Разработка автоматизированной системы салона красоты

1. Описание предметной области

Вопрос повышения собственной привлекательности берет своё начало из двух фундаментальных процессов, вокруг которых строится вся жизнедеятельность человека – выжить и размножиться. А история создания и использования инструментов, по повышению симпатичности своего физического тела является актуальной и по сегодняшний день.

Желание людей «иметь всё под рукой» и суммирование одних методов с другими привело к созданию салонов красоты, основной миссией которых являются максимальное удовлетворение потребностей самого широкого круга клиентов путем оказания услуг салона красоты, индивидуального и чуткого подхода высококвалифицированных работников к каждому клиенту при обеспечении справедливого отношения к своим сотрудникам.

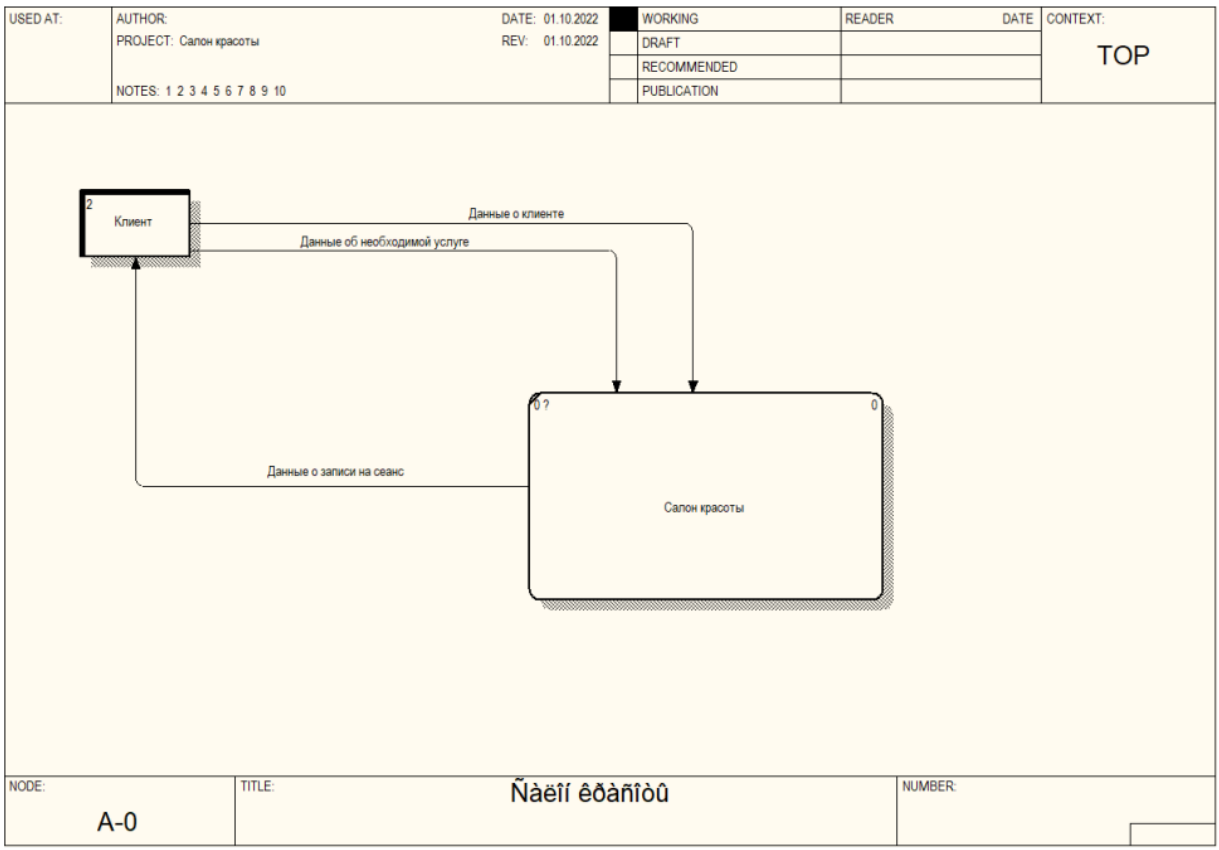
Наблюдая опыт создания салонов красоты с целью исполнения вышеописанной миссии, сформировался базовый набор услуг, которым должен обладать среднестатистический салон красоты:

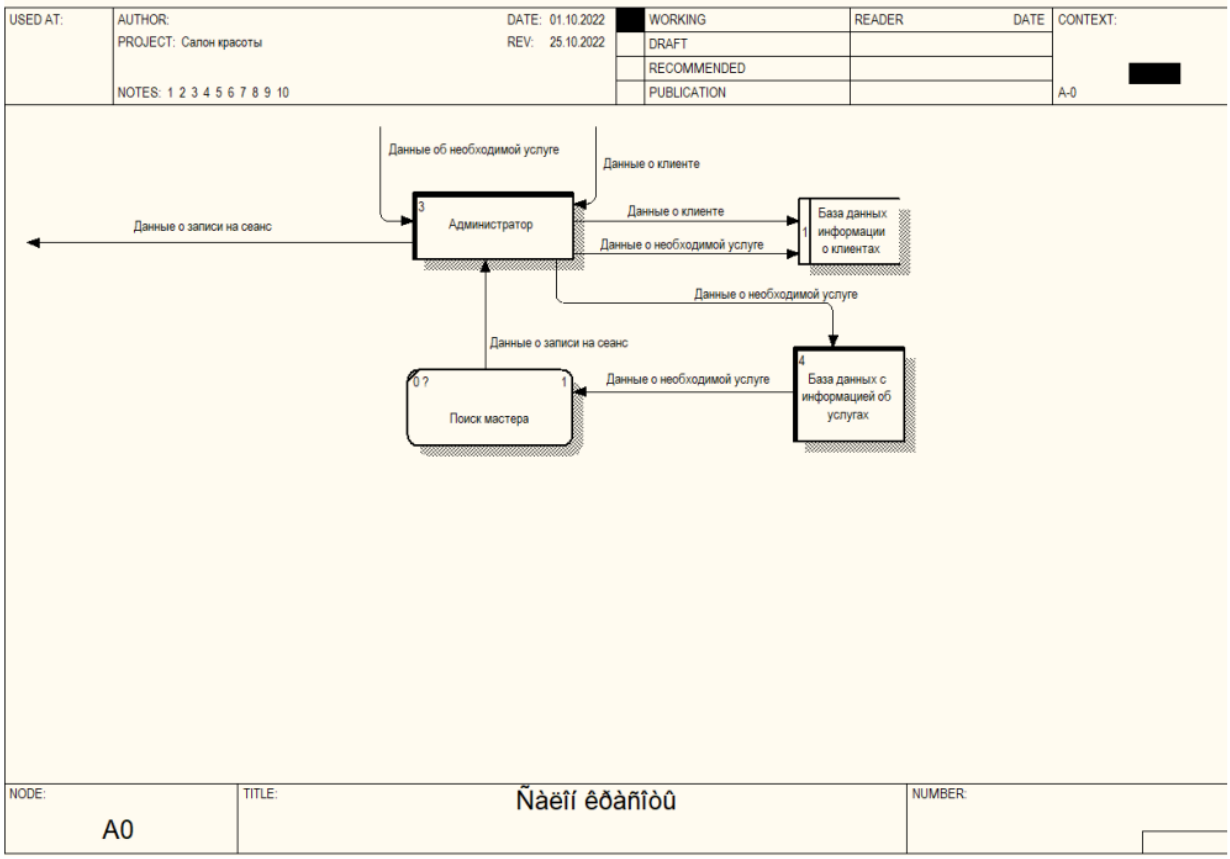
- парикмахерские услуги,
- маникюр (классический, французский, художественный маникюр),
- наращивание ногтей, реставрация ногтей,
- педикюр,
- косметолог - визажист.

При обращении клиента в салон красоты он взаимодействует с администратором, который проводит регистрацию обратившегося путём внесения в базу данных информацию о нём, после чего формируется запись на сеанс, в которой указываются: данные об администраторе, данные о клиенте, данные о мастере, данные о процедуре и данные об задействуемым в процедуре оборудованием. Таким образом в описываемой базе данных фигурируют следующие сущности: человек,

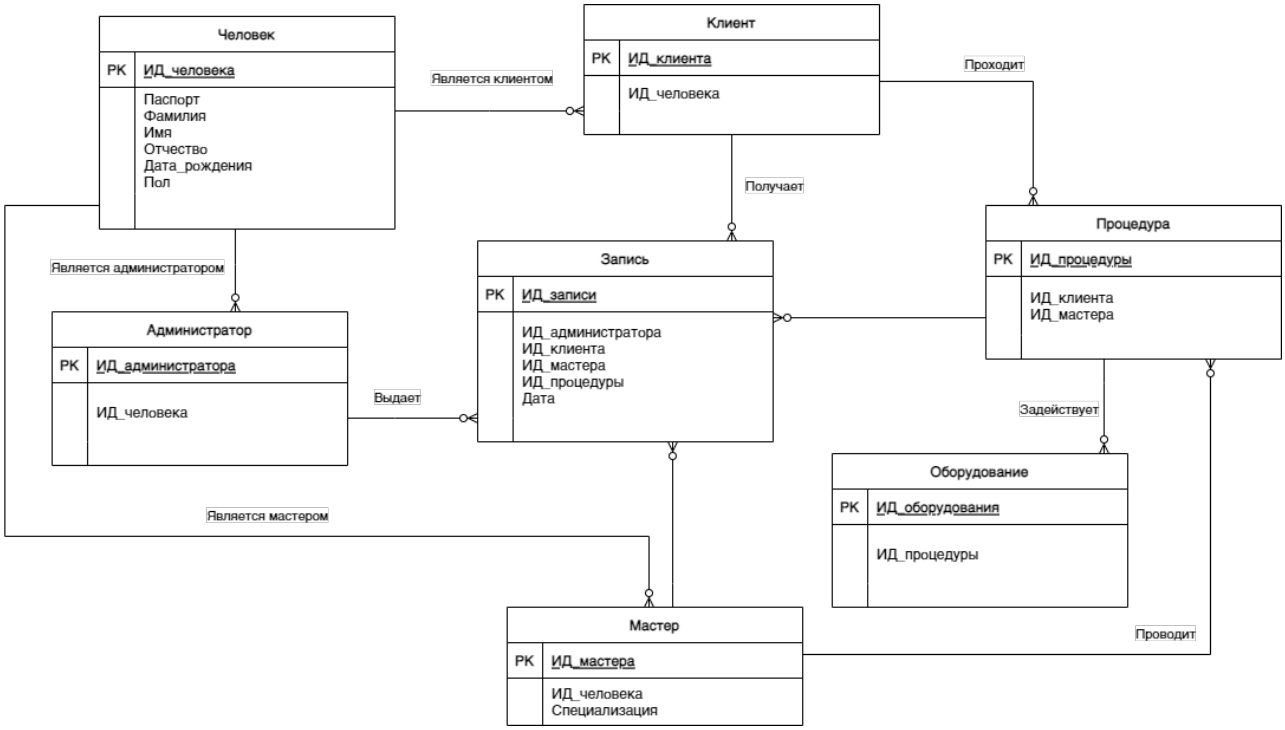
клиент, администратор, мастер, процедура, оборудование, и, формируемая на основе других сущностей таблица запись на сеанс.

2. Разработка DFD диаграммы





3. Разработка ER диаграммы



4. Генерация последовательностей

```
create sequence seq_adm
    as integer
    minvalue 0
    maxvalue 9999
    increment by 1;
```

```
create sequence seq_human
    as integer
    minvalue 0
    maxvalue 9999
    increment by 1;
```

```
create sequence seq_master
    as integer
    minvalue 0
    maxvalue 9999;
```

```
create sequence seq_proc
    as integer
    minvalue 0
    maxvalue 9999;
```

```
create sequence seq_client
    as integer
    minvalue 0
    maxvalue 9999
    increment by 1;
```

```
create sequence seq_eq
    as integer
    minvalue 0
    maxvalue 9999;
```

```
create sequence seq_rec
    as integer
    minvalue 0
    maxvalue 9999;
```

5. Генерация таблиц

```
create table HUMAN(
    ID_human integer NOT NULL PRIMARY KEY  ,
    human_passport_number integer NOT NULL UNIQUE ,
    human_surname varchar ,
    human_name varchar ,
    human_patronymic varchar ,
    human_birth_date date ,
    sex varchar
);
```

```
create table CLIENT(
    ID_client integer NOT NULL PRIMARY KEY ,
    ID_human integer NOT NULL UNIQUE ,
    CONSTRAINT fk_human FOREIGN KEY (ID_human) REFERENCES HUMAN (ID_human)
);
```

```
create table ADMINISTRATOR(
    ID_administrator integer NOT NULL PRIMARY KEY ,
    ID_human integer NOT NULL UNIQUE ,
    CONSTRAINT fk_human FOREIGN KEY (ID_human) REFERENCES HUMAN (ID_human)
);
```

```
create table MASTER(
    ID_master integer NOT NULL PRIMARY KEY ,
    ID_human integer NOT NULL UNIQUE ,
    master_specialization varchar ,
    CONSTRAINT fk_human FOREIGN KEY (ID_human) REFERENCES HUMAN (ID_human)
);
```

```
create table PROCEDURE(
    ID_procedure integer NOT NULL PRIMARY KEY ,
    ID_master integer NOT NULL UNIQUE ,
    ID_client integer NOT NULL UNIQUE ,
    CONSTRAINT fk_master FOREIGN KEY (ID_master) REFERENCES MASTER (ID_master) ,
    CONSTRAINT fk_client FOREIGN KEY (ID_client) REFERENCES CLIENT (ID_client)
);
```

```
create table EQUIPMENT(
    ID_equipment integer NOT NULL PRIMARY KEY ,
    ID_procedure integer NOT NULL UNIQUE ,
    CONSTRAINT fk_procedure FOREIGN KEY (ID_procedure) REFERENCES PROCEDURE
(ID_procedure)
);
```

```
create table RECORD(
    ID_record integer NOT NULL PRIMARY KEY ,
    ID_administrator integer NOT NULL UNIQUE ,
    ID_client integer NOT NULL UNIQUE ,
    ID_master integer NOT NULL UNIQUE ,
    ID_procedure integer NOT NULL UNIQUE ,
    record_date date ,
    CONSTRAINT fk_administrator FOREIGN KEY (ID_administrator) REFERENCES
ADMINISTRATOR (ID_administrator) ,
    CONSTRAINT fk_client FOREIGN KEY (ID_client) REFERENCES CLIENT (ID_client) ,
    CONSTRAINT fk_master FOREIGN KEY (ID_master) REFERENCES MASTER (ID_master) ,
    CONSTRAINT fk_procedure FOREIGN KEY (ID_procedure) REFERENCES PROCEDURE
(ID_procedure)
);
```

6. Процедуры

1)Добавление записи в таблицу HUMAN

```
create or replace procedure ADD_HUMAN(rec RECORD) as
    $$
    DECLARE
        id_human integer;
    BEGIN
        id_human := seq_human.nextval();
        begin
            insert into HUMAN
            values (id_human, rec.human_passport_number,
                    rec.human_surname, rec.human_name, rec.human_patronymic,
                    rec.human_birth_date, rec.sex);
            exception
                when unique_violation then raise NOTICE 'Запись уже существует';
        end;
    end;
    $$ LANGUAGE plpgsql;
```

2)Добавление записи в таблицу CLIENT

```
create or replace procedure ADD_CLIENT(id_human integer) as
    $$
    DECLARE
        id_client integer;
    BEGIN
        id_client := seq_client.nextval();
        begin
            insert into CLIENT
            values (id_human, id_client);
            exception
                when unique_violation then raise NOTICE 'Запись уже существует';
        end;
    end;
    $$ LANGUAGE plpgsql;
```

3)Добавление записи в таблицу ADMINISTRATOR

```
create or replace procedure ADD_ADMINISTRATOR(id_human integer) as
    $$
    DECLARE
        id_administrator integer;
    BEGIN
        id_administrator := seq_adm.nextval();
        begin
            insert into ADMINISTRATOR
            values (id_human, id_administrator);
            exception
                when unique_violation then raise NOTICE 'Запись уже существует';
        end;
    end;
    $$ LANGUAGE plpgsql;
```

4)Добавление записи в таблицу MASTER

```
create or replace procedure ADD_MASTER(id_human integer, master_spec varchar) as
    $$
    DECLARE
        id_master integer;
    BEGIN
        id_master := seq_master.nextval();
        begin
            insert into MASTER
            values (id_master, id_human, master_spec);
            exception
                when unique_violation then raise NOTICE 'Запись уже существует';
        end;
    end;
    $$ LANGUAGE plpgsql;
```

5)Добавление записи в таблицу PROCEDURE и EQUIPMENT

```
create or replace procedure PROCEDURE(name_master integer, name_client integer)
as
    $$
    DECLARE
        id_procedure integer;
        id_equipment integer;
        id_master_for_insert integer;
        id_client_for_insert integer;
    begin
        id_procedure = seq_proc.nextval();
        id_equipment := seq_eq.nextval();
        select id_master from master join human using(ID_human) where human_name
= name_master into id_master_for_insert;
        select id_client from client join human using(ID_human) where human_name
= name_master into id_client_for_insert;
        begin
            insert into PROCEDURE
            values (id_procedure, id_master_for_insert, id_client_for_insert);
            insert into EQUIPMENT
            values (id_equipment, id_procedure);
        end;
    end;
    $$LANGUAGE plpgsql;
```

7)Реализация бизнес процесса «назначение сеанса»

```
create or replace procedure RECORD(id_adm integer, id_client integer, id_master
integer, id_procedure integer,id_equipment integer, date_rec date,
specialization varchar) as
    $$
    DECLARE
        id_rec integer;
        v_master integer;
    begin
        id_rec = seq_rec.nextval();
        begin
            insert into RECORD
            values (id_rec, id_client, id_master, id_procedure, id_equipment,
date_rec);
        end;
    end;

    $$LANGUAGE plpgsql;
```

7. Добавление пользователей

```
CREATE ROLE administrator WITH
   LOGIN
   SUPERUSER
   CREATEDB
   CREATEROLE
   INHERIT
   REPLICATION
   CONNECTION LIMIT -1;
```

```
CREATE ROLE master WITH
   LOGIN
   SUPERUSER
   CREATEDB
   CREATEROLE
   INHERIT
   REPLICATION
   CONNECTION LIMIT -1;
```

```
CREATE ROLE "read_only_role";
CREATE ROLE "read_write_role";
```

```
GRANT SELECT ON TABLE
ADMINISTRATOR,CLIENT,EQUIPMENT,HUMAN,MASTER,PROCEDURE,RECORD TO read_only_role;

GRANT SELECT, INSERT, UPDATE, DELETE on table
ADMINISTRATOR,CLIENT,EQUIPMENT,HUMAN,MASTER,PROCEDURE,RECORD TO read_write_role;
```

```
GRANT read_only_role TO master;
GRANT read_write_role TO administrator;
```