

SQL - Avançado

- Inserção de dados;
- Atualização de dados;
- Remoção de dados;
- Projeção;
- Seleção;
- Junções;
- Operadores: aritméticos, de comparação, de agregação e lógicos;
- Outros comandos relacionados.

SQL

- SQL (Structured Query Language);
- Usa uma combinação da álgebra relacional e construções de cálculo relacional;
- Foi desenvolvida pela IBM no início dos anos 70 e mais tarde se tornou um padrão ANSI;
- Se estabeleceu como a linguagem padrão para banco de dados relacional;
- Embora seja chamada de “linguagem de consulta” ela contém outras capacidades além de consultas a banco de dados;
- Inclui recursos para definição de estruturas, modificação e restrição de dados (DDL), e para consulta e manipulação de dados (DML).

Comandos de Manipulação

- Inserção:

```
insert into conta (nome_agencia, numero_conta,  
nome_cliente, saldo)  
values ('Central', '9732', 'Smith', '1200')
```
- Atualização:

```
update conta  
set saldo = saldo * 1,05  
where saldo > 1000
```
- Remoção:

```
Delete from conta  
where nome_cliente = 'Smith'
```

Comandos de Consulta

- A estrutura básica de uma expressão SQL para consulta consiste em três cláusulas: **Select**, **From** e **Where**.
 - **Select**: Corresponde à operação *projeção* da álgebra relacional. É usada para listar os atributos desejados no resultado de uma consulta;
 - **From**: Corresponde à operação *produto cartesiano* da álgebra relacional. Ela lista as relações a serem examinadas na avaliação da expressão;
 - **Where**: Corresponde ao predicado da *seleção* da álgebra relacional. Consiste em um predicado envolvendo atributos de relações que aparecem na cláusula From.

Projeção

- Utilizada para recuperar colunas específicas de dados;
- SELECT é o comando que executa essa operação;
- Exemplo:

```
select nome, cidade from cliente;
```

Seleção

- Utilizada para selecionar linhas específicas de dados;
- Esta operação é possível através da cláusula WHERE;
- Exemplo:
`select nome, cidade, idade`
`from cliente`
`where nome like '%Smith%'`

Junção (Join)

- Retorna os dados da consulta de uma ou mais tabelas em um único conjunto;
- Esta operação também é possível através da cláusula WHERE;
- Existem quatro tipos de Join (**Inner Join, Left Join, Right Join e Outer Join**);

Junção (Exemplos)

- Inner Join:

```
select cliente.nome, cliente.idade, conta.saldo
from conta, cliente
where conta.codigo_cliente = cliente.codigo
```

- Left / Right Join:

```
select cliente.nome, cliente.idade, conta.saldo
from conta, cliente
where conta.codigo_cliente *= cliente.codigo / Left
where conta.codigo_cliente =* cliente.codigo / Right
```

- Outer Join: (não é suportado pelos SGBDs)

```
select cliente.nome, cliente.idade, conta.saldo
from conta, cliente
where conta.codigo_cliente ** cliente.codigo
```


Operadores Aritméticos

- Executam operações matemáticas em duas expressões (ou campos) de tipo numérico;
- Operações suportadas:

+ : adição	* : multiplicação
- : subtração	/ : divisão
	% : módulo

- Exemplo:
update conta
set saldo = saldo * 1,05
where nome_cliente = 'Smith'

Operadores de Comparação

- Testa se duas expressões (ou campos) são iguais ou não;
- Operações suportadas:

= : igual	>= : maior igual
> : maior que	<= : menor igual
< : menor que	<> : diferente

```
select nome_cliente, saldo
from conta
where saldo >= 1000
```

Operadores de Agregação

- São utilizadas para prover vários tipos de informações sumarizadas;
- Operações Suportadas:
 - **AVG**: Retorna a média dos valores não nulos da coluna;
 - **COUNT**: Conta a ocorrência dos valores não nulos de uma coluna;
 - **COUNT DISTINCT**: Conta a ocorrência dos valores únicos e não nulos de uma coluna;
 - **COUNT(*)**: Conta todos os registros de uma tabela;
 - **MAX**: Retorna o maior não nulo de uma coluna;
 - **MIN**: Retorna o menor não nulo de uma coluna;
 - **SUM**: Soma todos os valores não nulos de uma coluna;

Cláusulas GROUP BY e HAVING

- São empregadas nas consultas que utilizam funções de agregação;
- GROUP BY é utilizado para retornar colunas com valores normais e valores agregados;
- Todas as colunas não agregadas devem ser incluídas no GROUP BY;
- HAVING adiciona condições de busca no resultado da cláusula GROUP BY;
- HAVING não afeta o cálculo da agregação, mas sim as linhas retornadas pela consulta.

Agregação (Exemplos)

```
select nome_agencia,  
       count (nome_cliente),  
       sum (saldo)  
from conta  
where  
      banco = "Banco do Brasil"  
group by  
      nome_agencia  
having  
      count (nome_cliente) > 10
```

Operadores Lógicos

- São comumente usados na cláusula WHERE para testar alguma condição;
- Operadores lógicos retornam um valor booleano **TRUE** ou **FALSE**;
- Operações Suportadas:

ALL	IN
AND	LIKE
ANY	NOT
IS NULL	SOME
BETWEEN	OR
EXISTS	

Operadores Lógicos (Exemplos)

- **AND:**
select cliente.nome, cliente.idade, conta.saldo
from conta, cliente
where conta.codigo_cliente = cliente.codigo and
saldo > 1000
- **OR:**
select codigo_cliente, saldo
from conta
where saldo < 100 or saldo > 1000
- **LIKE:**
select nome, idade
from cliente
where nome like "S%"

Cláusula ORDER BY

- Utilizada para ordenar os valores das colunas do resultado da consulta;
- Na ausência do ORDER BY os valores do resultado são ordenados de forma crescente, a partir do primeiro campo projetado;
- Exemplo:

```
select cliente.nome, cliente.idade, conta.saldo
from conta, cliente
where conta.codigo_cliente = cliente.codigo
order by
    cliente.nome asc,
    saldo desc,
    idade desc
```


Consultas Aninhadas

- Uma consulta aninhada é uma consulta que tem outra consulta embutida dentro dela;
- A consulta embutida é chamada subconsulta;
- Expressa uma condição que referencia uma tabela que precisa ser computada;
- Exemplo:

```
select cliente.nome, cliente.idade
from cliente, conta
where conta.codigo_cliente = cliente.codigo and
       conta.saldo >= (select avg(conta.saldo)
                       from conta)
```

Comandos de Definição

- São utilizados para descrever / definir o esquema do banco de dados;
- Estão definidos na **Data Definition Language (DDL)**;
- Contém comandos para criar, modificar e excluir definições do esquema do banco de dados;
- Também define restrições de integridade, direitos de acesso e privilégios para tabelas e visões.

Comando de Criação

- Especificado pela palavra reservada **CREATE**.
- Usada para:
 - CREATE DATABASE
 - CREATE TABLE
 - CREATE VIEW
 - CREATE INDEX
 - CREATE PROCEDURE

```
CREATE TABLE deposito (  
    numero_agencia integer not null,  
    numero_conta integer not null,  
    nome_cliente string,  
PRIMARY KEY (numero_agencia, numero_conta));
```

Comando de Alteração

- Especificado pela palavra reservada **ALTER**.
- Usada para:
 - ALTER TABLE
 - ALTER VIEW
 - ALTER PROCEDURE
 - ALTER TRIGGER

```
ALTER TABLE deposito  
ADD saldo float,  
DROP nome_cliente;
```

```
ALTER TABLE deposito  
ADD CONSTRAINT fk_deposito FOREIGN KEY  
(numero_agencia) REFERENCES agencia  
(numero_agencia) ON DELETE CASCADE;
```

Comando de Remoção

- Especificado pela palavra reservada **DROP**.
- Usada para:
 - DROP DATABASE
 - DROP TABLE
 - DROP VIEW
 - DROP INDEX
 - DROP PROCEDURE

```
DROP TABLE deposito;
```