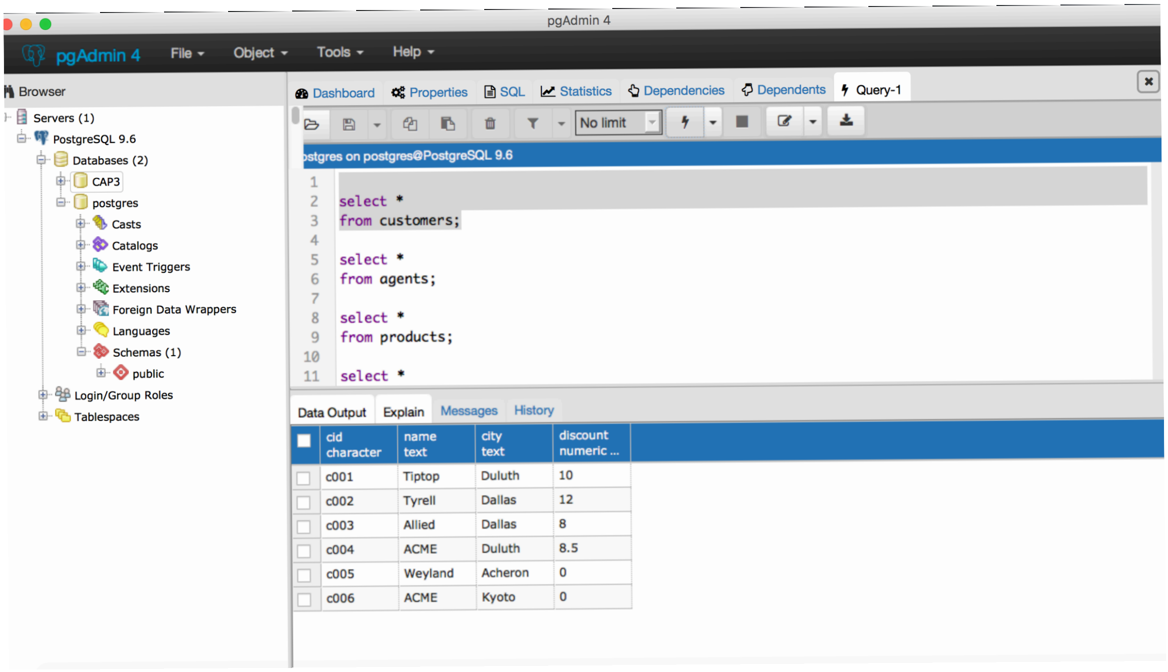
Reyhan Ayhan
09/15/16
Lab 2

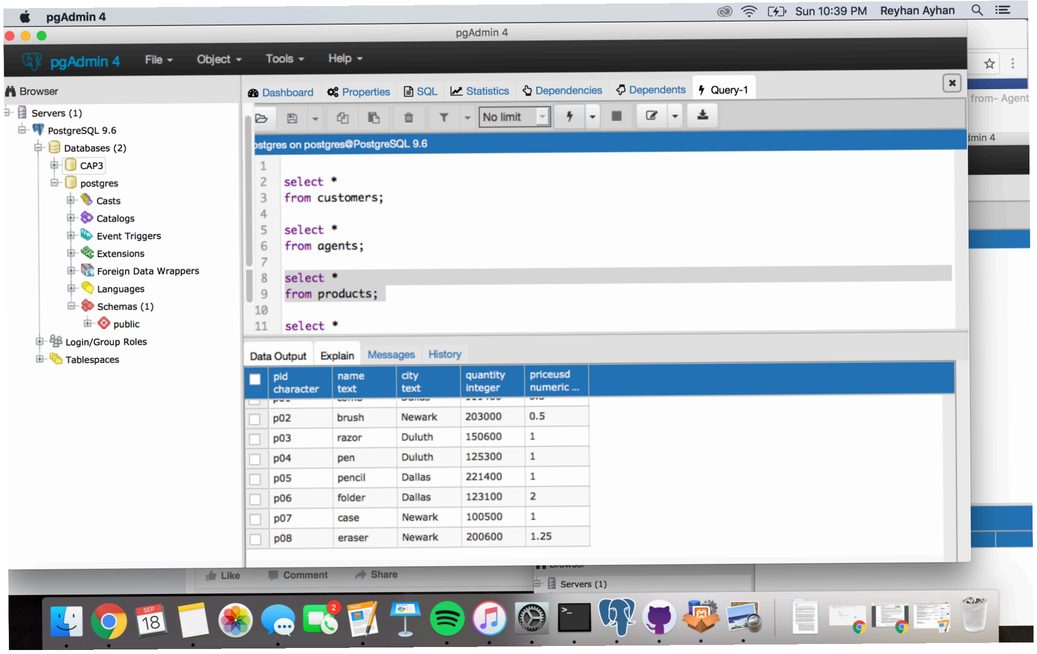
1. Select customers



```
select *
from customers;

select *
from agents;

select *
from products;

select *
```

| cid<br>character | name<br>text | city<br>text | discount<br>numeric ... |
|---|---|---|---|
| c001 | Tiptop | Duluth | 10 |
| c002 | Tyrell | Dallas | 12 |
| c003 | Allied | Dallas | 8 |
| c004 | ACME | Duluth | 8.5 |
| c005 | Weyland | Acheron | 0 |
| c006 | ACME | Kyoto | 0 |

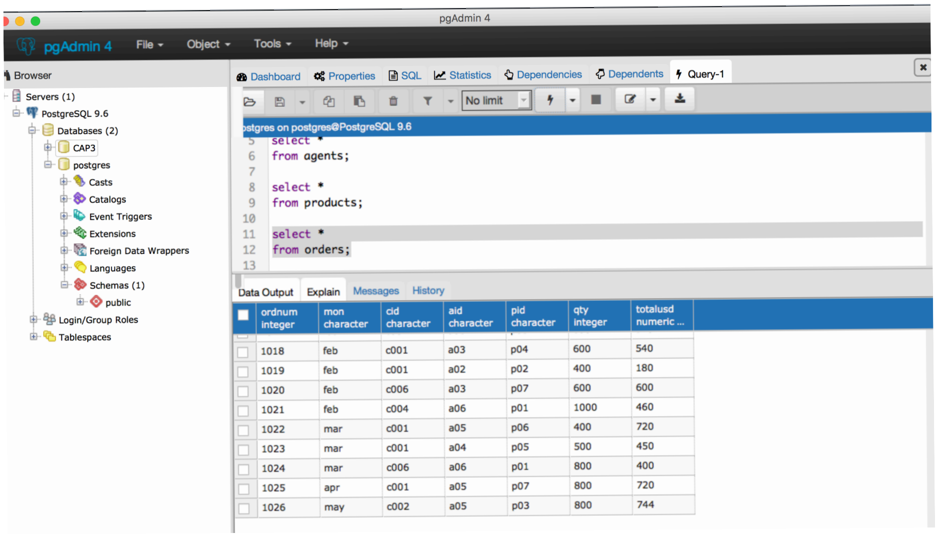
Select products



```
select *
from customers;

select *
from agents;

select *
from products;

select *
```

| pid<br>character | name<br>text | city<br>text | quantity<br>integer | priceusd<br>numeric ... |
|---|---|---|---|---|
| p01 | comb | Dallas | 111400 | 0.5 |
| p02 | brush | Newark | 203000 | 0.5 |
| p03 | razor | Duluth | 150600 | 1 |
| p04 | pen | Duluth | 125300 | 1 |
| p05 | pencil | Dallas | 221400 | 1 |
| p06 | folder | Dallas | 123100 | 2 |
| p07 | case | Newark | 100500 | 1 |
| p08 | eraser | Newark | 200600 | 1.25 |

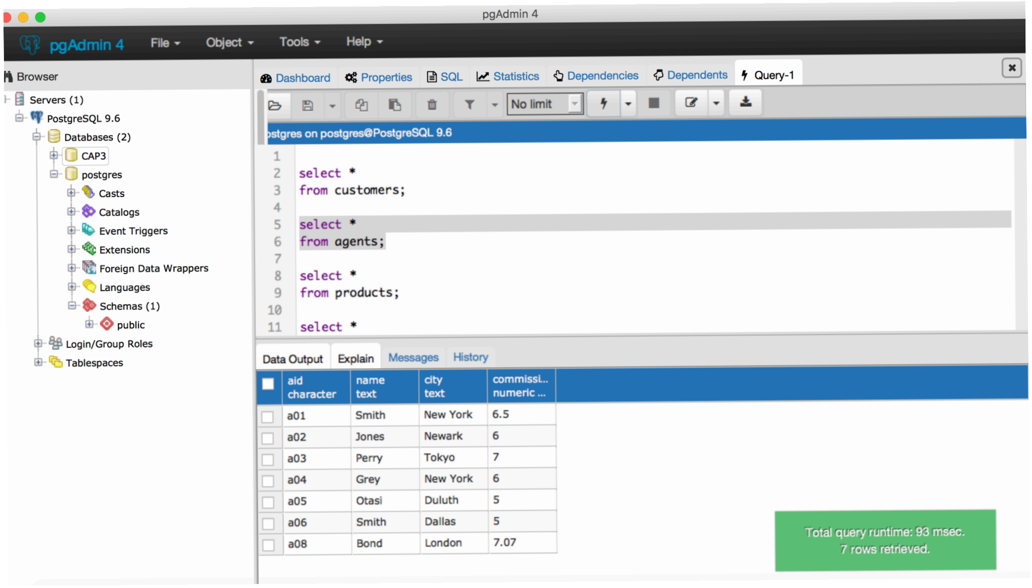
Select ordes



```
select *
from agents;

select *
from products;

select *
from orders;
```

| ordnum<br>integer | mon<br>character | cid<br>character | aid<br>character | pid<br>character | qty<br>integer | totalusd<br>numeric ... |
|---|---|---|---|---|---|---|
| 1018 | feb | c001 | a03 | p04 | 600 | 540 |
| 1019 | feb | c001 | a02 | p02 | 400 | 180 |
| 1020 | feb | c006 | a03 | p07 | 600 | 600 |
| 1021 | feb | c004 | a06 | p01 | 1000 | 460 |
| 1022 | mar | c001 | a05 | p06 | 400 | 720 |
| 1023 | mar | c001 | a04 | p05 | 500 | 450 |
| 1024 | mar | c006 | a06 | p01 | 800 | 400 |
| 1025 | apr | c001 | a05 | p07 | 800 | 720 |
| 1026 | may | c002 | a05 | p03 | 800 | 744 |

Select agents



```

select *
from customers;

select *
from agents;

select *
from products;

select *
```

| aid<br>character | name<br>text | city<br>text | commissi...<br>numeric ... |
|---|---|---|---|
| a01 | Smith | New York | 6.5 |
| a02 | Jones | Newark | 6 |
| a03 | Perry | Tokyo | 7 |
| a04 | Grey | New York | 6 |
| a05 | Otasi | Duluth | 5 |
| a06 | Smith | Dallas | 5 |
| a08 | Bond | London | 7.07 |

Total query runtime: 93 msec.
7 rows retrieved.

2. Explain the distinctions among the terms primary key, candidate key, and super key.

A primary key is the candidate key attribute that is suited to keep the uniqueness in a table. The candidate key is the attribute, or a set of attributes, that allows for uniqueness for each row. There could be several candidate key for a relation. It's otherwise known as a minimal Super key. Super keys are a superset of the Candidate key. Once you add a new attribution to the Candidate Key, it will then be some form of a Super Key.

3. Write a short essay on data types. Select a topic for which you might create a table. Name the table and list its Wields (columns). For each Wield, give its data type and whether or not it is nullable.

A data type essentially takes the format a piece of data can take. There are several types of datatypes, one of which is a boolean data type. A boolean data type or a domain is a set or range of values a database can old. A numeric and in numeric domain are also data types. An innumeric domain has something to remain constant (i.e. grades, days of the week). A tricky part to data types is understanding the appropriate use of data comparisons. For example, the operations defined for dates include comparing dates and allowing them to be sortable in a domain.

A topic where a user might create a table is possibly when referring to superheroes and their powers. I would name the table SP for Super Powers. The first column would entail a superpower, the next column would entail a section of another superpower. The first column enlisting the first superpower would not be nullable because some data is needed in order for the the super hero to qualify to be an entity of the table. The second super power column could be nullable because each superhero is not required to have a secondary super power.

4. Explain the following relational "rules" with examples and reasons why they are important.

a. The "Wirst normal form" rule

All "fields" (rows and columns) are atomic. By atomic, we're referring to the smallest possible unit and there would be no internal structure. There can't be duplicate data that sin't described as a key. In the example of creating a table for superpowers, there are originally two rows and two columns. The two columns remain superpower 1 and superpower 2 while the two rows are individually labeled with Sean and Pierce. Sean has the superpower of pronouncing his s' with the -sh sound, therefore s=sh for superpower 1. Now for Pierce, rather than having two pieces of data describe his superpowers, we include his other superpower within the super power 2 column on that same row. This shows the idea of remaining atomic where the smallest unit fits into another category rather than having to attributes belonging under a single column. This is important because some would think that a DOB column might violate this first rule but it's untrue because it is represented by a datatype that the database supports and a datatype is atomic.

b. The "access rows by content only" rule

This rule is about the "what, not the where". A user is not allowed to access data according to where it is in the table. The reason this is important is because if in the superhero example, I would refer to the s=sh superpower as "row 1, column 1", then in the instance of moving a row around and having Pierce be row 1 rather than Sean, now "row 1, column 1" carries different values and is referring to different things. These incidents lead to inaccurate data and can cause for major errors when attempting to retrieve data. In order to avoid this, the user has to specify what's there- include column names, not numbers.

c. The "all rows must be unique" rule

All rows must be unique because it is the natural consequence of set theory. In set theory, there are sets, an intersection and union of the sets, and the subtraction of them. In an example where there is a set with two of the same numbers, when determining the intersection of those sets, you include the number once and not twice although it appears twice. The same rule applies for the union of the sets, the number is only written once. This idea applies to when creating and using a database, all rows must be unique because having duplicates could lead to complications and mix-matched data.