

Complementos de Bases de Dados – *Metadata* –

Engenharia Informática

2º Ano / 1º Semestre

Cláudio Miguel Sapateiro

claudio.sapateiro@estsetubal.ips.pt

DSI :: Escola Superior de Tecnologia de Setúbal :: Instituto Politécnico de Setúbal

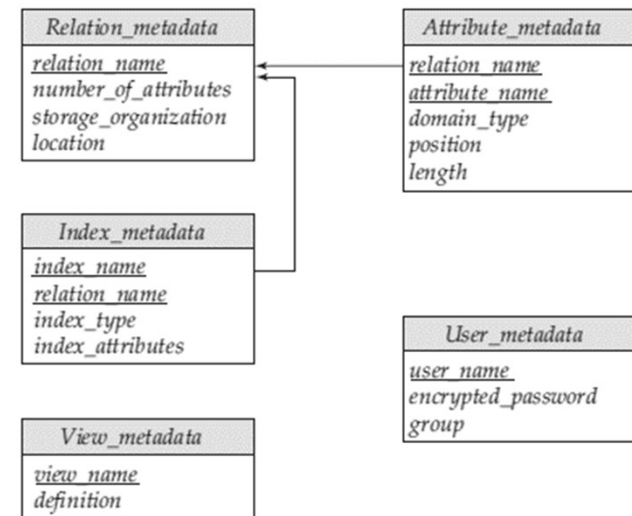
O que são?

METADADOS (CATÁLOGO)

Metadados

Uma BD sobre a BD !

- A BD tem de manter um conjunto de dados sobre os dados!
- Esta informação está persistida no que se costuma designar o **Catalogo** ou **Dicionário**
- Exemplos:
 - Nomes das relações
 - Nomes dos atributos (e tipos)
 - Nomes e definições das *views* existentes
 - Restrições de integridade (e.g. key constraints)
 - Índices
- Adicionalmente
 - Utilizadores, credenciais e permissões
 - Ficheiros físicos e sua organização
- e ainda, informação complementar como
 - Numero de tuplos de uma relação, estatísticas, ...



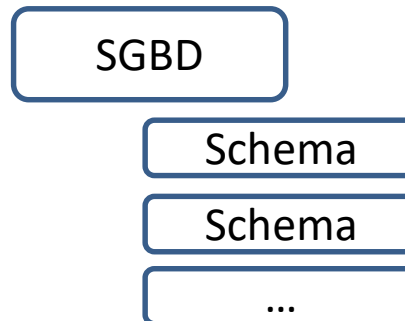
Que utilizações?

METADADOS

Schemas

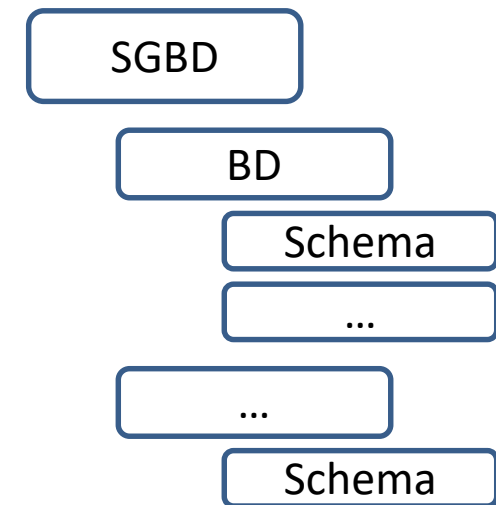
- MySQL:

- Schema = BD



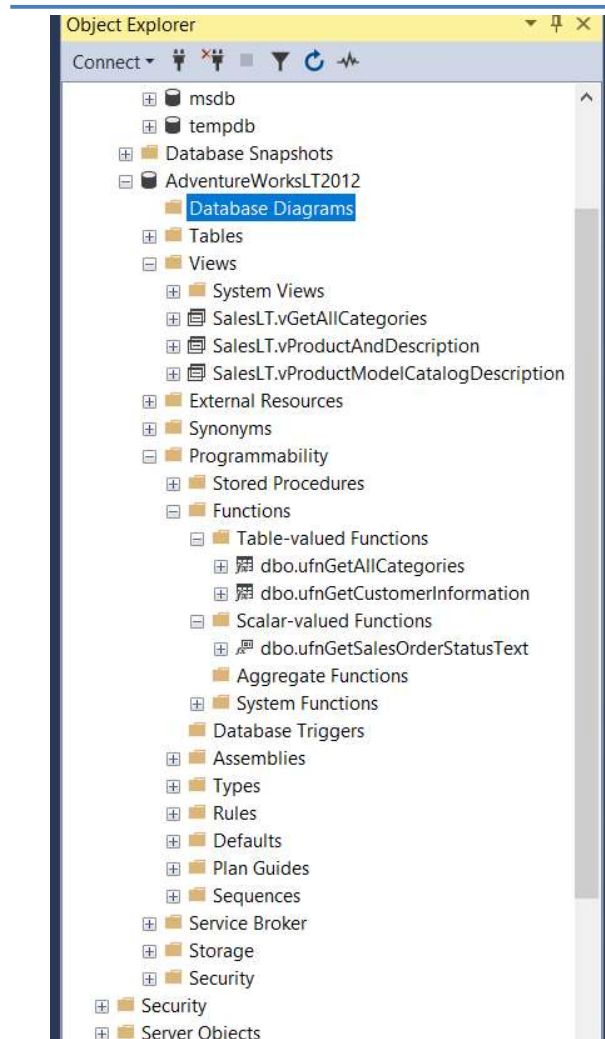
- MS SQL:

- BDs podem ter multiplos schemas

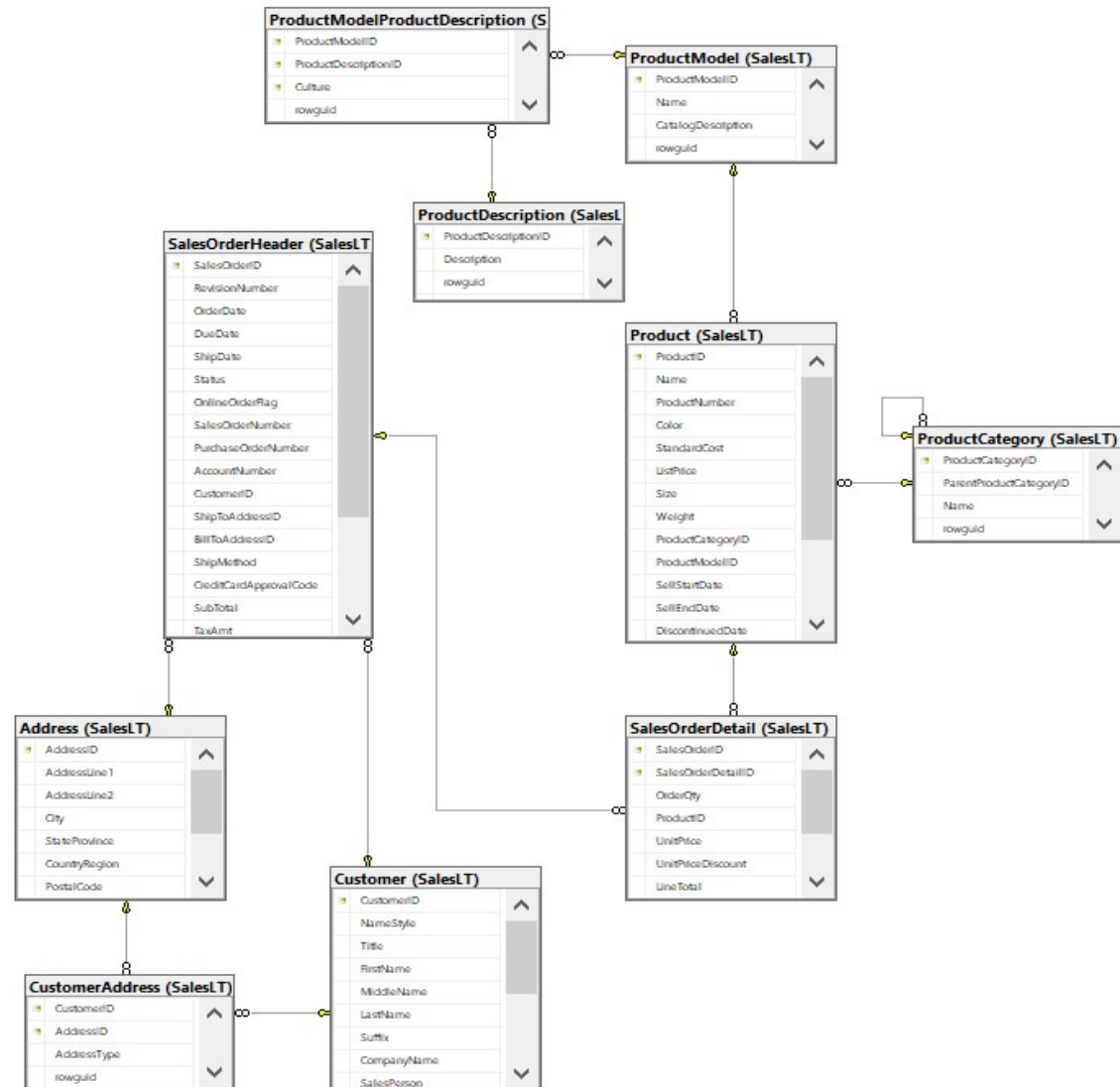


❖ Schema ~ namespace/package

AdventureWorksLT2012

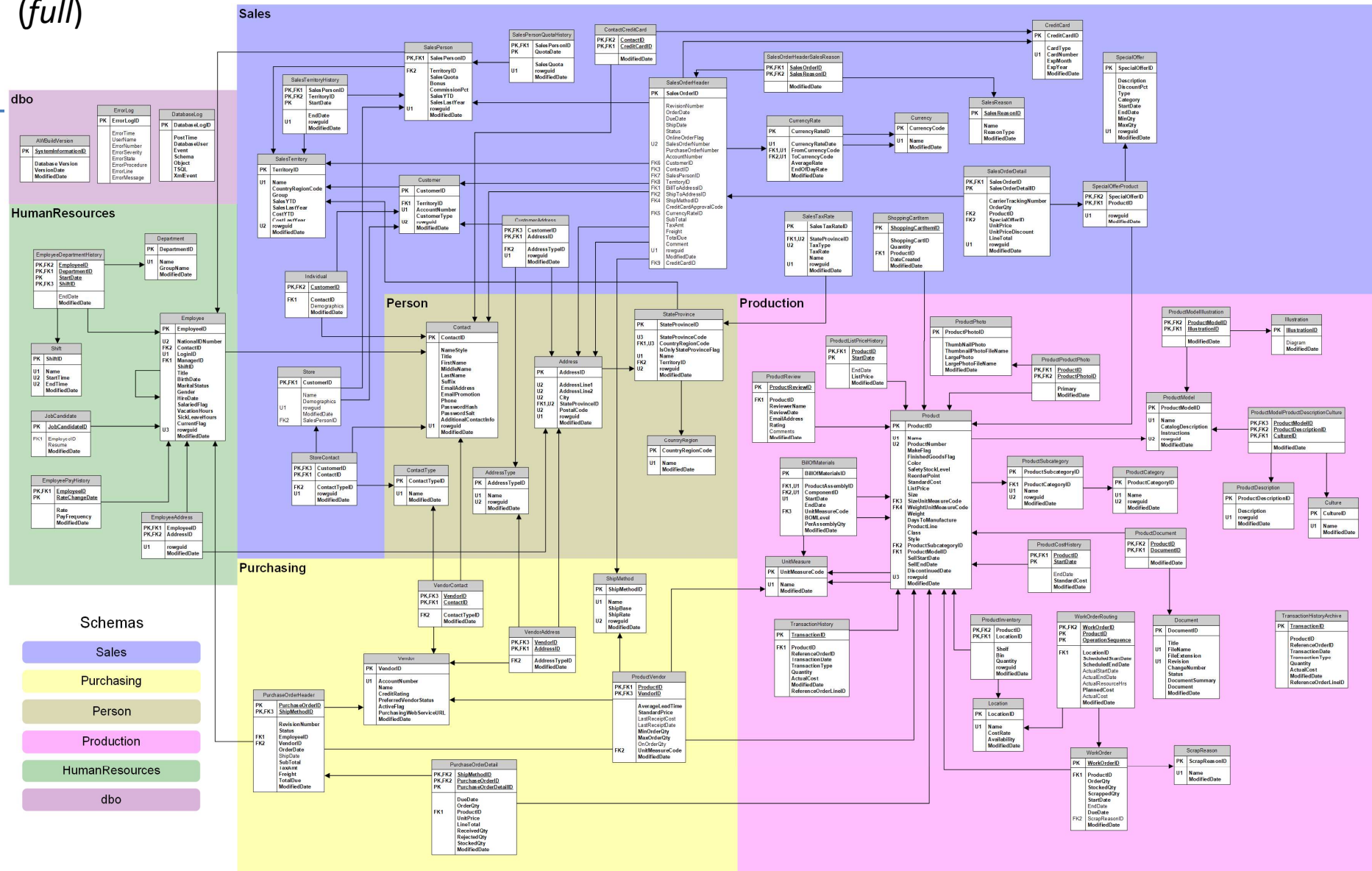


DSI::EST-IPS



Complementos de Bases de Dados

Adventure Works (full)



Acesso aos Metadados – *MS SQL*

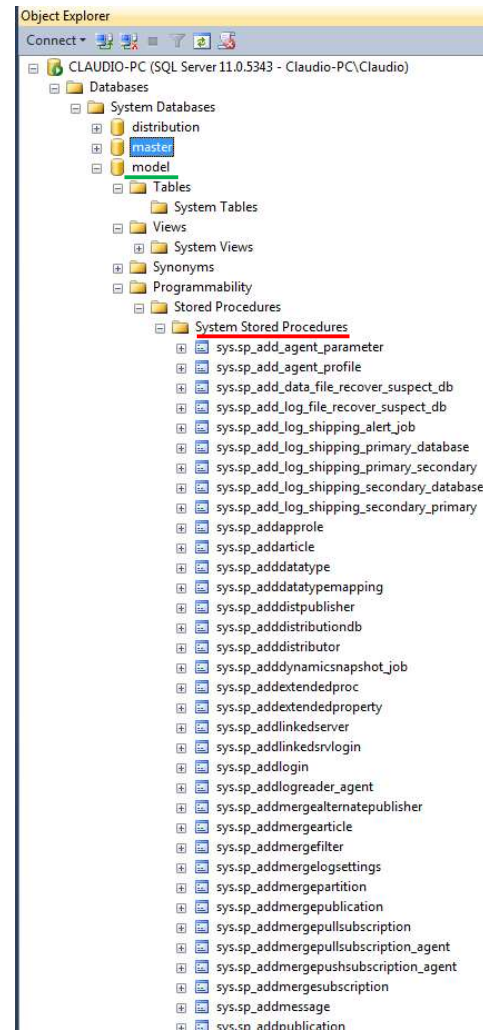
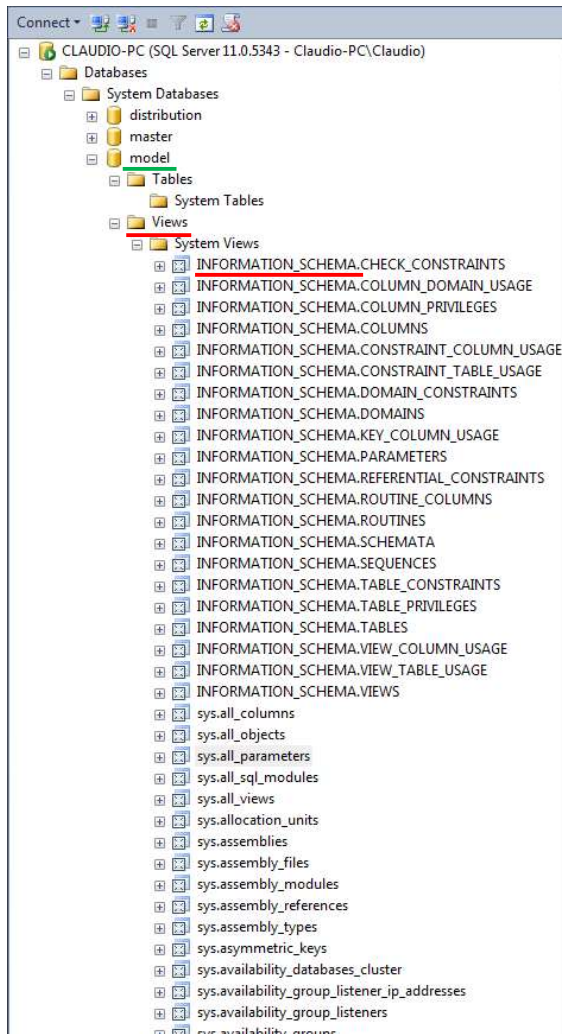
- ***Sys Schema Views***

- Uma das formas (preferencial = desempenho) de aceder aos metadados
- Por serem *views* suportam a independência de eventuais alterações “físicas” às tabelas de sistema
- Quer as *views* quer as suas colunas são auto-descritivas, de forma a apreender a informação relativa aos metadados solicitados

- ***Information_Schema views***

- Seguem as definições standard ISO relativas às vistas sobre o catalogo
- Apresentam a informação dos metadados em formato independente de qualquer implementação das tabelas do catálogo
- As aplicações quando usam esta coleção de *views* são tendencialmente mais portáteis entre diferentes SGBDs.

Metadados



Information Schema views

Mapping

SQL Server name	Maps to this equivalent SQL standard name
Database	Catalog
Schema	Schema
Object	Object
user-defined data type	Domain



Views

CHECK_CONSTRAINTS	REFERENTIAL_CONSTRAINTS
COLUMN_DOMAIN_USAGE	ROUTINES
COLUMN_PRIVILEGES	ROUTINE_COLUMNS
COLUMNS	SCHEMATA
CONSTRAINT_COLUMN_USAGE	TABLE_CONSTRAINTS
CONSTRAINT_TABLE_USAGE	TABLE_PRIVILEGES
DOMAIN_CONSTRAINTS	TABLES
DOMAINS	VIEW_COLUMN_USAGE
KEY_COLUMN_USAGE	VIEW_TABLE_USAGE
PARAMETERS	VIEWS

Exemplo:

```
SELECT TABLE_CATALOG, TABLE_SCHEMA, TABLE_NAME,  
       COLUMN_NAME, COLUMN_DEFAULT  
FROM  
AdventureWorksLT2012.INFORMATION_SCHEMA.COLUMNS  
WHERE TABLE_NAME = N'Product';
```

“Equivalentes”

SQL

Information_Schema

```
SELECT TABLE_CATALOG, TABLE_SCHEMA,  
TABLE_NAME, COLUMN_NAME  
FROM INFORMATION_SCHEMA.COLUMNS  
WHERE TABLE_NAME = N'Product';
```

Sys Schema

```
select s.name as 'SchemaName',  
o.name as 'TableName',  
c.name as 'ColumnName'  
from sys.schemas as s  
    inner join sys.all_objects as o  
        on s.schema_id = o.schema_id  
    inner join sys.all_columns as c  
        on c.object_id = o.object_id  
where o.name like N'Product' and o.type = 'U'  
order by SchemaName, TableName, ColumnName;
```

InformationSchema vs Sys Schema Views

Information Schema

- + “names friendly”
- + Joins através de *names*
- + Standard/potencialmente mais interoperavel
- Informação mais limitada
- Desempenho pode ser inferior

Sys Schema

- + Melhor desempenho
- + Informação mais promenorizada
- Orientado a objectos
- Joins por objectID
- Menos inteligível
- Proprietário

Exemplos de Objetos

SQL

use AdventureWorks2012

```
SELECT *  
FROM sys.all_objects
```

	name	object_id	principal_id	schema_id	parent_object_id	type	type_desc
1	sp_MSreadyhavegeneration	-1073624922	NULL	4	0	P	SQL_STORED_PROCEDURE
2	sp_MSwritermergeperfcounter	-1072815163	NULL	4	0	P	SQL_STORED_PROCEDURE
3	TABLE_PRIVILEGES	-1072372588	NULL	3	0	V	VIEW
4	sp_replicatsyncstatus	-1071944761	NULL	4	0	X	EXTENDED_STORED_PROCEDURE
5	sp_replicshowcmds	-1070913306	NULL	4	0	P	SQL_STORED_PROCEDURE
6	sp_publishdb	-1070573756	NULL	4	0	P	SQL_STORED_PROCEDURE
7	sp_addqueued_artinfo	-1068897509	NULL	4	0	P	SQL_STORED_PROCEDURE
8	sp_replicounters	-1068756304	NULL	4	0	X	EXTENDED_STORED_PROCEDURE
9	sp_MSget_subscription_dts_info	-1068452095	NULL	4	0	P	SQL_STORED_PROCEDURE
10	sp_help_spatial_geometry_index_xml	-1068265529	NULL	4	0	P	SQL_STORED_PROCEDURE
11	sp_password	-1067822458	NULL	4	0	P	SQL_STORED_PROCEDURE
12	sp_MSstopdistribution_agent	-1067634502	NULL	4	0	P	SQL_STORED_PROCEDURE
13	sp_replicmonitorrefreshjob	-1067473073	NULL	4	0	P	SQL_STORED_PROCEDURE
14	sp_redirect_publisher	-1065960762	NULL	4	0	P	SQL_STORED_PROCEDURE
15	sp_MSenumpartialchangesdirect	-1065074012	NULL	4	0	P	SQL_STORED_PROCEDURE
16	sp_MSupdate_subscriber_info	-1064594347	NULL	4	0	P	SQL_STORED_PROCEDURE
17	sp_MSdrop_distribution_agent	-1064529606	NULL	4	0	P	SQL_STORED_PROCEDURE
18	sp_bindsession	-1064199433	NULL	4	0	X	EXTENDED_STORED_PROCEDURE
19	sp_MSallocate_new_identity_range	-1064127790	NULL	4	0	P	SQL_STORED_PROCEDURE

```
SELECT  
    distinct ot.type,  
    ot.type_desc  
FROM sys.all_objects ot
```

	type	type_desc
1	FN	SQL_SCALAR_FUNCTION
2	IF	SQL_INLINE_TABLE_VALUED_FUNCTION
3	C	CHECK_CONSTRAINT
4	UQ	UNIQUE_CONSTRAINT
5	SQ	SERVICE_QUEUE
6	F	FOREIGN_KEY_CONSTRAINT
7	U	USER_TABLE
8	FS	CLR_SCALAR_FUNCTION
9	D	DEFAULT_CONSTRAINT
10	PK	PRIMARY_KEY_CONSTRAINT
11	V	VIEW
12	AF	AGGREGATE_FUNCTION
13	S	SYSTEM_TABLE
14	IT	INTERNAL_TABLE
15	P	SQL_STORED_PROCEDURE
16	X	EXTENDED_STORED_PROCEDURE
17	TF	SQL_TABLE_VALUED_FUNCTION
18	TR	SQL_TRIGGER
19	PC	CLR_STORED_PROCEDURE

BDs de Sistema

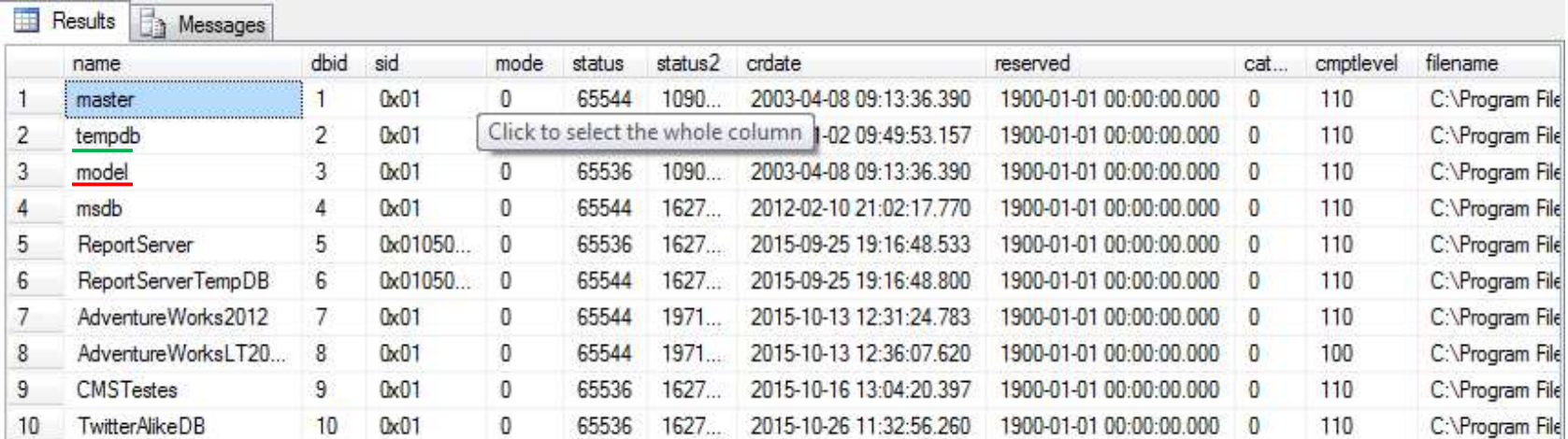
Exemplo

Use Master

GO

SELECT *

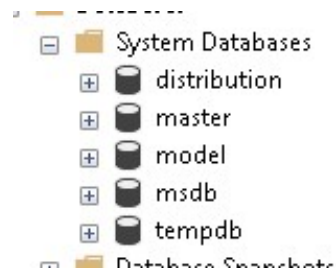
FROM sys.databases



	name	dbid	sid	mode	status	status2	crdate	reserved	cat...	cmptlevel	filename
1	master	1	0x01	0	65544	1090...	2003-04-08 09:13:36.390	1900-01-01 00:00:00.000	0	110	C:\Program File
2	tempdb	2	0x01	0	65536	1090...	2003-04-08 09:13:36.390	1900-01-01 00:00:00.000	0	110	C:\Program File
3	model	3	0x01	0	65536	1090...	2003-04-08 09:13:36.390	1900-01-01 00:00:00.000	0	110	C:\Program File
4	msdb	4	0x01	0	65544	1627...	2012-02-10 21:02:17.770	1900-01-01 00:00:00.000	0	110	C:\Program File
5	ReportServer	5	0x01050...	0	65536	1627...	2015-09-25 19:16:48.533	1900-01-01 00:00:00.000	0	110	C:\Program File
6	ReportServerTempDB	6	0x01050...	0	65544	1627...	2015-09-25 19:16:48.800	1900-01-01 00:00:00.000	0	110	C:\Program File
7	AdventureWorks2012	7	0x01	0	65544	1971...	2015-10-13 12:31:24.783	1900-01-01 00:00:00.000	0	110	C:\Program File
8	AdventureWorksLT20...	8	0x01	0	65544	1971...	2015-10-13 12:36:07.620	1900-01-01 00:00:00.000	0	100	C:\Program File
9	CMSTestes	9	0x01	0	65536	1627...	2015-10-16 13:04:20.397	1900-01-01 00:00:00.000	0	110	C:\Program File
10	TwitterAlikeDB	10	0x01	0	65536	1627...	2015-10-26 11:32:56.260	1900-01-01 00:00:00.000	0	110	C:\Program File

BDs de Sistema

SQL Server includes the following system databases.



System database	Description
master Database	Records all the system-level information for an instance of SQL Server.
msdb Database	Is used by SQL Server Agent for scheduling alerts and jobs.
model Database	Is used as the template for all databases created on the instance of SQL Server. Modifications made to the model database, such as database size, collation, recovery model, and other database options, are applied to any databases created afterward.
Resource Database	Is a read-only database that contains system objects that are included with SQL Server. System objects are physically persisted in the Resource database, but they logically appear in the sys schema of every database.
tempdb Database	Is a workspace for holding temporary objects or intermediate result sets.

BDs de Sistema

MS SQL

- Contem a BD: **master**
 - armazena informações sobre as bases de dados e seus objetos existentes no SGBD
 - Muito importante pois preserva (meta) informação sobre *user* DBs (e e.g. *logins*)
 - contudo há que assegurar que não se inscreva diretamente objetos sobre esta (**USE 'uBD'**)
- Objetos de uma BD
 - Tabelas que suportam os registos
 - Tipos de Dados (de sistema e definidos pelo utilizador)
 - Constraints
 - Indices
 - Views
 - Stored Procedures
 - Funções
 - Triggers
 - ...

Information_Schema

SQL

Views dos Metadados

- Cada *view* do **INFORMATION_SCHEMA** contém meta informação sobre os objetos armazenados numa base de dados

INFORMATION_SCHEMA (Database metadata)	
CHECK_CONSTRAINTS	one row for each CHECK constraint.
COLUMNS	one row for each column.
KEY_COLUMN_USAGE	one row for each column that is constrained as a key.
REFERENTIAL_CONSTRAINTS	one row for each FOREIGN KEY constraint.
TABLE_CONSTRAINTS	one row for each table constraint.
TABLES	one row for each table in the current database.
VIEW_COLUMN_USAGE	one row for each column that is used in a view definition.
VIEW_TABLE_USAGE	one row for each table that is used in a view.
VIEWS	one row for each view.
COLUMN_DOMAIN_USAGE	one row for each column that has an alias data type.
COLUMN_PRIVILEGES	one row for each column that has a privilege that is either granted to or granted.
CONSTRAINT_COLUMN_USAGE	one row for each column that has a constraint defined on the column.
CONSTRAINT_TABLE_USAGE	one row for each table that has a constraint defined on the table.
DOMAIN_CONSTRAINTS	one row for each alias data type that has a rule bound to it.
DOMAINS	one row for each alias data type.
PARAMETERS	one row for each parameter of a user-defined function or stored procedure.
ROUTINES	one row for each stored procedure and function.
ROUTINE_COLUMNS	one row for each column returned by the table-valued functions.
SCHEMATA	one row for each schema in the current database.
TABLE_PRIVILEGES	one row for each table privilege that is granted to or granted by the current user.

Sys Schema

SQL

Views dos Metadados

- Cada *view* do **SYS SCHEMA** contém meta informação sobre os objetos armazenados numa base de dados

```
select s.name as 'SchemaName', o.name as 'TableName'
from sys.schemas as s
      inner join sys.all_objects as o
      on s.schema_id = o.schema_id
where s.name='sys' and
      o.type = 'V'
order by SchemaName, TableName;
```

mini Sumário

1. O que são schemas?
2. O que é o catalogo de uma BD?
3. Quais as principais diferenças entre o InformationSchema e o Sys schema?

Exercício

AdventureWorksLT2012



Escolher a(s) opções de query para cada um dos pontos:

1. Listar todas as tabelas da BD?
2. Quantas tabelas tem a BD?
3. Quantas colunas tem cada tabela?

10:00

```
--A select * from INFORMATION_SCHEMA.TABLES;      --C select * from sys.tables;
--B select * from INFORMATION_SCHEMA.TABLES t      --D select count(*)
where t.TABLE_TYPE=upper('base table');           from sys.tables t ;

--E select c.TABLE_NAME, count(c.COLUMN_NAME)
from INFORMATION_SCHEMA.COLUMNS c join INFORMATION_SCHEMA.TABLES t
on t.TABLE_NAME=c.TABLE_NAME where t.TABLE_TYPE='base table'
group by c.TABLE_NAME;

--F select t.name, count(*)
from sys.all_columns as c inner join sys.tables t on c.object_id=t.object_id
group by t.name;
```

SYSTEM STORED PROCEDURES AND FUNCTIONS

System Stored Procedures and Functions

- *stored procedures* e *functions* de sistema retornam informação do catalogo
- Tratam-se de *sps* e *functions* específicas do MS SQL
 - Contudo isoladas da implementação do catalogo subjacente

Procedures

sp_column_privileges	sp_special_columns
sp_columns	sp_sproc_columns
sp_databases	sp_statistics
sp_fkeys	sp_stored_procedures
sp_pkeys	sp_table_privileges
sp_server_info	sp_tables

Functions

@@PROCID	INDEX_COL
APP_NAME	INDEXKEY_PROPERTY
APPLOCK_MODE	INDEXPROPERTY
APPLOCK_TEST	NEXT VALUE FOR
ASSEMBLYPROPERTY	OBJECT_DEFINITION
COL_LENGTH	OBJECT_ID
COL_NAME	OBJECT_NAME
COLUMNPROPERTY	OBJECT_SCHEMA_NAME
DATABASE_PRINCIPAL_ID	OBJECTPROPERTY
DATABASEPROPERTYEX	OBJECTPROPERTYEX
DB_ID	ORIGINAL_DB_NAME
DB_NAME	PARSENAME
FILE_ID	SCHEMA_ID
FILE_INDEX	SCHEMA_NAME
FILE_NAME	SCOPE_IDENTITY
FILEGROUP_ID	SERVERPROPERTY
FILEGROUP_NAME	STATS_DATE
FILEGROUPPROPERTY	TYPE_ID
FILEPROPERTY	TYPE_NAME
FULLTEXTCATALOGPROPERTY	TYPEPROPERTY
FULLTEXTSERVICEPROPERTY	

Stored Procedures

SQL

sp de Sistema vs Consultas (a tabelas de sistema)

- Lista de tabelas de utilizador numa BD

SELECT name

FROM sys.all_objects

WHERE type = 'U'

- ou

EXEC sp_tables

The screenshot displays the SQL Server Enterprise Manager interface on the left and a SQL Query Window on the right. In the Enterprise Manager, the 'AdventureWorksLT2012' database is selected, and the 'System Tables' folder under 'Tables' is expanded, showing a list of system tables including 'BuildVersion', 'ErrorLog', 'Address', 'Customer', 'CustomerAddress', 'Product', 'ProductCategory', 'ProductDescription', 'ProductModel', 'ProductModelProductDescription', 'SalesOrderDetail', and 'SalesOrderHeader'. The SQL Query Window shows a query that uses the 'AdventureWorksLT2012' database and selects the 'name' column from 'sysobjects' where the 'type' is 'U'. The 'Results' tab is active, showing a table with 12 rows of system table names.

	name
1	BuildVersion
2	Address
3	Customer
4	CustomerAddress
5	Product
6	ProductCategory
7	ProductDescription
8	ProductModel
9	ProductModelProductDescription
10	SalesOrderDetail
11	SalesOrderHeader
12	ErrorLog

Stored Procedures

sp's para acesso a Meta informação


System stored procedures that implement data dictionary functions.	
sp_column_privileges	Returns column privilege information for a single table in the current environment.
sp_columns	Returns column information for the specified objects that can be queried in the current environment.
sp_fkeys	Returns logical foreign key information for the current environment.
sp_pkeys	Returns primary key information for a single table in the current environment.
sp_server_info	Returns a list of attribute names and matching values for SQL Server
sp_special_columns	Returns the optimal set of columns that uniquely identify a row in the table.
sp_sproc_columns	Returns column information for a single stored procedure or user-defined function in the current environment.
sp_statistics	Returns a list of all indexes and statistics on a specified table or indexed view.
sp_stored_procedures	Returns a list of stored procedures in the current environment.
sp_table_privileges	Returns a list of table permissions (such as INSERT, DELETE, UPDATE, SELECT, REFERENCES) for the specified table or tables.
sp_tables	Returns a list of objects that can be queried in the current environment. This means any table or view, except synonym objects.

Stored Procedures

SQL

Exemplo

(Adventure Work LT)

- sys.sp_columns 'customer'  tabela

	TABLE_QUALIFIER	TABLE_OWNER	TABLE_NAME	COLUMN_NAME	DATA_TYPE	TYPE_NAME	PRECISION
1	AdventureWorksLT2012	SalesLT	Customer	CustomerID	4	int identity	10
2	AdventureWorksLT2012	SalesLT	Customer	NameStyle	-7	NameStyle	1
3	AdventureWorksLT2012	SalesLT	Customer	Title	-9	nvarchar	8
4	AdventureWorksLT2012	SalesLT	Customer	FirstName	-9	Name	50
5	AdventureWorksLT2012	SalesLT	Customer	MiddleName	-9	Name	50
6	AdventureWorksLT2012	SalesLT	Customer	LastName	-9	Name	50
7	AdventureWorksLT2012	SalesLT	Customer	Suffix	-9	nvarchar	10
8	AdventureWorksLT2012	SalesLT	Customer	CompanyName	-9	nvarchar	128
9	AdventureWorksLT2012	SalesLT	Customer	SalesPerson	-9	nvarchar	256
10	AdventureWorksLT2012	SalesLT	Customer	EmailAddress	-9	nvarchar	50
11	AdventureWorksLT2012	SalesLT	Customer	Phone	-9	Phone	25
12	AdventureWorksLT2012	SalesLT	Customer	PasswordHash	12	varchar	128
13	AdventureWorksLT2012	SalesLT	Customer	PasswordSalt	12	varchar	10
14	AdventureWorksLT2012	SalesLT	Customer	rowguid	-11	uniqueidenti...	36
15	AdventureWorksLT2012	SalesLT	Customer	ModifiedDate	11	datetime	23

Metadata Functions

SQL

```
SELECT DB_NAME()
```

```
SELECT DB_ID(N'AdventureWorksLT2012');
```

```
SELECT DB_NAME(database_id)
FROM sys.databases
WHERE database_id = DB_ID();
```

Functions

@@PROCID	INDEX_COL
APP_NAME	INDEXKEY_PROPERTY
APPLOCK_MODE	INDEXPROPERTY
APPLOCK_TEST	NEXT VALUE FOR
ASSEMBLYPROPERTY	OBJECT_DEFINITION
COL_LENGTH	OBJECT_ID
COL_NAME	OBJECT_NAME
COLUMNPROPERTY	OBJECT_SCHEMA_NAME
DATABASE_PRINCIPAL_ID	OBJECTPROPERTY
DATABASEPROPERTYEX	OBJECTPROPERTYEX
DB_ID	ORIGINAL_DB_NAME
DB_NAME	PARSENAME
FILE_ID	SCHEMA_ID
FILE_INDEX	SCHEMA_NAME
FILE_NAME	SCOPE_IDENTITY
FILEGROUP_ID	SERVERPROPERTY
FILEGROUP_NAME	STATS_DATE
FILEGROUPPROPERTY	TYPE_ID
FILEPROPERTY	TYPE_NAME
FULLTEXTCATALOGPROPERTY	TYPEPROPERTY
FULLTEXTSERVICEPROPERTY	

Metadata Functions

```
SELECT OBJECT_ID(N'SalesLT.Product');
```

```
SELECT OBJECT_NAME(OBJECT_ID(N'SalesLT.Product'));
```

```
SELECT OBJECT_NAME(c.object_id)
```

```
FROM sys.columns c
```

```
join sys.tables t
```

```
on t.object_id=c.object_id
```

```
WHERE c.is_nullable = 1
```

```
GROUP BY c.object_id
```

O que devolve?

Functions

@@PROCID	INDEX_COL
APP_NAME	INDEXKEY_PROPERTY
APPLOCK_MODE	INDEXPROPERTY
APPLOCK_TEST	NEXT VALUE FOR
ASSEMBLYPROPERTY	OBJECT_DEFINITION
COL_LENGTH	OBJECT_ID
COL_NAME	OBJECT_NAME
COLUMNPROPERTY	OBJECT_SCHEMA_NAME
DATABASE_PRINCIPAL_ID	OBJECTPROPERTY
DATABASEPROPERTYEX	OBJECTPROPERTYEX
DB_ID	ORIGINAL_DB_NAME
DB_NAME	PARSENAME
FILE_ID	SCHEMA_ID
FILE_INDEX	SCHEMA_NAME
FILE_NAME	SCOPE_IDENTITY
FILEGROUP_ID	SERVERPROPERTY
FILEGROUP_NAME	STATS_DATE
FILEGROUPPROPERTY	TYPE_ID
FILEPROPERTY	TYPE_NAME
FULLTEXTCATALOGPROPERTY	TYPEPROPERTY
FULLTEXTSERVICEPROPERTY	

Metadata Functions

SQL

O que devolve?

```
SELECT OBJECT_NAME(c.object_id)
FROM sys.columns c
join sys.tables t
on t.object_id=c.object_id
WHERE c.is_nullable = 1
GROUP BY c.object_id
```

-- ajuda

```
-- descriptions
-- columns
select * from sys.system_columns c
where c.object_id in
    (SELECT sv.object_id FROM sys.system_views sv
     where sv.name='columns' and sv.schema_id=schema_ID('sys'));

-- tables
select * from sys.system_columns c
where c.object_id in
    (SELECT sv.object_id FROM sys.system_views sv
     where sv.name='tables' and sv.schema_id=schema_ID('sys'));
```

Metadata Functions

“Categorias” da *metadata functions*

Categories				
Object	Database	Column	File	Schema
Object_Id()	Databaseproperty()	Col_Length()	File_Id()	Schema_Id()
Object_Name()	Databasepropertyex()	Col_Name()	File_Name()	Schema_Name()
Objectproperty()	Db_Id()	Columnproperty()	Filegroup_Id()	Object_Schema_Name()
Objectpropertyex()	Db_Name()		Filegroup_Name()	
Object_Schema_Name()			Filegroupproperty()	
Object_Definition()			Fileproperty()	

Index	FullText	Others	
Index_Col()	Fulltextcatalogproperty()	@@Procid	Stats_Date()
Indexkey_Property()	Fulltextserviceproperty()	Next Value For	Applock_Mode()
Indexproperty()		Typeproperty()	Applock_Test()
		Sql_Variant_Property()	ParseName()
		APP_NAME()	
		ASSEMBLYPROPERTY()	
		Scope_Identity()	

mini Sumário

1. Indique duas formas possíveis de aceder aos metadados?
2. As funções *OBJECT_ID()* e *OBJECT_NAME()* podem ser utilizadas em *queries* às *views* do *Sys Schema* e do *Information_Schema*?
3. As *sp* de sistema que facilitam o acesso a metadados pertencem ao *Sys schema*, ao *Information_Schema* ou ambos?

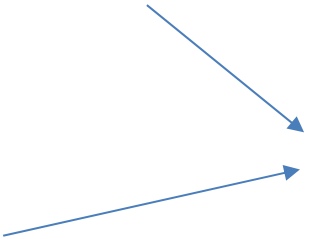
Exemplos

SQL

```
SELECT name, column_id  
FROM sys.columns  
WHERE object_id = OBJECT_ID(N'SalesLT.ProductModel');
```

ou

```
SELECT c.COLUMN_NAME,  
       c.ORDINAL_POSITION  
FROM INFORMATION_SCHEMA.COLUMNS c  
WHERE c.TABLE_NAME='ProductModel'
```



	COLUMN_NAME	ORDINAL_POSITION
1	ProductModelID	1
2	Name	2
3	CatalogDescription	3
4	rowguid	4
5	ModifiedDate	5

Exemplos

SQL

```
SELECT
    name,
    max_column_id_used as 'Numero de colunas'
FROM sys.tables st
WHERE schema_id = SCHEMA_ID(N'SalesLT');
```

O que devolve ?

	name	Numero de colunas
1	Address	9
2	Customer	15
3	CustomerAddress	5
4	Product	17
5	ProductCategory	5
6	ProductDescription	4
7	ProductModel	5
8	ProductModelProductDescripti...	5
9	SalesOrderDetail	9
10	SalesOrderHeader	22

O que pode
estar aqui mal?

Exemplos

SQL

Solução?

```
SELECT st.name, count(sc.name) as 'Numero de colunas'
FROM sys.columns sc
join sys.tables st
on sc.object_id=st.object_id
WHERE st.schema_id= schema_ID(N'SalesLT')
GROUP BY st.name
```

	name	Numero de colunas
1	Address	9
2	Customer	15
3	CustomerAddress	5
4	Product	17
5	ProductCategory	5
6	ProductDescription	4
7	ProductModel	5
8	ProductModelProductDescripti...	5
9	SalesOrderDetail	9
10	SalesOrderHeader	22

Exemplos

SQL

Solução? (alternativa – utilizando INFORMATION.SCHEMA)

Exercício

```
SELECT ct.TABLE_NAME, COUNT(*) as 'Numero de colunas'  
FROM INFORMATION_SCHEMA.COLUMNS ct  
WHERE ct.TABLE_SCHEMA = 'SalesLT'  
group by ct.TABLE_NAME
```

	TABLE_NAME	Numero de colunas
1	Address	9
2	Address Type	4
3	BusinessEntity	3
4	BusinessEntityAddress	5
5	BusinessEntityContact	5
6	Contact Type	3
7	CountryRegion	3
8	EmailAddress	5
9	Password	5
10	Person	13
11	PersonPhone	4
12	PhoneNumberType	3
13	StateProvince	8
14	vAdditionalContactInfo	17
15	vStateProvinceCountryRegion	7



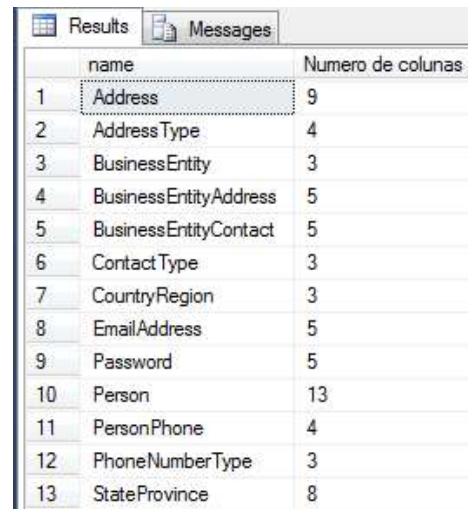
	name	Numero de colunas
1	Address	9
2	Address Type	4
3	BusinessEntity	3
4	BusinessEntityAddress	5
5	BusinessEntityContact	5
6	Contact Type	3
7	CountryRegion	3
8	EmailAddress	5
9	Password	5
10	Person	13
11	PersonPhone	4
12	PhoneNumberType	3
13	StateProvince	8

Exemplos

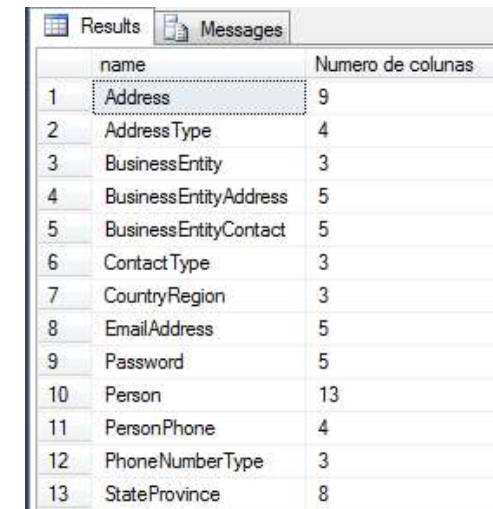
SQL

Solução? (alternativa revista – uma hipótese)

```
SELECT ct.TABLE_NAME, COUNT(*) as 'Numero de colunas'
FROM INFORMATION_SCHEMA.COLUMNS ct
WHERE ct.TABLE_SCHEMA = 'SalesLT' AND
      ct.TABLE_NAME not in (
        SELECT v.TABLE_NAME
        FROM INFORMATION_SCHEMA.VIEWS v
      )
group by ct.TABLE_NAME
```



	name	Numero de colunas
1	Address	9
2	Address Type	4
3	BusinessEntity	3
4	BusinessEntityAddress	5
5	BusinessEntityContact	5
6	Contact Type	3
7	CountryRegion	3
8	EmailAddress	5
9	Password	5
10	Person	13
11	PersonPhone	4
12	PhoneNumberType	3
13	StateProvince	8



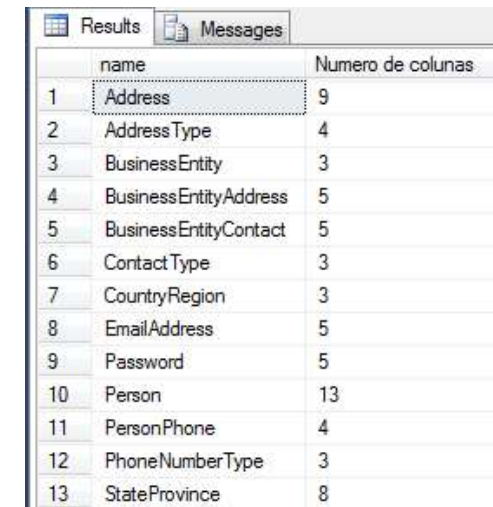
	name	Numero de colunas
1	Address	9
2	Address Type	4
3	BusinessEntity	3
4	BusinessEntityAddress	5
5	BusinessEntityContact	5
6	Contact Type	3
7	CountryRegion	3
8	EmailAddress	5
9	Password	5
10	Person	13
11	PersonPhone	4
12	PhoneNumberType	3
13	StateProvince	8

Exemplos

SQL

Solução? (alternativa revista – outra hipótese)

```
SELECT col.table_name, COUNT(col.column_name) 'Numero de colunas'
FROM information_schema.columns col
JOIN information_schema.tables tbl
  ON tbl.table_name = col.table_name
  AND tbl.table_schema = col.table_schema
  AND tbl.table_catalog = col.table_catalog
  AND tbl.table_type <> 'VIEW'
WHERE tbl.TABLE_SCHEMA = 'Person'
GROUP BY col.table_name
```



The screenshot shows a SQL Server Results window with two tabs: 'Results' and 'Messages'. The 'Results' tab is active, displaying a table with two columns: 'name' and 'Numero de colunas'. The table contains 13 rows of data, representing various tables in the 'Person' schema. The 'name' column lists the table names, and the 'Numero de colunas' column shows the count of columns for each table.

	name	Numero de colunas
1	Address	9
2	AddressType	4
3	BusinessEntity	3
4	BusinessEntityAddress	5
5	BusinessEntityContact	5
6	ContactType	3
7	CountryRegion	3
8	EmailAddress	5
9	Password	5
10	Person	13
11	PersonPhone	4
12	PhoneNumberType	3
13	StateProvince	8

5:00

Exercício

quiz



Qual dos comandos produz o resultado da figura?

A. `exec sp_tables;`

B. `select schema_name(t.schema_id), t.name,
t.type_desc
from sys.tables t;`

C. `select *
from INFORMATION_SCHEMA.TABLES;`

	TABLE_QUALIFIER	TABLE_OWNER	TABLE_NAME	TABLE_TYPE
7	AdventureWorksLT2012	SalesLT	Product	TABLE
8	AdventureWorksLT2012	SalesLT	ProductCategory	TABLE
9	AdventureWorksLT2012	SalesLT	ProductDescription	TABLE
10	AdventureWorksLT2012	SalesLT	ProductModel	TABLE
11	AdventureWorksLT2012	SalesLT	ProductModelProductDescripi...	TABLE
12	AdventureWorksLT2012	SalesLT	SalesOrderDetail	TABLE
13	AdventureWorksLT2012	SalesLT	SalesOrderHeader	TABLE
14	AdventureWorksLT2012	sys	trace_xe_action_map	TABLE
15	AdventureWorksLT2012	sys	trace_xe_event_map	TABLE
16	AdventureWorksLT2012	INFORMATION_SCHEMA	CHECK_CONSTRAINTS	VIEW
17	AdventureWorksLT2012	INFORMATION_SCHEMA	COLUMN_DOMAIN_USAGE	VIEW
18	AdventureWorksLT2012	INFORMATION_SCHEMA	COLUMN_PRIVILEGES	VIEW
19	AdventureWorksLT2012	INFORMATION_SCHEMA	COLUMNS	VIEW
20	AdventureWorksLT2012	INFORMATION_SCHEMA	CONSTRAINT_COLUMN_USA...	VIEW
21	AdventureWorksLT2012	INFORMATION_SCHEMA	CONSTRAINT_TABLE_USAGE	VIEW

5:00

Exercício

quiz



Que “problema” podemos apontar a esta utilização conjunta dos 2 *schemas*?

```
SELECT isv.TABLE_NAME, isv.VIEW_DEFINITION
FROM INFORMATION_SCHEMA.VIEWS isv
where isv.TABLE_NAME in
    ( SELECT
        OBJECT_NAME(sv.object_id)
      FROM sys.views sv
      where sv.is_replicated = 0
    )
```

Exemplos e Exercícios

Outras Funções “uteis”

Exemplos

<https://msdn.microsoft.com/en-us/library/ms176105.aspx>

OBJECTPROPERTY()

OBJECT_DEFINITION()

HasAfterTrigger	Table, view	Table or view has an AFTER trigger. 1 = True 0 = False
HasDeleteTrigger	Table, view	Table or view has a DELETE trigger. 1 = True 0 = False

```
SELECT OBJECTPROPERTY(OBJECT_ID(N'SalesLT.Product'), 'HasDeleteTrigger');
```

```
SELECT OBJECTPROPERTY(OBJECT_ID(N'SalesLT.Customer'), 'IsUserTable');
```

```
SELECT OBJECT_DEFINITION(OBJECT_ID('dbo.uspLogError'))
```

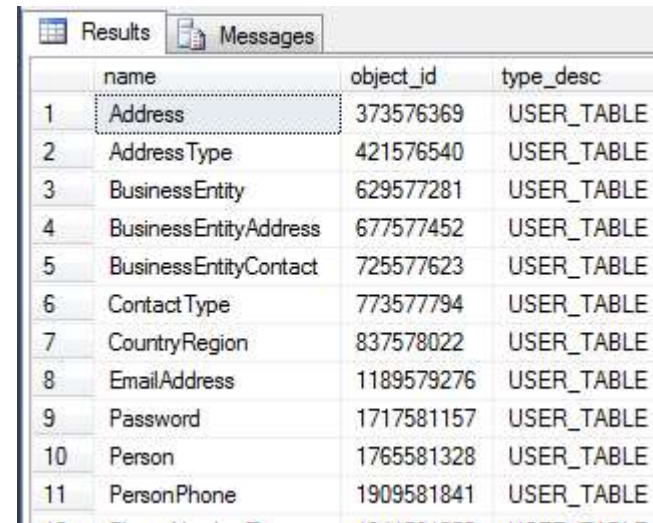
Exemplos e Exercícios

SQL

Exercício

O que faz?

```
SELECT name, object_id, type_desc
FROM sys.all_objects
WHERE OBJECTPROPERTY(object_id, N'SchemaId') = SCHEMA_ID(N'SalesLT')
AND OBJECTPROPERTY(object_id, N'ISTABLE')=1
ORDER BY type_desc, name;
```



The screenshot shows a SQL Server Results window with two tabs: 'Results' and 'Messages'. The 'Results' tab is active, displaying a table with four columns: 'name', 'object_id', and 'type_desc'. The table contains 11 rows of data, all of which are 'USER_TABLE' type. The first row is 'Address' with object_id 373576369. The last row is 'PersonPhone' with object_id 1909581841. The 'name' column is highlighted with a mouse cursor.

	name	object_id	type_desc
1	Address	373576369	USER_TABLE
2	AddressType	421576540	USER_TABLE
3	BusinessEntity	629577281	USER_TABLE
4	BusinessEntityAddress	677577452	USER_TABLE
5	BusinessEntityContact	725577623	USER_TABLE
6	ContactType	773577794	USER_TABLE
7	CountryRegion	837578022	USER_TABLE
8	EmailAddress	1189579276	USER_TABLE
9	Password	1717581157	USER_TABLE
10	Person	1765581328	USER_TABLE
11	PersonPhone	1909581841	USER_TABLE

EXERCÍCIOS COM PC

Exemplos e Exercícios

SQL

Exercício

- Liste todos os campos que são *nullable* da tabela Product

```
SELECT name, is_nullable
FROM sys.columns sc
WHERE object_id = OBJECT_ID(N'SalesLT.Product')
      and is_nullable =1
```

- Liste o tipo de todos os campos da tabela Customer

```
SELECT name ColumnName,
       TYPE_NAME(system_type_id) SystemType
FROM sys.columns
WHERE object_id = OBJECT_ID(N'SalesLT.Customer')
```

Exemplos e Exercícios

SQL

Exercício

- Liste o conjunto de chaves Primária(s) e Estrangeira(s) associadas a uma tabela; e.g. Product

```
SELECT  
DISTINCT  
Constraint_Name AS [Constraint],  
Table_Schema AS [Schema],  
Table_Name AS [TableName]  
FROM INFORMATION_SCHEMA.KEY_COLUMN_USAGE  
WHERE TABLE_NAME='Product'
```

	Constraint	Schema	TableName
1	AK_Product_Name	SalesLT	Product
2	AK_Product_ProductNumber	SalesLT	Product
3	AK_Product_rowguid	SalesLT	Product
4	FK_Product_ProductCategory_ProductCategory...	SalesLT	Product
5	FK_Product_ProductModel_ProductModelID	SalesLT	Product
6	PK_Product_ProductID	SalesLT	Product

Exemplos e Exercícios

SQL

Exercício (alternativa)

- Liste o conjunto de chaves Primária(s) e Estrangeira(s) associadas a uma tabela; e.g. Product

```
SELECT OBJECT_NAME(OBJECT_ID) AS NameofConstraint,  
SCHEMA_NAME(schema_id) AS SchemaName,  
OBJECT_NAME(parent_object_id) AS TableName,  
type_desc AS ConstraintType  
FROM sys.objects  
WHERE type_desc IN  
( 'FOREIGN_KEY_CONSTRAINT', 'PRIMARY_KEY_CONSTRAINT' )  
AND OBJECT_NAME(parent_object_id)='Product'
```

	NameofConstraint	SchemaNa...	TableName	ConstraintType
1	PK_Product_ProductID	SalesLT	Product	PRIMARY_KEY_CONSTRAI...
2	FK_Product_ProductCategory_ProductCategory...	SalesLT	Product	FOREIGN_KEY_CONSTRAI...
3	FK_Product_ProductModel_ProductModelID	SalesLT	Product	FOREIGN_KEY_CONSTRAI...

Exemplos e Exercícios

Exercício

- Liste as tabelas referenciadas pelas Chaves Estrangeira(s) de uma tabela; e.g. SalesLT.Product

```
SELECT
    object_name(parent_object_id) Origem,
    object_name(referenced_object_id) Referenciado,
    name
FROM sys.foreign_keys
WHERE parent_object_id = object_id('SalesLT.Product')
```

	Origem	Referenciado	name
1	Product	ProductCategory	FK_Product_ProductCategory_ProductCategory...
2	Product	ProductModel	FK_Product_ProductModel_ProductModelID

Exemplos e Exercícios

Exercício: O que faz?

```
SELECT
    c.name 'Column Name',
    t.Name 'Data type',
    c.max_length 'Max Length',
    c.precision ,
    c.scale ,
    c.is_nullable,
    ISNULL(i.is_primary_key, 0) 'Primary Key'
FROM sys.columns c
INNER JOIN sys.types t ON c.user_type_id = t.user_type_id
LEFT OUTER JOIN sys.index_columns ic
    ON ic.object_id = c.object_id
    AND ic.column_id = c.column_id
LEFT OUTER JOIN sys.indexes i
    ON ic.object_id = i.object_id AND ic.index_id = i.index_id
WHERE
    c.object_id = OBJECT_ID('MYTABLE')
```

Complementos de Bases de Dados – *Metadata* –

Engenharia Informática

2º Ano / 1º Semestre

Cláudio Miguel Sapateiro

claudio.sapateiro@estsetubal.ips.pt

DSI :: Escola Superior de Tecnologia de Setúbal :: Instituto Politécnico de Setúbal