### Министерство науки и высшего образования Российской Федерации

### ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

### НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

### **ITMO University**

Отчет по лабораторной работе № 3
По дисциплине Проектирование баз данных/Базы данных
<b>Обучающиеся</b> Наус Михаил Романович, Савинова Алин- Константиновна
<b>Факультет</b> Факультет технологического менеджмента и инноваций
Группа U3275
Направление подготовки 27.03.05 Инноватика
Образовательная программа Технологии и инновации

Ооучающиеся	паус М. Р. Савинова А. К.
•	•

Преподаватель Ромакина О.М.

#### Основная часть

**Цель работы:** овладеть практическими навыками создания запросов для базы данных PostgreSQL 10 (11), добавление новых атрибутов, выборка различных значений.

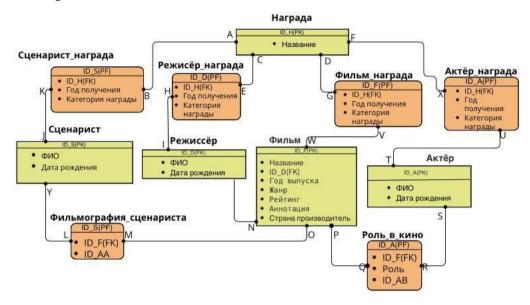
### Практическое задание:

Состоит из создания ряда запросов, а именно:

- 1. Модификация таблицы путем добавления какого-либо атрибута
- 2. Выборка всех данных в таблице
- 3. Выборка различных значений какого-либо столбца таблицы
- 4. Выборка строк таблицы, где значения какого-либо атрибута принадлежат заданному диапазону
- 5. Выборка строк таблицы, где значения какого-либо атрибута принадлежат заданному набору значений
- 6. Выборка строк таблицы, где значения какого-либо атрибута соответствуют заданному шаблону
- 7. Выборка строк таблицы, где значение какого-либо атрибута не пусто
- 8. Сортировка строк таблицы по двум ключам сортировки
- 9. Внутреннее (естественное) соединение таблиц
- 10. Правое соединение таблиц
- 11. Левое соединение таблиц
- 12. Полное соединение таблиц
- 13. Объединение двух таблиц
- 14. Группировка записей по двум или более полям
- 15. Вложенный подзапрос
- 16. Создание модифицируемого представления

### 17. Создание не модифицируемого представления

### 0. Диаграмма IDEF1X.



(Рисунок 1. Диаграмма IDEF1X)

# 1. Модификация таблицы путем добавления какого-либо атрибута.

Для выполнения этого пункта мы напишем следующий запрос (см. Листинг 1):

```
-- Модификация таблицы путем добавления какого-либо атрибута
ALTER TABLE Награды ADD COLUMN Дата_одобрения_награды DATE;
(Листинг 1)
```

Далее мы получим следующий вывод (см. Рисунок 2):

	id_h [PK] integer	<b>Название</b> character varying (100)	<b>Дата_одобрения_награды /</b> date
1	1	Оскар	[null]
2	2	Золотой глобус	[null]
3	3	Бафта	[null]

(Рисунок 2. Вывод таблицы после добавления атрибута)

### 2. Выборка всех данных в таблице.

Для выборки всех данных в таблице (что мы сделали в предыдущем пункте для демонстрации успешного добавления атрибута) используем следующий запрос (см. Листинг 2):

#### SELECT \* FROM Награды;

(Листинг 2)

После чего получим вывод (см. Рисунок 2), ранее мы уже его видели.

**3.** Выборка различных значений какого-либо столбца таблицы. Для выполнения этого запроса мы будем использовать выборку различных значений, однако при игнорировании дублирования (см. Листинг 3):

#### SELECT DISTINCT Имя\_Фамилия FROM Режиссёр;

(Листинг 3)

После чего получим следующий вывод (см. Рисунок 3):

Ž	Имя_Фамилия character varying (100)
1	Ольга Кузнецова
2	Дмитрий Орлов
3	Сергей Сидоров

(Рисунок 3)

## 4. Выборка строк таблицы, где значения какого-либо атрибута принадлежат заданному диапазону.

Для выполнения этого запроса мы будем использовать следующий запрос, где будет идти выборка элементов в диапазоне 2000-2019 года получений наград режиссерами (см. Листинг 4):

### SELECT \* FROM Награды\_сценариста WHERE Year\_of\_awards BETWEEN 2000 AND 2019; (Листинг 4)

После чего мы получим следующий вывод в нашем рабочем окружении (см. Рисунок 4):

	id_h integer	id_scenarist_awards integer	year_of_awards smallint	category_name category_awards
1	1	1	2015	personal
2	2	2	2018	group

(Рисунок. 4. Вывод данных в ограничениях)

# 5. Выборка строк таблицы, где значения какого-либо атрибута принадлежат заданному набору значений.

Для реализации данного пункта мы будем использовать ключевое слово **IN**, для задания определенного конкретного соответствия с набором данных, составим следующий запрос (см. Листинг 5):

### SELECT \* FROM Hаграды\_актера WHERE Category\_name IN ('personal', 'other'); (Листинг 5)

После чего можем увидеть следующую картину на выходе (см. Рисунок 5):

	id_h integer €	id_actor_awards integer	year_of_awards smallint	category_name category_awards
1	1	1	2018	personal
2	3	3	2022	other

(Рисунок 5)

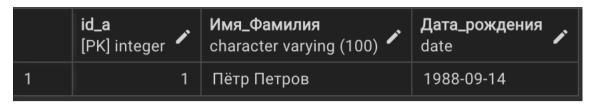
# 6.Выборка строк таблицы, где значения какого-либо атрибута соответствуют заданному шаблону.

Здесь мы будем использовать шаблон данных с которым нужно будет сделать соответствие и вывести все удовлетворяющие ему набор данных (см. Листинг 6):

SELECT \* FROM Актёр WHERE Имя\_Фамилия LIKE 'Петр %';

(Листинг 6)

Далее мы получим следующий вывод (см. Рисунок 6):



(Рисунок 6. Вывод данных соответствующих шаблону)

# 7. Выборка строк таблицы, где значение какого-либо атрибута не пусто.

Здесь мы будем использовать уже другую конструкцию из ключевых слов, а именно **IS NOT NULL**, что прямо говорит нам, данные не могут быть пустыми. Давайте составим запрос (см. Листинг 7):

#### SELECT \* FROM Фильм WHERE Аннотация IS NOT NULL;

(Листинг 7)

Теперь мы можем увидеть следующий вывод (см. Рисунок 7):



(Рисунок 7. Вывод не пустых данных)

### 8. Сортировка строк таблицы по двум ключам сортировки.

Для выполнения этого задания мы воспользуемся следующим запросом (см. Листинг 8):

SELECT \* FROM Фильмография сценариста ORDER BY ID F ASC, ID S DESC;

(Листинг 8)

После чего получим следующий вывод (см. Рисунок 8):

	id_aa [PK] integer	id_f integer ✓	id_s integer /
1	1	1	1
2	2	2	2
3	3	3	3

(Рисунок 8)

### 9. Внутреннее (естественное) соединение таблицы.

Для выполнения этого запроса мы будем использовать следующий запрос (см. Листинг 9):

SELECT \* FROM Награды\_сценариста INNER JOIN Награды\_режиссера ON Награды\_сценариста.ID\_H = Награды\_режиссера.ID\_H;

(Листинг 9)

После чего получим следующий вывод (см. Рисунок 9):

	id_h integer	id_scenarist_awards integer		category_name category_awards	id_h integer	id_rezhiser_awards integer	year_of_awards smallint	category_name category_awards
1			2015	personal			2012	personal
2			2018	group			2016	group
3			2020	other			2019	other

(Рисунок 9)

### 10. Правое соединение таблиц.

Для выполнения этого задания воспользуемся знаниями о ключевом слове **RIGHT JOIN**, используя его напишем запрос

(см. Листинг 10):

SELECT \* FROM Награды\_сценариста RIGHT JOIN Награды\_фильма ON Награды\_сценариста.ID\_H = Награды\_фильма.ID\_H;

(Листинг 10)

Далее можем лицезреть следующий вывод (см. Рисунок 10):



(Рисунок 10)

### 11. Левое соединения таблиц.

Для выполнения этого пункта, мы будем использовать противоположное ключевое слово – **LEFT JOIN**, благодаря ему напишем следующий запрос (см. Листинг 11):

И получим вот такой вывод (см. Рисунок 11):

	id_h integer	id_actor_awards integer	year_of_awards smallint	category_name category_awards	id_h integer	Название character varying (100)	Дата_одобрения_награды date
1			2018	personal		Оскар	[null]
2			2020	group		Золотой глобус	[null]
3			2022	other		Бафта	[null]

(Рисунок 11)

### 12. Полное соединение таблиц.

Для выполнения этого пункта нам потребуются знания, как работает **FULL OUTER JOIN,** после чего мы сможем составить следующий запрос (см. Листинг 12):

SELECT \* FROM Haграды\_фильма FULL OUTER JOIN Фильм ON Haграды\_фильмa.ID\_Film\_Awards = Фильм.ID\_F;

(Листинг 12)

После чего получим следующий вывод (см. Рисунок 12):



(Рисунок 12)

### 13. Объединение двух таблиц.

Для выполнения этого пункта мы составим запрос с помощью ключевого слова **UNION** (см. Листинг 13):

SELECT \* FROM Актёр UNION SELECT \* FROM Роль\_в\_кино; (Листинг 13)

Ну и можем ознакомиться с полученным выводом (см. Рисунок 13):

	id_a integer	Имя_Фамилия character varying (100)	Дата_рождения date
1	2	Ольга Кузнецова	1973-03-17
2	1	Пётр Петров	1988-09-14
3	2	Елена Иванова	1990-02-28
4	3	Анна Смирнова	1977-11-11
5	1	Сергей Сидоров	1955-12-01
6	3	Дмитрий Орлов	1985-06-25

(Рисунок 13)

### 14. Группировка записей по двум или более полям.

Для выполнения этого запроса мы воспользуемся ключевым словом

**– GROUP BY,** составим же запрос (см. Листинг 14):

SELECT Название, Рейтинг, COUNT(\*) FROM Фильм GROUP BY Название, Рейтинг; (Листинг 14)

После можем ознакомиться с выводом (см. Рисунок 14):

	Название character varying (100)	<b>Рейтинг</b> numeric (3,1)	count bigint
1	Новая жизнь	6.4	1
2	Великий фильм	8.5	1
3	Мир будущего	7.9	1

(Рисунок 14)

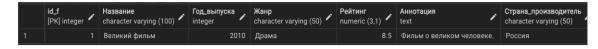
### 15. Вложенный подзапрос.

Для выполнения этого пункта мы воспользуемся еще одним запросом для упрощения нашей жизни, давайте посмотрим на следующий запрос (см. Листинг 15):

SELECT \* FROM Фильм WHERE Рейтинг = (SELECT MAX(Рейтинг) FROM Фильм);

(Листинг 15)

Получим такой вот вывод (см. Рисунок 15):



(Рисунок 15)

### 16. Создание модифицируемого представления.

Чтобы выполнить этот пункт нам потребуется знание ключевого слова **VIEW**, мы будем использовать его в следующем запросе (см. Листинг 16):



(Листинг 16)

Чтобы получить вывод, воспользуемся аналогичной командой, как из пункта 2 (см. Рисунок 16):



(Рисунок 16)

### 17. Создание не модифицируемого представления.

Тут все проще, потому что представления по умолчанию не модифицируемые (см. Листинг 17):

CREATE VIEW View\_Награды\_актера AS SELECT ID\_H, ID\_Actor\_Awards, Year\_of\_awards, Category FROM Награды\_актера;

(Листинг 17)

Ознакомимся с данными (см. Рисунок 17):

	id_h integer	id_actor_awards integer	year_of_awards smallint	category_name category_awards
1	1	1	2018	personal
2	2	2	2020	group
3	3	3	2022	other

(Рисунок 17)

### Вывод

В ходе работы мы смогли научиться строит работающие запросы для баз данных и получать нужные данные различными способами. Также мы ознакомились с довольно широким спектром команд, которые позволяют манипулировать с выборкой данных так, как нам нужно. С полным кодом этой лабораторной работы можно ознакомиться на гитхабе у одного из участников команды:

https://github.com/fallayn/Database\_Laboratory\_No\_4.git