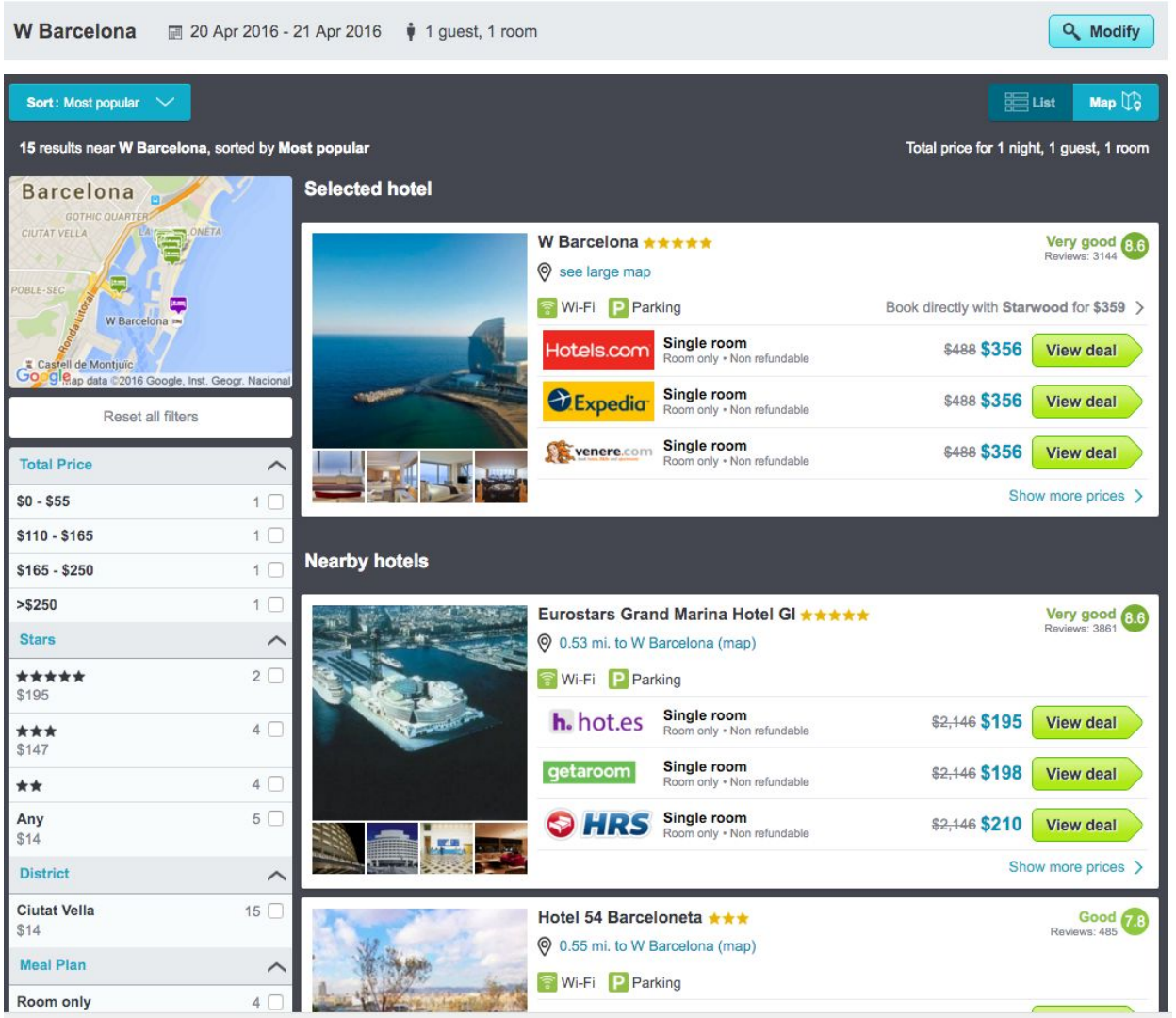
Ingesting 35M images with Python

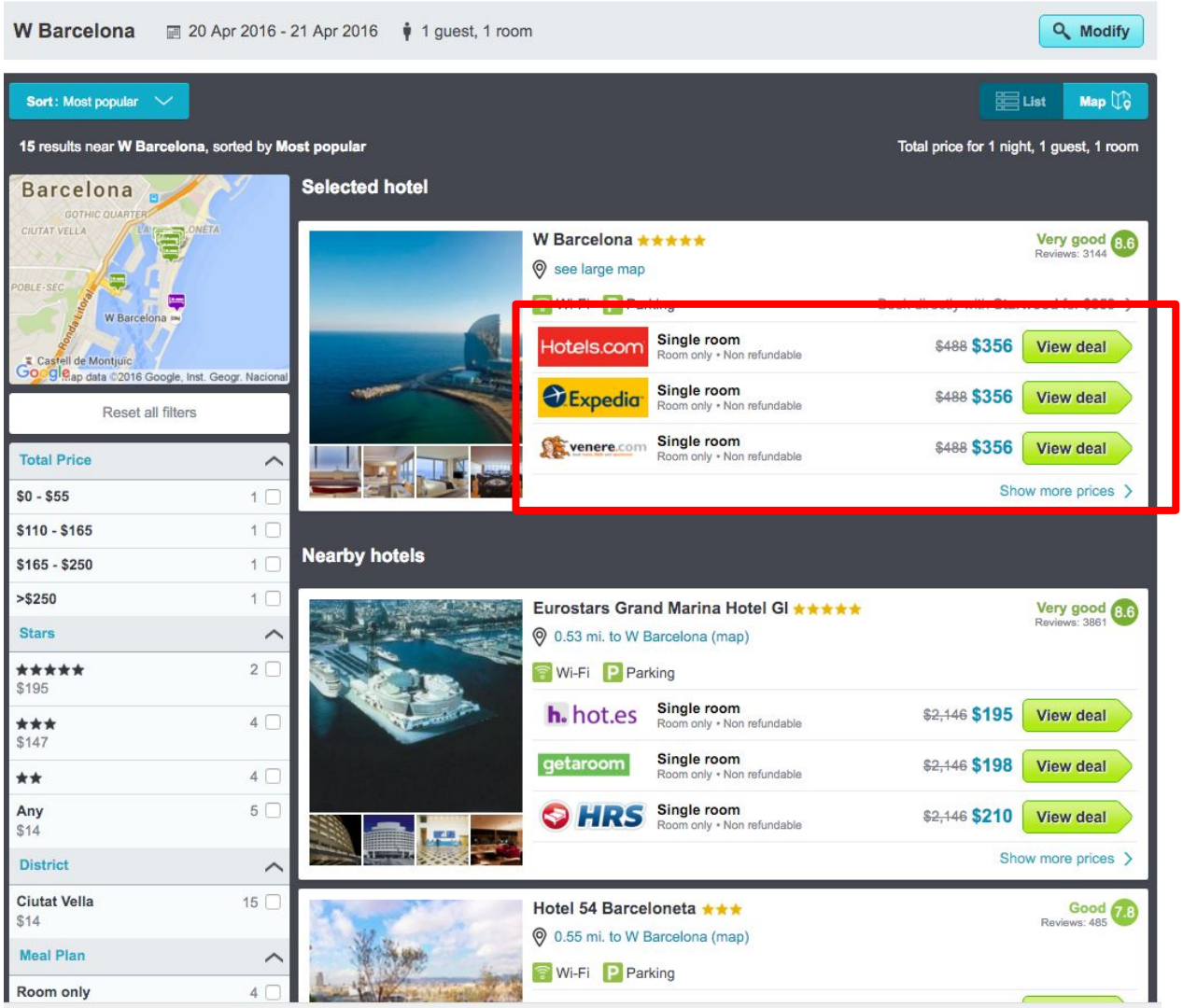
In the cloud.

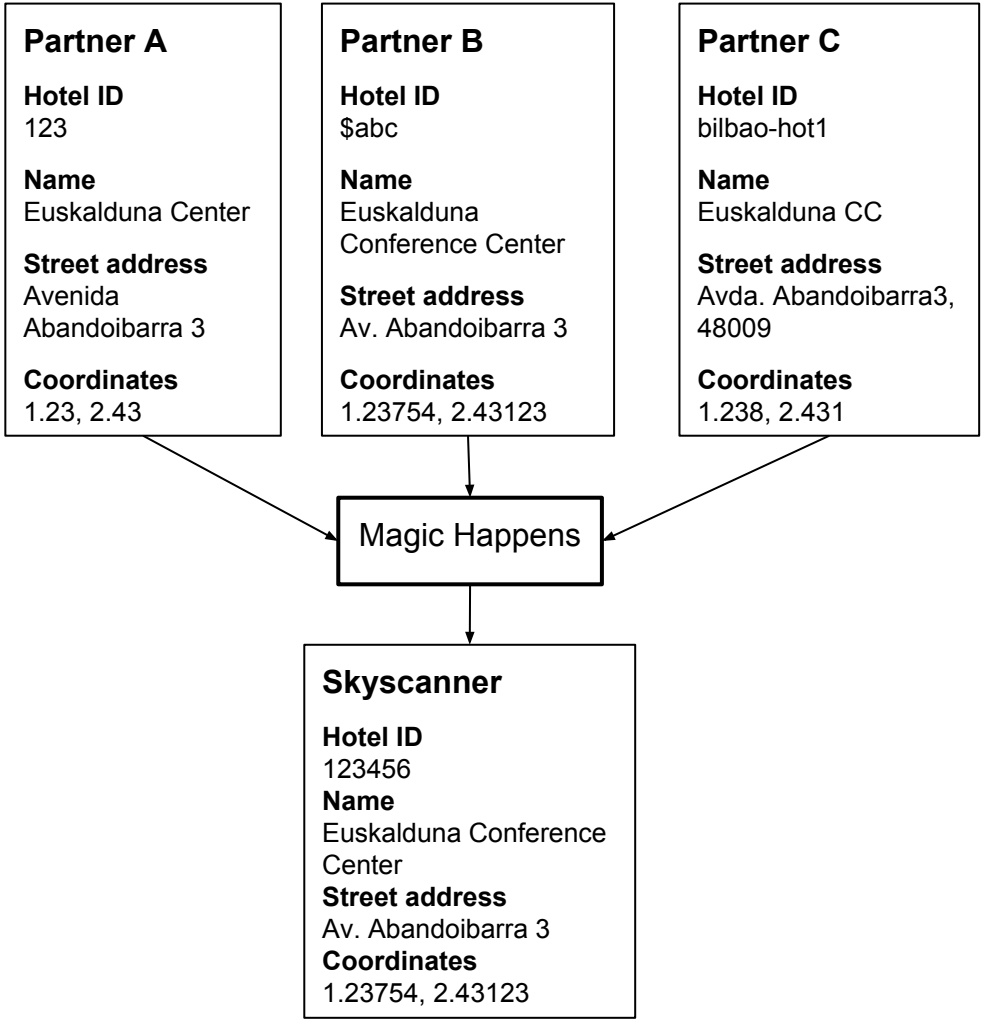
Àlex Vinyals
Software Engineer @ Hotels Data

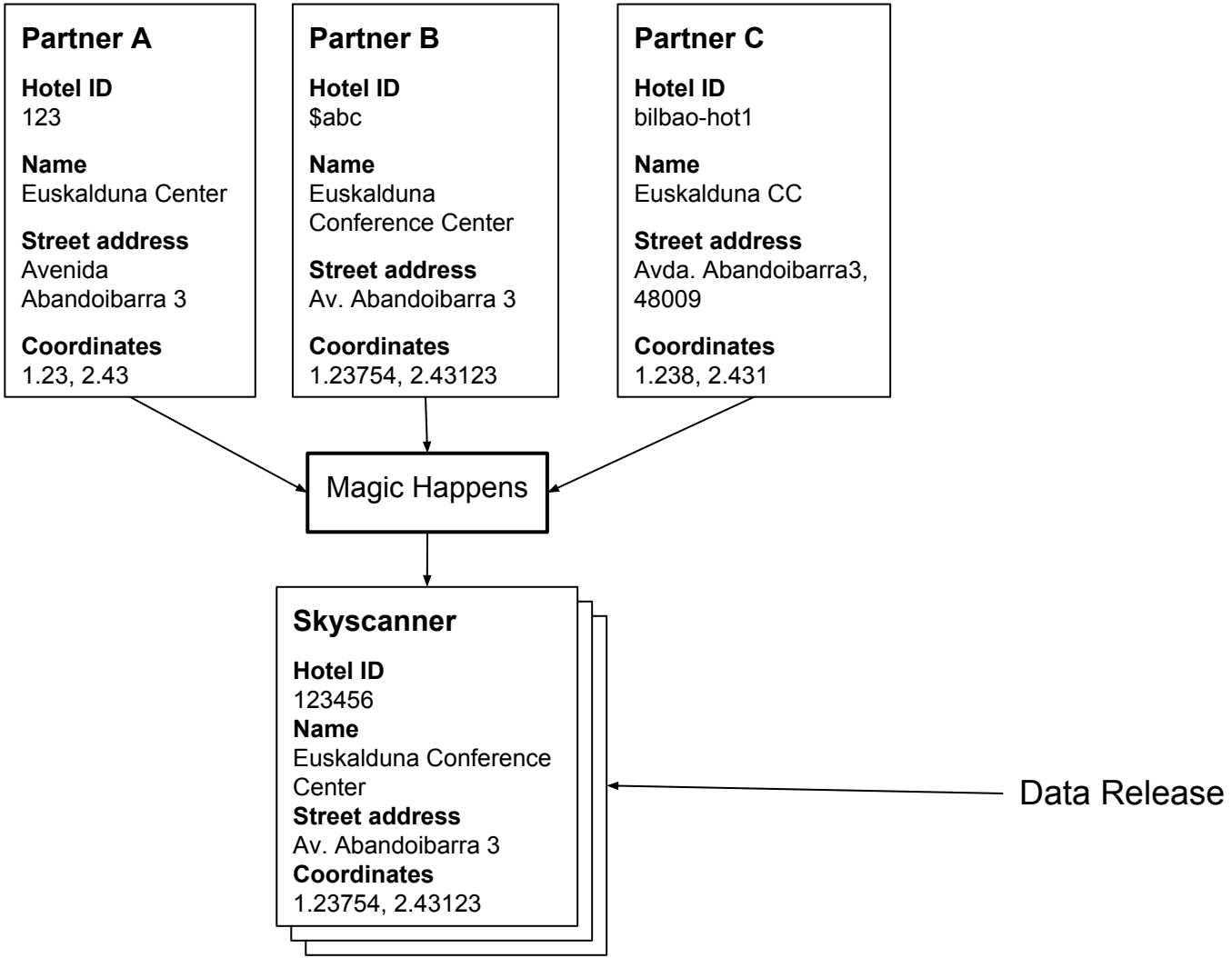
Unify all the data

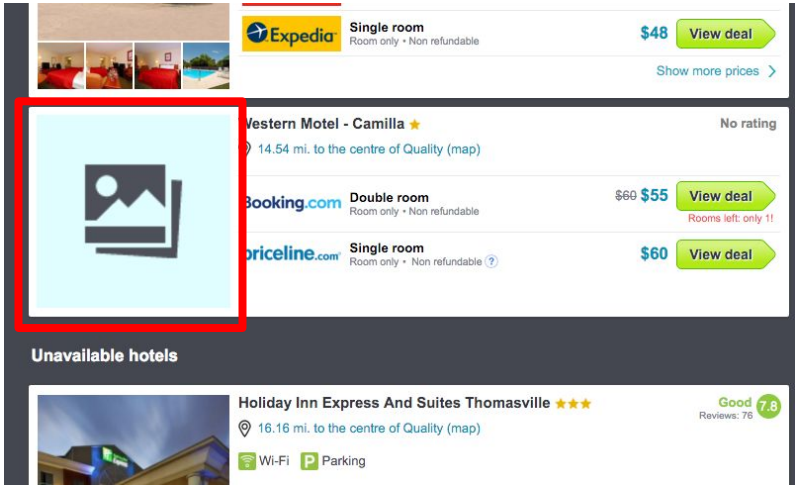
Challenges of a metasearch


W Barcelona
20 Apr 2016 - 21 Apr 2016
1 guest, 1 room
Modify
Sort: Most popular
List
Map
15 results near W Barcelona, sorted by Most popular
Total price for 1 night, 1 guest, 1 room
Barcelona
GOTHIC QUARTER
CIUTAT VELLA
POBLE-SEC
W Barcelona
Ronda Litoral
Castell de Montjuic
Google
Map data ©2016 Google, Inst. Geogr. Nacional
Reset all filters
Total Price
$0 - $55 1
$110 - $165 1
$165 - $250 1
>$250 1
Stars
★★★★★ $195 2
★★★ $147 4
★★ 4
Any $14 5
District
Ciutat Vella $14 15
Meal Plan
Room only 4
Selected hotel
W Barcelona ★★★★★
see large map
Very good 8.6
Reviews: 3144
Wi-Fi
Parking
Book directly with Starwood for $359
Hotels.com
Single room
Room only • Non refundable
$488 $356
View deal
Expedia
Single room
Room only • Non refundable
$488 $356
View deal
venere.com
Single room
Room only • Non refundable
$488 $356
View deal
Show more prices
Nearby hotels
Eurostars Grand Marina Hotel Gl ★★★★★
0.53 mi. to W Barcelona (map)
Very good 8.6
Reviews: 3861
Wi-Fi
Parking
h.hot.es
Single room
Room only • Non refundable
$2,146 $195
View deal
getaroom
Single room
Room only • Non refundable
$2,146 $198
View deal
HRS
Single room
Room only • Non refundable
$2,146 $210
View deal
Show more prices
Hotel 54 Barceloneta ★★★
0.55 mi. to W Barcelona (map)
Good 7.8
Reviews: 485
Wi-Fi
Parking

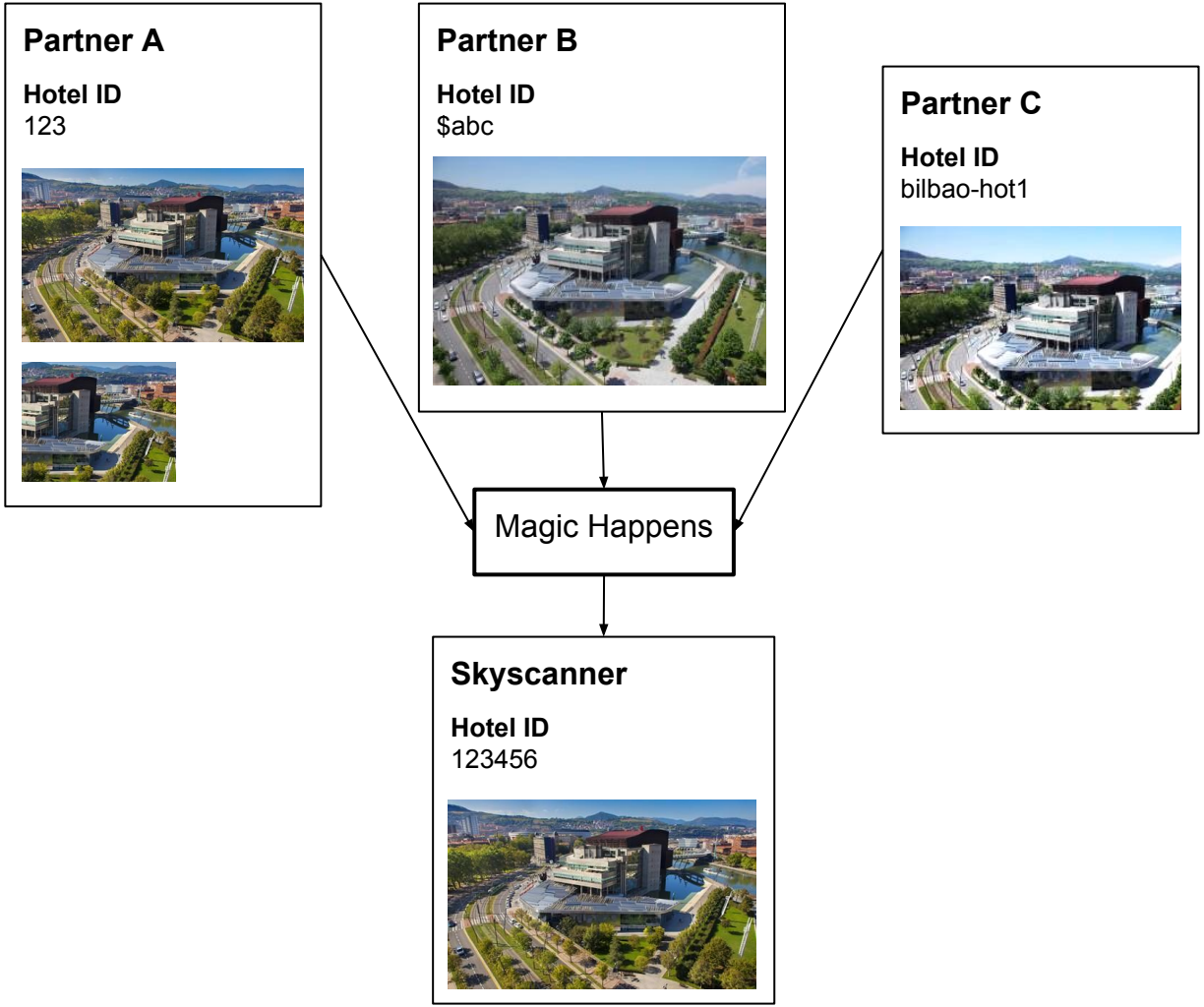

W Barcelona
20 Apr 2016 - 21 Apr 2016
1 guest, 1 room
Modify
Sort: Most popular
List
Map
15 results near W Barcelona, sorted by Most popular
Total price for 1 night, 1 guest, 1 room
Barcelona
GOTHIC QUARTER
CIUTAT VELLA
POBLE-SEC
W Barcelona
Ronda Litoral
Castell de Montjuic
Google
Map data ©2016 Google, Inst. Geogr. Nacional
Reset all filters
Total Price
$0 - $55 1
$110 - $165 1
$165 - $250 1
>$250 1
Stars
★★★★★ $195 2
★★★ $147 4
★★ 4
Any $14 5
District
Ciutat Vella $14 15
Meal Plan
Room only 4
Selected hotel
W Barcelona ★★★★★
see large map
Very good 8.6
Reviews: 3144
Hotels.com Single room Room only • Non refundable $488 $356 View deal
Expedia Single room Room only • Non refundable $488 $356 View deal
venere.com Single room Room only • Non refundable $488 $356 View deal
Show more prices
Nearby hotels
Eurostars Grand Marina Hotel Gl ★★★★★
0.53 mi. to W Barcelona (map)
Very good 8.6
Reviews: 3861
Wi-Fi Parking
h.hot.es Single room Room only • Non refundable $2,146 $195 View deal
getaroom Single room Room only • Non refundable $2,146 $198 View deal
HRS Single room Room only • Non refundable $2,146 $210 View deal
Show more prices
Hotel 54 Barceloneta ★★★
0.55 mi. to W Barcelona (map)
Good 7.8
Reviews: 485
Wi-Fi Parking

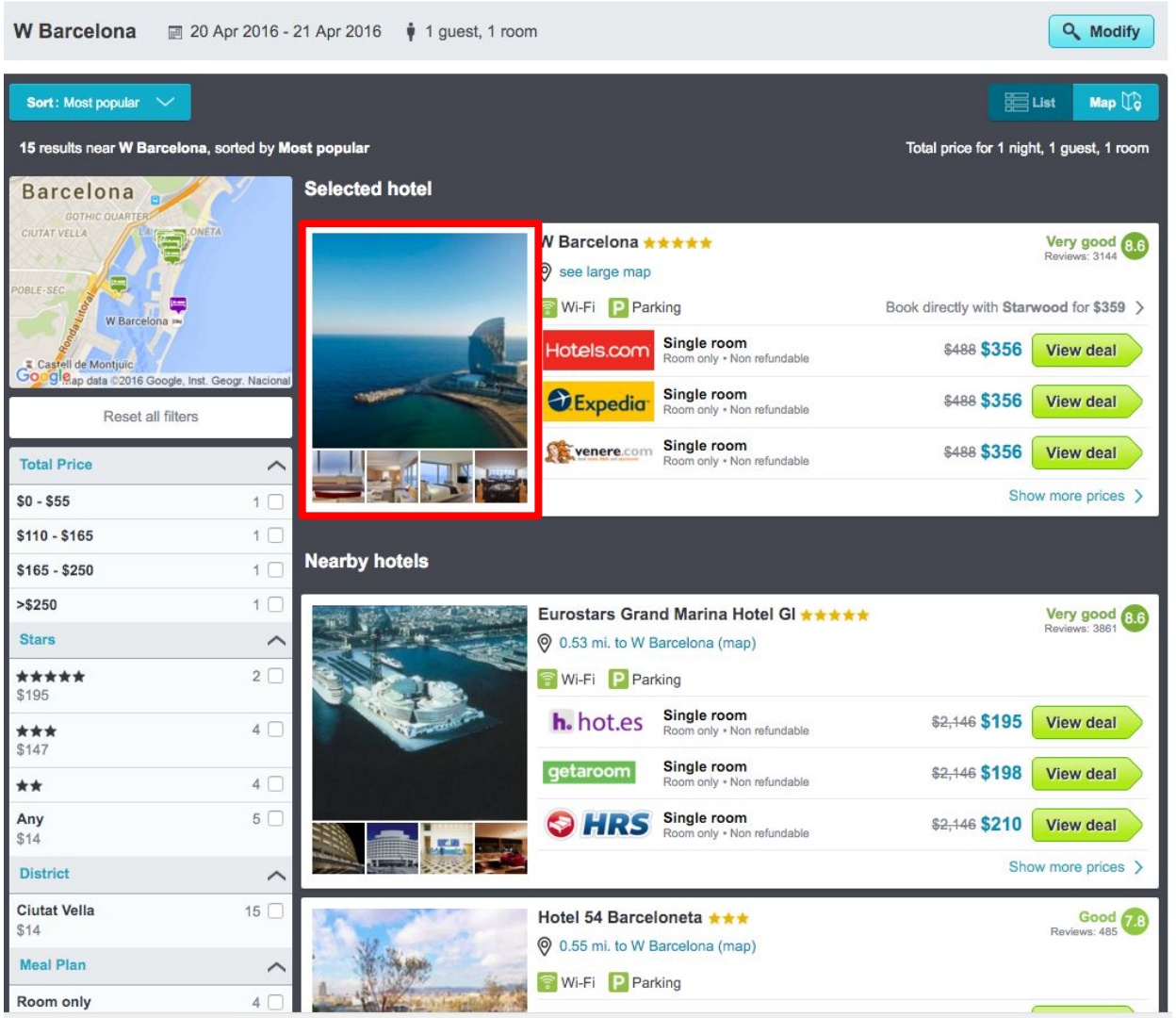

Partner A
Hotel ID
123
Name
Euskalduna Center
Street address
Avenida Abandoibarra 3
Coordinates
1.23, 2.43
Partner B
Hotel ID
$abc
Name
Euskalduna Conference Center
Street address
Av. Abandoibarra 3
Coordinates
1.23754, 2.43123
Partner C
Hotel ID
bilbao-hot1
Name
Euskalduna CC
Street address
Avda. Abandoibarra3, 48009
Coordinates
1.238, 2.431
Magic Happens
Skyscanner
Hotel ID
123456
Name
Euskalduna Conference Center
Street address
Av. Abandoibarra 3
Coordinates
1.23754, 2.43123


Partner A
Hotel ID
123
Name
Euskalduna Center
Street address
Avenida Abandoibarra 3
Coordinates
1.23, 2.43
Partner B
Hotel ID
$abc
Name
Euskalduna Conference Center
Street address
Av. Abandoibarra 3
Coordinates
1.23754, 2.43123
Partner C
Hotel ID
bilbao-hot1
Name
Euskalduna CC
Street address
Avda. Abandoibarra3, 48009
Coordinates
1.238, 2.431
Magic Happens
Skyscanner
Hotel ID
123456
Name
Euskalduna Conference Center
Street address
Av. Abandoibarra 3
Coordinates
1.23754, 2.43123
Data Release

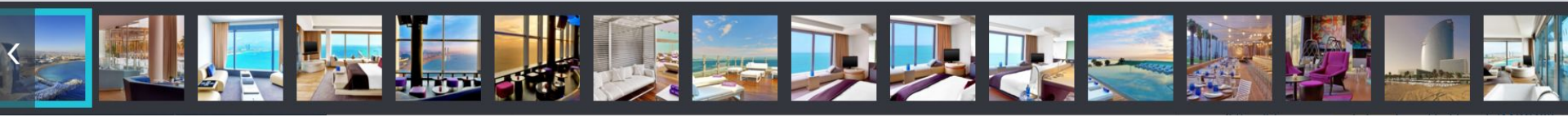


So what about the images?

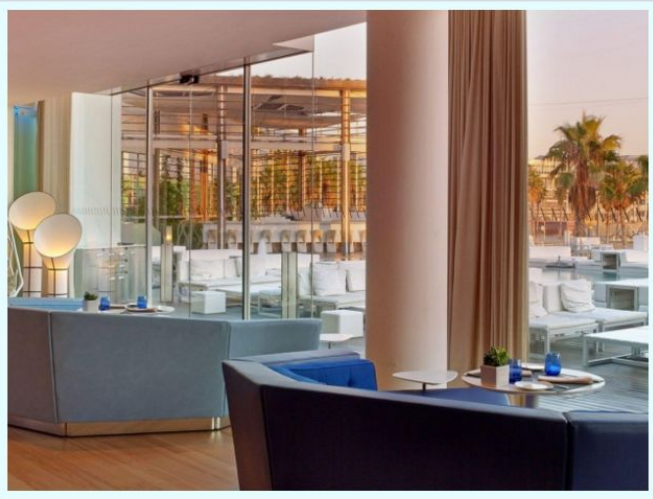

Partner A
Hotel ID
123
Partner B
Hotel ID
$abc
Partner C
Hotel ID
bilbao-hot1
Magic Happens
Skyscanner
Hotel ID
123456

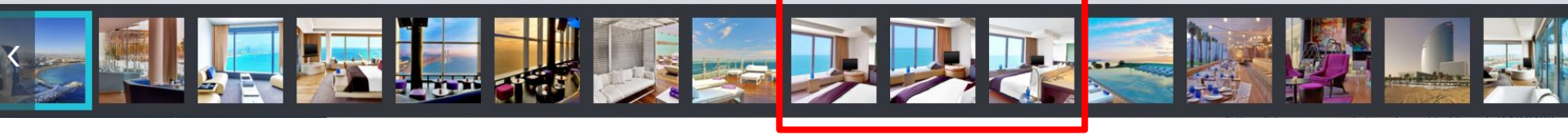

W Barcelona
20 Apr 2016 - 21 Apr 2016
1 guest, 1 room
Modify
Sort: Most popular
List
Map
15 results near W Barcelona, sorted by Most popular
Total price for 1 night, 1 guest, 1 room
Barcelona
GOTHIC QUARTER
CIUTAT VELLA
POBLE-SEC
W Barcelona
Ronda Litoral
Castell de Montjuic
Google
Map data ©2016 Google, Inst. Geogr. Nacional
Reset all filters
Total Price
$0 - $55 1
$110 - $165 1
$165 - $250 1
>$250 1
Stars
★★★★★ $195 2
★★★ $147 4
★★ 4
Any $14 5
District
Ciutat Vella $14 15
Meal Plan
Room only 4
Selected hotel
W Barcelona ★★★★★
see large map
Very good 8.6
Reviews: 3144
Wi-Fi
Parking
Book directly with Starwood for $359
Hotels.com
Single room
Room only • Non refundable
$488 $356
View deal
Expedia
Single room
Room only • Non refundable
$488 $356
View deal
venere.com
Single room
Room only • Non refundable
$488 $356
View deal
Show more prices
Nearby hotels
Eurostars Grand Marina Hotel Gl ★★★★★
0.53 mi. to W Barcelona (map)
Very good 8.6
Reviews: 3861
Wi-Fi
Parking
h. hot.es
Single room
Room only • Non refundable
$2,146 $195
View deal
getaroom
Single room
Room only • Non refundable
$2,146 $198
View deal
HRS
Single room
Room only • Non refundable
$2,146 $210
View deal
Show more prices
Hotel 54 Barceloneta ★★★
0.55 mi. to W Barcelona (map)
Good 7.8
Reviews: 485
Wi-Fi
Parking


Source: Leonardo




Source: Leonardo





With more than 200 partners

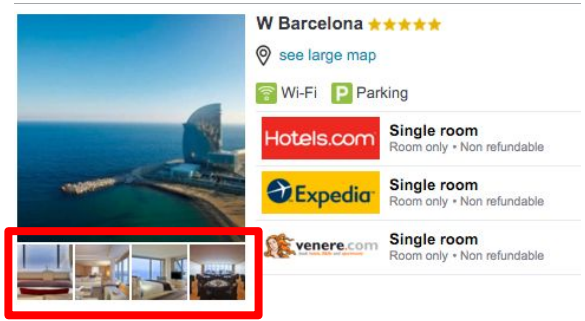
800.000 hotels reach production

Images to process = K * M * N ~ 35M images

K = number of partners
M = avg number of hotels per partner
N = avg number of images per partner hotel

Resizing is a thing

And we have 14 different configurations




```
[sizes]
628x470, scaledownfit
314x235, scaledownfit
2048x1536, scaledownfit
1024x768, scaledownfit
200x200, scalefill, crop
100x100, scalefill, crop
280x185, 185x280, scaledownfill, crop, contrast 1.05
182x182, scaledownfill, crop, contrast 1.05
102x78, scaledownfill, crop, contrast 1.1
58x58, scaledownfill, crop, contrast 1.1
480x319, 240x319, scaledownfill, crop, contrast 1.05
45x45, scaledownfill, crop, contrast 1.1
627x470, scalefill, crop
313x235, scalefill, crop
```

Tale of an image processing pipeline

Tech Stack

Riding on AWS

Tech Stack

Riding on AWS

SQS
Simple Queue Service

Tech Stack

Riding on AWS

Compute resources

SQS
Simple Queue Service

*with DjangoRestFramework
*without Django ORM

Libraries

*with DjangoRestFramework
*without Django ORM

Libraries

*with DjangoRestFramework
*without Django ORM

Libraries

Kombu

Messaging / queues / amqp

*with DjangoRestFramework
*without Django ORM

Libraries

SQLAlchemy

Kombu

Messaging / queues / amqp

Boto

Amazon stuff

*with DjangoRestFramework
*without Django ORM

Pillow

Image Processing

Libraries

Kombu

Messaging / queues / amqp

Boto

Amazon stuff

*with DjangoRestFramework
*without Django ORM

Pillow

Image Processing

Libraries

Python2.7

SQLAlchemy

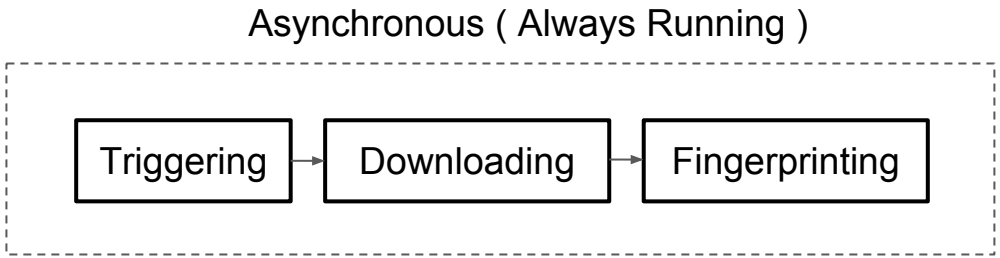
Kombu

Messaging / queues / amqp

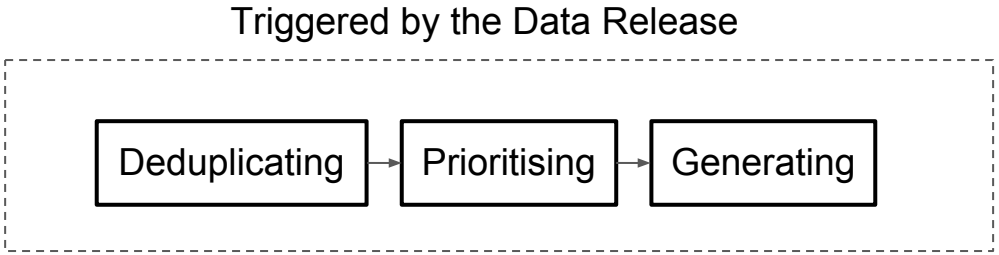
Boto

Amazon stuff

Tale of an image processing pipeline



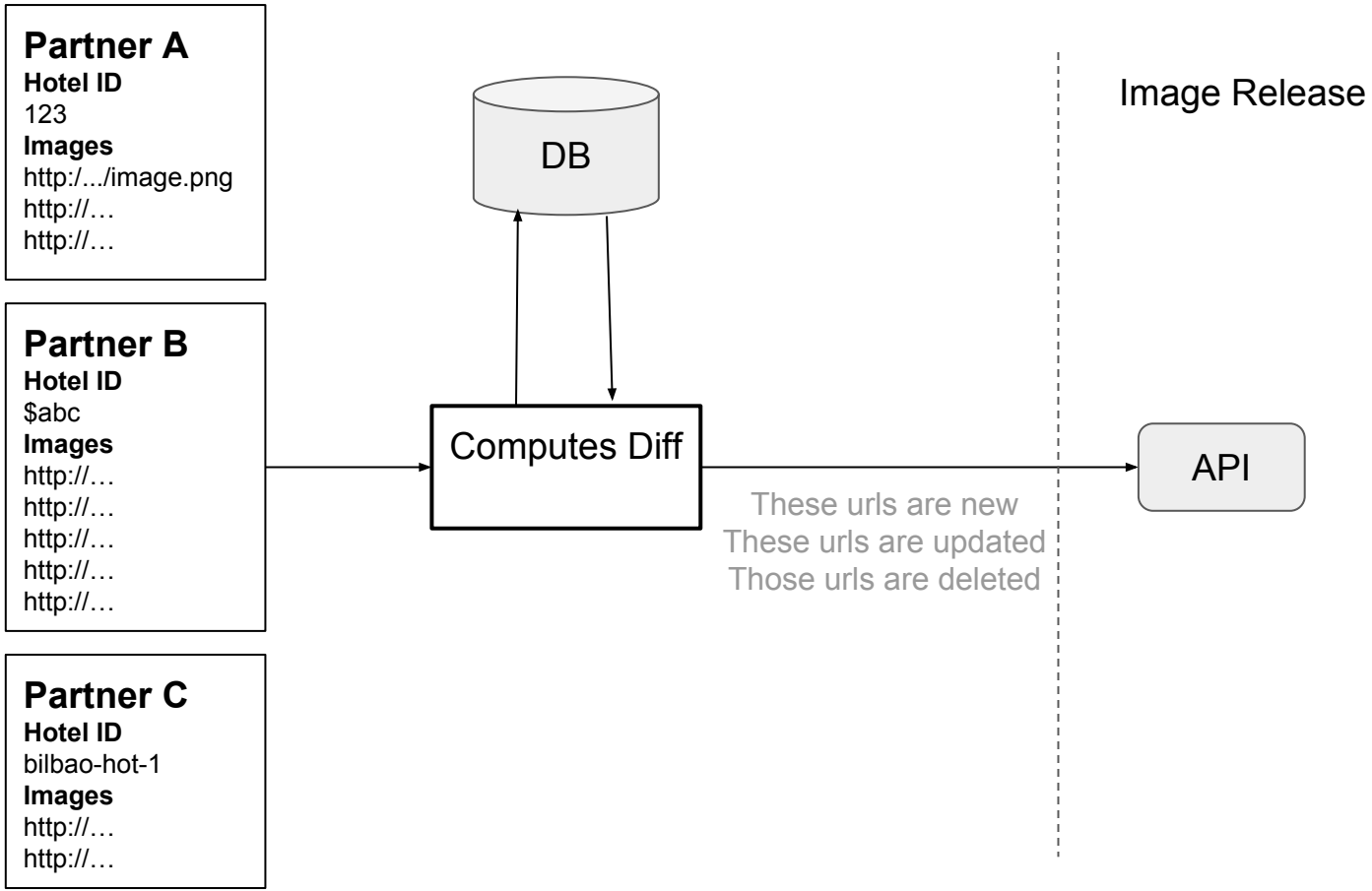
Tale of an image processing pipeline




Triggering
Downloading
Fingerprinting

Triggering


Deduplicating
Prioritising
Generating

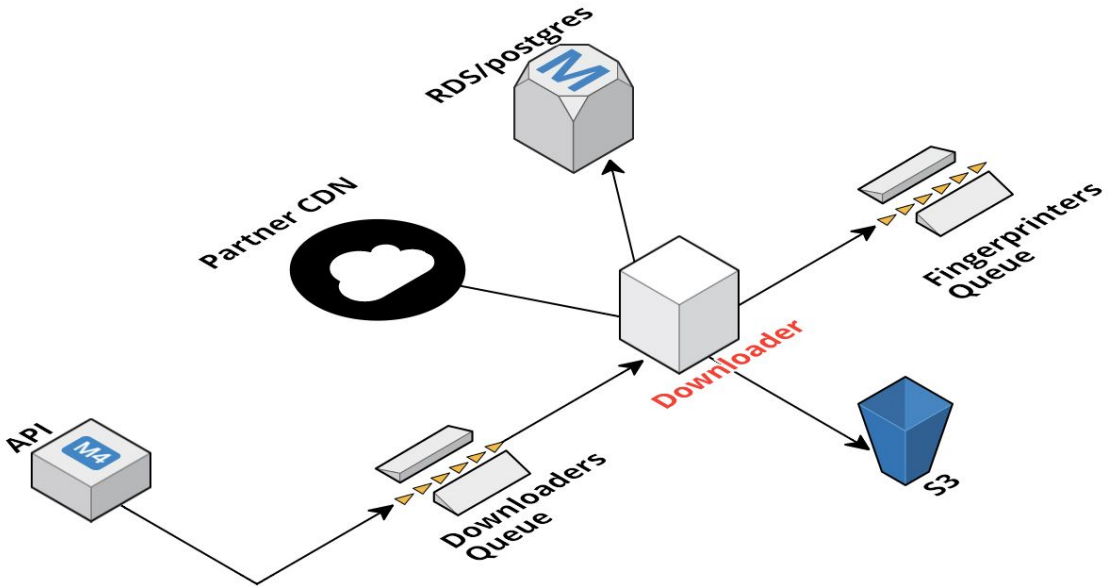

Partner A
Hotel ID
123
Images
http:/.../image.png
http://…
http://…
Partner B
Hotel ID
$abc
Images
http://…
http://…
http://…
http://…
http://…
Partner C
Hotel ID
bilbao-hot-1
Images
http://…
http://…
Catalogues
DB
Computes Diff
These urls are new
These urls are updated
Those urls are deleted
Image Release
API

id	url	partner_id	acc_id	status	en
23452927	http://hote[redacted]re.com/h	4	3969975	available	t
23452917	http://hote[redacted]re.com/h	4	3969975	available	t
23452848	http://img.[redacted]otels/CAM	5	THN3J4	production	t
23452841	http://img.[redacted]otels/CAM	5	THN3J4	production	t
23452846	http://img.[redacted]otels/CAM	5	THN3J4	production	t
23452844	http://img.[redacted]otels/CAM	5	THN3J4	production	t
23452851	http://img.[redacted]otels/CAM	5	THN3J4	production	t
23452796	http://d1pa[redacted]cloudfro	9	252966	production	t
23452797	http://d1pa[redacted]cloudfro	9	252966	production	t
23452794	http://d1pa[redacted]cloudfro	9	252966	production	t
23452788	http://www.[redacted].com/ima	18	931146	filtered	t
23452914	http://hote[redacted]re.com/h	4	3969975	available	t
23452798	http://d1pa[redacted]cloudfro	9	252966	production	t
23452802	http://d1pa[redacted]cloudfro	9	252966	production	t
23452922	http://hote[redacted]re.com/h	4	3969975	available	t
23452919	http://hote[redacted]re.com/h	4	3969975	available	t
23452795	http://d1pa[redacted]cloudfro	9	252966	production	t
23452924	http://hote[redacted]re.com/h	4	3969975	available	t
23452799	http://d1pa[redacted]cloudfro	9	252966	production	t

id	url	partner_id	acc_id	status	en
23452927	http://hote[redacted]re.com/h	4	3969975	available	t
23452917	http://hote[redacted]re.com/h	4	3969975	available	t
23452848	http://img.[redacted]otels/CAM	5	THN3J4	production	t
23452841	http://img.[redacted]otels/CAM	5	THN3J4	production	t
23452846	http://img.[redacted]otels/CAM	5	THN3J4	production	t
23452844	http://img.[redacted]otels/CAM	5	THN3J4	production	t
23452851	http://img.[redacted]otels/CAM	5	THN3J4	production	t
23452796	http://d1pa[redacted]cloudfro	9	252966	production	t
23452797	http://d1pa[redacted]cloudfro	9	252966	production	t
23452794	http://d1pa[redacted]cloudfro	9	252966	production	t
23452788	http://www.[redacted].com/ima	18	931146	filtered	t
23452914	http://hote[redacted]re.com/h	4	3969975	available	t
23452798	http://d1pa[redacted]cloudfro	9	252966	production	t
23452802	http://d1pa[redacted]cloudfro	9	252966	production	t
23452922	http://hote[redacted]re.com/h	4	3969975	available	t
23452919	http://hote[redacted]re.com/h	4	3969975	available	t
23452795	http://d1pa[redacted]cloudfro	9	252966	production	t
23452924	http://hote[redacted]re.com/h	4	3969975	available	t
23452799	http://d1pa[redacted]cloudfro	9	252966	production	t

id	url	partner_id	acc_id	status	en
23452927	http://hote[redacted]re.com/h	4	3969975	available	t
23452917	http://hote[redacted]re.com/h	4	3969975	available	t
23452848	http://img.[redacted]otels/CAM	5	THN3J4	production	t
23452841	http://img.[redacted]otels/CAM	5	THN3J4	production	t
23452846	http://img.[redacted]otels/CAM	5	THN3J4	production	t
23452844	http://img.[redacted]otels/CAM	5	THN3J4	production	t
23452851	http://img.[redacted]otels/CAM	5	THN3J4	production	t
23452796	http://d1pa[redacted] cloudfro	9	252966	production	t
23452797	http://d1pa[redacted] cloudfro	9	252966	production	t
23452794	http://d1pa[redacted] cloudfro	9	252966	production	t
23452788	http://www.[redacted].com/ima	18	931146	filtered	t
23452914	http://hote[redacted]re.com/h	4	3969975	available	t
23452798	http://d1pa[redacted] cloudfro	9	252966	production	t
23452802	http://d1pa[redacted] cloudfro	9	252966	production	t
23452922	http://hote[redacted]re.com/h	4	3969975	available	t
23452919	http://hote[redacted]re.com/h	4	3969975	available	t
23452795	http://d1pa[redacted] cloudfro	9	252966	production	t
23452924	http://hote[redacted]re.com/h	4	3969975	available	t
23452799	http://d1pa[redacted] cloudfro	9	252966	production	t

Downloading


RDS/postgres
Partner CDN
Fingerprinters Queue
Downloader
API
M4
Downloaders Queue
S3

```python
import io
import boto
import requests
from PIL import Image

s3 = boto.connect_s3()
bucket = s3.get_bucket('available-images')

@reliable_callback()
def downloader_callback(queued_image):
    """ Overly simplified downloading callback without
    error handling logic """
    response = requests.get(queued_image.url)
    blob = response.content
    key = bucket.new_key(queued_image.basename)
    key.set_contents_from_string(blob)
    image = Image.open(io.BytesIO(blob))
    if should_filter(image):
        return
    fingerprinting_producer.publish(queued_image)
```

```python
def should_filter(image):
  height, width = image.size

  short_size = min(width, height)
  if short_size < minimum_short:
      return True

  long_size = max(width, height)
  if long_size < minimum_long:
      return True

  total_pixels = width * height
  if total_pixels > max_pixels:
      return True

  return False
```

```python
import io
import boto
import requests
from PIL import Image

s3 = boto.connect_s3()
bucket = s3.get_bucket('available-images')

@reliable_callback()
def downloader_callback(queued_image):
    """ Overly simplified downloading callback without
    error handling logic """
    response = requests.get(queued_image.url)
    blob = response.content
    key = bucket.new_key(queued_image.basename)
    key.set_contents_from_string(blob)
    image = Image.open(io.BytesIO(blob))
    if should_filter(image):
        return
    fingerprinting_producer.publish(queued_image)
```

```python
def should_filter(image):
    height, width = image.size

    short_size = min(width, height)
    if short_size < minimum_short:
        return True

    long_size = max(width, height)
    if long_size < minimum_long:
        return True

    total_pixels = width * height
    if total_pixels > max_pixels:
        return True

    return False
```

```python
import io
import boto
import requests
from PIL import Image

s3 = boto.connect_s3()
bucket = s3.get_bucket('available-images')

@reliable_callback()
def downloader_callback(queued_image):
    """ Overly simplified downloading callback without
    error handling logic """
    response = requests.get(queued_image.url)
    blob = response.content
    key = bucket.new_key(queued_image.basename)
    key.set_contents_from_string(blob)
    image = Image.open(io.BytesIO(blob))
    if should_filter(image):
        return
    fingerprinting_producer.publish(queued_image)
```

```python
def should_filter(image):
  height, width = image.size

  short_size = min(width, height)
  if short_size < minimum_short:
      return True

  long_size = max(width, height)
  if long_size < minimum_long:
      return True

  total_pixels = width * height
  if total_pixels > max_pixels:
      return True

  return False
```

```python
import functools
import warnings
from PIL import Image

def reliable_callback():
    def decorator(func):
        @functools.wraps(func)
        def wrapper(*args, **kwargs):
            warnings.simplefilter('error', Image.DecompressionBombWarning)
            try:
                return func(*args, **kwargs)
            except BaseException:
                logger.error("Critical worker error", exc_info=True)
        return wrapper
    return decorator
```

```python
import functools
import warnings
from PIL import Image

def reliable_callback():
    def decorator(func):
        @functools.wraps(func)
        def wrapper(*args, **kwargs):
            warnings.simplefilter('error', Image.DecompressionBombWarning)
            try:
                return func(*args, **kwargs)
            except BaseException:
                logger.error("Critical worker error", exc_info=True)
        return wrapper
    return decorator
```

```python
import functools
import warnings
from PIL import Image

def reliable_callback():
    def decorator(func):
        @functools.wraps(func)
        def wrapper(*args, **kwargs):
            warnings.simplefilter('error', Image.DecompressionBombWarning)
            try:
                return func(*args, **kwargs)
            except BaseException:
                logger.error("Critical worker error", exc_info=True)
        return wrapper
    return decorator
```

```python
import functools
import warnings
from PIL import Image

def reliable_callback():
    def decorator(func):
        @functools.wraps(func)
        def wrapper(*args, **kwargs):
            warnings.simplefilter('error', Image.DecompressionBombWarning)
            try:
                return func(*args, **kwargs)
            except BaseException:
                logger.error("Critical worker error", exc_info=True)
        return wrapper
    return decorator
```

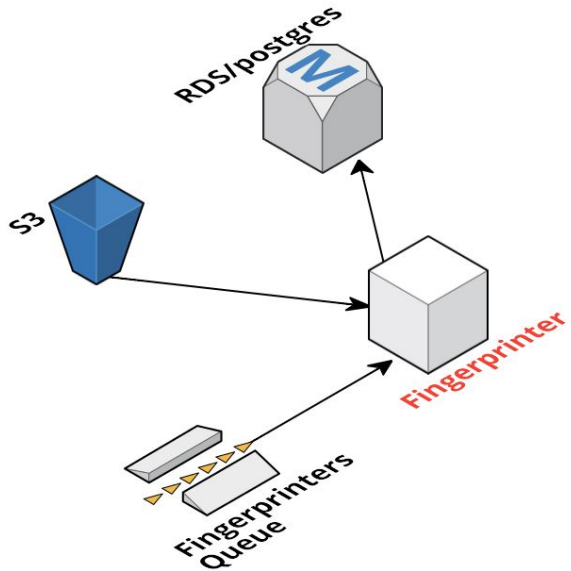
```python
from kombu import Connection, Consumer, Exchange, Queue, eventloop

class KombuConsumer(common.BaseConsumer):
    # ... bla bla

    def callback(self, body, message):
        self.handler(body)
        message.ack()

    def listen(self):
        with Connection(self.backend.broker, transport_options={'region': self.backend.region}) as connection:
            with Consumer(connection, self.queue, callbacks=[self.callback], accept=[self.backend.serializer]):
                for _ in eventloop(connection):
                    pass

# What a simplified worker looks like
# Broker URI stored on Backend object, looks like:
#   sqs://{s3_key}:{s3_secret}@
consumer = KombuConsumer(backend, handler=downloader.downloader_callback)
consumer.listen()
```

```python
from kombu import Connection, Consumer, Exchange, Queue, eventloop

class KombuConsumer(common.BaseConsumer):
    # ... bla bla

    def callback(self, body, message):
        self.handler(body)
        message.ack()

    def listen(self):
        with Connection(self.backend.broker, transport_options={'region': self.backend.region}) as connection:
            with Consumer(connection, self.queue, callbacks=[self.callback], accept=[self.backend.serializer]):
                for _ in eventloop(connection):
                    pass

# What a simplified worker looks like
# Broker URI stored on Backend object, looks like:
#   sqs://{s3_key}:{s3_secret}@
consumer = KombuConsumer(backend, handler=downloader.downloader_callback)
consumer.listen()
```

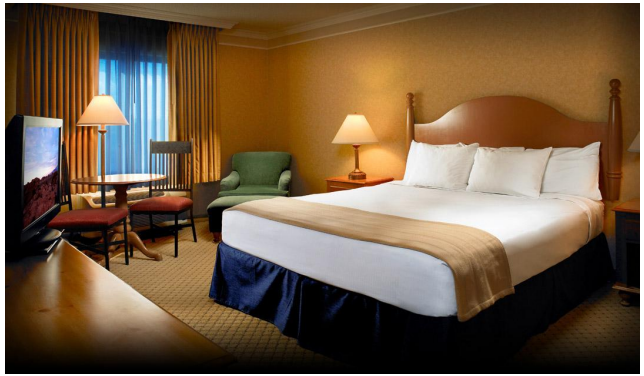
```python
from kombu import Connection, Consumer, Exchange, Queue, eventloop

class KombuConsumer(common.BaseConsumer):
    # ... bla bla

    def callback(self, body, message):
        self.handler(body)
        message.ack()

    def listen(self):
        with Connection(self.backend.broker, transport_options={'region': self.backend.region}) as connection:
            with Consumer(connection, self.queue, callbacks=[self.callback], accept=[self.backend.serializer]):
                for _ in eventloop(connection):
                    pass

# What a simplified worker looks like
# Broker URI stored on Backend object, looks like:
#   sqs://{s3_key}:{s3_secret}@
consumer = KombuConsumer(backend, handler=downloader.downloader_callback)
consumer.listen()
```

```python
from kombu import Connection, Consumer, Exchange, Queue, eventloop

class KombuConsumer(common.BaseConsumer):
    # ... bla bla

    def callback(self, body, message):
        self.handler(body)
        message.ack()

    def listen(self):
        with Connection(self.backend.broker, transport_options={'region': self.backend.region}) as connection:
            with Consumer(connection, self.queue, callbacks=[self.callback], accept=[self.backend.serializer]):
                for _ in eventloop(connection):
                    pass

# What a simplified worker looks like
# Broker URI stored on Backend object, looks like:
#   sqs://{s3_key}:{s3_secret}@
consumer = KombuConsumer(backend, handler=downloader.downloader_callback)
consumer.listen()
```

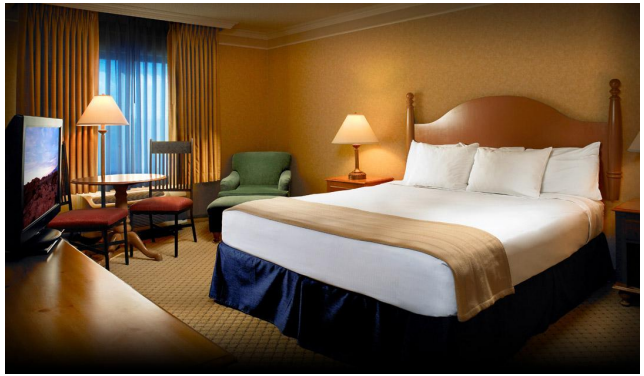
Fingerprinting

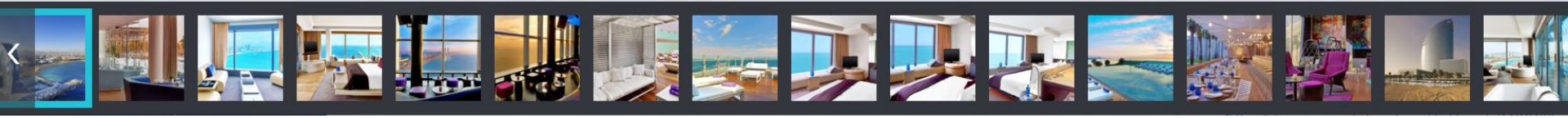

RDS/postgres
S3
Fingerprinter
Fingerprinters
Queue

Are those images the same?



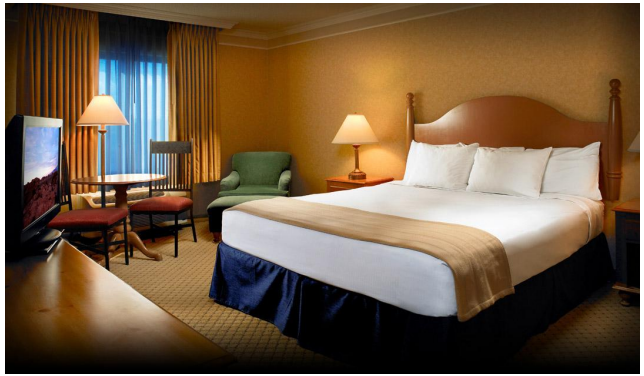
Are those images the same?





Are those images the same?

Yes, they are



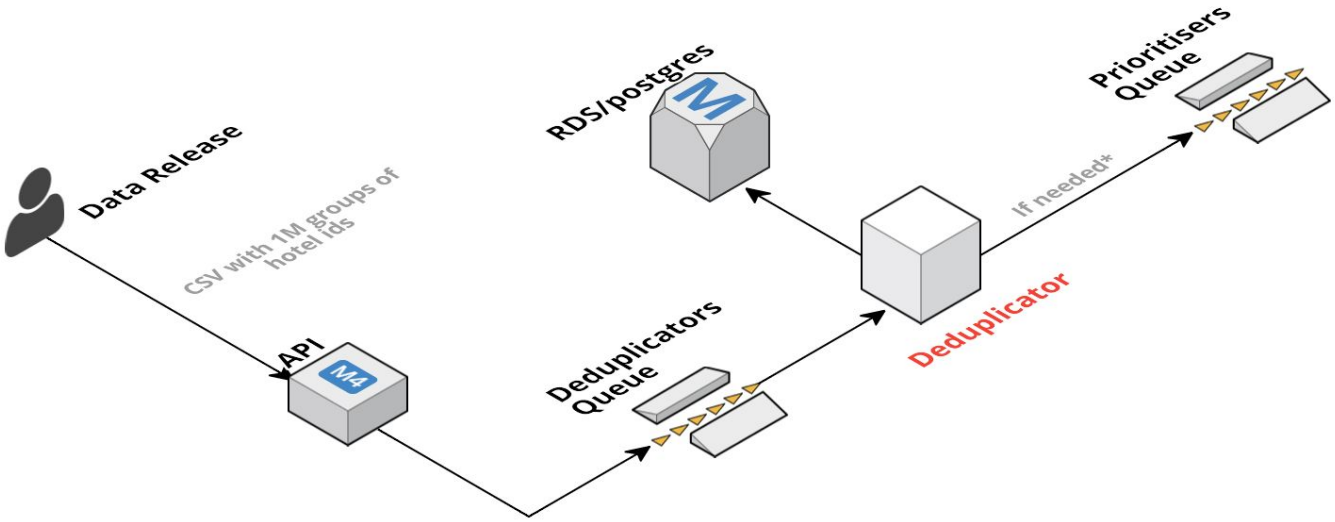
```python
import imagehash

def fingerprint_callback(queued_image):
    blob = download_image_blob(queued_image.basename)
    image = Image.open(BytesIO(blob))
    result = cropped_hash(image, imagehash.phash)
    store_hashes(queued_image.image_id, result)
```

```python
import imagehash

def fingerprint_callback(queued_image):
    blob = download_image_blob(queued_image.basename)
    image = Image.open(BytesIO(blob))
    result = cropped_hash(image, imagehash.dhash)
    store_hashes(queued_image.image_id, result)
```

```python
def cropped_hash(image, algorithm, steps=range(0, 51, 10)):
    result = []
    w, h = image.size
    # We want to cut by steps % of the image (default 0%, 10%...50%), which
    # means we need to cut half of that from each side:
    # | N%/2 |          | N%/2 |
    # +------+----------+------+--
    # |      :          :      | N%/2
    # +- - - +----------+ - - -+--
    # |      |          |      |
    # |      |          |      |
    # +- - - +----------+ - - -+--
    # |      :          :      | N%/2
    # +------+----------+------+--
    for x in steps:
        x_band = x * w / 200
        for y in steps:
            y_band = y * h / 200
            with image.crop((x_band, y_band, w-x_band, h-y_band)) as sub_image:
                sub_hash = algorithm(sub_image)
                result.append(hash_to_int(sub_hash))
```

```python
def cropped_hash(image, algorithm, steps=range(0, 51, 10)):
    result = []
    w, h = image.size
    # We want to cut by steps % of the image (default 0%, 10%...50%), which
    # means we need to cut half of that from each side:
    # | N%/2 |          | N%/2 |
    # +------+----------+------+--
    # |      :          :      | N%/2
    # +- - - +----------+ - - -+--
    # |      |          |      |
    # |      |          |      |
    # +- - - +----------+ - - -+--
    # |      :          :      | N%/2
    # +------+----------+------+--
    for x in steps:
        x_band = x * w / 200
        for y in steps:
            y_band = y * h / 200
            with image.crop((x_band, y_band, w-x_band, h-y_band)) as sub_image:
                sub_hash = algorithm(sub_image)
                result.append(hash_to_int(sub_hash))
```

Deduplication Time


Data Release
CSV with 1M groups of hotel ids
API
M4
Deduplicators Queue
RDS/postgres
M
Deduplicator
If needed*
Prioritisers Queue

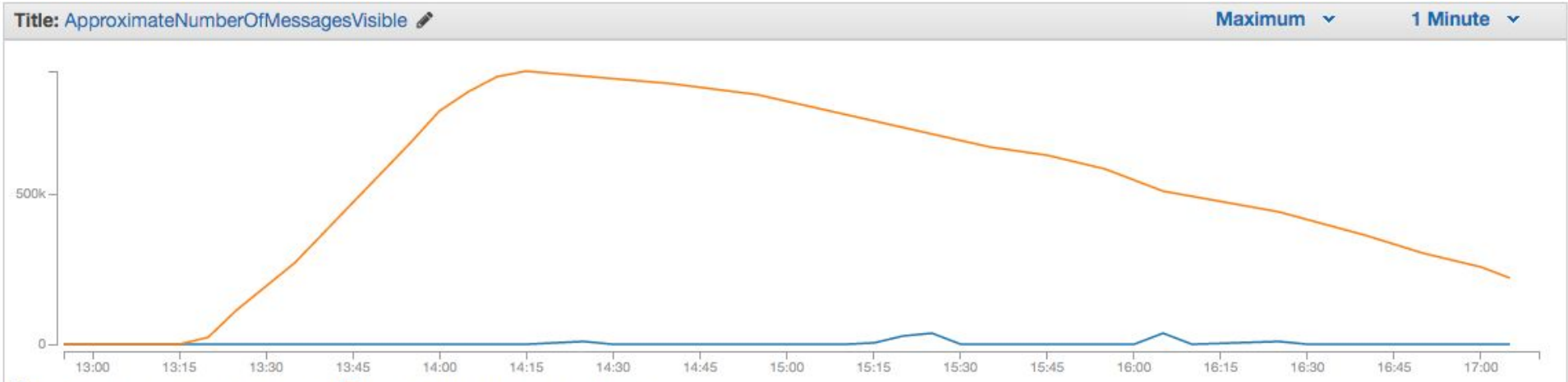
What is a “group”?

id	dr	group_id	status	elements
10220644	70	2053940	done	17,454995,4,501446,13,1003798
10220643	70	2054825	done	34,102912,4,363502,28,37152,46
10220642	70	2054040	done	32,E7D860D5-0576-4BBD-AA26-418
10220641	70	2054394	done	129,243137,131,41113,5,PHAIGV,
10220640	70	2054826	done	33,NOGTIEHTBUMBAVEEBVCV,4,3153

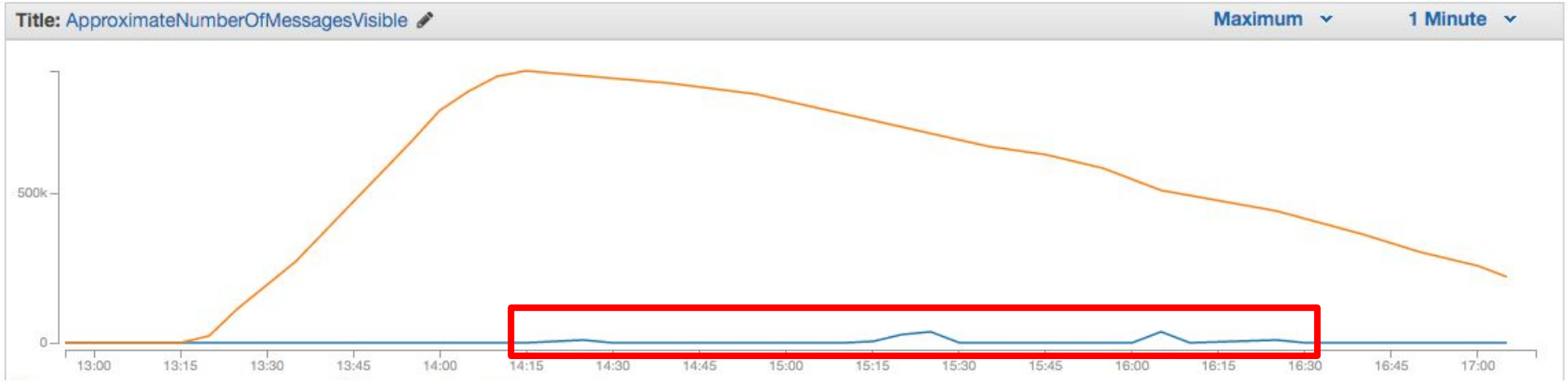
What is a “group”?

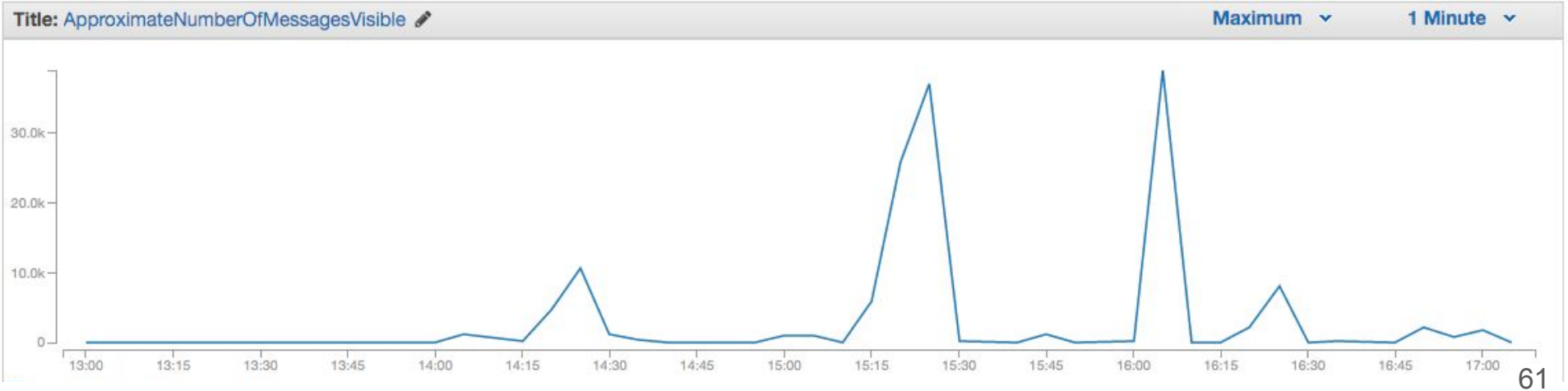
id	dr	group_id	status	elements
10220644	70	2053940	done	17,454995,4,501446,13,1003798
10220643	70	2054825	done	34,102912,4,363502,28,37152,46
10220642	70	2054040	done	32,E7D860D5-0576-4BBD-AA26-418
10220641	70	2054394	done	129,243137,131,41113,5,PHAIGV,
10220640	70	2054826	done	33,NOGTIEHTBUMBAVEEBVCV,4,3153

“If needed”


Title: ApproximateNumberOfMessagesVisible
Maximum
1 Minute
500k
0
13:00
13:15
13:30
13:45
14:00
14:15
14:30
14:45
15:00
15:15
15:30
15:45
16:00
16:15
16:30
16:45
17:00

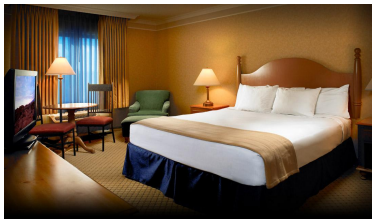
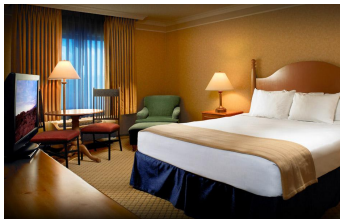
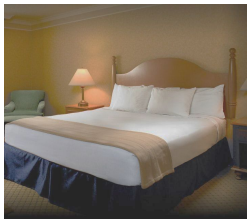
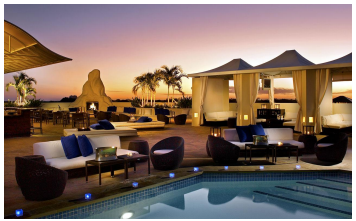
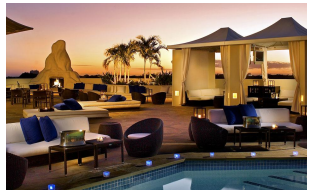
“If needed”


Title: ApproximateNumberOfMessagesVisible
Maximum
1 Minute
500k
0
13:00
13:15
13:30
13:45
14:00
14:15
14:30
14:45
15:00
15:15
15:30
15:45
16:00
16:15
16:30
16:45
17:00

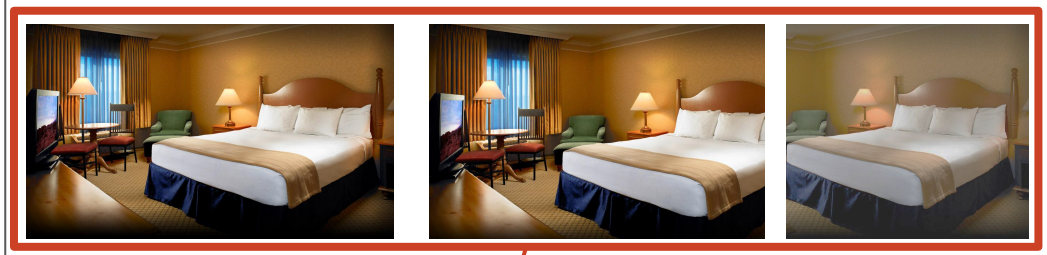

Title: ApproximateNumberOfMessagesVisible
Maximum
1 Minute
30.0k
20.0k
10.0k
0
13:00
13:15
13:30
13:45
14:00
14:15
14:30
14:45
15:00
15:15
15:30
15:45
16:00
16:15
16:30
16:45
17:00

Hotel Group 123
[(partner_id$_1$, accomodation_id$_1$), …, (partner_id$_n$, accomodation_id$_n$)]

Hotel Group 123
[(partner_id$_1$, accomodation_id$_1$), …, (partner_id$_n$, accomodation_id$_n$)]

Hotel Group 123
[(partner_id$_1$, accomodation_id$_1$), …, (partner_id$_n$, accomodation_id$_n$)]



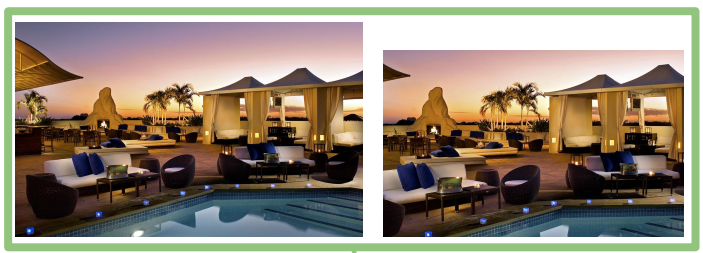
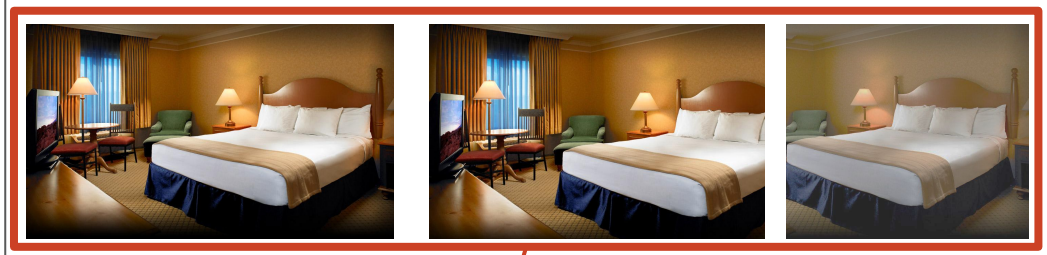


Image Group 456

Image Group 203

Hotel Group 123
[(partner_id$_1$, accomodation_id$_1$), …, (partner_id$_n$, accomodation_id$_n$)]



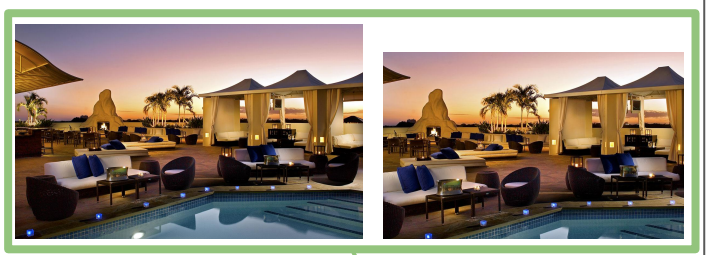


Image Group 456

Image Group 203

Hotel Group 123 has two image groups:
[456, 203]

```
def deduplicate_group(all_hotel_images):
    all_images = set(all_hotel_images)
    groups = []
    while all_images:
        seed_image = all_images.pop()
        group = {seed_image}
        new_additions = {seed_image}
        while new_additions:
            me = new_additions.pop()
            for other in all_images:
                if is_same_picture(me.hashes, other.hashes):
                    group.add(other)
                    new_additions.add(other)
            all_images = all_images - group
        groups.append(group)
    return groups
```

```
def deduplicate_group(all_hotel_images):
    all_images = set(all_hotel_images)
    groups = []
    while all_images:
        seed_image = all_images.pop()
        group = {seed_image}
        new_additions = {seed_image}
        while new_additions:
            me = new_additions.pop()
            for other in all_images:
                if is_same_picture(me.hashes, other.hashes):
                    group.add(other)
                    new_additions.add(other)
            all_images = all_images - group
        groups.append(group)
    return groups
```

```
def deduplicate_group(all_hotel_images):
    all_images = set(all_hotel_images)
    groups = []
    while all_images:
        seed_image = all_images.pop()
        group = {seed_image}
        new_additions = {seed_image}
        while new_additions:
            me = new_additions.pop()
            for other in all_images:
                if is_same_picture(me.hashes, other.hashes):
                    group.add(other)
                    new_additions.add(other)
            all_images = all_images - group
        groups.append(group)
    return groups
```

```
def deduplicate_group(all_hotel_images):
    all_images = set(all_hotel_images)
    groups = []
    while all_images:
        seed_image = all_images.pop()
        group = {seed_image}
        new_additions = {seed_image}
        while new_additions:
            me = new_additions.pop()
            for other in all_images:
                if is_same_picture(me.hashes, other.hashes):
                    group.add(other)
                    new_additions.add(other)
            all_images = all_images - group
        groups.append(group)
    return groups
```

```
def deduplicate_group(all_hotel_images):
    all_images = set(all_hotel_images)
    groups = []
    while all_images:
        seed_image = all_images.pop()
        group = {seed_image}
        new_additions = {seed_image}
        while new_additions:
            me = new_additions.pop()
            for other in all_images:
                if is_same_picture(me.hashes, other.hashes):
                    group.add(other)
                    new_additions.add(other)
            all_images = all_images - group
        groups.append(group)
    return groups
```

```
def is_same_picture(left_hashes, right_hashes):
    for left_hash in left_hashes:
        for right_hash in right_hashes:
            ham_dist = hamdist(left_hash, right_hash)
            if ham_dist < threshold:
                return True
    return False
```

```
def is_same_picture(left_hashes, right_hashes):
    for left_hash in left_hashes:
        for right_hash in right_hashes:
            ham_dist = hamdist(left_hash, right_hash)
            if ham_dist < threshold:
                return True
    return False
```

How do you tune this step?

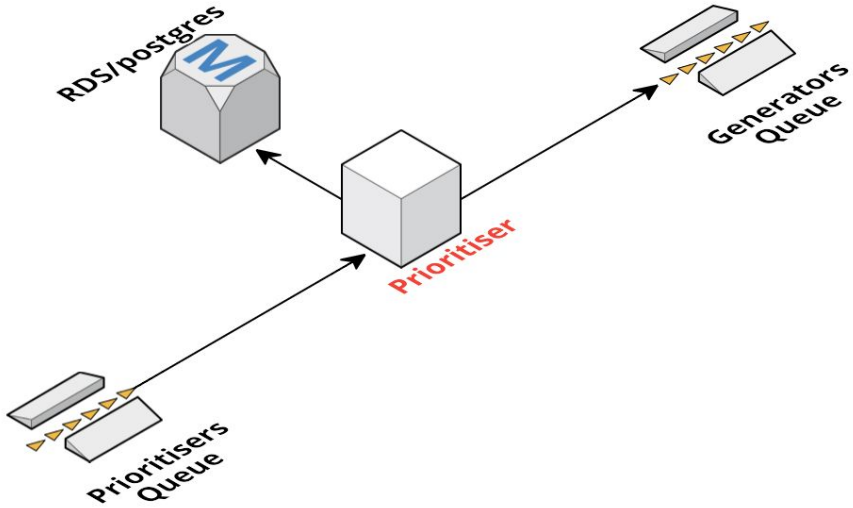
Guarantees are needed

You build a corpus.


IMPURE
INCOMPLETE
IMPURE
INCOMPLETE
IMPURE
INCOMPLETE
IMPURE
COMPLETE
IMPURE
COMPLETE
IMPURE
COMPLETE
IMPURE
URL MD5: 4fdfdd1b909e3555e1af7b0af5a376f5
Group ID: 213

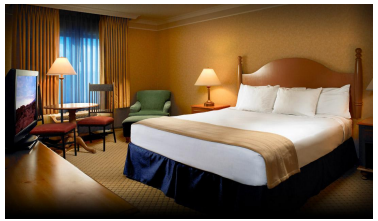
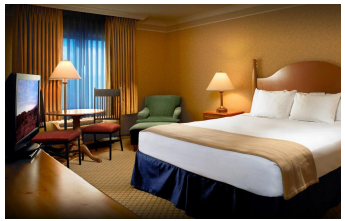
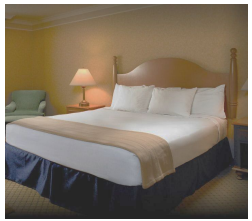
Measure	Run #1	Run #2	Run #3	Run #4	Run #5	Run #6	Run #44	Run #157
General measures								
Homogeneity	83.10%	63.53%	33.76%	99.59%	98.01%	92.20%	99.50%	99.50%
Completeness	99.02%	96.97%	97.74%	99.11%	99.15%	99.27%	99.13%	99.13%
V-measure	90.37%	76.77%	50.18%	99.35%	98.58%	95.60%	99.31%	99.31%
A-score	55.52%	27.61%	10.79%	95.77%	91.06%	72.16%	95.86%	95.86%
Group measures								
Groups complete	88.92%	79.20%	86.03%	91.14%	91.35%	93.12%	91.29%	91.29%
Groups pure	83.81%	97.55%	98.32%	98.23%	93.88%	90.42%	97.82%	97.82%
Image release								
Image loss	29.46%	34.47%	64.13%	-1.80%	5.01%	18.44%	-1.20%	-1.20%
Duplicate probability	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%
Average duplicates	4.14%	10.15%	5.13%	4.46%	3.87%	2.81%	4.26%	4.26%

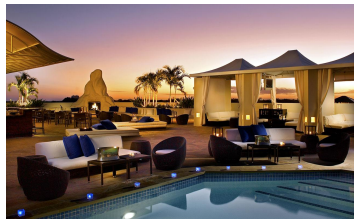
Prioritisation


RDS/postgres
Generators Queue
Prioritiser
Prioritisers Queue

Hotel Group 123
[ImageGroup_{406}, ImageGroup_{203}]

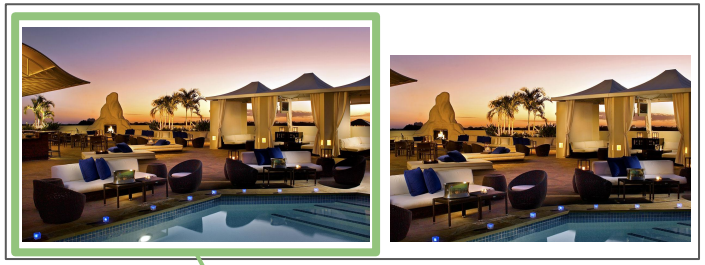
Hotel Group 123
[ImageGroup_{406}, ImageGroup_{203}]




Hotel Group 123
[ImageGroup_{406}, ImageGroup_{203}]

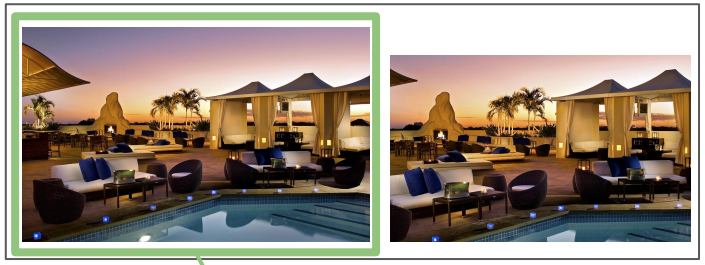
"Best image"



"Best Image"

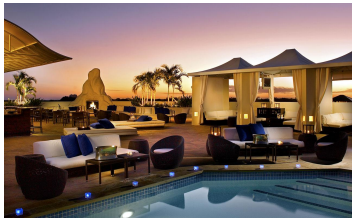
Hotel Group 123
[ImageGroup_{406}, ImageGroup_{203}]

"Best image"

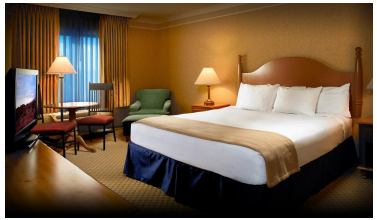


"Best Image"

"Best order"



1

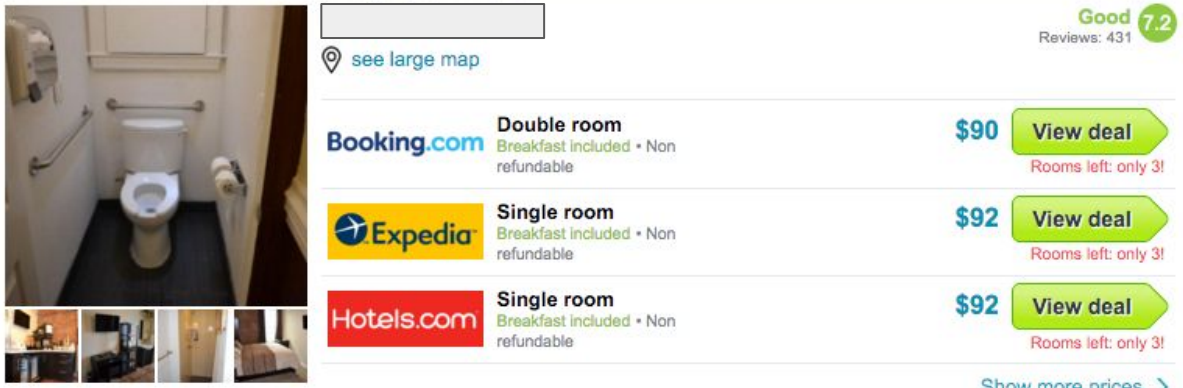


2

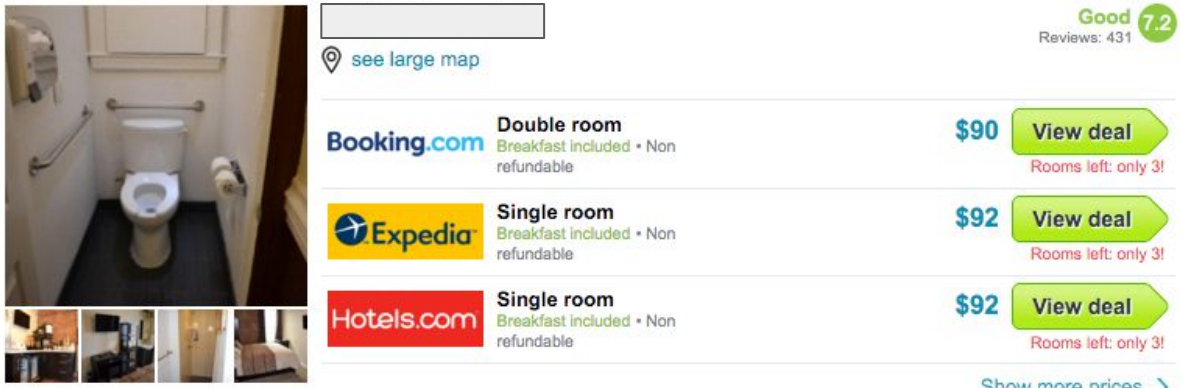
reaches production

What could go wrong?

What could go wrong?



What could go wrong?

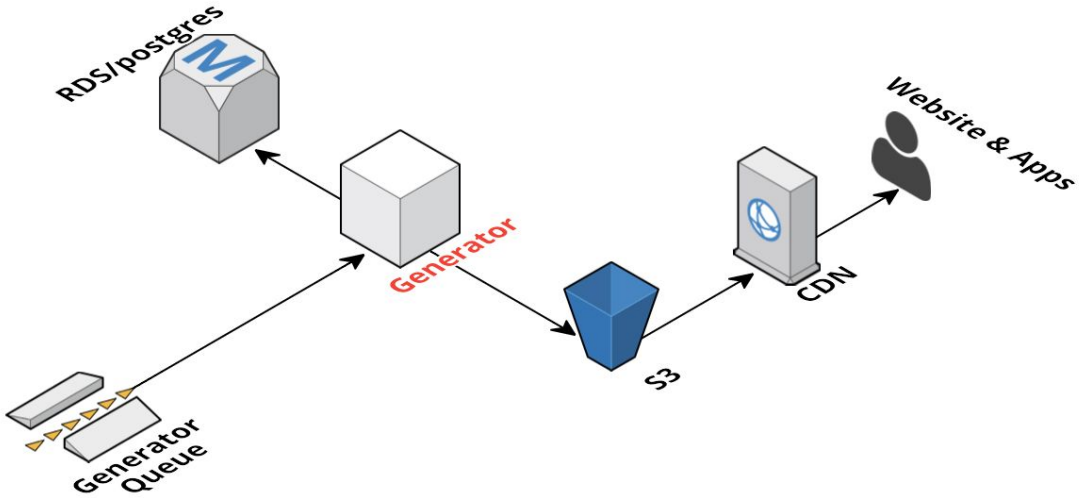


* Detect features, prioritise based on that.
* Tools to manually fix data.


Triggering
Downloading
Fingerprinting

Generation


Deduplicating
Prioritising
Generating


RDS/postgres
Generator
Website & Apps
CDN
S3
Generator
Queue

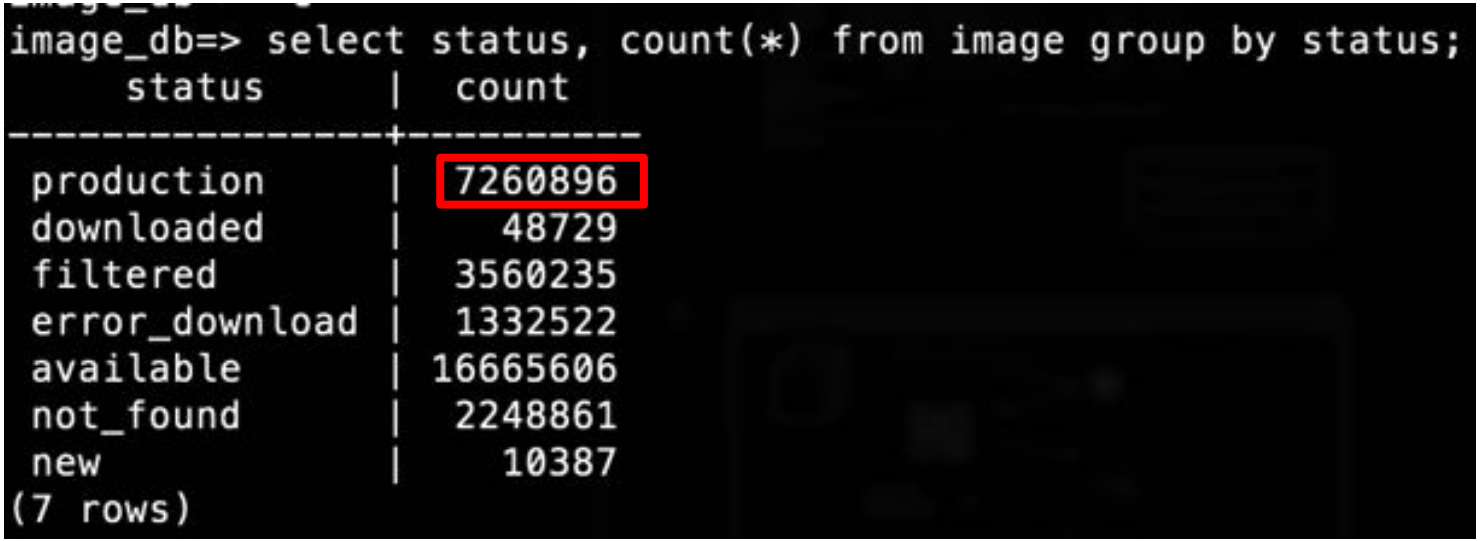
```
from PIL import ImageEnhance

def scalefit(image):
        sz = image.size
        bs = best_size(image)

        # A bit of math:
        # - if we scale to fit width, the scaling factor is: scale_width = bs[0]/sz[0]
        # - if we scale to fit height, the scaling factor is: scale_height = bs[1]/sz[1]
        # We want to scale to the smaller of them (so that the image fits in both), so we scale to width if:
        # scale_width < scale_height => bs[0]/sz[0] < bs[1]/sz[1] => bs[0]*sz[1] < bs[1]*sz[0]
        # Having it in this form means no floats are needed; all in integers.
        if bs[0] * sz[1] < bs[1] * sz[0]:
            # Scale to width
            w = bs[0]
            h = sz[1] * bs[0] / sz[0]
        else:
            # Scale to height
            w = sz[0] * bs[1] / sz[1]
            h = bs[1]

        return image.resize((w, h), "bilinear")

def contrast(image, value):
        enhancer = ImageEnhance.Contrast(image)
        return enhancer.enhance(float(value))
```

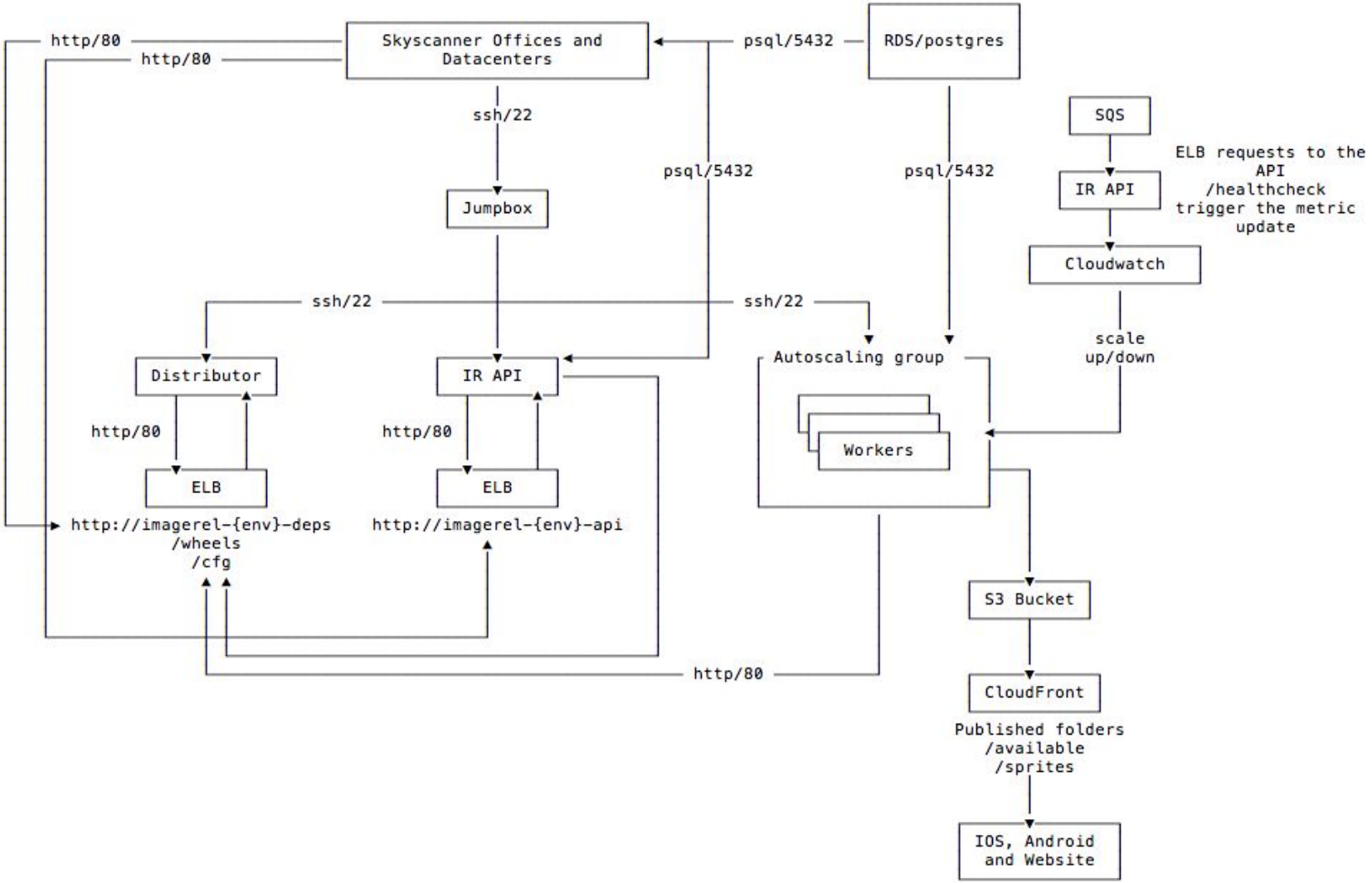
```
from PIL import ImageEnhance

def scalefit(image):
        sz = image.size
        bs = best_size(image)

        # A bit of math:
        # - if we scale to fit width, the scaling factor is: scale_width = bs[0]/sz[0]
        # - if we scale to fit height, the scaling factor is: scale_height = bs[1]/sz[1]
        # We want to scale to the smaller of them (so that the image fits in both), so we scale to width if:
        # scale_width < scale_height => bs[0]/sz[0] < bs[1]/sz[1] => bs[0]*sz[1] < bs[1]*sz[0]
        # Having it in this form means no floats are needed; all in integers.
        if bs[0] * sz[1] < bs[1] * sz[0]:
            # Scale to width
            w = bs[0]
            h = sz[1] * bs[0] / sz[0]
        else:
            # Scale to height
            w = sz[0] * bs[1] / sz[1]
            h = bs[1]

        return image.resize((w, h), "bilinear")

def contrast(image, value):
        enhancer = ImageEnhance.Contrast(image)
        return enhancer.enhance(float(value))
```

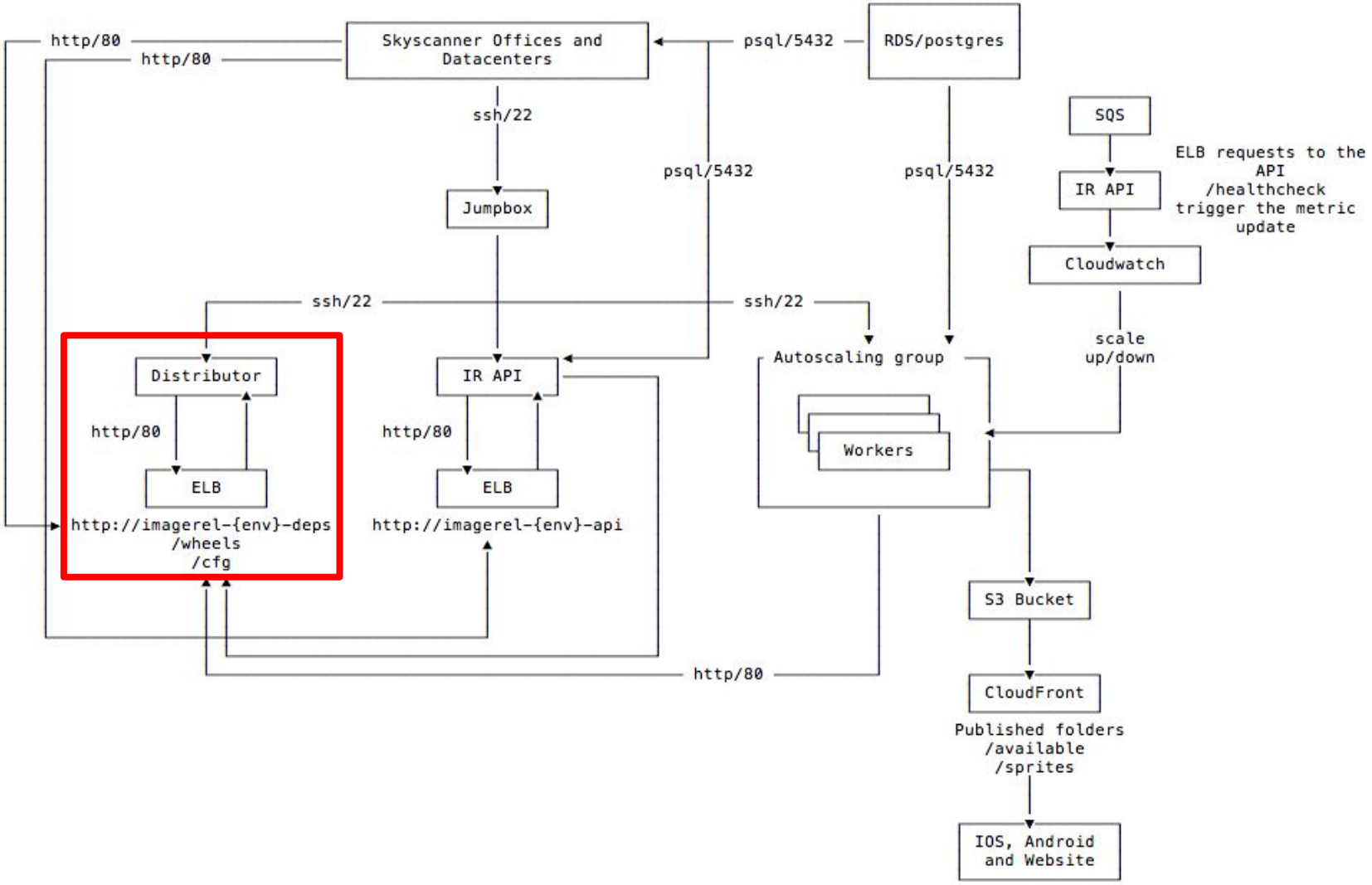
```python
from PIL import ImageEnhance

def scalefit(image):
        sz = image.size
        bs = best_size(image)

        # A bit of math:
        # - if we scale to fit width, the scaling factor is: scale_width = bs[0]/sz[0]
        # - if we scale to fit height, the scaling factor is: scale_height = bs[1]/sz[1]
        # We want to scale to the smaller of them (so that the image fits in both), so we scale to width if:
        # scale_width < scale_height => bs[0]/sz[0] < bs[1]/sz[1] => bs[0]*sz[1] < bs[1]*sz[0]
        # Having it in this form means no floats are needed; all in integers.
        if bs[0] * sz[1] < bs[1] * sz[0]:
            # Scale to width
            w = bs[0]
            h = sz[1] * bs[0] / sz[0]
        else:
            # Scale to height
            w = sz[0] * bs[1] / sz[1]
            h = bs[1]

        return image.resize((w, h), "bilinear")

def contrast(image, value):
        enhancer = ImageEnhance.Contrast(image)
        return enhancer.enhance(float(value))
```

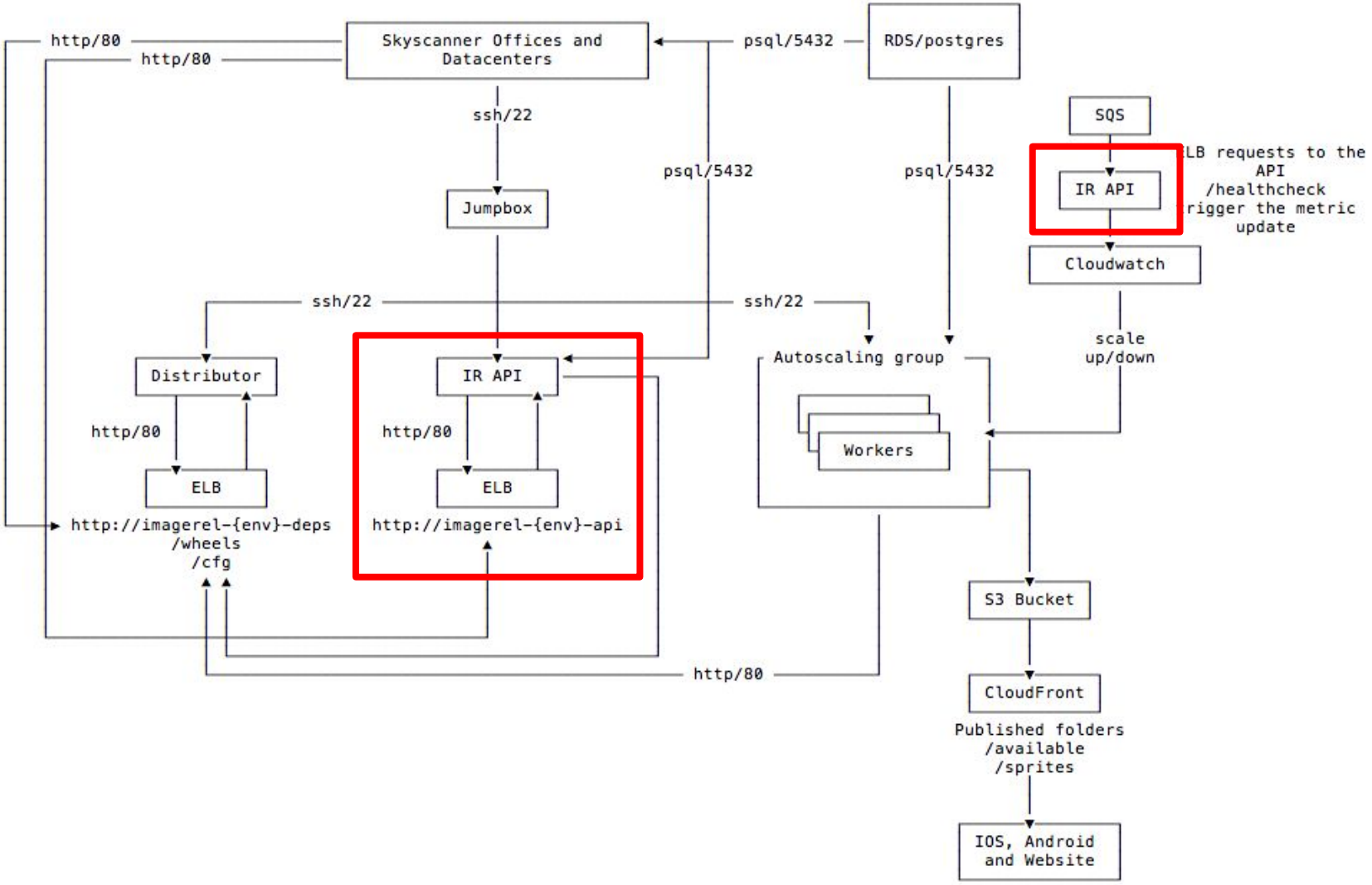
And that’s it for the pipeline

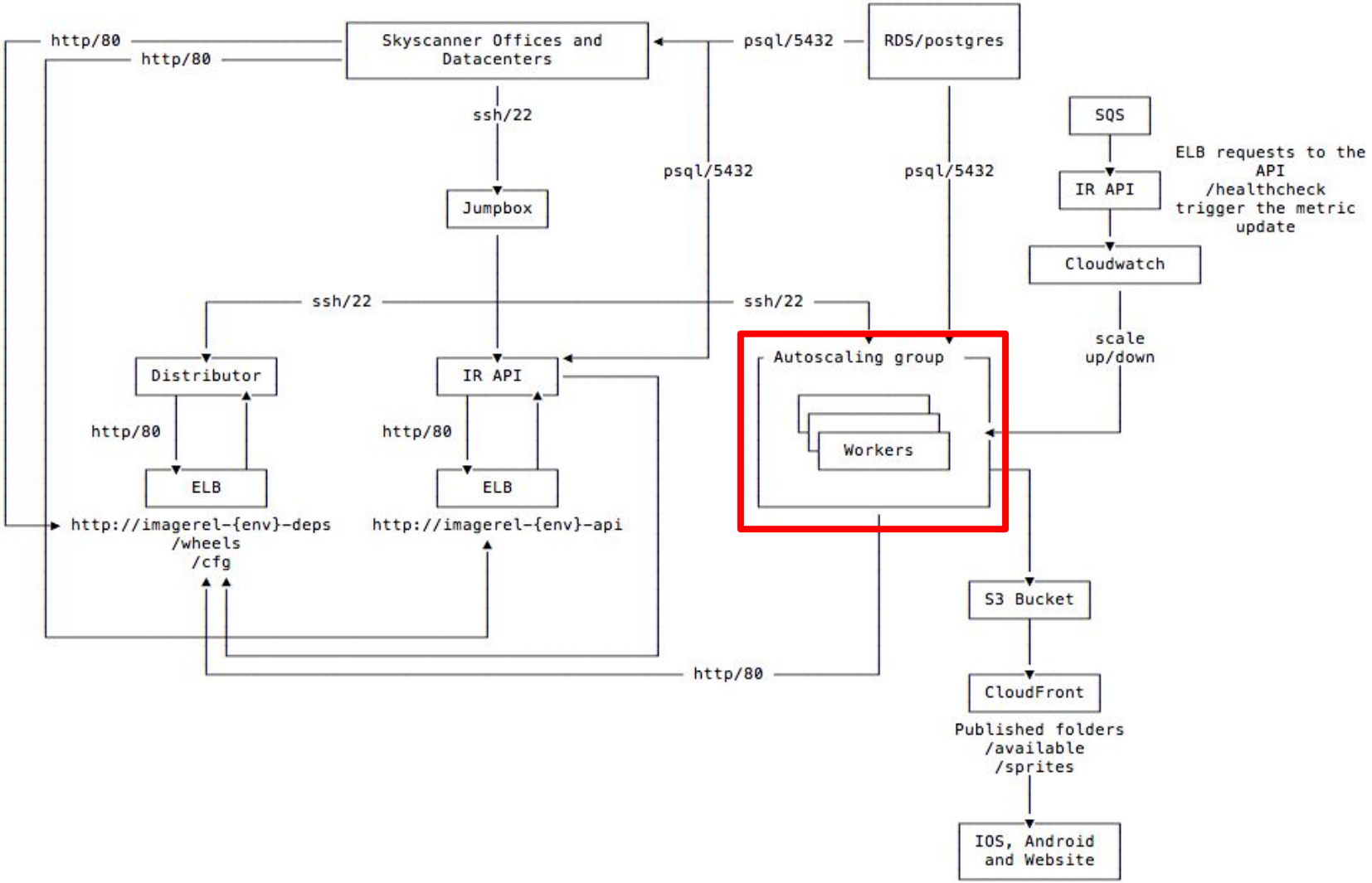
```
image_db=> select status, count(*) from image group by status;
     status     |  count
----------------+----------
 production     |  7260896
 downloaded     |    48729
 filtered       |  3560235
 error_download |  1332522
 available      | 16665606
 not_found      |  2248861
 new            |    10387
(7 rows)
```

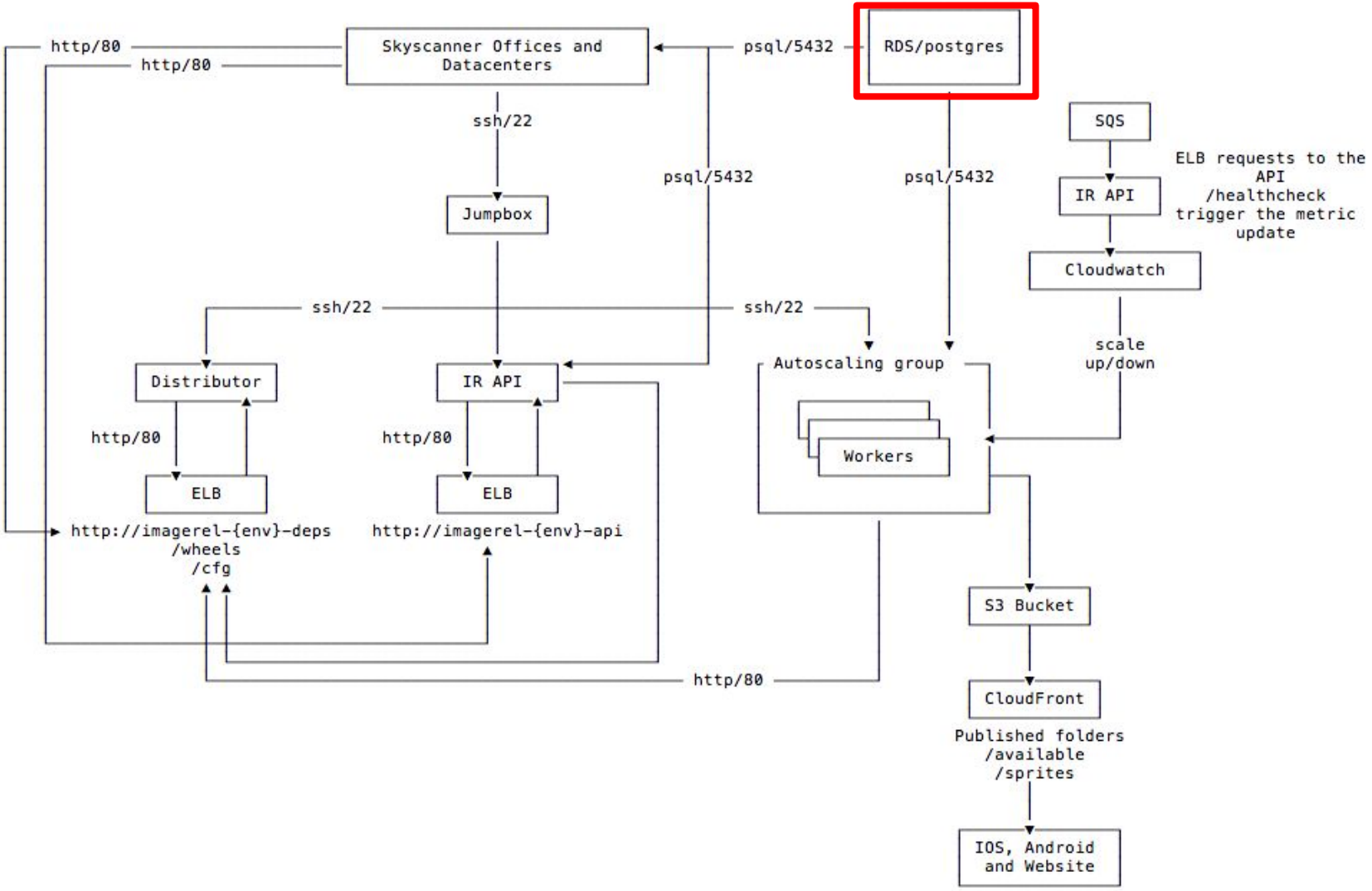

image_db=> select status, count(*) from image group by status;
status | count
production | 7260896
downloaded | 48729
filtered | 3560235
error_download | 1332522
available | 16665606
not_found | 2248861
new | 10387
(7 rows)

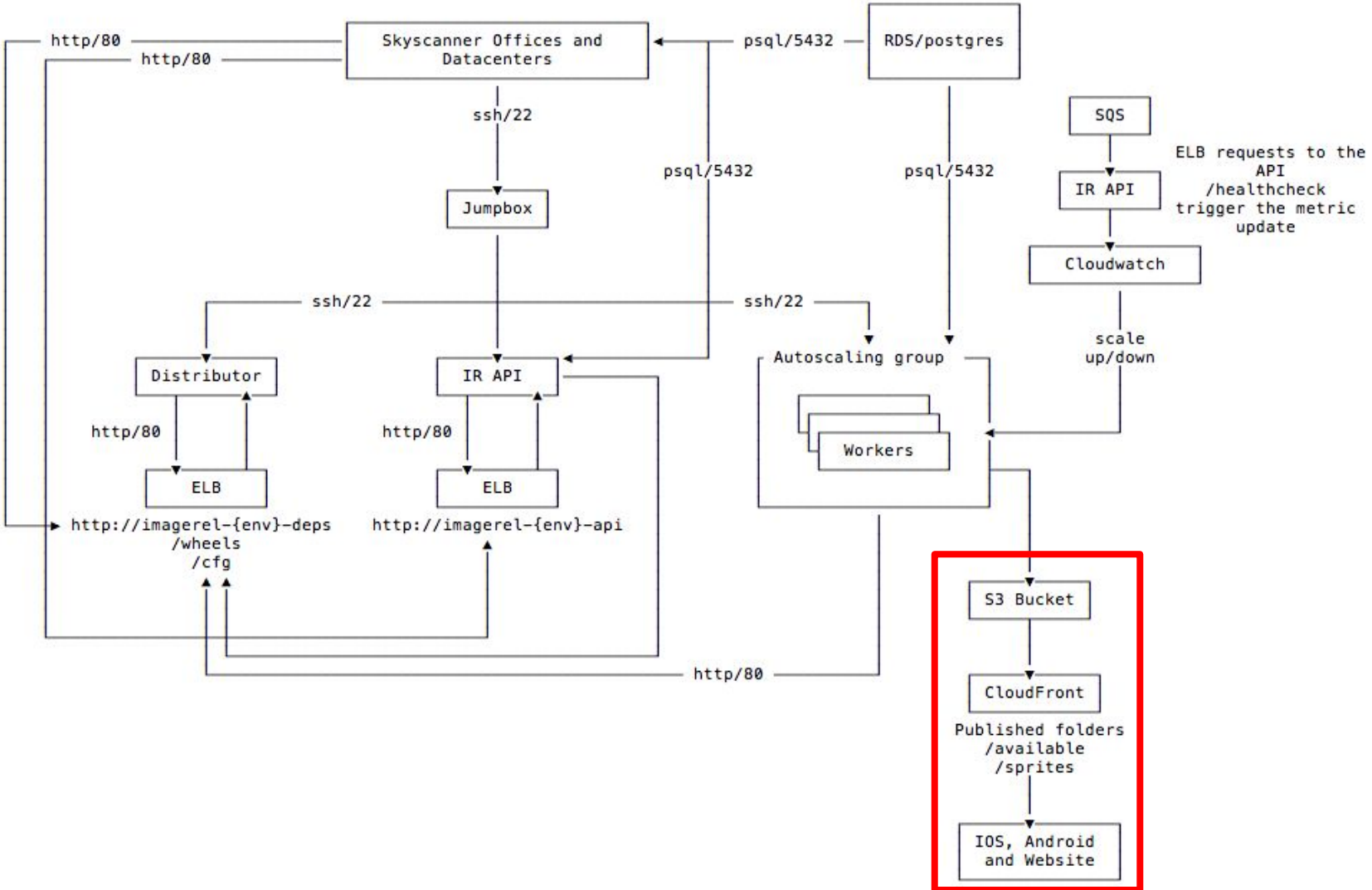

http/80
http/80
Skyscanner Offices and Datacenters
psql/5432
RDS/postgres
ssh/22
psql/5432
psql/5432
SQS
ELB requests to the API /healthcheck trigger the metric update
IR API
Jumpbox
Cloudwatch
ssh/22
ssh/22
scale up/down
Distributor
IR API
Autoscaling group
http/80
http/80
Workers
ELB
ELB
http://imagerel-{env}-deps /wheels /cfg
http://imagerel-{env}-api
S3 Bucket
http/80
CloudFront
Published folders /available /sprites
IOS, Android and Website


http/80
http/80
Skyscanner Offices and Datacenters
psql/5432
RDS/postgres
ssh/22
psql/5432
psql/5432
SQS
ELB requests to the API /healthcheck trigger the metric update
IR API
Jumpbox
Cloudwatch
ssh/22
ssh/22
scale up/down
Autoscaling group
Distributor
IR API
http/80
http/80
Workers
ELB
ELB
http://imagerel-{env}-deps
/wheels
/cfg
http://imagerel-{env}-api
S3 Bucket
http/80
CloudFront
Published folders
/available
/sprites
IOS, Android and Website


http/80
http/80
Skyscanner Offices and Datacenters
psql/5432
RDS/postgres
ssh/22
SQS
psql/5432
psql/5432
IR API
LB requests to the API
/healthcheck
rigger the metric update
Jumpbox
Cloudwatch
ssh/22
ssh/22
scale up/down
Distributor
IR API
Autoscaling group
http/80
http/80
Workers
ELB
ELB
http://imagerel-{env}-deps
/wheels
/cfg
http://imagerel-{env}-api
S3 Bucket
http/80
CloudFront
Published folders
/available
/sprites
IOS, Android and Website


http/80
http/80
Skyscanner Offices and Datacenters
psql/5432
RDS/postgres
ssh/22
psql/5432
psql/5432
SQS
ELB requests to the API /healthcheck trigger the metric update
IR API
Jumpbox
Cloudwatch
ssh/22
ssh/22
scale up/down
Distributor
IR API
Autoscaling group
Workers
http/80
http/80
ELB
ELB
http://imagerel-{env}-deps /wheels /cfg
http://imagerel-{env}-api
S3 Bucket
http/80
CloudFront
Published folders /available /sprites
IOS, Android and Website


http/80
http/80
Skyscanner Offices and Datacenters
psql/5432
RDS/postgres
ssh/22
psql/5432
psql/5432
SQS
ELB requests to the API /healthcheck trigger the metric update
IR API
Jumpbox
Cloudwatch
ssh/22
ssh/22
scale up/down
Distributor
IR API
Autoscaling group
http/80
http/80
Workers
ELB
ELB
http://imagerel-{env}-deps /wheels /cfg
http://imagerel-{env}-api
S3 Bucket
http/80
CloudFront
Published folders /available /sprites
IOS, Android and Website


http/80
http/80
Skyscanner Offices and Datacenters
psql/5432
RDS/postgres
ssh/22
psql/5432
psql/5432
SQS
ELB requests to the API /healthcheck trigger the metric update
IR API
Jumpbox
Cloudwatch
ssh/22
ssh/22
scale up/down
Distributor
IR API
Autoscaling group
http/80
http/80
Workers
ELB
ELB
http://imagerel-{env}-deps /wheels /cfg
http://imagerel-{env}-api
S3 Bucket
http/80
CloudFront
Published folders /available /sprites
IOS, Android and Website

Thanks for listening

Any questions?