Megan Poyntz
Database Management Systems – Lab 2

Explain the distinctions among the terms primary key, candidate key, and superkey.

Keys are used to uniquely identify rows within data. A superkey is any set of columns that, when put together, uniquely identifies every row in the table. There are no two distinct rows, or tuples, that have the same values for the attributes in the set. When looking for the superkey to identify every row in the table, you must consider all columns in the table. Stemming from the superkey, a candidate key is an identifier that uses the smallest number of columns to achieve uniqueness of every row in the column. A candidate key cannot have any columns removed from it without losing the unique identification property. From a candidate key, we get a primary key. A primary key is the chosen candidate key to achieve uniqueness throughout the rows when you have multiple candidate keys. Out of the three keys listed, a primary key is the smallest identifier of uniqueness.

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

There are many kinds of data types within a relation schema in SQL. Data types include character strings, bit strings, Boolean data, integers, floating-point numbers, and dates and times. Character strings can be of either fixed or varying length. Fixed-length character strings are denoted by the term CHAR, whereas strings of "x" amount of characters are denoted by VARCHAR in SQL. Bit strings are similar to characters, except their values are in bits. BIT denotes the fixed amount of bits, whereas BITVARYING means that there is a varying length of bits. Boolean data is a type whose value is logical. For example, Boolean data would be TRUE, FALSE, or UNKNOWN. Integer (INT or INTEGER) is any typical integer value. Within INTEGER, there is SHORTINT, which also denotes an integer, but there are less numbers of bits permitted. As for floating numbers, they are denoted by the term FLOAT, REAL or DECIMAL. For more exact floating numbers, there is DOUBLE PRECISION and DECIMAL (n,d). DECMIAL (n,d) allows you to exactly determine what the number will look like; n is the amount of real numbers, and d represents the positions from the right of the decimal point. Finally, dates and times are denoted by DATE and TIME. Because dates and times require a special form, these data types are strung together in a unique way to create that special form that makes sense as a date or time.

A topic that serves as a good example of data types within a table would be a table of Employee Profiles for an organization. This table would keep all the information about the company's employees, including everything from their personal information to their organizational information. Some of the fields of the table, for example, would include employee

first name (VARCHAR), employee last name (VARCHAR), middle initial (CHAR), age (INT), department (VARCHAR), position (CHAR or VARCHAR), date of birth (DATE), marital status (Boolean TRUE, FALSE or UNKNOWN), salary (fixed salary with INT, hourly wage with DECIMAL or DOUBLE PRECISION). For all of these fields, I do not think that there are any that are nullable. There is an answer for each field, even if the answer is unknown (for example, marital status). I believe the only time these fields can be nullable is when an employee is initially entered into the system (for example, a new hire). It may take time to get the paperwork that has the information on the employee's profile, so in the short time that there is no information, "nullable" is acceptable.

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

The "first normal form" rule

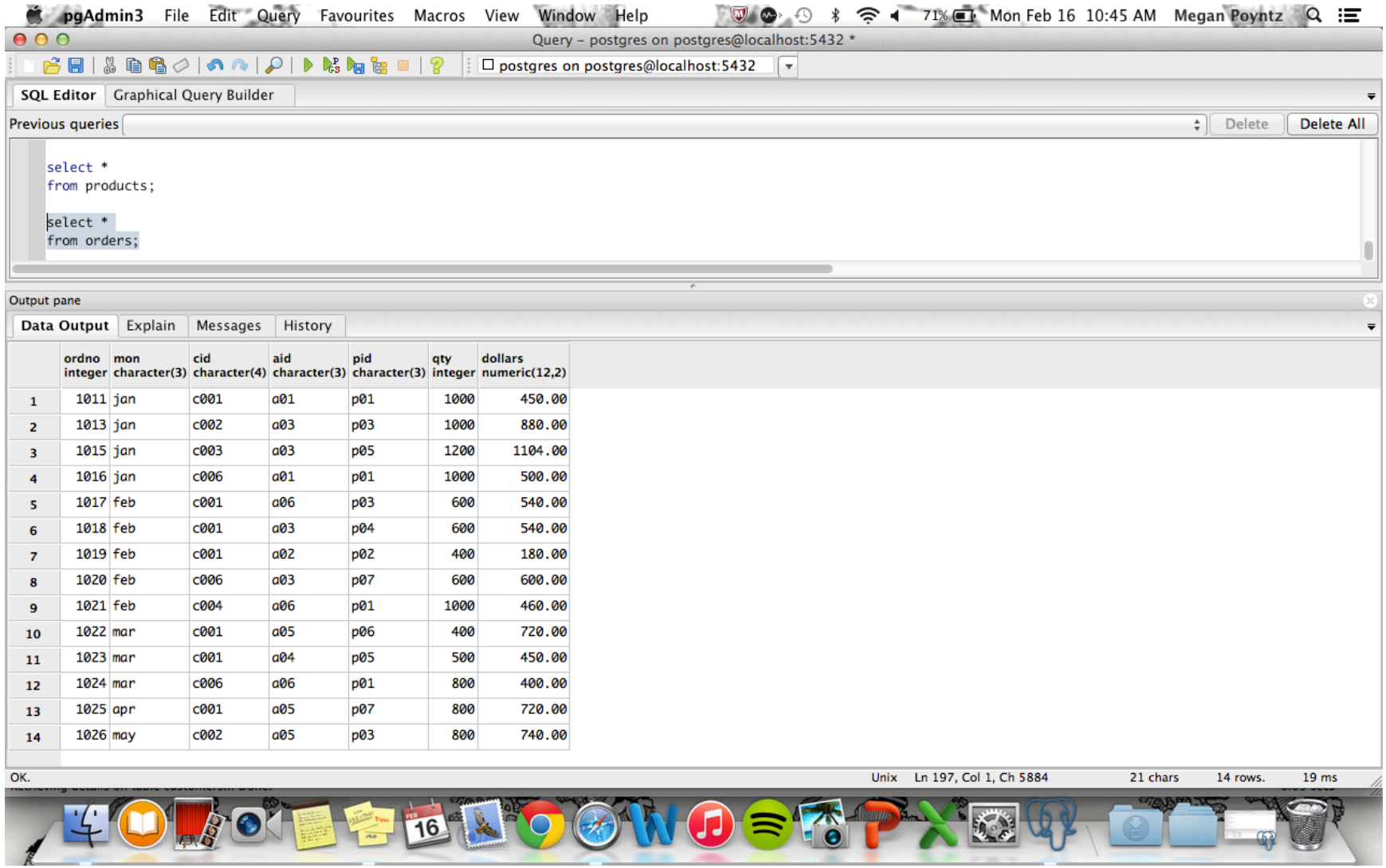
The first relational rule states that there should be no multi-valued attributes in the fields of a table. This means that there cannot be more than one data type in a column. If there needs to be a variety of data types associated with a field, then there needs to be a cross reference between different tables in the database. Information can only be represented in one way in a table. Going off of my previous example of Employee Profiles, one of the fields could be the number of dependents. In this field, there would be a number that would then reference another table listing the specifics of each dependent. The field of "number of dependents" in the Employee Profile table cannot list a various amount of data types, such as each name of the dependent. If it did that, then it would not be able to list all the information for each dependent.

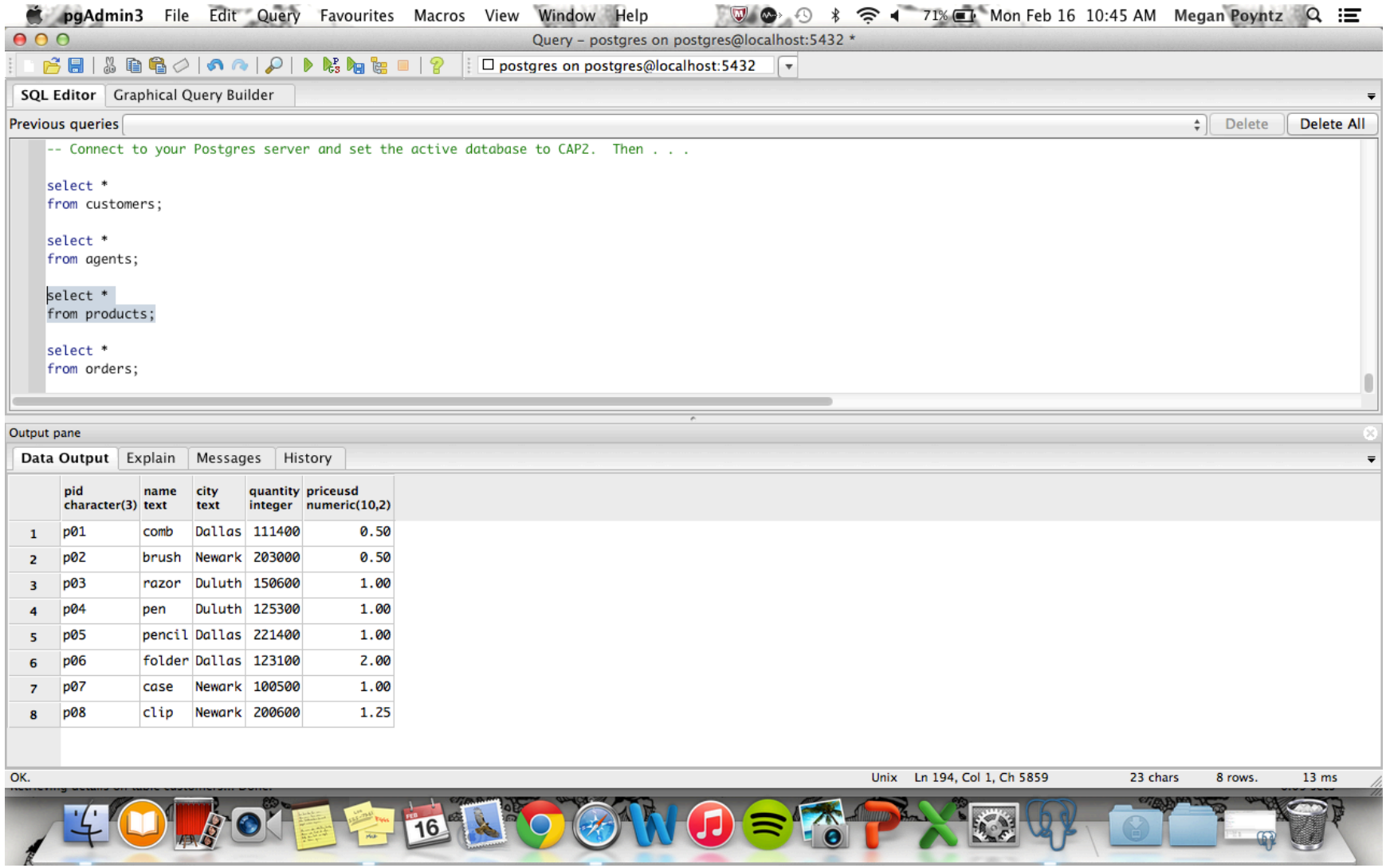
The "access rows by content only" rule

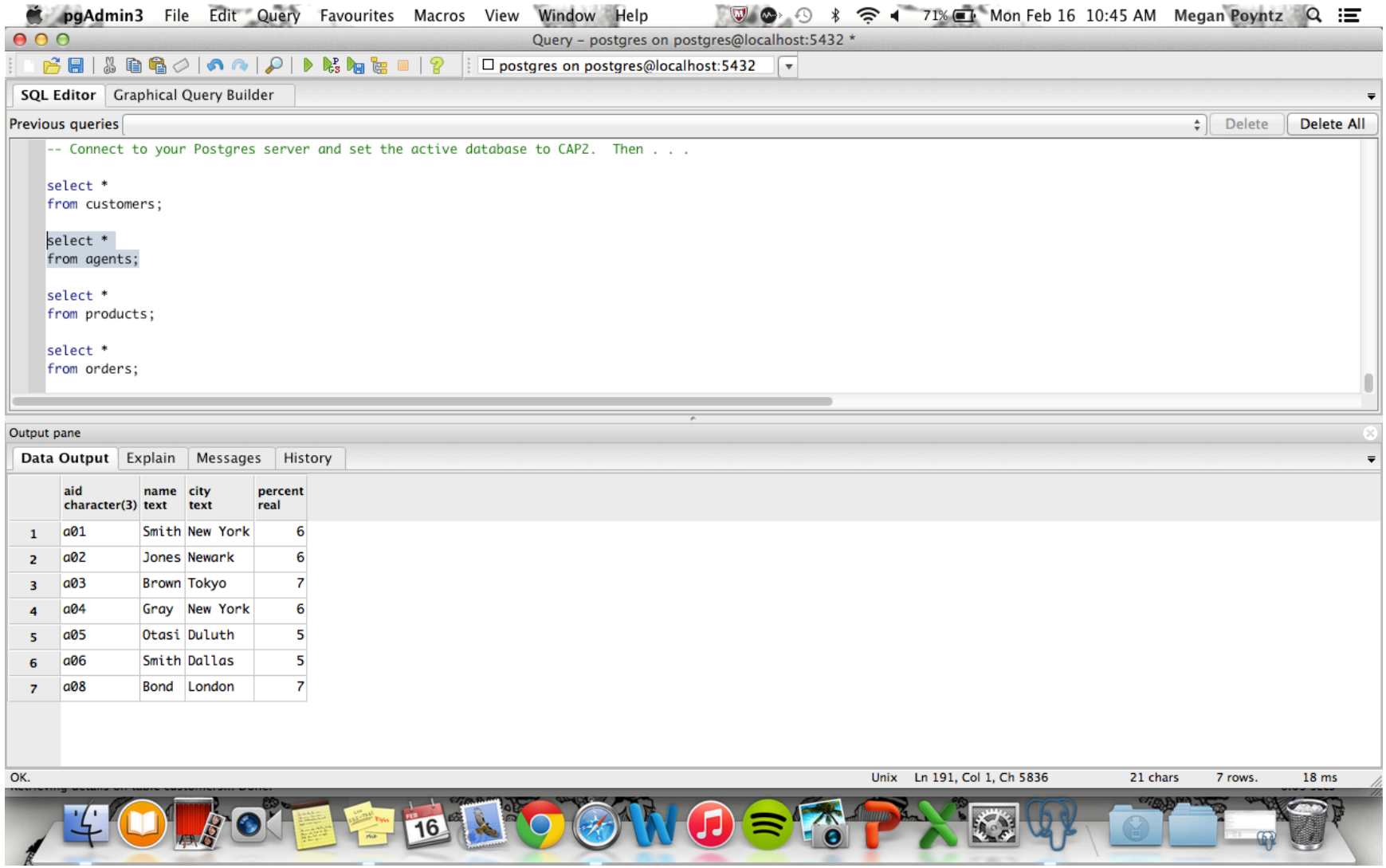
The second relation rule states that, when asking about data, ask "what, not where." This rule is especially important because by specifying *what* data you're asking for, rather than the location of the data (*where* it is), you are requesting specifically what you are searching for. If you ask *where* (for example, "for the information in column three"), then there is a chance that the information in column three has changed.

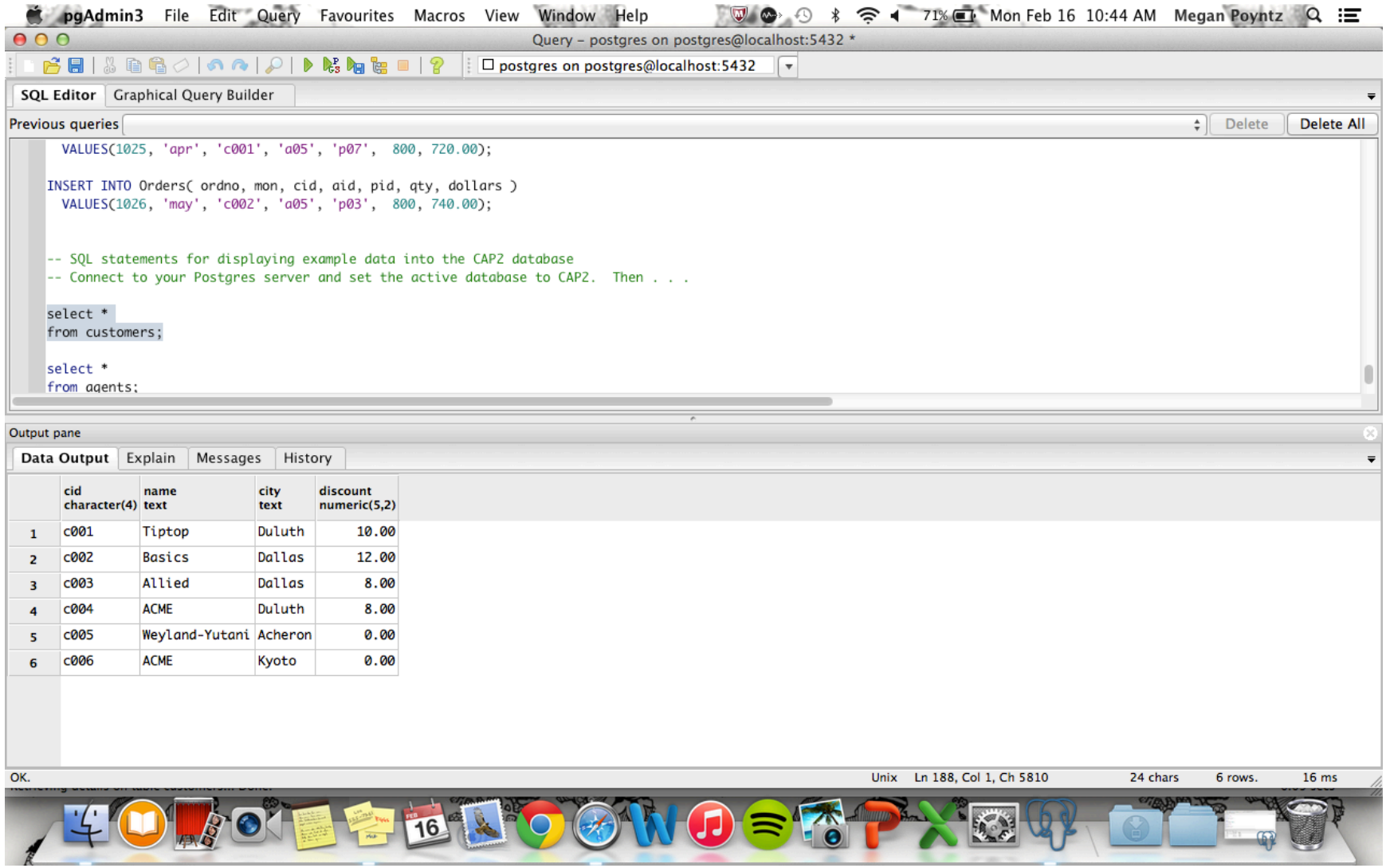
The "all rows must be unique" rule

The third relation rule states that all rows in a table must be unique. For each row in a table, every field is in the row. If there is repeated information in the rows, then you cannot specify *what* information you need (this relates back to the previous rule). For example, if there are two rows within the Employee Profile table with the same information (same first name, same last name, same department, etc.), then there is no way to tell them apart. This is the reason for keys – we need to uniquely identify rows from one another in tables.


pgAdmin3 File Edit Query Favourites Macros View Window Help
71% Mon Feb 16 10:45 AM Megan Poyntz
Query - postgres on postgres@localhost:5432 *
postgres on postgres@localhost:5432
SQL Editor
Graphical Query Builder
Previous queries
Delete
Delete All
select *
from products;
select *
from orders;
Output pane
Data Output
Explain
Messages
History
ordno integer
mon character(3)
cid character(4)
aid character(3)
pid character(3)
qty integer
dollars numeric(12,2)
1 1011 jan c001 a01 p01 1000 450.00
2 1013 jan c002 a03 p03 1000 880.00
3 1015 jan c003 a03 p05 1200 1104.00
4 1016 jan c006 a01 p01 1000 500.00
5 1017 feb c001 a06 p03 600 540.00
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 460.00
10 1022 mar c001 a05 p06 400 720.00
11 1023 mar c001 a04 p05 500 450.00
12 1024 mar c006 a06 p01 800 400.00
13 1025 apr c001 a05 p07 800 720.00
14 1026 may c002 a05 p03 800 740.00
OK.
Unix Ln 197, Col 1, Ch 5884
21 chars
14 rows.
19 ms


pgAdmin3 File Edit Query Favourites Macros View Window Help
71% Mon Feb 16 10:45 AM Megan Poyntz
Query - postgres on postgres@localhost:5432 *
postgres on postgres@localhost:5432
SQL Editor
Graphical Query Builder
Previous queries
Delete
Delete All
-- Connect to your Postgres server and set the active database to CAP2. Then . . .
select *
from customers;
select *
from agents;
select *
from products;
select *
from orders;
Output pane
Data Output
Explain
Messages
History
pid character(3)
name text
city text
quantity integer
priceusd numeric(10,2)
1 p01 comb Dallas 111400 0.50
2 p02 brush Newark 203000 0.50
3 p03 razor Duluth 150600 1.00
4 p04 pen Duluth 125300 1.00
5 p05 pencil Dallas 221400 1.00
6 p06 folder Dallas 123100 2.00
7 p07 case Newark 100500 1.00
8 p08 clip Newark 200600 1.25
OK.
Unix Ln 194, Col 1, Ch 5859
23 chars
8 rows.
13 ms


pgAdmin3 File Edit Query Favourites Macros View Window Help
71% Mon Feb 16 10:45 AM Megan Poyntz
Query - postgres on postgres@localhost:5432 *
postgres on postgres@localhost:5432
SQL Editor
Graphical Query Builder
Previous queries
Delete
Delete All
-- Connect to your Postgres server and set the active database to CAP2. Then . . .
select *
from customers;
select *
from agents;
select *
from products;
select *
from orders;
Output pane
Data Output
Explain
Messages
History
aid character(3)
name text
city text
percent real
1 a01 Smith New York 6
2 a02 Jones Newark 6
3 a03 Brown Tokyo 7
4 a04 Gray New York 6
5 a05 Otasi Duluth 5
6 a06 Smith Dallas 5
7 a08 Bond London 7
OK.
Unix Ln 191, Col 1, Ch 5836 21 chars 7 rows. 18 ms


pgAdmin3 File Edit Query Favourites Macros View Window Help
71% Mon Feb 16 10:44 AM Megan Poyntz
Query - postgres on postgres@localhost:5432 *
postgres on postgres@localhost:5432
SQL Editor
Graphical Query Builder
Previous queries
Delete
Delete All
VALUES(1025, 'apr', 'c001', 'a05', 'p07', 800, 720.00);
INSERT INTO Orders(ordno, mon, cid, aid, pid, qty, dollars)
VALUES(1026, 'may', 'c002', 'a05', 'p03', 800, 740.00);
-- SQL statements for displaying example data into the CAP2 database
-- Connect to your Postgres server and set the active database to CAP2. Then . . .
select *
from customers;
select *
from agents;
Output pane
Data Output
Explain
Messages
History
cid character(4)
name text
city text
discount numeric(5,2)
1 c001 Tiptop Duluth 10.00
2 c002 Basics Dallas 12.00
3 c003 Allied Dallas 8.00
4 c004 ACME Duluth 8.00
5 c005 Weyland-Yutani Acheron 0.00
6 c006 ACME Kyoto 0.00
OK.
Unix Ln 188, Col 1, Ch 5810 24 chars 6 rows. 16 ms