1-3

**Создание DB**

CREATE DATABASE IF NOT EXISTS library

**Таблица авторов**

CREATE TABLE authors (

id INT(11) NOT NULL,

name VARCHAR(255) NOT NULL,

country VARCHAR(255) NULL DEFAULT NULL,

PRIMARY KEY (id)

);

INSERT INTO authors (id, name, birthdate, country)

VALUES

(NULL, ‘Patricia Aakhus’, ‘1952-05-17’, ‘USA’),

(NULL, ‘Peter Hoffman’, ‘1944-03-21’, ‘Germany’),

(NULL, ‘…’, ’…’);

**Таблица Автор – Книга**

CREATE TABLE book\_authors\_1 (

id INT(11) NOT NULL AUTO\_INCREMENT,

book\_id INT(11) NOT NULL,

author\_id INT(11) NOT NULL,

PRIMARY KEY (id)

);

**Внешние ключи для таблицы**

ALTER TABLE book\_authors\_1

ADD INDEX book\_author\_fk\_1\_idx (author\_id ASC),

ADD INDEX book\_author\_fk\_2\_idx (book\_id ASC);

;

ALTER TABLE book\_authors\_1

ADD CONSTRAINT book\_author\_fk\_1

FOREIGN KEY (author\_id)

REFERENCES authors (id)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

ADD CONSTRAINT book\_author\_fk\_2

FOREIGN KEY (book\_id)

REFERENCES books (id)

ON DELETE NO ACTION

ON UPDATE NO ACTION;

ALTER TABLE book\_authors\_1

ADD CONSTRAINT book\_authors\_fk\_\_1

FOREIGN KEY (author\_id)

REFERENCES authors\_1 (id)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

ADD CONSTRAINT book\_authors\_fk\_\_2

FOREIGN KEY (book\_id)

REFERENCES books\_1 (id)

ON DELETE NO ACTION

ON UPDATE NO ACTION;

**Таблица книги**

[CREATE](http://127.0.0.1/openserver/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/5.5/en/create-table.html) [TABLE](http://127.0.0.1/openserver/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/5.5/en/create-table.html) library\_books (

id [INT](http://127.0.0.1/openserver/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/5.5/en/numeric-types.html)(11) [NOT](http://127.0.0.1/openserver/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/5.5/en/logical-operators.html#operator_not) NULL AUTO\_INCREMENT,

title [VARCHAR](http://127.0.0.1/openserver/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/5.5/en/string-types.html)(255) [NOT](http://127.0.0.1/openserver/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/5.5/en/logical-operators.html#operator_not) NULL ,

description [TEXT](http://127.0.0.1/openserver/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/5.5/en/string-types.html) NULL [DEFAULT](http://127.0.0.1/openserver/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/5.5/en/miscellaneous-functions.html#function_default) NULL ,

ISBN [VARCHAR](http://127.0.0.1/openserver/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/5.5/en/string-types.html)(64) NULL [DEFAULT](http://127.0.0.1/openserver/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/5.5/en/miscellaneous-functions.html#function_default) NULL ,

genre [INT](http://127.0.0.1/openserver/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/5.5/en/numeric-types.html)(11) NULL [DEFAULT](http://127.0.0.1/openserver/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/5.5/en/miscellaneous-functions.html#function_default) NULL ,

published [DATE](http://127.0.0.1/openserver/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/5.5/en/date-and-time-types.html) NULL [DEFAULT](http://127.0.0.1/openserver/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/5.5/en/miscellaneous-functions.html#function_default) NULL ,

PRIMARY KEY (id))

;

INSERT INTO books (id, title, description, ISBN, genre, published)

VALUES

(NULL, "The Voyage of Mael Duins Curragh" , "The Voyage of Mael Duins Curragh is a 1990 novel written by Patricia Aakhus. The novel was Aakhus first published book, and retells the ancient Irish legend of Mael Duin, an adopted son of a chieftains widow who accidentally learns of his true parents. The novel retrieved significant acclaim upon its release, including a national review by the New York Times", '978-1-10-769989-2', 1, '1990-01-28')

INSERT INTO books (id, title, description, ISBN, genre\_id, published)

VALUES

(NULL, "Преступление и наказание", "Преступление и наказание» — социально-психологический и социально-философский роман Фёдора Михайловича Достоевского, над которым писатель работал в 1865—1866 годах.", "9122204857897", 2, "1866-01-22"),

(NULL, "Мастер и Маргарита", "«Ма́стер и Маргари́та» — роман Михаила Афанасьевича Булгакова, работа над которым началась в конце 1920-х годов и продолжалась вплоть до смерти писателя. Роман относится к незавершённым произведениям; редактирование и сведение воедино черновых записей осуществляла после смерти мужа вдова писателя", "9783598215421", 3, "1967-10-16"),

(NULL, "Дама с собачкой", "«Да́ма с соба́чкой» — рассказ русского писателя и драматурга Антона Павловича Чехова, написанный в 1898 году. Впервые опубликован в журнале «Русская мысль» № 12 в 1899 году.", "978-3-598-21541-4", 2, "1899-07-20"),

(NULL, "Руслан и Людмила", "«Русла́н и Людми́ла» — первая законченная поэма Александра Сергеевича Пушкина; волшебная сказка, вдохновлённая древнерусскими былинами.", "978-3-598-21542-1", 4, "1820-02-05"),

(NULL, "Мёртвые души", "«Мёртвые ду́ши» — произведение Николая Васильевича Гоголя, жанр которого сам автор обозначил как поэму. Изначально задумано как трёхтомник. Первый том был издан в 1842 году. Практически готовый второй том был утерян, но сохранилось несколько глав в черновиках", "978-3-598-21543-8", 4, "1842-11-18"),

(NULL, "Память, говори", "«Память, говори» — автобиографическая книга Владимира Набокова, вышедшая на английском языке, собрание систематически связанных личных воспоминаний, охватывающее 37 лет.", "978-3-598-21596-4", 5, "1951-02-01"),

(NULL, "Алгоритмы: построение и анализ", "Алгори́тмы: построе́ние и ана́лиз — книга по алгоритмам и структурам данных, написанная Томасом Корменом, Чарльзом Лейзерсоном, Рональдом Ривестом и Клиффордом Штайном. Книга используется во многих учебных заведениях в качестве пособия и справочника, на книгу имеется более 4000 ссылок на CiteSeerX.", "944-3-598-21598-8", 2, "1989-03-30"),

(NULL, "Introduction to Algorithms, Second Edition, German translation", "The latest edition of the essential text and professional reference, with substantial new material on such topics as vEB trees, multithreaded algorithms, dynamic programming, and edge-based flow.", "978-3-888-21599-5", 3, "2004-01-02"),

(NULL, "Книга Алгоритмы. Построение и анализ", "Книга Алгоритмы. Построение и анализ удачно объединяет в себе полноту охвата и строгость изложения материала.", "978-3-598-21562-9", 5, "2015-09-10");

**Таблица жанры**

CREATE TABLE genres\_1 (

id INT(11) NOT NULL AUTO\_INCREMENT,

genre VARCHAR(255) NOT NULL,

PRIMARY KEY (id));

INSERT INTO genres\_1 (id, genre)

VALUES

(NULL, "mathematics"),

(NULL, "economics"),

(NULL, "history"),

(NULL, "science"),

(NULL, "psychology"),

(NULL, "fiction"),

(NULL, "novel"),

(NULL, "nonfiction"),

(NULL, "poems"),

(NULL, "comic");

**Таблица читатели**

CREATE TABLE readers\_1 (

id INT(11) NOT NULL AUTO\_INCREMENT,

reader\_name VARCHAR(255) NOT NULL,

phone VARCHAR(255) NULL DEFAULT NULL,

city VARCHAR(255) NULL DEFAULT NULL,

street VARCHAR(255) NULL DEFAULT NULL,

genre\_of\_interest INT(11) NULL DEFAULT NULL,

country VARCHAR(255) NULL,

PRIMARY KEY (id)

);

Перенос одних данных в другую

INSERT INTO readers\_1 select \* from readers

Таблица Информация по книге

CREATE TABLE book\_info\_1 (

id INT(11) NOT NULL AUTO\_INCREMENT,

book\_id INT(11) NULL,

date\_of\_issue DATE NULL DEFAULT NULL,

date\_of\_return DATE NULL DEFAULT NULL,

expected\_date\_of\_return DATE NULL DEFAULT NULL,

reader\_id INT(11) NULL DEFAULT NULL,

PRIMARY KEY (id)

);

**4. Создать запрос - количество книг в библиотеке**

SELECT COUNT(id) AS BookCount

FROM books;

**5. Создать запрос - выбрать все книги, которые в данный момент ни у кого не на руках**

SELECT title FROM books

LEFT JOIN book\_info ON books.id = book\_info.book\_id

WHERE book\_info.date\_of\_return IS NULL AND book\_info.expected\_date\_of\_return IS NULL

**6. Создать запрос - выбрать все книги, которые взяли почитать минимум четыре раза**

SELECT books.title

FROM books

LEFT JOIN book\_info ON books.id = book\_info.book\_id

GROUP BY book\_info.book\_id

HAVING COUNT(book\_info.id) >= 4;

**7. Создать запрос - выбрать все книги, которые не вернули с указанием их жанра, писателя и всей информацией (через JOIN)**

SELECT books.title, books.description, books.ISBN, genres.genre, books.published

FROM books

JOIN book\_info ON books.id = book\_info.book\_id

JOIN genres ON books.genre\_id = genres.id

WHERE book\_info.date\_of\_return IS NULL

**8. Создать запрос - выбрать всех авторов, которые написали минимум две книги в соавторстве с другими авторами**

SELECT name

FROM authors

JOIN book\_author ON authors.id = book\_author.author\_id

GROUP BY author\_id

HAVING COUNT(book\_id) >= 2

**9. Создать запрос - выбрать топ-10 читателей, которые интересуются жанрами фантастика, кулинария и взяли больше всех книг в этих жанрах за последний год**

SELECT readers.reader\_name, COUNT(books.id) AS Count\_of\_books

FROM readers

JOIN genres ON readers.genre\_of\_interest = genres.id

JOIN book\_info ON book\_info.reader\_id = readers.id

JOIN books ON book\_info.book\_id = books.id AND genres.id = books.genre\_id

WHERE genres.genre IN ('novel', 'comic') AND book\_info.date\_of\_issue <= '2019-05-10'

GROUP BY readers.id

ORDER BY Count\_of\_books

LIMIT 10

**10. Создать запрос - удалить историю посещения библиотеки одного конкретного пользователя.**

DELETE

FROM library.book\_info

WHERE reader\_id = 7;

**11. В библиотеке проблема - в каталоге случайно перед названиями всех книг поставились пробелы. Воссоздай проблему (добавить два пробела в название всех книг через UPDATE) а потом почини её соответствующим образом**

UPDATE books

SET title = concat(' ', title);

UPDATE books

SET title = TRIM(title);

**12. Создать запрос - посчитать количество читателей в библиотеке за последние 8 месяцев от текущего. Например, май 2020 - 100 читателей, апрель 2020 - 89 читателей и так далее. Позже, расширить группировку до год-месяц-жанр (май 2020 ужасы - 90 читателей)**

**SELECT DATE\_FORMAT(date\_of\_issue, '%m-%Y') as month, COUNT(date\_of\_issue) as count\_in\_month**

**FROM book\_info**

**WHERE date\_of\_issue > DATE\_SUB(NOW(), INTERVAL 8 MONTH)**

**GROUP BY YEAR(date\_of\_issue), MONTH(date\_of\_issue)**

**SELECT DATE\_FORMAT(date\_of\_issue, '%m-%Y') as month, genres.genre, COUNT(date\_of\_issue) as count\_in\_month**

**FROM book\_info**

**LEFT JOIN books ON book\_info.book\_id = books.id**

**LEFT JOIN genres ON books.genre\_id = genres.id**

**WHERE date\_of\_issue > DATE\_SUB(NOW(), INTERVAL 8 MONTH)**

**GROUP BY YEAR(date\_of\_issue), MONTH(date\_of\_issue), genres.genre**

**13. Создать запрос-представление (SQL View) в котором будет топ-5 самых злостных невозвратчиков книг (те, кто имеет максимум опозданий по возращению или вообще невозвращений книг)**

CREATE VIEW badreader AS

SELECT DISTINCT(readers.id), readers.reader\_name

FROM readers

LEFT JOIN book\_info ON readers.id = book\_info.reader\_id

WHERE book\_info.expected\_date\_of\_return < book\_info.date\_of\_return

ORDER BY (UNIX\_TIMESTAMP(book\_info.date\_of\_return) - UNIX\_TIMESTAMP(book\_info.expected\_date\_of\_return)) DESC

LIMIT 5

**14. Создать запрос-транзакцию (атомарная последовательность запросов) - переместить все книги четного года выпуска жанра кулинария в жанр "кулинария четная" (в транзакции само собой будет запрос на создание жанра), потом осознать, что мы маемся дурью и вернуть все обратно с удалением ненужного жанра.**

Start transaction;

INSERT INTO genres (genre)

VALUES ('science\_even');

UPDATE books

LEFT JOIN genres ON genres.id = books.genre\_id

SET genre\_id = (SELECT id FROM genres WHERE genre = 'science\_even' LIMIT 1)

WHERE (YEAR(published)) % 2 = 0 AND genres.genre = 'science';

UPDATE books

LEFT JOIN genres ON genres.id = books.genre\_id

SET genre\_id = (SELECT id FROM genres WHERE genre = 'science' LIMIT 1)

WHERE genres.genre = 'science\_even';

DELETE FROM genres

WHERE genre = 'science\_even';

commit;

**15. Создать запрос, который сохранит данные выборки сразу же в отдельную таблицу - топ 10 книг у которых комбинация автор-название книги имеет максимальную длину.**

CREATE TABLE new\_table AS (SELECT books.title as book\_title, authors.name as author\_name

FROM authors

LEFT JOIN book\_author ON authors.id = book\_author.author\_id

LEFT JOIN books ON books.id = book\_author.book\_id

LEFT JOIN book\_info ON books.id = book\_info.book\_id

GROUP BY (char\_length(books.title) + char\_length(authors.name)) DESC

LIMIT 10

)

SELECT books.title as book\_title, authors.name as author\_name

FROM authors

LEFT JOIN book\_author ON authors.id = book\_author.author\_id

LEFT JOIN books ON books.id = book\_author.book\_id

LEFT JOIN book\_info ON books.id = book\_info.book\_id

GROUP BY (char\_length(books.title) + char\_length(authors.name)) DESC

LIMIT 10

**16. Создать декартово произведение таблиц авторы-жанры (попарное пересечение всех записей одной таблицы со всеми остальными) и вывести количество книг в каждой комбинации автор-жанр**

SELECT authors.name, genres.genre,

(SELECT COUNT(books.id) FROM books

WHERE books.genre\_id = genres.id

AND authors.id =

(SELECT DISTINCT(author\_id)

FROM book\_author

WHERE authors.id = book\_author.author\_id)) as count\_of\_books

FROM authors, genres

**17. Выбрать топ-10 авторов неудачников, которые выпустили 10 книг минимум, но которых не разу не брали читать за последний год. Результат сохранить во временную таблицу и тут же сделать из неё выборку**

SELECT authors.name, (SELECT COUNT(id) FROM books

WHERE book\_author.book\_id = books.id) as books\_count

FROM authors

LEFT JOIN book\_author ON authors.id = book\_author.author\_id

LEFT JOIN books ON books.id = book\_author.book\_id

WHERE (SELECT COUNT(book\_id) FROM book\_info

WHERE date\_of\_return >= NOW() - INTERVAL 1 YEAR AND books.id = book\_id) = 0

AND (SELECT COUNT(id) FROM books WHERE book\_author.book\_id = books.id) >= 1

GROUP BY authors.id

ORDER BY books\_count DESC

**18. Циклом добавить в таблицу авторов 1 миллионов авторов с именем Noname\_XXX, где XXX - переменная цикла, остальные данные заполнять рандомно, каждому из авторов добавить от 1 до 3 (рандомно) книг с такими же рандомными названиями**

DELIMITER $$

CREATE PROCEDURE LoopDemo()

BEGIN

DECLARE x INT;

SET x = 100;

loop\_label: LOOP

IF x > 1000000 THEN

LEAVE loop\_label;

END IF;

SET x = x + 1;

INSERT INTO authors (name) values (CONCAT('Noname', '\_', x));

END LOOP;

END$$

DELIMITER ;

**DELIMITER $$**

**CREATE PROCEDURE LoopAuthorBook()**

**BEGIN**

**DECLARE x INT;**

**DECLARE books\_count INT;**

**DECLARE book\_i INT;**

**DECLARE new\_author\_id INT;**

**DECLARE new\_book\_id INT;**

**SET x = 100;**

**loop\_label: LOOP**

**IF x > 1000000 THEN**

**LEAVE loop\_label;**

**END IF;**

**SET x = x + 1;**

**SET books\_count = FLOOR(1 + RAND()\*3);**

**SET book\_i = 1;**

**INSERT INTO authors (name) values (CONCAT('Noname', '\_', x));**

**SET new\_author\_id = last\_insert\_id();**

**loop\_label\_books: LOOP**

**IF book\_i > books\_count THEN**

**LEAVE loop\_label\_books;**

**END IF;**

**INSERT INTO books (title) values (CONCAT('Noname\_book', '\_', x));**

**SET new\_book\_id = last\_insert\_id();**

**INSERT INTO book\_author (author\_id, book\_id) VALUES (new\_author\_id, book\_id);**

**SET book\_i = book\_i + 1;**

**END LOOP;**

**END LOOP;**

**END$$**

**DELIMITER ;**

**19. Выбрать из нашей уже мусорной базы всех мусорных авторов, добавленных на прошлом шаге и все их книги, отсортировать по имени книг, при этом обратить внимание, как работают индексы с помощью EXPLAIN, ANALYZE и так далее. Затем создать индекс для ускорения выборки и форсировать его использование.**

SELECT authors.name, books.title

FROM authors

LEFT JOIN book\_author ON authors.id = book\_author.author\_id

LEFT JOIN books ON books.id = book\_author.book\_id

WHERE authors.name LIKE 'Noname%'

ORDER BY books.title

**20. С помощью команды MySQL сделать выгрузку таблицы авторов в CSV файл, создать вторую таблицу аналогичной структуры и импортировать в неё SQL командой то, что выгрузили ранее. Результат - таблица авторов и её копия идентичны**

SELECT \* INTO OUTFILE ' c:\ospanel\userdata\temptest.csv'

FIELDS TERMINATED BY ';' ENCLOSED BY '"' ESCAPED BY '\\'

LINES STARTING BY '' TERMINATED BY '\r\n'

FROM authors\_1

LOAD DATA **LOCAL** INFILE “c:\ospanel\userdata\temp\test.csv”

INTO TABLE authors\_2

FIELDS TERMINATED BY ‘;’

ENCLOSED BY ‘\\’

LINES TERMINATED BY ‘\r\n’

IGNORE 1 ROWS