

Modules Project 2(a)

1. `qu_table_create_1`

This has a function `create_table_one(database,num)`. It runs the sql query CREATE TABLE IF NOT EXISTS tablename (

```
Element1_rREAL,
    Element1_i    REAL,
    Element2_r    REAL,
    Element2_i    REAL,
    Element3_r    REAL,
    Element3_i    REAL,
    Element4_r    REAL,
    Element4_i    REAL,
    stringy TEXT UNIQUE);
```

2. `qu_table_create_2`

This is the same as module 1 but it runs the the sql query:

```
CREATE TABLE IF NOT EXISTS outcomes1d (
    stringy TEXT,
    matrices TEXT UNIQUE);
```

3. `qu_table_delete_1`

This has a function `delete_table1(database,num)`. It runs the sql query: DROP TABLE IF EXISTS tablenamenum;

4. `qu_table_delete_2`

This has a function `delete_table2(data_base,num)` and runs the sql query DROP TABLE IF EXISTS tablenamenumd;

5. `three_matrices_data`

This has a function `three_matrices(data_base)` similar to project 1 but has four destinations for the two sets of data outcomes1 and master_list and outcomes1d and master_listd. It imports the library module `sow_data_1` and `sow_data_2`, see later entries. It converts each of the h,t and s matrices into a data string (data = [(a,b,c,d,e,f,g,h)])

6. `reap1`

This has function *reap_data1_outcomes(data_base,num)* and runs the sql query SELECT Element1_r, Element1_i, Element2_r, Element2_i, Element3_r, Element3_i, Element4_r, Element4_i FROM tablenamenum;

7. reap1o_m

This has a function name is *reap_data1_outcomes(data_base,num)* it runt the sql query "SELECT * FROM tablenamenum;"

8. dot_prod_new

dot_product("qd1",i) - see separate document on this module

9. del_intersection

This has the function *d_inter(data_base,num)* and runs the following queries:

```
del_intersection.py - C:\Users\stall\Documents\database working files_nov\object_1_creating a db\project2\new_approach_new\lib1_backup\del_intersection.py (3.8.2)
File Edit Format Run Options Window Help
# delete dups in outcomes table
def d_inter(data_base,num):
    import sqlite3 as lite
    filename = str('%s\data_base'+str('.db'))
    conn = lite.connect(filename)
    cur = conn.cursor()
    if num == 1:
        cur.execute("""SELECT Element1_i || Element1_r || Element2_i || Element2_r || Element3_i || Element3_r || Element4_i || Element4_r FROM outcomes1
INTERSECT SELECT Element1_i || Element1_r || Element2_i || Element2_r || Element3_i || Element3_r || Element4_i || Element4_r FROM master_list; """)
        final_result = cur.fetchall()

    def deleteSqliteRecord(element):
        element = str(element)
        element = element.strip()
        element = element.replace(",","")
        element = element.replace("(","")
        element = element.replace(")","")
        element = element.replace(" ","")
        #print(element)

        sql_update_query = """DELETE from outcomes1 where Element1_i || Element1_r || Element2_i || Element2_r || Element3_i || Element3_r || Element4_i || Element4_r = ?"""
        cur.execute(sql_update_query, (element, ))
        conn.commit()
    for each in final_result:
        deleteSqliteRecord(each)
```

It is the same as del_intersection_t in project 1

10. sow_data1

This has the function *sow_data1(data_base,my_matrix,num)* and runs the query

"INSERT OR IGNORE INTO outcomes1 VALUES(?,?,?,?,?,?,?,?);",my_matrix

The ignore command in this is an improvement on the previous sow module used in Project 1.

not used

1. reformat_array_m
2. del_duplicates_t1 del_dup1(data_base,num)