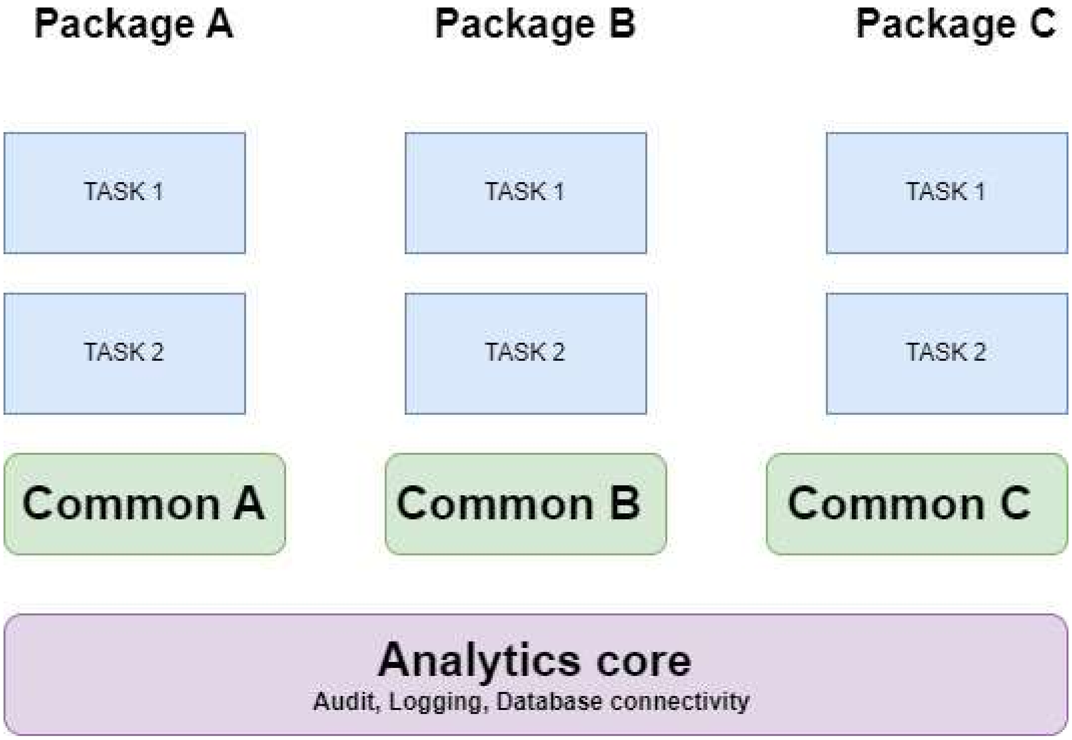
1. Package Organisation & Modularity

Last updated by | Harish Nedunuri | 21 Mar 2023 at 08:45 GMT

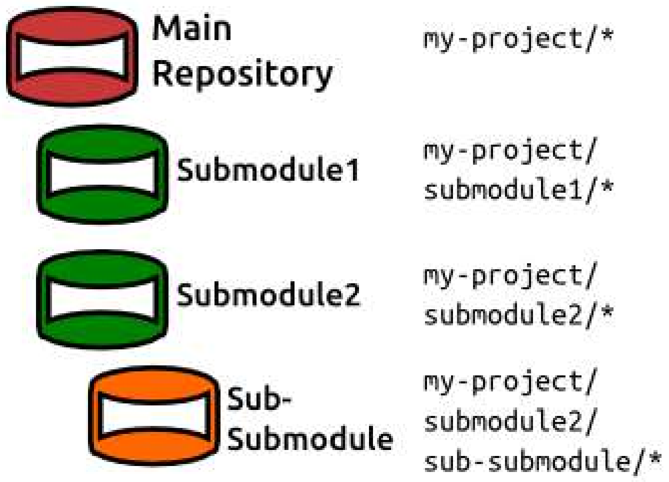
Pacakge's / Repos structure

- A production-quality code must be consistently structured and should avoid the repetition of code blocks as much as possible. A potential structure is outlined below.
- Package A, B, and C are responsible for specific functionality in the below case. Every package could contain individual tasks but use a common transformation (common A, B, and C).
- Every package would have a few core modules with (ex: Analytics core) logging, and database management.
- These core modules help achieve consistency in the data among several packages and improve reusability.



Git sub-modules

- Git submodules allow you to keep a git repository as a subdirectory of another git repository.
- Git submodules are simply a reference to another repository at a particular snapshot in time.
- Git submodules enable a Git repository to incorporate and track the version history of external code.
- In the above example, the analytics_core module could be a potential candidate for the Git sub-modules repository.


Main Repository
my-project/*
Submodule1
my-project/ submodule1/*
Submodule2
my-project/ submodule2/*
Sub-Submodule
my-project/ submodule2/ sub-submodule/*