TUPLE

TupleDesc
Field[]
RecordId
➔ Field Iterator
Object created by the access method HeapPage

TUPLE DESC

#Field
TDItem[]
➔ TDItem Iterator
Schema of the tuple also the header of the table.

Iterator

TD ITEM

Type
FieldName

HEAP PAGE

TupleDesc
Header[int]
Tuple[]
#Slots
➔ Tuple Iterator
Provide access methods read/write to a page of a file. Fixed num of byte sto store tuples in a slot, and additional bytes fora n header that has 1 bit for each slot to map if empty or not.

RECORD ID

Type
FieldName

BTREE LEAF PAGE

Header[int]
Tuple[]
Left/right sibling
#Slots
➔ Tuple Iterator
Add the pointer management methods for sibling leaf pages. In the Btree structure the leaf page is the only to contain the actual content o f a table (tuples)

BTREE PAGE ID

PageIDNum
TableID
PageCatalog (Internal, Leaf, Root etc.)
➔ GetPageNum()

GetChild: can be another internal page or a leaf page

BTREE ENTRY

Key
LeftChild
RightChild
➔ GetRight or Left child

BTREE INTERNAL PAGE

Header[int]
Children[PageIDNum]
Child category (leaf or internal
Keys[BTreeEntry.Key]
➔ BTreeEntry Iterator
Contain elements that allows the tree search based on the key. Doesn't contain content, only index references of keys that are in the leaf page child.

HEAP FILE

TupleDesc
File
#Pages
➔ Tuple Iterator (given)
➔ ReadPage
This is just the organizing class that provides access methods read and write to pages of a file from the disk, but it's the Buffer Pool that Manage the retrieve of single pages.

BTREE FILE

TupleDesc
File
TableId
➔ Tuple Iterator (given)
➔ BTreeEntry Iterator (given)
➔ ReadPage
➔ FindLeafPage
➔ Split
➔ Steal / Merge
Allow the DBMS to manage a balanced data structure. Use Iterator to navigate the structure.

ReadPage: used by the BP to rietrieve a page not in memory. Returns only page with tuples not structure pages

Iterator: gives the control to the buffer pool to get the page, and initialize his iterator on that page retrieved.

BUFFER POOL

HashMap <PageId, ….Page>
➔ GetPage (add a Page from disk)
➔ Flush (update page to disk)
➔ Evict (delete page)
➔ Insert/Delete tuple
Collection of pages that are currently resident in memory (that is the cache of a DBMS).

GetPage:Only for pages I don't have, I want this File to use the method to read this specific pageId

CATALOG

HashMap <TableId, Name>
HashMap <TableId, Pkey>
HashMap <TableId, DbFile>
➔ GetDBFile
List of all tables present in the DB (disk and memory). He is like the guardian, and everybody ask to him to have specific objects.

GetDBFile: used by BP and operators to retrieve a Table File object (with all the corresponding access methods)

SeqScan: ask for an iterator the tuples of this tableId I give you

SEQSCAN (operator)

TransactionID
TableId
TableAlias
➔ Tuple Iterator (given)
Operator responsible for the actual exec of querce. He ask fora n iterator through the tuples of a table he wants.

Btree Indexing: an index is a structure that improves query optimization to be faster. There are some data structures without data, where the only purpose is to help the navigation through the multitude of keys representing real data. In particular BTree structure avoid imbalance problem related to most of other tree structure.

BTREE ENTRY (1)

BT PageID left pointer	*BT Entry Key*	*BT Page ID right pointer*

BT entry iterator

BTREE ENTRY (2)