**Um estudo de caso Chado: um esquema modular baseado em ontologia para representar informações biológicas associadas ao genoma**

[Christopher J. Mungall](javascript:;), [David B. Emmert](javascript:;), [Consórcio FlyBase](javascript:;)

*Bioinformatics* , Volume 23, Issue 13, July 2007, Pages i337 – i346, <https://doi.org/10.1093/bioinformatics/btm189>

**Publicados:**

01 de julho de 2007

* [PDF](https://academic.oup.com/bioinformatics/article-pdf/23/13/i337/23705777/btm189.pdf)
* [Vista dividida](javascript:;)
* [Citar](https://academic.oup.com/bioinformatics/article/23/13/i337/229507?login=true)
* [Ícone de permissões Permissões](https://s100.copyright.com/AppDispatchServlet?publisherName=OUP&publication=1460-2059&title=A%20Chado%20case%20study%3A%20an%20ontology-based%20modular%20schema%20for%20representing%20genome-associated%20biological%20information&publicationDate=2007-07-01&volumeNum=23&issueNum=13&author=Mungall%2C%20Christopher%20J.%3B%20Emmert%2C%20David%20B.&startPage=i337&endPage=i346&contentId=10.1093%2Fbioinformatics%2Fbtm189&oa=&copyright=Oxford%20University%20Press&orderBeanReset=True)

[Compartilhar](javascript:;)

**Resumo**

**Motivação:** Alguns anos atrás, FlyBase se comprometeu a projetar um novo esquema de banco de dados para armazenar dados de *Drosophila* . Ele integraria totalmente a sequência genômica e os dados de anotação com dados bibliográficos, genéticos, fenotípicos e moleculares da literatura, representando uma destilação dos primeiros 100 anos de pesquisa neste importante sistema de modelo animal. Ao desenvolver este novo esquema integrado, FlyBase também se comprometeu a garantir que seu design fosse *genérico, extensível* e *disponível*como código-fonte aberto, para que possa ser empregado como o esquema central de qualquer repositório de dados de organismo modelo, evitando assim o desenvolvimento de software redundante e aumentando potencialmente a interoperabilidade. Nossa pergunta era se poderíamos criar um esquema de banco de dados relacional que seria reutilizado com sucesso.

**Resultados:** Chado é um esquema de banco de dados relacional que agora está sendo usado para gerenciar o conhecimento biológico para uma ampla variedade de organismos, de humanos a patógenos, especialmente as classes de informações que direta ou indiretamente podem ser associadas a sequências de genoma ou ao RNA primário e produtos proteicos codificados por um genoma. Os bancos de dados biológicos que estão em conformidade com esse esquema podem interoperar uns com os outros e com o software aplicativo do kit de ferramentas Generic Model Organism Database (GMOD). Chado é distinto porque seu design é movido por ontologias. O uso de ontologias (ou vocabulários controlados) é onipresente em todo o esquema, pois são usados ​​como um meio de *digitação*entidades. O esquema Chado é particionado em subesquemas integrados (módulos), cada um encapsulando um domínio biológico diferente, e cada um descrito usando representações em ontologias apropriadas. Para ilustrar essa metodologia, descrevemos aqui os módulos Chado usados ​​para descrever sequências genômicas.

**Disponibilidade:** GMOD é uma colaboração de vários grupos de banco de dados de organismo modelo, incluindo FlyBase, para desenvolver um conjunto de software de código aberto para gerenciar dados de organismo modelo. O esquema Chado é distribuído gratuitamente sob os termos da Licença Artística ( <http://www.opensource.org/licenses/artistic-license.php> ) do GMOD ( [www.gmod.org](http://www.gmod.org/) ).

**Contato:**[cjm@fruitfly.org](mailto:cjm@fruitfly.org) ou [emmert@morgan.harvard.edu](mailto:emmert@morgan.harvard.edu) .

**1. INTRODUÇÃO**

**1.1 Sobre a necessidade de esquemas de banco de dados padronizados**

Bancos de dados de genomas específicos de organismos são repositórios habilmente curados de dados e conhecimento relativos a uma espécie biológica particular ou uma coleção de espécies semelhantes intimamente relacionadas. Esses bancos de dados biológicos são tipicamente (mas nem sempre) implementados como bancos de dados relacionais que codificam seu modelo de domínio usando o modelo relacional. Um banco de dados relacional requer um sistema de gerenciamento de banco de dados (DBMS) para acessar e atualizar os dados. Os dados alojados em um banco de dados devem ser modelados de acordo com um *esquema de banco de dados* , uma descrição computável do domínio de dados, expressa principalmente como definições de tabela. Modeladores de dados, em conjunto com especialistas no domínio, projetam esquemas de banco de dados. Os usuários interagem com o banco de dados por meio de aplicativos de software e interfaces de usuário (muitas vezes por meio de outra camada de indireção, ou seja, um intermediário,camada de *middleware* ). O design e a implementação de aplicativos de banco de dados consomem muito tempo e trabalho. Além disso, quando os aplicativos de banco de dados são construídos para trabalhar com um determinado esquema ou conjunto de esquemas, as mudanças no esquema do banco de dados podem ditar mudanças recíprocas neste software. Tudo isso torna a *evolução do esquema* um assunto caro. A partir disso, parece resultar que um pequeno número de esquemas estáveis ​​seria favorecido em vez de uma infinidade de esquemas de evolução rápida, e ainda o último é mais comum em bioinformática. Por que isso acontece?

**1.1.1 Novos conhecimentos**

Os esquemas devem evoluir para lidar com as mudanças nos requisitos. Mais críticas são as mudanças na natureza dos dados subjacentes, que se acumulam e evoluem constantemente. A natureza dos dados biológicos se expandiu tremendamente ao longo do tempo, variando de estudos genéticos clássicos realizados um século atrás (Morgan, [1907](javascript:;) ), ao conhecimento molecular atual em escala de genoma. Por exemplo, um esquema de banco de dados construído em torno do dogma central único de "um gene codifica para uma enzima" (Beadle e Tatum, [1941](javascript:;) ) seria consideravelmente mais simples do que um esquema que representa com precisão nossa compreensão atual das complexidades da transferência de informações genéticas. . Como nossa compreensão do mundo natural muda com o tempo, os requisitos devem necessariamente mudar também.

**1.1.2 Novas técnicas experimentais**

Concomitante com nosso acúmulo de conhecimento biológico, estão os avanços nos métodos e materiais que usamos para obter esse conhecimento. Essas rápidas mudanças tecnológicas colocam requisitos adicionais no esquema. Durante o curto período de existência dos bancos de dados genéticos, vimos as técnicas experimentais se expandirem do mapeamento físico e PCR; ao sequenciamento de genomas inteiros; às modernas tecnologias de alto rendimento para análise de microarray e proteômica: todas as quais colocam demandas crescentes no esquema de banco de dados que deve representá-los.

**1.1.3 Organismos diferentes**

Uma ampla variedade de espécies é usada na pesquisa porque cada uma oferece uma vantagem única para explorar certos aspectos da vida. A variância taxonômica das propriedades biológicas, juntamente com os diferentes métodos experimentais utilizados nessas espécies, adicionam outra dimensão aos requisitos. Qualquer organismo é selecionado para um projeto de pesquisa com base em sua utilidade em responder a diferentes perguntas, e isso tornou historicamente difícil criar um esquema de banco de dados cego para espécies.

**1.1.4 Aceitabilidade**

Juntamente com a complexidade inata dos dados; as mudanças de requisitos conforme o progresso da ciência e das tecnologias; e a variabilidade entre os projetos de pesquisa, o projeto de esquemas estáveis ​​e compartilhados que sejam aceitáveis ​​para uma ampla variedade de projetos diferentes é uma tarefa desafiadora. Mesmo dentro do reino dos projetos de banco de dados de organismos modelo, a tendência histórica tem sido que cada projeto projete seu próprio esquema *de novo* ou, em alguns casos, comece com um esquema existente e personalize-o para satisfazer um conjunto diferente de requisitos. Essas personalizações inevitavelmente levam à divergência, perda de interoperabilidade e duplicação de esforços.

Como esses fatores dificultam a criação de esquemas estáveis, os esquemas estão em constante evolução com altos custos concomitantes de manutenção de software. O desafio do projeto de banco de dados biológico é como acompanhar um alvo em movimento.

**1.2 Abordagens existentes para esquemas biológicos**

Como era de se esperar, há uma grande variedade de abordagens no projeto de esquemas para bancos de dados biológicos. Alguns dos esquemas mais notáveis, com os quais temos experiência pessoal direta, incluem: ACeDB, ArkDB, Ensembl, Genomics Unified Schema (GUS) e Gadfly (Mungall *et al* ., [2002](javascript:;) ). ACeDB (A *C.elegans* Database) foi um dos primeiros bancos de dados de organismos modelo. Foi construído para *Caenorhabditis elegans* (Durbin e Theyry-Mieg, [1994](javascript:;) ) e é, na verdade, um DBMS que segue um modelo hierárquico em vez de relacional. O ACeDB foi adaptado para uso em uma série de projetos de organismos modelo (bem como em projetos não relacionados à biologia, testemunho de sua flexibilidade). O esquema ArkDB (Hu *et al*., [2001](javascript:;)) foi criado para atender às necessidades do subconjunto da comunidade de organismos modelo interessada em animais importantes para a agricultura. Tem sido usado com sucesso em diferentes espécies por diferentes comunidades, mas raramente é usado fora da comunidade agrícola. O sistema de banco de dados Ensembl e o esquema foram inicialmente construídos para analisar o genoma humano recém-sequenciado (Hubbard, 2002) e servir os resultados para a comunidade científica. Desde então, foi adotado por outros grupos e é usado para uma grande variedade de espécies (principalmente cordadas). Seu foco também se expandiu e agora o Ensembl inclui uma variedade de bancos de dados federados acessíveis por meio do EnsMart. Como os outros bancos de dados, o GUS foi construído especificamente para análise de transcrição e para atender às necessidades da comunidade de pesquisa do Plasmodium, e foi estendido para servir a comunidades adicionais.Anotações genômicas de *Drosophila* , mas também era usado para armazenar anotações de abelhas e *Anopheles* .

**1.3 Ontologias e terminologias**

Chado difere dos esquemas mencionados acima pela centralidade das ontologias e terminologias como um componente central. Chado usa ontologias não apenas para anotação de entidades biológicas, mas também como uma ampla esquema *entidade de digitação* e *entidade-relacionamento digitação* mecanismo. Esta metodologia de *design baseado em ontologia*é explorado neste artigo. Afirmamos que é a chave para o sucesso e flexibilidade do Chado e de sua adoção por uma ampla variedade de projetos de pesquisa. Em outros esquemas com os quais temos experiência, a digitação dos dados é aplicada na camada relacional. No Chado, ao contrário, a digitação de dados é conduzida por ontologias no módulo de vocabulários controlados, e isso torna possível que o mesmo esquema e aplicativo sejam reutilizados e evoluam ao longo do tempo.

**2 SISTEMA E MÉTODOS**

O pacote Chado usa postGreSQL e Perl. Além do Chado DDL (Data Definition Language), a instalação requer três pacotes Perl adicionais: bioperl-live, go-perl e DBIx :: DBStag. Para instalar no Fedora Core 1-5, OS X ou CentOS 4, você pode usar os pacotes RPM para instalar o Chado, e seus pré-requisitos, fornecidos por Allen Day ( [http://biopackages.net](http://biopackages.net/) ). Caso contrário, a instalação requer o check-out do pacote Chado via CVS anônimo e a execução de uma série de operações de linha de comando. Instanciações de Chado no idioma Oracle ou mySQL também estão disponíveis.

**3 ABORDAGEM DE DESIGN**

Como o Chado faz uso extensivo de ontologias (também conhecidas como vocabulários controlados [1](javascript:;) ) como um meio de *digitar* entidades no esquema e como *metadados* para propriedades de dados extensíveis, uma apreciação dos fundamentos das ontologias e como elas são codificadas no esquema Chado é requerido. A justificativa para essa abordagem é dupla. Ele aborda a questão significativa de requisitos em constante evolução e fornece suporte para o *raciocínio*. Uma ontologia é uma representação dos diferentes tipos de entidades que existem no mundo e dos relacionamentos que existem entre essas entidades. Os exemplos seriam o tipo anatômico 'olho' ou o tipo de processo 'biossíntese de cisteína'. Esses tipos mantêm certos relacionamentos uns com os outros; por exemplo, 'olho é\_um órgão sensorial' ou 'parte do ommatidium\_do olho composto'. Os relacionamentos em uma ontologia podem ser representados como um gráfico (freqüentemente, mas nem sempre um gráfico acíclico direcionado, ou DAG). O artigo sobre relações OBO de Smith *et al* . em [2005](javascript:;)fornece um tratamento detalhado dos tipos de relacionamento em ontologias biológicas. De particular interesse para Chado é a relação, que especifica uma relação de subtipagem entre dois termos ou classes. São as relações que existem entre os tipos na ontologia que fornecem um meio de sustentar o raciocínio.

**3.1 Chado questionando explora relações ontológicas**

Determinar as respostas a consultas simples envolve, em termos gerais, percorrer um DAG definido pela ontologia ao longo das bordas 'is\_a' e 'parte\_of'. Por exemplo, a ontologia de sequência (SO) (Eilbeck *et al* ., [2005](javascript:;) ), uma coleção de tipos de recursos de sequência, é usada para digitar recursos no módulo de sequência do Chado. Tabelas, como o **recurso**tabela, defina uma coluna de chave estrangeira para indicar o tipo específico ou classe de entidade para cada linha nessa tabela. Além dos recursos de digitação, todos os relacionamentos entre os recursos também são digitados de forma consistente, usando os relacionamentos definidos pelo SO. Por exemplo, SO tem um relacionamento part\_of entre o tipo 'exon' e o tipo 'transcrição'. De acordo com a definição de parte\_de na ontologia de relações OBO, isso significa que *todos os* exons são necessariamente parte\_de *algum*transcrito em algum momento de sua existência. Para responder à pergunta, 'encontrar todos os exons que contêm CDS e 5' UTR ', pode envolver a seleção de todas as linhas na tabela de características que são do tipo' 5\_prime\_UTR 'ou do tipo' CDS ', que também são' part\_of ' a mesma linha na tabela de características que é do tipo 'exon'. Implementar essas consultas de 'fechamento transitivo' pode ser difícil usando muitas implementações existentes de linguagens de consulta de banco de dados, então o Chado permite a pré-computação dessas inferências. Esses relacionamentos SO são aplicados uniformemente, ou seja, um único tipo de relacionamento entre recursos sempre implica a mesma coisa. Nos casos em que um novo tipo de relacionamento é necessário, um que ainda não está presente no SO (por exemplo, 'duplicado' para relacionamentos entre aberrações cromossômicas e genes),

**3.2 Esquemas genéricos**

A combinação da coluna type\_id e do relacionamento **is\_a** dá ao Chado um sistema de subclassificação de dados, além do possível com a semântica tradicional do banco de dados SQL. Chado usa a mesma tabela para todos os diferentes tipos de recursos e usa o SO como um sistema de digitação. Essa mesma estratégia, uma única tabela digitada de acordo com uma ontologia, é repetida em todo o esquema. Chado equilibra os pontos fortes e fracos dessa abordagem de esquema genérico usando um sistema de conformidade em camadas, uma nova abordagem para tipagem de banco de dados relacional. As diferentes camadas podem ser vistas como semelhantes a uma pilha de protocolo em sistemas de comunicação de rede.

Na parte inferior está a conformidade de nível 0, conformidade de esquema relacional. Como em implementações RDBMS padrão, o esquema relacional Chado define estruturas de tabela, chaves estrangeiras, exclusividade e outras restrições. Um exemplo de restrição aplicada no nível relacional é uma restrição na localização do recurso, que garante que a posição inicial de um recurso seja menor que a posição final ( **fmin** < **fmax**, consulte SQL). Os recursos integrados ao SGBD exigem que todos os dados estejam em conformidade com algum esquema e impõem automaticamente essas restrições relacionais, portanto, é impossível ter uma instância do Chado que não seja compatível com alguma versão do esquema. No entanto, ainda é possível falar de duas versões do esquema sendo incompatíveis uma com relação à outra, ou falar de uma instância de Chado sendo nível-0 compatível com versões anteriores com relação a outra.

O próximo nível é a conformidade de nível 1, com base em ontologias. Isso significa que todos os valores na coluna do identificador de tipo no banco de dados referem-se a alguns subtipos dos tipos de base oficial do Chado. Por exemplo, todos os tipos de recursos de sequência devem ser tipos de recursos de sequência; deve haver um caminho sobre a relação 'is\_a' do tipo de recurso para o tipo SO 'locatable\_sequence\_feature'. Da mesma forma, todos os identificadores de tipo de propriedade de recurso de sequência devem se referir a um tipo que pode rastrear um caminho 'is\_a' de volta ao tipo 'feature\_property' na ontologia de propriedade de recurso OBO.

Como o modelo relacional padrão não lida com subtipos, chamamos essas restrições extrarelacionais. Essas restrições podem ser codificadas em software externo e aplicadas como parte de uma verificação de validação de dados ou no SGBD como regras e automaticamente aplicadas. Se as restrições forem codificadas em software externo, como middleware, todos os aplicativos ou ferramentas capazes de modificar o banco de dados devem fazer a interface com o banco de dados por meio dessa camada de middleware; e isso pode ser excessivamente restritivo no desenvolvimento de software. As restrições codificadas no SGBD como regras podem ser aplicadas por meio de acionadores de banco de dados, e isso é preferível para garantir a consistência dos dados, mas inflige uma penalidade no desempenho da atualização dos dados.

**4 IMPLEMENTAÇÃO**

Chado é implementado como um RDBMS e, como tal, consiste principalmente em uma coleção de definições de tabela, cada uma correspondendo a uma categoria de entidade de alto nível. Acima dessas tabelas, o Chado é particionado logicamente em diferentes módulos, cada um dos quais modelando domínios distintos dos dados. Modularidade (ou encapsulamento) é um princípio fundamental no projeto de qualquer software ou sistema de informação de grande porte. Ele reduz a complexidade e as interdependências, em contraste com uma abordagem monolítica. Embora os módulos Chado não sejam completamente autônomos e existam algumas dependências entre módulos, eles são minimizados. Cada módulo apenas 'expõe' um subconjunto de suas tabelas a outros módulos. Este sistema modular permite que diferentes instalações locais do Chado conectem seus próprios módulos para estender o esquema, ou para substituir os módulos existentes por seus próprios módulos personalizados.

Cinco módulos 'principais' são necessários para todas as instalações do Chado. Além desses módulos principais, apenas o subconjunto dos módulos necessários para um projeto é instalado; com a ressalva de que, devido às interdependências acima mencionadas, às vezes o uso de um módulo implicará no uso de outro. Os cinco módulos principais cobrem: uso geral, como referências cruzadas de banco de dados; publicações e citações; auditoria; vocabulários controlados (ontologias); e recursos de sequência. As características da sequência biológica são fundamentais para qualquer implementação do Chado é um reflexo do fato de que no Chado, os dados são amarrados ao genoma de forma semelhante à forma como são organizados na natureza. Como tal, no Chado, os recursos de sequência são usados ​​de forma semelhante para relacionar dados genéticos, fenotípicos e funcionais, e esses cinco módulos de corte transversal são aplicáveis ​​a todos os domínios biológicos.

Os módulos opcionais do Chado cobrem uma ampla gama de domínios da biologia molecular. Existem módulos no Chado para análise comparativa, estudos de expressão (microarray e *in situ* ), dados de mapeamento, filogenética, genética, bibliotecas de clones, taxonomias de organismos, fenótipos e informações de endereços pessoais.

O módulo central é o módulo de **sequência** , que modela as *características da sequência* biológica . As características da sequência incluem as entidades geneticamente codificadas, como genes, produtos de genes, exons, regiões regulatórias e outras entidades genômicas hereditárias. Os recursos de sequência são centrais para o modelo Chado e são o foco desta discussão; somente quando o modelo de características do Chado é compreendido é que se pode passar para outros tipos de dados, como expressão e dados genéticos.

**4.1 Módulos principais no Chado**

**4.1.1 Módulo de uso geral**

O objetivo principal deste módulo é fornecer às entidades de dados identificadores estáveis, globais e exclusivos, embora também contenha outras tabelas de metadados de esquema. No Chado, todas as entidades de dados identificáveis ​​têm identificadores tripartidos, consistindo em um nome de banco de dados, uma adesão, junto com um sufixo de versão opcional. Na maioria das instâncias do Chado, a versão raramente é usada, então este é efetivamente um identificador bipartido.

Cada identificador é armazenado como uma linha no **dbxref**tabela, junto com uma coluna vinculando-a ao nome do banco de dados, que é armazenado em uma tabela separada. Um nome de banco de dados identifica exclusivamente a autoridade responsável por um determinado espaço de ID. Manter o nome do banco de dados em uma tabela separada garante que o esquema mantenha seu compromisso com a normalização (portanto, não pode haver dois bancos de dados distintos chamados 'GO' em uma única instância do Chado). A adesão (versão plus) deve ser exclusiva dentro de um espaço de ID (reforçada pelo esquema). Portanto, pode haver dois acessos '0008045', mas pode haver apenas um artefato de dados identificado como 'GO: 0008045'. Todos os identificadores estáveis ​​são armazenados, quer se refiram ou não a entidades externas. Chado não tem uma noção explícita de uma entidade de dados ser externa. Nomes de banco de dados podem ser associados a URIs para compatibilidade com outros esquemas de identificadores (por exemplo, LSID, Clark*et al* ., [2004](javascript:;) ). As entradas em outras tabelas referem-se a entradas na tabela **dbxref** por meio de *chaves estrangeiras* .

**4.1.2 Módulo de publicação**

A proveniência dos dados é de importância central em qualquer banco de dados com curadoria. No Chado, a mesa do **pub** , no módulo com o mesmo nome, trata da proveniência. O nome **pub** indica o uso mais comum para esta tabela, publicações. No entanto, o âmbito desta tabela não se limita aos documentos publicados, nos termos do Chado, uma publicação é qualquer fonte de dados atribuível, incluindo comunicações pessoais e análises de bases de dados.

**4.1.3 Módulo de auditoria**

Este módulo é gerado automaticamente a partir do próprio esquema do banco de dados. Para cada tabela no banco de dados, há um conjunto de funções de gatilho que preenchem uma única tabela de auditoria, chamada 'audit\_chado', de forma que quando houver uma atualização ou exclusão na tabela pai, haja uma inserção da chave exclusiva para o registro antigo na tabela de auditoria com um registro de data e hora e um identificador para o usuário que fez o commit.

**4.1.4 Módulo de ontologias e vocabulários controlados**

Ontologia e vocabulários controlados são parte integrante do Chado, permitindo o esquema genérico e a subclasse e, portanto, o módulo CV (vocabulário controlado) ocupa um lugar de destaque no esquema. No Chado, cada tipo é representado como uma entrada na tabela **cvterm** . A tabela **cvterm** também é usada para representar todos os tipos de relações. Os links entre os tipos são representados por entradas na tabela **cvterm\_relationship** . Os tipos podem ter sinônimos e definições (especificados usando linguagem natural e definições lógicas interpretadas por máquina). O módulo CV do esquema Chado representa esses dados e outros relacionados.

**4.2 Módulo de recurso de sequência**

O módulo de **sequência** , mais particularmente a tabela de **recursos** neste módulo, é central para o gerenciamento de dados de sequência Chado. Chado define uma característica como uma região de uma macromolécula biológica (ou seja, um DNA, RNA ou uma molécula de polipeptídeo) ou um agregado de regiões neste polímero. Como o termo é usado aqui, 'região' pode ser toda a extensão da molécula ou uma junção entre duas bases. Os recursos são digitados de acordo com o SO, podem ser localizados em relação a outros recursos e podem ter certas relações com outros recursos.

**4.2.1 Recursos e sequências**

No Chado, todas as entidades geneticamente codificadas ou transmitidas, incluindo genes, transcritos, proteínas, alelos e assim por diante, são modelados como entradas na tabela de **características** da **sequência**módulo. Absolutamente nenhuma distinção é feita entre uma entidade de característica e uma entidade de sequência, elas são consideradas a mesma coisa. As inscrições podem opcionalmente ter DNA ou sequência de resíduos de aminoácidos anexados, mas anexar resíduos não é obrigatório porque às vezes é necessário criar entradas de recursos para entidades cuja sequência é atualmente desconhecida (como um gene que é identificado apenas por meio de técnicas genéticas tradicionais, ou um cDNA que foi apenas parcialmente sequenciado). Este é um aspecto crucial do projeto Chado e está em contraste com outros esquemas e formatos de bioinformática comuns, onde os recursos são inerentemente artefatos com colocação obrigatória em algum sistema de coordenadas de sequência externa. Segue-se que, embora uma característica possa ou não ter sequência, toda sequência é uma característica: é impossível armazenar uma sequência no Chado, exceto como um recurso. Isso torna o Chado diferente de quase todos os outros esquemas usados ​​na genômica, pois seu design foi escolhido para refletir a realidade biológica e as considerações práticas - o Chado não precisa atribuir identificadores separados para entradas de sequência e recursos no banco de dados.

**4.2.2 Localização do recurso**

Os recursos de sequência são normalmente localizados usando um sistema de coordenadas. Chado usa um modelo de localização relativa: todas as localizações de recurso devem ser relativas a outro recurso. As características (por exemplo, exon) mantêm uma relação com um local, ou seja, coordenadas, que por sua vez mantém uma relação com uma característica de origem (por exemplo, cromossomo). Locais são armazenados na **featureloc** mesa.

Um recurso pode ter zero ou mais **featurelocs** , embora tipicamente têm ou um (para funções de que a localização é conhecida) ou zero (para características unlocalized tais como cromossomas, ou para as características para as quais o local ainda não é conhecida, descobriu usando técnicas clássicas de genética). Vários **featurelocs** são usados para alinhamentos de localizar. Um **featureloc** é um intervalo em coordenadas de sequência interbases ( [Fig. 1](javascript:;) ), delimitado pelas colunas **fmin** e **fmax** , respectivamente, representando a posição linear inferior e superior da fronteira entre as bases ou pares de bases, com direcionalidade indicada pela coluna de **vertente** .

**Figura 1.**

[Abrir em uma nova aba](https://academic.oup.com/view-large/figure/109790309/btm189f1.jpeg)[Baixe o slide](https://academic.oup.com/DownloadFile/DownloadImage.aspx?image=https://oup.silverchair-cdn.com/oup/backfile/Content_public/Journal/bioinformatics/23/13/10.1093_bioinformatics_btm189/2/btm189f1.jpeg?Expires=1619177759&Signature=A-oOoOqVwktTLD0ttEfWd5eD1pcryQQmKxsixwX4Iy1Mz9Ca8ZsGoFkdj65GOuF3OTKMXV-unU~Lt33sOF~3SqtV6qVK14aCH2iBDdEyOpc5gmT~jQf8HGNbvIY6CIbE-NLTHDVL6Ol9J2ZyNLMGMxrjb4cC7WcOQbVzZvLPvuW0bA29QXDQDetu33944SPslRZGi0EUv8YNX06aWNl96-qT-Kbbh1510eJbsEdOA~LnLjRlcmshicB107HB~s2HvxOHaq~9IXysbtPkto7eM3pnEjOh~wNAFyR6YSCsNsd0pDrPZlts1Ee-Wwi8zV00jcpIEQPX1k9Vwnic5TqGDA__&Key-Pair-Id=APKAIE5G5CRDK6RD3PGA&sec=109790309&ar=229507&xsltPath=~/UI/app/XSLT&imagename=&siteId=5139)

Chado usa o método interbase para gerenciar o comprimento da sequência. As contagens começam em zero, e o espaço entre duas bases é o que é contado. Neste exemplo, o códon para metionina começa em 0 e termina em 3.

Outros tipos de localização não orientados para sequência (como localização física de experimentos *in situ* ou localizações genéticas de estudos de ligação) são modelados fora do módulo de sequência, por exemplo, nos módulos de **expressão** ou **mapa** , mas uma discussão sobre eles está além o escopo deste artigo.

As coordenadas Interbase foram escolhidas ao invés do sistema de coordenadas orientado a base mais comumente usado porque elas são mais naturalmente receptivas às operações aritméticas padrão que são tipicamente executadas em coordenadas de sequência. Isso leva a uma lógica de codificação de banco de dados mais limpa e eficiente, provavelmente menos sujeita a erros. Obviamente, as coordenadas interbase são normalmente transformadas no sistema orientado a base mais comum usado por relatórios BLAST e, portanto, antes da apresentação ao usuário final. Conforme mencionado anteriormente, o esquema Chado inclui uma restrição que garante que **fmin**  <=  **fmax** seja sempre verdadeiro - qualquer tentativa de definir o banco de dados em um estado que viole isso irá sinalizar um erro. [2](javascript:;)

A tabela **featureloc** também contém o **srcfeature\_id** , ou seja, a chave estrangeira referenciando o recurso ao qual suas coordenadas são relativas. Não há nada no esquema que proíba as cadeias de localização; por exemplo, localizar domínios de proteína em relação ao (s) polipeptídeo (s) que, por sua vez, podem ser localizados em seus respectivos transcritos, ou localizar um exon em relação a um contig que por sua vez está localizado em relação a um cromossomo ( [Fig. 2](javascript:;)) A maioria das instâncias do banco de dados Chado não exigirá essa flexibilidade; todos os recursos são normalmente localizados em relação aos cromossomos ou estruturas cromossômicas. No entanto, a capacidade de armazenar tais redes de localização ou gráficos de localização são particularmente úteis para genomas inacabados ou partes de genomas, como heterocromatina (Hoskins *et al* ., [2002](javascript:;) ), em que é desejável localizar recursos relativos a contigs estáveis ​​ou andaimes, que eles próprios estão localizados em uma montagem instável de cromossomos ou estruturas de braços cromossômicos.

**Figura 2.**

[Abrir em uma nova aba](https://academic.oup.com/view-large/figure/109790313/btm189f2.jpeg)[Baixe o slide](https://academic.oup.com/DownloadFile/DownloadImage.aspx?image=https://oup.silverchair-cdn.com/oup/backfile/Content_public/Journal/bioinformatics/23/13/10.1093_bioinformatics_btm189/2/btm189f2.jpeg?Expires=1619177759&Signature=YeNnaw2NxYfwuN39x7fIjLOUAC-TOUzfq~7Tt9CnznycJTO5oQq3-gMW8UIefgXBQb7cgwJlLwW51Xh0Ti6R8lq2vc0i1ASXWbrx4yAi8nChfbE55H--lI~OB04yXKmafamePGSNhchC-ki4A8zbubWATSjSH7oLXuF8wk3rs5x8M9WNqZ4DIjQ0b7Li679eV6RgXfPaerhRyHPYgYX1Lll44VaruD8TiIbZQarFImdp9e5c5cxuT2027dzAz-rHwGJ6Z0HF48DGPZApLAtaTwd-H0hFVZrMajUk9guvj6bYP1mZghHoUtMEHL22Y1cYSP9ALA~Zkzdg8BdEYaq~SQ__&Key-Pair-Id=APKAIE5G5CRDK6RD3PGA&sec=109790313&ar=229507&xsltPath=~/UI/app/XSLT&imagename=&siteId=5139)

Localização de recursos representados no Chado. ( **a** ) visualização de um gene localizado em relação a um recurso contig. Os exons são numerados e representados em azul, o contig em verde. As linhas pontilhadas mostram a localização relativa do início do exon2 como 1 kb ao longo do contig e o final como 2 kb. ( **b** ) O mesmo gene em um nível de zoom diminuído. O contig na legenda é mostrado localizado 48 k a montante do início do cromossomo 2. As linhas pontilhadas mais finas mostram a localização do exon 2 projetada em chr2, 49 kb a montante da origem. ( **c**) Uma representação abstrata de linhas no banco de dados Chado, com recursos denotados por meio de caixas ovais e featurelocs denotados por setas. Exon2 está localizado diretamente em relação a contig234, contig234 está localizado diretamente em relação a chr2. Exon2 também está indiretamente localizado em relação a chr2 usando um featureloc com locgroup definido como 1.

Apresentaremos agora um breve tratamento formal das propriedades dessas hierarquias de localização usando a teoria dos grafos. Esse tratamento pode ser ignorado para fins de compreensão dos fundamentos do gerenciamento de dados de sequência no esquema Chado; o usuário final do banco de dados não terá conhecimento de tais detalhes técnicos. No entanto, para fins de engenharia de software e garantia de interoperabilidade entre diferentes instâncias de banco de dados Chado e diferentes aplicativos, tratamentos formais como esses são um requisito essencial para as especificações de software.

Definimos um grafo featureloc (LG) como um conjunto de vértices e arestas, com cada feature constituindo um vértice, e cada **featureureloc** constituindo uma aresta indo do vértice **feature\_id** pai para o vértice **srcfeature\_id** . O nó é marcado com valores de coluna a partir da **característica** de mesa, e a borda é marcado com os valores da coluna do **featureloc** tabela. O LG não pode conter ciclos: é um gráfico acíclico direcionado (DAG). Isso inclui ciclos automáticos, já que nenhum recurso pode ser localizado em relação a si mesmo.

As *raízes* do LG são as características que não têm uma localização associada - freqüentemente cromossomos ou braços cromossômicos, embora as raízes do LG também possam ser contigs desmontados, andaimes ou recursos para os quais a localização da sequência ainda não é conhecida. As *folhas* do LG são quaisquer recursos que nunca estão presentes como **srcfeature\_id** em qualquer linha de **featureloc** , normalmente a maior parte dos recursos, como exons, correspondências e assim por diante. A *profundidade* de um determinado LG *g* , denotado D ( *g* ), é o número máximo de arestas entre qualquer par folha-raiz. Como foi observado anteriormente, muitas implementações de Chado terão LGs com uma profundidade uniforme de 1. Esses LGs são considerados*simples* e os recursos dentro deles são considerados *singletons* . A profundidade máxima de todos os LGs em uma instância de banco de dados particular **i** é denotada LGD max ( **i** ).

O esquema não restringe a profundidade máxima do LG. Essa flexibilidade se mostra útil ao aplicar o Chado às necessidades altamente variáveis ​​de diferentes projetos de genoma; no entanto, pode levar a problemas de eficiência ao consultar o banco de dados. Também pode tornar mais difícil escrever software para interoperar com o banco de dados, já que o software deve levar em consideração profundidades LG arbitrárias.

Podemos resolver este problema, *colapso* da LG, ou seja, um gráfico de profundidade arbitrária é achatado a uma profundidade de 1, projetando **featurelocs** em níveis mais baixos para os recursos raiz. Os originais **featurelocs** são deixados inalterado no banco de dados, e redundante *inferir***featurelocs** entre características folhas e raízes são adicionados ao banco de dados. Na tabela de **featurelocs** , os **featurelocs** inferidos são diferenciados dos **featurelocs** diretos usando o **locgroup**coluna. As localizações diretas (não inferidas) são indicadas pela coluna locgroup assumindo o valor 0, e as localizações transitivas são indicadas por esta coluna tendo um valor> 0. O benefício de ser capaz de *recolher* a localização de recursos de sequência em locais inferidos é que isso permite implementações concisas de localização de recursos de qualquer profundidade, ao mesmo tempo que torna possível representar locais no formato achatado (profundidade = 1) comumente usado nas ferramentas de visualização de sequência atualmente.

Alinhamentos e características comparativas são tipicamente localizados utilizando pares de **featurelocs** . Esses recursos incluem acertos e pares de alta pontuação (HSPs) provenientes de programas de busca de sequência, como BLAST, regiões cromossômicas sintênicas e variações de sequência, como SNPs (polimorfismos de nucleotídeo único). Tais características têm dois **featurelocs** : um em relação ao recurso de consulta ou a variação, e um em relação ao tema (hit) ou recurso de referência. Nós diferenciar as duas **featurelocs** usando a **classificação** de coluna. Uma **classificação** de 0 indica uma localização em relação à consulta (como é o padrão para a maioria dos recursos), e uma **classificação** de 1 indica uma localização em relação ao recurso de assunto (ocorrência) ( [Fig. 3](javascript:;)) Para alinhamentos múltiplos (por exemplo, resultados do CLUSTALW, ver Higgins *et al* ., [1994](javascript:;) ), este esquema é estendido para classificações ilimitadas [0 ... *n* ], com ordenação arbitrária. Alinhamentos e sequências variantes podem ser armazenados na coluna **waste\_info** . O formato CIGAR definido pelo projeto Ensembl (Hubbard *et al* ., [2005](javascript:;) ) é usado para alinhamentos de pares. O **feature\_id, rank** e **locgroup** triple identificam exclusivamente um **featureloc** para qualquer **característica** particular . Isto significa que nenhum recurso pode ter mais de um **featureloc** com a mesma **classificação** e**locgroup** .

**Fig. 3.**

[Abrir em uma nova aba](https://academic.oup.com/view-large/figure/109790320/btm189f3.jpeg)[Baixe o slide](https://academic.oup.com/DownloadFile/DownloadImage.aspx?image=https://oup.silverchair-cdn.com/oup/backfile/Content_public/Journal/bioinformatics/23/13/10.1093_bioinformatics_btm189/2/btm189f3.jpeg?Expires=1619177759&Signature=t8x1mxyi1linQQ-fYaIQ7qi0tqsIwQYAIbQosF3Z0zQLNC6FueqzsYsN5BumOeNKUU7AwabjAN04YjXQLu9bEyMD2G6KiMRAufRpLcKgVMpyvLF93z-7sdVp0fD66ZTNoAWgIaiKwSVwETaClRYWIpZpDqMd10US6B-ulXbY~bfiJcjrQT1YgTYkwkg9-5B1oV9VXH3xIe0MtPTc-jTHKAYDhFXibPdD03e4P3jbehzpt54tO7hN-35tuL6wovbeq0bYTAiQzCV7LtcPsGK94WaSCKJc3zbMAYJGy~3XwbnHTTf8BcFy9AfC97FiliXqZkHPl0KlNNNH7irlTOpxyw__&Key-Pair-Id=APKAIE5G5CRDK6RD3PGA&sec=109790320&ar=229507&xsltPath=~/UI/app/XSLT&imagename=&siteId=5139)

Isso mostra as tabelas principais no módulo de sequência Chado. Algumas tabelas e colunas foram omitidas para tornar o diagrama mais conciso.

A implementação de todas as sequências como recursos, *incluindo aqueles com e sem sequência conhecida* , combinada com a tabela de **featureloc** - que permite que qualquer recurso seja localizado em relação a qualquer outro recurso - fornece uma vantagem expressiva significativa sobre outros modelos de sequência biológica (ver Stajich e Lapp, [2006](javascript:;) para uma visão geral), como GFF3 (Eilbeck e Lewis, [2004](javascript:;) ), GenBank, BioSQL ( [http://www.biosql.org](http://www.biosql.org/) ), BioPerl (Stajich *et al* ., [2002](javascript:;)), etc. No Chado, um recurso pode ser localizado para o recurso de origem mais apropriado. Por exemplo, o produto de proteína de um gene contendo íntron pode ser localizado diretamente como um único local contínuo sobre o transcrito que o codifica. Em outros modelos, a localização de tal proteína só pode ser expressa como o conjunto descontínuo de localizações genômicas correspondentes às regiões codificadoras de proteínas dos exons do gene. Assim, a localização do Chado reflete mais de perto a relação biológica real entre o transcrito e a proteína e, simultaneamente, evita ter que gerenciar localizações descontínuas para alguns recursos, que são propensos a ambigüidade.

**4.2.3 Relações entre recursos**

Além de relacionar características de sequência por sua localização em relação umas às outras, também é desejável capturar relacionamentos que não sejam baseados em localização. Por exemplo, os transcritos precisam estar relacionados à característica do gene de que fazem parte, da mesma forma as características do exon precisam estar relacionadas aos transcritos dos quais fazem parte, e um polipeptídeo particular pode estar em uma relação **derives\_from** a uma molécula de mRNA particular. A coleção de tais links de relacionamento é representada no Chado usando a tabela **feature\_relationship** e é conhecida como gráfico de características (FG). 'Sujeito' e 'Objeto' descreve o papel linguístico que os dois recursos desempenham em uma frase que descreve a relação **característica**. Em inglês, muitas sentenças seguem uma ordem de palavras sujeito, predicado, objeto. Dizer que 'exons são parte\_de transcrições' é a maneira correta de descrever uma relação biológica típica. Dizer que 'as transcrições são parte\_de exons' é gramatical ou biologicamente incorreto.

Uma vez que os tipos de relações de recursos podem ser tão variados quanto os recursos, ontologias são usadas para digitar essas relações em **feature\_relationship** . No caso de características de sequência, os tipos **feature\_relationship** são fornecidos por SO (Eilbeck *et al* ., [2005](javascript:;) ) ou pela ontologia de relacionamento OBO, OBO-REL (Smith *et al* ., [2005](javascript:;) ).

Alguns exemplos de gráficos de recursos (FGs) são mostrados ( [Fig. 4](javascript:;) ). O FG é independente do LG e, em geral, o FG e o LG não devem ter arestas em comum. Se houver um **featureloc** conectando dois recursos, a adição de um **feature\_relationship** entre esses mesmos dois recursos é redundante. O FG é necessário para consultar o banco de dados em busca de coisas como genes emendados alternadamente, exons compartilhados entre transcrições, etc.

**Fig. 4.**

[Abrir em uma nova aba](https://academic.oup.com/view-large/figure/109790326/btm189f4.jpeg)[Baixe o slide](https://academic.oup.com/DownloadFile/DownloadImage.aspx?image=https://oup.silverchair-cdn.com/oup/backfile/Content_public/Journal/bioinformatics/23/13/10.1093_bioinformatics_btm189/2/btm189f4.jpeg?Expires=1619177759&Signature=BDceODtHQkdXLt4OEzrySwrDRMY4aYBDJOjw9ZdHCUc0qRHPSopQEJQadumsPtCiZxotaqeZz3DCVYE9KRMtScEJ7fYrhDMN7ENwzeDfoslcmD5VExhN0R7wKSdj7JK2GwIQYqR3NN3NcQg9AechozQVZFdh5lHMbZUxzsMz~9hAZ3XjgJScM~z9tARnqSyzXYjjviN2tD5tdPpcoDzq6NUvMWYsH1u2cleK8VIRbrwVZ0IHTA6BIqgBEWcDW-5ZlU3HlvgsR8de5qU-0u5sXlmkx8hPXQ9rALXFmzHAELSsdQiDl5ipOj2cPb6jfiTNyWKhHeXG0SiZ5nTjiBbgLg__&Key-Pair-Id=APKAIE5G5CRDK6RD3PGA&sec=109790326&ar=229507&xsltPath=~/UI/app/XSLT&imagename=&siteId=5139)

Um gráfico de característica central do dogma Chado para um gene codificador de proteína com duas formas de splice. ( **a** ) Visualização baseada em glifos. As caixas mostram os exões, as linhas de ligação mostram os intrões. UTRs mostradas em azul, regiões CDS em vermelho. ( **b** ) Representação gráfica de feições de linhas em um banco de dados Chado: caixas ovais denotam feições e linhas denotam relações\_de\_características. As características são do tipo exon, mRNA, gene e proteína. As relações\_de\_características são do tipo part\_of, e as linhas verdes entre a proteína e o mRNA representam as relações derivadas\_de. Os exons 2 e 4 são compartilhados entre as duas formas de emenda. Observe que o gráfico de características transmite informações topológicas e temporais, não, por exemplo, a ordem linear dos exons (ou seja, os números dos exons não denotam ordem).

Embora o esquema Chado admita qualquer FG, certas configurações são biologicamente insignificantes e não devem ser usadas. Ao contrário do LG, o FG pode ser cíclico, embora os ciclos no FG não sejam comuns. O subconjunto do FG correspondente a certos tipos de relacionamento pode ser acíclico, por exemplo, o subconjunto das partes de conexão FG com todos via **part\_of** deve ser acíclico.

**4.2.4 Propriedades de recursos extensíveis**

A tabela de **recursos** tem um conjunto muito limitado de colunas para registrar atributos ou propriedades de recursos. Foi deliberadamente decidido não adicionar colunas como 'anticódon' à tabela de **recursos** , uma vez que, considerando os muitos tipos diferentes de recursos sendo armazenados e os muitos atributos que podem ou não estar associados a cada um desses tipos de recursos, o número de colunas na tabela estava sujeito a se tornar muito grande e difícil de gerenciar, com muitas colunas sendo nulas para a maioria dos recursos (por exemplo, 'anticódon' não se aplica a recursos não tRNA).

Chado usa uma única tabela chamada **featureprop** para armazenar atributos ou propriedades de qualquer recurso. Para recursos de sequência, a ontologia de propriedade de recurso de sequência (consulte o ramo sequence\_attribute em <http://www.sequenceontology.org/miSO/index.html> ) é normalmente usada para definir tipos de propriedade, de forma semelhante a como é usada para lidar **com** tipos **feature\_relationship** . Esse uso de um vocabulário controlado em vez de colunas de tabela explícitas permite que novas propriedades sejam adicionadas facilmente e sem qualquer interrupção do esquema ou de qualquer software que use o esquema. A procedência de cada atributo é anexada por meio do **link featureprop\_pub** tabela, que pode identificar a pessoa que fez a curadoria desse recurso, o programa de análise de sequência que o previu ou a publicação em que foi descrito pela primeira vez.

**4.2.5 Sinônimos de recursos**

Os recursos podem ter vários nomes e sinônimos. Isso é modelado no Chado com a tabela de **sinônimos** , que vincula a recursos por meio da tabela de vinculação **feature\_synonym** . Todos os links **feature\_synonym** têm um atributo booleano **is\_current** , que distingue entre nomes em uso atual e nomes alternativos ou obsoletos. Dependendo do histórico experimental de um recurso, vários recursos podem potencialmente compartilhar um sinônimo comum e um único recurso pode ter vários sinônimos. A proveniência de um sinônimo particular é indicada usando a chave estrangeira **pub\_id** , que faz referência à tabela **pub** .

**4.2.6 Anotações de recursos**

Anotações detalhadas de características, associando características com tipos da ontologia do gene (GO, Ashburner *et al* ., [2000](javascript:;) ) ou da ontologia celular (Bard *et al* ., [2005](javascript:;) ), podem ser realizadas usando a tabela de ligação **feature\_cvterm** . Vários termos de ontologia podem ser associados a cada recurso, e termos de várias ontologias também podem ser associados a cada recurso. Os dados de **proveniência** podem ser anexados com as tabelas de vinculação de ordem superior **feature\_cvtermprop** e **feature\_cvterm\_dbxref** .

Cabe à política de curadoria de cada instância individual do banco de dados Chado decidir quais tipos de recursos serão vinculados à ontologia e a outros termos de vocabulário controlado usando feature\_cvterm. Alguns podem vincular termos a características gênicas, outros a produtos gênicos distintos (RNAs e polipeptídeos processados) ligados às características gênicas.

**4.3 A API Chado SQL**

Chado tem uma biblioteca de funções SQL para realizar operações úteis em dados biológicos. A biblioteca de funções é organizada pelo módulo de esquema pai correspondente. Algumas funções incluem:

* Operações de alcance em recursos e recursos; por exemplo
  + Interseção
  + Sobreposição
  + Contenção
  + Diferença de alcance
* Projeção (ainda não implementado)
* Operações em sequências biológicas; por exemplo
  + Complementação reversa de DNA
  + Tradução de DNA para aminoácido
  + Unindo recursos, como exons em uma transcrição
* Operações gráficas em ontologias; por exemplo, fechamento transitivo dinâmico de relações
* Operações em árvores filogenéticas; por exemplo
  + Calculando a altura e profundidade de qualquer nó em uma árvore
  + Teste de monofileticidade
* Operações de fluxo de dados de curadoria; por exemplo
  + Fornecimento de novos identificadores **dbxref** usando regras padrão
  + Dividindo e mesclando recursos

Todas essas funções podem ser codificadas no software aplicativo ou no middleware e, em muitos casos, serão. No entanto, há um enorme benefício em fornecer também essas funções dentro do SGBD, uma vez que podem ser incorporadas às consultas SQL, aumentando o poder e a eficiência da linguagem de consulta.

Cada função tem uma *definição de interface* , que é independente do DBMS, e *implementações* específicas do DBMS . Atualmente, as únicas implementações são para a linguagem PL / PgSQL específica do PostgreSQL. No entanto, a maioria pode ser convertida para a sintaxe SQL99 padrão com bastante facilidade e, portanto, implementada em outros SGBDs, como DB2 e Oracle.

Nas implementações do PostgreSQL, usamos tipos de dados específicos do DBMS para estender o SQL padrão, a fim de fornecer operações mais rápidas em coisas como operações de intervalo ( <http://www.iscb.org/ismb2003/posters/hlappATgnf.org_326.html> ) Esses detalhes de implementação são ocultados do software que chama as funções, exceto na medida em que as torna mais rápidas.

**4.4 Interoperação de software e camadas de ponte**

Os recursos da sequência Chado foram incorporados às versões atuais de vários aplicativos de software, por exemplo: Apollo (Lewis *et al* ., [2002](javascript:;) ), Gbrowse (Stein *et al* ., [2002](javascript:;) ) e CGL (Yandell *et al* ., [2006](javascript:;) ). Conforme observado anteriormente, o uso do Chado da tabela de **recursos** para armazenar todos os recursos, não importa o tipo, é útil para reduzir as alterações de esquema. No entanto, pode ter o efeito prejudicial de introduzir uma camada extra de abstração, complicando consultas simples. Por exemplo, uma consulta para determinar quantos genes existem no banco de dados requer uma junção e restrição na coluna de nome no **cvterm**tabela. As visualizações, ou camadas de ponte, são nossa solução para esse problema.

**4.4.1 Visualizações DBMS**

Todos os principais SGBDs permitem a definição de visualizações do banco de dados; uma visão é uma espécie de tabela virtual, definida em termos de tabelas ou outras visões. As visualizações são usadas para apresentar um meio simplificado de consultar o banco de dados; isso é particularmente útil para ajudar os usuários [3](javascript:;) a formular consultas. Para fins de consulta ao banco de dados, as visualizações atuam como tabelas; o DBMS irá reescrever de forma transparente qualquer consulta que faça referência a uma visão em uma usando as tabelas correspondentes. As visualizações também podem ser usadas para gravar no banco de dados, mas isso requer a codificação de funções de gatilho específicas para cada visualização.

As visualizações também podem ser usadas como uma camada de isolamento, protegendo os aplicativos de alterações no esquema. A evolução do esquema tem o potencial de incorrer em altos custos de desenvolvimento de software, pois as alterações no banco de dados podem se infiltrar em vários aplicativos ou código de middleware. Se o esquema for modificado (de uma maneira não compatível com versões anteriores), as visualizações podem emular versões anteriores do esquema.

As visualizações também podem ser usadas para definir um 'esquema de exportação' comum para permitir que os aplicativos interoperem com vários esquemas diferentes. Isso também pode ser feito para oferecer suporte ao desenvolvimento de aplicativos, nos casos em que uma versão simplificada de um esquema complexo pode ser apresentada por meio de uma camada de visualização para determinados aplicativos. De forma semelhante, as visualizações podem ser usadas para fazer um esquema parecer outro esquema, chamamos isso de *ponte* ou camada de *compatibilidade* .

O nome coletivo para tabelas e visualizações é uma *relação* . [4](javascript:;) Chado é um esquema relacional e, como tal, é uma coleção de definições de relações. Essas definições são fornecidas como definições de tabela por uma questão de conveniência, mas não há razão para que qualquer uma dessas tabelas não deva ser trocada por uma visualização equivalente (com a condição de que as funções de gatilho apropriadas sejam criadas se o banco de dados for atualizado ) Ocasionalmente, isso pode ser desejável por motivos de desempenho.

As visualizações também podem ser transformadas em tabelas, que podem ser indexadas para eficiência. Isso é conhecido como *materialização da* visão . As visualizações materializadas são mais úteis em situações em que os dados são lidos com frequência e raramente ou nunca modificados, como uma instância de relatório / warehouse de um banco de dados.

As visualizações são indispensáveis ​​para consultar e desenvolver software interoperável para bancos de dados relacionais complexos como o Chado, onde o alto grau de abstração e normalização torna isso mais difícil. No Chado, as visualizações são organizadas por módulo. Conflitos de nomes entre as tabelas e visualizações centrais do Chado são evitados colocando-se as visualizações em diferentes *espaços de tabela* (também referidos como *esquemas* no contexto do PostgreSQL DBMS).

**4.4.2 Camadas da ponte Chado**

SOFA é um subconjunto estável de todo o SO; as liberações são feitas anualmente. A camada SOFA fornece uma visão (ou tabela) para cada tipo na ontologia SOFA. Essa estabilidade significa que o software codificado para a camada SOFA estará menos sujeito aos efeitos negativos da evolução do esquema. A camada SOFA é gerada automaticamente e uma camada padrão é fornecida. Os administradores do banco de dados Chado podem optar por regenerar essa camada como uma camada de visão materializada (tabela), se desejado.

Camadas de ponte também estão disponíveis para o banco de dados GO (Harris *et al* ., [2004](javascript:;) ), que é um recurso disponibilizado pelo GO Consortium, e uma parte central do GO DB é um conjunto de tabelas para modelar ontologias, com muitas semelhanças com o módulo Chado CV. Essas semelhanças tornam relativamente simples definir uma coleção de visualizações em um banco de dados Chado que permite que funcione com software desenvolvido para o banco de dados GO, incluindo software desenvolvido pelo GO Consortium, como o navegador de ontologia AmiGO (consulte [http: // www. godatabase.org/cgi-bin/amigo/go.cgi](http://www.godatabase.org/cgi-bin/amigo/go.cgi) ).

**4.5 Reutilizar e o projeto GMOD**

A tarefa de gerenciar com eficácia dados biológicos em escala de genoma é um enorme desafio de engenharia de software. Dados os recursos finitos, seria sensato compartilhar os componentes de software tanto quanto possível. Esta foi a força motriz por trás da criação do projeto GMOD. Apoiado por vários dos principais projetos de organismo modelo, que contribuíram com financiamento, software e desenvolvedores, o projeto GMOD se tornou uma grande força organizacional no desenvolvimento de software por projetos de organismo modelo em todo o mundo. O GMOD exigia que seu esquema adotado fosse *adaptável* , a fim de ser alterado rapidamente quando novos tipos de dados ou relações são exigidos para representar fielmente a ciência; que tenha uma *flexibilidade* adequadamodelo, que pode representar apenas aqueles dados que são pertinentes a uma determinada área de pesquisa; que deve espelhar a precisão dos dados experimentais e, portanto, as representações devem suportar uma *gama* variada *de precisão de classificação* ; que deve oferecer *suporte* efetivo e confiável de *consulta e inferência* ; que deve ser *interoperável* com outro software; e, finalmente, que deve *reduzir os custos de manutenção* . GMOD agora emprega Chado como seu esquema de banco de dados comum.

**5 DISCUSSÃO**

Chado é um esquema altamente genérico e flexível. Isso permite que um esquema uniforme seja usado por diferentes projetos cobrindo uma variedade de genomas, sem ter que antecipar em detalhes os requisitos individuais de cada um desses projetos. Ele também fornece uma grande quantidade de prova futura do esquema em face das mudanças em nossa compreensão da biologia e mudanças nos experimentos e processos relacionados à descoberta biológica. Sem essa flexibilidade, o esquema estaria em um estado constante de migração, gerando grandes custos para desenvolvedores e usuários. Além disso, o esquema provavelmente se bifurcaria sempre que diferentes desenvolvedores de software precisassem modificá-lo para atender às necessidades de seus projetos individuais.

Observamos que essa flexibilidade em um esquema tem seus próprios custos. O obstáculo mais significativo na adoção do Chado é estabelecer as melhores práticas para implementação em um esquema tão flexível. Sem restrições, grupos diferentes podem modelar tipos semelhantes de dados de maneiras diferentes. Por exemplo, na tipagem e topologia de FG, há muitas opções a serem feitas ao representar um modelo de gene: os recursos de íntron e UTR devem ser manifestados (ou seja, existem como linhas reais no **recurso**tabela) ou implícito à esquerda (ou seja, derivado do software conforme necessário)? Que tipo de SO deve ser usado para representar o produto translacional de um mRNA: CDS ou polipeptídeo? Em ambos os casos, as opções são logicamente válidas, mas o software que não está ciente da equivalência prática desses dois tipos não funcionará igualmente bem nessas diferentes instâncias de esquema. Sem um entendimento claro e diretrizes para implementar dados no Chado usando ontologias atuais, será difícil, senão impossível, continuar a desenvolver software de aplicativo interoperável de qualquer tipo de complexidade. Embora o uso da ontologia dentro do esquema seja de grande benefício, ainda há uma tarefa significativa enfrentada pelos desenvolvedores de aplicativos, que devem compreender os fundamentos da estrutura e da lógica das ontologias se quiserem alcançar a interoperabilidade.

O esquema flexível do Chado é menos genérico do que o modelo RDF (Brickley e Guha, [2000](javascript:;) ), embora os dois sejam comparáveis, como pode ser visto no padrão triplo 'sujeito-predicado-objeto' empregado no Chado. No entanto, também existem diferenças entre os sistemas, como pode ser visto pelo fato de que os triplos Chado costumam levar argumentos adicionais (como, por exemplo, no caso de ordenar exons em uma forma de splice) em contraste com o padrão estrito de 3-árias. em RDF. No entanto, é nossa intenção ser compatível e reutilizar a tecnologia conforme apropriado, de modo que continuamos atentos à evolução do RDF e da tecnologia associada.

Os planos futuros para o Chado incluem a criação de novos módulos e o refinamento dos módulos existentes para outros domínios, como associações genótipo-ambiente-fenótipo. Isso também exigirá o desenvolvimento de ontologias para esses domínios. Também pretendemos trabalhar mais um pacote de benchmarking que nos permitirá visar áreas nas quais podemos tornar os tempos de resposta de consulta mais rápidos.

Em resumo, o Chado usa um novo método de restringir a variedade de representações, ao mesmo tempo que mantém a flexibilidade útil, por meio do uso de ontologias e níveis de conformidade em camadas. O Chado respondeu com sucesso à nossa pergunta inicial, se poderíamos criar um esquema de banco de dados relacional que poderia e seria reutilizado, conforme atestado pelo número de grupos que adotaram o Chado para sistemas de produção. Além do FlyBase, esses grupos incluem: The Institute for Genomic Research; BeetleBase (Wang *et al* ., [2007](javascript:;) ); SpBase no Instituto de Tecnologia da Califórnia para Ouriços-do-mar; DictyBase na Northwestern University; GeneDB no Instituto Sanger para agentes patogênicos (por exemplo*Trypanosoma brucei, Leishmania major, Plasmodium falciparum, Staphylococcus aureus, Salmonella typhi, Schizosaccharomyces pombe* e pontuações mais); Sol Genomics Network na Cornell University for *Solanaceae* (tomates, etc.); Universidade de Utah, Laboratório Sanchez, para *Schmidtea mediterranea* (planaria); Banco de dados do genoma *Xenopus* na Universidade de Calgary; a base de dados Paramecium do Centre National de la Recherche Scientifique, (Arnaiz *et al* ., [2007](javascript:;) ); Banco de dados MicroArray da Princeton University e o Saccharomyces Genome Database-Lite; o Centro Nacional de Síntese Evolucionária para *Heliconius*(borboleta); o Daphnia Water Flea Genome Database Universidade de Indiana; VectorBase para A. gambiae na Notre Dame University; a Universidade de Wisconsin no Instituto de Pesquisa Wicell; Infobiogen na França; e na Universidade da Califórnia em Los Angeles, no Laboratório Patricia Johnson, para estudos de *Trichomonas vaginalis*e o Laboratório Stan Nelson para estudos em genética humana usando microarranjos. Embora nossa solução técnica não seja uma panacéia para o desafio de refletir com precisão o conhecimento biológico em um formato computável, é um passo na direção certa. Ao alavancar a representação da biologia codificada em ontologias, a representação física (no esquema) é mais adaptável e flexível e, portanto, reduz os custos de manutenção; ao mesmo tempo, as ontologias podem ser usadas para oferecer suporte a precisão variada na classificação, suporte de consulta e inferência mais eficaz e suporte à interoperabilidade com outro software.

**RECONHECIMENTOS**

Os autores agradecem o apoio e os conselhos de todos os nossos colegas do Consórcio FlyBase, cuja experiência e iniciativa levaram ao desenvolvimento de Chado. Também reconhecemos com gratidão a contribuição de todos no Consórcio GMOD, especialmente Scott Cain, Brian Osborne e Guanming Wu por levar este trabalho adiante para implementação por outros grupos de banco de dados. FlyBase é apoiado por uma bolsa dos Serviços de Saúde Pública (bolsa NIH 5P41 HG000739, através do Instituto Nacional de Pesquisa do Genoma Humano, W. Gelbart, PI) com suporte adicional para o desenvolvimento do Chado do HHMI (G. Rubin, PI). Finalmente, somos extremamente gratos aos usuários do Chado, cujo feedback e suporte continuam a melhorar o Chado e seu software associado.

Contribuições do Consórcio FlyBase Membros atuais e anteriores do Consórcio FlyBase que fizeram contribuições notáveis ​​para este projeto são William M. Gelbart, Aubrey de Gray, Stan Letovsky, Suzanna E. Lewis, Gerald M. Rubin, ShengQiang Shu, Colin Wiel, Peili Zhang e Pinglei Zhou.

*Conflito de interesses* : nenhum declarado.

† Os autores desejam que se saiba que, em sua opinião, os dois primeiros autores devem ser considerados como Primeiros Autores conjuntos.

1

Na verdade, existem diferenças cruciais entre ontologias e vocabulários. No entanto, nem todos concordam sobre o que são. Para os fins deste artigo, é mais simples ignorar essas diferenças.

2

Essa restrição pode ser relaxada se a intenção for modelar genomas circulares, como os encontrados em algumas bactérias.

3

Neste contexto, 'usuários' refere-se a usuários avançados interessados ​​em mineração de dados, bem como programadores que escrevem software de relatório, em oposição a usuários finais de biólogos de bancada.

4

Não deve ser confundido com o sentido de como essa palavra é usada em ontologias.

**REFERÊNCIAS**

Arnaiz

O

, et al.

ParameciumDB: um recurso da comunidade que integra a sequência do genoma do *Paramecium tetraurelia* com dados genéticos

,

*Res de ácidos nucléicos*

,

2007

, vol.

35

 (pág.

D439

-

D444

)

Epub

[Google Scholar](https://scholar.google.com/scholar_lookup?title=ParameciumDB%3A%20a%20community%20resource%20that%20integrates%20the%20Paramecium%20tetraurelia%20genome%20sequence%20with%20genetic%20data&author=O%20Arnaiz&publication_year=2007&journal=Nucleic%20Acids%20Res&volume=35&pages=D439-D444)

[Crossref](http://dx.doi.org/10.1093/nar/gkl777)

[PubMed](http://www.ncbi.nlm.nih.gov/pubmed/17142227)

Ashburner

M

, et al.

Ontologia genética: ferramenta para a unificação da biologia. O consórcio de ontologia genética

,

*Nat. Genet*

,

2000

, vol.

25

 (pág.

25

-

29

)

[Google Scholar](https://scholar.google.com/scholar_lookup?title=Gene%20ontology%3A%20tool%20for%20the%20unification%20of%20biology.%20The%20gene%20ontology%20consortium&author=M%20Ashburner&publication_year=2000&journal=Nat.%20Genet&volume=25&pages=25-29)

[Crossref](http://dx.doi.org/10.1038/75556)

[PubMed](http://www.ncbi.nlm.nih.gov/pubmed/10802651)

Bardo

J

, et al.

Uma ontologia para tipos de células

,

*Genome Biol*

,

2005

, vol.

6

 pág.

R21

[Google Scholar](https://scholar.google.com/scholar_lookup?title=An%20ontology%20for%20cell%20types&author=J%20Bard&publication_year=2005&journal=Genome%20Biol&volume=6&pages=R21)

[Crossref](http://dx.doi.org/10.1186/gb-2005-6-2-r21)

[PubMed](http://www.ncbi.nlm.nih.gov/pubmed/15693950)

Bedel

GW

,

Tatum

EL

.

Controle genético de reações bioquímicas em neurosspora

,

*Proc. Natl Acad. Sci*

,

1941

, vol.

27

 (pág.

499

-

506

)

[Google Scholar](https://scholar.google.com/scholar_lookup?title=Genetic%20control%20of%20biochemical%20reactions%20in%20neurospora&author=GW%20Beadle&author=EL%20Tatum&publication_year=1941&journal=Proc.%20Natl%20Acad.%20Sci&volume=27&pages=499-506)

[Crossref](http://dx.doi.org/10.1073/pnas.27.11.499)

Brickley

D

,

Guha

RV

.

Especificação de esquema 1.0 do framework de descrição de recursos (RDF)

,

*Recomendação do candidato W3C*

,

2000

[Google Scholar](https://scholar.google.com/scholar_lookup?title=W3C%20Candidate%20Recommendation&author=D%20Brickley&author=RV%20Guha&publication_year=2000&book=W3C%20Candidate%20Recommendation)

Clark

T

, et al.

Identificação de objetos distribuídos globalmente para bases de conhecimento biológico

,

*Breve Bioinform*

,

2004

, vol.

5

 (pág.

59

-

70

)

[Google Scholar](https://scholar.google.com/scholar_lookup?title=Globally%20distributed%20object%20identification%20for%20biological%20knowledgebases&author=T%20Clark&publication_year=2004&journal=Brief%20Bioinform&volume=5&pages=59-70)

[Crossref](http://dx.doi.org/10.1093/bib/5.1.59)

[PubMed](http://www.ncbi.nlm.nih.gov/pubmed/15153306)

Durbin

R

,

Theyry-Mieg

J

.

ACeDB

,

*Métodos Computacionais em Pesquisa de Genoma*

,

1994

Nova york

Plenum

[Google Scholar](https://scholar.google.com/scholar_lookup?title=Computational%20Methods%20in%20Genome%20Research&author=R%20Durbin&author=J%20Theirry-Mieg&publication_year=1994&book=Computational%20Methods%20in%20Genome%20Research)

Eilbeck

K

,

Lewis

S

.

Guia de anotação de ontologia de sequência

,

*Comp. Funct. Genômica*

,

2004

, vol.

5

 (pág.

642

-

647

)

[Google Scholar](https://scholar.google.com/scholar_lookup?title=Sequence%20ontology%20annotation%20guide&author=K%20Eilbeck&author=S%20Lewis&publication_year=2004&journal=Comp.%20Funct.%20Genomics&volume=5&pages=642-647)

[Crossref](http://dx.doi.org/10.1002/cfg.446)

[PubMed](http://www.ncbi.nlm.nih.gov/pubmed/18629179)

Eilbeck

K

, et al.

A ontologia de sequência: uma ferramenta para a unificação das anotações do genoma

,

*Genome Biol*

,

2005

, vol.

6

 pág.

R44

[Google Scholar](https://scholar.google.com/scholar_lookup?title=The%20sequence%20ontology%3A%20a%20tool%20for%20the%20unification%20of%20genome%20annotations&author=K%20Eilbeck&publication_year=2005&journal=Genome%20Biol&volume=6&pages=R44)

[Crossref](http://dx.doi.org/10.1186/gb-2005-6-5-r44)

[PubMed](http://www.ncbi.nlm.nih.gov/pubmed/15892872)

Harris

MA

, et al.

O banco de dados Gene Ontology (GO) e recurso de informática

,

*Res de ácidos nucléicos*

,

2004

, vol.

32

 (pág.

D258

-

D261

)

[Google Scholar](https://scholar.google.com/scholar_lookup?title=The%20Gene%20Ontology%20%28GO%29%20database%20and%20informatics%20resource&author=MA%20Harris&publication_year=2004&journal=Nucleic%20Acids%20Res&volume=32&pages=D258-D261)

[Crossref](http://dx.doi.org/10.1093/nar/gkh066)

[PubMed](http://www.ncbi.nlm.nih.gov/pubmed/14681407)

Higgins

D

, et al.

CLUSTALW: melhorando a sensibilidade do alinhamento de sequência múltipla progressiva por meio de ponderação de sequência, penalidades de lacunas específicas da posição e escolha de matriz de peso

,

*Res de ácidos nucléicos*

,

1994

, vol.

22

 (pág.

4673

-

4680

)

[Google Scholar](https://scholar.google.com/scholar_lookup?title=CLUSTALW%3A%20improving%20the%20sensitivity%20of%20progressive%20multiple%20sequence%20alignment%20through%20sequence%20weighting%2C%20position-specific%20gap%20penalties%20and%20weight%20matrix%20choice&author=D%20Higgins&publication_year=1994&journal=Nucleic%20Acids%20Res&volume=22&pages=4673-4680)

[Crossref](http://dx.doi.org/10.1093/nar/22.22.4673)

[PubMed](http://www.ncbi.nlm.nih.gov/pubmed/7984417)

Hoskins

RA

, et al.

Sequências heterocromáticas em uma montagem shotgun de genoma completo de Drosophila

,

*Genome Biol*

,

2002

, vol.

3

RESEARCH0085

[Google Scholar](https://scholar.google.com/scholar_lookup?title=Heterochromatic%20sequences%20in%20a%20Drosophila%20whole-genome%20shotgun%20assembly&author=RA%20Hoskins&publication_year=2002&journal=Genome%20Biol&volume=3&pages=)

Hu

J

, et al.

O ARKdb: bancos de dados de genoma para animais de criação e outros

,

*Res de ácidos nucléicos*

,

2001

, vol.

29

 (pág.

106

-

110

)

[Google Scholar](https://scholar.google.com/scholar_lookup?title=The%20ARKdb%3A%20genome%20databases%20for%20farmed%20and%20other%20animals&author=J%20Hu&publication_year=2001&journal=Nucleic%20Acids%20Res&volume=29&pages=106-110)

[Crossref](http://dx.doi.org/10.1093/nar/29.1.106)

[PubMed](http://www.ncbi.nlm.nih.gov/pubmed/11125062)

Hubbard

T

, et al.

Conjunto

,

*Res de ácidos nucléicos*

,

2005

, vol.

33

 (pág.

D447

-

D453

)

[Google Scholar](https://scholar.google.com/scholar_lookup?title=Ensembl&author=T%20Hubbard&publication_year=2005&journal=Nucleic%20Acids%20Res&volume=33&pages=D447-D453)

[Crossref](http://dx.doi.org/10.1093/nar/gki138)

[PubMed](http://www.ncbi.nlm.nih.gov/pubmed/15608235)

Lewis

SE

, et al.

Apollo: um editor de anotação de sequência

,

*Genome Biol*

,

2002

, vol.

3

RESEARCH0082

[Google Scholar](https://scholar.google.com/scholar_lookup?title=Apollo%3A%20a%20sequence%20annotation%20editor&author=SE%20Lewis&publication_year=2002&journal=Genome%20Biol&volume=3&pages=)

Morgan

º

.

A causa do ginandromorfismo em insetos

,

*Sou. Nat*

,

1907

, vol.

41

 (pág.

715

-

718

)

[Google Scholar](https://scholar.google.com/scholar_lookup?title=The%20cause%20of%20gynandromorphism%20in%20insects&author=TH%20Morgan&publication_year=1907&journal=Am.%20Nat&volume=41&pages=715-718)

[Crossref](http://dx.doi.org/10.1086/278863)

Mungall

CJ

, et al.

Um pipeline computacional integrado e banco de dados para oferecer suporte à anotação de sequência do genoma inteiro

,

*Genome Biol*

,

2002

, vol.

3

RESEARCH0081

[Google Scholar](https://scholar.google.com/scholar_lookup?title=An%20integrated%20computational%20pipeline%20and%20database%20to%20support%20whole-genome%20sequence%20annotation&author=CJ%20Mungall&publication_year=2002&journal=Genome%20Biol&volume=3&pages=)

Smith

B

, et al.

Relações em ontologias biomédicas

,

*Genome Biol*

,

2005

, vol.

6

 pág.

R46

[Google Scholar](https://scholar.google.com/scholar_lookup?title=Relations%20in%20biomedical%20ontologies&author=B%20Smith&publication_year=2005&journal=Genome%20Biol&volume=6&pages=R46)

[Crossref](http://dx.doi.org/10.1186/gb-2005-6-5-r46)

[PubMed](http://www.ncbi.nlm.nih.gov/pubmed/15892874)

Stajich

JE

, et al.

O kit de ferramentas Bioperl: módulos Perl para as ciências da vida

,

*Genome Res*

,

2002

, vol.

12

 (pág.

1611

-

1618

)

[Google Scholar](https://scholar.google.com/scholar_lookup?title=The%20Bioperl%20toolkit%3A%20Perl%20modules%20for%20the%20life%20sciences&author=JE%20Stajich&publication_year=2002&journal=Genome%20Res&volume=12&pages=1611-1618)

[Crossref](http://dx.doi.org/10.1101/gr.361602)

[PubMed](http://www.ncbi.nlm.nih.gov/pubmed/12368254)

Stajich

JE

,

Lapão

H

.

Ferramentas de código aberto e kits de ferramentas para bioinformática: importância, e onde estamos?

,

*Breve Bioinform*

,

2006

, vol.

7

 (pág.

287

-

296

)

[Google Scholar](https://scholar.google.com/scholar_lookup?title=Open%20source%20tools%20and%20toolkits%20for%20bioinformatics%3A%20significance%2C%20and%20where%20are%20we%3F&author=JE%20Stajich&author=H%20Lapp&publication_year=2006&journal=Brief%20Bioinform&volume=7&pages=287-296)

[Crossref](http://dx.doi.org/10.1093/bib/bbl026)

[PubMed](http://www.ncbi.nlm.nih.gov/pubmed/16899494)

Stein

LD

, et al.

O navegador genérico do genoma: um bloco de construção para um banco de dados de sistema de organismo modelo

,

*Genome Res*

,

2002

, vol.

12

 (pág.

1599

-

1610

)

[Google Scholar](https://scholar.google.com/scholar_lookup?title=The%20generic%20genome%20browser%3A%20a%20building%20block%20for%20a%20model%20organism%20system%20database&author=LD%20Stein&publication_year=2002&journal=Genome%20Res&volume=12&pages=1599-1610)

[Crossref](http://dx.doi.org/10.1101/gr.403602)

[PubMed](http://www.ncbi.nlm.nih.gov/pubmed/12368253)

Wang

eu

, et al.

BeetleBase: o banco de dados de organismos modelo para *Tribolium castaneum*

,

*Res de ácidos nucléicos*

,

2007

, vol.

35

 (pág.

D476

-

D479

)

[Google Scholar](https://scholar.google.com/scholar_lookup?title=BeetleBase%3A%20the%20model%20organism%20database%20for%20Tribolium%20castaneum&author=L%20Wang&publication_year=2007&journal=Nucleic%20Acids%20Res&volume=35&pages=D476-D479)

[Crossref](http://dx.doi.org/10.1093/nar/gkl776)

[PubMed](http://www.ncbi.nlm.nih.gov/pubmed/17090595)

Yandell

M

, et al.

Tendências em grande escala na evolução das estruturas gênicas em 11 genomas de animais

,

*PLoS Comput. Biol*

,

2006

, vol.

2

[Google Scholar](https://scholar.google.com/scholar_lookup?title=Large-scale%20trends%20in%20the%20evolution%20of%20gene%20structures%20within%2011%20animal%20genomes&author=M%20Yandell&publication_year=2006&journal=PLoS%20Comput.%20Biol&volume=2&pages=)

© 2007 O (s) Autor (es)

Este é um artigo de acesso aberto distribuído sob os termos da Licença Não Comercial de Atribuição Creative Commons (http://creativecommons.org/licenses/by-nc/2.0/uk/) que permite o uso não comercial irrestrito, distribuição e reprodução em qualquer meio, desde que o trabalho original seja devidamente citado.