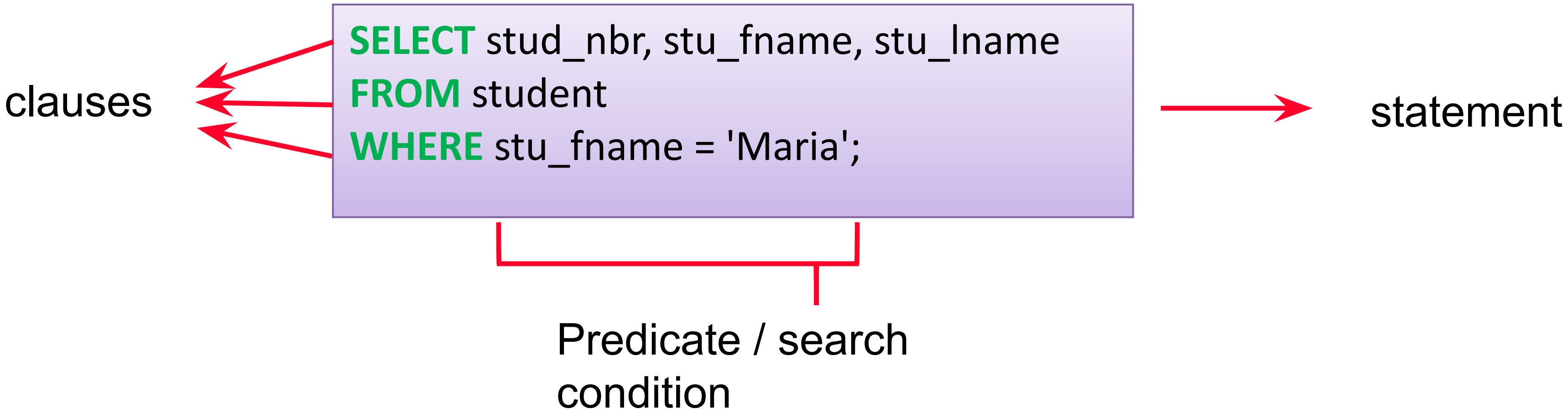
MONASH University

MONASH
INFORMATION
TECHNOLOGY

Structured Query Language (SQL) – Part 1

Anatomy of an SQL SELECT Statement



```sql
SELECT stud_nbr, stu_fname, stu_lname
FROM student
WHERE stu_fname = 'Maria';
```

SQL Predicates or Search Conditions

- The search conditions are applied on each row, and the row is returned if the search conditions are evaluated to be TRUE for that row.
- **Comparison**
  - Compare the value of one expression to the value of another expression.
  - Operators: =, !=,< >, <, >, <=, >=
  - Example: salary > 5000
- **Range**
  - Test whether the value of an expression falls within a specified range of values.
  - Operator: BETWEEN
  - Example: salary BETWEEN 1000 AND 3000 (both are inclusive)

SQL SELECT Statement - Usage

What column/s to display

What table(s) the data come from?

```sql
SELECT stud_nbr, stu_fname, stu_lname
FROM student
WHERE stu_fname = 'Maria';
```

What row/s to retrieve – the RESTRICTION on the select

SQL Predicates or Search Conditions

- **Set Membership**
  - To test whether the value of expression equals one of a set of values.
  - Operator: IN
  - Example : city IN ('Melbourne', 'Sydney')
- **Pattern Match**
  - To test whether a string (text) matches a specified pattern.
  - Operator: LIKE
  - Patterns:
    - % character represents any sequence of zero or more character.
    - _ character represents any single character.
  - Example:
    - WHERE city LIKE 'M%'
    - WHERE unit_code LIKE 'FIT20__'

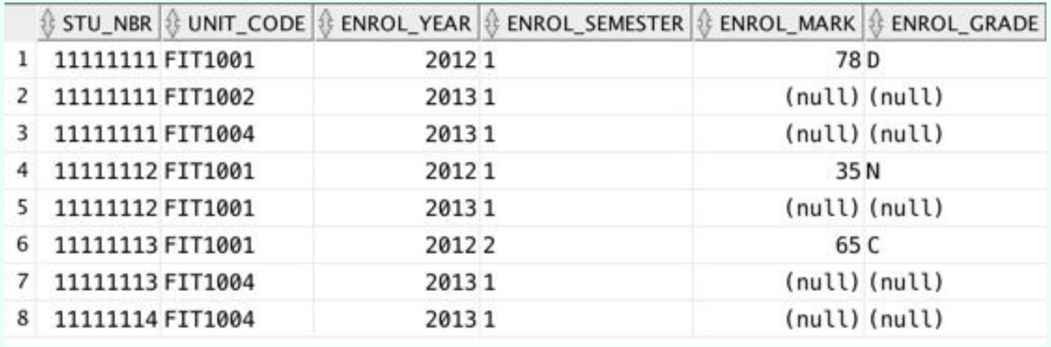
SQL Predicates or Search Conditions

- **NULL**
  - To test whether a column has a NULL (unknown) value.
  - Example: WHERE grade IS NULL.
- Use in subquery (to be discussed in the future)
  - ANY, ALL
  - EXISTS

What row will be retrieved?

- Predicate evaluation is done using three-valued logic.
  - **TRUE**, **FALSE** and **UNKNOWN**
- DBMS will evaluate the predicate against each row.
- Row that is evaluated to be **TRUE** will be retrieved.
- NULL is considered to be UNKNOWN.

Q1. Consider the predicate "enrol_mark >= 50", what row(s) will be selected for this predicate by the DBMS?



	STU_NBR	UNIT_CODE	ENROL_YEAR	ENROL_SEMESTER	ENROL_MARK	ENROL_GRADE
1	11111111	FIT1001	2012	1	78	D
2	11111111	FIT1002	2013	1	(null)	(null)
3	11111111	FIT1004	2013	1	(null)	(null)
4	11111112	FIT1001	2012	1	35	N
5	11111112	FIT1001	2013	1	(null)	(null)
6	11111113	FIT1001	2012	2	65	C
7	11111113	FIT1004	2013	1	(null)	(null)
8	11111114	FIT1004	2013	1	(null)	(null)

a. 1, 4 and 6

b. All rows

c. 1 and 6

d. All rows except row 4

Combining Predicates

- Logical operators
  - AND, OR, NOT
- Rules:
  - An expression is evaluated LEFT to RIGHT
  - Sub-expression in brackets are evaluated first
  - NOTs are evaluated before AND and OR
  - ANDs are evaluated before OR
  - Use of BRACKETS better alternative

Truth Table

- **AND** is evaluated to be TRUE if and only if **both** conditions are TRUE
- **OR** is evaluated to be TRUE if and only if at least one of the conditions is TRUE

AND

A \ B	T	U	F
T	T	U	F
U	U	U	F
F	F	F	F

OR

A \ B	T	U	F
T	T	T	T
U	T	U	U
F	T	U	F

T = TRUE
F = FALSE
U = Unknown

Unknown = NULL in relational database

Q2. What row will be retrieved when the WHERE clause predicate is written as

V_CODE = 21344 AND V_CODE = 24288 ?

	V_CODE
1	21344
2	20001
3	24288
4	20001
5	24288

a. 1,3,5

b. 1

c. 3,5

d. No rows will be retrieved

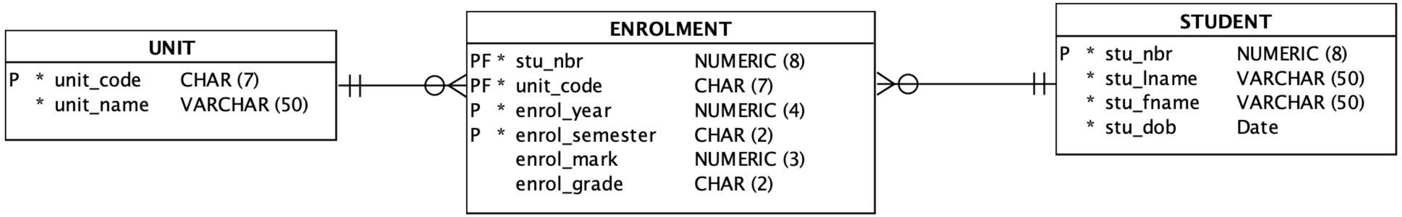
MONASH University

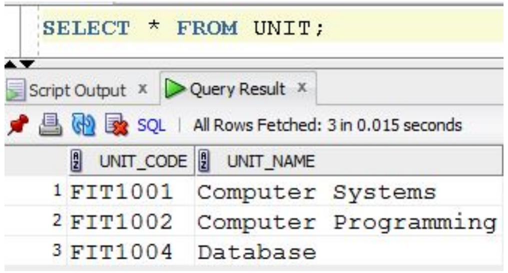
Q3. What row will be retrieved when the WHERE clause predicate is written as

V_CODE <> 21344 OR V_CODE <> 24288 ?

	V_CODE
1	21344
2	20001
3	24288
4	20001
5	24288

a. 1,3,5
b. 2,4
c. 3,5
d. 1,2,3,4,5

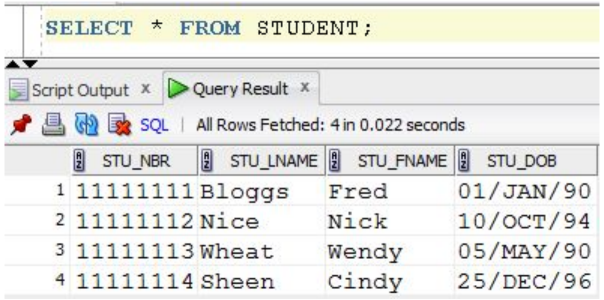




```sql
SELECT * FROM UNIT;
```

All Rows Fetched: 3 in 0.015 seconds

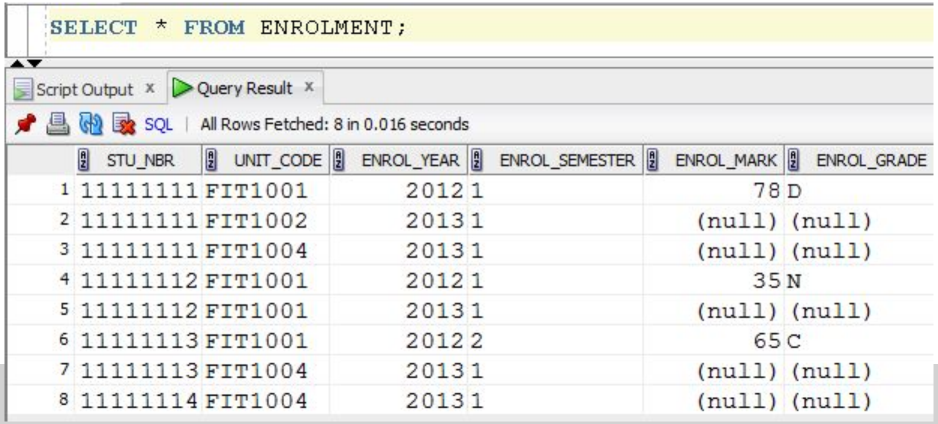
	UNIT_CODE	UNIT_NAME
1	FIT1001	Computer Systems
2	FIT1002	Computer Programming
3	FIT1004	Database



```sql
SELECT * FROM STUDENT;
```

All Rows Fetched: 4 in 0.022 seconds

	STU_NBR	STU_LNAME	STU_FNAME	STU_DOB
1	11111111	Bloggs	Fred	01/JAN/90
2	11111112	Nice	Nick	10/OCT/94
3	11111113	Wheat	Wendy	05/MAY/90
4	11111114	Sheen	Cindy	25/DEC/96

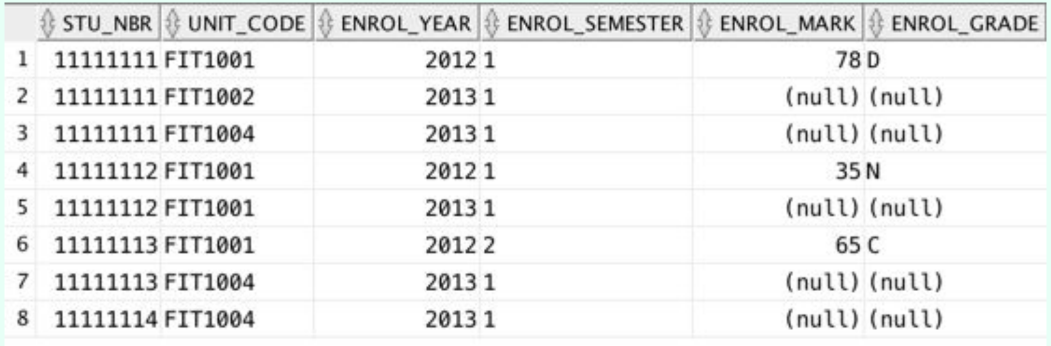


```sql
SELECT * FROM ENROLMENT;
```

All Rows Fetched: 8 in 0.016 seconds

	STU_NBR	UNIT_CODE	ENROL_YEAR	ENROL_SEMESTER	ENROL_MARK	ENROL_GRADE
1	11111111	FIT1001	2012	1	78	D
2	11111111	FIT1002	2013	1	(null)	(null)
3	11111111	FIT1004	2013	1	(null)	(null)
4	11111112	FIT1001	2012	1	35	N
5	11111112	FIT1001	2013	1	(null)	(null)
6	11111113	FIT1001	2012	2	65	C
7	11111113	FIT1004	2013	1	(null)	(null)
8	11111114	FIT1004	2013	1	(null)	(null)

Q4. What is the correct SQL predicate to retrieve those students who have passed and also those students who have not been awarded any mark?



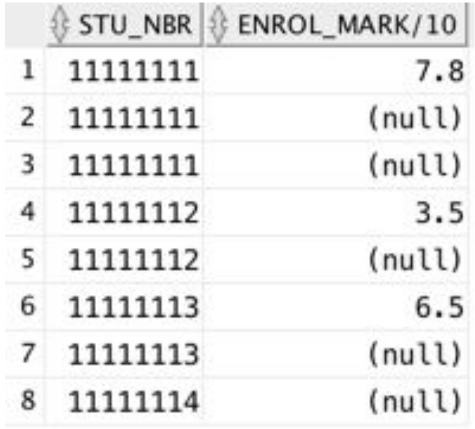
	STU_NBR	UNIT_CODE	ENROL_YEAR	ENROL_SEMESTER	ENROL_MARK	ENROL_GRADE
1	11111111	FIT1001	2012	1	78	D
2	11111111	FIT1002	2013	1	(null)	(null)
3	11111111	FIT1004	2013	1	(null)	(null)
4	11111112	FIT1001	2012	1	35	N
5	11111112	FIT1001	2013	1	(null)	(null)
6	11111113	FIT1001	2012	2	65	C
7	11111113	FIT1004	2013	1	(null)	(null)
8	11111114	FIT1004	2013	1	(null)	(null)

a. enrol_mark >= 50 AND enrol_mark IS NULL
b. enrol_mark >= 50 OR enrol_mark IS NULL
c. enrol_mark >= 50 AND enrol_mark IS NOT NULL
d. enrol_ mark >= 50 OR enrol_mark IS NOT NULL
e. None of the above

Arithmetic Operations

- Can be performed in SQL.
- For example:

```
SELECT stu_nbr, enrol_mark/10
FROM enrolment;
```

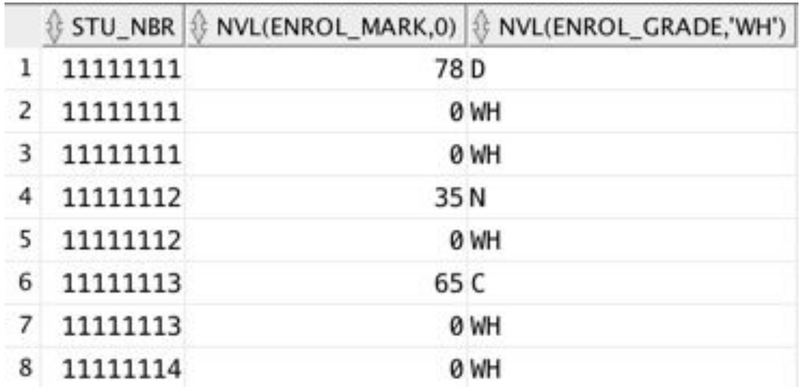


	STU_NBR	ENROL_MARK/10
1	11111111	7.8
2	11111111	(null)
3	11111111	(null)
4	11111112	3.5
5	11111112	(null)
6	11111113	6.5
7	11111113	(null)
8	11111114	(null)

Oracle NVL function

- It is used to replace a NULL with a value.

```
SELECT stu_nbr,
   NVL(enrol_mark,0),
   NVL(enrol_grade,'WH')
FROM enrolment;
```



	STU_NBR	NVL(ENROL_MARK,0)	NVL(ENROL_GRADE,'WH')
1	11111111	78	D
2	11111111	0	WH
3	11111111	0	WH
4	11111112	35	N
5	11111112	0	WH
6	11111113	65	C
7	11111113	0	WH
8	11111114	0	WH

MONASH University

Renaming Column

- Note column headings on slide 16
- Use the word "AS"
  - New column name in " " to maintain case or spacing
- Example

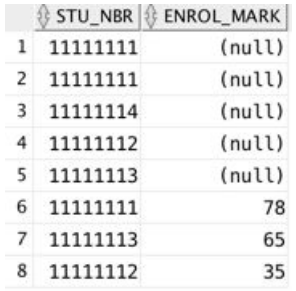
```
SELECT stu_nbr, enrol_mark/10 AS new_mark
FROM enrolment;

SELECT stu_nbr, enrol_mark/10 AS "New Mark"
FROM enrolment;
```

Sorting Query Result

- "ORDER BY" clause – *tuples have no order*
  - Must be used if more than one row may be returned
- Order can be ASCending or DESCending. The default is ASCending.
  - NULL values can be explicitly placed first/last using "NULLS LAST" or "NULLS FIRST" command
- Sorting can be done for multiple columns.
  - order of the sorting is specified for each column.
- Example:

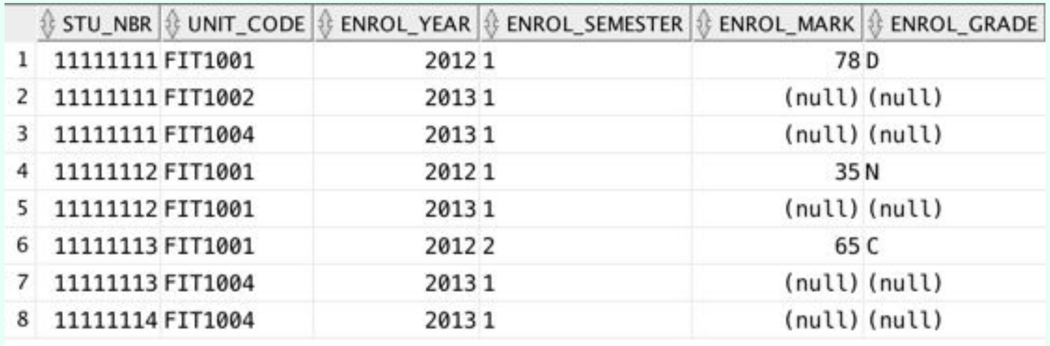
```
SELECT stu_nbr, enrol_mark
FROM enrolment
ORDER BY enrol_mark DESC
```



	STU_NBR	ENROL_MARK
1	11111111	(null)
2	11111111	(null)
3	11111114	(null)
4	11111112	(null)
5	11111113	(null)
6	11111111	78
7	11111113	65
8	11111112	35

Q5. What will be the output of the following SQL statement?

```
SELECT stu_nbr
FROM enrolment
WHERE enrol_mark IS NULL;
```



	STU_NBR	UNIT_CODE	ENROL_YEAR	ENROL_SEMESTER	ENROL_MARK	ENROL_GRADE
1	11111111	FIT1001	2012	1	78	D
2	11111111	FIT1002	2013	1	(null)	(null)
3	11111111	FIT1004	2013	1	(null)	(null)
4	11111112	FIT1001	2012	1	35	N
5	11111112	FIT1001	2013	1	(null)	(null)
6	11111113	FIT1001	2012	2	65	C
7	11111113	FIT1004	2013	1	(null)	(null)
8	11111114	FIT1004	2013	1	(null)	(null)

a.

11111111
11111112
11111113
11111114

b.

11111111
11111111
11111112
11111113
11111114

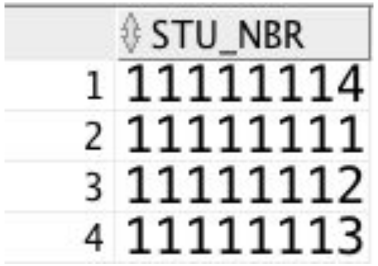
c.

11111111
11111112
11111113

Removing Duplicate Rows in the Query Result

- Use "DISTINCT" as part of SELECT clause
  - *use with care*

```
SELECT DISTINCT stu_nbr
FROM enrolment
WHERE enrol_mark IS NULL;
```



	STU_NBR
1	11111114
2	11111111
3	11111112
4	11111113

SQL NATURAL JOIN

STUDENT

sno	name
1	alex
2	maria
3	bob

QUALIFICATION

sno	degree	year
1	bachelor	1990
1	master	2000
2	PhD	2001

```
SELECT *
FROM student JOIN qualification
        ON student.sno = qualification.sno
ORDER BY student.sno
```

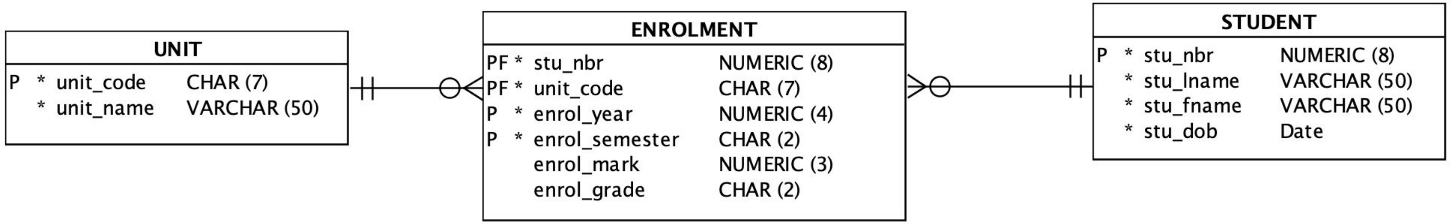
sno	name	degree	year
1	alex	bachelor	1990
1	alex	master	2000
2	maria	PhD	2001

SQL JOIN

- For database students are **required to use ANSI JOINS**
  - placing the join in the where clause is not acceptable and will be *marked as incorrect for all assessment purposes*
    - such a join is sometimes known as "implicit join notation" - effectively a cross product and then restricted by the where clause
- ANSI JOINS
  - ON
    - the general form which always works, hence the syntax we tend to use
    - FROM student JOIN qualification ON student.sno = qualification.sno
  - USING
    - requires matching attribute names for the PK and FK
    - FROM student JOIN qualification USING (sno)
  - NATURAL
    - requires matching attribute names for the PK and FK
    - FROM student NATURAL JOIN qualification

JOIN-ing Multiple Tables

Pair the PK and FK in the JOIN condition
Note table aliasing e.g. unit u in FROM clause



```
SELECT s.stu_nbr, s.stu_lname, u.unit_name
FROM ((unit u JOIN enrolment e ON u.unit_code=e.unit_code)
          JOIN student s ON e.stu_nbr=s.stu_nbr)
ORDER BY s.stu_nbr, u.unit_name;
```

MONASH University

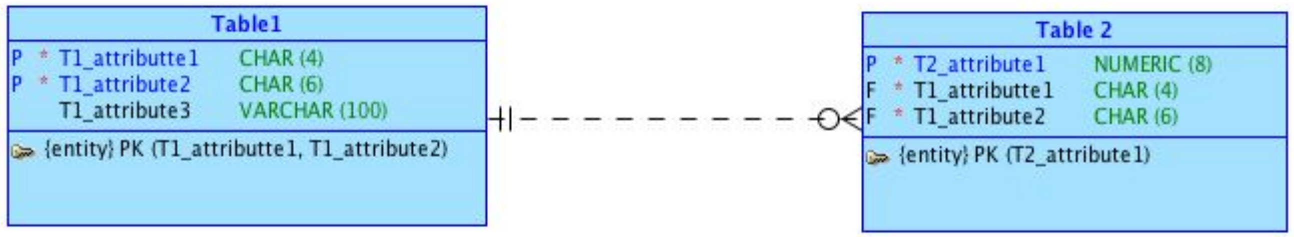


Table1
P * T1_attributte1 CHAR (4)
P * T1_attribute2 CHAR (6)
T1_attribute3 VARCHAR (100)
{entity} PK (T1_attributte1, T1_attribute2)
Table 2
P * T2_attribute1 NUMERIC (8)
F * T1_attributte1 CHAR (4)
F * T1_attribute2 CHAR (6)
{entity} PK (T2_attribute1)

How many conditions will be used to join the two tables?

```
SELECT *
FROM table1 t1 JOIN table2 t2 ON
       (t1.T1_attribute1 = t2.T1_attribute1
         AND
         t1.T1_attribute2 = t2.T1_attribute2)
ORDER BY t1.T1_attribute1, t1.T1_attribute2;
```

Oracle Date Data Type

Oracle Data Datatype

- Dates are stored differently from the SQL standard
  - standard uses two different types: date and time
  - Oracle uses one type: DATE
    - Stored in internal format contains date and time
      - Julian date as number (can use arithmetic)
    - Output is controlled by formatting
      - select **to_char**(sysdate,'dd-Mon-yyyy') from dual;
        - » 04-May-2020
      - select **to_char**(sysdate,'dd-Mon-yyyy hh:mi:ss PM') from dual;
        - » 04-May-2020 02:51:24 PM

- DATE data type should be formatted with **TO_CHAR** when selecting for **display**.
- Text representing date **must be formatted** with **TO_DATE** when **comparing** or **inserting/updating.**
- Example:

```
select studid,
   studfname || ' ' || studlname as StudentName,
   to_char(studdob,'dd-Mon-yyyy') as StudentDOB
from uni.student
where studdob >  to_date('01-Apr-1991','dd-Mon-yyyy')
order by studdob;
```

Current Date

- Current date can be queried from the DUAL table using the **SYSDATE** attribute.
  - SELECT **sysdate** FROM dual;
- Oracle internal attributes include:
  - **sysdate**: current date/time
  - **systimestamp**: current date/time as a timestamp
  - **user**: current logged in user