Database Forensics

Nelson, Chapter 12

Additional materials provided on Blackboard

Database Forensics

Data from Apps

Until this point, we have been recovering deleted files

The process is very analogous to computer forensics

But, in mobile devices

- *Almost all the user activity is done through apps*
- *Most of the apps use SQLite databases to store information*
- *It follows, therefore, that we need to examine the SQLite databases created by the Android apps*

Database Forensics

SQLite Databases

Can data be recovered from SQLite databases?

- *The answer is yes*
- *Deleted records are not deleted, but put into a freeblock list*
- *Records might eventually get overwritten, but no guarantee unless a "vacuum" command is issued*
  - Rebuilds database
  - Reclaims deleted records and space

Database Forensics

SQLite Database Carving

The process of recovering deleted information from a database is called *Database Carving*

We will use SQLite databases to demonstrate the concepts behind this

To understand how SQLite deleted records can be discovered, we need to understand the basic structure of the SQLite database

We will review this at a fairly high level

Database Forensics

SQLite: Storing and Managing Data

The SQLite database file is broken into “pages” of a fixed size

- *The size of each page is specified in the file header*
- *Each page may have one of many roles assigned by SQLite*
  - Lock-byte
  - Freelist
  - B-Tree
  - Payload overflow
  - Pointer map

Database Forensics

SQLite Database Header

- The first 100 bytes of the database file contains the database header
- In root page
- Note the “SQLite format 3” magic string
- Database page size
- Note that it contains pointer to freelist pages

Database Header Format

Offset	Size	Description
0	16	The header string: "SQLite format 3\000"
16	2	The database page size in bytes. Must be a power of two between 512 and 32768 inclusive, or the value 1 representing a page size of 65536.
18	1	File format write version. 1 for legacy; 2 for WAL.
19	1	File format read version. 1 for legacy; 2 for WAL.
20	1	Bytes of unused "reserved" space at the end of each page. Usually 0.
21	1	Maximum embedded payload fraction. Must be 64.
22	1	Minimum embedded payload fraction. Must be 32.
23	1	Leaf payload fraction. Must be 32.
24	4	File change counter.
28	4	Size of the database file in pages. The "in-header database size".
32	4	Page number of the first freelist trunk page.
36	4	Total number of freelist pages.
40	4	The schema cookie.
44	4	The schema format number. Supported schema formats are 1, 2, 3, and 4.
48	4	Default page cache size.
52	4	The page number of the largest root b-tree page when in auto-vacuum or incremental-vacuum modes, or zero otherwise.
56	4	The database text encoding. A value of 1 means UTF-8. A value of 2 means UTF-16le. A value of 3 means UTF-16be.
60	4	The "user version" as read and set by the user_version pragma.
64	4	True (non-zero) for incremental-vacuum mode. False (zero) otherwise.
68	4	The "Application ID" set by PRAGMA application_id.
72	20	Reserved for expansion. Must be zero.
92	4	The version-valid-for number.
96	4	SQLITE_VERSION_NUMBER

Database Forensics

Recovering Data in SQLite Databases

There are three places where data can be recovered in an SQLite database:

- *Within SQLite Database Image*
  - B-Tree pages
    - *Freeblock list*
    - *Unallocated space*
  - Free List pages
- *Write-Ahead Logs*
  - Separate files
  - It has same name as database file, but with –wal appended
  - Contains records not yet committed to the database

SQLite Database Internals

SQLite Database Page Roles

Lock-byte Page

- *Used by OS Interface (VFS) in implementing database locking*
- *Not relevant to database recovery*

Freelist Page

- *Unused pages*
- *Created when information is deleted from database*
- *Reused when additional pages are required*

SQLite Database Internals

SQLite Database Page Roles

Payload Overflow Page

When payload is too large for a B-Tree page

Data overflows onto pages of this type

Pointer Map Pages

Makes operation of **vacuum** *operation more efficient.*

SQLite Database Internals

Database Page Types

B-Tree Page

Used to store key and data storage for active database pages

Organized as a B-Tree

Self-balancing tree data structure

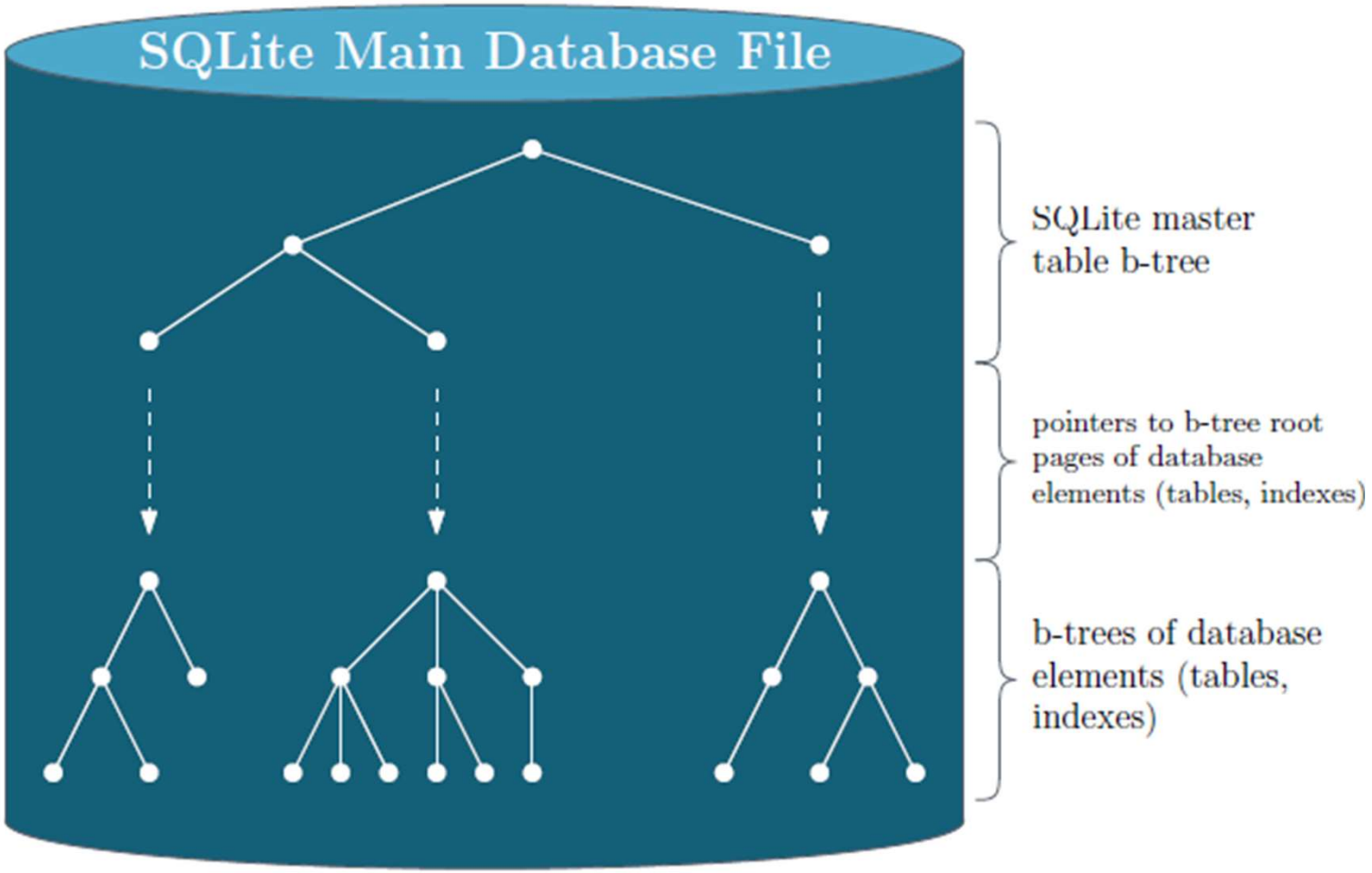
Enables efficient access, insertion and deletion

Types of B-Tree pages

Table Interior	(describes table schema)
Table leaf	(contains table data/payload)
Index Interior	(schema for indices)
Index leaf	(indices)

SQLite Database Internals

B-Tree Pages



SQLite Database Internals
SQLite B-Tree Logical Structure

B-Tree Page Hierarchy

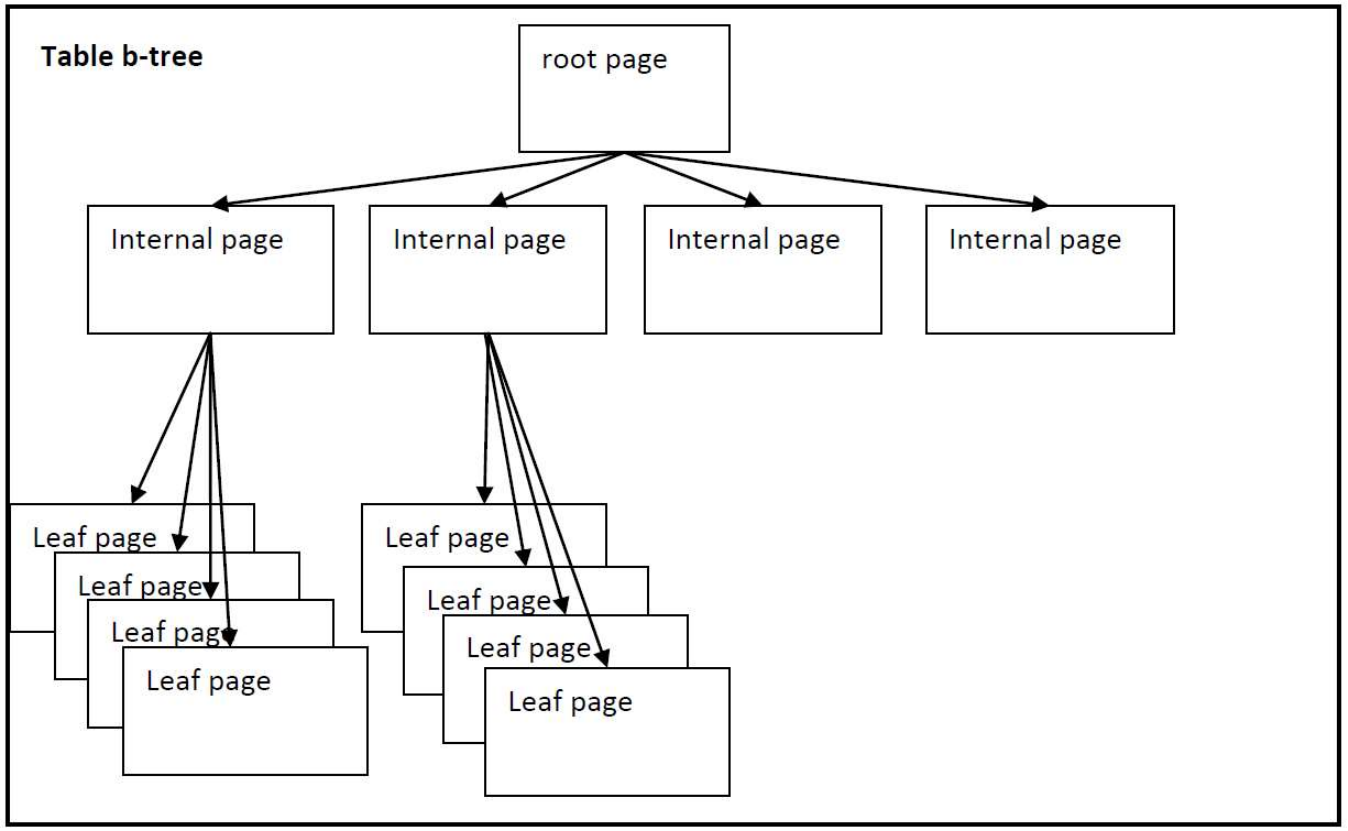


Figure 1. B-tree page hierarchy

SQLite Database Data Recovery

Recovering Data in B-Tree Pages

Data can be recovered in the following areas in a B-Tree page:

- *Freeblock*
  - Database records are designated as a *freeblock* when deleted
    - *Record header is overwritten with size of record and offset to next freeblock*
  - Freeblock records are arranged in a list
- *Unallocated space*
  - Fragments of deleted records may be found here
  - Can occur during defragmentation process

Database Data Recovery

SQLite B-Tree Logical Structure

B-Tree Page Layout

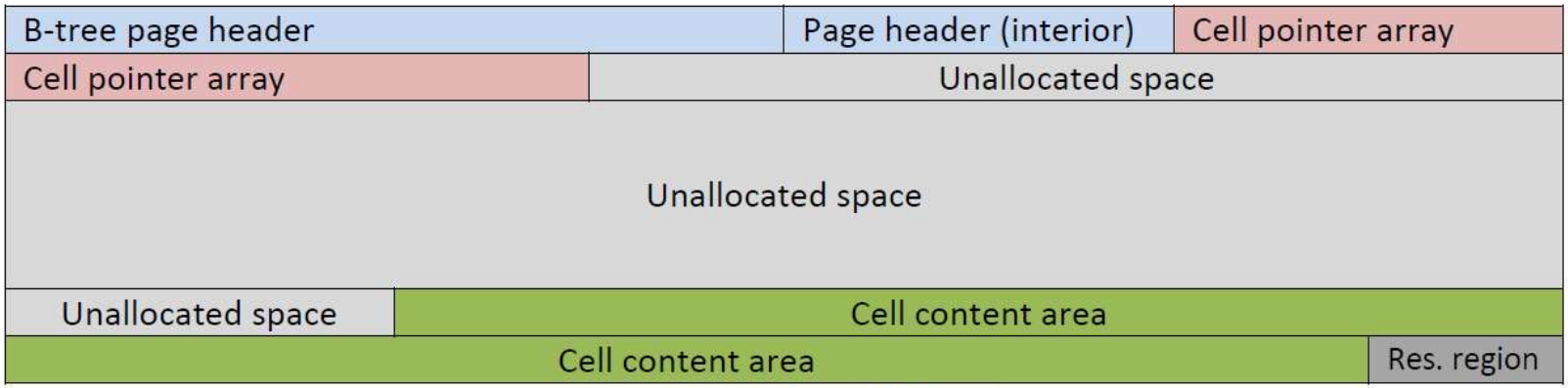


Figure 2. Sqlite b-tree page layout

B-Tree Page Header	
Byte 0	B-tree type (e.g., 0xD is leaf table)
Bytes 1-2	Byte offset into free freeblock
Bytes 3-4	Number of cells on page
Bytes 5-6	Offset into first byte of cell content area
Byte 7	Number of fragmented free bytes in cell area content
Bytes 8-11	Right most pointer (interior pages only)

Points to first block of freeblock list in Cell Content area

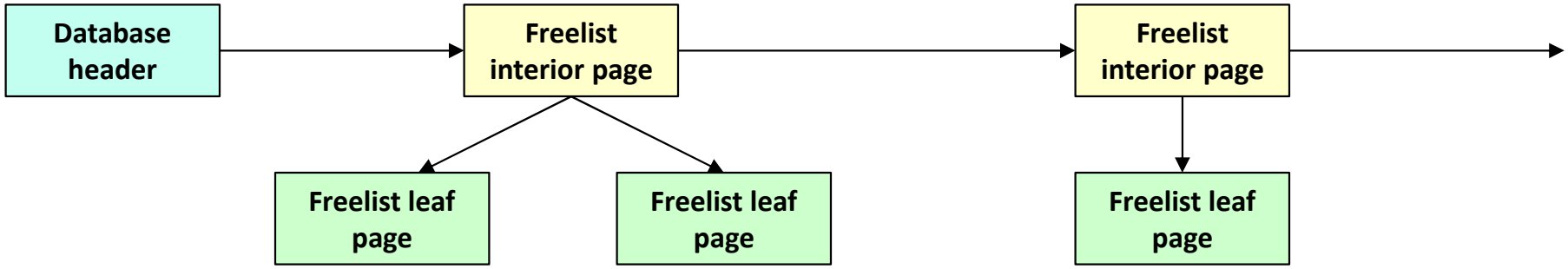
Database Data Recovery

SQLite Free List

When information is deleted from the database that frees up an entire page, it is stored on a freelist

Stored as a linked list of freelist pages

Beginning of list is at database header



Database Data Recovery

Write-Ahead Logs

Write-ahead logs (WALs) contain records not yet committed to the main database

- *Separate files with same database name and –wal appended*
- *There may be multiple WALs*

Typical scenario:

- *New or altered records are stored in a WAl*
- *Those records stay there until a "Checkpoint" event occurs (often, when the WAL reaches 1000 pages)*
- *Until then, the database will read new data from the WAL instead of using the main database.*

Database Data Recovery

Write-Ahead Logs

Why are WALs considered part of database forensics?

Consider:

- *The database contains an SMS/MMS chat information*
- *An entire chat session may never trigger a single SQLite Checkpoint*
- *All the chat info would NOT be in the main database file, but in the WAL file(s)*
- *A suspect's main SMS/MMS database could be empty as a result of a recent SQLite cleanup and standard SQLite database readers may not display any information*
- *Special SQLite forensic tools will recover the records in the WAL*

Database Data Recovery

SQLite Page and Record Recovery

So, where is the deleted data?

Freelist page list

Granularity: database page

To find list: Start from Database header

Freeblock list

Granularity: < database page; minimum 4 bytes

To find list: Start from B-tree page header

Unallocated space

Granularity: < database page;

To find list: deduce from B-tree page header information

Database Data Recovery

Write-Ahead Logs

So, where is the uncommitted data?

Write-ahead log(s)

Granularity: separate database files

Location: Same directory as main database file

To find WAL: Look for files with –WAL embedded in name

Database Data Recovery

SQLite Deleted Data Methodology

The basic methodology for recovering data in SQLite databases involves:

- *Find SQLite database ("SQLite Format 3" magic string if database was deleted)*
  - Locate beginning of Freelist
  - Traverse freelist records, looking for valid records
- *Database carving to find leaf nodes (0xD signals start of header)*
  - Use B-tree page headers to find freeblock list
  - Find unallocated area
  - More database carving based on tags, heuristics,etc. to rebuild records

Database Forensics Labs

Examining SQLite Forensic Corpus using:

DB Browser for SQLite

SQLite Forensic Explorer

SQLite Database Recovery

Autopsy (before and after SQLite plugins installation)

FQLite

Database Forensics Labs

Initial Setup

We'll be using a subset of databases that have been put together to practice SQLite database forensics

SQLite Forensic Corpus

https://digitalcorpora.org/corpora/sql/sqlite-forensic-corpus/

Stated purpose is to "evaluate and benchmark analysis methods and tools"

Corpus contains databases contain hidden/deleted records

It has been enhanced using anti-forensic techniques

Anti-Forensics

Definitions:

Attempts to negatively affect the existence, amount and/or quality of evidence from a crime scene or make the analysis and examination of evidence difficult or impossible to conduct – Marc Rogers (Purdue)

The removal, or hiding, of evidence in an attempt to mitigate the effectiveness of a forensics investigation – Phrack Magazine (2002)

Anti-forensics is more than a technology. It is an approach to criminal hacking that can be summed up like this: Make it hard for them to find you and impossible for them to prove they found you – Scott Berinato ("The Rise of Anti-Forensics"

Destroying ESI (Electronically Stored Information) that's potential evidence – Nelson text

Anti-Forensics is sometimes called *Anti-Computer Forensics*

Anti-Forensics

Anti-Forensics is often broken down into four sub-categories

- *Data Hiding*
  - E.g., encryption/steganography
- *Artifact wiping*
  - E.g., disk cleaning, file wiping, passing magnet over HDD
- *Trail obfuscation*
  - E.g., Metasploit
    - *Timestomp: gives ability to modify creation, access and modification time/dates*
    - *Transmogrify: modify file headers (e.g., jpg files)*
- *Attacks against the Computer Forensics process & tools*
  - E.g., target the integrity of the hash taken on forensic copy of evidence

Database Forensics Labs

SQLite Forensic Corpus

The SQLite [Forensic] Corpus is divided into 15 different categories

0-9: Test support for SQLite database features

A-E: Test support for deleted or overwritten records

11-19: Anti-forensic support

I've made available databases from the **A-E** categories as well as **17-18** (associated with recovering data in unallocated space)

Database Forensics Labs

Initial Setup

Create a Database Forensics working directory in your own working space (we'll call it <*Wdir*>)

Copy over the following files to <*Wdir*>

R:\Share\Labs\SQLite\SQLite Corpus

Copy over the following directory to <*Wdir*>

R:\Share\Tools\SQLite Tools

You should now have two subdirectories under <*Wdir*>

SQLite Corpus

SQLite Tools

Database Forensics Labs

Forensic examination using

DB Browser for SQLite

Database Forensics Labs

DB Browser for SQLite

We'll start out by using the official DB Browser for SQLite databases

DB Browser for SQLite

This can be found in the SQLite Tools subdirectory of your <*Wdir*> folder

We'll first need to install the tool

Database Forensics Labs

Install DB Browser for SQLite

Go to your *SQLiteTools/DB Browser for SQLite* subdirectory in your *<Wdir>* folder

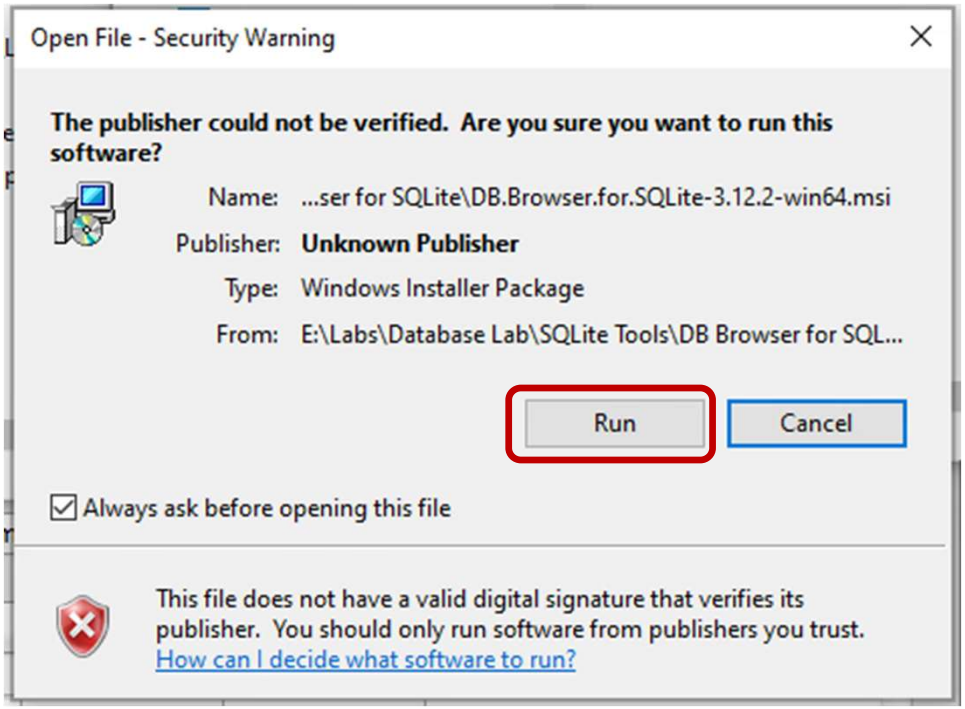
Double-click on the following file to start the installation:

`DB.Browser.for.SQLite-3.12.2-win64.msi`

Database Forensics Labs

Install DB Browser for SQLite

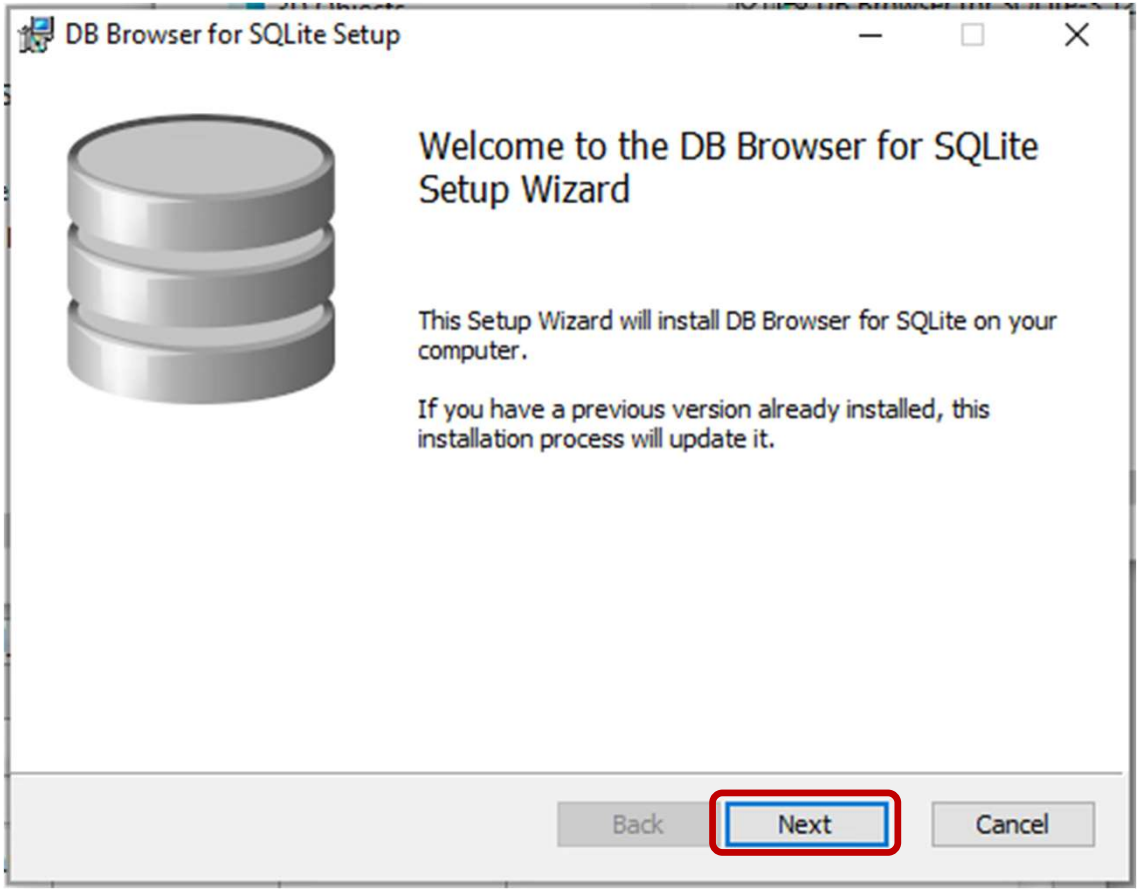
Click the “Run” button when the Security Warning pop-up window appears



Database Forensics Labs

Install DB Browser for SQLite

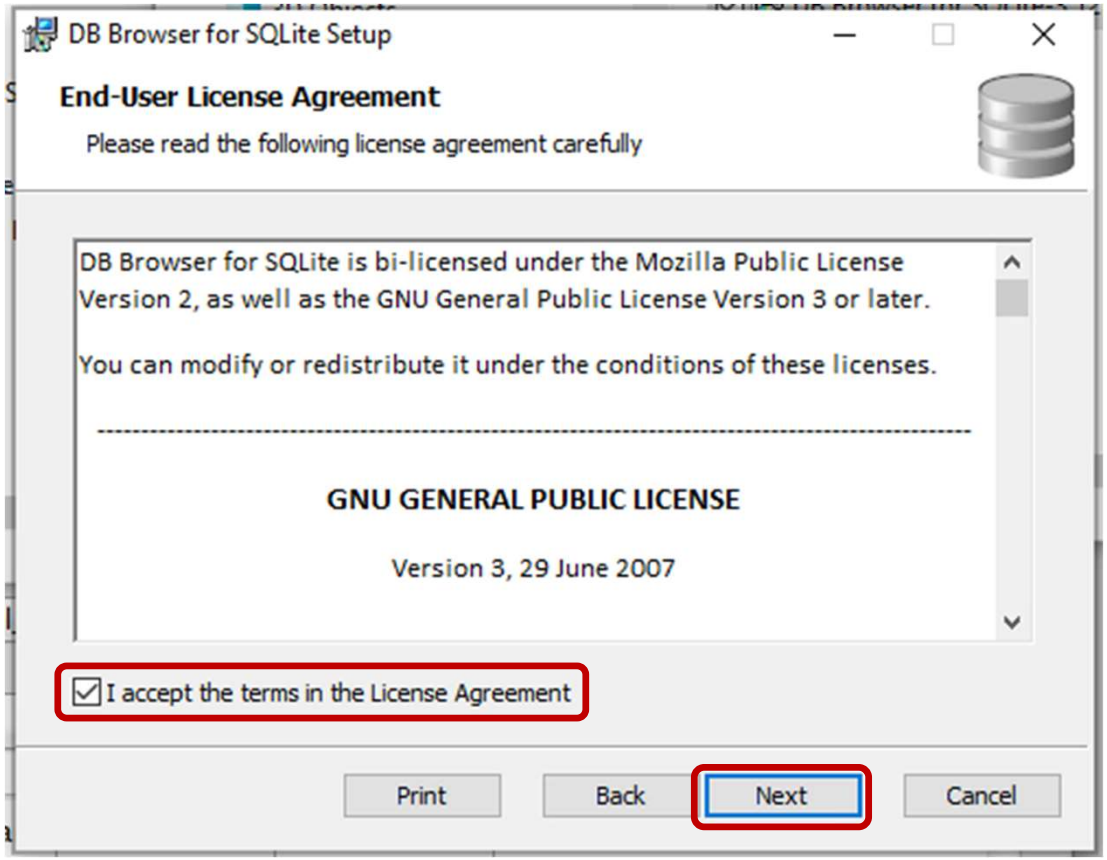
Click “Next” button when the Welcome window appears



Database Forensics Labs

Install DB Browser for SQLite

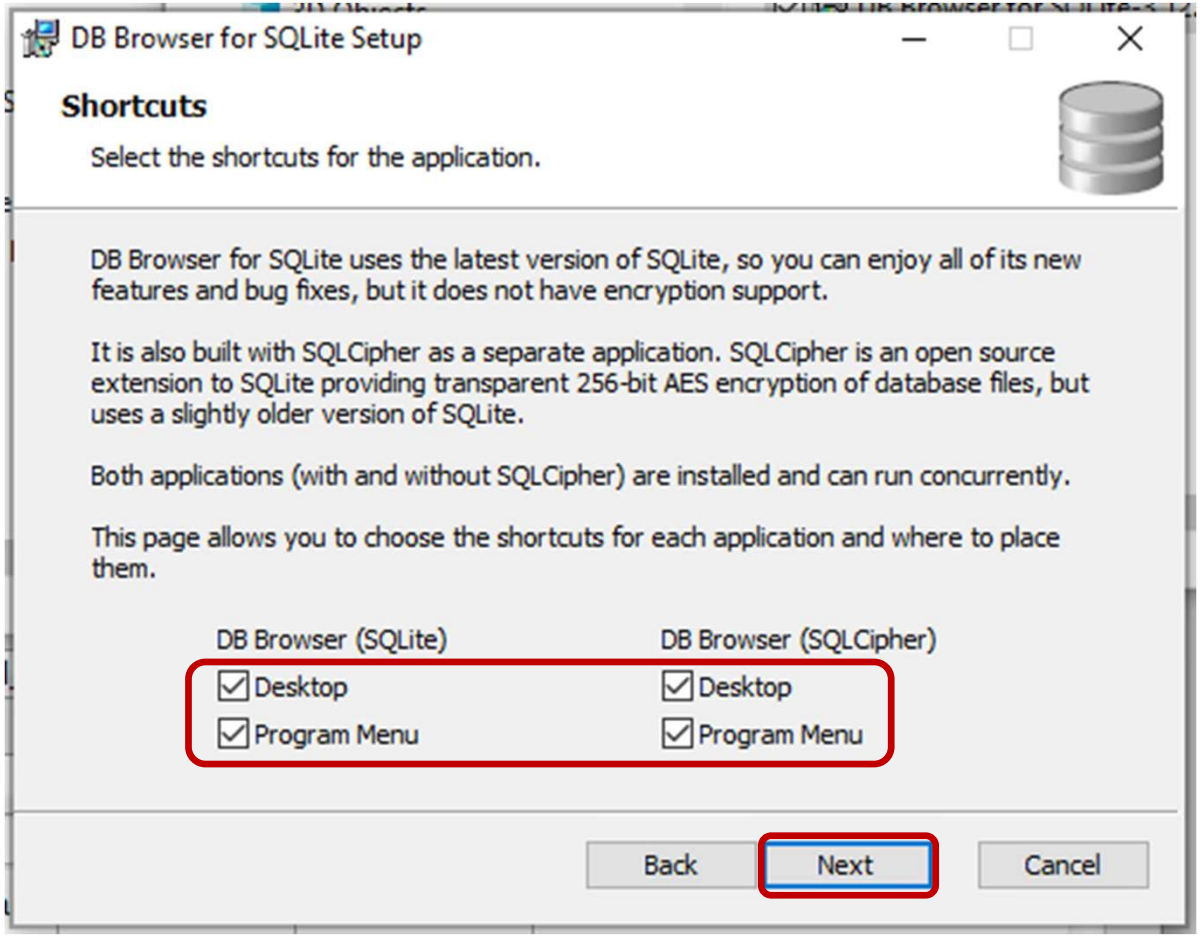
Accept the GNU General Public License Agreement, then click “Next”



Database Forensics Labs

Install DB Browser for SQLite

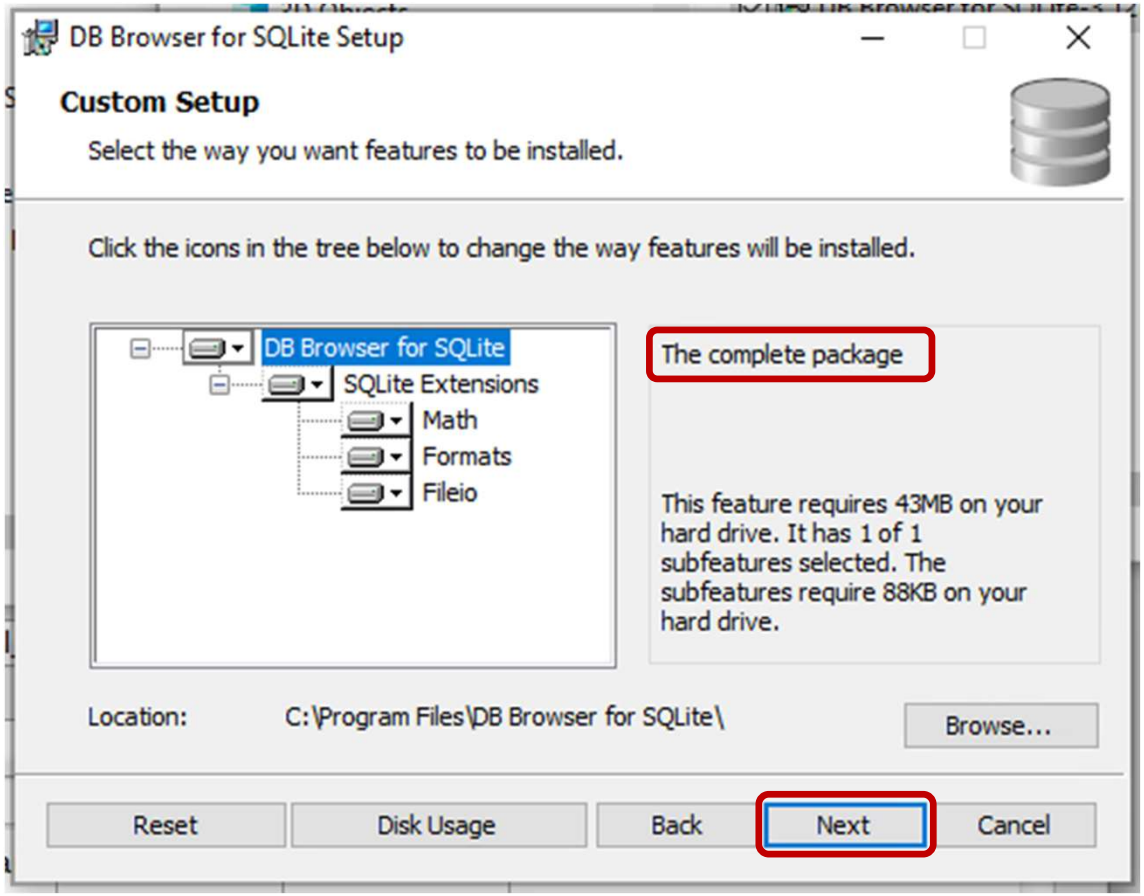
Click all 4 boxes and then click “Next”



Database Forensics Labs

Install DB Browser for SQLite

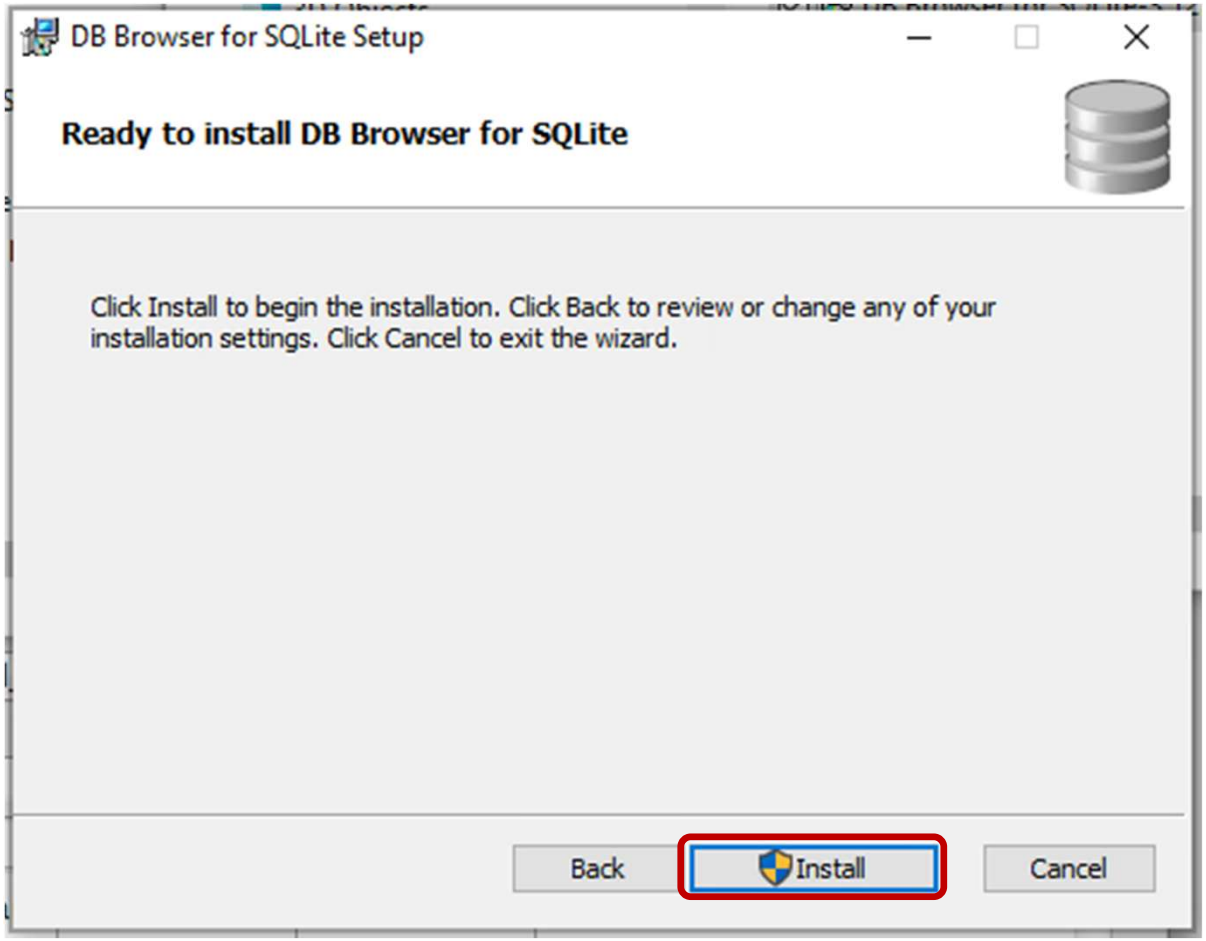
Make sure it says “The complete package”, then click “Next”



Database Forensics Labs

Install DB Browser for SQLite

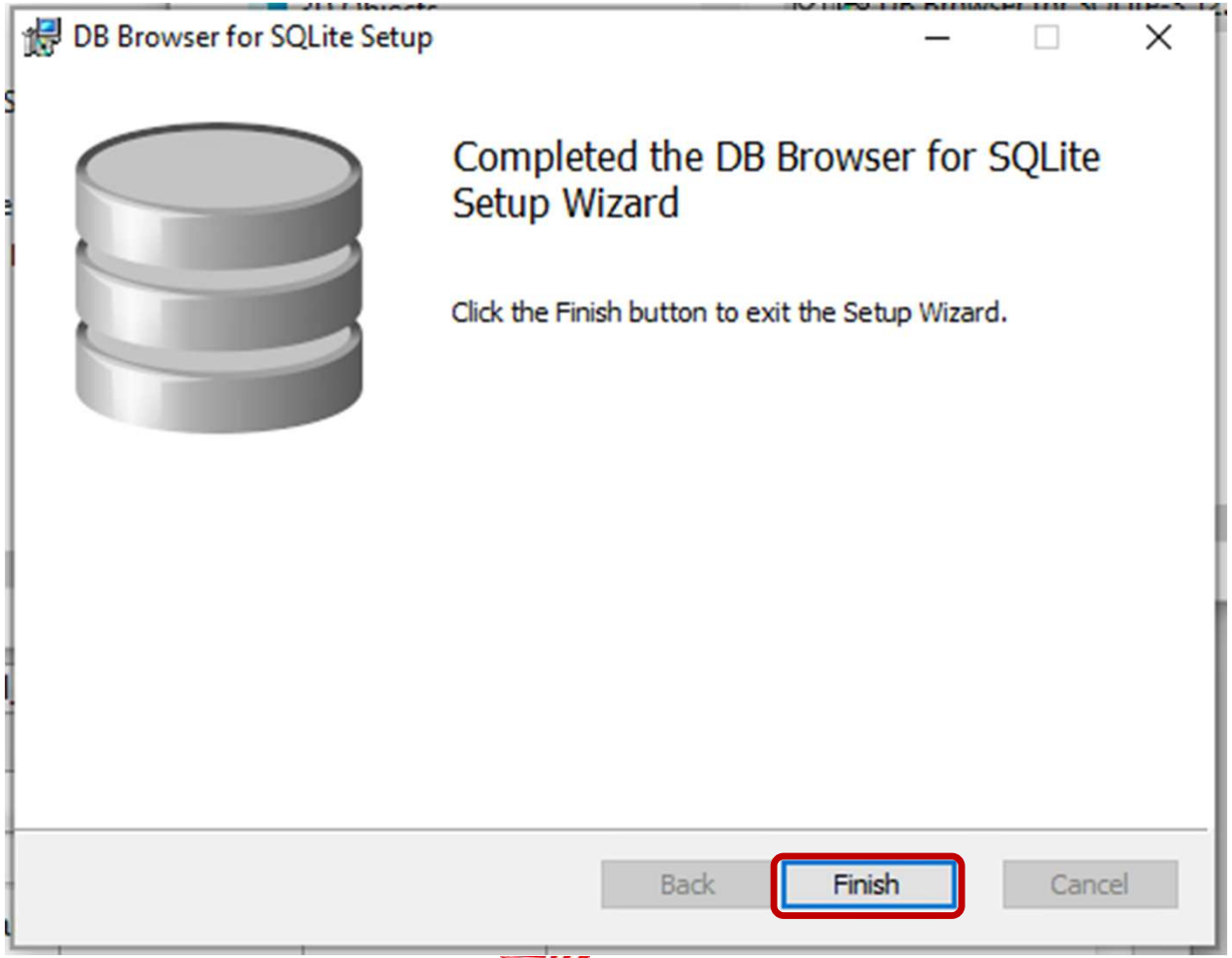
Click “Install”



Database Forensics Labs

Install DB Browser for SQLite

Click “Finish” when the installation has completed

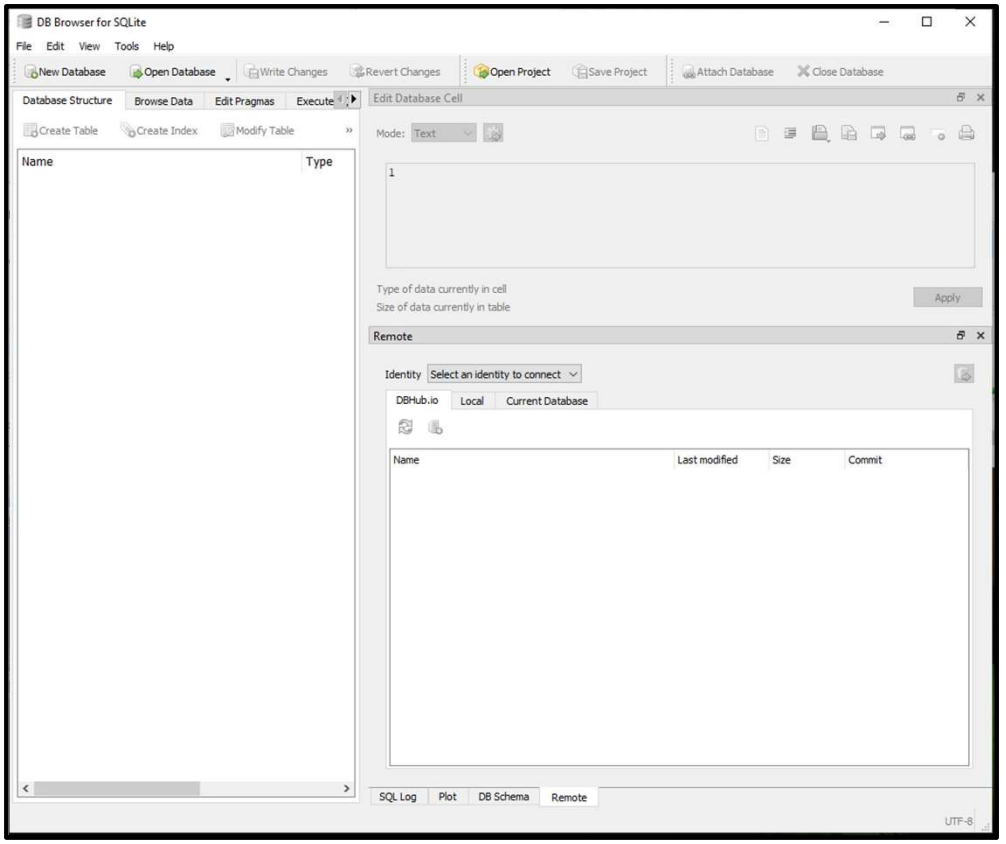


Database Forensics Labs

Run DB Browser for SQLite

Run the DB Browser for SQLite tool by clicking on the Windows start button and locating and clicking on the “DB Browser (SQLite)” app

A window much like the one to the right should open



DB Browser for SQLite

Deleted Tables

- We'll start by opening a database (0A-05.db) where tables and records within those tables have been deleted
- See the SQL file to the right to see what was put in the database and what was deleted

```
CREATE TABLE users (
        'id' INT UNSIGNED NOT NULL,
        'name' TEXT NOT NULL,
        'surname' TEXT NULL,
        'codeA' INT NULL,
        'codeB' FLOAT NULL
);

CREATE TABLE members (
        'mid' INT UNSIGNED NOT NULL,
        'mname' TEXT NOT NULL,
        'msurname' TEXT NULL,
        'mcodeA' INT NULL,
        'mcodeB' FLOAT NULL
);

INSERT INTO users
        (id, name, surname, codeA, codeB)
        VALUES
        (50001, 'Maja', 'Lang', -979099654, 694130400.98461),
        (50002, 'Liam', 'Franke', -1026737640, 613028297.95192),
        (50003, 'Leni', 'Groß', 1643371695, -3428082023.01528),
        (50004, 'Elisabeth', 'Mayer', 289485196, 107790182.86047),
        (50005, 'Claudia', 'Brandt', 682453881, 4895471814.03985),
        (50006, 'Kurt', 'Müller', 277430878, -3213573693.39726),
        (50007, 'Elias', 'Sauer', 715668940, -2734937992.75483),
        (50008, 'Else', 'Vogel', -364818672, 1959408562.14585),
        (50009, 'Tobias', 'Brandt', 36208673, -2418977668.23833),
        (50010, 'Anna', 'Berger', -400626611, 3459240369.71258);

INSERT INTO members
        (mid, mname, msurname, mcodeA, mcodeB)
        VALUES
        (20001, 'Gertrud', 'Busch', 1866760044, 668770743.37973),
        (20002, 'Thomas', 'Neumann', 94561521, 1099693540.53014),
        (20003, 'Hanna', 'Herrmann', -1489592963, 676806483.26725),
        (20004, 'Katharina', 'Frank', -942844261, 2099325608.97401),
        (20005, 'Christine', 'Seidel', 1666394471, 2008802563.42180),
        (20006, 'Anne', 'Böhm', -181378817, -4445543153.88461),
        (20007, 'Lukas', 'Fischer', -819379123, -2827242011.12392),
        (20008, 'Mika', 'Lehmann', -163995928, 714404175.85634),
        (20009, 'Jens', 'Fuchs', 384961186, -3658189642.54958),
        (20010, 'Jens', 'Fischer', -1182984342, -4895035023.75215);

PRAGMA secure_delete=0;
PRAGMA secure_delete;

DELETE FROM users where id == 50007;
DELETE FROM users where id == 50010;
DELETE FROM members where mid == 20001;
DELETE FROM members where mid == 20006;
DELETE FROM users where id == 50005;
DELETE FROM members where mid == 20004;
DELETE FROM members where mid == 20002;
DELETE FROM users where id == 50003;
DELETE FROM members where mid == 20007;
DELETE FROM members where mid == 20003;
DELETE FROM users where id == 50001;
DELETE FROM users where id == 50006;
DELETE FROM members where mid == 20010;
DELETE FROM members where mid == 20005;
DELETE FROM users where id == 50004;
DELETE FROM users where id == 50002;
DELETE FROM members where mid == 20009;
DELETE FROM members where mid == 20008;
DELETE FROM users where id == 50009;
DELETE FROM users where id == 50008;

DROP TABLE users;
DROP TABLE members;
```

DB Browser for SQLite

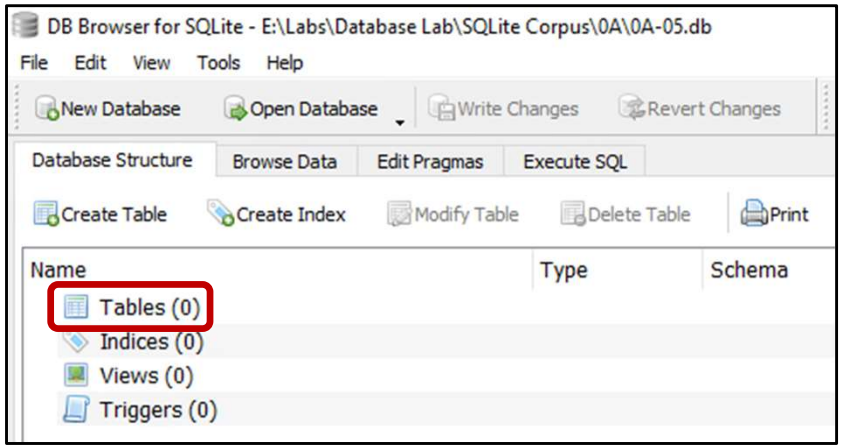
Deleted Tables

In DB Browser for SQLite, click on Open Database

Navigate to the <Wdir>\SQLite Corpus\0A\0A-05.db file

Click Open

You should now see the following database summary



Note that the database is empty (0 tables, etc.)

DB Browser for SQLite

Summary

DB Browser for SQLite is not designed to recover forensic data

Rather, it is designed to display the database contents that result from valid SQL commands

Therefore

Deleted tables and records do not appear

Database Forensics Labs

Forensic examination using

SQLite Forensic Explorer

Database Forensics Labs

SQLite Forensic Explorer

We'll next evaluate a trial version of a commercial SQLite forensics tool

SQLite Forensic Explorer

This can be found in the SQLite Tools subdirectory of your <*Wdir*> folder

We'll first need to install the tool

Database Forensics Labs

Install SQLite Forensic Explorer

Go to your *SQLiteTools/SQLite Forensic Explorer* subdirectory in your *<Wdir>* folder

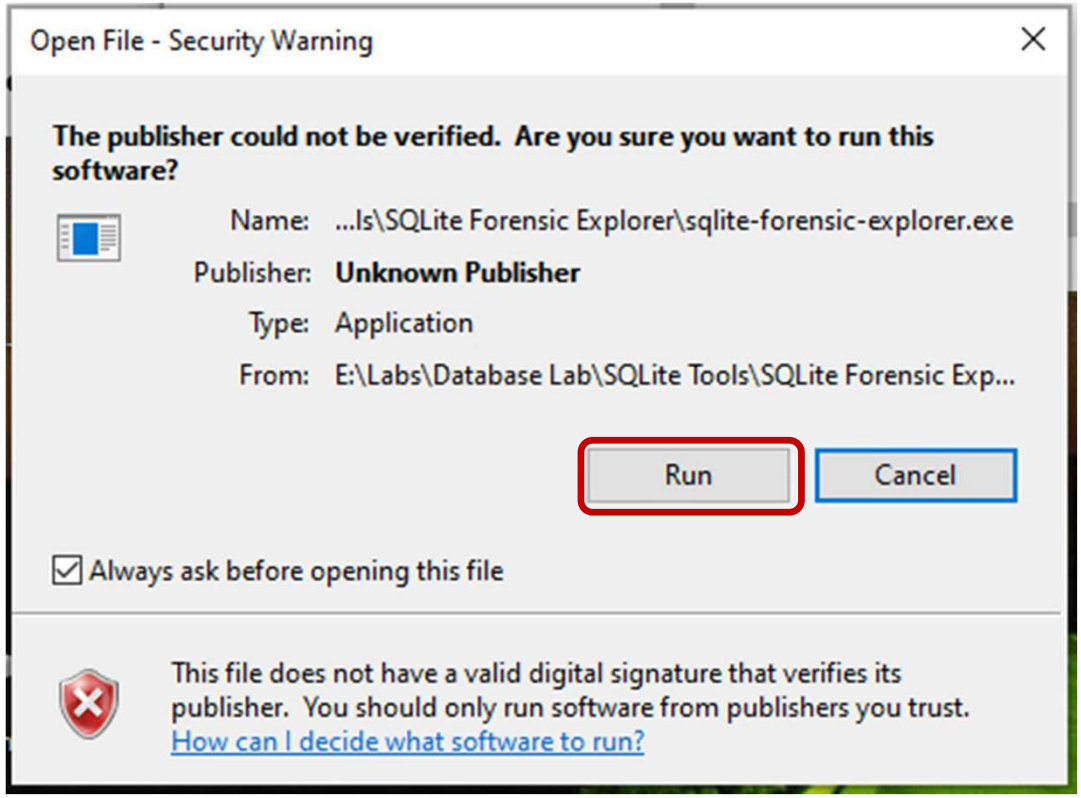
Double-click on the following file to start the installation:

`sqlite-forensic-explorer.exe`

Database Forensics Labs

Install SQLite Forensic Explorer

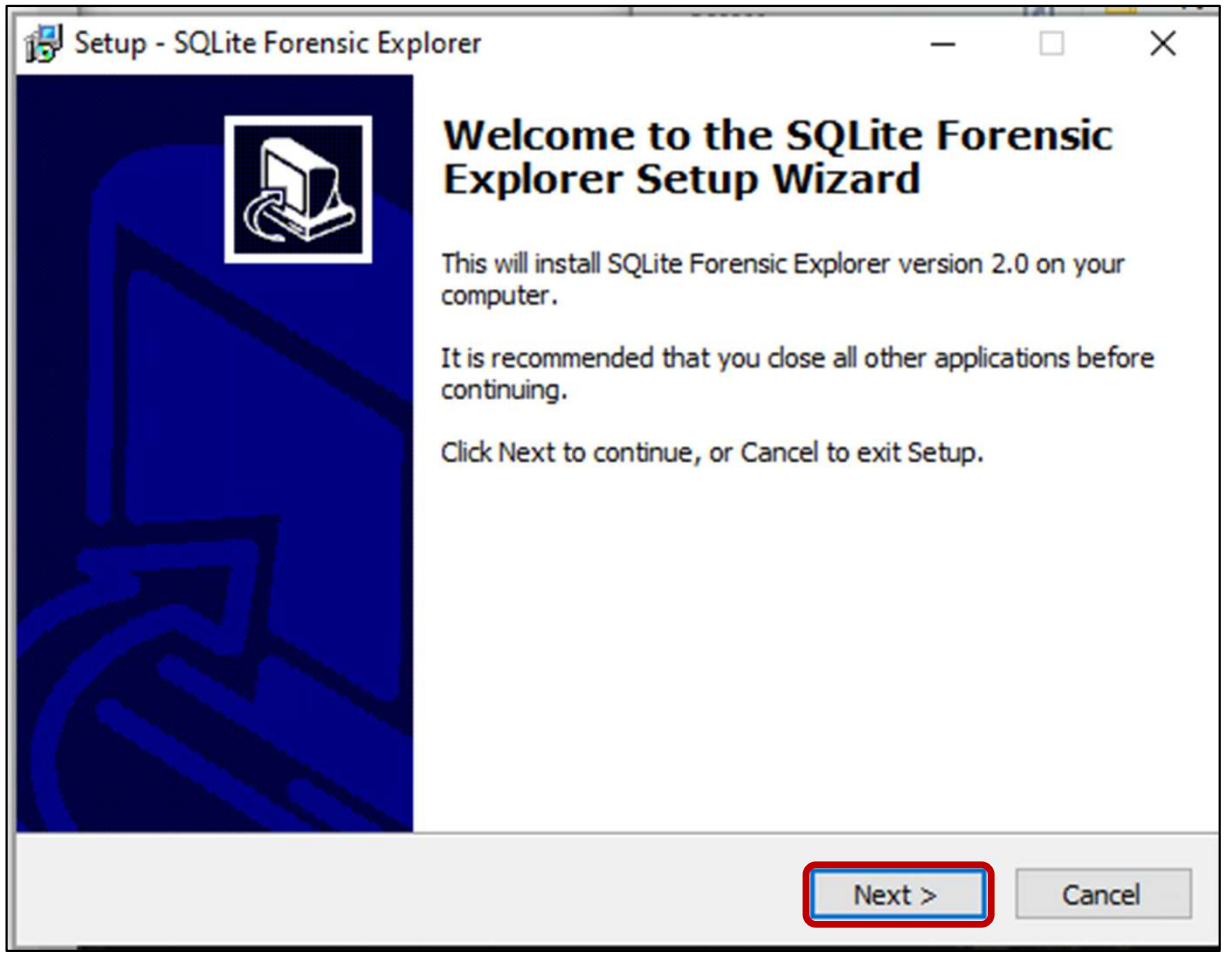
Click the “Run” button when the Security Warning pop-up window appears



Database Forensics Labs

Install SQLite Forensic Explorer

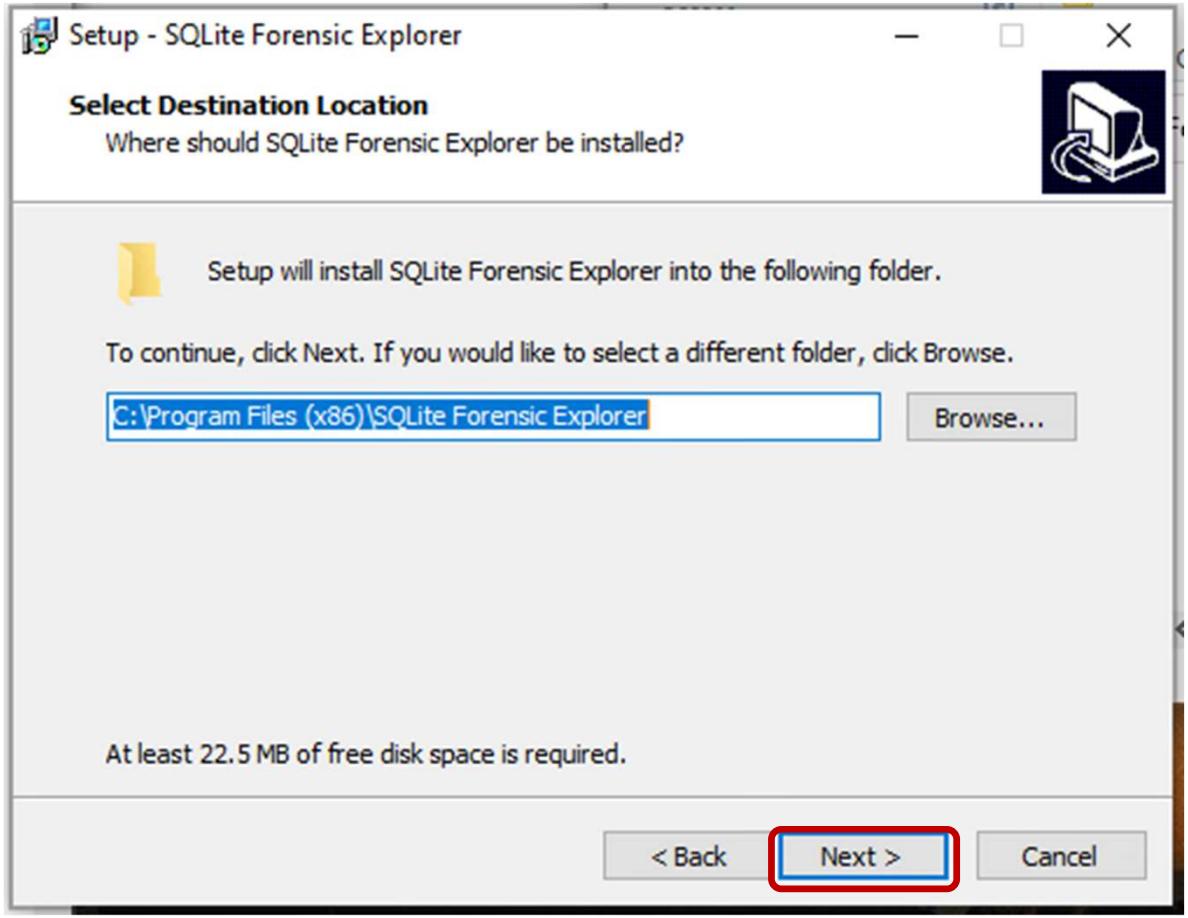
Click “Next” button when the Welcome window appears



Database Forensics Labs

Install SQLite Forensic Explorer

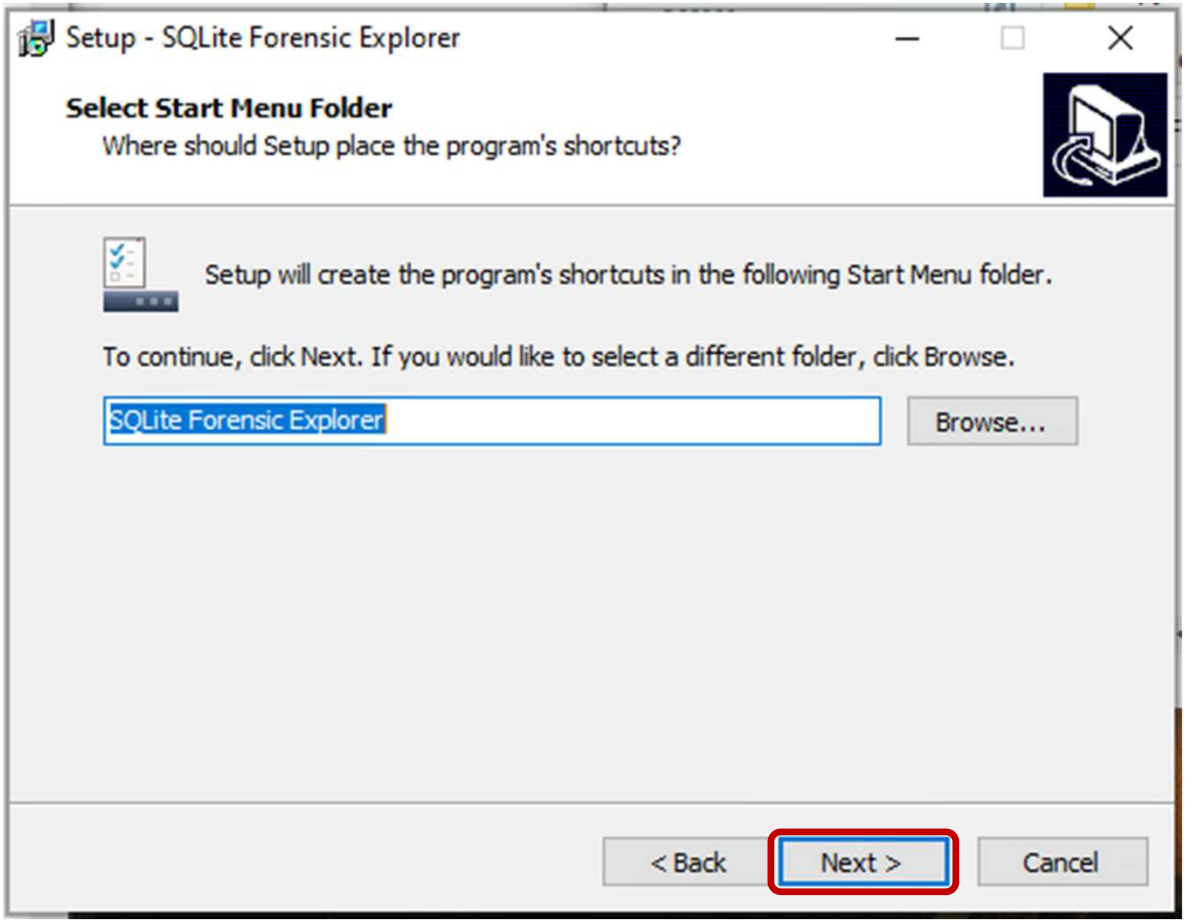
Click “Next” to select the default installation directory



Database Forensics Labs

Install SQLite Forensic Explorer

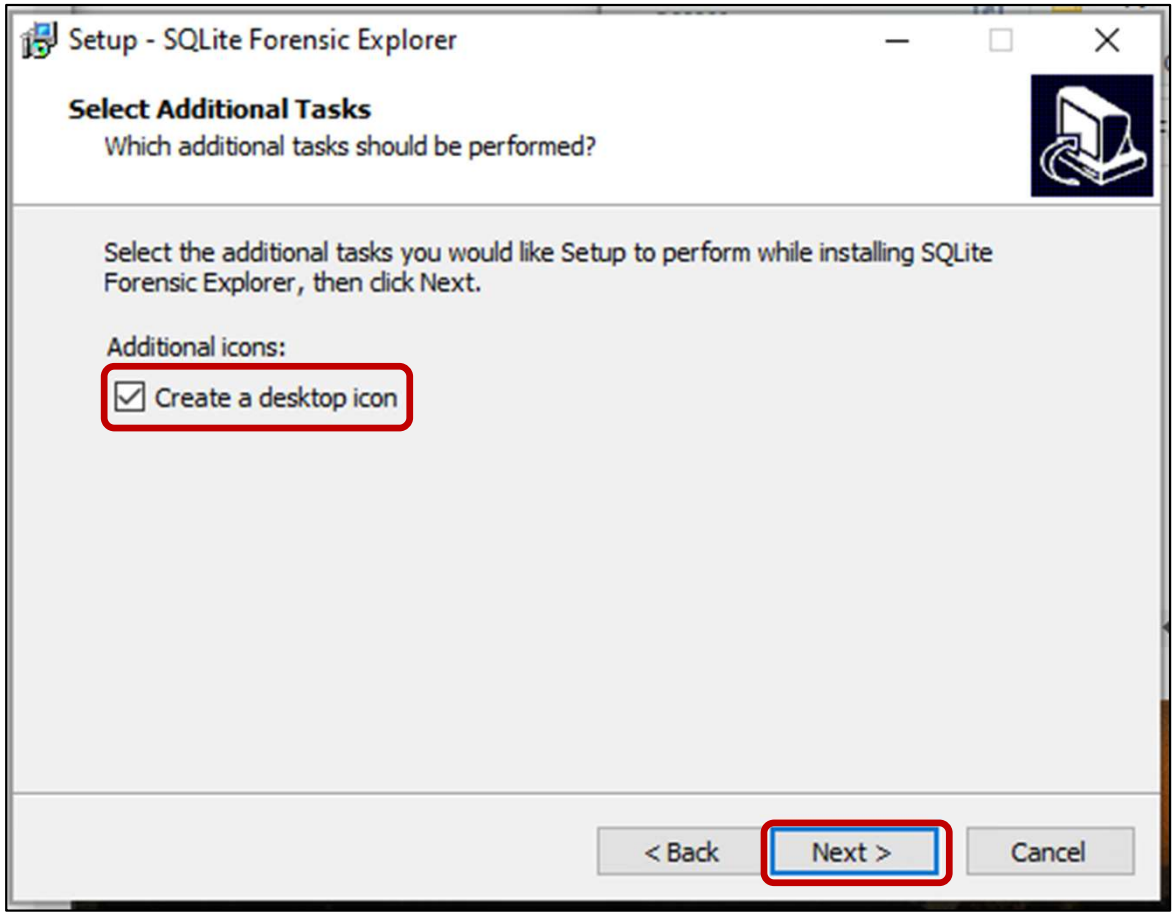
Click “Next” to select the default Start Menu folder



Database Forensics Labs

Install SQLite Forensic Explorer

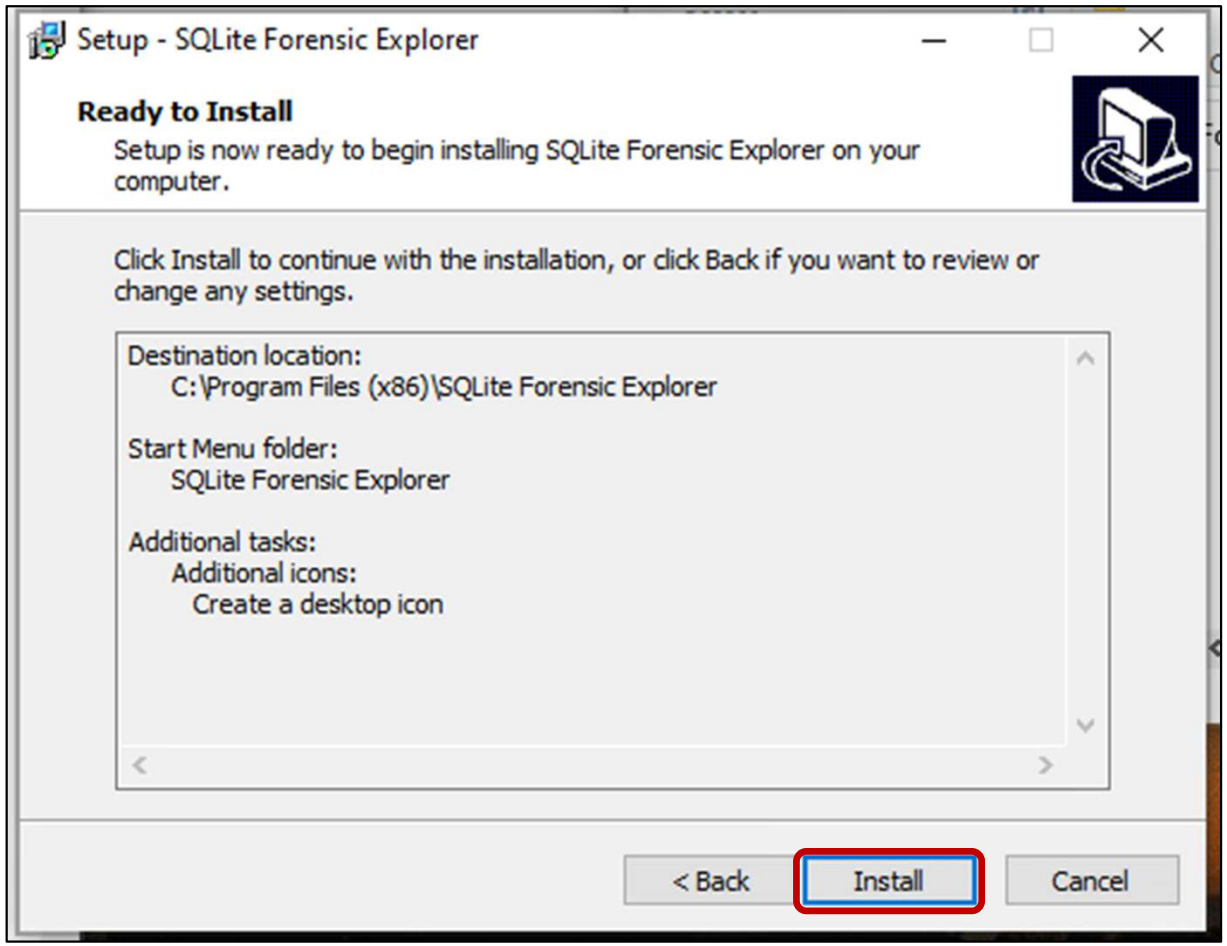
Select the “Create a desktop icon” checkbox and click “Next”



Database Forensics Labs

Install SQLite Forensic Explorer

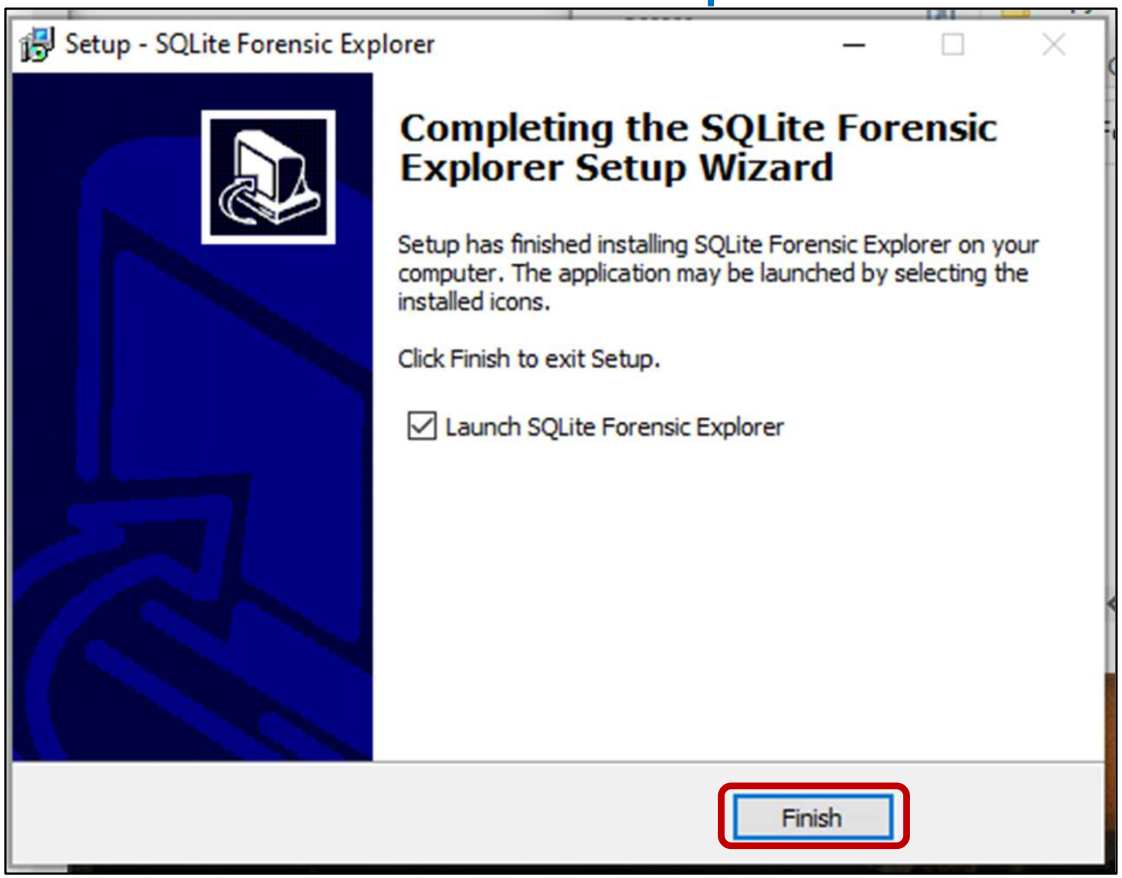
Click “Install” to begin the installation



Database Forensics Labs

Launch SQLite Forensic Explorer

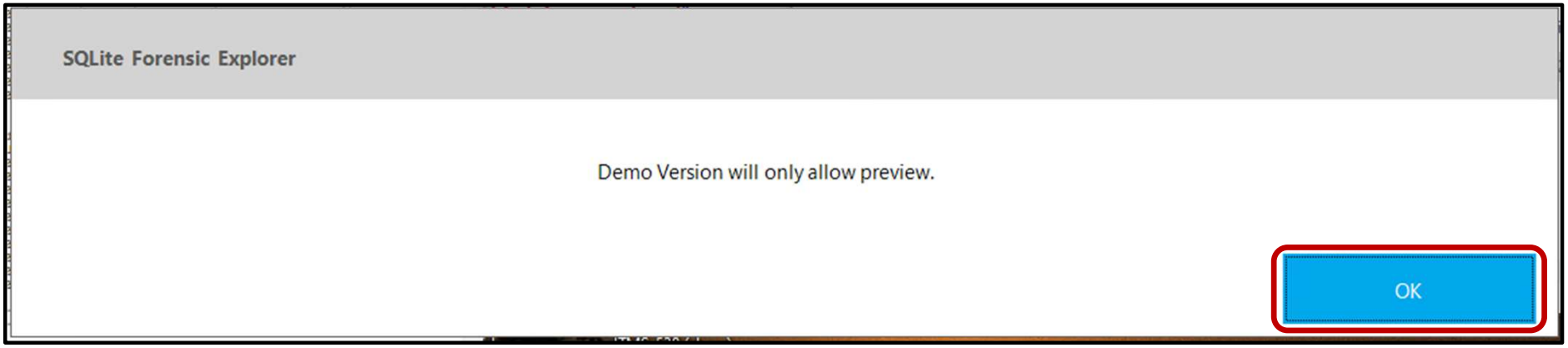
Click “Finish” when the installation has completed to launch the SQL Forensic Explorer



Database Forensics Labs

Launch SQLite Forensic Explorer

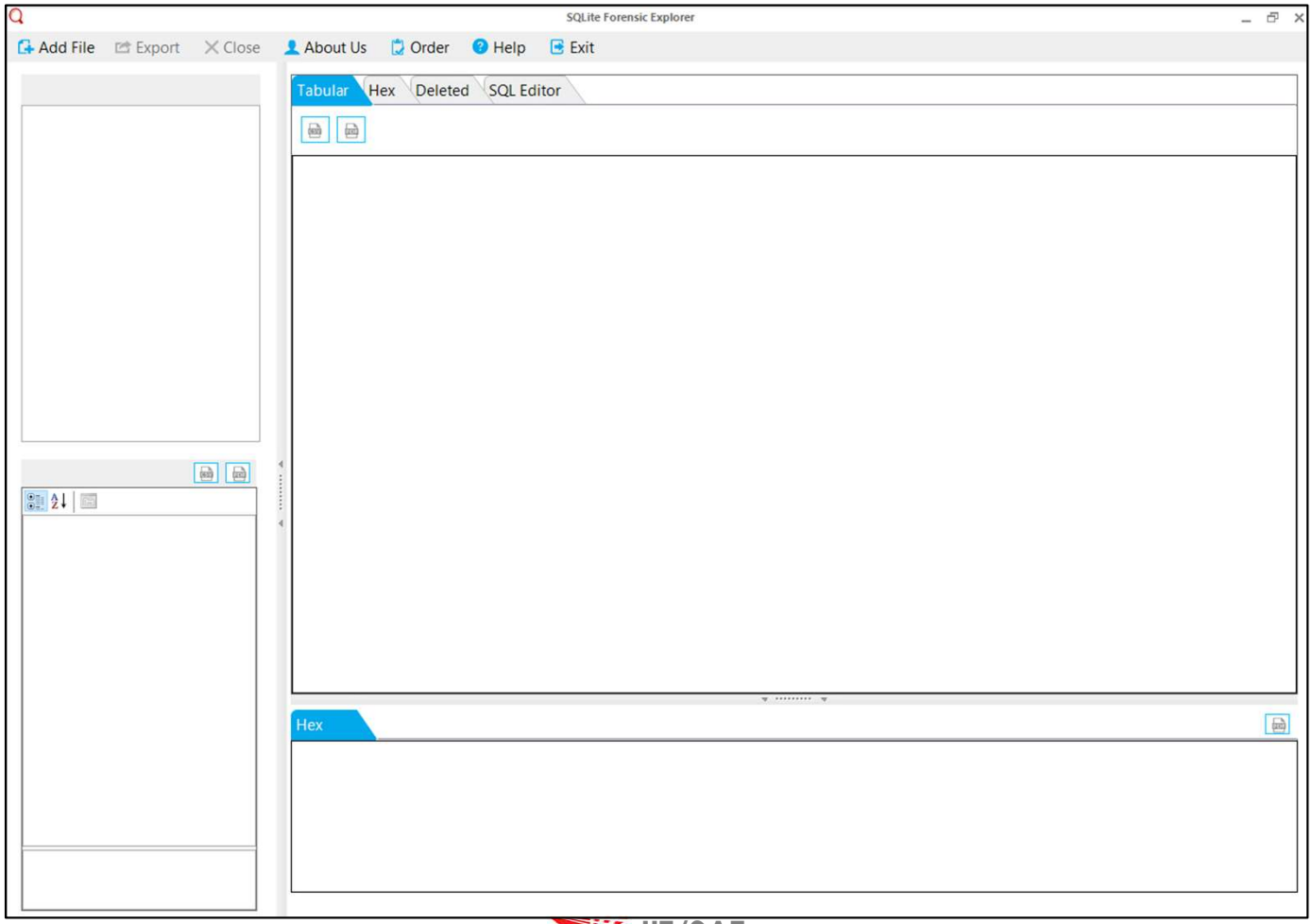
Click “OK” on the pop-up window that indicates this application is a preview (preview only)



Database Forensics Labs

Launch SQLite Forensic Explorer

You should now see a window as shown below:



SQLite Forensic Explorer

Deleted Tables

In SQLite Forensic Explorer, click on Add File

Click on the ellipsis (…) to the right of the SQLite File text box and navigate to the <Wdir>\SQLite Corpus\0A\0A-05.db file

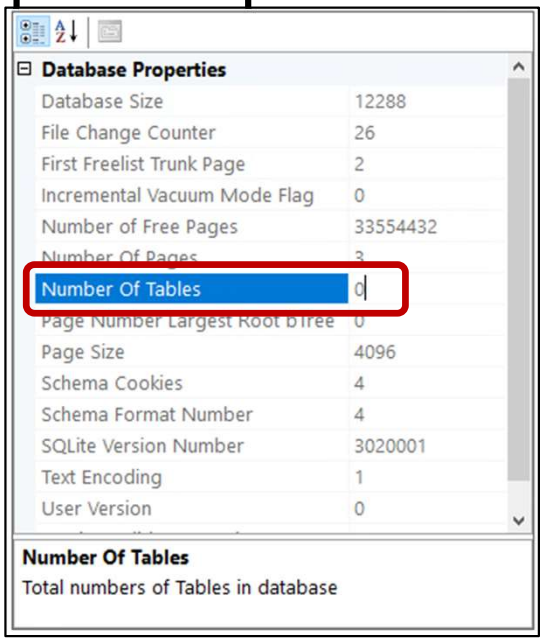
Click Open

Click Add

SQLite Forensic Explorer

Deleted Tables

You should now see the following database summary in the Database Properties panel on the lower left



Note that the database is empty (0 tables)

SQLite Forensic Explorer

Deleted Tables

Note that there is a Deleted Records category in the Tree Viewer on the upper right

Click on Deleted Records

Note that it takes you to a Deleted tab on the upper right

Also note that no deleted records have been detected

SQLite Forensic Explorer

Summary

SQLite Forensic Explorer IS designed to recover forensic data

However, it doesn't recover all the forensically useful data in our databases

We have observed that

Deleted tables are not recovered (0A-05.db)

Database Forensics Labs

Forensic examination using

SQLite Database Recovery

Database Forensics Labs

SQLite Database Recovery

We'll next evaluate a trial version of a commercial SQLite forensics tool

SQLite Database Recovery

This can be found in the SQLite Tools subdirectory of your <*Wdir*> folder

We'll first need to install the tool

Database Forensics Labs

Install SQLite Database Recovery

Go to your *SQLiteTools/SQLite Database Recovery* subdirectory in your *<Wdir>* folder

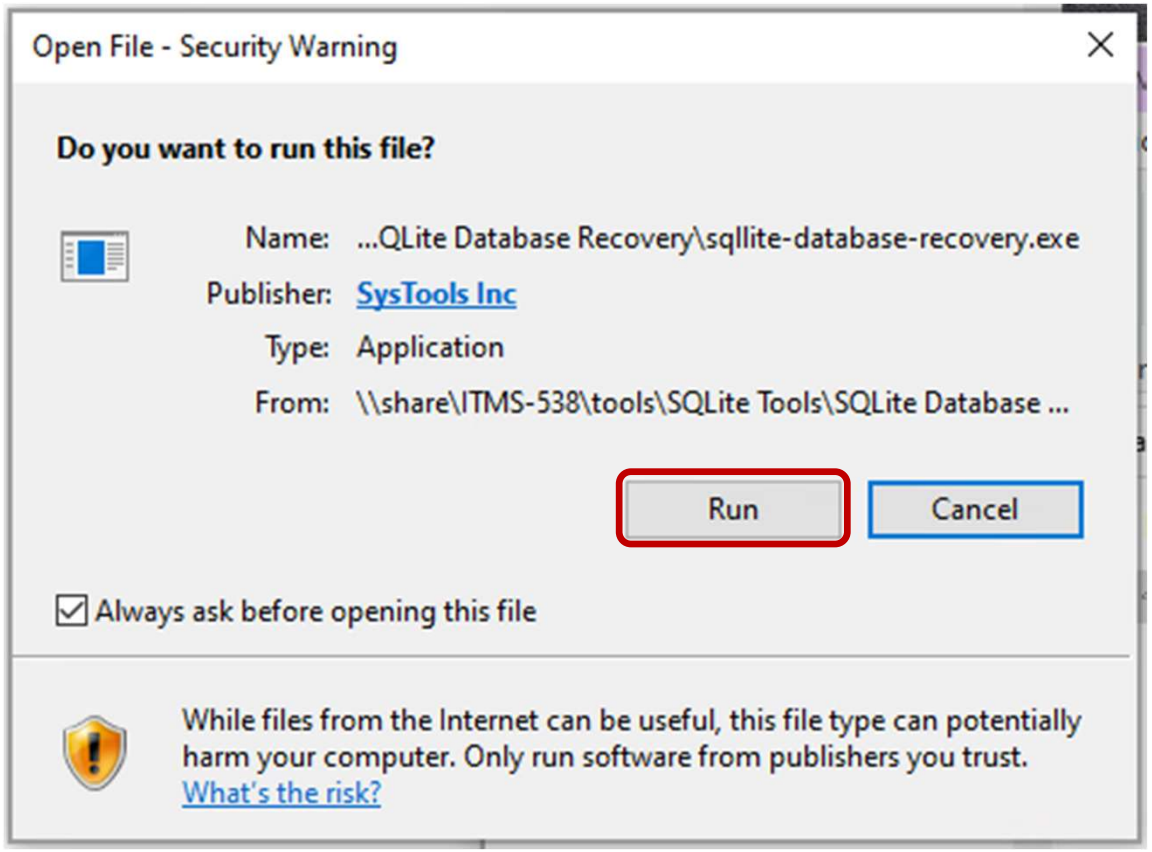
Double-click on the following file to start the installation:

`sqlite-database-recovery.exe`

Database Forensics Labs

Install SQLite Database Recovery

Click the “Run” button when the Security Warning pop-up window appears



Database Forensics Labs

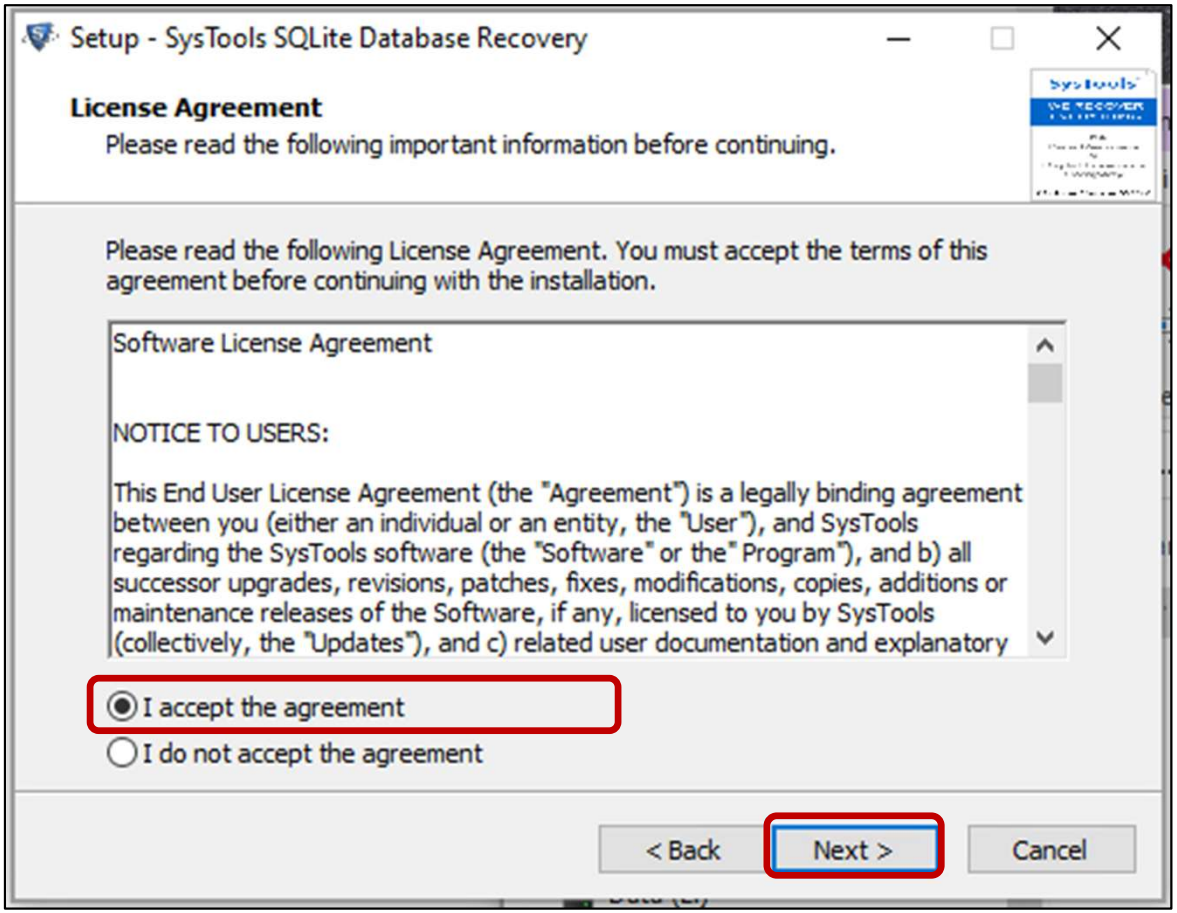
Install SQLite Database Recovery

Click “Next” button when the Welcome window appears

Database Forensics Labs

Install SQLite Database Recovery

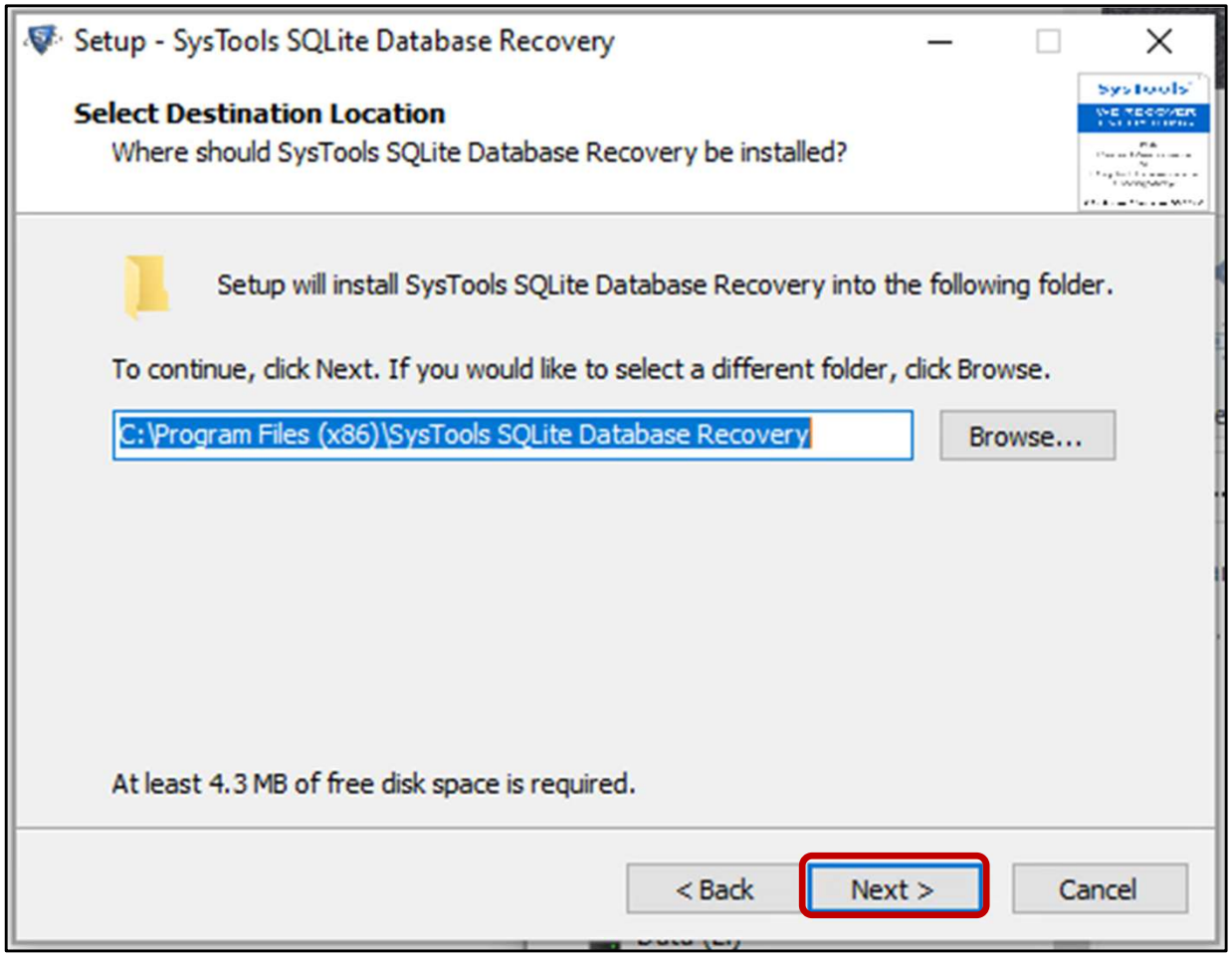
Accept the Software License Agreement, then click "Next"



Database Forensics Labs

Install SQLite Database Recovery

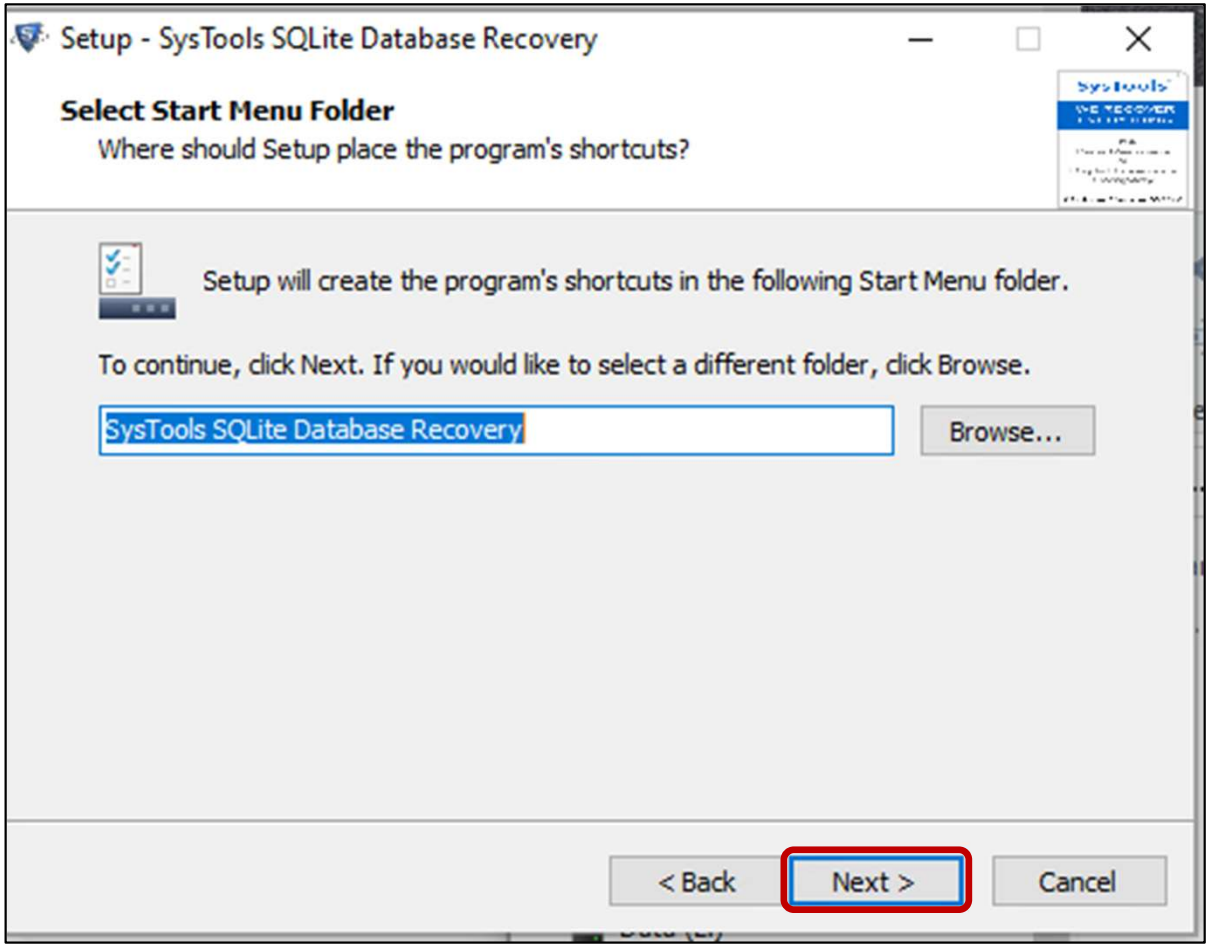
Click “Next” to select the default installation directory



Database Forensics Labs

Install SQLite Database Recovery

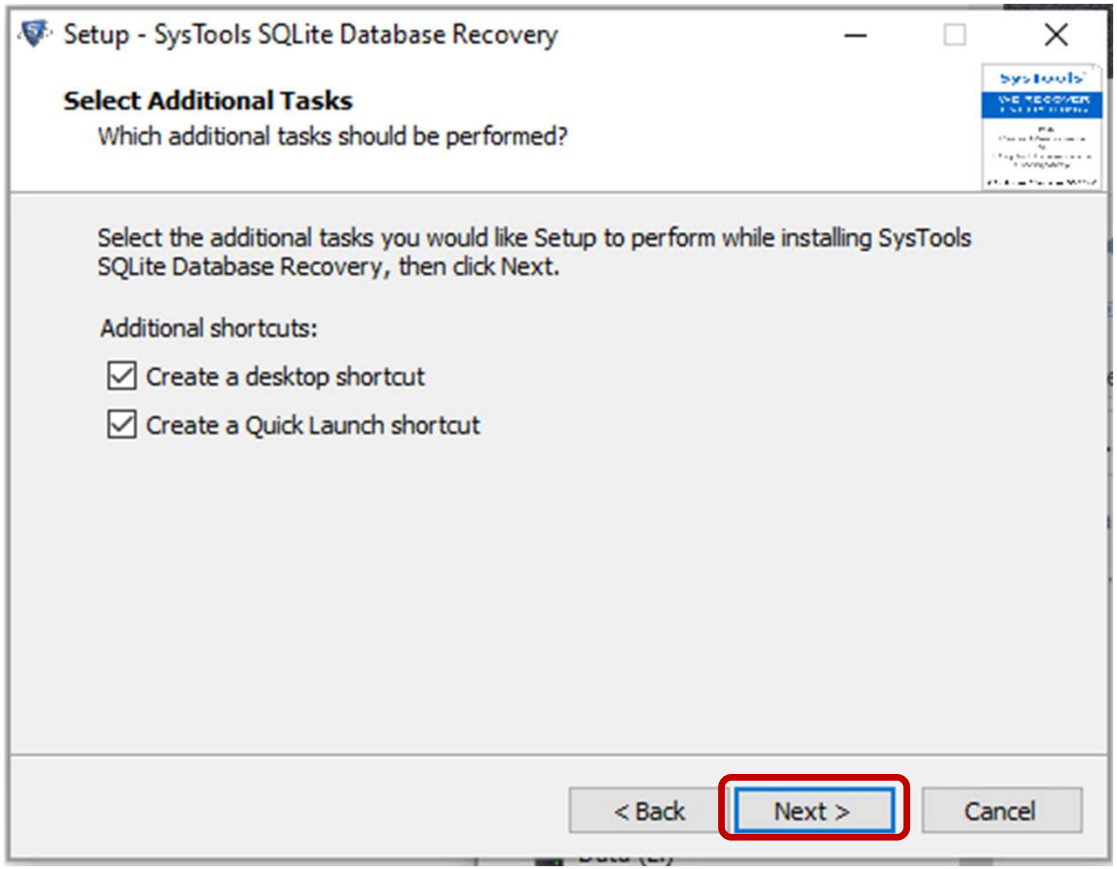
Click “Next” to select the default Start Menu folder



Database Forensics Labs

Install SQLite Database Recovery

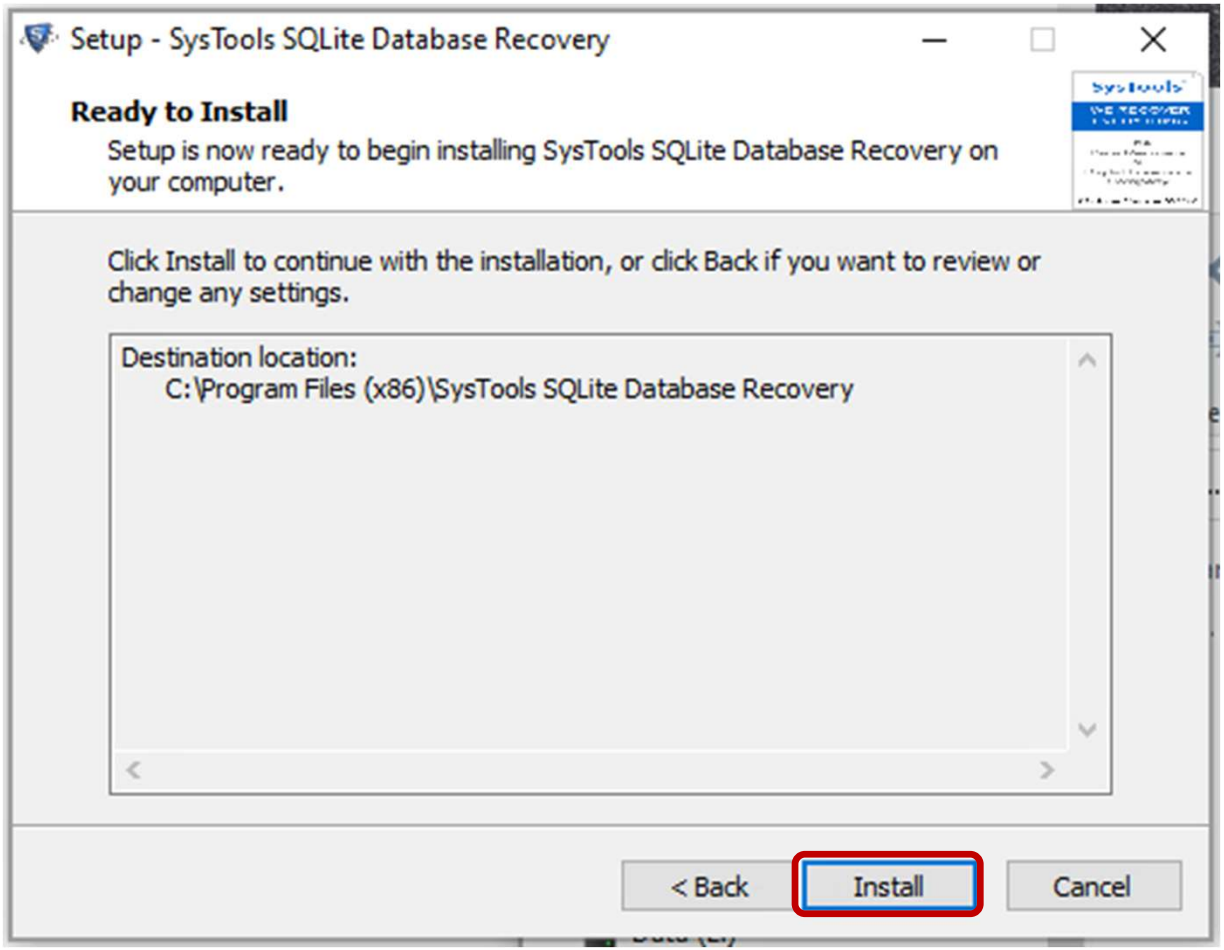
Click “Next” to accept the defaults on the Additional Tasks window



Database Forensics Labs

Install SQLite Database Recovery

Click “Install” to begin the installation



Database Forensics Labs

Install SQLite Database Recovery

You will be taken to a web site that will ask you to sign up for the SysTools newletter

Feel free to delete this tab

Database Forensics Labs

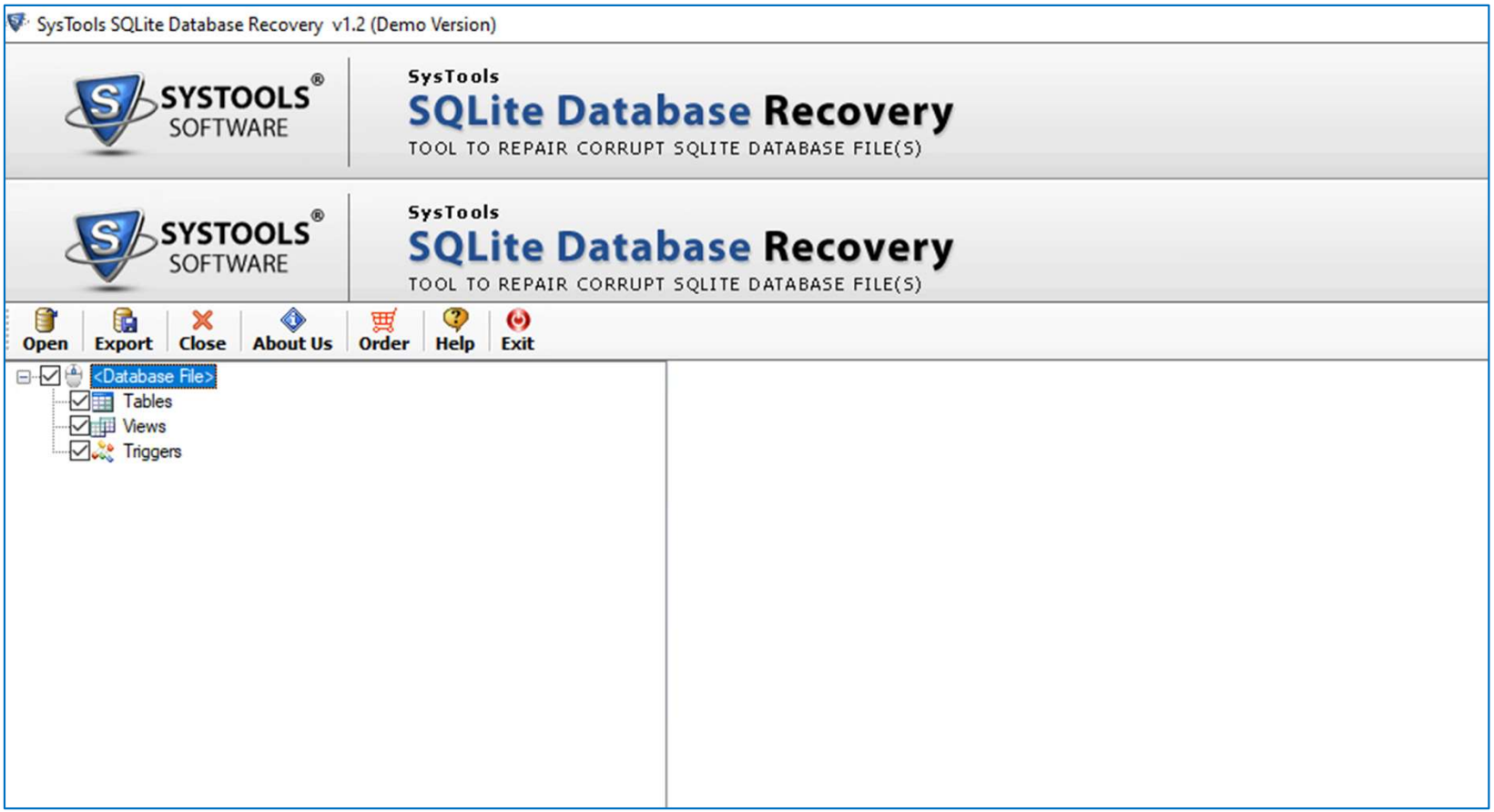
Launch SQLite Database Recovery

Click “Finish” when the installation has completed to launch a demo version of SQL Database Recovery

Database Forensics Labs

Launch SQLite Database Recovery

You should now see a window as shown below:



SQLite Database Recovery

Close 0A-05.db in other Applications

SQLite Database Recovery won't open databases that are open in other applications

Close the 0A-05.db database in

DB Browser for SQLite (Close Database)

SQLite Forensic Explorer

Select 0A-05 on the tree viewer

Click Close on the top menu bar

Click OK on the confirmation pop-up window

SQLite Database Recovery

Deleted Tables

In SQLite Database Recovery, click on Open

- *Navigate to the <Wdir>\SQLite Corpus\0A\0A-05.db file*
- *Click Open*
- *You will then get two pop-up windows*
  - A status window indicating the database scanning was successfully completed
    - *Click OK*
  - A scanning summary window
    - *Note that it found 0 tables*
    - *Click Close*

NOTE!

SQLite Database Recovery may require that you install the .NET framework before a database can be opened.

If this occurs, the instructions will guide you through the installation

SQLite Database Recovery

Deleted Tables

Navigate around on the home page to see if any deleted records or tables have been recovered

What do you discover?

SQLite Database Recovery

Summary

SQLite Database Recovery IS designed to recover forensic data

However, it doesn't recover all the forensically useful data in our databases

We have observed that

Deleted tables are not recovered (0A-05.db)

Database Forensics Labs

Forensic examination using

Autopsy

Database Forensics Labs

Autopsy

We'll next explore the database forensics capabilities associated with

Autopsy

As you may recall, Autopsy has an extensible architecture where additional capabilities can be added through *ingest modules*

We'll first start an Autopsy case file for our databases, then install two specific ingest modules that may provide additional database forensics capabilities

Parse_SQLite_Databases

Parse_SQLite_Del_Records

Autopsy
Create New Case

Open Autopsy and create a new case for our Database Forensics analysis

Call the case: DBForensicsLab

When you get to Add Data Source window, select Logical Files

On the next window

Click on the Add button

Navigate to the 0A-05.db database

Click Select

Click Next

Select all Ingest modules, then click Next

Click Finish

Autopsy
Navigate

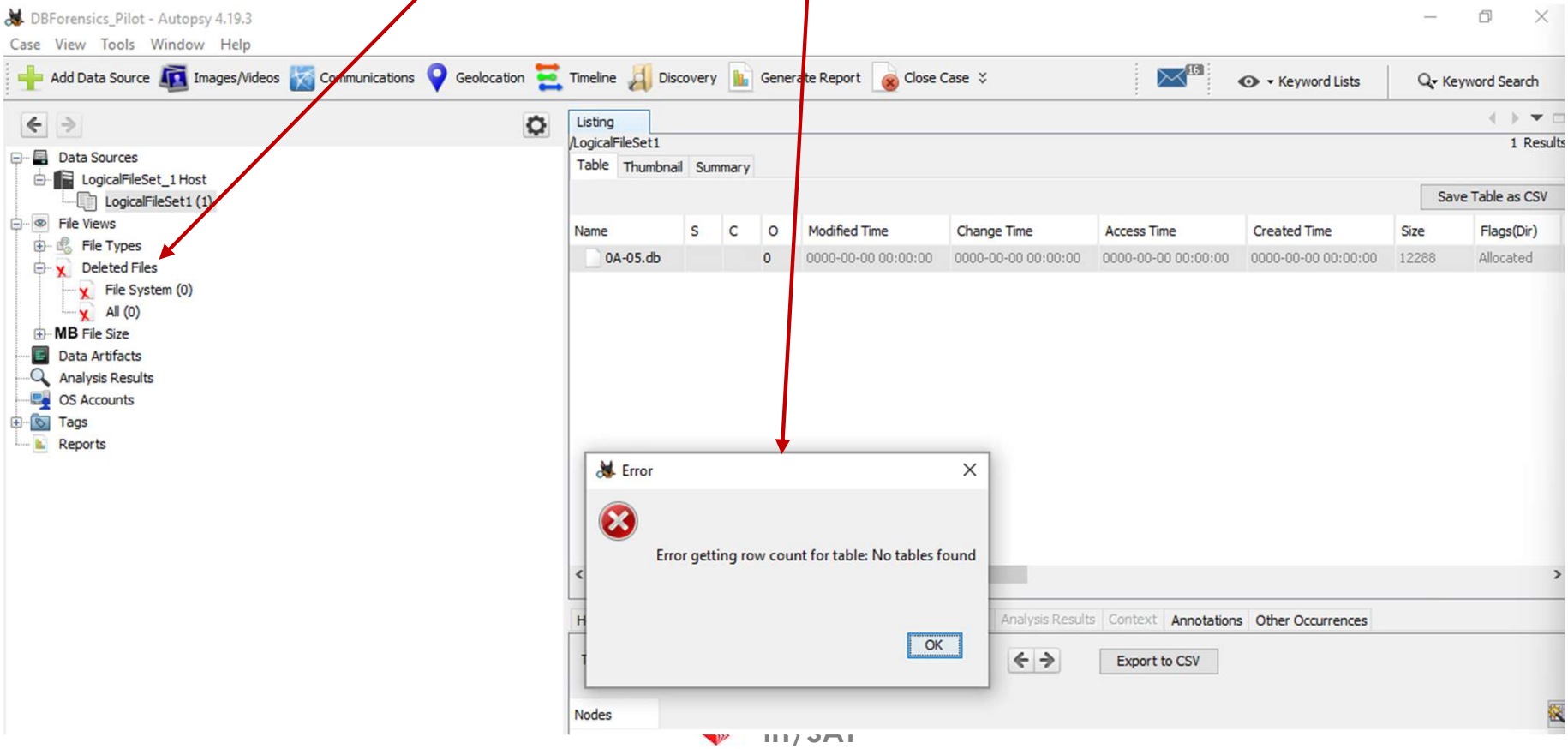
Use the Tree Viewer to

Navigate to the LogicalFileSet1 data source (our DB)

Double-click on the 0A-05.db in the Result Viewer

Note the pop-up window indicating no tables were found

Explore any deleted artifacts found (none)



Autopsy
Preliminary Summary

Autopsy IS designed to recover forensic data

However, with the default set of ingest modules, it doesn't have capabilities to perform database forensics

We will now install the following two ingest modules that will provide some of this capability

Parse_SQLite_Databases

Parse_SQLite_Del_Records

Database Forensics Labs

Install Autopsy Ingest Modules

First, close Autopsy

Click on Close Case

Exit Autopsy

Go to your *SQLiteTools/Autopsy* subdirectory in your *<Wdir>* folder

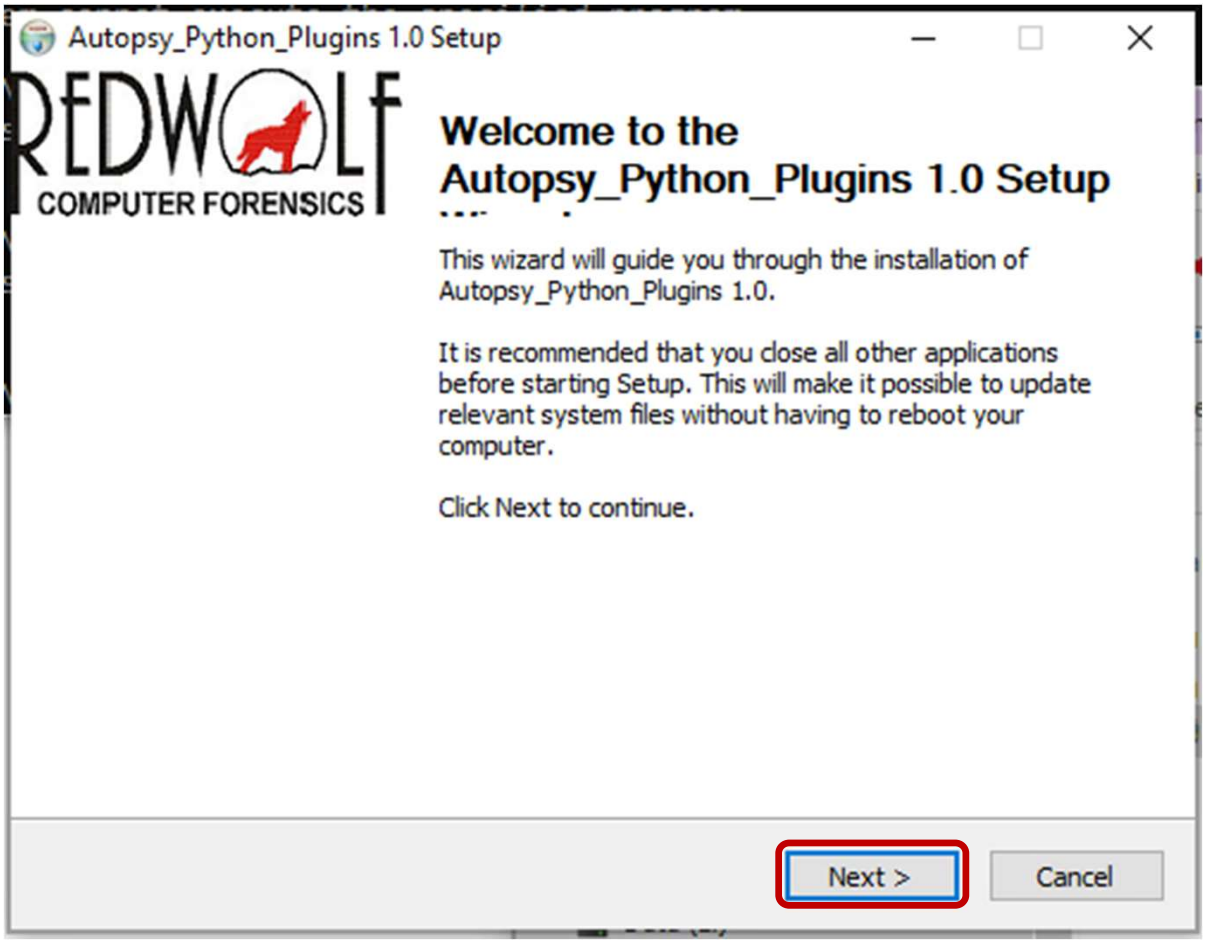
Double-click on the following file to start the ingest module installation:

`Autopsy_Python_Plugins.exe`

Database Forensics Labs

Install Autopsy Ingest Modules

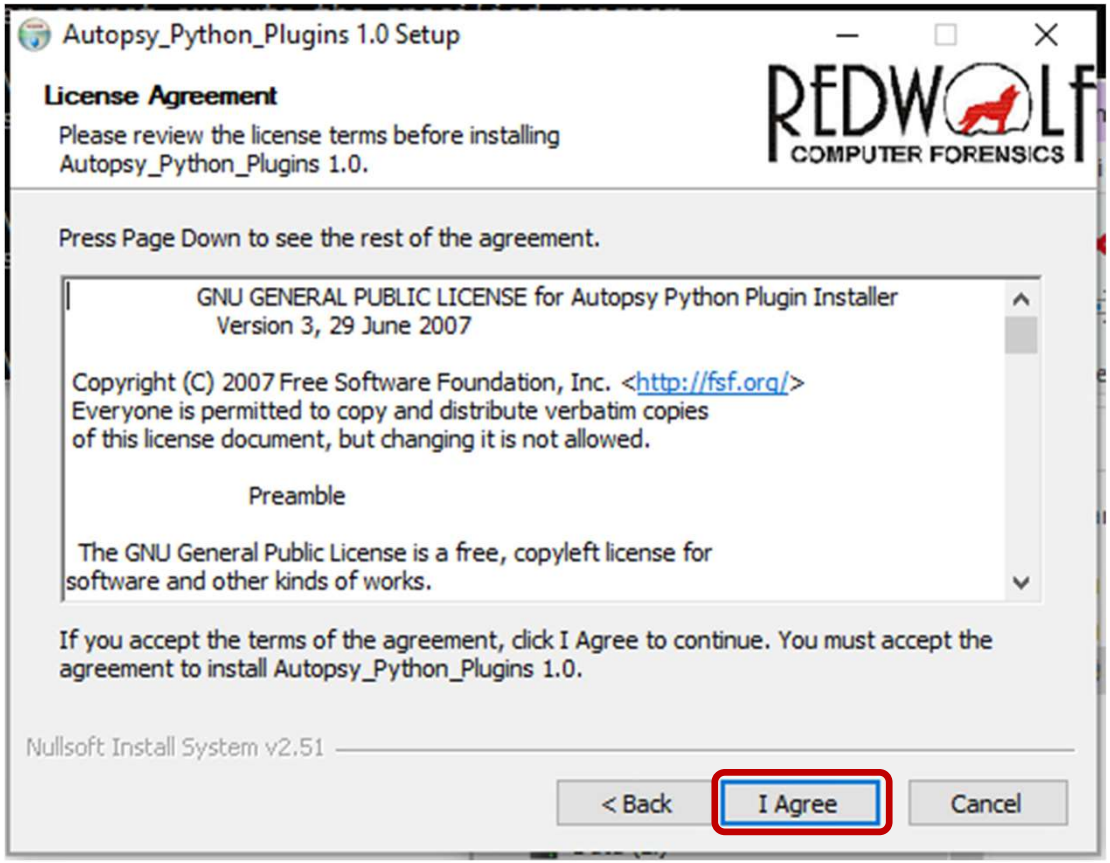
Click “Next” button when the Welcome window appears



Database Forensics Labs

Install Autopsy Ingest Modules

Click on “I Agree” to accept GNU General Public License Agreement



Database Forensics Labs

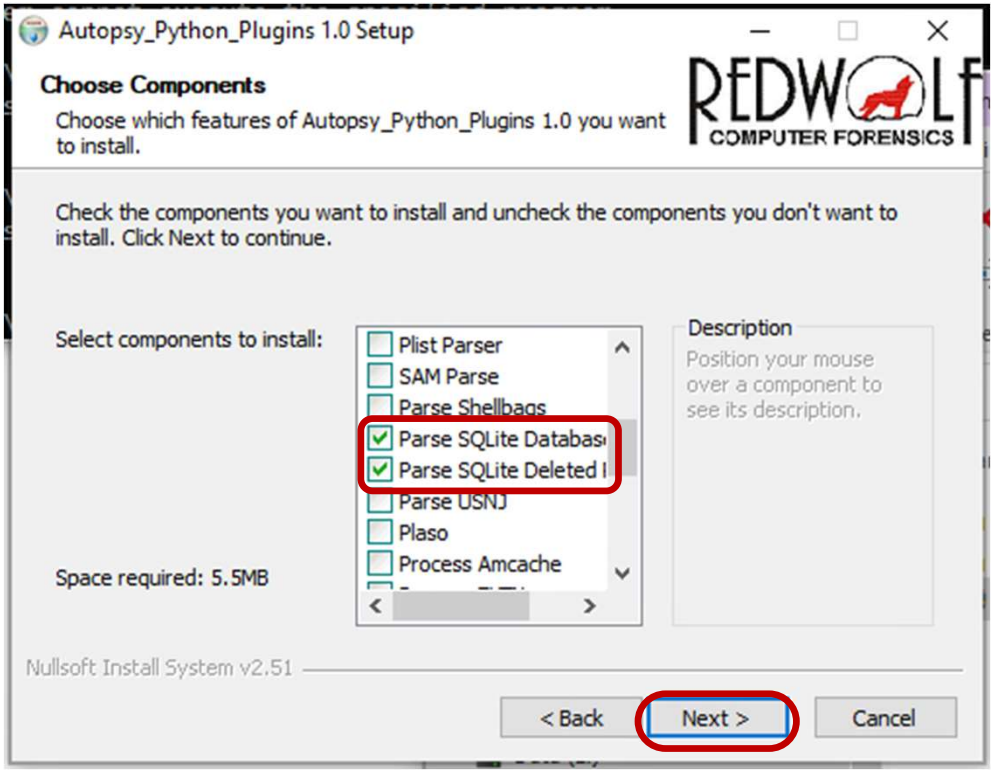
Install Autopsy Ingest Modules

In the Chose Components window, select only the following two modules

Parse SQLite Databases

Parse SQLite Deleted Records

Make sure ALL OTHER ingest modules are UNCHECKED



Database Forensics Labs

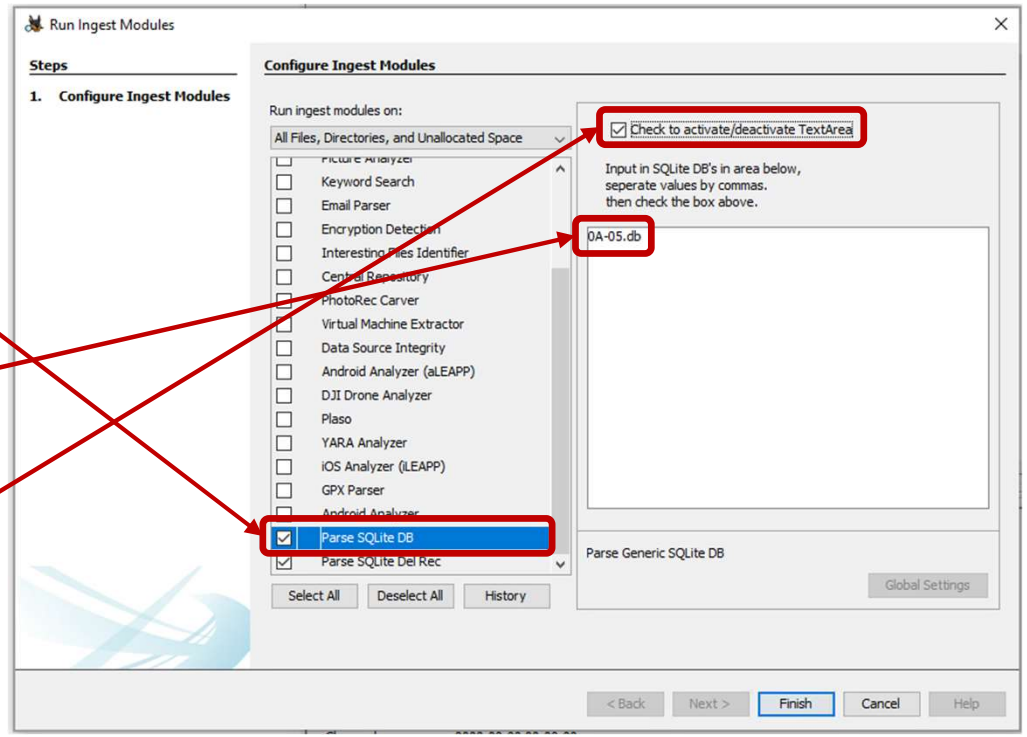
Install Autopsy Ingest Modules

For both of these ingest modules:

- *Select the ingest module by clicking on it*
- *Write the 0A-05.db database name in the text box*
- *Check the box at the top*

After both modules have been configured this way

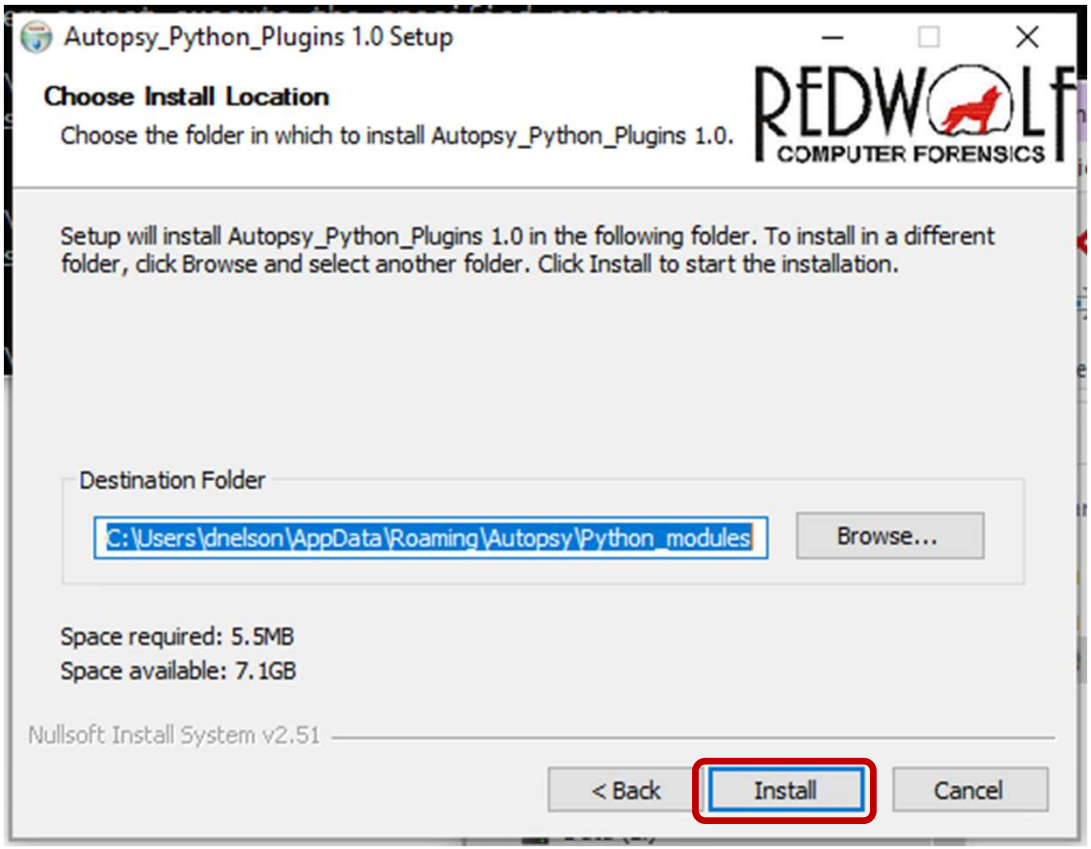
- *Click Next*



Database Forensics Labs

Install Autopsy Ingest Modules

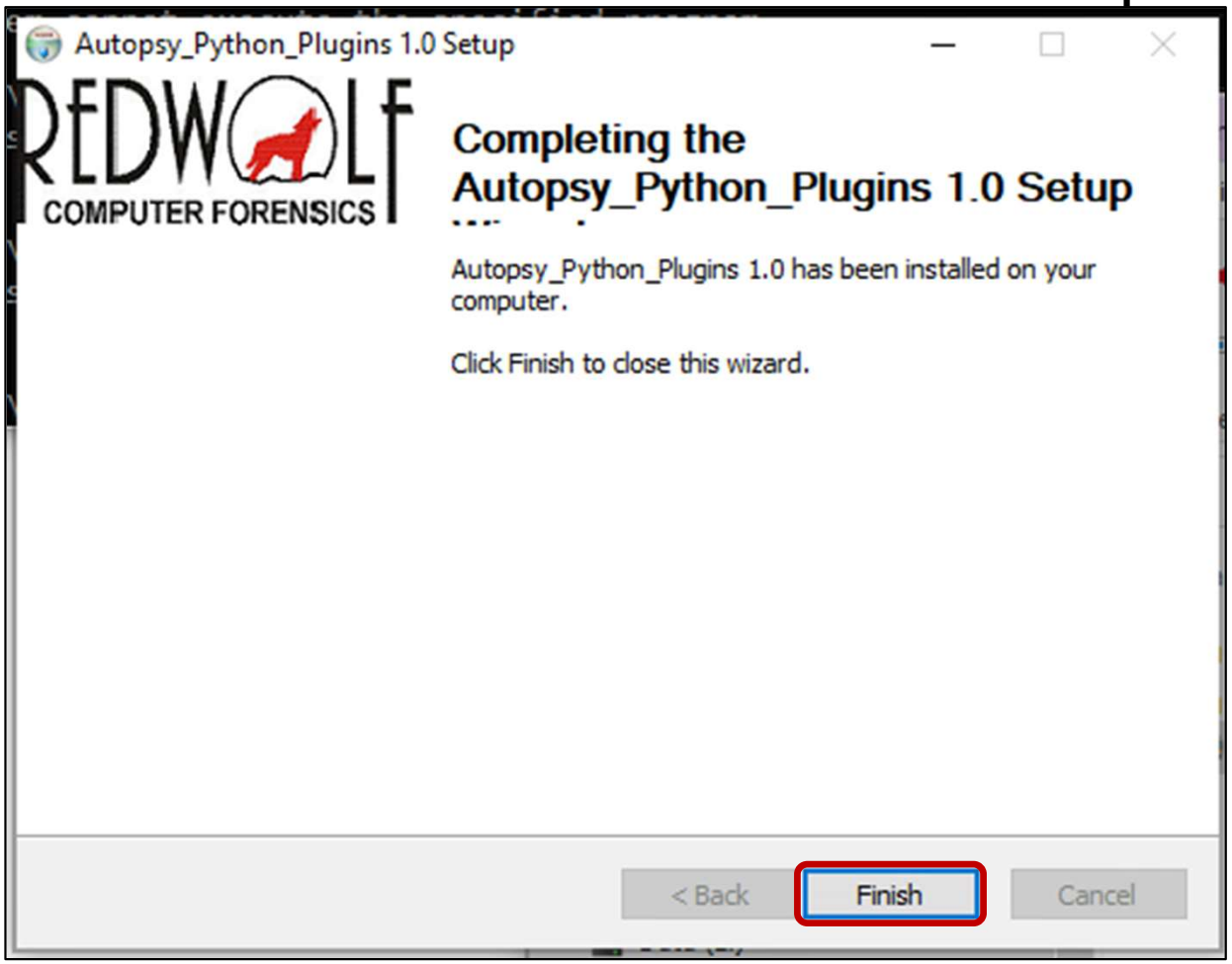
Click “Install” to select the default installation directory and begin the installation



Database Forensics Labs

Install Autopsy Ingest Modules

Click “Finish” when the installation has completed

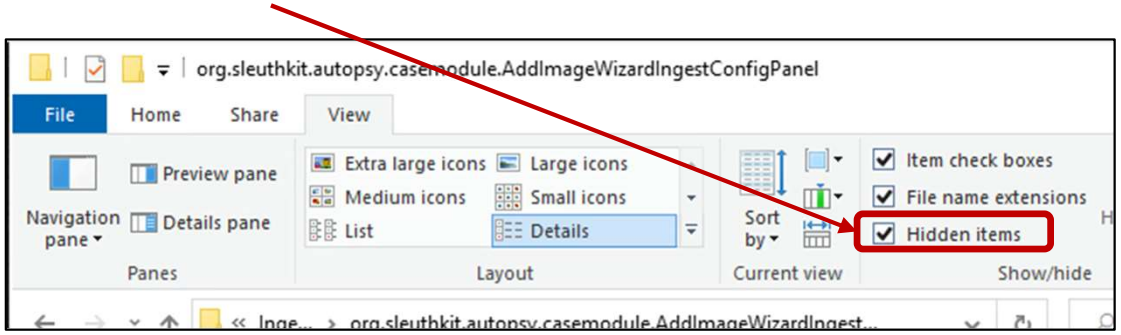


Autopsy
Cleanup Installation

Open File Explorer

Click on View menu item to open the View ribbon

Check the "Hidden items" box to show hidden files and directories



Navigate to

C:\Users\<user>\AppData\Roaming\autopsy\config\IngestModuleSettings\org.sleuthkit.autopsy.casemodule.AddImageWizardIngestConfigPanel

and

C:\Users\<user>\AppData\Roaming\autopsy\config\IngestModuleSettings\org.sleuthkit.autopsy.ingest.RunIngestModulesDialog

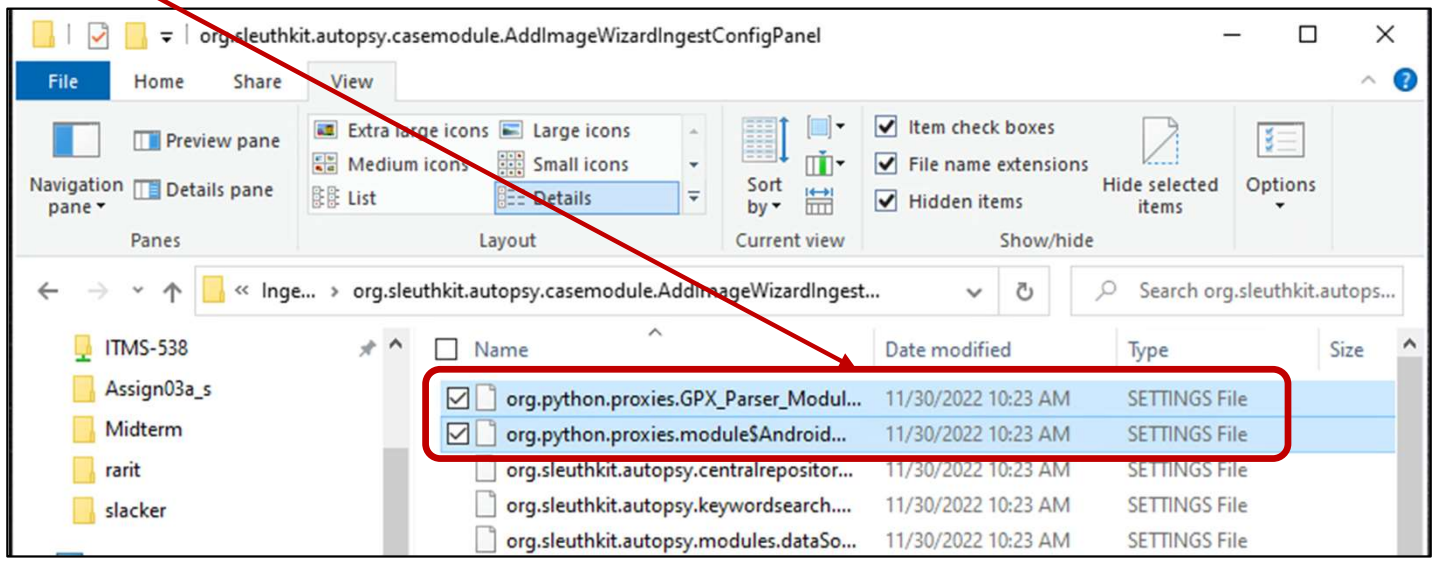
Where <user> is replaced by your userID

Autopsy
Cleanup Installation

In each of the two directories (folders)

Delete ALL the files that start with org.python.proxies

E.g.,



Autopsy
Run SQLite Ingest Modules

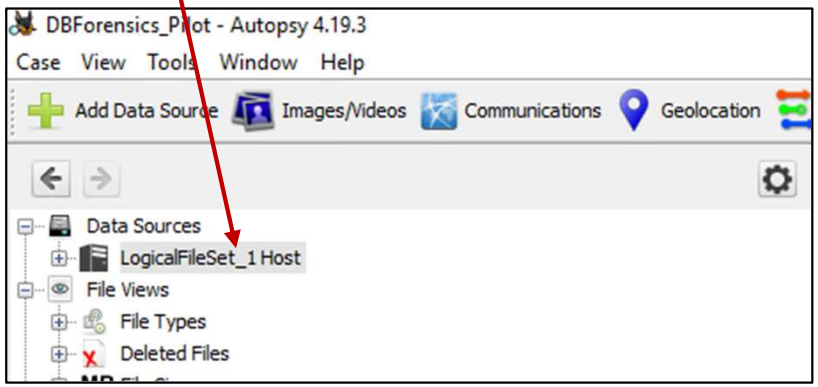
Open Autopsy

Click on Open Recent Case

Select the DBForensicsLab case from the Recent Cases list

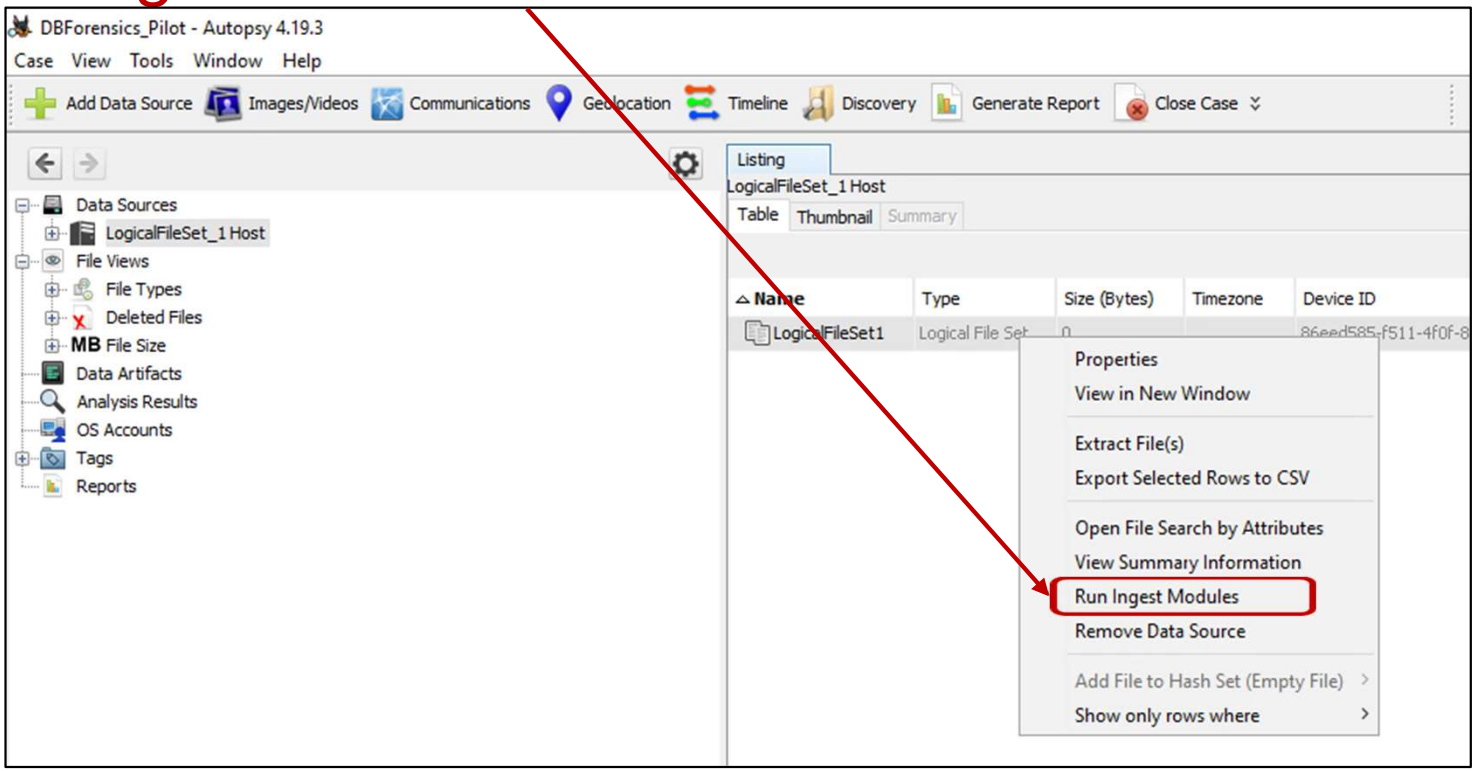
Click Open

Expand the Data Sources branch in the Tree Viewer as shown below



Autopsy
Run SQLite Ingest Modules

In the Results Viewer, right click on the Dataset (LogicalFileSet1 in the example below) and select Run Ingest Modules

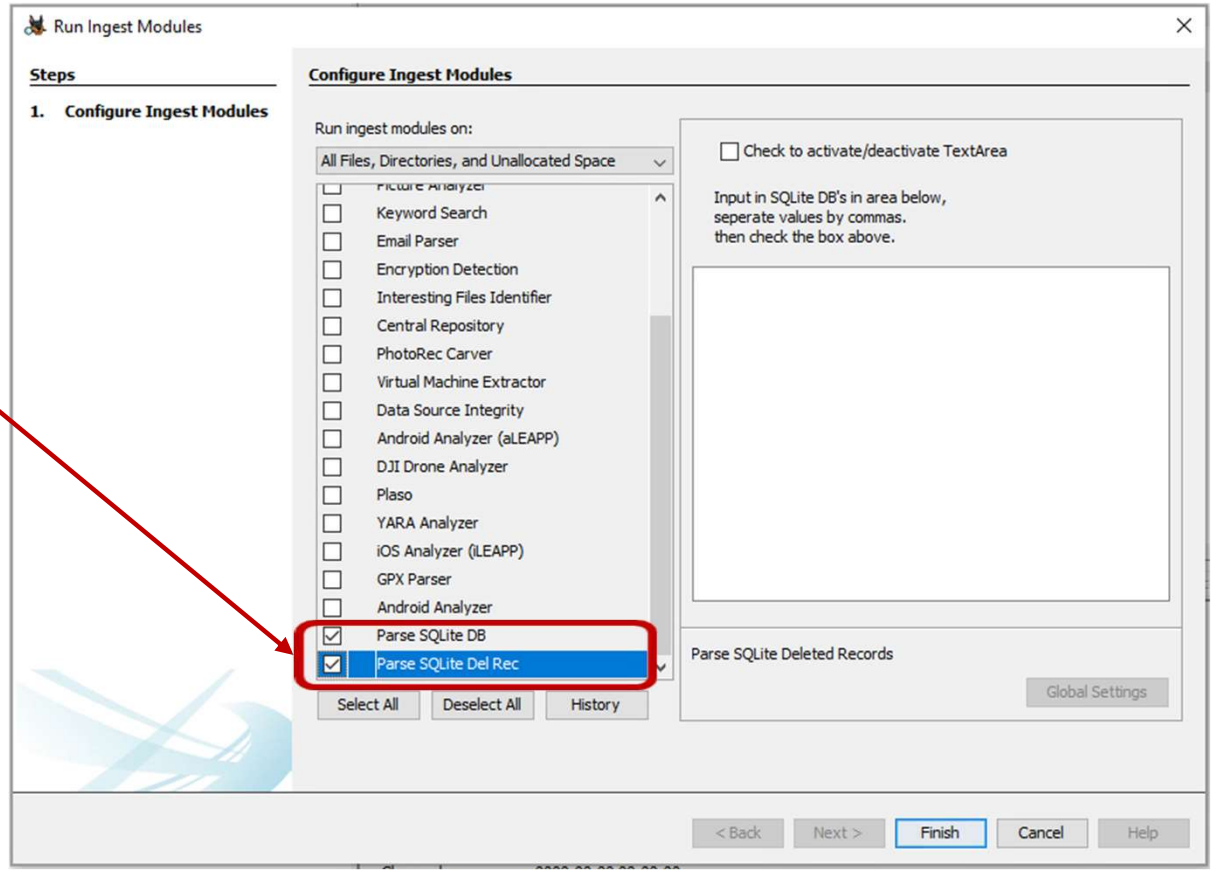


Autopsy
Run SQLite Ingest Modules

In the Run Ingest Modules window, select only the following two modules

Parse SQLite DB
Parse SQLiteDel Rec

Click Finish



Autopsy
Navigate

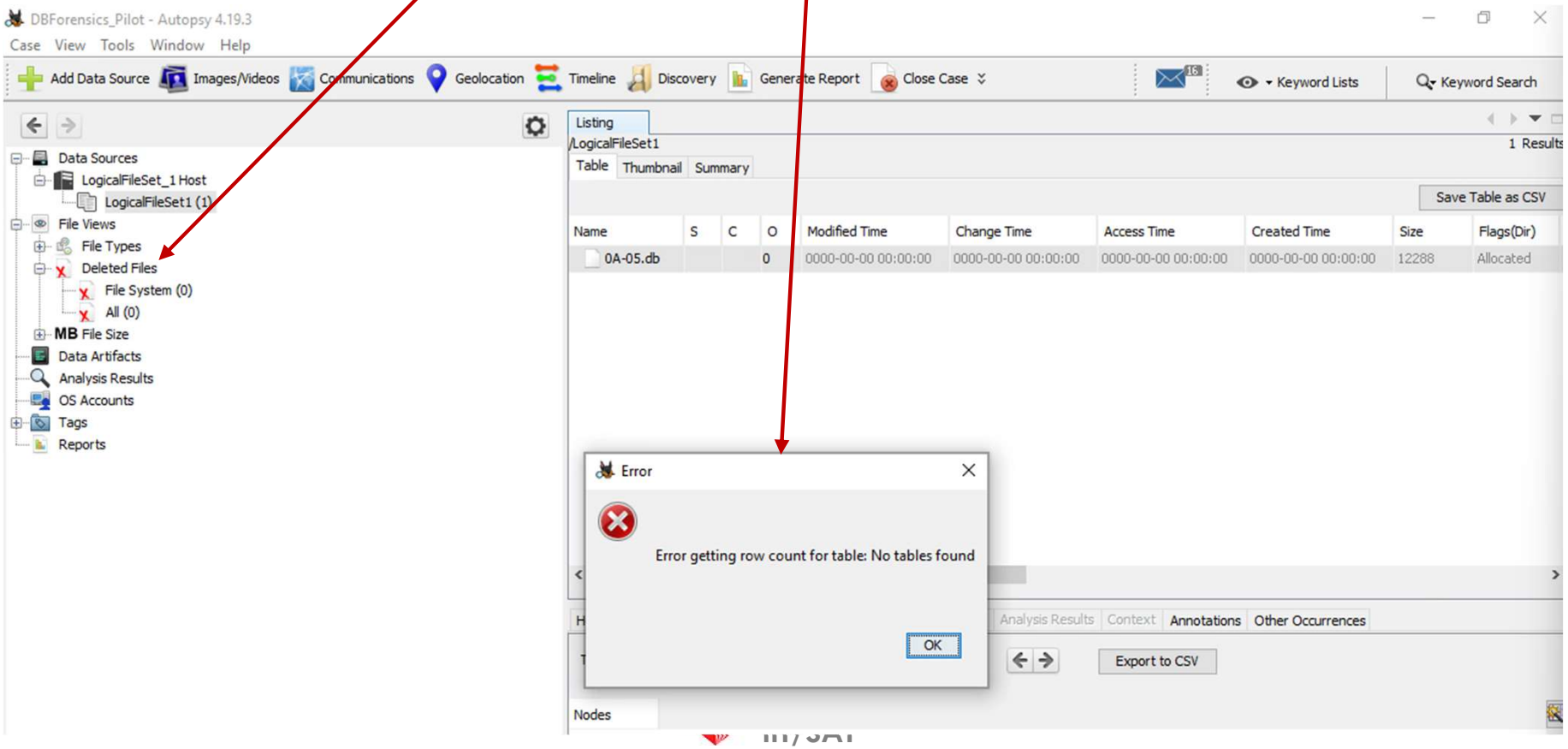
Try Once Again to Use the Tree Viewer to

Navigate to the LogicalFileSet1 data source (our DB)

Double-click on the 0A-05.db in the Result Viewer

Note the pop-up window indicating no tables were found

Explore any deleted artifacts found (none)



Autopsy
Summary

Autopsy IS designed to recover forensic data

Even specialized SQLite ingest modules weren't able to recover SQLite forensic data in the case where the database tables were deleted

Database Forensics Labs

Forensic examination using

FQLite

Database Forensics Labs

FQLite

We'll next evaluate an freeware SQLite forensics tool

FQLite

This can be found in the SQLite Tools subdirectory of your <*Wdir*> folder

This tool is a Java archive (JAR) which requires the Java Runtime Environment.

We'll first need to install JRE to run FQLite

Database Forensics Labs

Install Java Runtime Environment

In your web browser, go to the following link:

https://www.java.com/en/

Click on the Download Java button

You will be taken to a download page for Java Version 8.

Click on the Download Java button

The JRE installation file will begin downloading

When the download has completed, go to your web browser's download directory

Database Forensics Labs

Install Java Runtime Environment

The download file should have a name similar to:

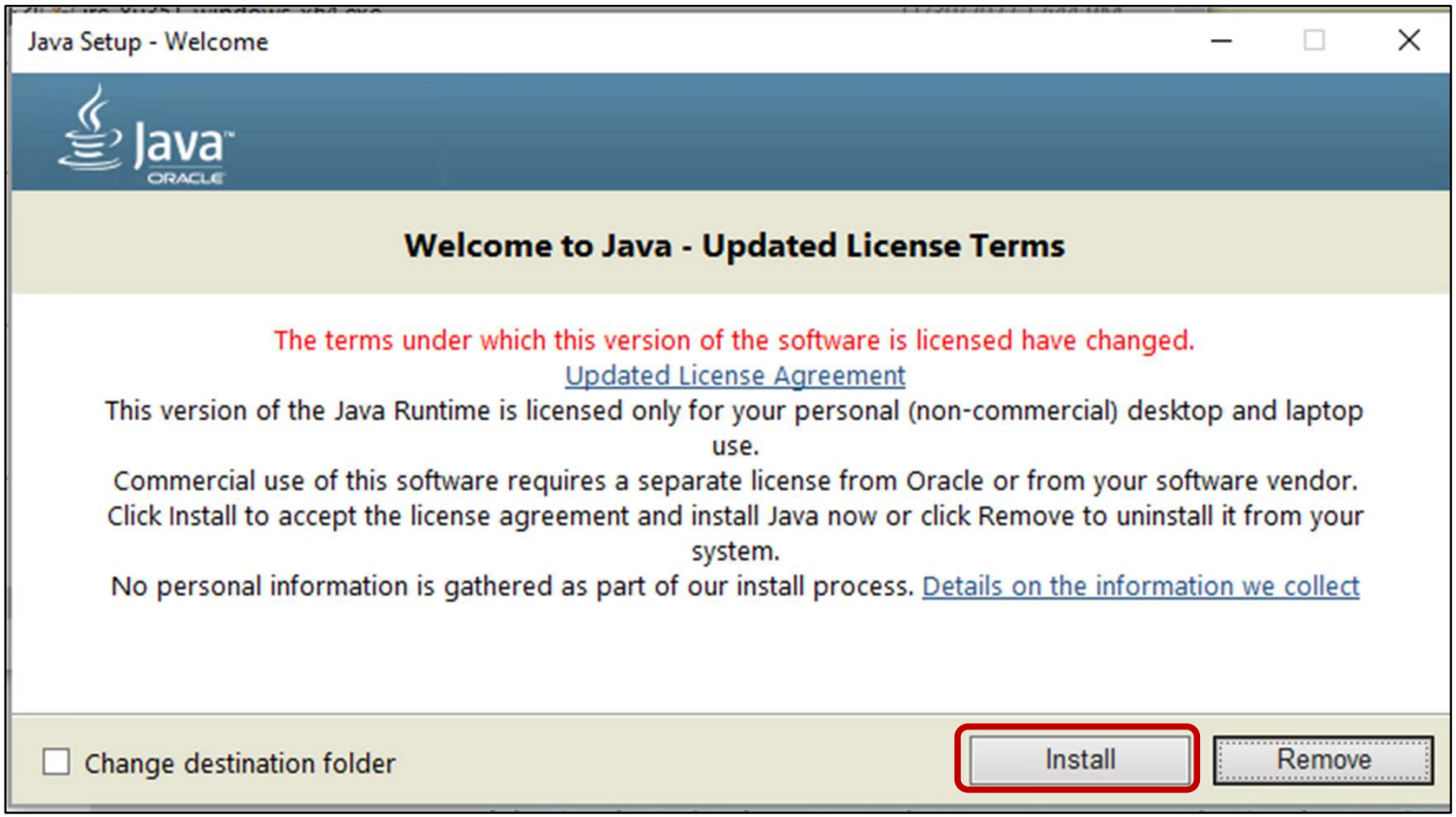
jre-8u351-windows-x64.exe

Double-click on this file from your File Explorer to begin the installation

Database Forensics Labs

Install JRE

Click “Install” when the Welcome window appears



Database Forensics Labs

Install JRE

Click “Close” the window appears that indicates you have successfully installed Java

Database Forensics Labs

Launch FQLite

Go to your *SQLiteTools/FQLite* subdirectory in your *<Wdir>* folder

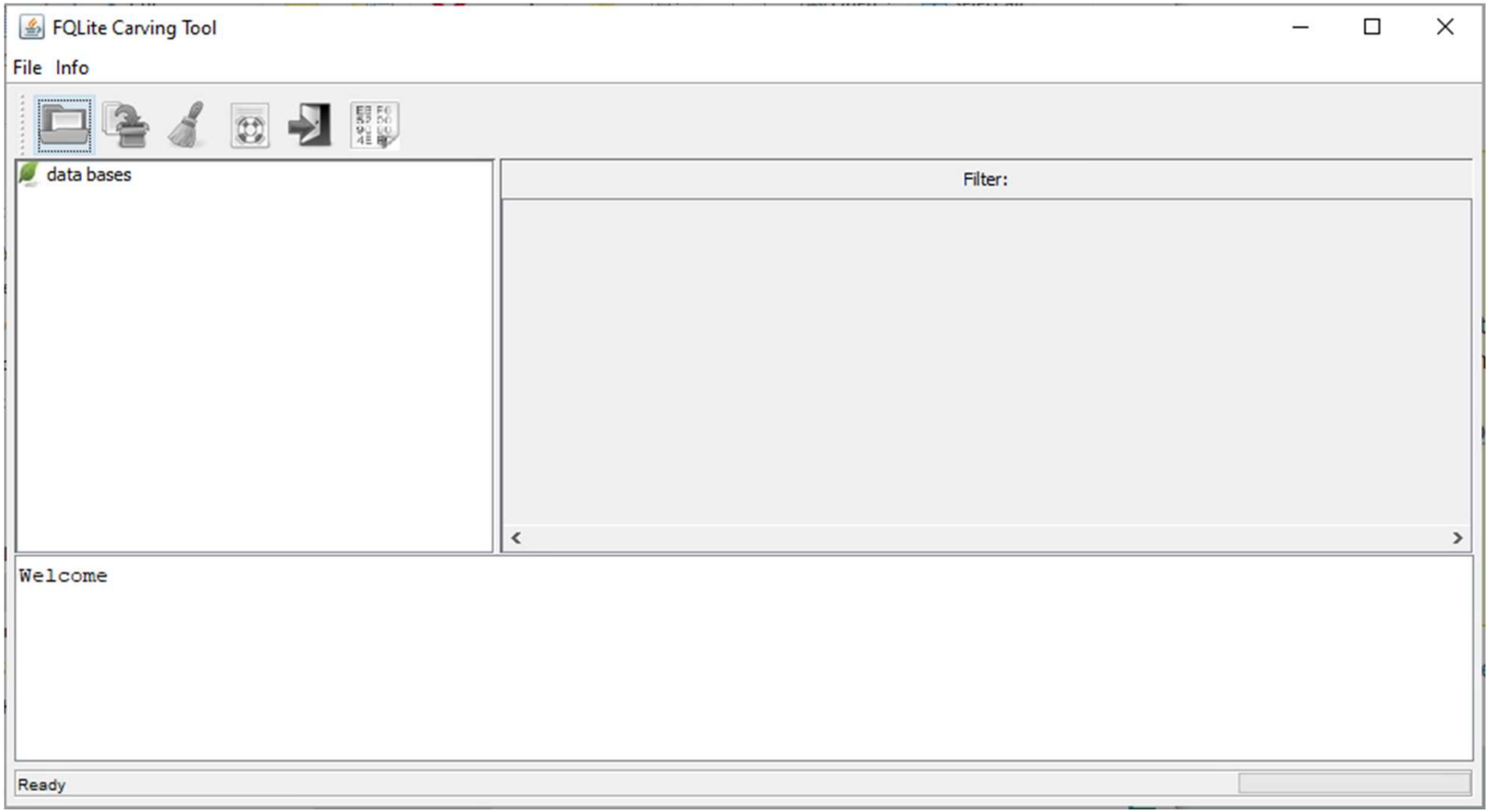
Double-click on the following file to launch FQLite:

`fqlite-v1.5.8.jar`

Database Forensics Labs

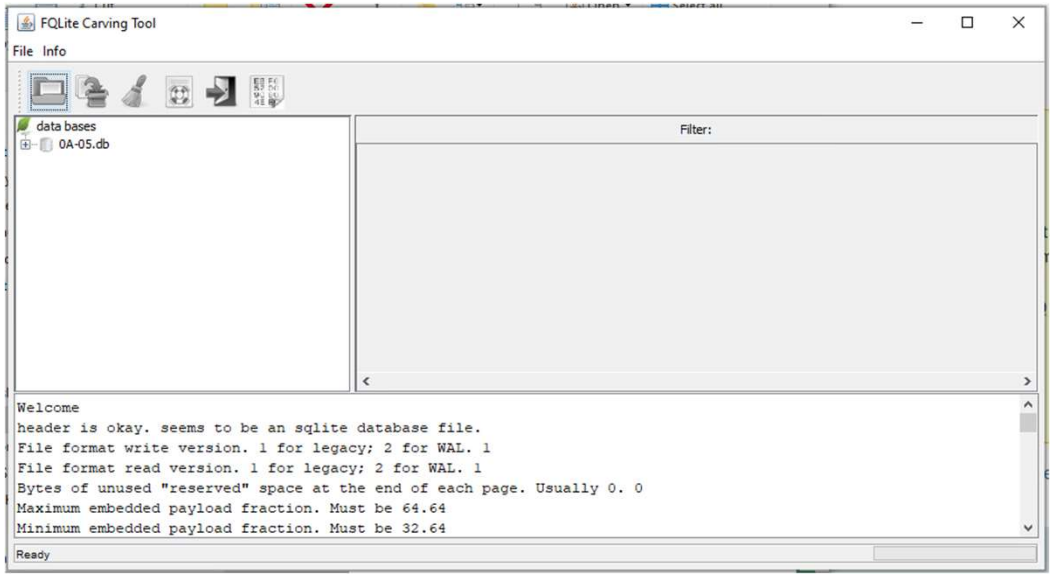
Launch FQLite

You should now see a window as shown below:



FQLite
Deleted Tables

- In FQLite, select the File > Open Database… menu item
  - *Navigate to the <Wdir>\SQLite Corpus\0A\0A-05.db file*
  - *Click Open*
- You will then see a series of messages scrolling in the bottom portion of the window
  - *There should be no errors*



FQLite
Deleted Tables

In the Tree Viewer on the left, expand 0A-05.db

Note

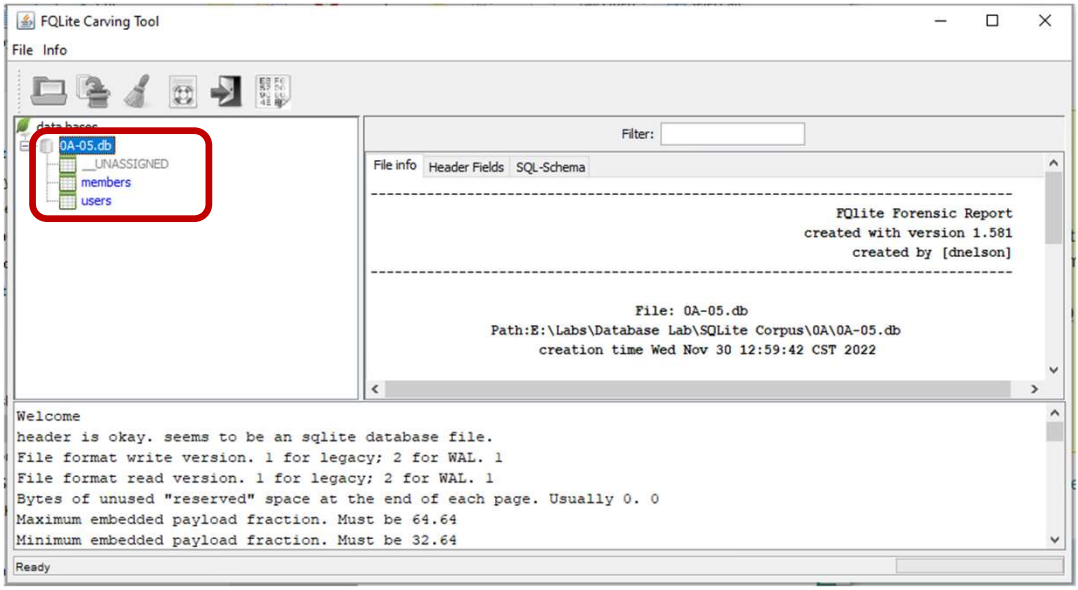
Clicking on the database itself prints out a short forensic report

Three tables are listed

This includes the two deleted tables!

members

users



FQLite

Recovering Deleted Tables

Now click on each of the deleted tables in the tree viewer and observe the records in the database viewer on the upper right panel

Note

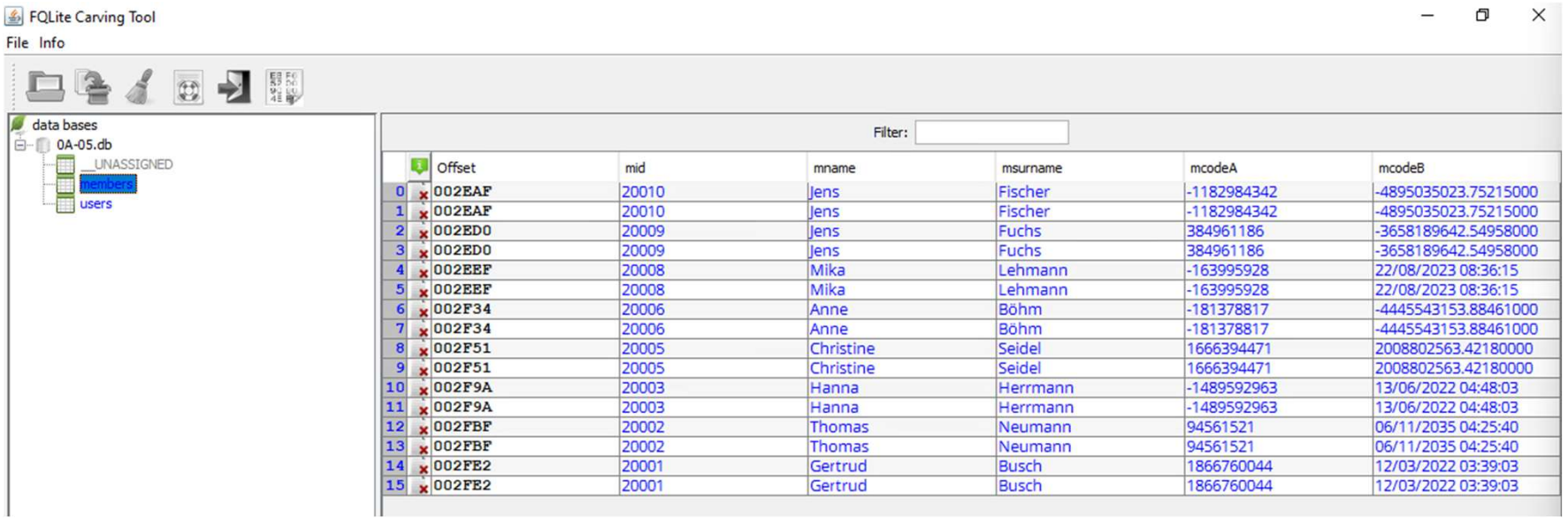
The deleted records have been recovered

A red "x" is placed in the information column to indicate the record was deleted

FQLite

Recovering Deleted Tables

members table:



	Offset	mid	mname	msurname	mcodeA	mcodeB
0	002EAF	20010	Jens	Fischer	-1182984342	-4895035023.75215000
1	002EAF	20010	Jens	Fischer	-1182984342	-4895035023.75215000
2	002ED0	20009	Jens	Fuchs	384961186	-3658189642.54958000
3	002ED0	20009	Jens	Fuchs	384961186	-3658189642.54958000
4	002EEF	20008	Mika	Lehmann	-163995928	22/08/2023 08:36:15
5	002EEF	20008	Mika	Lehmann	-163995928	22/08/2023 08:36:15
6	002F34	20006	Anne	Böhm	-181378817	-4445543153.88461000
7	002F34	20006	Anne	Böhm	-181378817	-4445543153.88461000
8	002F51	20005	Christine	Seidel	1666394471	2008802563.42180000
9	002F51	20005	Christine	Seidel	1666394471	2008802563.42180000
10	002F9A	20003	Hanna	Herrmann	-1489592963	13/06/2022 04:48:03
11	002F9A	20003	Hanna	Herrmann	-1489592963	13/06/2022 04:48:03
12	002FBF	20002	Thomas	Neumann	94561521	06/11/2035 04:25:40
13	002FBF	20002	Thomas	Neumann	94561521	06/11/2035 04:25:40
14	002FE2	20001	Gertrud	Busch	1866760044	12/03/2022 03:39:03
15	002FE2	20001	Gertrud	Busch	1866760044	12/03/2022 03:39:03

FQLite

Recovering Deleted Tables and Records

Compare the tables and records found against what was originally put into the database (but later deleted)

See next slide

SQL File that Created DB

```
CREATE TABLE users (
        'id' INT UNSIGNED NOT NULL,
        'name' TEXT NOT NULL,
        'surname' TEXT NULL,
        'codeA' INT NULL,
        'codeB' FLOAT NULL
);

CREATE TABLE members (
        'mid' INT UNSIGNED NOT NULL,
        'mname' TEXT NOT NULL,
        'msurname' TEXT NULL,
        'mcodeA' INT NULL,
        'mcodeB' FLOAT NULL
);

INSERT INTO users
        (id, name, surname, codeA, codeB)
        VALUES
        (50001, 'Maja', 'Lang', -979099654, 694130400.98461),
        (50002, 'Liam', 'Franke', -1026737640, 613028297.95192),
        (50003, 'Leni', 'Groß', 1643371695, -3428082023.01528),
        (50004, 'Elisabeth', 'Mayer', 289485196, 107790182.86047),
        (50005, 'Claudia', 'Brandt', 682453881, 4895471814.03985),
        (50006, 'Kurt', 'Müller', 277430878, -3213573693.39726),
        (50007, 'Elias', 'Sauer', 715668940, -2734937992.75483),
        (50008, 'Else', 'Vogel', -364818672, 1959408562.14585),
        (50009, 'Tobias', 'Brandt', 36208673, -2418977668.23833),
        (50010, 'Anna', 'Berger', -400626611, 3459240369.71258);

INSERT INTO members
        (mid, mname, msurname, mcodeA, mcodeB)
        VALUES
        (20001, 'Gertrud', 'Busch', 1866760044, 668770743.37973),
        (20002, 'Thomas', 'Neumann', 94561521, 1099693540.53014),
        (20003, 'Hanna', 'Herrmann', -1489592963, 676806483.26725),
        (20004, 'Katharina', 'Frank', -942844261, 2099325608.97401),
        (20005, 'Christine', 'Seidel', 1666394471, 2008802563.42180),
        (20006, 'Anne', 'Böhm', -181378817, -4445543153.88461),
        (20007, 'Lukas', 'Fischer', -819379123, -2827242011.12392),
        (20008, 'Mika', 'Lehmann', -163995928, 714404175.85634),
        (20009, 'Jens', 'Fuchs', 384961186, -3658189642.54958),
        (20010, 'Jens', 'Fischer', -1182984342, -4895035023.75215);

PRAGMA secure_delete=0;
PRAGMA secure_delete;

DELETE FROM users where id == 50007;
DELETE FROM users where id == 50010;
DELETE FROM members where mid == 20001;
DELETE FROM members where mid == 20006;
DELETE FROM users where id == 50005;
DELETE FROM members where mid == 20004;
DELETE FROM members where mid == 20002;
DELETE FROM users where id == 50003;
DELETE FROM members where mid == 20007;
DELETE FROM members where mid == 20003;
DELETE FROM users where id == 50001;
DELETE FROM users where id == 50006;
DELETE FROM members where mid == 20010;
DELETE FROM members where mid == 20005;
DELETE FROM users where id == 50004;
DELETE FROM users where id == 50002;
DELETE FROM members where mid == 20009;
DELETE FROM members where mid == 20008;
DELETE FROM users where id == 50009;
DELETE FROM users where id == 50008;

DROP TABLE users;
DROP TABLE members;
```

FQLite
Summary

FQLite Database Recovery IS designed to recover forensic data

We have observed that FQLite outperformed all the other database forensic tools in terms of recovering deleted tables and records

All deleted tables were recovered (0A-05.db)

Most of the deleted records were recovered

Your assignment

Assign06c_s

Complete the following table by opening the specified databases using the tools we've used today

- *Look at the .sql files (near the end) for each database to see what was deleted*
- *Enter "Yes", "No", or "Partial" in each empty cell*
- *You may have to close databases or even applications (e.g., FQLite) to open them in other applications*
- *You may have to repeat the cleanup steps in slides 88-89 every time you want to successfully re-run the new ingest modules on a new database*

Also submit a one-page document comparing the database forensic capabilities of these tools

Category	Database	Database Forensic Tools			
		SQLite Forensic Explorer	*SQLite Database Recovery*	*Autopsy*	*FQLite*
Deleted Tables	0A-05.db	No	No	No	Yes
Overwritten Tables	0B-02.db				
Deleted Records	0C-10.db				
Overwritten records	0D-08.db				
Deleted overflow pages	0E-02.db				
Unallocated areas - freeblock	17-13-antifor.db				
Unallocated areas - freelist	18-01-antifor.db				