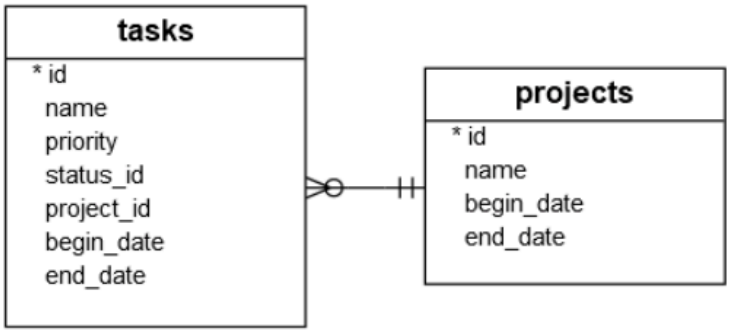
דוגמה ליצירת בסיס נתונים בעזרת sqlite3 בסקריפט של פייתון

For the demonstration, we will create two tables: `projects` and `tasks` as shown in the following database diagram:




```
import sqlite3
from sqlite3 import Error


def create_connection(db_file):
    """ create a database connection to the SQLite database
        specified by db_file
    :param db_file: database file
    :return: Connection object or None
    """
    conn = None
    try:
        conn = sqlite3.connect(db_file)
        return conn
    except Error as e:
        print(e)

    return conn


def create_table(conn, create_table_sql):
```

```
    """ create a table from the create_table_sql statement
    :param conn: Connection object
    :param create_table_sql: a CREATE TABLE statement
    :return:
    """
    try:
        c = conn.cursor()
        c.execute(create_table_sql)
    except Error as e:
        print(e)

def create_project(conn, project):
    """
    Create a new project into the projects table
    :param conn:
    :param project:
    :return: project id
    """
    sql = ''' INSERT INTO projects(name,begin_date,end_date)
              VALUES(?,?,?) '''
    cur = conn.cursor()
    cur.execute(sql, project)
    return cur.lastrowid


def create_task(conn, task):
    """
    Create a new task
    :param conn:
    :param task:
    :return:
    """

    sql = ''' INSERT INTO
tasks(name,priority,status_id,project_id,begin_date,end_date)
              VALUES(?,?,?,?,?,?) '''
    cur = conn.cursor()
    cur.execute(sql, task)
    return cur.lastrowid


def select_all_tasks(conn):
    """
    Query all rows in the tasks table
    :param conn: the Connection object
    :return:
    """
    cur = conn.cursor()
```

```
    cur.execute("SELECT * FROM tasks")

    rows = cur.fetchall()

    for row in rows:
        print(row)


def select_task_by_priority(conn, priority):
    """
    Query tasks by priority
    :param conn: the Connection object
    :param priority:
    :return:
    """
    cur = conn.cursor()
    cur.execute("SELECT * FROM tasks WHERE priority=?", (priority,))

    rows = cur.fetchall()

    for row in rows:
        print(row)


def main():
    database = r"newsqlite.db"

    sql_create_projects_table = """ CREATE TABLE IF NOT EXISTS projects (
                                        id integer PRIMARY KEY,
                                        name text NOT NULL,
                                        begin_date text,
                                        end_date text
                                    ); """

    sql_create_tasks_table = """CREATE TABLE IF NOT EXISTS tasks (
                                    id integer PRIMARY KEY,
                                    name text NOT NULL,
                                    priority integer,
                                    status_id integer NOT NULL,
                                    project_id integer NOT NULL,
                                    begin_date text NOT NULL,
                                    end_date text NOT NULL,
                                    FOREIGN KEY (project_id) REFERENCES projects (id)
                                );"""

    # create a database connection
    conn = create_connection(database)
```

```
    # create tables
    if conn is not None:
        # create projects table
        create_table(conn, sql_create_projects_table)

        # create tasks table
        create_table(conn, sql_create_tasks_table)
    else:
        print("Error! cannot create the database connection.")

    with conn:
        # create a new project
        project = ('Cool App with SQLite & Python', '2015-01-01', '2015-01-30');
        project_id = create_project(conn, project)

        # tasks
      task_1 = ('Analyze the requirements of the app', 1, 1, project_id, '2015-01-01', '2015-01-02')
      task_2 = ('Confirm with user about the top requirements', 1, 1, project_id, '2015-01-03', '2015-01-05')

        # create tasks
        create_task(conn, task_1)
        create_task(conn, task_2)

    with conn:
        print("1. Query task by priority:")
        select_task_by_priority(conn, 1)

        print('\n\n')

        print("2. Query all tasks")
        select_all_tasks(conn)




if __name__ == '__main__':
    main()
```