Bases de datos 1

SQL

Lenguaje DDL

- Data Definition Language.
- Lenguajes que usan los DBA’s y Diseñadores de BD para crear los esquemas y las descripciones de estos esquemas en el catálogo de la BD.
- Create table.
- Drop Table.
- Alter table.

Lenguaje DML

- Data Manipulation Language.
- El SABD proporciona el lenguaje DML.
- Cuando ya se tiene información (datos) en la BD, se va a requerir manipular esos datos.
- Seleccionar.
- Insertar.
- Borrar.
- Modificar.

Lenguaje DDL

Tablespace

- Creación del tablespace.

Schema (crear un esquema)

- Conjunto de tablas y objetos de la cuenta de un usuario.

Create table (crear una tabla)

```
CREATE TABLE employee
(
    employee_id    NUMBER(6),
    first_name    VARCHAR2(20),
    last_name    VARCHAR2(25) CONSTRAINT employee_lastname_nn NOT NULL,
    email    VARCHAR2(25) CONSTRAINT employee_email_nn    NOT NULL,
    CONSTRAINT    employee_email_uk   UNIQUE (email),
    phone_number    VARCHAR2(20),
    hire_date    DATE  DEFAULT SYSDATE CONSTRAINT employee_hiredate_nn NOT NULL,
    salary    NUMBER(8,2) CONSTRAINT    employee_salary_nn  NOT NULL,
    CONSTRAINT    employee_salary_min CHECK (salary > 0),
    department_id    NUMBER(4)
);
```

Create table (crear una tabla)

```
CREATE TABLE employee
(
    employee_id    NUMBER(6),
    first_name    VARCHAR2(20),
    last_name    VARCHAR2(25) CONSTRAINT emp_last_name_nn_demo NOT NULL,
    email    VARCHAR2(25) CONSTRAINT emp_email_nn_demo     NOT NULL,
    phone_number    VARCHAR2(20),
    ……..
)

TABLESPACE ge_data
STORAGE
(
    INITIAL    6144
    NEXT    6144
    MINEXTENTS    1
    MAXEXTENTS    5
);
```

Create table (crear una tabla)

```
CREATE TABLE department
(
    department_id    NUMBER(4)   PRIMARY KEY,
    department_name    VARCHAR2(30) CONSTRAINT  dept_name_nn  NOT NULL,
    manager_id    NUMBER(6),
    location_id     NUMBER(4),
    dn    VARCHAR2(300)
)

TABLESPACE ge_data
STORAGE
(
    INITIAL    6144
    NEXT    6144
    MINEXTENTS    1
    MAXEXTENTS    5
);
```

Create table (crear una tabla)

```
CREATE TABLE range_sales
(
    prod_id    NUMBER(6),
    cust_id    NUMBER,
    time_id    DATE,
    channel_id    CHAR(1),
    promo_id    NUMBER(6),
    quantity_sold    NUMBER(3),
    amount_sold    NUMBER(10,2)
)

PARTITION BY RANGE (time_id)
(  PARTITION SALES_Q1_1998 VALUES LESS THAN (TO_DATE('01-APR-1998','DD-MON-YYYY')),
  PARTITION SALES_Q2_1998 VALUES LESS THAN (TO_DATE('01-JUL-1998','DD-MON-YYYY')),
  PARTITION SALES_Q3_1998 VALUES LESS THAN (TO_DATE('01-OCT-1998','DD-MON-YYYY')),
  PARTITION SALES_Q4_1998 VALUES LESS THAN (TO_DATE('01-JAN-1999','DD-MON-YYYY')),
  PARTITION SALES_Q1_1999 VALUES LESS THAN (TO_DATE('01-APR-1999','DD-MON-YYYY')),
  PARTITION SALES_Q2_1999 VALUES LESS THAN (TO_DATE('01-JUL-1999','DD-MON-YYYY')),
  PARTITION SALES_Q3_1999 VALUES LESS THAN (TO_DATE('01-OCT-1999','DD-MON-YYYY')),
  PARTITION SALES_Q4_1999 VALUES LESS THAN (TO_DATE('01-JAN-2000','DD-MON-YYYY')),
  PARTITION SALES_Q1_2000 VALUES LESS THAN (TO_DATE('01-APR-2000','DD-MON-YYYY')),
  PARTITION SALES_Q2_2000 VALUES LESS THAN (TO_DATE('01-JUL-2000','DD-MON-YYYY')),
  PARTITION SALES_Q3_2000 VALUES LESS THAN (TO_DATE('01-OCT-2000','DD-MON-YYYY')),
  PARTITION SALES_Q4_2000 VALUES LESS THAN (MAXVALUE));
```

Create table (crear una tabla)

```
CREATE TABLE employee
(
    employee_id    NUMBER(6),
    CONSTRAINT    pk_employee    PRIMARY KEY (employee_id ),
    first_name    VARCHAR2(20),
    last_name    VARCHAR2(25) CONSTRAINT emp_last_name_nn_demo NOT NULL,
    email    VARCHAR2(25) CONSTRAINT emp_email_nn_demo     NOT NULL,
    phone_number    VARCHAR2(20),
    ……..
);
```

Alter table (modifica una tabla)

```
ALTER TABLE employee
    ADD CONSTRAINT pk_employee PRIMARY KEY (employee_id)
    USING INDEX
    TABLESPACE ge_ind PCTFREE 20
    STORAGE (INITIAL10K NEXT10K PCTINCREASE 0);
```

Alter table (modifica una tabla)

```
ALTER TABLE employee
    ADD birthdate DATE;

ALTER TABLE employee
    RENAME COLUMN birthdate TO birth_date;
```

Delete table (borra información de una tabla)

```
DELETE FROM Employee;

DELETE FROM Employee
    WHERE employee_id = 123456;
```


UN MINUTO DE SILENCIO POR LOS QUE HICIERON UPDATE Y DELETE
SIN EL WHERE
memegenerator.net

Drop table (borra una tabla)

```
DROP TABLE Employee;
```

Modify table (modifica una tabla)

```
ALTER TABLE table_name
    MODIFY column_name  DATATYPE;

ALTER TABLE table_name
    MODIFY (column1_name  column1_datatype,
                column2_name  column2_datatype,
                column3_name  column3_datatype,
                 column4_name  column4_datatype);
```

Modify table (modifica una tabla)

```
ALTER TABLE customer
    MODIFY (name varchar2(100) not null,
                hair_color  varchar2(20));
```

Create view (Crea una vista)

```
CREATE VIEW emp_sal (emp_id, last_name, email)
    AS SELECT employee_id, last_name, email FROM employee;
```

Create index (Crea un índice en una tabla)

```
CREATE UNIQUE INDEX department_name_ui ON department
(department_name)
    TABLESPACE ge_ind;
```

Constraints (Creación de restricciones)

```
CREATE TABLE customer
(
    customer_id    NUMBER(6),
    first_name    VARCHAR2(20),
    last_name    VARCHAR2(25) CONSTRAINT cust_lastname_nn NOT NULL,
    email    VARCHAR2(25) CONSTRAINT cust_email_nn     NOT NULL,
    CONSTRAINT cust_email_uk     UNIQUE (email),
    phone_number   VARCHAR2(20),
    birth_date   DATE  DEFAULT SYSDATE CONSTRAINT cust_birth_date_nn NOT NULL,
    salary NUMBER(8,2) CONSTRAINT    cust_salary_nn  NOT NULL,
    CONSTRAINT    cust_salary_min CHECK ( salary > 0)
);
```

Defaults (Creación de valores por defecto)

```
CREATE TABLE customer
(
    customer_id    NUMBER(6),
    first_name    VARCHAR2(20),
    last_name    VARCHAR2(25) CONSTRAINT cust_lastname_nn NOT NULL,
    email    VARCHAR2(25) CONSTRAINT cust_email_nn     NOT NULL,
    CONSTRAINT cust_email_uk     UNIQUE (email),
    phone_number   VARCHAR2(20),
    birth_date   DATE  DEFAULT SYSDATE CONSTRAINT cust_birth_date_nn NOT NULL,
    salary NUMBER(8,2) CONSTRAINT    cust_salary_nn  NOT NULL,
    CONSTRAINT    cust_salary_min CHECK ( salary > 0)
);
```

Foreing Key

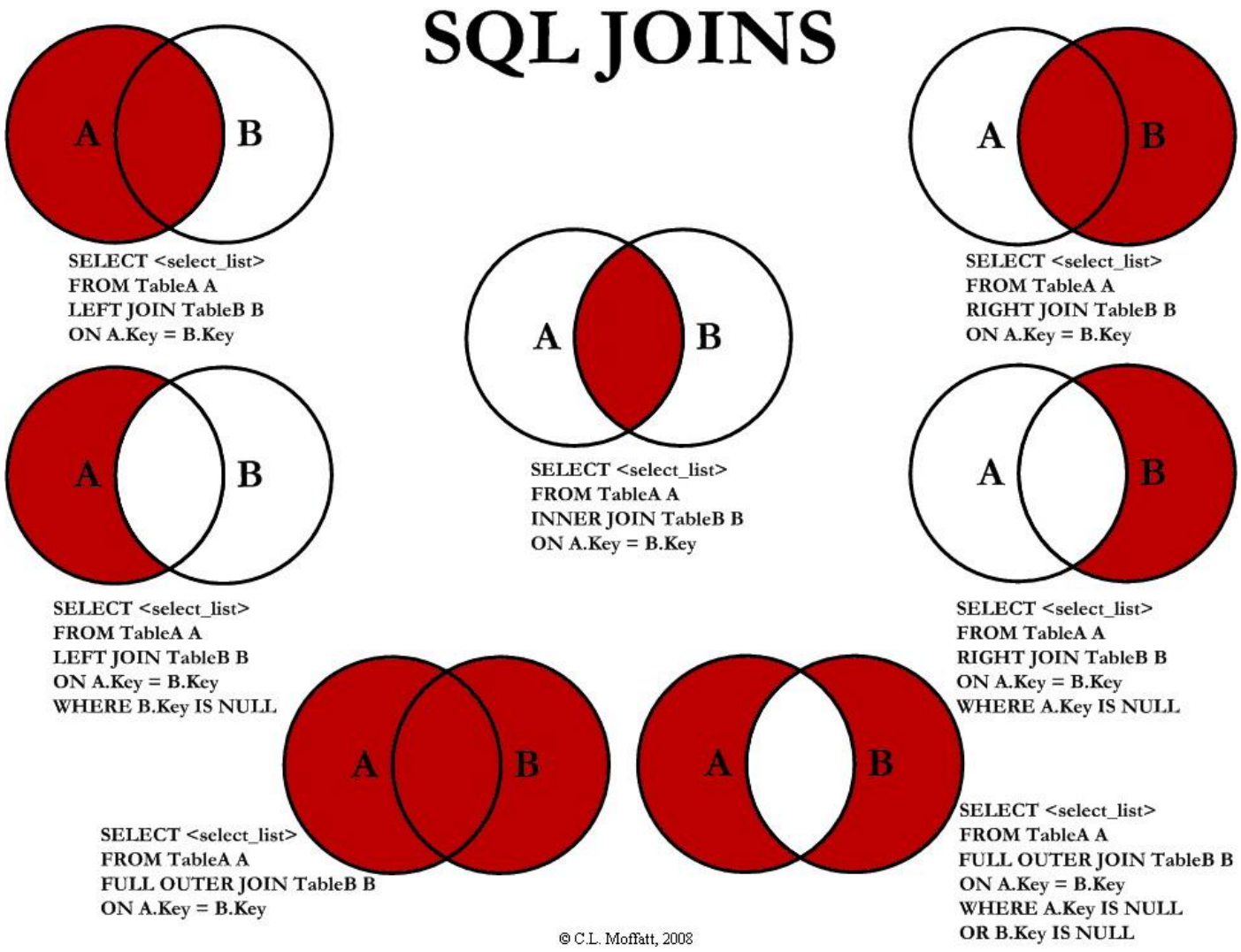
```
ALTER TABLE product_order
    ADD CONSTRAINT FK_productorder_customer FOREIGN KEY  (customer_id)
    REFERENCES customer(customer_id);
```

Lenguaje DML

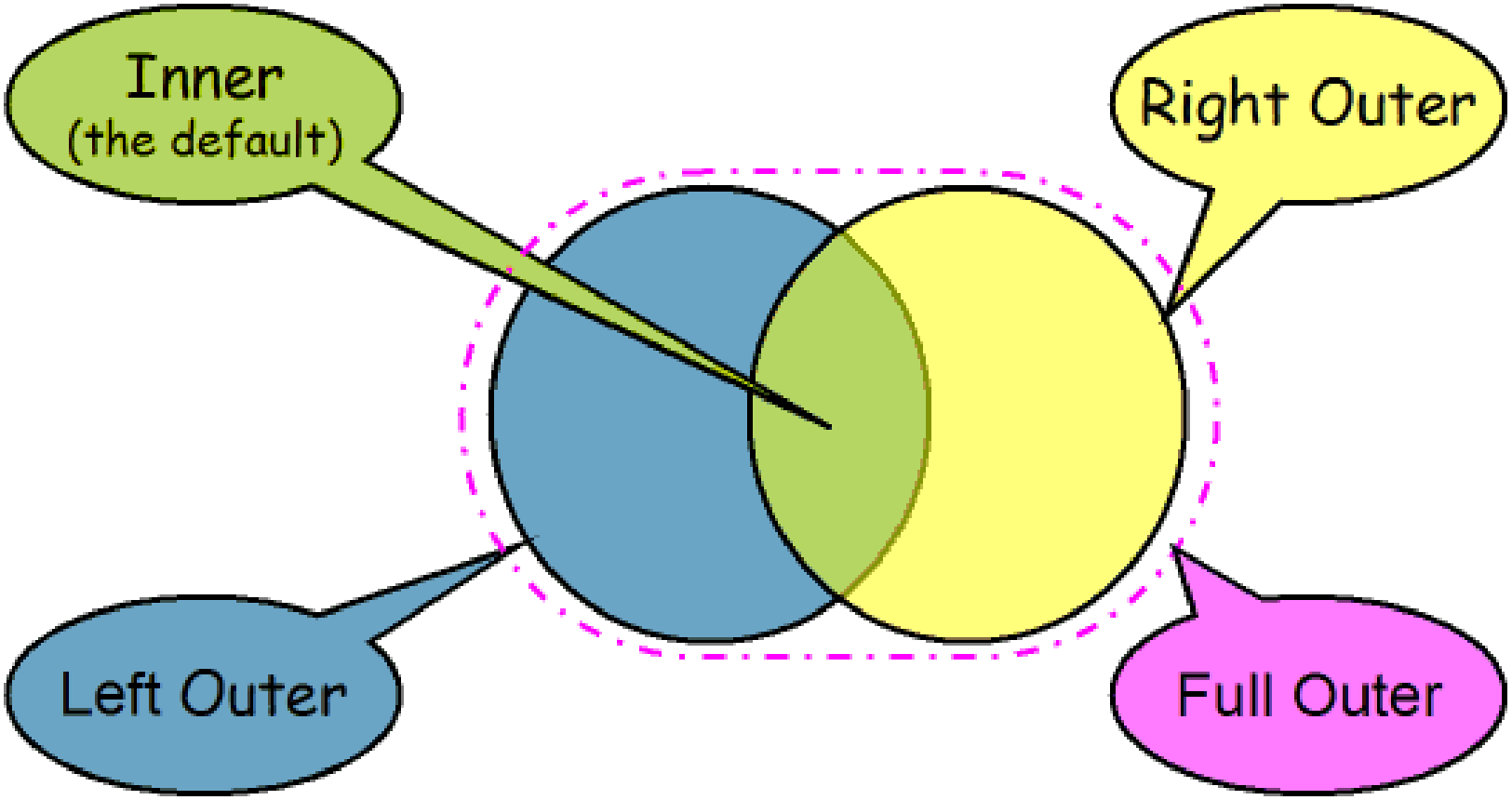
SQL


Our Servers are running slow. Check Indexes,
Wait Stats, Query Joins and Various Bottlenecks. NOW!
Now?

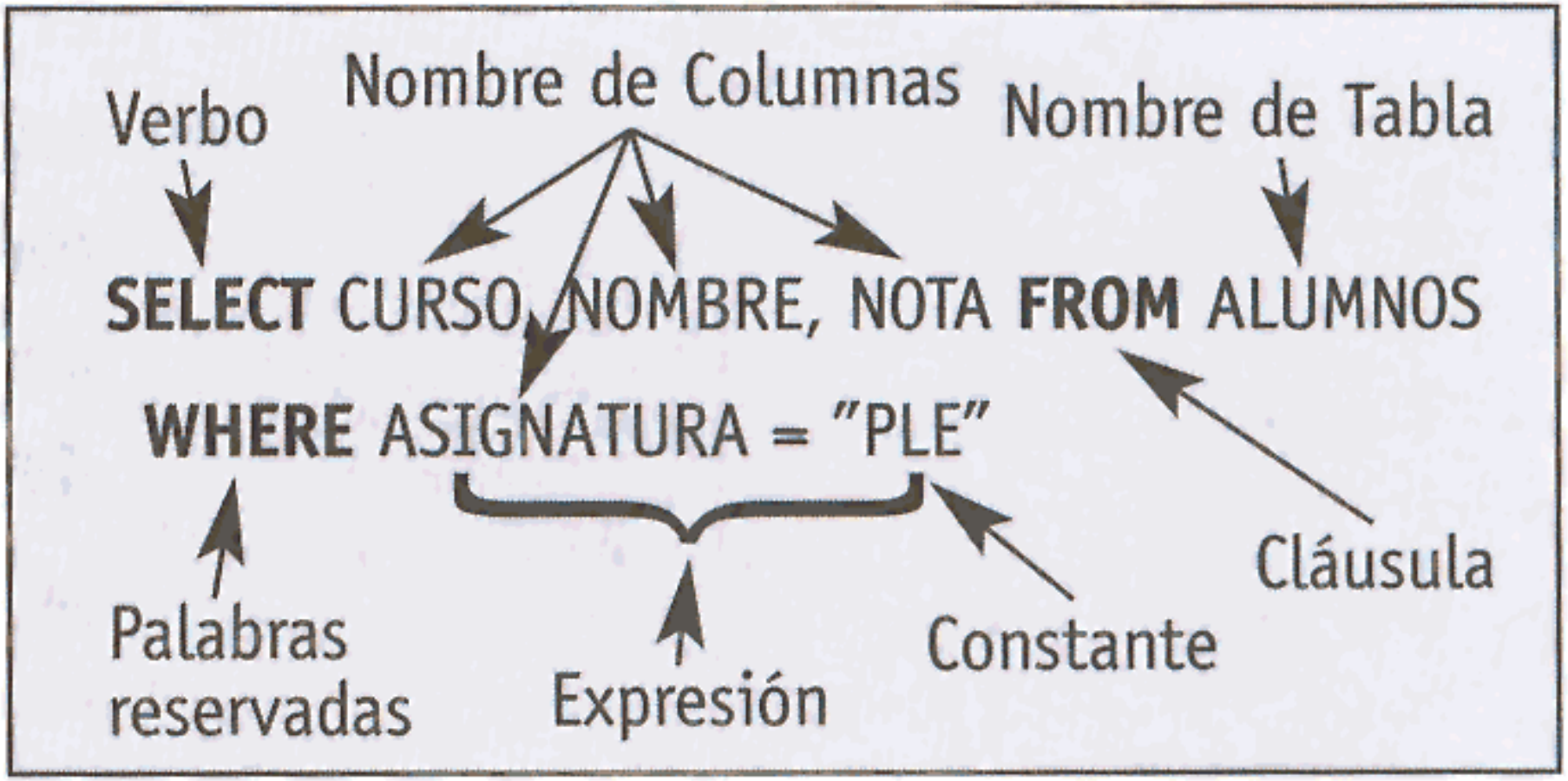
Introducción al SQL



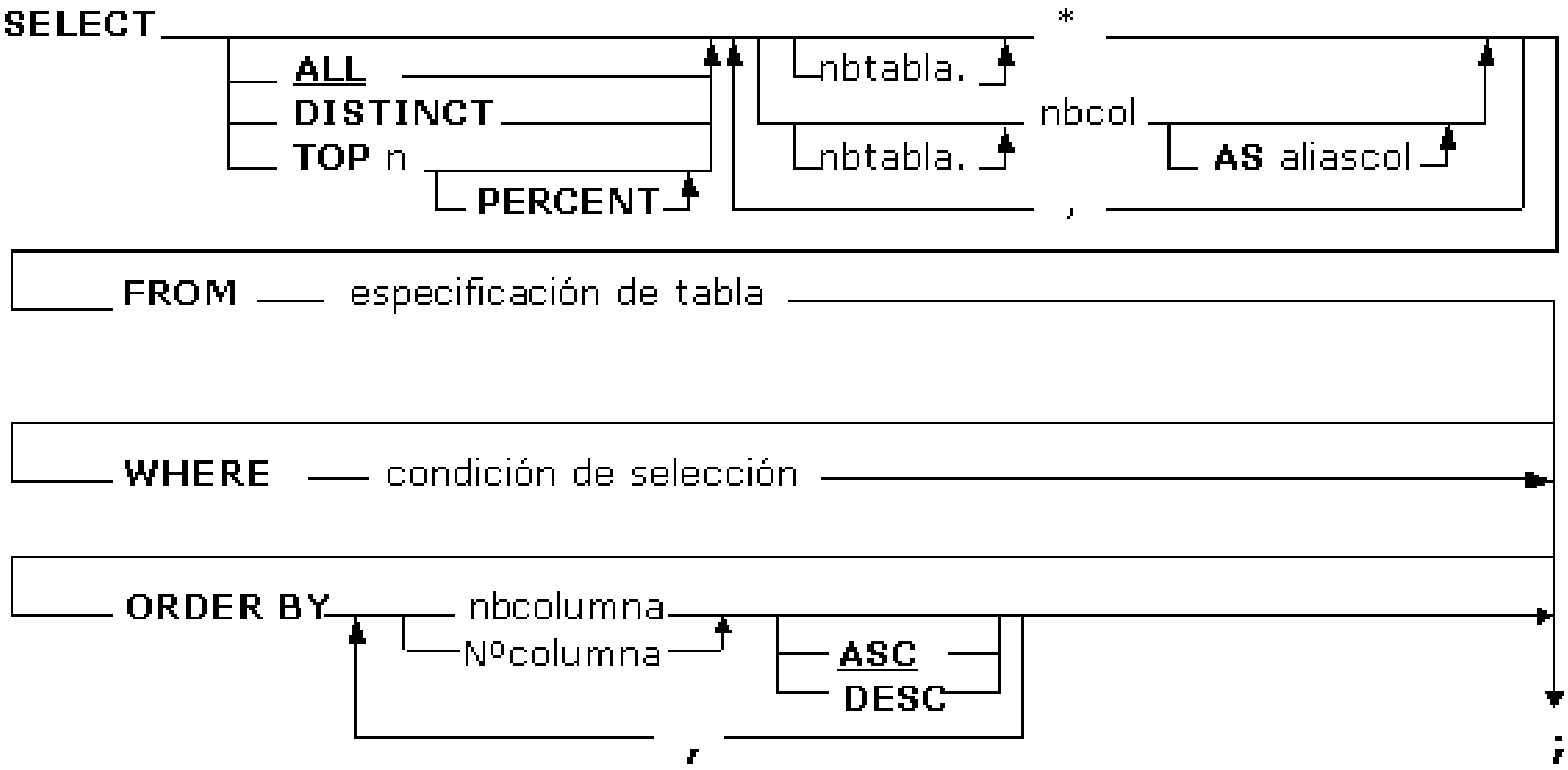
Introducción al SQL



Generación de sentencias SQL



Generación de sentencias SQL



Consultas

- Select
- Join
- Group by
- Order by
- Exists

Select

- Select //lista de atributos a desplegar
- From //lista de nombres de las relaciones involucradas
- Where //predicado de calificación sobre las tuplas – condiciones -
- Group by //columna(s) que permiten realizar agrupaciones de las tuplas que satisfacen al condición Where
- Having //predicado que deben satisfacer los grupos que se formen por la clausula Group by
- Union //permite hacer la unión de dos bloques select.
- Order by //permite ordenar la tuplas

Select (Retorna registros de una tabla)

```
SELECT *
FROM employee
WHERE department_id = 30
ORDER BY last_name desc

SELECT last_name, job_id, salary, department_id
FROM employee
WHERE NOT (job_id = 'PU_CLERK' AND department_id = 30)
ORDER BY last_name;
```

Select (Retorna registros de una tabla)

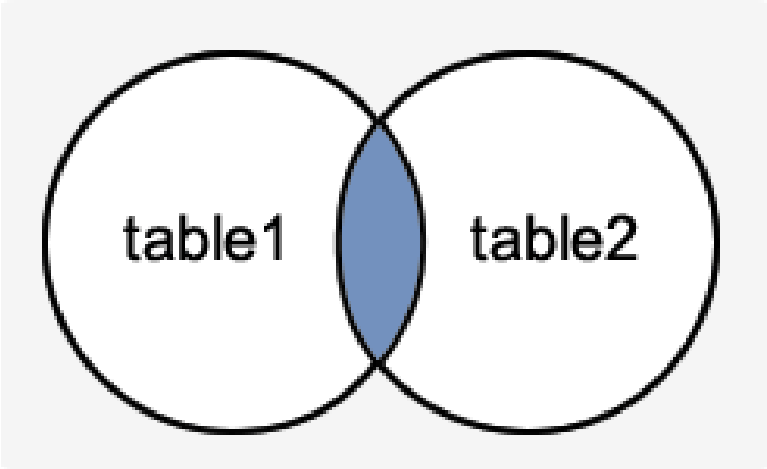
```
SELECT department_id, MIN(salary), MAX (salary)
FROM employee
GROUP BY department_id
ORDER BY department_id;

SELECT department_id, MIN(salary), MAX (salary)
FROM employees
WHERE job_id = 'PU_CLERK'
GROUP BY department_id
ORDER BY department_id;
```

Select (Retorna registros de una tabla)

```
SELECT department_id, MIN(salary), MAX (salary)
FROM employees
GROUP BY department_id
HAVING MIN(salary) < 5000
ORDER BY department_id;
```

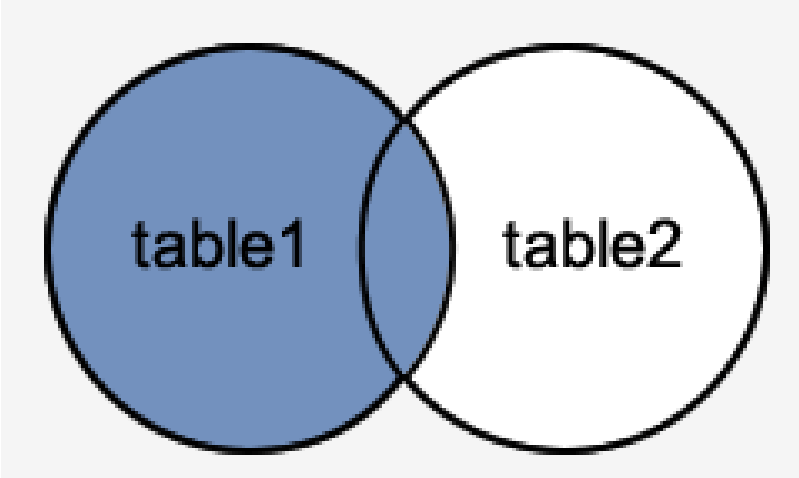
Simple join / Inner Join

- SELECT columns
- FROM table1
- INNER JOIN table2
- ON table1.column = table2.column;



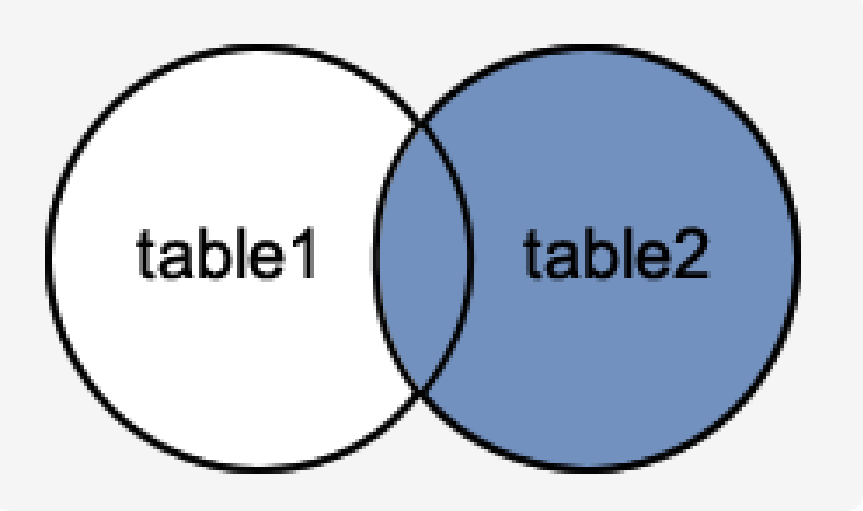
LEFT OUTER JOIN

- SELECT columns
- FROM table1
- LEFT [OUTER] JOIN table2
- ON table1.column = table2.column;



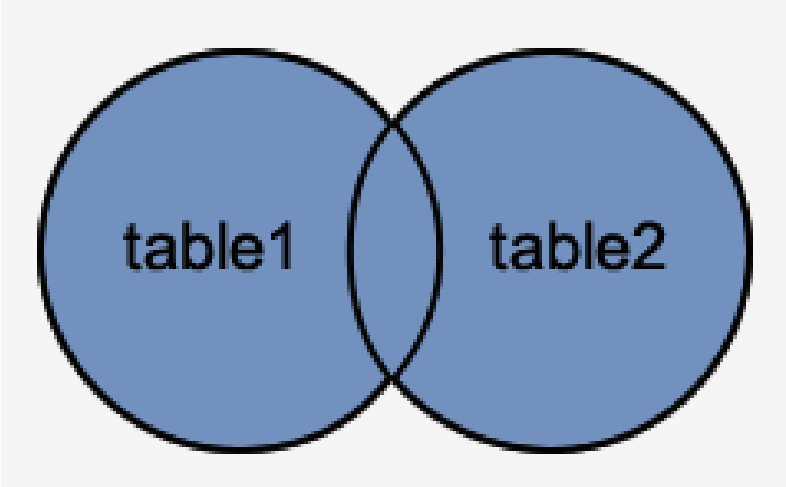
RIGHT OUTER JOIN

- SELECT columns
- FROM table1
- RIGHT [OUTER] JOIN table2
- ON table1.column = table2.column;



FULL OUTER JOIN

- SELECT columns
- FROM table1
- FULL [OUTER] JOIN table2
- ON table1.column = table2.column;



Select

```
SELECT a.department_id "Department",
    a.num_emp/b.total_count "%_Employees",
    a.sal_sum/b.total_sal "%_Salary"
FROM
    (SELECT department_id, COUNT(*) num_emp,
        SUM(salary) sal_sum
          FROM employees
          GROUP BY department_id) as a,
     (SELECT COUNT(*) total_count, SUM(salary)
          total_sal
          FROM employees) as b
ORDER BY a.department_id;
```

Insert (Inserta registros en una tabla)

```
INSERT INTO department (id, descripcion, id_manager)
VALUES (280, 'Recreation', 1);

INSERT INTO department
VALUES (280, 'Recreation', DEFAULT, 1700);

INSERT INTO bonuses
SELECT employee_id, salary*1.1
FROM employees
WHERE commission_pct > 0.25;
```

Insert (Inserta registros en una tabla)

```
INSERT INTO employee
(employee_id, last_name, email, hire_date, job_id, salary,
commission_pct)
VALUES
(207, 'Gregory', 'pgregory@oracle.com', sysdate, 'PU_CLERK',
1500, NULL);

INSERT INTO employee
(employee_id, last_name, email,vhire_date, job_id, salary,
commission_pct)
(SELECT employee_id, last_name, email, hire_date, job_id, salary,
commission_pct FROM employee_temporal);
```

Insert (Inserta registros en una tabla)

```
INSERT INTO bonuses
SELECT employee_id, salary*1.1
FROM employees
WHERE commission_pct > 0.25;

INSERT INTO
(SELECT employee_id, last_name, email, hire_date, job_id, salary,
commission_pct FROM employees)
VALUES (207, 'Gregory', 'pgregory@oracle.com', sysdate,
'PU_CLERK', 1500, NULL);
```

Insert (Inserta registros en una tabla)

```
INSERT INTO department
VALUES  (departments_seq.nextval, 'Entertainment', 162, 1400);
```

Delete (Borra registros en una tabla)

```
DELETE FROM product_descriptions
WHERE language_id = 'AR';

DELETE FROM employee
WHERE job_id = 'SA_REP'
AND commission_pct < 0.2;

DELETE FROM (SELECT * FROM employee)
WHERE job_id = 'SA_REP'
AND commission_pct < 0.2;
```

Update (Cambia información en registros en una tabla)

```
UPDATE user
SET name = ‘John'
WHERE id = ‘107890876';

UPDATE employee
SET job = ‘Vendedor', salary = salary + 1000, department_id  = 140
WHERE id_employee = ‘107890876‘;

UPDATE employee
SET salary = salary * 1.1
WHERE department_id = 100
```

Update (Cambia información en registros en una tabla)

```
UPDATE employee
SET JOB = 'MANAGER'
WHERE id_employee = ' 123740652';
```

Exists

```
SELECT *
FROM employee e
WHERE EXISTS
    (SELECT *
     FROM department d
     WHERE  e.department_id = d.department_id
     AND d.department_id=20);
```

- SQL Statements that use the SQL EXIST Condition are very inefficient since the sub-query is RE-RUN for EVERY row in the outer query's table. There are more efficient ways to write most queries, that do not use the SQL EXISTS Condition.

Not Exists

```
SELECT *
FROM employee e
WHERE NOT EXISTS
    (SELECT Select *
     FROM department d
     WHERE e.department_id = d.department_id
     AND d.department_id=20)
```

- SQL Statements that use the SQL EXIST Condition are very inefficient since the sub-query is RE-RUN for EVERY row in the outer query's table. There are more efficient ways to write most queries, that do not use the SQL EXISTS Condition.

.

Comments

```
COMMENT ON TABLE employee
IS  'This is a table for Employee.';

COMMENT ON COLUMN employee.name
IS  ‘Nombre de pila del empleado';
```

Comments

```
SELECT comments
FROM sys.all_col_comments
WHERE OWNER = :MyOwner
AND TABLE_NAME= :MyTable
AND COLUMN_NAME= :MyColumn;
```

Dates

- http://oracletuts.net/tutorials/how-to-calculate-difference-between-dates-in-oracle-sql/

Funciones de cálculo

- Count(*): brinda el número de tuplas que satisfacen la cláusula WHERE.
- Count(a): brinda el número de valores de la columna a.
- Sum(a): brinda la suma total de los valores en la columna a.
- Max(a): brinda el mayor valor de la columna a.
- Min(a): brinda el menor valor de la columna a.
- Avg(a): brinda el valor promedio de los valores de la columna a

Having

```
SELECT tipo
FROM sitio
GROUP BY tipo
HAVING COUNT(*) > = 3 ;
```

Gracias - ¿Preguntas?