DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

Databases Project – Spring 2020

Team No: 42

Members: Wenuka Gunarathna, Marouane Jaakik, Jérôme Lüscher

Contents

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

Deliverable 1

Assumptions

1. It is more spatially efficient to have all opening times for each weekday than having it specified for each business as there will be lots of businesses than the 15-minute time intervals per week.
2. A business has to be classified in at least one category
3. A business should possess opening hours
4. The friends of the user should be users present inside our sample of the dataset
5. The same address and postal code combination cannot occur twice

Entity Relationship Schema

Schema

The Entity-Relationship diagram is shown in the Figure1 below.

Description

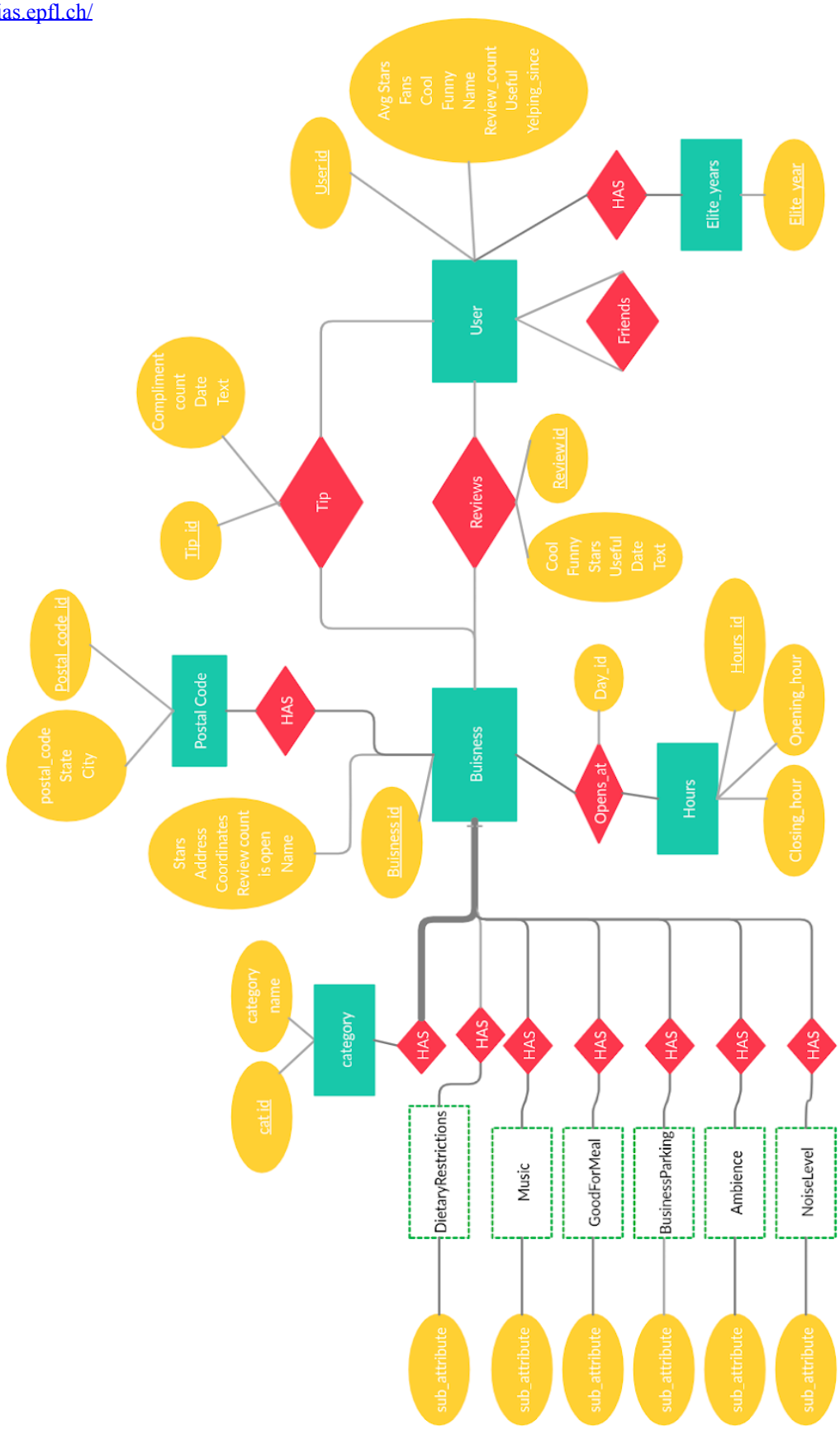
The business entity is the main component we identified in the Yelp-data set. To uniquely identify each business, we used “Business_id” as the primary key of the “Business” entity.

To identify the opening hours of each business, we came up with an “Opening hours” entity, which will hold all possible times (with 15-minute intervals) for each week-day with an id. Therefore the “Opens_at” relation will hold the business id with the link to Opening hours for both opening and closing times. Further, this will be a weak entity as opening hours to exist for a business, a business should exist first.

Businesses can be reviewed by Users as well as Users can give Tips about the Businesses. Those users can have friends, who are other Users from the dataset. Furthermore, a User can be an elite member for multiple years and that is shown using an “ISA” relationship.

Regarding Business Attributes, each main attribute will be stored as an Enum while each sub-attribute will be stored in the form of an integer. To map these integers with their proper human-readable names, we will use a separate table (not drawn in the ER diagram above). The business attribute will also be a weak entity as in order for a category to exist for a business, a business should exist first. The business categories can be one of the thousands. Therefore we do not use an Enum format for the categories.

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/



DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

Relational Schema

ER schema to Relational schema

Translating ER schema to Relational schema was straight forward.

DDL

Please refer the Appendix 1 for the complete DDL used in this project.

General Comments

Initial DDL and the ER diagram submitted was changed according to the feedback we got. The ER diagram and DDL commands attached herewith are the updated versions after the feedback.

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

Deliverable 2

Assumptions

The postal code ids were created for all the businesses if at least one attribute out of city, state or postal code is present.

Data Loading/Cleaning

Data loading has been done directly using import function in Oracle Sql Developer for the tables BUSINESS, REVIEWS, TIPS and USERS.

The POSTAL_CODE has been filled using a seperate table name Temp_business with business_id, city, state and the postal_code columns. The following queries were used to load the data to it and update the new postal_code_id to the business table.

SQL statement to update postal_code table

```
INSERT INTO POSTAL_CODE (CITY, POSTAL_CODE, STATE) select distinct postal_code,city,state from
temp_business;
```

SQL statement to update postal_code_id in business table

```
UPDATE
( select postal_code.postal_code_id as new_pid, Temp_Business.business_id  as up_b_id,
business.postal_code_id as old_pid
from business join Temp_Business on business.business_id=Temp_Business.business_id
inner join postal_code on Temp_Business.city=postal_code.city and
Temp_Business.postal_code=postal_code.postal_code and Temp_Business.state=postal_code.state
) t
SET t.old_pid = t.new_pid;
```

Further a separate script was written to exclude special characters causing issues in the csv file containing reviews.

For all other tables, a python script was written using Pandas Dataframes.

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

Query Implementation

Query 1:

Description of logic:

selecting the average reviews per user

SQL statement

```
Select
                AVG(Review_Count)
          FROM
                Users
```

Query result

AVG(REVIEW_COUNT)

37.652487443026465001650290052924866210 92
Took 0.069 seconds

Query 2:

Description of logic:

the number of businesses in the provinces of Québec and Alberta, we make sure that the postal code id of the 2 tables match

SQL statement

```
Select
              count(*)
           FROM
              Business B, Postal_code P
           WHERE
              P.postal_code_id = B.postal_code_id AND (P.state = 'AB' OR P.state =
           'QC')
```

Query result

COUNT(*)

17126
Took 0.12 seconds

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

Query 3:

Description of logic:

the maximum number of categories assigned to a business , the table is ordered in descending order on the field category count and only the first row is selected.

SQL statement

```
select count(cat_id) as cat_count,business_id from category group by business_id order by
cat_count desc fetch first 1 rows only;
```

Query result

```
CAT_COUNT | BUSINESS_ID
----------|--------------------
37        | IjsLANGkmAqCsF6-zgIA8w
```

Took 0.113 seconds

Query 4:

Description of logic:

Number of businesses that are labelled as "Dry Cleaners" or “Dry Cleaning”, we match the category id and the category name thanks to the category map.

SQL statement

```
select
        Count(Business_id)
    From
        Category_map mapi , CATEGORY cat
    Where
    mapi.cat_id = cat.cat_id and mapi.cat_name = 'dry cleaning' or mapi.cat_name =
    'dry cleaners'
```

Query result

```
COUNT(BUSINESS_ID)
--------------------------
          436
```

Took 0.09 seconds

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

Query 5:

Description of logic:

The overall number of reviews for all the businesses that have more than 150 reviews and least any 2 dietary restriction categories. We only consider the business ids of the businesses present at least once in the dietary restrictions table.

SQL statement

```
select

        SUM(Review_Count)
    FROM
        Business B,
        Attr_DietaryRestrictions A
    where
        B.Review_Count > 150 AND B.Business_id IN ( Select business_id from
    Attr_DietaryRestrictions
       group by business_id having count(business_id)>1)
```

Query result

```
SUM(REVIEW_COUNT)
--------------------------------
          21164
Took 0.077 seconds
```

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

Query 6:

Description of logic:

Displaying the user id and the number of friends of the top 10 users by number of friends, and ordering the results by the number of users descending (the user with the highest number of friends first).
Inhere we select count for for each user id in both columns from friends table and then group by to add the count. Later we select the maximum friends count and fetch the top ten rows.

SQL statement

```
with
userFriends(friendsC
nt, userId) as
                        (
                        (select count(*) as countV, user_id1 as user_id from friends
                        group by (user_id1))
                           UNION
                           (select count(*) as countV, user_id2 as user_id from
                           friends group by (user_id2))
                        )
                        select sum(friendsCnt) as fullcount, userId from userFriends
                        group by user_id order by fullcount desc FETCH FIRST 10 ROWS
                        ONLY;
```

Query result

FULLCOUNT	USERID
4919	8DEyKVyplnOcSKx39vatbg
4603	Oi1qbcz2m2SnwUeztGYcnQ
4597	ZlOCmdFaMIF56FR-nWr_2A
4437	yLW8OrR8Ns4X1oXJmkKYgg
4222	YttDgOC9AlM4HcAlDsbB2A
4211	djxnI8Ux8ZYQJhiOQkrRhA
4134	qVc8ODYU5SZjKXVBgXdI7w
4067	iLjMdZi0Tm7DQxX1C1_2dg
3943	F_5_UNX-wrAFCXuAkBZRDw
3923	MeDuKsZcnI3IU2g7OlV-hQ

Took 6.026 seconds

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

Query 7:

Description of logic:

Showing the business name, number of stars, and the business review count of the top-5 businesses based on their review count that are currently open in the city of San Diego, this query displays more results for other cities such as Toronto, Calgary and Montreal.

SQL statement

```
select

        Name, Stars, Review_Count
FROM
        Business B, Postal_code P
where
       B.is_open = 1 AND P.city= 'San diego' AND B.postal_code_id = P.postal_code_id
order by Review_Count DESC fetch first 5 rows only
```

Query result

NAME	STARS	REVIEW_COUNT
Brooks Photography	1.5	35

Took 0.099 seconds

Query 8:

Description of logic:

Show the state name and the number of businesses for the state with the highest number of businesses. using indexes to speed up the lookup (in the unique constraint from the DDL)

SQL statement

```
select

        State,
        COUNT(Business_id) as compte
FROM
        Postal_code P, Business B
Where B.postal_code_id = P.postal_code_id
GROUP BY P.State order by compte DESC fetch first 1 rows only
```

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

Query result

STATE	COMPTE
AZ	56493

Took 0.032 seconds

Query 9:

Description of logic:

The total average of "average star" of elite users, grouped by the year in which they started to be elite users. Display the required averages next to the appropriate years.
In here we select the users and first year they get selected as an Elite user from the sub-query A and then join it with the user table to select the stars creating the sub-query Star_table. Then we simpy group by the elite year and take the average.

SQL statement

```
select Star_table.elite_year,
avg(Star_table.stars) as
avg_stars_for_the_year
                                        from (select A.e_year as elite_year,
                                        U.avg_stars as stars
                                        from (select user_id, min(elite_year) as
                                        e_year from elite group by (user_id)
                                        )A
                                        join users U on U.user_id=A.user_id
                                        )Star_table group by (Star_table.elite_year)
```

Query result

ELITE_YEAR	AVG_STARS_FOR_THE_YEAR
2011	3.779149117209104453775792384599021484 79
2014	3.862611683848797257509601778855872245 81
2009	3.740818340104468957515960533952408589 67
2007	3.772474489795918374553571428571428571 43
2010	3.757612061939690310941320293398533007 33
2008	3.761238798102266746125461254612546125 46
2006	3.807508417508417518406285072951739618 41
2015	3.889616685456595271183765501691093573 84
2012	3.789392482517482523819930069930069930 07

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

2017	3.97701380897583429784234752589182968 93
2016	3.93532216638749302668900055834729201563
2018	4.01841752398566640228875274534735868686
2013	3.82695691405368825249266862170087976 54

Took 0.865 seconds

Query 10:

Description of logic:

List the names of the top-10 businesses ordered by the median "star" rating, that are currently open in the city of New York.

SQL statement

```
Select
name
            from Business B
            Where B.is_open = 1 and Business_id IN (
            Select
                Business_id
            From
                Reviews
            Group by
                Business_id
            Order by
                PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY stars DESC) fetch first
            10 rows only )
```

Query result

NAME

Adore Organic Innovation
Q's Nails
Sorrento Villas
Ted Wiens Tire & Auto
Sandcastle Water Park
Showtime Tours
Great Clips

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

Avis Car
Domino's Pizza
Advantage Rent-A-Car

Took 3.499 seconds

Query 11:

Description of logic:

Displaying the minimum, maximum, mean, and median number of categories per business.

SQL statement

```
select

        AVG(COUNT(cat_id)) AS averages,
        MAX(COUNT(cat_id)) AS maxi,
        MIN(COUNT(cat_id)) AS mini ,
        PERCENTILE_CONT(0.5)WITHIN GROUP(ORDER BY COUNT(cat_id) DESC) as
        medianito
        FROM
        Category cat
        group by cat.business_id
```

Query result

AVERAGES	MAXI	MINI	MEDIANITO
4.102911095265111098388045407464853976 8	37	1	4

Took 0.325 seconds

Query 12:

Description of logic:

Displaying the businesses (show 'name', 'stars', 'review count') in the city of Las Vegas possessing 'valet' parking and open between '19:00' and '23:00' hours on a Friday.

In this query we were joining each table with given conditions to select the business_ids which meets the given conditions, then selecting the name, stars and review count attributes for those business_ids.

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

SQL statement

```
SELECT name,stars,
review_count
                                    FROM business b
                                    JOIN postal_code p
                                    ON (b.postal_code_id =
                                    p.postal_code_id)
                                    JOIN attr_businessparking a
                                    ON (a.business_id = b.business_id)
                                    JOIN open_at o
                                    ON (o.business_id = b.business_id)
                                    where p.city = 'Las Vegas'
                                    AND a.sub_attr_id = 1
                                    AND o.day_id=5
                                    AND o.hours_id <= 19*60000
                                    AND MOD(o.hours_id , 10000) >= 23*60
```

Query result

NAME	STARS	REVIEW_COUNT
Paradise Spa	4	10

Took 4.435 seconds

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

General Comments

We updated our ER model firstly by taking into account the feedback for milestone 1.

We did so by removing the ISA relationship for Elite, creating weak_entities for the State and Postal code.

We mapped the sub_attributes for each attribute to integers and we then created tuples of business_ids and sub_attributes_ids stored in 6 different tables, one for each attribute. That way, we are able to store only the necessary values, we do not store anything for a business that does not possess a certain sub_attribute. Also, the mapping to integers significantly reduces the storage needed.

Most queries had straightforward implementation and thus needed no detailed logic description as it can be completely understood from the sql statement itself.

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

Deliverable 3

Assumptions

The counts given in the CSV files such as review_count and stars in business csv file and avg_stars and review_count in users csv file are accurate. Even though the data consist of the given csv doesn't always reflect the above mentioned values, it was assumed the dataset given was not complete while the given values are correct when considering the full dataset.

Query 1:

Description of logic:

What is the total number of businesses in the province of Ontario that have at least 6 reviews and a rating above 4.2?

We first join the two tables to create a single table which contains both the information on the states and the information of the ratings of the businesses, then we simply select the businesses in Ontario and having the required rating. Finally we perform a count on the number of results obtained.

SQL statement

```
select

        count(Business_id) as total
    from
        Business B,
        Postal_Code P
    Where
      B.Postal_Code_id = P.postal_code_id and P.State= 'ON' and review_count > 5 and
    stars > 4.2
```

Query result

```
TOTAL
-----------
2882

Took  0.133 seconds
```

Query 2:

Description of logic:

What is the average difference in review scores for businesses that are considered "good for dinner" that have noise levels "loud" or "very loud", compared to ones with noise levels "average" or "quiet"?

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

SQL statement

```
SELECT
BB2.avg_star-
BB1.avg_star
                    FROM
                       (SELECT  AVG(B1.stars) as avg_star
                           FROM business B1 JOIN attr_goodformeal G1 ON
                    b1.business_id=g1.business_id JOIN attr_noiselevel N1 ON
                    b1.business_id=n1.business_id
                           where g1.sub_attr_id IN (SELECT sub_attr_id FROM
                    attr_goodformeal_map WHERE sub_attr_name like 'dinner')
                           AND n1.sub_attr_id IN (SELECT sub_attr_id FROM
                    attr_noiselevel_map WHERE sub_attr_name like '%loud')
                       )BB1,
                       (SELECT  AVG(B1.stars) as avg_star
                        FROM business B1 JOIN attr_goodformeal G1 ON
                    b1.business_id=g1.business_id JOIN attr_noiselevel N1 ON
                    b1.business_id=n1.business_id
                        where g1.sub_attr_id IN (SELECT sub_attr_id FROM
                    attr_goodformeal_map WHERE sub_attr_name like 'dinner')
                        AND (n1.sub_attr_id IN (SELECT sub_attr_id FROM
                    attr_noiselevel_map WHERE sub_attr_name like 'average')
                        OR n1.sub_attr_id IN (SELECT sub_attr_id FROM
                    attr_noiselevel_map WHERE sub_attr_name like 'quiet')
                       ))BB2;
```

Query result

BB2.AVG_STAR-BB1.AVG_STAR

0.309531722381320155923569200766403984 21

Took 0.488 seconds

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

Query 3:

Description of logic:

List the "name", "star" rating, and "review_count" of the businesses that are tagged as "Irish Pub" and offer "live" music

In this we first select the sub_attribute_id's which has sub_attribute like 'live' and also cat_ids which has 'irish pub'. Then we select the business_id's which has both those above attribute and category and selecting the "name", "star" rating, and "review_count" attributes for those business_id's.

Here we have used upper() function to convert every cat_name to upper case to increase the search results. However it was not necessary for the attribute name as we designed the database.

SQL statement

```
SELECT name,stars,review_count FROM
business b JOIN attr_music m ON
b.business_id = m.business_id
                                    JOIN category c ON b.business_id =
                                    c.business_id
                                    WHERE m.sub_attr_id IN (SELECT sub_attr_id
                                    FROM attr_music_map WHERE sub_attr_name
                                    like 'live')
                                    AND c.cat_id IN (SELECT cat_id FROM
                                    category_map WHERE UPPER(cat_name) like
                                    'IRISH PUB');
```

Query result

NAME	STARS	REVIEW_COUNT
Tim Finnegan's Irish Restaurant and Pub	4.5	110
Irish Wolfhound	4	359
McFadden's Restaurant and Saloon	2	448
Hennessey's Tavern	3	384
Whelan's Gate Irish Pub	3	27
Blarney's Gate	4	4
Rosie McCaffrey's Irish Pub & Restaurant	3.5	326
Fibber Magees	3.5	246
The Skeptical Chymist	3.5	380
Murphy's Tap Room	3.5	13
McMullan's Irish Pub	4.5	519
Grumpy's Bar	4	29

Took 0.142 seconds

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

Query 4:

Description of logic:

looking up the average number of attribute “useful” of the users whose average rating falls in the following 2 ranges: [2-4), [4-5]

SQL statement

```
Select  avg(review_count) as
avgEL24_avgEL45_avgreg24_avgreg45
From
   Users U , Elite E
where
U.user_id = E.user_id and U.avg_stars >= 2
and U.avg_stars <4  union
Select  avg(review_count)
From
   Users U , Elite E
where
U.user_id = E.user_id and  U.avg_stars>=4
and U.avg_Stars <= 5 union
Select  avg(review_count)
From
   Users U left join Elite E on U.user_id
= E.user_id
where
E.user_id is null and U.avg_stars >= 2 and
U.avg_stars <4  union
Select  avg(review_count)
From
   Users U left join Elite E on U.user_id
= E.user_id
where
E.user_id is null and U.avg_stars>=4 and
U.avg_stars <= 5
```

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

Query result

```
AVGEL24    AVGEL45    AVGreg24    AVGreg45
-----------|----------|-----------|----------
14.1261    27.008024  264.2452    364.9278416
```

Took 0.99 seconds

Query 5:

Description of logic:

Find the average rating and number of reviews for all businesses which have at least two categories and more than (or equal to) one parking type.

SQL statement

```
SELECT

        AVG(B.stars) as average_rating, AVG(B.review_count) as
    average_review_count
    FROM
        Business B
    WHERE
        B.Business_id IN(SELECT business_id FROM attr_businessparking
            group by business_id having count(business_id)>=1)
        AND B.Business_id IN(SELECT business_id FROM Category
            group by business_id having count(business_id)>1)
```

Query result

```
AVERAGE_RATING      AVERAGE_REVIEW_COUNT
----------------------  |  ---------------------------------
3.70928597128           72.8708828050
```

Took 0.178 seconds

Query 6:

Description of logic:

What is the fraction of businesses (of the total number of businesses) that are considered "good for late night meals"?

Here we selected the count for the businesses which are considered good for late night meals and then divide it by the total count of businesses.

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

SQL statement

```
SELECT
A.late_meal_count/B.total_count AS
percentage
FROM
                                        (SELECT DISTINCT COUNT(*) AS late_meal_count
                                        FROM business b JOIN attr_goodformeal a ON
                                        b.business_id = a.business_id
                                        WHERE a.sub_attr_id IN ( SELECT sub_attr_id FROM
                                        attr_goodformeal_map where sub_attr_name like
                                        'latenight%' )
                                        )A
                                        JOIN
                                        (SELECT count(*) as total_count from business)B
                                        ON 1=1;
```

Query result

PERCENTAGE

0.010388922636013893431771101038892263601 4

Took 0.068 seconds

Query 7:

Description of logic:

Find the names of the cities where all businesses are closed on Sundays, the basic idea behind our implementation is that we intersect two tables of total number of businesses per city and total number of closed business per city on the requirement that the total closed = total businesses.

SQL statement

```
Select
q2.city
            from
             (Select
                city, Count(B.business_id) as closed
             from
               Business B Inner Join Open_at O on O.business_id=B.Business_id  Inner
            join Postal_code P on P.postal_code_id
```

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

```
  =B.postal_code_id
Where
   O.Day_id!=0
group by
   city ) q1
inner join (Select
   city, Count(B.business_id) as total
from
  Business B Inner Join Open_at O on O.business_id=B.Business_id  Inner
join Postal_code P on P.postal_code_id
  =B.postal_code_id
group by
   city )q2  on (q1.closed = q2.total and q1.city=q2.city)
```

Query result

CITY

Auburn Twp
moon
Phx
Old Scottsdale
Brooklyn Heights
Bradfordwoods
Saint-Basile-Le-Grand
Moon Twp
Yorkville
Westview
Lancaster
Tremont
South Hills
Ahwahtukee
Scotesdale
CAVE CREEK
Montréal-West
Lakewood, Oh

Took 12.141 seconds

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

Query 8:

Description of logic:

Find the ids of the businesses that have been reviewed by more than 1030 unique users

SQL statement

```
SELECT

   R.Business_id

FROM

   Reviews R

   GROUP BY Business_id having COUNT(DISTINCT(user_id))>
1030
```

Query result

BUSINESS_ID

4JNXUYY8wbaaDmk3BPzlWw
RESDUcs7fliihp38-d6_6g

Took 1.873 seconds - 0.454 after adding an index

Query 9:

Description of logic:

Find the top-10 (by the number of stars) businesses (business name, number of stars) in the state of California.

In here select businesses from California and ordered them by stars and then selected the first 10 businesses.

SQL statement

```
SELECT name,stars FROM business WHERE
business.postal_code_id IN (SELECT postal_code_id FROM
POSTAL_CODE WHERE state='CA')
                                                    ORDER BY business.stars
                                                    desc FETCH FIRST 10 ROWS
                                                    ONLY
```

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

Query result

NAME	STARS
Goldeen Myofascial Release Therapy	5
Jaclyn Webb Healing	5
Beachside Tans	5
Finest-Edge Precision Sharpening Service	5
Fireplace Door Guy	5
Pretty Girl Lingo	5
Melody Events	5
Respclearance	5
Core Pest Solutions	4.5
Rebecca Vinacour Photography	4.5

Took 0.048 seconds

Query 10:

Description of logic:

Finding the top-10 (by number of stars) ids of businesses per state. To achieve that, we partition over the joint tables Businesses and Postal_Code by the attribute state, and for each of the latter we rank the businesses by stars fetching the first 10 rows.

SQL statement

```
WITH TOPTEN
AS (
                    SELECT Business_id, state, ROW_NUMBER()
                     over (
                         PARTITION BY state
                         order by stars
                     ) AS rank
                    from Business B left join Postal_code P on B.postal_code_id=
                P.postal_code_id
                )
                SELECT * from TOPTEN WHERE rank <= 10
```

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

Query result

BUSINESS_ID	STATE	RANK
sx5I_M3uJC5cFD1Ag50qyA	AB	1
G04Pvc4jGE4RCLlYr3-tBA	AB	2
jtOLRQ9TC99BzHEf5SYkiQ	AB	3
MvxLJ-yBj1TQxbr0LjaPtQ	AB	4
EErAwIyl6OQHyLzICnSBLQ	AB	5
kUEBUoKxJLhgs2Bp_gN_7w	AB	6
oz6IloSPWLxIEK__WQ0SgA	AB	7
GKkU1U84DLOaoN-NQfOuJg	AB	8
QqZQpcQsoLO0Yb8GI5OR0A	AB	9
xHNUYkPu9yXfi7wnfjOWng	AB	10
WwwYZakdSQM9174gdZdUIA	AK	1
W839iVmIb3dKrU_-eyOhQA	AK	2
SP_tfo1VdYlrXBG_6OMZdA	CA	3
dPlwOjYe6gFT-w85Qg6GeA	CA	4
lfK_nFK50w2Gr6IMZviyIQ	CA	5
f4SdDMDdsCspFFom9J-pqg	CA	6
t8xvlKk5axVYy2c2p70NRg	CA	7
Lbq36N-MFtn9ozAYs1ZiCA	CA	8
YMeWjOd1svHDGdDCKoiGgg	CA	9
gavl0UJkI0Z5Dzs_tHXQ9A	CA	10

Took 0.553 seconds

Query 11:

Description of logic:

Find and display all the cities that satisfy the following: each business in the city has at least two reviews.

SQL statement

```
select

        Distinct(p.city)
    FROM
        Business B
        LEFT JOIN Postal_code P ON (b.postal_code_id =
    p.postal_code_id)
    WHERE
```

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

```
b.business_id IN(SELECT business_id FROM Reviews
    group by business_id having count(business_id)>1);
```

Query result

CITY

Charlotte
las vegas
Surprise
Lakewood
Willoughby
Northfield Center Township
Cornelius
Turtle Creek
Laveen
Aurora
Tolleson
Monona
New Kensington
Canonsburg
Pickering

Took 2.259 seconds - only 0.636 seconds after indexing

Query 12:

Description of logic:

Find the number of businesses for which every user that gave the business a positive tip (containing 'awesome') has also given some business a positive tip within the previous day.
In here we first selecting all tips which has word “Awesome” in tip text with tip posted time casted to date. Then we join it with the same result but with a one-day shift to the posted date so that inner result will have all the users who has given tips in two consecutive days with word “Awesome” in it. Then we select the count of reviews for each businesses, then try to count businesses which has equal total tips counts and total such tips (positive tips with user tipped another business positive previous day).

SQL statement

```
SELECT
COUNT(*)
FROM
        (SELECT A.business_id as bid, count(*) as positive_count  FROM
        (SELECT CAST(posted_date AS DATE) as first_date, business_id ,user_id,
        tip_id FROM tips where UPPER(tip_text) like '%AWESOME%')A
```

```
INNER JOIN (SELECT posted_date-1 AS prv_date ,user_id, tip_id FROM tips
where UPPER(tip_text) like '%AWESOME%')  B
ON A.first_date=B.prv_date AND A.user_id=B.user_id group by
A.business_id)C
INNER JOIN (SELECT count(*) as full_count,business_id FROM tips where
UPPER(tip_text) like '%AWESOME%' group by business_id) D on
D.business_id=C.bid AND D.full_count=C.positive_count
```

Query result

COUNT()

239

Took 9.809 seconds

Query 13:

Description of logic:

Finding the maximum number of different businesses any user has ever reviewed.

SQL statement

```
Select
    count( Distinct Business_id) as reviewed
from
    Reviews R
group by
    User_id
order by reviewed DESC fetch first 1 rows
only
```

Query result

REVIEWED

793

Took 0.81 seconds

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

Query 14:

Description of logic:

What is the difference between the average useful rating of reviews given by elite and non-elite users?
We created two different tables for elite and non-elite users, for the elite users, we simply joined the table with the elite table, for the non-elite users, we did the same but all the entries were the user id was equal to null were selected.

SQL statement

```
SELECT
U1.avg_useful_el
-U2.avg_useful_el
                        FROM
                        (SELECT AVG(re.useful) as avg_useful_el FROM Reviews RE INNER
                        JOIN Elite E ON (re.user_id = e.user_id))U1,
                        (SELECT AVG(re.useful) as avg_useful_el FROM Reviews RE LEFT
                        JOIN Elite E ON (re.user_id = e.user_id) WHERE e.user_id IS
                        NULL)U2
                        ;
```

Query result

U1.AVG_USEFUL_EL-U2.AVG_USEFUL_EL

2.00180139367162924475469401

Took 0.684 seconds

Query 15:

Description of logic:

List the name of the businesses that are currently 'open', possess a median star rating of 4.5 or above, considered good for 'brunch', and open on weekends.
This query was only selecting businesses which satisfies given criterias. First matching the tables of interest and then selecting the entries that match the criterias given above.

SQL statement

```
SELECT
DISTINCT name
                    FROM business b
                    JOIN attr_goodformeal a ON b.business_id = a.business_id
```

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

```
JOIN open_at o ON (o.business_id = b.business_id)
where b.stars >= 4.5
AND b.is_open > 0
AND a.sub_attr_id IN ( SELECT sub_attr_id FROM attr_goodformeal_map
where sub_attr_name like 'brunch%' )
AND (o.day_id=6 OR o.day_id=0)
```

Query result

NAME

Ladera Taverna y Cocina
Knotts Family Diner
Lidet Cafe and Bakery
Tinh Tam Trai
Juice 'N' Go
Sunrise Donuts
Mana'ish Global Flatbread Cafe
TangTangTang 2
Nuro bistro
Worth Takeaway

Took 0.555 seconds

Query 16:

Description of logic:

List the 'name', 'star' rating, and 'review_count' of the top-5 businesses in the city of 'los angeles' based on the average 'star' rating that serve both 'vegetarian' and 'vegan' food and open between '14:00' and '16:00' hours. Note: The average star rating should be computed by taking the mean of 'star' ratings provided in each review of this business.

We first match the Business table with the postal code, open at and dietary restrictions table, then we only select the entries that match our criteria from this big table. Lastly, we order our results according to the business stars rating and we only select the top 5.

SQL statement

```
SELECT B2.name, B2.stars, B2.review_count
FROM (SELECT B.name, B.stars, B.review_count, O.hours_id, a.sub_attr_id
   FROM Business B
   INNER JOIN (Select P.postal_code_id From Postal_code P where UPPER(P.city) like '%LOS ANGELES%') P2
ON (P2.postal_code_id = B.postal_code_id)
```

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

```
    INNER JOIN attr_dietaryrestrictions a ON (b.business_id = a.business_id )
    INNER JOIN open_at O ON (O.business_id = B.business_id)
    WHERE a.sub_attr_id IN ( SELECT sub_attr_id FROM attr_dietaryrestrictions_map where
UPPER(sub_attr_name) IN ('%VEGETARIAN%', '%VEGAN%'))
    AND o.hours_id IN (SELECT H.hours_id FROM Hours H WHERE CAST(H.opening_time as FLOAT) <
CAST('14:00:00' as FLOAT)  AND CAST(H.closing_time as FLOAT) > CAST('16:00:00' as FLOAT))
    ) B2
ORDER BY B2.stars DESC FETCH FIRST 5 ROWS ONLY
;
```

Query result

Empty table

Took 0.077 seconds

Query 17:

Description of logic:

Compute the difference between the average 'star' ratings (use the reviews for each business to compute its average star rating) of businesses considered 'good for dinner' with a (1) "divey" and (2) an "upscale" ambience.

The end goal is to create two different tables that contain the divey and upscale businesses respectively and then compute their averages of star ratings. In order to do this, we matched the businesses in the business table with the ones in the goodformeal and ambience attribute tables, then we only selected the sub attributes ids that matched the string attributes we wanted for each table.

SQL statement

```
SELECT
AVG(rd.stars) -
AVG(ru.stars)

                    FROM Reviews RD, Reviews RU, attr_ambience A1, attr_ambience
                    A2, attr_goodformeal G1, attr_goodformeal G2


                    WHERE

                       g1.business_id = rd.business_id
```

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

```
   AND g2.business_id = ru.business_id
   AND a1.business_id = rd.business_id
   AND a2.business_id = ru.business_id
   AND g1.sub_attr_id IN ( SELECT sub_attr_id FROM
attr_goodformeal_map where sub_attr_name like 'dinner%')
   AND g2.sub_attr_id IN ( SELECT sub_attr_id FROM
attr_goodformeal_map where sub_attr_name like 'dinner%')
   AND a1.sub_attr_id IN ( SELECT sub_attr_id FROM
attr_ambience_map where sub_attr_name like 'divey%')
   AND a2.sub_attr_id IN ( SELECT sub_attr_id FROM
attr_ambience_map where sub_attr_name like 'upscale%');
```

Query result

```
AVG(RD.STARS)-AVG(RU.STARS)
---------------------------------------
0.172694989979925774
```

Took 14.875 seconds

Query 18:

Description of logic:

Find the number of cities that satisfy the following: the city has at least five businesses and each of the top-5 (in terms of number of reviews) businesses in the city has a minimum of 100 reviews.
This query was selecting cities which satisfies given criterias. First we select the cities with more than 100 reviews, then we join this table with a table B containing all the cities with more than 5 businesses.

SQL statement

```
SELECT
count(*)
             from (SELECT count(*)
                 from (SELECT business_id,review_count,postal_code_id from business
             where review_count >=100)A
                  join postal_code p on (p.postal_code_id = A.postal_code_id) group
             by p.city having count(*)>=5)B
```

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

Query result

COUNT

82

Took 0.066 seconds

Query 19:

Description of logic:

Finding the names of the cities that satisfy the following: the combined number of reviews for the top-100 (by reviews) businesses in the city is at least double the combined number of reviews for the rest of the businesses in the city. We intersect two tables of combined number reviews of rest per city with the top 100 reviews per city on the requirement that the latter is at least double the former.

SQL statement

```
with top100 as ( select
city,sum(review_count) as
sumtop from
                                  (SELECT Business_id,review_count, city,
ROW_NUMBER()
                                  over (
                                      PARTITION BY city
                                      order by review_count
                                  ) as rank
                                  from Business B inner join Postal_code P on
B.postal_code_id= P.postal_code_id)
                                  where rank <=100 group by city
),
bottom as ( select city,sum(review_count) as sumrest
from
                                  (SELECT Business_id,review_count, city,
ROW_NUMBER()
```

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

```
    over (
        PARTITION BY city
        order by review_count
    ) AS rank
    from Business B inner join Postal_code P on
B.postal_code_id= P.postal_code_id
    ) where rank>100 group by city)

select * from top100 topo inner join bottom botto on
topo.city=botto.city and
topo.sumtop>=botto.sumrest*2
```

Query result

CITY

Amherst
Sun City West
Euclid
Pointe-Claire

Took 3.717 seconds

Query 20:

Description of logic:

For each of the top-10 (by the number of reviews) businesses, find the top-3 reviewers by activity among those who reviewed the business. Reviewers by activity are defined and ordered as the users that have the highest numbers of total reviews across all the businesses (the users that review the most).
We first join the Users and the Business table, then we match the businesses in the reviews and the business tables. Then, we proceed to select the top 10 businesses with the most reviews, and among the users who reviewed these businesses, we select the top 3 most active users.

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

SQL statement

```
SELECT U.user_id,
U.review_count
                        FROM Business B, Reviews R
                        LEFT JOIN Users U ON(U.user_id = R.user_id)

                        WHERE
                           R.business_id = B.business_id
                           AND B.business_id IN (  SELECT business_id
                                                   FROM (SELECT b1.business_id FROM
Business B1 ORDER BY b1.review_count DESC)
                                                   WHERE ROWNUM < 11)
                           AND U.user_id IN(    SELECT user_id
                                                   FROM (SELECT u3.user_id FROM Users U3
ORDER BY u3.review_count DESC)
                                                   WHERE ROWNUM < 4);
```

Query result

USER_ID	REVIEW_COUNT
RtGqdDBvvBCjcu5dUqwfzA	12390
hWDybu_KvYLSdEFzGrniTw	10022

Took 0.636 seconds

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

Query Performance Analysis – Indexing

Optimized Query 1- Deliverable3 Query 1
Initial Running time/IO: 0.367
Optimized Running time/IO: 0.133
Explain the improvement:

We perform an inner Join on the Postal code table and the Business table in order to first produce a table which contains all the information matched on the postal_code_id before we proceed with the query instead of comparing the postal code id after the fact.

Optimized Query 2- Deliverable3 Query 7
Initial Running time/IO: 12.141 Seconds
Optimized Running time/IO: 7.955 Seconds
Explain the improvement: we have added an index as a unique constraint to the Business table on the attribute postal_code_id

Optimized Query 3- Deliverable3 Query 8
Initial Running time/IO: 1.873 Seconds
Optimized Running time/IO: 0.454 Seconds
Explain the improvement:

We have added an index to the review table for the User_id column and for the Business_id columns which made the time taken by 75.76%. This huge performance improvement is happening because the query uses group_by over business_id and counting distinct user_ids over each group to check with the threshold.

Optimized Query 4- Deliverable3 Query 11
Initial Running time/IO: 2.259 Seconds
Optimized Running time/IO: 0.636 Seconds
Explain the improvement:

This query was also optimized using the index we created on the review table for the business_id. In this query we were grouping reviews for each business_id to select business which has at least two reviews. This is a good example to show how one query optimization leads to optimization of another query. This was a 71.85% improvement when compared to the initial run. Further we could optimize the results by adding an index for postal_code for business and also if necessary on postal_code. However we did not go that far as the query was sending the result back in under 1 seconds. As optimizing further will not much affect the result, but only increasing the size of the database.

Optimized Query 5- Deliverable3 Query 15
Initial Running time/IO: 0.763 Seconds
Optimized Running time/IO: 0.555 Seconds
Explain the improvement:

We improved performance by first joining the attr_goodformeal and the open_at table with the business table in order to have a large table with all the fields of interest readily accessible. We perform the where criteria selection after the joins.

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

General Comments

Work Allocation

ER diagram was designed at a group meeting, while Jerome took charge of implementing it nicely on an online platform called Creately.

DDL diagram was designed based on the ER diagram by Jaakik with Wenuka implementing it on the server with proper modification wherever necessary.

Data preprocessing was mainly done by Jerome and Wenuka to fit with the database requirements while Jaakik took charge of adding data to the server.

For the deliverable 2, the contribution was as follows

- Jaakik: Queries 1, 4, 7, 10
- Jerome: Queries 2, 5, 8, 11
- Wenuka: Queries 3, 6, 9, 12

For the deliverable 3, contribution for each query was as follows,

- Jaakik: Queries 1, 4, 7, 10, 13, 16, 19
- Jerome: Queries 2, 5, 8, 11, 14, 17, 20
- Wenuka: Queries 3, 6, 9, 12, 15, 18

Even though we divided queries among each other, we decided how to tackle queries harder than others in group meetings and helped each other.

Query optimizations were done as a group once everyone finishes the queries.

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

Appendix

Appendix 1: DDL Commands

```
/* Entity descriptions */

CREATE TABLE Attr_Ambience (
   Business_id CHAR(22),
         Sub_Attr_id INTEGER,
   PRIMARY KEY (Business_id, Sub_Attr_id),
   FOREIGN KEY (Sub_Attr_id) references Attr_Ambience(Sub_Attr_id),
   FOREIGN KEY (Business_id) references Business(Business_id)
)

CREATE TABLE Attr_BusinessParking (
   Business_id CHAR(22),
         Sub_Attr_id INTEGER,
   PRIMARY KEY (Business_id, Sub_Attr_id),
   FOREIGN KEY (Sub_Attr_id) references Attr_Ambience(Sub_Attr_id),
   FOREIGN KEY (Business_id) references Business(Business_id)
)

CREATE TABLE Attr_DietaryRestrictions (
   Business_id CHAR(22),
         Sub_Attr_id INTEGER,
   PRIMARY KEY (Business_id, Sub_Attr_id),
   FOREIGN KEY (Sub_Attr_id) references Attr_Ambience(Sub_Attr_id),
   FOREIGN KEY (Business_id) references Business(Business_id)
)

CREATE TABLE Attr_GoodForMeal (
   Business_id CHAR(22),
         Sub_Attr_id INTEGER,
   PRIMARY KEY (Business_id, Sub_Attr_id),
   FOREIGN KEY (Sub_Attr_id) references Attr_Ambience(Sub_Attr_id),
   FOREIGN KEY (Business_id) references Business(Business_id)
)

CREATE TABLE Attr_Music (
   Business_id CHAR(22),
         Sub_Attr_id INTEGER,
   PRIMARY KEY (Business_id, Sub_Attr_id),
   FOREIGN KEY (Sub_Attr_id) references Attr_Ambience(Sub_Attr_id),
```

```
   FOREIGN KEY (Business_id) references Business(Business_id)
)

CREATE TABLE Attr_NoiseLevel (
   Business_id CHAR(22),
         Sub_Attr_id INTEGER,
   PRIMARY KEY (Business_id, Sub_Attr_id),
   FOREIGN KEY (Sub_Attr_id) references Attr_Ambience(Sub_Attr_id),
   FOREIGN KEY (Business_id) references Business(Business_id)
)

CREATE TABLE Attr_Ambience_map (
   Sub_Attr_id INTEGER,
         Sub_Attr_name VARCHAR(50),
   UNIQUE (Sub_Attr_name),
   PRIMARY KEY (Sub_Attr_id)
)

CREATE TABLE Attr_BusinessParking_map (
   Sub_Attr_id INTEGER,
         Sub_Attr_name VARCHAR(50),
   UNIQUE (Sub_Attr_name),
   PRIMARY KEY (Sub_Attr_id)
)

CREATE TABLE Attr_DietaryRestrictions_map (
   Sub_Attr_id INTEGER,
         Sub_Attr_name VARCHAR(50),
   UNIQUE (Sub_Attr_name),
   PRIMARY KEY (Sub_Attr_id)
)

CREATE TABLE Attr_GoodForMeal_map (
   Sub_Attr_id INTEGER,
         Sub_Attr_name VARCHAR(50),
   UNIQUE (Sub_Attr_name),
   PRIMARY KEY (Sub_Attr_id)
)

CREATE TABLE Attr_Music_map (
   Sub_Attr_id INTEGER,
         Sub_Attr_name VARCHAR(50),
   UNIQUE (Sub_Attr_name),
```

```
   PRIMARY KEY (Sub_Attr_id)
)

CREATE TABLE Attr_NoiseLevel_map (
   Sub_Attr_id INTEGER,
         Sub_Attr_name VARCHAR(50),
   UNIQUE (Sub_Attr_name),
   PRIMARY KEY (Sub_Attr_id)
)

CREATE TABLE Business (
   Business_id CHAR(22),
         Address VARCHAR(200),
         Latitude REAL,
         Longitude REAL,
         Review_Count INTEGER,
         is_open Boolean,
         Name VARCHAR(80),
         Postal_Code_id INTEGER,
         Stars REAL,
         UNIQUE (Address, Postal_Code_id),
   PRIMARY KEY (Business_id),
   FOREIGN KEY (Postal_Code_id) references Postal_Code(Postal_Code_id)
)

CREATE TABLE Category_map (
   Category_id INTEGER,
         Category_name VARCHAR(200),
   UNIQUE (Category_name),
   PRIMARY KEY (Category_id)
)

CREATE TABLE Category (
   Category_id INTEGER,
         Business_id CHAR(22),
   PRIMARY KEY (Business_id, Category_id),
   FOREIGN KEY (Category_id) references Category(Category_id)
   FOREIGN KEY (Business_id) references Business(Business_id)
)

CREATE TABLE Elite (
   User_id CHAR(22),
         Elite_year INTEGER,
```

```
   PRIMARY KEY (User_id, Elite_year),
   FOREIGN KEY (User_id) references Users(User_id)
)

CREATE TABLE FRIENDS (
        user_id_1 CHAR(22),
        user_id_2 CHAR(22),
        PRIMARY KEY (user_id_1, user_id_2),
        FOREIGN KEY (user_id_1) references Users(User_id),
   FOREIGN KEY (user_id_2) references Users(User_id)
)

CREATE TABLE Hours (
   Opening_time VARCHAR(8),
        Closing_time VARCHAR(8),
        Hours_id INTEGER,
        PRIMARY KEY (Hours_id)
)

CREATE TABLE Open_at (
        Business_id CHAR(22),
   Opening_hours_id INTEGER,
        Day_id INTEGER,
        PRIMARY KEY (Business_id, Opening_hours_id),
   FOREIGN KEY (Business_id) references Business(Business_id),
        FOREIGN KEY (Opening_hours_id) references Hours(Hours_id),
)

CREATE TABLE Postal_code (
        Postal_code_id INTEGER,
   Postal_code VARCHAR(10),
   City VARCHAR(50),
   State VARCHAR(3),
        UNIQUE (Postal_code, City, State),
        PRIMARY KEY (Postal_code_id)
)

CREATE TABLE Users (
        User_id CHAR(22),
        Avg_Stars REAL,
        Fans INTEGER,
        Useful_Count INTEGER,
        Review_Count INTEGER,
```

```
        Yelping_Since DATETIME,
        Cool INTEGER,
        Funny INTEGER,
        Name CHAR (40),
  Compliment_cool INTEGER,
  Compliment_cute INTEGER,
  Compliment_funny INTEGER,
  Compliment_hot INTEGER,
  Compliment_list INTEGER,
  Compliment_more INTEGER,
  Compliment_note INTEGER,
  Compliment_photos INTEGER,
  Compliment_plain INTEGER,
  Compliment_profile INTEGER,
  Compliment_writer INTEGER,
        PRIMARY KEY (User_id)
)

/* Relationships Description) */

CREATE TABLE Reviews (
        Review_id CHAR(22),
        User_id CHAR(22),
        Business_id CHAR(22),
        review_text CLOB,
        stars INTEGER,
        Cool INTEGER,
        Funny INTEGER,
        Useful INTEGER,
        posted_date TIMESTAMP,
        PRIMARY KEY (Review_id),
        FOREIGN KEY (User_id) references Users(User_id),
  FOREIGN KEY (Business_id) references Business(Business_id)
)

CREATE TABLE Tips (
        Tip_id INTEGER,
        User_id CHAR(22),
        Business_id CHAR(22),
        Tip_text CLOB,
        compliment_count INTEGER,
        posted_date TIMESTAMP,
        PRIMARY KEY (Tip_id),
```

DIAS: Data-Intensive Applications and Systems Laboratory
School of Computer and Communication Sciences
Ecole Polytechnique Fédérale de Lausanne
Building BC, Station 14
CH-1015 Lausanne
URL: http://dias.epfl.ch/

```
        FOREIGN KEY (User_id) references Users(User_id),
    FOREIGN KEY (Business_id) references Business(Business_id)
)
```