

Insert and Update

1. **Create** a table with the following parameters:

- CustomerID
- CustomerName
- Address
- City
- PostalCode
- Country
- Email

```
CREATE TABLE customer (  
customer_id SERIAL PRIMARY KEY,  
customer_name VARCHAR (255) NOT NULL,  
address VARCHAR(255)NOT NULL,  
city VARCHAR(255),  
postal_code int(9),  
country VARCHAR(55),  
email VARCHAR(55)  
);
```

2. Insert 3 rows of data into these columns using **INSERT**. The data you insert should make sense for the column.

- 3.

```
INSERT INTO customer (customer_name, address, city, postal_code, country, email)  
VALUES
```

```
('Marta', '12398 35 avenue', 'Sault Lake', 11254, 'USA', 'marta@gmail.com'),
```

```
('Alan', '12347 34 avenue', 'Portland', 15754, 'USA', 'alan@gmail.com'),
```

```
('Carlos', '126977 32 avenue', 'Greenport', 11274, 'USA', 'carlosa@gmail.com');
```

Result Grid Filter Rows: Export: Wrap Cell Content:							
	customer_id	customer_name	address	city	postal_code	country	email
▶	1	Marta	12398 35 avenue	Sault Lake	11254	USA	marta@gmail.com
	2	Alan	12347 34 avenue	Portland	15754	USA	alan@gmail.com
	3	Carlos	126977 32 avenue	Greenport	11274	USA	carlosa@gmail.com

4. Use an **UPDATE** to modify any portion of the data

UPDATE customer SET city = 'NewPort' WHERE customer_id = 3;

Result Grid

Filter Rows:

Export:



Wrap Cell Content:


	customer_id	customer_name	address	city	postal_code	country	email
▶	1	Marta	12398 35 avenue	Sault Lake	11254	USA	marta@gmail.com
	2	Alan	12347 34 avenue	Portland	15754	USA	alan@gmail.com
	3	Carlos	126977 32 avenue	NewPort	11274	USA	carlosa@gmail.com


5. Finally, write a statement to **delete** one row of data.

DELETE FROM customer WHERE customer_id = 1;

Result Grid



Filter Rows:

Export:


Wrap Cell Content:


	customer_id	customer_name	address	city	postal_code	country	email
▶	2	Alan	12347 34 avenue	Portland	15754	USA	alan@gmail.com
	3	Carlos	126977 32 avenue	NewPort	11274	USA	carlosa@gmail.com

Part 2

DROP TABLE IF EXISTS student;

CREATE TABLE student

(

id serial PRIMARY KEY,

first_name VARCHAR(255),

last_name VARCHAR(255),

email VARCHAR(255),

gender VARCHAR(255),

work_phone VARCHAR(55),

book_preference_hardcopy boolean

);

COPY student(first_name,last_name,email,gender,work_phone,book_preference_hardcopy)

-- set the path for file location of student_data.csv

FROM 'C:\Program Files\PostgreSQL\14\bin\Student_data.csv'

delimiter ',' CSV header;

SELECT * FROM student;

SELECT*FROM student_marks;

DROP TABLE IF EXISTS student_marks;

CREATE TABLE student_marks

(

id serial PRIMARY KEY,

student_reg_id integer,

student_id integer,

unit2 integer,

unit3 integer,

unit4 integer,

unit5 integer

);

COPY student_marks(student_reg_id,student_id,unit2,unit3,unit4,unit5)

--set the path for file location of student_marks.csv

FROM 'C:\Program Files\PostgreSQL\14\bin\student_marks.csv'

delimiter ',' CSV header

-- Questions

-- students with the highest marks in Unit 4

```
SELECT student.first_name, student.last_name, student.id, student_marks.id,  
student_marks.unit4
```

```
FROM student INNER JOIN student_marks
```

```
ON student.id=student_marks.id ORDER BY unit4 DESC;
```













Query	Query History	Data output	Messages	Notifications	
<div><div><div><div></div></div><div><div></div></div><div><div></div></div><div><div></div></div><div><div></div></div><div><div></div></div></div></div>					
	<div><div>first_name</div><div>character varying (255)</div><div></div></div>	<div><div>last_name</div><div>character varying (255)</div><div></div></div>	<div><div>id</div><div>integer</div><div></div></div>	<div><div>id</div><div>integer</div><div></div></div>	<div><div>unit4</div><div>integer</div><div></div></div>
1	Wittie	Reinard	704	704	100
2	Garrik	Brussels	970	970	100
3	Kacie	Kiddle	109	109	100
4	Tadeas	McDavitt	428	428	100
5	Thomasin	Melmoth	48	48	100
6	Boothe	Vonderdell	49	49	100
7	Anette	Polding	159	159	100
8	Ramsey	Figgess	942	942	100
9	Caritta	Janek	129	129	100
10	Vinson	Castiblanco	442	442	100
11	Georgie	Elcom	422	422	100
12	Leeann	Vynarde	732	732	100
13	Huntlee	Clopton	229	229	100
14	Cinda	Chittie	615	615	100
15	Edee	Marnes	890	890	100
16	Clarey	Riceards	812	812	100
17	Brynne	Astill	692	692	100
18	Clareta	Walleth	406	406	100
19	Marcia	Yeomans	360	360	100
20	Shara	Prattington	392	392	100
21	Mira	Doppler	898	898	100
Total rows: 1000 of 1000		Query complete 00:00:00.060			

-- students scored between 89 and 100 unit4

**SELECT student.first_name, student.last_name, student.id, student_marks.id,
student_marks.unit4**

FROM student INNER JOIN student_marks

**ON student.id=student_marks.id WHERE unit4 BETWEEN 89 AND 100 ORDER BY
unit4 ASC;**

Query Query History <u>Data output</u> Messages Notifications					
      					
	first_name character varying (255) 	last_name character varying (255) 	id integer 	id integer 	unit4 integer 
1	Barbi	Fourmy	321	321	89
2	Clemmie	Ronan	437	437	89
3	Louisa	Beteriss	268	268	89
4	Harriett	Lockner	252	252	89
5	Joete	Syer	542	542	89
6	Mikey	Cahalin	467	467	89
7	Susi	Rickarsey	959	959	89
8	Augustus	Helwig	770	770	89
9	Tasia	Stigers	95	95	89
10	Lyman	Boam	258	258	89
11	Wheeler	Mulvenna	259	259	89
12	Sam	Aupol	554	554	89
13	Ailina	Detloff	481	481	89
14	Gavin	Atkyns	982	982	89
15	Bord	Hunnicot	755	755	89
16	Harvey	Kinzel	143	143	89
17	Hobey	Ridge	742	742	89
18	Rainer	Schneidau	606	606	89
19	Shina	Freund	4	4	89
20	Elinore	Bulbeck	681	681	89
21	Herbert	Tallboy	196	196	89
Total rows: 604 of 604			Query complete 00:00:00.069		

-- get student full name+gender=female LIKE condition

SELECT first_name, last_name, gender FROM student WHERE first_name LIKE 'An%' AND gender = 'Female';

	first_name character varying (255)	last_name character varying (255)	gender character varying (255)
1	Anette	Mallon	Female
2	Anette	Polding	Female
3	Anna-diane	Cargill	Female
4	Antonella	Maroney	Female
5	Anastasia	Tale	Female
6	Annissa	Bessell	Female
7	Anatola	Gridon	Female
8	Annora	Gilford	Female
9	Annabelle	Stenhouse	Female
10	Annette	Tulk	Female
11	Andreana	Stitt	Female
12	Anthia	Kimbley	Female
13	Anet	Tilne	Female








-- get student full name+gender=male LIKE condition

SELECT first_name, last_name, gender FROM student WHERE first_name LIKE '%an' AND gender = 'Male';

	first_name character varying (255)	last_name character varying (255)	gender character varying (255)
1	Gian	Jaskowicz	Male
2	Tedman	Endersby	Male
3	Sherman	Albin	Male
4	Julian	Mugleston	Male
5	Hagan	Pllu	Male
6	Nolan	Edmunds	Male
7	Vaughan	Liversage	Male
8	Dunstan	Letixier	Male
9	Hadrian	Bourdon	Male
10	Lyman	Boam	Male
11	Riordan	Climer	Male
12	Sloan	Hartzenberg	Male
13	Mathian	Botham	Male
14	Truman	Delia	Male
15	Sherman	Astbury	Male
16	Fran	Chaucer	Male
17	Gaelan	McWilliam	Male
18	Jonathan	McQuillan	Male
19	Logan	Izhak	Male








-- get email/ name from students which name start with Mi

```
SELECT first_name, last_name, email FROM student WHERE first_name LIKE 'Mi%'
ORDER BY last_name;
```

	Query	Query History	Data output	Messages	Notifications
<div></div>					
	first_name character varying (255)	last_name character varying (255)	email character varying (255)		
1	Mikey	Cahalin	mcahalincy@yale.edu		
2	Minna	Dawdry	mdawdry5@jugem.jp		
3	Mira	Doppler	mdopplerox@instagra...		
4	Mikaela	Ekins	mekins49@blogspot.c...		
5	Minerva	Freiberg	mfreibergdq@economi...		
6	Michel	Gaughan	mgaughan7o@msu.edu		
7	Mitchael	Loadwick	mloadwickhf@paypal.c...		
8	Minda	MacElroy	mmacelroylb@jugem.jp		
9	Millisent	McCaster	mmccasterg@scribd.c...		
10	Millard	Millins	mmillinsg4@kickstart...		
11	Michaelina	Nary	mnary4c@yelp.com		
12	Mill	Rubanenko	mrubanenko41@apach...		
13	Mirabel	Summerlie	msummerliel8@gizmo...		
14	Miguelita	Terrans	mterransob@prweb.com		

-- total grade avg from all units

```
SELECT AVG (unit2+unit3+unit4+unit5)/4 AS Average
FROM student_marks;
```

	Query	Query History	Data output
<div></div>			
	average numeric		
1	93.23700000		

-- individual total grade avg

```
SELECT student_id, avg(unit2+unit3+unit4+unit5)/4 AS Average
FROM student_marks
GROUP BY student_id;
```

	student_id integer	average numeric
1	652	94.2500000000000000
2	273	91.2500000000000000
3	51	93.0000000000000000
4	951	93.0000000000000000
5	839	91.5000000000000000
6	70	96.2500000000000000
7	350	97.7500000000000000
8	758	97.0000000000000000
9	539	93.5000000000000000
10	874	97.5000000000000000
11	278	90.7500000000000000
12	946	95.5000000000000000
13	176	90.0000000000000000
14	576	95.5000000000000000
15	292	93.0000000000000000
16	929	95.7500000000000000
17	663	88.7500000000000000
18	770	92.2500000000000000
19	271	97.2500000000000000
20	22	93.7500000000000000
21	417	92.5000000000000000
Total rows: 1000 of 1000		Query complete 00:00:00.00

-- find student where the phone number start with 401

SELECT first_name, last_name, work_phone FROM student WHERE work_phone LIKE '401%' ORDER BY last_name;

	first_name character varying (255)	last_name character varying (255)	work_phone character varying (55)
1	Joseph	Tourner	401-942-8362

-- number of students who are girls

SELECT first_name, last_name, gender FROM student WHERE gender = 'Female';

	first_name character varying (255)	last_name character varying (255)	gender character varying (255)
1	Ree	Cornish	Female
2	Shina	Freund	Female
3	Susy	Widdison	Female
4	Naoma	Truin	Female
5	Luise	Light	Female
6	Friederike	Izakov	Female
7	Millisent	McCaster	Female
8	Roobbie	Thomas	Female
9	Karine	Renzini	Female
10	Fionnula	Bagnell	Female
11	Arline	Meneghi	Female
12	Kayla	Cline	Female
13	Mollie	Maccrie	Female
14	Ally	Filipovic	Female
15	Marcelline	Janicek	Female
16	Ki	Kavanagh	Female
17	Rivy	Shorland	Female
18	Carita	Issacof	Female
19	Jehanna	Simonsson	Female
20	Kassi	Suddard	Female
21	Madonna	Hedaux	Female
Total rows: 475 of 475		Query complete 00:00:00.087	

-- number of students who are boys

SELECT first_name, last_name, gender FROM student WHERE gender = 'Male';

	first_name character varying (255) 🔒	last_name character varying (255) 🔒	gender character varying (255) 🔒
1	Tiebold	Steers	Male
2	Pippo	Mougeot	Male
3	Darby	Winley	Male
4	Nero	Vigours	Male
5	Emmet	Valencia	Male
6	Koenraad	Dugdale	Male
7	Kenneth	Frankish	Male
8	Elijah	Helgass	Male
9	Gian	Jaskowicz	Male
10	Philip	Butterly	Male
11	Jed	Overil	Male
12	Montague	Dunkley	Male
13	Blaine	Pesterfield	Male
14	Ruddie	Drivers	Male
15	Lucien	Slidders	Male
16	Norry	Sprake	Male
17	Beauregard	Reubens	Male
18	Averill	Eveque	Male
19	Mead	Kyneton	Male
20	Weber	Trevithick	Male
21	Edd	McDermott-Row	Male
Total rows: 525 of 525		Query complete 00:00:00.055	

-- students and grades

ON student.id=student_marks.id ORDER BY unit4 DESC;

[illegible]