Design Document

Requirements

1. GUI for SQL database

- ☒ DB Creation (CLI allowed)
- ☒ DB Population (CLI allowed)
- ☒ Creation - add new contacts via a contact entry form.
- ☒ Retrieval
  - Search on multiple fields - name, address or phone
  - Full Name displayed, direct access to modify or delete
  - All the contacts simultaneously
- ☒ Update + Deletion
  - Modify contact entry form, and ability to delete fields or entries. ### 2. Backend OPs
- ☒ Schema definition and table creation
- ☒ Population of DB from CSV (deconstruct the file and add the fields to the tables with normalization).
- ☒ Table to language connector - API
- ☒ CRUD on DB
  - SQL command to add a new entry
  - SQL command to modify an entry
    - Change name fields
    - Change address field, add new address or delete address
    - Change phone number field, add new phone number or delete a phone number
    - Change date field, add or remove a date.
  - SQL command to delete an entry
  - SQL command to get all entries
  - SQL command to search all entries and filter.

Design

1. Frontend Views

- [A] Contact Display Window
- [B] Search Window
- [C] New Contact Entry Form
- [D] Modify Contact Entry Form
- [PH] Phone Edit Form
- [AD] Address Edit Form
- [DA] Date Edit Form ### 2. Backend Query Pre-Planning + Frontend Wiring
- [A] Display Window queries
  1. select * from join of all tables of PK==FK
  2. On edit: retrieve single row from table and populate modify contact

form w/ it —> [D]
3. On add new: open new contact entry form, get fields vals & update table w/ entry. —> [C]
4. On search —> [B]

- [B] Search window
  1. Initially show top 5 entries
  2. user inputs comma delimited keywords and hits search
  3. Trigger query with keywords being matched to all fields
  4. For matches in fk tables, search primary table for fk==pk match and display names + names matched in primary.
  5. Edit entry —> [D]
- [C] New Contact Form
  1. Name fields
  2. Add ph. no. —> Adds set of sub-fields to be populated by [PH]
  3. Add. address —> Adds set of sub-fields to be populated by[AD]
  4. Add date —> Adds set of sub-fields to be populated by [DA]
- [D] Modify Contact Entry Form
  1. Pre-poulate [C] with all existing data
  2. Provide name fields, [PH], [AD] & [DA] links.
- [PH] Phone Edit Form
  1. All fields in a single phone number entry
- [AD] Address Edit Form
  1. All fields in a single address entry
- [DA] Date Edit Form
  1. All fields in a single date entry ### 3. Phases

1. UI Wireframe Mockups
2. Schema definition & table creation
3. CSV parsing & table creation <—> flat table entry object propagating to GUI for editing.
4. Query Planning & direct SQLite Testing
5. Backend Query Implementation using ORM
6. Testing Backend API
7. Model definition - typed dictionaries as object models + object relational model combined.
8. Model view interaction - populating a view with its model
9. Contact Form MVC implementation
   - [contact_form(cid_object) —> edited_cid_object]
   - [contact_form(None) —> new_cid_object]
   - Model - view interaction: populating form with existing entry
   - Editing Address/Phone/Date & MVC for sub-views
     - [address(ad_object) —> new_ad_object]
     - [phone(ph_object) —> new_ph_object]
     - [date(date_object) —> new_date_object]
10. Testing object change sanity
11. Controller implementation - manipulating model from view actuated com-

mands
12. View - Controller wiring (user action —> controller —> model changed —> view changed)
13. Search view implementation
14. Testing application