



МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

**Институт информационных технологий (ИИТ)
Кафедра цифровой трансформации (ЦТ)**

ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ
по дисциплине «Разработка баз данных»

Практическое занятие №1

Студенты группы *ИКБО-66-23 Смирнов А.Ю.*

(подпись)

Ассистент *Копылова Я.А*

(подпись)

Отчет представлен «__» _____ 2025 г.

Москва 2025 г.

ПРАКТИЧЕСКАЯ РАБОТА №1. ОСНОВЫ DDL И ЗАПРОСЫ НА ВЫБОРКУ ДАННЫХ В POSTGRES PRO

Постановка задачи: основываясь на логической модели данных, спроектированной в рамках курса «Проектирование баз данных» в предыдущем семестре, выполните следующие шаги:

1. Письменно опишите не менее 5 различных бизнес-правил и не менее 3 ограничений целостности для таблиц. Выбор бизнес-правил и ограничений целостности производится на усмотрение студента. Результаты представить в виде таблицы.
2. С использованием DDL-оператора CREATE TABLE создать все необходимые таблицы (согласно созданной в прошлом семестре логической модели данных) в СУБД Postgres Pro, корректно реализовав все описанные ограничения целостности.
3. Заполнить созданные таблицы согласованными тестовыми данными (не менее 5-7 записей на таблицу, где это применимо) с помощью DML-оператора INSERT INTO.
4. Составить и выполнить не менее 6 SQL-запросов к таблицам, иллюстрирующих использование различных элементов списка выборки и условных выражений WHERE, согласно перечню, указанному в задании (см. Ход выполнения работы). В запросах должны быть использованы все приведённые операторы (4 для SELECT и 5 для WHERE).
5. Составить и выполнить по два SQL-запроса к таблицам для демонстрации работы ORDER BY, GROUP BY и HAVING.
6. Каждый SQL-запрос сопровождать комментарием, объясняющим его назначение и логику работы.

Таблица 1. Описание ограничений таблицы Payment Method

Название столбца	Тип данных	Ограничение	Обоснование (Бизнесправило)
id_payment_method	VARCHAR(5)	PRIMARY KEY	Уникальный идентификатор способа оплаты.
reference	VARCHAR(30)	NOT NULL	Название способа оплаты (например, наличные, карта). Не может быть пустым.

Таблица 2. Описание ограничений таблицы Payment

Название столбца	Тип данных	Ограничение	Обоснование (Бизнесправило)
id_payment	VARCHAR(20)	PRIMARY KEY	Уникальный идентификатор платежа.
id_invoice	VARCHAR(15)	NOT NULL, FOREIGN KEY (Invoice)	Ссылка на счёт, к которому относится оплата.
id_payment_method	VARCHAR(5)	NOT NULL, FOREIGN KEY (Payment_Method)	Оплата должна быть привязана к способу оплаты.
amount	VARCHAR(2500)	NOT NULL	Сумма платежа обязательна.
payment_date	VARCHAR(24)	NOT NULL	Дата проведения платежа обязательна.
method	VARCHAR(10)	NOT NULL	Краткое обозначение способа оплаты (например, Cash, Card).

Таблица 3. Описание ограничений таблицы Employee

Название столбца	Тип данных	Ограничение	Обоснование (Бизнесправило)
id_employee	VARCHAR(15)	PRIMARY KEY	Уникальный идентификатор сотрудника.
first_name	VARCHAR(20)	NOT NULL	Имя сотрудника обязательно.

last_name	VARC HAR(20)	NOT NULL	Фамилия сотрудника обязательна.
middle_name	VARCHAR(20)	NOT NULL	Отчество сотрудника обязательно.
contact	VARCHAR(30)	NOT NULL	Контактные данные обязательны.
position	VARCHAR(5)	NOT NULL	Должность обязательна.

Таблица 4. Описание ограничений таблицы Car_Type

Название столбца	Тип данных	Ограничение	Обоснование (Бизнесправило)
id_car_type	VARCHAR(5)	PRIMARY KEY	Уникальный идентификатор типа авто.
reference	BIGINT	NOT NULL	Код типа автомобиля обязателен.

Таблица 5. Описание ограничений таблицы Car

Название столбца	Тип данных	Ограничение	Обоснование (Бизнесправило)
Car_id	VARCHAR(30)	PRIMARY KEY	Уникальный идентификатор автомобиля.
id_car_type	VAR	NOT NULL,	Автомобиль

	CHAR(5)	FOREIGN KEY (Car_Type	должен относиться к определённому типу.
brand	VARCHAR(5)	NOT NULL	Марка автомобиля обязательна.
model	VARCHAR(5)	NOT NULL	Модель автомобиля обязательна.
license_plate	VARCHAR(30)	NOT NULL	Номерной знак обязателен.
status	VARCHAR(50)	NOT NULL	Статус автомобиля обязателен (например, Свободен, Забронирован).
production_year	DATE	NOT NULL	Год выпуска обязателен.

Таблица 6. Описание ограничений таблицы Booking_Status

Название столбца	Тип данных	Ограничение	Обоснование (Бизнесправило)
id_booking_status	VARCHAR(30)	PRIMARY KEY	Уникальный идентификатор статуса брони.
status	VARCHAR(50)	NOT NULL	Название статуса обязательно (Активна,

			Отменена, Завершена). обязательным параметром.
--	--	--	---

Таблица 7. Описание ограничений Invoice

Название столбца	Тип данных	Ограничение	Обоснование (Бизнесправило)
id_invoice	VARCHAR(20)	PRIMARY KEY	Уникальный идентификатор счёта.
cost	VARCHAR(2500)	NOT NULL	Стоимость обязательна.

Таблица 8. Описание ограничений таблицы Client

Название столбца	Тип данных	Ограничение	Обоснование (Бизнесправило)
Client_id	VARCHAR(15)	PRIMARY KEY	Уникальный идентификатор клиента.
first_name	VARCHAR(20)	NOT NULL,	Имя клиента обязательно.
last_name	VARCHAR(20)	NULL	Отчество клиента отсутствовать.
driver_license	VARCHAR(15)	NOT NULL	Водительское удостоверение обязательно. находиться в допустимом

			диапазоне (например, от 1 до 10).
contact	VARCHAR(50)	NO T NULL	Контактные данные обязательны.

Таблица 9. Описание ограничений таблицы *Tariff*

Название столбца	Тип данных	Ограничение	Обоснование (Бизнесправило)
genre_id	VARCHAR(5)	PRIMARY KEY	Уникальный идентификатор жанра, генерируется автоматически.
id_car_type	VARCHAR(5)	NOT NULL, FOREIGN KEY (Car_Type)	Тариф должен относиться к типу автомобиля.
daily_rate	VARCHAR(2500)	NOT NULL	Стоимость аренды в сутки обязательна.

Таблица 10. Описание ограничений таблицы *movie_genre_link*

Название столбца	Тип данных	Ограничение	Обоснование (Бизнесправило)
id_booking	VARCHAR(400)	PRIMARY KEY	Уникальный идентификатор связи,

			генерируется автоматически.
id_client	VARCHAR(100)	NOT NULL, FOREIGN KEY (Client)	Каждая бронь связана с клиентом.
id_car	VARCHAR(40)	NOT NULL, FOREIGN KEY (Car)	Бронирование обязательно связано с автомобилем.
id_invoice	VARCHAR(20)	NOT NULL, FOREIGN KEY (Invoice)	Каждое бронирование связано со счётом.
id_employee	VARCHAR(20)	NOT NULL, FOREIGN KEY (Employee)	Бронь оформляет сотрудник.
client_car_link	VARCHAR(15)	NOT NULL	Служебное поле для связи клиента и авто.
id_booking_status	VARCHAR(30)	NOT NULL, FOREIGN KEY (Booking_Status)	Каждая бронь имеет статус.


```

CREATE TABLE Payment_Method (
    id_payment_method varchar(5) NOT NULL PRIMARY KEY,
    reference varchar(30) NOT NULL
);

CREATE TABLE Payment (
    id_payment varchar(20) NOT NULL PRIMARY KEY,
    id_invoice varchar(15) NOT NULL,
    id_payment_method varchar(5) NOT NULL,
    amount varchar(2500) NOT NULL,
    payment_date varchar(24) NOT NULL,
    method varchar(10) NOT NULL
);

CREATE TABLE Employee (
    id_employee varchar(15) NOT NULL PRIMARY KEY,
    first_name varchar(20) NOT NULL,
    last_name varchar(20) NOT NULL,
    middle_name varchar(20) NOT NULL,
    position varchar(5) NOT NULL,
    contact varchar(30) NOT NULL
);

CREATE TABLE Car_Type (
    id_car_type varchar(5) NOT NULL PRIMARY KEY,
    reference bigint NOT NULL
);

CREATE TABLE Car (
    id_car varchar(30) NOT NULL PRIMARY KEY,
    id_car_type varchar(5) NOT NULL,
    brand varchar(5) NOT NULL,
    model varchar(5) NOT NULL,
    production_year date NOT NULL,
    license_plate varchar(30) NOT NULL,
    status varchar(50) NOT NULL
);

CREATE TABLE Booking_Status (
    id_booking_status varchar(30) NOT NULL PRIMARY KEY,
    status varchar(50) NOT NULL
);

CREATE TABLE Invoice (
    id_invoice varchar(20) NOT NULL PRIMARY KEY,
    cost varchar(2500) NOT NULL
);

CREATE TABLE Client (
    id_client varchar(15) NOT NULL PRIMARY KEY,
    first_name varchar(20) NOT NULL,
    last_name varchar(20) NOT NULL,
    middle_name varchar(20),
    driver_license varchar(15) NOT NULL,
    contact varchar(50) NOT NULL
);

CREATE TABLE Tariff (
    id_tariff varchar(5) NOT NULL PRIMARY KEY,
    id_car_type varchar(5) NOT NULL,
    daily_rate varchar(2500) NOT NULL
);

CREATE TABLE Booking (
    id_booking varchar(400) NOT NULL PRIMARY KEY,
    id_client varchar(100) NOT NULL,
    id_car varchar(40) NOT NULL,
    id_booking_status varchar(30) NOT NULL,
    id_employee varchar(20) NOT NULL,
    id_invoice varchar(20) NOT NULL,
    client_car_link varchar(15) NOT NULL
);

```

Рис.1-2 Скрипт создания таблиц

```

-- Payment methods
INSERT INTO Payment_Method (id_payment_method, reference) VALUES
('01', 'Cash'),
('02', 'Card'),
('03', 'Online');

-- Employees
INSERT INTO Employee (id_employee, first_name, last_name, middle_name, position, contact) VALUES
('E001', 'Anna', 'Ivanova', 'Petrovna', 'Mngr', '89991234567'),
('E002', 'Igor', 'Smirnov', 'Aleksievich', 'Admin', '89997654321');

-- Car types
INSERT INTO Car_Type (id_car_type, reference) VALUES
('T1', 100),
('T2', 200);

-- Cars
INSERT INTO Car (id_car, id_car_type, brand, model, production_year, license_plate, status) VALUES
('A001', 'T1', 'BMW', 'X5', '2022-01-01', 'A123BC77', 'Available'),
('A002', 'T2', 'Audi', 'Q7', '2023-03-15', 'B456NE77', 'Booked');

-- Booking statuses
INSERT INTO Booking_Status (id_booking_status, status) VALUES
('ST1', 'Active'),
('ST2', 'Cancelled'),
('ST3', 'Completed');

-- Clients
INSERT INTO Client (id_client, first_name, last_name, middle_name, driver_license, contact) VALUES
('C001', 'Oleg', 'Morozov', 'Valeryevich', '1234567890', '89881112233'),
('C002', 'Larisa', 'Kuznetsova', NULL, '0987654321', '89880001122');

-- Invoices
INSERT INTO Invoice (id_invoice, cost) VALUES
('S001', '2500'),
('S002', '4800'),
('S003', '3200');

-- Payments
INSERT INTO Payment (id_payment, id_invoice, id_payment_method, amount, payment_date, method) VALUES
('P001', 'S001', '01', '2500', '2025-06-01', 'Cash'),
('P002', 'S002', '02', '4800', '2025-06-01', 'Card'),
('P003', 'S003', '03', '3200', '2025-06-02', 'Online');

```

Рис 3 — Скрипт заполнения таблиц

```

ALTER TABLE Car
ADD CONSTRAINT Car_id_car_type_fk FOREIGN KEY (id_car_type) REFERENCES Car_Type (id_car_type);

ALTER TABLE Booking
ADD CONSTRAINT Booking_id_car_fk FOREIGN KEY (id_car) REFERENCES Car (id_car);

ALTER TABLE Booking
ADD CONSTRAINT Booking_id_client_fk FOREIGN KEY (id_client) REFERENCES Client (id_client);

ALTER TABLE Booking
ADD CONSTRAINT Booking_id_employee_fk FOREIGN KEY (id_employee) REFERENCES Employee (id_employee);

ALTER TABLE Booking
ADD CONSTRAINT Booking_id_booking_status_fk FOREIGN KEY (id_booking_status) REFERENCES Booking_Status (id_booking_status);

ALTER TABLE Booking
ADD CONSTRAINT Booking_id_invoice_fk FOREIGN KEY (id_invoice) REFERENCES Invoice (id_invoice);

ALTER TABLE Payment
ADD CONSTRAINT Payment_id_payment_method_fk FOREIGN KEY (id_payment_method) REFERENCES Payment_Method (id_payment_method);

ALTER TABLE Payment
ADD CONSTRAINT Payment_id_invoice_fk FOREIGN KEY (id_invoice) REFERENCES Invoice (id_invoice);

ALTER TABLE Tariff
ADD CONSTRAINT Tariff_id_car_type_fk FOREIGN KEY (id_car_type) REFERENCES Car_Type (id_car_type);

```

Рис 4 — Скрипт реализующий ограничения целостности

Выполнение пунктов 4-6:

Составить и выполнить не менее 6 SQL-запросов к таблицам, иллюстрирующих использование различных элементов списка выборки и условных выражений WHERE, согласно перечню, указанному в задании (см. Ход выполнения работы). В запросах должны быть использованы все приведённые операторы (4 для SELECT и 5 для WHERE).

Запрос с DISTINCT

The screenshot shows a database client interface with a SQL editor and a results pane. The SQL editor contains the following query:

```
-- Найти уникальные номера водительских прав всех клиентов
SELECT DISTINCT license_number
FROM smirnov_aiu.client;
```

The results pane displays a table with the following data:

	AZ license_number
1	B1234567
2	D3456789
3	C2345678

Показывает все уникальные номера водительских прав без повторений

The screenshot shows a database client interface with a SQL editor and a results pane. The SQL editor contains the following query:

```
-- Вывести список клиентов с переименованными столбцами
SELECT
    name AS "Имя",
    surname AS "Фамилия",
    contact AS "Телефон"
FROM smirnov_aiu.client;
```

The results pane displays a table with the following data:

	AZ Имя	AZ Фамилия	AZ Телефон
1	John	Doe	12345678901
2	Alice	Smith	10987654321
3	Bob	Johnson	11223344556

Выводит информацию о клиентах

SQL Script 1:

```
-- Посчитать общее количество автомобилей
SELECT COUNT(*) AS "Всего автомобилей"
FROM smirnov_aiu.car;
```

Result 4:

Всего автомобилей
123

Подсчитывает общее количество автомобилей в базе данных.

SQL Script 2:

```
-- Классифицировать автомобили по марке
SELECT
  make,
  model,
  CASE
    WHEN make = 'Toyota' THEN 'Японский автопром'
    WHEN make = 'BMW' THEN 'Немецкий автопром'
    ELSE 'Другие производители'
  END AS category
FROM smirnov_aiu.car;
```

Result 4:

AZ make	AZ model	AZ category
1 Toyota	Camry	Японский автопром
2 BMW	X5	Немецкий автопром
3 Ford	Focus	Другие производители

Разделяет автомобили на категории в зависимости от марки

SQL Script 3:

```
-- Найти автомобили новее 2020 года
SELECT *
FROM smirnov_aiu.car
WHERE year > '2020-01-01';
```

Result 4:

id_car	id_car_type	AZ make	AZ model	year	AZ license_plate	AZ status
1 123	2	BMW	X5	2022-05-20	EF5678GH	booked

Выводит автомобили, выпущенные после 2020 года.

Script-1 Script-2 Script-3 booking Console

```
-- Найти сотрудников с ID в определенном диапазоне
SELECT *
FROM smirnov_aiu.employee
WHERE id_employee BETWEEN 1 AND 5;
```

car 1 client 2 client 3 Результат 4 car 5 car 6 car 7 employee

SELECT * FROM smirnov_aiu.employee WHERE id_employee | Введите SQL выражение чтобы отфильтровать резу

id_employee	AZ name	AZ surname	AZ position	AZ contact
1	Karen	White	Manager	12312312311
2	David	Black	Agent	32132132122

Выводит сотрудников с идентификаторами в указанном диапазоне.

Script-1 Script-2 Script-3 booking Console X

```
-- Найти автомобили конкретных марок
SELECT *
FROM smirnov_aiu.car
WHERE make IN ('Toyota', 'Honda', 'Lada');
```

car 1 client 2 client 3 Результат 4 car 5 car 6 car 7 employee 8 employee 9 emp

SELECT * FROM smirnov_aiu.car WHERE make IN ('Toyota', 'Honda', 'Lada'); Введите SQL выражение чтобы отфильтровать результаты

id_car	id_car_type	AZ make	AZ model	year	AZ license_plate	AZ status
1	1	Toyota	Camry	2020-01-01	AB1234CD	available

Показывает автомобили только указанных марок.

*<dbstud> Script-1 *<dbstud> Script-2 *<dbstud> Script-3 booking

```
-- Посмотреть все фамилии клиентов
SELECT DISTINCT surname
FROM smirnov_aiu.client
ORDER BY surname;
```

car 1 client 2 client 3 Результат 4 car 5 car 6 car 7

SELECT DISTINCT surname FROM smirnov_aiu.client ORDER BY surname ASC Введите SQL выражение чтобы отфильтровать результаты

Таблица

	A-Z surname
1	Doe
2	Johnson
3	Smith

Текст

Сначала посмотреть какие фамилии вообще есть в базе.

*<dbstud> Script-1 *<dbstud> Script-2 *<dbstud> Script-3 booking *<dbstud> Console X

```
-- Сортировать клиентов по фамилии (А-Я)
SELECT *
FROM smirnov_aiu.client
ORDER BY surname ASC;
```

car 1 client 2 client 3 Результат 4 car 5 car 6 car 7 employee 8 employee 9

SELECT * FROM smirnov_aiu.client ORDER BY surname ASC Введите SQL выражение чтобы отфильтровать результаты

Таблица

	123 id_client	A-Z name	A-Z surname	A-Z fathers_name	A-Z license_number	A-Z contact
1	1	John	Doe	Michael	B1234567	12345678901
2	3	Bob	Johnson	Edward	D3456789	11223344556
3	2	Alice	Smith	[NULL]	C2345678	10987654321

Сортировать клиентов по фамилии

*<dbstud> Script-1 *<dbstud> Script-2 *<dbstud> Script-3 booking *<dbstud> Console X

```

-- Сортировать автомобили по году выпуска (сначала новые)
SELECT *
FROM smirnov_aiu.car
ORDER BY year DESC;

```

client 3 Результат 4 car 5 car 6 car 7 employee 8 employee 9 employee 10 car 11

SELECT * FROM smirnov_aiu.car ORDER BY year DESC | Введите SQL выражение чтобы отфильтровать результаты

	123 id_car	123 id_car_type	A-Z make	A-Z model	year	A-Z license_plate	A-Z status
1	2	2	BMW	X5	2022-05-20	EF5678GH	booked
2	1	1	Toyota	Camry	2020-01-01	AB1234CD	available
3	3	3	Ford	Focus	2019-09-15	IJ9012KL	maintenance

Сортировать автомобили по году выпуска

*<dbstud> Script-1 *<dbstud> Script-2 *<dbstud> Script-3 booking *<dbstud> Console

```

-- Найти статусы бронирований, где минимальная сумма заказа меньше 5000
SELECT bs.status, MIN(i.amount::numeric) as min_amount
FROM smirnov_aiu.booking b
JOIN smirnov_aiu.booking_status bs ON b.id_booking_status = bs.id_booking_status
JOIN smirnov_aiu.invoice i ON b.id_invoice = i.id_invoice
GROUP BY bs.status
HAVING MIN(i.amount::numeric) < 5000
ORDER BY min_amount ASC;

```

car 11 client 12 client 13 client 14 client 15 car 16 car_type 17

SELECT bs status, MIN(i.amount::numeric) as min_amount FROM | Введите SQL выражение чтобы отфильтровать резул

	A-Z status	123 min_amount
1	cancelled	3 000

Найти статусы бронирований, где минимальная сумма заказа меньше 5000(HAVING)

The screenshot shows a database client window with multiple tabs. The active tab is labeled '*<dbstud> Console'. It contains a SQL query in Russian: '-- Найти уникальные типы автомобилей, которые есть в автопарке'. The query is: `SELECT DISTINCT ct.description as car_type FROM smirnov_aiu.car c JOIN smirnov_aiu.car_type ct ON c.id_car_type = ct.id_car_type ORDER BY car_type;`. Below the query editor, there is a toolbar with buttons for 'employee 9', 'employee 10', 'car 11', 'client 12', 'client 13', 'client 14', 'client 15', and 'car 16'. Below the toolbar is a search bar with the text 'Введите SQL выражение чтобы отфильтровать результаты'. At the bottom, there is a table with the following data:

Таблица	AZ car_type
1	Hatchback
2	SUV
3	Sedan

Показывает сколько бронирований в каждом статусе, исключая статусы без бронирований. (Distinct)

Вывод: научился создавать ограничения в таблицах и использовать SQL-запросы для фильтрации данных