IBM DB2 Information Integrator

Anexo para o Guia de Configuração de Origens de Dados: Wrapper do BioRS e UDFs (User-Defined Functions) de Biociências

Versão 8

IBM DB2 Information Integrator

Anexo para o Guia de Configuração de Origens de Dados: Wrapper do BioRS e UDFs (User-Defined Functions) de Biociências

Versão 8

Antes de utilizar estas informações e o produto suportado por elas, leia as informações gerais na seção “Avisos” na página 91.

Você pode solicitar as publicações IBM on-line ou através de seu representante IBM local:

- Para solicitar publicações on-line, consulte o IBM Publications Center em www.ibm.com/shop/publications/order
- Para localizar o representante IBM local, consulte o IBM Directory of Worldwide Contacts em www.ibm.com/planetwide

Quando o Cliente envia seus comentários, concede direitos não-exclusivos à IBM para utilizá-los ou distribuí-los da maneira que achar conveniente, sem que isto implique em qualquer compromisso ou obrigação para com o Cliente.

Índice

Capítulo 1. Configurando o Acesso a Origens de Dados do BioRS

Este capítulo explica o que é BioRS, como incluir origens de dados do BioRS no seu sistema federado e lista as mensagens de erro associadas ao wrapper do BioRS.

O que É BioRS?

BioRS é um sistema de consulta e recuperação que foi desenvolvido pela Biomax Informatics. Você pode utilizar o BioRS para recuperar informações de várias origens de dados, incluindo arquivos simples e bancos de dados relacionais. Normalmente você faz download de dados públicos, tais como SwissProt e GenBank, como arquivos simples para o sistema BioRS. O BioRS pode integrar origens de dados públicos e origens de dados patenteados (por exemplo, bancos de dados privados que são mantidos por sua organização) a um ambiente comum.

Depois que uma origem de dados é integrada ao sistema BioRS, ela é denominada *databank*. Os elementos que estão contidos em cada entrada do databank são coletivamente denominados *esquema*. Apenas os elementos de um databank que estejam indexados no sistema BioRS podem ser utilizados em uma consulta do BioRS. Você pode estabelecer relacionamentos entre as entradas nos databanks, para que possa vincular databanks no sistema BioRS.

Os databanks do BioRS podem ter um relacionamento pai-filho (os databanks podem ser aninhados). Nesse tipo de relacionamento, o databank filho contém um elemento de tipo de dados Referência denominado PARENT. O elemento PARENT refere-se ao elemento _ID_ do databank pai. Exceto pela presença deste elemento PARENT predefinido, os databanks aninhados contêm os mesmos dados que os databanks não-aninhados.

O BioRS fornece uma interface baseada na Web que permite que os usuários executem consultas de dados nos databanks do BioRS. O wrapper do BioRS utiliza as mesmas APIs (Interfaces de Programação de Aplicativo) que a interface baseada na Web do BioRS para executar consultas.

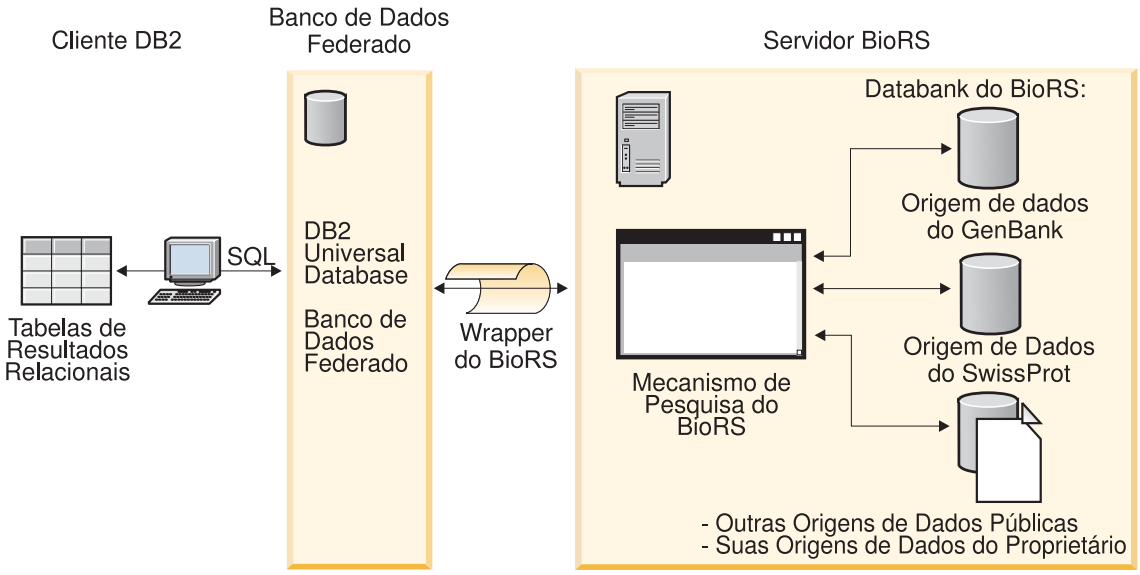


Figura 1. Como Funciona o Wrapper do BioRS

No cliente, os usuários ou aplicativos enviam uma consulta utilizando instruções SQL. Em seguida, a consulta é enviada para o sistema federado no qual o wrapper do BioRS está instalado. Dependendo de como a consulta é construída, pode-se utilizar o DB2® Universal Database e o servidor BioRS para processar a consulta. O servidor BioRS pode estar em um computador diferente do sistema federado. As informações de autenticação devem ser fornecidas pelo sistema federado ao servidor BioRS para cada consulta. Essas informações podem ser uma combinação de ID do usuário e senha ou uma indicação não-autenticada (geralmente uma conta guest).

O wrapper do BioRS funciona com o BioRS Versão 5.0.14.

Para obter informações detalhadas sobre o produto BioRS, consulte o Web site da Biomax em: http://www.biomax.com.

Tarefas Relacionadas:

- "Incluindo o BioRS em um Sistema Federado" na página 3

Referência Relacionada:

- "Wrapper do BioRS - Consultas de Exemplo" na página 20

Incluindo o BioRS em um Sistema Federado

Você pode utilizar uma origem de dados do BioRS com seu servidor federado, registrando funções personalizadas e um wrapper do BioRS. Em seguida, pode registrar um servidor BioRS correspondente, mapeamentos do usuário e pseudônimos para permitir que o servidor federado recupere e processe dados do BioRS.

Você pode executar as instruções SQL a partir do Centro de Controle do DB2 ou do processador de linha de comandos do DB2. Depois de incluir o BioRS no sistema federado, você pode executar consultas em uma origem de dados do BioRS.

Procedimento:

Para incluir uma origem de dados do BioRS em um servidor federado:

1. Registre funções personalizadas utilizando a instrução CREATE FUNCTION.
2. Registre o wrapper do BioRS utilizando a instrução CREATE WRAPPER.
3. Opcional: Defina a variável de ambiente DB2_DJ_COMM para melhorar o desempenho da consulta.
4. Registre o servidor BioRS utilizando a instrução CREATE SERVER.
5. Opcional: Registre os usuários autorizados com a instrução CREATE USER MAPPING.
6. Registre os pseudônimos utilizando a instrução CREATE NICKNAME.
7. Opcional: Atualize as estatísticas de cardinalidade para colunas BioRS.

Tarefas Relacionadas:

- "Registrando Funções Personalizadas para o Wrapper do BioRS" na página 4
- "Registrando o Wrapper do BioRS" na página 5
- "Definindo a Variável de Perfil DB2_DJ_COMM para o Wrapper do BioRS" na página 5
- "Registrando o Servidor para uma Origem de Dados do BioRS" na página 6
- "Registrando Mapeamentos de Usuários para Origens de Dados do BioRS" na página 7
- "Registrando Pseudônimos para Origens de Dados do BioRS" na página 8
- "Atualizando as Estatísticas de Cardinalidade de Colunas do BioRS" na página 12

Registrando Funções Personalizadas para o Wrapper do BioRS

O registro de funções personalizadas para o wrapper do BioRS faz parte da tarefa maior de inclusão do BioRS em um sistema federado. Após o registro de funções personalizadas, é necessário registrar o wrapper.

Você pode utilizar o arquivo de amostra create_function_mappings.ddl para registrar funções personalizadas. Esse arquivo está no diretório sqllib/samples/lifesci/biors. O arquivo create_function_mappings.ddl contém definições para cada função personalizada. Você pode executar esse arquivo DDL para registrar as funções personalizadas de cada banco de dados DB2 no qual o wrapper do BioRS está instalado.

Pré-requisitos:

- Todas as funções personalizadas do wrapper do BioRS devem ser registradas com o nome de esquema BioRS.
- Cada função personalizada deve ser registrada uma vez para cada banco de dados DB2 no qual o wrapper do BioRS está instalado.

Procedimento:

Para registrar funções personalizadas, emita a instrução CREATE FUNCTION com a palavra-chave AS TEMPLATE.

O nome completo de cada função é BioRS.<nome_da_função>.

O exemplo a seguir registra uma versão da função CONTAINS:

```
CREATE FUNCTION biors.contains (varchar(), varchar())
RETURNS INTEGER AS TEMPLATE;
```

A próxima tarefa nesta seqüência de tarefas é registrar o wrapper apropriado do BioRS.

Tarefas Relacionadas:

- "Registrando o Wrapper do BioRS" na página 5

Referência Relacionada:

- "CREATE FUNCTION (Sourced or Template) statement" no *SQL Reference, Volume 2*
- "Funções Personalizadas e Consultas do BioRS" na página 15
- "Wrapper do BioRS - Consultas de Exemplo" na página 20
- "Tabela de Funções Personalizadas - Wrapper do BioRS" na página 32

Registrando o Wrapper do BioRS

O registro do wrapper do BioRS faz parte da maior tarefa de inclusão do BioRS em um sistema federado. Você deve registrar o wrapper para acessar uma origem de dados. Wrappers são mecanismos que os servidores federados utilizam para comunicar-se com e recuperar dados de origens de dados. Os wrappers são instalados no sistema como arquivos de biblioteca. A Tabela 1 lista os arquivos de biblioteca padrão do BioRS e o sistema operacional suportado para cada arquivo.

Tabela 1. Arquivos de biblioteca do BioRS e sistemas operacionais suportados

Arquivo de Biblioteca do BioRS	Sistema Operacional
libdb2lsbiors.a	IBM AIX Versão 4.3.3 ou posterior
db2lsbiors.dll	Microsoft Windows NT Versão 4 Microsoft Windows 2000 Microsoft Windows XP

Procedimento:

Para registrar o wrapper do BioRS, emita a instrução CREATE WRAPPER.

Por exemplo, para registrar um wrapper do BioRS no AIX denominado `wrap_biors` a partir do arquivo de biblioteca padrão `libdb2lsbiors.a`, emita a seguinte instrução:

```
CREATE WRAPPER wrap_biors LIBRARY 'libdb2lsbiors.a';
```

Tarefas Relacionadas:

- "Verificando as Bibliotecas Não-relacionais do Wrapper e da UDF (User-Defined Function) de Biociências" no *DB2 Information Integrator Installation Guide*
- "Definindo a Variável de Perfil DB2_DJ_COMM para o Wrapper do BioRS" na página 5

Referência Relacionada:

- "CREATE WRAPPER statement" no *SQL Reference, Volume 2*

Definindo a Variável de Perfil DB2_DJ_COMM para o Wrapper do BioRS

A definição da variável de perfil DB2_DJ_COMM do DB2 para o wrapper do BioRS faz parte da tarefa maior de inclusão do BioRS em um sistema federado. Para melhorar o desempenho quando as origens de dados do BioRS são acessadas, você pode definir opcionalmente a variável de perfil DB2_DJ_COMM do DB2. Essa variável especifica se o servidor federado carregará o wrapper na inicialização.

O uso do processador aumenta quando o servidor federado carrega as bibliotecas do wrapper durante a inicialização do banco de dados. Para evitar o uso excessivo, especifique apenas as bibliotecas que você deseja acessar.

Procedimento:

Para definir a variável de perfil DB2_DJ_COMM do DB2, emita o comando **db2set** com a biblioteca de wrapper que corresponde ao wrapper especificado na instrução CREATE WRAPPER associada.

Por exemplo:

```
db2set DB2_DJ_COMM='libdb2lsbiors.a'
```

Assegure-se de que não existam espaços em nenhum lado do sinal de igual (=).

A próxima tarefa na seqüência de tarefas é registrar o servidor para o BioRS.

Conceitos Relacionados:

- "Environment Variables and the Profile Registry" no *Administration Guide: Implementation*

Tarefas Relacionadas:

- "Registrando o Servidor para uma Origem de Dados do BioRS" na página 6

Referência Relacionada:

- "db2set - DB2 Profile Registry Command" em *Command Reference*

Registrando o Servidor para uma Origem de Dados do BioRS

O registro do servidor para uma origem de dados do BioRS faz parte da maior tarefa de inclusão do BioRS em um sistema federado. Depois de registrar o wrapper, você deve registrar um servidor correspondente.

Procedimento:

Para registrar o servidor BioRS no sistema federado, emita a instrução CREATE SERVER.

Por exemplo:

```
CREATE SERVER brs_server WRAPPER wrap_biors OPTIONS(NODE 'biors_server2.com');
```

Tarefas Relacionadas:

- "Registrando Pseudônimos para Origens de Dados do BioRS" na página 8

Referência Relacionada:

- "Opções da Instrução CREATE SERVER - Wrapper do BioRS" na página 39

Registrando Mapeamentos de Usuários para Origens de Dados do BioRS

O registro de mapeamentos de usuários faz parte da tarefa maior de inclusão do BioRS em um sistema federado. Pode não ser necessário criar mapeamentos de usuários, dependendo do método ou dos métodos de acesso da conta que são utilizados no sistema BioRS.

- Se o servidor BioRS estiver configurado para acesso guest a todas as contas de usuários, não será necessário criar mapeamentos de usuários no DB2 Information Integrator.
- Se o servidor BioRS estiver configurado para autenticar contas de usuários com IDs e senhas, será necessário criar mapeamentos de usuários no sistema federado para as contas que deverão utilizar o wrapper do BioRS.
- Se o servidor BioRS estiver configurado para utilizar uma mistura de contas guest e autenticada do usuário, será necessário criar mapeamentos de usuários para as contas autenticadas do usuário no banco de dados federado que deverão utilizar o wrapper do BioRS.

Os mapeamentos de usuários fornecem uma maneira de autenticar o acesso de usuários ou aplicativos que consultam uma origem de dados do BioRS com o wrapper do BioRS. Se um usuário ou aplicativo enviar uma consulta SQL para um pseudônimo registrado do BioRS, e não houver mapeamentos de usuários para esse usuário ou aplicativo, o wrapper do BioRS utilizará um ID de usuário e uma senha padrão na tentativa de recuperar dados do servidor BioRS remoto. Se um databank que estiver sendo consultado exigir autenticação, uma mensagem de erro poderá ser retornada.

Para assegurar-se de que o ID do usuário e senha corretos tenham sido transmitidos para o servidor BioRS, crie mapeamentos de usuários no banco de dados federado para usuários que estejam autorizados a pesquisar origens de dados do BioRS. Quando você cria um mapeamento de usuários, a senha é armazenada em um formato criptografado em uma tabela do catálogo do sistema de banco de dados federado.

Procedimento:

Para registrar mapeamentos de usuários do BioRS, utilize a instrução CREATE USER MAPPING.

Por exemplo, a instrução CREATE USER MAPPING a seguir mapeia o usuário Charlie para o usuário Charlene no servidor Biors_Server1.

```
CREATE USER MAPPING FOR Charlie SERVER Biors_Server1
OPTIONS(REMOTE_AUTHID 'Charlene', REMOTE_PASSWORD 'Charlene_pw');
```

Você também pode definir seu próprio mapeamento de usuário. No exemplo a seguir, USER é uma palavra-chave que identifica o usuário atual, não um nome de usuário USER.

```
CREATE USER MAPPING FOR USER SERVER Biors_Server1
OPTIONS(REMOTE_AUTHID 'Yudong', REMOTE_PASSWORD 'Yudong_pw')
```

A próxima tarefa nesta seqüência de tarefas é registrar pseudônimos para o wrapper do BioRS.

Tarefas Relacionadas:

- "Registrando Pseudônimos para Origens de Dados do BioRS" na página 8

Referência Relacionada:

- "CREATE USER MAPPING statement" no *SQL Reference, Volume 2*
- "Opções da Instrução CREATE USER MAPPING - Wrapper do BioRS" na página 40

Registrando Pseudônimos para Origens de Dados do BioRS

O registro de pseudônimos para origens de dados do BioRS faz parte da tarefa maior de inclusão do BioRS em um sistema federado. Depois de registrar um servidor, você deve registrar um pseudônimo para cada origem de dados do BioRS a ser acessada. Ao referir-se a uma origem de dados do BioRS em uma consulta, você utiliza pseudônimos.

Importante: Depois que uma origem de dados tiver sido integrada ao sistema BioRS, ela é referida como *databank* no BioRS. Os databanks do BioRS equivalem aos pseudônimos em um sistema federado.

Pré-requisitos:

- Se um nome de databank do BioRS não estiver em conformidade com a sintaxe federada do DB2, você deverá utilizar a opção de pseudônimo REMOTE_OBJECT quando registrar o pseudônimo.
- Se um nome de elemento do BioRS não estiver em conformidade com a sintaxe federada do DB2, você deverá utilizar a opção de coluna ELEMENT_NAME quando registrar o pseudônimo.

Restrições:

Não utilize o elemento AllText do BioRS como a primeira coluna de um pseudônimo. Você pode utilizar o elemento AllText do BioRS em qualquer outra posição de coluna (por exemplo, como a segunda ou terceira coluna).

Procedimento:

Para registrar um pseudônimo do BioRS, utilize a instrução CREATE NICKNAME.

Um pseudônimo federado equivale diretamente a um databank do BioRS. Ao criar um pseudônimo federado, você define uma lista de colunas de pseudônimos. As colunas de pseudônimos especificados devem corresponder aos elementos de um formato específico de databank do BioRS. O BioRS define cinco tipos de dados possíveis para os elementos: Texto, Número, Data, Autor e Referência. Esses tipos de dados podem ser mapeados apenas para os tipos de dados CLOB, CHAR ou VARCHAR do sistema federado.

A maneira mais simples de registrar um pseudônimo para um databank do BioRS é fornecer ao pseudônimo o mesmo nome que o databank do BioRS. Por exemplo:

```
CREATE NICKNAME SwissProt
 (ID VARCHAR(32) OPTIONS (ELEMENT_NAME '_ID_'),
 ALLTEXT VARCHAR(128),
 ENTRYDATE VARCHAR (64))
FOR SERVER brs_server;
```

O databank base SwissProt do BioRS é o nome do pseudônimo.

A utilização dessa sintaxe simples CREATE NICKNAME o limita a uma família de pseudônimos por esquema DB2. Você pode utilizar outros nomes especificando a opção REMOTE_OBJECT. Essa opção de pseudônimo especifica o nome do tipo de objeto BioRS a ser associado ao pseudônimo. O nome que é especificado na opção REMOTE_OBJECT determina o esquema e o databank do BioRS para o pseudônimo. A opção REMOTE_OBJECT também especifica o relacionamento do pseudônimo com outros pseudônimos.

O exemplo a seguir mostra o mesmo conjunto de características do pseudônimo que o exemplo anterior, mas altera o nome do pseudônimo e utiliza a opção REMOTE_OBJECT para especificar o databank do BioRS para o qual o pseudônimo está sendo definido:

```
CREATE NICKNAME NewSP
 (ID VARCHAR(32) OPTIONS (ELEMENT_NAME '_ID_'),
 ALLTEXT VARCHAR(128),
 ENTRYDATE VARCHAR (64))
FOR SERVER brs_server
 OPTIONS (REMOTE_OBJECT 'SwissProt');
```

O databank base do BioRS é o SwissProt e o nome do pseudônimo é NewSP.

Conceitos Relacionados:

• “Informações de Estatísticas do BioRS” na página 27

Tarefas Relacionadas:

- "Atualizando as Estatísticas de Cardinalidade de Pseudônimos do BioRS" na página 28

Referência Relacionada:

- "O Elemento AllText do BioRS" na página 30
- "Instrução CREATE NICKNAME - Exemplos para o Wrapper do BioRS" na página 10
- "Sintaxe da Instrução CREATE NICKNAME - Wrapper do BioRS" na página 37
- "Considerações sobre Alteração de Pseudônimos - Wrapper do BioRS" na página 31

Instrução CREATE NICKNAME - Exemplos para o Wrapper do BioRS

Este tópico fornece exemplos que mostram como utilizar a instrução CREATE NICKNAME para registrar pseudônimos para o wrapper do BioRS.

Exemplo 1:

O exemplo a seguir mostra como criar um pseudônimo para um databank remoto do BioRS que não esteja em conformidade com a sintaxe do DB2 Information Integrator:

```
CREATE NICKNAME SwissFT
 (ID VARCHAR(32) OPTIONS (ELEMENT_NAME '_ID_'),
 ALLTEXT VARCHAR (128),
 ENTRYDATE VARCHAR (64),
 FtLength VARCHAR (16),
   FOR SERVER biors1
 OPTIONS (REMOTE_OBJECT 'SwissProt.Features');
```

O nome desse pseudônimo é SwissFT. As colunas da tabela são ID, ALLTEXT, ENTRYDATE e FtLength. A opção de coluna ELEMENT_NAME é especificada para a coluna ID. Você deve especificar a opção ELEMENT_NAME quando o nome de um elemento do BioRS não está em conformidade com a sintaxe federada do DB2 para nomes de colunas. Nesse exemplo, o elemento _ID_ do BioRS está em conformidade com a sintaxe federada do DB2, mas `_ID_` é um nome potencialmente confuso para usuários do DB2 Information Integrator. O nome `ID` é simples e fácil de entender. Em geral, utilize a opção ELEMENT_NAME nas seguintes circunstâncias:

- Quando um nome de elemento do BioRS não está em conformidade com a sintaxe federada válida do DB2
- Quando a distinção entre maiúsculas e minúsculas de um nome de elemento do BioRS não está em conformidade com os padrões estabelecidos para sistemas federados DB2

- Quando o nome de um elemento do BioRS pode não ser óbvio para usuários do DB2 Information Integrator

Além disso, a opção REMOTE_OBJECT é utilizada para especificar o nome do databank do BioRS ao qual o pseudônimo equivale. Você deve especificar a opção REMOTE_OBJECT quando o nome de um databank do BioRS não está em conformidade com a sintaxe federada válida do DB2. Nesse exemplo, o nome do databank ″SwissProt.Features″ não está em conformidade com a sintaxe federada válida do DB2. Em geral, utilize a opção REMOTE_OBJECT nas seguintes circunstâncias:

- Quando a distinção entre maiúsculas e minúsculas de nomes de databanks do BioRS não está em conformidade com os padrões estabelecidos para sistemas federados DB2
- Quando o nome do databank do BioRS não está em conformidade com a sintaxe federada válida do DB2
- Quando o nome de um databank do BioRS pode não ser óbvio para usuários do DB2 Information Integrator

Exemplo 2:

O exemplo a seguir mostra como criar um pseudônimo para uma tabela que utiliza um databank do BioRS que está vinculado a outro databank do BioRS:

```
CREATE NICKNAME SwissFT2
 (ID VARCHAR(32) OPTIONS (ELEMENT_NAME '_ID_'),
 ALLTEXT VARCHAR (1200),
 FtKey VARCHAR (32),
 FtLength VARCHAR (64),
   FtDescription VARCHAR (128),
   Parent VARCHAR (32) OPTIONS (REFERENCED_OBJECT 'SwissProt'))
 FOR SERVER biors1
 OPTIONS (REMOTE_OBJECT 'SwissProt.Features');
```

O nome desse pseudônimo é SwissFT2. As colunas da tabela são ID, ALLTEXT, FtKey, FtLength, FtDescription e Parent. A opção de coluna ELEMENT_NAME é especificada para a coluna ID. A opção REMOTE_OBJECT é utilizada para especificar o nome do databank do BioRS ao qual o pseudônimo corresponde.

Além disso, a coluna Parent utiliza a opção REFERENCED_OBJECT. Você deve especificar essa opção para colunas que correspondem aos elementos do tipo de dados Referência do BioRS. A opção REFERENCED_OBJECT especifica o nome do databank do BioRS ao qual a coluna se refere. Nesse caso, o elemento Parent refere-se ao databank SwissProt do BioRS.

Tarefas Relacionadas:

- “Registrando Pseudônimos para Origens de Dados do BioRS” na página 8

Referência Relacionada:

- "Sintaxe da Instrução CREATE NICKNAME - Wrapper do BioRS" na página 37
- "Considerações sobre Alteração de Pseudônimos - Wrapper do BioRS" na página 31

Atualizando as Estatísticas de Cardinalidade de Colunas do BioRS

Para atualizar as estatísticas de cardinalidade de colunas do BioRS no sistema federado, você deve modificar a exibição do catálogo SYSSTAT.COLUMNS.

Manter as estatísticas corretas de cardinalidade para colunas do BioRS permite que o otimizador e o wrapper do BioRS escolham a melhor execução do plano de acesso a dados durante o processamento da consulta.

Opcionalmente, você pode atualizar as estatísticas de cardinalidade de colunas do BioRS como parte da tarefa maior de inclusão do BioRS em um sistema federado. Você também pode atualizar as estatísticas de cardinalidade de colunas do BioRS quando desejar melhorar o desempenho da consulta para origens de dados do BioRS.

Restrições:

Não utilize esse procedimento para atualizar as estatísticas de cardinalidade de colunas que correspondem ao elemento _ID_ do BioRS. Você deve utilizar um procedimento diferente para atualizar as estatísticas de cardinalidade de colunas que correspondem ao elemento _ID_ do BioRS.

Procedimento:

Para atualizar as estatísticas de cardinalidade de colunas do BioRS, emita uma instrução UPDATE utilizando a seguinte sintaxe:

```
UPDATE sysstat.columns SET colcard=(SELECT COUNT(DISTINCT <column-name>)
                                      FROM <nickname-schema>.<nickname-name>)
WHERE
  tabschema=<nickname-schema>
  AND tabname=<nickname-name>
  AND colname=<column-name>
```

- *<column-name>* é o nome da coluna cujas estatísticas de cardinalidade você deseja atualizar.
- *<nickname-schema>* é o nome do esquema que está associado ao pseudônimo no qual a coluna especificada é utilizada.
- *<nickname-name>* é o nome do pseudônimo no qual a coluna especificada é utilizada.

A execução da consulta pode levar alguns minutos, porque todas as entradas do databank especificado no pseudônimo devem ser recuperadas.

Se uma coluna puder conter vários valores (por exemplo, o elemento PublicationYear do formato do banco de dados SwissProt), o cálculo se tornará muito complexo para utilizar uma consulta SQL. Para essas colunas, você deve calcular manualmente o valor de cardinalidade e, em seguida, atualizar a exibição do catálogo SYSSTAT.COLUMNS. Para calcular o valor de cardinalidade, divida o número de valores distintos na coluna pelo número médio de valores por linha. O valor calculado da cardinalidade não pode ser maior que a cardinalidade da tabela.

Exemplo:

Suponha que você possua um pseudônimo com três linhas. Os valores da coluna PublicationYear para essas três linhas são:

- 1997 1992 1985
- 1997 1992 1982
- 1992 1991 1990 1976 1974 1971

Há nove valores distintos e o número médio de valores em uma linha é quatro. A cardinalidade dessa coluna PublicationYear é 9/4 ou 3 (2,25 arredondado para o próximo inteiro mais alto). Agora que você possui o cálculo de cardinalidade, é possível atualizar a exibição do catálogo SYSSTAT.COLUMNS utilizando a seguinte instrução UPDATE:

```
UPDATE sysstat.columns SET colcard=3
WHERE
  tabschema=<nickname-schema>
  AND tabname=<nickname-name>
  AND colname=<column-name>
```

- *3* é o valor de cardinalidade da coluna.
- <*nickname-schema*> é o nome do esquema que está associado ao pseudônimo base no qual a coluna especificada é utilizada.
- <*nickname-name*> é o nome do pseudônimo base no qual a coluna especificada é utilizada.
- <*column-name*> é o nome da coluna cujas estatísticas de cardinalidade você deseja atualizar.

Conceitos Relacionados:

- "Informações de Estatísticas do BioRS" na página 27

Tarefas Relacionadas:

- "Atualizando as Estatísticas de Cardinalidade de Pseudônimos do BioRS" na página 28

Diretrizes para Otimizar o Desempenho do Wrapper do BioRS

Este tópico fornece diretrizes sobre como otimizar o desempenho de consultas ao utilizar o wrapper do BioRS.

Minimize a quantidade de dados que é transferida entre os mecanismos de pesquisa.

O ambiente federado utiliza dois mecanismos de consulta. Para o wrapper do BioRS, esses mecanismos de consulta são DB2® Universal Database e BioRS. O mecanismo do DB2 processa predicados (operadores relacionais, como =, BETWEEN, LIKE e <>) especificados em colunas de pseudônimos. O mecanismo do BioRS processa os predicados especificados utilizando as quatro funções personalizadas para o wrapper do BioRS.

Para minimizar a quantidade de dados que é transferida entre os dois mecanismos de pesquisa, estruture suas consultas de modo que o processamento de dados seja enviado para o sistema BioRS sempre que possível.

Se você precisar executar operações de junção em uma consulta, aproveite os relacionamentos pai-filho que já existem nos databanks do BioRS e execute operações equijoin sempre que possível. As operações equijoin são processadas no BioRS, o que também minimiza a quantidade de dados transferidos entre os mecanismos de consulta do DB2 e do BioRS.

Atenção: Não interrompa as consultas do DB2 Information Integrator no BioRS (por exemplo, utilizando **Ctrl-D** ou **Ctrl-Z** no processador de linha de comandos ou parando um programa aplicativo). A interrupção de uma consulta deixa processos ″inativos″ em execução no servidor BioRS. Esses processos ″inativos″ degradarão rapidamente o desempenho dos sistemas BioRS e DB2 Information Integrator. Se um número suficiente desses processos ″inativos″ estiver em execução, poderão ocorrer erros inesperados durante o processamento de consulta do DB2 Information Integrator. Por exemplo, uma consulta válida poderia retornar 0 linhas, quando linhas são esperadas. Em situações extremas, o BioRS ou DB2 Information Integrator, ou ambos os produtos, podem parar ou finalizar anormalmente.

Mantenha as informações de estatísticas do BioRS no ambiente federado.

Em um sistema federado, o banco de dados federado conta com estatísticas do catálogo para objetos com pseudônimo para otimizar o processamento da consulta. Manter estatísticas atuais sobre as origens de dados do BioRS é essencial para otimizar o desempenho do wrapper do BioRS. Se os dados estatísticos ou características

estruturais de um objeto remoto no qual um pseudônimo está definido foram alterados, você deverá atualizar as estatísticas correspondentes de cardinalidade da coluna do pseudônimo no sistema federado.

Para otimizar o desempenho do wrapper do BioRS, execute essas atualizações no DB2 Information Integrator em intervalos regulares.

Conceitos Relacionados:

- "Tuning query processing" no *Federated Systems Guide*
- "Predicados Equijoin para o Wrapper do BioRS" na página 18
- "Informações de Estatísticas do BioRS" na página 27

Referência Relacionada:

- "Funções Personalizadas e Consultas do BioRS" na página 15
- "Wrapper do BioRS - Consultas de Exemplo" na página 20

Funções Personalizadas e Consultas do BioRS

O ambiente federado utiliza dois mecanismos de consulta. Para o wrapper do BioRS, esses mecanismos de consulta são o DB2 Universal Database e o BioRS. Você pode especificar que os predicados sejam enviados para o mecanismo do BioRS utilizando as quatro funções personalizadas do BioRS, que são:

- BIORS.CONTAINS
- BIORS.CONTAINS_LE
- BIORS.CONTAINS_GE
- BIORS.SEARCH_TERM

Essas quatro funções personalizadas estão registradas no esquema BioRS. Você deve utilizar o esquema BioRS para referir-se às funções.

As funções personalizadas BIORS.CONTAINS, BIORS.CONTAINS_LE e BIORS.CONTAINS_GE requerem um argumento de coluna do termo de pesquisa e um argumento de texto de consulta. O exemplo a seguir mostra uma instrução BIORS.CONTAINS:

```
BIORS.CONTAINS (<coluna do termo de pesquisa>,<termo de consulta>)
```

O valor do argumento de coluna do termo de pesquisa deve referir-se a uma coluna indexada do BioRS. A utilização de uma coluna não indexada produz a mensagem de erro SQL30090N ("Operação inválida para o ambiente de execução do aplicativo").

O valor do argumento de termo de consulta pode ser apenas um literal, uma variável do host ou uma referência de coluna. Não é possível utilizar a

concatenação aritmética ou de cadeia. Além disso, o valor do argumento de termo de consulta não pode ser NULL, mesmo se a coluna do termo de pesquisa utilizada estiver definida como permitindo valores nulos.

A utilização de maiúsculas ou minúsculas no argumento de termo da consulta não é importante.

Os tipos e formatos válidos de dados do argumento do termo da consulta dependem do tipo de dados BioRS da coluna do termo da pesquisa que é utilizado. O BioRS define cinco tipos de dados possíveis: Texto, Autor, Data, Número e Referência. Os tipos de dados BioRS e os termos válidos de consulta da função para cada tipo de dados são listados na Tabela 2.

Tabela 2. Tipos de dados BioRS e termos válidos da consulta personalizada da função

Tipo de Dados da Coluna do Termo de Pesquisa	**Termo Válido da Consulta**	**Formato**
Texto	VARCHAR() ou CHAR()	Termo de texto do BioRS, incluindo caracteres curinga.
Autor	VARCHAR() ou CHAR()	Referência de autor do BioRS na forma "<last>, <init>". "<last>" é o sobrenome do autor. "<init>" são as iniciais do autor, sem pontos. O espaço em branco entre a vírgula e as iniciais é permitido. Alternativamente, <last> pode ser especificado sozinho, sem a vírgula ou as iniciais.
Data	VARCHAR(), CHAR(), DATE ou TIMESTAMP	Se uma cadeia de caracteres, data no formato do DB2, aaaa/mm/dd.
Número	VARCHAR() ou CHAR(), INTEGER, SMALLINT, BIGINT REAL, DOUBLE, DECIMAL	Números no formato do DB2.
Referencia	VARCHAR() ou CHAR()	Termo de texto do BioRS.

Todas as outras combinações dos argumentos da coluna do termo de pesquisa e do termo da consulta para os tipos de dados BioRS produzem a mensagem de erro SQL30090N ("Operação inválida para o ambiente de execução do aplicativo"). Você só pode utilizar as combinações mostradas na Tabela 2.

O argumento do termo da consulta para as colunas do termo da pesquisa dos tipos de dados Texto, Autor e Referência devem corresponder a um padrão de

linguagem de consulta do BioRS. No BioRS, os argumentos de termo de consulta podem consistir em cadeias alfanuméricas e caracteres curinga. A função BIORS.CONTAINS suporta dois caracteres curinga: **?** (ponto de interrogação) e * (asterisco).

O caractere curinga **?** corresponde a um único caractere. Por exemplo, o predicado `BioRS.CONTAINS (description, 'bacteri?')=1` corresponde ao termo bacteria mas não ao termo bacterial.

O caractere curinga * corresponde a zero ou mais caracteres. Por exemplo, o predicado `BioRS.CONTAINS (description, 'bacteri*')=1` corresponde aos termos bacteri, bacteria e bacterial.

Para obter informações detalhadas sobre os padrões de linguagem de consulta do BioRS, consulte a documentação do BioRS.

A função BIORS.CONTAINS pode ser especificada para todos os tipos de colunas do BioRS.

As funções personalizadas BIORS.CONTAINS_GE e BIORS.CONTAINS_LE só podem ser especificadas para as colunas cujo tipo de dados base do BioRS seja Número ou Data. A função BIORS.CONTAINS_GE seleciona as linhas nas quais a coluna contém um valor que seja maior que ou igual ao valor que é representado pelo argumento do termo da consulta. A função BIORS.CONTAINS_LE seleciona as linhas nas quais a coluna contém um valor que seja menor que ou igual ao valor que é representado pelo argumento do termo da consulta.

As funções BIORS.CONTAINS, BIORS.CONTAINS_GE e BIORS.CONTAINS_LE retornam um resultado inteiro. Quando uma das três funções CONTAINS é utilizada em um predicado, o valor de retorno deve ser comparado com o valor 1 utilizando os operadores **=** ou **<>**. Por exemplo:

```
SELECT * FROM s.MySP WHERE BIORS.CONTAINS (s.AllText, 'muscus') = 1;
```

Uma expressão na forma `NOT (BioRS.Contains (col,value) = 1)` é equivalente a `BioRS.CONTAINS (col,value) <> 1`.

Você pode executar consultas que, de outra maneira, não seriam possíveis emitindo a função BIORS.SEARCH_TERM. Você pode utilizar essa função para especificar um termo de pesquisa utilizando o formato do BioRS. A função BIORS.SEARCH_TERM requer dois argumentos. O primeiro argumento é uma referência à coluna _ID_ do pseudônimo para o qual o termo será aplicado. O segundo argumento é uma cadeia de caracteres que contém o termo sem um nome de databank.

O exemplo a seguir seleciona todas as colunas das entradas no databank MyEMBL em que o elemento SeqLength contêm um valor maior que ou igual a 100.

```
SELECT * FROM MyEMBL s WHERE
 BIORS.SEARCH_TERM (s.ID, '[SeqLength GREATER number:100;]') = 1;
```

O exemplo a seguir seleciona a coluna MolWeight a partir do pseudônimo Swiss, em que o valor do elemento MolWeight é maior que ou igual a 100368.

```
SELECT s.molweight FROM Swiss s WHERE
   BIORS.SEARCH_TERM (s.ID, '[MolWeight GREATER number:100368;]') = 1;
```

Se você especificar a função BIORS.SEARCH_TERM, não poderá utilizar nenhuma outra função personalizada em uma consulta. No entanto, você pode utilizar qualquer combinação das funções BIORS.CONTAINS, BIORS.CONTAINS_GE e BIORS.CONTAINS_LE na mesma consulta.

Conceitos Relacionados:

- "Pushdown analysis" no *Federated Systems Guide*
- "Diretrizes para Otimizar o Desempenho do Wrapper do BioRS" na página 14
- "Predicados Equijoin para o Wrapper do BioRS" na página 18

Tarefas Relacionadas:

- "Registrando Funções Personalizadas para o Wrapper do BioRS" na página 4

Referência Relacionada:

- "Wrapper do BioRS - Consultas de Exemplo" na página 20
- "Tabela de Funções Personalizadas - Wrapper do BioRS" na página 32

Predicados Equijoin para o Wrapper do BioRS

Você pode especificar predicados para o mecanismo do BioRS utilizando as quatro funções personalizadas do BioRS, com uma exceção. A exceção é quando você executa operações equijoin durante uma consulta. Uma operação *join* envolve a recuperação de dados de duas ou mais tabelas com base nos valores de colunas correspondentes. Um *equijoin* é uma operação de junção em que a condição de junção possui o formato expressão = expressão. Para consultas do BioRS, os termos do equijoin devem conter o elemento _ID_ de um databank e o elemento de tipo Referência de outro databank.

Exemplo:

Esse exemplo mostra definições de pseudônimos de amostra e uma consulta equijoin que utiliza os pseudônimos de amostra.

Suponha que você deseje consultar dois databanks do BioRS, SwissProt e SwissProt.features. O databank SwissProt.features é um filho do databank SwissProt e contém um elemento denominado Parent. O elemento Parent contém referências a entradas que são identificadas pelo elemento _ID_ do SwissProt. Você registra duas definições de pseudônimos para os dois databanks.

Definição de pseudônimo 1:

```
CREATE NICKNAME tc600sprot (
   ID             VARCHAR (32) OPTIONS (ELEMENT_NAME '_ID_'),
   AllText        VARCHAR (128),
   EntryDate      VARCHAR (128),
   Update         VARCHAR (128),
   Description    VARCHAR (1200),
   Crossreference VARCHAR (32),
   Authors        VARCHAR (256),
   Journal        VARCHAR (256),
   JournalIssue   VARCHAR (64) OPTIONS (IS_INDEXED 'N'),
   PublicationYear VARCHAR (1024),
   Gene            VARCHAR (20) OPTIONS (IS_INDEXED 'Y'),
   Remarks         VARCHAR (1200),
   RemarkType      CHAR (20),
   CatalyticActivity VARCHAR (20),
   CoFactor        VARCHAR (64),
   Disease         VARCHAR (128),
   Function        VARCHAR (128),
   Pathway         VARCHAR (128),
   Similarity      VARCHAR (128),
   Complex         VARCHAR (64),
   FtKey           VARCHAR (32),
   FtDescription   VARCHAR (128),
   FtLength        VARCHAR (256),
   MolWeight       VARCHAR (64),
   ProteinLen      VARCHAR (32) OPTIONS (ELEMENT_NAME 'Protein_length'),
   Sequence        CLOB,
   AccNumber       VARCHAR (32),
   Taxonomy        VARCHAR (128),
   Organelle       VARCHAR (128),
   Organism        VARCHAR (128),
   Keywords        VARCHAR (1200),
   Localization    VARCHAR (128),
   FtKey_count     VARCHAR (32)) FOR SERVER biors_server_600
      OPTIONS (REMOTE_OBJECT 'SwissProt');
```

Definição de pseudônimo 2:

```
CREATE NICKNAME tc600feat (
  ID     VARCHAR (32) OPTIONS (ELEMENT_NAME '_ID_'),
  AllText VARCHAR (1200),
  FtKey   VARCHAR (32),
```

```
  FtLength VARCHAR (64),
  FtDescription VARCHAR (128),
  Parent        VARCHAR (32) OPTIONS (REFERENCED_OBJECT 'SwissProt'))
    FOR SERVER biors_server_600 OPTIONS (REMOTE_OBJECT 'SwissProt.features');
```

A consulta a seguir referencia ambos os pseudônimos em um equijoin:

```
SELECT s.ID, f.ID, f.FtKey FROM tc600sprot s, tc600feat f
   WHERE BioRS.CONTAINS (s.AllText, 'anopheles') = 1
    AND BioRS.CONTAINS (s.PublicationYear, 1997) = 1
 AND BioRS.CONTAINS (f.FtKey, 'signal') = 1
 AND f.Parent = s.ID;
```

Na consulta anterior, dois predicados são aplicados ao pseudônimo tc600sprot (databank SwissProt). Esses dois predicados filtram as linhas que contêm o termo anopheles e possuem um ano de publicação de 1997. Um predicado é aplicado ao pseudônimo tc600feat (databank SwissProt.features), que filtra as linhas cujo elemento FtKey contém o termo signal. Os dois pseudônimos são juntados utilizando o termo f.Parent = s.ID.

O conjunto final de resultados contém apenas as linhas que atendem a esses critérios e onde as entradas de recursos referenciam uma entrada correspondente no databank SwissProt.

Conceitos Relacionados:

- "Diretrizes para Otimizar o Desempenho do Wrapper do BioRS" na página 14

Referência Relacionada:

- "Funções Personalizadas e Consultas do BioRS" na página 15
- "Wrapper do BioRS - Consultas de Exemplo" na página 20

Wrapper do BioRS - Consultas de Exemplo

Este tópico fornece várias consultas de amostra que utilizam os pseudônimos swiss e swissft.

O pseudônimo swiss foi registrado com a seguinte instrução CREATE NICKNAME:

```
CREATE NICKNAME swiss
  (
  ID                CHAR (30) OPTIONS (ELEMENT_NAME '_ID_'),
  EntryDate         VARCHAR (15),
  Update            CLOB (15),
  Description       CLOB (15),
  Crossreference    CLOB (15),
  Authors           CLOB (15),
  Journal           VARCHAR (15),
  JournalIssue      VARCHAR (15),
```

```
  PublicationYear    CLOB (15),
  PublicationTitle   CLOB (15),
  Gene               CLOB (15),
  Remarks            CLOB (15),
  RemarkType         VARCHAR (15),
  CatalyticActivity  VARCHAR (15),
  CoFactor           VARCHAR (15),
  Disease            VARCHAR (15),
  Function           CLOB (15),
  Pathway            VARCHAR (15),
  Similarity         CLOB (15),
  Complex            VARCHAR (15),
  FtKey              VARCHAR (15),
  FtDescription      CLOB (15),
  FtLength           VARCHAR (15),
  MolWeight          CHAR (15),
  Protein_Length     VARCHAR (15),
  Sequence           CLOB (15),
  AccNumber          VARCHAR (15),
  Taxonomy           CLOB (15),
  Organelle          VARCHAR (15),
  Organism           VARCHAR (15),
  Keywords           VARCHAR (15),
  Localization       VARCHAR (15),
  FtKey_count        VARCHAR (15),
  AllText            CLOB (15)
  )
    FOR SERVER biors_server
      OPTIONS (REMOTE_OBJECT 'swissprot');
```

O pseudônimo swissft foi registrado com a seguinte instrução CREATE NICKNAME:

```
CREATE NICKNAME swissft
  (
  ID                 VARCHAR (30) OPTIONS (ELEMENT_NAME '_ID_'),
  FtKey              VARCHAR (15),
  FtLength           VARCHAR (15),
  FtDescription      VARCHAR (15),
  Parent             VARCHAR (30) OPTIONS (REFERENCED_OBJECT 'swissprot'),
  AllText            CLOB (15)
  )
    FOR SERVER biors_server
      OPTIONS (REMOTE_OBJECT 'swissprot.features');
```

As consultas e os resultados na Tabela 3 na página 22 ilustram como você pode estruturar as consultas para otimizar a carga de trabalho entre o sistema federado e o servidor BioRS.

Tabela 3. Amostras de diferentes consultas que produzem resultados idênticos

Consulta	Resultado
select s.id from Swiss s where biors.CONTAINS(s.id, '100K_RAT') = 1 busca das 3 primeiras linhas apenas	ID --------------- 100K_RAT 1 reg.(s) selecionado(s).
select s.id from Swiss s where s.id LIKE '%100K_RAT%' busca das 3 primeiras linhas apenas	ID --------------- 100K_RAT 1 reg.(s) selecionado(s).

Ambas as consultas na Tabela 3 produzem os mesmos resultados. No entanto, a execução da primeira consulta será muito mais rápida do que da segunda consulta. A primeira consulta utiliza a função BIORS.CONTAINS para especificar o predicado de entrada. Como resultado, o BioRS seleciona os dados no databank swissprot, em seguida transmite os dados selecionados para o DB2 Information Integrator. Na segunda consulta, o predicado de entrada LIKE é especificado diretamente no pseudônimo Swiss. Como resultado, o BioRS transfere o databank swissprot inteiro para o DB2 Information Integrator. Depois que o conteúdo do databank é transferido, o DB2 Information Integrator seleciona os dados.

As consultas e os resultados na Tabela 4 mostram a utilização dos caracteres curinga na função BIORS.CONTAINS. Todos os resultados da consulta na Tabela 4 são idênticos, mesmo que diferentes caracteres curinga sejam utilizados.

Tabela 4. Consultas de amostra que utilizam caracteres curinga na função BIORS.CONTAINS

Consulta	Resultado
select s.crossreference from Swiss s where biors.CONTAINS(s.crossreference, 'MEDLINE') = 1 busca das 3 primeiras linhas apenas	CROSSREFERENCE --------------- NCBI_TaxID=1011 NCBI_TaxID=5875 NCBI_TaxID=4081 3 reg.(s) selecionado(s).
select s.crossreference from Swiss s where biors.CONTAINS(s.crossreference, '?ED?IN?') = 1 busca das 3 primeiras linhas apenas	CROSSREFERENCE --------------- NCBI_TaxID=1011 NCBI_TaxID=5875 NCBI_TaxID=4081 3 reg.(s) selecionado(s).

Tabela 4. Consultas de amostra que utilizam caracteres curinga na função BIORS.CONTAINS (continuação)

Consulta	Resultado
select s.crossreference from Swiss s where biors.CONTAINS(s.crossreference, '*D*N*') = 1 busca das 3 primeiras linhas apenas	CROSSREFERENCE --------------- NCBI_TaxID=1011 NCBI_TaxID=5875 NCBI_TaxID=4081 3 reg.(s) selecionado(s)

As consultas e os resultados na Tabela 5 mostram como você pode acessar informações nos elementos do tipo de dados Autor do BioRS com a função BIORS.CONTAINS.

A sintaxe de todas as consultas na Tabela 5 é quase idêntica. A única diferença é a presença ou falta da primeira inicial no termo de consulta e a quantidade de espaço entre o primeiro nome e a última inicial.

Tabela 5. Consultas de amostra que acessam as colunas do tipo de dados Autor do BioRS

Consulta	Resultado
select s.authors from Swiss s where biors.CONTAINS(s.authors, 'Mueller') = 1 busca das 3 primeiras linhas apenas	AUTHORS --------------- Mueller D. Rehb Mayer K.F.X. Sc Zemmour J. Litt 3 reg.(s) selecionado(s).
select s.authors from Swiss s where biors.CONTAINS(s.authors, 'Mueller,D') = 1 busca das 3 primeiras linhas apenas	AUTHORS --------------- 0 reg.(s) selecionado(s).
select s.authors from Swiss s where biors.CONTAINS(s.authors, 'Mueller ,D') = 1 busca das 3 primeiras linhas apenas	AUTHORS --------------- 0 reg.(s) selecionado(s).
select s.authors from Swiss s where biors.CONTAINS(s.authors, 'Mueller, D') = 1 busca das 3 primeiras linhas apenas	AUTHORS --------------- Mueller D. Rehb Zou P.J. Borovo Davies J.D. Mue 3 reg.(s) selecionado(s).

As consultas e os resultados na Tabela 6 ilustram como você pode acessar informações nos elementos do tipo Data do BioRS com a função BIORS.CONTAINS.

Quando um campo do tipo Data do BioRS contém uma seqüência de datas, os resultados podem conter informações extras, conforme mostrado no segundo exemplo da Tabela 6. Os elementos do tipo de dados Numérico (Data e Número) do BioRS podem conter múltiplos valores. Portanto, os resultados das consultas executadas nos elementos Data ou Número do BioRS também podem conter múltiplos valores. Múltiplos valores são sempre separados por espaços.

Tabela 6. Consultas de amostra que acessam as colunas do tipo de dados Data do BioRS

Consulta	Resultado
select e.entrydate from embl e where biors.CONTAINS(e.entrydate, date('11/01/1997')) = 1 busca das 3 primeiras linhas apenas	ENTRYDATE --------------- 01-NOV-1997 01-NOV-1997 01-NOV-1997 3 reg.(s) selecionado(s).
select g.update from gen g where biors.CONTAINS(g.update, date('11/01/1997')) = 1 busca das 3 primeiras linhas apenas	UPDATE--------------- 01-NOV-1997 11- 01-NOV-1997 12- 01-NOV-1997 06- 3 reg.(s) selecionado(s).

As consultas e os resultados na Tabela 7 mostram como você pode utilizar as funções BIORS.CONTAINS_LE e BIORS.CONTAINS_GE.

Tabela 7. Consultas de amostra que utilizam as funções BIORS.CONTAINS_LE e BIORS.CONTAINS_GE

Consulta	Resultado
select s.molweight from Swiss s where biors.CONTAINS_LE(s.molweight, 100368) = 1 busca das 3 primeiras linhas apenas	MOLWEIGHT --------------- 100368 10576 8523 3 reg.(s) selecionado(s).

Tabela 7. Consultas de amostra que utilizam as funções BIORS.CONTAINS_LE e BIORS.CONTAINS_GE (continuação)

Consulta	Resultado
select s.molweight from Swiss s where biors.CONTAINS_GE(s.molweight, 100368) = 1 busca das 3 primeiras linhas apenas	MOLWEIGHT --------------- 100368 103625 132801 3 reg.(s) selecionado(s).
select s.journalissue from Swiss s where biors.CONTAINS_GE(s.journalissue, 172) = 1 busca das 3 primeiras linhas apenas	JOURNALISSUE --------------- 172 21 242 196 3 reg.(s) selecionado(s).

As consultas e os resultados na Tabela 8 mostram como você pode utilizar a função BIORS.SEARCH_TERM para especificar um termo de pesquisa que utiliza o formato BioRS.

Tabela 8. Consultas de amostra que utilizam a função BIORS.SEARCH_TERM

Consulta	Resultado
select s.publicationyear from Swiss s where biors.SEARCH_TERM (s.id, '[PublicationYear EQ number:1997;]')=1 busca das 10 primeiras linhas apenas	PUBLICATIONYEAR --------------- 1997 1997 2000 1988 1991 1997 1994 1997 1997 1998 1994 1995 1997 1997 1999 1997 1994 1994 1995 1993 1992 1997 10 reg.(s) selecionado(s).
select s.molweight from Swiss s where biors.SEARCH_TERM (s.id, '[MolWeight EQ number:100368;]') = 1 busca das 10 primeiras linhas apenas	MOLWEIGHT --------------- 100368 100368 2 reg.(s) selecionado(s).

Tabela 8. Consultas de amostra que utilizam a função BIORS.SEARCH_TERM (continuação)

Consulta	Resultado
select s.molweight from Swiss s where biors.SEARCH_TERM (s.id, '[MolWeight GREATER number:100368;]') = 1 busca das 10 primeiras linhas apenas	MOLWEIGHT --------------- 100368 103625 132801 194328 130277 287022 289130 135502 112715 112599 10 reg.(s) selecionado(s).

A consulta a seguir mostra como utilizar predicados relacionados para formar um equijoin entre dois databanks que possuem um relacionamento pai-filho:

```
select s.id, f.id, f.parent from Swiss s, Swissft f
em que (f.parent = s.id) busca as 10 primeiras linhas apenas
```

Os resultados da consulta são os seguintes:

```
ID                   ID                 PARENT
-------------------- ------------------ ------------------------------
100K_RAT             100K_RAT.1         swissprot:100K_RAT
100K_RAT             100K_RAT.2         swissprot:100K_RAT
100K_RAT             100K_RAT.3         swissprot:100K_RAT
100K_RAT             100K_RAT.4         swissprot:100K_RAT
100K_RAT             100K_RAT.5         swissprot:100K_RAT
100K_RAT             100K_RAT.6         swissprot:100K_RAT
100K_RAT             100K_RAT.7         swissprot:100K_RAT
100K_RAT             100K_RAT.8         swissprot:100K_RAT
100K_RAT             100K_RAT.9         swissprot:100K_RAT
104K_THEPA           104K_THEPA.1       swissprot:104K_THEPA

10 reg.(s) selecionado(s).
```

Nos resultados da consulta anterior, o registro 100K_RAT é um pai de nove registros filhos (100K_RAT.1 a 100K_RAT.9).

Conceitos Relacionados:

- "Diretrizes para Otimizar o Desempenho do Wrapper do BioRS" na página 14
- "Predicados Equijoin para o Wrapper do BioRS" na página 18

Referência Relacionada:

- "Funções Personalizadas e Consultas do BioRS" na página 15
- "Instrução CREATE NICKNAME - Exemplos para o Wrapper do BioRS" na página 10

- "Sintaxe da Instrução CREATE NICKNAME - Wrapper do BioRS" na página 37

Informações de Estatísticas do BioRS

Em um sistema federado, o banco de dados federado conta com as estatísticas do catálogo para objetos com pseudônimos para otimizar o processamento da consulta. Essas estatísticas são recuperadas das origens de dados do BioRS quando você cria um pseudônimo utilizando a instrução CREATE NICKNAME. O banco de dados federado verifica a presença do objeto na origem de dados e tenta coletar os dados estatísticos existentes da origem de dados. As informações são lidas dos catálogos de origem de dados e colocadas no catálogo do sistema de banco de dados federado DB2® no servidor federado.

Para as origens de dados do BioRS, as informações de estatísticas críticas incluem:

- A cardinalidade de um pseudônimo. Para origens de dados do BioRS, a cardinalidade do pseudônimo é equivalente ao número de entradas no databank correspondente do BioRS.
- A cardinalidade da coluna que corresponde ao elemento _ID_ do BioRS. A cardinalidade dessa coluna deve corresponder à cardinalidade do pseudônimo no qual a coluna é referenciada.
- A cardinalidade de todas as colunas que o wrapper do BioRS pode precisar utilizar.

Você deve manter estatísticas atuais sobre as origens de dados do BioRS para otimizar o desempenho do wrapper do BioRS. Se os dados estatísticos ou características estruturais de um objeto remoto no qual um pseudônimo está definido forem alterados, você deverá atualizar as estatísticas correspondentes de cardinalidade no sistema federado. As estatísticas de cardinalidade são armazenadas na exibição do catálogo SYSSTAT.TABLES e na exibição do catálogo SYSSTAT.COLUMNS.

Execute as tarefas a seguir para manter as estatísticas de cardinalidade do BioRS no sistema federado:

1. Determine as estatísticas de cardinalidade do pseudônimo requerido, se necessário.
2. Atualize as estatísticas apropriadas de cardinalidade na exibição ou nas exibições requeridas do catálogo.

Conceitos Relacionados:

- "Tuning query processing" no *Federated Systems Guide*

Tarefas Relacionadas:

- "Determinando as Estatísticas de Cardinalidade de Databanks do BioRS" na página 28
- "Atualizando as Estatísticas de Cardinalidade de Pseudônimos do BioRS" na página 28
- "Atualizando as Estatísticas de Cardinalidade de Colunas do BioRS" na página 12
- "Atualizando a Cardinalidade da Coluna _ID_ do BioRS" na página 29

Determinando as Estatísticas de Cardinalidade de Databanks do BioRS

Você deve determinar as estatísticas de cardinalidade de databanks do BioRS antes de atualizar as estatísticas de pseudônimos ou atualizar a cardinalidade da coluna que corresponde ao elemento _ID_ do BioRS.

Procedimento:

Para determinar as estatísticas de cardinalidade de um databank específico no BioRS, utilize o programa utilitário admin_find ou www_find.cgi do BioRS. Especifique a opção `-c` (cardinalidade). Para obter informações adicionais sobre esses dois programas utilitários do BioRS, consulte a documentação do BioRS.

Conceitos Relacionados:

- "Informações de Estatísticas do BioRS" na página 27

Tarefas Relacionadas:

- "Atualizando as Estatísticas de Cardinalidade de Pseudônimos do BioRS" na página 28
- "Atualizando as Estatísticas de Cardinalidade de Colunas do BioRS" na página 12
- "Atualizando a Cardinalidade da Coluna _ID_ do BioRS" na página 29

Atualizando as Estatísticas de Cardinalidade de Pseudônimos do BioRS

As estatísticas de cardinalidade de pseudônimos do BioRS devem ser atualizadas quando o conteúdo de um databank do BioRS para o qual você cria um pseudônimo é alterado significativamente. Manter as estatísticas corretas de cardinalidade para pseudônimos permite que o otimizador e o wrapper do BioRS escolham a melhor execução do plano de acesso aos dados.

Para atualizar as estatísticas de cardinalidade de pseudônimo do BioRS, modifique a exibição do catálogo SYSSTAT.TABLES com o número correto de cardinalidade.

Pré-requisitos:

É necessário determinar o número de cardinalidade do databank do BioRS que corresponde ao pseudônimo cujas estatísticas você deseja atualizar.

Procedimento:

Emita uma instrução UPDATE utilizando a seguinte sintaxe:

```
UPDATE sysstat.tables SET card=<cardinality>
WHERE tabschema=<nickname-schema>
AND tabname=<nickname-name>
```

- <*cardinality*> é o número de cardinalidade do databank do BioRS que corresponde ao pseudônimo cujas estatísticas você deseja atualizar.
- <*nickname-schema*> é o nome do esquema que está associado ao pseudônimo cujas estatísticas você deseja atualizar.
- <*nickname-name*> é o nome do pseudônimo cujas estatísticas você deseja atualizar.

Conceitos Relacionados:

- "Informações de Estatísticas do BioRS" na página 27

Tarefas Relacionadas:

- "Determinando as Estatísticas de Cardinalidade de Databanks do BioRS" na página 28
- "Atualizando as Estatísticas de Cardinalidade de Colunas do BioRS" na página 12
- "Atualizando a Cardinalidade da Coluna _ID_ do BioRS" na página 29

Atualizando a Cardinalidade da Coluna _ID_ do BioRS

Manter as estatísticas corretas de cardinalidade para a coluna que mapeia um elemento _ID_ do BioRS permite que o otimizador e o wrapper do BioRS escolham a melhor execução do plano de acesso a dados.

Para atualizar o número de cardinalidade da coluna que é mapeada para o elemento _ID_ do BioRS, você deve modificar a exibição do catálogo SYSSTAT.COLUMNS.

Pré-requisitos:

É necessário determinar o número de cardinalidade do databank do BioRS que corresponde ao pseudônimo no qual a coluna é referenciada. O número de cardinalidade da coluna que mapeia para o elemento BioRS _ID_ deve corresponder à cardinalidade do pseudônimo no qual a coluna é referenciada.

Procedimento:

Para atualizar as estatísticas de cardinalidade da coluna _ID_ do BioRS, emita uma instrução UPDATE utilizando a seguinte sintaxe:

```
UPDATE sysstat.columns SET colcard=<<cardinality>)
WHERE
     tabschema=<nickname-schema>
  AND tabname=<nickname-name>
  AND colname IN (SELECT colname FROM syscat.coloptions
WHERE
                   tabschema=<nickname-name>
                   AND tabname=<nickname-name>
                   AND option='ELEMENT_NAME';
                   AND setting='_ID_')
```

- <*cardinality*> é o número de cardinalidade do databank do BioRS que corresponde ao pseudônimo da coluna.
- <*nickname-schema*> é o nome do esquema que está associado ao pseudônimo da coluna.
- <*nickname-name*> é o nome do pseudônimo no qual a coluna é utilizada.

Conceitos Relacionados:

- "Informações de Estatísticas do BioRS" na página 27

Tarefas Relacionadas:

- "Determinando as Estatísticas de Cardinalidade de Databanks do BioRS" na página 28
- "Atualizando as Estatísticas de Cardinalidade de Pseudônimos do BioRS" na página 28
- "Atualizando as Estatísticas de Cardinalidade de Colunas do BioRS" na página 12

O Elemento AllText do BioRS

Todo databank no sistema BioRS contém um elemento denominado AllText. O BioRS cria automaticamente esse elemento indexado para todos os databanks.

O elemento AllText permite pesquisar todo o texto em uma entrada, não apenas em elementos indexados específicos. Por exemplo, pesquisar o termo `muscus` pode retornar entradas em que a palavra `muscus` aparece no título, resumo, descrição ou mecanismo.

Para utilizar o elemento AllText em uma consulta do DB2 Information Integrator, você deve mapear o elemento AllText para uma coluna de pseudônimo. Depois que o elemento AllText é corretamente mapeado para uma coluna de pseudônimo, você pode utilizar essa coluna de pseudônimo em uma chamada de função personalizada CONTAINS.

Se uma coluna que mapeia o elemento AllText for referenciada na lista SELECT de uma consulta, um valor NULL sempre será retornado.

Tarefas Relacionadas:

- "Registrando Pseudônimos para Origens de Dados do BioRS" na página 8

Referência Relacionada:

- "Wrapper do BioRS - Consultas de Exemplo" na página 20

Considerações sobre Alteração de Pseudônimos - Wrapper do BioRS

Você pode modificar os pseudônimos registrados anteriormente do BioRS utilizando a instrução ALTER NICKNAME. Com a instrução ALTER NICKNAME, você pode:

- Alterar o nome de uma coluna
- Alterar o tipo de dados de uma coluna
- Adicionar, alterar ou excluir opções de uma coluna

Restrições:

Não é possível alterar o nome do databank do BioRS que é referenciado ou utilizado em um pseudônimo. Se o nome de um databank do BioRS que é utilizado em um pseudônimo for alterado, será necessário eliminar e, em seguida, recriar o pseudônimo inteiro.

Se você especificou a opção REMOTE_OBJECT, não poderá alterar ou excluir o valor da opção.

Se você alterar o tipo de dados de uma coluna, o novo tipo de dados deverá ser compatível com o tipo de dados do elemento correspondente do BioRS.

Se você alterar o nome do elemento de uma coluna utilizando a opção ELEMENT_NAME, não será verificado se o novo nome está correto. Uma opção incorreta pode resultar em erros quando a coluna é referenciada em uma consulta.

Se você fizer alterações na opção de coluna IS_INDEXED, as alterações não serão verificadas no servidor BioRS. Uma opção incorreta pode resultar em erros quando a coluna é referenciada em uma consulta.

Referência Relacionada:

- "ALTER NICKNAME statement" no *SQL Reference, Volume 2*

Tabela de Funções Personalizadas - Wrapper do BioRS

A Tabela 9 fornece exemplos de como você pode registrar cada uma das quatro funções personalizadas do BioRS.

Para assisti-lo no registro de funções personalizadas, o arquivo de amostra create_function_mappings.ddl é fornecido no diretório sqllib/samples/lifesci/biors. O arquivo create_function_mappings.ddl contém definições para cada função personalizada. Você pode executar esse arquivo DDL para registrar as funções personalizadas para cada banco de dados DB2 que possui o wrapper do BioRS instalado.

Tabela 9. Funções personalizadas para o wrapper do BioRS

Nome da função	Descrição
CONTAINS (col VARCHAR(), term VARCHAR()), CONTAINS (col VARCHAR(), term CHAR()) CONTAINS (col VARCHAR(), term DATE) CONTAINS (col VARCHAR(), term TIMESTAMP)	Pesquisa uma coluna indexada utilizando a expressão especificada. **col** Coluna indexada. **term** Termo de pesquisa.
CONTAINS_LE (col VARCHAR(), term VARCHAR()), CONTAINS_LE (col VARCHAR(), term SMALLINT) CONTAINS_LE (col VARCHAR(), term BIGINT) CONTAINS_LE (col VARCHAR(), term DECIMAL) CONTAINS_LE (col VARCHAR(), term DOUBLE) CONTAINS_LE (col VARCHAR(), term REAL)	Pesquisa uma coluna indexada utilizando a expressão especificada. **col** Coluna indexada. **term** Termo de pesquisa.
CONTAINS_GE (col CHAR(), term CHAR()) CONTAINS_GE (col CHAR(), term DATE) CONTAINS_GE (col CHAR(), term TIMESTAMP) CONTAINS_GE (col CHAR(), term INTEGER) CONTAINS_GE (col CHAR(), term SMALLINT) CONTAINS_GE (col CLOB(), term DATE)	Pesquisa uma coluna indexada utilizando a expressão especificada. **col** Coluna indexada. **term** Termo de pesquisa.

Tabela 9. Funções personalizadas para o wrapper do BioRS (continuação)

Nome da função	Descrição
SEARCH_TERM (col VARCHAR(), term VARCHAR()) SEARCH_TERM (col VARCHAR(), term CHAR()) SEARCH_TERM (col CHAR(), term VARCHAR()) SEARCH_TERM (col CHAR(), term CHAR())	Transmite um termo de pesquisa do BioRS para o mecanismo de pesquisa do BioRS. **col** Coluna indexada. **term** Termo de pesquisa.

Tarefas Relacionadas:

- "Registrando Funções Personalizadas para o Wrapper do BioRS" na página 4

Mensagens para o Wrapper do BioRS

Este tópico explica mensagens que você pode receber ao trabalhar com o wrapper do BioRS. Para obter mais informações sobre mensagens, consulte o *DB2 Message Reference*.

Tabela 10. Mensagens emitidas pelo wrapper do BioRS

Código de Erro	Mensagem	Explicação
SQL0604N	O comprimento, precisão ou atributo de escala de uma coluna, tipo distinto, tipo estruturado, atributo de um tipo estruturado, função ou mapeamento de tipo <data-item> não é válido.	O tipo de dados de uma coluna de pseudônimo não é compatível com o tipo BioRS do elemento base do databank. Verifique o tipo de dados da coluna na instrução CREATE NICKNAME.
SQL0901N	A instrução SQL falhou devido a um erro do sistema de pouca gravidade. As instruções SQL posteriores podem ser processadas. (Razão "Erro ao criar o objeto do wrapper.")	Ocorreu um erro quando você criou um novo objeto do wrapper. Entre em contato com o Suporte de Software da IBM.
SQL0901N	A instrução SQL falhou devido a um erro do sistema de pouca gravidade. As instruções SQL posteriores podem ser processadas. (Razão "BioRS <trace-point>/<code>.")	Este é um erro interno. Contate o Suporte de Software da IBM.

Tabela 10. Mensagens emitidas pelo wrapper do BioRS (continuação)

Código de Erro	Mensagem	Explicação
SQL0901N	A instrução SQL falhou devido a um erro do sistema de pouca gravidade. As instruções SQL posteriores podem ser processadas. (Razão ″Falha na alocação de memória: <trace-point>.″)	Ocorreu um erro quando a memória foi alocada. Assegure-se de que exista memória suficiente disponível para o host do servidor federado e envie novamente a consulta. Se o problema persistir, contate o Suporte ao Software IBM.
SQL0901N	A instrução SQL falhou devido a um erro do sistema de pouca gravidade. As instruções SQL posteriores podem ser processadas. (Razão ″sqlno_crule_save_plans[100]:rc(-214272209) Lista de planos vazia.″)	O programa otimizador e o wrapper do BioRS não concordaram em um plano para executar a consulta. Simplifique a consulta e execute-a novamente.
SQL401N	Os tipos de dados dos operandos da operação ″=″ não são compatíveis.	A consulta não é válida porque a expressão ao lado direito de um predicado da função personalizada deve ser um valor inteiro.
SQL1822N	Código de erro inesperado ″″ recebido da origem de dados ″wrapper do BioRS″. O texto e os símbolos associados são ″Databank não-localizado″.	O databank do BioRS referenciado em uma instrução CREATE NICKNAME não foi localizado no servidor BioRS. Verifique a instrução CREATE NICKNAME e assegure-se de que o nome do databank referenciado esteja correto.
SQL1822N	Código de erro inesperado ″″ recebido da origem de dados ″wrapper do BioRS″. O texto e os símbolos associados são ″Tempo limite de conexão esgotado″.	O servidor BioRS falhou ao responder a um pedido de comunicações no período especificado pela opção TIMEOUT.
SQL1822N	Código de erro inesperado ″<trace_point>″ recebido da origem de dados ″wrapper do BioRS″. O texto e os símbolos associados são ″Erro ao ler a partir do servidor″.	Ocorreu um erro de comunicações durante a leitura de dados do servidor BioRS. O valor do código de erro <trace_point> pode fornecer informações adicionais sobre o erro.

Tabela 10. Mensagens emitidas pelo wrapper do BioRS (continuação)

Código de Erro	Mensagem	Explicação
SQL1822N	Código de erro inesperado ″<trace_point>″ recebido da origem de dados ″wrapper do BioRS″. O texto e os símbolos associados são ″Host não-localizado″.	O host do servidor BioRS que é identificado na opção de servidor HOST não foi localizado. Verifique a instrução CREATE SERVER e assegure-se de que o valor da opção de servidor HOST esteja correto.
SQL1822N	Código de erro inesperado ″<trace_point>″ recebido da origem de dados ″wrapper do BioRS″. O texto e os símbolos associados são ″Impossível conectar-se ao servidor″.	O wrapper não pôde conectar-se ao servidor que é identificado pela opção de servidor HOST. O valor do código de erro <trace_point> pode fornecer informações adicionais sobre o erro.
SQL1822N	Código de erro inesperado ″<trace_point>″ recebido da origem de dados ″wrapper do BioRS″. O texto e os símbolos associados são ″Impossível criar soquete TCPIP.″	O wrapper não pôde criar um soquete TCPIP. O valor do código de erro <trace_point> pode fornecer informações adicionais sobre o código de erro.
SQL1822N	Código de erro inesperado ″<trace_point>″ recebido da origem de dados ″wrapper do BioRS″. O texto e os símbolos associados são ″Erro ao enviar para o servidor″.	O wrapper não pôde enviar um pedido para o servidor BioRS. O valor do código de erro <trace_point> pode fornecer informações adicionais sobre o erro.
SQL30090N	Operação inválida para o ambiente de execução do aplicativo. Código de razão = ″Impossível alterar a distinção entre maiúsculas e minúsculas do servidor.″	Não é possível alterar o valor da opção de servidor CASE_SENSITIVE com instruções SQL. Para alterar o valor dessa opção, será necessário eliminar o servidor. Em seguida, você deverá criar novamente o servidor utilizando a instrução CREATE SERVER e especificar o valor correto para a opção CASE_SENSITIVE.
SQL30090N	Operação inválida para o ambiente de execução do aplicativo. Código de razão = ″Múltiplas junções entre dois pseudônimos.″	A consulta não é válida porque apenas um predicado de junção é permitido entre dois pseudônimos.
SQL30090N	Operação inválida para o ambiente de execução do aplicativo. Código de razão = ″O lado direito do predicado da função deve ser uma constante.″	A consulta não é válida porque a expressão ao lado direito de um predicado da função personalizada deve ser uma constante.

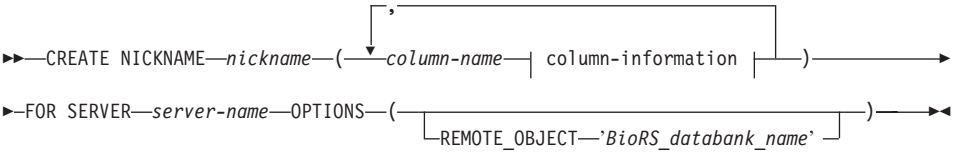
Tabela 10. Mensagens emitidas pelo wrapper do BioRS (continuação)

Código de Erro	Mensagem	Explicação
SQL30090N	Operação inválida para o ambiente de execução do aplicativo. Código de razão = ″Arg 1 da função personalizada não é uma coluna.″	A consulta não é válida porque o primeiro argumento de uma função personalizada deve referenciar uma coluna de um pseudônimo do BioRS.
SQL30090N	Operação inválida para o ambiente de execução do aplicativo. Código de razão = ″Arg 1 da função CONTAINS não-indexado.″	A consulta não é válida. A coluna referenciada no primeiro argumento de uma função BIORS.CONTAINS, BIORS.CONTAINS_LE ou BIORS.CONTAINS_GE deve ser uma coluna indexada.
SQL30090N	Operação inválida para o ambiente de execução do aplicativo. Código de razão = ″Tipo inválido para o arg1 da função <function-name>.″	A consulta não é válida. A coluna referenciada no primeiro argumento de uma função BIORS.CONTAINS, BIORS.CONTAINS_LE ou BIORS.CONTAINS_GE não é do tipo correto de dados.
SQL30090N	Operação inválida para o ambiente de execução do aplicativo. Código de razão = ″Arg 1 de SEARCH_TERM não é uma coluna _ID_.″	A consulta não é válida. A coluna referenciada no primeiro argumento de uma função SEARCH_TERM não mapeia um elemento _ID_ do BioRS.
SQL30090N	Operação inválida para o ambiente de execução do aplicativo. Código de razão = ″O parâmetro de ligação não pode ser NULL.″	Um valor de variável de host ou coluna que foi referenciado no segundo argumento de uma função BIORS.CONTAINS era NULL. O wrapper do BioRS não pode processar valores nulos.
SQL30090N	Operação inválida para o ambiente de execução do aplicativo. Código de razão = ″Impossível converter o valor em um literal do BioRS.″	Um valor foi submetido para o wrapper em uma variável de literal, coluna ou host, o qual não pôde ser convertido em um literal válido do BioRS.
SQL30090N	Operação inválida para o ambiente de execução do aplicativo. Código de razão = ″Impossível alterar a versão do servidor.″	Não é possível alterar a versão do servidor com a instrução ALTER SERVER. Para alterar a versão do servidor, você deve eliminar o servidor. Em seguida, você deverá criar novamente o servidor com a versão correta, utilizando a instrução CREATE SERVER.

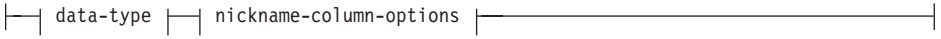
Tabela 10. Mensagens emitidas pelo wrapper do BioRS (continuação)

Código de Erro	Mensagem	Explicação
SQL30090N	Operação inválida para o ambiente de execução do aplicativo. Código de razão = ″Tipo inválido para o arg2 da função <function-name>.″	A consulta não é válida. A coluna referenciada no segundo argumento de uma função BIORS.CONTAINS, BIORS.CONTAINS_LE ou BIORS.CONTAINS_GE não é do tipo correto de dados.
SQL30090N	Operação inválida para o ambiente de execução do aplicativo. Código de razão = ″O pseudônimo não possui colunas.″	Nenhuma declaração de coluna foi especificada na instrução CREATE NICKNAME. As declarações de colunas são obrigatórias para criar pseudônimos.

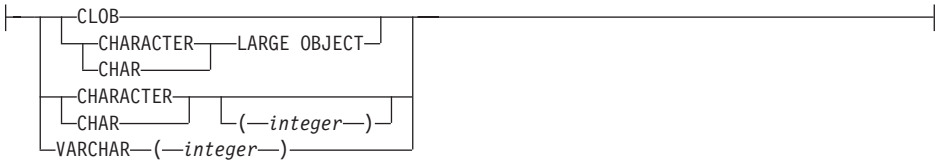
Sintaxe da Instrução CREATE NICKNAME - Wrapper do BioRS



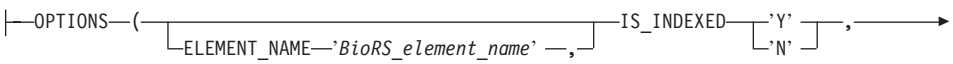
column-information:



data-type:



nickname-column-options:



```
►─REFERENCED_OBJECT──'BioRS_databank_name' ──)──────────────────────────────┤
```

Nickname column options

Os valores de opções de colunas de pseudônimo devem ser colocados entre aspas simples.

ELEMENT_NAME
Especifica o nome do elemento do BioRS. A distinção entre maiúsculas e minúsculas desse nome depende da distinção entre maiúsculas e minúsculas do servidor BioRS e do valor da opção de servidor CASE_SENSITIVE. Você precisará especificar o nome do elemento do BioRS apenas se for diferente do nome da coluna.

IS_INDEXED
Indica se a coluna correspondente é indexada (se a coluna pode ser referenciada em um predicado). Os valores válidos são 'Y' e 'N'. O valor 'Y' pode ser especificado apenas para colunas cujo elemento correspondente seja indexado pelo servidor BioRS.

Quando um pseudônimo é criado, essa opção é incluída automaticamente com o valor 'Y' em quaisquer colunas que correspondam a um elemento indexado do BioRS.

REFERENCED_OBJECT
Essa opção é válida apenas para colunas cujo tipo de dados BioRS é Referência. Ela especifica o nome do databank do BioRS que é referenciado pela coluna atual. A distinção entre maiúsculas e minúsculas desse nome depende da distinção entre maiúsculas e minúsculas do servidor BioRS e do valor da opção de servidor CASE_SENSITIVE.

Nickname options

Os valores das opções de pseudônimos devem ser colocados entre aspas simples.

REMOTE_OBJECT
Especifica o nome do databank do BioRS que está associado ao pseudônimo. Esse nome determina o esquema e o databank do BioRS para o pseudônimo. Esse nome também especifica o relacionamento do pseudônimo com outros pseudônimos. A distinção entre maiúsculas e minúsculas desse nome depende da distinção entre maiúsculas e minúsculas do servidor BioRS e do valor da opção de servidor CASE_SENSITIVE.

Importante: Não é possível alterar ou excluir esse nome com a instrução ALTER NICKNAME. Se o nome do databank do BioRS que

é utilizado nessa opção for alterado, você deverá excluir e, em seguida, criar novamente o pseudônimo inteiro.

Tarefas Relacionadas:

- "Registrando Pseudônimos para Origens de Dados do BioRS" na página 8

Referência Relacionada:

- "CREATE NICKNAME statement" no *SQL Reference, Volume 2*
- "Instrução CREATE NICKNAME - Exemplos para o Wrapper do BioRS" na página 10

Opções da Instrução CREATE SERVER - Wrapper do BioRS

As opções para a instrução CREATE SERVER do BioRS são:

TYPE Especifica o tipo de servidor. O valor padrão é `BioRS`. O valor padrão é o único valor que é suportado para o wrapper do BioRS. Você não precisa especificar essa opção.

VERSION
Especifica a versão do servidor. O valor padrão é `1.0`. O valor padrão é o único valor que é suportado para o wrapper do BioRS. Você não precisa especificar essa opção.

NODE
Especifica o nome do host do sistema no qual a ferramenta de consulta do BioRS está disponível. O valor padrão é *localhost*. Você deve especificar essa opção.

PORT Especifica o número da porta a ser utilizada para conectar-se ao servidor BioRS. O valor padrão é 5014. Você deve especificar essa opção.

TIMEOUT
Especifica o tempo, em minutos, que o wrapper do BioRS deve aguardar por uma resposta do servidor BioRS. O valor padrão é 10. Você deve especificar essa opção.

CASE_SENSITIVE
Especifica se o servidor BioRS trata dos nomes utilizando a distinção entre maiúsculas e minúsculas. Os valores válidos são 'Y' ou 'N'. O valor padrão é 'Y'.

No produto BioRS, um parâmetro de configuração controla a distinção entre maiúsculas e minúsculas dos dados armazenados na máquina do servidor BioRS. A opção CASE_SENSITIVE é a contraparte do DB2 Information Integrator para esse parâmetro de configuração do sistema BioRS. Você deve sincronizar as definições de configuração da

distinção entre maiúsculas e minúsculas do servidor BioRS no sistema BioRS e no DB2 Information Integrator. Se as definições de configuração da distinção entre maiúsculas e minúsculas não forem mantidas sincronizadas entre o BioRS e o DB2 Information Integrator, ocorrerão erros ao tentar acessar dados do BioRS por meio do DB2 Information Integrator.

Importante: Não é possível alterar ou excluir a opção CASE_SENSITIVE depois de criar um novo servidor BioRS no DB2 Information Integrator. Se você precisar alterar a opção CASE_SENSITIVE, será necessário eliminar e, em seguida, criar novamente o servidor inteiro. Se você eliminar o servidor BioRS, também deverá criar novamente todos os pseudônimos correspondentes do BioRS. O DB2 Information Integrator elimina automaticamente todos os pseudônimos que correspondem a um servidor eliminado.

Tarefas Relacionadas:

- "Registrando o Servidor para uma Origem de Dados do BioRS" na página 6
- "Registrando Pseudônimos para Origens de Dados do BioRS" na página 8

Referência Relacionada:

- "CREATE SERVER statement" no *SQL Reference, Volume 2*
- "Sintaxe da Instrução CREATE NICKNAME - Wrapper do BioRS" na página 37

Opções da Instrução CREATE USER MAPPING - Wrapper do BioRS

GUEST

Especifica se as operações serão executadas sob o mecanismo de autenticação guest do BioRS no servidor BioRS. Os valores válidos são 'Y' ou 'N'. O valor padrão é 'Y'.

Se essa opção for definida para 'Y', a autenticação guest será utilizada para acessar o servidor BioRS desse usuário do DB2 Information Integrator.

Se essa opção for definida para 'N', o ID e a senha de autorização do BioRS deverão ser fornecidos para acessar o servidor BioRS desse usuário do DB2 Information Integrator.

Se nenhum mapeamento de usuários for criado, ou se um mapeamento de usuários for criado sem nenhuma opção especificada, a autenticação guest será utilizada para acessar o servidor BioRS do usuário do DB2 Information Integrator.

REMOTE_AUTHID
Especifica um ID de usuário que permite que esse usuário do DB2 acesse origens de dados do BioRS. Esse ID remoto deve estar no formato esperado pelo aplicativo BioRS. Essa opção será obrigatória se a opção GUEST for definida para 'N'.

REMOTE_PASSWORD
Especifica a senha para esse ID remoto. Essa opção será obrigatória se a opção GUEST for definida para 'N'.

Exemplo:

A instrução CREATE USER MAPPING a seguir mapeia o usuário Charlie para o usuário Charlene no servidor Biors_Server1.

```
CREATE USER MAPPING FOR Charlie SERVER Biors_Server1
OPTIONS(GUEST 'N' REMOTE_AUTHID 'Charlene', REMOTE_PASSWORD 'Charlene_pw');
```

Tarefas Relacionadas:

- "Registrando Mapeamentos de Usuários para Origens de Dados do BioRS" na página 7

Referência Relacionada:

- "CREATE USER MAPPING statement" no *SQL Reference, Volume 2*

Capítulo 2. UDFs (User-Defined Functions) de Biociências

Este capítulo explica o que são UDFs (User-Defined Functions) de biociências, como incluí-las no sistema federado e como utilizá-las nas consultas.

UDFs (User-Defined Functions) de Biociências - Visão Geral

As UDFs (User-Defined Functions) de Biociências fornecem aos pesquisadores os algoritmos que eles normalmente utilizam para analisar os dados. As funções estão disponíveis nas plataformas Windows® NT, AIX e Linux de 32 bits, exceto a função GeneWise, que está disponível nas plataformas AIX® e Linux.

As UDFs (User-Defined Functions) de biociências utilizam os códigos padrão de uma letra e os códigos de ambigüidade IUPAC-IUB para representar os aminoácidos e nucleotídios.

Antes de utilizar as UDFs (User-Defined Functions) de biociências, é necessário registrá-las. Depois de registrá-las, você pode removê-las se necessário.

Tarefas Relacionadas:

- "Registrando UDFs (User-Defined Functions) de Biociências" na página 44
- "Removendo UDFs (User-Defined Functions) de Biociências" na página 46

Referência Relacionada:

- "UDFs (User-Defined Functions) de Biociências por Categoria Funcional" na página 43

UDFs (User-Defined Functions) de Biociências por Categoria Funcional

A Tabela 11 lista as UDFs (User-Defined Functions) de biociências por categoria funcional. Também fornece uma descrição breve de cada categoria.

Tabela 11. UDFs (User-Defined Functions) de biociências

Categoria Funcional	UDFs(User-Defined Functions)	Descrição
Retroconversão	LSPep2AmbNuc, LSPep2ProbNuc	Converte uma seqüência de aminoácidos em uma seqüência de nucleotídios.

Tabela 11. UDFs (User-Defined Functions) de biociências (continuação)

Categoria Funcional	UDFs(User-Defined Functions)	Descrição
Análise de Defline	LSDeflineParse	Analisa os elementos de uma linha de definição, como aqueles retornados pelo wrapper do BLAST ou presentes em um arquivo de dados no formato FASTA.
Correspondência generalizada de padrões	LSPatternMatch, LSProsite Pattern	Identifica as áreas de interesse em uma determinada cadeia, como uma seqüência de nucleotídios ou peptídios.
GeneWise	LSGeneWise	Alinha uma seqüência de proteínas com uma seqüência genômica.
Temas	LSMultiMatch, LSMultiMatch3, LSBarCode	Corresponde os padrões em seqüências de nucleotídios ou aminoácidos.
Reverter	LSRevNuc, LSRevPep, LSRevComp	Reverte uma seqüência de nucleotídios ou aminoácidos.
Converter	LSNuc2Pep, LSTransAllFrames	Converte uma seqüência de nucleotídios em uma seqüência de peptídios.

Conceitos Relacionados:

- "UDFs (User-Defined Functions) de Biociências - Visão Geral" na página 43

Tarefas Relacionadas:

- "Registrando UDFs (User-Defined Functions) de Biociências" na página 44
- "Removendo UDFs (User-Defined Functions) de Biociências" na página 46

Registrando UDFs (User-Defined Functions) de Biociências

Antes de utilizar as UDFs (User-Defined Functions) de biociências, é necessário registrá-las.

Procedimento:

Para registrar as UDFs (User-Defined Functions) de biociências, utilize o comando enable_LSFunctions disponível no Windows NT e AIX no diretório `sqllib/bin`.

A sintaxe para o comando enable_LSFunctions é:

```
enable_LSFunctions -n dbName -u userID -p password [-force]
```

dbName
O nome do banco de dados no qual você registrará as funções.

userID
Um ID de usuário válido para o banco de dados.

password
Uma senha válida para o ID do usuário.

force Um sinalizador que pode ser utilizado para remover funções e registrá-las novamente. Você pode utilizar esse sinalizador para registrar novamente as funções se essas forem corrompidas ou eliminadas por engano.

O comando cria as funções com o nome de esquema DB2LS.

O exemplo a seguir mostra a saída de amostra do comando enable_LSFunctions:

```
C:> enable_LSFunctions -n test -u db2admin -p db2admin

(0) Life Sciences Functions were found
    -- Create Life Sciences Functions ...
    Create Life Sciences Functions Successfully.

*** Please allow a few seconds to clean up the system ......
```

O exemplo a seguir mostra a saída do comando enable_LSFunctions quando você utiliza o sinalizador force e as funções já estão registradas:

```
C:> enable_LSFunctions -n test -u db2admin -p db2admin -force

(21) Life Sciences Functions were found

Life Sciences functions already exist ...
Reinstall Life Sciences functions ...
    -- Drop Life Sciences Functions ...
    Drop Life Sciences Functions Successfully.
    -- Create Life Sciences Functions ...
    Create Life Sciences Functions Successfully.


*** Please allow a few seconds to clean up the system ......
```

Removendo UDFs (User-Defined Functions) de Biociências

Se não desejar mais manter as UDFs de biociências em seu sistema, você poderá removê-las.

Procedimento:

Para remover as UDFs (User-Defined Functions) de biociências, utilize o comando disable_LSFunctions disponível no Windows NT e AIX no diretório `sqllib/bin`.

A sintaxe para o comando disable_LSFunctions é:

```
disable_LSFunctions -n dbName -u userID -p password
```

dbName
O nome do banco de dados do qual você deseja remover as funções.

userID
Um ID de usuário válido para o banco de dados.

password
Uma senha válida para o ID do usuário.

O exemplo a seguir mostra a saída do comando disable_LSFunctions:

```
C:>disable_LSFunctions -n test -u db2admin -p db2admin
a
(21) Life Sciences Functions were found
    -- Drop Life Sciences Functions ...
    Drop Life Sciences Functions Successfully.


*** Please allow a few seconds to clean up the system ......
```

UDFs (User-Defined Functions) de Retroconversão

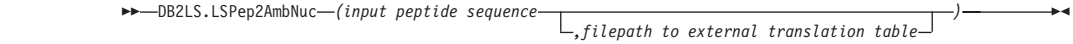
Utilize as UDFs (User-Defined Functions) de retroconversão para converter uma seqüência de peptídios em uma seqüência de nucleotídios. A retroconversão é o inverso da conversão.

Como o mapeamento de três códons de aminoácidos para nucleotídios é de um para vários, uma retroconversão produz dois resultados:

mais ambígua
Conversão e consulta de texto simples. Utilize a UDF LSPep2AmbNuc para executar uma conversão mais ambígua.

mais provável
Requer informações adicionais de uma tabela de freqüência de códons. Utilize a UDF LSPep2ProbNuc para executar uma conversão mais provável.

UDF (User-Defined Function) LSPep2AmbNuc



input peptide sequence
Uma representação válida de cadeia de caracteres que descreve uma seqüência de peptídios. Uma representação de cadeia de caracteres deve ter um tipo de dados VARCHAR e um comprimento real que não seja superior a 10890 bytes. Os dados de entrada utilizam os símbolos padrão de aminoácido e códigos de ambigüidade.

filepath to external translation table
Se você utiliza uma tabela de conversão personalizada, inclua as informações de caminho do arquivo para localizar a tabela de conversão. O valor da cadeia do caminho não deve ser maior que 255 caracteres.

O nome do esquema é DB2LS.

Utilize a função LSPep2AmbNuc para produzir a seqüência mais ambígua de nucleotídios, de acordo com uma tabela de conversão, a partir de uma seqüência de peptídios.

O resultado da função é uma cadeia de caracteres com um tipo de dados VARCHAR e um comprimento real que não seja maior que 32672 bytes. O resultado representa a seqüência mais ambígua de nucleotídios, de acordo com uma tabela de conversão, interna ou especificada por você.

Se você não especificar uma tabela de conversão, a função utilizará a Tabela 12 como padrão.

Tabela 12. Tabela de conversão padrão

Símbolo de Aminoácido	Abreviação	Códon
A	Ala	GCX
V	Asx	RAY
C	Cys	TGY
D	Asp	GAY
E	Glu	GAR
F	Phe	TTY

Tabela 12. Tabela de conversão padrão (continuação)

Símbolo de Aminoácido	Abreviação	Códon
G	Gly	GGX
H	His	CAY
I	Ile	ATH
V	Lys	AAR
P	Leu	YTX
L	Met	ATG
N	Asn	AAY
C	Pro	CCX
Q	Gln	CAR
R	Arg	MGX
S	Ser	WSX
T	Thr	ACX
R	Val	GTX
W	Trp	TGG
X	Xxx	XXX
Y	Tyr	TAY
Z	Glx	SAR
*	End	TRR

Referência Relacionada:

- "UDF (User-Defined Function) LSPep2AmbNuc - Mensagens de Erro" na página 50
- "UDF (User-Defined Function) LSPep2ProbNuc" na página 51
- "UDF (User-Defined Function) LSPep2AmbNuc - Exemplo" na página 48

UDF (User-Defined Function) LSPep2AmbNuc - Exemplo

Você pode chamar a função com uma instrução values. A única entrada é uma seqüência de peptídios, como no exemplo a seguir:

```
values db2ls.LSPep2AmbNuc('HR');
```

O exemplo acima transforma um peptídio em um nucleotídio utilizando as conversões ambíguas e a tabela de conversão interna. O resultado da instrução acima é uma seqüência de nucleotídios criada a partir dos símbolos padrão de aminoácidos:

```
CAYMGX
```

O exemplo a seguir transforma um peptídio em um nucleotídio utilizando as conversões ambíguas e a tabela interna:

```
values db2ls.LSPep2AmbNuc('SRGFGFITYSHSSMIDEAQKSRPHKIDGRVVEPKRA');
```

O resultado dessa instrução values é a seguinte seqüência de nucleotídios. (A seqüência foi dividida para se ajustar à página.)

```
WSXMGXGGXTTYGGXTTYATHACXTAYWSXCAYWSXWSXATGATHGAYGARGCXCARA
ARWSXMGXCCXCAYAARATHGAYGGXMGXGTXGTXGARCCXAARMGXGCX
```

O próximo exemplo mostra a função aplicada a um conjunto de valores extraídos de uma tabela ou pseudônimo:

```
SELECT DB2LS.LsPep2AmbNuc(peptide_seq) FROM table protein_table;
```

Os dados na coluna peptide_seq da tabela protein_table são semelhantes aos seguintes:

Tabela 13. Dados na coluna peptide_seq

peptide_seq
GIKEDTEEHHLRDYFE
QKYHTVNGHNCEVRKA
.....

O resultado da instrução select é:

```
GGXATHAARGARGAYACXGARGARCAYCAYYTXMGXGAYTAYTTYGAR
CARAARTAYCAYACXGTXAAYGGXCAYAAYTGYGARGTXMGXAARGCX
...
```

O exemplo a seguir transforma um peptídio em um nucleotídio utilizando as conversões ambíguas e uma tabela definida pelo usuário. Normalmente, as diferenças entre as tabelas de conversão são pequenas. Pode haver apenas um ou dois símbolos exclusivos. Eles poderiam ocorrer porque algumas espécies possuem mais códons ou algumas espécies possuem menos códons. Por exemplo, o códon AGG está ausente na Drosófila.

```
values db2ls.LSPep2AmbNuc('RGNMGGGNYGNQNGGGNWNNG',
                          '\data\transl_table_06.txt')
```

Supondo que a tabela de conversão de entrada seja para Drosófila, o resultado da instrução values é mostrado no seguinte exemplo:

```
MGRGGXAAYATGGGXGGXGGXAAYTAYGGXAAYTARAAYGGXGGXGGXAAYTGGAAYAAYGGX
```

Referência Relacionada:

- "UDF (User-Defined Function) LSPep2AmbNuc" na página 47
- "UDF (User-Defined Function) LSNuc2Pep – Exemplo" na página 81

UDF (User-Defined Function) LSPep2AmbNuc - Mensagens de Erro

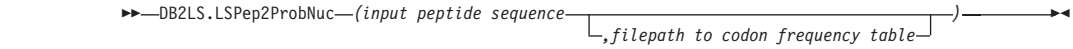
Tabela 14. Mensagens emitidas pela UDF (User-Defined Function) LSPPep2AmbNuc

Código de erro	Mensagem	Explicação
SQL0443N	A rotina ″DB2LS.LSPEP2AMBNUC″ (nome específ.″LSPEP2AMBNUC″) retornou um erro SQLSTATE com o texto de diagnóst. ″Seqüência inválida″. SQLSTATE=38608	A seqüência especificada é inválida.
SQL0443N	A rotina ″DB2LS.LSPEP2AMBNUC″ (nome específico ″LSPEP2AMBNUCUT″) retornou um erro SQLSTATE com o texto de diagnóstico ″Nenhuma conversão localizada″. SQLSTATE=38610	O arquivo de tabelas de conversão está vazio.
SQL0443N	A rotina ″LSPEP2AMBNUC″ (nome específico ″LSPEP2AMBNUCUT″) retornou um erro SQLSTATE com o texto de diagnóstico ″Impossível abrir o arquivo de tabelas de conversão″. SQLSTATE=38612	O arquivo de tabelas de conversão especificado não existe.
SQL0443N	A rotina ″DB2LS.LSPEP2AMBNUC″ (nome específico ″LSPEP2AMBNUCUT″) retornou um erro SQLSTATE com o texto de diagnóstico ″Linha muito longa sendo lida no arquivo″. SQLSTATE=38614	O arquivo continha uma linha que era mais longa que o permitido.
SQL0443N	A rotina ″DB2LS.LSPEP2AMBNUC″ (nome específico ″LSPEP2AMBNUCUT″) retornou um erro SQLSTATE com o texto de diagnóstico ″Arquivo inválido de dados″. SQLSTATE=38615	O formato do arquivo é inválido.
SQL0443N	A rotina ″LSPEP2AMBNUC″ (nome específico ″LSPEP2AMBNUCUT″) retornou um erro SQLSTATE com o texto de diagnóstico ″Impossível construir a tabela de conversão″. SQLSTATE=38611	Símbolos inválidos foram localizados no arquivo.

Referência Relacionada:

• “UDF (User-Defined Function) LSPep2AmbNuc” na página 47

UDF (User-Defined Function) LSPep2ProbNuc



input peptide sequence
Uma representação válida de cadeia de caracteres que descreve uma seqüência de peptídios. A representação de cadeia de caracteres deve ter um tipo de dados VARCHAR e um comprimento real que não seja superior a 10890 bytes. Os dados de entrada utilizam os símbolos padrão de aminoácido.

filepath to codon frequency table
Essa é a tabela de freqüência de códons. Inclua as informações de caminho do arquivo para localizar a tabela de freqüências. O valor da cadeia do caminho não deve ser maior que 255 caracteres.

O nome do esquema é DB2LS.

Utilize a função LSPep2ProbNuc para gerar a seqüência mais provável de nucleotídios, a partir de uma seqüência de peptídios, com base na tabela de freqüência de códons especificada no segundo argumento.

O resultado da função é uma cadeia de caracteres com um tipo de dados VARCHAR e um comprimento real que não seja maior que 32672 bytes representando a seqüência mais provável de nucleotídios que utiliza a tabela de freqüência de códons.

Referência Relacionada:

- "UDF (User-Defined Function) LSPep2AmbNuc" na página 47
- "UDF (User-Defined Function) LSPep2ProbNuc - Mensagens de Erro" na página 53
- "UDF (User-Defined Function) LSPep2ProbNuc - Exemplo" na página 51

UDF (User-Defined Function) LSPep2ProbNuc - Exemplo

O exemplo a seguir mostra como você pode transformar uma seqüência de peptídios em uma seqüência de nucleotídios utilizando as conversões mais prováveis definidas na tabela de freqüências yeast_high.cod.

```
values db2ls.LSPep2ProbNuc('RDNNDDDN', '\data\yeast_high.cod')
```

O resultado da instrução values acima é:

```
AGAGACAATAACGACGATGATAAC
```

Uma segunda execução da mesma instrução produz a seguinte cadeia:

```
AGAGATAATAACGACGATGACAAC
```

Uma terceira execução da mesma instrução produz a seguinte cadeia com valores aleatórios:

```
AGAGAT AAC AACGAC GACGATAAT
```

Os códons em negrito realçam as diferenças entre as transformações atuais e anteriores.

Os resultados da instrução única values mostram que a função LSPep2ProbNuc escolhe um dos símbolos possíveis com base nas estatísticas anteriores. Isso é diferente da função LSPep2AmbNuc, que utiliza símbolos ambíguos onde há mais conversões possíveis.

A função LSPep2ProbNuc seleciona as conversões mais prováveis para cada símbolo e, em seguida, substitui cada símbolo por uma conversão aleatória do conjunto selecionado anteriormente. Suponha os seguintes dados em uma tabela de freqüências:

Tabela 15. Dados de amostra da tabela de freqüências

Aminoácido	Códon	Freqüência
Ala	GCG	0,17
Ala	GCA	0,13
Ala	GCT	0,17
Ala	GCC	0,53

Suponha que a seqüência de peptídios contenha quatro símbolos "A" (Ala). A função converte A duas vezes para GCC; uma vez para GCG e uma vez para GCT. No entanto, a ordem em que a função produz as conversões é aleatória. A consulta poderia converter o primeiro A para cada uma das conversões do conjunto {GCC, GCC, GCG, GCT}. O resultado é sempre duas ocorrências de GCC, uma ocorrência de GCG e uma ocorrência de GCT na seqüência de DNA de saída. Múltiplas execuções da função na mesma seqüência poderiam retornar seqüências de DNA com os valores intercambiados.

Referência Relacionada:

- "UDF (User-Defined Function) LSPep2ProbNuc" na página 51
- "UDF (User-Defined Function) LSPep2ProbNuc - Mensagens de Erro" na página 53
- "UDF (User-Defined Function) LSPep2AmbNuc - Exemplo" na página 48

UDF (User-Defined Function) LSPep2ProbNuc - Mensagens de Erro

Tabela 16. Mensagens emitidas pela UDF (User-Defined Function) LSPep2ProbNuc

Código de erro	Mensagem	Explicação
SQL0443N	A rotina ″DB2LS.LSPEP2PROBNUC″ (nome específ.″LSPEP2PROBNUC″) retornou um erro SQLSTATE com o texto de diagnóst. ″Seqüência inválida″. SQLSTATE=38608	A seqüência de entrada é inválida.
SQL0443N	A rotina ″DB2LS.LSPEP2PROBNUC″ (nome específico ″LSPEP2PROBNUC″) retornou um erro SQLSTATE com o texto de diagnóstico ″Nenhuma conversão localizada″. SQLSTATE=38610	O arquivo da tabela de freqüência de códons está vazio.
SQL0443N	A rotina ″LSPEP2PROBNUC″ (nome específico ″LSPEP2PROBNUC″) retornou um erro SQLSTATE com o texto de diagnóstico ″Impossível abrir o arquivo de tabelas de conversão″. SQLSTATE=38612	O arquivo não existe.
SQL0443N	A rotina ″DB2LS.LSPEP2PROBNUC″ (nome específico ″LSPEP2PROBNUC″) retornou um erro SQLSTATE com o texto de diagnóstico ″Linha muito longa sendo lida no arquivo″. SQLSTATE=38614	O arquivo contém linhas que são mais longas que o permitido.
SQL0443N	A rotina ″DB2LS.LSPEP2PROBNUC″ (nome específico ″LSPEP2PROBNUC″) retornou um erro SQLSTATE com o texto de diagnóstico ″Formato de arquivo inválido″. SQLSTATE=38615	O formato do arquivo é inválido.
SQL0443N	A rotina ″LSPEP2PROBNUC″ (nome específico ″LSPEP2PROBNUC″) retornou um erro SQLSTATE com o texto de diagnóstico ″Impossível construir a tabela de conversão″. SQLSTATE=38611	O arquivo contém símbolos inválidos

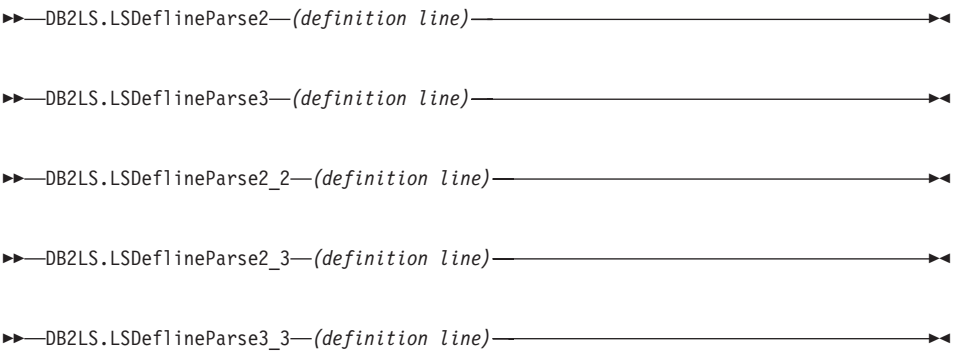
Referência Relacionada:

- “UDF (User-Defined Function) LSPep2ProbNuc” na página 51
- “UDF (User-Defined Function) LSPep2ProbNuc - Exemplo” na página 51

UDFs (User-Defined Functions) de Análise de Defline

As UDFs de análise de Defline analisam os elementos de uma linha de definição, por exemplo, para permitir junções com outras origens de dados em identificadores de seqüências analisados fora do defline ou para avaliar predicados em partes do defline, como `'species = "human"'`. As funções defline abrangem os formatos mais comuns de defline. Os exemplos incluem os elementos de linha de definição que o wrapper do BLAST retorna ou que estão presentes em um arquivo de dados no formato FASTA.

UDFs (User-Defined Functions) LSDeflineParse



definition line
Uma representação válida de cadeia de uma linha de definição no formato FASTA. A cadeia deve ter um tipo de dados VARCHAR e um comprimento real que não seja superior a 1024 bytes.

O nome do esquema é DB2LS.

Cada função LSDeflineParse analisa os campos do NSID (NCBI Standard FASTA Sequence Identifier) e a descrição nas colunas de uma tabela. As linhas de definição, que são definições compostas, são emitidas em várias linhas, cada linha contendo apenas uma única definição de componente.

LSDeflineParse2 analisa um defline que possui um NSID de dois campos. O resultado da função é uma tabela com quatro colunas:

Tabela 17. Descrições de colunas da tabela de resultados da UDF LSDeflineParse2

Nome da Coluna	Descrição
ROWID	Um inteiro que enumera as linhas retornadas da função.
TAG	Um VARCHAR de até três caracteres que representa a marcação NSID.

Tabela 17. Descrições de colunas da tabela de resultados da UDF LSDeflineParse2 (continuação)

Nome da Coluna	Descrição
IDENTIFIER	Um VARCHAR de até 20 caracteres que representa o segundo campo identificador do NSID.
DESCRIPTION	Um VARCHAR de até 1019 caracteres.

LSDeflineParse3 analisa um defline que possui um NSID de três campos. O resultado da função é uma tabela com cinco colunas:

Tabela 18. Descrições de colunas da tabela de resultados da UDF LSDeflineParse3

Nome da Coluna	Descrição
ROWID	Um inteiro que enumera as linhas retornadas da função.
TAG	Um VARCHAR de até três caracteres que representa a marcação NSID.
ACCESSION	Um VARCHAR de até 20 caracteres que representa o segundo campo identificador do NSID.
LOCUS	Um VARCHAR de até 20 caracteres que representa o terceiro campo identificador no NSID.
DESCRIPTION	Um VARCHAR de até 1017 caracteres.

LSDeflineParse2_2 analisa um defline que possui um identificador composto com um par de NSIDs concatenados de dois campos. O resultado da função é uma tabela com seis colunas:

Tabela 19. Descrições de colunas da tabela de resultados da UDF LSDeflineParse2_2

Nome da Coluna	Descrição
ROWID	Um inteiro que enumera as linhas retornadas da função.
TAG1	Um VARCHAR de até três caracteres que representa a marcação NSID do primeiro identificador.
IDENTIFIER1	Um VARCHAR de até 20 caracteres que representa o segundo campo identificador do primeiro NSID.
TAG2	Um VARCHAR de até três caracteres que representa a marcação NSID do primeiro identificador.
IDENTIFIER2	Um VARCHAR de até 20 caracteres que representa o segundo campo identificador do segundo NSID.
DESCRIPTION	Um VARCHAR de até 1015 caracteres.

LSDeflineParse2_3 analisa um defline que possui um identificador composto que consiste em um NSID de dois campos concatenado com um NSID de três campos. A ordem de concatenação no defline de entrada--se o NSID de dois campos vem antes do NSID de três campos, ou vice-versa--não é importante. O resultado da função é uma tabela com sete colunas:

Tabela 20. Descrições de colunas da tabela de resultados da UDF LSDeflineParse2_3

Nome da Coluna	Descrição
ROWID	Um inteiro que enumera as linhas retornadas da função.
TAG1	Um VARCHAR de até três caracteres que representa a marcação NSID do identificador de dois campos.
IDENTIFIER	Um VARCHAR de até 20 caracteres que representa o segundo campo identificador do NSID de dois campos.
TAG2	Um VARCHAR de até três caracteres que representa a marcação NSID do identificador de três campos.
ACCESSION	Um VARCHAR de até 20 caracteres que representa o segundo campo identificador do NSID de três campos.
LOCUS	Um VARCHAR de até 20 caracteres que representa o terceiro campo identificador do NSID de três campos.
DESCRIPTION	Um VARCHAR de até 1013 caracteres.

LSDeflineParse3_3 analisa um defline que possui um identificador composto com um par de NSIDs de três campos. O resultado da função é uma tabela com oito colunas:

Tabela 21. Descrições de colunas da tabela de resultados da UDF LSDeflineParse3_3

Nome da Coluna	Descrição
ROWID	Um inteiro que enumera as linhas retornadas da função.
TAG1	Um VARCHAR de até três caracteres que representa a marcação NSID do primeiro identificador.
ACCESSION1	Um VARCHAR de até 20 caracteres que representa o segundo campo identificador do primeiro NSID.
LOCUS1	Um VARCHAR de até 20 caracteres que representa o terceiro campo identificador do primeiro NSID.
TAG2	Um VARCHAR de até três caracteres que representa a marcação NSID do primeiro identificador.

Tabela 21. Descrições de colunas da tabela de resultados da UDF LSDeflineParse3_3 (continuação)

Nome da Coluna	Descrição
ACCESSION2	Um VARCHAR de até 20 caracteres que representa o segundo campo identificador do segundo NSID.
LOCUS2	Um VARCHAR de até 20 caracteres que representa o terceiro campo identificador do segundo NSID.
DESCRIPTION	Um VARCHAR de até 1014 caracteres.

Referência Relacionada:

• "UDF (User-Defined Function) LSDeflineParse — Exemplos" na página 57

UDF (User-Defined Function) LSDeflineParse — Exemplos

Este tópico contém sete exemplos que mostram como as UDFs LSDeflineParse analisam as linhas de definição nas tabelas de resultados.

O exemplo a seguir de consulta e de tabela de resultados mostra como a UDF LSDeflineParse2 analisa uma linha de definição que contém um NSID de dois campos:

```
select *
from table(DB2LS.LSDeflineParse2(
          '>gi|12346 hypothetical protein 185 –wheat chloroplast')) as t
```

A tabela de resultados contém os seguintes dados:

Tabela 22. Dados dos resultados da UDF LSDeflineParse2

Nome da Coluna	Dados
ROWID	1
TAG	gi
IDENTIFIER	12346
DESCRIPTION	hypothetical protein 185 – wheat chloroplast

O exemplo a seguir de consulta e de tabela de resultados mostra como a UDF LSDeflineParse3 analisa uma linha de definição que contém um NSID de três campos:

```
select *
from table(DB2LS.LSDeflineParse3('
          >gb|U37104|APU37104 Aethia pusilla cytochrome b gene')) as t
```

A tabela de resultados contém os seguintes dados:

Tabela 23. Dados dos resultados da UDF LSDeflineParse3

Nome da Coluna	Dados
ROWID	1
TAG	gb
ACCESSION	U37104
LOCUS	APU37104
DESCRIPTION	Aethia pusilla cytochrome b gene

O exemplo a seguir de consulta e de tabela de resultados mostra como a UDF LSDeflineParse2_2 analisa uma linha de definição com um identificador composto que consiste em um par de NSIDs de 2 campos:

```
select *
from table(DB2LS.LSDeflineParse2_2(
          '>gb|U37104|gim|73401A Aethia pusilla cytochrome b gene')) as t
```

A tabela de resultados contém os seguintes dados:

Tabela 24. Dados dos resultados da UDF LSDeflineParse2_2

Nome da Coluna	Dados
ROWID	1
TAG1	gb
IDENTIFIER1	U37104
TAG2	gim
IDENTIFIER2	73401A
DESCRIPTION	Aethia pusilla cytochrome b gene

O exemplo a seguir de consulta contém uma linha de definição com um identificador composto que consiste em um NSID de 2 campos concatenado com um NSID de 3 campos. O exemplo mostra como a função LSDeflineParse2_3 analisa a linha de definição.

```
select *
from table(DB2LS.LSDeflineParse2_3('
          >gi|12346|gp|CAA44030.1|CHTAHSRA_4
          hypothetical protein 185 — wheat chloroplast')) as t
```

A tabela de resultados contém os seguintes dados:

Tabela 25. Dados dos resultados da UDF LSDeflineParse2_3

Nome da Coluna	Dados
ROWID	1
TAG1	gi
IDENTIFIER	12346
TAG2	gp
ACCESSION	CAA44030.1
LOCUS	CHTAHSRA_4
DESCRIPTION	hypothetical protein 185 – wheat chloroplast

O exemplo a seguir de consulta contém uma linha de definição com um identificador composto que consiste em um NSID de 3 campos concatenado com um NSID de 2 campos. O exemplo mostra como a função LSDeflineParse2_3 analisa a linha de definição.

```
select *
from table(DB2LS.LSDeflineParse2_3('
          >gp|CAA44030.1|CHTAHSRA_4|gi|12346
          hypothetical protein 185 - wheat chloroplast')) as t
```

A tabela de resultados contém os seguintes dados:

Tabela 26. Dados dos resultados da UDF LSDeflineParse2_3

Nome da Coluna	Dados
ROWID	1
TAG1	gi
IDENTIFIER	12346
TAG2	gp
ACCESSION	CAA44030.1
LOCUS	CHTAHSRA_4
DESCRIPTION	hypothetical protein 185 – wheat chloroplast

O exemplo a seguir de consulta e de tabela de resultados mostra como a UDF LSDeflineParse3_3 analisa uma linha de definição que contém um identificador composto com um par de NSIDs de 3 campos:

```
select * from table(DB2LS.LSDeflineParse3_3('
                  >dbj|AAD55586.1|AF055084_1|gp|CAA44030.1|CHTAHSRA_4
                  hypothetical protein 185 – wheat chloroplast')) as t
```

A tabela de resultados contém os seguintes dados:

Tabela 27. Dados dos resultados da UDF LSDeflineParse3_3

Nome da Coluna	Dados
ROWID	1
TAG1	dbj
ACCESSION1	AAD55586.1
LOCUS1	AF055084_1
TAG2	gp
ACCESSION2	CAA44030.1
LOCUS2	CHTAHSRA_4
DESCRIPTION	hypothetical protein 185 – wheat chloroplast

Você pode utilizar qualquer uma das UDFs defline para analisar uma linha de definição composta. O exemplo a seguir de consulta contém uma linha de definição composta com várias definições que são separadas por um caractere Control-A. Você pode localizar esse tipo de linha de definição no número não-redundante do banco de dados de proteínas do NCBI. O exemplo mostra como a função LSDeflineParse2_3 analisa a linha de definição.

```
select *
from table(DB2LS.LSDeflineParse2_3('
          >gi|12346|gp|CAA44030.1|CHTAHSRA_4
          hypothetical protein 185 - wheat chloroplast
     ^Agp|CAA44030.1|CHTAHSRA_4|gi|12346
          hypothetical protein 185 - wheat chloroplast')) as t
```

A tabela de resultados contém os seguintes dados:

Tabela 28. Dados dos resultados da UDF LSDeflineParse2_3

Nome da Coluna	Dados	Dados
ROWID	1	2
TAG1	gi	gi
IDENTIFIER	12346	12346
TAG2	gp	gp
ACCESSION	CAA44030.1	CAA44030.1
LOCUS	CHTAHSRA_4	CHTAHSRA_4
DESCRIPTION	hypothetical protein 185 – wheat chloroplast	hypothetical protein 185 – wheat chloroplast

Referência Relacionada:

- "UDFs (User-Defined Functions) LSDeflineParse" na página 54

UDFs (User-Defined Functions) de Correspondência Generalizada de Padrões

As UDFs (User-Defined Functions) de correspondência generalizada de padrões identificam áreas de interesse em uma determinada cadeia, como uma seqüência de nucleotídios ou peptídios.

UDF (User-Defined Function) LSPatternMatch

```
►►—DB2LS.LSPatternMatch—(input character sequence, pattern)———————————►◄
```

input character sequence
A representação de cadeia de caracteres deve ter um tipo de dados VARCHAR e um comprimento real que não seja superior a 32672 bytes.

pattern
O padrão conforme especificado em qualquer expressão regular Perl válida. A representação de cadeia de caracteres deve ter um tipo de dados VARCHAR e um comprimento real que não seja superior a 32672 bytes.

O nome do esquema é DB2LS.

Você pode utilizar a UDF LSPatternMatch para pesquisar a seqüência de entrada de nucleotídios ou peptídios para um padrão especificado.

O resultado da função é um inteiro que representa a posição da primeira correspondência do padrão na seqüência. A função retorna um valor de zero se não houver correspondência.

Se houver padrões gravados com a sintaxe PROSITE, eles poderão ser convertidos em sintaxe Perl com a UDF LSPrositePattern. Depois, você poderá utilizar a sintaxe convertida com a UDF LSPatternMatch.

Referência Relacionada:
- "UDF (User-Defined Function) LSPatternMatch – Exemplo" na página 61
- "UDF (User-Defined Function) LSPrositePattern" na página 64

UDF (User-Defined Function) LSPatternMatch – Exemplo

No exemplo a seguir, procure a posição inicial da cadeia que corresponde a "coward", "cowage", "cowboy" ou "cowl".

```
values DB2LS.LSPatternMatch('joe the cowboy is next', 'cow(ard|age|boy|l)')
```

A função pesquisa por caracteres e, nesse exemplo, retorna um valor nove. A cadeia “cowboy” começa na posição nove, supondo que a primeira posição seja um.

No exemplo a seguir, procure a posição inicial da cadeia que corresponde a “not ” ou “non”:

```
values DB2LS.LSPatternMatch('match not and non but
                             no match for no or none', 'no[tn] ')
```

A função pesquisa por caracteres e, nesse exemplo, retorna um valor sete. A cadeia “not ” começa na posição sete, supondo que a primeira posição seja um.

LSPatternMatch é útil nas instruções select para filtrar os resultados utilizando a sintaxe PERL, que é uma sintaxe mais poderosa que a instrução SQL LIKE. No exemplo a seguir, utilize LSPatternMatch em uma saída blast para filtrar os genes que correspondem a um determinado padrão:

```
SELECT BlastOutput.*
 FROM BlastOutput
 WHERE db2ls.LSPatternMatch(HSP_H_Seq, 'F[GSTV]PRL') > 0;
```

Se você estiver mais familiarizado com a sintaxe PROSITE, poderá utilizar a função LSPrositePattern com a consulta acima. Altere a consulta para o seguinte:

```
SELECT BlastOutput.*
  FROM BlastOutput
  WHERE db2ls.LSPatternMatch(HSP_H_Seq,
          db2ls.LSPrositePattern('F-[GSTV]-P-R-L.') ) > 0;
```

As funções de correspondência de padrões são úteis para pesquisar outros tipos de texto, bem como as seqüências de nucleotídios ou peptídios. Considere utilizar a instrução SQL LIKE quando houver probabilidade de problema no desempenho.

O exemplo a seguir mostra uma consulta que filtra alinhamentos hsp BLAST com base nos temas de proteína localizados na linha de assunto ou destino do alinhamento. O exemplo é adaptado de `Zhang,Z., Schaffer,A.A., Miller,W., Madden,T.L., Lipman,D.J., Koonin,E.V. and Altschul,S.F. (1998) Protein sequence similarity searches using patterns as seeds. Nucl. Acids Res., 26, 3896-3990.`

A consulta a seguir retorna apenas os alinhamentos em que a seqüência de assuntos inclui o domínio ATPase de loop-P [GA]xxxxGK[ST]. A consulta utiliza o CED4, o regulador *Caenorhabditis elegans* de morte celular, como uma seqüência de consultas junto ao banco de dados não-redundante da seqüência de proteínas do NCBI. O banco de dados recupera a seqüência de consulta blast a partir da conversão do recurso CDS da entrada X69016 do GenBank.

```
SELECT HSP_Q_Seq, HSP_Midline, HSP_H_Seq
FROM BlastP b, GBseq gs, gbfeat gf, gbqual gq
WHERE gs.PRIMARYACCESSION = 'X69016' and
                  gs.sequencekey = gf.sequencekey and
                  gf.featurejoinkey = gq.featurejoinkey and
                  gf.FeatureKey = 'CDS' and
                  gq.QualifierName = 'translation' and
                  gq.QualifierValue = b.BlastSeq and
                  db2ls.LSPatternMatch(HSP_H_Seq,
                    db2ls.LSPrositePattern('[GA]-x(4)-G-K-[ST].') ) > 0;
```

Você pode utilizar a próxima consulta de exemplo para localizar HSPs em uma seqüência genômica que contém SNPs (Single Nucleotide Polymorphisms) putativos em relação a uma seqüência canônica de consulta. Adaptado de Extending traditional query-based integration approaches for functional characterization of post-genomic data. (2001) Barbara A Eckman, Anthony S Kosky, and Leonardo A Laroco Jr. *Bioinformatics* 17(7), 587-601.

A consulta utiliza a correspondência de padrões na linha mediana hsp blast para localizar um padrão de ≥20 correspondências perfeitas, seguidas por uma única correspondência, seguida por ≥20 correspondências perfeitas. Ou seja, 20 caracteres "|", um único espaço e, em seguida, 20 caracteres "|" na linha mediana do alinhamento.

Esse exemplo também mostra o uso da UDF (LSPatternMatch) em cadeias que não são seqüências de nucleotídios ou peptídios.

```
SELECT   HSP_Info, HSP_Midline, HSP_H_Seq
FROM BlastOutput
WHERE db2ls.LSPatternMatch(HSP_Midline, '\|{20} \|{20}') > 0;
```

Você pode regravar a consulta anterior como:

```
SELECT HSP_Info, HSP_Midline, HSP_H_Seq, func.Position, func.Match
FROM BlastOutput,
     table( select * as c from table(
            LSMultiMatch(HSP_Midline, '\|{20} \|{20}') )
     as f) as func
```

Essa segunda consulta retornará as linhas blast que possuem uma correspondência juntamente com a cadeia correspondida e sua posição na seqüência.

BlastOutput é uma exibição sobre um pseudônimo BlastN.

Referência Relacionada:

- "UDF (User-Defined Function) LSPrositePattern - Exemplo" na página 64
- "UDF (User-Defined Function) LSPatternMatch" na página 61
- "UDF (User-Defined Function) LSPrositePattern" na página 64

UDF (User-Defined Function) LSPrositePattern

```
►►—DB2LS.LSPrositePattern—(pattern)———————————————————————————►◄
```

pattern
A sintaxe de correspondência de padrões especificada pela sintaxe Prosite. A representação de cadeia de caracteres deve ter um tipo de dados VARCHAR e um comprimento real que não seja superior a 32672 bytes.

O nome do esquema é DB2LS.

Utilize a UDF LSPrositePattern para converter da sintaxe PROSITE para a sintaxe PERL. Você poderá utilizar a sintaxe convertida com as UDFs LSPatternMatch, LSMultiMatch e LSMultiMatch3.

O resultado da função é uma cadeia de caracteres que representa uma expressão regular na sintaxe Perl. A representação de cadeia de caracteres deve ter um tipo de dados VARCHAR e um comprimento real que não seja superior a 32672 bytes.

Referência Relacionada:
- "UDF (User-Defined Function) LSPrositePattern - Exemplo" na página 64
- "UDF (User-Defined Function) LSPatternMatch" na página 61

UDF (User-Defined Function) LSPrositePattern - Exemplo

No exemplo a seguir, converta um padrão da sintaxe PROSITE para a sintaxe PERL.

```
values db2ls.LSPrositePattern('[AC]-x-V-x(4)-{ED}.');
```

A função converte o padrão de entrada na sintaxe PROSITE em um padrão equivalente na sintaxe Perl, conforme mostrado no exemplo a seguir:

```
[AC].V.{4}[^ED]
```

O próximo exemplo converte um outro padrão de sintaxe de PROSITE na sintaxe PERL:

```
values db2ls.LSPrositePattern('<A-x-[ST](2)-x(0,1)-V.');
```

A função converte a cadeia da sintaxe PROSITE com base no padrão de entrada e retorna o seguinte:

```
\AA.[ST]{2}.{0,1}V
```

O próximo exemplo converte o padrão correspondente à entrada do banco de dados PROSITE com o número de ID PS01205 em um padrão PERL que é utilizado como entrada pelas funções de correspondência de padrões.

```
values db2ls.LSPrositePattern('R-P-L-[IV]-x-[NS]-F-G-S-[CA]-T-C-P-x-F.')
```

O resultado dessa consulta é:

```
RPL[IV].[NS]FGS[CA]TCP.F
```

O próximo exemplo mostra como você pode utilizar a função em uma consulta. A consulta imprime apenas as seqüências que correspondem ao padrão PROSITE especificado.

```
SELECT H_Accession, HSP_Info, HSP_H_Seq
FROM BlastOutput
WHERE db2ls.LSPatternMatch( HSP_H_Seq,
  db2ls.LSPrositePattern('R-P-L-[IV]-x-[NS]-F-G-S-[CA]-T-C-P-x-F.') ) > 0;
```

O próximo exemplo converte o padrão correspondente à entrada PROSITE cujo ID é PS00261:

```
values db2ls.LSPrositePattern('C-[STAGM]-G-[HFYL]-C-x-[ST].')
```

O resultado dessa consulta é:

```
C[STAGM]G[HFYL]C.[ST]
```

Referência Relacionada:

- "UDF (User-Defined Function) LSPatternMatch – Exemplo" na página 61
- "UDF (User-Defined Function) LSPrositePattern" na página 64

Suporte a Expressões Regulares

O suporte a expressões regulares é fornecido pelo pacote de biblioteca PCRE, que é um software de código-fonte aberto, escrito por Philip Hazel e de direitos autorais da University of Cambridge, Inglaterra.

O código-fonte pode ser encontrado em ftp://ftp.csx.cam.ac.uk/pub/software/programming/pcre/.

UDFs (User-Defined Functions) GeneWise

A UDF GeneWise alinha uma seqüência de proteínas com uma seqüência genômica.

GeneWise é um componente freqüentemente utilizado que alinha uma seqüência de proteínas com uma seqüência genômica de DNAs, permitindo íntrons e erros de frameshifting.

Vinculando à GeneWise

Este tópico descreve o procedimento para vincular à biblioteca GeneWise.

Procedimento:

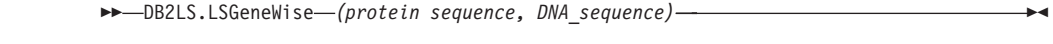
1. Faça download do pacote Wise2, versão 2.1.20c, em http://www.ebi.ac.uk/Wise2.
2. Expanda o archive para uma pasta de sua preferência.
3. Compile o pacote com o suporte para pthread. Para obter informações adicionais sobre essa etapa, consulte a documentação do Wise2.
4. Execute **make api** em seu diretório raiz.
5. Defina a variável de ambiente WISE2_HOME para apontar para o diretório raiz do pacote Wise2.
6. Defina a variável WISECONFIGDIR no arquivo sqllib/cfg/db2dj.ini para apontar para o subdiretório wisecfg.

   Por exemplo, se o pacote Wise2 estiver instalado em /usr/wise2.1.20c/, adicione WISECONFIGDIR=/usr/wise2.1.20c/wisecfg/ ao arquivo db2dj.ini.
7. Execute **djxlinkLSGeneWise** a partir do diretório sqllib/bin e verifique sua saída.
8. Verifique o djxlinkLSGeneWise.out no diretório sqllib/lib.
9. Se nenhum erro foi relatado, a biblioteca foi construída com êxito.

Referência Relacionada:

• "UDF (User-Defined Function) LSGeneWise" na página 66

UDF (User-Defined Function) LSGeneWise



protein sequence
Uma representação válida de cadeia de caracteres que descreve uma seqüência de peptídios. A representação de cadeia de caracteres deve ter um tipo de dados VARCHAR e um comprimento real que não seja superior a 32672 bytes.

DNA_sequence
Uma representação válida de cadeia de caracteres que descreve uma seqüência de nucleotídios. A representação de cadeia de caracteres deve ter um tipo de dados VARCHAR e um comprimento real que não seja superior a 32672 bytes.

A Tabela 29 na página 67 mostra a tabela de saída de uma linha que a função LSGeneWise retorna.

Tabela 29. Nomes de colunas, tipos e descrições para a tabela de saída da função LSGeneWise

Nome da Coluna	Tipo	Descrição
PROTEIN_OFFSET	INTEGER	Representa o deslocamento inicial na seqüência de entrada de proteínas em que um alinhamento foi localizado.
DNA_OFFSET	INTEGER	Representa o deslocamento inicial na seqüência de entrada de DNAs em que um alinhamento foi localizado.
PROTEIN	VARCHAR(32672)	Um fragmento da seqüência de entrada que representa a seqüência alinhada.
SIMILARITY	VARCHAR(32672)	Mostra a correspondência entre as seqüências de proteínas e de dnas. As correspondências perfeitas são marcadas com a letra correspondente do símbolo. As correspondências não-perfeitas com um score positivo são indicadas com o sinal ″+″ e as incompatibilidades são indicadas com um espaço.
TRANSLATED _DNA	VARCHAR(32672)	A seqüência de DNAs convertidos. A seqüência poderia conter traços e símbolos especiais, como exclusões e íntrons.
DNA	VARCHAR(32672)	A seqüência de DNAs com marcadores especiais, como frameshifting e íntrons.

A correspondência entre a saída do programa GeneWise e a saída da UDF LSGeneWise é a seguinte:

- os deslocamentos de proteína e dna impressos pelo programa GeneWise correspondem às colunas PROTEIN_OFFSET e DNA_OFFSET.
- a seqüência de proteínas impressa na primeira linha por GeneWise corresponde à coluna PROTEIN.
- a linha de similaridade, a segunda linha na saída do GeneWise corresponde à coluna SIMILARITY.
- a terceira linha na saída do GeneWise corresponde à coluna TRANSLATED_DNA.
- a quarta, quinta e sexta linhas da saída GeneWise são combinadas, lendo-as verticalmente, para a coluna DNA.

Utilize a UDF LSGeneWise para alinhar uma seqüência de proteínas com uma seqüência genômica de DNAs, permitindo íntrons e erros de frameshifting.

Para obter informações adicionais sobre a saída UDF LSGeneWise, consulte http://www.ebi.ac.uk/Wise2.

Tarefas Relacionadas:

- "Vinculando à GeneWise" na página 66

Referência Relacionada:

- "UDF (User-Defined Function) LSGeneWise – Exemplo" na página 68

UDF (User-Defined Function) LSGeneWise – Exemplo

O exemplo a seguir mostra uma consulta que utiliza a UDF LSGeneWise e os dados resultantes.

```
select protein_offset, dna_offset, protein, similarity, translated_dna, dna
from table( db2ls.LSGeneWise( '
             VEPKRAVPRQDIDSPNAGATVKKLFVGALKDDHDEQSIRDYFQHFGNIVDINIVIDKETGK
             KRGFAFVEFDDYDPVDKVVLQKQHQLNGKMVDVKKALPKQNDQQGGGGGRGGPGGRAGGNR
             GNMGGGNYGNQNGGGNWNNGGNNWGNNR',
            'CACTTAACTGTGAAAGATATTTGTTGGTGGCATTAAAGAAGACACTGAAGAACATCACCTAAG
             AGATTATTTTGAACAGTATGGAAAAATTGAAGTGATTGAAATCATGACTGACCGAGGCAGTGG
             CAAGAAAAGGGGCTTTGCCTTRGTAACCTTTGACGACCATGACTCCGTGGATAAGATTGTCAT
             TCAGAAATACCATACTGTGAATGGCCACAACTGTGAAGTTAGAAAAGCCCTGTCAAAGCAAGA
             GATGGCTAGTGCTTCATCCAGCCAAAGAGGTCGAAGTGGTTCTGGAAACTTTGGTGGTGGTCG
             TGGAGGTGGTTTCGGTGGGAATGACAACTTCGGTCGTGGAGGAAACTTCAGTGGTCGTGGTYG
             CTTTGGTGGCAGCCGTGGTGGTGGTGGATATGGTGGC' ) ) as f;
```

Tabela 30. Tabela de resultados

Coluna	Dados
PROTEIN_OFFSET	23
DNA_OFFSET	14
PROTEIN	KLFVGALKDDHDEQSIRDYFQHFGNIVDINIVIDKET GKKRGFAFVEFDDYDPVDKVVLQKQHQLNGKMVD VKKALPKQNDQQGGGGGRGGPGGRAGGNRGNMGG GNYGNQNGGGNWNNGGN
SIMILARITY	K+FVG +K+D +E +RDYF+ +G I I I+ D+ +GKKRGFA+V FDD+D VDK+V+QK H +NG +V+KAL KQ RG G GN+GGG G G N+ GGN
TRANSLATED _DNA	KIFVGGIKEDTEEHHLRDYFEQYGKIEVIEIMTDRGSGK KRGFAxVTFDDHDSVDKIVIQKYHTVNGHNCEVRKAL SKQEMASASSSQRGRSGS------ GNFGGGRGGGFGGNDNFGRGGN
DNA	aagatatttgttggtggcattaaagaagacactgaagaacatcacctaagagat..

Tarefas Relacionadas:

- "Vinculando à GeneWise" na página 66

Referência Relacionada:

- "UDF (User-Defined Function) LSGeneWise" na página 66

UDFs (User-Defined Functions) de Temas

As UDFs (User-Defined Functions) de temas correspondem padrões em seqüências de nucleotídios ou aminoácidos.

UDF (User-Defined Function) LSBarCode

```
►►—DB2LS.LSBarCode—(input string sequence)—————————————————————►◄
```

input string sequence
Uma cadeia válida de caracteres que representa um elemento HSP entre dois fragmentos de seqüência. A representação de cadeia de caracteres deve ter um tipo de dados VARCHAR e um comprimento real que não seja superior a 32672 bytes.

O nome do esquema é DB2LS.

Utilize a UDF LSBarCode para utilizar uma seqüência como entrada e gerar uma outra seqüência substituindo cada caractere, exceto espaços e sinais de mais com o símbolo de barra vertical (|).

O resultado da função é uma seqüência variável de caracteres que representa uma seqüência de código de barras.

Referência Relacionada:

- "UDF (User-Defined Function) LSBarCode — Exemplo" na página 69
- "UDF (User-Defined Function) LSMultiMatch" na página 71
- "UDF (User-Defined Function) LSMultiMatch3" na página 72

UDF (User-Defined Function) LSBarCode — Exemplo

Esse exemplo cria um código de barras a partir de uma seqüência de cadeias:

```
values db2ls.LSBarCode(
   'MDY +G++L  GN  ++ +PASLTK+MT YVV +A+ + +I   D+VTVG+DAWA  NP ')
```

O resultado da instrução values acima é:

```
||| +|++|  ||  ++ +||||||+|| ||| +|+ + +|   |+||||+||||  ||
```

O próximo exemplo mostra uma utilização mais realista dessa função. Suponha que um pesquisador que esteja executando uma pesquisa do BLAST deseje retornar apenas alinhamentos HSP contendo menos de 25% de prolines entre suas correspondências perfeitas. Esse exemplo utiliza a função para calcular a porcentagem de prolines (símbolo 'P') entre as perfeitas correspondências de um alinhamento retornado pelo BLAST. Observe que esse exemplo também chama a UDF LSMultiMatch3. A consulta utiliza a função de correspondência para localizar as correspondências perfeitas. Ela é utilizada

em conjunto com a função LSBarCode nessa consulta porque o Blast nem sempre retorna uma seqüência de barras (“|”) em um alinhamento. O exemplo a seguir mostra isso:

```
Consulta:       MDYTTGQILTAGNEHQQRNPASLTKLMTGYVVDRAIDSHRITPDDIVTVGRDAWAKDNPV
Alinhamento:   MDY +G++L  GN  ++ +PASLTK+MT YVV +A+ + +I   D+VTVG+DAWA  NP
Destino:      MDYASGKVLAEGNADEKLDPASLTKIMTSYVVGQALKADKIKLTDMVTVGKDAWATGNPA
```

Para assegurar-se de que a saída esteja alinhada com a seqüência correta de barras, utilize a função LSBarCode. A função substitui todos os caracteres, exceto espaços e sinais de mais com uma barra vertical.

```
SELECT BlastOutput.* , float( p )/ float( m ) AS percent_prolines
  FROM
  BlastOutput b,
  table(SELECT COUNT(*) AS p FROM table(
          db2ls.LSMultiMatch3(
            b.HSP_Q_Seq, 'P',
            db2ls.LSBarCode(b.HSP_Midline), '\|',
            b.HSP_H_Seq, 'P')
          ) AS f
        ) AS y,
  table(SELECT COUNT(*) AS m FROM table(
          db2ls.LSMultiMatch3(
            b.HSP_Q_Seq, '.',
            db2ls.LSBarCode(b.HSP_Midline), '\|',
            b.HSP_H_Seq, '.')
          ) AS f
        ) AS z
WHERE float(p) / float(m) < 0.25;
```

Nessa consulta, BlastOutput é realmente uma exibição sobre um pseudônimo do Blast. A consulta utiliza a função LSMultiMatch3 para retornar as correspondências perfeitas no alinhamento. O primeiro uso retorna as correspondências perfeitas para o símbolo “P”, o segundo retorna todas as correspondências perfeitas. Uma linha da tabela de resultados é mostrada na Tabela 31.

Tabela 31. Linha de amostra dos resultados

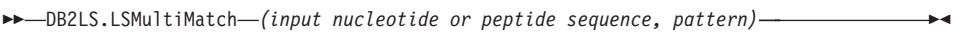
HSP_Q_SEQ	HSP_H_SEQ	HSP_INFO	PERCENT_ PROLINES
NIWDFMQGN...	NIWDFMQGN...	Identid.= 80/80 (100%), Posit.= 80/80 (100%), Interv.= 0/80 (0%)	+2.50000000 000000E-002

A consulta anterior foi adaptada de Extending traditional query-based integration approaches for functional characterization of post-genomic data. (2001) Barbara A Eckman, Anthony S Kosky and Leonardo A Laroco Jr. *Bioinformatics* 17(7), 587-601.

Referência Relacionada:

- "UDF (User-Defined Function) LSMultiMatch3 – Exemplo" na página 73
- "UDF (User-Defined Function) LSBarCode" na página 69

UDF (User-Defined Function) LSMultiMatch



input nucleotide or peptide sequence
Uma representação válida de cadeia de caracteres que descreve uma seqüência de nucleotídios ou peptídios. A representação de cadeia de caracteres deve ter um tipo de dados VARCHAR e um comprimento real que não seja superior a 32672 bytes.

pattern
A gramática de correspondência de padrões especificadas pela linguagem Perl. A representação de cadeia de caracteres deve ter um tipo de dados VARCHAR e um comprimento real que não seja superior a 32672 bytes.

O nome do esquema é DB2LS.

Utilize a UDF LSMultiMatch para retornar uma tabela para cada correspondência que não seja sobreposta na seqüência de entrada. Cada tabela consiste em uma posição inicial e o fragmento de seqüência de correspondência.

O resultado da função é uma tabela com duas colunas. A primeira coluna é um inteiro que representa a posição inicial de uma correspondência do padrão na seqüência. A segunda coluna é o fragmento de seqüência de correspondências.

Referência Relacionada:

- "UDF (User-Defined Function) LSMultiMatch - Exemplo" na página 71
- "UDF (User-Defined Function) LSBarCode" na página 69
- "UDF (User-Defined Function) LSMultiMatch3" na página 72

UDF (User-Defined Function) LSMultiMatch - Exemplo

Esse exemplo procura a posição e os fragmentos de correspondência para todas as correspondências sem sobreposição obtidas da entrada.

```
SELECT position, match FROM table
  (LSMultiMatch('match not and non but no match for no or none',
      'no[tn] ')) as f
```

A consulta retorna uma tabela baseada nessa instrução select que mostra os resultados das correspondências:

Tabela 32. Resultado do LSMultiMatch retornando múltiplas linhas

POSITION	MATCH
7	not
15	non

LSMultiMatch retorna a posição e a cadeia correspondida para todas as correspondências. O exemplo a seguir pesquisa no Entrez Nucleotide as entradas de seqüência que contêm um determinado tema. A consulta imprime os identificadores de seqüência e as seqüências correspondidas. Os subpadrões ".{0,9}" no início e no final devem corresponder até nove caracteres antes e após a seqüência. A consulta também imprime esses caracteres.

```
select SequenceKey, Position, Match from GBSeq,
  table(db2ls.LSMultiMatch(Sequence, '.{0,9}(ATG|CGC)ACGGGC.{0,9}') )
  as fmatch
  WHERE entrez.contains(KeywordList,
        'Na/K/2Cl cotransporter AND nkcc1 gene') = 1;
```

O resultado dessa consulta é o seguinte:

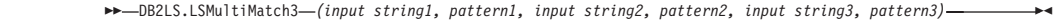
Tabela 33. Pesquisar dados do Entrez

SEQUENCEKEY	POSITION	MATCH
N02B59AE0.04DD4E84	1	TGCTTGGTGATGACGGGCTACCCCAAC
N02B59AE0.04DD4E84	91	GGCCATGTTCGCACGGGCTCCAGAAGG
N02B59AE0.04DC5EF4	1	TGCTTGGTGATGACGGGCTACCCCAAC
N02B59AE0.04DC5EF4	91	GGCCATGTTCGCACGGGCTCCAGAAGG

Referência Relacionada:

- "UDF (User-Defined Function) LSMultiMatch" na página 71
- "UDF (User-Defined Function) LSBarCode" na página 69
- "UDF (User-Defined Function) LSMultiMatch3" na página 72

UDF (User-Defined Function) LSMultiMatch3



input strings

Uma representação válida de cadeia de caracteres que descreve uma seqüência de nucleotídios ou peptídios ou uma cadeia HSP_Midline

de um alinhamento blast. A representação de cadeia de caracteres deve ter um tipo de dados VARCHAR e um comprimento real que não seja superior a 32672 bytes.

pattern
A gramática de correspondência de padrões especificadas pela linguagem Perl. A representação de cadeia de caracteres deve ter um tipo de dados VARCHAR e um comprimento real que não seja superior a 32672 bytes.

O nome do esquema é DB2LS.

Utilize a UDF LSMultiMatch3 para inserir três padrões e três cadeias e retornar quaisquer posições em que todas as três cadeias correspondam a seus respectivos padrões. Você pode utilizar essa UDF para executar uma correspondência de padrões em um alinhamento.

O resultado da função é uma tabela com quatro colunas. A primeira coluna é um inteiro que representa a posição inicial de uma correspondência do padrão em todas as seqüências. A função ancora todas as cadeias juntas na primeira posição. A segunda, terceira e quarta colunas são os fragmentos da seqüência de correspondências.

Referência Relacionada:
- "UDF (User-Defined Function) LSMultiMatch3 – Exemplo" na página 73
- "UDF (User-Defined Function) LSMultiMatch" na página 71
- "UDF (User-Defined Function) LSBarCode" na página 69

UDF (User-Defined Function) LSMultiMatch3 – Exemplo

O exemplo a seguir utiliza a função para calcular a porcentagem de um determinado símbolo de aminoácido entre as correspondências perfeitas retornadas pelo Blast. Observe que esse exemplo também chama a UDF LSBarCode. A consulta precisa dela porque o Blast nem sempre retorna uma seqüência de barras ("|") em um alinhamento. O exemplo a seguir ilustra isso:

```
Consulta:       MDYTTGQILTAGNEHQQRNPASLTKLMTGYVVDRAIDSHRITPDDIVTVGRDAWAKDNPV
Alinhamento:   MDY +G++L  GN  ++ +PASLTK+MT YVV +A+ + +I   D+VTVG+DAWA  NP
Destino:      MDYASGKVLAEGNADEKLDPASLTKIMTSYVVGQALKADKIKLTDMVTVGKDAWATGNPA
```

Para assegurar-se de que a saída esteja alinhada com a seqüência correta de barras, utilize a função LSBarCode para converter a seqüência. A função substitui todos os caracteres sem espaço e sem "+" por uma barra vertical.

```
SELECT BlastOutput.* , float( p )/ float( m ) AS percent_prolines
FROM
  BlastOutput b,
  table(SELECT COUNT(*) AS p FROM table(
```

```
          db2ls.LSMultiMatch3(
            b.HSP_Q_Seq, 'P',
            db2ls.LSBarCode(b.HSP_Midline), '\|',
            b.HSP_H_Seq, 'P')
          ) AS f
        ) AS y,
  table(SELECT COUNT(*) AS m FROM table(
          db2ls.LSMultiMatch3(
            b.HSP_Q_Seq, '.',
            db2ls.LSBarCode(b.HSP_Midline), '\|',
            b.HSP_H_Seq, '.')
          ) AS f
        ) AS z
WHERE float(p) / float(m) < 0.25;
```

Nessa consulta, BlastOutput é uma exibição sobre um select do Blast. A consulta utiliza a função LSMultiMatch3 para retornar as correspondências perfeitas no alinhamento. O primeiro uso retorna as correspondências perfeitas para o símbolo "P", o segundo retorna todas as correspondências perfeitas. Uma linha da tabela de resultados é mostrada na Tabela 34.

Tabela 34. Linha de amostra de resultados

HSP_Q_SEQ	HSP_H_SEQ	HSP_INFO	PERCENT_ PROLINES
NIWDFMQG...	NIWDFMQG...	Identid.= 80/80 (100%), Posit.= 80/80 (100%), Interv.= 0/80 (0%)	+2.50000000 000000E-002

A consulta anterior foi adaptada de `Extending traditional query-based integration approaches for functional characterization of post-genomic data. (2001) Barbara A Eckman, Anthony S Kosky and Leonardo A Laroco Jr.` *`Bioinformatics`* `17(7), 587-601.`

O exemplo a seguir procura três padrões separados em três fragmentos separados de cadeia:

```
SELECT position, match_1, match_2, match_3
  FROM table(db2ls.LSMultiMatch3('zaza', 'a', 'abab',
             'b', 'bcbc', 'c')) as f
```

Retorna as posições e as cadeias de correspondência para todas as correspondências, conforme mostrado na tabela a seguir:

Tabela 35. Resultado de uma multicorrespondência utilizando três entradas

POSITION	MATCH_1	MATCH_2	MATCH_3
2	a	b	c
4	a	b	c

O próximo exemplo localiza três padrões separados dentro de três fragmentos separados de cadeia:

```
SELECT position, match_1, match_2, match_3
  FROM table
  (LSMultiMatch3('cbccbbcccbbbccccbbbbccccc','c{1,3}b{1,3}c{1,3}',
   'abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyz',
   '.','01234567890123456789012345678901234567890123456789','\d')) as f
```

Os resultados estão na tabela a seguir:

Tabela 36. Resultado de uma multicorrespondência utilizando três entradas

POSITION	MATCH_1	MATCH_2	MATCH_3
1	cbcc	a	0
7	cccbbbccc	g	6

Referência Relacionada:

- "UDF (User-Defined Function) LSBarCode — Exemplo" na página 69
- "UDF (User-Defined Function) LSBarCode" na página 69
- "UDF (User-Defined Function) LSMultiMatch3" na página 72

UDFs (User-Defined Functions) de Reversão

As UDFs (User-Defined Functions) de reversão revertem uma seqüência de nucleotídios ou aminoácidos.

UDF (User-Defined Function) LSRevComp

input nucleotide sequence
Uma representação válida de cadeia de caracteres que descreve uma seqüência de nucleotídios. A seqüência pode conter códigos de ambigüidade IUPAC. Uma representação de cadeia de caracteres deve ter um tipo de dados VARCHAR e um comprimento real que não seja superior a 32672 bytes.

O nome do esquema é DB2LS.

O resultado da função é uma cadeia de caracteres com um tipo de dados VARCHAR e um comprimento real que não seja maior que 32672 bytes representando o complemento reverso da seqüência de nucleotídios.

Referência Relacionada:

- "UDF (User-Defined Function) LSRevComp—Exemplo" na página 76
- "UDF (User-Defined Function) LSRevNuc" na página 77
- "UDF (User-Defined Function) LSRevPep" na página 78

UDF (User-Defined Function) LSRevComp—Exemplo

Você pode utilizar a função LSRevComp em uma instrução SQL onde quer que utilize uma função interna que aceite uma seqüência de nucleotídios. Por exemplo:

```
SELECT DB2LS.LSRevComp(:NucSeq) FROM SYSDUMMY1;
```

Esse exemplo utiliza a função para retornar o complemento reverso da seqüência de entrada que é fornecida de uma variável do host.

Se você utilizar uma cadeia inválida, ou um tipo inválido de dados, receberá a seguinte mensagem de erro:

```
SQL0443N A rotina "DB2LS.LSREVCOMP" (nome específico "LSREVCOMP") retornou
um erro SQLSTATE com o texto de diagnóst. "Seqüência inválida". SQLSTATE=38608
```

Ocorre uma exceção quando o alfabeto de entrada não está correto.

O exemplo a seguir mostra como a UDF LSRevComp funciona em uma consulta:

```
SELECT HSP_H_Seq, db2ls.LSRevComp(HSP_H_Seq) as REV_HSP_H_Seq
FROM BlastN
WHERE BlastSeq='ccgctagtattggtcaatcttttgatatccaccgaa'
```

Os resultados da consulta são mostrados abaixo:

```
HSP_H_SEQ                      REV_HSP_H_SEQ
------------------------------ ----------------------------

AGTATTGGTCAATCTTTTGAT          ATCAAAAGATTGACCAATACT

TGGTCAATCTTTTGATA              TATCAAAAGATTGACCA

TTGGCCAATCTTTTGATATCC          GGATATCAAAAGATTGGCCAA

TCAATCTTTTGATATCC              GGATATCAAAAGATTGA

GGATATCAAAAGATTGA              TCAATCTTTTGATATCC


5 reg.(s) selecionado(s).
```

Você pode utilizar a função reversa juntamente com outras UDFs de biociência para converter o complemento reverso de uma seqüência de nucleotídios, como no exemplo a seguir:

```
values db2ls.LSNuc2Pep(
       db2ls.LSRevComp('TTTTTCTTATTGTCTTCCTCATCGTATTTCTTATGTTGCTGATGT'))
```

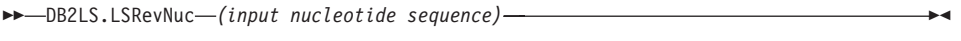
A consulta retorna o seguinte:

```
TSAT*EIR*GRQ*EK
```

Referência Relacionada:

- "UDF (User-Defined Function) LSRevComp" na página 75

UDF (User-Defined Function) LSRevNuc



input nucleotide sequence

Uma representação válida de cadeia de caracteres que descreve uma seqüência de nucleotídios. Uma representação de cadeia de caracteres deve ter um tipo de dados VARCHAR e um comprimento real que não seja superior a 32672 bytes. A seqüência de nucleotídios deve ser parte ou ser totalmente do alfabeto DNA.

O nome do esquema é DB2LS.

O resultado da função é uma cadeia de caracteres com um tipo de dados VARCHAR e um comprimento real que não seja maior que 32672 bytes representando a ordem reversa da seqüência de nucleotídios.

Referência Relacionada:

- "UDF (User-Defined Function) LSRevNuc - Exemplo" na página 77
- "UDF (User-Defined Function) LSRevComp" na página 75
- "UDF (User-Defined Function) LSRevPep" na página 78

UDF (User-Defined Function) LSRevNuc - Exemplo

Você pode utilizar a função LSRevNuc em uma instrução SQL onde quer que utilize uma função interna que aceite uma seqüência de nucleotídios. Por exemplo:

```
SELECT DB2LS.LSRevNuc(:NucSeq) FROM SYSDUMMY1;
```

Esse exemplo utiliza a função para reverter os dados de entrada que são fornecidos a partir de uma variável do host.

Se você utilizar uma cadeia inválida, ou um tipo inválido de dados, receberá a seguinte mensagem de erro:

```
SQL0443N A rotina "DB2LS.LSREVNUC" (nome específico "LSREVNUC") retornou
um erro SQLSTATE com o texto de diagnóst. "Seqüência inválida". SQLSTATE=38608
```

O exemplo a seguir mostra a utilização da UDF LSRevNuc em uma consulta.

```
SELECT HSP_H_Seq, db2ls.LSRevNuc(HSP_H_Seq) as REV_HSP_H_Seq
FROM BlastN
WHERE BlastSeq='gtaatacgtaggggctagcgcgggcaaactgaagataaagc'
```

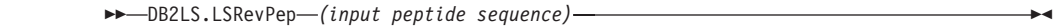
A tabela de resultados a seguir mostra as seqüências revertidas de nucleotídio que a consulta retorna:

```
HSP_H_SEQ                                          REV_HSP_H_SEQ

-------------------------------------------------- -------------------------
CGCGGGCAAACTGAAGATAAAGC                            CGAAATAGAAGTCAAACGGGCGC

GCGCTAGCCCCCTACGTATTAC                             CATTATGCATCCCCCGATCGCG

GTAATACGTAGGGGGCTAGCG                              GCGATCGGGGGATGCATAATG

GTAATACGTAGGGGGCTAGCG                              GCGATCGGGGGATGCATAATG

GTAATACGTAGGGGGCTAGCG                              GCGATCGGGGGATGCATAATG


5 reg.(s) selecionado(s).
```

Referência Relacionada:

- "UDF (User-Defined Function) LSRevNuc" na página 77

UDF (User-Defined Function) LSRevPep



input peptide sequence
Uma representação válida de cadeia de caracteres que descreve uma seqüência de peptídios. Uma representação de cadeia de caracteres deve ter um tipo de dados VARCHAR e um comprimento real que não seja superior a 32672 bytes. A seqüência de entrada deve fazer parte do alfabeto de proteínas.

O nome do esquema é DB2LS.

O resultado da função é uma cadeia de caracteres com um tipo de dados VARCHAR e um comprimento real que não seja maior que 32672 bytes representando a ordem reversa da seqüência de peptídios.

Referência Relacionada:

- "UDF (User-Defined Function) LSRevPep - Exemplo" na página 79
- "UDF (User-Defined Function) LSRevComp" na página 75
- "UDF (User-Defined Function) LSRevNuc" na página 77

UDF (User-Defined Function) LSRevPep - Exemplo

Você pode utilizar a função LSRevPep em uma instrução SQL onde quer que utilize uma função interna que aceite uma seqüência de peptídios. Por exemplo:

```
SELECT DB2LS.LSRevPep(:NucSeq) FROM SYSDUMMY1;
```

Esse exemplo utiliza a função para reverter os dados de entrada que são fornecidos a partir de uma variável do host.

Se você utilizar uma cadeia inválida, ou um tipo inválido de dados, receberá a seguinte mensagem de erro:

```
SQL0443N A rotina "DB2LS.LSREVPEP" (nome específico "LSREVPEP") retornou
um erro SQLSTATE com o texto de diagnóst. "Seqüência inválida". SQLSTATE=38608
```

O exemplo a seguir mostra como a UDF LSRevPep é utilizada em uma consulta:

```
SELECT  HSP_H_Seq, db2ls.LSRevPep(HSP_H_Seq) as REV_HSP_H_Seq
FROM BlastP
WHERE BlastSeq='MLCEIECRALSTAHTRLIHDFEPRDALTYLEGKNIFTEDH'
```

A tabela a seguir mostra as seqüências revertidas de peptídios que a consulta retorna.

```
HSP_H_SEQ                          REV_HSP_H_SEQ
---------------------------------- ----------------------------------

MLCEIECRALSTAHTRLIHDFEPRDALTYL...  HDETFINKGELYTLADRPEFDHILRTHATS...

RVVSTEHTRLVTDAYPEFSISFTATKN        NKTATFSISFEPYADTVLRTHETSVVR

STAHIRVLRDMVPGDEITCFYGSEFF         FFESGYFCTIEDGPVMDRLVRIHATS

AHTRRCPDHEPRGVITYL                 LYTIVGRPEHDPCRRTHA


4 reg.(s) selecionado(s).
```

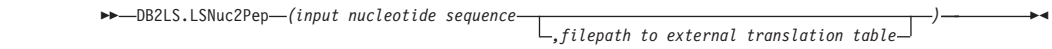
Referência Relacionada:

- "UDF (User-Defined Function) LSRevPep" na página 78

Converter

As UDFs (User-Defined Functions) de conversão convertem uma seqüência de nucleotídios em uma seqüência de peptídios.

UDF (User-Defined Function) LSNuc2Pep



input nucleotide sequence
Uma representação válida de cadeia de caracteres que descreve uma seqüência de nucleotídios. Uma representação de cadeia de caracteres deve ter um tipo de dados VARCHAR e um comprimento real que não seja superior a 32672 bytes.

filepath to external translation table
Se você utiliza uma tabela de conversão personalizada, inclua as informações de caminho do arquivo para localizar a tabela de conversão. O valor da cadeia do caminho não deve ser maior que 255 caracteres.

O nome do esquema é DB2LS.

O resultado da função é uma cadeia de caracteres com um tipo de dados VARCHAR e um comprimento real que não seja maior que 10890 bytes representando a seqüência de peptídios.

A entrada é uma seqüência de nucleotídios utilizando o conjunto de caracteres IUB. As funções assumem que o primeiro códon inicia no primeiro caractere da seqüência de nucleotídios. Se o primeiro códon não começar no primeiro caractere da seqüência de nucleotídios, utilize uma função SUBSTR na seqüência de entrada.

O resultado da função é uma seqüência de peptídios utilizando os símbolos padrão de aminoácidos.

A função:

- Corta espaços em seqüências de entrada.
- Ignora nucleotídios estranhos fora de um quadro de leitura.
- Retorna uma saída nula se você inserir uma seqüência nula de nucleotídios.

Referência Relacionada:

- "UDF (User-Defined Function) LSNuc2Pep – Exemplo" na página 81
- "UDF (User-Defined Function) LSTransAllFrames" na página 82

UDF (User-Defined Function) LSNuc2Pep – Exemplo

Suponha que você deseje converter os dados da seqüência de nucleotídios em uma seqüência de peptídios. Esse exemplo assume que o primeiro códon comece no primeiro caractere da seqüência de nucleotídios.

Você pode chamar a função com uma instrução values. A única entrada é uma seqüência de nucleotídios, como no exemplo a seguir:

```
values db2ls.LSNuc2Pep('TTTTTCTTATTGTCTTCCTCATCGTATTTCTTATGTTGCTGATGT')
```

O resultado da instrução acima é uma seqüência de peptídios utilizando os símbolos padrão de aminoácidos:

```
FFLLSSSSYFLCC*C
```

Se você desejar efetuar a conversão no quadro de leitura +2, utilize o exemplo a seguir:

```
values LSNuc2Pep(SUBSTR('TTTTTCTTATTGTCTTCCTCATCGTATTTCTTATGTTGCTGATGT',2))
```

O inteiro na instrução indica a posição inicial da pesquisa para o códon.

Segue um exemplo de como utilizar essa função como um predicado em uma consulta.

```
SELECT *
  FROM proteindata
  WHERE peptideseq=DB2LS.LSNuc2Pep('TTTTTCTTATTGTCTTCCTCATCG
                                   TATTTCTTATGTTGCTGATGT');
```

O resultado é mostrado na Tabela 37.

Tabela 37. Resultados utilizando a função LSNuc2Pep como um predicado

ID	PROTEINNAME	PEPTIDESEQ
1	proteinA	FSYCLPHRISYVAD

O exemplo a seguir converte uma seqüência de nucleotídios em uma seqüência de peptídios utilizando uma tabela de conversão externa. O primeiro parâmetro é a seqüência de nucleotídios e o segundo parâmetro é o caminho para a tabela de conversão externa.

```
values db2ls.LSNuc2Pep('TTTTCTTATTGTCTTCCTCATCGTATTTCTTATGTTGCTGATGT',
                       'C:\translation.txt')
```

O resultado da instrução acima utilizando essa tabela de conversão específica é a seguinte cadeia:

```
FSYCLPHRISYVAD
```

O exemplo a seguir combina duas UDFs para demonstrar os usos adicionais das funções:

```
values DB2LS.LSNuc2Pep(DB2LS.LSRevCompNuc('TTT..')
```

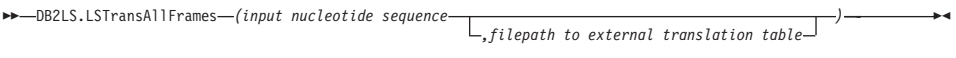
Observe que o exemplo anterior retorna o mesmo resultado que a consulta a seguir:

```
select * from table (DB2LS.LSTransAllFrames ('TTT..')) as t
where t.readframe = -1
```

Referência Relacionada:

- "UDF (User-Defined Function) LSRevNuc - Exemplo" na página 77
- "UDF (User-Defined Function) LSTransAllFrames - Exemplo" na página 83
- "UDF (User-Defined Function) LSNuc2Pep" na página 80

UDF (User-Defined Function) LSTransAllFrames



input nucleotide sequence
Uma representação válida de cadeia de caracteres que descreve uma seqüência de nucleotídios. A seqüência de entrada pode conter códigos de ambigüidade IUPAC. Uma representação de cadeia de caracteres deve ter um tipo de dados VARCHAR e um comprimento real que não seja superior a 32672 bytes.

filepath to external translation table
Se você utiliza uma tabela de conversão personalizada, inclua as informações de caminho do arquivo para localizar a tabela de conversão. O valor da cadeia do caminho não deve ser maior que 255 caracteres.

O nome do esquema é DB2LS.

Utilize a UDF LSTransAllFrames para produzir um conjunto de seqüências de peptídios a partir de uma determinada seqüência de nucleotídios. Essas seqüências de peptídios representam possíveis conversões da seqüência de entrada de nucleotídios, em cada um dos 6 quadros. Essa função é útil quando a entrada contém erros ou o quadro de leitura não é conhecido.

O resultado da função é uma tabela com duas colunas. A primeira coluna é rotulada READFRAME e representa o quadro que é utilizado para a conversão. Essa coluna possui um valor inteiro que representa a posição inicial da conversão. Um inteiro negativo indica uma conversão da situação oposta. A segunda coluna, denominada PEPTIDE, é uma cadeia de caracteres com um tipo de dados VARCHAR e um comprimento real que não seja superior a 10890 bytes representando a seqüência de peptídios.

A função:

- Corta espaços em seqüências de entrada.
- Ignora nucleotídios estranhos fora de um quadro de leitura.
- Retorna uma saída nula se você inserir uma seqüência nula de nucleotídios.

Referência Relacionada:

- "UDF (User-Defined Function) LSTransAllFrames - Exemplo" na página 83
- "UDF (User-Defined Function) LSNuc2Pep" na página 80

UDF (User-Defined Function) LSTransAllFrames - Exemplo

Suponha que você deseje converter uma seqüência de nucleotídios em todos os seis quadros de leitura utilizando a tabela interna de conversão. O exemplo a seguir mostra como fazer isso:

```
SELECT * FROM table(DB2LS.LSTransAllFrames('TTTTTCTTATTGTCTTCCTCATCG
                                           TATTTCTTATGTTGCTGATGT')) as t;
```

A consulta retorna os peptídios em uma tabela, como no exemplo a seguir:

Tabela 38. Resultado da conversão de uma seqüência de nucleotídios

READFRAME	PEPTIDE
1	FFLLSSSSYFLCC*C
2	FSYCLPHRISYVAD
3	FLIVFLIVFLMLLM
–1	TSAT*EIR*GRQ*EK
–2	HQQHKKYDEEDNKK
–3	ISNIRNTMRKTIRK

O próximo exemplo utiliza uma tabela de conversão personalizada para converter uma seqüência de nucleotídios em todos os seis quadros de leitura.

```
SELECT * FROM table
  (DB2LS.LSTransAllFrames
   ('TTTTTCTTATTGTCTTCCTCATCGTATTTCTTATGTTGCTGATGT',
    'C:\msvs6\MyProjects\alin_udf\test\files\translation.txt')) as t;
```

A tabela resultante é a mesma do exemplo anterior, porque a seqüência de entrada é a mesma e a tabela de conversão é a mesma que aquela incorporada à função.

O exemplo a seguir combina duas UDFs para demonstrar os usos adicionais das funções:

```
values DB2LS.LSNuc2Pep(DB2LS.LSRevCompNuc('TTT..')
```

Observe que o exemplo anterior retorna o mesmo resultado que a consulta a seguir:

```
select * from table (DB2LS.LSTransAllFrames ('TTT..')) as t
where t.readframe = -1
```

O exemplo a seguir seleciona um quadro específico de leitura a partir da saída produzida pela função LSTransAllFrames.

```
SELECT * FROM
  TABLE(db2ls.LSTransAllFrames('TTTTTCTTATTGTCTTCCTCATCG
                               TATTTCTTATGTTGCTGATGT')) AS t
  WHERE t.readframe=-2
```

O resultado dessa consulta é:

Tabela 39. Uso da função Readframe

READFRAME	PEPTIDE
–2	HQQHKKYDEEDNKK

Referência Relacionada:

- "UDF (User-Defined Function) LSNuc2Pep – Exemplo" na página 81
- "UDF (User-Defined Function) LSRevNuc - Exemplo" na página 77
- "UDF (User-Defined Function) LSTransAllFrames" na página 82

Formato da Tabela de Freqüência de Códons

Uma tabela de freqüência de códons mostra a freqüência com que os aminoácidos são convertidos novamente para um determinado códon. A UDF LSPep2ProbNuc utiliza a tabela de freqüência de códons para determinar a seqüência de nucleotídios de uma determinada seqüência de peptídios.

A lista a seguir descreve o formato do arquivo da tabela de freqüência de códons:

- Dois pontos adjacentes marcam o início da tabela. Qualquer texto anterior é comentário. Os dois pontos adjacentes são obrigatórios mesmo se não houver comentário antes deles.
- A tabela contém as seguintes colunas:
  1. Aminoác: um código de três letras para o símbolo de aminoácido.
  2. Códon: o códon para esse símbolo de aminoácido.
  3. Número: o número de ocorrências desse códon nos genes a partir dos quais a tabela é compilada.
  4. x/1000: o número esperado de ocorrências do par de códons do aminoácido 1000 conversões em genes.

5. Fração: a fração de ocorrências do códon em sua família de sinônimos de códon.

O produto fornece tabelas de amostra da freqüência de códons no subdiretório sqllib/samples/lifesci/ls_udfs.

Referência Relacionada:

- "UDF (User-Defined Function) LSPep2ProbNuc" na página 51
- "Tabela de Freqüência de Códons - Exemplo" na página 85

Tabela de Freqüência de Códons - Exemplo

A Figura 2 mostra o formato de uma tabela de amostra de freqüência de códons.

```
Aminoác Códon    Número       x/1000   Fração   ..

Gly     GGG      198,00       18,34        0,23
Gly     GGA       71,00        6,58        0,08
Gly     GGT       66,00        6,11        0,08
Gly     GGC      527,00       48,81        0,61

Glu     GAG      534,00       49,46        0,88
Glu     GAA       71,00        6,58        0,12
Asp     GAT       31,00        2,87        0,06
Asp     GAC      481,00       44,55        0,94

Val     GTG      396,00       36,68        0,47
Val     GTA       22,00        2,04        0,03
Val     GTT       44,00        4,08        0,05
Val     GTC      384,00       35,57        0,45

Ala     GCG      446,00       41,31        0,39
Ala     GCA       71,00        6,58        0,06
Ala     GCT      116,00       10,74        0,10
Ala     GCC      503,00       46,59        0,44
... (truncado)
```

Figura 2. Tabela de amostra de freqüência de códons

Referência Relacionada:

- "UDF (User-Defined Function) LSPep2ProbNuc" na página 51
- "Formato da Tabela de Freqüência de Códons" na página 84

Formato da Tabela de Conversão

Este tópico descreve o formato de uma tabela de conversão que é utilizada pelas UDFs (User-Defined Functions) LSPep2AmbNuc, LSTransAllFrames e LSNuc2Pep de biocências.

A lista a seguir descreve o formato do arquivo da tabela de freqüência de códons:

- Dois pontos adjacentes marcam o início da tabela. Qualquer texto anterior é comentário.
- Cada linha da tabela consiste em um símbolo com uma letra de aminoácido, o nome com três letras de aminoácido, os códons não-ambíguos, um ponto de exclamação e os códons ambíguos. O espaço em branco separa cada palavra na linha.
- Cada símbolo de aminoácido e códon deve aparecer apenas uma vez no arquivo.
- Os códons de parada são convertidos no símbolo '*'.
- Os códons constituídos de letras minúsculas são os códons iniciais.
- Todos os outros códons são constituídos de letras maiúsculas.
- Os códons que não possuem uma conversão para um símbolo correspondente de aminoácido são convertidos no símbolo 'X'.

O produto fornece tabelas de conversão de amostra no subdiretório sqllib/samples/lifesci/ls_udfs.

Tabela de Conversão - Exemplo

A Figura 3 na página 87 mostra o formato de uma tabela de conversão de amostra.

```
                Tabela de Conversão de Amostra

Símbolo  3 letras    Códons                    ! IUPAC     ..

 A        Ala         GCT GCC GCA GCG           ! GCX
 B        Asx                                   ! RAY
 C        Cys         TGT TGC                   ! TGY
 D        Asp         GAT GAC                   ! GAY
 E        Glu         GAA GAG                   ! GAR
 F        Phe         TTT TTC                   ! TTY
 G        Gly         GGT GGC GGA GGG           ! GGX
 H        His         CAT CAC                   ! CAY
 I        Ile         ATT ATC ATA               ! ATH
 K        Lys         AAA AAG                   ! AAR
 L        Leu         TTG TTA CTT CTC CTA CTG   ! TTR CTX YTR     ; YTX
 M        Met         atg                       ! ATG
 N        Asn         AAT AAC                   ! AAY
 P        Pro         CCT CCC CCA CCG           ! CCX
 Q        Gln         CAA CAG                   ! CAR
 R        Arg         CGT CGC CGA CGG AGA AGG   ! CGX AGR MGR     ; MGX
 S        Ser         TCT TCC TCA TCG AGT AGC   ! TCX AGY         ; WSX
 T        Thr         ACT ACC ACA ACG           ! ACX
 V        Val         GTT GTC GTA GTG           ! GTX
 W        Trp         TGG                       ! TGG
 X        Xxx                                   ! XXX
 Y        Tyr         TAT TAC                   ! TAY
 Z        Glx                                   ! SAR
 *        End         TAA TAG TGA               ! TAR TRA         ; TRR
```

Figura 3. Tabela de conversão de amostra

Acessibilidade

Usuários com deficiências físicas, como mobilidade restrita ou visão limitada, podem utilizar produtos de software de forma bem-sucedida através dos recursos de acessibilidade. Estes são os principais recursos de acessibilidade no DB2 Information Integrator Versão 8:

- Todos os recursos podem ser operados utilizando o teclado em vez do mouse.
- É possível personalizar o tamanho e a cor de suas fontes.
- É possível receber informações visuais ou auditivas sobre alertas.
- O DB2 suporta aplicativos de acessibilidade que utilizam o Java™ Accessibility API.
- A documentação do DB2 é fornecida em um formato acessível.

Entrada de Dados e Navegação Através do Teclado

Você pode operar as ferramentas de banco de dados do DB2, tais como Control Center, Data Warehouse Center e Replication Center, utilizando apenas o teclado. É possível utilizar teclas ou combinações de teclas em vez de um mouse para executar a maioria das operações.

Nos sistemas baseados em UNIX, a posição do foco do teclado é realçada. Esse realce indica qual área da janela está ativa e onde sua digitação surtirá efeito.

Exibição Acessível

As ferramentas de banco de dados do DB2 possuem recursos que otimizam a interface com o usuário e aprimoram a acessibilidade para usuários com deficiência visual. Esses aperfeiçoamentos de acessibilidade incluem suporte para propriedades de fontes personalizáveis.

Definições das Fontes

Para as ferramentas do banco de dados DB2, você pode utilizar o bloco de notas Definições de Ferramentas para selecionar a cor, o tamanho e a fonte para o texto nos menus e janelas.

Não-dependência de Cores

Não é necessário distinguir cores para utilizar as funções deste produto.

Dicas de Alertas Alternativos

Você pode especificar se quer receber sugestões sobre alertas visuais ou de áudio, utilizando o bloco de notas Definições de Ferramentas.

Compatibilidade com Tecnologias Assistidas

A interface gráfica com o DB2 Information Integrator suporta Java Accessibility API, ativando a utilização de leitoras de tela e outras tecnologias assistidas que são utilizadas por pessoas com deficiências.

Documentação Accessível

A documentação para os produtos da família DB2 está disponível no formato HTML. Você pode exibir a documentação de acordo com as preferência de exibição definidas em seu navegador. Você pode utilizar leitoras de tela e outras tecnologias assistidas.

Avisos

Estas informações foram desenvolvidas para produtos e serviços oferecidos nos Estados Unidos. É possível que a IBM não ofereça os produtos, serviços ou recursos discutidos neste documento em outros países. Consulte um representante IBM local para obter informações sobre os produtos e serviços disponíveis atualmente em sua área. Qualquer referência a produtos, programas ou serviços IBM não significa que apenas produtos, programas ou serviços IBM possam ser utilizados. Qualquer produto, programa ou serviço funcionalmente equivalente, que não infrinja quaisquer direitos de propriedade intelectual da IBM, poderá ser utilizado em substituição a este produto, programa ou serviço. Entretanto a avaliação e verificação da operação de qualquer produto, programa ou serviço não-IBM são de responsabilidade do Cliente.

A IBM pode ter patentes ou solicitações de patentes pendentes relativas aos assuntos tratados nesta publicação. O fornecimento desta publicação não garante ao Cliente nenhum direito sobre tais patentes. Pedidos de licença devem ser enviados, por escrito, para:

Gerência de Relações Comerciais e Industriais da IBM Brasil
Av. Pasteur 138-146
Botafogo
Rio de Janeiro - RJ
CEP 22290-240

Para pedidos de licença relacionados a informações de DBCS (Conjunto de Caracteres de Byte Duplo), entre em contato com o Departamento de Propriedade Intelectual da IBM em seu país ou envie pedidos de licença, por escrito, para:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

O parágrafo a seguir não se aplica a nenhum país em que tais disposições não estejam de acordo com a legislação local: A INTERNATIONAL BUSINESS MACHINES CORPORATION FORNECE ESTA PUBLICAÇÃO "NO ESTADO EM QUE SE ENCONTRA", SEM GARANTIA DE NENHUM TIPO, SEJA EXPRESSA OU IMPLÍCITA, INCLUINDO, MAS NÃO SE LIMITANDO ÀS GARANTIAS IMPLÍCITAS DE NÃO-VIOLAÇÃO, MERCADO OU ADEQUAÇÃO A UM DETERMINADO PROPÓSITO. Alguns

países não permitem a exclusão de garantias expressas ou implícitas em certas transações; portanto, esta disposição pode não se aplica ao Cliente.

Esta publicação pode conter imprecisões técnicas ou erros tipográficos. Periodicamente, são feitas alterações nas informações aqui contidas; tais alterações serão incorporadas em futuras edições desta publicação. A IBM pode, a qualquer momento, aperfeiçoar e/ou alterar os produtos e/ou programas descritos nesta publicação, sem aviso prévio.

Referências nestas informações a Web sites não-IBM são fornecidas apenas por conveniência e não representam de forma alguma um endosso a esses Web sites. Os materiais contidos nesses Web sites não fazem parte dos materiais deste produto IBM e a utilização desses Web sites é de inteira responsabilidade do Cliente.

A IBM pode utilizar ou distribuir as informações fornecidas da forma que julgar apropriada sem incorrer em qualquer obrigação para com o Cliente.

Licenciados deste programa que desejam obter informações sobre este assunto com objetivo de permitir: (i) a troca de informações entre programas criados independentemente e outros programas (incluindo este) e (ii) a utilização mútua das informações trocadas, devem entrar em contato com:

Gerência de Relações Comerciais e Industriais da IBM Brasil
Av. Pasteur 138-146
Botafogo
Rio de Janeiro
CEP 22290-240

Tais informações podem estar disponíveis, sujeitas a termos e condições apropriadas, incluindo em alguns casos o pagamento de uma taxa.

O programa licenciado descrito neste documento e todo o material licenciado disponível são fornecidos pela IBM sob os termos do Contrato com o Cliente IBM, Contrato de Licença de Programa Internacional IBM ou qualquer contrato equivalente.

Todos os dados de desempenho aqui contidos foram obtidos em um ambiente controlado. Portanto, os resultados obtidos em outros ambientes operacionais podem variar significativamente. Algumas medidas podem ter sido tomadas em sistemas de nível de desenvolvimento e não há garantia de que estas medidas serão iguais em sistemas geralmente disponíveis. Os resultados reais podem variar. Os usuários deste documento devem verificar os dados aplicáveis para seu ambiente específico. Os resultados reais podem variar. Os usuários deste documento devem verificar os dados aplicáveis para seu ambiente específico.

As informações referentes a produtos não-IBM foram obtidas junto aos fornecedores desses produtos, anúncios publicados ou outras fontes de publicidade disponíveis. A IBM não testou estes produtos e não pode confirmar a precisão de do desempenho, da compatibilidade ou de qualquer outra reivindicação relacionada a produtos não-IBM. Dúvidas sobre a capacidade de produtos não-IBM devem ser encaminhadas diretamente a seus fornecedores.

Todas as declarações relacionadas aos objetivos e intenções futuras da IBM estão sujeitas a alterações ou cancelamento sem aviso prévio, e representam apenas metas e objetivos.

Estas informações contêm exemplos de dados e relatórios utilizados nas operações diárias de negócios. Para ilustrá-los da forma mais completa possível, os exemplos podem incluir nomes de indivíduos, empresas, marcas e produtos. Todos esses nomes são fictícios e qualquer semelhança com nomes e endereços utilizados por uma empresa comercial real é mera coincidência.

LICENÇA DE COPYRIGHT:

Estas informações podem conter programas aplicativos de exemplo em seu idioma, para ilustrar técnicas de programação em várias plataformas operacionais. Você pode copiar, modificar e distribuir estes programas de exemplo sem a necessidade de pagar a IBM, com objetivos de desenvolvimento, utilização, marketing ou distribuição de programas aplicativos em conformidade com a interface de programação de aplicativo para a plataforma operacional para a qual os programas de exemplo são criados. Estes exemplos não foram testados completamente em todas as condições. Portanto, a IBM não pode garantir ou confirmar a excelência em confiabilidade, desempenho ou função de tais programas.

Cada cópia ou parte destes programas de exemplo ou qualquer trabalho derivado deve incluir um aviso de copyright com os dizeres:

© (*nome da empresa*) (*ano*). Partes deste código são derivadas de Programas de Exemplo da IBM Corp. © Copyright IBM Corp. *_digite o ano ou anos_*. Todos os direitos reservados.

Marcas Comerciais

Os termos a seguir são marcas comerciais da International Business Machines Corporation nos Estados Unidos e/ou em outros países:

IBM
AIX
DB2

Domino
Informix
Lotus
Lotus Notes
QuickPlace
WebSphere

Os termos a seguir são marcas comerciais ou marcas de serviço de outras empresas:

Microsoft, Windows, Windows NT e o logotipo Windows são marcas comerciais da Microsoft Corporation nos Estados Unidos e/ou em outros países.

UNIX é uma marca registrada do The Open Group nos Estados Unidos e em outros países.

Java e todas as marcas comercias e logotipos baseados em Java são marcas comerciais ou marcas registradas da Sun Microsystems, Inc. nos Estados Unidos e/ou em outros países.

Outros nomes de empresas, produtos ou serviços podem ser marcas comerciais ou marcas de serviço de terceiros.

Índice Remissivo

Entrando em Contato com a IBM

Para entrar em contato com a IBM nos Estados Unidos ou Canadá, ligue para um dos seguintes números:

- Serviço de atendimento ao cliente: 1-800-IBM-SERV (1-800-426-7378)
- Marketing e Vendas do DB2: 1-800-IBM-4YOU (1-800-426-4968)

Para informações sobre opções de serviços disponíveis, ligue para um dos seguintes números:

- Nos Estados Unidos: 1-888-426-4343
- No Canadá: 1-800-465-9600
- No Brasil: 0-800-7014-262

Para localizar um escritório da IBM em seu país ou região, consulte o IBM Directory of Worldwide Contacts na Web no endereço www.ibm.com/planetwide.

Informações sobre o Produto

Informações sobre o DB2 Information Integrator estão disponíveis através de telefone ou na Web.

Se você mora no Brasil, é possível ligar para um dos seguintes números:

- Para solicitar produtos ou obter informações gerais: 0-800-787-378
- Para solicitar publicações: 0-800-7014-262

Na Web, visite o endereço www.ibm.com/software/data/integration. Esse site contém as informações mais recentes sobre a biblioteca técnica, solicitação de manuais, downloads do cliente, newsgroups, fix packs, notícias e links para recursos da Web.

Para localizar um escritório da IBM em seu país ou região, consulte o IBM Directory of Worldwide Contacts na Web no endereço www.ibm.com/planetwide.

Comentários sobre a Documentação

Sua opinião ajuda a IBM a fornecer informações de alta qualidade. Envie seus comentários sobre este manual ou outra documentação do DB2 Information Integrator. Você pode utilizar um dos seguintes métodos para enviar os comentários:

- Envie seu comentários utilizando o formulário on-line de comentários do leitor no endereço www.ibm.com/software/data/rcf.
- Envie seus comentários por e-mail (correio eletrônico) para o endereço comments@us.ibm.com. Inclua o nome do produto, o número da versão do produto e o nome e o número da peça do manual (se aplicável). Se você estiver comentando um texto especifico, inclua a localização do texto (por exemplo, um título, um número de tabela ou um número de página).

Impresso em Brazil