

Introduction to IBM i

Keyed Files



Objectives

- Explain:
 - ◆ File organization and access methods
 - ◆ Relative record numbers
 - ◆ Primary and secondary keys
 - ◆ Sequential, and indexed sequential organization
 - ◆ Multilevel indices
 - ◆ Access paths and arrival sequence

File Organization

- File organization
 - ◆ Manner in which data records are stored in a file
- Two types of file organization
 - ◆ Sequential
 - ◆ Indexed Sequential

Access Methods

- Sequential

- ◆ Storage device traverses all storage locations that physically precede the storage location being sought

- Direct

- ◆ Storage device goes directly to the storage location being sought

Identifying Records

- **Relative Record Number**
 - ◆ Each record is given a number based on its relative position within the file
- **Primary Key**
 - ◆ A field or fields that can be used to identify a record. The value is unique for each record
- **Secondary Key**
 - ◆ Field or fields that can be used to identify a record
 - ◆ The value does not have to be unique for each record

Sequential File Organization

- Data records are stored one after another in the order they are written
- A key can be defined for a file and records can be written in key order
- To access data:
 - ◆ Records are read sequentially until a records key value matches the value being searched for
- Identifying a particular record in a non-keyed sequential file is done with a relative record number

Advantages and Disadvantages of Sequential Files

- Accessing a single record in a large file is very slow
- Accessing a large percentage of records (especially in key order) is very fast
- Uses a minimal amount of storage space

Indexed Sequential Files Characteristics

- A primary key is defined for the file
- Records are grouped in blocks
- Each block contains records for a certain range of key values
- Within the block, the records are stored sequentially in key order
- Blocks do not have to be in key order

What is an Index?

- An index contains a record for each block of data records
- The index record contains two fields
 - ◆ The highest key value in the block
 - ◆ The starting storage location/address of the block
- Index records are stored in key order
- The number of index records is much smaller than the number of data records
- Index records are smaller than data records

How to Access Indexed Seq Files

- Read the index sequentially
- Each index record's key is checked against the value being searched for
- When the index value \geq the value being searched for, the block that contains the data record has been found
- The system goes directly to the block and sequentially reads each data record until the record key value equals the value being searched for

Advantages of Indexed Sequential File

- Retrieving a single record much faster
- Entire file can still be accessed sequentially by key

Disadvys of Indexed Seq Files

- Indexed sequential files cannot be stored on tape
- Indexed sequential files take up more space because of the index
- Sequential processing of an indexed file will be slower than processing a sequential file

Arrival Sequence

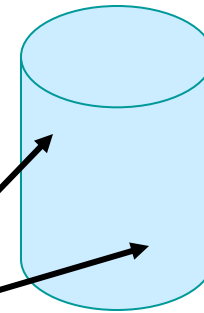
- DB2 supports access to records in the order they are written to a file (i.e. arrival sequence)
- Records are not physically stored in arrival order, they are usually scattered across physical devices
- A user or program can access records in the order in which they were written to a file through an ASAP
- Arrival sequence access is similar to sequential access

Access Paths

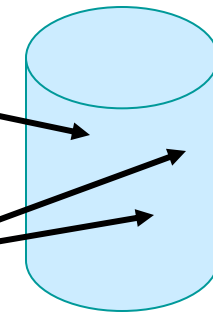
- File access is provided by an access path
- An access path is similar to an index
- An arrival sequential access path could simply consist of a list of storage locations

ASAP

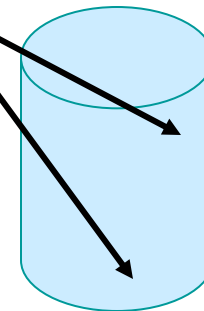
1	
2	
3	
4	
5	
6	
7	



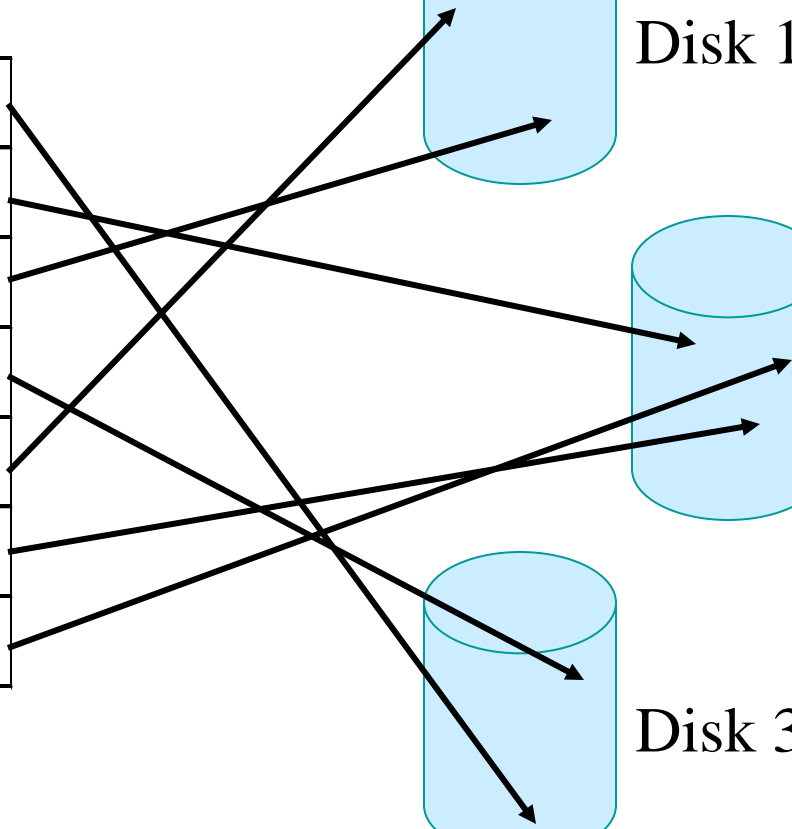
Disk 1



Disk 2



Disk 3



Accessing Records Using An ASAP

- An arrival sequence access path allows users to access data in sequential order by relative record number
- Retrieving a record for a specific key value can be done by either sequentially searching the records or using a hashing routine to yield a relative record number

Keyed Sequence Access Paths

- Allow access to records in both arrival sequence and keyed sequence
- Allow individual records to be retrieved according to a specific key value
- Utilities will default to either Arrival or Keyed sequence when accessing a keyed file
 - ◆ DSPPFM uses arrival sequence
 - ◆ CPYF uses keyed sequence

Multilevel Index

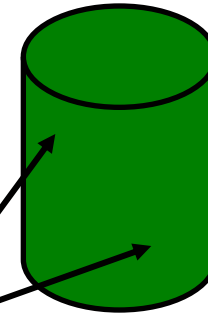
- An index for the index
- The higher level index points to the arrival sequence access path, not the data record

Higher Level Index

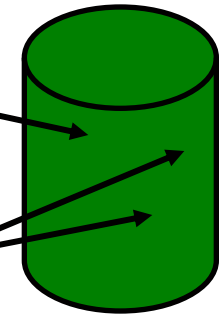
ASAP

Adams	
Baker	
Cole	
Dunne	
Elder	
Frost	
Grey	

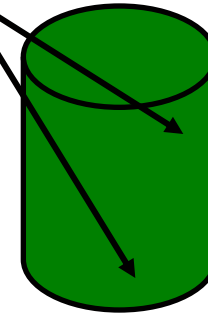
1	
2	
3	
4	
5	
6	
7	



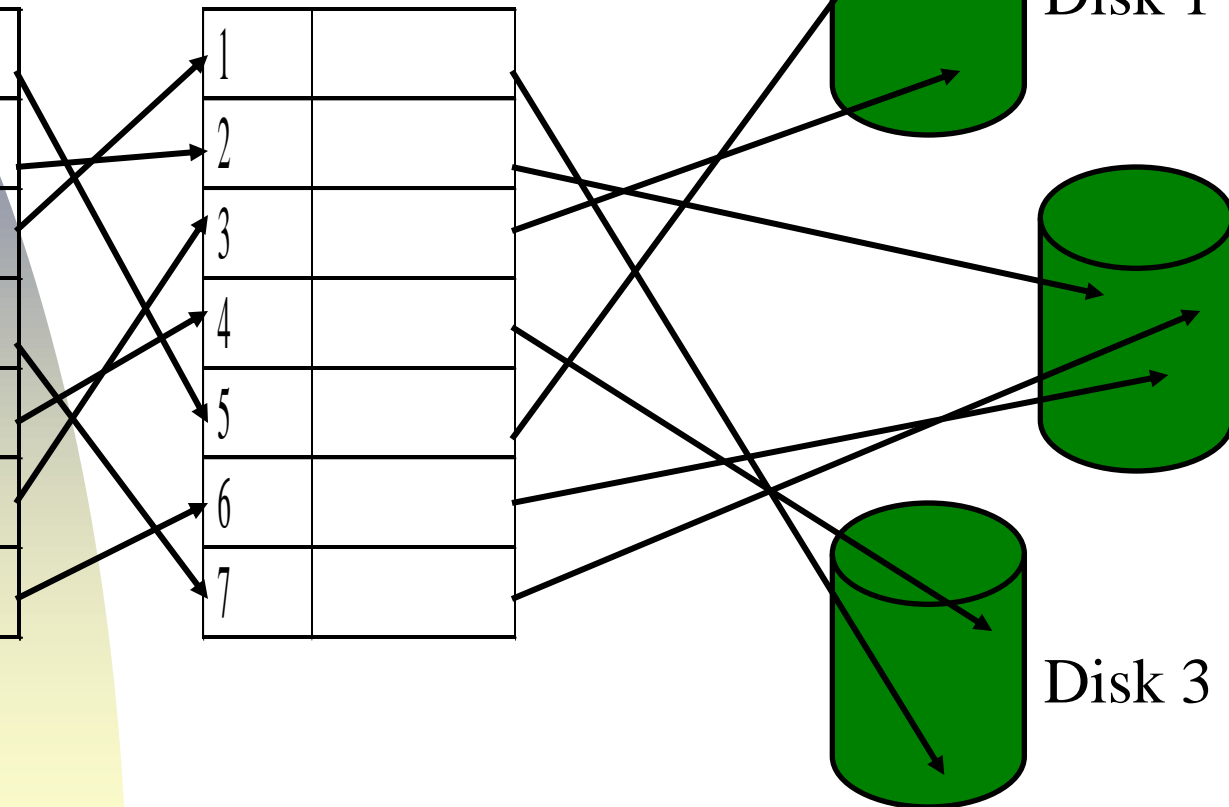
Disk 1



Disk 2



Disk 3



Using a Multilevel Index

- The high-level index is searched for the key value
- When the index record containing the value is found, the ASAP record location will be in the index record
- The system goes directly to the indicated storage location of the ASAP record, and reads the storage location of the data
- The system goes directly to the data storage location, and reads the data

KSAP

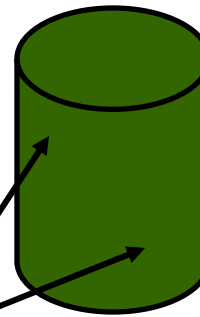
- The higher level index and the arrival sequence access path comprise the keyed sequence access path
- Users access the file either through the keyed index
- Or, through the arrival sequence access path
- The user never accesses the data records directly

H LI

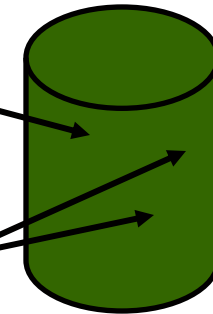
ASAP

Adams	5
Baker	2
Dunne	7
Jones	1
Logan	6
Nash	4
Wynn	3

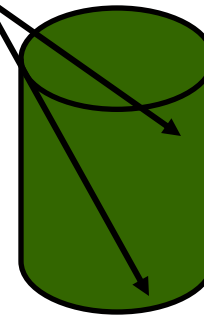
1	
2	
3	
4	
5	
6	
7	



Disk 1

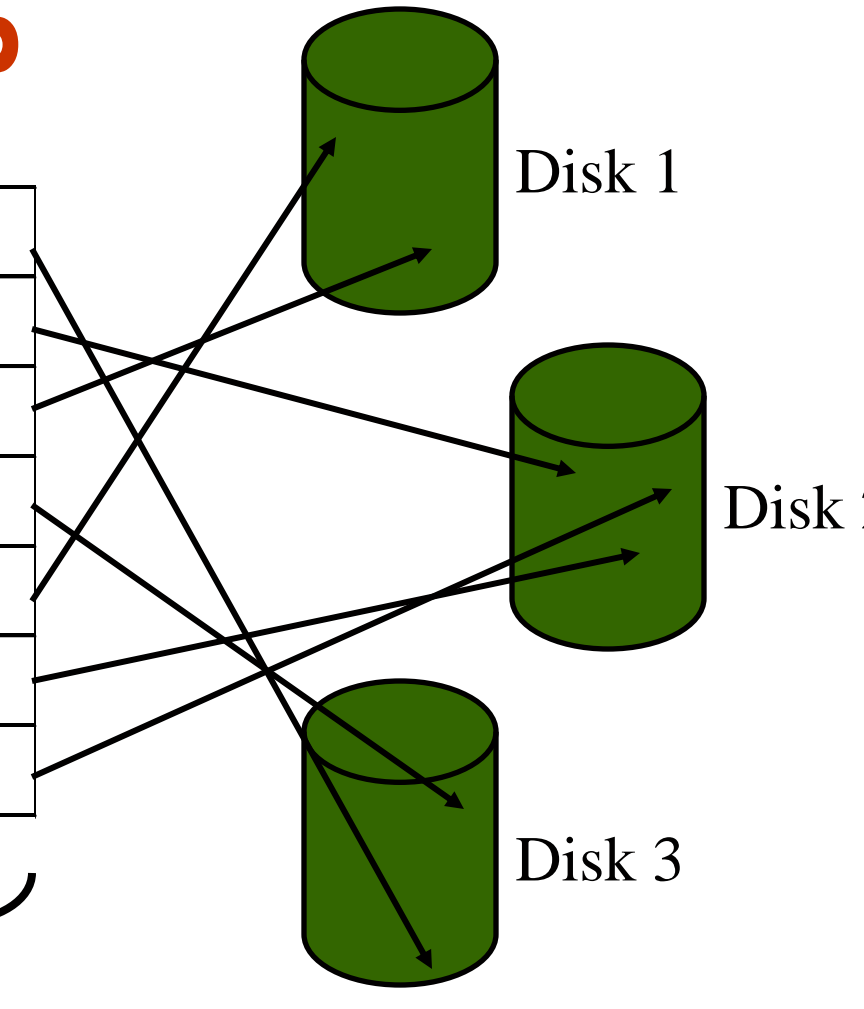


Disk 2



Disk 3

KSAP



Defining a Key using DDS

REF (FRF)

R	EMPREC	
	LNAME	R
	FNAME	R
	GENDER	R
	SALARY	R
	DEPT	R
K	SALARY	

Key specification:

K in column 17

Follows last field specification

Key field must also have a field specification

Defining a Multi-field Key

REF (FRF)

R EMPREC
LNAME R
FNAME R
GENDER R
SALARY R
DEPT R

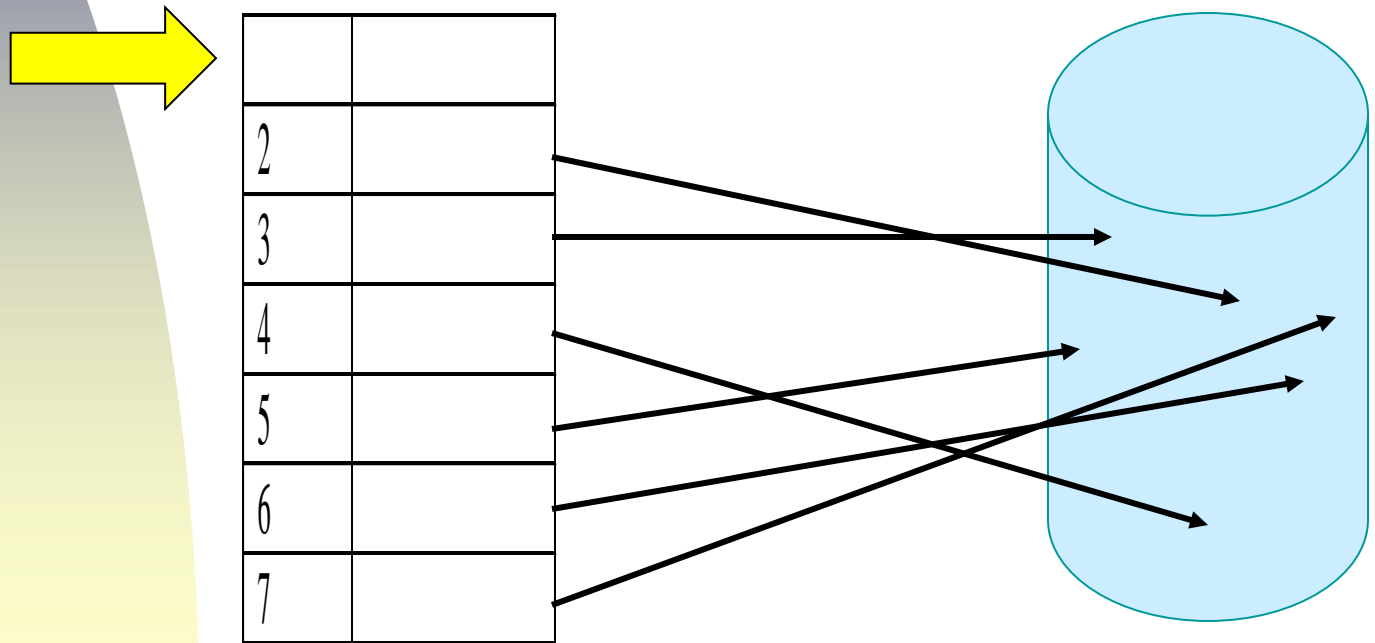
K LNAME
K FNAME
K DEPT

DESCEND

Jones	Alan	M	30000	1234
Jones	Betty	F	42000	1333
Jones	John	M	36000	9333
Jones	John	M	38000	4444

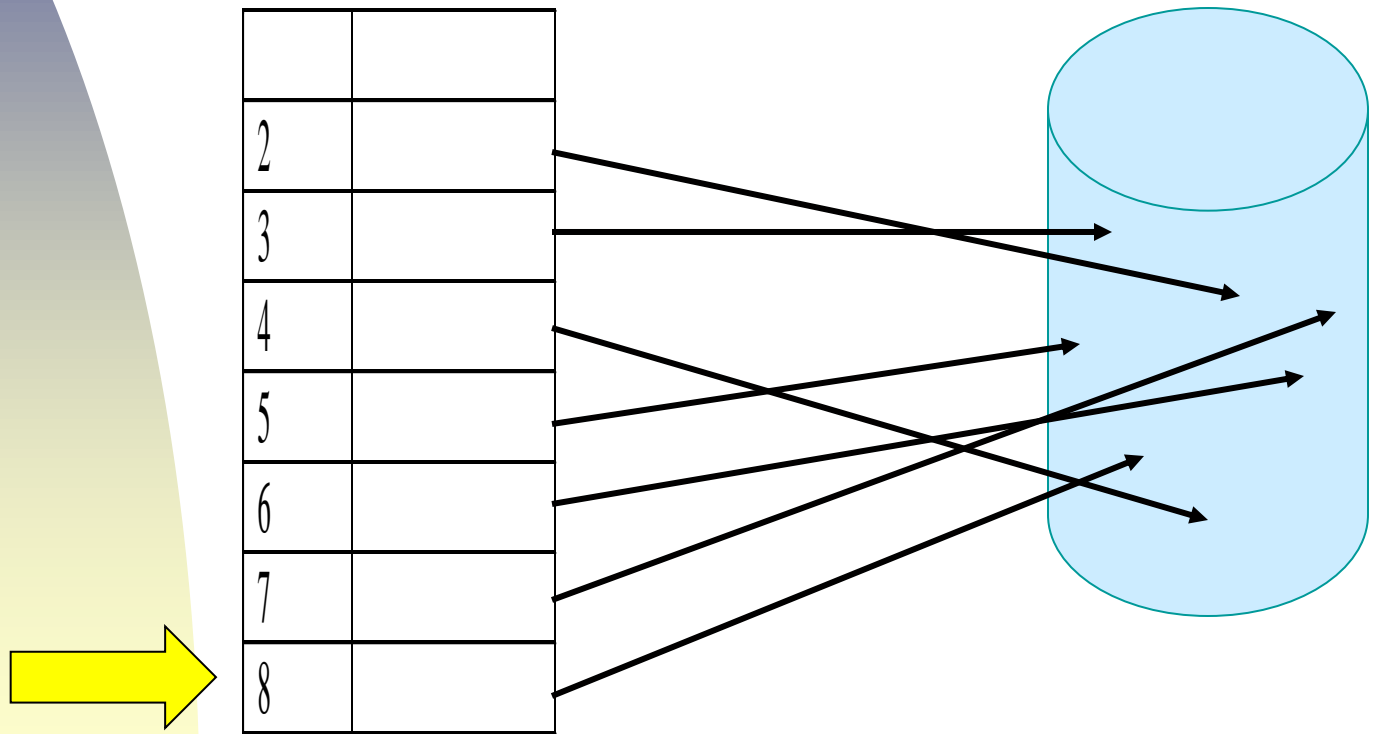
File Space Reclamation

- Deleting records does not reduce the size of a file
- When a record is deleted it is simply no longer accessible (E.g. record 1)



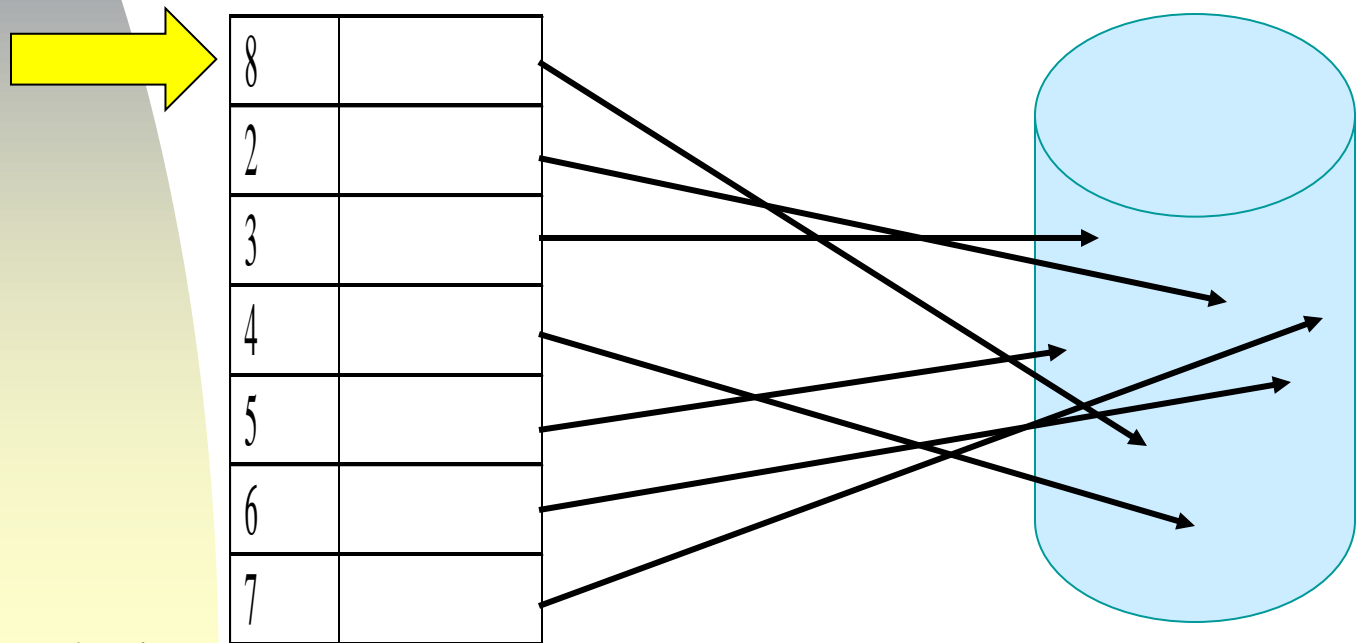
File Space Reclamation

- When a record is added the default is to put it at the end of the file



File Space Reclamation

- However, a files REUSEDLT parameter controls whether deleted space is used
- If CHGF filename REUSEDLT(*YES) then record 8 goes into the empty space



Problems with REUSEDLT(*YES)

- Arrival sequence access to the file is lost
- Adds and deletes take longer
- The file will contain empty space if there are more deletes than adds
- The file stays the same size or grows with each add
- **Solution: RGZPFM**

RGZPFM

- Preserves a files arrival sequence
- Eliminates any empty space

BEFORE

2	
3	
4	
5	
7	
8	

AFTER

2	
3	
4	
5	
7	
8	

Points to Remember

- We covered two data organization methods: sequential and indexed sequential
- There are two primary access methods: sequential and direct
- In DB2, files are accessed using:
 - ◆ Arrival Sequence Access Paths
 - ◆ Keyed Sequence Access Paths
- An access path is similar to an index