Storage of Databases

- Need data to be stored “permanently” or *persistently* for long periods of time

File Organizations

- File organization refers to the way records are stored in terms of blocks and the way blocks are placed on the storage medium.
- Types of organizations
  - Unsorted
  - Sorted
  - Hashing

Records

- A file is a *sequence* of records
- Records could be either *fixed-length* or *variable-length*
- Records comprise of a sequence of *fields* (column, attribute)

Blocks

- A block is **the largest contiguous amount of disk space that can be allocated to a file** and is therefore the largest amount of data that can be accessed in a single I/O operation.

Blocking Factor

The number of records that are stored in a block is called the “blocking factor”. Blocking factor is constant across blocks if record length is fixed, or variable otherwise.

If B is block size and R is record size, then blocking factor is:

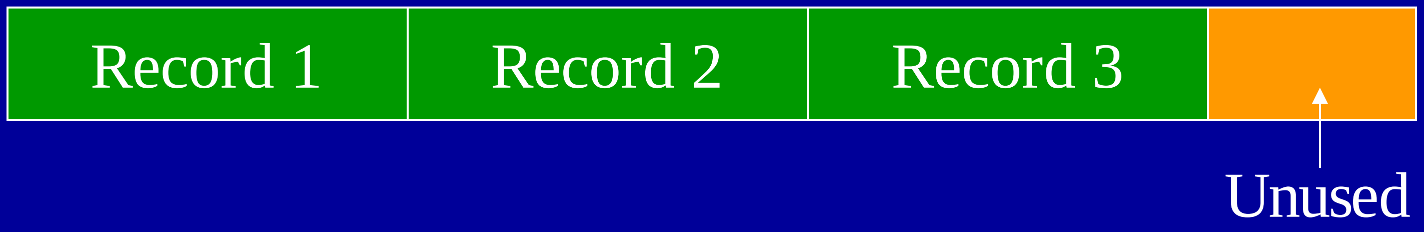
$$\mathrm{bfr} = \lfloor B/R \rfloor$$

Since R may not exactly divide B, there could be some left-over space in each block equal to:

$B - (\mathrm{bfr} * R)$ bytes.

Spanned and Unspanned Records

When extra space in blocks are left unused, the record organization is said to be “unspanned”.



Record 1

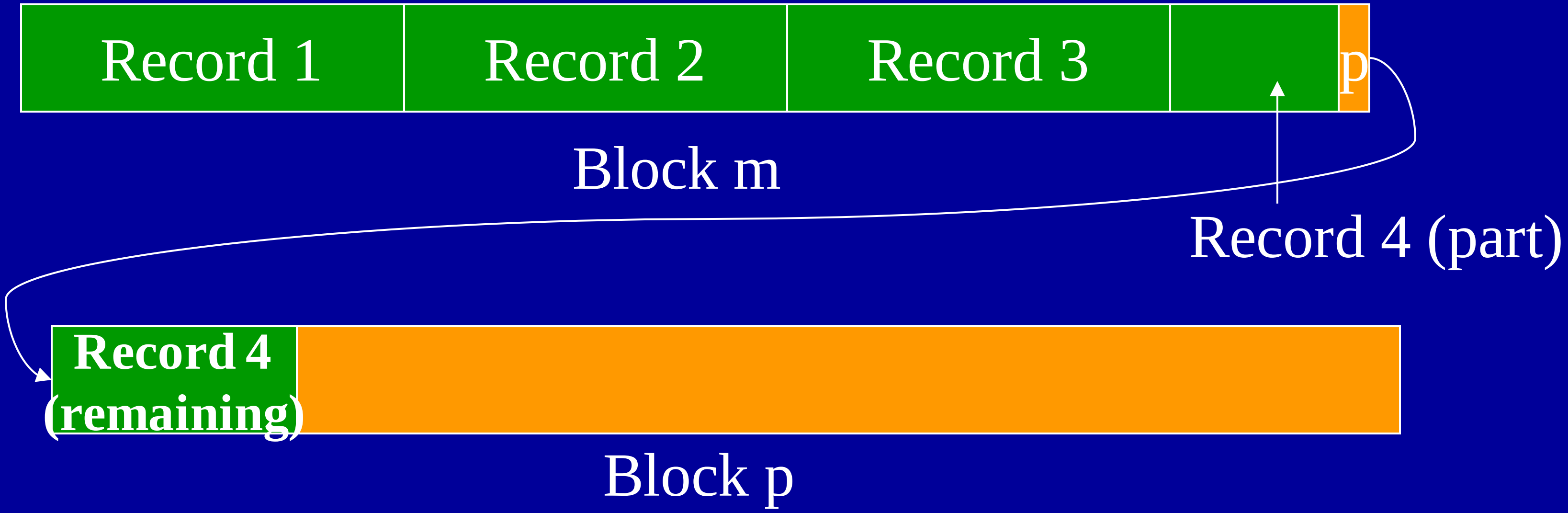
Record 2

Record 3

Unused

Spanned and Unspanned Records

In “spanned” record storage, records can be split so that the “span” across blocks.



Record 1

Record 2

Record 3

p

Block m

Record 4 (part)

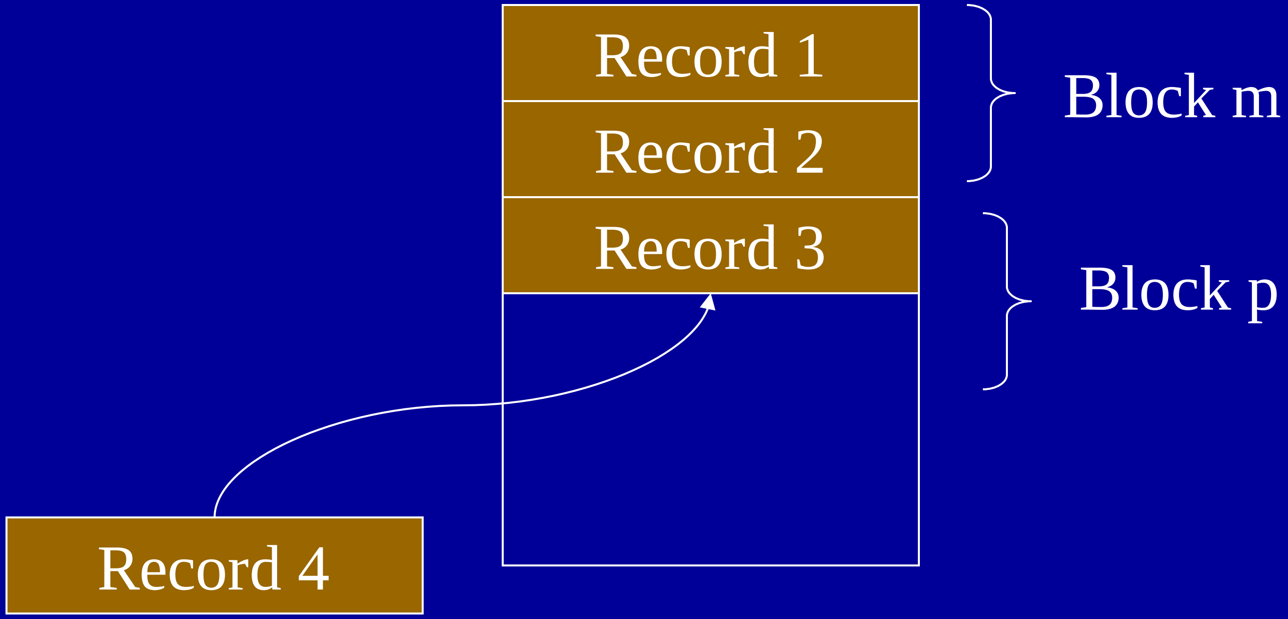
Record 4 (remaining)

Block p

Spanned and Unspanned Records

When record size is greater than block size (i.e. R > B), use of “spanned” record storage is compulsory.

Unordered File Organization

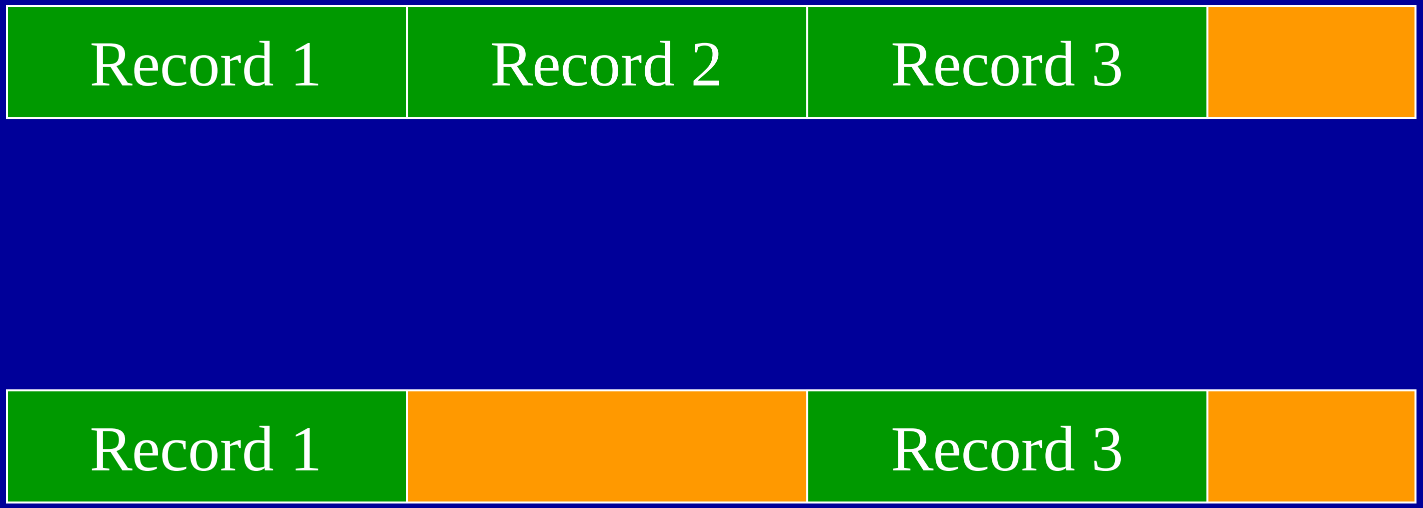


Also called a “pile file”…

Pile Files

- Simplest file organization
- Records are inserted in the order of their arrival
- Insertion: very easy

Record Deletion in Pile Files

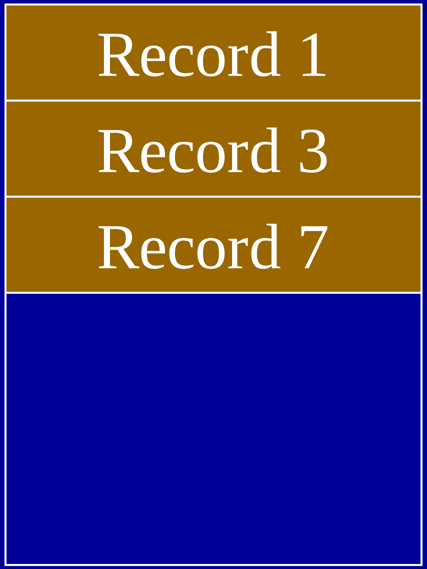


Deletion by fragmentation.
Inefficient in terms of space usage.

Sorted Files

- File organization where records are sorted based on the value of some field called the ordering field
- Ordering field should be a key field (unique for each record) and belong to an ordinal domain (where values can be compared)
- Insertion and Deletion: both expensive
- Updation may involve physical migration of record, if ordering field is modified
- Searching: Binary search based on ordering field efficient

Sorted Files



Record 1

Record 3

Record 7

(Sorted)

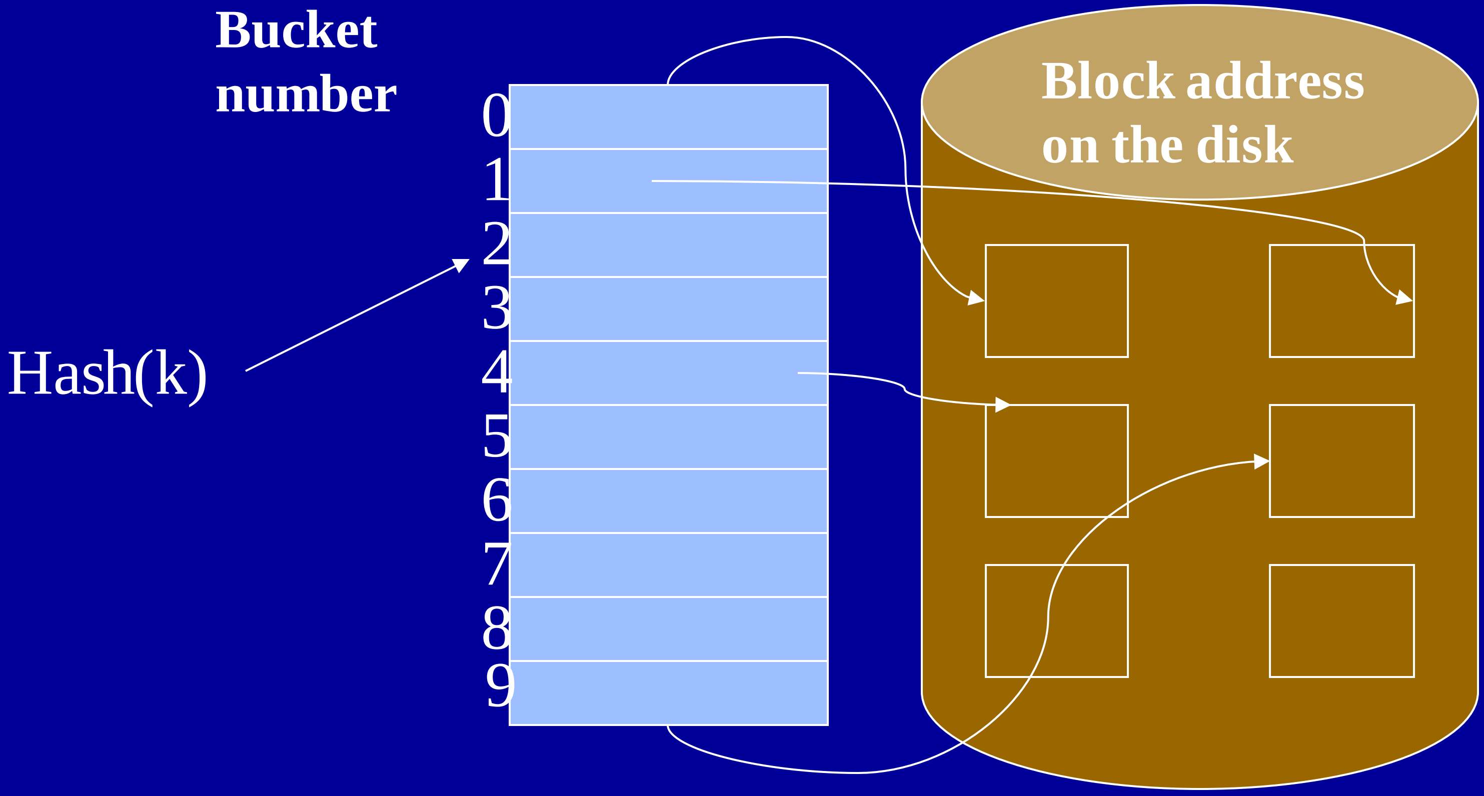
Sorted Files

- More efficient than pile-files for key-based searches
- A better choice when database is mostly read-only and queries are mostly key-based retrievals

Hashing Techniques

- Provide very fast access to records on certain search conditions
- Search condition should be an equality condition on a “key” field
- Uses a “hashing function” to map keys onto “buckets” hosting records

Hashing



Hashing

- Uses two levels of indirection: hashing to buckets and searching within buckets
- A bucket is a disk block or a set of contiguous blocks
- A bucket can hold more than one record
- Sequential search within buckets

Overflows

Occurs when a bucket receives more records than it can hold. Overflow management techniques:

- Open addressing
  - Use the next available bucket.
- Chaining
  - Pointers to extra buckets called overflow buckets.
- Rehashing
  - Use a second hashing function to see if it can hash without overflow.

Dynamic Hashing

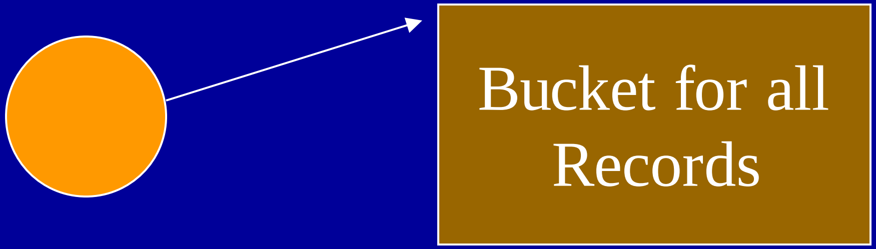
- Hashing where the number of buckets are fixed are called *static* hashing
- Inefficient when data set is skewed. Few overflowing buckets and large number of nearly empty buckets
- Extensible or Dynamic hashing: number of buckets change dynamically in accordance with data requirements

Dynamic Hashing

- No. of buckets may grow or shrink depending on additions and deletions
- Overall strategy:

1. start with a single bucket.
2. Once full, split the bucket into two.
3. Redistribute records between the two split buckets using some method ensuring near-uniform distribution

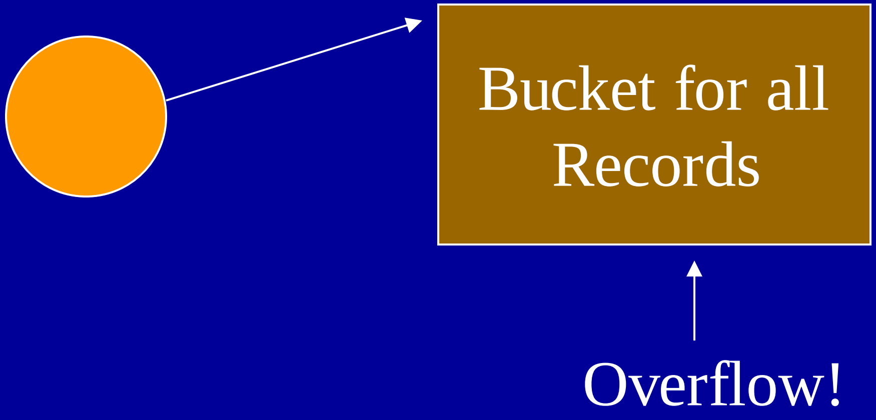
Dynamic Hashing



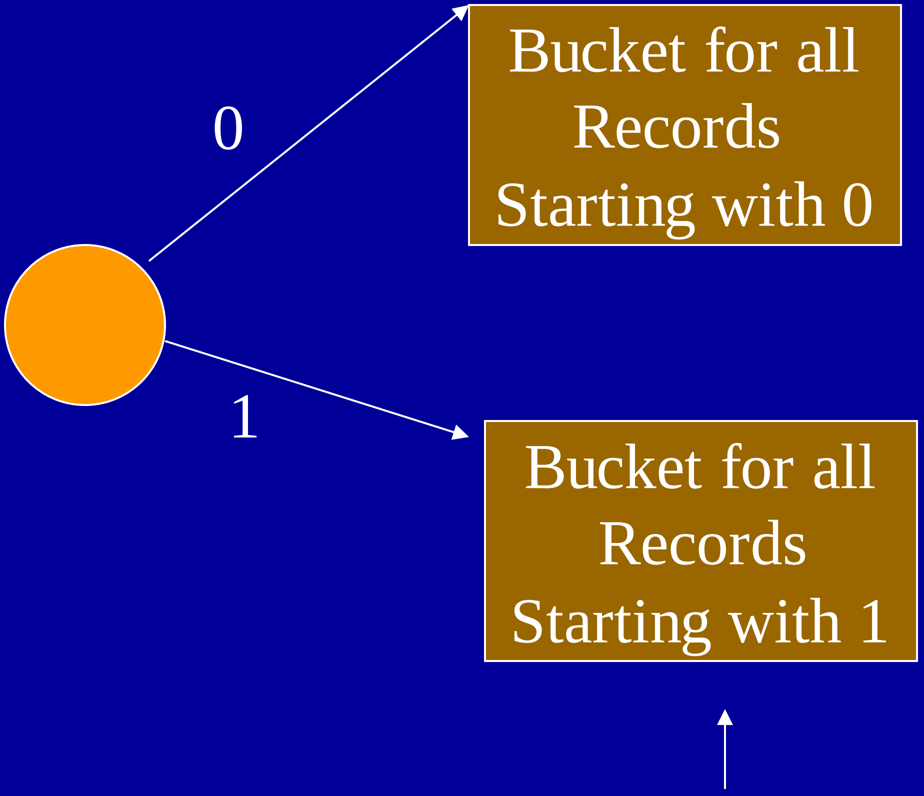
Bucket for all Records

Assume keys to be made of binary strings…

Dynamic Hashing



Dynamic Hashing



0

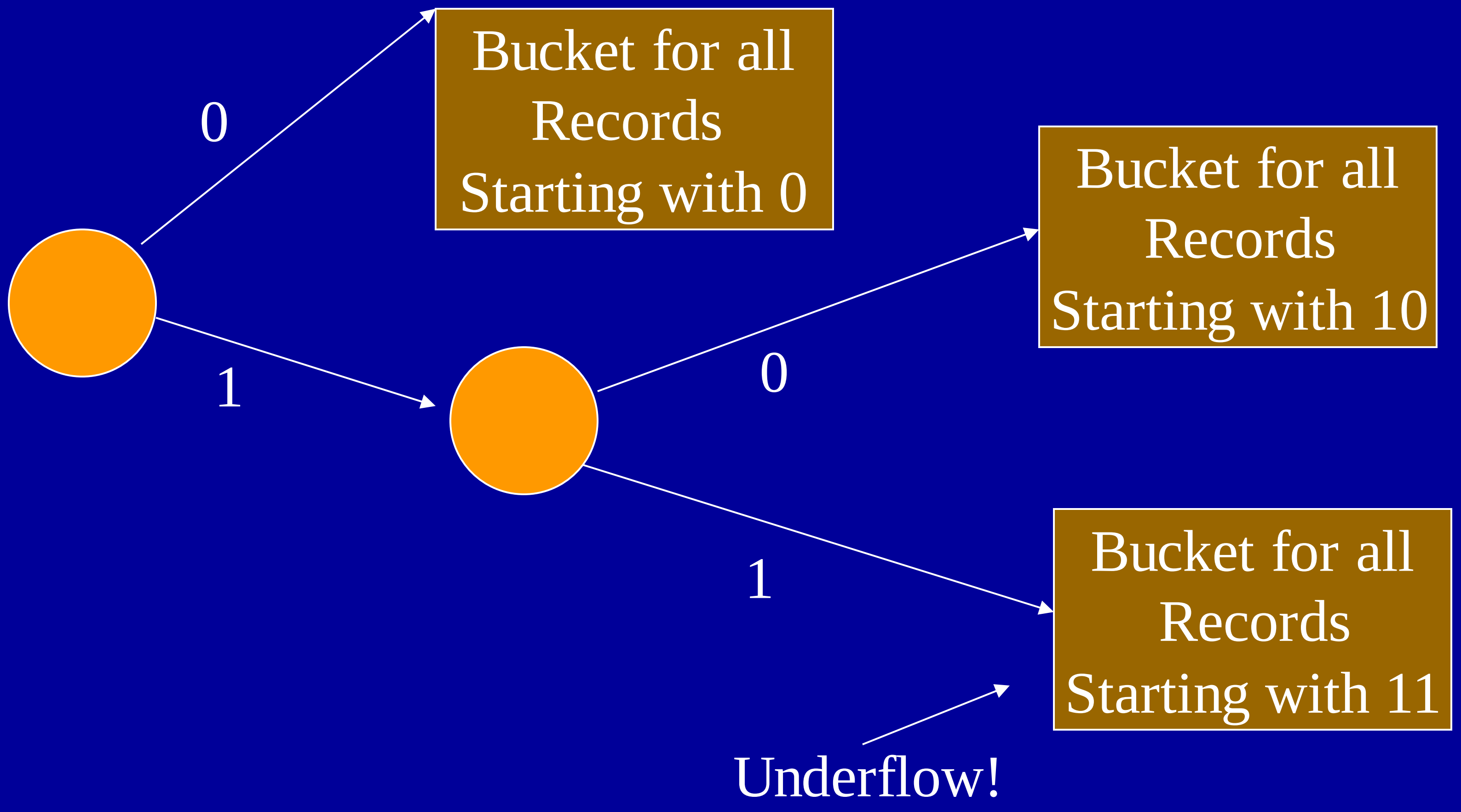
1

Bucket for all Records Starting with 0

Bucket for all Records Starting with 1

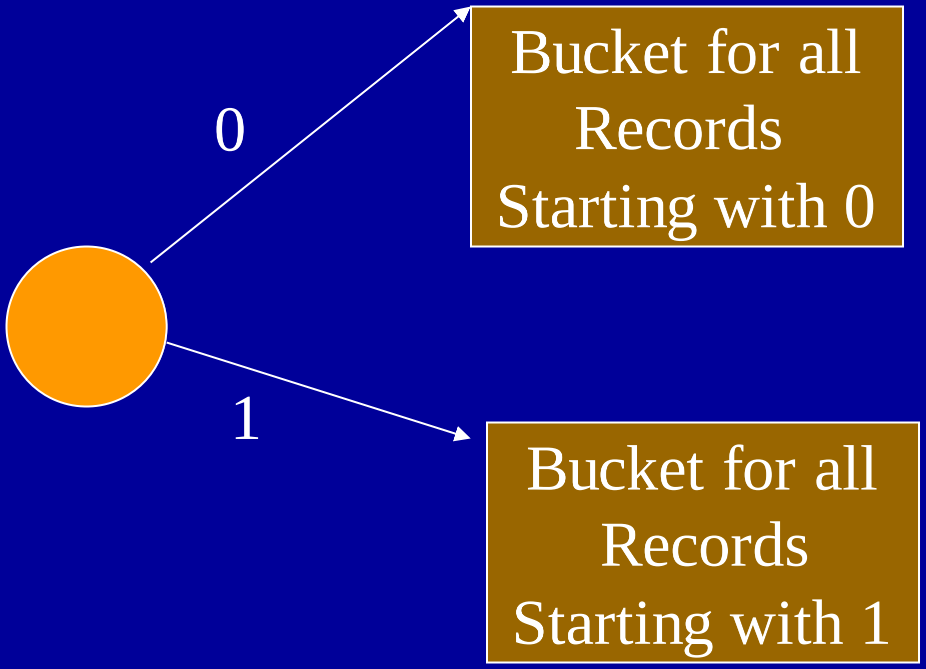
Overflow!

Dynamic Hashing



0
Bucket for all Records Starting with 0
1
Bucket for all Records Starting with 10
0
1
Bucket for all Records Starting with 11
Underflow!

Dynamic Hashing



Bucket for all Records Starting with 0

Bucket for all Records Starting with 1

Summary

- Storage Media and their characteristics
- Records, Blocks and Files
- Spanning and Unspanning records
- Pile Files
- Sorted Files
- Hash Files
  - Static versus dynamic hashing