Unidad 2
Lenguaje de Manipulación de Datos (DML)

2.1 Inserción, eliminación, modificación.
2.2 Consultas y Vistas.
2.3 Funciones de agrupación.
2.4 Subconsultas.
2.5 Operadores de Conjuntos.
2.6 Inserción, eliminación y modificación de tuplas en forma múltiple.

2.1 Insertar, eliminar, modificar.

Para poner en práctica las expresiones de esta unidad debe usar uno de los esquemas creados en la unidad anterior.

- Los comandos DML mostrados en las diapositivas están diseñados a propósito con diferencias respecto al esquema de la Unidad 1 por lo que usted deberá hacer los ajustes necesarios a las expresiones.

- ***Insert***.
  Añadir una tupla a una tabla.
- ***Delete***.
  Eliminar una tupla de una tabla.
- ***Update***.
  Modificar uno o más de los atributos de una tupla.

Tabla **ALUMNOS**

IdAlumno	NumControl	EscuelaProcede	IdPersona
1	98040151	Prepa PUMAS	1
2	97040587	Palacio Nacional	3
3	97040014	Colegio Vizcaya	6
4	96040121	LPGA	7
5	98040150	Colegio Alemán	2

tabla **InasistAlum**

IdInasistAlum	IdAlumno	FechaHora	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

Tabla **PERSONAS**

IdPersona	Nombre	Apellidos	Calle	NumExt	Poblacion	Pais	FechaNac	Sexo	CURP
1	Parejita	López	Zarco	123	Cd de México	México	07-02-1981	Hombre	L1
2	Johaness	Gutenberg	Carlomagno	1	Mainz	Alemania	12-01-1398	Hombre	G2
3	Benito	Juárez García	Monte Albán	100	Cd de México	México	21-03-1806	Hombre	J4
4	Luis	Pasteur	Campos Elíseos	234	París	Francia	20-03-1850	Hombre	P1
5	Abraham		Calle del	347	Jerusalén	Israel	11-04-1890	Hombre	A0
6	José	Revueltas	Negrete	1002	Durango	México	24-03-1982	Hombre	R7
7	Lorena	Ochoa	Fresno	1410	Guadalajara	México	23-06-1981	Mujer	O1
8	Aristóteles		Templo Atenea	542	Atenas	Grecia	23-07-1905	Hombre	A1
9		Tchaikovski	Plaza Roja	471	Moscú	Rusia	13-08-1920	Hombre	T4
10		Botticelli	Filipo Lippi	2	Florencia	Italia	07-09-1919	Hombre	B9
11	José Luis	López	Francisco Zarco	123	Cd de México	México	09-03-1961	Hombre	JLL01
12	Friele	Gensfleisch	Carlomagno	1	Mainz	Alemania	09-03-1370	Hombre	GF7
13	Antonio	Maza	Guelaguetza	201	Oaxaca	México	15-05-1780	Hombre	AM01
14	Margarita	Maza	Monte Albán	100	Cd de México	México	19-08-1820	Mujer	MM01


```
set dateformat DMY
insert into
    Personas(Nombre,Apellidos,Calle,NumExt,Poblacion,Pais,FechaNac,Sexo,Curp)
Values
    ('Parejita','López','Zarco',123,'Cd de México','México','07-02-1981','Hombre','L1'),
    ('Johaness','Gutenberg','Carlomagno',1,'Mainz','Alemania','12-01-1398','Hombre','G2'),
    ('Benito','Juárez García','Monte Albán',100,'Cd de México','México','21-03-1806','Hombre','J4'),
    ('Luis','Pasteur','Campos Elíseos',234,'París','Francia','20-03-1850','Hombre','P1'),
    ('Abraham','Zabludovsky','Calle del camello',347,'Jerusalén','Israel','11-04-1890','Hombre','A0'),
    ('José','Revueltas','Negrete',1002,'Durango','México','24-03-1982','Hombre','R7'),
    ('Lorena','Ochoa','Fresno',1410,'Guadalajara','México','23-06-1981','Mujer','O1'),
    ('Aristóteles','Onasis','Templo Atenea',542,'Atenas','Grecia','23-07-1905','Hombre','A1'),
    ('Piotr','Chaikovski','Plaza Roja',471,'Moscú','Rusia','13-08-1920','Hombre','T4'),
    ('Sandro','Botticelli','Filipo Lippi',2,'Florencia','Italia','07-09-1919','Hombre','B9'),
    ('José Luis','López','Francisco Zarco',123,'Cd de México','México','09-03-1961','Hombre','JLL01'),
    ('Friele','Gensfleisch','Carlomagno',1,'Mainz','Alemania','09-03-1370','Hombre','GF7'),
    ('Antonio','Maza','Guelaguetza',201,'Oaxaca','México','15-05-1780','Hombre','AM01'),
    ('Margarita','Maza','Monte Albán',100,'Cd de México','México','19-08-1820','Mujer','MM01')
```

2.1 Insertar, eliminar, modificar.

Algunos detalles respecto a la sintaxis básica de ***insert***:

- No se debe asignar un valor para la llave primaria.
    - Al ser llave identidad, el DBMS asigna la id siguiente de manera automática.
- No se debe asignar un valor para las columnas calculadas (*computed column*).
    - El DBMS calcula el valor de acuerdo a la expresión correspondiente.
- Los atributos que permiten nulos pueden omitirse, pero se debe dejar la coma separadora correspondiente.

2.1 Insertar, eliminar, modificar.

Es posible que le resulten algunos errores al ejecutar la operación ***insert***, por ejemplo:

- ***Cannot insert the value NULL into column 'x'***
  - Está omitiendo el valor de un atributo o asignando NULL pero está indicado en el esquema que no se admiten valores nulos.
  - Si el atributo es llave identidad es posible que no lo haya establecido como llave primaria y llave identidad pero usted cree que lo hizo y no lo asigna.
- **There are fewer columns in the INSERT statement than values specified in the VALUES clause.**
  - El número de atributos y valores en las listas correspondientes no coincide.

2.1 Insertar, eliminar, modificar.

Otros errores al ejecutar la operación ***insert***:

- ***Cannot insert explicit value for identity column in table 'x'***
  - No se debe indicar un valor explícito para la llave identidad e una table.
- ***The conversion of a varchar data type to a datetime data type resulted in an out-of-range value.***
  - Conviene indicar el orden de las partes de una fecha mediante set dateformat DMY/MDY ya que por default puede no coincidir con lo que usted cree.

2.1 Insertar, eliminar, modificar.

Cuando falla un ***insert***, la ***id*** correspondiente se incrementa aunque la tupla no haya sido añadida, por lo que usted observará que los valores de las *id* no serán consecutivos. Lo anterior no tiene relevancia ya que los id's son datos internos que no tienen significado para los usuarios de las aplicaciones de la Base de Datos.

Sin embargo, si desea reiniciar los valores de las llaves identidad, para efectos de hacer las pruebas que se solicitan en estos ejercicios, puede eliminar las tuplas empezando por las tablas dependientes y finalizando por las tablas dominantes y luego reiniciar el conteo de las llaves primarias usando las siguientes instrucciones:

delete from *nombre de la tabla* -- borra todas las tuplas de la tabla
DBCC CHECKIDENT('*NombreTabla*' , RESEED, 0) -- reinicia el valor de la id

2.1 Insertar, eliminar, modificar.

```
insert into Alumnos(NumControl,EscuelaProcede,idPersona)
Values    ('98040151','Prepa PUMAS',1),
          ('97040587','Palacio Nacional',3),
          ('97040014','Colegio Vizcaya',6),
          ('96040121','LPGA',7),
          ('98040150','Colegio Alemán',2)
```

```
insert into AlumnosInasist(idAlumno,FechaHora,Motivo)
   values(1,'2001-03-05 16:00','Deportes'),
          (1,'2001-03-06 16:00','Enfermedad'),
          (1,'2001-03-07 16:00','Injustificada'),
          (1,'2001-03-08 16:00','Injustificada'),
          (3,'2001-03-01 09:00','Enfermedad'),
          (3,'2001-03-01 10:00','Injustificada'),
          (3,'2001-03-01 11:00','Injustificada'),
          (4,'2001-03-09 16:00','Enfermedad’)
```

2.1 Insertar, eliminar, modificar.

Otros errores muy frecuentes:

- ***The INSERT statement conflicted with the CHECK constraint "X"***
  - Uno de los valores insertados viola una restricción *Check*.
- ***Violation of UNIQUE KEY constraint 'X'. Cannot insert duplicate key in object 'Y'.***
  - Se está intentando registrar una tupla con un valor de llave única que ya se encuentra en una tupla existente.
- ***The INSERT statement conflicted with the FOREIGN KEY constraint "Z".***
  - Violación de integridad referencial. Se está intentando registrar una tupla en una tabla dependiente y el valor de la ***id*** relacionada no es un valor existente en la tabla dominante.

2.1 Insertar, eliminar, modificar.

*El **télex** y el **fax** se usaron ampliamente en el siglo XX pero prácticamente están desaparecidos ya que el email los reemplazó.*

```
/* El fax usaba un dispositivo conectado a la línea telefónica,
por lo que se decide conservar el número de teléfono en la BD*/
update FormasContacto
   set Tipo='Teléfono'
   where Tipo='Fax’

/* El télex usaba la infraestructura del telégrafo, esa forma
de contacto ya es obsoleta y se decide eliminar esa información */
delete from FormasContacto
   where Tipo='Télex'
```

2.2.1 Consultas

Obtener las tuplas de las inasistencias a las 4pm:

```
select * from InasistAlum
   where convert(time,FechaHora)='16:00’
```

Se excluye la hora ya que no es necesaria al tener cada tupla el mismo valor:

```
select convert(date,FechaHora),Motivo,idAlumno
   from InasistAlum
   where convert(time,FechaHora)='16:00'
```

2.2.1 Consultas

Obtener las tuplas correspondientes a estudiantes que faltaron por la mañana (antes de las 12pm):

```
select * from InasistAlum
   Where convert(time,FechaHora)<'12:00'
```

tabla **InasistAlum**

IdInasistAlum	IdAlumno	FechaHora	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

IdInasistAlum	IdAlumno	FechaHora	Motivo
5	3	01/03/2001 09:00	Enfermedad
6	3	01/03/2001 10:00	Injustificada
7	3	01/03/2001 11:00	Injustificada

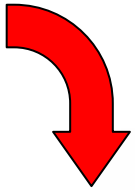
2.2.1 Consultas

Las idAlumno de los estudiantes que faltaron por la mañana:

```
select idAlumno from InasistAlum
   Where convert(time,FechaHora)<'12:00'
```

tabla **InasistAlum**

IdInasistAlum	IdAlumno	FechaHora	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



IdAlumno
3
3
3

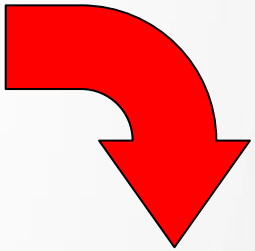
Este resultado no es la respuesta exacta a lo que se solicita porque, si solo faltó el alumno 3 a esa hora, en la lista solo debiera aparecer una vez su i*d.*

2.2.1 Consultas

```
select distinct idAlumno from InasistAlum
Where convert(time,FechaHora)<'12:00'
```

tabla **InasistAlum**

IdInasistAlum	IdAlumno	FechaHora	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



IdAlumno
3

2.2.1 Consultas

Obtener el número de control, la fecha y hora, así como el motivo de inasistencia de quienes faltaron a las 4 pm.

```
select NumControl,FechaHora,Motivo
from Alumnos inner join InasistAlum
      on Alumnos.idAlumno=InasistAlum.idAlumno
where convert(time,FechaHora)='16:00'
```

```
select NumControl,FechaHora,Motivo
from Alumnos a inner join InasistAlum i
      on a.idAlumno=i.idAlumno
where convert(time,FechaHora)='16:00'
```

Para acortar la expresión de consulta se renombran las tablas (solo tiene efecto durante la propia consulta)

2.2.1 Consultas

Obtener el Número de Control, nombre completo, fecha y hora de los alumnos con inasistencias injustificadas.

```
Select NumControl,
       concat_ws(' ',Nombre,Apellidos) NombreCompleto,
       convert(date,FechaHora) FechaFalta,
       convert(time,FechaHora) HoraFalta
from
    Alumnos a
inner join
    Personas p on a.idPersona=p.idPersona
inner join
    AlumnosInasist i on i.idAlumno=a.idAlumno
where i.Motivo='Injustificada'
```

2.2.2 Vistas

Son objetos de la B.D. equivalentes a una expresión de consulta que pueden ser tratados como una tabla **para efectos de consulta** y **combinación** con otras vistas o tablas.

SQL Server y otros DBMS's **permiten la modificación de datos** a través de una vista como si fuera una tabla.

Cada vez que se incluye una vista en otra consulta, se ejecuta la consulta de la vista de manera que los datos estén actualizados permanentemente.

2.2.2 Vistas

SQL Server permite crear vistas a las que les llama ***materializadas***, que cuentan con la característica de que los resultados de las consultas se guardan en la BD, actualizándose continuamente para minimizar el tiempo de obtención de resultados de consultas complejas.

Objetivos de la vistas:

- ✓ Permiten que los **usuarios del DBMS** **vean los datos de la forma más conveniente** de acuerdo a su nivel de conocimientos y experiencia.
- ✓ Incluso para los usuarios especializados, las vistas **simplifican los esquemas** para facilitar la comprensión del esquema y la obtención de consultas más complejas.

2.2.2 Vistas

Tabla **PERSONAS**

IdPersona	Nombre	Apellidos	Calle	NumExt	Poblacion	Pais	FechaNac	CURP
1	Parejita	López	Zarco	123	Cd de México	México	07-02-1981	L1
2	Johaness	Gutenberg	Carlomagno	1	Mainz	Alemania	12-01-1398	G2
3	Benito	Juárez García	Monte Albán	100	Cd de México	México	21-03-1806	J4
4	Luis	Pasteur	Campos Elíseos	234	París	Francia	20-03-1850	P1
5	Abraham		Calle del camello	347	Jerusalén	Israel	11-04-1890	A0
6	José	Revueltas	Negrete	1002	Durango	México	24-03-1982	R7
7	Lorena	Ochoa	Fresno	1410	Guadalajara	México	23-06-1981	O1
8	Aristóteles		Templo Atenea	542	Atenas	Grecia	23-07-1905	A1
9		Tchaikovski	Plaza Roja	471	Moscú	Rusia	13-08-1920	T4
10		Botticelli	Filipo Lippi	2	Florencia	Italia	07-09-1919	B9
11	José Luis	López	Francisco Zarco	123	Cd de México	México	09-03-1961	JLL01
12	Friele	Gensfleisch	Carlomagno	1	Mainz	Alemania	09-03-1370	GF7
13	Antonio	Maza	Guelaguetza	201	Oaxaca	México	15-05-1780	AM01
14	Margarita	Maza	Monte Albán	100	Cd de México	México	19-08-1820	MM01

Tabla **ALUMNOS**

IdAlumno	NumControl	EscuelaProcede	IdPersona
1	98040151	Prepa PUMAS	1
2	97040587	Palacio Nacional	3
3	97040014	Colegio Vizcaya	6
4	96040121	LPGA	7
5	98040150	Colegio Alemán	2

A partir de este esquema, escribir una consulta con los datos de todos los alumnos (ambas tablas).
Esta consulta nos servirá para crear una vista y facilitar otras consultas.

creación de la Vista vAlumnos

```
create view vAlumnos as
select
    a.idAlumno, NumControl, CURP, Nombre, Apellidos,
    concat_ws(' ',Nombre,Apellidos) NombreCompleto,
    Calle, NumExt, Poblacion, Pais, FechaNac, EscuelaProcede,
    a.idPersona
from
    Alumnos a
inner join
    Personas p on a.idPersona=p.idPersona
```

(Permite realizar otras consultas de manera más simple. idAlumno e *idPersona* se incluyen en la Vista para poder relacionarla con otras Vistas o Tablas.

2.2.2 Vistas

Otras ventajas del uso de Vistas:

- **Mecanismo de seguridad**. Los usuarios de los datos (*Usuarios Sofisticados* o programadores inexpertos solo tienen acceso a los datos que el DBA decida).
- **Capa para soporte de código heredado.** Las vistas ayudan a evitar trabajar con las tablas directamente para obtener consultas; por lo tanto, si se hacen cambios importantes a los esquemas, producto de normalización, por ejemplo, se pueden crear Vistas con nombres similares a las tablas obsoletas para disminuir la cantidad de cambios al código.

2.2.2 Vistas

Algunas consultas sobre la Vista *vAlumnos*.

Obtener los datos de los alumnos que ingresaron en el año 1998:

Cláusula LIKE:

```
select NumControl,Nombre,Apellidos
   from vAlumnos
   where NumControl like '98%'
```

LIKE se emplea para hacer una comparación entre una cadena y una plantilla. Verifica si el NumControl de cada tupla ***se parece*** a la plantilla "98%".

Esta plantilla que usa el "%", hace que una tupla se incluya en la consulta resultante si el Numero de Control inicia con los caracteres "98" sin importar lo que vaya después.

2.2.2 Vistas

Una lista de los estudiantes que provienen de un CBTIS.

```
select * from vAlumnos
   where EscuelaProcede like '%CBTIS%'
```

Una lista de los estudiantes que se trasladaron al Tec de Durango provenientes de otro Tec.

```
select * from vAlumnos
   where NumControl not like '__04%'
```

Un guión bajo representa cualquier posición dentro del valor del atributo.

2.2.2 Vistas

```
select *
   from vAlumnos
   order by País, Poblacion
```

La tabla resultante se muestra ordenada por País y Población. Todas las ciudades de un país juntas ordenadas por población.

*Una lista de los datos de los alumnos que nacieron en la **"década de los 40"**, ordenada por Número de Control.*

```
select *
from vAlumnos
where
   year(fechaNacimiento) between 1998 and 2003
```

Evita tener que escribir:

```
year(fechaNacimiento)>=1998 and
year(fechaNacimiento)<=2003
```

2.2.2 Vistas

Ejercicio:

- Ejecute el procedimiento de la diapositiva siguiente para añadir unas 100 tuplas de inasistencias de alumnos con fechas diversas, a la tabla *AlumnosInasist*.
- Antes de ejecutarlo haga los ajustes necesarios de acuerdo al esquema de su Base de Datos.

Inicia un bloque de código

Declara una tabla temporal en memoria

```
BEGIN
    DECLARE @Motivos TABLE (Motivo VARCHAR(30));
    DECLARE @idsAlumnos TABLE (idAlumno INT);
    INSERT INTO @Motivos
    VALUES ('Permiso'), ('Actividad Cultural'), ('Injustificada'),('Enfermedad'),
       ('Deportes');
    INSERT INTO @idsAlumnos
    VALUES (1),(3),(4),(5),(6),(7),(8),(10);
    DECLARE @i INT = 0;
    WHILE @i < 100
    BEGIN
       DECLARE @idAlumno INT= (SELECT TOP 1 idAlumno FROM @idsAlumnos ORDER BY NEWID())
       DECLARE @Motivo VARCHAR(30) = (SELECT TOP 1 Motivo FROM @Motivos ORDER BY NEWID(
       DECLARE @FechaHora DATETIME;
       DECLARE @FechaInicial DATETIME = '2020-01-01 00:00:00';
       DECLARE @FechaFinal DATETIME = '2024-01-01 00:00:00';
       DECLARE @Dias INT = DATEDIFF(DAY, @FechaInicial, @FechaFinal);
       -- Generar una fecha aleatoria
       DECLARE @FechaAzar DATETIME = DATEADD(DAY, ABS(CHECKSUM(NEWID())) % @Dias,
       @FechaInicial);
       -- Generar una hora aleatoria entre las 7 AM y las 8 PM
       DECLARE @HoraAzar INT = 7 + ABS(CHECKSUM(NEWID())) % 13;
       -- Combinar la fecha y la hora aleatorias
       SET @FechaHora = DATEADD(HOUR, @HoraAzar, @FechaAzar);
       INSERT INTO AlumnosInasist (idAlumno, FechaHora, Motivo)
       VALUES (@idAlumno, @FechaHora, @Motivo);
       SET @i = @i + 1;
    END;
END;
```

Se genera un identificador único diferente en cada llamada

2.2.2 Vistas

- Apoyándose en las tuplas recién añadidas con el *Script* de la diapositiva anterior, escriba una expresión de consulta para obtener una lista ordenada (por Apellidos y Nombre).
- La lista debe contener número de control, Nombre Completo de todos los alumnos y fecha de inasistencia, solo de aquellos alumnos con inasistencias para cierto periodo de cierto año.

2.2.2 Vistas

Obtener una lista con el número de control y el nombre de los alumnos que no tienen ninguna falta.

```
Select
    NumControl, NombreCompleto
from
    vAlumnos a
left join
    AlumnosInasist i on i.idAlumno=a.idAlumno
where Motivo is null
```

LEFT JOIN se comporta de una interesante manera: básicamente es un producto cartesiano (como INNER JOIN) pero que adicionalmente incluye las tuplas de la tabla de la IZQUIERDA, aunque no tengan correspondencia en la tabla de la derecha de acuerdo con la condición establecida en ON. Los atributos de la tabla de la derecha cuando no hay correspondencia, aparecen con valores NULL.

2.2.2 Vistas

Para entender la expresión anterior, pruebe las siguientes consultas:

```
select *
from
   vALumnos a
left join
   AlumnosInasist i on i.idAlumno=a.idAlumno
where Motivo is null


Select *
from
   vALumnos a
left join
   AlumnosInasist i on i.idAlumno=a.idAlumno
where Motivo is not null
```

Tabla **AlumnosRelaciones**

IdAlumnoRelacion	IdAlumno	IdPersona	Relacion
1	1	11	Padre
2	1	11	Tutor Legal
3	2	13	Tutor Legal
4	5	12	Padre
5	2	14	Esposa
6	1	11	Emergencia

Añada a estas tablas al menos las tuplas que se muestran para probar la vista y consultas de las diapositivas siguientes (datos de contacto de las personas *Relacionadas con los Alumnos*)

Tabla **FormasContacto**

IdFormaContacto	IdPersona	TipoContacto	Valor
1	1	Teléfono Fijo	819-27-37
2	1	Email	parejita@gmail.com
3	3	Email	benitojuarez@hotmail.com
4	4	Teléfono Fijo	818-04-11
5	5	Teléfono Celular	618 8189875
6	6	Teléfono Fijo	803-17-13
7	7	Teléfono Fijo	800-06-06
8	8	Teléfono Fijo	801-00-00
9	9	Teléfono Fijo	874-65-02
10	10	Teléfono Fijo	830-77-55
11	11	Teléfono Fijo	829-17-12

2.2.2 Vistas

```
insert into AlumnosRelaciones (idAlumno,idPersona,Relacion)
Values  (1,11,'Padre’),
        (1,11,'Tutor Legal’),
        (2,13,'Tutor Legal’),
        (5,12,'Padre’),
        (2,14,'Esposa’),
        (1,11,'Emergencia')

insert into FormasContacto (idPersona, Tipo, Valor)
Values  (1,'Teléfono Fijo','819-27-37’),
        (1,'Email','parejita@gmail.com’),
        (3,'Email','benitojuarez@hotmail.com’),
        (4,'Teléfono Fijo','818-04-11’),
        (5,'Teléfono Celular','618 8189875’),
        (6,'Teléfono Fijo','803-17-13’),
        (7,'Teléfono Fijo','800-06-06’),
        (8,'Teléfono Fijo','801-00-00’),
        (9,'Teléfono Fijo','874-65-02’),
        (10,'Teléfono Fijo','830-77-55’),
        (11,'Teléfono Fijo','829-17-12’),
        (11,'Email','pareja@gmail.com’)
```

2.2.2 Vistas

Crear una Vista llamada **vPersonasContacto:**

```
create view vPersonasContacto as
select p.idPersona, CURP, Nombre, Apellidos,
   concat_ws(' ',Nombre,Apellidos) NombreCompleto,
   Calle, NumExt, Poblacion, Pais,
   concat_ws(' ',Calle,numext,Poblacion,Pais)  Domicilio,
   FechaNac, fc.Tipo, fc.Valor Contacto
from
   Personas p
left join
   FormasContacto fc
on fc.idPersona=p.idPersona
```

Se usa **LEFT JOIN** para que en la consulta aparezcan las tuplas de la tabla ***Personas*** aunque no tenga formas de contacto en la tabla FormasContacto.

2.2.2 Vistas

Consulta: obtener una lista con los nombres de los alumnos y sus padres/madres, así como las formas de contacto de estos últimos.

```
select
    NumControl,
    a.NombreCompleto NombreAlumno,
    pc.NombreCompleto NombrePersonaRelacionada,
    Relacion,
    pc.Domicilio DomicilioPersonaRelacionada,
    pc.Tipo ContactoPersonaRelacionada,
    pc.Contacto
from
    vAlumnos a
inner join
    AlumnosRelaciones ar on a.idAlumno=ar.idAlumno
inner join
    vPersonasContacto pc on pc.idPersona=ar.idPersona
where Relacion='Padre' or Relacion='Madre'
```

2.3 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:
  - **Avg(atributo)**. Calcula la media de los valores de un atributo numérico **para un grupo de tuplas**.
  - **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.

2.3 Funciones de Agrupación

- Funciones:
  - **Sum(atributo)**. Calcula la suma de todos 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 de **"atributo"** puede ir la palabra **"distinct"**.
- Se debe indicar el criterio de agrupación usando la cláusula **group by**.
- En la tabla resultante aparecerá **una tupla por cada grupo formado**.

2.3 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

Al no llevar **group by**, se formará solo un grupo (con todas las tuplas).

2.3 Funciones de Agrupación

tabla **InasistAlum**

IdInasistAlum	IdAlumno	FechaHora	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

```
select Motivo,count(*) TotalFaltas
    from InasistAlum
    group by Motivo
    order by TotalFaltas desc
```

Obtener una tabla con los motivos y el total de inasistencias según cada uno de los diferentes motivos.
La tabla debe ordenarse en forma descendente por el total.

2.3 Funciones de Agrupación

Igual a la anterior, pero en la tabla resultante solo se deben incluir los motivos **mayores a 200** inasistencias.

```
select Motivo,count(*) TotalFaltas
    from InasistAlum
    group by motivo
    having TotalFaltas>200
    order by TotalFaltas desc
```

2.3 Funciones de Agrupación

Número de inasistencias en total de cada alumno.

```
select
   idAlumno,count(*) as TotalFaltas
from
   InasistAlum
group by
   idAlumno
order by
   TotalFaltas desc
```

2.3 Funciones de Agrupación

Número de inasistencias de cada alumno por mes, ordenadas de más a menos considerando solo faltas del año 2020.

```
set dateFormat DMY
select
    idAlumno,
    month(FechaHora) as Mes,
    count(*) as TotalFaltas
from
    AlumnosInasist
Where
    convert(date,FechaHora) between
    '01/01/2020' and '31/12/2020'
group by idAlumno,month(FechaHora)
order by TotalFaltas desc
```

2.3 Funciones de Agrupación

Modifique la consulta siguiente (que probamos en una de las diapositivas anteriores) para que se obtenga de una vista llamada *vAlumnosInasist* (entre *vALumnos* y *AlumnosInasist*) y se muestre el número de control, nombre completo y total de faltas.

```
select
   idAlumno,count(*) as TotalFaltas
from
   InasistAlum
group by
   idAlumno
order by
   TotalFaltas desc
```

Consulta para obtener la *id* del alumno, el número de fechas distintas en las que tuvo faltas y el número de faltas en total.

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

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

```
select
    count(distinct convert(
          DATE,FechaHora))
    as TotDias,
    count(*) as TotFaltas,
    IdAlumno
from
    AlumnosInasist
where
    YEAR(FechaHora)=2001
group by
    IdAlumno
order by
    TotDias desc,TotFaltas desc
```

2.3 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.

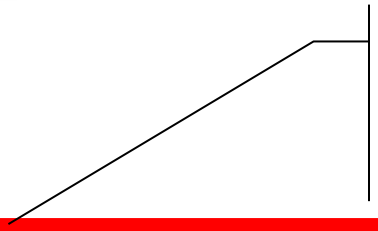
select avg(creditos) from Materias

o

select sum(creditos)/count(*) from Materias

2.3 Funciones de Agrupación

Número de inasistencias de cada alumno por mes (todos los años), el resultado debe mostrarse como una tabla con los meses como columnas.



Pruebe esta consulta en primer lugar para comprender mejor la expresión completa

```
SELECT *
FROM (
    SELECT idAlumno, MONTH(FechaHora) AS mes
    FROM InasistAlum
    ) AS TablaOrigen
PIVOT (
    count(mes) FOR mes IN
    ([1],[2],[3],[4],[5],[6],[7],[8],[9],[10],[11],[12])
    ) AS TablaResultante
```

2.4 Subconsultas

El uso de subconsultas es un recurso válido, pero tiende a complicar el código, como se puede ver en la expresión siguiente que obtiene domicilio y los datos de contacto de los estudiantes.

```
select NumControl,Nombre,Calle,NumExt,TipoContacto,Valor
from (select
   NumControl,Nombre,Calle,NumExt,Personas.idPersona
   from Alumnos,Personas where
   Alumnos.idPersona=Personas.idPersona) A2
   left join FormasContacto
on A2.idPersona=FormasContacto.idPersona
```

Es obligatorio renombrar el resultado de la subconsulta (A2)

El código en rojo equivale a la vista *vAlumnos* con la que se resuelve esta consulta de manera mas simple.

2.5 Operadores de Conjuntos
Operador IN (Pertenencia)

Obtener los números de control de alumnos que no son maestros.

```
select Control from Alumnos
    where idPersona not in
        (select idPersona from Maestros)
```

Modifique la expresión para obtener el Número de Control y el Nombre.

Modifique la consulta para que resulten los números de control de los alumnos que son maestros.

2.5 Operadores de Conjuntos
Operador IN (Pertenencia)

Pruebe las siguientes consultas:

```
select * from Materias
   where creditos in
   (select max(creditos) from Materias)

select * from Materias
   where creditos in
   (select min(creditos) from Materias)
```

Ejercicio:
Usando el operador de pertenencia obtenga una lista de los números de control y nombres de los alumnos que no tienen ninguna inasistencia.

2.6 INSERT, UPDATE, DELETE con una expresión de consulta

Para borrar el historial de inasistencias de aquellos alumnos que tengan como máximo tres faltas en la tabla InasistAlum, considerando solo las faltas del año 2001.

InasistAlum

IdInAlum	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

```
delete from
  AlumnosInasist
where
  IdAlumno in
     (select IdAlumno from AlumnosInasist
     where year(FechaHora)=2001
     group by idAlumno having count(*)<=3)
  and YEAR(FechaHora)=2001
```

Pruebe esta subconsulta por separado.

Modifique la expresión para que se elimine el historial, siempre y cuando las faltas no sean injustificadas.

2.6 INSERT, UPDATE, DELETE con una expresión de consulta

InasistAlum

IdInAlum	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

Tabla CALIFICACIONES

IdCalificacion	IdAlumno	IdMateria	Nota	Etapa
1	1	1	70	Especial
2	1	2	71	Regul1
3	1	3	70	Regul2
4	2	3	90	Normal
5	4	1	95	Normal
6	4	4	100	Normal

Añada a la Base de Datos la tabla Calificaciones con el esquema que se muestra y verifique la expresión de abajo que disminuye un punto en todas las calificaciones de aquellos alumnos con 2 o más inasistencias injustificadas.

```
update
    Calificaciones
set
    Nota = Nota - 1
where
    IdAlumno in
    (select IdAlumno
    from AlumnosInasist
    where Motivo='Injustificada'
    and year(FechaHora)=2001
    group by IdAlumno
    having count(*)>=2)
```

Pruebe también esta subconsulta por separado.

2.6 INSERT, UPDATE, DELETE con una expresión de consulta

La siguiente vista se basa en la consulta de la página 34. Sirve para obtener los correos electrónicos de los padres y madres de los alumnos.

```
create view vAlumnosCorreoPadres as
select
    a.idAlumno,
    NumControl,
    a.NombreCompleto NombreAlumno,
    pc.NombreCompleto NombrePadreMadre,
    pc.Contacto
from
    vAlumnos a
inner join
    AlumnosRelaciones ar on a.idAlumno=ar.idAlumno
inner join
    vPersonasContacto pc on pc.idPersona=ar.idPersona
Where
    (Relacion='Padre' or Relacion='Madre') and pc.Tipo='Email'
```

2.6 INSERT, UPDATE, DELETE con una expresión de consulta

insert puede combinarse con **select** para añadir varias tuplas a la vez.

Ejemplo: A partir de la vista **vAlumnosCorreoPadres** y la tabla **AlumnosInasist**, añadir a la tabla **MensajesCorreo** (hay que crear esta tabla) las tuplas correspondientes a los mensajes de correo que se deben enviar al padre o madre del estudiante para avisarle de las inasistencias.

vAlumnosCorreoPadres

idAlumno	NumControl	NombreAlumno	NombrePadreMadre	Email
3	13030002	Mickey Mouse	Michael Mouse	MichaelMouse@gmail.com
4	13040154	Pata Daisy	Mamá Pato	MamaDaisy@hotmail.com

InasistAlum

IdInasistAlum	idAlumno	FechaHora	Motivo
1	3	2016-09-10 11:00:00.000	Injustificada
2	4	2016-09-11 10:00:00.000	Evento Deportivo
3	4	2016-09-11 11:00:00.000	Evento Deportivo

MensajesCorreo

idMnsCorreo	Destinatario	Mensaje	FechaEnvio
1	MichaelMouse @ gmail.com	su hijo Mickey Mouse tiene una inasistencia el Sep 10 2016 11:00AM motivo Injustificada	NULL
2	MamaDaisy @ hotmail.com	su hijo Pata Daisy tiene una inasistencia el Sep 11 2016 10:00AM motivo Evento Deportivo	NULL
3	MamaDaisy @ hotmail.com	su hijo Pata Daisy tiene una inasistencia el Sep 11 2016 11:00AM motivo Evento Deportivo	NULL

2.6 INSERT, UPDATE, DELETE con una expresión de consulta

```
insert into
    MensajesCorreo
Select
    Email,
    concat_ws(' ','su hijo',NombreAlumno,'tuvo una inasistencia el',
              cast(FechaHora as varchar),'motivo',Motivo)
from
    vAlumnosCorreoPadres a
inner join
    InasistAlum i on i.idAlumno=a.idAlumno
```

vAlumnosCorreoPadres

idAlumno	NumControl	NombreAlumno	NombrePadreMadre	Email
3	13030002	Mickey Mouse	Michael Mouse	MichaelMouse@gmail.com
4	13040154	Pata Daisy	Mamá Pato	MamaDaisy@hotmail.com

InasistAlum

IdInasistAlum	idAlumno	FechaHora	Motivo
1	3	2016-09-10 11:00:00.000	Injustificada
2	4	2016-09-11 10:00:00.000	Evento Deportivo
3	4	2016-09-11 11:00:00.000	Evento Deportivo

MensajesCorreo

idMnsCorreo	Destinatario	Mensaje	FechaEnvio
1	MichaelMouse @ gmail.com	su hijo Mickey Mouse tiene una inasistencia el Sep 10 2016 11:00AM motivo Injustificada	NULL
2	MamaDaisy @ hotmail.com	su hijo Pata Daisy tiene una inasistencia el Sep 11 2016 10:00AM motivo Evento Deportivo	NULL
3	MamaDaisy @ hotmail.com	su hijo Pata Daisy tiene una inasistencia el Sep 11 2016 11:00AM motivo Evento Deportivo	NULL