CSI 2132 Database I

Deliverable 2 Report

Written by:
Alex Nadeau - 300117775
Joseph Abonasara - 300115223
Ricardo Saikali - 300117848
Stefano Stella - 300113450
Elias Maalouf - 300118475

for Professor Olubisi Runsewe

Part A

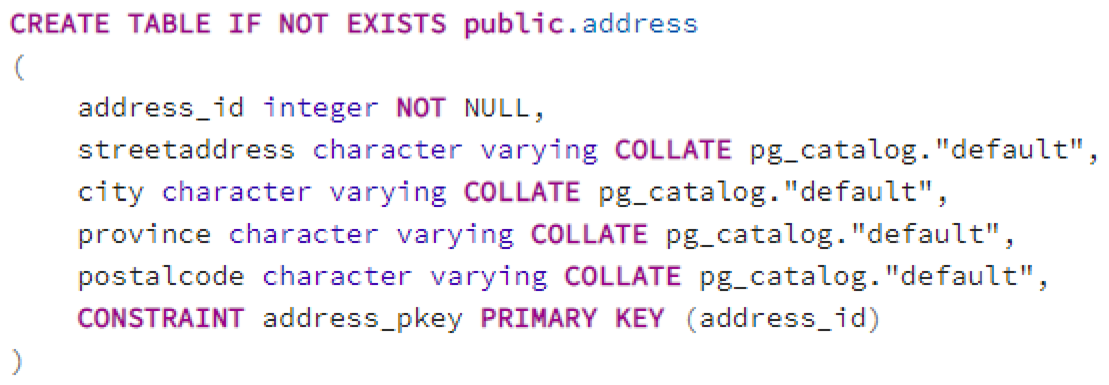
The technologies used (e.g., DBMS & programming languages) used for the implementation of your application.

DBMS: We used PostgreSQL and PGAdmin to store data regarding all branches, dentists, patients and more.

Programming languages: This program was made using Java, we used the PostgreSQL Driver to connect the application to the online database. We also used Java Swing for the front-end of the application. Additionally, we used python to create scripts that generate information that was stored in the database.

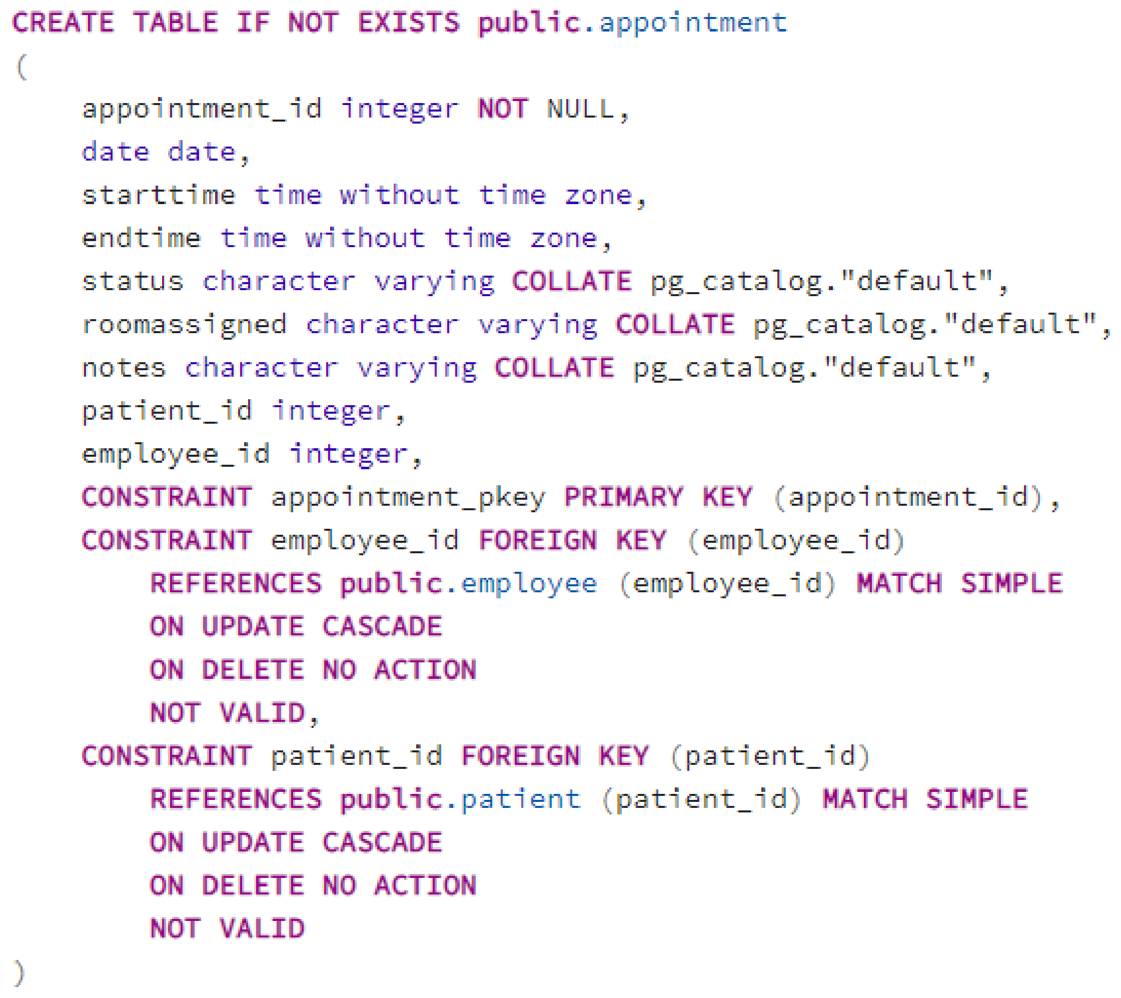
Part B: A list with the DDLs used to create your database

Table 1: public.address



```
CREATE TABLE IF NOT EXISTS public.address
(
    address_id integer NOT NULL,
    streetaddress character varying COLLATE pg_catalog."default",
    city character varying COLLATE pg_catalog."default",
    province character varying COLLATE pg_catalog."default",
    postalcode character varying COLLATE pg_catalog."default",
    CONSTRAINT address_pkey PRIMARY KEY (address_id)
)
```

Table 2: public.appointment



```
CREATE TABLE IF NOT EXISTS public.appointment
(
    appointment_id integer NOT NULL,
    date date,
    starttime time without time zone,
    endtime time without time zone,
    status character varying COLLATE pg_catalog."default",
    roomassigned character varying COLLATE pg_catalog."default",
    notes character varying COLLATE pg_catalog."default",
    patient_id integer,
    employee_id integer,
    CONSTRAINT appointment_pkey PRIMARY KEY (appointment_id),
    CONSTRAINT employee_id FOREIGN KEY (employee_id)
        REFERENCES public.employee (employee_id) MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE NO ACTION
        NOT VALID,
    CONSTRAINT patient_id FOREIGN KEY (patient_id)
        REFERENCES public.patient (patient_id) MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE NO ACTION
        NOT VALID
)
```

Table 3: public.appointmentprocedure

```
CREATE TABLE IF NOT EXISTS public.appointmentprocedure
(
    procedurecode integer NOT NULL,
    description character varying COLLATE pg_catalog."default",
    toothinvolved character varying COLLATE pg_catalog."default",
    amountprocedures integer,
    appointment_id integer,
    CONSTRAINT appointmentprocedure_pkey PRIMARY KEY (procedurecode),
    CONSTRAINT appointment_id FOREIGN KEY (appointment_id)
        REFERENCES public.appointment (appointment_id) MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE NO ACTION
        NOT VALID
)
```

Table 4: public.appointmenttype

```
CREATE TABLE IF NOT EXISTS public.appointmenttype
(
    type_id integer NOT NULL,
    appointmenttype character varying COLLATE pg_catalog."default",
    appointment_id integer,
    CONSTRAINT appointmenttype_pkey PRIMARY KEY (type_id),
    CONSTRAINT appointment_id FOREIGN KEY (appointment_id)
        REFERENCES public.appointment (appointment_id) MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE NO ACTION
        NOT VALID
)
```

Table 5: public.branch

```sql
CREATE TABLE IF NOT EXISTS public.branch
(
    branch_id integer NOT NULL,
    address_id integer,
    CONSTRAINT branch_pkey PRIMARY KEY (branch_id),
    CONSTRAINT address_id FOREIGN KEY (address_id)
        REFERENCES public.address (address_id) MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE NO ACTION
        NOT VALID
)
```

Table 6: public.contactinformation

```sql
CREATE TABLE IF NOT EXISTS public.contactinformation
(
    contactinfo_id integer NOT NULL,
    email character varying(30) COLLATE pg_catalog."default",
    phonenumber character varying COLLATE pg_catalog."default",
    CONSTRAINT contactinformation_pkey PRIMARY KEY (contactinfo_id)
)
```

Table 7: public.employee

```
CREATE TABLE IF NOT EXISTS public.employee
(
    employee_id integer NOT NULL,
    employee_role character varying COLLATE pg_catalog."default",
    employee_type character varying COLLATE pg_catalog."default",
    salary double precision,
    branch_id integer,
    user_id integer,
    CONSTRAINT employee_pkey PRIMARY KEY (employee_id),
    CONSTRAINT branch_id FOREIGN KEY (branch_id)
        REFERENCES public.branch (branch_id) MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE NO ACTION
        NOT VALID,
    CONSTRAINT user_id FOREIGN KEY (user_id)
        REFERENCES public."user" (user_id) MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE NO ACTION
        NOT VALID
)
```

Table 8: public.feecharge

```
CREATE TABLE IF NOT EXISTS public.feecharge
(
    fee_id integer NOT NULL,
    procedure character varying COLLATE pg_catalog."default",
    feecode integer,
    charge double precision,
    appointment_id integer,
    procedurecode integer,
    CONSTRAINT feecharge_pkey PRIMARY KEY (fee_id),
    CONSTRAINT appointment_id FOREIGN KEY (appointment_id)
        REFERENCES public.appointment (appointment_id) MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE NO ACTION
        NOT VALID,
    CONSTRAINT procedurecode FOREIGN KEY (procedurecode)
        REFERENCES public.appointmentprocedure (procedurecode) MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE NO ACTION
        NOT VALID
)
```

Table 9: public.insuranceclaim

```
CREATE TABLE IF NOT EXISTS public.insuranceclaim
(
    claim_id integer NOT NULL,
    amountclaimed double precision,
    patient_id integer,
    CONSTRAINT insuranceclaim_pkey PRIMARY KEY (claim_id),
    CONSTRAINT patient_id FOREIGN KEY (patient_id)
        REFERENCES public.patient (patient_id) MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE NO ACTION
        NOT VALID
)
```

Table 10: public.invoice

```
CREATE TABLE IF NOT EXISTS public.invoice
(
    invoice_id integer NOT NULL,
    dateissued date,
    contactinfo character varying COLLATE pg_catalog."default",
    patientcharge double precision,
    insurancecharge double precision,
    totalcharge double precision,
    discount double precision,
    penalty double precision,
    procedurecode integer,
    CONSTRAINT invoice_pkey PRIMARY KEY (invoice_id),
    CONSTRAINT procedurecode FOREIGN KEY (procedurecode)
        REFERENCES public.appointmentprocedure (procedurecode) MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE NO ACTION
        NOT VALID
)
```

Table 11: public.patient

```
CREATE TABLE IF NOT EXISTS public.patient
(
    patient_id integer NOT NULL,
    userid integer,
    insurancenumber character varying COLLATE pg_catalog."default",
    CONSTRAINT patient_pkey PRIMARY KEY (patient_id),
    CONSTRAINT user_id FOREIGN KEY (userid)
        REFERENCES public."user" (user_id) MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE NO ACTION
        NOT VALID
)
```

Table 12: public.patientbilling

```
CREATE TABLE IF NOT EXISTS public.patientbilling
(
    billing_id integer NOT NULL,
    paymentmethod character varying COLLATE pg_catalog."default",
    paymentportion double precision,
    insuranceportion double precision,
    totalamount double precision,
    patient_id integer,
    claim_id integer,
    invoice_id integer,
    procedurecode integer,
    CONSTRAINT patientbilling_pkey PRIMARY KEY (billing_id),
    CONSTRAINT claim_id FOREIGN KEY (claim_id)
        REFERENCES public.insuranceclaim (claim_id) MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE NO ACTION
        NOT VALID,
    CONSTRAINT invoice_id FOREIGN KEY (invoice_id)
        REFERENCES public.invoice (invoice_id) MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE NO ACTION
        NOT VALID,
    CONSTRAINT patient_id FOREIGN KEY (patient_id)
        REFERENCES public.patient (patient_id) MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE NO ACTION
        NOT VALID,
    CONSTRAINT procedurecode FOREIGN KEY (procedurecode)
        REFERENCES public.appointmentprocedure (procedurecode) MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE NO ACTION
        NOT VALID
)
```

Table 13: public.proceduretype

```
CREATE TABLE IF NOT EXISTS public.proceduretype
(
    proceduretype_id integer NOT NULL,
    type character varying COLLATE pg_catalog."default",
    procedurecode integer,
    CONSTRAINT proceduretype_pkey PRIMARY KEY (proceduretype_id),
    CONSTRAINT procedurecode FOREIGN KEY (procedurecode)
        REFERENCES public.appointmentprocedure (procedurecode) MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE NO ACTION
        NOT VALID
)
```

Table 14: public.record

```
CREATE TABLE IF NOT EXISTS public.record
(
    record_id integer NOT NULL,
    notes character varying COLLATE pg_catalog."default",
    employee_id integer,
    patient_id integer,
    CONSTRAINT record_pkey PRIMARY KEY (record_id),
    CONSTRAINT employee_id FOREIGN KEY (employee_id)
        REFERENCES public.employee (employee_id) MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE NO ACTION
        NOT VALID,
    CONSTRAINT patient_id FOREIGN KEY (patient_id)
        REFERENCES public.patient (patient_id) MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE NO ACTION
        NOT VALID
)
```

Table 15: public.review

```
CREATE TABLE IF NOT EXISTS public.review
(
    review_id integer NOT NULL,
    "professionalism " integer,
    communication integer,
    cleanliness integer,
    date date,
    patient_id integer,
    branch_id integer,
    CONSTRAINT review_pkey PRIMARY KEY (review_id),
    CONSTRAINT branch_id FOREIGN KEY (branch_id)
        REFERENCES public.branch (branch_id) MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE NO ACTION
        NOT VALID,
    CONSTRAINT patient_id FOREIGN KEY (patient_id)
        REFERENCES public.patient (patient_id) MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE NO ACTION
        NOT VALID
)
```

Table 16: public.treatment

```
CREATE TABLE IF NOT EXISTS public.treatment
(
    treatment_id integer NOT NULL,
    medication character varying COLLATE pg_catalog."default",
    symptoms character varying COLLATE pg_catalog."default",
    toothinvolved character varying COLLATE pg_catalog."default",
    comments character varying COLLATE pg_catalog."default",
    record_id integer,
    appointment_id integer,
    type_id integer,
    CONSTRAINT treatment_pkey PRIMARY KEY (treatment_id),
    CONSTRAINT appointment_id FOREIGN KEY (appointment_id)
        REFERENCES public.appointment (appointment_id) MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE NO ACTION
        NOT VALID,
    CONSTRAINT record_id FOREIGN KEY (record_id)
        REFERENCES public.record (record_id) MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE NO ACTION
        NOT VALID,
    CONSTRAINT type_id FOREIGN KEY (type_id)
        REFERENCES public.appointmenttype (type_id) MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE NO ACTION
        NOT VALID
)
```

Table 17: public.treatmenttype

```
CREATE TABLE IF NOT EXISTS public.treatmenttype
(
    treatmenttype_id integer NOT NULL,
    type character varying COLLATE pg_catalog."default",
    CONSTRAINT treatmenttype_pkey PRIMARY KEY (treatmenttype_id)
)
```

Table 18: public.user

```
CREATE TABLE IF NOT EXISTS public."user"
(
    firstname character varying(30) COLLATE pg_catalog."default",
    middlename character varying(30) COLLATE pg_catalog."default",
    lastname character varying(30) COLLATE pg_catalog."default",
    gender character varying(30) COLLATE pg_catalog."default",
    ssn integer,
    datebirth date,
    contactinfo_id integer,
    user_id integer NOT NULL,
    address_id integer,
    CONSTRAINT user_id PRIMARY KEY (user_id),
    CONSTRAINT address_id FOREIGN KEY (address_id)
        REFERENCES public.address (address_id) MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE NO ACTION
        NOT VALID,
    CONSTRAINT contactinfo_id FOREIGN KEY (contactinfo_id)
        REFERENCES public.contactinformation (contactinfo_id) MATCH SIMPLE
        ON UPDATE CASCADE
        ON DELETE NO ACTION
        NOT VALID
)
```

To connect to the database you will need the following credentials:

- **Host**: ec2-52-21-136-176.compute-1.amazonaws.com
- **Port**: 5432
- **Username**: hinlvtrnpvrmez
- **Password**: 842f2aa30defab20969c9698647aaa9403793f2866f1059f0b23a459b07ea193
- **Database**: ddh0f405ip9rmj

Part C: Installation Guide

1. Download the Zip file “databaseProject.zip”
2. Make sure you have Java Installed on your machine
3. Unzip the “databaseProject” file
4. Navigate to src -> main -> java
5. Open a command prompt in the java folder
6. Compile the main.java file (javac main.java)
7. Run the main.java file with the postgresql-42.3.3.jar
   (java -cp ".\postgresql-42.3.3.jar;.\" main OR java -cp ".\postgresql-42.3.3.jar;" main)

If you have any questions related to this process email anade080@uottawa.ca or anyone else on the team.

Some of the valid **employeeIDs** are:
ReceptionistID: range from 4 to 6
DentistID: range from 7 to 39
Managers: range from 1 to 3

Valid **PatientIDs** range from 0 to 40. For example patient 40 is Paul Cook.