Dan Grogan

Lab 2

9/15/2016

2.

A candidate key is any column or attribute that will provide uniqueness in each row of your relational table. There can be multiple candidate keys. The primary key is the candidate key that you select to be a reference to your relational table. A superkey is a key that combines two or more columns/attributes in order to reference the data in the table.

3.

There are six different data types that we can use in our database. There are character strings, bit strings, boolean values, integer values, floating-point numbers and date/time values. Character strings and bit strings are both strings of fixed or varying length, however character strings are made up of characters and bit strings are made up of bits. Boolean values are the logical values of true and false. Integer values are different from floating-point values because floating-point values have a decimal point that you can fix for the value. Date and time values have their own data type, however they could be stored in character strings or integers as well, depending on the situation. Say that I was making a relational table to keep track of the stats for a Marist volleyball match. We could name this table "*match1"* to signify that these are the stats for the first match of the year. The table *match1* could contain fields such as *player#, lastname, ptsplayed, kills, assists, digs, aces* ect. The fields *player#, ptsplayed, kills, assists, digs,* and *aces* would all consist of INT values. The field *lastname* would have a CHAR value, as names are composed of characters. The only value that would not be nullable would be the *player#*

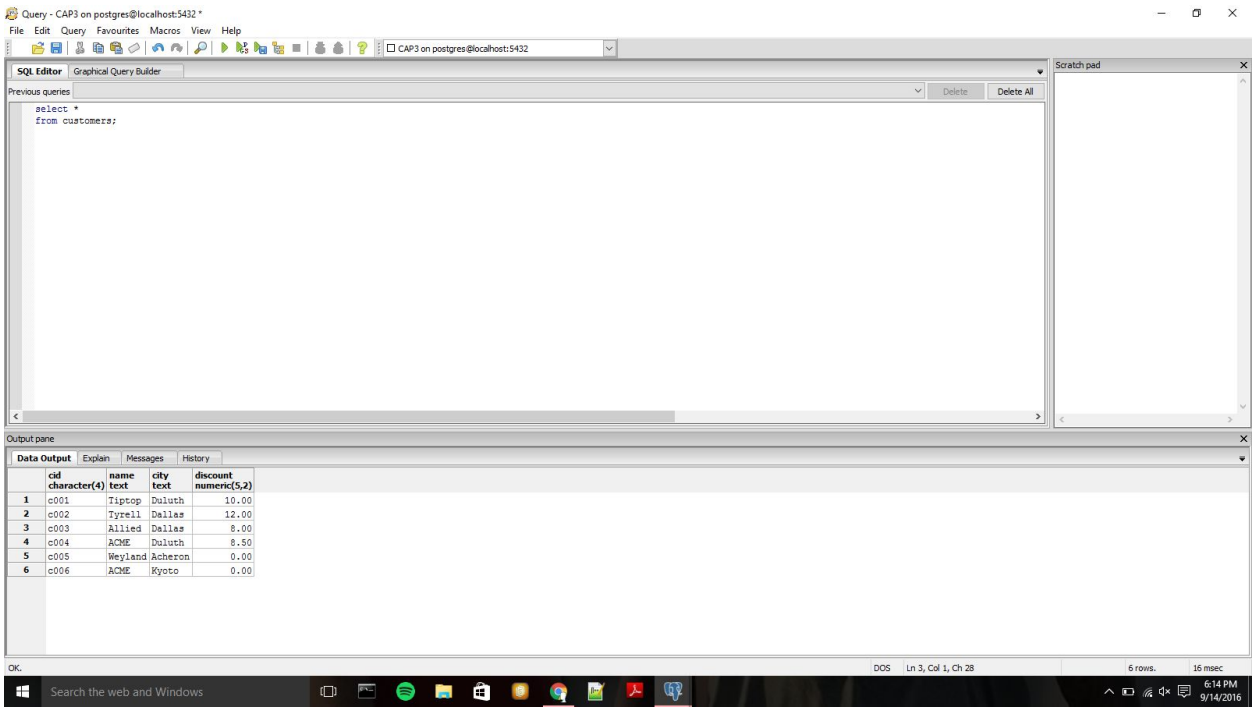
value because that would be used as the primary key since there are no repeated jersey numbers on one team.

4.

The first normal form rule states that each value in the relational data table are atomic, or indivisible. This means that each element within a column must contain only one value pertaining to that column. For example, in the world of Pokemon, some Pokemon have one elemental type and others have two. In the Pokedex (a database of Pokemon), there are two columns for elemental types, *Primary Type* and *Secondary Type*, in order for the Pokemon with two types to be accurately represented.The Pokemon with only one type would have a null value in the *Secondary Type* column. If there were only one column for type, then you would have to store both type values into one column, thus nonatomic and the relational table would not be of the first normal form. This is important because it helps eliminate repetition within the table.

Accessing rows by content only is also an important rule in relational databases. This rule allows the user to reference information using context instead of using numbers to identify the column and row. If you were to identify the value by the column number and row, then the value that you referenced would have no context. For example, if I were to look at a relational table referring to the New York Giants stats against the Dallas Cowboys last Sunday, I could reference specific stats by searching by row and column number, and it would give me a numeric value, however there would be no context as to what stat I referred to. However, if you were to look up a specific stat by referencing the column title, such as touchdowns, you would have context to the value that you looked up.

Lastly, an important rule is that all rows are unique. This is important because if two rows were the same, then your table would have useless repetition. This also puts an emphasis on providing your relational table with a key, so that way you can easily reference the data within the table. Say you had a relational table to keep track of your company's sales with three fields, *FirstName, LastName, AmountPurchased*. It is possible that two people with the same name make the same purchase, which leads to repetition and provides no information about other information that you may have in your database, such as *address* or even *state*. If you were to add a key that would make the rows unique, then you would be able to distinguish which customer is which, providing you with more information.



Query - CAP3 on postgres@localhost:5432 *

File Edit Query Favourites Macros View Help

CAP3 on postgres@localhost:5432

SQL Editor | Graphical Query Builder

Previous queries | Delete | Delete All

```
select *
from agents;
```

Scratch pad

Output pane

Data Output | Explain | Messages | History

	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 3, Col 1, Ch 25 | 7 rows. | 16 msec

Search the web and Windows | 6:15 PM 9/14/2016

Query - CAP3 on postgres@localhost:5432 *

File Edit Query Favourites Macros View Help

CAP3 on postgres@localhost:5432

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 1, Ch 1 | 8 rows. | 13 msec

Search the web and Windows | 6:15 PM 9/14/2016

Query - CAP3 on postgres@localhost:5432 *

File Edit Query Favourites Macros View Help

CAP3 on postgres@localhost:5432

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 2, Col 12, Ch 22 | 14 rows. | 14 msec

Search the web and Windows

6:15 PM
9/14/2016