1. Criando uma tabela no banco de dados

O comando SQL que cria tabelas é:

CREATE TABLE <nome_da_tabela>
(
<definição de coluna>,
<definição de coluna>,
.....
<definição de coluna>,
<definição de restrição>,
<definição de restrição>,
.....
<definição de restrição>
);

onde <definição de coluna> é:

<nome da coluna> <tipo dos dados da coluna> [default <expr>] [<restrições da coluna>]

2. Inserindo linhas no banco de dados

O comando SQL que insere linhas em tabelas é:

INSERT INTO <nome_da_tabela>
[(<coluna>{, coluna })] **VALUES** (< expressão {, expressão });

Exemplo:

```
insert into zipcodes values (67226, 'Madureira');
```

3. Modificando linhas do banco de dados

O comando SQL que modifica linhas de tabelas é:

UPDATE <nome_da_tabela>
SET <nome_do_campo_a_ser_modificado> = <novo_valor_do_campo>
[, <nome_do_campo_a_ser_modificado> = <novo_valor_do_campo>]
.....,
[, <nome_do_campo_a_ser_modificado> = <novo_valor_do_campo>]
[**WHERE** condição]

Exemplo:

```
UPDATE      fornecedores
SET         endforn = 'Rua do Trevo 35 casa 3'
WHERE       nomeforn = 'Maria'
```

Modifica o endereço de Maria para Rua do Trevo 35 casa 3.

4. Removendo linhas do banco de dados

O comando SQL que remove linhas de tabelas é:

DELETE FROM <nome_da_tabela>
[**WHERE** condição]

Exemplo:

```
DELETE      FROM  fornecedores
WHERE       nomeforn = 'Maria'
```

Remove a linha da fornecedora cujo nome é 'Maria' da tabela fornecedores.

5. Consultas ao banco de dados

Realizadas através do comando select, que tem a seguinte sintaxe:

select [distinct] <expressão> { , <expressão> }
from < tabela > [<alias>] { , <tabela> [<alias>] }
[**where** <condição>];
[**group by** <coluna> { , <coluna>}]
[**having** < condição>]
[**order by** coluna [**asc** | **desc**] {, coluna [**asc** | **desc**] }]

Ordem conceitual da execução de um comando select:

- O produto de todas as tabelas da cláusula **from** é formado;
- A cláusula **where** é avaliada para eliminar as linhas que não satisfazem a <condição> da cláusula **where**;
- As linhas são agrupadas utilizando a cláusula **group by**;
- Grupos que não satisfazem à condição da cláusula **having** são eliminados;
- As expressões existentes na cláusula **select** são avaliadas;
- Se a palavra **distinct** está presente na cláusula **select**, linhas duplicadas são eliminadas;
- As linhas resultantes são ordenadas de acordo com as linhas especificadas na cláusula **order by**.

Exemplos:

```
select * from  peças;

select    ono, cname
from      orders, cursomers
where     customers.cno = orders.cno and shipped is null;

select    sid
from      scores
where     points between 50 and 70;

select    distinct y1.pno
from      orders x1, orders x2, odetails y1, odetails y2
where     y1.pno = y2.pno and y1.ono = x1.ono and
          y2.ono = x2.ono and x1.cno < x2.cno;
```

6. Utilização das funções agregadas

Estaremos utilizando as funções :

AVG	Retorna a média aritmética de um conjunto de números
COUNT	Retorna o número de linhas da consulta
MAX	Retorna o valor máximo de um conjunto de números
MIN	Retorna o valor mínimo de um conjunto de números
SUM	Retorna a soma de um conjunto de números

Exemplos:

Fornecedores

CodForn	NomeForn	EndForn
1	Alberto	Rua 3, 44
2	Maria	Rua 44, 3
3	Josefa	Av. Ramalho, 334
4	Paulo	Rua Irmã Serafina 303
5	Ana	Av. Fo. Glicério 33
6	Wilson	Rua 32, 45 apto 4
7	Bete	Av. Sete de Setembro, 88
8	Daniel	Rua 32 , 444, apto 2
9	Graziela	Av. Sul 245 apto. 42
10	Eduardo	Rua Jardim Botânico 779

Peças

CodPe	NomePe	Cor
1	Copo Grande	Azul
2	Copo Grande	Vermelho
3	Copo Pequeno	Vermelho
4	Copo Pequeno	Verde
5	Prato grande	Amarelo
6	Prato pequeno	Amarelo
7	Garfo de plástico	Vermelho
8	Garfo de plástico	Verde
9	Faca de plástico	Vermelho
10	Faca de plástico	Branco
11	Garfo de metal	Verde
12	Garfo de metal	Amarelo

Catalogo

CodForn	CodPe	Preço (R$)
1	1	110,5
5	1	99,3
7	1	98,9
8	1	104
1	2	80,7
2	2	70,6
10	2	75
3	3	55,5
9	3	80,9
10	3	77,4
5	4	60,9
9	4	45,3
10	4	67,4
6	7	30,4
10	7	25,2
1	8	23,3
2	8	38,5
3	9	60,8
5	9	56,7
9	9	54,8
2	10	60
2	11	200

```
select count(*) from peças;

select avg(preco), max(preco), min(preco)
from catalogo;

select avg(preço), sum(preço)
from catalogo
where codpe=2;
```

7. Ordenando os resultados

O SQL permite especificar a ordem ***de exibição das linhas do resultado de uma consulta***, através do comndo:

ORDER BY <expressão> [**ASC** | **DESC**]

onde expressão é o nome de uma coluna ou expressão usada para criar uma coluna calculada.

```
select    codfon, codpe, preço
from      catalogo
order by  codforn;
```

8. Agrupando dados

O SQL permite o agrupamento de linhas de uma tabela baseado em algum critério.

O comando que permite este agrupamento é:

GROUP BY lista de colunas

Exemplos:

```
select avg(preço), sum(preço)
from catalogo
group by codpe;

select codpe, count(*)
from catalogo
group by codpe;
```

A combinação do uso de WHERE e GROUP BY permite filtrar as linhas que serão agrupadas.

Exemplo:

```
select codpe, count(*)
from catalogo
where preço < 100
group by codpe;
```

A cláusula HAVING permite filtrar grupos de registros já criados pelo comando GROUP BY.

Exemplo:

```
select codpe , avg(preco)
from catalogo
group by codpe
having max(preco)<100;
```

A diferença entre o uso combinado de WHERE e GROUP BY e o uso combinado de GROUP BY e HAVING é que no primeiro caso a filtragem ocorre antes do agrupamento (linhas são filtradas antes do agrupamento), enquanto que no segundo caso a filtragem é realizada depois do agrupamento (grupos que não satisfazem a condição da cláusula HAVING são eliminados da resposta da consulta).