Raul G. Martinez (PID: A12461871)

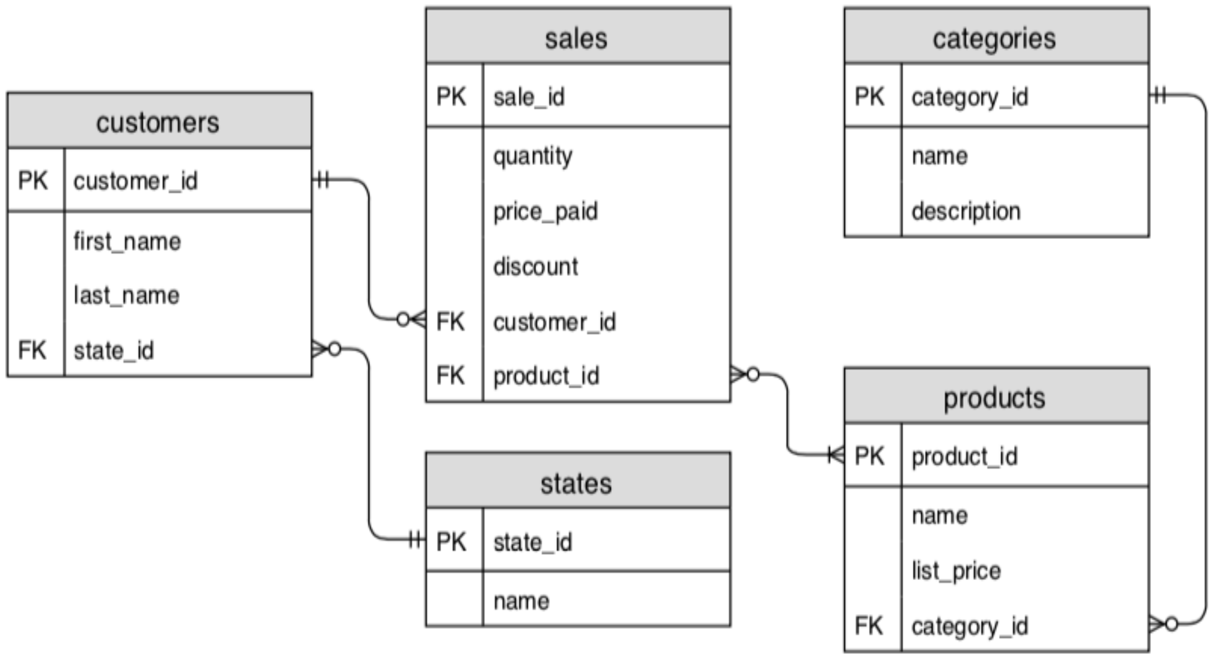
MAS DSE 201 Homework: Sales Cube

Milestone I [due Jan 28 at midnight]

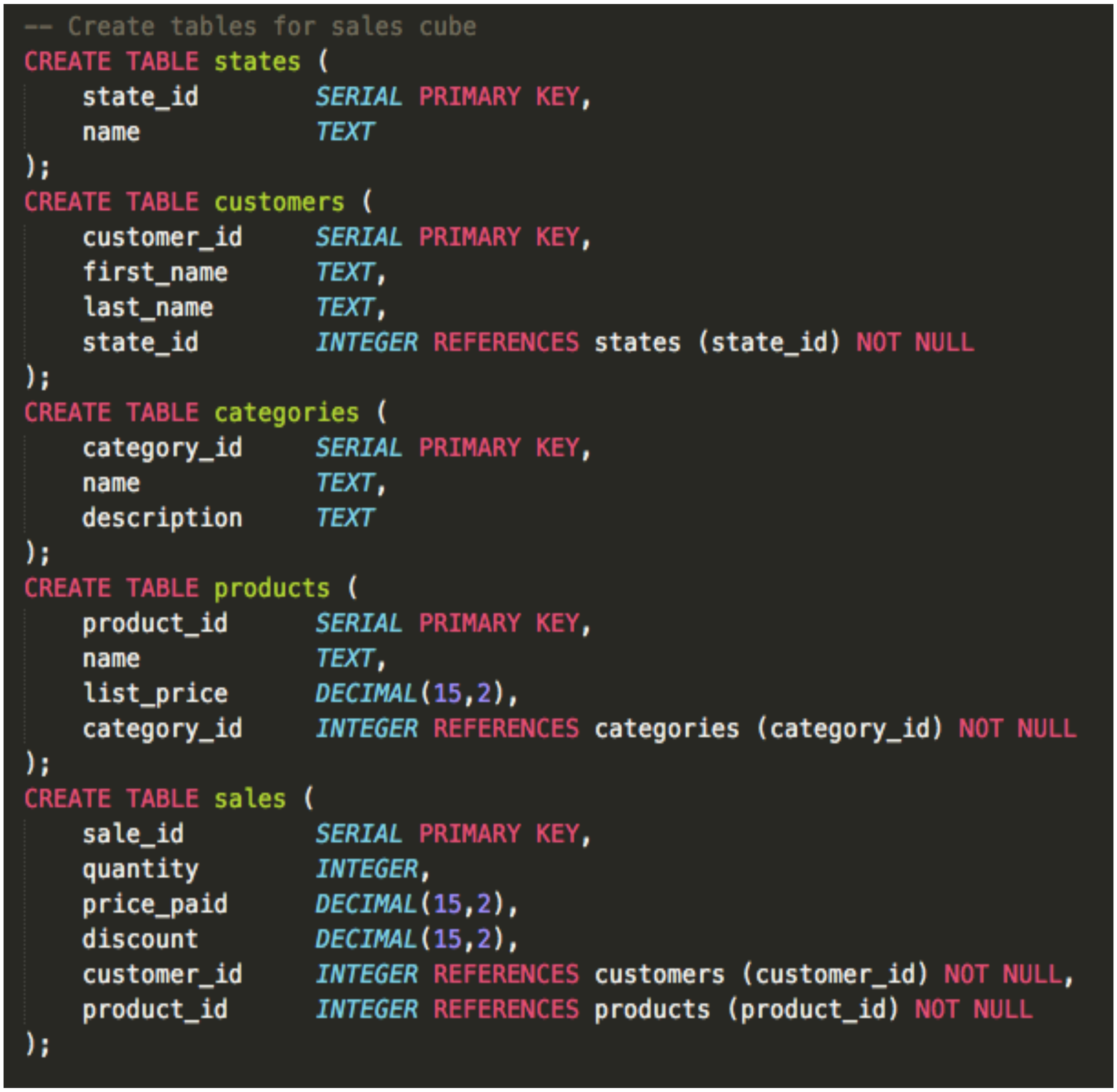
Consider a database that captures customers with a name and state (of residence); states with a name; products that have a list price, a name and belong to a category; categories have names and descriptions; sales of a product to a customer capturing quantity and price paid (may be discounted on a case-by-case basis).

Produce an SQL schema that captures the above information.

Sales Cube - Entity-Relationship Model



Sales Cube - SQL schema



```sql
-- Create tables for sales cube
CREATE TABLE states (
    state_id        SERIAL PRIMARY KEY,
    name            TEXT
);
CREATE TABLE customers (
    customer_id     SERIAL PRIMARY KEY,
    first_name      TEXT,
    last_name       TEXT,
    state_id        INTEGER REFERENCES states (state_id) NOT NULL
);
CREATE TABLE categories (
    category_id     SERIAL PRIMARY KEY,
    name            TEXT,
    description     TEXT
);
CREATE TABLE products (
    product_id      SERIAL PRIMARY KEY,
    name            TEXT,
    list_price      DECIMAL(15,2),
    category_id     INTEGER REFERENCES categories (category_id) NOT NULL
);
CREATE TABLE sales (
    sale_id         SERIAL PRIMARY KEY,
    quantity        INTEGER,
    price_paid      DECIMAL(15,2),
    discount        DECIMAL(15,2),
    customer_id     INTEGER REFERENCES customers (customer_id) NOT NULL,
    product_id      INTEGER REFERENCES products (product_id) NOT NULL
);
```

MAS DSE 201 Homework: 201Cats

Milestone I [due Jan 28 at midnight]

The 201Cats web application provides sophisticated cat video viewing to its users. Each user has a user name and logs in the 201Cats using his Facebook log-in. Consequently, the company regularly obtains information of which ones of the 201Cats users are Facebook followers of other 201Cats users.

When a user logs in, the web application suggests to her 10 cat videos – more on this below. The user may

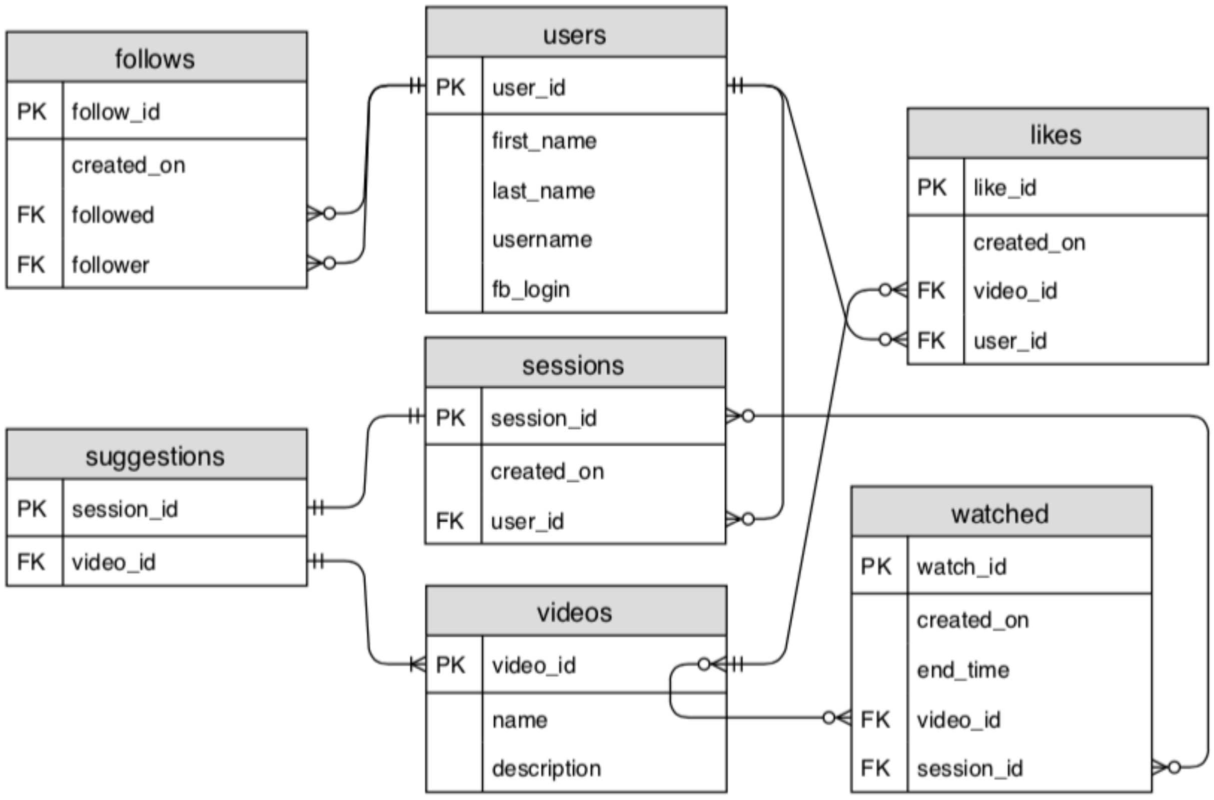
- *Watch one of the suggested videos*
- *Like a suggested video; may like a video even without watching it. A user may like a video just once. Clicking many times on the like does not result on "liking many times".*

The 201Cats database captures the following information, with minimum redundancy:

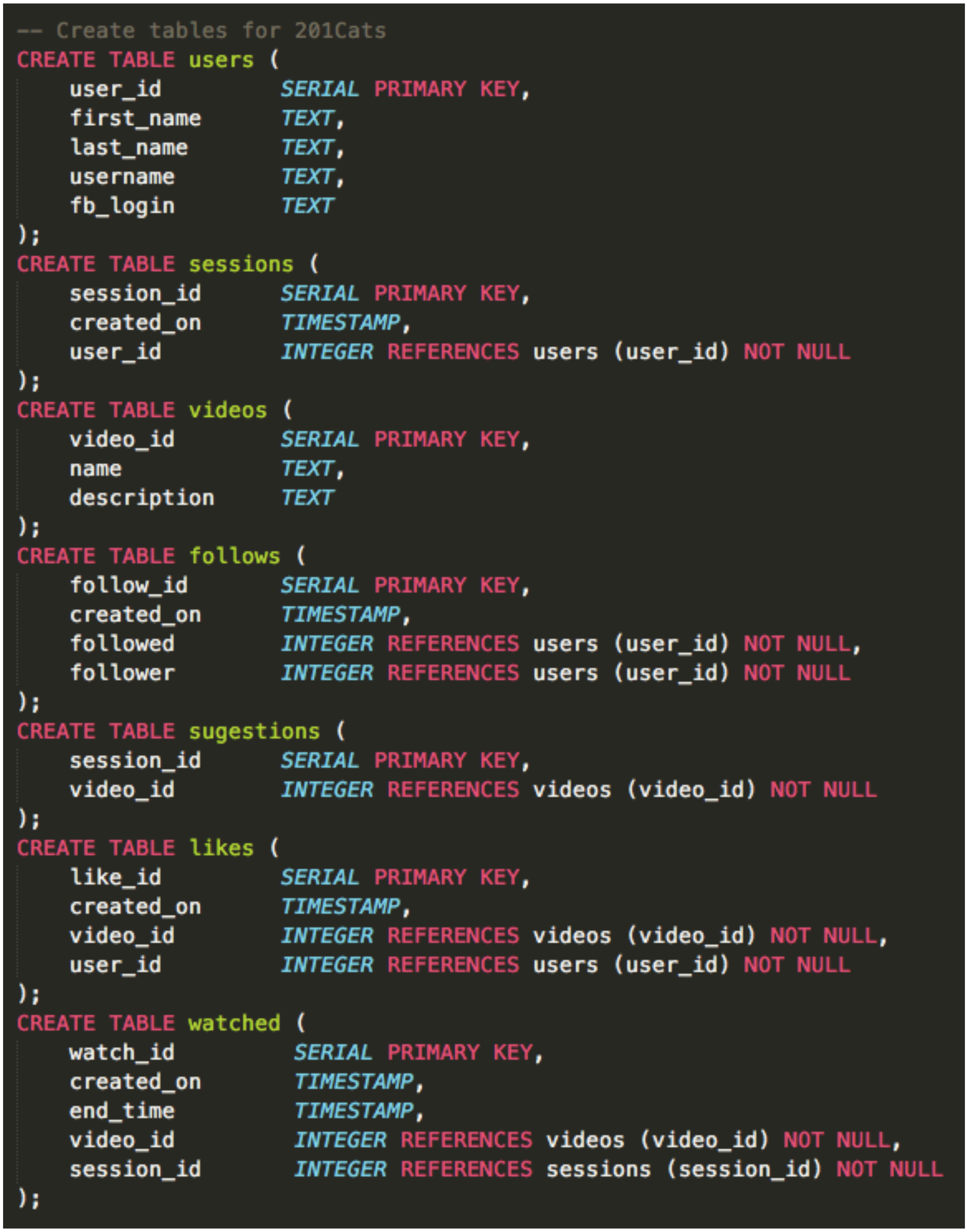
- *The user's name and Facebook login – password not needed.*
- *The user's "like" activity: store which video were liked and when.*
- *The user's "watch" activity: store which videos were watched and when.*
- *The times the user logged in and the videos that were suggested to the user to watch when she logged in.*
- *Which 201Cats users are friends of each user. You are allowed some redundancy here: It is OK if the database captures both that "X is friend of Y" and "Y is friend of X", despite the fact that this is redundant since Facebook friendships are symmetric.*

Produce an SQL schema that captures the above information. Optionally (and not graded), submit the corresponding E/R design – if you designed the schema using the E/R technique (something we recommend highly).

201Cats - Entity-Relationship Model



201Cats - SQL schema



```sql
-- Create tables for 201Cats
CREATE TABLE users (
    user_id         SERIAL PRIMARY KEY,
    first_name      TEXT,
    last_name       TEXT,
    username        TEXT,
    fb_login        TEXT
);
CREATE TABLE sessions (
    session_id      SERIAL PRIMARY KEY,
    created_on      TIMESTAMP,
    user_id         INTEGER REFERENCES users (user_id) NOT NULL
);
CREATE TABLE videos (
    video_id        SERIAL PRIMARY KEY,
    name            TEXT,
    description     TEXT
);
CREATE TABLE follows (
    follow_id       SERIAL PRIMARY KEY,
    created_on      TIMESTAMP,
    followed        INTEGER REFERENCES users (user_id) NOT NULL,
    follower        INTEGER REFERENCES users (user_id) NOT NULL
);
CREATE TABLE sugestions (
    session_id      SERIAL PRIMARY KEY,
    video_id        INTEGER REFERENCES videos (video_id) NOT NULL
);
CREATE TABLE likes (
    like_id         SERIAL PRIMARY KEY,
    created_on      TIMESTAMP,
    video_id        INTEGER REFERENCES videos (video_id) NOT NULL,
    user_id         INTEGER REFERENCES users (user_id) NOT NULL
);
CREATE TABLE watched (
    watch_id         SERIAL PRIMARY KEY,
    created_on       TIMESTAMP,
    end_time         TIMESTAMP,
    video_id         INTEGER REFERENCES videos (video_id) NOT NULL,
    session_id       INTEGER REFERENCES sessions (session_id) NOT NULL
);
```