

# Preparation for the workshop - ready, set .....

Please

- connect to Flux - flux.qa and be ready to answer questions
- login to the Oracle database (local install, MoVE or ORDS)
- on campus students nominate someone as your table leader for this week

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder





MONASH  
University

MONASH  
INFORMATION  
TECHNOLOGY

## Assignment Project Exam Help

Week 7

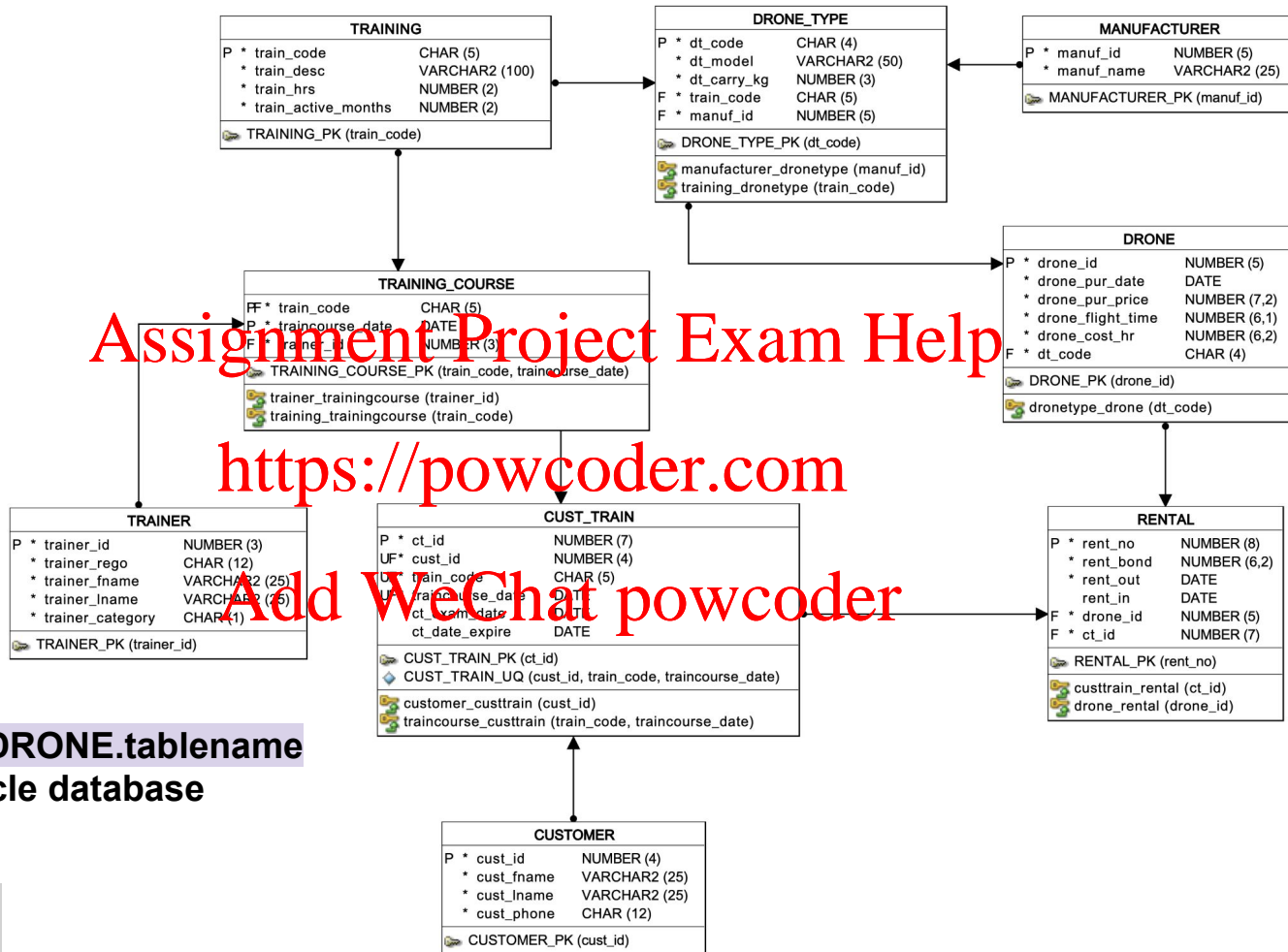
<https://powcoder.com>

Structured Query Language (SQL) – Part I

Add WeChat powcoder

Workshop 2022 S1





Access tables via **DRONE.tablename**  
in the Monash Oracle database



# Anatomy of an SQL SELECT Statement

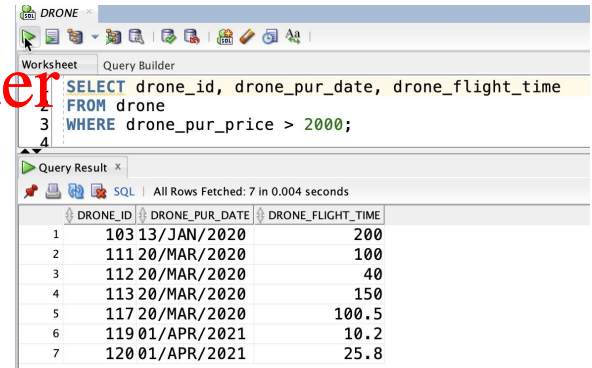
clauses

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

statement

Predicate / search condition

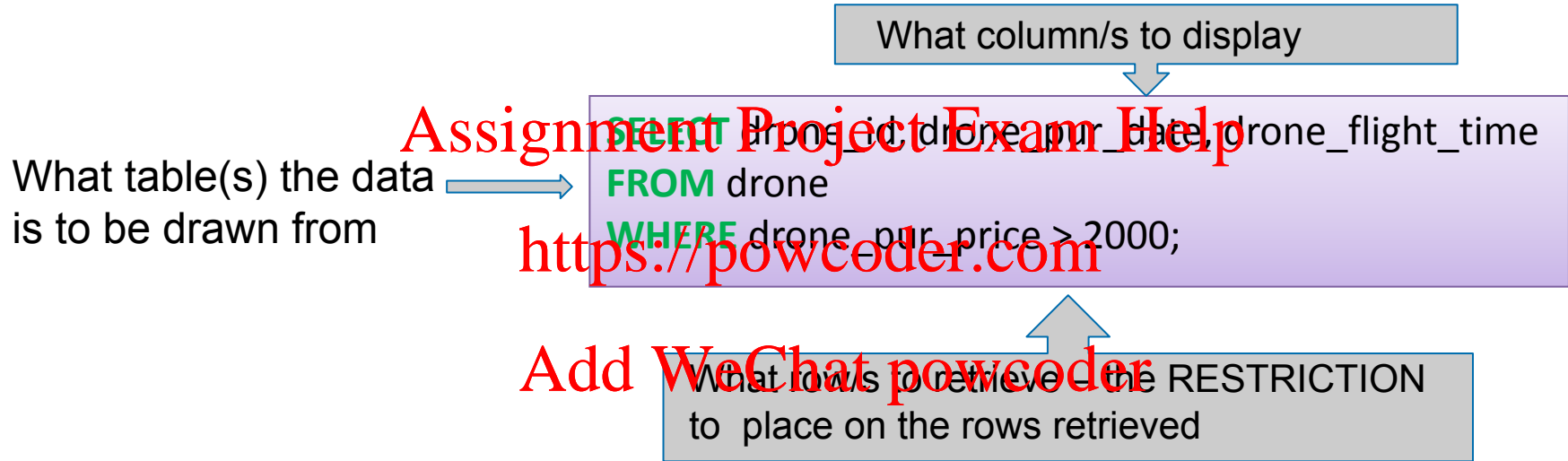
Note slides use *tablename* not *drone.tablename* - command as run by drone account user



The screenshot shows a database application window titled 'DRONE'. It has a 'Query Builder' tab where the SQL statement is entered: `SELECT drone_id, drone_pur_date, drone_flight_time FROM drone WHERE drone_pur_price > 2000;`. Below the query, the 'Query Result' tab shows a table with 7 rows of data. The columns are `DRONE_ID`, `DRONE_PUR_DATE`, and `DRONE_FLIGHT_TIME`. The data is as follows:

	DRONE_ID	DRONE_PUR_DATE	DRONE_FLIGHT_TIME
1	103	13/JAN/2020	200
2	111	20/MAR/2020	100
3	112	20/MAR/2020	40
4	113	20/MAR/2020	150
5	117	20/MAR/2020	100.5
6	119	01/APR/2021	10.2
7	120	01/APR/2021	25.8

# SQL SELECT Statement - Usage



***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 where drone_pur_price BETWEEN 3000 AND 5300;`
- B. `SELECT * FROM drone where drone_pur_price >= 3000 or drone_pur_price <= 5300;`
- C. `SELECT * FROM drone where drone_pur_price IN (3000,5300);`
- D. `SELECT * FROM drone where drone_pur_price >= 3000 and drone_pur_price <= 5300;`
- E. `SELECT * FROM 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 rental where rent_in = null;`
- B. `select * from rental where rent_in is null;`
- C. `select * from rental where rent_in is not null;`
- D. `select * from rental where rent_in is empty;`

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder



# SQL Predicates or Search Conditions

## ■ NULL

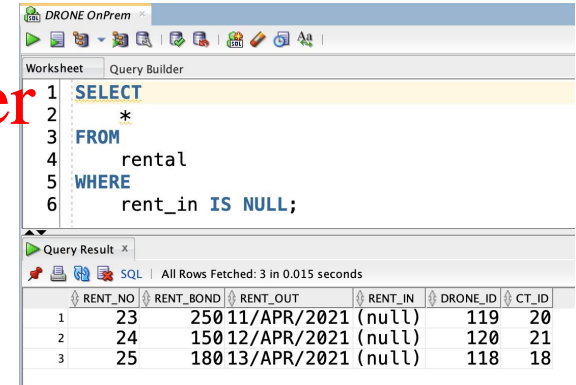
- To test whether a column has a NULL (unknown) value.

- Example: WHERE rent\_in IS NULL.

- Use in subquery (to be discussed in the future)

- ANY, ALL

- EXISTS



Query Builder

```
1 SELECT
2 *
3 FROM
4 rental
5 WHERE
6 rent_in IS NULL;
```

Query Result

All Rows Fetched: 3 in 0.015 seconds

	RENT_NO	RENT_BOND	RENT_OUT	RENT_IN	DRONE_ID	CT_ID
1	23	250	11/APR/2021	(null)	119	20
2	24	150	12/APR/2021	(null)	120	21
3	25	180	13/APR/2021	(null)	118	18

## 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**

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

# 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

Add WeChat powcoder

Unknown = NULL in  
relational database

Q3. Find all the training courses which are not run by the trainer with trainer\_id 1 or the trainer with trainer\_id 2:

TRAIN_CODE	TRAINCOURSE_DATE	TRAINER_ID
DJIHY	14/FEB/2020	1
DJIPR	18/FEB/2020	2
PARPO	25/APR/2020	3
SWELL	10/MAY/2020	1
DJIPR	10/APR/2021	1

Assignment Project Exam Help

<https://powcoder.com>

- A. select \* from training\_course where trainer\_id <>1 or trainer\_id <> 2;
- B. select \* from training\_course where trainer\_id <> (1 or 2);
- C. select \* from training\_course where trainer\_id <>1 and trainer\_id <> 2;
- D. select \* from training\_course where trainer\_id <> (1 and 2);

# 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_ID	DRONE_COST_HR/60
100		0.25
101		0.25
102		0.15
103	0.9166666666666667	0.25
111		0.75
112		0.75
113		0.75
117		0.75
118	0.2666666666666667	0.25
119		1
120		1
121	0.2666666666666667	0.25

## Formatting?



# Oracle NVL function

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

UNIT		
P	* unit_code	CHAR (7)
	* unit_name	VARCHAR (50)

ENROLMENT		
PF	* stu_nbr	NUMERIC (8)
P	* unit_code	CHAR (7)
P	* enrol_year	NUMERIC (4)
P	* enrol_semester	CHAR (2)
	* enrol_mark	NUMERIC (3)
	* enrol_grade	CHAR (2)

STUDENT		
P	* stu_nbr	NUMERIC (8)
	* stu_lname	VARCHAR (50)
	* stu_fname	VARCHAR (50)
	* stu_dob	Date

<https://powcoder.com>

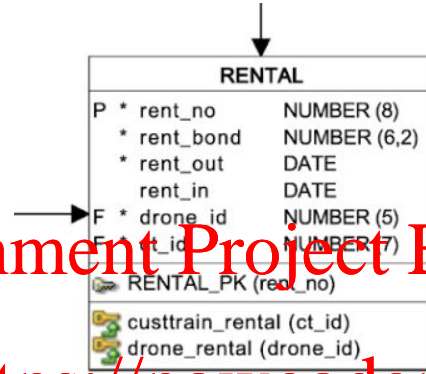
```
SELECT stu_nbr, NVL(enrol_mark,0),  
       NVL(enrol_grade,'WH')  
FROM enrolment;
```

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





# Oracle NVL function continued



RENTAL		
P *	rent_no	NUMBER (8)
*	rent_bond	NUMBER (6,2)
*	rent_out	DATE
	rent_in	DATE
F *	drone_id	NUMBER (5)
F *	rent_id	NUMBER (7)
	RENTAL_PK (rent_no)	
	custtrain_rental (ct_id)	
	drone_rental (drone_id)	

Assignment Project Exam Help

<https://powcoder.com>

select rent\_no, drone\_id, rent\_out,  
nvl(rent\_in, 'Still With') from rental;

Add WeChat powcoder



**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

<https://powcoder.com>

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

Add WeChat powcoder

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

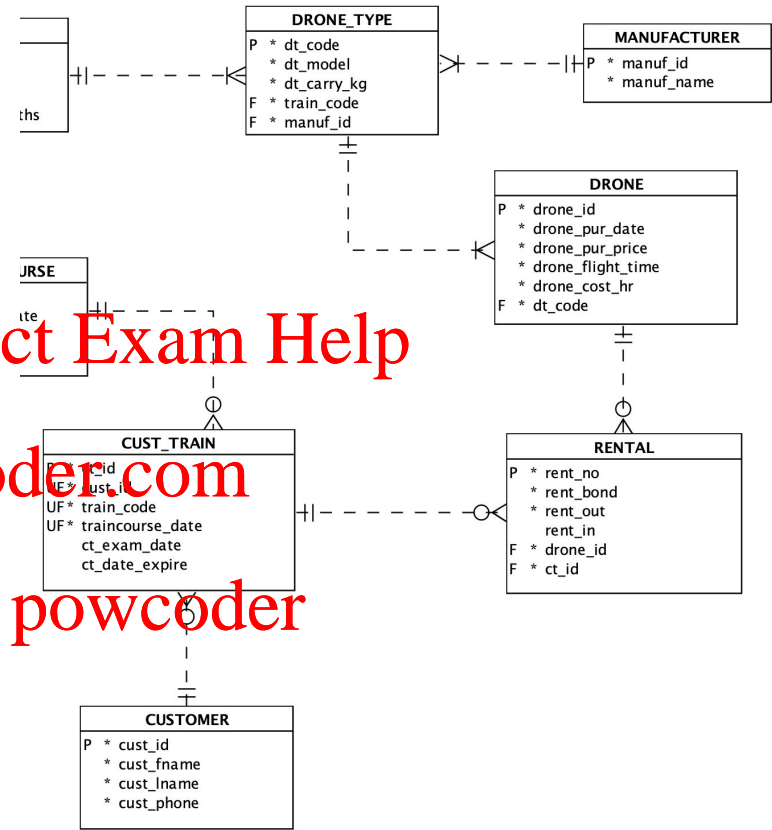
#### Q4. Write a query to satisfy the following requirements:

- 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

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder



Obtain the ids of those drones  
which have been rented?

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder



MONASH  
University



*Run this command against the Oracle Database*

# 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 rental  
order by drone_id;
```

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



# 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 trainer JOIN training\_course ON trainer.trainer\_id = training\_course.trainer\_id
  - USING
    - requires matching attribute/s in the two tables
    - FROM trainer JOIN training\_course USING (trainer\_id)
  - NATURAL
    - requires matching attribute/s in the two tables
    - FROM trainer NATURAL JOIN training\_course

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder



# SQL EQUI JOIN

TRAINER

TRAINER_ID	TRAINER_REGO	TRAINER_FNAME	TRAINER_LNAME	TRAINER_CATEGORY
1	DR778589-191	Clementius	Cambell	F
2	DR055102-311	Kerwinn	Booeln	C
3	DR322351-719	Charmain	Jado	F
4	DR655884-106	Gaylord	Colegate	F
5	DR820983-603	Garv	Gretton	C

TRAINING\_COURSE

TRAIN_CODE	TRAINCOURSE_DATE	TRAINER_ID
DJIHY	14/FEB/2020	1
DJIPR	18/FEB/2020	2
PARPO	25/APR/2020	3
SWELL	10/MAY/2020	4
DJIPR	10/APR/2021	1

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

Worksheet Query Builder

```
1 SELECT
2 *
3 FROM
4     trainer
5 JOIN training_course
6 ON trainer.trainer_id = training_course.trainer_id
7 ORDER BY
8     traincourse_date,
9     train_code;
```

Query Result x

All Rows Fetched: 5 in 0.004 seconds

	TRAINER_ID	TRAINER_REGO	TRAINER_FNAME	TRAINER_LNAME	TRAINER_CATEGORY	TRAIN_CODE	TRAINCOURSE_DATE	TRAINER_ID_1
1	1	DR778589-191	Clementius	Cambell	F	DJIHY	14/FEB/2020	1
2	2	DR055102-311	Kerwinn	Booeln	C	DJIPR	18/FEB/2020	2
3	3	DR322351-719	Charmain	Jado	F	PARPO	25/APR/2020	3
4	4	DR655884-106	Gaylord	Colegate	F	SWELL	10/MAY/2020	4
5	1	DR778589-191	Clementius	Cambell	F	DJIPR	10/APR/2021	1



# Special form of EQUI: SQL NATURAL JOIN

TRAINER

TRAINER_ID	TRAINER_REGO	TRAINER_FNAME	TRAINER_LNAME	TRAINER_CATEGORY
1	DR778589-191	Clementius	Cambell	F
2	DR055102-311	Kerwinn	Booeln	C
3	DR322351-719	Charmain	Jado	F
4	DR655884-106	Gaylord	Colegate	F
5	DR820983-603	Garv	Gretton	C

TRAINING COURSE

TRAIN_CODE	TRAINCOURSE_DATE	TRAINER_ID
DJIHY	14/FEB/2020	1
DJIPR	18/FEB/2020	2
PARPO	25/APR/2020	3
SWELL	10/MAY/2020	4
DJIPR	10/APR/2021	1

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

```

1 SELECT
2   train_code,
3   traincourse_date,
4   trainer.trainer_id,
5   trainer_fname,
6   trainer_lname
7 FROM
8   trainer
9 JOIN training_course
10  ON trainer.trainer_id = training_course.trainer_id
11 ORDER BY
12   traincourse_date,
13   train_code;
    
```

Query Result x

All Rows Fetched: 5 in 0.018 seconds

TRAIN_CODE	TRAINCOURSE_DATE	TRAINER_ID	TRAINER_FNAME	TRAINER_LNAME
1 DJIHY	14/FEB/2020	1	Clementius	Cambell
2 DJIPR	18/FEB/2020	2	Kerwinn	Booeln
3 PARPO	25/APR/2020	3	Charmain	Jado
4 SWELL	10/MAY/2020	4	Gaylord	Colegate
5 DJIPR	10/APR/2021	1	Clementius	Cambell

```

1 SELECT
2   train_code,
3   traincourse_date,
4   trainer_id,
5   trainer_fname,
6   trainer_lname
7 FROM
8   trainer
9 NATURAL JOIN training_course
10 ORDER BY
11   traincourse_date,
12   train_code
    
```

Query Result x

All Rows Fetched: 5 in 0.003 seconds

TRAIN_CODE	TRAINCOURSE_DATE	TRAINER_ID	TRAINER_FNAME	TRAINER_LNAME
1 DJIHY	14/FEB/2020	1	Clementius	Cambell
2 DJIPR	18/FEB/2020	2	Kerwinn	Booeln
3 PARPO	25/APR/2020	3	Charmain	Jado
4 SWELL	10/MAY/2020	4	Gaylord	Colegate
5 DJIPR	10/APR/2021	1	Clementius	Cambell



## Q5. Find the full name and contact number for all customers who have completed a training course run by trainer id 1

1. Identify the source tables
2. Build the JOIN table by table (re-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

### Output required:

CUST_NAME	CUST_PHONE
Christiana Brightey	214848997962
Jamill Flannery	982489099853
Lennard Dudgeon	245445205577
Manolo Waren	826097815268
Raychel Roussel	745110667679
Serene Pabst	872528687851
Townsend Dunlap	769076023768

<https://powcoder.com>

Add WeChat powcoder

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

# 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

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder



Assignment Project Exam Help

Oracle Date Data Type Revisited

<https://powcoder.com>

Add WeChat powcoder

# 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-Apr-2021
      - select  
**to\_char**(sysdate,'dd-Mon-yyyy hh:mi:ss AM')  
from dual;  
» 20-Apr-2020 02:51:24 PM

- DATE data type **must be formatted** with **TO\_CHAR** when selecting for **display**. to\_char can also be used to format numbers
- Text representing date **must be formatted** with **TO\_DATE** when **comparing** or **inserting/updating**.

Assignment Project Exam Help

<https://powcoder.com>

Add WeChat powcoder

Report drones purchased after  
1st March 2020?

DRONE_ID	PURCHASE_DATE	PURCHASE_PRICE	FLIGHT_TIME
111	20-Mar-2020	\$4200.00	100.0
112	20-Mar-2020	\$4200.00	40.0
113	20-Mar-2020	\$4200.00	150.0
117	20-Mar-2020	\$4200.00	100.5
118	01-Apr-2020	\$1599.00	56.3
119	01-Apr-2021	\$5600.80	10.2
120	01-Apr-2021	\$5600.80	25.8
121	17-Apr-2021	\$1610.00	0.0

Worksheet Query Builder

```
1 SELECT
2   drone_id,
3   to_char(drone_pur_date, 'dd-Mon-yyyy') AS purchase_date,
4   to_char(drone_pur_price, '$99999.99') AS purchase_price,
5   to_char(drone_flight_time, '99990.9') AS flight_time
6 FROM
7   drone
8 WHERE
9   drone_pur_date > TO_DATE('01-Mar-2020', 'dd-Mon-yyyy')
10 ORDER BY
11   drone_id;
```

Query Result x

SQL All Rows Fetched: 8 in 0.002 seconds

	DRONE_ID	PURCHASE_DATE	PURCHASE_PRICE	FLIGHT_TIME
1	111	20-Mar-2020	\$4200.00	100.0
2	112	20-Mar-2020	\$4200.00	40.0
3	113	20-Mar-2020	\$4200.00	150.0
4	117	20-Mar-2020	\$4200.00	100.5
5	118	01-Apr-2020	\$1599.00	56.3
6	119	01-Apr-2021	\$5600.80	10.2
7	120	01-Apr-2021	\$5600.80	25.8
8	121	17-Apr-2021	\$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,  
       nvl(rent_in,'Still out') from rental;
```

- rent\_in is **date**, 'Still out' is string (**char**)

```
select rent_no, drone_id,  
       to_char(rent_out,'dd-Mon-yyyy') as dateout,  
       nvl(to_char(rent_in,'dd-Mon-yyyy'),'Still out')  
       as datein  
from rental;
```

RENT_NO	DRONE_ID	DATEOUT	DATEIN
1	100	20-Feb-2020	20-Feb-2020
2	101	21-Feb-2020	22-Feb-2020
3	102	22-Feb-2020	23-Feb-2020
4	100	22-Feb-2020	25-Feb-2020
5	101	25-Feb-2020	25-Feb-2020
6	103	28-Feb-2020	28-Mar-2020
7	103	01-Mar-2020	02-Mar-2020
8	103	03-Mar-2020	04-Mar-2020
9	103	06-Mar-2020	10-Mar-2020
10	101	10-Mar-2020	18-Mar-2020
11	111	26-Apr-2020	28-Apr-2020
12	112	26-Apr-2020	27-Apr-2020
13	113	28-Apr-2020	29-Apr-2020
14	117	28-Apr-2020	05-May-2020
15	103	01-May-2020	02-May-2020
16	103	03-May-2020	10-May-2020
17	112	03-May-2020	07-May-2020
18	113	03-May-2020	12-May-2020
19	118	17-May-2020	18-May-2020
20	118	19-May-2020	23-May-2020
21	118	28-May-2020	29-May-2020
22	118	01-Jun-2020	07-Jun-2020
23	119	11-Apr-2021	Still out
24	120	12-Apr-2021	Still out
25	118	13-Apr-2021	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:mm:ss AM') AS current_datetime  
FROM  
  dual;
```

- Oracle internal attributes include:
  - **sysdate**: current date/time
  - **systimestamp**: current date/time as a timestamp
  - **user**: current logged in user

