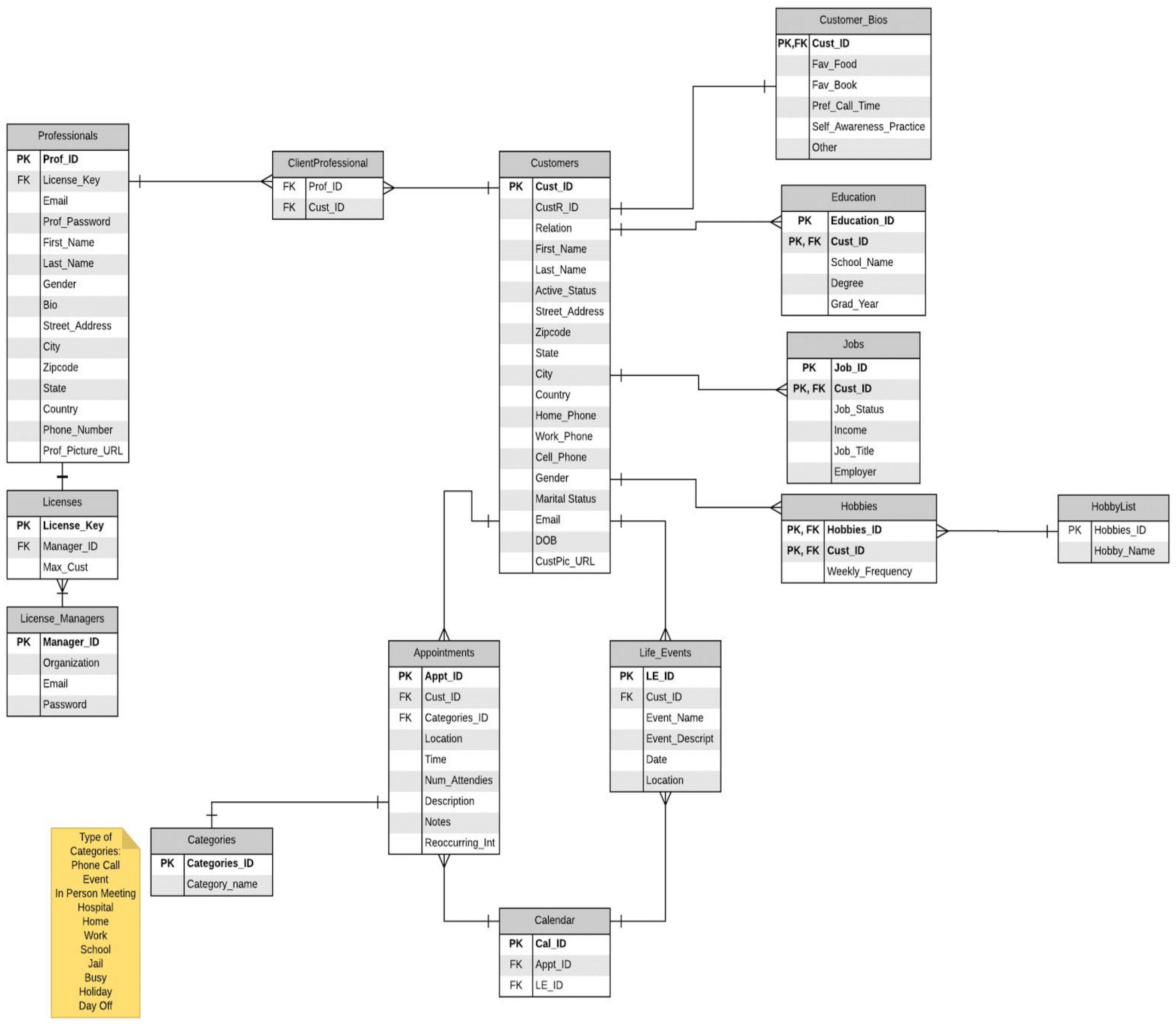
Salty Groundhogs November 2, 2016

ER Diagram & Documentation

In this document we, The Salty Groundhogs, have broken down our ER diagram into a universally communicable explanation of our tables and relationships.

ER Diagram



Professionals Table

The *Professionals Table* stores all the data for the professionals. We have chosen "Prof_ID" to be our primary key so we can uniquely identify all professionals in the table. There is a foreign key, "License_Key", which references the *Licenses Table*, and is used to enforce that all professionals on True Course have paid and are legitimate users. The *Licenses Table* and the *Professionals Table* share a one-to-one relationship which means there is one license for one professional and one professional per one license. The *Professionals Table* and the *Customers Table* share a many-to-many Weak Entity relationship from the professional table to the *ClientProfessional* Table to the *Customers Table*. This relationship allows one professional to have many customers and one customer to have many professionals. This way professionals can share their clients with other professionals without violating atomicity.

<u>Columns</u>

FK License Key:References the License Key in the License Table in order to enforce that all professional on True Course have paid and are legitimate users.
Email: Stores the Professional's email.
Prof_Password: Stores the Professional's SHA-2 encrypted password
Bio: Stores the Professional's biography information
Prof_Picture_URL: Stores the Professional's Profile Picture URL in order to reference the location of the Profile Picture in the File Server, where the pictures are stored.

Licenses Table

The *Licenses Table* stores all the data for user access licenses. We have chosen "License_Key" to be our primary key so we can uniquely identify all licenses that True Course gives out. The *License Table* and the *Professional Table* share a one-to-one relationship, which means there is one license per one professional and one professional per one license. There is a foreign key, "Manager_ID", which references the *License_Managers Table*, and is used to enable an organization to distribute license keys to their professionals.

<u>Columns</u>

Max_Cust: Stores the amount of clients a single Professional can have under their license key.

Customers Table
The *Customers Table* stores all the data on True Course customers. We have chosen "Cust_ID" to be our primary key so we can uniquely identify all customers in the table. The *Customers Table* and the *Professionals Table* share a many-to-many relationship through the *ClientProfessional Table*. This relationship allows one professional to have many customers, and one customer to have many professionals, we have implemented this feature in order for professionals to share their customers with other professionals. The *Customers Table* and the *Customer_Bios Table* share a one-to-one relationship, which enforces that one customer has one bio. The *Customers Table* and the *Education Table* share a one-to-many relationship, originating from the *Customers Table*, in order to connect Customers to the many educations they have, or have had in the past. The *Customers Table* and the *Jobs Table* share a one-to-many relationship, originating from the *Customers Table*, in order to connect Customers to the many Jobs they have, or have had in the past. The *Customers Table* and the *HobbyList Table* share a many-to-many weak entity, *Hobbies Table*. This relationship allows one customer to have multiple hobbies, and one hobby to have multiple customers. The *Customers Table* and the *Life_Events Table* share a one-to-many relationship, from the *Customers Table* side, which allows for a customer to have many life events. The *Customer Table* and the *Appointments Table* share a one-to-many relationship, from the *Customer Table* side, which allows one customer to have many appointments.

<u>Columns</u>
CustR_ID: stores the Cust_Id for a relative of the customer. This allows True Course to understand and keep track of the customers family tree.
Relation: stores the relationship of the family member, relative to the customer.
Active_Status: stores the account status of the customer. If they are an active customer for the professional their status will be active, and if not it will be inactive
DOB: stores the customer's date of birth
CustPic_URL: Stores the customer's Profile Picture URL in order to reference the location of the Profile Picture in the File Server, where the pictures are stored.

Customer_Bios Table
The *Customer_Bios Table* stores all of the customers biography data that is relevant to True Course. We have chosen "Cust_ID" to be the primary key for the *Customer_Bios Table*, which references the *Customers Table*, and is used to connect customer bios to their respective customers and to enforce one customer has one customer bio. The *Customers Table* and *Customer_Bios Table* share a one-to-one relationship, which enforces that one customer has one bio.

<u>Columns</u>
Pref_Call_Time: stores the customer's preferred time to be reached by telephone data.
Self_Awareness_Practice: stores the customer's self awareness practice details data.

Life_Events Table
The *Life_Events Table* stores all the data for customer life events. We have chosen "LE_ID" to be our primary key so all life events can be uniquely identified. There is a foreign key "Cust_ID", which references the *Customers Table*, and is used to connect customers to their life events. There is a one-to-many relationship between the *Customers Table* and the *Life_Events Table*, from the *Customers Table* side, which allows one customer to have many life events. There is a one-to-many relationship between the *Calendar Table* and the *Life_Events Table*, from the *Calendar Table* side, which allows the calendar to store many life events.

<u>Columns</u>
Event_Name: stores the name of the customer's life event.
Event_Description: stores the description of the customer's life event.
Date: stores the date of the customer's life event.
Location: stores the location of the customer's life event.

Appointments Table
The *Appointments Table* stores all the data for customer appointments. We have chosen "Appt_ID' to be our primary key so all appointments can be uniquely identified. There is a foreign key "Cust_ID", which references the *Customers Table*, and is used to connect customers to their appointments. There is a foreign key "Categories_ID", which references the *Categories Table*, and is used to connect appointments to their specific category of appointment. There is a one-to-many relationship between the *Customers Table* and the *Appointments Table*, from the *Customers Table* side, which allows one customer to schedule many appointments. There is a one-to-one relationship between the *Appointments Table* and the *Categories Table*, which allows one appointment to have one category. There is a one-to-many relationship between the *Calendar Table* and the *Appointments Table*, from the *Calendar Table* side, which allows a calendar to store many appointments.

Columns
Location: stores the location of the customer's appointment.
Time: stores the time of the customer's appointment.
Num_Attendies: stores the number of people who attended the customer's appointment.
Description: stores the description of the customer's appointment
Reoccuring_Int: stores the recurrence of the customer's appointment.

Calendar Table
The *Calendar Table* stores all data for managing customer calendars. We have chosen "Cal_ID" to be our primary key so all calendar events can be uniquely identified. There is a foreign key "Appt_ID", which references the *Appointments Table*, and is used to connect appointments to the calendar. There is a one-to-many relationship between the *Calendar Table* and the *Appointment Table*, from the *Calendar Table* side, which allows for a calendar to store many appointments. There is a foreign key "LE_ID', which references the *Life Events Table*, and is used to connect life events to the calendar. There is a one-to-many relationship between the *Calendar Table* and the *Life Events Table*, from the *Calendar Table* side, which allows for a calendar to store many life events.

Jobs Table
The *Jobs Table* stores all the data for the customer's jobs. We have chosen "Job_ID" to be our primary key so all jobs can be uniquely identified. There is a foreign key "Cust_ID", which references the *Customers Table*, and is used to connect customers to their jobs. There is a one-to-many relationship between the *Jobs Table* and the *Customers Table*, from the *Customers Table* side, which allows for one customer to have many jobs.

Columns
Job_Status: stores the status of the customer's job: past, current, retired.

Education Table
The *Education Table* stores all the data for the customer's education. We have chosen "Education_ID" to be our primary key so all educations can be uniquely identified. There is a foreign key "Cust_ID", which references the *Customers Table*, and is used to connect customers to their education. There is a one-to-many relationship between the *Education Table* and the *Customers Table*, from the *Customers Table* side, which allows for one customer to have many educations.

Columns
School_Name: stores the name of the school the customer's education comes from.
Degree: stores the title of the degree the customer received.

Hobbies Table
The *Hobbies Table* stores all the data for the customer's hobbies. There is a Primary/Foreign key "Cust_ID", which references the *Customers Table*, and is used to uniquely identify every customer in the *Hobbies Table*. There is a Primary/Foreign key "Hobbies_ID", which references the *HobbyList Table*, and is used to uniquely identify every Hobby in the *Hobbies Table*. The composite primary key("Cust_ID","Hobbies_ID") enforces that one customer cannot have duplicate hobbies. The *Hobbies Table* is a many-to-many weak entity between the *Customers Table* and the *HobbyList Table*. This table allows for one customer to have many hobbies and one hobby to have many customers.

Columns
Weekly_Frequency: stores the frequency in which a customer does a specific hobby per week.

HobbyList Table
The *HobbyList Table* stores all the data for the various hobbies a customer can have. There is a primary key “Hobbies_ID”, which uniquely identifies all the hobbies in the *HobbyList Table*. There is a one-to-many relationship between the *HobbyList Table* and the weak entity *Hobbies Table*, originating from the *HobbyList Table* side, which allows for one hobby to have many customers.

<u>Columns</u>
Hobby_Name: stores the name of the hobby.

Categories Table
The *Categories Table* stores all the data for the various categories an appointment can have. There is a primary key “Categories_ID”, which uniquely identifies all the categories in the *Categories Table*. There is a one-to-many relationship between the *Categories Table* and the *Appointments Table*, originating from the *Categories Table* side, which allows for one category to have many appointments

<u>Columns</u>
Category_Name: stores the name of the category.

ClientProfessional Table
The *ClientProfessional Table* manages the many-to-many relationship between the *Professional Table* and the *Customers Table*. There is a foreign key “Cust_ID”, which references the *Customers Table*, and is used to identify the customer and their respective professional. There is a foreign key “Prof_ID”, which references the *Professionals Table*, and is used to identify the professional and their respective customers. There is a one-to-many relationship between the *ClientProfessional Table* and the *Professionals Table*, originating from the *Professionals Table*, which allows for one professional to have many clients. There is a one-to-many relationship between the *ClientProfessional Table* and the *Customers Table*, originating from the *Customers Table*, which allows for one customer to have many professionals.

License_Managers Table
The License_Managers table stores all the data for the license managers or organization representatives. There is a primary key “Manager_ID”, which is used to uniquely identify all of the license managers in the *License_Managers Table.* There is a one-to-many relationship between the *License_Manager Table*, originating from the *License_Managers Table*, which allows for one license manager to give out many licenses.

<u>Columns</u>
Organization: stores the name of the license manager’s organization.
Password: Stores the license manager’s SHA-2 encrypted password