| Index Node 1 | Index Node 2 | Index Node 3 | Index Node 4 |
|---|---|---|---|
| Direct Block | Direct Block | Direct Block | Direct Block |
| Indirect Block | Indirect Block | Indirect Block | Indirect Block |
| Double Indirect Block | Double Indirect Block | Double Indirect Block | Double Indirect Block |
| Triple Indirect Block | Triple Indirect Block | Triple Indirect Block | Triple Indirect Block |

An innovative design of the index node, called extent.

Index node structure

Direct Block #0
Direct Block #1
...
Direct Block #11
Indirect Block
Double Indirect Block
Triple Indirect Block

1st layer of indirect blocks

2nd layer of indirect blocks

Data Block storing data.
Data Block storing block address.

Detailed structure of the index nodes will be talked later

Data Area

| Inode | FREE | File 1 | Dir 1 | File 2 (Under Dir 1) |
|---|---|---|---|---|

An innovative design of the index node, called extent.
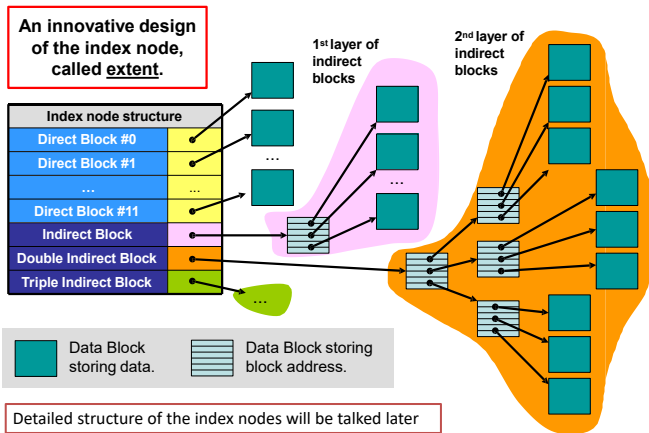
These two kinds of blocks are both stored inside the Data Area, which is the main part of this disk, including all files and directories, and they are treated the same as other blocks by the FS. These blocks are not same as Directories however, since they do not need to maintain a Look-up table, hence no more visiting to Index Node area before jumping to next block.

They act like the address double-word in Indirect memory visiting method, but they are not restricted to fragments nevertheless, and remember that I/O are always done block by block, therefore the indirect blocks store a bunch of addresses