Модул 6. Бази данни

Тема 1. Въведение в бази данни

Задача 1.1. Изтегляне и инсталиране на MySQL Server и Workbench

Изтеглете и инсталирайте **MySQL Server** и **Workbench**.
http://dev.mysql.com/downloads/mysql/

Задача 1.2. Създаване на база данни minions

Създайте нова база данни наречена **minions**. Добавете таблица **minions** (id, name, age). След това добавете нова таблица **towns** (id, name). Сложете колоната **id** на двете таблици да бъде първичен ключ като ограничение. Добавете колона **townid** в таблица **minions** и го направете външен ключ към колона **id** от таблица **towns**. Добавете данни в таблиците.

Решение

```
CREATE SCHEMA IF NOT EXISTS `minions`;
USE `minions`;

CREATE TABLE IF NOT EXISTS `minions`
(
  `id` INT NOT NULL,
  `name` VARCHAR(50) NOT NULL,
  `age` INT(3) NULL,
  PRIMARY KEY (`id`)
);

INSERT INTO `minions` (`id`, `name`, `age`) VALUES ('1', 'Kevin', '15');
INSERT INTO `minions` (`id`, `name`, `age`) VALUES ('2', 'Bob', '22');
INSERT INTO `minions` (`id`, `name`) VALUES ('3', 'Steward');

CREATE TABLE IF NOT EXISTS `towns`
(
   `id` INT NOT NULL,
   `name` VARCHAR(40) NOT NULL,
   PRIMARY KEY (`id`)
);

INSERT INTO `towns` (`id`,`name`) VALUES (1,"Burgas");
INSERT INTO `towns` (`id`,`name`) VALUES (2,"Sofia");
INSERT INTO `towns` (`id`,`name`) VALUES (3,"Varna");

ALTER TABLE `minions`
```

```
ADD COLUMN `townid` INT NULL AFTER `age`;

ALTER TABLE `minions`
ADD CONSTRAINT `fk_minions_towns`
  FOREIGN KEY (`id`)
  REFERENCES `towns` (`id`)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION;
```

Задача 1.3. Създаване на база данни school

Създайте нова база данни наречена **school**. Добавете следните таблици: **students** (id, name, age, phone_number), **classes** (id, name, max_students), **teachers** (id, name, class), **towns** (id, name). Направете id полетата първични ключове. Добавете данни в таблиците.

Решение

```
CREATE SCHEMA IF NOT EXISTS `school`;
USE `school`;

CREATE TABLE IF NOT EXISTS `students`
(
    `id` INT(3) NOT NULL,
    `name` VARCHAR(50) NOT NULL,
    `age` INT(3) NULL,
    `phone_number` VARCHAR(50)  NULL,
    PRIMARY KEY (`id`)
);
INSERT INTO `students` (`id`, `name`, `age`) VALUES ('1', 'Mitko', '42');
INSERT INTO `students` (`id`, `name`, `age`) VALUES ('2', 'Ani', '36');
INSERT INTO `students` (`id`, `name`, `age`) VALUES ('3', 'Peter', '4');

CREATE TABLE IF NOT EXISTS `classes`
(
    `id` INT(3) NOT NULL,
    `name` VARCHAR(50) NOT NULL,
    `max_students` INT(3) NULL,
    PRIMARY KEY (`id`)
);
INSERT INTO `classes` (`id`, `name`, `max_students`) VALUES ('1', 'Introduction
to programming', '15');
INSERT INTO `classes` (`id`, `name`, `max_students`) VALUES ('2', 'Algorithms and
data structure', '10');
```

```
INSERT INTO `classes` (`id`, `name`, `max_students`) VALUES ('3', 'Databases',
'2');

CREATE TABLE IF NOT EXISTS `teachers`
(
    `id` INT(3) NOT NULL,
    `name` VARCHAR(50) NOT NULL,
    `class` TEXT NULL,
    PRIMARY KEY (`id`)
);
INSERT INTO `teachers` (`id`, `name`, `class`) VALUES ('1', 'Dimitar Minchev',
'Introduction to programming');
INSERT INTO `teachers` (`id`, `name`, `class`) VALUES ('2', 'Dimitar Minchev',
'Algorithms and data structure');
INSERT INTO `teachers` (`id`, `name`, `class`) VALUES ('3', 'Dimitar Minchev',
'Databases');

CREATE TABLE IF NOT EXISTS `towns`
(
   `id` INT(3) NOT NULL,
   `name` VARCHAR(40) NOT NULL,
   PRIMARY KEY (`id`)
);
INSERT INTO `towns` (`id`,`name`) VALUES (1,"Burgas");
INSERT INTO `towns` (`id`,`name`) VALUES (2,"Sofia");
INSERT INTO `towns` (`id`,`name`) VALUES (3,"Varna");

ALTER TABLE `students`
ADD COLUMN `townid` INT NULL AFTER `phone_number`;

ALTER TABLE `students`
ADD CONSTRAINT `fk_students_towns`
  FOREIGN KEY (`id`)
  REFERENCES `towns` (`id`)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION;
```

Задача 1.4. Създайте таблицата People

С помощта на SQL заявка, създайте таблица users с колони:

- id – unique number for every person there will be no more than people. (Auto incremented)
- name – full name of the person will be no more than 200 Unicode characters. (Not null)

- picture – image with size up to 2 MB. (Allow nulls)
- height – In meters. Real number precise up to 2 digits after floating point. (Allow nulls)
- weight – In kilograms. Real number precise up to 2 digits after floating point. (Allow nulls)
- gender – Possible states are m or f. (Not null)
- birthdate – (Not null)
- biography – detailed biography of the person it can contain max allowed Unicode characters. (Allow nulls)
- id – уникален номер за всеки човек там ще бъде не повече от 2^{31}-1 души. (Автоувеличава)
- name – пълно име на лицето ще бъде не повече от 200 Unicode символа. (Не null)
- picture – изображение с размер до 2 MB. (Разрешени са празни стойности)
- height – в метри. Реално число с точност до 2 цифри след десетична запетая. (Разрешени са празни стойности)
- weight – в килограми. Реално число с точност до 2 цифри след десетичната запетая. (Разрешени са празни стойности)
- gender- възможни състояния са m или f. (не null)
- birthdate – (не null)
- biography –подробна биография на лицето, може да съдържа максимално позволен брой знаци в Unicode. (Разреши празни стойности)

Направете ИД първичен ключ. Попълните таблицата с 5 записа. Изпратете вашите CREATE и INSERT заявки до системата за проверка като стартирате заявките и проверите базата от данни (Run queries & check DB).

Решение

```
CREATE DATABASE IF NOT EXISTS `people`;
USE `people`;

CREATE TABLE IF NOT EXISTS `people`
(
    `id` INT AUTO_INCREMENT NOT NULL UNIQUE,
    `name` VARCHAR(200) NOT NULL,
    `picture` MEDIUMBLOB NULL,
    `height` NUMERIC(10,2) NULL,
    `weight` NUMERIC(10,2) NULL,
    `gender` CHAR(1) NULL,
    `birthdate` DATETIME NOT NULL,
    `biography` VARCHAR(10000) NULL,
    CONSTRAINT pk_people PRIMARY KEY (`id`)
);
```

```
INSERT INTO `people` (`name`,`height`,`weight`,`gender`,`birthdate`)
VALUES ("Dimitar Minchev", "183","85","m","1978-08-23");
INSERT INTO `people` (`name`,`height`,`weight`,`gender`,`birthdate`)
VALUES ("Anelia Tzvetanova", "172","65","f","1986-09-11");
INSERT INTO `people` (`name`,`height`,`weight`,`gender`,`birthdate`)
VALUES ("Peter Minchev", "106","15","m","2015-06-30");
```

Задача 1.5. Направете таблицата потребители Users

С помощта на SQL заявка създайте таблица users с колони:

- id – unique number for every user. There will be no more than users. (Auto incremented)
- username – unique identifier of the user will be no more than 30 characters (non Unicode). (Required)
- password – password will be no longer than 26 characters (non Unicode). (Required)
- profile_picture – image with size up to 900 KB.
- last_login_time
- is_deleted – shows if the user deleted his/her profile. Possible states are true or false.
- id – уникално число за всеки потребител. Ще има не повече от 2^{63-1} потребители. (автоувеличаване)
- name – уникален идентификатор на потребителя, ще бъде не повече от 30 знака (не Unicode). (Задължително)
- password - парола ще бъде не по-дълга от 26 символа (не Unicode). (Задължително)
- profile_picture – изображение с размер до 900 KB.
- last_login_time
- is_deleted - показва, ако потребителят е изтрил профила си. Възможни състояния са true или false.

Направете ИД първичен ключ. Попълните таблицата с 5 записа. Изпратете CREATE и INSERT заявки. Изпратете заявката си до системата за проверка като стартирате заявките и проверите базата от данни (Run queries & check DB).

Решение

```
CREATE TABLE IF NOT EXISTS `users`
(
    `id` BIGINT AUTO_INCREMENT NOT NULL,
    `username` VARCHAR(30) NOT NULL,
    `password` VARCHAR(26) NOT NULL,
    `profile_picture` BLOB,
    `last_login_time` DATETIME,
    `is_deleted` INT NULL,
```

```
    CONSTRAINT `pk_users` PRIMARY KEY (`id`)
);

INSERT INTO `users` (`username`,`password`,`last_login_time`,`is_deleted`)
VALUES ("Pesho","1234",now(),0);
INSERT INTO `users` (`username`,`password`,`last_login_time`,`is_deleted`)
VALUES ("Maria","qwerty",now(),0);
INSERT INTO `users` (`username`,`password`,`last_login_time`,`is_deleted`)
VALUES ("Georgi","gogo",now(),1);
```

Задача 1.6. База данни Movies

С помощта на SQL заявки създайте база данни за филми **movies** със следните записи:

- directors (id, director_name, notes)
- genres (id, genre_name, notes)
- categories (id, category_name, notes)
- movies (id, title, director_id, copyright_year, length, genre_id, category_id, rating, notes)

Задайте най-подходящи типове данни за всяка колона. Задайте първичен ключ за всяка таблица. Попълнете всяка таблица с 5 записа. Уверете се, че колоните, които се намират в 2 таблици ще бъдат от един и същ тип данни. Помислете кои полета са винаги задължителни и които не са задължителни. . Изпратете заявката си до системата за проверка като стартирате заявките и проверите базата от данни (Run queries & check DB).

Решение

```
CREATE DATABASE IF NOT EXISTS `movies`;
USE `movies`;

CREATE TABLE IF NOT EXISTS `directors`
(
    `id` INT AUTO_INCREMENT NOT NULL UNIQUE,
    `director_name` TEXT NULL,
    `notes` TEXT NULL,
    CONSTRAINT `pk_directors` PRIMARY KEY (`id`)
);

INSERT INTO `directors` (`director_name`) VALUES ("Steven King");
INSERT INTO `directors` (`director_name`) VALUES ("Steven Spelberg");
INSERT INTO `directors` (`director_name`) VALUES ("Steven Segal");
INSERT INTO `directors` (`director_name`) VALUES ("Steven Hawkins");

CREATE TABLE IF NOT EXISTS `genres`
```

```
(
    `id` INT AUTO_INCREMENT NOT NULL UNIQUE,
    `genre_name` TEXT NULL,
    `notes` TEXT NULL,
    CONSTRAINT `pk_genres` PRIMARY KEY (`id`)
);

INSERT INTO `genres` (`genre_name`) VALUES ("comedy");
INSERT INTO `genres` (`genre_name`) VALUES ("tragedy");
INSERT INTO `genres` (`genre_name`) VALUES ("drama");
INSERT INTO `genres` (`genre_name`) VALUES ("horror");
INSERT INTO `genres` (`genre_name`) VALUES ("science - fiction");

CREATE TABLE IF NOT EXISTS `categories`
(
    `id` INT AUTO_INCREMENT NOT NULL UNIQUE,
    `category_name` TEXT NULL,
    `notes` TEXT NULL,
    CONSTRAINT `pk_categories` PRIMARY KEY (`id`)
);

INSERT INTO `categories` (`category_name`) VALUES ("16+");
INSERT INTO `categories` (`category_name`) VALUES ("18+");
INSERT INTO `categories` (`category_name`) VALUES ("21+");

CREATE TABLE IF NOT EXISTS `movies`
(
    `id` INT AUTO_INCREMENT NOT NULL UNIQUE,
    `title` TEXT NULL,
    `director_id` INT NOT NULL,
    `copyright_year` DATE NULL,
    `length` TIME NULL,
    `genre_id` INT NOT NULL,
    `category_id` INT NOT NULL,
    `rating` INT(1) DEFAULT 0,
    `notes` TEXT NULL,
    CONSTRAINT `pk_movies` PRIMARY KEY (`id`),
    CONSTRAINT `fk_director` FOREIGN KEY (`director_id`) REFERENCES
`directors`(`id`),
    CONSTRAINT `fk_genre` FOREIGN KEY (`genre_id`) REFERENCES `genres`(`id`),
    CONSTRAINT `fk_category` FOREIGN KEY (`category_id`) REFERENCES
`categories`(`id`)
```

```
);

INSERT INTO `movies` (`title`, `director_id`, `copyright_year`, `length`,
`genre_id`, `category_id`, `rating`)
VALUES ('Game of nerves', '4','1982-01-01','01:20','1','3','5');
 INSERT INTO `movies` (`title`, `director_id`, `copyright_year`, `length`,
`genre_id`, `category_id`, `rating`)
VALUES ('Star Wars', '2','1985-02-03','02:10','5','1','5');
 INSERT INTO `movies` (`title`, `director_id`, `copyright_year`, `length`,
`genre_id`, `category_id`, `rating`)
VALUES ('2 men and 1/2', '3','2009-04-05','00:47','3','1','4');
```

Задача 1.7. База данни Коли под наем

С помощта на SQL заявки създаване база от данни **carrental** със следните таблици:

- categories (id, category, daily_rate, weekly_rate, monthly_rate, weekend_rate)
- cars (id, plate_number, make, model, car_year, category_id, doors, picture, car_condition, available)
- employees (id, first_name, last_name, title, notes)
- customers (id, driver_licence_number, full_name, address, city, zip_code, notes)
- rental_orders (id, employee_id, customer_id, car_id, car_condition, tank_level, kilometrage_start, kilometrage_end, total_kilometrage, start_date, end_date, total_days, rate_applied, tax_rate, order_status, notes)

Задайте най-подходящи типове данни за всяка колона. Задай първичен ключ за всяка таблица. Попълнете всяка таблица с 3 записа. Уверете се, че колоните, които се намират в 2 таблици ще бъдат от един и същ тип данни. Помислете кои полета са винаги задължителни и които не са задължителни . Изпратете заявката си до системата за проверка като стартирате заявките и проверите базата от данни (Run queries & check DB).

Решение

```
CREATE DATABASE IF NOT EXISTS `carrental`;
USE `carrental`;

CREATE TABLE IF NOT EXISTS `categories`
(
    `id` INT NOT NULL AUTO_INCREMENT,
    `category` VARCHAR(50) NOT NULL,
    `daily_rate` INT NULL,
    `weekly_rate` INT NULL,
    `monthly_rate` INT NULL,
    `weekend_rate` INT NULL,
```

```
    PRIMARY KEY(id)
);

CREATE TABLE IF NOT EXISTS `car`
(
    `id` INT NOT NULL AUTO_INCREMENT,
    `model` VARCHAR(50) NOT NULL,
    `plate_number` VARCHAR(50) NOT NULL,
    `car_year` DATETIME NULL,
    `category_id` INT NOT NULL,
    `doors` INT NULL,
    `picture` BLOB NULL,
    `car_condition` VARCHAR(50) NULL,
    `available` BOOL NOT NULL,
    PRIMARY KEY(`id`)
);

CREATE TABLE IF NOT EXISTS `employees`
(
    `id` INT NOT NULL AUTO_INCREMENT,
    `first_name` VARCHAR(50) NOT NULL,
    `last_name` VARCHAR(50) NOT NULL,
    `title` VARCHAR(50) NULL,
    `notes` TEXT NULL,
    PRIMARY KEY(`id`)
);

CREATE TABLE IF NOT EXISTS `customers`
(
    `id` INT NOT NULL AUTO_INCREMENT,
    `driver_licence_number` VARCHAR(50) NOT NULL,
    `full_name` VARCHAR(50) NOT NULL,
    `address` VARCHAR(50) NULL,
    `city` VARCHAR(50) NULL,
    `zip_code` VARCHAR(50) NULL,
    `notes` TEXT NULL,
    PRIMARY KEY(`id`)
);

CREATE TABLE IF NOT EXISTS `rental_orders`
(
    `id` INT NOT NULL AUTO_INCREMENT,
```

```
    `employee_id` INT NOT NULL,
    `customer_id` INT NOT NULL,
    `car_id` INT NULL,
    `car_condition` VARCHAR(50) NULL,
    `tank_level` INT NULL,
    `kilometrage_start` INT NULL,
    `kilometrage_end` INT NULL,
    `total_kilometrage` INT NULL,
    `start_date` DATETIME NULL,
    `end_date` DATETIME NULL,
    `total_days` INT NULL,
    `rate_applied` VARCHAR(50) NULL,
    `tax_rate` VARCHAR(50) NULL,
    `order_status` VARCHAR(50) NULL,
    `notes` TEXT NULL,
    PRIMARY KEY(`id`)
);

ALTER TABLE `car`
ADD CONSTRAINT `fk_car_category` FOREIGN KEY(`id`)
    REFERENCES `car-rental`.`categories`(`id`)
    ON DELETE NO ACTION
    ON UPDATE NO action;

ALTER TABLE `rental_orders`
ADD CONSTRAINT `fk_employee_order` FOREIGN KEY(`id`)
    REFERENCES `car-rental`.`employees`(`id`)
    ON DELETE NO ACTION
    ON UPDATE NO action;

ALTER TABLE `rental_orders`
ADD CONSTRAINT `fk_car_order` FOREIGN KEY(`id`)
    REFERENCES `car-rental`.`car`(`id`)
    ON DELETE NO ACTION
    ON UPDATE NO action;

ALTER TABLE `rental_orders`
ADD CONSTRAINT `fk_order_customer` FOREIGN KEY(`id`)
    REFERENCES `car-rental`.`customers`(`id`)
    ON DELETE NO ACTION
    ON UPDATE NO action;
```

Задача 1.8. База данни Хотел

С помощта на SQL заявките създаде база от данни **hotel** със следните таблици:

- employees (id, first_name, last_name, title, notes)
- customers (account_number, first_name, last_name, phone_number, emergency_name, emergency_number, notes)
- room_status (room_status, notes)
- room_types (room_type, notes)
- bed_types (bed_type, notes)
- rooms (room_number, room_type, bed_type, rate, room_status, notes)
- payments (id, employee_id, payment_date, account_number, first_date_occupied, last_date_occupied, total_days, amount_charged, tax_rate, tax_amount, payment_total, notes)
- occupancies (id, employee_id, date_occupied, account_number, room_number, rate_applied, phone_charge, notes)

Задайте най-подходящи типове данни за всяка колона. Задай първичен ключ за всяка таблица. Попълнете всяка таблица с 3 записа. Уверете се, че колоните, които се намират в 2 таблици ще бъдат от един и същ тип данни. Помислете кои полета са винаги задължителни и които не са задължителни. . Изпратете заявката си до системата за проверка като стартирате заявките CREATE TABLE и INSERT и проверите базата от данни (Run queries & check DB).

Решение

```
CREATE SCHEMA IF NOT EXISTS `hotel`;
USE `hotel`;

CREATE TABLE IF NOT EXISTS `employees`
(
    `id` INT NOT NULL,
    `first_name` TEXT,
    `last_name` TEXT,
    `title` TEXT,
    `notes` TEXT,
    PRIMARY KEY (`id`)
);

CREATE TABLE IF NOT EXISTS `customers`
(
    `account_number` INT NOT NULL,
    `first_name` TEXT,
    `last_name` TEXT,
    `phone_number` TEXT,
```

```
    `emergency_name` TEXT,
    `emergency_number` TEXT,
    `notes` TEXT,
    PRIMARY KEY (`id`)
);

CREATE TABLE IF NOT EXISTS `room_status`
(
    `room_status` INT NOT NULL,
    `notes` TEXT,
    PRIMARY KEY (`id`)
);

CREATE TABLE IF NOT EXISTS `room_types`
(
    `room_type` INT NOT NULL,
    `notes` TEXT,
    PRIMARY KEY (`id`)
);

CREATE TABLE IF NOT EXISTS `bed_types`
(
    `bed_type` INT NOT NULL,
    `notes` TEXT,
    PRIMARY KEY (`id`)
);

CREATE TABLE IF NOT EXISTS `rooms`
(
    `room_number` INT NOT NULL,
    `room_type` INT,
    `bed_type` INT,
    `rate` INT,
    `room_status` INT,
    `notes` TEXT,
    PRIMARY KEY (`id`)
);

CREATE TABLE IF NOT EXISTS `payments`
(
    `id` INT NOT NULL,
    `employee_id` INT,
```

```
    `payment_date` DATE,
    `account_number` INT,
    `first_date_occupied` DATE,
    `last_date_occupied` DATE,
    `total_days` INT,
    `tax_rate` FLOAT,
    `tax_amount` FLOAT,
    `payment_total` FLOAT,
    `notes` TEXT,
    PRIMARY KEY (`id`)
);

CREATE TABLE IF NOT EXISTS `occupancies`
(
    `id` INT NOT NULL,
    `employee_id` INT,
    `date_occupied` DATE,
    `account_number` INT,
    `room_number` INT,
    `rate_applied` INT,
    `phone_charge` INT,
    `notes` TEXT,
    PRIMARY KEY (`id`)
);
```

Тема 2. Моделиране на релационни бази данни

Задача 2.1. One-To-One връзка

Създайте две таблици. Използвайте подходящи типове данни.

Persons			
person_id	first_name	Salary	passport_id
1	Roberto	43300.00	102
2	Tom	56100.00	103
3	Yana	60200.00	101

Passports	
passport_id	passport_number
101	N34FG21B
102	K65LO4R7
103	ZE657QP2

Вкарайте данните от примера по-горе.

Изменете таблицата persons, така че да направите person_id първичен ключ. Създайте външен ключ между persons и passports чрез passport_id колоната.

Решение

```
/* Create Tables */
CREATE TABLE `passports`
(
    `passport_id` INT NOT NULL PRIMARY KEY,
    `passport_number` TEXT
);

CREATE TABLE `persons`
(
    `person_id` INT NOT NULL PRIMARY KEY,
    `first_name` TEXT,
    `salary` FLOAT,
    `passport_id` INT,

    /* Foreign key: passport_id => passports.passport_id */
    CONSTRAINT `fk_persons_passports`
    FOREIGN KEY (`passport_id`)
    REFERENCES `passports`(`passport_id`)
);

/* Insert Data */
INSERT INTO `passports` (`passport_id`,`passport_number`) VALUES
(101,"N34FG21B");
INSERT INTO `passports` (`passport_id`,`passport_number`) VALUES
(102,"K65LO4R7");
INSERT INTO `passports` (`passport_id`,`passport_number`) VALUES
(103,"ZE657QP2");

INSERT INTO `persons` (`person_id`,`first_name`,`salary`,`passport_id`) VALUES
(1,"Roberto",43300.00,102);
INSERT INTO `persons` (`person_id`,`first_name`,`salary`,`passport_id`) VALUES
(2,"Tom",56100.00,103);
INSERT INTO `persons` (`person_id`,`first_name`,`salary`,`passport_id`) VALUES
(3,"Yana",60200.00,101);
```

Задача 2.2. One-To-Many връзка

Създайте две таблици. Използвайте подходящи типове данни.

manufacturers		
manufacturer_id	name	established_on
1	BMW	01/03/1916
2	Tesla	01/01/2003
3	Lada	01/05/1966

models		
model_id	name	manufacturer_id
101	X1	1
102	i6	1
103	Model S	2
104	Model X	2
105	Model 3	2
106	Nova	3

Вкарайте данните от примера по-горе. Добавете първични ключове и външни ключове.

Решение

```
/* Create Tables */
CREATE TABLE `manufacturers`
(
    `manufacturer_id` INT NOT NULL PRIMARY KEY,
    `name` TEXT,
    `established_on` DATE
);

CREATE TABLE `models`
(
    `model_id` INT NOT NULL PRIMARY KEY,
    `name` TEXT,
    `manufacturer_id` INT,

    /* Foreign key: manufacturer_id => manufacturers.manufacturer_id */
    CONSTRAINT `fk_models_manufacturers`
    FOREIGN KEY (`manufacturer_id`)
    REFERENCES `manufacturers`(`manufacturer_id`)
);

/* Insert Data */
INSERT INTO `manufacturers` (`manufacturer_id`,`name`,`established_on`) VALUES
(1,"BMW", "01/03/1916");
INSERT INTO `manufacturers` (`manufacturer_id`,`name`,`established_on`) VALUES
(2,"Tesla","01/01/2003");
```

```
INSERT INTO `manufacturers` (`manufacturer_id`,`name`,`established_on`) VALUES
(3,"Lada","01/05/1966");

INSERT INTO `models` (`model_id`,`name`,`manufacturer_id`) VALUES (101,"X1",1);
INSERT INTO `models` (`model_id`,`name`,`manufacturer_id`) VALUES (102,"i6",1);
INSERT INTO `models` (`model_id`,`name`,`manufacturer_id`) VALUES (103,"Model
S",2);
INSERT INTO `models` (`model_id`,`name`,`manufacturer_id`) VALUES (104,"Model
X",2);
INSERT INTO `models` (`model_id`,`name`,`manufacturer_id`) VALUES (10,"Model
3",2);
INSERT INTO `models` (`model_id`,`name`,`manufacturer_id`) VALUES (106,"Nova",3);
```

Задача 2.3. Many-To-Many връзка

Създайте три таблици. Използвайте подходящи типове данни.

students	
student_id	name
1	Mila
2	Toni
3	Ron

exams	
exam_id	Name
101	Spring MVC
102	Neo4j
103	Oracle 11g

students_exams	
student_id	exam_id
1	101
1	102
2	101
3	103
2	102
2	103

Вкарайте данните от примера по-горе. Добавете първични ключове и външни ключове. Забележете, че таблицата student_exams трябва да има съставен (от повече от една колони) първичен ключ.

Решение

```
/* Create Tables */
CREATE TABLE `students`
(
```

```
    `student_id` INT NOT NULL PRIMARY KEY,
    `name` TEXT
);

CREATE TABLE `exams`
(
    `exam_id` INT NOT NULL PRIMARY KEY,
    `name` TEXT
);

CREATE TABLE `students_exams`
(
    `student_id` INT,
    `exam_id` INT,

    /* Primary key */
    CONSTRAINT `pk_students_exams`
    PRIMARY KEY(`student_id`, `exam_id`),

    /* Foreign key: student_id => students.student_id */
    CONSTRAINT `fk_student_id`
    FOREIGN KEY(`student_id`)
    REFERENCES `students`(`student_id`),

    /* Foreign key: exam_id => exams.exam_id */
    CONSTRAINT `fk_exam_id`
    FOREIGN KEY(`exam_id`)
    REFERENCES `exams`(`exam_id`)
);

/* Insert Data */
INSERT INTO `students` (`student_id`,`name`) VALUES (1,"Mila");
INSERT INTO `students` (`student_id`,`name`) VALUES (2,"Toni");
INSERT INTO `students` (`student_id`,`name`) VALUES (3,"Ron");

INSERT INTO `exams` (`exam_id`,`name`) VALUES (101,"Spring MVC");
INSERT INTO `exams` (`exam_id`,`name`) VALUES (102,"Neo4j");
INSERT INTO `exams` (`exam_id`,`name`) VALUES (103,"Oracle 11g");

INSERT INTO `students_exams` (`student_id`,`exam_id`) VALUES (1, 101);
INSERT INTO `students_exams` (`student_id`,`exam_id`) VALUES (1, 102);
INSERT INTO `students_exams` (`student_id`,`exam_id`) VALUES (2, 101);
```

```
INSERT INTO `students_exams` (`student_id`,`exam_id`) VALUES (3, 103);
INSERT INTO `students_exams` (`student_id`,`exam_id`) VALUES (2, 102);
INSERT INTO `students_exams` (`student_id`,`exam_id`) VALUES (2, 103);
```

Задача 2.4. Самообръщаща се връзка

Създайте една таблица. Използвайте подходящи типове данни.

teachers		
teacher_id	name	manager_id
101	John	
102	Maya	106
103	Silvia	106
104	Ted	105
105	Mark	101
106	Greta	101

Вкарайте данните от примера по-горе. Добавете първични ключове и външни ключове. Външният ключ трябва да е между manager_id и teacher_id.

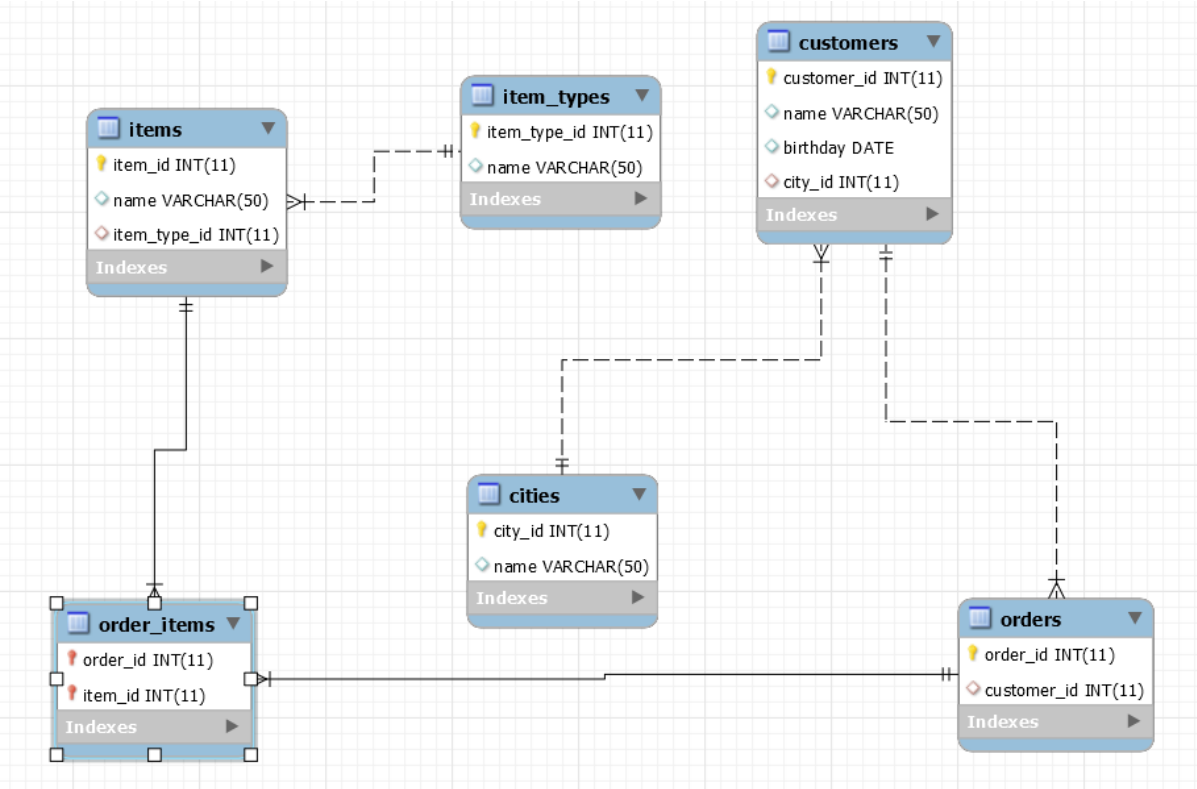
Решение

```
/* Create Table */
CREATE TABLE `teachers`
(
    `teacher_id` INT PRIMARY KEY,
    `name` TEXT,
    `manager_id` INT,

    /* Foreign key: manager_id => teachers.teacher_id */
    CONSTRAINT `fk_teacher_manager`
    FOREIGN KEY (`manager_id`)
    REFERENCES `teachers`(`teacher_id`)
);

/* Insert Data*/
INSERT INTO `teachers` (`teacher_id`,`name`) VALUES (101,"John");
INSERT INTO `teachers` (`teacher_id`,`name`,`manager_id`) VALUES
(105,"Mark",101);
INSERT INTO `teachers` (`teacher_id`,`name`,`manager_id`) VALUES
(106,"Greta",101);
INSERT INTO `teachers` (`teacher_id`,`name`,`manager_id`) VALUES
(102,"Maya",106);
```

```
INSERT INTO `teachers` (`teacher_id`,`name`,`manager_id`) VALUES
(103,"Silvia",106);
INSERT INTO `teachers` (`teacher_id`,`name`,`manager_id`) VALUES (104,"Ted",105);
```

Задача 2.5. База данни за онлайн магазин

Създайте нова база данни **onlineshop** според следната структра:



Решение

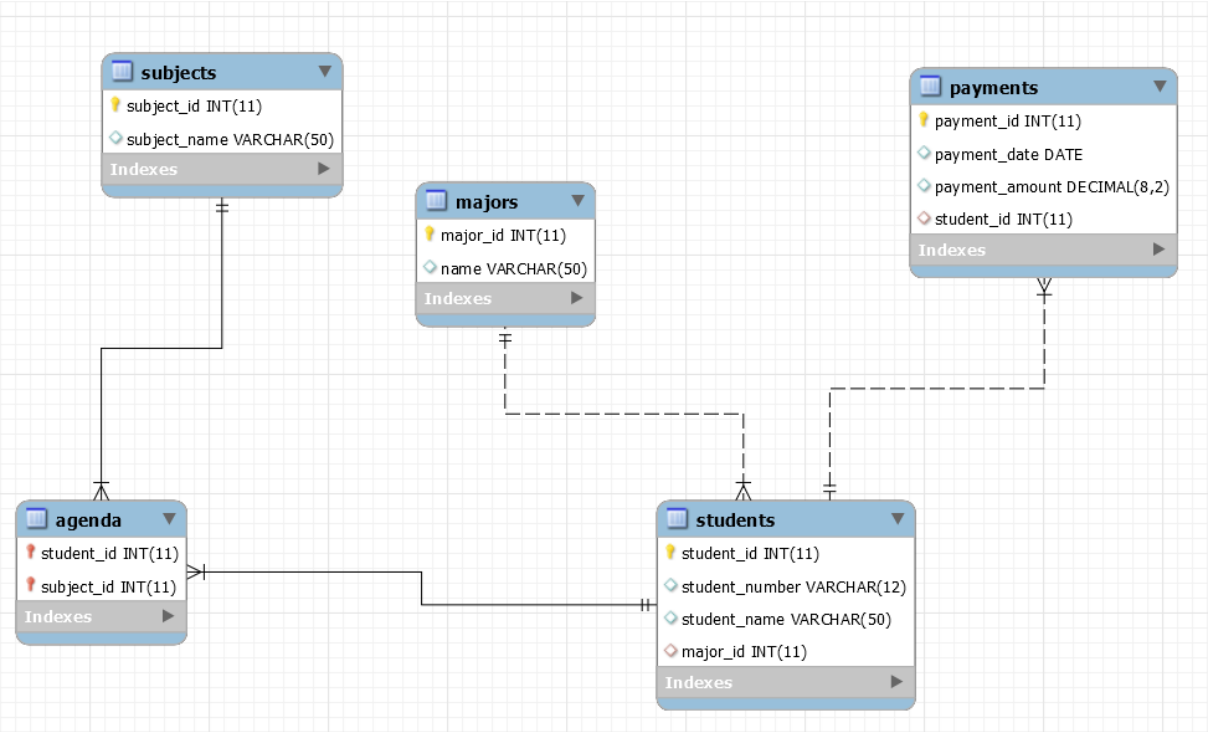
```
CREATE SCHEMA `onlineshop`;
USE `onlineshop`;

CREATE TABLE `cities`
(
   `city_id` INT(11) PRIMARY KEY,
   `name` VARCHAR(50)
);
CREATE TABLE `item_types`
(
   `item_type_id` INT(11) PRIMARY KEY,
   `name` VARCHAR(50)
);
```

```
CREATE TABLE `items`
(
   `item_id` INT(11) PRIMARY KEY,
   `name` VARCHAR(50),
   `item_type_id` INT(11),
   CONSTRAINT `fk_item_type`
   FOREIGN KEY(`item_type_id`)
   REFERENCES `item_types`(`item_type_id`)
);
CREATE TABLE `customers`
(
   `customer_id` INT(11) PRIMARY KEY,
   `name` VARCHAR(50),
   `birthday` DATE,
   `city_id` INT(11),
   CONSTRAINT `fk_city`
   FOREIGN KEY(`city_id`)
   REFERENCES `cities`(`city_id`)
);
CREATE TABLE `orders`
(
   `order_id` INT(11) PRIMARY KEY,
   `customer_id` INT(11),
   CONSTRAINT `fk_customer`
   FOREIGN KEY(`customer_id`)
   REFERENCES `customers`(`customer_id`)
);
CREATE TABLE `order_items`
(
   `order_id` INT(11),
   `item_id` INT(11),

   CONSTRAINT `pk_order_items`
   PRIMARY KEY(`order_id`, `item_id`),

   CONSTRAINT `fk_orders`
   FOREIGN KEY(`order_id`)
   REFERENCES `orders`(`order_id`),

   CONSTRAINT `fk_items`
   FOREIGN KEY(`item_id`)
   REFERENCES `items`(`item_id`)
```

```
);
```

Задача 2.6. Университетска база данни

Създайте нова база данни **university** според следната структура:



Решение

```
create schema `university`;
use `university`;

create table `subjects`
(
    `subject_id` INT(11) NOT NULL primary key,
    `subject_name` varchar(50)
);

create table `majors`
(
    `major_id` INT(11) NOT NULL primary key,
    `name` varchar(50)
);

create table `students`
(
```

```
    `student_id` INT(11) NOT NULL primary key,
    `student_number` varchar(12),
    `student_name` varchar(50),
    `major_id` int(11),
     constraint `fk_major`
    foreign key(`major_id`)
    references `majors` (`major_id`)
);

create table `payments`
(
    `payment_id` INT(11) NOT NULL primary key,
    `payment__date` date,
    `payment_amount` decimal(8,2),
    `student_id` int(11),
     constraint `fk_student`
    foreign key(`student_id`)
    references `students` (`student_id`)
);

create table `agenda`
(
    `student_id` int(11),
    `subject_id` int(11),

    constraint `pk_student_subject`
    primary key(`student_id`,`subject_id`),

    constraint `fk_students`
    foreign key(`student_id`)
    references `students` (`student_id`),

    constraint `fk_subjects`
    foreign key(`subject_id`)
    references `subjects` (`subject_id`)
);
```

Тема 3. Заявки за извличане и промяна на данни

Задача 3.0. Разглеждане на Базата от Данни

Изтеглете и се запознайте с **базите данни** softuni, diablo и geography, схемите и таблиците. Ще ги използвате в това и следващите упражнения за да напишете заявките.

Задача 3.1. Намерете цялата информация за отделите

Напишете SQL заявка, за да намерите цялата налична информация за отделите. Изпратете вашите заявки като Подготвите БД & изпълните заявките (Prepare DB & run queries.)

Пример

department_id	name	manager_id
1	Engineering	12
2	Tool Design	4
3	Sales	273
...	...	...

Решение

```
SELECT `department_id`, `name`, `manager_id`
FROM `soft_uni`.`departments`;
```

Задача 3.2. Намерете всички имена на отдели

Напишете SQL заявка, за да намерите всички имена на отдел. Изпратете вашите заявки като Подготвите БД & изпълните заявките (Prepare DB & run queries.)

Пример

name
Engineering
Tool Design
Sales
...

Решение

```
SELECT DISTINCT `name`
FROM `soft_uni`.`departments`;
```

Задача 3.3. Намерете заплатата на всеки служител

Напишете SQL заявка, която намера собственото име, фамилното име и заплата на всеки служител. Изпратете вашите заявки като Подготвите БД & изпълните заявките (Prepare DB & run queries.)

Пример

first_name	last_name	salary
Guy	Gilbert	12500.00
Kevin	Brown	13500.00
Roberto	Tamburello	43300.00
...	...	...

Решение

```
SELECT `first_name`,`last_name`,`salary`
FROM `soft_uni`.`employees`
```

Задача 3.4. Намерете пълното име на всеки служител

Напишете SQL заявка, която намира личното, бащиното и фамилното име на всеки служител. Изпратете вашите заявки като Подготвите БД & изпълните заявките (Prepare DB & run queries.)

Пример

first_name	middle_name	last_name
Guy	R	Gilbert
Kevin	F	Brown
Roberto	NULL	Tamburello
...	...	...

Решение

```
SELECT `first_name`,`middle_name`,`last_name`
FROM `soft_uni`.`employees`
```

Задача 3.5. Намерете имейл адреса на всеки служител

Напишете SQL заявка, която намира имейл адреса на всеки служител. (от неговото собствено и фамилно име). приемете, че имейл домейна е softuni.bg. Имейлите трябва да изглеждат като "John.Doe@softuni.bg". Резултатната колона трябва да бъде наречена "full_ email_address". Изпратете вашите заявки като Подготвите БД & изпълните заявките (Prepare DB & run queries.)

Пример

full_email_address
Guy.Gilbert@softuni.bg
Kevin.Brown@softuni.bg
Roberto.Tamburello@softuni.bg
...

Решение

```
SELECT concat(`first_name`,".",`last_name`,"@softuni.bg") AS 'full_email_address'
FROM `soft_uni`.`employees`
```

Задача 3.6. Намерете всички различни работни заплати

Напишете SQL заявка, за да намерите всичките различни работни заплати. Покажи само заплатите. Изпратете вашите заявки като Подготвите БД & изпълните заявките (Prepare DB & run queries.)

Пример

Salary
000.00
9300.00
9500.00
...

Решение

```
SELECT DISTINCT `salary`
FROM `soft_uni`.`employees`;
```

Задача 3.7. Намерете цялата информация за служители

Напишете SQL заявка, за да намерите цялата информация за служителите, чиято длъжност е "Търговски представител" (Sales Representative). Изпратете вашите заявки като Подготвите БД & изпълните заявките (Prepare DB & run queries.)

Пример

id	First Name	Last Name	Middle Name	Job Title	DeptID	Mngr ID	Hire Date	salary	address_id
275	Michael	Blythe	G	Sales Representative	3	268	...	23100.00	60
276	Linda	Mitchell	C	Sales Representative	3	268	...	23100.00	170
277	Jillian	Carson	NULL	Sales Representative	3	268	...	23100.00	61
...	...	...	...	...	...	...	...	...	...

Решение

```
SELECT
    `employee_id` as "id",
    `first_name` as "First Name",
    `last_name` AS "Last Name",
    `middle_name` AS "Middle Name",
    `job_title` AS "Job Title",
    `department_id` AS "DeptID",
    `manager_id` AS "Mngr ID",
    `hire_date` AS "HireDate",
    `salary`,`address_id`
FROM `soft_uni`.`employees`
WHERE `job_title` = 'Sales Representative';
```

Задача 3.8. Намерете имената на всички служители със заплата в диапазон

Напишете SQL заявка, която намира личното име, фамилно име и длъжност на всички служители, чиято заплата е в диапазона [20000, 30000]. Изпратете вашите заявки като Подготвите БД & изпълните заявките (Prepare DB & run queries.)

Пример

first_name	last_name	JobTitle
Rob	Walters	Senior Tool Designer
Thierry	D'Hers	Tool Designer
JoLynn	Dobney	Production Supervisor
...	...	...

Решение

```
SELECT `first_name`,`last_name`,`job_title` AS "JobTitle"
FROM `soft_uni`.`employees`
```

```
WHERE `salary` BETWEEN 20000 AND 30000;
```

Задача 3.9. Намерете имената на всички служители

Напишете SQL заявка, която намира пълното име на всички служители, чиято заплата е 25000, 14000, 12500 или 23600. Пълното име е комбинация от личното, бащиното и фамилното име (разделени с единичен интервал) и те трябва да бъдат в една колона, наречена "Пълно име" "Full Name". Изпратете вашите заявки като Подготвите БД & изпълните заявките (Prepare DB & run queries.)

Пример

full_name
Guy R Gilbert
Thierry B D'Hers
JoLynn M Dobney

Решение

```
SELECT concat(`first_name`," ",`middle_name`," ",`last_name`) AS "full_name"
FROM `soft_uni`.`employees`
WHERE `salary` in (25000, 14000, 12500, 23600);
```

Задача 3.10. Намерете всички служители без мениджър

Напишете SQL заявка да намери собствено и фамилно име за тези служители, които не са мениджъри. Представят отчети вашата заявка като подготвят DB & изпълните заявки. Изпратете вашите заявки като Подготвите БД & изпълните заявките (Prepare DB & run queries.)

Пример

first_name	last_name
Ken	Sanchez
Svetlin	Nakov
...	...

Решение

```
SELECT `first_name`,`last_name`
FROM `soft_uni`.`employees`
WHERE `manager_id` IS NULL;
```

Задача 3.11. Намерете всички служители със заплата повече от 50000

Напишете SQL заявка, която намира собственото име, фамилното име и заплатата на тези служители, които имат заплата повече от 50000. Подредете ги в намаляващ ред от заплата. Изпратете вашите заявки като Подготвите БД & изпълните заявките (Prepare DB & run queries.)

Пример

first_name	last_name	salary
Ken	Sanchez	125500.00
James	Hamilton	84100.00
...	...	...

Решение

```
SELECT `first_name`,`last_name`,`salary`
FROM `soft_uni`.`employees`
WHERE `salary` >= 50000;
```

Задача 3.12. Намете 5 най-добре платени служителя

Напишете SQL заявка, която намира собственото и фамилното име на 5 най-добре платени служители, подредени в низходящ ред по заплата Изпратете вашите заявки като Подготвите БД & изпълните заявките (Prepare DB & run queries.)

Пример

first_name	last_name
Ken	Sanchez
James	Hamilton
...	...

Решение

```
SELECT `first_name`,`last_name`
FROM `soft_uni`.`employees`
ORDER BY `salary` DESC
LIMIT 5;
```

Задача 3.13. Намерете всички служители, които не са от отдел Marketing

Напишете SQL заявка, която намира собственото и фамилното име на всички служители чиито номер на отдел е различен от 4. Изпратете вашите заявки като Подготвите БД & изпълните заявките (Prepare DB & run queries)

Пример

first_name	last_name
Guy	Gilbert
Roberto	Tamburello
Rob	Walters

Решение

```
SELECT `first_name`,`last_name`
FROM `soft_uni`.`employees`
WHERE `department_id` NOT IN (4);
```

Задача 3.14. Различни длъжности

Напишете SQL заявка,която намира всички различни длъжности. Изпратете вашите заявки като Подготвите БД & изпълните заявките (Prepare DB & run queries.)

Пример

Job_title
Accountant
Accounts Manager

Accounts Payable Specialist
...

Решение

```
SELECT DISTINCT `job_title`
FROM `soft_uni`.`employees`
ORDER BY `job_title`;
```

Задача 3.15. Намерете първите 10 започнати проекти

Напишете SQL заявка, която намира първите 10 започнати проекти. Изберете цялата информация за тях и ги подредете по начална дата, след това по име. Изпратете вашите заявки като Подготвите БД & изпълните заявките (Prepare DB & run queries.)

Пример

id	Name	Description	start_date	end_date
6	HL Road Frame	Research, design and development of HL Road ...	1998-05-02 00:00:00	2003-06-01 00:00:00
2	Cycling Cap	Research, design and development of C...	2001-06-01 00:00:00	2003-06-01 00:00:00
5	HL Mountain Frame	Research, design and development of HL M...	2001-06-01 00:00:00	2003-06-01 00:00:00
...	...	...	...	...

Решение

```
SELECT *
FROM `soft_uni`.`projects`
ORDER BY `start_date`,`name`
LIMIT 10;
```

Задача 3.16. Последните 7 наети служители

Напишете SQL заявка, която намира последните 7 наети служители. Изберете техните собствени имена, фамилни имена и датата им на наемане. Изпратете вашите заявки като Подготвите БД & изпълните заявките (Prepare DB & run queries.).

Пример

first_name	last_name	hire_date
Rachel	Valdez	2005-07-01 00:00:00
Lynn	Tsoflias	2005-07-01 00:00:00
Syed	Abbas	2005-04-15 00:00:00
...	...	...

Решение

```
SELECT `first_name`,`last_name`,`hire_date`
FROM `soft_uni`.`employees`
ORDER BY `hire_date` DESC
LIMIT 7;
```

Задача 3.17. Увеличаване на заплати

Напишете SQL заявка за увеличаване на заплатите на всички служители, които са в отделите Engineering, Tool Design, Marketing или Information

Services с 12 %. След това изберете колоната със заплатите от таблицата Emmployees. Изпратете вашите заявки като Подготвите БД & изпълните заявките (Prepare DB & run queries.).

Пример

Salary
12500.00
15120.00
48496.00
33376.00
...

Решение

```
SELECT `department_id` as "id"
FROM `soft_uni`.`departments`
WHERE `name` IN ("Engineering", "Tool Design", "Marketing",  "Information
Services");
/* Result: 1, 2, 4, 11 */

UPDATE `soft_uni`.`employees`
SET `salary` = `salary` * 1.12
WHERE `department_id` IN (1, 2, 4, 11);
/* Result: 28 affected rows*/
```

Задача 3.18. Всички планински върхове

Показване на всички планински върхове в азбучен ред. . Изпратете вашите заявки като Подготвите БД & изпълните заявките (Prepare DB & run queries.)

Пример

peak_name
Aconcagua
Banski Suhodol
Batashki Snezhnik
...

Решение

```
SELECT `peak_name`
FROM `geography`.`peaks`
ORDER BY `peak_name`;
```

Задача 3.19. Най-големи по население страни

Намерете 30 най-големи по население страни в Европа. Покажете името на страната и населението. Сортирайте резултатите по население (от най-големите до най-малката), след това по страна по азбучен ред.. Изпратете вашите заявки като Подготвите БД & изпълните заявките (Prepare DB & run queries.).

Пример

country_name	population

Russia	140702000
Germany	81802257
France	64768389
...	...

Решение

```
SELECT `country_name`,`population`
FROM `geography`.`countries`
WHERE `continent_code`="EU"
ORDER BY `population` DESC
LIMIT 30;
```

Задача 3.20. Страни и валута (Euro / Not Euro)

Намерете всички страни заедно с информация за своята валута. Покажете името на страната, код на страната и информация за валутата ѝ: "Euro" или "Not euro". Сортирайте резултатите по име на страната по азбучен ред. . Изпратете вашите заявки като Подготвите БД & изпълните заявките (Prepare DB & run queries.).

Пример

country_name	country_code	currency
Afghanistan	AF	Not Euro
Aland	AX	Euro
Albania	AL	Not Euro
...	...	...

Решение

```
SELECT `country_name`,`country_code`,
IF(`currency_code` = "EUR", "Euro" , "Not Euro") as "Currency"
FROM `geography`.`countries`
ORDER BY `country_name`;
```

Задача 3.21. Всички символи Diablo

Изведете всички символи в азбучен ред . Изпратете вашите заявки като Подготвите БД & изпълните заявките (Prepare DB & run queries.)

Пример

name
Amazon
Assassin
Barbarian
...

Решение

```
SELECT `Name`
FROM `diablo`.`characters`
ORDER BY `Name`;
```

Задача 3.22. Вмъкване на данни

Използвайте базата данни на SoftUni и вмъкнете някои данни, като SQL заявки.

- towns: Sofia, Plovdiv, Varna, Burgas
- departments: Engineering, Sales, Marketing, Software Development, Quality Assurance
- employees:

name	job_title	department	hire_date	salary
Ivan Ivanov Ivanov	.NET Developer	Software Development	01/02/2013	3500.00
Petar Petrov Petrov	Senior Engineer	Engineering	02/03/2004	4000.00
Maria Petrova Ivanova	Intern	Quality Assurance	28/08/2016	525.25
Georgi Terziev Ivanov	CEO	Sales	09/12/2007	3000.00
Peter Pan Pan	Intern	Marketing	28/08/2016	599.88

Изпратете заявката си до системата за проверка като стартирате заявките и проверите базата от данни (Run queries & check DB).

Решение

```
use `soft_uni`;

insert into `towns` (`town_id`, `name`) values (1,"Sofia");
insert into `towns` (`town_id`, `name`) values (2,"Plovdiv");
insert into `towns` (`town_id`, `name`) values (3,"Varna");
insert into `towns` (`town_id`, `name`) values (4,"Burgas");

insert into `departments` (`department_id`, `name`) values (1,"Engineering");
insert into `departments` (`department_id`, `name`) values (2,"Sales");
insert into `departments` (`department_id`, `name`) values (3,"Marketing");
insert into `departments` (`department_id`, `name`) values (4,"Software
Development");
insert into `departments` (`department_id`, `name`) values (5,"Quality
Assurance");

insert into `employees` (`first_name`,
`middle_name`,`last_name`,`job_title`,`department`,`hire_date`,`salary`)  values
("Ivan","Ivanov","Ivanov",".NET Developer",4,"01/02/2013",3500.00);
insert into `employees` (`first_name`,
`middle_name`,`last_name`,`job_title`,`department`,`hire_date`,`salary`)  values
("Petar","Petrov","Petrov","Senior Engineer",1,"02/03/2004",4000.00);
insert into `employees` (`first_name`,
`middle_name`,`last_name`,`job_title`,`department`,`hire_date`,`salary`)  values
("Maria","Petrova','Ivanova","Intern",5,"28/08/2016",525.25);
```

```
insert into `employees` (`first_name`,
`middle_name`,`last_name`,`job_title`,`department`,`hire_date`,`salary`)  values
("Georgi","Terziev","Ivanov","CEO",2,"09/12/2007",3000.00);
insert into `employees` (`first_name`,
`middle_name`,`last_name`,`job_title`,`department`,`hire_date`,`salary`)  values
("Peter","Pan","Pan","Intern",3,"28/08/2016",599.88);
```

Задача 3.23. Основно избиране на всички полета

Използвайте базата данни softuni и първо изберете всички записи от towns, след това от departments и накрая от таблица employees. Използвайте SQL заявки и ги изпратете на системата за проверка наведнъж. Изпратете заявката си до системата за проверка като стартирате заявките и проверите базата от данни (Run queries & check DB).

Решение

```
use `soft_uni`;
select * from `towns`;
select * from `departments`;
select * from `employees`;
```

Задача 3.24. Основно избиране на няколко полета

Променете заявките от предишните задачи за да се показват само някои от колоните. За таблица:

- towns – name
- department –name
- employees – first_name, last_name, job_title, salary

Запазете подредбата от предишната задача. Изпратете заявката си до системата за проверка като стартирате заявките и проверите базата от данни (Run queries & check DB).

Решение

```
use `soft_uni`;
select `name` from `towns`;
select `name` from `departments`;
select `first_name`, `last_name`, `job_title`, `salary` from `employees`;
```

Задача 3.25. Увеличете заплатата на работника

Използвайте базата от данни softuni и увеличете заплатата на всички служители с 10 %. Изберете само колоната salary от таблицата на служителите. Изпратете заявката си до системата за проверка като стартирате заявките и проверите базата от данни (Run queries & check DB).

Решение

```
use `soft_uni`;
update `employees` set `salary` = `salary` * 1.10;
```

Задача 3.26. Намалете процента на данъка

Използвайте базата от данни hotel и намалете данъчната ставка с 3 % към всички плащания. Изберете само колоната tax_rate от таблицата payments. Изпратете заявката си до системата за проверка като стартирате заявките и проверите базата от данни (Run queries & check DB).

Решение

Задача 3.27. Изтриване на всички записи

Използвайте базата от данни Hotel и изтрийте всички записи от таблицата occupancies. Използват SQL заявка. Изпратете заявката си до системата за проверка като стартирате заявките и проверите базата от данни (Run queries & check DB).

Решение

Тема 4. Сложни заявки за извличане на данни

Задача 4.1. Намерете всички служители със заплата над 50000

Напишете заявка, с която да откриете собственото име, фамилията и заплатата на тези служители, които получават повече от 50000. Подредете ги в низходящ ред по заплата.

Пример

first_name	last_name	Salary
Ken	Sanchez	125500.00
James	Hamilton	84100.00
...	...	...

Решение

```
use `soft_uni`;
select `first_name`,`last_name`,`salary`
from `employees`
where `salary` >= 50000
order by `salary` desc;
```

Задача 4.2. Намерете 5 най-добре платени служители.

Напишете заявка, с която да откриете собствените и фамилните имена на 5 най-добре платени служители, подредени низходящо по заплатата си.

Пример

first_name	last_name

Ken	Sanchez
James	Hamilton
...	...

Решение

```
use `soft_uni`;
select `first_name`,`last_name`
from `employees`
order by `salary` desc
limit 5;
```

Задача 4.3. Сортирайте таблицата със служители

Напишете заявка, която сортира всички записи в таблицата със служители по следните критерии:

- Първо по заплата в низходящ ред
- После по собствено име по азбучен ред
- После по фамилия в низходящ ред
- После по второ име по азбучен ред

Пример

id	First Name	Last Name	Middle Name	job_title	DeptID	MngrID	HireDate	salary	address_id
109	Ken	Sanchez	J	Chief Executive Officer	16	NULL	...	125500.00	177
148	James	Hamilton	R	Vice President of Production	7	109	...	84100.00	158
273	Brian	Welcker	S	Vice President of Sales	3	109	...	72100.00	134
...	...	...	...	...	...	...	...	...	...

Решение

```
use `soft_uni`;
select *
from `employees`
order by
    `salary` desc,
    `first_name` asc,
    `last_name` desc,
    `middle_name` asc;
```

Задача 4.4. Намерете първите 10 започнати проекта

Напишете заявка, с която да откриете първите 10 започнати проекта. Изберете цялата информация за тях и ги сортирайте по начална дата, после по име.

Пример

id	Name	Description	start_date	end_date
6	HL Road Frame	Research, design and development of HL Road ...	1998-05-02 00:00:00	2003-06-01 00:00:00
2	Cycling Cap	Research, design and development of C...	2001-06-01 00:00:00	2003-06-01 00:00:00
5	HL Mountain Frame	Research, design and development of HL M...	2001-06-01 00:00:00	2003-06-01 00:00:00
...	...	...	...	...

Решение

```
use `soft_uni`;
select *
from `projects`
order by `start_date`, `name`
limit 10;
```

Задача 4.5. Последните 7 наети служителя

Напишете заявка, с която да откриете последните 7 наети служителя. Изберете техните собствено и фамилно име и датата, на която са наети.

Пример

first_name	last_name	hire_date
Rachel	Valdez	2005-07-01 00:00:00
Lynn	Tsoflias	2005-07-01 00:00:00
Syed	Abbas	2005-04-15 00:00:00
...	...	...

Решение

```
use `soft_uni`;
select `first_name`,`last_name`,`hire_date`
from `employees`
order by `hire_date` asc
limit 7;
```

Задача 4.6. Всички планински върхове

Покажете всички планински върхове, подредени по азбучен ред.

Пример

peak_name
Aconcagua
Banski Suhodol
Batashki Snezhnik
...

Решение

```
use `geography`;
select `peak_name`
from `peaks`
order by `peak_name`;
```

Задача 4.7. Най-големи държави по население

Намерете 30 най-големи държави по население в Европа. Покажете името на държавата и населението. Сортирайте резултатите по население (от най-голямо към най-малко), после по държава по азбучен ред.

Пример

country_name	population
Russia	140702000
Germany	81802257
France	64768389
...	...

Решение

```
use `geography`;
select `country_name`,`population`
from `countries`
where `continent_code` = "EU"
order by `population` desc
limit 30;
```

Задача 4.8. Държави и валути (Евро / Не евро)

Намерете всички държави заедно с информация за тяхната *валута*. Покажете името на държавата, нейният код и информация за нейната валута: или "Евро", или "Не евро". Сортирайте резултатите по име на страната по азбучен ред.

Пример

country_name	country_code	currency
Afghanistan	AF	Not Euro
Aland	AX	Euro
Albania	AL	Not Euro
...	...	...

Решение

```
use `geography`;
select `country_name`,`country_code`,
if(`currency_code`="EUR","Euro","Not Euro") as 'currency'
from `countries`
order by `country_name` asc;
```

Задача 4.9. Най-ниско платени служители

Напишете заявка, с която да намерите имената и заплатите на служителите, които получават най-ниски заплати в компанията.

Пример:

first_name	last_name	salary
Susan	Eaton	9000.0000
Kim	Ralls	9000.0000
Jimmy	Bischoff	9000.0000

Решение

```
use `soft_uni`;
select `first_name`,`last_name`,`salary`
from `employees`
where `salary` =
(
    select `salary`
    from `employees`
    order by `salary` asc
    limit 1
);
```

Задача 4.10. Служители с близки до най-ниските заплати

Напишете заявка, с която да намерите имената и заплатите на служителите (общо 35) с заплати до 10% по-високи от най-ниските в компанията.

Пример:

first_name	last_name	salary
Jimmy	Bischoff	9000.0000
Kim	Ralls	9000.0000
Susan	Eaton	9000.0000
Pat	Coleman	9300.0000
Jo	Berry	9300.0000
Stuart	Macrae	9300.0000
Lori	Penor	9300.0000
Vamsi	Kuppa	9500.0000
Benjamin	Martin	9500.0000
Jimmy	Bischoff	9000.0000
...	...	...

Решение

```
use `soft_uni`;
select `first_name`,`last_name`,`salary`
from `employees`
where `salary` BETWEEN
(
    select `salary`
    from `employees`
    order by `salary` asc
    limit 1
)
AND
(
    select (`salary`*1.10) as 'salary'
    from `employees`
```

```
    order by `salary` asc
    limit 1
)
order by `salary` asc;
```

Задача 4.11. Всички мениджъри

Напишете заявка, която показва името, фамилията и длъжността на всички мениджъри в компанията, сортирани по име и фамилия.

Пример

first_name	last_name	job_title
A. Scott	Wright	Master Scheduler
Amy	Alberts	European Sales Manager
Andrew	Hill	Production Supervisor
Brenda	Diaz	Production Supervisor
...	...	...

Решение

```
use `soft_uni`;
select `first_name`,`last_name`,`job_title`
from `employees`
where `employee_id` in
(
    select distinct `manager_id`
    from `employees`
    where `manager_id` is not null
)
order by `first_name`,`last_name`;
```

Задача 4.12. Служителите от San Francisco

Напишете заявка, която показва имената на всички служители, живеещи в град San Francisco

employee_id	employee_name
Michael	Raheem
Shu	Ito

Решение

```
use `soft_uni`;
select `employee_id`, concat(`first_name`," ",`last_name`) as 'employee_name'
from `employees`
where `address_id` in
(
    select `address_id`
    from `addresses`
    where `town_id` =
    (
        select `town_id`
        from `towns`
```

```
        where `name` = "San Francisco"
    )
);
```

Задача 4.13. Най-висока заплата по длъжности

Напишете заявка, която показва най-високата заплата, давана на служител за всяка длъжност. Списъкът да е подреден по заплати в низходящ ред и по длъжност, в азбучен.

Пример

job_title	salary
Chief Executive Officer	125500.0000
Vice President of Production	84100.0000
Vice President of Sales	72100.0000
Vice President of Engineering	63500.0000
...	...

Решение

```
USE `soft_uni`;
SELECT DISTINCT e.`job_title`, e.`salary`
FROM `employees` AS e
WHERE e.`salary` =
(
    SELECT es.`salary`
    FROM `employees` AS es
    WHERE es.`job_title` = e.`job_title`
    ORDER BY es.`salary`
    DESC LIMIT 1
)
ORDER BY e.`salary` DESC, e.`job_title`
```

Задача 4.14. Най-ниско платени служители по отдели

Напишете заявка, която извежда името и фамилията, заплатата и името на отдела на всички служители, получаващи най-ниска заплата в своя отдел. Резултатът да е сортиран по заплата, име и фамилия, във възходящ ред.

Пример

first_name	last_name	salary	department
Jimmy	Bischoff	9000.0000	Shipping and Receiving
Kim	Ralls	9000.0000	Shipping and Receiving
Susan	Eaton	9000.0000	Shipping and Receiving
Jo	Berry	9300.0000	Facilities and Maintenance
...	...	...	...

Решение

```
USE `soft_uni`;
```

```
SELECT `first_name`,`last_name`,`salary`,`department_id`
FROM `employees` AS e
WHERE `salary` =
(
    SELECT `salary`
    FROM `employees` AS em
    WHERE e.`department_id`=em.`department_id`
    ORDER BY `salary` ASC
    LIMIT 1
)
ORDER BY `salary`,`first_name`,`last_name`;
```

Задача 4.15. Мениджъри с същото презиме

Изведете името и фамилията на всички мениджъри, на които презимето им съвпада с това на някой от техните подчинени.

Пример

first_name	last_name
Andrew	Hill
Brenda	Diaz
David	Bradley
Jack	Richins
Jinghao	Liu
Jo	Brown
John	Campbell
Shane	Kim

Решение

```
USE `soft_uni`;
SELECT `manager`.`first_name`, `manager`.`last_name`
FROM `employees` as `manager`
WHERE (`manager`.`employee_id`, `manager`.`middle_name`) IN
(
    SELECT `manager_id`, `middle_name`
    FROM `employees`
)
ORDER BY `manager`.`first_name`, `manager`.`last_name`;
```

Задача 4.16. Мениджъри с по-ниска заплата

Напишете заявка, извеждаща списък на всички мениджъри с поне един подчинен, който има по-висока заплата от тяхната.

Пример

first_name	last_name
Roberto	Tamburello
Peter	Krebs

Решение

```
USE `soft_uni`;
SELECT `manager`.`first_name`, `manager`.`last_name`
FROM `employees` AS `manager`
WHERE `manager`.`employee_id` IN
(
    SELECT DISTINCT `manager_id`
    FROM `employees`
)
AND `manager`.`salary` < ANY
(
    SELECT `salary`
    FROM `employees`
    WHERE `manager_id` = `manager`.`employee_id`
);
```

Задача 4.17. Мениджъри с точно 5 подчинени

Изведете името и фамилията на всички мениджъри с точно 5 подчинени.

Пример

first_name	last_name
Cristian	Petculescu
Jeff	Hay
Katie	McAskill-White
Lori	Kane
Paula	Barreto de Mattos
Pilar	Ackerman
Shane	Kim
Zheng	Mu

Решение

```
USE `soft_uni`;
SELECT `manager`.`first_name`, `manager`.`last_name`
FROM `employees` AS `manager`
WHERE `manager`.`employee_id` IN
(
    SELECT DISTINCT `manager_id`
    FROM `employees`
)
AND EXISTS
(
    SELECT 1
    FROM `employees`
```

```
    WHERE `manager_id` = `manager`.`employee_id`
    LIMIT 4,1
)
AND NOT EXISTS
(
    SELECT 1
    FROM `employees`
    WHERE `manager_id` = `manager`.`employee_id`
    LIMIT 5,1
)
ORDER BY `manager`.`first_name`, `manager`.`last_name`;
```

Задача 4.18. Планините в България

Напишете заявка, която изброява планини в България и за всяка - най-високия ѝ връх и неговата височина. Заявката да е сортирана по височината на планините, от най-високата към най-ниската.

Пример

mountain_range	peak_name	elevation
Rila	Musala	2925
Pirin	Vihren	2914
Balkan Mountains	Botev	2376
Rhodope Mountains	Golyam Perelik	2191
Strandza	NULL	NULL
Vitosha	NULL	NULL

Решение

```
USE `geography`;
SELECT
    `mountains`.`mountain_range`,
    `peaks`.`peak_name`,
    `peaks`.`elevation`

FROM `peaks`, `mountains`

WHERE `mountains`.`id` = `peaks`.`mountain_id` AND
`mountains`.`id` IN
(
    SELECT `mountain_id`
    FROM `mountains_countries`
    WHERE `country_code`="BG"
)
ORDER BY `elevation` DESC;
```

Задача 4.19. Неописаните планини в България

Напишете заявка, която изброява имената на планините в България, за които в базата данни няма информация за върховете им.

Пример

mountain_range
Strandza
Vitosha

Решение

```
USE `geography`;
SELECT  `mountain_range`
FROM `mountains`
WHERE `mountains`.`id` IN (
        SELECT `mountain_id`
        FROM `mountains_countries`
        WHERE `country_code`="BG"
) AND `mountains`.`id` NOT IN
(
    SELECT DISTINCT `mountain_id`
    FROM `peaks`
    WHERE `mountain_id` IN (
        SELECT `mountain_id`
        FROM `mountains_countries`
        WHERE `country_code`="BG"
    )
);
```

Задача 4.20. Служители и техните мениджъри

Изведете справка, показваща списък на името и фамилията на всеки служител и имената на неговия мениджър. За тези, които нямат такъв, изведете "(no manager)". Сортирайте резултата по имената на мениджъра, в азбучен ред.

Пример

first_name	last_name	manager_name
Ken	Sanchez	(no manager)
George	Denchev	(no manager)
Svetlin	Nakov	(no manager)
Martin	Kulov	(no manager)
William	Vong	A. Scott Wright
Sairaj	Uddin	A. Scott Wright
Brian	LaMee	A. Scott Wright
Alan	Brewer	A. Scott Wright
Ranjit	Varkey Chudukatil	Amy Alberts
...	...	...

Решение

```
USE `soft_uni`;
(
    /* managers */
    SELECT `first_name`, `last_name`, 'No manager' as `manager_name`
    FROM `employees`
    WHERE `manager_id` IS NULL
)
UNION
(
    /* employees */
    SELECT `e`.`first_name`, `e`.`last_name`, concat(`m`.`first_name`,'
','`m`.`last_name`) as `manager_name`
    FROM `employees` AS `e`
    LEFT JOIN  `employees` AS `m` ON `m`.`employee_id` = `e`.`manager_id`
    WHERE `e`.`manager_id` IS NOT NULL
)
ORDER BY `manager_name`;
```

Задача 4.21. Тримата най-добре платени

Изведете в една справка класация на тримата най-добре платени мениджъри и тримата най-добре платени служители. Справката трябва да съдържа името и фамилията на всеки от тях, позицията му ("manager" или съответно "employee") и заплатата му да е сортирана по заплатата, в намаляващ ред и после по имената, в азбучен ред.

Пример

first_name	last_name	manager_name
Ken	Sanchez	(no manager)
George	Denchev	(no manager)
Svetlin	Nakov	(no manager)
Martin	Kulov	(no manager)
William	Vong	A. Scott Wright
Sairaj	Uddin	A. Scott Wright
Brian	LaMee	A. Scott Wright
Alan	Brewer	A. Scott Wright
Ranjit	Varkey Chudukatil	Amy Alberts
...	...	...

Решение

```
USE `soft_uni`;
(
    /* managers */
    SELECT `first_name`, `last_name`, 'No manager' as `manager_name`
    FROM `employees`
```

```
    WHERE `manager_id` IS NULL
    ORDER BY `salary` DESC
    LIMIT 3
)
UNION
(
    /* employees */
    SELECT `e`.`first_name`, `e`.`last_name`, concat(`m`.`first_name`,'
','m`.`last_name`) as `manager_name`
    FROM `employees` AS `e`
    LEFT JOIN  `employees` AS `m` ON `m`.`employee_id` = `e`.`manager_id`
    WHERE `e`.`manager_id` IS NOT NULL
    ORDER BY `e`.`salary` DESC
    LIMIT 3
);
```

Задача 4.22. Планините в България

Напишете заявка, която изброява всички планини в България и за всяка - най-високия ѝ връх и неговата височина. Ако липсва информация за някой връх, вместо това да се изведе текста „no info". Заявката да е сортирана по азбучен ред на планините.

Пример

mountain_range	peak_name	elevation
Balkan Mountains	Botev	2376
Pirin	Vihren	2914
Rhodope Mountains	Golyam Perelik	2191
Rila	Musala	2925
Strandza	no	info
Vitosha	no	info

Решение

```
USE `geography`;
(
    /* Планинска верига, връх и височина */
    SELECT `mountains`.`mountain_range`,
    (
        SELECT `peak_name`
        FROM `peaks`
        WHERE `mountains`.`id` = `peaks`.`mountain_id`
        ORDER BY `elevation` DESC
        LIMIT 1
    ) as `peak_name`,
    (
        SELECT `elevation`
```

```
        FROM `peaks`
        WHERE `mountains`.`id` = `peaks`.`mountain_id`
        ORDER BY `elevation` DESC
        LIMIT 1
    ) as `elevation`
    FROM `mountains`
    LEFT JOIN `peaks`  ON `mountains`.`id` = `peaks`.`mountain_id`
    WHERE `mountains`.`id` IN
    (
        SELECT `mountain_id`
        FROM `mountains_countries`
        WHERE `mountains`.`id` = `mountains_countries`.`mountain_id`
AND  `country_code` = "BG"
    )
    AND `peaks`.`peak_name` IS NOT NULL
)
UNION
(
    /* Планинска верига, връх и височина */
    SELECT `mountains`.`mountain_range`, "no" as `peak_name`, "info" as
`elevation`
    FROM `mountains`
    LEFT JOIN `peaks`  ON `mountains`.`id` = `peaks`.`mountain_id`
    WHERE `mountains`.`id` IN
    (
        SELECT `mountain_id`
        FROM `mountains_countries`
        WHERE `mountains`.`id` = `mountains_countries`.`mountain_id`
AND  `country_code` = "BG"
    )
    AND `peaks`.`peak_name` IS NULL
)
```

Задача 4.23. Всички географски обекти в България

Напишете заявка, която изброява всички планини, реки и върхове в България. Заявката да съдържа името на съответния географски обект и вида му („mountain" за планините, "peak" за върховете и "river" за реките) и нещо най-характерно за него (за върховете - височината, за планините - височината на най-високия им връх, за реките - дължината) и да е сортирана по азбучен ред за имената на обектите.

Пример

name	type	info
Balkan Mountains	mountain	2376

Banski Suhodol	peak	2884
Batashki Snezhnik	peak	2082
Botev	peak	2376
Danube	river	2888
Golyam Perelik	peak	2191
Golyam Persenk	peak	2091
Golyam Polezhan	peak	2851
Kamenitsa	peak	2822
...	...	...

Решение

```
USE `geography`;
(
    /* Планински вериги */
    SELECT  `mountain_range` as `name`, 'mountain' as `type`,
    (
       SELECT MAX(`elevation`)
       FROM `peaks`
       WHERE `mountains`.`id` = `peaks`.`mountain_id`
    ) as `info`
    FROM `mountains`
    WHERE `id` IN
    (
        SELECT `mountain_id`
        FROM `mountains_countries`
        WHERE `country_code` = "BG"
    )
)
UNION
(
    /* Върхове */
    SELECT `peak_name` AS `name`, "peak" as `type`, `elevation`
    FROM `peaks`
    WHERE `mountain_id` IN
    (
        SELECT `mountain_id`
        FROM `mountains_countries`
        WHERE `country_code` = "BG"
    )
)
UNION
(
    /* реки */
    SELECT `river_name` AS `name`, "river" AS `type`, `length`
```

```
    FROM `rivers`
    WHERE `id` IN
    (
        SELECT `river_id`
        FROM `countries_rivers`
        WHERE `country_code` = "BG"
    )
)
ORDER BY `name` ASC;
```

Тема 5. Съединения на таблици

Задача 5.1. Адрес на служител

Напишете заявка, която избира:

- employee_id
- job_title
- address_id
- address_text

Върнете първите 5 реда, сортирани по address_id във възходящ ред.

Пример:

employee_id	job_title	address_id	address_text
142	Production Technician	1	108 Lakeside Court
30	Human Resources Manager	2	1341 Prospect St
...	...	...	...

Решение

```
USE `soft_uni`;

/* v1 */
SELECT `employee_id`, `job_title`, `e`.`address_id`, `address_text`
FROM `employees` AS `e`
JOIN `addresses` AS `a` ON `a`.`address_id` = `e`.`address_id`
ORDER BY `e`.`address_id` ASC
LIMIT 5;

/* v2 */
SELECT `employee_id`, `job_title`, `e`.`address_id`, `address_text`
FROM `employees` AS `e`, `addresses` AS `a`
WHERE `a`.`address_id` = `e`.`address_id`
ORDER BY `e`.`address_id` ASC
LIMIT 5;
```

Задача 5.2. Служител по продажбите

Напишете заявка, която избира:

- employee_id
- first_name
- last_name
- department_name

Сортирайте по employee_id в низходящ ред. Изберете само служители от отдел "Продажби".

Пример:

employee_id	first_name	last_name	department_name
290	Lynn	Tsoflias	Sales
289	Rachel	Valdez	Sales
...	...	...	...

Решение

```
USE `soft_uni`;
SELECT `employee_id`, `first_name`, `last_name`, `departments`.`name` AS
`deparment_name`
FROM `employees`
JOIN `departments` ON `departments`.`department_id` = `employees`.`department_id`
WHERE `departments`.`name` = "Sales"
ORDER BY `employees`.`employee_id` DESC;
```

Задача 5.3. Служебни отдели

Напишете заявка, която избира:

- employee_id
- first_name
- salary
- department_name

Филтрирайте само служители със заплата по-голяма от 15000. Върнете първите 5 реда, сортирани по department_id в низходящ ред.

Пример:

employee_id	first_name	salary	department_name
109	Ken	125500.00	Executive
140	Laura	60100.00	Executive
...	...	...	...

Решение

```
USE `soft_uni`;
SELECT `employee_id`, `first_name`, `salary`, `departments`.`name` AS
`deparment_name`
FROM `employees`
JOIN `departments` ON `departments`.`department_id` = `employees`.`department_id`
```

```
WHERE `salary` >= 15000
ORDER BY `departments`.`department_id`  DESC
LIMIT 5
```

Задача 5.4. Служители без проект

Напишете заявка, която избира:

- employee_id
- first_name

Филтрирайте само служители без проект. Върнете първите 3 реда, сортирани по employee_id в низходящ ред.

Пример:

employee_id	first_name
293	George
292	Martin
...	...

Решение

```
USE `soft_uni`;
SELECT `employees`.`employee_id`, `first_name`
FROM `employees`
LEFT JOIN `employees_projects`
ON `employees_projects`.`employee_id` = `employees`.`employee_id`
WHERE `employees_projects`.`project_id` IS NULL
ORDER BY `employees`.`employee_id` DESC
LIMIT 3
```

Задача 5.5. Мениджър на служителите

Напишете заявка, която избира:

- employee_id
- first_name
- manager_id
- manager_name

Филтрирайте всички служители с мениджър, чието ID е равно на 3 или 7. Върнете всички редове, сортирани по първото име на служителя във възходящ ред.

Пример

employee_id	first_name	manager_id	manager_name
122	Bryan	7	JoLynn
158	Dylan	3	Roberto
...	...	...	...

Решение

```
USE `soft_uni`;
SELECT `e`.`employee_id`, `e`.`first_name`, `e`.`manager_id`,
```

```
       concat(`m`.`first_name`," ",`m`.`last_name`) AS `manager_name`
FROM `employees` AS `e`
LEFT JOIN `employees` AS `m` ON `e`.`manager_id` = `m`.`employee_id`
WHERE `e`.`manager_id` IN (3,7)
ORDER BY `e`.`first_name` ASC;
```

Задача 5.6. Минимална заплата

Напишете заявка, която връща стойността на най-ниската заплата от всички отдели и името на отдела, в който се дава.

Пример:

salary	name
9000.0000	Shipping and Receiving

Решение

```
USE `soft_uni`;
SELECT `employees`.`salary`, `departments`.`name`
FROM `employees`, `departments`
WHERE `employees`.`department_id` = `departments`.`department_id` AND
     `employees`.`department_id` =
(
    SELECT `department_id`
    FROM `employees`
    ORDER BY `salary` ASC
    LIMIT 1
)
LIMIT 1
```

Задача 5.7. Държави без планини

Изведете списък на всички държави без нито една планина (общо би трябвало да са 231).

Пример

country_name
Andorra
United Arab Emirates
Afghanistan
...

Решение

```
USE `geography`;

/* v1 */
SELECT DISTINCT `country_name`
FROM `countries` AS `c`
LEFT JOIN `mountains_countries` AS `mc` ON `mc`.`country_code`=`c`.`country_code`
WHERE `mc`.`mountain_id` NOT IN
(
    SELECT `id`
```

```
    FROM `mountains`
    WHERE `mountains`.`id` = `mc`.`mountain_id`
);

/* v2 */
SELECT `country_name`
FROM `countries`
WHERE `country_code` NOT IN
(
    SELECT DISTINCT `country_code`
    FROM `mountains_countries`
);
```

Задача 5.8. Адреси с градове

Напишете заявка, която избира:

- first_name
- last_name
- town
- address_text

Сортирайте по first_name във възходящ ред, после по last_name. Изберете първите 5 служители.

Пример:

first_name	last_name	town	address_text
A.Scott	Wright	Newport Hills	1400 Gate Drive
Alan	Brewer	Kenmore	8192 Seagull Court
...	...	...	...

Решение

```
USE `soft_uni`;
SELECT `e`.`first_name`, `e`.`last_name`, `t`.`name` AS `town`,
`a`.`address_text`
FROM `employees` AS `e`
INNER JOIN `addresses` AS `a` ON `a`.`address_id` = `e`.`address_id`
INNER JOIN `towns` AS `t` ON `t`.`town_id` = `a`.`town_id`
ORDER BY `e`.`first_name`, `e`.`last_name`
LIMIT 5;
```

Задача 5.9. Служители, наети по-късно

Напишете заявка, която избира:

- first_name
- last_name
- hire_date
- dept_name

Филтрирайте само служители, които са наети след 1/1/1999 и са от отделите "Продажби" или "Финанси". Сортирайте по hire_date (възходящо).

Пример:

first_name	last_name	hire_date	dept_name
Debora	Poe	2001-01-19 00:00:00	Finance
Wendy	Kahn	2001-01-26 00:00:00	Finance
...	...	...	...

Решение

```
USE `soft_uni`;
SELECT `e`.`first_name`, `e`.`last_name`, `e`.`hire_date`, `d`.`name` AS
`dept_name`
FROM `employees` AS `e`
INNER JOIN `departments` AS `d` ON
(
    `d`.`department_id` = `e`.`department_id` AND
    `d`.`name` IN ("Sales", "Finance") AND
    `e`.`hire_date` >= 1/1/1999
)
ORDER BY `e`.`hire_date`;
```

Задача 5.10. Служители с проект

Напишете заявка, която избира:

- employee_id
- first_name
- project_name

Филтрирайте само служители с проект, който е започнат след 13.08.2002 и все още продължава (няма крайна дата). Върнете първите 5 реда, сортирани по first_name и после по име на проекта, и двете във възходящ ред.

Пример

employee_id	first_name	project_name
44	A. Scott	Hitch Rack - 4-Bike
170	Alan	LL Touring Handlebars
...	...	...

Решение

```
USE `soft_uni`;
SELECT `e`.`employee_id`, `e`.`first_name`, `p`.`name` AS `project_name`
FROM `employees` AS `e`
INNER JOIN `employees_projects` AS `ep` ON `ep`.`employee_id` = `e`.`employee_id`
INNER JOIN `projects` AS `p` ON
(
      `p`.`project_id` = `ep`.`project_id` AND
     `p`.`start_date` > 13/8/2002 AND
```

```
    `p`.`end_date` IS NULL
)
ORDER BY `e`.`first_name`, `p`.`name`
LIMIT 5;
```

Задача 5.11. Резюме на служителите

Напишете заявка, която избира:

- employee_id
- employee_name
- manager_name
- department_name

Покажете първите 5 служителя (само от тези с мениджър) заедно с техните мениджъри и отделите, в които са (покажете отделите на служителите). Подредете по employee_id.

Пример

employee_id	employee_name	manager_name	department_name
1	Guy Gilbert	Jo Brown	Production
2	Kevin Brown	David Bradley	Marketing
...	...	...	...

Решение

```
USE `soft_uni`;

SELECT `e`.`employee_id`,
  concat(`e`.`first_name`," ",`e`.`last_name`) AS `employee_name`,
  concat(`m`.`first_name`," ",`m`.`last_name`) AS `manager_name`,
  `d`.`name` AS `department_name`

FROM `employees` AS `e`
INNER JOIN `employees` AS `m` ON  `m`.`employee_id` = `e`.`manager_id`
INNER JOIN `departments` AS `d` ON `d`.`department_id` = `e`.`department_id`

ORDER BY `e`.`employee_id`
LIMIt 5;
```

Задача 5.12. Най-високи върхове в България

Напишете заявка, която избира:

- country_code
- mountain_range
- peak_name
- elevation

Филтрирайте всички върхове в България с височина над 2835. Върнете всички редове, сортирани по височина в низходящ ред.

Пример

country_code	mountain_range	peak_name	elevation
BG	Rila	Musala	2925
BG	Pirin	Vihren	2914
...	...	...	...

Решение

```
USE `geography`;

SELECT `mc`.`country_code`, `m`.`mountain_range`, `p`.`peak_name`,
`p`.`elevation`
FROM `peaks` AS `p`
INNER JOIN `mountains` AS `m` ON `m`.`id` = `p`.`mountain_id`
INNER JOIN `mountains_countries` AS `mc` ON
(
    `mc`.`mountain_id` = `p`.`mountain_id` AND
    `mc`.`country_code` = "BG"
)
WHERE `p`.`elevation` >= 2835
ORDER BY `p`.`elevation` DESC;
```

Задача 5.13. Планински вериги

Напишете заявка, която избира:

- country_code
- country_name
- mountain_range

Филтрирайте планинските вериги в Съединените щати, Русия и България. Сортирайте резултата по country_code и mountain_range, в азбучен ред.

Пример

country_code	country_name	mountain_range
BG	Bulgaria	Balkan Mountains
BG	Bulgaria	Pirin
BG	Bulgaria	Rhodope Mountains
BG	Bulgaria	Rila
BG	Bulgaria	Strandza
BG	Bulgaria	Vitosha
RU	Russia	Caucasus
US	United States	Alaska Range

Решение

```
USE `geography`;

SELECT `mc`.`country_code`, `c`.`country_name`, `m`.`mountain_range`
FROM `mountains_countries` AS `mc`

INNER JOIN `countries` AS `c`  ON
```

```
(
    `mc`.`country_code` = `c`.`country_code` AND
    `mc`.`country_code` IN ("BG", "RU", "US")
)
INNER JOIN `mountains` AS `m` ON `mc`.`mountain_id` = `m`.`id`
ORDER BY `mc`.`country_code`, `m`.`mountain_range`;
```

Задача 5.14. Държави с реки

Напишете заявка, която избира:

- country_name
- river_name

Намерете първите 5 държави с или без реки в Африка. Сортирайте ги по country_name във възходящ ред.

Пример

country_name	river_name
Algeria	Niger
Angola	Congo
Benin	Niger
Botswana	NULL
Burkina Faso	Niger

Решение

```
USE `geography`;

SELECT `c`.`country_name`,  `r`.`river_name`
FROM `countries_rivers` AS `cr`
LEFT JOIN `rivers` AS `r` ON `r`.`id` = `cr`.`river_id`
LEFT JOIN `countries` AS `c` ON `c`.`country_code` = `cr`.`country_code`
WHERE `c`.`continent_code` = "AF"
ORDER BY `c`.`country_name`
LIMIT 5;
```

Задача 5.15.* Държави без планини

Изведете списък на всички държави без нито една планина (общо би трябвало да са 231). Използвайте за целта OUTER JOIN.

Пример

country_name
Andorra
United Arab Emirates
Afghanistan
...

Решение

```
USE `geography`;
```

```
SELECT `country_name`
FROM `countries` AS `c`
LEFT JOIN `mountains_countries` AS `mc`
      ON `mc`.`country_code` = `c`.`country_code`
WHERE `mc`.`mountain_id` IS NULL;
```

Задача 5.16. Планините в България

Напишете заявка, която изброява планини в България и за всяка - най-високия ѝ връх и неговата височина. Заявката да е сортирана по височината на планините, от най-високата към най-ниската. За целта използвайте OUTER JOIN.

Пример

mountain_range	peak_name	elevation
Rila	Musala	2925
Pirin	Vihren	2914
Balkan Mountains	Botev	2376
Rhodope Mountains	Golyam Perelik	2191
Strandza	NULL	NULL
Vitosha	NULL	NULL

Решение

```
USE `geography`;

SELECT `m`.`mountain_range`,`p`.`peak_name`, `p`.`elevation`
FROM `peaks` AS `p`
LEFT JOIN `mountains` AS `m` ON `m`.`id` = `p`.`mountain_id`
LEFT JOIN `mountains_countries` AS `mc` ON `mc`.`mountain_id`=`m`.`id`

WHERE  `mc`.`country_code`  = "BG"
ORDER BY `p`.`elevation` DESC;
```

Задача 5.17. Служители без проект

Напишете заявка, която избира:

- employee_id
- first_name
- last_name

Филтрирайте само служители, които нямат назначен проект, по който да работят. Върнете първите 3 реда, сортирани по first_name и после по last_name, и двете във възходящ ред.

Пример

employee_id	first_name	last_name
284	Amy	Alberts
264	Annette	Hill

198	Arvind	Rao

Решение

```
USE `soft_uni`;

SELECT `e`.`employee_id`, `first_name`, `last_name`
FROM `employees` AS `e`
LEFT JOIN `employees_projects` AS `ep`
       ON `ep`.`employee_id` = `e`.`employee_id`
WHERE `ep`.`project_id` IS NULL
ORDER BY `first_name`, `last_name`
LIMIT 3;
```

Задача 5.18. Служител 24

Напишете заявка, която избира:

- employee_id
- first_name
- project_name

Филтрирайте всички проекти служителя с id 24. Ако проектът е бил започнат след 2005 включително, върнатата стойност трябва да е NULL (използвайте оператор CASE за целта). Сортирайте резултата по име на проекта, по азбучен ред.

Пример

employee_id	first_name	project_name
24	David	NULL
24	David	NULL
...	...	...

Решение

```
USE `soft_uni`;

SELECT `e`.`employee_id`, `first_name`,
CASE
    WHEN `p`.`start_date` >= 1/1/2005 THEN NULL
    ELSE `p`.`start_date`
END AS `start_date`

FROM `employees` AS `e`
JOIN `employees_projects` AS `ep` ON `ep`.`employee_id` = `e`.`employee_id`
JOIN `projects` AS `p` ON `p`.`project_id`=`ep`.`project_id`

WHERE `e`.`employee_id` = 24;
```

Задача 5.18. Игра на континенти

За да научи своите ученици да намират континентите на глобуса, една ваша позната учителка им е измислила интересна игра: организира им виртуално междуконтинентално пътуване. За целта дава на всяко дете „билетче“, на което пише от кой континент ще излети и в кой континент ще кацне самолета му и то трябва да ги посочи на глобуса. Понеже вашата позната не разбира от програмиране, а е сигурна, че има лесен начин да се направят тези билетчета, е възложила задачата на вас. Моля, подсигурете се, че билетите ще обхванат всички възможни маршрути! Направете заявка, от която да могат да се отпечатат билетите. Сортирайте имената на континентите по азбучен ред - и по колоната FROM, от където започва „пътуването“, и по колоната TO, указваща крайната му точка.

Пример

FROM	TO
Africa	Africa
Africa	Antarctica
Africa	Asia
Africa	Europe
Africa	North America
Africa	Oceania
Africa	South America
Antarctica	Africa
Antarctica	Antarctica
Antarctica	Asia
Antarctica	Europe
...	...

Решение

```
USE `geography`;

SELECT `c1`.`continent_name` AS `FROM`,
       `c2`.`continent_name` AS `TO`
FROM `continents` AS `c1`
CROSS JOIN `continents` AS `c2`

ORDER BY `c1`.`continent_name`, `c2`.`continent_name`;
```

Задача 5.20. Европейското по футбол

Предстои нов тип европейското първенство по футбол. За да има повече футболни мачове и емоции и за да е по-оспорвано първенството, е решено абсолютно всяка европейска страна да играе срещу всяка друга. На вас се пада нелеката задача да подготвите таблата за срещите. Те трябва да съдържат града, в който ще е срещата (това винаги е столицата на страната домакин), име на страна домакин, място за отбелязани голове за домакина и друго за отбелязани голове на страната-гост и името ѝ. Тази таблица трябва да се попълни с имената на всички европейски страни,

разбъркани в случаен ред (ползвайте за целта ORDER BY RAND()). Подсигурете се, че всяка страна ще играе и като гост, и като домакин с всяка друга, но няма да се налага да играе със себе си. :-)

Пример

Place	Player 1	Host	Guest	Player 2
Ljubljana	Slovenia			Romania
Bratislava	Slovakia			Isle of Man
Stockholm	Sweden			Belgium
Brussels	Belgium			Albania
Tórshavn	Faroe Islands			Belarus
Pristina	Kosovo			France
...	...	...	...	...

Решение

```
USE `geography`;

SELECT `c1`.`capital` AS `Place`,
       `c1`.`country_name` AS `Player 1`,
       NULL AS `Host`,
       NULL AS `Guest`,
       `c2`.`country_name` AS `Player 2`
FROM `countries` AS `c1`
CROSS JOIN `countries` AS `c2`

WHERE `c1`.`continent_code` = "EU" AND
      `c2`.`continent_code` = "EU"

ORDER BY RAND();
```

Задача 5.21. Най-висок връх и най-дълга река по държава

За всяка държава, намерете височината на най-високия връх и дължината на най-дългата река, сортирани по височина на най-високия връх (от най-висок към най-нисък), после по дължина на най-дългата река (от най-дълга до най-къса), после по име на държавата (по азбучен ред). Покажете NULL когато за някоя от колоните не са намерени данни. Ограничете се само до първите 5 реда.

country_name	highest_peak_elevation	longest_river_length
China	8848	6300
India	8848	3180
Nepal	8848	2948
Pakistan	8611	3180
Argentina	6962	4880
...	...	...

Решение

```
USE `geography`;
```

```
SELECT  `c`.`country_name`,
        MAX(`p`.`elevation`) AS 'highest_peak_elevation',
        MAX(`r`.`length`) AS 'longest_river_length'

FROM `countries` AS `c`

LEFT JOIN `mountains_countries` AS `mc` ON `mc`.`country_code` =
`c`.`country_code`
LEFT JOIN `peaks` AS `p` ON `p`.`mountain_id` = `mc`.`mountain_id`
LEFT JOIN `countries_rivers` AS `cr` ON `cr`.`country_code` = `c`.`country_code`
LEFT JOIN `rivers` AS `r` ON `r`.`id` = `cr`.`river_id`

GROUP BY `c`.`country_name`
ORDER BY `highest_peak_elevation` DESC, `longest_river_length` DESC,
`c`.`country_name` ASC

LIMIT 5;
```

Задача 5.22. *Континенти и валути

Напишете заявка, която избира:

- continent_code
- currency_code
- currency_usage

Намерете всички континенти и най-използваната им валута. Филтрирайте всички валути, които се използват само в една държава. Сортирайте резултатите по continent_code и currency_code.

Пример

continent_code	currency_code	currency_usage
AF	XOF	8
AS	AUD	2
AS	ILS	2
EU	EUR	26
NA	XCD	8
OC	USD	8

Решение

```
USE `geography`;

SELECT `usages`.`continent_code`, `usages`.`currency_code`, `usages`.`usages`
FROM
(
```

```
    SELECT `con`.`continent_code`, `cu`.`currency_code`,
COUNT(`cu`.`currency_code`) AS `usages`
    FROM `continents` AS `con`
    INNER JOIN `countries` AS `c` ON `c`.`continent_code` =
`con`.`continent_code`
    INNER JOIN `currencies` AS `cu` ON `cu`.`currency_code` = `c`.`currency_code`
    GROUP BY `con`.`continent_code`, `cu`.`currency_code`
) AS `usages`
INNER JOIN
(
    SELECT `usages`.`continent_code`, MAX(`usages`.`usages`) AS `maxUsage`
    FROM
    (
        SELECT `con`.`continent_code`, `cu`.`currency_code`,
COUNT(`cu`.`currency_code`) AS `usages`
        FROM `continents` AS `con`
        INNER JOIN `countries` AS `c` ON `c`.`continent_code` =
`con`.`continent_code`
        INNER JOIN `currencies` AS `cu` ON `cu`.`currency_code` =
`c`.`currency_code`
        GROUP BY `con`.`continent_code`, `cu`.`currency_code`
        HAVING COUNT(`cu`.`currency_code`) > 1
    ) as `usages`
    GROUP BY `usages`.`continent_code`
) AS `max_usages`
ON `max_usages`.`continent_code` = `usages`.`continent_code` AND
`max_usages`.`maxUsage` = `usages`.`usages`

ORDER BY `usages`.`continent_code`, `usages`.`currency_code`;
```

Тема 6. Агрегация и групиране на данни

Задача 6.1. Брой на записите

Импортирайте базата данни Gringotts и изведете общият брой записи. Уверете се, че нищо не остава скрито-покрито.

Пример:

Count
162

Решение

```
USE `gringotts`;
SELECT COUNT(*) AS `Count`
FROM `wizzard_deposits`;
```

Задача 6.2. Най-дългата магическа пръчка

Изберете размера на най-дългата магическа пръчка. Преименувайте новата колона по подходящ начин.

Пример:

longest_magic_wand
31

Решение

```
USE `gringotts`;
SELECT MAX(`magic_wand_size`) AS `longest_magic_wand`
FROM `wizzard_deposits`;
```

Задача 6.3. Най-дългата магическа пръчка по депозитна група

Покажете дължината на най-дългата магическа пръчка за всяка от депозитните групи. Сортирайте резултатите по най-дълга магическа пръчка за всяка депозитна група в нарастващ ред, а след това и по deposit_group в азбучен ред. Именувайте новата колона по подходящ начин.

Пример:

deposit_group	longest_magic_wand
Human Pride	30
…	…

Решение

```
USE `gringotts`;
SELECT `deposit_group`, MAX(`magic_wand_size`) AS `longest_magic_wand`
FROM `wizzard_deposits` AS `w`
GROUP BY `deposit_group`
ORDER BY `deposit_group`;
```

Задача 6.4. Най-малката депозитна група с най-малката магическа пръчка

Изберете депозитната група, с най-малката средноаритметична стойност от размера на пръчките си.

Пример:

deposit_group
Troll Chest

Решение

```
USE `gringotts`;
SELECT `avg`.`deposit_group`
FROM
(
    SELECT `deposit_group`, AVG(`magic_wand_size`) AS `avg_magic_wand`
    FROM `wizzard_deposits` AS `w`
    GROUP BY `deposit_group`
    ORDER BY `avg_magic_wand` ASC
```

```
    LIMIT 1
) AS `avg`;
```

Задача 6.5. Сума от депозити

Изберете всички депозитни групи и тяхната обща депозитна сума. Сортирайте резултатите по total_sum в нарастващ ред.

Пример:

deposit_group	total_sum
Blue Phoenix	819598.73
...	...

Решение

```
USE `gringotts`;
SELECT `deposit_group`,
       SUM(`deposit_amount`) AS `total_sum`
FROM `wizzard_deposits`
GROUP BY `deposit_group`
ORDER BY `total_sum` ASC;
```

Задача 6.6. Депозитни суми за семейство Ollivander

Изберете всички депозитни групи и общата депозитна сума, но само за тези магьосници, чиято пръчка е измайсторена от семейство Ollivander. Сортирате резултатите по deposit_group в азбучен ред.

Пример:

deposit_group	total_sum
Blue Phoenix	52968.96
Human Pride	188366.86
...	...

Решение

```
USE `gringotts`;
SELECT `deposit_group`,
       SUM(`deposit_amount`) AS `total_sum`
FROM `wizzard_deposits` AS `w`
WHERE `magic_wand_creator` = "Ollivander family"
GROUP BY `deposit_group`
ORDER BY `deposit_group` ASC;
```

Задача 6.7. Филтър на депозити

Изберете всички депозитни групи и общата депозитна сума, но само за тези магьосници, чиято пръчка е измайсторена от семейство Ollivander. След това филтрирайте общата депозитна сума, така че да показва само тези под 150000. Подредете резултатите по общата сума в намалящ ред.

Пример:

deposit_group	total_sum

Troll Chest	126585.18
...	...

Решение

```
USE `gringotts`;
SELECT `deposit_group`,
       SUM(`deposit_amount`) AS `total_sum`
FROM `wizzard_deposits` AS `w`
WHERE `magic_wand_creator` = "Ollivander family"
GROUP BY `deposit_group`
HAVING `total_sum` < 150000
ORDER BY `total_sum` DESC;
```

Задача 6.8. Минимално зареждане на депозит

Създайте заявка, която извлича следната информация:

- Депозитна група
- Майстор на магическата пръчка
- Минимална сума за депозит за всяка група

Подредете информацията в нарастващ ред по magic_wand_creator и deposit_group.

Пример:

deposit_group	magic_wand_creator	min_deposit_charge
Blue Phoenix	Antioch Peverell	30.00
...	...	

Решение

```
USE `gringotts`;
SELECT `deposit_group`, `magic_wand_creator`,
       min(`deposit_charge`) AS `min_deposit_charge`
FROM `wizzard_deposits`
GROUP BY `deposit_group`
ORDER BY `magic_wand_creator`, `deposit_group`;
```

Задача 6.9. Възрастови групи

Напишете заявка, която създава 7 различни групи според тяхната възраст.

Възрастовите групи трябва да са както следва:

- [0-10]
- [11-20]
- [21-30]
- [31-40]
- [41-50]
- [51-60]
- [61+]

Заявката трябва да връща

- Възрастовите групи
- Броят на магьосниците в тях

Сортирайте резултатите по нарастващ ред според размера на всяка група.

Пример:

age_group	wizard_count
[11-20]	21
...	...

Решение

```
USE `gringotts`;
(
    select "[0-10]" as `age_group`, count(*) as `wizard_count`
    from `wizzard_deposits`
    where `age` between 0 and 10
)
union
(
    select "[11-20]" as `age_group`, count(*) as `wizard_count`
    from `wizzard_deposits`
    where `age` between 11 and 20
)
union
(
    select "[21-30]" as `age_group`, count(*) as `wizard_count`
    from `wizzard_deposits`
    where `age` between 21 and 30
)
union
(
    select "[31-40]" as `age_group`, count(*) as `wizard_count`
    from `wizzard_deposits`
    where `age` between 31 and 40
)
union
(
    select "[41-50]" as `age_group`, count(*) as `wizard_count`
    from `wizzard_deposits`
    where `age` between 41 and 50
)
union
```

```
(
    select "[51-60]" as `age_group`, count(*) as `wizard_count`
    from `wizzard_deposits`
    where `age` between 51 and 60
)
union
(
    select "[61+]" as `age_group`, count(*) as `wizard_count`
    from `wizzard_deposits`
    where `age` > 60
);
```

Задача 6.10. Първа буква

Напишете заявка, която връща всички уникални първи букви от първите имена на магьосници, които имат депозит от тип Troll Chest. Подредете ги в азбучен ред. Използвайте GROUP BY за уникалност.

Пример:

first_letter
A
...

Решение

```
USE `gringotts`;
SELECT LEFT(`first_name`, 1) AS `first_letter`
FROM `wizzard_deposits`
WHERE `deposit_group` = 'Troll Chest'
group by `first_letter`
ORDER BY `first_letter`;
```

Задача 6.11. Средна лихва

Господин Бодрог много се интересува от доходността. Той иска да знае средната лихва на всички депозитни групи разделени според това дали депозита е изтекъл или не. Но това не е всичко. Той иска да изберете депозитите със стартова дата след 01/01/1985. Подредете информацията в намаляващ ред по депозитна група и в нарастващ ред по изтичане.

Изходът трябва да се състои от следните колони:

Пример

deposit_group	is_deposit_expired	average_interest
Venomous Tongue	0	16.698947
...	...	

Решение

```
USE `gringotts`;
SELECT `deposit_group`, `is_deposit_expired`,
AVG(`deposit_interest`) AS `average_interest`
```

```
FROM `wizzard_deposits`
WHERE `deposit_start_date` >= 01/01/1985
GROUP BY `deposit_group`, `is_deposit_expired`
ORDER BY `deposit_group` DESC, `is_deposit_expired` ASC;
```

Тема 7. Скаларни функции, работа с дати, транзакции

Задача 7.1. Служители със заплата над 35000

Създайте съхранена процедура **usp_get_employees_salary_above_35000** която връща първото и последното име на всички служители, които имат заплата над 35000. Резултатът трябва да бъде сортиран по first_name, а след това и по last_name в азбучен ред.

Пример

first_name	last_name
Amy	Alberts
Brian	Welcker
Dan	Wilson
...	...

Решение

```
DELIMITER $$
CREATE PROCEDURE usp_get_employees_salary_above_35000()
BEGIN
    SELECT first_name, last_name
    FROM employees
    WHERE salary>35000;
END
$$
```

Задача 7.2. Служители със заплата над...

Създайте съхранена процедура **usp_get_employees_salary_above**, която приема число като параметър и връща първото и последното име на всички служители, които имат заплата равна на поне даденото число. Резултатът трябва да се сортира по first_name, а след това и по last_name в азбучен ред.

Пример

Числото в този пример е 48100.

first_name	last_name
Amy	Alberts
Brian	Welcker
Dylan	Miller
...	...

Решение

```
DELIMITER $$
CREATE PROCEDURE usp_get_employees_salary_above(salaryVal INT)
```

```
BEGIN
    SELECT first_name, last_name
    FROM employees
    WHERE salary>salaryVal;
END
$$
```

Задача 7.3. Градове започващи със...

Създайте съхранена процедура **usp_get_towns_starting_with** , която приема низ като параметър и връща всички имена на градове започващи с този низ. Резултатът трябва да бъде сортиран по името на града по азбучен ред.

Пример

Това е списъкът с всички градове започващи с "b".

town_name
Bellevue
Berlin
Bordeaux
Bothell

Решение

```
DELIMITER $$
CREATE PROCEDURE usp_get_towns_starting_with (start_letter TEXT)
BEGIN
    SELECT  `name`
    FROM towns
    WHERE LOWER(SUBSTRING(`name`,1, LENGTH(start_letter)))=LOWER(start_letter);
END
$$
```

Задача 7.4. Служители от град

Напишете съхранена процедура **usp_get_employees_from_town** която приема името на град като параметър и връща първото и последното име на всички служители, които живеят в дадения град. Резултатът трябва да бъде сортиран по first_name, а след това по last_name в азбучен ред.

Пример

Ето списък на служителите живеещи в Sofia.

first_name	last_name
George	Denchev
Martin	Kulov
Svetlin	Nakov

Решение

```
DELIMITER $$
CREATE PROCEDURE usp_get_employees_from_town (town_name TEXT)
```

```
BEGIN
    SELECT first_name, last_name
    FROM employees
    WHERE address_id = (SELECT id FROM  towns WHERE `name`=town_name LIMIT 1);
END
$$
```

Задача 7.5. Функция за ниво на заплата

Напишете функция **ufn_get_salary_level**, която получава заплата на служител и връща нивото на заплатата.

- Ако заплатата е < 30000 връща "Low"
- Ако заплатата е между 30000 и 50000 (вкл) връща "Average"
- Ако заплатата е > 50000 връща "High"

Пример

salary	salary_Level
13500.00	Low
43300.00	Average
125500.00	High

Решение

Задача 7.6. Дефинирайте функция

Дефинирайте функция **ufn_is_word_comprised**(set_of_letters, word), която връща true или false според това дали думата е съставена от даденото множество от букви.

Пример

set_of_letters	word	result
Oistmiahf	Sofia	1
Oistmiahf	halves	0
Bobr	Rob	1
Pppp	Guy	0

Решение

```
DELIMITER $$
CREATE FUNCTION ufn_is_word_comprised(set_of_letters TEXT, word TEXT)
RETURNS TEXT
DETERMINISTIC
BEGIN
      DECLARE a INT Default 0 ;
      simple_loop: LOOP
         SET a=a+1;
         IF (LOCATE(LOWER(SUBSTRING(`word`,a,1)),LOWER(set_of_letters)) <= 0)
THEN
```

```
        RETURN FALSE;
        END IF;
        IF a=length(word)-1 THEN
           LEAVE simple_loop;
        END IF;
   END LOOP simple_loop;
   RETURN TRUE;
END $$
```

Задача 7.7. Изтегляне на пари

Напишете съхранена процедура **usp_withdraw_money** (account_id, money_amount), която работи чрез транзакции.

Уверете се, че изтеглянето е приключило само ако балансът е достатъчен и money_amount е валидно положително число. Работете с точност до 4-тия знак след десетичната запетая.

Пример

Това е резултатът при account_id = 1 и money_amount = 10.

account_id	account_holder_id	balance
1	1	113.1200

Решение

Задача 7.8. Банков превод

Напишете съхранена процедура **usp_transfer_money**(from_account_id, to_account_id, amount), която трансферира пари от една сметка към друга. Разгледайте случаите, когато едно от account_id-тата е невалидно, сумата от пари е отрицателно число, балансът не е достатъчен или трансферирате от/към един и същ акаунт. Уверете се, че цялата процедура минава без грешки и ако се случи грешка, то това не изменя базата данни.

Пример

from_account_id = 1, to_account_id = 2 и money_amount = 10.

account_id	account_holder_id	balance
1	1	113.1200
2	3	4364.2300

Решение

Задача 7.9. Логове за банкови сметки

Създайте нова таблица – logs (log_id, account_id, old_sum, new_sum). Добавете тригер към таблицата accounts, който да добавя нов запис в logs всеки път, когато има промяна по банкова сметка.

Пример

log_id	account_id	old_sum	new_sum
1	1	123.12	113.12
...	...	...	...

Решение

Задача 7.10. Тригер за мейли

Създайте нова таблица – **notification_emails**(id, recipient, subject, body). Добавете тригер към таблицата с логовете от предната задача, за да създадете нов мейл всеки път, когато бъде добавен запис в logs таблицата. Следната информация е нужна за всеки мейл:

- recipient – account_id
- subject – "Balance change for account: {account_id}"
- body - "On {date} your balance was changed from {old} to {new}."

Пример

id	recipient	subject	body
1	1	Balance change for account: 1	On Sep 15 2016 at 11:44:06 AM your balance was changed from 133 to 143.
...	...	...	...

Решение

Съдържание