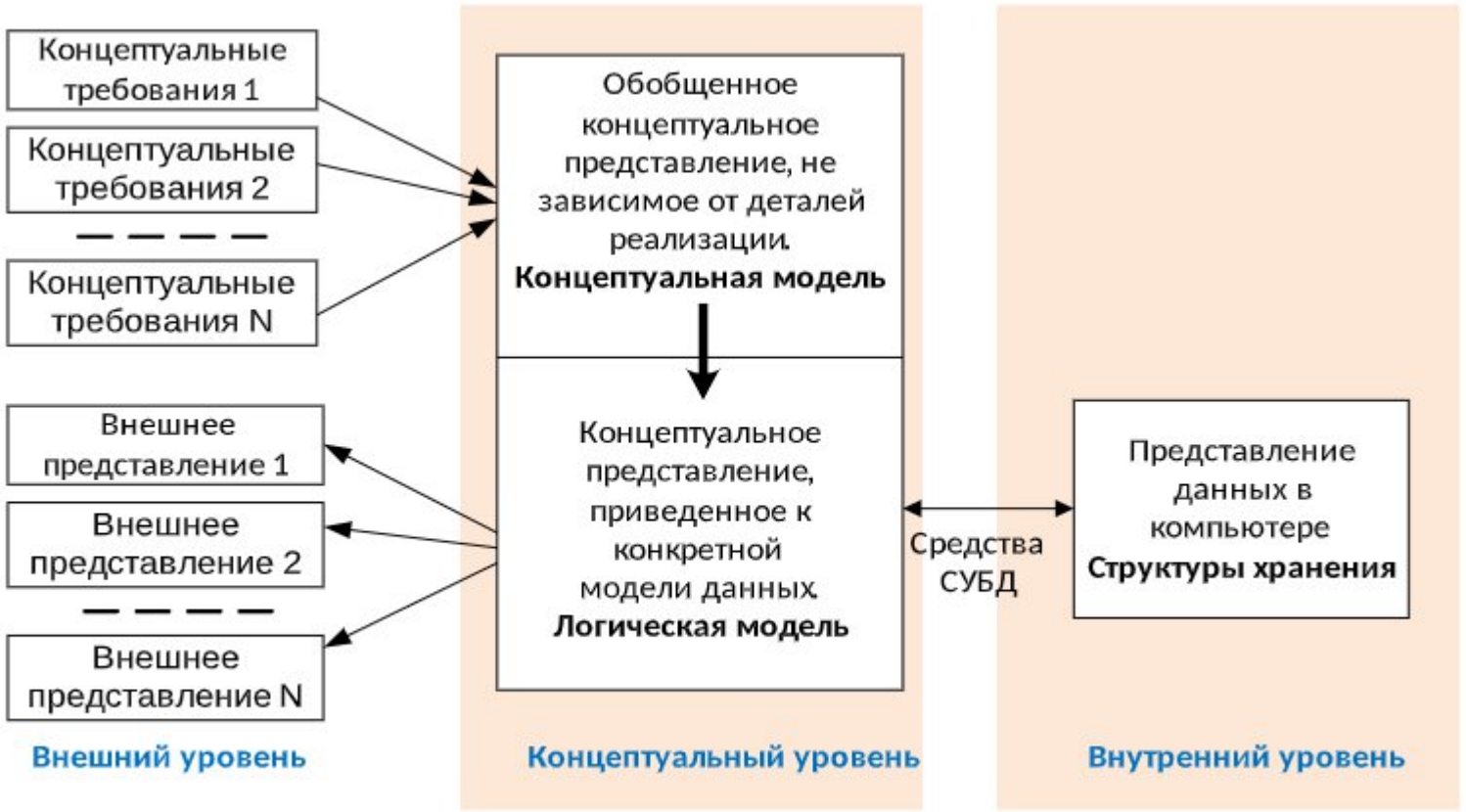
Лекция 8.

Внутреннее устройство БД

Трехуровневая архитектура данных



Логическая модель и внутреннее представление – для администратора базы данных.

Физические модели данных

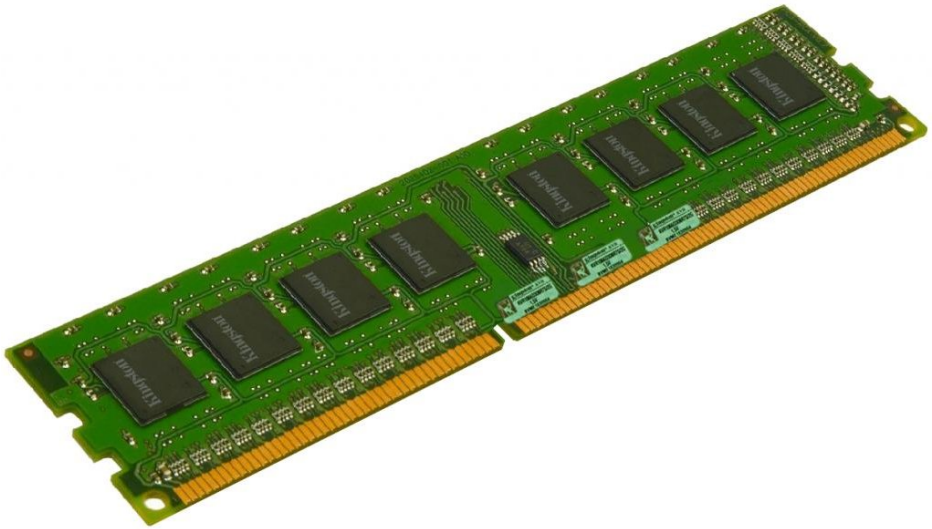
База данных хранится во внешней памяти, а процессор работает только с оперативной памятью.

Поэтому организация данных должна учитывать:

- специфику каждого вида памяти,
- способы их взаимодействия.

Оперативная память (RAM)

- Единицей памяти является **байт**.
- Память прямоадресуема (каждый байт имеет адрес).
- Процессор выбирает нужные данные, непосредственно адресуясь к последовательности байтов, содержащих эти данные.

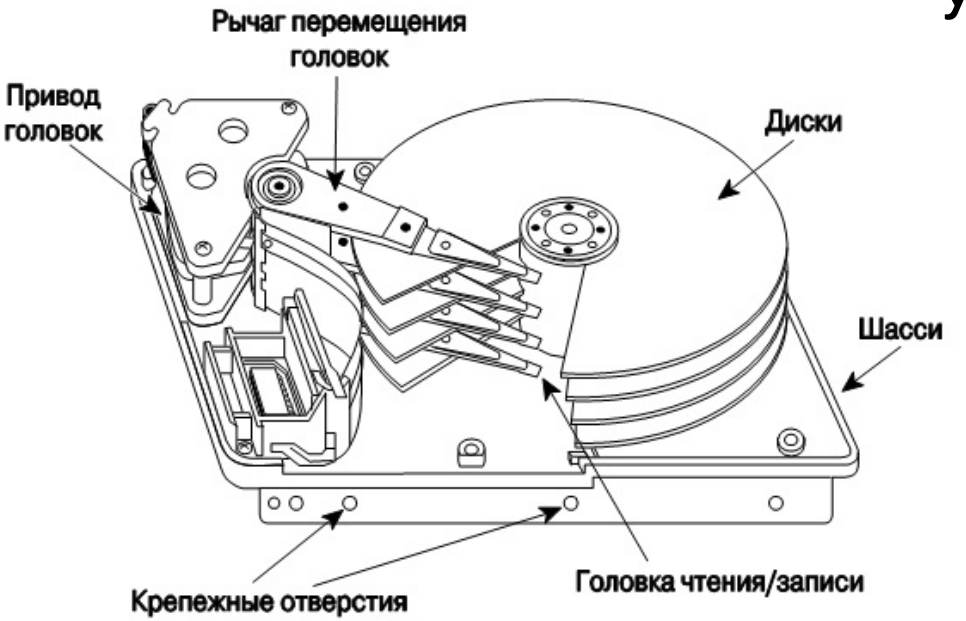


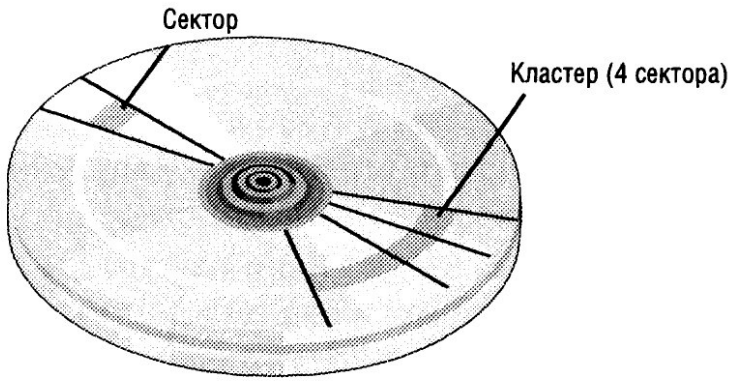
Внешняя память

- Минимальная адресуемая единица – **физическая запись**.

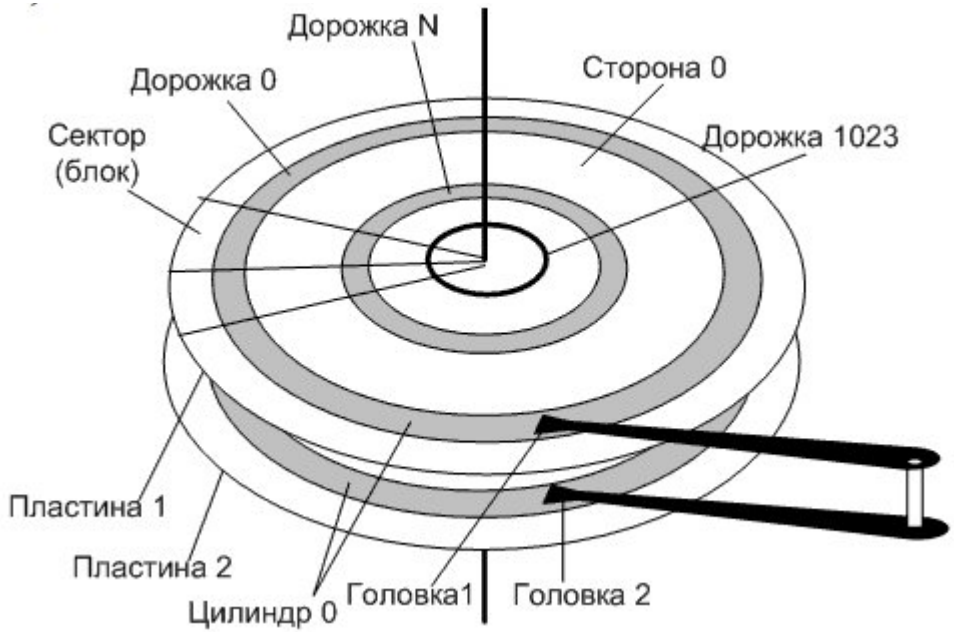
Время чтения 1 Mb данных:
RAM - 0,25 ms
SDD - 1 ms
HDD - 20 ms

Устройство жесткого диска





Секторы и кластер на диске



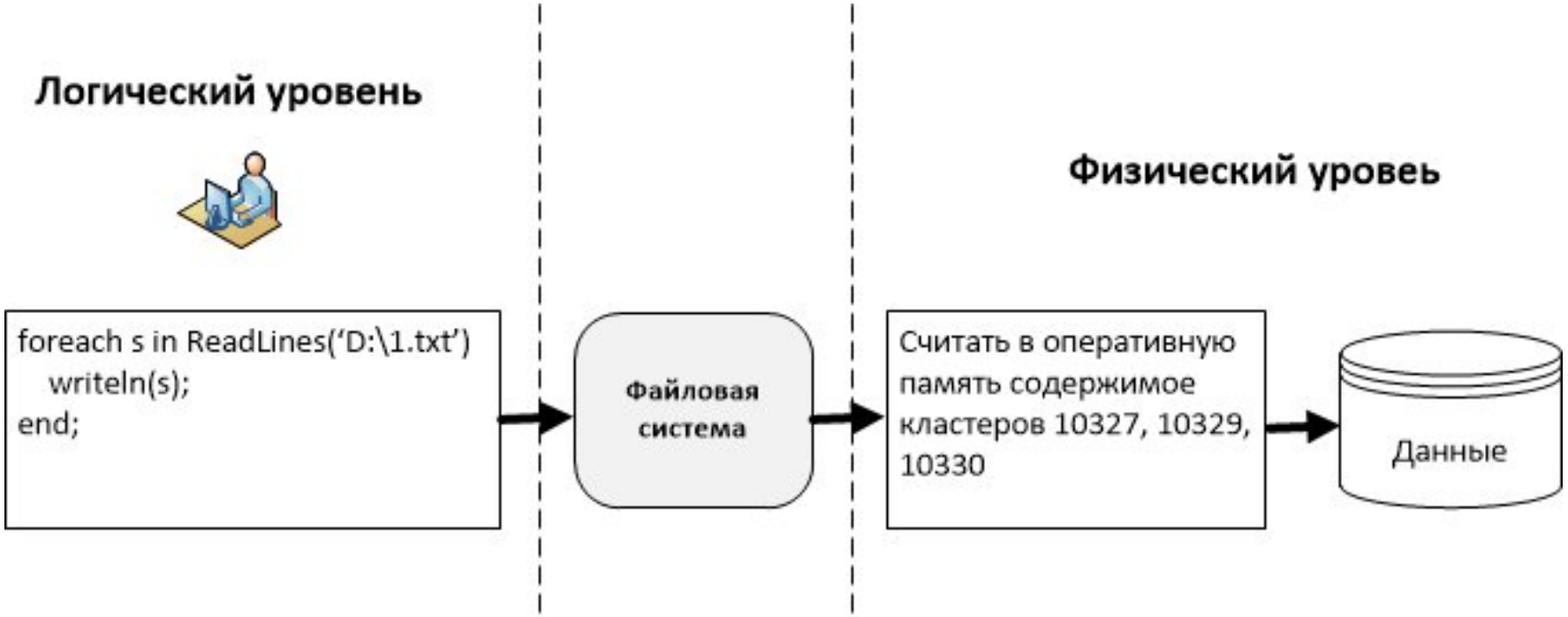
Логическая и физическая структура файлов

Для прикладной программы файл – это **именованная область внешней памяти**, в которую можно записывать и из которой можно считывать данные.

Файл на внешнем носителе – это **цепочка кластеров** (физических записей).

Файловая система

Пользователь/программист имеет дело только с логическими данными, не касаясь деталей фактического размещения данных.



C:\Документы\Мой файл.doc

Каталог файлов

Дескриптор файла 1
…
Дескриптор файла N

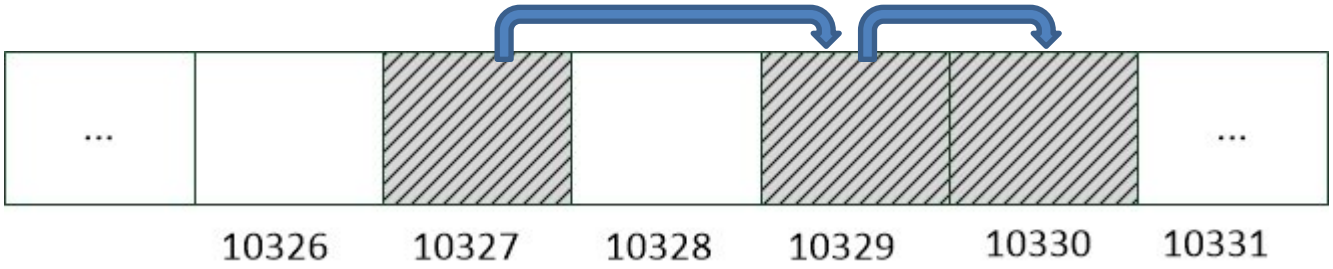
Дескриптор файла

Имя	*Мой файл.doc*
Дата создания	*01.02.2020*
Атрибуты	*Archive*
Первый кластер	*10327*
Размер	*9326*
…	…

Таблица размещения файлов (FAT)

№ кластера	Статус
10326	*Сбойный*
10327	*10329*
10328	*Свободный*
10329	*10330*
10330	*Конец цепочки*
10331	*Свободный*
…	…

Кластеры в файле организованы в **связный список**



Кластеры на диске

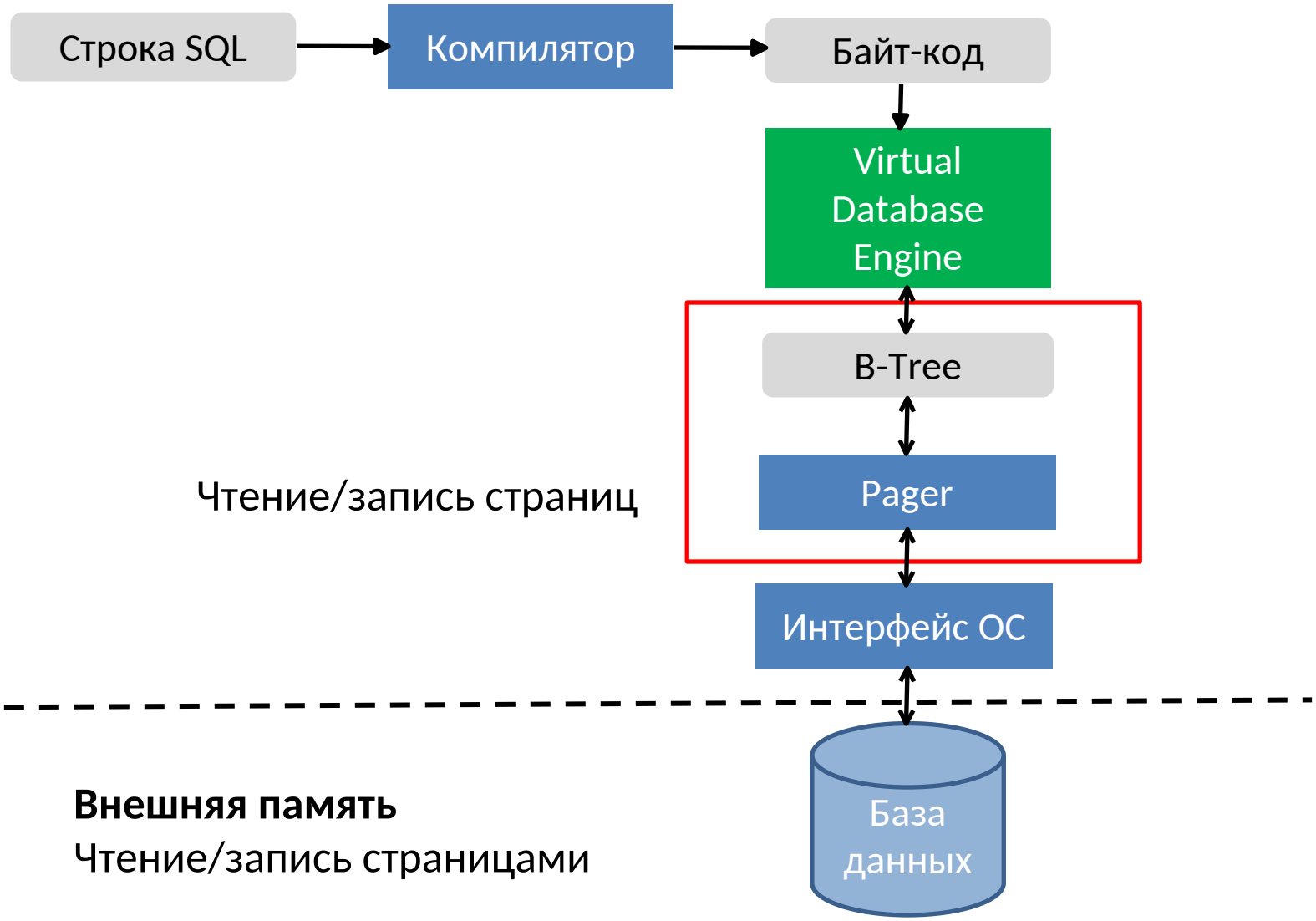
СУБД - абстракция над файловой системой

Дуэйн Ричард Хипп (D. Richard Hipp), 1961 г.р.

Создал СУБД SQLite, систему контроля версий Fossil, участвовал в разработке языка TCL

«Если мы подумаем о каждом SQL-операторе как о программе, то нужно просто взять эту программу и скомпилировать её в своего рода исполняемый код. Я придумал структуру байт-кода, который фактически запускал запрос, а затем написал компилятор, который переводил SQL в этот байт-код. И вуаля: родилась СУБД SQLite»
https://habr.com/ru/company/macloud/blog/566396/
https://habr.com/ru/company/macloud/blog/566540/

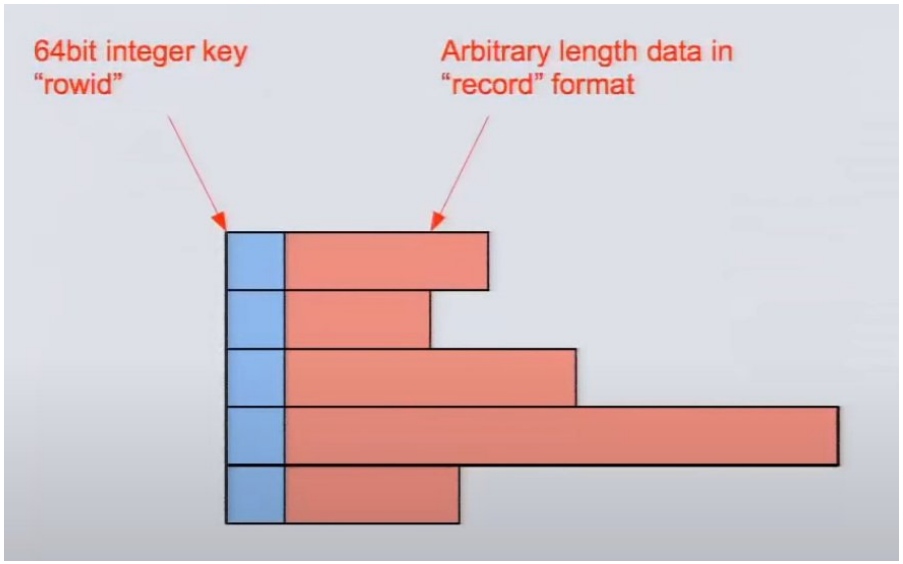
Архитектура SQLite



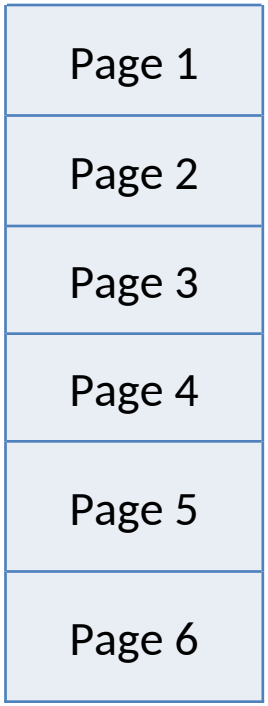
Внешняя память
Чтение/запись страницами

rowid	fruit	state	price
1	Orange	FL	0.85
2	Apple	NC	0.45
4	Peach	SC	0.60
5	Grape	CA	0.80
18	Lemon	FL	1.25
19	Strawberry	NC	2.45
23	Orange	CA	1.05

Key Data



Логическое представление таблицы



Физическая структура БД

Какую структуру данных лучше выбрать для представления таблицы в БД?

Критерий: минимальное количество операция чтения/записи страниц при поиске и изменения данных в таблице.

Вспоминаем файловую систему...

Связный список?

C:\Документы\Мой файл.doc

Каталог файлов

Дескриптор файла 1
…
Дескриптор файла N

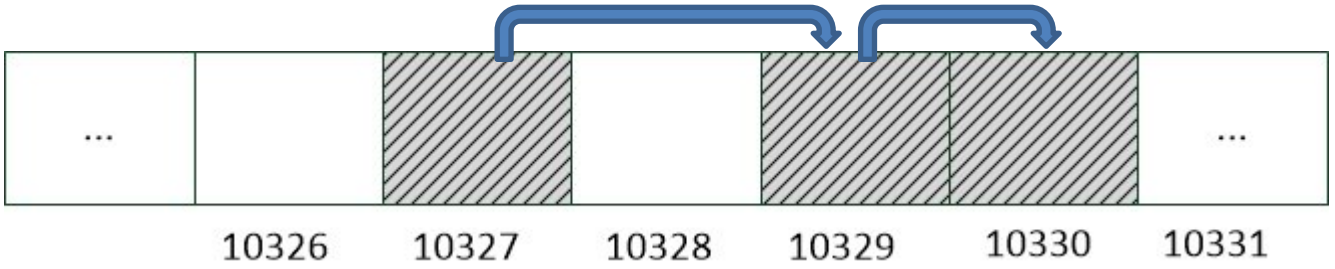
Дескриптор файла

Имя	*Мой файл.doc*
Дата создания	*01.02.2020*
Атрибуты	*Archive*
Первый кластер	*10327*
Размер	*9326*
…	…

Таблица размещения файлов (FAT)

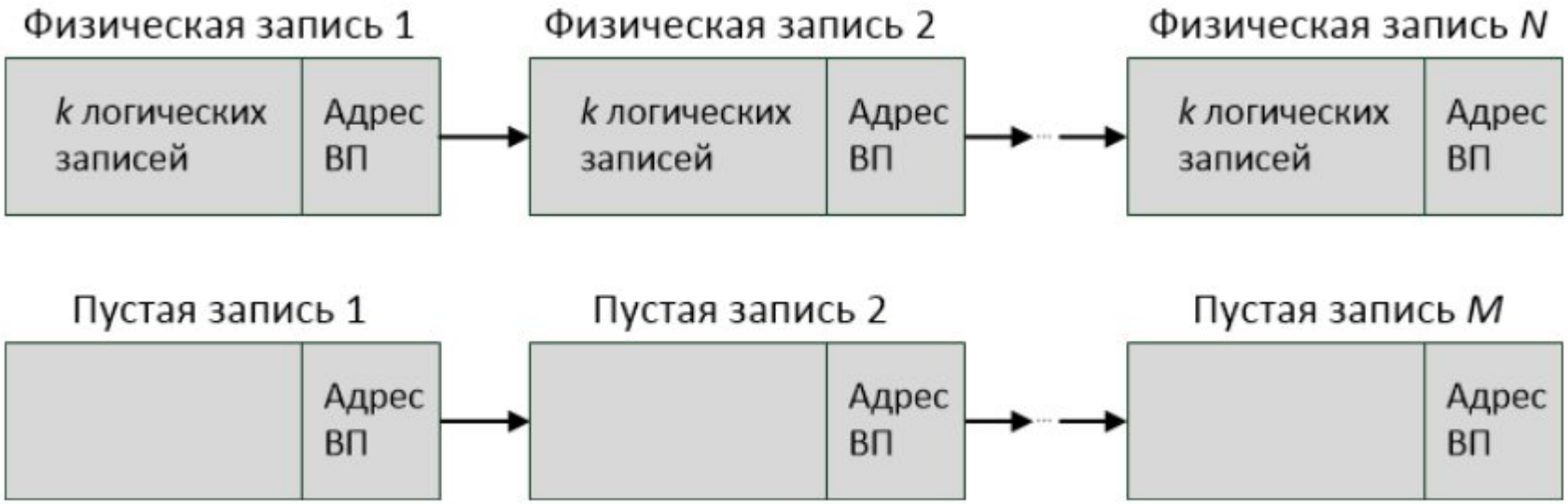
№ кластера	Статус
10326	*Сбойный*
10327	*10329*
10328	*Свободный*
10329	*10330*
10330	*Конец цепочки*
10331	*Свободный*
…	…

Кластеры в файле организованы в **связный список**



Кластеры на диске

Страницы для таблицы в виде списка



Физическая запись = страница
Адрес ВП = номер страницы

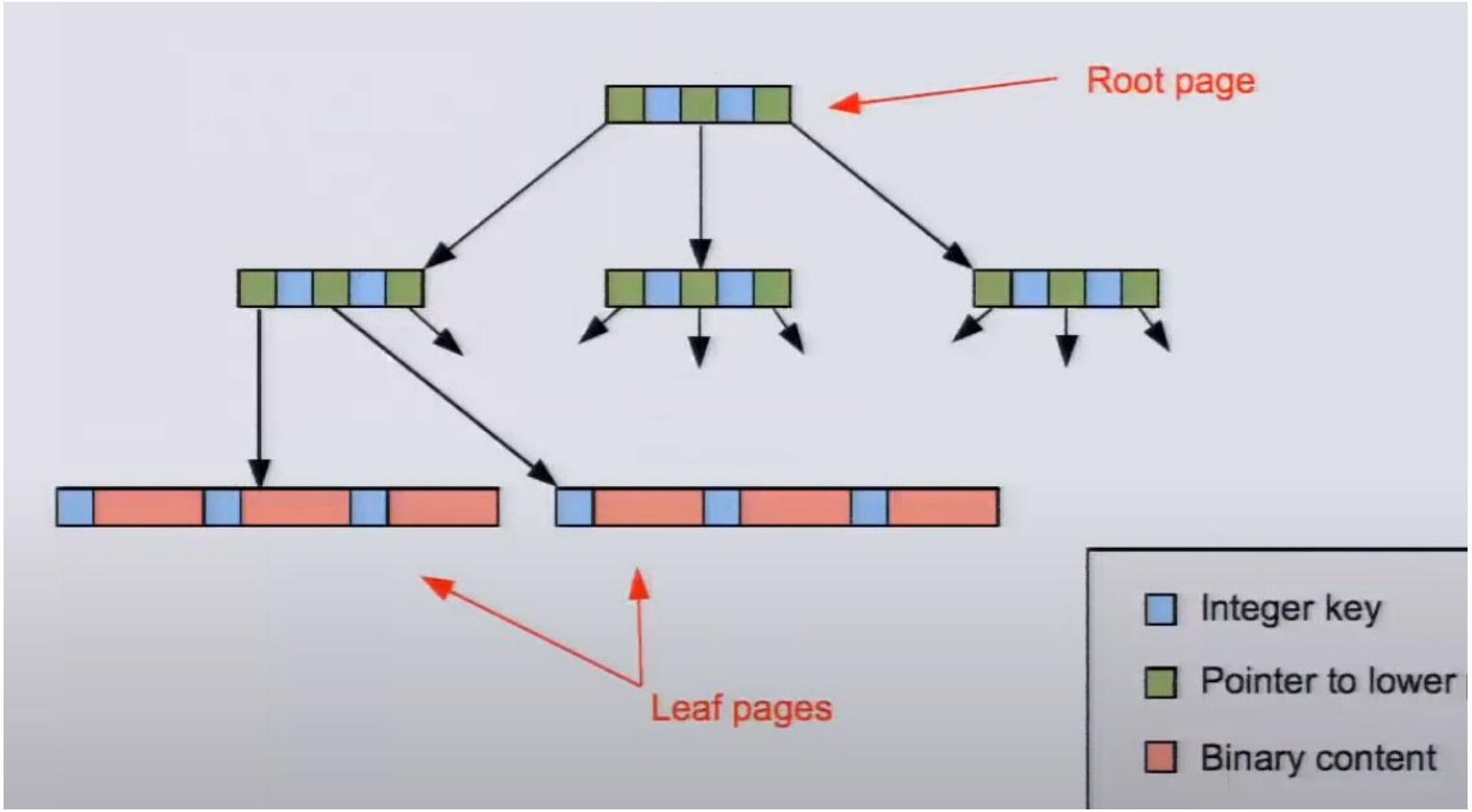
Страницы для таблицы в виде списка

Оценим T – среднее число обращений к ВП.

- **Поиск записи по ключу.** **$T=(1+[N/k])/2=O(N)$**
  - **Чтение записи.** $T=(1+[N/k])/2=O(N)$
  - **Корректировка записи.** $T=(1+[N/k])/2+1=O(N)$
  - **Удаление записи.** $T=(1+[N/k])/2+1=O(N)$

Проблемная операция: **поиск записи**

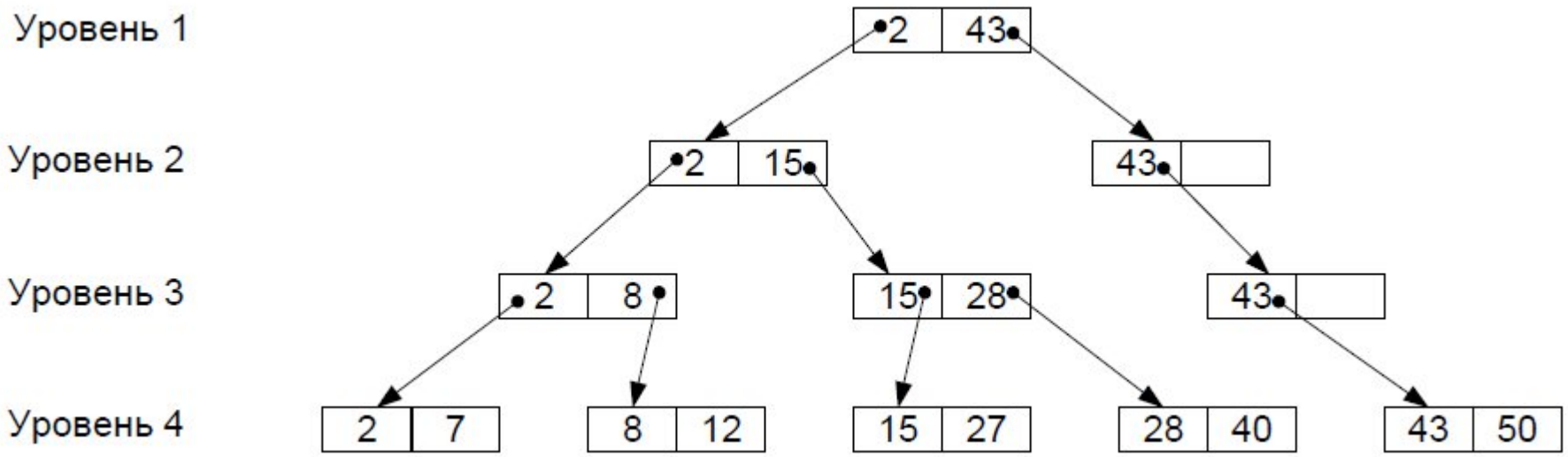
B-tree структура для страниц таблицы



B-дерево – это сбалансированное сильно ветвистое дерево поиска, в котором каждый узел содержит множество ключей и имеет более двух потомков.

Количество ключей в узле и количество его потомков зависит от порядка ***m*** B-дерева.

- Каждый узел имеет не более ***m*** ветвей.
- Каждый некорневой неконечный узел (не лист) имеет не менее ***m/2*** ветвей.
- Неконечный узел, у которого ***k*** ветвей, содержит ***k*-1** ключей.
- Ключи в элементах упорядочены.

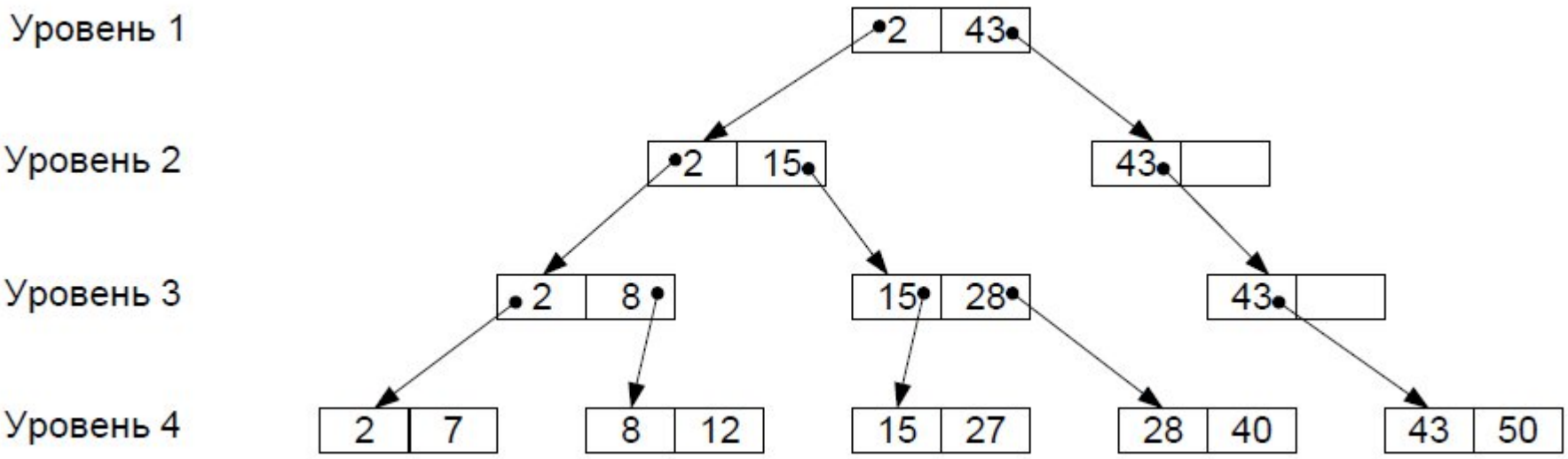


***m*=2**

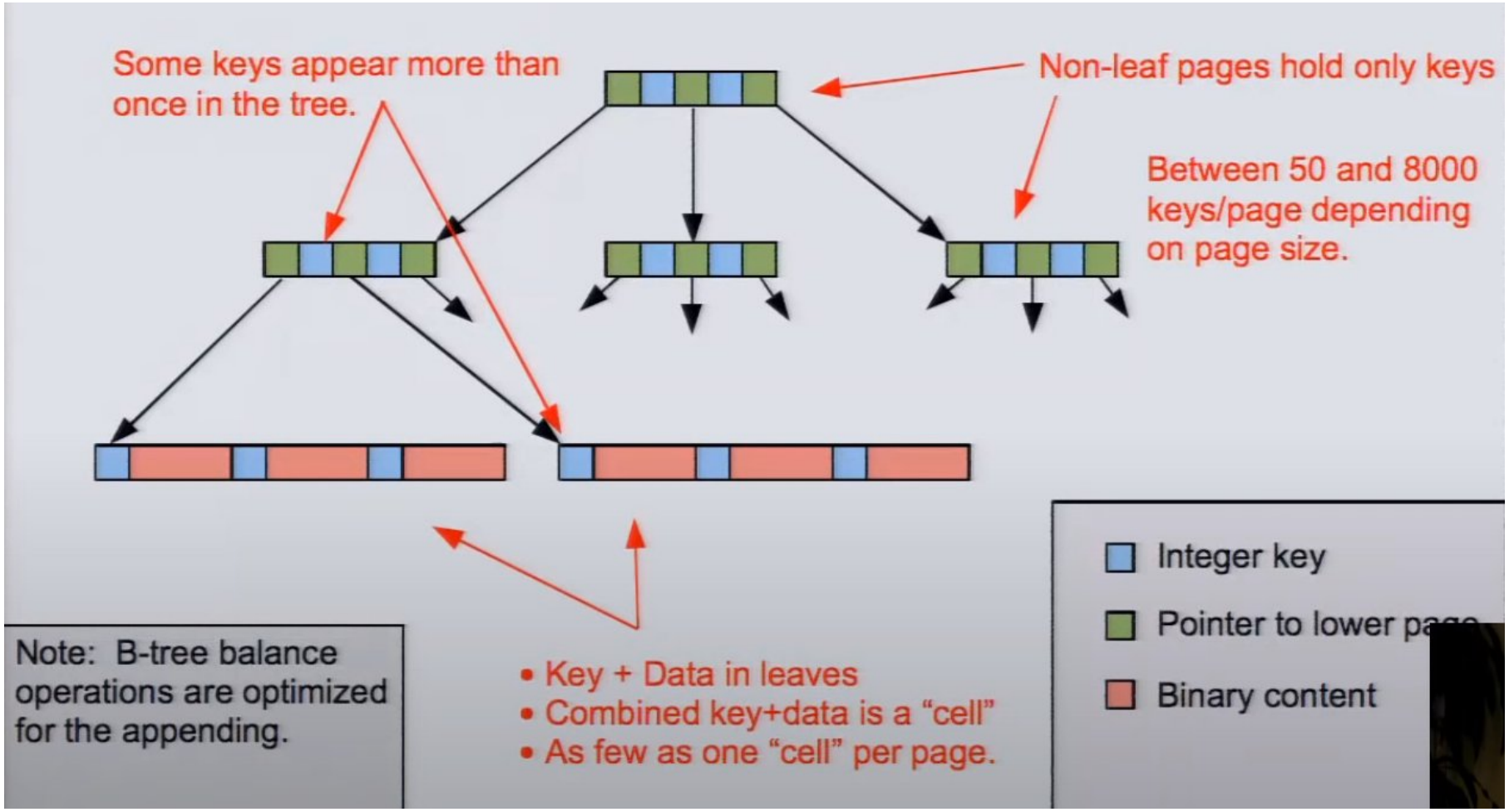
Поиск записи с заданным значением M ключа в B-дереве

1. Читаем верхний индекс.
2. Сравниваем число M со значением ключа записей индекса, начиная с большего значения. Если M больше или равно значению ключа очередной записи индекса, то по адресу связи, указанному в текущей записи, читаем блок записей индекса следующего уровня.
3. Повторяем процесс, пока не дойдем до последнего уровня.

Количество считываний узлов $T=O(\log_2 N)$

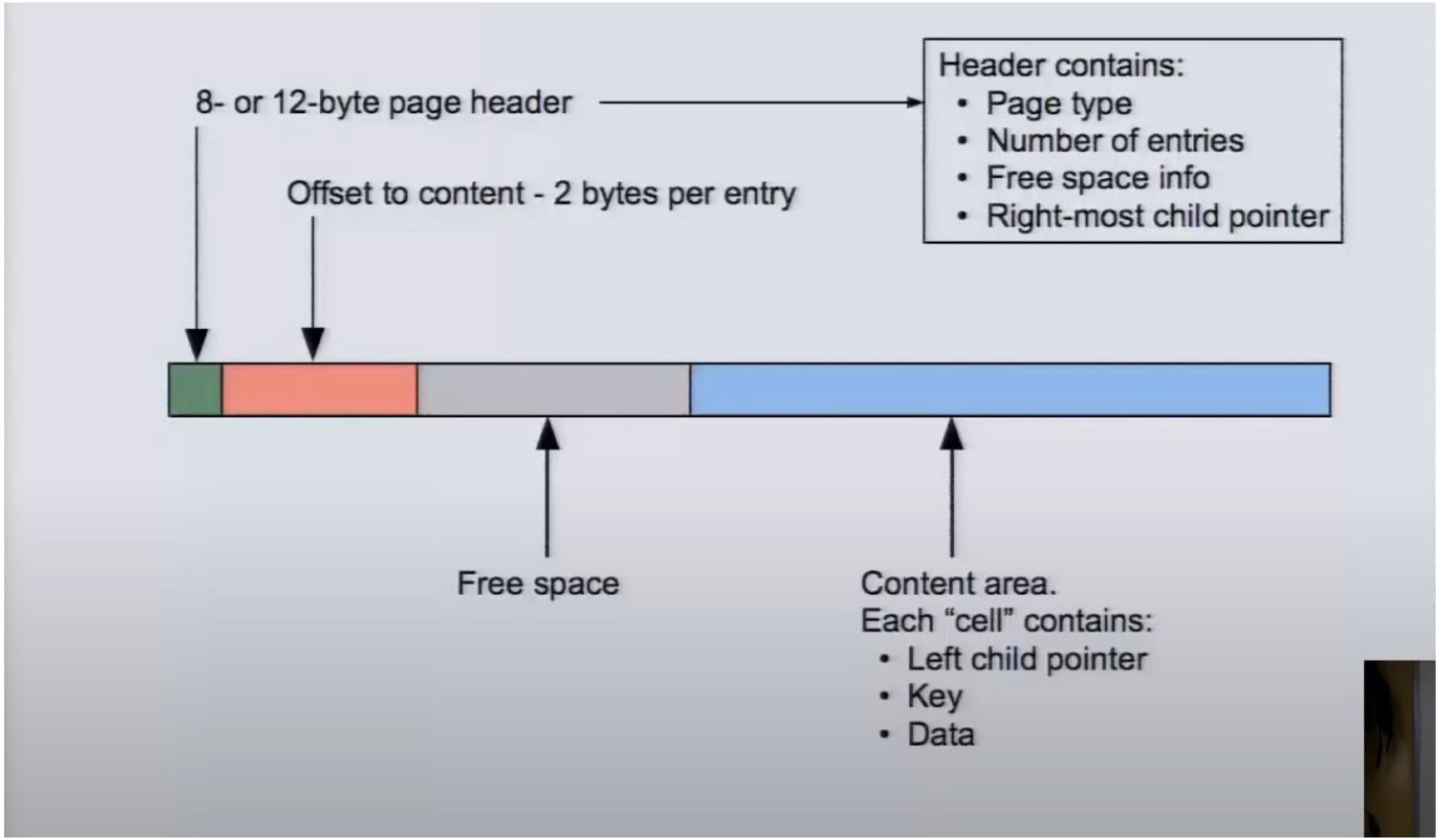


B-tree структура для страниц таблицы



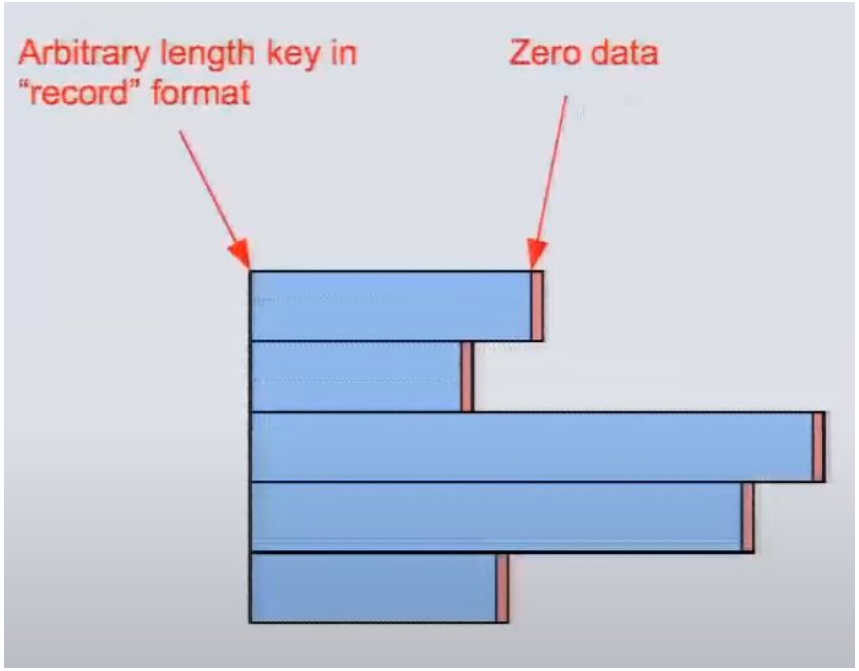
Узел = страница

Структура страницы в B-дереве



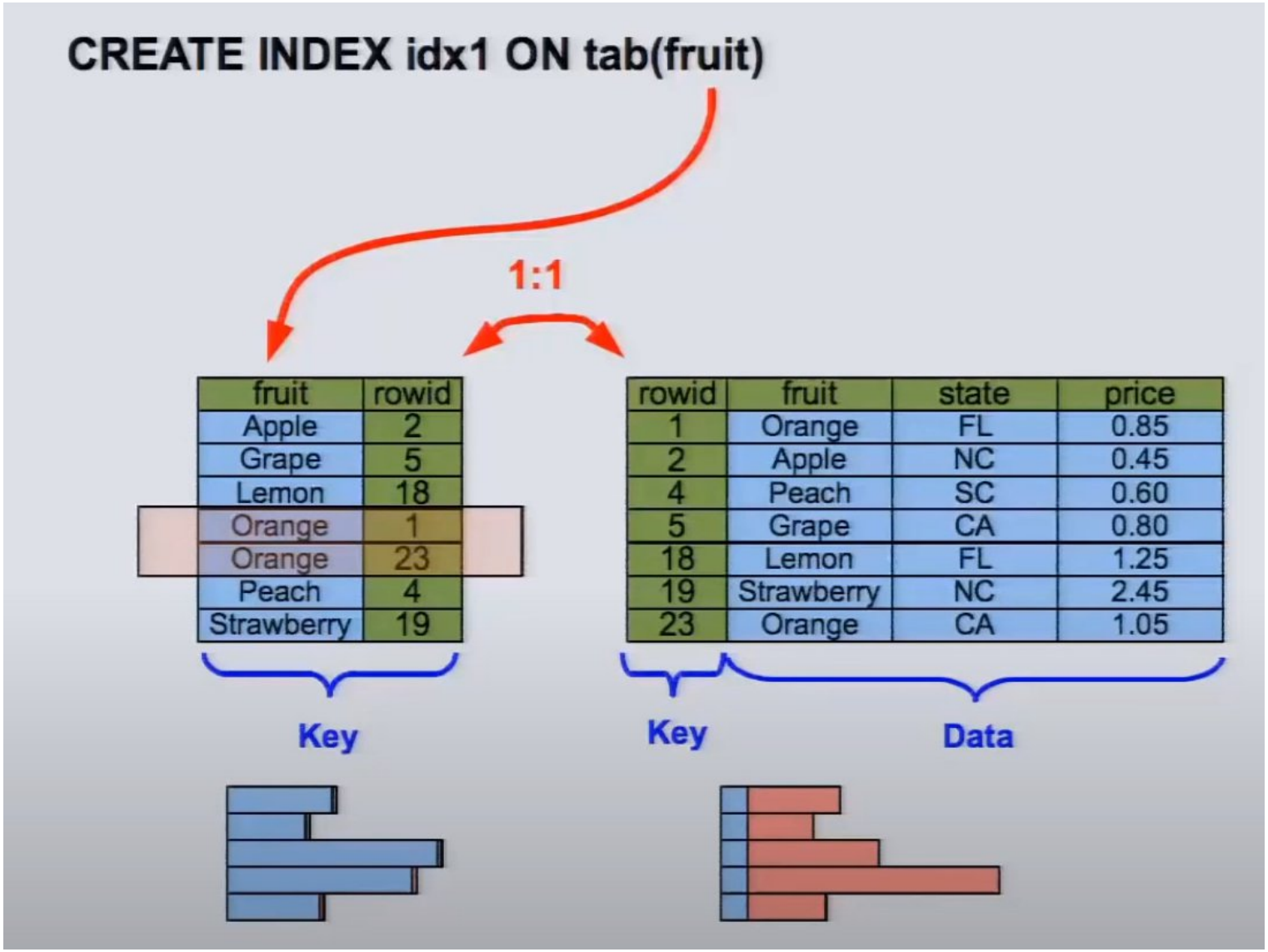
Логическая структура индекса

rowid	fruit	state	price
1	Orange	FL	0.85
2	Apple	NC	0.45
4	Peach	SC	0.60
5	Grape	CA	0.80
18	Lemon	FL	1.25
19	Strawberry	NC	2.45
23	Orange	CA	1.05

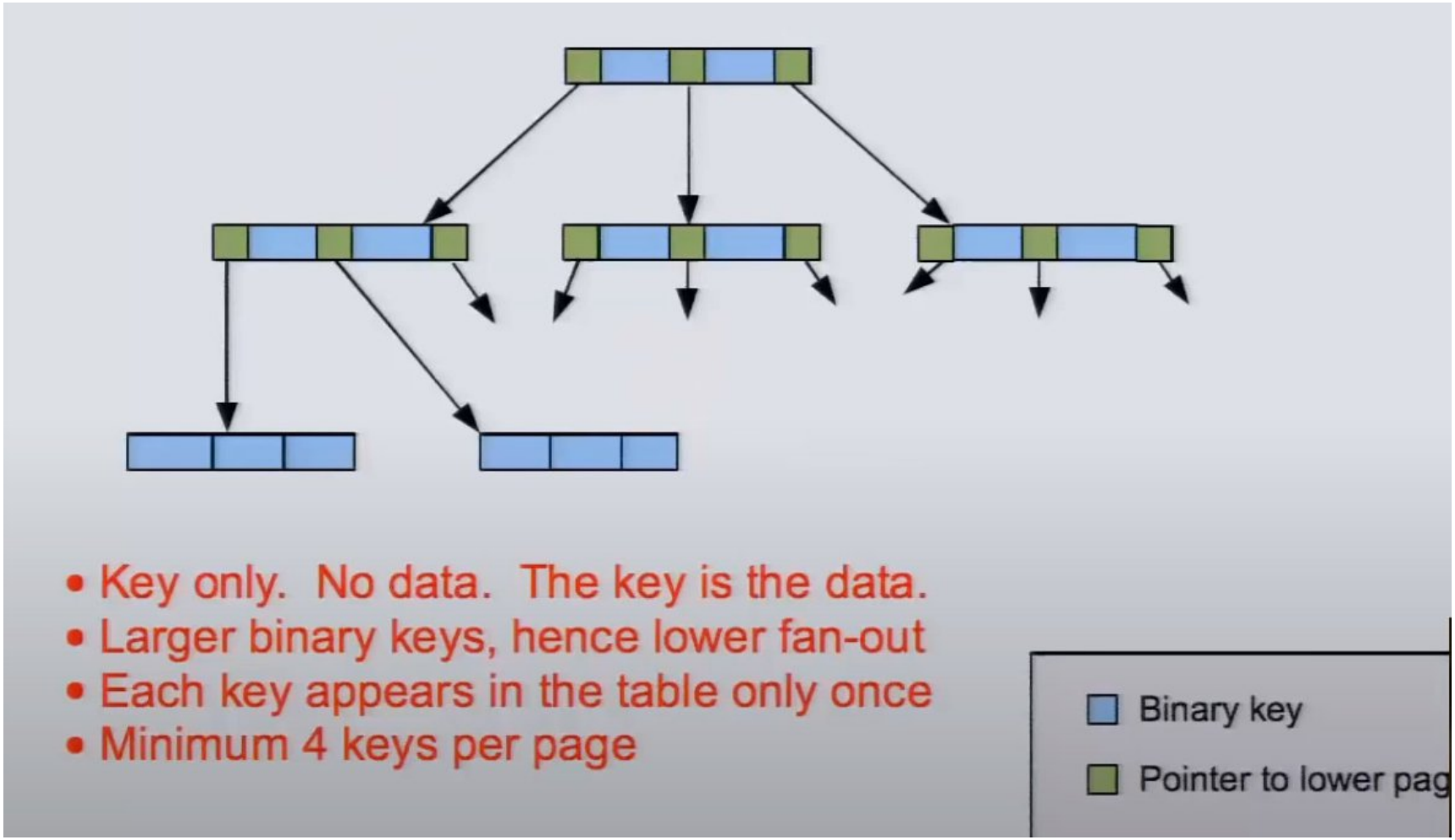


CREATE INDEX Idx1 ON fruits(fruit);

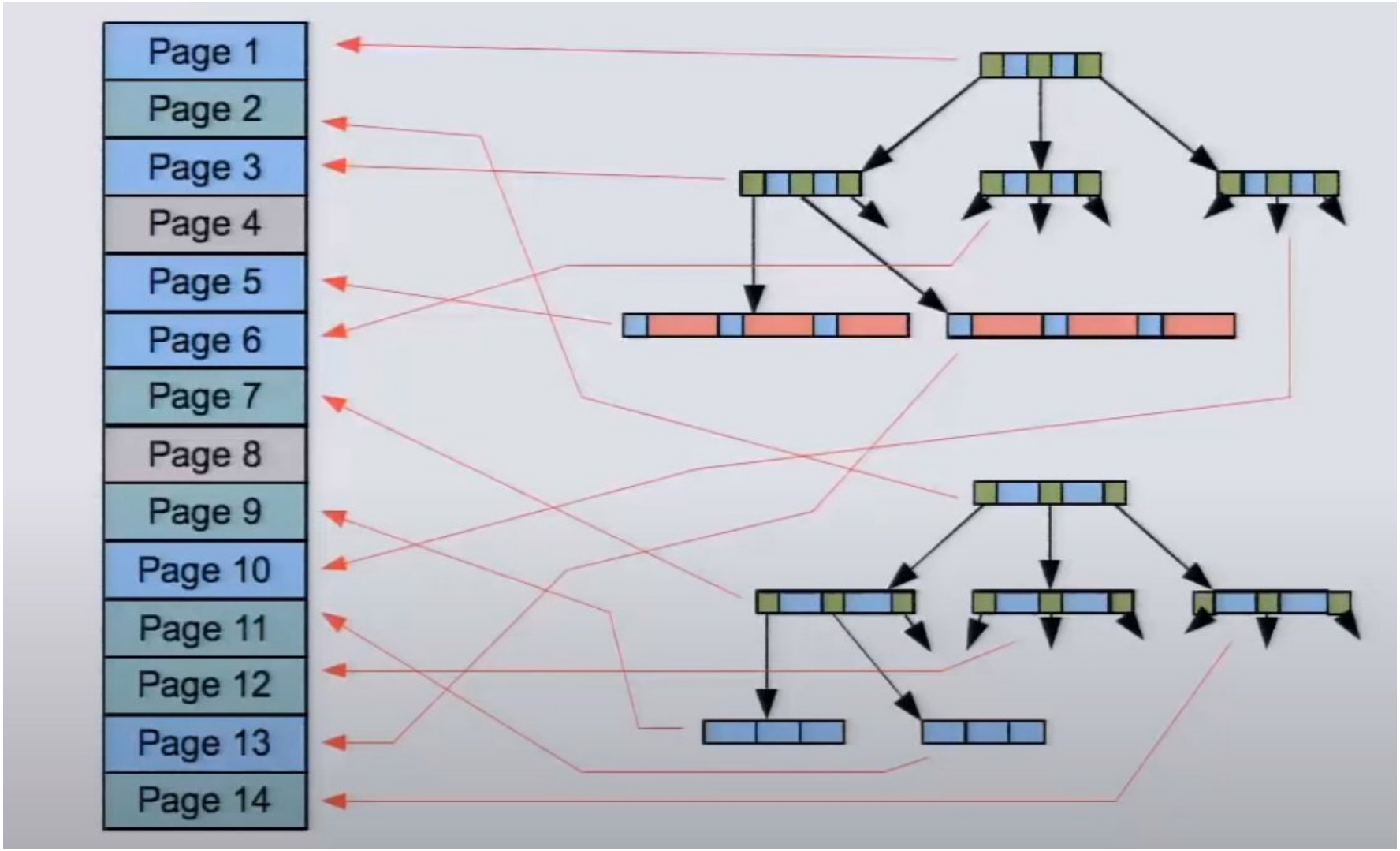
fruit	rowid
Apple	2
Grape	5
Lemon	18
Orange	1
Orange	23
Peach	4
Strawberry	19


CREATE INDEX idx1 ON tab(fruit)
1:1
fruit
rowid
Apple
2
Grape
5
Lemon
18
Orange
1
Orange
23
Peach
4
Strawberry
19
rowid
fruit
state
price
1
Orange
FL
0.85
2
Apple
NC
0.45
4
Peach
SC
0.60
5
Grape
CA
0.80
18
Lemon
FL
1.25
19
Strawberry
NC
2.45
23
Orange
CA
1.05
Key
Key
Data

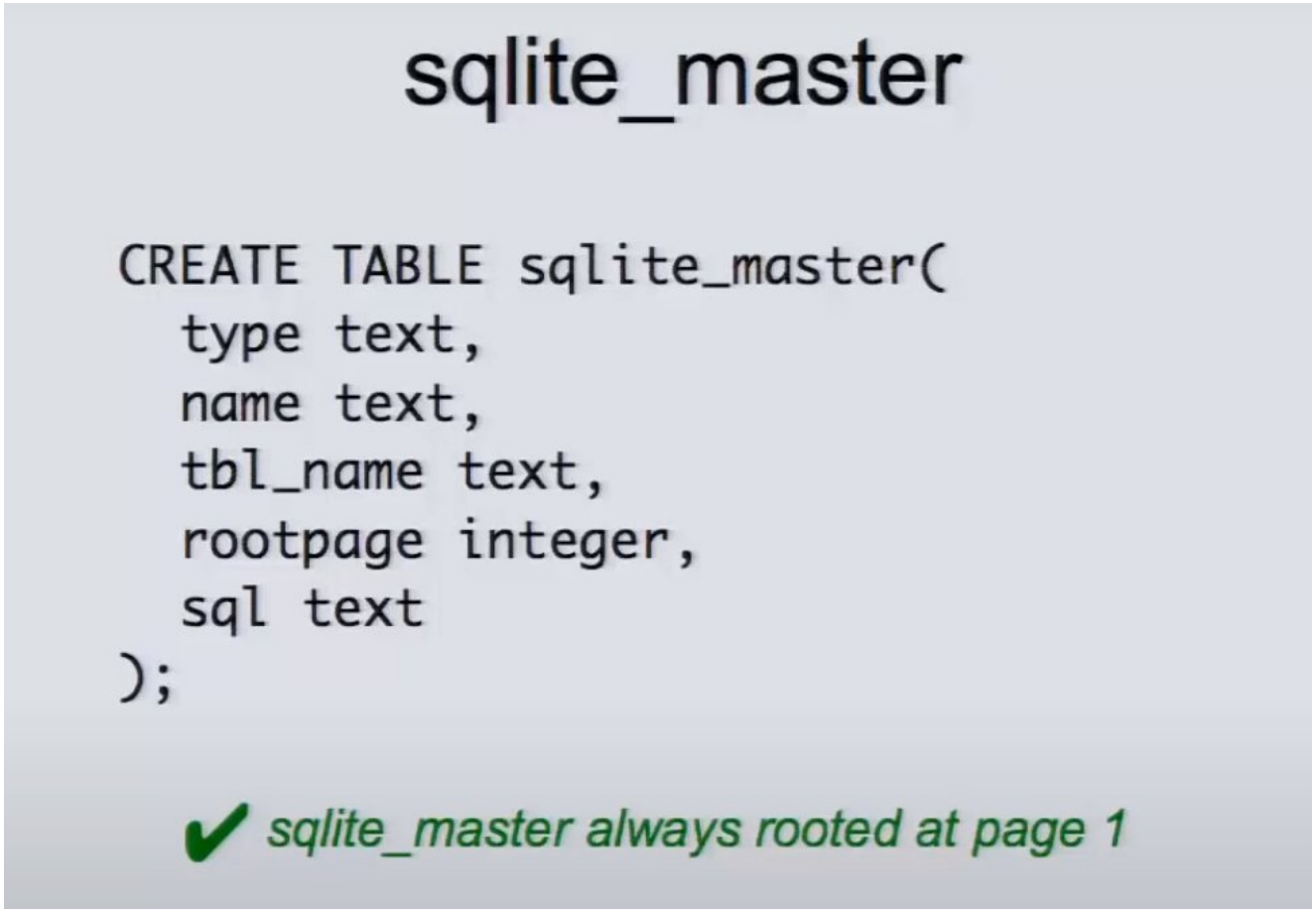
B-tree структура для страниц индекса



Соответствие между B-деревьями и страницами



```
Available pages: 1..1146
     1: root leaf of table [sqlite_master]
     2: root interior node of table [blob]
     3: root interior node of index [sqlite_autoindex_blob_1]
     4: root interior node of table [delta]
     5: root interior node of table [rcvfrom]
     6: root leaf of index [sqlite_autoindex_rcvfrom_1]
     7: root leaf of table [config]
     8: root leaf of index [sqlite_autoindex_config_1]
     9: root leaf of table [shun]
    10: root leaf of index [sqlite_autoindex_shun_1]
    11: root leaf of table [private]
…
   264: leaf of table [blob], child 201 of page 2
   265: leaf of table [blob], child 202 of page 2
   266: overflow 1 from cell 0 of page 268
   267: overflow 2 from cell 0 of page 268
   268: leaf of table [blob], child 203 of page 2
…
```

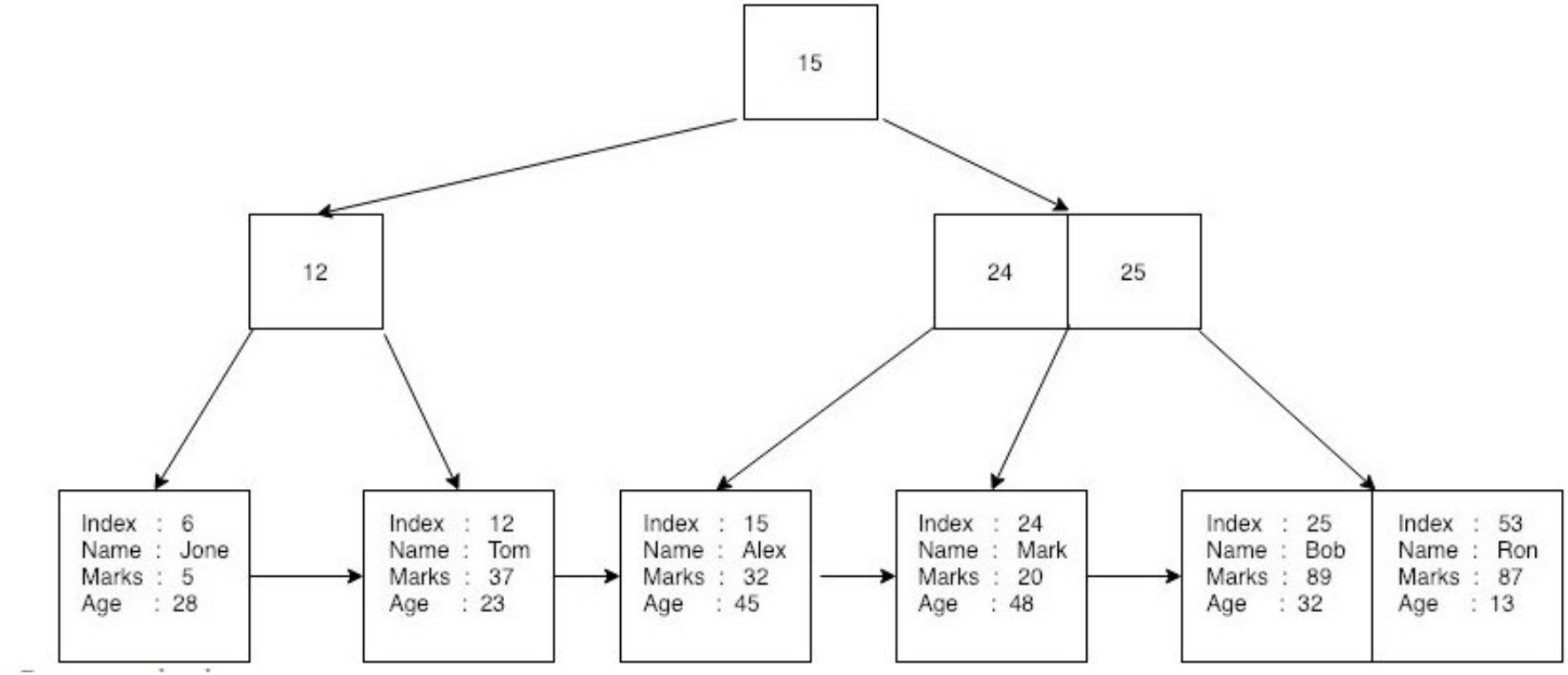


Альтернативные имена: sqlite_schema, sqlite_temp_master, sqlite_temp_master

rowid	Name	Marks	Age
6	Jone	5	28
12	Tom	37	23
15	Alex	32	45
24	Mark	20	48
25	Bob	89	32
53	Ron	87	13

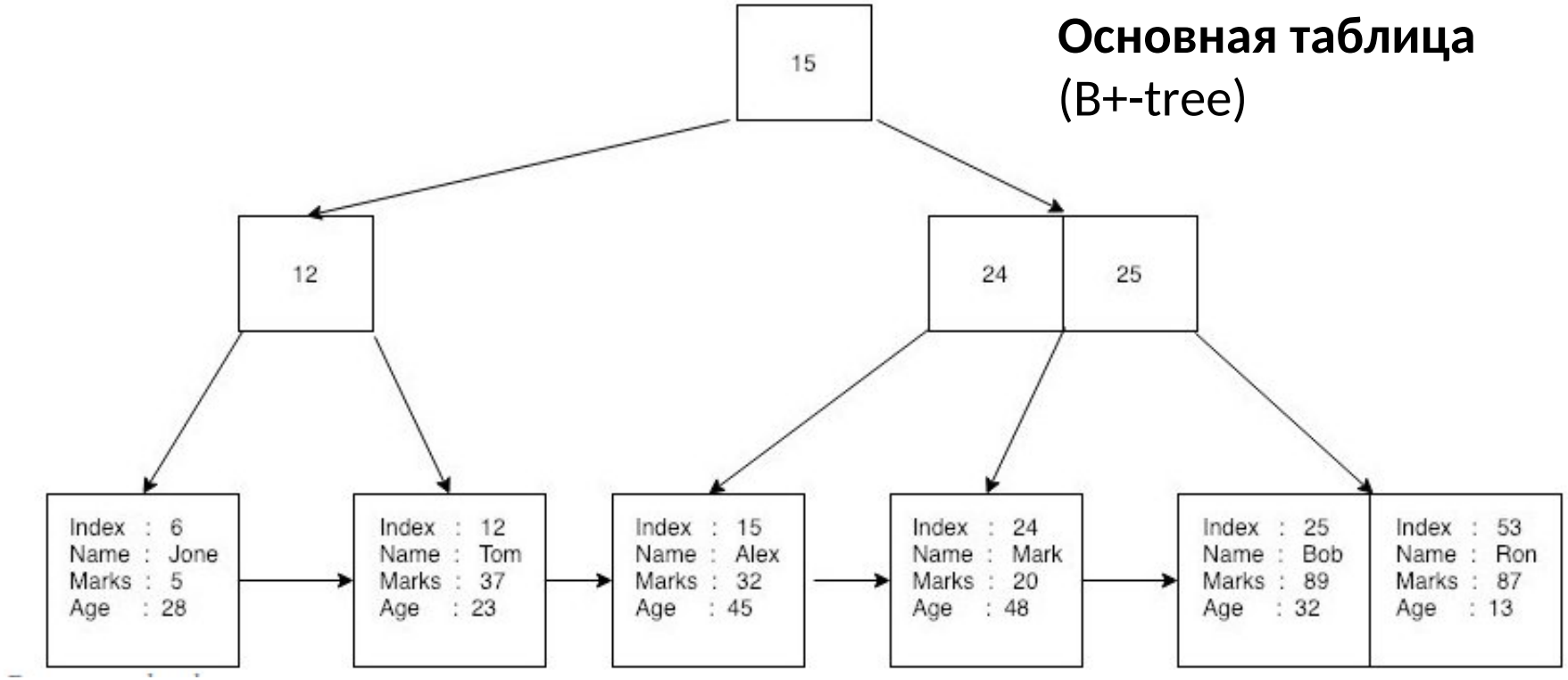
Поиск в таблице:

- Двоичный поиск по индексу
- Последовательный поиск по листьям

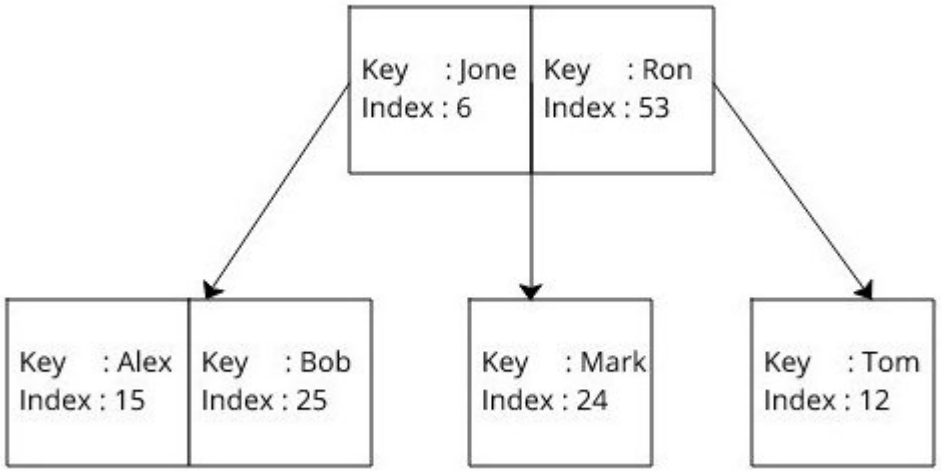


Каждый лист ссылается на следующий лист в дереве

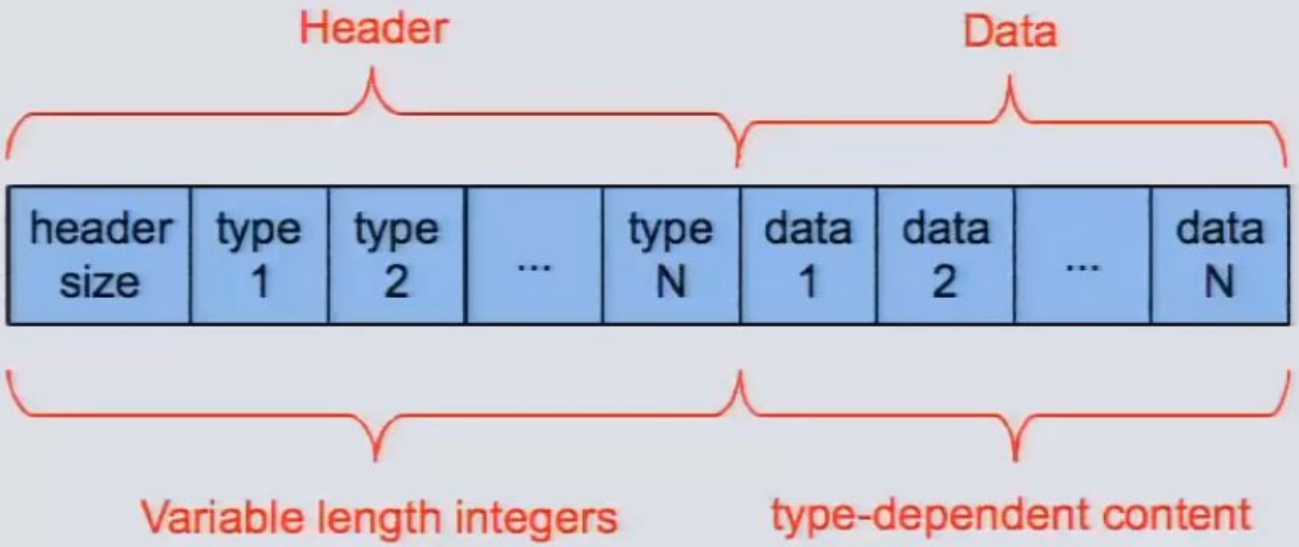
Основная таблица
(B+-tree)



Индекс по имени
(B-tree)

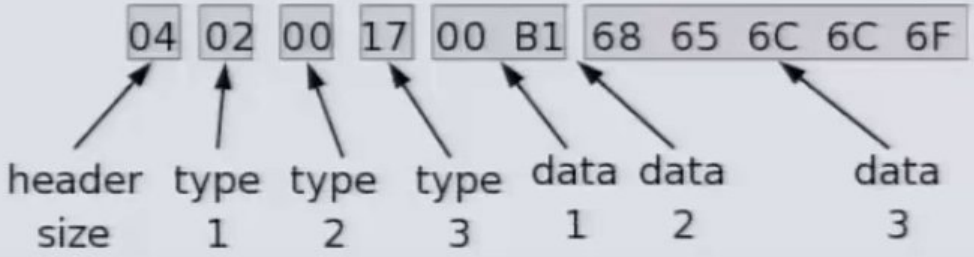


Record Format



Record Format Example

```
CREATE TABLE t1(a,b,c);
INSERT INTO t1 VALUES(177, NULL, 'hello');
```



✔ *Column datatypes are optional*
✔ *Datatypes are suggestions, not requirements.*