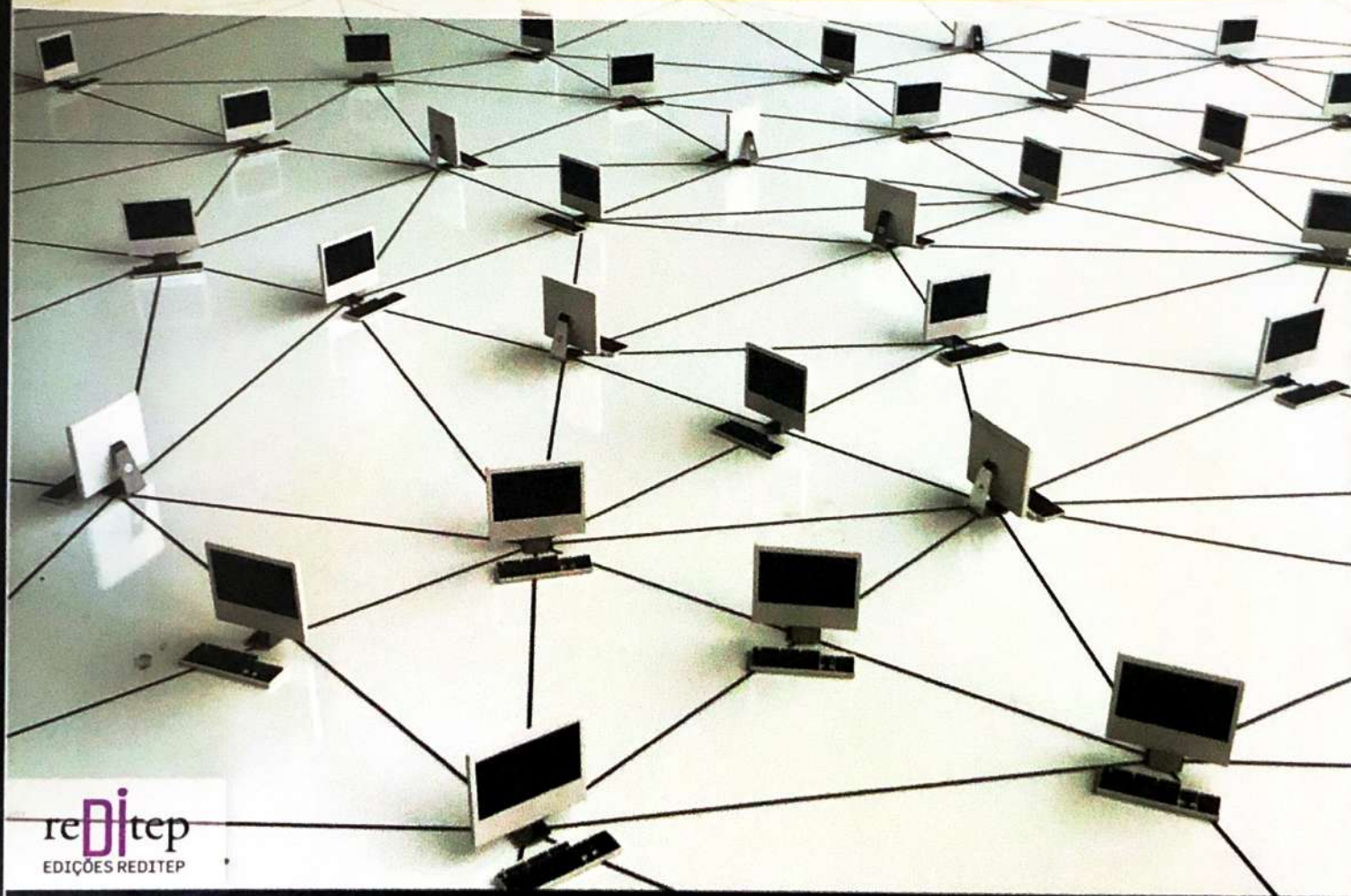
10ª CLASSE

REPÚBLICA DE ANGOLA | MINISTÉRIO DA EDUCAÇÃO

BASE DE DADOS E REDES DE COMPUTADORES 10

TEXTOS DE APOIO AO ALUNO



reditep
EDIÇÕES REDITEP

RETEP | REFORMA DO ENSINO TÉCNICO-PROFISSIONAL

Índice

Capítulo 1

BASES DE DADOS

CONTEÚDO

1.1 Introdução às bases de dados.

1.2 Tipos de base de dados.

1.3 Sistema de gestão de base de dados relacionais.

1.4 Utilização de uma ferramenta de base de dados.

1.5 Importação, ligação e exportação de dados.

OBJECTIVOS

- Conhecer as características principais de uma base de dados.
- Conhecer o ambiente de concepção de sistemas de gestão de bases de dados.
- Criar tabelas.
- Desenvolver formulários.
- Realizar consultas em bases de dados.
- Criar relatórios.
- Importar, ligar e exportar dados.

O desenvolvimento dos Sistemas de Gestão de Base de Dados Orientado a Objectos (SGBDO) teve origem na combinação de ideias dos modelos de dados tradicionais e de linguagens de programação orientada a objectos.

Os modelos de dados orientados a objectos são mais adequados para o tratamento de objectos complexos e dinâmicos, por possuírem maior naturalidade conceitual e por estarem em linha com as tendências em linguagens de programação e engenharia de *software*. No entanto, o modelo relacional foi evoluindo e gerou o modelo relacional estendido, que promove algumas características de Orientação por Objectos nas bases de dados e daí o modelo continuar a ser dos mais utilizados.

1.3 SISTEMA DE GESTÃO DE BASE DE DADOS RELACIONAIS

1.3.1 GESTÃO E ORGANIZAÇÃO DA INFORMAÇÃO NUMA BASE DE DADOS

Uma base de dados é considerada como um sistema de armazenamento de dados relacionados entre si, com redundância controlada, acessíveis e estruturados sobre a forma de ficheiro de dados ou tabelas.

Completando um pouco a definição apresentada, uma base de dados é uma colecção lógica e coerente de dados, sempre com um significado implícito.

O ciclo de vida da base de dados é a expressão utilizada para designar todos os eventos que acontecem desde a primeira vez que é reconhecida a necessidade de uma base de dados. Existe um modelo tradicional que identifica as fases do ciclo de vida, este modelo do ciclo de vida é constituído por oito fases. Apenas podemos passar à fase seguinte depois da anterior estar concluída; no entanto, por vezes, surge a necessidade de retroceder à fase anterior, para realizar determinados ajustes.

O modelo relacional de base de dados é actualmente o modelo de implementação mais utilizado. Trata-se de um modelo bastante potente e, ao mesmo tempo, bastante simples, que não representa problemas. O elemento principal deste modelo é a relação. Por tanto, podemos dizer que uma base de dados relacional é composto por um conjunto de relações.

Noção de relação

Associação estabelecida entre campos comuns (colunas) de duas tabelas, permitindo que a consciência da informação seja garantida.

O relacionamento entre os campos comuns das tabelas permite garantir que a consciência da informação não seja colocada em causa e, também, associar os dados de duas ou mais tabelas para a visualização, edição ou impressão da informação.

Noção de associação

Representa a forma como duas ou mais entidades se relacionam entre si.

Existem três tipos de associações:

- Unárias – relação entre uma entidade e ela própria.
- Binárias – existência de um qualquer tipo de relação entre duas entidades.
- Complexas – possibilidade de estabelecer relações entre mais do que duas entidades.

E uma Relação ...

A relação representa na forma de uma tabela; esta representa o que no modelo entidade-relação chamamos de entidade. Esta contém os atributos (colunas) e as tuplas (linhas/registos).

- Atributo: trata-se de cada uma das colunas da tabela; são definidas por um nome e podem conter um conjunto de valores.
- Tupla: trata-se de cada uma das linhas da tabela; é importante assinalar que não se podem ter tuplas duplicadas numa tabela.

Vistas

Trata-se de uma tabela fictícia, a qual mostra atributos de outras tabelas relacionadas. Desta forma, obtemos os dados que nos interessam de uma ou várias tabelas. É importante salientar que não se podem realizar operações sobre vistas.

1.3.2 CONCEITO DE ENTIDADE, TABELA, CAMPO E REGISTO

O diagrama de ENTIDADE-ASSOCIAÇÃO

A Entidade é um conceito abstracto. Pretende representar a realidade que queremos modelar. A definição de uma entidade passa pela identificação dos elementos e de um conjunto de atributos comuns do mundo real que estamos a analisar. Por exemplo, vamos considerar uma Escola. Nessa Escola, e dependendo do objectivo a alcançar, podemos identificar, as entidades ALUNO, PROFESSOR e DISCIPLINA. A entidade ALUNO representa todos os alunos da Escola e o aluno "António Buengo" é uma instância da entidade ALUNO.

Os Atributos representam os dados da entidade. Por exemplo, para representar a entidade ALUNO, torna-se necessário definir os atributos nome, apelido, morada, telefone, entre outros atributos. E cada atributo encontra-se definido num determinado domínio de valores. Assim, o atributo nome é definido por um conjunto de caracteres, telefone, é definido por um conjunto de nove números. Um conjunto de atributos, que fazem parte de uma Entidade, encontrámos um ou vários que no seu conjunto são particulares e chamamos de Chave Pri-

Nota: numa Entidade não podem existir dois atributos com a mesma designação

Alguns exemplos de SGBD's empresariais são ORACLE, Microsoft SQL Server e DB2 da IBM. Para computadores pessoais ou pequenas organizações temos, entre os mais conhecidos, o MySQL, Postgree e Microsoft Access. Os primeiros têm mais ferramentas e funcionalidades que se adequam ao mercado empresarial.

Em resumo, os sistemas de gestão de base de dados são programas que permitem criar e manipular as bases de dados, em que os dados estão estruturados e permitem o acesso a programas para a sua gestão.

1.2 TIPOS DE BASE DE DADOS

1.2.1 MODELO DE DADOS HIERÁRQUICOS

O modelo de dados hierárquico utiliza árvores para a sua representação dos dados. A árvore é construída por nós e o nível mais alto da árvore chamamos de raiz. Cada nó representa uma entidade com os seus campos.

A representação gráfica deste modelo é uma árvore invertida.

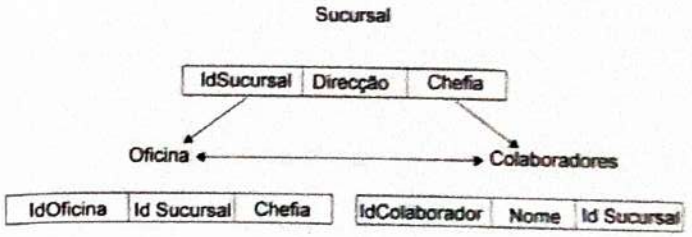
Representa só relações 1:N, por isso apresenta vários inconvenientes:

- Não tem relações N:N (muitos para muitos).
- Um nó não pode ter mais de um nó pai.
- Não permite mais de uma relação entre dois nós.
- Para aceder a qualquer segmento é necessário começar pelo nó raiz.
- A árvore só se consulta na ordem definida.

1.2.2 MODELO DE DADOS EM REDE

O modelo de dados em Rede representa-se como nós e suas relações são as linhas que os unem. Nesta estrutura, qualquer nó se pode relacionar com outros.

A principal diferença entre este modelo e o anterior é que o nó filho pode ter vários nós pai.



Assim, registamos sobre este modelo que:

- o tipo de entidade é representado num nó;
- o Elemento é um campo de dados.

Este modelo de dados permite representar relações N:N.

1.2.3 MODELO DE DADOS RELACIONAL

Este modelo é o mais utilizado uma vez que utiliza tabelas bidimensionais para a representação lógica dos dados e das suas relações.

O elemento principal deste modelo é a relação que é representada numa tabela.

CAMPO1	CAMPO2	CAMPO3
Valor1.1	Valor1.2	Valor1.3
Valor2.1	Valor2.2	Valor2.3

1.2.4 ORIENTADAS AO OBJECTO

No modelo orientado ao objecto (SGBDO, Sistema de Gestão de Bases de Dados por Objecto), os dados são armazenados sob a forma de objectos, quer dizer, de estruturas chamadas classes que apresentam dados membros. Os campos são instâncias destas classes.

1.1 INTRODUÇÃO ÀS BASES DE DADOS

1.1.1 OBJECTIVOS

O aluno deverá conhecer e identificar os conceitos de modelização, criação e implementação de base de dados, nomeadamente:

- conhecer o conceito de base de dados;
- conhecer o conceito de Sistemas de Gestão de Base de Dados;
- construir o modelo conceptual de base de dados;
- implementar, com recurso ao *Microsoft Access*, uma base de dados com os objectos comuns de tabelas, consultas, formulários e relatórios.

1.1.2 BASE DE DADOS

De uma forma genérica, é possível afirmar que qualquer conjunto de dados é uma base de dados definida como um repositório de dados sobre determinado assunto, tema ou solução.

O principal objectivo de criação de base de dados é permitir guardar o histórico dos dados e produzir informação, após o processamento dos mesmos.

Uma base de dados relacional é uma tabela ou um conjunto de tabelas relacionadas entre si e, por sua vez, uma tabela é um conjunto de registos relacionados entre si, que se desdobram em campos, a unidade capaz de armazenar dados.

Uma base de dados será então, um conjunto de ficheiros que armazenam os dados e geridos por Sistemas de Gestão de base de dados, conhecidos por SGBD's e que permitem o seguinte:

- **Definição de dados** – criar e alterar a estrutura da Base de Dados.
- **Consulta de dados** – obter e consultar os dados armazenados.
- **Manipulação de dados** – adicionar e alterar dados existentes.

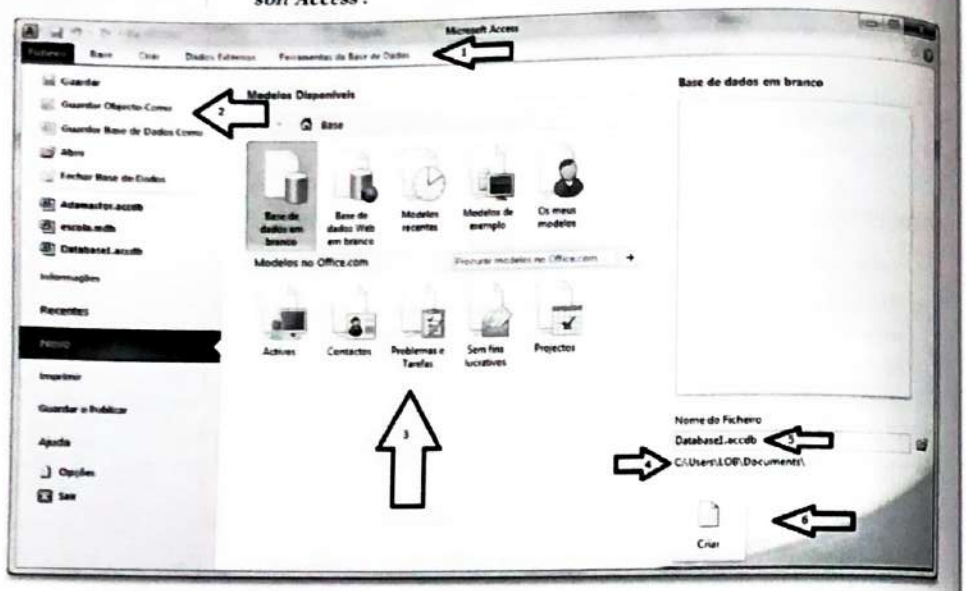
Além destas funcionalidades dos SGBD's, destacamos o seguinte:

- **Acesso simultâneo:** aceder e alterar a mesma Base de Dados em simultâneo.
- **Vistas:** acesso limitado a diferentes componentes registadas na Base de Dados.
- **Construção de aplicações:** gestão de dados com o desenvolvimento das aplicações informáticas.

1.4 UTILIZAÇÃO DE UMA FERRAMENTA DE BASE DE DADOS

Para a concepção deste manual foi utilizado o *Microsoft Access 2010*. Esta ferramenta, como já foi explicado no ponto anterior, é um Sistema de Gestão de Bases de Dados (SGBD) para computadores pessoais (PC). Adequa-se ao uso doméstico, em pequenas empresas ou como forma de aceder a Base de Dados instaladas em sistemas de grande porte.

Conheçamos um pouco melhor o nosso ambiente de trabalho do *Microsoft Access*:



Identificamos de seguida os objectos identificados:

1 Barra de menus

2 Opções do menu ficheiro

3 Modelos disponíveis

4 Caminho onde será guardado o ficheiro da base de dados

5 Nome do ficheiro da base de dados a ser criada

6 Botão de acção para criar a base de dados com o nome e no local definidos anteriormente.

1.4.1 TABELAS

Os dados registados, em Microsoft Access, necessitam de ser armazenados de forma permanente (que não desaparecem, cada vez que a base de dados feche).

Assim, para assegurar os dados criamos as tabelas. As tabelas são constituídas por campos relacionados, por exemplo, a ficha de um aluno tem o número e o nome; o campo número refere-se ao número de aluno e é do tipo numérico e o nome é o nome do aluno e é alfanumérico, isto é, poderá armazenar números e letras.

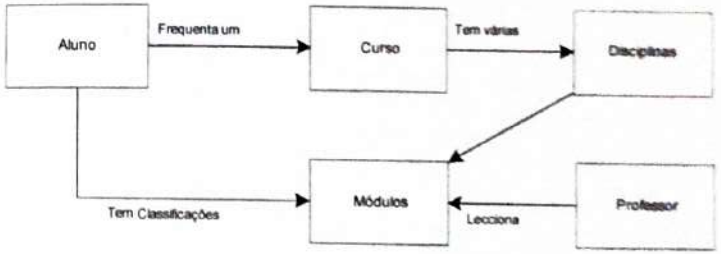
1.4.1.1 CRIAÇÃO

No âmbito do estudo da criação de base de dados e, consequentemente, de tabelas analisaremos um pequeno exemplo. Assim, pretendemos gerir a informação sobre a inscrição dos alunos no curso Técnico Profissional de Informática (usamos um curso mais geral) e sabemos o seguinte:

- Identificamos as entidades aluno, curso, disciplina e módulos da disciplina, professor.
- Sabemos que um aluno só frequenta um curso, mas um curso tem várias disciplinas e cada disciplina tem vários módulos.
- O aluno só termina o curso de obtiver aprovação em todos os módulos de todas as disciplinas do curso.
- O professor pode leccionar vários módulos da mesma disciplina e o mesmo módulo pode ser leccionado por mais do que um professor.

Adicionalmente, sabemos que no final pretendemos ter uma listagem do percurso escolar do aluno, bem com as classificações de cada disciplina. A qualquer momento deverá também ser possível obter um relatório com a listagem dos módulos em falta (ainda não obtiveram classificação).

Analisemos o seguinte diagrama:



Da análise efectuada ao diagrama, identificámos as entidades e as possíveis associações que fazem parte da solução do nosso problema.

chave. A terceira forma normal diz-nos, por outras palavras, que os atributos que não fazem parte da chave, dependem apenas de todos os campos da chave.

Existe uma dependência funcional entre dois atributos quando se consegue determinar um a partir de outro atributo. Por exemplo, sabendo-se o número do aluno, é fácil obter-se o seu nome. Diz-se, neste caso, que o nome do aluno depende funcionalmente do número do aluno, e representa-se do seguinte modo: Número→Nome.

Uma dependência transitiva consiste na seguinte sequência de dependências: o atributo B depende funcionalmente do atributo A (A→B) e o atributo C depende funcionalmente do atributo B (B→C), mas o atributo A não depende do atributo B.

Forma normal de Boyce-Codd

A forma normal de Boyce-Codd para uma relação implica que esta esteja na terceira forma normal e todos os campos que são determinantes fazem parte do conjunto de chaves candidatas.

Um determinante é um campo do qual dependem funcionalmente outros atributos. Assim sendo, uma chave é sempre um determinante, embora o contrário nem sempre seja verdade.

Quarta forma normal

A quarta forma normal implica que uma relação esteja na forma normal de Boyce-Codd e não possua dependências multi valor. Uma dependência multi valor existe quando para um só atributo existe mais do que uma dependência funcional para outros atributos independentes.

Quinta forma normal

A quinta forma normal de uma relação obriga a que esta esteja na quarta forma normal e que existam determinadas condições para a junção de tabelas.

Na prática, ninguém se preocupa em levar as relações de uma base de dados relacional até esta forma normal.

1.3.4 CONCEITO DE CHAVE PRIMÁRIA E ESTRANGEIRA

Chaves

Cada tupla de uma tabela tem que estar associada a uma chave única que permita identifica-la.

Uma chave pode estar composta por um ou mais atributos. Uma chave tem que ser única dentro de sua tabela e não se pode descartar nenhum atributo da mesma para identificar uma fila. Existem dois tipos de chaves:

1. Chave primária (Primary Key): é o valor ou conjunto de valores que identificam uma linha de uma tabela. Nunca pode ser NULL. Um exemplo claro de chave primária seria o Número de Bilhete de Identidade, que é único para cada pessoa e não pode ser NULL.
2. Chave estrangeira (Foreign Key): é o valor ou valores de uma tabela que corresponde com o valor de uma chave primária em outra tabela. Esta chave é a que representa as relações entre as tabelas.

REGRAS PARA A CHAVE PRIMÁRIA

1. Valor único: Não podem ser nulos (conter valor nulo).
2. Não redundante: no caso de uma chave primária ser composta, não devem ser incluídos mais atributos do que os mínimos necessários para identificar um registo de uma forma unívoca.

O TIPO DE CAMPO deve ser escolhido de acordo com as características dos dados que queremos que ele guarde.

A escolha acertada do tipo de campo facilita a sua utilização superior:

1. Se quisermos comparar datas, convém que o campo esteja definido como data/hora,
2. Se quisermos fazer cálculos, temos que o definir como numérico. Existir 2 registos com o mesmo conteúdo para a chave primária.
3. Não Nulo: Não podem existir nenhum atributo chave primária que contenha campo nulo

mária, é o conjunto mínimo de atributos que permite determinar univocamente uma instância numa entidade.

As Associações são utilizadas para relacionar entidades. As entidades interagem umas com as outras, através de associações.

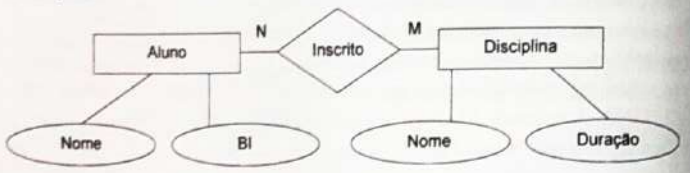
Existem dois aspectos fundamentais nas associações: Cardinalidade e Obrigatoriedade. A cardinalidade, também designada por tipo de associação, numa instância de uma entidade, indica o número de instâncias da outra entidade com que ela se relaciona. De uma forma geral, existem três tipos de associações: 1:1 (um-para-um), 1:N (um-para-muitos) e N:M (muitos-para-muitos).

- Uma associação 1:1 é quando cada elemento da entidade A está relacionado, no máximo, com um elemento da entidade B.
- Uma associação 1:N é definida se cada elemento da entidade A está relacionado com vários elementos da entidade B, mas cada elemento da entidade B apenas está relacionado com um elemento da entidade A.
- Uma associação M:N é definida se cada elemento da entidade A está relacionado com vários elementos da entidade B, e cada elemento da entidade B está relacionado com vários elementos da entidade A.

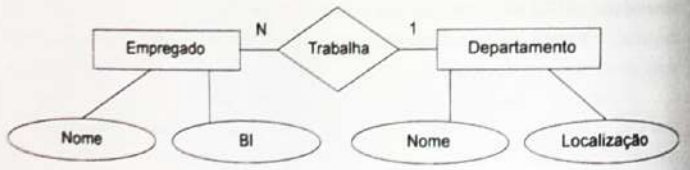
A obrigatoriedade da associação prende-se com o facto de pretendermos especificar se é obrigatório ou não que todas as instâncias estejam relacionadas a pelo menos uma instância da outra entidade.

Para representar as relações, utilizamos os diagramas de representação Entidade Associação. A Simbologia utilizada:

Exemplos:



Nota: A associação "inscrito" é chamada uma entidade associativa. E é uma forma de simplificar o desenho.



1.3.3 NORMALIZAÇÃO DE TABELAS

Os dados a armazenar numa base de dados não possuem, em geral, uma estrutura adequada ao seu armazenamento, seja esta relacional ou não. As alterações a efectuar sobre a estrutura em bruto dos dados, para que eles possam ser armazenados numa estrutura relacional, designa-se por normalização de dados.

A normalização de um esquema consiste numa sucessão de alterações ao esquema inicial que, gradualmente, vai eliminando as suas anomalias. Portanto, quanto mais se avança na sua forma normal menos anomalias se encontram.

Mas as formas normais não são a solução para todos os problemas. Muitas vezes, admitem-se esquemas não completamente normalizados, isto é, que cumprem a quinta forma normal. Esta falta de normalização é compensada por outras vantagens, principalmente em termos de desempenho.

As formas normais dividem-se em seis grupos que estudaremos, com algum detalhe, em seguida.

Primeira forma normal

A primeira forma normal requer que as tabelas tenham apenas duas dimensões, ou seja, cada um dos campos deve conter valores atómicos, e não listas de valores. Caso haja um atributo que não esteja de acordo com esta forma normal, a tabela de que ele faz parte deverá ser dividida em duas (ou mais), para que cada atributo seja atómico.

Por exemplo: as notas que um aluno tirou a cada disciplina ao longo do seu curso não poderão, numa tabela que esteja na primeira forma normal, fazer parte da mesma tabela que contém o número, nome e morada do aluno. As notas passarão a fazer parte de uma outra tabela que, para cada aluno e disciplina, contenha a nota do aluno (e possivelmente outros dados, como a data em que o aluno obteve a nota, se foi através de frequência, exame ou trabalho prático, etc.).

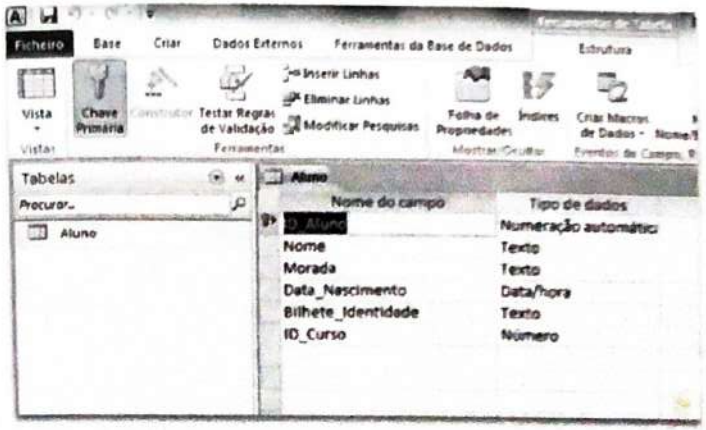
Segunda forma normal

As formas normais para além da primeira preocupam-se com dependências semânticas entre campos de uma mesma tabela. Uma relação que esteja na segunda forma normal possui a seguinte característica: todos os campos que não são chave dependem apenas de todos os campos que fazem parte da chave primária. De acordo com esta definição, todas as tabelas cuja chave primária possui apenas um campo estão na segunda forma normal.

Terceira forma normal

Uma relação está na terceira forma normal quando, está na segunda forma normal e não possui dependências transitivas entre os atributos que não fazem parte da

Vamos criar a tabela alunos de acordo com o seguinte exemplo e será esta a sua estrutura:



Como podemos observar, o campo "**ID_Aluno**" é um campo de numeração automática; não deixa de ser um número inteiro mas com a particularidade de incrementar automaticamente quando inserimos um novo registo. O campo "**ID_Aluno**" é também a "**Chave Primária**" da tabela. Existem situações em que não temos somente uma chave primária mas sim várias, bastando para as definir seleccionar os campos através do selector de campos e escolher a opção "**Chave Primária**" da estrutura de tabela.

Em seguida, criaremos todas as tabelas do nosso exercício.

Tabela Classificação

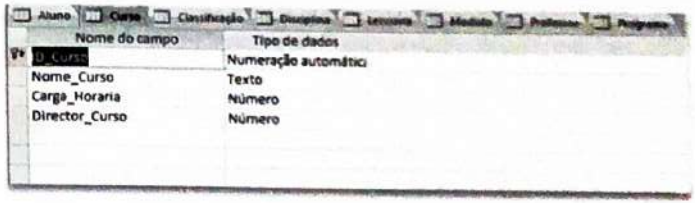
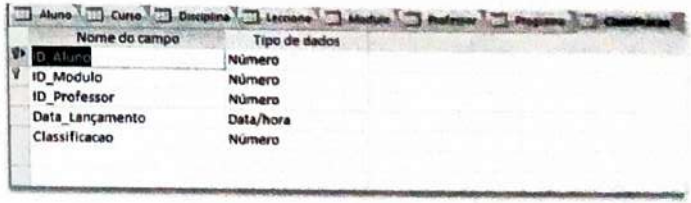


Tabela Curso



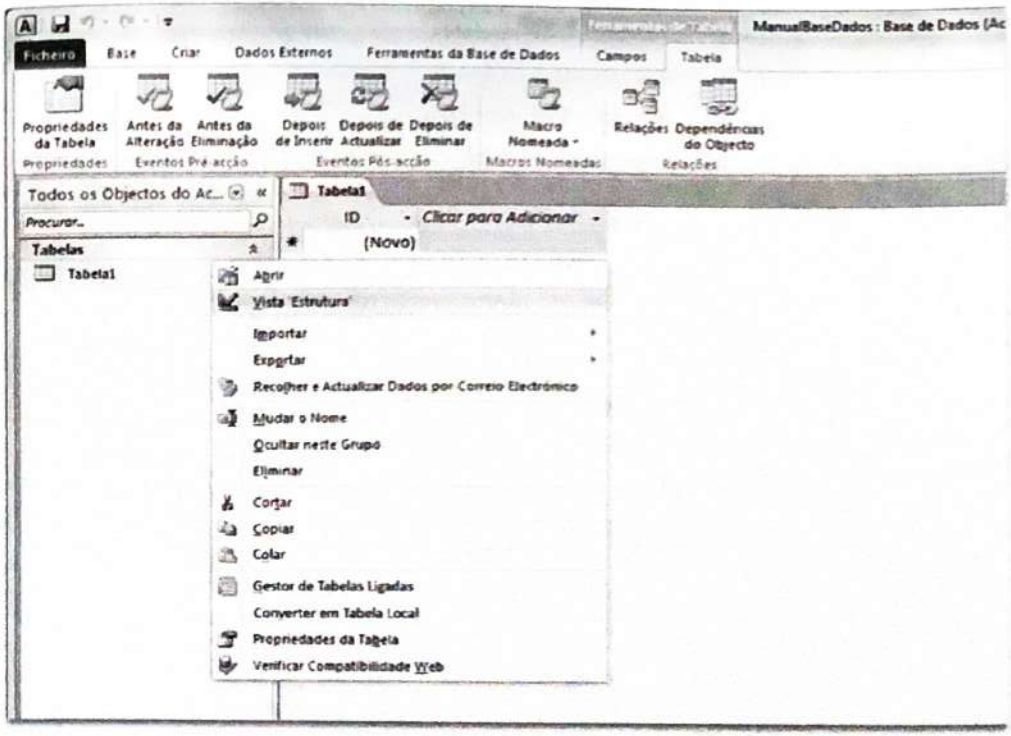
O *Microsoft Access* (*MSAccess*) permite criar as tabelas de várias formas, entre elas em "**Modo Estrutura**", com um acesso completo à definição dos tipos de dados, ou em modo folha de dados, em que é possível adicionar colunas (campos) à tabela. Neste último caso, para adicionar o campo bastará accionar o "**Clicar para adicionar**".

As opções disponíveis são:

- **Vista de folha de dados:** o utilizador introduz dados numa matriz semelhante a uma folha de cálculo e o *Access* procura inferir os tipos dos campos.
- **Vista de estrutura:** definição explícita da estrutura da tabela.
- **Assistente de Tabelas:** o assistente disponibiliza um conjunto de campos predefinidos e dividido em categorias com os quais o utilizador pode constituir a sua tabela.
- **Importação de Tabelas:** construção de tabelas a partir de ficheiros já existentes, por exemplo, documentos Excel ou ficheiros de texto.
- **Ligação de Tabelas:** possibilita que o *Access* tenha acesso a uma tabela de um aplicação/SGBD diferente sem ser preciso duplicar a tabela nem que os utilizadores da aplicação/SGBD original percam o acesso aos mesmos dados.

Embora tanto a vista de folha de dados como o assistente possam parecer mais apelativos à primeira vista, a utilização dessas opções trazem algumas limitações e dificuldades para além de serem específicas do *MSAccess* . Assim, será dado ênfase à vista de estrutura.

Em seguida, vamos criar as tabelas em modo estrutura. Para isso, copie as instruções ilustradas na imagem:



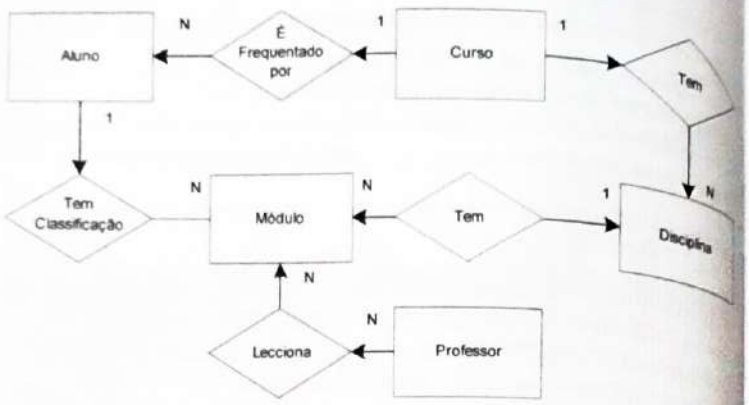
Primeiro clicamos na tabela com o botão direito do rato e depois escolhemos "**Vista Estrutura**".

Ao mudarmos de "**Vista de Folha de Dados**" para "**Vista Estrutura**", o *MSAccess* pede-nos a confirmação do nome da tabela que será "**Aluno**", onde vamos identificar os dados da tabela aluno.

Por defeito, o primeiro campo apresentado é o ID que é chave primária da tabela e está com o tipo "numeração automática", o que representa um inteiro incrementado de uma unidade por cada vez que insere um registo de forma automática, sem intervenção do utilizador. Para entendermos melhor as potencialidades e todos os tipos de campos disponíveis, vamos analisar os seguintes tipos:

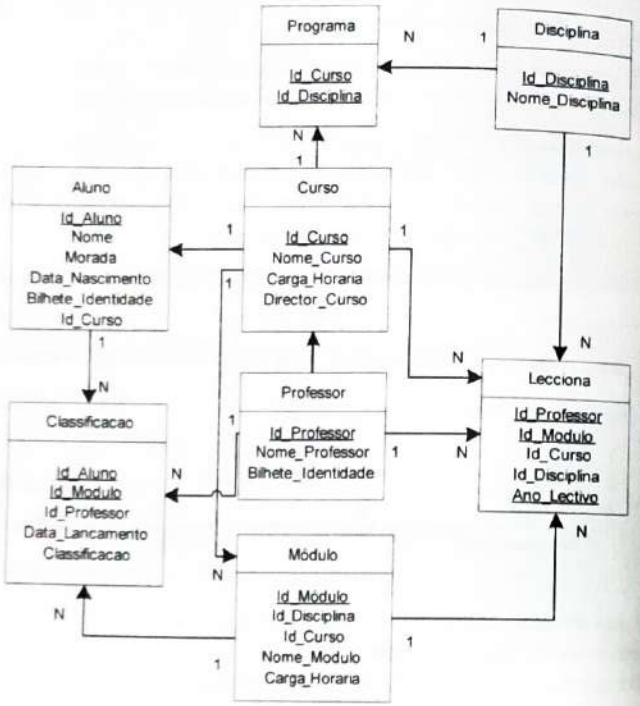
O tipo de dados determina quer o tipo de informação que o campo pode armazenar, quer o espaço (em *bytes*) que a informação ocupa. **Os tipos disponíveis no *Access* são:**

Recordando os conceitos expostos no ponto três deste manual poderíamos desenvolver o diagrama **Entidade-Associação** de forma completa.



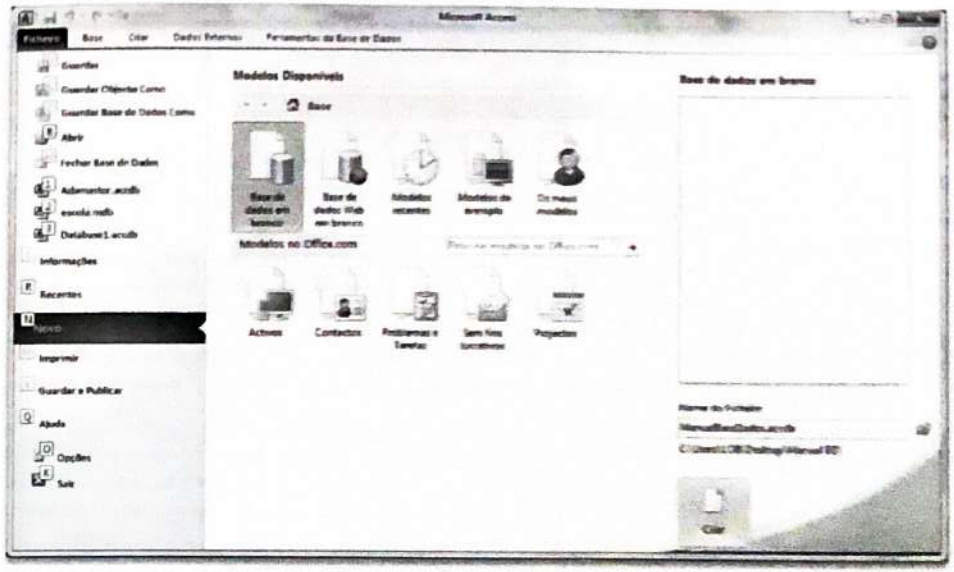
No decorrer da análise, teríamos que desenvolver as formas normais relativas à normalização dos dados necessários.

Obtendo a terceira forma normal para o problema proposto obtemos:

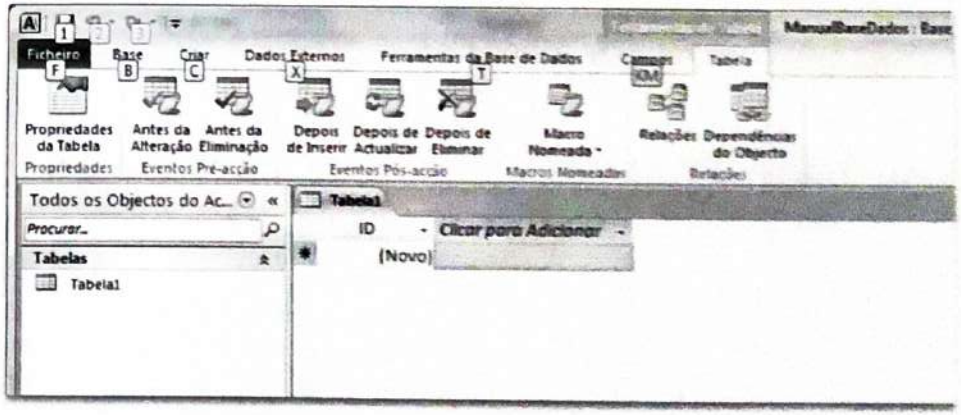


Esta é uma proposta de resolução do problema apresentado. A experiência do analista em conjunto com os *inputs* e *outputs* pretendidos, condiciona a normalização dos dados.

Iniciamos, agora, o exercício da base de dados do manual. Neste sentido, criar-se-á uma base de dados vazia com o nome **ManualBaseDados**.



Após a criação da base de dados **ManualBaseDados.accdb** temos acesso ao menu da aplicação do *Microsoft Access* com todas as ferramentas disponíveis. **Logo de seguida é-nos apresentada a seguinte janela:**



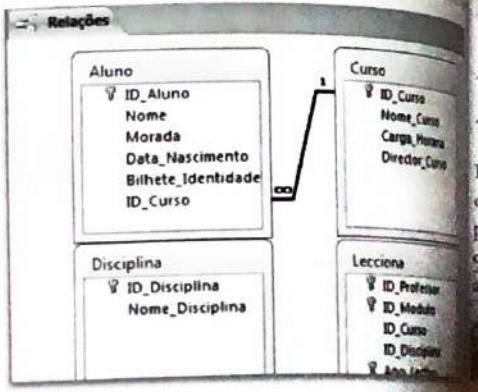
O *Microsoft Access* identificou o tipo de relação: um para muitos. Curso será "um" e a tabela Aluno será "muitos". Temos disponível a opção de "impor integridade referencial". Ao activar esta opção, estamos a garantir aspectos importantes entre os quais destacamos os seguintes:

- Se estiver activa esta opção, sempre que inserirmos um aluno teremos sempre de preencher o curso que frequenta, de acordo com a integridade referencial.
- Por outro lado, se activarmos a integridade, temos disponíveis duas opções:
  - **Propagar actualização dos campos relacionados**

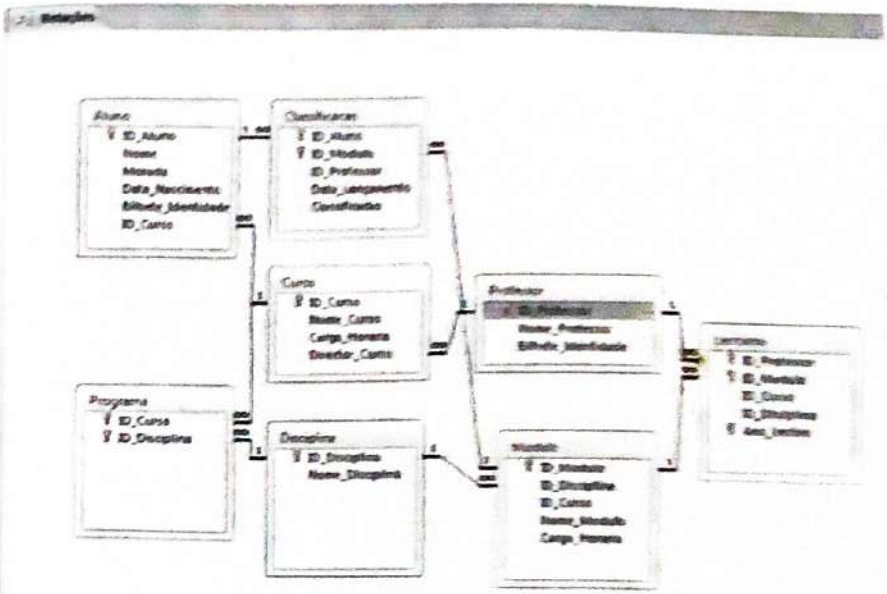
    Neste caso, se por acaso efectuarmos uma alteração na chave primária do curso, em todas as tabelas com que se relaciona, todos os campos serão actualizados automaticamente com o novo código. Esta situação é vantajosa porque não precisamos efectuar a actualização em todas as tabelas manualmente.
  - **Propagar eliminação dos registos relacionados**

    Neste caso, se por acaso efectuarmos uma eliminação com o código do curso, em todas as tabelas com que se relaciona, todos os registos serão eliminados automaticamente. Esta opção, embora muito útil, tem a sua dose de risco e deve ponderar muito bem se pretendemos esta eliminação automática.

Como queremos impor a integridade referencial, não só aqui como em todas, vamos activar a opção **"Impor integridade referencial"**.



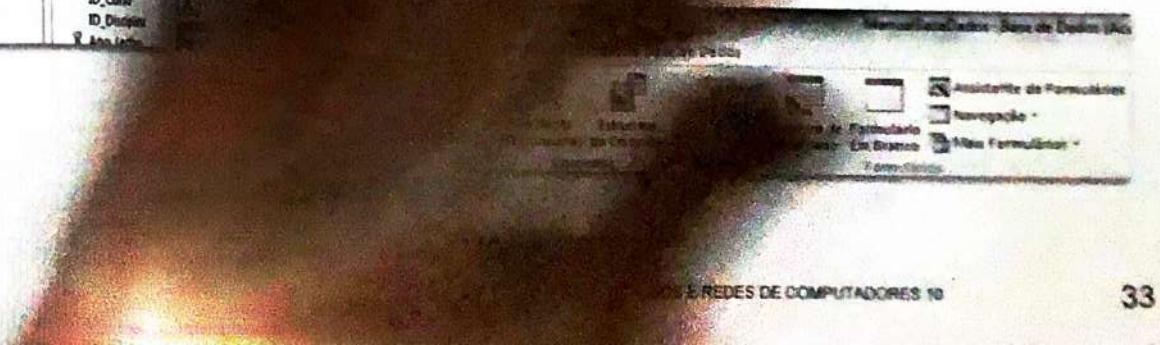
Em seguida, realizaremos as relações entre todas as tabelas. A disposição das tabelas é aleatória e não terá de obedecer a nenhuma regra, cada utilizador poderá definir com achar mais legível. Compare o seguinte resultado com o obtido:



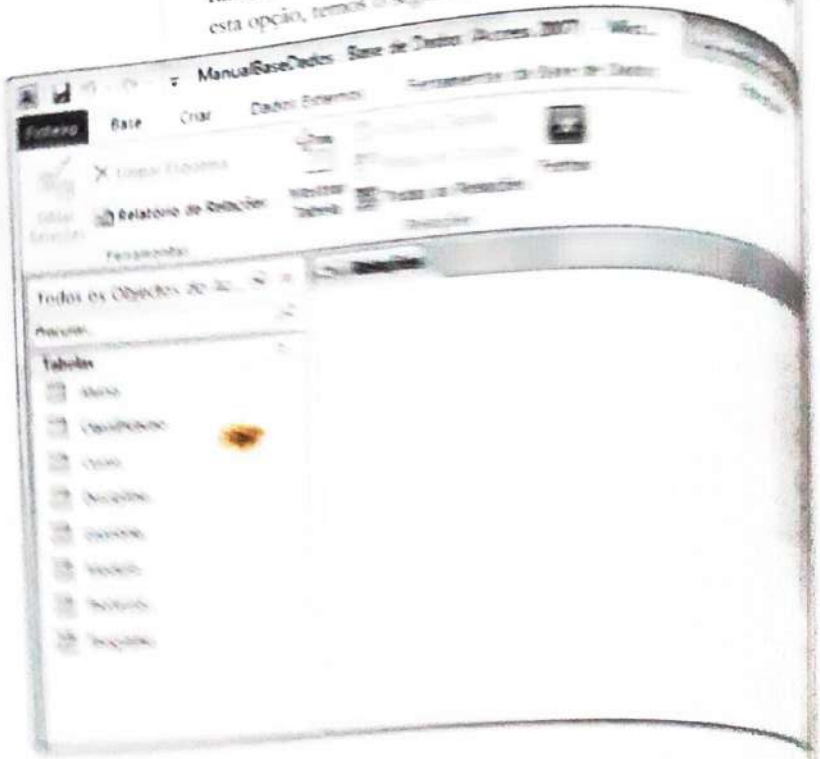
1.4.2 FORMULÁRIOS

1.4.2.1 MODOS DE CRIAÇÃO E TIPOS DE FORMULÁRIO

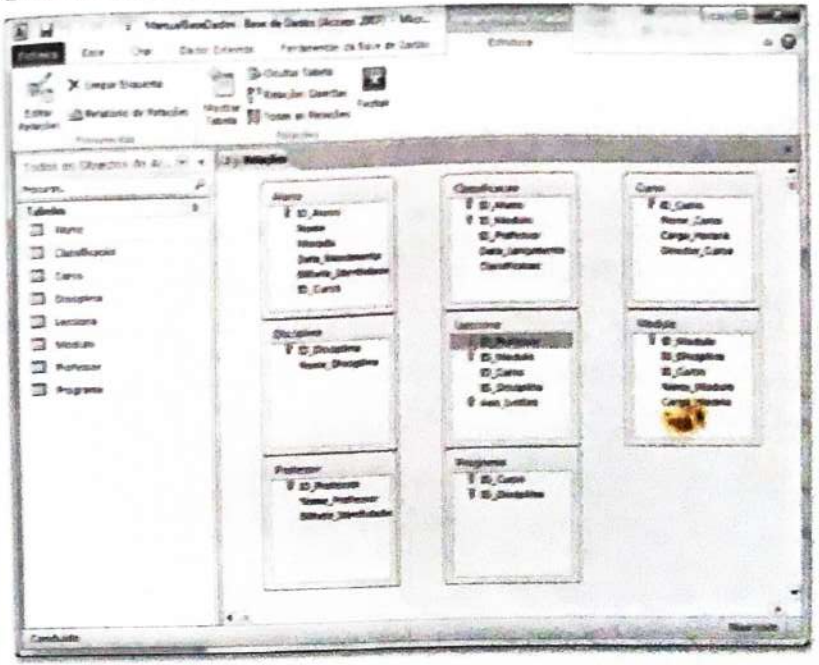
Depois de criada a base de dados, com as tabelas e com as relações, ao contrário de outros SGBD, o *Microsoft Access* permite criar outros objectos como é o exemplo de formulários. Os formulários [illegible] sobre as tabelas e/ou consultas que permitem gerir [illegible] inserir novos registos, alterar existentes [illegible] ao utilizador.



Primeiro exemplo Com a base de dados aberta, clique no separador **Ferramentas da Base de Dados** e seguidamente no botão **Relações**. Ao clicar nesta opção, temos o seguinte ambiente de trabalho:



O resultado obtido será:



Faremos então as relações, respeitando os requisitos que foram identificados no levantamento inicial. Então, sabemos que um aluno frequenta um curso, logo teremos analisar o tipo de relação. Neste caso chegamos à conclusão que um curso é frequentado por vários alunos e um aluno só pode frequentar um curso, logo teremos de ligar a chave primária da tabela curso para o campo "ID_Curso" da tabela Aluno.

Vamos efectuar a relação: arraste com o botão do rato o campo **"ID_Curso"** da tabela **"Curso"** para o campo **"ID_Curso"** da tabela **"Aluno"** e deverá aparecer a seguinte janela:

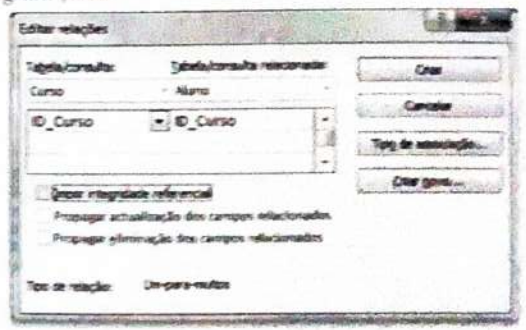
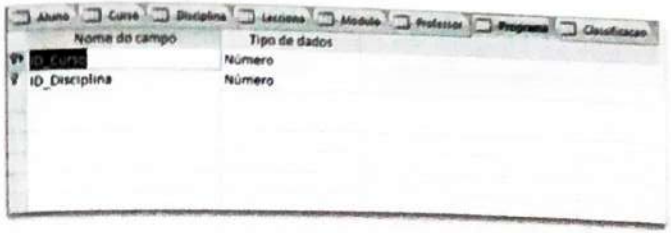


Tabela Programa



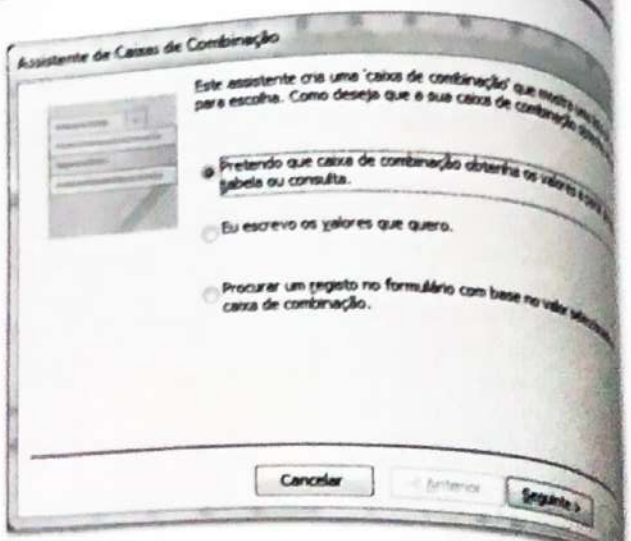
No final desta etapa, obtemos todas as tabelas que fazem parte do exercício. Tal como foi explicado inicialmente, esta resolução é apenas uma proposta, pelo que poderiam ter sido consideradas mais ou menos tabelas. No entanto, face aos requisitos definidos, estas tabelas são suficientes para os demonstrar.

1.4.1.2 RELACIONAMENTO ENTRE TABELAS

Após a definição das tabelas, precisamos de explicitar a forma como estes se relacionam. Na essência das bases de dados relacionais, podemos definir de uma forma muito leviana, como sendo um conjunto de tabelas relacionadas entre si. Assim, uma tabela é com conjunto de registos relacionados e um registo, é um conjunto de campos relacionados entre si.

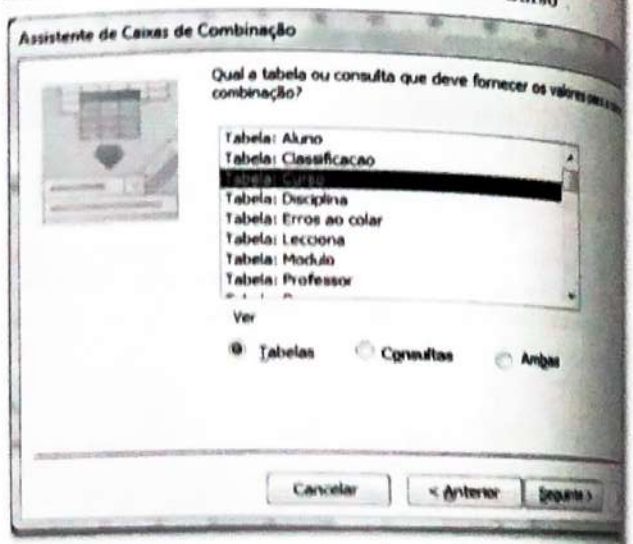
Como foi já exposto, as tabelas relacionam-se entre si em três formatos:

- **Relacionamento 1 para 1**
  - Quando é feito o relacionamento uma chave primária da tabela 1 com a chave primária da tabela 2, representando a mesma entidade mas potenciando as consultas de alguns campos mais necessários.
- **Relacionamento 1 para N**
  - Quanto um registo da tabela 1 poderá existir uma ou mais vezes na tabela 2.
- **Relacionamento N para N**
  - Quando o registo da tabela 1 poderá existir uma ou mais vezes na tabela 2 e quando um registo da tabela 2 poderá existir uma ou mais vezes na tabela 1

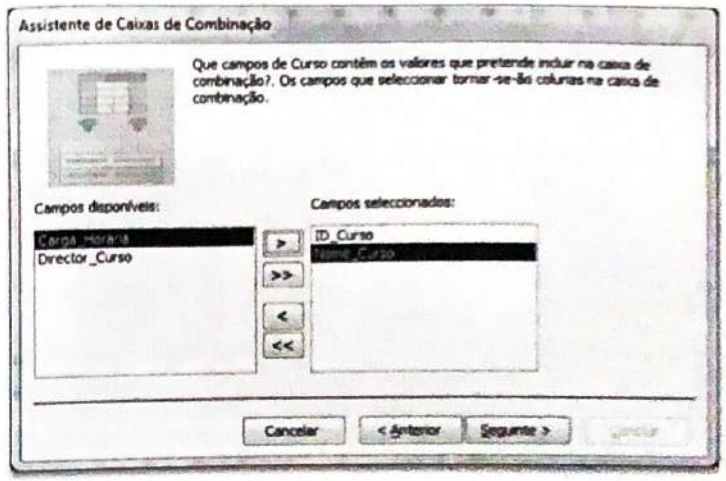


Neste exemplo específico, o nosso objectivo é seleccionar um curso
tes na tabela curso, e associar ao registo do aluno. Para isso, a
ção a criar terá de obter os dados de uma tabela (tabela curso),
o valor escolhido no campo da tabela aluno, de forma a garantir
referencial entre as duas tabelas.

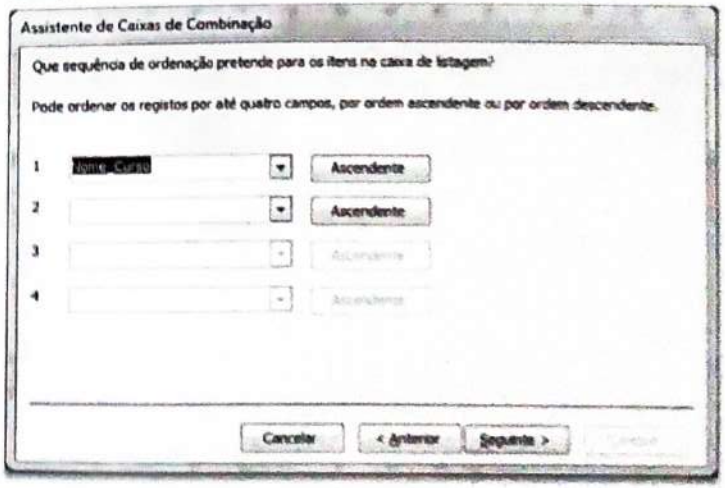
Escolhemos de seguida a tabela a utilizar: a tabela "**Curso**".



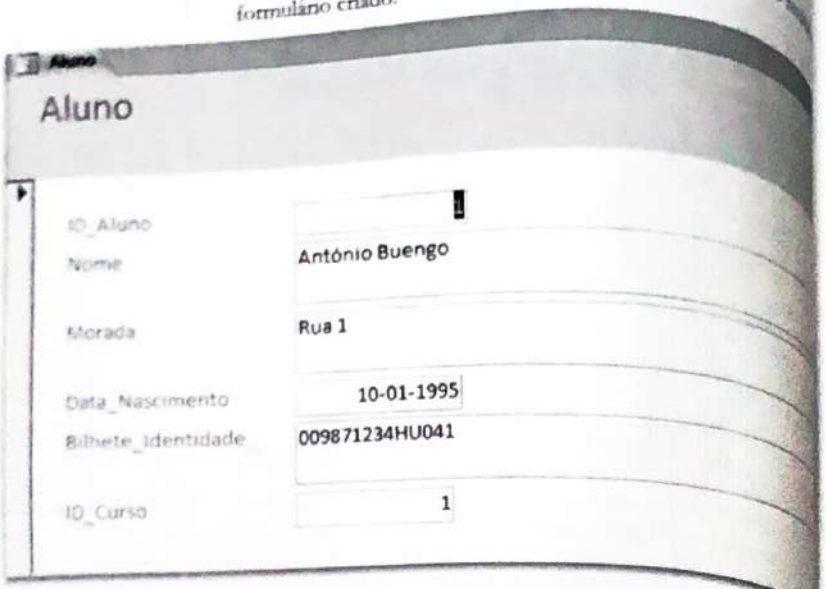
Sabe-se que na tabela curso temos vários campos e especificamente para este exemplo só interessa referenciar o curso que o aluno frequenta. Então, teremos de seleccionar o campo "**IDCurso**", para permitir relacionar as tabelas "**Aluno**" e "**Curso**", e o campo "**Curso**" para termos acesso ao descritivo do nome e não do número identificativo na tabela curso.



De seguida, ordenamos por descritivo de curso (esta operação é opcional).



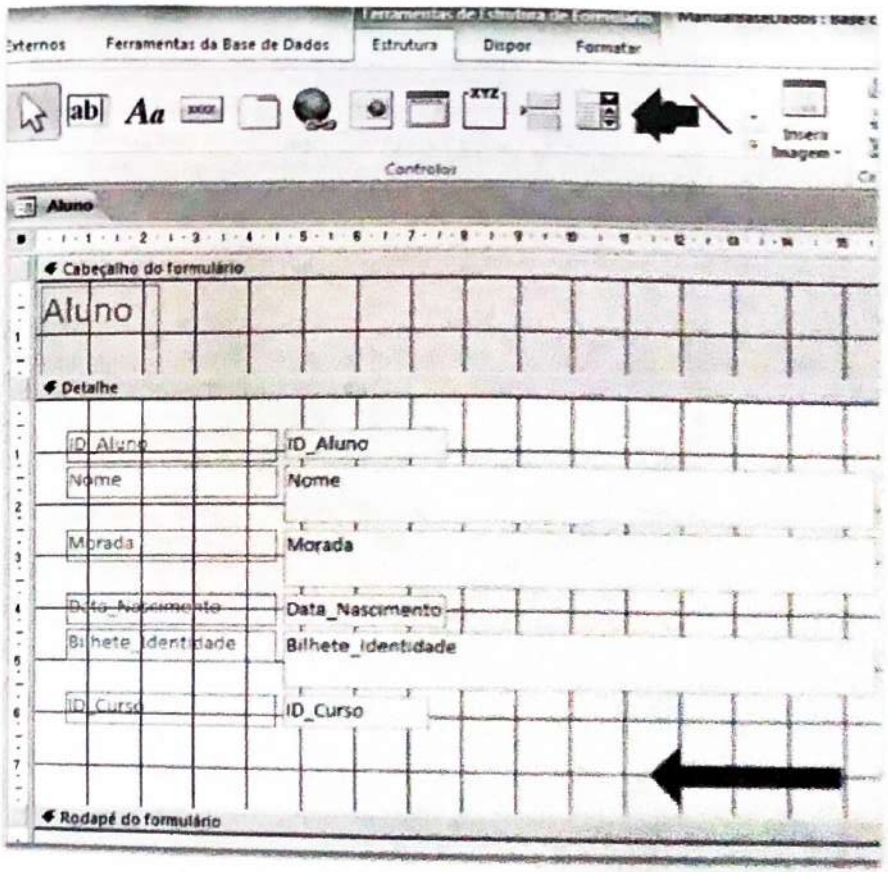
Em seguida, chegamos à janela final do assistente, sendo o
formulário criado.



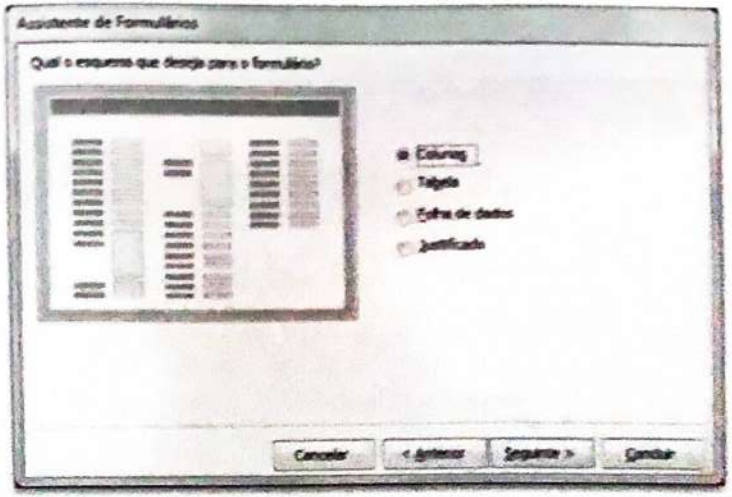
1.4.2.3 LIGAÇÃO DE OBJECTOS EXTERNOS A FORMULÁ

No entanto, podemos observar que, embora esteja com um aspe
para servir de interface, deparámo-nos com um problema: se
um novo aluno, como poderei indicar o curso que ele freq
temos acesso ao nome do curso mas somente ao código, o que
facilita a inserção.

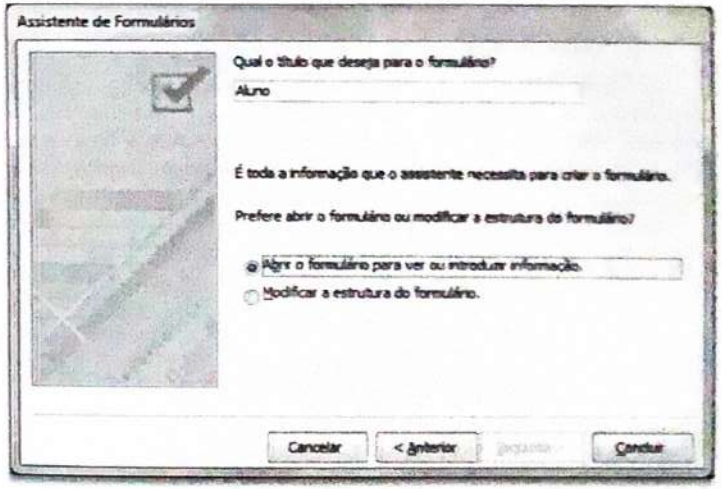
Vamos então criar uma caixa de combinação, ou combobox, para
curso, seleccionar os campos ID_Curso e Nome_Curso, e depois
formulário, este actualize correctamente com código do curso
Teremos então de mudar o modo de estrutura e adicionar novos

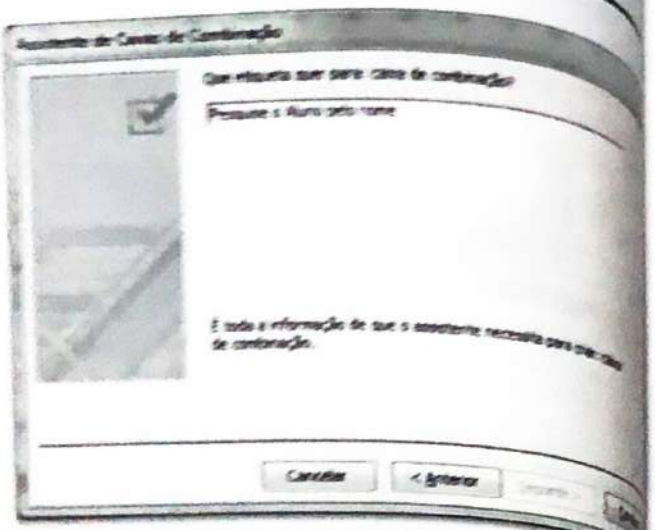


Depois de mudar de modo, vamos acrescentar uma caixa de combinação, para isso bastará seleccionar nos controlos o objecto "caixa de combinação" e arrastar para o final do formulário. Logo surgirá um assistente que o guiará na criação do objecto.

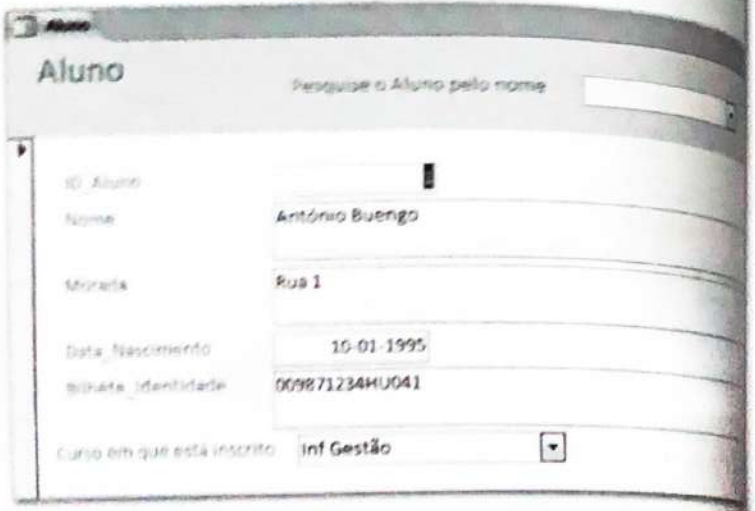


Nesta janela teremos de optar pelo tipo de formulário. Em colunas será o ideal quando temos mais do que cinco campos, uma vez que precisamos de uma zona disponível para organizar melhor a inserção de dados na base de dados. Em tabela será o formato ideal quando temos menos de cinco campos e são tabelas de dados normalmente tipificadas, sem necessidade de alteração contínua. O formato folha de dados é muito semelhante a uma folha típica de *Microsoft Excel*, será uma boa escolha quando temos vários campos pequenos a preencher rapidamente. O formato justificado é gerado automaticamente pelo *Microsoft Access*. Para este exemplo vamos escolher o formato coluna.





E vemos o resultado final. Desta forma, será mais fácil localizar um



1.4.3 CONSULTAS

O objecto consulta é um componente fornecido, e realmente necessário, nos sistemas de Gestão de Base de Dados. Com o objecto consulta conseguimos fazer acesso aos dados guardados nas tabelas e este também permitirá filtrar a informação pretendida, de forma a consultar somente o necessário.

Estas consultas servem de base a qualquer SGBD, uma vez que só através das consultas é que acedemos aos dados (em *MSAcess*, quando criamos um formulário pelo assistente é gerada uma consulta de forma automática, dispensando a intervenção do utilizador; no entanto só conseguiremos aceder aos dados no formulários com base na consulta).

Dada a importância deste objecto quando falámos em Gestão de Base de Dados, iremos analisar os vários tipos de consultas e também, os vários formatos de criação das mesmas.

1.4.3.1 CRIAÇÃO DE UMA CONSULTA POR SELECÇÃO

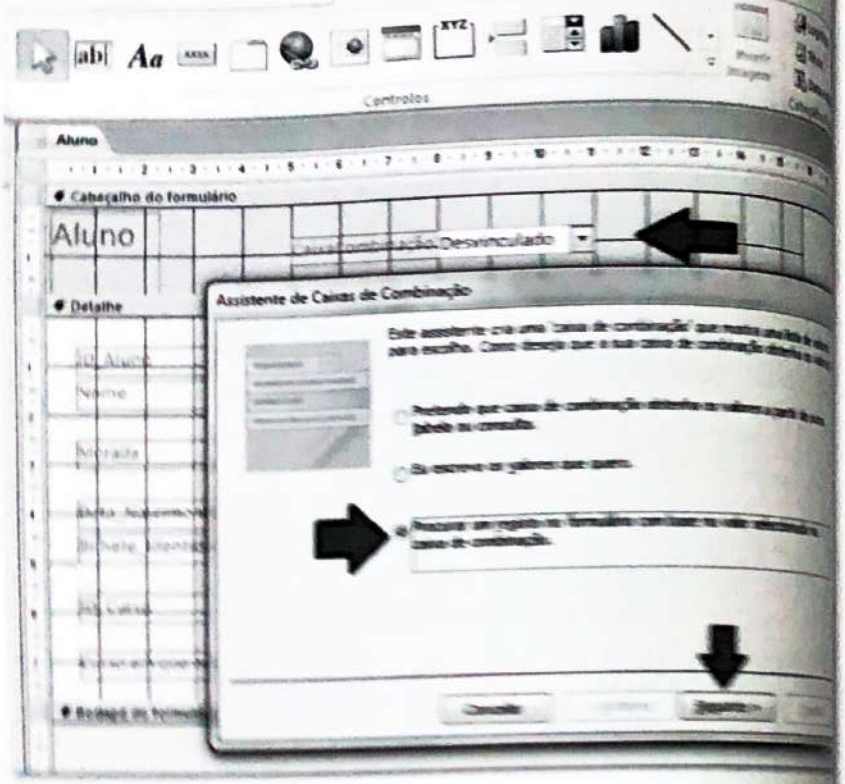
A consulta por selecção permitirá ao utilizador escolher as tabelas e os campos desejados, e pode, também, ser condicionada por filtros ou condições. Por exemplo, pretendo seleccionar todos os alunos inscritos no curso X, ou todos os alunos que sejam menores de idade, ou que tenham residência em determinada Localidade.

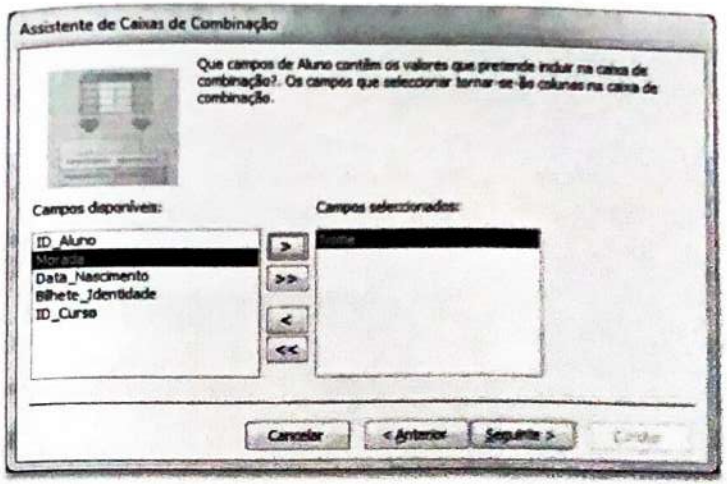
Analisemos a janela do *MSAcess* para a construção de Consultas. Podemos aceder de duas formas: através do menu de objectos, situado do lado esquerdo da janela, ou no menu criar, através das opções "utilizando assistente" ou vista de "estrutura".

Para testar esta funcionalidade, bastará abrir o formulário no modo **de Formulário**" e preencher dados exemplo. Recordo que neste ainda está disponível o campo "**ID_Curso**" no formulário. Depois funcionalidade da caixa de combinação, devem retirar o campo de substituir pela caixa criada.

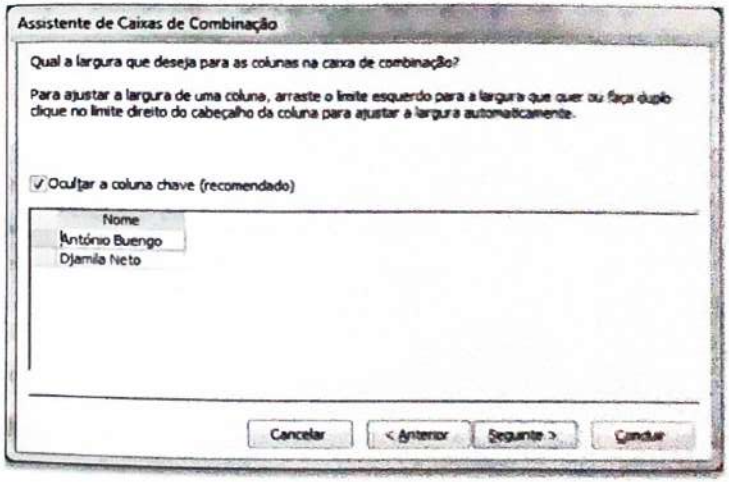
1.4.2.4 FILTRAGEM DE REGISTOS NUM FORMULÁRIO

Aproveitando o formulário gerado, o formulário "**Aluno**", vamos desenvolver mecanismos de consulta de registos num formulário. criar uma caixa de combinação para pesquisa de valores. Voltam **Estrutura**" do formulário "**Aluno**" e no cabeçalho vamos inserir "**Caixa de Combinação**" (como ilustrado na figura).





Como foi seleccionado o campo "**Nome**", quando escolhemos a opção "**Seguinte**", será apresentada uma listagem tipo por nome de aluno (estes dados só aparecem porque já foram introduzidos dois registos de alunos no formulário).



Para finalizar, atribuímos o nome à caixa de combinação: pesquise o aluno pelo nome.