Structured Query Language(SQL) IN ORACLE

SQL

- SQL ban đầu được phát triển tại IBM bởi Donald D. Chamberlin và Raymond F. Boyce vào những năm 1970.
- Phiên bản này được gọi là *SEQUEL* (*Structured English Query Language*).
- Vào cuối những năm 1970, Oracle thấy được tiềm năng của những khái niệm này và phát triển ngôn ngữ SQL riêng của họ dựa trên RDBMS.

Câu lệnh SQL trong Oracle

- Câu lệnh trong SQL của Oracle được phân làm những loại sau:
  - Ngôn ngữ định nghĩa dữ liệu (DDL).
  - Ngôn ngữ thao tác dữ liệu (DML).
  - Quản lý giao tác (Transaction Control Statements).
  - Quản lý session (Session Control Statements).
  - Quản lý hệ thống (System Control Statement).
  - Câu lệnh SQL nhúng (Embedded SQL Statements).

Ngôn ngữ truy vấn dữ liệu

```
SELECT [DISTINCT] col_name | expr
FROM table
[WHERE condition1]
[GROUP BY col_name]
[HAVING condition2]
[ORDER BY col_name ASC| DESC]
```

Toán tử BETWEEN

- Sử dụng Between để hiển thị những row dựa trên một miền giá trị

```
SELECT  last_name, salary
FROM    employees
WHERE  salary BETWEEN 2500 AND 3500;
```

Toán tử IN

- Sử dụng toán tử IN để kiểm tra giá trị có nằm trong một danh sách không

```
SELECT last_name, salary, manager_id
FROM employees
WHERE manager_id IN (100, 101, 201);
```

Sử dụng điều kiện NULL

- Kiểm tra null với việc sử dụng IS NULL

```
SELECT last_name, manager_id
FROM employees
WHERE manager_id IS NULL;
```

- Ghi chú: không thể sử dụng '=' khi làm việc với null
  - Null không thể bằng hay không bằng với bất kỳ giá trị nào.

Mệnh đề ORDER BY

- Dùng để sắp xếp dữ liệu:
  - ASC: sắp xếp tăng dần (mặc định)
  - DESC: sắp xếp giảm dần
- Mệnh đề order by xuất hiện sau cùng trong một câu truy vấn:

```
SELECT last_name, department_id, hire_date
FROM employees
ORDER BY hire_date;
```

Mệnh đề ORDER BY

- Sắp xếp theo thứ tự giảm dần:

```
SELECT  last_name, department_id, hire_date
FROM   employees
ORDER BY
```

- Sắp xếp dựa trên column alias:

```
SELECT   last_name, salary*12 annsal
FROM   employees
ORDER BY
```

Mệnh đề ORDER BY (cont.)

- Sắp xếp dựa trên vị trí column:

```
SELECT  last_name, job_id, hire_date, salary
FROM   employees
ORDER BY
```

- Sắp xếp nhiều column:

```
SELECT   last_name, job_id, salary
FROM   employees
ORDER BY
```

Sử dụng DEFINE

- Sử dụng DEFINE để tạo và gán giá trị cho một biến:
- Sử dụng **UNDEFINE** để loại bỏ một biến:

```
DEFINE

SELECT last_name, job_id, hire_date, salary
FROM employees
WHERE employee_id =

UNDEFINE employee_num
```

Hàm (function)

Hàm chuyển đổi hoa-thường

Hàm	Kết quả
LOWER('SQL Course')	sql course
UPPER('SQL Course')	SQL COURSE
INITCAP('SQL Course')	Sql Course

Hàm chuyển đổi hoa-thường Ví dụ

```
SELECT last_name, job_id, salary
FROM employees
WHERE last_name = 'peng';
```

0 rows returned.

```
SELECT   last_name, job_id, salary
FROM   employees
WHERE  LOWER(last_name) = 'peng';
```

1 rows returned.

Hàm xử lý ký tự

Hàm	Kết quả
SUBSTR('HelloWorld', 1,5)	Hello
LENGTH('HelloWorld')	10
INSTR('HelloWorld', 'W')	6
LPAD(salary, 10, '*')	*****24000
RPAD(salary, 10, '*')	24000*****

Hàm xử lý trên số

Hàm	Kết quả
ROUND(45.926, 2)	45.93
TRUNC(45.926, 2)	45.92
MOD(1600, 300)	100

Ví dụ: TRUNC

```
SELECT TRUNC(45.923)
FROM DUAL;
```

- Result: **45** (zero decimal places)
- Note **DUAL** is a dummy table you can use to view results from functions and calculations

Datetime function

- ADD_MONTHS, MONTHS_BETWEEN, ROUND (date)
- CURRENT_DATE, NEXT_DAY, LAST_DAY, SYSDATE
- EXTRACT (datetime), TO_CHAR (datetime)
  TRUNC (date)

Datetime Functions

ADD_MONTHS, MONTHS_BETWEEN, ROUND

- ADD_MONTHS
  - add_months('01-Aug-03', 3) = 01-Nov-03
  - add_months('01-Aug-03', -3) = 01-MAY-03
  - add_months('31-Jan-03', 1) = 28-FEB-03
- MONTHS_BETWEEN
  - months_between('02-Feb-1995', '01-Jan-1995') = 1.03225806
- ROUND(date [, fmt])
  - ROUND (TO_DATE ('27-OCT-00'),'YEAR') = 01-JAN-01

19

Datetime Functions

CURRENT_DATE, NEXT_DAY, LAST_DAY, SYSDATE

- CURRENT_DATE returns the current date in the session time zone
- NEXT_DAY('02-FEB-2001','TUESDAY') = 06-FEB-2001
- LAST_DAY('08-JAN-01') = 31-JAN-01
- SYSDATE trả về ngày giờ hiện tại của database

20

Datetime Functions

EXTRACT (datetime), TO_CHAR (datetime),TRUNC (date)

- EXTRACT({YEAR | MONTH | DAY | HOUR | MINUTE | SECOND } FROM datetime_value_expression)
  - EXTRACT(YEAR FROM DATE '1998-03-07') = 1998
  - EXTRACT(YEAR FROM sysdate)
- TO_CHAR(SYSDATE, 'fmDDTH') = '3RD'
- TO_CHAR(SYSDATE, 'YYYY') = '2013'
- TO_CHAR(SYSDATE, 'fmDAY') = 'THURSDAY'
- TRUNC(TO_DATE('27-OCT-92','DD-MON-YY'), 'YEAR') = 01-JAN-92

21

Datetime Functions
Ví dụ

```
SELECT MONTHS_BETWEEN (sysdate, hire_date)
FROM employees
WHERE empId = 100;
```

```
SELECT last_name,
                (SYSDATE-hire_date)/7 AS WEEKS
FROM employees
WHERE department_id = 90;
```

```
SELECT *
FROM employees
WHERE  EXTRACT(YEAR FROM hire_date) = 2006
```

```
SELECT  empId, firstname, TO_CHAR (hire_date, 'mm/yy')
FROM employees
```

NVL Function

- NVL chuyển đổi giá trị NULL thành một giá trị nào đó.
- Kiểu dữ liệu phải giống nhau
  - NVL(commision_pct, 0)
  - NVL(hire_date, '01-Jan-09')
  - NVL(job_title, 'No Job Yet')

Hàm gom nhóm

Hàm	Mô tả
AVG	Giá trị trung bình
COUNT	Đếm số dòng (row)
SUM	Hàm tính tổng
MAX/MIN	Giá trị lớn nhất / nhỏ nhất

Hàm gom nhóm và giá trị null

- Hàm gom nhóm sẽ bỏ qua giá trị null

```
SELECT   AVG(commision_pct)
FROM    employees;
```

- Hàm NVL giúp việc gom nhóm tính toán cả những giá trị null:

```
SELECT   AVG(NVL(commision_pct, 0))
FROM    employees;
```

Tạo các nhóm dữ liệu

- Bạn có thể phân chia các row thành những nhóm nhỏ hơn sử dụng mệnh đề group by

```
SELECT column, group_function(column)
FROM table
[WHREE condition]
[GROUP BY group_by_expression]
[ORDER BY column];
```

Ví dụ: GROUP BY

- Tất cả các cột (column) trong select (ngoại trừ những hàm tính toán) phải nằm trong Group by

```
SELECT   department_id, AVG(salary)
FROM     employees
GROUP BY    department_id;
```

Các câu truy vấn với Group by không hợp lệ

```
SELECT department_id, job_id,
COUNT(name)
FROM employees
GROUP BY department_id;
```

Either remove job_id, or Add job_id in the GROUP_BY

```
SELECT department_id, COUNT(name)
FROM employees;
```

A GROUP_BY clause must be added to count the name for each dept

Các câu truy vấn với Group by không hợp lệ

- Không thể sử dụng mệnh đề where để làm điều kiện lọc trên các nhóm

```
SELECT department_id, AVG(salary)
FROM employees
WHERE AVG(salary) > 8000
GROUP BY department_id;
```

- Sử dụng mệnh đề **HAVING**:

```
SELECT department_id, AVG(salary)
FROM employees
GROUP BY department_id
HAVING AVG(salary) > 8000
```

Truy vấn con

Truy vấn con (Nested SELECT)

```
SELECT select_list
FROM table
WHERE expr operator
                (SELECT select_list
                FROM table)
```

- Truy vấn con (inner query) được thực thi trước câu truy vấn chính (outer query)
- Kết quả của truy vấn con được sử dụng bởi câu truy vấn chính

Ví dụ: Subquery

```
SELECT last_name, salary
FROM employees
WHERE salary >
        (SELECT salary
        FROM employees
        WHERE last_name = 'King')
```

Sử dụng hàm gom nhóm trong Subquery

```
SELECT last_name, job_id, salary
FROM employees
WHERE salary =
        (SELECT MIN(salary)
        FROM employees);
```

Sử dụng hàm gom nhóm trong Subquery

- Có gì sai với câu truy vấn sau:

```
SELECT last_name, salary
FROM employees
WHERE salary =
        (SELECT MIN(salary)
        FROM employees
        GROUP BY department_id);
```

- Subquery trả về nhiều giá trị. Toán tử = chỉ so sánh trên một giá trị

Subquery trả về nhiều dòng dữ liệu

```
SELECT last_name, salary
FROM employees
WHERE salary  IN
        (SELECT MIN(salary)
        FROM employees
        GROUP BY department_id);
```

Câu lệnh INSERT

- Thêm dòng dữ liệu mới

```
INSERT INTO table
        [(column [, column…])]
VALUES (value [, value]);
```

```
INSERT INTO departments
   (department_id, department_name, manager_id, location_id)
VALUES (70, 'Sales', 100, 2000);
```

```
INSERT INTO departments
VALUES (100, 'Finance', NULL, NULL);
```

Copy row từ table khác

- Viết câu lệnh insert với một subquery

```
INSERT INTO sales_rep(id, name, salary, commission_pct)
    SELECT employee_id, last_name, salary, commission_pct
    FROM employees
    WHERE job_id LIKE '%REP%';
```

- Chú ý: không sử dụng từ khóa values

Câu lệnh Update

- Sửa đổi dữ liệu đã có trong table với câu lệnh update

```
UPDATE employees
SET department_id = 40, salary = salary + 10
WHERE employee_id = 101;
```

Câu lệnh Delete

- Xác định những dòng (row) được xóa sử dụng mệnh đề Where

```
DELETE FROM employees
WHERE department_name = 'SALES';
```

- Tất cả dữ liệu trong table sẽ bị xóa nếu không sử dụng where

```
DELETE FROM employees;
```

TRUNCATE

- Xóa tất cả dữ liệu từ table

```
TRUNCATE TABLE employees;
```

- Nó là một DDL hơn là một DML, không thể rollback.

Giao tác trong database

- Bắt đầu khi câu lệnh DML đầu tiên được thực thi.
- Kết thúc với một trong những sự kiện sau:
  - COMMIT hoặc ROLLBACK được thực thi.
  - Một câu lệnh DDL được thực thi (automatic commit)
  - Người dùng thoát khỏi SQL/PLUS
  - Hệ thống bị hỏng.

Giao tác trong database

- Những câu lệnh trong quản lý giao tác là:
  - COMMIT (Cố định thay đổi trong một transaction)
  - ROLLBACK (Hủy bỏ những thay đổi trong một transaction)
  - SAVEPOINT (Thiết lập một điểm để có thể quay lại)
  - SET TRANSACTION (Thiết lập một vài tính chất cho một giao tác)

Database Transactions
Ví dụ

```
INSERT INTO student VALUES (1,'Nguyen Van A');
SAVEPOINT A; --(*)
INSERT INTO student VALUES (2,'Nguyen Van B');
SAVEPOINT B;
INSERT INTO student VALUES (3,'Nguyen Van C');
SAVEPOINT C;
INSERT INTO student VALUES (4,'Nguyen Van D');
INSERT INTO student VALUES (5,'Nguyen Van E');
SAVEPOINT D;

COMMIT;
```

- SELECT * FROM student; -> result: 5 students
- ROLLBACK TO SAVEPOINT A; --rollback insert student 5.
- SELECT * FROM student; -> result: 4 students

Database Transactions
Ví dụ

```
SET TRANSACTION NAME 'Update salaries';

SAVEPOINT before_salary_update;

UPDATE employees SET salary=91 WHERE emp_id=1234;  -->DML statement

ROLLBACK TO SAVEPOINT before_salary_update;

UPDATE employees SET salary=92 WHERE emp_id=1234;  -->DML statement

COMMIT ;
```

Transaction Control Statements
Example

```
BEGIN
        FOR IX IN 9..12 LOOP
            IF IX = 9 THEN
               INSERT INTO NUMBERS VALUES (9);
             ELSIF IX = 11 THEN
               DELETE FROM NUMBERS;
            END IF;
            IF IX = 11 THEN
                ROLLBACK;
            ELSE
                COMMIT;
            END IF;
        END LOOP;
        COMMIT;
END;
```

NHANVIEN

MANV	HOTEN	NTNS	PHAI	MA_NQL	MaPH	LUONG
001	Vuong Ngoc Quyen	22/10/1957	Nu		QL	3.000.000
002	Nguyen Thanh Tung	09/01/1955	Nam	001	NC	2.500.000
003	Le Thi Nhan	18/12/1960	Nu	001	DH	2.500.000
004	Dinh Ba Tien	09/01/1968	Nam	002	NC	2.200.000
005	Bui Thuy Vu	19/07/1972	Nam	003	DH	2.200.000
006	Nguyen Manh Hung	15/09/1973	Nam	002	NC	2.000.000
007	Tran Thanh Tam	31/07/1975	Nu	002	NC	2.200.000
008	Tran Hong Minh	04/07/1976	Nu	004	NC	1.800.000

DEAN

MADA	TENDA	PHONG	NamThucHien
TH001	Tin hoc hoa 1	NC	2002
TH002	Tin hoc hoa 2	NC	2003
DT001	Dao tao 1	DH	2004
DT002	Dao tao 2	DH	2004

PHONGBAN

MAPH	TENPH	TRPH
QL	Quan Ly	001
DH	Dieu Hanh	003
NC	Nghien Cuu	002

PHANCONG

MANV	MADA	THOIGIAN
001	TH001	30,0
001	TH002	12,5
002	TH001	10,0
002	TH002	10,0
002	DT001	10,0
002	DT002	10,0
003	TH001	37,5
004	DT001	22,5
004	DT002	10,0
006	DT001	30,5
007	TH001	20,0
007	TH002	10,0
008	DT002	12,5