The World's Most Popular TCL Extension

D. Richard Hipp
16th Annual Tcl/Tk Conference
Portland, OR
2009-09-30

“The only SQL database engine specifically designed to work with TCL”


PostgreSQL


Apache Derby


IBM
DB2

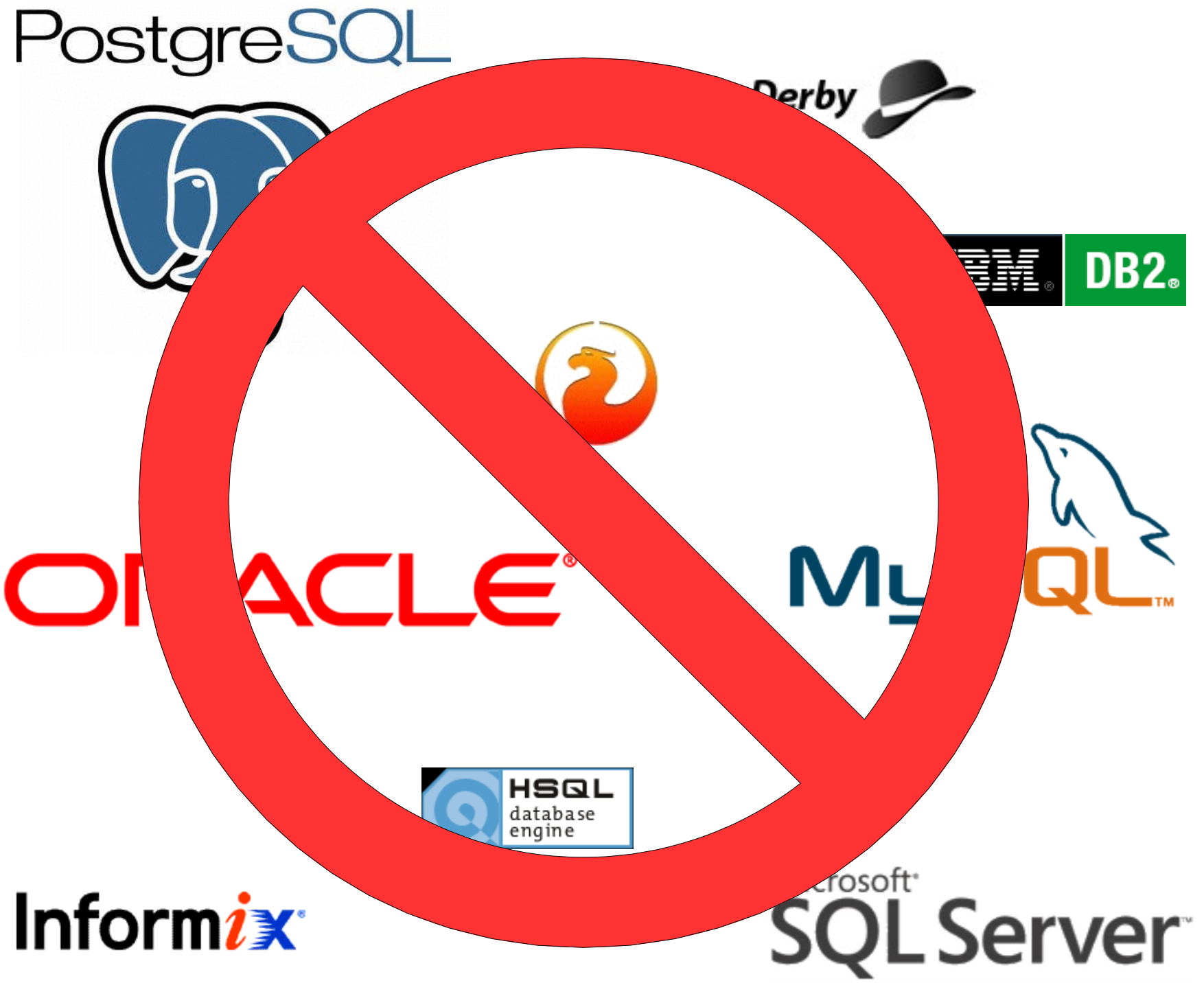

ORACLE


MySQL

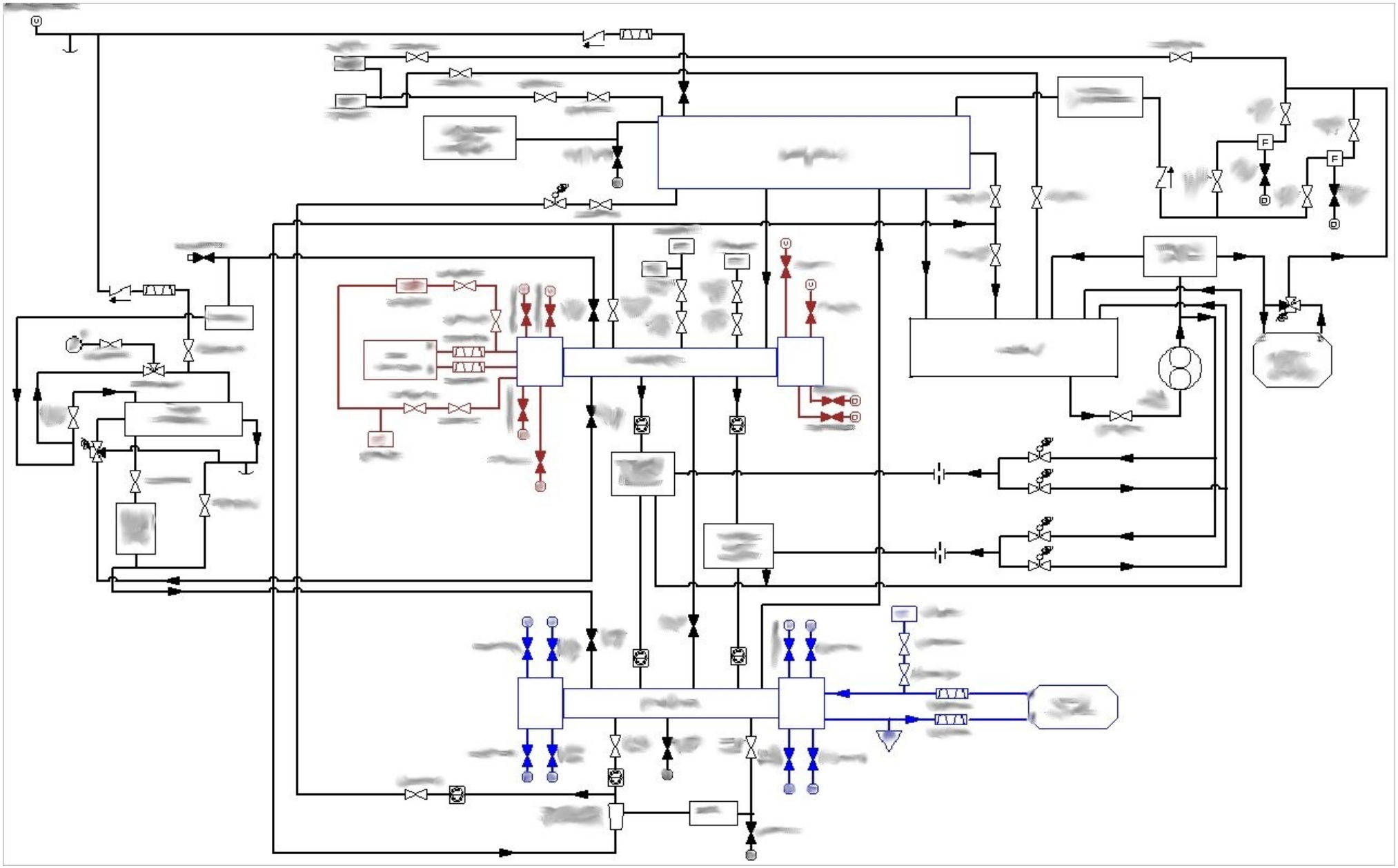

HSQL
database
engine

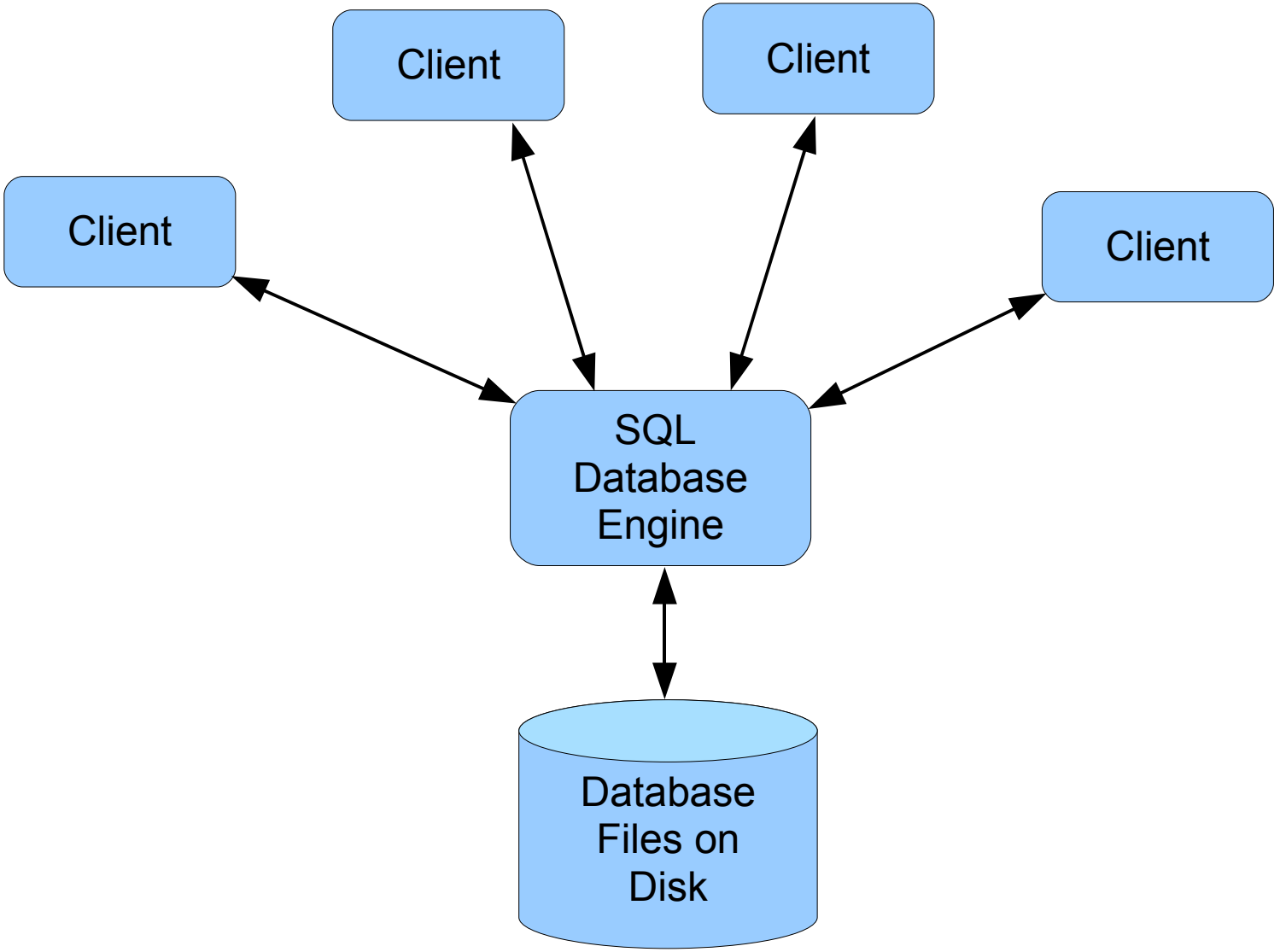

Informix

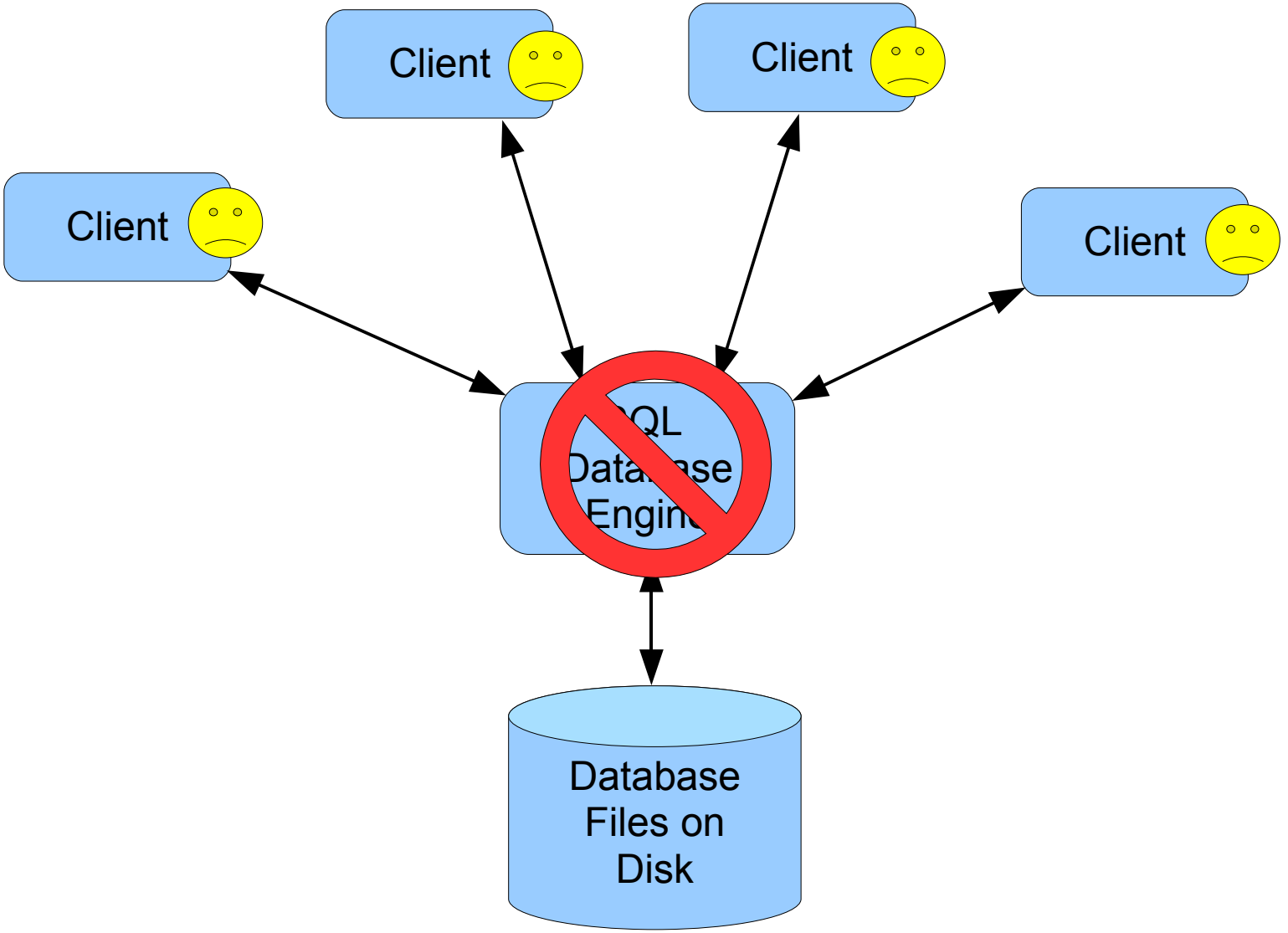

Microsoft
SQL Server

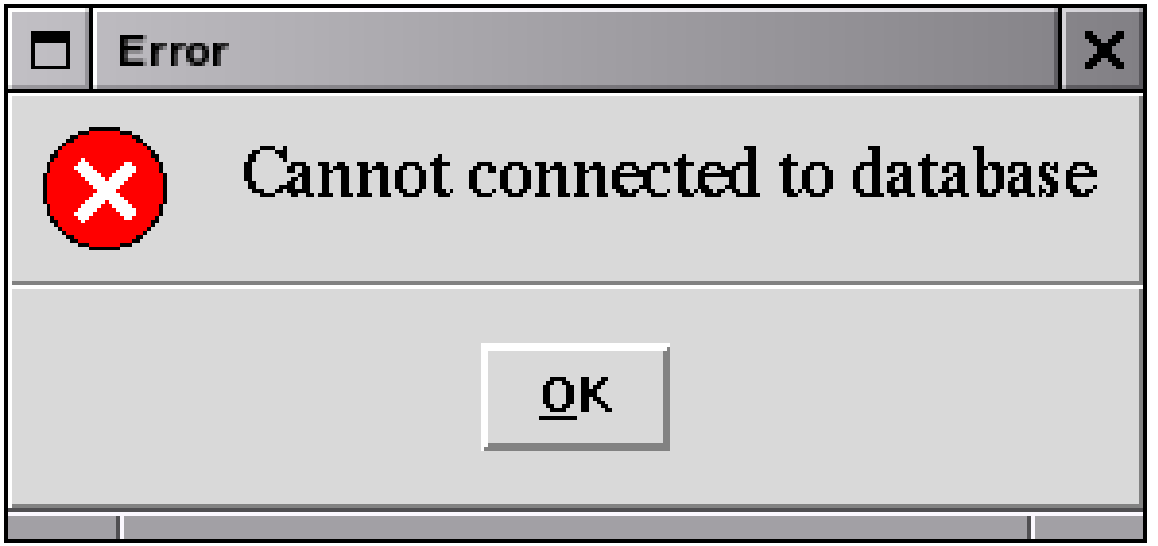

PostgreSQL
Derby
IBM
DB2
ORACLE
MySQL
HSQL
database
engine
Informix
Microsoft
SQL Server

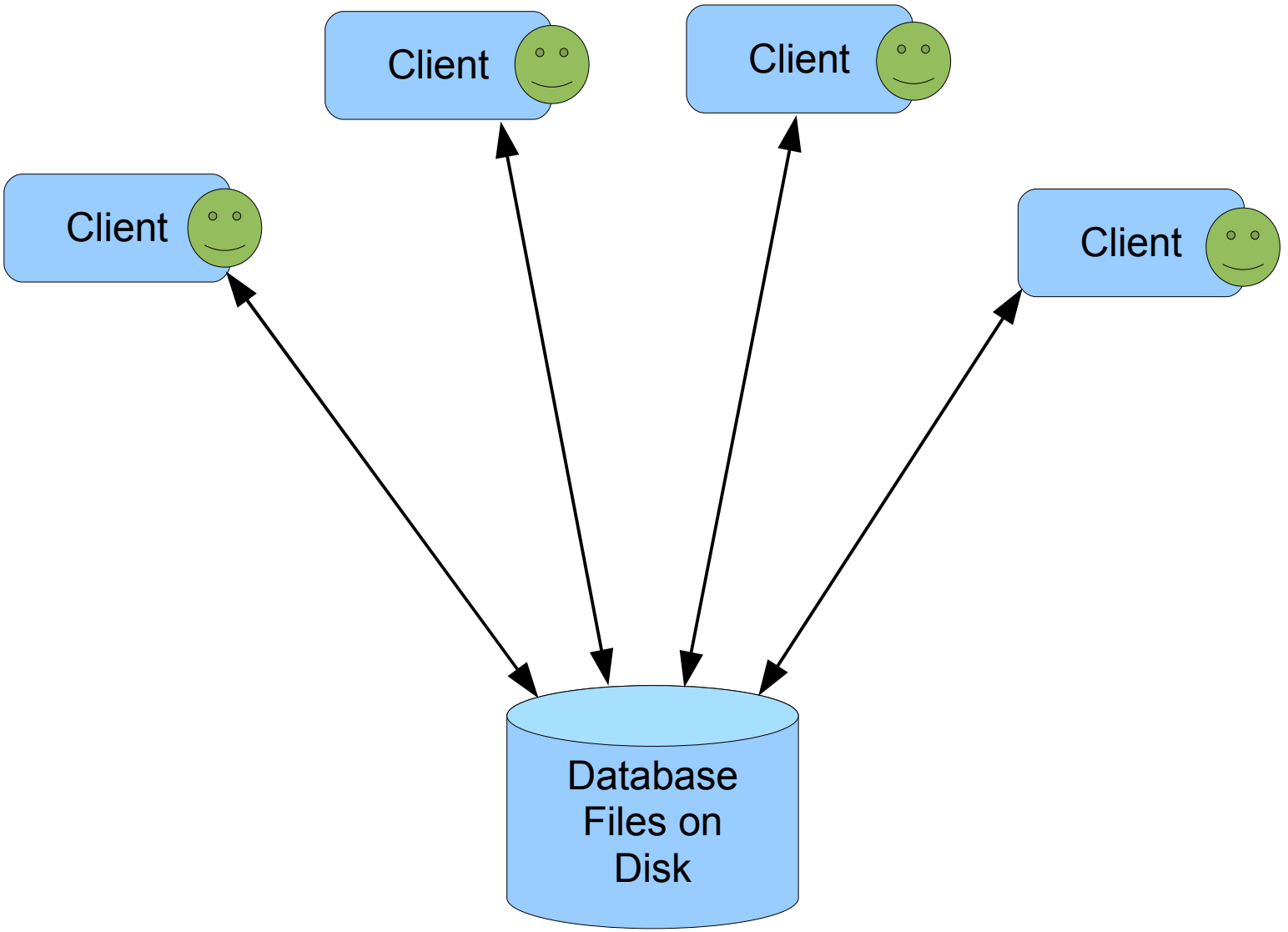
SQLite is a TCL Extension

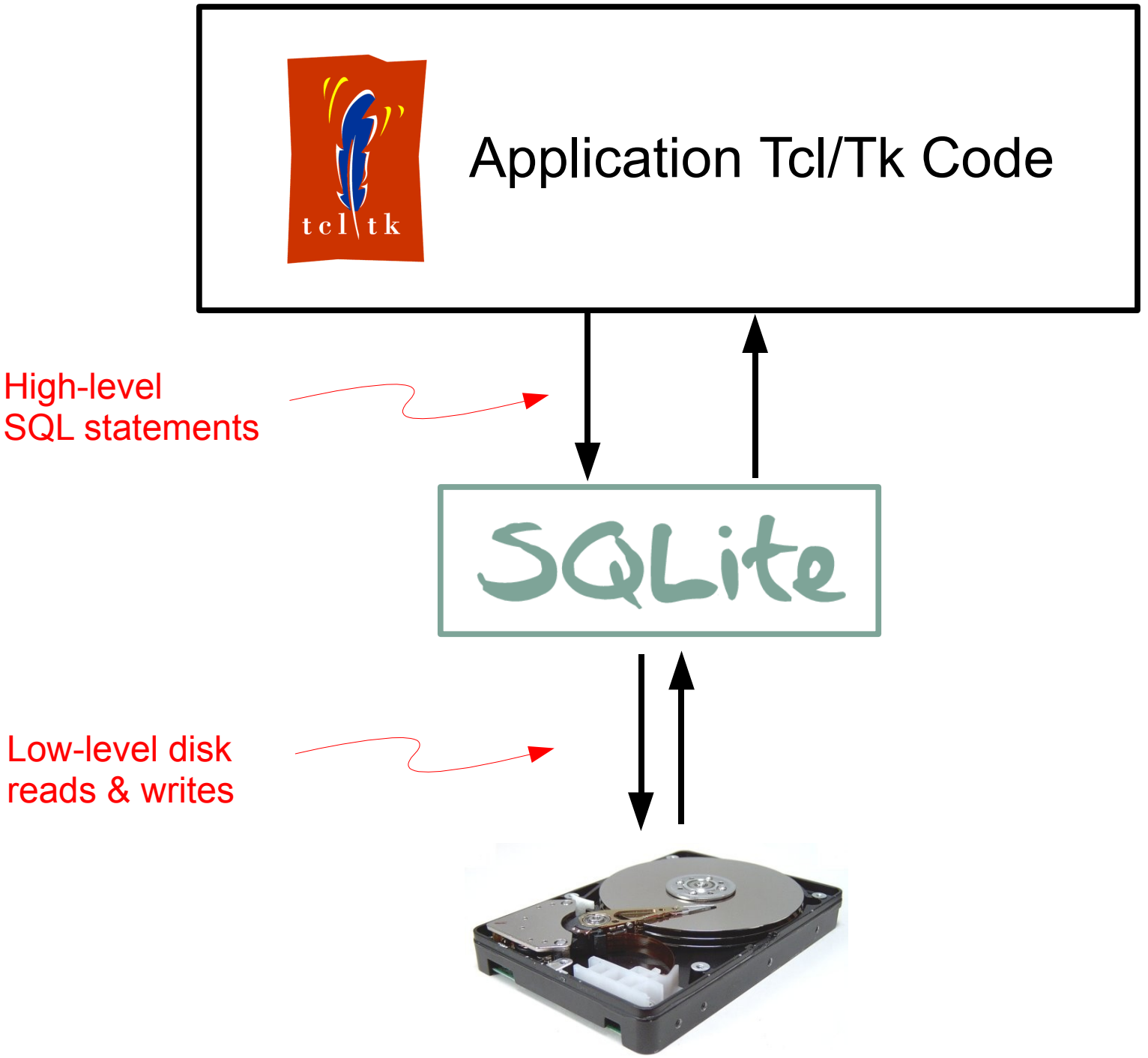



Client
Client
Client
Client
Client
SQL
Database
Engine
Database
Files on
Disk


Client
Client
Client
Client
Client
Database
Files on
Disk


Error
Cannot connected to database
OK


Client
Client
Client
Client
Database
Files on
Disk


Application Tcl/Tk Code
tcl tk
High-level
SQL statements
SQLite
Low-level disk
reads & writes

Why SQL?

- Program at a higher level
- Programming by specification
- Heads-up programming
  - Focus on your product, not on your underlying database
  - Maintain “situational awareness”

Do not underestimate the importance of situational awareness!

```
SELECT value FROM array
 WHERE name='userid';
```

Easier to do with an array or dict:

```
$array('userid')
```

How many lines of TCL are required to do this using just arrays and dicts?

```
SELECT eqptid, enclosureid
  FROM eqpt
 WHERE typeid IN (
   SELECT typeid FROM typespec
    WHERE attrid=(
      SELECT attrid FROM attribute
       WHERE name='detect_autoactuate'
    )
    AND value=1
   INTERSECT
   SELECT typeid FROM typespec
    WHERE attrid=(
      SELECT attrid FROM attribute
       WHERE name='algorithm'
    )
    AND value IN ('sensor','wetbulb')
 )
```


Firefox web browser | Faster, more secure, & customizable
http://www.sqli
Google
News
Projects
Biz
Jesus
Fossil: Timeline
SQLite Home Page
http://www.sqlite.org/draft/index.html
sqllogictest
http://www.sqlite.org/sqllogictest
Branch 3.6.4: Timeline
http://www.sqlite.org/br364/timeline
SQLite Documentation: SQLite Documenta
http://www.sqlite.org/docsrc
Branch 3.6.4: Further changes
http://www.sqlite.org/br364/wiki?name=Furth
SQLite CVSTrac
http://www.sqlite.org/cvstrac/tktview?tn=256
mozilla
Community
Abo
The Best
With more than 15,000 improvements, Firefox 3 is faster, safer and smarter than ever before.
Firefox 3
Free Download
3.0.6 for Mac OS X 10.4 and above
English (US) (17.2MB)
Release Notes
Other Systems and Languages
Done

```
SELECT h.url, h.title, f.url,
       (SELECT b.parent
          FROM moz_bookmarks b
               JOIN moz_bookmarks t ON t.id = b.parent AND t.parent != ?1
         WHERE b.type = 1 AND b.fk = h.id
         ORDER BY b.lastModified DESC LIMIT 1) AS parent,
       (SELECT b.title
          FROM moz_bookmarks b
               JOIN moz_bookmarks t ON t.id = b.parent AND t.parent != ?1
         WHERE b.type = 1 AND b.fk = h.id
         ORDER BY b.lastModified DESC LIMIT 1) AS bookmark,
       (SELECT GROUP_CONCAT(t.title, ',')
          FROM moz_bookmarks b
               JOIN moz_bookmarks t ON t.id = b.parent AND t.parent = ?1
         WHERE b.type = 1 AND b.fk = h.id) AS tags,
       h.visit_count, h.typed, h.frecency
  FROM moz_places_temp h
       LEFT OUTER JOIN moz_favicons f ON f.id = h.favicon_id
 WHERE h.frecency <> 0
UNION ALL
 SELECT h.url, h.title, f.url,
       (SELECT b.parent
          FROM moz_bookmarks b
               JOIN moz_bookmarks t ON t.id = b.parent AND t.parent != ?1
         WHERE b.type = 1 AND b.fk = h.id
         ORDER BY b.lastModified DESC LIMIT 1) AS parent,
       (SELECT b.title
          FROM moz_bookmarks b
               JOIN moz_bookmarks t ON t.id = b.parent AND t.parent != ?1
         WHERE b.type = 1 AND b.fk = h.id
         ORDER BY b.lastModified DESC LIMIT 1) AS bookmark,
       (SELECT GROUP_CONCAT(t.title, ',')
          FROM moz_bookmarks b
               JOIN moz_bookmarks t ON t.id = b.parent AND t.parent = ?1
         WHERE b.type = 1 AND b.fk = h.id) AS tags,
       h.visit_count, h.typed, h.frecency
  FROM moz_places h
       LEFT OUTER JOIN moz_favicons f ON f.id = h.favicon_id
 WHERE h.id NOT IN (SELECT id FROM moz_places_temp)
   AND h.frecency <> 0
ORDER BY 9 DESC
```

Other benefits SQL & SQLite:

- Persistent
- Transactional
- Cross-platform
- Widely known and understood
- Faster
- Fewer bugs
- SQL is good at doing the very few things that TCL does not already do well.

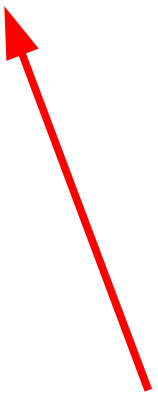

tcl/tk
SQLite

Aside: How do you classify SQLite in TCL?

- A “small language within a small language”?
- A meta-small language?

```
% package require sqlite3
3.6.19
% sqlite3 db database.db
%
```

New object for controlling the database



Name of the database file. A new one is created if it does not already exist.

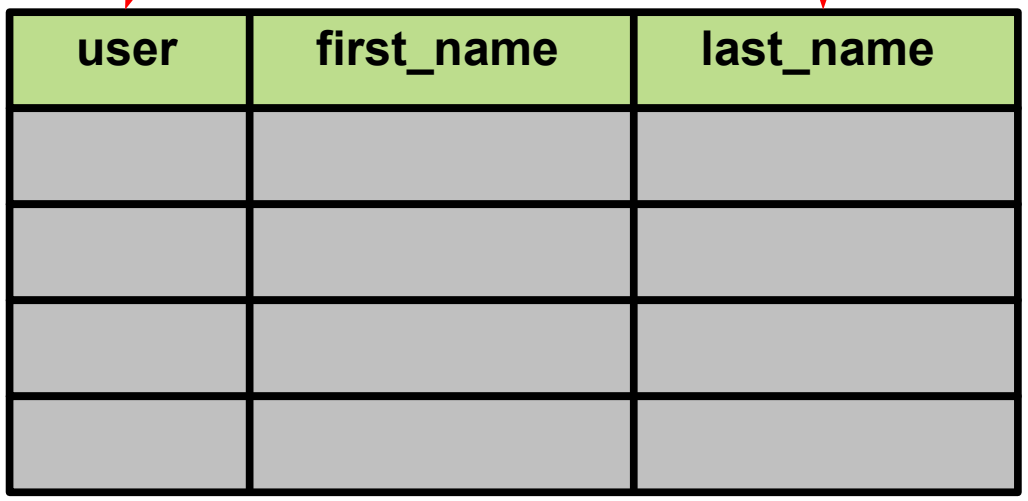
Use the "eval" method to run SQL

```
db eval {
    CREATE TABLE users(
        userid INTEGER,
        first_name VARCHAR(30),
        last_name VARCHAR(40)
    );
}
```

user	first_name	last_name

Semicolon separates multiple SQL statements.
Final semicolon is optional.

```
db eval {
    CREATE TABLE users(
      userid INTEGER,
      first_name VARCHAR(30),
      last_name VARCHAR(40)
          );
    }
```



```
db eval {
    CREATE TABLE users(
        userid INTEGER,
        first_name VARCHAR(30),
        last_name VARCHAR(40)
    );
}
```

user	first_name	last_name

Data types are ignored, mostly

Traditional SQL

- Rigid typing
- Types declared on containers
- Exceptions if type rules are violated

Traditional SQL

- Rigid typing
- Types declared on containers
- Exceptions if type rules are violated

- No types - everything is a string
- Internal dual representation
- Very flexible

Traditional SQL

- Rigid typing
- Types declared on containers
- Exceptions if type rules are violated

- No types - everything is a string
- Internal dual representation
- Very flexible

Traditional SQL

- Rigid typing
- Types declared on containers
- Exceptions if type rules are violated

- Type associated with values
- Containers have a “suggested type”
- All types accepted by every container

- No types - everything is a string
- Internal dual representation
- Very flexible

Values converted to integer if they can be. Otherwise stored as they are.

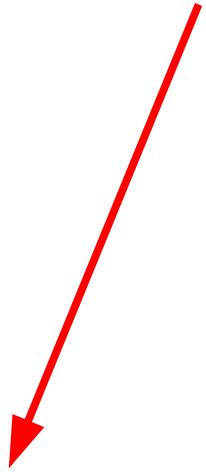
Values converted to strings. Length restrictions are ignored

```
db eval {
    CREATE TABLE users(
      userid INTEGER,
      first_name VARCHAR(30),
      last_name VARCHAR(40)
    );
}
```

user	first_name	last_name

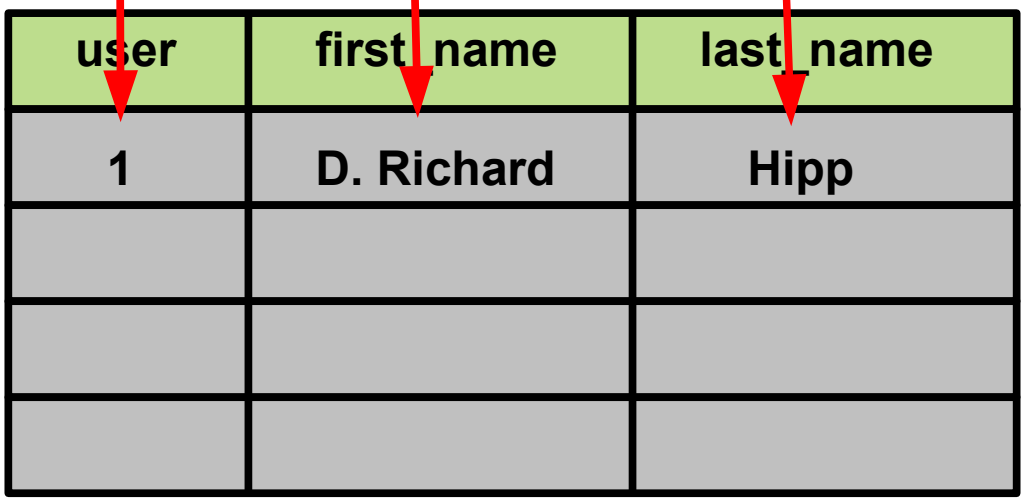
Additional information at http://www.sqlite.org/datatype3.html

Use an INSERT statement to add data



```
db eval {
        INSERT INTO users
        VALUES(1, 'D. Richard', 'Hipp')
}
```

```
db eval {
        INSERT INTO users
        VALUES(1, 'D. Richard', 'Hipp')
}
```



Use a SELECT statement to extract data from the database

```
db eval {
        SELECT user, first_name, last_name
          FROM users
}
1 {D. Richard} Hipp
```

```
db eval {
        SELECT user, first_name, last_name
          FROM users
}
1 {D. Richard} Hipp
```

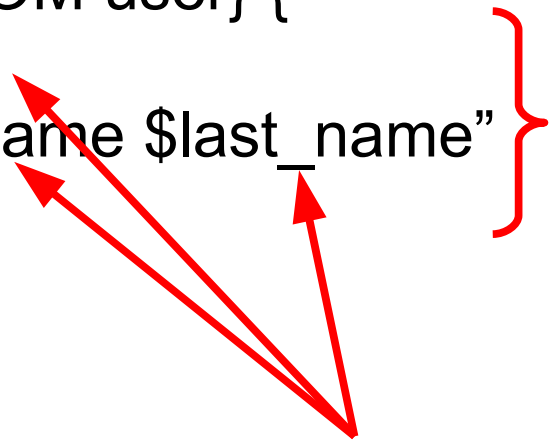
```
db eval {
        INSERT INTO users
        VALUES(2, 'Ginger', 'Wyrick')
}
```

user	first_name	last_name
1	D. Richard	Hipp
2	Ginger	Wyrick

```
db eval {
      SELECT * FROM users
}
1 {D. Richard} Hipp 2 Ginger Wyrick
```

Additional rows of data just make the returned list longer

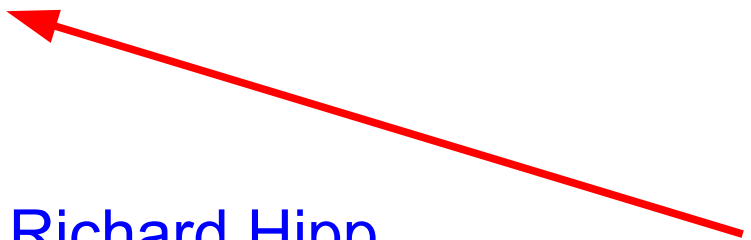
```
sqlite3 db database.db
db eval {SELECT * FROM user} {
    puts userid=$userid
    puts “name=$first_name $last_name”
}
userid=1
name=D. Richard Hipp
userid=2
name=Ginger Wyrick
```



Script runs once for each row in result set

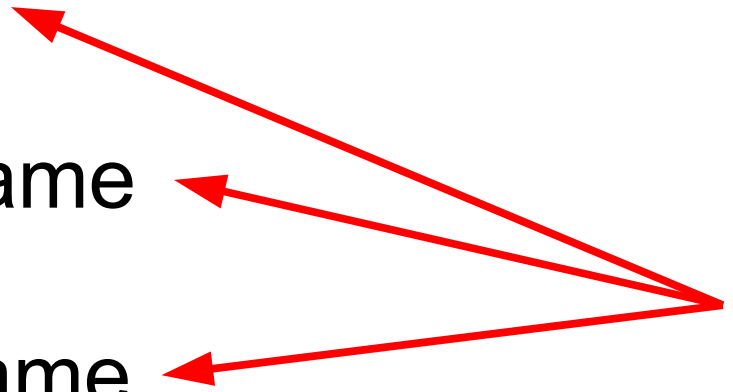
Column contents store in TCL variables

```
sqlite3 db database.db
db eval {SELECT * FROM user} {
    puts userid=$userid
    puts “name=$first_name $last_name”
    break
}
userid=1
name=D. Richard Hipp
```



“break” and “continue” work in the usual way

```
db eval {SELECT * FROM user} break
set userid
1
set first_name
D. Richard
set last_name
Hipp
```



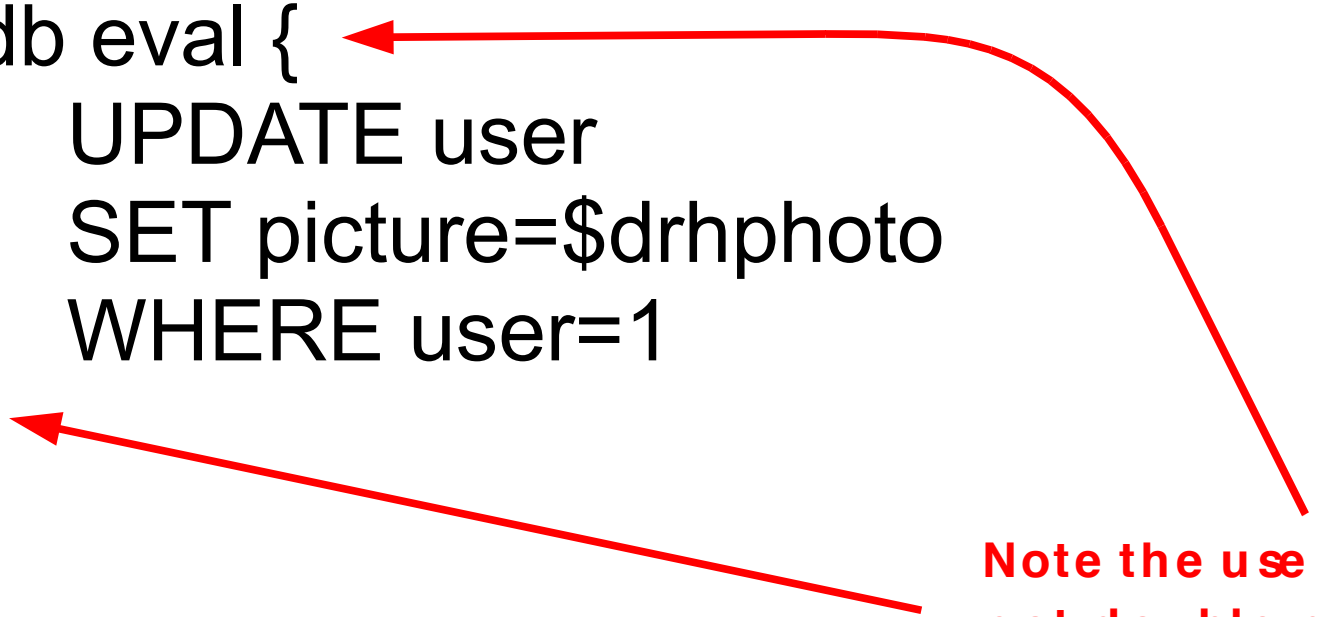
Variables persist after the last iteration of the loop

```
db eval {
   ALTER TABLE user
   ADD COLUMN picture;
}
```

user	first_name	last_name	picture
1	D. Richard	Hipp	

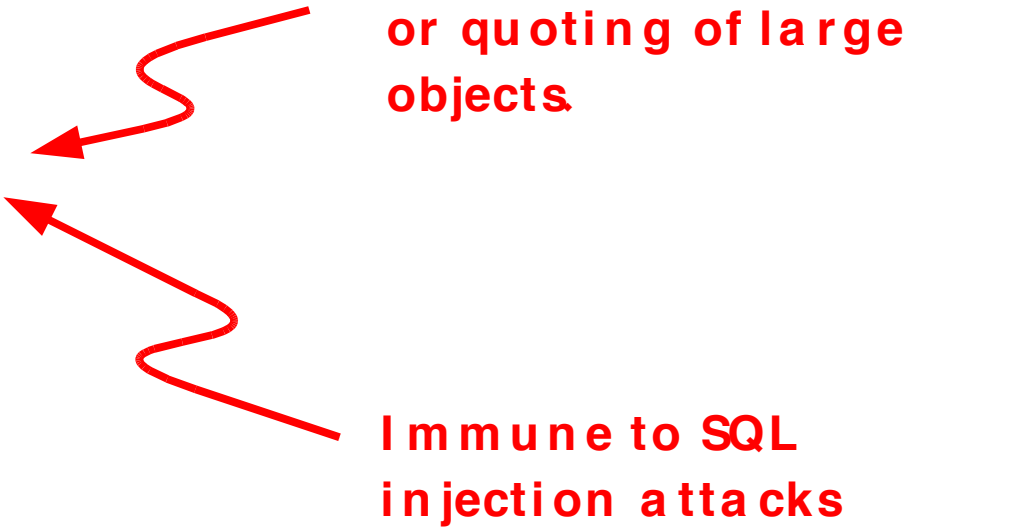
New Column Added

```
set in [open drh.jpg]
fconfigure $in -translation binary
set drhphoto [read $in]
close $in
db eval {
    UPDATE user
    SET picture=$drhphoto
    WHERE user=1
}
```



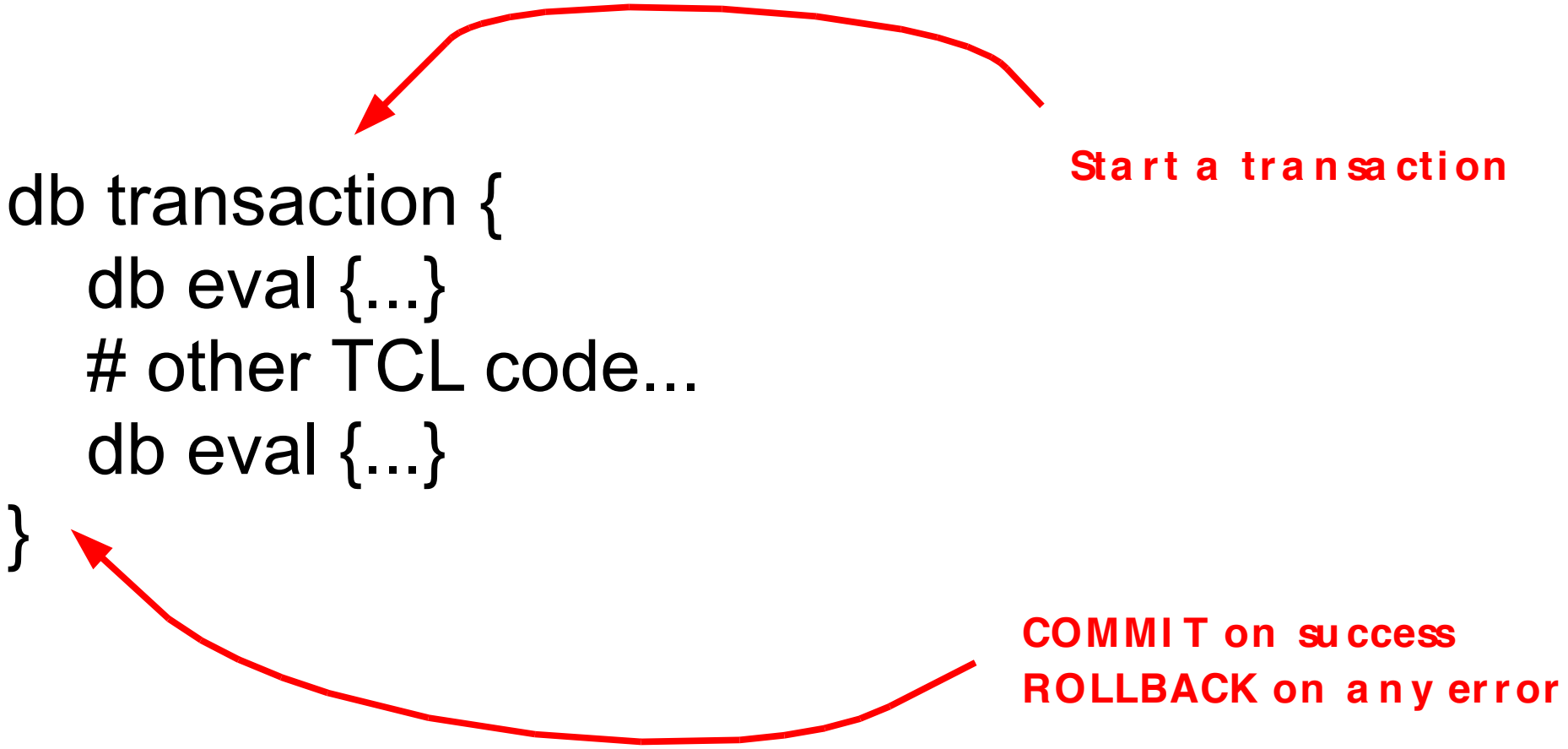
Note the use of curly-braces, not double-quotes

```
set in [open drh.jpg]
fconfigure $in -translation binary
set drhphoto [read $in]
close $in
db eval {
    UPDATE user
    SET picture=$drhphoto
    WHERE user=1
}
```



```
set in [open drh.jpg]
fconfigure $in -translation binary
set drhphoto [read $in]
close $in
db eval {
    UPDATE user
    SET picture=@drhphoto
    WHERE user=1
}
```

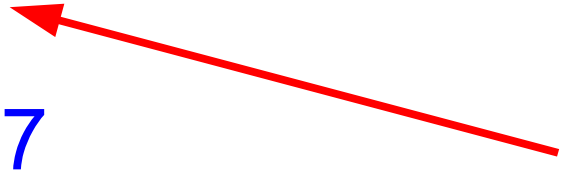
@ instead of $ to force the use of the ByteArray representation


db transaction {
db eval {...}
other TCL code...
db eval {...}
}
Start a transaction
COMMIT on success
ROLLBACK on any error

```
proc sqrtfunc {x} {
    return [expr {sqrt($x)}]
}
db function sqrt sqrtfunc
db eval {
   SELECT sqrt(id) FROM user
}
1.0 1.41421356237
```

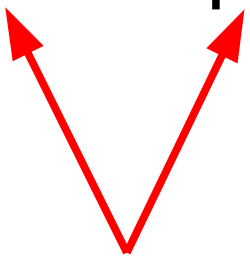
Define a new TCL function

Register the function with SQLite

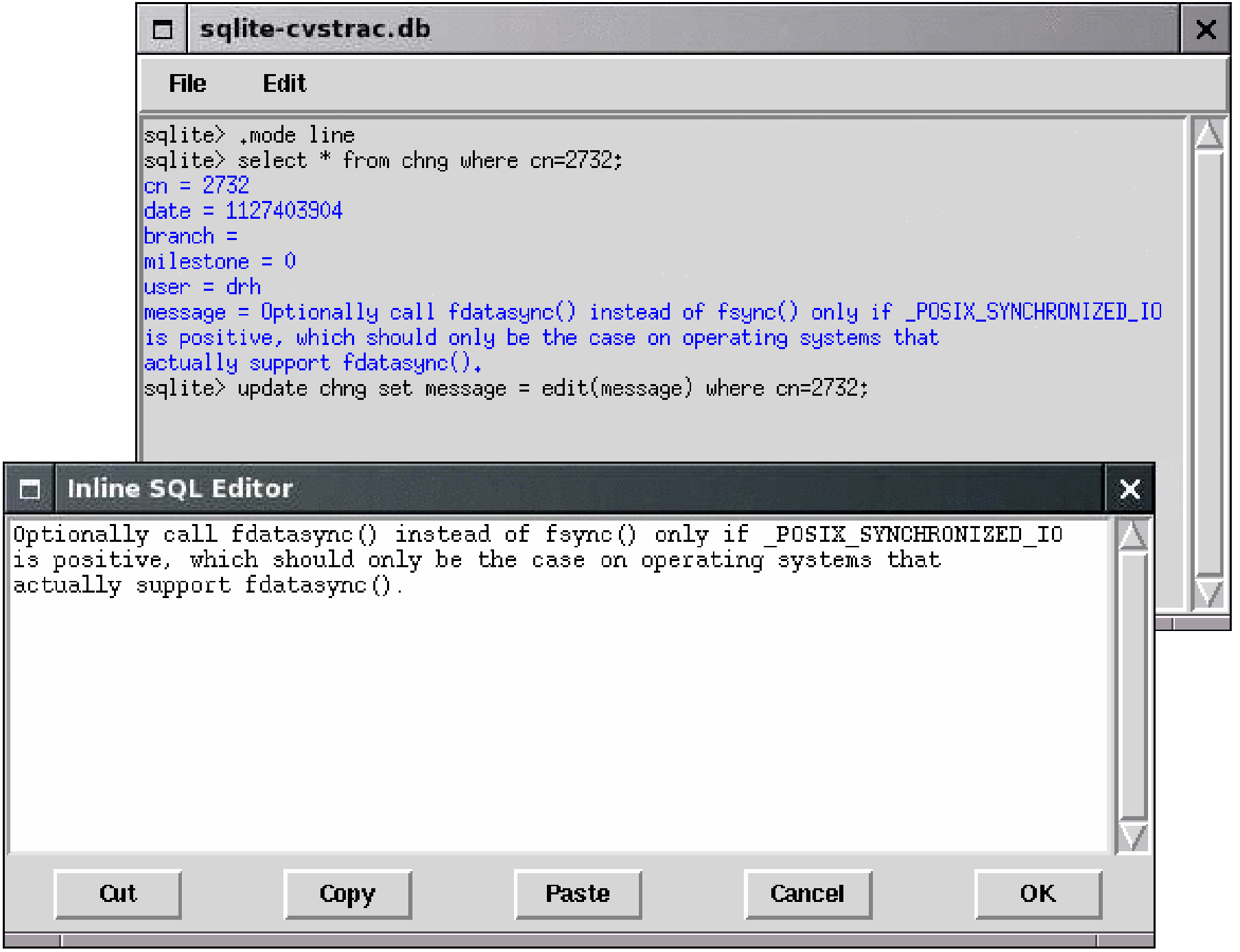


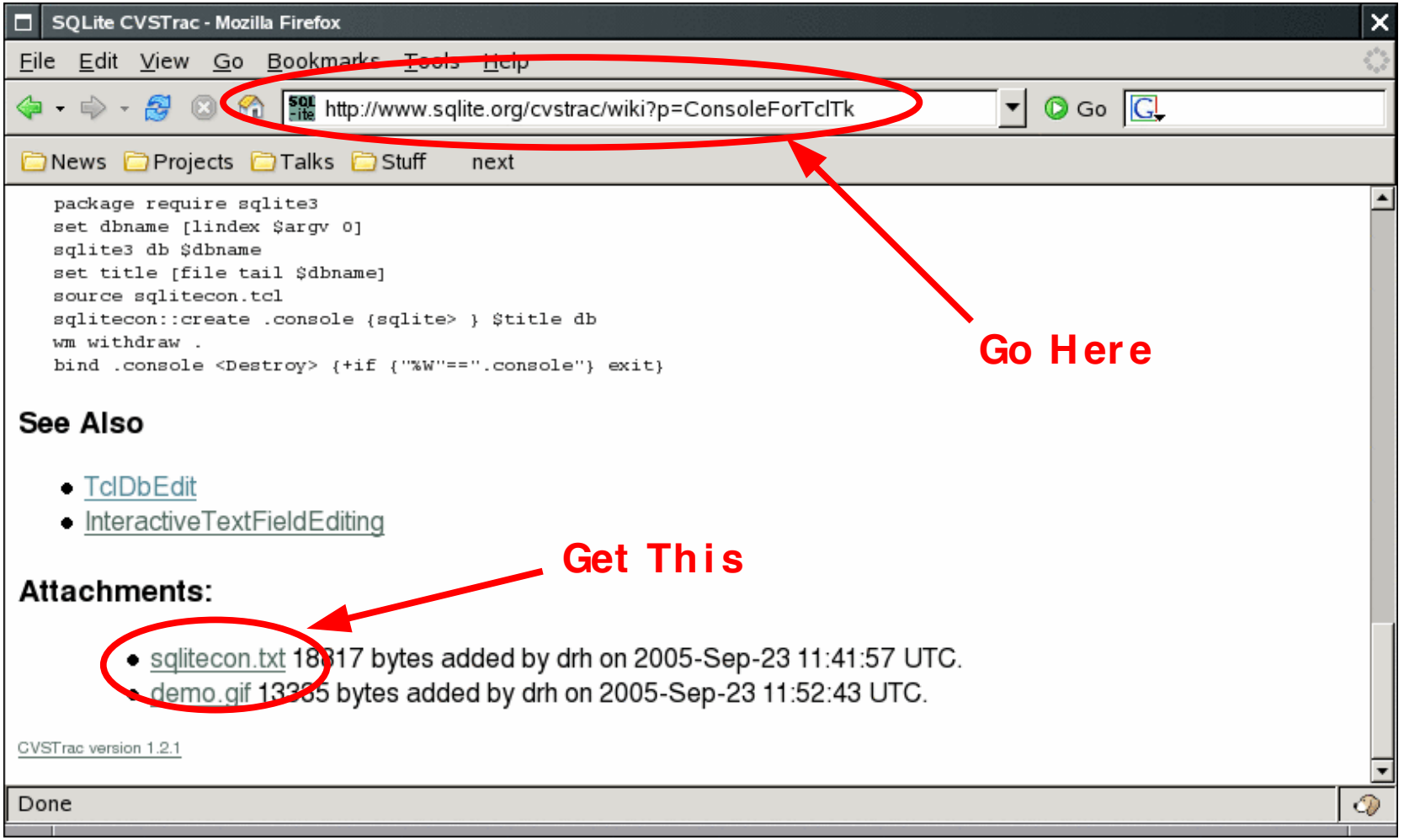
Use the TCL function in an SQLite query

```
proc sqlitecon::_edit {original_text} {
   # Code here to implement a GUI editor
   # for $original_text and return the result.
}
db function edit ::sqlite::_edit
```

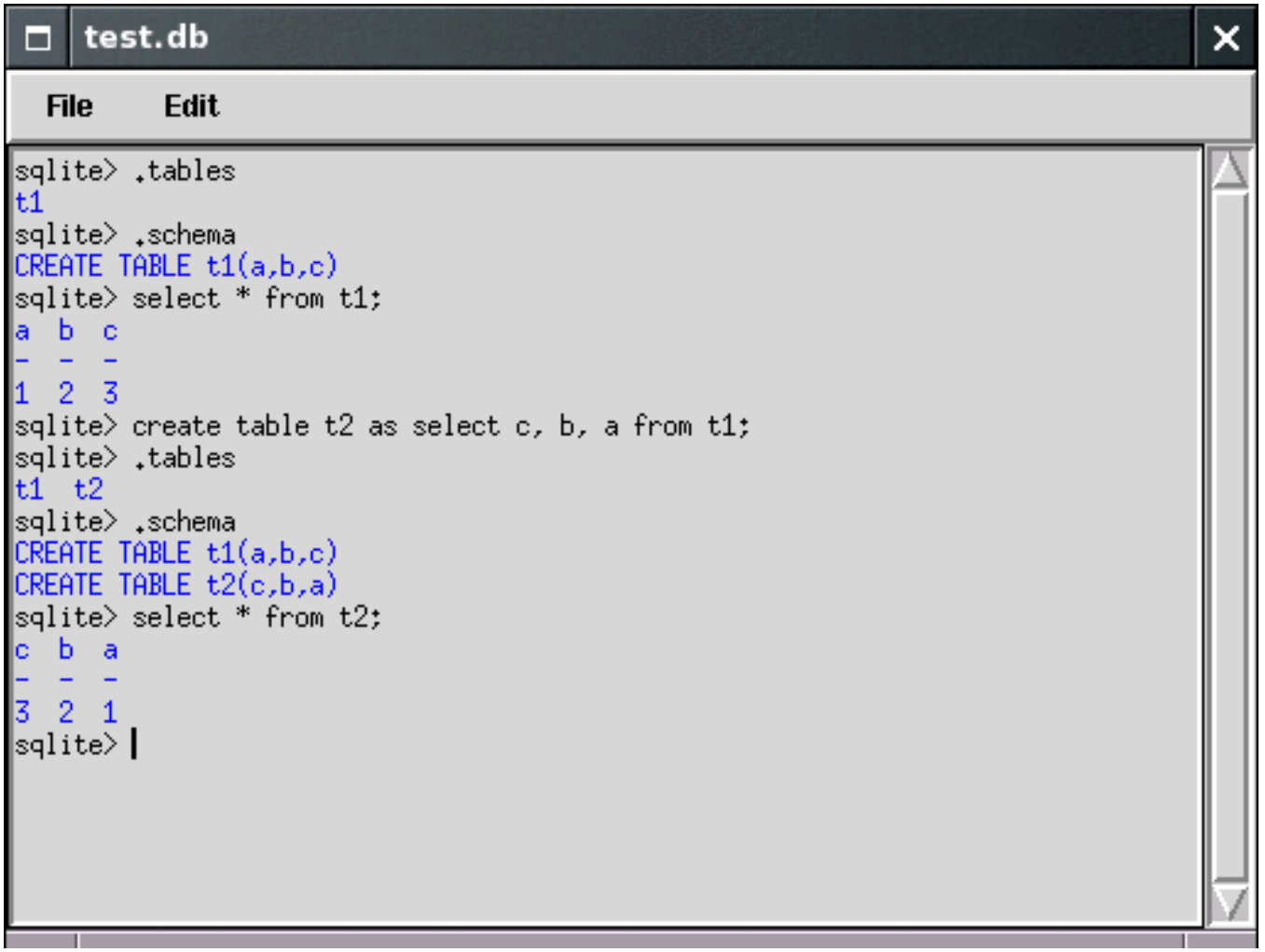


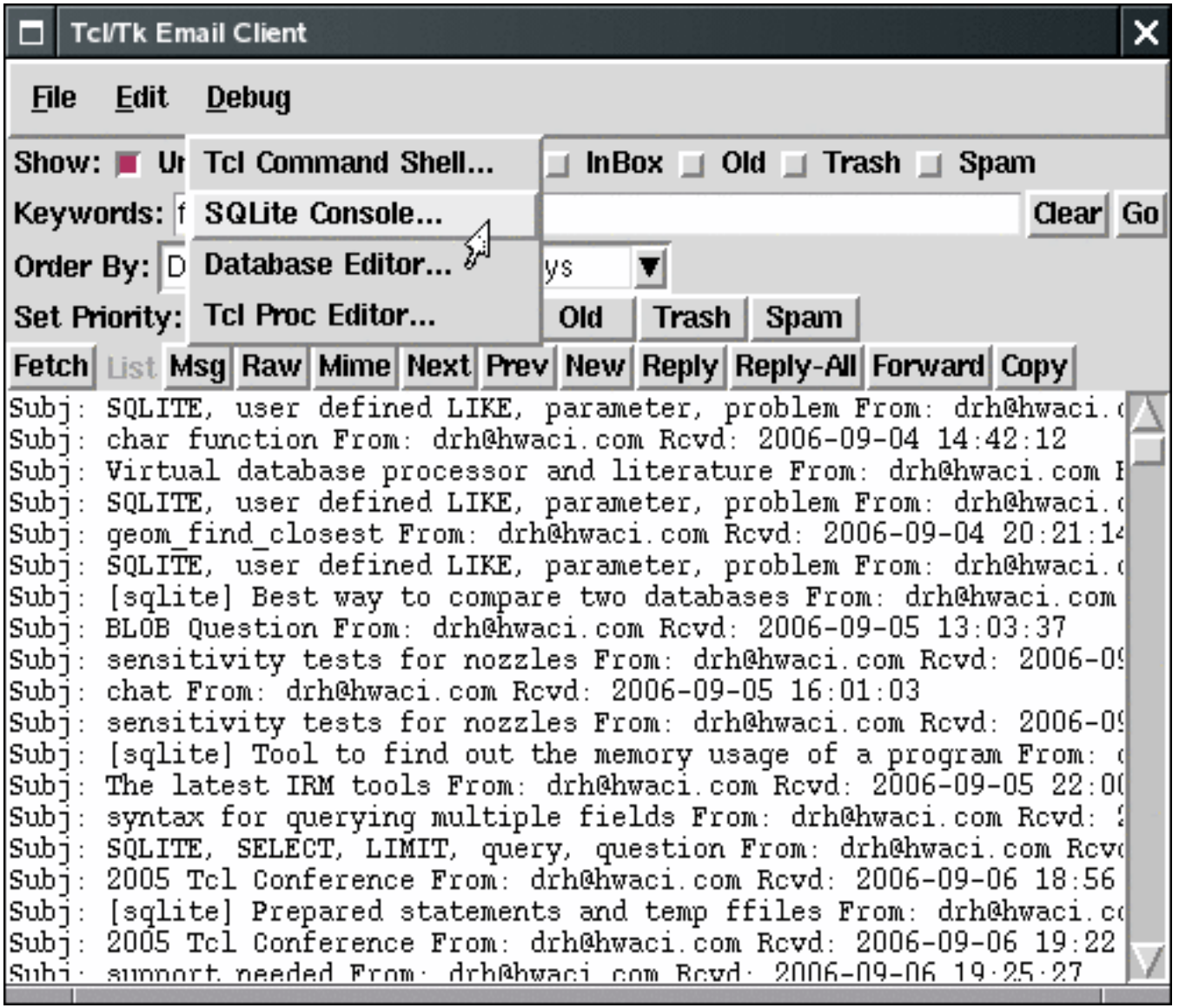
Create a new SQL function named "edit"
implemented by the TCL proc "::sqlite::_edit"

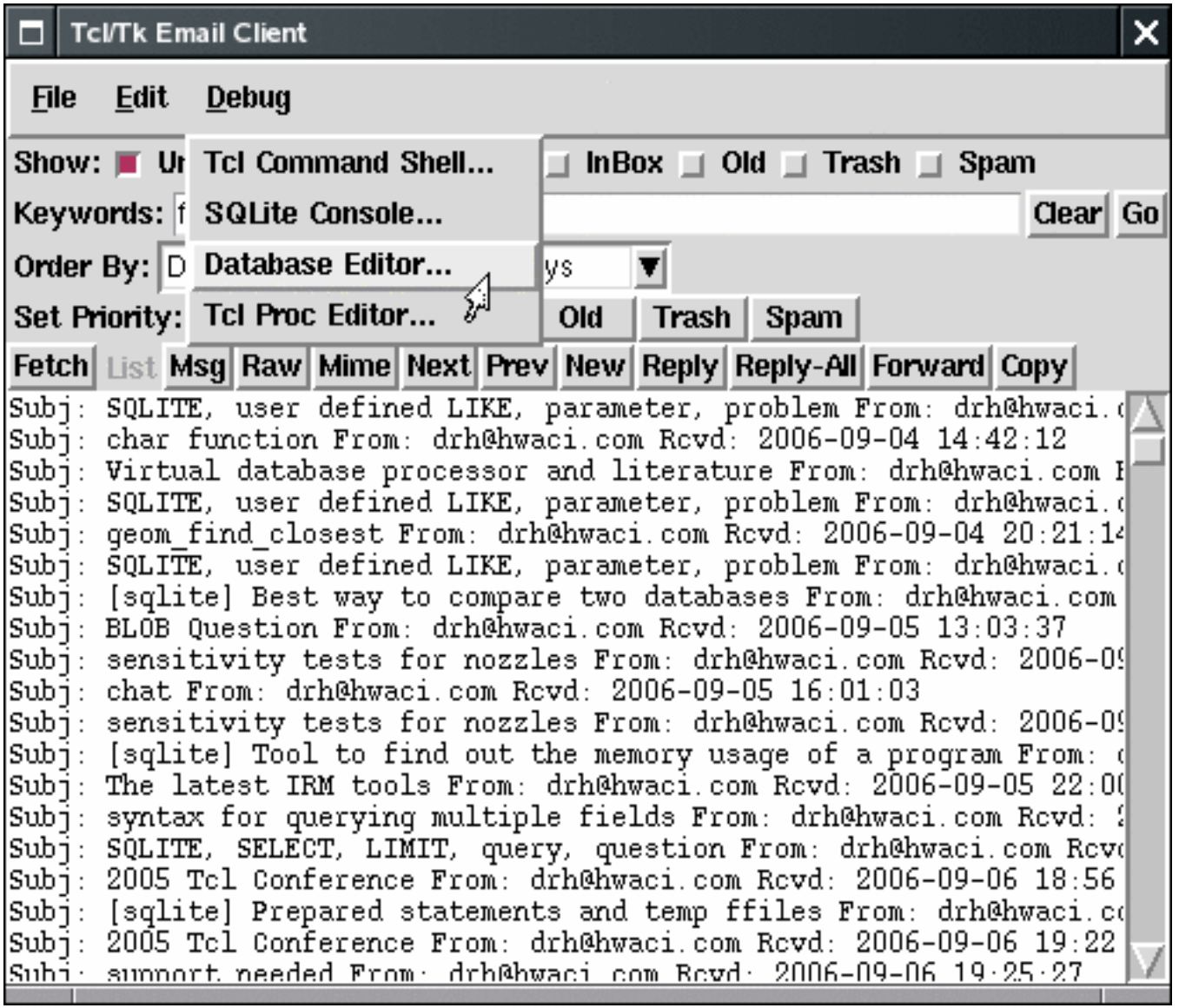

sqlite-cvstrac.db
File
Edit
sqlite> .mode line
sqlite> select * from chng where cn=2732;
cn = 2732
date = 1127403904
branch =
milestone = 0
user = drh
message = Optionally call fdatasync() instead of fsync() only if _POSIX_SYNCHRONIZED_IO
is positive, which should only be the case on operating systems that
actually support fdatasync().
sqlite> update chng set message = edit(message) where cn=2732;
Inline SQL Editor
Optionally call fdatasync() instead of fsync() only if _POSIX_SYNCHRONIZED_IO
is positive, which should only be the case on operating systems that
actually support fdatasync().
Cut
Copy
Paste
Cancel
OK

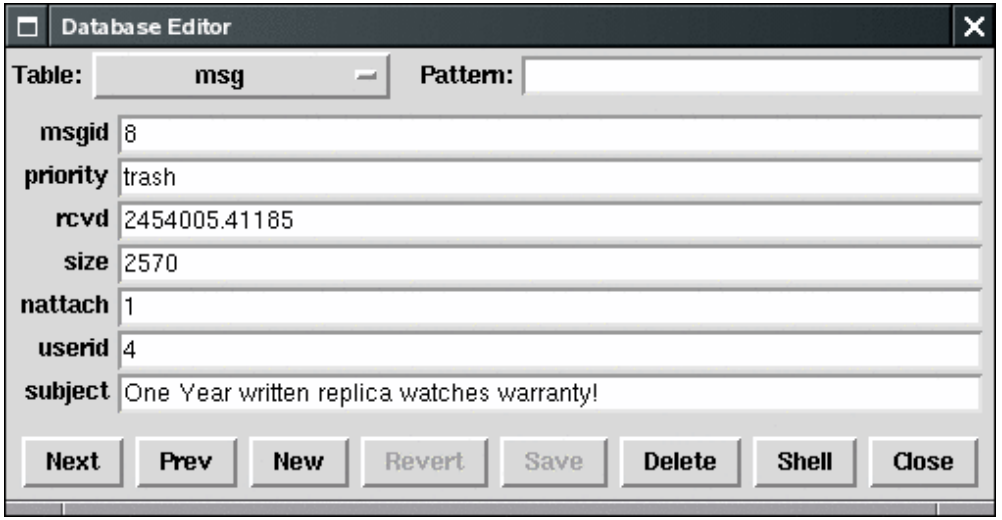

SQLite CVSTrac - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://www.sqlite.org/cvstrac/wiki?p=ConsoleForTclTk
Go
News Projects Talks Stuff next
package require sqlite3
set dbname [lindex $argv 0]
sqlite3 db $dbname
set title [file tail $dbname]
source sqlitecon.tcl
sqlitecon::create .console {sqlite> } $title db
wm withdraw .
bind .console <Destroy> {+if {"%W"==".console"} exit}
Go Here
See Also
TclDbEdit
InteractiveTextFieldEditing
Get This
Attachments:
sqlitecon.txt 18817 bytes added by drh on 2005-Sep-23 11:41:57 UTC.
demo.gif 13385 bytes added by drh on 2005-Sep-23 11:52:43 UTC.
CVSTrac version 1.2.1
Done

```
% package require Tk
% source sqlitecon.txt
% sqlitecon::create .console {sqlite> } test.db db
%
```



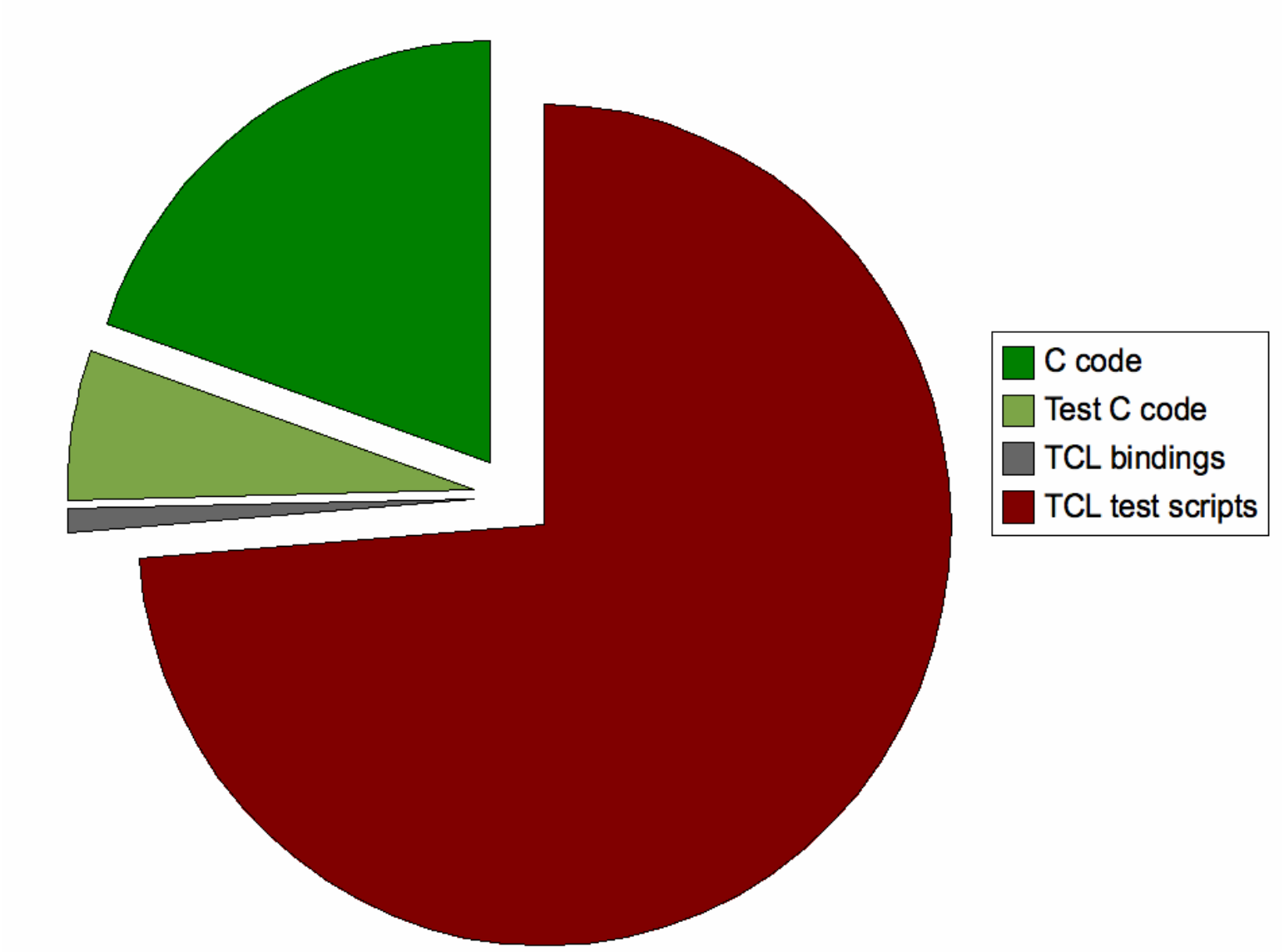

Tcl/Tk Email Client
File
Edit
Debug
Show:
Ur
Tcl Command Shell...
SQLite Console...
Database Editor...
Tcl Proc Editor...
InBox
Old
Trash
Spam
Keywords:
Clear
Go
Order By:
ys
Set Priority:
Old
Trash
Spam
Fetch
List
Msg
Raw
Mime
Next
Prev
New
Reply
Reply-All
Forward
Copy
Subj: SQLITE, user defined LIKE, parameter, problem From: drh@hwaci.
Subj: char function From: drh@hwaci.com Rcvd: 2006-09-04 14:42:12
Subj: Virtual database processor and literature From: drh@hwaci.com
Subj: SQLITE, user defined LIKE, parameter, problem From: drh@hwaci.
Subj: geom_find_closest From: drh@hwaci.com Rcvd: 2006-09-04 20:21:1
Subj: SQLITE, user defined LIKE, parameter, problem From: drh@hwaci.
Subj: [sqlite] Best way to compare two databases From: drh@hwaci.com
Subj: BLOB Question From: drh@hwaci.com Rcvd: 2006-09-05 13:03:37
Subj: sensitivity tests for nozzles From: drh@hwaci.com Rcvd: 2006-0
Subj: chat From: drh@hwaci.com Rcvd: 2006-09-05 16:01:03
Subj: sensitivity tests for nozzles From: drh@hwaci.com Rcvd: 2006-0
Subj: [sqlite] Tool to find out the memory usage of a program From:
Subj: The latest IRM tools From: drh@hwaci.com Rcvd: 2006-09-05 22:0
Subj: syntax for querying multiple fields From: drh@hwaci.com Rcvd:
Subj: SQLITE, SELECT, LIMIT, query, question From: drh@hwaci.com Rcv
Subj: 2005 Tcl Conference From: drh@hwaci.com Rcvd: 2006-09-06 18:56
Subj: [sqlite] Prepared statements and temp ffiles From: drh@hwaci.c
Subj: 2005 Tcl Conference From: drh@hwaci.com Rcvd: 2006-09-06 19:22
Subj: support needed From: drh@hwaci.com Rcvd: 2006-09-06 19:25:27


Tcl/Tk Email Client
File
Edit
Debug
Tcl Command Shell...
SQLite Console...
Database Editor...
Tcl Proc Editor...
Show:
InBox
Old
Trash
Spam
Keywords:
Clear
Go
Order By:
Set Priority:
Old
Trash
Spam
Fetch
List
Msg
Raw
Mime
Next
Prev
New
Reply
Reply-All
Forward
Copy
Subj: SQLITE, user defined LIKE, parameter, problem From: drh@hwaci.
Subj: char function From: drh@hwaci.com Rcvd: 2006-09-04 14:42:12
Subj: Virtual database processor and literature From: drh@hwaci.com
Subj: SQLITE, user defined LIKE, parameter, problem From: drh@hwaci.
Subj: geom_find_closest From: drh@hwaci.com Rcvd: 2006-09-04 20:21:1
Subj: SQLITE, user defined LIKE, parameter, problem From: drh@hwaci.
Subj: [sqlite] Best way to compare two databases From: drh@hwaci.com
Subj: BLOB Question From: drh@hwaci.com Rcvd: 2006-09-05 13:03:37
Subj: sensitivity tests for nozzles From: drh@hwaci.com Rcvd: 2006-0
Subj: chat From: drh@hwaci.com Rcvd: 2006-09-05 16:01:03
Subj: sensitivity tests for nozzles From: drh@hwaci.com Rcvd: 2006-0
Subj: [sqlite] Tool to find out the memory usage of a program From:
Subj: The latest IRM tools From: drh@hwaci.com Rcvd: 2006-09-05 22:0
Subj: syntax for querying multiple fields From: drh@hwaci.com Rcvd:
Subj: SQLITE, SELECT, LIMIT, query, question From: drh@hwaci.com Rcv
Subj: 2005 Tcl Conference From: drh@hwaci.com Rcvd: 2006-09-06 18:56
Subj: [sqlite] Prepared statements and temp ffiles From: drh@hwaci.c
Subj: 2005 Tcl Conference From: drh@hwaci.com Rcvd: 2006-09-06 19:22
Subj: support needed From: drh@hwaci.com Rcvd: 2006-09-06 19:25:27


Database Editor
Table:
msg
Pattern:
msgid
8
priority
trash
rcvd
2454005.41185
size
2570
nattach
1
userid
4
subject
One Year written replica watches warranty!
Next
Prev
New
Revert
Save
Delete
Shell
Close

Are you still not convinced
that SQLite is a TCL extension?

SQLite Written Mostly In TCL

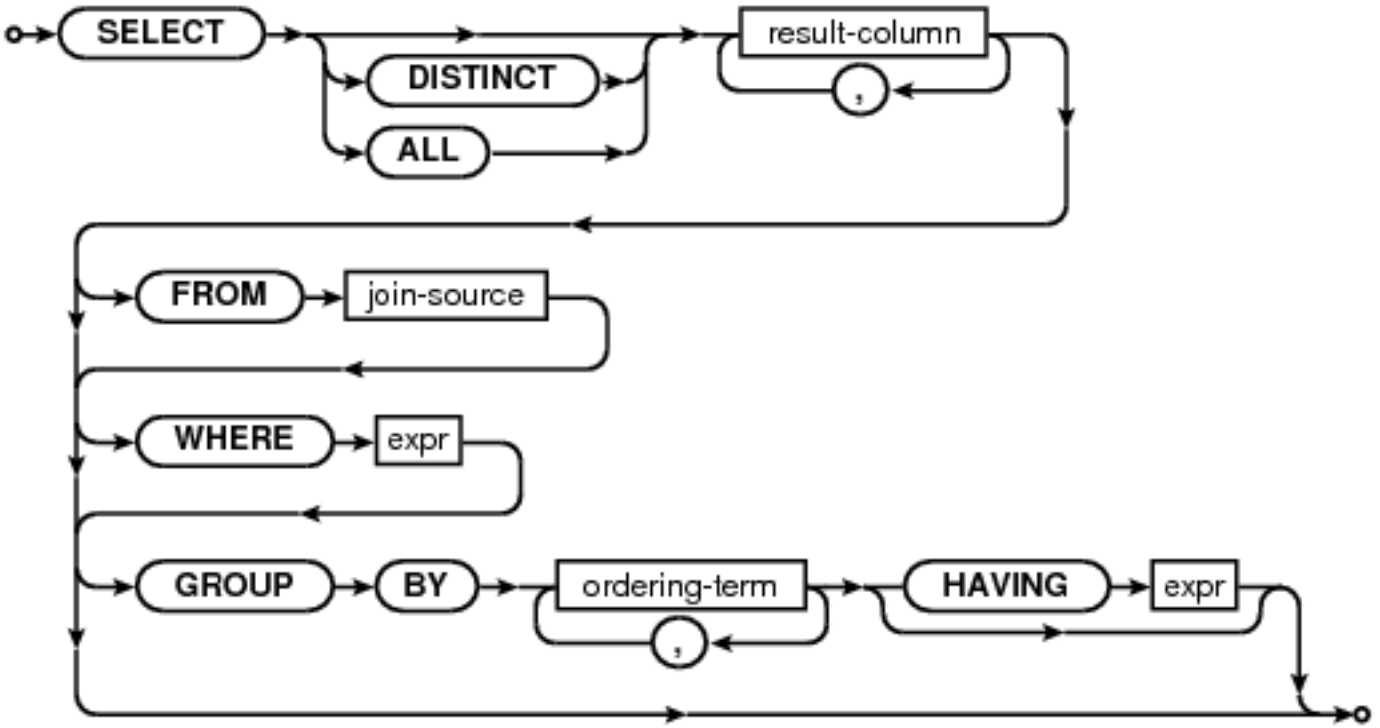


TCL must be installed on the development system in order to build SQLite.

TCL is required to test SQLite

All SQLite documentation is generated by TCL scripts.

```
select-core {
   stack
     {line SELECT {or nil DISTINCT ALL} {loop result-column ,}}
     {optx FROM join-source}
     {optx WHERE expr}
     {optx GROUP BY {loop ordering-term ,} {optx HAVING expr}}
}
```



See http://wiki.tcl.tk/21708 for additional information

“SQLite would not exist without TCL”

SQLite is “different”

is

“zero administration”

Database Administrators


iPod
Music
Photos
Extras
Settings
Shuffle Songs
MENU

ORACLE®

is to

SQLite

as

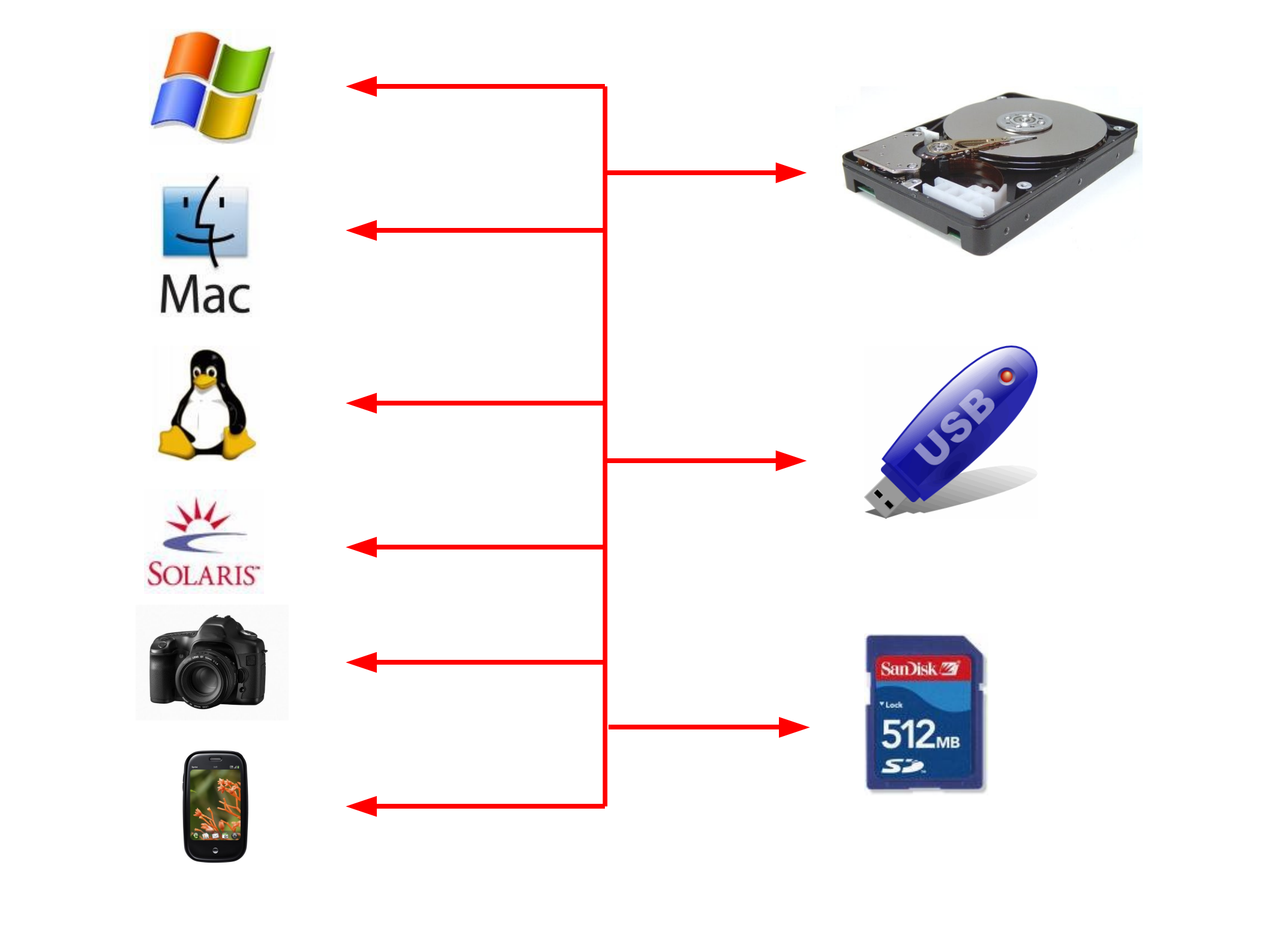
is to

- SQLite does not compete with Oracle

- SQLite does not compete with Oracle
- SQLite competes with **fopen()**

Portable File Format

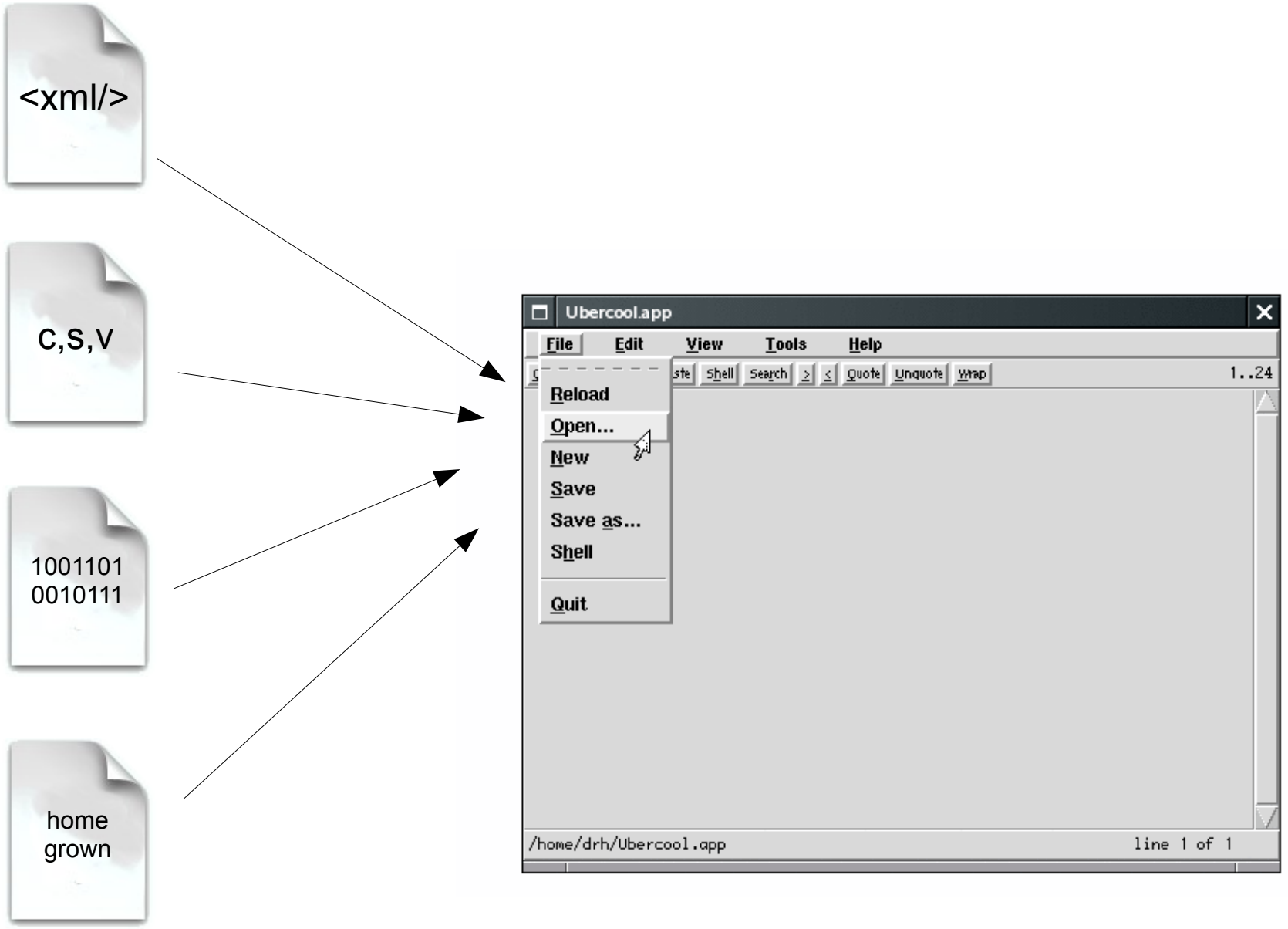
- A database is a single ordinary disk file
- No special naming conventions or required file suffixes
- Cross-platform: big/little-endian and 32/64-bit
- Backwards compatible through 3.0.0
- Promise to keep it compatible moving forward
- Not tied to any particular programming language.

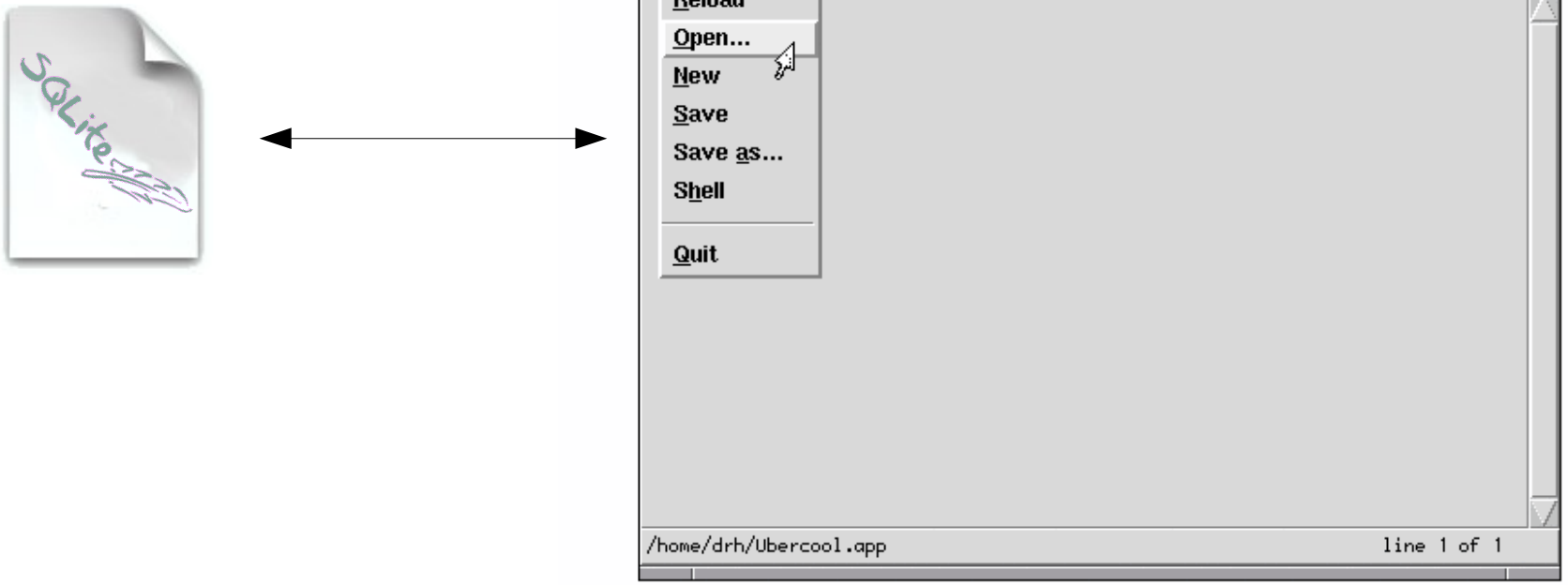

Mac
SOLARIS
USB
SanDisk
512MB
SD

Zero-Administration & Portable File Format

means SQLite makes a great

Application File Format


<xml/>
C,S,V
1001101
0010111
home
grown
Ubercool.app
File
Edit
View
Tools
Help
Shell
Search
Quote
Unquote
Wrap
1..24
Reload
Open...
New
Save
Save as...
Shell
Quit
/home/drh/Ubercool.app
line 1 of 1


SQLite
Ubercool.app
File
Edit
View
Tools
Help
Shell
Search
Quote
Unquote
Wrap
1..24
Reload
Open...
New
Save
Save as...
Shell
Quit
/home/drh/Ubercool.app
line 1 of 1

SQLite as file format freebies

- No parsing and generating code to write
- Atomic updates
- Fast, built-in searching
- Access via third-party tools
- Simplified upgrade migration
- Cross-platform file format
- High-level query language

Small Footprint

gcc -Os -DSQLITE_THREADSAFE=0

293 KiB

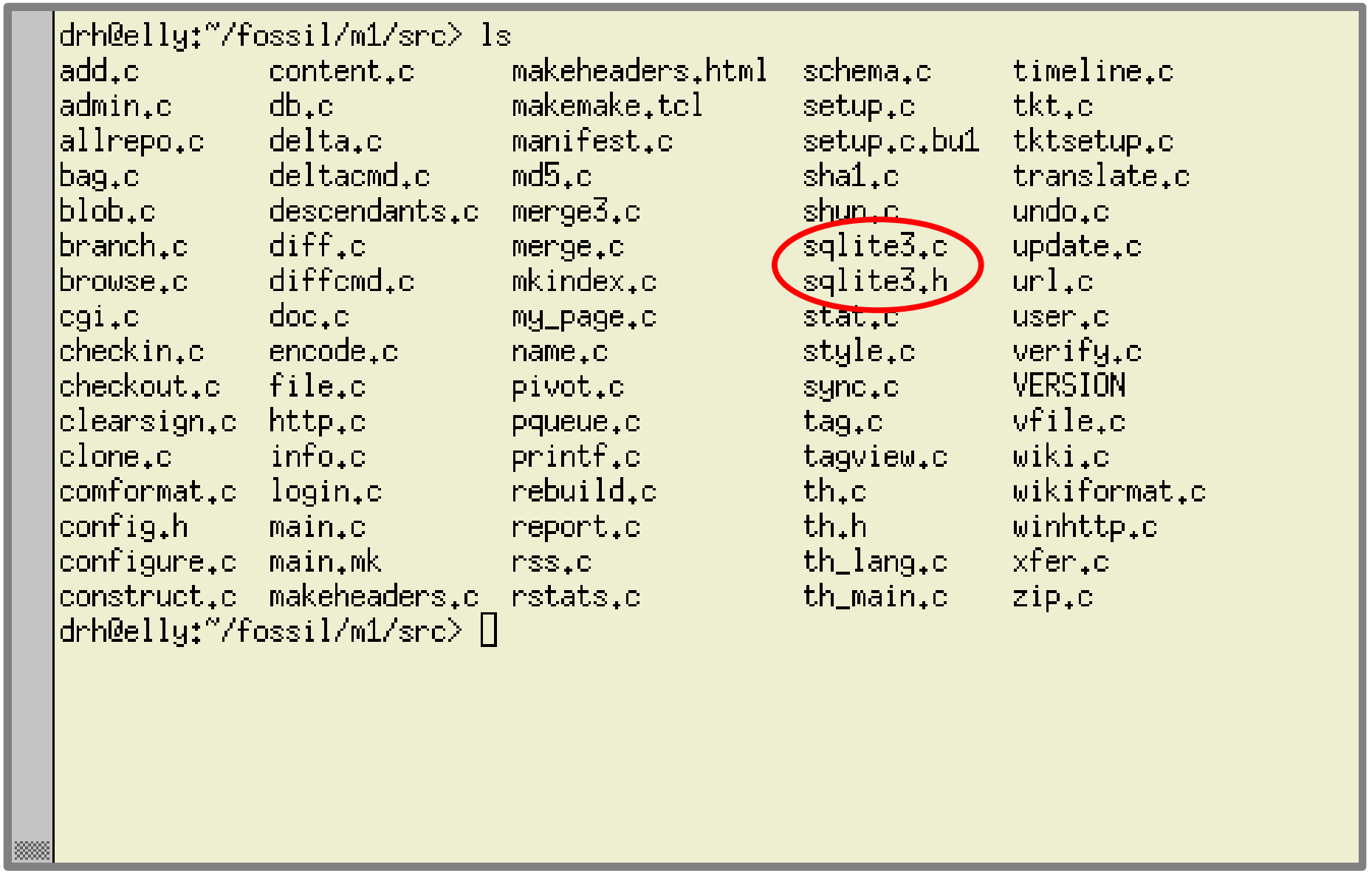
gcc -O3 -DSQLITE_ENABLE_FTS3=1 -DSQLITE_ENABLE_RTREE=1

845 KiB

Sizes Include TCL language bindings
Values current as of 2009-09-25

Single Source Code File

- The “amalgamation” source code file: **sqlite3.c** or **tclsqlite3.c**
- About 68,000 lines of ANSI C code
- 3.9 MB
- Few dependencies: libc and libtcl
- Very simple to add to a larger C program
- Very simple to build as a tclsh loadable library

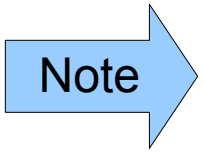

drh@elly:~/fossil/m1/src> ls
add.c content.c makeheaders.html schema.c timeline.c
admin.c db.c makemake.tcl setup.c tkt.c
allrepo.c delta.c manifest.c setup.c.bu1 tktsetup.c
bag.c deltacmd.c md5.c sha1.c translate.c
blob.c descendants.c merge3.c shun.c undo.c
branch.c diff.c merge.c sqlite3.c update.c
browse.c diffcmd.c mkindex.c sqlite3.h url.c
cgi.c doc.c my_page.c stat.c user.c
checkin.c encode.c name.c style.c verify.c
checkout.c file.c pivot.c sync.c VERSION
clearsign.c http.c pqueue.c tag.c vfile.c
clone.c info.c printf.c tagview.c wiki.c
comformat.c login.c rebuild.c th.c wikiformat.c
config.h main.c report.c th.h winhttp.c
configure.c main.mk rss.c th_lang.c xfer.c
construct.c makeheaders.c rstats.c th_main.c zip.c
drh@elly:~/fossil/m1/src>

The “generic” folder in the TEA distribution of SQLite contains exactly one file:

tclsqlite3.c

Building a tclsh loadable library

```
bash> gcc -shared tclsqlite3.c -o tclsqlite3.so
bash> tclsh
% load ./tclsqlite3.so
% sqlite3 db :memory:
% db eval {SELECT sqlite_version()}
3.6.19
```



Add -DSQLITE_THREADSAFE=0 for non-threadsafe tclsh

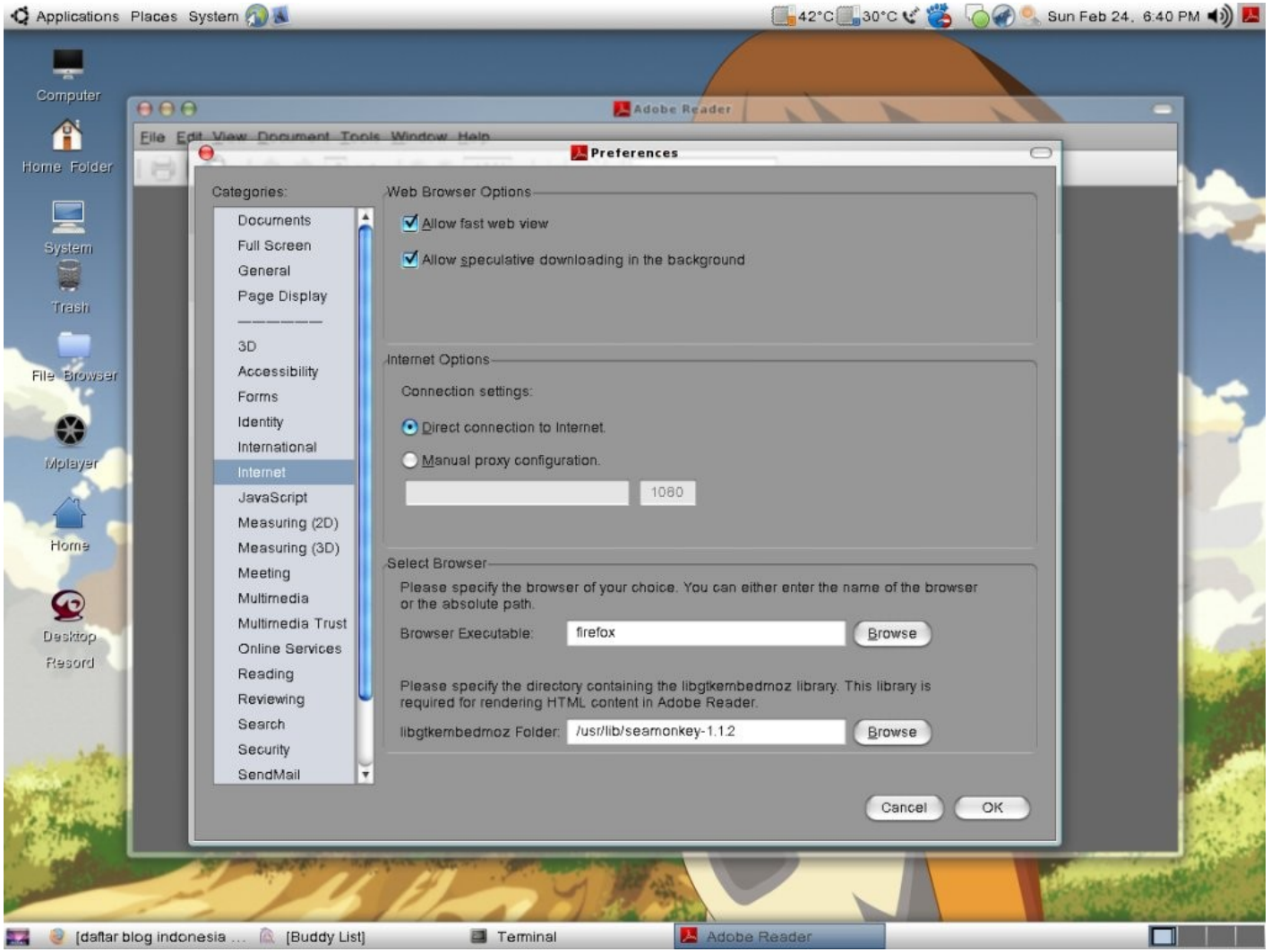
Other Features Of SQLite

- Gigibyte size BLOBs and strings
- Tebibyte size databases
- 100% branch test coverage
- Nested transactions
- Full text search
- R-Trees
- ATTACH DATABASE
- Robust against power loss, malloc() failures, and I/O errors.
- Referential integrity

Many companies and organizations use SQLite...

Adobe Photoshop Lightroom

Adobe Reader



Mozilla Firefox

Symbian/Nokia

Google Android

iPhone


Pizza
Frank's Pizza
http://www.nytimes.com
The New York Times
AT&T
9:42 AM
John Appleseed
calling...
End Call

iPod & iTunes


iTunes
Albums
Artists
Genres
Composers
Music
Movies
TV Shows
Podcasts
Applications
Radio
iTunes Store
Purchased
Steve's iPhone
Party Shuffle
Both Sides of the Gun
Ben Harper
Coco
Colbie Caillat
Datarock
Detours
Sheryl Crow
Eyes – Single
Rogue Wave
Flavors of Entanglement
Alanis Morissette
Fly Between Falls
ALO (Animal Liberation...
Gentlemen
The Afghan Whips
BECK
MODERN GUILT
Sleep Through the Static
Jack Johnson
Superhero Brother
G. Love & Special Sauce
DATAROCK

iStuff

Blackberry

Palm webOS

Skype

Sony Playstation

... and so forth


Route
Santa Monica Blvd
Santa Monica Fwy
10
Lincoln Blvd
Menu
1000 m
2000 m
Mio


SONY
HD
AVCHD
HANDYCAM
10.2 MEGA PIXELS
HYBRID
HDD

Open Source


yum
yellowdog updater modified


amarok


ActiveTcl


python™


php


SUBVERSION


trac
Integrated SCM & Project Management


RAILS


FOSSIL


monotone

Have you looked at SQLite lately...

- Faster
- CHECK constraints
- SAVEPOINT and nested transactions
- Enhanced query planner
- 100% branch test coverage
- Recursive triggers
- FOREIGN KEY constraints
- Sources managed using Fossil

Closing Thoughts

Shouldn't you be using SQLite instead of [open]?

Why isn't tclsqlite3.c part of the TCL core?

Can we get a TCL amalgamation?


tcl/tk
SQLite