```
select * from customer_table ;

alter table customer_table add test varchar(255)

alter table customer_table drop test ;

alter table customer_table drop column test ;

alter table customer_table alter column age type varchar(255) ;

alter table customer_table rename column first_nam to first_name ;

alter table customer_table alter column cust_id set not null ;

insert into customer_table(first_name, last_name, age, email_id) values
('aa', 'bb', '25' , 'abc@xyz.com') ;

alter table customer_table alter column cust_id drop not null ;

DELETE FROM customer_table WHERE cust_id IS NULL;

alter table customer_table add constraint cust_id check (cust_id>0) ;

alter table customer_table add primary key (cust_id) ;

UPDATE Science_class SET science_marks = 45 WHERE enrollment_no = 1;


select * from customer ;

select * from product ;
```

```
select * from sales ;

select * from customer where city in ('Philadelphia', 'Seattle') ;

select * from customer where city = 'Philadelphia' OR city ='Seattle' ;

select * from customer where age between 20 and 30 ;

select * from customer where age >= 20 and age <= 30 ;

select * from customer where age not between 20 and 30 ;

/*
multi line comments
*/
--single line comments
/* Like Comments*/

/*
Here % and _ is called as wild card
% represents n number of charachters
_ represents one charachter
*/

select * from customer where customer_name like 'J%';

select * from customer where customer_name like '%Nelson%';

select * from customer where customer_name like '____ %';
/*customer has 4 words in firtst name and with 'n' number of words as second name */
```

```
select * from customer where city not like 'S%';

select * from customer where customer_name like 'G\%';
/*Here the \ is know as the escape charachter which treats % as a charachter and not as wild card*/

/*exercise*/
select * from customer order by customer_name asc ;

select distinct city from customer where region in ('Central', 'East');

select * from sales where sales between 100 and 500 ;

select distinct customer_name from customer where customer_name like '% ____';

/* oder by */

select * from customer where state = 'California' order by customer_name;

select * from customer where state = 'California' order by customer_name asc;

select * from customer where state = 'California' order by customer_name desc;

select * from customer order by city asc, customer_name desc;

select * from customer where state = 'California' order by city asc, customer_name desc;
/*
ORDER BY 2 DESC;
where 2 indicates the column number without specifying te name of the column
*/
```

```
select * from customer order by 2 asc;

select * from customer order by age desc;

/*Limiting number of return outputs*/

select * from customer where age >= 25 order by age desc limit 8;

select * from customer where age > 25 order by age asc limit 10;

/* exercise */

select * from sales limit 5 ;

select * from sales where discount > 0 order by discount desc ;

select * from sales where discount > 0 order by discount desc limit 10;

/* AS alias
provides second name for the column name or table name
*/

select customer_id as "Serial Number", customer_name as "Name", age as "Customer_age" from customer ;

select customer_id as "Serial Number", customer_name as Name, age as Customer_age from customer ;

/* "" is used for names with spaces or to retain the capitial initials
if not used then the names are named as small letters like Name will be changed to name
run this querry for better understanding
*/
```

```
/* COUNT */

select count(*) from sales ;

select count(order_line) as "Number Of Products Ordered", count (distinct order_id) as "Number of Orders" from sales where customer_id = 'CG-12520';

/* SUM */

select sum(profit) as "Total Profit" from sales ;

select sum(quantity) as  "Total Quantity" from sales where product_id = 'FUR-TA-10000577' ;

/* AVERAGE */

select avg(age) as "Average Customer Age" from customer ;

select avg(sales * 0.10) as "Average Commision Value" from sales ;

/* MIN MAX */

select min(sales) as "Minimum Sales Value June" from sales where order_date between '2015-06-01' and '2015-06-30' ;

select sales from sales where order_date between '2015-06-01' and '2015-06-30' order by sales asc ;

select max(sales) as "Maximun Sales Value June" from sales where order_date between '2015-06-01' and '2015-06-30' ;

/* exercise */
```

```
select sum(sales) as "Total Sales" from sales ;

select count(distinct customer_id) from customer where age between 20 and 30 ;

select avg(age) as "Average Age of Customers In East Region" from customer where region in ('East')
;

select min(age) as "Minimum Age of Customer", max(age) as "Maximum Age Of Customer" from
customer where city in ('Philadelphia');

select min(age) as "Minimum Age of Customer", max(age) as "Maximum Age Of Customer" from
customer where city like ('P%a');

/* GROUP BY */

select region, count(customer_id) as "Customer Count" from customer group by region ;

select product_id, sum(quantity) as "Quantity Sold" from sales group by product_id order by
"Quantity Sold" desc ;

/* ALL QUEREY LEARNED SO FAR */

select customer_id, min(sales) as "Minimun Sales", max(sales) as "Maximun Sales", avg(sales) as
"Average Sales", sum(sales) as "Total Sales" from sales group by customer_id order by "Total Sales"
desc limit 5 ;

/* HAVING */

select region, count(customer_id) as "Customer Count" from customer group by region having
count(customer_id)>200;

select region, count(customer_id) as "Customer Count" from customer where customer_name like
'A%' group by region ;
```

```
select region, count(customer_id) as "Customer Count" from customer where customer_name like 'A%' group by region having count(customer_id) >15;


/* exercise */

select * from sales limit 1 ;


select * from customer limit 1 ;


select sum(sales) as "Total sales", sum(quantity) as "Total quantity", count(order_id) as "Number of Orders", max(sales) as "Max Sales Value", min(sales) as "Min Sales Value" , avg(sales) as "Average Sales Value" from sales ;


select product_id, count(product_id) as "List Of Product IDs" from sales group by product_id having count(quantity) > 10 ;


/* CASE EXPRESSIONS */


SELECT *,
        CASE WHEN age < 30 THEN 'Young'
                WHEN age > 60 THEN 'Citizen'
                ELSE 'Middle Aged'
                END AS Age_Category
        FROM customer ;


/* JOINS */


/*Creating sales table of year 2015 */

/* Creating table with customre age between 20 and 30 */



/*Creating sales table of year 2015*/
```

```
Create table sales_2015 as select * from sales where ship_date between '2015-01-01' and '2015-12-31';

select count(*) from sales_2015; --2131

select count(distinct customer_id) from sales_2015;--578


/* Customers with age between 20 and 60 */

create table customer_20_60 as select * from customer where age between 20 and 60;

select count (*) from customer_20_60;--597


/* INNER JOIN || (A n B)

Gives the Intersection of two tables

*/


select

	a.order_line,

	a.product_id,

	a.customer_id,

	a.sales,

	b.customer_name,

	b.age

from sales_2015 as a

inner join customer_20_60 as b

on a.customer_id = b.customer_id

order by customer_id ;


/* LEFT JOIN || (A U B')


outputs all of A and intersection of A and B


*/
```

```
select
	a.order_line,
	a.product_id,
	a.customer_id,
	a.sales,
	b.customer_name,
	b.age
from sales_2015 as a
left join customer_20_60 as b
on a.customer_id = b.customer_id
order by customer_id ;

/* RIGHT JOIN || (A' U B)

outputs all of B and intersection of A and B
to get all the values of right table, always remember to select the common coulmn
name from the right table rather than from the left table in right join

*/

select
	a.order_line,
	a.product_id,
	b.customer_id, --selecting the common coulmn from the right join table
	a.sales,
	b.customer_name,
	b.age
from sales_2015 as a
right join customer_20_60 as b
on a.customer_id = b.customer_id
```

```
order by customer_id ;

/* FULL JOIN || (A U B)
Like union join but different
It adds the extra colum and null rows to the output table
And full join could be performed only when there is atleast one relation between the two or more tables
*/

select
        a.order_line,
        a.product_id,
        a.customer_id,
        a.sales,
        b.customer_name,
        b.age,
        b.customer_id
from sales_2015 as a
full join customer_20_60 as b
on a.customer_id = b.customer_id
order by a.customer_id, b.customer_id ;

/* CROSS JOIN
creates cartesian product between two sets of data
*/

create table month_values (MM integer) ;
create table year_values (YYYY integer) ;

insert into month_values values (1), (2), (3),(4), (5), (6),(7), (8), (9), (10), (11),(12);
insert into year_values values (2011), (2012), (2013),(2014), (2015), (2016),(2017), (2018), (2019);
```

```
select * from month_values ;

select * from year_values ;


select a.YYYY , b.MM
        from year_values as a, month_values as b
        order by a.YYYY, b.MM ;



/* EXCEPT || (A n B')
output contains values of A exculded of the values common to A and B
*/


select customer_id
        from sales_2015
        except select customer_id from customer_20_60
        order by customer_id ;


/* UNION || (A U B)
The union adds the extra rows in the output if there is a common coulumn in both or relse it adds the coulum in output
It avoids the null data in the row or column I guess
Union can be used for 2 or more tables with or without common links between them
*/


select customer_id
        from sales_2015
        union select customer_id from customer_20_60
        order by customer_id;


/* exercise */
```

```
select * from sales _2015 limit 1;

select * from customer_20_60 limit 1;

select b.state , sum(sales) as total_sales
from sales_2015 as a left join customer_20_60 as b
on
a.customer_id = b.customer_id
group by
b.state ;


select * from sales limit 1;

select * from product;

select
a.*, sum( b.sales ) as total_sales , sum(quantity) as total_quantity
from product as a left join sales as b
on
a.product_id = b.product_id
group by
a.product_id

/* SUBQUERY */

select * from sales where customer_id in (select customer_id from customer where age > 60) ;

select
        a.product_id,
```

```
        a.product_name,
        a.category,
        b.quantity
        from product as a
        left join (select product_id, sum(quantity) as quantity from sales group by product_id) as b
        on a.product_id = b.product_id
        order by b.quantity desc ;



select customer_id, order_line,
        (select customer_name from customer where customer.customer_id = sales.customer_id)
        from sales
        order by customer_id ;

/* notes on subquery
subquery takes more time and resourses from database than joins
subquery must be enclosed within paranthesis
ulinke 'in' command, 'between' command cannot be used between query and subquery but
it can be used within subquery
*/

/* exercise */

select
        a.*,
        b.customer_name,
        b.age,
        b.product_name,
        b.category
        from sales as a
        left join (select
```

```
                    c.customer_name,
                    c.customer_id,
                    c.age,
                d.product_name,
                    d.category
                from customer as c
                full join product as d
                on c.customer_id = d.product_id ) as b
        on a.customer_id = b.customer_id ;
```

```
select
c.customer_name , c.age , sp.* from
customer as c
right join (select s.*,
p.product_name , p.category
from sales as s
left join product as p
on
s.product_id = p.product_id ) as sp
on
c.customer_id = sp.customer_id
```

```
/* VIEWS
used to control the data necessary for the team to visible
instead of creating a new table we are creating an instance of a table (i guess)
*/
```

```
create view logistics as
select a.order_line,a.order_id,b.customer_name,b.city,b.state,b.country
from sales as a
left join customer as b
on a.customer_id = b.customer_id
order by a.order_line ;

select * from logistics ;

/* create or replace view logistics as
this commandis best to use because in case if the instance of table has already
been created this command is used to update the instance as we need it
*/

drop view logistics ; --delete the view

--we can also update a view with new values but is not adviseable

/* INDEX */
/* Single column index is called as simple index
more than 1 coulmn index is known as composite index
*/
create index mon_idx
on month_values(MM);

/* Droping Index*/
/* drop index 'if exist command' is used to check wheather the index is created or not
if created it deletes the index else does not throw an error
advisable to use for droping anything (tables)
*/
/* CASCADE it deletes the dependable database object that relies on this index
```

```
*/
/* RESTICT throws an error if that index is used by other database
*/

drop index mon_idx ;

/* ALTER INDEX is used to rename a index
Syntax : Alter index [if exists] index_name, rename to new_index_name;
*/
/* good practices : it is best to create an index with column that has an integer type
because it requires less space compared to other index
*/

/* Exercise */

select * from sales order by order_date desc limit 5 ;

create view Daily_Billing as select order_line,product_id,sales,discount from sales where order_date
= '2017-12-30' ;

select * from Daily_Billing ;

drop view if exists Daily_Billing ;

create or replace view Daily_Billing as select order_line , order_date, product_id , sales, discount
from sales where order_date in (select max( order_date ) from sales ) ;

/* LENGTH (STRING) */

/* LENGTH (STRING)
```

returns number of characters in the string

*/

```
select customer_name, length(customer_name) as characters
from customer where age > 30 ;

select customer_name, length(customer_name) as characters
from customer where length(customer_name) > 15 order by characters desc;

/* UPPER and LOWER */

select upper('Start-Tech Achademy') ;

select lower ('Start-Tech Achademy') ;

/* REPLACE */
/* It is case sensitive. i.e searches for text with same mentioned text */

select customer_name, country, replace(country, 'United States', 'US') as country_new from
customer ;

select customer_name, country, replace(lower(country), 'united states', 'US') as country_new from
customer ;

/* TRIM, RTRIM, LTRIM */

select trim(leading ' ' from ' Start-Tech Academy ');

select trim(trailing ' ' from ' Start-Tech Academy ');

select trim(both ' ' from ' Start-Tech Academy ');
```

```
select trim(' Start-Tech Academy ');

select rtrim(' Start-Tech Academy ');

select rtrim(' Start-Tech Academy ', ' ');

select ltrim(' Start-Tech Academy ', ' ');

/* CONCAT (operator)
it joins several strings using double pipe operator '||'
*/

select customer_name, city || ' , '||state||' , '||country||' , ' as address from customer;

/* SUBSTRING */
/* used to take second string from a primary string */

select customer_id, customer_name, substring (customer_id for 2) as cust_group
from customer
where substring (customer_id for 2) = 'AB' ;


select customer_id, customer_name, substring (customer_id from 4 for 5) as cust_number
from customer
where substring (customer_id for 2) = 'AB' ;

/* STRING AGGREGATOR */

select order_id, string_agg(product_id, ' , ')
from sales
group by order_id
```

```
order by order_id;

/* Exercise */

select product_name, length(product_name) from product order by length(product_name) desc;

select *, product_name || ' , '||sub_category|| ' , '||category as product_detais from product ;

select product_id,substring(trim(product_id,' ') for 3) as catg,substring(product_id from 5 for 2) as
sub_catg, substring(product_id from 8 for 8) as id
from product ;

select product_name, sub_category from product ;

select sub_category, string_agg(product_name, ' , ')
from product where sub_category in ('Chairs', 'Tables')
group by sub_category
order by sub_category;

/* SOLUTIONS */

select max(length(
product_name )) from product ;

select product_name,sub_category , category , (product_name ||' , '||sub_category ||' ,
'||category) as product_details from product;

select product_id , substring(product_id for 3) as category_short , substring(product_id from
5 for 2) as sub_short , substring(product_id from 8) as id from product;
```

```
select string_agg (product_name ,', ') from product where sub_category in (' Chairs','Tables') ;

/* CEIL & FLOOR */
/* integer : 4.1
ceil 4.1 gives 5 and floor 4.1 gives 4
*/

select order_line, sales, ceil(sales) as first_max_round_off_sale,  floor(sales) as
first_min_round_off_sale from sales ;

/* RANDOM */
/* generally generates a random number between >=0 and <1 */
/* can be multiplied with a range
like random() between 10 to 50 means (50-10)+10
*/

select random() ;

select random()*(20-10)+10 ;

select floor(random()*(40)+10), ceil(random()*(50-10)+10) ;

/* SETSEED */
/* value can be from -1 to +1 */

select setseed (0.5);

select random(); --gives a

select random(); --gives b
```

```
select setseed (0.5);

select random(); --gives same a as before

select random(); --gives same b as before

/* ROUND */

select round(11.1); --like floor

select round(11.5); --like ceil

select round(11.9); --like ceil

/* POWER */

select power(6,2); -- gives 6² (6 power 2);

/* exercise */

select customer_id , random() as rand_n from customer order by rand_n limit 5;

select sum(ceil(sales)) as
higher_int_sales , sum(floor(sales)) as lower_int_sales ,
sum(round(sales)) as round_int_sales from sales;

/* CURRENT DATE & TIME */

select current_date, current_time, current_time(1), current_timestamp ;

/* AGE */
```

```
select age(current_date,'1996-07-24') ;

/* EXTRACT */

select extract(year from age(current_date,'1996-07-24')) ;

select extract(month from age(current_date,'1996-07-24')) ;

select extract(day from age(current_date,'1996-07-24')) ;

select extract(minute from current_timestamp) ;

select order_date,ship_date, (extract(epoch from (ship_date))- extract(epoch from (order_date))) as
sec_taken from sales ;

/* exercise */

select age(current_date,'1939-04-06') as age_of_batman;

select * from sales limit 1 ;

select extract(month from (order_date)) as sale_month, sum(sales) from sales group by sale_month
order by sale_month ;

select b.sub_category, extract(month from (order_date)) as sale_month, sum(sales)
        from sales as a
        left join product as b
        on a.product_id = (select product_id from product where sub_category = 'Chairs')
        group by b.sub_Category, sale_month order by sale_month ; --wrong, it doesn' work like
this. refer below solution
```

```
select extract(month from order_date ) as month_n , sum(sales) as total_sales from sales
where product_id in (select product_id from product where sub_category = 'Chairs')
group by month_n order by month_n ;

/* PATTERN MATCHING */

select * from customer where customer_name ~* '^a+[a-z\s]+$' ;

select * from customer where customer_name ~* '^(a|b|c|d)+[a-z\s]+$' order by customer_name;

select * from customer where customer_name ~* '^(a|b|c|d)+[a-z]{3}\s[a-z]{4}$' order by
customer_name;
```