BuzzFeed

Database Design Project

By: Dayna Eidle

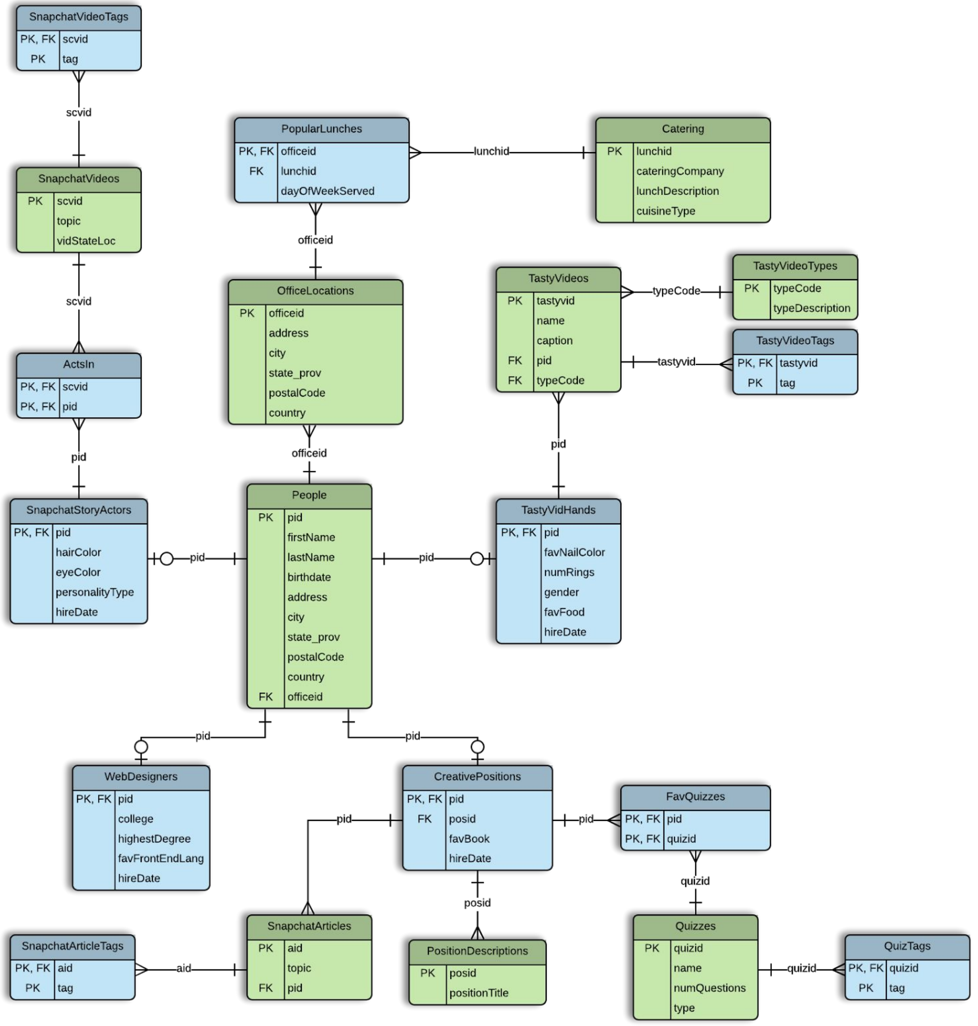
Table of Contents

Executive Summary

BuzzFeed contacted me because they were in need of a database for their company. With the numerous sectors of the company and types of employees, they needed something to keep track of it all in an organized manner. They love to expand and create new sections of their organization and so the database they asked for needed to be easy to modify and understand. The BuzzFeed database I have created meets those standards.

The following documentation goes into the major details of the database. This includes people, employee types, office locations, videos, articles, and more. In addition, some stored procedures, triggers, and views have been created to emphasize and adjust certain parts of the database based on specific scenarios. BuzzFeed has always been a quirky company and no one would want any restrictions to be put on their ability to be fun and creative because the database created for them didn't allow it. With this database, BuzzFeed can easily continue to add new sections of the company, employee types, and any other cool features they desire.

E/R Diagram


SnapchatVideoTags
PK, FK scvid
PK tag
scvid
SnapchatVideos
PK scvid
topic
vidStateLoc
scvid
ActsIn
PK, FK scvid
PK, FK pid
pid
SnapchatStoryActors
PK, FK pid
hairColor
eyeColor
personalityType
hireDate
pid
PopularLunches
PK, FK officeid
FK lunchid
dayOfWeekServed
lunchid
Catering
PK lunchid
cateringCompany
lunchDescription
cuisineType
officeid
OfficeLocations
PK officeid
address
city
state_prov
postalCode
country
officeid
People
PK pid
firstName
lastName
birthdate
address
city
state_prov
postalCode
country
FK officeid
TastyVideos
PK tastyvid
name
caption
FK pid
FK typeCode
typeCode
TastyVideoTypes
PK typeCode
typeDescription
tastyvid
TastyVideoTags
PK, FK tastyvid
PK tag
pid
TastyVidHands
PK, FK pid
favNailColor
numRings
gender
favFood
hireDate
pid
pid
pid
WebDesigners
PK, FK pid
college
highestDegree
favFrontEndLang
hireDate
CreativePositions
PK, FK pid
FK posid
favBook
hireDate
pid
pid
FavQuizzes
PK, FK pid
PK, FK quizid
quizid
posid
SnapchatArticleTags
PK, FK aid
PK tag
aid
SnapchatArticles
PK aid
topic
FK pid
PositionDescriptions
PK posid
positionTitle
Quizzes
PK quizid
name
numQuestions
type
quizid
QuizTags
PK, FK quizid
PK tag

Tables

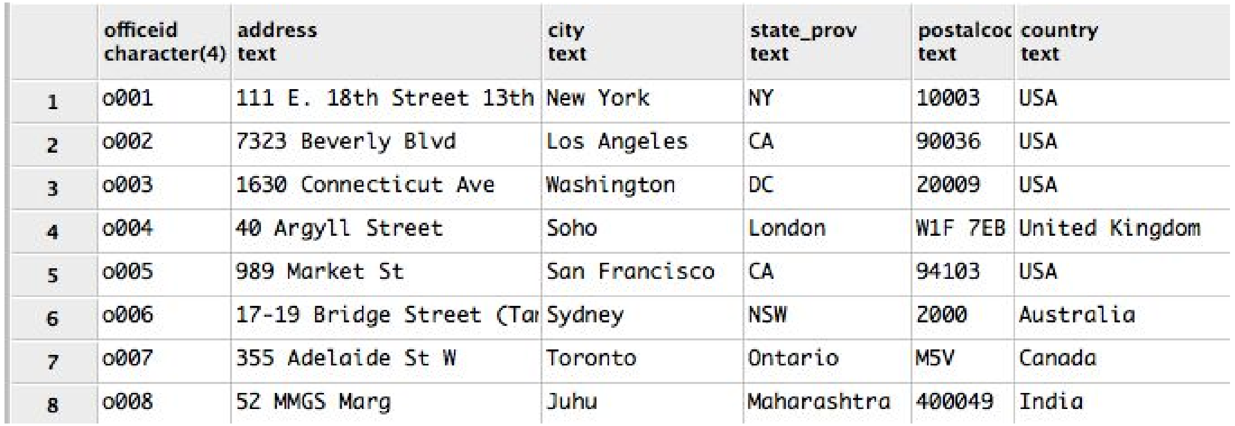
OfficeLocations Table

The **OfficeLocations** table lists the officeid, address, and country of each BuzzFeed office around the world.

Functional Dependencies:

officeid ⟶ address, city, state_prov, postalCode, country

```
CREATE TABLE OfficeLocations(
  officeid          char(4) not null,
  address           text not null,
  city              text not null,
  state_prov        text not null,
  postalCode        text not null,
  country           text not null,
 primary key(officeid)
);
```



	officeid character(4)	address text	city text	state_prov text	postalcoc text	country text
1	o001	111 E. 18th Street 13th	New York	NY	10003	USA
2	o002	7323 Beverly Blvd	Los Angeles	CA	90036	USA
3	o003	1630 Connecticut Ave	Washington	DC	20009	USA
4	o004	40 Argyll Street	Soho	London	W1F 7EB	United Kingdom
5	o005	989 Market St	San Francisco	CA	94103	USA
6	o006	17-19 Bridge Street (Tar	Sydney	NSW	2000	Australia
7	o007	355 Adelaide St W	Toronto	Ontario	M5V	Canada
8	o008	52 MMGS Marg	Juhu	Maharashtra	400049	India

People Table

The **People** table lists all employees and their general information within the BuzzFeed corporation, regardless of their position in the company. It lists firstName, lastName, birthdate, address, and officeid.

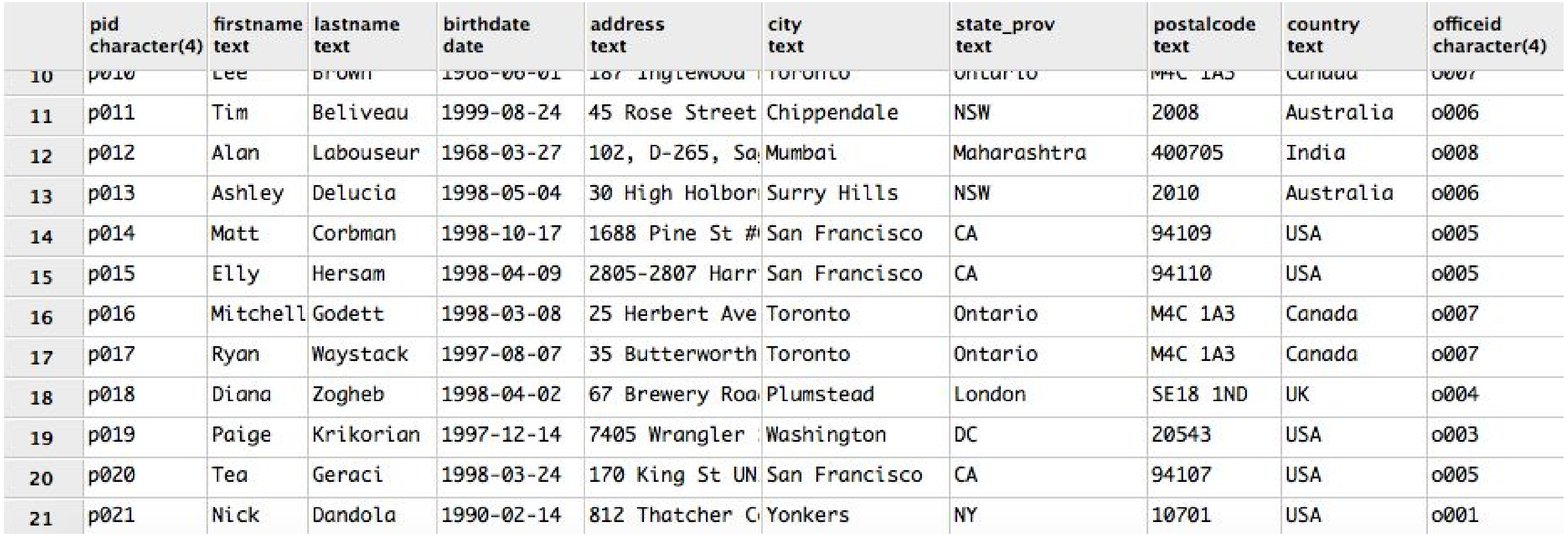
```
CREATE TABLE People (
  pid             char(4) not null,
  firstName       text not null,
  lastName        text not null,
  birthdate       date not null,
  address         text not null,
  city            text not null,
  state_prov      text not null,
  postalCode      text not null,
  country         text not null,
  officeid        char(4) not null references OfficeLocations(officeid),
 primary key(pid)
);
```

Functional Dependencies:

pid $\longrightarrow$ firstName, lastname, birthdate, address, city, state_prov, postalCode, country, officeid

People Table (cont.)

Sample Data:



	pid character(4)	firstname text	lastname text	birthdate date	address text	city text	state_prov text	postalcode text	country text	officeid character(4)
10	p010	Lee	Brown	1968-06-01	187 Inglewood	Toronto	Ontario	M4C 1A3	Canada	o007
11	p011	Tim	Beliveau	1999-08-24	45 Rose Street	Chippendale	NSW	2008	Australia	o006
12	p012	Alan	Labouseur	1968-03-27	102, D-265, Sa	Mumbai	Maharashtra	400705	India	o008
13	p013	Ashley	Delucia	1998-05-04	30 High Holbor	Surry Hills	NSW	2010	Australia	o006
14	p014	Matt	Corbman	1998-10-17	1688 Pine St #	San Francisco	CA	94109	USA	o005
15	p015	Elly	Hersam	1998-04-09	2805-2807 Harr	San Francisco	CA	94110	USA	o005
16	p016	Mitchell	Godett	1998-03-08	25 Herbert Ave	Toronto	Ontario	M4C 1A3	Canada	o007
17	p017	Ryan	Waystack	1997-08-07	35 Butterworth	Toronto	Ontario	M4C 1A3	Canada	o007
18	p018	Diana	Zogheb	1998-04-02	67 Brewery Roa	Plumstead	London	SE18 1ND	UK	o004
19	p019	Paige	Krikorian	1997-12-14	7405 Wrangler	Washington	DC	20543	USA	o003
20	p020	Tea	Geraci	1998-03-24	170 King St UN	San Francisco	CA	94107	USA	o005
21	p021	Nick	Dandola	1990-02-14	812 Thatcher C	Yonkers	NY	10701	USA	o001

SnapchatStoryActors Table

The **SnapchatStoryActors** table is a specific job position table branching off of the People table listing the actors for the videos on the BuzzFeed Snapchat story. This table lists the hairColor, eyeColor, personalityType, and hireDate of the actor

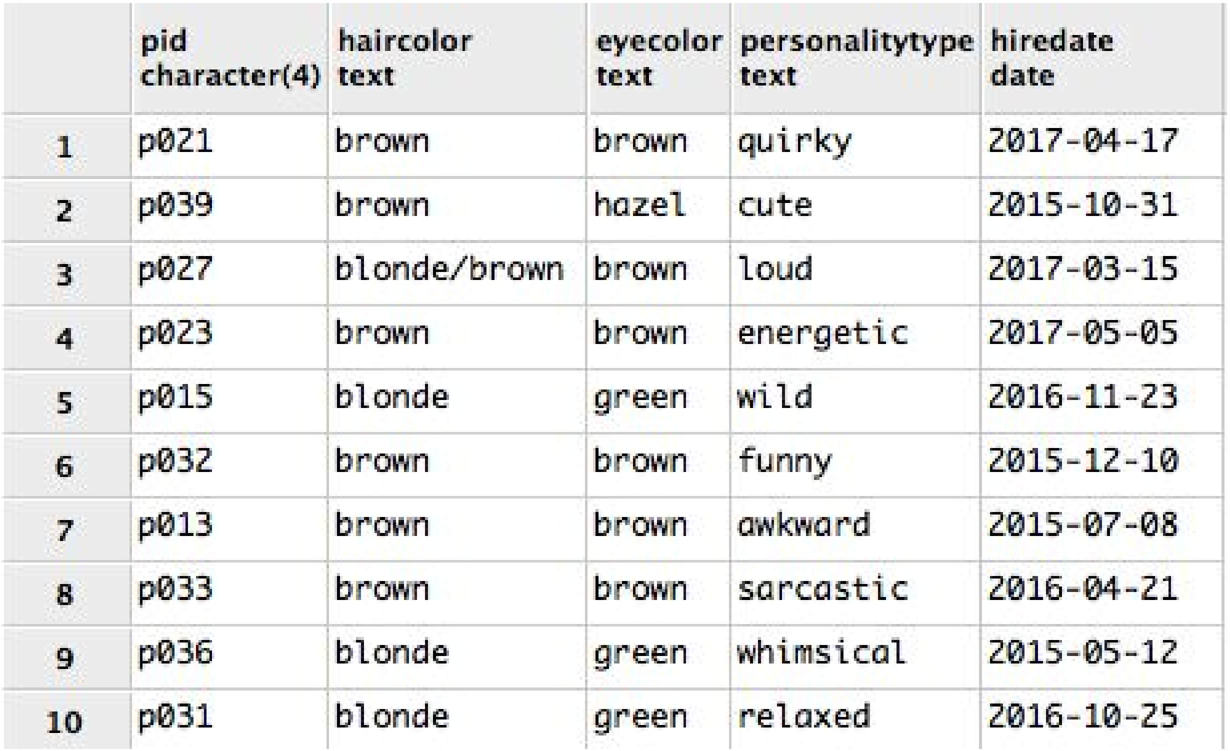
```
CREATE TABLE SnapchatStoryActors (
  pid              char(4) not null references People(pid),
  hairColor        text,
  eyeColor         text,
  personalityType  text,
  hireDate         date not null,
 primary key(pid)
);
```

Functional Dependencies:

pid ⟶ hairColor, eyeColor, personalityType, hireDate

SnapchatStoryActors Table (cont.)

Sample Data:



	pid character(4)	haircolor text	eyecolor text	personalitytype text	hiredate date
1	p021	brown	brown	quirky	2017-04-17
2	p039	brown	hazel	cute	2015-10-31
3	p027	blonde/brown	brown	loud	2017-03-15
4	p023	brown	brown	energetic	2017-05-05
5	p015	blonde	green	wild	2016-11-23
6	p032	brown	brown	funny	2015-12-10
7	p013	brown	brown	awkward	2015-07-08
8	p033	brown	brown	sarcastic	2016-04-21
9	p036	blonde	green	whimsical	2015-05-12
10	p031	blonde	green	relaxed	2016-10-25

TastyVidHands Table

The **TastyVidHands** table lists the people whose hands are in the Tasty food videos. It lists each person's pid, favNailColor, gender, favFood, and hireDate.

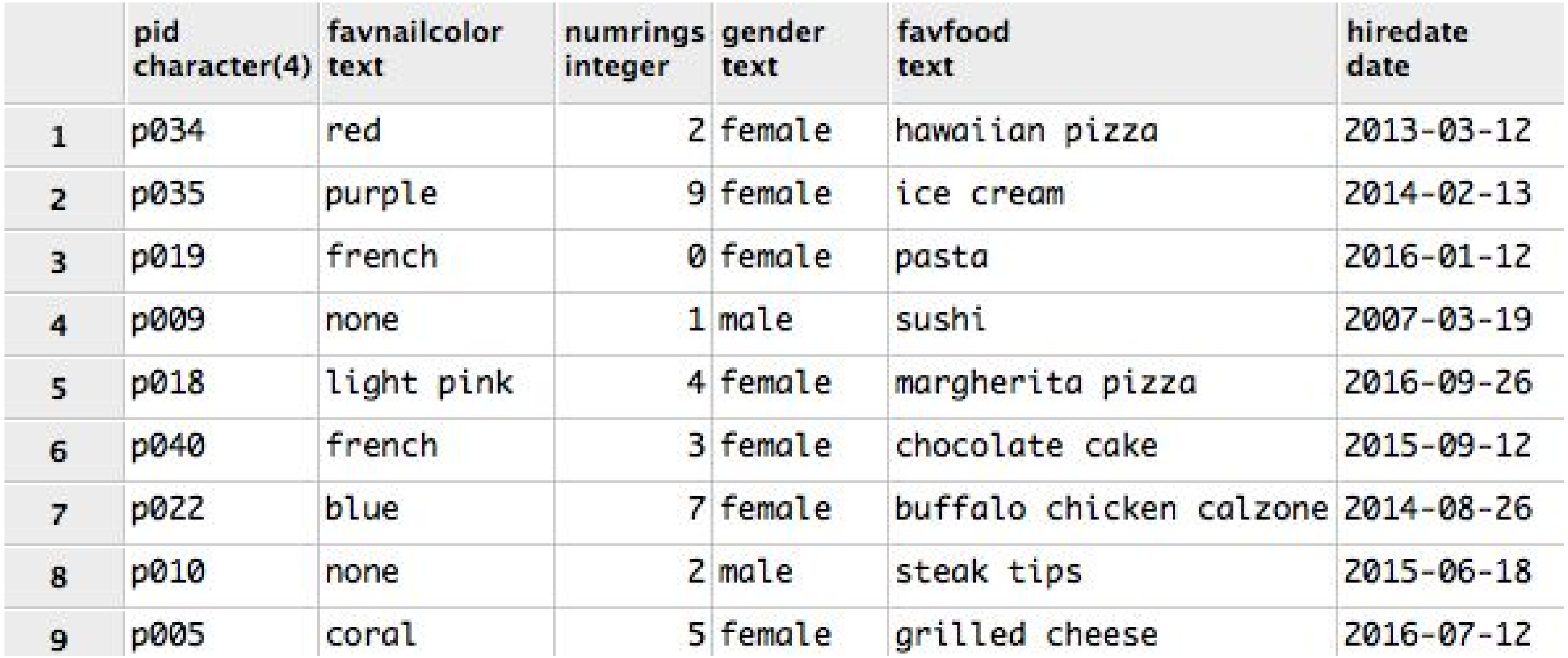
```
CREATE TABLE TastyVidHands (
  pid             char(4) not null references People(pid),
  favNailColor    text,
  numRings        integer,
  gender          text CHECK (gender = 'female' or gender =
                  'male' or gender = 'other'),
  favFood         text,
  hireDate        date not null,
 primary key(pid)
);
```

Functional Dependencies:

pid $\longrightarrow$ favNailColor, numRings, gender, favFood, hireDate

TastyVidHands Table (cont.)

Sample Data:

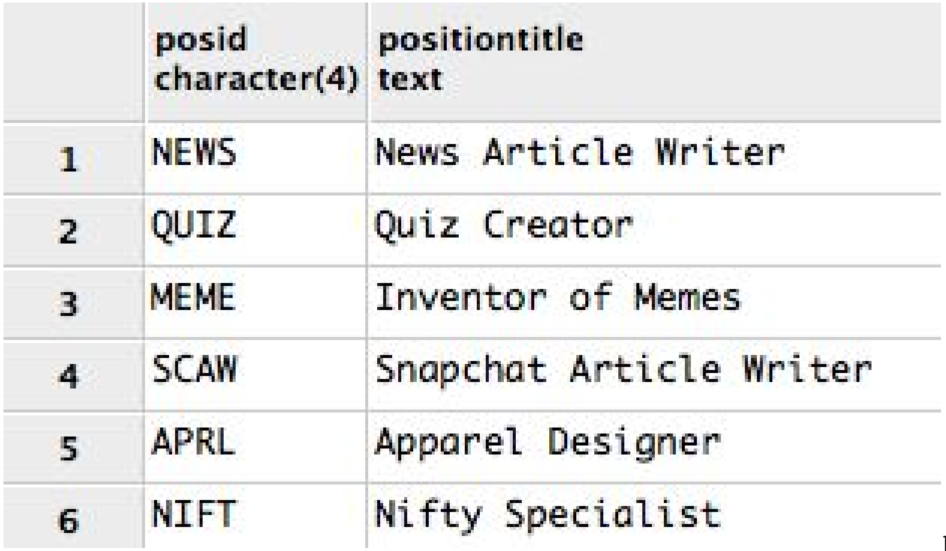


	pid character(4)	favnailcolor text	numrings integer	gender text	favfood text	hiredate date
1	p034	red	2	female	hawaiian pizza	2013-03-12
2	p035	purple	9	female	ice cream	2014-02-13
3	p019	french	0	female	pasta	2016-01-12
4	p009	none	1	male	sushi	2007-03-19
5	p018	light pink	4	female	margherita pizza	2016-09-26
6	p040	french	3	female	chocolate cake	2015-09-12
7	p022	blue	7	female	buffalo chicken calzone	2014-08-26
8	p010	none	2	male	steak tips	2015-06-18
9	p005	coral	5	female	grilled cheese	2016-07-12

PositionDescriptions Table

The **PositionDescriptions** table lists the posid and description for each of the different types of jobs that employees with a creative position can have.

```
CREATE TABLE PositionDescriptions(
  posid                    char(4) not null,
  positionTitle            text,
 primary key(posid)
);
```



	posid character(4)	positiontitle text
1	NEWS	News Article Writer
2	QUIZ	Quiz Creator
3	MEME	Inventor of Memes
4	SCAW	Snapchat Article Writer
5	APRL	Apparel Designer
6	NIFT	Nifty Specialist

Functional Dependencies:

posid $\longrightarrow$ positionTitle

CreativePositions Table

The **CreativePositions** table lists all the employees who hold a creative position at BuzzFeed. It lists the pid, posid, favBook, and hireDate of the employee.

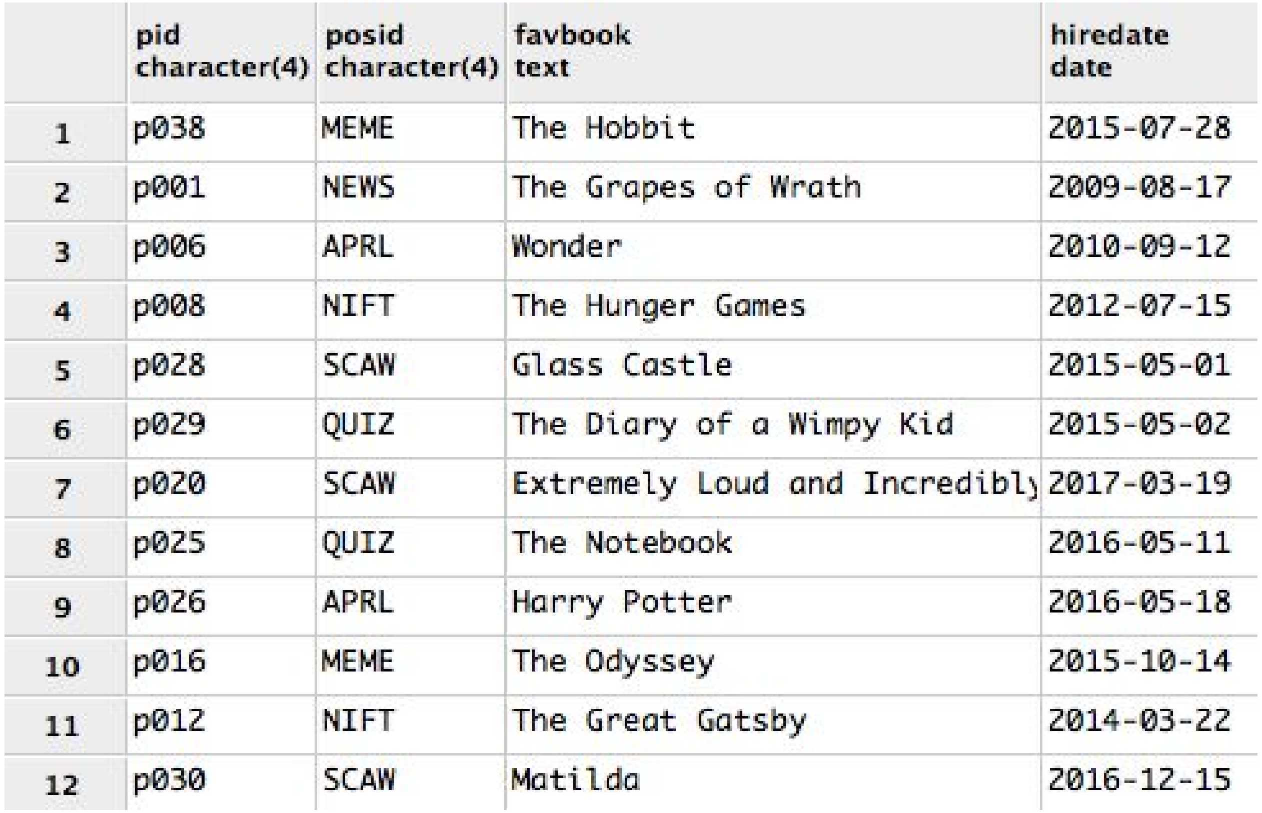
```
CREATE TABLE CreativePositions (
  pid            char(4) not null unique references People(pid),
  posid          char(4) not null references PositionDescriptions(posid) ,
  favBook        text,
  hireDate       date not null,
 primary key(pid)
);
```

Functional Dependencies:

pid $\longrightarrow$ posid, favBook, hireDate

CreativePositions Table (cont.)

Sample Data:



	pid character(4)	posid character(4)	favbook text	hiredate date
1	p038	MEME	The Hobbit	2015-07-28
2	p001	NEWS	The Grapes of Wrath	2009-08-17
3	p006	APRL	Wonder	2010-09-12
4	p008	NIFT	The Hunger Games	2012-07-15
5	p028	SCAW	Glass Castle	2015-05-01
6	p029	QUIZ	The Diary of a Wimpy Kid	2015-05-02
7	p020	SCAW	Extremely Loud and Incredibl	2017-03-19
8	p025	QUIZ	The Notebook	2016-05-11
9	p026	APRL	Harry Potter	2016-05-18
10	p016	MEME	The Odyssey	2015-10-14
11	p012	NIFT	The Great Gatsby	2014-03-22
12	p030	SCAW	Matilda	2016-12-15

WebDesigners Table

The **WebDesigners** table lists the employees who work on creating and maintaining the BuzzFeed website. This table lists the pid, college, highestDegree, favFrontEndLang, and hireDate for each person.

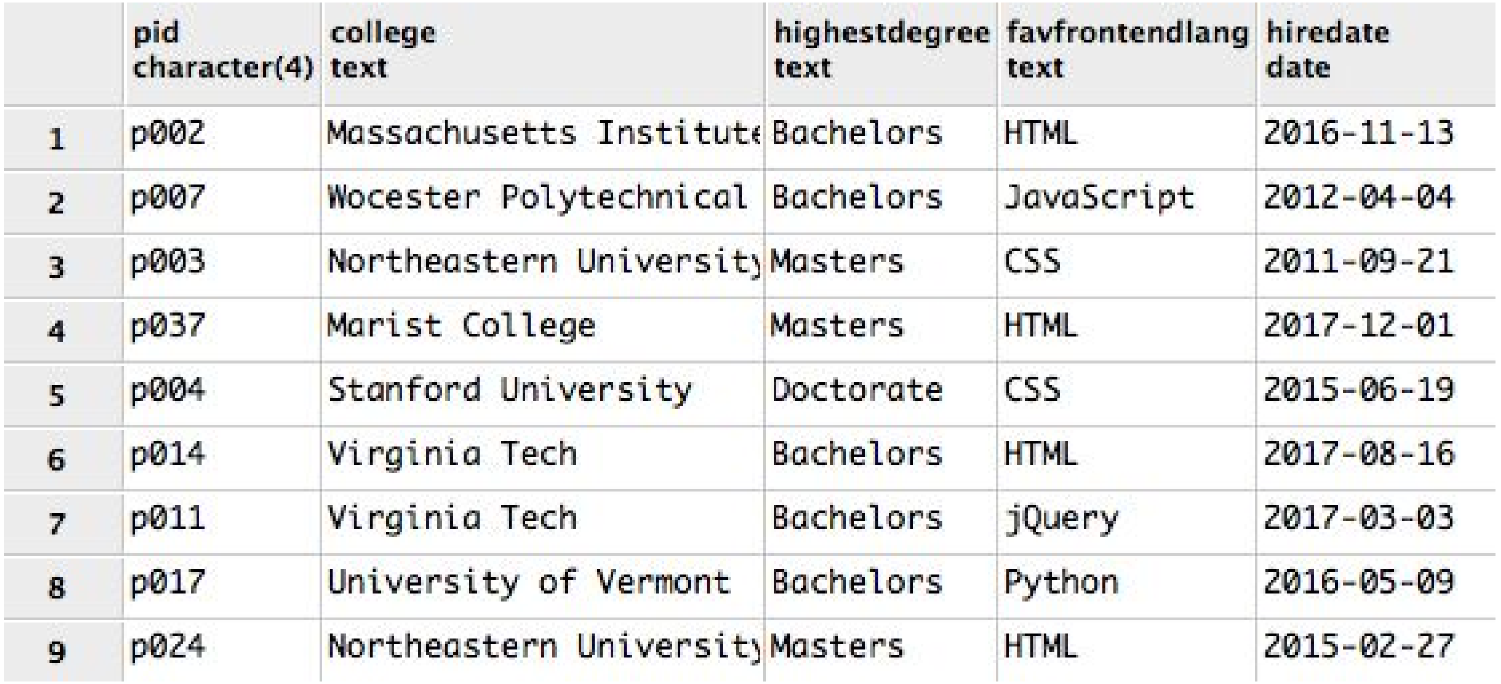
```
CREATE TABLE WebDesigners (
  pid                char(4) not null references People(pid),
  college            text,
  highestDegree      text,
  favFrontEndLang    text,
  hireDate           date not null,
 primary key(pid)
);
```

Functional Dependencies:

pid $\longrightarrow$ college, highestDegree, favFrontEndLang, hireDate

WebDesigners Table (cont.)

Sample Data:



	pid	college	highestdegree	favfrontendlang	hiredate
	character(4)	text	text	text	date
1	p002	Massachusetts Institute	Bachelors	HTML	2016-11-13
2	p007	Wocester Polytechnical	Bachelors	JavaScript	2012-04-04
3	p003	Northeastern Universit	Masters	CSS	2011-09-21
4	p037	Marist College	Masters	HTML	2017-12-01
5	p004	Stanford University	Doctorate	CSS	2015-06-19
6	p014	Virginia Tech	Bachelors	HTML	2017-08-16
7	p011	Virginia Tech	Bachelors	jQuery	2017-03-03
8	p017	University of Vermont	Bachelors	Python	2016-05-09
9	p024	Northeastern Universit	Masters	HTML	2015-02-27

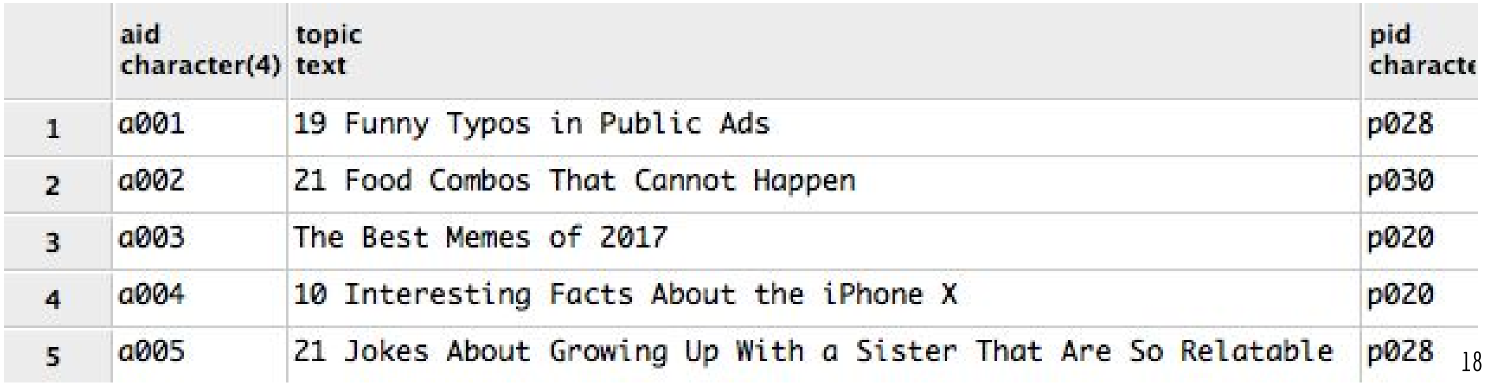
SnapchatArticles Table

The **SnapchatArticles** table lists the articles that are shown on the BuzzFeed Snapchat story. The table lists the aid, topic, and pid (author) of each article.

```
CREATE TABLE SnapchatArticles(
  aid          char(4) not null,
  topic        text,
  pid  char(4) not null references CreativePositions(pid),
 primary key(aid)
);
```

Functional Dependencies:

aid ⟶ topic, pid



	aid character(4)	topic text	pid characte
1	a001	19 Funny Typos in Public Ads	p028
2	a002	21 Food Combos That Cannot Happen	p030
3	a003	The Best Memes of 2017	p020
4	a004	10 Interesting Facts About the iPhone X	p020
5	a005	21 Jokes About Growing Up With a Sister That Are So Relatable	p028

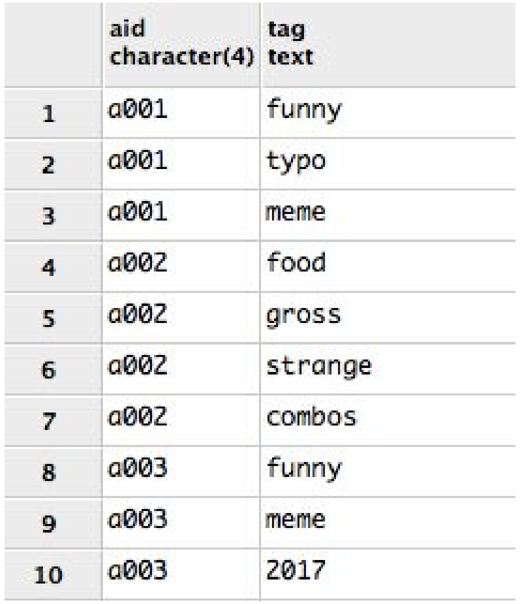
SnapchatArticleTags Table

The **SnapchatArticleTags** table lists the various tags for each article. These tags are categories that the article could fall under. It lists the aid and tag for each article.

```
CREATE TABLE SnapchatArticleTags(
  aid            char(4) not null references SnapChatArticles(aid),
  tag            text,
 primary key(aid, tag)
);
```

Functional Dependencies:

aid, tag $\longrightarrow$



	aid character(4)	tag text
1	a001	funny
2	a001	typo
3	a001	meme
4	a002	food
5	a002	gross
6	a002	strange
7	a002	combos
8	a003	funny
9	a003	meme
10	a003	2017

Quizzes Table

The **Quizzes** table lists the various quizzes that can be found on the BuzzFeed website. The quizzes have a variety of topics and formats. The table lists the quizid, name, numQuestions, and type of each quiz.

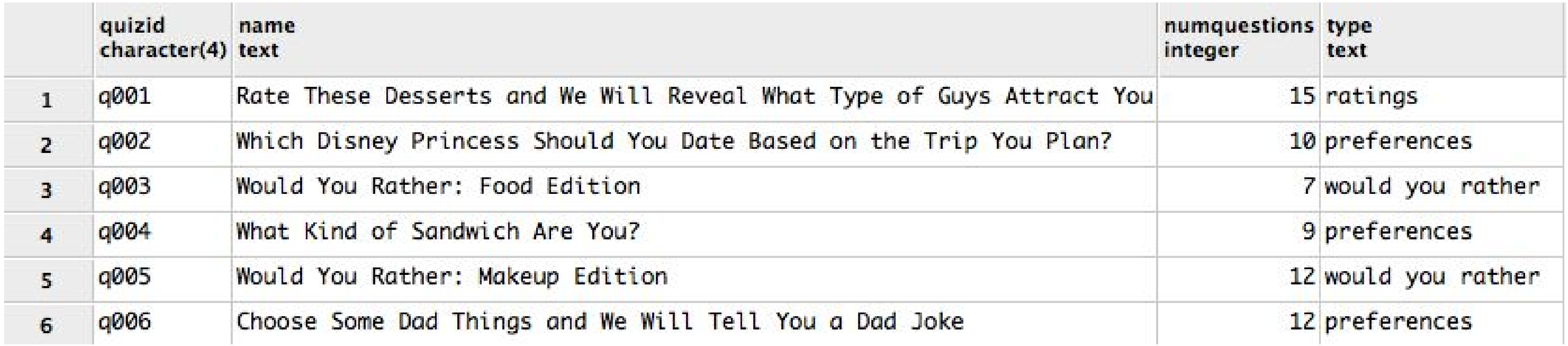
```
CREATE TABLE Quizzes(
  quizid              char(4) not null,
  name                text,
  numQuestions        integer,
  type                text,
 primary key(quizid)
);
```

Functional Dependencies:

quizid $\longrightarrow$ name, numQuestions, type

Quizzes Table (cont.)

Sample Data:



	quizid character(4)	name text	numquestions integer	type text
1	q001	Rate These Desserts and We Will Reveal What Type of Guys Attract You	15	ratings
2	q002	Which Disney Princess Should You Date Based on the Trip You Plan?	10	preferences
3	q003	Would You Rather: Food Edition	7	would you rather
4	q004	What Kind of Sandwich Are You?	9	preferences
5	q005	Would You Rather: Makeup Edition	12	would you rather
6	q006	Choose Some Dad Things and We Will Tell You a Dad Joke	12	preferences

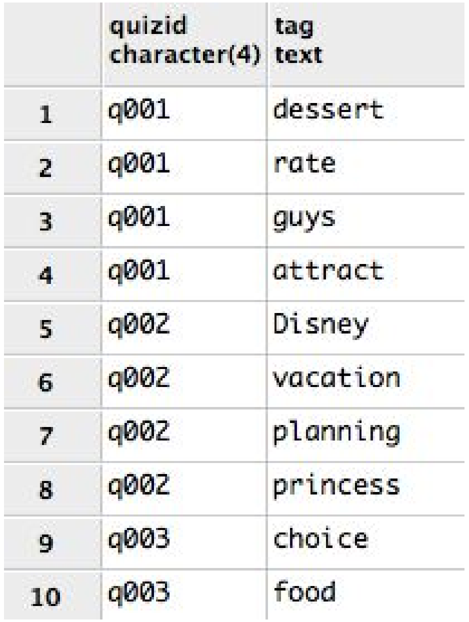
QuizTags Table

The **QuizTags** table shows the tags for each quiz on the website. The table lists the quizid and tag for each quiz.

```
CREATE TABLE QuizTags(
  quizid     char(4) not null references Quizzes(quizid),
  tag          text,
 primary key(quizid, tag)
);
```

Functional Dependencies:

quizid, tag $\longrightarrow$



	quizid character(4)	tag text
1	q001	dessert
2	q001	rate
3	q001	guys
4	q001	attract
5	q002	Disney
6	q002	vacation
7	q002	planning
8	q002	princess
9	q003	choice
10	q003	food

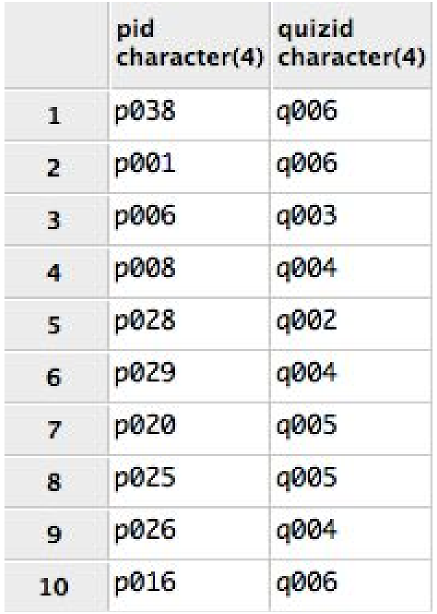
FavQuizzes Table

The **FavQuizzes** table lists the favorite quizzes of each of the employees who hold a creative position. This table lists the pid and quizid of the favorite quiz for each person.

```
CREATE TABLE FavQuizzes(
  pid         char(4) not null references CreativePositions(pid),
  quizid      char(4) not null references Quizzes(quizid),
 primary key(pid, quizid)
);
```

Functional Dependencies:

pid, quizid →



	pid character(4)	quizid character(4)
1	p038	q006
2	p001	q006
3	p006	q003
4	p008	q004
5	p028	q002
6	p029	q004
7	p020	q005
8	p025	q005
9	p026	q004
10	p016	q006

Catering Table

The **Catering** table lists the various catering companies BuzzFeed uses to cater the weekly company lunches. It lists the lunchid, cateringCompany, lunchDescription, and cuisineType for each lunch that each company caters.

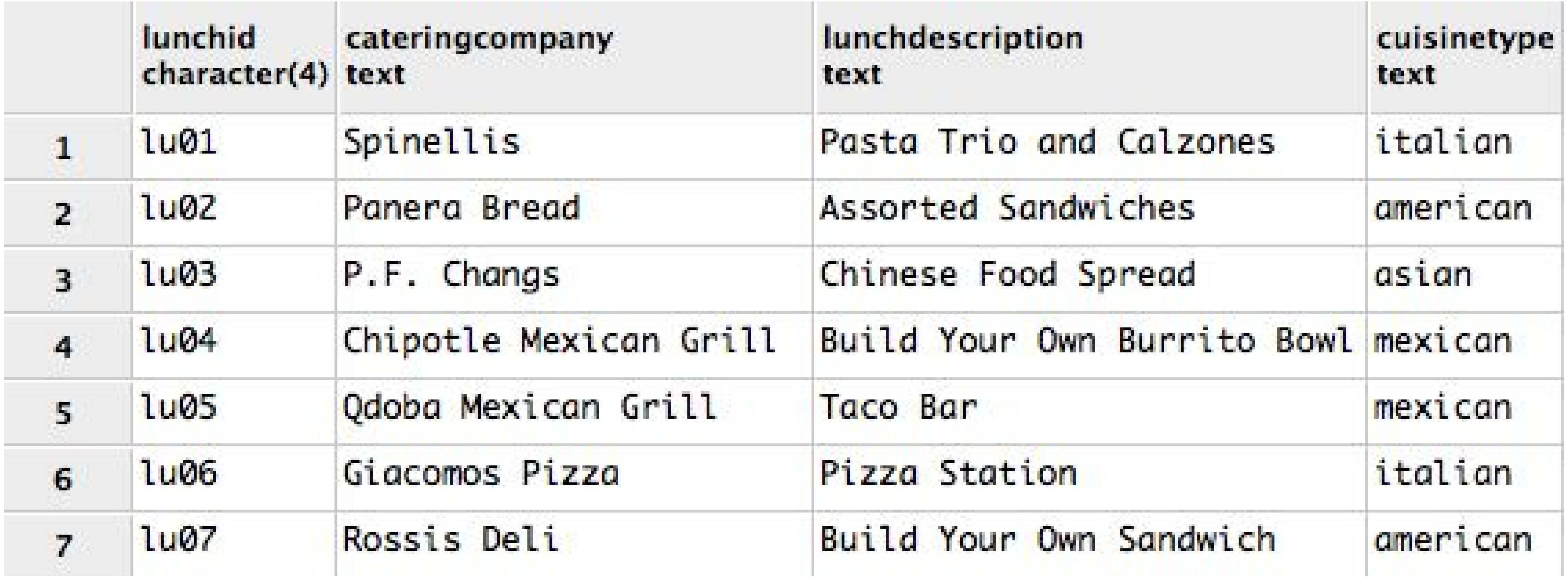
```
CREATE TABLE Catering(
  lunchid                 char(4) not null,
  cateringCompany         text,
  lunchDescription        text,
  cuisineType             text,
 primary key(lunchid)
);
```

Functional Dependencies:

lunchid $\longrightarrow$ cateringCompany, lunchDescription, cuisineType

Catering Table (cont.)

Sample Data:



	lunchid character(4)	cateringcompany text	lunchdescription text	cuisinetype text
1	lu01	Spinellis	Pasta Trio and Calzones	italian
2	lu02	Panera Bread	Assorted Sandwiches	american
3	lu03	P.F. Changs	Chinese Food Spread	asian
4	lu04	Chipotle Mexican Grill	Build Your Own Burrito Bowl	mexican
5	lu05	Qdoba Mexican Grill	Taco Bar	mexican
6	lu06	Giacomos Pizza	Pizza Station	italian
7	lu07	Rossis Deli	Build Your Own Sandwich	american

PopularLunches Table

The **PopularLunches** table lists the most popular lunch for each office. The table shows the officeid, lunchid, and dayOfWeekServed for each favored lunch.

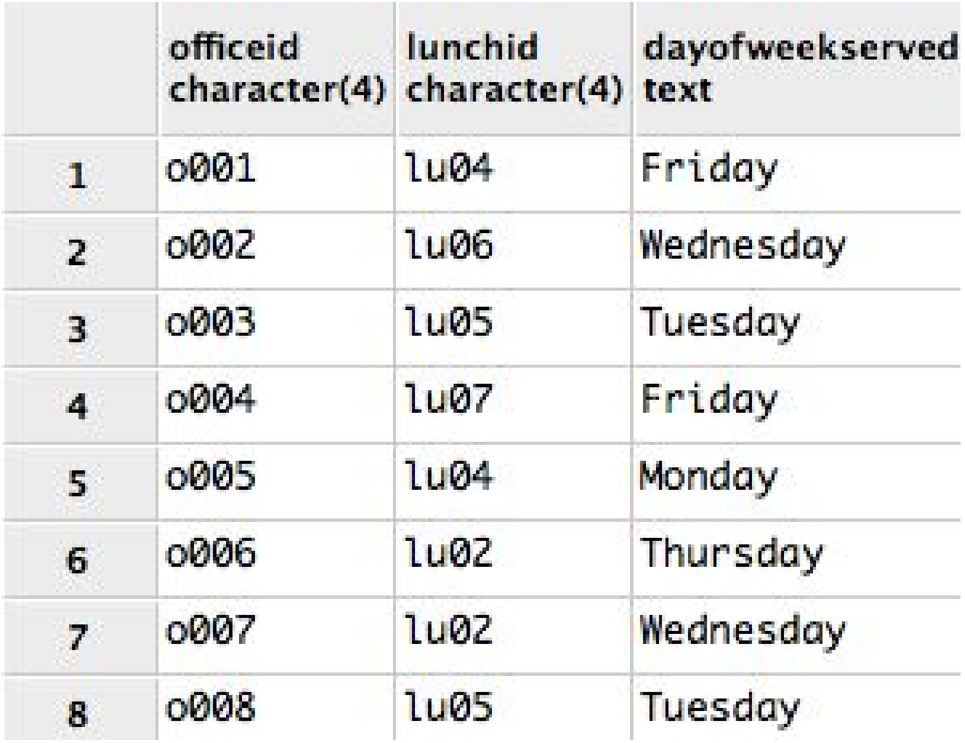
```
CREATE TABLE PopularLunches(
  officeid              char(4) not null references OfficeLocations(officeid),
  lunchid               char(4) not null references Catering(lunchid),
  dayOfWeekServed       text CHECK(dayOfWeekServed = 'Monday' or dayOfWeekServed = 'Tuesday' or
                        dayOfWeekServed = 'Wednesday' or dayOfWeekServed = 'Thursday' or
                        dayOfWeekServed = 'Friday' or dayOfWeekServed = 'Saturday' or
                        dayOfWeekServed = 'Sunday'),
 primary key(officeid)
);
```

Functional Dependencies:

officeid $\longrightarrow$ lunchid, dayOfWeekServed

PopularLunches Table (cont.)

Sample Data:



	officeid character(4)	lunchid character(4)	dayofweekserved text
1	o001	lu04	Friday
2	o002	lu06	Wednesday
3	o003	lu05	Tuesday
4	o004	lu07	Friday
5	o005	lu04	Monday
6	o006	lu02	Thursday
7	o007	lu02	Wednesday
8	o008	lu05	Tuesday

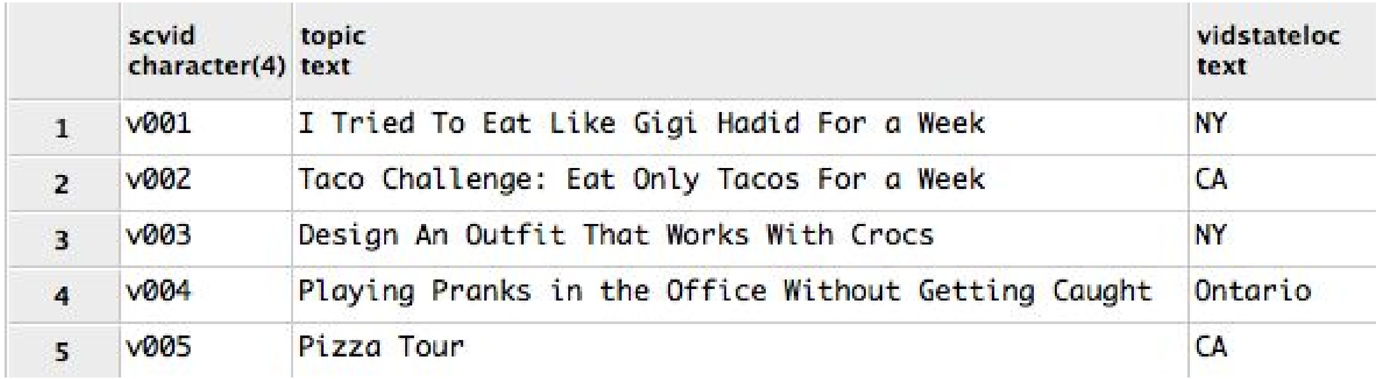
SnapchatVideos Table

The **SnapchatVideos** table shows the videos that are apart of the BuzzFeed Snapchat story. It lists the scvid, topic, and vidLoc for each video.

```
CREATE TABLE SnapchatVideos(
  scvid             char(4) not null,
  topic             text,
  vidStateLoc       text,
 primary key(scvid)
);
```

Functional Dependencies:

scvid $\longrightarrow$ topic, vidStateLoc

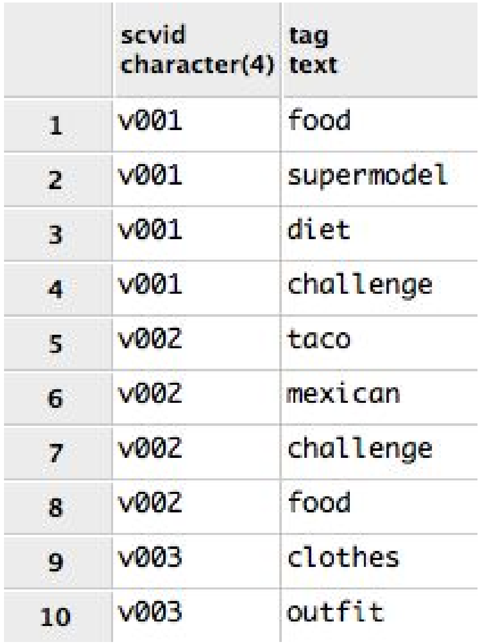


	scvid character(4)	topic text	vidstateloc text
1	v001	I Tried To Eat Like Gigi Hadid For a Week	NY
2	v002	Taco Challenge: Eat Only Tacos For a Week	CA
3	v003	Design An Outfit That Works With Crocs	NY
4	v004	Playing Pranks in the Office Without Getting Caught	Ontario
5	v005	Pizza Tour	CA

SnapchatVideoTags Table

The **SnapchatVideoTags** table shows the tags for each given video on the BuzzFeed Snapchat story. It includes the scvid and tag for each video.

```
CREATE TABLE SnapchatVideoTags(
  scvid      char(4) not null references SnapchatVideos(scvid),
  tag        text,
 primary key(scvid, tag)
);
```



	scvid character(4)	tag text
1	v001	food
2	v001	supermodel
3	v001	diet
4	v001	challenge
5	v002	taco
6	v002	mexican
7	v002	challenge
8	v002	food
9	v003	clothes
10	v003	outfit

Functional Dependencies:

scvid, tag →

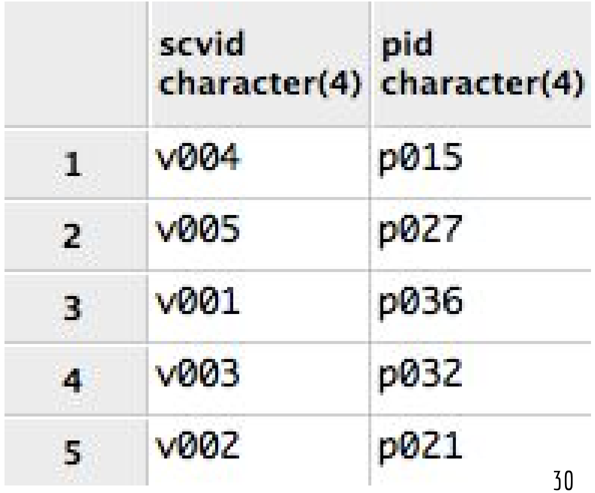
ActsIn Table

The **ActsIn** table shows the actors who are featured in each video on the BuzzFeed Snapchat story. It lists the scvid and pid for each actor/video combination.

```
CREATE TABLE ActsIn(
  scvid       char(4) not null references SnapchatVideos(scvid),
  pid         char(4) not null references SnapchatStoryActors(pid),
 primary key(scvid, pid)
);
```

Functional Dependencies:

scvid, pid $\longrightarrow$



	scvid character(4)	pid character(4)
1	v004	p015
2	v005	p027
3	v001	p036
4	v003	p032
5	v002	p021

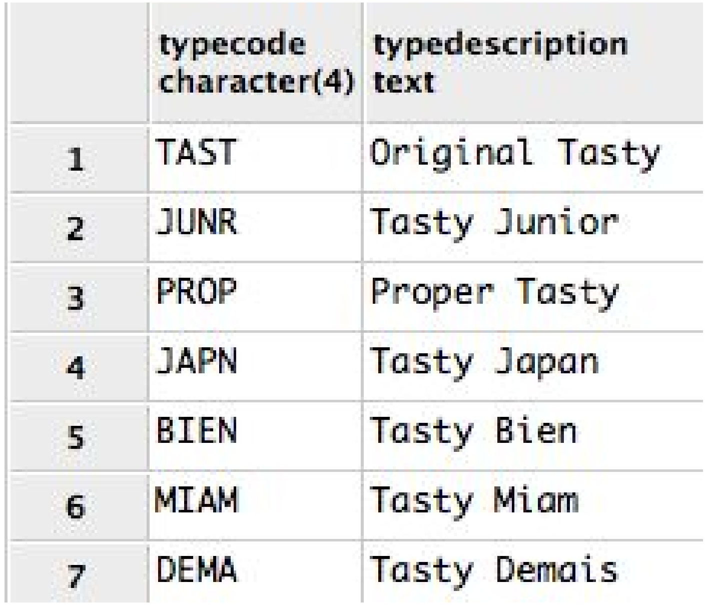
TastyVideoTypes Table

The **TastyVideoTypes** table describes all the different sections of BuzzFeed's Tasty. It lists the typeCode and typeDescription for each section.

```
CREATE TABLE TastyVideoTypes(
  typeCode             char(4) not null,
  typeDescription      text,
 primary key(typeCode)
);
```

Functional Dependencies:

typeCode $\longrightarrow$ typeDescription



	typecode character(4)	typedescription text
1	TAST	Original Tasty
2	JUNR	Tasty Junior
3	PROP	Proper Tasty
4	JAPN	Tasty Japan
5	BIEN	Tasty Bien
6	MIAM	Tasty Miam
7	DEMA	Tasty Demais

TastyVideos Table

The **TastyVideos** table includes all of the Tasty videos that are shown on the website and on social media. It lists tastyvid, name, caption, pid, and typeCode.

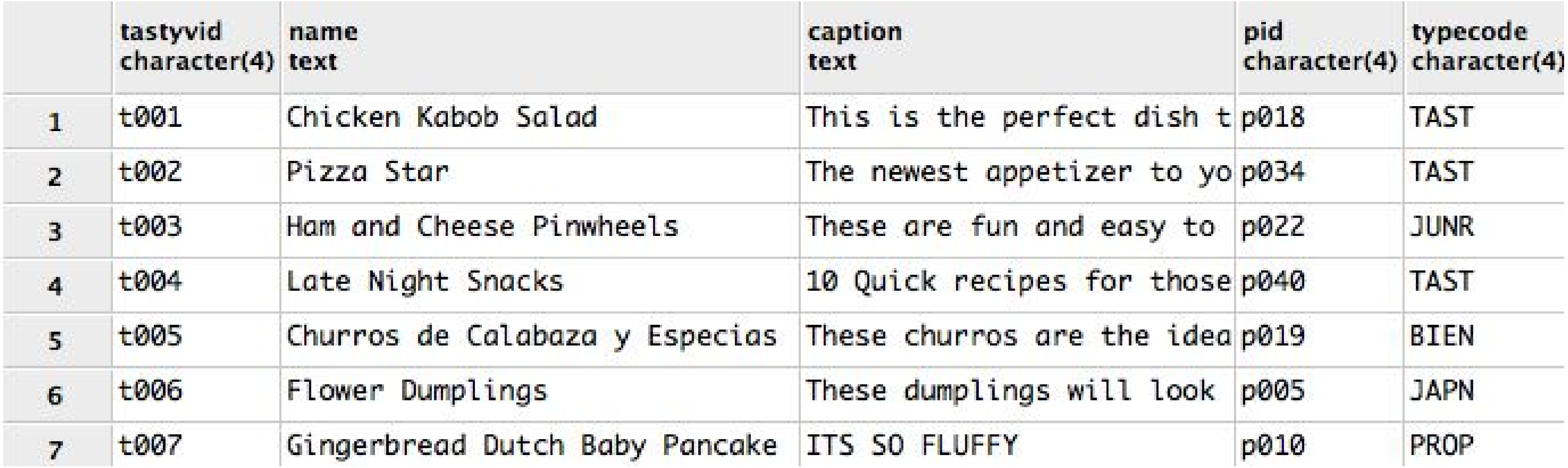
Functional Dependencies:

tastyvid $\longrightarrow$ name, caption, pid, typeCode

```
CREATE TABLE TastyVideos(
  tastyvid        char(4) not null,
  name            text,
  caption         text,
  pid             char(4) not null references TastyVidHands(pid),
  typeCode        char(4) not null references TastyVideoTypes(typeCode),
 primary key (tastyvid)
);
```

TastyVideos Table (cont.)

Sample Data:



	tastyvid character(4)	name text	caption text	pid character(4)	typecode character(4)
1	t001	Chicken Kabob Salad	This is the perfect dish t	p018	TAST
2	t002	Pizza Star	The newest appetizer to yo	p034	TAST
3	t003	Ham and Cheese Pinwheels	These are fun and easy to	p022	JUNR
4	t004	Late Night Snacks	10 Quick recipes for those	p040	TAST
5	t005	Churros de Calabaza y Especias	These churros are the idea	p019	BIEN
6	t006	Flower Dumplings	These dumplings will look	p005	JAPN
7	t007	Gingerbread Dutch Baby Pancake	ITS SO FLUFFY	p010	PROP

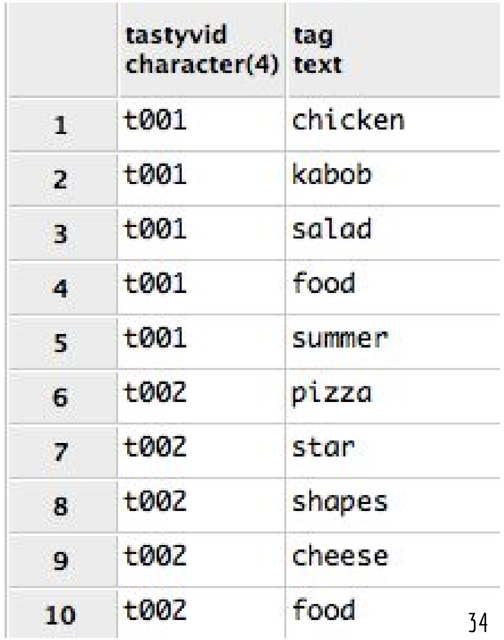
TastyVideoTags Table

The **TastyVideoTags** table shows all of the tags associated with each Tasty video. It lists the tastyvid and tag for each video,

```
CREATE TABLE TastyVideoTags(
  tastyvid          char(4) not null references TastyVideos(tastyvid),
  tag               text not null,
 primary key(tastyvid, tag)
);
```

Functional Dependencies:

tastyvid, tag $\longrightarrow$

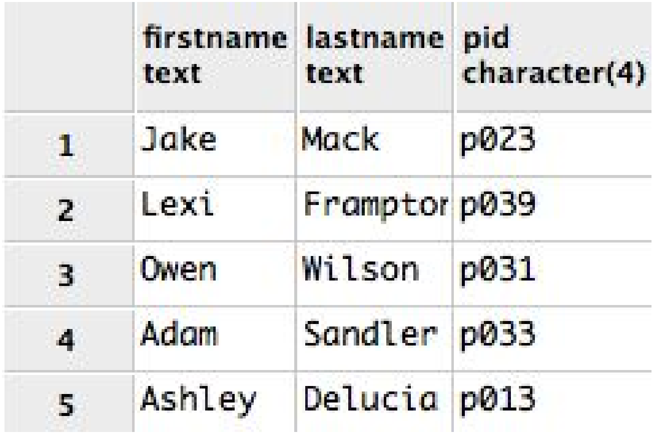


	tastyvid character(4)	tag text
1	t001	chicken
2	t001	kabob
3	t001	salad
4	t001	food
5	t001	summer
6	t002	pizza
7	t002	star
8	t002	shapes
9	t002	cheese
10	t002	food

Views and Reports

UnusedActors View

The **UnusedActors** view shows the firstName, lastName, and pid of Snapchat story actors who are not currently in a video.



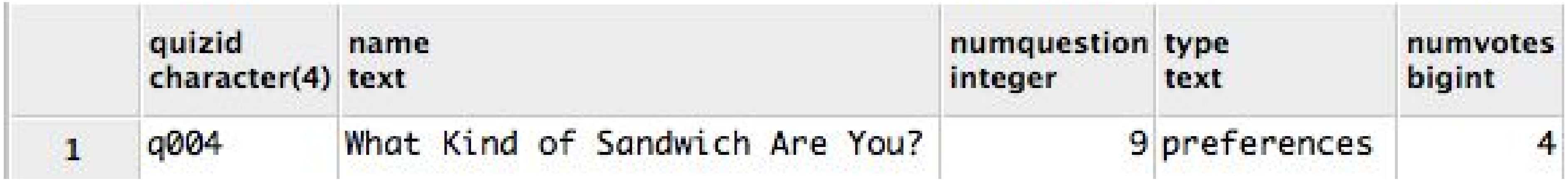
	firstname text	lastname text	pid character(4)
1	Jake	Mack	p023
2	Lexi	Framptor	p039
3	Owen	Wilson	p031
4	Adam	Sandler	p033
5	Ashley	Delucia	p013

```
create view UnusedActors (firstName, lastName, pid)
as
 select p.firstName, p.lastName, s.pid
 from snapchatstoryactors s left outer join people p on s.pid = p.pid
                                                  left outer join actsin a on s.pid = a.pid
where scvid is NULL;
```

MostPopularQuiz View

The **MostPopularQuiz** view shows the most popular quiz for the employees who hold a creative position.

```
create view MostPopularQuiz (quizid)
as
  select quizid, count(quizid) as numVotes
  from favQuizzes
  group by quizid
  order by count(quizid) DESC
  limit 1;
```

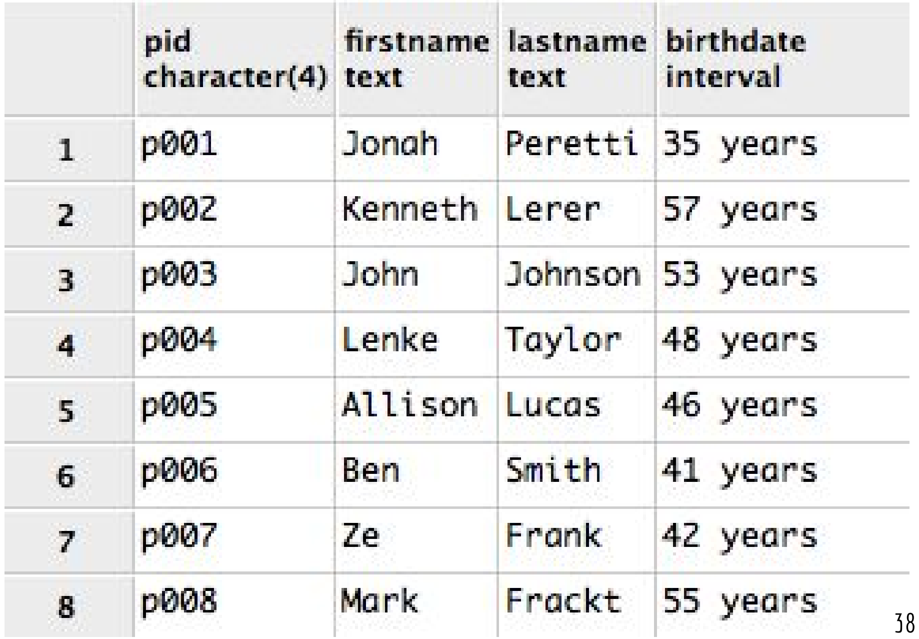


	quizid character(4)	name text	numquestion integer	type text	numvotes bigint
1	q004	What Kind of Sandwich Are You?	9	preferences	4

Ages View

The **Ages** view computes the age of each person based on their birthdate.

```
create view Ages(pid, firstName, lastName,
birthdate)
as
  select pid, firstName, lastName,
         date_trunc('year',age(birthdate))
  from people;
```



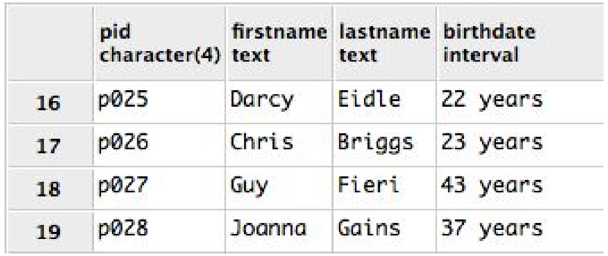
	pid character(4)	firstname text	lastname text	birthdate interval
1	p001	Jonah	Peretti	35 years
2	p002	Kenneth	Lerer	57 years
3	p003	John	Johnson	53 years
4	p004	Lenke	Taylor	48 years
5	p005	Allison	Lucas	46 years
6	p006	Ben	Smith	41 years
7	p007	Ze	Frank	42 years
8	p008	Mark	Frackt	55 years

Reports

❖ All people who are 21 and over

➢ Purposes: List for bar at holiday office parties, selective spots to visit/try out in Snapchat story videos

```
select *
from ages
where birthdate >= '21 years';
```

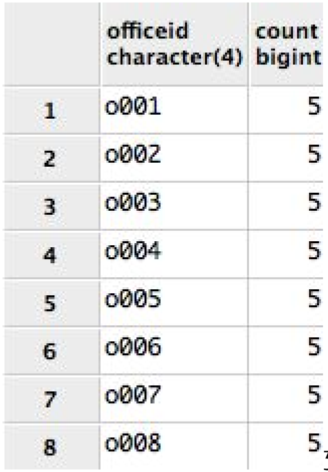


	pid character(4)	firstname text	lastname text	birthdate interval
16	p025	Darcy	Eidle	22 years
17	p026	Chris	Briggs	23 years
18	p027	Guy	Fieri	43 years
19	p028	Joanna	Gains	37 years

❖ Number of people at each office

➢ Note: In this snapshot, all offices do have the same number of people

```
select officeid, count(officeid)
from people
group by officeid
order by officeid ASC;
```



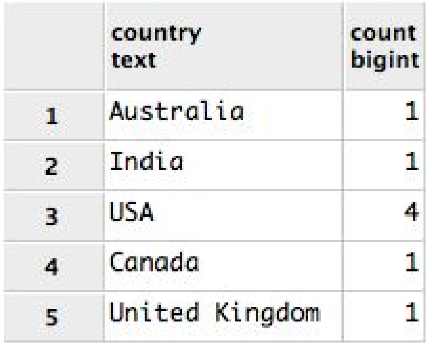
	officeid character(4)	count bigint
1	o001	5
2	o002	5
3	o003	5
4	o004	5
5	o005	5
6	o006	5
7	o007	5
8	o008	5

Reports (cont.)

❖ Number of offices by country

➢ Useful for looking at where to expand the company to in the world

```
select country, count(country)
from officelocations
group by country;
```



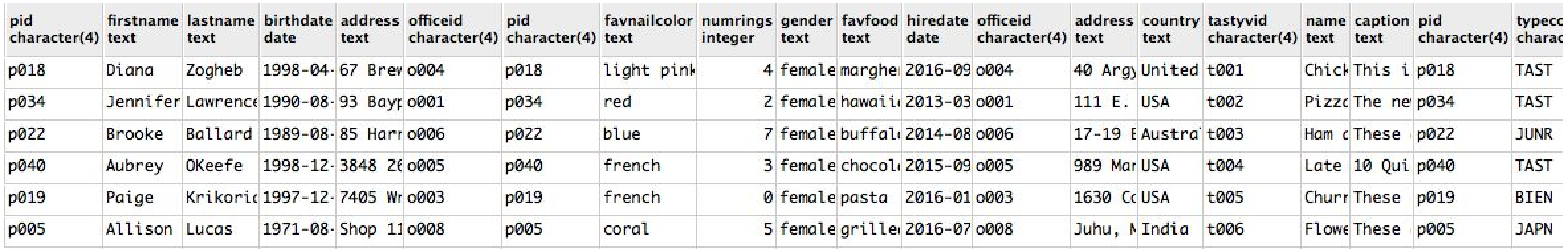
	country text	count bigint
1	Australia	1
2	India	1
3	USA	4
4	Canada	1
5	United Kingdom	1

❖ All info by employee position

➢ Ex: Tasty Video employees

➢ Can be easily changed for each position using different joins

```
select *
from people p inner join tastyvidhands t on p.pid = t.pid
        inner join officelocations l on p.officeid = l.officeid
        inner join tastyvideos v on t.pid = v.pid;
```



pid character(4)	firstname text	lastname text	birthdate date	address text	officeid character(4)	pid character(4)	favnailcolor text	numrings integer	gender text	favfood text	hiredate date	officeid character(4)	address text	country text	tastyvid character(4)	name text	caption text	pid character(4)	typecc charac
p018	Diana	Zogheb	1998-04-	67 Brev	o004	p018	light pink	4	female	marghe	2016-09	o004	40 Argy	United	t001	Chick	This i	p018	TAST
p034	Jennifer	Lawrence	1990-08-	93 Bayp	o001	p034	red	2	female	hawaii	2013-03	o001	111 E.	USA	t002	Pizz	The ne	p034	TAST
p022	Brooke	Ballard	1989-08-	85 Harr	o006	p022	blue	7	female	buffal	2014-08	o006	17-19 E	Austra	t003	Ham c	These	p022	JUNR
p040	Aubrey	OKeefe	1998-12-	3848 26	o005	p040	french	3	female	chocol	2015-09	o005	989 Mar	USA	t004	Late	10 Qui	p040	TAST
p019	Paige	Krikoric	1997-12-	7405 Wr	o003	p019	french	0	female	pasta	2016-01	o003	1630 Cc	USA	t005	Churr	These	p019	BIEN
p005	Allison	Lucas	1971-08-	Shop 11	o008	p005	coral	5	female	grille	2016-07	o008	Juhu, M	India	t006	Flowe	These	p005	JAPN

Stored Procedures and Triggers

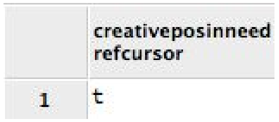
creativePosInNeed

```
create or replace function creativePosInNeed(char(4))
returns refcursor as
$$
declare
   selectedPos        char(4) :=$1;
   count        int;
begin
           count = (select count(posid)
                      from creativepositions
                      where posid=selectedPos
                      group by posid);

           if (count < 2) then
                     return true;
           else
                     return false;
           end if;
end;
$$
language plpgsql;
```

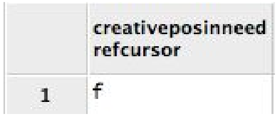
The **creativePosInNeed** stored procedu takes in the posid of a given creative position and returns true if that position is in need of more employees or false if there are currently a sufficient amount of employees in that department.

```
select creativePosInNeed('NEWS');
```



	creativeposinneed refcursor
1	t

```
select creativePosInNeed('SCAW);
```



	creativeposinneed refcursor
1	f

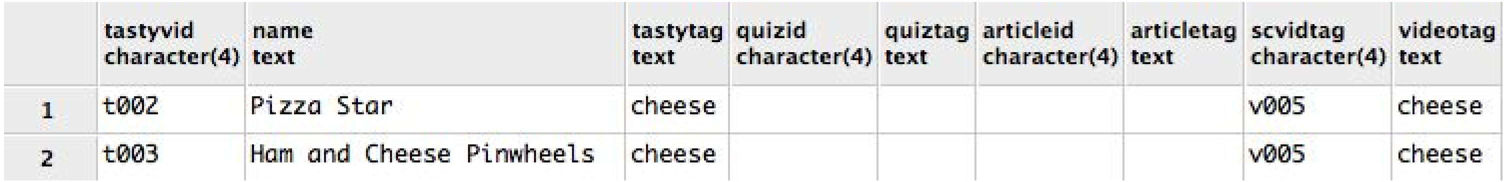
commonTastyTags

```
create or replace function commonTastyTags(text, REFCURSOR) returns refcursor as
$$
declare
   commonTag text       := $1;
   resultset   REFCURSOR := $2;
begin
   open resultset for
   select distinct tv.tastyvid, tv.name, t.tag as tastyTag , qt.quizid,
                               qt.tag as quizTag, a.aid as articleid, a.tag as articleTag,v.scvid as scVidTag,
                               v.tag as videoTag
   from TastyVideoTags t inner join TastyVideos tv on t.tastyvid = tv.tastyvid
                                 left outer join QuizTags qt on t.tag = qt.tag
                                 left outer join SnapchatArticleTags a on t.tag = a.tag
                                 left outer join SnapchatVideoTags v on t.tag = v.tag
   where t.tag = commonTag;
   return resultset;
end;
$$
language plpgsql;
```

The **commonTastyTag** stored procedure takes in a tag and lists all the Tasty videos with this tag, in addition to the Snapchat articles, videos, and quizzes that also have this tag.

commonTastyTags (cont.)

select commonTastyTags('cheese', 'results');
Fetch all from results;



	tastyvid character(4)	name text	tastytag text	quizid character(4)	quiztag text	articleid character(4)	articletag text	scvidtag character(4)	videotag text
1	t002	Pizza Star	cheese					v005	cheese
2	t003	Ham and Cheese Pinwheels	cheese					v005	cheese

❖ This stored procedure can be switched around so that the main item is something else. For example, here it is Tasty videos, but it can be changed to Snapchat articles by adjusting the joins statements.

quizLength

```
create or replace function quizLength()
returns trigger as
$$
begin
  if NEW.numQuestions > 25 then
    delete from Quizzes where numQuestions = NEW.numQuestions;
  end if;
  return NEW;
end;
$$
language plpgsql;
```

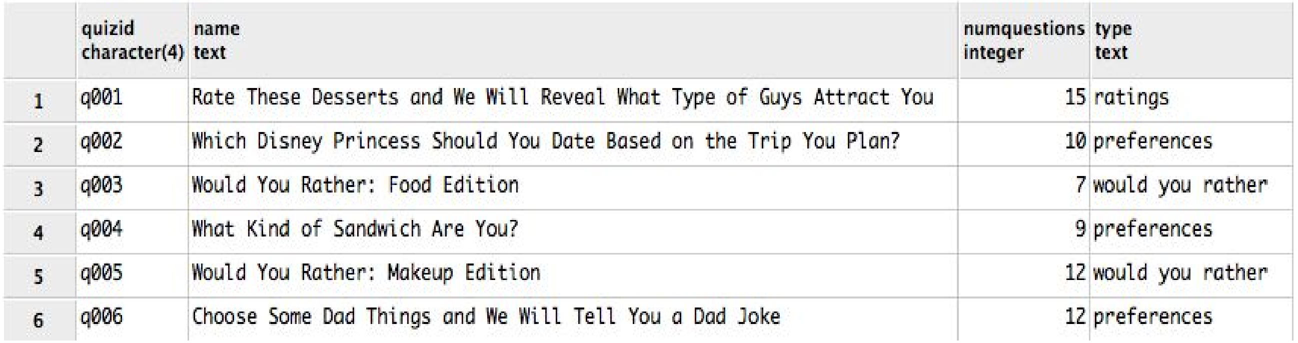
The **quizLength** stored procedure checks if a new quiz in the Quizzes table has more than 25 questions. If it does, it is deleted from the table because people will get bored with the quiz if it’s too long.

checkQuizLength

The **checkQuizLength** trigger sets off the quizLength stored procedure.

```
create trigger checkQuizLength
after insert on quizzes
for each row
execute procedure quizLength();
```

```
insert into Quizzes(quizid, name,
numquestions, type)
values('q007', 'Make The Perfect
           Cookie Dough And We Will
           Tell You How Many Kids You
           Will Have', 35,'preferences');
```



	quizid character(4)	name text	numquestions integer	type text
1	q001	Rate These Desserts and We Will Reveal What Type of Guys Attract You	15	ratings
2	q002	Which Disney Princess Should You Date Based on the Trip You Plan?	10	preferences
3	q003	Would You Rather: Food Edition	7	would you rather
4	q004	What Kind of Sandwich Are You?	9	preferences
5	q005	Would You Rather: Makeup Edition	12	would you rather
6	q006	Choose Some Dad Things and We Will Tell You a Dad Joke	12	preferences

After the above insert statement is executed, the **checkQuizLength** trigger also executes, causing the new quiz to be deleted from the Quizzes table because it has more than 25 questions.

Security

User Roles

Admin: Has access to entirety of the database
create role admin;
grant all on all tables in schema public to admin;

CEO: Only other member of BuzzFeed with total access to the database
create role CEO;
grant all on all tables in schema public to CEO;

departmentChairs: Have access to all parts of the various departments in case employees move around within the company - levels of access vary based on table
create role departmentChairs;
revoke all on all tables in schema public from departmentChairs;
grant select on QuizTags, FavQuizzes, SnapchatArticleTags, TastyVideoTags, SnapchatVideoTags to departmentChairs;
grant select, insert, update on PositionDescriptions, Quizzes, SnapchatArticles, TastyVideos, TastyVideoTypes, SnapchatVideos, ActsIn to departmentChairs;
grant select, insert, update, delete on CreativePositions, WebDesigners, TastyVidHands,
SnapchatStoryActors to departmentChairs;

Implementation Notes, Known Problems, and Future Enhancements

Implementation Notes

- Although not all columns have a “not null’ restriction on them, the database will be most functional and accurate if all data for each table is entered upon insert.
- There are no restrictions on tags for videos, articles, and quizzes, meaning that users with access to those tables can add any tag to any element. That being said, tags that are added to the database should be relevant to the element they are being associated with so that they actually serve their correct purpose.

Known Problems/Future Enhancements

- It may be more efficient for some columns to have check constraints or foreign key constraints, like cuisineType. However, I was also unsure if that would be too restrictive for some columns; I didn't know if it were better to have a constraint or let it be based on user entry.
- There is currently no easy way to relate and show common tags between different BuzzFeed elements. The commonTastyTags table does this in a way, however, there is a lot of repetition and it is not displayed in an easily understandable way.
- In the future, I would like to have a table for interviewed people. This table would be useful for the creativePosInNeed stored procedure. If the stored procedure returned "true", the function could then randomly select a person from interviewed people and have them fill the position in need.