	4	Выпускник	Бакалавриат
440854	4	Выпускник	Бакалавриат
491385	4	Выпускник	Бакалавриат
494417	4	Выпускник	Бакалавриат
536284	4	Выпускник	Бакалавриат
604512	4	Выпускник	Бакалавриат
985251	4	Выпускник	Бакалавриат
604510	4	Выпускник	Бакалавриат
793978	4	Выпускник	Бакалавриат
823779	4	Выпускник	Бакалавриат
440853	4	Выпускник	Бакалавриат
734627	4	Выпускник	Бакалавриат
491384	4	Выпускник	Бакалавриат
604511	4	Выпускник	Бакалавриат
689102	4	Выпускник	Бакалавриат
823780	4	Выпускник	Бакалавриат
164935	4	Выпускник	Бакалавриат
182994	4	Выпускник	Бакалавриат
593773	4	Выпускник	Бакалавриат
602823	4	Выпускник	Бакалавриат
793980	4	Выпускник	Бакалавриат
440851	4	Выпускник	Бакалавриат
536281	4	Выпускник	Бакалавриат
678306	4	Выпускник	Бакалавриат
793977	4	Выпускник	Бакалавриат
264239	4	Выпускник	Бакалавриат
491382	4	Выпускник	Бакалавриат
602820	4	Выпускник	Бакалавриат
604509	4	Выпускник	Бакалавриат
669644	4	Выпускник	Бакалавриат
689100	4	Выпускник	Бакалавриат
706938	4	Выпускник	Бакалавриат
985248	4	Выпускник	Бакалавриат
602821	4	Выпускник	Бакалавриат
669645	4	Выпускник	Бакалавриат
678307	4	Выпускник	Бакалавриат
706939	4	Выпускник	Бакалавриат
788347	4	Выпускник	Бакалавриат
985249	4	Выпускник	Бакалавриат
823778	4	Выпускник	Бакалавриат
894006	4	Выпускник	Бакалавриат
164933	4	Выпускник	Бакалавриат
264240	5	Обучается	Специалитет
494415	5	Обучается	Специалитет
689101	5	Обучается	Специалитет
164934	5	Обучается	Специалитет
494416	5	Обучается	Специалитет
706940	5	Обучается	Специалитет
734626	5	Обучается	Специалитет
182993	5	Обучается	Специалитет
536282	5	Обучается	Специалитет
894007	5	Обучается	Специалитет
602822	5	Обучается	Специалитет
894008	5	Обучается	Специалитет
593770	5	Обучается	Специалитет
734625	5	Обучается	Специалитет
788346	5	Обучается	Специалитет
593771	5	Обучается	Специалитет
823781	5	Обучается	Специалитет
669647	5	Обучается	Специалитет
788349	5	Обучается	Специалитет
182991	5	Обучается	Специалитет
182992	5	Обучается	Специалитет
494414	5	Обучается	Специалитет

(80 строк)

3. Для изменения оценки при успешной пересдаче экзамена

```
postgres=# CREATE OR REPLACE PROCEDURE add_reattempt_attestation(
postgres(#   attestation_id_param integer,
postgres(#   new_mark_param character(1),
postgres(#   new_date_param date
postgres(# )
postgres=# LANGUAGE plpgsql
postgres=# AS $$
postgres=# DECLARE
postgres=#   current_attempt integer;
postgres=# BEGIN
postgres=#   SELECT attempt INTO current_attempt
postgres=#   FROM "Session".attestation
postgres=#   WHERE attestation_id = attestation_id_param;
postgres=#   INSERT INTO "Session".attestation (discipline_in_syllabus_id, professor_id, studying_student_id, mark, attempt, attestation_date)
postgres=#   SELECT discipline_in_syllabus_id, professor_id, studying_student_id, new_mark_param, current_attempt + 1, new_date_param
postgres=#   FROM "Session".attestation
postgres=#   WHERE attestation_id = attestation_id_param;
postgres=# END;
postgres=# $$;
CREATE PROCEDURE
```

```
postgres=# SELECT * FROM "Session".attestation WHERE attestation_id = 30;
attestation_id | discipline_in_syllabus_id | professor_id | studying_student_id | mark | attempt | attestation_date
-----+-----+-----+-----+-----+-----+-----
30 | 466 | 36 | 887161 | F | 1 | 2024-01-14
(1 строка)

postgres=# CALL add_reattempt_attestation(30, 'A', '2024-01-15');
CALL
postgres=# SELECT * FROM "Session".attestation WHERE attestation_id = 324;
attestation_id | discipline_in_syllabus_id | professor_id | studying_student_id | mark | attempt | attestation_date
-----+-----+-----+-----+-----+-----+-----
324 | 466 | 36 | 887161 | A | 2 | 2024-01-15
(1 строка)
```

Модифицировать триггер

```
postgres=# CREATE OR REPLACE FUNCTION fn_check_time_punch() RETURNS TRIGGER AS $$
postgres=# DECLARE
postgres=#     last_action BOOLEAN;
postgres=# BEGIN
postgres=#     -- Проверка на существование сотрудника
postgres=#     IF NOT EXISTS (
postgres=#         SELECT 1
postgres=#         FROM employee
postgres=#         WHERE id = NEW.employee_id
postgres=#     ) THEN
postgres=#         RETURN NULL;
postgres=#     END IF;
postgres=#
postgres=#     -- Проверка на наличие предыдущего действия для сотрудника
postgres=#     SELECT is_out_punch INTO last_action
postgres=#     FROM time_punch
postgres=#     WHERE employee_id = NEW.employee_id
postgres=#     ORDER BY punch_time DESC, id DESC
postgres=#     LIMIT 1;
postgres=#
postgres=#     -- Проверка на отсутствие предыдущего входа перед выходом
postgres=#     IF last_action IS NULL AND NEW.is_out_punch THEN
postgres=#         RETURN NULL;
postgres=#     END IF;
postgres=#
postgres=#     -- Проверка на повторный выход
postgres=#     IF last_action = NEW.is_out_punch THEN
postgres=#         RETURN NULL;
postgres=#     END IF;
postgres=#
postgres=#     -- Проверка на попытку выхода до времени входа
postgres=#     IF NEW.is_out_punch THEN
postgres=#         IF EXISTS (
postgres=#             SELECT 1
postgres=#             FROM time_punch tps
postgres=#             WHERE tps.employee_id = NEW.employee_id AND tps.is_out_punch = FALSE
postgres=#             ORDER BY tps.punch_time DESC, tps.id DESC
postgres=#             LIMIT 1
postgres=#         ) THEN
postgres=#             IF NEW.punch_time <= (
postgres=#                 SELECT tps.punch_time
postgres=#                 FROM time_punch tps
postgres=#                 WHERE tps.employee_id = NEW.employee_id AND tps.is_out_punch = FALSE
postgres=#                 ORDER BY tps.punch_time DESC, tps.id DESC
postgres=#                 LIMIT 1
postgres=#             ) THEN
postgres=#                 RETURN NULL;
postgres=#             END IF;
postgres=#         END IF;
postgres=#     END IF;
postgres=#     RETURN NEW;
postgres=# END;
postgres=# $$ LANGUAGE plpgsql;
CREATE FUNCTION
postgres=#
postgres=# CREATE TRIGGER check_time_punch BEFORE INSERT ON time_punch
postgres=# FOR EACH ROW EXECUTE FUNCTION fn_check_time_punch();
CREATE TRIGGER
```

```
postgres=# INSERT INTO public.employee(id, username) VALUES (4, 'Николай');
INSERT 0 1
postgres=# INSERT INTO time_punch (employee_id, is_out_punch, punch_time) VALUES (4, TRUE, NOW());
INSERT 0 0
postgres=# INSERT INTO time_punch (employee_id, is_out_punch, punch_time) VALUES (4, FALSE, '2023-01-01 12:00:00');
INSERT 0 1
postgres=# INSERT INTO time_punch (employee_id, is_out_punch, punch_time) VALUES (4, FALSE, '2023-01-01 13:00:00');
INSERT 0 0
postgres=# INSERT INTO time_punch (employee_id, is_out_punch, punch_time) VALUES (4, TRUE, '2023-01-01 13:00:00');
INSERT 0 1
postgres=# INSERT INTO time_punch (employee_id, is_out_punch, punch_time) VALUES (4, TRUE, '2023-01-01 14:00:00');
INSERT 0 0
postgres=# INSERT INTO time_punch (employee_id, is_out_punch, punch_time) VALUES (4, FALSE, '2023-01-01 15:00:00');
INSERT 0 1
postgres=# INSERT INTO time_punch (employee_id, is_out_punch, punch_time) VALUES (4, TRUE, '2023-01-01 10:00:00');
INSERT 0 0
```

```
postgres=# SELECT * FROM time_punch WHERE employee_id = 4;
 id | employee_id | is_out_punch |      punch_time
----+-----+-----+-----
 19 |          4 | f           | 2023-01-01 12:00:00
 20 |          4 | t           | 2023-01-01 13:00:00
 21 |          4 | f           | 2023-01-01 15:00:00
(3 строки)
```

Вывод

В ходе выполнения лабораторной работы были успешно освоены основные навыки создания, настройки и эффективного использования процедур, функций и триггеров в системе управления базами данных PostgreSQL. Этот опыт позволил ознакомиться с ключевыми концепциями и возможностями, которые предоставляет PostgreSQL для управления данными на более высоком уровне.