Oracle Database on AWS

Getting the Best out of RDS and EC2

MARIS ELSINS
Lead Database Consultant
Pythian

Maris Elsins

Lead Database Consultant

At Pythian since 2011

Located in Riga, Latvia

Oracle [Apps] DBA since 2005

Speaker at conferences since 2007

@MarisDBA

elsins@pythian.com

ABOUT PYTHIAN

Pythian’s 400+ IT professionals help companies adopt and manage disruptive technologies to better compete

EXPERIENCED

11,800

Systems currently managed by Pythian

GLOBAL

400

Pythian experts in 35 countries

EXPERTS

2

Millennia of experience gathered and shared over 19 years

Oracle Database on AWS

Getting the BEST out of RDS and EC2

It's a nice title, but....
WTH* did you mean here?

* WTH = WTF

What do you expect?


We Do Three Types of Jobs Here...
GOOD, FAST AND CHEAP
You May Choose Any Two!
If It Is Good and Cheap
It Will Not Be Fast.
If It Is Good and Fast
It Will Not Be Cheap.
If It Is Fast and Cheap
It Will Not Be Good.

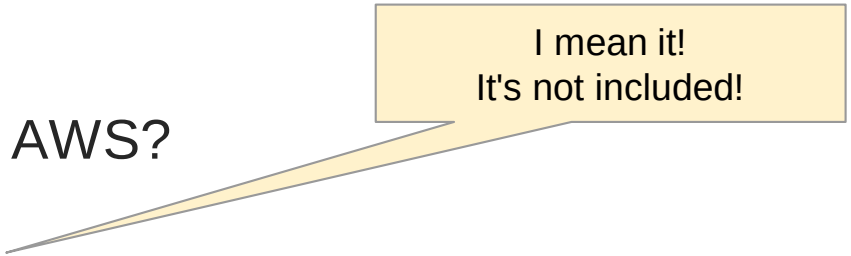

There is no cloud
it's just someone else's computer

Why this Presentation?

- 2 ways to run Oracle Databases on AWS
  - RDS
  - EC2
- Help Choosing between Them

NOT ON AGENDA

- Basics of AWS
- Why cloud and why AWS?
- The right way of ...



Definitions from AWS Documentation

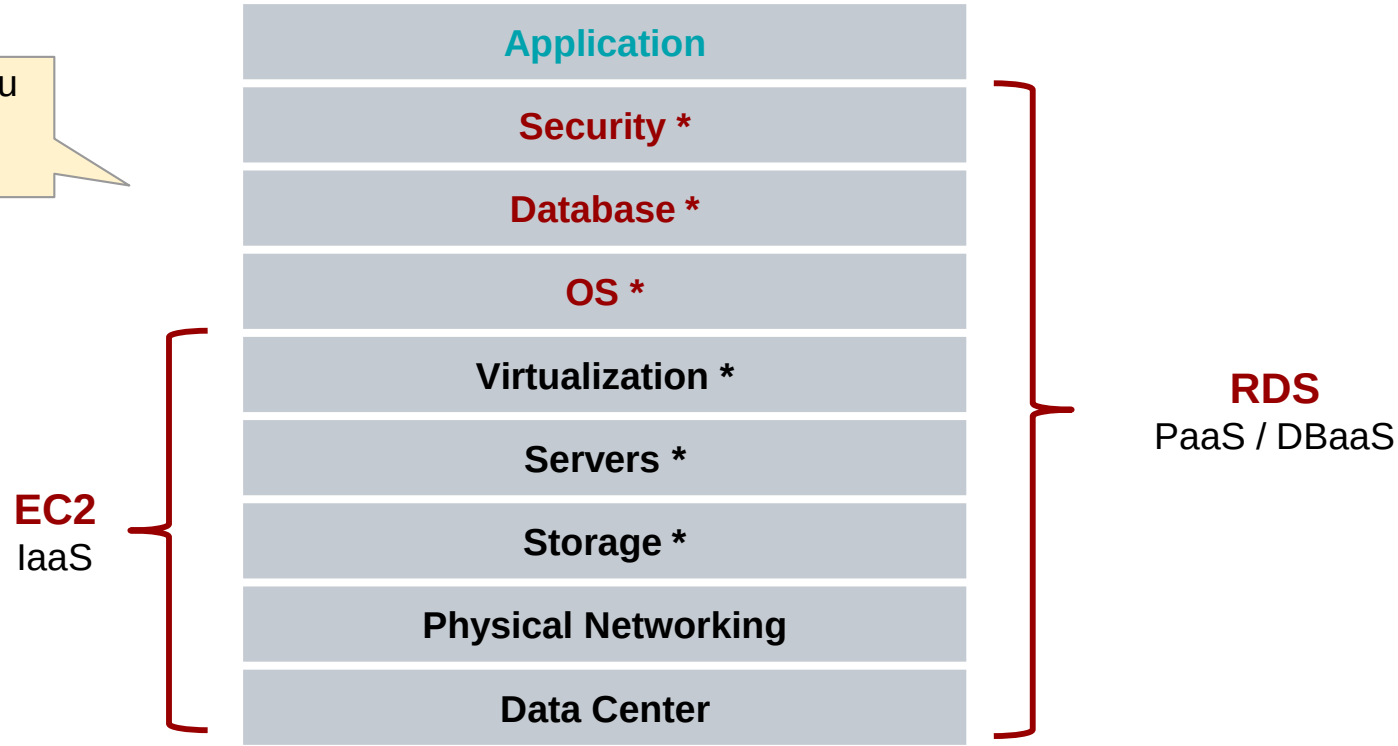
- EC2 (IaaS)
  - Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides secure, resizable compute capacity in the cloud. It is designed to make web-scale cloud computing easier for developers.
- RDS (PaaS / DBaaS)
  - Amazon Relational Database Service (Amazon RDS) is a web service that makes it easier to set up, operate, and scale a relational database in the cloud. It provides cost-efficient, resizable capacity for an industry-standard relational database and manages common database administration tasks.

They don't say they'll do it for you!

... it will just be easier :)

When to choose EC2 and RDS to run Oracle DB?

How much do you want to control these?



For me as a DBA the biggest challenge with RDS is: No host access and no SYSDBA access.

When to choose RDS over EC2 to run Oracle DB?

- RDS is managed by AWS
  - It removes lots of headache
  - Automates routine tasks
    - Backup/Recovery
    - Multi-AZ (Failover, Switchover, DR*)
    - Patching and Upgrades
    - Host Maintenance
    - CloudWatch Monitoring and Alerting
  - But be prepared for
    - limitations
    - specific way of operation

The process is very well tested and is reliable. It's done thousands of times

SERVERS / VIRTUALIZATION

- EC2 gives you more instance types to choose from!
- Having more options is sometimes important:
  - no RAC on AWS - so vertical scaling options are important
    - (3rd party software exists to allow this)
  - Choose a better balance/cost between:
    - CPU capacity
    - Memory
    - Instance Store SSD*

SERVERS / VIRTUALIZATION

WARNING:
Obsolete slide

- Current generation instances for EC2 (74):
  - t2.nano, t2.micro, t2.small, t2.medium, t2.large, t2.xlarge, t2[illegible], m4.2xlarge, m4.4xlarge, m4.10xlarge, m4.16xlarge, m3.medium, m3.lar[illegible].large, c5.xlarge, c5.2xlarge, c5.4xlarge, c5.9xlarge, c5.18xlarge, c4.large, c4.xla[illegible].2xlarge, c4.4xlarge, c4.8xlarge, c3.large, c3.xlarge, c3.2xlarge, c3.4xlarge, c3.8xlarge, f1.2xla[illegible], f1.16xlarge, g3.4xlarge, g3.8xlarge, g3.16xlarge, g2.2xlarge, g2.8xlarge, p2.xlarge, p2.8xlarge, [illegible].16xlarge, p3.2xlarge, p3.8xlarge, p3.16xlarge, r4.large, r4.xlarge, r4.2xlarge, r4.4xlarge, r4.[illegible]xlarge, r4.16xlarge, r3.large, r3.xlarge, r3.2xlarge, r3.4xlarge, r3.8xlarge, x1.16xlarge, **x1e.32xlarge**, **x1.32xlarge**, d2.xlarge, d2.2xlarge, d2.4xlarge, d2.8xlarge, **i2.xlarge, i2.2xlarge, i2.4xlarge, i2.8xlarge, i3.large, i3.xlarge, i3.2xlarge, i3.4xlarge, i3.8xlarge, i3.16xlarge**

X1e Memory Optimized
128 vCPU / 4T RAM

I3 instances = High I/O instances with NVMe Instance Store

- Current generation instances for RDS (23):
  - db.m4.10xlarge, db.m4.16xlarge, db.m4.2xlarge, db.m4.4xlarge, db.m4.large, db.m4.xlarge, db.r3.2xlarge, db.r3.4xlarge, db.r3.8xlarge, db.r3.large, db.r3.xlarge, **db.r4.16xlarge**, db.r4.2xlarge, db.r4.4xlarge, db.r4.8xlarge, db.r4.large, db.r4.xlarge, db.t2.2xlarge, db.t2.[illegible]arge, db.t2.medium, db.t2.micro, db.t2.small, db.t2.xlarge

Up to 64 vCPU &
488 GB RAM

OS

- The OS on the EC2 is:
  - **My**-kind-of-linux (OL / RHEL / ....)
  - This is really important in some situations!
  - Access to the OS is useful
    - install the tools you need (Oswatcher/TFA)
    - Obtain diagnostic info (i.e. Incident trace file from ADR)

- The OS on the RDS is:
  - Some-kind-of-linux

```
SQL> select dbms_utility.port_string from dual;

PORT_STRING
-------------------
x86_64/Linux 2.4.xx
```

but what kind of linux?

  - Doesn't really matter - you don't have access to the OS

STORAGE

- EBS volumes used in both cases:
  - SSD:
    - Provisioned IOPS SSD (the expensive SSD)
    - General Purpose SSD (the cheap SSD)
  - HDD (unsuitable in most cases)
    - RDS: Magnetic (not recommended by AWS)
    - EC2: Throughput Optimized HDD
    - EC2: Cold HDD

- RDS vs EC2
  - EC2 - you set it up how you want it.
    - i.e. ASM with multiple EBS volumes..
  - RDS sets up the specified type/size storage for you

It appears to be a single EBS volume behind it.

(might change over time, or depending on size of the instance)

DATABASE

- EC2 - You install what you need and how you like it

- RDS - Predefined engine versions
  - 11gR2:
    - 11.2.0.4.v1 - 11.2.0.4.v17 - ...
  - 12cR1:
    - 12.1.0.1.v1 - 12.1.0.1.v6 (discontinued)
    - 12.1.0.2.v1 - 12.1.0.2.v13 - …
  - 12cR2: not available yet on RDS (GA March 2017 on prem)
  - Engine versions differ by included patches
    - PSU / DST / GG / + critical one-offs
  - Old engines become unavailable

What if I need a specific one-off?

Options and Features

- Features not available on RDS (12c)
  - ASM
  - Database Vault
  - Java Support
  - Multitenant Database
  - Real Application Clusters (RAC)
  - Unified Auditing
  - Data Guard / Active Data Guard

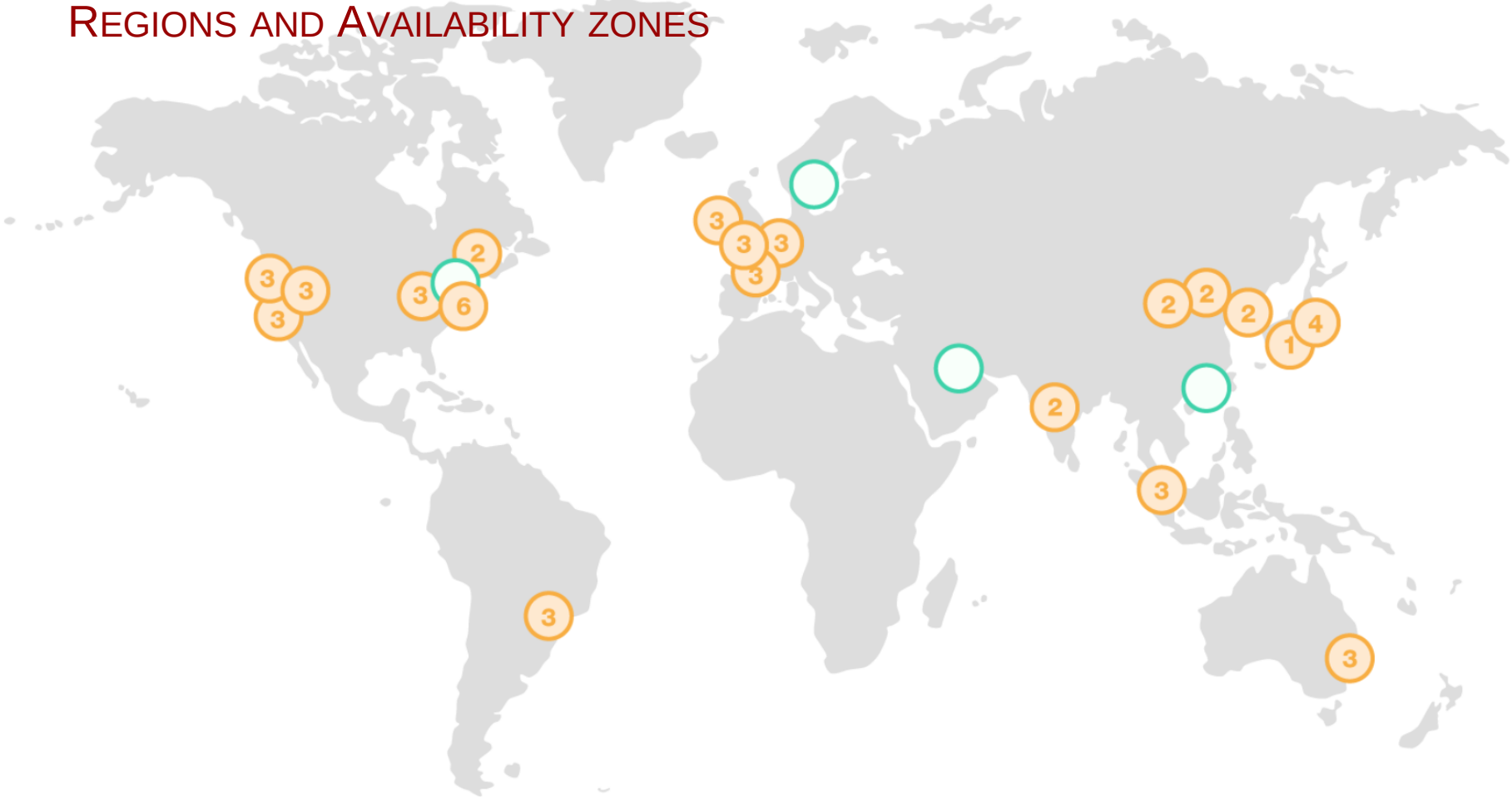
RAC is not supported on EC2 too

Hey, but what about the Disaster Recovery?

Global Infrastructure

Data Centers

Regions and Availability zones



DR BETWEEN REGIONS

- EC2
  - You're responsible for the DR...
  - Use Data Guard if you like
  - Logical replication...
- RDS
  - Multi-AZ
  - How to failover from one region to another?
  - Copy snapshots? What about archived logs?
  - Logical replication (DMS or GoldenGate)?

February 28, 2017
Complete S3 outage in US-EAST-1 Region
The Whole Region was affected

Multi-AZ is a synchronous physical replication between 2 **AZs** in the same region.

Didn't we just loose all the simplicity?

Running and Monitoring RDS...
Life Without "SYSDBA"

- Only SQL*Net Connectivity
- Master User
  - Limited set of privileges
- Common DBA Tasks for Oracle DB Instances
  - Review them carefully, and prepare for action!
  - Some tasks are done differently ...
    - Killing a session
      - exec RDSADMIN.RDSADMIN_UTIL.KILL(..., ..., ...);
    - Creating an AWR report (with no access to the OH)
      - DBMS_WORKLOAD_REPOSITORY.AWR_REPORT_TEXT
  - rdsadmin.rdsadmin_util

http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Appendix.Oracle.CommonDBATasks.html

Running and Monitoring RDS...

Life Without Host Access

- No host access
  - Prevents setting up some types of monitoring
    - Cron jobs (everyone's favorite!)
    - Certain 3rd party monitoring agents
- What's available?
  - Cloudwatch monitoring + alerts
    - 1 minute interval / difficult to correlate different metrics
  - Enhanced monitoring (no alerts)
    - 1s - 60s monitoring interval
  - OEM Option (11g DB Control / Database Expres 12c)
  - OEM_AGENT Option !! (Announced Sep, 2016), supports 13c too!
  - "Performance Insights" for Oracle RDS
  - Custom monitoring over SQL*Net (Pythian's Avail)

Not GA yet, but you already can sign up for the preview

http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Appendix.Oracle.CommonDBATasks.html

Best way of learning is
learning from someone else's mistakes!

CASE 1
OPTION GROUPS AND PARAMETER GROUPS

- Option group
  - Defines the set of enabled options
    - Epex, OEM, STATSPACK, ...
  - can be assigned to multiple RDS instances
- Default options groups:
  - “default:oracle-ee-11-2” and “default:oracle-ee-12-1”
  - has no options enabled by default
  - Can't be modified

http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Appendix.Oracle.Options.html

CASE 1
PARAMETER GROUPS

- Remember,
  - you can't “/ as sysdba” and “alter system …”
- Parameter Groups
  - Defines the init parameters
  - Can be assigned to multiple instances
  - Some parameters are derived from instance settings
    - i.e. DB_NAME={DbName}
- Default parameter groups can't be changed
  - Even modifiable settings require reboot.
  - We used a workaround in few cases - a logon trigger

http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Appendix.Oracle.Options.html

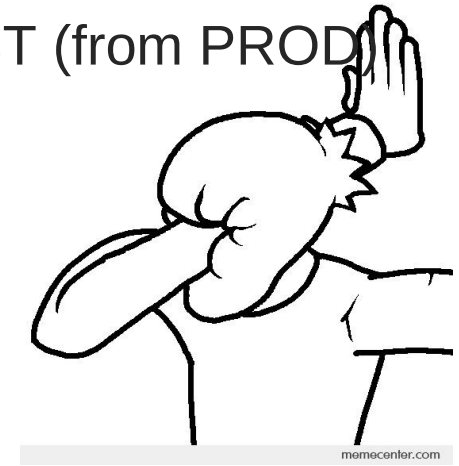
CASE 1
ORA-01555

Ouch !

- UNDO_RETENTION not set (the default is 900 seconds)
- We're using parameter group “default.oracle-ee-11.2”

- The client asked to refresh one schema in TEST (from PROD)
  - “expdp … consistent=y”

```
ORA-31693: Table data object "XYZ"."BLAH_BLAH" failed to
load/unload and is being skipped due to error:
ORA-02354: error in exporting/importing data
ORA-01555: snapshot too old: rollback segment number 82 with
name "_SYSSMU82_540458409$" too small
```



How to fix it?

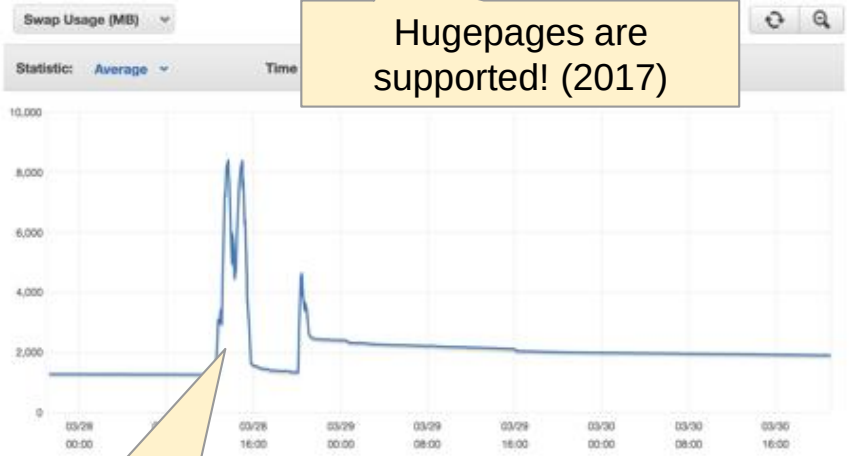
CASE 2
SWAPPING

I prefer ASMM

Ouch !

- MEMORY_TARGET = {DBInstanceClassMemory*3/4}
  - AMM means hugepages are not used
  - Swapping is possible

Hugepages are supported! (2017)



Connection pool misconfiguration

Swap usage MB

Case 3
Latency issue (Not AWS's Fault)

- In the middle of the migration project
  - One database was moved to Oracle RDS
  - Another interfacing system remained on-prem
- A batch job (ETL) runs a LOT SLOWER
  - Tracing reveals row-by-row processing between 2 DBs
  - A local DB <-> AWS RDS
  - Latency issue, each network round-trip took a lot longer.

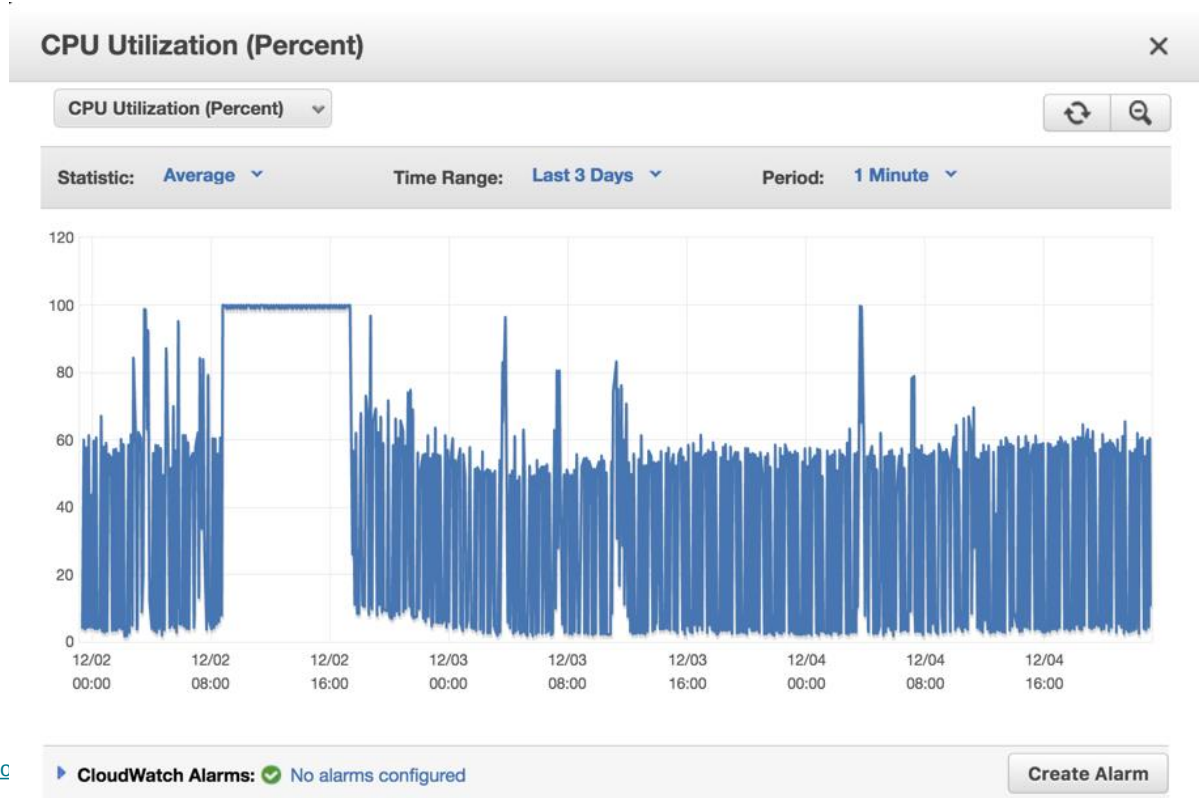
CASE 4
IO BOTTLENECK

- A tiny OBIEE Database on RDS (40G)
  - Db.m4.xlarge
  - 4 vCPU + 16G RAM
- The problem
  - IO slowness
  - Beyond point where IOPS is reasonable to increase to
- The Solution
  - Upgrade to Db.r3.xlarge (memory optimized)
  - 4 vCPU + 30.5 RAM
  - Set parameter _serial_direct_reads=never

CASE 5

PLAN (IN)STABILITY

- A sudden execution plan change for a popular query



https://calculator.s3.amazo

CASE 5
PLAN (IN)STABILITY

- Saturated CPU is difficult to troubleshoot
  - Extremely slow response times
  - Resource Manager can help!
  - Reboot the instance?
- “Plan stability” measures
  - Careful with histograms!
  - SQL Plan Baselines
- Our fix:
  - 1) Resource Manager
  - 2) Customized capture of execution plans in SQL Tuning Sets
  - 3) one-off SQL Plan Baselines are created when needed

CASE 6
COMBINED MAINTENANCE

- Combined maintenance - 30 minutes allocated
  - Take a snapshot
  - Change the instance size

- Oops!
  - Snapshot takes just few seconds
    - … before it's copied to S3
  - “Creating Snapshot” for 45 minutes

https://calculator.s3.amazonaws.com/index.html

Is RDS good for anything?

Is RDS Good For Anything?

RDS is very good if...

- Your want your DB “managed by AWS”
- Your requirements match the RDS' capabilities
- You're aware of the trade-offs and limitations
- You have access to the "Know-how"!
- You test the planned activities

I/O

... how it really behaves.

Sizing the Instance Correctly

It's Not Very Simple...

- Instance size (CPU / RAM) is Important:
  - Size it too big - cost-efficiency reduces
  - Size it too small - performance suffers
  - Relatively simple to size
- Storage size (GB / IOPS) is Important:
  - Size it too big - cost-efficiency reduces
  - Size it too small - performance suffers
  - Extremely difficult to plan and the potential impact is huge
    - Imagine a large table scan

I did some benchmarking to find out more

IO
EBS Volume Types

PIOPS =
Provisioned IO per Second

Magnetic (Previous gen / obsolete)	**General Purpose SSD**	**Provisioned IOPS**
• Spinning disks • Pay per Use + Size • <200 IOPS • 40–90 MiB/s • 1MiB IO size	• SSD • Cheap & Low Latency • 3 IOPS * 1GiB <= 10000 • Pay per Size • Up to 160 MiB/s • Burstable (3000 IOPS)	• SSD • Expensive & Low Latency • 10 IOPS * 1GiB <=20000 • Pay per Size + PIOPS • Up to 500 MiB/s

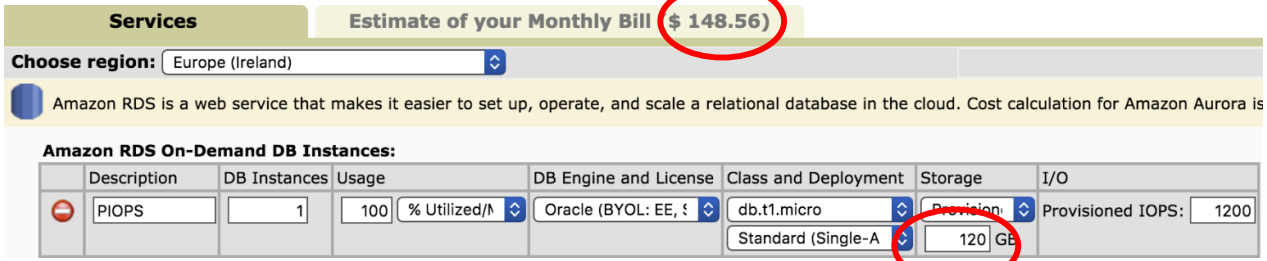
http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/CHAP_Storage.html
http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSVolumeTypes.html

This is super - complicated, read this!

IO Cost

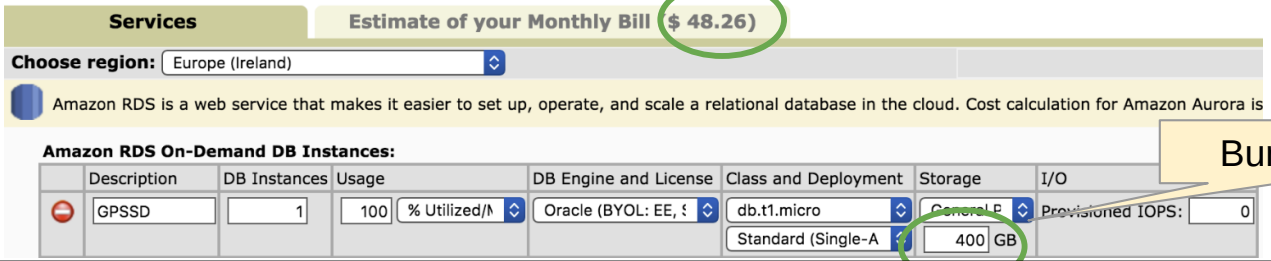
Storage - What about cost of **120G** and **1200** IOPS?

- P-IOPS



- GP-SSD

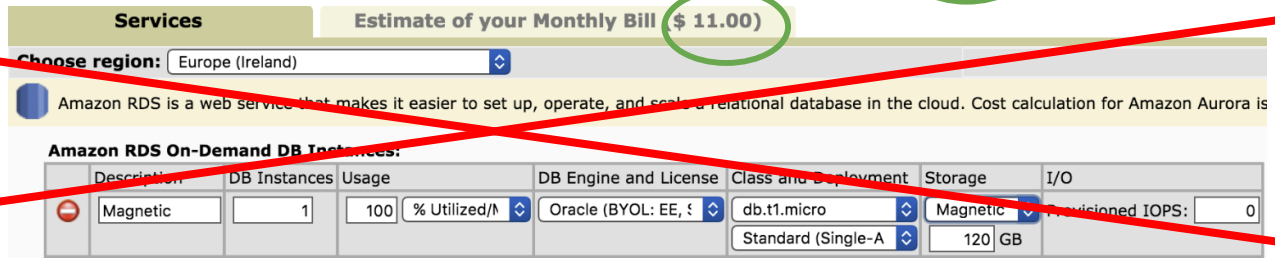
My favourite :)



Burstable!

- Magnetic

No IOPS guarantee / obsolete

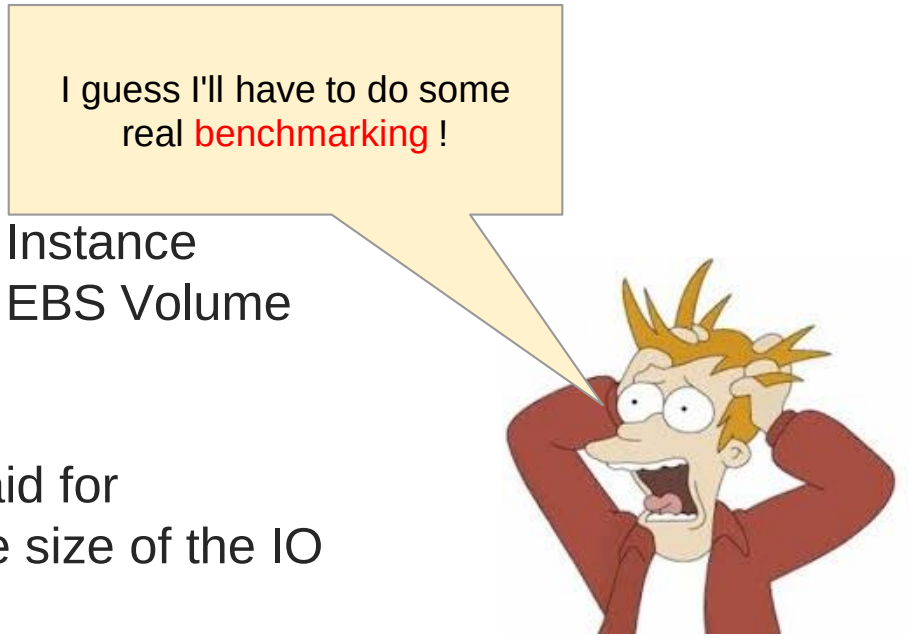


https://calculator.s3.amazonaws.com/index.html

IO

Lots of Limits

- Different throughput/IO limits apply depending on
  - Instance Type
  - Storage Type
  - Storage Size
  - PIOPS
- Limits like:
  - EBS Throughput (Mbps) per Instance
  - EBS Throughput (Mbps) per EBS Volume
  - Max IOPS per Instance
  - Max IOPS per Volume
  - P-IOPS you reserved and paid for
  - IOPS differ depending on the size of the IO



https://calculator.s3.amazonaws.com/index.html

IO

STORAGE TYPES - THE DOCUMENTATION SAYS

*While Provisioned IOPS (**io1** storage) **can work with I/O sizes up to 256 KB**, most databases do not typically use such large I/O. **An I/O request smaller than 32 KB is handled as one I/O**; for example, 1000 16 KB I/O requests are treated the same as 1000 32 KB requests. **I/O requests larger than 32 KB consume more than one I/O request**; Provisioned IOPS consumption is a linear function of I/O request size above 32 KB. For example, a 48 KB I/O request consumes 1.5 I/O requests of storage capacity; a 64 KB I/O request consumes 2 I/O requests, etc. … Note that I/O size does not affect the IOPS values reported by the metrics, which are based solely on the number of I/Os over time. This means that it is possible to consume all of the IOPS provisioned with fewer I/Os than specified if the I/O sizes are larger than 32 KB. For example, a system provisioned for 5,000 IOPS can attain a maximum of 2,500 IOPS with 64 KB I/O or 1,250 IOPS with 128 KB IO.*

… and …

*I/O requests larger than 32 KB are treated as more than one I/O for the purposes of PIOPS capacity consumption. A 40 KB I/O request will consume 1.25 I/Os, a 48 KB request will consume 1.5 I/Os, a 64 KB request will consume 2 I/Os, and so on. **The I/O request is not split into separate I/Os; all I/O requests are presented to the storage device unchanged.** For example, if the database submits a 128 KB I/O request, it goes to the storage device as a single 128 KB I/O request, but it will consume the same amount of PIOPS capacity as four 32 KB I/O requests.*

IO Benchmarking
Interesting Findings

- Documentation is not telling everything
  - *Who from you have made career by writing documentation?*

  Bryn Llewellyn

- 2 Blog posts (http://blog.pythian.com)
  - Investigating IO Performance on Amazon RDS for Oracle
  - Investigating IO Performance on Amazon EC2

IO BENCHMARKING

THE TESTS

- Crated a RDS / EC2 instance with a DB on it
- Created a large table T
- Run script:

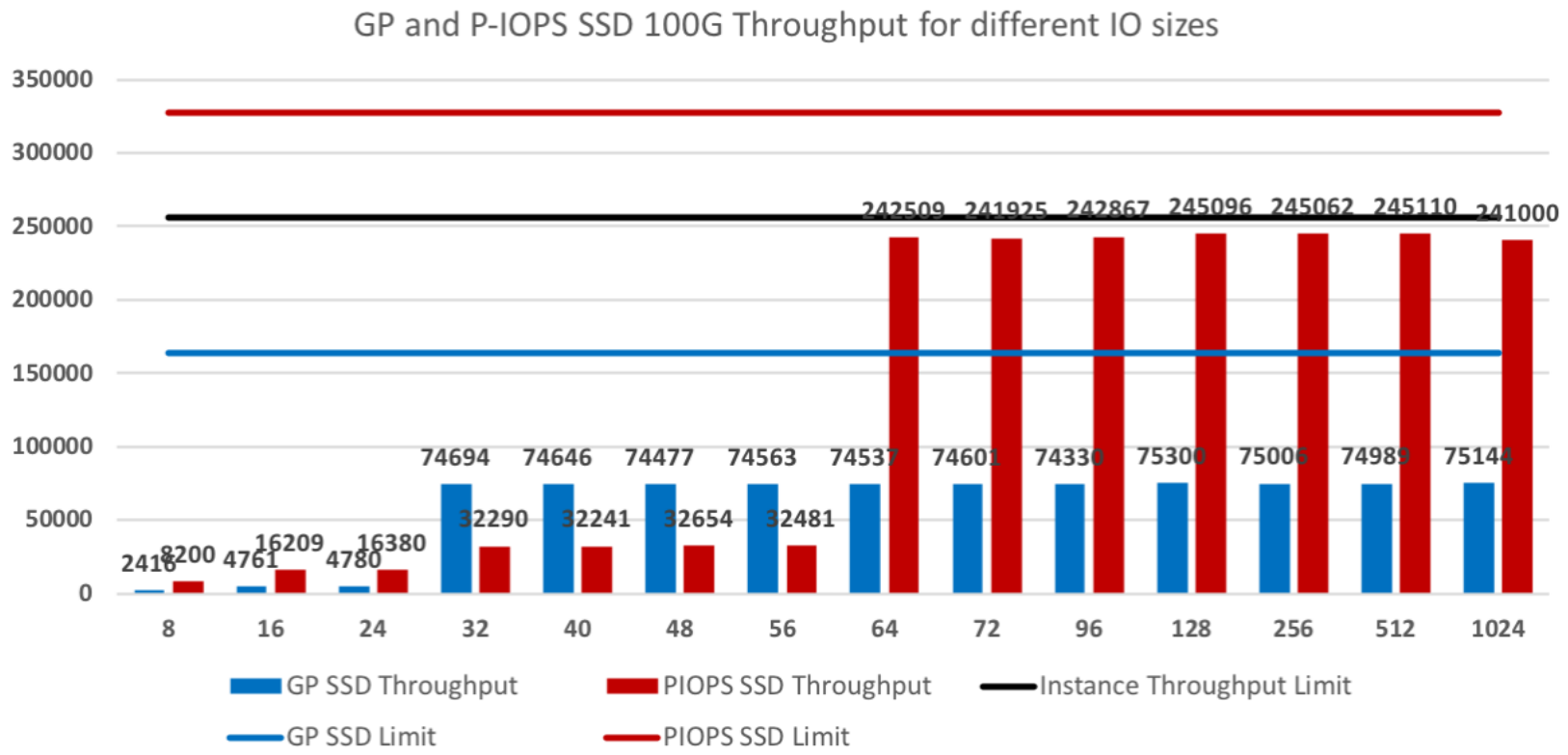
```
exec rdsadmin.rdsadmin_util.flush_buffer_cache;
alter system flush fuffer_cache;
alter session set "_serial_direct_read"=always;
alter session set db_file_multiblock_read_count=&1;

-- Run FTS against T table forever.
declare
  n number:=1;
begin
  while n>0
  loop
    select /*+ full(t) */ count(*) into n from t;
  end loop;
end;
/
```

- Measure IOPS / Read Throughput from CloudWatch or iostat.

IO BENCHMARKING

RDS GP AND P-IOPS THROUGHPUT COMPARISON

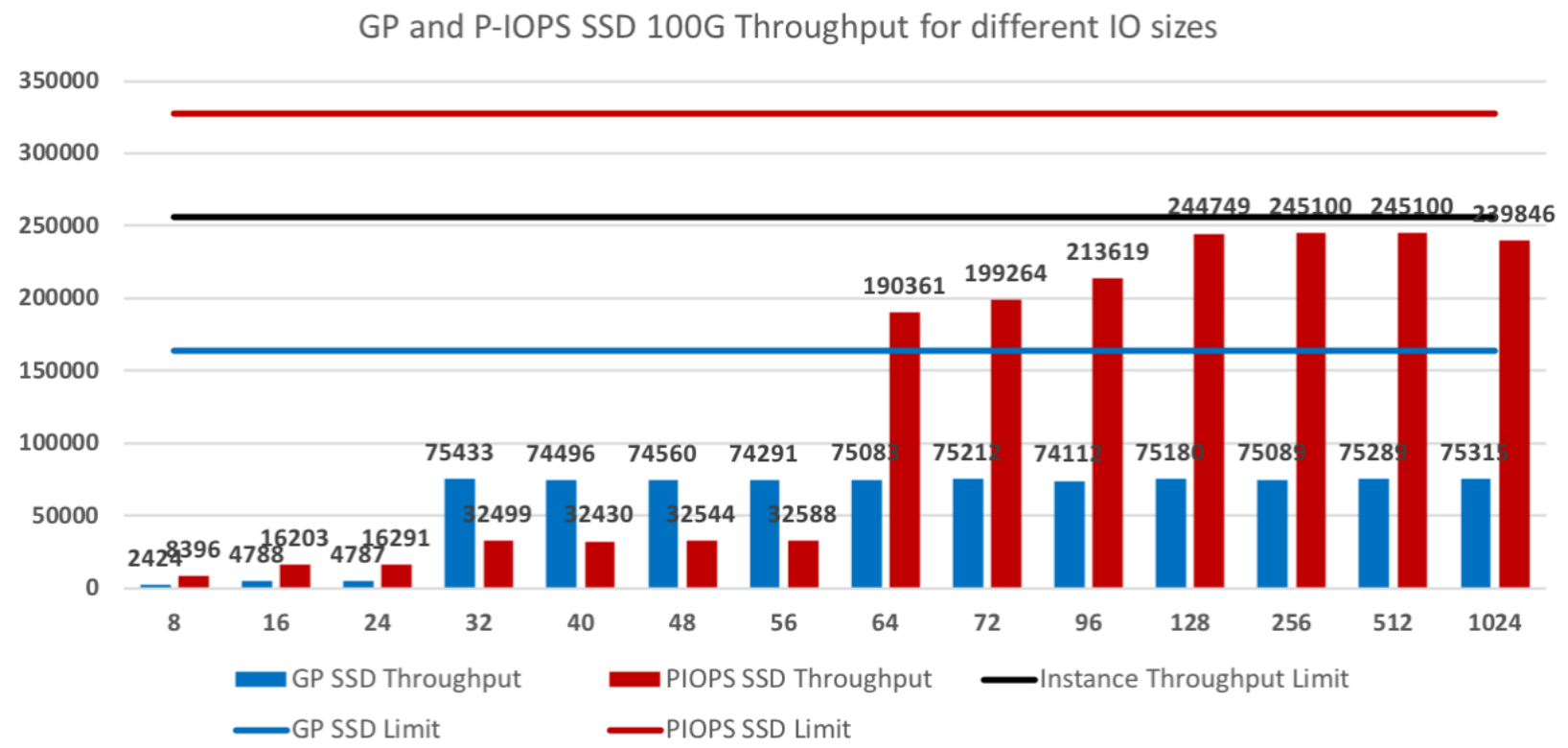
- Db.m4.4xlarge / 100G io1 1000 IOPS vs. 100G gp2 300 IOPS



IO BENCHMARKING

EC2 GP AND P-IOPS THROUGHPUT COMPARISON

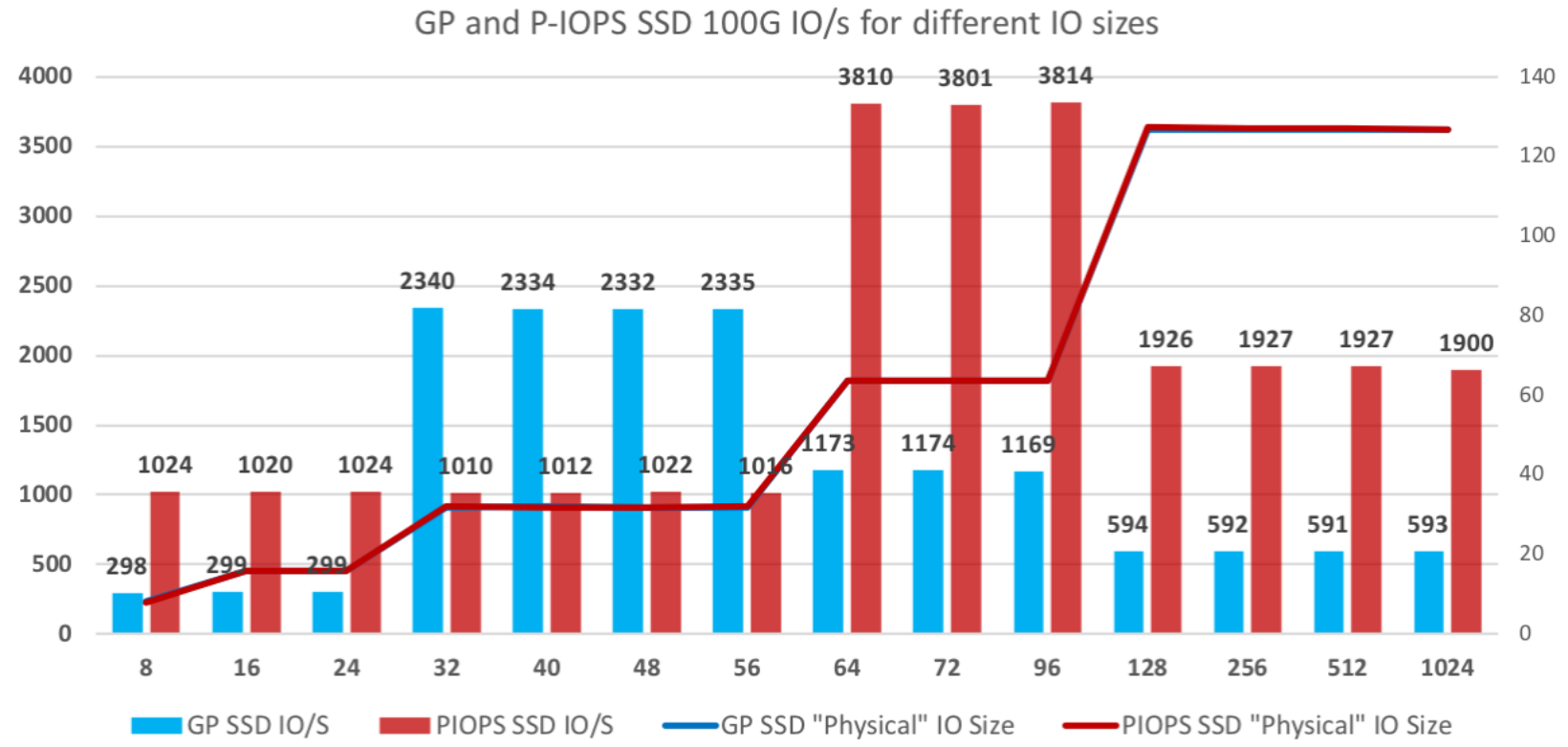
- Db.m4.4xlarge / 100G io1 1000 IOPS vs. 100G gp2 300 IOPS



IO BENCHMARKING

RDS GP AND P-IOPS IO/s COMPARISON

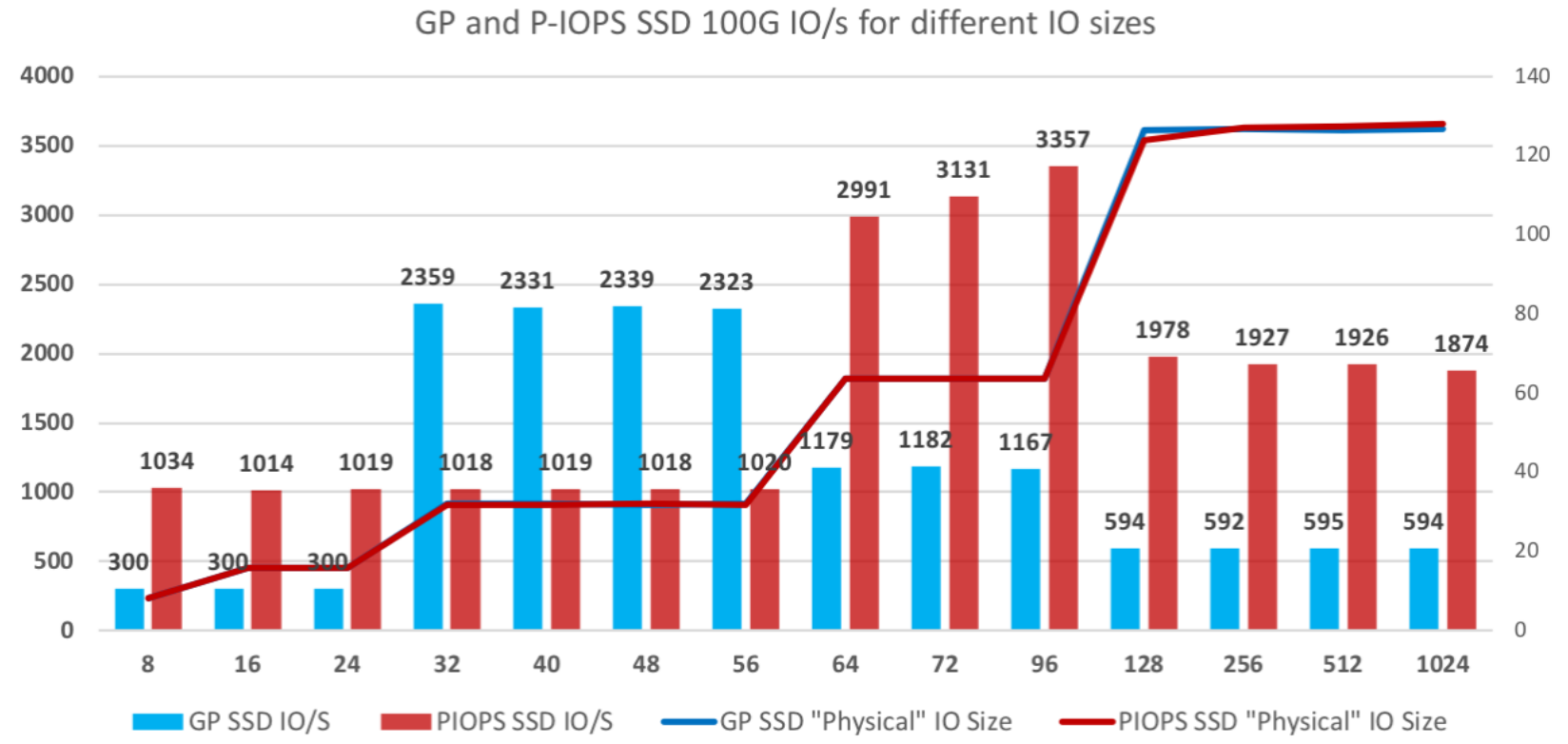
- Db.m4.4xlarge / 100G io1 1000 IOPS vs. 100G gp2 300 IOPS



IO BENCHMARKING

EC2 GP AND P-IOPS IO/s COMPARISON

- Db.m4.4xlarge / 100G io1 1000 IOPS vs. 100G gp2 300 IOPS



IO BENCHMARKING

CONCLUSIONS

- SSD EBS volumes dynamically choose the “accounted” IO size based on the size of the incoming IO request (16K, 32K, 64K, 128K)
- The IOPS limit applies to smaller physical IO sizes
  - General Purpose SSD up to 16K
  - Provisioned IOPS SSD up to 32K
- The Throughput limit applies to larger IO sizes
  - You can get more IOPS than provisioned or the baseline is
  - Unclear how the throughput limit is determined for gp2 when small EBS volumes are used (benchmark more?)
- Bursting applies to IOPS (3000), and throughput (unclear)

IO BENCHMARKING
EC2 FLEXIBILITY

Cost?
~ $0.50 = 5*1G gp2
~ $33.00 = 5G 500IOPS io1 (125 MB/s)

- ASM and 5* 1G (100 IOPS) General Purpose SSD
  - The volume limits apply separately
  - 500 IOPS baseline
  - Up to 800MB/s Throughput (but limited by instance)
- Bursting? Yes!
  - 15000 Max IOPS burst performance!
  - Lasts for ~2000 seconds, or more
  - “Refills” when IO consumption is below baseline

IO BENCHMARKING

EC2 5*GP2 - 8K READS

Volume limits apply to each EBS volume separately

- Bursting performance:
  - **6024 IO/s**
  - 48214 KB/s

```
[root@ip-172-31-21-241 ~]# iostat 5 500
Device:            tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
xvda              0.00         0.00         0.00          0          0
xvdf              3.40         0.00        16.80          0         84
xvdg           1203.40      9632.00         1.60      48160          8
xvdi           1199.60      9596.80         0.00      47984          0
xvdj           1211.60      9691.20         3.20      48456         16
xvdk           1208.60      9670.40         0.00      48352          0
xvdl           1203.00      9625.60         3.20      48128         16
```

IO BENCHMARKING

EC2 5*GP2 - 1M READS

- Bursting performance:
  - **245111 KB/s** (Instance limit)
    - It was impossible to reach this performance with just 1 volume
  - 1925 IO/s

```
[root@ip-172-31-21-241 ~]# iostat 5 500
Device:            tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
xvda              0.00         0.00         0.00          0          0
xvdf              3.40         0.00        16.80          0         84
xvdg            384.40     48820.80         0.80     244104          4
xvdi            385.80     49155.20         0.00     245776          0
xvdj            385.00     49014.40         6.40     245072         32
xvdk            386.80     49225.60         0.00     246128          0
xvdl            385.00     48897.60         6.40     244488         32
```

IO BENCHMARKING

CHEATING THE SYSTEM

- The Best IO config for Oracle DB on AWS?
  - EC2
  - Multiple General Purpose SSD volumes
  - ASM
  - BLOCK_SIZE = 32K
- No “artificial” IOPS limit
- Only instance/volume throughput limits apply.

What if IO performance is still insufficient?

Or you simply want to reduce the DB's IOPS requirement to pay less

Instance Store

What it is?

Nonpersistent storage.
Local SSD.
IOPS are not accounted

Look for **Storage Optimized** instances for NVMe SSD

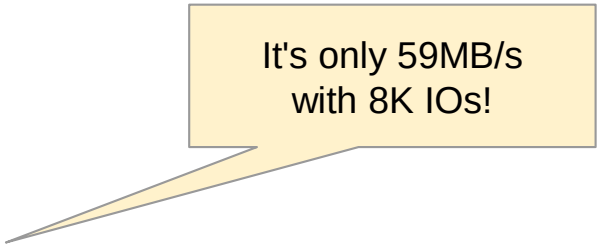
Instance Types Matrix

Instance Type	vCPU	Memory (GiB)	Storage (GB)	Networking Performance	…ssor	Clock Speed (GHz)	Intel AVX†	Intel AVX2†	Int… Turbo	…working†
r4.8xlarge	32	244	-	10 Gi…	Intel Xeon E5-2686 v4	2.3	Yes		Yes	Yes	Yes
r4.16xlarge	64	488	-	…0 Gigabit	Intel Xeon E5-2686 v…	…2.3	Yes	Yes	Yes	Yes	Yes
r3.large	2	15.25	1 x 32 SSD	Mode…te	Intel Xeon E5-2670 v2	2.5	Yes	-	Yes	-	Yes
r3.xlarge	4	30.5	1 x 80 SSD	Moderate	Intel Xeon E5-2670 v2	2.5	Yes	-	Yes	Yes	Yes
r3.2xlarge	8	61	1 x 160 SSD	High	Intel Xeon E5-2670	2.5	Yes	-	Yes	Yes	Yes

Instance Store

Oracle Smart Flash Cache - Requires EC2 with Oracle Linux

- Environment
  - R3.4xlarge
    - 16 vCPU, 122G RAM, **320G SSD Instance Store**
  - 12.1.0.2
  - Sga_target = 70G
  - Oracle Linux 6.x
  - 1.5T database
  - ASM: 5 * 500G gp2 (7500 baseline IOPS)



- Enabling Oracle Smart Flash Cache
  - db_flash_cache_file = '/dev/xvdb2'
  - db_flash_cache_size = '300G'

Oracle Smart Flash Cache

Does it help?

Top 10 Foreground Events by Total Wait Time

Event	Waits	Total Wait Time (sec)	Wait Avg(ms)	% DB time	Wait Class
DB CPU		80.2K		56.9	
db flash cache single block physical read	1.1E+08	29K	0.25	20.6	User I/O
db flash cache multiblock physical read	31,429,386	19.1K	0.61	13.6	User I/O
direct path read	7,659,211	9556.2	1.25	6.8	User I/O
db file sequential read	5,500,648	4241	0.77	3.0	User I/O
read by other session	7,002,787	1842.5	0.26	1.3	User I/O
TCP Socket (KGAS)	31,684	1727.7	54.53	1.2	Network
log file sync	1,054,332	1094.8	1.04	.8	Commit
enq: TX - row lock contention	502	991.7	1975.59	.7	Application
cursor: pin S wait on X	1,360	641.6	471.77	.5	Concurrency

- 95% of potential IO is provided from OSFC!
- Performed ~3 times faster compared the EBS IO

Not accounted and not limited!

STORAGE OPTIMIZED INSTANCES…

… WITH ASM PREFERRED READ FAILURE GROUPS

- ASM
  - Normal redundancy
  - 2 failover groups
    - EBS volumes
    - NVMe SSD devices
- Reads are served by NVMe
- Writes are simultaneous

- NVMe is ephemeral
  - Data may disappear

Model	vCPU	Mem (GiB)	Storage (TB)
i3.large	2	15.25	**1 x 0.475 NVMe SSD**
i3.xlarge	4	30.5	**1 x 0.95 NVMe SSD**
i3.2xlarge	8	61	**1 x 1.9 NVMe SSD**
i3.4xlarge	16	122	**2 x 1.9 NVMe SSD**
i3.8xlarge	32	244	**4 x 1.9 NVMe SSD**
i3.16xlarge	64	488	**8 x 1.9 NVMe SSD**

CONCLUSIONS

SUMMARY

- You can definitely run Oracle on AWS!

- RDS is simpler to use and requires less attention
  - Might be preferred for setting up unattended development DBs
  - It's useful for Production too
  - Learning curve
  - Understand the trade-offs
  - Testing is still relevant! (for non-managed maintenance tasks)

- EC2 will perform better than RDS if done right
  - You're in charge of everything
    - It's good and bad at the same time
  - Think about “smart IO architecture“

THANK YOU

@MarisDBA

elsins@pythian.com