

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ  
Федеральное государственное бюджетное образовательное учреждение высшего  
образования

“Московский авиационный институт”

(национальный исследовательский университет)

**Факультет №3** «Системы управления, информатика и электроэнергетика»

Отчет по домашней работе №4  
по курсу «**Управление базами данных**»

**Выполнили:**

студенты группы ЗО-218М-19:

Пономарев Роман

**Принял:**

Доцент, к.т.н. Моргунов Е.П.

Москва, 2020

## 1. Задание №2

Посмотрите, какие ограничения уже наложены на атрибуты таблицы «Успеваемость» (progress). Воспользуйтесь командой \d утилиты psql. А теперь предложите для этой таблицы ограничение уровня таблицы.

В качестве примера рассмотрим такой вариант. Добавьте в таблицу progress еще один атрибут — «Форма проверки знаний» (test\_form), который может принимать только два значения: «экзамен» или «зачет». Тогда набор допустимых значений атрибута «Оценка» (mark) будет зависеть от того, экзамен или зачет предусмотрены по данной дисциплине. Если предусмотрен экзамен, тогда допускаются значения 3, 4, 5, если зачет — тогда 0 (не зачтено) или 1 (зачтено). Не забудьте, что значения NULL для атрибутов test\_form и mark не допускаются.

Новое ограничение может быть таким:

```
ALTER TABLE progress
ADD CHECK (
  ( test_form = 'экзамен' AND mark IN ( 3, 4, 5 ) )
OR
  ( test_form = 'зачет' AND mark IN ( 0, 1 ) )
);
```

Проверьте, как будет работать новое ограничение в модифицированной таблице progress. Для этого выполните команды INSERT, как удовлетворяющие ограничению, так и нарушающие его.

В таблице уже было ограничение на допустимые значения атрибута mark. Как вы думаете, не будет ли оно конфликтовать с новым ограничением? Проверьте эту гипотезу. Если ограничения конфликтуют, тогда удалите старое ограничение и снова попробуйте добавить строки в таблицу. Подумайте, какое еще ограничение уровня таблицы можно предложить для этой таблицы?

*Ответ:*

Созданное ограничение будет конфликтовать с уже имеющимся, так как при значении атрибута test\_form “зачет” значение атрибута mark будет равно либо 1, либо 0, что недопустимо в существующем ограничении progress\_mark\_check.

```
INSERT INTO students (record_book, name, doc_ser, doc_num)
VALUES (1, 'Петр', 1, 1),
(2, 'Иван', 1, 2),
(3, 'Борис', 1, 3),
(4, 'Николай', 1, 4),
(5, 'Федор', 1, 5);
ALTER TABLE progress DROP constraint progress_mark_check ;
```

```
ALTER TABLE progress ADD COLUMN test_form varchar(7) NOT NULL CHECK
(test_form IN ('зачет', 'экзамен')) CHECK ((test_form = 'экзамен' AND mark IN
(3,4,5)) OR (test_form = 'зачет' AND mark IN (0,1)));
```

```
INSERT INTO students (record_book, name, doc_ser, doc_num) VALUES (1,
'Роман', 4611, 234567), (2, 'Николай', 1111,123321 );
```

```
INSERT INTO progress (record_book, subject, acad_year, term, mark, test_form)
VALUES (1, 'матан', 2019, 1, 5, 'экзамен'), (2, 'ангем', 2019, 1, 1, 'зачет'
);
```

```
File Edit View Search Terminal Help

edu=# ALTER TABLE progress ADD CHECK ((test_form = 'экзамен' AND mark IN (3,4,5)) OR (test_form = '
зачет' AND mark IN (0,1)));ALTER TABLE
edu=# INSERT INTO progress (record_book, subject, acad_year, term, mark, test_form) VALUES (1, 'мат
ан', 2019, 1, 5, 'экзамен'), (2, 'ангем', 2019, 1, 1, 'зачет' );
INSERT 0 2
edu=# \d progress

          Table "public.progress"
   Column   |          Type          | Collation | Nullable | Default
-----|-----|-----|-----|-----
record_book | numeric(5,0)           |           | not null |
subject     | text                   |           | not null |
acad_year   | text                   |           | not null |
term        | numeric(1,0)           |           | not null |
mark        | numeric(1,0)           |           | not null | 5
test_form   | character varying(7)   |           | not null |

Check constraints:
    "progress_check" CHECK (test_form::text = 'экзамен'::text AND (mark = ANY (ARRAY[3::numeric, 4:
:numeric, 5::numeric])) OR test_form::text = 'зачет'::text AND (mark = ANY (ARRAY[0::numeric, 1::nu
meric])))
    "progress_term_check" CHECK (term = 1::numeric OR term = 2::numeric)
    "progress_test_form_check" CHECK (test_form::text = ANY (ARRAY['зачет'::character varying, 'эка
мен'::character varying]::text[]))
Foreign-key constraints:
    "progress_record_book_fkey" FOREIGN KEY (record_book) REFERENCES students(record_book) ON UPDAT
E CASCADE ON DELETE CASCADE

edu=#
```

Некорректная строка:

```
INSERT INTO progress (record_book, subject, acad_year, term, mark, test_form)
VALUES (3, 'Физика', 2019, 1, 1, 'экзамен');
```

Также возможно добавить ограничения, например:

```
ALTER TABLE progress
ADD CONSTRAINT subject_and_term_check CHECK (
```

```
File Edit View Search Terminal Help

edu=# INSERT INTO progress (record_book, subject, acad_year, term, mark, test_form) VALUES (1, 'мат
ан', 2019, 1, 5, 'экзамен'), (2, 'ангем', 2019, 1, 1, 'зачет' );
INSERT 0 2
edu=# \d progress

          Table "public.progress"
   Column   |          Type          | Collation | Nullable | Default
-----|-----|-----|-----|-----
record_book | numeric(5,0)           |           | not null |
subject     | text                   |           | not null |
acad_year   | text                   |           | not null |
term        | numeric(1,0)           |           | not null |
mark        | numeric(1,0)           |           | not null | 5
test_form   | character varying(7)   |           | not null |

Check constraints:
    "progress_check" CHECK (test_form::text = 'экзамен'::text AND (mark = ANY (ARRAY[3::numeric, 4:
:numeric, 5::numeric])) OR test_form::text = 'зачет'::text AND (mark = ANY (ARRAY[0::numeric, 1::nu
meric])))
    "progress_term_check" CHECK (term = 1::numeric OR term = 2::numeric)
    "progress_test_form_check" CHECK (test_form::text = ANY (ARRAY['зачет'::character varying, 'эка
мен'::character varying]::text[]))
Foreign-key constraints:
    "progress_record_book_fkey" FOREIGN KEY (record_book) REFERENCES students(record_book) ON UPDAT
E CASCADE ON DELETE CASCADE

edu=# INSERT INTO students ( record_book, name, doc_ser, doc_num )
edu=# VALUES ( 12300, '', 0402, 543281 );
INSERT 0 1
edu=#
```

```
(subject = 'матан' OR subject = 'ангем' OR subject = 'история' AND term = 1)
```

## 2. Задание №9

В таблице «Студенты» (students) есть текстовый атрибут name, на который наложено ограничение NOT NULL. Как вы думаете, что будет, если при вводе новой строки в эту таблицу дать атрибуту name в качестве значения пустую строку?

Наверное, проектируя эту таблицу, мы хотели бы все же, чтобы пустые строки в качестве значения атрибута name не проходили в базу данных? Какое решение вы можете предложить? Видимо, нужно добавить ограничение CHECK для столбца name. Если вы еще не изучили команду ALTER TABLE, то удалите таблицу students и создайте ее заново с учетом нового ограничения, а если вы уже познакомились с командой ALTER TABLE, то сделайте так:

```
ALTER TABLE students ADD CHECK ( name <> ' ' );
```

Добавив ограничение, попробуйте теперь вставить в таблицу students строку (row), в которой значение атрибута name было бы пустой строкой (string).

Давайте продолжим эксперименты и предложим в качестве значения атрибута name строку, содержащую сначала один пробел, а потом – два пробела.

```
INSERT INTO students VALUES ( 12346, ' ', 0406, 112233 );
```

```
INSERT INTO students VALUES ( 12347, '  ', 0407, 112234 );
```

Для того чтобы «увидеть» эти пробелы в выборке, сделаем так:

```
SELECT *, length( name ) FROM students;
```

Оказывается, эти невидимые значения имеют ненулевую длину. Что делать, чтобы не допустить таких значений-невидимок? Один из способов: возложить проверку таких ситуаций на прикладную программу. А что можно сделать на уровне определения таблицы students? Какое ограничение нужно предложить? В разделе 9.4 документации «Строковые функции и операторы» есть функция trim. Попробуйте воспользоваться ею. Если вы еще не изучили команду ALTER TABLE, то удалите таблицу students и создайте ее заново с учетом нового ограничения, а если уже познакомились с ней, то сделайте так:

```
ALTER TABLE students ADD CHECK ( ... );
```

Есть ли подобные слабые места в таблице «Успеваемость» (progress)?

Есть, так как не наложены никакие ограничения на значения атрибутов subject и acad\_year.

*Ответ:*

Строка вставиться, так как пустая строка это не NULL;

```
ALTER TABLE students ADD CHECK (length (trim(both from name)) > 0);
```

## 3. Задание №17

Представления могут быть, условно говоря, вертикальными и горизонтальными. При создании вертикального представления в список его столбцов включается лишь часть столбцов базовой таблицы (таблиц). Например:

```
CREATE VIEW airports_names AS
SELECT airport_code, airport_name, city
FROM airports;
SELECT * FROM airports_names;
```

В горизонтальное представление включаются не все строки базовой таблицы (таблиц), а производится их отбор с помощью фраз WHERE или HAVING.

Например:

```
CREATE VIEW siberian_airports AS
SELECT * FROM airports
WHERE city = 'Новосибирск' OR city = 'Кемерово';
SELECT * FROM siberian_airports;
```

Конечно, вполне возможен и смешанный вариант, когда ограничивается как список столбцов, так и множество строк при создании представления.

Подумайте, какие представления было бы целесообразно создать для нашей базы данных «Авиаперевозки». Необходимо учесть наличие различных групп пользователей, например: пилоты, диспетчеры, пассажиры, кассиры.

Создайте представления и проверьте их в работе.

*Ответ:*

1. Для проведения экономического анализа было бы полезно в конце каждого дня получать информацию о среднем значении полной стоимости бронирования:

```
CREATE VIEW avg_total_amount_today_view AS
SELECT avg(total_amount) FROM bookings
WHERE book_date = current_date;
```

```
SELECT * FROM avg_total_amount_today_view;
```

Но так как в таблице нет сегодняшней даты, то изменим дату на имеющуюся:

```
CREATE VIEW avg_total_amount_today_view AS
SELECT avg(total_amount) FROM bookings
WHERE book_date = '2016-09-12';
```

2. Также бортпроводникам полезна информация о местах на рейсах и номерах посадочных талонов при посадке пассажиров на определенном рейсе:

```
CREATE VIEW seat_to_boarding_view AS
SELECT boarding_no, seat_no FROM boarding_passes
WHERE flight_id = 2055;
```

```
SELECT * FROM seat_to_boarding_view;
```

3. Диспетчерам аэропорта «Внуково» нужна информация о вылетевших самолетах и месте их назначения.

```
CREATE VIEW VKO_on_time_view AS
SELECT aircraft_code, arrival_airport FROM flights
WHERE departure_airport = 'VKO' AND status = 'On Time';
```

```
SELECT * FROM VKO_on_time_view;
```

4. А пилотам нужна информация о их расписании:

```
CREATE VIEW PG0405_schedule_view AS
SELECT scheduled_departure, scheduled_arrival, departure_airport,
arrival_airport FROM flights
WHERE flight_no = 'PG0405'
ORDER BY scheduled_departure;
```

```
SELECT * FROM PG0405_schedule_view;
```

```
Файл  Правка  Вид  Bookmarks  Настройка  Справка
lights
demo=# WHERE flight_no = 'PG0405'
demo=# ORDER BY scheduled_departure;
CREATE VIEW
demo=# SELECT * FROM avg_total_amount_today_view;
      avg
-----
121400.000000000000000
(1 строка)

demo=# SELECT * FROM seat_to_boarding_view;
 boarding_no | seat_no
-----+-----
          1 | 1C
          2 | 1D
          3 | 2B
          4 | 2C
          5 | 2D
          6 | 3B
          7 | 3C
          8 | 4A
          9 | 4B
         10 | 4C
         11 | 4D
         12 | 5A
         13 | 5B
         14 | 5D
         15 | 6A
         16 | 6C
         17 | 7B
         18 | 18A
         19 | 18B
         20 | 18C
         21 | 18D
         22 | 19A
         23 | 19B
         24 | 19D
         25 | 20A
         26 | 20B
         27 | 20C
         28 | 20D
         29 | 21C
         30 | 21D
         31 | 22A
         32 | 22B
         33 | 22D
         34 | 23B
(34 строки)

demo=#
```

stud : psql

```

Файл  Правка  Вид  Bookmarks  Настройка  Справка
demo=# SELECT * FROM VKO_on_time_view;
 aircraft_code | arrival_airport
-----+-----
321            | LED
321            | LED
321            | LED
CR2            | OMS
773            | PEE
SU9            | VOG
CR2            | MMK
CR2            | PES
CN1            | JOK
CN1            | JOK
319            | DYR
CR2            | SLY
CR2            | NFG
CR2            | RTW
SU9            | ULV
763            | VVO
SU9            | REN
CR2            | CSY
CR2            | STW
CR2            | MQF
SU9            | BZK
SU9            | BZK
SU9            | BZK
763            | AER
CR2            | KRO
SU9            | OGZ
CR2            | SKX
(27 строк)

demo=# SELECT * FROM PG0405_schedule_view;
 scheduled_departure | scheduled_arrival | departure_airport | arrival_airport
-----+-----+-----+-----
2016-09-13 12:35:00+07 | 2016-09-13 13:30:00+07 | DME               | LED
2016-09-14 12:35:00+07 | 2016-09-14 13:30:00+07 | DME               | LED
2016-09-15 12:35:00+07 | 2016-09-15 13:30:00+07 | DME               | LED
2016-09-16 12:35:00+07 | 2016-09-16 13:30:00+07 | DME               | LED
2016-09-17 12:35:00+07 | 2016-09-17 13:30:00+07 | DME               | LED
2016-09-18 12:35:00+07 | 2016-09-18 13:30:00+07 | DME               | LED
2016-09-19 12:35:00+07 | 2016-09-19 13:30:00+07 | DME               | LED
2016-09-20 12:35:00+07 | 2016-09-20 13:30:00+07 | DME               | LED
2016-09-21 12:35:00+07 | 2016-09-21 13:30:00+07 | DME               | LED
2016-09-22 12:35:00+07 | 2016-09-22 13:30:00+07 | DME               | LED
2016-09-23 12:35:00+07 | 2016-09-23 13:30:00+07 | DME               | LED
2016-09-24 12:35:00+07 | 2016-09-24 13:30:00+07 | DME               | LED
2016-09-25 12:35:00+07 | 2016-09-25 13:30:00+07 | DME               | LED
2016-09-26 12:35:00+07 | 2016-09-26 13:30:00+07 | DME               | LED
2016-09-27 12:35:00+07 | 2016-09-27 13:30:00+07 | DME               | LED
2016-09-28 12:35:00+07 | 2016-09-28 13:30:00+07 | DME               | LED

```

stud : psql

## 4. Задание №18

Предположим, что нам понадобилось иметь в базе данных сведения о технических характеристиках самолетов, эксплуатируемых в авиакомпании. Пусть это будут такие сведения, как число членов экипажа (пилоты), тип двигателей и их количество.

Следовательно, необходимо добавить новый столбец в таблицу «Самолеты» (aircrafts). Дадим ему имя specifications, а в качестве типа данных выберем jsonb. Если впоследствии потребуется добавить и другие характеристики, то мы сможем это сделать, не модифицируя определение таблицы.

```
ALTER TABLE aircrafts ADD COLUMN specifications jsonb;
```

```
ALTER TABLE
```

Добавим сведения для модели самолета Airbus A320-200:

```
UPDATE aircrafts
SET specifications =
'{ "crew": 2,
  "engines": { "type": "IAE V2500",
               "num": 2
            }
} '::jsonb
WHERE aircraft_code = '320';
UPDATE 1
```

Посмотрим, что получилось:

```
SELECT model, specifications
```

```
FROM aircrafts
WHERE aircraft_code = '320';
model | specifications
-----+-----
```

```
Airbus A320-200 | {"crew": 2, "engines": {"num": 2, "type":
"IAE V2500"}}
(1 строка)
```

Можно посмотреть только сведения о двигателях:

```
SELECT model, specifications->'engines' AS engines
FROM aircrafts
WHERE aircraft_code = '320';
model | engines
-----+-----
```

```
Airbus A320-200 | {"num": 2, "type": "IAE V2500"}
(1 строка)
```

Чтобы получить еще более детальные сведения, например, о типе двигателей, нужно учитывать, что созданный JSON-объект имеет сложную структуру: он содержит вложенный JSON-объект. Поэтому нужно использовать оператор #> для указания пути доступа к ключу второго уровня.

```
SELECT model, specifications #> '{ engines, type }'
FROM aircrafts
WHERE aircraft_code = '320';
model | ?column?
-----+-----
```

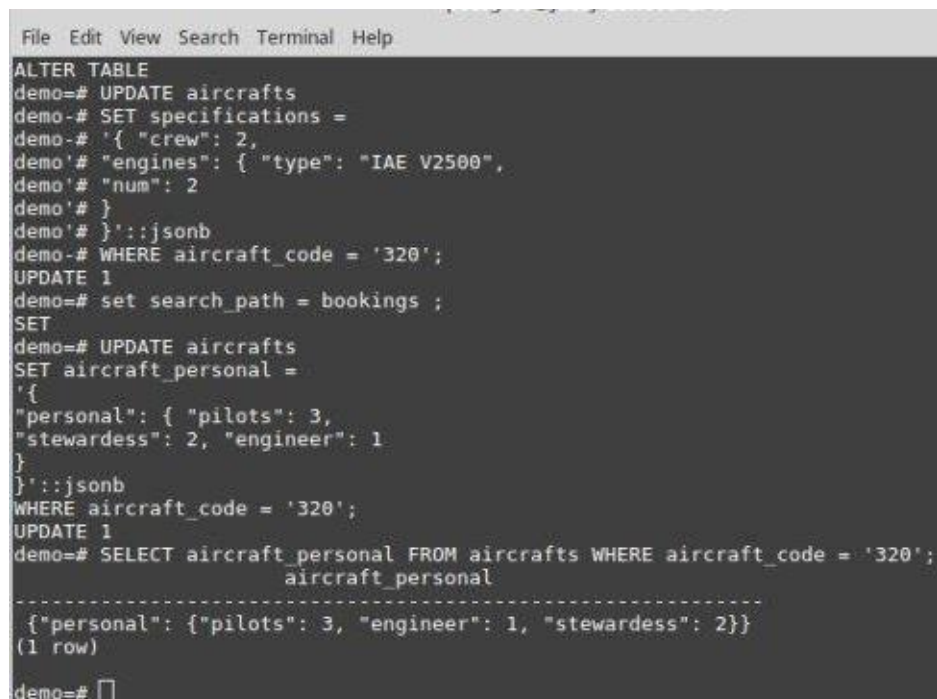
```
Airbus A320-200 | "IAE V2500"
(1 строка)
```

**Задание.** Подумайте, какие еще таблицы было бы целесообразно дополнить столбцами типа json/jsonb. Вспомните, что, например, в таблице «Билеты» (tickets) уже есть столбец такого типа contact\_data. Выполните модификации таблиц и измените в них одну-две строки для проверки правильности ваших решений.

**Ответ:**

Можно дополнить столбцами формата jsonb таблицу aircrafts для указания, например, численности персонала закрепленного за конкретным самолетом

```
UPDATE aircrafts
SET aircraft_personal =
'{
"personal": { "pilots": 3,
"stewardess": 2, "engineer": 1
}
}'::jsonb
WHERE aircraft_code = '320';
```



```
File Edit View Search Terminal Help
ALTER TABLE
demo=# UPDATE aircrafts
demo=# SET specifications =
demo=# '{ "crew": 2,
demo=# "engines": { "type": "IAE V2500",
demo=# "num": 2
demo=# }
demo=# }'::jsonb
demo=# WHERE aircraft_code = '320';
UPDATE 1
demo=# set search_path = bookings ;
SET
demo=# UPDATE aircrafts
SET aircraft_personal =
'{
"personal": { "pilots": 3,
"stewardess": 2, "engineer": 1
}
}'::jsonb
WHERE aircraft_code = '320';
UPDATE 1
demo=# SELECT aircraft_personal FROM aircrafts WHERE aircraft_code = '320';
aircraft_personal
-----
{"personal": {"pilots": 3, "engineer": 1, "stewardess": 2}}
(1 row)
demo=#
```