UNIVERSIDAD
AUSTRAL

SOMOSAUSTRAL

Ingeniería Inversa

Modelo Entidad/Relación

Interpretación de una Base de Datos a partir de su estructura física

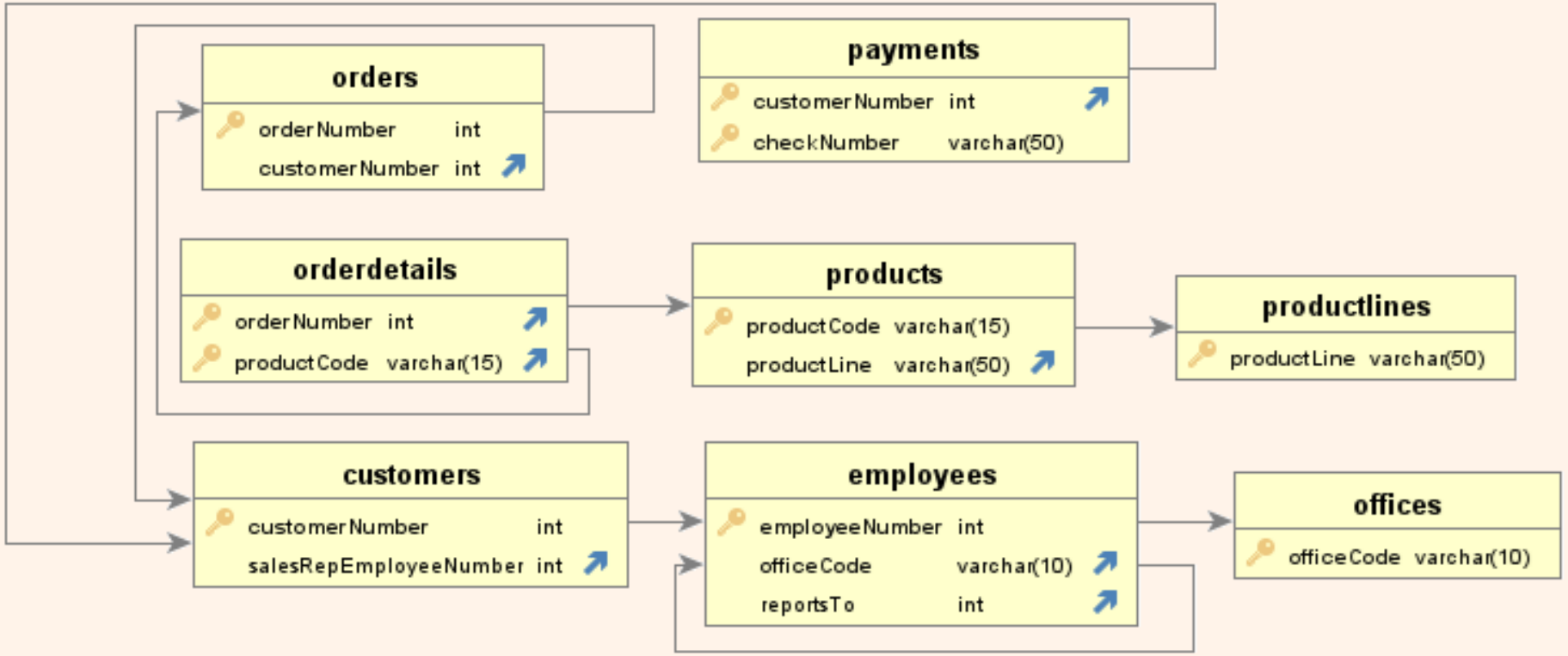
1. Distinguir Tablas “entidades” de Tablas “relaciones N:N”
2. Encontrar e interpretar claves primarias y secundarias
3. Reconocer cardinalidad
4. Interpretar atributos a partir de su nombre y dominio

Análisis de Datos

1. Chequear interpretaciones, realizar cálculos, sacar estadísticas…
2. Crear vistas del tipo “una fila por caso”

Ingeniería Inversa

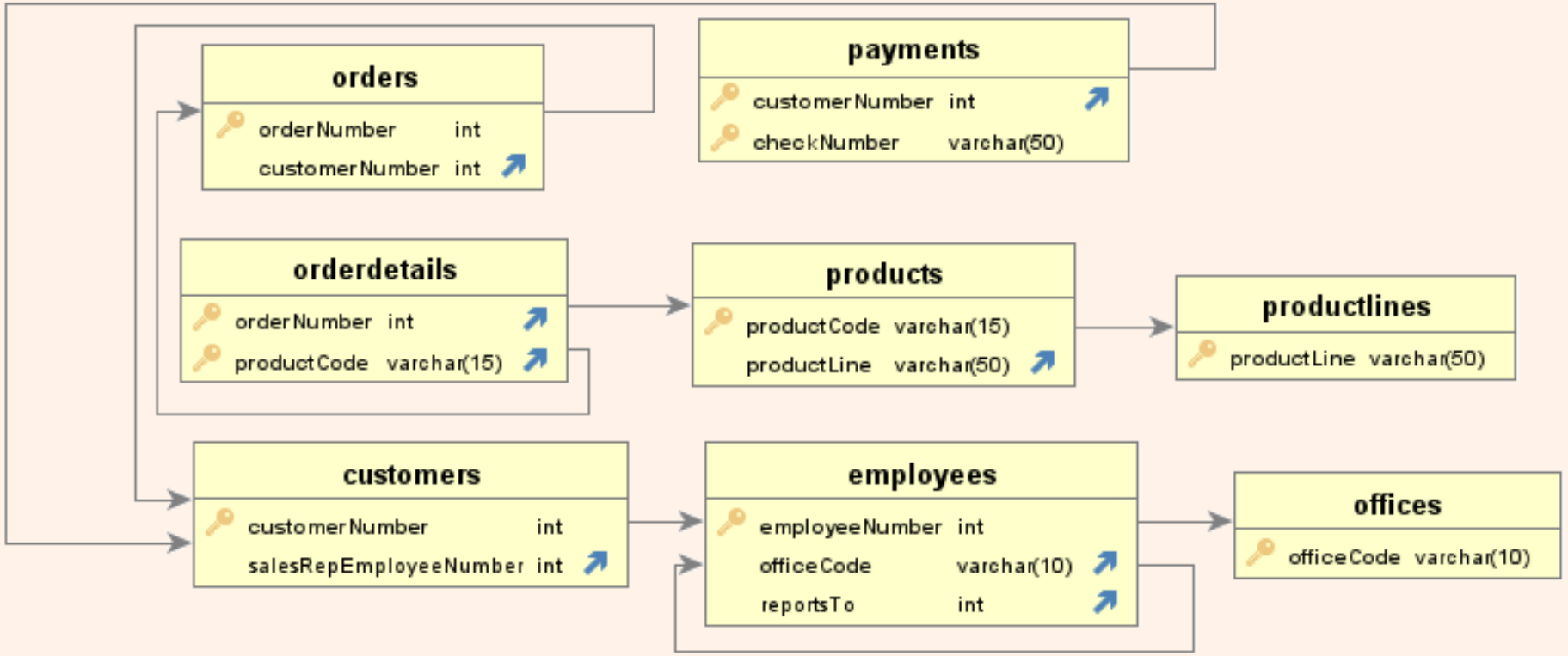
Ejemplo 1



DB: classicmodels
By: DbVisualizer

Ingeniería Inversa

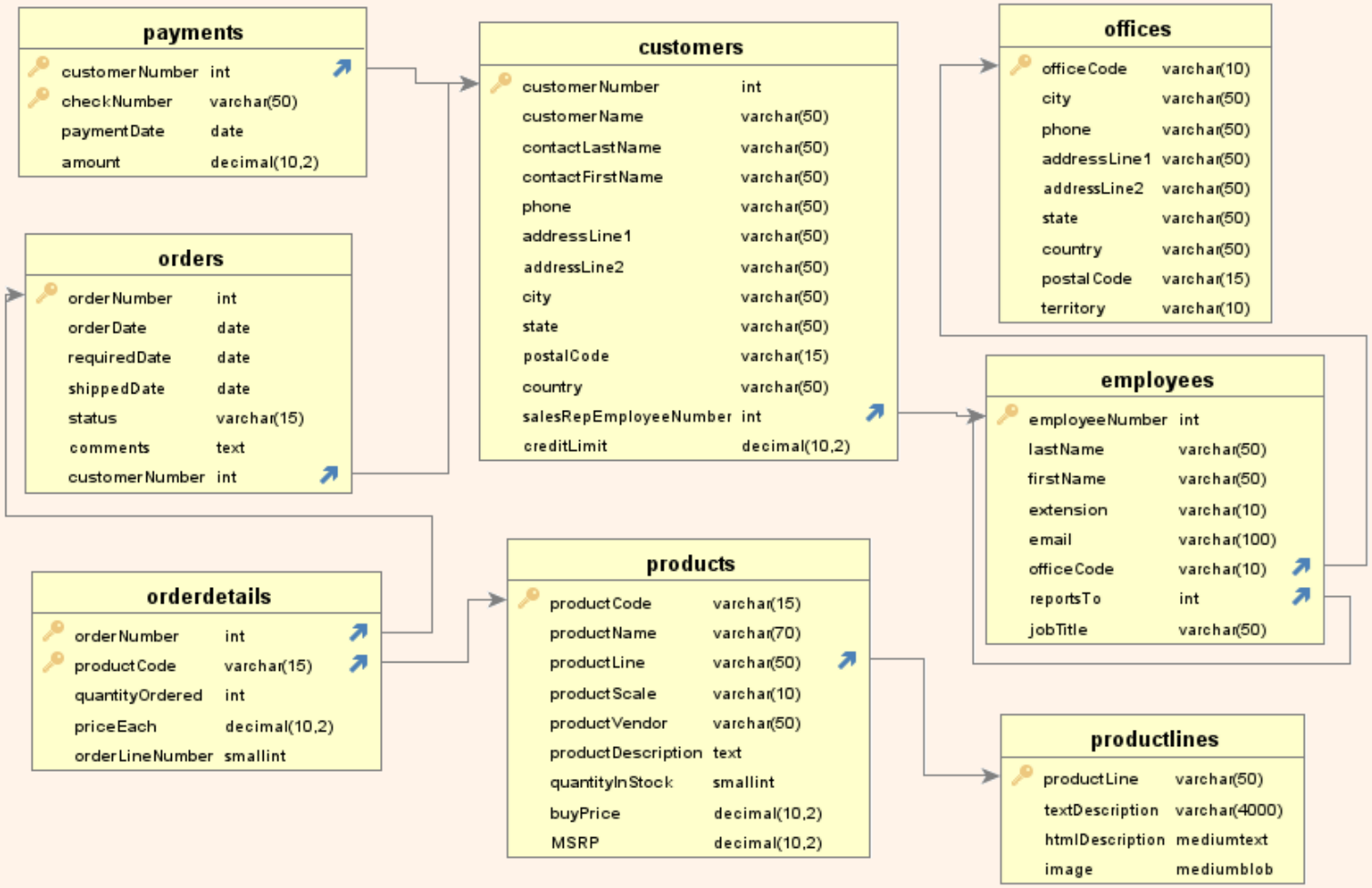
Ejemplo 1 – sólo claves



DB: classicmodels
By: DbVisualizer

Ingeniería Inversa

Ejemplo 1 – detallado



DB: classicmodels
By: DbVisualizer

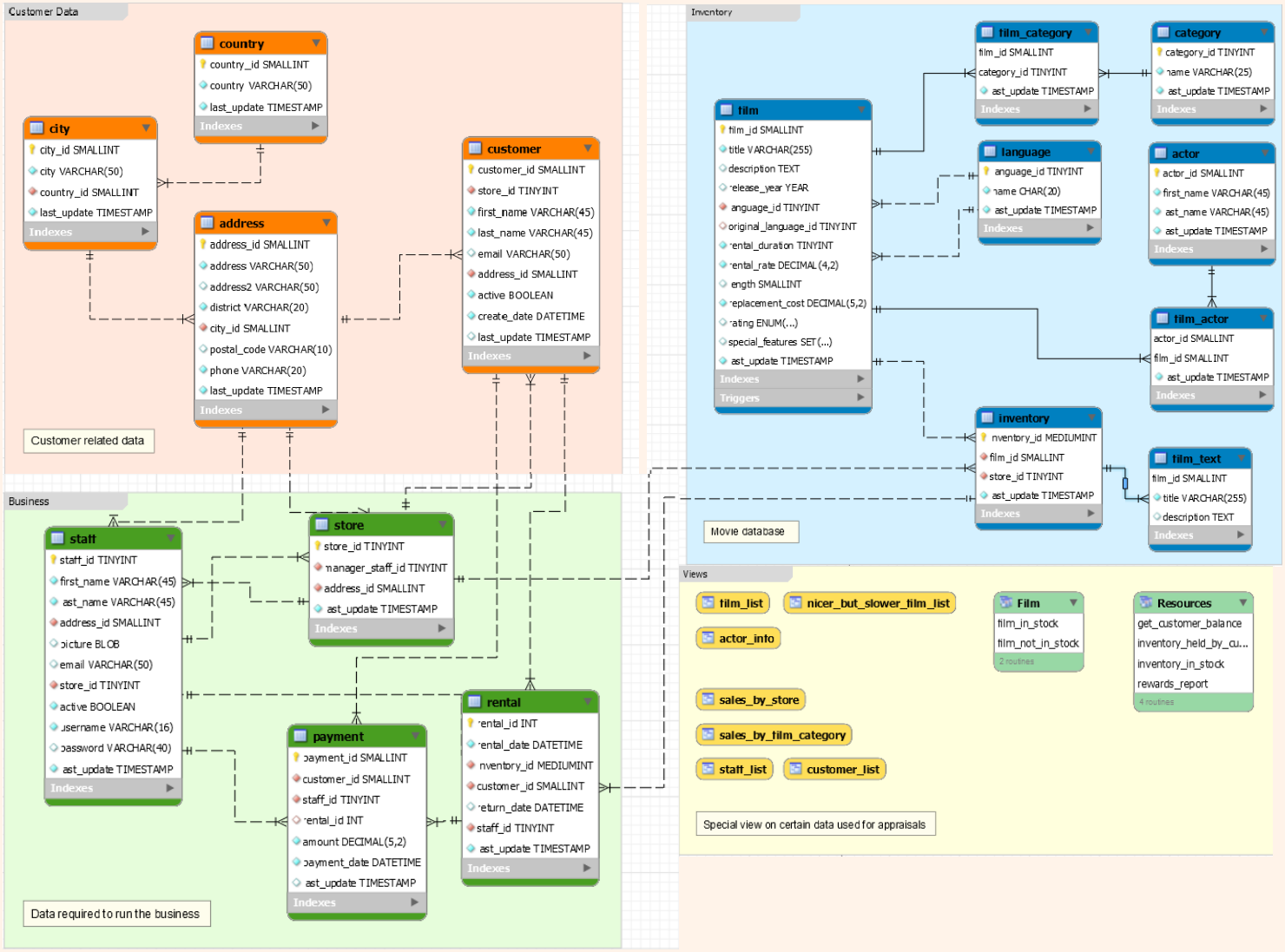
Ingeniería Inversa

Ejemplo 1 – Detalle tabla

```
CREATE TABLE `customers` (
      `customerNumber` int(11) NOT NULL,
      `customerName` varchar(50) NOT NULL,
      `contactLastName` varchar(50) NOT NULL,
      `contactFirstName` varchar(50) NOT NULL,
      `phone` varchar(50) NOT NULL,
      `addressLine1` varchar(50) NOT NULL,
      `addressLine2` varchar(50) DEFAULT NULL,
      `city` varchar(50) NOT NULL,
      `state` varchar(50) DEFAULT NULL,
      `postalCode` varchar(15) DEFAULT NULL,
      `country` varchar(50) NOT NULL,
      `salesRepEmployeeNumber` int(11) DEFAULT NULL,
      `creditLimit` decimal(10,2) DEFAULT NULL,
      PRIMARY KEY (`customerNumber`),
      KEY `salesRepEmployeeNumber` (`salesRepEmployeeNumber`),
      CONSTRAINT `customers_ibfk_1` FOREIGN KEY
(`salesRepEmployeeNumber`) REFERENCES `employees`
(`employeeNumber`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

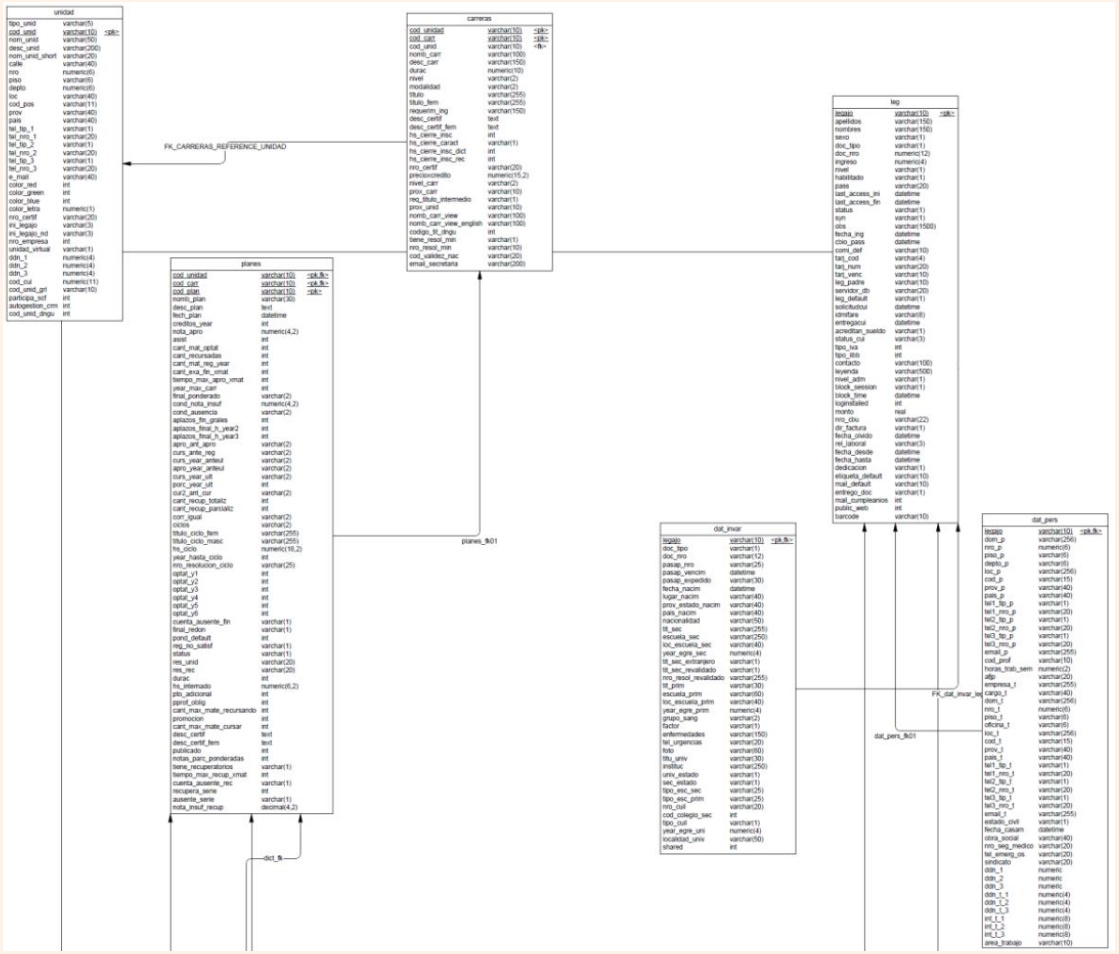
Ingeniería Inversa

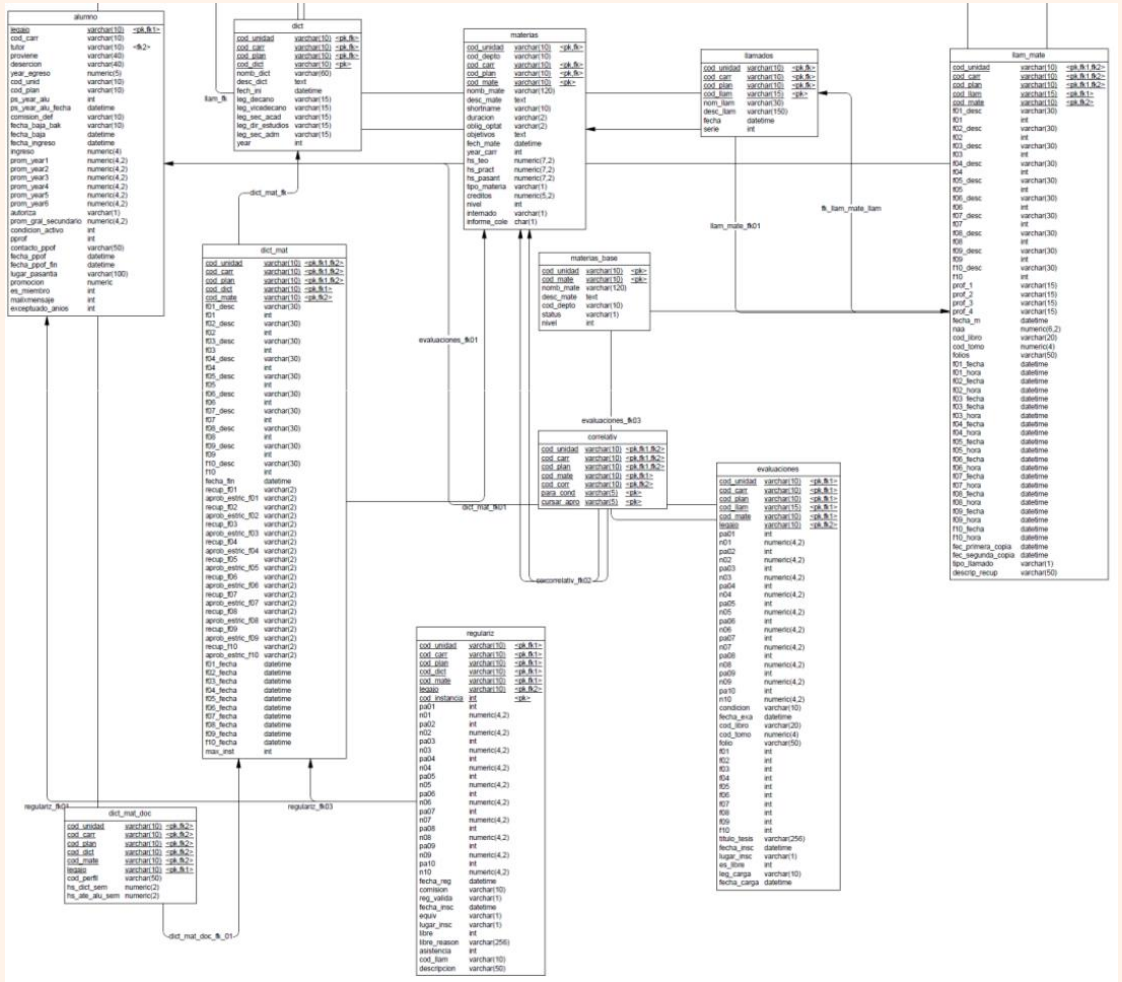
Ejemplo 2 – modelo nivel medio



Ingeniería Inversa

Ejemplo 3 – modelo nivel maestría





SOMOSAUSTRAL

Structured Query Language

Programar “consultas”

UNIVERSIDAD AUSTRAL

INGENIERÍA

Introducción a SQL (Structure Query Language)

UNIVERSIDAD AUSTRAL
INGENIERÍA

Lenguaje de consulta estructurado, es un lenguaje informático de base de datos diseñado para la gestión de datos en sistemas de gestión de bases de datos relacionales (RDBMS), y originalmente basado en álgebra relacional y cálculo. Su alcance incluye inserción, consulta, actualización y eliminación de datos, creación de esquemas y modificación y control de acceso a datos.

Introducción a SQL

UNIVERSIDAD AUSTRAL
INGENIERÍA

¿Qué es SQL?

- SQL son las siglas de Structured Query Language
- SQL le permite acceder y manipular bases de datos
- SQL es un estándar ANSI (American National Standards Institute)

¿Qué puedo hacer con SQL?

- SQL puede manipular los datos de la base de datos (DML)
- SQL puede definir las estructuras de los datos de la base de datos (DDL)
- SQL puede controlar los permisos de acceso a la base de datos (DCL)

¿Qué puedo hacer con SQL? Detallado

UNIVERSIDAD AUSTRAL
INGENIERÍA

- Ejecutar consultas contra una base de datos
- Recuperar datos de una base de datos
- Insertar registros en una base de datos
- Actualizar registros en una base de datos
- Eliminar registros de una base de datos
- Crear nuevas bases de datos
- Crear nuevas tablas en una base de datos
- Crear stored procedures en una base de datos
- Crear vistas en una base de datos
- Definir permisos en tablas, procedimientos y vistas

Data Manipulation Language (DML)

Consulta:

- **SELECT**: extraer datos de una base de datos

Modificación:

- **UPDATE**: actualizar los datos en una base de datos
- **DELETE**: eliminar datos de una base de datos
- **INSERT** INTO: insertar nuevos datos en una base de datos

Data Definition Language (DDL)

UNIVERSIDAD AUSTRAL
INGENIERÍA

- **CREATE**: Crea
- **ALTER**: Modifica
- **DROP**: Elimina

Se aplican a:

- DATABASE
- TABLE
- INDEX*
- VIEW *
- CONSTRAINT *

(*) Sólo CREATE y DROP

Data Control Language (DCL)

UNIVERSIDAD AUSTRAL
INGENIERÍA

- GRANT: autoriza a uno o más usuarios a realizar una operación o conjunto de operaciones sobre un objeto.
- REVOKE: elimina una concesión, que puede ser la concesión predeterminada.

SOMOSAUSTRAL

Sentencias SQL

```
SELECT      EmployeeID, FirstName, LastName, HireDate, C
FROM        Employees
WHERE       HireDate BETWEEN '1-june-2012' AND '15-decem

SELECT EmployeeID, FirstName, LastName, HireDate, City
WHERE City IN ('Seattle', 'Tacoma', 'Redmond')

SELECT      EmployeeID, FirstName, LastName, HireDate, C
FROM        Employees
WHERE       HireDate NOT BETWEEN '1-june-2012' AND '15-
```

UNIVERSIDAD AUSTRAL

INGENIERÍA

DDL – Create Table

El Lenguaje SQL

Ejemplo: creación del siguiente esquema de BD.

- CLIENTES (DNI, NOMBRE, DIR)
- SUCURSALES (NSUC, CIUDAD)
- CUENTAS (COD, DNI, NSUCURS, SALDO)

```
CREATE TABLE CLIENTES (
  DNI VARCHAR(9) NOT NULL,
  NOMBRE VARCHAR(20),
  DIR  VARCHAR(30),
  PRIMARY KEY (DNI)
);
```

```
CREATE TABLE SUCURSALES (
  NSUC VARCHAR(4) NOT NULL,
  CIUDAD  VARCHAR(30),
  PRIMARY KEY (NSUC)
);
```

```
CREATE TABLE CUENTAS (
  COD VARCHAR(4) NOT NULL,
  DNI VARCHAR(9) NOT NULL,
  NSUCURS VARCHAR(4) NOT NULL,
  SALDO INT DEFAULT 0,
  PRIMARY KEY (COD, DNI, NSUCURS),
  FOREIGN KEY (DNI) REFERENCES CLIENTES (DNI),
  FOREIGN KEY (NSUCURS) REFERENCES SUCURSALES (NSUC) );
```

Las claves candidatas, que no deben alojar valores repetidos, se pueden indicar con la cláusula **UNIQUE**.

DML – Select

El Lenguaje SQL

```
SELECT A1, …, An
```

-Describe la salida deseada con:
- Nombres de columnas
- Expresiones aritméticas
- Literales
- Funciones escalares
- Funciones de columna

```
FROM T1, …, Tn
WHERE P
```

- Nombres de las tablas / vistas
- Condiciones de selección de filas

```
GROUP BY Ai1, …, Ain
HAVING Q
```

- Nombre de las columnas
- Condiciones de selección de grupo

```
ORDER BY Aj1, …, Ajn
```

- Nombres de columnas como criterio para ordenar

DML – Select (estructura básica)

El Lenguaje SQL

- Consta de tres cláusulas:
  - **`SELECT`**
    - La lista de los atributos que se incluirán en el resultado de una consulta.
  - **`FROM`**
    - Especifica las relaciones que se van a usar como origen en el proceso de la consulta.
  - **`WHERE`**
    - Especifica la condición de filtro sobre las tuplas en términos de los atributos de las relaciones de la cláusula **`FROM`**.

DML – Select (estructura básica)

El Lenguaje SQL

UNIVERSIDAD AUSTRAL
INGENIERÍA

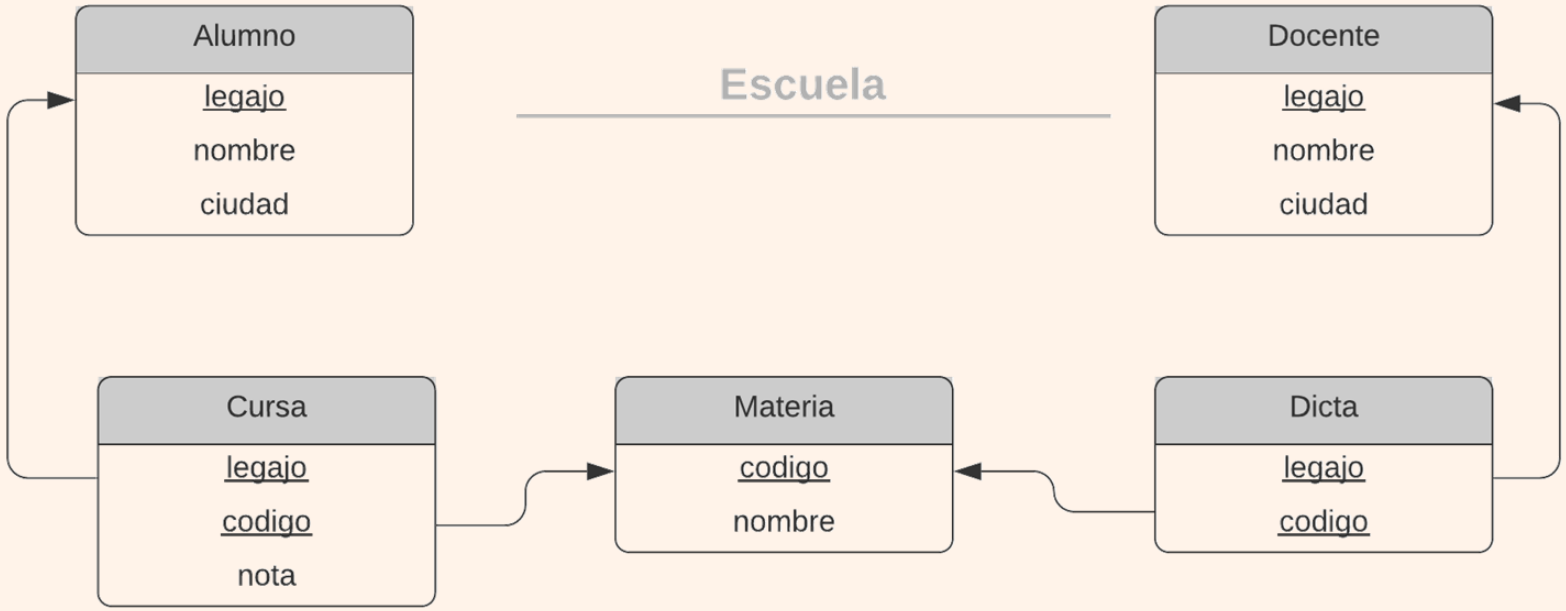
- Una consulta SQL tiene la forma:

```
SELECT A1, ..., An      /* Lista de atributos */
FROM R1, ..., Rm        /*Lista de relaciones*/
WHERE P;                /* Condición. Cláusula opcional */
```

  - Es posible que exista el mismo nombre de atributo en dos relaciones distintas.
  - Se añade "*NOMBRE_RELACION*." antes del nombre para desambiguar.

DML – Select

Ejemplos



Ver el listado de nombres de los alumnos

```
select nombre
from escuela.alumno
```

NOMBRE
Arturo Illia
Arturo Frondizi
Hipolito Yrigoyen
Juan Domingo Peron
Bartolome Mitre

DML – Select (estructura básica)

El Lenguaje SQL

- SQL permite duplicados en el resultado
- Para eliminar las tuplas repetidas se utiliza la cláusula **`DISTINCT`**.
- También es posible pedir explícitamente la inclusión de filas repetidas mediante el uso de la cláusula **`ALL`**.

Ver el listado de ciudades de nuestros alumnos

```
select distinct alumno.ciudad
from escuela.alumno
```

CIUDAD
Buenos Aires
Lobos
Paso de los Libres
Pergamino

DML - Select

Ejemplos

Ver el listado de todos los datos de los alumnos, ordenado por nombre

```
select *
from escuela.alumno
order by nombre
```

LEGAJO	NOMBRE	CIUDAD
2	Arturo Frondizi	Paso de los Libres
1	Arturo Illia	Pergamino
5	Bartolome Mitre	Buenos Aires
3	Hipolito Yrigoyen	Buenos Aires
4	Juan Domingo Peron	Lobos

DML - Select … Where …

Selección de filas

- La cláusula WHERE permite filtrar las filas de la relación resultante.
  - La condición de filtrado se especifica como un predicado.

- El predicado de la cláusula WHERE puede ser simple o complejo
  - Se utilizan los conectores lógicos **AND** (conjunción), **OR** (disyunción) y **NOT** (negación)

- Las expresiones pueden contener operadores de comparación estándar: =, <>, <, <=, >=, >.

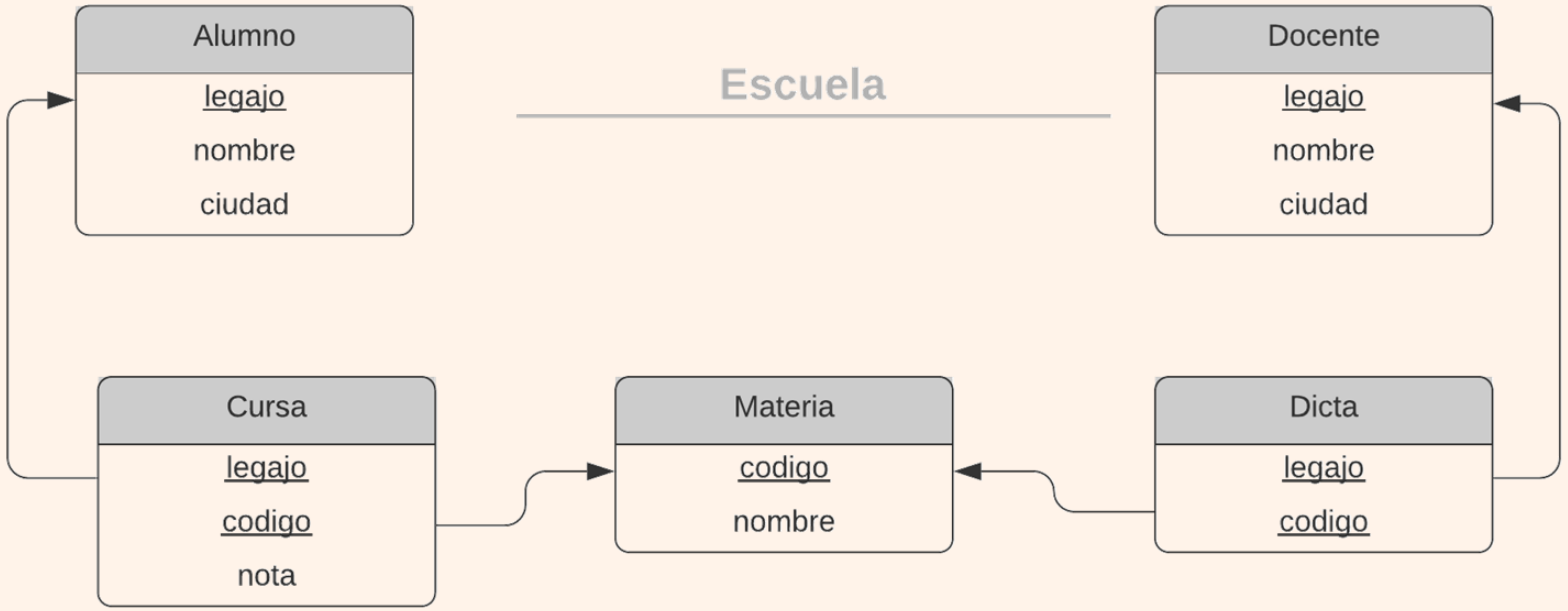
DML - Select … Where …

Selección de filas

- Las expresiones pueden contener predicados de comparación:
  - BETWEEN / NOT BETWEEN (Intervalos)
    - BETWEEN **Op1** AND **Op2**
  - IN / NOT IN (el valor se encuentra en una lista):
    - IN (**Val1**, **Val2**, …) → *Enumeración de datos*
    - IN (SELECT … FROM …) → *Subconsulta*

DML – Select … Where …

Ejemplos



Ver el listado de alumnos que son de Buenos Aires

```
select alumno.nombre
from escuela.alumno
where alumno.ciudad='Buenos Aires'
```

NOMBRE
Hipolito Yrigoyen
Bartolome Mitre

DML – Select… Where…

Selección de filas

- Otros predicados de comparación

  - LIKE / NOT LIKE (Comparación de cadenas de caracteres)
    - SQL distingue entre mayúsculas y minúsculas
    - Las cadenas de caracteres se incluyen entre comillas simples
    - SQL permite definir patrones a través de los siguientes caracteres:
      - '%' es equivale a "cualquier subcadena de caracteres"
      - '_' equivale a "cualquier carácter"

  - IS NULL / IS NOT NULL
  - ALL, SOME/ANY (subconsultas)
  - EXISTS (subconsultas)

DML – Select … Where … Like…

Ejemplos de búsqueda parcial

Ver el listado de alumnos que con nombre que empieza con A

```
select nombre
from escuela.alumno
where nombre like 'A%'
```

NOMBRE
Arturo Illia
Arturo Frondizi

Otras aplicaciones:

```
SELECT LASTNAME
FROM EMPLOYEE
WHERE LASTNAME LIKE '%SON';
```

THOMPSON
HENDERSON
ADAMSON
JEFFERSON
JOHNSON

```
SELECT LASTNAME
FROM EMPLOYEE
WHERE LASTNAME LIKE '%M%N%';
```

THOMPSON
ADAMSON
MARINO

DML – Select … Where …

Ejemplos

Ver el listado de alumnos que con nombre que NO empieza con A

```
select nombre
from escuela.alumno
where nombre not like 'A%'
```

NOMBRE
Hipolito Yrigoyen
Juan Domingo Peron
Bartolome Mitre

Ver cursadas con nota mayor a 8

```
select *
from escuela.cursa
where nota > 8
```

LEGAJO	CODIGO	NOTA
1	5	9
1	6	10
2	1	10
2	3	9
2	5	9
4	2	9
4	6	9
5	1	10
5	5	10

DML – Select… Where…

Selección de filas

- Otros predicados de comparación
  - IS NULL / IS NOT NULL
    - Campos vacíos (o no vacíos)
  - ALL, SOME/ANY (subconsulta)
    - Atributo *operador* ALL/ANY (Select…)
  - EXISTS / NOT EXISTS (subconsulta)
    - Atributo EXISTS/NOT EXISTS (Select…)
    - Predicado lógico (True/False)
    - Se puede incluir referencias a consulta superior (fuera de la subconsulta)

DML - Select

Selección de filas

- Funciones matemáticas (según DBMS)

Descripción	IBM DB2	SQL Server	Oracle	MySQL
Valor absoluto	ABSs	ABS	ABS	ABS
Menor entero >= valor	CEIL	CEILING	CEIL	CEILING
Mayor entero <= valor	FLOOR	FLOOR	FLOOR	FLOOR
Potencia	POWER	POWER	POWER	POWER
Redondeo a un número de cifras decimales	ROUND	ROUND	ROUND	ROUND
Módulo (resto de la division)	MOD.	%	MOD.	%

DML - Select

Selección de filas

- Ej: Necesito obtener el salario, la comisión y los ingresos totales de todos los empleados que tengan un salario menor de $100000, ordenado por número de empleado:

```
SELECT  EMPNO, SALARY, COMM, SALARY + COMM AS TOTAL
FROM EMPLOYEE
WHERE SALARY < 20000
ORDER BY EMPNO
```

EMPNO	SALARY	COMM	TOTAL
000210	18270.00	1462.00	19732.00
000250	19180.00	1534.00	20714.00
000260	17250.00	1380.00	18630.00
000290	15340.00	1227.00	16567.00
000300	17750.00	1420.00	19170.00
000310	15900.00	1272.00	17172.00
000320	19950.00	1596.00	21546.00

DML – Select… From … Join

Consulta de múltiples tablas

- Diferentes JOIN
  - **INNER JOIN**: Devuelve las filas donde hay coincidencia en cierto campo de ambas tablas
  - **OUTER JOIN**: Devuelve las filas donde hay coincidencia en al menos una de las tablas
    - **LEFT OUTER JOIN**: Incluye todas las filas de la tabla izquierda, aunque no haya coincidencia en la table derecha.
    - **RIGHT OUTER JOIN**: Incluye todas las filas de la tabla derecha, aunque no haya coincidencia en la table izquierda
  - **NATURAL JOIN**: Devuelve las filas donde hay coincidencia en el campo de igual nombre de ambas tablas

Sintaxis

Select …
From Tabla1 **inner/left/right join** Tabla2 **on** Tabla1.atributo1 = Tabla2.atributo2
Where …

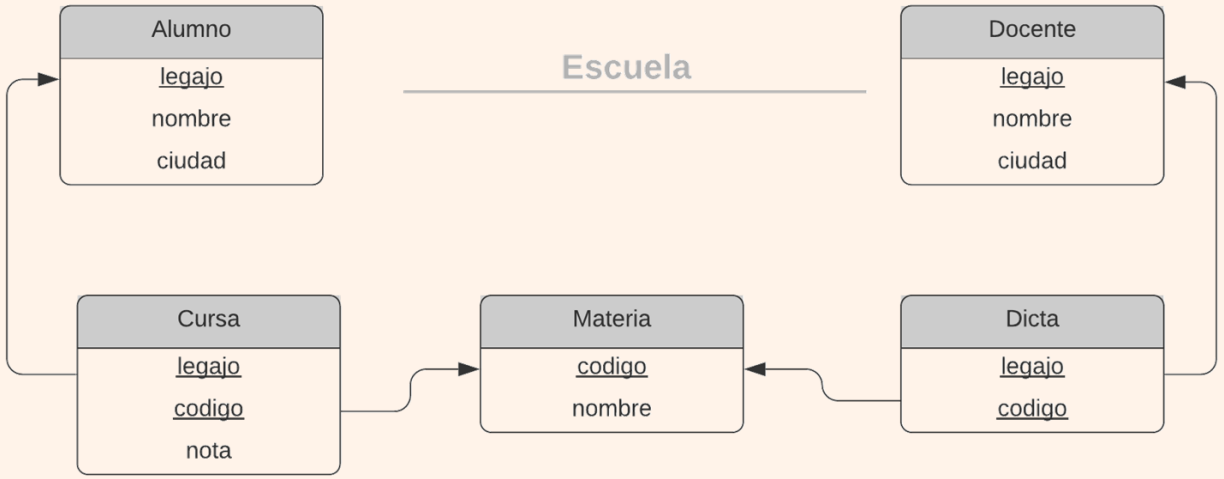
From Tabla1 **natural join** Tabla2

DML – Select… From … Join

Consulta de múltiples tablas

Ver las notas de Arturo Illia

```
Select C.nota
From escuela.alumno A inner join escuela.Cursa C on A.legajo = C.legajo
Where A.nombre = 'Arturo Illia'
```



NOTA
7
7
8
7
9
10

Alternativa:
(mayor costo computacional)

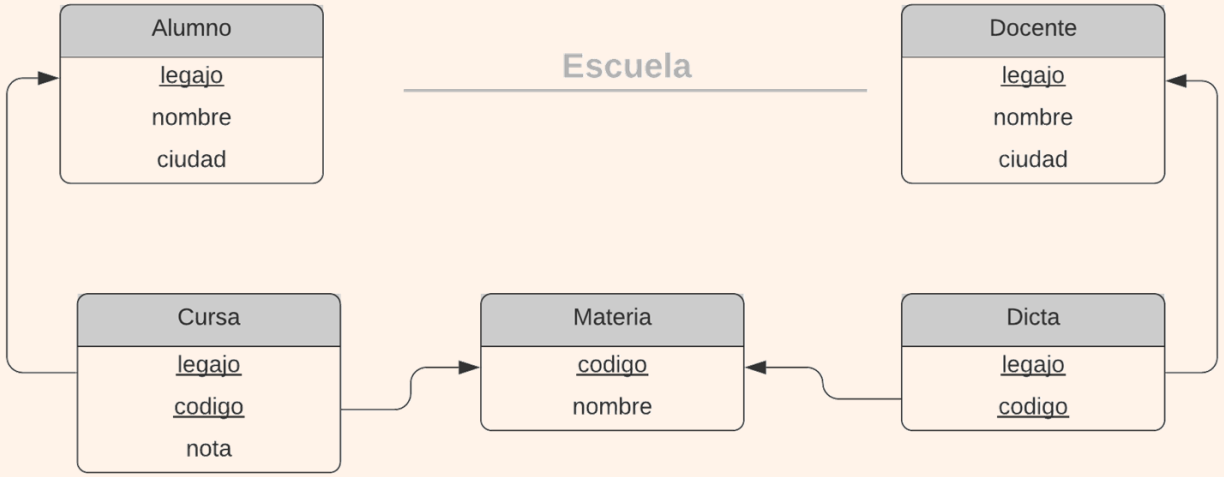
```
Select C.nota
From escuela.alumno A, escuela.Cursa C
Where  A.legajo = C.legajo AND A.nombre = 'Arturo Illia'
```

DML – Select… From … Join

Consulta de múltiples tablas

Listado de materias con los alumnos que las cursan, ordenado por materia:

```
Select M.nombre Materia, A.nombre Alumno
From escuela.alumno A inner join escuela.cursa C on
A.legajo = C.legajo inner join escuela.materia M on
C.codigo = M.codigo
order by M.nombre
```



MATERIA	ALUMNO
Arte	Arturo Illia
Arte	Arturo Frondizi
Arte	Bartolome Mitre
Educacion fisica	Arturo Illia
Educacion fisica	Juan Domingo Peron
Geografia	Arturo Illia
Geografia	Arturo Frondizi
Historia	Arturo Illia
Historia	Juan Domingo Peron
Literatura	Arturo Illia
Literatura	Juan Domingo Peron
Matematica	Arturo Illia
Matematica	Arturo Frondizi
Matematica	Bartolome Mitre

DML – Union

Funciones de conjunto

- Conecta 2 consultas completas
- Las columnas de cada Select deben tener tipos de datos compatibles
- **UNION** elimina duplicados
- Si se indica ORDER BY, debe ser la última cláusula de la sentencia

Ej:

```
Select alumno.nombre
From escuela.alumno
Where alumno.nombre like 'J%'
Union
Select docente.nombre
From escuela.docente
Where docente.nombre like 'J%'
```

NOMBRE
Jose de San Martin
Juan Domingo Peron
Juan Martin de Pueyrredon

DML – Select…

Funciones agregadas

- Las funciones de agregación son funciones que toman una colección (conjunto o multiconjunto) de valores de entrada y devuelve un solo valor.
- Las funciones de columna disponibles son: **AVG, MIN, MAX, SUM, COUNT.**
- Los datos de entrada para **SUM** y **AVG** deben ser una colección de números, pero el resto de operadores pueden ser datos de tipo no numérico.
- Por defecto las funciones se aplican a todas las tuplas resultantes de la consulta.
- Podemos agrupar las tuplas resultantes para poder aplicar las funciones de columna a grupos específicos utilizando la cláusula **GROUP BY**.
- En la cláusula **SELECT** de consultas que utilizan funciones de columna solamente pueden aparecer funciones de columna.
- En caso de utilizar **GROUP BY**, también pueden aparecer columnas utilizadas en la agrupación.
- Adicionalmente se pueden aplicar condiciones sobre los grupos utilizando la cláusula **HAVING**.

DML – Select…

Funciones agregadas

- Cálculo del total → **`SUM (expresión)`**
- Cálculo de la media → **`AVG (expresión)`**
- Obtener el valor mínimo → **`MIN (expresión)`**
- Obtener el valor máximo → **`MAX (expresión)`**
- Contar el número de filas que satisfacen la condición de búsqueda → **`COUNT(*)`**
  - Los valores NULL SI se cuentan.
- Contar el número de valores distintos en una columna → **`COUNT (DISTINCT nombre-columna)`**
  - Los valores NULL NO se cuenta.

DML – Select…

Funciones agregadas

- Listado de alumnos con su promedio

```
Select A.nombre, AVG(C.nota) Promedio
From escuela.alumno A inner join escuela.cursa C on A.legajo = C.legajo
Group by A.nombre
Order by A.nombre
```

NOMBRE	PROMEDIO
Arturo Frondizi	9
Arturo Illia	8
Bartolome Mitre	10
Juan Domingo Peron	8

- Listado de alumnos con promedio mayor o igual a 9

```
Select A.nombre, AVG(C.nota) Promedio
From escuela.alumno A inner join escuela.cursa C on A.legajo = C.legajo
Group by A.nombre
Having AVG(C.nota)>=9
Order by A.nombre
```

NOMBRE	PROMEDIO
Arturo Frondizi	9
Bartolome Mitre	10

DML – Select…

Funciones agregadas

- Listado de alumnos que sólo cursan dos materias

```
Select A.nombre, count(*)
From escuela.alumno A inner join escuela.cursa C on A.legajo = C.legajo
Group by A.nombre
Having count(*)=2
```

NOMBRE	COUNT(*)
Bartolome Mitre	2

- Listado de las materias con su nota mayor y menor

```
select m.codigo, m.nombre, min(nota) MENOR, max(nota) MAYOR
from escuela.cursa c join escuela.materia m on c.codigo=m.codigo
group by m.codigo, m.nombre
```

CODIGO	NOMBRE	MENOR	MAYOR
1	Matematica	7	10
2	Literatura	7	9
3	Geografia	8	9
4	Historia	7	7
5	Arte	9	10
6	Educacion fisica	9	10

DML – Select… Limit, Offset, Top

Filtro por número de filas

Se puede acotar el número de filas de la tabla resultante

- Limit: cantidad de filas
- Offset: a partir de qué fila

```
select *
from escuela.cursa
order by nota, código, legajo
limit 5 offset 7
```

→ `limit 7, 5`

- Top: primeras N filas

```
select * top 10
from escuela.cursa
order by nota, código, legajo
```

LEGAJO	CODIGO	NOTA
1	1	7
1	2	7
1	4	7
4	4	7
1	3	8
4	2	9
2	3	9
1	5	9
2	5	9
4	6	9
2	1	10
5	1	10
5	5	10
1	6	10

→

LEGAJO	CODIGO	NOTA
1	5	9
2	5	9
4	6	9
2	1	10
5	1	10

DML – Select… Case When

El "if" de SQL

Evalúa condiciones en el resultado y modifica la salida:

- CASE
  WHEN condición_1 THEN resultado_1
  WHEN condición_2 THEN resultado_2
  ...
  ELSE resultado_else
  END

Ej: Cálculo de impuesto a las ganancias, según monto del sueldo.

DML – Select… Case When

El "if" de SQL

UNIVERSIDAD AUSTRAL
INGENIERÍA

```
select alumno.nombre,
case when avg(cursa.nota)=10 then 'Brillante'
     when avg(cursa.nota)>8 then 'Buen alumno'
     when avg(cursa.nota)>6 then 'Normal'
     else 'Regular'
end Tipo
from escuela.cursa natural join escuela.alumno
group by alumno.nombre
```

NOMBRE	TIPO
Arturo Frondizi	Buen alumno
Arturo Illia	Normal
Bartolome Mitre	Brillante
Juan Domingo Peron	Normal

DML – Select… Exists

Exists en detalle

Permite especificar una subconsulta para probar la existencia de filas. Sintaxis:

SELECT A.n

FROM A

WHERE *(NOT)* EXISTS (subconsulta)

- Devuelve verdadero si la subconsulta contiene filas. De lo contrario, devuelve falso.
- Finaliza el procesamiento de la consulta ni bien encuentra una fila, por lo tanto se puede usar para mejorar el rendimiento de una consulta.
- Si la subconsulta regresa un valor NULL, el operador EXISTS sigue dando verdadero. El operador EXISTS es el único operador que permite utilizar tablas de la selección externa en la selección interna (subconsulta).
- Ej:

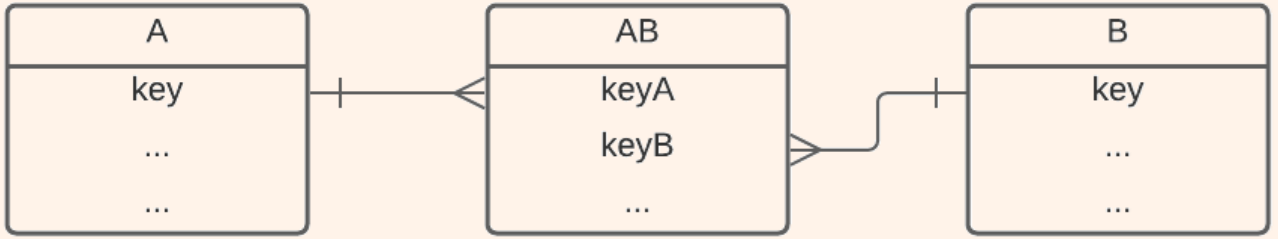
```
select alumno.nombre
from escuela.alumno
where exists
        (select 1
        from escuela.cursa
        where cursa.legajo=alumno.legajo)
```

DML – Select… Exists

Sentencia de "División"

Permite encontrar los registros de una tabla que están relacionados con "todos" los registros de otra tabla, en una relación N:M. Ej: Ver las tuplas de A que están relacionados con todas las tuplas de B.

```
SELECT A.n
FROM A
WHERE NOT EXISTS
        (SELECT 1
        FROM B
        WHERE NOT EXISTS
                (SELECT 1
                FROM AB
                WHERE A.key = AB.keyA
                and B.key = AB.keyB))
```



DML – Select… Exists

Sentencia de "División"

Ejercicio: Los alumnos que cursan todas las materias

```
SELECT a.nombre
FROM escuela.alumno a
WHERE NOT EXISTS
        (SELECT 1
        FROM escuela.materia m
        WHERE NOT EXISTS
                (SELECT 1
                FROM escuela.cursa c
                WHERE a.legajo = c.legajo
                AND m.codigo = c.codigo))
```

NOMBRE
Arturo Illia

SOMOSAUSTRAL

Muchas gracias.

www.austral.edu.ar

UNIVERSIDAD AUSTRAL