

A SQL Approach to Exploring ELF Objects

SCaLE 2024

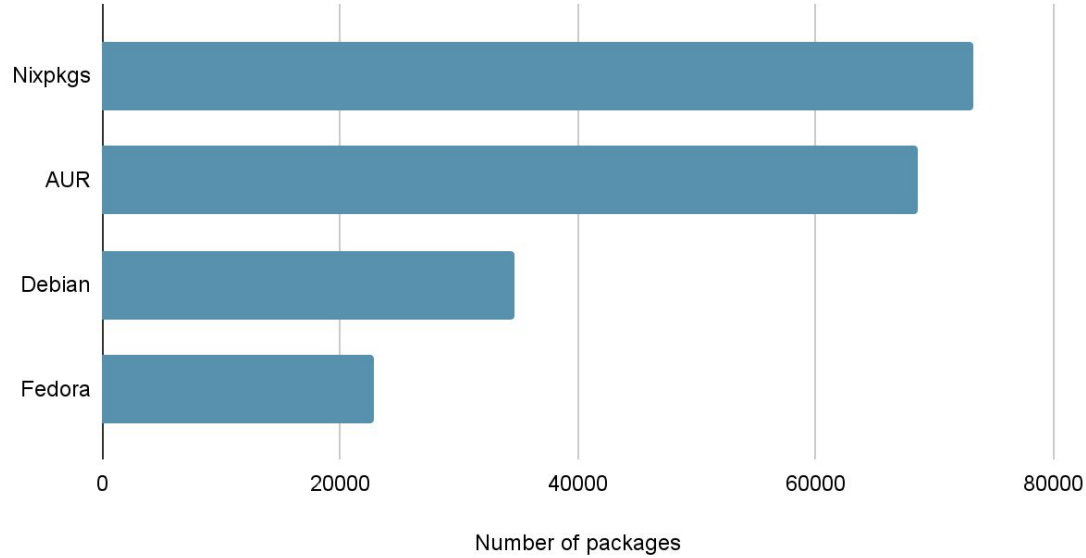
Farid Zakaria <fmzakari@ucsc.edu>

Who am I?

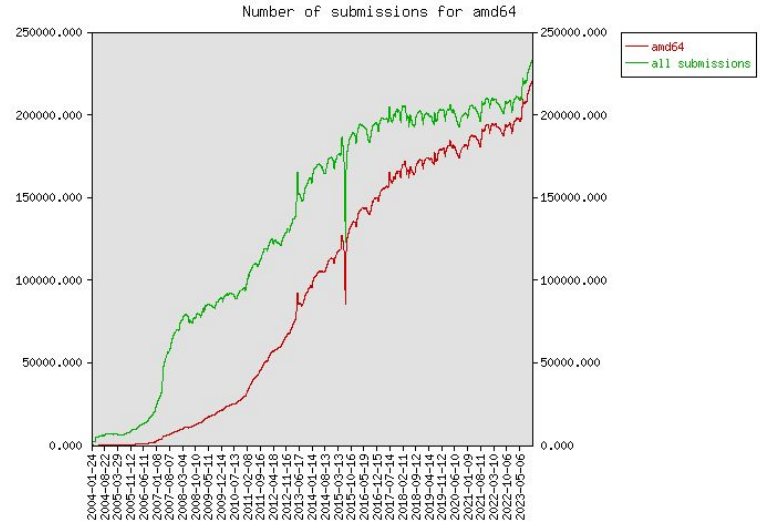
- Farid Zakaria
 - Father of 3 *young* boys
 - Working out of Santa Cruz, California
- PhD (Computer Science) Student at University of California Santa Cruz
 - Advised by Dr. Andrew Quinn
- Employed by Google
 - Currently working on ML Compilers but previously spent ~4 years in the BI space
- I'm also at SCaLE for #NixCon 2024



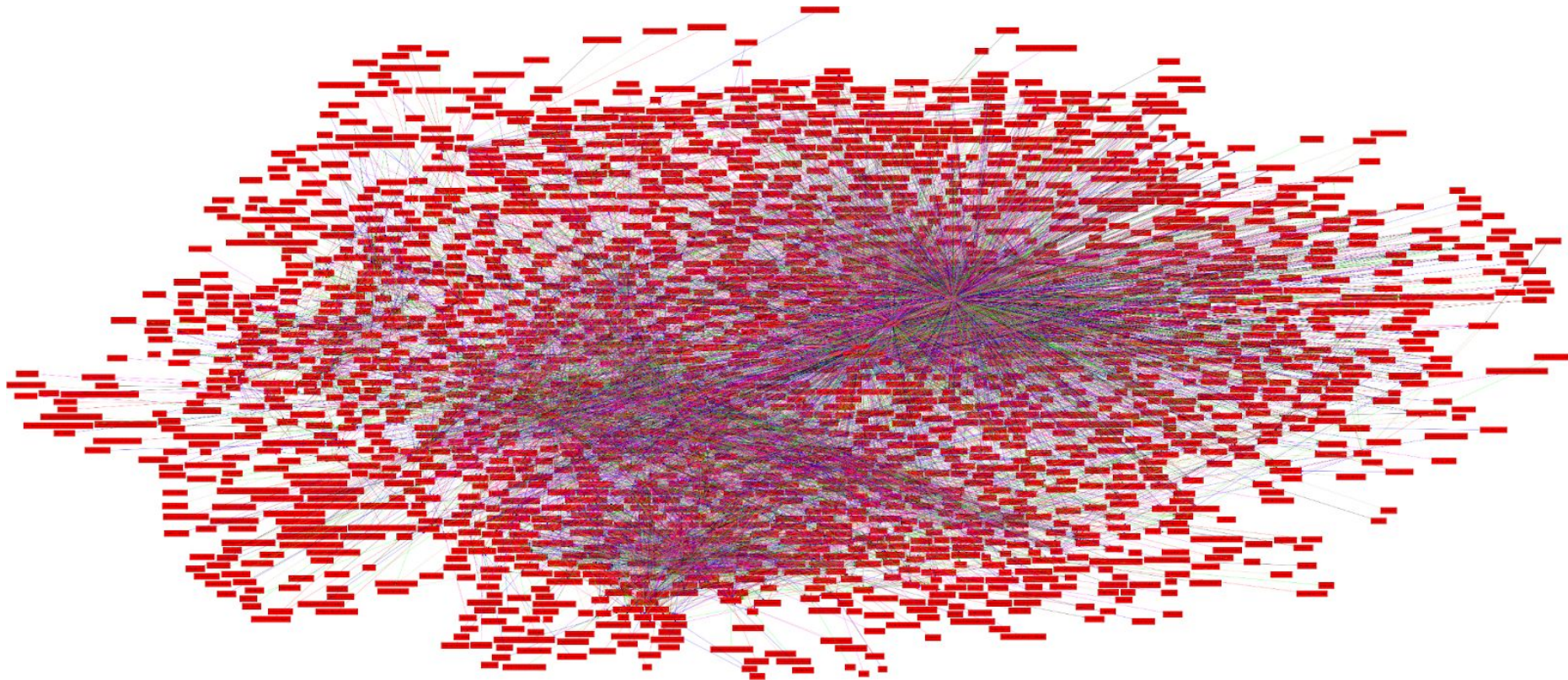
Linux Distributions



<https://repology.org/repositories/statistics/total>

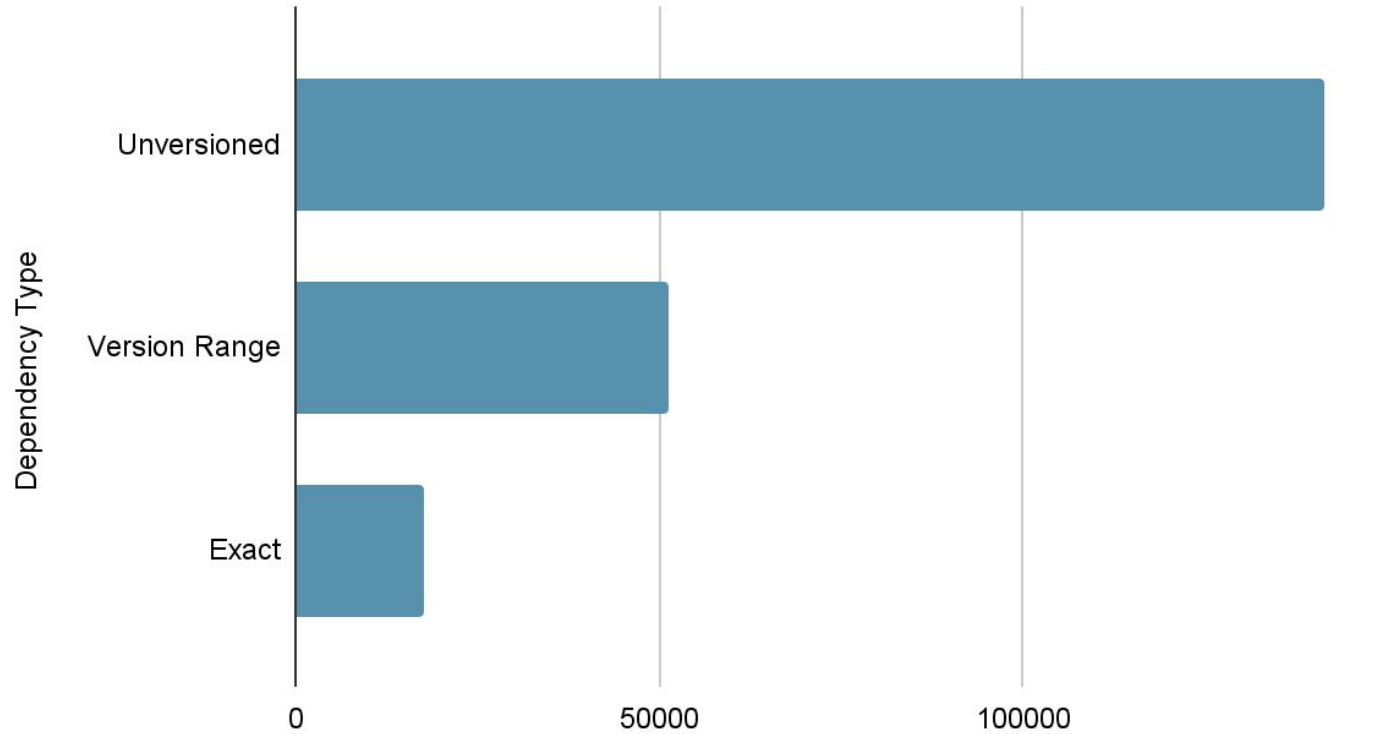


<https://popcon.debian.org/>



```
nix-store --query --include-outputs --graph $(nix-store --query --deriver $(which ruby)) >  
/tmp/ruby_full_closure.dot  
sfdp -x -Goverlap=prism -Tpng /tmp/ruby_full_closure.dot > output.png
```

Debian package dependencies by type



Diversity

Complexity

Specificity

One of the fundamental data management units within a Linux system are the shared object files that are loaded into memory by dynamically linked processes at startup.

Shared libraries

The cause of, and solution to, all of life's (security, maintenance and packaging) problems.

- Tom Scogland

Shared libraries are not a good thing in general. They add a lot of overhead in this case, but more importantly they also add lots of unnecessary dependencies and complexity, and almost no shared libraries are actually version-safe, so it adds absolutely zero upside.

...

I think people have this incorrect picture that "shared libraries are inherently good". They really really aren't. They cause a lot of problems, and the advantage really should always be weighed against those (big) disadvantages.

...

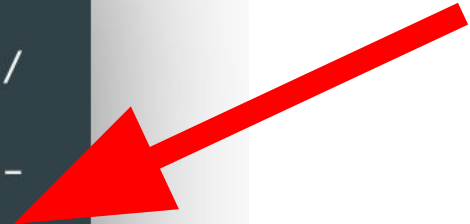
Linus Torvalds

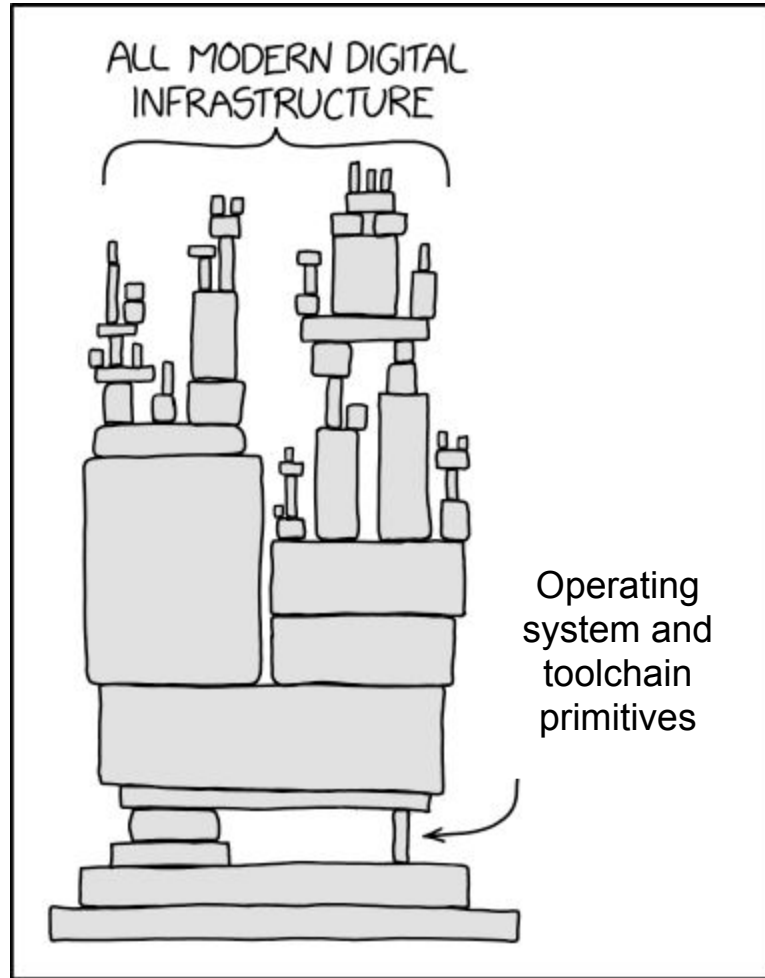
https://lore.kernel.org/lkml/CAHk-=whs8QZf3YnifdLv57+FhBi5_WeNTG1B-suOES=RcUSmQg@mail.gmail.com/

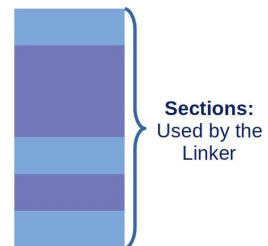
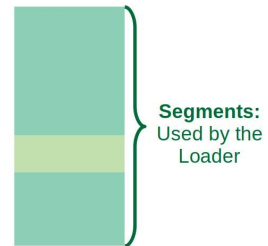
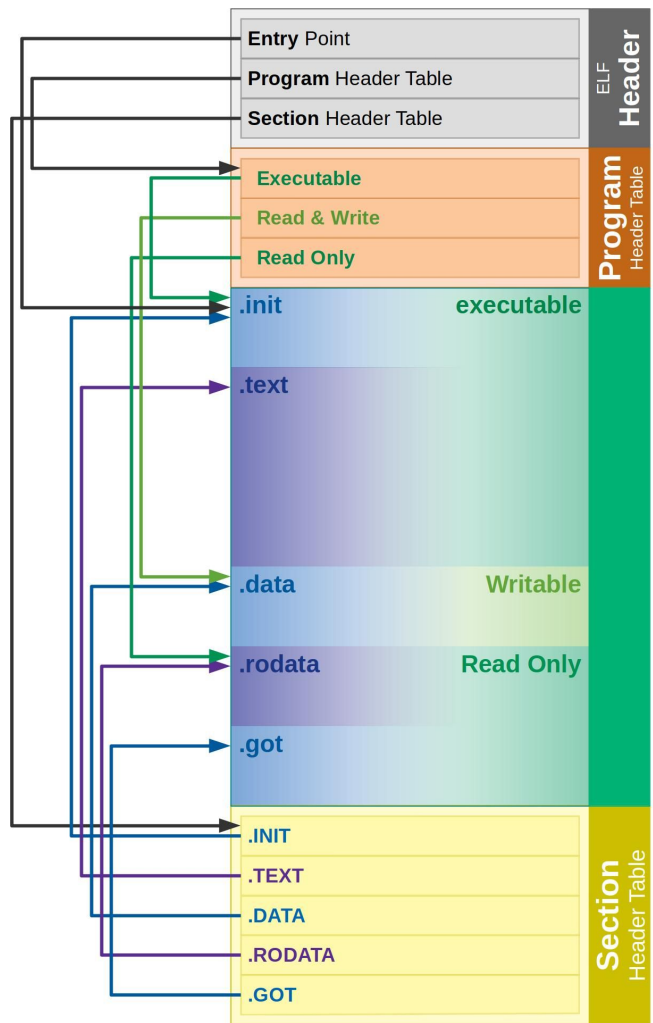


ImportError:

```
/opt/conda/envs/pytorch/lib/python3.10/  
site-  
packages/flash_attn_2_cuda.cpython-310-  
x86_64-linux-gnu.so: undefined symbol:  
_ZN2at4_ops5zeros4callEN3c108ArrayRefIN  
S2_6SymIntEEENS2_8optionalINS2_10Scalar  
TypeEEENS6_INS2_6LayoutEEENS6_INS2_6Dev  
iceEEENS6_IbEE
```





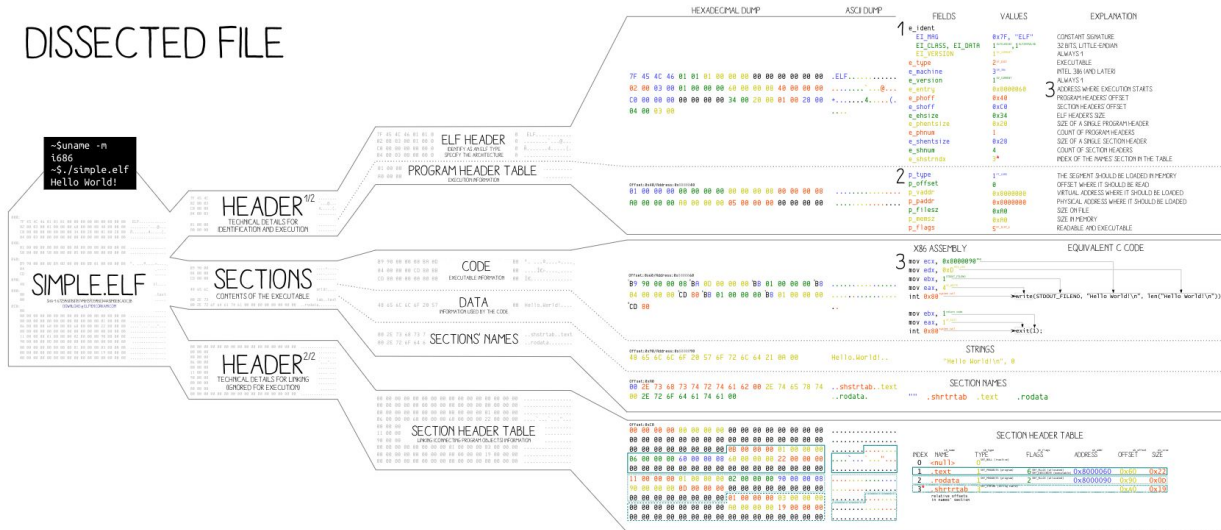


ELF¹⁰¹ a Linux executable walk-through

ANGE ALBERTINI
CORKAMI.COM



DISSECTED FILE



LOADING PROCESS

1 HEADER

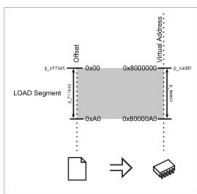
THE ELF HEADER IS PARSED
THE PROGRAM HEADER IS PARSED
(SECTIONS ARE NOT USED)

2 MAPPING

THE FILE IS MAPPED IN MEMORY
ACCORDING TO ITS SEGMENT(S)

3 EXECUTION

ENTRY IS CALLED
SYSCALLS¹ ARE ACCESSED VIA:
- SYSCALL NUMBER IN THE EAX REGISTER
- CALLING INTERRUPT 0X80



TRIVIA

THE ELF WAS FIRST SPECIFIED BY U.S. L¹ AND UJ¹
FOR UNIX SYSTEM V, IN 1989

THE ELF IS USED, AMONG OTHERS, IN:

- LINUX, ANDROID, *BSD, SOLARIS, BEOS
- PSP, PLAYSTATION 2-4, DREAMCAST, GAMECUBE, Wii
- VARIOUS OSes MADE BY SAMSUNG, ERICSSON, NOKIA,
- MICROCONTROLLERS FROM ATMEL, TEXAS INSTRUMENTS

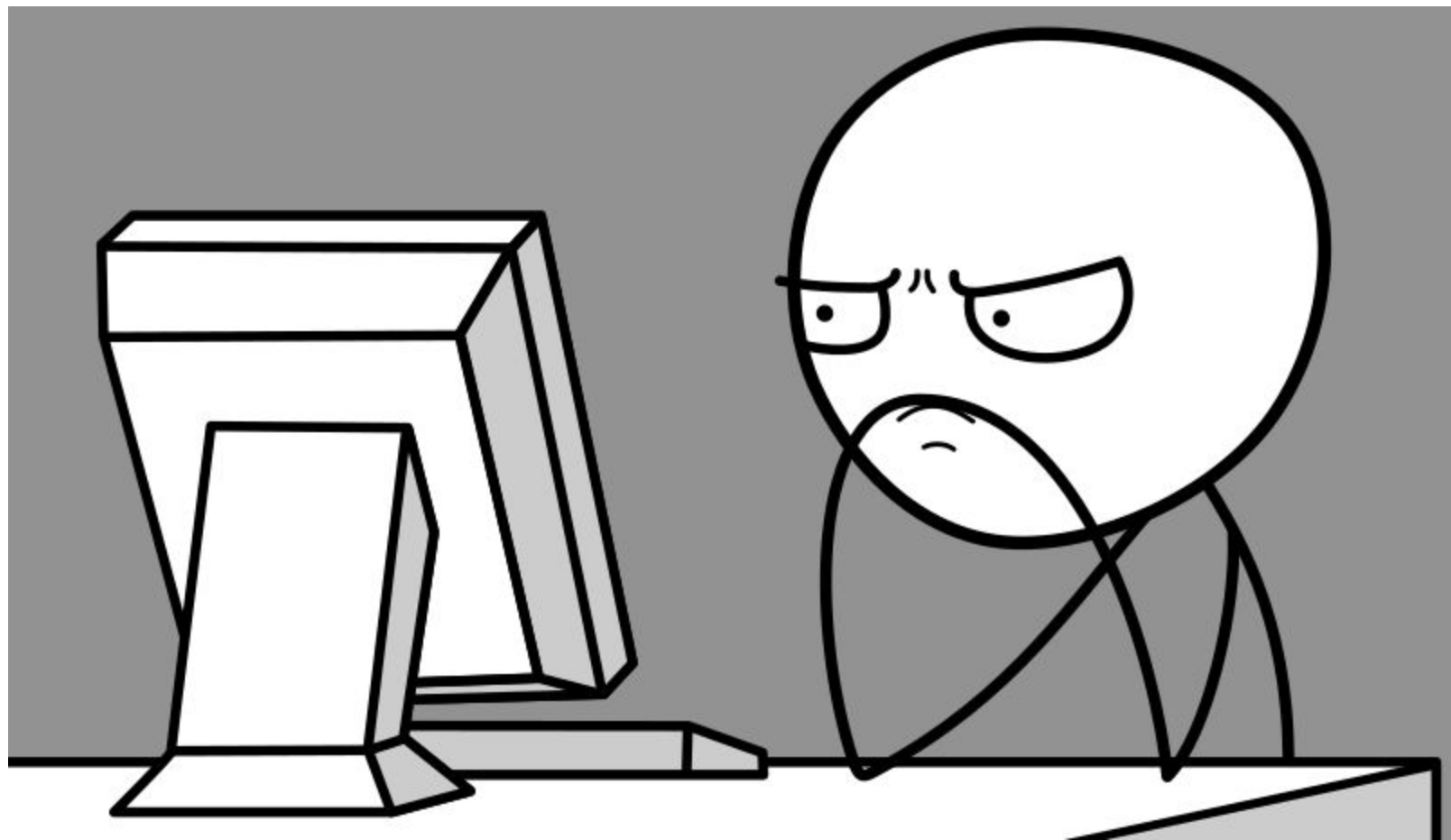


```
$ readelf --demangle --dyn-syms /usr/bin/ruby
```

```
Symbol table '.dynsym' contains 22 entries:
```

Num:	Value	Size	Type	Bind	Vis	Ndx	Name
0:	0	0	NOTYPE	LOCAL	DEFAULT	UND	
1:	0	0	FUNC	GLOBAL	DEFAULT	UND	ruby_run_node
2:	0	0	NOTYPE	WEAK	DEFAULT	UND	__gmon_start__
3:	0	0	FUNC	GLOBAL	DEFAULT	UND	ruby_init
4:	0	0	FUNC	GLOBAL	DEFAULT	UND	ruby_options
5:	0	0	NOTYPE	WEAK	DEFAULT	UND	_ITM_deregis
6:	0	0	NOTYPE	WEAK	DEFAULT	UND	_ITM_registe

Go away or I
will replace you
with a very small
shell script.

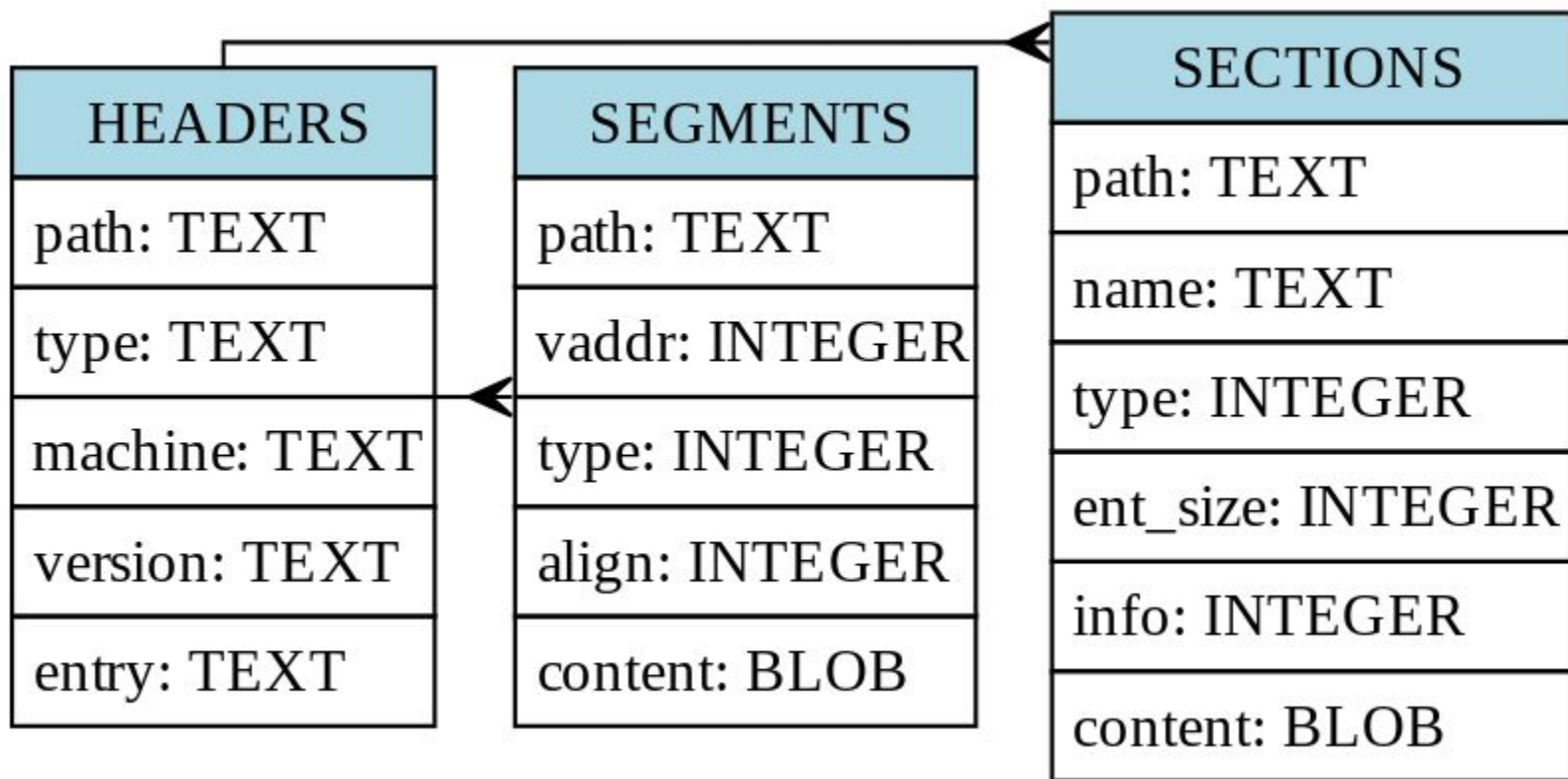




SQ1  *ELF*

sqlite







SQL ELF

Images Perspectives Videos File Commands Shopping News Maps Books

Get an AI-powered overview for this search? [Generate](#)



GitHub
https://github.com · fzakaria · sqlelf

fzakaria/sqlelf: Explore ELF objects through the power of SQL
A tool that utilizes SQLite's virtual table functionality to allow you to explore Linux ELF objects through SQL. Traditionally exploring an ELF file was ...

People also ask

What is ELF command?

What is the difference between bin and ELF?

What does ELF stand for in programming?

What is ELF in advanced operating system?

Ask a follow up...



Feedback



sqlelf.com
https://sqlelf.com

Sqlelf - AI chat-based SQL client

Apr 11, 2023 — Query your database with natural language. Chat with SQLelf and let its GPT4 do its magic. It supports MySQL and Postgres while support from ...



fzakaria.com
https://fzakaria.com · 2023/09/11 · quick-insights-using...

Quick insights using sqlelf - Farid Zakaria's Blog

Sep 11, 2023 — Come on a stroll with me on a few case studies I've gone through on how SQL-guided analysis wins out. Symbol Resolution. One of the primary data ...



Microsoft Community Hub
https://techcommunity.microsoft.com · ba-p

SQL Server on Linux: ELF and PE Images Just Work

Feb 24, 2022 — ELF image format is the image format known to the Linux kernel. PE image format is the image format known the Windows kernel.



Blogger.com
http://sqlelf.blogspot.com

SQL ELF

Eric is a former President of the Sacramento SQL Server User Group, and actively speaks at SQL Server events. In his free time, Eric enjoys camping, BBQ's, ...



fzakaria.com
https://fzakaria.com · 2023/03/19 · sqlelf-and-20-years...

sqlelf and 20 years of Nix - Farid Zakaria's Blog

Mar 19, 2023 — Using the power of SQLite and the Virtual Tables concept, I wrote sqlelf: Explore ELF objects through the power of SQL. > ...



sociallinuxexpo
1 month ago

sociallinuxexpo

Discover 'sqlelf' at #ScalaE21x, explore ELF objects using SQL for more advanced and versatile analysis. Say goodbye to unstructured output ...



SQL ELF

Images Perspectives Videos File Commands Shopping News Maps Books

Get an AI-powered overview for this search? [Generate](#)



GitHub
https://github.com · fzakaria · sqlelf

fzakaria/sqlelf: Explore ELF objects through the power of SQL

A tool that utilizes SQLite's virtual table functionality to allow you to explore Linux ELF objects through SQL. Traditionally exploring an ELF file was ...



sqlelf.com
https://sqlelf.com

Sqlelf - AI chat-based SQL client

Apr 11, 2023 — Query your database with natural language. Chat with SQLelf and let its GPT4 do its magic. It supports MySQL and Postgres while support from ...



fzakaria.com
https://fzakaria.com · 2023/09/11 · quick-insights-using...

Quick insights using sqlelf - Farid Zakaria's Blog

Sep 11, 2023 — Come on a stroll with me on a few case studies I've gone through on how SQL-guided analysis wins out. Symbol Resolution. One of the primary data ...



Microsoft Community Hub
https://techcommunity.microsoft.com · ba-p

SQL Server on Linux: ELF and PE Images Just Work

Feb 24, 2022 — ELF image format is the image format known to the Linux kernel. PE image format is the image format known the Windows kernel.



Blogger.com
http://sqlelf.blogspot.com

SQL ELF

Eric is a former President of the Sacramento SQL Server User Group, and actively speaks at SQL Server events. In his free time, Eric enjoys camping, BBQ's, ...



fzakaria.com
https://fzakaria.com · 2023/03/19 · sqlelf-and-20-years...

sqlelf and 20 years of Nix - Farid Zakaria's Blog

Mar 19, 2023 — Using the power of SQLite and the Virtual Tables concept, I wrote sqlelf: Explore ELF objects through the power of SQL. > ...



You

Is there any tool that combines ELF and SQL?



ChatGPT

Combining ELF (Executable and Linkable Format) and SQL (Structured Query Language) is somewhat unusual since they serve very different purposes in software development. ELF is a common file format for executables and libraries in Unix-like systems, while SQL is a language used for managing and querying data in databases.



You

Are there any tools that let me analyze ELF files with the power of SQL?



ChatGPT

There aren't many tools specifically designed to analyze ELF (Executable and Linkable Format) files using SQL (Structured Query Language) directly. However, you can potentially combine tools and techniques from different domains to achieve this. One approach is to extract data from ELF files and import it into a SQL database, which then allows you to query and analyze this data using SQL. Here's a general outline of how you might set this up:



You

What's the best way to perform in-depth aggregate analysis on ELF files on the contents of the file and its structure?



ChatGPT

Performing in-depth aggregate analysis on the contents and structure of ELF (Executable and Linkable Format) files involves a combination of binary analysis techniques and tools. The process typically includes examining the binary structure, headers, sections, symbols, dependencies, and other metadata. Here's an effective approach:

1. **Understand ELF Structure:**

- Gain a thorough understanding of the ELF file format, including headers, sections, segments, symbol tables, and other components. Resources like the ELF specification document can be invaluable.

2. **Use Binary Analysis Tools:**

- `readelf` and `objdump`: Part of the GNU Binutils, these tools provide detailed information about ELF file headers, section headers, symbol tables, etc.
- `nm`: Lists symbols from object files, useful for inspecting symbol tables in ELF files.

Your main branch isn't protected

Protect this branch from force pushing or deletion, or require status checks before merging. [View documentation.](#)

Protect this branch

main

4 Branches 4 Tags

Go to file

Add file

Code



fzakaria

Object relocation example keep main.o

4c40044 · last week

101 Commits



.github

Added FUNDING.yml file

5 months ago



.vscode

Introduce new dwarf_debug_lines table (#18)

last month



benchmarks

Minor fixes to benchmarks and added new find_shadow...

3 months ago



examples

Object relocation example keep main.o

last week



sqlElf

Add elf_relocations table (#20)

last week



tests

Introduce new dwarf_debug_lines table (#18)

last month



.envrc

[Add direnv .envrc for automatic venv source](#)

5 months ago



.gitignore

Add elf_relocations table (#20)

last week



LICENSE

Create LICENSE

last year



Makefile

Added copy.bara.sky for pyelftools

3 months ago



README.md

Add another ecample

last month



copy.bara.sky

Added copy.bara.sky for pyelftools

3 months ago



mypy.ini

Beginning of the DWARF table

3 months ago



pyproject.toml

Add elf_relocations table (#20)

last week



setup.cfg

Beginning of the DWARF table

3 months ago

README

MIT license

sqlElf

pypi v0.4
python 3.10 | 3.11
Build & Test passing

About

Explore ELF objects through the power of SQL

[sql](#)
[sqlite](#)
[elf](#)

Readme

MIT license

Activity

117 stars

4 watching

7 forks

Releases 4

v0.4 (Latest) last month

+ 3 releases

Sponsor this project



fzakaria Farid Zakaria

Sponsor

Learn more about GitHub Sponsors

Packages

No packages published
[Publish your first package](#)

Contributors 2



fzakaria Farid Zakaria

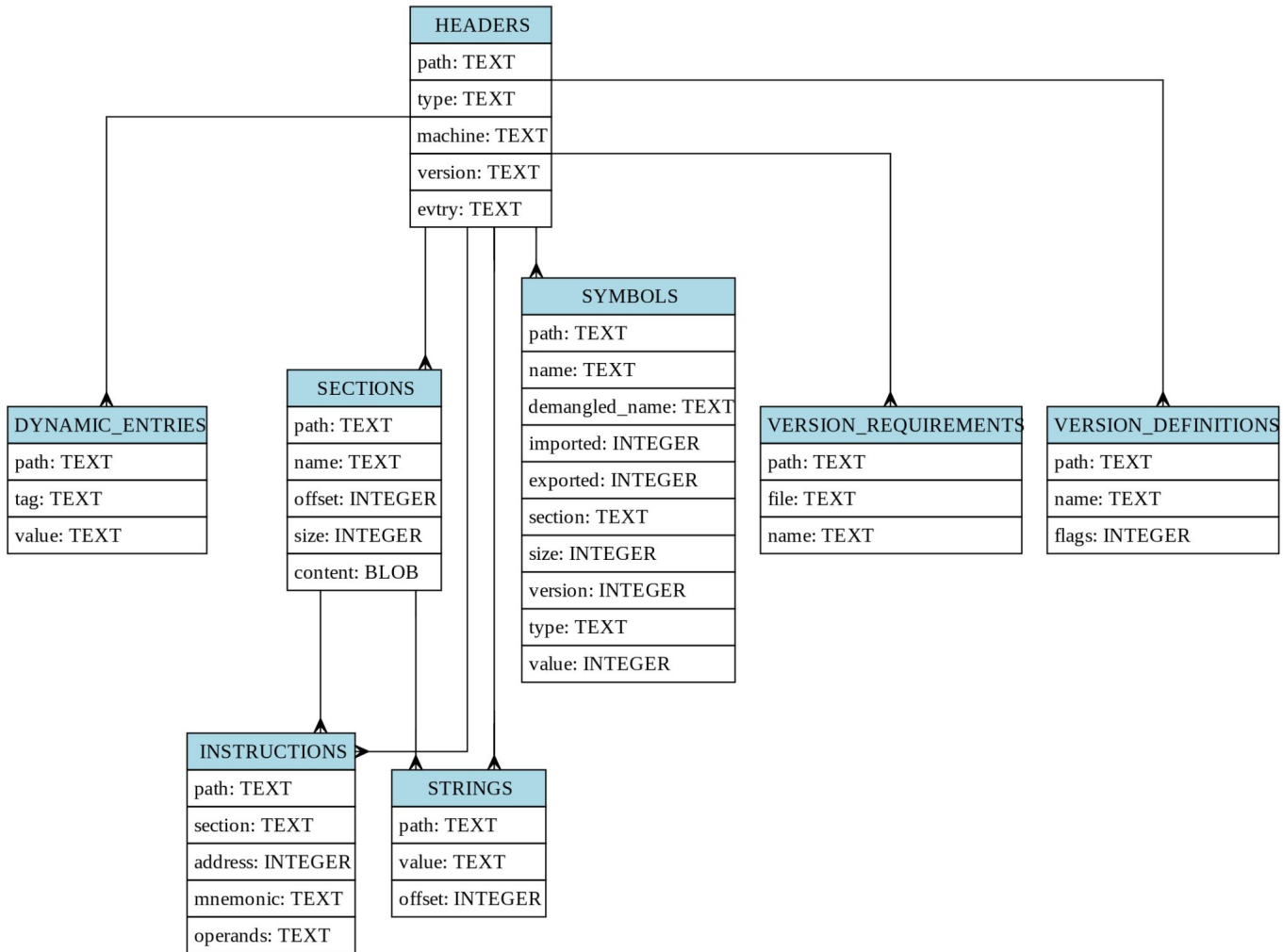


markwilliams Mark Williams

Deployments 4

pypi last month

+ 3 deployments




```

> sql elf ./examples/shadowed-symbols/exe --recursive --sql "
SELECT name, version, count(*) as symbol_count, GROUP_CONCAT(path, ':') as libraries
FROM elf_symbols
WHERE exported = TRUE
GROUP BY name, version
HAVING count(*) >= 2;"

```

name	version	symbol_count	libraries
foo	NULL	2	/usr/local/google/home/fmzakari/code/github.com/fzakaria/sql elf/examples/shadowed-symbols/x/libx.so:/usr/local/google/home/fmzakari/code/github.com/fzakaria/sql elf/examples/shadowed-symbols/x/libx2.so

```

> sql elf extension.cpython-311-x86_64-linux-gnu.so \
> --sql "SELECT
CASE name
  WHEN 'initextension' THEN 2
  WHEN 'PyInit_extension' THEN 3
  WHEN '_cffi_pypyinit_extension' THEN 2
  ELSE -1
END AS python_version
FROM elf_symbols
WHERE name IN ('initextension', 'PyInit_extension',
'_cffi_pypyinit_extension')
AND exported = TRUE
AND type = 'FUNC'
LIMIT 1
"

```

python_version
3

```

sql elf ./examples/nested-symbols/exe --sql "
SELECT outer_symbol.path,
outer_symbol.name AS outer_symbol_name,
inner_symbol.name AS inner_symbol_name
FROM
elf_symbols AS outer_symbol,
elf_symbols AS inner_symbol
WHERE
inner_symbol.section = '.text' AND
outer_symbol.section = '.text' AND
inner_symbol.path = outer_symbol.path AND
inner_symbol.value > outer_symbol.value AND
(inner_symbol.value + inner_symbol.size) < (outer_symbol.value + outer_symbol.size)
AND inner_symbol.name != outer_symbol.name LIMIT 5;"

```

path	outer_symbol_name	inner_symbol_name
./examples/nested-symbols/nested	outer_function	inner_symbol


```

> sql elf /usr/bin/ruby --sql \
"SELECT EI.address, ES.name, mnemonic, operands, EI.size
FROM ELF_SYMBOLS ES
JOIN ELF_INSTRUCTIONS EI
ON EI.PATH = ES.PATH
WHERE
EI.address >= ES.value
AND EI.address <= ES.value + ES.size
ORDER BY EI.address ASC;"

```

address	name	mnemonic	operands	size
4400	_start	xor	ebp, ebp	2
4402	_start	mov	r9, rdx	3
4405	_start	pop	rsi	1
4406	_start	mov	rdx, rsp	3
4409	_start	and	rsp, 0xfffffffffffffff0	4
4413	_start	push	rax	1
4414	_start	push	rsp	1
4415	_start	xor	r8d, r8d	3
4418	_start	xor	ecx, ecx	2
4420	_start	lea	rdi, [rip - 0x9b]	7
4427	_start	call	qword ptr [rip + 0x2ea7]	6
4433	_start	hlt		1
4434	_start	nop	word ptr cs:[rax + rax]	10

```

> sql elf examples/nested-symbols/exe --sql
\SELECT
    name AS function_name,
    (high_pc - low_pc) AS function_size
FROM
    dwarf_dies
WHERE
    tag = 'DW_TAG_subprogram'
ORDER BY
    function_size DESC
LIMIT 10;"

```

function_name	function_size
outer_function	38
main	21

```
> sqlielf /usr/bin/ruby /usr/bin/python /usr/bin/perl /usr/bin/ls \  
> --sql "SELECT * from ELF_HEADERS"
```

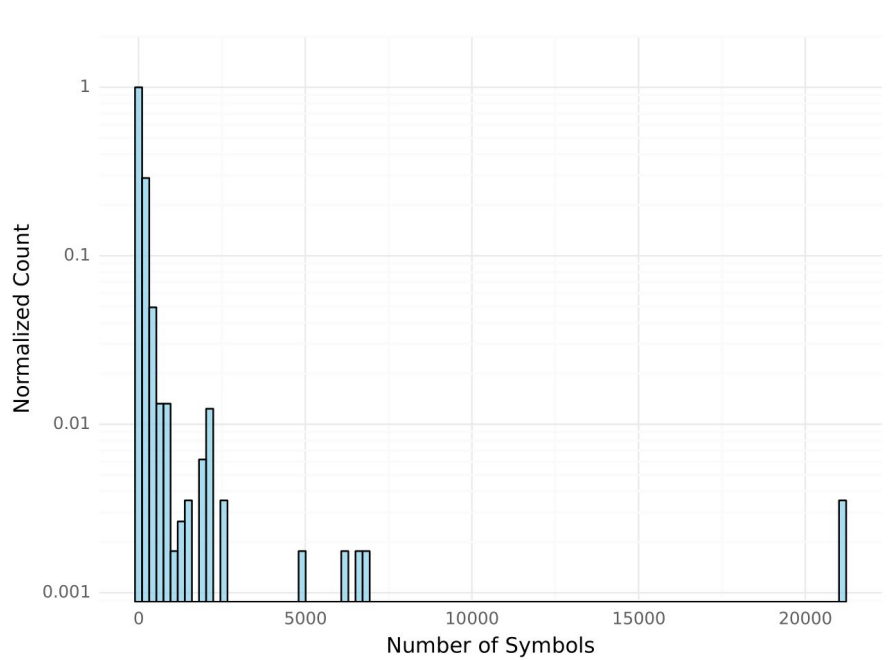
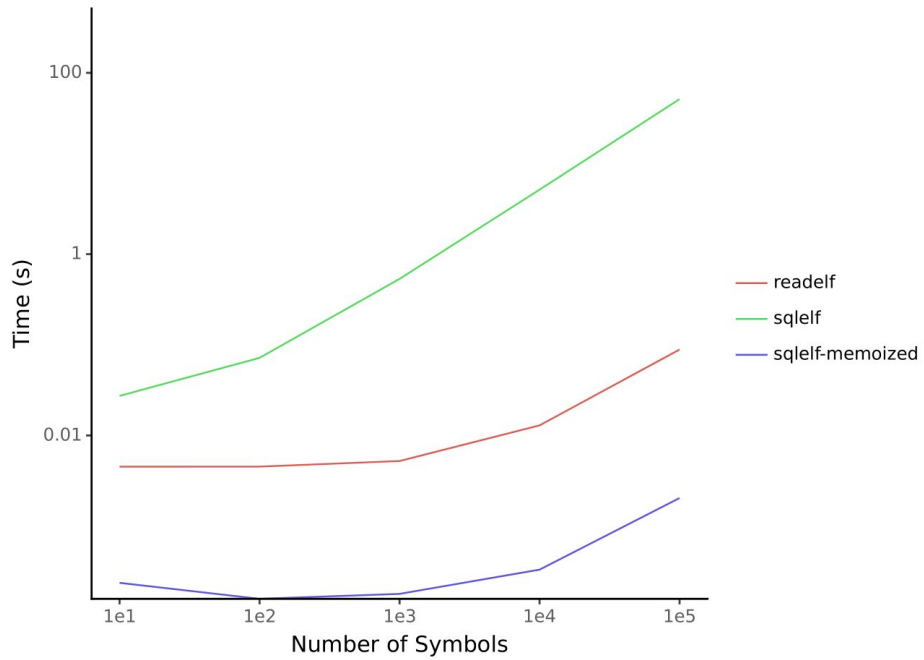
path	type	machine	version	entry	is_pie
/usr/bin/ruby	DYNAMIC	x86_64	CURRENT	4400	1
/usr/bin/perl	DYNAMIC	x86_64	CURRENT	304336	1
/usr/bin/ls	DYNAMIC	x86_64	CURRENT	25040	1

```
> sql elf examples/object-relocations/sum.o --sql \
> "SELECT * FROM ELF_RELOCATIONS LIMIT 5"
```

path	addend	info	is_rela	purpose	section	symbol	symbol_table	type
examples/object-relocations/sum.o	0	4	1	OBJECT	.debug_info		.symtab	R32
examples/object-relocations/sum.o	11	6	1	OBJECT	.debug_info		.symtab	R32
examples/object-relocations/sum.o	92	7	1	OBJECT	.debug_info		.symtab	R32
examples/object-relocations/sum.o	0	7	1	OBJECT	.debug_info		.symtab	R32
examples/object-relocations/sum.o	0	2	1	OBJECT	.debug_info		.symtab	R64

```
sql elf> SELECT DDL.filename, DIES.name, COUNT(DISTINCT DDL.line) AS line_count
..> FROM dwarf_dies AS DIES
..> JOIN dwarf_debug_lines DDL ON DIES.cu_offset = DDL.cu_offset
..> WHERE DDL.address >= DIES.low_pc AND DDL.address < DIES.high_pc
..> AND tag = 'DW_TAG_subprogram'
..> AND name IS NOT NULL
..> GROUP BY DDL.filename, DIES.name
..> ORDER BY line_count DESC
..> LIMIT 10;
```

filename	name	line_count
./build-bash/subst.c	param_expand	665
read.c	read_builtin	500
./build-bash/subst.c	expand_word_internal	461
./build-bash/shell.c	main	439
./build-bash/lib/readline/display.c	update_line	408
./build-bash/lib/readline/histexpand.c	history_expand	401
./build-bash/y.tab.c	yyparse	395
declare.c	declare_internal	378
./build-bash/variables.c	initialize_shell_variables	345
./build-bash/lib/readline/display.c	rl_redisplay	342



```
./tools/docker2sql elf debian:stable-20240211
```

```
./tools/docker2sql elf debian:buster-20240211
```

```
SELECT version
FROM (
    SELECT version, 'stable' AS debian_version
FROM `debian-stable-20240211`.elf_symbols
    UNION ALL
    SELECT version, 'buster' AS debian_version
FROM `debian-buster-20240211`.elf_symbols
)
WHERE version LIKE 'GLIBC\_%' ESCAPE '\'
GROUP BY version
HAVING COUNT(DISTINCT debian_version) = 1;
```

```
version
-----
GLIBC_2.30
GLIBC_2.31
GLIBC_2.32
GLIBC_2.35
GLIBC_ABI_DT_RELR
```

elf_sections

<https://datasette.io/>

27,629 rows

 = [Apply](#)[View and edit SQL](#)This data as [json](#), [CSV \(advanced\)](#)Suggested facets: [type](#)

Link	rowid ▼ ⚙	path ⚙	name ⚙	offset ⚙	size ⚙	type ⚙	content ⚙
1	1	/bin/du		0	0	NULL	<Binary: 0 bytes> ↓
2	2	/bin/du	.interp	792	28	PROGBITS	<Binary: 28 bytes> ↓
3	3	/bin/du	.note.gnu.property	824	48	NOTE	<Binary: 48 bytes> ↓
4	4	/bin/du	.note.gnu.build-id	872	36	NOTE	<Binary: 36 bytes> ↓
5	5	/bin/du	.note.ABI-tag	908	32	NOTE	<Binary: 32 bytes> ↓
6	6	/bin/du	.gnu.hash	944	40	GNU_HASH	<Binary: 40 bytes> ↓
7	7	/bin/du	.dynsym	984	2760	DYNSYM	<Binary: 2,760 bytes> ↓
8	8	/bin/du	.dynstr	3744	1176	STRTAB	<Binary: 1,176 bytes> ↓
9	9	/bin/du	.gnu.version	4920	230	HIOS	<Binary: 230 bytes> ↓
10	10	/bin/du	.gnu.version_r	5152	144	GNU_VERNEED	<Binary: 144 bytes> ↓

