Bringing Cloud Databases On-Premises with Greenplum and Kubernetes

November 19, 2019
Choon Sam Park
Director, Data Business Korea/Japan
Part 1: Why

AI and Ease of Use
“Software Ate The World, Now AI Is Eating Software”
AI is eating software – AI at the core of new startups

BenchSci blog: since November 2017, listed 158 startups using machine learning to research and develop drugs

AI-powered code generation tools like TabNine, TypeSQL and BAYOU

Getting answers to any question about your medical data, from natural language to AI generated SQL (Question-to-SQL)

Source: https://www.forbes.com/sites/cognitiveworld/2019/08/29/software-ate-the-world-now-ai-is-eating-software/#388361a85810
Customers Bring AI to their core business w/ Greenplum
Greenplum: Integrated Analytic Platform

- Started Greenplum in California since 2003
- Acquired by EMC in 2010
  - Launched the Data Computing Appliance
- Pivotal Created in 2013
  - Greenplum, Gemfire, Pivotal Labs, Spring, Cloud Foundry
- Greenplum is Opensource based Analytic Platform
  - Distribution Architecture, ML/AI, Anywhere, K8s
Greenplum = Massively Parallel Postgres for Analytics

World’s #1 Open Source Data Warehouse

Critical Capabilities Use-Case Graphics

Product or Service Scores for Traditional Data Warehouse

- Teradata
- Oracle (Oracle Exadata)
- Pivotal (Pivotal Greenplum)
- SAP (SAP HANA)
- Google (BigQuery)
- Micro Focus (Vertica)
- Gbase (GBase 8X)
- IBM (Db2)
- Snowflake
- Amazon Web Services (Amazon Redshift)
- Microsoft (Azure SQL Data Warehouse)
- Alibaba Cloud (MaxCompute)
- Huawei (FusionInsight Big Data)
- MarkLogic
- MapR Technologies (MapR Data Platform)
- Hortonworks (Hortonworks Data Platform)
- Cloudera (Cloudera Enterprise)
- Arm Treasure Data
- Neo4j

As of 31 January 2019
Source: Gartner (March 2019)
Greenplum = A massively parallel Postgres for AI
1. Greenplum is embedded in containers for portability and dependency management

1. Each container is managed by Kubernetes for higher availability & elasticity

1. Kubernetes operator is used for automation

1. PKS for multi-cloud and day-2 operations with full-stack support
Part 2: Scenario

A day in the Life of a Data Scientist

This is (almost) a real scenario

https://github.com/pnagula/Greenplum-Super-Query
Where is Captain America?
1. I need an AI Platform
My friendly Ops Team has done some “One-Time Setup” for me.

K8s Cluster Ready
Operator Ready

- Downloaded Greenplum for K8s
- Uploaded images to registry
- Created K8s cluster
- Deployed Greenplum Operator
- Prepared instance manifest
  - add extensions
  - adjust storage, sizing, etc.
I get to decide what options to use!

For Best Performance:
- Backed by a local SSD
- XFS filesystem, using readahead cache

- Only 2 Segments to get started
- 1 GB each because we are in Dev.

- No Mirrors we are in Dev.
- AntiAffinity turned off with no mirrors
Same Command

- Initialize Greenplum Workbench
- Update Configuration
- Upgrade Minor Versions
- Apply Patches

Kubectl apply -f my-gp.yaml

Options installed automatically
### Status and Age

<table>
<thead>
<tr>
<th>NAME</th>
<th>STATUS</th>
<th>AGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>greenplumcluster.greenplum.pivotal.io/my-greenplum</td>
<td>Running</td>
<td>94s</td>
</tr>
<tr>
<td>greenplumtextservice.greenplum.pivotal.io/my-greenplum-gptext</td>
<td></td>
<td>93s</td>
</tr>
<tr>
<td>greenplumpxfservice.greenplum.pivotal.io/my-greenplum-pxf</td>
<td></td>
<td>94s</td>
</tr>
<tr>
<td>pod/greenplum-operator-7fbffdcf64-w6vzw</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td>pod/master-0</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td>pod/master-1</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td>pod/my-greenplum-gptext-solr-0</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td>pod/my-greenplum-gptext-zookeeper-0</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td>pod/my-greenplum-gptext-zookeeper-1</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td>pod/my-greenplum-gptext-zookeeper-2</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td>pod/my-greenplum-pxf-d5489784b-rhghts</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td>pod/my-greenplum-pxf-d5489784b-sst9n</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td>pod/segment-a-0</td>
<td>1/1</td>
<td>Running</td>
</tr>
<tr>
<td>pod/segment-a-1</td>
<td>1/1</td>
<td>Running</td>
</tr>
</tbody>
</table>
2. I need to load up some data
Greenplum can access it all.
3. I need to run a complex query
User question

“Find anyone whose names sound like ‘Steve’ or “Peggy” and who were at WW2 and knows each other directly and have withdrawn at least $20 after 1945 less than 20 KM from a reference latitude and longitude (Peggy’s parents)”
An interesting Challenge!!!

Language Analytics
Are these the same words?

Graph Analytics
Who do they know?

Time Series
When did it happen?

Geospatial Analytics
Where are they?
We have Legacy Data Lake/Swamp

Using a Hadoop Ecosystem: 10 steps, 3000+ Lines of code across 4 different systems

1. LOAD customer data from HDFS and put to HIVE
2. DESCRIPTION Column needs to be indexed
3. SEARCH IN Column & WRITE Result to HDFS
4. WRITE CODE: Pulling Data Into Spark Data Frame
5. WRITE CODE: CHECK Soundex
6. WRITE CODE: MATCH SOLR Result
7. WRITE CODE: GRAPH LINK Analysis
8. WRITE CODE: POSTGIS Distance Calculation
9. WRITE CODE: GRAPH LINK Analysis
10. WRITE CODE: WRITE RESULTS TO HIVE TABLE

32 Lines one query!!!
“Find anyone whose names sound like ‘Steve’ or ‘Peggy’ and who were at WW2. Knows each other directly.”
GPText and Greenplum

Extract and Transform

- Fast text extraction, indexing/search
- Open source analytics with MPP processing
- Index/store metadata only, avoid data ETL
- Search-engine like syntax
- Better matching for more relevant results
- Many sources and formats, w/o complexity

Explore and Analyze

- Part of Speech Detection
- Named Entity Recognition
- Categorization (via MADLib)
- Topic Modeling (via MADLib)
- Classification/Sentiment (via MADlib, Python, R libraries)

Identify language that signals interesting behaviors and events for the use case
I really love my Ops - GPText Installed by Default

- Installed by automatically
- Scale GPText resources independently of GPDB
- Running 3 instances.
"Find anyone whose names sound like ‘Steve’ or ‘Peggy’ and who were at WW2 and knows each other directly and have withdrawn at least $20 after 1945 less than 20 KM from a reference latitude and longitude (Peggy’s parents)"

drop function if exists get_people(text,text,integer,integer,float,float);
CREATE FUNCTION get_people(text,text,integer,integer,float,float) RETURNS integer
AS $$
declare
linkchk integer; v1 record; v2 record;
begin
execute 'truncate table results;';
for v1 in select distinct a.id,a.firstname,a.lastname,amount,tran_date,c.lat,c.lng,address,a.description,d.score from people a,transactions b,location c,
(SELECT w.id, q.score FROM people w, gptext.search(TABLE(SELECT 1 SCATTER BY 1), 'gpadmin.public.people', 'World War 2', null) q
WHERE (q.id::integer) = w.id order by 2 desc) d
where soundex(firstname)=soundex($1) and a.id=b.id and amount > $3 and (extract(epoch from tran_date)
- extract(epoch from now()))/3600 < $4
and st_distance_sphere(st_makepoint($5, $6),st_makepoint(c.lng, c.lat))/1000.0 <= 20.0 and b.locid=c.locid and a.id=d.id
loop
for v2 in select distinct a.id,a.firstname,a.lastname,amount,tran_date,c.lat,c.lng,address,a.description,d.score from people a,transactions b,location c,
(SELECT w.id, q.score FROM people w, gptext.search(TABLE(SELECT 1 SCATTER BY 1), 'gpadmin.public.people', 'Pivotal', null) q
WHERE (q.id::integer) = w.id order by 2 desc) d
where soundex(firstname)=soundex($2) and a.id=b.id and amount > $3 and extract(year from tran_date) > $4
and st_distance_sphere(st_makepoint($5, $6),st_makepoint(c.lng, c.lat))/1000.0 <= 2.0 and b.locid=c.locid and a.id=d.id
loop
execute 'DROP TABLE IF EXISTS out, out_summary;';
execute  'SELECT madlib.graph_bfs(''people'',''id'',''links'',NULL,'||v1.id||',''out'');'  ;
select 1 into linkchk from out where dist=1 and id=v2.id;
if linkchk is not null  then
insert into results values (v2.id,v2.firstname,v2.lastname,v2.amount,v2.tran_date,v2.lat,v2.lng,v2.address,v2.description,v2.score);
end if;
end loop;
end loop;
return 0;
$$ LANGUAGE plpgsql;
-- person1, person 2, amount, year, longtitude, latitude (in question)
select get_people('Steve','Peggy',20, 1945, 37.926868, -78.024902) ;
Graph Analytics - finding networks.

Social Network

Epidemiology

MMO Role-Playing Game

Chemistry

Bank Risk

1st Party Fraud

Gene

Manufacturing

* Grandjean, M. (2016)
* http://www.netminer.com/community
* www.researchgate.net
* https://www.nature.com/articles/
* https://researchgate.net
* www.infoglide.com
* www.cambridge-intelligence.com
* https://blog.trifinance.com
"Find anyone whose names sound like ‘Steve’ or ‘Peggy’ and who were at WW2 and knows each other directly and have withdrawn at least $20 after 1945 less than 20 KM from a reference latitude and longitude (Peggy’s parents)"

```
DROP function if exists get_people(text, text, integer, integer, float, float);
CREATE FUNCTION get_people(text, text, integer, integer, float, float) RETURNS integer
AS $$
declare
linkchk integer; v1 record; v2 record;
begin
execute 'truncate table results;';
for v1 in select distinct a.id, a.firstname, a.lastname, amount, tran_date, c.lat, c.lng, address, a.description, d.score from people a, transactions b, location c,
(SELECT w.id, q.score FROM people w, gptext.search(TABLE(SELECT 1 SCATTER BY 1), 'gpadmin.public.people', 'World War 2', null) q
WHERE (q.id::integer) = w.id order by 2 desc) d
where soundex(firstname)=soundex($1) and a.id=b.id and amount > $3 and (extract(epoch from tran_date) - extract(epoch from now()))/3600 < $4
and st_distance_sphere(st_makepoint($5, $6), st_makepoint(c.lng, c.lat))/1000.0 <= 20.0 and b.locid=c.locid and a.id=d.id
loop
for v2 in select distinct a.id, a.firstname, a.lastname, amount, tran_date, c.lat, c.lng, address, a.description, d.score from people a, transactions b, location c,
(SELECT w.id, q.score FROM people w, gptext.search(TABLE(SELECT 1 SCATTER BY 1), 'gpadmin.public.people', 'Pivotal', null) q
WHERE (q.id::integer) = w.id order by 2 desc) d
where soundex(firstname)=soundex($2) and a.id=b.id and amount > $3 and extract(year from tran_date) > $4
and st_distance_sphere(st_makepoint($5, $6), st_makepoint(c.lng, c.lat))/1000.0 <= 2.0 and b.locid=c.locid and a.id=d.id
loop
execute 'DROP TABLE IF EXISTS out, out_summary;';
execute 'SELECT madlib.graph_bfs(''people'',''id'',''links'',NULL,'||v1.id||',''out'');' ;
select 1 into linkchk from out where dist=1 and id=v2.id;
if linkchk is not null then
insert into results values (v2.id, v2.firstname, v2.lastname, v2.amount, v2.tran_date, v2.lat, v2.lng, v2.address, v2.description, v2.score);
end if;
end loop;
end loop;
end if;
return 0;
end $$ LANGUAGE plpgsql;
-- person1, person2, amount, year, longtitude, latitude (in question)
select get_people('Steve', 'Peggy', 20, 1945, 37.926868, -78.024902);
withdrawn an amount > $20 after 1945
less than 20 KM from a reference latitude
and longitude (Peggy’s parents)
Greenplum is a Geospatial Database
"Find anyone whose names sound like 'Steve' or "Peggy" and who were at WW2 and knows each other directly and have withdrawn at least $20 after 1945 less than 20 KM from a reference latitude and longitude (Peggy’s parents)"

```
drop function if exists get_people(text,text,integer,integer,float,float);
CREATE FUNCTION get_people(text,text,integer,integer,float,float) RETURNS integer
AS $$
declare
linkchk integer; v1 record; v2 record;
begin
execute 'truncate table results;';
for v1 in select distinct a.id,a.firstname,a.lastname,amount,tran_date,c.lat,c.lng,address,a.description,d.score from people a,transactions b,location c,
(SELECT w.id, q.score FROM people w, gptext.search(TABLE(SELECT 1 SCATTER BY 1), 'gpadmin.public.people', 'World War 2', null) q
WHERE (q.id::integer) = w.id order by 2 desc) d
where
soundex(firstname)=soundex($1) and
a.id=b.id and
amount > $3 and
(extract(epoch from tran_date) - extract(epoch from now()))/3600 < $4
and
st_distance_sphere(st_makepoint($5, $6),st_makepoint(c.lng, c.lat))/1000.0 <= 20.0 and
b.locid=c.locid and
a.id=d.id
loop
for v2 in select distinct a.id,a.firstname,a.lastname,amount,tran_date,c.lat,c.lng,address,a.description,d.score from people a,transactions b,location c,
(SELECT w.id, q.score FROM people w, gptext.search(TABLE(SELECT 1 SCATTER BY 1), 'gpadmin.public.people', 'Pivotal', null) q
WHERE (q.id::integer) = w.id order by 2 desc) d
where
soundex(firstname)=soundex($2) and
a.id=b.id and
amount > $3 and
(extract(year from tran_date) > $4)
and
st_distance_sphere(st_makepoint($5, $6),st_makepoint(c.lng, c.lat))/1000.0 <= 20.0 and
b.locid=c.locid and
a.id=d.id
loop
execute 'DROP TABLE IF EXISTS out, out_summary;';
execute  'SELECT
madlib.graph_bfs(
''people'',''id'',''links'',NULL
)||v1.id||',''out'');'  ;
select 1 into linkchk from out where
if linkchk is not null
then
insert into results values (v2.id,v2.firstname,v2.lastname,v2.amount,v2.tran_date,v2.lat,v2.lng,v2.address,v2.description,v2.score);
end if;
end loop;
return 0;
end $$ LANGUAGE plpgsql;

-- person1 , person 2, amount, year, longitude, latitude (in question)
select get_people('Steve','Peggy',20, 1945, 37.926868, -78.024902) ;
```
4. I need seamless Day 2 operations

No manual recovery needed;
Just re-run the query!

Master dies and is recovered in 34s.
Same process applies to segments.

Even if its host dies, the master (or segment) will recover on another host because of compute-storage separation.

If you use remote storage then mirrors are not required for auto-recovery.
If I had to go into Production - Not Today :)
Real Time Scoring For Apache MADlib

Single command to deploy a MADlib trained model from Pivotal Greenplum / Postgres to Docker, PCF or Kubernetes

User Operations

1. ML Training
   - Train models in Pivotal Greenplum using Apache MADlib

2. madlib --deploy
   - Auto-deploy model from Greenplum to Docker, PCF or Kubernetes

3. Docker Build
   - Build docker container with optimized PostgreSQL and MADlib

4. Synchronize Model
   - Extract model and feature components from Pivotal Greenplum and deploy to container

5. Load Model
   - Load model and feature components into containerized PostgreSQL and MADlib engine

6. Deploy ML Engine
   - Deploy docker container as ML REST end point in target PKS environment

$ madlib --deploy