```
select *
from student
where name = 'Honny';

-- Go through the query stats.
-- Execution Plan
-- Full Table Scan
-- Optimizer calculates the Query Cost and based on the query cost, the decision will be taken on which query to execute.
-- query cost = 3.75

select *
from student
where id = 10;
-- query cost = 1.0

-- A table is by default sorted by its PK.
-- Every table will have an index on the PK by default.
-- Single Row.

ALTER table student add index student_name_idx (name);

select *
from student
where name = 'Honny';

-- Query cost = 0.35
-- scan -> Non Unique Key lookup.

select o.order_id, o.shipped_date, c.customer_id, c.first_name, oi.product_id, oi.quantity,
            p.name, p.unit_price
from orders o
join customers c
on o.customer_id = c.customer_id
join order_items oi
on o.order_id = oi.order_id
join products p
on oi.product_id = p.product_id;

-- orders customers order_items products ->

-- Joins are very costly.
-- JOINs -> read
--
```

```
-- O(N^3) -> Most optimal solution.
-- O(N^2) -> O(NlogN) -> O(N)

-- As a assignment, try to change the order of joins and then analyse the cost.

select o.order_id, o.shipped_date, c.customer_id, c.first_name, oi.product_id, oi.quantity
from orders o
join customers c
on o.customer_id = c.customer_id
join order_items oi
on o.order_id = oi.order_id;

select * from orders;


-- A join B
-- for every row A : O(N)
--          for every row of B : -> O(N) worst
                    -- check the condition ON a.batch_id = b.batch_id
                            -- if the condition satisfies -> add in the output.
```