

ФАЙЛОВАЯ СТРУКТУРА

СОДЕРЖАНИЕ

2.1	СТРУКТУРА ФАЙЛОВ В ОС РІСК.....	2-3
2.2	ДОСТУП К ФАЙЛАМ.....	2-6
2.3	СЛОВАРИ.....	2-8
2.4	СЛОВАРИ, КОТОРЫМ СООТВЕТСТВУЕТ НЕСКОЛЬКО ФАЙЛОВ ДАНЫХ	2-11
2.5	БАЗА И МОДУЛО.....	2-13
2.6	ВЫБОР МОДУЛО.....	2-15
2.7	ФИЗИЧЕСКАЯ СТРУКТУРА ЗАПИСИ.....	2-17
2.8	ЛОГИЧЕСКАЯ СТРУКТУРА ЗАПИСИ.....	2-19
2.9	ХРАНЕНИЕ ЗАПИСИ НА ДИСКЕ И АЛГОРИТМ ХЕШИРОВАНИЯ.....	2-21
2.10	ЗАПИСИ-ОПРЕДЕЛИТЕЛИ ФАЙЛОВ.....	2-23
2.11	ЗАПИСИ-СИНОНИМЫ ОПРЕДЕЛИТЕЛЕЙ ФАЙЛОВ.....	2-25
2.11.1	Q-УКАЗАТЕЛИ: РЕФЛЕКСИВНАЯ ФОРМА.....	2-27
2.11.2	Q-УКАЗАТЕЛИ: СПЕЦИФИКАЦИЯ СЧЕТА.....	2-28
2.11.3	Q-УКАЗАТЕЛИ: СПЕЦИФИКАЦИЯ ФАЙЛА.....	2-28
2.11.4	Q-УКАЗАТЕЛИ: СПЕЦИФИКАЦИЯ РАЗДЕЛА ДАНЫХ ФАЙЛА.....	2-28
2.12	ЗАПИСИ-ОПРЕДЕЛИТЕЛИ АТРИБУТОВ.....	2-29
2.13	СЛОВАРНЫЕ ЗАПИСИ: ЗАКЛЮЧЕНИЕ.....	2-31
2.14	СИСТЕМНЫЕ ФАЙЛЫ И СЛОВАРИ.....	2-33
2.15	ПРОЦЕССОРЫ УПРАВЛЕНИЯ ФАЙЛАМИ.....	2-35
2.16	СОЗДАНИЕ НОВЫХ ФАЙЛОВ: ПРОЦЕССОР CREATE-FILE.....	2-36
2.17	ПРОЦЕССОР CLEAR-FILE.....	2-38
2.18	ПРОЦЕССОР DELETE-FILE.....	2-39
2.19	КОПИРОВАНИЕ: ПРОЦЕССОР COPY.....	2-41
2.20	КОПИРОВАНИЕ ДАНЫХ ИЗ ФАЙЛА В ФАЙЛ.....	2-42
2.21	КОПИРОВАНИЕ: ОПЦИИ ПРОЦЕССОРА COPY.....	2-44

2.1. СТРУКТУРА ФАЙЛОВ В ОС РІСК

В настоящем разделе описывается иерархическая структура файлов в ОС РІСК.

Файлы в ОС РІСК организованы иерархически: каждый файл более высокого уровня указывает на один или несколько файлов более низкого уровня. Всего существует 4 уровня файловой иерархии - системный словарь (SYSTEM), главные словари пользовательских счетов (MD), словари файлов и файлы данных.

Под "файлом" в ОС РІСК понимается механизм организации однородных записей в единое логическое целое. Доступ к хранящимся в файле данным происходит через словарный раздел (словарь) этого файла. Словарь файла - это некоторая надстройка над записями, содержащими данные, своего рода "предметный указатель". Файлы, с которыми работает пользователь, имеют двухуровневую структуру (за исключением главного словаря счета), т. е. состоят из словаря и одного или нескольких разделов данных, но элементы разных уровней файла также представляют собой файлы, т.е. группы логически объединенных записей. Словарные записи определяют структурные элементы ("поля" или "атрибуты") записей из раздела данных того же файла.

ОС РІСК может поддерживать неограниченно большое количество файлов. Каждый файл может содержать произвольное число записей. Размер файла при его создании не фиксируется: по мере добавления в него новых записей файлы могут разрастаться до любых размеров. Единственное ограничение на размер файла - объем диска. Каждая запись в файле может иметь произвольную длину и содержать любое количество полей. Размер записи не может превышать 32267 байт.

СИСТЕМНЫЙ СЛОВАРЬ (SYSTEM)

На верхнем уровне файловой иерархии в ОС РІСК находится системный словарь (SYSTEM). В качестве записей этого словаря выступают указатели на пользовательские счета (так называемые записи - определители счетов", названия которых совпадают с именами пользовательских счетов). Записи - определители счетов содержат такую информацию, как пароли, коды доступа к записям в счете, уровень системных привилегий и адрес главного словаря соответствующего счета на диске.

ГЛАВНЫЕ СЛОВАРИ ПОЛЬЗОВАТЕЛЬСКИХ СЧЕТОВ (MD)

Главные словари (MD) пользовательских счетов занимают следующий - второй - уровень файловой иерархии. В каждом счете может быть только один MD. Названия записей в главном словаре определяют круг доступных пользователю системных команд (глаголов, процедур, и т.д.), служебные слова языка ACCESS, а также файлы, доступные из данного счета (так называемые "записи-определители файлов").

При создании пользовательского счета в его MD автоматически записывается стандартный набор команд, доступный всем пользователям. Имена этих команд можно произвольно менять, создавая в MD записи-синонимы. Пользователь может самостоятельно расширять набор доступных ему команд, создавая собственные процедуры и занося их в MD в виде записей под соответствующими именами. Помимо команд, процедур и служебных слов языка

доступных ему команд, создавая собственные процедуры и занося их в MD в виде записей под соответствующими именами. Помимо команд, процедур и служебных слов языка ACCESS в главном словаре счета хранятся также указатели на файлы, которые могут

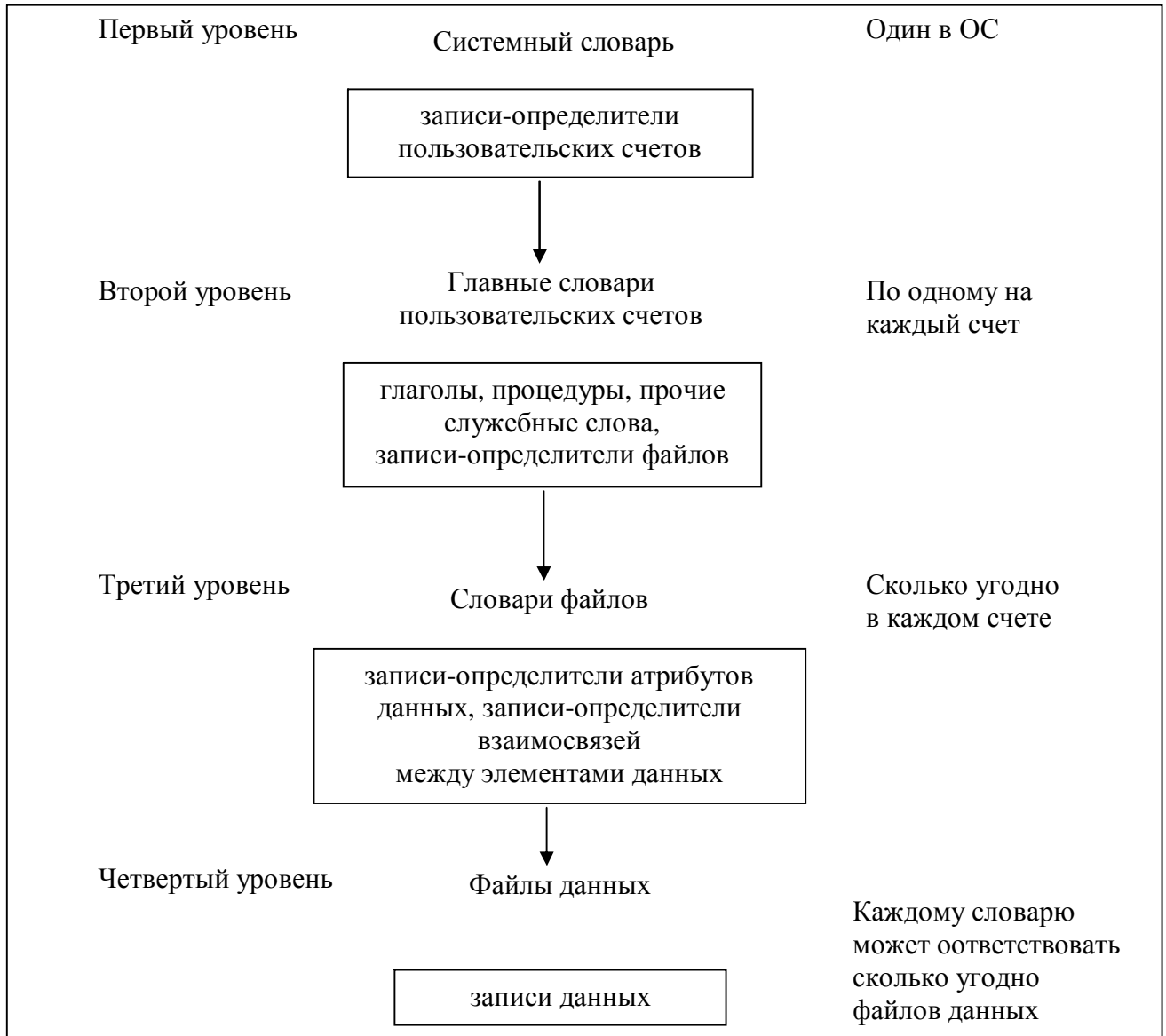
ссылаться на любой файл в системе (необязательно только на те, которые созданы в данном счете).

СЛОВАРИ ФАЙЛОВ

Словари файлов содержат записи, описывающие структуру данных в соответствующих файлах данных, а также указатели на эти файлы данных. Поскольку на уровне пользователя (пользовательского счета) доступ осуществляется в основном к двухуровневым файлам, занимающим последние два уровня в файловой иерархии, во всех случаях, где это не оговорено особо, мы будем понимать под словом "файл" именно такую двухуровневую структуру, т.е. словарь и один или несколько файлов данных, и говорить о словарном разделе и о разделе данных этого файла.

ФАЙЛЫ ДАННЫХ

В файлах данных (разделах данных двухуровневого файла) хранятся собственно данные, организованные в записи произвольной длины. Записи данных подразделяются на поля произвольной длины, именуемые в ОС RISC "атрибутами". Каждый атрибут записи может содержать одно или несколько значений (соответственно, мы будем говорить об "однозначных" и "многозначных" атрибутах), а каждое значение - любое количество подзначений. Таким образом, всякая запись в ОС RISC логически может быть представлена в виде трехмерного массива (матрицы), каждый элемент которой имеет три индекса - номер (или имя) атрибута, номер значения и номер подзначения. Все элементы записи могут иметь произвольную длину.



Четырехуровневая иерархия файлов в ОС РІСК

2.2. ДОСТУП К ФАЙЛАМ

Система доступа к содержимому файлов позволяет получить доступ к одной, нескольким или всем записям в файле одновременно.

Под "файлом" в связывающая набор записей в одно целое и позволяющая получить доступ к этим записям для чтения и внесения в них изменений.

Все данные в файле могут иметь разную длину, которая не должна, однако, превышать 32267 байт. Никаких ограничений на число записей в файле нет, также как нет никаких ограничений на число файлов в каждом счете. Все записи в файле различаются по своим именам ("идентификаторам" в терминологии ОС PICK), которые среди записей этого файла должны быть неповторяющимися.

Записи хранятся в файле в "псевдослучайной" последовательности. Местонахождение их внутри файла определяется методом хэширования, основанном на использовании идентификатора записи для получения псевдослучайного числа, по которому определяется номер группы фреймов, в которую попадает эта запись.

Доступ к записям в файле может быть либо прямым, если указываются идентификаторы требуемых записей, либо последовательным, когда порядок доступа к записям определяется псевдослучайным порядком их хранения в файле. Доступ к записям в упорядоченной последовательности производится в два этапа. На первом просматривается весь файл и генерируется требуемый упорядоченный список идентификаторов записей, который может быть сохранен на диске для многократного использования (см. описание глаголов SORT и SSELECT, генерирующих списки, а также глаголов SAVE-LIST и GET-LIST, позволяющих сохранять и считывать сохраненные списки, в гл. "Язык ACCESS"). На втором осуществляется прямой доступ к записям, попавшим в этот список.

Метод прямого доступа к записям в файле, использующий для определения местонахождения записи внутри файла имя этой записи, является весьма эффективным методом поиска данных и хорошо сочетается с диалоговой природой ОС PICK. Благодаря использованию алгоритма хэширования для определения местонахождения записи, время поиска ее в файле оказывается независимым от размеров этого файла.

Записи данных в ОС PICK являются структурированными. Они делятся на атрибуты, атрибуты - на значения, значения - на подзначения. Это требует использования специальных разделителей. Так, в качестве разделителя атрибутов выступает шестнадцатиричный код X'FE', который на большинстве клавиатур вводится нажатием на клавиши CTRL, SHIFT и N одновременно и обычно отображается на экране в виде символа "^". Таким образом, любая последовательность символов, заключенная в записи между парой кодов X'FE', будет интерпретироваться ОС PICK как содержимое атрибута (впрочем, справа атрибут может быть также ограничен и меткой конца записи). В качестве разделителей значений выступает шестнадцатиричный код X'FG' (CTRL_SHIFT_M, отображается на экране как "]"), подзначений - шестнадцатиричный код X'FC' (отображается как "\"). Именно использование разделителей, обозначающих конец и начало каждого элемента записи (атрибутов, значений и подзначений) позволяет ОС PICK работать с полями переменной длины. Подробнее об этом мы поговорим в разделе, посвященном физической структуре записей.

ОС РІСК может поддерживать в оперативном режиме:

- любое количество файлов, которые могут содержать
 - любое количество записей, которые могут содержать
 - любое количество атрибутов (полей), которые могут содержать
 - любое количество значений, которые могут содержать
 - любое количество подзначений;
- все файлы, записи, атрибуты, значения и подзначения могут иметь произвольную длину;
- верхний предел длины записи составляет 32267 байт.

Структура файлов в ОС РІСК

2.3. СЛОВАРИ

Словари определяют (описывают) данные, хранящиеся в файлах данных, соответствующих этому словарю. Словарные файлы могут располагаться на разных уровнях файловой иерархии.

Выше, в разделе "Структура файлов в ОС РІСК", мы уже упоминали о том, что словари в ОС РІСК могут располагаться на следующих трех уровнях файловой иерархии:

- уровень системного словаря (один файл на систему);
- уровень главных словарей пользовательских счетов (по одному на каждый счет);
- уровень словарей файлов (по одному на файл данных или на группу файлов данных).

Поскольку каждый словарь в ОС РІСК также является файлом, он, как и файл данных, также представляет собой набор записей. Словарные записи указывают местонахождение файлов следующего более низкого уровня на диске и определяют атрибуты хранящихся в них записей. В словарях хранятся записи следующих трех типов:

- записи-определители файлов, идентификаторами которых являются имена определяемых ими файлов; эти записи называются также D-указателями или записями типа D;
- записи-синонимы определителей файлов, идентификаторами которых являются синонимы имен файлов, на которые они указывают; эти записи называются также Q-указателями;
- записи-определители атрибутов, идентификаторами которых являются имена определяемых ими атрибутов; они называются также A-записями или записями типа A.

Записи-определители файлов (D-записи) и записи-синонимы определителей файлов (Q-записи) определяют файлы, находящиеся на следующем уровне иерархии. Их идентификаторами являются имена файлов, которые они определяют или на которые они указывают. Имена файлов в ОС РІСК должны представлять собой произвольную последовательность буквенно-цифровых символов, не содержащую запятых и точек с запятыми. Длина имени файла может быть произвольной.

Рекомендуется назначать файлам мнемонические имена, например, "ЦЕХ", "ОТЧЕТ", "СОТРУДНИКИ". Всякому файлу в ОС РІСК (за исключением системного словаря) должна соответствовать запись типа D в словаре (файле) предшествующего уровня, определяющая этот файл. Поскольку пользовательские счета в ОС РІСК - это тоже файлы, только с трехуровневой структурой с трехуровневой структурой (главный словарь счета - словари файлов -файлы данных), всякому пользовательскому счету соответствует запись типа D, его определяющая, в системном словаре. Пользователи работают в основном с файлами с двухуровневой структурой (словарь файла -файлы данных). Они могут самостоятельно добавлять к главному словарю своего счета записи типа D, указывающие на словари файлов. Определители файлов данных хранятся в виде D-указателей в словарях файлов. Заметим, что создание нового файла в счете предполагает не только создание соответствующего указателя в главном словаре, но и выбор места хранения этого файла на диске через процедуру хэширования, поэтому создавать новые файлы "вручную", через редактирование главного словаря своего счета (это единственный трехуровневый файл, доступный пользователям),

нельзя. Для создания новых файлов существует специальный процессор CREATE-FILE, при обращении к которому требуется указать все необходимые сведения о новом файле. Этот процессор автоматически формирует D-запись в MD, определяющую словарную часть создаваемого файла, и D-запись в словарной части, определяющую раздел данных, а также резервирует первичные области хранения обеих частей создаваемого файла на диске (см. описание процессора CREATE-FILE).

Чтобы иметь возможность обращаться из своего счета к файлам, хранящимся в счетах других пользователей, необходимо добавить в MO своего счета Q-указатели на эти файлы. Это сделать несложно, поэтому Q-указатели обычно создаются редактированием главного словаря через процессор EDITOR (для этого можно также создать специальную процедуру). Заметим, что доступ к файлам из других счетов ограничен системной защитой данных от несанкционированного доступа (необходимо обратиться к администратору системы и к владельцу другого счета).

Указатели типа Q могут использоваться также для назначения синонимов файлам, хранящимся в собственном счете: например, файлу с именем "СОТРУДНИКИ" можно назначить синоним "СОТР". что короче и удобнее вводить.

Словарные записи, также как и записи данных, разделяются на атрибуты (которые могут быть многозначными).

Процессор ACCESS работает со словарными записями специального типа - записями-определителями атрибутов (записи типа A), которые определяют структуру данных, хранящихся в разделе данных соответствующего файла. Записи типа A содержат информацию, аналогичную той, которая хранится в записях типа D, и, кроме того, еще следующую информацию:

- спецификации конверсий (входных преобразований), которые используются для форматирования данных перед выводом их на печать;
- спецификации корреляций, которые задают формулу вычисления значений атрибута на основании значений, хранящихся в других записях того же файла или в других файлах;
- способ выравнивания данных в поле (левое или правое).

Обращение к файлу данных производится по его имени. Обращение к словарю производится по имени файла, которому предшествует приставка DICT. Впрочем, это возможно, только если словарю соответствует единственный файл данных, поскольку в этом случае можно обойтись одним общим именем для обоих разделов файла. Если словарю соответствует несколько файлов данных, при обращении к файлу данных используется составное имя, образованное конкатенацией имени словаря и имени раздела данных через запятую, а обращение к словарю - через имя этого словаря и приставку DICT.

Словари могут содержать записи следующих типов:

1. Записи-определители файлов, (записи типа D), определяющие местонахождение файла следующего, более низкого уровня. на диске и другие его параметры.
2. Записи-синонимы определителей файлов (Q-указатели), которые указывают на файлы, определенные D-записями в том же или других счетах.
3. Записи-определители атрибутов (записи типа A), которые определяют структуру данных, хранящихся в файле данных.

Кроме того, в главном словаре могут храниться:

1. Глаголы (см. главу по системным командам TCL).
2. Процедуры (см. главу о процессоре PROC).
3. Служебные слова языка ACCESS (см. главу о языке ACCESS).

Типы словарных записей.

2.4. СЛОВАРИ. КОТОРЫМ СООТВЕТСТВУЮТ НЕСКОЛЬКО ФАЙЛОВ ДАННЫХ

Один и тот же словарь может относиться к нескольким файлам данных, обладающих схожей структурой. 1

Словари, занимающие третий (предпоследний) уровень в иерархии файлов, могут ссылаться на несколько файлов данных одновременно. Обычно это используется, когда файлы данных обладают сходной структурой.

В качестве примера рассмотрим гипотетическое предприятие, в котором есть несколько цехов, по которым собираются одинаковые показатели - фонд зарплаты, производительность труда, выпуск готовой продукции и т.д. Показатели, относящиеся к разным цехам, можно хранить в разных файлах данных, но словарь этих файлов данных рекомендуется сделать общим, поскольку данные по цехам собираются одинаковые, что позволяет избежать дублирования.

Для обращения к файлу данных, который пользуется общим словарем с другими файлами данных, следует указывать и имя словаря, и имя самого файла данных в следующей форме:

имя-файла (, имя-файла-данных).

Первый параметр ("имя-файла") при обращении к файлам всегда подразумевает имя словаря файла, т.е. имя, зарегистрированное в главном словаре счета. Второй параметр ("имя_файла_данных") необязательным и используется только в том случае, если одному словарю соответствует несколько файлов данных. Если между словарем и файлом данных имеется взаимнооднозначное соответствие, имя словаря является общим является общим как для словарного раздела, так и для раздела данных. Именно в этом смысле следует трактовать параметр "имя-файла" в настоящем руководстве. Итак, при обращении к файлу данных, который является единственным файлом данных, "прикрепленным" к некоторому словарю, зарегистрированному в главном словаре счета под именем "имя_файла", достаточно указать:

имя-файла

Возвращаясь к нашему примеру, пусть имя общего для нескольких файлов данных словаря будет "ЦЕХ", а имена файлов данных - "КРАСИЛЬНЫЙ", "ТКАЦКИЙ" и "СУШИЛЬНЫЙ". Тогда при обращении к данным, хранящимся в этих файлах, следует указывать:

ЦЕХ, КРАСИЛЬНЫЙ или
ЦЕХ, СУШИЛЬНЫЙ.

Ранее уже говорилось о том, что в словарях файлов хранятся записи типа D, которые определяют относящиеся к нему файлы данных. Если словарь обслуживает только один файл данных, идентификатор D-записи, определяющей этот файл данных в словаре файла, будет совпадать с именем самого словаря, т.е. с именем соответствующей D-записи в главном словаре счета. Например, если в пользовательском счете имеется файл СКЛАД, состоящий из словаря и единственного файла данных, в его словарной части будет храниться только одна D-запись, также называемая СКЛАД. Напротив, в словаре файла ЦЕХ будет храниться столько D-записей, сколько цехов имеется на нашем гипотетическом предприятии.

При создании файла, имеющего несколько разделов данных, используется следующая последовательность шагов:

- во-первых, создается словарь файла

```
>CREATE-FILE DICT ЦЕХ m <БК>
```

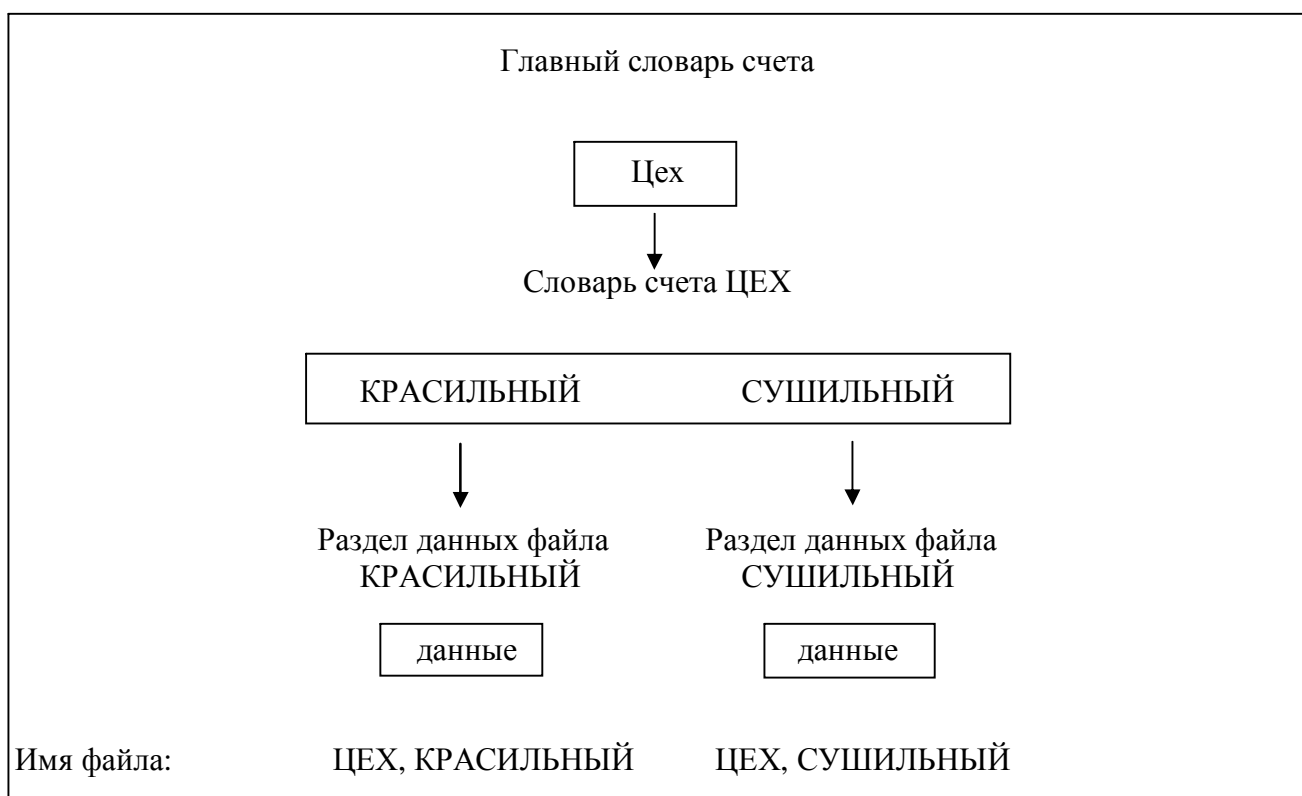
где m - модуло создаваемого словаря;

- во-вторых создаются разделы данных:

```
>CREATE-FILE DATA ЦЕХ,КРАСИЛЬНЫЙ m <БК>
```

```
>CREATE-FILE DATA ЦЕХ,ТКАЦКИЙ m <БК> и т.д.
```

где приставка DATA указывает, что создается файл данных.



Пример двухуровневого файла ЦЕХ с двумя разделами данных.

2.5. БАЗА И МОДУЛО

Физические границы файла в ОС PCK задаются двумя параметрами - базой (идентификатор начального фрейма файла на диске) и модулю (количество фреймов, резервируемых под файл при его создании).

Физические границы хранения файла на диске записываются в соответствующих атрибутах (полях) D-записи, его определяющей, т.е. записи, носящей то же имя, что и файл, и находящейся в словаре этого файла.

Процессор CREATE-FILE назначает создаваемым файлам два параметра хранения его на диске - базу и модулю.

БАЗА Физический адрес (номер) первого фрейма из числа блока непрерывных (смежных) фреймов, резервируемых под хранение данного файла на диске в момент его создания. ыбирается автоматически.

МОДУЛО Количество этих непрерывных фреймов. Выбирается пользователем.

Название "модулю" происходит от латинского "modulo" - "по модулю", т.е. остаток от деления одного целого числа на другое, именуемое "модулем". Так, 10 по модулю 7 будет 3. Выбор этого названия объясняется тем, что количество первоначально выделяемых под файл фреймов определяет также число групп фреймов, по которым используемый в ОС PCK алгоритм хэширования распределяет записи файла: по идентификатору записи вычисляется целое число, которое делится на модуле и остаток от деления дает номер группы. Внутри группы записи располагаются последовательно, в порядке их создания. Начальным фреймом в каждой группе служит один из фреймов из первичной области хранения файла, резервируемой процессором CREATE-FILE при его создании. При добавлении в файл новых записей группы, как правило, переполняются, тогда к ним автоматически подсоединяются новые свободные фреймы из области свободных фреймов (области переполнения).

Выбор модуле - очень важный момент, т.к. от этого выбора во многом зависит скорость доступа к хранящимся в файле записям. Алгоритм выбора оптимального модуле описывается в следующем разделе.

База и модулю файла записываются в соответствующую D-запись процессором CREATE-FILE в момент создания этого файла. Пользователь ни при каких обстоятельствах не должен пытаться их изменять.

Итак, в момент создания файла на диске резервируется некоторое число непрерывных (смежных) фреймов, которые составляют первичную область хранения. Размер этой первичной области определяется выбором модулю. По мере добавления к файлу новых записей его размеры могут превысить первоначальную область хранения вследствие переполнения групп. В этом случае система будет автоматически подсоединять к переполненной группе недостающие фреймы из области свободных фреймов (области переполнения). Теоретически, подобное "разрастание" файла ничем не ограничено, разве что пространства. На практике, однако, следует избегать чрезмерного переполнения файлов, т.к. это замедляет доступ. Избежать чрезмерного переполнения позволит выбор оптимального модулю для файла в момент его создания.

Запись "СОТРУДНИКИ" в главном словаре счета:

СОТРУДНИКИ

001 D

002 17324

003 3

Первичная область хранения словарного раздела
файла СОТРУДНИКИ:

N фрейма

17324	<input type="text"/>	1-я группа
17326	<input type="text"/>	2-я группа
17326	<input type="text"/>	3-я группа

Запись СОТРУДНИКИ в словаре СОТРУДНИКИ:

001 D

002 17573

003 373

Первичная область хранения раздела данных
файла СОТРУДНИКИ

17573	<input type="text"/>	1-я группа
17574	<input type="text"/>	2-я группа

.. и т.д. (373 раза).

Иллюстрация понятий базы и модуле файла.

2.6. ВЫБОР МОДУЛО

Скорость доступа к данным в файле и эффективность использования дискового пространства зависят от правильности выбора модуло.

При создании файла в глаголе CREATE-FILE необходимо указать имя создаваемого файла и его модуло. Модуло определяет количество фреймов в блоке непрерывных фреймов, резервируемых под файл при его создании, который называется также первичной областью хранения файла. Каждый фрейм из первичной области является начальным фреймом в группе, таким образом, модуле определяет количество групп, по которым алгоритм хэширования распределяет записи файла. По мере добавления к файлу новых записей, группы, как правило, переполняются. К переполнившейся группе автоматически подсоединяется новый фрейм из области свободных фреймов (области переполнения). Группа, в которую попадает та или иная запись при ее создании, определяется делением целого числа, однозначно задаваемого ее идентификатором, на модуло. Остаток от деления и дает номер группы. При необходимости получить доступ к этой записи система вновь генерирует это целое число на основании идентификатора записи, определяет номер группы, а затем последовательно просматривает все записи в группе в поиске нужной. Правильный выбор модуле играет большую роль для сокращения времени этого поиска. Дело в том, что количество попыток чтения с диска - фактор, который существенно замедляет реакцию системы - быстро возрастает с ростом числа фреймов в группе. Это связано с тем, что при обновлении той или иной записи в группе примерно половину содержимого всей группы приходится заново переписывать. Таким образом, чтобы обновить единственную запись в группе, приходится считывать все фреймы в группе, а потом вновь записывать на диск примерно половину из них, проверяя к тому же их содержимое с помощью контрольных сумм. Чтобы по-возможности сократить эту нагрузку на систему, рекомендуется ответственно подходить к выбору модуло и пользоваться для этого приведенными ниже таблицами.

Ниже приводятся две таблицы, пользуясь которыми можно подобрать оптимальное модуло для вновь создаваемого файла. Для этого необходимо знать два параметра: средний размер записей в файле и примерное количество этих записей. Зная средний размер записей в группе, по первой таблице определяем рекомендуемое количество записей в группе: это количество должно быть, по возможности небольшим, но чтобы пространство диска не пропадало зря, умножение количества записей в группе на средний объем записи должно давать число, кратное целому числу фреймов (примерно). Именно с последним обстоятельством связано то, что рекомендованное количество записей в группе не является монотонно убывающей функцией от их среднего объема - указанный в крайнем справа столбце первой таблицы показатель среднего количества байт в группе всегда близок к емкости целого числа фреймов (500, 1000, 1500 и т.д.). В последней строке таблицы рекомендованное количество записей в группе составляет 0.8. Такое значение годится для файлов, в которых сравнительно немного записей, но в среднем эти записи велики, напрмер, для файлов, в которых хранятся бейсиковские или ассембдеровские программы. Если в таком файле хранится много записей, рекомендуется скорректировать рекомендуемое число записей в группе в сторону его повышения, иначе может пропасть много пространства на диске. Зная рекомендованное количество записей в группе и ожидаемое число записей в файле, по алгоритму, описанному во второй таблице, подбираем модуло.

Рекомендуется, чтобы модуле было простым числом. Во всяком случае, оно не должно быть четным и не должно делиться на 5. В таблице приведены лишь ориентировочные значения, по которым несложно подобрать ближайшее простое число.

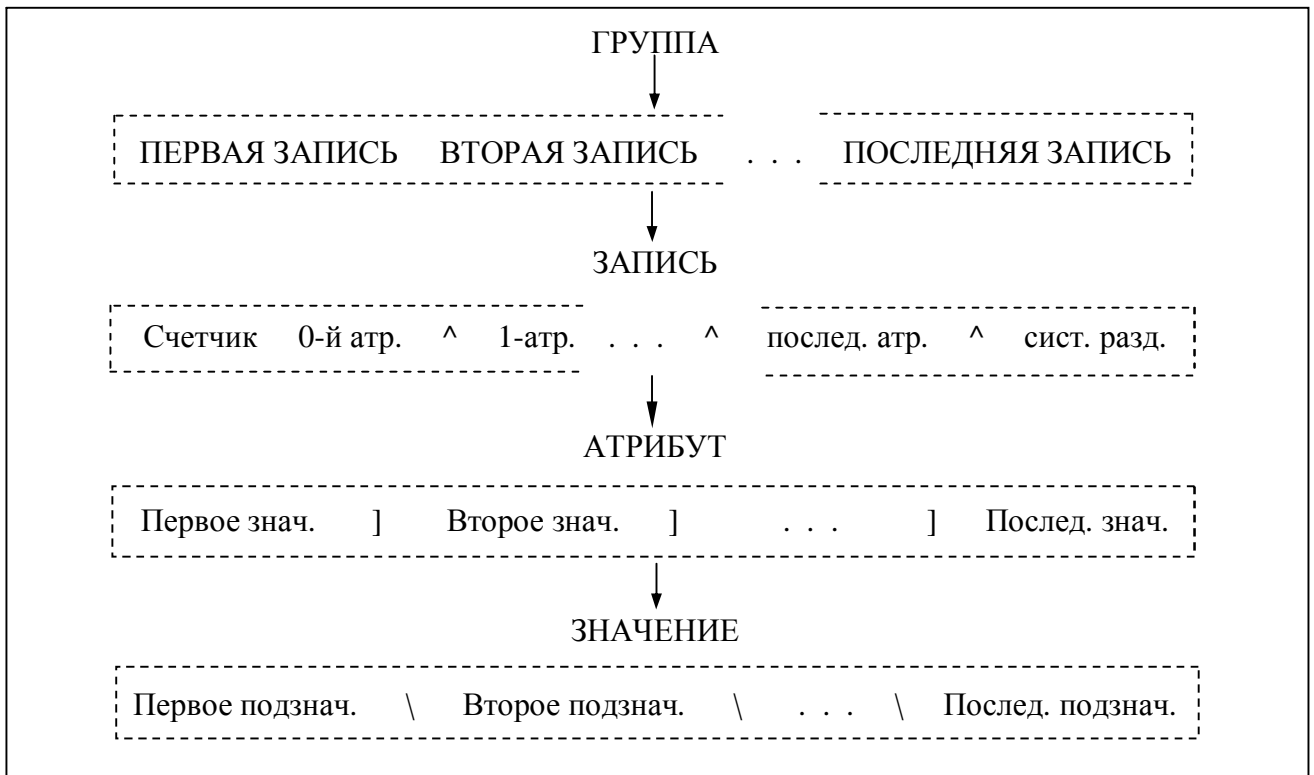
Средний размер записи	Рекомендуемое количество записей в группе	Тогда среднее число байт в группе будет
20	22.0	440
35	13.0	455
50	9.0	450
75	12.0	900
100	9.0	900
125	7.5	937
150	6.0	900
175	8.0	1400
200	7.0	1400
250	5.8	1450
300	6.4	1920
350	5.5	1925
400	4.8	1920
500	3.8	1900
1000	3.0	3000
5000	0.8	4000

Выбор количества записей в группе в зависимости от среднего размера записей в файле

Средний объем записи		Ожидаемое число записей в файле	Число записей в группе (берется из верхней таблицы)	Модуло (примерный ориентир)
-----		-----	-----	
20	→	800	/ 22.0	= 36
40	→	5000	/ 11.0	= 454
210	→	1800	/ 7.0	= 257
4000	→	230	/ 1.0	= 230

Порядок расчета модуло

Доступ к записям происходит следующим образом: вначале с помощью алгоритма хеширования определяется группа, в которой хранится запись, затем внутри группы осуществляется последовательный поиск нужной записи по имени. При этом для "перескакивания" с одного имени записи на другое используется значение счетчика (длина записи в байтах). Если при переходе к очередной записи обнаруживается, что вместо поля счетчика в этом месте стоит системный разделитель (конец группы), значит, просмотр группы завершился. Под пустой группой понимается такая группа, у которой на месте первого же значения данных стоит системный разделитель. Процессоры CREATE-FILE и CLEAR-FILE возвращают файлы, все группы в которых пусты.



Физическая структура записи и ее элементов в группе

2.8. ЛОГИЧЕСКАЯ СТРУКТУРА ЗАПИСИ

В разделе рассматривается логическая структура записи.

Хотя иметь четкое представление о физической структуре записи на диске очень важно, при обычной работе с системой доступ к записям осуществляется на более "абстрактном" или "высоком" уровне. Обращение к файлам происходит по именам, к записям внутри файла - по имени файла и идентификатору записи. Многие процессоры выводят на экран атрибуты в виде строк, поэтому наряду с термином "атрибут" для обозначения поля записи часто используется также слово "строка". На рис. А показан пример выполнения глагола COPY, который копирует содержимое записи ЗАПИСЬХ.

Процессоры COPY и EDITOR работают с записями в файле на уровне строк-атрибутов. Они не проводят никакого логического различия между строками и отслеживают только исключительно по их номерам.

Процессор ACCESS, напротив, по-разному интерпретирует записанную в атрибутах информацию, поскольку он обращается к ней через словарные записи, в которых описывается характер этой информации и как с ней следует обращаться.

Понятие "записи" в ОС PICK несколько отличается от общепринятого понятия записи. Различие заключается в том, что записи в ОС PICK структурированы. Логически их можно уподобить трехмерным массивам, поскольку местоположение всякого элемента в записи определяется по трем индексам - номеру (имени) атрибута, номеру значения внутри атрибута и номеру подзначения внутри значения. Если подзначений в записи нет, по аналогичной схеме ее можно уподобить двумерному массиву (номер атрибута, номер значения), если запись состоит только из атрибутов, она аналогична вектору. Такое логическое представление записи позволяет интерпретировать ее как множество элементарных записей, заключенных в одну физическую оболочку одной сложной записи и распределенных на всем протяжении последней с определенным шагом. Более наглядно это можно пояснить, вернувшись к аналогии с матрицей. Представим себе двумерную матрицу $n \times p$, где $n=100$, а $p=5$ (цифры не имеют значения, но для создания мысленного образа матрица должна быть "вытянутой"). Попробуем теперь взглянуть на эту матрицу как на запись, состоящую из 100 атрибутов, каждый из которых имеет пять значений. Строки этой матрицы (их пять) и будут теми элементарными логическими записями, из которых состоит сложная запись-матрица: первое значение берется в ней из первого атрибута, второе - из второго и т.д. С этой точки зрения главной функцией разделителей значений становится разграничение содержимого одной элементарной записи от другой внутри атрибута, а разделителей подзначений - разграничение содержимого элементарных записей на уровне значений (вычленение линейных векторов из трехмерного массива). Извлечение элементарных записей из сложных производится с помощью команд языка ACCESS, причем над этими элементарными записями можно производить те же операции, что и над обычными - упорядочивать их, производить выборку по критериям и формировать из них отчеты.

Описанный выше формат логической записи используется всеми процессорами ОС PICK. Задача пользователя - задать дополнительные (качественные) характеристики атрибутов, связанные с оформлением их вывода на экран или на принтер. В приведенных ниже примерах показано содержимое одной и той же записи - в первом случае, обработанной процессором COPY, во втором - процессором ACCESS. Процессор COPY

просто выводит содержимое атрибутов на экран в виде последовательности пронумерованных строк, а процессор ACCESS обращается к словарю файла, в котором хранится запись данных ЗАПИСЬХ, из которого узнает, что второй атрибут (вторая строка) называется ИМЯ, третий атрибут - адрес, что позволяет не только соответствующим образом озаглавить содержимое атрибутов при выводе на экран (в виде колонтитулов), но и обращаться к атрибутам внутри записи по их символьным номерам.

Заметим также, что процессор COPY выводит содержимое первого атрибута записи в виде 3746, тогда как в отчете, сформированном процессором ACCESS вместо этого значения фигурирует "04/03/78" (день-месяц..год), что является результатом конвертирования хранящегося в записи значения из внутреннего формата во внешний (см. главу, посвященную языку ACCESS).

```
>COPY МОЙ_ФАЙЛ ЗАПИСЬХ (Т
```

	ЗАПИСЬХ	имя записи
001	3746	1-й атрибут
002	ИВАНОВ П.В.	2-й атрибут
003	МОСКВА	3-й атрибут

Содержимое записи, выданное на экран процессором COPY.

```
>LIST МОЙ_ФАЙЛ "ЗАПИСЬХ" ДАТА ИМЯ АДРЕС
```

СТР 1		09:28:32	12 ЯНВ 1978
МОЙ_ФАЙЛ	ДАТА	ИМЯ	АДРЕС
ЗАПИСЬХ	04/03/78	ИВАНОВ П.В.	МОСКВА

Содержимое записи, выданное на экран процессором ACCESS

2.9. ХРАНЕНИЕ ЗАПИСИ НА ДИСКЕ И АЛГОРИТМ ХЕШИРОВАНИЯ

Для определения местонахождения записи на диске в ОС PISC используется алгоритм хеширования, входными параметрами которого являются идентификатор записи и модуло файла. Этот метод позволяет определить адрес группы (номер начального фрейма), в которой хранится требуемая запись или в которую она записывается при создании.

Ниже приводится формула, по которой в ОС PISC проводится хэширование при обращении к записям в файле. Идентификатор записи интерпретируется в этом алгоритме как строка двоичных байтов произвольной длины, на основании которой генерируется некое целое число (методом последовательного сложения всех байтов, причем каждая частичная сумма умножается на 10). Деление этого числа на модуле дает целый остаток ($0 \leq \text{остаток} < \text{модуло}$), который и принимается за номер группы, в которой должна храниться запись с указанным идентификатором. Если к этому значению прибавить номер начального фрейма файла (его базу), мы получим физический адрес группы на диске, вернее, адрес первого фрейма в ней.

Определив адрес начального фрейма группы, в которой хранится запись, система проверяет имена всех записей в ней, пока не отыщет нужное. Все фреймы, образующие группу (кроме первого и последнего), содержат адреса логически последующего и логически предшествующего фреймов (в терминологии ОС PISC эти адреса называются "ссылка вперед" и "ссылка назад", соответственно), поэтому логически группа представляется как непрерывная строка, состоящая из следующих одна за одной записей, хотя границы образующих группу фреймов могут приходиться на середину любой записи,

Первоначально выделенные под группы фреймы могут переполняться на диске, именуемое "первичной областью хранения файла" - т смежных пустых фреймов, где m - модуло файла. Этот объем непрерывного пространства на диске считается постоянно закрепленным за файлом. По мере того, как в файл добавляются все новые и новые записи, первоначально выделенные под группы фреймы могут переполняться. Тогда к группе автоматически подцепляется" новый фрейм и длина логической последовательности (группы) фреймов возрастает на единицу. При удалении из файла ненужных записей группы могут сокращаться в размере - тогда все незанятые фреймы, не входящие в область первичного хранения файла, возвращаются в пул свободных фреймов.

```
X=0
FOR J=1 TO LEN (ИМЯ_ЗАПИСИ)
X=X*10+SEQ (ИМЯ_ЗАПИСИ[J ,1])
NEXT J
GROUP=REM(X,MODULO)
FID=GROUP+BASE
```

Здесь:

ИМЯ_ЗАПИСИ	-	идентификатор записи
LEN	-	функция, возвращающая число символов, содержащееся в документе
ИМЯ_ЗАПИСИ[J,I]	-	извлечение j-го символа из имени записи
SEQ	-	функция, преобразующая указанный символ в двоичное число
REM	-	функция, возвращающая остаток от деления числа X на модулю
FID	-	номер начального фрейма в группе
BASE	-	база
MODULO	-	модулю

Алгоритм хеширования, записанный в виде программы на PICK/BASIC.

2.10. ЗАПИСИ-ОПРЕДЕЛИТЕЛИ ФАЙЛОВ

Словарные записи-определители файлов определяют словари, находящиеся на более низких уровнях файловой иерархии и файлы данных. Эти записи называются также записями типа D, поскольку в первом их атрибуте указан код, начинающийся с буквы "D" - "D", "DX" или "DY". Записи типа D создаются автоматически при выполнении глагола CREATE-FILE.

На уровне системного словаря, находящегося на верхнем уровне файловой иерархии в системе, записи типа 0 (записи-определители файлов) определяют словари пользовательских счетов. На уровне главного словаря пользовательского счета записи типа D определяют словари пользовательских файлов. Записи типа D, хранящиеся в словарях файлов, определяют формы данных, относящиеся к этому словарю. Все атрибуты записей-определителей файлов, включая их идентификатор (нулевой атрибут), содержат ту или иную (обязательную или необязательную) информацию, относящуюся к словарному файлу более низкого уровня или к файлу данных.

Идентификатор Идентификатор записи-определителя представляет собой имя файла (словаря или файла данных), который эта запись определяет.

1-й атр. В этом атрибуте указывается код D, за которым может следовать одно- или двубуквенный код. Код D заносится в этот атрибут процессором CREATE-FILE автоматически при создании файла. Возможные его варианты:

DX Не записывать данный файл на ленту в ходе процедуры FILE-SAVE (после выполнения очередной процедуры FILE-RESTORE он будет потерян).

DY Не записывать содержимое данного файла на ленту в ходе процедуры FILE-SAVE (после выполнения очередной процедуры FILE-RESTORE он окажется пустым).

DC Файл содержит бинарные данные (в настоящей версии ОС PICK это используется только применительно к системному файлу POINTER-FILE).

2-й атр. Содержит базу (номер начального фрейма) файла десятичных кодах. Значение выбирается процессором CREATE-FILE.

3-й атр. Содержит модуль файла. Значение модуля выбирается пользователем и заносится в запись-определитель файла через процессор CREATE-FILE.

4-й атр. Сепарация. В R83 сепарация всегда равна единице.

Предупреждение: в атрибуты 2-4 данные заносятся только процессором CREATE-FILE. Их ни в коем случае нельзя изменять вручную.

5-12-й атр. То же, что и в записях-определителях атрибутов (см. соотв. разд.).

13-й атр. Параметры переопределения файла. В этом атрибуте можно указывать новое значение модуля для файла, которое будет использоваться при выделении ему места на диске в ходе очередной процедуры FILE-RESTORE (см.). Формат:

(m)

где m - десятичное простое число, задающее новое модуло файла.

В нижеприведенном примере представлены две записи типа D, одна из которых хранится в главном словаре пользовательского счета и определяет файл СОТРУДНИКИ. В словарном разделе этого файла также хранится запись типа D, определяющая файл данных с тем же именем. Обе записи имеют одинаковые названия, но хранятся в разных словарях и указывают на файлы разного уровня: первая - на словарный файл, вторая - на файл данных.

Идентификатор Записи	СОТРУДНИКИ	СОТРУДНИКИ
	001 D	001 D
	002 17324	002 17573
	003 3	003 373
	004 1	004 1
	005	005
	006	006
	007	007
	008 L	008 R
	009 10	009 7

Заметим, что запись СОТРУДНИКИ в главном словаре счета содержит параметры словарных записей файла сотрудники (так, в 9-м атрибуте в ней указано "L", в 10-м атрибуте - "10"), тогда как запись СОТРУДНИКИ в словаре файла содержит параметры записей данных этого файла (9-й атрибут = "R", 10-й = 7).

Примеры записей-определителей файлов.

2.11. ЗАПИСИ-СИНОНИМЫ ОПРЕДЕЛИТЕЛЕЙ ФАЙЛОВ

Записи-синонимы определителей файлов позволяют получать из своего счета доступ к файлам, хранящимся в счетах других пользователей, а также назначать новые имена файлам, хранящимся в собственном счете, не копируя их содержимое и сохраняя оба (или все) имена в силе. В первом атрибуте этих записей стоит код Q, поэтому они называются также Q-указателями.

Название (идентификатор) и содержимое атрибутов записей-синонимов определителей файлов:

Идентификатор	Новое имя файла или имя, по которому из данного счета можно будет обращаться к файлу, хранящемуся в другом счете.
1-й атр.	Содержит код Q.
2-й атр. хранится	В этом атрибуте должно быть указано имя счета, в котором исходный файл. Имя счета должно быть указано в том виде. В котором оно зарегистрировано в системном словаре. Если назначается синоним собственно файлу счета, оставьте этот атрибут пустым.
3-й атр.	Должен содержать идентификатор исходной записи-определителя файла, т.е. имя файла в указанном счете, которому назначается синоним. Если оставить этот атрибут пустым, система назначает синоним главному словарю текущего счета.
4-й атр.	Не используется.
5-10-й атр. пределителях	Совпадают с соответствующими атрибутами в записях-атрибутов (см.).

Q-указатели добавляются в главный словарь счета через обработку его редактором EDITOR. Можно воспользоваться также специальной процедурой, именуемой SET-PILE, которая создает временный Q-указатель под именем QFILE. Описание этой процедуры приводится в главе, посвященной командам TCL.

В следующих примерах рассматривается запись-синоним определителя файла СОТРУДНИКИ, которая позволяет обращаться к этому файлу, хранящемуся в счете БУХ, под именем СОТР. Приводится также пример Q-указателя, который назначает синоним КОМНАТА3 главному словарю счета КАНЦЕЛЯРИЯ и пример Q-указателя, который ссылается на другой файл из текущего счета.

СОТР	КОМНАТАЗ	ПРИМЕР
001 Q	001 Q	001 Q
002 БУХ	002 КАНЦЕЛЯРИЯ	002
003 СОТРУДНИКИ	003	003 ПРИМЕР_ФАЙЛА

Примеры Q-указателей

```

>EDIT MD СОТР
NEW ITEM
TOP
.I                <BK>
001+Q            <BK>
002+БУХ         <BK>
003+СОТРУДНИКИ <BK>
004+
TOP
.FI
'СОТР' FILED

```

<BK> означает нажатие на клавишу "возврат каретки".

Последовательность действий при создании Q-указателя
через редактор EDITOR

2.11.1. Q-указатели: рефлексивная форма

Если в Q-указателе второй и третий атрибут пусты, значит, он указывает на файл, в котором сам находится. Это применяется в двух случаях. Во-первых, чтобы иметь возможность ссылаться на главный словарь собственного счета: наберите команды ED MD MD, и вы увидите, что в записи MD файла MD содержится только код Q в первом атрибуте. Это наиболее эффективный способ доступа к главному словарю собственного счета.

Второй случай - ссылка на пользовательский файл, состоящий только из словарной части. Чтобы не пришлось всякий раз ссылаясь на этот файл, указывать перед его именем приставку DICT, создайте в нем Q-указатель, имеющий то же имя, что и сам файл.

В главном словаре счета:

MD	- название записи, указывающей на файл MD
001 Q	- возврат в MD

В словаре файла с ИМЕНА, состоящего только из словарной части:

ИМЕНА	- идентификатор, совпадающий с именем файла
001 Q	- вернуться к словарю

Использование рефлексивной формы Q-указателей

Идентификатор Q-указателя используется в системе только до тех пор, пока она не отыщет первую запись типа 0 в цепочке. Так, обращаясь к Q-указателю Q-FILE, который ссылается на файл НАСТОЯЩИЙ-ФАЙЛ, заставит систему искать запись типа D в словаре файла НАСТОЯЩИЙ-ФАЙЛ, а не запись Q-FILE. Исключение в этом отношении составляет язык ACCESS, который отыскивает Q-указатель и пытается считать из него спецификацию конверсии, длину поля и тип выравнивания. Если Q-указатель не содержит такой информации, процессор ACCESS пытается считать эту информацию из записи D-указателя, Если и в D-указателе этих сведений нет, конверсия будет считаться отсутствующей, тип выравнивания - левым, длина поля равной 9-ти байтам. Это позволяет для одного и того же атрибута задавать разные форматы вывода, что иногда необходимо при упорядочивании и выводе на экран.

2.11.2. Q-указатели: спецификация счета

Во втором атрибуте Q-указателя специфицируется имя счета, в котором содержится исходный файл. Если этот атрибут оставить пустым, система будет искать исходный файл в текущем счете. Если имя счета указано, поиск счета с таким именем будет производиться по системному словарю (SYSTEM). Если при этом окажется, что счет с таким именем в словаре не зарегистрирован, будет выдано сообщение об ошибке.

Ссылки на главные словари других счетов делаются так: во втором атрибуте Q-указателя проставляется имя D-указателя на этот счет в системном словаре (т.е. указывается имя счета), а третий атрибут оставляется пустым.

2.11.3. Q-указатели: спецификация файла

В 3-м атрибуте указывается имя файла, на который ссылается Q-указатель. Если 3-й атрибут пуст, по умолчанию в качестве имени файла берется имя самого Q-указателя.

Как правило, имя файла в 3-ем атрибуте Q-указателя должно быть именем D-указателя в главном словаре счета. Имя счета указывается во 2-м атрибуте.

2.11.4. Q-указатели: спецификация раздела данных файла

В третьем атрибуте Q-указателя может быть записано имя файла в виде ИМЯ-СЛОВАРЯ, ИМЯ-ФАЙЛА-ДАННЫХ. В этом случае Q-указатель будет относиться только к файлу данных с именем ИМЯ-ФАЙЛА-ДАННЫХ, а все остальные файлы данных, на которые ссылается словарь ИМЯ-СЛОВАРЯ, будут проигнорированы. Это позволяет существенно сократить тексты бейсиковских программ и процедур, которые ссылаются на разделы данных файлов со сложной структурой.

Q-ФАЙЛ 001 Q 002 ИМЯ_СЧЕТА 003 ИМЯ_СЛОВАРЯ, ИМЯ_ФАЙЛА_ДАННЫХ

Пример Q-указателя, ссылающегося на раздел данных файла,
включающего несколько разделов данных

2.12. ЗАПИСИ-ОПРЕДЕЛИТЕЛИ АТТРИБУТОВ

Записи-определители атрибутов определяют различные атрибуты (строки или поля) в записях файлов следующего, более низкого уровня и используются при обработке содержимого записей данных процессором ACCESS. В первом атрибуте записей-определителей атрибутов указывается код А.

Запись-определитель атрибута определяет природу и формат представления данных, хранящихся в соответствующем атрибуте в записях файла более низкого уровня. Эти сведения используются процессором ACCESS. Во всякой записи-определителе атрибута имеется строка, именуемая "АМС" ("Attribute Mark Count" - номер атрибута), значение которой служит адресом определяемого ею атрибута - в этой строке указывается порядковый номер атрибута в записи. Запись-определитель атрибута относится к атрибутам с указанным номером во всех записях соответствующего файла (или файлов) данных. В записи-определителе атрибута специфицируется также символьное имя этого атрибута, используемое при распечатке отчетов в качестве колонтитула (заголовка столбца, в котором печатаются значения этого атрибута из выбранных записей). Записи-определители атрибутов строятся следующим образом:

Идентификатор	Символьное имя (как правило, мнемоническое) определяемого атрибута. По этому имени можно вместо номера сослаться на атрибут в предложениях ACCESS.
1-й атр.	Должен содержать буквенный код А (или Х)
2-й атр.	Номер определяемого атрибута внутри записи. Если в этом месте указать ZERO, значит, данная запись относится к имени (идентификатору) записи данных. Если указывается 0 или фиктивное значение, заведомо большее, чем максимальное число атрибутов, значит, данная запись относится не к реальным данным, а к вычисляемым.
3-й атр.	Может содержать колонтитул, распечатываемый при табличной выдаче на экран или на принтер в виде заголовка столбца, содержащего значения соответствующего атрибута из выбранных записей.
4-й атр.	Структурный код (см. главу, посвященную языку ACCESS).
7-й атр.	Содержит спецификацию конверсий (выходных преобразований), которая используется для преобразования хранящихся в файле данных из формата обработки во внешний формат представления.
8-й атр.	Содержит спецификацию корреляции (формулу преобразования значений из внутреннего формата в формат обработки).
9-й атр.	Специфицирует тип выравнивания (левое или правое) данных внутри поля при выводе на экран или на принтер.
10-й атр.	Максимальная длина значений атрибута (в байтах). Это обязательный параметр.

Ниже приводятся примеры записей-определителей атрибутов, который определяют различные поля записей, хранящихся в файле СОТРУДНИКИ (в его разделе данных).

Идентификатор:	ФИО	ДАТА_РОЖД	ДОЛЖНОСТЬ
	001 А	001 А	001 А
	002 4	002 5	002 7
	003	003 Родился	003
	004	004	004
	005	005	005
	006	006	006
	007	007 MD	007
	008	008	008
	009 L	009	009 L
	010 7	010 10	010 15

Примеры записей-определителей атрибутов, хранящихся в словаре файла СОТРУДНИКИ. Эти записи определяют 4-е, 5-е и 7-е поля (атрибуты) записей из раздела данных в файле с тем же названием.

2.13. СЛОВАРНЫЕ ЗАПИСИ: ЗАКЛЮЧЕНИЕ

В данном разделе дается краткая справка по типам словарных записей, используемым в системе.

ЗАПИСИ-ОПРЕДЕЛИТЕЛИ ФАЙЛОВ И АТТРИБУТОВ

Структура записей-определителей файлов, записей-определителей атрибутов и записей-определителей синонимов файлов и записей-определителей синонимов атрибутов приводится в нижеследующей таблице.

ЗАПИСИ СИСТЕМНОГО СЛОВАРЯ

Системный словарь в системе один. Он находится на вершине файловой иерархии и содержит только записи, в первом атрибуте которых содержится один из кодов D, DX, DY или Q. Записи в системном словаре указывают на пользовательские счета и на некоторые специальные системные файлы. На основании содержимого этих записей процессор LOGON (процессор, регистрирующий пользователей при входе в систему) судит о том, может ли тот или иной пользователь получить доступ к затребованному счету. На каждый пользовательский счет в системном словаре может быть только по одному D-указателю. Если для одного и того же счета требуется назначить несколько имен, эти имена регистрируются в системном словаре в качестве Q-указателей. В структуре записи системного словаря (как Q, так и D) атрибуты с 5-ого по 8-ой играют иную роль нежели в структуре словарных записей других уровней. Записи в системном словаре полностью управляют процессом FILE-SAVE (процессом создания архивной копии системы на внешнем носителе - на ленте или на дискете).

ЗАПИСИ В ГЛАВНОМ СЛОВАРЕ СЧЕТА

В каждом пользовательском счете может быть только один главный словарь (MD). Как и всякий другой словарь или файл в ОС PICK, MD состоит из записей. Записи MD, в первом атрибуте которых указан код A, определяют форматы атрибутов (строк) записей файлов следующего уровня - уровня словарей файлов. Записи, в первом атрибуте которых стоит код D, указывают на файлы, созданные и хранящиеся в этом счете.

Помимо записей-определителей файлов и записей-определителей атрибутов в главном словаре хранятся также записи, определяющие глаголы, процедуры и различные служебные слова языка ACCESS. Все эти записи имеют нестандартную структуру (подробнее см. в главах, посвященных командам TCL, процессору PROC и языку ACCESS).

ЗАПИСИ-ОПРЕДЕЛИТЕЛИ			
Но. атр.	Файлов	Синонимов файлов	Атрибутов
1	D, DX, DY, DC ДСК, ДСУ	Q	A, X
2	база файла	имя счета	номер атр.
3	модуло	синоним	колонтитул
4	сепарация	не используется	структурный код
5	код (коды) доступа для чтения		зарезервирован
6	код (коды) доступа для записи		зарезервирован
7	спецификация конверсии		
8	зарезервированы		корреляция
9	способ выравнивания		
10	максимальная длина поля		
11	зарезервированы		
12	зарезервированы		
13	новое модуло и сепарация		зарезервирован

Структура записей-определителей атрибутов и записей-определителей файлов.

2.14. СИСТЕМНЫЕ ФАЙЛЫ И СЛОВАРИ

Файлы и словари, описываемые в настоящем разделе, выполняют в ОС PICK особые функции.

В ОС PICK имеется особый счет - счет SYSPROG, который необходим для поддержания системы и который в отличие от пользовательских счетов называется счетом системного администратора. В этом счете находится файл системных сообщений ERRMSG и файл-шаблон главных словарей вновь создаваемых пользовательских счетов NEWAC. Файл ERRMSG открыт для доступа всех пользователей - они могут считывать из него системные сообщения и любую другую необходимую информацию. Файл NEWAC используется для создания в системе новых пользовательских счетов (их главных словарей) и содержит стандартный набор глаголов и служебных слов. В счете SYSPROG хранятся тексты системных процедур FILE-SAVE и FILE-RESTORE, а также процедура инициализации (очистки) архивного файла ACC.

ФАЙЛ ERRMSG

Этот файл, хранящийся в счете SYSPROG, состоит из одного только словарного раздела и содержит тексты системных сообщений (сообщения об ошибках и другую служебную информацию; см. приложения). В главном словаре каждого пользовательского счета должен быть Q-указатель ERRMSG, ссылающийся на этот файл (создание такой записи в MD происходит автоматически в ходе процедуры CREATE-ACCOUNT).

ФАЙЛ SYSPROG-PL

Этот одноуровневый словарный файл содержит тексты системных процедур. Процедуры, в нем записанные, могут выполняться только из счета SYSPROG (см. разд. "Сопровождение системы").

ФАЙЛ NEWAC

Этот словарь определен в счете SYSPROG и представляет собой прототип (шаблон) главного словаря пользовательского счета. Этот шаблон копируется во вновь создаваемые счета в ходе процедуры CREATE-ACCOUNT.

АРХИВНЫЙ ФАЙЛ ACC

Файл ACC содержит как архивную информацию об эксплуатации системы, так и сведения о подключенных в текущий момент пользователях. Формат записей этого файла описан в разделе, посвященном процессам LOGON/LOGOFF. Файл ACC необходимо периодически очищать, чтобы не допустить его переполнения.

ФАЙЛ PROCLIB

В этом файле хранятся тексты всех общедоступных процедур (LISTU, CT и др.). В каждом MD пользовательских счетов имеется указатель на этот файл, носящий то же имя, что позволяет передавать управление из пользовательского счета нужной процедуре в файле PROCLIB (см. главу о процессоре PROC).

ФАЙЛ BLOCK-CONVERT

В этом файле содержатся записи, используемые глаголом BLOCK-PRINT для печати заголовков в матричном (блочном) формате.

ФАЙЛ POINTER-FILE

Каждый файл-указатель должен иметь в первом атрибуте записи, его определяющей, код DC. Он должен быть двухуровневым, но бывает удобно делать указатель уровня данных в словаре Q-указателем на самого себя. Имя POINTER-FILE является зарезервированным в том смысле, что именно к этому имени обращается обработчик процедур. Таким образом, можно дать истинному файлу-указателю имя, отличное от POINTER-FILE, а запись POINTER-FILE сделать Q-указателем на этот истинный файл-указатель. Таким способом можно обращаться к любым файлам-указателям в системе. Можно также определить несколько файлов-указателей в рамках одного счета и использовать каждый из них для своей группы задач.

Следует, однако, иметь в виду, что обработчик файлов-указателей может одновременно обращаться лишь к одному файлу-указателю и все процессы, запускаемые из одного счета, могут обращаться лишь к одному файлу-указателю (а именно, к тому, на который указывает Q-указатель POINTER-FILE).

ФАЙЛЫ БЕЙСИКОВСКИХ ПРОГРАММ

Всякий файл, в котором хранятся бейсиковские программы, обязательно должен иметь раздел словаря и один или несколько разделов данных, а в записи, определяющей этот файл в главном словаре счета, в первом атрибуте обязательно должен быть указан код DC. Исходные тексты программ должны быть записаны в файлах данных, а в словарном разделе хранятся указатели на исполняемые программы (их объектные коды). Если файлов данных, в которых хранятся бейсиковские программы, на один словарный файл приходится несколько, и если в разных файлах данных хранятся программы, носящие одно и то же имя, то по команде RUN будет исполняться та из них, которая была откомпилирована последней.

По команде CATALOG имя программы заносится в MD счета в виде указателя на файл данных, в котором хранится эта программа.

По команде DECATALOG объектный код указанной программы удаляется из системы. Выполнение этой команды не требует, чтобы указанная программа была предварительно каталогизирована.

2.15. ПРОЦЕССОРЫ УПРАВЛЕНИЯ ФАЙЛАМИ

В разделе приводятся краткие сведения о процессорах управления файлами в ОС РІСК.

Процессоры управления файлами позволяют генерировать, очищать, удалять файлы и производить над ними другие действия. К процессорам управления файлами относятся такие, как CREATE-FILE, CLEAR-FILE и DELETE-FILE.

Имеются и другие процедуры управления файлами, например, процедура создания пользовательского счета, процедуры сохранения и восстановления файлов, которые подробно описываются в главе "Сопровождение системы".

ПРОЦЕССОР CREATE-FILE

Этот процессор используется для создания новых словарей и файлов данных. Он создает словарь файла, регистрирует его в главном словаре счета (заносят туда соответствующую D-запись), а также резервирует и подсоединяет область первичного хранения файла. Процедура создания файла не требует вмешательства пользователя. Он должен только ввести глагол CREATE-FILE, указать в нем имя создаваемого файла и его модуль (количество фреймов, резервируемое под первоначальную область хранения), после чего нажать на клавишу <BK>.

ПРОЦЕССОР CLEAR-FILE

Этот процессор очищает специфицированный файл, т.е. переводит все его группы в состояние "пусто", помещая в первую позицию данных каждого первого фрейма в группе системный разделитель, обозначающий конец группы. Все подсоединенные к первоначальной области хранения файла фреймы при этом отсоединяются и возвращаются в область свободных фреймов (область переполнения). С помощью этого процессора можно очищать либо только словарь файла, либо только раздел данных. Если нужно очистить и то, и другое, глагол CLEAR-FILE необходимо исполнить дважды.

ПРОЦЕССОР DELETE-FILE

Процессор DELETE-FILE удаляет указанный файл из системы, причем с его помощью можно удалить либо только раздел данных, либо только словарь, либо и то, и другое вместе.

Если один общий словарь используется несколькими файлами данных, любой из этих файлов данных может быть удален, очищен или создан, не затрагивая других файлов данных, связанных с этим словарем.

2.16. СОЗДАНИЕ НОВЫХ ФАЙЛОВ: ПРОЦЕССОР CREATE-FILE

Процессор CREATE-FILE используется для создания в системе новых словарей и файлов данных.

Процессор CREATE-FILE используется для создания словарей файлов путем резервирования первичного пространства на диске под их хранение и добавления в главный словарь счета D-записи, указывающей на создаваемый словарь, а также для создания файлов данных, под которые он также резервирует первичное пространство и добавляет соответствующую D-запись в словарь файла. Резервируя первичную область хранения, процессор CREATE-FILE автоматически выбирает из свободного пространства на диске блок непрерывных (смежных) фреймов. От пользователя при этом требуется только указать имя создаваемого файла и его модуль (количество первоначально выделяемых под файл фреймов). Подробнее о выборе модуля см. в соотв. разд. настоящей главы.

Нельзя создать файл данных без словаря, на него указывающего, поэтому начинать создание нового файла всегда следует с создания словаря или создавать словарь и раздел данных одновременно. Для файлов, имеющих единственный раздел данных, рекомендуется создавать словарь и раздел данных одновременно. Формат команды:

```
CREATE-FILE имя-файла m1 m2
```

```
CREATE-FILE имя-словаря,имя-файла-данных m1 m2
```

где m1 - модуль словаря, m2 - модуль файла данных. Параметр "имя-файла-данных" - это произвольное имя создаваемого файла данных; он необходим только в том случае, если несколько файлов данных используют общий словарь. Указатель на файл данных помещается в словарь файла.

Файлы, состоящие только из одного словаря, создаются командой:

```
CREATE-FILE DICT имя-файла m1
```

Приставка DICT говорит о том, что создается только словарь. В MD счета при этом заносится соответствующая запись, под словарь резервируется m непрерывных (смежных) фреймов. Еще раз обращаем внимание на то, что если информация предполагается хранить в одноуровневом словарном файле (например, несколько процедур), раздел данных для него создавать не надо.

Если словарь файла уже существует, добавить к файлу новый раздел данных можно в любой момент. Общий формат используемой для этого команды:

```
CREATE-FILE DATA имя-словаря(,имя-файла-данных) m2
```

где приставка DATA указывает на то, что создается именно раздел данных, даже если он будет единственным файлом данных, относящимся к ранее созданному словарю, хотя чаще эта команда используется, когда необходимо добавить еще один файл данных к группе файлов данных, использующие общий словарь. Под создаваемый раздел данных будет выделено m2 непрерывных (смежных) фрейма, а в словарь файла будет помещена запись типа D, указывающая на зарезервированное место на диске. Рассмотрим пример:

```
CREATE-FILE СОТРУДНИКИ 3 373 <BK>
```

Создается новый файл с именем СОТРУДНИКИ, состоящий из словаря и единственного файла данных. Модуль словаря - 3, модуль раздела данных - 373. Одна запись с идентификатором СОТРУДНИКИ будет занесена в MD счета, другая будет добавлена к словарю файла. Другой пример:

```
CREATE-FILE DICT ТЕСТ-ФАЙЛ 7 <ВК>
```

Создается одноуровневый словарный файл с именем ТЕСТ-ФАЙЛ; в MD счета заносится соответствующая запись типа D, в словарь ТЕСТ-ФАЙЛ помещается запись типа D с тем же именем, указывающая на сам этот словарь.

```
CREATE-FILE DICT ЦЕХ 3 <ВК>
```

Создается одноуровневый словарный файл ЦЕХ.

```
CREATE-FILE DATA ЦЕХ,КРАСИЛЬНЫЙ 10 <ВК>
```

Создается новый раздел данных КРАСИЛЬНЫЙ, который будет пользоваться общим словарем ЦЕХ с несколькими другими файлами данных. В словарь файла ЦЕХ будет помещена запись типа D с идентификатором КРАСИЛЬНЫЙ. Впоследствии при обращении к созданному файлу данных придется указывать оба имени: "ЦЕХ,КРАСИЛЬНЫЙ", т.е. указывать и имя словаря, и имя файла данных.

```
CREATE-FILE DATA ЦЕХ,ТКАЦКИЙ 57 <ВК>
```

Создается новый раздел данных - "ТКАЦКИЙ". Ссылаться на него нужно будет по двойному имени: "ЦЕХ,ТКАЦКИЙ".

ЗАМЕЧАНИЕ: если требуется создать новый файл-указатель или файл для хранения текстов байсковских программ, вначале используется глагол CREATE-FILE, а затем с помощью редактора EDITOR код D в первом атрибуте записи в MD счета заменяется на код DC.

2.17. ПРОЦЕССОР CLEAR-FILE

Процессор CLEAR-FILE очищает содержимое заданного файла, но не удаляет из MD счета его D-записи и сохраняет за очищенным файлом его первичную область хранения.

Процессор CLEAR-FILE "очищает" файл, т.е. переводит его в пустое состояние, помещая в первую позицию данных каждой группы файла метку атрибута, означающую в данном случае "конец группы". Фреймы, подсоединенные к первичной области хранения, возвращаются при этом в область переполнения (пространство свободных фреймов). Если очищается только словарь, а относящийся к этому словарю файл или файлы данных сохраняются, все соответствующие D-записи в словаре будут сохранены. Процесс CLEAR-FILE можно прервать нажатием на клавишу BREAK.

Для очищения файла данных используется следующий формат команды:

CLEAR-FILE DATA имя-файла(,имя-файла-данных)

Если раздел данных в очищаемом файле один, указывается только имя файла, если их несколько, указывается и имя словаря (имя файла), и имя раздела данных.

Для очищения словаря файла используется следующий формат команды:

CLEAR-FILE DICT имя-словаря.

Примеры:

CLEAR-FILE DATA СОТРУДНИКИ <BK>

Очищается раздел данных файла сотрудники.

CLEAR-FILE DICT ТЕСТ-ФАЙЛ <BK>

Очищается словарь файла ТЕСТ-ФАЙЛ: удаляются все записи, за исключением записей типа D.

CLEAR-FILE DATA ЦЕХ,КРАСИЛЬНЫЙ <BK>

Очищает раздел данных КРАСИЛЬНЫЙ, относящийся к словарю ЦЕХ.

Примеры использования команды CLEAR-FILE.

2.18. ПРОЦЕССОР DELETE-FILE

Процессор DELETE-FILE используется для удаления словарей и файлов данных из системы.

Процессор DELETE-FILE позволяет удалять двухуровневые пользовательские файлы, состоящие из словаря и одного или нескольких файлов данных, словари (если в них нет D-указателей) и отдельные или все файлы данных, относящиеся к одному словарю. Если словарь содержит указатели (один или несколько) на файлы данных, удалить его нельзя. Все фреймы, занимаемые удаляемым файлом, будут возвращены в пространство свободных фреймов. Процесс DELETE-FILE нельзя прервать клавишей BREAK.

Для удаления словарного файла и всех относящихся к нему файлов данных используется команда

```
DELETE-FILE имя-файла <BK>
```

Для удаления одноуровневого словарного файла (не имеющего подсоединенных к нему файлов данных) используется команда

```
DELETE-FILE DICT имя-файла <BK>
```

В обоих случаях запись-определитель файла (D-указатель) удаляется из главного словаря (MD) счета, а все пространство на диске, им занимаемое, возвращается в пространство переполнения.

Для удаления файла данных используется команда:

```
DELETE-FILE DATA имя-файла(,имя-файла-данных)
```

Эта команда удаляет указатель на соответствующий файл данных из словаря файла и возвращает пространство, занимаемое им, в пространство свободных фреймов. Параметр "имя-файла-данных" необходимо указывать только в том случае, если одному словарю соответствует несколько файлов данных. В качестве параметра глагола DELETE-FILE нельзя указывать имена-синонимы файлов (имена Q-указателей).

DELETE-FILE СОТРУДНИКИ <BK>

Удаляет словарь СОТРУДНИКИ и все относящиеся к нему файлы данных.

DELETE-FILE DIST ТЕСТ-ФАЙЛ <BK>

Удаляется словарь файла ТЕСТ-ФАЙЛ. Данная команда не будет выполнена, если в словаре ТЕСТ-ФАЙЛ имеется хотя бы один D-указатель.

DELETE-FILE DATA ЦЕХ,КРАСИЛЬНЫЙ <BK>

Удаляется раздел данных КРАСИЛЬНЫЙ из сложного файла, образуемого словарем ЦЕХ.

Примеры использования команды DELETE-FILE.

2.19. КОПИРОВАНИЕ: ПРОЦЕССОР COPY

Процессор COPY используется для копирования записей из файлов на терминал, принтер или в другой файл, который может находиться либо в том же, либо в другом счете.

Обращение к процессору COPY происходит через глагол COPY, который относится к глаголам TCL второго уровня. Общий формат команды:

`COPY {DIST} имя-файла список-записей {(опции)}`

Параметр "имя-файла" задает копируемый файл ("источник"); "список-записей" может состоять из одного или нескольких идентификаторов записей, разделенных пробелами, или в этом месте можно указать "звезду" (*), которая будет означать "все записи". Этот параметр задает копируемые записи. Если в глаголе используются опции, они должны быть заключены в круглые скобки (об опциях речь пойдет ниже).

После того, как команда COPY введена, процессор будет реагировать по-разному, в зависимости от того, куда направляется вывод - на терминал, принтер или в файл. Направление вывода задается опциями: T обозначает вывод на терминал, P - на принтер. Если не указан ни один из этих кодов, предполагается копирование в файл.

Если копирование производится не на терминал или принтер, а в файл, процессор выдает на экран запрос:

TO:
(куда)

Общая форма ответа на этот запрос такова:

`{{(DIST} имя-файла)} {список-записей}`

Здесь: 1) имя файла необходимо указывать только в том случае, если копирование производится в другой файл, отличный от файла-источника; имя файла, если оно указывается, должно быть заключено в круглые скобки; имени файла может предшествовать приставка DIST, если копирование производится в словарь; 2) если данные копируются в рамках одного и того же файла, его имя не указывается; 3) если записи копируются под новыми именами (идентификаторами), необходимо указать эти новые имена в виде списка; 4) если в ответ на запрос "TO:" ничего не ввести, а просто нажать на клавишу <BK>, вся выдача пойдет на терминал, как если бы исполнялся глагол COPY с опцией T (см. также следующие разделы).

2.20. КОПИРОВАНИЕ ДАННЫХ ИЗ ФАЙЛА В ФАЙЛ

В разделе излагаются некоторые подробности, связанные с копированием данных из одного файла в другой или в пределах одного и того же файла.

В качестве параметров глагола COPY можно указывать списки записей. Список записей может содержать идентификаторы записей, которые копируются, или, если он вводится в ответ на системный запрос "TO:" (при копировании из файла в файл), новые идентификаторы этих записей в файле, в которой производится копирование. Идентификаторы записей в списке отделяются один от другого пробелами, если они не содержат их внутри себя. Если пробелы входят в идентификатор записи, он должен быть заключен в двойные кавычки.

Например, список записей может быть указан в следующей форме:

```
1024-24 1024-25 "ПЕРВАЯ ЗАПИСЬ" АБВГ
```

Идентификаторы записей в списке могут повторяться. Список копируемых записей и список имен, под которыми они копируются, может быть разное число элементов. Если первым будет исчерпан список копируемых записей, некоторые из новых имен останутся невостребованными. Если первым будет исчерпан список новых имен, остальные записи будут скопированы под своими прежними именами.

Если требуется, чтобы записи были скопированы под своими прежними именами, список новых имен указывать нет необходимости.

Если нужно скопировать все записи файла, вместо списка записей следует указать * (без кавычек).

Если требуется скопировать заранее созданный список записей, в глаголе COPY список записей указывать не нужно, т.к. заранее созданные списки обладают большим приоритетом. В этом случае перед глаголом COPY для создания нужного списка следует исполнить один из глаголов SELECT, QSELECT или GET-LIST (см. описание соотв. глаголов).

При копировании записей из словаря в словарь следует иметь в виду, что процессор COPY игнорирует все D-указатели. Записи типа D можно создавать только через процессор CREATE-FILE. Если требуется скопировать двухуровневый файл целиком, следует использовать последовательность команд, аналогичную приведенной на втором рисунке ниже.

```
>COPY DIST ПРИМЕР ЗАТРАТЫ (I) <BK> <--- копируется одна
ТО: СЕБЕСТОИМОСТЬ <BK> словарьная запись
```

```
СКОПИРОВАНО ЗАПИСЕЙ: 1
```

```
>COPY ПРИМЕР 1242-01 <BK> <----- копируется одна
ТО: 1242-99 <BK> запись данных
```

```
1 1242-01
```

```
СКОПИРОВАНО ЗАПИСЕЙ: 1
```

```
>COPY ЦВЕТА КРАСНЫЙ БЕЛЫЙ СИНИЙ <BK> <-- копируется список
ТО: АЛЬФА БЕТТА ГАММА <BK> записей
```

```
1 КРАСНЫЙ
```

```
2 БЕЛЫЙ
```

```
3 СИНИЙ
```

```
СКОПИРОВАНО ЗАПИСЕЙ: 3
```

Копирование записей в рамках одного файла.

```
>COPY DIST ПРИМЕР * (I) <BK> <----- копируются все
ТО: (DIST ЦВЕТА) <BK> словарьные записи
[418] ЗАПИСЬ-ОПРЕДЕЛИТЕЛЬ ФАЙЛА "ПРИМЕР" НЕ СКОПИРОВАНА
```

```
СКОПИРОВАНО ЗАПИСЕЙ: 2
```

Копирование записей из файла в файл.

```
>CREATE-FILE НОВ-ПРИМЕР 1 3 <BK> <----- создан новый
                                файл
[417] СОЗДАН ФАЙЛ "НОВ-ПРИМЕР". БАЗА = 15417, МОДУЛО=1, СЕПАРАЦИЯ
= 1
[417] СОЗДАН ФАЙЛ "НОВ-ПРИМЕР". БАЗА = 15418, МОДУЛО=1, СЕПАРАЦИЯ
= 1
```

```
>COPY DIST ПРИМЕР * (I) <BK> <----- копируются все словарьные
ТО: (DIST НОВ-ПРИМЕР) <BK> записи, за исключением
                                D-указателей
```

```
[418] ЗАПИСЬ-ОПРЕДЕЛИТЕЛЬ ФАЙЛА "ПРИМЕР" НЕ СКОПИРОВАНА
```

```
СКОПИРОВАНО ЗАПИСЕЙ: 3
```

```
>COPY ПРИМЕР * (I) <BK> <----- копируются все записи данных
ТО: (НОВ-ПРИМЕР) <BK>
```

```
СКОПИРОВАНО ЗАПИСЕЙ: 22
```

Копирование файла целиком, вместе с его словарем и разделом данных.

2.21. КОПИРОВАНИЕ: ОПЦИИ ПРОЦЕССОРА COPY

В разделе описываются опции, которые могут быть указаны в команде COPY, а также способ копирования данных на терминал и на принтер.

ФОРМАТ:

COPY {DIST} имя-файла список-записей {(опции)}

Если в команде указываются опции, они должны быть заключены в круглые скобки. Опции обозначаются буквами латинского алфавита. Если указывается несколько опций одновременно, их можно записывать слитно или через запятую. В нижеприведенной таблице перечисляются опции процессора COPY. Заметим, что в зависимости от того, куда осуществляется копирование - на терминал, принтер или в файл - некоторые опции могут действовать по-разному.

При копировании на терминал или на принтер данные выводятся в следующем формате:

Идентификатор
001 первый атрибут
002 второй атрибут
003 третий атрибут
.....
nnn последний атрибут

Так, запись ЗАПИСЬХ из файла ПРИМЕР будет выведена на терминал в следующем виде:

ЗАПИСЬХ
001 3746
002 ИВАНОВ П.В.
003 МОСКВА

ОПЦИЯ	ПРИМЕЧ.	ОПИСАНИЕ
D	1	Удалить запись. После копирования исходная запись удаляется.
F	2	Перевод страницы; каждая новая запись будет начинаться с новой страницы.
I	1	Не выводить имена записей.
N	1	Запрет на создание новых записей: копирование будет произведено только в том случае, если в файле уже существуют записи с такими названиями.
	2	Отмена остановок между страницами.
O	1	Перезапись ранее созданных записей.
P		Копирование на принтер.
S	1	Отмена вывода системных сообщений: не будут выдаваться сообщения о том, что записи не скопированы (409, 415 и 418).
	2	Отмена вывода номеров строк.
T		Копирование на терминал.
X	1	Вывод в шестнадцатеричном формате.
ПРИМЕЧАНИЯ:	1.	Действительно только при копировании из файла в файл.
	2.	Действительно только при копировании на терминал или на принтер.

Опции процессора COPY.