



# Introdução a Sistemas de Bancos de Dados

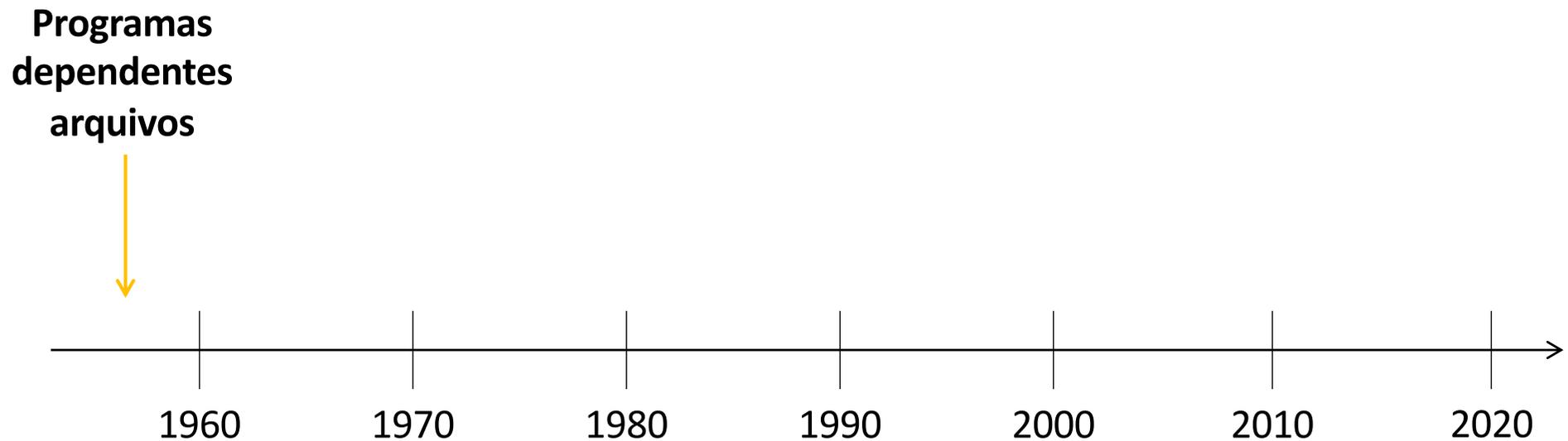
Gilberto Ribeiro de Queiroz

03 de Outubro de 2022

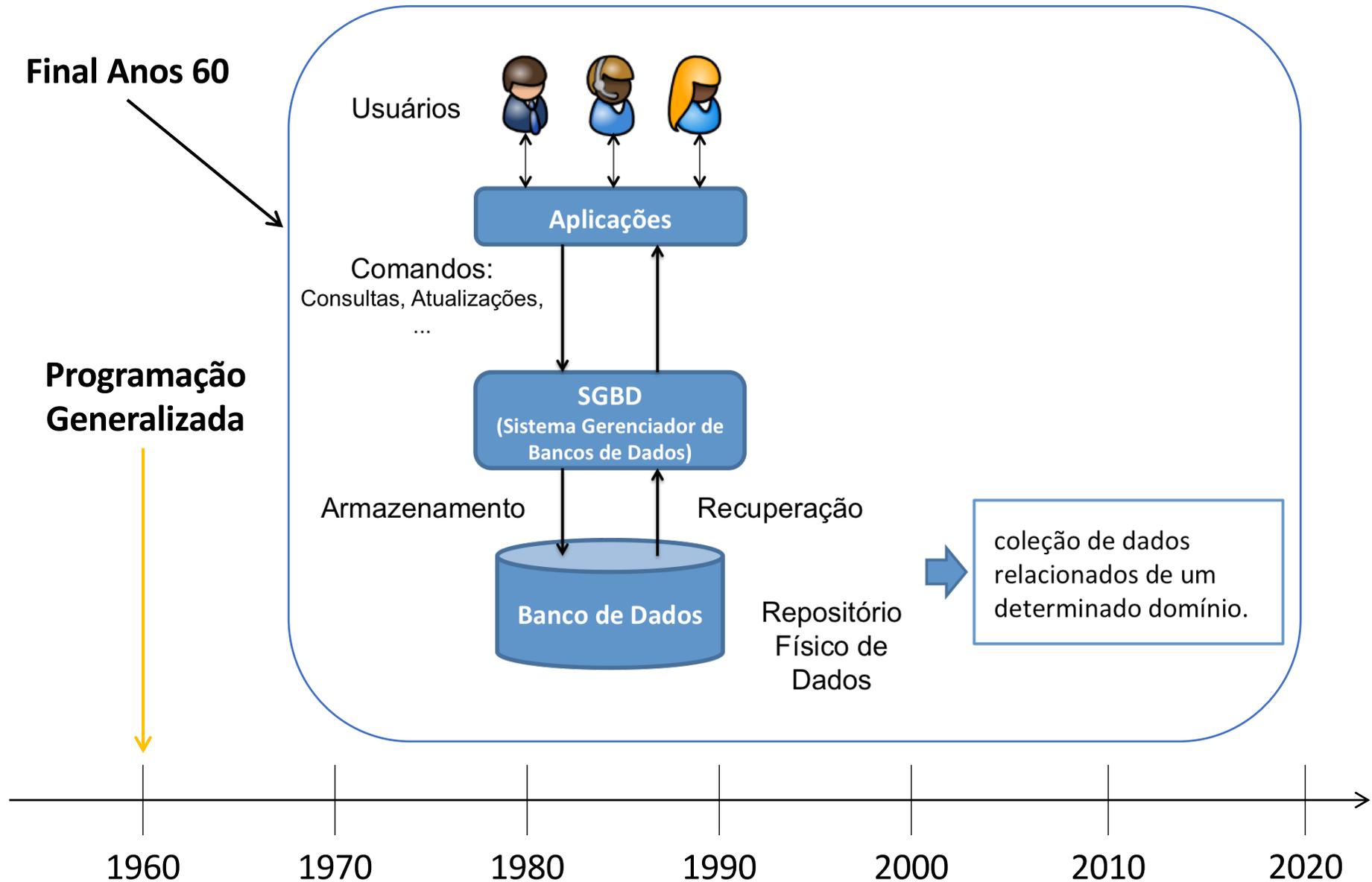
This work is licensed under a Creative Commons “Attribution-NonCommercial-ShareAlike 3.0 Unported” license.



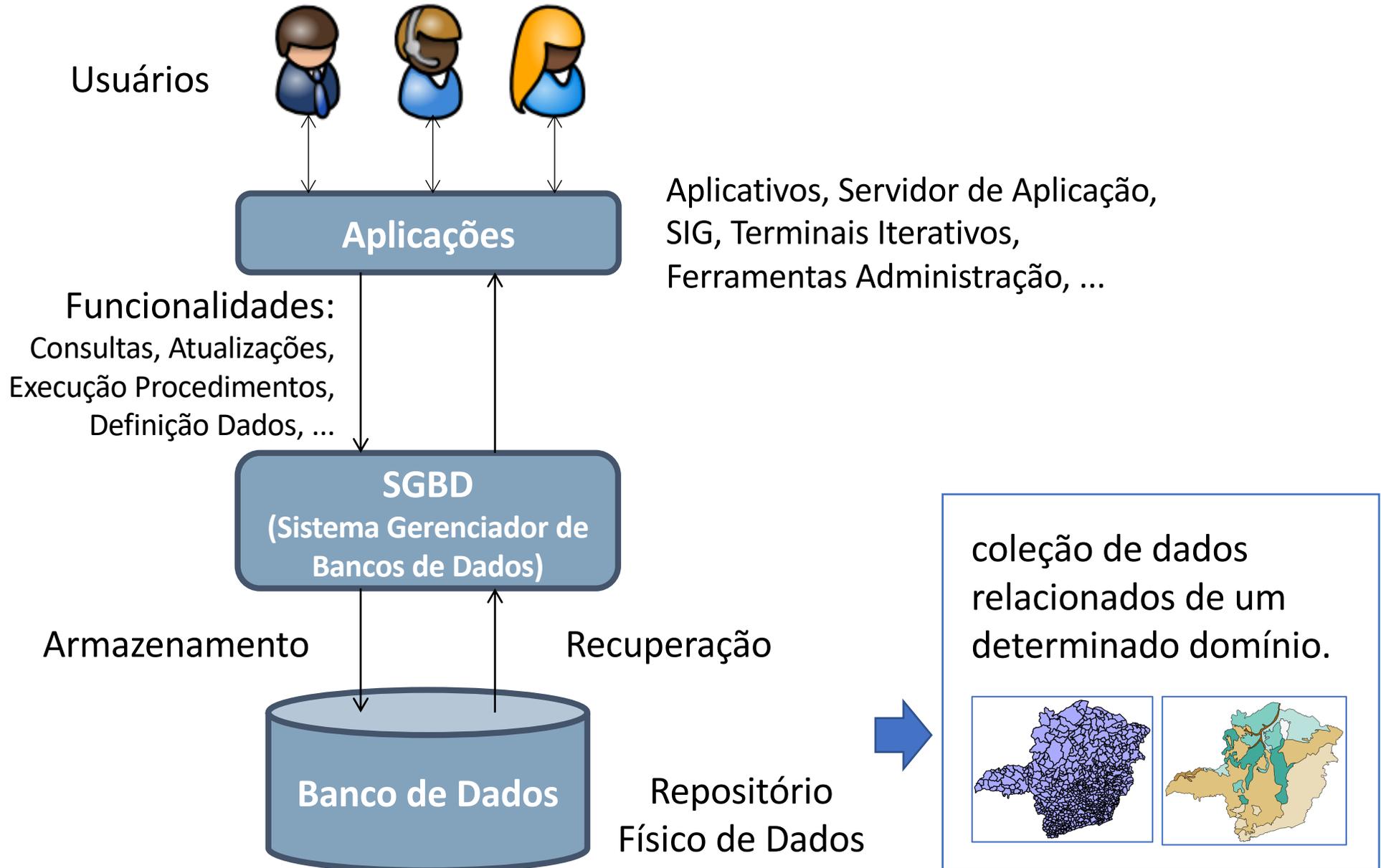
# Evolução das Tecnologias de Bancos Dados



# Evolução das Tecnologias de Bancos Dados



# Sistemas de Bancos de Dados

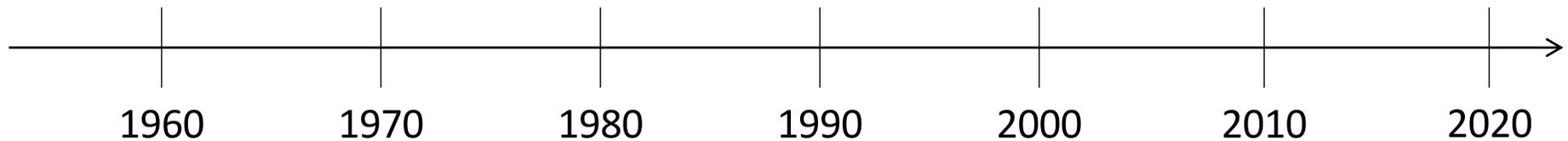


# Evolução das Tecnologias de Bancos Dados

**Modelo Bancos Dados Rede**

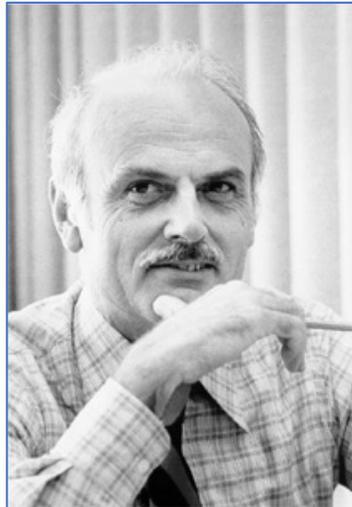


**Modelo Bancos Dados Hierárquico**



# Evolução das Tecnologias de Bancos Dados

ACM Turing Award (1981)



E. F. Codd. 1970. *A relational model of data for large shared data banks*. Communications of the ACM, v. 13, n. 6, June 1970, pp. 377-387.

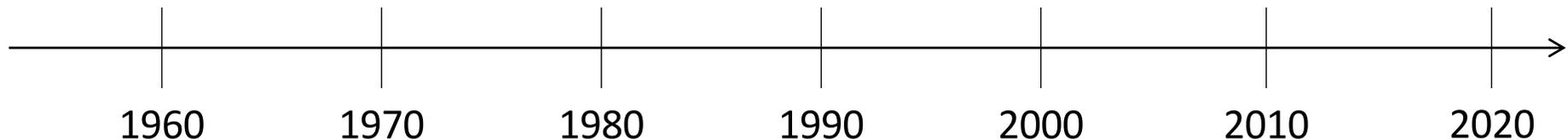
**Modelo Relacional**

Edgar Frank Codd  
Fonte: [Wikipedia](#)

**Bancos Dados Relacionais (SGBD-R)**

**Modelo Bancos Dados Rede**

**Modelo Bancos Dados Hierárquico**



Quais são os principais conceitos em  
Sistemas de Bancos de Dados  
Relacionais (SGBD-R)?

# Relação (ou Tabela)

- Um banco de dados relacional é organizado em uma coleção de relações (ou tabelas) possivelmente relacionadas entre si.

países			
id	nome	populacao	fronteira
1	Alemanha	82.000.000	
2	Brasil	190.000.000	
...	...	...	...

**Tabela** →

**Colunas** →

**Linha** →

**Esquema Tabela**

**Instância**

# Relacionamentos entre tabelas

países	
pid	nome
1	Alemanha
2	Brasil
...	...

cidades			
cid	nome	populacao	pais_id
3146107	Ouro Preto	70.227	2
3140001	Mariana	58.233	2
9879999	Munster	291.754	1
3131901	Itabirito	45.449	2
...	...	...	...

países_x_cidades				
pid	p_nome	cid	c_nome	c_populacao
2	Brasil	3146107	Ouro Preto	70.227
2	Brasil	3140001	Mariana	58.233
1	Alemanha	9879999	Munster	291.754
2	Brasil	3131901	Itabirito	45.449
...	...	...	...	...

# Chave Primária

- Campo ou conjunto de campos cujos valores identificam unicamente cada linha de uma tabela.
- Exemplo 1: Chave primária formada por um único campo.

países	
pid	nome
1	Alemanha
2	Brasil
...	...

cidades			
cid	nome	populacao	pais_id
3146107	Ouro Preto	70.227	2
3140001	Mariana	58.233	2
9879999	Munster	291.754	1
3131901	Itabirito	45.449	2
...	...	...	...

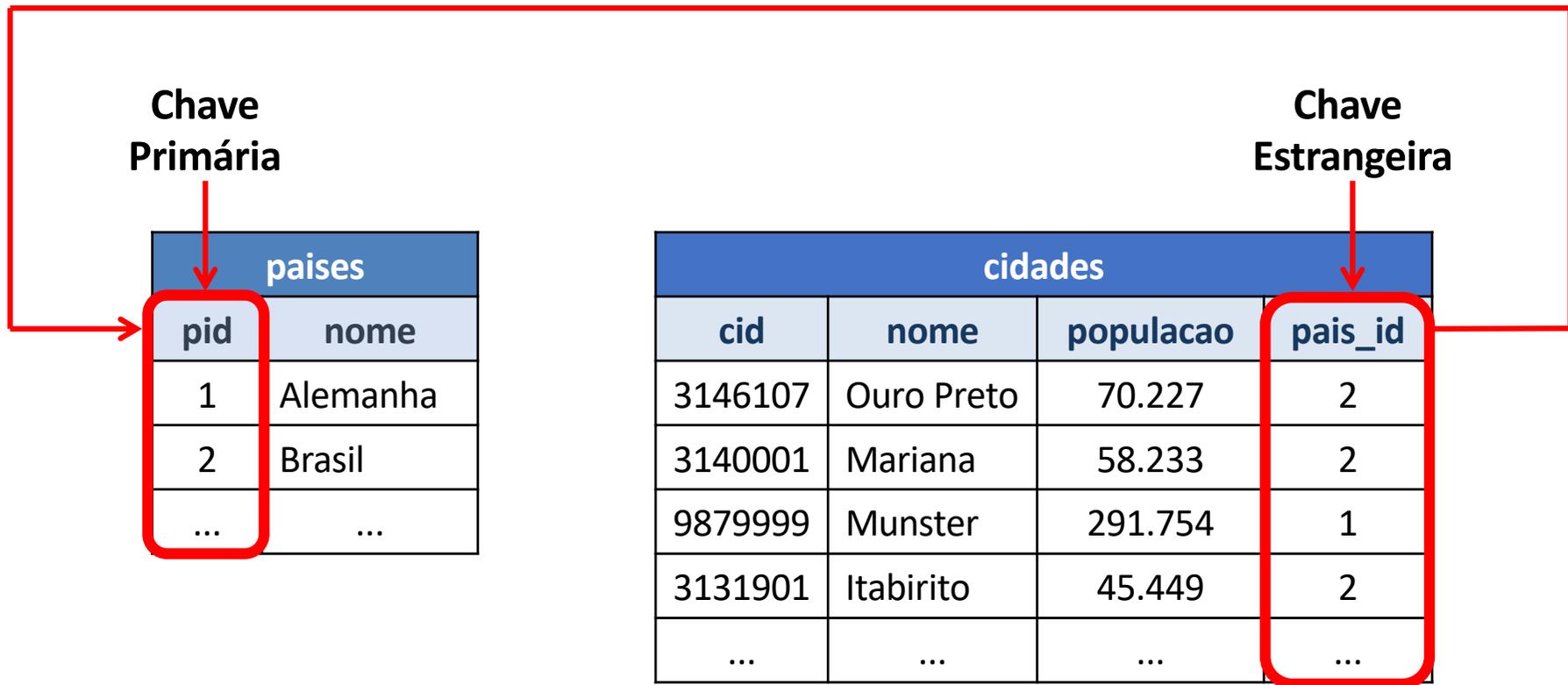
# Chave Primária (Primary Key)

- Campo ou conjunto de campos cujos valores identificam unicamente cada linha de uma tabela.
- Exemplo 2: Chave primária composta.

alunos_disciplinas		
id_aluno	id_disciplina	nota
5	1	9
5	2	10
3	8	8
4	6	7
2	1	3
2	8	10
...	...	...

# Chave Estrangeira (Foreign Key)

- Coluna ou combinação de colunas, cujos valores aparecem necessariamente na chave primária de uma outra tabela\*.



\*uma chave estrangeira não precisa ter o mesmo nome do que a chave primária correspondente na outra tabela (apenas o mesmo domínio)

# Álgebra Relacional

- **Exemplo:**

- **Operador de Seleção:** seleciona tuplas de uma relação que satisfazem um certo predicado (ou condição).
- **Consulta:** selecionar as cidades com população acima de 60.000 habitantes.

cidades			
cid	nome	populacao	pais_id
...	Ouro Preto	70.227	2
...	Mariana	58.233	2
...	Munster	291.754	1
...	Itabirito	45.449	2

Tabela/Relação de Entrada

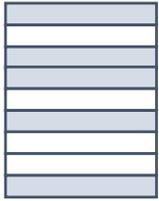
$\sigma_{populacao > 60000}(cidades)$

nova_relação			
cid	nome	populacao	pais_id
...	Ouro Preto	70.227	2
...	Munster	291.754	1

Tabela/Relação de Saída

# Álgebra Relacional: Operadores

Selecionar



Projetar



R1

a <sub>1</sub>	b <sub>1</sub>	c <sub>1</sub>
a <sub>2</sub>	b <sub>2</sub>	c <sub>2</sub>
a <sub>3</sub>	b <sub>3</sub>	c <sub>3</sub>

R2

d <sub>1</sub>	e <sub>1</sub>
d <sub>2</sub>	e <sub>2</sub>

Produto Cartesiano

a <sub>1</sub>	b <sub>1</sub>	c <sub>1</sub>	d <sub>1</sub>	e <sub>1</sub>
a <sub>1</sub>	b <sub>1</sub>	c <sub>1</sub>	d <sub>2</sub>	e <sub>2</sub>
a <sub>2</sub>	b <sub>2</sub>	c <sub>2</sub>	d <sub>1</sub>	e <sub>1</sub>
a <sub>2</sub>	b <sub>2</sub>	c <sub>2</sub>	d <sub>2</sub>	e <sub>2</sub>
a <sub>3</sub>	b <sub>3</sub>	c <sub>3</sub>	d <sub>1</sub>	e <sub>1</sub>
a <sub>3</sub>	b <sub>3</sub>	c <sub>3</sub>	d <sub>2</sub>	e <sub>2</sub>

R1

a <sub>1</sub>	b <sub>1</sub>	c <sub>1</sub>
a <sub>2</sub>	b <sub>2</sub>	c <sub>2</sub>
a <sub>3</sub>	b <sub>3</sub>	c <sub>3</sub>

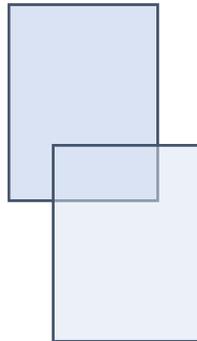
R2

c <sub>1</sub>	d <sub>1</sub>
c <sub>3</sub>	d <sub>2</sub>

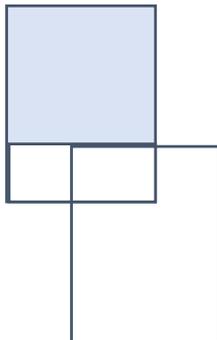
Junção (Natural)

a <sub>1</sub>	b <sub>1</sub>	c <sub>1</sub>	c <sub>1</sub>	d <sub>1</sub>
a <sub>3</sub>	b <sub>3</sub>	c <sub>3</sub>	c <sub>3</sub>	d <sub>2</sub>

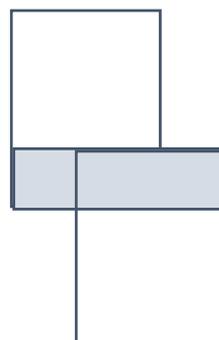
União



Diferença



Interseção



# Linguagem de Consulta: SQL

- O modelo relacional é a base para linguagens de alto nível:
  - Álgebra/Cálculo Relacional → Linguagem Declarativa → ISO/SQL (Structured Query Language)

```
CREATE TABLE paises  
(  
  pid INT4 PRIMARY KEY,  
  nome VARCHAR(50),  
);
```

Definição Dados

paises	
pid	nome

Manipulação  
Dados

paises	
pid	nome
1	Alemanha
2	Brasil
...	...

Manipulação  
Dados

```
INSERT INTO paises  
  VALUES (1, 'Alemanha');  
  
INSERT INTO paises  
  VALUES (2, 'Brasil');
```

# Linguagem de Consulta: SQL

- O modelo relacional é a base para linguagens de alto nível:
  - Álgebra/Cálculo Relacional → Linguagem Declarativa → ISO/SQL (Structured Query Language)

cidades			
cid	nome	populacao	pais_id
...	Ouro Preto	70.227	2
...	Mariana	58.233	2
...	Munster	291.754	1
...	Itabirito	45.449	2

```
SELECT nome  
FROM cidades  
WHERE populacao > 60000
```

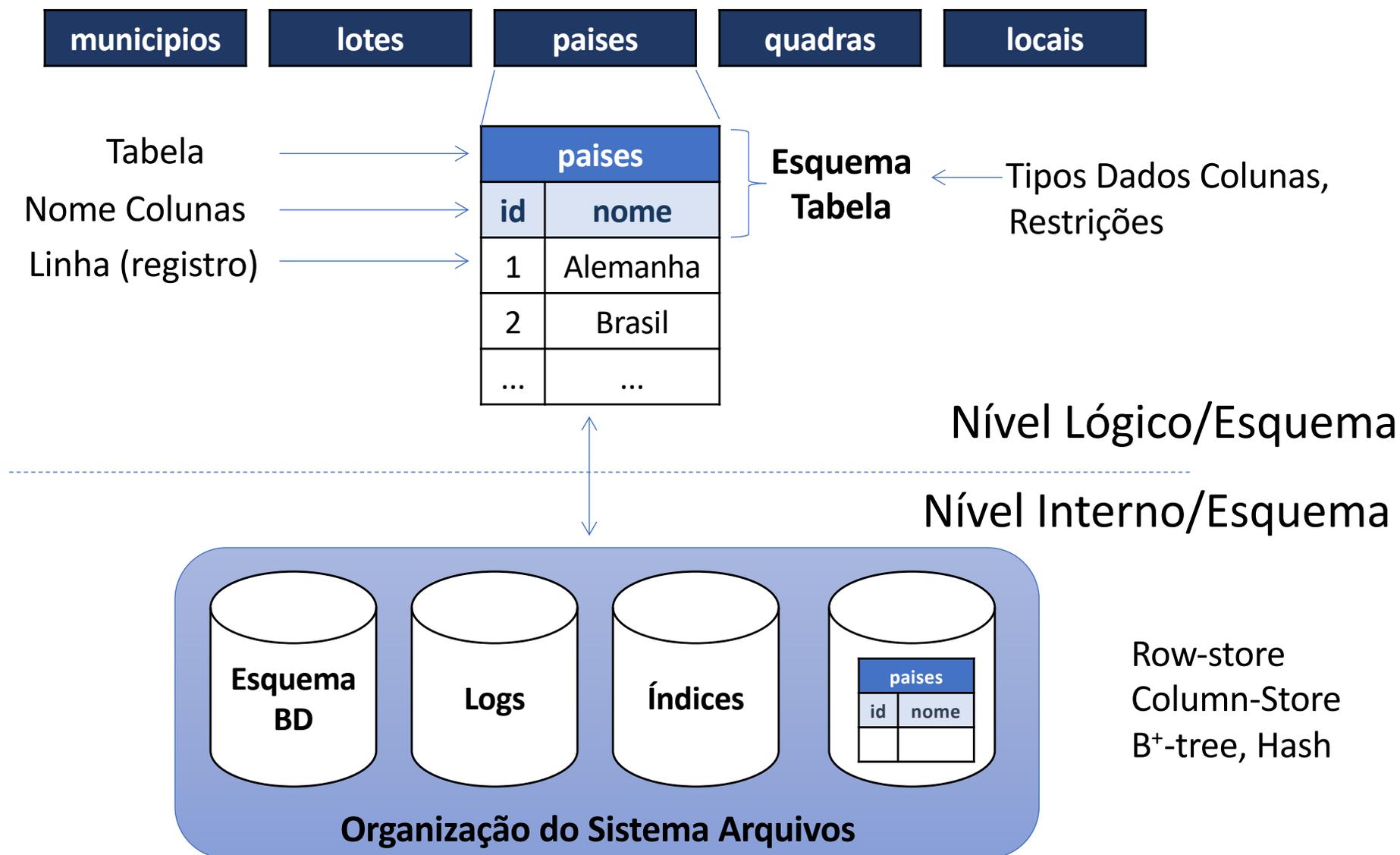
Consulta (Não-Procedural)

resultado
nome
Ouro Preto
Munster

# Prática

Criando e Consultando Tabelas no PostgreSQL

# Independência Física dos Dados

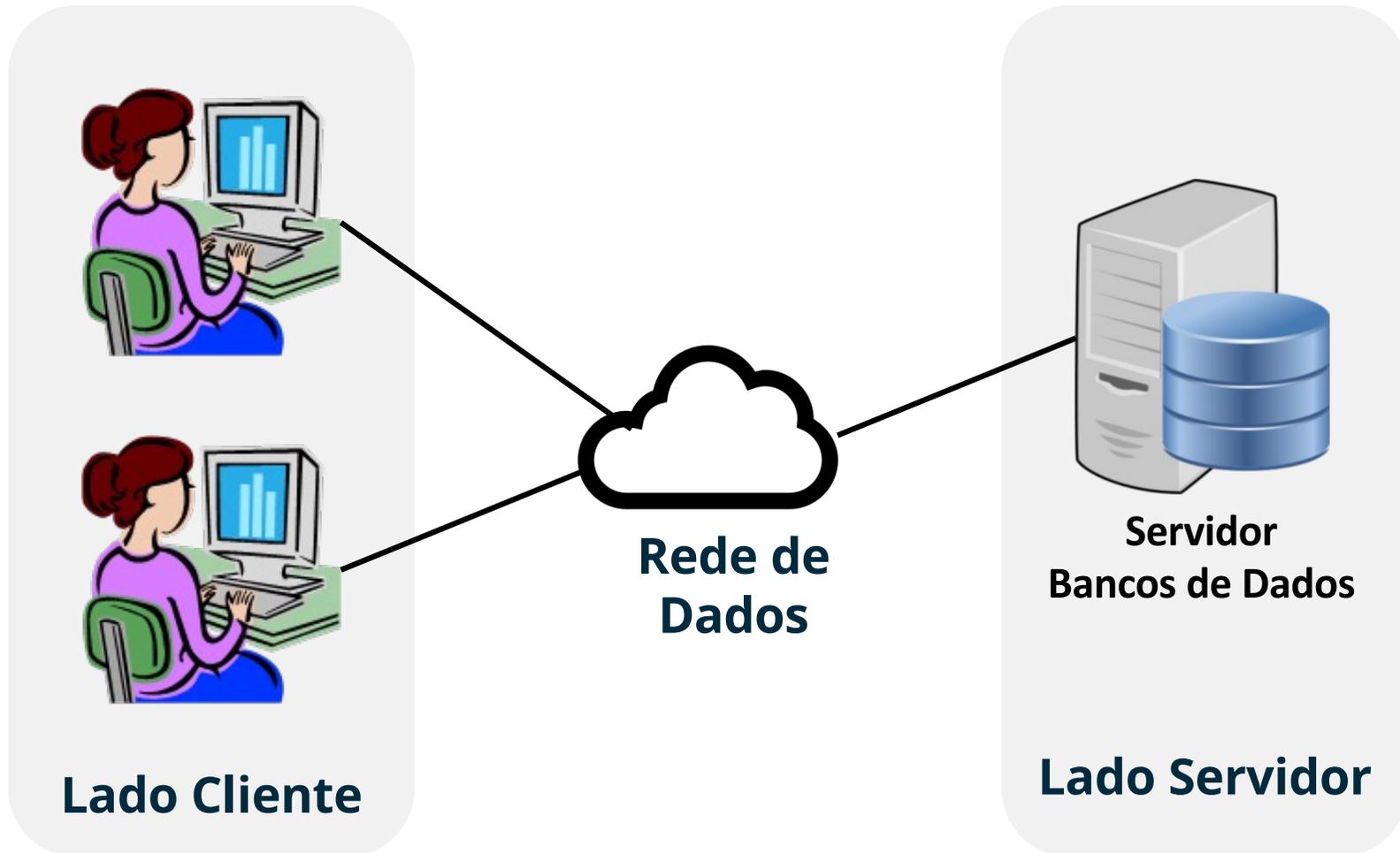


Fonte: Adaptado de Gray (1996)

# Arquiteturas de Sistemas Bancos de Dados Relacionais

- Cliente/Servidor
- Embutido (ou embarcado)
- Em memória (In-memory)
- Paralelos/Distribuídos
- Armazenamento Linha x Coluna

# Arquitetura: Cliente/Servidor



Consulte a definição na [Wikipedia](#)

# Arquitetura: Embedded

```
#include <sqlite3.h>

int main(int argc, char** argv) {
    int rc = sqlite3_open("/opt/data/mydb.sqlite", &db);

    if(rc) {
        fprintf(stderr, "Can't open database: %s\n", sqlite3_errmsg(db));
        sqlite3_close(db);
        return EXIT_FAILURE;
    }

    rc = sqlite3_exec(db, "Select * from tabela", callback, 0, &zErrMsg);

    if( rc!=SQLITE_OK ){
        char* zErrMsg = 0;
        fprintf(stderr, "SQL error: %s\n", zErrMsg);
        sqlite3_free(zErrMsg);
    }

    sqlite3_close(db);
    return EXIT_SUCCESS;
}
```



# Arquitetura: row store x column store

países		
id	nome	populacao
1	Alemanha	82.000.000
2	Brasil	190.000.000
...	...	...

Row Store

Layout em Disco

1	Alemanha	82 M	2	Brasil	190 M
---	----------	------	---	--------	-------

3	Argentina	...	...	...	...
---	-----------	-----	-----	-----	-----

Ex: PostgreSQL, MySQL

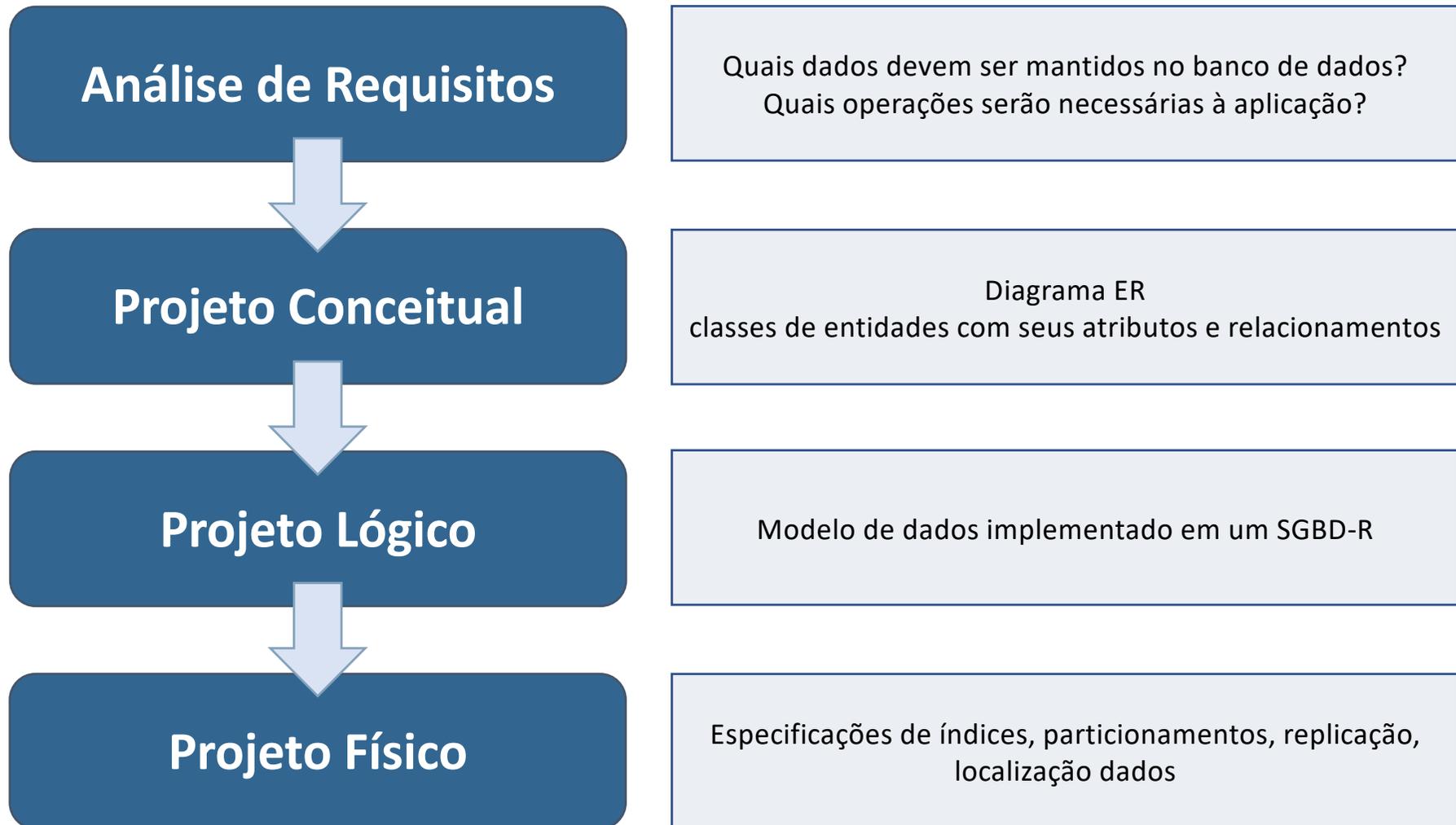
Column Store

Layout em Disco

1	Alemanha	82.000.000
2	Brasil	190.000.000
...	...	...

Ex: C-Store, MonetDB, Vertica

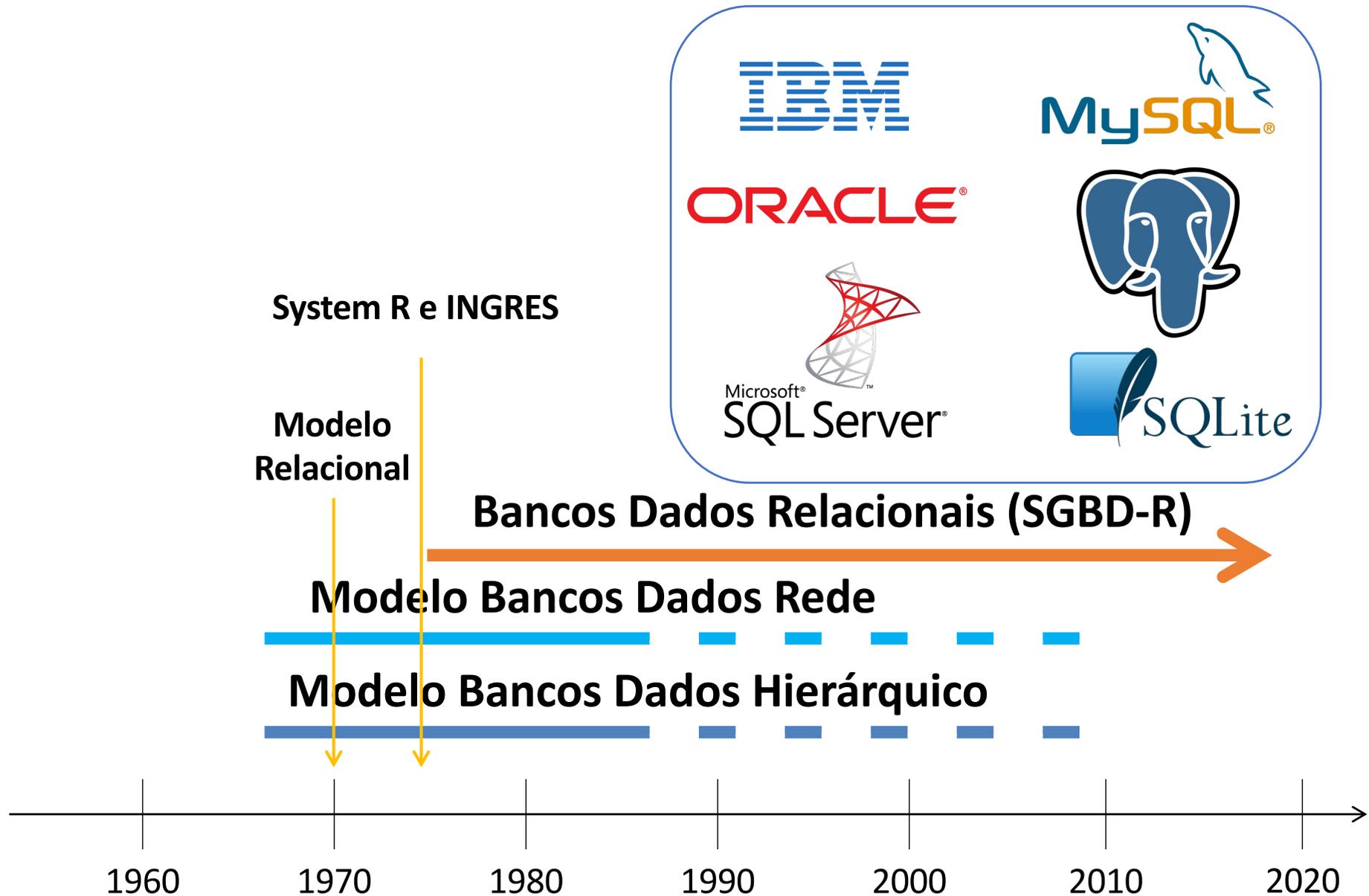
# Projeto de Bancos de Dados



# Outros Conceitos Importantes

- Normalização:
  - Evitar anomalias com o projeto do banco de dados.
- Transações (ACID).
- Gatilhos (Trigger).
- Procedimentos Armazenados (Stored Procedure).

# Evolução das Tecnologias de Bancos Dados



# Evolução das Tecnologias de Bancos Dados

**Aplicações emergentes e novas demandas:**  
CAD, SIG, Multimedia,  
OLAP, Real-time,  
Científicas

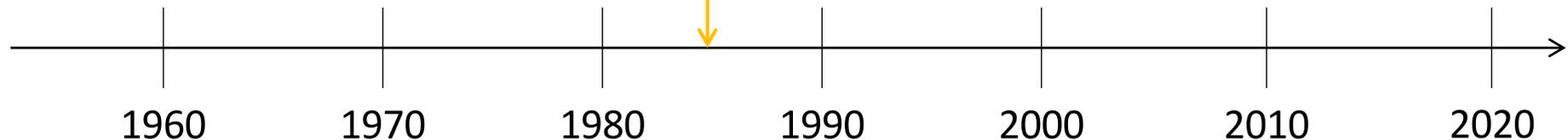
**Período de muita pesquisa sobre extensibilidade dos SGBDs**

**Protótipos de pesquisa no final dos anos 80 voltados a SIG:**  
Probe, DASDBS GEO-Kernel, Gral, SIRO-DBMS, Starburst, Geo++, GéoSabrina, GODOT, GeoO<sub>2</sub>, Paradise

**Bancos Dados Relacionais (SGBD-R)**

**Modelo Bancos Dados Rede**

**Modelo Bancos Dados Hierárquico**



# Gral: An Extensible Relational Database System for Geometric Applications

Ralf Hartmut Güting

Fachbereich Informatik, Universität Dortmund  
D-4600 Dortmund 50, West Germany

**Abstract:** We describe the architecture of a relational database system that is extensible by user-defined data types and operations, including relation operations. The central concept is to use languages based on many-sorted algebra to represent queries as well as query execution plans. This leads to a simple and clean extensible system architecture, eases the task of an application developer by providing a uniform framework, and also simplifies rule-based optimization. As a case study the extensions needed for a geometric database system are considered.

## 1. Introduction

Much of the database research of recent years was aimed at providing a better support for non-standard applications such as office information systems, geographic information systems, CAD databases, etc. A common need of these applications is the representation and manipulation of more complex objects than those representable by a tuple of a relation in the traditional relational model, for example, an office form, a complete map or a river, say, in a geographic information system, or the design of a VLSI circuit.

A fundamental choice for the representation of a complex

A lot of work has been done to support the modeling of visible object structures. Enhancements to the relational model have been proposed by linking together tuples to represent an object either explicitly [Co79, HaL82] or implicitly, through the use of nested relations [ScS86]. Most of the more recent data models and system proposals do also support structural object orientation, for example [MaD86, CaDV88, PiT86].

The idea of allowing application-specific abstract data types as base types, or attribute domains, of a database system was perhaps first put forward in [StRG83]. Since base types need to be implemented in a programming language and because they are application-specific, a user must be able to implement such a type and to add it to a database system. This observation has led to efforts by several groups to construct *extensible* database systems. Two directions can be distinguished. One is to select a data model and to implement for this data model a system with well-defined interfaces for user extensions. This is the approach chosen by the POSTGRES [StR86] and Starburst [Schw86] projects, based on the relational model, and within the PROBE project [Daya87] for an extended functional data model. A different view is taken in the EXODUS [Care86] and GENESIS [Bato86] projects where a collection of powerful tools for

R. H. Güting. 1989. Gral: an extensible relational database system for geometric applications. In Proceedings of the 15th international conference on Very large data bases (VLDB '89). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 33-44.

# THE **GEO++** SYSTEM: AN EXTENSIBLE GIS

Tom Vijlbrief

TNO Institute for Perception,

P.O. Box 23, 3769 ZG Soesterberg, The Netherlands.

Email: tom@izf.tno.nl

*and*

Peter van Oosterom

TNO Physics and Electronics Laboratory,

P.O. Box 96864, 2509 JG The Hague, The Netherlands.

Email: oosterom@fel.tno.nl

In this paper we present a classification of the architectures of Geographic Information Systems (GIS). Most commercial GISs are closed. This means that if certain functionality is not available, it is impossible for the users to extend or modify the system for their own purpose. We then present a solution based on the extensible database management system “Postgres”, in which new data types and operators may be defined. The resulting extensible GIS is called *GEO++*. We illustrate this powerful capability with two examples.

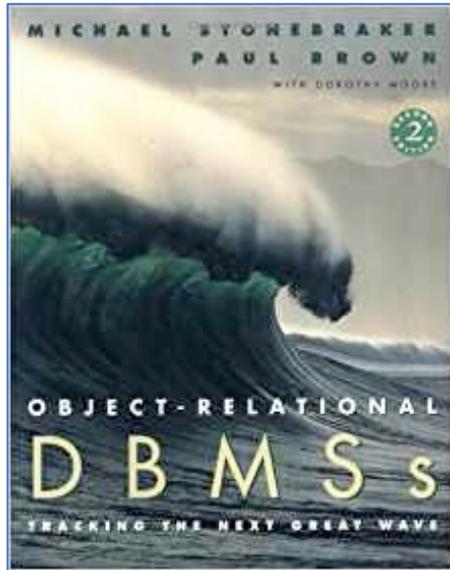
## 1 Introduction

Most commercial Geographic Information Systems (GISs) are based on a relational DataBase Management System (DBMS), such as Oracle or Ingres. One obvious drawback of the standard DBMSs is that they cannot manipulate geographic data. That is, there are no ge-

Rowe, & Hirohama, 1990) that we implemented. Section 4 enumerates the basic capabilities of our Postgres GIS front-end *GEO++*. The implementation of the system has been described in an earlier paper (van Oosterom & Vijlbrief, 1991). The real power of *GEO++* is demonstrated in Section 5, in which the system is extended with user-defined types.

Vijlbrief, T. and van Oosterom, P. The *GEO++* system: An extensible GIS. Proceedings of the Fifth **International Symposium on Spatial Data Handling**, Charleston, SC, 1992.

# Evolução das Tecnologias de Bancos Dados



Stonebraker et al. (1996)

Difusão dos SGBD-OR

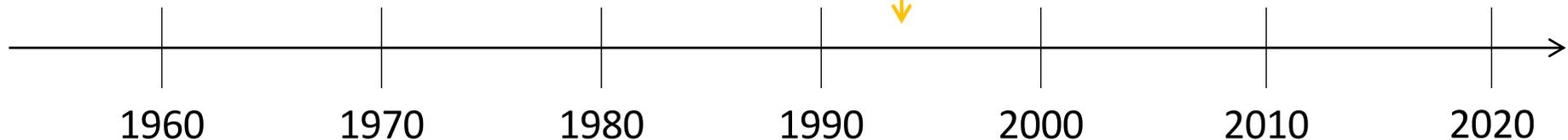
**Objeto Relacional**

**Bancos Dados Orientado Objeto**

**Bancos Dados Relacionais (SGBD-R)**

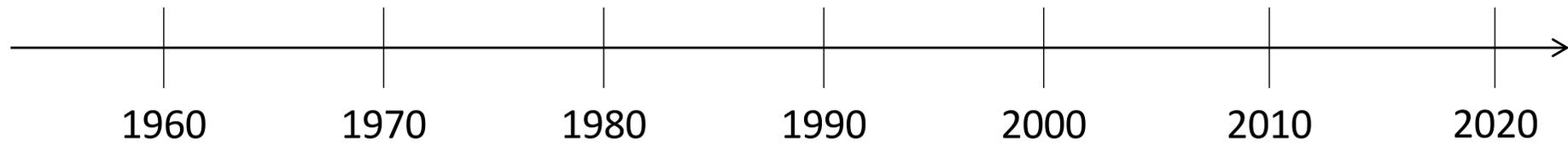
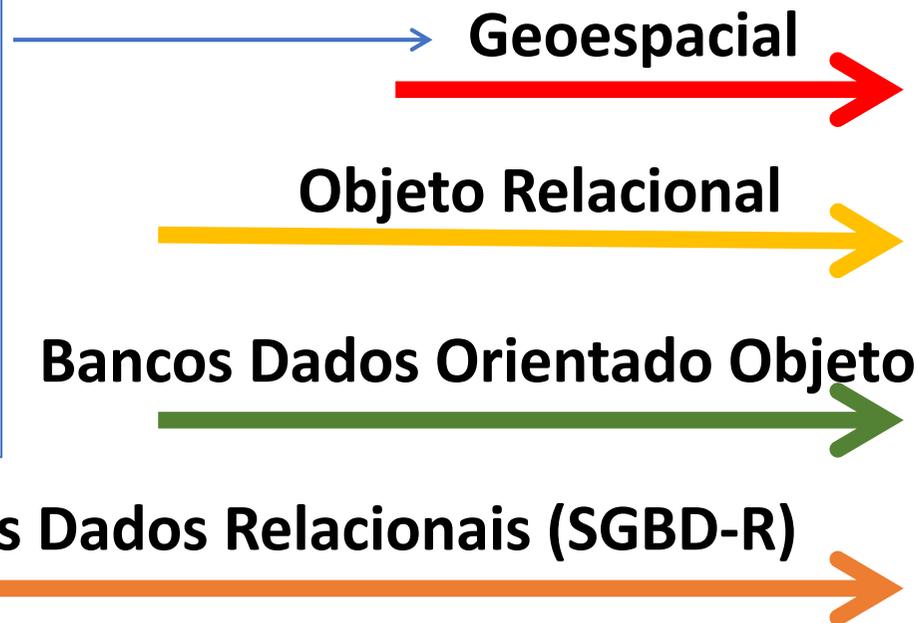
**Modelo Bancos Dados Rede**

**Modelo Bancos Dados Hierárquico**

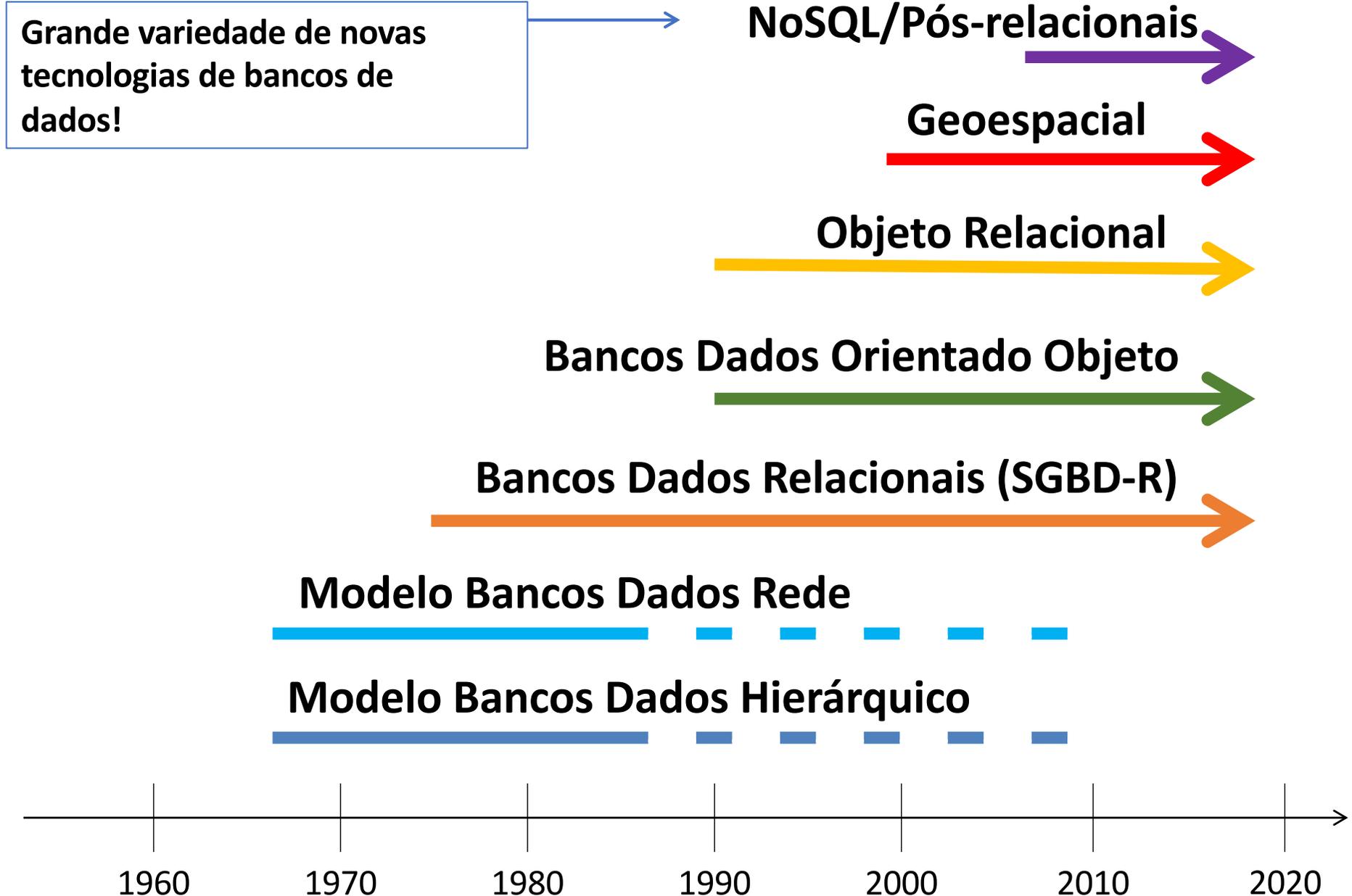


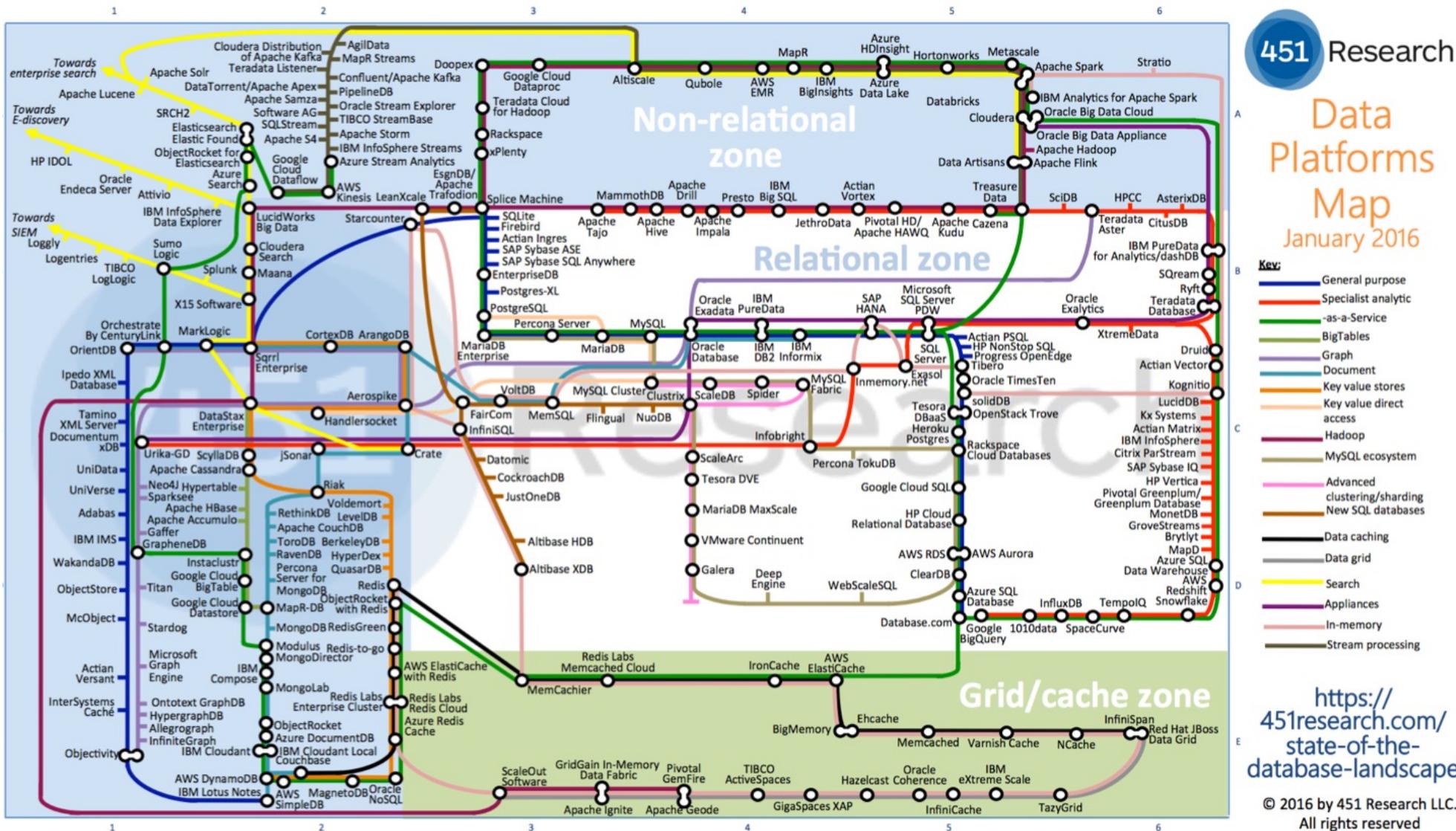
# Evolução das Tecnologias de Bancos Dados

PostgreSQL → PostGIS  
MySQL → Spatial and Geodetic Geography Types  
SQLite → Spatialite and RasterLite  
Oracle → Oracle Spatial, GeoRaster, Topology and Network Models  
IBM DB2 → Spatial Extender  
SQL Server (2008) → Spatial Types



# Evolução das Tecnologias de Bancos Dados





# Tecnologías de Bancos de Datos

# Considerações Finais

# Considerações Finais

- Nesta aula apresentamos um panorama geral das tecnologias de bancos dados e os conceitos básicos, com foco na tecnologia relacional.
- Um banco de dados é um conjunto organizado de dados que pode ser explorado das mais diversas formas.
- Sistema de Gerenciamento de Bancos de Dados (SGBD):
  - Formado por um conjunto de programas que permite criar, manter e manipular um banco de dados

# Considerações Finais

- Algumas vantagens do uso de um Sistema de Bancos de Dados:
  - Maior abstração dos dados, com o isolamento entre as aplicações e a forma física de armazenamento e recuperação dos dados.
  - Suporte a múltiplas visões dos dados.
  - Compartilhamento dos dados em um ambiente multiusuário, com controle de concorrência e recuperação a falhas, além de mecanismos de segurança para controle de acesso aos dados.
  - Manutenção de restrições de integridade.
  - Fornece acesso eficiente a grandes volumes de dados.

# Referências Bibliográficas

# Livros

- ELMASRI, R.; NAVATHE, S. B. ***Fundamentals of database systems***. Addison Wesley, 2006. 1139p.
- DATE, C. J. ***An introduction to database systems***. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1991.

# Artigos

- E. F. Codd. 1970. ***A relational model of data for large shared data banks.*** *Communications of the ACM*, v. 13, n. 6, June 1970, pp. 377-387.
- Chen, P. ***The Entity-Relationship Model-Toward a Unified View of Data.*** *ACM Transactions on Database Systems*, v. 1, n. 1, 1976, pp. 9-36.
- GRAY, J. ***Evolution of Data Management.*** *IEEE Computer*, v. 29, n. 10, 1996, pp. 38-46.

# Artigos

- R. H. Güting. ***Gral: an extensible relational database system for geometric applications***. In Proceedings of the 15th international conference on Very large data bases (VLDB '89). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1989, pp. 33-44.
- Vijlbrief, T., and P. van Oosterom. ***The GEO++ System: An Extensible GIS***. Proc. 5th Intl. Symposium on Spatial Data Handling, Charleston, South Carolina, 1992, pp. 40-50.
- Dangermond, J. ***A Classification of Software Components Commonly Used in Geographic Information Systems***. In Proceedings of the U.S.-Australia Workshop on the Design and Implementation of Computer-Based Geographic Information Systems, Honolulu, HI, 1982, pp. 70–91.