The use of

by SQLite

BerkeleyDB
LevelDB
RocksDB
etc ...

- Embedded
- Key/Value

SQLite

- Embedded
- SQL

MySQL
PostgreSQL
Microsoft SQL Server
ORACLE
VOLTDB

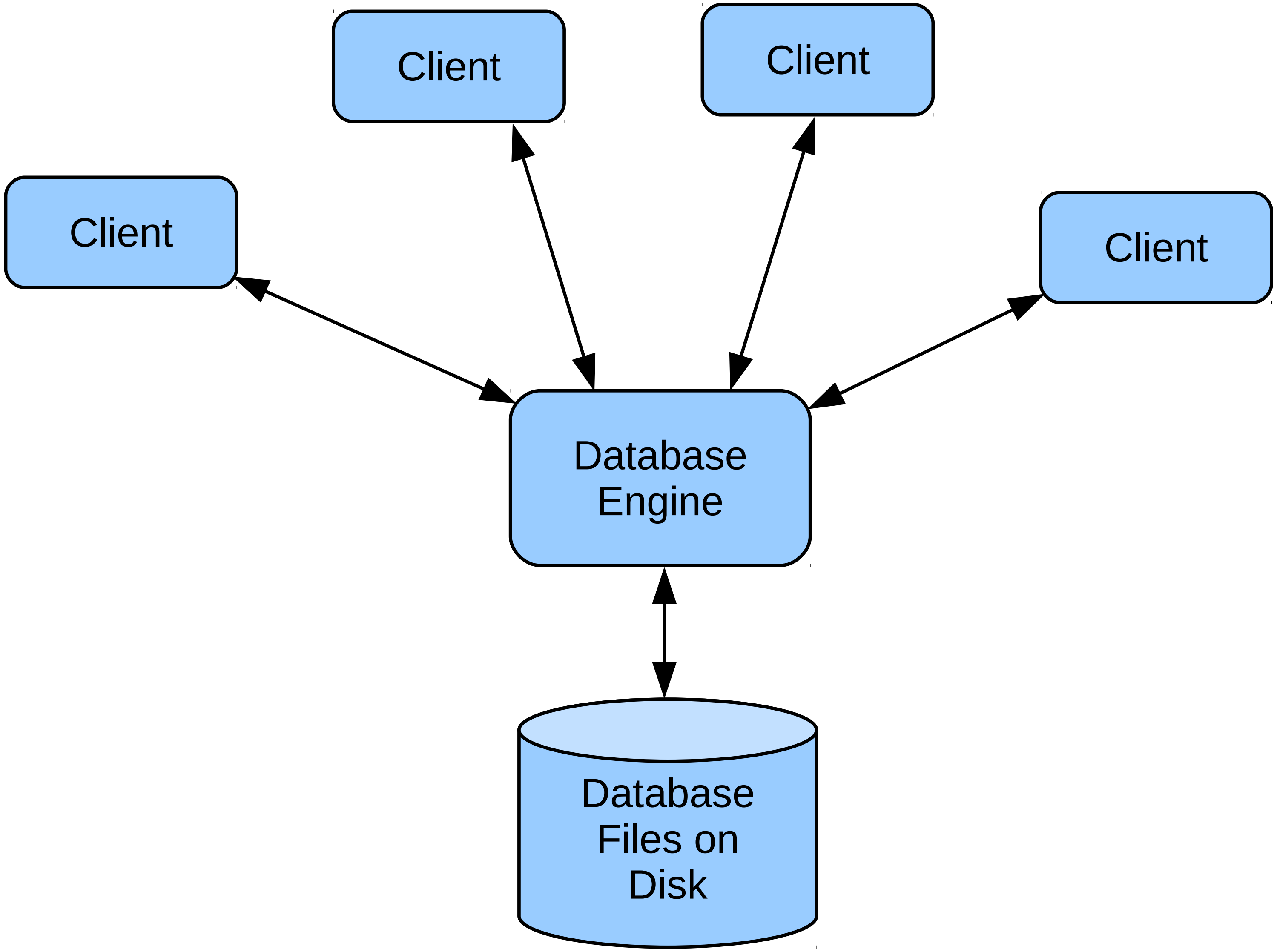
- Client/Server
- SQL

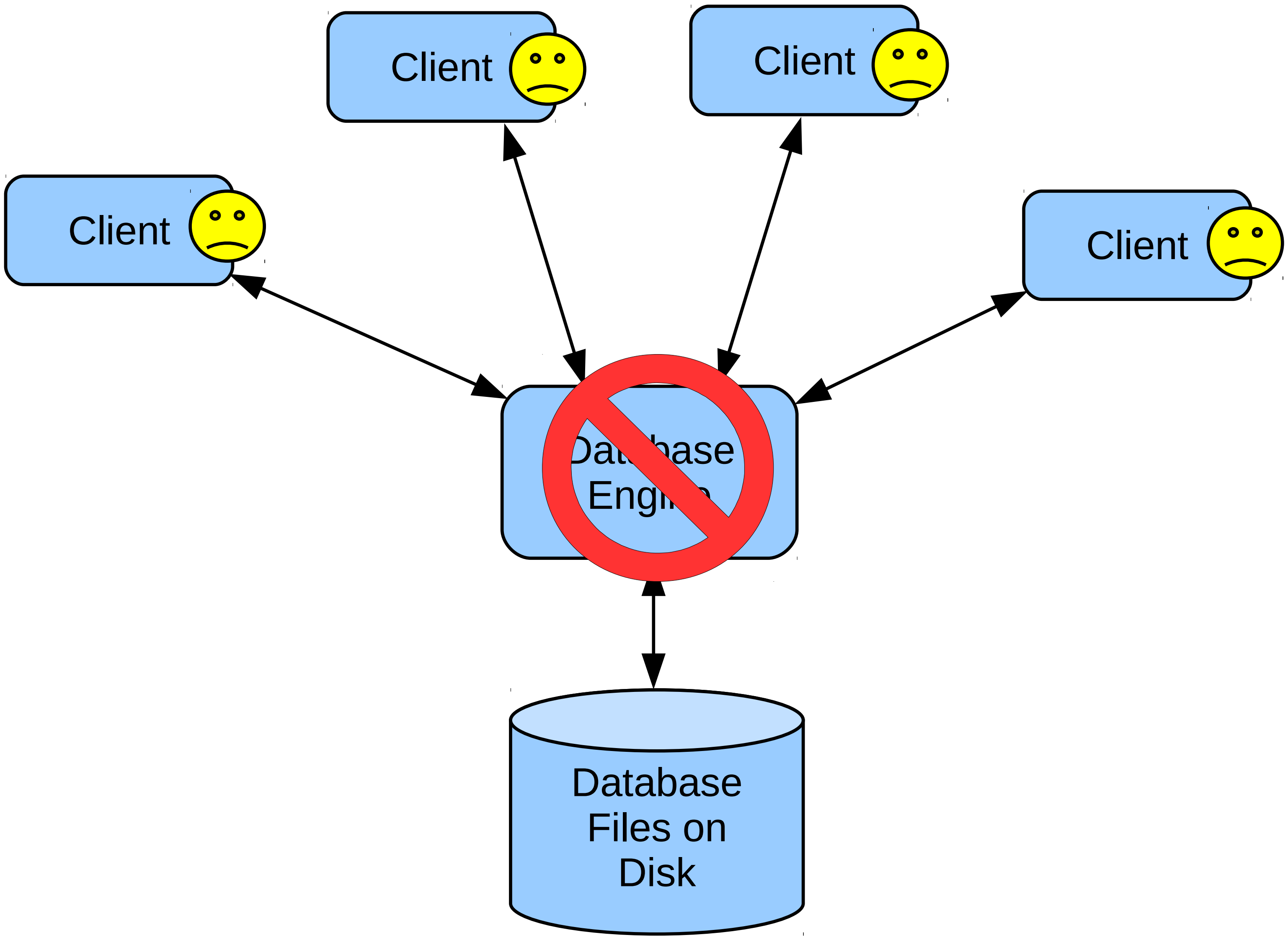
Unstoppable Ideas Behind SQL

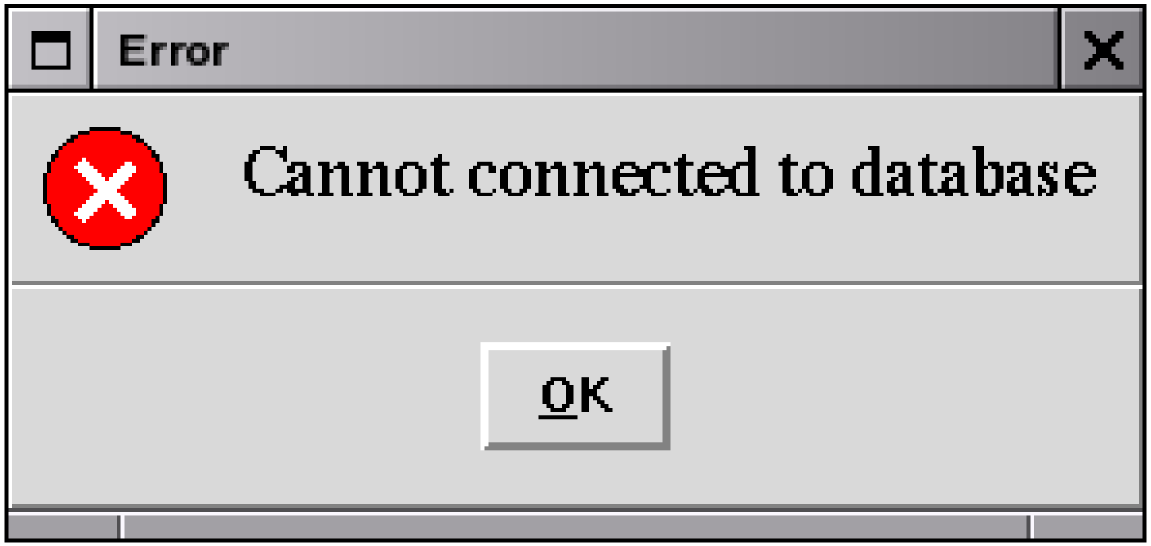
- Data Abstraction
  - *“Representation is the essence of computer programming.”*
- Declarative Language
  - *Push the semantics of the query down into the storage engine and let it figure out what to do.*
- Transactions

```sql
INSERT INTO users VALUES('alex','Alexander Fogg',29,3341);

DELETE FROM users WHERE uid='alex';

UPDATE users SET officeId=4217 WHERE uid='alex';
```

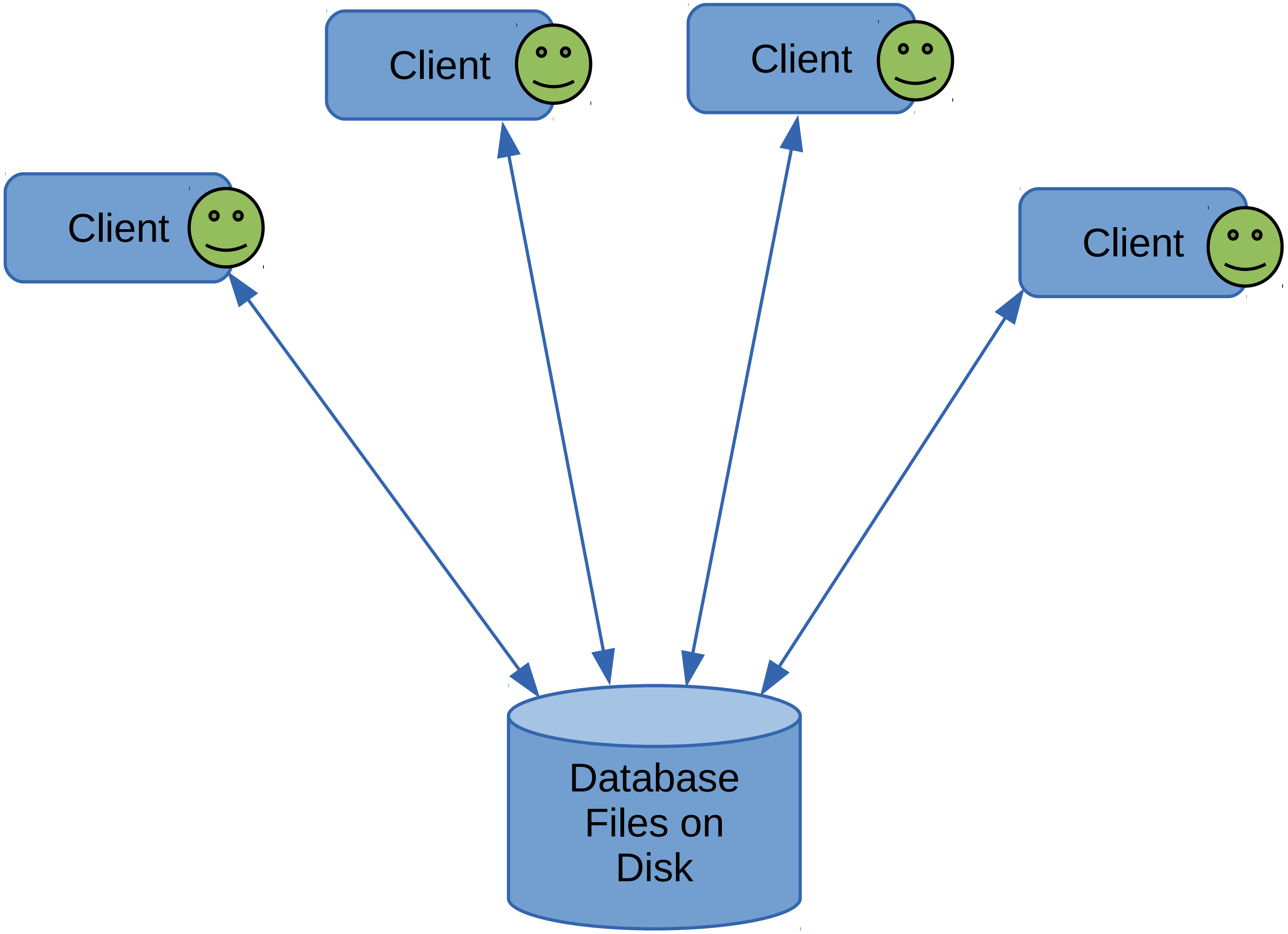
```
SELECT
  blob.rid,
  uuid,
  datetime(event.mtime,'localtime') AS timestamp,
  coalesce(ecomment, comment),
  coalesce(euser, user),
  (SELECT count(*) FROM plink WHERE pid=blob.rid AND isprim=1),
  (SELECT count(*) FROM plink WHERE cid=blob.rid),
  NOT EXISTS(SELECT 1 FROM plink
              WHERE pid=blob.rid
               AND coalesce((SELECT value FROM tagxref
                             WHERE tagid=8 AND rid=plink.pid), 'trunk')
                 = coalesce((SELECT value FROM tagxref
                             WHERE tagid=8 AND rid=plink.cid), 'trunk')),
  bgcolor,
  event.type,
  (SELECT group_concat(substr(tagname,5), ', ') FROM tag, tagxref
    WHERE tagname GLOB 'sym-*' AND tag.tagid=tagxref.tagid
      AND tagxref.rid=blob.rid AND tagxref.tagtype>0),
  tagid,
  brief
 FROM event JOIN blob
WHERE blob.rid=event.objid
 ORDER BY event.mtime DESC LIMIT 20;
```


Client
Client
Client
Client
Database Engine
Database Files on Disk


Client
Client
Client
Client
Database Engine
Database Files on Disk



Cannot connected to database



First code: 2000-05-29

SQLite used in....

- Every Android device (~2 billion)
- Every Mac and iOS device (~1 billion)
- Every Win10 machine (~500 million)
- Every Chrome and Firefox browser (~2 billion)
- Every Skype, iTunes, WhatApp (~2 billion)
- Millions of other applications
- Many billions of running instances
- 100s of billions, perhap trillions, of databases

More Copies of SQLite Than...

- Linux
- Windows
- MacOS and iOS
- All other database engines combined
- Any application
- Any other library[1]

[1]*except maybe zLib*

One File Of C-code

sqlite3.c

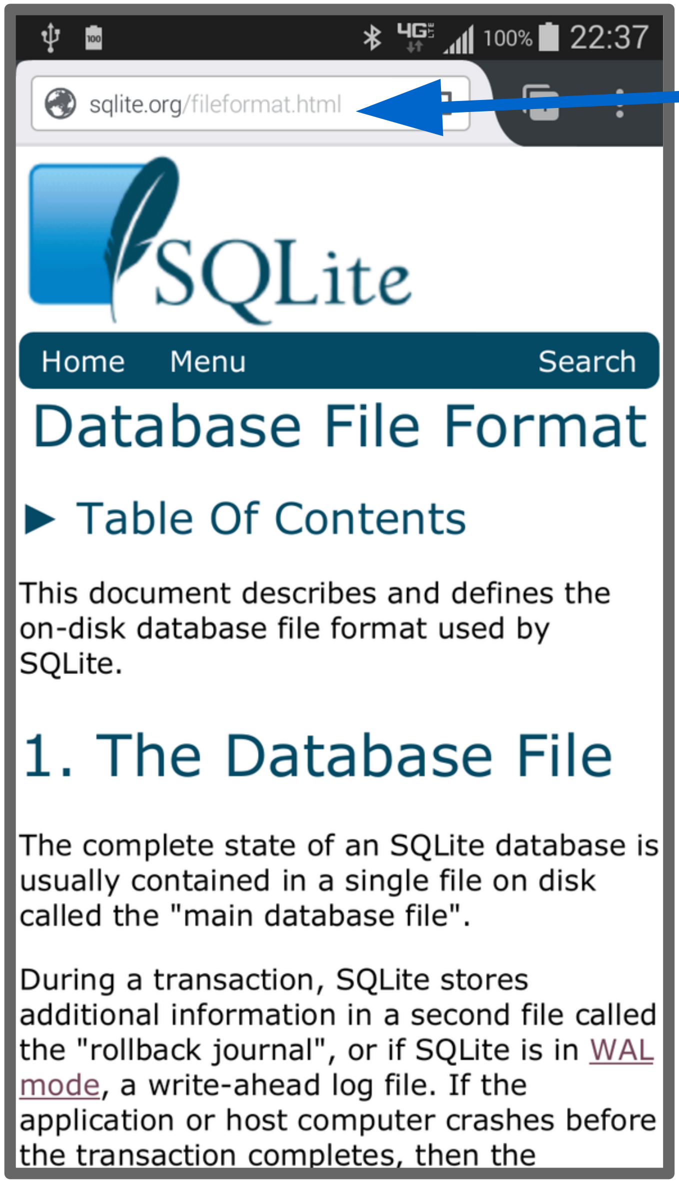
- 204K lines
- 125K SLOC[1]
- 7.2MB

Also: **sqlite3.h**

- 10.7K lines
- 1.5K SLOC
- 0.5MB

[1]SLOC: “Source Lines Of Code” - Lines of code not counting comments and blank lines.

Open File Format



Database File Format

▶ Table Of Contents

This document describes and defines the on-disk database file format used by SQLite.

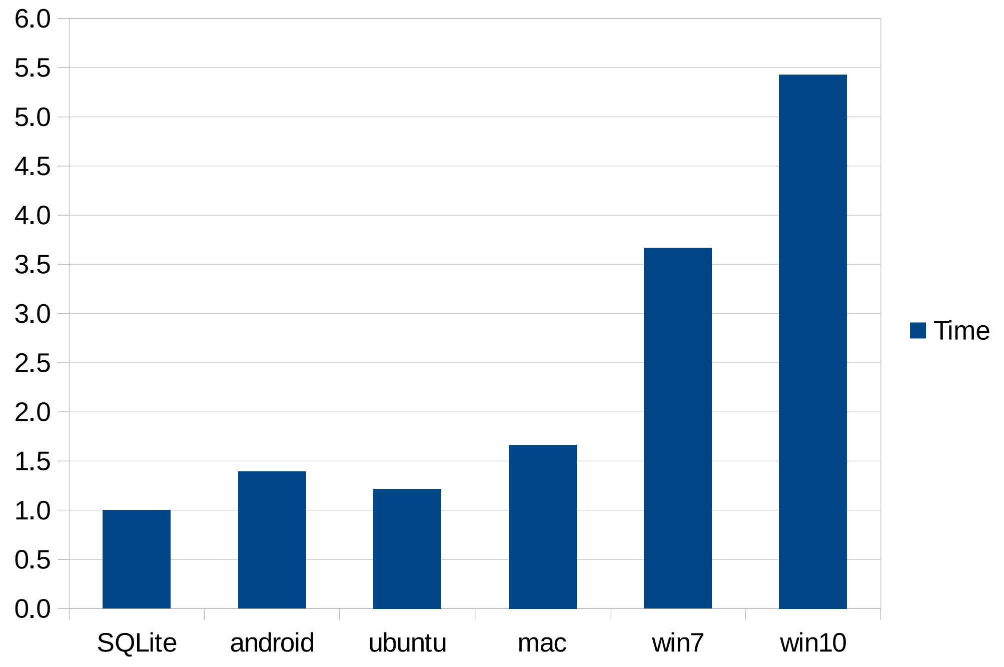
1. The Database File

The complete state of an SQLite database is usually contained in a single file on disk called the "main database file".

During a transaction, SQLite stores additional information in a second file called the "rollback journal", or if SQLite is in WAL mode, a write-ahead log file. If the application or host computer crashes before the transaction completes, then the

- sqlite.org/fileformat.html
- Single-file database
- Cross-platform
  - 32-bit ↔ 64-bit
  - little-endian ↔ big-endian
- Backwards-compatible
- Space efficient encoding
- Readable by 3rd-party tools

Faster Than The File System



Time to read 100,000 BLOBs with average size of 10,000 bytes from SQLite versus directly from a file on disk.

https://sqlite.org/fasterthanfs.html

Aviation-Grade Testing

- DO-178B development process
- 100% MC/DC, as-deployed, with independence

results in

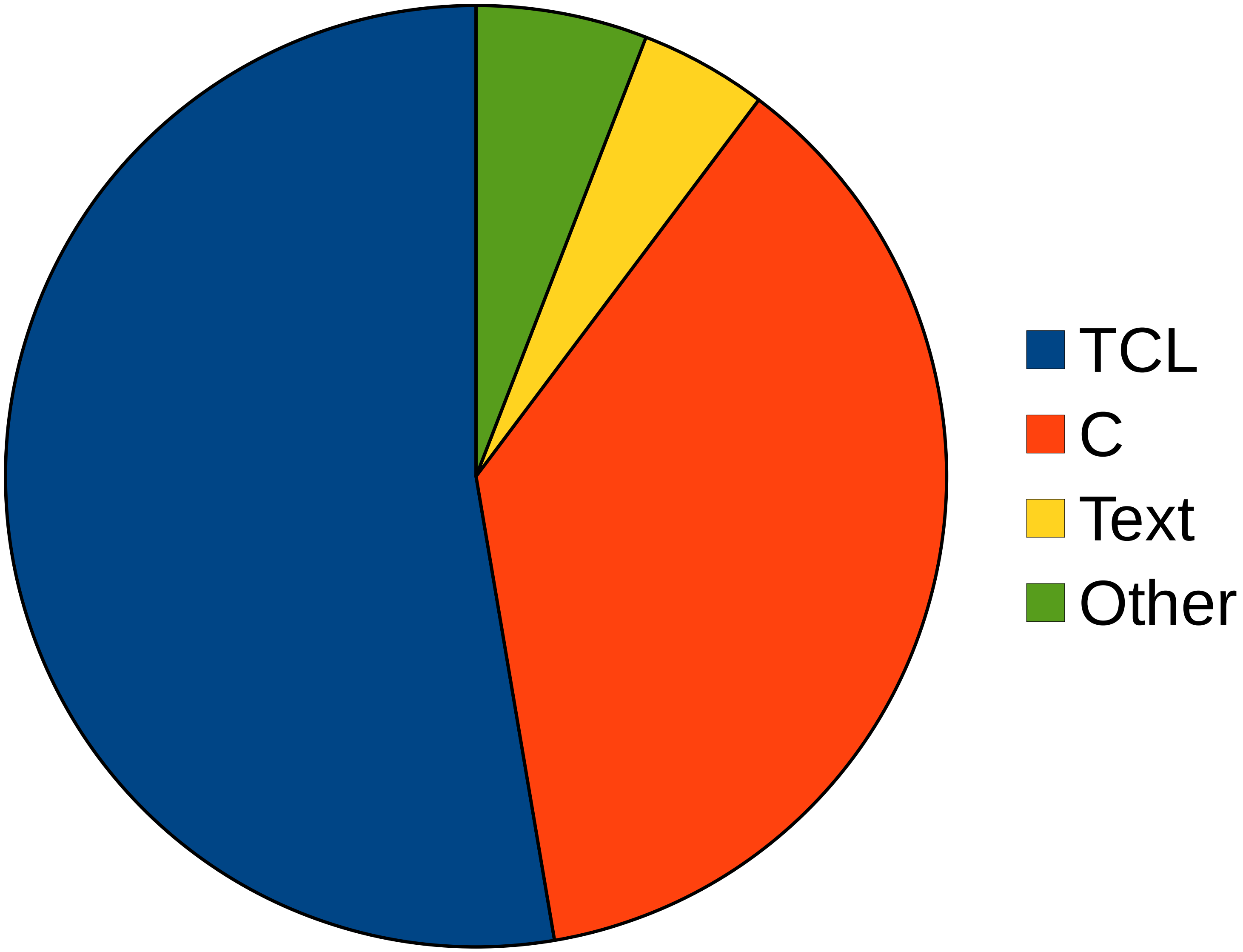
- Robust and reliable code - Very few bugs
- Refactor and optimize without breaking things
- Maintainable by just 3 people

https://sqlite.org/testing.html

Copyright

SQLite is a Tcl Extension that has escaped into the wild

SQLite Mostly Written In TCL



tclsqlite.c

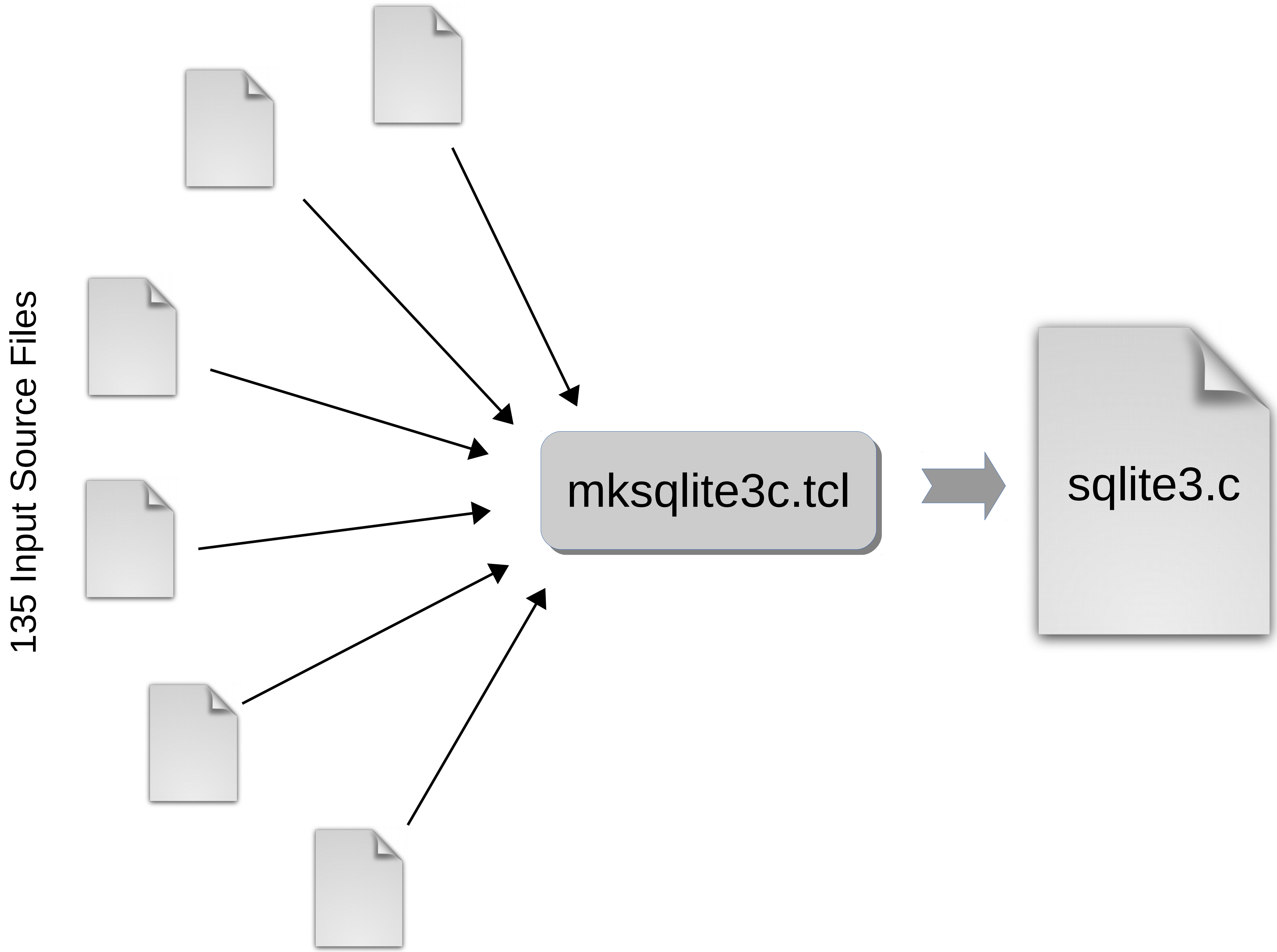
- Implements a TCL interface to SQLite
- Included with the first SQLite check-in on 2000-05-29
- The only language interface (other than C/C++) supported by the SQLite core

Tcl-like Type System

- **SQLite 1** (2000..2001)
- **SQLite 2** (2001..2004)

$\}$ Everything is a string

- **SQLite 3** (2004 onward) $\rightarrow$ Dual representation


135 Input Source Files
mksqlite3c.tcl
sqlite3.c

Other Code Generator Scripts

- **mksqlite3h.tcl** → Create “sqlite3.h” from “sqlite.h.in”, inserting version strings, etc.
- **mkshellc.tcl** → Create “shell.c” from “shell.c.in” plus extensions
- **mkopcodec.tcl mkopcodeh.tcl** → Assign numbers to symbolic opcode names (ex: OP_Add) subject to various constraints
- **addopcodes.tcl** → Add supplimental token codes to the lemon-generated parse.h file

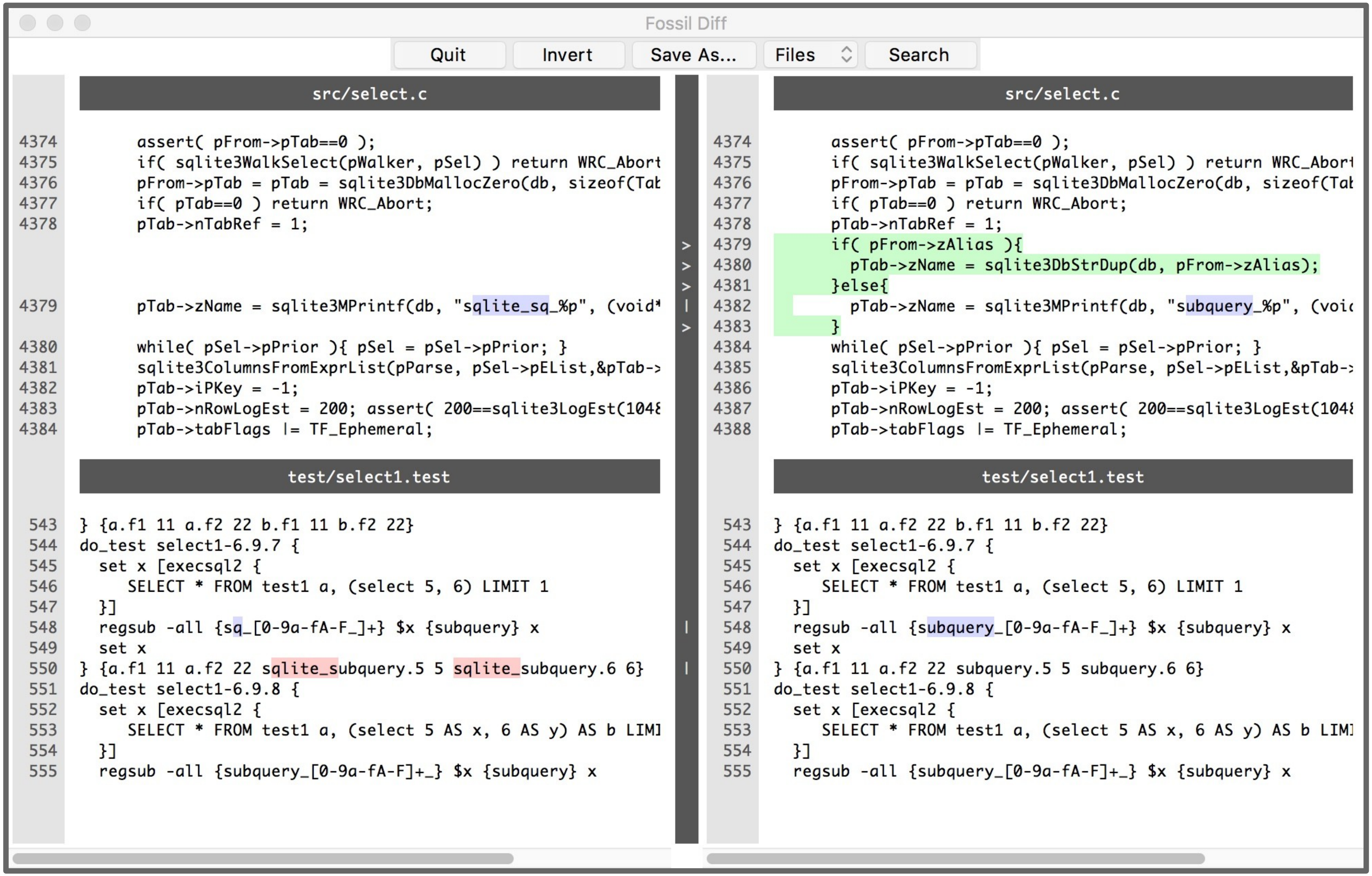
The sqlite3_analyzer.exe Utility

- **sqlite3_analyzer** *filename.db*
  - 1000’s of lines of output
  - Relative sizes of all tables
  - Average and max row sizes
  - Packing efficiency and overhead
- Precompiled binaries available on all major platforms
- Used by tens of thousands of developers
- Written in Tcl!

sqlite3_analyzer --tclsh

```
set line {}
while {![eof stdin]} {
  if {$line!=""} {
    puts -nonewline "> "
  } else {
    puts -nonewline "% "
  }
  flush stdout
  append line [gets stdin]
  if {[info complete $line]} {
    if {[catch {uplevel #0 $line} result]} {
      puts stderr "Error: $result"
    } elseif {$result!=""} {
      puts $result
    }
    set line {}
  } else {
    append line \n
  }
}
```

fossil diff --tk



src/select.c

```
      assert( pFrom->pTab==0 );
      if( sqlite3WalkSelect(pWalker, pSel) ) return WRC_Abort
      pFrom->pTab = pTab = sqlite3DbMallocZero(db, sizeof(Tab
      if( pTab==0 ) return WRC_Abort;
      pTab->nTabRef = 1;



      pTab->zName = sqlite3MPrintf(db, "sqlite_sq_%p", (void*

      while( pSel->pPrior ){ pSel = pSel->pPrior; }
      sqlite3ColumnsFromExprList(pParse, pSel->pEList,&pTab->
      pTab->iPKey = -1;
      pTab->nRowLogEst = 200; assert( 200==sqlite3LogEst(1048
      pTab->tabFlags |= TF_Ephemeral;
```

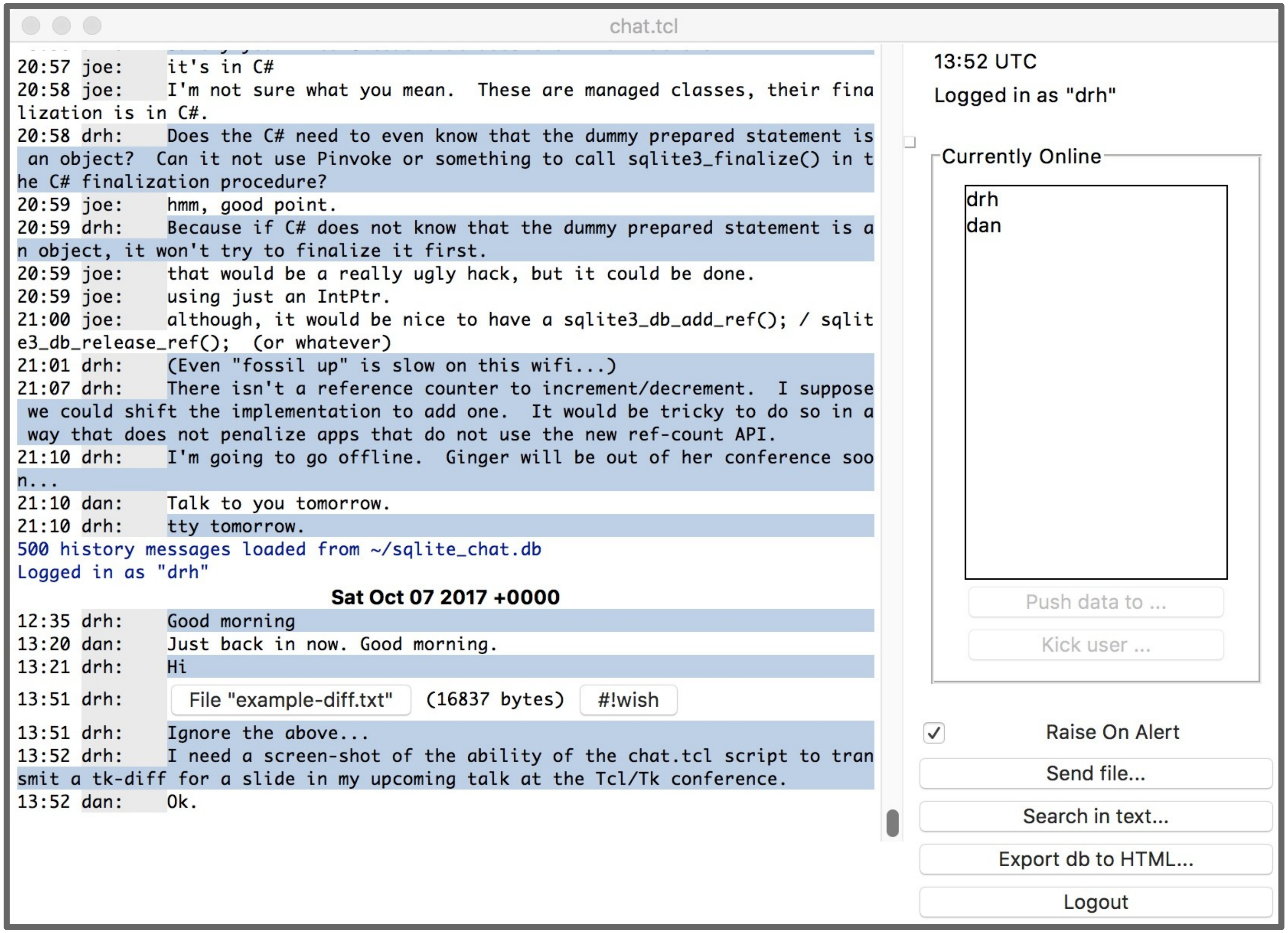
```
      assert( pFrom->pTab==0 );
      if( sqlite3WalkSelect(pWalker, pSel) ) return WRC_Abort
      pFrom->pTab = pTab = sqlite3DbMallocZero(db, sizeof(Tab
      if( pTab==0 ) return WRC_Abort;
      pTab->nTabRef = 1;
      if( pFrom->zAlias ){
        pTab->zName = sqlite3DbStrDup(db, pFrom->zAlias);
      }else{
        pTab->zName = sqlite3MPrintf(db, "subquery_%p", (voi
      }
      while( pSel->pPrior ){ pSel = pSel->pPrior; }
      sqlite3ColumnsFromExprList(pParse, pSel->pEList,&pTab->
      pTab->iPKey = -1;
      pTab->nRowLogEst = 200; assert( 200==sqlite3LogEst(1048
      pTab->tabFlags |= TF_Ephemeral;
```

test/select1.test

```
} {a.f1 11 a.f2 22 b.f1 11 b.f2 22}
do_test select1-6.9.7 {
  set x [execsql2 {
     SELECT * FROM test1 a, (select 5, 6) LIMIT 1
  }]
  regsub -all {sq_[0-9a-fA-F_]+} $x {subquery} x
  set x
} {a.f1 11 a.f2 22 sqlite_subquery.5 5 sqlite_subquery.6 6}
do_test select1-6.9.8 {
  set x [execsql2 {
     SELECT * FROM test1 a, (select 5 AS x, 6 AS y) AS b LIMI
  }]
  regsub -all {subquery_[0-9a-fA-F]+_} $x {subquery} x
```

```
} {a.f1 11 a.f2 22 b.f1 11 b.f2 22}
do_test select1-6.9.7 {
  set x [execsql2 {
     SELECT * FROM test1 a, (select 5, 6) LIMIT 1
  }]
  regsub -all {subquery_[0-9a-fA-F_]+} $x {subquery} x
  set x
} {a.f1 11 a.f2 22 subquery.5 5 subquery.6 6}
do_test select1-6.9.8 {
  set x [execsql2 {
     SELECT * FROM test1 a, (select 5 AS x, 6 AS y) AS b LIMI
  }]
  regsub -all {subquery_[0-9a-fA-F]+_} $x {subquery} x
```

chat.tcl



```
20:57 joe:     it's in C#
20:58 joe:     I'm not sure what you mean.  These are managed classes, their fina
lization is in C#.
20:58 drh:     Does the C# need to even know that the dummy prepared statement is
 an object?  Can it not use Pinvoke or something to call sqlite3_finalize() in t
he C# finalization procedure?
20:59 joe:     hmm, good point.
20:59 drh:     Because if C# does not know that the dummy prepared statement is a
n object, it won't try to finalize it first.
20:59 joe:     that would be a really ugly hack, but it could be done.
20:59 joe:     using just an IntPtr.
21:00 joe:     although, it would be nice to have a sqlite3_db_add_ref(); / sqlit
e3_db_release_ref();  (or whatever)
21:01 drh:     (Even "fossil up" is slow on this wifi...)
21:07 drh:     There isn't a reference counter to increment/decrement.  I suppose
 we could shift the implementation to add one.  It would be tricky to do so in a
 way that does not penalize apps that do not use the new ref-count API.
21:10 drh:     I'm going to go offline.  Ginger will be out of her conference soo
n...
21:10 dan:     Talk to you tomorrow.
21:10 drh:     tty tomorrow.
500 history messages loaded from ~/sqlite_chat.db
Logged in as "drh"
                      Sat Oct 07 2017 +0000
12:35 drh:     Good morning
13:20 dan:     Just back in now. Good morning.
13:21 drh:     Hi
13:51 drh:     File "example-diff.txt"  (16837 bytes)  #!wish
13:51 drh:     Ignore the above...
13:52 drh:     I need a screen-shot of the ability of the chat.tcl script to tran
smit a tk-diff for a slide in my upcoming talk at the Tcl/Tk conference.
13:52 dan:     Ok.
```

13:52 UTC

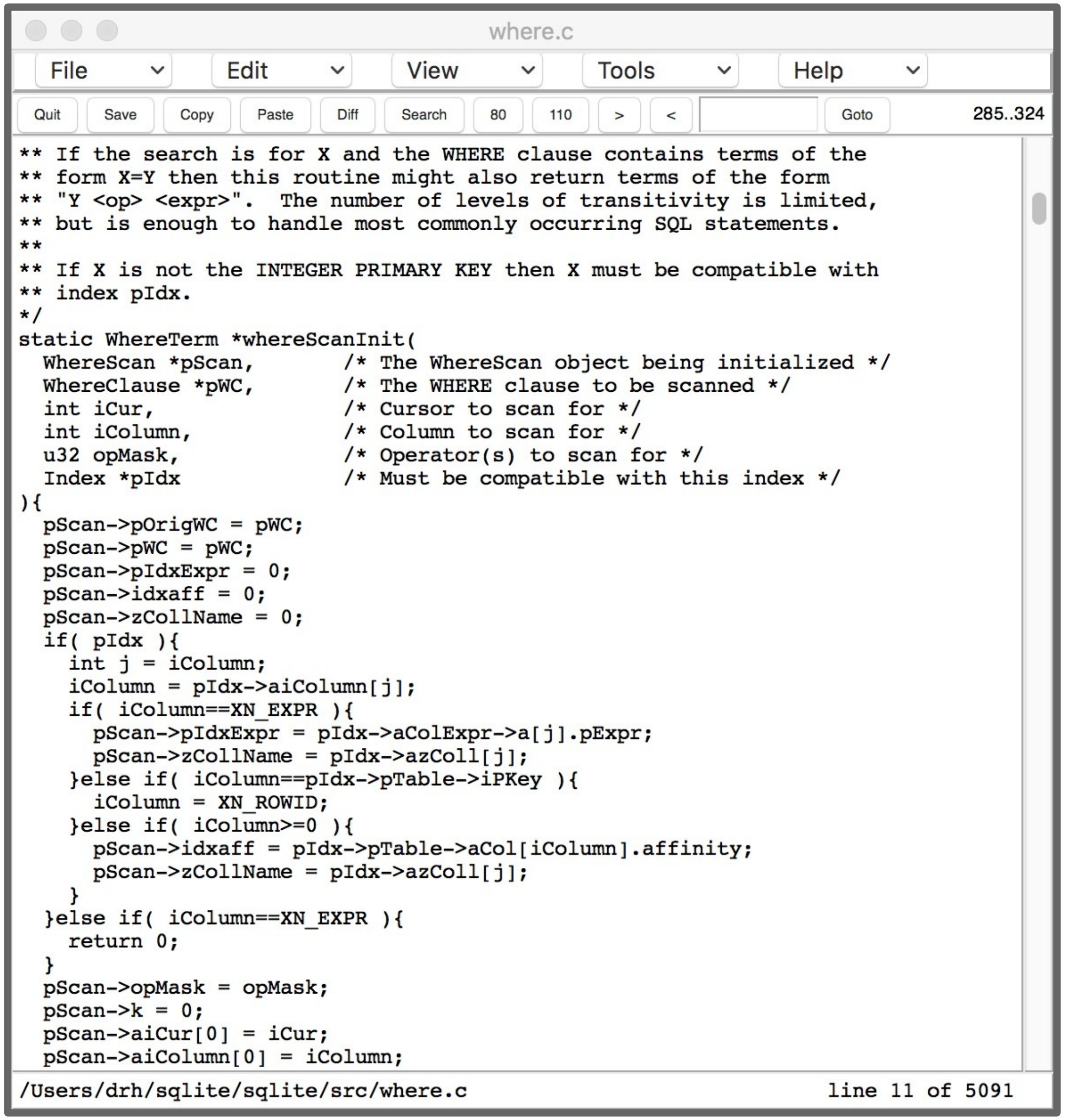
Logged in as "drh"

Currently Online

drh
dan

Raise On Alert [✓]

The “e” Text Editor



```
** If the search is for X and the WHERE clause contains terms of the
** form X=Y then this routine might also return terms of the form
** "Y <op> <expr>".  The number of levels of transitivity is limited,
** but is enough to handle most commonly occurring SQL statements.
**
** If X is not the INTEGER PRIMARY KEY then X must be compatible with
** index pIdx.
*/
static WhereTerm *whereScanInit(
  WhereScan *pScan,        /* The WhereScan object being initialized */
  WhereClause *pWC,        /* The WHERE clause to be scanned */
  int iCur,                /* Cursor to scan for */
  int iColumn,             /* Column to scan for */
  u32 opMask,              /* Operator(s) to scan for */
  Index *pIdx              /* Must be compatible with this index */
){
  pScan->pOrigWC = pWC;
  pScan->pWC = pWC;
  pScan->pIdxExpr = 0;
  pScan->idxaff = 0;
  pScan->zCollName = 0;
  if( pIdx ){
    int j = iColumn;
    iColumn = pIdx->aiColumn[j];
    if( iColumn==XN_EXPR ){
      pScan->pIdxExpr = pIdx->aColExpr->a[j].pExpr;
      pScan->zCollName = pIdx->azColl[j];
    }else if( iColumn==pIdx->pTable->iPKey ){
      iColumn = XN_ROWID;
    }else if( iColumn>=0 ){
      pScan->idxaff = pIdx->pTable->aCol[iColumn].affinity;
      pScan->zCollName = pIdx->azColl[j];
    }
  }else if( iColumn==XN_EXPR ){
    return 0;
  }
  pScan->opMask = opMask;
  pScan->k = 0;
  pScan->aiCur[0] = iCur;
  pScan->aiColumn[0] = iColumn;
```

```
/Users/drh/sqlite/sqlite/src/where.c                     line 11 of 5091
```

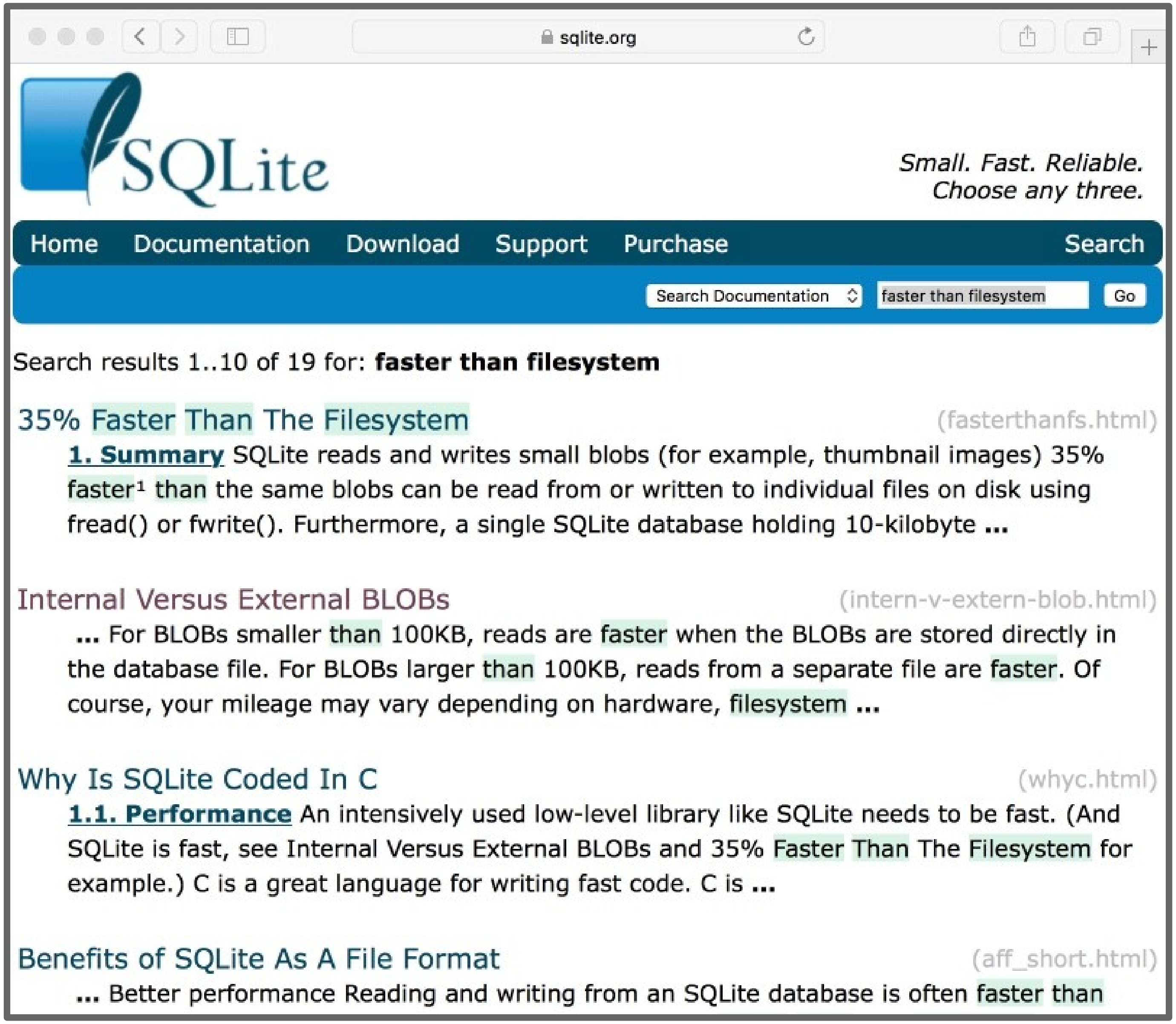
The “open” Command

```
$ open main.c
$ open logo.jpg
$ open doc.html
$ man mmap | open -f
```

SQLite Documentation

- 137 files of HTML with extensions:
  - <tcl>...</tcl>
  - [hyperlink]
  - ^(...)^
- Translated into pure HTML using TCL
- Many docs extracted from source code comments using scripts inside <tcl>...</tcl>
  - C/C++ interface spec from sqlite3.h
  - Byte-code opcodes from vdbe.c
- Automatic requirement numbering and tracking

Documentation Search



Search results 1..10 of 19 for: **faster than filesystem**

35% Faster Than The Filesystem (fasterthanfs.html)

1. Summary SQLite reads and writes small blobs (for example, thumbnail images) 35% faster¹ than the same blobs can be read from or written to individual files on disk using fread() or fwrite(). Furthermore, a single SQLite database holding 10-kilobyte **...**

Internal Versus External BLOBs (intern-v-extern-blob.html)

... For BLOBs smaller than 100KB, reads are faster when the BLOBs are stored directly in the database file. For BLOBs larger than 100KB, reads from a separate file are faster. Of course, your mileage may vary depending on hardware, filesystem **...**

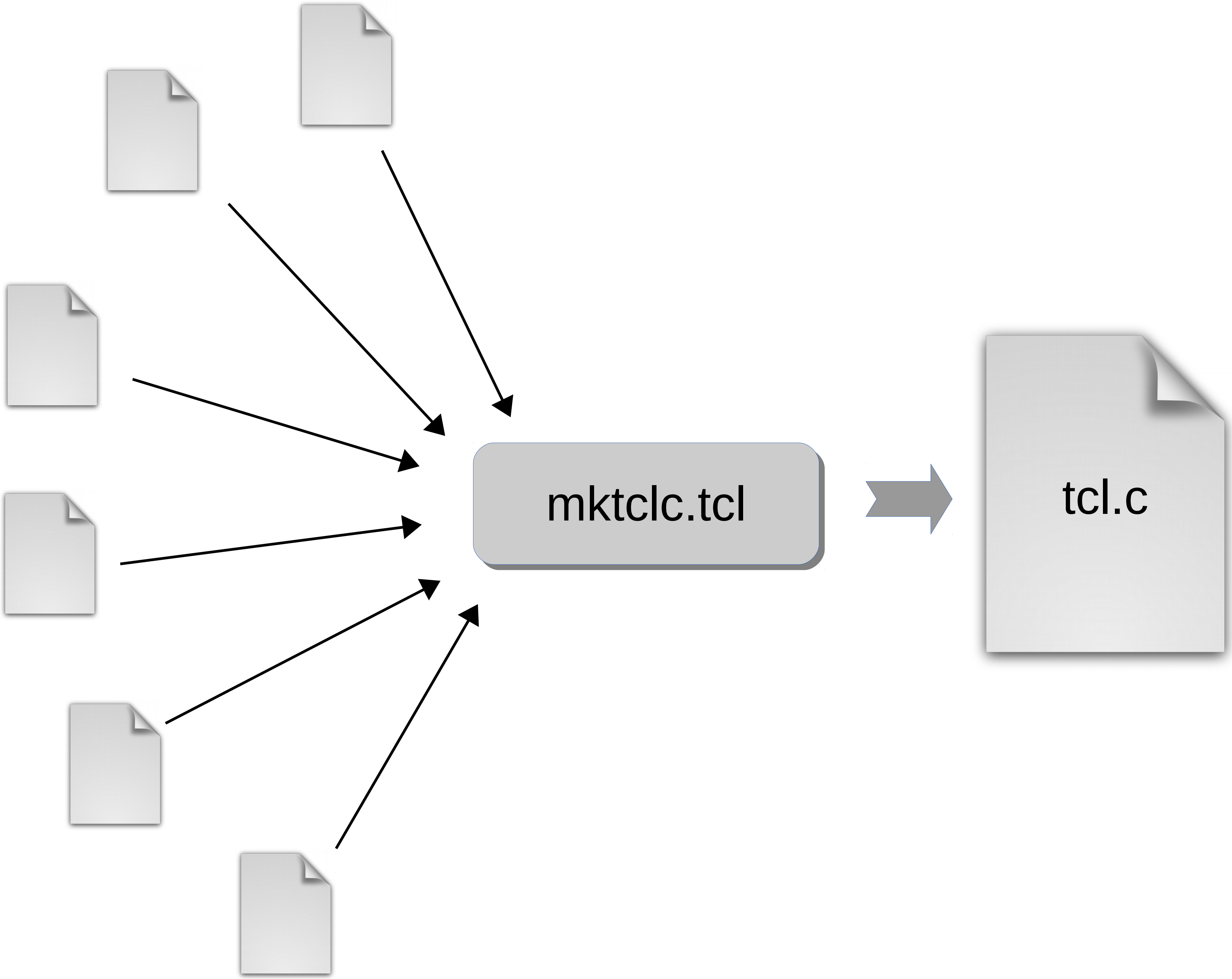
Why Is SQLite Coded In C (whyc.html)

1.1. Performance An intensively used low-level library like SQLite needs to be fast. (And SQLite is fast, see Internal Versus External BLOBs and 35% Faster Than The Filesystem for example.) C is a great language for writing fast code. C is **...**

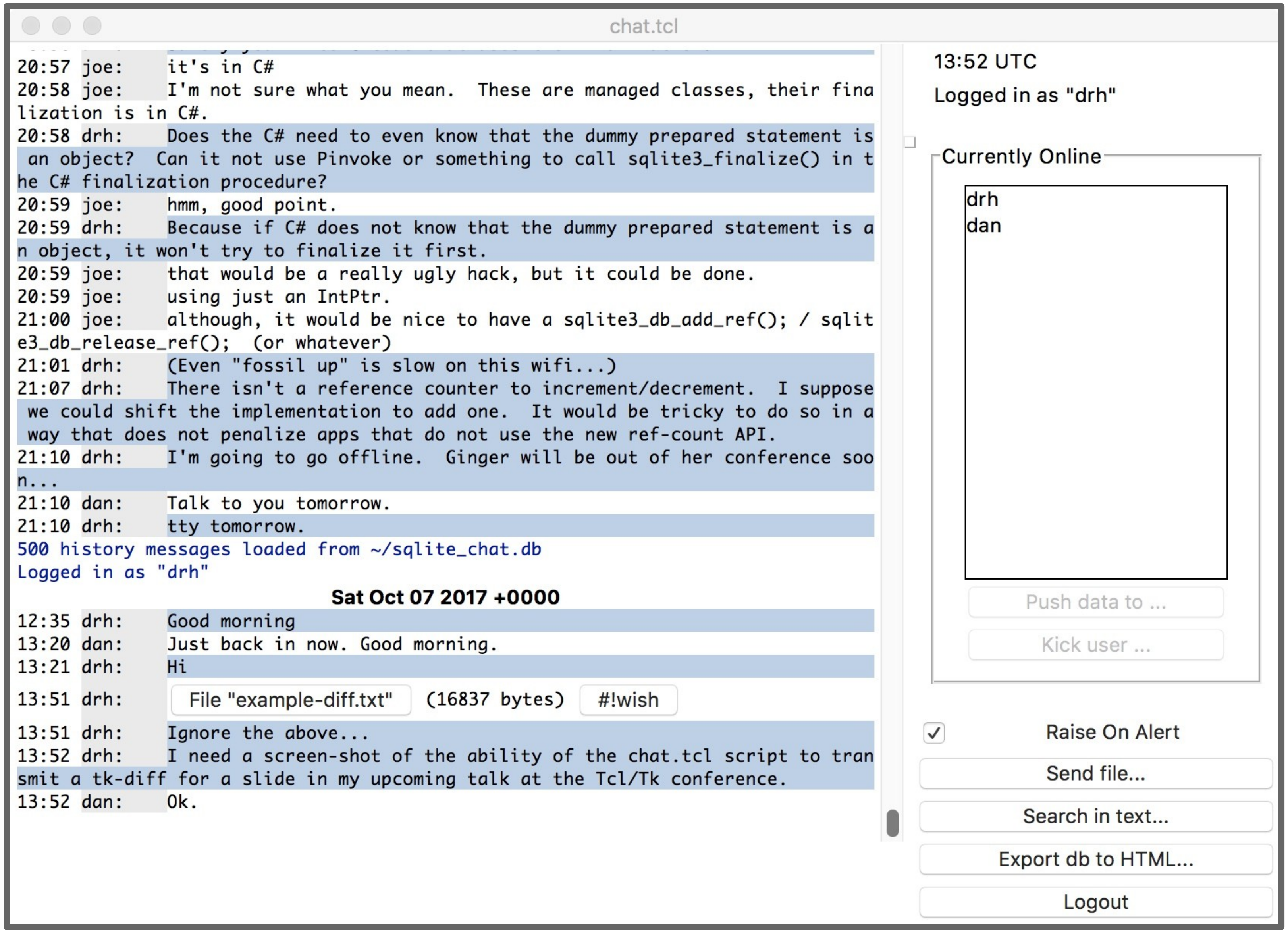
Benefits of SQLite As A File Format (aff_short.html)

... Better performance Reading and writing from an SQLite database is often faster than

SQLite


mktclc.tcl
tcl.c

Can We Have Built-in Crypto?



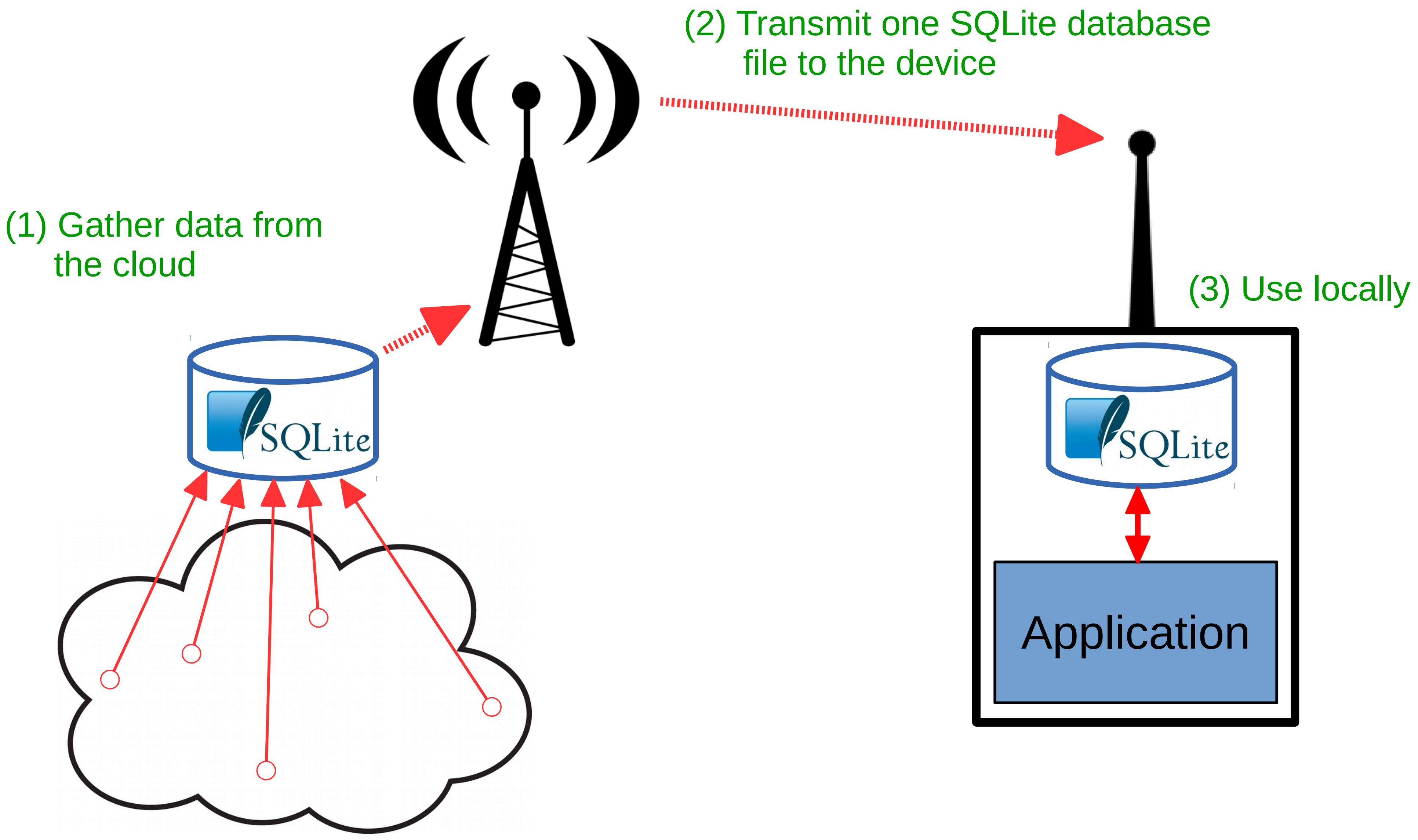
```
20:57 joe:    it's in C#
20:58 joe:    I'm not sure what you mean.  These are managed classes, their fina
lization is in C#.
20:58 drh:    Does the C# need to even know that the dummy prepared statement is
 an object?  Can it not use Pinvoke or something to call sqlite3_finalize() in t
he C# finalization procedure?
20:59 joe:    hmm, good point.
20:59 drh:    Because if C# does not know that the dummy prepared statement is a
n object, it won't try to finalize it first.
20:59 joe:    that would be a really ugly hack, but it could be done.
20:59 joe:    using just an IntPtr.
21:00 joe:    although, it would be nice to have a sqlite3_db_add_ref(); / sqlit
e3_db_release_ref();  (or whatever)
21:01 drh:    (Even "fossil up" is slow on this wifi...)
21:07 drh:    There isn't a reference counter to increment/decrement.  I suppose
 we could shift the implementation to add one.  It would be tricky to do so in a
 way that does not penalize apps that do not use the new ref-count API.
21:10 drh:    I'm going to go offline.  Ginger will be out of her conference soo
n...
21:10 dan:    Talk to you tomorrow.
21:10 drh:    tty tomorrow.
500 history messages loaded from ~/sqlite_chat.db
Logged in as "drh"
                         Sat Oct 07 2017 +0000
12:35 drh:    Good morning
13:20 dan:    Just back in now. Good morning.
13:21 drh:    Hi
13:51 drh:    File "example-diff.txt"  (16837 bytes)  #!wish
13:51 drh:    Ignore the above...
13:52 drh:    I need a screen-shot of the ability of the chat.tcl script to tran
smit a tk-diff for a slide in my upcoming talk at the Tcl/Tk conference.
13:52 dan:    Ok.
```

Bringing SQLite Back Home

Bringing SQLite Back Home

1) Built-in **sqlite3** command

SQLite = Data Container


(2) Transmit one SQLite database file to the device
(1) Gather data from the cloud
(3) Use locally
SQLite
SQLite
Application

SQLite Archiver

```
CREATE TABLE sqlar(
  name TEXT PRIMARY KEY,  -- name of the file
  mode INT,               -- access permissions
  mtime INT,              -- last modification time
  sz INT,                 -- original file size
  data BLOB               -- compressed content
);
```

- https://sqlite.org/sqlar
- Transactional
- Concurrent & random access
- File size similar to ZIP

SQLite versus ZIP

Ye	Container for files	Ye
s	10^{12} instances in the wild	s
Ye	Compact	Ye
s	Well-defined, open format	s
Ye	Container for small objects	Ye
s	Cross-platform objects	s
Ye	ACID transactions	Ye
s	Query language	s
Ye	Secondary indexes	No
s	Triggers and Views	No
Ye		No
s		No

Bringing SQLite Back Home

1) Built-in **sqlite3** command
2) tclsh *database*

Bringing SQLite Back Home

1) Built-in **sqlite3** command
2) tclsh *database*
3) $tcl_library points to a database

Bringing SQLite Back Home

1) Built-in **sqlite3** command
2) tclsh *database*
3) $tcl_library points to a database
4) Database as a TclVFS

Bringing SQLite Back Home

1) Built-in **sqlite3** command
2) tclsh *database*
3) $tcl_library points to a database
4) Database as a TclVFS
5) Tk-based graphical database explorer

Bringing SQLite Back Home

1) Built-in **sqlite3** command
2) tclsh *database*
3) $tcl_library points to a database
4) Database as a TclVFS
5) Tk-based graphical database explorer
6) Tk widget hierarchy or canvas as an SQLite virtual table

SQLite