BANCOS DE DADOS GEOGRÁFICOS

Structured Query Language (SQL)

MAJOR IVANILDO BARBOSA

SEÇÃO DE ENSINO DE ENGENHARIA CARTOGRÁFICA

Contents

Conceitos Básicos

Definição de elementos

Consultas em SQL

Álgebra Relacional

Conceitos Básicos

Conceitos Básicos

- A linguagem SQL surgiu em meados da década de 70 (IBM)
- Foco em desenvolver uma linguagem que se adaptasse ao modelo relacional.
- O ANSI (*American National Standarts Institute*) padronizou as implementações do SQL.
- Possui comandos capazes de
  - criar elementos como bancos de dados, tabelas e índices.
  - manipular as estruturas dos elementos criados: alterar e renomear;
  - inserir e remover registros de uma tabela;
  - realizar consultas personalizadas;
  - remover dados e destruir elementos.

Conceitos Básicos

Cada atributo é definido juntamente com o respectivo tipo;

- Bit, Bit Varying
- Date, Time, Timestamp
- Decimal, Real, Double Precision, Float, Smallint, Integer, Interval
- Character, Character Varying, National Character, National Character Varying

Diferentes SGBD possuem tipos próprios, de processamento mais eficiente, ou variações dos tipos ANSI SQL;

Exemplo: No PostgreSQL, existem os tipos *text*, *boolean*, *json*, *macaddr*, *serial*, *uuid* e *XML*.

Data type	Access	SQLServer	Oracle	MySQL	PostgreSQL
boolean	Yes/No	Bit	Byte	N/A	Boolean
integer	Number (integer)	Int	Number	Int Integer	Int Integer
float	Number (single)	Float Real	Number	Float	Numeric
currency	Currency	Money	N/A	N/A	Money
string (fixed)	N/A	Char	Char	Char	Char
string (variable)	Text (<256) Memo (65k+)	Varchar	Varchar Varchar2	Varchar	Varchar
binary object	OLE Object Memo	Binary (fixed up to 8K) Varbinary (<8K) Image (<2GB)	Long Raw	Blob Text	Binary Varbinary

Fonte: www.w3schools.com

Conceitos Básicos

São previstas restrições (*constraints*), que atuam como modificadores das declarações ordinárias.

NOT NULL – a coluna não permite valores NULL, i.e., vazios;

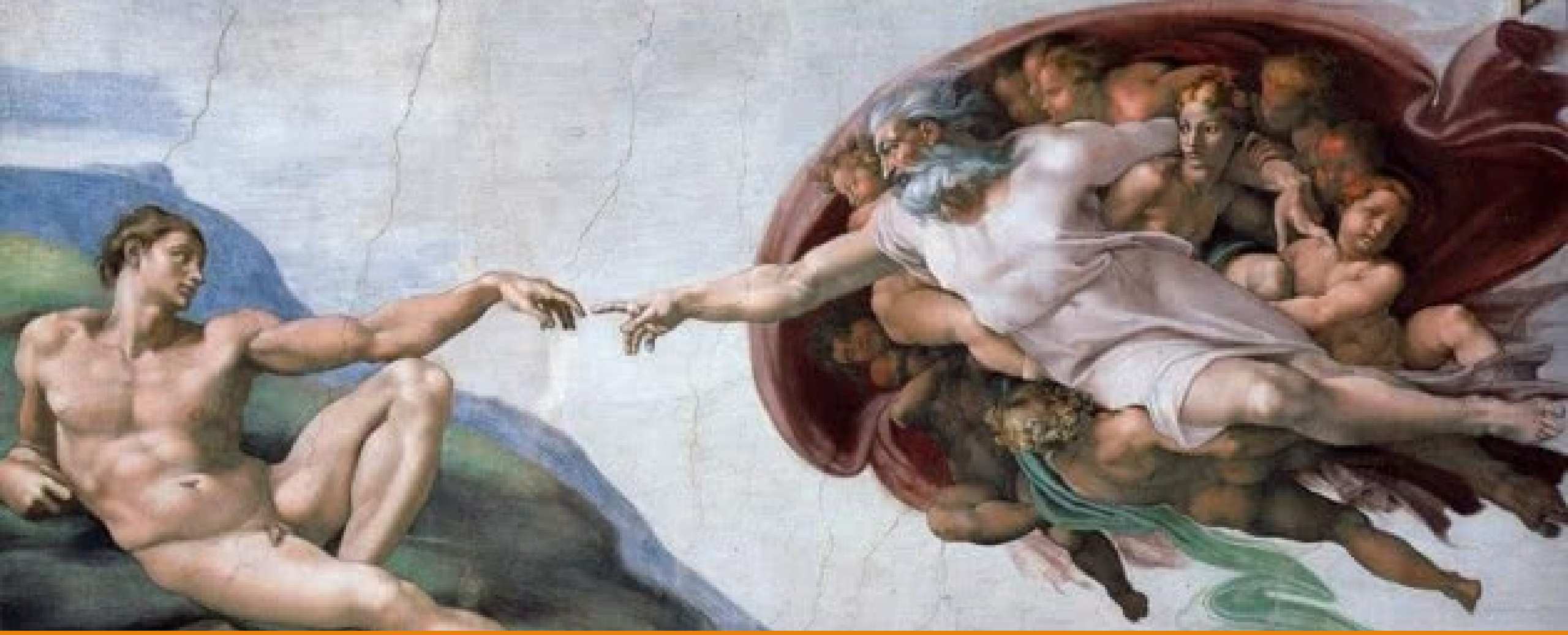
UNIQUE – assegura que uma coluna não receba dois registros com valores iguais.

PRIMARY KEY – define que os valores da coluna servirão como referência unívoca dos registros, herdando as características NOT NULL e UNIQUE.

FOREIGN KEY – Assegura a integridade referencial dos dados de uma tabela que devem combinar com os dados de uma coluna de uma tabela diferente.

CHECK – Garante que o valor inserido em uma coluna atende a uma determinada condição

DEFAULT – especifica um valor padrão para a coluna



Definição de elementos

Definição de elementos

```
CREATE DATABASE my_db;

USE my_db;
```

```
CREATE TABLE table_name
(
column_name1 data_type(size),
column_name2 data_type(size),
column_name3 data_type(size),
....
);
USE table_name;
```

```
CREATE TABLE table_name
(
column_name1 data_type(size) constraint_name,
column_name2 data_type(size) constraint_name,
column_name3 data_type(size) constraint_name,
....
);
USE table_name;
```

Definição de elementos

A cláusula DROP é empregada para excluir esquemas, tabelas, domínios, índices, tabelas virtuais, restrições, etc. Pode ter dois comportamentos:

· CASCADE: exclui também todos os objetos relacionados ao objeto excluído

· RESTRICT: o objeto só e excluído se nao há nenhum outro objeto relacionado a ele. (opcao default)

```
DROP TABLE table_name;
DROP DATABASE database_name;
```

Definição de elementos

Após a criação da tabela, é possível alterar a sua estrutura

- ALTER TABLE aluno ADD COLUMN endereco VARCHAR(200)
- ALTER TABLE aluno ALTER COLUMN endereco VARCHAR(400)
- ALTER TABLE aluno ALTER COLUMN endereco TYPE VARCHAR(400)
- ALTER TABLE aluno RENAME COLUMN endereco TO endereco2
- ALTER TABLE aluno DROP COLUMN endereco2
- ALTER TABLE aluno RENAME TO aluno2

A tabela também pode ser renomeada como

```
RENAME TABLE old_table TO backup_table, new_table TO old_table;
```

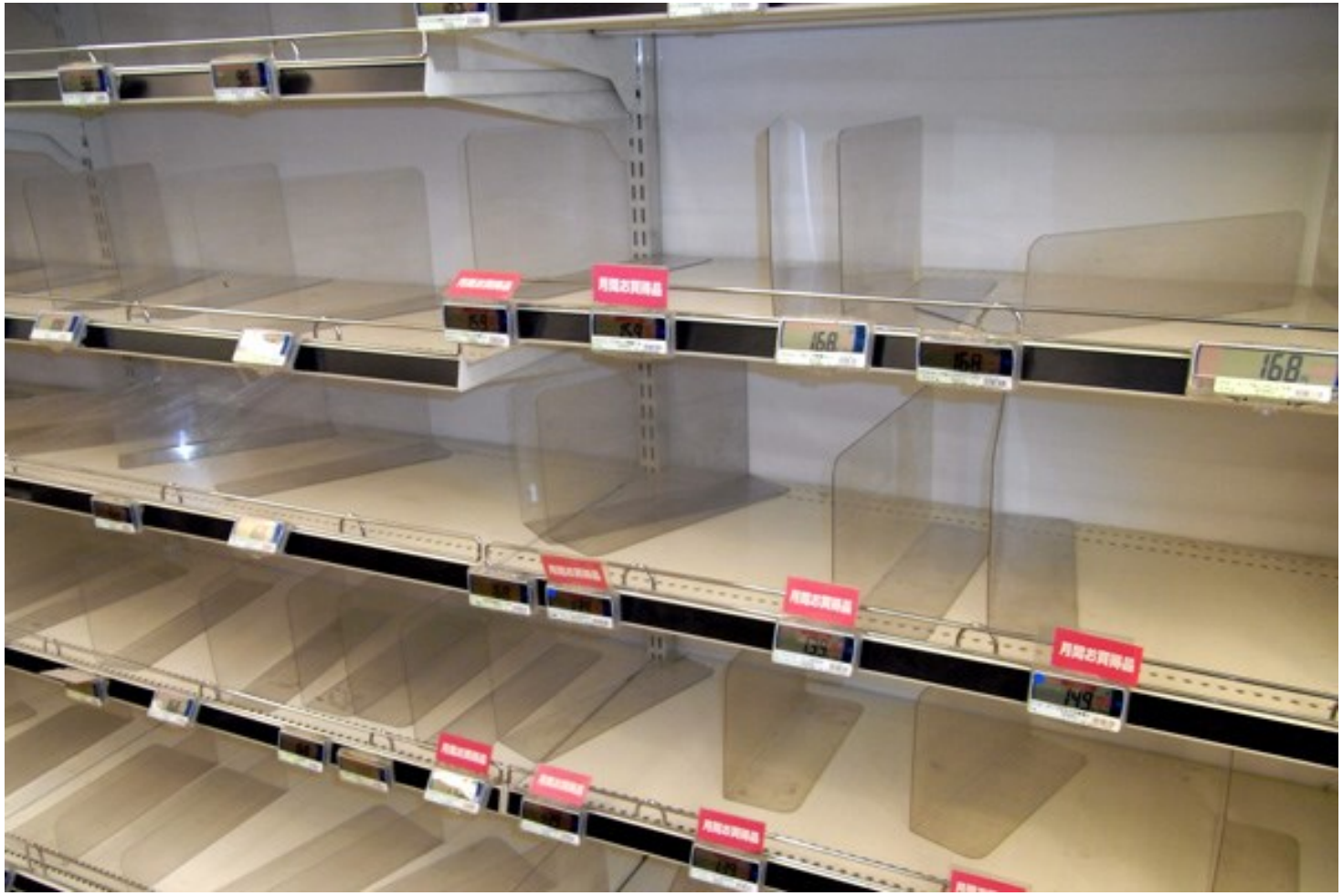
Definição de elementos

Definida a estrutura, os dados podem ser inseridos, atualizados e apagados, respeitadas as regras de integridade definidas no projeto lógico

```
INSERT INTO nome_da_tabela (atributo1, atributo2, ...)
VALUES (valor1, valor2, ...);

UPDATE nome_da_tabela
SET atributo = valor
WHERE condições

DELETE FROM table_name
WHERE some_column=some_value;
```



Consultas em SQL

Consultas em SQL

A cláusula SELECT seleciona dados em um banco, de acordo com os parâmetros inseridos pelo usuário.

```
SELECT column1, column2 FROM table_name
WHERE column_name operator value;
```

Emprega-se o * para selecionar todas as colunas da tabela

```
SELECT * FROM table_name //
```

A cláusula DISTINCT altera a declaração SELECT para retornar apenas valores distintos.

```
SELECT DISTINCT column_name,column_name FROM table_name;
```

Consultas em SQL

Valores de texto devem aparecer entre aspas simples (coluna = 'Teste’).

Operadores possíveis: =, <>, >, <, >=, <=, BETWEEN, LIKE, IN

```
SELECT * FROM table_name WHERE column BETWEEN ‘A’ AND ‘G’;
SELECT * FROM table_name WHERE column LIKE ‘%est%’;
SELECT * FROM table_name WHERE column IN {a,b,c,...};
```

A cláusula ALIAS atribui um nome temporário a uma coluna ou tabela.

```
SELECT a.column_name1 AS col1,b.column_name2 AS col2
FROM table_name1 AS a, table_name2 AS b
WHERE condições
```

Consultas em SQL

Funções Agregadas SQL

AVG(): calcula média dos valores de um campo do tipo numérico.

COUNT(): número de linhas.

FIRST(): retorna o primeiro valor.

LAST(): retorna o último valor.

MAX(): retorna o maior valor.

MIN(): retorna o menor valor.

SUM(): retorna a soma de valores de um campo do tipo numérico.

GROUP BY: é empregado juntamente com funções agregadas para agrupar os resultados por uma ou mais colunas.

Consultas em SQL

Funções Agregadas SQL

HAVING: funciona como WHERE, em caso do emprego de funções agregadas.

```
SELECT column_name, aggregate_function(column_name)
FROM table_name
WHERE column_name operator value
GROUP BY column_name;
HAVING aggregate_function(column_name) operator value;
```

Álgebra Relacional

Álgebra Relacional

Consiste de um conjunto de operações

- **entrada**: uma ou duas relações;
- **saída**: uma nova relação resultado;Consequentemente, algumas entidades ou relacionamentos podem gerar mais de uma tabela;

Fundamentais

– seleção;

– projeção;

– produto cartesiano;

– renomear;

– união;

– diferença de conjuntos.

Adicionais

– intersecção de conjuntos;

– junção natural;

– divisão;

– agregação.

Álgebra Relacional

Seleção

Seleciona tuplas que satisfaçam à condição de seleção.

nro_cli	nome_cli	end_cli	saldo	cod_vend
1	Márcia	Rua X	100,00	1
2	Cristina	Avenida 1	10,00	1
3	Manoel	Avenida 3	234,00	1
4	Rodrigo	Rua X	137,00	2

Álgebra Relacional

Projeção

Projeta as colunas solicitadas, i.e. produz um subconjunto vertical.

nro_cli	nome_cli	end_cli	saldo	cod_vend
1	Márcia	Rua X	100,00	1
2	Cristina	Avenida 1	10,00	1
3	Manoel	Avenida 3	234,00	1
4	Rodrigo	Rua X	137,00	2

Produto Cartesiano

Combina tuplas de duas relações, que não precisam ter atributos em comum

cliente (nro_cli, nome_cli, end_cli, saldo, cod_vend)

nro_cli	nome_cli	end_cli	saldo	cod_vend
1	Márcia	Rua X	100,00	1
2	Cristina	Avenida 1	10,00	1
3	Manoel	Avenida 3	234,00	1
4	Rodrigo	Rua X	137,00	2

vendedor (cod_vend, nome_vend)

cod_vend	nome_vend
1	Adriana
2	Roberto

nro_cli	nome_cli	end_cli	saldo	cliente. cod_vend	vendedor. cod_vend	nome_vend
1	Márcia	Rua X	100,00	1	1	Adriana
1	Márcia	Rua X	100,00	1	2	Roberto
2	Cristina	Avenida 1	10,00	1	1	Adriana
2	Cristina	Avenida 1	10,00	1	2	Roberto
3	Manoel	Avenida 3	234,00	1	1	Adriana
3	Manoel	Avenida 3	234,00	1	2	Roberto
4	Rodrigo	Rua X	137,00	2	1	Adriana
4	Rodrigo	Rua X	137,00	2	2	Roberto

grau: número de atributos de cliente + número de atributos de vendedor

número de tuplas: número de tuplas de cliente * número de tuplas de vendedor

Álgebra Relacional

União

Combina o resultado de duas ou mais operações SELECT. Cada subconjunto deve ter o mesmo número de colunas e o mesmo domínio, assim como as colunas a serem unidas devem estar na mesma ordem.

São considerados apenas valores distintos: para considerar valores repetidos, emprega-se a expressão UNION ALL .

```
SELECT column_name(s) FROM table1
UNION
SELECT column_name(s) FROM table2;
```

Álgebra Relacional

Diferença

Retorna as tuplas presentes na primeira relação e ausentes na segunda. Não se trata de uma opção comutativa;

```
SELECT table1.id FROM table1
WHERE table1.id NOT IN
(SELECT id FROM table2)
```

Álgebra Relacional

Interseção

Retorna as linhas representadas tanto no primeiro quanto no segundo subconjunto. As condições para o emprego do UNION também se aplicam ao INTERSECT.

```
SELECT column_name(s) FROM table1
INTERSECT
SELECT column_name(s) FROM table2;
```

Álgebra Relacional

Junção

Concatena tuplas relacionadas de duas relações. Inicialmente, forma um produto cartesiano das relações e, em seguida, faz uma seleção forçando igualdade sobre os atributos que aparecem nas relações.

```
SELECT nro_cli, nome_cli,
end_cli,saldo, vendedor.cod_vend,
nome_vend
FROM cliente (INNER) JOIN
vendedor
ON cliente.cod_vend =
vendedor.cod_vend
```

R

A	B	C
1	a	x
2	b	y
3	a	y
4	c	y

S

A	D
1	d
2	d
5	e

R ⋈ S

R.A	S.A	B	C	D
1	1	a	x	d
2	2	b	y	d

Álgebra Relacional

Left Join

Mantém as colunas da relação da esquerda. Se houver combinação com a relação da direita, os campos são preenchidos com os valores dessa tabela. Caso contrário, os campos são preenchidos com o valor NULL .

```
SELECT column_name(s)
FROM table1
LEFT (OUTER) JOIN table2
ON
table1.column_name=table2.column_
name;
```

R

A	B	C
1	a	x
2	b	y
3	a	y
4	c	y

S

A	D
1	d
2	d
5	e

R ⟕ S

R.A	S.A	B	C	D
1	1	a	x	d
2	2	b	y	d
3	Null	a	y	Null
4	Null	c	y	Null

Álgebra Relacional

Right Join

Mantém as colunas da relação da direita. Se houver combinação com a relação da esquerda, os campos são preenchidos com os valores dessa tabela. Caso contrário, os campos são preenchidos com o valor NULL .

```
SELECT column_name(s)
FROM table1
RIGHT (OUTER) JOIN table2
ON
table1.column_name=table2.column_
name;
```

R

A	B	C
1	a	x
2	b	y
3	a	y
4	c	y

S

A	D
1	d
2	d
5	e

R ⟖ S

R.A	S.A	B	C	D
1	1	a	x	d
2	2	b	y	d
Null	5	Null	Null	e

Álgebra Relacional

Full Join

Mantém cada uma das colunas de ambas as relações. Se houver combinação, os campos são preenchidos com os valores correspondentes. Caso contrário, os campos são preenchidos com o valor NULL .

```
SELECT column_name(s)
FROM table1
FULL (OUTER) JOIN table2
ON
table1.column_name=table2.column_
name;
```

R

A	B	C
1	a	x
2	b	y
3	a	y
4	c	y

S

A	D
1	d
2	d
5	e

R ⟗ S

R.A	S.A	B	C	D
1	1	a	x	d
2	2	b	y	d
3	Null	a	y	Null
4	Null	c	y	Null
Null	5	Null	Null	e

Álgebra Relacional

Processamento da Consulta SQL

- Aplica-se o predicado que aparece na cláusula WHERE
- Agrupam-se as tuplas que satisfazem a cláusula WHERE por meio da cláusula GROUP BY;
- Aplica-se a cláusula HAVING a cada grupo
- Removem-se os grupos que não satisfazem o predicado da cláusula HAVING
- Verifica-se se há limites para exibição
- Exibem-se as colunas listadas na cláusula SELECT

Resumo

Definição de Elementos: CREATE, DROP, INSERT INTO, ALTER e DELETE;

Consultas SQL: SELECT, DISTINCT, ORDER BY, GROUP BY, ALIAS, LIMIT;

Álgebra Relacional: Operações Fundamentais

- seleção
- projeção
- produto cartesiano
- renomear
- união
- diferença de conjuntos

Álgebra Relacional: Operações Adicionais

- intersecção de conjuntos
- junção natural: interna e externa
- divisão
- agregação