MONASH University

MONASH
INFORMATION
TECHNOLOGY

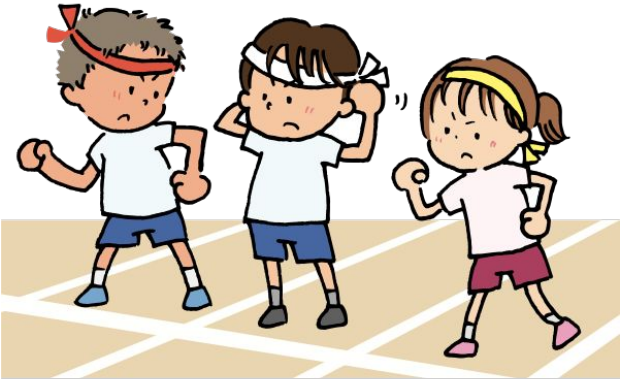
Topic 8
Structured Query Language (SQL) – Part 1

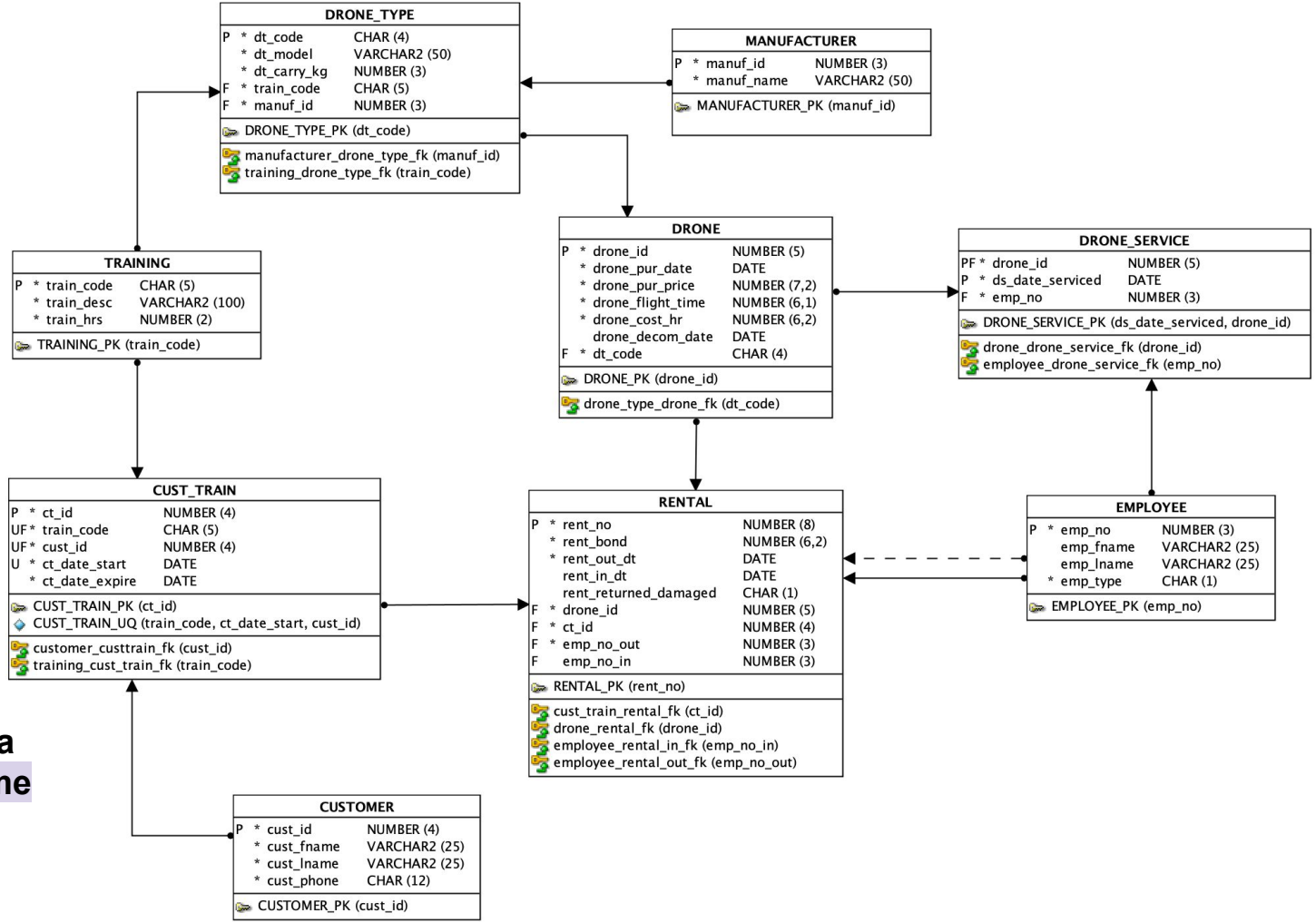
Workshop 2025 S1

Preparation for the workshop - ready, set ……

Please

- connect to Poll Everywhere and be ready to answer questions
- login to the Oracle database via Visual Studio Code or ORDS: https://ora-fit.ocio.monash.edu:8441/ords/sql-developer





Access tables via DRONE.tablename in the Monash Oracle database

Anatomy of an SQL SELECT Statement

clauses

```
SELECT drone_id, drone_pur_date, drone_flight_time
FROM drone.drone
WHERE drone_pur_price > 2000;
```

statement

Predicate / search condition

Worksheet | Query Builder

```
1  SELECT
2      drone_id,
3      drone_pur_date,
4      drone_flight_time
5  FROM
6      drone.drone
7  WHERE
8      drone_pur_price > 2000;
```

Query Result

SQL | All Rows Fetched: 7 in 0.006 seconds

	DRONE_ID	DRONE_PUR_DATE	DRONE_FLIGHT_TIME
1	103	13/JAN/21	200
2	111	20/MAR/21	100
3	112	20/MAR/21	40
4	113	20/MAR/21	150
5	117	20/MAR/21	100.5
6	119	01/APR/22	10.2
7	120	01/APR/22	25.8

SQL SELECT Statement - Usage

What column/s to display

What table(s) the data is to be drawn from

```
SELECT drone_id, drone_pur_date, drone_flight_time
FROM drone.drone
WHERE drone_pur_price > 2000;
```

What row/s to retrieve – the RESTRICTION to place on the rows retrieved

Run this command against the Oracle Database

Q1. List all the drones which cost from $3000 to $5300 to purchase (multiple answers may be selected):

A. SELECT * FROM drone.drone where drone_pur_price BETWEEN 3000 AND 5300;
B. SELECT * FROM drone.drone where drone_pur_price >= 3000 or drone_pur_price <= 5300;
C. SELECT * FROM drone.drone where drone_pur_price IN (3000,5300);
D. SELECT * FROM drone.drone where drone_pur_price >= 3000 and drone_pur_price <= 5300;
E. SELECT * FROM drone.drone where drone_pur_price >= 3000 or <= 5300;

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: drone_pur_price > 2000
- **Range**
  - Test whether the value of an expression falls within a specified range of values.
  - Operator: BETWEEN
  - Example: drone_pur_price BETWEEN 3000 AND 5300 (both are inclusive)

SQL Predicates or Search Conditions

- **Set Membership**
  - To test whether the value of expression equals one of a set of values.
  - Operator: IN
  - Example : dt_code in ('DMA2','DSPA') -> which drones of this type?
- **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 dt_model LIKE 'DJI%' -> drone type models starting with DJI
    - WHERE train_code LIKE '__I__' -> drone types with a train_code with an I in the middle

Q2. To list the rentals which have not been returned, the SQL would be:

A. select * from drone.rental where rent_in_dt = null;
B. select * from drone.rental where rent_in_dt is null;
C. select * from drone.rental where rent_in_dt is not null;
D. select * from drone.rental where rent_in_dt is empty;

SQL Predicates or Search Conditions

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

Worksheet | Query Builder

```sql
SELECT
    *
FROM
    drone.rental
WHERE
    rent_in_dt IS NULL;
```

Query Result

SQL | All Rows Fetched: 3 in 0.01 seconds

	RENT_NO	RENT_BOND	RENT_OUT_DT	RENT_IN_DT	RENT_RETURNED_DAMAG
1	23	250	21/AUG/22	(null)	(null)
2	24	150	22/AUG/22	(null)	(null)
3	25	180	23/AUG/22	(null)	(null)

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.

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

Q3. Find all the services which have not been carried out by the employee with emp_no 3 or the employee with emp_no 8:

	DRONE_ID	DS_DATE_SERVICED	EMP_NO
1	100	21/FEB/21	3
2	100	26/FEB/21	8
3	101	19/MAR/21	3
4	101	22/FEB/21	3
5	101	26/FEB/21	8
6	102	24/FEB/21	3
7	102	29/MAR/21	8
8	103	05/MAR/21	8
9	103	11/MAR/21	3
10	103	03/MAY/21	2
11	103	11/MAY/21	2

Only partial data shown

A. select * from drone.drone_service where emp_no <>3 or emp_no <> 8;
B. select * from drone.drone_service where emp_no <> (3 or 8);
C. select * from drone.drone_service where emp_no <>3 and emp_no <> 8;
D. select * from drone.drone_service where emp_no <> (3 and 8);

Arithmetic Operations

- Can be performed in SQL.
- For example, what is the drone cost per minute:

select drone_id, drone_cost_hr/60 from drone.drone;

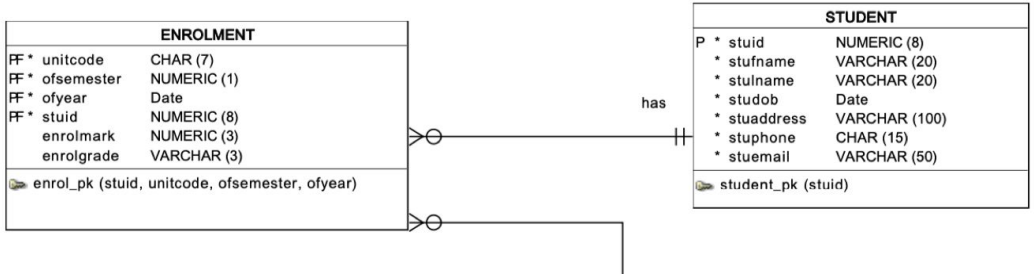
DRONE_ID	DRONE_COST_HR/60
100	0.25
101	0.25
102	0.15
103	0.9166666666666666666666666666666666666667
111	0.75
112	0.75
113	0.75
117	0.75
118	0.2666666666666666666666666666666666666667
119	1
120	1
121	0.2666666666666666666666666666666666666667

Formatting?

Oracle NVL function

- It is used to replace a NULL with a value (numeric OR character/string)

```
SELECT stuid,
   enrolmark,
   enrolgrade
FROM uni.enrolment;
```

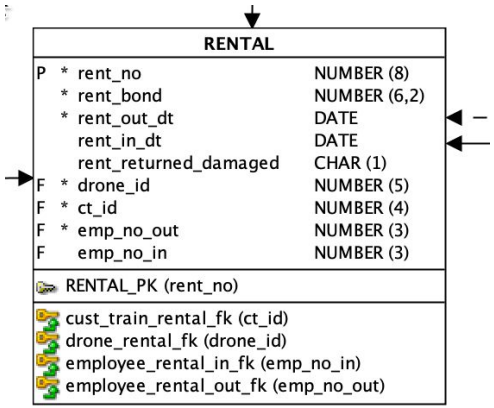


	STUID	ENROLMARK	ENROLGRADE
83	12609485	44	N
84	12802225	(null)	WH
85	12842838	(null)	WH
86	13028303	(null)	(null)
87	13119134	(null)	(null)
88	13390148	(null)	(null)
89	13413109	69	C
90	13453333	52	P
91	13742778	58	P
92	13880303	(null)	DEF

```
SELECT stuid,
   NVL(enrolmark,0),
   NVL(enrolgrade,'WH')
FROM uni.enrolment;
```

	STUID	NVL(ENROLMARK,0)	NVL(ENROLGRADE,'WH')
83	12609485	44	N
84	12802225	0	WH
85	12842838	0	WH
86	13028303	0	WH
87	13119134	0	WH
88	13390148	0	WH
89	13413109	69	C
90	13453333	52	P
91	13742778	58	P
92	13880303	0	DEF

Oracle NVL function continued




```
select rent_no, drone_id, rent_out_dt,
       nvl(rent_in_dt,'Still out') from  drone.rental;
```

Run this command against the Oracle Database

What happens, why?

Renaming Column

- Note column heading from **drone_cost_hr/60**
- Use the word "AS"
  - New column name in " " to maintain case, special characters or spacing
- Example

```
select drone_id, drone_cost_hr/60 as costpermin
from drone.drone;

select drone_id, drone_cost_hr/60 as "COST/MIN"
from drone.drone;
```

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:

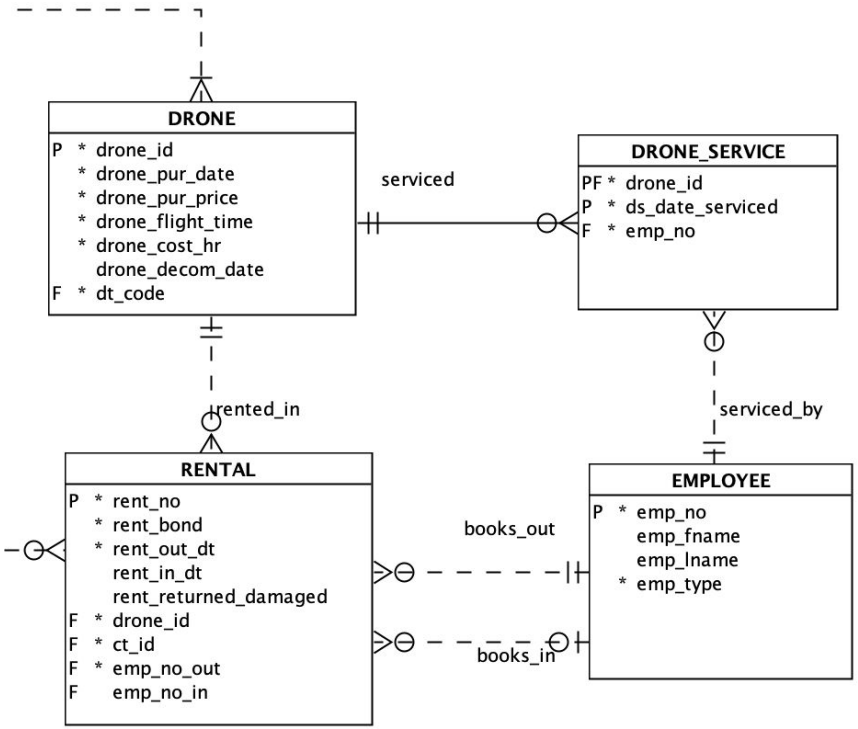
```
select drone_id, drone_flight_time
from  drone.drone order by
drone_flight_time desc, drone_id;
```

DRONE_ID	DRONE_FLIGHT_TIME
103	200
113	150
117	100.5
100	100
111	100
101	60
118	56.3
102	45.5
112	40
120	25.8
119	10.2
121	0

Q4. Write a query to satisfy the following requirements:

- Show the rental number, when the rental was taken out and when the rental was returned
  - no attribute formatting is necessary, use the table column names directly
- The output should show
  - the most recently returned rental first
  - show nulls at the end of the output

Obtain the ids of those drones which have been rented?



Removing Duplicate Rows in the Query Result

- Use "DISTINCT" as part of SELECT clause
  - *use with care*
  - Which of our drones have been rented?

```
select distinct drone_id
from drone.rental
order by drone_id;
```

DRONE_ID
100
101
102
103
111
112
113
117
118
119
120

Run this command against the Oracle Database and note the difference

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 join and then restricted by the where clause
- ANSI JOINS
  - ON
    - the general form which always works, hence the syntax we tend to use
    - FROM drone.manufacturer JOIN drone.drone_type
      ON manufacturer.manuf_id = drone_type.manuf_id
  - USING
    - requires matching attribute/s in the two tables
    - FROM drone.manufacturer JOIN drone.drone_type USING (manuf_id)
  - NATURAL
    - requires matching attribute/s in the two tables
    - FROM drone.manufacturer NATURAL JOIN drone.drone_type

SQL EQUI JOIN

MANUFACTURER

MANUF_ID	MANUF_NAME
10	DJI Da-Jiang Innovations
20	Parrot
30	SwellPro

DRONE_TYPE

DT_CODE	DT_MODEL	DT_CARRY_KG	TRAIN_CODE	MANUF_ID
DMA2	DJI Mavic Air 2 Flymore Combo	0	DJIHY	10
DSPA	DJI Spark	2	DJIHY	10
DIN2	DJI Inspire 2	5	DJIPR	10
PAPR	Parrot Pro	5	PARPO	20
SWPS	SwellPro Spry	0	SWELL	30

Worksheet | Query Builder

```
1  SELECT
2      *
3  FROM
4      drone.manufacturer
5      JOIN drone.drone_type
6      ON manufacturer.manuf_id = drone_type.manuf_id
7  ORDER BY
8      dt_code;
```

Query Result 2 | Query Result 3

SQL | All Rows Fetched: 5 in 0.016 seconds

	MANUF_ID	MANUF_NAME	DT_CODE	DT_MODEL	DT_CARRY_KG	TRAIN_CODE	MANUF_ID_1
1	10	DJI Da-Jiang Innovations	DIN2	DJI Inspire 2	5	DJIPR	10
2	10	DJI Da-Jiang Innovations	DMA2	DJI Mavic Air 2 Flymore Combo	0	DJIHY	10
3	10	DJI Da-Jiang Innovations	DSPA	DJI Spark	2	DJIHY	10
4	20	Parrot	PAPR	Parrot Pro	5	PARPO	20
5	30	SwellPro	SWPS	SwellPro Spry	0	SWELL	30

Special form of EQUI: SQL NATURAL JOIN

MANUFACTURER

MANUF_ID	MANUF_NAME
10	DJI Da-Jiang Innovations
20	Parrot
30	SwellPro

DRONE_TYPE

DT_CODE	DT_MODEL	DT_CARRY_KG	TRAIN_CODE	MANUF_ID
DMA2	DJI Mavic Air 2 Flymore Combo	0	DJIHY	10
DSPA	DJI Spark	2	DJIHY	10
DIN2	DJI Inspire 2	5	DJIPR	10
PAPR	Parrot Pro	5	PARPO	20
SWPS	SwellPro Spry	0	SWELL	30

Worksheet | Query Builder

```
1  SELECT
2      *
3  FROM
4      drone.manufacturer
5      NATURAL JOIN drone.drone_type
6  ORDER BY
7      dt_code;
```

Query Result 2 | Query Result 3

SQL | All Rows Fetched: 5 in 0.01 seconds

	MANUF_ID	MANUF_NAME	DT_CODE	DT_MODEL	DT_CARRY_KG	TRAIN_CODE
1	10	DJI Da-Jiang Innovations	DIN2	DJI Inspire 2	5	DJIPR
2	10	DJI Da-Jiang Innovations	DMA2	DJI Mavic Air 2 Flymore Combo	0	DJIHY
3	10	DJI Da-Jiang Innovations	DSPA	DJI Spark	2	DJIHY
4	20	Parrot	PAPR	Parrot Pro	5	PARPO
5	30	SwellPro	SWPS	SwellPro Spry	0	SWELL

Natural Join Caution

```
SELECT
    *
FROM
    drone.rental
    NATURAL JOIN drone.employee
ORDER BY
    rent_no;
```

	RENT_NO	RENT_BOND	RENT_OUT_DT	RENT_IN_DT	RENT_RETURNED_DAMAGED	DRONE_ID	CT_ID	EMP_NO_OUT	EMP_NO_IN	EMP_NO	EMP_FNAME	EMP_LNAME	EMP_TYPE
1	1	100	20/FEB/21	20/FEB/21	N	100	1	1	1	1	Malika	Casey	F
2	1	100	20/FEB/21	20/FEB/21	N	100	1	1	1	2	Jayden-James	Redman	F
3	1	100	20/FEB/21	20/FEB/21	N	100	1	1	1	3	Jozef	Snow	F
4	1	100	20/FEB/21	20/FEB/21	N	100	1	1	1	5	Efa	Mann	C
5	1	100	20/FEB/21	20/FEB/21	N	100	1	1	1	8	Kajus	Tran	C
6	1	100	20/FEB/21	20/FEB/21	N	100	1	1	1	9	Khadijah	Wong	C
7	1	100	20/FEB/21	20/FEB/21	N	100	1	1	1	10	Jamila	Sutton	C
8	1	100	20/FEB/21	20/FEB/21	N	100	1	1	1	11	Yasmeen	Mohamed	F
9	2	100	21/FEB/21	22/FEB/21	Y	101	2	1	2	1	Malika	Casey	F
10	2	100	21/FEB/21	22/FEB/21	Y	101	2	1	2	2	Jayden-James	Redman	F
11	2	100	21/FEB/21	22/FEB/21	Y	101	2	1	2	3	Jozef	Snow	F
12	2	100	21/FEB/21	22/FEB/21	Y	101	2	1	2	5	Efa	Mann	C

Q5. Find the full name and contact number for all customers who have completed a training course which ran for more than four hours

1. Identify the source tables
2. Build the JOIN table by table (here use ON), maintain all attributes so you can see what is happening
3. Limit rows (where) and attributes (select list)
4. Order by customer name

Special note: the Oracle symbol to concatenate two strings is ||

Output required:

CUST_NAME	CUST_PHONE
Beverie Huntriss	881887799419
Buddy Juden	513324079405
Farly Harcombe	883745850835
Gannon Brenneke	114737189771
Gwynne Reder	730998445142
Jamill Flannery	982489099853
Norrie Severy	403542653485
Robbyn Lintall	867460881352
Serene Pabst	872528687851
Townsend Dunlap	769076023768

Using SQL Sub or Nested Select in DML

- As discussed last week, a SELECT statement can be used to obtain a value or set of values from the database as part of a DML statement
- *Update the hire cost for all drones manufactured by DJI Da-Jiang Innovations by 20%*

```
UPDATE drone
SET
    drone_cost_hr = drone_cost_hr * 1.2
WHERE
    dt_code IN (
        SELECT
            dt_code
        FROM
                drone_type t
            JOIN manufacturer m
            ON t.manuf_id = m.manuf_id
        WHERE
            upper(manuf_name) = upper('DJI Da-Jiang Innovations')
    );
```

Summary

- SQL statement, clause, predicate.
- Writing SQL predicates.
  - Comparison, range, set membership, pattern matching, is NULL
  - Combining predicates using logic operators (AND, OR, NOT)
- Arithmetic operation.
  - NVL function
- Column alias.
- Ordering (Sorting) result.
- Removing duplicate rows.
- JOIN-ing tables

Oracle Date Data Type Revisited

Oracle Date 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 (advantage can use arithmetic)
    - **Output** is controlled by formatting via **to_char**
      - select **to_char**(sysdate,'dd-Mon-yyyy') from dual;
        - » 20-Aug-2022
      - select
        to_char(sysdate,'dd-Mon-yyyy hh:mi:ss AM')
        from dual;
        - » 20-Aug-2022 02:51:24 PM

Oracle Date Datatype - cont'd

- DATE data type **must be formatted** with **TO_CHAR** when selecting for **display**. to_char can also be used to format numbers
- As previously discussed - text representing date **must be formatted** with **TO_DATE** when **comparing** or **inserting/updating.**

Example of Date/Number Output Formats and Comparisons

Report drones purchased after 1st March 2021?

Worksheet | Query Builder

```sql
SELECT
    drone_id,
    to_char(drone_pur_date, 'dd-Mon-yyyy') as purchase_date,
    to_char(drone_pur_price, '$9990.99') as purchase_price,
    to_char(drone_flight_time, '99990.9') as flight_time
FROM
    drone.drone
WHERE
    drone_pur_date > TO_DATE('01-Mar-2021','dd-Mon-yyyy')
ORDER BY
    drone_id;
```

Query Result

SQL | All Rows Fetched: 8 in 0.009 seconds

	DRONE_ID	PURCHASE_DATE	PURCHASE_PRICE	FLIGHT_TIME
1	111	20-Mar-2021	$4200.00	100.0
2	112	20-Mar-2021	$4200.00	40.0
3	113	20-Mar-2021	$4000.00	150.0
4	117	20-Mar-2021	$4000.00	100.5
5	118	01-Apr-2021	$1599.00	56.3
6	119	01-Apr-2022	$5600.80	10.2
7	120	01-Apr-2022	$4200.80	25.8
8	121	17-Apr-2022	$1610.00	0.0

Returning to Oracle NVL function

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

```
select rent_no, drone_id, rent_out_dt,
      nvl(rent_in_dt,'Still out') from  drone.rental;
```

- rent_in_dt is date, 'Still out' is string (char)

```
select rent_no, drone_id,
      to_char(rent_out_dt,'dd-Mon-yyyy') as dateout,
      nvl(to_char(rent_in_dt,'dd-Mon-yyyy'),'Still out')
         as datein
      from  drone.rental;
```

	RENT_NO	DRONE_ID	DATEOUT	DATEIN
1	1	100	20-Feb-2021	20-Feb-2021
2	2	101	21-Feb-2021	22-Feb-2021
3	3	102	22-Feb-2021	23-Feb-2021
4	4	100	22-Feb-2021	25-Feb-2021
5	5	101	25-Feb-2021	25-Feb-2021
6	6	102	28-Feb-2021	28-Mar-2021
7	7	103	01-Mar-2021	02-Mar-2021
8	8	103	03-Mar-2021	04-Mar-2021
9	9	103	06-Mar-2021	10-Mar-2021
10	10	101	10-Mar-2021	18-Mar-2021
11	11	111	26-Apr-2021	28-Apr-2021
12	12	112	26-Apr-2021	27-Apr-2021
13	13	113	28-Apr-2021	29-Apr-2021
14	14	117	28-Apr-2021	05-May-2021
15	15	103	01-May-2021	02-May-2021
16	16	103	03-May-2021	10-May-2021
17	17	112	03-May-2021	07-May-2021
18	18	113	03-May-2021	12-May-2021
19	19	118	17-May-2021	18-May-2021
20	20	118	19-May-2021	23-May-2021
21	21	118	28-May-2021	29-May-2021
22	22	118	01-Jun-2021	07-Jun-2021
23	23	119	21-Aug-2022	Still out
24	24	120	22-Aug-2022	Still out
25	25	118	23-Aug-2022	Still out

Current Date

- Current date can be queried from the DUAL table (used to evaluate expressions/functions) by calling **SYSDATE**

```
SELECT
    to_char(sysdate, 'dd-Mon-yyyy hh:mi:ss AM') AS current_datetime
FROM
    dual;
```

- Oracle internal attributes include:
  - **sysdate**: current date/time for database server
  - **current_date**: current date/time for session
  - **systimestamp**: current database server date/time as a timestamp
  - **user**: current logged in user

What is the current date and time according to Oracle