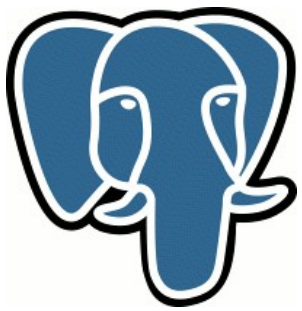
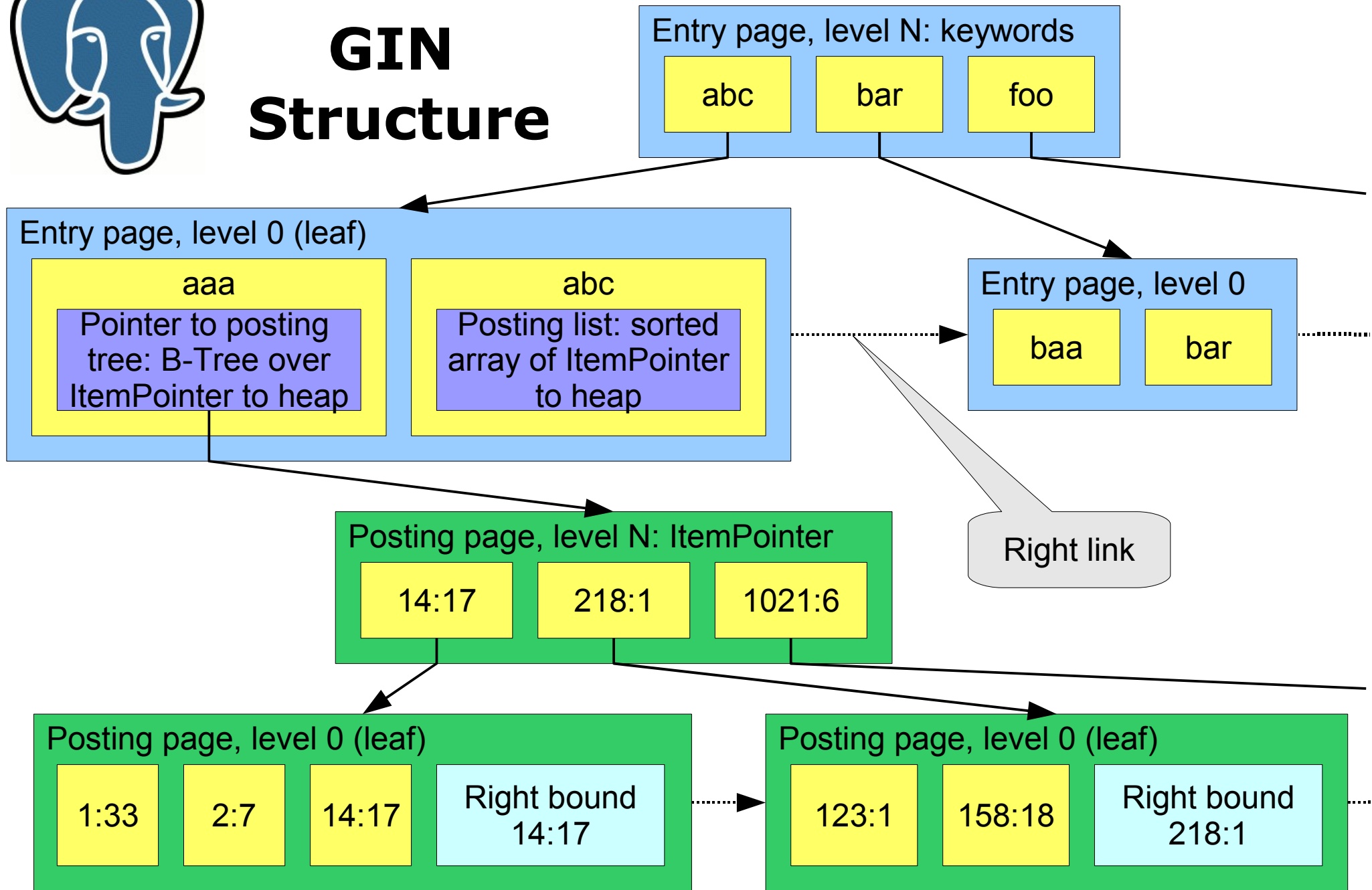


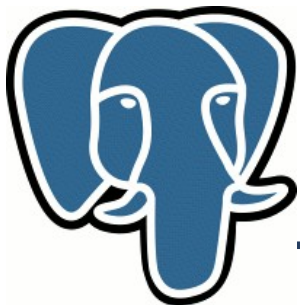
Generalized Inverted Index

- An inverted index is an index structure storing a set of (key, posting list) pairs, where 'posting list' is a set of documents in which the key occurs.
- Generalized means that the index does not know which operation it accelerates. It works with custom strategies, defined for specific data types. GIN is similar to GiST and differs from B-Tree indices, which have predefined, comparison-based operations.



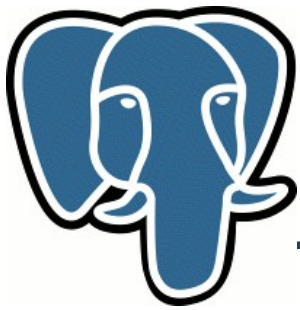
GIN Structure





GIN features

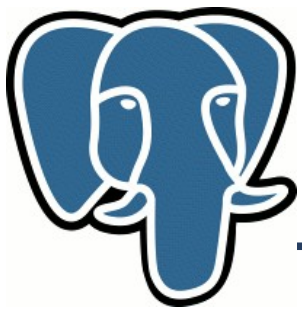
- Concurrency
 - Lehman and Yao's high-concurrency B-tree management algorithm
- WAL
 - Recovery
- User-defined opclasses
 - The scheme is similar to GiST



GIN Interface

Four interface functions (pseudocode):

- Datum* extractValue(Datum inputValue, uint32* nentries)
- int compareEntry(Datum a, Datum b)
- Datum* extractQuery(Datum query, uint32* nentries, StrategyNumber n)
- bool consistent(bool check[], StrategyNumber n, Datum query)

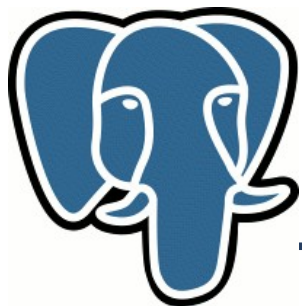


GIN Interface: extractValue

`Datum* extractValue(Datum inputValue,
uint32* nentries)`

Returns an array of Datum of entries of the value to be indexed. `nentries` should contain the number of returned entries.

Tsearch2 example: `inputValue` is `tsvector`, output is array of text type, containing lexemes.

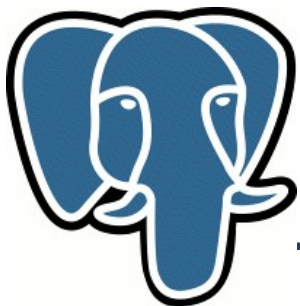


GIN Interface: compareEntry

```
int compareEntry(Datum a, Datum b)
```

Compares two entries (not the indexing values), returns <0 , 0 , >0

Tsearch2 example: built-in `btextcmp()`, used for built-in B-Tree index over texts.

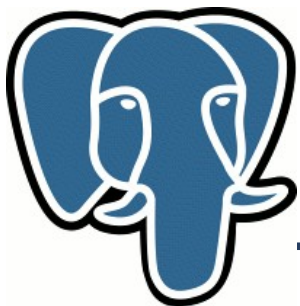


GIN Interface: extractQuery

Datum* extractQuery(Datum query,
uint32* nentries, StrategyNumber n)

Returns an array of Datum of entries of the query to be executed. n is the strategy number of the operation. Depending on n, query can be different type.

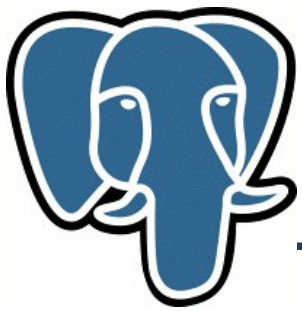
Tsearch2 example: query is tsquery, output is array of text type, containing lexemes.



GIN Interface: consistent

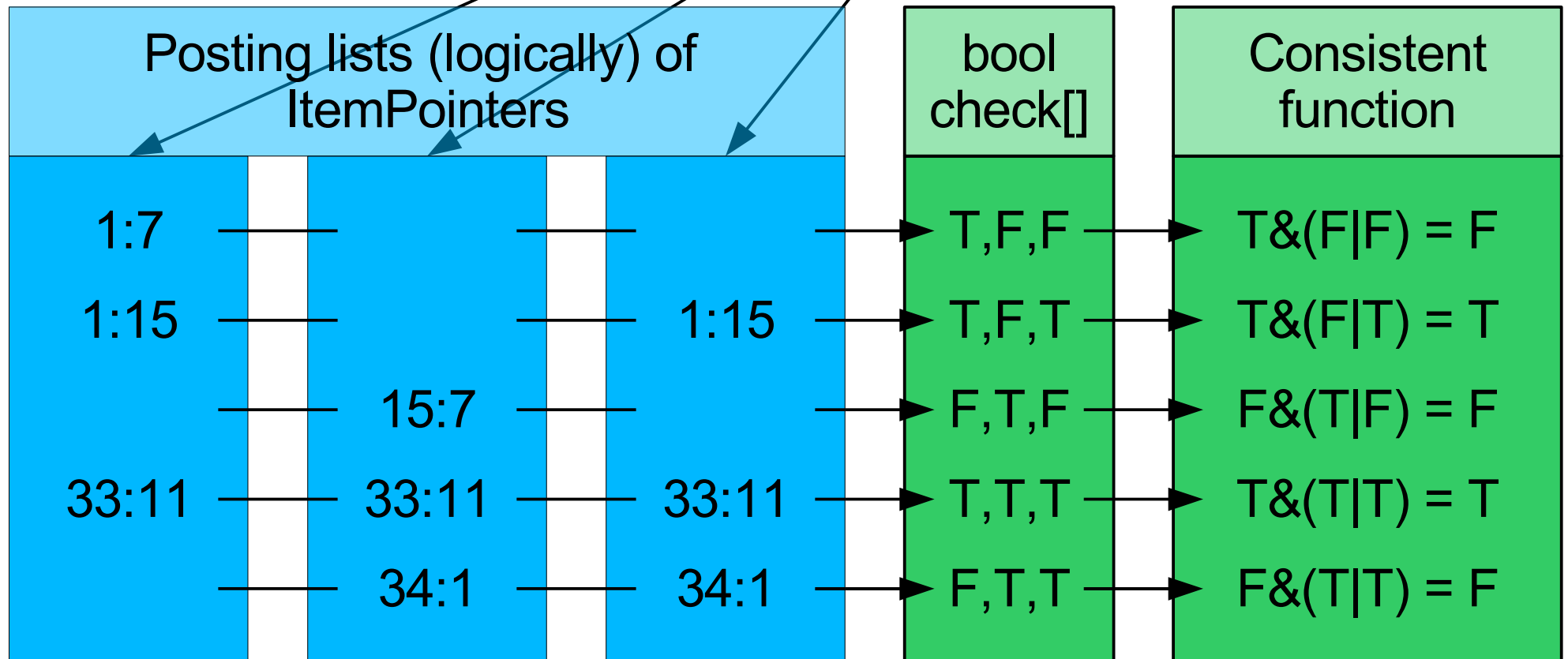
```
bool consistent(bool check[],  
                StrategyNumber n, Datum query)
```

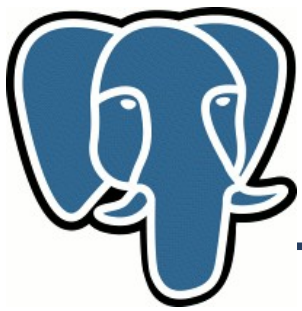
Each element of the check array is true if the indexed value has a corresponding entry in the query: if (check[i] = TRUE) then the i-th entry of the query is present in the indexed value. The function should return true if the indexed value matches by StrategyNumber and the query.



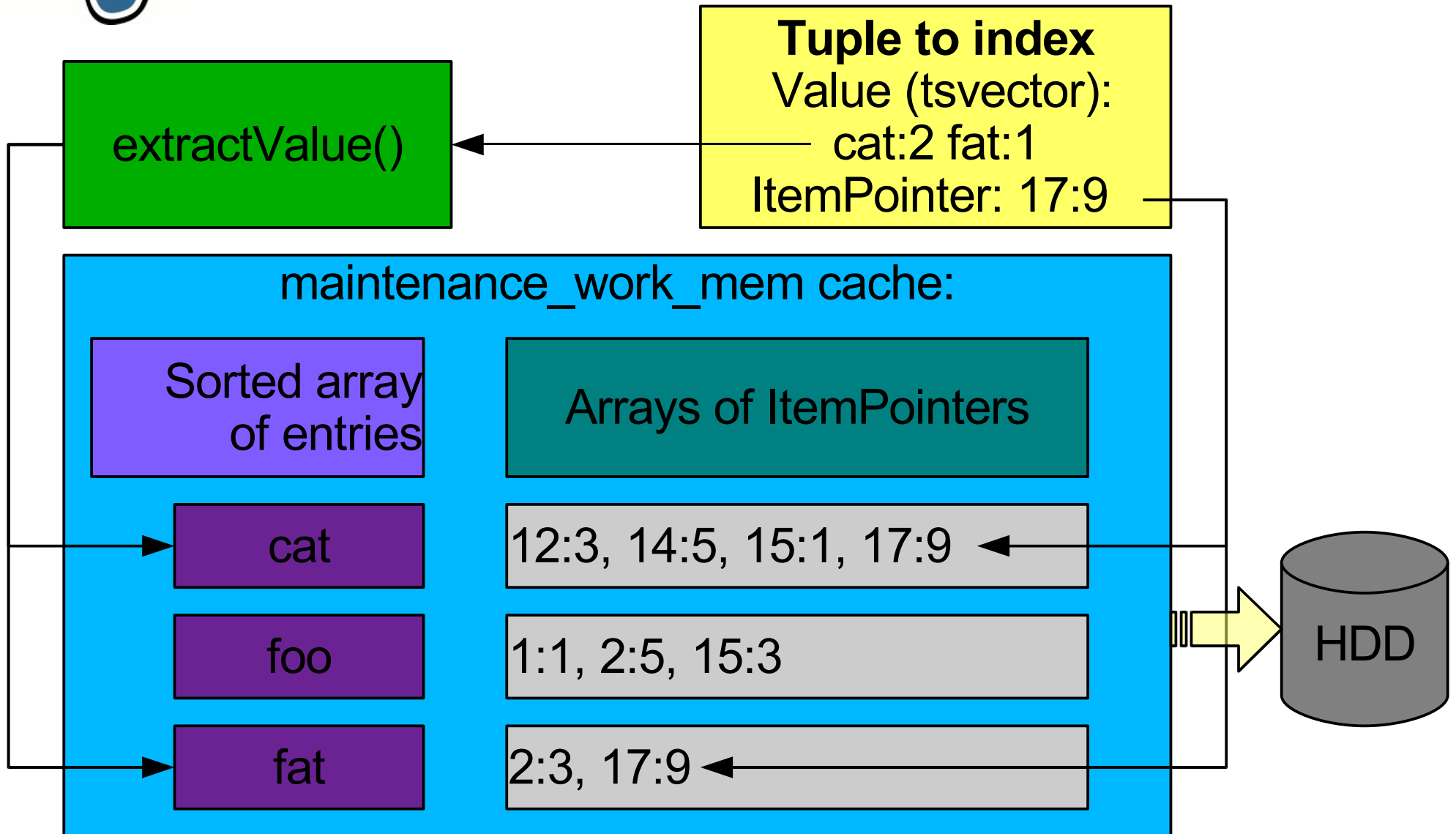
GIN Interface: consistent

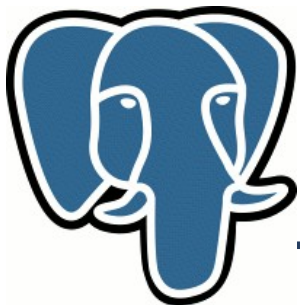
Tsquery: fat & (cat | dog)





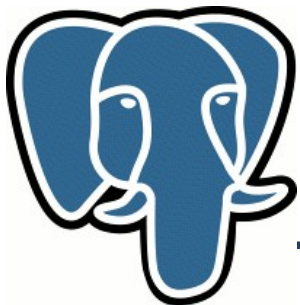
GIN: create index flow





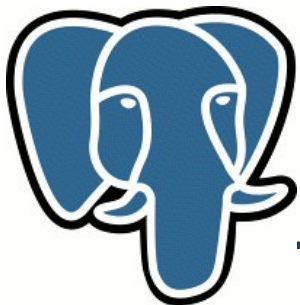
Gin opclasses

- Built-in support for any one-dimensional array
 - && - overlap
 - @ - contains
 - ~ - contained
- Tsearch2
- Intarray – enhanced support for int4[]



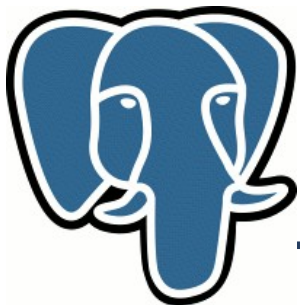
GIN tips

- GUC variable:
`gin_fuzzy_search_limit` - soft upper limit on the returned results for very frequent words
- Create is much faster than inserts



GIN limitations

- No support for multicolumn indices
- GIN doesn't use `scan->kill_prior_tuple` & `scan->ignore_killed_tuples`
- GIN searches entries only by equality matching
- GIN doesn't support full scans of index
- GIN doesn't index NULL values



???

- Two kinds of NULL
 - `(NULL = NULL)` is NULL
 - `('{NULL}'::int[]='{NULL}')` is TRUE
- Multidimensional arrays: `&&`, `@`, `~` ?
 - `'{{1,2},{3,4}}' @ '{2,3}' - ?`
- Recent fillfactor patch – nested B-Tree