XCOMdB

By Kevin Mirsky

Table of Contents

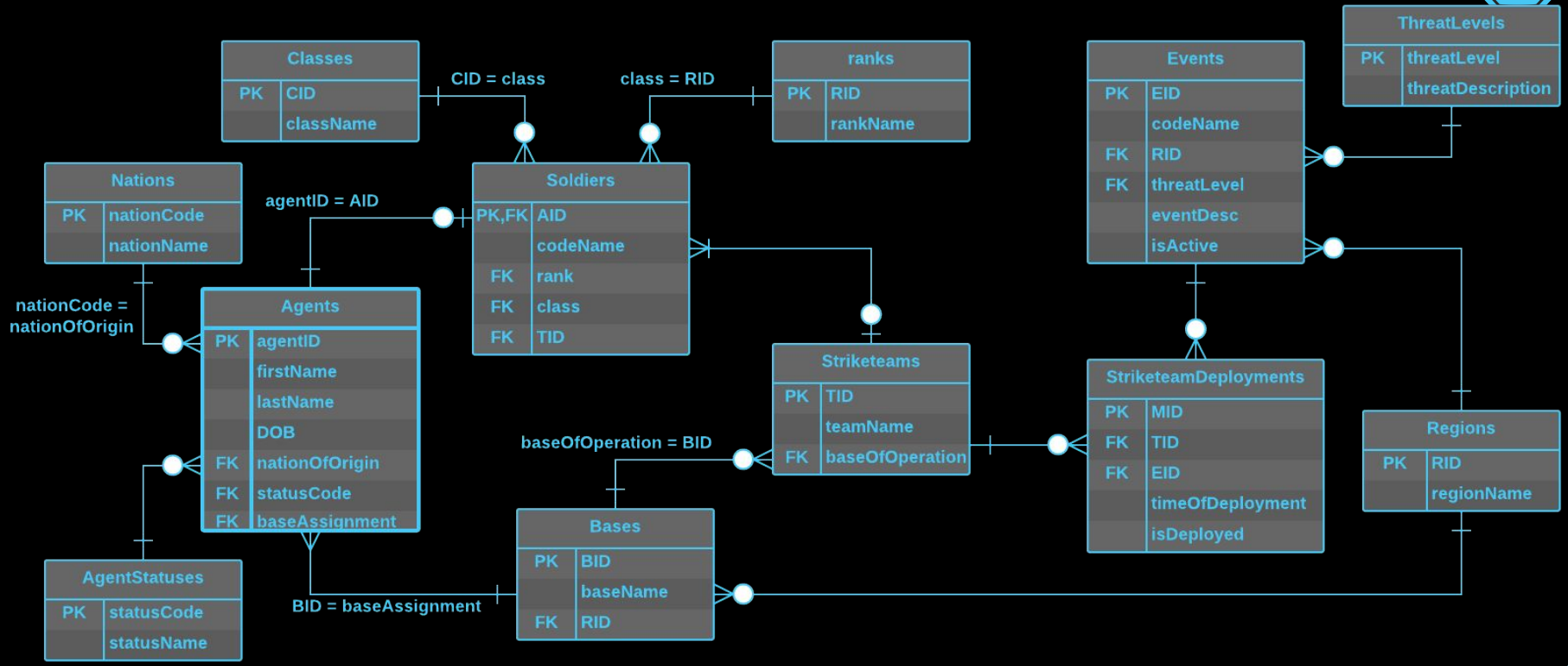
Executive Summary

In March 2015, hostile extraterrestrial activity was confirmed on Earth. In response, a secret council of nations approved the full activation of the XCOM Project, an international joint military organization tasked with combating the alien threat.
With the aliens confirmed hostile, the organization conducts routine combat operations to thwart alien attacks and missions. Since XCOM is tasked with the protection of the entire planet, it is critical they operate multiple bases and have teams ready to go at all times.

The purpose of this database is to assist XCOM in the assignment and deployment of soldiers towards counter-alien operations. The database tracks all agents of XCOM (including soldiers), strike teams of soldiers, and XCOM bases. It also tracks alien events and the deployment of strike teams to events. This will allow the Commander to quickly assess the situation and rapidly deploy teams with the most information at hand.

This is imperative to the success of the XCOM project as a team can make or break a mission. Providing this information to the Commander will ensure the greatest success of future missions and the survival of humanity as a whole.

ER Diagram


Classes
PK CID
className
CID = class
class = RID
ranks
PK RID
rankName
ThreatLevels
PK threatLevel
threatDescription
Events
PK EID
codeName
FK RID
FK threatLevel
eventDesc
isActive
Nations
PK nationCode
nationName
agentID = AID
Soldiers
PK,FK AID
codeName
FK rank
FK class
FK TID
nationCode = nationOfOrigin
Agents
PK agentID
firstName
lastName
DOB
FK nationOfOrigin
FK statusCode
FK baseAssignment
Striketeams
PK TID
teamName
FK baseOfOperation
StriketeamDeployments
PK MID
FK TID
FK EID
timeOfDeployment
isDeployed
Regions
PK RID
regionName
baseOfOperation = BID
Bases
PK BID
baseName
FK RID
AgentStatuses
PK statusCode
statusName
BID = baseAssignment

Tables

Regions

The region table lists all regions used to approximate the location of Bases and alien Events.

Create Statement:

```
CREATE TABLE Regions (
    RID          serial UNIQUE NOT NULL,
    regionName   text,
PRIMARY KEY(RID)
);
```

Functional Dependencies:

RID → regionName

rid integer	regionname text
1	North America
2	South America
3	Europe
4	Middle East
5	Africa
6	Asia
7	Oceania

Bases

This table lists all bases currently operated by XCOM.

Create Statement:

```
CREATE TABLE Bases (
     BID            serial UNIQUE NOT NULL,
     baseName       text,
     RID            integer references
Regions(RID) NOT NULL,
PRIMARY KEY(BID)
);
```

Functional Dependencies:

BID → baseName, RID

bid integer	basename text	rid integer
1	Area 51	1
2	Firebase Alpaca	2
3	Jackal Base	5
4	Kennedy Base	1
5	Alps Strikebase	3
6	Asian Coalition Base	6
7	Outback Base	7
8	Pyramid Base	4

ThreatLevels

This table lists the levels of threats used to evaluate alien events

Create Statement:

```
CREATE TABLE ThreatLevels (
     threatLevel    serial UNIQUE NOT NULL,
     threatName     text,
PRIMARY KEY(threatLevel)
);
```

Functional Dependencies:

threatLevel → threatName

threatlevel integer	threatname text
1	Minimal
2	Minor
3	Moderate
4	Substantial
5	High
6	Severe
7	Extreme
8	Critical

Events

This table lists the alien events recorded by XCOM which should be responded to.

Create Statement:

```
CREATE TABLE Events (
     EID            serial  UNIQUE NOT NULL,
     codeName       text,
     RID            integer references
Regions(RID),
     threatLevel    integer references
threatLevels(threatLevel),
     eventDesc      text,
     isActive       boolean NOT NULL,
     timeDetected   timestamp,
PRIMARY KEY(EID)
);
```

Functional Dependencies:

EID → codeName, RID, threatLevel, eventDesc, isActive, timeDetected

Continued...

Events

eid integer	codename text	rid integer	threatlevel integer	eventdesc text	isactive boolean	timedetected timestamp without time zon...
1	Fallen Star	5	3	A UFO touched down in the Nigerian interior	true	2017-04-20 15:10:18
2	Little Thieves	1	3	Reports of abductions in rural Kansas	false	2017-04-28 02:02:12
3	Streaked Sky	1	2	Possible UFO spotting in Canada	false	2016-12-15 20:32:08
4	Vengeful Demon	6	5	Alien attack on Chinese city	false	2017-02-01 10:45:59
5	Scornful Father	3	1	Signs of alien activity in German forest	false	2017-03-11 04:22:13
6	Growling Dirt	2	2	Reports of alien scouts in Peruvian outskirts	true	2017-04-20 08:01:02
7	Big Ocean	1	2	Reports of submerged UFO in Southern Atlantic Oc...	true	2017-04-30 15:13:38

Nations

This table lists all the nations from which XCOM agents are recruited from.

Create Statement:

```
CREATE TABLE Nations (
     nationCode          text UNIQUE NOT NULL,
     nationName          text NOT NULL,
PRIMARY KEY(nationCode)
);
```

Functional Dependencies:

nationCode → nationName

nationcode text	nationname text
US	United States of America
CA	Canada
MX	Mexico
BR	Brazil
CL	Chile
JP	Japan
KR	South Korea
CN	China
TH	Thailand
AU	Australia
GB	United Kingdom
DE	Germany
FR	France
PL	Poland
ZA	South Africa

AgentStatuses

This table lists all the statuses (such as Active, MIA, KIA) for agents of XCOM.

Create Statement:

```
CREATE TABLE AgentStatuses (
     statusCode     serial UNIQUE NOT NULL,
     statusName     text,
PRIMARY KEY(statusCode)
);
```

Functional Dependencies:

statusCode → statusName

statuscode integer	statusname text
0	KIA
1	Active
2	Wounded
3	Gravely Wounded
4	MIA
5	Retired
6	Removed from Duty

Agents

This table lists all employees of XCOM, which are referred to as “Agents”

Create Statement:

```
CREATE TABLE Agents (
     AID            serial UNIQUE NOT NULL,
     firstName      text,
     lastName       text,
     DOB            date,
     nationOfOrigin text references
Nations(nationCode),
     statusCode     integer references
AgentStatuses(statusCode) NOT NULL DEFAULT 1,
baseAssignment integer references Bases(BID),
PRIMARY KEY(AID)
);
```

Functional Dependencies:

AID → firstName, lastName, DOB, nationOfOrigin, statusCode, baseAssignment

Continued...

Agents

aid integer	firstname text	lastname text	dob date	nationoforigin text	statuscode integer	baseassign... integer
1	Peter	Van Doorn	1987-11-02	US	1	5
2	Alan	Labouseur	[null]	US	1	2
3	Maria	Klein	1975-02-23	DE	1	5
4	Wiktor	Przybylowicz	1992-09-01	PL	1	5
5	Akio	Takahashi	1993-03-15	JP	0	6
6	Jung	Kim	1990-01-19	KR	1	6
7	Kathy	Taylor	1991-07-07	GB	2	6
8	Tien	Liengtiraphan	1996-05-25	TH	1	2
9	David	Windon	1991-11-21	AU	1	3
10	Juan	García	1989-02-11	MX	1	3
11	James	Wells	1985-06-12	GB	1	3
12	Hanna	Windon	1993-05-27	AU	1	7
13	Maria	Perez	1992-11-05	MX	1	7
14	Lucas	Qi	1995-02-12	CA	1	8
15	Kate	Shepard	1988-11-29	US	1	8
16	Hans	Weber	1992-03-11	DE	1	6

Ranks

This table lists all the ranks given to XCOM soldiers.

Create Statement:

```
CREATE TABLE Ranks (
     RID            serial UNIQUE NOT NULL,
     rankName       text,
     PRIMARY KEY(RID)
     );
```

Functional Dependencies:

RID → rankName

rid integer	rankname text
1	Squaddie
2	Lance Corporal
3	Corporal
4	Sergeant
5	Staff Sergeant
6	Master Sergeant
7	Lieutenant
8	Captain
9	Major
10	Colonel
11	Field Commander

Classes

This table lists all the skill classes XCOM soldiers can classified in.

Create Statement:

```
CREATE TABLE Classes (
     CID            serial UNIQUE NOT NULL,
     className      text,
     PRIMARY KEY(CID)
     );
```

Functional Dependencies:

CID → className

cid integer	classname text
1	Rookie
2	Assault
3	Grenadier
4	Gunner
5	Ranger
6	Sharpshooter
7	Shinobi
8	Specialist
9	Technical
10	Psi Operative

Striketeams

This table lists the teams XCOM soldiers can be organized into.

Create Statement:

```
CREATE TABLE Striketeams (
     TID             serial UNIQUE NOT NULL,
     teamName        text,
     baseOfOperation integer references
Bases(BID),
PRIMARY KEY(TID)
);
```

Functional Dependencies:

TID → teamName, baseOfOperation

tid integer	teamname text	baseofoper... integer
1	Final Normal Form	2
2	One Man Army	5
3	The Hounds	3
4	Wavemakers	7
5	Spectres	8
6	Snakehead	6

Soldiers

This table lists all XCOM agents which are classified as soldiers for anti-alien operations.

Create Statement:

```
CREATE TABLE Soldiers (
      AID         integer references Agents(AID) UNIQUE NOT NULL,
      codeName    text,
      rank        integer references Ranks(RID) DEFAULT 1,
      class integer references Classes(CID) DEFAULT 1,
      TID         integer references Striketeams(TID),
PRIMARY KEY(AID)
);
```

Functional Dependencies:

AID → codeName, rank, class, TID

Continued...

Soldiers

aid integer	codename text	rank integer	class integer	tid integer
1	The General	10	5	2
2	The Normalizer	8	8	1
4		1	1	[null]
5	Wolf	3	2	[null]
7	Kat	2	5	[null]
8	Flourish	6	7	1
9	Goat	4	4	3
10		3	8	3
11	H.G.	5	6	3
12	Sparkle	1	1	4
13	Dragontooth	7	9	4
14	Maple	3	4	5
15	Renegade	9	6	5
16	Viper	7	7	6

StriketeamDeployments

This table lists the missions striketeams are deployed on.

Functional Dependencies:
MID → TID, EID, timeOfDeployment, isDeployed

Create Statement:

```
CREATE TABLE StriketeamDeployments (
      MID   serial UNIQUE NOT NULL,
      TID   integer references Striketeams(TID) NOT NULL,
      EID   integer references Events(EID) NOT NULL,
      timeOfDeployment  timestamp,
      isDeployed        boolean NOT NULL,
PRIMARY KEY(MID)
);
```

mid integer	tid integer	eid integer	timeofdeployment timestamp without time zone	isdeplo... boolean
1	1	6	2017-04-20 08:12:32	true
2	3	5	2017-03-11 08:52:22	false
3	6	4	2017-02-01 11:01:22	false
4	1	3	2016-12-16 01:22:21	false
5	4	2	2017-04-28 02:04:03	false

Views

SoldierInfo

Returns all relevant info about each soldier with readability in mind.

Create Statement:

```
CREATE OR REPLACE VIEW SoldierInfo AS
SELECT
	Soldiers.AID,
	Agents.firstName,
	Agents.lastName,
	Soldiers.CodeName,
	Nations.nationName,
	Ranks.rankName,
	Classes.className,
	Bases.baseName,
	Striketeams.teamName,
	AgentStatuses.statusName,
	Agents.DOB
FROM Soldiers
		INNER JOIN Agents ON Agents.AID = Soldiers.AID
		LEFT JOIN Nations ON Nations.nationCode =
Agents.nationOfOrigin
		LEFT JOIN Bases ON Bases.BID =
Agents.baseAssignment
		LEFT JOIN Ranks ON Ranks.RID = Soldiers.rank
		LEFT JOIN Classes ON Classes.CID = Soldiers.class
		LEFT JOIN AgentStatuses ON
AgentStatuses.statusCode = Agents.statusCode
		LEFT JOIN Striketeams ON Striketeams.TID =
Soldiers.TID;
```

Continued...

SoldierInfo

aid integer	firstname text	lastname text	codename text	nationname text	rankname text	classname text	basename text	teamname text	statusname text	dob date
1	Peter	Van Doorn	The General	United States of America	Colonel	Ranger	Alps Strikebase	One Man Army	Active	1987-11-02
2	Alan	Labouseur	The Normalizer	United States of America	Captain	Specialist	Firebase Alpaca	Final Normal Form	Active	[null]
4	Wiktor	Przybylowicz		Poland	Squaddie	Rookie	Alps Strikebase	[null]	Active	1992-09-01
5	Akio	Takahashi	Wolf	Japan	Corporal	Assault	Asian Coalition Base	[null]	KIA	1993-03-15
7	Kathy	Taylor	Kat	United Kingdom	Lance Corporal	Ranger	Asian Coalition Base	[null]	Wounded	1991-07-07
8	Tien	Liengtiraphan	Flourish	Thailand	Master Sergeant	Shinobi	Firebase Alpaca	Final Normal Form	Active	1996-05-25
9	David	Windon	Goat	Australia	Sergeant	Gunner	Jackal Base	The Hounds	Active	1991-11-21
10	Juan	García		Mexico	Corporal	Specialist	Jackal Base	The Hounds	Active	1989-02-11
11	James	Wells	H.G.	United Kingdom	Staff Sergeant	Sharpshooter	Jackal Base	The Hounds	Active	1985-06-12
12	Hanna	Windon	Sparkle	Australia	Squaddie	Rookie	Outback Base	Wavemakers	Active	1993-05-27
13	Maria	Perez	Dragontooth	Mexico	Lieutenant	Technical	Outback Base	Wavemakers	Active	1992-11-05
14	Lucas	Qi	Maple	Canada	Corporal	Gunner	Pyramid Base	Spectres	Active	1995-02-12
15	Kate	Shepard	Renegade	United States of America	Major	Sharpshooter	Pyramid Base	Spectres	Active	1988-11-29
16	Hans	Weber	Viper	Germany	Lieutenant	Shinobi	Asian Coalition Base	Snakehead	Active	1992-03-11

UnrespondedEvents

It's important that all alien events are responded to. In order to ensure none are forgotten, the view displays all active events that have not yet had a striketeam dispatched. It also displays how much time has passed since the Event was logged, giving the user a better idea where to focus first.

Create Statement:

```
CREATE OR REPLACE VIEW UnrespondedEvents AS
SELECT
      Events.EID,
      Events.CodeName,
      Regions.RegionName,
      ThreatLevels.threatName,
      Events.EventDesc,
      Age(now(), Events.timeDetected) as timeSinceReported
FROM Events
      LEFT JOIN Regions ON Regions.RID = Events.RID
      LEFT JOIN ThreatLevels ON ThreatLevels.ThreatLevel = Events.ThreatLevel
WHERE
      EID NOT IN (SELECT EID FROM StriketeamDeployments)
      AND isActive = True;
```

Continued...

UnrespondedEvents

eid integer	codename text	regionname text	threatname text	eventdesc text	timesincereported interval
1	Fallen Star	Africa	Moderate	A UFO touched down in the Nigerian interior	11 days 09:30:51.722134
7	Big Ocean	North America	Minor	Reports of submerged UFO in Southern Atlantic Ocean	1 day 09:27:31.722134

EventHistory

It is important to review past events, as well as XCOM's response to them. This view displays every logged Event alongside the records of what team(s) responded to it.

The time it took to for the striketeam to be deployed is calculated and displayed, where applicable.

The data is ordered by Event date, sorted in Descending order.

Create Statement:

```
CREATE OR REPLACE VIEW EventHistory AS
SELECT
      Events.EID,
      Events.CodeName,
      Regions.RegionName,
      ThreatLevels.threatName,
      Events.EventDesc,
      Events.timeDetected,
      Striketeams.teamName as RespondingTeam,
      Age(StriketeamDeployments.timeOfDeployment,
Events.timeDetected) as ResponseTime,
      Events.isActive
FROM Events
      LEFT JOIN Regions ON Regions.RID =
Events.RID
      LEFT JOIN ThreatLevels ON
ThreatLevels.ThreatLevel = Events.ThreatLevel
      LEFT JOIN StriketeamDeployments ON
StriketeamDeployments.EID = Events.EID
      LEFT JOIN Striketeams ON Striketeams.TID =
StriketeamDeployments.TID
ORDER BY Events.timeDetected DESC;
```

Continued...

EventHistory

eid integer	codename text	regionname text	threatname text	eventdesc text	timedetected timestamp without time zo...	respondingteam text	responsetime interval	isactive boolean
7	Big Ocean	North America	Minor	Reports of submerged UFO in Southern Atlantic Oc...	2017-04-30 15:13:38	[null]	[null]	true
2	Little Thieves	North America	Moderate	Reports of abductions in rural Kansas	2017-04-28 02:02:12	Wavemakers	00:01:51	false
1	Fallen Star	Africa	Moderate	A UFO touched down in the Nigerian interior	2017-04-20 15:10:18	[null]	[null]	true
6	Growling Dirt	South America	Minor	Reports of alien scouts in Peruvian outskirts	2017-04-20 08:01:02	Final Normal Form	00:11:30	true
5	Scornful Father	Europe	Minimal	Signs of alien activity in German forest	2017-03-11 04:22:13	The Hounds	04:30:09	false
4	Vengeful Demon	Asia	High	Alien attack on Chinese city	2017-02-01 10:45:59	Snakehead	00:15:23	false
3	Streaked Sky	North America	Minor	Possible UFO spotting in Canada	2016-12-15 20:32:08	Final Normal Form	04:50:13	false

Reports

Lifetime Events by Region

It is important to know where the most alien activity occurs. This report creates an easy to read table that shows the number of events per region.

Query:

```
SELECT Regions.RegionName, count(Events.RID) as
LifetimeEvents
FROM Regions
      INNER JOIN Events ON Regions.RID =
Events.RID
GROUP BY Regions.RegionName
ORDER BY LifetimeEvents DESC;
```

regionname text	lifetimeevents bigint
North America	3
Africa	1
Asia	1
South America	1
Europe	1

Ready Soldiers by Class

This query retrieves the number of **combat ready** soldiers for each class. This way XCOM can know what skillsets they are lacking and optimize the training of new troops.

classname text	numsoldiers bigint
Rookie	2
Sharpshooter	2
Specialist	2
Gunner	2
Shinobi	2
Technical	1
Ranger	1
Psi Operative	0
Assault	0
Grenadier	0

Query:

```
SELECT Classes.ClassName,
count(activeSoldiers.class) as NumSoldiers
FROM Classes
      LEFT JOIN (
                  SELECT Soldiers.Class
                  FROM  Soldiers
                  WHERE Soldiers.AID IN
                        (
                              SELECT Agents.AID
                              FROM Agents
                              WHERE
Agents.statusCode = 1
                        )
                  )
            AS ActiveSoldiers
            ON ActiveSoldiers.class = Classes.CID
GROUP BY Classes.ClassName
ORDER BY numSoldiers DESC;
```

Time since last event per Region

Gets time since last event per region. Allows XCOM to see what regions have been hit most recently and can also help see what regions may be "overdue" for hostilities.

The table displays the regions which have had an attack most recently first.

Query:

```
SELECT Regions.RegionName,
Min(Age(Events.TimeDetected)) as TimeSinceLast
FROM Regions
      LEFT JOIN Events ON Regions.RID = Events.RID
GROUP BY Regions.RegionName
ORDER BY TimeSinceLast ASC;
```

regionname text	timesincelast interval
North America	1 day 08:46:22
Africa	11 days 08:49:42
South America	11 days 15:58:58
Europe	1 mon 21 days 19:37:47
Asia	3 mons 13:14:01
Oceania	[null]
Middle East	[null]

Stored Procedures

rebaseTeam()

Many times a striketeam will be relocated to another base. This can be tedious if updated manually, as each member has to be moved. This has the potential for error. Instead, this function can take a TeamID and a BaseID, and it will relocate each team member to the given base.

The team will then be marked as operating from that base as well.

Create Statement:

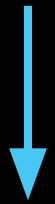
```
CREATE OR REPLACE FUNCTION rebaseTeam(int, int)
RETURNS void AS
$rebaseTeam$
      DECLARE
            teamID int := $1;
            baseID int := $2;
      BEGIN
            UPDATE Striketeams
            SET baseOfOperation = baseID
            WHERE TID = teamID;
            UPDATE Agents
            SET baseAssignment = baseID
            WHERE AID IN (    SELECT AID
                              FROM Soldiers
                              WHERE TID = teamID
                        );
      END; $rebaseTeam$ LANGUAGE plpgsql;
```

rebaseTeam() Results

rebaseTeam(1, 8)

firstname text	lastname text	codename text	rankname text	teamname text	basename text
Alan	Labouseur	The Normalizer	Captain	Final Normal Form	Firebase Alpaca
Tien	Liengtiraphan	Flourish	Master Sergeant	Final Normal Form	Firebase Alpaca

tid integer	teamname text	baseofoper... integer
1	Final Normal Form	2



firstname text	lastname text	codename text	rankname text	teamname text	basename text
Alan	Labouseur	The Normalizer	Captain	Final Normal Form	Pyramid Base
Tien	Liengtiraphan	Flourish	Master Sergeant	Final Normal Form	Pyramid Base

tid integer	teamname text	baseofoper... integer
1	Final Normal Form	8

completeMission()

When a team returns from a mission, the event should typically be resolved. Using this function, it will set the striketeam's deployment as no longer deployed, and then mark the event as over. Simply enter the MID for the mission as an argument.

Create Statement:

```
CREATE OR REPLACE FUNCTION completeMission(int)
RETURNS void AS
$completeMission$
      DECLARE
            missionID int := $1;
      BEGIN
            UPDATE StriketeamDeployments
            SET isDeployed = FALSE
            WHERE MID = missionID;
            UPDATE Events
            SET isActive = FALSE
            WHERE EID in
                  (
                        SELECT EID
                        FROM StriketeamDeployments
                        WHERE MID = missionID
                  );
      END
$completeMission$ LANGUAGE plpgsql;
```

completeMission() Results

completeMission(1)

eid integer	codename text	rid integer	threatlevel integer	eventdesc text	isactive boolean	timedetected timestamp without time zone
6	Growling Dirt	2	2	Reports of alien scouts in Peruvian outskirts	true	2017-04-20 08:01:02

mid integer	tid integer	eid integer	timeofdeployment timestamp without time zone	isdeplo... boolean
1	1	6	2017-04-20 08:12:32	true

eid integer	codename text	rid integer	threatlevel integer	eventdesc text	isactive boolean	timedetected timestamp without time zone
6	Growling Dirt	2	2	Reports of alien scouts in Peru...	false	2017-04-20 08:01:02

mid integer	tid integer	eid integer	timeofdeployment timestamp without time zone	isdeplo... boolean
1	1	6	2017-04-20 08:12:32	false

Triggers

autoMoveToTeamLocation

A soldier needs to be with their teammates to function as a team. Effectively, a soldier should always be in the same base as the team. This function will automatically relocate a soldier to the proper base when assigned a new team.

Create Statement:

```
CREATE OR REPLACE FUNCTION moveToTeam()
RETURNS TRIGGER AS
$moveToTeam$
      DECLARE
            baseAgent int := NULL;
            baseTeam  int := NULL;
            newTeam   int := NULL;
      BEGIN
            SELECT
                  Striketeams.baseOfOperation
            INTO baseTeam
            FROM Striketeams
            WHERE Striketeams.TID = NEW.TID;
            SELECT
                  Agents.baseAssignment
            INTO baseAgent
            FROM Agents
            WHERE Agents.AID = NEW.AID;
            IF (baseTeam != baseAgent) THEN
                  UPDATE Agents
                  SET baseAssignment = baseTeam
                  WHERE Agents.AID = NEW.AID;
            END IF;
            RETURN NEW;
      END;
$moveToTeam$ LANGUAGE plpgsql;


CREATE TRIGGER autoMoveToTeamLocation
      BEFORE INSERT OR UPDATE OF TID ON Soldiers
      FOR EACH ROW
      EXECUTE PROCEDURE moveToTeam();
```

autoMoveToTeamLocation

aid integer	firstname text	lastname text	basename text	teamname text
1	Peter	Van Doorn	Alps Strikebase	One Man Army
2	Alan	Labouseur	Pyramid Base	Final Normal Form

```
UPDATE SOLDIERS
SET TID = 1
WHERE AID = 1;
```

aid integer	firstname text	lastname text	basename text	teamname text
1	Peter	Van Doorn	Pyramid Base	Final Normal Form
2	Alan	Labouseur	Pyramid Base	Final Normal Form

check_if_active

A striketeam cannot be deployed on a mission when it is already deployed on another mission. This trigger checks to see if the team a user is trying to deploy is already deployed. If so, it raises an exception. Otherwise, it allows the insert to proceed as normal.

Create Statement:

```
CREATE OR REPLACE FUNCTION checkIfActive()
RETURNS TRIGGER AS
$checkIfActive$
      BEGIN
            IF (NEW.isDeployed = TRUE) THEN
                  IF EXISTS
                  (
                        SELECT
StriketeamDeployments.isDeployed
                        FROM StriketeamDeployments
                        WHERE
StriketeamDeployments.TID = NEW.TID AND isDeployed
= TRUE
                  )
                  THEN
                        RAISE EXCEPTION 'Cannot
deploy a team that is already deployed!';
                        RETURN NULL;
                  END IF;
            END IF;
            RETURN NEW;
      END
$checkIfActive$ LANGUAGE plpgsql;

CREATE TRIGGER check_if_active
      BEFORE INSERT ON StriketeamDeployments
      FOR EACH ROW
      EXECUTE PROCEDURE checkIfActive();
```

check_if_active

mid integer	tid integer	eid integer	timeofdeployment timestamp without time zone	isdeplo... boolean
4	1	3	2016-12-16 01:22:21	false
1	1	6	2017-04-20 08:12:32	true

```
INSERT INTO StriketeamDeployments (EID, TID, timeOfDeployment, isDeployed)
VALUES
     (5, 1, '2017-04-20 08:12:32', TRUE);
```

```
ERROR:  Cannot deploy a team that is already deployed!
CONTEXT:  PL/pgSQL function checkifactive() line 11 at RAISE
********** Error **********

ERROR: Cannot deploy a team that is already deployed!
SQL state: P0001
Context: PL/pgSQL function checkifactive() line 11 at RAISE
```

Security

Admin

The admin role is given to trusted XCOM IT technicians whose main responsibility is overseeing the functionality of the database. This means they have full access to the database in order to maintain it.

Create Statement:

```
CREATE ROLE admin;
GRANT SELECT, INSERT, UPDATE, DELETE
      ON ALL TABLES IN SCHEMA PUBLIC
TO admin;
```

Commander

The Commander is the appointed leader of XCOM and oversees ALL of its functions, from recruitment to event response.

The main purpose of this database is to assist him/her in their operation of XCOM.
As such, he has almost complete control over the database.

However, there is little reason to give the Commander the ability to DELETE records on most tables, as this should not be required in day-to-day operation. This should only be necessary on Striketeams to remove a team.

Should the need arise, an Admin can be called upon by the Commander.

Create Statement:

```
CREATE ROLE Commander;
GRANT SELECT, INSERT, UPDATE
      ON ALL TABLES IN SCHEMA PUBLIC
TO Commander;
GRANT DELETE
      ON Striketeams
TO Commander;
```

Officer

Officers, or soldiers in combat leadership positions, are allowed to access information about other soldiers for the purposes of putting together striketeams.

However, since the decision is ultimately up to the Commander, they can make no modifications.

Create Statement:

```
CREATE ROLE Officer;
GRANT SELECT ON
      Agents,
      Nations,
      AgentStatuses,
      Soldiers,
      Striketeams,
      Bases,
      Regions,
      Ranks,
      Classes
TO Officer;
```

Dispatch

24/7, certain agents are assigned to monitor for alien events. When one is detected, they must be able to log it into the database. This role gives those agents that power.

Create Statement:

```
CREATE ROLE Dispatch;
GRANT SELECT ON
      threatLevels,
      Regions,
      Events
TO Dispatch;
GRANT INSERT, UPDATE
      ON Events
TO Dispatch;
```

HR

The Commander has more important things to than input the data of all incoming Agents. Therefore, some agents are in charge of the processing of incoming Agents. Thus, they must have the ability to perform the required operations.

Create Statement:

```
CREATE ROLE HR;
GRANT SELECT ON
      Nations,
      Agents,
      AgentStatuses,
      Bases,
      Regions,
      Soldiers,
      Classes,
      Ranks
TO HR;
GRANT INSERT, UPDATE
      ON Agents, Soldiers
TO HR;
```

Implementation Notes

Implementation Notes

-- The primary purpose of this database is the coordination of the military operations of XCOM. Non-combat staff can be placed in the system, but there is no administrative capabilities on them besides their current status.

-- The provided sample data does not include enough non-combat staff to populate all bases. In reality, there should be agents at every base. However, those were omitted in order to provide a clearer picture of the main purpose of the database.

Known Issues

Known Issues

-- In the UnrespondedEvents view, the timeSinceReported values display seconds to an obnoxiously long decimal. This is because the Age() function computes from midnight instead the current time, requiring a calculation instead to avoid, resulting in the decimals.

-- Agents with statuses that should prevent them from serving (such as KIA, retired, etc.) are not prevented from being enrolled in teams.

-- A soldier can be moved to different base from their team, leaving the team fragmented. Ideally, they should be dropped from the team or have the team moved with them.

Future Enhancements

Future Enhancements

-- Map Nations to Regions and use that in place of Regions for Events to better pinpoint location.

-- Add more checks and policies for Soldiers who have been wounded or no longer serving XCOM.

-- Implement more checks on updated/inserted data to avoid erroneous input

-- Create Table for Skyrangers (Troop Transport Jets) to coordinate Striketeam Deployments. If there are no Skyrangers available at the base, a Striketeam cannot deploy to an event.

-- Expand database to cover the other branches of XCOM staff, such as Engineering and Science departments