National University
of computer and emerging sciences

STATE OFFICE DATABASE APPLICATION

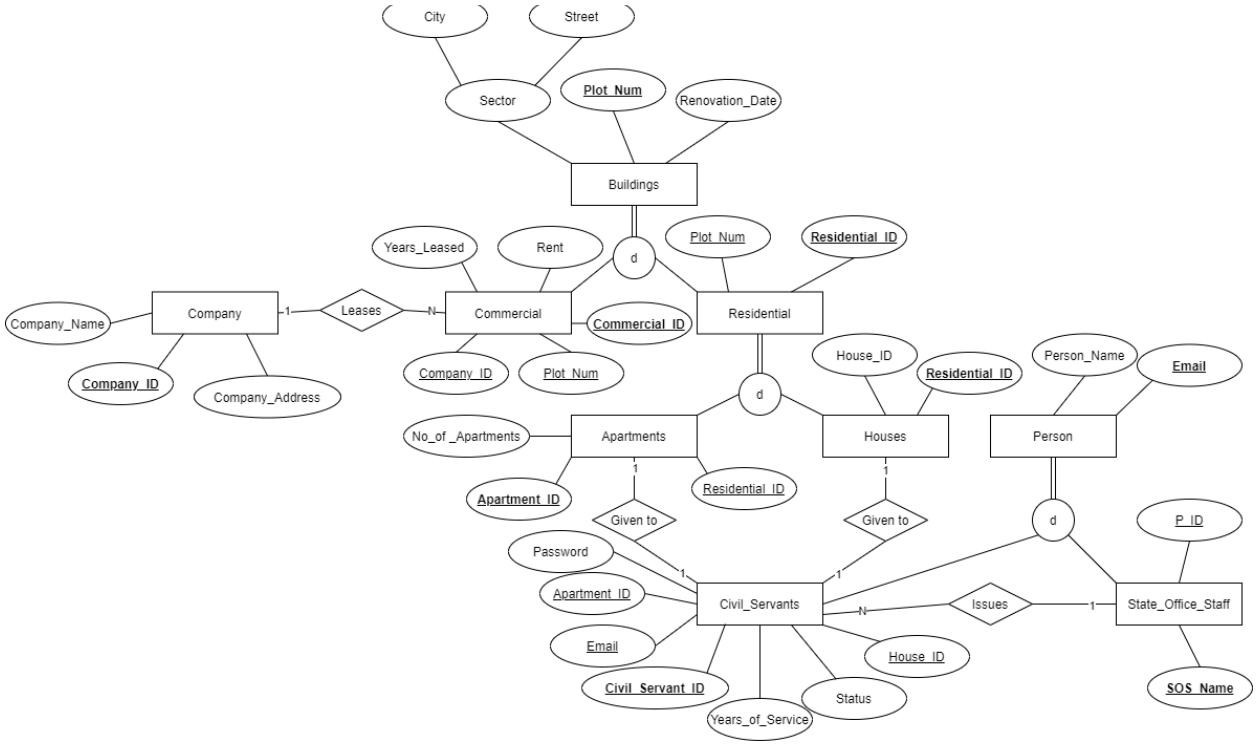
Group Members:

Maaz Tariq 19I-0660

Azwar Shariq 19I-0728

Talha 19I-2049

Enhanced Entity Relation Diagram



Relational Schema

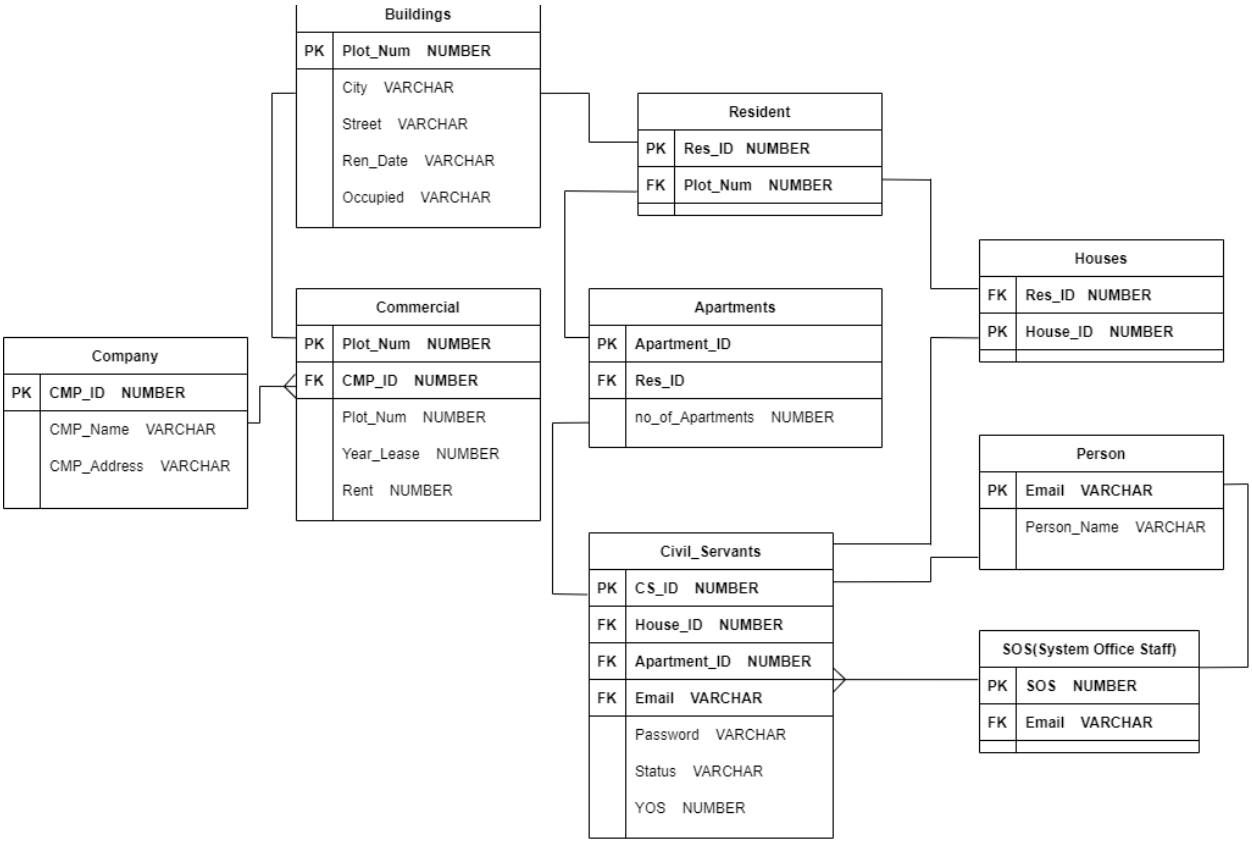


Table Descriptions

CREATE TABLE BUILDINGS (

PLOT_NUM *NUMBER*,

CITY *NUMBER*,

STREET *NUMBER*,

REN_DATE *VARCHAR* (20),

OCCUPIED *VARCHAR* (30),

CONSTRAINT PK_BUILD_1 PRIMARY KEY (PLOT_NUM)

);

This table basically serves as a foundation, or basis for any and all real estate types, be it commercial or residential. It keeps track of the plot number, renovation date and address associated to each real estate property.

CREATE TABLE RESIDENT (

RES_ID *NUMBER*,

PLOT_NUM *NUMBER*,

CONSTRAINT PK_RES_1 PRIMARY KEY (RES_ID),

CONSTRAINT FK_PLOT_1 FOREIGN KEY (PLOT_NUM) REFERENCES BUILDINGS (PLOT_NUM)

);

The resident table contains all those cumulative plots that are of residential type, which can further be subdivided between apartments and houses.

CREATE TABLE HOUSES (

RES_ID *NUMBER*,

HOUSE_ID *NUMBER*,

CONSTRAINT PK_H_ID_1 PRIMARY KEY (HOUSE_ID),

CONSTRAINT FK_RES_1 FOREIGN KEY (RES_ID) REFERENCES RESIDENT (RES_ID)

);

This table contains all those residential properties that are of type house. It contains the house number of each such property

CREATE TABLE COMPANY (

CMP_ID *NUMBER*,

CMP_NAME *VARCHAR* (20),

CMP_ADD *VARCHAR* (20),

CONSTRAINT PK_CMP_2 PRIMARY KEY (CMP_ID)

);

CREATE TABLE COMMERCIAL (

CMP_ID *NUMBER*,

PLOT_NUM *NUMBER*,

YEAR_LEASE *NUMBER*,

RENT NUMBER,

CONSTRAINT PK_PLOT_1 PRIMARY KEY (PLOT_NUM),

CONSTRAINT FK_PLOT_3 FOREIGN KEY (PLOT_NUM) REFERENCES BUILDINGS (PLOT_NUM),

CONSTRAINT FK_CMP_2 FOREIGN KEY (CMP_ID) REFERENCES COMPANY (CMP_ID)

);

CREATE TABLE APPARTMENTS (

APPARTMENT_ID NUMBER,

RES_ID NUMBER,

NO_OF_APPARTMENTS *NUMBER*,

CONSTRAINT PK_APP_1 PRIMARY KEY (APPARTMENT_ID),

CONSTRAINT FK_RES_2 FOREIGN KEY (RES_ID) REFERENCES RESIDENT (RES_ID)

);

This table contains all those residential properties that are of type apartments. It contains the apartment number of each such property, along with the number of apartments.

CREATE TABLE PERSON (

EMAIL *VARCHAR* (30),

PERSON_NAME VARCHAR (20),

CONSTRAINT PK_EMAIL_1 PRIMARY KEY (email)

);

The persons table acts as a parent entity to the Civil Servant and the State Office Staff table, and as such contains the name, email and password for each user/admin.

CREATE TABLE SOS (

SOS_ID *NUMBER*,

EMAIL *VARCHAR* (30),

CONSTRAINT PK_SOS_1 PRIMARY KEY (SOS_ID),

CONSTRAINT FK_Email_1 FOREIGN KEY (EMAIL) REFERENCES person (EMAIL)

);

CREATE TABLE CIVIL_SERVANTS (

CS_ID *NUMBER*,

EMAIL *VARCHAR* (30),

PASSWORD *VARCHAR* (20),

STATUS *VARCHAR* (20),

YOS *NUMBER*,

ACCESSTYPE VARCHAR (20),

HOUSE_ID *NUMBER*,

APPARTMENT_ID NUMBER,

CONSTRAINT PK_CS_1 PRIMARY KEY (CS_ID),

CONSTRAINT FK_Email_2 FOREIGN KEY (EMAIL) REFERENCES person (EMAIL),

CONSTRAINT FK_house_id_2 FOREIGN KEY (HOUSE_ID) REFERENCES houses (HOUSE_ID),

CONSTRAINT FK_app_id_2 FOREIGN KEY (APPARTMENT_ID) REFERENCES APPARTMENT (APPARTMENT_ID)

);

The civil servants table is, at its core a table of 'users'. These users may or may not be present in a waiting list to have a residential property allotted to them. Alongside this, their method of access is predetermined. Moreover, additional details such as years of service, name and the ID of any property allotted is also present. The state office state is basically an 'admin' entity. It contains an email address identifying each member of the state office staff. The company table gives details about the company such as company name and the company he address. Lastly, the commercial table details the plots held by a single company, including the plot number and the years the plot is leased for.

Table Relations

Commercial and Resident, are both a form of buildings, and as such though values such as plot number are inherited, both entities have their own separate attributes.

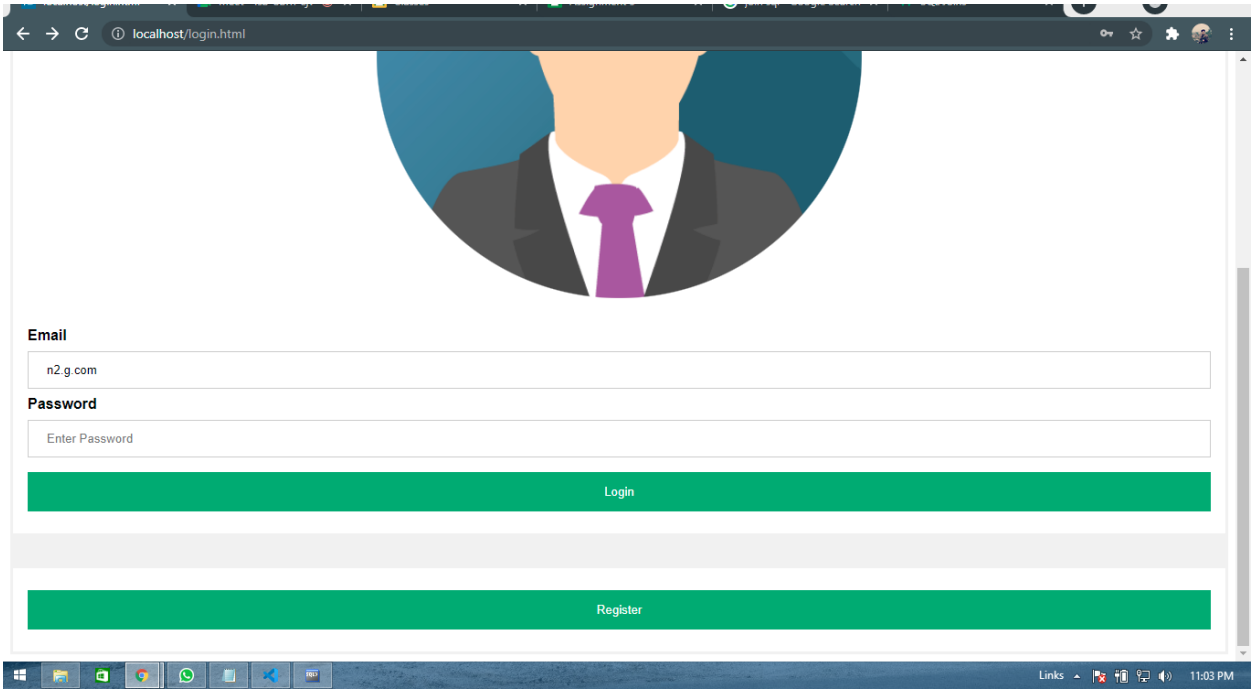
Company and commercial have a 1-to-many relationship, as a single company may choose to lease out multiple plots.

Apartments and houses are both a form of residential properties. As such, they carry over values held by the residents table but each has their own specific attributes, such as number of apartments in the apartment entity.

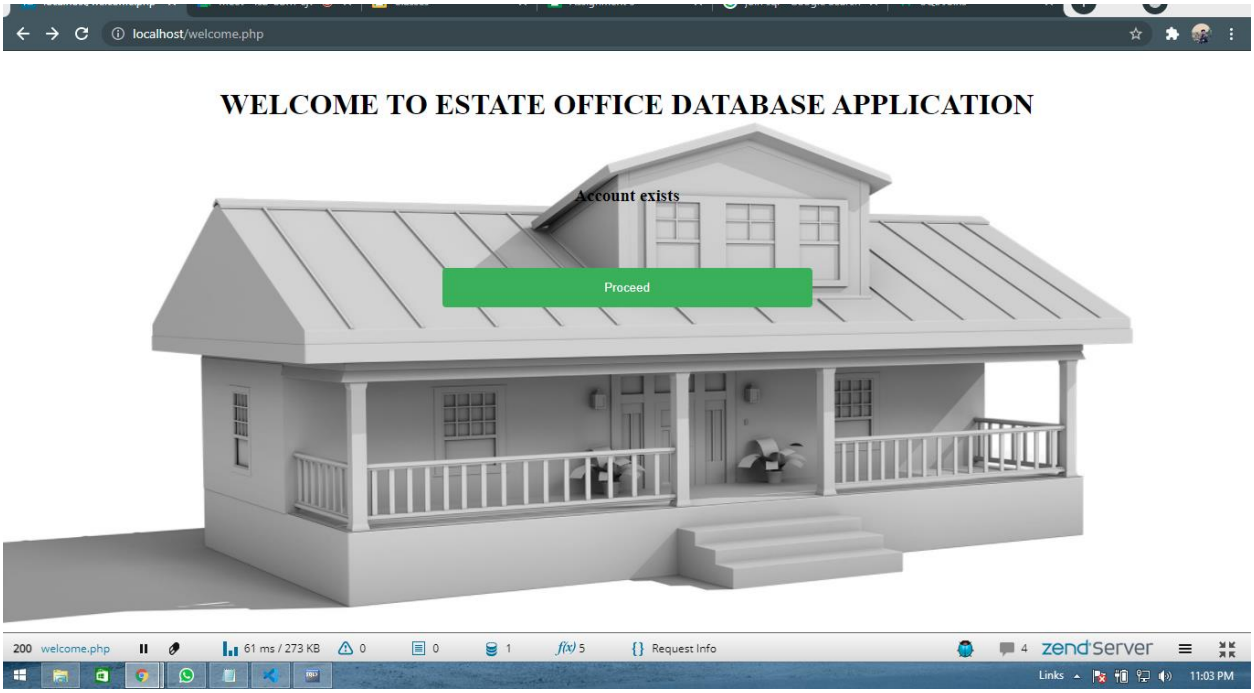
The civil servants table is connected to both apartments and house, and it therefore can accurately show which residential property belongs to the civil servant.

Lastly, both civil servants and staff members contain attributes from the persons table, as they both effectively belong to a subclass of the persons table.

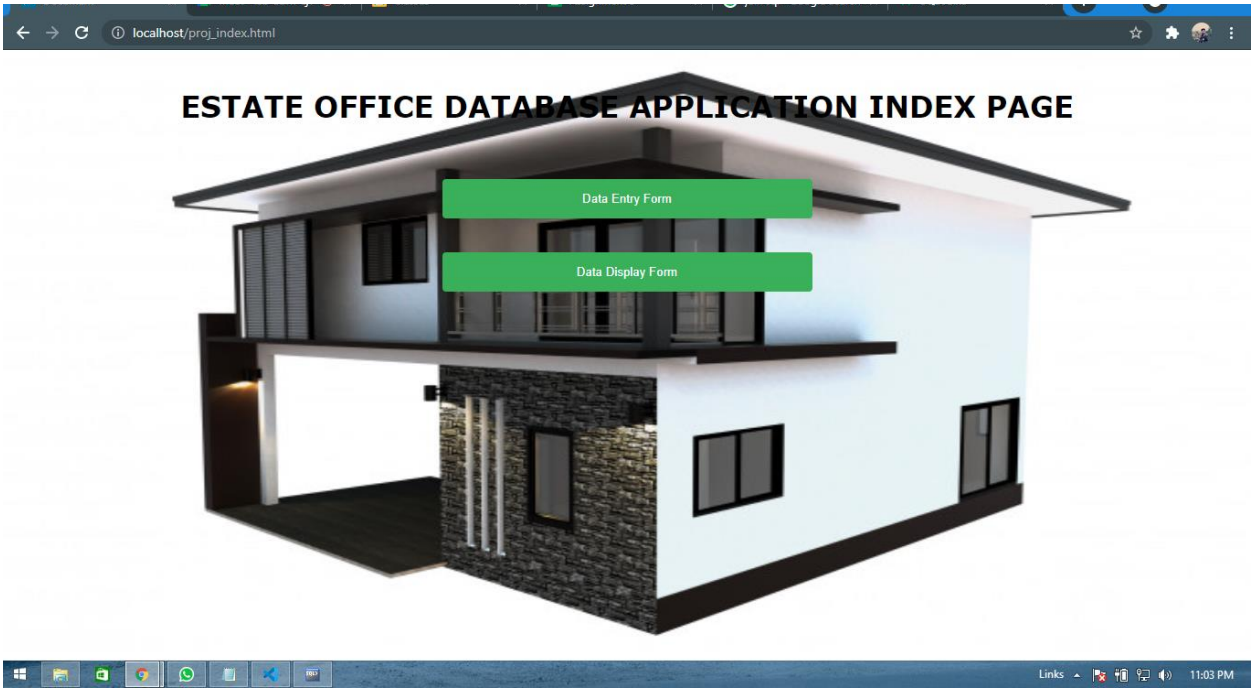
Login Page



Welcome Page

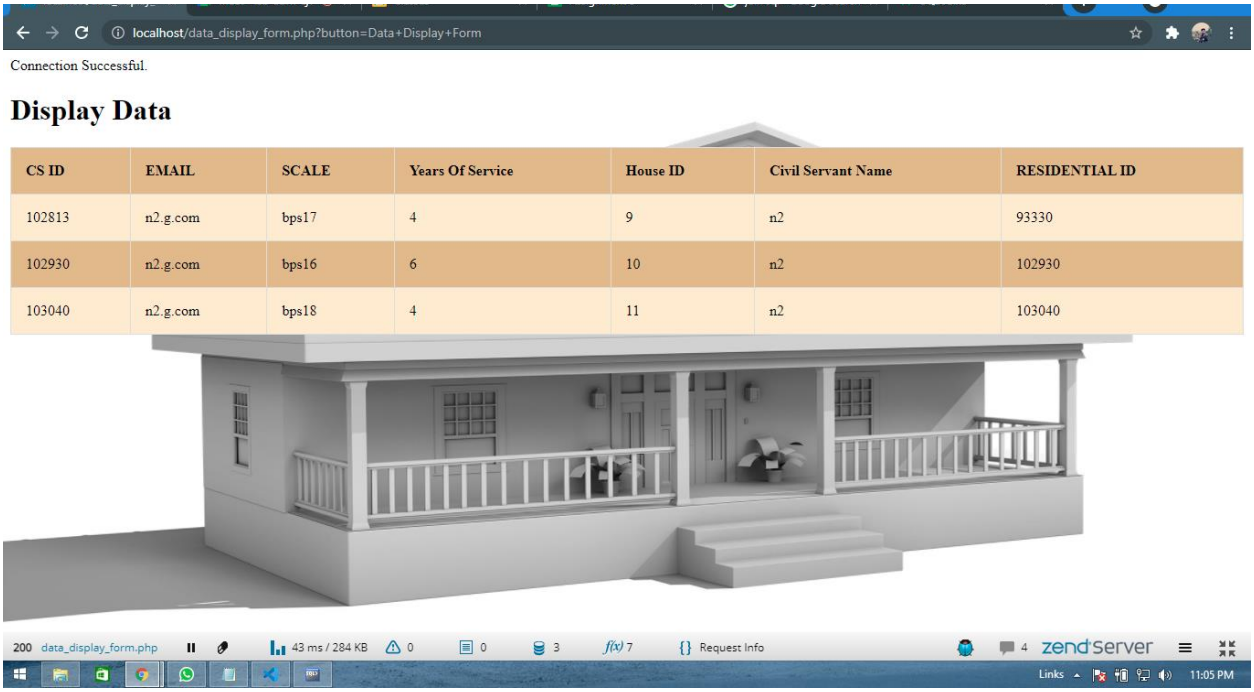


Index Page



Data Entry Form

Data Display Form



CS ID	EMAIL	SCALE	Years Of Service	House ID	Civil Servant Name	RESIDENTIAL ID
102813	n2.g.com	bps17	4	9	n2	93330
102930	n2.g.com	bps16	6	10	n2	102930
103040	n2.g.com	bps18	4	11	n2	103040

Table Descriptions

```
SQL> desc buildings
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------

 PLOT_NUM                                  NOT NULL NUMBER
 CITY                                               VARCHAR2(20)
 STREET                                             NUMBER
 REN_DATE                                           VARCHAR2(20)
 OCCUPIED                                           VARCHAR2(30)

SQL> desc  commercial
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------

 CMP_ID                                             NUMBER
 PLOT_NUM                                  NOT NULL NUMBER
 YEAR_LEASE                                         NUMBER
 RENT                                               NUMBER

SQL> desc  company
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------

 CMP_ID                                    NOT NULL NUMBER
 CMP_NAME                                           VARCHAR2(20)
 CMP_ADD                                            VARCHAR2(20)

SQL> desc  residential
ERROR:
ORA-04043: object residential does not exist


SQL> desc  resident
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------

 RES_ID                                    NOT NULL NUMBER
 PLOT_NUM                                           NUMBER

SQL> desc  houses
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------

 RES_ID                                             NUMBER
 HOUSE_ID                                  NOT NULL NUMBER

SQL> desc  appartments
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------

 APPARTMENT_ID                             NOT NULL NUMBER
 RES_ID                                             NUMBER
 NO_OF_APPARTMENTS                                  NUMBER
```

```
SQL> desc  civil_servants
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------

 CS_ID                                     NOT NULL NUMBER
 EMAIL                                              VARCHAR2(30)
 SCALE                                              VARCHAR2(20)
 STATUS                                             VARCHAR2(20)
 YOS                                                NUMBER
 ACCESSTYPE                                         VARCHAR2(20)
 HOUSE_ID                                           NUMBER
 APPARTMENT_ID                                      NUMBER

SQL> desc  sos
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------

 SOS_ID                                    NOT NULL NUMBER
 EMAIL                                              VARCHAR2(30)

SQL> desc  person
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------

 PERSON_NAME                                        VARCHAR2(20)
 EMAIL                                     NOT NULL VARCHAR2(30)
 PWD                                                VARCHAR2(30)
```

Reflections and what could have been done better

In the scenario implemented, semantics and normalization could have been used to make the database implementation more efficient and effective. Boolean checks for retirement and similar attributes could be used to provide a much better implementation of the data.

Despite having a logical structure and a planned methodology on the implementation of access control, this aspect proved to be more difficult than expected. As such, by the use of checks and the “if” clause in PHP, one could easily divide the data control to be provided to users on the basis of whether the access set for them by the admin is either “edit, view or add”.

Software Used

We used Draw.io, now diagrams.net to make our EERD and Schema. Visual Studio Code for PHP and sql for uh sql.