UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação
ISSN 0103-2569

Manual da Ferramenta iDFQL
Interactive Data Flow Query Language

Ana Paula Appel
Caetano Traina Júnior

Nº 214

RELATÓRIOS TÉCNICOS

São Carlos – SP
Set./2003

Universidade do Estado de São Paulo - USP
Instituto de Ciências Matemáticas e de Computação - ICMC
Departamento de Ciências da Computação e Estatística

Manual da Ferramenta *iDFQL*

Interactive Data Flow Query Language[1]

Ana Paula Appel
Caetano Traina Júnior

Avenida Trabalhador Sãocarlense, 400 – 13.566-590 São Carlos - SP

[anaappel, caetano]@icmc.usp.br

ICMC-USP, São Carlos

18 de Setembro de 2003

[1]Este trabalho foi suportado pelo CNPq

Sumário

Lista de Figuras

Lista de Tabelas

Resumo

Apesar de muito trabalho já haver sido efetuado na área de linguagens de consulta para Sistemas de Gerenciamento de Bancos de Dados Relacionais - SGBDR, existem apenas dois paradigmas básicos para essas linguagens, que são a linguagem de comando e a de preenchimento de formulários, representados respectivamente por SQL *(Structured Query Language)* e por QBE *(Query by Example)*.

Este Trabalho apresenta um terceiro paradigma para representar consultas em bancos de dados relacionais, baseado em Diagramas de Fluxo de Dados. Ele permite a criação de uma linguagem gráfica para representação da consulta usando esse paradigma (a *Data Flow Query Language* - DFQL), permitindo que a correspondente interface de consulta a um Sistema de Gerenciamento de Bancos de Dados Relacional também tenha uma maneira gráfica de interação com o usuário. A interface é suportada por uma ferramenta que permite a edição de diagramas de fluxo de dados, através de um conjunto de operadores representados graficamente. A execução das consultas representadas nos diagramas é feita analisando-os e gerando comandos equivalentes em SQL, os quais são submetidos a um Sistema de Gerenciamento de Banco de Dados Relacional, sendo os resultados gravados ou mostrados ao usuário.

Capítulo

1

Introdução

Embora bastante esforço tenha sido dispendido no desenvolvimento de linguagens de consulta a Sistemas de Gerenciamendo de Bancos de Dados - SGBD, excluindo linguagens de propósitos específicos, todas as linguagens existentes são extensões diretas de apenas duas linguagens básicas: a *Structured Query Language* - SQL [Chamberlin and Boyce, 1974] e a *Query by Example* - QBE [Zloof, 1975]. Ambas as linguagens foram desenvolvidas quase ao mesmo tempo, nos primórdios do desenvolvimento do modelo e dos sistemas relacionais. A linguagem SQL é o padrão de fato para o acesso aos SGBDs Relacionais (SGBDRs) a partir de aplicativos, e a própria QBE está presente na maioria dos produtos dos fabricantes de SGBDRs, em geral em ferramentas auxiliares para acesso esporádico à base. Apesar das diferenças nos diversos SGBDs quanto à interface que as ferramentas em QBE apresentam para o usuário, elas seguem basicamente o mesmo estilo tabular.

Um aspecto que em determinadas aplicações pode ser negativo, é que a linguagem SQL é puramente textual, e não existe uma maneira natural de estendê-la para permitir consultas envolvendo dados que não tenham uma descrição textual adequada. Por exemplo, consultas que envolvem referências a imagens somente podem ser escritas representando-se uma imagem através de um texto que as identifique - como por exemplo, o nome de um arquivo do sistema operacional que as armazena. Embora isso seja adequado para uma linguagem de comando embutida entre um aplicativo e o SGBD, a quebra do paradigma da linguagem para uso como linguagem de consulta casual envolvendo usuários humanos é no mínimo desconfortável. A integração de uma linguagem que represente graficamente

os comandos de consulta com as imagens faz com que a representação das imagens seja mais natural. Por outro lado diagramas de fluxo de dados são bastante utilizados para a representação de processos em geral em vários campos de conhecimento.

Utilizando-se disto foi desenvolvida, uma nova forma de representação de consultas a SGBDRs, através de um terceiro paradigma para representar consultas em bancos de dados relacionais, baseado em Diagramas de Fluxo de Dados. Sobre esse paradigma, define-se também uma interface interativa que permite representar graficamente os comandos de consulta. Essa representação mantém o mesmo poder de representação das duas anteriores, uma vez que pode também ser convertido para a mesma representação algébrica.

Assim esse relatório técnico tem por objetivo apresentar a definição de uma nova linguagem de consulta a bancos de dados relacionais que utiliza o paradigma de fluxo de dados, e mostrar a ferramenta implementada para comprovação da linguagem, que utiliza diagramas de fluxo de dados para a criação de diagramas de consulta e a execução das mesmas.

O restante do relatório está organizado da seguinte maneira: o capítulo 2 versa sobre a representação de comandos através de diagramas de fluxo de dados. O capítulo 3 apresenta uma ferramenta que permite validar a representação proposta e explorar a representação de consultas através do paradigma de fluxo de dados mostrando a construção de uma consulta na ferramenta *iDFQL*. Por fim o capítulo 4 resume as conclusões deste trabalho.

Capítulo

2

Expressando Consultas através de Fluxo de Dados

Diagramas de fluxo de dados são representações espaciais de processos interconectados por ligações denominadas fluxos, os quais indicam a passagem dos dados que chegam e saem de cada processo. Cada processo pode receber um conjunto de fluxos de dados e pode gerar outros fluxos de dados. Todos os fluxos de dados têm um tipo de dado associado, e diz-se que o fluxo de dados é do mesmo tipo que o tipo de dados associado a ele. Existem diagramas em que todos os fluxos são do mesmo tipo, e outros diagramas em que uma coleção de tipos de fluxos, usualmente pequena, pode estar envolvida. Um diagrama de fluxo de dados em geral envolve processos de vários tipos, sendo que cada tipo de processo tem uma definição precisa de quais tipos de fluxo, e em que quantidade, podem ser tanto recebidos como entrada para serem processados, quanto produzidos como resultado desse processamento. Dessa maneira, pode-se representar graficamente como os processos componentes estão interconectados para produzir um processo complexo.

As diversas operações de recuperação de dados em SGBDRs podem ser consideradas como processos componentes, e as consultas podem ser consideradas processos complexos. Assim, consultas podem ser expressas através de diagramas de fluxo de dados. Essa forma de representação carrega consigo as vantagens típicas desses diagramas, principalmente a representação visual e intuitiva das atividades (no caso, das consultas) e a interatividade na elaboração e execução das mesmas.

A linguagem definida permite a representação de consultas através de diagramas de

fluxos de dados, fazendo uso de uma linguagem gráfica composta por: uma coleção de tipos de processos que realizam operações de manipulação de dados, de condições, preparo de conjuntos de atributos, etc.; uma coleção de tipos de fluxos de dados que permitem interconectar os processos; uma coleção de requisições de operações, que permitem executar todo ou parte do diagrama disponível, conectar com uma base de dados, etc. Essa linguagem é chamada **DFQL** ("*Data Flow Query Language*").

Em DFQL, os operadores da álgebra relacional são representados como processos que podem ser conectados em um diagrama. Como operadores de uma álgebra, todos os operadores relacionais devem operar sobre elementos de um mesmo domínio - as relações - bem como gerar como resultado um elemento desse mesmo domínio. Em um diagrama de fluxo de dados, essa situação corresponde a que todos os fluxos de dados sejam do mesmo tipo. Embora seja possível criar uma representação através de fluxos de dados que utilize todos os fluxos de dados como associados ao tipo de dados "relação", a proposta apresentada aqui procurou flexibilizar essa restrição, para ampliar seu poder de representação.

Assim, os operadores tradicionais da álgebra relacional sofreram algumas modificações conceituais em DFQL, permitindo que os processos equivalentes possam receber e gerar mais de um tipo de fluxo de dados. Os operadores de junção e o operador de seleção da álgebra relacional incorporam a especificação de suas respectivas condições de junção e de seleção como parte do operador. Em sua versão como processos, esses operadores foram alterados para que o tratamento das respectivas condições seja feita por um processo específico, gerando um duto de dados específico para o tipo de dado "condição".

DFQL define quatro tipos de dutos (ligações entre operadores no diagrama) para expressar uma consulta em um diagrama de fluxo de dados, que são:

- **Fluxo de Tuplas (ou de Dados) (D)** - Esses dutos correspondem aos dados propriamente ditos, que são processados desde as tabelas armazenadas na base até que se obtenham as respostas à consulta;

- **Fluxo de Condições (C)** - Esses dutos levam as condições que são utilizadas pelos operadores de seleção e pelos vários operadores de junção;

- **Fluxo de Atributos (A)** - Esses dutos especificam conjuntos de atributos que são utilizados, por exemplo, pelos operadores de projeção;

- **Fluxo de Funções (F)** - Tais dutos especificam funções que são utilizadas, por exemplo, em operadores de cálculo de atributos resultantes de operações de agregados e estatísticas(embora ainda não tenham sido implementados operadores de agrupamento e cálculo de agregados, os dutos necessários já estão operacionais).

Operador	Significado	Conexão
π	Projeção	ED, SD, EA
σ	Seleção	ED, SD, EC
$\times$	Produto Cartesiano	2ED, SD
$\boxtimes$	Junção	2ED, SD, EC
$\boxtimes$	Junção Externa à Esquerda	2ED, SD, EC
$\boxtimes$	Junção Externa à Direita	2ED, SD, EC
$\bowtie$	Junção Externa Completa	2ED, SD, EC
$\cup$	União	2ED, SD
	Leitura de tabela armazenada na base de dados	SD
?	Condição Manual	SC
{}	Lista de Atributos Manual	SA
	Saída de Resultados	ED

Tabela 2.1: Significados e tipos de conexões dos operadores tradicionais da álgebra relacional.

DFQL define doze tipos de processos, os quais correspondem a operadores da álgebra relacional, ou a elementos usados para compor esses operadores. Além disso, existem um operador usado para efetuar as operações de leitura de tabelas na base de dados, e um operador para produzir o resultado final da consulta. A Tabela 2.1 apresenta os tipos de processo definidos, a descrição de cada um, e a indicação dos tipos de dutos que podem ser associados a cada tipo de processo através de conexões. Assim, indica-se o tipo de conexão (D, C, A ou F), e se ela é de Entrada (E) ou Saída (S). Como exemplo, a junção natural recebe como **E**ntrada dois dutos (conexões) de tuplas (2ED) e um duto de condição (EC), e gera como **S**aída um duto de tuplas (SD).

Os tipos de processo "Condição Manual" e "Lista de Atributos Manual" foram definidos para produzir respectivamente dutos de condições e dutos de atributos. A indicação das condições e atributos é feita diretamente pelo usuário nas janelas de parametrização específicas, acessíveis por um toque duplo sobre seu ícone no painel. A combinação de um processo de seleção e um processo de "condição manual" é equivalente à definição do operador de seleção na álgebra relacional; a combinação de um processo de junção e um processo de "condição manual" é equivalente à definição do operador de junção na álgebra relacional; e a combinação de um processo "lista de atributos manual" e um processo de projeção é equivalente à definição do operador de projeção na álgebra relacional. Note-se que um processo de junção requer a definição do fluxo de condições associado. Portanto, em DFQL um processo de junção não é equivalente a um processo "produto cartesiano" sem condições, como é assumido pela sintaxe de SQL. Um processo do tipo "saída de resultados" recebe o resultado da consulta. Em cada diagrama de fluxo de dados em

DFQL deve existir exatamente um processo do tipo "saída de resultados".

A definição dos múltiplos tipos de fluxo de dados permite aumentar o poder representativo da nova linguagem para suportar novos tipos de processos, não previstos na álgebra relacional. Dentre esses tipos surgem como aplicações imediatas os processos de mineração de dados e operações de suporte para armazenagem e recuperação de dados complexos, como por exemplo imagens, sequências de DNA, impressões digitais, entre outros. Por exemplo, um processo de descoberta de agrupamentos em mineração de dados gera uma coleção de condições para definir cada classe, condições essas que podem ser entrada para processos de qualquer tipo que recebam fluxos de tipo condição. Da mesma maneira, um processo de redução de dimensionalidade [Traina et al., 2000] gera uma lista de atributos, que correspondem às dimensões escolhidas no processo e que pode ser entrada para processos que recebem fluxo de tipo "lista de atributos". Em aplicação que requerem a armazenagem e representação de imagens, tanto processos que geram condições quanto os que geram listas de atributos e de funções podem ser parametrizados através de processos específicos que ofereçam suporte a imagens. A versão de DFQL apresentada neste trabalho incorpora apenas os recursos básicos, que a tornam equivalente em poder de expressão de consultas à linguagem SQL. No entanto, a extensão para suportar novos tipos de processos específicos para atividades de mineração de dados e para suporte a tipos de dados complexos pode ser feita com facilidade, uma vez que o desmembramento dos tipos de fluxo nos quatro tipos definidos foi feita levando em conta as necessidades para essas extensões.

Capítulo

3

Construindo uma Consulta com a Ferramenta Interactive Data Flow Query Language - iDFQL

Para mostrar a aplicabilidade do paradigma de fluxo de dados para a expressão de consultas, foi desenvolvida uma ferramenta que implementa uma interface de consulta a um SGBD Relacional utilizando a linguagem DFQL. Essa interface é composta por um editor de consultas, que é formado basicamente por uma coleção de tipos de processos e por um painel onde o usuário pode montar o diagrama de fluxo de dados que representa uma consulta, instanciando, parametrizando e interconectando os tipos de processos. A ferramenta que incorpora o editor , e o modo de execução que permite executar cada consulta preparada como um diagrama de fluxo de dados é chamada de Editor Interativo de Consultas por Diagramas de Fluxo de Dados (iDFQL - *Interactive Data-Flow Query Language*). Embora o projeto dessa ferramenta inclua a extensão para suportar processos específicos para determinadas aplicações (como por exemplo: processos de mineração de dados e operadores de processamento de imagens, e mesmo os operadores de agregados e estatísticos), os tipos de processos descritos e implementados estão restritos aos operadores usuais da álgebra relacional até o presente momento.

A interface com o usuário é feita por uma ferramenta que permite a edição de diagramas de fluxo de dados e a execução desses diagramas, realizando as operações de consulta na base de dados.

3.1 Modo de Edição

Durante a fase de edição, o usuário pode escolher os processos que irão compor sua consulta, escolhendo os respectivos tipos de processo, e "carimbá-los" no painel. Ao mesmo tempo o usuário vai definindo as conexões entre os processos instanciados e seus parâmetros. A ferramenta iDFQL*(Interactive Data Flow Query Language)*, tem sua tela principal inicial mostrada na Figura 3.1, onde estão indicadas as áreas principais de interação com o usuário.

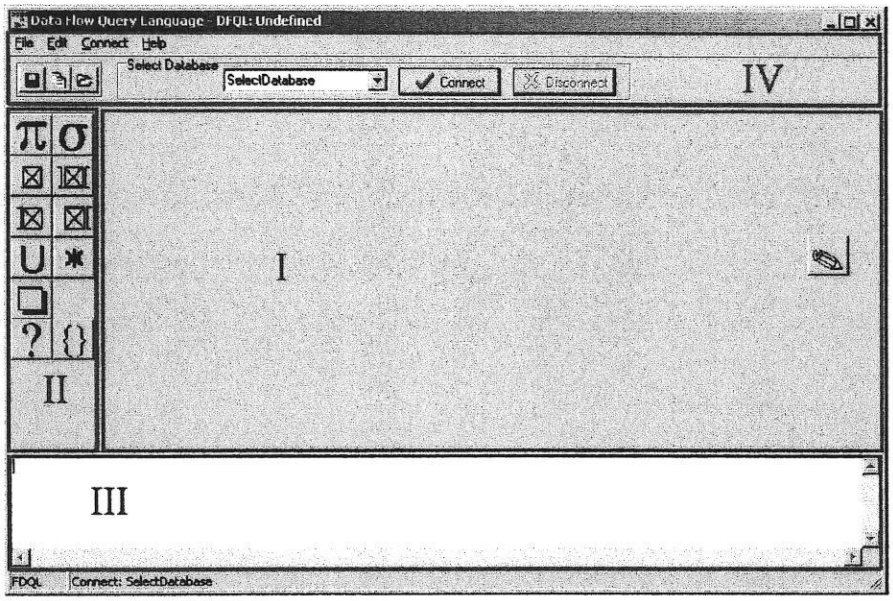


Figura 3.1: Tela principal da ferramenta *iDFQL*

A área I na Figura 3.1 corresponde ao **painel** onde a consulta é construída, "carimbando-se" os operadores que o compõem. A Figura 3.16, mostra uma consulta já construída. O que é feito sobre a área I da tela da Figura 3.1. Cada processo da consulta é representado por um ícone, que indica a operação que ele realiza.

A área II da Figura 3.1 contém o **conjunto de tipos de processos** disponíveis, isto é os operadores disponíveis, onde o usuário escolhe os tipos de processos para serem instanciados e carimbados no painel. Note-se que como deve existir sempre exatamente um processo do tipo "saída de resultados", a ferramenta *iDFQL* mantém sempre um ícone para esse processo, que não pode ser nem instanciado (pois não existe um ícone correspondente no conjunto de tipos de processos na área II), nem removido do painel. Esse operadores estão detalhados na figura 3.2.

A área III na Figura 3.1 é usada para as **mensagens** da ferramenta. Por exemplo, na área de mensagens são mostradas as consultas equivalentes em SQL quando se solicita

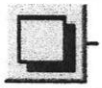
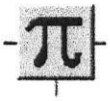
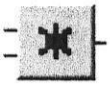
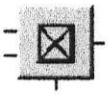
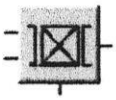
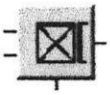
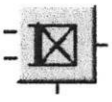
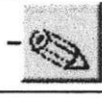
Tabela - Este operador representa a entrada dos dados, na qual é feita a escolha e leitura da tabela no diagrama, possui um duto **Saída do Fluxo de Tuplas**.	
Condição - Este operador representa a escolha da condição da seleção ou da junção, dependendo do operador que esteja conectado com ele. Possui um duto de **Saída do Fluxo de Condições**.	
Lista de Atributos - Este operador representa a escolha dos atributos da projeção. Possui um duto de **Saída do Fluxo de Atributos**.	
Seleção - Este operador representa a operação de seleção. Possui três tipos de dutos: um duto de **Entrada do Fluxo de Tuplas**, um de **Entrada do Fluxo de Condições** e outro de **Saída do Fluxo de Tuplas**.	
Projeção - Este operador representa a operação de projeção. Possui três tipos de dutos: um duto de *Entrada do Fluxo de Tuplas*, um de **Entrada do Fluxo de Atributos** e outro de **Saída do Fluxo de Tuplas**.	
Produto Cartesiano - Este operador representa a operação de produto cartesiano. Possui dois tipos de dutos: dois dutos de **Entrada do Fluxo de Tuplas** e um duto de **Saída do Fluxo de Tuplas**.	
Junção Natural - Este operador representa a operação de junção natural. Possui três tipos de dutos: dois dutos de **Entrada do Fluxo de Tuplas** e um duto de **Entrada do Fluxo de Condição** e outro de **Saída do Fluxo de Tuplas**.	
Junção Externa Total - Este operador representa a operação de junção externa total. Possui três tipos de dutos: dois dutos de **Entrada do Fluxo de Tuplas**, um duto de **Entrada do Fluxo de Condição** e outro de **Saída do Fluxo de Tuplas**.	
Junção Externa à Esquerda - Este operador representa a operação de junção externa à esquerda. Possui três tipos de dutos: dois dutos de **Entrada do Fluxo de Tuplas**, um duto de **Entrada do Fluxo de Condição** e outro de **Saída do Fluxo de Tuplas**.	
Junção Externa à Direita - Este operador representa a operação de junção externa à direita. Possui três tipos de dutos: dois dutos de **Entrada do Fluxo de Tuplas**, um duto de **Entrada do Fluxo de Condição** e outro de **Saída do Fluxo de Tuplas**.	
Saída dos Resultados - Este operador representa a tabela resultante da consulta, possuí apenas o duto de **Entrada do Fluxo de Tuplas**.	

Figura 3.2: Operadores disponíveis na ferramenta iDFQ

a execução dos diagramas e as consultas são submetidas ao SGBD. A área IV mostra os **menus** para controlar a ferramenta e solicitar tarefas específicas, como por exemplo a solicitação de conexão com a base de dados.

Os menus mostrados na área IV da figura 3.1 são descritos abaixo.

New	Cria uma nova área para a criação de um diagrama de Fluxo de Dados
Open	Abre uma consulta (um diagrama) salvo
Save	Salva um diagrama (consulta) criado no painel de edição
Sabe as	Salva um novo diagrama criado no painel de edição
Exit	Fecha a ferramenta *iDFQL*

Tabela 3.1: Menu File

Select All	Seleciona todos os módulos que fazem parte da consulta(diagrama) criada no painel de edição

Tabela 3.2: Menu Edit

Disconnet Database	Desconecta da Base de Dados

Tabela 3.3: Menu Connect

Além destes menus, existem alguns ícones na área IV da Figura 3.1 que servem de atalho para alguma dessas operações. Estes ícones são descritos a seguir na tabela 3.5.

3.1.1 Conexão a uma base de dados

Antes de usar a ferramenta iDFQL, é necessário armazenar os dados a serem visualizados em uma relação de uma base de dados, a qual deve estar registrada no BDE. Para se conectar pela primeira vez a uma base de dados na ferramenta, é preciso criar uma conexão BDE (veja "BDE Administrator" no Painel de Controle do Windows) ou ODBC (veja também o "Adminsitrador de fonte de dados ODBC") como mostrado nas Figuras 3.3, 3.4 respectivamente.

O usuário pode escolher entre se conectar com a base logo que entra na ferramenta (Figura 3.5), ou se conectar quando for parametrizar os operadores. Para fazer a escolha da base de dados, o usuário deve selecionar um dos itens da caixa de seleção *"Select Database"* e depois clicar no botão *"Connect"* como mostrado na Figura 3.5. Se houver algum controle de acesso na base selecionada, o sistema pede a identificação e a senha do usuário.

About	Mostra a tela com informações sobre a ferramenta *iDFQL*

Tabela 3.4: Menu Help

	New	Cria uma nova área para a criação de um diagrama de Fluxo de Dados
	Save	Salva um diagrama (consulta) criado no painel de edição
	Open	Abre uma consulta (um diagrama) salvo

Tabela 3.5: Ícones

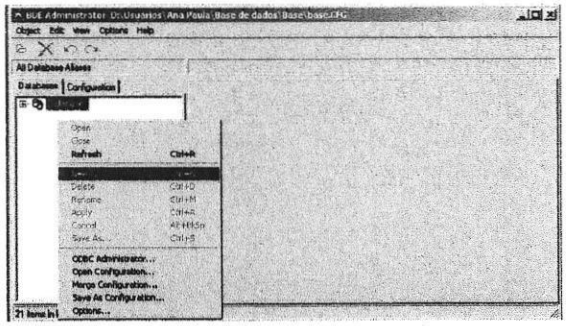


Figura 3.3: *Borland Database Engine* - BDE

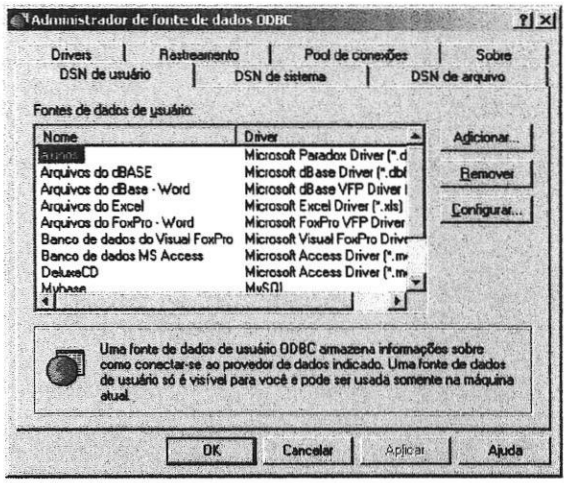


Figura 3.4: *Open Database Connectivity* - ODBC

A barra de *status* mostrada na área III da Figura 3.6 indica com que base de dados o usuário se conectou. Para trocar de base ou apenas se desconectar o usuário deve clicar no botão *"Disconnect"* mostrado na Figura 3.6 área I ou na opção *"Disconnect"* do menu *"Connect"* mostrado também na Figura 3.6 área II.

3.1.2 Edição de um fluxo de dados

Os processos são instanciados e colocados no painel clicando no ícone do tipo desejado no conjunto de tipos de processos disponíveis, e carimbando-os como uma instância de processo no painel. A seguir, cada processo pode ser parametrizado em uma janela específica, acessível através de um toque duplo sobre seu ícone no painel.

A representação de uma consulta típica em DFQL pode ser planejada e representada da mesma maneira que em álgebra relacional. Por exemplo, considerando-se uma base de dados de matrículas que possui uma tabela de alunos e uma tabela das matrículas em

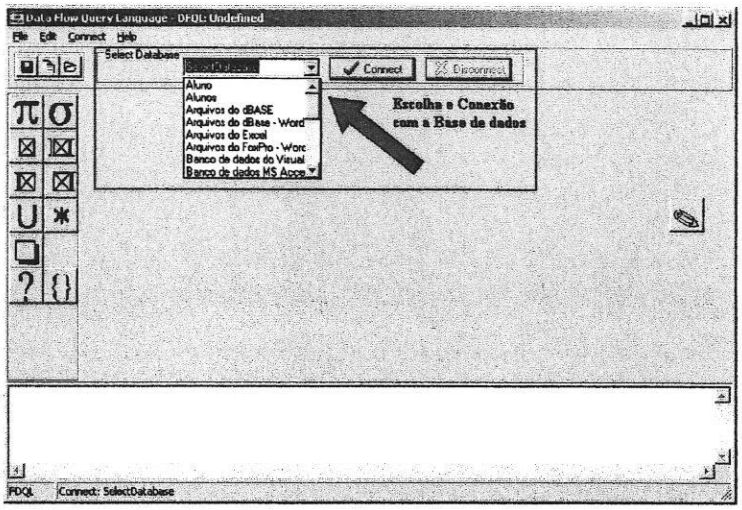


Figura 3.5: Escolha da Base de Dados

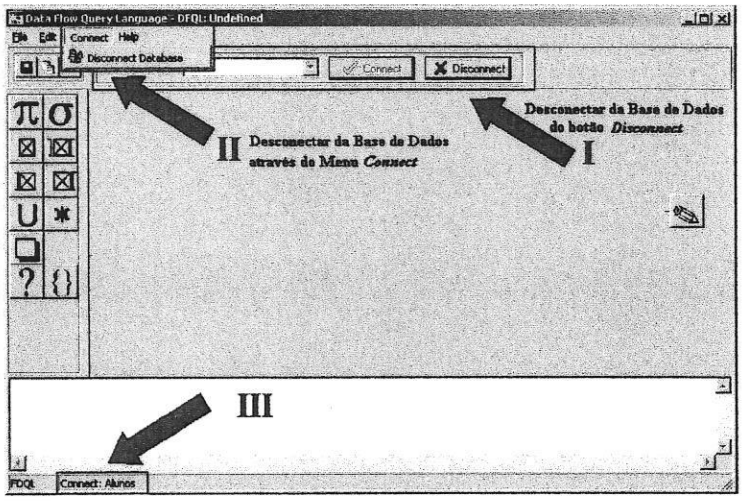


Figura 3.6: Seleção e Conexão com a Base de dados

disciplinas feitas pelos alunos, com as respectivas notas obtidas, a consulta: *"Quais são as notas que cada aluno obteve em cada disciplina em que foi aprovado?"* pode ser expressa em SQL como:

```
SELECT *
FROM Aluno A, AulaMinistrada M
WHERE A.NumAluno = M.NumAluno AND M.Nota >= 6.0;
```

Uma possível expressão dessa consulta em álgebra relacional é a seguinte:

$$\sigma_{(M.Nota \geqslant 6.0)}(Aluno \boxtimes_{(A.NumAluno=M.NumAluno)} AulaMinistrada)$$

Utilizando a representação pelo paradigma de fluxo de dados, o mesmo exemplo pode ser representado como mostrado no painel da Figura 3.16. Verifica-se portanto que a representação em fluxo de dados de uma consulta típica é praticamente a mesma representação gráfica da notação algébrica da consulta. A representação gráfica preserva também todas as possibilidades de otimização que a notação algébrica propicia.

Para carimbar os operadores no editor de consultas é necessario antes o usuário clicar sobre o símbolo do operador desejado na paleta mostrada na área II da Figura 3.1 e depois no painel representado pela área I da Figura 3.1.

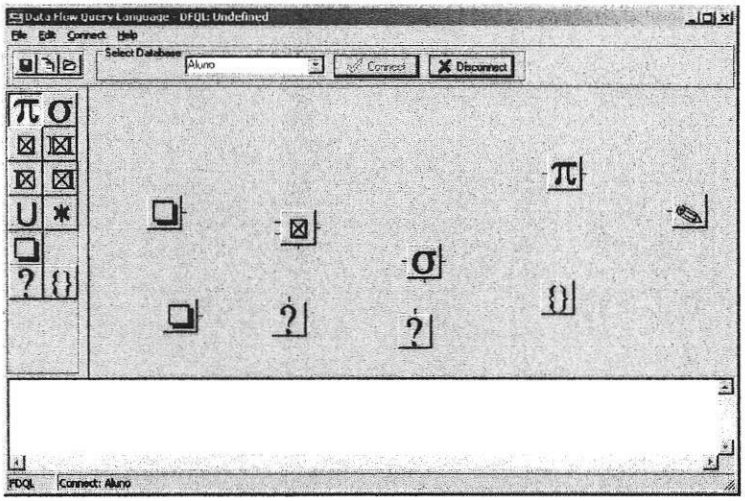


Figura 3.7: Operadores carimbados no painel

Dessa forma o usuário vai instanciando os operadores que farão parte da consulta que está sendo construída como mostra a Figura 3.7.

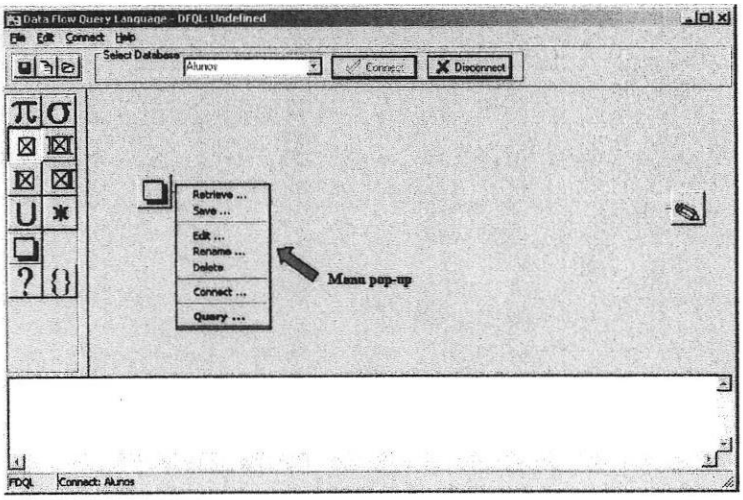


Figura 3.8: *Menu pop-up* com as operações disponíveis sobre os operadores

Depois de carimbado no painel é possivel fazer diversas operações com este operador. Essas operações estão disponíveis no menu *pop-up* que se abre ao clicar sobre o operador com o botão direito do mouse. Esse menu pode ser visto na figura 3.8 e tem suas opções explicadas abaixo na tabela 3.1.2.

Retrieve	recuperação de um operador salvo
Save	salvar apenas o operador
Edit	modificar o operador
Rename	renomear o operdor
Delete	deletar o operador
Connect	conectar dois operadores
Query	executar o pedido de consulta sobre o operador

Tabela 3.6: Operações do *menu pop-up*

Depois de ter os operadores escolhidos carimbados no painel, esses operdores devem ser conectados de forma a montar o diagrama que representará a consulta. Para criar uma conexão(fluxo) entre dois operadores basta clicar com o botão direito do mouse sobre um dos operadores a serem ligados e escolher a opção *"Conectar"* que aparecerá no menu *pop-up* como mostra o área I da Figura 3.9 e em seguida clicar sobre o outro operador representado pela área II da Figura 3.9. Com isso é aberta uma nova janela mostrada Figura 3.9 área III chamada *"ConnectForm"* já com o tipo de fluxo a ser criado entre os dois operadores.

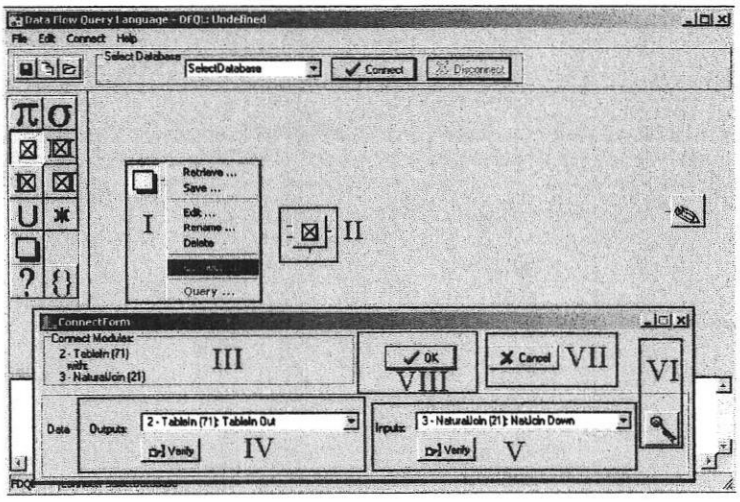


Figura 3.9: Passos para a criação da conexão entre dois operadores

Esse nova janela aberta contém a ferramenta para fazer a criação do fluxo entre os dois operadores escolhido. Para efetuar essa criação o usuário deve conferir se o nome do

operador de entrada indicado pelo área V na Figura 3.9 está correto, caso não esteja o nome correto deve ser selecionado na caixa de seleção *"INPUTS"*. O mesmo procedimento deve ser feito com o nome do operador de saída na caixa de seleção *"OUTPUTS"* Figura 3.9 área IV.

Depois de escolher os operadores que se deseja conectar o usuário deve clicar no botão de criação do fluxo indicado pela Figura 3.9 área VI para fazer a ligação entre os operadores. A criação da conexão entre os operador é confirmada clicando-se no botão "OK" mostrado na área VII da Figura 3.9 que fecha essa tela de conexão e retorna para a ferramenta que agora exibe o fluxo criado entre o operador Tabela de Entrada e Junção Natural como mostrado na Figura 3.10. A área VIII da Figura 3.9 mostra o botão "Cancel" que cancela a operação de criação de um fluxo entre dois operadores.

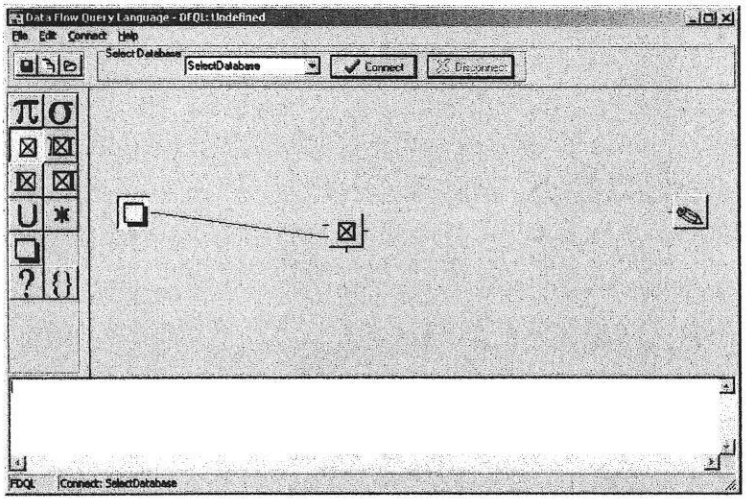


Figura 3.10: Dois operadores conectados por um Fluxo de Dados

Vale lembrar que esse processo de conexão dos operadores deve ser executado em todos os operadores existentes no painel construindo assim o diagrama de fluxo de dados.

3.1.3 Parametrização dos processos

Como parte da "montagem" da consulta, o usuário deve também parametrizar os operadores utilizados. Cada tipo de processo demanda um conjunto de parâmetros específico. Por exemplo, processos do tipo "leitura de tabela" precisam da definição da tabela que deve ser lida em cada processo instanciado; processos do tipo "condição manual" precisam da definição das condições a serem submetidas aos processos que recebem o correspondente fluxo de condição gerado; e os processos do tipo "Lista de atributos manual" precisam da definição dos atributos escolhidos.

Cada tipo de processo tem a especificação de seus parâmetros em uma janela específica, ativada através de um toque duplo no ícone do processo em questão. Para ilustrar, a escolha da tabela a ser lida num operador de leitura é mostrada na janela ativa da Figura 3.11. Para que o usuário possa escolher a tabela dentre as disponíveis na base de dados, como mostrado nessa figura, é necessário que o usuário já tenha realizado a conexão com uma base de dados usando o comando de conexão, disponível na área IV da ferramenta, conforme apresentado na Figura 3.1.

A parametrização dos operadores pode ser feita em qualquer parte do processo de criação do diagrama de fluxo de dados, para isso basta que a ferramenta já esteja conectada com uma base de dados. Não são todos os operadores que podem ser paramentrizados, apenas o operador Tabela de Entrada, de Condição e Lista de Atributos Manuais. Os outros operadores apenas processam as informações vindas desses operadores segundo a operação que representa.

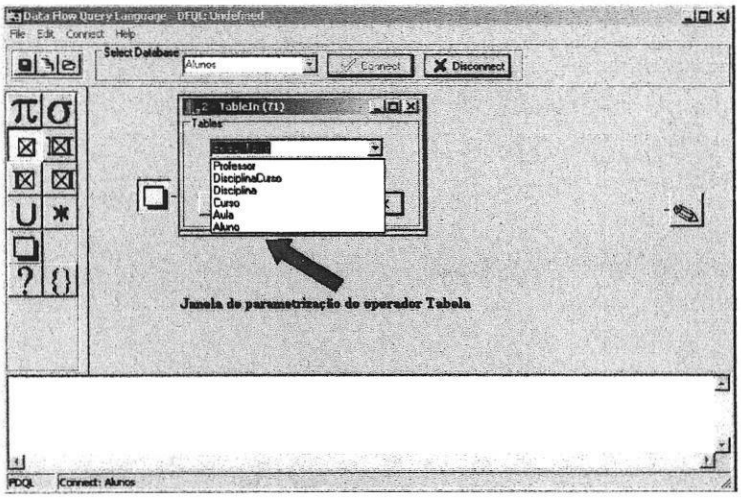


Figura 3.11: Parametrização do operador Tabela

Para o operador Tabela de entrada, a janela de parametrização exibida contém uma caixa de seleção com todas as tabelas existentes na base selecionada como mostra a Figura 3.11.

O operador de "Condição" deve ser paramentrizado para uma condição de seleção ou de junção escolhendo respectivamente a opção *"Value"* ou *"Atributtes"*. Para a condição de seleção o usuário deve selecionar o nome da tabela e o atributo da condição, operador e atribuir um valor para a condição como mostrado na Figura 3.12. Já para a condição de junção, o usuário deverá escolher as duas tabelas e em seguida o atributo de cada tabela que será comparado formando a condição como mostrado na Figura 3.13.

No operador Lista de Atributos Manuais a parametrização é feita escolhendo os atri-

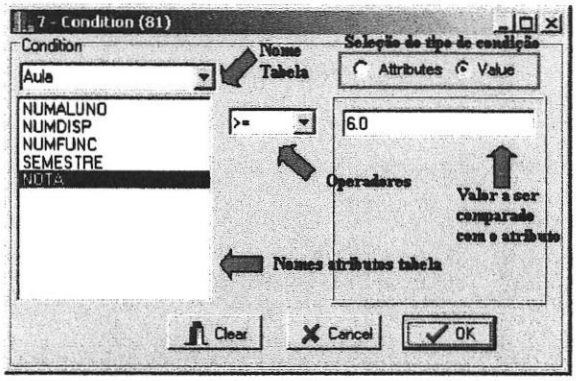


Figura 3.12: Parametrização do operador de condição de seleção

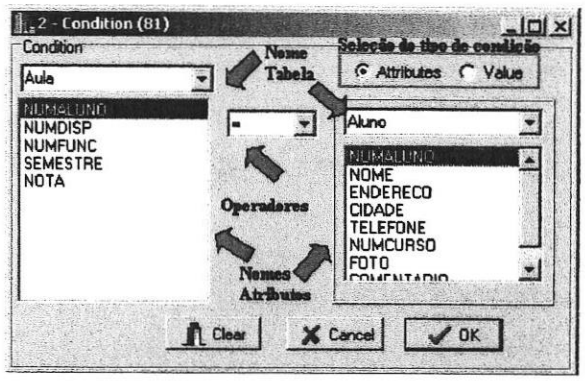


Figura 3.13: Parametrização do operador de condição de junção

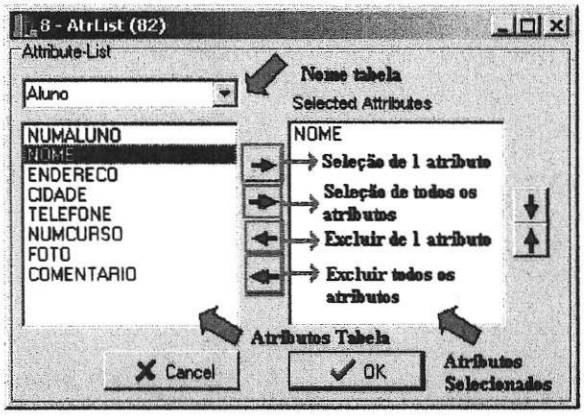


Figura 3.14: Parametrização do operador Lista de Atributos Manuais

butos dentre as diversas tabelas contidas na base de dados. Essa escolha é feita escolhendo uma tabela por vez e selecionado o(s) atributo(s) desejado(s) na tabela e direcionando esse atributo para a lista de atributos escolhidos(*"Selected Attributes"*) através dos botões de "Seleção de Atributos" como mostrado pela Figura 3.14.

Depois do operador ser parametrizado é exibida uma legenda com o conteúdo do operador embaixo do ícone que o representa. Como por exemplo o nome da tabela escolhido no operador Tabela de Entrada como mostrada na Figura 3.15.

O processo de parametrização deve ser feito em todos os operadores de Tabela de Entrada, Condição e Lista de Atributos Manuais existentes no diagrama de fluxo de dados.

Quando os parâmetros e conexões de todos os processos instanciados no painel tiverem sido definidos, o usuário pode então solicitar a execução da consulta criada.

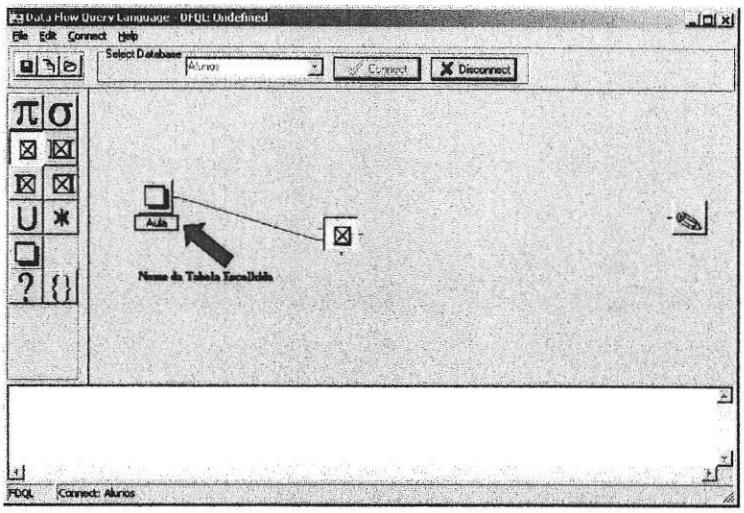


Figura 3.15: Operador Tabela já parametrizado

3.2 Modo Execução da Consulta

Após o diagrama estar construído como mostrado na 3.16, isto é todos os operadores estarem conectados e parametrizados, a opção de executar a consulta é habilitada.

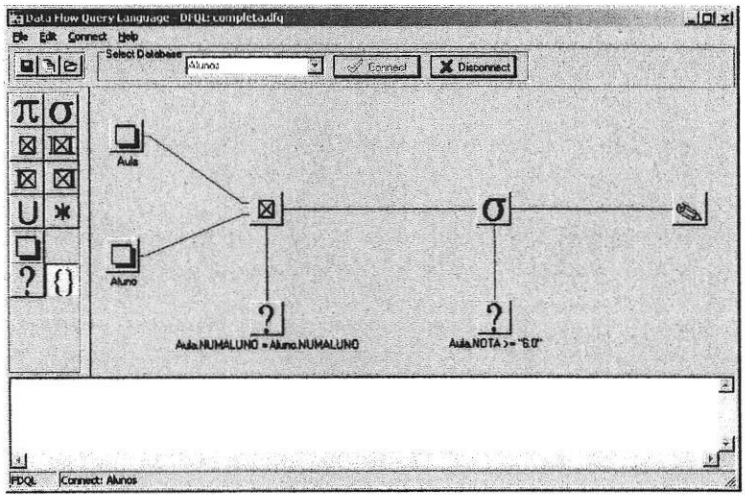


Figura 3.16: Diagrama completo da consulta

O pedido de execução de uma consulta é feito através de um clique sobre um operador e a escolha da opção *query* no menu exibido conforme podem ser vistos na Figura 3.17.

Existem duas maneiras de solicitar a execução de comandos na ferramenta iDFQL: execução completa ou execução parcial para visualização de resultados intermediários.

A visualização de resultados intermediários pode ser solicitada a partir de qualquer processo existente no diagrama de fluxo de dados. Para isso o usuário deve indicar o

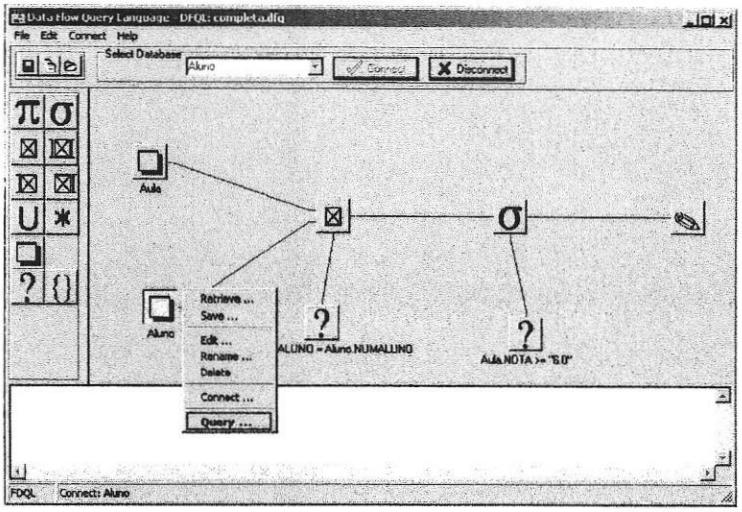


Figura 3.17: Pedido de execução da consulta intermediária sobre a Tabela Aluno

processo que se pretende verificar o resultado. A Figura 3.17 exemplifica esse modo de consulta. Nesse exemplo foi solicitada a visualização do operador de leitura da tabela Aluno (parametrizada anteriormente como mostrado na Figura 3.11).

A execução completa ocorre quando a solicitação é feita a partir do operador ✎, "saída dos resultados". Como mostrado na figura 3.18.

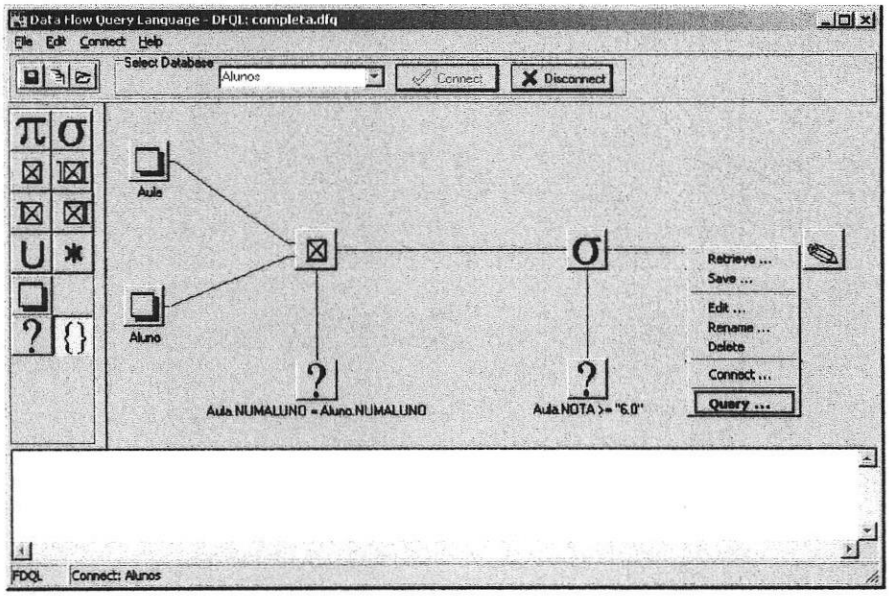


Figura 3.18: Pedido de execução da consulta completa

Nos dois tipos de execução existentes na ferramenta, execução parcial ou total, o resultado pode ser vizualizado numa janela, e/ou armazenado em algum arquivo ou gravado numa tabela. A figura 3.19 mostra o resultado da consulta parcial sobre a tabela aluno.

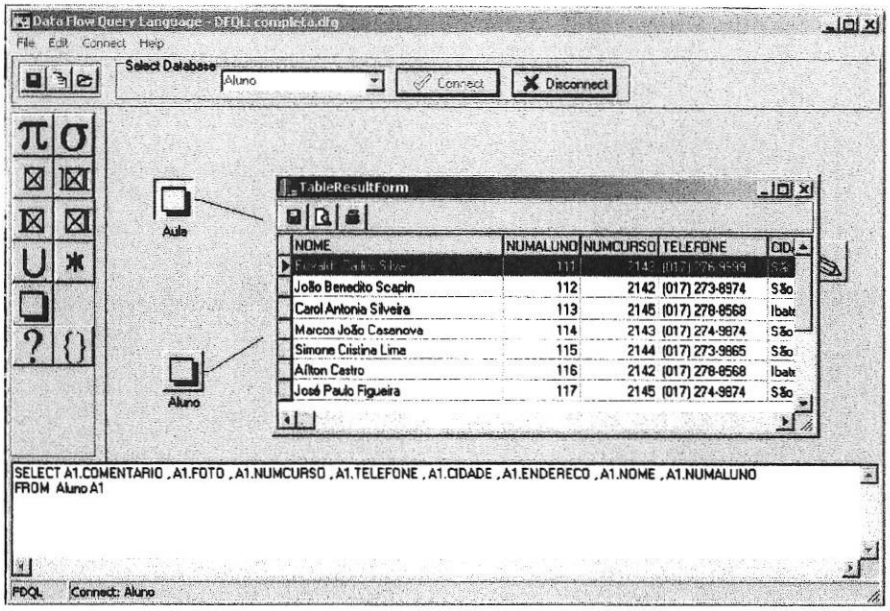


Figura 3.19: O resultado da consulta parcial a partir do processo de leitura da tabela `Aluno`

A execução de um diagrama na ferramenta *iDFQL* é realizada da seguinte maneira:

Sempre que é solicitada a execução de uma consulta, seja ela completa ou parcial, o diagrama é rastreado a partir do processo correspondente ao ícone onde a solicitação foi efetuada, voltando-se passo a passo para cada um dos fluxos de dados recebidos pelos processos visitados, até os processos de leitura das tabelas e demais processos que não recebem fluxos de entrada, chamados processos iniciais. A partir dos módulos iniciais, a navegação volta no sentido inverso, compondo-se a consulta que deve ser efetuada, recuperando-se os parâmetros de cada processo. Essa consulta obtida é submetida à base de dados e a resposta é mostrada em uma janela temporária. Como mostrado no exemplo, a Figura 3.19 mostra o resultado da consulta parcial apresentada na Figura 3.17, solicitada a partir do processo de leitura associado à tabela `Aluno`. Como um outro exemplo, a Figura 3.20 mostra o resultado de uma execução solicitada sobre o processo de junção da consulta exemplificada na Figura 3.16. A Figura 3.21 mostra a visualização da consulta completa, também visualizada em uma janela.

Nota-se que, em qualquer visualização, (Figuras 3.19, 3.20 e 3.21) a área III da ferramenta sempre mostra a consulta em comandos SQL equivalente à consulta efetuada para a execução do diagrama.

A fase de execução de consultas é implementada analisando-se a rede e gerando uma representação interna da consulta. Se essa interface fosse implementada atrelada a um interpretador de consultas completo, essa representação interna corresponderia à árvore

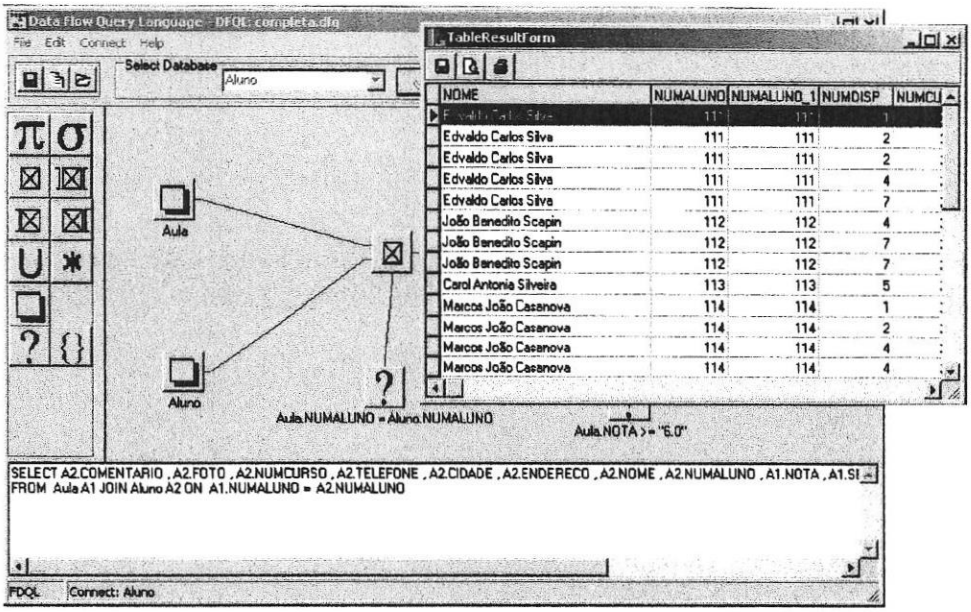


Figura 3.20: Visualização parcial dos resultados a partir do processo de junção

de comandos, equivalente à que é normalmente gerada durante a fase de interpretação dos comando em SQL em um servidor relacional. Embora computacionalmente mais eficiente, essa abordagem requer forte integração com um gerenciador de banco de dados relacional em particular, e tornaria a ferramenta aqui descrita restrita à operação junto com esse gerenciador específico. Para evitar essa dependência, a ferramenta foi implementada representando-se a consulta extraída da rede de fluxo de dados em um ou mais comandos em SQL. Exceto por uma (pequena) redução de eficiência na interpretação dos comandos, essa abordagem apresenta os mesmos benefícios que seriam alcançados se a integração fosse feita internamente a um interpretador de consultas. Além disso, implementada como uma ferramenta independente, é possível dar suporte à implementação de processos que não necessariamente precisam ser executados em um servidor de bases de dados, tais como processos de mineração de dados, e operações de processamento de imagens.

Com os recursos apresentados, a ferramenta *iDFQL* executa um diagrama em DFQL gerando comandos em SQL que representa a consulta a ser emitida para cada operação de visualização de resultados e/ou materialização de tabelas intermediárias. Esses comandos são submetidos ao gerenciador de banco de dados e o resultado é mostrado ou gravado de acordo com os parâmetros de execução parcial ou completa, conforme solicitado.

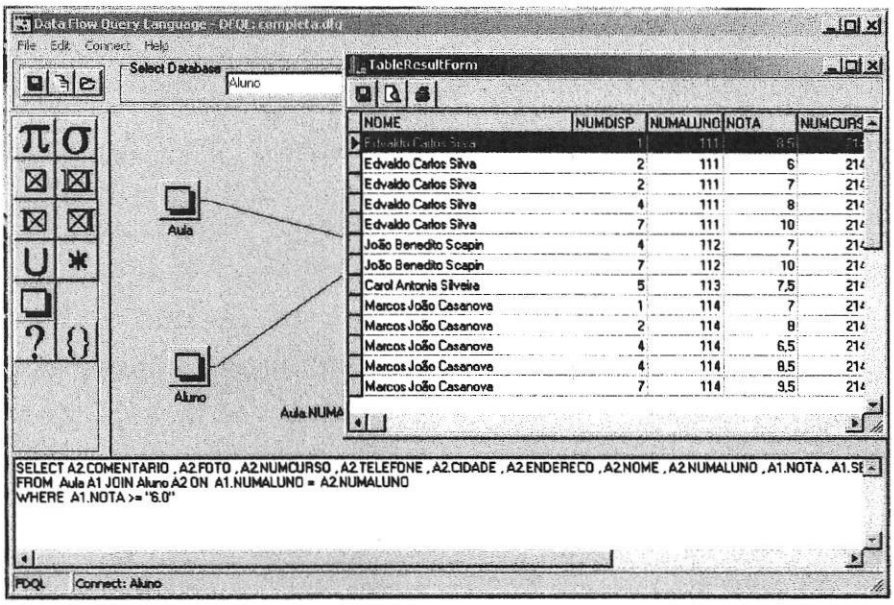


Figura 3.21: O resultado da consulta completa

Capítulo

4

Conclusão

Este relatório apresentou a linguagem DFQL para consulta a bancos de dados relacionais utilizando o paradigma de fluxo de dados, e mostra a construção da ferramenta iDFQL que a implementa, permitindo o acesso a uma base de dados relacional a partir de uma consulta explicitada por um diagrama de fluxo de dados.

A utilização do paradigma de fluxo de dados para o acesso aos dados de um gerenciador relacional nunca foi explorada anteriormente, pois o acesso normalmente é feito utilizando o próprio paradigma relacional através da linguagem SQL (e suas variantes), ou através do paradigma de preenchimento de formulários, através da linguagem QBE. Esses dois paradigmas são adequados para o tratamento dos tipos de dados tradicionalmente suportados pelos gerenciadores relacionais, que são representados em formatos numéricos ou de textos curtos.

No entanto, com o crescimento do uso de gerenciadores relacionais para o suporte a tipos de dados mais complexos, como imagens, áudio, séries temporais, dados de estruturas genéticas, entre outros, linguagens puramente textuais e/ou baseadas em preenchimento de formulários baseados em textos, como é o caso das linguagens SQL e QBE, tornam-se inadequadas.

O desenvolvimento de uma linguagem para acesso a dados armazenados em gerenciadores relacionais baseado no paradigma de fluxo de dados constitui-se em uma terceira alternativa para a expressão das consultas, e tem o mesmo poder expressivo de SQL. Essa alternativa permite a construção de ferramentas de consultas mais interativas e mais "vi-

sual", e pode ser conveniente em algumas aplicações que envolvam dados que tenham uma representação visual mais complexa, como imagens e impressões digitais, pois permite uma representação gráfica da consulta que homogeniza no mesmo paradigma a representação dos dados e da consulta, enquanto mantém o mesmo poder de representação da álgebra relacional. Essa alternativa habilita também a inclusão natural de processos que não são necessariamente "de consulta" aos diagramas, como por exemplo processos de mineração de dados, mas que são executados inter-dispersos aos comandos de consulta.

É interessante notar que a ferramenta iDFQL implementada suporta a linguagem DFQL e é útil também para o ensino dos conceitos de álgebra relacional, dada sua similaridade com os conceitos envolvidos e sua natureza gráfica intuitiva. Além disso, o iDFQL é útil para o preparo de consultas mais elaboradas que serão posteriormente incluídas como comandos SQL embutidos no código de aplicativos, pois o usuário pode experimentar e testar os comandos conforme vão sendo construídos, através das consultas parciais. Nesse sentido, é interessante notar que o preparo inicial de uma consulta em DFQL tende a ser mais trabalhosa do que em SQL, porém por ser muito mais interativa, a DFQL é mais fácil de ser testada, modificada e corrigida, a partir de um processo de interação com o usuário bem mais intuitivo.

Atualmente, a linguagem DFQL e a ferramenta iDFQL estão sendo estendidas para incluir operações de processamento de dados e visualização de informações, bem como para incluir alguns processos de mineração de dados, sendo a seleção de atributos um dos primeiros tratados. Além disso, planeja-se dotar a ferramenta de recursos para suportar consultas baseadas em amostras de dados, permitindo testar com maior precisão a execução de comandos sobre bases de dados com tabelas que já incluem grande quantidade de tuplas.

Referências Bibliográficas

[Chamberlin and Boyce, 1974] Chamberlin, D. D. and Boyce, R. F. (1974). Sequel: A structured english query language. In Rustin, R., editor, *ACM-SIGMOD Workshop on Data Description, Access and Control*, volume 1, pages 249–264, Ann Arbor, MI. ACM Press.

[Traina et al., 2000] Traina, Caetano, J., Traina, A. J. M., Wu, L., and Faloutsos, C. (2000). Fast feature selection using fractal dimension. In Medeiros, C. B. and Becker, K., editors, *XV Brazilian Database Symposium*, pages 158–171, João Pessoa - PA - Brazil.

[Zloof, 1975] Zloof, M. M. (1975). Query by example. In Press, A., editor, *AFIPS National Computer Conference*, volume 44, pages 431–438, Anaheim, CA.

[American National Standards Institute, 1992] American National Standards Institute (1992). *American national standard for information systems: database language — SQL: ANSI X3.135-1992.* Revision and consolidation of ANSI X3.135-1989 and ANSI X3.168-1989, Approved October 3, 1989.

[Batini et al., 1991] Batini, C., Catarci, T., Costabile, M. F., and Levialdi, S. (1991). Visual query systems: A taxonomy. In *VDB*, pages 153–168.

[Catarci and Costabile, 1996] Catarci, T. and Costabile, M. F. (1996). Visual query systems. *J. Visual Languages and Computing*, 7(3):243–245. Guest Editor's Forword.

[Catarci et al., 1997] Catarci, T., Costabile, M. F., Levialdi, S., and Batini, C. (1997). Visual query systems for databases: A survey. *J. Visual Languages and Computing*, 8(2):215–260.

[Codd, 1975] Codd, E. F. (1975). Implementation of relational data base management systems (ncc 1975 panel). *FDT - Bulletin of ACM SIGMOD*, 7(3):3–22.

[Date, 2000] Date, C. J. (2000). *Introdução a Sistemas de Banco de Dados - Tradução da 7ª Edição*. Editora Campus, 7 edition.

[Elmasri and Navathe, 1999] Elmasri, R. and Navathe, S. B. (1999). *Fundamentals of Database Systems*. Benjamin Cummings, third edition.

[Korth and Silberschatz, 1999] Korth, H. and Silberschatz, A. (1999). *Sistema de Bancos de Dados (3º Edição)*. Makron Books, 3 edition.

[Stonebraker et al., 1976] Stonebraker, M., Wong, E., and Kreps, P. (1976). The design and implementation of ingres. *ACM Transactions on Database Systems*, 1(3):189–222.