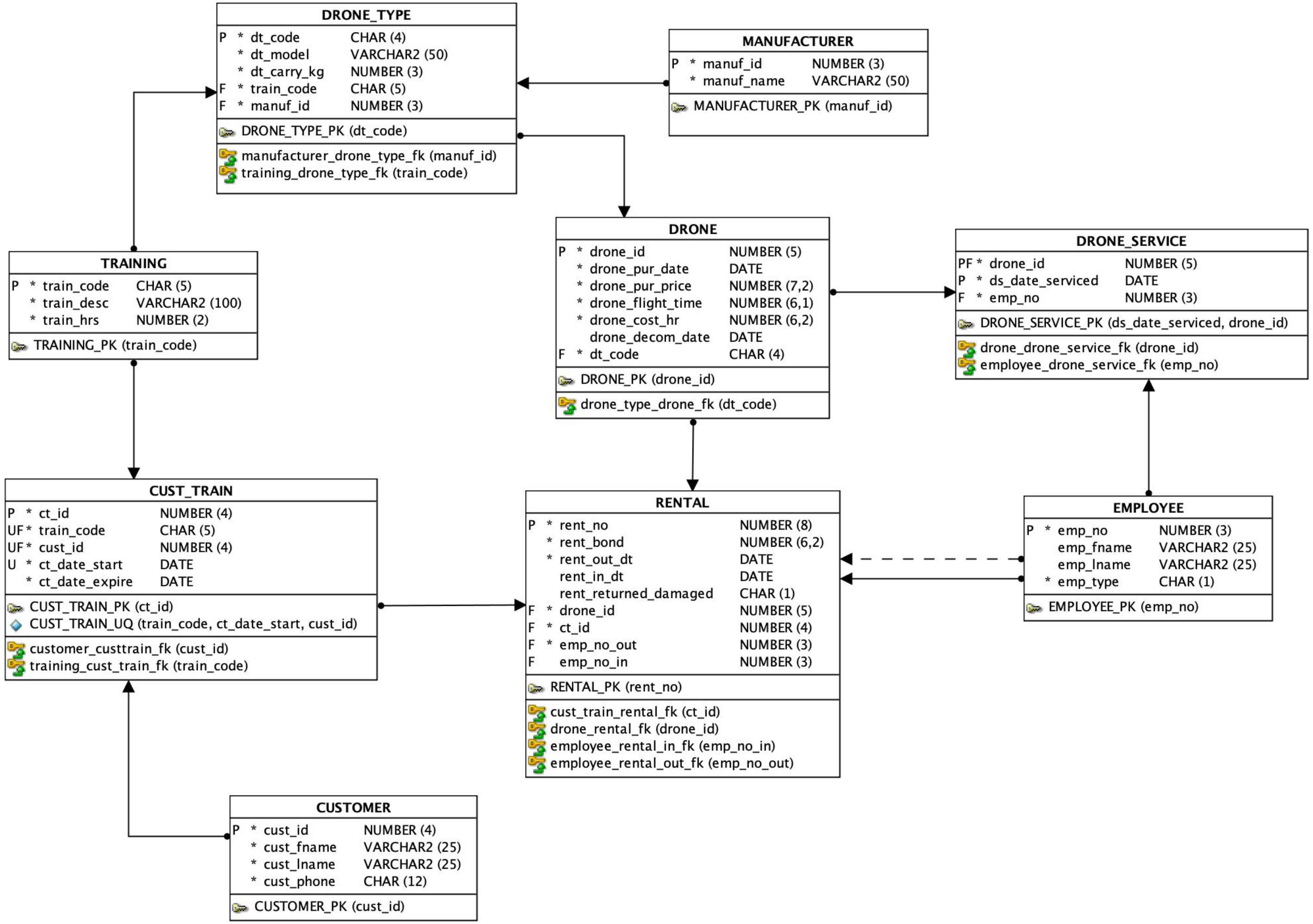
MONASH
University

MONASH
INFORMATION
TECHNOLOGY

Topic 9
SQL Intermediate

Workshop 2025 S1

GROUP
OF EIGHT
AUSTRALIA



DRONE_TYPE

P	*	dt_code	CHAR (4)
	*	dt_model	VARCHAR2 (50)
	*	dt_carry_kg	NUMBER (3)
F	*	train_code	CHAR (5)
F	*	manuf_id	NUMBER (3)

DRONE_TYPE_PK (dt_code)

manufacturer_drone_type_fk (manuf_id)
training_drone_type_fk (train_code)

MANUFACTURER

P	*	manuf_id	NUMBER (3)
	*	manuf_name	VARCHAR2 (50)

MANUFACTURER_PK (manuf_id)

TRAINING

P	*	train_code	CHAR (5)
	*	train_desc	VARCHAR2 (100)
	*	train_hrs	NUMBER (2)

TRAINING_PK (train_code)

DRONE

P	*	drone_id	NUMBER (5)
	*	drone_pur_date	DATE
	*	drone_pur_price	NUMBER (7,2)
	*	drone_flight_time	NUMBER (6,1)
	*	drone_cost_hr	NUMBER (6,2)
		drone_decom_date	DATE
F	*	dt_code	CHAR (4)

DRONE_PK (drone_id)

drone_type_drone_fk (dt_code)

DRONE_SERVICE

PF	*	drone_id	NUMBER (5)
P	*	ds_date_serviced	DATE
F	*	emp_no	NUMBER (3)

DRONE_SERVICE_PK (ds_date_serviced, drone_id)

drone_drone_service_fk (drone_id)
employee_drone_service_fk (emp_no)

CUST_TRAIN

P	*	ct_id	NUMBER (4)
UF	*	train_code	CHAR (5)
UF	*	cust_id	NUMBER (4)
U	*	ct_date_start	DATE
	*	ct_date_expire	DATE

CUST_TRAIN_PK (ct_id)
CUST_TRAIN_UQ (train_code, ct_date_start, cust_id)

customer_custtrain_fk (cust_id)
training_cust_train_fk (train_code)

RENTAL

P	*	rent_no	NUMBER (8)
	*	rent_bond	NUMBER (6,2)
	*	rent_out_dt	DATE
		rent_in_dt	DATE
		rent_returned_damaged	CHAR (1)
F	*	drone_id	NUMBER (5)
F	*	ct_id	NUMBER (4)
F	*	emp_no_out	NUMBER (3)
F		emp_no_in	NUMBER (3)

RENTAL_PK (rent_no)

cust_train_rental_fk (ct_id)
drone_rental_fk (drone_id)
employee_rental_in_fk (emp_no_in)
employee_rental_out_fk (emp_no_out)

EMPLOYEE

P	*	emp_no	NUMBER (3)
		emp_fname	VARCHAR2 (25)
		emp_lname	VARCHAR2 (25)
	*	emp_type	CHAR (1)

EMPLOYEE_PK (emp_no)

CUSTOMER

P	*	cust_id	NUMBER (4)
	*	cust_fname	VARCHAR2 (25)
	*	cust_lname	VARCHAR2 (25)
	*	cust_phone	CHAR (12)

CUSTOMER_PK (cust_id)

Access tables via DRONE.tablename in Monash Oracle database

MONASH University

Aggregate Functions

- COUNT, MAX, MIN, SUM, AVG
- Example:

```
SELECT
    MAX(drone_flight_time)
FROM
    drone.drone;
```

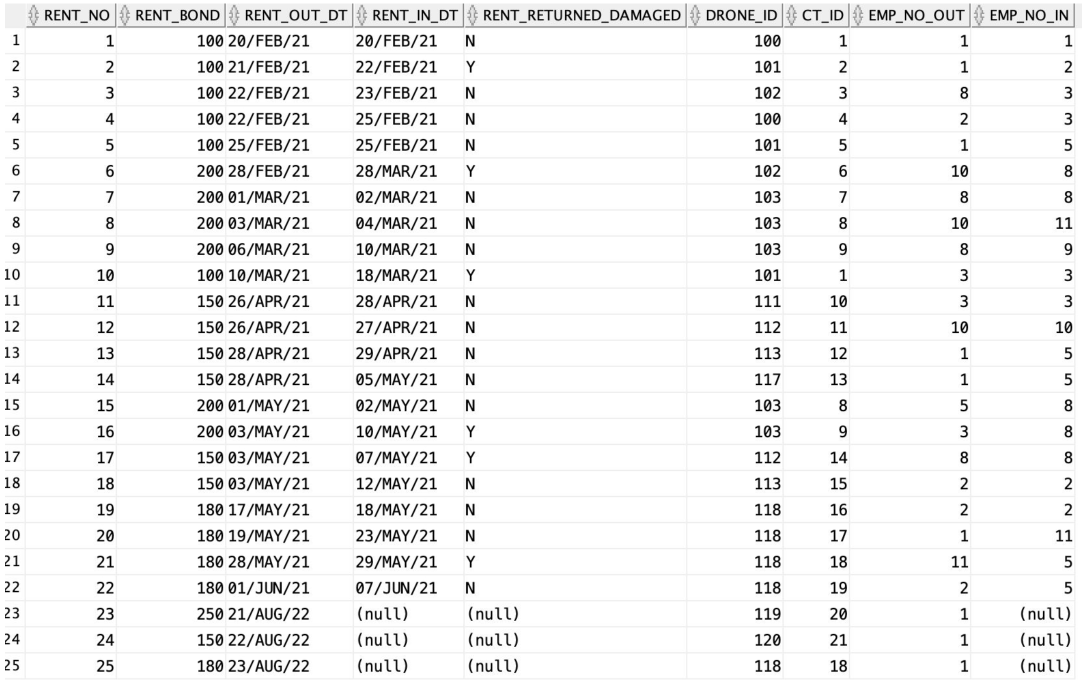
```
SELECT
    MIN(drone_flight_time)
FROM
    drone.drone;
```

```
SELECT
    AVG(drone_flight_time)
FROM
    drone.drone;
```

```
SELECT COUNT(*)
FROM drone.drone
WHERE drone_flight_time > 100;
```

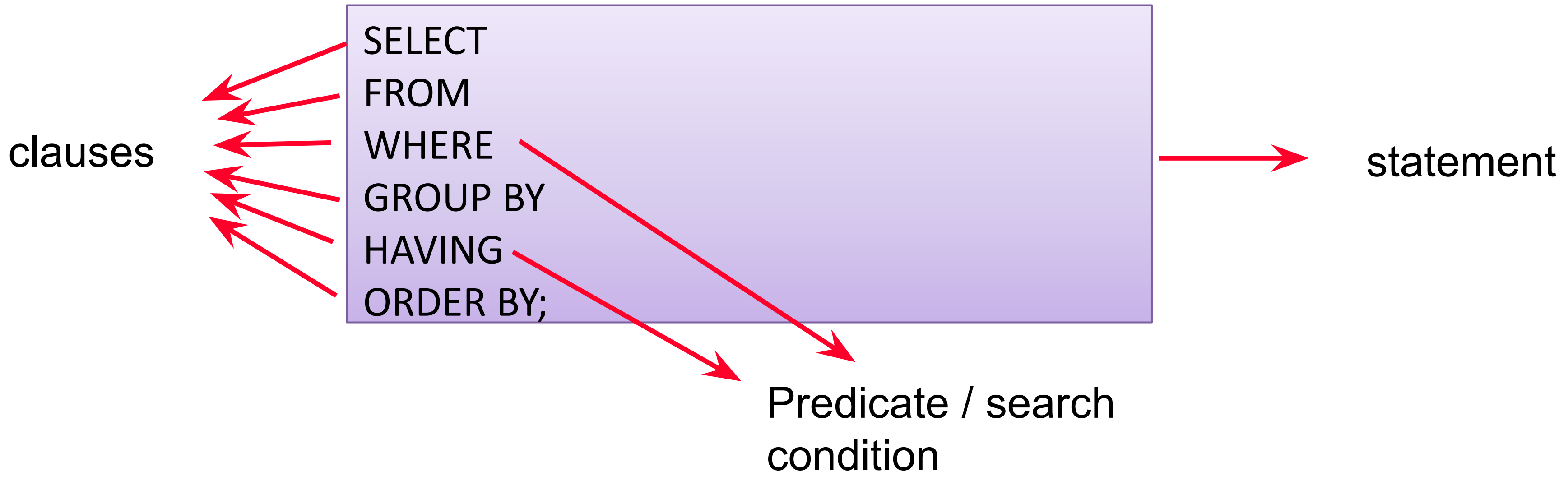
count(*) and count(column_name)

```
SQL> SELECT
  2      COUNT(*),
  3      COUNT(rent_out_dt),
  4      COUNT(rent_in_dt)
  5  FROM
  6      drone.rental;

  COUNT(*) COUNT(RENT_OUT_DT) COUNT(RENT_IN_DT)
---------- ------------------ -----------------
        25                 25                22
```



	RENT_NO	RENT_BOND	RENT_OUT_DT	RENT_IN_DT	RENT_RETURNED_DAMAGED	DRONE_ID	CT_ID	EMP_NO_OUT	EMP_NO_IN
1	1	100	20/FEB/21	20/FEB/21	N	100	1	1	1
2	2	100	21/FEB/21	22/FEB/21	Y	101	2	1	2
3	3	100	22/FEB/21	23/FEB/21	N	102	3	8	3
4	4	100	22/FEB/21	25/FEB/21	N	100	4	2	3
5	5	100	25/FEB/21	25/FEB/21	N	101	5	1	5
6	6	200	28/FEB/21	28/MAR/21	Y	102	6	10	8
7	7	200	01/MAR/21	02/MAR/21	N	103	7	8	8
8	8	200	03/MAR/21	04/MAR/21	N	103	8	10	11
9	9	200	06/MAR/21	10/MAR/21	N	103	9	8	9
10	10	100	10/MAR/21	18/MAR/21	Y	101	1	3	3
11	11	150	26/APR/21	28/APR/21	N	111	10	3	3
12	12	150	26/APR/21	27/APR/21	N	112	11	10	10
13	13	150	28/APR/21	29/APR/21	N	113	12	1	5
14	14	150	28/APR/21	05/MAY/21	N	117	13	1	5
15	15	200	01/MAY/21	02/MAY/21	N	103	8	5	8
16	16	200	03/MAY/21	10/MAY/21	Y	103	9	3	8
17	17	150	03/MAY/21	07/MAY/21	Y	112	14	8	8
18	18	150	03/MAY/21	12/MAY/21	N	113	15	2	2
19	19	180	17/MAY/21	18/MAY/21	N	118	16	2	2
20	20	180	19/MAY/21	23/MAY/21	N	118	17	1	11
21	21	180	28/MAY/21	29/MAY/21	Y	118	18	11	5
22	22	180	01/JUN/21	07/JUN/21	N	118	19	2	5
23	23	250	21/AUG/22	(null)	(null)	119	20	1	(null)
24	24	150	22/AUG/22	(null)	(null)	120	21	1	(null)
25	25	180	23/AUG/22	(null)	(null)	118	18	1	(null)

Anatomy of an SQL Statement - Revisited



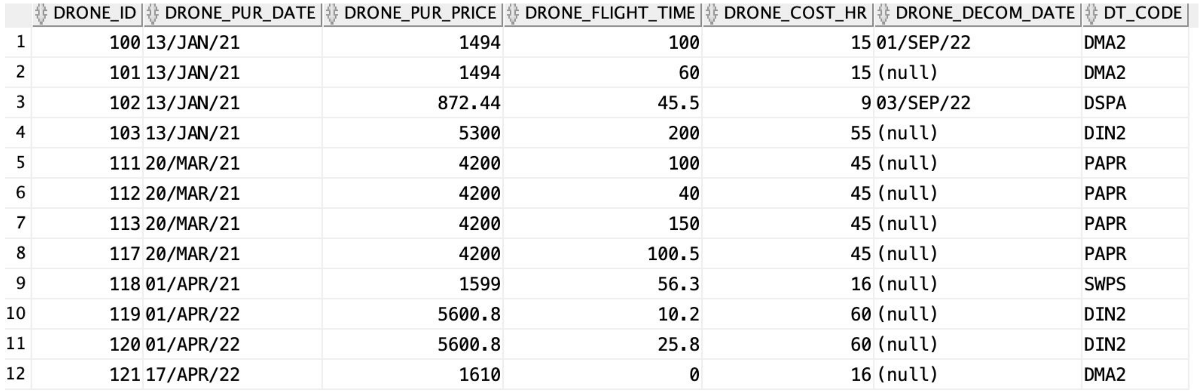
GROUP BY

- If a GROUP BY clause is used with aggregate function, the DBMS will apply the aggregate function to the different groups defined in the clause rather than all rows.

```
SELECT
 AVG(drone_flight_time)
FROM
    drone.drone;
```

```
SELECT dt_code, AVG(drone_flight_time)
FROM  drone.drone
GROUP BY  dt_code
ORDER BY dt_code;
```

```
SQL> SELECT
  2      AVG(drone_flight_time)
  3  FROM
  4      drone.drone;

AVG(DRONE_FLIGHT_TIME)
----------------------
                74.025
SQL>
SQL> SELECT
  2      dt_code,
  3      AVG(drone_flight_time)
  4  FROM
  5      drone.drone
  6  GROUP BY
  7      dt_code
  8  ORDER BY
  9      dt_code;

DT_C AVG(DRONE_FLIGHT_TIME)
---- ----------------------
DIN2             78.6666667
DMA2             53.3333333
DSPA                   45.5
PAPR                 97.625
SWPS                   56.3
```



	DRONE_ID	DRONE_PUR_DATE	DRONE_PUR_PRICE	DRONE_FLIGHT_TIME	DRONE_COST_HR	DRONE_DECOM_DATE	DT_CODE
1	100	13/JAN/21	1494	100	15	01/SEP/22	DMA2
2	101	13/JAN/21	1494	60	15	(null)	DMA2
3	102	13/JAN/21	872.44	45.5	9	03/SEP/22	DSPA
4	103	13/JAN/21	5300	200	55	(null)	DIN2
5	111	20/MAR/21	4200	100	45	(null)	PAPR
6	112	20/MAR/21	4200	40	45	(null)	PAPR
7	113	20/MAR/21	4200	150	45	(null)	PAPR
8	117	20/MAR/21	4200	100.5	45	(null)	PAPR
9	118	01/APR/21	1599	56.3	16	(null)	SWPS
10	119	01/APR/22	5600.8	10.2	60	(null)	DIN2
11	120	01/APR/22	5600.8	25.8	60	(null)	DIN2
12	121	17/APR/22	1610	0	16	(null)	DMA2

Q1. List all customer ids and the total number of courses taken by each customer:

A.
```
select cust_id, count(*) as no_of_courses_taken
from drone.cust_train
order by cust_id;
```

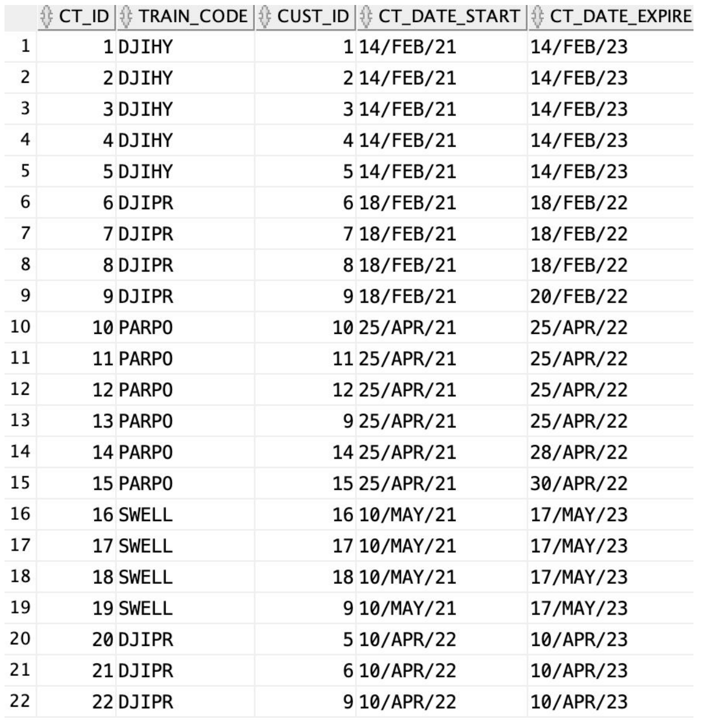
B.
```
select cust_id, sum(train_code) as no_of_courses_taken
from drone.cust_train
group by cust_id
order by cust_id;
```

C.
```
select cust_id, count(*) as no_of_courses_taken
from drone.cust_train
group by cust_id
order by cust_id;
```

D. None of the above

What output is produced?

```
SELECT count(*)
FROM drone.cust_train;

SELECT cust_id, COUNT(*) AS no_courses_taken
FROM drone.cust_train
GROUP BY cust_id
ORDER BY cust_id;

SELECT AVG(COUNT(*))
       AS average_no_courses_taken
FROM drone.cust_train
GROUP BY cust_id;
```



	CT_ID	TRAIN_CODE	CUST_ID	CT_DATE_START	CT_DATE_EXPIRE
1	1	DJIHY	1	14/FEB/21	14/FEB/23
2	2	DJIHY	2	14/FEB/21	14/FEB/23
3	3	DJIHY	3	14/FEB/21	14/FEB/23
4	4	DJIHY	4	14/FEB/21	14/FEB/23
5	5	DJIHY	5	14/FEB/21	14/FEB/23
6	6	DJIPR	6	18/FEB/21	18/FEB/22
7	7	DJIPR	7	18/FEB/21	18/FEB/22
8	8	DJIPR	8	18/FEB/21	18/FEB/22
9	9	DJIPR	9	18/FEB/21	20/FEB/22
10	10	PARPO	10	25/APR/21	25/APR/22
11	11	PARPO	11	25/APR/21	25/APR/22
12	12	PARPO	12	25/APR/21	25/APR/22
13	13	PARPO	9	25/APR/21	25/APR/22
14	14	PARPO	14	25/APR/21	28/APR/22
15	15	PARPO	15	25/APR/21	30/APR/22
16	16	SWELL	16	10/MAY/21	17/MAY/23
17	17	SWELL	17	10/MAY/21	17/MAY/23
18	18	SWELL	18	10/MAY/21	17/MAY/23
19	19	SWELL	9	10/MAY/21	17/MAY/23
20	20	DJIPR	5	10/APR/22	10/APR/23
21	21	DJIPR	6	10/APR/22	10/APR/23
22	22	DJIPR	9	10/APR/22	10/APR/23

```
SQL> SELECT count(*)
  2  FROM drone.cust_train;

  COUNT(*)
----------
        22
```

```
SQL> SELECT cust_id, COUNT(*) AS
no_courses_taken
  2  FROM drone.cust_train
  3  GROUP BY cust_id
  4  ORDER BY cust_id;

   CUST_ID NO_COURSES_TAKEN
---------- ----------------
         1                1
         2                1
         3                1
         4                1
         5                2
         6                2
         7                1
         8                1
         9                4
        10                1
        11                1
        12                1
        14                1
        15                1
        16                1
        17                1
        18                1

17 rows selected.
```

```
SQL> SELECT AVG(COUNT(*))
  2      AS average_no_courses_taken
  3  FROM drone.cust_train
  4  GROUP BY cust_id;

AVERAGE_NO_COURSES_TAKEN
------------------------
              1.29411765
```

MONASH University

Q2. List all customer ids and the number of times each customer has taken a specific course:

A.
```
select cust_id, train_code, count(*) as no_of_courses_taken
from drone.cust_train
group by cust_id
order by cust_id;
```

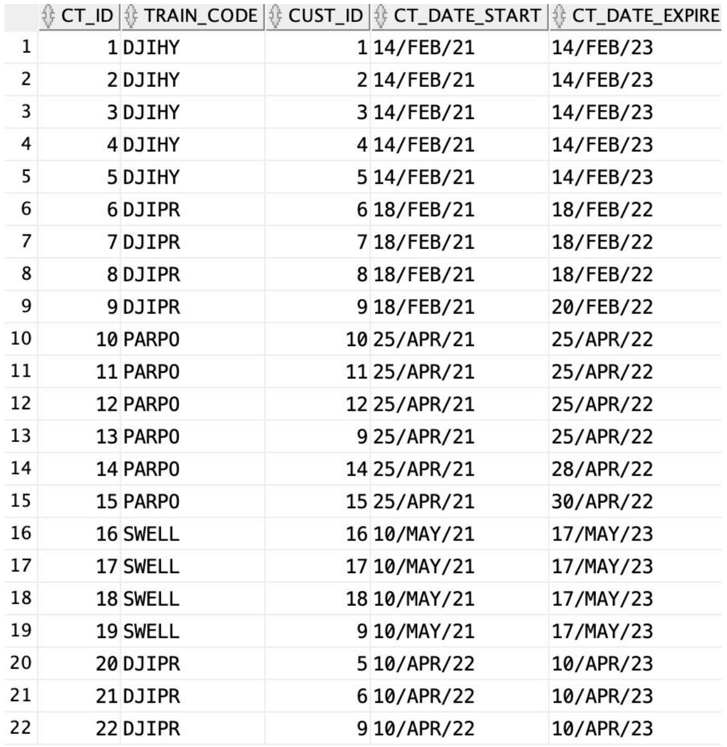
B.
```
select cust_id, train_code, count(*) as no_of_courses_taken
from drone.cust_train
group by cust_id, train_code
order by cust_id, train_code;
```

C.
```
select cust_id, count(*) as no_of_courses_taken
from drone.cust_train
group by train_code
order by train_code;
```

D. None of the above

What output is produced?

```
SELECT cust_id, train_code, count(train_code)
        as no_of_courses_taken
FROM drone.cust_train
GROUP BY cust_id, train_code
ORDER BY cust_id, train_code;
```



	CT_ID	TRAIN_CODE	CUST_ID	CT_DATE_START	CT_DATE_EXPIRE
1	1	DJIHY	1	14/FEB/21	14/FEB/23
2	2	DJIHY	2	14/FEB/21	14/FEB/23
3	3	DJIHY	3	14/FEB/21	14/FEB/23
4	4	DJIHY	4	14/FEB/21	14/FEB/23
5	5	DJIHY	5	14/FEB/21	14/FEB/23
6	6	DJIPR	6	18/FEB/21	18/FEB/22
7	7	DJIPR	7	18/FEB/21	18/FEB/22
8	8	DJIPR	8	18/FEB/21	18/FEB/22
9	9	DJIPR	9	18/FEB/21	20/FEB/22
10	10	PARPO	10	25/APR/21	25/APR/22
11	11	PARPO	11	25/APR/21	25/APR/22
12	12	PARPO	12	25/APR/21	25/APR/22
13	13	PARPO	9	25/APR/21	25/APR/22
14	14	PARPO	14	25/APR/21	28/APR/22
15	15	PARPO	15	25/APR/21	30/APR/22
16	16	SWELL	16	10/MAY/21	17/MAY/23
17	17	SWELL	17	10/MAY/21	17/MAY/23
18	18	SWELL	18	10/MAY/21	17/MAY/23
19	19	SWELL	9	10/MAY/21	17/MAY/23
20	20	DJIPR	5	10/APR/22	10/APR/23
21	21	DJIPR	6	10/APR/22	10/APR/23
22	22	DJIPR	9	10/APR/22	10/APR/23

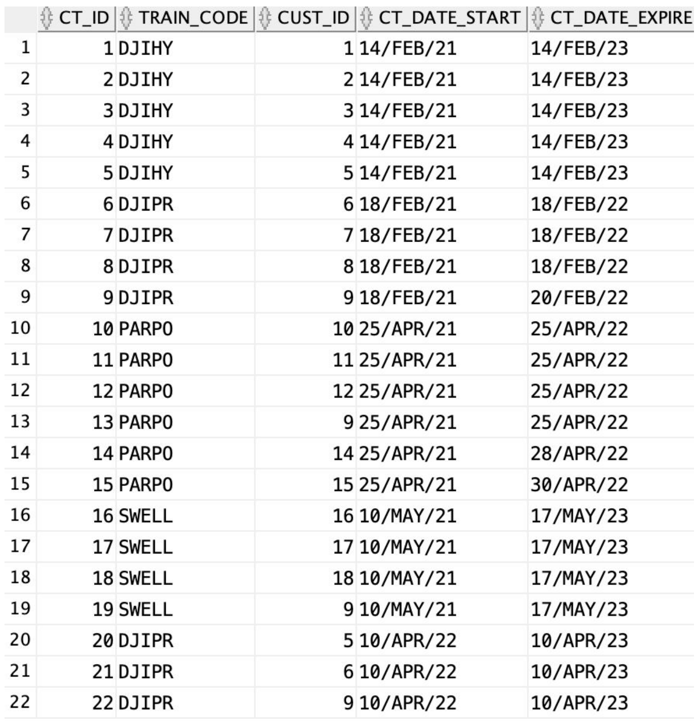
```
SQL> SELECT cust_id, train_code, count(train_code) as no_of_courses_taken
  2  FROM drone.cust_train
  3  GROUP BY cust_id, train_code
  4  ORDER BY cust_id, train_code;

   CUST_ID TRAIN NO_OF_COURSES_TAKEN
---------- ----- -------------------
         1 DJIHY                   1
         2 DJIHY                   1
         3 DJIHY                   1
         4 DJIHY                   1
         5 DJIHY                   1
         5 DJIPR                   1
         6 DJIPR                   2
         7 DJIPR                   1
         8 DJIPR                   1
         9 DJIPR                   2
         9 PARPO                   1
         9 SWELL                   1
        10 PARPO                   1
        11 PARPO                   1
        12 PARPO                   1
        14 PARPO                   1
        15 PARPO                   1
        16 SWELL                   1
        17 SWELL                   1
        18 SWELL                   1

20 rows selected.
```

What output is produced?

```
SELECT cust_id,
        to_char(ct_date_start, 'yyyy') as licence_start_year,
        count(train_code) as no_of_courses_taken
FROM drone.cust_train
GROUP BY cust_id, to_char(ct_date_start, 'yyyy')
ORDER BY cust_id, licence_start_year;
```

Note: column alias cannot be used in group by clause

WHY?



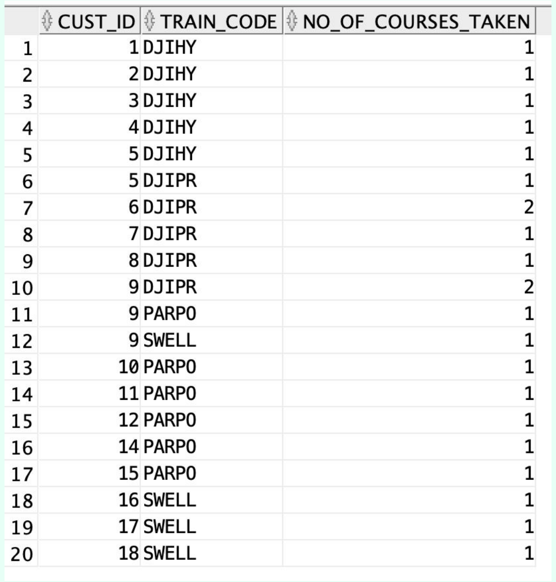
	CT_ID	TRAIN_CODE	CUST_ID	CT_DATE_START	CT_DATE_EXPIRE
1	1	DJIHY	1	14/FEB/21	14/FEB/23
2	2	DJIHY	2	14/FEB/21	14/FEB/23
3	3	DJIHY	3	14/FEB/21	14/FEB/23
4	4	DJIHY	4	14/FEB/21	14/FEB/23
5	5	DJIHY	5	14/FEB/21	14/FEB/23
6	6	DJIPR	6	18/FEB/21	18/FEB/22
7	7	DJIPR	7	18/FEB/21	18/FEB/22
8	8	DJIPR	8	18/FEB/21	18/FEB/22
9	9	DJIPR	9	18/FEB/21	20/FEB/22
10	10	PARPO	10	25/APR/21	25/APR/22
11	11	PARPO	11	25/APR/21	25/APR/22
12	12	PARPO	12	25/APR/21	25/APR/22
13	13	PARPO	9	25/APR/21	25/APR/22
14	14	PARPO	14	25/APR/21	28/APR/22
15	15	PARPO	15	25/APR/21	30/APR/22
16	16	SWELL	16	10/MAY/21	17/MAY/23
17	17	SWELL	17	10/MAY/21	17/MAY/23
18	18	SWELL	18	10/MAY/21	17/MAY/23
19	19	SWELL	9	10/MAY/21	17/MAY/23
20	20	DJIPR	5	10/APR/22	10/APR/23
21	21	DJIPR	6	10/APR/22	10/APR/23
22	22	DJIPR	9	10/APR/22	10/APR/23

```
SQL> SELECT cust_id,
  2  to_char(ct_date_start, 'yyyy') as licence_start_year, count(train_code) as no_of_courses_taken
  3  FROM drone.cust_train
  4  GROUP BY cust_id, to_char(ct_date_start, 'yyyy')
  5  ORDER BY cust_id, licence_start_year;

   CUST_ID LICE NO_OF_COURSES_TAKEN
---------- ---- -------------------
         1 2021                   1
         2 2021                   1
         3 2021                   1
         4 2021                   1
         5 2021                   1
         5 2022                   1
         6 2021                   1
         6 2022                   1
         7 2021                   1
         8 2021                   1
         9 2021                   3
         9 2022                   1
        10 2021                   1
        11 2021                   1
        12 2021                   1
        14 2021                   1
        15 2021                   1
        16 2021                   1
        17 2021                   1
        18 2021                   1
20 rows selected.
```

Q3. Which rows that will be returned by this select statement:

```
SELECT cust_id, train_code, count(train_code)
    as no_of_courses_taken
FROM drone.cust_train
GROUP BY cust_id, train_code
HAVING count(train_code) > 1
ORDER BY cust_id, train_code;
```

A. all rows
B. 7, 10
C. none of them
D. all rows except row 7 and 10



	CUST_ID	TRAIN_CODE	NO_OF_COURSES_TAKEN
1	1	DJIHY	1
2	2	DJIHY	1
3	3	DJIHY	1
4	4	DJIHY	1
5	5	DJIHY	1
6	5	DJIPR	1
7	6	DJIPR	2
8	7	DJIPR	1
9	8	DJIPR	1
10	9	DJIPR	2
11	9	PARPO	1
12	9	SWELL	1
13	10	PARPO	1
14	11	PARPO	1
15	12	PARPO	1
16	14	PARPO	1
17	15	PARPO	1
18	16	SWELL	1
19	17	SWELL	1
20	18	SWELL	1

HAVING clause

- It is used to put a condition or conditions on the groups defined by GROUP BY clause.

```
SELECT cust_id, train_code, count(train_code)
    as no_of_courses_taken
FROM drone.cust_train
GROUP BY cust_id, train_code
HAVING count(train_code) > 1
ORDER BY cust_id, train_code;
```

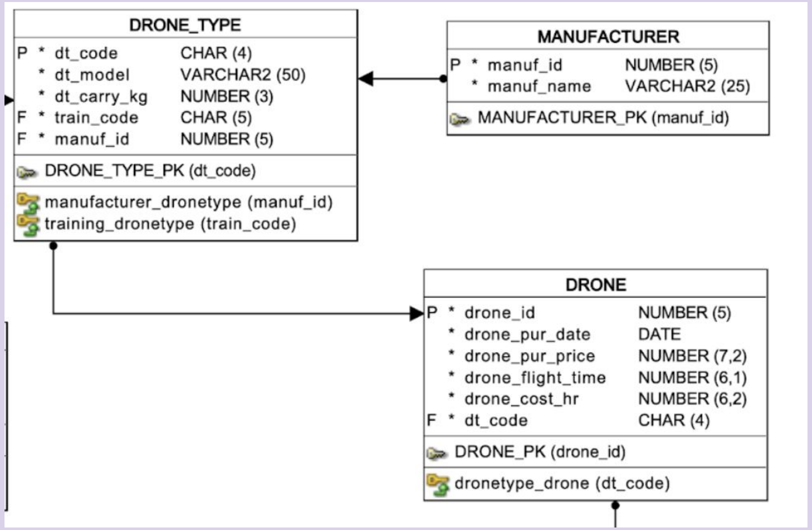
What output is produced?

```
SELECT cust_id, train_code, count(train_code) as no_of_courses_taken
FROM drone.cust_train
GROUP BY cust_id, train_code
HAVING count(train_code) > 1
ORDER BY cust_id, train_code;
```

```
SELECT dt_code, AVG(drone_flight_time) as average_drone_flight
FROM drone.drone
GROUP BY dt_code
HAVING AVG(drone_flight_time)>50
ORDER BY dt_code;
```

```
SQL> SELECT cust_id, train_code, count(train_code) as no_of_courses_taken
  2  FROM drone.cust_train
  3  GROUP BY cust_id, train_code
  4  HAVING count(train_code) > 1
  5  ORDER BY cust_id, train_code;

   CUST_ID TRAIN NO_OF_COURSES_TAKEN
---------- ----- -------------------
         6 DJIPR                   2
         9 DJIPR                   2

SQL> SELECT dt_code, AVG(drone_flight_time) as average_drone_flight
  2  FROM drone.drone
  3  GROUP BY dt_code
  4  HAVING AVG(drone_flight_time)>50
  5  ORDER BY dt_code;

DT_C AVERAGE_DRONE_FLIGHT
---- --------------------
DIN2           78.6666667
DMA2           53.3333333
PAPR               97.625
SWPS                 56.3
```

Q4. Write the SQL Query to report the average drone flight time for each type of drone. Display the average for only those types that have an average flight time of more than 50 minutes and for drones which were purchased in 2021.



HAVING and WHERE clauses

```
SELECT dt_code, AVG(drone_flight_time) as average_drone_flight
FROM drone.drone
WHERE to_char(drone_pur_date,'yyyy') = '2021'
GROUP BY dt_code
HAVING AVG(drone_flight_time)>50
ORDER BY dt_code;
```

- The WHERE clause is applied to ALL rows in the table.
- The HAVING clause is applied to the groups defined by the GROUP BY clause.
- The order of operations performed is FROM, WHERE, GROUP BY, HAVING and then ORDER BY.
- On the above example, the logic of the process will be:
    - All rows where drone purchase year = 2021 are retrieved. (due to the WHERE clause)
    - The retrieved rows then are grouped into different dt_code.
    - If the average flight time in a group is greater than 50, the dt_code and the average flight time is displayed. (due to the HAVING clause)

```
SQL> SELECT
  2      dt_code,
  3      AVG(drone_flight_time) AS average_drone_flight
  4  FROM
  5      drone.drone
  6  WHERE
  7      to_char(drone_pur_date, 'yyyy') = '2021'
  8  GROUP BY
  9      dt_code
 10  HAVING
 11      AVG(drone_flight_time) > 50
 12  ORDER BY
 13      average_drone_flight desc;

DT_C AVERAGE_DRONE_FLIGHT
---- --------------------
DIN2                  200
PAPR               97.625
DMA2                   80
SWPS                 56.3
```

```
SELECT cust_id, train_code, count(*) as no_of_courses_taken
FROM drone.cust_train
GROUP BY cust_id
ORDER BY cust_id;
```

The above SQL generates error message

```
SQL Error: ORA-00979: not a GROUP BY expression
00979. 00000 -  "not a GROUP BY expression"
```

Why and how to fix this?

- Why? Because the grouping is based on the cust_id, whereas the display is based on cust_id and train_code. The two groups may not have the same members.
- How to fix this?
  - Include the train_code as part of the GROUP BY condition.
- Attributes that are used in the SELECT, HAVING and ORDER BY must be included in the GROUP BY clause (reverse is not necessary).

Subqueries

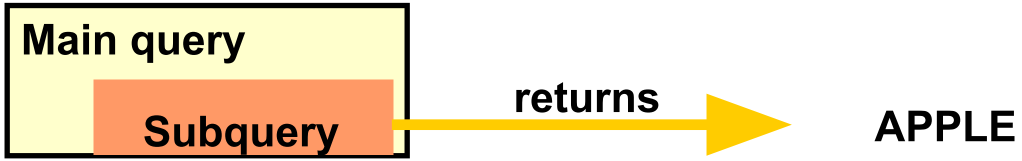
Query within a query.

"Find all drones which flight time is higher than the average flight time of all drones"

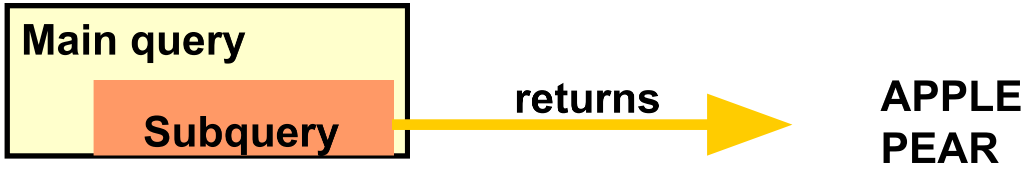
```
SELECT *
FROM drone.drone
WHERE drone_flight_time >
    (
        SELECT AVG(drone_flight_time)
        FROM drone.drone
    )
ORDER BY drone_id;
```

Types of Subqueries

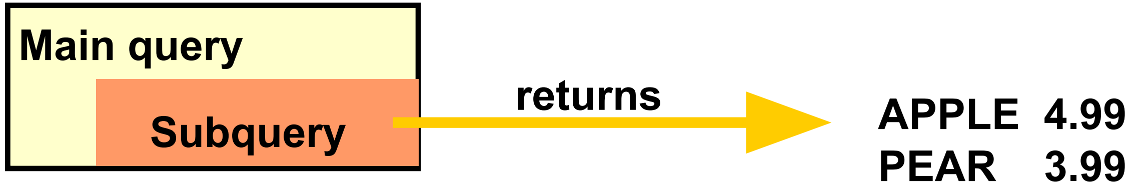
Single-value



Multiple-row subquery (a list of values – many rows, one column)



Multiple-column subquery (many rows, many columns)



Q5. What will be returned by the *inner query*?

```
SELECT *
FROM drone.drone
WHERE drone_pur_price > (SELECT AVG(drone_pur_price)
                         FROM drone.drone
                         GROUP BY drone_pur_date)
```

A. A value (a single column, single row).

B. A list of values.

C. Multiple columns, multiple rows.

D. None of the above.

```
SQL> SELECT
  2      *
  3  FROM
  4      drone.drone
  5  WHERE drone_pur_price > (SELECT AVG(drone_pur_price)
  6                           FROM drone.drone
  7                           GROUP BY drone_pur_date);

Error starting at line : 1 in command -
SELECT
    *
FROM
    drone.drone
WHERE drone_pur_price > (SELECT AVG(drone_pur_price)
                         FROM drone.drone
                         GROUP BY drone_pur_date)
Error report -
ORA-01427: single-row subquery returns more than one row
```

Q6. What will be returned by the *inner query*?

```
SELECT dt_code,dt_model,drone_id, drone_pur_price
FROM drone.drone_type natural join drone.drone
WHERE (dt_code, drone_pur_price) IN
                (SELECT dt_code, MAX(drone_pur_price)
                 FROM drone.drone_type NATURAL JOIN drone.drone
                 GROUP BY dt_code)
```

A. A value (a single column, single row).

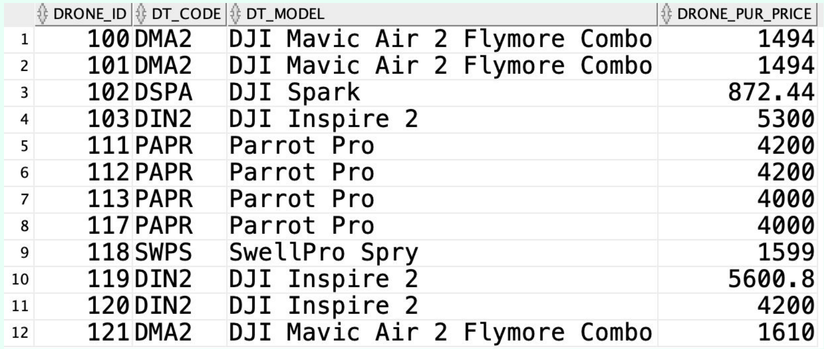
B. A list of values.

C. Multiple columns, multiple rows.

D. None of the above.

Comparison Operators for Subquery

- Operator for single value comparison.
  =, <, >
- Operator for multiple rows or a list comparison.
  - equality
    - IN
  - inequality
    - ALL, ANY combined with <, >



	DRONE_ID	DT_CODE	DT_MODEL	DRONE_PUR_PRICE
1	100	DMA2	DJI Mavic Air 2 Flymore Combo	1494
2	101	DMA2	DJI Mavic Air 2 Flymore Combo	1494
3	102	DSPA	DJI Spark	872.44
4	103	DIN2	DJI Inspire 2	5300
5	111	PAPR	Parrot Pro	4200
6	112	PAPR	Parrot Pro	4200
7	113	PAPR	Parrot Pro	4000
8	117	PAPR	Parrot Pro	4000
9	118	SWPS	SwellPro Spry	1599
10	119	DIN2	DJI Inspire 2	5600.8
11	120	DIN2	DJI Inspire 2	4200
12	121	DMA2	DJI Mavic Air 2 Flymore Combo	1610

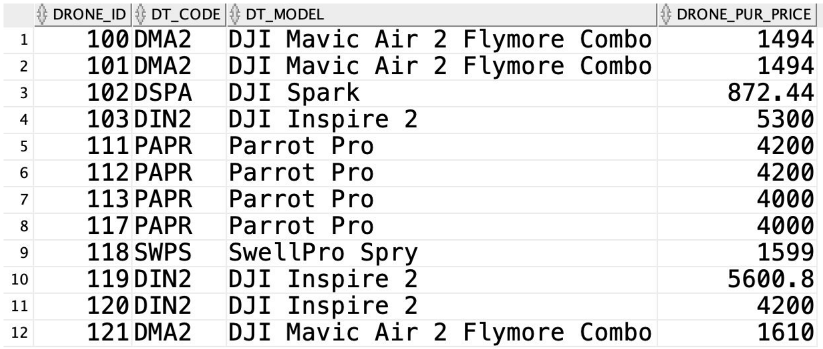
Q7. Which row(s) in the above table will be retrieved by the following SQL statement?

```
SELECT *
FROM dronetypeprice
WHERE drone_pur_price IN (SELECT MAX(drone_pur_price)
                                             FROM dronetypeprice GROUP BY dt_code)
```

A. 3,5,6,9,10,12

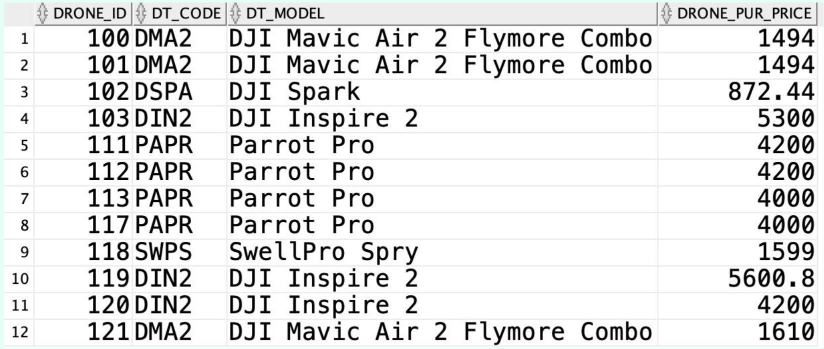
B. 10

C. 3,5,6,9,10,11,12

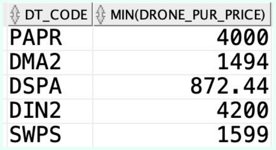


	DRONE_ID	DT_CODE	DT_MODEL	DRONE_PUR_PRICE
1	100	DMA2	DJI Mavic Air 2 Flymore Combo	1494
2	101	DMA2	DJI Mavic Air 2 Flymore Combo	1494
3	102	DSPA	DJI Spark	872.44
4	103	DIN2	DJI Inspire 2	5300
5	111	PAPR	Parrot Pro	4200
6	112	PAPR	Parrot Pro	4200
7	113	PAPR	Parrot Pro	4000
8	117	PAPR	Parrot Pro	4000
9	118	SWPS	SwellPro Spry	1599
10	119	DIN2	DJI Inspire 2	5600.8
11	120	DIN2	DJI Inspire 2	4200
12	121	DMA2	DJI Mavic Air 2 Flymore Combo	1610

```
SQL> SELECT
  2      *
  3  FROM
  4      dronetypeprice
  5  WHERE drone_pur_price IN (SELECT MAX(drone_pur_price)
  6                            FROM dronetypeprice
  7                            GROUP BY dt_code)
  8  order by drone_id;

  DRONE_ID DT_C DT_MODEL                       DRONE_PUR_PRICE
---------- ---- ------------------------------ ---------------
       102 DSPA DJI Spark                               872.44
       111 PAPR Parrot Pro                                4200
       112 PAPR Parrot Pro                                4200
       118 SWPS SwellPro Spry                             1599
       119 DIN2 DJI Inspire 2                           5600.8
       120 DIN2 DJI Inspire 2                             4200
       121 DMA2 DJI Mavic Air 2 Flymore Combo             1610
```



	DRONE_ID	DT_CODE	DT_MODEL	DRONE_PUR_PRICE
1	100	DMA2	DJI Mavic Air 2 Flymore Combo	1494
2	101	DMA2	DJI Mavic Air 2 Flymore Combo	1494
3	102	DSPA	DJI Spark	872.44
4	103	DIN2	DJI Inspire 2	5300
5	111	PAPR	Parrot Pro	4200
6	112	PAPR	Parrot Pro	4200
7	113	PAPR	Parrot Pro	4000
8	117	PAPR	Parrot Pro	4000
9	118	SWPS	SwellPro Spry	1599
10	119	DIN2	DJI Inspire 2	5600.8
11	120	DIN2	DJI Inspire 2	4200
12	121	DMA2	DJI Mavic Air 2 Flymore Combo	1610



DT_CODE	MIN(DRONE_PUR_PRICE)
PAPR	4000
DMA2	1494
DSPA	872.44
DIN2	4200
SWPS	1599

Q8. Which row/s in the above table will be retrieved by the following SQL statement?

```
SELECT  *
FROM dronetypeprice
WHERE drone_pur_price >
    ANY (SELECT MIN(drone_pur_price)
        FROM dronetypeprice
        GROUP BY dt_code)
```

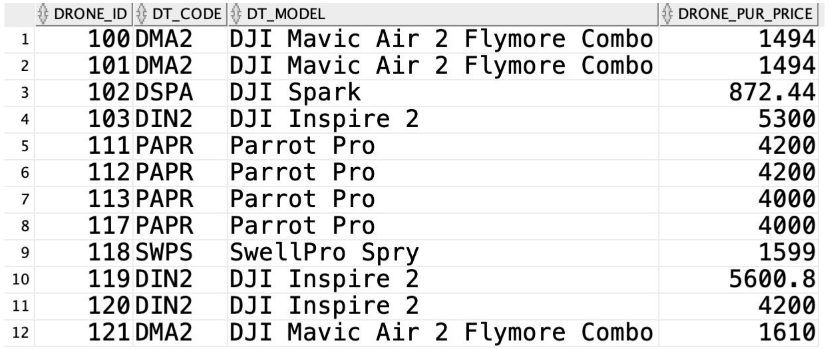
A. 10

B. 1,2,4,5,6,7,8,9,10,11,12

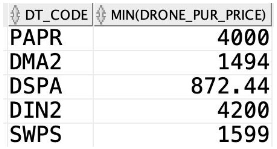
C. 4,10

D. No rows will be returned

MONASH University

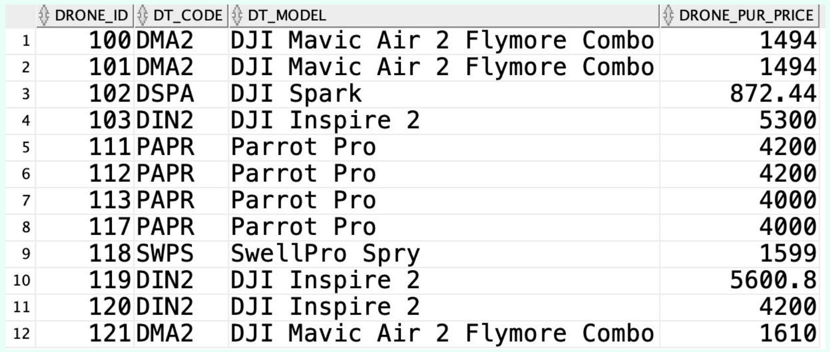


	DRONE_ID	DT_CODE	DT_MODEL	DRONE_PUR_PRICE
1	100	DMA2	DJI Mavic Air 2 Flymore Combo	1494
2	101	DMA2	DJI Mavic Air 2 Flymore Combo	1494
3	102	DSPA	DJI Spark	872.44
4	103	DIN2	DJI Inspire 2	5300
5	111	PAPR	Parrot Pro	4200
6	112	PAPR	Parrot Pro	4200
7	113	PAPR	Parrot Pro	4000
8	117	PAPR	Parrot Pro	4000
9	118	SWPS	SwellPro Spry	1599
10	119	DIN2	DJI Inspire 2	5600.8
11	120	DIN2	DJI Inspire 2	4200
12	121	DMA2	DJI Mavic Air 2 Flymore Combo	1610

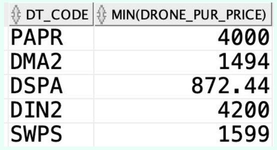


DT_CODE	MIN(DRONE_PUR_PRICE)
PAPR	4000
DMA2	1494
DSPA	872.44
DIN2	4200
SWPS	1599

```
SQL> SELECT *
  2  FROM dronetypeprice
  3  WHERE drone_pur_price >
  4         ANY (SELECT MIN(drone_pur_price)
  5              FROM dronetypeprice
  6              GROUP BY dt_code)
  7  ORDER BY drone_id;

  DRONE_ID DT_C DT_MODEL                       DRONE_PUR_PRICE
---------- ---- ------------------------------ ---------------
       100 DMA2 DJI Mavic Air 2 Flymore Combo             1494
       101 DMA2 DJI Mavic Air 2 Flymore Combo             1494
       103 DIN2 DJI Inspire 2                             5300
       111 PAPR Parrot Pro                                4200
       112 PAPR Parrot Pro                                4200
       113 PAPR Parrot Pro                                4000
       117 PAPR Parrot Pro                                4000
       118 SWPS SwellPro Spry                             1599
       119 DIN2 DJI Inspire 2                           5600.8
       120 DIN2 DJI Inspire 2                             4200
       121 DMA2 DJI Mavic Air 2 Flymore Combo             1610
```



	DRONE_ID	DT_CODE	DT_MODEL	DRONE_PUR_PRICE
1	100	DMA2	DJI Mavic Air 2 Flymore Combo	1494
2	101	DMA2	DJI Mavic Air 2 Flymore Combo	1494
3	102	DSPA	DJI Spark	872.44
4	103	DIN2	DJI Inspire 2	5300
5	111	PAPR	Parrot Pro	4200
6	112	PAPR	Parrot Pro	4200
7	113	PAPR	Parrot Pro	4000
8	117	PAPR	Parrot Pro	4000
9	118	SWPS	SwellPro Spry	1599
10	119	DIN2	DJI Inspire 2	5600.8
11	120	DIN2	DJI Inspire 2	4200
12	121	DMA2	DJI Mavic Air 2 Flymore Combo	1610



DT_CODE	MIN(DRONE_PUR_PRICE)
PAPR	4000
DMA2	1494
DSPA	872.44
DIN2	4200
SWPS	1599

Q9. Which row/s in in the above table will be retrieved by the following SQL statement?

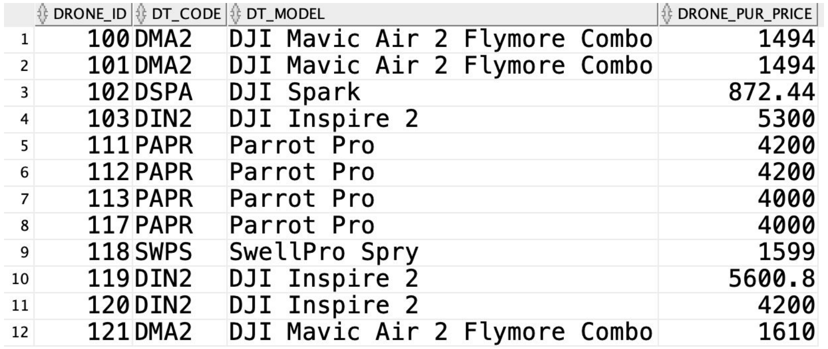
```
SELECT *
FROM dronetypeprice
WHERE drone_pur_price >
    ALL (SELECT MIN(drone_pur_price)
         FROM dronetypeprice
         GROUP BY dt_code)
ORDER BY drone_id;
```

A. 10

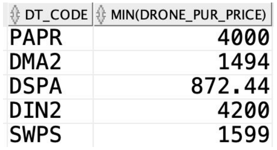
B. 1,2,4,5,6,7,8,9,10,11,12

C. 4,10

D. No rows will be returned



	DRONE_ID	DT_CODE	DT_MODEL	DRONE_PUR_PRICE
1	100	DMA2	DJI Mavic Air 2 Flymore Combo	1494
2	101	DMA2	DJI Mavic Air 2 Flymore Combo	1494
3	102	DSPA	DJI Spark	872.44
4	103	DIN2	DJI Inspire 2	5300
5	111	PAPR	Parrot Pro	4200
6	112	PAPR	Parrot Pro	4200
7	113	PAPR	Parrot Pro	4000
8	117	PAPR	Parrot Pro	4000
9	118	SWPS	SwellPro Spry	1599
10	119	DIN2	DJI Inspire 2	5600.8
11	120	DIN2	DJI Inspire 2	4200
12	121	DMA2	DJI Mavic Air 2 Flymore Combo	1610



DT_CODE	MIN(DRONE_PUR_PRICE)
PAPR	4000
DMA2	1494
DSPA	872.44
DIN2	4200
SWPS	1599

```
SQL> SELECT *
  2  FROM dronetypeprice
  3  WHERE drone_pur_price >
  4         ALL (SELECT MIN(drone_pur_price)
  5              FROM dronetypeprice
  6              GROUP BY dt_code)
  7  ORDER BY drone_id;

  DRONE_ID DT_C DT_MODEL         DRONE_PUR_PRICE
---------- ---- ---------------- ---------------
       103 DIN2 DJI Inspire 2               5300
       119 DIN2 DJI Inspire 2             5600.8
```

Q10. Write the SQL Query to find the details of all drones which have a purchase price less than the average purchase price for all drones manufactured by *DJI Da-Jiang Innovations*.

Begin by your listing the steps which need to be taken

After this code the SQL step by step.

Your output must show the drone id, the type code, the purchase price, the year purchased and the manufacturers name.

Order the output by drone id.

```
SELECT
    drone_id,
    dt_code,
    drone_pur_price,
    to_char(drone_pur_date,'yyyy') as yearpurchased,
    manuf_name
FROM
         drone.drone
    NATURAL JOIN drone.drone_type
    NATURAL JOIN drone.manufacturer
WHERE
    drone_pur_price < (
        SELECT
            AVG(drone_pur_price)
        FROM
                 drone.drone
            NATURAL JOIN drone.drone_type
            NATURAL JOIN drone.manufacturer
        WHERE
            upper(manuf_name) = upper('DJI Da-Jiang Innovations')
    )
ORDER BY
    drone_id;
```

Summary

- Aggregate Functions
  - –count, min, max, avg, sum
- GROUP BY and HAVING clauses.
- Subquery
  - –Inner vs outer query
  - –comparison operators (IN, ANY, ALL)