Казанский (Приволжский) федерльный университет  
  
  
  
  
  
  
  
Отчет по учебной практике

(практика по получению первичных профессиональных умений и навыков, в том числе первичных умений и навыков научно-исследовательской деятельности)

Студент

гр.09-822 Шабаков Ильвар Жомортханович

Руководитель

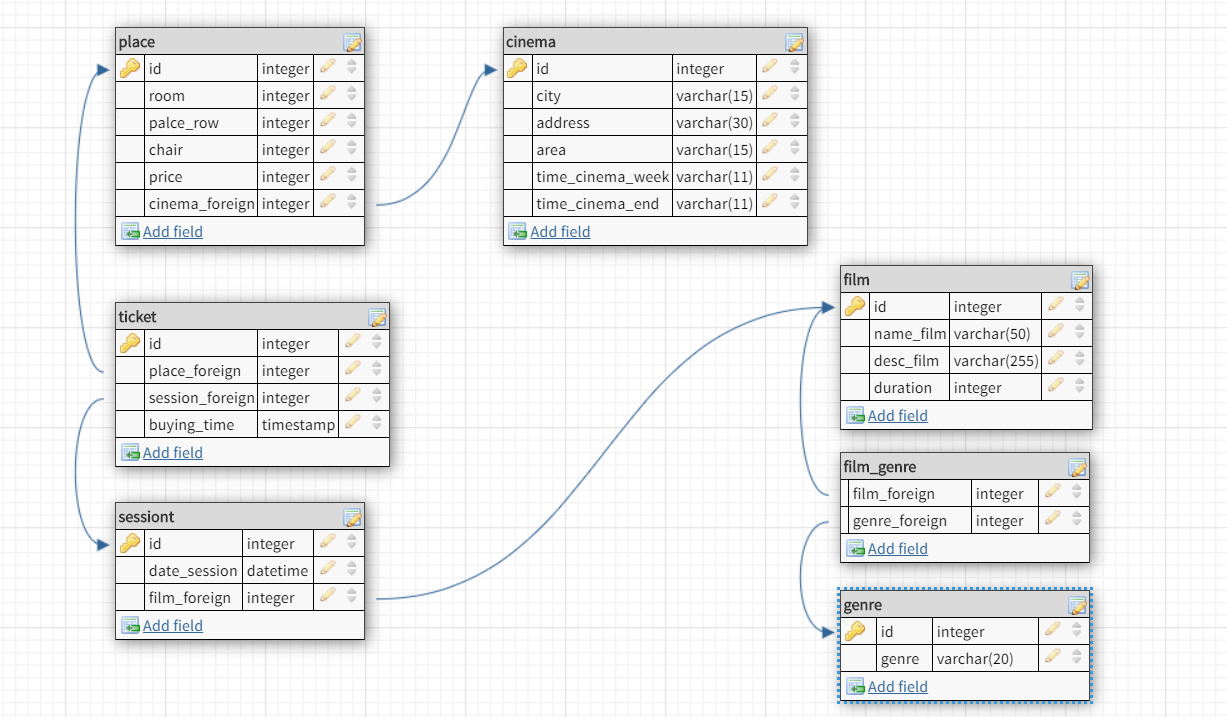
Ст. преп. кафедры ПМ Осипов Евгений Александрович

Казань - 2020

Индивидуальное задание на практику

Задание практики заключалось в работе над индивидуальный проектом: создание базы данных с минимум 5 таблицами и организацией связи между ними, реализация сложных выборок из таблицы, создание хранимых процедур и триггеров.

**Схема БД:**



**Описание таблиц**:

В таблице **cinema(id, city, address, area,time\_cinema\_week, time\_cinema\_end)** хранятся записи кинотеатров: город, адрес, район и время работы

В таблице **place(id,room, place\_row,chair,price,cinema\_foreign)** хранятся залы, места и цены на каждое место(цена меняется в зависимости от выбранного ряда). cinema\_foreign является внешним ключом к таблице cinema(cвязь один ко многим)

В таблице **session(id, date\_session, film\_foreign)** хранятся сеансы фильмов(время сеанса и id фильма). film\_foreign является внешним ключом для таблицы film. Оттуда и берутся фильмы для сеанса

Таблица **ticket(id, place\_foreign, session\_foreign, buying\_time)** является продуктом связи между таблицами place и sessiont, так как множество мест может принадлежать одному сеансу, как и множество сеансов могут принадлежать к одному месту(например, сеанс одного и того же фильма в разное время)

В таблице **film(id, name\_film, desc\_film, duration)** хранятся название фильмов, их описание и длительность в минутах

В таблице **genre(id, genre)** хранятся существующие жанры кино

Таблица **film\_genre(film\_foreign, genre\_foreign)** является продуктом связи между таблицей film и genre(связь многие ко многим, так как у одного фильма может быть множество жанров и наоборот)

**SELECT-запросы:**

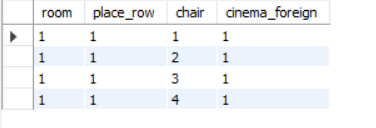
**1)С подзапросом:**

SELECT room,place\_row, chair, cinema\_foreign

FROM place

WHERE id IN(SELECT place\_foreign FROM ticket WHERE session\_foreign=37);

Результат:



Запрос выводит все купленные места в сеансе c id=37

**2)С UNION:**

SELECT chair FROM place

WHERE room=1 AND cinema\_foreign=1

UNION ALL

SELECT chair FROM place

WHERE room=2 AND cinema\_foreign=1

Результат: вывод кресел 1 и 2 зала первого кинотеатра

**3)С JOIN’ом:**

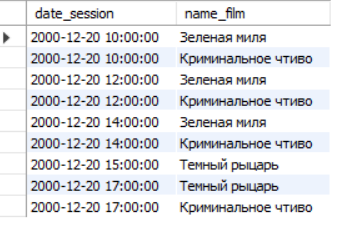
SELECT sessiont.date\_session, film.name\_film

FROM sessiont

INNER JOIN film

ON sessiont.film\_foreign = film.id;

Результат:



Произошло соединение таблиц session и fim, где вместо внешнего ключа таблицы session(session\_foreign) стоят названия фильмов.

**4)Сортировка по текстовому полю:**

SELECT date\_session FROM sessiont

order by date\_session ASC

Результат:



Сортировка времени сеансов по времени

**Процедуры**:

**Надстройка над INSERT INTO**:

**cinemas(city, address, area, time\_cinema\_week, time\_cinema\_end)**:

CREATE DEFINER=`root`@`localhost` PROCEDURE `cinemas`(IN city VARCHAR(15),IN address VARCHAR(30), IN area VARCHAR(15),IN time\_cinema\_week VARCHAR(11),IN time\_cinema\_end VARCHAR(11))

begin

DECLARE EXIT HANDLER FOR SQLEXCEPTION SELECT 'Дубликат записи';

INSERT INTO cinema(city,address,area,time\_cinema\_week,time\_cinema\_end)

VALUES(city,address,area,time\_cinema\_week,time\_cinema\_end);

end

**places(room, r, c, id)**:

CREATE DEFINER=`root`@`localhost` PROCEDURE `places`(IN room INT, in r INT, in c int, in id int)

BEGIN

DECLARE i INT Default 1;

DECLARE j INT Default 1;

DECLARE EXIT HANDLER FOR SQLEXCEPTION SELECT 'Дубликат записи';

WHILE i!=r+1 DO

WHILE j!=c+1 DO

IF i=1 THEN

INSERT INTO place(room,place\_row,chair,init,price,cinema\_foreign)

VALUES(room,i,j,'300',id);

ELSE

INSERT INTO place(room,place\_row,chair,init,price,cinema\_foreign)

VALUES(room,i,j,'200',id);

END IF;

SET j=j+1;

END WHILE;

SET j=1;

SET i=i+1;

END WHILE;

end

**sessions(date\_session, film\_foreign)**:

CREATE DEFINER=`root`@`localhost` PROCEDURE `sessions`(IN date\_session DATETIME,IN film\_foreign VARCHAR(50))

BEGIN

DECLARE f INT Default (SELECT id FROM film WHERE name\_film=film\_foreign);

DECLARE EXIT HANDLER FOR SQLEXCEPTION SELECT 'Дубликат записи/отсутствует фильм';

INSERT INTO sessiont(date\_session, film\_foreign)

VALUES(date\_session, f);

END

**films(name\_film, desc\_film, duration)**:

CREATE DEFINER=`root`@`localhost` PROCEDURE `films`(IN name\_film VARCHAR(50),IN desc\_film VARCHAR(255),IN duration INT)

BEGIN

DECLARE EXIT HANDLER FOR SQLEXCEPTION SELECT 'Дубликат записи';

INSERT INTO film(name\_film, desc\_film, duration)

VALUES(name\_film, desc\_film, duration);

END

**film\_genres(film\_foreign, genre\_foreign):**

CREATE DEFINER=`root`@`localhost` PROCEDURE `film\_genres`(IN film\_foreign VARCHAR(50), IN genre\_foreign VARCHAR(255))

BEGIN

DECLARE f int Default (SELECT id FROM film WHERE name\_film=film\_foreign);

DECLARE g int Default (SELECT id FROM genre WHERE genre=genre\_foreign);

DECLARE EXIT HANDLER FOR SQLEXCEPTION SELECT 'Дубликат записи/отсутствует фильм или жанр';

INSERT INTO film\_genre(film\_foreign, genre\_foreign)

VALUES(f, g);

END

**genres(genre\_create)**:

CREATE DEFINER=`root`@`localhost` PROCEDURE `genres`(IN genre\_create VARCHAR(20))

begin

DECLARE EXIT HANDLER FOR SQLEXCEPTION SELECT 'Дубликат жанра';

INSERT INTO genre(genre)

VALUES(genre\_create);

end

**tickets(room\_place, row\_place1,chair1, cinema1, session\_id)**:

CREATE DEFINER=`root`@`localhost` PROCEDURE `tickets`(IN room\_place INT, IN row\_place1 INT, IN chair1 INT, IN cinema1 INT, IN session\_id INT)

BEGIN

DECLARE id\_chair INT DEFAULT (SELECT id FROM place WHERE room=room\_place AND place\_row=row\_place1 AND chair=chair1 AND cinema\_foreign=cinema1);

DECLARE EXIT HANDLER FOR SQLEXCEPTION SELECT 'Дубликат записи';

INSERT INTO ticket(place\_foreign, session\_foreign)

VALUES(id\_chair,session\_id);

END

**Математические процедуры**:

**session\_sum(session\_id)**:

CREATE DEFINER=`root`@`localhost` PROCEDURE `session\_sum`(IN session\_id INT)

BEGIN

SELECT SUM(price)

FROM place

WHERE id IN(SELECT place\_foreign FROM ticket WHERE session\_foreign=session\_id);

END

Результат: выводит полученную выручку за сеанс

**tickets\_sold(session\_id)**:

CREATE DEFINER=`root`@`localhost` PROCEDURE `tickets\_sold`(IN session\_id INT)

BEGIN

SELECT COUNT(id)

FROM place

WHERE id IN(SELECT place\_foreign FROM ticket WHERE session\_foreign=session\_id);

END

Результат: выводит количество проданных билетов на сеанс

**Триггеры**:

**sessiont\_AFTER\_DELETE**:

CREATE DEFINER=`root`@`localhost` TRIGGER `sessiont\_AFTER\_DELETE` AFTER DELETE ON `sessiont` FOR EACH ROW BEGIN

DECLARE f VARCHAR(50) Default (SELECT name\_film FROM film WHERE id=old.film\_foreign);

DECLARE exist TINYINT(1) Default (SELECT EXISTS(SELECT ticket\_film FROM ticket\_audit WHERE ticket\_film =f));

IF exist=0 THEN

INSERT INTO ticket\_audit(ticket\_film)

VALUES(f);

END IF;

SET SQL\_SAFE\_UPDATES = 0;

UPDATE ticket\_audit

SET session\_number=session\_number+1

WHERE ticket\_film=f;

SET SQL\_SAFE\_UPDATES = 1;

END

Результат: после удаления сеанса записывает в таблицу ticket\_audit данные о количестве проведенных сеансов фильма(если фильма не существует- создает запись, в противном случае просто меняет счетчик)

**sessiont\_BEFORE\_DELETE**:

CREATE DEFINER=`root`@`localhost` TRIGGER `sessiont\_BEFORE\_DELETE` BEFORE DELETE ON `sessiont` FOR EACH ROW BEGIN

DECLARE ticket\_counter INT Default (SELECT COUNT(id) FROM ticket WHERE session\_foreign=old.id);

INSERT INTO session\_audit(session\_id,tickets\_sold)

VALUES(old.id, ticket\_counter);

END

Результат: перед удалением записи записывает в таблицу session\_audit количество проданных билетов на сеанс

**План выполнения запросов:**

1)Создаем кинотеатр в таблице cinema

2)Генерируем залы с местами для данного кинотеатра в таблице place(лучше использовать процедуру places для быстрой генерации нужного количества рядов и мест в нем)

3)Заполняем таблицу film фильмами

4)Создаем сеанс фильма в sessiont, указывая время и ссылаясь на нужный фильм из таблицы film

5)Продаем билеты через таблицу ticket, ссылаясь на сеанс из таблицы session и место из таблицы place)