НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені Ігоря Сікорського»

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

Кафедра системного програмування та спеціалізованих комп'ютерних систем

Лабораторна робота №1

з дисципліни

«Бази даних і засоби управління»

Тема: «Проектування бази даних та ознайомлення з базовими операціями СУБД PostgreSQL»

Виконав: студент III курсу

ФПМ групи КВ-84

Лазуткин.

Перевірив:

Київ – 2020

Варіант (опис обраної предметної галузі):

База даннх для обробки кинопоказу

Вимоги до звітування щодо пунктів 1-4 завдання:

У звіті щодо пункту №1 завдання має бути:

- перелік сутностей з описом їх призначення;
- графічний файл розробленої моделі «сутність-зв’язок»;
- назва нотації.

У звіті щодо пункту №2 завдання має бути:

- опис процесу перетворення (наприклад, “сутність А було перетворено у таблицю А, а зв’язок R (M:N) зумовив появу додаткової таблиці R1 тощо);
- схему бази даних у графічному вигляді **з назвами таблиць (!) та зв’язками між ними.**

У звіті щодо пункту №3 завдання має бути:

- пояснення щодо відповідності схеми бази даних нормальним формам НФ1, НФ2 та НФ3. У випадку невідповідності надати опис необхідних змін у схемі;
- У випадку проведення змін у схемі бази даних надати оновлену версію схеми, інакше - не наводити схему.

У звіті щодо пункту №4 завдання має бути:

- навести копії екрану з pgAdmin4, що відображають назви та типи

стовпців (доступне у закладці “Columns” властивостей “Properties” таблиць дерева об’єктів у pgAdmin4);

- навести копії екрану з pgAdmin4, що відображають вміст таблиць бази даних у PostgreSQL. Таблиці на зображенні обов'язково **повинні мати назву**!

Додаток А. Концептуальна модель предметної області

Перелік сутностей з описом їх призначення:

Сутність "Viewer" з атрибутам: Name,Surname,Date of Birth. Призначенна для встановленя особистості відвідувача

Сутність "Ticket" з атрибутам:Film demonstration ID,Price,Seat

Призначенна для встановлення кінопоказу та місця посадки глядача

Сутність "Film demonstration" з атрибутом:Date

Призначенна для встановлення дати показу фільма

Сутність "Film" з атрибутам:Title,duration,agelimit

Призначенна для встановлення фільму що пакузується на кінопоказі

Сутність "Producer" з атрибутам:Name,Surname

Призначенна для встановлення продюсера що створив фільм що показується

Між Сутностями "Viewer" і "Ticket" звязок 1:N оскільки 1 глядач може купити декілька квитків но один квиток не може мати двух глядачів які його купили

Між Сутностями "Film demonstration" і "Ticket" звязок 1:1 оскільки 1 квиток дає прохід на 1 кінопоказ

Між Сутностями "Film demonstration" і "Film" звязок N:1 оскільки один фільм може мати багато кінопоказів но на кінопоказі не може бути більше одного фільма

Між Сутностями "Producer" і "Film" звязок N:N оскільки у одного фільма може бути декілька продюсерів і у один продюсер може працювати над декількаами фільмами

Графічна модель створенна в drawIo

Графічний файл розробленої моделі «сутність-зв’язок»:

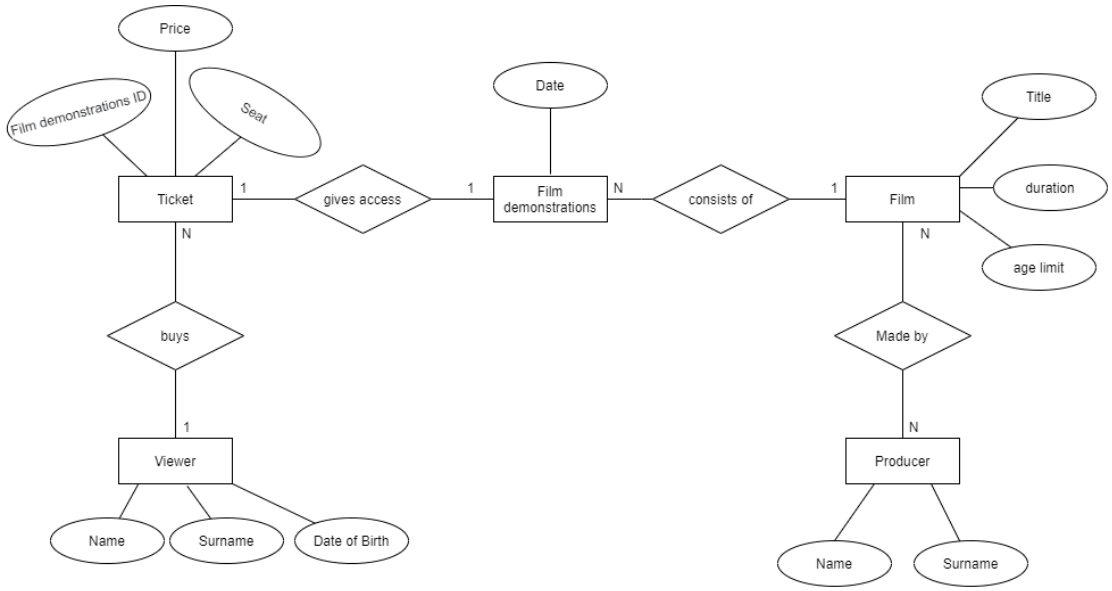


Рисунок 1 - Концептуальна модель предметної області

Додаток Б. Логічна модель (схема) БД

Опис процесу перетворення: Сутності “Viewer”, “Ticet” ,"Film demonstration" "Film" та “Producer” було перетворено у таблиці з відповідними . Зв’язок “Made by” зумовив появу додаткової таблиці

“Film_producer".

Модель побудована в dbdesigner.

Схема бази даних у графічному вигляді:

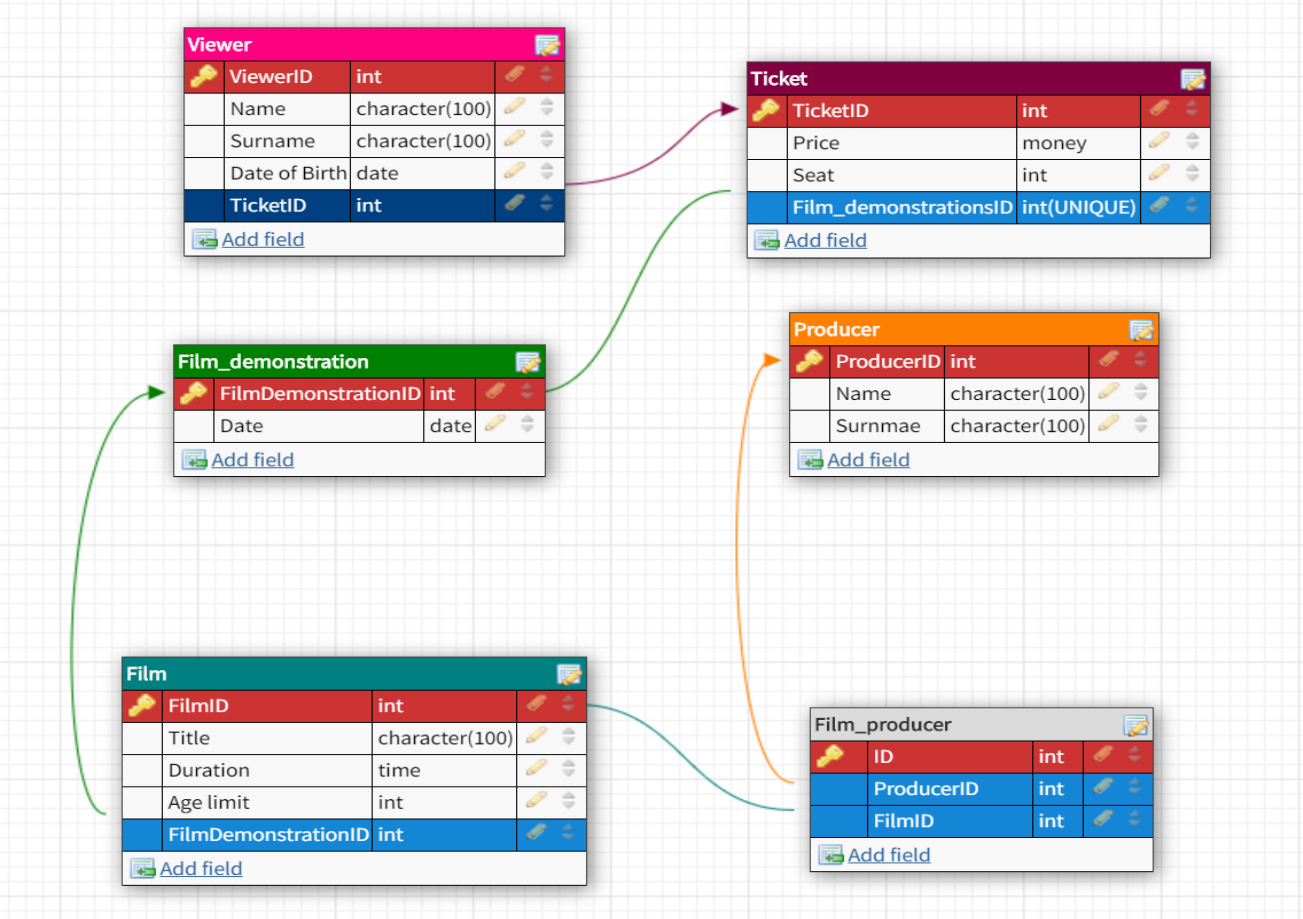


Рисунок 2 - Логічна модель предметної області

Звіт щодо пункту №3 завдання:

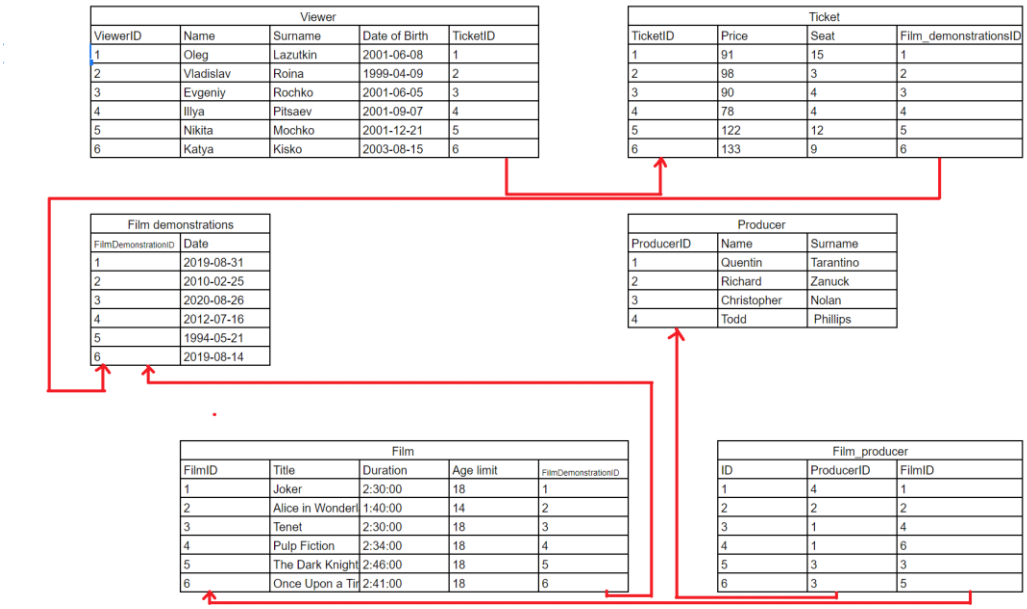
Пояснення щодо відповідності схеми бази даних нормальним формам:

1.Схема бази даних відповідає 1НФ тому, що всі рядки унікальні, всі атрибути прості і не мають нереляційних структур

2.Схема бази даних відповідає 2НФ тому, що вона відповідає 1НФ і всі не ключові атрибути залежать від первинного ключа.

3.Схема бази даних відповідає 3НФ тому, що вона відповідає 2НФ і всі не ключові атрибути нетранзитивно залежні від первинного ключа.

Додаток В. Структура БД “Film demonstration”



Viewer

ViewerID	Name	Surname	Date of Birth	TicketID
1	Oleg	Lazutkin	2001-06-08	1
2	Vladislav	Roina	1999-04-09	2
3	Evgeniy	Rochko	2001-06-05	3
4	Illya	Pitsaev	2001-09-07	4
5	Nikita	Mochko	2001-12-21	5
6	Katya	Kisko	2003-08-15	6

Ticket

TicketID	Price	Seat	Film_demonstrationsID
1	91	15	1
2	98	3	2
3	90	4	3
4	78	4	4
5	122	12	5
6	133	9	6

Film demonstrations

FilmDemonstrationID	Date
1	2019-08-31
2	2010-02-25
3	2020-08-26
4	2012-07-16
5	1994-05-21
6	2019-08-14

Producer

ProducerID	Name	Surname
1	Quentin	Tarantino
2	Richard	Zanuck
3	Christopher	Nolan
4	Todd	Phillips

Film

FilmID	Title	Duration	Age limit	FilmDemonstrationID
1	Joker	2:30:00	18	1
2	Alice in Wonderl	1:40:00	14	2
3	Tenet	2:30:00	18	3
4	Pulp Fiction	2:34:00	18	4
5	The Dark Knight	2:46:00	18	5
6	Once Upon a Ti	2:41:00	18	6

Film_producer

ID	ProducerID	FilmID
1	4	1
2	2	2
3	1	4
4	1	6
5	3	3
6	3	5

Рисунок 3 - Структурна модель предметної області

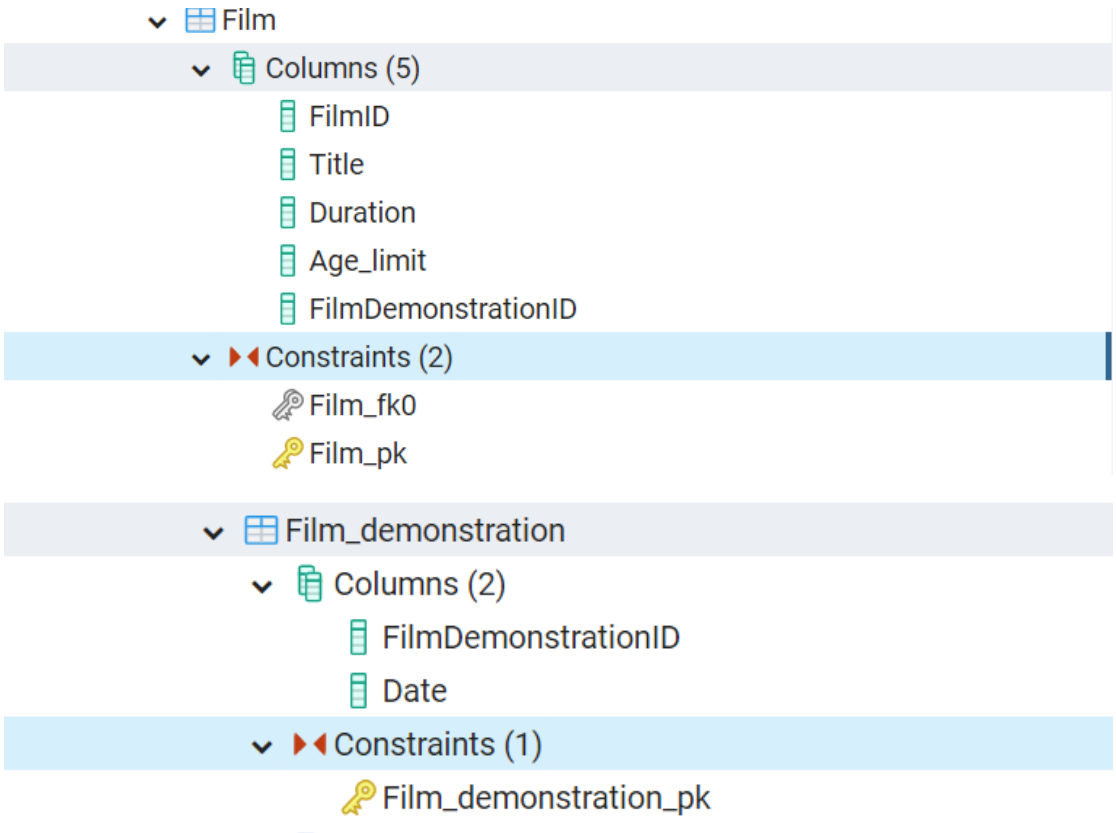
Додаток Г. Опис структури БД

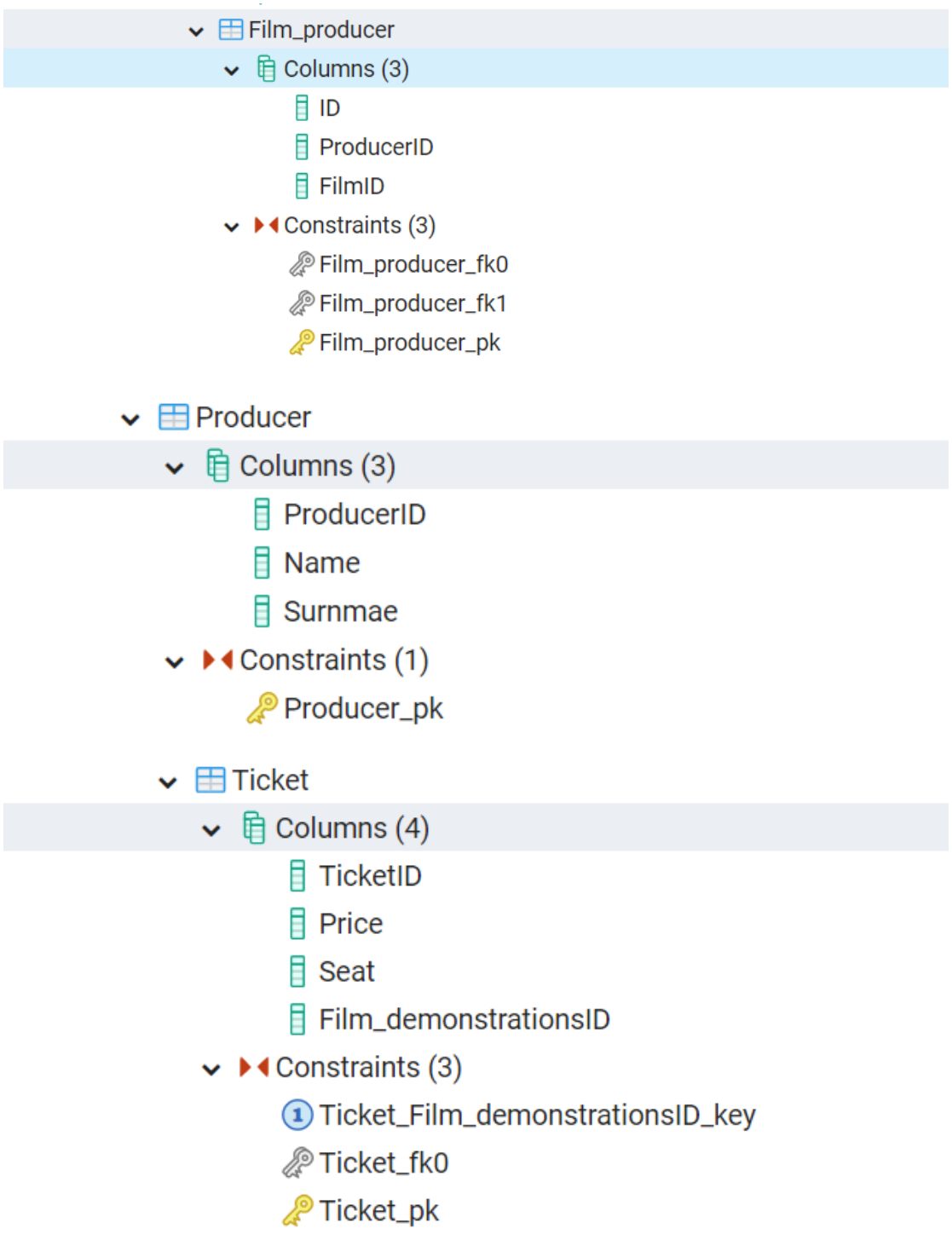
Текстове представлення логічної моделі (схеми) БД

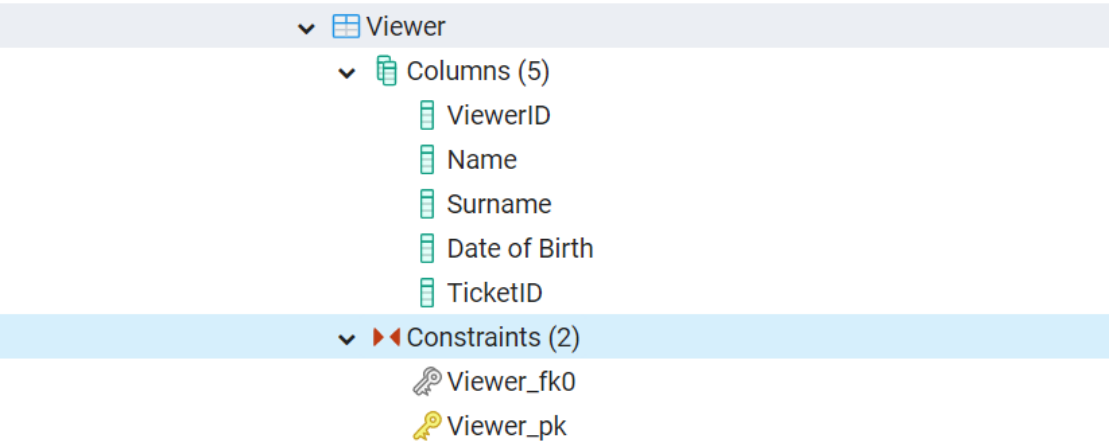
Відношення	Атрибут	Тип
Відношення “Viewer” містить інформацію про глядача	ViewerId – унікальний ID глядача в БД Name – ім’я глядача. Не допускає NULL. Унікальне Surname - призвіще глядача. Не допускає NULL. Унікальне Date of Birth- Дата народження глядача .Не допускає NULL. TicketID-унікальне ID квитка. Не допускає NULL	Числовий Текстовий(100) Текстовий(100) Числовий Числовий
Відношення ”Ticket” містить інформацію про квиток	TicketId – унікальний ID квитка в БД. Price– Ціна квитка. Не допускає NULL. Унікальне Seat – номер посадочного місця. Не допускає NULL. Film_demonstrationId – унікальний ID кінопоказу. Не	Числовий Числовий Числовий Числовий

	допускає NULL.	
Відношення ”Film Demonstration ” містить інформацію про кінопоказ	Film_demonstrationId – унікальний ID кінопоказу в БД. Date-дата кінопоказу	Числовий Числовий
Відношення ”Producer” містить інформацію про продюсер.	ProducerId – унікальний ID продюсера в БД. Name – ім’я продюсера Не допускає NULL. Surname – прізвище продюсера. Не допускає NULL.	Числовий Текстовий(100) Текстовий(100)
Відношення “Film” містить інформацію про фільм	FilmId – унікальний ID фільма в БД Title – назва фільма. Не допускає NULL. Duration- тривалість фільма. Не допускає NULL. Agelimit- вікове обмеження для фільма. Не допускає NULL. Film_demonstrationId – унікальний ID кінопоказу. Не допускає NULL.	Числовий Текстовий(100) Числовий Числовий Числовий
Відношення “Film_producer” містить інформацію про продюсера фільму разом з фільмом	Id – унікальний ID в БД ProducerId – унікальний ID продюсера в БД. FilmId – унікальний ID фільма в БД	Числовий Числовий Числовий

Додаток Г. Структура БД




Film_producer
Columns (3)
ID
ProducerID
FilmID
Constraints (3)
Film_producer_fk0
Film_producer_fk1
Film_producer_pk
Producer
Columns (3)
ProducerID
Name
Surnmae
Constraints (1)
Producer_pk
Ticket
Columns (4)
TicketID
Price
Seat
Film_demonstrationsID
Constraints (3)
Ticket_Film_demonstrationsID_key
Ticket_fk0
Ticket_pk



Фотографії таблиць БД

```
CREATE TABLE public."Viewer"
(
    "ViewerID" integer NOT NULL DEFAULT nextval('"Viewer_ViewerID_seq"'::regclass),
    "Name" character(100) COLLATE pg_catalog."default" NOT NULL,
    "Surname" character(100) COLLATE pg_catalog."default" NOT NULL,
    "Date of Birth" date NOT NULL,
    "TicketID" integer NOT NULL,
    CONSTRAINT "Viewer_pk" PRIMARY KEY ("ViewerID"),
    CONSTRAINT "Viewer_fk0" FOREIGN KEY ("TicketID")
        REFERENCES public."Ticket" ("TicketID") MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
)

TABLESPACE pg_default;

ALTER TABLE public."Viewer"
    OWNER to postgres;
```

```
CREATE TABLE public."Ticket"
(
    "TicketID" integer NOT NULL DEFAULT nextval('"Ticket_TicketID_seq"'::regclass),
    "Price" money NOT NULL,
    "Seat" integer NOT NULL,
    "Film_demonstrationsID" integer NOT NULL,
    CONSTRAINT "Ticket_pk" PRIMARY KEY ("TicketID"),
    CONSTRAINT "Ticket_Film_demonstrationsID_key" UNIQUE ("Film_demonstrationsID"),
    CONSTRAINT "Ticket_fk0" FOREIGN KEY ("Film_demonstrationsID")
        REFERENCES public."Film_demonstration" ("FilmDemonstrationID") MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
)

TABLESPACE pg_default;

ALTER TABLE public."Ticket"
    OWNER to postgres;
```

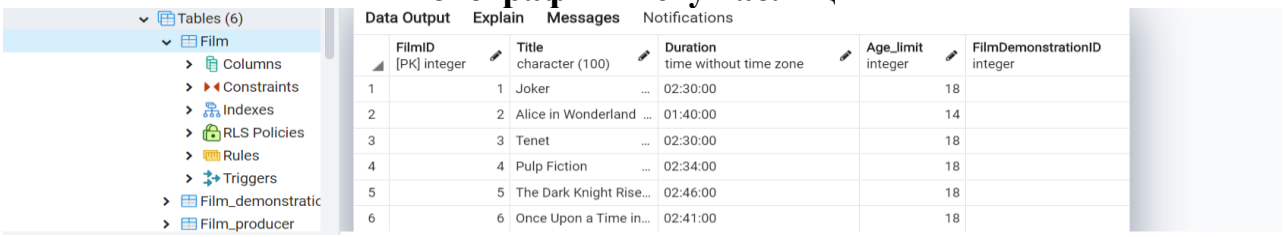
```sql
CREATE TABLE public."Film_demonstration"
(
    "FilmDemonstrationID" integer NOT NULL DEFAULT nextval('"Film_demonstration_FilmDemonstrationID_seq"'::regclass),
    "Date" date NOT NULL,
    CONSTRAINT "Film_demonstration_pk" PRIMARY KEY ("FilmDemonstrationID")
)

TABLESPACE pg_default;

ALTER TABLE public."Film_demonstration"
    OWNER to postgres;
```

```sql
CREATE TABLE public."Producer"
(
    "ProducerID" integer NOT NULL DEFAULT nextval('"Producer_ProducerID_seq"'::regclass),
    "Name" character(100) COLLATE pg_catalog."default" NOT NULL,
    "Surnmae" character(100) COLLATE pg_catalog."default" NOT NULL,
    CONSTRAINT "Producer_pk" PRIMARY KEY ("ProducerID")
)

TABLESPACE pg_default;

ALTER TABLE public."Producer"
    OWNER to postgres;
```

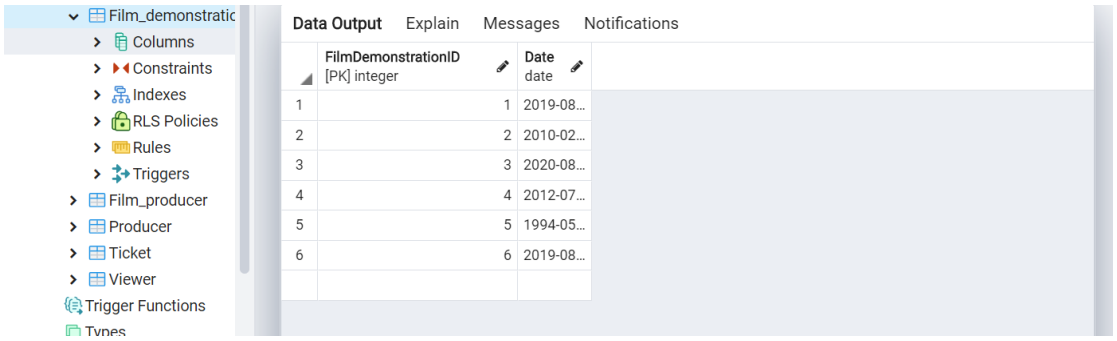
```sql
CREATE TABLE public."Film"
(
    "FilmID" integer NOT NULL DEFAULT nextval('"Film_FilmID_seq"'::regclass),
    "Title" character(100) COLLATE pg_catalog."default" NOT NULL,
    "Duration" time without time zone NOT NULL,
    "Age_limit" integer NOT NULL,
    "FilmDemonstrationID" integer NOT NULL,
    CONSTRAINT "Film_pk" PRIMARY KEY ("FilmID"),
    CONSTRAINT "Film_fk0" FOREIGN KEY ("FilmDemonstrationID")
        REFERENCES public."Film_demonstration" ("FilmDemonstrationID") MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
)

TABLESPACE pg_default;

ALTER TABLE public."Film"
    OWNER to postgres;
```

```sql
CREATE TABLE public."Film_producer"
(
    "ID" integer NOT NULL DEFAULT nextval('"Film_producer_ID_seq"'::regclass),
    "ProducerID" integer NOT NULL,
    "FilmID" integer NOT NULL,
    CONSTRAINT "Film_producer_pk" PRIMARY KEY ("ID"),
    CONSTRAINT "Film_producer_fk0" FOREIGN KEY ("ProducerID")
        REFERENCES public."Producer" ("ProducerID") MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION,
    CONSTRAINT "Film_producer_fk1" FOREIGN KEY ("FilmID")
        REFERENCES public."Film" ("FilmID") MATCH SIMPLE
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
)

TABLESPACE pg_default;

ALTER TABLE public."Film_producer"
    OWNER to postgres;
```

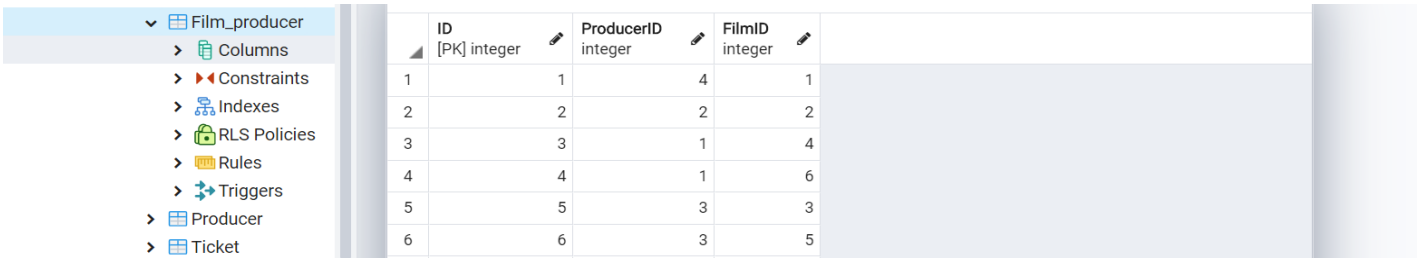
Фотографії вмісту таблиць



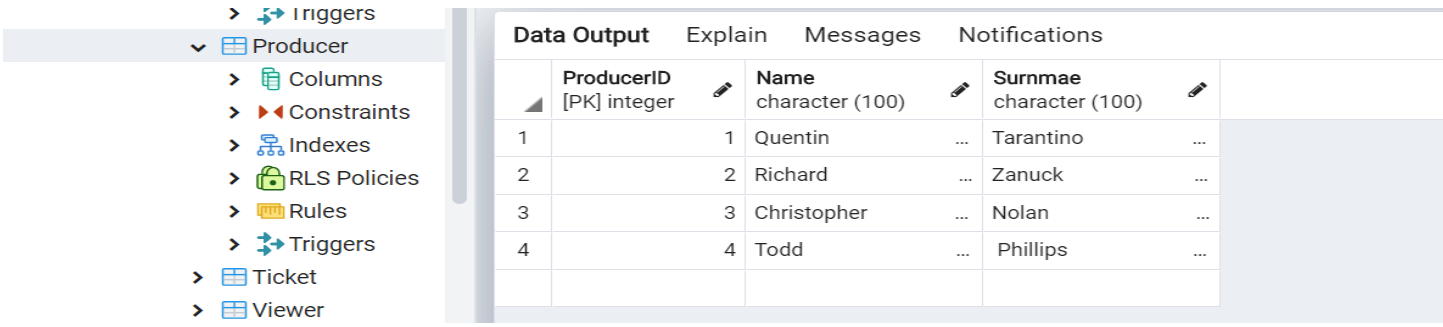
	FilmID [PK] integer	Title character (100)	Duration time without time zone	Age_limit integer	FilmDemonstrationID integer
1	1	Joker	02:30:00	18	
2	2	Alice in Wonderland	01:40:00	14	
3	3	Tenet	02:30:00	18	
4	4	Pulp Fiction	02:34:00	18	
5	5	The Dark Knight Rise...	02:46:00	18	
6	6	Once Upon a Time in...	02:41:00	18	

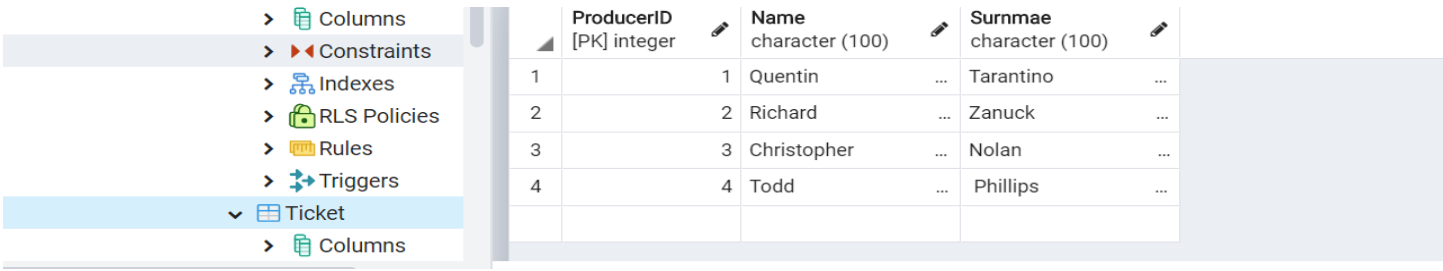


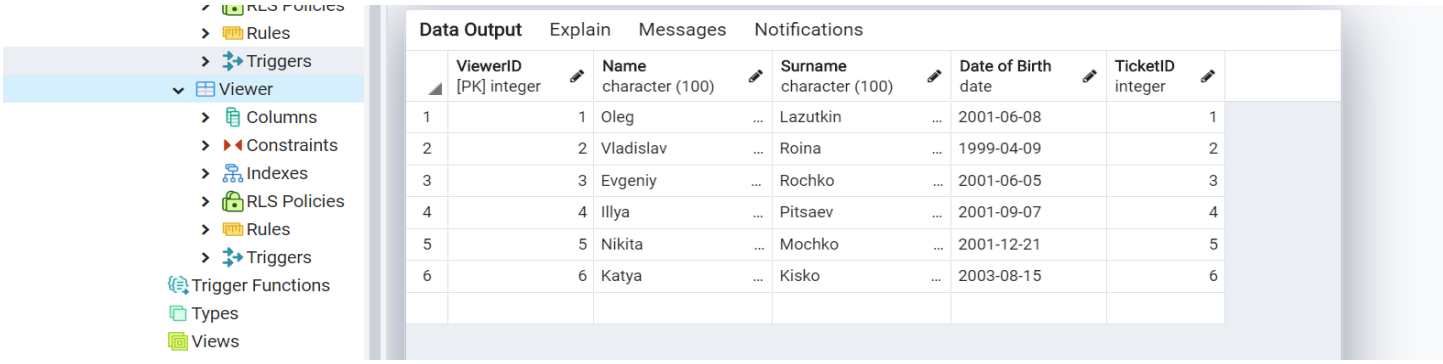
	FilmDemonstrationID [PK] integer	Date date
1	1	2019-08...
2	2	2010-02...
3	3	2020-08...
4	4	2012-07...
5	5	1994-05...
6	6	2019-08...



	ID [PK] integer	ProducerID integer	FilmID integer
1	1	4	1
2	2	2	2
3	3	1	4
4	4	1	6
5	5	3	3
6	6	3	5


Triggers
Producer
Columns
Constraints
Indexes
RLS Policies
Rules
Triggers
Ticket
Viewer
Data Output
Explain
Messages
Notifications
ProducerID [PK] integer
Name character (100)
Surnmae character (100)
1 1 Quentin ... Tarantino ...
2 2 Richard ... Zanuck ...
3 3 Christopher ... Nolan ...
4 4 Todd ... Phillips ...


Columns
Constraints
Indexes
RLS Policies
Rules
Triggers
Ticket
Columns
ProducerID [PK] integer
Name character (100)
Surnmae character (100)
1 1 Quentin ... Tarantino ...
2 2 Richard ... Zanuck ...
3 3 Christopher ... Nolan ...
4 4 Todd ... Phillips ...


Rules
Triggers
Viewer
Columns
Constraints
Indexes
RLS Policies
Rules
Triggers
Trigger Functions
Types
Views
Data Output
Explain
Messages
Notifications
ViewerID [PK] integer
Name character (100)
Surname character (100)
Date of Birth date
TicketID integer
1 1 Oleg ... Lazutkin ... 2001-06-08 1
2 2 Vladislav ... Roina ... 1999-04-09 2
3 3 Evgeniy ... Rochko ... 2001-06-05 3
4 4 Illya ... Pitsaev ... 2001-09-07 4
5 5 Nikita ... Mochko ... 2001-12-21 5
6 6 Katya ... Kisko ... 2003-08-15 6