CATEGORIES

DML = พวกเปลี่ยนแปลง แก้ไข INSERT , UPDATE , DELETE , MERGE

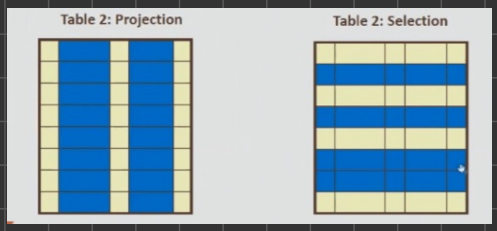
DDL = CREATE , ALTER , DROP , RENAME , TRUNCATE

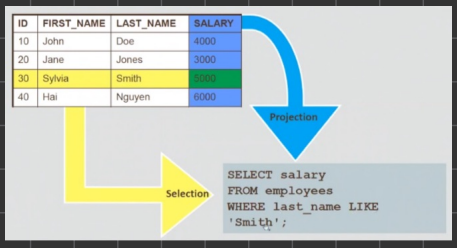
TCL = จัดการการเปลี่ยนแปลง COMMIT เซฟงานลง SERVER → คล้าย DELETE แต่อันนี้ลบถาวร คนอื่นมองเห็นได้

ROLLBACK ย้อนกลับ SAVEPOINT

DCL = GRANT ให้สิทธิคนอื่น SELECT REVOKE ยึดสิทธิคืน

SELECT STATEMENT





NULL = ไม่มีค่า ≠0 แล้วแต่สถานการณ์ เจอตัวนี้แปลว่าข้อมูลไม่สมบูรณ์ NULL + - x ÷ ก็ได้ NULL

ALIASES เปลี่ยนชื่อหัวคอลัมน์

```
SELECT * |column|expr [ AS alias], .....
FROM      table;
```

• Examples:

```
SELECT last_name AS name,
       commission_pct AS comm
FROM employees;
```

NAME	COMM
King	-
Kochhar	-
De Haan	-

```
SELECT last_name "Name",
       salary*12 "Annual Salary"
FROM employees;
```

Name	Annual Salary
King	288000
Kochhar	204000
De Haan	204000

DOUBLE CODE

```
SELECT department_name, location_id
FROM departments
WHERE location_id NOT IN (1700,1800);
```

DEPARTMENT_NAME	LOCATION_ID
Shipping	1500
IT	1400
Sales	2500

DESCRIBE ดูโครงสร้างตาราง DESC <table name>; มาก→น้อย

AND เป็นจริงทั้งคู่ = จริง OR อันนึงเป็นจริง = จริง NOT ทำให้ค่าค.จริงเป็นนิเสธ

ORDER	OPERATOR
1	Arithmetic + - * /
2	Concatenation||
3	Comparison <, <=, >, >=, <>
4	IS (NOT) NULL, LIKE, (NOT) IN
5	(NOT) BETWEEN
6	NOT
7	AND
8	OR

ทำอันไหนก่อนหลัง

① From มาจากตารางไหน
② Where ตัด row ที่ไม่ตรงกับเงื่อนไข
③ SELECT
④ ORDER BY

DUAL TABLE สร้างตารางให้แค่เห็น

```
SELECT (319/29) + 12
FROM DUAL;
```

(319/29)+12
23

UPPER

- UPPER(column | expression) converts alpha characters to upper-case

```
SELECT last_name
FROM empl    oyees
WHERE UPPER(last_name) = 'ABEL';
```

LOWER

- INITCAP(column | expression) converts alpha character values to uppercase for the first letter of each word

```
SELECT last_name
FROM employees
WHERE INITCAP(last_name) = 'Abel';
```

CONCAT

- CONCAT: Joins two values together
- Takes 2 character string arguments, and joins the second string to the first. could also be written using the concatenation operator - 'Hello' || 'World'

Examples:	Result
SELECT CONCAT('Hello', 'World') FROM DUAL;	HelloWorld
SELECT CONCAT(first_name, last_name) FROM employees;	EllenAbel CurtisDavies ...

SUBSTR

Examples:	Result
SELECT SUBSTR('HelloWorld', 1, 5) FROM DUAL;	Hello
SELECT SUBSTR('HelloWorld', 6) FROM DUAL;	World
SELECT SUBSTR(last_name, 1, 3) FROM employees;	Abe Dav

INSTR หาตำแหน่งคำ

Examples:	Result
SELECT INSTR('HelloWorld', 'W') FROM DUAL;	6
SELECT last_name, INSTR(last_name, 'a') FROM employees;	Abel 0 Davies 2 ...

LPAD ใส่ด้านซ้ายจนกว่าค.ยาวจะครบ

Examples:	Result
SELECT LPAD('HelloWorld', 15, '-') FROM DUAL;	-----HelloWorld
SELECT LPAD(last_name, 10, '*') FROM employees;	******Abel ****Davies ...

RPAD ด้านขวา

Examples:	Result
SELECT RPAD('HelloWorld', 15, '-') FROM DUAL;	HelloWorld-----
SELECT RPAD(last_name, 10, '*') FROM employees;	Abel****** Davies**** ...

TRIM ตัดหัว-ท้าย

Examples:	Result
SELECT TRIM(LEADING 'a' FROM 'abcba') FROM DUAL;	bcba
SELECT TRIM(TRAILING 'a' FROM 'abcba') FROM DUAL;	abcb
SELECT TRIM(BOTH 'a' FROM 'abcba') FROM DUAL;	bcb

REPLACE

```
REPLACE (string1, string_to_replace, [replacement_string] )
```

- string1 is the string that will have characters replaced in it
- string_to_replace is the string that will be searched for and taken out of string1
- [replacement_string] is the new string to be inserted in string1

Examples:	Result
SELECT REPLACE('JACK and JUE', 'J', 'BL') FROM DUAL;	BLACK and BLUE
SELECT REPLACE('JACK and JUE', 'J') FROM DUAL;	ACK and UE
SELECT REPLACE(last_name, 'a', '*') FROM employees;	Abel D*vies De H**n

+ Ex

มี POP-UP ถ้าใส่ :

- If this was the original query:

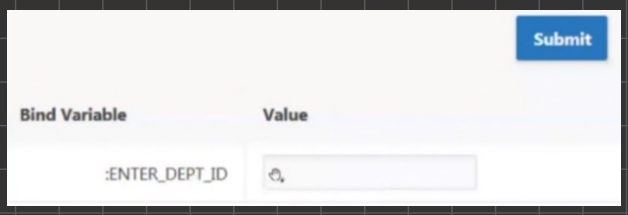
```
SELECT first_name, last_name, salary, department_id
FROM employees
WHERE department_id= 10;
```

  - Then run it again with different values: 20, 30, 40... etc.
- It could be re-written as:

```
SELECT first_name, last_name, salary, department_id
FROM employees
WHERE department_id=:enter_dept_id;
```

- Note the use of : in front of enter_dept_id

Ex ๒



NUMBER

• ROUND ปัดเศษขึ้น - ลง

- Syntax:

```
ROUND(column|expression, decimal places)
```

- ROUND(45.926) 46
- ROUND(45.926, 0) 46

- ROUND(45.926, 2) 45.93
- If the number of decimal places is a negative number, the number is rounded to that number of decimal places to the left of the decimal point
- ROUND(45.926, -1) 50

• TRUNC ปัดเศษลงอย่างเดียว

- Syntax:

```
TRUNC(column|expression, decimal places)
```

- TRUNC (45.926, 2) 45.92

• MOD หารเอาเศษ

MOD of 5 divided by 2 is 1

• SYSDATE วันที่ปัจจุบัน

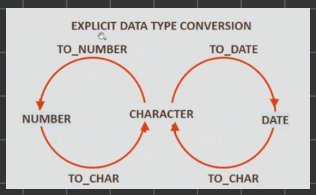
```
SELECT SYSDATE
FROM dual;
```

SYSDATE
01-Jul-2017

Function	Description
MONTHS_BETWEEN	Number of months between two dates
ADD_MONTHS	Add calendar months to date
NEXT_DAY	Date of the next occurance of day of the week specified
LAST_DAY	Last day of the month
ROUND	Round date
TRUNC	Truncate date

Implicit data type conversions

FROM	TO
VARCHAR2 or CHAR	NUMBER
VARCHAR2 or CHAR	DATE
NUMBER	VARCHAR2
DATE	VARCHAR2



TO_CHAR DD-Mon-YYYY

```
TO_CHAR (date column name, 'format model you specify')
```

YYYY	Full year in numbers
YEAR	Year spelled out
MM	Two-digit value for month
MONTH	Full name of the month
MON	Three-letter abbreviation of the month
DY	Three-letter abbreviation of the day of the week
DAY	Full name of the day of the week
DD	Numeric day of the month
DDspth	FOURTEENTH
Ddspth	Fourteenth
ddspth	fourteenth
DDD or DD or D	Day of year, month or week
HH24:MI:SS AM	15:45:32 PM
DD "of" MONTH	12 of October

- The tables show the different format models that can be used
- When specifying time elements, note that hours (HH), minutes (MI), seconds (SS), and AM or PM can also be formatted

Examples:	Output
SELECT TO_CHAR(hire_date, 'Month dd, YYYY') FROM employees;	... June 07, 1994 ...
SELECT TO_CHAR(hire_date, 'fmMonth dd, YYYY') FROM employees;	... June 7, 1994 ...
SELECT TO_CHAR(hire_date, 'fmMonth ddth, YYYY') FROM employees;	June 7th, 1994 January 3rd, 1990 ...

Examples:	Output
SELECT TO_CHAR(hire_date, 'fmDay ddth Mon, YYYY') FROM employees;	Tuesday 7th Jun, 1994
SELECT TO_CHAR(hire_date, 'fmDay ddthsp Mon, YYYY') FROM employees;	Tuesday, seventh Jun, 1994
SELECT TO_CHAR(hire_date, 'fmDay, ddthsp "of" Month, Year') FROM employees;	Tuesday, seventh of June, Nineteen Ninety-Four

Examples:	Output
SELECT TO_CHAR(SYSDATE, 'hh:mm') FROM dual;	02:07
SELECT TO_CHAR(SYSDATE, 'hh:mm pm') FROM dual;	02:07 am
SELECT TO_CHAR(SYSDATE, 'hh:mm:ss pm') FROM dual;	02:07:23 am

number

```
TO_CHAR(number, 'format model')
```

- The table illustrates some of the format elements available to use with TO_CHAR functions

```
SELECT TO_CHAR(salary,
  '$99,999') AS "Salary"
FROM employees;
```

Salary
$24,000
$17,000

ELEMENT	DESCRIPTION	EXAMPLE	RESULT
9	Numeric position (# of 9's determine width)	999999	1234
0	Display leading zeros	099999	001234
$	Floating dollar sign	$999999	$1234
L	Floating local currency symbol	L999999	FF1234
.	Decimal point in position specified	999999.99	1234.00
,	Comma in position specified	999,999	1,234
MI	Minus signs to right (negative values)	999999MI	1234-
PR	Parenthesize negative numbers	999999PR	<1234>
EEEE	Scientific notation (must have four EEEE)	99.999EEEE	1.23E+03
V	Multiply by 10 n times (n= number of 9's after V)	9999V99	9999v99
B	Display zero values as blank, not 0	B9999.99	1234.00

SQL:	Output
SELECT TO_CHAR(3000, '$99999.99') FROM dual;	$3000.00
SELECT TO_CHAR(4500, '99,999') FROM dual;	4,500
SELECT TO_CHAR(9000, '99,999.99') FROM dual;	9,000.00
SELECT TO_CHAR(4422, '0009999') FROM dual;	0004422

TO_NUMBER

```
TO_NUMBER(character string, 'format model')
```

- The format model is optional, but should be included if the character string being converted contains any characters other than numbers
- You cannot reliably perform calculations with character data

```
SELECT TO_NUMBER('5,320', '9,999')
AS "Number"
FROM dual;
```

Number
5320

TO_DATE

```
SELECT TO_DATE('May10,1989', 'fxMonDD,YYYY') AS "Convert"
FROM DUAL;
```

ORACLE

CONVERT
10-May-1989

CASE Expression ทำงานเองไม่ได้ ต้องอยู่ใน cluse ทำงาน ลักษณะ IF-THEN-ELSE

```
CASE expr WHEN comparison_expr1 THEN return_expr1
          [WHEN comparison_expr2 THEN return_expr2
           WHEN comparison_exprn THEN return_exprn
           ELSE else_expr]
END
```

ex.

```
SELECT last_name,
CASE department_id
   WHEN 90 THEN 'Management'
   WHEN 80 THEN 'Sales'
   WHEN 60 THEN 'It'
   ELSE 'Other dept.'
END AS "Department"
FROM employees;
```

DECODE เป็นฟังก์ชัน

```
DECODE(column|expression, search1, result1
       [, search2, result2,...,]
       [, default])
```

ex.

```
SELECT last_name,
  DECODE(department_id,
    90, 'Management',
     80, 'Sales',
    60, 'It',
    'Other dept.')
AS "Department"
FROM employees;
```

ใช้ร่วม WHERE CLUSE จะช้ามาก

NATURAL JOIN

เอาคอลัมต์ที่ชื่อเหมือนกันมาเทียบเท่ากัน

```
SELECT first_name, last_name, job_id, job_title
FROM employees NATURAL JOIN jobs
WHERE department_id > 80;
```

FIRST_NAME	LAST_NAME	JOB_ID	JOB_TITLE
Steven	King	AD_PRES	President
Neena	Kochhar	AD_VP	Administration Vice President
Lex	De Haan	AD_VP	Administration Vice President
Shelley	Higgins	AC_MGR	Accounting Manager
William	Gietz	AC_ACCOUNT	Public Accountant

ex. e.first_name ห้ามใช้ Qualified

```
SELECT first_name, last_name, department_id, department_name
FROM employees JOIN departments USING (department_id);
```

FIRST_NAME	LAST_NAME	DEPARTMENT_ID	DEPARTMENT_NAME
Jennifer	Whalen	10	Administration
Michael	Hartstein	20	Marketing
Pat	Fay	20	Marketing
...	...	...	...

USING จะใช้คอลัมต์ไหน JOIN

Cross Join เอาคูณกัน ex. 8 x 20 = 160

```
SELECT last_name, department_name
FROM employees CROSS JOIN
departments;
```

ON CLUSE เห็นชัดอะไร JOIN อะไร ด้วยเงื่อนไขอะไร

- In this example, the ON clause is used to join the employees table with the jobs table

```
SELECT last_name, job_title
FROM employees e JOIN jobs j
  ON (e.job_id = j.job_id);
```

- A join ON clause is required when the common columns have different names in the two tables

LAST_NAME	JOB_TITLE
King	President
Kochhar	Administration Vice President
De Haan	Administration Vice President
Whalen	Administration Assistant
Higgins	Accounting Manager
Gietz	Public Accountant
Zlotkey	Sales Manager
Abel	Sales Representative
Taylor	Sales Representative
...	

ใช้ BETWEEN กับ AND

```
SELECT last_name, salary, grade_level, lowest_sal,
highest_sal
FROM employees JOIN job_grades
  ON(salary BETWEEN lowest_sal AND highest_sal);
```

LAST_NAME	SALARY	GRADE_LEVEL	LOWEST_SAL	HIGHEST_SAL
Vargas	2500	A	1000	2999
Matos	2600	A	1000	2999
Davies	3100	B	3000	5999
Rajs	3500	B	3000	5999
Lorentz	4200	B	3000	5999
Whalen	4400	B	3000	5999
Mourgos	5800	B	3000	5999
Fay	6000	C	6000	9999
...				

Hierarchical

```
SELECT employee_id, last_name, job_id, manager_id
FROM employees
START  WITH  employee_id = 100
CONNECT BY PRIOR employee_id = manager_id
```

EMPLOYEE_ID	LAST_NAME	JOB_ID	MANAGER_ID
100	King	AD_PRES	-
101	Kochhar	AD_VP	100
200	Whalen	AD_ASST	101
205	Higgins	AC_MGR	101
206	Gietz	AC_ACCOUNT	205
102	De Haan	AD_VP	100
103	Hunold	IT_PROG	102
104	Ernst	IT_PROG	103

```
SELECT table1.column, table2.column
FROM table1, table2
WHERE table1.column1 = table2.column2;
```

ไม่รวม null
Group ใช้ใน WHERE ไม่ได้

COUNT(*) นับทุก record

Group by อยากให้ใช้ Group กับอะไร
ให้ดิ้วตาราง ที่จะโชว์มาใส่ Group by ทั้งหมด

Having เลือกว่าจะแสดง group ไหน ออกมาเป็นผลลัพธ์
where เลือกเพื่อประมวลผล
ใช้ group ได้
function