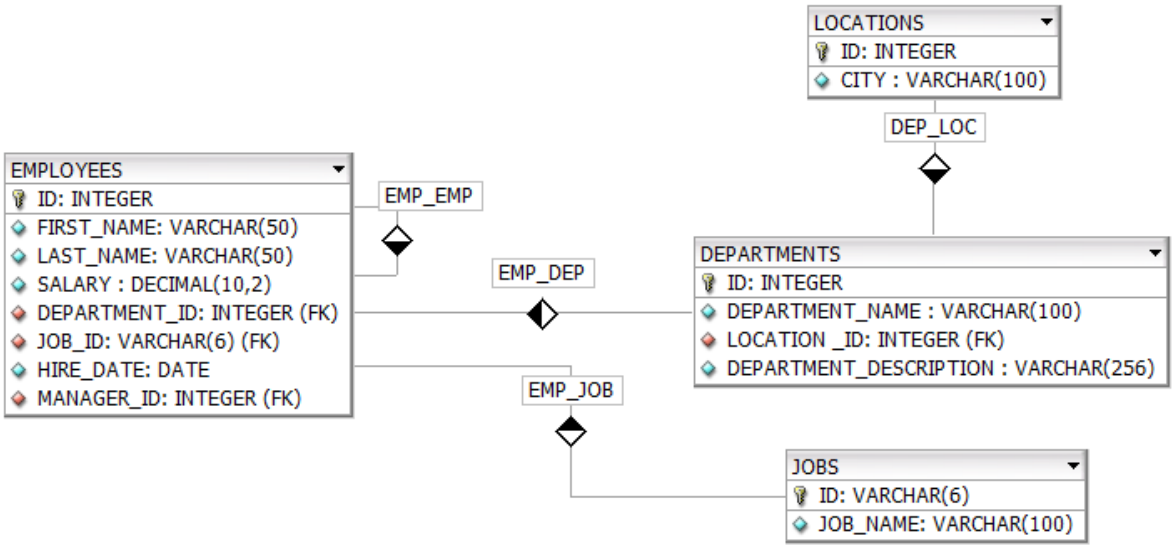
Nociones SQL - Práctica
V2 - 16/10/2015



A – Recuperación básica de datos

SQL Básico / Tipos de datos / Operaciones / Alias de columnas / Valor nulo

Sintaxis:
SELECT *|{[DISTINCT] *column|expression* [*alias*],...}
FROM *table;*

1- Recuperar lista de empleados

2- Recuperar **id, apellido, fecha de contratación** de los empleados

3- Recuperar id, apellido, fecha de contratación, **salario** de los empleados.
Tip: notar presencia de valores nulos

4- Recuperar id, apellido, fecha de contratación, **salario anual** de los empleados.
Tip: Calcular el salario anual como 12 veces el salario. Usar alias para el sueldo anual.

5- Recuperar id, **apellido y nombre**, fecha de contratación, salario anual de los empleados.
Tip: Concatenar usando ||. Notar que los operadores a usar dependen del tipo de dato de los campos.

6- Recuperar lista de departamentos que tienen empleados:
6.a- Recuperar lista de departamentos de los empleados
6.b- Recuperar lista no repetida de departamentos de los empleados

B – Comparaciones simples y especiales / Comparaciones nulas

Sintaxis:
SELECT *|{[DISTINCT] *column|expression* [*alias*],...}
FROM *table*
[WHERE *condition(s)*];

7- Recuperar lista de empleados cuyo departamento sea 10.

8- Recuperar lista de empleados cuyo salario sea menor a 2000.

9- Recuperar lista de empleados cuyo salario sea entre 1800 y 3000
Tip: usar cláusula "between". Notar diferencia con el uso de 2 condiciones.

10- Recuperar lista de empleados cuyo departamento sea 10 o 30 o 31.
Tip: usar cláusula "in".

11- Recuperar lista de empleados cuyo apellido empiece con F.
Tip: usar cláusula "like". Notar que los operadores a usar dependen del tipo de dato de los campos.

12- Recuperar lista de empleados cuyo job_id:
12.a- no hay sido definido
12.b- haya sido definido.

13- Recuperar lista de empleados cuyo job_id sea distinto de 'AD_CTB'. Tip: Notar comportamiento de la condición con jobs nulos.

C- Comparaciones con nexos lógicos / Precedencia de condiciones

14- Recuperar lista de empleados cuyo job_id sea distinto de 'AD_CTB' **y** cuyo salario sea mayor a 1900.

15- Recuperar lista de empleados cuyo job_id sea distinto de 'AD_CTB' **o** cuyo salario sea mayor a 1900.

16- Recuperar lista de empleados cuyo job_id **sea 'AD_CTB' o 'FQ_GRT'** (sin usar IN) y cuyo salario sea mayor a 1900.
Tip: Probar precedencia de condiciones con o sin paréntesis.

D- Ordenamiento

SELECT *|{[DISTINCT] *column|expression* [*alias*],...}
FROM *table* [*alias*]
[WHERE *condition(s)*]
[ORDER BY {*column, expr, alias*} [ASC|DESC]];

17- Recuperar empleados ordenados por fecha de ingreso (desde más viejo a más nuevo).

18- Recuperar empleados ordenados por fecha de ingreso (desde más nuevo a más viejo).

19- Recuperar empleados ordenados por fecha de ingreso descendente y apellido ascendente.

20- Recuperar apellido y salario anual de empleados ordenados por salario anual.
Tip: Usar alias de columna para ordenar por salario anual.

E- Recuperación de datos de múltiples tablas

SELECT *|{[DISTINCT] *column|expression* [*alias*],...}
FROM {*table* [*alias*],…}
[WHERE *condition(s)*]
[ORDER BY {*column, expr, alias*} [ASC|DESC]];

21- Recuperar lista de empleados con la descripción del departamento al que cada uno pertenece.
Tip: evitar producto cartesiano.
Completar: select * from TEST.EMPLOYEES, …

22- Seleccionar apellido de empleado y nombre de departamento

23- Agregar id de empleado y id de departamento
Tip: desambiguar campos usando alias de tablas.

24- Recuperar lista de empleados con descipción de departamentos y ciudades.

F- Uso de cláusula JOIN

25- Recuperar lista de empleados con la descripción del departamento al que cada uno pertenece.
Completar: select * from TEST.EMPLOYEES join …
26- Recuperar lista de empleados con la descripción del departamento, tengan o no departamento asignado.
27- Recuperar lista de departamentos con empleados de cada departamento, tengan o no empleados asociados.

G- Selfjoin

28- Recuperar lista de subordinados por cada manager

H- Funciones de agrupamiento

SELECT [*column*,] *group_function*(*column*), ...
FROM *table*
[WHERE *condition*]
[GROUP BY *column*]
[ORDER BY *column*];

29- Recuperar máximo salario de los empleados.

30- Recuperar máximo, mínimo, promedio, y suma total de salarios de los empleados.

31- Recuperar máximo, mínimo, promedio, y suma total de fecha de contratación de los empleados.
Tip: Notar que las funciones de agrupamiento permitidas dependen del tipo de dato.

32- Obtener la cantidad de empleados.

33- Obtener la cantidad de empleados cuyo departamento sea 10.

34- Obtener la cantidad de empleados de cada departamento.

35- Obtener la cantidad de empleados por cada departamento y job.

I- Condiciones de grupo

36- Recuperar los departamentos y el salario promedio de cada departamento.

37- Recuperar los departamentos y el salario promedio si es menor a 1200.

J- Creación de nuevos registros

INSERT INTO *table*
[({*column*, })]
VALUES ({DEFAULT|*value* , })

38- Crear un nuevo departamento

38.a- Caso 1: Crear insert de todos los campos en orden.
Tip: Notar restricciones de integridad por padre inexistente y por clave duplicada.
`Completar: insert into TEST.DEPARTMENTS VALUES …`

38.b- Caso 2: Crear insert de todos los campos en orden usando valores nulos.
Tip: Notar restricciones de no nulidad.

38.c- Crear insert usando solamente los campos obligatorios.
Tip: Especificar lista de campos obligatorios.
`Completar: insert into TEST.DEPARTMENTS (ID, …`

K- Creación de nuevos registros en base a registros existentes

INSERT INTO *table*
[({*column*, })]
SELECT …

39- Crear un nuevo empleado basado en los datos de Gustavo Boulette:
- cambiando su nombre
- aumentando su sueldo en $200.
- blanqueando su manager

L- Actualización de registros

UPDATE *table*
SET {*column = DEFAULT|value|expression*, …}
WHERE *condition*

40- Actualizar salario del empleado 10 a $1100.

41- Duplicar salario del empleado 11.

42- Aumentar salario en un 10% a todos los empleados del departamento 40.

M- Eliminación de registros

DELETE FROM *table*
WHERE *condition*

43- Eliminar departamentos cuyo id sea mayor a 50.
Tip: hacer un select antes y después para verificar usando la misma condición que para el delete.

44- Eliminar departamento 40.
Tip: notar resultado de las restricciones de integridad.

N-Crear una Función

45- Crear la función "fn_AntiguedadEmpleado" que retorne la antiguedad en años de cada empleado donde el parametro de ingreso es el id del empleado

O-Crear un Procedimiento almacenado

46 - Crear el Procedimiento almacenado "sp_GetNombreAntiguedad" que retorne el primer nombre y el apellido separados por una coma y en la segunda columna la antiguedad en año. Usar la función creada en el punto anterior.
Ordenar por antiguedad descendiente (mas antiguo primero)

NombreyApellido	Antiguedad
José, Michetti	7
Ernesto, Gonzalez	7