Madeline Atwood

Lab 2

September 10, 2020

Professor Labouseur

1) Screenshots

CAP/postgres@PostgreSQL

Query Editor | Query History

```
select *
from People;
select *
from Customers;
select *
from Agents;
select *
from Products;
select *
```

Data Output | Explain | Messages | Notifications

	pid [PK] integer	paymentterms text	discountpct numeric (5,2)
1	1	Net 30	21.12
2	4	Net 15	4.04
3	5	In Advance	5.50
4	7	On Receipt	2.00
5	8	Net 30	10.00

CAP/postgres@PostgreSQL

Query Editor | Query History | Scratch Pad

```
from Agents;
select *
from Products;
select *
from Orders;
```

Data Output | Explain | Messages | Notifications

	ordernum [PK] integer	dateordered date	custid integer	agentid integer	prodid character (3)	quantityordered integer	totalusd numeric (12,2)
1	1011	2020-01-23	1	2	p01	1100	58568.40
2	1012	2020-01-23	4	3	p03	1200	74871.83
3	1015	2020-01-23	5	3	p05	1000	15696.45
4	1016	2020-01-23	8	3	p01	1000	60750.00
5	1017	2020-02-14	1	3	p03	500	25643.88
6	1018	2020-02-14	1	3	p04	600	22244.16
7	1019	2020-02-14	1	2	p02	400	1735.36
8	1020	2020-02-14	4	5	p07	600	575.76
9	1021	2020-02-14	4	5	p01	1000	64773.00
10	1022	2020-03-15	1	3	p06	450	709.92
11	1023	2020-03-15	1	2	p05	500	6550.98
12	1024	2020-03-15	5	2	p01	880	56133.00
13	1025	2020-04-01	8	3	p07	888	799.20
14	1026	2020-05-01	8	5	p03	808	47282.54

Query Editor Query History

```
select *
from People;
select *
from Customers;
select *
from Agents;
select *
from Products;
select *
```

Data Output Explain Messages Notifications

	pid [PK] integer	prefix text	firstname text	lastname text	suffix text	homecity text	dob date
1	1	Dr.	Neil	Peart	Ph.D.	Toronto	1952-09...
2	2	Ms.	Regina	Schock	[null]	Toronto	1957-08...
3	3	Mr.	Bruce	Crump	Jr.	Jacksonville	1957-07...
4	4	Mr.	Todd	Sucherman	[null]	Chicago	1969-05...
5	5	Mr.	Bernard	Purdie	[null]	Teaneck	1939-06...
6	6	Ms.	Demetra	Plakas	Esq.	Santa Monica	1960-11...
7	7	Ms.	Terri Lyne	Carrington	[null]	Boston	1965-08...
8	8	Dr.	Bill	Bruford	Ph.D.	Kent	1949-05...
9	9	Mr.	Alan	White	III	Pelton	1949-06...

CAP/postgres@PostgreSQL

Query Editor Query History

```
select *
from People;
select *
from Customers;
select *
from Agents;
select *
from Products;
select *
```

Data Output Explain Messages Notifications

	prodid [PK] character (3)	name text	city text	qtyonhand integer	priceusd numeric (10,2)
1	p01	Heisen...	Dallas	47	67.50
2	p02	Univers...	Newark	2399	5.50
3	p03	Comm...	Duluth	1979	65.02
4	p04	LCARS ...	Duluth	3	47.00
5	p05	Remo d...	Dallas	8675309	16.61
6	p06	Trapper...	Dallas	1982	2.00
7	p07	Flux Ca...	Newark	1007	1.00
8	p08	HAL 90...	Newark	200	1.25
9	p09	Red Ba...	Toronto	1	379000.47

CAP/postgres@PostgreSQL

Query Editor Query History

```
select *
from People;
select *
from Customers;
select *
from Agents;
select *
from Products;
select *
```

Data Output Explain Messages Notifications

	pid [PK] integer	paymentterms text	discountpct numeric (5,2)
1	1	Net 30	21.12
2	4	Net 15	4.04
3	5	In Advance	5.50
4	7	On Receipt	2.00
5	8	Net 30	10.00

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

When creating databases, it is very important that information is certain and precise inorder to keep all of the data on the table accurate and less likely to be confused. To abide by this idea, there are relational rules. Relational rules tell us what variations in table structure are permitted and limit possible retrieval operations. The superkey is a subset of columns that distinguishes between any two rows of the table, an example of this would be the CUSTOMERS table because the single cid column distinguishes between any two rows since it is a customer identifier, it is unique for each row.

A candidate key is the minimal length of a super key. The value for a candidate key should always be unique and none-null. Candidate keys are often known as the various keys of a relation, the name implying that there is a selection process where one of the candidates will be designed as the primary key. According to definition 2.4.3, a primary key is the candidate key chosen by the database designer to uniquely identify specific rows. On the topic of keys, it is important to note that there is an important relationship between primary keys and foreign keys. A foreign key is a value in one table that must match the primary key of another table. The relationship between primary and foreign keys (making sure that they match) are called referential integrity which insures consistency and accuracy. Data cannot become information without referential integrity and with referential integrity we have quality data.

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.

An example of a strong database that has largely effected most of my life without me realizing is my grades. Since I was little my teachers and professors have used an electronic grading system to enter all my grades to properly track my progress and just keep all of my information in order. During such a large age of technology I realized that if its on the internet and keeping track of some kind of information, chances are that it is entered into a database somewhere along the way. My report card consistently has emitted information from my school's gradebook database. If I were to create a gradebook table, I would title it with my name and have 6 columns. The first would consist of the course identifier code (ex CMPT-308, String and Int), the second would be the actual course name (Database Management, String), third would be the grade (A, Character), fourth would be the class CRN number (11361, Int), fifth would be the professor's first name (Alan, String) and sixth would be the professor's last name (Labouseur, String). Since null values are generally unknown or not yet defined and a primary key cannot take on a null value because it is a unique identifier, the only values in the table that cannot be null are the identifier code (because this will identify what class you are in) and the CRN number (because this specifies the section). Arguably you could say that null could be used for the identifier code if you have the CRN because the CRN will identify the code under its own database, but occasionally the identifier code can become primary when speaking about the number of credits offered in the course or categorizing it into a subject (computer science). A primary key would be the CRN number column because that uniquely identifies each class (no

class has the same CRN). If I were to create another table that needed to list my schedule, I could use the CRN to find the specifics of that class sections.

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

   a. The "First normal form" rule

   The first normal form rule states that "In defining tables, the relational model insists that columns that have multi-valued attributes (sometimes called repeating fields) or have any internal structure (like a record) are not permitted. A table the obeys this is said to be a first normal form. This means that each value in a column must contain an atomic (single) value that cannot be subdivided. An example of this is instead of creating a column that lists all of the movies somebody rented (since it can vary), I could create a column that lists the number of movies rented and then have a separate table that lists the movie information based off of the number of movies and the ID from the first table. If that is too much you could always put the movie name (or primary identifier) in the column that requires that information then just have another row later on with the customer's same information but just change the movie name information in that column. This is also important because a column should contain data that is all the same type, so by creating a separate table with the movie information per ID of user then we are preventing redundancy and creating a stronger relationship between the primary and foreign keys. It is also important to ensure that each column has a unique name, an example of this would be a first name column and a last name column. Although these

are not primary keys because people can occasionally have the same name, it is still important to note that we do not want two columns that both are titled “name” due to how easy that would be to mix up. Sometimes you can’t tell if somebody’s last name is their first name.

b. The “access rows by content only” rule

The access rows by content only rule pretty much means what it’s called- you can only retrieve rows by their content which is the attribute value that exist in each row. This means that there should be no partial dependency between the rows (ex: row numbering should not be what identifies the row, it should be identified with the candidate key). Since tables are sets, elements are in no intrinsic order, so you should never identify a row based off of what number it is when you could down from the top. Inorder to enure that this happens, the first normal form rule must be in place to allow unique identifiers for each row. Instead of asking “what is the name of the student in row 4?” you should be asking “what is the name of the student with the CWID of 20111920?” You never know when a table is going to be updated or rearranged and suddenly the CWID that was in row 4 is now in row 6 and you were using the rows as the identifier. Use the content in the row! Use a superkey! Not the row number!

c. The “all rows must be unique” rule

The last normal form rule states that two tuples in a relation (rows in a table) cannot be identical in all column values at once. This should be a no-brainer but it is still important to note that a set should never contain two identical elements because each tuple must be

unique. If you have a good database, you will have referential integrity which will insure that you have a good (and existing) relationship between foreign and primary keys. The query language cannot retrieve a statement uniquely unless there is a way of distinguishing the different rows. If a column is a superkey, there should be no repeating of that value either way and every row should have a superkey in it. A good way to insure that you have a superkey is to create something such as a user ID which is just an artificial key.