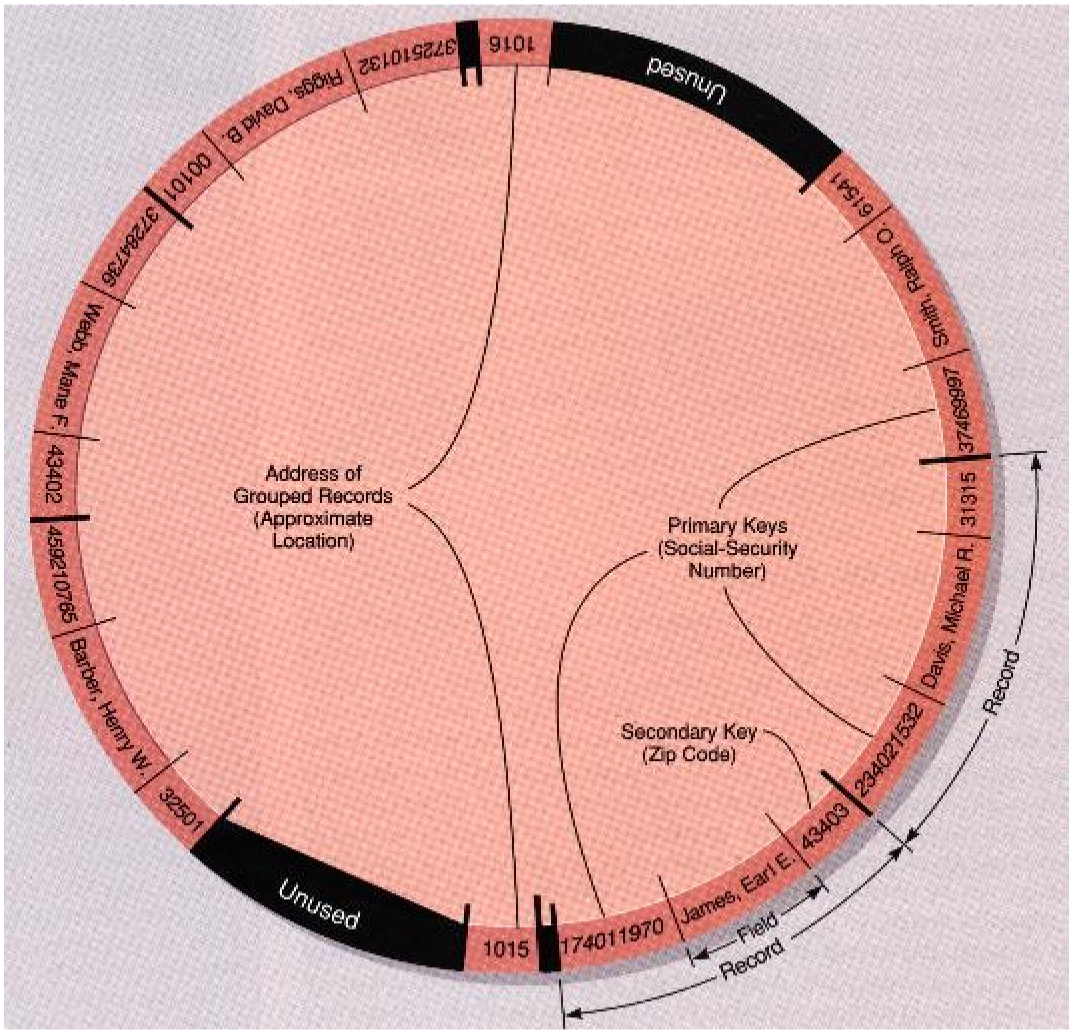
Indexed-Sequential File Design

Sequential processing is suitable for applications in which the proportion of records processed in an updating run is high. However, sequential files provide slow response times and cannot adequately handle file inquiries. On the other hand, direct-access processing is inappropriate for applications like payroll, where most records are processed during a single run. When a single file must be used for both batch processing and online processing, neither direct-access nor sequential file organization is appropriate. The same customer file that is used in a weekly batch run for preparing bills by the accounting department may be used daily by order entry personnel to record orders and check credit status. To some extent, the limitations of both types of file design can be minimized by using another approach to file organization, **indexed-sequential** design.

In this structure, the records are stored sequentially on a direct-access storage device according to a primary key. A **primary key** is a field that will be unique for each record on the file. In addition, secondary keys can also be established. **Secondary keys** are fields that are used to gain access to records on the file but may not be unique. For instance, if zip code is chosen as a secondary key, there may be several records with the same value. Records on an indexed-sequential file can be accessed randomly by using either the primary or one of the secondary keys, or the file can be read sequentially, in order according to primary key.

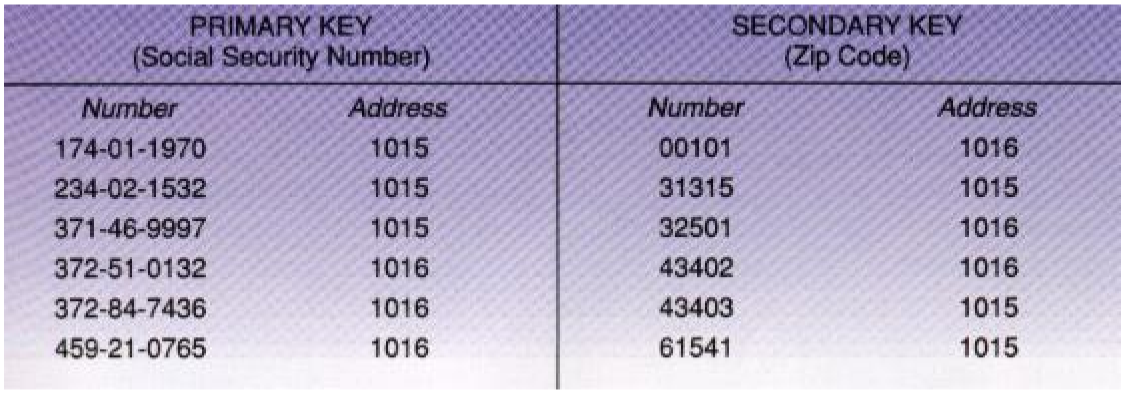
The method used to gain access to a record on an indexed-sequential file is a little different from the method used for a direct-access file. Every record on an indexed-sequential file may not have its own unique address. Rather, several records may be grouped together and one address given for the entire group. An index table is created for all fields that are primary or secondary keys. The index table lists the value of the key (such as social security number) and the corresponding address on the direct-access storage device at which the group containing that record can be found. (The index table can either be stored at the beginning of the file or a separate file of indexes may be created.) A key given by the user is matched against the index table to get an approximate address for the required record. The computer then goes to that location on the direct-access storage device and checks records sequentially until the desired record is found. In the case of secondary keys, all records with that key may be retrieved.

The Figure below shows the employee file from the previous section set up as an indexed-sequential file.



Indexed-Sequential Design

The primary key is the social security number, while zip code is a secondary key. Notice how the records are in sequence according to the social security number on the file. To locate an employee with a zip code of 43403, the computer goes to the index table for zip code (see Figure below).



PRIMARY KEY (Social Security Number)		SECONDARY KEY (Zip Code)	
Number	Address	Number	Address
174-01-1970	1015	00101	1016
234-02-1532	1015	31315	1015
371-46-9997	1015	32501	1016
372-51-0132	1016	43402	1016
372-84-7436	1016	43403	1015
459-21-0765	1016	61541	1015

Index Tables of Primary and Secondary Keys

Next to the value 43403 is the address on the direct-access storage device at which the group containing the record can be found, 1015. The computer goes to that address and reads each record in the group until the one with zip code 43403 is found. In this case, it is the first record in the group.

Thus, an indexed-sequential file provides direct-access capability. Since all the records are ordered according to a primary key, it also allows efficient sequential processing.

MAKING INQUIRIES TO INDEXED-SEQUENTIAL FILES.

The file could be read sequentially for a billing operation. In addition, it could be accessed one record at a time for order-entry transactions.

ASSESSMENT OF INDEXED-SEQUENTIAL FILE DESIGN

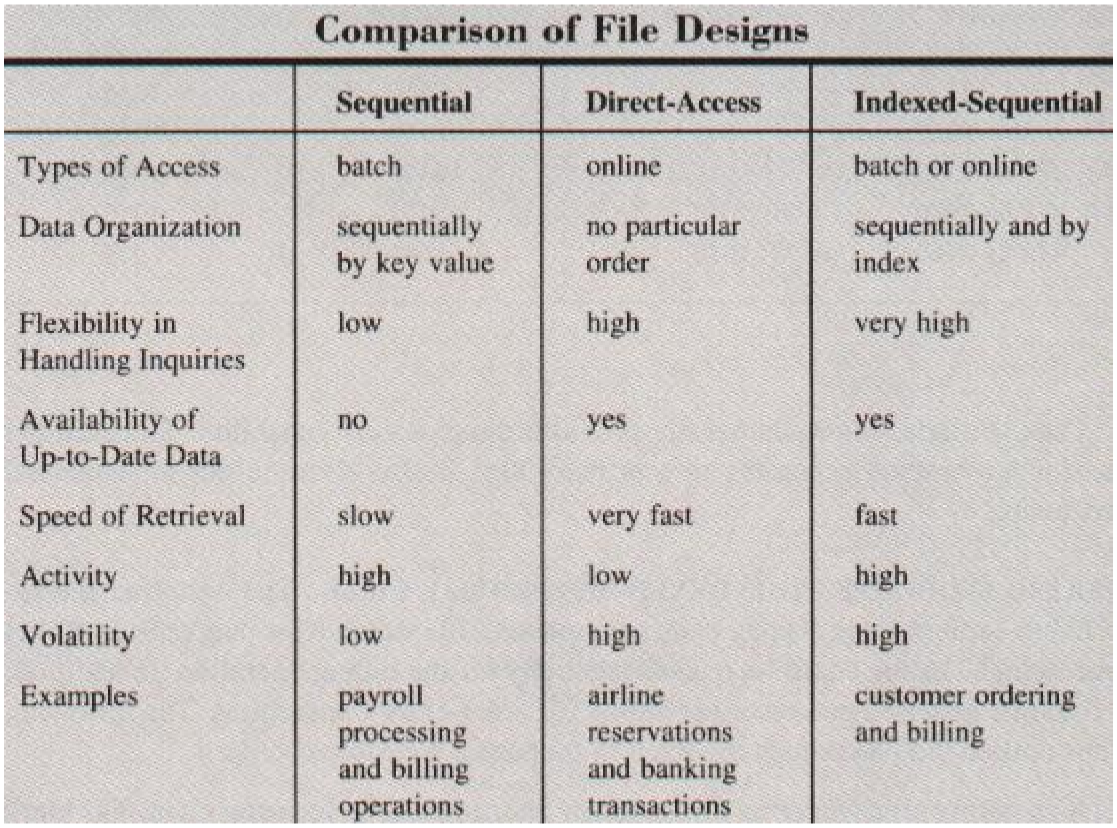
Indexed-sequential files have a built-in flexibility that is not available with either sequential or direct-access designs. They work well in an environment where transactions are batch processed and inquiries require the fast response of direct-access processing.

Advantages of indexed-sequential design include the following:

- Indexed-sequential files are well suited for both inquiries and large processing runs.
- Access time to specific records is faster than it would be if the file were sequentially organized.

Disadvantages of indexed-sequential design include the following:

- More direct-access storage space is required for an indexed-sequential file than for a sequential file holding the same data because of the storage space required for indexes. Therefore this type of system is more costly.
- Processing time for specific record selection is longer than it would be in a direct-access system.



Comparison of File Designs

	Sequential	Direct-Access	Indexed-Sequential
Types of Access	batch	online	batch or online
Data Organization	sequentially by key value	no particular order	sequentially and by index
Flexibility in Handling Inquiries	low	high	very high
Availability of Up-to-Date Data	no	yes	yes
Speed of Retrieval	slow	very fast	fast
Activity	high	low	high
Volatility	low	high	high
Examples	payroll processing and billing operations	airline reservations and banking transactions	customer ordering and billing

Last Updated Jan.6/99