Team--**DataOmini**

Milestone 3 – DB Population & Query

AWS RDS MySQL Database Connection Credentials:

Username: irisma Host: database-1.cksccuvxnwr.us-east-1.rds.amazonaws.com Port: 3306 Password: koala0312
Database Name: DataOmni_NGA

Feature #1 – 1) Filter by:

#1 – Nationality

- Filter Artworks by Nationality of artists

```
` SELECT O.title, O.attribution, O.objectID, OI.thumbURL,
FROM objects O JOIN objects_constituents OC
                JOIN constituents C
                JOIN objects_images OI
ON O.objectID = OC.objectID AND OC.constituentID = C.constituentID
AND O.objectID = OI.objectID
WHERE C.visualBrowserNationality = '${Nationality}'
LIMIT ${offset}, ${limit}; `
```

#2 – Artist lastName

- Filter Artworks by lastName of artists

```
`SELECT O.title, O.attribution, O.objectID, OI.thumbURL
FROM objects O JOIN objects_constituents OC
                JOIN constituents C
                JOIN objects_images OI
ON O.objectID = OC.objectID AND OC.constituentID = C.constituentID
AND O.objectID = OI.objectID
WHERE C.lastName = '${ArtistLastName}'
LIMIT ${offset}, ${limit}; `
```

#3 – Style

- Filter Artworks by Style (i.e whether they are of 'Baroque', 'Impressionist', 'Romantic', 'Minimalist'…)

```
`SELECT O.title, O.attribution, O.objectID, OI.thumbURL
FROM objects O JOIN objects_images OI
                JOIN objects_terms OT
ON O.objectID = OI.objectID AND O.objectID = OT.objectID
WHERE OT.term = '${StyleName}' AND OT.termType = 'Style'
LIMIT ${offset}, ${limit}; `
```

#4 – School

- Filter Artworks by School (i.e whether they are of 'Italian', 'French', 'Dutch', 'American', 'Netherlandish', 'Czechoslovakian'….)

```
` SELECT O.title, O.attribution, O.objectID, OI.thumbURL
FROM objects O JOIN objects_images OI
                JOIN objects_terms OT
    ON O.objectID =OI.objectID AND O.objectID =OT.objectID
WHERE OT.term = '${SchoolName}' AND OT.termType = 'School'
LIMIT ${offset}, ${limit}; `
```

#5 – Technique

- Filter Artworks by Style (i.e whether they are of 'painted surface', 'punched design', 'etching', 'graphite', 'drypoint', 'aquatint'……)

```
` SELECT O.title, O.attribution, O.objectID, OI.thumbURL
FROM objects O JOIN objects_images OI
                JOIN objects_terms OT
```

```
    ON O.objectID =OI.objectID AND O.objectID =OT.objectID
WHERE OT.term = '${TechniqueName}' AND OT.termType = 'Technique'
LIMIT ${offset}, ${limit}; `
```

6 – TimeSpan

- Filter Artworks by their time of completion (marked by "endYear" attribute of "objects"-relation)

```
` SELECT O.title, O.attribution, O.beginYear, O.endYear, O.objectID, OI.thumbURL
FROM objects O JOIN objects_images OI ON O.objectID =OI.objectID
WHERE O.beginYear >= '${beginYear}' AND O.endYear <= '${endYear}'
LIMIT ${offset}, ${limit}; `
```

Feature #1 – 2) Specific keyword Search:

#7 – Search For Artist Name

- User enters the name of the artist to search
- We will modify the user's input Artist Name with the wildcard character **%**
- i.e. if user entered "`Picasso`", it will be modified into "`%Picasso%`", so that we can match as much art-objects with artists' name(s) that are relevant to this searching keyword

```
SELECT O.title, O.attribution, C.beginYear, O.endYear, O.objectID, OI.thumbURL
FROM objects O JOIN objects_constituents OC
                JOIN constituents C
                JOIN objects_images OI
    ON O.objectID = OC.objectID AND OC.constituentID = C.constituentID AND O.objectID =OI.objectID
WHERE (O.attribution LIKE '${ArtistName}' OR O.attributionInverted LIKE '${ArtistName}' OR
        C.lastName LIKE '${ArtistName}' OR C.preferredDisplayName LIKE  '${ArtistName}' OR
        C.forwardDisplayName LIKE  '${ArtistName}')
ORDER BY O.attribution, C.lastName, C.beginYear, O.title
LIMIT ${offset}, ${limit};
```

#8 – Search For Artwork Title

- Note: we will modify the user's input Artwork Title with the wildcard character **%**
- i.e. if user entered "`Portrait`", it will be modified into "`%Portrait%`", so that we can match as much art-objects with titles that are relevant to this searching keyword

```
` SELECT O.title, O.attribution, O.endYear, O.objectID, OI.thumbURL
FROM objects O JOIN objects_images OI ON O.objectID =OI.objectID
WHERE (O.title LIKE '${ArtworkName}')
ORDER BY O.title, O.attribution
LIMIT ${offset}, ${limit}; `
```

Feature #1 – 3) Specific Artwork by its unique objectID

#9 – Display Specific Artwork (on a single webpage)

- Every time user click on to browse a specific artwork in detail, these 3 queries will be executed together (atomically) to get the full information about this artwork, and user will be redirected to a webpage dedicated to display this artwork

Part 1: get the info about this artwork (will always return a single tuple)

```
SELECT O.title, O.attribution, O.medium, O.dimensions, O.classification, O.series, O.portfolio, O.volume, OI.URL
FROM objects O JOIN objects_images OI ON O.objectID = OI.objectID
WHERE O.objectID = ${objectID};
```

Part 2: get the info of its composing artists (may return 1..* tuples)

```
SELECT C.preferredDisplayName,OC.displayOrder, C.beginYear, C.endYear, C.visualBrowserNationality
FROM objects_constituents OC JOIN constituents C ON OC.constituentID = C.constituentID
WHERE OC.objectID = ${objectID}
ORDER BY displayOrder;
```

Part 3: get the related semantic terms to this artwork (may return 1..* tuples)

```
SELECT OT.termType, OT.term
FROM objects_terms OT
WHERE OT.objectID = ${objectID}
ORDER BY termType;
```

Team--**DataOmini**

#10 – Similarity

- On the bottom of any artwork detailed-display page, we will show some recommended artworks (by similarity)
- We will first try the query for "Primary Recommendation", if get back less than 4 similar artworks, we will initiate the query for "Secondary Recommendation"
  - **Primary Recommandation:** find similar artworks based on similar portfolio, series, volume or composing artist

```
SELECT DISTINCT O.title, O.attribution, O.objectID, OI.thumbURL, O.series, O.portfolio, O.volume
FROM objects O JOIN objects_constituents OC
        JOIN constituents C
        JOIN objects_images OI
    ON O.objectID = OC.objectID AND
       OC.constituentID = C.constituentID AND
       O.objectID = OI.objectID
WHERE (O.objectID <> ${objectID} ) AND (O.classification = ) AND
      ( (O.portfolio LIKE '${portfolioName}') OR
        (O.series LIKE '${seriesName}') OR
        (O.volume LIKE '${volumnName}') OR
        (C.constituentID = ${constituentID}) )
ORDER BY O.series, O.portfolio, O.volume, O.attribution
LIMIT 4;
```

  - **Secondary Recommandation**: if no serie/volume/portfolio/artist in common, consider recommendation by Keyword, School, Style, Theme

```
SELECT O.title, O.attribution, O.objectID, OI.thumbURL, OT.termType, O.series, O.portfolio, O.volume
FROM objects O JOIN objects_images OI JOIN objects_terms OT
    ON O.objectID = OI.objectID AND O.objectID = OT.objectID
WHERE (O.objectID <> ${objectID} ) AND (O.classification = ${artworkClassification}) AND (
      (OT.termType = 'Style' AND OT.term = '${atrworkStyle}') OR
      (OT.termType = 'School' AND OT.term = '${artworkSchool}') OR
      (OT.termType = 'Keyword' AND OT.term = '${artworkKeyword}') OR
      (OT.termType = 'Theme' AND OT.term = '${artworkTheme}') )
ORDER BY termType
LIMIT 4;
```

Feature #2 – Naughty Search

#11 – User's Height

- Search by User's Height: User enters his/her height, we return a set of artworks that is around the similar height
- we use the ABS() function of MySQL, to rank and return the height of the artworks from the closest to the furthest to the user's height

```
` SELECT O.title, O.attribution, O.objectID, OI.thumbURL, OD.dimension, ABS( ${userHeight} - OD.dimension)
FROM objects O JOIN objects_images OI JOIN objects_dimensions OD
    ON O.objectID =OI.objectID AND O.objectID = OD.objectID
WHERE O.classification = '${artworkClassification}' AND
        OD.dimensionType = 'height' AND OD.unitName = 'centimeters'
ORDER BY ABS( ${userHeight} - OD.dimension), O.title
LIMIT ${offset}, ${limit}; `
```

#12 – BirthYear

- Search by User's BirthYear: User enters his/her year of birth, we return a set of artworks that was completed in this birthYear. Or user can also specify if he/she would like to see artworks that were completed 1 century (100 years), 2, 3…centuries ago
- we use the ABS() function of MySQL, to rank artworks based on their the completion years from the closest to the furthest to user's birthYear

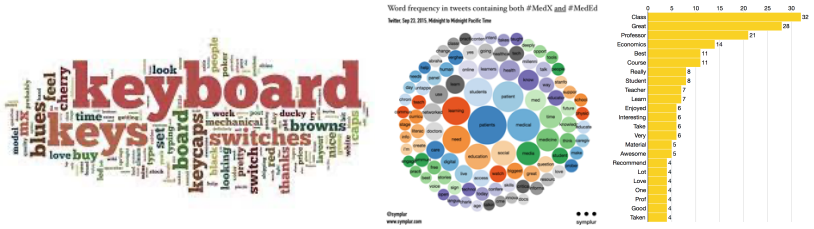
```
` SELECT O.title, O.attribution, O.objectID, O.endYear, ABS(${birthYear}-${yearsAgo}-O.endYear), OI.thumbURL,
OD.dimension
FROM objects O JOIN objects_images OI JOIN objects_dimensions OD
    ON O.objectID =OI.objectID AND O.objectID = OD.objectID
WHERE O.endYear IS NOT NULL AND OD.dimensionType = 'height'
ORDER BY ABS(${birthYear}-${yearsAgo}-O.endYear), OD.dimension DESC
LIMIT ${offset}, ${limit}; `
```

Team--**DataOmini**

Feature #3 – Semantic Analysis

#13 – Quantitative Semantic Analysis

- We will use the following queries to generate a “word frequency graph”, or “world frequency chart”
- graph or chart would look like this (just for idea demostration):



* https://www.reddit.com/r/MechanicalKeyboards/comments/2afcf8/meta_word_frequency_chart_for_rmechanicalkeyboards/
* https://twitter.com/tmlfox/status/647033025897730048
* https://cagrimmett.com/til/2016/06/22/frequency-for-better-word-clouds/

- the analysis result will contain some embedded hyperlinks, which lead the user to see the query results of relavent artworks

13.1) Semantic Analysis: Style

```
SELECT OT.term, COUNT(*) AS StyleCounts
FROM objects O JOIN objects_terms OT ON O.objectID = OT.objectID
WHERE OT.termType = 'Style'
GROUP BY OT.term
ORDER BY COUNT(*) DESC;
```

13.2) Semantic Analysis: School

```
SELECT OT.term, COUNT(*) AS SchoolCounts
FROM objects O JOIN objects_terms OT ON O.objectID = OT.objectID
WHERE OT.termType = 'School'
GROUP BY OT.term
ORDER BY COUNT(*) DESC;
```

13.3) Semantic Analysis: Technique

```
SELECT OT.term, COUNT(*) AS TechniqueCounts
FROM objects O JOIN objects_terms OT ON O.objectID = OT.objectID
WHERE OT.termType = 'Technique'
GROUP BY OT.term
ORDER BY COUNT(*) DESC;
```

13.4) Semantic Analysis: Theme

```
SELECT OT.term, COUNT(*) AS ThemeCounts
FROM objects O JOIN objects_terms OT ON O.objectID = OT.objectID
WHERE OT.termType = 'Theme'
GROUP BY OT.term
ORDER BY COUNT(*) DESC;
```

13.5) Semantic Analysis: Keyword

```
SELECT OT.term, COUNT(*) AS KeywordCounts
FROM objects O JOIN objects_terms OT ON O.objectID = OT.objectID
WHERE OT.termType = 'Keyword'
GROUP BY OT.term
ORDER BY COUNT(*) DESC;
```

13.6) Semantic Analysis: Place Executed

```
SELECT OT.term, COUNT(*) AS PlaceExecutedCounts
FROM objects O JOIN objects_terms OT ON O.objectID = OT.objectID
WHERE OT.termType = 'Place Executed'
GROUP BY OT.term
ORDER BY COUNT(*) DESC;
```

Team--**DataOmini**

#14 – Portrait Across Time

- To get portraits of a certain time period, we ask user to enter:
    - 1) the upper-bound year and lower-bound year of the time period; AND
    - 2) Which t ype (i.e. paintings, drawings, or prints) of artwork the user would like to check
- Then we query the collection of artwork based on these specifications:

```
` SELECT O.title, O.attribution, O.classification, O.objectID, OI.thumbURL, O.endYear
        FROM objects O JOIN objects_images OI JOIN objects_terms OT
        ON O.objectID = OI.objectID AND O.objectID = OT.objectID
        WHERE OT.visualBrowserTheme = ‘portrait’ AND
                (O.endYear <= ${lowerYear} AND O.endYear >= ${upperYear}) AND
                classification = '${artworkClassification}'
        ORDER BY O.endYear
        LIMIT ${offset}, ${limit}; `
```

- We will use this big UNION-query to generate a general overview of artwork of portraits across 16th, 17th, 18th, 19th, 20th, 21th centuries

```
` (SELECT O.title, O.attribution, O.classification, O.objectID, OI.thumbURL, O.endYear
        FROM objects O JOIN objects_images OI JOIN objects_terms OT
        ON O.objectID =OI.objectID AND O.objectID = OT.objectID
        WHERE OT.visualBrowserTheme = 'portrait' AND ( O.endYear <= 1599 AND O.endYear >= 1500) AND classification =
'${artworkClassification}'
        ORDER BY O.endYear
        LIMIT 5)
 UNION
(SELECT O.title, O.attribution, O.classification,O.objectID, OI.thumbURL, O.endYear
        FROM objects O JOIN objects_images OI JOIN objects_terms OT
        ON O.objectID =OI.objectID AND O.objectID = OT.objectID
        WHERE OT.visualBrowserTheme = 'portrait' AND ( O.endYear <= 1699 AND O.endYear >= 1600) AND classification =
‘${artworkClassification}’
        ORDER BY O.endYear
        LIMIT 5)
UNION
(SELECT O.title, O.attribution, O.classification, O.objectID, OI.thumbURL, O.endYear
        FROM objects O JOIN objects_images OI JOIN objects_terms OT
        ON O.objectID =OI.objectID AND O.objectID = OT.objectID
        WHERE OT.visualBrowserTheme = 'portrait' AND ( O.endYear <= 1799 AND O.endYear >= 1700) AND classification =
‘${artworkClassification}’
        ORDER BY O.endYear
        LIMIT 5)
UNION
(SELECT O.title, O.attribution, O.classification, O.objectID, OI.thumbURL, O.endYear
        FROM objects O JOIN objects_images OI JOIN objects_terms OT
        ON O.objectID =OI.objectID AND O.objectID = OT.objectID
        WHERE OT.visualBrowserTheme = 'portrait' AND ( O.endYear <= 1899 AND O.endYear >= 1800) AND classification =
‘${artworkClassification}’
        ORDER BY O.endYear
        LIMIT 5)
UNION
(SELECT O.title, O.attribution, O.classification, O.objectID, OI.thumbURL, O.endYear
        FROM objects O JOIN objects_images OI JOIN objects_terms OT
        ON O.objectID =OI.objectID AND O.objectID = OT.objectID
        WHERE OT.visualBrowserTheme = 'portrait' AND ( O.endYear <= 1999 AND O.endYear >= 1900) AND classification =
‘${artworkClassification}’
        ORDER BY O.endYear
        LIMIT 5)
UNION
(SELECT O.title, O.attribution, O.classification, O.objectID, OI.thumbURL, O.endYear
        FROM objects O JOIN objects_images OI JOIN objects_terms OT
        ON O.objectID =OI.objectID AND O.objectID = OT.objectID
        WHERE OT.visualBrowserTheme = 'portrait' AND ( O.endYear <= 2099 AND O.endYear >= 2000) AND classification =
‘${artworkClassification}’
        ORDER BY O.endYear
        LIMIT 5);
```