VE482 Lab Report

Lab 4 - Fall 2020

Name: Wu Qinhang
ID: 518370910041
Email: william_wu@sjtu.edu.cn

Table of Contents

Database

database system installation

- **What are the most common database systems?**
  - MySQL, MariaDB, MongoDB, Redis, PostgreSQL
- **Briefly list the pros and cons of the three most common ones.** [1]
  - MySQL
    - pro: high performance with large dataset; open source
    - con: uneasy implementation of incremental backups; no support for XML
  - MariaDB
    - pro: high performance; multilevel encryption
    - con: uneasy data migration
  - MongoDB
    - pro: high performance; easy configuration
    - con: high usage of memory; no security setting by default

```
sqlite3 l4.db

CREATE TABLE db
(
  hash TEXT NOT NULL,
  name TEXT NOT NULL,
  subject TEXT
);
CREATE TABLE time_stamp
(
  hash TEXT NOT NULL,
  name TEXT NOT NULL,
```

```
  dates TEXT,
  tstamp INT
);

.separator "|"
.import db.psv db
.import timestamp.psv time_stamp
```

database queries

- **Who are the top five contributors to the Linux kernel since the beginning?**
  - Linus Torvalds, David S. Miller, Takashi Iwai, Mark Brown, Arnd Bergmann

```
SELECT
  name, count(*)
FROM
  time_stamp
GROUP BY
  name
ORDER BY
  count(*) DESC
LIMIT 5;
```

Output:

```
Linus Torvalds|30702
David S. Miller|13180
Takashi Iwai|7726
Mark Brown|7670
Arnd Bergmann|7520
```

- **Who are the top five contributors to the Linux kernel for each year over the past five years?** [2]
  - 2016: Linus Torvalds, Arnd Bergmann, David S. Miller, Chris Wilson, Mauro Carvalho Chehab
  - 2017: Linus Torvalds, David S. Miller, Arnd Bergmann, Chris Wilson, Arvind Yadav
  - 2018: Linus Torvalds, David S. Miller, Arnd Bergmann, Christoph Hellwig, Colin Ian King
  - 2019: Linus Torvalds, David S. Miller, Chris Wilson, YueHaibing, Christoph Hellwig
  - 2020: Linus Torvalds, David S. Miller, Christoph Hellwig, Mauro Carvalho Chehab, Chris Wilson

```
SELECT
  name, count(*)
FROM
  time_stamp
WHERE
  tstamp >= 1577836800 AND tstamp < 1609459200
GROUP BY
  name
ORDER BY
  count(*) DESC
LIMIT 5;
```

Output:

```
# 2016
# tstamp >= 1451606400 AND tstamp < 1483228800
Linus Torvalds|2273
Arnd Bergmann|1185
David S. Miller|1150
Chris Wilson|992
Mauro Carvalho Chehab|975

# 2017
# tstamp >= 1483228800 AND tstamp < 1514764800
Linus Torvalds|2303
David S. Miller|1420
Arnd Bergmann|1123
Chris Wilson|1028
Arvind Yadav|827

# 2018
# tstamp >= 1514764800 AND tstamp < 1546300800
Linus Torvalds|2168
David S. Miller|1405
Arnd Bergmann|922
Christoph Hellwig|818
Colin Ian King|798

# 2019
# tstamp >= 1546300800 AND tstamp < 1577836800
Linus Torvalds|2386
David S. Miller|1206
Chris Wilson|1173
YueHaibing|930
Christoph Hellwig|911

# 2020
# tstamp >= 1577836800 AND tstamp < 1609459200
Linus Torvalds|1886
David S. Miller|924
Christoph Hellwig|806
Mauro Carvalho Chehab|770
```

```
39  Chris Wilson|644
```

- **What is the most common "commit subject" ?**
  - `Merge git://git.kernel.org/pub/scm/linux/kernel/git/davem/net`

```
SELECT
  subject, count(*)
FROM
  db
GROUP BY
  subject
ORDER BY
  count(*) DESC
LIMIT 5;
```

Output:

```
Merge git://git.kernel.org/pub/scm/linux/kernel/git/davem/net|670
Merge branch 'for-linus' of
git://git.kernel.org/pub/scm/linux/kernel/git/dtor/input|301
Merge branch 'x86-urgent-for-linus' of
git://git.kernel.org/pub/scm/linux/kernel/git/tip/tip|275
Merge git://git.kernel.org/pub/scm/linux/kernel/git/davem/net-2.6|262
Merge branch 'perf-urgent-for-linus' of
git://git.kernel.org/pub/scm/linux/kernel/git/tip/tip|248
```

- **On which day is the number of commits the highest?**
  - `2008-01-30`

```
SELECT
  date(dates), count(*)
FROM
  time_stamp
GROUP BY
  date(dates)
ORDER BY
  count(*) DESC
LIMIT 5;
```

Output:

```
2008-01-30|1031
2006-12-07|683
2007-05-08|649
2013-07-03|626
2007-10-16|613
```

- **Determine the average time between two commits for the five main contributor.**
  - Linus Torvalds: 15881 secs
  - David S. Miller: 36956 secs
  - Takashi Iwai: 63301 secs
  - Mark Brown: 59933 secs

- Arnd Bergmann: 63807 secs

```
SELECT
  (MAX(tstamp) - MIN(tstamp)) / (count(*) - 1)
FROM
  time_stamp
WHERE
  name = "Arnd Bergmann";
```

Output:

```
# 487552654 30701
15881
# 487045012 13179
36956
# 489001082 7725
63301
# 459628018 7669
59933
# 479764856 7519
63807
```

Debugging

- **How to enable built-in debugging in gcc?**
  - by specifying `-g` to enable the extra debugging options (using GCC)
- **What is the meaning of GDB?**
  - ~~GDB cannot be eaten.~~ GDB stands for GNU debugger, which executes the program, makes it stop on specified conditions, examine the reason why it stops and changes things for the user to experiment with correcting the effects of existing bugs.

basic GDB usage

- **Find the homepage of the GDB project.**
  - https://www.gnu.org/software/gdb/
- **What languages are supported by GDB?** [3]
  - Ada
  - Assembly
  - C
  - C++
  - D
  - Fortran
  - Go
  - Objective-C
  - OpenCL
  - Modula-2
  - Pascal
  - Rust

- **What are the following GDB commands doing:** [4]
  - `backtrace` summarizes how the program got where it is, starting with the current executing frame.
  - `where` is an additional alias for `backtrace`.
  - `finish` can be used to continue execution of the current function until it returns to its caller.
  - `delete` can be used to delete a specified breakpoint
  - `info breakpoints` can be used to display defined breakpoints.
- **Search the documentation and explain how to use conditional breakpoints.**
  - format: `condition <breakpoint_number> the_condition`, where `the_condition` stands for a user-specified condition, such as `i>5`
- **What is -tui option for GDB?** [5]
  - enable GDB Text User Interface. It can also be activated with `C-x C-a`. Four text windows are available: command, source code, assembly code, and register.
- **What is the "reverse step" in GDB and how to enable it. Provide the key steps and commands.**
  - reverse step:
  - firstly, set two breakpoints (2, 3)
  - secondly, set the rule that when breakpoint 3 is hit, rerun the program: `command 3 \ run \ end`
  - set the rule that when breakpoint 2 is hit, start the recording: `command 2 \ record \ continue \ end`
  - `set pagination off`
  - `run`

Reference

1. "Most Popular Databases in 2020: Here's How They Stack Up - Ormuco," *Ormuco*, 24-Jan-2020. [Online]. Available: https://ormuco.com/blog/most-popular-databases. [Accessed: 13-Oct-2020]. ↩

2. "Unix Time Stamp - Epoch Converter," *Unixtimestamp.com*, 2020. [Online]. Available: https://www.unixtimestamp.com/index.php. [Accessed: 13-Oct-2020]. ↩

3. "GDB: The GNU Project Debugger," *Gnu.org*, 2020. [Online]. Available: https://www.gnu.org/software/gdb/. [Accessed: 13-Oct-2020]. ↩

4. https://users.ece.utexas.edu/~adnan/gdb-refcard.pdf ↩

5. CppCon, "CppCon 2015: Greg Law ' Give me 15 minutes & I' ll change your view of GDB,' " *YouTube*. 21-Oct-2015. ↩