

5.1 Introducción

- **SQL** es un lenguaje de Consultas amigable para los usuarios de la Base de Datos.
 - Algebra Relacional (la base de SQL) está orientado a ingenieros especializados en Bases de Datos.
- SQL es el lenguaje comercial más popular y se ha convertido en el lenguaje estándar de las Bases de Datos Relacionales.
- Significa **S**tructured **Q**uery **L**anguage, aunque el nombre original era **SEQUEL*** desarrollado como parte del proyecto del sistema **R** de **IBM**.

* **S**equential **E**nglish **Q**Uery **L**anguage.

5.1 Introducción

- SQL es parte del Lenguaje de Manipulación de Datos (**DML**) y no a la inversa como a veces se dice.
 - **DML** consta de instrucciones para añadir, modificar, eliminar, datos a la base de datos, además de las instrucciones de consulta correspondientes a **SQL**.
- Otro lenguaje importante de los DBMS es **DDL** (Lenguaje de Definición de Datos) que es el lenguaje que permite crear o modificar el esquema de una BD.

5.1 Introducción

Una consulta en SQL
consta de tres cláusulas:
select, from y where

Π ... **Select**
 $\bowtie$... **From**
 σ ... **Where**

No debe confundir **select** con la operación seleccionar del álgebra relacional, de hecho la discrepancia es un error desde la creación de SQL.

Select $A_1, A_2, \dots, A_n$ from $r_1, r_2, \dots, r_m$ where P

Equivale a:

$$\Pi_{A_1, A_2, \dots, A_n} (\sigma_P (r_1 \bowtie r_2 \bowtie \dots \bowtie r_m))$$

Ejecute el código DDL de esta diapositiva y las siguientes para crear una BD llamada ***SQLejemplos***, usando **SQL Server**. Como se puede ver, hay que crear las tablas ***Alumnos*** e ***InasistAlum*** usando SQL Server e insertar las tuplas con el código que se indica para efectuar las primeras consultas de ejemplo.

```
CREATE DATABASE SQLejemplos
USE SQLejemplos
```

```
CREATE TABLE Alumnos(
    idAlumno INT PRIMARY KEY IDENTITY NOT NULL,
    NumControl VARCHAR(10) UNIQUE,
    Nombre VARCHAR(30),
    Domicilio VARCHAR(30),
    FechaNacimiento DATE NOT NULL,
    Curp VARCHAR(18) NOT NULL)
```

```
INSERT INTO Alumnos VALUES
('98042151', 'Pato Lucas', 'Zarco 123', '1981-02-07', 'LP11')
INSERT INTO Alumnos VALUES
('97041587', 'Atomic Ant', 'Negrete 1002', '1982-03-24', 'AA22')
INSERT INTO Alumnos VALUES
('97043014', 'Mickey Mouse', 'Fresno 1410', '1981-06-23', 'MM33')
INSERT INTO Alumnos VALUES
('96042121', 'Johnny Bravo', 'Patoni 100', '1982-03-31', 'BJ44')
INSERT INTO Alumnos VALUES
('96044350', 'Robin Hood', 'Aquiles Serdán 1102', '1979-03-20', 'RH99')
INSERT INTO Alumnos VALUES
('97111432', 'Oso Yogui', 'Victoria 345', '1980-04-11', 'OY88')
INSERT INTO Alumnos VALUES
('98042012', 'Hércules', 'Juárez 543', '1981-09-24', 'HH66')
```

CREATE TABLE

```
InasistAlum(  
  idInasistAlum INT PRIMARY KEY IDENTITY NOT NULL,  
  idAlumno INT REFERENCES Alumnos NOT NULL,  
  Fecha DATE NOT NULL,  
  Hora TIME(5) NOT NULL,  
  Motivo VARCHAR(20) NOT NULL,  
  UNIQUE (idAlumno, Fecha, Hora)  
)
```

```
INSERT INTO InasistAlum VALUES (1, '2001-03-05', '16:00', 'Deportes')  
INSERT INTO InasistAlum VALUES (1, '2001-03-06', '16:00', 'Enfermedad')  
INSERT INTO InasistAlum VALUES (1, '2001-03-07', '16:00', 'Injustificada')  
INSERT INTO InasistAlum VALUES (1, '2001-03-08', '16:00', 'Injustificada')  
INSERT INTO InasistAlum VALUES (3, '2001-03-01', '09:00', 'Enfermedad')  
INSERT INTO InasistAlum VALUES (3, '2001-03-01', '10:00', 'Injustificada')  
INSERT INTO InasistAlum VALUES (3, '2001-03-01', '11:00', 'Injustificada')  
INSERT INTO InasistAlum VALUES (4, '2001-03-09', '16:00', 'Enfermedad')
```

5.1 Estructura Básica

relación **InasistAlum**

IdAlumno	Fecha	Hora	Motivo
1	05/03/2001	16:00	Deportes
1	06/03/2001	16:00	Enfermedad
1	07/03/2001	16:00	Injustificada
1	08/03/2001	16:00	Injustificada
3	01/03/2001	09:00	Enfermedad
3	01/03/2001	10:00	Injustificada
3	01/03/2001	11:00	Injustificada
4	09/03/2001	16:00	Enfermedad

$\sigma_{\text{hora}='16:00'}(\text{InasistAlum})$

SELECT

*

FROM

InasistAlum

WHERE

hora= '16:00'

En la expresión de este ejemplo en Álgebra Relacional, no hay *Proyectar*, sin embargo en SQL es obligatoria la palabra SELECT, por eso se indica con un asterisco que se van a mostrar todas las columnas de la tabla.

5.1 Estructura Básica

Alumnos

IdAlumno	Control	Nombre	Domicilio	FechaNac	CURP
1	98042151	Pato Lucas	Zarco 123	07/02/1981	LP11
2	97041587	Atomic Ant	Negrete 1002	24/03/1982	AA22
3	97043014	MickeyMouse	Fresno 1410	23/06/1981	MM33
4	96042121	Johnny Bravo	Patoni 100	31/03/1982	JB44
5	96044350	Robin Hood	Aquiles Serdán 1102	20/03/1979	RH99
6	97111432	Oso Yogui	Victoria 345	11/04/1980	OY88
7	98042012	Hércules	Juárez 543	24/09/1981	HH66

$\Pi_{\text{nombre, curp}}(\text{Alumnos})$

SELECT

Nombre, Curp

FROM

Alumnos

La cláusula WHERE se puede omitir cuando la expresión equivalente en Álgebra Relacional no contiene

σ

5.1 Estructura Básica

relación **InasistAlum**

IdAlumno	Fecha	Hora	Motivo
1	05/03/2001	16:00	Deportes
1	06/03/2001	16:00	Enfermedad
1	07/03/2001	16:00	Injustificada
1	08/03/2001	16:00	Injustificada
3	01/03/2001	09:00	Enfermedad
3	01/03/2001	10:00	Injustificada
3	01/03/2001	11:00	Injustificada
4	09/03/2001	16:00	Enfermedad

$\Pi_{\text{fecha, motivo, IdAlumno}}(\sigma_{\text{hora}='16:00'}(\text{InasistAlum}))$

SELECT

Fecha, Motivo, idAlumno

FROM

InasistAlum

WHERE

Hora= '16:00'

Fecha	Motivo	IdAlumno
05/03/2001	Deportes	1
06/03/2001	Enfermedad	1
07/03/2001	Injustificada	1
08/03/2001	Injustificada	1
09/03/2001	Enfermedad	4

5.1 Estructura Básica

relación **InasistAlum**

IdAlumno	Fecha	Hora	Motivo
1	05/03/2001	16:00	Deportes
1	06/03/2001	16:00	Enfermedad
1	07/03/2001	16:00	Injustificada
1	08/03/2001	16:00	Injustificada
3	01/03/2001	09:00	Enfermedad
3	01/03/2001	10:00	Injustificada
3	01/03/2001	11:00	Injustificada
4	09/03/2001	16:00	Enfermedad

$\sigma_{\text{hora} < '12:00'} (\text{InasistAlum})$

SELECT

*

FROM

InasistAlum

WHERE

hora < '12:00'

5.1 Estructura Básica

relación **InasistAlum**

IdAlumno	Fecha	Hora	Motivo
1	05/03/2001	16:00	Deportes
1	06/03/2001	16:00	Enfermedad
1	07/03/2001	16:00	Injustificada
1	08/03/2001	16:00	Injustificada
3	01/03/2001	09:00	Enfermedad
3	01/03/2001	10:00	Injustificada
3	01/03/2001	11:00	Injustificada
4	09/03/2001	16:00	Enfermedad

$\Pi_{\text{IdAlumno}}(\sigma_{\text{hora} < '12:00'}(\text{InasistAlum}))$

```
SELECT
    idAlumno
FROM
    InasistAlum
WHERE
    hora < '12:00'
```

```
SELECT
    DISTINCT idAlumno
FROM
    InasistAlum
WHERE
    hora < '12:00'
```

5.1 Estructura Básica

Alumnos

IdAlumno	Control	Nombre	Domicilio	FechaNac	CURP
1	98042151	Pato Lucas	Zarco 123	07/02/1981	LP11
2	97041587	Atomic Ant	Negrete 1002	24/03/1982	AA22
3	97043014	MickeyMouse	Fresno 1410	23/06/1981	MM33
4	96042121	Johnny Bravo	Patoni 100	31/03/1982	JB44
5	96044350	Robin Hood	Aquiles Serdán 1102	20/03/1979	RH99
6	97111432	Oso Yogui	Victoria 345	11/04/1980	OY88
7	98042012	Hércules	Juárez 543	24/09/1981	HH66

relación InasistAlum

IdAlumno	Fecha	Hora	Motivo
1	05/03/2001	16:00	Deportes
1	06/03/2001	16:00	Enfermedad
1	07/03/2001	16:00	Injustificada
1	08/03/2001	16:00	Injustificada
3	01/03/2001	09:00	Enfermedad
3	01/03/2001	10:00	Injustificada
3	01/03/2001	11:00	Injustificada
4	09/03/2001	16:00	Enfermedad

$\Pi_{\text{nombre, fecha}}(\sigma_{\text{hora}='16:00' \wedge \text{Alumnos.IdAlumno}=\text{InasistAlum.IdAlumno}}(\text{Alumnos} \times \text{InasistAlum}))$

SELECT

Nombre, Fecha

FROM

Alumnos, InasistAlum

WHERE

Alumnos.idAlumno=InasistAlum.idAlumno
and Hora='16:00'

Esta es la expresión más simple para resolver este problema, ya que solo se emplea un operador de cada tipo.

5.1 Estructura Básica

Alumnos

IdAlumno	Control	Nombre	Domicilio	FechaNac	CURP
1	98042151	Pato Lucas	Zarco 123	07/02/1981	LP11
2	97041587	Atomic Ant	Negrete 1002	24/03/1982	AA22
3	97043014	MickeyMouse	Fresno 1410	23/06/1981	MM33
4	96042121	Johnny Bravo	Patoni 100	31/03/1982	JB44
5	96044350	Robin Hood	Aquiles Serdán 1102	20/03/1979	RH99
6	97111432	Oso Yogui	Victoria 345	11/04/1980	OY88
7	98042012	Hércules	Juárez 543	24/09/1981	HH66

relación **InasistAlum**

IdAlumno	Fecha	Hora	Motivo
1	05/03/2001	16:00	Deportes
1	06/03/2001	16:00	Enfermedad
1	07/03/2001	16:00	Injustificada
1	08/03/2001	16:00	Injustificada
3	01/03/2001	09:00	Enfermedad
3	01/03/2001	10:00	Injustificada
3	01/03/2001	11:00	Injustificada
4	09/03/2001	16:00	Enfermedad

Ejercite con las siguientes consultas:

- ▶ Obtener una tabla que contenga (únicamente) las fechas en que hubo inasistencias Injustificadas.
- ▶ Obtener una tabla que contenga el nombre y domicilio, así como la fecha y hora de inasistencia de aquellos quienes tuvieron faltas Injustificadas.
- ▶ Obtener una tabla que contenga el número de control y nombre de los alumnos que han tenido al menos una inasistencia injustificada.

5.1 Estructura Básica

Relación **Materias**

IdMateria	Clave	Nombre
1	8807	Estructuras de Datos I
2	8821	Bases de Datos I
3	8011	Matemáticas IV
4	8927	Bases de Datos I
5	8806	Programación II

Escriba los comandos
CREATE e INSERT para
generar esta tabla y pueda
probar la consulta.

$\Pi_{\text{Materias.nombre}} (\sigma_{\text{Materias.nombre}=\text{Mat2.nombre} \wedge \text{Materias.IdMateria} \neq \text{Mat2.IdMateria}} ($
Materias X $\rho_{\text{Mat2}}(\text{Materias})))$

SELECT DISTINCT

Materias.Nombre

FROM

Materias, Materias Mat2

WHERE

Materias.Nombre=Mat2.Nombre AND

Materias.idMateria!=Mat2.idMateria

RENOMBRAR

5.1 Estructura Básica

Subconsultas

Las expresiones de esta página y la siguiente, equivalen a las expresiones en AR que se usaron para ilustrar la optimización de consultas por parte de los DBMSs. Se dejan aquí como ejemplo de subconsultas.

Alumnos

IdAlumno	Control	Nombre	Domicilio	FechaNac	CURP
1	98042151	Pato Lucas	Zarco 123	07/02/1981	LP11
2	97041587	Atomic Ant	Negrete 1002	24/03/1982	AA22
3	97043014	MickeyMouse	Fresno 1410	23/06/1981	MM33
4	96042121	Johnny Bravo	Patoni 100	31/03/1982	JB44
5	96044350	Robin Hood	Aquiles Serdán 1102	20/03/1979	RH99
6	97111432	Oso Yogui	Victoria 345	11/04/1980	OY88
7	98042012	Hércules	Juárez 543	24/09/1981	HH66

relación InasistAlum

IdAlumno	Fecha	Hora	Motivo
1	05/03/2001	16:00	Deportes
1	06/03/2001	16:00	Enfermedad
1	07/03/2001	16:00	Injustificada
1	08/03/2001	16:00	Injustificada
3	01/03/2001	09:00	Enfermedad
3	01/03/2001	10:00	Injustificada
3	01/03/2001	11:00	Injustificada
4	09/03/2001	16:00	Enfermedad

$$\Pi_{\text{nombre, fecha}}(\sigma_{\text{hora}='16:00' \wedge \text{Alumnos.IdAlumno}=\text{InasistAlum.IdAlumno}}(\Pi_{\text{IdAlumno, nombre}}(\text{Alumnos}) \times \text{InasistAlum}))$$

Obligatorio RENOMBRAR
las subconsultas

SELECT Nombre, Fecha
FROM

(SELECT idAlumno, Nombre FROM Alumnos) Alum2, InasistAlum
WHERE

Hora= '16:00' and Alum2.IdAlumno=InasistAlum.IdAlumno

5.1 Estructura Básica

Subconsultas

relación **InasistAlum**

Alumnos

IdAlumno	Control	Nombre	Domicilio	FechaNac	CURP
1	98042151	Pato Lucas	Zarco 123	07/02/1981	LP11
2	97041587	Atomic Ant	Negrete 1002	24/03/1982	AA22
3	97043014	MickeyMouse	Fresno 1410	23/06/1981	MM33
4	96042121	Johnny Bravo	Patoni 100	31/03/1982	JB44
5	96044350	Robin Hood	Aquiles Serdán 1102	20/03/1979	RH99
6	97111432	Oso Yogui	Victoria 345	11/04/1980	OY88
7	98042012	Hércules	Juárez 543	24/09/1981	HH66

IdAlumno	Fecha	Hora	Motivo
1	05/03/2001	16:00	Deportes
1	06/03/2001	16:00	Enfermedad
1	07/03/2001	16:00	Injustificada
1	08/03/2001	16:00	Injustificada
3	01/03/2001	09:00	Enfermedad
3	01/03/2001	10:00	Injustificada
3	01/03/2001	11:00	Injustificada
4	09/03/2001	16:00	Enfermedad

$\Pi_{\text{nombre, fecha}}(\sigma_{\text{Alumnos.IdAlumno}=\text{InasistAlum.IdAlumno}}$

$(\Pi_{\text{IdAlumno, nombre}}(\text{Alumnos}) \times \sigma_{\text{hora}='16:00'}(\text{InasistAlum}))$

SELECT

Nombre, Fecha

FROM

(SELECT idAlumno, Nombre FROM Alumnos) Alum2,

(SELECT * FROM InasistAlum WHERE Hora='16:00') Inasist2

WHERE

Alum2.idAlumno=Inasist2.idAlumno

5.1 Estructura Básica

USE SQLejemplos

```
CREATE TABLE Maestros(  
    idMaestro INT PRIMARY KEY IDENTITY NOT NULL,  
    Curp VARCHAR(18) UNIQUE NOT NULL,  
    Nombre VARCHAR(30),  
    Domicilio VARCHAR(30),  
    FechaNacimiento DATE NOT NULL  
)
```

INSERT INTO
 Maestros

VALUES

```
( 'X11', 'Xavier López Chabelo', 'Templo Atenea 111', '1961-12-17' ),  
( 'A22', 'Aristóteles', 'Coliseo 122', '1965-11-20' ),  
( 'T44', 'Tchaikovski', 'Insurgentes 100', '1964-05-03' ),  
( 'RH99', 'Robin Houd', 'Aquiles Serdán 1102', '1979-03-20' ),  
( 'B33', 'Boticelli Sandro', 'Cataluña 343', '1963-03-29' ),  
( 'F77', 'Fernández José Ramón', 'Coapa 3221', '1953-07-31' )
```

5.1 Estructura Básica

Alumnos

IdAlumno	Control	Nombre	Domicilio	FechaNac	CURP
1	98042151	Pato Lucas	Zarco 123	07/02/1981	LP11
2	97041587	Atomic Ant	Negrete 1002	24/03/1982	AA22
3	97043014	MickeyMouse	Fresno 1410	23/06/1981	MM33
4	96042121	Johnny Bravo	Patoni 100	31/03/1982	JB44
5	96044350	Robin Hood	Aquiles Serdán 1102	20/03/1979	RH99
6	97111432	Oso Yogui	Victoria 345	11/04/1980	OY88
7	98042012	Hércules	Juárez 543	24/09/1981	HH66

Maestros

IdMaestro	CURP	Nombre	Domicilio	FechaNac
1	X11	Xavier Lopez Chabelo	Templo Atenea 111	17/12/1961
2	A22	Aristóteles	Coliseo 122	20/11/1965
3	T44	Tchaikovski	Insurgentes 1000	03/05/1964
4	RH99	Robin Houd	Aquiles Serdán 1102	20/03/1979
5	B33	Botticelli Sandro	Cataluña 343	29/03/1963
6	F77	Fernández José Ramón	Coapa 3221	31/07/1953

$\Pi_{\text{Nombre, Day(FechaNac)}} (\sigma_{\text{Month (FechaNac)=3}}(\text{Alumnos})) \cup \Pi_{\text{Nombre, Day(FechaNac)}} (\sigma_{\text{Month (FechaNac)=3}}(\text{Maestros}))$

Los DBMS's cuentan con atributos tipo *fecha*, por lo que hay que usar una función para obtener por separado cada una de las partes de la fecha:

```
SELECT Nombre, DAY(FechaNac) FROM Alumnos WHERE MONTH(FechaNac)=3
      UNION
```

```
SELECT nombre, DAY(FechaNac) FROM Maestros WHERE MONTH(FechaNac)=3
```

5.1 Estructura Básica

Misma consulta de la página anterior añadiendo el renombrado de columnas (color rojo):

```
SELECT Nombre, DAY(FechaNac) AS Dia
      FROM Alumnos
      WHERE MONTH(FechaNac)=3
      union
SELECT nombre, DAY(FechaNac) as Dia
      FROM Maestros
      WHERE MONTH(FechaNac)=3
```

(observe la diferencia con respecto al resultado de la consulta similar de la diapositiva anterior)

5.1 Estructura Básica

Para probar la siguiente consulta, añade el atributo ***Creditos*** a la tabla ***Materias***

Relación **Materias**

IdMateria	Clave	Nombre	Creditos
1	8807	Estructuras de Datos I	8
2	8821	Bases de Datos I	6
3	8011	Matemáticas IV	4
4	8927	Bases de Datos I	8
5	8806	Programación II	10
6	8126	Bioingeniería	10

El Operador **BETWEEN** permite simplificar consultas que contienen una cláusula **where** que especifica que cierto atributo se encuentre dentro de un rango de valores.

En vez de:

```
SELECT * FROM Materias WHERE Creditos >= 6 AND Creditos <= 8
```

Puede escribirse:

```
SELECT * FROM Materias WHERE Creditos BETWEEN 6 AND 8
```

Se busca que **SQL** se parezca más al lenguaje natural para facilidad del usuario

5.1 Estructura Básica

relación **InasistAlum**

IdAlumno	Fecha	Hora	Motivo
1	05/03/2001	16:00	Deportes
1	06/03/2001	16:00	Enfermedad
1	07/03/2001	16:00	Injustificada
1	08/03/2001	16:00	Injustificada
3	01/03/2001	09:00	Enfermedad
3	01/03/2001	10:00	Injustificada
3	01/03/2001	11:00	Injustificada
4	09/03/2001	16:00	Enfermedad

SELECT DISTINCT

IdAlumno

FROM

InasistAlum

WHERE

Fecha BETWEEN '2001/03/06' AND '2001/03/08'

5.1 Estructura Básica

Operador LIKE

Es útil cuando no es posible resolver consultas aunque los atributos tengan dominio atómico.

Se aplica a atributos char empleando los símbolos % y _ para representar secciones del valor del atributo.

Caracter % (representa una subcadena de cualquier tamaño dentro del valor del atributo)

Caracter _ (representa a cualquier carácter dentro del valor del atributo)

5.1 Estructura Básica

Alumnos

IdAlumno	Control	Nombre	Calle	Colonia	FechaNac	CURP
1	98042151	Pato Lucas	Zarco 123	Centro	07/02/1981	LP11
2	97041587	Atomic Ant	Negrete 1002	Del Lago	24/03/1982	AA22
3	97043014	MickeyMouse	Fresno 1410	Real del Prado	23/06/1981	MM33
4	96042121	Johnny Bravo	Patoni 100	Centro	31/03/1982	JB44
5	96044350	Robin Hood	Aquiles Serdán 1102	Nueva Vizcaya	20/03/1979	RH99
6	97111432	Oso Yogui	Victoria 345	Centro	11/04/1980	OY88
7	98042012	Hércules	Juárez 543	Centro	24/09/1981	HH66

Encontrar a los alumnos que tienen un número de control que no corresponde al ITD (valor distinto de '04' en las posiciones 3 y 4).

SELECT

*

FROM

Alumnos

WHERE

NumControl not like '__04%'

5.2 Estructura Básica

Maestros

IdMaestro	CURP	Nombre	Calle	FechaNac
1	X11	Xavier Lopez Chabelo	Templo Atenea 111	17/12/1961
2	A22	Aristóteles	Coliseo 122	20/11/1965
3	T44	Tchaikovski	Insurgentes 1000	03/05/1964
4	RH99	Robin Hood	Aquiles Serdán 1102	20/03/1979
5	B33	Botticelli Sandro	Cataluña 343	29/03/1963
6	F77	Fernández José Ramón	Coapa 3221	31/07/1953

```
SELECT * FROM Maestros  
WHERE Calle LIKE 'C%'
```

```
SELECT Nombre from Maestros  
WHERE Nombre LIKE '_o%'
```

5.2 Estructura Básica

Relación **Materias**

IdMateria	Clave	Nombre	Creditos
1	8807	Estructuras de Datos I	8
2	8821	Bases de Datos I	6
3	8011	Matemáticas IV	4
4	8927	Bases de Datos I	8
5	8806	Programación II	10
6	8126	Bioingeniería	10

```
SELECT * FROM Materias  
WHERE Nombre LIKE '%datos%'
```

```
SELECT * FROM Materias  
WHERE Nombre LIKE '%DATOS%'
```

5.2 Estructura Básica

Alumnos

IdAlumno	Control	Nombre	Domicilio	FechaNac	CURP
1	98042151	Pato Lucas	Zarco 123, Centro	07/02/1981	LP11
2	97041587	Atomic Ant	Negrete 1002, Fracc. Del Lago	24/03/1982	AA22
3	97043014	MickeyMouse	Fresno 1410, Col. Real del Prado	23/06/1981	MM33
4	96042121	Johnny Bravo	Patoni 100, Centro	31/03/1982	JB44
5	96044350	Robin Hood	Aquiles Serdán 1102, Col. Nueva Vizcaya	20/03/1979	RH99
6	97111432	Oso Yogui	Victoria 345, Centro	11/04/1980	OY88
7	98042012	Hércules	Calle Real # 300 Int. 1 Col. Loma Dorada	24/09/1981	HH66

Encontrar los nombres de los alumnos que viven en la calle "Real".

```
SELECT Nombre FROM Alumnos
WHERE Domicilio LIKE '%Real%'
```

El resultado de esta consulta no es satisfactorio porque hay calles y colonias que contienen esa palabra.

5.2 Estructura Básica

Alumnos

IdAlumno	Control	Nombre	Calle	NumExt	NumInt	Colonia	FechaNac	CURP
1	98042151	Pato Lucas	Zarco	123		Centro	07/02/1981	LP11
2	97041587	Atomic Ant	Negrete	1002		Fracc. del Lago	24/03/1982	AA22
3	97043014	MickeyMouse	Fresno	1410	B	Real del Prado	23/06/1981	MM33
4	96042121	Johnny Bravo	Patoni	100		Centro	31/03/1982	JB44
5	96044350	Robin Hood	Aquiles Serdán	1102		Nueva Vizcaya	20/03/1979	RH99
6	97111432	Oso Yogui	Victoria	345		Centro	11/04/1980	OY88
7	98042012	Hércules	Real	300	1	Loma Dorada	24/09/1981	HH66

```
SELECT Nombre FROM Alumnos  
WHERE Calle='Real'
```

Los esquemas que cuentan con todos los atributos con dominio atómico, facilitan la obtención de consultas precisas. **Observe que no se requiere usar LIKE.**

5.2 Estructura Básica

Cláusula Order by

Alumnos

IdAlumno	Control	Nombre	Calle	NumExt	NumInt	Colonia	FechaNac	CURP
1	98042151	Pato Lucas	Zarco	123		Centro	07/02/1981	LP11
2	97041587	Atomic Ant	Negrete	1002		Fracc. del Lago	24/03/1982	AA22
3	97043014	MickeyMouse	Fresno	1410	B	Real del Prado	23/06/1981	MM33
4	96042121	Johnny Bravo	Patoni	100		Centro	31/03/1982	JB44
5	96044350	Robin Hood	Aquiles Serdán	1102		Nueva Vizcaya	20/03/1979	RH99
6	97111432	Oso Yogui	Victoria	345		Centro	11/04/1980	OY88
7	98042012	Hércules	Real	300	1	Loma Dorada	24/09/1981	HH66

Obtener una relación de los alumnos ordenada en base a la colonia.

```
SELECT * FROM Alumnos ORDER BY Colonia
```

```
SELECT * FROM Alumnos ORDER BY 7
```

5.2 Estructura Básica

Cláusula Order by

Alumnos

IdAlumno	Control	Nombre	Calle	NumExt	NumInt	Colonia	FechaNac	CURP
1	98042151	Pato Lucas	Zarco	123		Centro	07/02/1981	LP11
2	97041587	Atomic Ant	Negrete	1002		Fracc. del Lago	24/03/1982	AA22
3	97043014	MickeyMouse	Fresno	1410	B	Real del Prado	23/06/1981	MM33
4	96042121	Johnny Bravo	Patoni	100		Centro	31/03/1982	JB44
5	96044350	Robin Hood	Aquiles Serdán	1102		Nueva Vizcaya	20/03/1979	RH99
6	97111432	Oso Yogui	Victoria	345		Centro	11/04/1980	OY88
7	98042012	Hércules	Real	300	1	Loma Dorada	24/09/1981	HH66

Obtener una relación con el domicilio y nombre de los alumnos ordenada en primer lugar por la colonia, en 2º lugar por la calle y por último números exterior e interior.

```
SELECT Colonia,Calle,NumeroExt,NumeroInt,Nombre
FROM Alumnos
ORDER BY Colonia,Calle,NumeroExt,NumeroInt
```

Añada tuplas con colonia y calle iguales para observar el efecto de la expresión.

5.3 Funciones de Agrupación

Para los ejemplos de las diapositivas siguientes, añade más tuplas a la tabla **InasistAlum**

```
INSERT INTO InasistAlum VALUES
(5, '2002-10-25', '10:00', 'Deportes'),
(5, '2003-04-18', '12:00', 'Enfermedad'),
(6, '2002-05-10', '15:00', 'Injustificada'),
(6, '2002-11-20', '15:00', 'Injustificada'),
(7, '2001-09-15', '20:00', 'Enfermedad'),
(7, '2002-08-31', '20:00', 'Injustificada'),
(5, '2003-04-11', '12:00', 'Injustificada'),
(5, '2002-04-12', '13:00', 'Enfermedad')
```

idInasistAlum	idAlumno	Fecha	Hora	Motivo
1	1	05/03/2001	16:00	Deportes
2	1	06/03/2001	16:00	Enfermedad
3	1	07/03/2001	16:00	Injustificada
4	1	08/03/2001	16:00	Injustificada
5	3	01/03/2001	09:00	Enfermedad
6	3	01/03/2001	10:00	Injustificada
7	3	01/03/2001	11:00	Injustificada
8	4	09/03/2001	16:00	Enfermedad
9	5	25/10/2002	10:00	Deportes
10	5	18/04/2003	12:00	Enfermedad
11	6	10/05/2002	15:00	Injustificada
12	6	20/11/2002	15:00	Injustificada
13	7	15/09/2001	20:00	Enfermedad
14	7	31/08/2002	20:00	Injustificada
15	5	11/04/2003	12:00	Injustificada
16	5	12/04/2002	13:00	Enfermedad

5.3 Funciones de Agrupación

- Operan sobre **grupos de tuplas**.
- Al usar una de las funciones siguientes, se aplica a cada grupo por separado y se obtendrá el resultado indicado:
 - **MIN(atributo)**.
Determina el valor menor de un atributo de un **grupo de tuplas**.
 - **MAX(atributo)**.
Determina el valor mayor de un atributo de un grupo de tuplas.
 - **SUM(atributo)**.
Calcula la suma de los valores de un atributo de un grupo de tuplas.

5.3 Funciones de Agrupación

- Funciones:
 - **AVG(atributo)**.
Calcula el promedio de los valores de un atributo de un grupo de tuplas.
 - **COUNT(atributo)** o **Count(*)**.
Cuenta el número de tuplas que hay en un grupo.
- Antes del **atributo** puede ir la cláusula **DISTINCT**.
- Se debe indicar el criterio de agrupación usando la cláusula **GROUP BY**.
- En la tabla resultante aparecerá **una tupla por cada grupo formado**.

5.3 Funciones de Agrupación

Ejemplo

Obtener una relación de los motivos y su total de inasistencias correspondiente, ordenada en forma descendente por el total.

idInasistAlum	idAlumno	Fecha	Hora	Motivo
1	1	05/03/2001	16:00	Deportes
2	1	06/03/2001	16:00	Enfermedad
3	1	07/03/2001	16:00	Injustificada
4	1	08/03/2001	16:00	Injustificada
5	3	01/03/2001	09:00	Enfermedad
6	3	01/03/2001	10:00	Injustificada
7	3	01/03/2001	11:00	Injustificada
8	4	09/03/2001	16:00	Enfermedad
9	5	25/10/2002	10:00	Deportes
10	5	18/04/2003	12:00	Enfermedad
11	6	10/05/2002	15:00	Injustificada
12	6	20/11/2002	15:00	Injustificada
13	7	15/09/2001	20:00	Enfermedad
14	7	31/08/2002	20:00	Injustificada
15	5	11/04/2003	12:00	Injustificada
16	5	12/04/2002	13:00	Enfermedad

```
SELECT Motivo, COUNT(*)  
FROM InasistAlum  
GROUP BY Motivo  
ORDER BY 2 DESC
```



Motivo	Count
Injustificada	8
Enfermedad	6
Deportes	2

5.3 Funciones de Agrupación

Número de inasistencias en total de cada alumno.

```
SELECT idAlumno, COUNT(*) Faltas
FROM InasistAlum
GROUP BY idAlumno
ORDER BY Faltas DESC
```

Número de inasistencias de cada alumno por mes, sin tomar en cuenta el año, ordenadas por alumno y el numero de faltas de más a menos.

```
SELECT idAlumno, MONTH(Fecha) Mes, COUNT(*) Faltas
FROM InasistAlum
GROUP BY idAlumno, MONTH(Fecha)
ORDER BY idAlumno ASC, Faltas DESC
```

5.3 Funciones de Agrupación

Obtener el total de inasistencias durante cierto periodo, agrupadas por motivo:

```
SELECT Motivo, COUNT(*)  
FROM InasistAlum  
WHERE Fecha BETWEEN '2001-03-01' AND '2001-03-06'  
GROUP BY Motivo
```

Encontrar el horario más temprano en el que hay inasistencias para cada uno de los motivos.

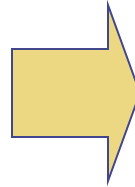
```
SELECT Motivo,  
       MIN(Hora) HoraPrimeraFalta, MAX(Hora) HoraUltimaFalta  
FROM InasistAlum  
GROUP BY Motivo  
ORDER BY 2
```

5.3 Funciones de Agrupación

Número de inasistencias en total de cada alumno y número de días distintos en los que tuvo faltas ordenado en primer lugar en base al número total de días en los que tuvo faltas.

relación **InasistAlum**

IdAlumno	Fecha	Hora	Motivo
1	05/03/2001	16:00	Deportes
1	06/03/2001	16:00	Enfermedad
1	07/03/2001	16:00	Injustificada
1	08/03/2001	16:00	Injustificada
3	01/03/2001	09:00	Enfermedad
3	01/03/2001	10:00	Injustificada
3	01/03/2001	11:00	Injustificada
4	09/03/2001	16:00	Enfermedad



Relación RESULTANTE

IdAlumno	TotFaltas	TotDias
1	4	4
3	3	1
4	1	1

SELECT

 idAlumno, **COUNT**(*) TotalFaltas,
 COUNT(**DISTINCT** Fecha) TotalDias

FROM InasistAlum

GROUP BY idAlumno

ORDER BY TotalDias **DESC**, TotalFaltas **DESC**

5.3 Funciones de Agrupación

Número de alumnos que vive en cada colonia.

```
SELECT Colonia, COUNT(*)  
FROM Alumnos  
GROUP BY Colonia  
ORDER BY 2 DESC
```

```
SELECT Colonia, COUNT(*)  
FROM Alumnos  
GROUP BY Colonia  
HAVING COUNT(*) > 50  
ORDER BY 2 DESC
```

Observe que es básicamente la misma expresión de arriba excepto por la cláusula HAVING que ocasiona que la tabla resultante solo incluya las Colonias con número de estudiantes **mayor a 50**.

5.3 Funciones de Agrupación

Contar el número de materias

```
SELECT COUNT(*) from Materias  
SELECT COUNT(Nombre) from Materias  
SELECT COUNT(Creditos) from Materias
```

Contar el número distinto de créditos de las materias

```
SELECT COUNT(distinct Creditos)  
from Materias
```

En estos ejemplos, al no llevar **GROUP BY**, se formará solo un grupo (considerando todas las tuplas de la tabla) y en la tabla resultante solo aparece una tupla.

5.3 Funciones de Agrupación

Encontrar el número máximo de créditos.

```
SELECT MAX(Creditos) FROM Materias
```

Número mínimo de créditos.

```
SELECT MIN(Creditos) FROM Materias
```

Número de créditos promedio de todas las materias.

```
SELECT AVG(Creditos) FROM Materias
```

O

```
SELECT SUM(Creditos)/COUNT(*) FROM Materias
```

5.4 Consultas sobre Múltiples Tablas

5.4.1 Subconsultas

Diferencia e intersección

Las operaciones **MINUS** e **INTERSECT** eran parte del SQL original, pero se hizo innecesaria por la incorporación del operador de conjuntos **IN** (para intersección, **NOT IN** en vez de MINUS).

5.4.1 Subconsultas

Diferencia e intersección

Para encontrar el número de créditos mayor de la tabla Materias, es suficiente con:

```
SELECT MAX(Creditos) FROM Materias
```

Sin embargo, como ejercicio, escribamos la expresión siguiente en SQL:

$\Pi_{\text{creditos}}(\text{Materias}) -$

$(\Pi_{\text{Materias.creditos}}(\sigma_{\text{Materias.creditos} < \text{Materias2.creditos}}(\text{Materias} \times \rho_{\text{Materias2}}(\text{Materias})))$

```
SELECT
    DISTINCT Creditos
FROM
    Materias
WHERE
    Creditos NOT IN
    (SELECT Materias.Creditos
     FROM Materias, Materias Mat2
     WHERE Materias.Creditos < Mat2.Creditos)
```

5.4.1 Subconsultas Diferencia e intersección

Para obtener los números de Control de los alumnos que no son maestros.

$$\Pi_{\text{Alumnos.control}} \left(\sigma_{\text{Alumnos.curp}=\text{Temporal.curp}} \left(\text{Alumnos} \times \rho_{\text{Temporal}} \left(\Pi_{\text{curp}}(\text{Alumnos}) - \Pi_{\text{curp}}(\text{Maestros}) \right) \right) \right)$$

Ejercicio:

Escriba un comando SQL equivalente a esta expresión del Álgebra Relacional.

5.4.1 Subconsultas

Diferencia e intersección

En el ejercicio anterior se obtiene una expresión algo compleja, en realidad, con el uso del operador **IN** (en este caso **NOT IN**) se puede resolver el mismo problema de forma muy simple:

```
SELECT NumControl
FROM Alumnos
WHERE Curp NOT IN
      (SELECT Curp FROM Maestros)
```

5.4.1 Subconsultas

Diferencia e intersección

Y para obtener los nombres y claves de las materias con mayor número de créditos, la expresión en SQL sería simple:

```
SELECT
    Nombre,Clave
FROM
    Materias
WHERE
    Creditos IN
    (SELECT MAX(Creditos) FROM Materias)
```

5.4.1 Subconsultas

Diferencia Diferencia e intersección

El Nombre y Número de Control de los Alumnos
que no son Maestros:

```
SELECT
    Nombre, NumControl
FROM
    Alumnos
WHERE
    Curp NOT IN (SELECT Curp FROM Maestros)
```

5.4.1 Subconsultas

Diferencia e intersección

Encontrar el RFC, Nombre y Máximo grado de estudios de aquellos maestros que son también alumnos.

```
SELECT
    RFC, Nombre, MaximoGradoEstudios
FROM
    Maestros
WHERE
    Curp IN
    (SELECT Curp FROM Alumnos)
```

5.4.2 Operadores JOIN

```
SELECT *  
FROM  
    Alumnos  
INNER JOIN  
    InasistAlum ON Alumnos.IdAlumno=InasistAlum.IdAlumno
```

INNER (interno) es una palabra opcional, **JOIN** por sí mismo implica un producto cartesiano.

*La expresión de arriba (sea **INNER JOIN** o **JOIN**) equivale exactamente a:*

```
SELECT *  
FROM Alumnos, InasistAlum  
WHERE Alumnos.idAlumno=InasistAlum.idAlumno
```

5.4.2 Operadores JOIN

```
SELECT *  
FROM
```

```
Alumnos
```

```
LEFT OUTER JOIN
```

```
InasistAlum ON Alumnos.IdAlumno=InasistAlum.IdAlumno
```

Pruebe ahora la consulta cambiando
INNER JOIN por LEFT JOIN.

- **OUTER** (externo) es una palabra opcional, ya que LEFT o RIGHT implican una combinación externa.
- **LEFT** significa que deben aparecer todas las tuplas de la tabla de la izquierda, aunque no haya tuplas correspondientes en la tabla de la derecha (*tuplas **fuera** de la condición*).
- **RIGHT** significa que deben aparecer todas las tuplas de la tabla de la derecha, aunque no haya tuplas correspondientes en la tabla de la izquierda (*tuplas **fuera** de la condición*).

5.4.2 Operadores JOIN

```
SELECT DISTINCT
    NumControl, Nombre, Fecha
FROM
    Alumnos
LEFT JOIN
    InasistAlum ON Alumnos.idAlumno=InasistAlum.idAlumno
ORDER BY
    NumControl, Fecha
```

Se obtiene una tabla que contiene una lista de **todos los alumnos** con las fechas en las que tuvo inasistencias. Los alumnos sin ninguna falta también aparecen (con un valor NULL en la Fecha)

5.4.2 Operadores JOIN

Aprovechando el resultado de la consulta de la diapositiva anterior, podemos obtener el Número de Control y Nombre de los alumnos que no han tenido ninguna inasistencia.

```
SELECT DISTINCT
```

```
    NumControl, Nombre
```

```
FROM
```

```
    Alumnos
```

```
LEFT JOIN
```

```
    InasistAlum ON Alumnos.IdAlumno=InasistAlum.IdAlumno
```

```
WHERE
```

```
    Fecha IS NULL
```

Esta condición produce los datos únicamente de quienes no han tenido ninguna falta

5.4.2 Operadores JOIN

Tabla CALIFICACIONES

IdCalificacion	IdAlumno	IdMateria	IdPeriodo	Nota	Etap
1	1	1	5	70	Especial
2	1	2	3	71	Regul1
3	1	3	1	0	Regul2
4	2	3	7	90	Normal
5	4	1	5	95	Normal
6	4	4	5	100	Normal

Tabla MATERIAS

IdMateria	Clave	Nombre	Créditos	HsTeoria	HsPractica
1	8807	Estructuras de Datos I	8	4	0
2	8815	Estructuras de Datos II	8	4	0
3	8927	Bases de Datos I	10	4	2
4	8821	Bases de Datos I	8	4	0

Tabla Alumnos

idAlumno	NumControl	Nombre	Domicilio	FechaNac	Curp
1	98040151	Parejita López	Zarco 123	07/02/1981	LP-11
2	97040587	Alberto M. Alvarado	Negrete 1002	24/03/1882	AMA-22
3	97040014	Mario Molina	Fresno 1410	23/06/1931	MM-33
4	96040121	Benito Juárez García	Patoni 100	21/03/1806	JGB-44

Obtener una lista con el nombre del alumno, nombre de la materia y calificación obtenida.

```
SELECT Alumnos.Nombre, Materias.Nombre, Nota
FROM
    Calificaciones
INNER JOIN
    Alumnos ON Calificaciones.IdAlumno=Alumnos.IdAlumno
INNER JOIN
    Materias ON Calificaciones.IdMateria=Materias.IdMateria
```

5.4.2 Operadores JOIN

La tabla **Grupos** de abajo muestra los datos de los grupos ofrecidos cada semestre. **GruposMaestros** relaciona las tablas Grupos y Maestros. En esa tabla hay un grupo que no está relacionado, significa que no tiene profesor asignado. Hay que crear ambas tablas para resolver las consultas de la página siguiente.

Grupos

IdGrupo	IdMateria	Periodo	Paquete	MaxAlum	Horario
1	5	2005A	4B	35	L Ma Mi J V S 7-9 T9
2	2	2005A	4Z	30	L Ma Mi J V S 9-11 T4
3	3	2003B	6X	30	L Ma Mi J 10-11 T1
4	3	2001V	6X	35	L Ma J V 11-12 T5
5	4	2007A	3C	35	L Mi J V 7-8 T1
6	4	2007A	3D	30	L Mi J V 16-17 T10
7	1	2002V	3X	25	L Ma Mi J 10-11 T1
8	1	2002V	3Y	25	L Ma Mi J 9-10 T9

GruposMaestros

IdGpoMae	IdGrupo	IdMaestro
1	1	2
2	2	2
3	3	1
4	4	5
5	7	6
6	6	2

Maestros

IdMaestro	Nombre	Domicilio	FechaNac	CURP
1	Xavier Lopez Chabelo	Templo	17/12/1961	X11
2	Aristóteles	Coliseo	20/11/1965	A22
3	Tchaikovski	Insurgentes	03/05/1964	T44
4	Robin Hood	Aquiles	20/03/1979	RH99
5	Botticelli Sandro	Cataluña ...	29/03/1963	B33
6	Fernández José Ramón	Coapa ...	31/07/1953	F77

5.4.2 Operadores JOIN

SELECT

M.idMaestro, idMateria, Periodo, Paquete, Horario, M.Nombre

FROM

Grupos G

INNER JOIN

GruposMaestro GM **ON** G.idGrupo=GM.idGrupo

INNER JOIN

Maestros M **ON** GM.idMaestro=M.idMaestro

SELECT

M.idMaestro, idMateria, Periodo, Paquete, Horario, M.Nombre

FROM

Grupos G

LEFT JOIN

GruposMaestro GM **ON** G.idGrupo=GM.idGrupo

LEFT JOIN

Maestros M **ON** GM.idMaestro=M.idMaestro

Con la primera consulta (**INNER JOIN**) se obtiene una tabla solo con los grupos que tienen profesor asignado, con **LEFT JOIN** se obtiene una tabla de todos los grupos, tengan profesor asignado o no.

5.4.2 Operadores JOIN

PRODUCTO NATURAL

- **NATURAL JOIN** no es un standard de SQL, **SQL Server** no lo reconoce.
- Aunque está soportado por **MySQL**, depende fuertemente de que los nombres de las columnas se llamen exactamente igual, sean del mismo tipo y tengan el mismo significado.
 - Por ejemplo, si hay una columna llamada *Nombre* en dos tablas distintas y se hace *Producto Natural* entre ellas, va a relacionarlas y el resultado seguramente no sería el esperado.

```
SELECT *  
FROM Alumnos  
NATURAL JOIN InasistAlum
```

Para probar esta consulta, hay que seguir las instrucciones de las dos diapositivas siguientes.

5.4.2 Operadores JOIN PRODUCTO NATURAL

1. Descargue **MySQL** y **Workbench** de la liga siguiente
<https://felipealanis.org/Cursos/Fundamentos%20de%20Bases%20de%20Datos/Unidad%205/>
2. Instale ambos archivos.
3. Cree las tablas de esta página y la siguiente, también añada las tuplas que se indican para que pueda probar la expresión de la pagina anterior.

```
CREATE DATABASE SQLejemplos;  
USE SQLejemplos;  
CREATE TABLE Alumnos(  
    idAlumno INT AUTO_INCREMENT PRIMARY KEY NOT NULL,  
    NumControl VARCHAR(10) UNIQUE,  
    Nombre VARCHAR(30),  
    Domicilio VARCHAR(30),  
    FechaNacimiento DATE NOT NULL,  
    Curp VARCHAR(18) NOT NULL)
```

```
INSERT INTO Alumnos VALUES (0, '98042151', 'Pato Lucas', 'Zarco 123', '19810207', 'LP11');  
INSERT INTO Alumnos VALUES (0, '97041587', 'Atomic Ant', 'Negrete 1002', '19820324', 'AA22');  
INSERT INTO Alumnos VALUES (0, '97043014', 'Mickey Mouse', 'Fresno 1410', '19810623', 'MM33');  
INSERT INTO Alumnos VALUES (0, '96042121', 'Johnny Bravo', 'Patoni 100', '19820331', 'BJ44');  
INSERT INTO Alumnos VALUES (0, '96044350', 'Robin Hood', 'Aquiles Serdán 1102', '19790320', 'RH99');  
INSERT INTO Alumnos VALUES (0, '97111432', 'Oso Yogui', 'Victoria 345', '19800411', 'OY88');  
INSERT INTO Alumnos VALUES (0, '98042012', 'Hércules', 'Juárez 543', '19810924', 'HH66');
```

5.4.2 Operadores JOIN

PRODUCTO NATURAL

```
CREATE TABLE InasistAlum (  
    idInasistAlum INT NOT NULL AUTO_INCREMENT,  
    idAlumno INT NOT NULL,  
    Fecha DATE NOT NULL,  
    Hora TIME NOT NULL,  
    Motivo VARCHAR(20) NOT NULL,  
    PRIMARY KEY (idInasistAlum),  
    FOREIGN KEY (idAlumno) REFERENCES Alumnos(idAlumno)  
        ON DELETE RESTRICT  
        ON UPDATE CASCADE,  
    UNIQUE KEY unica_inasistencia (idAlumno, Fecha, Hora))
```

```
INSERT INTO InasistAlum VALUES (0,1,'20010305','16:00','Deportes');  
INSERT INTO InasistAlum VALUES (0,1,'20010306','16:00','Enfermedad');  
INSERT INTO InasistAlum VALUES (0,1,'20010307','16:00','Injustificada');  
INSERT INTO InasistAlum VALUES (0,1,'20010308','16:00','Injustificada');  
INSERT INTO InasistAlum VALUES (0,3,'20010301','09:00','Enfermedad');  
INSERT INTO InasistAlum VALUES (0,3,'20010301','10:00','Injustificada');  
INSERT INTO InasistAlum VALUES (0,3,'20010301','11:00','Injustificada');  
INSERT INTO InasistAlum VALUES (0,4,'20010309','16:00','Enfermedad');
```

5.5 Vistas

- La creación de vistas permite que los usuarios técnicos, tengan acceso a los datos de una forma más conveniente, simplificando los esquemas.
 - En una Base de Datos bien diseñada, los datos se encuentran dispersos en varias tablas para evitar redundancia, pero se hacen más complejos los esquemas y las consultas se obtienen con mayor esfuerzo.
- Las vistas se crean a partir de expresiones de consulta y **siguen siendo consultas**. Aunque parecen tablas y se tratan como tales, son consultas que se ejecutan cada vez que se usan.

5.5 Vistas

InasistAlum

idInasistAlum	idAlumno	Fecha	Hora	Motivo
1	1	05/03/2001	16:00	Deportes
2	1	06/03/2001	16:00	Enfermedad
3	1	07/03/2001	16:00	Injustificada
4	1	08/03/2001	16:00	Injustificada
5	3	01/03/2001	09:00	Enfermedad
6	3	01/03/2001	10:00	Injustificada
7	3	01/03/2001	11:00	Injustificada
8	4	09/03/2001	16:00	Enfermedad
9	5	25/10/2002	10:00	Deportes
10	5	18/04/2003	12:00	Enfermedad
11	6	10/05/2002	15:00	Injustificada
12	6	20/11/2002	15:00	Injustificada
13	7	15/09/2001	20:00	Enfermedad
14	7	31/08/2002	20:00	Injustificada
15	5	11/04/2003	12:00	Injustificada
16	5	12/04/2002	13:00	Enfermedad

Ejemplo:

A partir de estas tablas, escriba una expresión de consulta para obtener el Número de Control, Nombre, Fecha, Hora y Motivo de los alumnos que tienen faltas.

Alumnos

IdAlumno	Control	Nombre	Calle	NumExt	NumInt	Colonia	FechaNac	CURP
1	98042151	Pato Lucas	Zarco	123		Centro	07/02/1981	LP11
2	97041587	Atomic Ant	Negrete	1002		Fracc. del Lago	24/03/1982	AA22
3	97043014	MickeyMouse	Fresno	1410	B	Real del Prado	23/06/1981	MM33
4	96042121	Johnny Bravo	Patoni	100		Centro	31/03/1982	JB44
5	96044350	Robin Hood	Aquiles Serdán	1102		Nueva Vizcaya	20/03/1979	RH99
6	97111432	Oso Yogui	Victoria	345		Centro	11/04/1980	OY88
7	98042012	Hércules	Real	300	1	Loma Dorada	24/09/1981	HH66

5.5 Vistas

Pruebe primero la consulta y luego añada y ejecute el CREATE VIEW para crear la vista

```
CREATE VIEW vInasistAlum AS
```

```
SELECT
```

```
    NumControl, Nombre, Fecha FechaFalta, Hora HoraFalta, Motivo
```

```
FROM
```

```
    Alumnos
```

```
INNER JOIN
```

```
    InasistAlum ON Alumnos.idAlumno=InasistAlum.idAlumno
```

```
SELECT * FROM vInasistAlum
```

```
SELECT NumControl, Nombre, COUNT(*) TotalFaltas
```

```
FROM vInasistAlum
```

```
GROUP BY NumControl, Nombre
```

5.5 Vistas

```
SELECT Nombre, FechaFalta  
FROM vInasistAlum  
WHERE HoraFalta='16:00'
```

```
SELECT NumControl, Nombre, FechaFalta, HoraFalta  
FROM vInasistAlum  
WHERE Motivo='Injustificada'  
ORDER BY Nombre
```

```
SELECT NumControl, Nombre, COUNT(*) TotalFaltas,  
       COUNT(DISTINCT FechaFalta) TotalDias  
FROM vInasistAlum  
GROUP BY NumControl, Nombre  
ORDER BY TotalDias DESC, TotalFaltas DESC
```

5.5 Vistas

Observaciones importantes respecto a las Vistas

- Cuando se hace una consulta sobre una vista, se realiza la consulta correspondiente a la vista y a continuación, ya con la tabla resultante se hace la consulta solicitada.
 - En realidad el optimizador de consultas puede combinar ambas consultas para invertir menos tiempo en la solución pero el resultado es como si se hicieran una luego de la otra.
- La **redundancia** de datos de una vista es **aparente** ya que los datos se encuentran en las tablas correspondientes.
- Las expresiones usando vistas **son mas simples** y permiten que el **DBA** diseñe correctamente el esquema pero manipular los datos usando vistas para obtener más fácilmente las consultas.
- La mayoría de los DBMS's permiten a los usuarios hacer ***update*** sobre una vista.
 - los datos se actualizan en las tablas originales.

Ejercicio

- A partir de las tablas de la pagina siguiente, escriba las vistas necesarias para obtener una consulta donde aparezcan los datos de los grupos ofrecidos, tengan o no profesor asignado.
- Deben mostrarse la clave y nombre de la materia, el nombre del profesor, además del resto de los atributos de los grupos.
- Puede crear **vistas anidadas** (vistas a partir de otras vistas o tablas).
- Se sugiere crear vistas de 2 o 3 tablas/vistas para facilitar la resolución de la consulta.

5.4.2 Operadores JOIN

Grupos

IdGrupo	IdMateria	Periodo	Paquete	MaxAlum	Horario
1	5	2005A	4B	35	L Ma Mi J V S 7-9 T9
2	2	2005A	4Z	30	L Ma Mi J V S 9-11 T4
3	3	2003B	6X	30	L Ma Mi J 10-11 T1
4	3	2001V	6X	35	L Ma J V 11-12 T5
5	4	2007A	3C	35	L Mi J V 7-8 T1
6	4	2007A	3D	30	L Mi J V 16-17 T10
7	1	2002V	3X	25	L Ma Mi J 10-11 T1
8	1	2002V	3Y	25	L Ma Mi J 9-10 T9

GruposMaestro

IdGpoMae	IdGrupo	IdMaestro
1	1	2
2	2	2
3	3	1
4	4	5
5	7	6
6	6	2

Maestros

IdMaestro	Nombre	Domicilio	FechaNac	CURP
1	Xavier Lopez Chabelo	Templo	17/12/1961	X11
2	Aristóteles	Coliseo	20/11/1965	A22
3	Tchaikovski	Insurgentes	03/05/1964	T44
4	Robin Hood	Aquiles	20/03/1979	RH99
5	Botticelli Sandro	Cataluña ...	29/03/1963	B33
6	Fernández José Ramón	Coapa ...	31/07/1953	F77

Materias

IdMateria	Clave	Nombre	Creditos
1	8807	Estructuras de Datos I	8
2	8821	Bases de Datos I	6
3	8011	Matemáticas IV	4
4	8927	Bases de Datos I	8
5	8806	Programación II	10
6	8126	Bioingeniería	10

5.6 Manipulación de la Base de Datos

Comandos de **DDL**

CREATE DATABASE ITD

Nombre de la Base de Datos

USE ITD

Los comandos se referirán a esta Base de Datos

CREATE TABLE Alumnos

(IdAlumno **INT IDENTITY PRIMARY KEY**,
NumControl **VARCHAR(8) UNIQUE**,
Curp **VARCHAR(18) UNIQUE**,
Paterno **VARCHAR(20)**,
Materno **VARCHAR(20)**,
Nombre **VARCHAR(20)**,
Calle **VARCHAR(15)**,
NumeroExt **INT**,
NumeroInterior **VARCHAR(6)**,
Colonia **VARCHAR(15)**,
FechaNacimiento **DATE**)

El DBMS
impedirá que
se añadan
tuplas con
valores iguales
en las llaves
únicas

La llave
primaria
permite
establecer
integridad
referencial

5.6 Manipulación de la Base de Datos

```
CREATE TABLE Materias (  
  IdMateria INT IDENTITY PRIMARY KEY,  
  Clave VARCHAR(4) UNIQUE,  
  Nombre VARCHAR(20),  
  Creditos INT,  
  HorasTeoria INT,  
  HorasPractica INT)
```

Mediante esta sintaxis, se establece integridad referencial.

```
CREATE TABLE Calificaciones  
  idCalificacion INT IDENTITY PRIMARY KEY,  
  idAlumno INT REFERENCES Alumnos,  
  idMateria INT REFERENCES Materias,  
  Oportunidad VARCHAR(20),  
  Resultado INT,  
  UNIQUE (IdAlumno, IdMateria))
```

IdAlumno e
IdMateria son
llaves foráneas

5.6 Manipulación de la Base de Datos

```
ALTER TABLE ALUMNOS ADD TUTOR CHAR (30)  
SELECT * FROM Alumnos
```

```
ALTER TABLE ALUMNOS MODIFY COLUMN TUTOR CHAR(45)  
SELECT * FROM Alumnos
```

```
ALTER TABLE ALUMNOS DROP TUTOR  
SELECT * FROM Alumnos
```

5.6 Manipulación de la Base de Datos

Comandos **DML**

INSERT INTO (Añadir tuplas a una tabla)

Materias

IdMateria	Clave	Nombre	Creditos	HorasTeo	HorasPra
1	1810	Estructuras de Datos	10	4	2
2	1533	Inv. De Operaciones II	8	4	0
3	1001	Matemáticas I	8	4	0
4	1805	Probabilidad y Estadística	6	3	0
5	1806	Programación orientada a objetos	8	3	2
6	1813	Fundamentos de Bases de Datos	8	4	0

INSERT INTO

Materias

(Clave, Nombre, HorasTeoria, HorasPractica)

VALUES

('1215', 'DIBUJO', 4, 4, 0)

5.6 Manipulación de la Base de Datos

relación **InasistAlum**

IdAlumno	Fecha	Hora	Motivo
1	05/03/2001	16:00	Deportes
1	06/03/2001	16:00	Enfermedad
1	07/03/2001	16:00	Injustificada
1	08/03/2001	16:00	Injustificada
3	01/03/2001	09:00	Enfermedad
3	01/03/2001	10:00	Injustificada
3	01/03/2001	11:00	Injustificada
4	09/03/2001	16:00	Enfermedad

DELETE FROM <tabla> WHERE <expresión>

```
DELETE FROM
  InasistAlumnos
WHERE
  idAlumno IN
    (SELECT idAlumno
     FROM InasistAlumnos
     GROUP BY idAlumno
     HAVING COUNT(*)=1)
```

Ejemplo:

Eliminar las tuplas de inasistencia de los alumnos que tuvieron solo una.

5.6 Manipulación de la Base de Datos

Materias

IdMateria	Clave	Nombre	Creditos	HorasTeo	HorasPra
1	1810	Estructuras de Datos	10	4	2
2	1533	Inv. De Operaciones II	8	4	0
3	1001	Matemáticas I	8	4	0
4	1805	Probabilidad y Estadística	6	3	0
5	1806	Programación orientada a objetos	8	3	2
6	1813	Fundamentos de Bases de Datos	8	4	0

UPDATE <tabla>

SET <expresión asignación> **WHERE** <condición>

UPDATE

Materias

SET

Creditos=HorasTeoria*3+HorasPractica*2

WHERE

Clave LIKE '18%'

Modifica el número de créditos de todas las tuplas de acuerdo con la formula indicada.

5.6 Manipulación de la Base de Datos

Califics

IdAlumno	IdMateria	Result
1	1	100
1	2	90
1	5	77
2	1	100
2	3	98
2	4	0
3	1	87
3	2	99
3	3	76
4	2	100
4	5	90
4	6	80
5	1	88
6	2	66
6	4	82
7	2	100

Disminuye un punto de cada calificación a los alumnos con más de 2 faltas

relación **InasistAlum**

IdAlumno	Fecha	Hora	Motivo
1	05/03/2001	16:00	Deportes
1	06/03/2001	16:00	Enfermedad
1	07/03/2001	16:00	Injustificada
1	08/03/2001	16:00	Injustificada
3	01/03/2001	09:00	Enfermedad
3	01/03/2001	10:00	Injustificada
3	01/03/2001	11:00	Injustificada
4	09/03/2001	16:00	Enfermedad

```
UPDATE Calificaciones
SET Resultado = Resultado - 1
WHERE idAlumno IN
(SELECT idAlumno
FROM InasistAlumnos
GROUP BY idAlumno
HAVING COUNT(*) > 2)
```