

TEMA 4.
DISEÑO CONCEPTUAL Y LÓGICO DE BASES DE DATOS.

Material para revisar en clase.


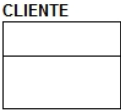
1. FORMATOS EMPLEADOS EN EL DISEÑO CONCEPTUAL Y LÓGICO DE UNA BD.

- Para el modelo conceptual:
 - ** Formato de Chen (Chen's Format). Esto en honor a su creador *Peter Pin-Shan Chen* en 1976.
- Para el modelo lógico:
 - Formato Relacional (Relational Format)
 - ** Formato IE (International Engineering Format) / Formato de Martín (Martin's Format) / Modelo Pata de gallo (Crow's foot Format).
 - ** Formato IDEF1X.
 - Originalmente desarrollado por "The Computer System Laboratory of the National Institute of Standards and Technology " en diciembre del 1993.

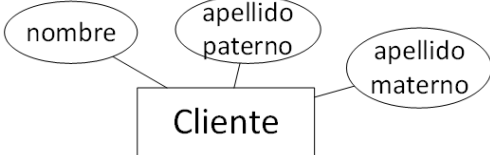
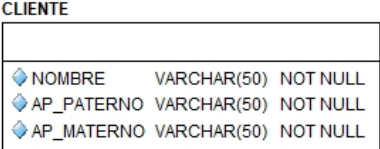
** Estos 3 formatos son los que se emplearán en este capítulo.

2. REPRESENTACIÓN DE ENTIDADES Y ATRIBUTOS.

Representación de Entidades.

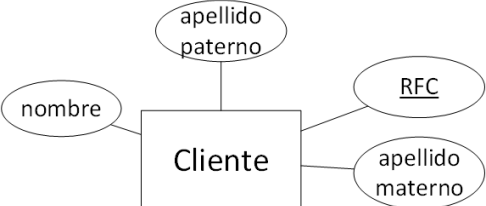
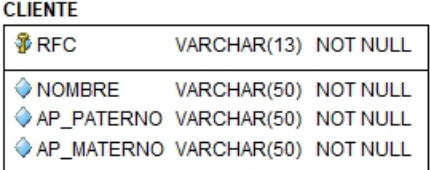
Diseño conceptual	Diseño lógico Crow's Foot e IDEF1X
	

Representación general de atributos.

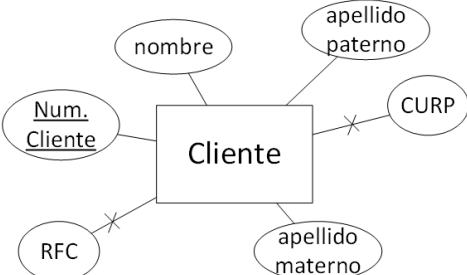
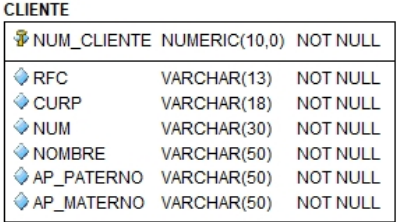
Diseño conceptual	Diseño lógico Crow's Foot e IDEF1X
	

3. CLASIFICACIÓN DE ATRIBUTOS.

Atributos clave y llaves primarias

Diseño conceptual	Diseño lógico Crow's Foot e IDEF1X
	











Claves candidatas y llaves primarias candidatas.

Diseño conceptual	Diseño lógico Crow's Foot e IDEF1X
	











Clave artificial y llave primaria artificial

Diseño conceptual	Diseño lógico Crow's Foot e IDEF1X																
<pre>graph LR Cliente[Cliente] --- nombre((nombre)) Cliente --- apellido_paterno((apellido paterno)) Cliente --- id_cliente((<u>id Cliente</u>)) Cliente --- apellido_materno((apellido materno))</pre>	<div>CLIENTE</div> <table><tr><td></td><td>CLIENTE_ID</td><td>NUMERIC(10,0)</td><td>NOT NULL</td></tr><tr><td></td><td>NOMBRE</td><td>VARCHAR(50)</td><td>NOT NULL</td></tr><tr><td></td><td>AP_PATERNO</td><td>VARCHAR(50)</td><td>NOT NULL</td></tr><tr><td></td><td>AP_MATERNO</td><td>VARCHAR(50)</td><td>NOT NULL</td></tr></table>		CLIENTE_ID	NUMERIC(10,0)	NOT NULL		NOMBRE	VARCHAR(50)	NOT NULL		AP_PATERNO	VARCHAR(50)	NOT NULL		AP_MATERNO	VARCHAR(50)	NOT NULL
	CLIENTE_ID	NUMERIC(10,0)	NOT NULL														
	NOMBRE	VARCHAR(50)	NOT NULL														
	AP_PATERNO	VARCHAR(50)	NOT NULL														
	AP_MATERNO	VARCHAR(50)	NOT NULL														

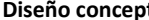






Atributo opcional.

Diseño conceptual	Diseño lógico Crow's Foot e IDEF1X									
	<div>CLIENTE</div> <table><tr><td> CLIENTE_ID</td><td>NUMERIC(10,0)</td><td>NOT NULL</td></tr><tr><td> RFC</td><td>VARCHAR(13)</td><td>NULL</td></tr><tr><td> CURP</td><td>VARCHAR(18)</td><td>NULL</td></tr></table>	 CLIENTE_ID	NUMERIC(10,0)	NOT NULL	 RFC	VARCHAR(13)	NULL	 CURP	VARCHAR(18)	NULL
 CLIENTE_ID	NUMERIC(10,0)	NOT NULL								
 RFC	VARCHAR(13)	NULL								
 CURP	VARCHAR(18)	NULL								

Atributo requerido.

Diseño conceptual	Diseño lógico Crow's Foot e IDEF1X												
	<div>CLIENTE</div> <table><tr><td></td><td>CLIENTE_ID</td><td>NUMERIC(10,0)</td><td>NOT NULL</td></tr><tr><td></td><td>RFC</td><td>VARCHAR(13)</td><td>NOT NULL</td></tr><tr><td></td><td>CURP</td><td>VARCHAR(18)</td><td>NOT NULL</td></tr></table>		CLIENTE_ID	NUMERIC(10,0)	NOT NULL		RFC	VARCHAR(13)	NOT NULL		CURP	VARCHAR(18)	NOT NULL
	CLIENTE_ID	NUMERIC(10,0)	NOT NULL										
	RFC	VARCHAR(13)	NOT NULL										
	CURP	VARCHAR(18)	NOT NULL										

Atributo simple

Diseño conceptual	Diseño lógico Crow's Foot e IDEF1X						
	<div>CLIENTE</div> <table><tr><td> CLIENTE_ID</td><td>NUMERIC(10,0)</td><td>NOT NULL</td></tr><tr><td> COLONIA</td><td>VARCHAR(100)</td><td>NOT NULL</td></tr></table>	 CLIENTE_ID	NUMERIC(10,0)	NOT NULL	 COLONIA	VARCHAR(100)	NOT NULL
 CLIENTE_ID	NUMERIC(10,0)	NOT NULL					
 COLONIA	VARCHAR(100)	NOT NULL					

Atributo compuesto.

Diseño conceptual	Diseño lógico Crow's Foot e IDEF1X
	Atributo compuesto ¿Qué opciones habrá para el diseño Lógico? (Ver en Pizarrón).

Atributos derivados.

Diseño conceptual	Diseño lógico Crow's Foot e IDEF1X									
	<div>ESTUDIANTE</div> <table><tr><td> ESTUDIANTE_ID</td><td>NUMERIC(10,0)</td><td>NOT NULL</td></tr><tr><td> FECHA_NACIMIENTO</td><td>DATE</td><td>NOT NULL</td></tr><tr><td> EDAD</td><td>NUMERIC(3,0)</td><td>NOT NULL</td></tr></table>	ESTUDIANTE_ID	NUMERIC(10,0)	NOT NULL	FECHA_NACIMIENTO	DATE	NOT NULL	EDAD	NUMERIC(3,0)	NOT NULL
ESTUDIANTE_ID	NUMERIC(10,0)	NOT NULL								
FECHA_NACIMIENTO	DATE	NOT NULL								
EDAD	NUMERIC(3,0)	NOT NULL								

Atributo de valor simple.

Diseño conceptual	Diseño lógico Crow's Foot e IDEF1X						
<pre>graph TD; E([Estudiante]) --- P([<u>Num. estudiante</u>]); E --- F([fecha nacimiento]);</pre>	<p>ESTUDIANTE</p> <table><tr><td> ESTUDIANTE_ID</td><td>NUMERIC(10,0)</td><td>NOT NULL</td></tr><tr><td> FECHA_NACIMIENTO</td><td>DATE</td><td>NOT NULL</td></tr></table>	ESTUDIANTE_ID	NUMERIC(10,0)	NOT NULL	FECHA_NACIMIENTO	DATE	NOT NULL
ESTUDIANTE_ID	NUMERIC(10,0)	NOT NULL					
FECHA_NACIMIENTO	DATE	NOT NULL					

Atributo de valores múltiples.

Diseño conceptual	Diseño lógico Crow's Foot e IDEF1X
	<ul style="list-style-type: none"> En el modelo relacional un atributo no puede tener múltiples valores para un mismo registro. Para resolver este caso se emplean diversas estrategias. Ver en pizarrón.

4. REPRESENTACIÓN DE RELACIONES ENTRE ENTIDADES.

Revisar en Pizarrón.

5. TIPOS DE DATOS EMPLEADOS EN MODELOS RELACIONALES.

- Antes de iniciar con el modelado de casos de estudio, el último concepto a revisar es el asociado con los tipos de datos. Cuando se agrega cada una de las columnas de una tabla es indispensable indicar el tipo de dato que debe contener la columna. Adicional a las restricciones que pudiera tener una columna en una tabla, es importante definir el tipo de dato y su dominio. Dicha definición debe reflejar de forma correcta la naturaleza de los datos: *valores monetarios, fechas, cantidades, nombres*, etc.
- Para implementar este requerimiento, el estándar SQL ha definido una serie de tipos de datos que todo manejador debe implementar. La realidad es que no todos los manejadores cumplen al 100% con el estándar. Algunos definen variantes o sus propios tipos de datos.

Clasificación de los tipos de datos SQL:

- Cadenas
 - Cadenas de caracteres de longitud fija
 - Cadenas de caracteres de longitud variable
 - Cadenas binarias
- Numéricos
 - Exactos
 - Aproximados
- Tiempo y fechas
- Otros:
 - boolean (Estándar, MySQL, PostgreSQL)
 - ROWID (Oracle), UROWID (Oracle)
 - BFILE (Oracle)
 - XML

Cadenas de caracteres

[Conjunto o secuencia de caracteres que pertenecen a un “Juego de caracteres” predefinido. La longitud de la cadena está definida por el número de caracteres que contiene. El número de bytes de espacio requerido para almacenar una cadena depende del **juego de caracteres** con el que se represente. Internamente, cada carácter es representado generalmente por un valor numérico que se obtiene a través de la aplicación de algún algoritmo de codificación. Algunas normas de codificación (juego de caracteres):

ISO 646	ASCII	UNICODE	ISO 8859	Juego de caracteres de Windows	UTF8
---------	-------	---------	----------	--------------------------------	------

Ejemplos:

Carácter	ISO-8859-1	UTF-8	UTF-16	
a	0x61	0x61	0x00	0x61
b	0x62	0x62	0x00	0x62

Uno de los puntos importantes al momento de instalar una base de datos, es la asignación del **juego de caracteres** que se empleará para almacenar los datos. Esto permitirá la correcta codificación, representación y almacenamiento de las cadenas de caracteres en la base de datos. Dependiendo de la diversidad de símbolos y caracteres que puedan ser enviados a la base de datos, se deberá elegir el juego de caracteres a configurar. Por default, se selecciona el juego de caracteres que emplea el sistema operativo.

Cadenas de caracteres de longitud fija:

Si un atributo en la BD se define como una cadena de longitud fija N, como máximo se podrán insertar N caracteres. Si la cadena es menor a N, esta se rellena con espacios. Ejemplo: Cadena de caracteres de longitud 8 (fija).

A	R	P	A				
---	---	---	---	--	--	--	--

M	U	S	I	C	A		
---	---	---	---	---	---	--	--

P	I	A	N	O			
---	---	---	---	---	--	--	--

El tipo de dato que representa a una cadena de longitud fija es **CHAR**. (Ver tabla siguiente). Sintaxis: **CHAR** [ACTER] [(n)]

El contenido que esta entre [], es opcional. El valor de “n” representa al número de caracteres de la cadena. Si no se especifica, es equivalente a **CHAR (1)**, un carácter. Ejemplos:

Ejemplo	Descripción
CHAR (10)	Cadena de caracteres de longitud 10
CHAR	Cadena de caracteres de longitud 1

Cadenas de caracteres de longitud variable.

A diferencia de las cadenas de longitud fija, solo se almacena el número real de caracteres que contiene la cadena. A nivel de definición del campo solo se define la capacidad máxima.

Ejemplo: cadena de caracteres de longitud 8 (variable)

A	R	P	A
---	---	---	---

M	U	S	I	C	A
---	---	---	---	---	---

P	I	A	N	O
---	---	---	---	---

El tipo de dato que representa a una cadena de longitud fija es **VARCHAR**. (Ver tabla siguiente). Sintaxis: **VARCHAR** (n)

En este caso, “n” define el número máximo de caracteres que puede tener la cadena: **VARCHAR (1)**, **VARCHAR (100)**, etc.

Para el caso particular de Oracle, se recomienda emplear **VARCHAR2** (este será el tipo de dato a emplear en el curso para las prácticas de SQL).

Cadenas de caracteres de gran tamaño.

En algunas ocasiones no se conoce con precisión el tamaño máximo que pudiera tener una cadena a insertar en la base de datos. Lo único que se conoce es que pueden ser cadenas muy grandes. Para estos casos, existe en tipo de dato **CLOB** (Character Large Object).

Se emplea para almacenar grandes cantidades de caracteres, textos, artículos, contenido de un libro, etc.

Cadenas de caracteres en diferente juego de caracteres.

Los tipos de datos **NCHAR** Y **NCHAR VARYING (N)**, **NVARCHAR2** (para Oracle) se emplean para guardar cadenas representadas en un juego de caracteres diferente al configurado en la base de datos. Para el caso de Oracle, se emplea como juego de caracteres **UNICODE**, aunque el estándar define **CHARACTER VARYING (n) CHARACTER SET <name>** en el que se especifica el nombre del juego de caracteres a emplear, Oracle siempre emplea **UNICODE**. La principal razón: **UNICODE** es el juego de caracteres universal, por lo que se puede representar cualquier carácter.

Clasificación de los tipos de datos para representar cadenas de caracteres

La siguiente tabla muestra la clasificación de las cadenas de caracteres. Del lado izquierdo, representa al tipo de dato definido por el estándar SQL, y las demás columnas representan la forma en la que cada manejador implementa al tipo de dato.

SQL 2003 (especificación)	ORACLE 11g	DB2 9.5	SQL Server 2008	PostgreSQL 8.x	MySQL 5.x
CHAR [ACTER] [(n)]	CHAR [ACTER] [(n)]	CHAR [ACTER] [(n)]	CHAR [ACTER] [(n)]	CHAR [ACTER] [(n)]	CHAR [(M)]
CHAR [ACTER] VARYING (n) ó VARCHAR (n)	CHAR [ACTER] VARYING (n) ó VARCHAR (n) ó VARCHAR2 (n)	CHAR [ACTER] VARYING (n) ó VARCHAR (n) ó LONGVARCHAR	CHAR [ACTER] VARYING (n) ó VARCHAR [(n)] TEXT	CHAR [ACTER] VARYING (n) ó VARCHAR (n)	VARCHAR (M) ó TINYTEXT ó TEXT ó MEDIUMTEXT ó LONGTEXT
CLOB ó CHARACTER LARGE OBJECT	CLOB ó LONG [VARCHAR]	CLOB [(n) [K M,G]]	VARCHAR (MAX)	TEXT	BINARY [(M)] VARBINARY (M)

SQL 2003 (especificación)	ORACLE 11g	DB2 9.5	SQL Server 2008	PostgreSQL 8.x	MySQL 5.x
NATIONAL CHAR[ACTER] [(n)] ó NCHAR [(n)] ó CHARACTER [(n)] CHARACTER SET <charset-name>	NATIONAL CHAR[ACTER] [(n)] ó NCHAR [(n)]	GRAPHIC [(n)]	NATIONAL CHAR[ACTER] [(n)] ó NCHAR [(n)] ó NTEXT	-----	NATIONAL CHARACTER (n) ó CHAR (n) CHARACTER SET <name> ó NCHAR (n)
NATIONAL CHAR[ACTER] VARYING (n) ó NCHAR VARYING (n) ó CHARACTER VARYING (n) CHARACTER SET <name>	NATIONAL CHAR[ACTER] VARYING (n) ó NCHAR VARYING (n) ó NCHARACTER2 (n)	VARGRAPHIC (n) ó LONG VARGRAPHIC (n)	NATIONAL CHARACTER VARYING [(n)] ó NTEXT	-----	VARCHAR (n) CHARACTER SET <name> ó NATIONAL VARCHAR (n) ó NCHAR VARCHAR (n) ó NATIONAL CHAR[ACTER] VARYING (n)
NATIONAL CHARACTER LARGE OBJECT	NCLOB	DBCLOB [(n) [K M G]]	NVARCHAR (MAX)	-----	-----

- Las palabras entre [] son opcionales, las palabras entre <> se sustituyen por un valor.

Rangos soportados por los principales manejadores para cadenas de caracteres

Tipo de dato	Oracle	Tipo de dato	DB2	Tipo de dato	SQL Server	Tipo de dato	MySQL
CHAR	1-2000	CHAR	1-254	CHAR	1-8000	CHAR	1-255
VARCHAR2	1-4000	VARCHAR	1-32672	VARCHAR	1-8000	VARCHAR	1-65535
NCHAR (unicode)	1-2000	LONG VARCHAR	1-32700	TEXT	2 GB	TINYTEXT	0-255
NVARCHAR (unicode)	1-4000	CLOB	2 GB	NCHAR NVARCHAR	1-4000	TEXT	0-65535
CLOB	8 TB	GRAPHIC	1-127	NTEXT	1 GB	MEDIUM TEXT	0-16777215
NCLOB	8 TB	VARGRAPHIC	1-16, 336	VARCHAR (MAX) NVARCHAR	2 GB	LONGTEXT	0- 4294967295
		LONG VARGRAPHIC	16, 350				

Tipo de dato	PostgreSQL
CHAR	1 GB
VARCHAR	1-GB
TEXT	1-GB

Cadenas binarias

Se emplean para almacenar secuencia de bytes en la base de datos, es decir, se almacenan archivos binarios: fotos, música, videos, huellas, documentos, etc. El tipo de dato más común empleado es BLOB (Binary Large Object), de forma similar a CLOB, no es necesario escribir la longitud máxima que puede almacenarse en la BD.

Clasificación:

SQL 2003 (especificación)	ORACLE 11g	DB2 9.5	SQL Server 2008	PostgreSQL 8.x	MySQL 5.x
BLOB ó BINARY LARGE OBJECT	BLOB ó LONGRAW ó RAW (n)	BLOB [(n) [K M G]]	VARBINARY (MAX) ó VARBINARY [(n)] ó IMAGE	BYTEA	BINARY [(n)] ó VARBINARY (M) ó TINYBLOB ó BLOB ó MEDIUM BLOB ó LONG BLOB

Rangos cadenas binarias.

Tipo de dato	Oracle	Tipo de dato	DB2	Tipo de dato	SQL Server
BLOB	8 TB	BLOB	2 GB	BINARY	8000
RAW	4000 BYTES			VARBINARY	8000
				IMAGE	2147483647
				VARBINARY (MAX)	2 GB

¡Observar que en el caso de Oracle es posible almacenar hasta un archivo de 8 TB!

Tipos de datos numéricos.

Existen 2 clasificaciones de los tipos de datos numéricos:

- Tipos de datos numéricos exactos
- Tipos de datos numéricos aproximados.

Tipos de datos numéricos exactos

Los tipos de datos numéricos exactos pueden ser números enteros finitos o números con parte decimal finita.

Al definir un atributo de una tabla con un tipo de dato numérico exacto se puede definir 2 elementos:

- Precisión: Número total de dígitos: Parte entera + parte decimal
- Escala: Número de dígitos que formarán a la parte decimal.

Por lo anterior, el número máximo de dígitos que podrá tener la parte entera de un valor numérico será:

#dígitosParteEntera = precisión- escala.

Ejemplo.

¿Cuál será el dominio de la siguiente columna? PRECIO NUMBER(10,4)

Precision = 10

Escala = 4

Parte entera = 10-4 =6

Dominio: [0,999999.9999]

Clasificación de los tipos de datos numéricos exactos.

SQL 2003 (especificación)	ORACLE 11g	DB2 9.5	SQL Server 2008	PostgreSQL 8.x	MySQL 5.x
INT[EGER]	NUMBER (n)	INT[EGER] ó BIGINT	INT[EGER] ó BIGINT	INTEGER ó BIGSERIAL ó SERIAL ó BIGINT	INT[(M)] ó MEDIUMINT[(M)] ó BIGINT[(M)]
SMALLINT	SMALLINT NUMBER (n)	SMALLINT	SMALLINT ó TINYINT	SMALLINT	SMALLINT[(M)] ó TINYINT[(M)]
NUMERIC[(P[,S])] ó DEC[IMAL] [(P[,S])] ó DEC[IMAL] [(P[,S])]	NUMERIC [(P[,S])] ó DEC[IMAL] [(P[,S])] ó NUMBER[(P[,S])]	NUMERIC [(P[,S])] ó DEC[IMAL] [(P[,S])]	NUMERIC [(P[,S])] ó DEC[IMAL] [(P[,S])] MONEY SMALLMONEY	NUMERIC [(P[,S])] ó DEC[IMAL] [(P[,S])]	

Tipos de datos numéricos aproximados.

Se emplean cuando no se conoce con exactitud los valores de la precisión y/o escala que puede tener los valores de una columna.

Clasificación de los tipos de datos numéricos aproximados.

SQL 2003 (especificación)	ORACLE 11g	DB2 9.5	SQL Server 2008	PostgreSQL 8.x	MySQL 5.x
FLOAT[(P)]	FLOAT[(P)] ó NUMBER	FLOAT[(P)]	FLOAT[(P)]	-----	FLOAT[(P[,S])]
REAL	REAL NUMBER	REAL	REAL	REAL	REAL[(P[,S])]
DOUBLE PRECISION	DOUBLE PRECISION NUMBER	DOUBLE [PRECISION]	DOUBLE PRECISION	DOUBLE PRECISION	DOUBLE [PRECISION]

Rangos tipos de datos numéricos para Oracle:

- INTEGER, SMALLINT se convierten a NUMBER (38)
- NUMERIC, DECIMAL se convierten a NUMBER
- Rango: 1×10^{-130} a 9×10^{125} (38 nueves).
- Oracle permite la definición de escalas negativas empleadas para redondear. Por ejemplo:
 - NUMBER(10,2), 6,345,768,903.678 será redondeado a 6,345,768,904

Observar que, para el caso de Oracle, en todos los tipos que define el estándar, Oracle lo representa a través del tipo de dato NUMBER. Por lo anterior, se recomienda que todos los tipos de datos numéricos (tanto exactos como aproximados) sean representados empleando NUMBER.

Tipos de datos para el manejo de fechas.

Esta clasificación de tipos de dato es la más complicada en el sentido de que prácticamente ningún manejador sigue el estándar SQL:

A nivel del estándar:

- DATE representa una fecha hasta el nivel de días (Año, Mes, Día)
- TIME Representa tiempo a nivel de HH:MM:SS
- TIMESTAMP Representa una combinación: Año, Mes, Día, Hora, Minuto, Segundo, Milisegundo

Es importante recalcar que, en la mayoría de las aplicaciones o clientes gráficos empleados para consultar los datos, estos emplean un formato predefinido para representar a una fecha. Ejemplos:

- 02/03/08
- 02-03-2008
- 02 de marzo de 2008, etc.

Lo anterior no significa que en la base de datos una fecha se almacene como una cadena con un determinado formato. Normalmente las fechas se almacenan internamente como datos numéricos.

Clasificación de los tipos de datos para manejo de fechas.

SQL Estándar (especificación)	ORACLE 11g	DB2 9.5	SQL Server 2008	PostgreSQL 8.x	MySQL 5.x
DATE	DATE	DATE	DATETIME ó SMALLDATETIME	DATE	DATE
TIME [WITH TIME ZONE]	DATE	TIME	DATETIME ó SMALLDATETIME	TIME [(P)] [WITH [OUT] TIMEZONE]	TIME
TIMESTAMP [(P)] [WITH TIME ZONE]	DATE ó TIMESTAMP [(P)] [WITH TIME ZONE]	TIMESTAMP	DATETIME SMALLDATETIME	TIMESTAMP [(P)] [WITH [OUT] TIMEZONE]	DATETIME TIMESTAMP
INTERVAL	INTERVAL DAY [(P)] TO SECOND [(P)] ó INTERVAL YEAR [(P)] TO MONTH	-----	-----	INTERVAL [fields] [(p)] Fields: YEAR, MONTH, DAY, HOUR, etc.	YEAR

Observar nuevamente, para el caso de Oracle, se emplea el tipo de dato DATE para representar fechas, tiempo o ambas.

Otros tipos de datos.

BOOLEAN

- Se agrega al estándar SQL 2003. Posibles valores para este tipo de dato: TRUE, FALSE.
- Oracle, DB2, SQL server no tienen este tipo de dato. Para el caso de Oracle se emplea NUMBER (1, 0) . Típicamente 1 significa TRUE, 0 significa FALSE.
- En MySQL: BOOL, BOOLEAN, TINYINT (1)
- En PostgreSQL: BOOLEAN

ROWID: Como se revisó en temas anteriores, ROWID es exclusivo de Oracle, se emplea para almacenar una dirección única para cada registro de una tabla que se crea para localizar los datos de una forma directa.

UROWID: Similar a ROWID, solo que este se emplea para tablas indexadas.

BFILE: Tipo dato exclusivo de Oracle, se emplea para leer datos binarios que están almacenados fuera de la base de datos (se considera una especie de puntero hacia la ubicación física del archivo).

XML: Tipo de dato empleado para almacenar documentos XML. Permite el manejo y explotación directa del documento a través de SQL en combinación de lenguaje empleado para navegar sobre los elementos de un documento XML.

6. MODELADO DE CASOS DE ESTUDIO

Base de datos para una empresa que administra colegios.

Las siguientes reglas de negocio resultaron de las entrevistas con los dueños de una empresa que administra colegios particulares la cual ha solicitado el diseño de una base de datos.

- Al final se anexa una propuesta de diseño conceptual. Completar el diagrama indicando los nombres de las entidades en los espacios correspondientes.
- Realizar el diseño lógico empleando para ello la transformación del modelo E/R del punto anterior a un modelo relacional con notación **crow's foot**.
 - Cada colegio está formado por varias escuelas. De cada colegio se requiere almacenar su nombre.
 - Para cada escuela se requiere almacenar el nombre y su clave. Cada escuela es administrada por un director el cual a su vez es un profesor. Cada director puede administrar una sola escuela. Para un profesor no es requisito que administre una escuela.
 - Cada escuela está formada por varios departamentos. Se requiere almacenar el nombre del departamento.
 - Cada departamento esta dividido en un conjunto de áreas funcionales. Se desea almacenar los nombres de las áreas funcionales del departamento.

- Cada departamento ofrece cursos; otros departamentos se clasifican, por ejemplo, como departamentos de investigación por lo que no ofrecen cursos. Se debe almacenar el tipo de departamento ("de investigación", o "regular"). Cuando se registra el curso se debe indicar en nombre del curso, el cupo máximo, número de alumnos inscritos y el departamento al que pertenece.
- Cada curso se imparte en varias clases a la semana. Para cada clase se indica la hora inicio, hora fin, el día de la semana y el salón (un mismo curso puede impartirse en varios salones).
- Para cada curso, en una semana se pueden dar como mínimo una clase, y como máximo 5 clases. El curso lo imparte un único profesor.
- El colegio requiere guardar los siguientes datos de cada salón: clave de 5 caracteres, y cupo máximo. No en todos los salones se imparten cursos.
- Para contar con una mejor comprensión de un curso, en algunos casos se recomienda tomar un curso previo. Para dichos casos se requiere asociar el curso recomendado.
- Cada profesor pertenece a un departamento. Uno de estos profesores es el encargado de administrarlo (Jefe de departamento). Los datos que se registran de un profesor son: Nombre, apellidos, y RFC. Algunos profesores no imparten cursos.
- Un estudiante puede estar inscrito en 1 y hasta en 6 cursos en un mismo periodo. Cada curso puede tener como mínimo 5 estudiantes, y como máximo 35. Al final del periodo se registra la calificación del estudiante. Se requiere almacenar nombre, apellidos del estudiante y número de matrícula. El estudiante debe proporcionar al menos un correo electrónico como medio de contacto.
- Cada estudiante pertenece a una carrera. El colegio cuenta con un catálogo de carreras en el que se almacena clave de la carrera, nombre de la carrera, y total de créditos.
- Cada estudiante puede tener un asesor, el cual es un profesor. Solo aquellos profesores que así lo deseen pueden asesorar hasta 5 estudiantes.
- Cada clase se imparte en un salón, y cada salón se encuentra localizado en un edificio perteneciente a la escuela. Se requiere almacenar los datos del edificio: Clave de 2 letras, y descripción. No en todos los edificios se imparten cursos.
- Cada año un asesor puede recibir varios bonos por su buen desempeño. Se requiere registrar la fecha de pago, el monto del bono y un número consecutivo (folio de pago) que indica el número de pago. El folio está formado por 5 dígitos. Ejemplo: 00001 para el primer pago, 0002 para el segundo pago, etc.

