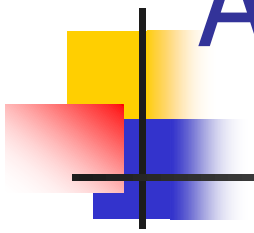
Arquivos: Organização X Acesso



Organização Física:

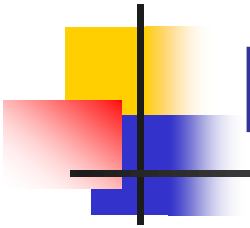
- registros de tamanho variável
- registros de tamanho fixo

Acesso ao Arquivo:

- Sequencial
- Direto

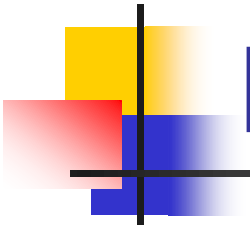
O que influencia:

- o uso que se fará do arquivo
- facilidades da linguagem de programação usada



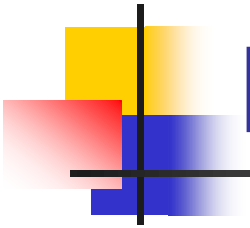
Para o usuário:

- Foco no conteúdo do arquivo, e não no seu formato;
- Acesso à informação, e não a registros e campos;
- Distância maior entre a organização física e lógica do arquivo



Modelos Abstratos de Dados

- Focar no conteúdo da informação, ao invés de no seu formato físico
  - As informações atuais tratadas pelos computadores (som, imagens, documentos, etc.) não se ajustam bem à metáfora de dados armazenados como sequências de registros separados em campos

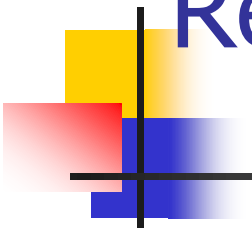


Modelos Abstratos de Dados

- É mais fácil pensar em dados deste tipo como objetos que representam som, imagens, etc. e que têm a sua própria maneira de serem manipulados

- O termo **modelo abstrato de dados** captura a noção de que o dado não precisa ser visto da forma como está armazenado - ou seja, permite uma visão dos dados orientada à aplicação, e não ao meio no qual eles estão armazenados

Registro Cabeçalho (*header record*)

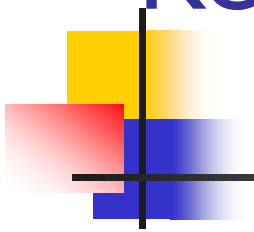


- Em geral, é interessante manter algumas informações sobre o arquivo para uso futuro
- Essas informações podem ser mantidas em um cabeçalho no início do arquivo
- A existência de um registro cabeçalho torna um arquivo um objeto auto-descrito
  - O software pode acessar arquivos de forma mais flexível

Registro Cabeçalho (*header record*)

- Algumas informações típicas
  - Número de registros
  - Tamanho de cada registro
  - Nomes dos campos de cada registro
  - Tamanho dos campos
  - Datas de criação e atualização

Registro Cabeçalho (*header record*)

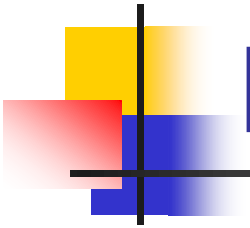


- Vantagem dessas abordagem
  - Podemos criar um programa que lê/escreve um grande numero de arquivos com diferentes características (número de campos por registro, comprimento de campos)
  - Quanto mais informações houver no header, menos o o programa precisa saber sobre a estrutura específica de um arquivo em particular
- Desvantagem
  - Programa que lê/escreve mais sofisticado para interpretar diferentes headers

Metadados

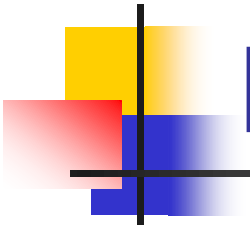
- São dados que descrevem os dados primários em um arquivo
- Exemplo: Formato FITS (*Flexible Image Transport System*)
- Armazena imagens de astronomia
- Cada imagem é precedida por um cabeçalho FITS: uma coleção de blocos de **2880 bytes** contendo registros de **80 bytes** ASCII, com dados sobre a imagem: posição do céu, data de captura, telescópio usado, etc. São chamados metadados
- O FITS utiliza o formato ASCII para o cabeçalho e o formato binário para os dados primários

```
SIMPLE = T / Conforms to basic format
BITPIX = 16 / Bits per pixel
NAXIS = 2 / Number of axes
   ...
DATE = '22/09/1989 ' / Date of file written
TIME = '05:26:53' / Time of file written
END
```



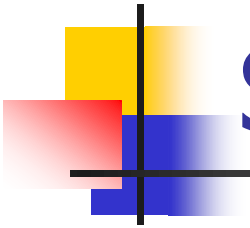
Metadados

- Vantagens de incluir metadados junto com os dados
  - Torna viável o acesso ao arquivo por terceiros (conteúdo auto-explicativo)
  - Portabilidade
    - Define-se um padrão para todos os que geram/acessam certos tipos de arquivo
    - PDF, PS, HTML, TIFF
    - Permite conversão entre padrões



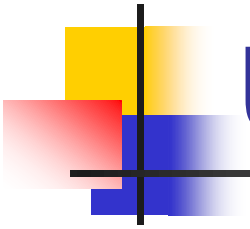
Metadados

- Bom uso para etiquetas e palavras-chave
  - *keyword=value*
  - Espaço ocupado relativo é muito pequeno em FITS: 0.02%
- Se bem descrito, arquivo pode conter muitos dados de formatos e origens diferentes
  - Acesso orientado a objetos
  - "Extensibilidade"



Suponha que:

- O astrônomo resolvesse associar um documento (notas) com suas observações sobre cada imagem.
- Assim, haveria 3 registros de tamanhos variáveis (header, notas, imagem) associados ao objeto.
- Generaliza-se a noção de keywords para um arquivo de objetos mistos

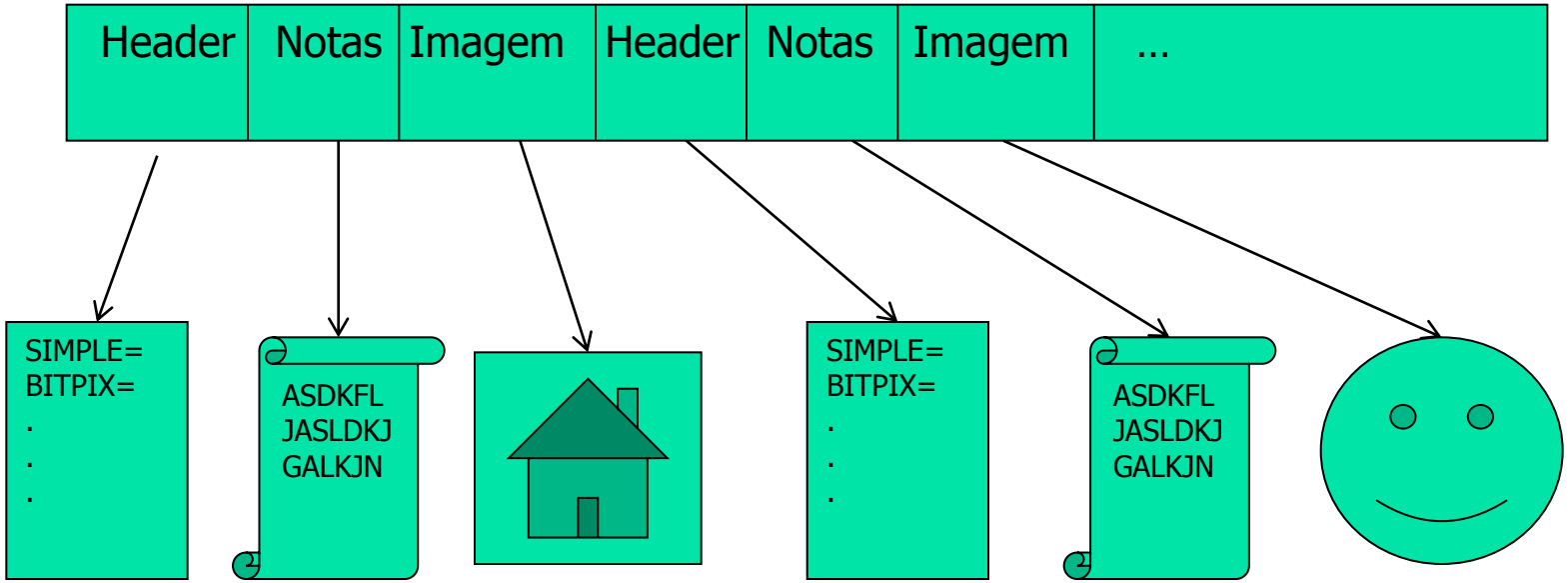


Uso de Tags

- Cada registro de tamanho variável passa a ser indexado por uma *tag*
- No caso: header, notas, image
- Nos registros das tags, guardar o nome da keyword, o deslocamento (byte offset) no arquivo daquela informação, e a indicação do seu tamanho em bytes

Um arquivo "tagged"

Tabela de Índices com Tags

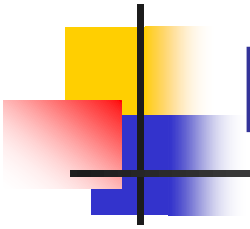


Arquivo com dados primários

➔ Métodos de leitura/escrita para cada tipo de tag

Exemplos de Formatos com estrutura Tag

- TIFF – Tagged Image File Format
- HDF – Hierarchical Data Format
- SGML – Standard General Markup Language
  - Linguagem para descrever estruturas de documentos e para definir tags usadas nas estruturas (HTML, XML, etc.)



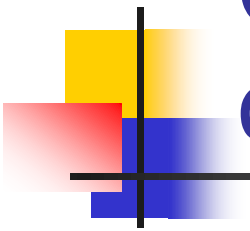
Portabilidade e Padronização

- Formas de codificação de arquivos devem ser "estar de acordo" com a visão de outras pessoas, softwares e computadores

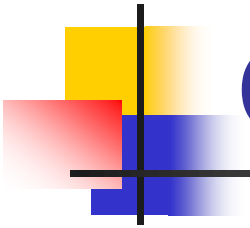
- Fatores que afetam portabilidade
  - Diferenças entre sistemas operacionais
  - Diferenças entre linguagens de programação
  - Diferenças entre arquiteturas de computadores
  - Etc.

- Muitas vezes são necessários conversores de formatos

Organização de arquivos para desempenho



- Organização de arquivos visando desempenho
  - Complexidade de espaço
    - Compressão (tornar menor) e compactação (eliminar espaços vazios) de dados
    - Reuso de espaço
  - Complexidade de tempo
    - Ordenação e busca de dados



Compressão de dados

- A *compressão de dados* envolve a codificação da informação de modo que o arquivo ocupe menos espaço
  - Transmissão mais rápida
  - Processamento sequencial mais rápido
  - Menos espaço para armazenamento

- Algumas técnicas são gerais, e outras específicas para certos tipos de dados, como voz, imagem ou texto
  - Técnicas reversíveis vs. irreversíveis
  - A variedade de técnicas é enorme