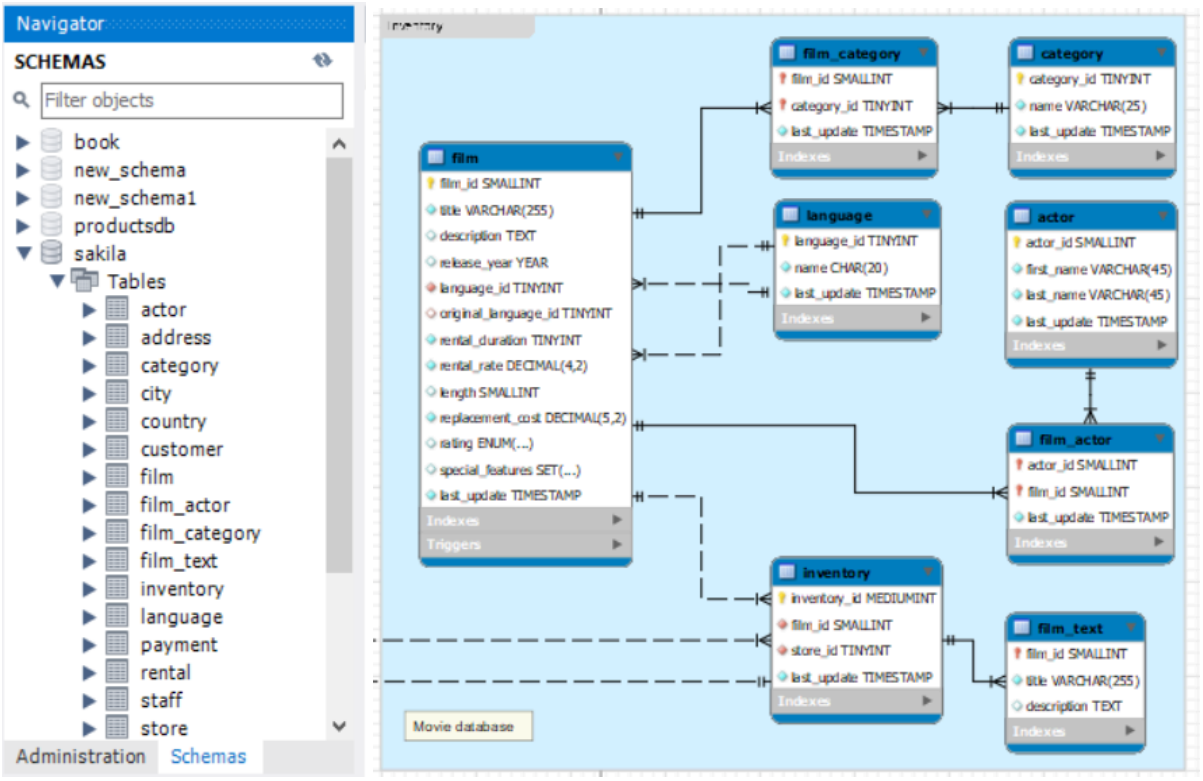
Лабораторная работа № 14

Создание, редактирование и удаление представлений.

Цель: Сформировать умения создавать, редактировать и удалять представления.

3. Задание

Для создания представлений будет использована учебная БД sakila. При полной установке MySql ее можно найти в навигаторе БД MySQL Workbench. При ее отсутствии необходимо скачать учебную БД с официального сайта.



1. Создать представления на основе следующих запросов:

1.1 Вывести названия языков из таблицы **language** в алфавитном порядке.

1.2. Вернуть полные имена (имя и фамилия) актеров с «SON» в их фамилии, упорядоченные по их именам.

1.3. Найдите все адреса, где второй адрес не пустой (т.е. содержит некоторый текст), упорядочить данные по полю district.

1.4. Вывести имена и фамилии актеров, сыгравших в фильмах «BERETS AGENT» и «ANGELS LIFE», вместе с годом выпуска фильма, отсортированным по фамилиям актеров.

1.5. Найдите все категории фильмов, в которых есть от 55 до 65 фильмов. Выведите название этих категорий и количество фильмов в каждой категории, отсортированное по количеству фильмов.

1.6. Найдите имена (имя и фамилию) всех актеров, чье имя совпадает с именем имя актера с ID 8. Не возвращайте самого актера с ID 8. Обратите

внимание, что вы не можете использовать имя актера с ID 8 в качестве константы (только ID).

2. Исследовать команды редактирования и удаления представлений.

3. Произвести выборку из созданных представлений.

3. Определить какие из созданных представлений можно отнести к модифицируемым представлениям, а какие можно использовать только для чтения. Сделать пояснения после каждого представления.

Теоретические сведения

Рассмотрим следующие вопросы:

- создание представлений с помощью оператора *CREATE VIEW*;
- удаление представлений с помощью оператора *DROP VIEW*.

Основная структурная единица реляционных БД – таблицы, но язык SQL предоставляет еще один способ организации данных. Представление – это запрос на выборку, которому присваивается уникальное имя и который можно сохранять или удалять из БД как хранимую процедуру. Представления позволяют увидеть результаты сохраненного запроса так, как будто это полноценная таблица. MySQL, встретив в запросе ссылку на представление, ищет его определение в БД. Пользовательский запрос с участием представления преобразуется в эквивалентный запрос к исходным таблицам. Если определение представления простое, то каждая строка представления формируется «на лету». Если определение сложное, MySQL материализует представление в виде временной таблицы. Клиент, обращаясь к представлению, будет видеть только столбцы результирующей таблицы. Не имеет значения, сколько столбцов в исходной таблице и является ли запрос, лежащий в основе представления, одно- или многотабличным. Клиенту можно запретить обращаться к исходным таблицам, но снабдить привилегиями обращения к представлениям. На одном наборе таблиц можно создать гибкие системы доступа.

Преимущества представлений:

- *безопасность* – каждый пользователь имеет доступ к небольшому числу представлений, содержащих только ту информацию, которую ему позволено знать;
- *простота запросов* – с помощью представления можно извлечь данные из нескольких таблиц и представить их как одну таблицу (запрос ко многим таблицам заменяется однотабличным запросом к представлению);
- *простота структуры* – представления позволяют создать для каждого пользователя собственную структуру БД (отображаются данные, которые ему нужны);

- *защита от изменений* – таблицы и их структура могут постоянно изменяться и переименовываться; представления позволяют создавать виртуальные таблицы со старыми именами и структурой, позволяя избежать модификации приложений.

Недостатки представлений:

- *производительность* – представления создают видимость существования таблицы, и MySQL приходится преобразовывать запрос к представлению в запрос к исходным таблицам; если представление отображает многотабличный запрос, то простой запрос к представлению становится сложным объединением;
- *ограничение на обновление* – когда пользователь пытается обновить строки представления, MySQL необходимо обновить строки в исходных таблицах; это возможно только для простых представлений, сложные представления доступны только для выборки.

Поэтому не стоит везде применять представления вместо исходных таблиц.

Создание представлений. Осуществляется при помощи оператора

CRFATE [OR REPLACE] [ALGORITHM = {UNDEFINED / MERGE / TEMPTABLE}] VIEW view_name [(column_list)] AS select_statement
[WITH [CASCADED / LOCAL] CHECK OPTION];

Оператор создает представление *view_name* со столбцами, перечисленными в *column_list*, на основании запроса *select_statement*. Рассмотрим создание представления *cat*, которое дублирует таблицу *catalogs* базы данных *book*:

```
mysql> CREATE VIEW cat AS  SELECT  *  FROM catalogs;
Query OK, 0 rows affected (0.02 sec)

mysql> SELECT  *  FROM cat;
+-------+------------------+
| catID | cat_name         |
+-------+------------------+
|     1 | Программирование |
|     2 | Интернет         |
|     3 | Базы данных      |
|     4 | Сети             |
|     5 | Мультимедиа      |
+-------+------------------+
5 rows in set (0.00 sec)
```

Представление рассматривается как полноценная таблица и может быть просмотрено в списке таблиц БД при помощи оператора *SHOW TABLES*:

```
mysql> SHOW TABLES;
+----------------+
| Tables_in_book |
+----------------+
| books          |
| cat            |
| catalogs       |
| orders         |
| users          |
+----------------+
5 rows in set (0.00 sec)
```

При создании представления можно явно указать список столбцов, изменить их названия и порядок следования, например:

```
mysql> CREATE OR REPLACE VIEW cat (catalog, id)
    -> AS SELECT cat_name, catID FROM catalogs;
Query OK, 0 rows affected (0.02 sec)

mysql> SELECT * FROM cat;
+------------------+----+
| catalog          | id |
+------------------+----+
| Программирование |  1 |
| Интернет         |  2 |
| Базы данных      |  3 |
| Сети             |  4 |
| Мультимедиа      |  5 |
+------------------+----+
5 rows in set (0.00 sec)
```

Представления, не содержащие дополнительных столбцов, называются *вертикальными представлениями.* Они применяются для ограничения доступа пользователей к столбцам таблицы. Пример вертикального представления см. ниже.

Кроме вертикальных представлений используются *горизонтальные представления,* которые делают видимыми только те строки, с которыми работают пользователи. Например, чтобы в электронном магазине каждый менеджер видел только те товарные позиции, за которые отвечает, можно создать представления для менеджеров. Учетные записи менеджеров следует лишить привилегии доступа к таблице и разрешить просматривать только свои представления.

Создадим представление *manager1* для менеджера, работающего с каталогами «Интернет» и «Сети»:

```
mysql> CREATE VIEW manager1
    -> AS SELECT * FROM books
    -> WHERE b_catID IN (SELECT catID
    -> FROM catalogs
    -> WHERE cat_name = 'Интернет' OR cat_name = 'Сети')
    -> ORDER BY b_name;
Query OK, 0 rows affected (0.01 sec)
```

```
mysql> SELECT b_name, b_price, b_count FROM manager1;
+-----------------------------------------+---------+---------+
| b_name                                  | b_price | b_count |
+-----------------------------------------+---------+---------+
| Web-конструирование                     |  177.00 |       6 |
| Анализ и диагностика компьютерных сетей |  344.00 |       3 |
| Безопасность сетей                      |  462.00 |       5 |
| Компьютерные сети                       |  630.00 |       6 |
| Общение в Интернете                     |   85.00 |       5 |
| Принципы маршрутизации в Internet       |  428.00 |       4 |
| Поиск в Internet                        |  107.00 |       2 |
| Популярные интернет-браузеры            |   82.00 |       6 |
| Локальные вычислительные сети           |   82.00 |       8 |
| Сети. Поиск неисправностей              |  434.00 |       4 |
| Самоучитель Интернет                    |  121.00 |       4 |
+-----------------------------------------+---------+---------+
11 rows in set (0.05 sec)
```

Наиболее удобно использовать представления для формирования сгруппированных таблиц. При работе с такими таблицами MySQL самостоятельно формирует временную таблицу – см. пункт «Пример выполнения работы» (пример 2).

Удаление представлений. Выполняется с помощью оператора:
DROP VIEW [IF EXISTS] view_name [, view_name] … ;

Оператор позволяет уничтожить одно или несколько представлений, например:

```
mysql> DROP VIEW cat, list_user, price;
Query OK, 0 rows affected (0.03 sec)

mysql> SHOW TABLES;
+---------------+
| Tables_in_book |
+---------------+
| books          |
| catalogs       |
| orders         |
| users          |
+---------------+
4 rows in set (0.00 sec)
```

Практическая работа

Пример выполнения работы

1. Создадим вертикальное представление *list_user*, которое будет отображать фамилию и инициалы покупателей, скрывая другие поля.

```
mysql> CREATE OR REPLACE VIEW list_user
    -> AS SELECT CONCAT(u_surname, " ",
    -> SUBSTRING(u_name,1,1), ".",
    -> SUBSTRING(u_patronymic,1,1), ".") AS name
    -> FROM users ORDER BY name;
Query OK, 0 rows affected (0.08 sec)

mysql> SELECT * FROM list_user;
+---------------+
| name          |
+---------------+
| Кузнецов М.П. |
| Корнеев А.А.  |
| Петров А.Ю.   |
| Иванов А.В.   |
| Лосев С.И.    |
| Симонов И.Н.  |
+---------------+
6 rows in set (0.02 sec)
```

2. Создадим представление *price* с общей стоимостью книг в каждом каталоге.

```
mysql> CREATE VIEW price
    -> AS SELECT b_catID, SUM(b_price*b_count) AS price
    -> FROM books
    -> GROUP BY b_catID
    -> ORDER BY price;
Query OK, 0 rows affected (0.08 sec)

mysql> SELECT * FROM price;
+---------+----------+
| b_catID | price    |
+---------+----------+
|       3 |  2908.00 |
|       2 |  4389.00 |
|       4 |  9514.00 |
|       1 | 12123.00 |
|       5 | 15177.00 |
+---------+----------+
5 rows in set (0.01 sec)
```

Сформируем запрос к таблице *catalogs* и представлению *price*.

```
mysql> SELECT catalogs.cat_name, price.price
    -> FROM price, catalogs
    -> WHERE price.b_catID = catalogs.catID
    -> ORDER BY catalogs.catID;
+------------------+----------+
| cat_name         | price    |
+------------------+----------+
| Программирование | 12123.00 |
| Интернет         |  4389.00 |
| Базы данных      |  2908.00 |
| Сети             |  9514.00 |
| Мультимедиа      | 15177.00 |
+------------------+----------+
5 rows in set (0.02 sec)
```

Сформируем запрос к представлению *price* (получение минимального и максимального значений стоимости книг в каталогах и общей стоимости всех книг).

```
mysql> SELECT MIN(price), MAX(price), SUM(price) FROM price;
+------------+------------+------------+
| MIN(price) | MAX(price) | SUM(price) |
+------------+------------+------------+
|    2908.00 |   15177.00 |   44111.00 |
+------------+------------+------------+
1 row in set (0.02 sec)
```

6.Форма отчета о работе

Номер учебной группы _____
Фамилия, инициалы обучающегося _____
Дата выполнения работы _____
Тема работы: _____
Цель работы: _____
Задание: _____
Оснащение работы: _____
Результат выполнения работы: _____

7. Контрольные вопросы и задания

1. Приведите синтаксис команд для создания, редактирования, удаления представлений.

2. Дайте понятие о вертикальном, горизонтальном, смешанном представлении.

3. Опишите преимущества и недостатки представлений.

4. Опишите условия при которых представление может быть модифицируемым.

Рекомендуемая литература

1. Дейт, К.Дж. Введение в системы баз данных / К.Дж. Дейт. М. : Вильямс, 2018. 1382 с.
2. Лазицкас, Е.А. Базы данных и системы управления базами данных / Е.А. Лазицкас, И.Н. Загуменникова, П.Г. Гилевский. Минск : РИПО, 2016. 268 с.
3. Федорова, Г. Разработка и администрирование баз данных / Г. Федорова. М. : Академия, 2015. 313 с.
4. SQL справочник / К. Кляйн [и др.]. СПб. : Символ-плюс, 2016. 56 с.
5. Артеменко, Ю.Н. MySQL. Справочник по языку / Ю.Н. Артеменко. М. : Диалектика, 2005. 429 с.
6. Базы данных : учеб. / А.Д. Хомоненко [и др.]. СПб. : КОРОНА-ВЕК, 2010. 736 с.
7. Дюбуа, П. MySQL / П. Дюбуа. М. : Вильямс, 2007. 1168 с.