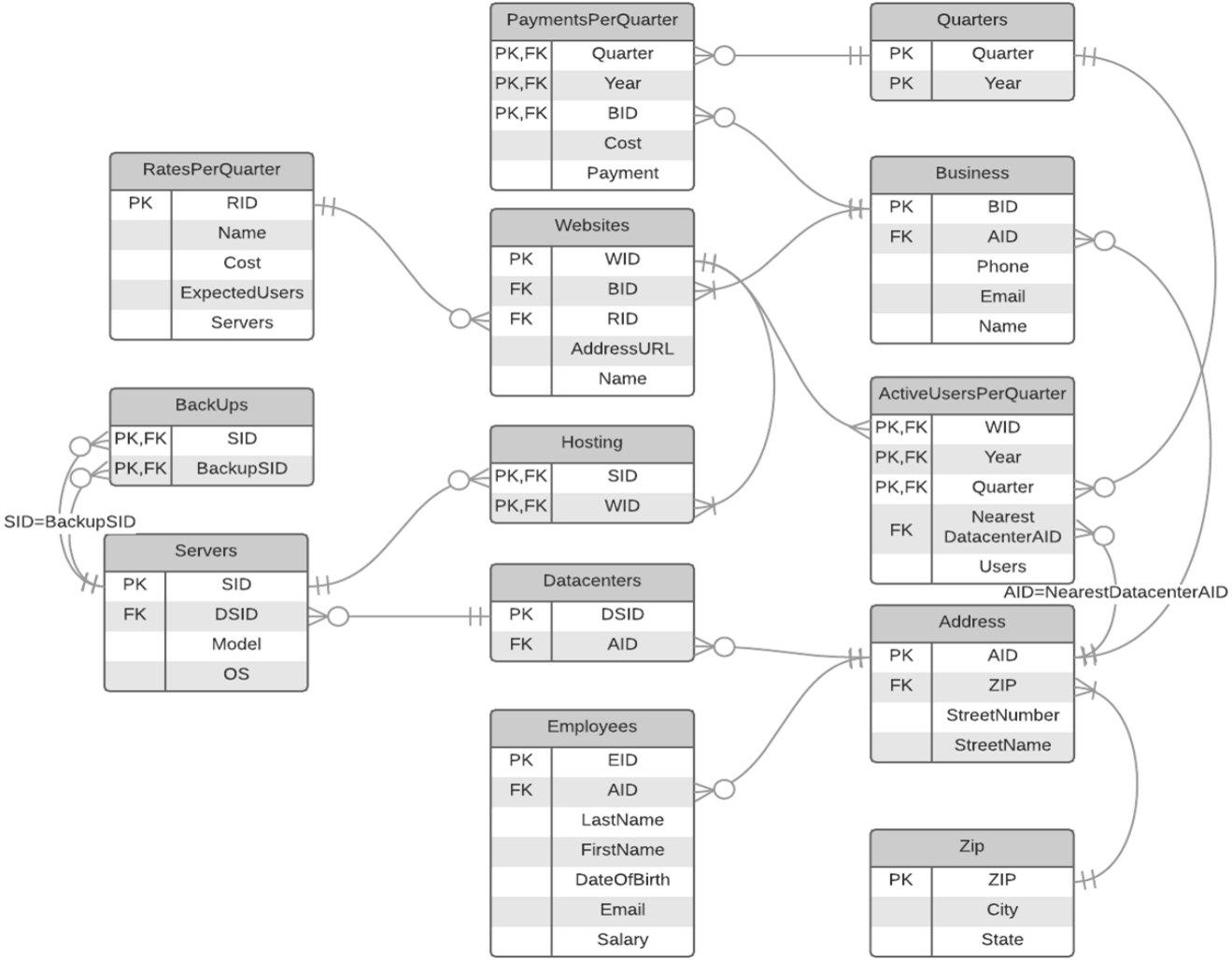
Cumulus

By Evan McElheny

Summary

Welcome to Cumulus, your source of all of you web hosting needs. All our clients have their own website (s). These have to be hosted on servers (with backups at a different location). Our rates are based on the average monthly users for a particular website. Naturally there is the option for premium servers, with more redundancy, near high traffic areas to reduce latency. We hope to provide for all of your web hosting needs

ERD


PaymentsPerQuarter
PK,FK Quarter
PK,FK Year
PK,FK BID
Cost
Payment
Quarters
PK Quarter
PK Year
RatesPerQuarter
PK RID
Name
Cost
ExpectedUsers
Servers
Websites
PK WID
FK BID
FK RID
AddressURL
Name
Business
PK BID
FK AID
Phone
Email
Name
BackUps
PK,FK SID
PK,FK BackupSID
SID=BackupSID
Hosting
PK,FK SID
PK,FK WID
ActiveUsersPerQuarter
PK,FK WID
PK,FK Year
PK,FK Quarter
FK Nearest DatacenterAID
Users
AID=NearestDatacenterAID
Servers
PK SID
FK DSID
Model
OS
Datacenters
PK DSID
FK AID
Address
PK AID
FK ZIP
StreetNumber
StreetName
Employees
PK EID
FK AID
LastName
FirstName
DateOfBirth
Email
Salary
Zip
PK ZIP
City
State

Tables

RatesPerQuarter

Contains the different quarterly rates that we offer businesses.

```
CREATE TABLE RatesPerQuarter(
  RID CHAR(5) NOT NULL,
  Name CHAR(15) NOT NULL,
  Cost NUMERIC(10,2) NOT NULL,
  ExpectedUsers INTEGER,
  Servers SMALLINT NOT NULL,
  PRIMARY KEY (RID)
);
```

RID -> Name, Cost, ExpectedUsers, Servers

RID	Name	Cost	ExpectedUsers
r0001	Basic	5000	1500
r0002	Redundant	7500	2500
r0003	Preimum	12500	4500

Quarters

Contains a row for every quarter that our business has operated.

```
CREATE TABLE Quarters(
  Quarter SMALLINT NOT NULL CHECK (Quarter = 1 OR Quarter = 2 OR Quarter = 3
OR Quarter = 4),
  Year SMALLINT NOT NULL,
  UNIQUE (Quarter, Year),
  PRIMARY KEY(Quarter, Year)
);
```

Quarter	Year
3	2015
4	2015
1	2016
2	2016
3	2016
4	2016
1	2017

Zip

Contains a zip code for every address.

```
CREATE TABLE Zip (
  ZIP CHAR(5) NOT NULL,
  City CHAR(20) NOT NULL,
  State CHAR(2) NOT NULL,
  PRIMARY KEY (ZIP)
);
```

ZIP -> City, State

ZIP	CIty	State
7090	Westfield	NJ
20815	Chevy Chase	MD
60185	West Chicago	IL
32904	Melbourne	FL
1801	Woburn	MA
19064	Springfield	PA
11967	Shirley	NY
8865	Phillipsburg	NJ
54115	De Pere	WI
14701	Jamestown	NY
12601	Poughkeepsie	NY

Address

Contains an address for every entity that has a location (Datacenters, Employees Businesses and averageUserLocation)

```
CREATE TABLE Address (
  AID CHAR(5) NOT NULL,
  ZIP CHAR(5) NOT NULL REFERENCES Zip(ZIP),
  StreetNumber SMALLINT NOT NULL,
  StreetName CHAR(25) NOT NULL,
  PRIMARY KEY (AID)
);
```

AID -> ZIP, StreetNumber, StreetName

AID	ZIP	StreetNumber	StreetName
a0001	7090	635	Lenox Ave.
a0002	12601	3399	North Rd.
a0003	7090	634	Lenox Ave.
a0004	20815	71	Pilgrim, Ave.
a0005	60185	44	Shirley Ave.
a0006	32904	123	6th St
a0007	54115	599	Glendale Street
a0008	8865	912	Old Talbot Dr.
a0009	11967	410	Brickyard Dr.
a0010	19064	545	St Margarets Ave.
a0011	1801	160	S. Woodsman Court
a0012	1801	12	Main St
a0013	1801	15	Main St
a0014	14701	797	W. High Point St.
a0015	14701	9187	Green Ave.

Datacenters

Contains every datacenter and where the datacenter is.

```
CREATE TABLE Datacenters (
  DSID CHAR(5) NOT NULL,
  AID CHAR(5) NOT NULL REFERENCES Address(AID),
  PRIMARY KEY (DSID)
);
```

DSID -> AID

DSID	AID
ds001	a0011
ds002	a0012
ds003	a0013
ds004	a0014
ds005	a0015
ds006	a0004
ds007	a0005

Employees

Contains all relevant information on our employees

```
CREATE TABLE Employees (
  EID CHAR(5) NOT NULL,
  AID CHAR(5) NOT NULL REFERENCES Address(AID),
  LastName CHAR(25) NOT NULL,
  FirstName CHAR(15) NOT NULL,
  DateOfBirth DATE NOT NULL,
  Email CHAR(35) NOT NULL,
  Salary NUMERIC(10,2) NOT NULL,
  PRIMARY KEY (EID)
);
```

EID -> AID, LastName, FirstName, DateOfBirth, Email, Salary

EID	AID	LastName	FirstName	DateOfBirth	Email	Salary
e0001	a0003	McElheny	Evan	1997-08-29	mcelevan12@gmail.com	1000000
e0002	a0004	Gornendaz	Sean	1993-11-09	sgorrrr@gmail.com	35000
e0003	a0002	Labouseur	Alan	1963-02-01	alan@3NFconsulting.com	12345
e0004	a0001	Liengtiraphan	Pradon	1991-05-06	Minion@3NFconsulting.com	54321

Business

Contains information on every business that we have hosted a website for.

```
CREATE TABLE Business (
  BID CHAR(5) NOT NULL,
  AID CHAR(5) NOT NULL REFERENCES Address(AID),
  Phone CHAR(14) NOT NULL,
  Email CHAR(55) NOT NULL,
  Name CHAR(50) NOT NULL,
  PRIMARY KEY (BID)
);
```

BID -> AID, Phone, Email, Name

BID	AID	Phone	Email	Name
b0001	a0005	(908)-380-8693	Cool@coolCatz.com	CoolCatz
b0002	a0006	(202)-555-0132	SOEA@SOE.com	SOE
b0003	a0007	(302)-555-0192	JenidMarkets@Jenid.com	Jenid Markets
b0004	a0008	(317)-555-0114	Marc@MarcRentals.com	Marc Rentals
b0005	a0009	(410)-555-0124	ASEE@ASEE.com	American Society Electrical Engineering

Websites

Contains information on every website that we have hosted (and are currently hosting).

```
CREATE TABLE Websites (
  WID CHAR(5) NOT NULL,
  BID CHAR(5) NOT NULL REFERENCES Business(BID),
  RID CHAR(5) NOT NULL REFERENCES RatesPerQuarter(RID),
  AddressURL CHAR(70) NOT NULL,
  Name CHAR(55) NOT NULL,
  UNIQUE(AddressURL),
  PRIMARY KEY (WID)
);
```

WID -> BID, RID, AddressURL, Name

WID	BID	RID	AddressURL	Name
w0001	b0001	r0001	CoolCatz.com	CoolCatz
w0002	b0001	r0001	UglyDogs.com	Ugly Dogs
w0003	b0002	r0002	SOEA.com	SOE Americia
w0004	b0003	r0001	JenidMarkets.com	Jenid Markets
w0005	b0003	r0003	JenidBlackMarkets.onion	Jenid Black Markets
w0006	b0004	r0001	MarcRentals.com	Marcs Rentals
w0007	b0005	r0001	AmericanSocietyElectricalEngineering.edu	American Society Electrical Engineering

ActiveUsersPerQuarter

Contains information about every websites active users. How many as well as where most users are from.

```
CREATE TABLE ActiveUsersPerQuarter(
  WID CHAR(5) NOT NULL REFERENCES Websites(WID),
  Quarter SMALLINT NOT NULL,
  Year SMALLINT NOT NULL,
  NearestDatacenterAID CHAR(5) NOT NULL REFERENCES Address(AID),
  Users INTEGER NOT NULL,
  FOREIGN KEY (Quarter, Year) REFERENCES Quarters(Quarter, Year),
  PRIMARY KEY (WID, Quarter, Year)
);
```

WID, Quarter, Year->NearistDatacenterAID, Users

WID	Quarter	Year	NearestDatacenterAID	Users
w0001	1	2016	a0011	750
w0001	2	2016	a0012	583
w0001	3	2016	a0011	1067
w0001	4	2016	a0011	1007
w0001	1	2017	a0011	1381
w0002	3	2016	a0011	305
w0002	4	2016	a0013	248
w0002	1	2017	a0012	638
w0003	3	2015	a0015	173
w0003	4	2015	a0015	1002
w0003	1	2016	a0015	1356

WID	Quarter	Year	NearestDatacenterAID	Users
w0003	2	2016	a0012	1482
w0003	3	2016	a0015	2038
w0003	4	2016	a0014	2344
w0003	1	2017	a0015	2612
w0004	1	2016	a0012	243
w0004	2	2016	a0012	214
w0004	3	2016	a0013	148
w0004	4	2016	a0012	342
w0004	1	2017	a0011	178
w0005	1	2016	a0013	2043
w0005	2	2016	a0014	2531

WID	Quarter	Year	NearestDatacenterAID	Users
w0005	4	2016	a0012	3521
w0005	1	2017	a0015	3957
w0006	3	2015	a0013	213
w0006	4	2015	a0013	423
w0006	1	2016	a0013	311
w0006	2	2016	a0013	184
w0007	3	2016	a0015	1384
w0007	4	2016	a0015	762
w0007	1	2017	a0015	926

Servers

Contains information on every server as well as where it is stored.

```
CREATE TABLE Servers (
  SID CHAR(5) NOT NULL,
  DSID CHAR(5) NOT NULL REFERENCES Datacenters(DSID),
  Model CHAR(45) NOT NULL,
  OS CHAR(35) NOT NULL,
  PRIMARY KEY (SID)
);
```

SID ->DSID, Model, OS

SID	DSID	Model	OS
s0001	ds001	HP ProLiant DL360p Gen8	Windows Server 2003
s0002	ds001	HP ProLiant ML115 G5 Server	Red Hat Enterprise
s0003	ds003	HP ProLiant DL360p Gen8	Ubuntu
s0004	ds003	HP ProLiant DL360 G6	Windows Server 2003
s0005	ds004	HPE BladeSystem	Ubuntu
s0006	ds005	Dell PowerEdge T620	Red Hat Enterprise
s0007	ds001	HP ProLiant DL360p Gen8	Windows Server 2008
s0008	ds002	HPE BladeSystem	Ubuntu
s0009	ds007	Plex	Windows Server 2008
s0010	ds003	HPE BladeSystem	Red Hat Enterprise
s0011	ds002	Serviio	Windows Server 2012
s0012	ds006	HP ProLiant DL360p Gen8	Ubuntu

PaymentsPerQuarter

Contains a record of every payment a business has made for our services

```
CREATE TABLE PaymentsPerQuarter (
  BID CHAR(5) NOT NULL REFERENCES Business(BID),
  Quarter SMALLINT NOT NULL,
  Year SMALLINT NOT NULL,
  Cost NUMERIC(10,2) NOT NULL,
  Payment NUMERIC(10,2),
  FOREIGN KEY (Quarter, Year) REFERENCES Quarters(Quarter, Year),
  PRIMARY KEY (BID, Quarter, Year)
);
```

BID, Quarter, Year -> Cost, Payment

BID	Quarter	Year	Cost	Payment
b0001	1	2016	5000	5000
b0001	2	2016	5000	5000
b0001	3	2016	10000	10000
b0001	4	2016	10000	10000
b0001	1	2017	10000	10000
b0002	3	2015	5000	5000
b0002	4	2015	5000	5000
b0002	1	2016	7500	7500
b0002	2	2016	7500	7500
b0002	3	2016	7500	7500
b0002	4	2016	7500	7500
b0002	1	2017	7500	7500

BID	Quarter	Year	Cost	Payment
b0003	1	2016	17500	17500
b0003	2	2016	17500	17500
b0003	3	2016	17500	17500
b0003	4	2016	17500	17500
b0003	1	2017	17500	17500
b0004	3	2015	5000	5000
b0004	4	2015	5000	5000
b0004	1	2016	5000	5000
b0004	2	2016	5000	5000
b0005	3	2016	5000	5000
b0005	4	2016	5000	5000
b0005	1	2017	5000	<NULL>

Hosting

Contains where every website is being hosted (if the website is being hosted at all).

```
CREATE TABLE Hosting (
  SID CHAR(5) NOT NULL REFERENCES Servers(SID),
  WID CHAR(5) NOT NULL REFERENCES Websites(WID),
  PRIMARY KEY (SID, WID)
);
```

SID, WID

SID	WID
s0001	w0001
s0003	w0002
s0005	w0003
s0008	w0003
s0012	w0003
s0004	w0004
s0001	w0005
s0005	w0005
s0007	w0007
s0001	w0001

Backups

Contains which servers are backing up other servers

```
CREATE TABLE Backups (
  SID CHAR(5) NOT NULL REFERENCES Servers(SID),
  BackupSID CHAR(5) NOT NULL REFERENCES Servers(SID)
);
```

SID, WID

SID	BackupSID
s0001	s0009
s0003	s0006
s0005	s0009
s0008	s0006
s0012	s0002
s0004	s0009
s0001	s0009
s0005	s0010
s0008	s0006
s0007	s0012

Views

BusinessPayments

Shows the total cost and payment of every business.

```
CREATE VIEW BusinessPayments(BID, Name, TotalCost, TotalPayment)
AS
SELECT Business.BID, Business.Name, SUM(PaymentsPerQuarter.Cost) AS TotalCost,
SUM(PaymentsPerQuarter.Payment) AS TotalPayment
FROM Business
INNER JOIN PaymentsPerQuarter on Business.BID = PaymentsPerQuarter.BID
GROUP BY Business.BID;
```

BID	Name	TotalCost	TotalPayment
b0001	CoolCatz	40000	40000
b0003	Jenid Markets	87500	87500
b0002	SOE	47500	47500
b0005	American Society Electrical Engineering	15000	10000
b0004	Marc Rentals	20000	20000

ServerLocation

Shows where every server is by its AID and Zip code.

```
CREATE VIEW ServerLocation(SID, AID, ZIP) AS
SELECT S.SID, A.AID, Z.ZIP
FROM Servers S
  INNER JOIN Datacenters D ON S.DSID = D.DSID
  INNER JOIN Address A ON D.AID = A.AID
  INNER JOIN Zip Z ON A.ZIP = Z.ZIP;
```

SID	AID	ZIP
s0001	a0011	1801
s0002	a0011	1801
s0003	a0013	1801
s0004	a0013	1801
s0005	a0014	14701
s0006	a0015	14701
s0007	a0011	1801
s0008	a0012	1801
s0009	a0005	60185
s0010	a0013	1801
s0011	a0012	1801
s0012	a0004	20815

FullAddress

Shows a full address by combining the Address and Zip tables

```
CREATE VIEW FullAddress (AID, StreetNumber, StreetName, ZIP, City, State) AS
SELECT Address.AID, Address.StreetNumber, Address.StreetName, Zip.ZIP,
Zip.City, Zip.State
FROM Address INNER JOIN Zip ON Address.ZIP = Zip.ZIP;
```

AID	StreetNumber	Street	Zip	City	State
a0001	635	Lenox Ave.	7090	Westfield	NJ
a0002	3399	North Rd.	12601	Poughkeepsie	NY
a0003	634	Lenox Ave.	7090	Westfield	NJ
a0004	71	Pilgrim, Ave.	20815	Chevy Chase	MD
a0005	44	Shirley Ave.	60185	West Chicago	IL
a0006	123	6th St	32904	Melbourne	FL
a0007	599	Glendale Street	54115	De Pere	WI
a0008	912	Old Talbot Dr.	8865	Phillipsburg	NJ

AID	StreetNumber	Street	Zip	City	State
a0009	410	Brickyard Dr.	11967	Shirley	NY
a0010	545	St Margarets Ave.	19064	Springfield	PA
a0011	160	S. Woodsman Court	1801	Woburn	MA
a0012	12	Main St	1801	Woburn	MA
a0013	15	Main St	1801	Woburn	MA
a0014	797	W. High Point St.	14701	Jamestown	NY
a0015	9187	Green Ave.	14701	Jamestown	NY

AverageUsers

Shows the average users from every website that is being hosted

```
CREATE VIEW AverageUsers(WID, Name, AverageUsers, ExpectedUsers) AS
SELECT W.WID, W.Name, AVG(A.Users), R.ExpectedUsers
FROM Websites W
  INNER JOIN ActiveUsersPerQuarter A ON W.WID = A.WID
  INNER JOIN RatesPerQuarter R ON R.RID = W.RID
  INNER JOIN Hosting H ON H.WID = W.WID
GROUP BY W.WID, R.ExpectedUsers;
```

WID	Name	AverageUsers	ExpectedUsers
w0002	Ugly Dogs	397	1500
w0001	CoolCatz	957.6	1500
w0006	Marcs Rentals	282.75	1500
w0005	Jenid Black Markets	3254.2	4500
w0003	SOE Americia	1572.428571	2500
w0004	Jenid Markets	225	1500
w0007	American Society Electrical Engineering	1024	1500

Stored Procedures

expectedRevenue

This function will return a resultset of the expected revenue. It takes into account every active website, the rate that it is hosted and sums it or each business.

```
CREATE FUNCTION expectedRevenue(ResultSet
REFCURSOR) RETURNS REFCURSOR
AS
$$
DECLARE
  --
BEGIN
  OPEN ResultSet FOR
    SELECT DISTINCT B.BID, SUM(R.Cost) AS
ExpectedRevenue
    FROM RatesPerQuarter R
      INNER JOIN Websites W ON R.RID = W.RID
      INNER JOIN Business B ON W.BID = B.BID
      INNER JOIN Hosting H ON W.WID = H.WID
    GROUP BY B.BID;
  RETURN ResultSet;
END;
$$
LANGUAGE plpgsql;
```

newQuarter

This function will be called anytime the business wants to prepare for the next quarter. It will add a new row to the Quarters table as well as inserting values for PaymentsPerQuarter, and ActiveUsersPerQuarter.

```
CREATE FUNCTION newQuarter()
RETURNS BOOLEAN AS
$$
DECLARE
  nQuarter INT;
  nYear INT;
  cWID TEXT;
  cBID TEXT;
  cCost DOUBLE PRECISION;
BEGIN
```

```
--select most recent quarter--
  SELECT Quarter, Year
  INTO nQuarter, nYear
  FROM Quarters
  ORDER BY Year DESC, Quarter DESC
  LIMIT 1;
  --increment quarter, year--
  IF nQuarter = 4
    THEN
      nYear := nYear + 1;
      nQuarter := 1;
    ELSE
      nQuarter := nQuarter + 1;
  END IF;
  --Create new quarter--
  INSERT INTO Quarters(Quarter, Year)
  VALUES
  (nQuarter, nYear);
```

newQuarter (cont.)

```
  --select from active websites--
  FOR cWID IN (
    SELECT DISTINCT W.WID
    FROM Websites W
      INNER JOIN Hosting H ON W.WID = H.WID
  )
  LOOP
    --insert active websites into
ActiveUsersPerQuarter--
    INSERT INTO ActiveUsersPerQuarter(WID,
Quarter, Year)
    VALUES
    (cWID, nQuarter, nYear);
  END LOOP;
  --select expected costs for every buisness--
  FOR cBID, cCost IN (
    SELECT DISTINCT B.BID, SUM(R.Cost)
    FROM RatesPerQuarter R
      INNER JOIN Websites W ON R.RID = W.RID
      INNER JOIN Business B ON W.BID = B.BID
      INNER JOIN Hosting H ON W.WID = H.WID
    GROUP BY B.BID
  )
  LOOP
    --insert into PaymentsPerQuarter--
    INSERT INTO PaymentsPerQuarter (BID, Quarter, Year,
Cost)
    VALUES
    (cBID, nQuarter, nYear, cCost);
  END LOOP;

  RETURN true;
END;
$$
LANGUAGE PLPGSQL;
```

checkBackupLocation

This function will be called any time Backups changes and ensure that every server is being backed up by a server in a different location.

```
CREATE FUNCTION checkBackupLocation()
RETURNS TRIGGER AS
$$
DECLARE
   mainZip   TEXT;
   backupZip TEXT;
BEGIN
  SELECT zMain.ZIP,
         zBack.ZIP
  INTO mainZip, backupZip
  FROM Backups INNER JOIN Servers sMain
ON sMain.SID = Backups.SID
    INNER JOIN Datacenters dMain ON dMain.DSID =
sMain.DSID
    INNER JOIN Address aMain ON aMain.AID = dMain.AID
    INNER JOIN Zip zMain ON zMain.ZIP = aMain.ZIP
    INNER JOIN Servers sBack ON sBack.SID =
Backups.BackupSID
    INNER JOIN Datacenters dBack ON dBack.DSID =
sBack.DSID
    INNER JOIN Address aBack ON aBack.AID = dBack.AID
    INNER JOIN Zip zBack ON zBack.ZIP = aBack.ZIP;
  IF mainZip = backupZip
    THEN
     RAISE NOTICE 'BackupServer in same ZIP code
location as main server.';
     RETURN NULL;
  END IF;
  RETURN NEW;
END;
$$
LANGUAGE PLPGSQL;
```

Triggers

BackupLocation

These triggers will call checkBackupLocation() anytime Backups changes or any of the tables that its foreign keys depend on change.

```
CREATE TRIGGER backupLocationInsert
AFTER INSERT ON Backups
EXECUTE PROCEDURE checkBackupLocation();

CREATE TRIGGER backupLocationUpdate
AFTER UPDATE ON Backups
EXECUTE PROCEDURE checkBackupLocation();
```

```
CREATE TRIGGER
backupLocationServersInsert
AFTER INSERT ON Servers
EXECUTE PROCEDURE checkBackupLocation();
CREATE TRIGGER
backupLocationServersUpdate
AFTER UPDATE ON Servers
EXECUTE PROCEDURE checkBackupLocation();
CREATE TRIGGER
backupLocationDatacentersInsert
AFTER INSERT ON Datacenters
EXECUTE PROCEDURE checkBackupLocation();
CREATE TRIGGER
backupLocationDatacentersUpdate
AFTER UPDATE ON Datacenters
EXECUTE PROCEDURE checkBackupLocation();
```

Security

CEO

Allows the CEO to access anything

```
CREATE ROLE CEO;
REVOKE ALL ON ALL TABLES TO CEO;
GRANT ALL ON ALL TO CEO;
```

CFO

Allows the CFO to access anything related to finance

```
CREATE ROLE CFO;
REVOKE ALL ON ALL TABLES TO CFO;
GRANT SELECT ON PaymentsPerQuarter TO CFO;
GRANT INSERT, UPDATE, DELETE, SELECT, ALTER ON Websites TO CFO;
GRANT SELECT ON Business TO CFO;
GRANT INSERT, UPDATE, DELETE, SELECT, ALTER ON Employees TO CFO;
GRANT INSERT, UPDATE, DELETE, SELECT, ALTER ON RatesPerQuarter TO CFO;
```

CTO

Allows the CTO to access anything related to information technology

```
CREATE ROLE CTO;
REVOKE ALL ON ALL TO CTO;
GRANT INSERT, UPDATE, DELETE, SELECT, ALTER ON Backups TO CTO;
GRANT INSERT, UPDATE, DELETE, SELECT, ALTER ON Hosting TO CTO;
GRANT INSERT, UPDATE, DELETE, SELECT, ALTER ON Servers TO CTO;
GRANT SELECT ON ActiveUsersPerQuarter TO CTO;
GRANT UPDATE, SELECT, ON Websites TO CTO;
GRANT SELECT ON Business TO CTO;
GRANT INSERT, UPDATE, DELETE, SELECT, ALTER ON Datacenters TO CTO;
```

CMO

Allows the CMO to access anything related to marketing

```
CREATE ROLE CMO;
REVOKE ALL ON ALL TABLES TO CMO;
GRANT SELECT ON PaymentsPerQuarter TO CMO;
GRANT SELECT ON ActiveUsersPerQuarter TO CMO;
GRANT SELECT ON Websites TO CMO;
GRANT SELECT ON Business TO CMO;
GRANT SELECT ON Datacenters TO CMO;
GRANT INSERT, UPDATE, DELETE, SELECT, ALTER ON RatesPerQuarter TO CMO;
```

Future Plans

Implementation Notes

This database focus on a hypothetical web-hosting business Cumulus. It stores data on client businesses and the websites that they are hosting. It was only built to handle local business and can not have any addresses abroad.

Known Problems

The backup checks are not recursive, so a main server can backup a server that it is backing up.

The only way to tell if a website is active is to look in the hosting table. It might be better to have a table for inactive websites.

Future enhancements

Datacenters and servers should have more networking information so there can be an estimate of how many users a server can serve.

The websites table only stores the front page of the website. It would be nice to have the full tree of each website as well as how large each page is.