ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«ПОВОЛЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ И ИНФОРМАТИКИ»

Факультет 1

**ОТЧЕТ**

**ПО ЛАБОРАТОРНОЙ РАБОТЕ №** 3

по дисциплине Системы управления базами данных

|  |  |  |
| --- | --- | --- |
| **ВЫПОЛНИЛ** | | |
| студент | РПИС-11  (группа) | Яковлев П. Н.  (ФИО) |
|  | **ПРОВЕРИЛ** |  |
| (должность) | | Борисов В.В.  (ФИО) |

Самара

2023

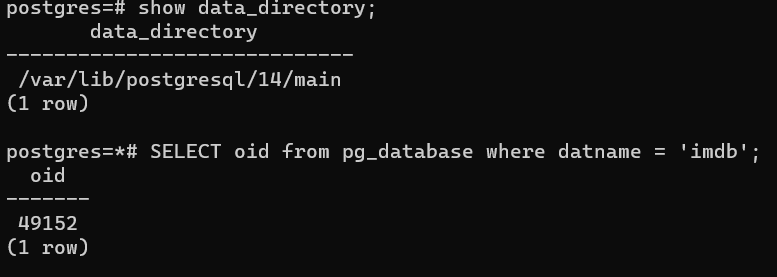
**Директория базы:**

show data\_directory;

чтобы узнать, где хранятся файлы вашей базы. После этого выполните:

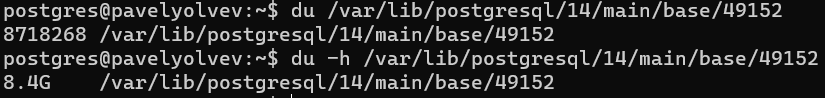
SELECT oid from pg\_database where datname = 'imdb';

чтобы узнать идентификатор базы.

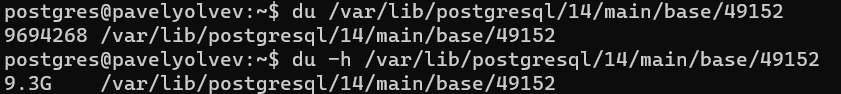


Директория с файлами базы находится по пути:  
/var/lib/postgresql/14/main/base/49152

**Размер базы в байтах и гигабайтах соответственно:**



После создания индексов:



**Создание условного индекса**

CREATE INDEX index\_name on title (kind\_id, production\_year) where (production\_year>2000 and kind\_id=1);



**Создание внешнего ключа для таблицы movie\_info и просмотр ее внешних ключей:**

**Пример создания внешнего ключа:**

alter table movie\_info add foreign key (movie\_id) references title(id);

Запрос для просмотра существующих внешних ключей:

SELECT

tc.constraint\_name, tc.table\_name, kcu.column\_name,

ccu.table\_name AS foreign\_table\_name,

ccu.column\_name AS foreign\_column\_name

FROM

information\_schema.table\_constraints AS tc

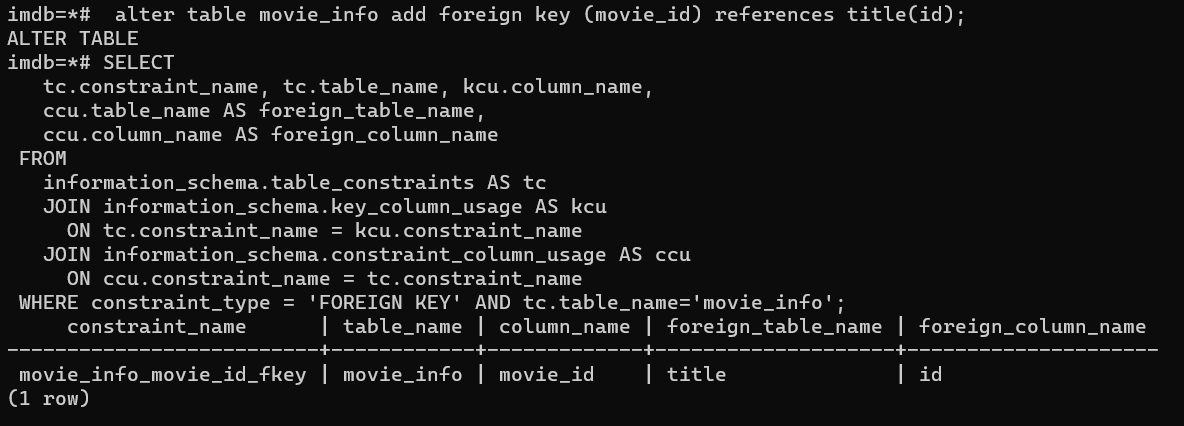
JOIN information\_schema.key\_column\_usage AS kcu

ON tc.constraint\_name = kcu.constraint\_name

JOIN information\_schema.constraint\_column\_usage AS ccu

ON ccu.constraint\_name = tc.constraint\_name

WHERE constraint\_type = 'FOREIGN KEY' AND tc.table\_name='movie\_info';



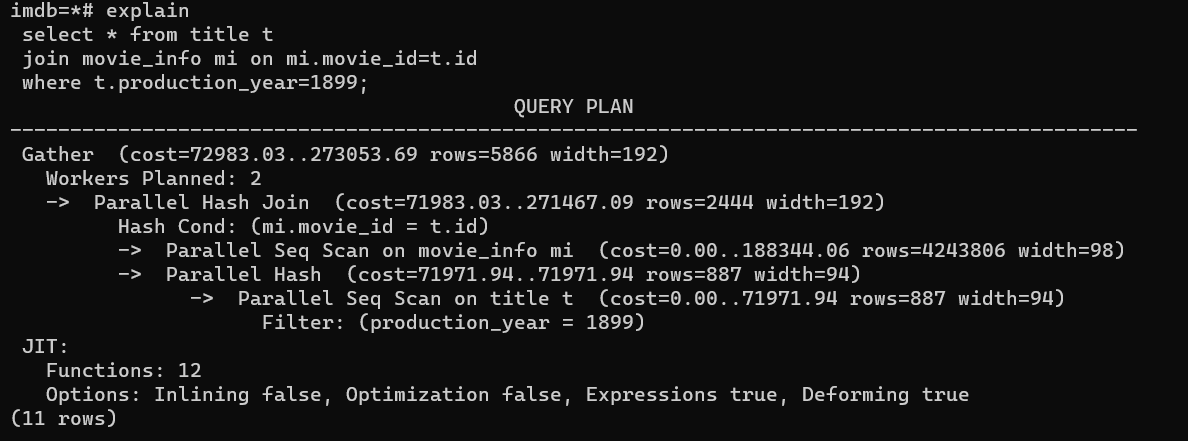
Пример плана запроса после создания внешнего ключа:

explain

select \* from title t

join movie\_info mi on mi.movie\_id=t.id

where t.production\_year=1899;



**Использование EXPLAIN:**

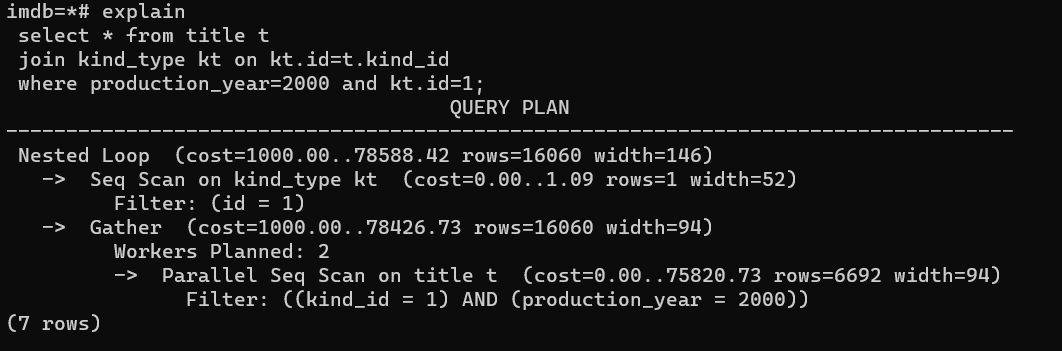
Пример команды:

explain

select \* from title t

join kind\_type kt on kt.id=t.kind\_id

where production\_year=2000 and kt.id=1;

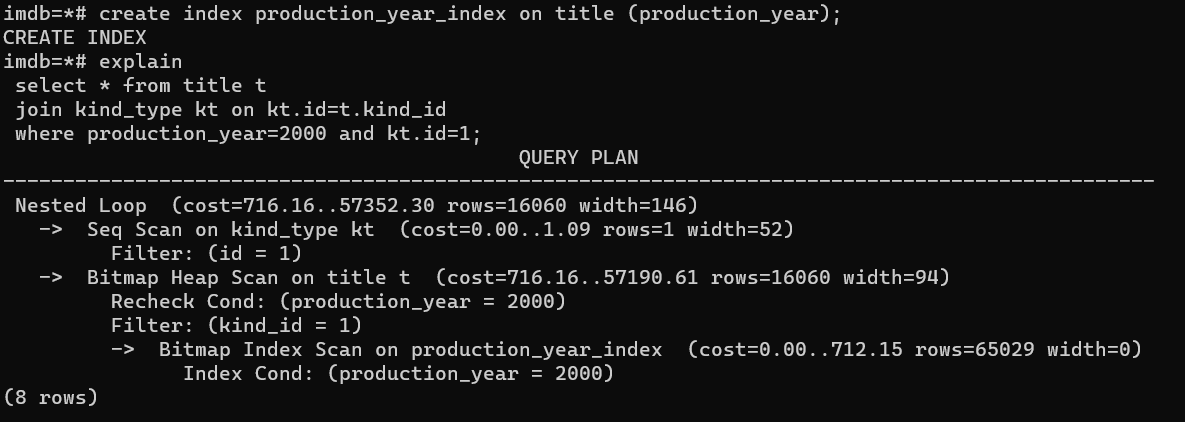


Здесь Nested Loop означает, что внутри запроса выполняется цикл. Этот цикл состоит из последовательного просмотра записей через первичный ключ (Seq Scan).

Если добавить индекс:

create index production\_year\_index on title (production\_year);

То план запроса изменится:

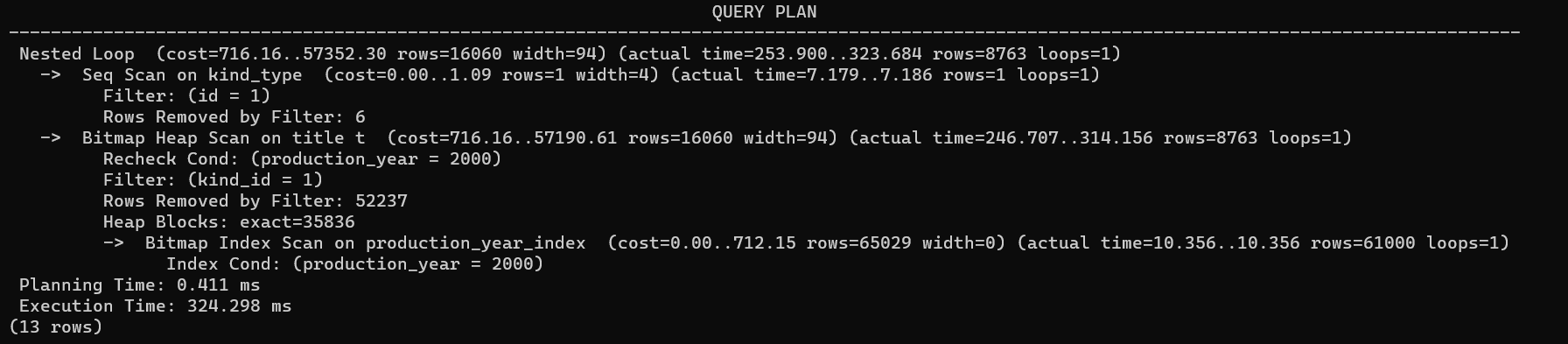


**EXPLAIN ANALYZE**

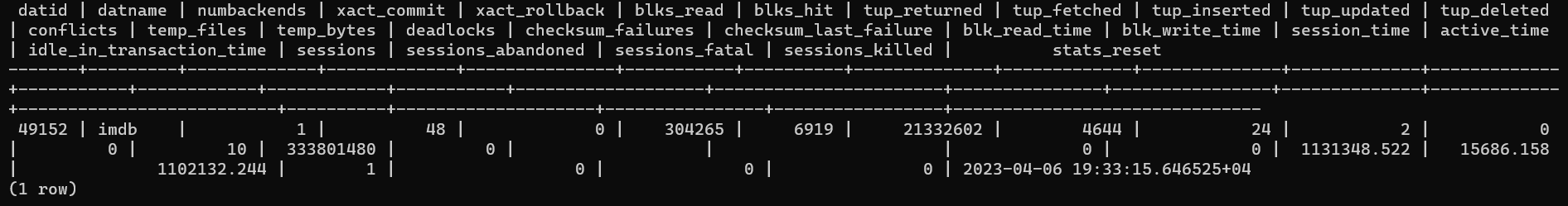
Пример запроса:

explain ANALYZE

select \* from title t where production\_year=2000 and kind\_id in (select id from kind\_type where id=1);



статистики выполнения операций в базе:



**Оптимизация запросов из 2 лабораторной работы:**

**1 запрос:**

select name

from name

where not exists

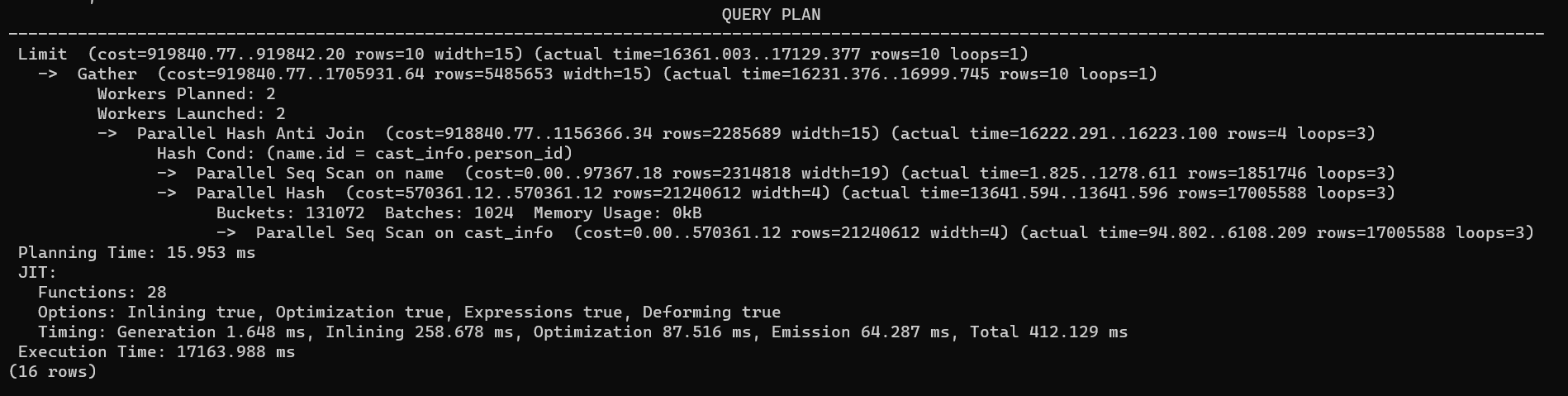
( select cast\_info.person\_role\_id

from cast\_info

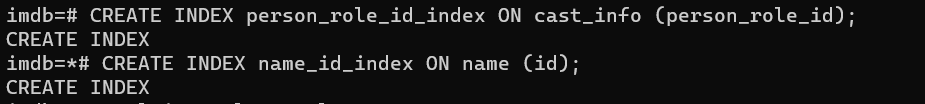
where cast\_info.person\_id=name.id )

limit 10;

Explain Analyze до создания индекса:



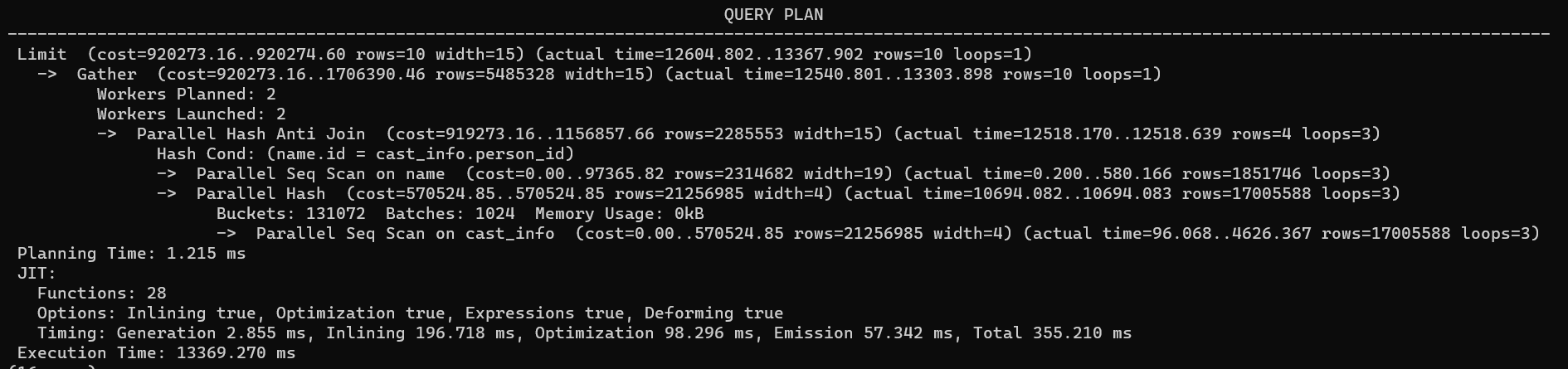
Создание индексов:



CREATE INDEX person\_role\_id\_index ON cast\_info (person\_role\_id);

CREATE INDEX name\_id\_index ON name (id);

После:



SELECT title.title,

COUNT(ci1.movie\_id) as "Employees",

COUNT(ci2.movie\_id) as "Actors"

FROM title

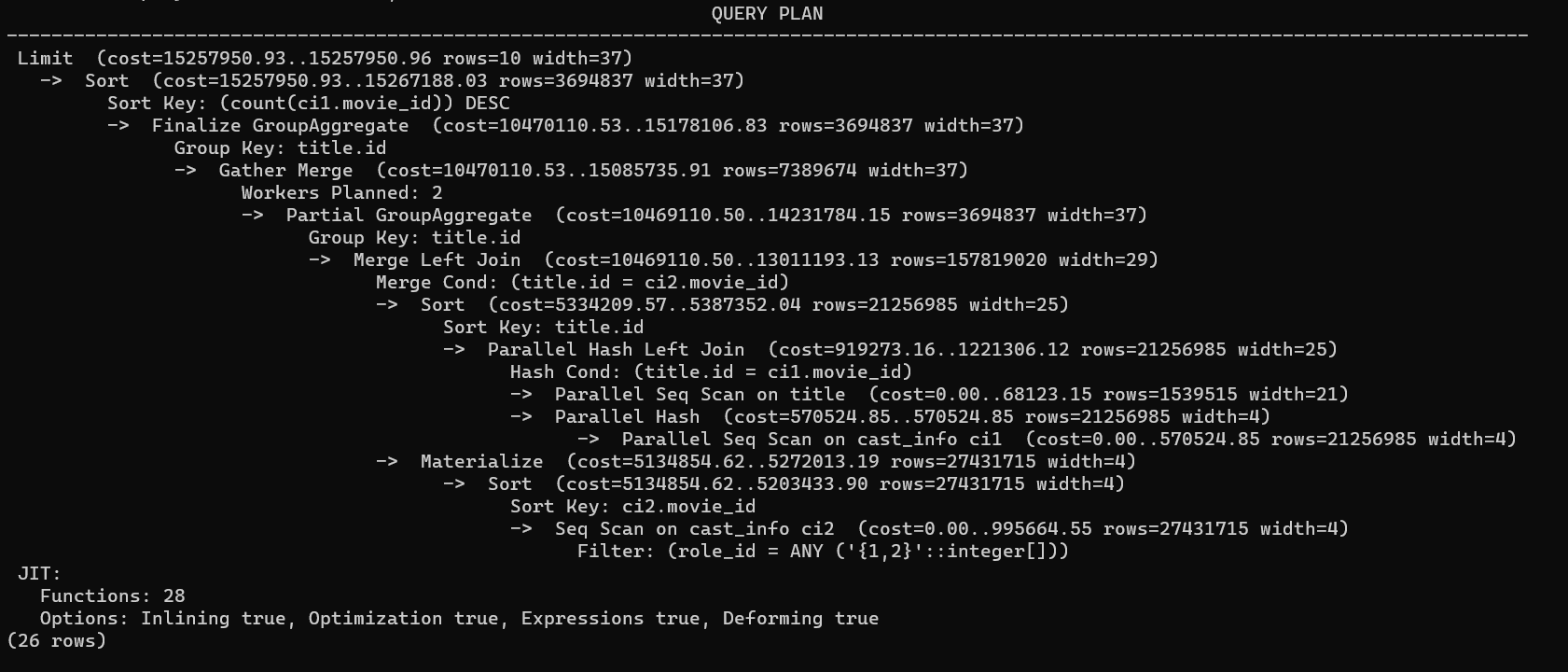
LEFT JOIN cast\_info ci1 ON ci1.movie\_id = title.id

LEFT JOIN cast\_info ci2 ON ci2.movie\_id = title.id AND ci2.role\_id IN (1, 2)

GROUP BY title.id

ORDER BY "Employees" DESC limit 10;

До:



Создание внешних ключей:

alter table movie\_info add foreign key (movie\_id) references title(id);

Создание индексов:

CREATE INDEX movie\_id\_index ON cast\_info (movie\_id);

После:

