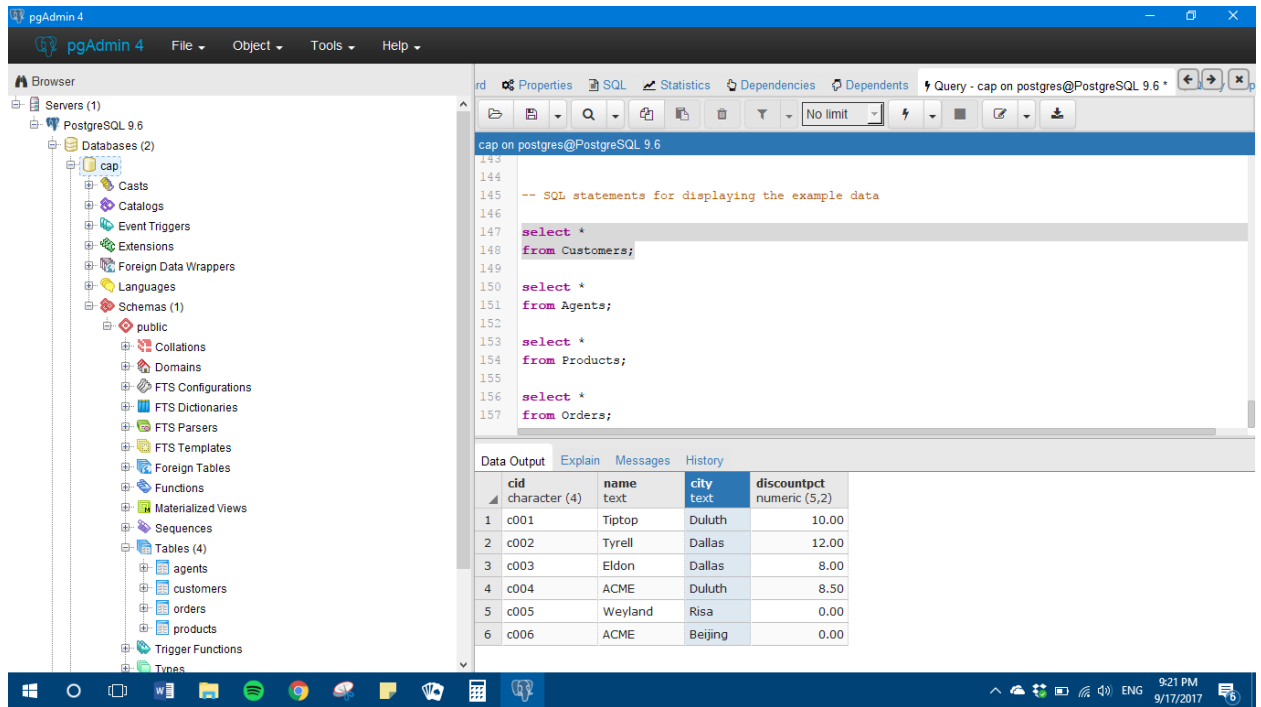


1. 

The screenshot shows the pgAdmin 4 interface with the 'cap' database selected. The query window displays the following SQL statements for displaying example data:

```
-- SQL statements for displaying the example data

select *
from Customers;

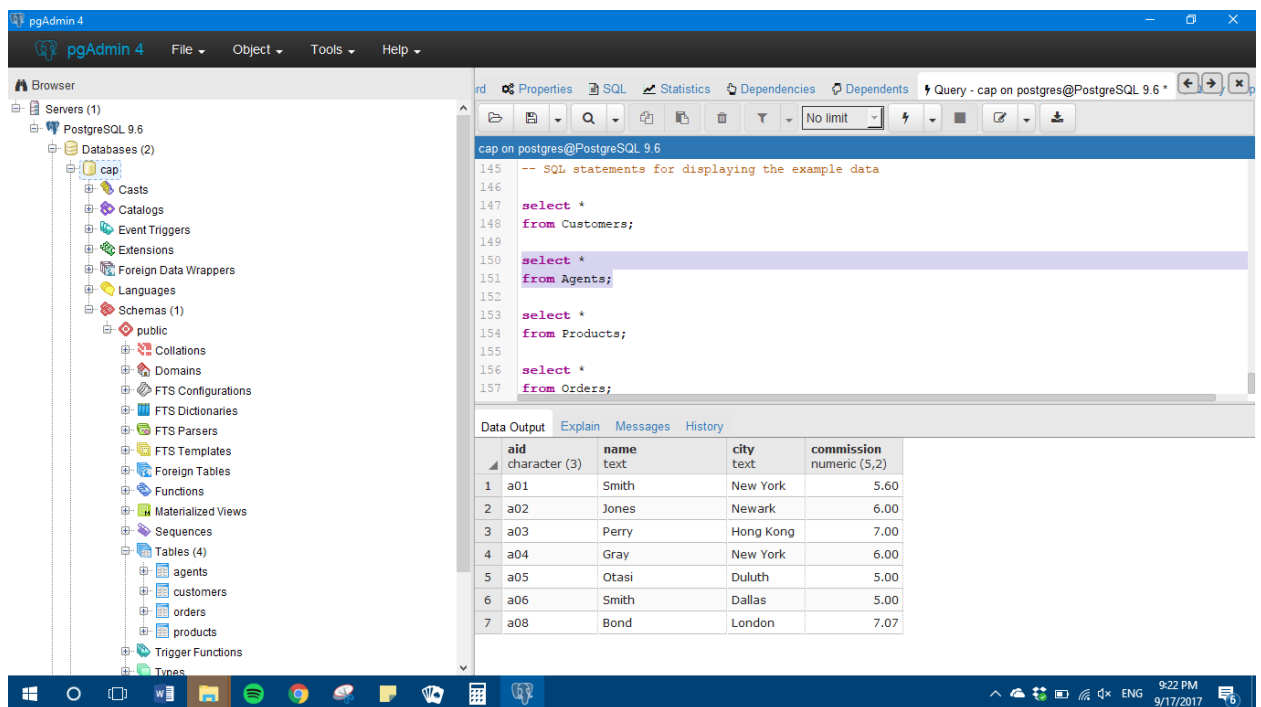
select *
from Agents;

select *
from Products;

select *
from Orders;
```

The Data Output tab shows the following results:

	cid	name	city	discountpct
1	c001	Tiptop	Duluth	10.00
2	c002	Tyrell	Dallas	12.00
3	c003	Eldon	Dallas	8.00
4	c004	ACME	Duluth	8.50
5	c005	Weyland	Risa	0.00
6	c006	ACME	Beijing	0.00



The screenshot shows the pgAdmin 4 interface with the 'cap' database selected. The query window displays the following SQL statements for displaying example data:

```
-- SQL statements for displaying the example data

select *
from Customers;

select *
from Agents;

select *
from Products;

select *
from Orders;
```

The Data Output tab shows the following results:

	aid	name	city	commission
1	a01	Smith	New York	5.60
2	a02	Jones	Newark	6.00
3	a03	Perry	Hong Kong	7.00
4	a04	Gray	New York	6.00
5	a05	Otasi	Duluth	5.00
6	a06	Smith	Dallas	5.00
7	a08	Bond	London	7.07

The screenshot shows the pgAdmin 4 interface. On the left, the 'Browser' pane displays the database structure for 'PostgreSQL 9.6', including 'Databases (2)', 'cap', 'Catalogs', 'Event Triggers', 'Extensions', 'Foreign Data Wrappers', 'Languages', 'Schemas (1)', 'public', 'Collations', 'Domains', 'FTS Configurations', 'FTS Dictionaries', 'FTS Parsers', 'FTS Templates', 'Foreign Tables', 'Functions', 'Materialized Views', 'Sequences', 'Tables (4)', 'agents', 'customers', 'orders', 'products', 'Trigger Functions', and 'Types'. The 'Query' pane on the right shows a query executed on 'postgres@PostgreSQL 9.6' with the following SQL:

```
1 select *
2 from Customers;
3
4 select *
5 from Agents;
6
7 select *
8 from Products;
9
10 select *
11 from Orders;
```

The 'Data Output' pane displays the results of the query, showing a table with 8 rows and 6 columns: pid, name, city, qty, priceusd, and priceusd. The data is as follows:

pid	name	city	qty	priceusd	priceusd
1	p01	Heisenberg compensator	Dallas	111400	0.50
2	p02	universal translator	Newark	203000	0.50
3	p03	Commodore PET	Duluth	150600	1.00
4	p04	LCARS module	Duluth	125300	1.00
5	p05	pencil	Dallas	221400	1.00
6	p06	trapper keeper	Dallas	123100	2.00
7	p07	flux capacitor	Newark	100500	1.00
8	p08	HAL 9000 memory core	Newark	200600	1.25

The screenshot shows the pgAdmin 4 interface. On the left, the 'Browser' pane displays the database structure for 'PostgreSQL 9.6', including 'Databases (2)', 'cap', 'Catalogs', 'Event Triggers', 'Extensions', 'Foreign Data Wrappers', 'Languages', 'Schemas (1)', 'public', 'Collations', 'Domains', 'FTS Configurations', 'FTS Dictionaries', 'FTS Parsers', 'FTS Templates', 'Foreign Tables', 'Functions', 'Materialized Views', 'Sequences', 'Tables (4)', 'agents', 'customers', 'orders', 'products', 'Trigger Functions', and 'Types'. The 'Query' pane on the right shows a query executed on 'postgres@PostgreSQL 9.6' with the following SQL:

```
7 select *
8 from Products;
9
10 select *
11 from Orders;
```

The 'Data Output' pane displays the results of the query, showing a table with 14 rows and 8 columns: ordno, month, cid, aid, pid, quantity, and totalusd. The data is as follows:

ordno	month	cid	aid	pid	quantity	totalusd	
1	1011	Jan	c001	a01	p01	1100	495.00
2	1012	Jan	c002	a03	p03	1200	1056.00
3	1015	Jan	c003	a03	p05	1000	920.00
4	1016	Jan	c006	a01	p01	1000	500.00
5	1017	Feb	c001	a06	p03	500	540.00
6	1018	Feb	c001	a03	p04	600	540.00
7	1019	Feb	c001	a02	p02	400	180.00
8	1020	Feb	c006	a03	p07	600	600.00
9	1021	Feb	c004	a06	p01	1000	457.50
10	1022	Mar	c001	a05	p06	450	810.00
11	1023	Mar	c001	a04	p05	500	450.00
12	1024	Mar	c006	a06	p01	880	400.00
13	1025	Apr	c001	a05	p07	888	799.20
14	1026	May	c002	a05	p03	808	711.04

2. A primary key is a column in a table that uniquely identifies each row. Values cannot repeat, nor can they be null. Candidate keys are the columns that are eligible to be the primary key. A superkey is a set of attributes that contains a key, which means every key is a superkey. Superkeys do not need to be minimal, though.
3. A data type is an attribute that describes an object in a table. If there was a table named Cars, the data fields would be make (string), model (string), year (integer), color (string), deluxe (Boolean), and VIN. The VIN number would be the primary key, because no 2 cars have the same VIN number. All data fields would not be nullable, except for deluxe because not every car may be the deluxe model with all the add-on gadgets.
4. Rules
  - a. The “first normal form” rule states that each cell must be unstructured and should not be able to be broken down (for example, name can be broken down into first and last name, therefore it violates the rule). This rule is important because it helps keep the database organized well and any piece of data can be queried.
  - b. The “access rows by content only” rule states that you should search a row by what is in it, not by its position. This is important because the row can change positions based on the current query. For example, if the current query lists the data in ascending alphabetical order (Chevrolet, Ford, Kia), the column with Kia is listed as the 3<sup>rd</sup> column. If the next query lists the data in descending alphabetical order (Kia, Ford, Chevrolet), then the column with Kia is listed as the 1<sup>st</sup> column.
  - c. The “all rows must be unique” rule states each row should be different from each other. If more than one row had the same name, whenever you performed a query

for one you would get the other. When you write a query for specific information, you should be able to retrieve only the information you are looking for. For example, if you have 2 columns called “name” (one for first names and one for last names) and you only need the last names, there’s no way for the program to know which column you were referring to.