Copy data to create independent arrays:

High Performance

var numbers2=[...numbers]

Low Performance

var numbers2=JSON.parse(JSON.stringify(numbers))

var numbers2=numbers.concat()

var numbers2=numbers.slice()

splice will remove all the elements from array1.

var numbers2=numbers.splice(0)

**Iterate through the object:**

for(itr in object){

// object[itr]

}

**Conversions**:

Number(value),

parseInt(value),

parseFloat(value),

value.toString()

+value // converts string into number

-value // converts string to number

**------------------------------------------------------SQL---------------------------------------------**

-- table cars

/\*

1 "swift" "maruti & suzuki" "2018" "1000000"

3 "alto" "maruti & suzuki" "2019" "400000"

4 "polo" "volkswagon" "2015" "700000"

6 "laura" "skoda" "2017" "1500000"

2 "swift s-cross" "maruti & suzuki" "2017" "900000"

5 "vento" "volkswagon" "2018" "900000"

\*/

-- update public.cars

-- set year='2017'

-- where id=2;

-- select \* from public.cars

-- delete from public.cars

-- where id=3;

-- insert into public.cars

-- values(4,'polo','volkswagon','2015',700000),

-- (5,'vento','volkswagon','2016',900000),

-- (6,'laura','skoda','2017',1500000)

-- select \* from public.cars where model like '\_\_t%';

-- select \* from public.cars limit 3;

-- select \* from public.cars order by year desc,id ;

-- select year,sum(price) as sum\_price from public.cars where year='2018' group by year;

-- select distinct year from public.cars;

-- select year,model from public.cars except select year,model from public.cars;

-- select year,model from public.cars intersect select year,model from public.cars;

-- select year,model from public.cars union select year,model from public.cars;

-- select year,model from public.cars union all select year,model from public.cars;

-- select \* from public.cars where year is null;

-- select \* from public.cars where year is not null;

-- select id,model as Car\_Model, brand as Car\_Brand from public.cars ;

-- create table sales(

-- id int not null,

-- sold decimal not null,

-- primary key (id));

-- insert into public.sales values

-- (2015,1000000),(2017,3000000),(2018,2000000),(2019,500000);

-- select \* from public.sales;

/\*

create table sales\_count(

id int not null,

count int not null,

primary key (id)

);

insert into sales\_count values

(2015,3),

(2017,5),

(2018,4),

(2019,1);

select \* from public.sales\_count;

\*/

-- select \* from public.cars as cars ,public.sales as sales where cars.year=sales.id;

-- select \* from public.cars as cars left join public.sales as sales on cars.year=sales.id;

-- select \* from public.cars as cars full join public.sales as sales on cars.year=sales.id;

-- select \* from public.cars as cars1 full join public.cars as cars2 on cars1.year=cars2.year;

-- create view cars\_view as select id,model,year,price from public.cars;

-- select year, sum(price) from public.cars group by year having sum(price)>1300000;

-- create table cars\_temp as select \* from public.cars;

-- select \* from cars\_temp;

-- select \* from public.cars where year in (select year from public.cars where price > 100000);

DBLink:

SELECT \*  
FROM student st INNER JOIN  
dblink(  
    'dbname=pginit\_nats port=5432 host=localhost user=postgres password=root',  
    'SELECT id, name, student\_id FROM subject'  
)  
AS su(id INT, "name" VARCHAR,student\_id INT)  
ON st.id = su.student\_id;