UNIVERSIDADE DE SÃO PAULO – CAMPUS DE SÃO CARLOS – ICMC

ALLAN SILVA DOMINGUES 9293290

EDUARDO GARCIA MISIUK 9293230

RAUL WAGNER MARTINS COSTA 9293032

TRABALHO 01 – BASE DE DADOS DE SÉRIES

Trabalho apresentado à professora Cristina D. A. Ciferri da disciplina Organização de Arquivos (SCC0215) como requisito parcial para obtenção de média semestral.

SÃO CARLOS

02 DE MAIO, 2016

ÍNDICE

1. GERAÇÃO DOS DADOS

Como os dados das séries deveriam ser o mais realista possível, optamos por criar um *script* em Python para obtê-los diretamente do IMDb (*Internet Movie Database*). Esses dados são gravados, com a execução do código, em um arquivo de texto para que posteriormente sejam lidos pelo programa em C, que gerará os IDs dos arquivos e os armazenará em um arquivo de dados (com os dados dispostos conforme sua organização na memória e não necessariamente visíveis textualmente).

Para a realização dessa tarefa, foi utilizada uma biblioteca (*Beautiful Soup 4*) para criar uma estrutura pela qual se pudesse navegar e obter os links para as páginas das séries a partir de uma lista do IMDb e então obter os dados de cada uma delas.

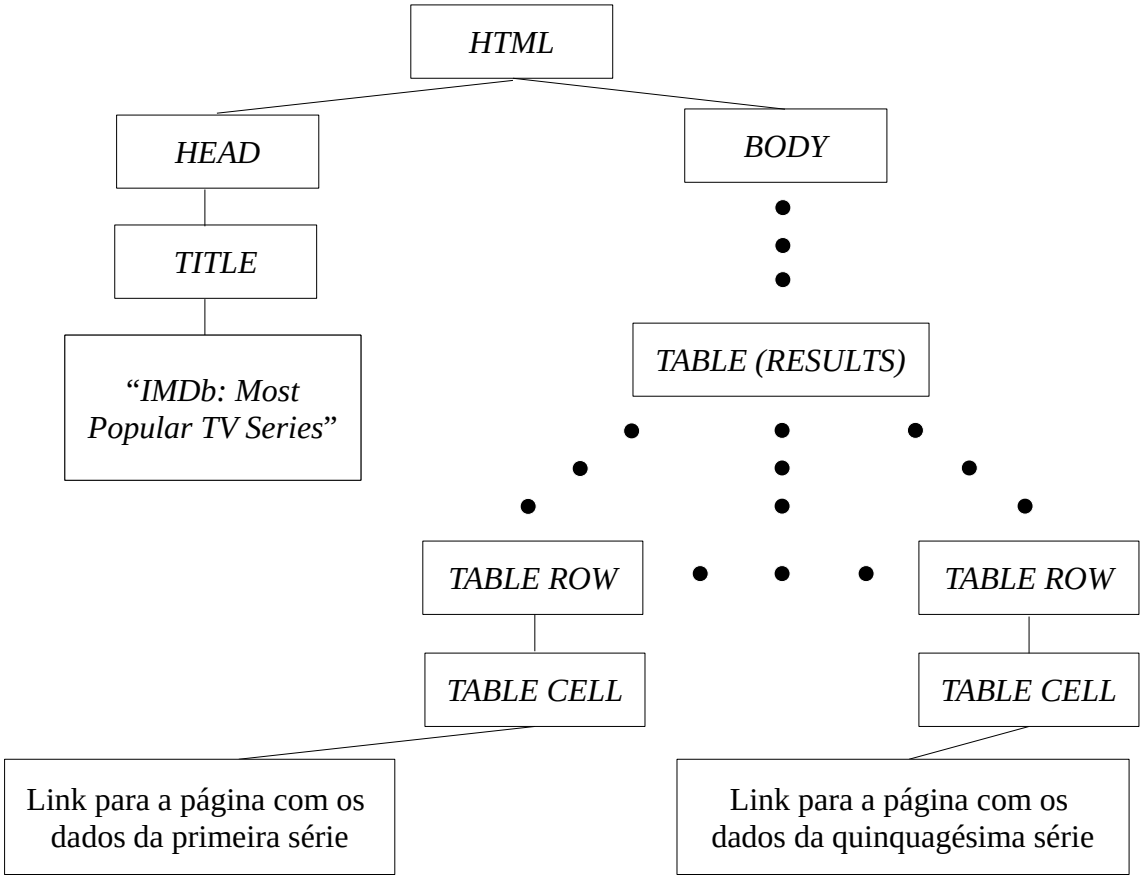

Diagrama 1 – Exemplo de *Parsing Tree* na qual links estão em uma tabela.

2. DEFINIÇÃO DOS CAMPOS DE TAMANHO FIXO

Era necessário que fosse decidido quais campos teriam seu tamanho fixo e como organizá-los no arquivo. Considerando que séries possuem diversos dados cujo tamanho do campo pode variar muito e que o objetivo do banco de dados era tão somente guardar esses decidiu-se que a menor quantidade possível de campo seria truncada em algum momento. Assim, dentre o universo de campos, os campos referentes ao ID da série, ao ano em que seu primeiro episódio foi lançado, ao principal país de produção e à quantidade de temporadas existentes seriam de tamanho fixo.

Os tamanhos definidos para cada um dos campos fixos estão diretamente relacionados com os tipos de variáveis definidos na linguagem C. O campo ID deve possuir o tamanho de um ***int*** (varia de acordo com a plataforma, mas, geralmente, é de 4 bytes), o do ano de lançamento, o tamanho de um ***short int*** (geralmente, 2 bytes), o país de produção equivalente ao espaço ocupado por 60 caracteres (incluindo um caractere terminador de ***String***, geralmente, 60 bytes) e, finalmente, o campo que guarda a quantidade de temporadas é do tamanho de um ***char*** (geralmente, 1 byte).

Assim, de acordo com padrões recentes como o ISO/IEC9899, há a garantia de que ao menos 65534 IDs únicos por série poderiam ser guardados, séries de diversos anos diferentes poderiam ser armazenadas (todos os anos do intervalo [-32767, 32767]), haveria suporte para países de cujo nome não tenha mais de 59 caracteres (comum quando não considerados os nomes oficiais, mas aqueles que geralmente são usados) e as séries poderiam ter até 127 temporadas (desconsiderando que há 127 números negativos dentre os 255 suportados).

Tais tamanhos foram escolhidos para que coubesse uma quantidade razoável de dados, mas com um tamanho sustentável da base de dados, eliminando situações que, atualmente, são absurdas, como suportar uma série de TV que possua centenas de temporadas.

3. DEFINIÇÃO DOS CAMPOS DE TAMANHO VARIÁVEL

Decididos os campos de tamanho fixo, os restantes seriam de tamanho variável, são eles: título da série, descrição da série e gênero da série.

Os primeiros foram escolhidos dessa forma porque podem variar muito em comprimento e não seria viável determinar que se usasse o maior valor por dois motivos: uma nova série poderia surgir e ter um desses campos com comprimento maior do que o antigo maior comprimento ou muito espaço seria desperdiçado, considerando que o maior comprimento só ocorreria uma vez, enquanto outros muito menores ocorreriam diversas outras. Quanto ao terceiro campo determinado como variável, seria possível utilizar o tamanho máximo de um gênero para guardá-lo ou simplesmente guardar um código para cada gênero de série em um ***char***, mas seria inadequado fazê-lo por motivos de desperdício de espaço e perda de versatilidade do código, respectivamente.

Em relação a adoção de tamanhos fixos mas de comprimento médio ou curto nesses campos, trata-se de outra media não viável, pois esses campos são aqueles que fornecem mais informações específicas em relação às séries. Com muito truncamento nos mesmos, algumas descrições não seriam compreensíveis e algumas séries poderiam não ser facilmente identificadas.

4. SISTEMA OPERACIONAL E PLATAFORMA UTILIZADOS

Para o desenvolvimento e testes do programa foram utilizadas diversas máquinas. A plataforma mais antiga utilizada foi um computador com sistema operacional Ubuntu 12.04.5 LTS e GCC 4.6.3. No entanto, fizemos testes em plataformas mais atualizadas com GCC 5.3.0 e o programa se comportou da mesma forma. Não foram utilizadas bibliotecas vinculadas ao Linux, Windows ou Mac no programa principal, nem qualquer biblioteca que não seja padrão.

Apesar disso, o *script* em Python 2 para retirar as informações diretamente do portal IMDb foi feito em um sistema Ubuntu 14.04 e requere uma biblioteca que não é padrão (*Beautiful Soup 4*). O Apêndice A explica como instalar essa biblioteca.

5. DESCRIÇÃO DOS CAMPOS

Abaixo, encontra-se quais são os nomes de cada campo dos registros das séries e uma curta descrição de sua utilidade e formatação:

1. **idSerie:** campo de tamanho fixo com um código identificador da série, esse campo é uma chave primária e é de tipo ***int***;
2. **tituloSerie:** campo de tamanho variável que armazena o título da série, esse campo é um ponteiro para **chars** de tamanho variável, terminado com um '|' (*pipe*) no arquivo de dados;
3. **descSerie:** campo de tamanho variável que armazena uma sinopse da série, esse campo é um ponteiro para caracteres durante seu uso no programa e pode ser lido textualmente no arquivo de dados, seu terminador também é um '|';
4. **producao:** nome do país em que, majoritariamente, a série foi produzida, esse campo é de tamanho variável e, durante o programa, também é tratado como um ponteiro para um conjunto de caracteres, seu delimitador também é um *pipe*;
5. **anoLancamento:** campo de tamanho variável no qual o ano de lançamento de uma determinada série é armazenado, esse campo é tratado como um ***short int*** durante a execução do programa;
6. **temporada:** outro campo de tamanho fixo que armazena a quantidade de temporadas já existentes para uma determinada série, esse campo possui tamanho fixo e é tratado como um ***char*** durante a execução do programa;
7. **generoSerie:** outro campo de tamanho variável terminado por um *pipe* no arquivo de dados e tratado como ponteiro para caracteres no programa, armazena o valor do principal gênero no qual a série se encaixa (considerado apenas o primeiro listado no IMDb).

6. ORGANIZAÇÃO DOS REGISTROS NO ARQUIVO

Como já fora comentado, os delimitadores dos campos de tamanho variável são *pipes* (caractere de número 179 na ASCII), um para cada campo, no final. Para terminar registros o caractere escolhido foi o de número 186 na ASCII, ele não pode ser impresso mas é lido normalmente por programas.

Além de definir quais arquivos seriam de tamanho fixo e quais teriam comprimento variável, optou-se por manter os de tamanho fixo no início do registro para que os mesmos pudessem ser facilmente passados quando desejado. Assim, os campos dos registros estão dispostos da seguinte maneira, considerando, para os *byte offsets* os tamanhos utilizados no capítulo 2.

Campo	idSerie	producao	anoLancamento	temporada	tituloSerie	descSerie	generoSerie
Byte offset	0	4	64	66	67	67 + x	67 + y

Tabela 1 – *Offsets* dos campos em um mesmo registro, com os campos de tamanho fixo posicionados no início e considerando x como o tamanho do título da série e y como o tamanho do título da série adicionado ao tamanho da descrição da série

Exemplo de um registro nesse formato:

567				U	S	A	*b*	*b*	*b*	*b*	*b*	*b*	*b*	*b*	*b*	*b*	*b*	*b*	*b*
b	*b*	*b*	*b*	*b*	*b*	*b*	*b*	*b*	*b*	*b*	*b*	*b*	*b*	*b*	*b*	*b*	*b*	*b*	*b*
b	*b*	*b*	*b*	*b*	*b*	*b*	*b*	*b*	*b*	*b*	*b*	*b*	*b*	*b*	*b*	*b*	*b*	*b*	*b*
b	*b*	*b*	*b*	2011		7	G	A	M	E		O	F		T	H	R	O	N
E	S	\0	|	W	H	I	L	E		A		C	I	V	I	L		W	A
R		B	R	E	W	S		B	E	T	W	E	E	N		S	E	V	E
R	A	L		N	O	B	L	E		F	A	M	I	L	I	E	S		I
N		W	E	S	T	E	R	O	S	,		T	H	E		C	H	I	L
D	R	E	N		O	F		T	H	E		F	O	R	M	E	R		R
L	E	S	⋯	\0	|	A	D	V	E	N	T	U	R	E	\0	|	\186	*b*	*b*

Figura 1 – Exemplo de registro gravado no formato escolhido

7. FUNCIONALIDADES DO PROGRAMA

Durante essa fase de desenvolvimento do programa, estão prontas as funcionalidades de gerar um arquivo de dados com cem séries não ordenadas e que, a cada vez em que o arquivo é gerado, possuem IDs diferentes (de 0 a 999) e estão em diferentes posições do arquivo de dados, de buscar por uma chave primária (ID) em um arquivo de dados previamente existente ou de buscar por todos os dados de um determinado arquivo de dados.

O arquivo de dados gerado baseia-se em dados obtidos diretamente do IMDb e, devido a isso, são bem próximos à realidade. A busca por um ID imprime se o ID foi encontrado ou não e os seus dados, se possível, e a busca por todos os registros imprime todos os registros existentes.

8. INTERFACE

Optou-se por criar uma interface textual, a interface possui opções de criar um arquivo de dados com 100 registros aleatórios baseados em um arquivo de texto previamente criado, procurar por um registro pela sua chave primária ou buscar todos os registros e exibí-los. Essas opções são escolhidas através da inserção de um inteiro no programa. Se um inteiro inválido for digitado, há um aviso de que uma opção é inválida. No entanto, não há garantia de que, se o usuário digitar uma sequência de caracteres, por exemplo, 'batata', toda a *string* será lida de uma vez. Nesse caso, uma mensagem de opção inválida seria impressa para cada caractere.

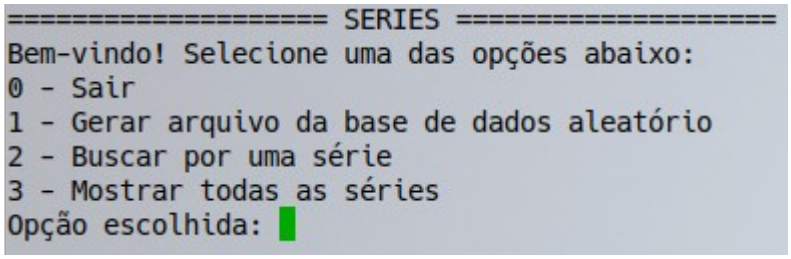

Illustration 1: parte da interface na qual se escolhe a opção desejada

Abaixo, há imagens com exemplos de saída para cada opção. A primeira opção gera um arquivo de dados conforme descrito na seção 6.

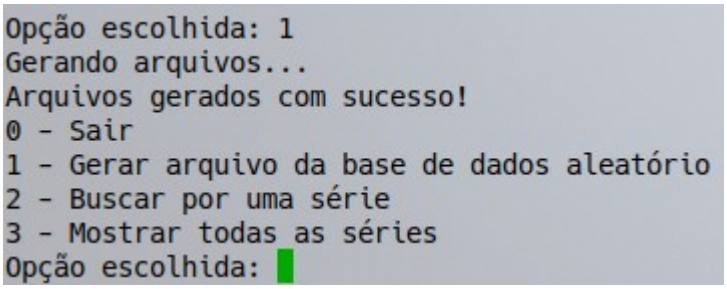

Illustration 2: resposta da interface ao pedido de geração de arquivo de dados

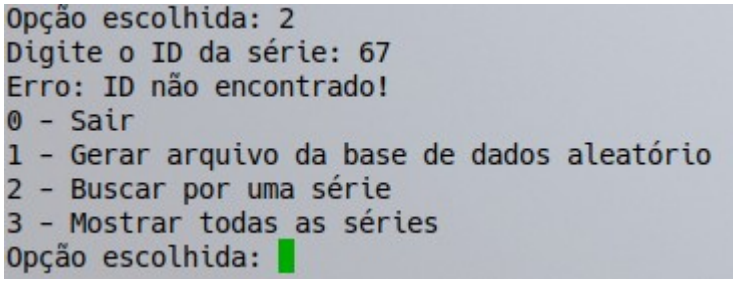

Illustration 3: exemplo de retorno quando à busca de um registro por um ID não utilizado

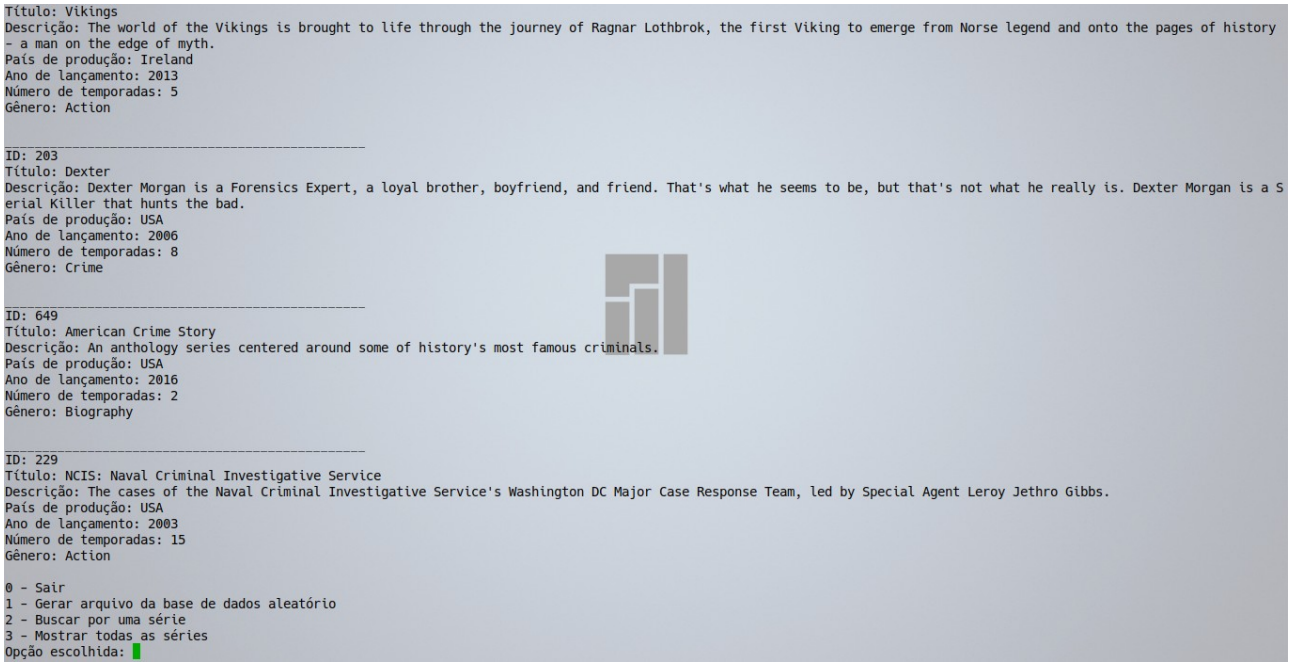

Illustration 4: parte do retorno quando a opção selecionada é a busca por todos os registros, há uma divisão entre os registros e uma tag para cada campo

Ao finalizar o programa, há uma mensagem imediatamente anterior à finalização do mesmo que informa ao usuário que o programa está sendo finalizado. Entre a mensagem e a finalização do programa, há a liberação de toda a memória utilizada e fechamento dos arquivos caso isso ainda não tenha sido feito.

9. REALIZAÇÃO DE TESTES

Para a realização de testes, existem dois métodos: manual e automático. Caso haja algum erro durante a execução, será criado um arquivo *err.log* na pasta raiz contendo a saída de erros. O trabalho rodará no programa *Valgrind* por padrão.

A realização manual de testes é feita rodando somente *make* na linha de comando na pasta raiz.

Para realizar o teste automático, é necessário criar um arquivo *.in* e adicionar ao *Makefile* seu caminho. Após esta mudança, realiza-se o comando *make runtest*. Ele irá excluir os arquivos inúteis, compilar o código e então rodar.

Além disso, é importante notar que, como os IDs são gerados aleatoriamente e os dados estão dispostos aleatoriamente (sendo esses dois processos não vinculados), não há como saber qual ID pesquisar em uma busca por um registro. Logo, é necessário tentar diversas chaves aleatórias até que se descubra uma que foi utilizada. Enquanto isso, o programa irá informar que o ID é inválido.

APÊNDICE A – USANDO O SCRIPT PARA OBTER OS DADOS DO IMDb

Para utilizar o *script* para obter os dados das séries diretamente do IMDb. É necessária a instalação de uma biblioteca chamada *Beautiful Soup 4*. Para utilizar essa biblioteca em um sistema Linux, por exemplo, basta instalar a plataforma Python 2 diretamente do repositório oficial da distribuição e o PIP (acrônimo recursivo, *PIP Installs Packages*) para instalar a biblioteca. Se em um mesmo sistema com Python 2 e 3 instalado, é necessário realizar a conversão do *script* para Python 3 e instalar a biblioteca com PIP ou utilizar o PIP específico para o Python 2.

Em uma das versões da distribuição Ubuntu que possuiria apenas Python 2, por exemplo, executar-se-iam os comandos abaixo:

```
sudo apt-get install python2 pip
sudo pip install beautifulsoup4
```