4.2. ПОСЛІДОВНІ ФАЙЛИ: МЕТОДИ ОБРОБКИ ТА ПРИСКОРЕННЯ ПОШУКУ. ІНВЕРТОВАНІ ФАЙЛИ

В послідовних файлах записи розташовані послідовно один за одним. В одних програмних системах використовуються роздільники записів, в інших, при фіксованих довжинах полів, роздільники можуть не використовуватися. Пошук інформацій в самому узагальненому випадку здійснюється теж послідовно. При початку роботи файлу вказівник номера запису має значення «1» або «0» в залежності від прийнятої системи нумерації. При наявності запису з потрібним значення поля (полів) у файлі час пошуку в середньому є лінійна функція від кількості записів `N`, точніше – `N/2`, а при відсутності – `N`. Для прискорення пошуку в послідовних файлах використовують різни методи. Розглянемо деякі з них.

а) *Упорядкування (сортування) записів за значенням одного або сукупності полів.* В цьому випадку як при наявності потрібного запису, так і при відсутності, час пошуку пропорційний `N/2`. Це вірно тільки для тих випадків, коли пошук здійснюється за упорядкованим полем. Для того, щоб була можливість здійснювати пошук і за іншими полями, треба створювати «братів» цих файлів, відсортувавши основний файл за потрібним полем. Такі файли називаються інвертованими відносно основного файлу. Деякі з ранніх систем керування базами даних мали у своїй концепції цей підхід.

б) *Статистичне упорядкування записів.* Це можна продемонструвати на прикладі телефонного довідника. Після створення файлу-довідника можна підрахувати питому вагу записів для кожної літери, з яких починаються прізвища абонентів і закласти це в програму. Тоді, наприклад, пошук абонента, прізвище якого починається з літери «К» можна починати із запису, який розташований на певній відстані від початку, знаючі питому вагу записів, що починаються з букв, що передують «К».

в) *Імовірнісне упорядкування.* В результаті статистичних спостережень для кожного поля визначається імовірність звернення до нього, після чого файл сортується за зменшенням імовірності звернення.

г) *Бінарний пошук.* Він може бути застосований тільки для упорядкованих файлів. Суть його полягає в тому, що спочатку вказівник записів встановлюється на середину файлі і значення шуканого поля порівнюється з крайніми елементами напівфайлів, щоб визначити, в

якій половині треба продовжити пошук. Так цей процес продовжується до тих пір, доки не отримуємо відповідь, є чи нема шуканого запису. Час пошуку пропорційний $\log_2 N$. Цей метод є досить ефективним.

д) *Пошук з використанням B-дерев. B-дерево* – це m-арне дерево, на кожному рівні якого є однакова кількість вузлів, а між коренем дерева і будь-яким його листом однакова кількість рівнів. Пошук з використання такого дерева нагадує за алгоритмом пошук по бінарному дереву. Це основний метод пошуку на фізичному рівні, прийнятий в базах даних. Справа в тому, що для файлу з великою кількістю записів бінарне дерево теж може бути великим, тому пошук може бути довгим.

Тепер розглянемо, як здійснюються операції додавання, редагування і видалення інформації. При реалізації вказаних операцій виходять із того, що ПФ мають великі обсяги, звідси – «перебудова» таких файлів є небажаною. Тому записи, що додаються, дописуються в кінець файлу, а записи, що видаляються – помічаються, але фізично не видаляються. Оновлення (редагування) здійснюється за місцем знаходження поля (полів). Фізичне видалення логічно видалених (помічених для видалення) записів, а також перебудова (нове упорядкування) файлу з врахуванням доданих записів проводиться за певним регламентом.

Як було згадано вище пошук інформації у сортованому файлі відбувається ефективно, але сам процес сортування є достатньо довготривалою операцією і потребує додатково дискової пам'яті від 1/2 до 1 довжини файлу, який сортується. Нажаль, файл можна відсортувати тільки за одним полем.

Як вихід, можна застосувати так звані зв'язні списки (не плутати із динамічними списками), які дозволяють виконати логічно упорядкування елементів послідовності, які у загальному випадку не упорядковані фізично. Для створення зв'язного списку до кожного запису додаються додаткові поля – поля посилань. Ці поля містять адресу (у нашому випадку – відносний номер) наступного запису в логічній послідовності. Таким чином файл може бути «відсортований» по декільком полям, хоча фізично файл має структуру, яка була сформована при його формуванні.

Приклад упорядкування файлу списку студентів та дисциплін, які вони вивчають по двом полям: номер студента та дисципліна.

Відносний номер запису у файлі	Номер студента	Номер дисципліни	Посилання на студента	Посилання на дисципліну
1	200	70	4	5
2	100	30	6	1
3	300	20	5	4
4	200	30	3	2
5	300	70	0	0
6	100	20	1	3

Початок списку студентів – 2;
Початок списку дисциплін – 6;
0 – означає кінець списку.

В такий спосіб можна організувати «сортування» файлу по декільком полям. Вставка і видалення елементів відбувається достатньо просто на рівні зміни відносних адрес посилань.