Bancos de Dados Geográficos

#04 – Introdução a Sistemas de Bancos de Dados

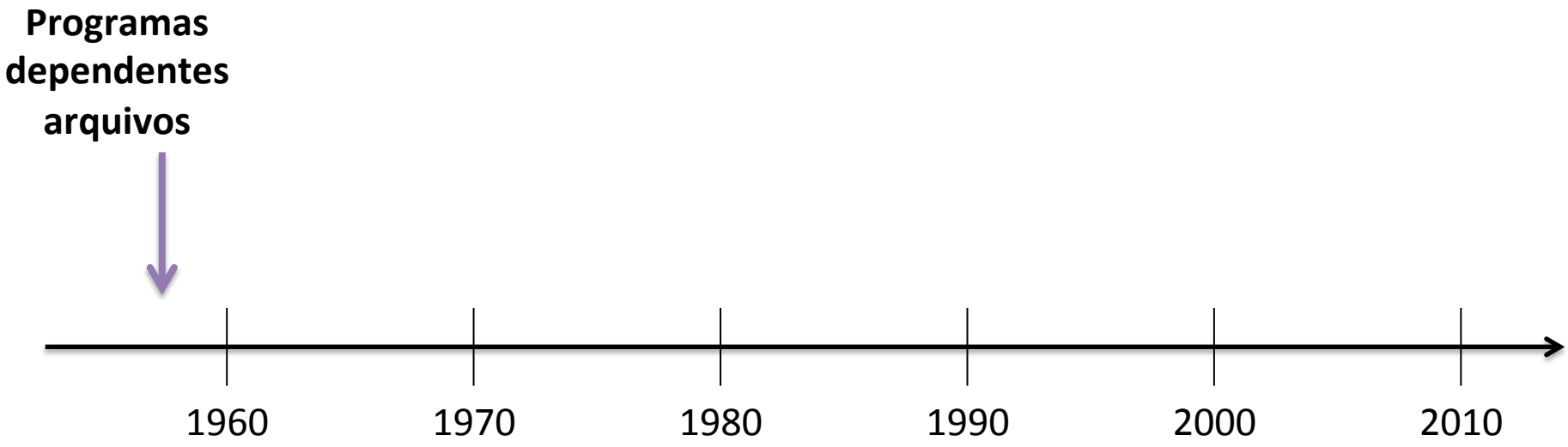
Dr. Gilberto Ribeiro de Queiroz <gribeiro@dpi.inpe.br>

Dr. Eymar Lopes <eymar@dpi.inpe.br>

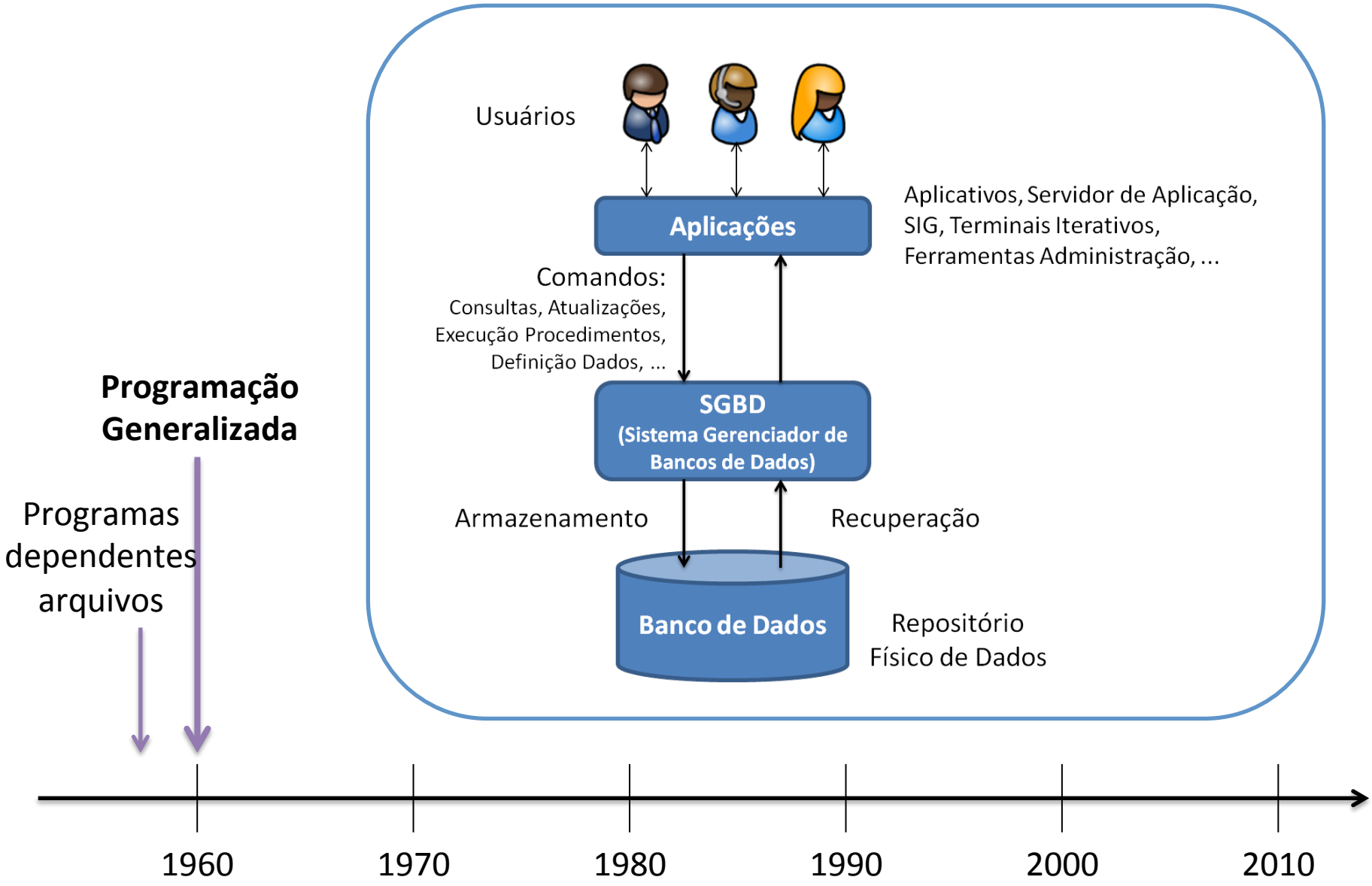
SGBD: uma tecnologia amplamente difundida

- A tecnologia de bancos de dados tem sido um componente fundamental em quase todos os tipos de aplicações:
  - Conta bancária: depósitos e saques
  - Reservas de passagens aéreas
  - Reservas em hotéis
  - Compras de livros, CDs, DVDs e outros bens (Amazon)
  - Busca por artigos em uma revista eletrônica (Transactions of GIS ou ACM digital library)
  - Sites de mapeamento: OpenStreetMap, GoogleMaps e Bing Maps

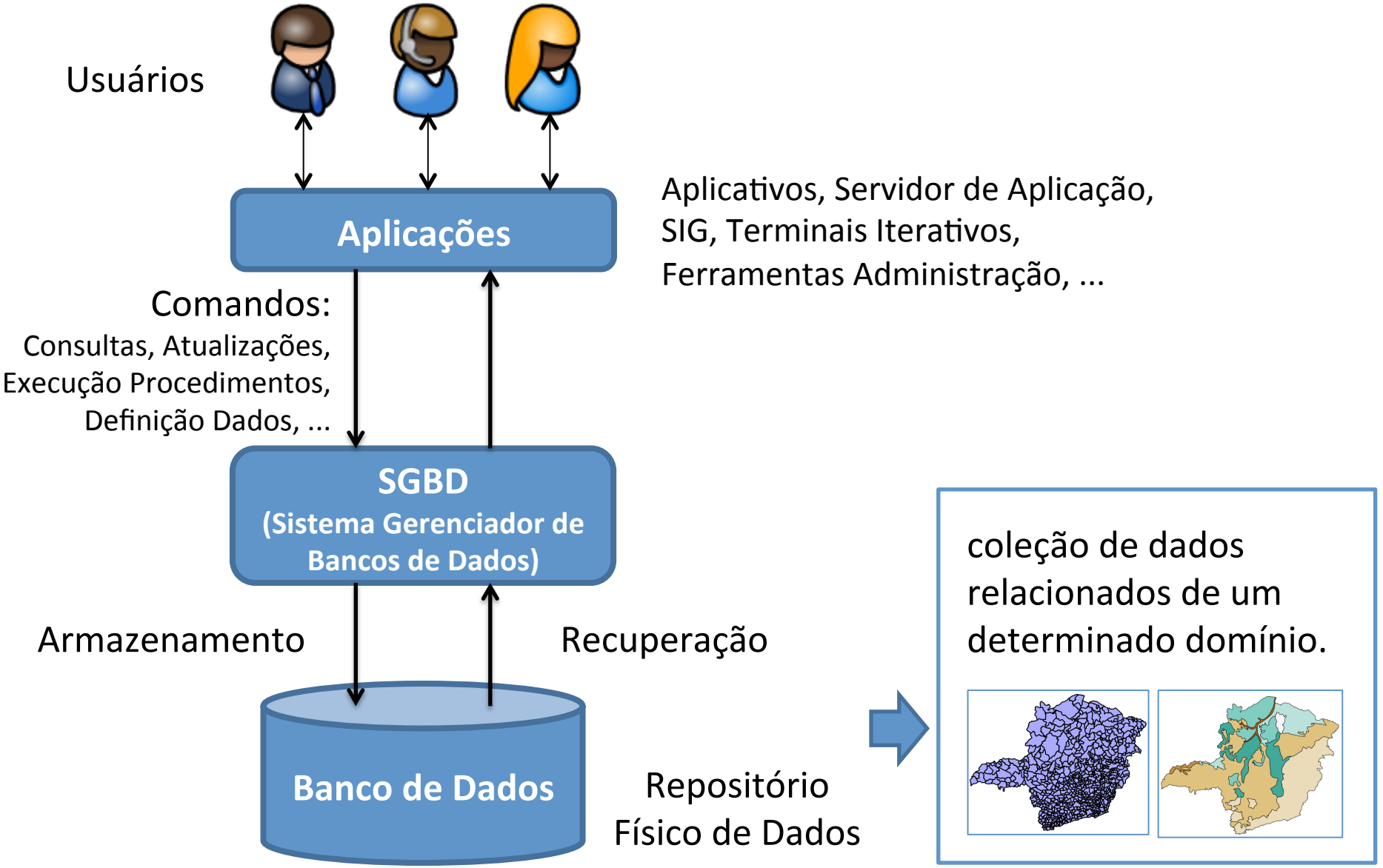
Evolução das Tecnologias de Bancos Dados



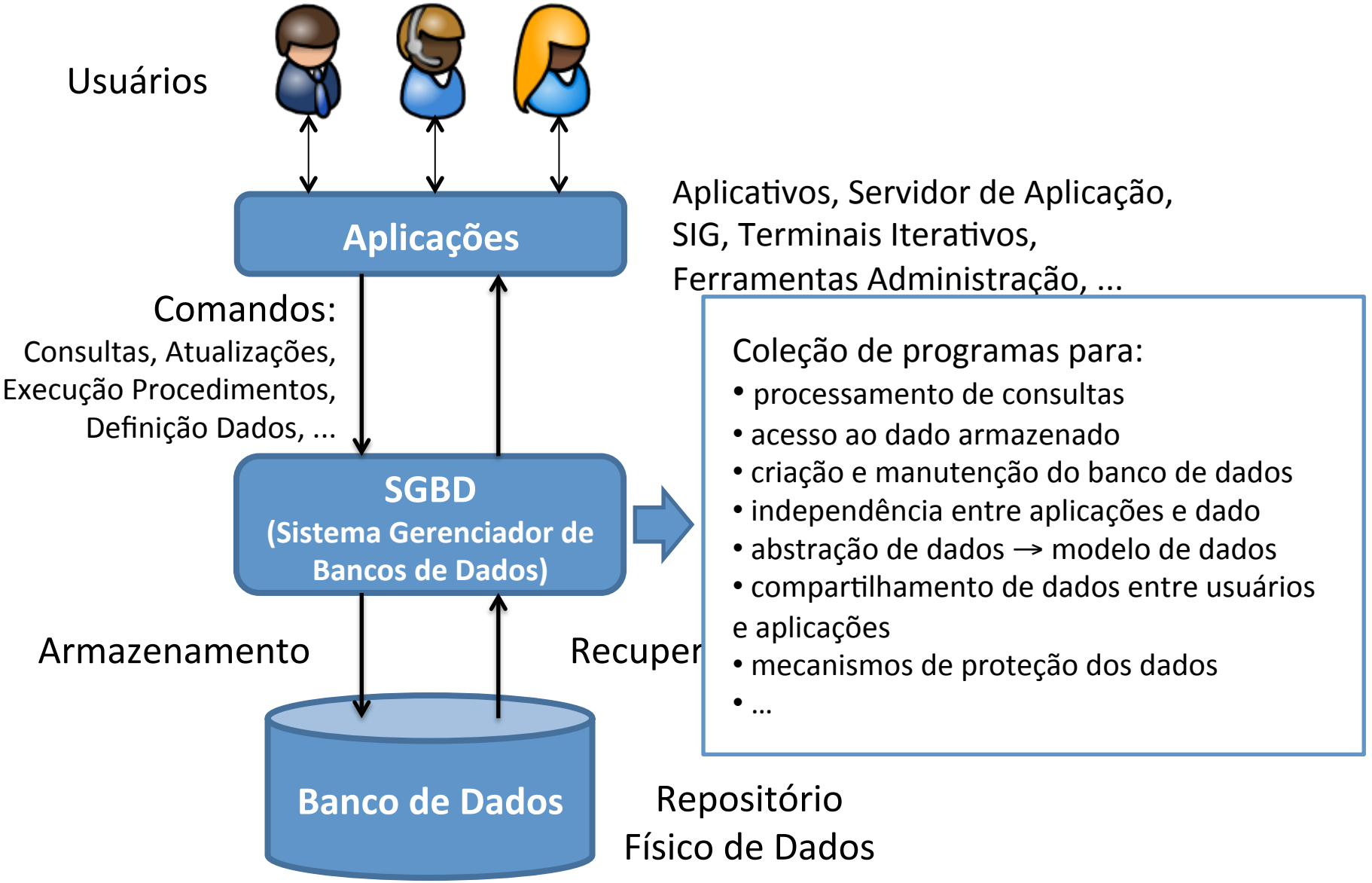
Evolução das Tecnologias de Bancos Dados



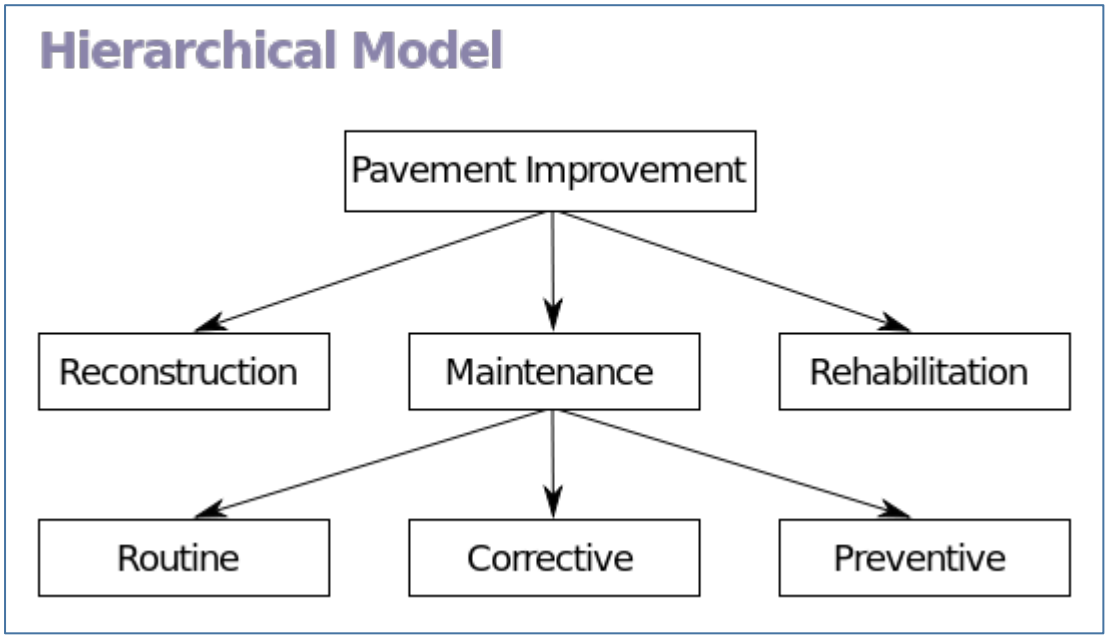
Sistemas de Bancos de Dados



Sistemas de Bancos de Dados



Evolução das Tecnologias de Bancos Dados



Fonte: Wikipedia

Programação Generalizada

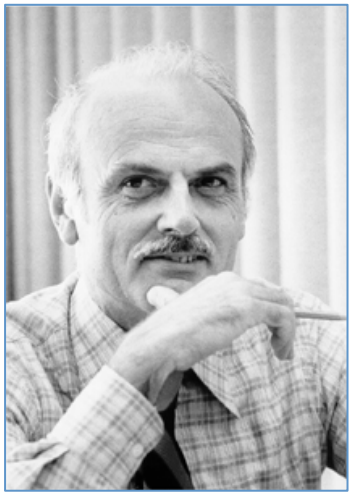
Programas dependentes arquivos

Modelo Banco Dados Hierárquico (IBM IMS)

1960 1970 1980 1990 2000 2010

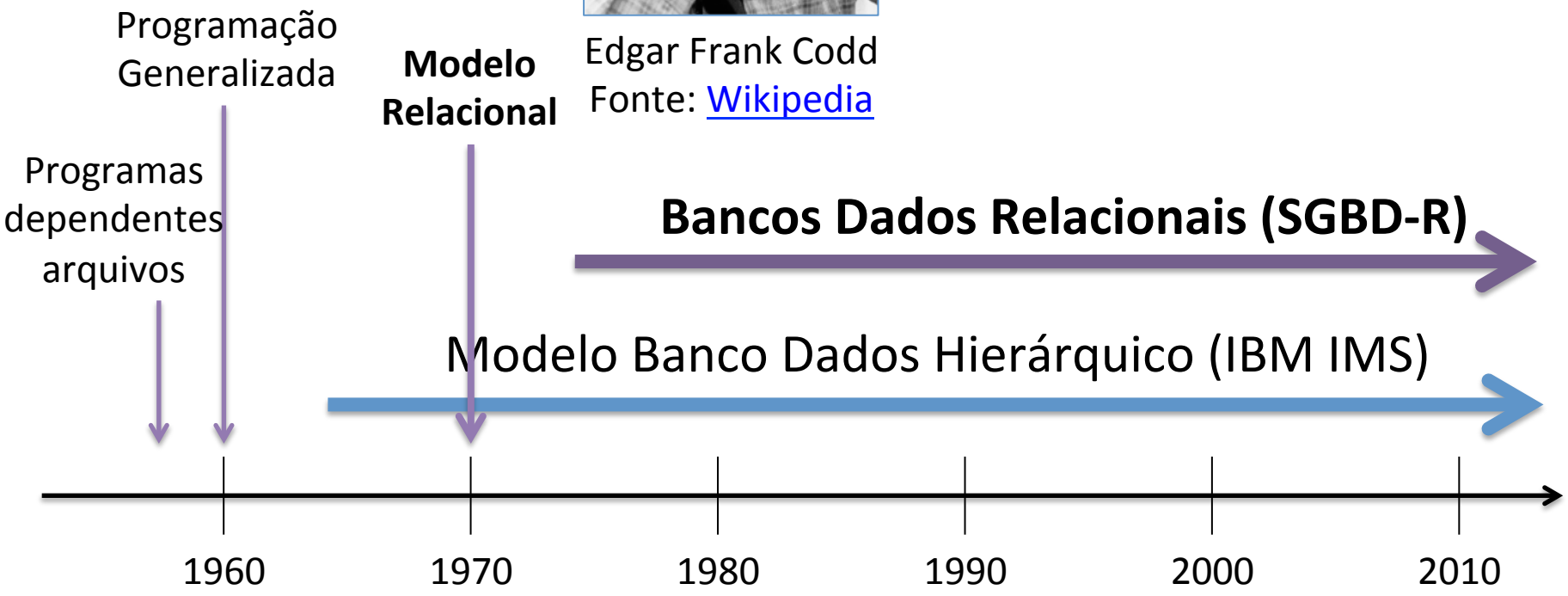
Evolução das Tecnologias de Bancos Dados

ACM Turing Award (1981)

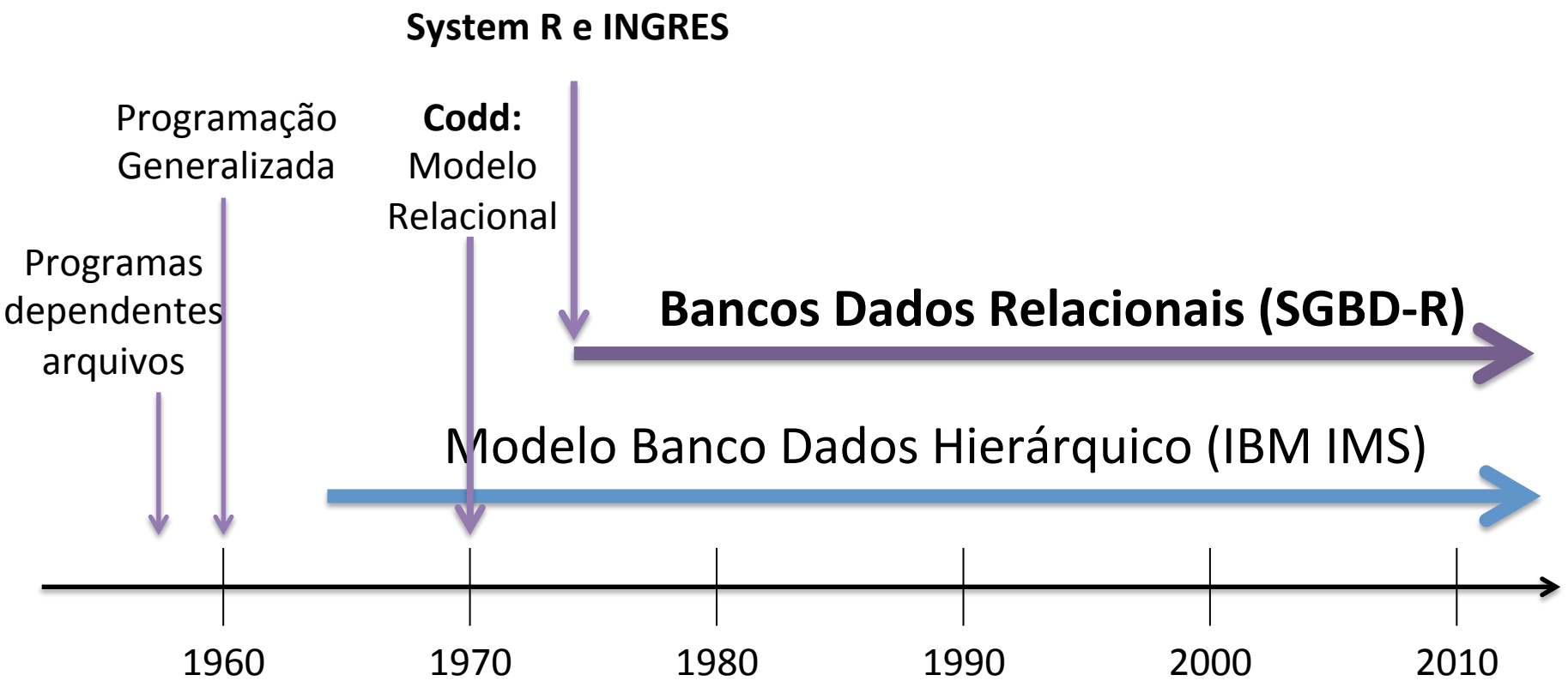


Edgar Frank Codd
Fonte: Wikipedia

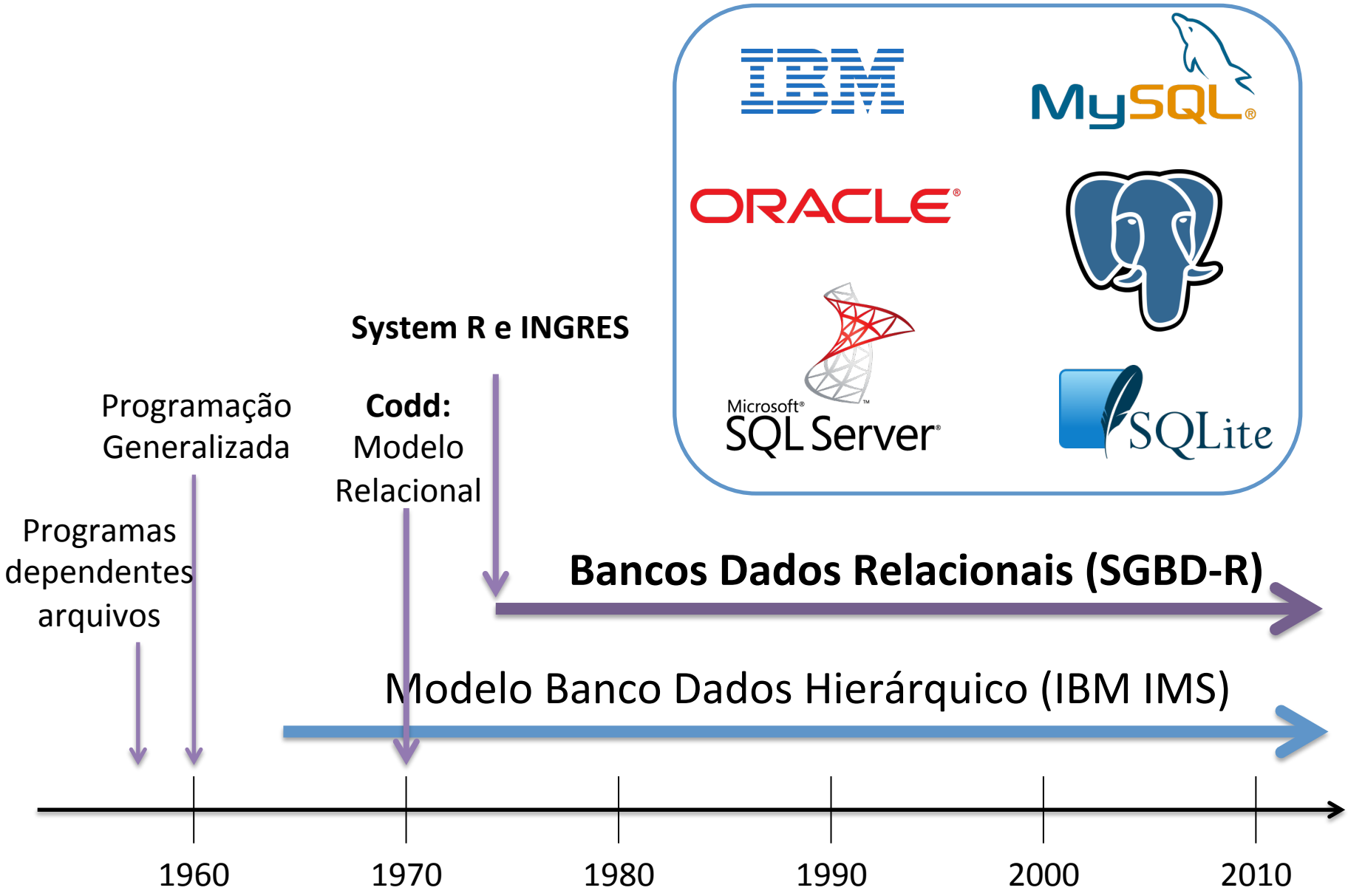
E. F. Codd. 1970. ***A relational model of data for large shared data banks***. Communications of the ACM, v. 13, n. 6, June 1970, pp. 377-387.



Evolução das Tecnologias de Bancos Dados



Evolução das Tecnologias de Bancos Dados



Quais são os principais conceitos em bancos de dados relacionais?

Relação (ou Tabela)

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

Tabela →
Colunas →
Linha →

paises			
id	**nome**	**populacao**	**fronteira**
1	Alemanha	82.000.000	
2	Brasil	190.000.000	
...	...	...	...

Esquema Tabela

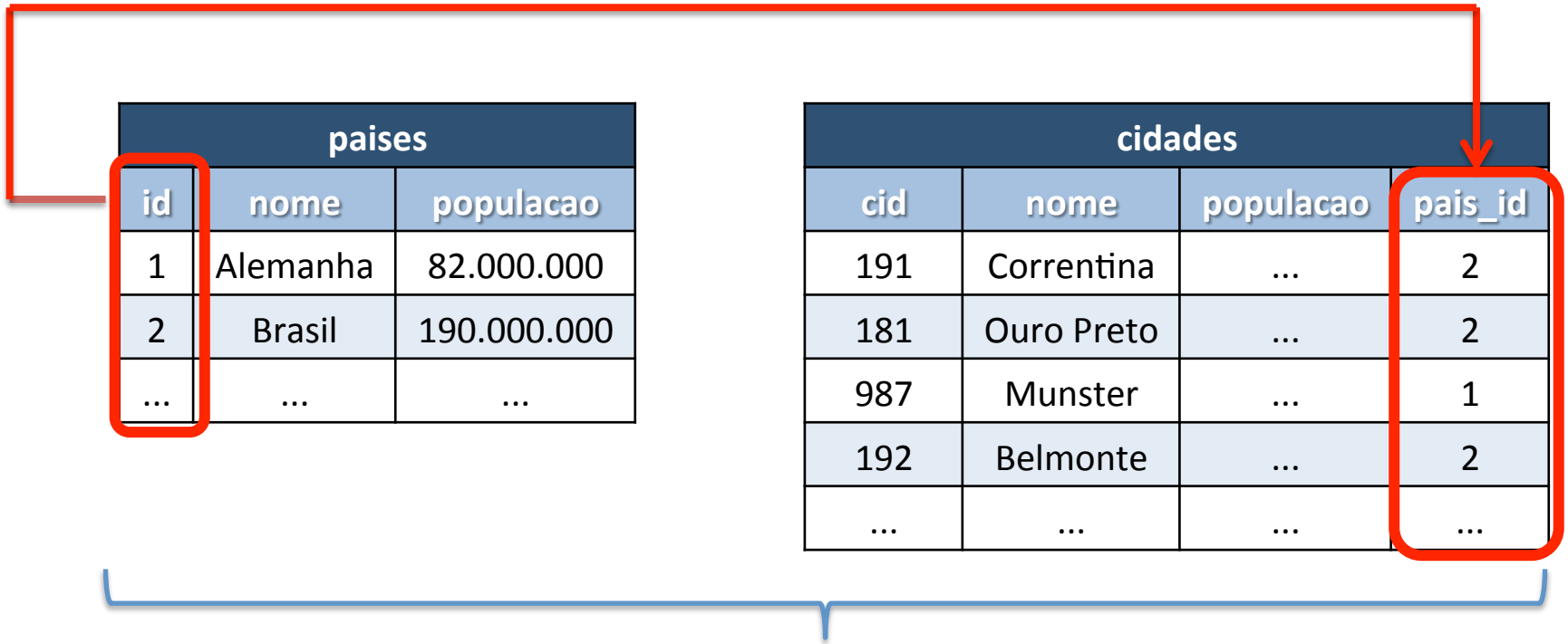
Instância

Modelo Relacional

- Toda tabela (ou relação) possui um nome:
  - Em geral, esse nome é único dentro de um mesmo banco de dados.*
- As colunas de uma tabela são também chamadas de:
  - campos, domínios ou atributos.
- Cada coluna possui um nome e deve ter um tipo de dado associado:
  - Numérico, Cadeia de Caracteres, Data e Hora, Geométrico.
- As linhas também são conhecidas por:
  - tuplas ou registros.

* Conforme veremos mais adiante os SGBD-R podem relaxar esta afirmação com o uso de esquemas (ou *namespaces*)

Relacionamentos entre tabelas



paises

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

cidades

cid	nome	populacao	pais_id
191	Correntina	...	2
181	Ouro Preto	...	2
987	Munster	...	1
192	Belmonte	...	2
...	...	...	...

paises_ x_cidades

pid	p_nome	p_populacao	cid	c_nome	c_populacao
2	Brasil	190000000	191	Correntina	...
2	Brasil	190000000	181	Ouro Preto	...
1	Alemanha	82000000	987	Munster	...
2	Brasil	190000000	192	Belmonte	...
...			...	...	...

Chave Primária (Primary Key)

- Campo ou conjunto de campos cujos valores identificam unicamente cada linha de uma tabela.

Chave Primária

paises		
id	**nome**	**populacao**
1	Alemanha	82.000.000
2	Brasil	190.000.000
...	...	...

Chave Primária

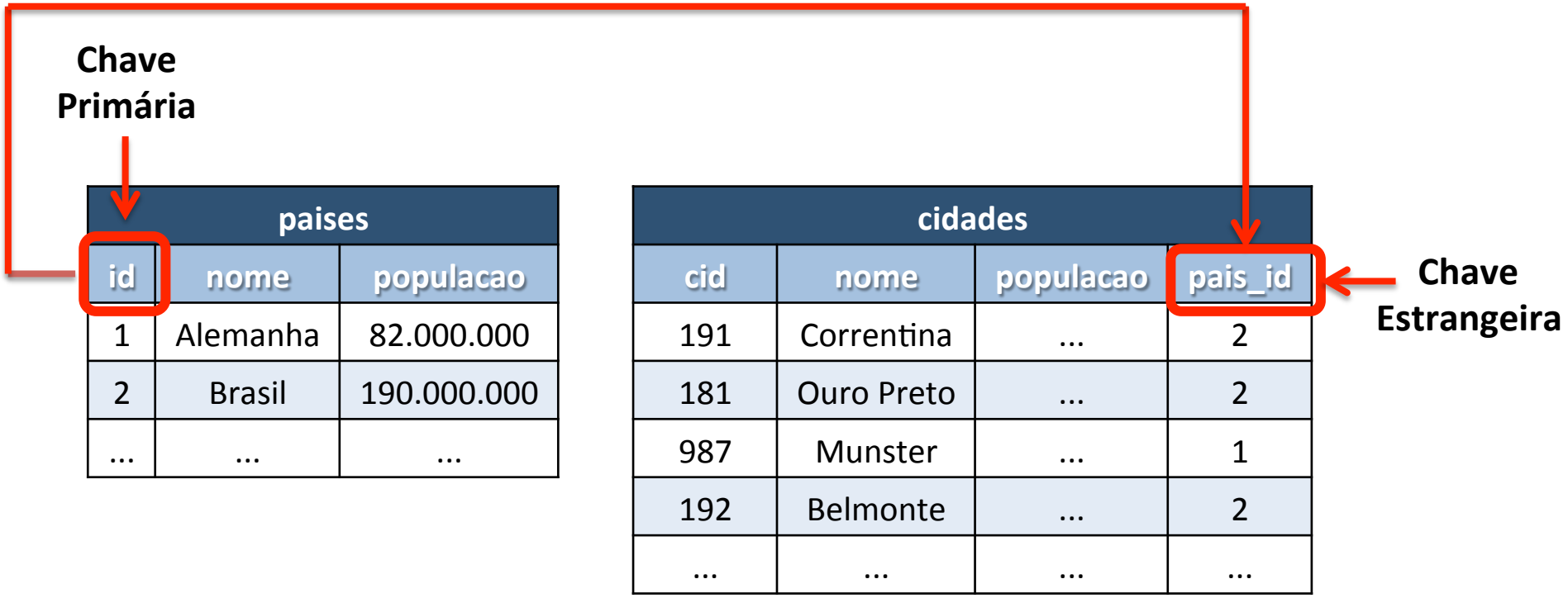
cidades			
cid	**nome**	**populacao**	**pais_id**
191	Correntina	...	2
181	Ouro Preto	...	2
987	Munster	...	1
192	Belmonte	...	2
...	...	...	...

Chave Primária Composta

cliente_telefone		
ncid	**fone**	**tipo**
1	555-7654	residencial
1	345-9876	comercial
2	888-7777	residencial

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)

Restrições de Integridade (Constraints)

- Permitem estabelecer critérios para manutenção da consistência dos dados no banco de dados:
  - **Restrições de domínio:**
    - Especifica os possíveis valores de uma coluna (tipo de dado de uma coluna).
  - **Chave primária.**
  - **Chave estrangeira (ou restrição de integridade referencial):**
    - Uma chave estrangeira especifica uma restrição de integridade referencial entre duas relações R_1 e R_2 sobre um conjunto de colunas.
  - **Valor único:**
    - Garante que o valor em um campo ou conjunto de campos sejam únicos dentro da tabela (semelhante ao conceito de chave primária).
  - **Restrições de nulidade:**
    - especifica se o valor de uma coluna pode ou não ser nulo.
  - **Restrições de valores:**
    - Possibilitam avaliar se o valor de uma ou mais colunas satisfaz uma determinada expressão (fórmula).
- Veremos ao longo do curso de forma prática o que significa cada uma dessas restrições e suas implicações.

Álgebra Relacional

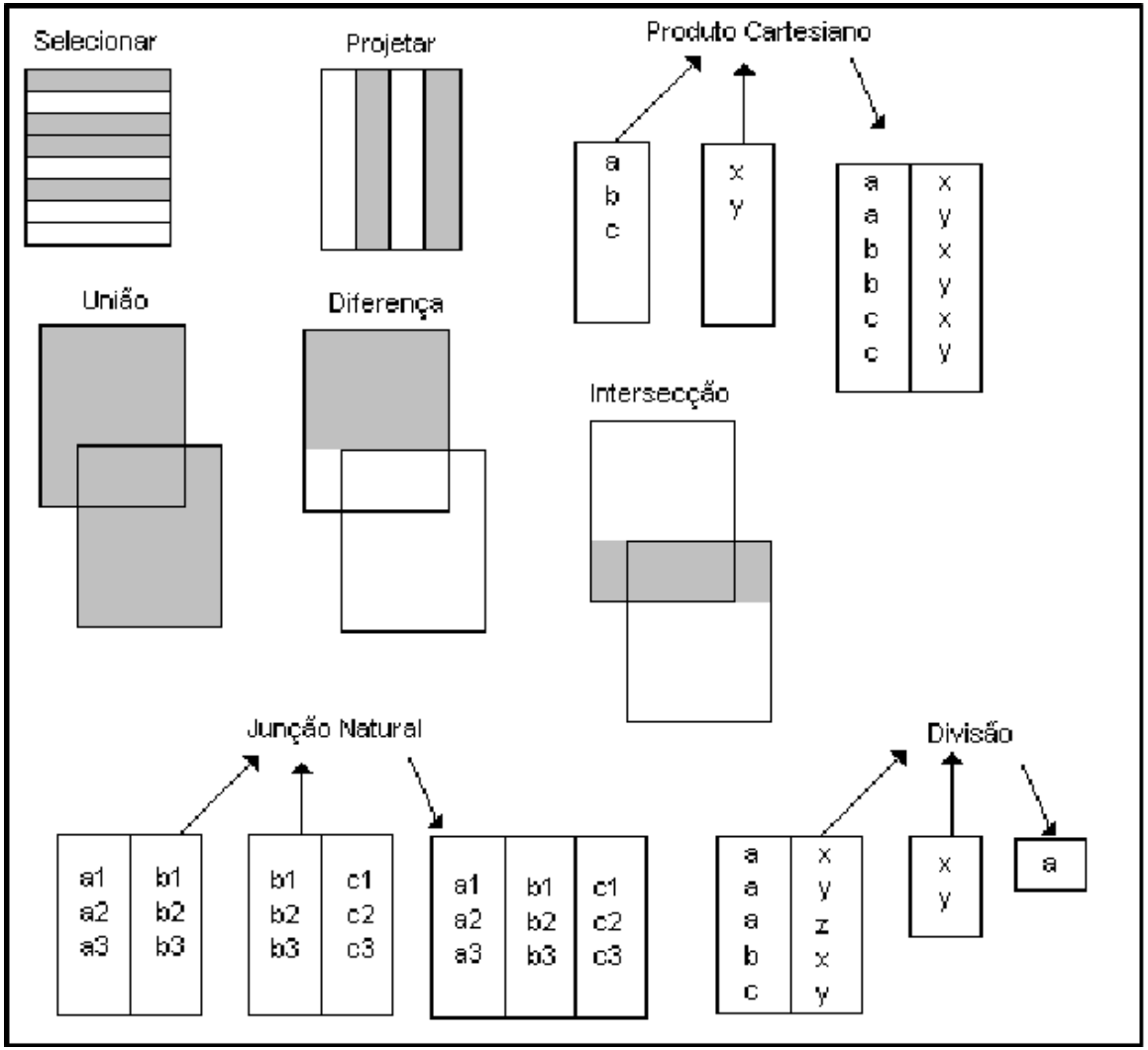
- Linguagem formal de consulta.

- Conjunto de operações que usam uma ou mais relações como entrada e geram uma nova relação de saída:
  - operação (R_1) → R_n
  - operação (R_1, R_2) → R_n

- Operações básicas:
  - Operações unárias: seleção, projeção.
  - Operações binárias: produto cartesiano, junção, interseção, união e diferença.

- Os operadores podem ser combinados de forma a realizar operações mais complexas.

Álgebra Relacional: Operadores



Fonte: C. J. Date (1993)

Álgebra Relacional: Seleção

- Este operador seleciona tuplas (linhas) de uma relação que satisfazem um certo predicado ou condição.

- Exemplo: para a relação “paises”, selecionar as tuplas cuja população seja maior que 100.000.000.

paises		
id	**nome**	**populacao**
1	Alemanha	82.000.000
2	Brasil	190.000.000
...	...	...

Tabela de Entrada

$$\sigma_{populacao \geq 10^8}(paises)$$

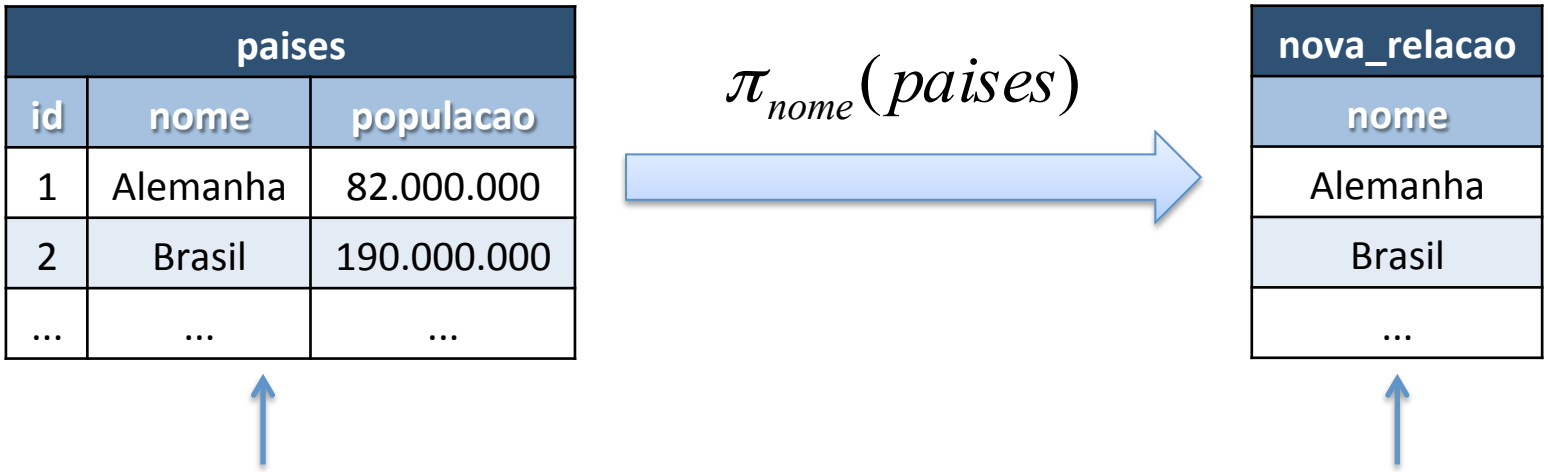
nova_relacao		
id	**nome**	**populacao**
2	Brasil	190.000.000
...	...	...

Tabela de Saída

Álgebra Relacional: Projeção

- Este operador gera uma nova relação contendo apenas as colunas desejadas de uma relação de entrada.

- Exemplo: projetar o atributo nome sobre a relação “paises”.



Álgebra Relacional: Produto Cartesiano

- Este operador gera uma nova relação formada pela combinação de todas as tuplas de duas relações de entrada.

$$(paises) \times (cidades)$$

nova_relacao						
id	**nome**	**populacao**	**cid**	**nome**	**populacao**	**pais_id**
1	Alemanha	82000000	191	Correntina	...	2
1	Alemanha	82000000	181	Ouro Preto	...	2
1	Alemanha	82000000	987	Munster	...	1
1	Alemanha	82000000	192	Belmonte	...	2
2	Brasil	190.000.000	191	Correntina	...	2
2	Brasil	190.000.000	181	Ouro Preto	...	2
2	Brasil	190.000.000	987	Munster	...	1
2	Brasil	190.000.000	192	Belmonte	...	2
...	...	...	...	...	...	...

Álgebra Relacional: Junção (Join)

- Produto cartesiano seguido de uma seleção.

$$(paises)\theta(cidades) \Leftrightarrow \sigma_{paises.id=cidades.pais\_id}(paises \times cidades)$$

nova_relacao						
id	**nome**	**populacao**	**cid**	**nome**	**populacao**	**pais_id**
1	Alemanha	82000000	987	Munster	...	1
2	Brasil	190.000.000	191	Correntina	...	2
2	Brasil	190.000.000	181	Ouro Preto	...	2
2	Brasil	190.000.000	192	Belmonte	...	2
...	...	...	...	...	...	...

Linguagem de Consulta: SQL

- O modelo relacional (Codd, 1970) é a base para linguagens de alto nível:
  - Álgebra/Cálculo Relacional → Linguagem Declarativa → ISO/SQL (Structured Query Language)

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

Definição Dados

paises		
id	nome	populacao

Manipulação Dados

```
INSERT INTO paises
      VALUES (1, 'Alemanha', 82000000)

INSERT INTO paises
      VALUES (2, 'Brasil', 190000000)
```

Manipulação Dados

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

Linguagem de Consulta: SQL

- O modelo relacional (Codd, 1970) é a base para linguagens de alto nível:
  - Álgebra/Cálculo Relacional → Linguagem Declarativa → ISO/SQL (Structured Query Language)

paises		
id	**nome**	**populacao**
1	Alemanha	82.000.000
2	Brasil	190.000.000
...	...	...

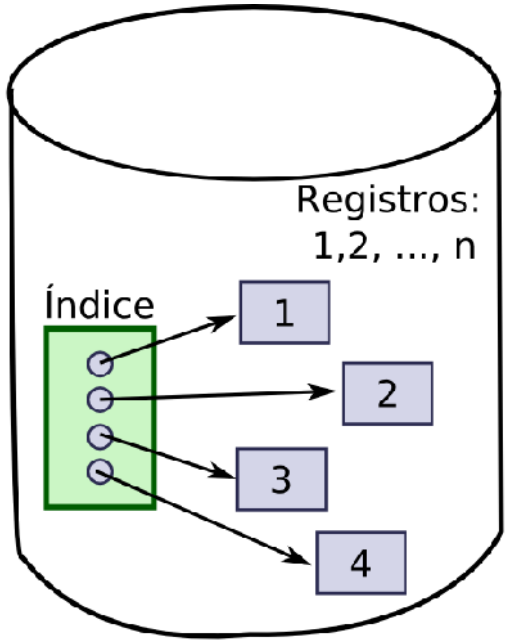

```
SELECT nome
 FROM paises
WHERE populacao > 80000000
```

Nota: stored procedures ou procedural languages: PL/SQL, T-SQL, PL/pgSQL

Métodos de Acesso (Indexação)

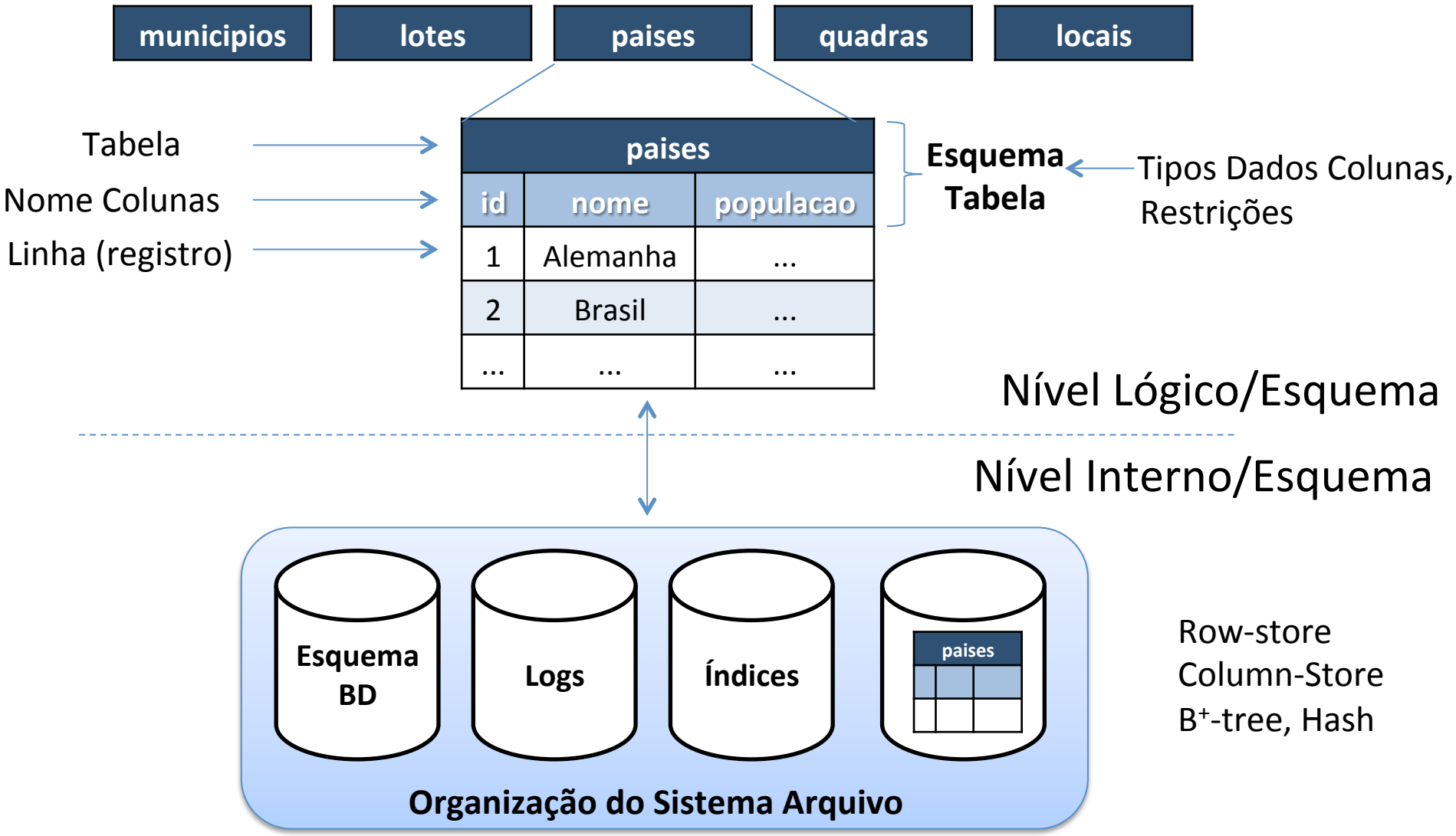
- Problema: Como processar de forma eficiente as consultas?
  - Através do uso de estruturas de dados conhecidas como Índices ou Métodos de Acesso;

- Os índices reduzem o conjunto de objetos a serem verificados durante o processamento das consultas:
  - Normalmente, uma consulta envolve apenas uma pequena parcela do banco de dados;
  - Neste caso, percorrer todo o banco pode ser bastante ineficiente;
  - Portanto, um plano de execução eficiente para a consulta tipicamente considera a existência de índices.



Registros de um arquivo e o índice associado a este arquivo

Independência Física dos Dados



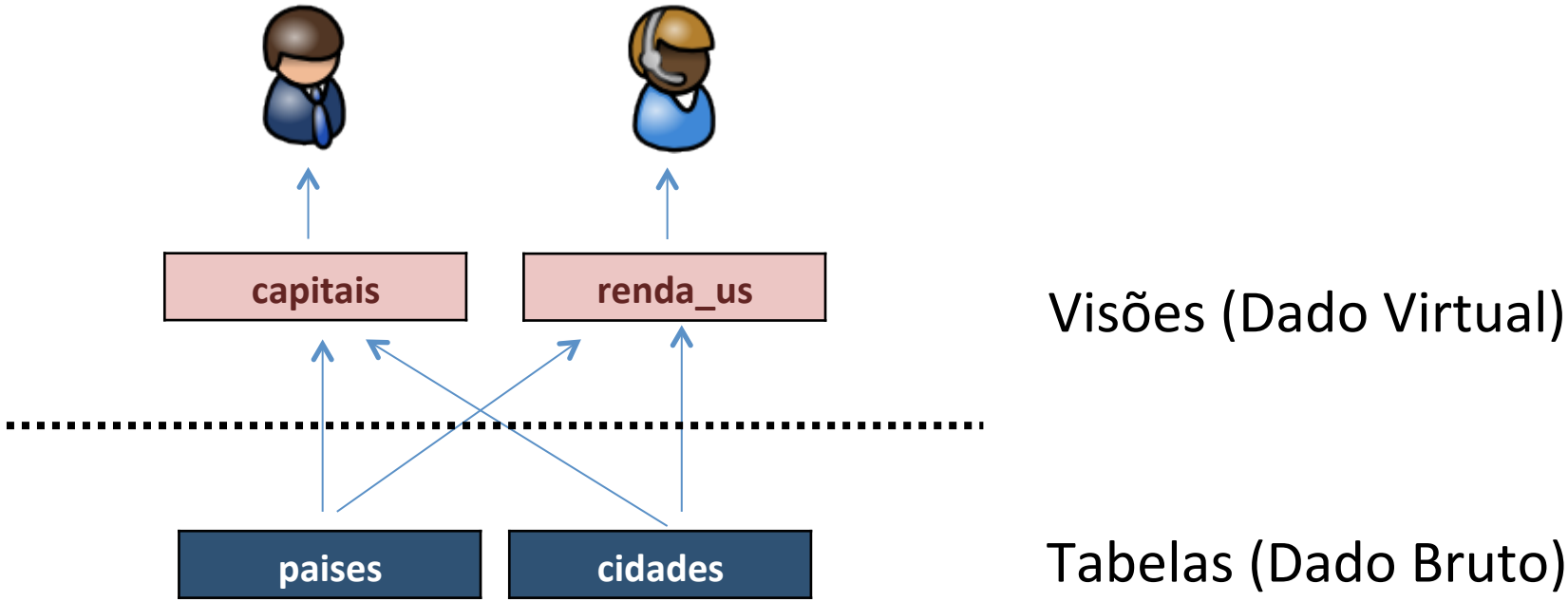
Fonte: Adaptado de Gray (1996)

Como a independência física é alcançada?

- Esquema do Banco de Dados:
  - Uma característica fundamental de um SGBD-R é que ele não contém apenas os dados brutos sobre o domínio de interesse;
  - Todo SGBD-R mantém a definição ou descrição da estrutura do banco de dados (*self-describing*);
  - Essas informações são mantidas no catálogo do sistema (ou dicionário do sistema) e são denominadas de metadados do banco de dados.
  - Na prática os SGBD-R armazenam essas informações de definição em tabelas do próprio sistema (tabelas de metadado ou tabelas do catálogo).
- O modelo de dados relacional fornece para as aplicações uma abstração independente da representação física dos dados.

Visões (Views)

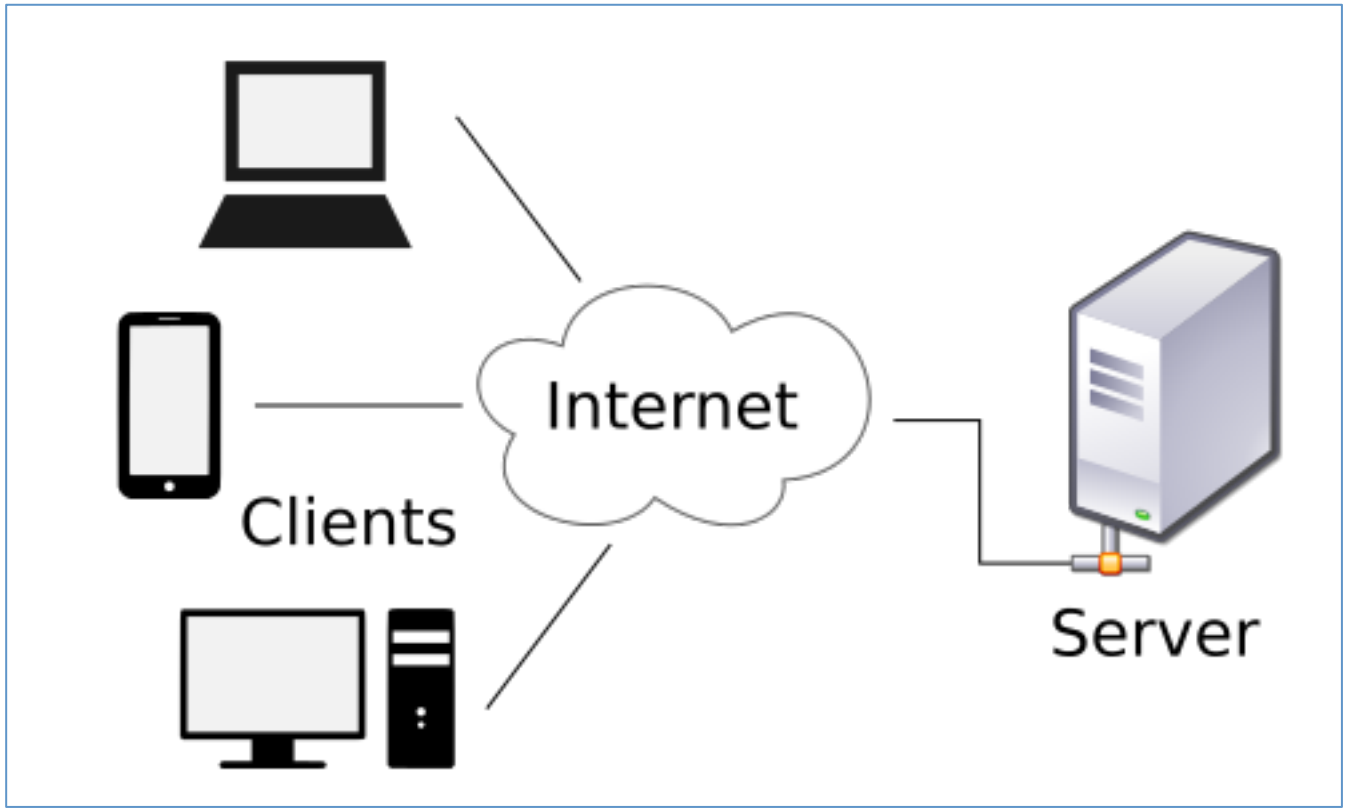
- Muitas vezes pode ser necessário fornecer diferentes perspectivas do banco de dados dependendo do usuário. Uma visão (ou view) pode ser:
  - um subconjunto dos dados do banco de dados
  - pode conter dados derivados do banco de dados



Arquiteturas de SGBD-R

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

Arquiteturas de SGBD-R: Cliente Servidor

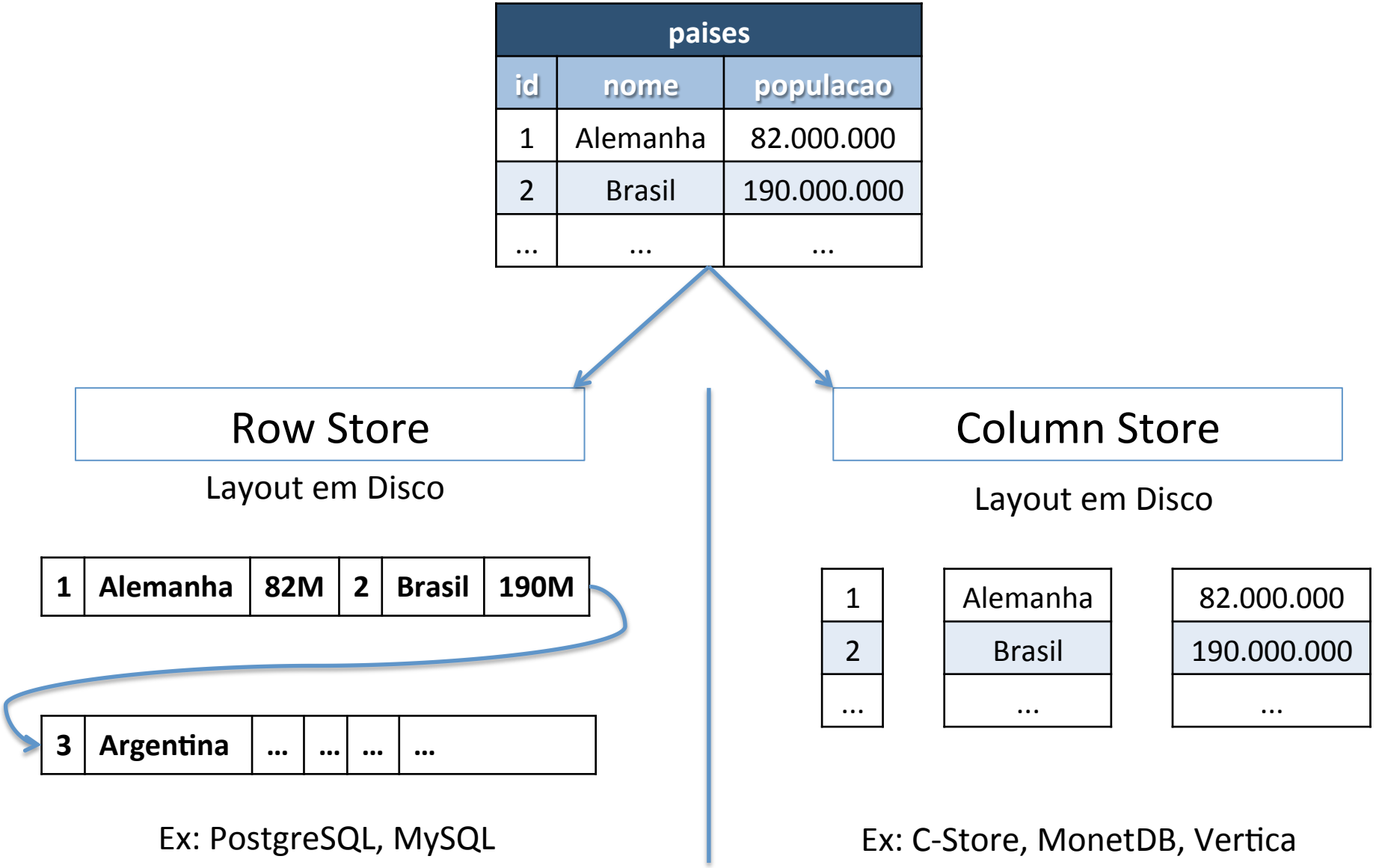


Fonte: Wikipedia

Arquiteturas de SGBD-R: Embedded

```c
#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;
}
```

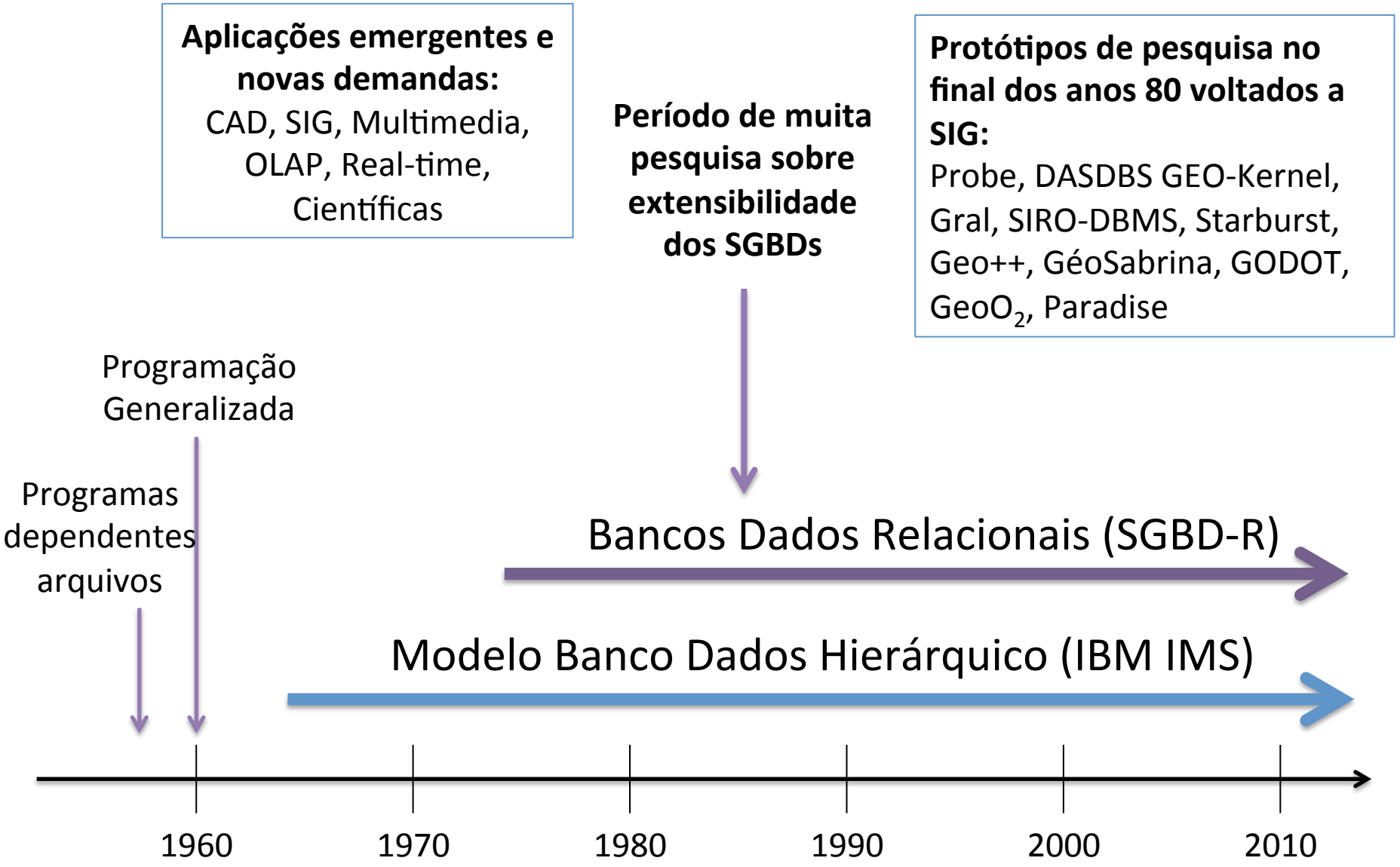
Arquiteturas de SGBD-R: row x column store



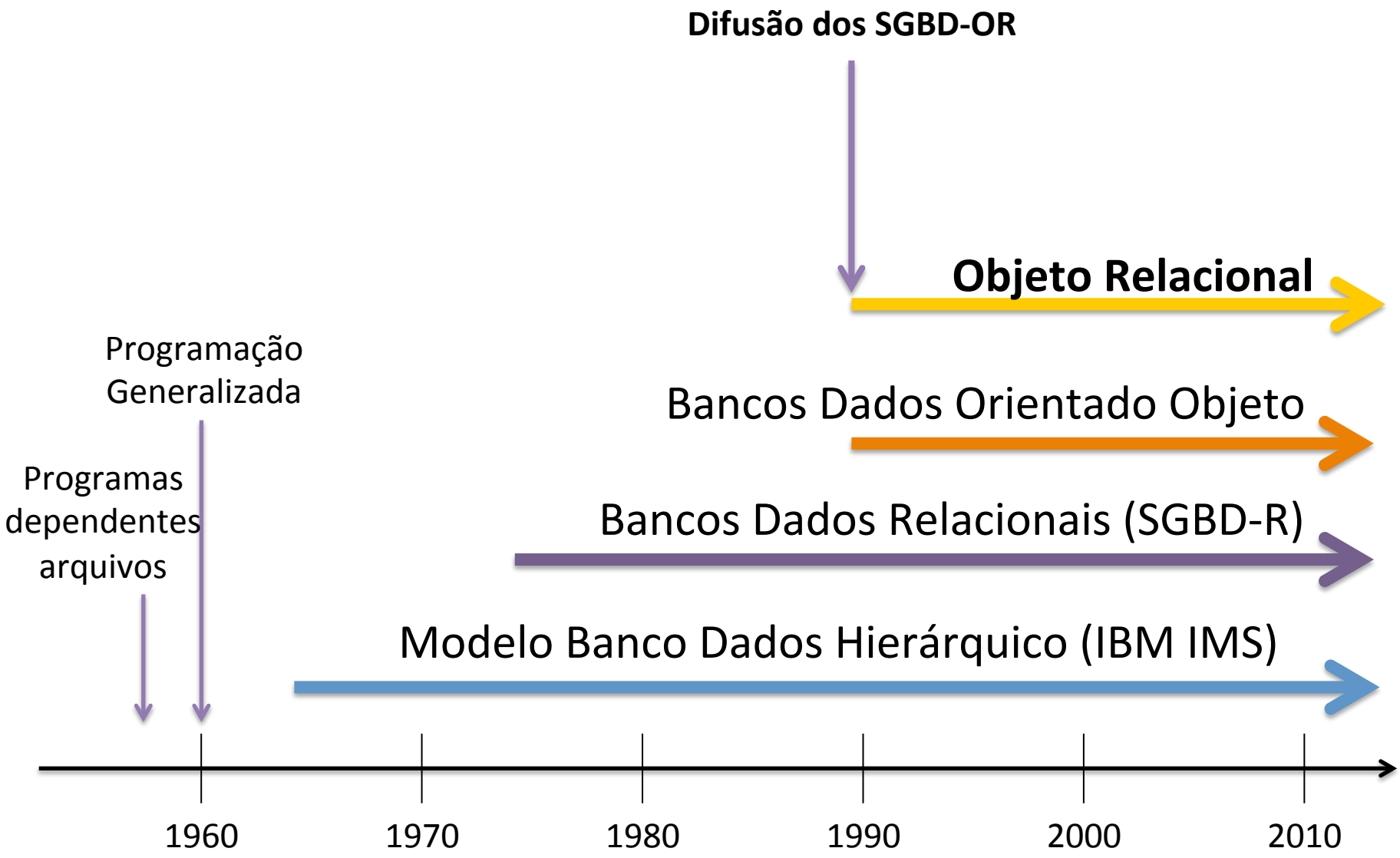
Outros Conceitos Importantes

- Projeto de bancos de dados:
  - Modelo Entidade-Relacionamento (ER).
- 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



SGBD-OR: User Defined Types (UDT)

```
CREATE TYPE geo_point AS
(
  x    REAL,
  y    REAL,
  srid INTEGER
);
```

SGBD-OR: User Defined Types (UDT)

```
CREATE TABLE sedes_municipais
(
  id INTEGER PRIMARY KEY,
  location GEO_POINT
);
```

```
INSERT INTO sedes_municipais
     VALUES (1, '(1, 2, 4326)'::GEO_POINT);
```

SGBD-OR: User Defined Functions (UDF)

- Possibilita criar ou estender a álgebra de um determinado tipo de dado.

```
CREATE OR REPLACE FUNCTION less_than(first GEO_POINT, second GEO_POINT)
RETURNS REAL
AS $$
BEGIN
  IF(first.x < second.x)
  THEN
    RETURN TRUE;
  END IF;

  IF(first.x > second.x)
  THEN
    RETURN FALSE;
  END IF;
  ...
  RETURN FALSE;
END;
$$
LANGUAGE plpgsql;
```

SGBD-OR: User Defined Functions (UDF)

- Possibilita criar ou estender a álgebra de um determinado tipo de dado.

```
CREATE OR REPLACE FUNCTION distance(first GEO_POINT, second GEO_POINT)
RETURNS REAL
AS $$
DECLARE
  dx REAL;
  dy REAL;
BEGIN
  dx = (first.x - second.x) * (first.x - second.x);

  dy = (first.y - second.y) * (first.y - second.y);

  RETURN sqrt(dx + dy);
END;
$$
LANGUAGE plpgsql;
```

SGBD-OR: User Defined Functions (UDF)

- UDFs passam a fazer parte da linguagem de consulta do SGBD:

```
SELECT less_than('(1, 2, 4326)'::GEO_POINT, '(10, 20, 4326)'::GEO_POINT);

SELECT less_than('(1, 2, 4326)'::GEO_POINT, '(-1, 2, 4326)'::GEO_POINT);

SELECT distance('(1, 2, 4326)'::GEO_POINT, '(10, 20, 4326)'::GEO_POINT);
```

SGBD-OR: Sobrecarga de Operadores

```
CREATE OPERATOR <
(
  leftarg = GEO_POINT,
  rightarg = GEO_POINT,
  procedure = less_than,
  commutator = >,
  negator = >=
);
```

```
SELECT '(1, 2, 4326)'::GEO_POINT < '(10, 2, 4326)'::GEO_POINT;
```

SGBD-OR: User Defined Access Methods

B-tree

Operation	Strategy Number
less than	1
less than or equal	2
equal	3
greater than or equal	4
greater than	5

Hash

Operation	Strategy Number
equal	1

GiST – Rtree 2D

Operation	Strategy Number
strictly left of	1
does not extend to right of	2
overlaps	3
does not extend to left of	4
strictly right of	5
same	6
contains	7
contained by	8
does not extend above	9
strictly below	10
strictly above	11
does not extend below	12

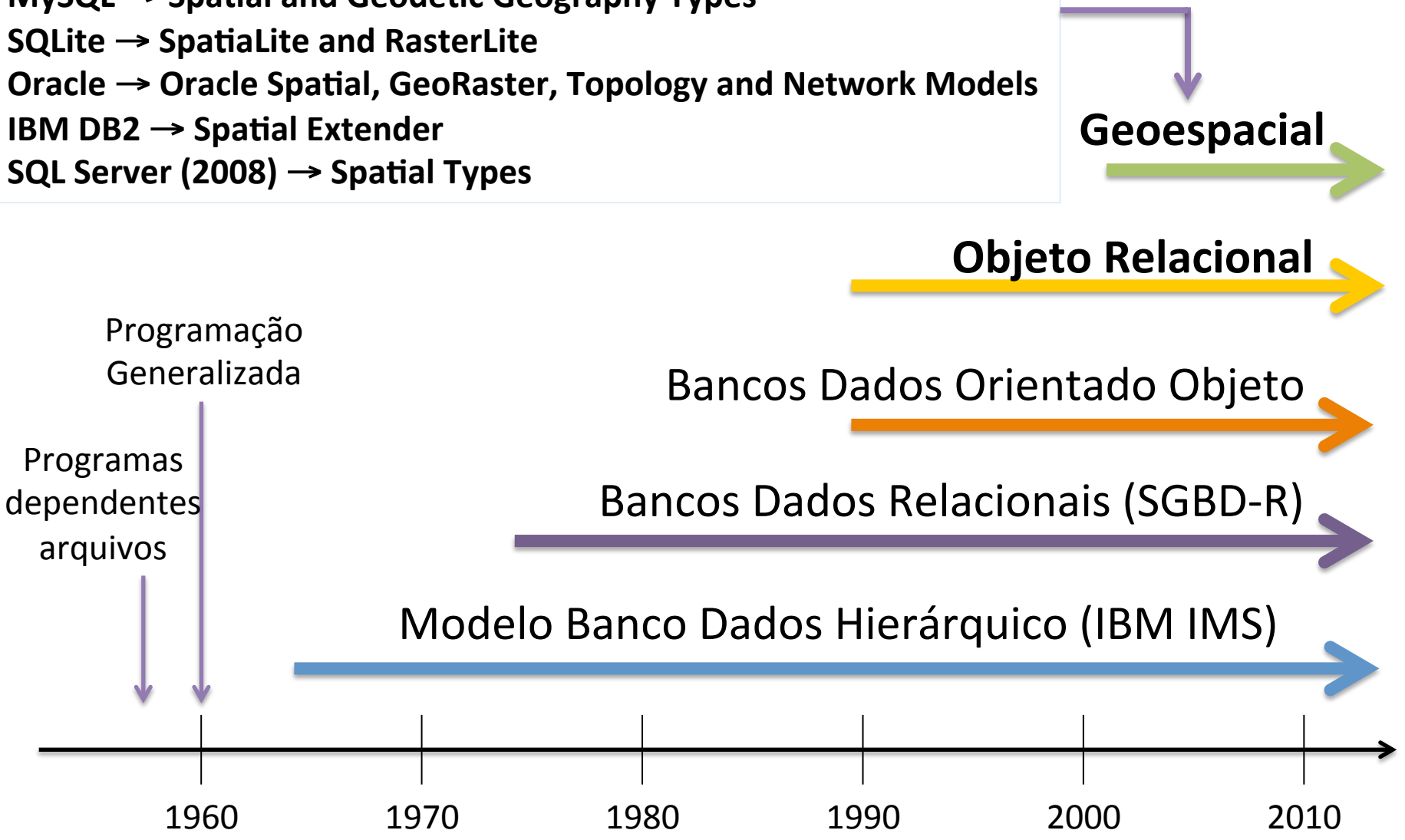
SGBD-OR: UDTs mais Complexos

```
CREATE TYPE Geometry
(
internallength = variable,
input = geometry_in,
output = geometry_out,
send = geometry_send,
receive = geometry_recv,
typmod_in = geometry_typmod_in,
typmod_out = geometry_typmod_out,
delimiter = ':',
alignment = double,
analyze = geometry_analyze,
storage = main);
```

```
CREATE OR REPLACE FUNCTION _ST_Touches(geom1 geometry, geom2 geometry)
        RETURNS boolean
        AS '$libdir/postgis-2.1','touches'
        LANGUAGE 'c' IMMUTABLE STRICT
        COST 100;
...
```

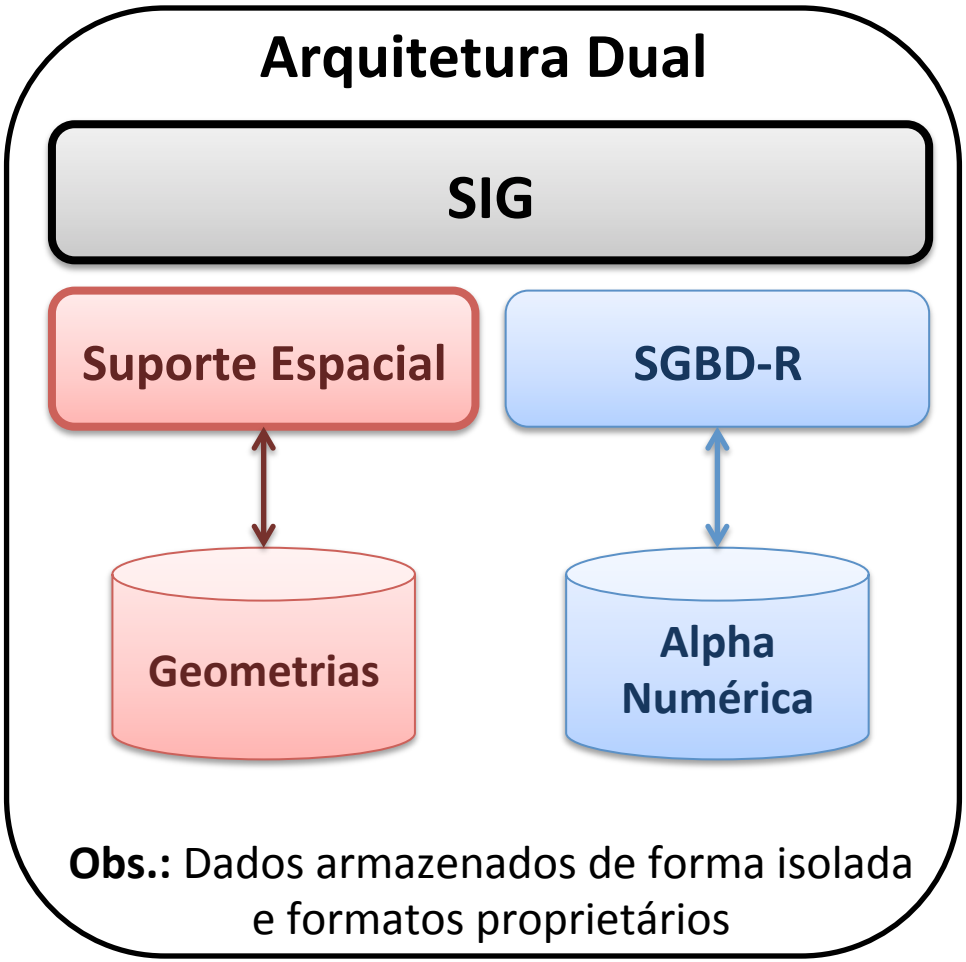
Evolução das Tecnologias de Bancos Dados

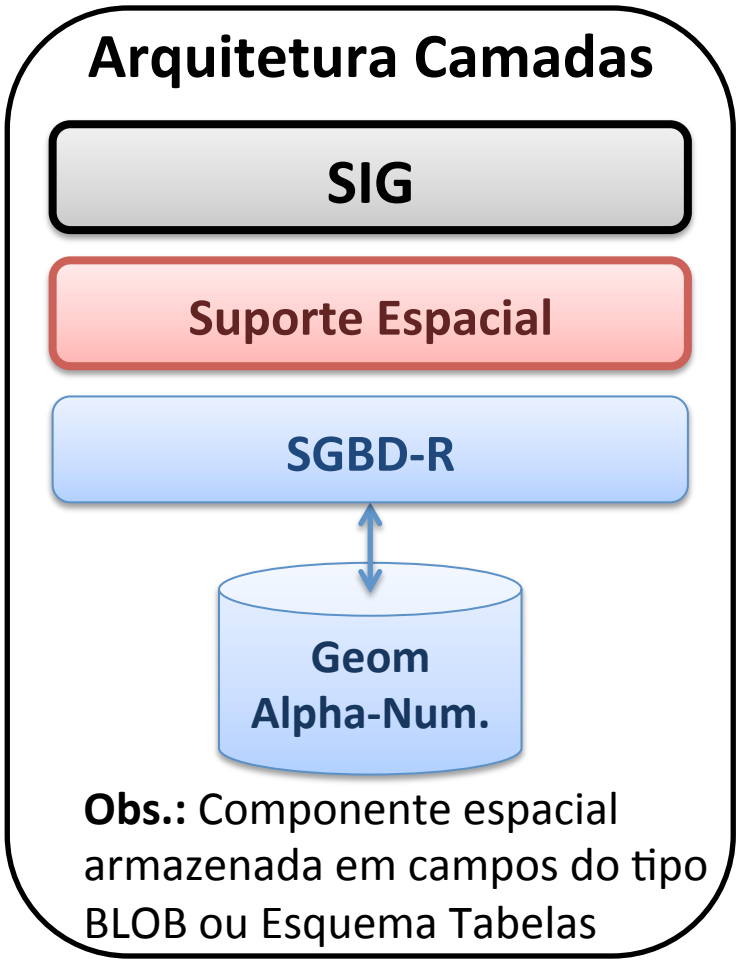
PostgreSQL → PostGIS
MySQL → Spatial and Geodetic Geography Types
SQLite → SpatiaLite and RasterLite
Oracle → Oracle Spatial, GeoRaster, Topology and Network Models
IBM DB2 → Spatial Extender
SQL Server (2008) → Spatial Types



SIG e SGBD-R

- Como era a integração SIG e SGBD-R antes da inclusão do suporte espacial?





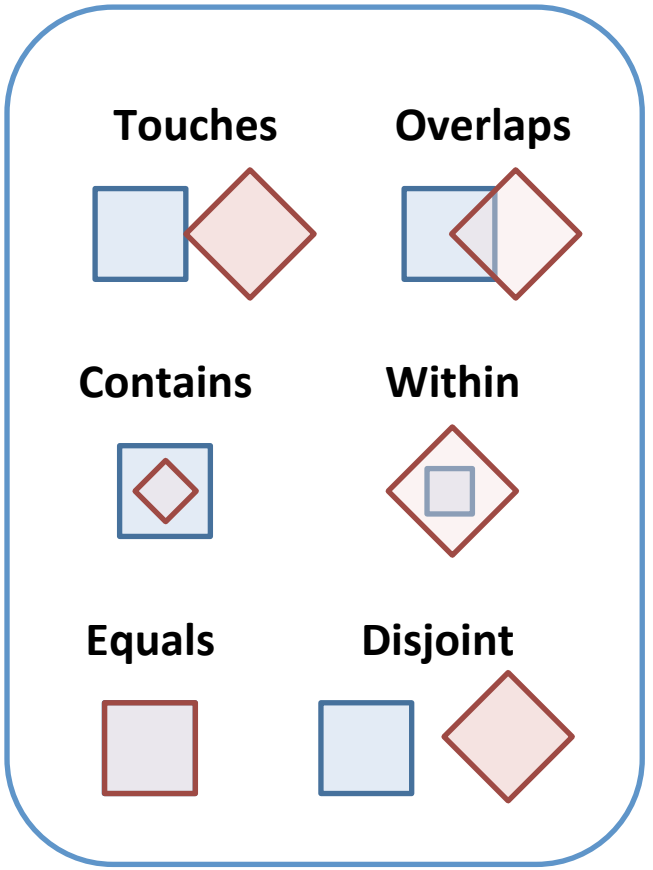
SIG e SGBD-R: Como passou a ser esta integração?

- Arquitetura Integrada: Tipos de Dados Geoespaciais
- Padronização: OGC Simple Features e ISO/SQL-MM Spatial

Tabelas com feições: geometrias vetoriais

paises			
id	**nome**	**populacao**	**fronteira**
1	Alemanha	82.000.000	
2	Brasil	190.000.000	
...	...	...	...

Operações espaciais



O que mais existe nesta integração entre SGBD-R e Dados Geográficos?

- Índices Espaciais: árvores-R, Quadtrees, Fixed-Grid.
- Armazenamento dados matriciais.
- Armazenamento baseado em modelos topológicos.
- Redes espaciais: roteamento, análise de fluxo.

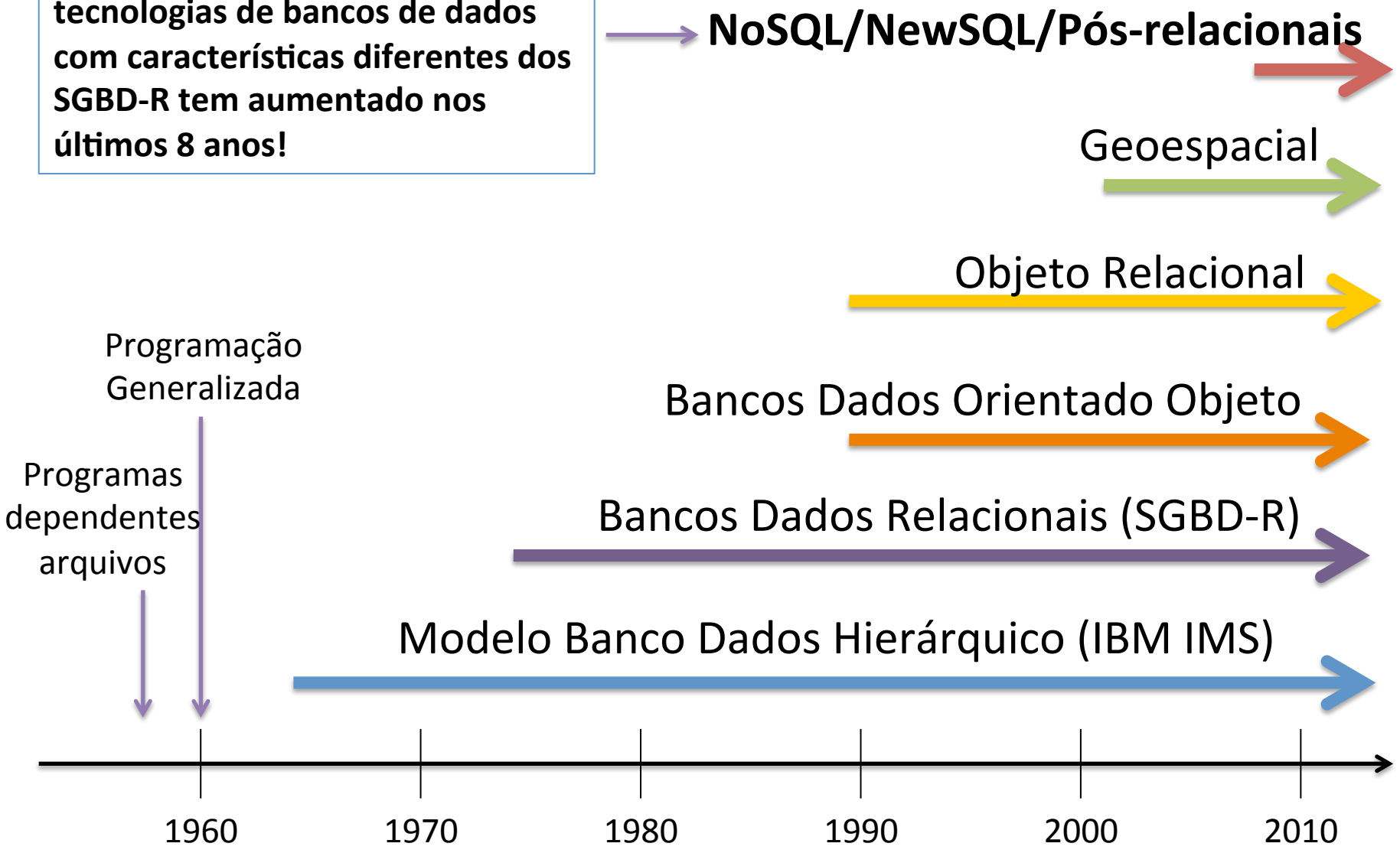
Bancos de Dados x Information Retrieval

- Bancos de dados → Informações estruturadas
  - Esquemas
  - SQL

- IR → mais voltado para informações não estruturadas como processamento de documentos e texto livre.
  - Web Search Engines
  - SVM (Support Vector Machines)

Evolução das Tecnologias de Bancos Dados

Interessante: o número de tecnologias de bancos de dados com características diferentes dos SGBD-R tem aumentado nos últimos 8 anos!



O “cardápio” de opções aumentou?

- *Sistemas Não-Relacionais* ou *Not Only SQL* ou *Pós-relacionais*:
  - http://nosql-database.org/
  - https://en.wikipedia.org/wiki/NoSQL

- Diferentes modelos de dados:
  - Document Oriented: MongoDB, CouchDB;
  - Column Stores: Cassandra;
  - Graph Databases: OrientDB, Neo4J;
  - Array Databases: SciDB, Rasdaman.

- Nem todos são baseados no paradigma de transações ACID.

- Escalabilidade: Horizontal x Vertical

Suporte Espacial em NoSQL

- MongoDB
- CouchDB
- Apache Solr
- Neo4J Spatial

Referências

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, vl. 1, n. 1. March 1976, pp. 9-36.

- GRAY, J. ***Evolution of Data Management***. IEEE Computer 29(10): 38-46, 1996.

- Vijlbrief, T., and P. van Oosterom. ***The GEO++ System: An Extensible GIS***. Proc. 5th Intl. Symposium on Spatial Data Handling, Charleston, South Carolina, 1992, 40-50.

Especificações e Padrões

- OGC. ***OpenGIS Implementation Specification for Geographic information - Simple feature access - Part 1: Common architecture***. Available at: http://www.opengeospatial.org. Access: October, 2012.

- OGC. ***OpenGIS Implementation Specification for Geographic information - Simple feature access - Part 2: SQL option***. Available at: http://www.opengeospatial.org. Access: October, 2012.

- ISO. ***SQL Multimedia and Application Packages – Part 3: Spatial***.

Slides

- NAUGHTON, J. F. ***DBMS Research: First 50 Years, Next 50 Years***. Kynote speaker' slides at ICDE 2010. Disponível em: http://pages.cs.wisc.edu/~naughton/naughtonicde.pptx. Acesso: Abril de 2013.