**Nome**

URI :: db - URIs de banco de dados

**Sinopse**

|  |
| --- |
| use [URI](https://metacpan.org/pod/URI);  my $db\_uri = URI->new('db:<pg://user@localhost>');  my $pg\_uri = URI->new('<postgres://example.com/template1>');  my $sl\_uri = URI->new('sqlite:/var/db/widgets.db'); |

**Descrição**

Esta classe fornece suporte para URIs de banco de dados. Eles são inspirados por [URIs JDBC](http://docs.oracle.com/cd/B14117_01/java.101/b10979/urls.htm" \l "BEIJFHHB) e [URIs](http://docs.oracle.com/cd/B14117_01/java.101/b10979/urls.htm" \l "BEIJFHHB)[PostgreSQL](http://www.postgresql.org/docs/9.3/static/libpq-connect.html#LIBPQ-CONNSTRING) , embora sejam um pouco mais formais. A especificação de seu formato está documentada em [*README.md*](https://metacpan.org/github.com/theory/db-uri/) .

**Aviso:** esta é uma versão alfa. Farei o meu melhor para preservar a funcionalidade daqui para frente, especialmente porque o [Sqitch](https://metacpan.org/pod/App::Sqitch) usa este módulo. No entanto, conforme a especificação do URI do banco de dados avança, as alterações *podem* exigir alterações incompatíveis com versões anteriores. Advertência Hackor.

**Formato**

Um URI de banco de dados é composto por estas partes:

|  |
| --- |
| db:engine:[//[user[:password]@][host][:port]/][dbname][?params][#fragment] |

**db**

A string literal dbé o esquema que define um URI de banco de dados. Opcional para motores conhecidos.

**engine**

Uma string que identifica o mecanismo de banco de dados.

**user**

O nome de usuário a ser usado ao se conectar ao banco de dados.

**password**

A senha a ser usada ao se conectar ao banco de dados.

**host**

O endereço do host ao qual se conectar.

**port**

A porta de rede à qual se conectar.

**dbname**

O nome do banco de dados. Para alguns mecanismos, esse será um nome de arquivo; nesse caso, pode ser um caminho completo ou local, conforme apropriado.

**params**

Uma string de consulta GET padrão URI que representa parâmetros adicionais a serem passados ​​para o mecanismo.

**fragment**

Identifica uma parte do banco de dados, como uma tabela ou exibição.

**Exemplos**

Alguns exemplos:

* db:sqlite
* db:sqlite:dbname
* db:sqlite:/path/to/some.db
* sqlite:../relative.db
* db:firebird://localhost/%2Fpath/to/some.db
* db:firebird://localhost//path/to/some.db
* firebird://localhost/relative.db
* db:pg://
* db:pg://localhost
* db:pg://localhost:5433
* db:pg://localhost/mydb
* db:pg://user@localhost
* db:pg://user:secret@/mydb
* pg:///mydb
* pg://other@localhost/otherdb?connect\_timeout=10&application\_name=myapp
* db://localhost/mydb
* db:unknown://example.com/mydb

**Interface**

Existem as seguintes diferenças em comparação com a URIinterface da classe:

**Método de Classe**

**default\_port**

Retorna a porta padrão do mecanismo. Este é um valor de método de classe definido por cada mecanismo URI reconhecido.

**Construtores**

**new**

|  |
| --- |
| my $uri = URI::db->new($string);  my $uri = URI::db->new($string, $base); |

Sempre retorna um objeto URI :: db. $basepode ser outro objeto URI ou string. Ao contrário da [URI](https://metacpan.org/pod/URI) s' new(), o esquema será sempre aplicada ao URI se não tiver uma.

**Acessores**

**scheme**

|  |
| --- |
| my $scheme = $uri->scheme;  $uri->scheme( $new\_scheme ); |

Obtém ou define a parte do esquema do URI. Para db:URIs, o esquema não pode ser alterado para qualquer valor diferente de "db" (ou qualquer variação de caso disso). Para não db:URIs, o esquema pode ser alterado para qualquer valor, embora o objeto URI possa não ser mais um URI de banco de dados.

**engine**

|  |
| --- |
| my $engine = $uri->engine;  $uri->engine( $new\_engine ); |

Obtém ou define a parte do mecanismo do URI, que pode ser qualquer valor de esquema de URI válido, embora os mecanismos reconhecidos forneçam contexto adicional, como o default\_port()e um específico do driver dbi\_dsn().

Se chamado com um argumento, ele atualiza o mecanismo, possivelmente alterando a classe do URI, e retorna o valor do mecanismo antigo.

**canonical\_engine**

|  |
| --- |
| my $canonical\_engine = $uri->canonical\_engine; |

Retorna o mecanismo canônico. Vários nomes de mecanismo são aliases para outros mecanismos. Este método retornará o nome do mecanismo sem alias. Por exemplo, o postgresmecanismo retornará o mecanismo canônico pg, sqlite3retornará o mecanismo canônico sqlitee mariaretornará o mecanismo canônico mysql.

**dbname**

|  |
| --- |
| my $dbname = $uri->dbname;  $uri->dbname( $new\_dbname ); |

Obtém ou define o nome do banco de dados. Se chamado com um argumento, o caminho também será atualizado.

**host**

|  |
| --- |
| my $host = $uri->host;  $uri->host( $new\_host ); |

Obtém ou define o host ao qual se conectar.

**port**

|  |
| --- |
| my $port = $uri->port;  $uri->port( $new\_port ); |

Obtém ou define a porta à qual se conectar.

**user**

|  |
| --- |
| my $user = $uri->user;  $uri->user( $new\_user ); |

Obtém ou define o nome do usuário.

**password**

|  |
| --- |
| my $password = $uri->password;  $uri->password( $new\_password ); |

Obtém ou define a senha.

**uri**

Retorna o URI do mecanismo subjacente. Para URIs começando com db:, esse será o URI que se segue. Para URIs de banco de dados sem db:, o próprio URI será retornado.

**Métodos de Instância**

**has\_recognized\_engine**

|  |
| --- |
| my $has\_recognized\_engine = $uri->has\_recognized\_engine; |

Retorna verdadeiro se o mecanismo é reconhecido por URI :: db e falso se não é. Um mecanismo reconhecido é simplesmente aquele que herda de URI::\_db.

**query\_params**

|  |
| --- |
| my @params = $uri->query\_params; |

Retorna uma lista de pares de chave / valor representando todos os parâmetros de consulta. Os parâmetros especificados mais de uma vez serão retornados mais de uma vez, portanto, evite atribuir a um hash. Se você quiser um hash, use [URI :: QueryParam](https://metacpan.org/pod/URI::QueryParam) 's query\_from\_hash(), onde chaves duplicadas levam a uma matriz de valores para essa chave:

|  |
| --- |
| use [URI::QueryParam](https://metacpan.org/pod/URI::QueryParam);  my $params = $uri->query\_form\_hash; |

**dbi\_driver**

|  |
| --- |
| if ( my $driver = $uri->dbi\_driver ) {      eval "require DBD::$driver" or die;  } |

Retorna uma string que representa o nome do driver [DBI](https://metacpan.org/pod/DBI) para o mecanismo de banco de dados, se algum for conhecido. Retorna undefse nenhum driver for conhecido.

**dbi\_dsn**

|  |
| --- |
| DBI->connect( $uri->dbi\_dsn, $uri->user, $uri->password ); |

Retorna um [DBI](https://metacpan.org/pod/DBI) DSN apropriado para uso em uma chamada para DBI->connect. Os atributos geralmente serão extraídos do nome do host URI, da porta e do nome do banco de dados, bem como dos parâmetros de consulta. Se nenhum driver for conhecido para o URI, a dbi:$driver:parte do DSN será omitida; nesse caso, você pode usar a $DBI\_DRIVERvariável de ambiente para identificar um driver apropriado. Se o URI suportar múltiplos drivers, passe o nome daquele que você deseja dbi\_dsn(). Atualmente, apenas URI :: myssql suporta drivers alternativos, ADO, ODBC ou Sybase. Caso contrário, cada URI de banco de dados fará o melhor para criar um DBI DSN válido. Alguns exemplos:

|  |
| --- |
| | URI                                  | DSN                                              |  |--------------------------------------+--------------------------------------------------|  | db:pg:try                            | dbi:Pg:dbname=try                                |  | db:[mysql://localhost:33/foo](mysql://localhost:33/foo%C2%A0%C2%A0%C2%A0%C2%A0%C2%A0%C2%A0%C2%A0%C2%A0%C2%A0) | dbi:mysql:host=localhost;port=33;database=foo    |  | db:[db2://localhost:33/foo](db2://localhost:33/foo%C2%A0%C2%A0%C2%A0%C2%A0%C2%A0%C2%A0%C2%A0%C2%A0%C2%A0%C2%A0%C2%A0) | dbi:DB2:HOSTNAME=localhost;PORT=33;DATABASE=foo  |  | db:vertica:dbadmin                   | dbi:ODBC:DSN=dbadmin                             |  | db:<mssql://foo.com/pubs?Driver=MSSQL> | dbi:ODBC:Host=foo.com;Database=pubs;Driver=MSSQL | |

**dbi\_params**

|  |
| --- |
| my @params = $uri->dbi\_params; |

Retorna uma lista de pares de chave / valor usados ​​como parâmetros no [DBI](https://metacpan.org/pod/DBI) DSN, incluindo parâmetros de consulta. Os parâmetros especificados mais de uma vez serão retornados mais de uma vez, portanto, evite atribuir a um hash.

**abs**

|  |
| --- |
| my $abs = $uri->abs( $base\_uri ); |

Para db:URIs, simplesmente retorna o próprio objeto URI :: db. Para não db:URIs, o comportamento é o mesmo que para [URI,](https://metacpan.org/pod/URI) incluindo respeito por $URI::ABS\_ALLOW\_RELATIVE\_SCHEME.

**rel**

|  |
| --- |
| my $rel = $uri->rel( $base\_uri ); |

Para db:URIs, simplesmente retorna o próprio objeto URI :: db. Para não db:URIs, o comportamento é o mesmo que para [URI](https://metacpan.org/pod/URI) .

**canonical**

|  |
| --- |
| my $canonical\_uri = $uri->canonical; |

Retorna uma versão normalizada do URI. Esse comportamento é o mesmo para outros URIs, exceto que o mecanismo será substituído pelo valor de canonical\_enginese ainda não for o mecanismo canônico.

**Apoio, suporte**

Este módulo é armazenado em um [repositório GitHub](https://github.com/theory/uri-db/) aberto . Sinta-se à vontade para distribuir e contribuir!

Envie relatórios de bug por meio do [GitHub Issues](https://github.com/theory/uri-db/issues/) ou enviando um e-mail para [bug-URI-db@rt.cpan.org](mailto:bug-URI-db@rt.cpan.org) .