

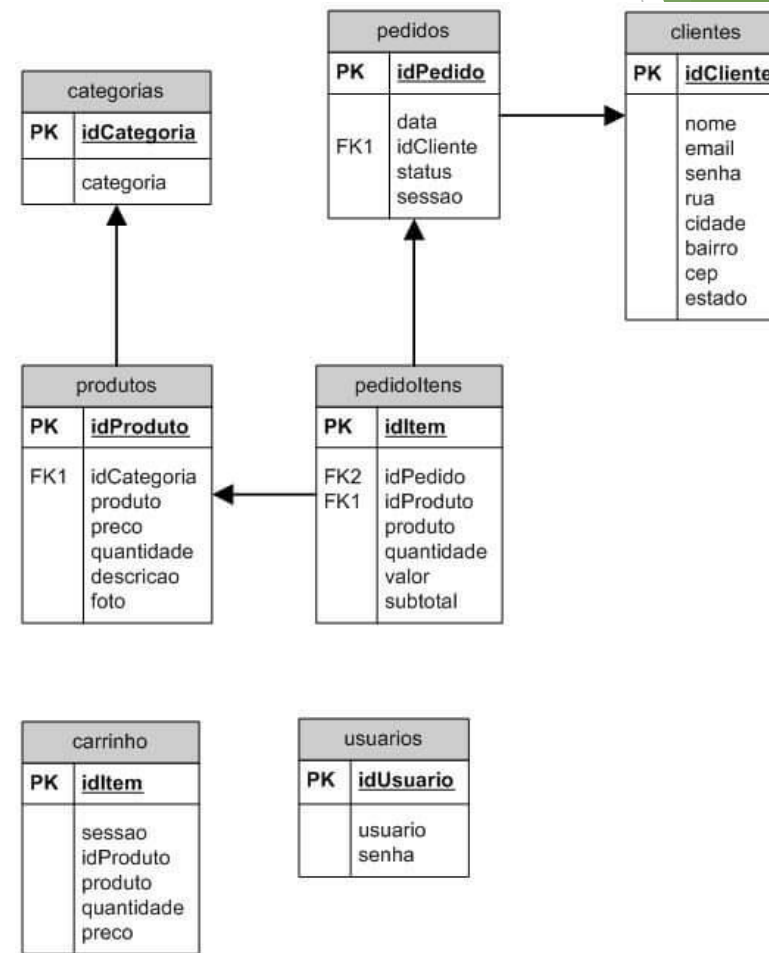
The background features abstract, overlapping green geometric shapes, primarily triangles and polygons, in various shades of green, creating a modern and dynamic visual effect.

Modelagem de Dados

1. Modelo lógico

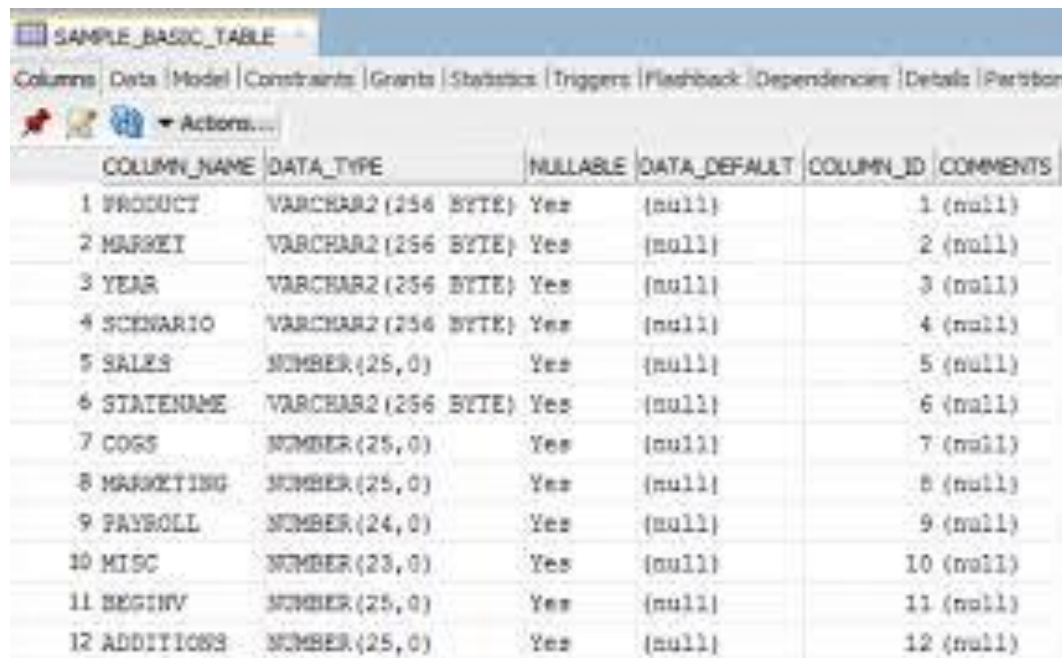
O modelo lógico já leva em conta algumas limitações e implementa recursos como adequação de padrão e nomenclatura, define as chaves primárias e estrangeiras, normalização, integridade referencial, entre outras.

Para o modelo lógico deve ser criado levando em conta os exemplos de modelagem de dados criados no modelo conceitual. Exemplo do diagrama de banco de dados gerado no Microsoft Visio.



2. Modelo físico

No modelo físico fazemos a modelagem física do modelo de banco de dados. Neste caso leva-se em conta as limitações impostas pelo SGBD escolhido e deve ser criado sempre com base nos exemplos de modelagem de dados produzidos no item anterior, modelo lógico.



The image shows a screenshot of a database management tool interface. At the top, there's a tab labeled 'SAMPLE_BASIC_TABLE'. Below it, a row of tabs includes 'Columns', 'Data', 'Model', 'Constraints', 'Grants', 'Statistics', 'Triggers', 'Flashback', 'Dependencies', 'Details', and 'Partitions'. The 'Columns' tab is active. Below the tabs, there's a section with icons and a dropdown menu labeled 'Actions...'. The main area displays a table with the following columns: 'COLUMN_NAME', 'DATA_TYPE', 'NULLABLE', 'DATA_DEFAULT', 'COLUMN_ID', and 'COMMENTS'. The table contains 12 rows of data, numbered 1 through 12 in the 'COLUMN_ID' column.

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	PRODUCT	VARCHAR2(256 BYTE)	Yes	{null}	1	{null}
2	MARKEI	VARCHAR2(256 BYTE)	Yes	{null}	2	{null}
3	YEAR	VARCHAR2(256 BYTE)	Yes	{null}	3	{null}
4	SCENARIO	VARCHAR2(256 BYTE)	Yes	{null}	4	{null}
5	SALES	NUMBER(25,0)	Yes	{null}	5	{null}
6	STATENAME	VARCHAR2(256 BYTE)	Yes	{null}	6	{null}
7	COGS	NUMBER(25,0)	Yes	{null}	7	{null}
8	MARKETING	NUMBER(25,0)	Yes	{null}	8	{null}
9	PAYROLL	NUMBER(24,0)	Yes	{null}	9	{null}
10	MISC	NUMBER(23,0)	Yes	{null}	10	{null}
11	BEGINV	NUMBER(25,0)	Yes	{null}	11	{null}
12	ADDITIONS	NUMBER(25,0)	Yes	{null}	12	{null}

3. O que é SQL?

SQL significa “*Structured Query Language*”, ou “Linguagem de Consulta Estruturada”, em português.

Resumidamente, é uma linguagem de programação para lidar com banco de dados relacional (baseado em tabelas).

Foi criado para que vários desenvolvedores pudessem acessar e modificar dados de uma empresa simultaneamente, de maneira descomplicada e unificada.

E se quiser conferir cursos sobre o tema, confira os links abaixo:

KORTH, H.F. e SILBERSCHATZ, A.; Sistemas de Bancos de Dados, Makron Books, 2a. edição revisada, 1994.

DATE, C.J.; Int. a Sistemas de Bancos de Dados, tradução da 4a.edição norte-americana, Editora Campus, 1991.

3.1 Para que serve?

A programação SQL pode ser usada para analisar ou executar tarefas em tabelas, principalmente através dos seguintes comandos: inserir (*'insert'*), pesquisar (*'search'*), atualizar (*'update'*) e excluir (*'delete'*). Porém, isso não significa que o SQL não possa fazer coisas mais avançadas, como escrever *queries* (comandos de consulta) com múltiplas informações.

3.2 Para que serve?

A programação SQL pode ser usada para analisar ou executar tarefas em tabelas, principalmente através dos seguintes comandos: inserir (*'insert'*), pesquisar (*'search'*), atualizar (*'update'*) e excluir (*'delete'*). Porém, isso não significa que o SQL não possa fazer coisas mais avançadas, como escrever *queries* (comandos de consulta) com múltiplas informações.

4. 0 - Gerenciamento de tabelas (criação, alteração e exclusão)

Gerenciamento de tabelas (criação, alteração e exclusão)

Sumário

- Gerenciamento de tabelas
- Criando tabelas com create table
- Alterando tabelas com alter table
- Excluindo tabelas com drop table

Gerenciamento de tabelas (criação, alteração e exclusão)

Gerenciamento de tabelas (DDL SQL)

Os comandos que permitem o gerenciamento das tabelas em SQL estão na divisão DDL SQL. Os principais comandos são **create, alter e drop**.

Por exemplo:

- **create table** para criar tabelas;
- **alter table** para alterar tabelas;
- **drop table** para excluir tabelas.



Fonte: elaborado pelo professor
Imagem 1: divisões da linguagem SQL e os principais comandos - DDL em destaque

Para Criar a 1ª Base de Dados :
Utilizamos Create Database;

Gerenciamento de tabelas (criação, alteração e exclusão)

Criando tabelas com create table

Sintaxe:

```
1 create table nome_da_tabela (  
2   coluna1 tipo,  
3   coluna2 tipo,  
4   coluna3 tipo,  
5   ...  
6   );|
```

Fonte: elaborado pelo professor
Imagem 2: sintaxe do comando create table

Gerenciamento de tabelas (criação, alteração e exclusão)

Criando tabelas com create table

Alguns tipos de campos (PostgreSQL):

- **int** para inteiros
- **decimal(total de dígitos, casas decimais)**
para números com casas decimais
- **serial** para campos do tipo autonumeração
- **date** para datas
- **time** para horas
- **timestamp** para data e hora
- **varchar(tamanho)** para caracteres
- **text** para caracteres

Gerenciamento de tabelas (criação, alteração e exclusão)

Criando tabelas com create table

departamentos (id, nomedepto)

funcionarios(id, nomefunc, departamento_id)

departamento_id referencia departamentos(id)

Lembrando o modelo logico, podemos criar as tabelas com as chaves primarias e chaves estrangeiras.

Gerenciamento de tabelas (criação, alteração e exclusão)

Criando tabelas com create table

departamentos (id, nomedepto)

funcionarios(id, nomefunc, departamento_id)

departamento_id referencia departamentos(id)

```
1 create table departamentos (  
2     id serial,  
3     nomedepto varchar(50),  
4     primary key(id)  
5 )
```

Fonte: elaborado pelo professor

Imagem 3: exemplo do comando create table para a tabela departamentos

As duas tabelas...criadas podem ser testadas nos ambiente sugeridos

Gerenciamento de tabelas (criação, alteração e exclusão)

Criando tabelas com create table

departamentos (id, nomedepto)

funcionarios(id, nomefunc, departamento_id)

departamento_id referencia departamentos(id)

```
1 create table funcionarios(  
2     id serial,  
3     nomefunc varchar(50),  
4     departamento_id int,  
5     primary key(id),  
6     foreign key(departamento_id) references departamentos(id)  
7 )  
8
```

Fonte:elaborado pelo professor

Imagem 4: exemplo do comando create table para a tabela funcionarios

Gerenciamento de tabelas (criação, alteração e exclusão)

Criando tabelas com create table

```
1 create table departamentos (  
2     id serial,  
3     nomedepto varchar(50),  
4     primary key(id)  
5 )
```

```
1 create table funcionarios(  
2     id serial,  
3     nomefunc varchar(50),  
4     departamento_id int,  
5     primary key(id),  
6     foreign key(departamento_id) references departamentos(id)  
7 )  
8
```