NXL

National Xball League

Designed by Chris Cordaro

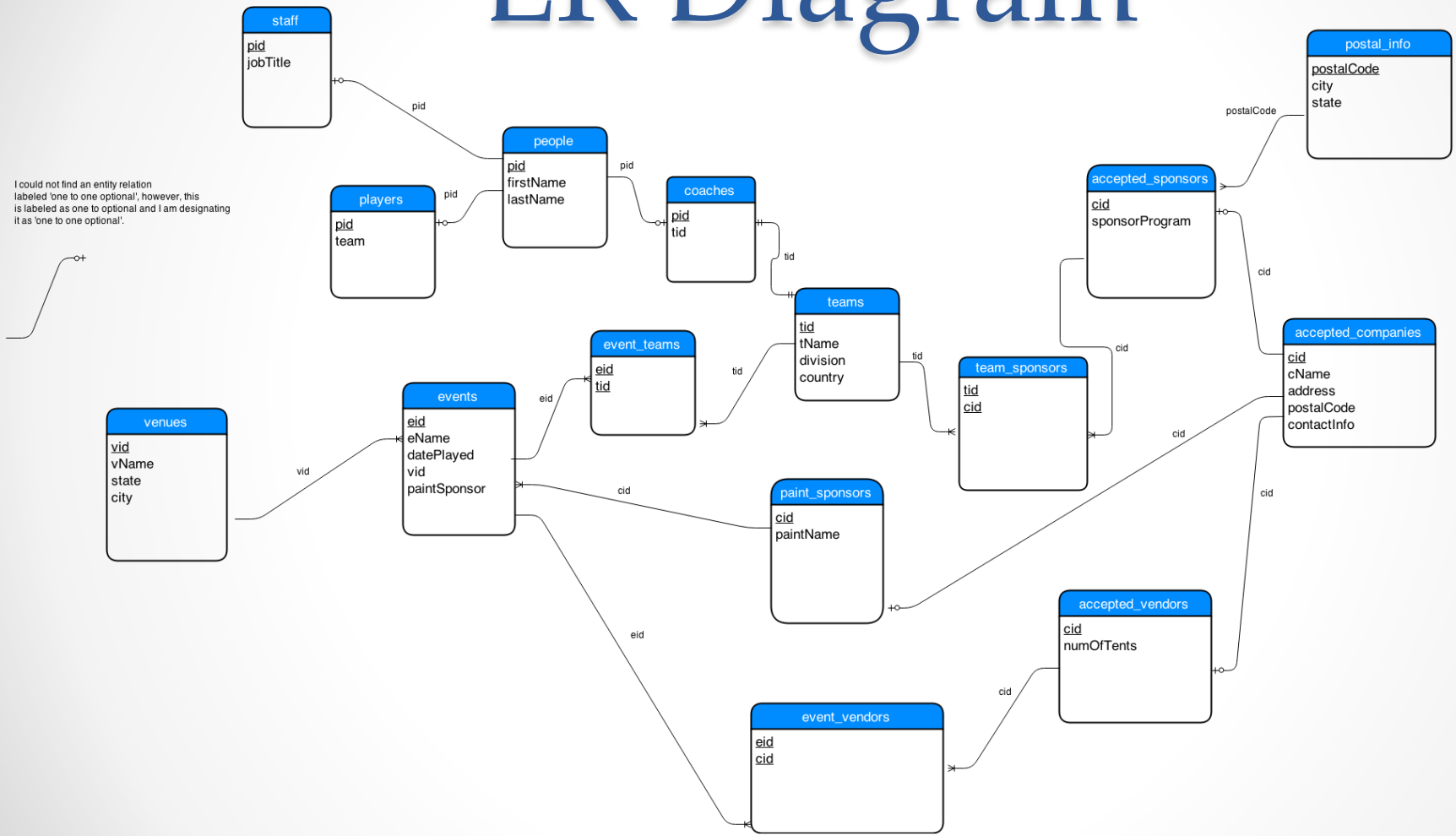
Table of Contents

Executive Summary.....3
ER Diagram.....4
Tables.....5
Views.....21
Stored Procedures.....24
Queries.....26
Securities.....29
Known Issues.....30
Further Enhancements.....31

Executive Summary

- This document demonstrates my implementation of a database designed for the National Xball League, an organization which runs the professional paintball scene in America. The database is a tool that could be used by League organizers and executives to portray and organize the intricacies of a professional sport in a(relatively) simply manner. The desired goal is to create a database which could be easily implemented into the NXL's business model for more effective data-analysis.

ER Diagram


staff
pid
jobTitle
pid
people
pid
firstName
lastName
pid
players
pid
team
pid
coaches
pid
tid
tid
teams
tid
tName
division
country
tid
team_sponsors
tid
cid
cid
accepted_sponsors
cid
sponsorProgram
postalCode
postal_info
postalCode
city
state
cid
accepted_companies
cid
cName
address
postalCode
contactInfo
cid
I could not find an entity relation labeled 'one to one optional', however, this is labeled as one to optional and I am designating it as 'one to one optional'.
event_teams
eid
tid
tid
eid
events
eid
eName
datePlayed
vid
paintSponsor
vid
venues
vid
vName
state
city
cid
paint_sponsors
cid
paintName
cid
eid
event_vendors
eid
cid
cid
accepted_vendors
cid
numOfTents
cid

Tables

PEOPLE

```
CREATE TABLE people (
   pid         CHAR(5) NOT NULL,
   firstName   TEXT NOT NULL,
   lastName    TEXT NOT NULL,
   PRIMARY KEY(pid)

);
```

	pid character(5)	firstname text	lastname text
1	p1001	Dave	Bains
2	p1002	Keith	Brown
3	p1003	Chad	Busiere
4	p1004	Nick	Leival
5	p1005	Josh	Ouimet
6	p1006	Justin	Rabackoff
7	p1007	Ranev	Stanczak
8	p1008	Zane	Yachimec
9	p1009	Zacharv	Yachimec
10	p1010	Bart	Yachimec
11	p2001	AJ	Lawhead

Lists all people involved with the NXL events along with their standard attributes.

Functional Dependencies:
pid->firstName, lastName

TEAMS

```
CREATE TABLE teams(
    tid         CHAR(4) NOT NULL,
    tName       TEXT NOT NULL,
    division    TEXT NOT NULL,
    country     TEXT NOT NULL,

    PRIMARY KEY (tid)

);
```

	tid character(4)	tname text	division text	country text
1	1001	Edmonton Impact	Champions	Canada
2	2002	Chicago Aftershock	Champions	USA
3	3003	San Diego Dynasty	Champions	USA

Lists registered teams along with their basic attributes.
Additionally, there is nothing stopping a team from sharing the same name as another registered team.
Because of this 'tid' must be created as a primary key.

Functional Dependencies:
tid->tName, division, country

COACHES

```
CREATE TABLE coaches (
   pid         CHAR(5) NOT NULL REFERENCES people(pid),
   tid CHAR(4) NOT NULL REFERENCES teams(tid),
   PRIMARY KEY(pid)

);
```

	pid character(5)	tid character(4)
1	p1010	1001
2	p2009	2002
3	p3008	3003

Functional Dependencies:
pid->tid

Lists 'potential' coaches along with the team they are coaching. It should be noted that there is a restriction that states that a coach must be present on all teams. Additionally, NXL rules require that one coach can only ever coach one team at a time. This creates a one-to-one relationship between coaches and teams; and although one may argue that the table coaches is now unnecessary, it exists to create a more logical schema.

PLAYERS

```
CREATE TABLE players(
   pid          CHAR(5) NOT NULL REFERENCES people(pid),
   team         CHAR(4) NOT NULL REFERENCES teams(tid),

   PRIMARY KEY (pid)

);
```

	pid character(5)	team character(4)
1	p1001	1001
2	p1002	1001
3	p1003	1001
4	p1004	1001
5	p1005	1001
6	p1006	1001
7	p1007	1001
8	p1008	1001
9	p1009	1001
10	p2001	2002
11	p2002	2002

Functional Dependencies:
pid->team

Lists all players and their current team. A player cannot be registered without a team and he/she cannot be on more than one team.

STAFF

```
CREATE TABLE staff(
  pid         CHAR(5) NOT NULL REFERENCES people(pid),
  jobTitle    TEXT NOT NULL,
  PRIMARY KEY (pid)

);
```

	pid character(5)	jobtitle text
1	p5006	Commentator
2	p5007	Commentator
3	p5008	On feild reporter

Lists all staff members that are present at each NXL event. Excludes referees. Additionally, it is implied that each staff member can only can have one job title.

Functional Dependencies:
pid->jobtitle

POSTAL INFO

```
CREATE TABLE postalInfo(
   postalCode CHAR(5) NOT NULL,
   city       TEXT NOT NULL,
   state      TEXT NOT NULL,
   PRIMARY KEY(postalCode)

);
```

	postalcode character(5)	city text	state text
1	90210	San Diego	California
2	10505	Springfield	Massachussets
3	12345	Springfield	Illinois
4	22134	New York	New York
5	10589	Tampa	Florida
6	82654	Poughkeepsie	New York
7	80808	Newport	Rhode Island

This table exists in an attempt to solve the problem of postal codes violating 2NF. Many companies can share the same postal code.

ACCEPTED COMPANIES

```
CREATE TABLE accepted_companies(
   cid           CHAR(3) NOT NULL,
   cName         TEXT NOT NULL,
   address       TEXT NOT NULL,
   postalCode    CHAR(5) NOT NULL,
   contactInfo   TEXT NOT NULL,
   PRIMARY KEY (cid)

);
```

Lists all companies that the NXL recognizes and allows to participate as a sponsor/vendor. A company can exist as both a sponsor or a vendor. Additionally, a company could be listed as an accepted company but not be present at any events, either as a sponsor or vendor.

	cid character(3)	cname text	address text	postalcode character(5)	contactinfo text
1	111	Planet Eclipse	PE Road	90210	PlanetEclipsePaintball@PE.com
2	112	DYE Percision	DYE Sports Way	10505	DYEPaintball@dyeproducts.com
3	113	KEE Action Sports	KEY Paintball Drive	12345	KEE-ACTION@keeaction.com
4	114	MacDev Paintball	BackRoad Drive	22134	MacDevSupport@MacDev.com
5	115	Virtue Paintball	BackRoad Drive	22134	MacDevSupport@PE.com
6	116	GI Sports	GI Sports Lane	10509	GIproducts@gisports.com
7	117	HK Army	Hustle-Kids Lane	82654	Info@HKarmy.com
8	118	BNKER Kings	118 Central Drive	10505	BNKERInfo@BNKER.com
9	119	Bob Long	118 Central Drive	10505	BobSupport@boblong.com
10	120	Invert Paintball	Invert Drive	80808	KEE-ACTION@kccaction.com

Functional Dependencies:
cid->cName, address, postalCode, contactInfo

ACCEPTED SPONSORS

```
CREATE TABLE accepted_sponsors(
    cid                 CHAR(3) NOT NULL REFERENCES accepted_companies(cid),
    sponsorProgram      TEXT NOT NULL,
    PRIMARY KEY (cid)
);
```

	cid character(3)	sponsorprogram text
1	113	Full
2	111	Full
3	112	Full
4	114	Full
5	115	Partial
6	116	Partial
7	117	Partial

Lists all companies that are accepted as sponsors, as well as their degree of sponsorship program. A company that offers full sponsor program implicitly offers a partial program. All accepted sponsors are accepted companies, but not all accepted companies are sponsors.

Functional Dependencies:
cid->sponsorProgram

TEAM SPONSORS

```
CREATE TABLE team_sponsors(
   tid         CHAR(4) NOT NULL REFERENCES teams(tid),
   cid         CHAR(3) NOT NULL REFERENCES accepted_sponsors(cid),
   PRIMARY KEY (tid, cid)
);
```

	tid character(4)	cid character(3)
1	1001	113
2	1001	111
3	1001	117
4	2002	112
5	2002	114
6	2002	116

Lists the sponsors that each team has. Many teams can have many sponsors but since Alan would rather die than have a many to many relationship, ‘team_sponsors’ serves as the medium between ‘teams’ and ‘accepted_sponsors’.

PAINT SPONSORS

```
CREATE TABLE paint_sponsors(
   cid        CHAR(3) NOT NULL REFERENCES accepted_sponsors,
   paintName TEXT NOT NULL,
   PRIMARY KEY (cid)

);
```

	cid character(3)	paintname text
1	113	RPS PREMIUM
2	111	PE GOLD
3	112	DYE CG
4	114	Macdev Prime
5	116	GI 5 Star
6	117	Hk Supreme

Each event has one paint sponsor, and one paint sponsor can be at multiple events. A paint sponsor has to already be an accepted sponsor, but an accepted sponsor does not have to be a paint sponsor, hence the one-to-one optional relationship.

Functional Dependencies:
cid->paintName

VENUES

```
CREATE TABLE venues(
   vid        CHAR(5) NOT NULL,
   vName      TEXT NOT NULL,
   state      TEXT NOT NULL,
   city       TEXT NOT NULL,
   PRIMARY KEY (vid)

);
```

	vid character(5)	vname text	state text	city text
1	11111	Cousins Paintball	Texas	Forney
2	22222	OXCC Paintball	Maryland	Chesapeake
3	33333	CPX Sports	Illinois	Joliet
4	44444	Cousins Paintball	California	Los Angeles
5	55555	Fantasy of Flight	Florida	Polk City

Functional Dependencies:
vid->vName, state, city

Lists the available venues along with their appropriate attributes. One venue can host many events throughout a season(s). With this snapshot it seems that city determines the state, but obviously city names can repeat between states, and as new venues get added the likely-hood of repeating city names increases.

EVENTS

```
CREATE TABLE events(
   eid           CHAR(5) NOT NULL,
   eName         TEXT NOT NULL,
   datePlayed    DATE NOT NULL,
   vid           CHAR(5) NOT NULL REFERENCES venues(vid) ,
   paintSponsor CHAR(5) NOT NULL REFERENCES paint_sponsors(cid),
   PRIMARY KEY (eid)
);
```

	eid character(5)	ename text	dateplayed date	vid character(5)	paintsponsor character(5)
1	12345	Dallas Open	2015-01-10	11111	113
2	56789	Mid Atlantic Open	2015-05-10	22222	113
3	54321	Chicago Open	2015-06-10	33333	112
4	98765	West Coast Open	2015-09-10	44444	116
5	10101	World Cup	2015-09-10	55555	117

Lists NXL events along with their name, location, date played, venue, and paint sponsor. The primary key ‘eid’ has to be used since the event names get repeated season to season. The current snapshot represents one section of a season, in which no venue is used twice, however, in a full season/multiple seasons a venue can host more than one event.

Functional Dependencies:
eid->eName, datePlayed, vid, paintSponsor

EVENT TEAMS

```
CREATE TABLE event_teams(
    eid         CHAR(5) NOT NULL REFERENCES events(eid),
    tid         CHAR(4) NOT NULL REFERENCES teams(tid),
    PRIMARY KEY (eid, tid)
);
```

	eid character(5)	tid character(4)
1	12345	1001
2	12345	2002
3	56789	1001
4	56789	2002
5	54321	1001
6	54321	2002
7	98765	1001
8	98765	2002
9	10101	1001
10	10101	2002

Demonstrates which teams are playing at what events. Since many teams can play at many events throughout a season, there exists a many to many relationship between teams and events. Since the PK of 'event_teams' exists as a composite of the two foreign keys (eid) and (tid), many teams can play at many events.

ACCEPTED VENDORS

```
CREATE TABLE accepted_vendors(
    cid           CHAR(3) NOT NULL REFERENCES accepted_companies(cid),
    numOfTents    INT,
    PRIMARY KEY (cid)

);
```

	cid character(3)	numoftents integer
1	119	2
2	111	5
3	118	1
4	120	1

Functional Dependencies:
cid->numOfTents

This table consists are all the vendors that are present at NXL events along with their number of tents. The primary key exists as a foreign key from 'accepted_companies' as all accepted vendors are present in 'accepted_companies', but not all companies are vendors. Additionally, it should be noted that under these conditions, a vendor keeps the same number of tents from event to event.

EVENT VENDORS

```
CREATE TABLE event_vendors(
    eid      CHAR(5) NOT NULL REFERENCES events(eid),
    cid      CHAR(3) NOT NULL REFERENCES accepted_companies(cid),
    PRIMARY KEY (eid, cid)
);
```

	eid character(5)	cid character(3)
1	10101	119
2	10101	111
3	12345	119
4	12345	111
5	98765	112

This table demonstrates the vendors present at each individual event.

VIEWS

teamSponsors view

Lists all teams by their name and their appropriate sponsors.

```
CREATE VIEW teamSponsers AS
SELECT tname, accepted_companies.cname
FROM teams, team_sponsors, accepted_companies
WHERE teams.tid = team_sponsors.tid
AND team_sponsors.cid = accepted_companies.cid
ORDER BY tname DESC;
```

	tname text	cname text
1	Edmonton Impact	Planet Eclipse
2	Edmonton Impact	HK Army
3	Edmonton Impact	GI Sports
4	Edmonton Impact	KEE Action Sports
5	Chicago Aftershock	DYE Percision
6	Chicago Aftershock	MacDev Paintball
7	Chicago Aftershock	GI Sports

This view can be given to different companies so they can see the current sponsorship status of all teams and plan their marketing strategy accordingly. New and old companies can use this view to analyze where they can fit and develop a niche amongst teams.

VIEWS

eventVendors view

Displays all vendors and the events which they are present at.

```
CREATE VIEW eventVendors AS
SELECT accepted_companies.cName, event_vendors.eid, accepted_vendors.numOfTents
FROM accepted_companies, accepted_vendors, event_vendors
WHERE accepted_companies.cid = accepted_vendors.cid
AND accepted_vendors.cid = event_vendors.cid;
```

	cname text	eid character(5)	numoftents integer
1	Bob Long	10101	2
2	Planet Eclipse	10101	5
3	DYE Percision	10101	1
4	KEE Action Sports	10101	3
5	MacDev Paintball	10101	5
6	Bob Long	12345	2
7	Planet Eclipse	12345	5
8	DYE Percision	98765	1
9	Invert Paintball	98765	1
10	Invert Paintball	54321	1
11	Virtue Paintball	54321	2

This view can be used by NXL executives for the financial analysis of leasing vendor space. They can then see how well they entice vendors to purchase tent space, and what companies buy more/less and adjust their marketing accordingly.

VIEWS

teamsPlayingEvents view

Displays the teams playing at each event

```
CREATE VIEW teamsPlayingEvents AS
SELECT teams.tid, teams.tName, events.eid, events.eName
FROM teams, event_teams, events
WHERE teams.tid = event_teams.tid
AND event_teams.eid = events.eid
ORDER BY teams.tName DESC;
```

	tid character(4)	tname text	eid character(5)	ename text
1	1001	Edmonton Impact	56789	Mid Atlantic Open
2	1001	Edmonton Impact	98765	West Coast Open
3	1001	Edmonton Impact	54321	Chicago Open
4	1001	Edmonton Impact	10101	World Cup
5	1001	Edmonton Impact	12345	Dallas Open
6	2002	Chicago Aftershock	10101	World Cup
7	2002	Chicago Aftershock	12345	Dallas Open
8	2002	Chicago Aftershock	56789	Mid Atlantic Open
9	2002	Chicago Aftershock	54321	Chicago Open
10	2002	Chicago Aftershock	98765	West Coast Open

This view can be used by sponsors and NXL management alike. A sponsor might require certain teams to play in a certain amount of events in order to maintain their sponsorship package. Additionally, NXL management might be interested in what teams are specifically playing certain events in order to better market the event or ensure they are properly staffed. Furthermore, since it is possible for two teams to have the same name, 'tid' is present to distinguish the two. It should be noted that although uncommon, teams can opt out of events at their choosing.

STORED PROCEDURES

get_paint_sponsor

```
CREATE OR REPLACE FUNCTION get_paint_sponsor(CHAR(3), refcursor) RETURNS refcursor AS
$$
DECLARE
  paintCompany CHAR(3)       := $1;
  resultset    REFCURSOR := $2;

BEGIN
open resultset FOR
   SELECT eid, eName
   FROM events
   WHERE paintSponsor = paintCompany;
 RETURN resultset;
END;
$$
LANGUAGE plpgsql;

SELECT get_paint_sponsor('113', 'results');
FETCH ALL FROM results;
```

Allows a user to search which events are sponsored by a specific paint sponsor.

	eid character(5)	ename text
1	12345	Dallas Open
2	56789	Mid Atlantic Open

STORED PROCEDURES

get_team_sponsors

```
CREATE OR REPLACE FUNCTION get_team_sponsor(CHAR(4), REFCURSOR) RETURNS refcursor AS
$$
DECLARE
  teamInQuestion CHAR(4) := $1;
  resultset    REFCURSOR := $2;
BEGIN
open resultset FOR
  SELECT team_sponsors.cid, accepted_companies.cName
  FROM team_sponsors, accepted_companies, accepted_sponsors
  WHERE team_sponsors.tid = teamInQuestion
  AND team_sponsors.cid = accepted_sponsors.cid
  AND accepted_sponsors.cid = accepted_companies.cid;
 RETURN resultset;
END;
$$
LANGUAGE plpgsql;
```

	cid character(3)	cname text
1	113	KEE Action Sports
2	111	Planet Eclipse
3	117	HK Army
4	116	GI Sports

Allows for a user, most likely an accepted sponsor, to look up the sponsors of a specific team. Can be used in the case where one company wishes to sponsor a team but needs to know what companies the team endorses in an attempt to get a better understanding of the teams style of play.

QUERIES

```
SELECT(
 SELECT count(people.pid)
 FROM people
 ) AS num_of_people,
 (
 SELECT count(players.pid)
 FROM players
 )AS num_of_players
 FROM people
 LIMIT 1
```

Displays the number of players in respect to the number of participating people.

	num_of_people bigint	num_of_players bigint
1	31	17

QUERIES

```
SELECT accepted_companies.cid, cName, address, postalInfo.postalCode, postalInfo.state, postalInfo.city, contactInfo
FROM accepted_companies, postalInfo, accepted_sponsors
WHERE accepted_companies.cid = accepted_sponsors.cid
AND accepted_companies.postalCode = postalInfo.postalCode
```

	cid character(3)	cname text	address text	postalcode character(5)	state text	city text	contactinfo text
1	113	KEE Action Sports	KEY Paintball Drive	12345	Illinois	Springfield	KEE-ACTION@keeaction.com
2	111	Planet Eclipse	PE Road	90210	California	San Diego	PlanetEclipsePaintball@PE.com
3	112	DYE Percision	DYE Sports Way	10505	Massachussets	Springfield	DYEPaintball@dyeproducts.com
4	114	MacDev Paintball	BackRoad Drive	22134	New York	New York	MacDevSupport@MacDev.com
5	115	Virtue Paintball	BackRoad Drive	22134	New York	New York	MacDevSupport@PE.com
6	116	GI Sports	GI Sports Lane	10589	Florida	Tampa	GIproducts@gisports.com
7	117	HK Army	Hustle-Kids Lane	82654	New York	Poughkeepsie	Info@HKarmy.com

This query displays all accepted sponsors along with the entirety of their contact info.

QUERIES

```
SELECT count(players.pid) AS num_of_players, players.team
FROM players, teams
WHERE players.team = teams.tid
GROUP BY players.team
```

	num_of_players bigint	team character(4)
1	8	2002
2	9	1001

Portrays the number of players on each team

SECURITIES

There are three different users in this database at its current state. The first is the database administrator; he/she is given full select/insert/delete privileges on all tables in the database. Additionally, there is an event coordinator who is able to select, insert, and update on tables solely regarding events. Finally, there exists a NXL recruiter who is tasked to recruit not only new people/teams/coaches/staff but also new companies that can serve as sponsors and/or vendors. However, they gain no permissions on any table regarding events since their focus is to keep track of what people or companies are participating in the NXL.

```
CREATE ROLE database_admin

GRANT SELECT, INSERT, UPDATE, DELETE
ON ALL TABLES IN SCHEMA public
TO database_admin
```

```
CREATE ROLE event_coordinator

GRANT SELECT, INSERT, UPDATE
ON venues, events, event_teams, paint_sponsors, event_vendors
TO event_coordinator
```

```
CREATE ROLE nxl_recruiter

GRANT SELECT, INSERT, UPDATE
ON people, staff, players, coaches, teams, accepted_companies, accepted_sponsors,
   postal_info, team_sponsors, accepted_vendors
TO nxl_recruiter
```

KNOWN ISSUES

- Currently, based on the entity subtype relationships between accepted companies and accepted sponsors/paint sponsors, if a new company is added by the NXL as a team sponsor, they need to be inserted into three tables: 'accepted_companies', 'accepted_sponsors' and 'paint_sponsors'.

- The same goes for 'event_vendors'.

- Vendors are not allowed to change the number of tents they purchased. This means that if 'Bob Long' wants 10 tents for the first event because he foresees that many of his customers will attend, but wishes to reduce his tent number to 2 at the fifth event because his customer base isn't focused in the events city, he cannot do so; as that update would be deleting the history of him having the original 10 tents.

FURTHER IMPROVEMENTS

- With the current design, the database does not take into account what products companies sponsor certain teams with. This can be a problem if a company wishes to request information regarding a team sponsors, since they will not be able to see what individual products teams are using. In order to do this a new table would need to be created to hold all the NXL approved products along with what companies supply them.

- Additionally, in a future implication of this database, it would be useful to portray how much money a vendor is paying the NXL to be present at each event. However, in order for this to be calculated correctly the database would have to also display the cost NXL pays per event.

- Furthermore, it would be useful to keep track of what field layout is being used for each event. In order to accomplish this, a new table containing all possible field layouts would have to be created, having a one to many relationship from field layouts to events.