SUB QUERY WITH WHERE CLAUSE

STUDENTS

ID	Name	PSP	BATCHID
1	A	96	1
2	B	90	2
3	C	95	1
4	D	91	1
5	E	94	2

TEACHING ASSISTANT

ID	STUDENTID
1	2
2	3
3	4
4	1
5	NULL

EMPLOYEE

ID	NAME	DEPARTMENT	SALARY

Q1: Find all the students who have a PSP greater than PSP of student with id = 18?

```
SELECT S1.*
FROM STUDENTS S1
INNER JOIN STUDENTS S2
ON S1.PSP > S2.PSP
WHERE S2.ID = 18;

SELECT * FROM STUDENTS WHERE PSP > (SELECT PSP FROM STUDENT WHERE ID =
18);
```

Q2: Print the name of students who are TA as well.

```
SELECT * FROM STUDENTS WHERE ID IN (SELECT STUDENTID FROM TA WHERE
STUDENTID IS NOT NULL);
```

Q3: Select all students having PSP greater than students of batch 3?

```
SELECT * FROM STUDENTS WHERE PSP > (SELECT MAX(PSP) FROM STUDENTS GROUP BY
BATCHID HAVING BATCHID = 3);

SELECT * FROM STUDENTS WHERE PSP > ALL(SELECT PSP FROM STUDENTS WHERE
BATCHID=3 )
```

Q4: Select all the employees having salary more than all employees of department HR?

```
SELECT * FROM EMPLOYEES WHERE SALARY > (SELECT MAX(SALARY) FROM EMPLOYEES
GROUP BY DEPARTMENT HAVING DEPARTMENT = "HR");

SELECT * FROM EMPLOYEES WHERE SALARY > ALL(SELECT SALARY FROM EMPLOYEES
WHERE DEPARTMENT= "HR" );
```

Q5: Select any student having PSP greater than students of batch 3?

```
SELECT * FROM STUDENTS WHERE PSP > ANY(SELECT PSP FROM STUDENTS WHERE
BATCHID=3 )
```

CO RELATED SUB QUERIES

Q6: Select all the students whose PSP is greater than average PSP of their batch.

```
SELECT * FROM STUDENTS S WHERE PSP > (SELECT AVG(PSP) FROM STUDENTS GROUP
BY BATCHID HAVING BATCHID = S.BATCHID);
```

SUB QUERY WITH FROM CLAUSE

ID	Name	PSP	AVERAGE_PSP	BATCHID

```
SELECT * FROM STUDENT S;

SELECT AVG(PSP) AS AVERAGE_PSP FROM STUDENTS GROUP BY BATCHID HAVING
BATCHID = S.BATCHID


SELECT S.*, (SELECT AVG(PSP) AS AVERAGE_PSP FROM STUDENTS GROUP BY BATCHID
HAVING BATCHID = S.BATCHID
) FROM STUDENT S;
```

WHERE
Find products that are more expensive than lettuce(id=3)?

```
SELECT * FROM PRODUCTS
WHERE UNIT_PRICE > (
      SELECT UNIT_PRICE FROM PRODUCTS
      WHERE PRODUCT_ID = 3;
);
```

IN
Find the products that have never been ordered?

Products that have been ordered:
```
SELECT DISTINCT PRODUCT_ID FROM ORDER_ITEMS;
```

Products that have never been ordered:
```
SELECT * FROM PRODUCTS
WHERE PRODUCT_ID NOT IN (
      SELECT DISTINCT PRODUCT_ID
      FROM ORDER_ITEMS
);
```

USING JOINS
```
SELECT * FROM PRODUCTS
LEFT JOIN ORDER_ITEMS
USING(PRODUCT_ID)
WHERE ORDER_ITEMS_ID IS NULL;
```

MAX & ALL
Select all invoices larger than all invoices of client 3?

```
SELECT * FROM INVOICES
WHERE INVOICE_TOTAL > (
      SELECT MAX(INVOICE_TOTAL)
      FROM INVOICES
      WHERE CLIENT_ID = 3
);
```

USING ALL

```
SELECT * FROM INVOICES
WHERE INVOICE_TOTAL > ALL(150, 130, 167, 140);
```

DBMS will look at invoices table. For each row it will compare INVOICE_TOTAL with all the values (150, 130, 167, 140). If the INVOICE_TOTAL is greater than all these values, that row will be returned in the final result set.

```
SELECT * FROM INVOICES
WHERE INVOICE_TOTAL > ALL(
      SELECT INVOICE_TOTAL
      FROM INVOICES
      WHERE CLIENT_ID = 3
);
```

IN & ANY

Select CLIENTS WITH ATLEAST TWO INVOICES?

```
SELECT * FROM CLIENTS
WHERE CLIENT_ID IN(
      SELECT CLIENT_ID FROM INVOICES
      GROUP BY CLIENT_ID
      HAVING COUNT(*) >= 2
);

SELECT * FROM CLIENTS
WHERE CLIENT_ID = ANY(
      SELECT CLIENT_ID FROM INVOICES
      GROUP BY CLIENT_ID
      HAVING COUNT(*) >= 2
);
```

CORELATED SUBQUERIES

Select employees whose salary is greater than the average salary in their office?

```
SELECT * FROM EMPLOYEES E
WHERE SALARY > (
      SELECT AVG(SALARY) FROM EMPLOYEES
      WHERE OFFICE_ID = E.OFFICE_ID
);
```

EXISTS

Select clients that have an invoice?

```
SELECT * FROM CLIENTS C
WHERE CLIENT_ID IN(
      SELECT DISTINCT CLIENT_ID FROM INVOICES
);

SELECT * FROM CLIENTS C
WHERE CLIENT_ID = ANY(
      SELECT DISTINCT CLIENT_ID FROM INVOICES
);
```

Better Option

```
SELECT * FROM CLIENTS C
WHERE EXISTS (
      SELECT CLIENT_ID FROM INVOICES
      WHERE CLIENT_ID = C.CLIENT_ID
);
```

SUBQUERIES IN SELECT

```
SELECT INVOICE_ID, INVOICE_TOTAL,
      (SELECT AVG(INVOICE_TOTAL) FROM INVOICES) AS INVOICE_AVERAGE,
      INVOICE_TOTAL - (SELECT INVOICE_AVERAGE) AS DIFFERENCE
FROM INVOICES;
```

SUBQUERIES IN FROM

```
SELECT * FROM (
      SELECT INVOICE_ID, INVOICE_TOTAL,
            (SELECT AVG(INVOICE_TOTAL) FROM INVOICES) AS INVOICE_AVERAGE,
            INVOICE_TOTAL - (SELECT INVOICE_AVERAGE) AS DIFFERENCE
      FROM INVOICES;
) AS SALES_SUMMARY;
```