**Team Name:** DogDB
**Team Members:** Julie Pražáková & Guilherme S Malachias
**Website:**Web.engr.oregonstate.edu/~seabramg/
**Project title:** Online diary / planner for meetings and contacts records

**Changes made since step 4**

started working on database integration between html and sql code thought node.js
started converting most of website to handlebars
started implementing js code

a) **Overview:**

Our Project idea is a management app for mid-sized companies to manage their employees. The app would do tasks such as:

- Displaying future events or meetings, people attending said events as well as the location, time and date of an event.
- Having individual profiles for employees, where management would be able to search for anyone and show their detailed profiles with entries for things like names, age, gender identity, job and Groups certain employees are part of.
- Have information on an employee's designated area of work as well as the location of events and meetings through the company.
- The ability of designating and contacting a certain group – an employee can be a member of a group depending on their position and department within the company. Because certain people in the company – mainly upper management, may have access to more work groups, we need to create another table covering this M:N relation.

**b) Database Outline:**

- person – *Because this is being used by a company,the job field will always need to be filled in.*
  - id_person: integer, auto_increment, not NULL, PK
  - first_name: varchar(255), not NULL
  - last_name: varchar(100), not NULL
  - id_location: integer, not NULL, FK
  - day_of_birth: date
  - gender: char
  - job: varchar(45), not NULL
  - id_group: integer, not NULL

- event – *Can be saved with only a date and any other information should be Optional.*
  - id_event: integer, auto_increment, not NULL, PK
  - date: date, not NULL
  - description: varchar(200)
  - duration: time
  - id_location: integer, FK

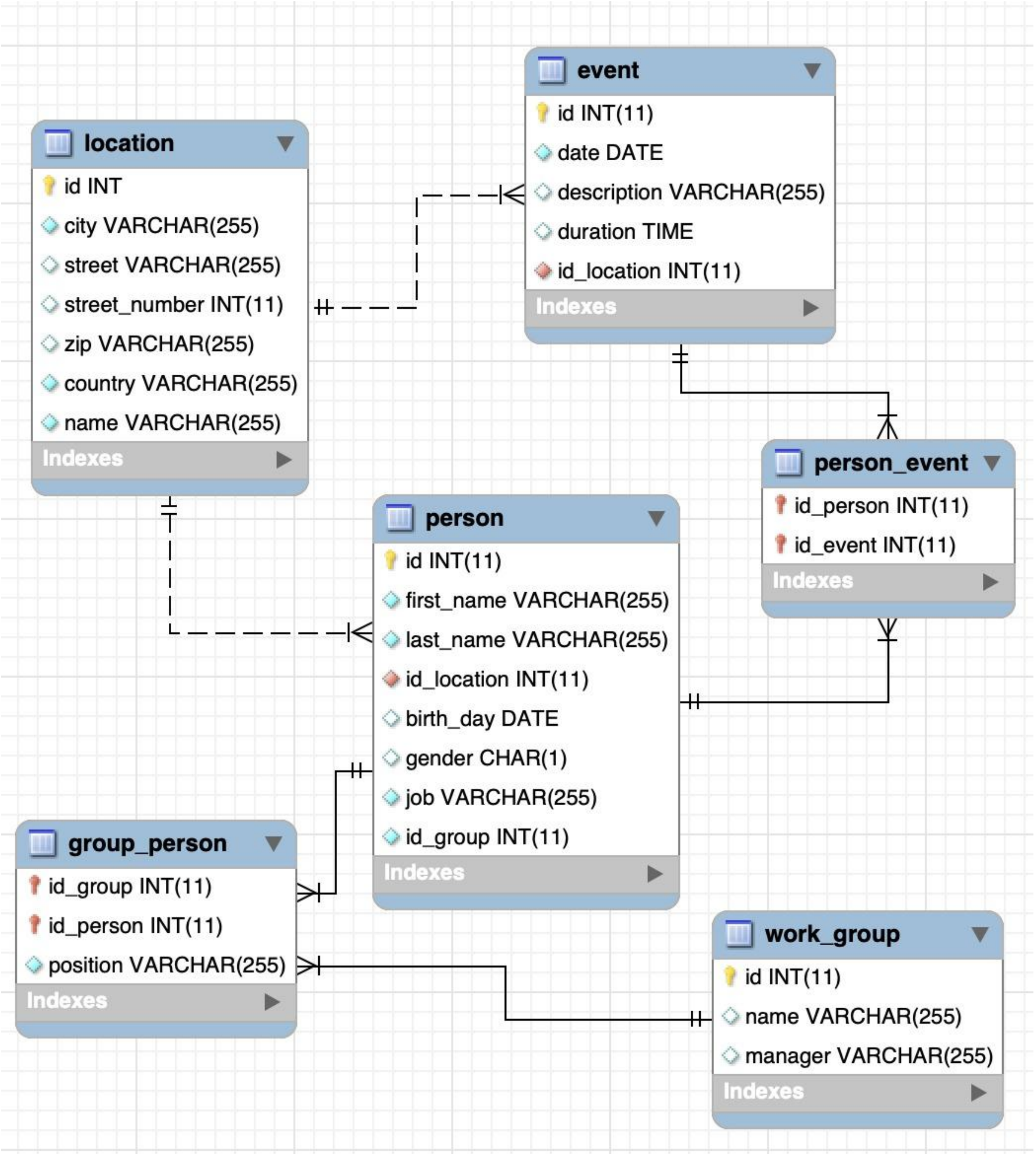
- location – *We can have the option of only making mandatory the country – city – name for* the location and then the rest of the information would be Optional.
    - id_location: integer, auto_increment, not NULL, PK
    - city: varchar(100), not NULL
    - street: varchar(100)
    - street_number: integer
    - zip: varchar(45)
    - country: varchar(100), not NULL
    - name: varchar(100), not NULL

- work_group
  - id: integer(11), auto_increment, not NULL, PK
  - name: varchar(255), not NULL, FK
  - manager: varchar(255), not NULL

Many employees can attend 1 event and one person can attend multiple events – that indicates the many to many relation M:N. For that we will have to apply another table solving this relation. The same goes for any group and person relation.

- person_event
    - id_person: integer, not NULL, PK, FK
    - id_event: integer, not NULL, PK, FK

- group_person
    - id_group: integer, not NULL, PK, FK
    - id_person: integer, not NULL, PK, FK
    - position: varchar(100), not NULL



**person**(id_person, first_name, last_name, id_location, day_of_birth, gender, job, id_group)
id_person: integer(11), auto_increment, not NULL, PK
first_name: varchar(255), not NULL
last_name: varchar(255), not NULL
id_location: integer, not NULL, FK
day_of_birth: date
gender: varchar(255),not NULL
job: varchar(255), not NULL
Id_group: integer(11), not NULL

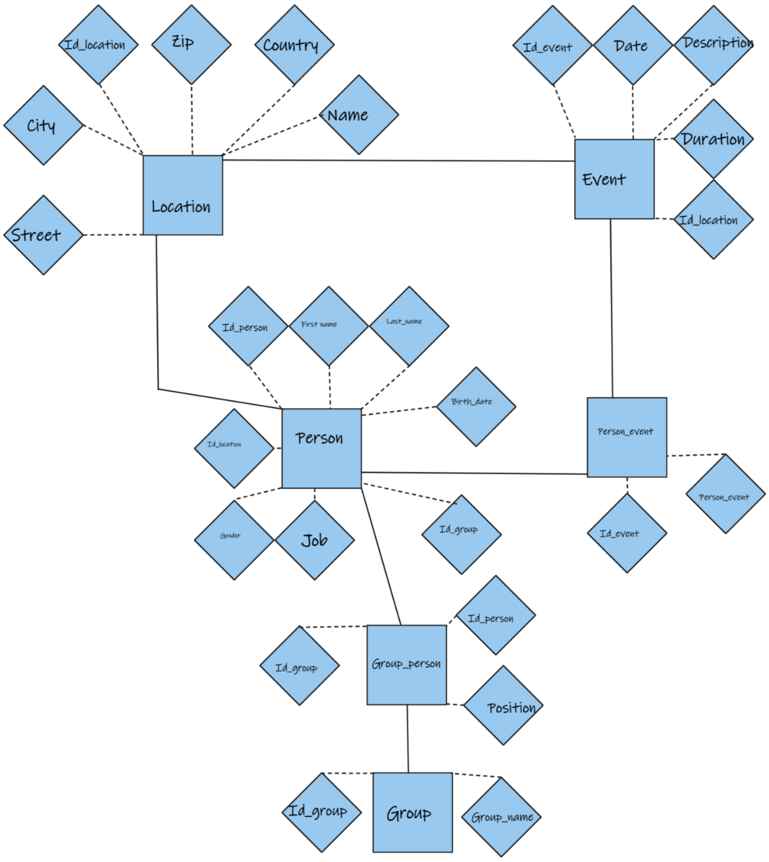
**event**(Id_event, date, description, duration, id_location)
id_event: integer, auto_increment, not NULL, PK
date: date, not NULL
description: varchar(255)
duration: time
id_location: integer(11), FK

**location**(id_location, city, street, street_number, zip, country, name)
Id_location: integer, auto_increment, not NULL, PK
city: varchar(255), not NULL
street: varchar(255)
street_number: integer(11)
zip: varchar(255)
country: varchar(255), not NULL
name: varchar(255), not NULL

**work_group**(id_group, id_person, group_name)
id: integer(11), auto_increment, not NULL, PK
name: varchar(255), not NULL, FK
manager: varchar(255), not NULL

**person_event**(id_person, id_event)
id_person: integer(11), not NULL, PK, FK
id_event: integer(11), not NULL, PK, FK

**group_person**(Id_group, position, id_person)
id_group: integer(11), not NULL, PK, FK
id_person: integer(11), not NULL, PK, FK
position: varchar(255), not NULL



## Feedback and Actions taken.

All the feedback given by our peers was of extreme value and we analyzed and considered many of their great ideas and suggestions when updating our HTML website . We concatenated the best pieces of their feedback into our latest version of DogDb , we added add buttons to every page for adding new tables to the database ,as well as improved on the design and layout and gave our website a complete rework when it came to looks and aesthetics .Functionality is still being implemented slowly but their feedback gave us ideas to what are they current key features that DogDb is lacking and what steps we can take to design and implement those features such as input boxes and buttons ,as well as setting up the foundation to our backend by implementing new functions (updated from the ER diagram) to further improve our project .