SQL (*Structured Query Language*)

- Linguagem comercial para BD relacional
  - padrão ISO desde a década de 80
    - SQL-1 (86); SQL-2 (92); SQL-3 (99)
  - não é apenas uma linguagem de consulta!
    - como o nome sugere...
- Base Formal
  - álgebra relacional e cálculo relacional
- Funcionalidades principais
  - definição (DDL) e manipulação (DML) de dados
  - definição de visões e autorizações de acesso
  - definição de restrições de integridade
  - definição de transações
  - comandos para embutimento em LPs

SQL - DDL

- Criação de um BD
  - SQL padrão não oferece tal comando
    - BDs são criados via ferramentas do SGBD
  - alguns SGBDs (SQL Server, DB2, MySQL) oferecem este comando
    - `create database` *nome_BD*
    - `drop database` *nome_BD*

SQL - DDL

- Comandos para definição de esquemas
  - `create table`
    - define a estrutura da tabela, suas restrições de integridade e cria uma tabela vazia
  - `alter table`
    - modifica a definição de uma tabela (I / E / A atributos; I / E RIs)
  - `drop table`
    - remove uma tabela com todas as suas tuplas

SQL – *Create Table*

```
CREATE TABLE nome_tabela (
   nome_atributo_1 tipo_1 [[NOT]NULL][UNIQUE]
   [{, nome_atributo_n     tipo_n}]
   [, PRIMARY KEY (nome(s)_atributo(s))]
   [{, FOREIGN KEY (nome_atributo)
            REFERENCES nome_tabela}]
)
```

- Principais tipos de dados do MySQL
  - *int, smallint, tinyint, numeric(tamanho[,nro_casas_decimais]), char(tamanho), varchar(tamanho), date, time, datetime, ...*
  - formato para data e hora
    - "YYYY-MM-DD HH:MM:SS"

Exemplos de Criação de Tabela

```
CREATE TABLE Ambulatorios (
        nroa            int,
        andar           numeric(3) NOT NULL,
        capacidade      smallint,
        PRIMARY KEY(nroa)
)

CREATE TABLE Medicos (
   codm             int,
   nome             varchar(40) NOT NULL,
   idade             smallint NOT NULL,
   especialidade     char(20),
   CPF               numeric(11) UNIQUE,
   cidade            varchar(30),
   nroa              int,
   PRIMARY KEY(codm),
   FOREIGN KEY(nroa) REFERENCES Ambulatorios
)
```

SQL – *Alter Table*

```
ALTER TABLE nome_tabela
ADD [COLUMN] nome_atributo_1 tipo_1 [{RIs}]
   [{, nome_atributo_n  tipo_n [{RIs}]}]
|
MODIFY [COLUMN] nome_atributo_1 tipo_1 [{RIs}]
   [{, nome_atributo_n  tipo_n [{RIs}]}]
|
DROP COLUMN nome_atributo_1
   [{, nome_atributo_n }]
|
ADD CONSTRAINT nome_RI_1 def_RI_1
   [{, nome_RI_n def_RI_n}]
|
DROP CONSTRAINT nome_RI_1
   [{, nome_RI_n}]
|
[ADD|DROP] [PRIMARY KEY ...|FOREIGN KEY ...]
```

Exemplos de Alteração de Tabelas

```
ALTER TABLE Ambulatórios
   ADD nome VARCHAR(30)

ALTER TABLE Médicos DROP PRIMARY KEY

ALTER TABLE Pacientes DROP COLUMN doenca,
  DROP COLUMN cidade

ALTER TABLE Funcionários
ADD FOREIGN KEY(nroa)REFERENCES Ambulatórios

ALTER TABLE Funcionarios
   ADD constraint fk_nroa
   FOREIGN KEY(nroa) REFERENCES Ambulatorios
```

SQL – Índices

- Definidos sobre atributos para acelerar consultas a dados
- Índices são definidos automaticamente para chaves primárias
- Operações

```
CREATE [UNIQUE] INDEX nome_índice ON
nome_tabela (nome_atributo_1[{, nome_atributo_n }])

DROP INDEX nome_índice ON nome_tabela
```

- Exemplos

```
CREATE UNIQUE INDEX indPac_CPF ON Pacientes (CPF)

DROP INDEX indPac_CPF ON Pacientes
```

SQL – DML

- Define operações de manipulação de dados
  - I `(INSERT)`
  - A `(UPDATE)`
  - E `(DELETE)`
  - C `(SELECT)`

- Instruções declarativas
  - manipulação de conjuntos
  - especifica-se o *que fazer* e não *como fazer*

SQL – DML

- Inserção de dados

```
INSERT INTO nome_tabela [(lista_atributos)]
VALUES (lista_valores_atributos)
       [, (lista_valores_atributos)]
```

MySQL

- Exemplos

```
INSERT INTO Ambulatorios VALUES (1, 1, 30)

INSERT INTO Medicos
(codm, nome, idade, especialidade, CPF, cidade)
VALUES (4, 'Carlos', 28,'ortopedia',
          11000110000, 'Joinville');
```

SQL – DML

- Alteração de dados

```
UPDATE nome_tabela
SET nome_atributo_1 = Valor
    [{, nome_atributo_n = Valor}]
[WHERE condição]
```

- Exemplos

```
UPDATE Medicos
SET cidade = 'Florianopolis'

UPDATE Ambulatorios
SET capacidade = capacidade + 5, andar = 3
WHERE nroa = 2
```

SQL – DML

- Exclusão de dados

```
DELETE FROM nome_tabela
[WHERE condição]
```

- Exemplos

```
DELETE FROM Ambulatorios

DELETE FROM Medicos
WHERE especialidade = 'cardiologia'
or cidade < > 'Florianopolis'
```