



15-721 Project Final Presentation

By Abigale Kim, Yuchen Liang, Chi Zhang
May 5th, 2023

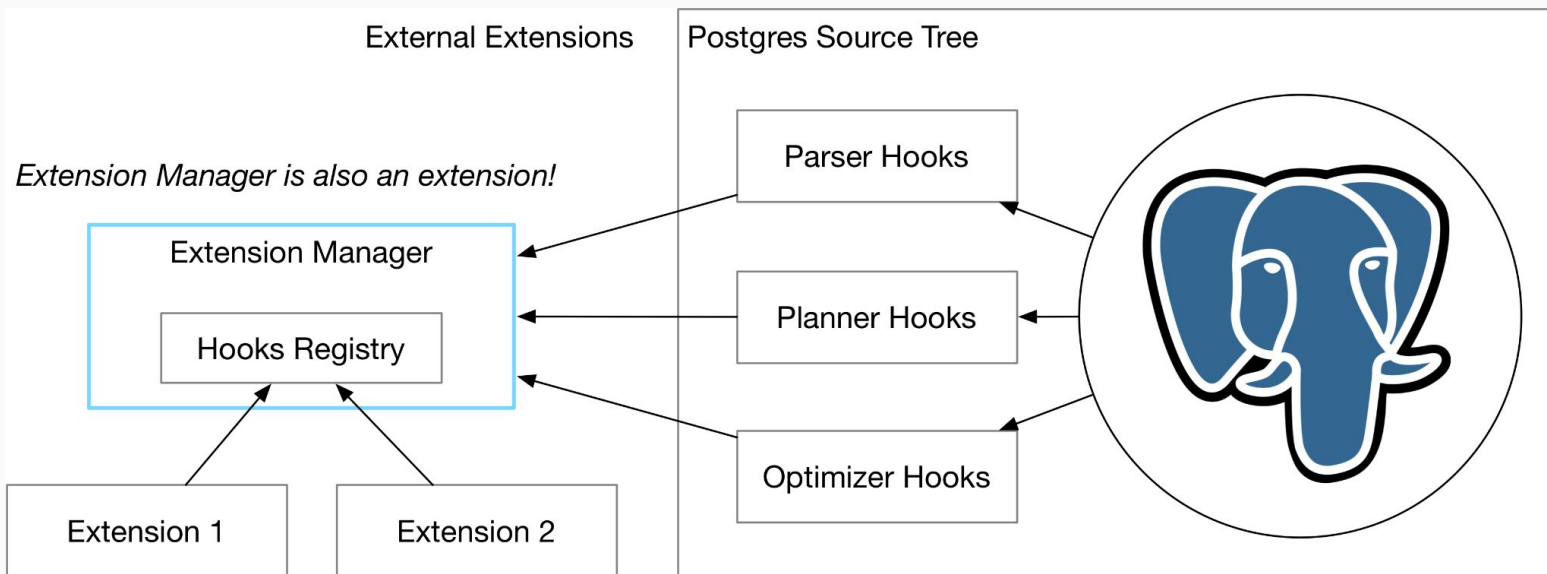


Update on Goals

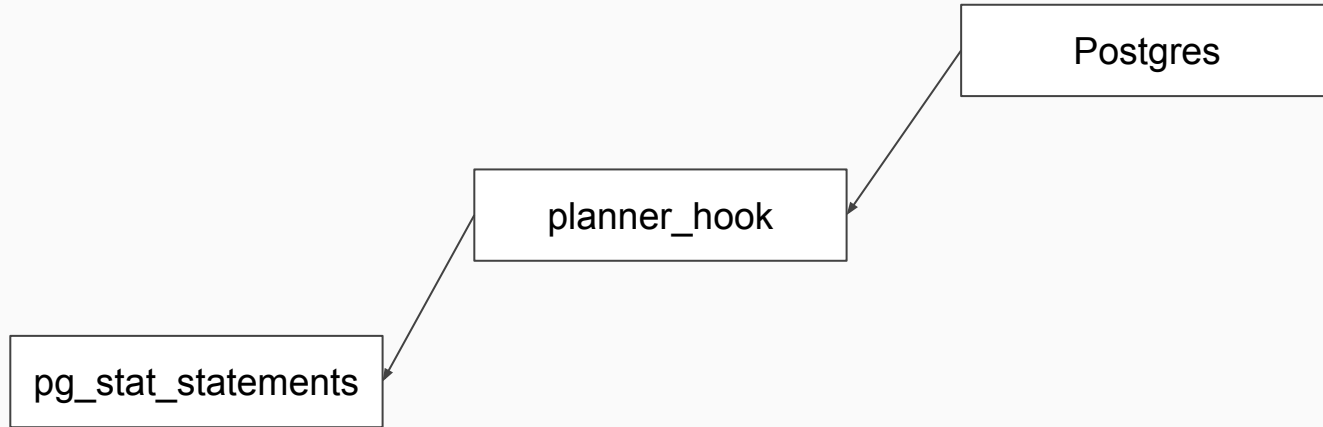
- Write a framework that determines whether extensions are likely incompatible with each other and fixes some incompatibilities (partially )
- Create an extension that manages extensions, get this to work for `pg_stat_statement` and `pg_hint_plan`; only need to `#include "pgextmgr.h"` without significant code change. ()

pgextmgrext

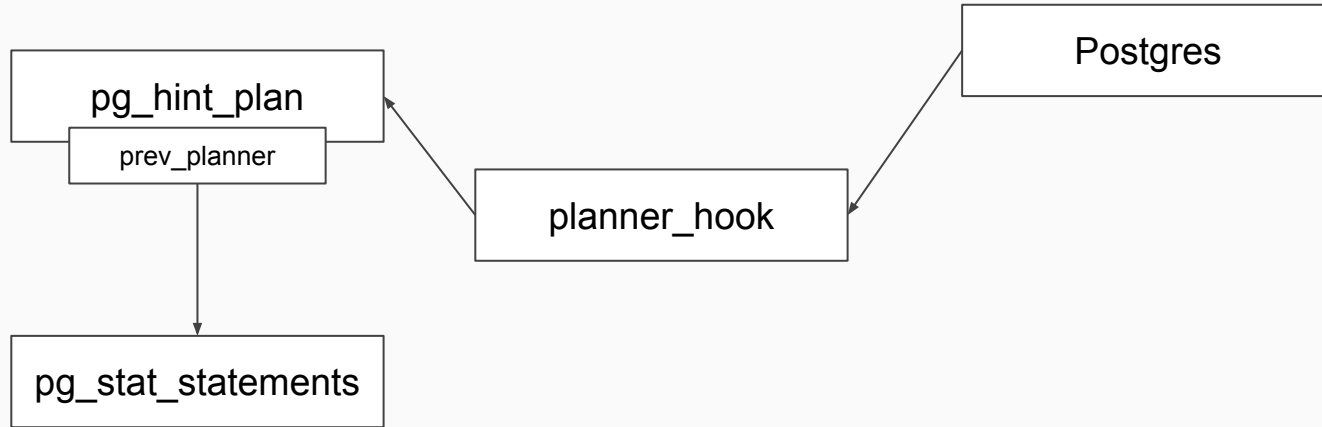
- Postgres Extension Manager as an Extension



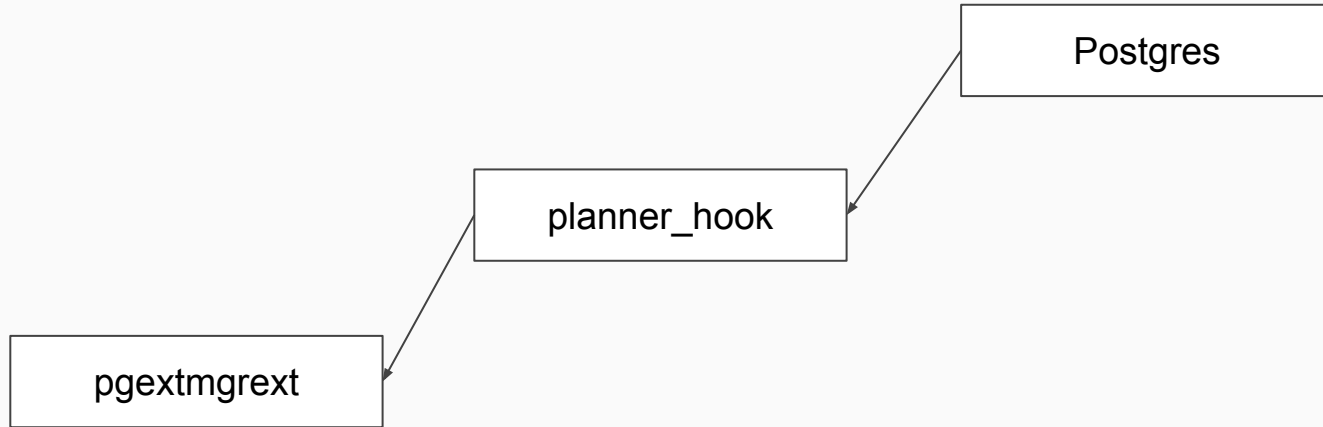
pgextmgrext in compatibility mode



pgextmgrext in compatibility mode

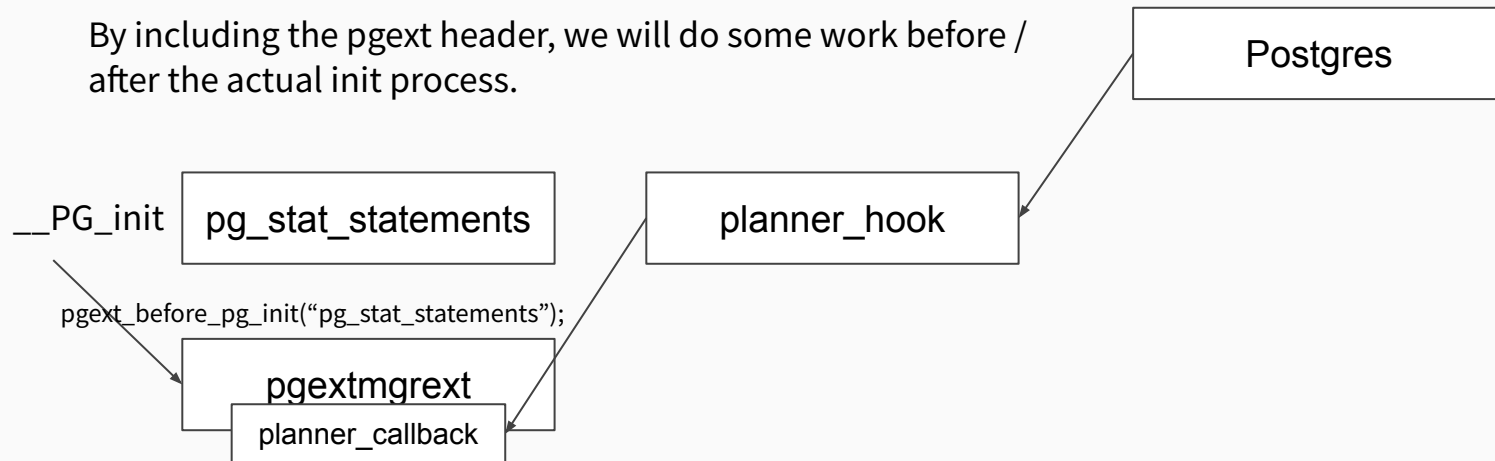


pgextmgrext in compatibility mode



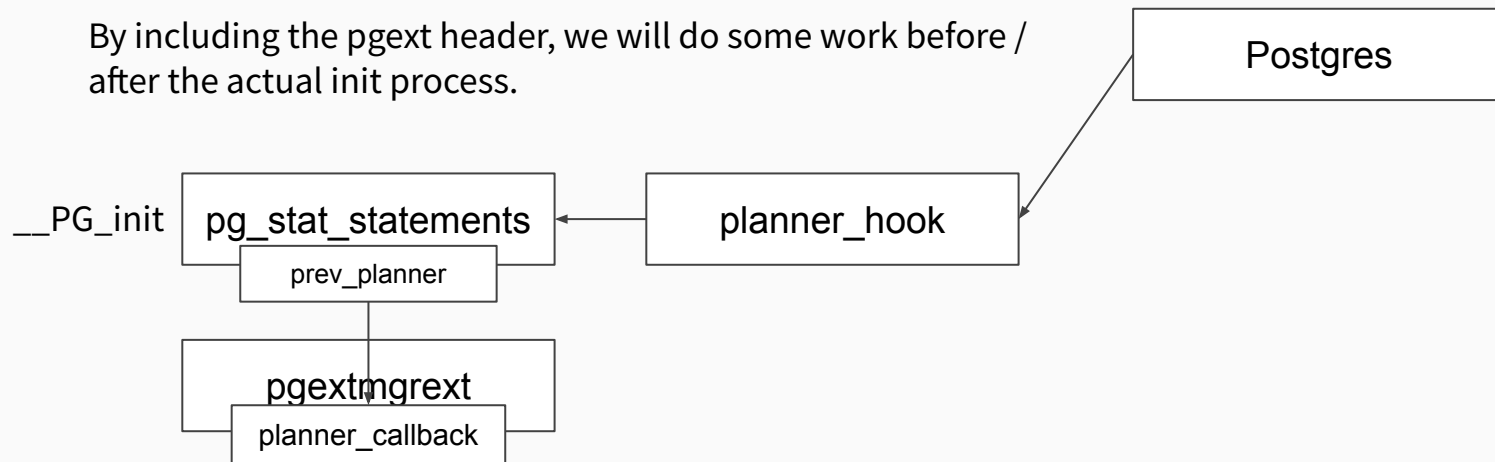
pgextmgrext in compatibility mode

By including the pgext header, we will do some work before / after the actual init process.



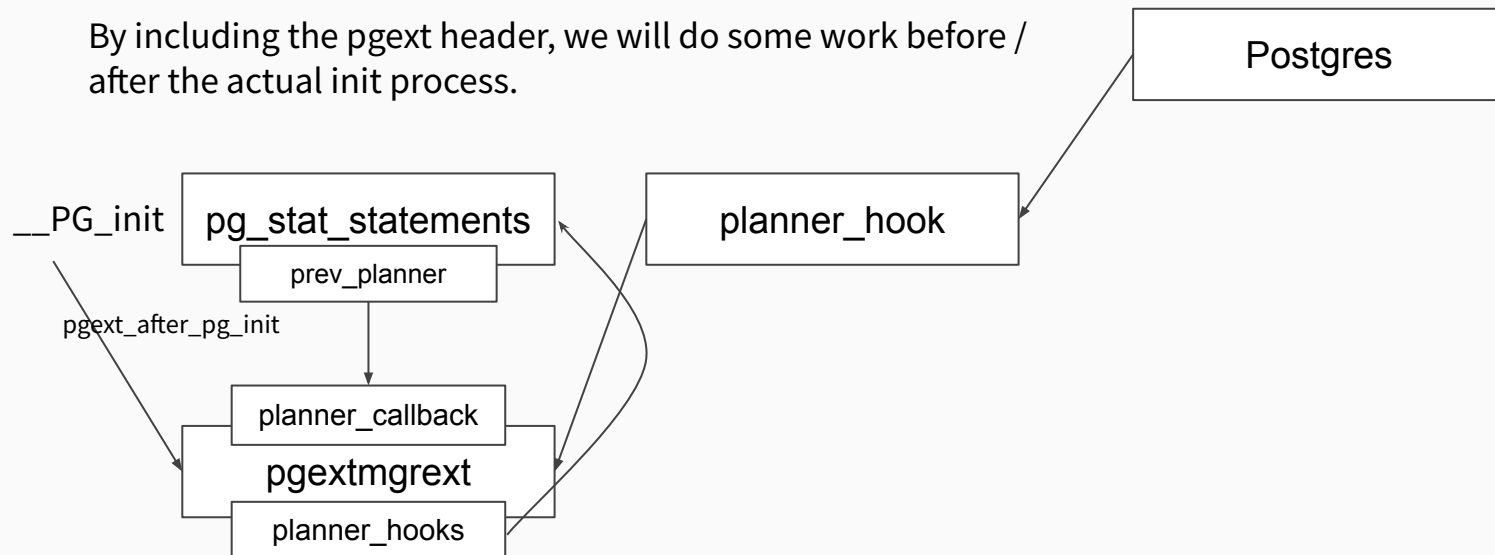
pgextmgrext in compatibility mode

By including the pgext header, we will do some work before / after the actual init process.

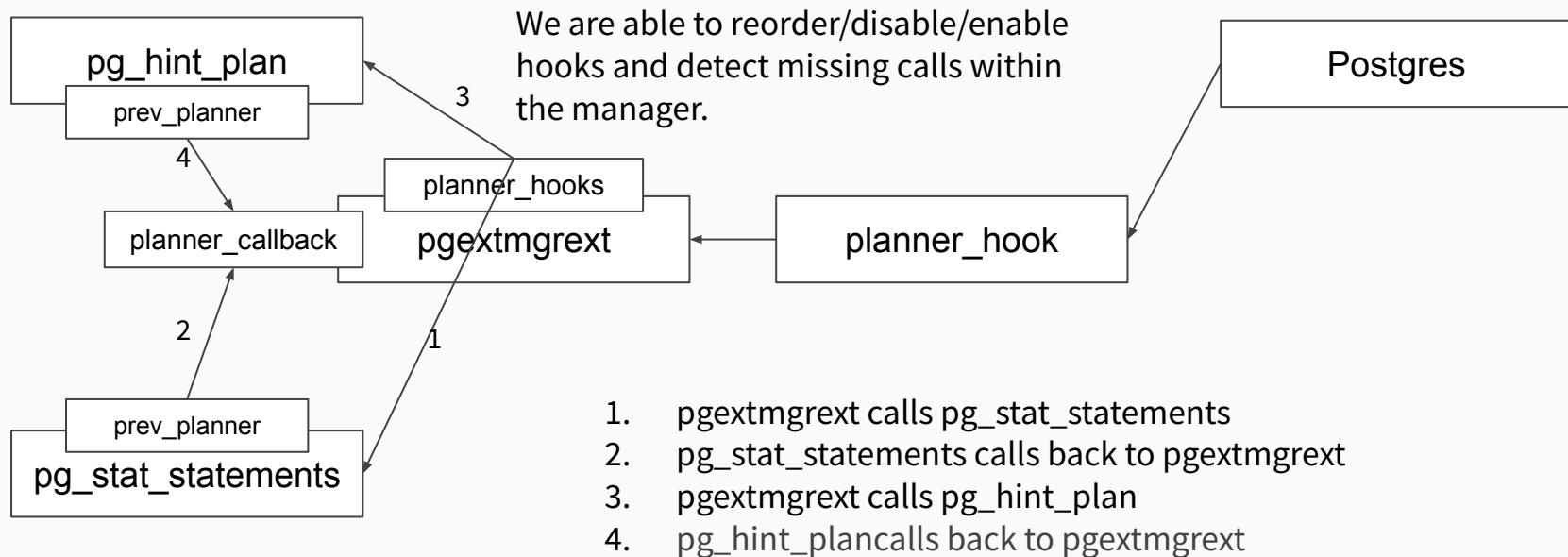


pgextmgrext in compatibility mode

By including the pgext header, we will do some work before / after the actual init process.



pgextmgrext in compatibility mode



pgextmgr

- Easy to integrate – include a single header to use existing extensions with pgextmgr.
- Easy to adapt – no need to build Postgres from source code, it is an extmgr as an extension!
- Extensibility – adding new APIs and functionalities over existing Postgres hooks so that developers can build extensions easier and write less code.

Demo Video: [here](#)

pg_poop

- Toy extension that uses our extension as a manager to replace all VARCHAR/TEXT output with poop emojis
- 200 LoCs -> 70 LoCs by using pgextmgr!
- Using pgextmgr's output rewriter APIs (which is build upon existing ExecutorRun hook)



Postgres Extension Compatibility Analysis

- Spent time understanding the extension environment (hook case studies, extension case studies)
- Tested general compatibility (installation, basic tests) of 38 extensions with each other
 - Determined that only 57 of these pairs failed (8% incompatibility)

... since that is mostly unreadable...

- TLDR: 8% is a *low* result—we're aiming for a higher incompatibility rate!
- Potentially interesting findings
 - Some extensions have checks to ensure that proper ordering (e.g. pg_queryid)
 - Some heavyweight extensions tend to clash with a lot of other extensions (Citus, plprofiler)
 - These are pretty good leads!

Testing + Code Quality

- General extension testing framework is well-tested (interface works for a lot of different extensions)
- pgextmgrext has a proof of concept implementation (pg_poop) but is otherwise not tested much (not production ready!)
- Code quality:
 - Ran cargo fmt + cargo clippy to ensure decent code style
 - We have validation of arguments in our software
 - Code is not documented very well 😭

Future Work

- Focus extension compatibility testing to extensions that share the same hooks, and extend testing to try to catch more conflicts
- Redesigning PostgreSQL extensions API using findings from `pgextmgrext` and related work (e.g. MySQL server, Redis modules, eBPF)

Thank you for listening!