**Курсовая работа**

**по дисциплине**

**«Базы данных»**

Выполнил:

Солонин Егор Витальевич

Группа А-14-19

Преподаватель:

Титов Д.А.

**Часть 1. Изучение предметной области.**

Предметная область: email-рассылка определенной аудитории докторов. В нашей базе данных хранится информация о докторах.

Данные:

* Фамилия
* Имя
* Отчество
* E-mail
* Телефон
* Специальность
* Организация
* Регион
* Стаж работы

Операции:

* фильтр по специальности
* фильтр по организации
* фильтр по региону

Специальность представляет из себя сущность, состоящую из узконаправленной специальности и более общей. Например, стоматолог-хирург (стоматолог); травматолог-ортопед (хирург), аритмолог (кардиолог).

Организация – название организации, где врач осуществляет свою деятельность. Бывает нескольких типов: государственная поликлиника, частная клиника, больница, самозанятый.

Описание структур таблиц:

Основная таблица докторов:

|  |  |  |  |
| --- | --- | --- | --- |
| **Имя поля** | **Тип данных** | **Размер данных** | **Ключевое** |
| ID | числовой | длинное целое | да |
| Фамилия | строка | 255 | нет |
| Имя | строка | 255 | нет |
| Отчество | строка | 255 | нет |
| Email | строка | 255 | нет |
| Телефон | строка | 255 | нет |
| Специальность | строка | 255 | нет |
| Организация | строка | 255 | нет |
| Регион | строка | 255 | нет |
| Стаж работы | число | 2 | нет |

Таблица узких специальностей:

|  |  |  |  |
| --- | --- | --- | --- |
| **Имя поля** | **Тип данных** | **Размер данных** | **Ключевое** |
| ID специальности | числовой | длинное целое | да |
| Название специальности | строка | 255 | нет |
| ID общей специальности | числовой | длинное целое | нет |

Таблица общих специальностей:

|  |  |  |  |
| --- | --- | --- | --- |
| **Имя поля** | **Тип данных** | **Размер данных** | **Ключевое** |
| ID специальности | числовой | длинное целое | да |
| Название специальности | строка | 255 | нет |

Таблица организаций:

|  |  |  |  |
| --- | --- | --- | --- |
| **Имя поля** | **Тип данных** | **Размер данных** | **Ключевое** |
| ID организации | числовой | длинное целое | да |
| Название организации | строка | 255 | нет |
| Тип | числовой | длинное целое | нет |

Таблица типов организаций:

|  |  |  |  |
| --- | --- | --- | --- |
| **Имя поля** | **Тип данных** | **Размер данных** | **Ключевое** |
| ID типа | числовой | длинное целое | да |
| Название типа | короткий текст | 255 | нет |

Таблица регионов:

|  |  |  |  |
| --- | --- | --- | --- |
| **Имя поля** | **Тип данных** | **Размер данных** | **Ключевое** |
| ID региона | числовой | длинное целое | да |
| Название региона | строка | 255 | нет |

**Часть 2. Инфологическая модель.**

**Доктор**

Фамилия

Имя

Отчество

Специальность

Email

Телефон

Организация

Регион

Стаж

**Узкая спец-ть**

ID узкой спец-ти

Название

ID общей спец-ти

1

М

1

**Общая спец-ть**

ID

Название

М

**Организация**

ID

Название

ID типа

1

М

1

**Тип организации**

ID

Название

1

М

**Регион**

ID

Название

**Часть 3. Реляционная модель.**

Specialties

ID

Name

Sub-specialties

ID

Name

specialtyID

Doctors

ID

SubSpecialtyID

FirstName

LastName

MiddleName

Email

Phone

OrganizationID

RegionID

Experience

Organization Types

ID

Name

Organizations

ID

Name

orgTypeID

Regions

ID

Name

**Часть 4. Нормализация.**

**Первая нормальная форма (1НФ)**

Переменная отношения находится в первой нормальной форме тогда и только тогда, когда в любом допустимом значении этой переменной каждый кортеж отношения содержит только одно значение для каждого из атрибутов.

**Вторая нормальная форма (2НФ)**

Переменная отношения находится во второй нормальной форме тогда и только тогда, когда она находится в первой нормальной форме и каждый неключевой атрибут **неприводимо** (функционально полно) зависит от её потенциального ключа.

Функционально полная зависимость означает, что если потенциальный ключ является составным, то атрибут зависит от всего ключа и не зависит от его частей.

Вторая нормальная форма по определению запрещает наличие неключевых атрибутов, которые вообще не зависят от потенциального ключа.

**Третья нормальная форма (3НФ)**

Переменная отношения находится в третьей нормальной форме тогда и только тогда, когда она находится во второй нормальной форме, и отсутствуют транзитивные функциональные зависимости неключевых атрибутов от ключевых.

… «каждый атрибут должен предоставлять информацию о ключе, полном ключе и ни о чём, кроме ключа»

Таблицы в 3НФ (жирным шрифтом выделен ключ):

Таблица Doctors

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** | SpecialtyID | OrgID | RegionID | FirstName | LastName | MiddleName | Email | Phone | Experience |

Таблица Sub-specialties

|  |  |  |
| --- | --- | --- |
| **ID** | Name | SpecialtyID |

Таблица Specialties

|  |  |
| --- | --- |
| **ID** | Name |

Таблица Organizations

|  |  |  |
| --- | --- | --- |
| **ID** | Name | OrgTypeID |

Таблица Organization Types

|  |  |
| --- | --- |
| **ID** | Name |

Таблица Regions

|  |  |
| --- | --- |
| **ID** | Name |

Работа с базой данных SQL реализована с помощью платформы nodeJS. Используется библиотека mysql для подключения к БД. Результаты выборок выводятся в таблицы csv. Ссылка на реализованные процедуры:

<https://github.com/DerFunkeLeben/DB_CW>

Таблица докторов:

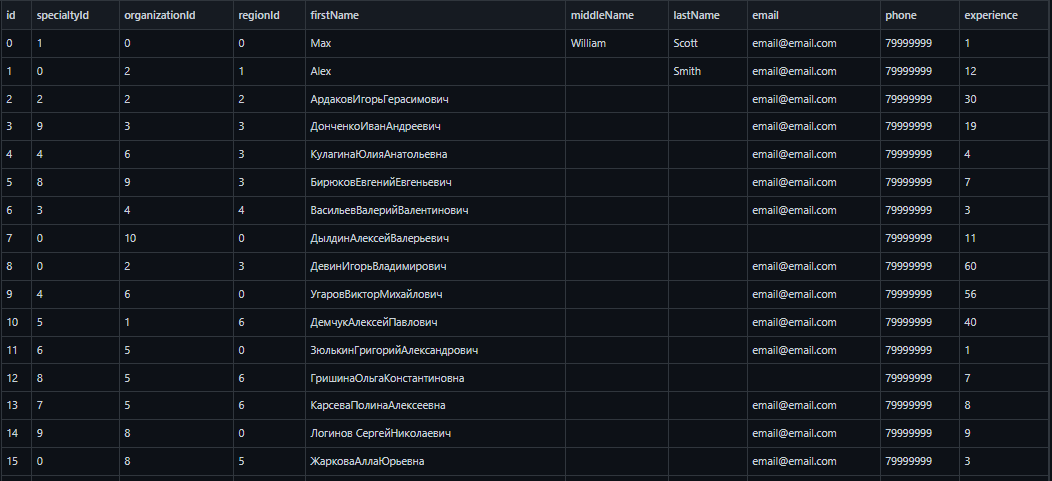


Таблица организаций:

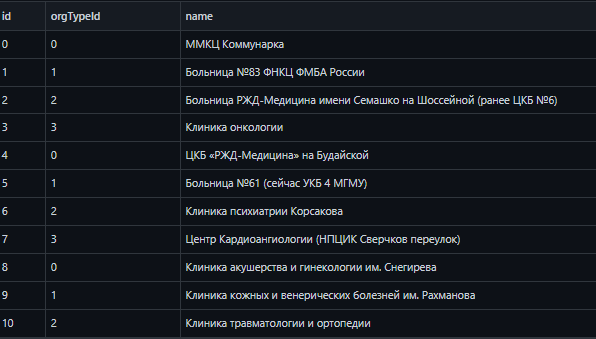


Таблица типов организаций:

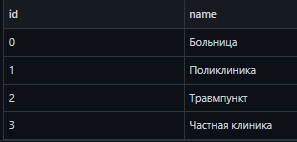


Таблица регионов:

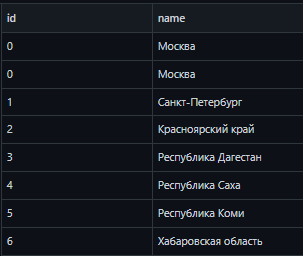
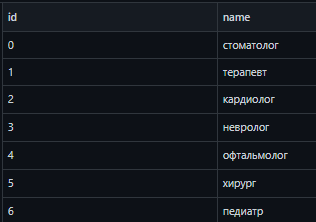


Таблица узких специальностей:



Таблица специальностей:



**Часть 5. Выборка данных**

**1. Список врачей указанной специальности, с непустым email.**

const selectBySpecialty = specialty => `

    SELECT

        doctors.firstName AS name,

        doctors.email AS email,

        subSpecialties.name AS specialty

    FROM doctors

    INNER JOIN subSpecialties

    ON ( doctors.specialtyId = subSpecialties.id )

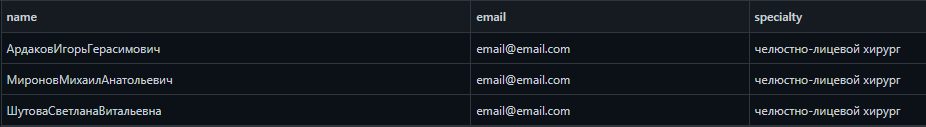
    WHERE subSpecialties.name = '${specialty}'

    AND doctors.email != ''

`

selectBySpecialty('челюстно-лицевой хирург')

Полученный результат:



**2. Список врачей со стажем работы более N лет, отсортированных по убыванию стажа.**

const selectOnExperience = exp => `

    SELECT

        doctors.firstName AS name,

        doctors.email AS email,

        doctors.experience AS experience

    FROM doctors

    WHERE doctors.experience > '${exp}'

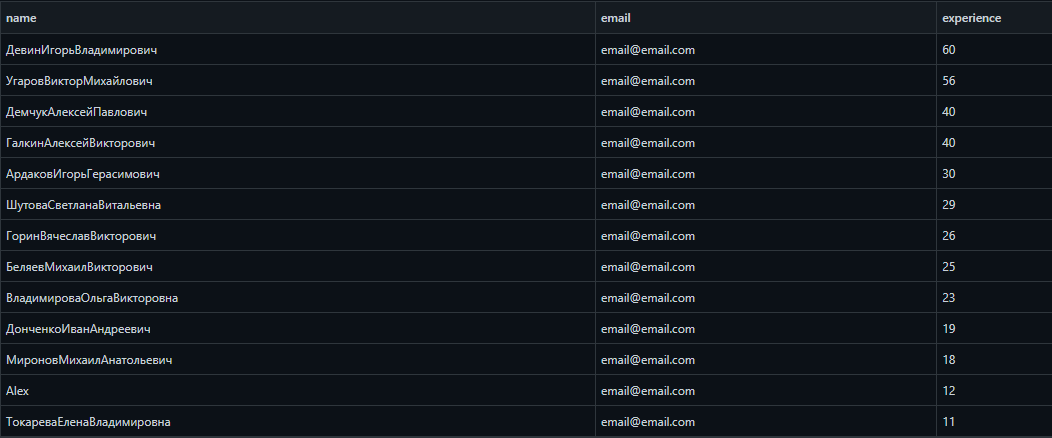
    AND doctors.email != ''

    ORDER BY experience DESC

`

selectOnExperience(10)

Полученный результат:



**3. Список врачей, работающих в поликлиниках Москвы.**

const selectPolyclinicMSK = () => `

    SELECT

        doctors.firstName AS name,

        organizations.name AS orgName,

        regions.name AS region

    FROM doctors

    RIGHT JOIN organizations

    ON doctors.organizationId = organizations.id

    RIGHT JOIN organizationTypes

    ON organizations.orgTypeId = organizationTypes.id

    RIGHT JOIN regions

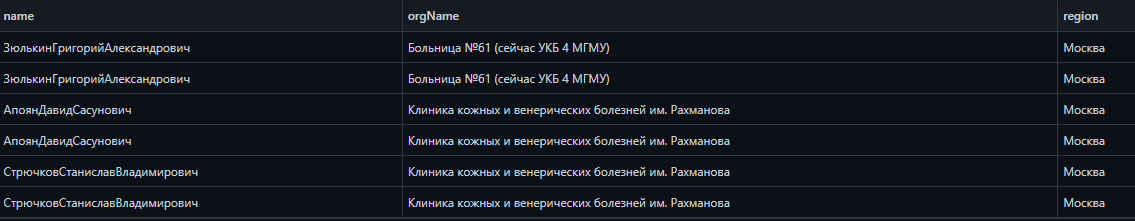
    ON regions.id = doctors.regionId

    WHERE regions.name = 'Москва'

    AND organizationTypes.name = 'Поликлиника'

`

Полученный результат:



**4. Список организаций, в которых нет врачей заданной специальности**

const selectOrgsFilteredBySpec = specialty => `

    SELECT organizations.name AS orgName FROM organizations

    WHERE organizations.id NOT IN (

        SELECT

            organizations.id AS orgId

        FROM organizations

        INNER JOIN doctors

        ON doctors.organizationId = organizations.id

        INNER JOIN subSpecialties

        ON doctors.specialtyId = subSpecialties.id

        WHERE subSpecialties.name = '${specialty}'

        GROUP BY orgId

    )

`

selectOrgsFilteredBySpec('стоматолог-хирург')

Полученный результат:



**5. Список врачей из выбранного региона.**

const selectOnRegion = region => `

    SELECT

        doctors.firstName AS name,

        regions.name AS region

    FROM doctors

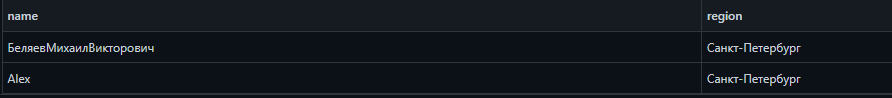
    RIGHT JOIN regions

    ON regions.id = doctors.regionId

    WHERE regions.name = '${region}'

`

selectOnRegion('Санкт-Петербург')

Полученный результат:

**Часть 6. Вставка, обновление и удаление данных**

**Вставка данных**.

***1. Добавление новой специальности***

INSERT INTO specialties

(id, name) VALUES

(7, 'гинеколог');

***2. Добавление нового врача из Красноярского края.***

const insertDocKrasnoyarsk = `

    INSERT INTO doctors

    (id, specialtyId, organizationId, regionId, firstName, middleName, lastName, email, phone, experience)

    SELECT

        100,

        1,

        1,

        region.id,

        'Виктор',

        'Андреевич',

        'Смирнов',

        'a@a.ru',

        '8989898',

        '3'

    FROM regions

    WHERE region.name = 'Красноярский край'

    LIMIT 1;

`

**Обновление данных.**

***1. Обновление данных о докторе.***

const updateExample = `

    UPDATE doctors

    SET

        phone = +7998254,

        email = test@example.com

    WHERE doctors.id = 1;

`

***2. Изменение кода региона в двух таблицах – врачей и регионов.***

const updateID = `

    UPDATE doctors

    LEFT JOIN regions

    ON ( regions.id = doctors.regionId )

    SET

        doctors.regionId = 777,

        regions.id = 777

    WHERE regions.name = 'Республика Коми';

`

**Удаление данных.**

***1. Удаление доктора.***

const deleteExample = `

    DELETE FROM doctors WHERE doctors.id = 5;

`

***2. Удаление всех докторов из больницы 'ММКЦ Коммунарка'***

const deleteCommunarka = `

    DELETE FROM doctors

    WHERE doctors.id IN (

        SELECT doctors.id

        FROM doctors

        INNER JOIN organizations

        ON ( doctors.organizationId = organizations.id )

        WHERE organizations.name = 'ММКЦ Коммунарка'

    );

`