

# Apostila de Banco de dados

Bancos de dados, (ou bases de dados), são conjuntos de dados com uma estrutura regular que organizam informação. Um banco de dados normalmente agrupa informações utilizadas para um mesmo fim.

Um banco de dados é usualmente mantido e acessado por meio de um software conhecido como [Sistema Gerenciador de Banco de Dados](#) (SGBD). Muitas vezes o termo banco de dados é usado como sinônimo de [SGDB](#).

O modelo de dados mais adotado hoje em dia é o [modelo relacional](#), onde as estruturas têm a forma de tabelas, compostas por linhas e colunas.

Resumindo, um banco de dados é uma coleção de dados relacionados. Entende-se por dado, toda a informação que pode ser armazenada e que apresenta algum significado implícito dentro do contexto ao qual ele se aplica. Por exemplo, num sistema bancário, uma pessoa é identificada pelo seu cpf(cliente). Em um sistema escolar a pessoa é identificada pelo seu número de matrícula(aluno). Além disso, os dados que serão armazenados em cada situação podem diferir consideravelmente.

## Modelo Relacional

O [modelo relacional](#) é uma teoria matemática desenvolvida por Edgar Frank Codd para descrever como as bases de dados devem funcionar. Embora esta teoria seja a base para o software de bases de dados relacionais, muito poucos sistemas de gestão de bases de dados seguem o modelo de forma restrita, e todos têm funcionalidades que violam a teoria, desta forma variando a complexidade e o poder. A discussão se esses bancos de dados merecem ser chamados de relacional ficou esgotada com tempo, com a evolução dos bancos existentes.

De acordo com a arquitetura [ANSI](#) / SPARC em três níveis, os [Bancos de dados relacionais](#) consistem de três componentes:

- uma coleção de estruturas de dados, formalmente chamadas de relações, ou informalmente tabelas, compondo o nível conceitual;
- uma coleção dos operadores, a álgebra e o cálculo relacionais, que constituem a base da linguagem [SQL](#); e
- uma coleção de restrições da integridade, definindo o conjunto consistente de estados de base de dados e de alterações de estados. As restrições de integridade podem ser de quatro tipos:
  - domínio (ou tipo de dados),
  - atributo,
  - relvar e
  - restrições de base de dados.

De acordo com o Princípio de Informação: toda informação tem de ser representada como dados; qualquer tipo de atributo representa relações entre conjuntos de dados.

Nos bancos de dados relacionais os relacionamentos entre as tabelas não são codificados explicitamente na sua definição. Em vez disso, se fazem implicitamente pela presença de atributos chave. As bases de dados relacionais permitem aos utilizadores (incluindo programadores) escreverem consultas (queries), reorganizando e utilizando os dados de forma flexível e não necessariamente antecipada pelos projetistas originais. Esta flexibilidade é especialmente importante em bases de dados que podem ser

utilizadas durante décadas, tornando as bases de dados relacionais muito populares no meio comercial.

Um dos pontos fortes do modelo relacional de banco de dados é a possibilidade de definição de um conjunto de restrições de integridade. Estas definem os conjuntos de estados e mudanças de estado consistentes do banco de dados, determinando os valores que podem e os que não podem ser armazenados.

## Aplicações de bancos de dados

Bancos de dados são usados em muitas aplicações, enquanto atravessando virtualmente a gama inteira de software de computador. Bancos de dados são o método preferido de armazenamento para aplicações multiusuárias grandes onde a coordenação entre muitos usuários é necessária. Até mesmo usuários individuais os acham conveniente, entretanto, e muitos programas de [correio eletrônico](#) e os organizadores pessoais estão baseado em tecnologia de banco de dados standard.

### Aplicativo de Banco de Dados

Um Aplicativo de Banco de dados é um tipo de software exclusivo para gerenciar um banco de dados. Aplicativos de banco de dados abrangem uma vasta variedade de necessidades e objetivos, de pequenas ferramentas como uma agenda, até complexos sistemas empresariais para desempenhar tarefas como a contabilidade.

O termo "Aplicativo de Banco de dados" usualmente se refere a softwares que oferecem uma interface para o banco de dados. O software que gerencia os dados é geralmente chamado de sistema gerenciador de banco de dados (SGBD) ou (se for embarcado) de "database engine".

Exemplos de aplicativos de banco de dados são [Microsoft Visual FoxPro](#), [Microsoft Access](#), [dBASE](#), [FileMaker](#) , (em certa medida) [HyperCard](#), [MySQL](#), [PostgreSQL](#), [Microsoft SQL Server](#) e [Oracle](#).

## Descrição do Banco de dados Acadêmico

O banco de dados descrito representado abaixo é de um pequeno sistema escolar, onde existem basicamente dois componentes que são os alunos matriculados na instituição, bem como as notas obtidas por eles em todas avaliações realizadas durante um período escolar.

Uma vez definido o escopo da aplicação, ou seja, o seu propósito, o próximo passo é identificar os elementos que a constituem, e por consequência definir todos os dados relevantes para cada item existente. Esses elementos são comumente chamados de entidades, e que por questões de facilidade, são representadas por tabelas.

Matrícula	Nome	Série	Turma	Telefone	Data de Nascimento
1	José da Silva	Oitava	1	(31) 3666-9090	05-10-192
2	Ana Maria	Sétima	1	(21) 1234-4567	12-11-1983
3	Paulo Simon	Quinta	1	(31) 8890-7654	17-04-1984
4	Carla Beatriz	Sexta	1	(45) 9946-8989	30-07-1979
5	Ana Paula	Oitava	2	(62) 7878-0909	22-01-1980

Matrícula	Nome	Série	Turma	Telefone	Data de Nascimento
6	Joana Prado	Quinta	2	(35) 8878-0099	06-05-1986

Tabela 1: Definição de ENTIDADE alunos

Matrícula	Data do Teste	Ponto
1	25-03-2004	5.5
2	25-03-2004	6
3	25-03-2004	8
4	25-03-2004	10
5	25-03-2004	7.8
6	25-03-2004	4.6
1	18-05-2004	7.2
2	18-05-2004	9.5
6	18-05-2004	10

Tabela 2: Definição de PONTUAÇÕES

A segunda entidade identificada no problema são as pontuações obtidas por cada aluno. Vale ressaltar que durante um período letivo poderão existir várias avaliações, geralmente em datas diferentes, onde deverão ser armazenados os resultados de todos os alunos para cada um destes testes. A tabela 2 descreve a entidade pontuações, que serve para o propósito exposto anteriormente.

Cada entidade é representada por uma tabela, sendo que neste universo de discussão ou modelo, existem apenas duas tabelas e um relacionamento entre elas, já que cada entidade aluno está ligada à entidade pontuação. Em aplicações mais complexas, poderão existir inúmeras tabelas e relacionamentos de forma a permitir a representação do problema abordado.

### **Atributos definem uma entidade**

Cada entidade, no exemplo alunos e pontuações, é representada por uma tabela, que por sua vez são constituídas de linhas e colunas. Cada coluna representa um fragmento de dado e o conjunto de todas as colunas constitui a entidade propriamente dita. No contexto de banco de dados cada coluna é chamada de atributo e uma entidade será formada por um ou vários atributos.

Um atributo define uma característica da entidade, no exemplo aluno é constituído de seis atributos, que são o número de matrícula, nome, a série que está cursando, a sua turma, o seu telefone residencial e a data de nascimento. O atributo matrícula possui um papel importante no modelo servindo como identificador único para cada aluno. Em um banco de dados caso ocorram registros com valores idênticos não será possível determinar um contexto que os identifiquem unicamente. Por isso deve existir uma chave ou atributo que identifique unicamente cada registro. Ao observar a Tabela 1, percebe-se que não há dois alunos cadastrados com o mesmo número de matrícula. Portanto, este é o atributo chave da entidade, utilizado para a pesquisa de um registro nesta tabela.

A entidade pontuações necessita identificar o aluno, a data da avaliação e a pontuação

atingida pelo aluno. Neste caso, como cada aluno é identificado unicamente pela sua matrícula, este atributo será inserido na tabela de pontuações para permitir associar o aluno à nota registrada, conforme visto na Tabela 2.

Percebe-se que cada atributo possui um conjunto de valores válidos e aceitáveis, que é definido como domínio do atributo. Todas informações vistas na tabela são textuais, isto é, sequências de letras e números, mas é notório que o conjunto de dados contido em cada coluna é diferente umas das outras. No caso de matrícula do aluno, o domínio dos dados é o conjunto dos números inteiros positivos, já que para cada aluno é atribuído um código numérico que denota a ordem em que este foi matriculado na escola. Ou seja, o texto contido nesta coluna é formado por uma combinação de números, portanto não existem letras.

## Modelagem de dados

É a criação de uma estrutura de dados eletrônica (banco de dados) que representa um conjunto de informações. Esta estrutura permite ao usuário recuperar dados de forma rápida e eficiente. O objetivo é incluir dados em uma estrutura que possibilite transformar os dados originais em vários tipos de saídas como formulários, relatórios, etiquetas ou gráficos.

Essa capacidade de transformar informações caracteriza as operações de banco de dados e é a chave de sua utilidade.

Um Banco de Dados – BD, representa uma coleção de dados que possui algum significado e objetiva atender a um conjunto de usuários. Por exemplo, um catálogo telefônico pode ser considerado um BD. Sendo assim, um BD não necessariamente está informatizado.

Quando resolvemos informatizar um BD, utilizamos um programa especial para realizar essa tarefa. Tal programa é denominado SGBD – Sistema Gerenciador de Banco de Dados.

Em um SGBD relacional, enxergamos os dados armazenados em uma estrutura chamada tabela. Neste modelo, as tabelas de um BD são relacionadas, permitindo assim que possamos recuperar informações envolvendo várias delas. Observe o exemplo abaixo:

Clientes		
<i>Codigo</i>	<i>Nome</i>	<i>DataNascimento</i>
1	Marcio	1/6/1975
2	Marlos	5/8/1980
3	Luciane	10/5/1970
4	Wilkie	12/3/1974

Telefones		
<i>Codigo</i>	<i>Fone</i>	<i>Tipo</i>
1	22548954	Residencial
1	88512547	Celular
3	89665485	Celular
4	26539955	Residencial

Neste caso, a tabela Clientes está relacionada com a tabela Telefones. Note que o cliente *Márcio* possui dois telefones: um residencial e um celular. A cliente *Luciane* possui um telefone celular, *Wilkie* possui um residencial e *Marlos* não possui telefone. Entretanto, para que possamos implementar, de forma correta, um BD utilizando algum SGBD, temos que passar por uma fase intermediária – e não menos importante – chamada modelagem de dados.

Quando estamos aprendendo a programar, em geral dividimos esta tarefa em três fases:

- Entendimento do problema;
- Construção do algoritmo;
- Implementação (linguagem de programação).

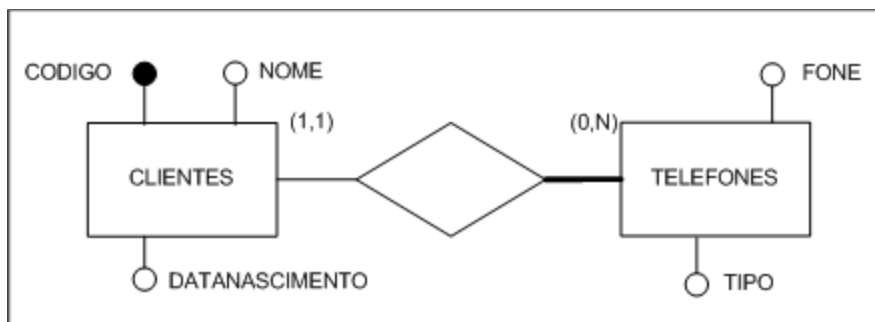
Em se tratando de banco de dados não é muito diferente:

- Entendimento do problema;
- Construção do modelo ER – entidade e relacionamento;
- Implementação (SGBD).

Entender determinado problema nem sempre é uma tarefa fácil, principalmente se você não está familiarizado com a área de atuação de seu cliente.

Antes da implementação em um SGBD, precisamos de uma descrição formal da estrutura de um banco de dados, de forma independente do SGBD. Essa descrição formal é chamada modelo conceitual.

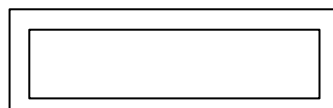
Costumamos representar um modelo conceitual através da abordagem entidade-relacionamento (ER). Nesta abordagem construímos um diagrama, chamado diagrama entidade-relacionamento. Observe abaixo o diagrama que originou as tabelas Clientes e Telefones:



Entidade pode ser entendida como uma “coisa” ou algo da realidade modelada onde deseja-se manter informações no banco de dados (BD). Por exemplo, em um sistema escolar, algumas entidades podem ser os alunos, professores, horário, disciplinas e avaliações. Note que uma entidade pode representar tanto objetos concretos (alunos), quanto objetos abstratos (horário). A entidade forte é representada por um retângulo e a entidade fraca por dois retângulos, um dentro do outro, onde contém o nome da entidade. Observe o exemplo abaixo.



Entidade Forte

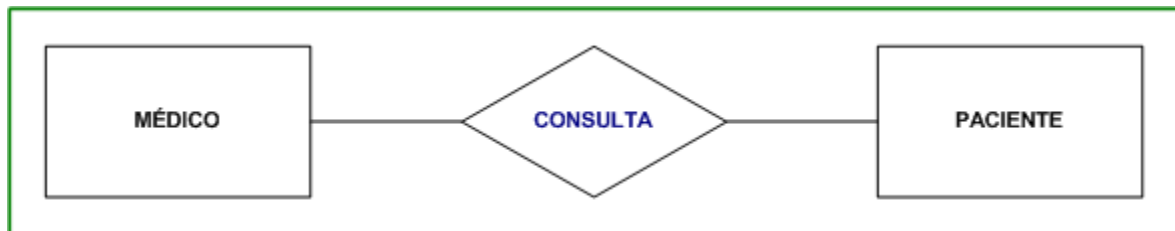


## Entidade Fraca



A entidade ALUNO representa todos os estudantes sobre os quais se deseja manter informações no BD. Quando é necessário especificar um objeto particular (para o exemplo, determinado estudante) usa-se o termo ocorrência de entidade.

Relacionamento é um conjunto de associações entre entidades. O relacionamento é representado por um losango. Esse losango é ligado por linhas aos retângulos que representam as entidades participantes do relacionamento. O exemplo abaixo possui duas entidades, MÉDICO e PACIENTE, e um relacionamento chamado CONSULTA.



O modelo acima informa que o BD mantém informações sobre médicos, pacientes, além de um conjunto de associações (consulta), cada uma ligando um médico a um paciente. Quando é necessário especificar um relacionamento particular (para o exemplo, determinada consulta) usa-se o termo ocorrência do relacionamento. Uma ocorrência de consulta envolve a ocorrência de determinado médico e a ocorrência de determinado paciente.

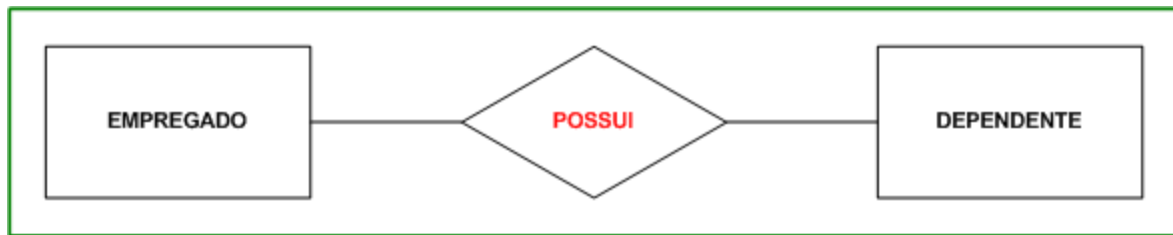
Um relacionamento pode envolver ocorrências de uma mesma entidade. Neste caso, estamos diante de um auto-relacionamento. Observe o exemplo:



CASAMENTO é um relacionamento que envolve duas ocorrências da entidade PESSOA. Para facilitar o entendimento, em geral costumamos identificar o papel de cada entidade no relacionamento (para o exemplo, marido e esposa).

## Cardinalidade do relacionamento

Observe o modelo abaixo:



Estamos diante de um relacionamento (possui) entre as entidades EMPREGADO e DEPENDENTE. Considere as seguintes questões:

- Um empregado pode não ter dependentes?
- Um dependente pode ter mais de um empregado associado ?
- Determinado empregado pode possuir mais de um dependente?
- Pode existir dependente sem algum empregado associado?

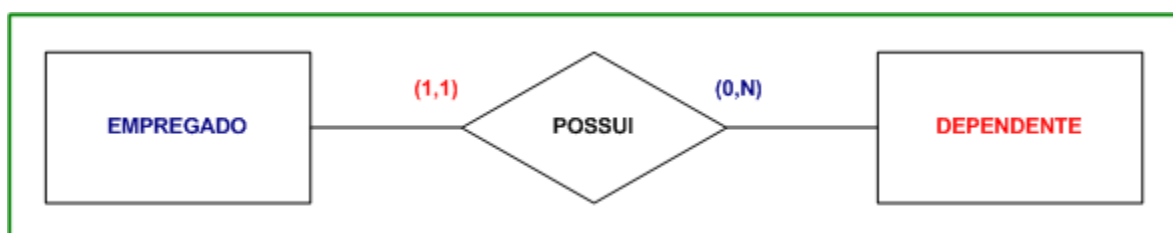
Na realidade, as respostas desses questionamentos dependem do problema sendo modelado. Entretanto, para que possamos expressar essas idéias no modelo, é necessário definir uma propriedade importante do relacionamento - sua cardinalidade.

A cardinalidade é um número que expressa o comportamento (número de ocorrências) de determinada entidade associada a uma ocorrência da entidade em questão através do relacionamento.

Existem dois tipos de cardinalidade: mínima e máxima. A cardinalidade máxima, expressa o número máximo de ocorrências de determinada entidade, associada a uma ocorrência da entidade em questão, através do relacionamento. A cardinalidade mínima, expressa o número mínimo de ocorrências de determinada entidade associada a uma ocorrência da entidade em questão através do relacionamento. Usaremos a seguinte convenção para expressar a cardinalidade:

Número (Mínimo, Máximo)

Observe as cardinalidades mínima e máxima representadas no modelo abaixo:



Para fazermos a leitura do modelo, partimos de determinada entidade e a cardinalidade correspondente a essa entidade é representada no lado oposto. Em nosso exemplo, a cardinalidade (0:N) faz referência a EMPREGADO, já a cardinalidade (1:1), faz referência

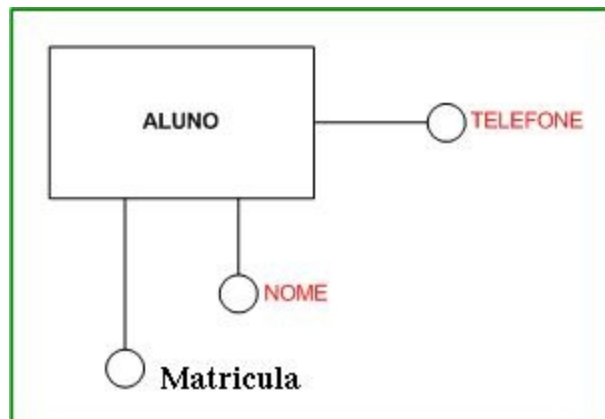


a DEPENDENTE. Isso significa que:

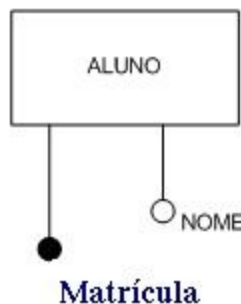
- Uma ocorrência de empregado pode não estar associada a uma ocorrência de dependente ou pode estar associada a várias ocorrências dele (determinado empregado pode não possuir dependentes ou pode possuir vários);
- Uma ocorrência de dependente está associada a apenas uma ocorrência de empregado (determinado dependente possui apenas um empregado responsável).

Observação: Na prática, para as cardinalidades máximas, costumamos distinguir dois tipos: 1 (um) e N (cardinalidades maiores que 1). Já para as cardinalidades mínimas, costumamos distinguir dois tipos: 0 (zero) e 1 (um).

Atributo é uma característica relevante associada a cada ocorrência de entidade ou Relacionamento. Observe no modelo abaixo a notação utilizada para atributos:

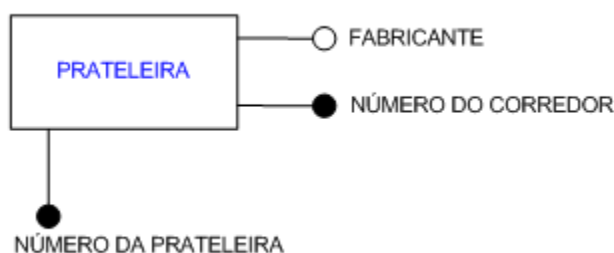


Para deixarmos o modelo de entidade e relacionamentos mais preciso, é necessário que haja uma forma de distinguir uma ocorrência da entidade das demais ocorrências da mesma entidade. Sendo assim, cada entidade deve possuir um identificador. Há várias formas de identificarmos entidades. Observe o modelo abaixo:

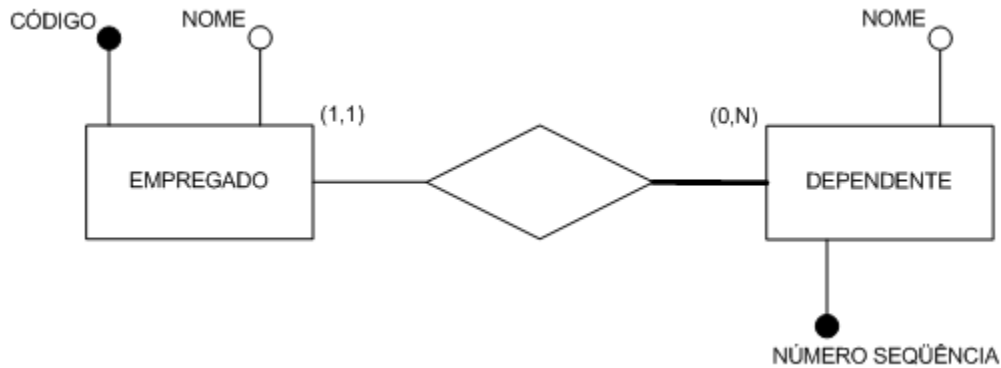


Neste caso, a entidade aluno possui um único identificador (Matricula). Em outras palavras, cada aluno deve possuir uma matrícula diferente.

Existem situações onde é necessário mais de um atributo para identificar determinada entidade. Observe:



Imagine uma biblioteca onde os livros ficam armazenados em prateleiras. Estas prateleiras encontram-se organizadas em corredores. Dessa forma, para identificar uma prateleira é necessário conhecer seu número, além do número do corredor correspondente. Observe o modelo abaixo:



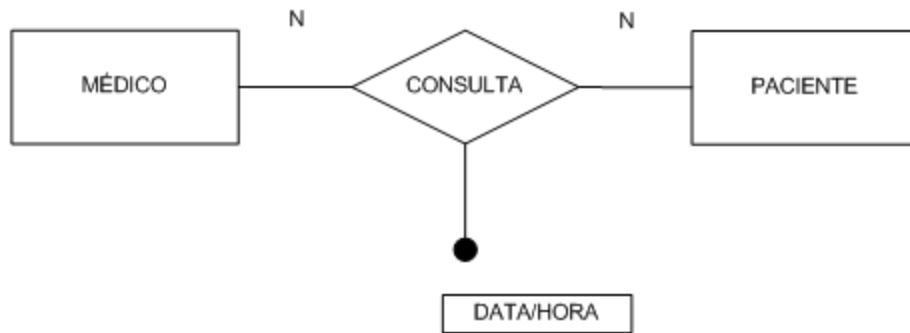
Aqui, o identificador da entidade dependente é composto do atributo NÚMERO SEQÜÊNCIA, além do empregado ao qual o dependente está relacionado. Neste caso, estamos diante de um relacionamento identificador. O relacionamento identificador é identificado por uma linha mais densa.

Vimos que o identificador de entidade corresponde a um conjunto de atributos e relacionamentos cujos valores diferenciam cada ocorrência de entidade. No caso de relacionamentos, em geral a identificação ocorre através das ocorrências das entidades que fazem parte dele. Observe o exemplo:



O modelo mostra que para cada par (analista, projeto) há no máximo um relacionamento de alocação.

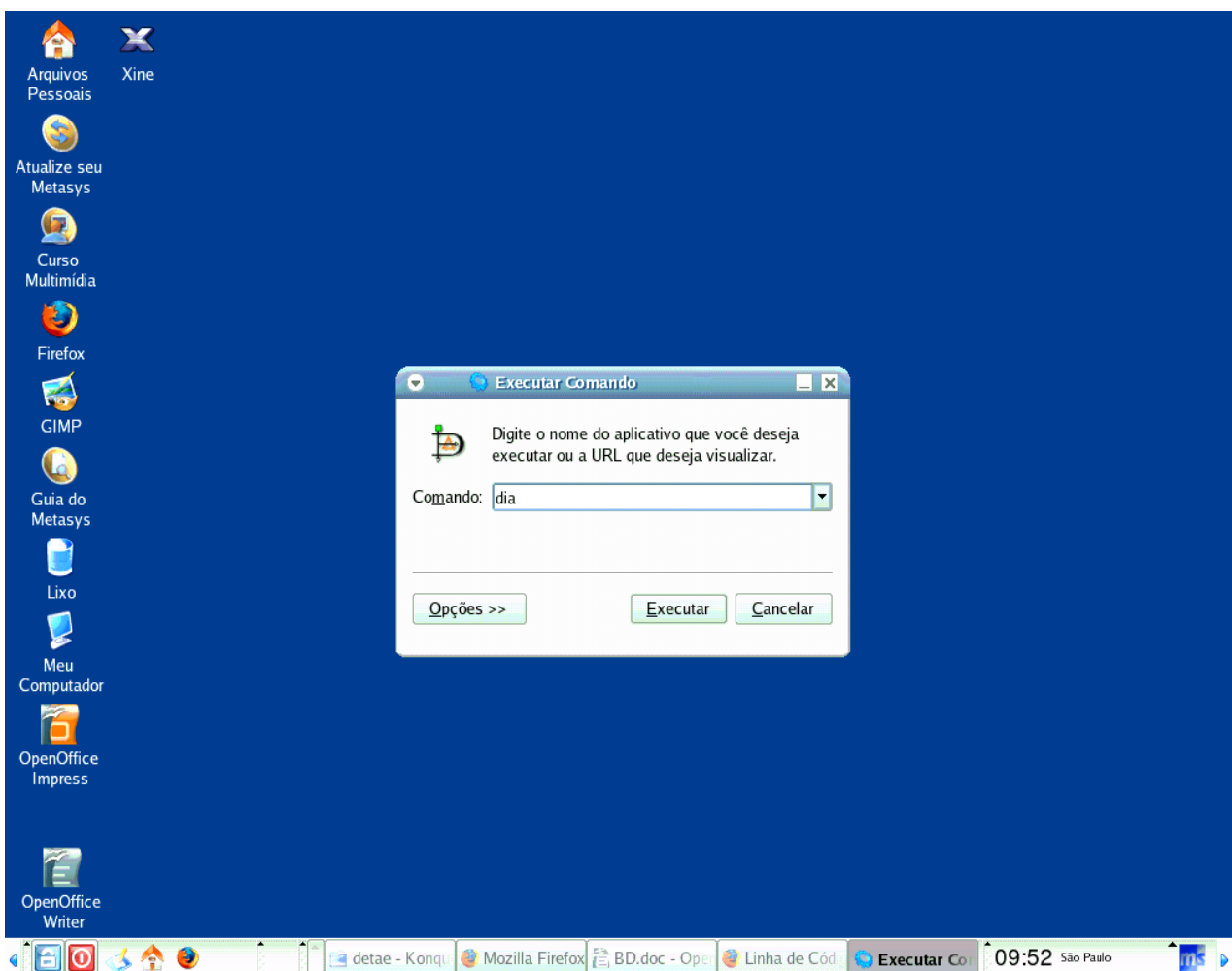
No entanto, há casos onde pode ser necessário relacionar ocorrências de mesmas entidades mais de uma vez. Por exemplo, em um modelo de consultas médicas, determinado paciente pode realizar consultas mais de uma vez com o mesmo médico. Neste caso, podemos utilizar um atributo identificador no relacionamento (data/hora).



## Dia: O Editor de diagrama (Microsoft Visio) para Linux

### Apresentação geral

- Para executar o Dia, aperte "Alt" + F2
- Digite o comando **dia**
- Clique em Executar



O download pode ser encontrado em:

- <http://www.gnome.org/projects/dia/downld.html>

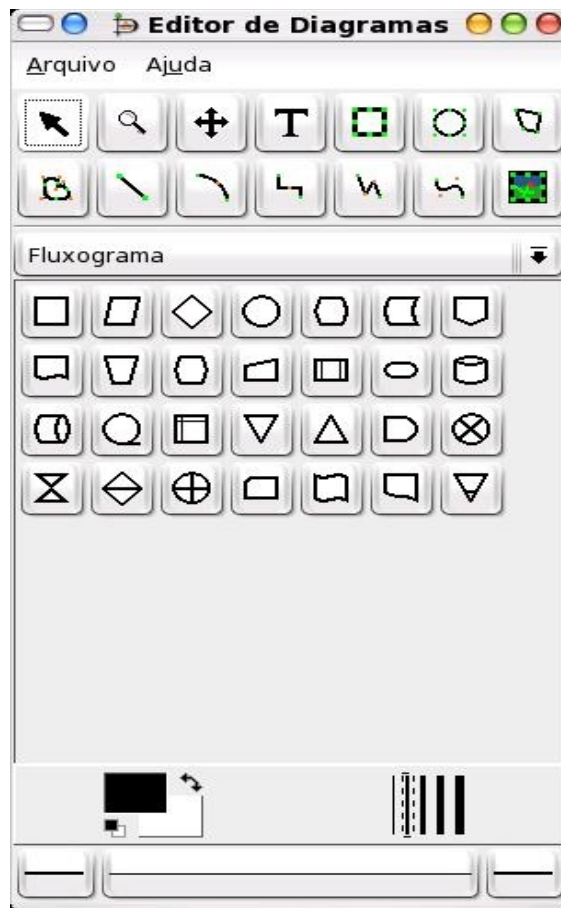
Inclusive neste link podemos encontrar versões compiladas para outras plataformas.



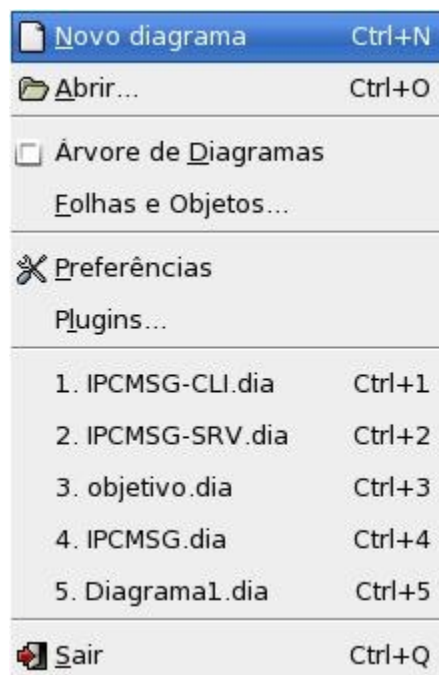
Esta ferramenta é de fácil utilização, portanto apresentaremos apenas alguns recursos em função da simples operacionalidade.

## Conhecendo a interface do dia

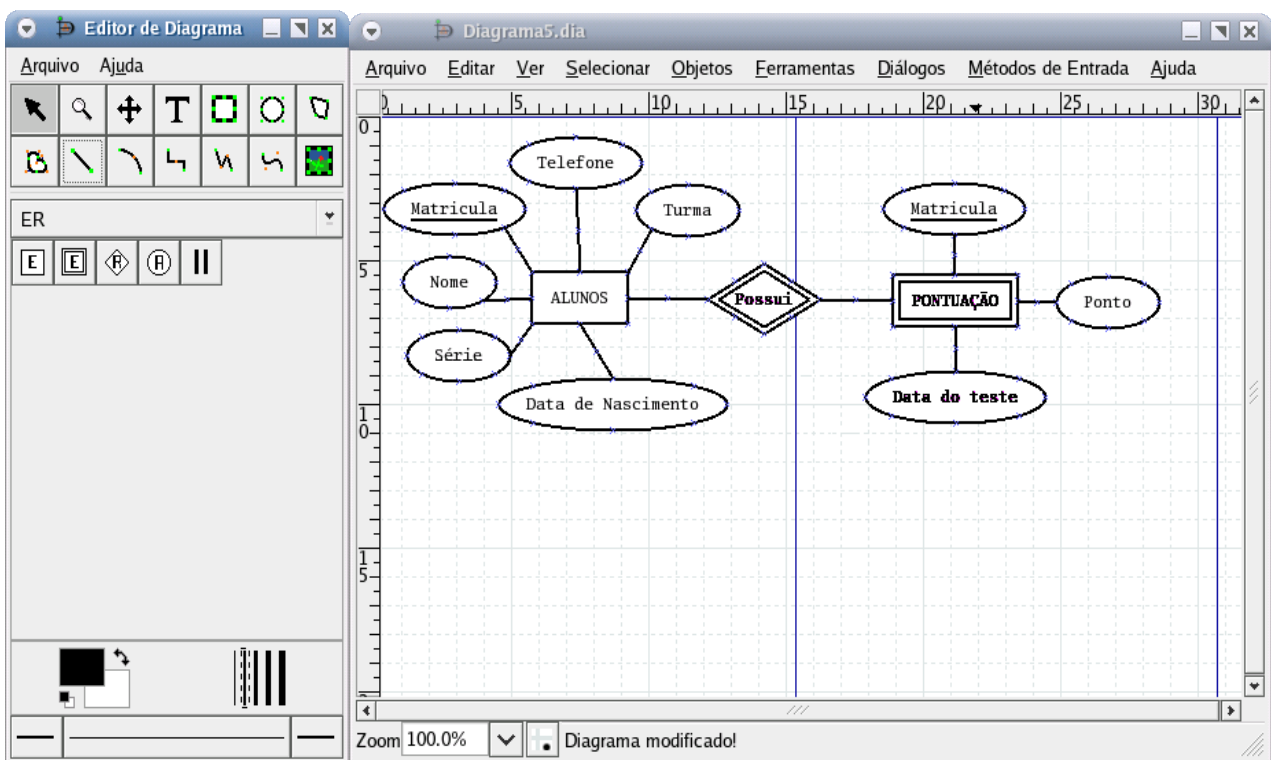
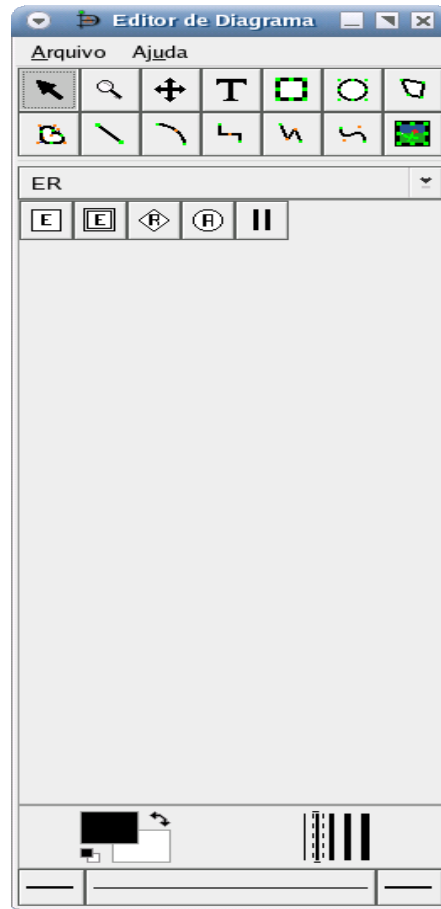
Na figura abaixo, visualizamos a janela de ferramentas principal:



No primeiro bloco encontramos as ferramentas de ações como selecionar, mover, zoom, imagem, linha, círculo e outros.



- Selecionar a opção ER(Entidade Relacionamento)

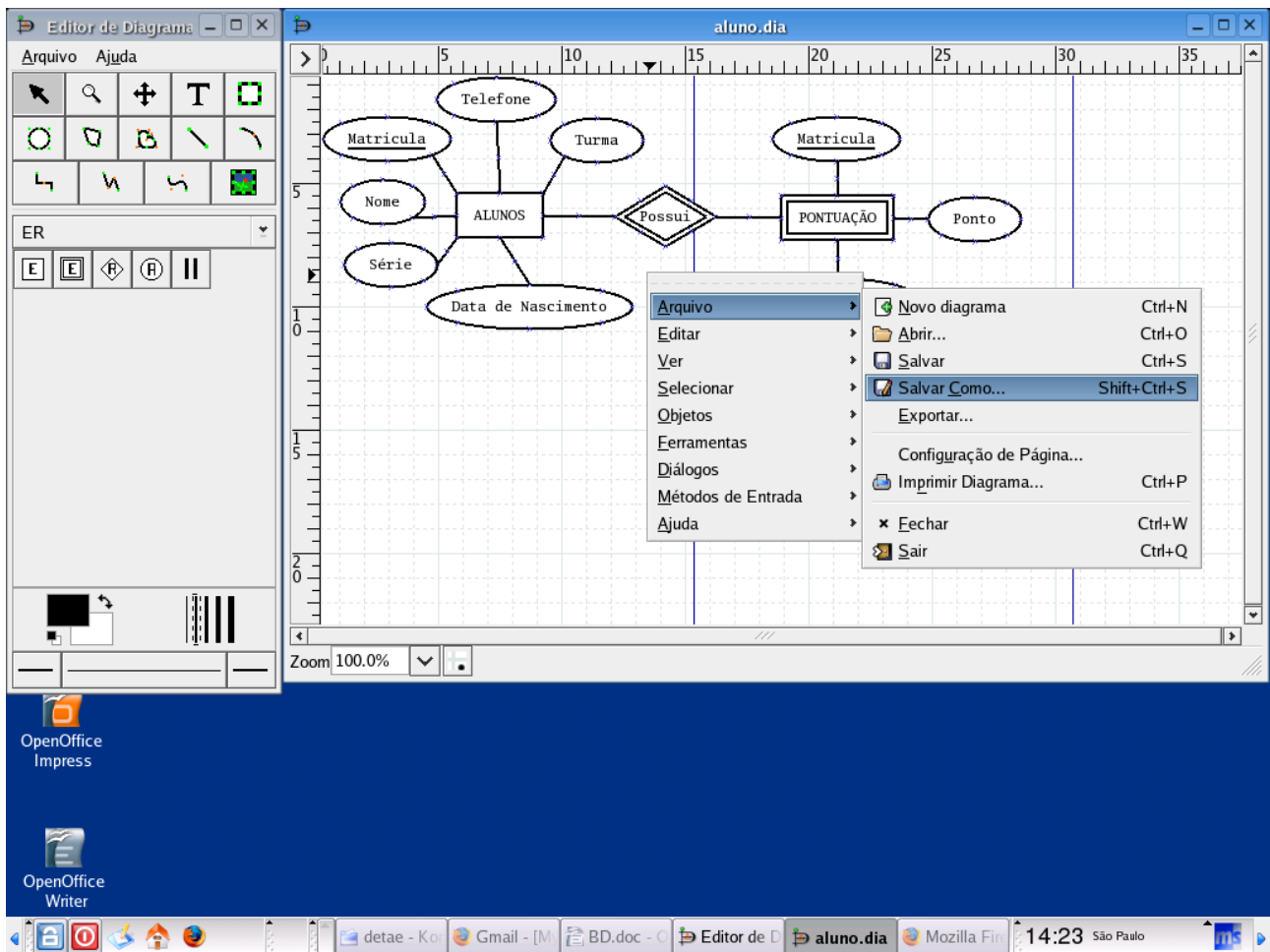


Clicando o botão direito sobre a área de trabalho do diagrama, teremos um menu que nos

proporciona diversos recursos de edição do objetos. Na figura abaixo podemos visualizar as opções principais:

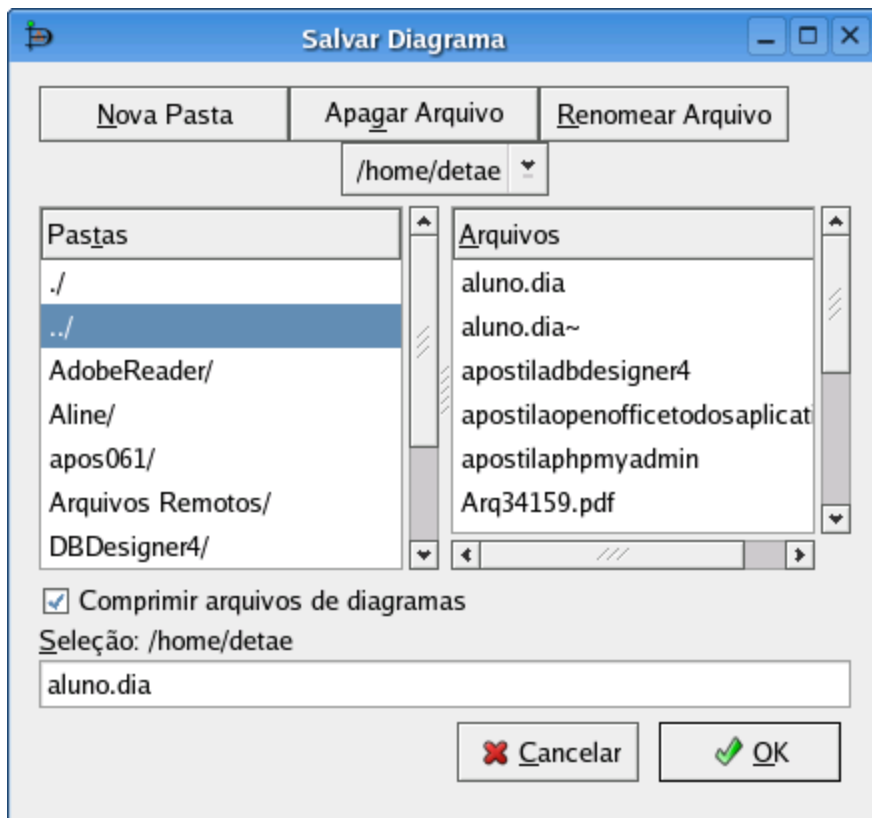


No menu principal, podemos salvar nosso diagrama.



Iremos salvar como aluno.dia.





## Exemplo prático

### 1. Descrição de um sistema para controle de uma empresa

Para facilitar o entendimento e ilustrar os elementos que definem um diagrama ER, será abordado um problema relacionado a uma pequena empresa de construção civil. Assim, será introduzido o escopo do problema, ou seja, o seu propósito e as necessidades que este deve atender. Assim, a partir desta descrição deve-se construir um modelo ER para esta aplicação.

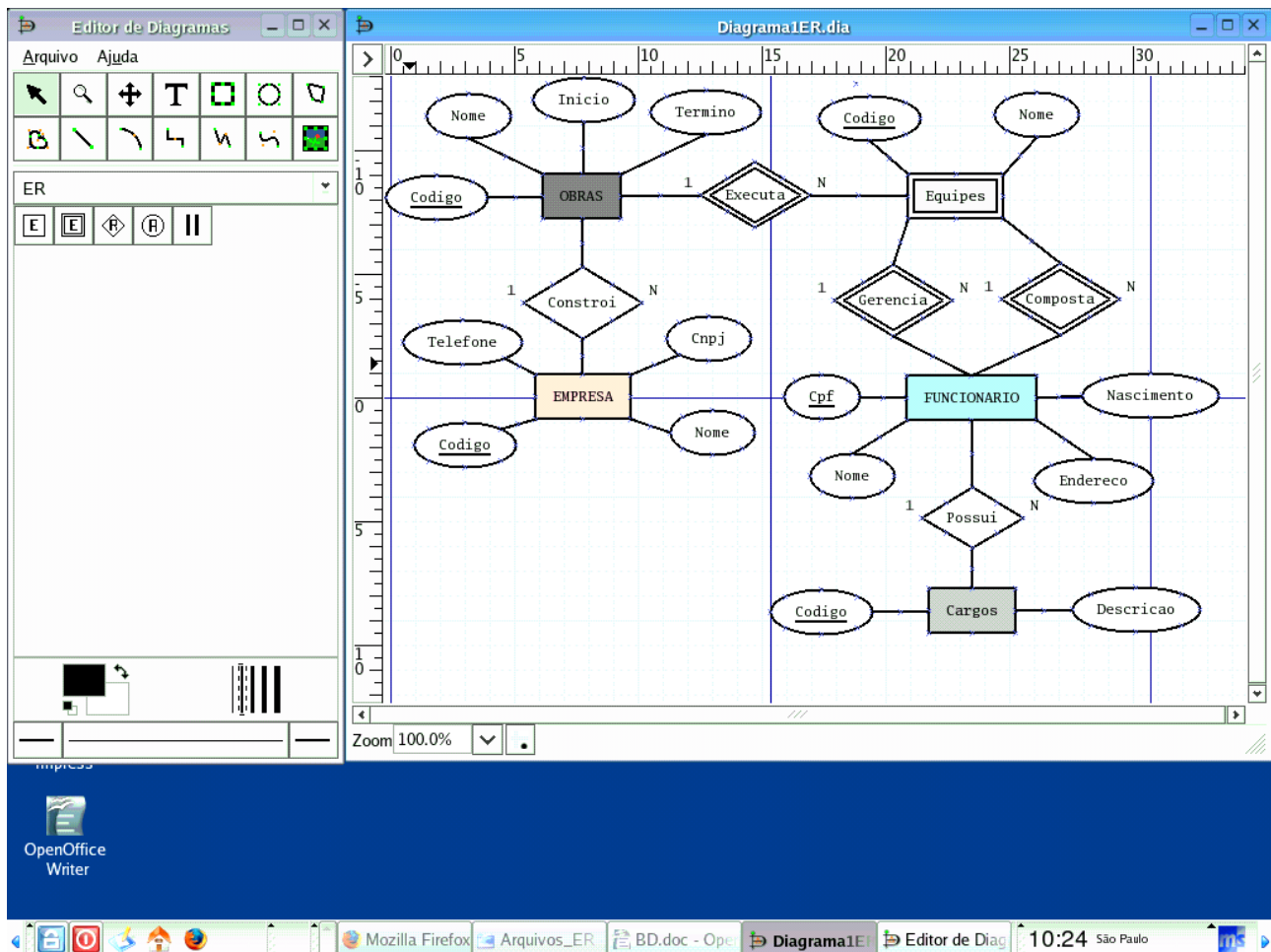
#### 1.1 Descrição de uma empresa de construção civil

A empresa de construção civil tem como objetivo projetar e construir obras tais como prédios, casas, pontes, estradas, para citar algumas de suas atividades. Para isto, é preciso que a empresa possua pessoas ou funcionários capazes de desempenhar as diversas tarefas relacionadas a este ramo de negócios. Por exemplo, é necessário que a empresa contenha em seu quadro de funcionários engenheiros e arquitetos responsáveis pelo projeto e cálculo da infra-estrutura da obra. Além disso, é necessário ainda que haja pedreiros e mestres de obras que serão incumbidos de executar o projeto definido pela equipe de engenheiros. Finalmente, devem-se ter profissionais como eletricitas, bombeiros hidráulicos e carpinteiros para que a obra possa ser executada com sucesso. Para facilitar a coordenação dos trabalhos destes profissionais, a empresa organiza as pessoas em equipes de acordo com as suas especialidades. Desta forma, estas equipes são alocadas em uma ou mais obras que estejam sendo desenvolvidas pela empresa de construção civil. Vale ressaltar que para fazer parte de uma equipe a pessoa deve fazer parte do quadro de funcionários da empresa, e caso um funcionário seja afastado da

empresa, o mesmo deve ser imediatamente retirado da equipe à qual ele pertença. Cada equipe possui um gerente, responsável por coordenar os trabalhos delegados a ela, sendo que este deve ser necessariamente um funcionário da própria empresa. É preciso salientar que ocorrem situações onde há mais de uma obra em andamento, simultaneamente. Daí, cada obra terá várias equipes envolvidas, já que várias habilidades são necessárias para executar a construção da mesma.

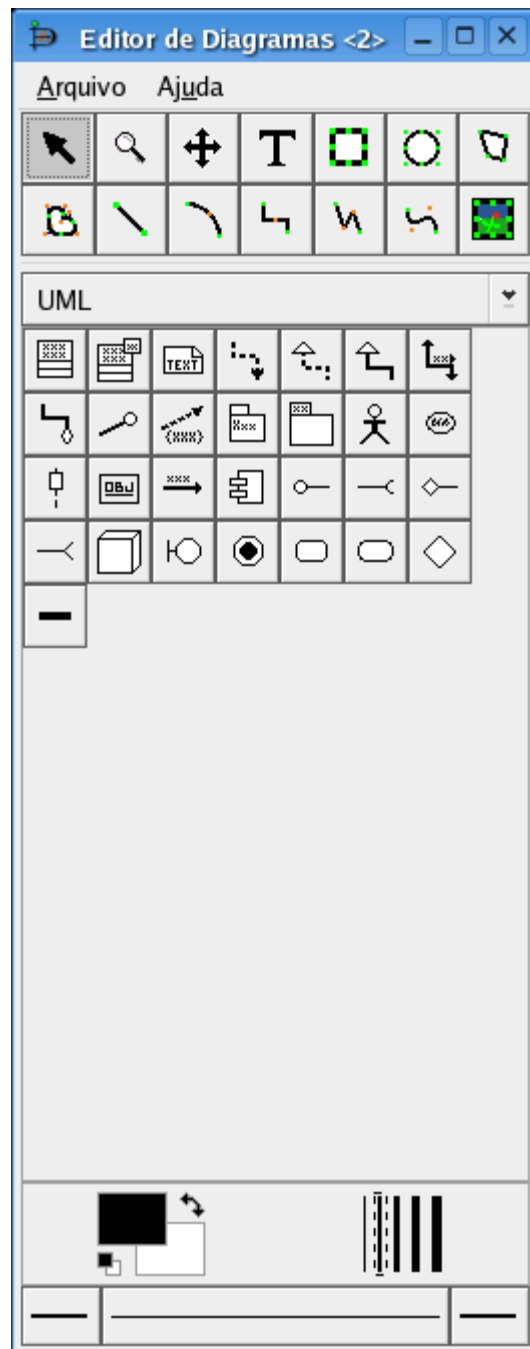
Existem equipes que participam de várias obras ao mesmo tempo, já que as suas tarefas não requerem dedicação exclusiva a um projeto. Este é o caso dos engenheiros, que podem projetar e acompanhar o desenvolvimento de diversas obras simultaneamente, sem que haja comprometimento na qualidade do seu trabalho ou até mesmo prejuízos para o cronograma de execução das mesmas.

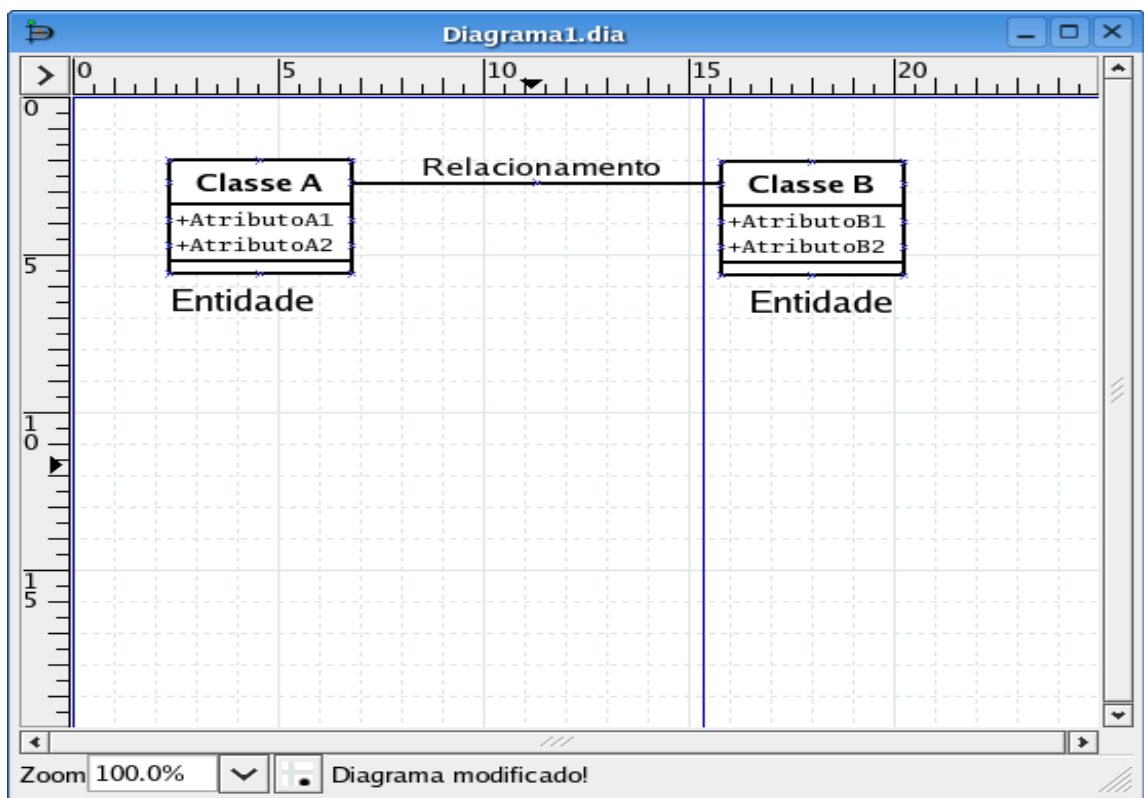
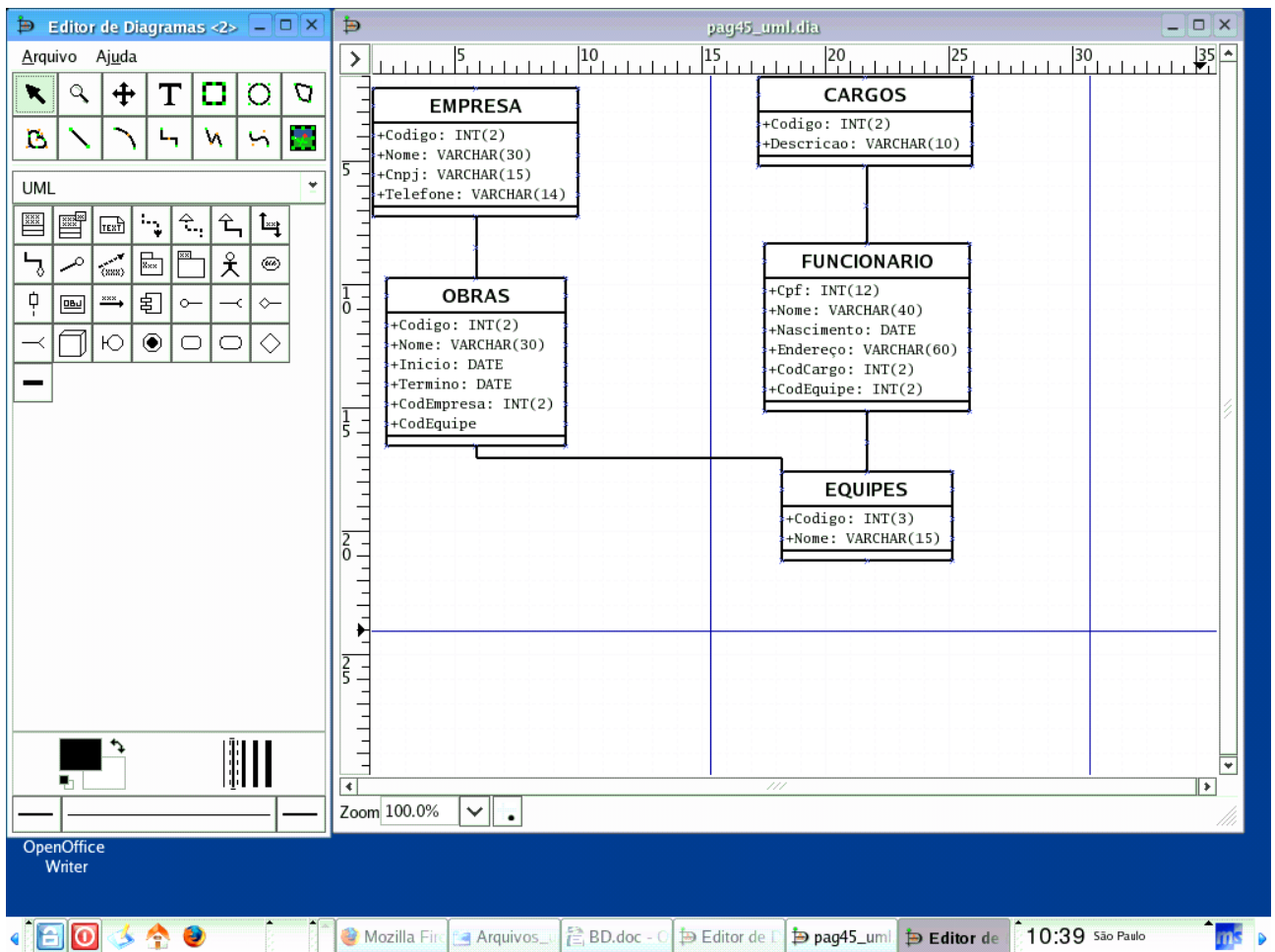
Este é um cenário que descreve uma aplicação real de um sistema de banco de dados. Percebe-se que este apresenta várias entidades e restrições que devem ser respeitadas a fim de que o sistema funcione da forma esperada. Para isto, será criado como exemplo, um modelo ER que descreva todas as particularidades expostas anteriormente, servindo de base para ilustrar os conceitos que envolvem a modelagem de um banco de dados relacional.



## Exemplo em UML

Mudar para opção UML





## Instalando o DBDesigner

DBDesigner é um programa muito bom para quem trabalha com MySQL. Através dele pode-se fazer a modelagem das tabelas, ou seja, representá-las de forma gráfica, bem como seus relacionamentos.

Depois de feita essa modelagem, é possível conectar-se ao MySQL para sincronização. Através do DBDesigner também é possível trabalhar com os dados das tabelas.

O DBDesigner 4 pode ser baixando em:

- <http://www.fabforce.net/downloads.php>

Depois de baixar o arquivo, basta descompactá-lo:

```
$ tar zxfs DBDesigner4.0.5.4.tar.gz
```

Entre no diretório DBDesigner4 e execute o programa:

```
$/DBDesigner4
```

Se tudo correr bem, o DBDesigner já vai ser carregado e você pode usá-lo perfeitamente. Mas infelizmente nem sempre é assim. As vezes costumam acontecer alguns erros. No caso do seguinte erro:

```
libborqt-6.9-qt2.3.so: cannot open shared object file: No such file or directory
```

Segundo o erro, não foi encontrada essa biblioteca libborqt-6.9-qt2.3.so.

É só baixar a biblioteca em:

<http://prdownloads.sourceforge.net/kylixlibs/kylixlibs3-borqt-3.0-2.tar.gz>

Depois de baixar o arquivo descompacte:

```
$ tar zxfs kylixlibs3-borqt-3.0-2.tar.gz
```

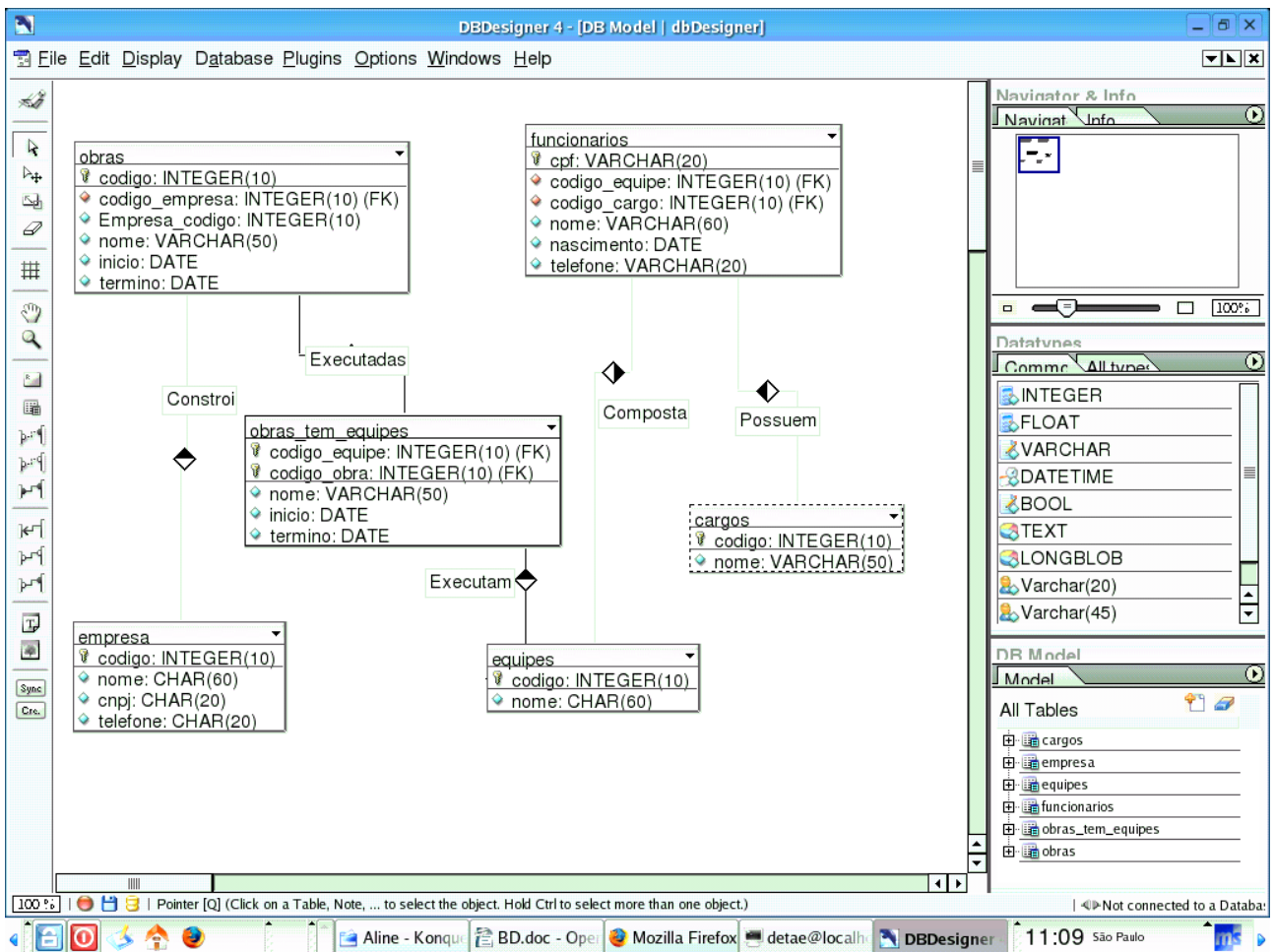
Entre no diretório kylixlibs3-borqt e execute o instalador:

```
# ./install.sh
```

Ela foi instalada em /usr/lib/kylix3/libborqt-6.9.0-qt2.3.so.

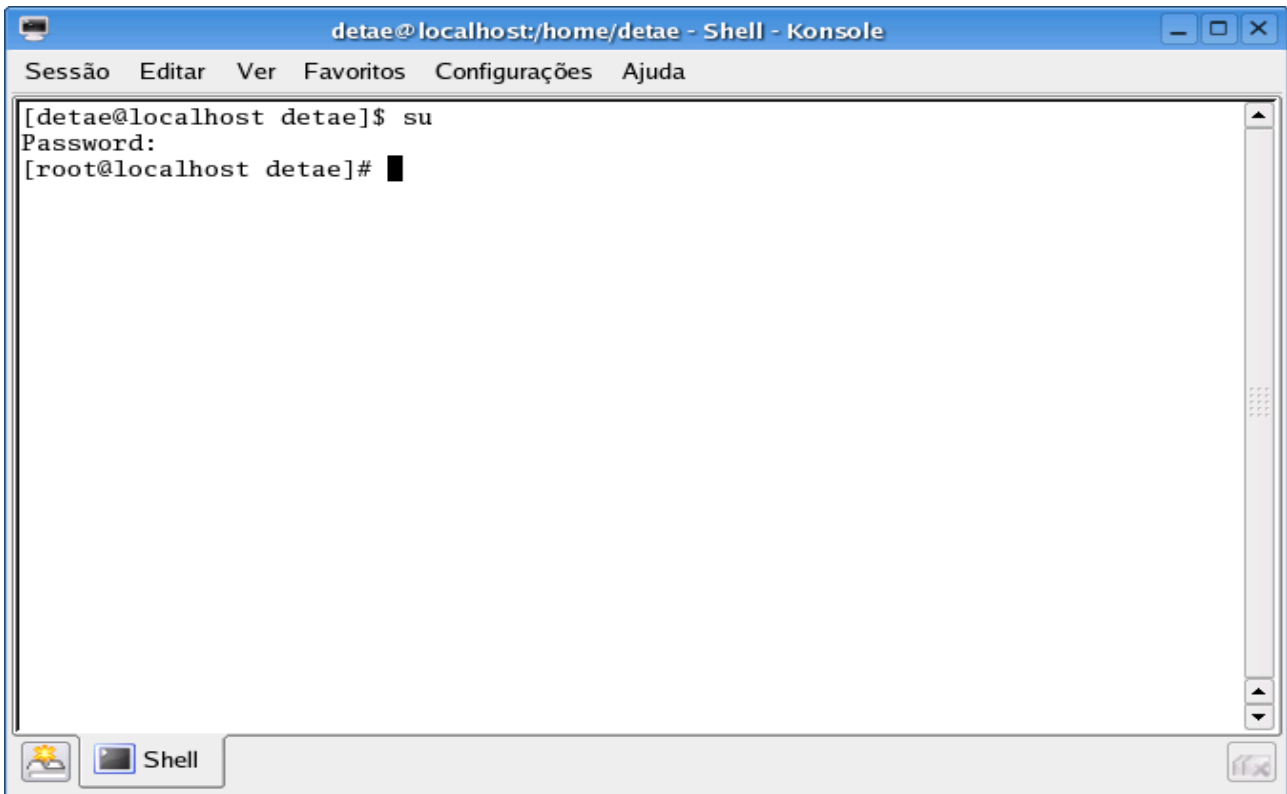
Agora tente executar o DBDesigner, ele deve funcionar normalmente, pois agora pode encontrar a biblioteca.

## Exemplo no DBDesigner:



# MySQL

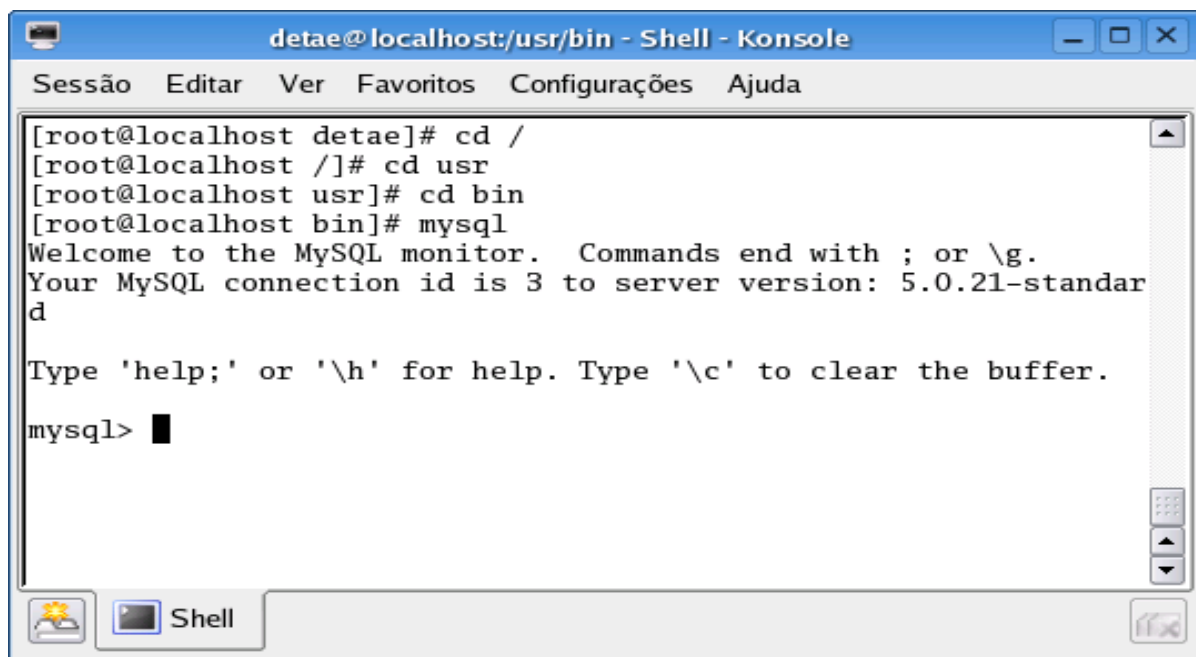
1 – Entre no konsole e mude o usuário para root.



```
detae@localhost:/home/detae - Shell - Konsole
Sessão Editar Ver Favoritos Configurações Ajuda
[detae@localhost detae]$ su
Password:
[root@localhost detae]#
```

2 – Entre na pasta onde o MySQL está instalado

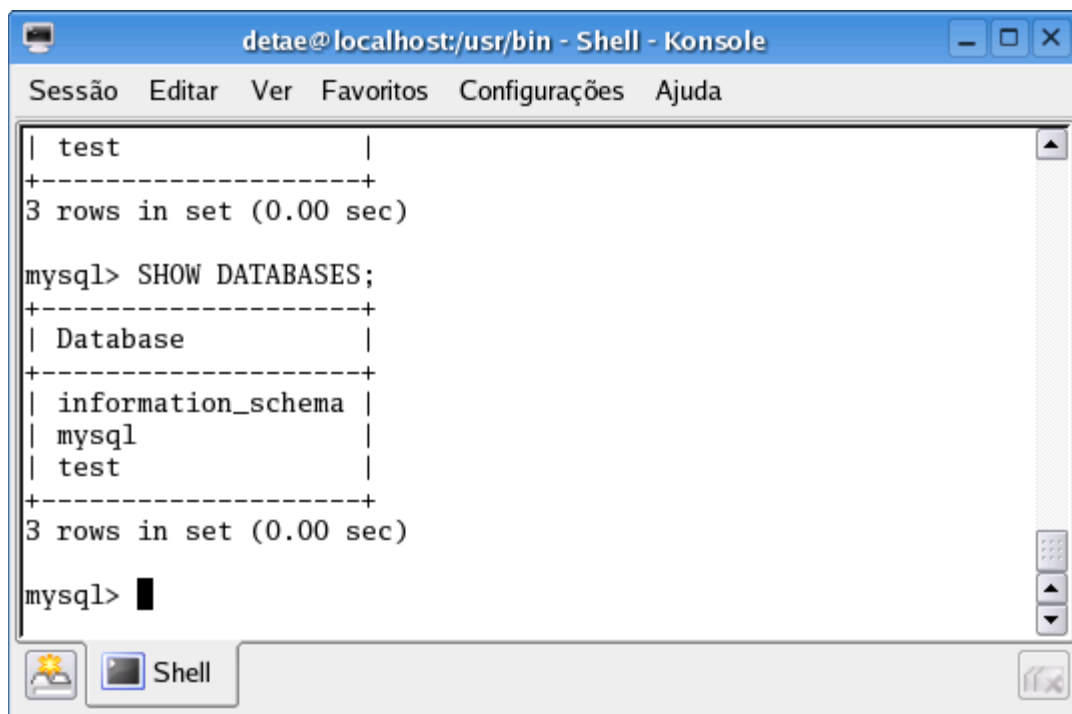
No meu caso, o mysql foi instalado na pasta usr/bin.



```
detae@localhost:/usr/bin - Shell - Konsole
Sessão Editar Ver Favoritos Configurações Ajuda
[root@localhost detae]# cd /
[root@localhost /]# cd usr
[root@localhost usr]# cd bin
[root@localhost bin]# mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 3 to server version: 5.0.21-standar
d

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.
mysql>
```

3 – Visualizando os bancos de dados existentes com o comando:  
**SHOW DATABASES;**



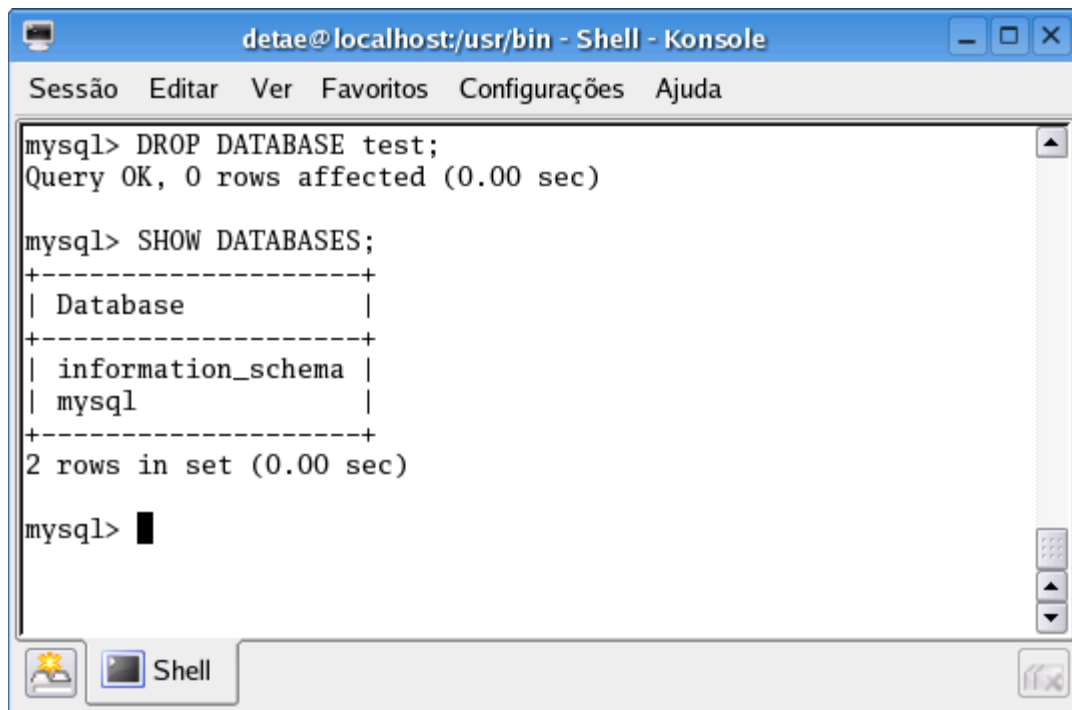
```
detae@localhost:/usr/bin - Shell - Konsole
Sessão  Editar  Ver  Favoritos  Configurações  Ajuda

| test |
+-----+
3 rows in set (0.00 sec)

mysql> SHOW DATABASES;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| test |
+-----+
3 rows in set (0.00 sec)

mysql> █
```

4 – Excluindo um banco de dados existente com o comando:  
**DROP DATABASE test;**



```
detae@localhost:/usr/bin - Shell - Konsole
Sessão  Editar  Ver  Favoritos  Configurações  Ajuda

mysql> DROP DATABASE test;
Query OK, 0 rows affected (0.00 sec)

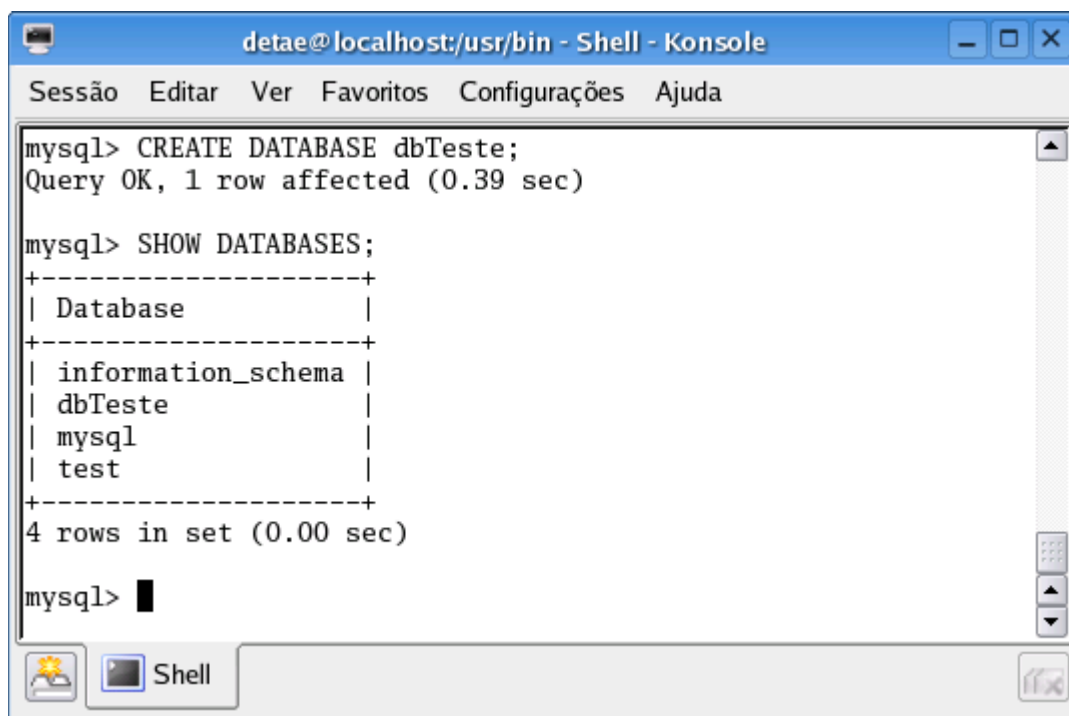
mysql> SHOW DATABASES;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
+-----+
2 rows in set (0.00 sec)

mysql> █
```



5 - Criando um novo banco de dados com o comando:

**CREATE DATABASE dbTeste;**



```
detae@localhost:/usr/bin - Shell - Konsole
Sessão Editar Ver Favoritos Configurações Ajuda

mysql> CREATE DATABASE dbTeste;
Query OK, 1 row affected (0.39 sec)

mysql> SHOW DATABASES;
+-----+
| Database                |
+-----+
| information_schema      |
| dbTeste                 |
| mysql                   |
| test                    |
+-----+
4 rows in set (0.00 sec)

mysql>
```

Foi criado o banco dbTeste, depois foi usando o comando show, para exibir nosso banco já criado.

6 – Para usar o banco, digite o comando :

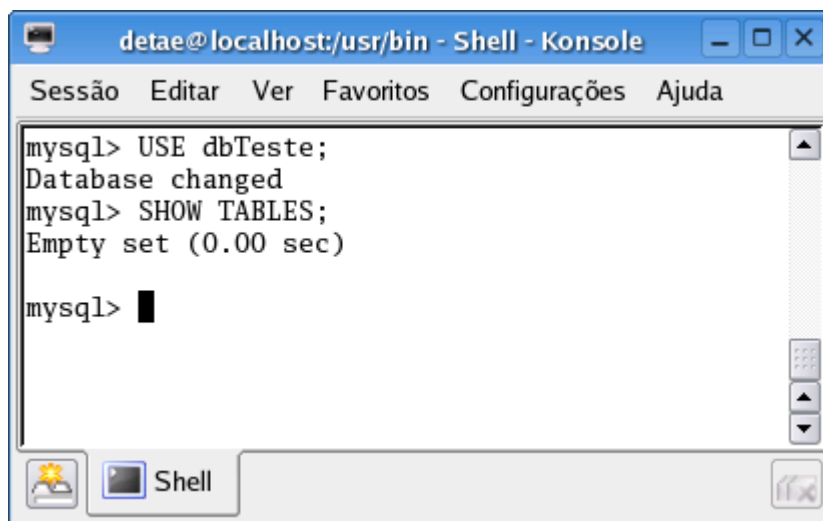
**USE dbTeste**

A partir daí estamos usando o banco de dados que criamos.

Com o comando:

**SHOW TABLES**

Exibimos as tabelas do banco de dados. No nosso caso está vazio, pois ainda não criamos nenhuma tabela.

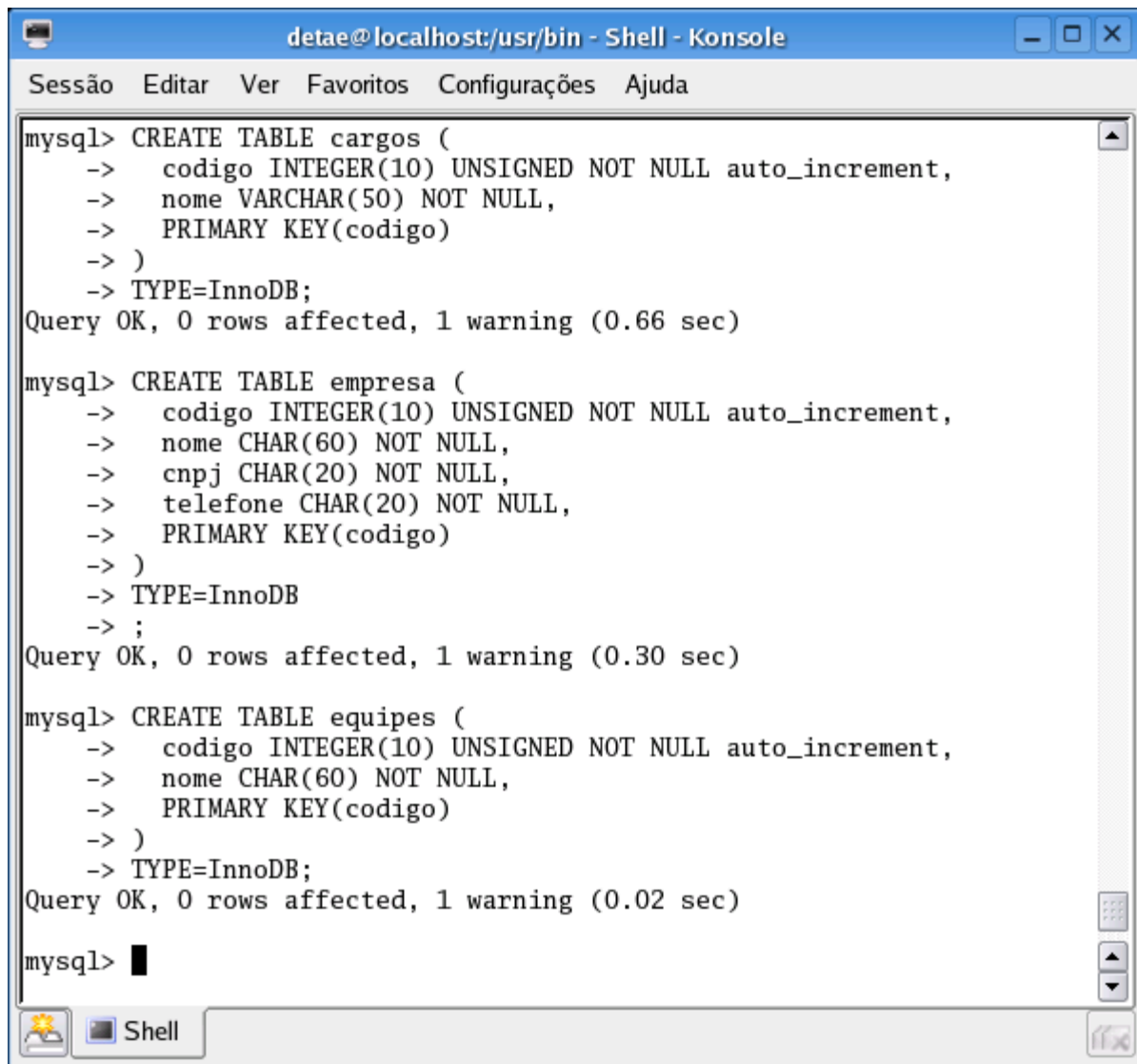


```
detae@localhost:/usr/bin - Shell - Konsole
Sessão Editar Ver Favoritos Configurações Ajuda

mysql> USE dbTeste;
Database changed
mysql> SHOW TABLES;
Empty set (0.00 sec)

mysql>
```

7 – Criando tabelas com o comando :  
**CREATE TABLE**



```
detae@localhost:/usr/bin - Shell - Konsole
Sessão  Editar  Ver  Favoritos  Configurações  Ajuda

mysql> CREATE TABLE cargos (
->   codigo INTEGER(10) UNSIGNED NOT NULL auto_increment,
->   nome VARCHAR(50) NOT NULL,
->   PRIMARY KEY(codigo)
-> )
-> TYPE=InnoDB;
Query OK, 0 rows affected, 1 warning (0.66 sec)

mysql> CREATE TABLE empresa (
->   codigo INTEGER(10) UNSIGNED NOT NULL auto_increment,
->   nome CHAR(60) NOT NULL,
->   cnpj CHAR(20) NOT NULL,
->   telefone CHAR(20) NOT NULL,
->   PRIMARY KEY(codigo)
-> )
-> TYPE=InnoDB
-> ;
Query OK, 0 rows affected, 1 warning (0.30 sec)

mysql> CREATE TABLE equipes (
->   codigo INTEGER(10) UNSIGNED NOT NULL auto_increment,
->   nome CHAR(60) NOT NULL,
->   PRIMARY KEY(codigo)
-> )
-> TYPE=InnoDB;
Query OK, 0 rows affected, 1 warning (0.02 sec)

mysql> █
```

```
detae@localhost:/usr/bin - Shell - Konsole
Sessão Editar Ver Favoritos Configurações Ajuda

mysql> CREATE TABLE funcionarios (
->   cpf VARCHAR(20) NOT NULL,
->   codigo_equipe INTEGER(10) UNSIGNED NOT NULL,
->   codigo_cargo INTEGER(10) UNSIGNED NOT NULL,
->   nome VARCHAR(60) NOT NULL,
->   nascimento DATE NULL DEFAULT '0000-00-00',
->   telefone VARCHAR(20) NULL,
->   PRIMARY KEY(cpf),
->   FOREIGN KEY(codigo_equipe) REFERENCES equipes (codigo) ON DELETE CASCADE ON
-> UPDATE CASCADE,
->   FOREIGN KEY(codigo_cargo) REFERENCES cargos (codigo) ON DELETE CASCADE ON
-> UPDATE CASCADE
-> )
-> TYPE=InnoDB;
Query OK, 0 rows affected, 1 warning (0.30 sec)

mysql> CREATE TABLE obras (
->   codigo INTEGER(10) UNSIGNED NOT NULL,
->   codigo_empresa INTEGER(10) UNSIGNED NOT NULL,
->   nome VARCHAR(50) NOT NULL,
->   inicio DATE NOT NULL DEFAULT '0000-00-00',
->   termino DATE NOT NULL DEFAULT '0000-00-00',
->   PRIMARY KEY(codigo),
->   FOREIGN KEY(codigo_empresa) REFERENCES empresa (codigo) ON DELETE CASCADE ON
-> UPDATE CASCADE
-> )
-> TYPE=InnoDB;
Query OK, 0 rows affected, 1 warning (0.01 sec)

mysql> CREATE TABLE obras_tem_equipes (
->   codigo_equipe INTEGER(10) UNSIGNED NOT NULL,
->   codigo_obra INTEGER(10) UNSIGNED NOT NULL,
->   nome VARCHAR(50) NULL,
->   PRIMARY KEY(codigo_equipe, codigo_obra),
->   FOREIGN KEY(codigo_equipe) REFERENCES equipes (codigo) ON DELETE CASCADE ON
-> UPDATE CASCADE,
->   FOREIGN KEY(codigo_obra) REFERENCES obras (codigo) ON DELETE CASCADE ON
-> UPDATE CASCADE
-> )
-> TYPE=InnoDB;
Query OK, 0 rows affected, 1 warning (0.26 sec)
```

8 – Checando as tabelas criadas com o comando:  
**SHOW TABLES;**

```
detae@localhost:/usr/bin - Shell - Konsole
Sessão Editar Ver Favoritos Configurações Ajuda

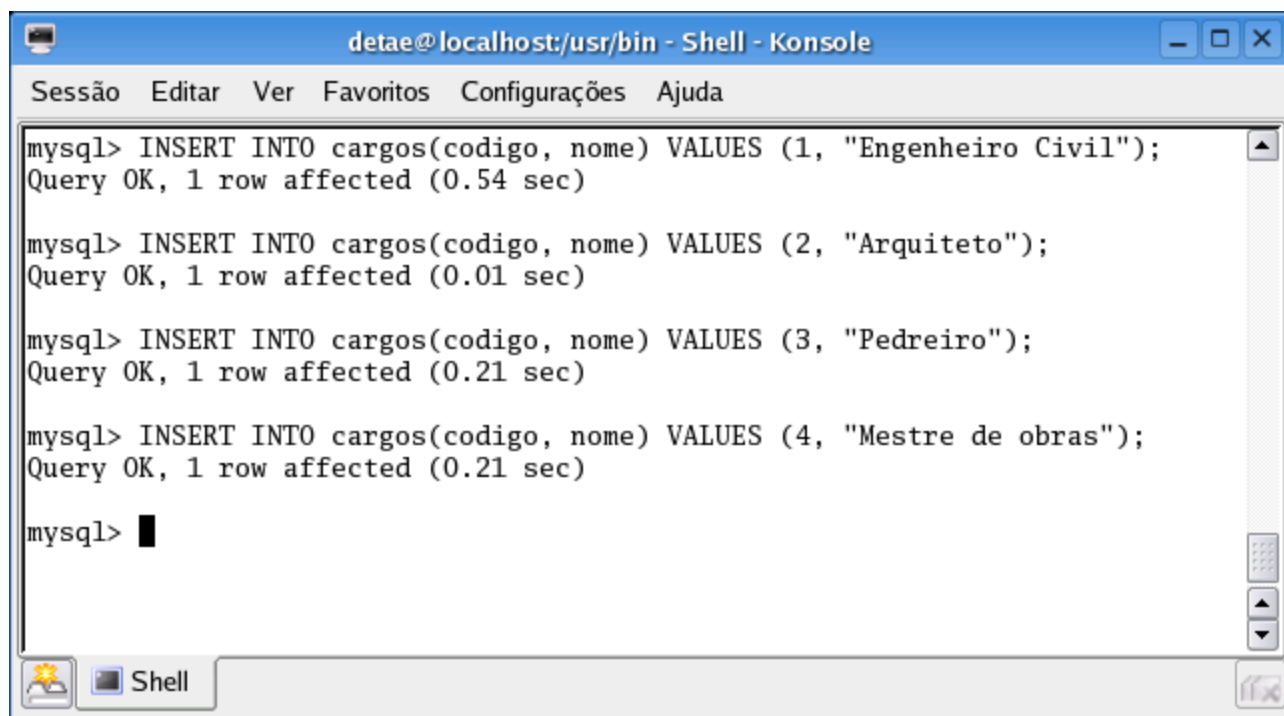
mysql> SHOW TABLES;
+-----+
| Tables_in_dbTeste |
+-----+
| cargos             |
| empresa            |
| equipes            |
| funcionarios       |
| obras              |
| obras_tem_equipes |
+-----+
6 rows in set (0.00 sec)

mysql> █
```

8 – Incluindo dados com o comando:

## INSERT INTO

Inserindo dados na tabela cargos.



```
detae@localhost:/usr/bin - Shell - Konsole
Sessão Editar Ver Favoritos Configurações Ajuda

mysql> INSERT INTO cargos(codigo, nome) VALUES (1, "Engenheiro Civil");
Query OK, 1 row affected (0.54 sec)

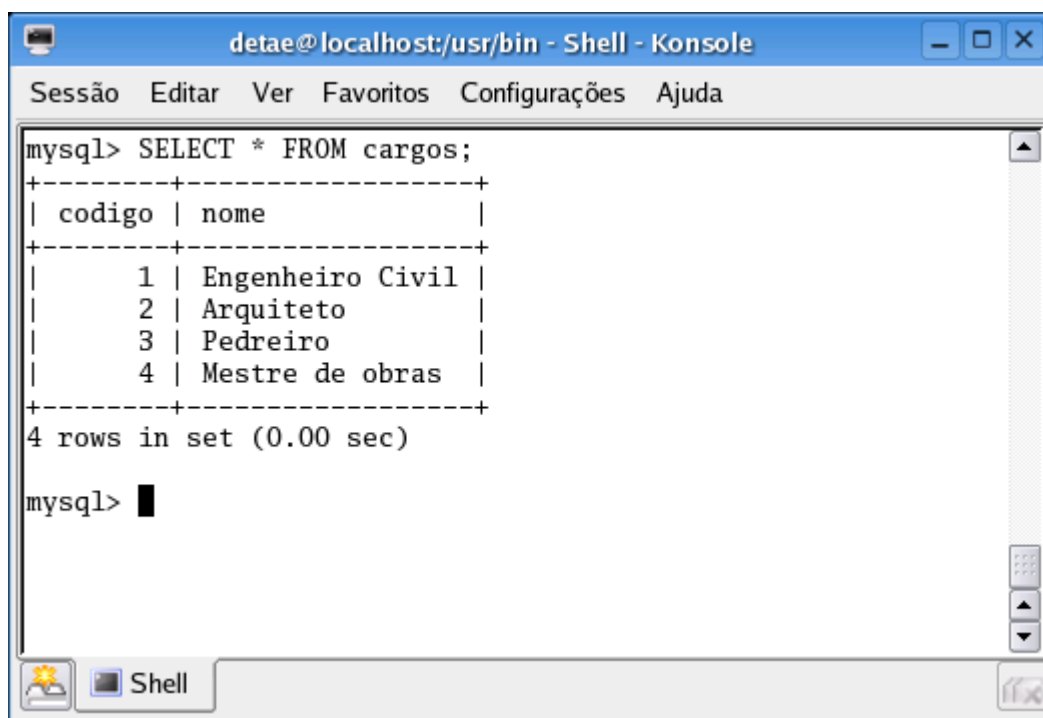
mysql> INSERT INTO cargos(codigo, nome) VALUES (2, "Arquiteto");
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO cargos(codigo, nome) VALUES (3, "Pedreiro");
Query OK, 1 row affected (0.21 sec)

mysql> INSERT INTO cargos(codigo, nome) VALUES (4, "Mestre de obras");
Query OK, 1 row affected (0.21 sec)

mysql> █
```

Visualizando os dados da tabela cargos.



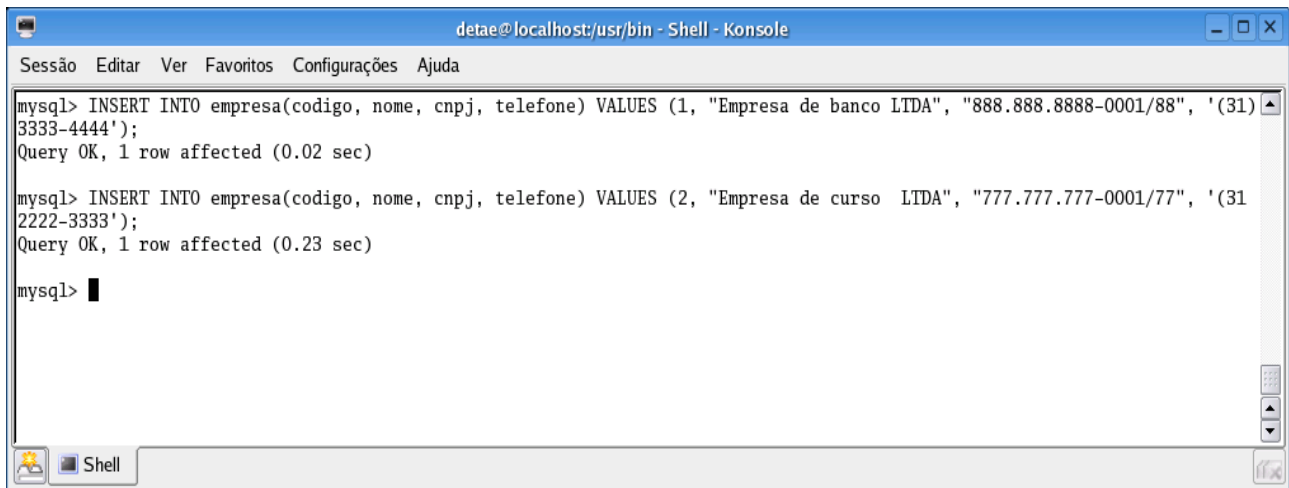
```
detae@localhost:/usr/bin - Shell - Konsole
Sessão Editar Ver Favoritos Configurações Ajuda

mysql> SELECT * FROM cargos;
+-----+-----+
| codigo | nome          |
+-----+-----+
|      1 | Engenheiro Civil |
|      2 | Arquiteto      |
|      3 | Pedreiro       |
|      4 | Mestre de obras |
+-----+-----+
4 rows in set (0.00 sec)

mysql> █
```

Para visualizar os dados inseridos, usamos o comando SELECT, o qual falaremos mais detalhadamente adiante.

Inserindo dados na tabela empresa.

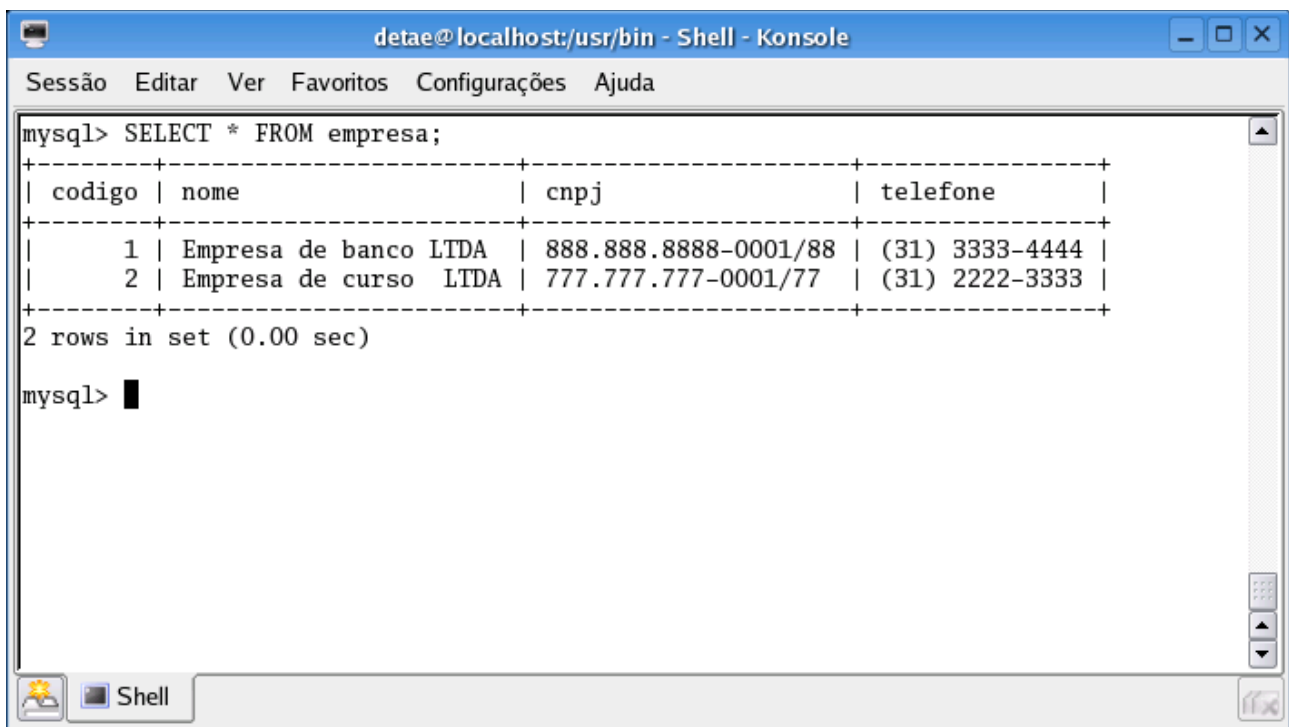


```
detae@localhost:/usr/bin - Shell - Konsole
Sessão Editar Ver Favoritos Configurações Ajuda
mysql> INSERT INTO empresa(codigo, nome, cnpj, telefone) VALUES (1, "Empresa de banco LTDA", "888.888.8888-0001/88", '(31) 3333-4444');
Query OK, 1 row affected (0.02 sec)

mysql> INSERT INTO empresa(codigo, nome, cnpj, telefone) VALUES (2, "Empresa de curso LTDA", "777.777.777-0001/77", '(31) 2222-3333');
Query OK, 1 row affected (0.23 sec)

mysql> █
```

Visualizando os dados da tabela empresa.

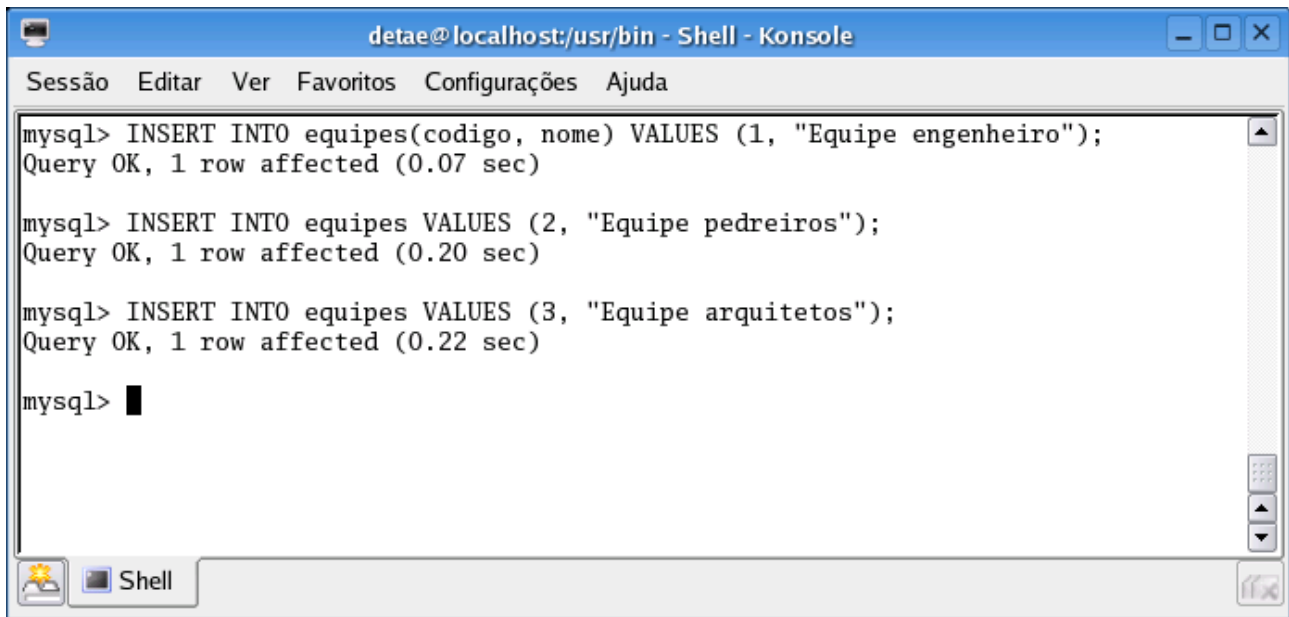


```
detae@localhost:/usr/bin - Shell - Konsole
Sessão Editar Ver Favoritos Configurações Ajuda
mysql> SELECT * FROM empresa;
+-----+-----+-----+-----+
| codigo | nome                | cnpj                | telefone            |
+-----+-----+-----+-----+
|      1 | Empresa de banco LTDA | 888.888.8888-0001/88 | (31) 3333-4444     |
|      2 | Empresa de curso LTDA | 777.777.777-0001/77 | (31) 2222-3333     |
+-----+-----+-----+-----+
2 rows in set (0.00 sec)

mysql> █
```



Inserindo dados na tabela equipes.



```
detae@localhost:/usr/bin - Shell - Konsole
Sessão  Editar  Ver  Favoritos  Configurações  Ajuda

mysql> INSERT INTO equipes(codigo, nome) VALUES (1, "Equipe engenheiro");
Query OK, 1 row affected (0.07 sec)

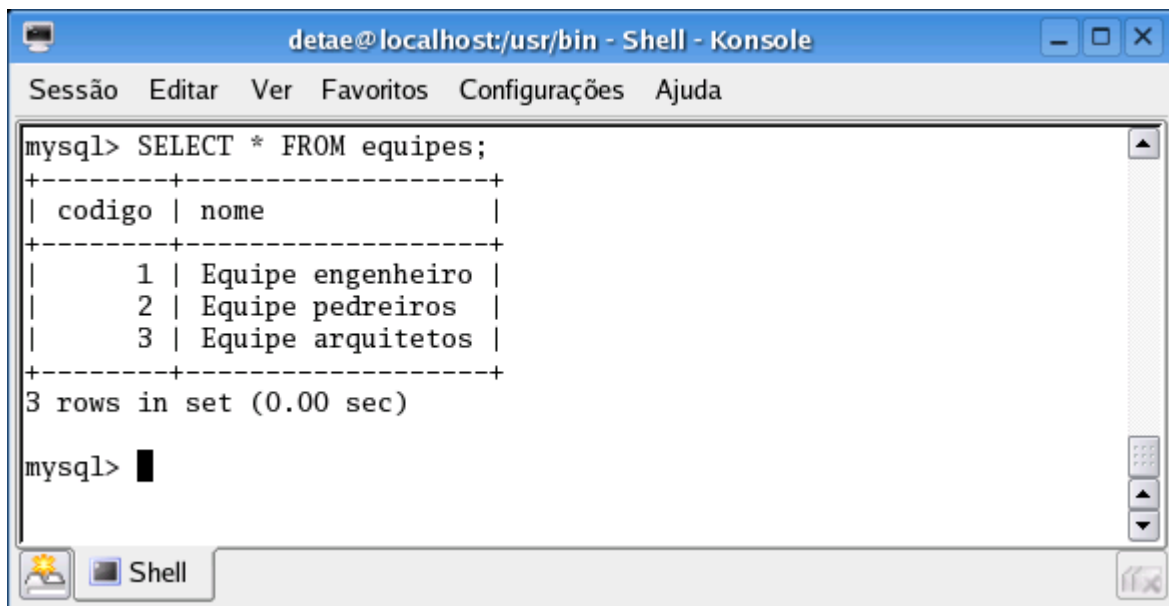
mysql> INSERT INTO equipes VALUES (2, "Equipe pedreiros");
Query OK, 1 row affected (0.20 sec)

mysql> INSERT INTO equipes VALUES (3, "Equipe arquitetos");
Query OK, 1 row affected (0.22 sec)

mysql> █
```

- Observe, que no primeiro INSERT, nós colocamos os nomes das colunas, mas nos demais não colocamos. Não é necessário colocar os nomes das colunas, se você vai inserir a quantidade exata de registro na ordem exata da tabela.

Visualizando os dados da tabela equipes.



```
detae@localhost:/usr/bin - Shell - Konsole
Sessão  Editar  Ver  Favoritos  Configurações  Ajuda

mysql> SELECT * FROM equipes;
+-----+-----+
| codigo | nome          |
+-----+-----+
|      1 | Equipe engenheiro |
|      2 | Equipe pedreiros  |
|      3 | Equipe arquitetos |
+-----+-----+
3 rows in set (0.00 sec)

mysql> █
```

Inserindo dados na tabela funcionários.

```
detae@localhost:/usr/bin - Shell - Konsole
Sessão Editar Ver Favoritos Configurações Ajuda
mysql> INSERT INTO funcionarios (cpf, codigo_equipe, codigo_cargo, nome, nascimento, telefone) VALUES ("111.111.111-11", 1, 1, "José de Alecar", "1970-10-12", "(31)1111-1111");
Query OK, 1 row affected (0.15 sec)

mysql> INSERT INTO funcionarios (cpf, codigo_equipe, codigo_cargo, nome, nascimento, telefone) VALUES ("222.222.222-22", 1, 3, "Paulo Goular", "1978-12-18", "(31)2222-2222");
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO funcionarios VALUES ("333.333.333-33", 2, 3, "Antonio Carlos", "1968-10-18", "(31)3333-3333");
Query OK, 1 row affected (0.21 sec)

mysql> INSERT INTO funcionarios VALUES ("444.444.444-44", 4, 2, "Ana Claudia", "1969-09-28", "(31)4444-4444");
Query OK, 1 row affected (0.00 sec)

mysql>
```

Visualizando os dados da tabela funcionários.

```
detae@localhost:/usr/bin - Shell - Konsole
Sessão Editar Ver Favoritos Configurações Ajuda
mysql> select * from funcionarios;
+-----+-----+-----+-----+-----+-----+
| cpf          | codigo_equipe | codigo_cargo | nome          | nascimento | telefone |
+-----+-----+-----+-----+-----+-----+
| 111.111.111-11 | 1             | 1             | José de Alecar | 1970-10-12 | (31)1111-1111 |
| 222.222.222-22 | 1             | 3             | Paulo Goular  | 1978-12-18 | (31)2222-2222 |
| 333.333.333-33 | 2             | 3             | Antonio Carlos | 1968-10-18 | (31)3333-3333 |
| 444.444.444-44 | 3             | 2             | Ana Claudia   | 1969-09-28 | (31)4444-4444 |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)

mysql>
```

Inserindo dados na tabela obras.

```
detae@localhost:/usr/bin - Shell - Konsole
Sessão Editar Ver Favoritos Configurações Ajuda
mysql> INSERT INTO obras VALUES (1, 1, "Ponte", "2005-01-01", "2005-06-01");
Query OK, 1 row affected (0.21 sec)

mysql> INSERT INTO obras VALUES (2, 1, 'Prédio', '2005-11-09', '2006-06-01');
Query OK, 1 row affected (0.00 sec)

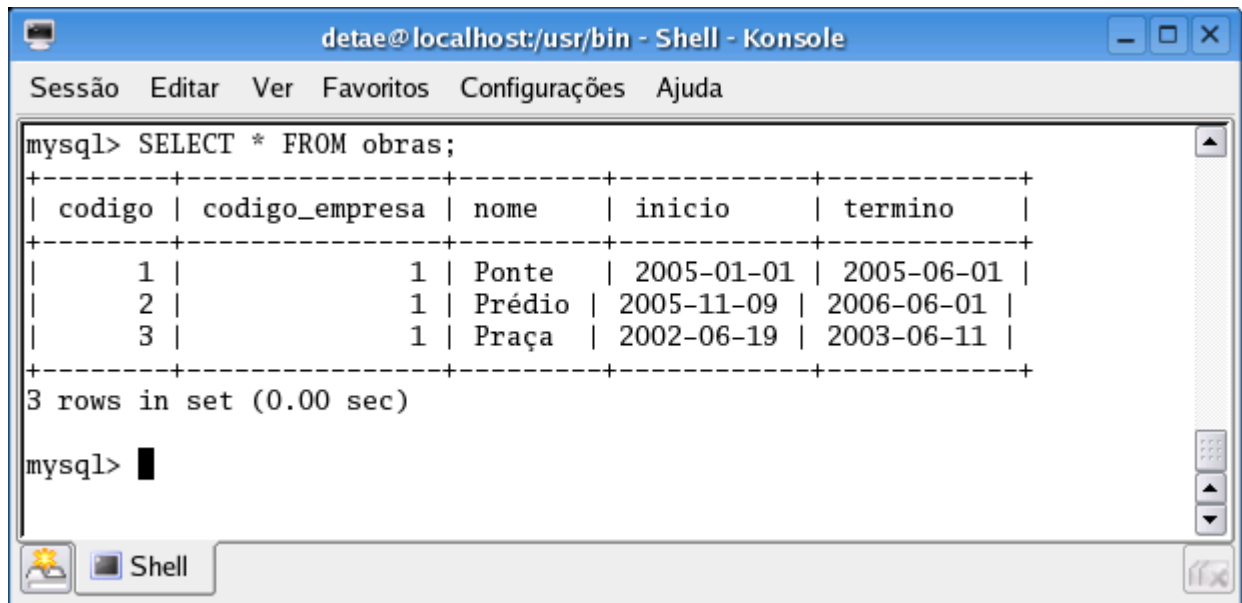
mysql> INSERT INTO obras VALUES (3, 1, 'Praça', '2002-06-19', '2003-06-11');
Query OK, 1 row affected (0.23 sec)

mysql>
```

– Observe que podemos usar aspas simples, ao invés de aspas duplas.



Visualizando os dados da tabela obras.

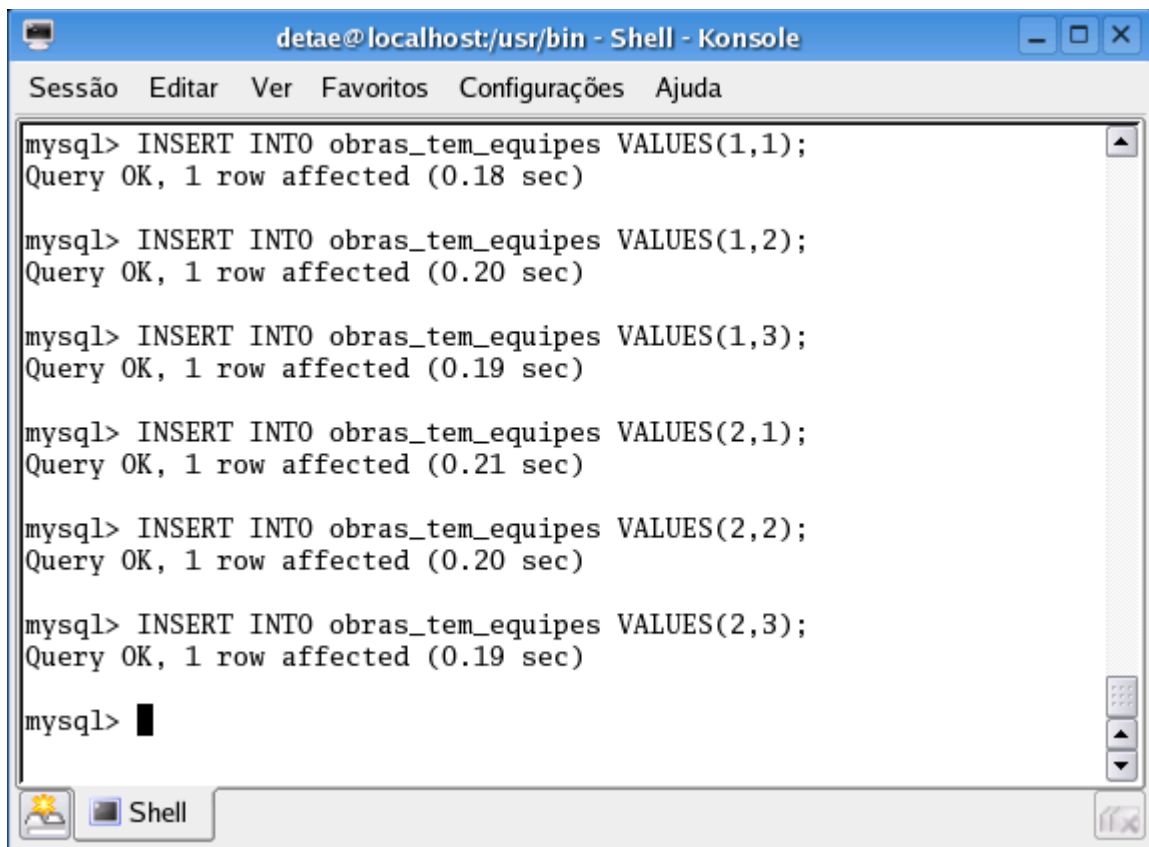


```
detae@localhost:/usr/bin - Shell - Konsole
Sessão  Editar  Ver  Favoritos  Configurações  Ajuda

mysql> SELECT * FROM obras;
+-----+-----+-----+-----+-----+
| codigo | codigo_empresa | nome   | inicio   | termino |
+-----+-----+-----+-----+-----+
|      1 |              1 | Ponte  | 2005-01-01 | 2005-06-01 |
|      2 |              1 | Prédio | 2005-11-09 | 2006-06-01 |
|      3 |              1 | Praça  | 2002-06-19 | 2003-06-11 |
+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)

mysql> █
```

Inserindo dados na tabela obras\_tem\_equipes.



```
detae@localhost:/usr/bin - Shell - Konsole
Sessão  Editar  Ver  Favoritos  Configurações  Ajuda

mysql> INSERT INTO obras_tem_equipes VALUES(1,1);
Query OK, 1 row affected (0.18 sec)

mysql> INSERT INTO obras_tem_equipes VALUES(1,2);
Query OK, 1 row affected (0.20 sec)

mysql> INSERT INTO obras_tem_equipes VALUES(1,3);
Query OK, 1 row affected (0.19 sec)

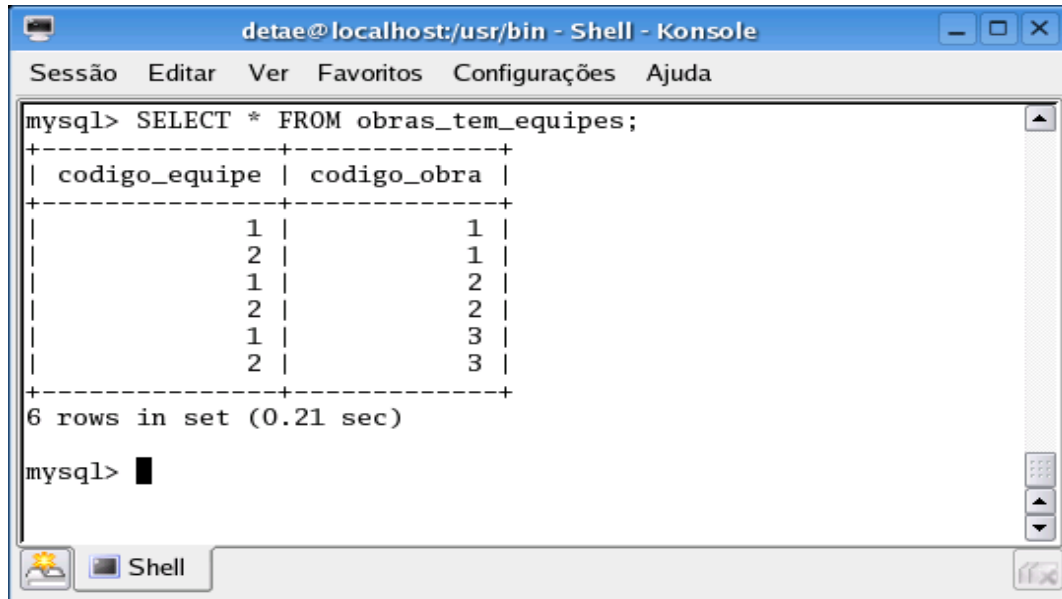
mysql> INSERT INTO obras_tem_equipes VALUES(2,1);
Query OK, 1 row affected (0.21 sec)

mysql> INSERT INTO obras_tem_equipes VALUES(2,2);
Query OK, 1 row affected (0.20 sec)

mysql> INSERT INTO obras_tem_equipes VALUES(2,3);
Query OK, 1 row affected (0.19 sec)

mysql> █
```

Visualizando os dados da tabela obras\_tem\_equipes;

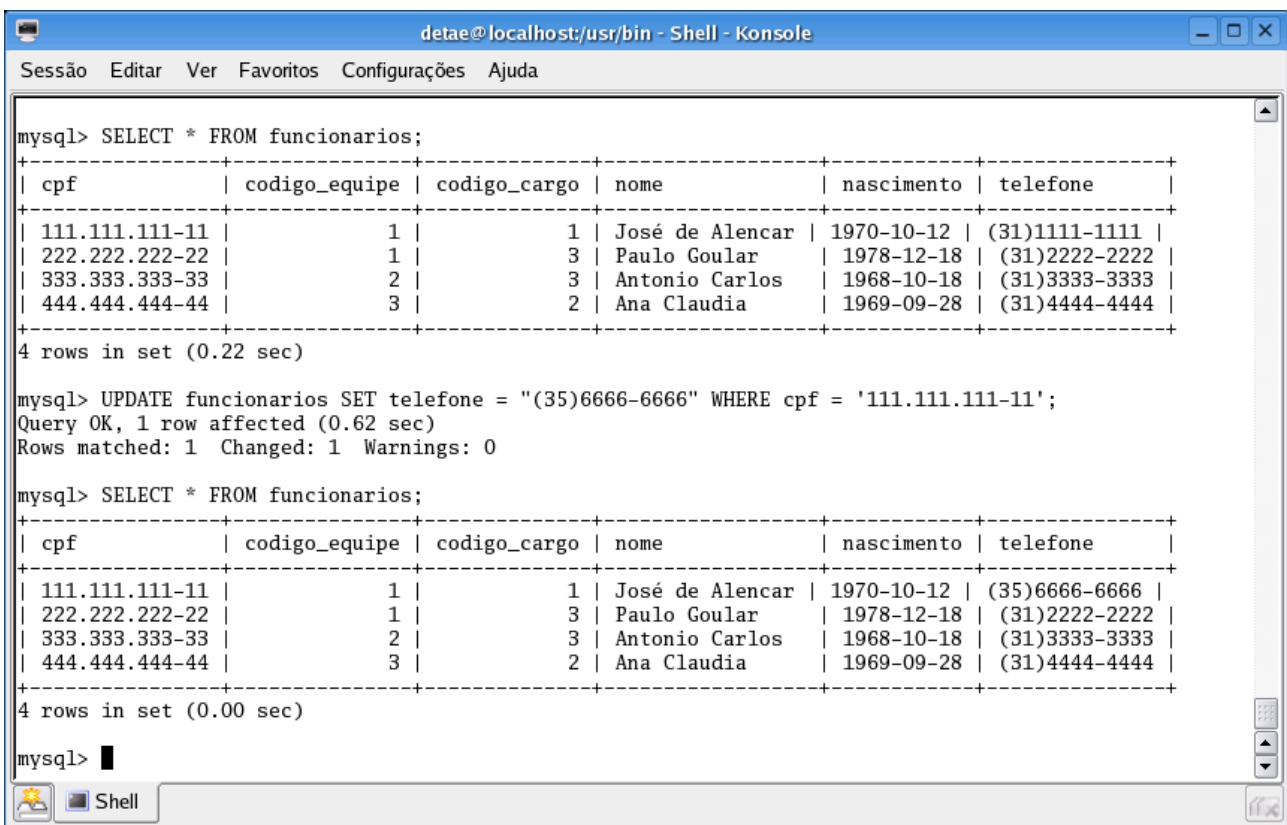


```
detae@localhost:/usr/bin - Shell - Konsole
Sessão Editar Ver Favoritos Configurações Ajuda

mysql> SELECT * FROM obras_tem_equipes;
+-----+-----+
| codigo_equipe | codigo_obra |
+-----+-----+
| 1 | 1 |
| 2 | 1 |
| 1 | 2 |
| 2 | 2 |
| 1 | 3 |
| 2 | 3 |
+-----+-----+
6 rows in set (0.21 sec)

mysql>
```

9 – Alterando dados com o comando:  
**UPDATE**



```
detae@localhost:/usr/bin - Shell - Konsole
Sessão Editar Ver Favoritos Configurações Ajuda

mysql> SELECT * FROM funcionarios;
+-----+-----+-----+-----+-----+-----+
| cpf          | codigo_equipe | codigo_cargo | nome          | nascimento | telefone |
+-----+-----+-----+-----+-----+-----+
| 111.111.111-11 | 1 | 1 | José de Alencar | 1970-10-12 | (31)1111-1111 |
| 222.222.222-22 | 1 | 3 | Paulo Goular   | 1978-12-18 | (31)2222-2222 |
| 333.333.333-33 | 2 | 3 | Antonio Carlos | 1968-10-18 | (31)3333-3333 |
| 444.444.444-44 | 3 | 2 | Ana Claudia   | 1969-09-28 | (31)4444-4444 |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.22 sec)

mysql> UPDATE funcionarios SET telefone = "(35)6666-6666" WHERE cpf = '111.111.111-11';
Query OK, 1 row affected (0.62 sec)
Rows matched: 1 Changed: 1 Warnings: 0

mysql> SELECT * FROM funcionarios;
+-----+-----+-----+-----+-----+-----+
| cpf          | codigo_equipe | codigo_cargo | nome          | nascimento | telefone |
+-----+-----+-----+-----+-----+-----+
| 111.111.111-11 | 1 | 1 | José de Alencar | 1970-10-12 | (35)6666-6666 |
| 222.222.222-22 | 1 | 3 | Paulo Goular   | 1978-12-18 | (31)2222-2222 |
| 333.333.333-33 | 2 | 3 | Antonio Carlos | 1968-10-18 | (31)3333-3333 |
| 444.444.444-44 | 3 | 2 | Ana Claudia   | 1969-09-28 | (31)4444-4444 |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)

mysql>
```

- Primeiro exibe os dados da tabela funcionários antes da alteração;
- Faz o UPDATE
- Depois exibe novamente os dados da tabela funcionários, já com o telefone alterado.

## 10 – Excluindo dados com o comando DELETE

```
detae@localhost:/usr/bin - Shell - Konsole
Sessão Editar Ver Favoritos Configurações Ajuda

mysql> SELECT * FROM empresa;
+-----+-----+-----+-----+
| codigo | nome                | cnpj                | telefone          |
+-----+-----+-----+-----+
|      1 | Empresa de banco LTDA | 888.888.8888-0001/88 | (31) 3333-4444 |
|      2 | Empresa de curso LTDA | 777.777.777-0001/77 | (31) 2222-3333 |
+-----+-----+-----+-----+
2 rows in set (1.52 sec)

mysql> DELETE FROM empresa WHERE codigo = 2;
Query OK, 1 row affected (0.41 sec)

mysql> SELECT * FROM empresa;
+-----+-----+-----+-----+
| codigo | nome                | cnpj                | telefone          |
+-----+-----+-----+-----+
|      1 | Empresa de banco LTDA | 888.888.8888-0001/88 | (31) 3333-4444 |
+-----+-----+-----+-----+
1 row in set (0.00 sec)

mysql>
```

## 11 – Comando SELECT:

```
detae@localhost:/usr/bin - Shell - Konsole
Sessão Editar Ver Favoritos Configurações Ajuda

mysql> SELECT * FROM equipes;
+-----+-----+
| codigo | nome                |
+-----+-----+
|      1 | Equipe engenheiro  |
|      2 | Equipe pedreiros   |
|      3 | Equipe arquitetos  |
+-----+-----+
3 rows in set (1.00 sec)

mysql> SELECT * FROM funcionarios;
+-----+-----+-----+-----+-----+-----+
| cpf          | codigo_equipe | codigo_cargo | nome                | nascimento | telefone |
+-----+-----+-----+-----+-----+-----+
| 111.111.111-11 | 1 | 1 | José de Alencar | 1970-10-12 | (35)6666-6666 |
| 222.222.222-22 | 1 | 3 | Paulo Goular   | 1978-12-18 | (31)2222-2222 |
| 333.333.333-33 | 2 | 3 | Antonio Carlos | 1968-10-18 | (31)3333-3333 |
| 444.444.444-44 | 3 | 2 | Ana Claudia   | 1969-09-28 | (31)4444-4444 |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.21 sec)

mysql> SELECT nome, cpf FROM funcionarios;
+-----+-----+
| nome                | cpf          |
+-----+-----+
| José de Alencar   | 111.111.111-11 |
| Paulo Goular      | 222.222.222-22 |
| Antonio Carlos    | 333.333.333-33 |
| Ana Claudia       | 444.444.444-44 |
+-----+-----+
4 rows in set (0.02 sec)

mysql>
```

## Selecionando registros com o **WHERE**

```
detae@localhost:/usr/bin - Shell - Konsole
Sessão Editar Ver Favoritos Configurações Ajuda

mysql> SELECT * FROM funcionarios WHERE codigo_cargo = 3;
+-----+-----+-----+-----+-----+-----+
| cpf          | codigo_equipe | codigo_cargo | nome          | nascimento | telefone    |
+-----+-----+-----+-----+-----+-----+
| 222.222.222-22 | 1 | 3 | Paulo Goular | 1978-12-18 | (31)2222-2222 |
| 333.333.333-33 | 2 | 3 | Antonio Carlos | 1968-10-18 | (31)3333-3333 |
+-----+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)

mysql> SELECT nome, codigo_cargo, nascimento FROM funcionarios WHERE codigo_cargo = 3 and nascimento >= '1978-01-01';
+-----+-----+-----+
| nome          | codigo_cargo | nascimento |
+-----+-----+-----+
| Paulo Goular | 3 | 1978-12-18 |
+-----+-----+-----+
1 row in set (0.00 sec)

mysql> SELECT nome FROM funcionarios WHERE nome LIKE "Ana%";
+-----+
| nome          |
+-----+
| Ana Claudia  |
+-----+
1 row in set (0.00 sec)

mysql>
```

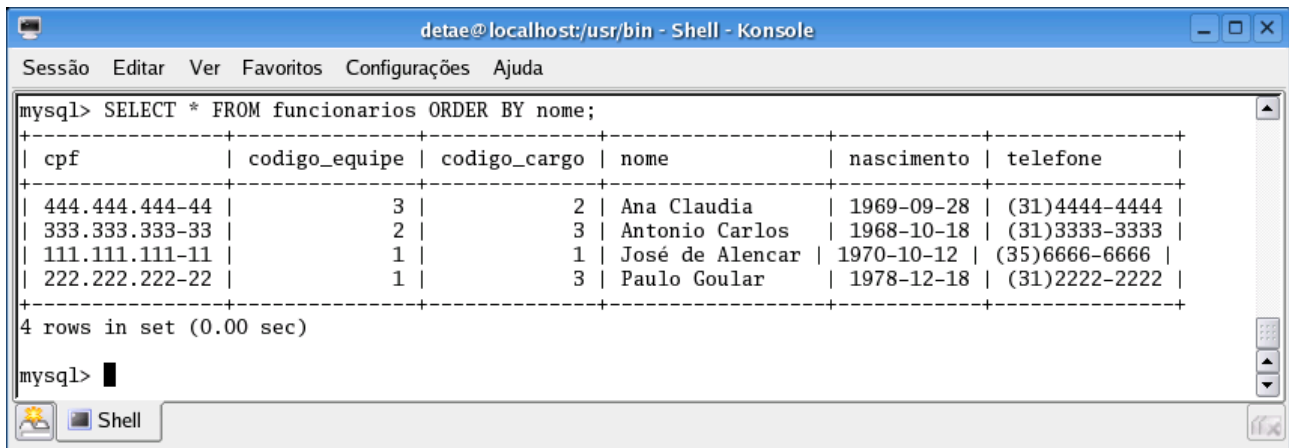
## Agrupando com o **GROUP BY**

```
detae@localhost:/usr/bin - Shell - Konsole
Sessão Editar Ver Favoritos Configurações Ajuda

mysql> SELECT codigo_obra, COUNT(*) FROM obras_tem_equipes GROUP BY codigo_obra;
+-----+-----+
| codigo_obra | COUNT(*) |
+-----+-----+
| 1 | 2 |
| 2 | 2 |
| 3 | 2 |
+-----+-----+
3 rows in set (0.22 sec)

mysql>
```

## Ordenando com o **ORDER BY**



The screenshot shows a terminal window titled "detae@localhost:/usr/bin - Shell - Konsole". The terminal displays a MySQL command and its output. The command is "mysql> SELECT \* FROM funcionarios ORDER BY nome;". The output is a table with 6 columns: cpf, codigo\_equipe, codigo\_cargo, nome, nascimento, and telefone. The results are sorted by the 'nome' column. Below the table, it says "4 rows in set (0.00 sec)". The terminal prompt "mysql>" is visible at the bottom.

```
mysql> SELECT * FROM funcionarios ORDER BY nome;
```

cpf	codigo_equipe	codigo_cargo	nome	nascimento	telefone
444.444.444-44	3	2	Ana Claudia	1969-09-28	(31)4444-4444
333.333.333-33	2	3	Antonio Carlos	1968-10-18	(31)3333-3333
111.111.111-11	1	1	José de Alencar	1970-10-12	(35)6666-6666
222.222.222-22	1	3	Paulo Goular	1978-12-18	(31)2222-2222

4 rows in set (0.00 sec)

```
mysql>
```