Counter-strike

Global Offensive

Amateur League

By Chris Piccirillo

Table of Contents

Executive Summary

The goal of this database is to keep track of all games played in my (imaginary) amateur league for the competitive online FPS game Counter-strike: Global Offensive. Every active team is listed, including each player on each team. Information about the teams and players is available for fans or whoever wishes to view that data. Data such as players last name, first name, in-game name, gun used in game, age, email address, position, and sex are provided. Records of past games, a team's win/loss record and ranking in the league are also stored in the database. The ability to view which shout caster casted what game is also available for viewing. Potential users include fans, players, admins, and sponsors. If a new team joins the league or if a team drops out they can easily be added into the database or removed.

Counter-strike Global Offensive

Amateur League

E-R Diagram

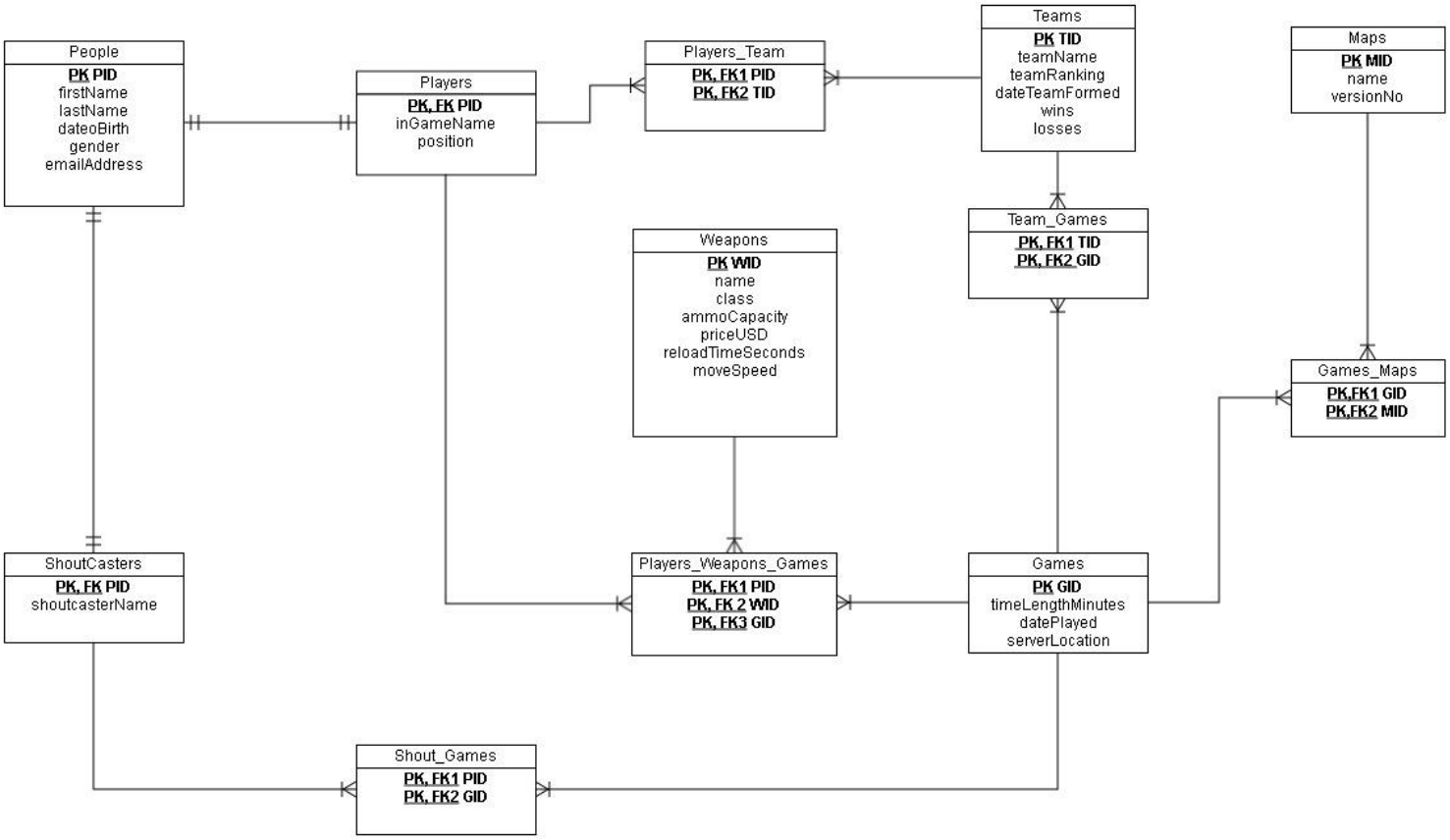

Table Create

Statements

This section will provide the create statements used to construct the tables of this database. The functional dependencies for each table will also be provided as well as a short description of what each unique field in each table is for.

People Table

The People Table has five unique fields, firstName, lastName, dateofBirth, gender, emailAddress. These attributes are used to keep track of the basic information tied to each person.

Create Statements (SQL)

```
CREATE TABLE People (
pid serial UNIQUE NOT NULL PRIMARY KEY,
firstName varchar NOT NULL,
lastName varchar NOT NULL,
dateofBirth date NOT NULL,
gender char(1) check(Gender='M' or Gender='F'),
emailAddress varchar NOT NULL,
UNIQUE(firstname,lastName,dateofbirth,Gender,Emailaddress)
) ;
```

Functional Dependencies

PID → firstName, lastName, dateofBirth, gender, emailaddress

Sample DATA(People)

	pid integer	firstname character varying	lastname character varying	dateofbirth date	gender character(1)	emailaddress character varying
1	1	Pablo	Gonzales	1990-01-30	M	Pablo32@msn.com
2	2	Michael	Smith	1980-10-21	M	Michael1@optonline.net
3	3	Sarah	Sandon	1992-12-12	F	SarahS1@gmail.com
4	4	Richard	Liao	1984-10-01	M	RLiao@yahoo.com
5	5	Bob	Burnz	1990-11-05	M	BB123@facebook.com
6	6	Yolano	Equerez	1979-05-28	M	Yolano43@gmail.com
7	7	Martin	Smith	1982-09-19	M	MysteryM1@yahoo.com
8	8	Maria	Jones	1990-10-11	F	Mj3x@yahoo.com
9	9	Brian	Kyren	1992-07-18	M	BKyren12@msn.com
10	10	Charles	Gonzales	1985-06-16	M	CG23@yahoo.com
11	11	Nikau	Rio	1990-04-02	M	NioRiox@yahoo.com
12	12	Paul	Patter	1994-10-10	M	PaulP343@optonline.net
13	13	Michelle	Jamison	1987-06-12	F	MMJX@gmail.com
14	14	Fred	Red	1989-05-09	M	Fredred@yahoo.com
15	15	Albert	Mannarino	1974-05-20	M	AB40@msn.com
16	16	John	Jacob	1978-09-22	M	JohnJay@gmail.com
17	17	Jack	Gonzales	1990-10-13	M	JackG123@yahoo.com
18	18	Pablo	Escobar	1982-04-30	M	Pabzso@yahoo.com
19	19	William	Wallace	1980-03-13	M	WillWW@optonline.net
20	20	Noah	Ark	1990-12-03	M	Noahsark23@msn.com
21	21	Oliver	Nanjo	1985-05-21	M	Nanjo1@yahoo.com
22	22	Mohammed	Alkor	1995-06-16	M	MoeAlk1@yahoo.com
23	23	Adam	Smith	1993-08-18	M	Pizzaandfries@msn.com
24	24	Edward	Zerner	1994-10-14	M	EZXX@yahoo.com
25	25	Victor	Vice	1989-01-17	M	VVICE@gmail.com
26	26	Arnold	Salar	1970-01-23	M	ASala@msn.com
27	27	Amelia	Smith	1980-06-17	F	Amelxx@yahoo.com
28	28	Miles	Smith	1990-10-11	M	MilesSmith123@gmail.com
29	29	Danielle	Smith	1993-11-04	F	DannySS@optonline.net
30	30	Samuel	Brooks	1981-02-24	M	SamBrooktheWall@faceboo

Players Table

The Players Table has two unique fields, inGameName and position. These attributes are used to keep track of in-game information for each player.

Create Statements (SQL)

```
CREATE TABLE Players (
inGameName varchar NOT NULL,
position varchar,
Primary Key(pid)
) INHERITS(People);
```

Functional Dependencies

PID → inGameName, position

Sample DATA(Players)

Data Output | Explain | Messages | History

	pid integer	firstname character varying	lastname character varying	dateofbirth date	gender character(1)	emailaddress character varying	ingamename character varying	position character varying
1	1	Pablo	Gonzales	1990-01-30	M	Pablo32@msn.com	AbPolo	AWPer
2	2	Michael	Smith	1980-10-21	M	Michael1@optonline.net	Mikey	Entry Fragger
3	3	Sarah	Sandon	1992-12-12	F	SarahS1@gmail.com	SSGX	AWPer
4	4	Richard	Liao	1984-10-01	M	RLiao@yahoo.com	APersuasion	Rifler
5	5	Bob	Burnz	1990-11-05	M	BB123@facebook.com	Burned	Top Fragger
6	6	Yolano	Equerez	1979-05-28	M	Yolano43@gmail.com	YOLO	Entry Fragger
7	7	Martin	Smith	1982-09-19	M	MysteryM1@yahoo.com	MartinTheMartia	Support
8	8	Maria	Jones	1990-10-11	F	Mj3x@yahoo.com	Maria	Top Fragger
9	9	Brian	Kyren	1992-07-18	M	BKyren12@msn.com	BKK	Top Fragger
10	10	Charles	Gonzales	1985-06-16	M	CG23@yahoo.com	RedHot	AWPer
11	11	Nikau	Rio	1990-04-02	M	NioRiox@yahoo.com	NioRio	Rifler
12	12	Paul	Patter	1994-10-10	M	PaulP343@optonline.net	PPPaul	Support
13	13	Michelle	Jamison	1987-06-12	F	MMJX@gmail.com	MichyJJ	Top Fragger
14	14	Fred	Red	1989-05-09	M	Fredred@yahoo.com	RedFred	Rifler
15	15	Albert	Mannarino	1974-05-20	M	AB40@msn.com	OldTimer	Entry Fragger
16	16	John	Jacob	1978-09-22	M	JohnJay@gmail.com	JJ	Support
17	17	Jack	Gonzales	1990-10-13	M	JackG123@yahoo.com	123 its Jack G	Entry Fragger
18	18	Pablo	Escobar	1982-04-30	M	Pabzso@yahoo.com	EscoDesco	AWPer
19	19	William	Wallace	1980-03-13	M	WillWW@optonline.net	WW	Support
20	20	Noah	Ark	1990-12-03	M	Noahsark23@msn.com	TheARK	Top Fragger
21	21	Oliver	Nanjo	1985-05-21	M	Nanjo1@yahoo.com	NanjotheBanjo	AWPer
22	22	Mohammed	Alkor	1995-06-16	M	MoeAlk1@yahoo.com	Moham	Rifler
23	23	Adam	Smith	1993-08-18	M	Pizzaandfries@msn.com	IlovePizza	Entry Fragger
24	24	Edward	Zerner	1994-10-14	M	EZXX@yahoo.com	E Z Kills	Rifler
25	25	Victor	Vice	1989-01-17	M	VVICE@gmail.com	Death Vice	Support

Shout Casters Table

The Shoutcasters Table has one unique field, shoutcasterName. This attribute is used to keep track of the name each shout caster goes by.

Create Statements (SQL)

```
CREATE TABLE Shoutcasters (
shoutcasterName varchar NOT NULL,
primary key(pid)
) INHERITS(People);
```

Functional Dependencies

PID → shoutcasterName

Sample DATA(Shout casters)

	pid integer	firstname character varying	lastname character varying	dateofbirth date	gender character(1)	emailaddress character varying	shoutcastername character varying
1	26	Arnold	Salar	1970-01-23	M	ASala@msn.com	Fire
2	27	Amelia	Smith	1980-06-17	F	Amelxx@yahoo.com	VMelia
3	28	Miles	Smith	1990-10-11	M	MilesSmith123@gmail.com	Ice
4	29	Danielle	Smith	1993-11-04	F	DannySS@optonline.net	Danny
5	30	Samuel	Brooks	1981-02-24	M	SamBrooktheWall@facebook.com	S Brook

Teams Table

The Teams Table has five unique fields, teamName, teamRanking, dateTeamFormed, wins, losses. These attributes are used to keep track of certain statistics for each team.

Create Statements (SQL)

```
CREATE TABLE Teams (
tid serial NOT NULL PRIMARY KEY,
teamName varchar NOT NULL,
teamRanking int NOT NULL check (teamRanking > 0),
dateTeamFormed date NOT NULL,
wins int NOT NULL,
losses int NOT NULL,
UNIQUE (teamName),
UNIQUE (teamRanking)
);
```

Functional Dependencies

TID→ teamName,teamRanking,dateTeamFormed,wins,losses

Sample DATA(TEAMS)

	tid integer	teamname character varying	teamranking integer	dateteamformed date	wins integer	losses integer
1	1	The Rockets	1	2013-01-01	12	0
2	2	The Zoo Keepers	3	2012-02-24	6	6
3	3	Flaming Monkeys	5	2011-05-10	1	11
4	4	One-Armed Banditz	2	2013-05-28	8	4
5	5	CSGO Pros	4	2012-05-17	3	9

MAPS Table

The Maps Table has two unique fields, name and versionNo. These attributes are used to keep track of certain information for each map.

Create Statements (SQL)

```
CREATE TABLE Maps (
mid serial UNIQUE NOT NULL PRIMARY KEY,
name varchar NOT NULL,
versionNo numeric NOT NULL
);
```

Functional Dependencies

MID → name,versionNo

Sample DATA(Maps)

	mid integer	name character varying	versionno numeric
1	1	Dust II	1.01
2	2	Gwailor	2.01
3	3	Cache	1.0
4	4	Ali	1.12
5	5	Chinatown	1.34
6	6	Agency	1.56
7	7	Seaside	1.10
8	8	Siege	3.01
9	9	Ruins	4.15
10	10	Mirage	1.0
11	11	Inferno	1.07
12	12	Train	1.89
13	13	Nuke	2.43
14	14	Dust	1.01
15	15	Aztec	1.01
16	16	Vertigo	3.65
17	17	Militia	2.0
18	18	Assault	2.01
19	19	Italy	1.23
20	20	Office	1.54

Weapons Table

The Weapons Table has six unique fields, name, class, ammoCapacity, priceUSD, reloadTimeSeconds, moveSpeed. These attributes are used to keep track of key information for each gun.

Create Statements (SQL)

```
CREATE TABLE Weapons (

wid serial UNIQUE NOT NULL PRIMARY KEY,

name varchar NOT NULL,

class varchar NOT NULL check (class ='Pistol' OR class = 'Submachine Gun' OR class =  'Heavy' OR class =  'Rifle'),

ammoCapacity int NOT NULL check (ammoCapacity >= 36 and ammoCapacity < 351),

priceUSD money NOT NULL,

reloadTimeSeconds numeric NOT NULL,

moveSpeed int NOT NULL,

UNIQUE (name));
```

Functional Dependencies

TID → name, class, ammoCapacity, priceUSD, reloadtimeSeconds, moveSpeed

Sample DATA(Weapons)

	wid integer	name character varying	class character varying	ammocapacity integer	priceusd money	reloadtimeseconds numeric	movespeed integer
1	1	Galil AR	Rifle	125	$2,000.	3.3	215
2	2	AK47	Rifle	120	$2,700.	2.5	215
3	3	SSG 08	Rifle	100	$2,000.	3.9	230
4	4	SG 553	Rifle	120	$3,000.	3	220
5	5	AWP	Rifle	40	$4,750.	2.7	200
6	6	G3SG1	Rifle	110	$5,000.	5	215
7	7	FAMAS	Rifle	115	$2,250.	3.8	220
8	8	M4A4	Rifle	120	$3,100.	3.4	225
9	9	M4A1-S	Rifle	60	$2,900.	3.4	225
10	10	AUG	Rifle	120	$3,300.	3.8	221
11	11	SCAR-20	Rifle	110	$5,000.	5	215
12	12	Glock-18	Pistol	140	$200.00	2.2	240
13	13	P250	Pistol	65	$300.00	2.5	240
14	14	Desert Eagle	Pistol	42	$800.00	2.2	230
15	15	Dual Berettas	Pistol	150	$700.00	5	240
16	16	Tec-9	Pistol	155	$500.00	2.7	240
17	17	P2000	Pistol	65	$200.00	2.5	240
18	18	USP-S	Pistol	36	$200.00	2.5	240
19	19	Five-SeveN	Pistol	120	$500.00	2.5	240
20	20	Nova	Heavy	40	$1,200.	0	220
21	21	XM1014	Heavy	39	$2,200.	4.2	240
22	22	Sawed-Off	Heavy	39	$1,200.	0	210
23	23	Mag-7	Heavy	37	$1,800.	3	225
24	24	Mac-10	Submachine Gun	130	$1,050.	3.45	240
25	25	MP7	Submachine Gun	150	$1,700.	3.45	220
26	26	MP9	Submachine Gun	150	$1,250.	2.3	240
27	27	UMP-45	Submachine Gun	125	$1,200.	3.75	230
28	28	PP-Bizon	Submachine Gun	184	$1,400.	2.5	240
29	29	P90	Submachine Gun	150	$2,350.	3.5	230

Games Table

The Games Table has three unique fields, timeLengthMinutes, datePlayed, and serverLocation. These attributes are used to keep track of certain information for each game.

Create Statements (SQL)

```
CREATE TABLE Games (
gid serialNOT NULL PRIMARY KEY,
timeLengthMinutes int NOT NULL,
datePlayed date NOT NULL,
serverLocation varchar NOT NULL
);
```

Functional Dependencies

GID → timeLengthMinutes, datePlayed, serverLocation

Sample DATA(Games)

	gid integer	timelengthminutes integer	dateplayed date	serverlocation character varying
1	1	100	2013-03-09	California
2	2	30	2013-12-29	New York
3	3	110	2013-01-07	Georgia
4	4	117	2013-07-03	California
5	5	123	2013-04-07	California
6	6	56	2013-12-24	New York
7	7	90	2013-11-06	Texas
8	8	92	2013-12-23	Texas
9	9	97	2013-06-02	California
10	10	25	2013-05-01	Georgia
11	11	56	2013-04-12	Georgia
12	12	78	2013-03-20	Georgia
13	13	80	2013-02-19	California
14	14	92	2013-12-17	California
15	15	100	2013-04-10	New York
16	16	101	2013-09-11	California
17	17	24	2013-07-10	New York
18	18	51	2013-06-09	Texas
19	19	57	2013-04-08	California
20	20	73	2013-05-23	Texas
21	21	51	2013-11-05	New York
22	22	61	2013-11-05	New York
23	23	62	2013-12-24	New York
24	24	85	2013-02-12	Texas
25	25	102	2013-03-22	Georgia
26	26	62	2013-06-26	California
27	27	43	2013-04-29	California
28	28	56	2013-01-30	Georgia
29	29	71	2013-05-11	California
30	30	86	2013-01-22	California

Shout_games table

This table has no unique fields to it. This table is used to eliminate the many to many relationship between shout casters and games. Therefore there are no functional dependencies.

Create Statements (SQL)

```
CREATE TABLE Shout_Games(

pid serial NOT NULL,

gid serial NOT NULL,

primary key(pid,gid),

foreign key(pid) references Shoutcasters(pid),

foreign key(gid) references Games(gid)


);
```

Functional Dependencies

None

Sample DATA(SHOUT_GAMES)

	pid integer	gid integer
1	26	1
2	28	2
3	29	3
4	26	4
5	26	5
6	28	6
7	26	7
8	26	8
9	30	9
10	26	10
11	28	11
12	29	12
13	27	13
14	26	14
15	30	15
16	26	16
17	30	17
18	26	18
19	29	19
20	28	20
21	26	21
22	30	22
23	26	23
24	28	24
25	27	25
26	29	26
27	29	27
28	30	28
29	26	29
30	30	30

Players_Team Table

This table has no unique fields to it. This table is used to eliminate the many to many relationship between players and teams. Therefore there are no functional dependencies.

Create Statements (SQL)

```
CREATE TABLE Players_Team(
pid serial NOT NULL,
tid serial NOT NULL,
primary key(pid,tid),
foreign key(pid) references Players(pid),
foreign key(tid) references Teams(tid)
);
```

Functional Dependencies

None

Sample DATA(Players_team)

	pid integer	tid integer
1	1	1
2	6	1
3	7	1
4	11	1
5	5	1
6	3	2
7	15	2
8	12	2
9	14	2
10	13	2
11	21	3
12	17	3
13	22	3
14	25	3
15	9	3
16	10	4
17	2	4
18	16	4
19	24	4
20	8	4
21	18	5
22	23	5
23	19	5
24	4	5
25	20	5

Teams_Games Table

This table has no unique fields to it. This table is used to eliminate the many to many relationship between teams and games played. Therefore there are no functional dependencies.

Create Statements (SQL)

```
CREATE TABLE Teams_Games(
tid serial NOT NULL,
gid serial NOT NULL,
primary key(tid,gid),
foreign key(gid) references Games(gid),
foreign key(tid) references Teams(tid)
);
```

Functional Dependencies

None

Sample DATA(Teams_Games)

	tid integer	gid integer
1	1	1
2	2	1
3	1	2
4	3	2
5	1	3
6	2	3
7	3	4
8	4	4
9	1	5
10	4	5
11	2	6
12	4	6
13	2	7
14	4	7
15	4	8
16	3	8
17	2	9
18	5	9
19	1	10
20	5	10
21	1	11
22	5	11
23	3	12
24	2	12
25	2	13
26	5	13
27	3	14
28	5	14
29	1	15
30	5	15
31	3	16
32	5	16
33	1	17
34	5	17
35	2	18
36	5	18
37	5	19
38	3	19
39	1	20
40	5	20
41	2	21
42	3	21
43	1	22
44	2	22
45	4	23

Games_Maps table

This table has no unique fields to it. This table is used to eliminate the many to many relationship between games and the maps in which they are played on. Therefore there are no functional dependencies.

Create Statements (SQL)

```
CREATE TABLE Games_Maps(
gid serial NOT NULL,
mid serial NOT NULL,
primary key(gid,mid),
foreign key(gid) references Games(gid),
foreign key(mid) references Maps(mid)
);
```

Functional Dependencies

None

Sample DATA(Teams_Games)

	gid integer	mid integer
1	1	16
2	2	18
3	3	1
4	4	3
5	5	2
6	6	5
7	7	4
8	8	12
9	9	6
10	10	1
11	11	13
12	12	13
13	13	15
14	14	20
15	15	17
16	16	19
17	17	11
18	18	10
19	19	5
20	20	12
21	21	9
22	22	2
23	23	7
24	24	4
25	25	1
26	26	15
27	27	1
28	28	17
29	29	8
30	30	14

Players_weapons_games Table

This table has no unique fields to it. This table is used to eliminate the many to many relationships between players/weapons, games/weapons, and players/games. Therefore there are no functional dependencies.

This table displays which weapon was used by which players in which game.

There is a view below to actually see the players name and the gun name

Create Statements (SQL)

```
CREATE TABLE Games_Maps(
gid serial NOT NULL,
mid serial NOT NULL,
primary key(gid,mid),
foreign key(gid) references Games(gid),
foreign key(mid) references Maps(mid)
);
```

Functional Dependencies

None

Sample DATA(Teams_Games)

	pid integer	wid integer	gid integer
1	1	8	1
2	6	22	1
3	7	5	1
4	11	2	1
5	5	1	1
6	3	8	1
7	15	1	1
8	12	1	1
9	14	1	1
10	13	1	1
11	1	3	2
12	6	22	2
13	7	17	2
14	11	18	2
15	5	27	2
16	9	8	2
17	25	11	2
18	22	1	2
19	17	10	2
20	21	2	2
21	1	18	3
22	6	20	3
23	7	15	3
24	11	12	3
25	5	10	3
26	3	8	3
27	15	12	3
28	12	19	3
29	14	3	3
30	13	24	3
31	21	17	4
32	17	22	4
33	22	5	4
34	25	2	4
35	9	11	4
36	10	16	4
37	2	16	4
38	16	2	4
39	24	11	4
40	8	3	4
41	1	17	5
42	6	22	5
43	7	5	5
44	11	21	5
45	5	14	5

Views

Create Statements (SQL)

View to see what gun each player used in each game

```
Create View  PlayersGunsInGames AS

Select p.firstname, p.lastname, w.name, g.gid

from players p, weapons w, games g,players_weapons_games pwg

Where p.pid = pwg.pid and g.gid = pwg.gid and w.wid = pwg.wid
```

Sample

(Next Page)

	firstname character varying	lastname character varying	name character varying	gid integer
1	Pablo	Gonzales	M4A4	1
2	Yolano	Equerez	Sawed-Off	1
3	Martin	Smith	AWP	1
4	Nikau	Rio	AK47	1
5	Bob	Burnz	Galil AR	1
6	Sarah	Sandon	M4A4	1
7	Albert	Mannarino	Galil AR	1
8	Paul	Patter	Galil AR	1
9	Fred	Red	Galil AR	1
10	Michelle	Jamison	Galil AR	1
11	Pablo	Gonzales	SSG 08	2
12	Yolano	Equerez	Sawed-Off	2
13	Martin	Smith	P2000	2
14	Nikau	Rio	USP-S	2
15	Bob	Burnz	UMP-45	2
16	Brian	Kyren	M4A4	2
17	Victor	Vice	SCAR-20	2
18	Mohammed	Alkor	Galil AR	2
19	Jack	Gonzales	AUG	2
20	Oliver	Nanjo	AK47	2
21	Pablo	Gonzales	USP-S	3
22	Yolano	Equerez	Nova	3
23	Martin	Smith	Dual Berettas	3
24	Nikau	Rio	Glock-18	3
25	Bob	Burnz	AUG	3
26	Sarah	Sandon	M4A4	3
27	Albert	Mannarino	Glock-18	3
28	Paul	Patter	Five-SeveN	3
29	Fred	Red	SSG 08	3
30	Michelle	Jamison	Mac-10	3
31	Oliver	Nanjo	P2000	4
32	Jack	Gonzales	Sawed-Off	4
33	Mohammed	Alkor	AWP	4
34	Victor	Vice	AK47	4
35	Brian	Kyren	SCAR-20	4
36	Charles	Gonzales	Tec-9	4
37	Michael	Smith	Tec-9	4

View to see what team each player is on

Create View PlayersonTeams AS

Select p.firstname, p.lastname, t.teamname

From players p, teams t, players_team pt

Where p.pid = pt.pid and t.tid = pt.tid

Sample

	firstname character varying	lastname character varying	teamname character varying
1	Pablo	Gonzales	The Rockets
2	Yolano	Equerez	The Rockets
3	Martin	Smith	The Rockets
4	Nikau	Rio	The Rockets
5	Bob	Burnz	The Rockets
6	Sarah	Sandon	The Zoo Keepers
7	Albert	Mannarino	The Zoo Keepers
8	Paul	Patter	The Zoo Keepers
9	Fred	Red	The Zoo Keepers
10	Michelle	Jamison	The Zoo Keepers
11	Oliver	Nanjo	Flaming Monkeys
12	Jack	Gonzales	Flaming Monkeys
13	Mohammed	Alkor	Flaming Monkeys
14	Victor	Vice	Flaming Monkeys
15	Brian	Kyren	Flaming Monkeys
16	Charles	Gonzales	One-Armed Bandi
17	Michael	Smith	One-Armed Bandi
18	John	Jacob	One-Armed Bandi
19	Edward	Zerner	One-Armed Bandi
20	Maria	Jones	One-Armed Bandi
21	Pablo	Escobar	CSGO Pros
22	Adam	Smith	CSGO Pros
23	William	Wallace	CSGO Pros
24	Richard	Liao	CSGO Pros
25	Noah	Ark	CSGO Pros

This view shows what games each shout caster casted.

Select s.firstname, s.lastname, s.shoutcastername, g.gid

From shoutcasters s, games g, shout_games sg

Where s.pid = sg.pid and g.gid = sg.gid

Sample

(Next Page)

	firstname character varying	lastname character varying	shoutcastername character varying	gid integer
1	Arnold	Salar	Fire	1
2	Miles	Smith	Ice	2
3	Danielle	Smith	Danny	3
4	Arnold	Salar	Fire	4
5	Arnold	Salar	Fire	5
6	Miles	Smith	Ice	6
7	Arnold	Salar	Fire	7
8	Arnold	Salar	Fire	8
9	Samuel	Brooks	S Brook	9
10	Arnold	Salar	Fire	10
11	Miles	Smith	Ice	11
12	Danielle	Smith	Danny	12
13	Amelia	Smith	VMelia	13
14	Arnold	Salar	Fire	14
15	Samuel	Brooks	S Brook	15
16	Arnold	Salar	Fire	16
17	Samuel	Brooks	S Brook	17
18	Arnold	Salar	Fire	18
19	Danielle	Smith	Danny	19
20	Miles	Smith	Ice	20
21	Arnold	Salar	Fire	21
22	Samuel	Brooks	S Brook	22
23	Arnold	Salar	Fire	23
24	Miles	Smith	Ice	24
25	Amelia	Smith	VMelia	25
26	Danielle	Smith	Danny	26
27	Danielle	Smith	Danny	27
28	Samuel	Brooks	S Brook	28
29	Arnold	Salar	Fire	29
30	Samuel	Brooks	S Brook	30

This view shows what map each game was played on.

Select g.gid,g.timelengthminutes, g.dateplayed,g.serverlocation, m.name, m.versionno

From maps m, games g, games_maps gm

Where m.mid = gm.mid and g.gid = gm.gid

Sample

(Next Page)

	gid integer	timelengthminutes integer	dateplayed date	serverlocation character varying	name character varying	versionno numeric
1	1	100	2013-03-0	California	Vertigo	3.65
2	2	30	2013-12-2	New York	Assault	2.01
3	3	110	2013-01-0	Georgia	Dust II	1.01
4	4	117	2013-07-0	California	Cache	1.0
5	5	123	2013-04-0	California	Gwailor	2.01
6	6	56	2013-12-2	New York	Chinatown	1.34
7	7	90	2013-11-0	Texas	Ali	1.12
8	8	92	2013-12-2	Texas	Train	1.89
9	9	97	2013-06-0	California	Agency	1.56
10	10	25	2013-05-0	Georgia	Dust II	1.01
11	11	56	2013-04-1	Georgia	Nuke	2.43
12	12	78	2013-03-2	Georgia	Nuke	2.43
13	13	80	2013-02-1	California	Aztec	1.01
14	14	92	2013-12-1	California	Office	1.54
15	15	100	2013-04-1	New York	Militia	2.0
16	16	101	2013-09-1	California	Italy	1.23
17	17	24	2013-07-1	New York	Inferno	1.07
18	18	51	2013-06-0	Texas	Mirage	1.0
19	19	57	2013-04-0	California	Chinatown	1.34
20	20	73	2013-05-2	Texas	Train	1.89
21	21	51	2013-11-0	New York	Ruins	4.15
22	22	61	2013-11-0	New York	Gwailor	2.01
23	23	62	2013-12-2	New York	Seaside	1.10
24	24	85	2013-02-1	Texas	Ali	1.12
25	25	102	2013-03-2	Georgia	Dust II	1.01
26	26	62	2013-06-2	California	Aztec	1.01
27	27	43	2013-04-2	California	Dust II	1.01
28	28	56	2013-01-3	Georgia	Militia	2.0
29	29	71	2013-05-1	California	Siege	3.01
30	30	86	2013-01-2	California	Dust	1.01

REPORTS AND THEIR QUERIES

Report to Show the Most Popular Guns in the past games played.

```
SELECT w.name, COUNT(*)

FROM players_weapons_Games pwg, weapons w

Where pwg.wid = w.wid

GROUP BY w.name, pwg.wid

Order by count desc
```

Sample

	name character varying	count bigint
1	AK47	23
2	Galil AR	21
3	AUG	19
4	Tec-9	15
5	Desert Eagle	15
6	Glock-18	14
7	Sawed-Off	14
8	SCAR-20	13
9	Five-SeveN	12
10	P250	12
11	P2000	12
12	SSG 08	11
13	M4A4	11
14	FAMAS	10
15	M4A1-S	9
16	USP-S	9
17	AWP	9
18	Nova	8
19	Mac-10	8
20	SG 553	7
21	MP7	7
22	G3SG1	6
23	XM1014	6
24	P90	6
25	Mag-7	6
26	Dual Berettas	6
27	PP-Bizon	5
28	UMP-45	5
29	MP9	1

Report to Show the Most Popular Maps chosen in the 30 games played so far.

```
SELECT m.name, COUNT(*)
FROM games_maps gm, maps m
Where gm.mid = m.mid
GROUP BY m.name, gm.mid
Order by count desc
```

Sample

	name character varying	count bigint
1	Dust II	4
2	Ali	2
3	Gwailor	2
4	Nuke	2
5	Train	2
6	Aztec	2
7	Militia	2
8	Chinatown	2
9	Assault	1
10	Dust	1
11	Office	1
12	Inferno	1
13	Siege	1
14	Cache	1
15	Vertigo	1
16	Italy	1
17	Seaside	1
18	Mirage	1
19	Agency	1
20	Ruins	1

Stored

Procedures

This stored procedure checks if any player doesn't have a preferred or designated position on their team and replaces the null value with 'No Position Specified'.

```
CREATE OR REPLACE FUNCTION playerWithNoPosition()
returns trigger as $$
BEGIN
IF (new.position is null)
FROM PLAYERS
WHERE players.position is null
THEN
UPDATE players SET position = 'No Position Specified'
WHERE position IS NULL;
END if;
Return new;
END
$$LANGUAGE plpgsql;
```

Triggers

This trigger is used to check if there was an insert or update on the players table and runs the "playerWithNoPosition" procedure.

Create Trigger noPosition

After Insert or Update

On Players

FOR EACH ROW Execute Procedure playerWithNoPosition();

Security

The security in place for this database would create three levels of users. An admin, player, and a fan. An admin would have access to anything, players can insert a new player into the system(themselves), and fans can only view the information in the database.

CREATE USER Counterstrikegoadmin WITH PASSWORD 'aDmIncsgo143';

ADMIN

Revoke all on People from Counterstrikegoadmin;
Revoke all on Players from Counterstrikegoadmin;
Revoke all on Weapons from Counterstrikegoadmin;
Revoke all on Games from Counterstrikegoadmin;
Revoke all on Shoutcasters from Counterstrikegoadmin;
Revoke all on Maps from Counterstrikegoadmin;
Revoke all on Shout_games from Counterstrikegoadmin;
Revoke all on Teams from Counterstrikegoadmin;
Revoke all on Players_Weapons_games from Counterstrikegoadmin;
Revoke all on Teams_games from Counterstrikegoadmin;
Revoke all on Players_Team from Counterstrikegoadmin;
Revoke all on Games_Maps from Counterstrikegoadmin;

Grant insert, update, delete, select on People to Counterstrikegoadmin;
Grant insert, update, delete, select on Players to Counterstrikegoadmin;
Grant insert, update, delete, select on Weapons to Counterstrikegoadmin;
Grant insert, update, delete, select on Games to Counterstrikegoadmin;
Grant insert, update, delete, select on Shoutcasters to Counterstrikegoadmin;
Grant insert, update, delete, select on Maps to Counterstrikegoadmin;
Grant insert, update, delete, select on Shout_games to Counterstrikegoadmin;
Grant insert, update, delete, select on Teams to Counterstrikegoadmin;
Grant insert, update, delete, select on Players_Weapons_games to Counterstrikegoadmin;
Grant insert, update, delete, select on Teams_games to Counterstrikegoadmin;
Grant insert, update, delete, select on Games_maps to Counterstrikegoadmin;
Grant insert, update, delete, select on Players_Team to Counterstrikegoadmin;

CREATE USER CounterstrikeGOplayer; WITH PASSWORD 'playercsgo174';

PLAYER

Revoke all on People from CounterstrikeGOplayer;
Revoke all on Players from CounterstrikeGOplayer;
Revoke all on Weapons from CounterstrikeGOplayer;
Revoke all on Games from CounterstrikeGOplayer;
Revoke all on Shoutcastersfrom CounterstrikeGOplayer;
Revoke all on Maps from CounterstrikeGOplayer;
Revoke all on Shout_games from CounterstrikeGOplayer;
Revoke all on Teams from CounterstrikeGOplayer;
Revoke all on Players_Weapons_games from CounterstrikeGOplayer;
Revoke all on Teams_gamesfrom CounterstrikeGOplayer;
Revoke all on Games_Maps from CounterstrikeGOplayer;
Revoke all on Players_Team from CounterstrikeGoplayer;

Grant select on People from CounterstrikeGOplayer;
Grant select, **insert** on Players from CounterstrikeGOplayer;
Grant select on Weapons from CounterstrikeGOplayer;
Grant select on Games from CounterstrikeGOplayer;
Grant select on Shoutcastersfrom CounterstrikeGOplayer;
Grant select on Maps from CounterstrikeGOplayer;
Grant select on Shout_games from CounterstrikeGOplayer;
Grant select on Teams from CounterstrikeGOplayer;
Grant select on Players_Weapons_games from CounterstrikeGOplayer;
Grant select on Teams_gamesfrom CounterstrikeGOplayer;
Grant select on Games_Maps from CounterstrikeGOplayer;
Grant select on Players_Team from CounterstrikeGoplayer;

CREATE USER CounterstrikeGOfan; WITH PASDSWORD 'fan123';

FAN

Revoke all on People from CounterstrikeGOfan;
Revoke all on Players from CounterstrikeGOfan;
Revoke all on Weapons from CounterstrikeGOfan
Revoke all on Games from CounterstrikeGOfan;
Revoke all on Shoutcastersfrom CounterstrikeGOfan;
Revoke all on Maps from CounterstrikeGOfan;
Revoke all on Shout_games from CounterstrikeGOfan;
Revoke all on Teams from CounterstrikeGOfan;
Revoke all on Players_Weapons_games from CounterstrikeGOfan;
Revoke all on Teams_gamesfrom CounterstrikeGOfan;
Revoke all on Games_Maps from CounterstrikeGOfan;
Revoke all on Players_Team from CounterstrikeGofan;

Grant select on People from CounterstrikeGOfan;
Grant select on Players from CounterstrikeGOfan;
Grant select on Weapons from CounterstrikeGOfan;
Grant select on Games from CounterstrikeGOfan;
Grant select on Shoutcastersfrom CounterstrikeGOfan;
Grant select on Maps from CounterstrikeGOfan;
Grant select on Shout_games from CounterstrikeGOfan;
Grant select on Teams from CounterstrikeGOfan;
Grant select on Players_Weapons_games from CounterstrikeGOfan;
Grant select on Teams_gamesfrom CounterstrikeGOfan;
Grant select on Games_Maps from CounterstrikeGOfan;
Grant select on Players_Team from CounterstrikeGofan;

IMPLEMENTATION NOTES

- Implementation for this database should be fairly simple
- Run the create statements and insert the proper data
- Make sure each user is created properly with the right permissions
- Allow fans access to views for quick and easy information
- Allow players one insert per account (For Themselves)
- Implement a banning system to ban malicious users

Known Problems

- Wins and losses are not automatically update as of yet. Need a trigger/procedure to update wins/losses after a new game is played
- Two entries are required to display both teams for one game. A method to just require one entry should be implemented
- Players labeled classes ex. “AWPer”,”Rifler” don’t necessarily correspond to their guns used in games as they can use whatever they want each game and aren’t tied down to their one classes gun.

FUTURE ENHANCEMENTS

- Track players kills and deaths and have a kill/death ratio available for viewing
- Account for Terrorist and Counter Terrorist stats (Both sides are played each game)
- Account for players picking up other players weapons mid game.