

Sentencias SQL

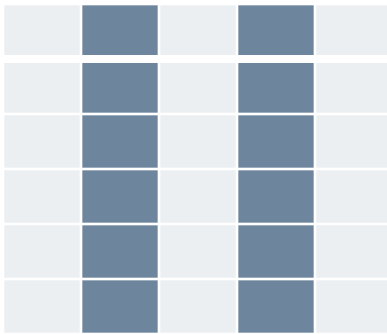
Laboratorio Base de Datos I

Objetivos de la Clase

- ☐ Sentencias SQL Select.
- ☐ Clausula WHERE.
- ☐ Clausula ORDER BY
- ☐ Clausula GROUP BY y HAVING

Lenguaje de consulta SQL

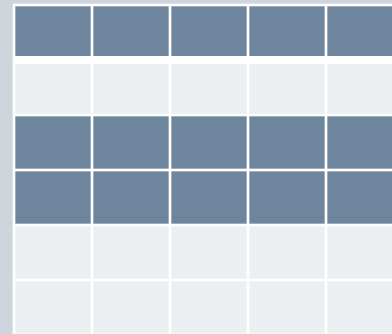
Proyección



A 6x5 grid representing 'Tabla 1'. The second and fourth columns are highlighted in dark blue, while the first, third, and fifth columns are light gray.

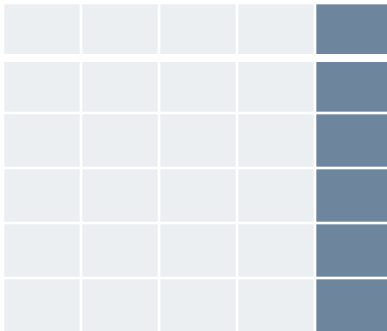
Tabla 1

Selección



A 6x5 grid representing 'Tabla 1' enclosed in a light blue box. The top two rows and the middle two rows (rows 3 and 4) are highlighted in dark blue. The first and fifth columns are light gray.

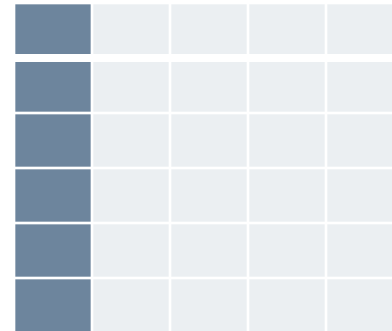
Tabla 1



A 6x5 grid representing 'Tabla 1'. The fifth column is highlighted in dark blue, while the first four columns are light gray.

Tabla 1

Combinación



A 6x5 grid representing 'Tabla 2'. The first column is highlighted in dark blue, while the second through fifth columns are light gray.

Tabla 2

Lenguaje SQL

□ SELECT

SELECT

[TOP expresión [PERCENT] [WITH TIES]]

<lista seleccionada> [INTO nueva_tabla]

[FROM tabla]

[WHERE condición]

[GROUP BY expresión]

[HAVING condición]

[ORDER BY expresión [ASC | DESC]]

□ Join

- CROSS

- INNER

- OUTER (LEFT, RIGHT, FULL)

Clausula WHERE

La cláusula **WHERE** puede usarse para determinar qué registros de las tablas enumeradas en la cláusula FROM aparecerán en los resultados de la instrucción **SELECT**

□ [**WHERE** <condición>]

SELECT

[TOP expresión [PERCENT] [WITH TIES]]

<lista seleccionada> [INTO nueva_tabla]

[FROM tabla]

[**WHERE** condición]

[GROUP BY expresión]

[HAVING condición]

[ORDER BY expresión [ASC | DESC]]

Clausula WHERE

Ejemplos:

```
USE AdventureWorks ;  
GO  
SELECT * FROM Person.Address  
WHERE City = 'San Francisco';
```

```
USE AdventureWorks ;  
GO  
SELECT * FROM Sales.SalesOrderHeader  
WHERE TotalDue > 1000;
```

Clausula WHERE

<condición>: Define la condición que se debe cumplir para que se devuelvan las filas.

□ <condición>:

```
{ [ NOT ] <predicate> | ( <search_condition> ) }  
[ { AND | OR } [ NOT ] { <predicate> | ( <search_condition> ) } ]  
[ ,...n ]
```

```
<predicate> ::=  
{ expression { = | < | > | != | > | > = | ! > | < | < = | ! < } expression  
| string_expression [ NOT ] LIKE string_expression  
| expression [ NOT ] BETWEEN expression AND expression  
| expression IS [ NOT ] NULL
```

Clausula WHERE

Ejemplos:

```
USE AdventureWorks ;  
GO  
SELECT * FROM Person.Address  
WHERE City = 'San Francisco' OR City = 'Orleans';
```

```
USE AdventureWorks ;  
GO  
SELECT * FROM Sales.SalesOrderHeader  
WHERE TotalDue > 1000 AND TaxAmt <500;
```


Clausula WHERE

<condición>: Define la condición que se debe cumplir para que se devuelvan las filas.

□ OPERADORES:

- Operadores comparación:

{ = | < | > | != | > | > = | ! > | < | < = | ! < |

ALL | ANY | **BETWEEN** | EXISTS | **IN** | **LIKE** | SOME }

- Operadores lógicos:

NOT | OR | AND

- **IS [NOT] NULL**

Clausula WHERE

□ OPERADORES:

Operadores de comparación:

Operador	Uso
<	Menor que
>	Mayor que
<>	Distinto de
<=	Menor ó Igual que
>=	Mayor ó Igual que
=	Igual que
BETWEEN	Utilizado para especificar un intervalo de valores.
LIKE	Utilizado en la comparación de un modelo
IN	Utilizado para especificar registros de una base de datos

Clausula WHERE

☐ OPERADORES:

Operadores lógicos:

Operador	Uso
AND	Es el "y" lógico. Evalúa dos condiciones y devuelve un valor de verdad sólo si ambas son ciertas.
OR	Es el "o" lógico. Evalúa dos condiciones y devuelve un valor de verdad si alguna de las dos es cierta.
NOT	Negación lógica. Devuelve el valor contrario de la expresión.

Clausula WHERE

□ OPERADORES:

Operadores lógicos:

<expresión1>	Operador	<expresión2>	Resultado
Verdad	AND	Falso	Falso
Verdad	AND	Verdad	Verdad
Falso	AND	Verdad	Falso
Falso	AND	Falso	Falso
Verdad	OR	Falso	Verdad
Verdad	OR	Verdad	Verdad
Falso	OR	Verdad	Verdad
Falso	OR	Falso	Falso

Clausula WHERE

☐ Operador BETWEEN

Para indicar que deseamos recuperar los registros según el intervalo de valores de un campo

*<expression [NOT] **BETWEEN** valor1 **AND** valor2>*

Ejemplo:

```
USE AdventureWorks
GO
SELECT ListPrice
FROM Production.ProductListPriceHistory
WHERE ListPrice BETWEEN 9 AND 15
```

Clausula WHERE

☐ Operador BETWEEN

Ejemplo:

```
USE AdventureWorks  
GO  
SELECT ModifiedDate  
FROM Production.ProductListPriceHistory  
WHERE ModifiedDate  
BETWEEN '2002-06-30' AND '2003-06-10'
```

Clausula WHERE

☐ Operador LIKE

Se utiliza para comparar una expresión de cadena con un modelo en una expresión SQL

*<expresión **LIKE** modelo>*

En donde *modelo* es una cadena modelo o campo contra el que se compara *expresión*

Ejemplo:

```
USE AdventureWorks  
GO  
SELECT ProductModelID, Name  
FROM Production.ProductModel  
WHERE Name LIKE 'Mountain%'
```

Clausula WHERE

□ Operador LIKE

La utilización de caracteres comodín hace que el operador LIKE sea más flexible <expresión **LIKE** *__modelo%*>

Carácter comodín	Descripción	Ejemplo
%	Cualquier cadena de cero o más caracteres.	WHERE title LIKE '%computer%' busca todos los títulos de libros que contengan la palabra 'computer' en el título.
_ (carácter de subrayado)	Cualquier carácter.	WHERE au_fname LIKE '_ean' busca todos los nombres de cuatro letras que terminen en ean (Dean, Sean, etc.)

Clausula WHERE

☐ Operador LIKE

Ejemplos:

```
USE AdventureWorks  
GO  
SELECT ProductModelID, Name  
FROM Production.ProductModel  
WHERE Name LIKE '%Seat/Saddle%'
```

```
USE AdventureWorks  
GO  
SELECT ProductModelID, Name  
FROM Production.ProductModel  
WHERE Name LIKE '____Road%'
```

Clausula WHERE

☐ Operador IN

Devuelve aquellos registros cuyo campo indicado coincide con algún valor de una lista.

*<expresión [NOT] **IN** lista>*

Ejemplo:

```
USE AdventureWorks
GO
SELECT CountryRegionCode, Name
FROM Person.StateProvince
WHERE CountryRegionCode IN ('US','CA','DE')
```

Clausula ORDER BY

Especifica el orden utilizado en las columnas devueltas en una instrucción **SELECT**

- **[ORDER BY {expresión [ASC | DESC]}**
SELECT
[TOP expresión [PERCENT] [WITH TIES]]
<lista seleccionada> [INTO nueva_tabla]
[FROM tabla]
[WHERE condición]
[GROUP BY expresión]
[HAVING condición]
[**ORDER BY** expresión [ASC | DESC]]

Clausula ORDER BY

Ejemplos:

```
USE AdventureWorks  
GO  
SELECT ProductModelID, Name  
FROM Production.ProductModel  
WHERE Name LIKE '____Road%'  
ORDER BY ProductModelID DESC
```

```
USE AdventureWorks  
GO  
SELECT ProductID, ListPrice AS precio  
FROM Production.ProductListPriceHistory  
WHERE ListPrice BETWEEN 9 AND 15  
ORDER BY StartDate DESC, precio ASC
```

Funciones de Fila

Las funciones a nivel de filas operan sobre ellas y devuelven un resultado por cada una.

Existen diferentes tipos. Entre ellas:

Categoría de la función	Descripción
<u>Funciones de fecha y hora</u>	Llevan a cabo operaciones sobre un valor de entrada de fecha y hora, y devuelven un valor numérico, de cadena o de fecha y hora.
<u>Funciones matemáticas</u>	Realizan cálculos basados en valores de entrada proporcionados como parámetros a las funciones y devuelven valores numéricos.
<u>Funciones de cadena</u>	Realizan operaciones en el valor de entrada de una cadena (char o varchar) y devuelven una cadena o un valor numérico.

Funciones de Fila

Funciones de Fecha y Hora

GETDATE: Devuelve la fecha y hora actuales del sistema en el formato interno estándar de SQL Server 2005 para los valores **datetime**.

<GETDATE ()>

Ejemplo:

```
SELECT GETDATE ();
```

Funciones de Fila

Funciones de Fecha y Hora

DATEDIFF: Devuelve el número de límites de fecha y hora entre dos fechas especificadas.

<**DATEDIFF** (datepart , startdate , enddate)>

Ejemplos:

```
USE AdventureWorks
```

```
GO
```

```
SELECT BirthDate as FechaNac, DATEDIFF(year,  
BirthDate, getdate()) as Edad
```

```
FROM HumanResources.Employee
```

```
WHERE DATEDIFF (year, BirthDate, getdate()) > 65
```

Funciones de Fila

Funciones matemáticas

ROUND: Devuelve un valor numérico, redondeado a la longitud o precisión especificadas.

<**ROUND** (numeric_expression , length [,function])>

Ejemplo:

```
SELECT ROUND(123.9994, 3), ROUND(123.9995, 3);  
GO  
SELECT ROUND(123.4545, 2);  
GO  
SELECT ROUND(114.45, -2);  
GO  
SELECT ROUND(150.75, 0), ROUND(150.75, 0, 1);
```


Funciones de Fila

Funciones de cadena

LEN: Devuelve el número de caracteres de la expresión de cadena especificada, excluidos los espacios en blanco finales.

<**LEN** (string_expression)>

Ejemplo:

```
USE AdventureWorks;  
GO  
SELECT LEN(FirstName) AS Length, FirstName,  
LastName  
FROM Sales.vIndividualCustomer  
WHERE CountryRegionName = 'Australia';
```

Funciones de Fila

Función CASE

CASE: Evalúa una lista de condiciones y devuelve una de las varias expresiones de resultado posibles.

La expresión CASE tiene dos formatos:

```
CASE input_expression  
    WHEN when_expression THEN result_expression [ ...n ]  
    [ ELSE else_result_expression ]  
END
```

Searched CASE expression:

```
CASE  
    WHEN Boolean_expression THEN result_expression [ ...n ]  
    [ ELSE else_result_expression ]  
END
```

Funciones de Fila

Función CASE

Ejemplo:

```
USE AdventureWorks;  
GO  
SELECT ProductNumber, Category =  
    CASE ProductLine  
        WHEN 'R' THEN 'Road'  
        WHEN 'M' THEN 'Mountain'  
        WHEN 'T' THEN 'Touring'  
        WHEN 'S' THEN 'Other sale items'  
        ELSE 'Not for sale'  
    END,  
    Name  
FROM Production.Product  
ORDER BY ProductNumber;
```

Funciones de Fila

Función CASE

Ejemplo:

```
USE AdventureWorks;  
GO  
SELECT ProductNumber, Name, 'Price Range' =  
CASE  
  WHEN ListPrice = 0 THEN 'Mfg item - not for resale'  
  WHEN ListPrice < 50 THEN 'Under $50'  
  WHEN ListPrice >= 50 and ListPrice < 250 THEN 'Under $250'  
  WHEN ListPrice >= 250 and ListPrice < 1000 THEN 'Under $1000'  
  ELSE 'Over $1000'  
END  
FROM Production.Product  
ORDER BY ProductNumber ;
```

Funciones de agregado (GRUPO)

Las funciones de agregado realizan un cálculo sobre un conjunto de valores y devuelven un solo valor.

Algunas funciones:

AVG([DISTINCT] columna)

MIN([DISTINCT] columna)

SUM([DISTINCT] columna)

COUNT(*|[DISTINCT] columna)

MAX([DISTINCT] columna)

Funciones de agregado (GRUPO)

Ejemplo:

```
USE AdventureWorks;  
GO  
SELECT AVG(VacationHours) as 'Average vacation hours', SUM  
(SickLeaveHours) as 'Total sick leave hours'  
FROM HumanResources.Employee  
WHERE Title LIKE 'Vice President%';
```

```
USE AdventureWorks;  
GO  
SELECT AVG(DISTINCT ListPrice)  
FROM Production.Product;
```

Clausula GROUP BY

Combina los registros con valores idénticos en la lista de campos especificados, en un único registro. Para cada nuevo registro se crea un valor resumen si se incluye una función agregada (o función de grupo) en la lista de la SELECT.

□ [**GROUP BY** expresión]

[**HAVING** condición]

```
SELECT  
[TOP expresión [PERCENT] [ WITH TIES ] ]  
<lista seleccionada> [ INTO nueva_tabla ]  
[ FROM tabla ]  
[ WHERE condición ]  
[ GROUP BY expresión ]  
[ HAVING condición ]  
[ ORDER BY expresión [ ASC | DESC ] ]
```

Clausula GROUP BY

Ejemplo:

```
USE AdventureWorks ; GO
SELECT SalesOrderID, SUM(LineTotal) AS SubTotal
FROM Sales.SalesOrderDetail sod
GROUP BY SalesOrderID
ORDER BY SalesOrderID ;
```

```
USE AdventureWorks;
GO
SELECT AVG(DISTINCT ListPrice)
FROM Production.Product;
```


Clausula HAVING

Especifica una condición de búsqueda para un grupo o agregado

- [**GROUP BY** expresión]
[**HAVING** condición]

<condición>

Especifica la condición de búsqueda del grupo o del agregado que se debe cumplir.

Clausula HAVING

Ejemplos:

```
USE AdventureWorks ;  
GO  
SELECT SalesOrderID, SUM(LineTotal) AS SubTotal  
FROM Sales.SalesOrderDetail  
GROUP BY SalesOrderID  
HAVING SUM(LineTotal) > 100000.00  
ORDER BY SubTotal DESC;
```

```
SELECT DATEPART(yyyy,OrderDate) AS Year,  
SUM(TotalDue) AS AverageOrderAmt  
FROM Sales.SalesOrderHeader  
GROUP BY DATEPART(yyyy,OrderDate)  
HAVING DATEPART(yyyy,OrderDate) NOT BETWEEN 2001 AND  
2003  
ORDER BY DATEPART(yyyy,OrderDate);
```

Fin Tema

☐ ¿Preguntas?