

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО  
ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ  
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ  
ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.  
Г. ШУХОВА» (БГТУ им. В.Г. Шухова)**

Кафедра программного обеспечения вычислительной техники и  
автоматизированных систем

**Лабораторная работа №2**

по дисциплине: Базы данных

тема: «Создание объектов базы данных в СУБД»

Выполнил: ст. группы ПВ-223  
Игнатъев Артур Олегович

Проверил:  
Панченко Максим Владимирович

Белгород 2024 г.

### **Вариант 3**

**Цель работы:** изучить основные возможности языка SQL для создания структуры базы данных. Научиться создавать базы данных, таблицы, связи, ограничения, а также создавать, изменять и удалять данные.

**Задание:**

1. Составить SQL-запросы для создания структуры базы данных, полученной в результате лабораторной работы №1. Указать используемые типы данных, ограничения значений полей; для связей: действия с записями подчинённой таблицы при удалении и изменении соответствующей записи главной таблицы.

2. С помощью SQL-запросов выполнить добавление 3–4 записей в каждую таблицу, изменение и удаление нескольких записей.

**Задание 1.** Составить SQL-запросы для создания структуры базы данных, полученной в результате лабораторной работы №1. Указать используемые типы данных, ограничения значений полей; для связей: действия с записями подчинённой таблицы при удалении и изменении соответствующей записи главной таблицы.

Тип данных / ограничение / правило	описание
BIGSERIAL	Целочисленный тип, рекомендуется для записи ID. Используется для создания автоинкрементных первичных ключей.
VARCHAR (N)	Строка на N символов. Используется для текстовых полей с фиксированным максимальным количеством символов, например для ФИО или названия.
TIMESTAMP	Хранит дату и время. Подходит для временных меток, таких как дата создания заказа.
INTEGER	Целочисленный тип. Подходит для хранения числовых значений, например количества или контактных данных.
NUMERIC(N, M)	Точный вещественный тип, где N – количество цифр целой части, а M – количество цифр в вещественной части. Применяется для полей, где важна точность, например, для суммы или веса.
NOT NULL	Поле не может быть NULL. Требует обязательного значения, например, для полей, которые не могут оставаться пустыми.
ON DELETE SET NULL	Когда удаляется экземпляр связанной таблицы, значение в поле становится NULL. Подходит для слабых зависимостей, где при удалении внешней записи можно оставить запись, но с NULL значением во внешнем ключе.
ON DELETE CASCADE	Когда удаляется экземпляр связанной таблицы, экземпляр этой таблицы также удаляется. Используется для строгих зависимостей, где отсутствие записи в одной таблице делает

	невозможным существование связанных записей в другой таблице.
--	---

### Код с описанием правил связей:

```
-- Сначала создаем таблицу статусов, чтобы можно было ссылаться на неё из
других таблиц
CREATE TABLE IF NOT EXISTS status (
    id BIGSERIAL PRIMARY KEY, -- Уникальный идентификатор статуса
    name VARCHAR(50) NOT NULL -- Название статуса (например, "В процессе"),
не может быть NULL
);

-- Таблица водителей
CREATE TABLE IF NOT EXISTS driver (
    id BIGSERIAL PRIMARY KEY, -- Уникальный идентификатор водителя
    license_number INTEGER NOT NULL, -- Номер водительского удостоверения,
не может быть NULL
    name VARCHAR(100) NOT NULL -- Имя и ФИО водителя, не может быть NULL
);

-- Таблица транспортных компаний
CREATE TABLE IF NOT EXISTS transport_company (
    id BIGSERIAL PRIMARY KEY, -- Уникальный идентификатор компании
    name VARCHAR(100) NOT NULL, -- Название компании, не может быть NULL
    driver_count INTEGER DEFAULT 0 -- Количество водителей, по умолчанию 0
);

-- Таблица поставщиков
CREATE TABLE IF NOT EXISTS provider (
    id BIGSERIAL PRIMARY KEY, -- Уникальный идентификатор поставщика
    name VARCHAR(100) NOT NULL -- Название поставщика, не может быть NULL
);

-- Таблица потребителей
CREATE TABLE IF NOT EXISTS consumer (
    id BIGSERIAL PRIMARY KEY, -- Уникальный идентификатор потребителя
    contact_details BIGINT NOT NULL, -- Контактные данные (например, номер
телефона), не может быть NULL
    name VARCHAR(100) NOT NULL -- Имя и ФИО потребителя, не может быть NULL
);

-- Таблица заказов
CREATE TABLE IF NOT EXISTS orders (
    id BIGSERIAL PRIMARY KEY, -- Уникальный идентификатор заказа
    weight INTEGER NOT NULL, -- Вес заказа, не может быть NULL
    status_id BIGINT REFERENCES status(id) ON DELETE SET NULL, -- Статус
заказа с внешним ключом, если статус удаляется, поле становится NULL
    consumer_id BIGINT REFERENCES consumer(id) ON DELETE SET NULL -- Внешний
ключ к потребителю, при удалении потребителя значение становится NULL
);

-- Таблица продуктов
CREATE TABLE IF NOT EXISTS product (
    id BIGSERIAL PRIMARY KEY, -- Уникальный идентификатор продукта
    name VARCHAR(100) NOT NULL -- Название продукта, не может быть NULL
);

-- Таблица, связывающая водителей и транспортные компании (работа водителя в
компании)
CREATE TABLE IF NOT EXISTS driver_company (
    driver_id BIGINT NOT NULL REFERENCES driver(id) ON DELETE CASCADE, --
Внешний ключ к водителю, при удалении водителя запись удаляется
```

```

        company_id BIGINT NOT NULL REFERENCES transport_company(id) ON DELETE
        CASCADE, -- Внешний ключ к компании, при удалении компании запись удаляется
        PRIMARY KEY (driver_id, company_id) -- Составной первичный ключ для
        уникальности пар (водитель, компания)
    );

-- Таблица, связывающая поставщиков и продукты (товары, поставляемые
поставщиками)
CREATE TABLE IF NOT EXISTS provider_product (
    provider_id BIGINT NOT NULL REFERENCES provider(id) ON DELETE CASCADE, -
-- Внешний ключ к поставщику, при удалении поставщика запись удаляется
    product_id BIGINT NOT NULL REFERENCES product(id) ON DELETE CASCADE, --
Внешний ключ к продукту, при удалении продукта запись удаляется
    PRIMARY KEY (provider_id, product_id) -- Составной первичный ключ для
уникальности пар (поставщик, продукт)
);

```

**Задание 2.** С помощью SQL-запросов выполнить добавление 3–4 записей в каждую таблицу, изменение и удаление нескольких записей.

Добавление записей в таблицы driver, transport\_company, provider, consumer, order, status, и product:

```

-- Таблица водителей
INSERT INTO driver (id, license_number, name)
VALUES
    (1, 123456, 'Иван Иванов'),
    (2, 654321, 'Петр Петров'),
    (3, 789123, 'Сергей Сергеев');

-- Таблица транспортных компаний
INSERT INTO transport_company (id, name, driver_count)
VALUES
    (1, 'ТрансЛогистик', 15),
    (2, 'ЭкоДрайв', 10),
    (3, 'АвтоЛидер', 20);

-- Таблица поставщиков
INSERT INTO provider (id, name)
VALUES
    (1, 'Компания А'),
    (2, 'Компания Б'),
    (3, 'Компания В');

-- Таблица потребителей
INSERT INTO consumer (id, contact_details, name)
VALUES
    (1, 1234567890, 'Алексей Смирнов'),
    (2, 9876543210, 'Ольга Петрова'),
    (3, 5556667778, 'Наталья Иванова');

-- Таблица заказов
INSERT INTO orders (id, weight)
VALUES
    (1, 500),
    (2, 1200),
    (3, 800);

-- Таблица статусов
INSERT INTO status (id, name)
VALUES
    (1, 'В процессе'),
    (2, 'Отправлен'),
    (3, 'Доставлен');

```

```
-- Таблица продуктов
INSERT INTO product (id, name)
VALUES
    (1, 'Товар А'),
    (2, 'Товар Б'),
    (3, 'Товар В');
```

Добавление записей о заказе, который выполнен водителем и отправлен потребителю:

```
-- Создание записи о новом заказе с указанием транспортной компании и
потребителя
INSERT INTO orders (id, weight)
VALUES
    (4, 750); -- добавляем заказ весом 750 кг

-- Добавление статуса заказа
INSERT INTO status (id, name)
VALUES
    (4, 'Ожидает отправки');

-- Обновление статуса существующего заказа
UPDATE orders
    SET weight = 650
    WHERE id = 1;

-- Снижение количества водителей в компании
UPDATE transport_company
    SET driver_count = driver_count - 1
    WHERE id = 1;
```

## Пример удаления данных из таблиц

```
-- Удаление записи из таблицы заказов
DELETE FROM orders WHERE id = 2;

-- Удаление продукта
DELETE FROM product WHERE id = 3;

-- Удаление потребителя
DELETE FROM consumer WHERE id = 1;
```

## Примеры работы ограничений

### 1. Проверка NOT NULL ограничения

Попробуем вставить запись с пустым значением в поле, для которого установлено ограничение NOT NULL. Например, добавим водителя без имени.

```
1 -- Попытка вставить водителя без имени, что нарушает ограничение NOT NULL
2 v INSERT INTO driver (license_number, name)
3 VALUES (12345, NULL); -- ОШИБКА: значение NULL в столбце "name" нарушает ограничение NOT NULL
4
```

Data Output Messages Notifications

ERROR: Ошибочная строка содержит (1, 12345, null).значение NULL в столбце "name" отношения "driver" нарушает ограничение NOT NULL

ОШИБКА: значение NULL в столбце "name" отношения "driver" нарушает ограничение NOT NULL

SQL state: 23502

Detail: Ошибочная строка содержит (1, 12345, null).

## 2. Проверка ON DELETE CASCADE

Ограничение ON DELETE CASCADE означает, что если удалить запись из главной таблицы, то все связанные записи в подчиненной таблице также будут удалены. Например, удалим водителя, который связан с транспортной компанией через таблицу `driver_company`.

	id [PK] bigint	license_number integer	name character varying (100)
1	1	123456	Иван Иванов
2	2	654321	Петр Петров
3	3	789123	Сергей Сергеев

	id [PK] bigint	name character varying (100)	driver_count integer
1	1	ТрансЛогистик	15
2	2	ЭкоДрайв	10
3	3	АвтоЛидер	20

```
1  -- Связываем водителя с транспортной компанией в таблице driver_company
2  INSERT INTO driver_company (driver_id, company_id) VALUES (1, 1);
3
```

Data Output Messages Notifications

INSERT 0 1

Query returned successfully in 40 msec.




```
1  -- Теперь удаляем водителя, и запись в driver_company также будет удалена
2  DELETE FROM driver WHERE id = 1;
```

Data Output Messages Notifications

```
DELETE 1
```

Query returned successfully in 42 msec.

	<b>driver_id</b> [PK] bigint 	<b>company_id</b> [PK] bigint 
--	---	--

	id [PK] bigint 	name character varying (100) 	driver_count integer 
1	1	ТрансЛогистик	15
2	2	ЭкоДрайв	10
3	3	АвтоЛидер	20

**Вывод:** в ходе работы изучены основные возможности языка SQL для создания структуры базы данных. Научились создавать базы данных, таблицы, связи, ограничения, а также создавать, изменять и удалять данные.