

Michael Read

Lab 2

September 15, 2016

### 1) Query and results screen shots

The screenshot shows the PostgreSQL Query Editor interface. The SQL Editor contains the query `select * from agents;`. The Output pane displays the results of the query in a table format. The table has five columns: `aid` (character(3)), `name` (text), `city` (text), and `commission` (numeric(5,2)). The results show 7 rows of data.

	aid character(3)	name text	city text	commission numeric(5,2)
1	a01	Smith	New York	6.50
2	a02	Jones	Newark	6.00
3	a03	Perry	Tokyo	7.00
4	a04	Grey	New York	6.00
5	a05	Otasi	Duluth	5.00
6	a06	Smith	Dallas	5.00
7	a08	Bond	London	7.07

OK. DOS Ln 1, Col 21, Ch 21 7 rows. 52 msec

The screenshot shows the PostgreSQL Query Editor interface. The SQL Editor contains the query `select * from customers;`. The Output pane displays the results of the query in a table format. The table has five columns: `cid` (character(4)), `name` (text), `city` (text), and `discount` (numeric(5,2)). The results show 6 rows of data.

	cid character(4)	name text	city text	discount numeric(5,2)
1	c001	Tiptop	Duluth	10.00
2	c002	Tyrell	Dallas	12.00
3	c003	Allied	Dallas	8.00
4	c004	ACME	Duluth	8.50
5	c005	Weyland	Acheron	0.00
6	c006	ACME	Kyoto	0.00

OK. DOS Ln 1, Col 25, Ch 25 6 rows. 76 msec

Query - postgres on postgres@localhost:5432 \*

File Edit Query Favourites Macros View Help

SQL Editor Graphical Query Builder

Previous queries  Delete Delete All

```
select * from products;
```

Scratch pad

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	eraser	Newark	200600	1.25

OK. DOS Ln 1, Col 23, Ch 23 8 rows. 91 msec

Query - postgres on postgres@localhost:5432 \*

File Edit Query Favourites Macros View Help

SQL Editor Graphical Query Builder

Previous queries  Delete Delete All

```
select * from orders;
```

Scratch pad

Output pane

Data Output Explain Messages History

	ordnum integer	mon character(3)	cid character(4)	aid character(3)	pid character(3)	qty integer	totalusd 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	744.00

OK. DOS Ln 1, Col 21, Ch 21 14 rows. 64 msec

The data in the results of the queries matched the data shown in the CAP snapshot

## 2) Distinctions between primary key, candidate key, and superkey

In relational databases, a superkey is a group of columns in a table which uniquely identifies any row in that table. This means that no two rows in a table will have the same data stored in the superkey, as knowing the data in the superkey tells you exactly which row in the table you are looking at. When you reduce the number of columns in the superkey to the minimum number of columns required to uniquely identify each row in a table, the superkey becomes the candidate key. If the candidate key for a table is a single column, it is called a primary key. For example, a superkey in a table holding data about airline flights might be the columns of `departure_time`, `destination`, and `flight_number`. However, maybe in order to uniquely identify each row, you only need `flight_number`. That means that you can simplify the superkey down to a candidate key of just `flight_number`. This candidate key is special, as it only contains one column, which means that the airline flights table has the `flight_number` column as its primary key.

### 3) Short essay on data types

In a database table holding information about patients in an Emergency Room, one might name the table ER\_Patient. Some example columns in the ER\_Patient table could be date\_admitted, birth\_date, name, weight\_in\_lbs, height\_in\_ft, house\_address, occupation, place\_of\_work, contact\_phone\_number, contact\_name, and contact\_relationship. The fields contact\_relationship, contact\_name, contact\_phone\_number, place\_of\_work, occupation, house\_address, and name would be of the data type text, because they are mostly composed of text, and the operations you would want to use on them would be those of type text. Both weight\_in\_lbs and height\_in\_ft would be of type double, as they would be decimal numbers, and used in numerical calculations making the type double the right choice. The column birth\_date would be of type date, as it would record the date of birth, and the operations used on it would fit the type date. Finally, date\_admitted would be of type timestamp, as both the date and time that the patient was admitted would be important, making timestamp the best option. The columns date\_admitted, birth\_date, name, weight\_in\_lbs, and height\_in\_ft would be not nullable, as those fields can be filled in for every patient admitted to the ER. The columns of house\_address, occupation, place\_of\_work, contact\_phone\_number, contact\_name, and contact\_relationship are not all able to be filled in for every patient, as some may not have an address, some might be out of work, and others might not have anyone to contact. This means that those columns would be nullable.

#### 4) Explain the 3 relational “rules”

1. The “first normal form” rule means that all values in the intersection of a column and a row should be atomic, or a single value. This means that no entry should contain a list of multiple values, and each column should be designed to only hold one value. An example of this is in a table storing data about students in a class, where one of the columns is called `students_pet_name`. While this would work fine if every student in the class only had a maximum of one pet, if there was a student in the class with multiple pets, then in this design, all of those pets names would have to be stored in the one column, which violates this rule. Making sure that all entries are atomic is important, because if they are not, then the structure of the database deteriorates, and searching for a single attribute becomes difficult, because if you find multiple values in one column, then you may not know which value is the one you want.
2. The “access rows by content only” rule means that the only way you should be looking up values in a table is by naming what column and what row you are looking in. Instead of querying for the data in row 2, column 4, you instead should query for what is in the superpower column of the Sean Connery row. The reason this is important is because databases are defined as sets of tables, which are in turn sets of columns and rows. Set theory tells us that elements in a set have no inherent order, which means that technically speaking, column 4 in the table could be superpowers, or it could be `height_in_ft`, both of which are valid for the table. In order to keep tables as sets of columns and rows, you must always query by content, and not by position.
3. The “all rows must be unique” rule means that no two rows in a table should have the same values for all of their columns. When you have multiple rows with the same information, you run into a duplication problem, where changing one piece of information means you have to change it in multiple places, and when you go searching for something, you might not know which row you should be looking in. For example, in a table of students, if you have two rows which both have the same information, you would have two instances of the same student in the table. If you then went to change something, but didn’t change it in both places, then when to search your table, you might not be sure which entry of that student you should use, as they would both be for the same student, but would have different information about that student.