

SQL

Servando Campillay

INTRODUCCIÓN

- SQL (Structured Query Language)
- SQL emplea los términos tabla, fila y columna en vez de relación, tupla y atributo, respectivamente.
- Las instrucciones SQL2 para definir datos son CREATE, ALTER y DROP



CONCEPTO DE ESQUEMA Y CATÁLOGO

- Un esquema SQL se identifica con un **nombre de esquema** y consta de un identificador de autorización, que indica el usuario o la cuenta propietaria del esquema, además de los **descriptores** de cada elemento del esquema.
- Los elementos del esquema comprenden tablas, restricciones, vistas, dominios y otros.



CONCEPTO DE ESQUEMA Y CATÁLOGO

- Un esquema se crea mediante la sentencia `CREATE SCHEMA`, por ejemplo:
- **`CREATE SCHEMA EMPRESA AUTHORIZATION JPEREZ`**; Catálogo es un conjunto de esquemas, con un nombre. El catálogo, siempre contiene un esquema especial, llamado `INFORMATION_SCHEMA`, que proporciona información sobre todos los esquemas del catálogo.
- La ventaja de los catálogos es que se pueden definir restricciones de integridad, sobre relaciones que están en el mismo catálogo. Además los esquemas del mismo catálogo pueden compartir ciertos elementos, como definiciones de dominio.



CREATE TABLE Y TIPOS DE DATOS

- La instrucción CREATE TABLE permite crear relaciones, dándole un nombre y especificando atributos y restricciones.
- Los atributos se especifican con un nombre, un tipo de datos (para especificar su dominio de valores) y cualquier restricción del mismo, por ejemplo: NOT NULL.
- Se especifican las restricciones de clave: de integridad de la entidad y referencial



CREATE TABLE Y TIPOS DE DATOS

```
CREATE TABLE EMPLEADO
(NOMBRE      VARCHAR(15)  NOT NULL,
INIC        CHAR,
APELLIDO    VARCHAR(15)  NOT NULL,
NUMERO      CHAR(9)      NOT NULL,
FECHA_NAC   DATE,
DIRECCION   VARCHAR(30),
SEXO        CHAR,
SALARIO     DECIMAL(10,2),
NRO_SUP     CHAR(9),
NRO_DEPTO   INT          NOT NULL,
PRIMARY KEY (NUMERO),
FOREIGN KEY (NRO_SUP) REFERENCES EMPLEADO (NUMERO),
FOREIGN KEY (NRO_DEPTO) REFERENCES DEPARTAMENTO (ND));
```



TIPOS DE DATOS Y DOMINIOS

- Tipos de Datos para atributos: numéricos, cadena de caracteres, cadena de bits, fecha y hora.
- Numéricos (de distintos tamaños, enteros):
INTEGER o INT y SMALLINT y reales de diversas precisiones (FLOAT; REAL; DOUBLE PRECISION). Formatos DECIMAL(i,j) o DEC(i,j) o NUMERIC(i,j), donde i es la precisión (nro de digitos) y j la escala (numero de decimales)



TIPOS DE DATOS Y DOMINIOS

- Cadena de caracteres, de longitud fija (CHAR(n) o CHARACTER(n), donde n es el número de caracteres) o de longitud variable (VARCHAR(n) o CHARVARYING(n), donde es el número máximo de caracteres)
- Cadena de bits, pueden ser de longitud fija n (BIT(n)) o longitud variable (BIT VARIYNG(n)), donde n es el número máximo de bits.



TIPOS DE DATOS Y DOMINIOS

- Tipo de dato para fecha es DATE, tiene 10 posiciones y sus componentes son YEAR, MONTH y DAY, por lo regular de la forma YYYY-MM-DD.
- Tipo de dato para tiempo es TIME, tiene 8 posiciones, con los componentes HOUR, MINUTE y SECOND, por lo general de la forma HH:MM:SS



TIPOS DE DATOS Y DOMINIOS

- Dominios: esto permite cambiar tipos de datos de un dominio utilizado por un gran numero de atributos más fácilmente. Por ejemplo, dominio TIPO_NUMERO así:
- **CREATE DOMAIN TIPO_NUMERO AS CHAR(9),**



ESPECIFICACIÓN DE RESTRICCIONES

- Dado que SQL permite NULL como valor para un atributo, se puede especificar la restricción NOT NULL, si es que no queremos nulos.
- Valor por omisión de un atributo, añadiendo la cláusula DEFAULT <valor> a la definición de un atributo
- La cláusula PRIMARY KEY y FOREIGN KEY permiten especificar las pk y fk.
- La cláusula UNIQUE especifica claves alternativas



ESPECIFICACIÓN DE RESTRICCIONES

- El diseñador debe especificar las acciones que deben realizarse cuando se transgrede una restricción, asociada a la inserción, modificación y eliminación de un valor de atributo de clave foránea y asociada al modificarse un valor de clave primaria referenciada, añadiendo una cláusula de **acción de disparo referencial** a una restricción de clave externa.
- Las opciones son: SET NULL, CASCADE y SET DEFAULT, las opciones son ON DELETE o ON CASCADE.



ESPECIFICACIÓN DE RESTRICCIONES

- CREATE TABLE EMPLEADO
 (...,
 NRO_DEPTO INT **NOT NULL** DEFAULT 1,
 PRIMARY KEY (NUMERO),
 FOREIGN KEY (NRO_SUP) **REFERENCES** EMPLEADO
 (NUMERO)
 ON DELETE SET NULL ON UPDATE CASCADE,
 FOREIGN KEY (NRO_DEPTO) **REFERENCES** DEPARTAMENTO
 (ND) ON
 DELETE SET DEFAULT ON UPDATE CASCADE);
- Se tiene SET NULL ON DELETE y CASCADE ON UPDATE para la clave externa NRO_SUP de EMPLEADO. Esto significa que si se elimina la tupla de empleado supervisor, el valor de NRO_SUP pasa automáticamente a NULL en todas las tuplas de EMPLEADO que hagan referencia a la tupla eliminada.



ESPECIFICACIÓN DE RESTRICCIONES

- Si se actualiza el valor de NUMERO de un empleado supervisor, el nuevo valor se propaga a NRO_SUP para todas las tuplas de empleado que hagan referencia a la tupla del empleado actualizada.
- En general, la acción emprendida por el SGBD cuando se especifica SET NULL o SET DEFAULT es la misma para ON DELETE que para ON UPDATE: el valor de los atributos referenciados afectados se cambia a NULL en el caso de SET NULL y al valor por omisión especificado en el caso de SET DEFAULT.



ESPECIFICACIÓN DE RESTRICCIONES

- La acción correspondiente a CASCADE ON DELETE es eliminar todas las tuplas reverenciadoras, mientras que la acción correspondiente a CASCADE ON UPDATE es cambiar el valor de la clave externa al nuevo valor actualizado de la clave primaria en todas las tuplas reverenciadoras.
- Por lo general, la opción CASCADE es adecuada para tablas del tipo TRABAJA_EN, para relaciones que representan atributos multivaluados como LOCALIZACION_DEPTO y para relaciones que representan tipos de entidad débiles como DEPENDIENTE.



ESPECIFICACIÓN DE RESTRICCIONES

- Las relaciones declaradas con `CREATE TABLE` se denominan tablas base. El SGBD las almacena como archivos.
- En SQL, los atributos de una relación de una tabla base están ordenados en la secuencia en que se especifican en la sentencia `CREATE TABLE`. Las filas no están ordenadas.



DROP SCHEMA Y DROP TABLE

- Existen dos opciones para eliminar: CASCADE y RESTRICT.
- Si se desea eliminar el esquema EMPRESA con todas sus tablas, dominios, etc. Se utiliza CASCADE:
- **DROP SCHEMA EMPRESA CASCADE;**
- Si se elige la opción RESTRICT, el esquema se eliminará sólo si no tiene elementos; en caso contrario no se ejecutará la instrucción DROP



DROP SCHEMA Y DROP TABLE

- DROP TABLE DEPENDIENTE CASCADE;
- Elimina la tabla DEPENDIENTE. Con la opción CASCADE, todas las restricciones y vistas que hagan referencia a la tabla se eliminarán automáticamente del esquema, junto con la propia tabla.



ALTER TABLE

- Las posibles acciones de alterar tablas incluyen agregar atributos, eliminar atributos, la modificación de la definición de una columna y la agregación y eliminación de restricciones de tabla.
- **ALTER TABLE EMPRESA.EMPLEADO ADD PUESTO VARCHAR(12);** Agrega un atributo. Para darle valores a este nuevo atributo se puede utilizar UPDATE o bien darle un valor por defecto. Si no se hace esto último, el nuevo atributo tendrá valor NULL
- **ALTER TABLE EMPRESA.EMPLEADO DROP DIRECCION CASCADE;** Elimina atributo DIRECCION de la tabla EMPLEADO



CONSULTAS BASICAS

- La sentencia SELECT permite recuperar información de la BD. No tiene nada que ver con la operación de selección del álgebra relacional.
- SQL y el Modelo relacional tienen diferencias: SQL permite filas duplicadas, es decir, una tabla SQL no es un conjunto de tuplas, porque los conjuntos no permiten elementos repetidos. A veces a las tablas SQL se les llama bolsa, bag o multiconjunto.
- La cláusula DISTINCT elimina las filas repetidas.



SELECT-FROM-WHERE

```
SELECT <lista de atributos>  
FROM   <lista de tablas>  
WHERE  <condición>
```

```
SELECT FECHA_NAC, DIRECCION  
FROM EMPLEADO  
WHERE Nombre='Juan' AND APELLIDO = 'Pérez';
```



EJERCICIO

- Dada la siguiente BDR:
- EMPLEADO(Nombre, Apellido, NroEmpleado, FechaNacimiento, Dirección, Sexo, Salario, NroSuperior, NroDepto)
- DEPARTAMENTO(NombreDepto, NroDepto, NroGerente, FechaInicioGerente)
- LUGARESDEPTO(NroDepto, LugarDepto)
- PROYECTO(Nombreproyecto, NroProyecto, LugarProyecto, NroDepto)
- TRABAJAEN(NroEmpleado, NroProyecto, Horas)
- DEPENDIENTE(NroEmpleado, NombreDependiente, Sexo, FechaNacimiento, Parentesco)



SELECT-FROM-WHERE

- C2 Recupere el nombre, apellido y dirección de todos los empleados que trabajan en el departamento de Investigación.

```
SELECT NOMBRE, APELLIDO, DIRECCION  
FROM EMPLEADO, DEPARTAMENTO  
WHERE NOMBREDEPTO = 'Investigación'  
AND DEPARTAMENTO.NRODEPTO =  
EMPLEADO.NRODEPTO;
```



SELECT-FROM-WHERE

- C3 De cada proyecto ubicado en Concepción, haga una lista con el número de proyecto, el numero de departamento controlador y el apellido, dirección y fecha de nacimiento del jefe de departamento.

```
SELECT NROPROYECTO, NRODEPTO, APELLIDO,  
DIRECCION, FECHANACIMIENTO  
FROM EMPLEADO, DEPARTAMENTO, PROYECTO  
WHERE PROYECTO.NRODEPTO =  
DEPARTAMENTO.NRODEPTO AND  
DEPARTAMENTO.NROGERENTE =  
EMPLEADO.NROEMPLEADO AND  
LUGARPROYECTO = 'Concepción';
```



CONSULTAS BASICAS

- Cláusulas Where no especificadas (todas las tuplas son consideradas)
- C4 Seleccione todos los números de empleado de EMPLEADO
SELECT NROEMPLEADO FROM EMPLEADO;
- C5 Seleccione todas las combinaciones posibles de números de empleado y números de departamento
SELECT NROEMPLEADO,
NRODEPARTAMENTO
FROM EMPLEADO, DEPARTAMENTO;
- Esta consulta realiza el producto cartesiano.



CONSULTAS BASICAS

- Uso del *: Recupera los valores de todos los atributos.

```
SELECT *  
FROM EMPLEADO  
WHERE NroDepto = 5;
```



CONSULTAS BASICAS

- Tablas como conjuntos en SQL:
- SQL no elimina las filas repetidas en los resultados de las consultas, por las siguientes razones:
 - La eliminación de duplicados es una operación costosa. Una forma de implementarla es ordenar las tuplas primero y luego eliminar los duplicados.
 - Es posible que el usuario requiera ver las tuplas repetidas en el resultado de la consulta
 - Cuando se aplica una función agregada, en la mayoría de los casos no se quiere eliminar los duplicados.



CONSULTAS BASICAS

- Si queremos eliminar tuplas repetidas en el resultado de una consulta, se usa la palabra clave **DISTINCT** en la cláusula **SELECT**.
- Recupere el salario de todos los empleados y los valores de todos los salarios distintos

```
SELECT ALL SALARIO (o SELECT simple)
FROM EMPLEADO; (incluye repetidos)
SELECT DISTINCT SALARIO
FROM EMPLEADO; (no incluye repetidos)
```



CONSULTAS BASICAS

- Operaciones de Conjunto: UNION, EXCEPT e INTERSECT, equivalentes a la unión, diferencia e intersección de conjuntos. Las tuplas repetidas se eliminan del resultado y se debe asegurar que las relaciones sean compatibles en su esquema (mismos atributos y el orden).
- Lista con todos los números de proyecto en los que participa un empleado de apellido Pérez, sea como trabajador o como jefe del departamento que controla el proyecto.



CONSULTAS BASICAS

```
(SELECT DISTINCT NROPROYECTO
FROM PROYECTO, DEPARTAMENTO, EMPLEADO
WHERE PROYECTO.NRODEPTO =
      DEPARTAMENTO.NRODEPTO AND
      DEPARTAMENTO.NROGERENTE =
      EMPLEADO.NROEMPLEADO AND APELLIDO = 'Pérez')
UNION
(SELECT DISTINCT NROPROYECTO
FROM TRABAJA_EN, EMPLEADO
WHERE TRABAJA_EN. NROEMPLEADO =
      EMPLEADO.NROEMPLEADO AND APELLIDO = 'Pérez');
```



COMPARACIONES, ORDENACIÓN

- Comparación cadena de caracteres: operador LIKE, cadenas parciales %: sustituye un número arbitrario de caracteres y _ sustituye a un solo carácter.
- Recupere todos los empleados cuya dirección sea Talcahuano.

```
SELECT NOMBRE, APELLIDO  
FROM EMPLEADO  
WHERE DIRECCION LIKE '%Talcahuano';
```



COMPARACIONES, ORDENACIÓN

- Aritmética en Consultas: Los operadores suma(+), resta(-), multiplicación (*) y división (/) se pueden aplicar a valores numéricos o atributos con dominios numéricos.
- Muestre los salarios resultantes si cada empleado que trabaja en el proyecto 'Producto X' recibe un aumento del 10%.

```
SELECT NOMBRE, APELLIDO, 1.1*SALARIO  
FROM EMPLEADO, TRABAJA_EN, PROYECTO  
WHERE TRABAJA_EN.NROEMPLEADO =  
      EMPLEADO.NROEMPLEADO AND  
      TRABAJA_EN.NROPROYECTO = PROYECTO.NROPROYECTO  
AND NOMBREPROYECTO = 'Producto X';
```



COMPARACIONES, ORDENACIÓN

- En el caso de tipos de datos de cadena, se puede usar el operador concatenación(//) en una consulta para anexar un valor de cadena a otro.
- En el caso de tipos de datos fecha, tiempo, los operadores incluyen el incremento (+) o disminución (-), en un intervalo compatible con el tipo de una fecha, o tiempo.



COMPARACIONES, ORDENACIÓN

- Operador BETWEEN
- Recupere todos los empleados del departamento 5 cuyo salario esté entre 30000 y 40000 dólares

```
SELECT *  
FROM EMPLEADO  
WHERE (SALARIO BETWEEN 30000 AND  
40000) AND NRODEPTO= 5;
```



COMPARACIONES, ORDENACIÓN

- Cláusula ORDER BY
- Obtenga una lista de los empleados y de los proyectos en los que trabajan, ordenados por departamento, y dentro de cada departamento, alfabéticamente por apellido y nombre

```
SELECT NOMBREDEPTO, APELLIDO, NOMBRE,  
        NOMBREPROYECTO  
FROM DEPARTAMENTO, EMPLEADO, TRABAJA_EN,  
        PROYECTO  
WHERE DEPARTAMENTO.NRODEPTO =  
        EMPLEADO.NRODEPTO AND EMPLEADO.NROEMPLEADO  
        = TRABAJA_EN.NROEMPLEADO AND  
        TRABAJA_EN.NRO.PROY = PROYECTO.NROPROY  
ORDER BY NOMBREDEPTO, APELLIDO, NOMBRE;
```



COMPARACIONES, ORDENACIÓN

- El orden por omisión es ascendente. Se usa la palabra DESC para ordenar en forma descendente.
ORDER BY NOMBREDEPTO DESC, APELLIDO ASC, NOMBRE ASC



CONSULTAS COMPLEJAS

- Estas consultas requieren valores de la BD para usarlos después en una condición de comparación.
- Consultas anidadas: bloques select-from-where dentro de la cláusula where de otra consulta. A esta última consulta se le llama consulta externa



CONSULTAS COMPLEJAS

```
SELECT DISTINCT NROPROY  
FROM PROYECTO  
WHERE NROPROY IN ( SELECT NROPROY  
    FROM PROYECTO, EMPLEADO  
    WHERE PROYECTO.NRODEPTO=DEPTO.NRODEPTO  
    AND NRO NROGERENTE = NROSUPERIOR AND  
    APELLIDO ='Pérez')  
OR  
    NROPROY IN (SELECT NROPROY FROM TRABAJA_EN,  
        EMPLEADO  
    WHERE EMPLEADO.NROEMP = TRABAJA_EN.NROEMP  
    AND APELLIDO = 'Pérez');
```



CONSULTAS COMPLEJAS

- La primera consulta anidada selecciona los números de proyectos en que un Pérez participa como jefe.
- La segunda consulta anidada selecciona los números de proyectos en que un Pérez participa como trabajador.
- La consulta externa selecciona una tupla PROYECTO si el valor de NROPROY de esa tupla está en el resultado de cualquiera de las dos consultas anidadas.
- El operador de comparación IN compara el valor v con un conjunto (o multiconjunto) de valores V y evalúa a TRUE para comprobar si v es uno de los elementos de V



CONSULTAS ANIDADAS

CORRELACIONADAS

- Siempre que una condición en la cláusula WHERE de una consulta anidada hace referencia a un atributo de una relación declarada en la consulta externa, se dice que las dos consultas están correlacionadas.
- La consulta anidada se evalúa una sola vez para cada tupla (o combinación de tuplas) en la consulta externa



CONSULTAS ANIDADAS CORRELACIONADAS

- Recupere el nombre de cada empleado que tenga un familiar dependiente con el mismo nombre de pila y sexo que el empleado.

```
SELECT E.NOMBRE, E.APELLIDO  
FROM EMPLEADO AS E, DEPENDIENTE AS D  
WHERE E.NROEMPLEADO =D.NROEMPLEADO  
AND E.SEXO =D.SEXO AND E.NOMBRE  
=D.NOMBREDEP;
```



FUNCIÓN EXISTS

- EXISTS sirve para comprobar si el resultado de una consulta anidada correlacionada es o no vacío.

```
SELECT E.NOMBRE, E.APELLIDO  
FROM EMPLEADO AS E  
WHERE EXISTS (SELECT *  
                FROM DEPENDIENTE  
                WHERE E.NROEMP =  
                DEPENDIENTE.NROEMP AND SEXO = E.SEXO  
                AND E.NOMBRE =NOMBREDEPENDIENTE);
```



FUNCIÓN EXISTS

- Recupere los nombres de empleados que no tienen familiares dependientes

```
SELECT NOMBRE, APELLIDO  
FROM EMPLEADO  
WHERE NOT EXISTS (SELECT *  
                   FROM DEPENDIENTE  
                   WHERE EMPLEADO.NROEMPLEADO =  
                     DEPENDIENTE.NROEMPLEADO);
```



CONJUNTOS EXPLÍCITOS Y NULL

- Recupere el número de empleado de todos los que trabajan en los proyectos 1, 2 o 3.

```
SELECT DISTINCT NROEMPLEADO  
FROM EMPLEADO  
WHERE NROPROY IN (1,2,3);
```

- Recupere los nombres de todos los empleados que no tienen supervisores

```
SELECT NOMBRE, APELLIDO  
FROM EMPLEADO  
WHERE NROSUPERVISOR IS NULL;
```



FUNCIONES AGREGADAS Y AGRUPACIÓN

- Funciones COUNT (cuenta numero de tuplas), SUM, MAX, MIN, AVG
- Halle la suma de los salarios de todos los empleados, el salario máximo, el mínimo y el salario medio.

```
SELECT SUM(SALARIO), MAX(SALARIO),  
       MIN(SALARIO), AVG(SALARIO)  
FROM EMPLEADO;
```



FUNCIONES AGREGADAS Y AGRUPACIÓN

- Recupere el total de empleados de la empresa

```
SELECT COUNT(*)  
FROM EMPLEADO;
```

Recupere el número de empleados del departamento de Investigación.

```
SELECT COUNT(*)  
FROM EMPLEADO, DEPARTAMENTO  
WHERE DEPTO.NRODEPTO =  
      EMPLEADO.NRODEPTO AND NOMBREDEPTO  
      ='Investigación';
```



FUNCIONES AGREGADAS Y AGRUPACIÓN

- Contar valores de atributos en vez de tuplas.

```
SELECT COUNT(DISTINCT SALARO)  
FROM EMPLEADO;
```



AGRUPACIÓN

- A veces es necesario aplicar funciones agregadas a subgrupos de tuplas de una relación, por ejemplo conocer el salario medio de los empleados de cada departamento, o el número de empleados que trabajan en que cada proyecto.
- En estos casos se necesita agrupar las tuplas que tienen el mismo valor para ciertos atributos, que se llaman atributos de agrupación y aplicar la función de manera independiente a cada uno de esos grupos.
- SQL tiene la cláusula GROUP BY



AGRUPACIÓN

- Recupere el número de depto y el número de empleados de cada departamento y su salario medio.

```
SELECT NRODEPTO, COUNT(*), AVG  
  (SALARIO)  
FROM EMPLEADO GROUP BY NRODEPTO;
```



AGRUPACIÓN

- De cada proyecto, recupere su número, nombre y el número de empleados que trabajan en él

```
SELECT NROPROY, NOMBREPROY, COUNT(*)  
FROM PROYECTO, TRABAJA_EN  
WHERE PROYECTO.NROPROY =  
        TRABAJA_EN.NROPROY  
GROUP BY NROPROY, NOMBREPROY;
```



AGRUPACIÓN

- A veces se requiere recuperar valores agrupados cuando se cumplan ciertas condiciones. Por ejemplo, la consulta anterior, pero para aquellos proyectos que tengan más de dos empleados.
- Se ocupa la cláusula HAVING, que puede aparecer con la cláusula GROUP BY.
- HAVING especifica una condición en términos del grupo de tuplas asociado a cada valor de los atributos de agrupación. Sólo los grupos que satisfagan la condición entrarán en el resultado de la consulta.



AGRUPACIÓN

```
SELECT NROPROY, NOMBREPROY, COUNT(*)  
FROM PROYECTO, TRABAJA_EN  
WHERE PROYECTO.NROPROY =  
        TRABAJA_EN.NROPROY  
GROUP BY NROPROY, NOMBREPROY  
HAVING COUNT(*) >2;
```



ESPECIFICACIÓN DE RESTRICCIONES GENERALES

```
CREATE ASSERTION RESTRIC_SALARIO  
CHECK (NOT EXISTS (SELECT * FROM  
EMPLEADO E, EMPLEADO M,  
DEPARTAMENTO D  
WHERE E.SALARIO > M.SALARIO AND  
E.ND = D.NUMEROD AND  
D.NSS_JEFE = M.NSS))
```

- Se puede usar el CHECK en la creación de dominios.



ASSERTION/CHECK

- Especifica cuando se viola una restricción general, lo que implica el aborto de una actualización.

