MINI PROJECT

(2022-23)

“DBInserter”

Project Report

Institute of Engineering & Technology

Submitted By -

Puroo Kulshrestha (201500535)

Under the Supervision of

Mrs. Madhu

Technical Trainer

Department of Computer Engineering & Applications

Department of Computer Engineering and Applications

GLA University, 17 km. Stone NH#2, Mathura-Delhi Road,

Chaumuha, Mathura – 281406 U.P (India)

Declaration

I/we hereby declare that the work which is being presented in the Bachelor of technology. Project **"DBInserter"**, in partial fulfillment of the requirements for the award of the ***Bachelor of Technology*** in Computer Science and Engineering and submitted to the Department of Computer Engineering and Applications of GLA University, Mathura, is an authentic record of my/our own work carried under the supervision of **Mrs. Madhu, Technical Trainer, Dept. of CEA, GLA University.**

The contents of this project report, in full or in parts, have not been submitted to any other Institute or University for the award of any degree.

Sign: *purookulshrestha*

Name of Candidate: Puroo Kulshrestha

University Roll No.: 201500535

Department of Computer Engineering and Applications

GLA University, 17 km. Stone NH#2, Mathura-Delhi Road,

Chaumuha, Mathura – 281406 U.P (India)

Certificate

This is to certify that the project entitled “DBInserter”, carried out in Mini Project – I Lab, is a bonafide work by Puroo Kulshrestha and is submitted in partial fulfillment of the requirements for the award of the degree Bachelor of Technology (ComputerScience & Engineering).

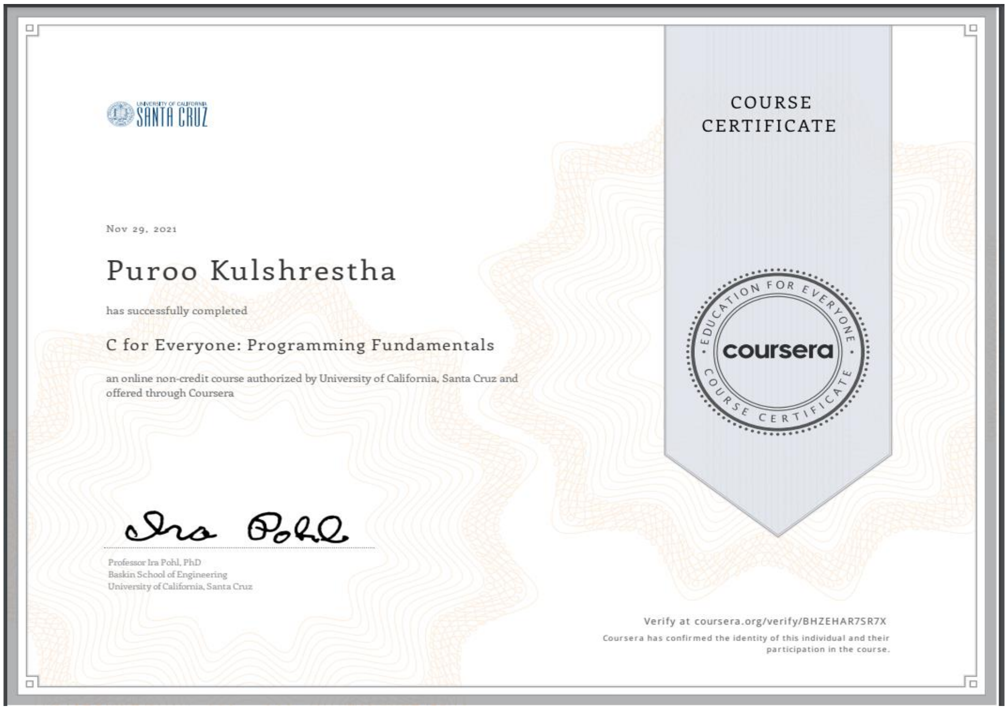
Signature of Supervisor:

Name of Supervisor: Mrs. Madhu

Date:
25-11-2022

Training Certificates

- **Puroo Kulshrestha**



UNIVERSITY OF CALIFORNIA SANTA CRUZ

COURSE CERTIFICATE

Nov 29, 2021

Puroo Kulshrestha

has successfully completed

C for Everyone: Programming Fundamentals

an online non-credit course authorized by University of California, Santa Cruz and offered through Coursera

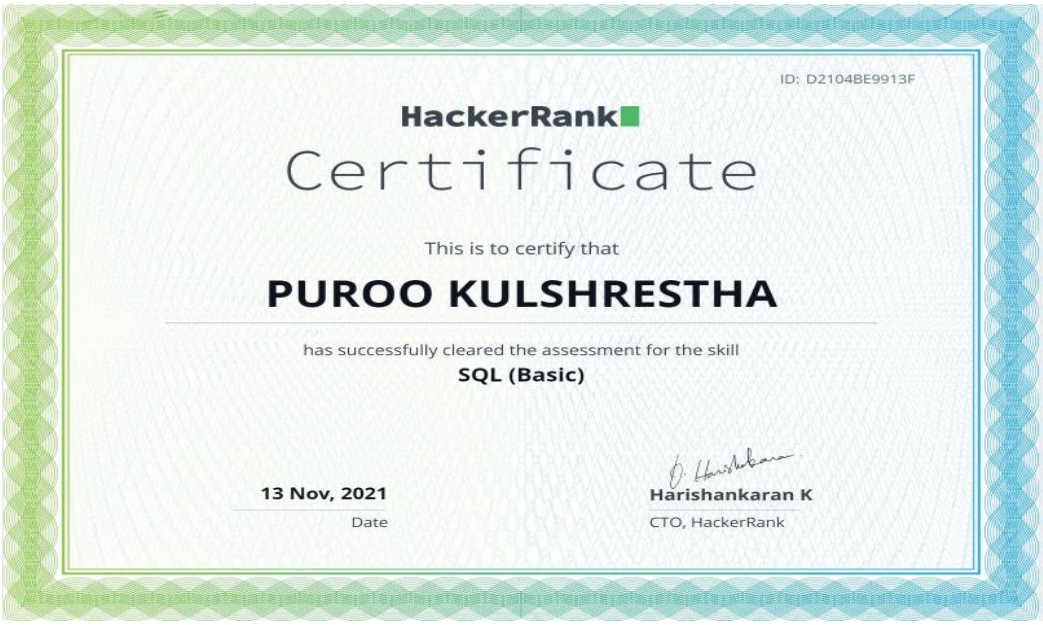
EDUCATION FOR EVERYONE coursera COURSE CERTIFICATE

Ira Pohl

Professor Ira Pohl, PhD
Baskin School of Engineering
University of California, Santa Cruz

Verify at coursera.org/verify/BHZEHAR7SR7X

Coursera has confirmed the identity of this individual and their participation in the course.



ID: D2104BE9913F

HackerRank

Certificate

This is to certify that

PUROO KULSHRESTHA

has successfully cleared the assessment for the skill

SQL (Basic)

13 Nov, 2021
Date

Harishankaran K
CTO, HackerRank

Department of Computer Engineering and Applications

GLA University, 17 km. Stone NH#2, Mathura-Delhi Road,

Chaumuha, Mathura – 281406 U.P (India)

ACKNOWLEDGEMENT

Presenting the ascribed project paper report in this very simple and official form, we would like to place my deep gratitude to GLA University for providing us the instructor Mrs Madhu, our technical trainer and supervisor.

She has been helping us since Day 1 in this project. He provided us with the roadmap, the basic guidelines explaining on how to work on the project. He has been conducting regular meeting to check the progress of the project and providing us with the resources related to the project. Without his help, we wouldn’t have been able to complete this project.

And at last but not the least we would like to thank our dear parents and Seniors for helping us to grab this opportunity to get trained.

Thanking You

Sign: *purookulshrestha*

Name of Candidate: Puroo Kulshrestha

University Roll No.:201500535

ABSTRACT

The emergence of modern technologies has had profound impacts on the education landscape, with online learning now an integral part of the learning process. The main advantages of online learning are flexibility and accessibility (Wu, Tennyson, & Hsia, 2010). Student access to educators to assist them is no longer restricted to the hours of operation of schools and universities, but can be provided anytime and anywhere. Face-to-face tutoring is a well-established, and effective, instructional method. However, there is a need for more empirical research to be directed toward investigating users' experiences with online tutoring services, their impact on academic confidence (self-efficacy), and achievement scores. The purpose of this project is to develop a back-end database for to record all the data in your applications to store in your database and queries using graphical user interface. **Mini-projects** are quick, low stakes and engaging exercises that help us apply course material in a memorable way. The key is to not overcomplicate the exercises or make them too time consuming. DBInserter is implemented in C programming language. DBInserter connects to sqlite database, and it insert the record from the given ".csv" file into Database.

CONTENTS

LIST OF FIGURES

INTRODUCTION

1.1 Context

The database Management system with help of “C” and “SQL” programming languages has been submitted in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering at GLA University, Mathura supervised by Ms. Madhu. This project has been completed approximately three months and has been executed in modules, meetings have been organized to check the progress of the work and for instructions and guidelines.

1.2 Client Identification/Need Identification/Identification of relevant Contemporary issue

Data requirements definition establishes the process used to identify, prioritize, precisely formulate, and validate the data needed to achieve business objectives. When documenting data requirements, data should be referenced in business language, reusing approved standard business terms if available. Businesses are free to keep client records according to their specific needs, so there is not a particular set of information that a company must keep on clients. That said, all businesses should store basic personal information in a database so that they can easily look up client records and identify customers. Names, company names, account numbers, Social Security numbers and client numbers are examples of identifying information that a business can store in a database. Databases should store client usernames and passwords, and passwords should be encrypted so to protect client accounts. A database can also contain information about a client's Web preferences, such as a wish list of goods that a client wants to purchase and preferred shipment and payment methods. Contemporary Issues in Database Design and Information gathers the latest development in the area to make this the most up-to-date reference source for educators and practioners alike. Information systems development activities enable many organizations to effectively compete and innovate, as new database and information systems applications are constantly being developed. Contemporary Issues in Database Design and Information Systems Development presents the latest research ideas and topics on databases and software development.

1.3 Identification of Problem

Problem solving is a process of transforming the description of a problem into the solution of that problem by using our knowledge of the problem domain and by relying on our ability to select and use appropriate problem-solving Strategies, Techniques and Tools. Problem solving (within the context of developing programs) refers to analyzing a problem with the intention of deriving a solution for the problem. The optimum situation is that your developers give you the SQL they're planning to use in custom code (or better yet, they have a dba write it in the first place). More likely, developers will identify a problem area where "the database is slow", and can give you the SQL from that part of their code. I also ask on any new table creation that the developers provide the SQL that they intend to use, so I can index for it ahead of time. I've seen significant site slowdowns caused by no indexes on a custom table. Many of the routines in the SQLite C-language Interface return numeric result codes indicating either success or failure, and in the event of a failure, providing some idea of the cause of the failure. This document strives to explain what each of those numeric result codes means. "Error codes" are a subset of "result codes" that indicate that something has gone wrong. There are only a few non-error result codes: SQLITE_OK, SQLITE_ROW, and SQLITE_DONE. The term "error code" means any result code other than these three.

1.4 Identification of Tasks

Each process is named by a *process ID* number, a value of type pid_t. A process ID is allocated to each process when it is created. Process IDs are reused over time. The lifetime of a process ends when the parent process of the corresponding process waits on the process ID after the process has terminated. The pid_t data type is a signed integer type which is capable of representing a process ID. In the GNU C Library, this is an int. Task analysis is the process of learning about ordinary users by observing them in action to understand in detail how they perform their tasks and achieve their intended goals. Tasks analysis helps identify the tasks that your website and applications must support and can also help you refine or re-define your site's navigation or search by determining the appropriate content scope.

1.5 Timeline

A timeline can really help you out when you need to report “big picture” project information. It’s a professional-looking snapshot of key tasks**.** A visual reference point that allows teams to understand the flow of steps necessary to complete a project on time. To develop fully DBInserter software you will easily took approximately 1-2 month. To learn and gain more knowledge about C programming and database you have to prerequisite basic knowledge. You have install some sqlite libraries for C to understand**.** To create database and give input inside c file. Use of file handling in C.

1.6 SOURCES

The source of our project (including all the project work, documentations and presentations) will is available at the following link https://github.com/purookulsh13/DBInserter .

CHAPTER -2

LITERATURE REVIEW/BACKGROUND STUDY

2.1. Timeline of the reported problem

The first computer database was built in the 1960s, but the history of databases as we know them, really begins in 1970. In 1978, Brian Kernighan and Dennis Ritchie published the first edition of The C Programming Language. This book, known to C programmers as K&R, served for many years as an informal specification of the language. The version of C that it describes is commonly referred to as "K&R C".

2.2. Proposed solutions

The history of databases begins with the two earliest computerised examples. Charles Bachman designed the first computerised database in the early 1960s. This first database was known as the Integrated Data Store, or IDS. This was shortly followed by the Information Management System, a database created by IBM. The Solution Database is a repository of information which is stored as problems and solutions, and is indexed for immediate retrieval. It also provides a multiple language support. Every database can be architected or evaluated by six interdependent database objectives. Four of these objectives are primarily a function of design, development, and implementation: usability, extensibility, data integrity, and performance.

2.3. Bibliometric analysis

In-database analytics is a technology that allows data processing to be conducted within the database by building analytic logic into the database itself. Doing so eliminates the time and effort required to transform data and move it back and forth between a database and a separate analytics application. Database Management Software or DBMS is used for storing, manipulating, and managing data in a database environment. It enables users to design a personalized database engine to meet their analytics and reporting needs. Database software design also supports creating, implementing, and maintaining an organization-wide data management system. Generally, most DBMS packages allow users to perform operations like database creation, storing data, updating data through SQL queries.

2.4. Review Summary

Creating a database application in C is a daunting task, especially for a novice programmer. Although the actually code is quite simple, it is the configuration issues such as importing right library, drivers to use, how to access them, and so forth, that make it an uphill battle. If you think you need to sharpen those C Modern high-level languages are pretty straightforward in these matters. They are quite simple and convenient with an all-in-one-place library with very few configuration troubles. One of the interesting aspects of using a high-level language is that you can almost start coding with even a little understanding of the underlying principles. C/C++, on the other hand, is a different breed. It is not easy to tread even a step superficially. This makes it more intriguing, challenging, that which tests your skills with every code you write. But, once you overcome the preliminary hurdles, there is nothing like C. Most database vendors provide a native client library to access a database. Client libraries are vendor specific; that means that the API provided by one is different from another although the underlying principles are the same.

2.5. Problem Definition

The Linux OS which has the GCC compiler so I was previously facing the issue that GCC could not find sqlite3.h as a directory and when I use sqlite3_close it says undefined reference sqlite3_close.

The solution of above query is Try adding -lsqlite3 to the compiler command. Here is an example: **“g++ mail.cpp -lsqlite3”**

Unable to connect sqlite3 in C program under windows 10
The solution of above query is

- Downloaded the one for 64-bit - sqlite-dll-win64-x64-3240000.zip and extracted to *C:\sqlite* folder.
- Converted the sqlite3.def file to sqlite3.lib by using Developer Command Prompt for VS 2017 and executing /def:sqlite3.def /out:sqlite3.lib /MACHINE:X64 command from *C:\sqlite* folder.

- This time, the command gcc test.c -o test -I\sqlite -L\sqlite -l sqlite3 from *C:\proj* folder was successful.

2.6. Goals/Objectives

SQLite is an embedded relational database engine. Its developers call it a self-contained, serverless, zero-configuration, and transactional SQL database engine. It is currently very popular and there are hundreds of millions copies worldwide in use today. SQLite is used in the Solaris 10, Mac OS, Android, or in the iPhone. The Qt4 library has built-in support for SQLite as well as the Python and PHP. SQLite is used to develop embedded software for devices like televisions, cell phones, cameras, etc. It can manage low to medium-traffic HTTP requests. SQLite can change files into smaller size archives with lesser metadata. SQLite is used as a temporary dataset to get processed with some data within an application.

CHAPTER – 3
DESIGN FLOW / PROCESS

3.1. Evaluation & Selection of Specifications/Features

1. As of now it DBInserter works for below table:

```
"CREATE TABLE Library(" \
    "book_id INT PRIMARY KEY NOT NULL," \
    "book_NAME TEXT NOT NULL," \
    "author_NAME TEXT NOT NULL," \
    "genre TEXT NOT NULL," \
    "book_price INT NOT NULL );";
```

2. This pick csv file from hardcoded path "library.csv" in source folder. Sample csv is also given.

3. Sqlite3, Mingw and visual studio code should already be installed at your machine, prior to compiling and executing this project.

4. This project is only compitable with sqlite3 database, and it “test.db” created in source folder only.

5. To compile download this project in any location at your machine, and change the "home" path given in Makefile located in source folder.

6. For compling at windows through mingW use "mingw32-make".

3.2. Design Constraints

Databases are becoming increasingly important in many areas, including scientific research, technological development, law enforcement, commerce, and government. As an integral part of this process, databases are evolving and mutating in ways that may seem increasingly strange and even threatening. For example, the following trends are easily documented, and have been much discussed in the computer science and/or popular literatures:

1. increasing size, tending towards the truly vast;
2. increasing sophistication and convenience of access mechanisms, tending towards analysis and away from mere query;
3. increasing invisibility, by absorption into the application and/or the user interface;
4. increasing circulation and sharing of information;
5. increasing coordination with other databases and applications;
6. increasing amounts of personal information;
7. increasing commercialization of information; and
8. improving security for database owners -- but not for ordinary consumers, citizens, etc.

It should be evident that there are ethical implications to the ways in which the information in databases is gathered and used, and that the combination of the eight trends listed above is a cause for serious ethical concern. Although these trends are confirmed by our analysis of the data of this study, they are not our main concern. Instead we focus on an issue that may be less obvious but more fundamental, that ethical values are embedded in the ways that databases are structured, and more specifically, in the ways that database access is structured. This supports analyses and arguments that are more empirical and more refined than those that are usually found in the literature on ethics. In fact, there are significant methodological difficulties with the analysis of databases. Here we take the view that a database must not be separated from its use, where "use" includes not only the entire social context, but also the user interface, which embodies the modes of access that are allowed to the database.

The SQLite API's or libraries code is in the public domain, and is free for use by anyone and for any purpose. No license is required.

3.3. Analysis and Feature finalization subject to constraints

We have to modify whenever there is new version of Sqlite comes out in market and then we have install and give path the installed new sqlite libraries or API's in stored folder. As required for our database we change in our .csv file and in source code to work with properly sqlite libraries.

This utility will take input of csv file. This will have columns (RollNumber, Name, age) we change as per our required or database required in "csv_reader.c" file with the path.

Main menu module

1. Creating a database
2. Add items
3. Delete items
4. Search items
5. Edit database record
6. Close Application

If you enter any number wrong attributes except the given one in csv file then it will pop up a message "Can't Open database".

3.4. Design Flow

1. High level Design

===================

a) Line by Line, Read the Csv file--> Read the text in Csv, Example of line is given below

i.e 1,Puroo,20

b) Need to parse the above record and put into a variable of below structre

c)

```
structure
{
        int rollnumber;
        char Name[25];
        int age;
}
Roll nUmber=1
Name =Puroo
Age: 20
```

d) Need to connect sqllite DB from C program

e) prepare insert command and execute it;

2. Low level Design

====================

a) Line by Line, Read the Csv file--> Read the text in Csv, Example of line is given below

a.Need to make file read function separate.

b) Need to parse the above record and put into a variable of below structre(strtok)

c) sqllite libary(Code in binary form) + API (Add(2,4)

(Exe(API) --> LIbrary)

d) Create separate file for each .h , source file and data base connect file source:

main.c

csv_reader.c

database_connect.c

include:

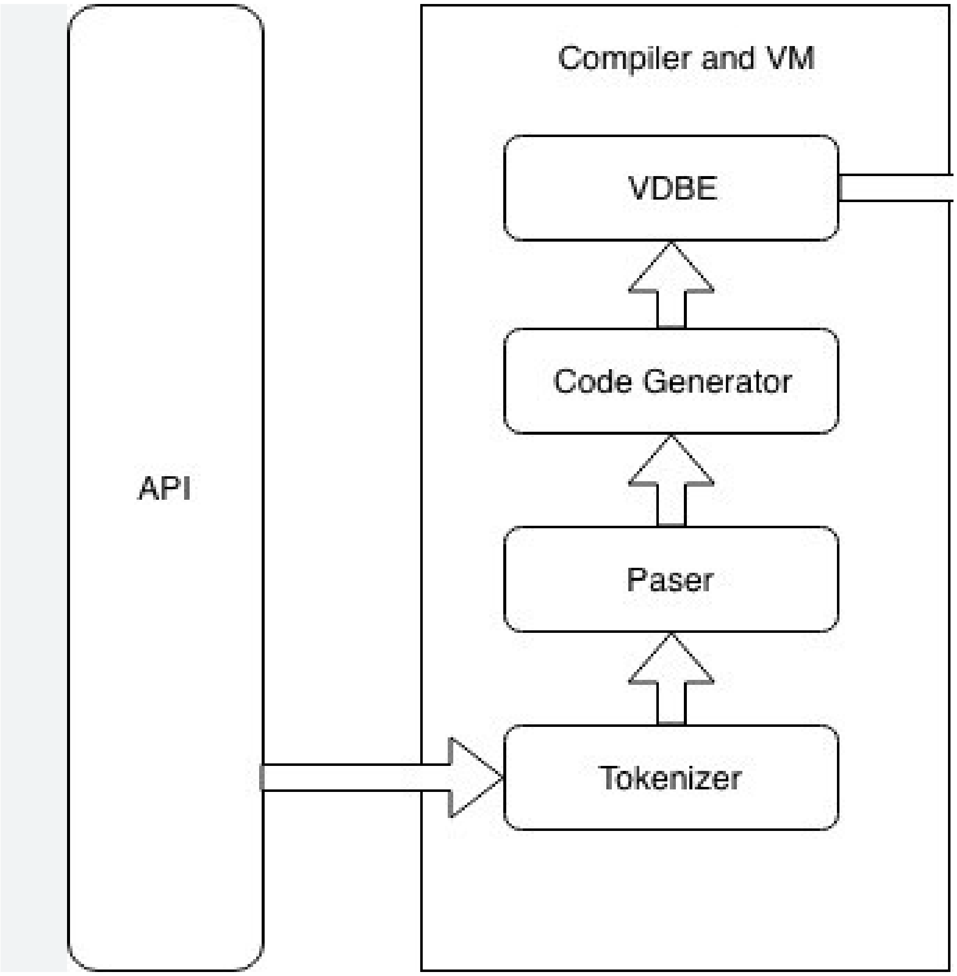
csv_reader.h

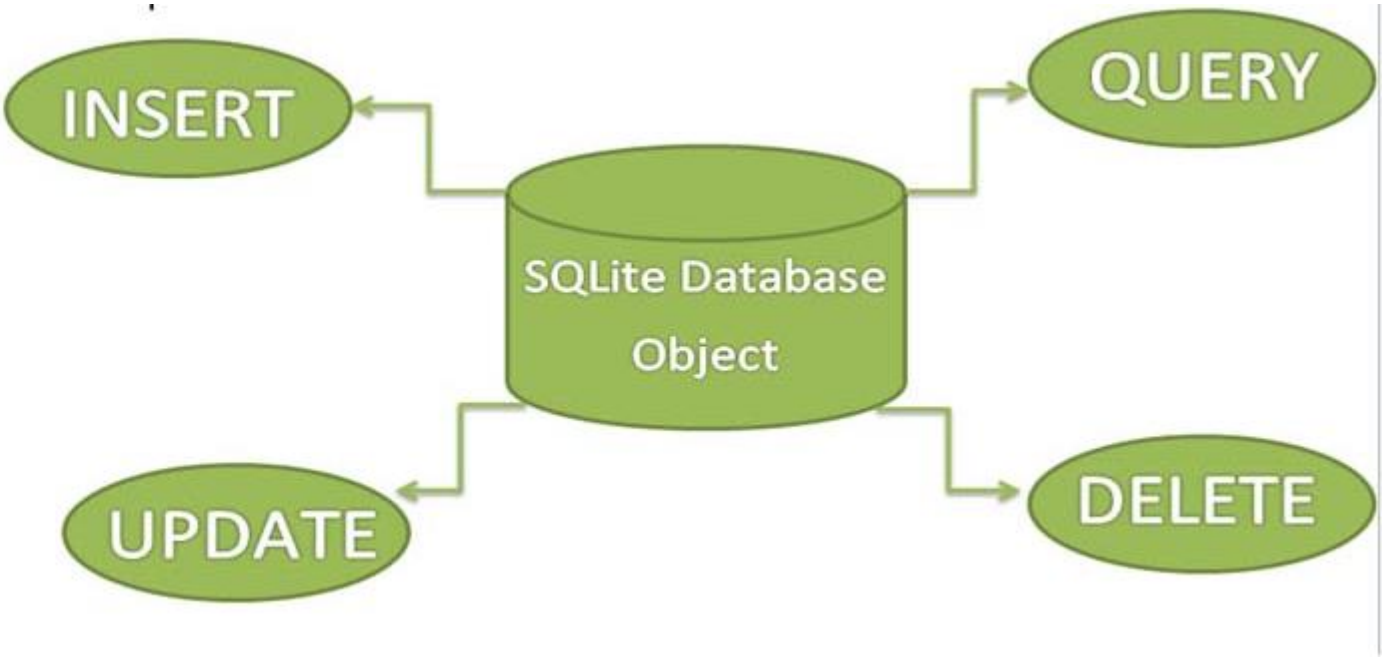
database_connect.h

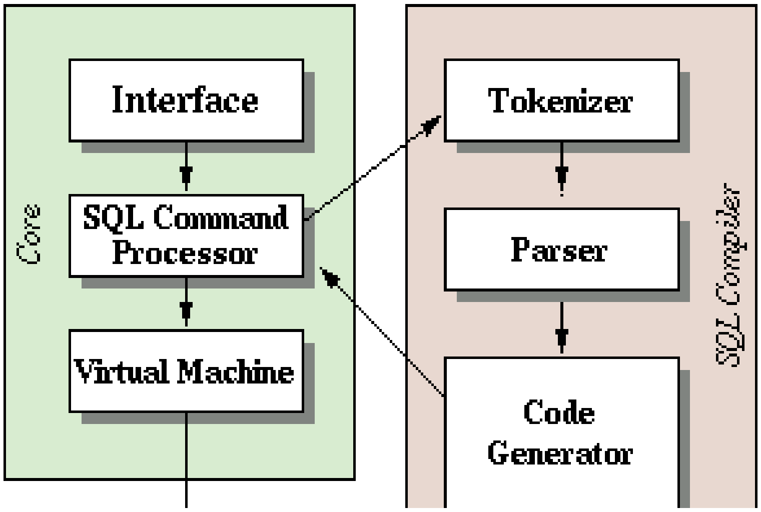
3.5. Design selection

Both the design explain us how the software work and how we can source code and path as per required our system. Low level design just explain us about how csv file help us in to insert data and this utility will write these records into Database table student, whereas High level design explain about how sqlite libraries helps to take data from csv file and insert into database with what other files works and help in our DBInserter software. To compile download this project in any location at your machine, and change the "home" path given in Makefile located in source folder.

3.6. Implementation plan/methodology




INSERT
QUERY
SQLite Database Object
UPDATE
DELETE


Core
Interface
SQL Command Processor
Virtual Machine
Tokenizer
Parser
Code Generator
SQL Compiler

CHAPTER – 4

RESULTS ANALYSIS AND VALIDATION

4.1 Implementation of solution

SQLite is an embedded relational database engine. Its developers call it a self-contained, serverless, zero-configuration, and transactional SQL database engine. It is currently very popular and there are hundreds of millions copies worldwide in use today. SQLite is used in the Solaris 10, Mac OS, Android, or in the iPhone. The Qt4 library has built-in support for SQLite as well as the Python and PHP. Many popular applications use SQLite internally such as Firefox, Google Chrome, or Amarok.

The sqlite3 tool is a terminal based frontend to the SQLite library. It evaluates queries interactively and displays the results in multiple formats. It can also be used within scripts. It has its own set of meta commands including .tables, .load, .databases, or .dump. To get the list of all instructions, we type the .help command.

Now we are going to use the sqlite3 tool to create a new database.

```
$ sqlite3 test.db

SQLite version 3.8.2 2013-12-06 14:53:30

Enter ".help" for instructions

Enter SQL statements terminated with a ";"
```

We provide a parameter to the sqlite3 tool; test.db is a database name. It is a file on our disk. If it is present, it is opened. If not, it is created.

```
sqlite> .tables

sqlite> .exit

$ ls
```

```
test.db
```

The .tables command gives a list of tables in the test.db database. There are currently no tables. The .exit command terminates the interactive session of the sqlite3 command line tool. The ls Unix command shows the contents of the current working directory. We can see the test.db file. All data will be stored in this single file.

This tutorial uses C99. For GNU C compiler, we need to use the -std=c99 option. For Windows users, the Pelles C IDE is highly recommended. (MSVC does not support C99.)

```
int rc = sqlite3_open("test.db", &db);
```

In C99, we can mix declarations with code. In older C programs, we would need to separate this line into two lines.

The sqlite3_libversion() function returns a string indicating the SQLite library.

```
#include <sqlite3.h>
```

This header file defines the interface that the SQLite library presents to the client programs. It contains definitions, function prototypes, and comments. It is an authoritative source for SQLite API.

```
$ gcc -o version version.c -lsqlite3 -std=c99
```

We compile the program with the GNU C compiler.

```
$ ./version
```

Display the version in your system.

The sqlite3 structure defines a database handle. Each open SQLite database is represented by a database handle.

```
sqlite3_stmt *res;
```

The sqlite3_stmt structure represents a single SQL statement.

```
int rc = sqlite3_open(":memory:", &db);
```

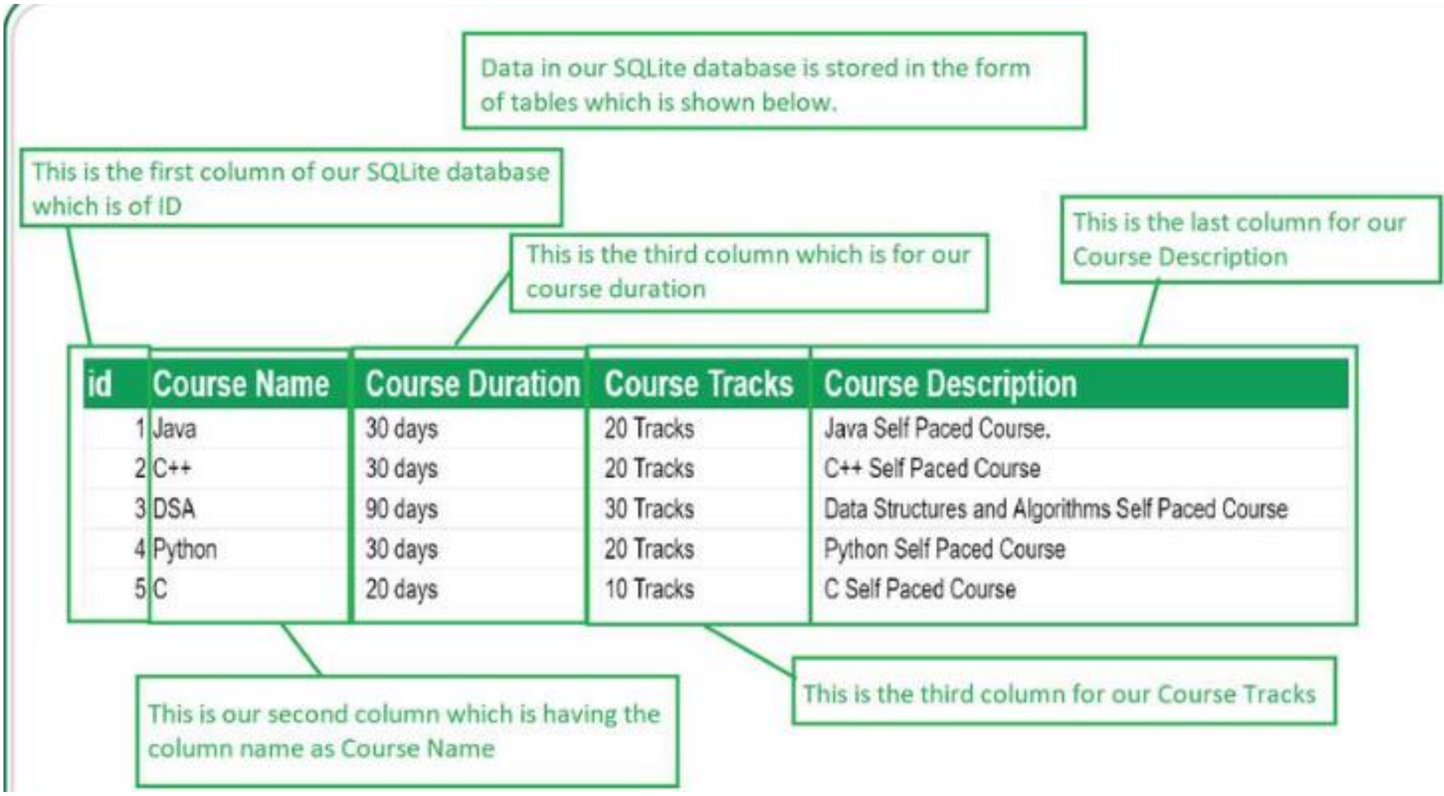
The sqlite3_open() function opens a new database connection. Its parameters are the database name and the database handle. The memory is a special database name using which results in opening an in-memory database. The function's return code indicates whether the database was successfully opened. The SQLITE_OK is returned when the connection was successfully established.

If the return code indicates an error, we print the message to the console, close the database handle, and terminate the program. The sqlite3_errmsg() function returns a description of the error. Whether or not an error occurs when it is opened, resources associated with the database connection handle should be released by passing it to sqlite3_close() function.

rc = sqlite3_prepare_v2(db, "SELECT SQLITE_VERSION()", -1, &res, 0);

Before an SQL statement is executed, it must be first compiled into a byte-code with one of the sqlite3_prepare* functions. (The sqlite3_prepare() function is deprecated.

We connect to the db database, create a required table, and insert rows into the created table. If an error occurs, this pointer will point a the created error message.



Data in our SQLite database is stored in the form of tables which is shown below.

This is the first column of our SQLite database which is of ID

This is the third column which is for our course duration

This is the last column for our Course Description

id	Course Name	Course Duration	Course Tracks	Course Description
1	Java	30 days	20 Tracks	Java Self Paced Course.
2	C++	30 days	20 Tracks	C++ Self Paced Course
3	DSA	90 days	30 Tracks	Data Structures and Algorithms Self Paced Course
4	Python	30 days	20 Tracks	Python Self Paced Course
5	C	20 days	10 Tracks	C Self Paced Course

This is the third column for our Course Tracks

This is our second column which is having the column name as Course Name

The above image is just an example how our DBInserter software takes the input and insert into the database.

- **Schedule progress against plan:** The project is done on time.
- **Current cost versus budget:** Its free cause all libraries , API's and software are free.
- **Current scope compared to plan:** The current scope of my project is to take data from csv file and insert into database and read from the database.
- **Planned versus actual resourcing:** NO
- **Overview of risks:** You should take care of file path and database name.
- **Current quality findings:** Yes, I already tested it 3 times.

CHAPTER – 5

CONCLUSION AND FUTURE WORK

5.1. Conclusion

The default configuration for SQLite works great for most applications. But sometimes user want to tweak the setup to try to squeeze out a little more performance, or take advantage of some obscure feature.

The sqlite in windows interface is used to make global, process-wide configuration changes for SQLite. The interface must be called before database connections are created. The interface allows the programmer to do things like:

Adjust how SQLite does memory allocation, including setting up alternative memory allocators appropriate for safety-critical real-time embedded systems and application-defined memory allocators.

Set up a process-wide error log.

Specify an application-defined page cache.

Adjust the use of mutexes so that they are appropriate for various csv files, or substitute an application-defined mutex system.

After process-wide configuration is complete and database connections have been created, individual database connections can be configured using calls to connect().

5.2. Future work

You have to change csv file according to your data and then come to the database_connecter file and change all the attributes provided in source code according to your Csv file. Now you have to change file path according to where your ".h" file is stored.

As for me my given source folder path is :

```
#include "C:\Users\ASUS\Desktop\DBInster tools\DBInserter-main\include\csv_reader.h"
#include "C:\Users\ASUS\Desktop\DBInster tools\DBInserter-main\include\database_connecter.h"
```

Now go to table_create() function in "database_connecter.c" file and and change sql variable attributes according to you sql table which given purpose to your csv file.

```
sql =  "CREATE TABLE Library(" \
      "book_id INT PRIMARY KEY NOT NULL," \
      "book_NAME TEXT NOT NULL," \
      "author_NAME TEXT NOT NULL," \
      "genre TEXT NOT NULL," \
      "book_price INT NOT NULL );";
```

Now come to insert() function and also change attributes same according to table_create() attributes to display.

```
sprintf(sql,"INSERT INTO library (book_id,book_name,author_name,genre,book_price) VALUES ( %d , '%s' , '%s' , '%s' , %d);",book_id,book_name,author_name,genre,book_price);
```

Also change your table name according to whatever you prefer.

Now come to the Makefile

```
home = "C:\Users\ASUS\Desktop\DBInster tools\DBInserter-main"
library_path = -L$(home)"\lib"
library = -lsqlite3
include = -I$(home)"\include"
Command = gcc
binary_name = db
source_file = main.c csv_reader.c database_connecter.c
db:
    $(Command) -o $(binary_name) $(source_file)  $(include) $(library_path) $(library)
```

Change your home path according to folder path wherever you store the project

And run the Make file in cmd and it gives what gcc command what you have to run in your system then run that command , it shows you have created your table in db database.

REFERENCES

1. https://learn.microsoft.com/en-us/azure/azure-sql/database/develop-cplusplus-simple?view=azuresql
2. https://books.google.com/books?id=OEJ1CQAAQBAJ
3. https://www.sqlite.org/cintro.html
4. https://cseweb.ucsd.edu/~goguen/papers/4s/4s.html
5. https://zetcode.com/db/sqlitec/
6. https://www.coursera.org/learn/c-structured-programming/home/welcome
7. https://www.hackerrank.com/dashboard
8. https://github.com/purookulsh13/DBInserter
9. https://www.geeksforgeeks.org/database-connectivity-using-cc/

APPENDIX

SQLite has the following characteristics:

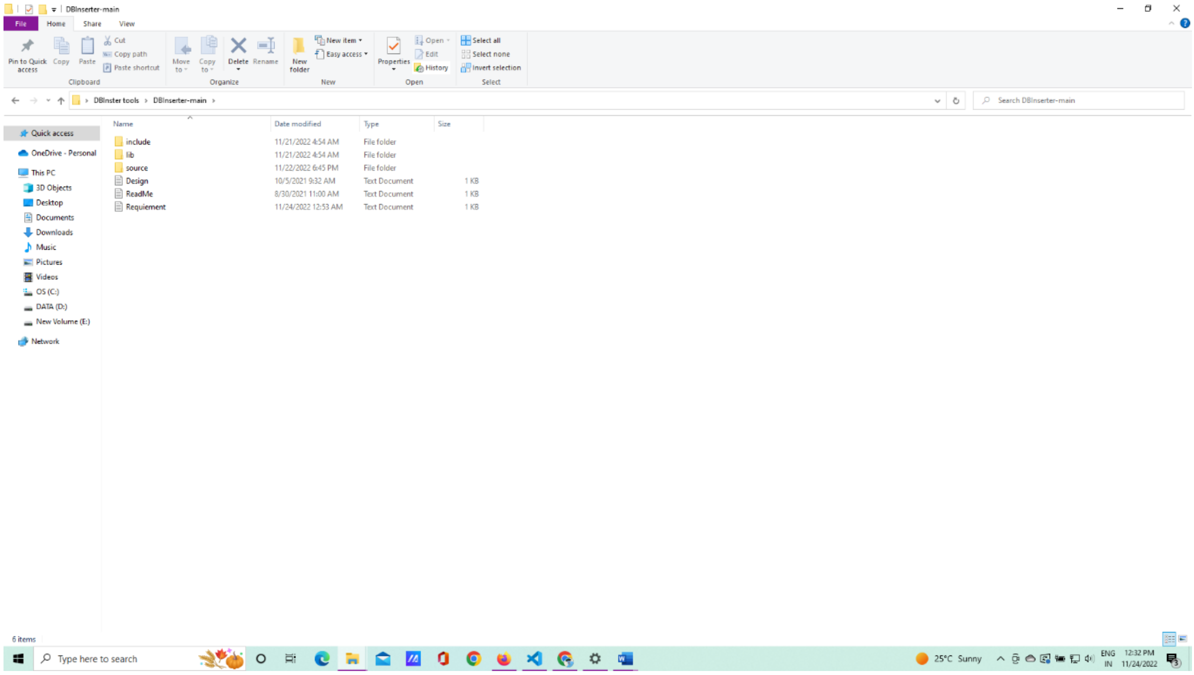
- It is embedded with the native sdk.
- It is ACID-compliant (atomicity, consistency, isolation, durability).
- It is weakly-typed i.e, any object can be stored in any column, regardless of how that column was declared. For example you can insert a string into a database column of type integer.
- It does not officially support foreign key constraints, although triggers can be used as a workaround.

The relevancy of a document may depend on something other than just the data available in the return value of matchinfo. For example each document in the database may be assigned a static weight based on factors unrelated to its content (origin, author, age, number of references etc.). These values can be stored by the application in a separate table that can be joined against the documents table in the sub-query so that the rank function may access them.

SQLite, like all SQL database engines, makes extensive use of malloc() (See the separate report on dynamic memory allocation in sqlite for additional detail.) On servers and workstations, malloc() never fails in practice and so correct handling of out-of-memory (OOM) errors is not particularly important. But on embedded devices, OOM errors are frighteningly common and since SQLite is frequently used on embedded devices, it is important that SQLite be able to gracefully handle OOM errors.

USER MANUAL

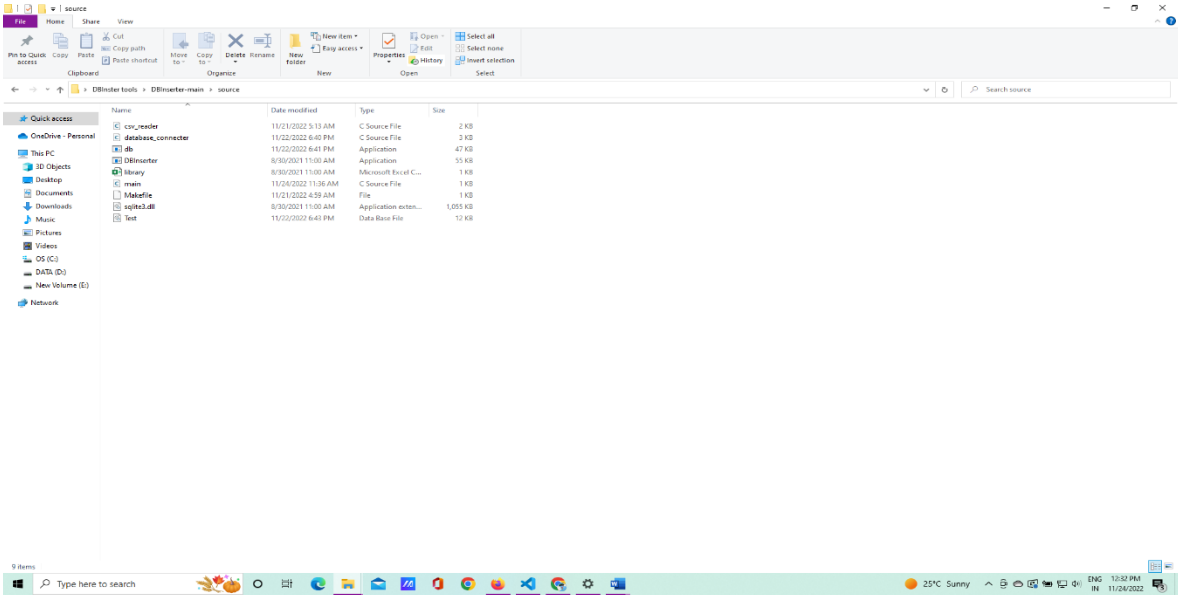
1. Open the DBInserter_main folder.



DBInster tools › DBInserter-main ›

Name	Date modified	Type	Size
include	11/21/2022 4:54 AM	File folder	
lib	11/21/2022 4:54 AM	File folder	
source	11/22/2022 6:45 PM	File folder	
Design	10/5/2021 9:32 AM	Text Document	1 KB
ReadMe	8/30/2021 11:00 AM	Text Document	1 KB
Requiement	11/24/2022 12:53 AM	Text Document	1 KB

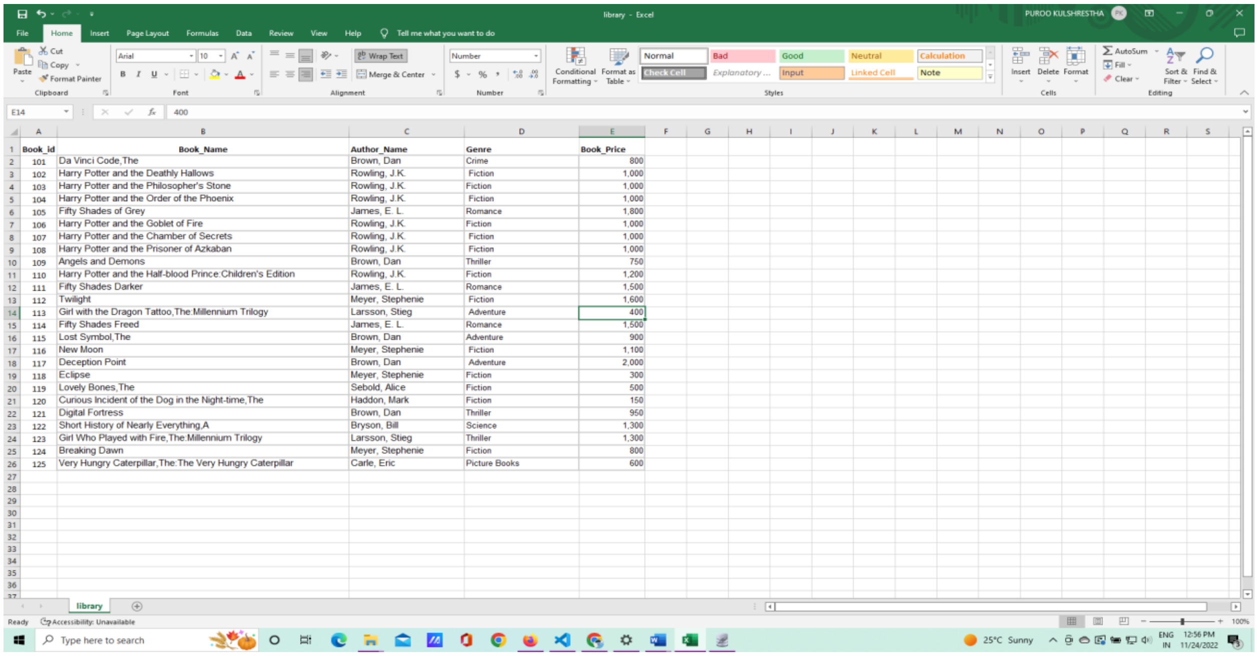
2. Open the Source folder.



DBInster tools › DBInserter-main › source

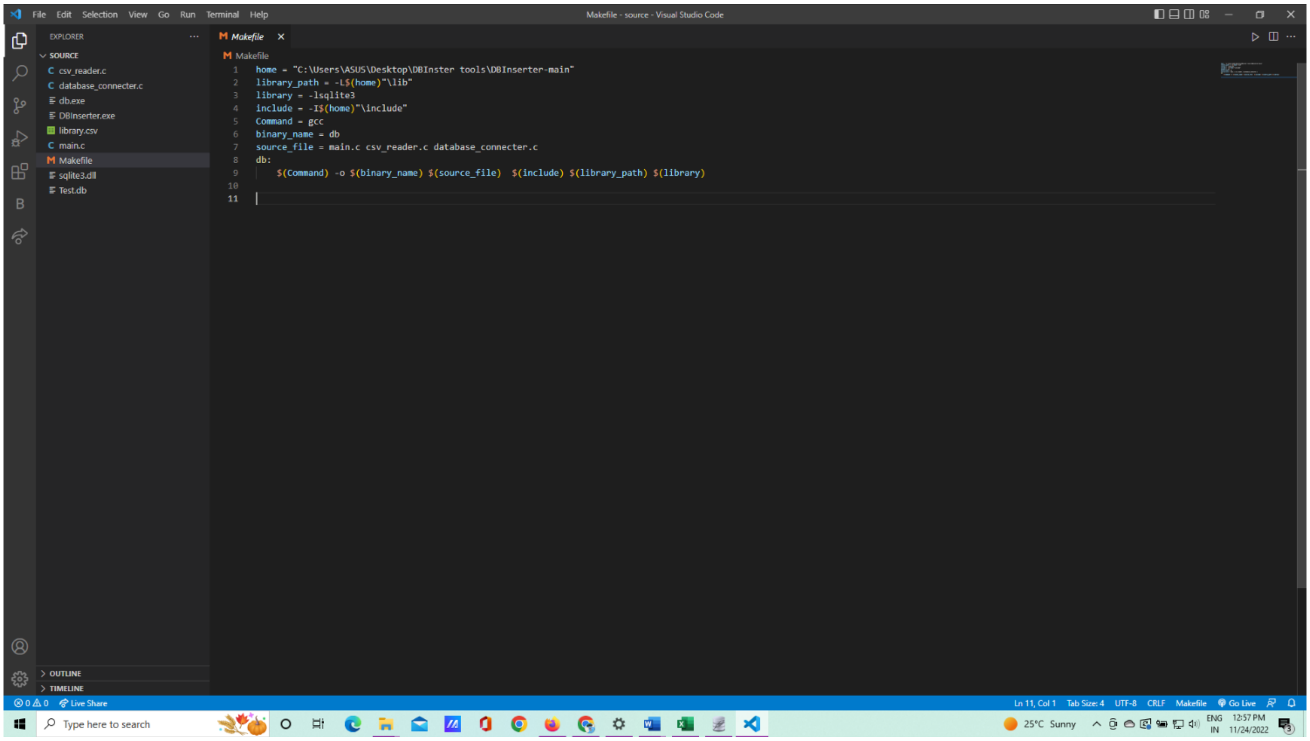
Name	Date modified	Type	Size
csv_reader	11/21/2022 5:13 AM	C Source File	2 KB
database_connecter	11/22/2022 6:40 PM	C Source File	3 KB
db	11/22/2022 6:41 PM	Application	47 KB
DBInserter	8/30/2021 11:00 AM	Application	55 KB
library	8/30/2021 11:00 AM	Microsoft Excel C...	1 KB
main	11/24/2022 11:36 AM	C Source File	1 KB
Makefile	11/21/2022 4:59 AM	File	1 KB
sqlite3.dll	8/30/2021 11:00 AM	Application exten...	1,055 KB
Test	11/22/2022 6:43 PM	Data Base File	12 KB

3. Open the .csv file



Book_id	Book_Name	Author_Name	Genre	Book_Price
101	Da Vinci Code,The	Brown, Dan	Crime	800
102	Harry Potter and the Deathly Hallows	Rowling, J.K.	Fiction	1,000
103	Harry Potter and the Philosopher's Stone	Rowling, J.K.	Fiction	1,000
104	Harry Potter and the Order of the Phoenix	Rowling, J.K.	Fiction	1,000
105	Fifty Shades of Grey	James, E. L.	Romance	1,800
106	Harry Potter and the Goblet of Fire	Rowling, J.K.	Fiction	1,000
107	Harry Potter and the Chamber of Secrets	Rowling, J.K.	Fiction	1,000
108	Harry Potter and the Prisoner of Azkaban	Rowling, J.K.	Fiction	1,000
109	Angels and Demons	Brown, Dan	Thriller	750
110	Harry Potter and the Half-blood Prince:Children's Edition	Rowling, J.K.	Fiction	1,200
111	Fifty Shades Darker	James, E. L.	Romance	1,500
112	Twilight	Meyer, Stephenie	Fiction	1,600
113	Girl with the Dragon Tattoo,The:Millennium Trilogy	Larsson, Stieg	Adventure	400
114	Fifty Shades Freed	James, E. L.	Romance	1,500
115	Lost Symbol,The	Brown, Dan	Adventure	900
116	New Moon	Meyer, Stephenie	Fiction	1,100
117	Deception Point	Brown, Dan	Adventure	2,000
118	Eclipse	Meyer, Stephenie	Fiction	300
119	Lovely Bones,The	Sebold, Alice	Fiction	500
120	Curious Incident of the Dog in the Night-time,The	Haddon, Mark	Fiction	150
121	Digital Fortress	Brown, Dan	Thriller	950
122	Short History of Nearly Everything,A	Bryson, Bill	Science	1,300
123	Girl Who Played with Fire,The:Millennium Trilogy	Larsson, Stieg	Thriller	1,300
124	Breaking Dawn	Meyer, Stephenie	Fiction	800
125	Very Hungry Caterpillar,The:The Very Hungry Caterpillar	Carle, Eric	Picture Books	600

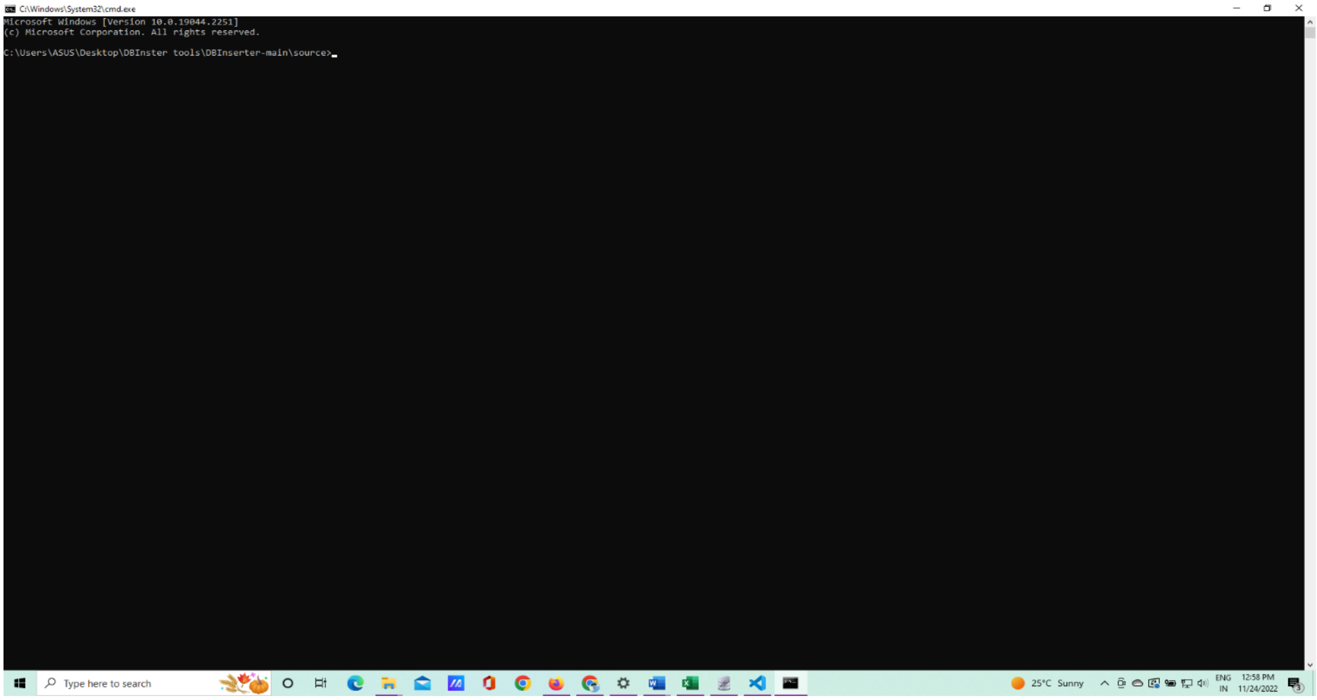
4. Open the Visual studio code and open the Makefile



```
home = "C:\Users\ASUS\Desktop\DBInster tools\DBInserter-main"
library_path = -L$(home)"\lib"
library = -lsqlite3
include = -I$(home)"\include"
Command = gcc
binary_name = db
source_file = main.c csv_reader.c database_connecter.c
db:
	$(Command) -o $(binary_name) $(source_file)  $(include) $(library_path) $(library)

```

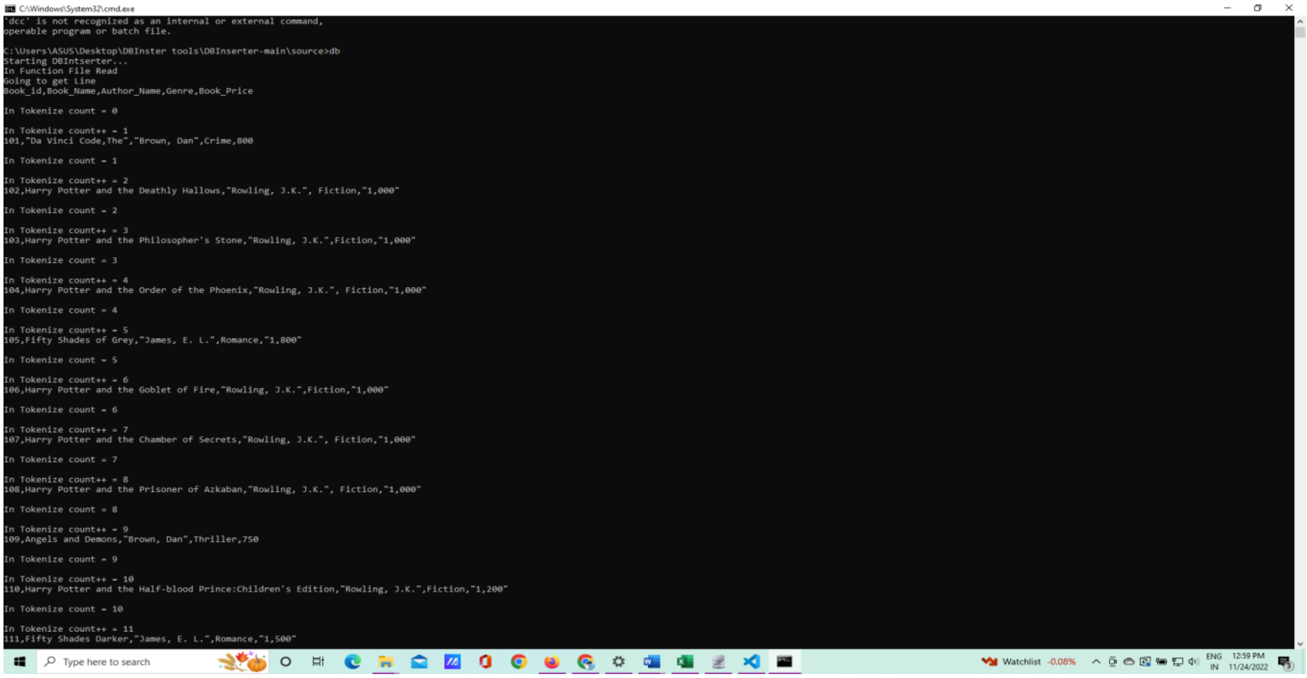
DBInserter

5. Open the command terminal in same folder



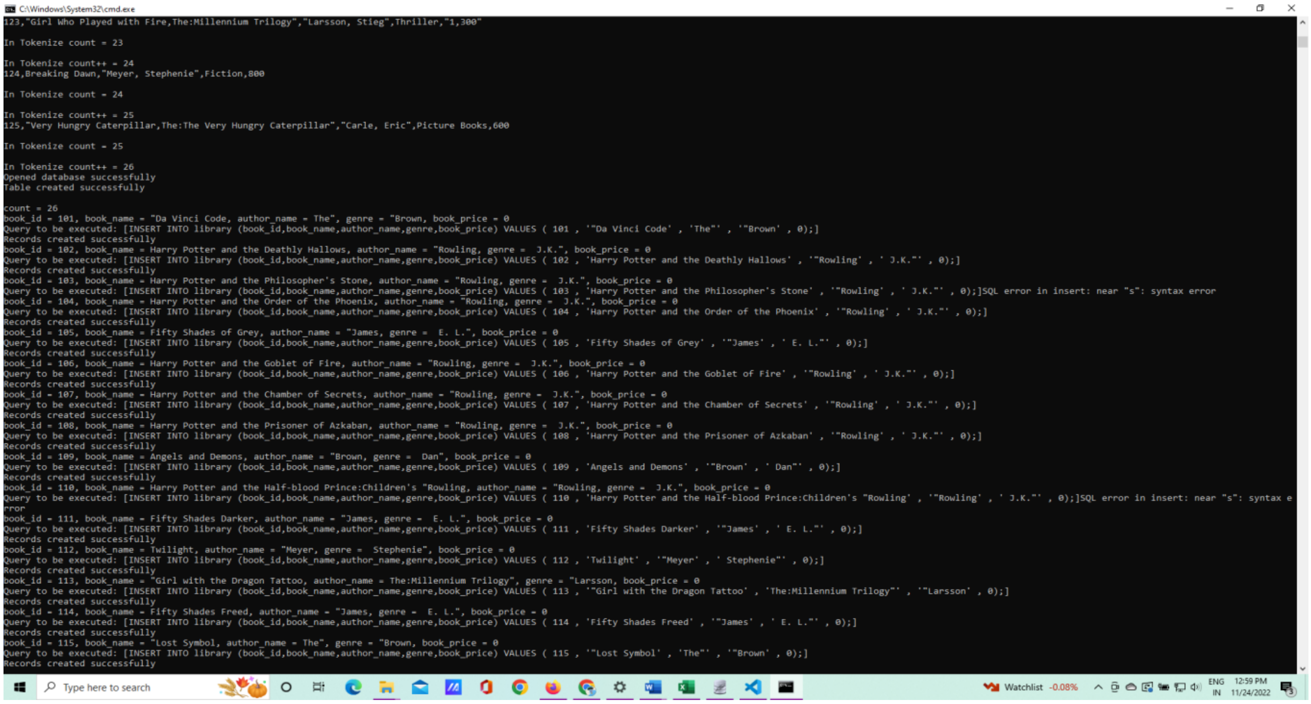
```
Microsoft Windows [Version 10.0.19044.2251]
(c) Microsoft Corporation. All rights reserved.

C:\Users\ASUS\Desktop\DBInster tools\DBInserter-main\source>_
```

6. Run the db file in cmd



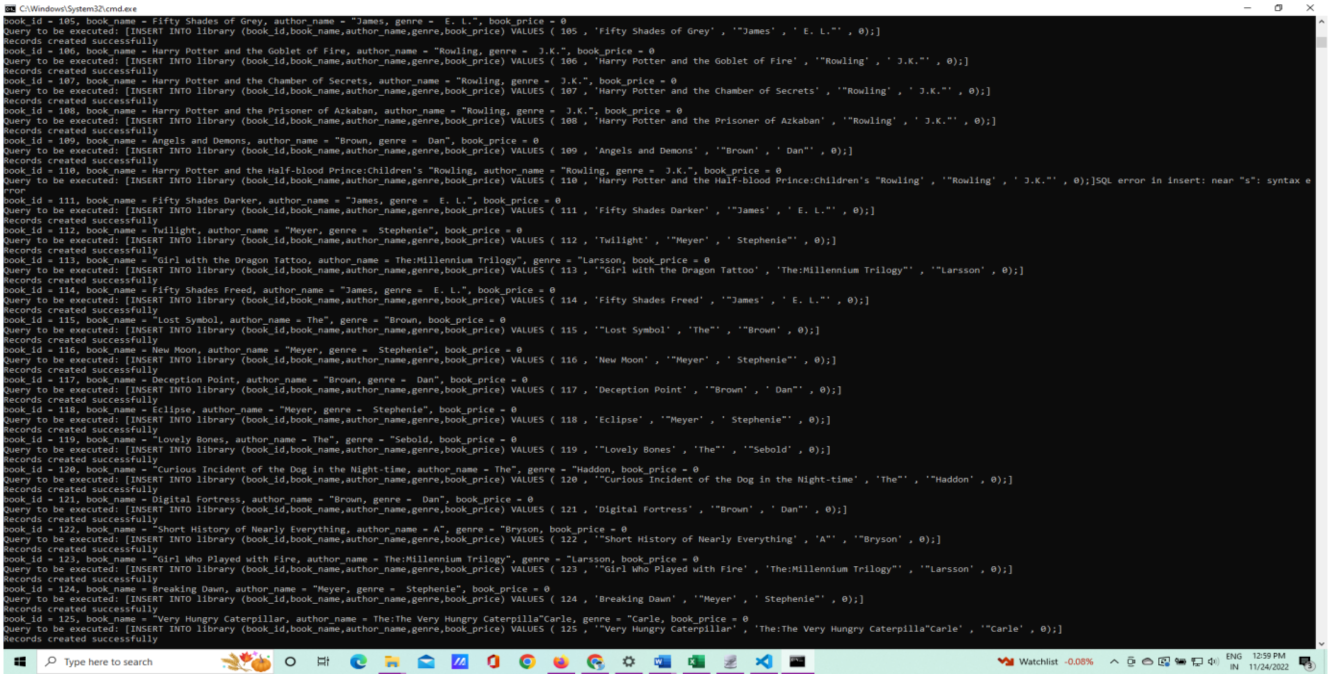
```
'dcc' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\ASUS\Desktop\DBInster tools\DBInserter-main\source>db
Starting DBIntserter...
In Function File Read
Going to get Line
Book_id,Book_Name,Author_Name,Genre,Book_Price

In Tokenize count = 0

In Tokenize count++ = 1
101,"Da Vinci Code,The","Brown, Dan",Crime,800

In Tokenize count = 1

In Tokenize count++ = 2
102,Harry Potter and the Deathly Hallows,"Rowling, J.K.", Fiction,"1,000"

In Tokenize count = 2

In Tokenize count++ = 3
103,Harry Potter and the Philosopher's Stone,"Rowling, J.K.",Fiction,"1,000"

In Tokenize count = 3

In Tokenize count++ = 4
104,Harry Potter and the Order of the Phoenix,"Rowling, J.K.", Fiction,"1,000"

In Tokenize count = 4

In Tokenize count++ = 5
105,Fifty Shades of Grey,"James, E. L.",Romance,"1,800"

In Tokenize count = 5

In Tokenize count++ = 6
106,Harry Potter and the Goblet of Fire,"Rowling, J.K.",Fiction,"1,000"

In Tokenize count = 6

In Tokenize count++ = 7
107,Harry Potter and the Chamber of Secrets,"Rowling, J.K.", Fiction,"1,000"

In Tokenize count = 7

In Tokenize count++ = 8
108,Harry Potter and the Prisoner of Azkaban,"Rowling, J.K.", Fiction,"1,000"

In Tokenize count = 8

In Tokenize count++ = 9
109,Angels and Demons,"Brown, Dan",Thriller,750

In Tokenize count = 9

In Tokenize count++ = 10
110,Harry Potter and the Half-blood Prince:Children's Edition,"Rowling, J.K.",Fiction,"1,200"

In Tokenize count = 10

In Tokenize count++ = 11
111,Fifty Shades Darker,"James, E. L.",Romance,"1,500"
```

7. Now cmd run the process and insert the data in test.db



```
123,"Girl Who Played with Fire,The:Millennium Trilogy","Larsson, Stieg",Thriller,"1,300"

In Tokenize count = 23

In Tokenize count++ = 24
124,Breaking Dawn,"Meyer, Stephenie",Fiction,800

In Tokenize count = 24

In Tokenize count++ = 25
125,"Very Hungry Caterpillar,The:The Very Hungry Caterpillar","Carle, Eric",Picture Books,600

In Tokenize count = 25

In Tokenize count++ = 26
Opened database successfully
Table created successfully

count = 26
book_id = 101, book_name = "Da Vinci Code, author_name = The", genre = "Brown, book_price = 0
Query to be executed: [INSERT INTO library (book_id,book_name,author_name,genre,book_price) VALUES ( 101 , '"Da Vinci Code' , 'The"' , '"Brown' , 0);]
Records created successfully
book_id = 102, book_name = Harry Potter and the Deathly Hallows, author_name = "Rowling, genre =  J.K.", book_price = 0
Query to be executed: [INSERT INTO library (book_id,book_name,author_name,genre,book_price) VALUES ( 102 , 'Harry Potter and the Deathly Hallows' , '"Rowling' , ' J.K."' , 0);]
Records created successfully
book_id = 103, book_name = Harry Potter and the Philosopher's Stone, author_name = "Rowling, genre =  J.K.", book_price = 0
Query to be executed: [INSERT INTO library (book_id,book_name,author_name,genre,book_price) VALUES ( 103 , 'Harry Potter and the Philosopher's Stone' , '"Rowling' , ' J.K."' , 0);]SQL error in insert: near "s": syntax error
book_id = 104, book_name = Harry Potter and the Order of the Phoenix, author_name = "Rowling, genre =  J.K.", book_price = 0
Query to be executed: [INSERT INTO library (book_id,book_name,author_name,genre,book_price) VALUES ( 104 , 'Harry Potter and the Order of the Phoenix' , '"Rowling' , ' J.K."' , 0);]
Records created successfully
book_id = 105, book_name = Fifty Shades of Grey, author_name = "James, genre =  E. L.", book_price = 0
Query to be executed: [INSERT INTO library (book_id,book_name,author_name,genre,book_price) VALUES ( 105 , 'Fifty Shades of Grey' , '"James' , ' E. L."' , 0);]
Records created successfully
book_id = 106, book_name = Harry Potter and the Goblet of Fire, author_name = "Rowling, genre =  J.K.", book_price = 0
Query to be executed: [INSERT INTO library (book_id,book_name,author_name,genre,book_price) VALUES ( 106 , 'Harry Potter and the Goblet of Fire' , '"Rowling' , ' J.K."' , 0);]
Records created successfully
book_id = 107, book_name = Harry Potter and the Chamber of Secrets, author_name = "Rowling, genre =  J.K.", book_price = 0
Query to be executed: [INSERT INTO library (book_id,book_name,author_name,genre,book_price) VALUES ( 107 , 'Harry Potter and the Chamber of Secrets' , '"Rowling' , ' J.K."' , 0);]
Records created successfully
book_id = 108, book_name = Harry Potter and the Prisoner of Azkaban, author_name = "Rowling, genre =  J.K.", book_price = 0
Query to be executed: [INSERT INTO library (book_id,book_name,author_name,genre,book_price) VALUES ( 108 , 'Harry Potter and the Prisoner of Azkaban' , '"Rowling' , ' J.K."' , 0);]
Records created successfully
book_id = 109, book_name = Angels and Demons, author_name = "Brown, genre =  Dan", book_price = 0
Query to be executed: [INSERT INTO library (book_id,book_name,author_name,genre,book_price) VALUES ( 109 , 'Angels and Demons' , '"Brown' , ' Dan"' , 0);]
Records created successfully
book_id = 110, book_name = Harry Potter and the Half-blood Prince:Children's "Rowling, author_name = "Rowling, genre =  J.K.", book_price = 0
Query to be executed: [INSERT INTO library (book_id,book_name,author_name,genre,book_price) VALUES ( 110 , 'Harry Potter and the Half-blood Prince:Children's "Rowling' , '"Rowling' , ' J.K."' , 0);]SQL error in insert: near "s": syntax e
rror
book_id = 111, book_name = Fifty Shades Darker, author_name = "James, genre =  E. L.", book_price = 0
Query to be executed: [INSERT INTO library (book_id,book_name,author_name,genre,book_price) VALUES ( 111 , 'Fifty Shades Darker' , '"James' , ' E. L."' , 0);]
Records created successfully
book_id = 112, book_name = Twilight, author_name = "Meyer, genre =  Stephenie", book_price = 0
Query to be executed: [INSERT INTO library (book_id,book_name,author_name,genre,book_price) VALUES ( 112 , 'Twilight' , '"Meyer' , ' Stephenie"' , 0);]
Records created successfully
book_id = 113, book_name = "Girl with the Dragon Tattoo, author_name = The:Millennium Trilogy", genre = "Larsson, book_price = 0
Query to be executed: [INSERT INTO library (book_id,book_name,author_name,genre,book_price) VALUES ( 113 , '"Girl with the Dragon Tattoo' , 'The:Millennium Trilogy"' , '"Larsson' , 0);]
Records created successfully
book_id = 114, book_name = Fifty Shades Freed, author_name = "James, genre =  E. L.", book_price = 0
Query to be executed: [INSERT INTO library (book_id,book_name,author_name,genre,book_price) VALUES ( 114 , 'Fifty Shades Freed' , '"James' , ' E. L."' , 0);]
Records created successfully
book_id = 115, book_name = "Lost Symbol, author_name = The", genre = "Brown, book_price = 0
Query to be executed: [INSERT INTO library (book_id,book_name,author_name,genre,book_price) VALUES ( 115 , '"Lost Symbol' , 'The"' , '"Brown' , 0);]
Records created successfully
```

8. In end it show Record completed succesfully



```
book_id = 105, book_name = Fifty Shades of Grey, author_name = "James, genre =  E. L.", book_price = 0
Query to be executed: [INSERT INTO library (book_id,book_name,author_name,genre,book_price) VALUES ( 105 , 'Fifty Shades of Grey' , '"James' , ' E. L."' , 0);]
Records created successfully
book_id = 106, book_name = Harry Potter and the Goblet of Fire, author_name = "Rowling, genre =  J.K.", book_price = 0
Query to be executed: [INSERT INTO library (book_id,book_name,author_name,genre,book_price) VALUES ( 106 , 'Harry Potter and the Goblet of Fire' , '"Rowling' , ' J.K."' , 0);]
Records created successfully
book_id = 107, book_name = Harry Potter and the Chamber of Secrets, author_name = "Rowling, genre =  J.K.", book_price = 0
Query to be executed: [INSERT INTO library (book_id,book_name,author_name,genre,book_price) VALUES ( 107 , 'Harry Potter and the Chamber of Secrets' , '"Rowling' , ' J.K."' , 0);]
Records created successfully
book_id = 108, book_name = Harry Potter and the Prisoner of Azkaban, author_name = "Rowling, genre =  J.K.", book_price = 0
Query to be executed: [INSERT INTO library (book_id,book_name,author_name,genre,book_price) VALUES ( 108 , 'Harry Potter and the Prisoner of Azkaban' , '"Rowling' , ' J.K."' , 0);]
Records created successfully
book_id = 109, book_name = Angels and Demons, author_name = "Brown, genre =  Dan", book_price = 0
Query to be executed: [INSERT INTO library (book_id,book_name,author_name,genre,book_price) VALUES ( 109 , 'Angels and Demons' , '"Brown' , ' Dan"' , 0);]
Records created successfully
book_id = 110, book_name = Harry Potter and the Half-blood Prince:Children's "Rowling, author_name = "Rowling, genre =  J.K.", book_price = 0
Query to be executed: [INSERT INTO library (book_id,book_name,author_name,genre,book_price) VALUES ( 110 , 'Harry Potter and the Half-blood Prince:Children's "Rowling' , '"Rowling' , ' J.K."' , 0);]SQL error in insert: near "s": syntax e
rror
book_id = 111, book_name = Fifty Shades Darker, author_name = "James, genre =  E. L.", book_price = 0
Query to be executed: [INSERT INTO library (book_id,book_name,author_name,genre,book_price) VALUES ( 111 , 'Fifty Shades Darker' , '"James' , ' E. L."' , 0);]
Records created successfully
book_id = 112, book_name = Twilight, author_name = "Meyer, genre =  Stephenie", book_price = 0
Query to be executed: [INSERT INTO library (book_id,book_name,author_name,genre,book_price) VALUES ( 112 , 'Twilight' , '"Meyer' , ' Stephenie"' , 0);]
Records created successfully
book_id = 113, book_name = "Girl with the Dragon Tattoo, author_name = The:Millennium Trilogy", genre = "Larsson, book_price = 0
Query to be executed: [INSERT INTO library (book_id,book_name,author_name,genre,book_price) VALUES ( 113 , '"Girl with the Dragon Tattoo' , 'The:Millennium Trilogy"' , '"Larsson' , 0);]
Records created successfully
book_id = 114, book_name = Fifty Shades Freed, author_name = "James, genre =  E. L.", book_price = 0
Query to be executed: [INSERT INTO library (book_id,book_name,author_name,genre,book_price) VALUES ( 114 , 'Fifty Shades Freed' , '"James' , ' E. L."' , 0);]
Records created successfully
book_id = 115, book_name = "Lost Symbol, author_name = The", genre = "Brown, book_price = 0
Query to be executed: [INSERT INTO library (book_id,book_name,author_name,genre,book_price) VALUES ( 115 , '"Lost Symbol' , 'The"' , '"Brown' , 0);]
Records created successfully
book_id = 116, book_name = New Moon, author_name = "Meyer, genre =  Stephenie", book_price = 0
Query to be executed: [INSERT INTO library (book_id,book_name,author_name,genre,book_price) VALUES ( 116 , 'New Moon' , '"Meyer' , ' Stephenie"' , 0);]
Records created successfully
book_id = 117, book_name = Deception Point, author_name = "Brown, genre =  Dan", book_price = 0
Query to be executed: [INSERT INTO library (book_id,book_name,author_name,genre,book_price) VALUES ( 117 , 'Deception Point' , '"Brown' , ' Dan"' , 0);]
Records created successfully
book_id = 118, book_name = Eclipse, author_name = "Meyer, genre =  Stephenie", book_price = 0
Query to be executed: [INSERT INTO library (book_id,book_name,author_name,genre,book_price) VALUES ( 118 , 'Eclipse' , '"Meyer' , ' Stephenie"' , 0);]
Records created successfully
book_id = 119, book_name = "Lovely Bones, author_name = The", genre = "Sebold, book_price = 0
Query to be executed: [INSERT INTO library (book_id,book_name,author_name,genre,book_price) VALUES ( 119 , '"Lovely Bones' , 'The"' , '"Sebold' , 0);]
Records created successfully
book_id = 120, book_name = "Curious Incident of the Dog in the Night-time, author_name = The", genre = "Haddon, book_price = 0
Query to be executed: [INSERT INTO library (book_id,book_name,author_name,genre,book_price) VALUES ( 120 , '"Curious Incident of the Dog in the Night-time' , 'The"' , '"Haddon' , 0);]
Records created successfully
book_id = 121, book_name = Digital Fortress, author_name = "Brown, genre =  Dan", book_price = 0
Query to be executed: [INSERT INTO library (book_id,book_name,author_name,genre,book_price) VALUES ( 121 , 'Digital Fortress' , '"Brown' , ' Dan"' , 0);]
Records created successfully
book_id = 122, book_name = "Short History of Nearly Everything, author_name = A", genre = "Bryson, book_price = 0
Query to be executed: [INSERT INTO library (book_id,book_name,author_name,genre,book_price) VALUES ( 122 , '"Short History of Nearly Everything' , 'A"' , '"Bryson' , 0);]
Records created successfully
book_id = 123, book_name = "Girl Who Played with Fire, author_name = The:Millennium Trilogy", genre = "Larsson, book_price = 0
Query to be executed: [INSERT INTO library (book_id,book_name,author_name,genre,book_price) VALUES ( 123 , '"Girl Who Played with Fire' , 'The:Millennium Trilogy"' , '"Larsson' , 0);]
Records created successfully
book_id = 124, book_name = Breaking Dawn, author_name = "Meyer, genre =  Stephenie", book_price = 0
Query to be executed: [INSERT INTO library (book_id,book_name,author_name,genre,book_price) VALUES ( 124 , 'Breaking Dawn' , '"Meyer' , ' Stephenie"' , 0);]
Records created successfully
book_id = 125, book_name = "Very Hungry Caterpillar, author_name = The:The Very Hungry Caterpilla"Carle, genre = "Carle, book_price = 0
Query to be executed: [INSERT INTO library (book_id,book_name,author_name,genre,book_price) VALUES ( 125 , '"Very Hungry Caterpillar' , 'The:The Very Hungry Caterpilla"Carle' , '"Carle' , 0);]
Records created successfully
```

9. Open the Sqlite Editor

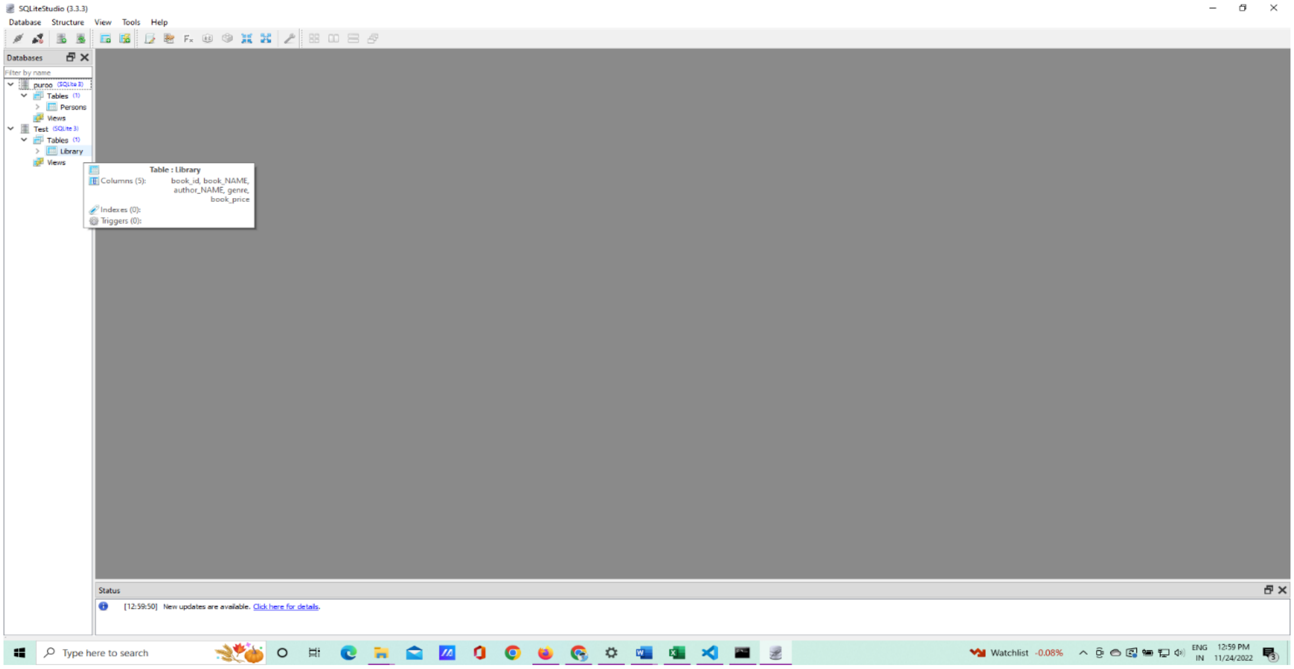


Table : Library
Columns (5): book_id, book_NAME, author_NAME, genre, book_price
Indexes (0):
Triggers (0):

[12:59:50] New updates are available. Click here for details.

10.Open the test.db and in their Library table created successfully

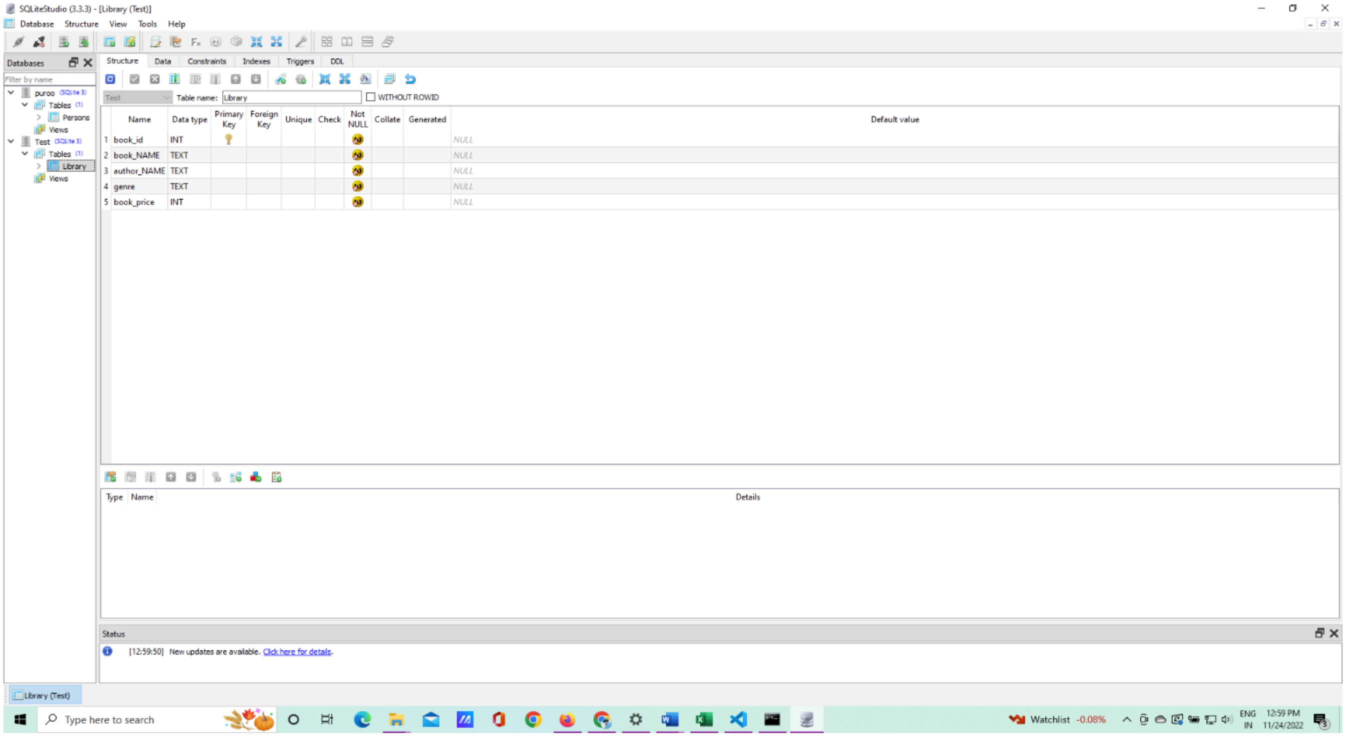
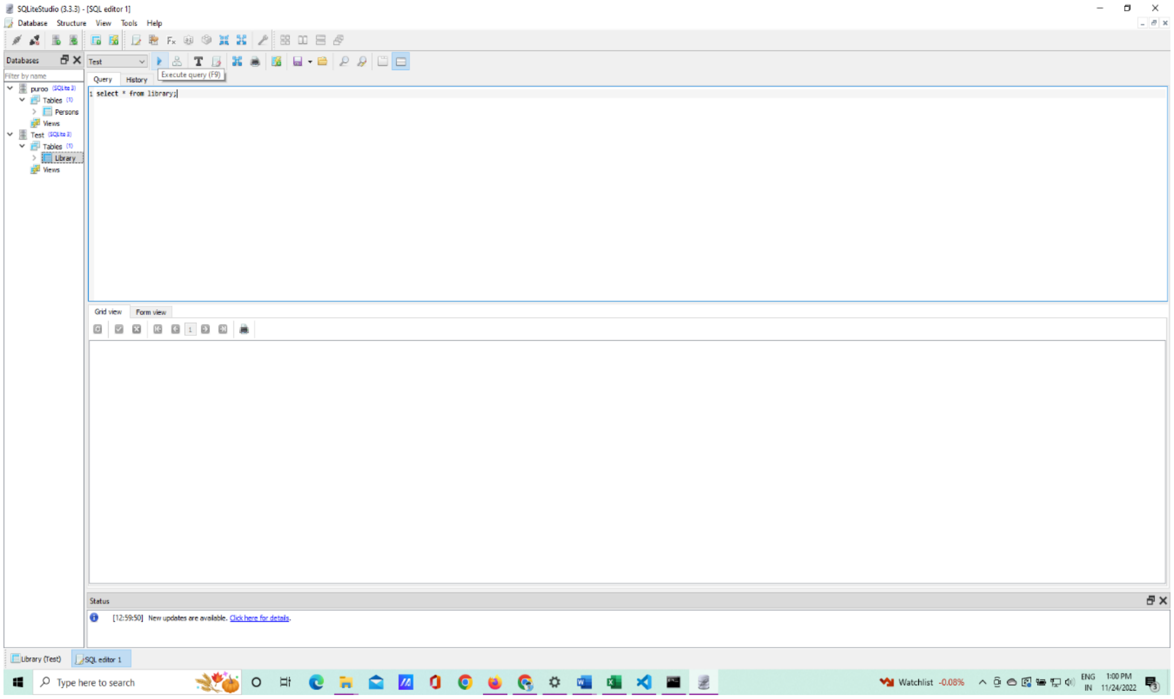


Table name: Library ☐ WITHOUT ROWID

	Name	Data type	Primary Key	Foreign Key	Unique	Check	Not NULL	Collate	Generated	Default value
1	book_id	INT	🔑				✓			NULL
2	book_NAME	TEXT					✓			NULL
3	author_NAME	TEXT					✓			NULL
4	genre	TEXT					✓			NULL
5	book_price	INT					✓			NULL

[12:59:50] New updates are available. Click here for details.

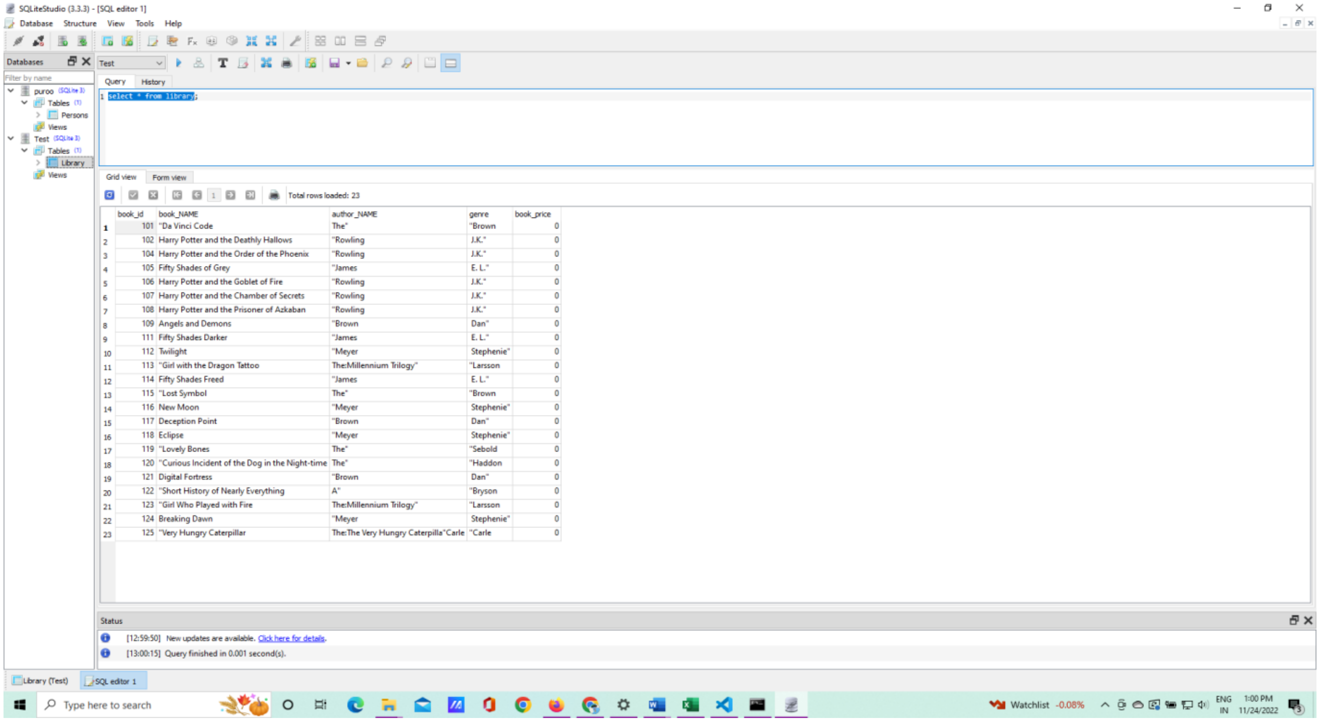
11.Now run the display command in sql command editor.



```
select * from library;
```

[12:59:50] New updates are available. Click here for details.

12. It display the result in your database.



```
select * from library;
```

Total rows loaded: 23

	book_id	book_NAME	author_NAME	genre	book_price
1	101	"Da Vinci Code	The"	"Brown	0
2	102	Harry Potter and the Deathly Hallows	"Rowling	J.K."	0
3	104	Harry Potter and the Order of the Phoenix	"Rowling	J.K."	0
4	105	Fifty Shades of Grey	"James	E. L."	0
5	106	Harry Potter and the Goblet of Fire	"Rowling	J.K."	0
6	107	Harry Potter and the Chamber of Secrets	"Rowling	J.K."	0
7	108	Harry Potter and the Prisoner of Azkaban	"Rowling	J.K."	0
8	109	Angels and Demons	"Brown	Dan"	0
9	111	Fifty Shades Darker	"James	E. L."	0
10	112	Twilight	"Meyer	Stephenie"	0
11	113	"Girl with the Dragon Tattoo	The:Millennium Trilogy"	"Larsson	0
12	114	Fifty Shades Freed	"James	E. L."	0
13	115	"Lost Symbol	The"	"Brown	0
14	116	New Moon	"Meyer	Stephenie"	0
15	117	Deception Point	"Brown	Dan"	0
16	118	Eclipse	"Meyer	Stephenie"	0
17	119	"Lovely Bones	The"	"Sebold	0
18	120	"Curious Incident of the Dog in the Night-time	The"	"Haddon	0
19	121	Digital Fortress	"Brown	Dan"	0
20	122	"Short History of Nearly Everything	A"	"Bryson	0
21	123	"Girl Who Played with Fire	The:Millennium Trilogy"	"Larsson	0
22	124	Breaking Dawn	"Meyer	Stephenie"	0
23	125	"Very Hungry Caterpillar	The:The Very Hungry Caterpilla"Carle	"Carle	0

[12:59:50] New updates are available. Click here for details.

[13:00:15] Query finished in 0.001 second(s).