CSI2132[A]

Hotel Project Deliverable 2 Group 19

Mustafa Ahmed 300242013| Ashvin Ramanathan 300242541| Connor States 300254333

March 31, 2023

DBMS Description:

We used a handful of technologies to design our DBMS. Firstly, our database for storing data was designed using PostgreSQL. The database was setup using PGAdmin4. Secondly the backend of our DBMS was programmed with Java using Apache Tomcat to communicate with the database. The full backend was designed using Intellij. The webpages themselves were designed with HTML. However the HTML files were converted to jsp files to bridge the connection between the database and the webpages.

Program Requirements:

To run the program you will need PGAdmin4 to build and run the PostgreSQL database. For the backend of the application, you will need to have java installed. You will also need Intellij installed in order to use Apache Tomcat. In Intellij the SmartTomCat plugin is required. No internet connection is required at this time because both our database and website are locally hosted. With more time, we would have hosted the app as a live service.

Changes Since Deliverable 1:

Our database has remained largely the same since deliverable 1. The key changes are to the *rents* table to include a customer’s credit card so that it is possible for an employee to approve a booking. There is now also an extra column in the *works\_at* table called position\_title to keep a track of each employees job when searching for them.

PosgreSQL Views:

Create view hotel\_rooms AS select contact\_phone, Count(\*) as num\_rooms from has Group by contact\_phone

Create view room\_by\_city AS Select city, SUM(num\_of\_rooms) AS total\_rooms from hotels group by city

PostgreSQL Triggers Used:

CREATE FUNCTION remove\_works\_at() RETURNS TRIGGER AS $$ BEGIN DELETE FROM works\_at WHERE emp\_num = OLD.emp\_num; RETURN OLD; END; $$ Language plpgsql;

CREATE trigger terminate\_employee BEFORE DELETE ON employees FOR EACH ROW EXECUTE FUNCTION remove\_works\_at();

CREATE trigger terminate\_employee BEFORE DELETE ON employees FOR EACH ROW EXECUTE FUNCTION remove\_works\_at();

CREATE FUNCTION remove\_rents() RETURNS TRIGGER AS $$ BEGIN DELETE FROM rents WHERE sin = OLD.sin; RETURN OLD; END; $$ Language plpgsql;

CREATE trigger cancel\_rents BEFORE DELETE ON customers FOR EACH ROW EXECUTE FUNCTION remove\_rents();

Sample Queries:

DELETE FROM customers WHERE sin = "sin" (java variable)

DELETE FROM employees WHERE emp\_num = "emp\_num" (java variable)

UPDATE employees set "column"(java variable) = "input"(java variable) WHERE emp\_num = "emp\_num" (java variable)

UPDATE works\_at set contact\_phone ="contact\_phone"(java variable) , chain\_name = "chain\_name"(java variable), position\_title = "title"(java variable) WHERE emp\_num = "emp\_num" (java variable)

Code to Build the Database Tables:

PLEASE NOTE: It is not ideal to copy the code from this file to build the tables due to inconsistencies when copy/pasting text from pdf files. Please refer to the SQL and txt files with the rest of our code to build the tables.

CREATE TABLE employees(

emp\_num SERIAL PRIMARY KEY,

sin VARCHAR,

family\_name VARCHAR,

given\_name VARCHAR,

address VARCHAR,

manager\_id SERIAL

);

CREATE TABLE hotel\_chains(

chain\_name VARCHAR PRIMARY KEY,

office\_location VARCHAR,

num\_hotels INTEGER,

phone\_numbers VARCHAR

);

CREATE TABLE hotels(

contact\_phone VARCHAR PRIMARY KEY,

contact\_email VARCHAR,

rating DOUBLE PRECISION,

address VARCHAR,

num\_of\_rooms INTEGER,

manager\_id INTEGER,

city VARCHAR,

hotel\_id SERIAL

);

CREATE TABLE rooms(

room\_id SERIAL PRIMARY KEY,

room\_num INTEGER,

price DOUBLE PRECISION,

amenities VARCHAR,

capacity VARCHAR,

sea\_view BOOLEAN,

is\_extendable BOOLEAN,

problems VARCHAR

);

CREATE TABLE customers(

sin VARCHAR PRIMARY KEY,

family\_name VARCHAR,

given\_name VARCHAR,

address varchar,

email varchar,

registration\_date DATE

);

CREATE TABLE works\_at(

emp\_num INTEGER REFERENCES employees(emp\_num),

contact\_phone VARCHAR REFERENCES hotels(contact\_phone),

chain\_name VARCHAR REFERENCES hotel\_chains(chain\_name),

position\_title VARCHAR

);

CREATE TABLE belongs\_to(

contact\_phone VARCHAR REFERENCES hotels(contact\_phone),

chain\_name VARCHAR REFERENCES hotel\_chains(chain\_name)

);

CREATE TABLE has(

contact\_phone VARCHAR REFERENCES hotels(contact\_phone),

room\_id INTEGER REFERENCES rooms(room\_id)

);

CREATE TABLE rents(

sin VARCHAR REFERENCES customers(sin),

room\_id INTEGER REFERENCES rooms(room\_id),

approved BOOLEAN,

credit\_card VARCHAR,

start\_date DATE,

end\_date DATE

);

CREATE FUNCTION remove\_works\_at()

RETURNS TRIGGER AS $$

BEGIN

DELETE FROM works\_at WHERE emp\_num = OLD.emp\_num;

RETURN OLD;

END;

$$ Language plpgsql;

CREATE trigger terminate\_employee BEFORE DELETE ON employees FOR EACH ROW EXECUTE FUNCTION remove\_works\_at();

CREATE FUNCTION remove\_rents()

RETURNS TRIGGER AS $$

BEGIN

DELETE FROM rents WHERE sin = OLD.sin;

RETURN OLD;

END;

$$ Language plpgsql;

CREATE trigger cancel\_rents BEFORE DELETE ON customers FOR EACH ROW EXECUTE FUNCTION remove\_rents();

Create view hotel\_rooms AS select contact\_phone, Count(\*) as num\_rooms from has Group by contact\_phone;

Create view room\_by\_city AS Select city, SUM(num\_of\_rooms) AS total\_rooms from hotels group by city;

CREATE INDEX employees\_sin ON employees(sin);

CREATE INDEX hotels\_phone ON hotels(contact\_phone);

CREATE INDEX has\_phone\_and\_id ON has(contact\_phone, room\_id);