Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Факультет инфокоммуникационных технологий

ОТЧЕТ
О ЛАБОРАТОРНОЙ РАБОТЕ № 4

по теме: ЗАПРОСЫ НА ВЫБОРКУ ДАННЫХ К БД POSTGRESQL. ПРЕДСТАВЛЕНИЯ В POSTGRESQL

по дисциплине: Проектирование и реализация баз данных

Специальность:

45.03.04 Интеллектуальные системы в гуманитарной сфере

Проверила:

Говорова М.М. _____

Дата: «__» _____ 2021 г.

Оценка _____

Выполнила:

студентка группы К3243

Белова А.С.

Санкт-Петербург 2020/2021

Создать запросы на выборку данных к базе данных

1. Составить список дисциплин, которые должны быть сданы заданной группой с указанием дат сдачи и фамилий преподавателей.
   SELECT date, sub_name, t_surname FROM public."Exam"
   INNER JOIN public."Subject" USING (sub_id)
   INNER JOIN public."Teacher" USING (t_id)
   WHERE group_id = 'K3243';

```
1  SELECT date, sub_name, t_surname FROM public."Exam"
2  INNER JOIN public."Subject" USING (sub_id)
3  INNER JOIN public."Teacher" USING (t_id)
4  WHERE group_id = 'K3243';
```

Результат Notifications

	date date	sub_name text	t_surname text
1	2021-01-25	Компьютерная лингвистика	Иванов
2	2020-01-16	Математика	Мельникова

2. Вывести список студентов, получивших двойки на первой попытке с указанием фамилии преподавателя, которым они должны пересдать экзамен.
   SELECT gradebook_id, s_surname, t_surname
   FROM public."Passes"
   INNER JOIN public."Exam" USING (ex_id)
   INNER JOIN public."Student" USING (gradebook_id)
   INNER JOIN public."Teacher" USING (t_id)
   WHERE attempt = 1 AND grades = 2;

```
SELECT gradebook_id, s_surname, t_surname
FROM public."Passes"
INNER JOIN public."Exam" USING (ex_id)
INNER JOIN public."Student" USING (gradebook_id)
INNER JOIN public."Teacher" USING (t_id)
WHERE attempt = 1 AND grades = 2;
```

Результат Notifications

	gradebook_id integer	s_surname text	t_surname text
1	288867	Комарова	Мельникова

3. Вывести фамилии студентов, получивших оценки по дисциплине, которые выше среднего балла по этой дисциплине.
SELECT gradebook_id,s_surname, grades, sub_id FROM public."Passes" one
INNER JOIN public."Exam" USING (ex_id)
INNER JOIN public."Student" USING (gradebook_id)
WHERE sub_id=1 AND grades > (SELECT AVG(grades)
FROM public."Passes" INNER JOIN public."Exam" USING (ex_id)
WHERE sub_id=1 GROUP BY sub_id);

```
SELECT gradebook_id,s_surname, grades, sub_id FROM public."Passes" one
INNER JOIN public."Exam" USING (ex_id)
INNER JOIN public."Student" USING (gradebook_id)
WHERE sub_id=1 AND grades > (SELECT AVG(grades)
FROM public."Passes" INNER JOIN public."Exam" USING (ex_id)
WHERE sub_id=1 GROUP BY sub_id);
```

Результат Notifications

	gradebook_id integer	s_surname text	grades integer	sub_id integer
1	287894	Шурупов	5	1
2	285474	Белова	5	1
3	285475	Иванов	5	1

4. Создать рейтинговый список групп по заданному направлению по результатам сдачи сессии, упорядочить его по убыванию.
SELECT spec_id, id_group, round FROM
(SELECT "Student".group_id AS id_group, ROUND(AVG(grades), 2)
FROM public."Passes"
INNER JOIN public."Student" USING (gradebook_id)
INNER JOIN public."Group" one ON one.group_id = one.group_id
GROUP BY id_group) AS two
INNER JOIN public."Group" one ON one.group_id = two.id_group

ORDER BY spec_id ASC, round DESC;

```
SELECT spec_id, id_group, round FROM
(SELECT "Student".group_id AS id_group, ROUND(AVG(grades), 2) FROM public."Passes"
INNER JOIN public."Student" USING (gradebook_id)
INNER JOIN public."Group" one ON one.group_id = one.group_id
GROUP BY id_group) AS two
INNER JOIN public."Group" one ON one.group_id = two.id_group
ORDER BY spec_id ASC, round DESC;
```

Результат | План выполнения | Сообщения | Notifications

	spec_id integer	id_group text	round numeric
1	90303	K3240	3.75
2	90303	K3241	3.75
3	110302	K3220	4.00
4	110302	K3221	3.00
5	450304	K3243	4.50
6	450304	K3242	4.25

5. Создайте списки студентов, упорядоченные по группам и фамилиям студентов, содержащие данные о средних баллах и назначении на стипендии. Студент получает стипендию, если он сдал сессию без троек. Если студент не назначен на стипендию, указать 0, если назначен – 1.
   SELECT group_id, gradebook_id, s_surname, min, round FROM
   (SELECT gradebook_id, MIN(CASE WHEN grades <= 3 THEN 0 ELSE 1 END), ROUND(AVG(grades), 2)
   FROM public."Passes"
   GROUP BY gradebook_id) AS t1
   INNER JOIN (SELECT group_id, s_surname, gradebook_id FROM public."Student") AS t2 USING (gradebook_id)

ORDER BY group_id, s_surname;

```
SELECT group_id, gradebook_id, s_surname, min, round FROM
(SELECT  gradebook_id,
         MIN(CASE WHEN grades <= 3 THEN 0 ELSE 1 END),
         ROUND(AVG(grades), 2)
FROM public."Passes"
GROUP BY gradebook_id) AS t1
INNER JOIN (SELECT group_id, s_surname, gradebook_id FROM public."Student") AS t2 USING (gradebook_id)
ORDER BY group_id, s_surname
```

Результат | План выполнения | Сообщения | Notifications

	group_id text	gradebook_id integer	s_surname text	min integer	round numeric
1	К3220	288857	Васильков	1	4.00
2	К3220	285477	Соколов	1	4.00
3	К3221	288867	Комарова	0	2.50
4	К3221	285457	Морева	1	4.00
5	К3240	287896	Карпов	0	3.50
6	К3240	285445	Киселев	0	4.00
7	К3241	287856	Быстрова	0	4.00
8	К3241	287857	Голикова	0	3.50
9	К3242	287895	Терехова	0	3.50
10	К3242	287894	Шурупов	1	5.00

6. Вывести список студентов, сдавших все положенные экзамены.

```
SELECT gradebook_id, group_id
FROM (SELECT gradebook_id,
SUM(CASE
WHEN grades < 3 THEN 0
ELSE 1
END)
FROM public."Passes"
GROUP BY gradebook_id) AS one
INNER JOIN public."Student" USING (gradebook_id)
INNER JOIN (SELECT group_id,  COUNT(ex_id)
FROM public."Exam" GROUP BY group_id) AS two USING
(group_id)
WHERE sum = count;
```

```sql
SELECT gradebook_id, group_id
FROM (SELECT gradebook_id,
SUM(CASE
WHEN grades < 3 THEN 0
ELSE 1
END)
FROM public."Passes"
GROUP BY gradebook_id) AS one
INNER JOIN public."Student" USING (gradebook_id)
INNER JOIN (SELECT group_id,  COUNT(ex_id)
FROM public."Exam" GROUP BY group_id) AS two USING (group_id)
WHERE sum = count;
```

Результат Notifications

	gradebook_id integer	group_id text
1	288857	K3220
2	285477	K3220
3	287895	K3242
4	287894	K3242
5	285475	K3243
6	285474	K3243
7	287896	K3240
8	285445	K3240
9	287857	K3241
10	287856	K3241

7. Вывести список студентов, получивших максимальный средний балл в своей группе.
   SELECT s_surname, group_id, round FROM
   (SELECT gradebook_id, ROUND(AVG(grades), 2)
   FROM public."Passes"
   GROUP BY "Passes".gradebook_id) as one
   INNER JOIN public."Student" USING (gradebook_id)
   WHERE (group_id, round) = ANY(SELECT group_id, MAX(round)
   FROM
   (SELECT gradebook_id, ROUND(AVG(grades), 2)
   FROM public."Passes"
   GROUP BY "Passes".gradebook_id) AS t1
   INNER JOIN public."Student" USING (gradebook_id)

GROUP BY group_id);

```
SELECT s_surname, group_id, round FROM
(SELECT  gradebook_id,
         ROUND(AVG(grades), 2)
FROM public."Passes"
GROUP BY "Passes".gradebook_id) as one
INNER JOIN public."Student" USING (gradebook_id)
WHERE (group_id, round) = ANY(SELECT  group_id, MAX(round) FROM
(SELECT  gradebook_id,
         ROUND(AVG(grades), 2)
FROM public."Passes"
GROUP BY "Passes".gradebook_id) AS t1
INNER JOIN public."Student" USING (gradebook_id)
GROUP BY group_id)
```

Результат Notifications

	s_surname text	group_id text	round numeric
1	Белова	К3243	5.00
2	Киселев	К3240	4.00
3	Соколов	К3220	4.00
4	Морева	К3221	4.00
5	Шурупов	К3242	5.00
6	Быстрова	К3241	4.00
7	Васильков	К3220	4.00

Создать представления на выборку данных к базе данных

1. Список студентов, получивших двойки на первой попытке с указанием фамилии преподавателя, которым они должны пересдать экзамен
   CREATE VIEW dstudents AS
   SELECT "Passes".gradebook_id AS gradebook_id,
   "Student".s_surname AS s_surname, "Passes".grades AS grades,
   "Teacher".t_surname AS t_surname
   FROM public."Passes"
   INNER JOIN public."Exam" USING (ex_id)
   INNER JOIN public."Student" USING (gradebook_id)
   INNER JOIN public."Teacher" USING (t_id)
   WHERE attempt = 1 AND grades < 3;

SELECT * FROM dstudents;

```
CREATE VIEW dstudents AS
SELECT "Passes".gradebook_id AS gradebook_id, "Student".s_surname AS s_surname,
"Passes".grades AS grades, "Teacher".t_surname AS t_surname
FROM public."Passes"
INNER JOIN public."Exam" USING (ex_id)
INNER JOIN public."Student" USING (gradebook_id)
INNER JOIN public."Teacher" USING (t_id)
WHERE attempt = 1 AND grades < 3;
SELECT * FROM dstudents;
```

Результат Notifications

	gradebook_id integer	s_surname text	grades integer	t_surname text
1	288867	Комарова	2	Мельникова

2. Данных о студентах при получении ими хотя бы одной оценки 2 (после 3-й попытки)

```
CREATE VIEW failed AS
SELECT "Student".s_surname AS s_surname
FROM public."Student"
WHERE 2 = ANY (SELECT "Passes".grades AS grades
                    FROM public."Passes"
                    NATURAL JOIN public."Exam"
                    WHERE "Student".gradebook_id =
"Passes".gradebook_id AND "Exam".attempt = 3);
```

SELECT * FROM failed;

```
CREATE VIEW failed AS
SELECT "Student".s_surname AS s_surname
FROM public."Student"
WHERE 2 = ANY (SELECT "Passes".grades AS grades
               FROM public."Passes"
               NATURAL JOIN public."Exam"
               WHERE "Student".gradebook_id = "Passes".gradebook_id AND "Exam".attempt = 3);

SELECT * FROM failed;
```

Результат Notifications

	s_surname text

Составить 3 запроса на модификацию данных

1. Для повышения стипендии отличникам на 10% (UPDATE).
   До:

```
1  SELECT scholarship FROM
2  public."Student";
```

Результат Notifications

	scholarship integer
1	2000
2	4000
3	2000
4	4000
5	2000
6	4000
7	4000
8	2000
9	2000
10	4000
11	2000
12	2000

Запрос:

```
UPDATE public."Student"
SET scholarship = scholarship * 1.1
WHERE gradebook_id IN (SELECT gradebook_id
FROM (SELECT gradebook_id,
SUM(CASE
WHEN grades != 5 THEN 0
ELSE 1
END)
FROM public."Passes"
GROUP BY gradebook_id) AS one
INNER JOIN public."Student" USING (gradebook_id)
INNER JOIN (SELECT group_id,  COUNT(ex_id)
FROM public."Exam" GROUP BY group_id) AS two USING
(group_id)
```

WHERE sum = count);

```
UPDATE public."Student"
SET scholarship = scholarship * 1.1
WHERE gradebook_id IN (SELECT gradebook_id
FROM (SELECT gradebook_id,
SUM(CASE
WHEN grades != 5 THEN 0
ELSE 1
END)
FROM public."Passes"
GROUP BY gradebook_id) AS one
INNER JOIN public."Student" USING (gradebook_id)
INNER JOIN (SELECT group_id,  COUNT(ex_id)
FROM public."Exam" GROUP BY group_id) AS two USING (group_id)
WHERE sum = count);
```

Сообщения

```
UPDATE 2
```

После:

```
select scholarship from
public."Student";
```

Результат Notifications

	scholarship integer
1	4000
2	2000
3	4000
4	4000
5	4000
6	2000
7	2000
8	4000
9	2000
10	2000
11	2200
12	2200

2. Добавить сдачу экзамена студенту К3243 группы на 16.01.2020 (INSERT)

До:

```
1  SELECT * FROM public."Passes"
2
```

Результат Notifications

	gradebook_id integer	ex_id integer	grades integer
10	285477	5	4
11	288867	6	2
12	285457	6	4
13	288867	7	3
14	287894	8	5
15	287895	8	3
16	285474	9	5
17	285475	9	5
18	285445	10	5
19	287896	10	3
20	287856	11	3
21	287857	11	4

Запрос:

INSERT INTO public."Passes"
VALUES (285596, (SELECT ex_id FROM public."Exam" WHERE group_id = 'K3243' AND date = '2020-01-16'), 4);

После:

```
1  SELECT * FROM public."Passes"
2
```

Результат Notifications

	gradebook_id integer	ex_id integer	grades integer
11	288867	6	2
12	285457	6	4
13	288867	7	3
14	287894	8	5
15	287895	8	3
16	285474	9	5
17	285475	9	5
18	285445	10	5
19	287896	10	3
20	287856	11	3
21	287857	11	4
22	285596	2	4

3. Удалить сдачу второго экзамена у студента, который получил за нее 4 (DELETE)
   Запрос:
   DELETE FROM public."Passes"
   WHERE gradebook_id = (SELECT gradebook_id FROM public."Passes" NATURAL JOIN public."Exam" WHERE ex_id = 2 AND grades = 4);

```
DELETE FROM public."Passes"
WHERE gradebook_id = (SELECT gradebook_id FROM public."Passes"
                      NATURAL JOIN public."Exam" WHERE ex_id = 2 AND grades = 4);
```

Сообщения

```
DELETE 1

Запрос завершён успешно, время выполнения: 224 msec.
```

После:

```
SELECT * FROM public."Passes"
```

Результат Notifications

	gradebook_id integer	ex_id integer	grades integer
10	285477	5	4
11	288867	6	2
12	285457	6	4
13	288867	7	3
14	287894	8	5
15	287895	8	3
16	285474	9	5
17	285475	9	5
18	285445	10	5
19	287896	10	3
20	287856	11	3
21	287857	11	4