



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-22-23 Платонов Т.А.*

(подпись)

Старший
преподаватель *Черняускас В. В*

(подпись)

Отчет представлен «___»_____2025 г.

Москва 2025 г.

СОДЕРЖАНИЕ

1. Постановка задачи.....	3
2. Выполнение практической работы.....	3
Задание 1	3
Задание 4	8
Задание 5	12

1. Постановка задачи

Для выполнения практической работы необходимо последовательно выполнить следующие шаги, основываясь на логической модели данных, которая была спроектирована в рамках курса «Проектирование баз данных» в предыдущем семестре.

2. Выполнение практической работы

Задание 1

Опишем ограничения для таблицы album

Название столбца	Тип данных	Ограничение	Обоснование
id_album	SERIAL	PRIMARY KEY	Уникальный идентификатор альбома, генерируется автоматически
title	VARCHAR(255)	NOT NULL	Название должно быть
releaseyear	DATE	NOT NULL, CHECK (releaseyear <= CURRENT DATE)	Год выхода альбома должен быть и не может быть в будущем
id_artist_fk	INTEGER	FOREIGN KEY	Ссылка на артиста. Каждая пластинка принадлежит какому-либо артисту
id_genre_fk	INTEGER	FOREIGN KEY	Ссылка на жанр. Каждая пластинка принадлежит какому-либо жанру

Опишем ограничения для таблицы record_condition

Название столбца	Тип данных	Ограничение	Обоснование
id_condition	SERIAL	PRIMARY KEY	Уникальный идентификатор состояния пластинки. Генерируется автоматически
conditioncode	VARCHAR(255)	NOT NULL	Код состояния должен указываться всегда
description	VARCHAR(255)	NOT NULL	Код состояния должен описываться всегда

Опишем ограничения для suppliertype

Название столбца	Тип данных	Ограничение	Обоснование
id_supplier_type	SERIAL	PRIMARY KEY	Уникальный код типа поставщика
name	VARCHAR(255)	NOT NULL	Название типа поставщика не может быть пустым

Опишем ограничения для таблицы purchaseorder

Название столбца	Тип данных	Ограничение	Обоснование (бизнес-правило)
id_purchase	SERIAL	PRIMARY KEY	Уникальный идентификатор покупки, генерируется автоматически

ordernumber	VARCHAR(255)	NOT NULL, CHECK (ordernumber >=0)	Номер заказа не должен быть пустым и не может быть отрицательным
orderdate	DATE	NOT NULL, CHECK (orderdate <= CURRENT DATE)	Дата заказа должна быть обязательно и не может быть в будущем
orderstatus	VARCHAR(255)	NOT NULL	Статус заказа не может быть пустым
totalamount	INTEGER	NOT NULL, CHECK (totalamount >=0)	Итоговая сумма не может быть пустой и отрицательной
id_supplier	INTEGER	FOREIGN KEY	Ссылка на поставщика. Заказ не может существовать без связи с поставщиком

Опишем ограничения для таблицы recordinstance

Название столбца	Тип данных	Ограничение	Обоснование
Id_record_instance	SERIAL	PRIMARY KEY	Уникальный идентификатор экземпляра пластинки, генерируется автоматически
Id_album_fk	INTEGER	FOREIGN KEY	Ссылка на альбом. Пластинка не может быть без связи с альбомом
Id_condition_fk	INTEGER	FOREIGN KEY	Ссылка на состояние. У каждой пластинки должно быть состояние

purchaseprice	INTEGER	NOT NULL, CHECK (purchaseprice >=0)	Цена покупки должна быть и не может быть отрицательной
acquisitiondate	DATE	NOT NULL, CHECK (acquisitiondate <= CURRENT DATE)	Дата приемки в магазин не может быть в будущем и должна существовать
notes	VARCHAR(255)		Здесь нет ограничений, т.к. это поле заполняется опциональной информацией о экземпляре

Задание 4

На рисунке 1 представлен SELECT SQL-запрос - вывод всех данных из таблицы album. Удобно для диагностики и проверки содержимого таблицы.

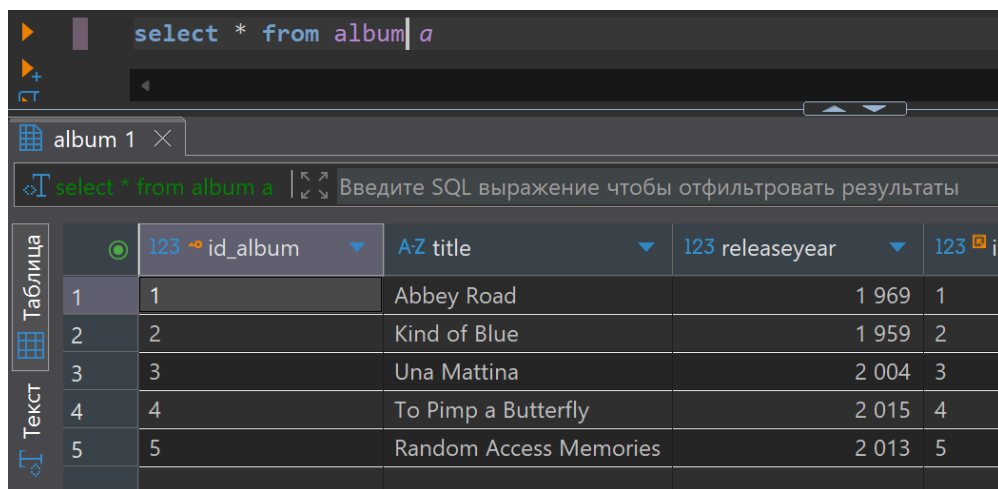


Рисунок 1 – Первый Select-запрос

На рисунке 2 представлен другой SELECT SQL-запрос, в котором выводится разница между ценой в заказе и закупочной ценой экземпляра записи

```

SELECT poi.id_purchase_order_item,
       ri.purchaseprice,
       poi.priceinorder,
       (poi.priceinorder - ri.purchaseprice) AS profit
FROM purchaseorderitem poi
JOIN recordinstance ri
  ON poi.id_record_instance_fk = ri.id_record_instance;

```

	id_purchase_order_item	purchaseprice	priceinorder	profit
1	123	25,5	25,5	0
2	123	30	30	0
3	123	20	20	0

Рисунок 2 – Второй Select-запрос

На рисунке 3 представлено использование where в запросе, благодаря чему получаем заказы, которые были закуплены именно у поставщика Vinyl Heaven

```

SELECT po.ordernumber,
       po.orderdate,
       s.name AS supplier_name
FROM purchaseorder po
JOIN supplier s ON po.id_supplier_fk = s.id_supplier
WHERE s.name = 'Vinyl Heaven';

```

	ordernumber	orderdate	supplier_name
1	PO-0001	2022-01-10	Vinyl Heaven

Рисунок 3 – Использование WHERE

```

SELECT name, quantity, (quantity * 1200)::numeric(12,2) AS stock_value
FROM product;

```

	name	quantity	stock_value
1	Amber Night	50	60 000
2	Ocean Breeze	30	36 000
3	Moisturizing Cream	100	120 000

Рисунок 4 – Третий Select-запрос

На рисунке 5 показано использование WHERE с AND.

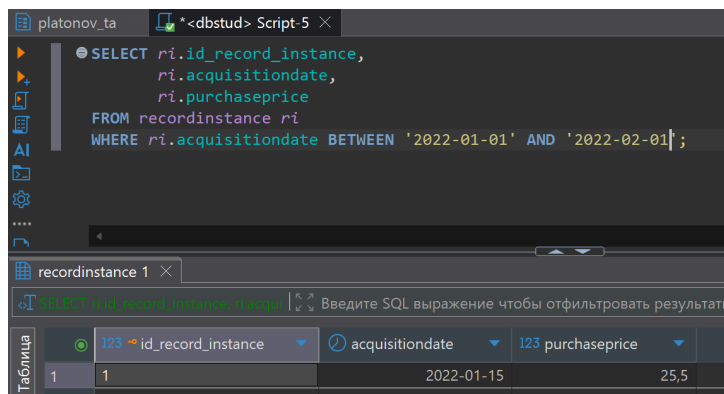


Рисунок 5 – Использование WHERE с AND

На рисунке 6 показано использование BETWEEN.

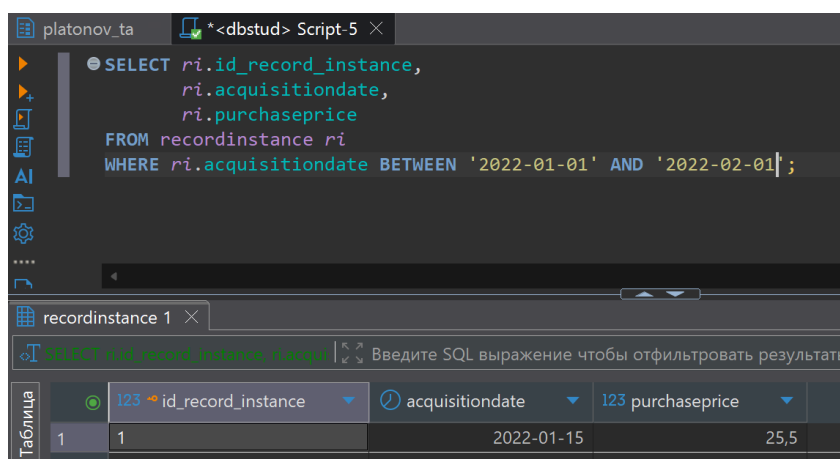


Рисунок 6 – Использование BETWEEN

На рисунке 7 показано использование LIKE. Найдём все адреса, которые начинаются с 'Springfield'.

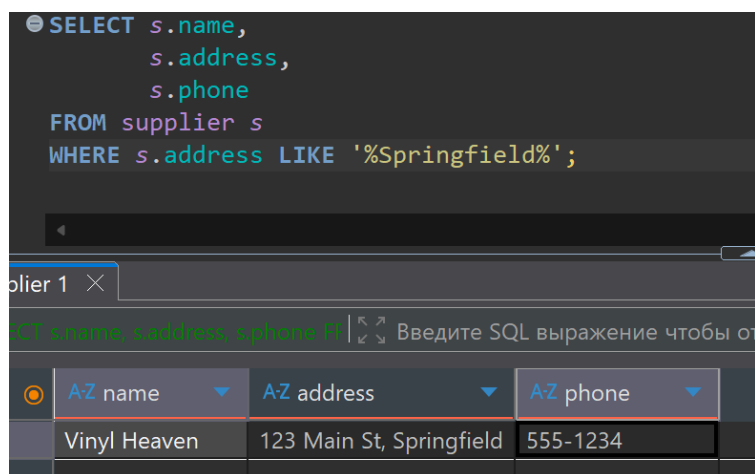
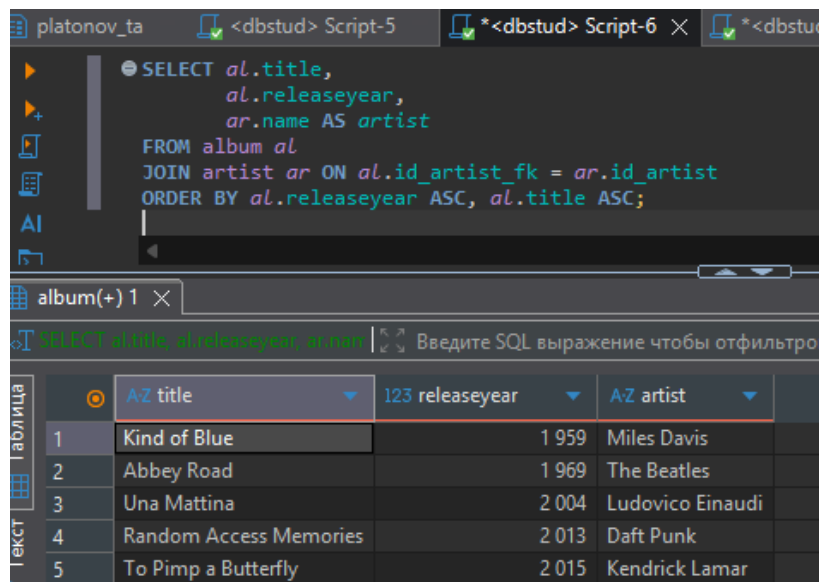


Рисунок 7 – Использование LIKE

Задание 5

На рисунке 8 показано использование ORDER BY – сортировка результатов, позволяющего сортировать данные по возрастанию года выпуска альбома.

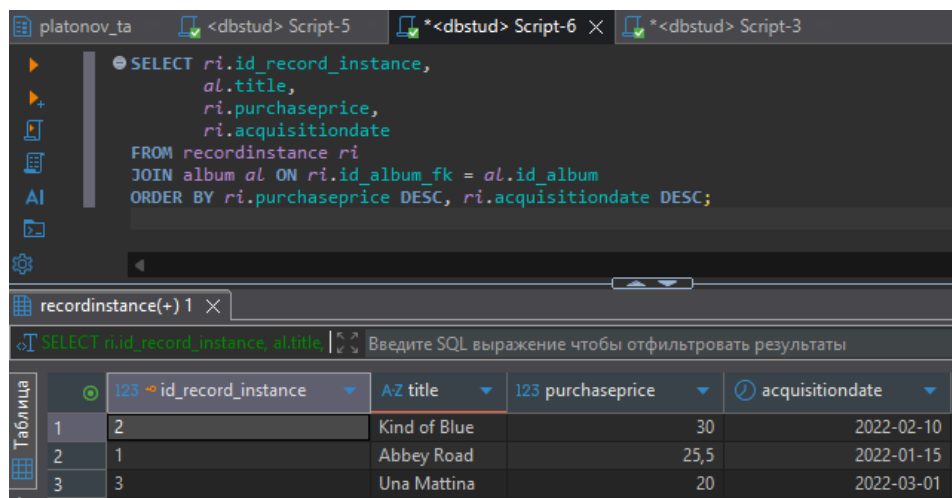


```
SELECT al.title,
       al.releaseyear,
       ar.name AS artist
FROM album al
JOIN artist ar ON al.id_artist_fk = ar.id_artist
ORDER BY al.releaseyear ASC, al.title ASC;
```

	AZ title	123 releaseyear	AZ artist
1	Kind of Blue	1 959	Miles Davis
2	Abbey Road	1 969	The Beatles
3	Una Mattina	2 004	Ludovico Einaudi
4	Random Access Memories	2 013	Daft Punk
5	To Pimp a Butterfly	2 015	Kendrick Lamar

Рисунок 8 – Использование ORDER BY ч.1

На рисунке 9 показана сортировка сначала по цене, затем по дате приемки.



```
SELECT ri.id_record_instance,
       al.title,
       ri.purchaseprice,
       ri.acquisitiondate
FROM recordinstance ri
JOIN album al ON ri.id_album_fk = al.id_album
ORDER BY ri.purchaseprice DESC, ri.acquisitiondate DESC;
```

	123 id_record_instance	AZ title	123 purchaseprice	acquisitiondate
1	2	Kind of Blue	30	2022-02-10
2	1	Abbey Road	25,5	2022-01-15
3	3	Una Mattina	20	2022-03-01

Рисунок 9 – Использование ORDER BY ч.2

На рисунке 10 показано использование GROUP BY - агрегации по группам. Здесь мы считаем сколько альбомов у каждого артиста.

```

SELECT ar.name AS artist,
       COUNT(al.id_album) AS albums_count
FROM artist ar
LEFT JOIN album al ON al.id_artist_fk = ar.id_artist
GROUP BY ar.name
ORDER BY albums_count DESC, artist ASC;

```

	A-Z artist	123 albums_count
1	Daft Punk	1
2	Kendrick Lamar	1
3	Ludovico Einaudi	1
4	Miles Davis	1
5	The Beatles	1
6	David Bowie	0
7	Led Zeppelin	0
8	Metallica	0
9	Nirvana	0
10	Pink Floyd	0
11	Queen	0
12	Radiohead	0
13	The Rolling Stones	0

Рисунок 10 – Использование GROUP BY ч.1

На рисунке 11 средняя цена по артистам

```

SELECT ar.name AS artist,
       ROUND(AVG(ri.purchaseprice), 2) AS avg_price
FROM recordinstance ri
JOIN album al ON ri.id_album_fk = al.id_album
JOIN artist ar ON al.id_artist_fk = ar.id_artist
GROUP BY ar.name
ORDER BY avg_price DESC;

```

	A-Z artist	123 avg_price
1	Miles Davis	30
2	The Beatles	25,5
3	Ludovico Einaudi	20

Рисунок 11 – Использование GROUP BY ч.2

На рисунке 12 показано использование HAVING - фильтр уже после группировки. Здесь поиск жанров, у которых количество альбомов больше и равно двух и они вышли после 1980 года.

```

SELECT g.name AS genre,
       COUNT(al.id_album) AS albums_count,
       ROUND(AVG(al.releaseyear), 0) AS avg_release_year
FROM genre g
JOIN album al ON al.id_genre_fk = g.id_genre
GROUP BY g.name
HAVING COUNT(al.id_album) >= 2
       AND AVG(al.releaseyear) > 1980
ORDER BY albums_count DESC, genre ASC;

```

AZ genre	123 albums_count	123 avg_release_year
Jazz	2	1 987

Рисунок 12 – Использование HAVING ч.1