Mecanismos de Recuperação

Banco de Dados: Teoria e Prática

André Santanchè/Luiz Celso Gomes Jr
Instituto de Computação – UNICAMP
Setembro 2013

Exercício extra*

Descreva, brevemente, seu cenário de apocalipse preferido (zombie, alienígenas, máquinas, meteoro, etc). Considere que você é o desenvolvedor do banco de dados de uma organização que trabalha com dados críticos para auxiliar a mitigação do cenário. O banco de dados trabalha com um alto índice de transações concorrentes. Descreva os mecanismos que você usaria para garantir a atomicidade e durabilidade das transações e como eles seriam importantes para garantir o futuro da humanidade. Dê exemplos de transações neste cenário e dos possíveis erros e falhas que seriam causados no banco de dados.

* Questão opcional, para a próxima aula (26/09), valendo pontos extras, equivalente a uma lista de exercícios. Pontuação baseada na criatividade e, sobretudo, rigor técnico. Máximo meia página de texto.

Mecanismos de Recuperação

Propósito

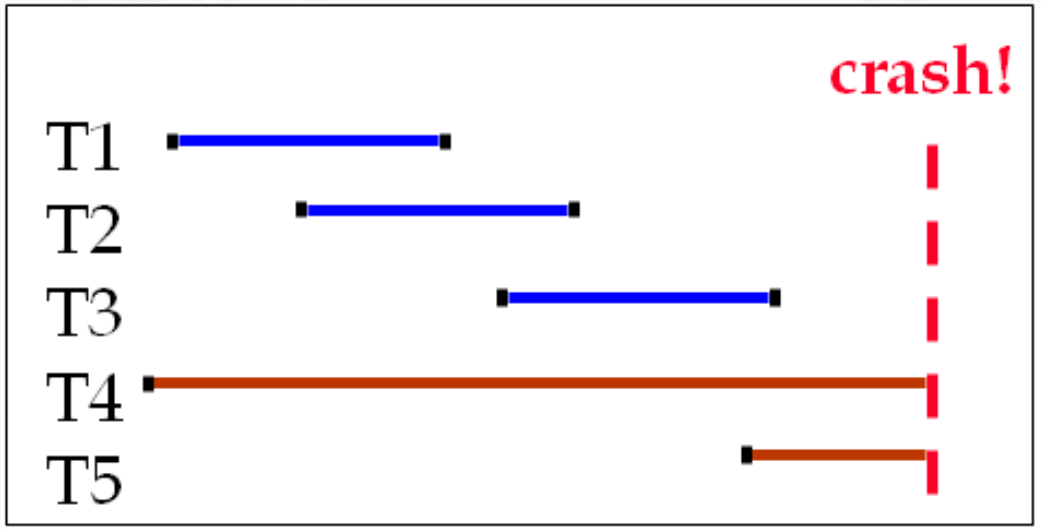
- Restaurar o BD ao seu último estado consistente antes de uma falha
- Preservar as propriedades ACID: **A**tomicidade e **D**urabilidade

Propriedades ACID

- **Atomicidade**: todas as operações da transação acontecem ou nenhuma acontece
- Preservação de **Consistência:** a execução completa de uma transação faz o BD passar de um estado consistente para outro
- **Isolamento**: uma transação deve ser executada como se estivesse isolada das demais
- **Durabilidade** ou permanência: se uma transação é efetivada, seu efeito persiste

Gerenciamento de Recuperação

- O gerenciamento de recuperação garante a Atomicidade e Durabilidade
  - **Atomicidade**: transações podem reverter (*rollback*)
  - **Durabilidade**: o que fazer se o SGBD parar?
- Exemplos:
  - T1, T2 & T3 tem que permanecer
  - T4 & T5 devem ser revertidas



(Ramakrishnan, 2003b)

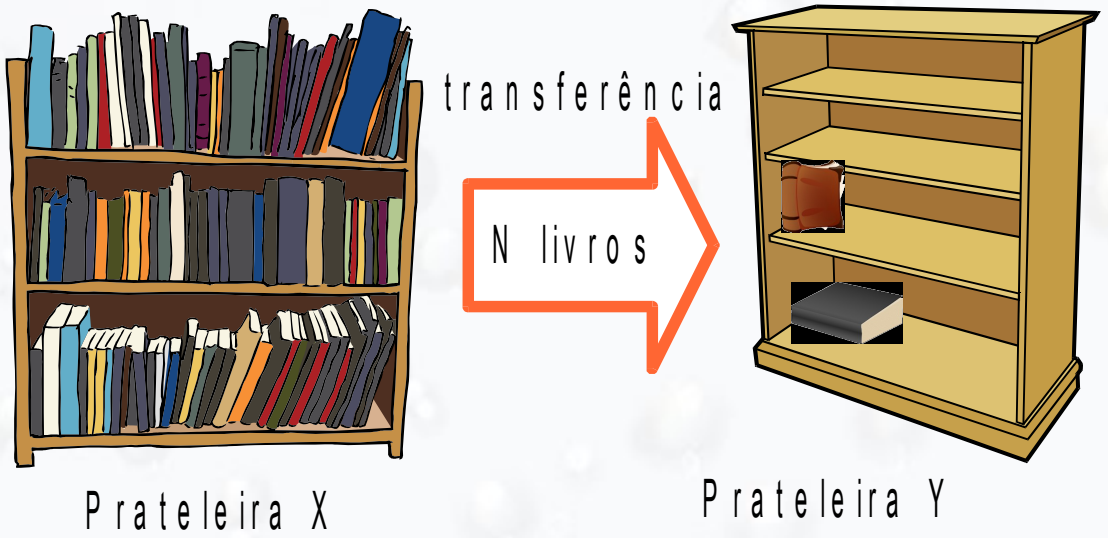
Falha

- Tipos de Falha:
  - Sem dano físico ao BD:
    - O computador falhar (crash ou queda de sistema)
    - Um erro de transação ou sistema
    - Erros locais ou condições de exceção detectadas pela transação
    - Imposição do controle de concorrência. Exemplos?
  - Com dano físico ao BD:
    - Falha de disco
    - Problemas físicos e catástrofes

Exemplo de cenário de recuperação
Transação 1: Transferência

T1

```
ler(X)
X = X - N
gravar(X)
ler(Y)
Y = Y + N
gravar(Y)
```



Exemplo de cenário de recuperação

Transação 2: Aquisição

T2

```
ler(X)
X = X + M
gravar(X)
```

Plano de Execução

T1	T2
início	
ler(X)	
X = X - N	
	início
gravar(X)	
	ler(X)
	X = X + M
ler(Y)	
	gravar(X)
Y = Y + N	
gravar(Y)	
commit	
	commit

Plano de Execução

S
início
ler(X)
X = X - N
início
gravar(X)
ler(X)
X = X + M
ler(Y)
gravar(X)
Y = Y + N
gravar(Y)
commit
commit

Exercício 1

Que ações o banco de dados deve tomar para garantir a consistência do banco de dados no caso de falhas independentes nos pontos marcados no plano abaixo? Quais informações são necessárias para desempenhar as ações?

S
início T1
ler(X)
X = X − N
início T2
gravar(X)
ler(X)
X = X + M
ler(Y)
gravar(X)
Y = Y + N
gravar(Y)
commit
commit

Falha (a)

Falha (b)

Log

- Mantém o registro sequencial das operações de transação que afetam itens do BD
- Estes dados podem ser necessários para:
  - desfazer ações de uma transação “abortada”
  - recuperar o sistema de falhas
  - Auditoria
- O Log é mantido em disco
  - afetado apenas pelas falhas em disco ou catastróficas
  - recomenda-se um disco separado

Protocolo Write-Ahead Logging (WAL)

- Grava registro de operação no disco de log antes que a modificação do item seja gravada em disco
  - garante atomicidade
- Todas as operações de uma transação são gravadas no disco de log antes do commit
  - garante durabilidade

(Ramakrishnan, 2003b)

Tipos de Registro do Log

- [**start_transaction**,T]
- [**write_item**,T,X,valor_antigo,novo_valor]
- [**read_item**,T,X]
- [**commit**,T]
- [**abort**,T]
- [**checkpoint**]

Log

- Cada transação T tem um identificador único gerado automaticamente pelo sistema
- Campos para recuperação (UNDO e REDO):
  - BFIM (Before Image): estado antes da alteração
    - usado para UNDO
  - AFIM (After Image): estado depois da alteração
    - usado para REDO

Exemplo de Log

X = 50 **Y = 110**
N = 20 **M = 40**

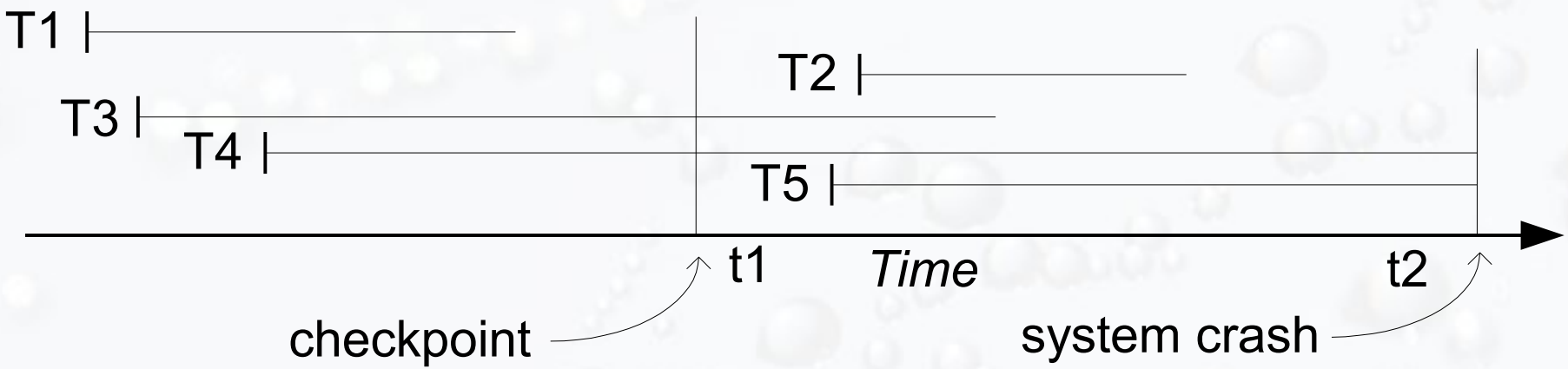
Plano	
início	
ler(X)	ler(50)
X = X – N	X = 50 – 20
início	
gravar(X)	gravar(30)
ler(X)	ler(30)
X = X + M	X = 30 + 40
ler(Y)	ler(110)
gravar(X)	gravar(70)
Y = Y + N	Y = 110 + 20
gravar(Y)	gravar(130)
commit	
commit	

LOG							
	Trans.	^Trás	^Frente	Op.	Item	BFIM	AFIM
1	T1	0	2	start			
2	T1	1	4	read	X		
3	T2	0	5	start			
4	T1	2	6	write	X	50	30
5	T2	3	7	read	X		
6	T1	4	9	read	Y		
7	T2	5	10	write	X	30	70
8	T1	6	9	write	Y	110	130
9	T1	9	-	commit			
10	T2	7	-	commit			

Checkpoint

- Entrada no LOG gravada periodicamente
- Indica gravação de todos os dados modificados do buffer em disco

Recuperação



(Elmasri, 2004)

Checkpoint Fuzzy

- Usa [begin_checkpoint] no início do processo e libera para outros processos
- Usa [end_checkpoint] no final
  - Não válido enquanto não alcança este ponto

Recuperação

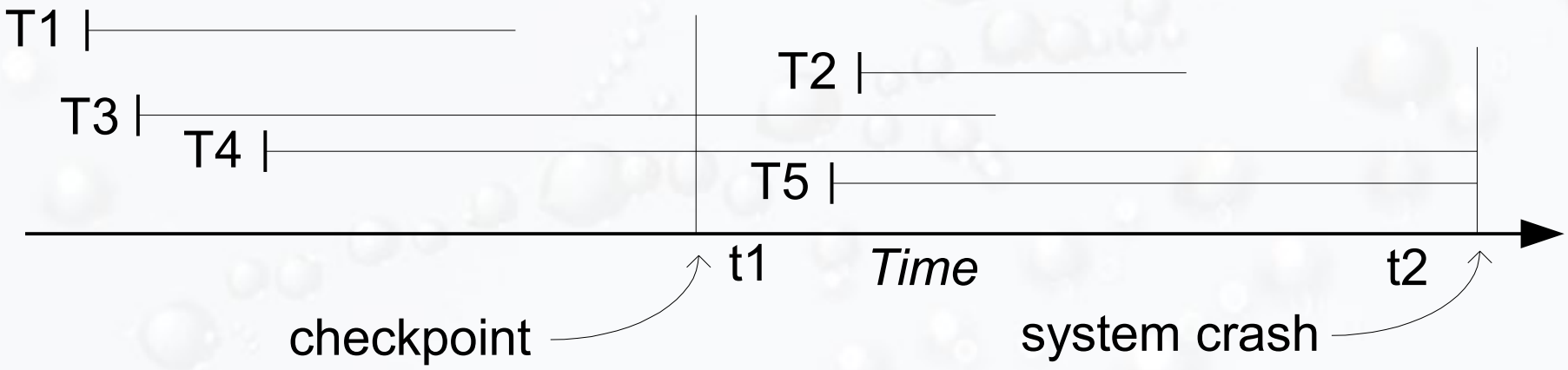
- Algoritmo básico:
  - Varre o log de trás para frente
    - cria listas de transações commit e rollback
    - desfaz alterações de transações ativas
  - Varre o log de frente para trás
    - refaz alterações de transações confirmadas
    - ignora transações abortadas
  - Restaura transações ativas?
- Operações devem ser idempotentes (execuções múltiplas têm o mesmo efeito de uma execução única)

Exercício 2

Considerando o plano abaixo, responda:

a) Quantas transações devem ser refeitas e desfeitas se **não há** um mecanismo de checkpoint?

b) Quantas transações devem ser refeitas e desfeitas se **existe** um mecanismo de checkpoint?



Cache

- Cache do SGBD
  - Baseado em páginas de disco mantidas pelo SO
  - SGBD chama rotinas de baixo nível do SO
  - Essencial para desempenho, porém adiciona complexidade aos mecanismos de log

Dados por Página do Cache

- Bit sujo
  - 0 → página não alterada
  - 1 → página alterada
- Bit preso-solto
  - 0 → página pode ser gravada
  - 1 → página ainda não pode ser gravada (e.g., espera de commit)
- Transações que modificaram a página

Cache
Atualização Shadow x In-place

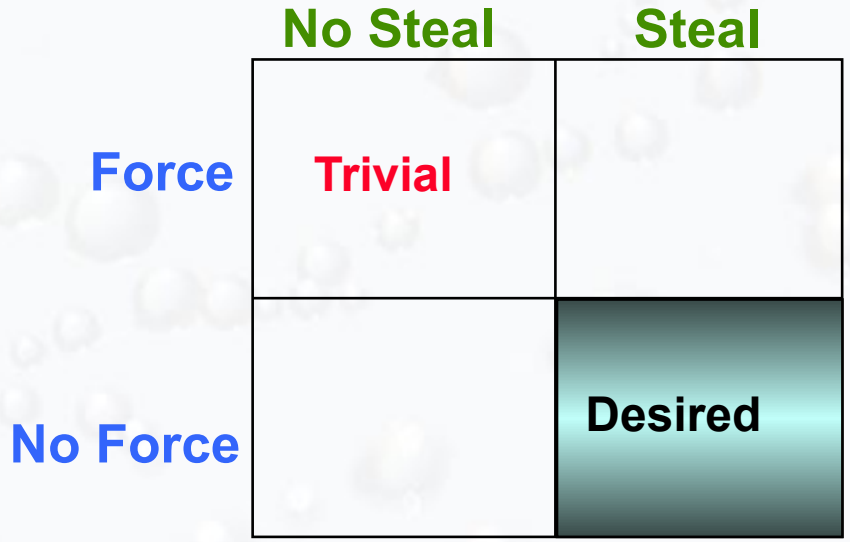
- Shadow: versão modificada de um item gravada em nova localização de disco
- In-place: versão modificada de item sobrescreve a anterior (recuperação por Log)

Gravando Cache no Disco

- Forçar gravação dos itens alterados no disco quando alterados na memória (antes do commit)?
  - Sim: Force
  - Não: No-force
- Permitir alguma gravação antes do commit (para substituição de páginas)?
  - Sim: Steal
  - Não: No-steal

Gerenciando o cache

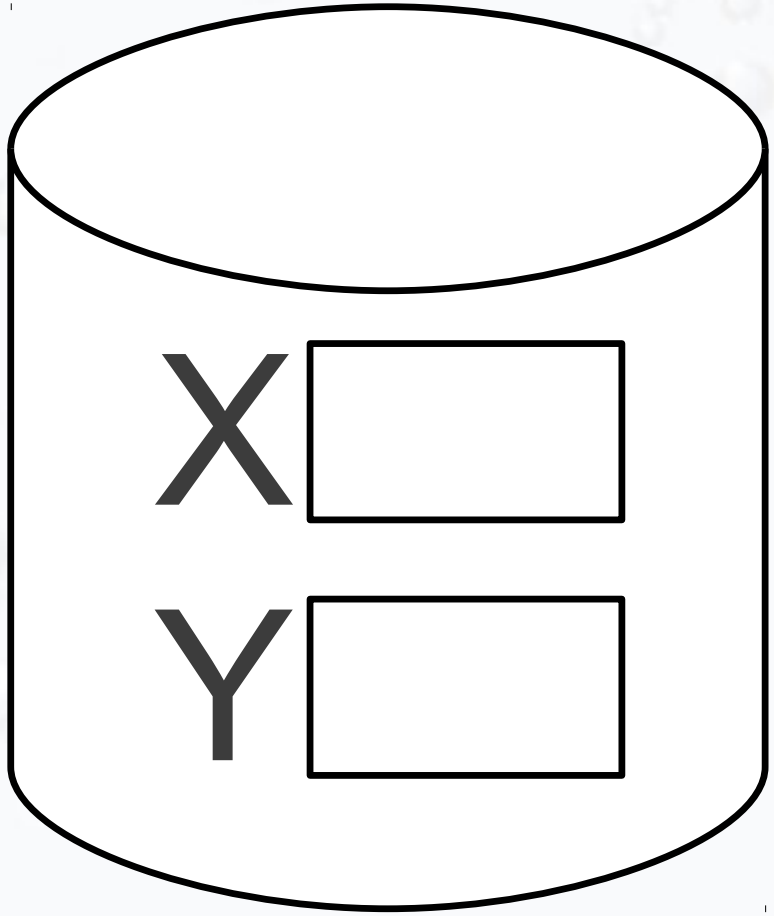
- Force
  - Alto tempo de resposta.
  - Provê durabilidade.
- Steal
  - Melhor throughput.
  - Como garantir atomicidade?



Force

S	LOG
início	□
ler(X)	□
X = X − N	□
início	□
gravar(X)	□
ler(X)	□
X = X + M	□
ler(Y)	□
gravar(X)	□
Y = Y + N	□
gravar(Y)	□
commit	□
commit	□
	□

X=50 N=20
Y=110 M=40



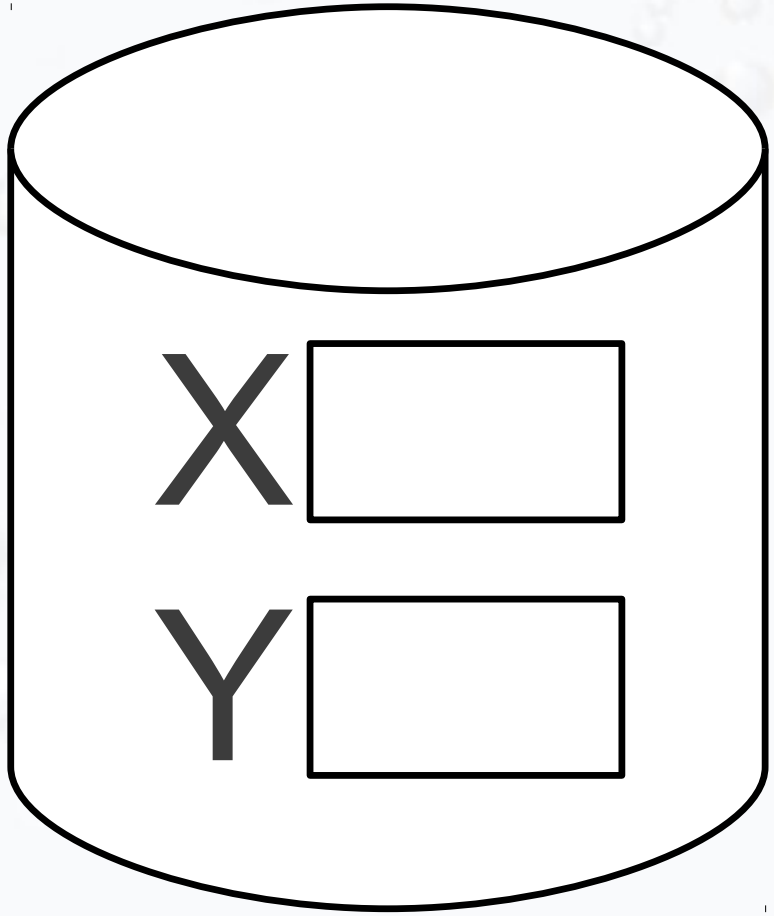
Force → No-REDO

Transações confirmadas já foram gravas no disco.

No-Steal

S	LOG
início	□
ler(X)	□
X = X − N	□
início	□
gravar(X)	□
ler(X)	□
X = X + M	□
ler(Y)	□
gravar(X)	□
Y = Y + N	□
gravar(Y)	□
commit	□
commit	□
	□

X=50 N=20
Y=110 M=40



No-Steal → No-UNDO

Transações permanecem na memória até o commit.

Recuperação e Steal/No-Steal x Force/No-Force

- Steal/No-Force (Undo/Redo)
- Steal/Force (Undo/No-redo)
- No-Steal/No-Force (Redo/No-undo)
- No-Steal/Force (No-undo/No-redo)

Exercício 3

Em caso de falha catastrófica do disco do banco de dados, pode ser necessário recuperar o banco a partir do log, o que implica em um processo lento de repetição da execução de todas as transações realizadas no banco desde sua criação. Como você adicionaria funcionalidades no sistema de log para aproveitar backups realizados a partir de dumps periódicos do banco?

Exercício 4

Associe letras A, C, I, D (respectivamente Atomicity, Consistency, Isolation, Durability) aos mecanismos essenciais para a garantia das respectivas propriedades.

a) transações
b) controle de concorrência
c) visões
d) log
e) restrições de integridade
f) detecção e tratamento de deadlocks
g) backup

André Santanchè
http://www.ic.unicamp.br/~santanche

Referências

- Elmasri, Ramez; Navathe, Shamkant B. (2005) **Sistemas de Bancos de Dados**. Addison-Wesley, 4ª edição em português.
- Elmasri, Ramez; Navathe, Shamkant B. (2010) **Sistemas de Banco de Dados**. Pearson, 6ª edição em português.
- Ramakrishnan, Raghu; Gehrke, Johannes (2003) **Database Management Systems.** McGraw-Hill, 3rd edition.
- Ramakrishnan, Raghu; Gehrke, Johannes (2003b) Database Management Systems. McGraw-Hill, 3rd edition (companion slides).

Licença

- Estes slides são concedidos sob uma Licença Creative Commons. Sob as seguintes condições: Atribuição, Uso Não-Comercial e Compartilhamento pela mesma Licença.
- Mais detalhes sobre a referida licença Creative Commons veja no link:
  http://creativecommons.org/licenses/by-nc-sa/3.0/

- Agradecimentos: fotografia da capa e fundo por Ben Collins - http://www.flickr.com/photos/graylight/.
  Ver licença específica em
  http://www.flickr.com/photos/graylight/261480919/

Controle de Cache