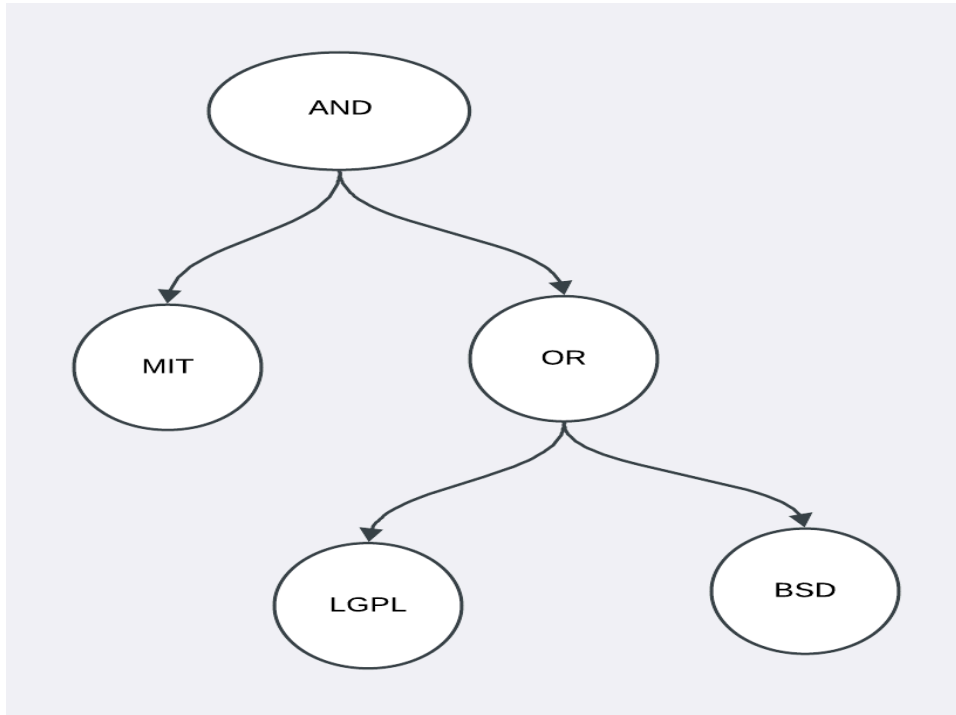# License Expression Schema

License Expression - MIT AND (LGPL OR BSD)

Abstract Syntax tree:

License_expression table schema:

- license_exp_pk - Primary key, Type bigint, Default nextval('license_ref_rf_pk_seq'::regclass)
- left_rf_fk - Type bigint
- right_rf_fk - Type bigint
- Operator - Type varchar

Left_rf_fk and right_rf_fk store rf_pk from license_ref or rf_pk from license_candidate or license_exp_pk from license_expression.

Database Snapshot for the above license expression:

License_ref

rf_pk	rf_shortname
1	MIT

License Expression Schema

2	LGPL
3	BSD

License_expression

license_exp_pk	left_rf_fk	right_rf_fk	operator
4	2	3	OR
5	1	4	AND

The root node of the tree that is license_exp_pk "5" can be stored as license expression id in license_file, clearing_event etc. We can also write recursive functions to obtain the license expression using root node as well as modify it.

Another way to store: JSON

Schema:

- license_exp_pk - Primary key, Type bigint, Default nextval('license_ref_rf_pk_seq'::regclass)
- expression - type JSONB

Example snapshot:

License_expression

license_exp_pk	expression

4	{
	    "Type": "expression",
	    "Operator": "AND",
	    "Left": {
	            "Type" : "license",
	            "rf_fk": 1,
	        },
	    "Right": {
	            "Type": "expression",
	            "Operator": "OR",
	            "Left": {
	                    "Type" : "license",
	                    "rf_fk": 2,

License Expression Schema

|  | ```
        },
        "Right": {
            "Type" : "license",
            "rf_fk": 3,
    },

    },
``` |
|---|---|

Other Tables:

- License_file - license_exp_pk as rf_fk
- Highlight - fl_pk from license_file as fl_fk to highlight license expression
- License_ref - License_expression will be inherited from license_ref. Rf_pk from license_ref stored as rf_fk in expression.
- Clearing_event - license_exp_pk as rf_fk [comment, reportinfo, acknowledgement from expression for each license of expression].
- Clearing_decision - fl_pk from license_file as pfile_fk