МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных технологий

Кафедра Информационные системы и технологии

Специальность 1–40 05 01 Информационные системы и технологии

Специализация 1-40 01 02 03 Информационные системы

и технологии (издательско-полиграфический комплекс)

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**К КУРСОВОМУ ПРОЕКТУ НА ТЕМУ:**

«Проектирование базы данных кинотеатра»

Выполнил студент Почта Кирилл Алексеевич

(Ф.И.О.)

Руководитель проекта доцент Блинова Е.А.

(учен. степень, звание, должность, Ф.И.О., подпись)

Заведующий кафедрой к.т.н., доц. Смелов В.В .

(учен. степень, звание, должность, Ф.И.О., подпись)

Консультант: ассистент Копыток Д.В

(учен. степень, звание, должность, Ф.И.О., подпись)

Нормоконтролер: ассистент Копыток Д.В

(учен. степень, звание, должность, Ф.И.О., подпись)

Курсовой проект защищен с оценкой

Минск 2022

**Содержание**

[Введение 3](#_Toc103250010)

[1 Постановка задачи 4](#_Toc103250011)

[2 Разработка модели базы данных 5](#_Toc103250012)

[3 Разработка необходимых объектов 6](#_Toc103250013)

[3.1 Таблицы 6](#_Toc103250014)

[3.2 Пользователи 8](#_Toc103250015)

[3.3 Хранимые процедуры 8](#_Toc103250016)

[4 Описание процедур импорта и экспорта 9](#_Toc103250017)

[5 Тестирование производительности 11](#_Toc103250018)

[6 Описание технологии и ее применения в базе данных 14](#_Toc103250019)

[7 Руководство пользователя 17](#_Toc103250020)

[Заключение 19](#_Toc103250021)

[Cписок источников 20](#_Toc103250022)

## **Введение**

Любая организация нуждается в своевременном доступе к информации. Ценность информации в современном мире очень высока. Роль распорядителей информации в современном мире чаще всего выполняют базы данных. Базы данных обеспечивают надежное хранение информации, в структурированном виде и своевременный доступ к ней. Практически любая современная организация нуждается в базе данных, удовлетворяющей те или иные потребности по хранению, управлению и администрированию данных.

За последние несколько лет наблюдается тенденция к усложнению структур данных. Простые виды информации, представимой в форме чисел и текстовых строк, не утратив своей значимости, дополняются сегодня многочисленными мультимедийными документами, графическими образами, хронологическими рядами, процедурными, или активными, данными и мириадами прочих сложных информационных форм.

На сегодняшний день на рынке представлено множество технологий доступа к данным и серверов баз данных, каждое, из которых имеет свои отличительные черты. Современные приложения обработки данных ориентированы на работу с большим количеством пользователей, на их удаленность от места расположения основного сервера БД.

Темой данного курсового проекта является разработка базы данных «Кинотеатр».

База данных «Кинотеатр» существенно упрощает работу сотрудников кинотеатров, а также предоставляет возможность администратору своевременно вносить необходимые изменения.

# **Постановка задачи**

Задача проекта: разработать построить базу выполнить тестирование готового продукта.

Функционально должны быть выполнены следующие задачи:

* изменение фильмов, сеансов, залов, режиссёров;
* возможность оформить заказ на определенный сеанс;
* оформление отчетов по прокатам фильмов и их посещаемости
* возможность осуществлять поиск фильмов и фильмов режиссёров;

Должны быть выполнены следующие требования:

* доступ к данным должен осуществляться только через соответствующие процедуры;
* должен быть проведен импорт данных из XML файлов, экспорт данных в формат XML;
* необходимо протестировать производительность базы данных на таблице, содержащей не менее 100 000 строк, и внести изменения в структуру в случае необходимости. Необходимо проанализировать планы запросов к таблице;
* применить технологию базы данных согласно выбранной теме: подробно описать применяемые системные пакеты, утилиты или технологии; показать применение указанной технологии в базе данных.

База данных приложения должна взаимодействовать с Oracle Database 12c.

# **Разработка модели базы данных**

Первым этапом курсового проекта будет создание логически взаимосвязанных таблиц. Чтобы составить визуальную взаимосвязанную структуру базы данных, нам необходимо продумать, какая информация будет храниться в этих таблицах, после этого создать связи с помощью первичных и внешних ключей.

База данных приложения состоит из 8 таблиц, взаимосвязанных между собой внешними ключами:

* HALL;
* SESSIONF;
* DIRECTORS;
* COUNTRY;
* ORDERS;
* CLIENTS;
* FILMS;
* GENRE;

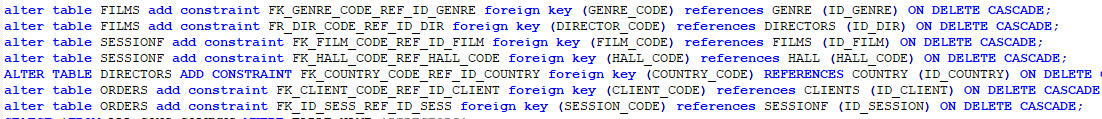


Рис 1 - Связи столбцов

# **Разработка необходимых объектов**

# **Таблицы**

Для реализации базы данных «Кинотеатр» было разработано 8 таблиц: HALL, SESSIONF, GENRE, COUNTRY, Orders, DIRECTORS, FILMS, CLIENTS.

Таблица Clients представляет список клиентов, состоит из столбцов (таблица 3.1):

Таблица 3.1 – Столбцы таблицы Clients

|  |  |  |
| --- | --- | --- |
| Наименование | Описание | Тип |
| ID\_CLIENT | идентификатор клиента, первичный ключ | integer |
| CNUMBER | Номер клиента | nvarchar2 |
| NAME | Имя клиента | nvarchar2 |
| FILM | фильм | nvarchar2 |
| BOUGHT\_TICKETS | Куплено билетов | nvarchar2 |

Таблица Sessionf представляет список сеансов состоит из столбцов (таблица 3.2):

Таблица 3.2 – Столбцы таблицы Sessionf

|  |  |  |
| --- | --- | --- |
| Наименование | Описание | Тип |
| ID\_SESSION | идентификатор сеанса | integer |
| FILM\_CODE | Код фильма | integer |
| NAMEOFGENRE | Название жанра | nvarchar2 |
| HALL\_CODE | Код зала | integer |
| NUMBEROFFREEPLACES | Количество свободных мест | integer |
| ALL\_PLACES | Всего мест на сеанс | integer |
| COST | пароль сотрудника | nvarchar2 |
| DATA\_OF\_SESSION |  | date |

Таблица HALL представляет список залов, таблица состоит из столбцов (таблица 3.3):

Таблица 3.3 – Столбцы таблицы HALL

|  |  |  |
| --- | --- | --- |
| Наименование | Описание | Тип |
| ID\_HALL | Идентификатор зала | int |
| HALL\_CODE | Код зала | int |
| NAME | Название зала | VARCHAR2 |
| NUMBEROFLINES id | Количество рядов | int |
| NUMBEROFPLACES | Количество свободных мест | int |

Таблица CLIENTS представляет список Режиссеров. Таблица состоит из столбцов описывающих режиссеров. (таблица 3.4):

Таблица 3.4 – Столбцы таблицы DIRECTORS

|  |  |  |
| --- | --- | --- |
| Наименование | Описание | Тип |
| ID\_DIR | идентификатор режиссеров | int |
| DIRECTOR\_CODE | Код режиссера | int |
| SURNAME | Фамилия | nvarchar2 |
| NAME | имя | nvarchar2 |
| COUNTRY\_CODE | Код страны | int |

Таблица ORDERS представляет список заказов клиентов, состоит из столбцов (таблица 3.5):

Таблица 3.5 – Столбцы таблицы ORDERS

|  |  |  |
| --- | --- | --- |
| Наименование | Описание | Тип |
| ID\_ORDER | Идентификатор заказа | int |
| CLIENT\_CODE | Код клиента | int |
| SESSION\_CODE | адрес поставщика | int |
| CLIENTNAME | Имя клиента | nvarchar2 |
| COUNT\_OF\_TICKETS | Сколько билетов купил | int |

Таблица FILMS представляет список фильмов, состоит из столбцов (таблица 3.6):

Таблица 3.6 – Столбцы таблицы FILMS

|  |  |  |
| --- | --- | --- |
| Наименование | Описание | Тип |
| ID\_FILM | идентификатор фильма | int |
| FILM\_CODE | Код фильма | int |
| FILM | Название фильма | int |
| GENRE\_CODE | Код жанра | int |
| DIRECTOR\_CODE | Код режиссера | int |
| YEAROFRELEAS | Дата выпуска | date |

Таблица COUNTRY представляет страны, где родились режиссеры, состоит из столбцов (таблица 3.7):

Таблица 3.7 – Столбцы таблицы COUNTRY

|  |  |  |
| --- | --- | --- |
| Наименование | Описание | Тип |
| ID\_COUNTRY number | Идентификатор страны | int |
| NAME | Название страны | int |
| COUNTRY\_CODE | Код страны | int |

Таблица GENRE представляет список жанров фильмов, состоит из столбцов (таблица 3.8):

Таблица 3.8 – Столбцы таблицы GENRE

|  |  |  |
| --- | --- | --- |
| Наименование | Описание | Тип |
| ID\_GENRE | идентификатор жанра | int |
| GENRE\_CODE | Код жанра | int |
| NAMEOFGENRE | Название жанра | char |

# **Пользователи**

Пользователь базы данных – это физическое или юридическое лицо, которое имеет доступ к БД и пользуется услугами информационной системы для получения информации.

При проектировании базы данных было использовано 2 пользователя. Первый пользователь – сотрудник, который является системным администратором, имеет доступ для чтения и изменения таблиц, связанных с клиентами.

Второй пользователь – клиент – имеет доступ к формированию заказов.

# **Хранимые процедуры**

Хранимая процедура – объект базы данных, представляющий собой набор SQL-инструкций, который компилируется один раз и хранится на сервере. При создании хранимой процедуры можно определить необязательный список параметров. Таким образом, процедура будет принимать соответствующие аргументы при каждом ее вызове. Все хранимые процедуры, созданные в данном курсовом проекте, содержат обработку исключений для того, чтобы ограничить пользователя от возможных ошибок.

При разработке курсового проекта было создано 22 процедур для следующих целей:

* выборка данных для таблиц;
* добавление/удаление режиссеров;
* работа режиссерами их фильмами и сеансами, удаление, создание добавление.
* добавление клиента;
* добавления заказа;
* обновление информации о фильмах;
* добавления фильмов;
* экспорт и импорт таблицы в формат xml.

# **Описание процедур импорта и экспорта**

В курсовом проекте был проведен импорт данных из XML файлов и экспорт данных в формат XML.

XML — это формат данных, используемый для обмена данными в форме, которая может быть легко использована и распространена. Часто возникает необходимость импортировать и экспортировать XML-файлы в Oracle, в данной курсовой работе используются пакеты DBMS\_XMLPARSER и DBMS\_XSLPROCESSOR (Пакет DBMS\_XSLPROCESSOR предоставляет интерфейс для управления содержимым и структурой XML-документов.) для импорта xml, и DBMS\_XMLDOM(DOM (от англ. Document Object Model — «объектная модель документа») — это независящий от платформы и языка программный интерфейс, позволяющий программам и скриптам получить доступ к содержимому HTML -, XHTML - и XML -документов, а также изменять содержимое, структуру и оформление таких документов.) для экспорта данных в xml формат. Для работы с файлами так же использовался пакет DBMS\_LOB. Данные пакеты были выбраны, потому что они обладают очень широким функционалом и гибкой настройкой xml документа. В данном курсовом проекте функции экспорта, импорта используются для таблицы COUNTRY. Пример создания процедуры экспорта можно посмотреть на рисунке 4.1.



Рисунок 4.1 – Пример создания процедуры ExportCountryToXml

Для сохранения и управления XML-данными в реляционной таблице применяется специальный тип данных XMLType. Тип данных XMLType поставляется с набором специальных XML-методов, которые можно использовать для работы с объектами XMLType. Эти методы можно применять для выполнения как типичных операций в базе данных, наподобие проверки на предмет существования узла или извлечения узла, так и нескольких специальных операций, позволяющих получать доступ к XML-данными и манипулировать ими в виде части обычного SQL-оператора.

XMLType это системный непрозрачный тип для обработки XML-данных. XMLType имеет предопределенные функции-члены для извлечения XML-узлов и фрагментов.

Вы можете создавать столбцы XMLTypeи вставлять в них XML-документы. Вы также можете создавать XML-документы как XMLTypeэкземпляры динамически, используя функции SYS\_XMLGENи SYS\_XMLAGG SQL.

# **Тестирование производительности**

Для проверки производительности базы данных необходимо заполнить ее большим количеством различных данных и узнать время выполнения одного запроса.

Для данной задачи мы создали анонимный блок, и так как работаем именно с действующей базой данных, то можем использовать готовую процедуру для вставки данных в таблицу. Разработанный анонимный блок позволяет добавить большое количество строк за одно выполнение (рисунок 5.1).

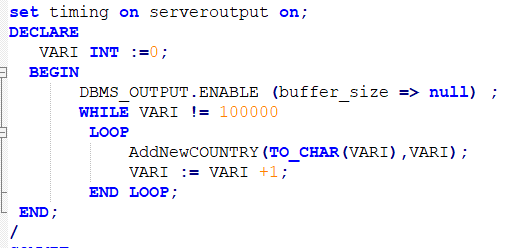


Рисунок 5.1 – Анонимный блок для заполнения таблицы большим количеством данных

Таким образом было добавлено 100000 строк в таблицу[1].

После этого в другом анонимном блоке, который содержит в себе дополнительно переменную для отслеживания времени выполнения, был сделан запрос, который должен возвращать в результате строки соответствующие условию выборки строк. Данный блок показан на рисунке 5.2.

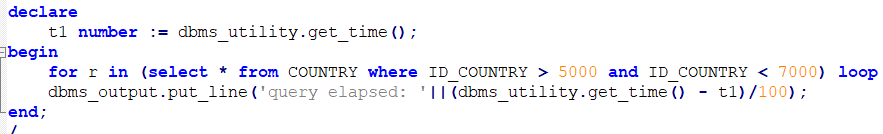


Рисунок 5.2 – Блок отбора данных

Далее после выполнения данного блока наша таблица будет содержать большое количество данных, и мы можем проанализировать время выполнения запроса. Также можно просмотреть план запроса используя стандартные средства Oracle, а именно кнопку на главной панели, предварительно выделив данный запрос. Покажем результат, в котором будет заметна разница во времени выполнение запроса до создания индекса на рисунке 5.3.

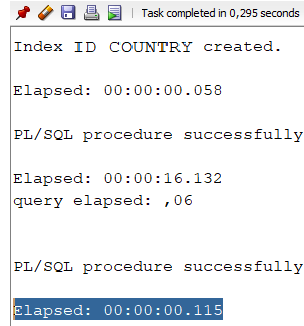


Рисунок 5.3 – Результат отбора данных

Теперь остаётся лишь создать индекс и протестировать время выполнения аналогичного запроса. Покажем это на рисунке 5.4.

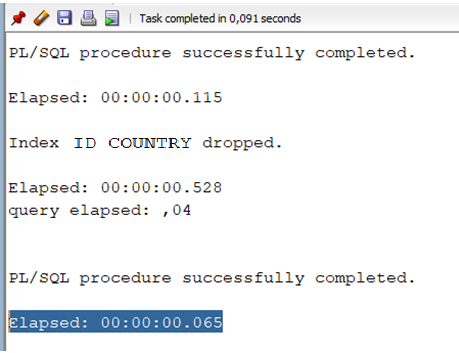


Рисунок 5.4 – Результат после создания индекса

Как видим, время запроса уменьшилось, но тем не менее даже без индекса[1] наша база данных успешно прошла тест на производительность.

# **Описание технологии и ее применения в базе данных**

В процессе разработки проекта была использована технология: резервное копирование и восстановление данных.

Традиционный пользовательский метод резервного копирования состоит в применении команд операционной системы для копирования необходимых файлов в другое место и/или на ленточное устройство.

В процессе разработки проекта резервное копирование и восстановление данных происходило при помощи утилиты RMAN. Резервное копирование файлов базы данных Oracle выполняться внутри базы данных посредством самого сервера баз данных. RMAN[3] умеет делать резервные копии и копии образов файлов данных, управляющих файлов, архивных журналов повторного выполнения, файлов SPFILE и фрагментов резервных копий RMAN.

Возможности RMAN включают следующее:

* выполнение полного резервирования и резервирования изменений;
* выполнение холодного/горячего резервирования;
* обнаружение поврежденных блоков;
* параллельное выполнения операций ввода/вывода;
* автоматическое протоколирование операций копирования и восстановления.

С помощью RMAN можно выполнять инкрементное резервное копирование. Размер резервных копий в таком случае зависит не от размера базы данных, а скорее от уровня активности внутри нее, поскольку во время инкрементного резервного копирования не измененные блоки пропускаются.

Существует ряд сущностей, которые позволяют утилите RMAN выполнять ее функции в области резервного копирования и восстановления. В данном случае была использована целевая база данных (target database). Так называется база данных, в отношении которой RMAN выполняет резервное копирование. Все операции по резервному копированию и восстановлению осуществляются при помощи запускаемых в целевой базе данных серверных сеансов RMAN.

Подключаться к RMAN можно путем ввода в приглашении операционной системы команды rman. После этого будет появляться приглашение RMAN>, позволяющее вводить различные команды RMAN.

Резервирование файлов базы данных: горячее полное резервирование БД и холодное резервирование БД.

Горячее резервирование:

* может выполняться в состоянии СУБД OPEN;
* может выполняться только при включенном режиме архивирования журналов.

В курсовом проекте был использован метод холодного резервирования. Режим архивирования журналов выключен, СУБД выполняется в режиме NOMOUNT. Очевидный недостаток данного метода: из «холодной» резервной копии можно восстановить только то состояние базы данных, которое было в момент останова; транзакции, сделанные после рестарта базы, в «холодную» резервную копию не попадут;

Для резервирования файлов данных сначала необходимо перевести базу данных в в режим архивирования затем открыть бд, далее произвести коннект к таргету rman : rman target /, после этого можно произвести backup database plus archivelog, это позволит сделать резервную копию всей бд, как показано на рисунке 6.1.

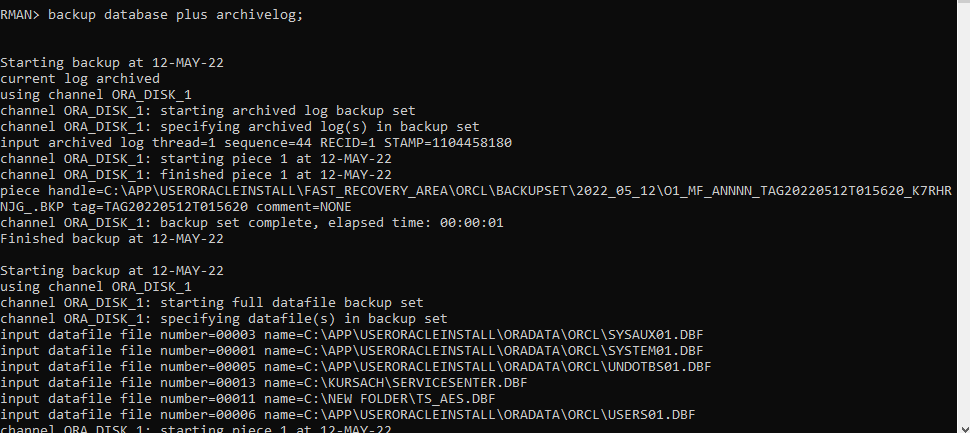


Рисунок 6.1 – Резервирование базы данных

Допустим мы потеряли контрольные файлы и файлы бд, для восстановления бд, сначала необходимо выключить instance – shutdown abort, переведем бд в mount режим – startup nomount; После этого можно восстановить контрольные файлы командой restore controlfile from ‘пусть к куску бэкапа, отвечающего за ’ контрольных файлов используется команда, после этого можно произвести restore database, ибо контрольные файлы восстановлены, а они знают все о файла бд. Результат можно посмотреть на рисунке 6.2.

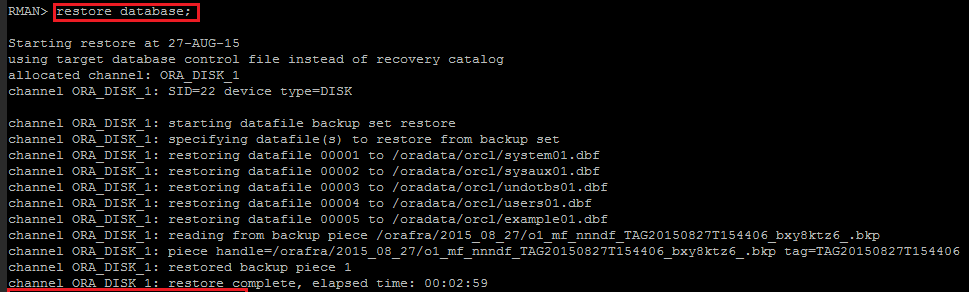


Рисунок 6.2 – Восстановление бд.

Чтобы получить информацию о созданных бэкапах нужно выполнить команду: RMAN> LIST BACKUP; LISTКоманда использует информацию в репозитории RMAN для предоставления списков резервных копий, архивных журналов и воплощений базы данных. Вы можете использовать выходные LISTданные для определения конкретных резервных копий, которые вы хотите использовать с другими командами RMAN, Менеджер резервного копирования и восстановления, поставляемый для баз данных Oracle, созданных корпорацией Oracle..

Восстановление данных. Для восстановления данных целевая БД должна находиться в состоянии NOMOUNT/ MOUNT/ OPEN в зависимости от характера восстановления, например:

* NOMOUNT: для восстановления контрольных файлов БД (фактически – СУБД)
* MOUNT: для восстановления БД целиком или табличного пространства SYSTEM
* OPEN: для восстановление табличных пространств, помимо SYSTEM (в этом случае перед процедурой восстановления само табличное пространство потребуется перевести в состояние OFFLINE).

# **Руководство пользователя**

Наша база данных имеет 22 процедуры, все они имеют примитивный принцип действия, надо написать анонимный блок и название процедуры, как показано на рисунке 7.1

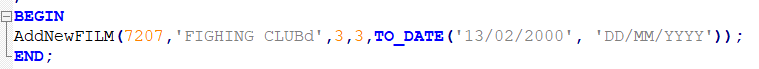


Рисунок 7.1 ­ Вызов процедуры

При успешном выполнении процедуры[3] вызывается пакет dbms и его функция put\_line, которая позволяет вывести сообщение во входной поток как показано на рисунке 7.2.



Рисунок 7.2 Вызов пакета dbms\_output

Так же учитывается возможная ошибка неправильного ввода в поля процедуры как можно увидеть ниже на рисунке 7.3

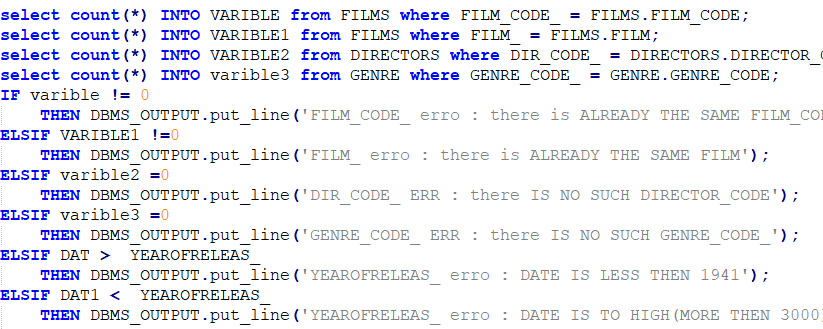


Рисунок 7.3 Обработка ошибок ввода данных

Как можно заметить используются неявные курсоры, важное их свойство, они возвращают только 1 строчку, наша процедура добавляет новый фильм в бд, при этом проверяется имеется ли такой фильм уже в базе данных, имеется ли код фильма в базе данных, имеется ли такой жанр фильма в базе данных, собственной наши курсоры возвращают количество.

Так же использовались явные курсор[2] как видно ниже, на рисунку 7.4

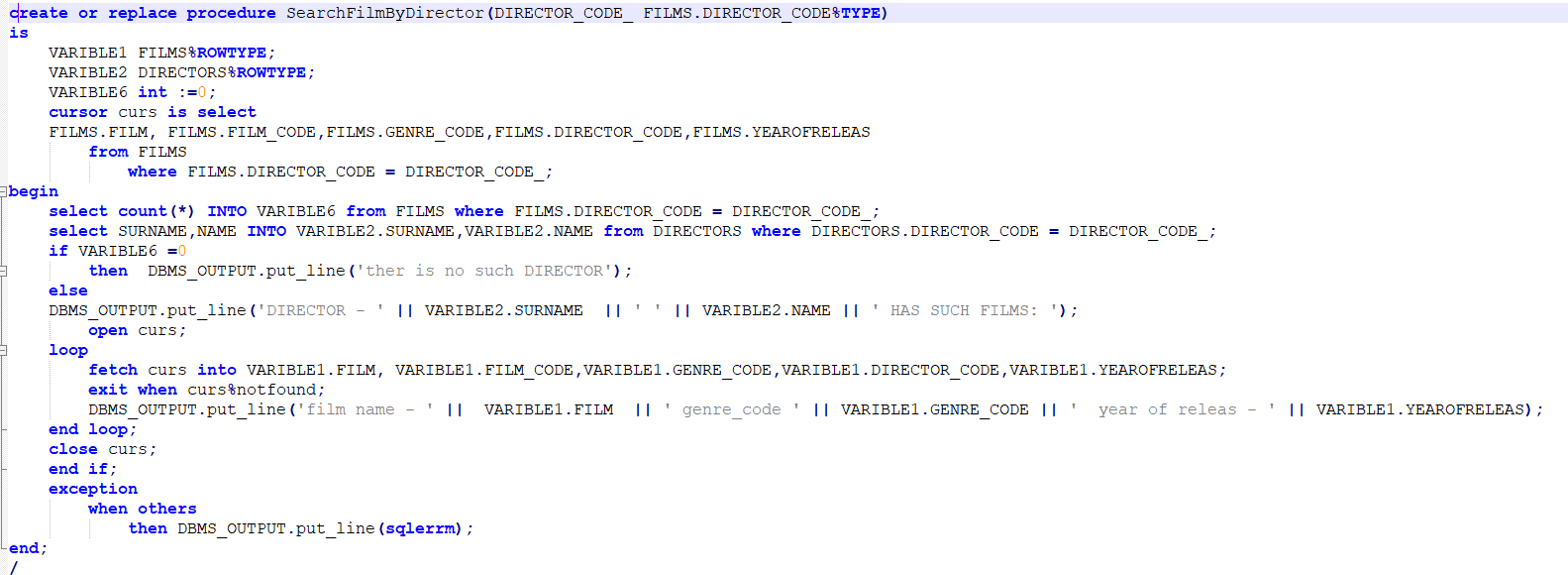


Рисунок 7.4 процедура поиска фильма по режиссеру

Так как у режиссера может быть несколько фильмов, то тогда неявный курсор не поможет надо использовать явный, ибо он возвращает несколько строк.

Все остальные процедуры описаны так же, при неправильном вводе выводится сообщение об ошибке, чтобы пользователь мог понять, что ввел не так.

# **Заключение**

В процессе решения поставленной задачи была достигнута поставленная цель по созданию базы данных «Кинотеатр», которая позволяет управлять кинотеатром. В данной работе использовалось СУБД Oracle DataBase 12c, которая является всемирно известным продуктом, о преимуществах которой можно говорить месяцами. При разработке курсового проекта использовались объекты: таблицы, хранимые процедуры, индексы, последовательности, связи пакеты.

Основной целью курсового проекта стало проектирование базы данных и написание процедур,, которое помогло облегчить взаимодействие с базой данных. Взаимодействие пользователя и базой данных было реализовано с помощью простых анонимных, в которых вызываются процедуры.

* выборка данных для таблиц;
* добавление/удаление режиссеров;
* работа режиссерами их фильмами и сеансами, удаление, создание добавление.
* добавление клиента;
* добавления заказа;
* обновление информации о фильмах;
* добавления фильмов;
* экспорт и импорт таблицы в формат xml.

Приложение прошло тестирование при использовании в БД[1] большого количество данных. Также были реализованы методы для импорта, экспорта данных в формат XML, через анонимные блоки.

Была реализована технология «Резервное копирование и восстановление».

В соответствии с полученным результатом работы программы можно сделать вывод, что разработанная программа работает верно, а требования технического задания выполнены в полном объёме.

# **Cписок источников**

1. Документация Oracle [Электронный ресурс] / Foundation, Inc. https://docs.oracle.com/cd/B28359\_01/server.111/b31222/toc.htm – Дата доступа: 09.04.2021.
2. Официальный сайт Oracle [Электронный ресурс] / Foundation, Inc. https://www.oracle.com/database/database-vault/index.html– Дата доступа: 11.04.2021.
3. Продукты Oracle [Электронный ресурс] [Электронный ресурс] / Foundation, Inc. http://www.interface.ru/home.asp?artId=24678 – Дата доступа: 17.04.2021.