Business Intelligence Workload

Query definitions were migrated to https://github.com/ldbc/ldbc_snb_docs. Please check that for the latest version.

~~Query 1 - Posting summary~~

~~Description:~~
~~Given a date, find all Messages created before that date.~~
~~Group them by a 3-level grouping:~~
~~1st: by year of creation~~
~~2nd: for each year, group into message types, i.e., Posts or Comments~~
~~3rd: for each year-type group, split into four groups based on length of their content:~~
~~0 <= length < 40 -> 'short'~~
~~40 <= length < 80 -> 'one liner'~~
~~80 <= length < 160 -> 'tweet'~~
~~160 <= length -> 'long'~~

~~Parameters:~~
~~date: Date~~

~~Result:~~
~~For every 3-level group, return:~~

- ~~year - 32 bit Integer~~
- ~~message type - String -> (post/comment)~~
- ~~length category - String -> (short/one liner/tweet/long)~~
- ~~message count - 32 bit Integer -> total number of Messages (Posts/Comments) in that group~~
- ~~average message length - 32 bit Integer -> average length of the Message content in that group~~
- ~~sum message length - 32 bit Integer -> sum of all message content lengths~~
- ~~per messages - 32 bit floating point -> number of messages in group as a percentage of all messages created before the given date~~
-

~~Sort:~~
~~1st: by year (descending)~~
~~2nd: by message type (Post 1st, Comment 2nd)~~
~~3rd: by size category (short 1st, one liner 2nd, tweet 3rd, long 4rth)~~

~~Query 2 - Top tags for country, age, gender, time~~

~~Description:~~
~~Select all Messages (Posts & Comments) created between date1-date2 (inclusive) by persons located in country1 or country2~~
~~Select the creators (Persons) and the Tags of these Messages~~
~~Split these Persons, Tags and Messages into a 5-level grouping:~~
~~1st: name of country of person~~
~~2nd: month message was created~~
~~3rd: gender of person~~
~~4th: age group of person, defined as:~~
~~years between person's birthday and end of simulation (2013-01-01), divided by 5, rounded down~~
~~5th: name of tag attached to message~~
-

~~Parameters:~~
~~date1 - Date~~
~~date2 - Date~~
~~country1 - String~~
~~country2 - String~~

~~Result:~~
~~For each of the 5-level group return:~~
~~Country.name - String~~
~~month - 32 bit Integer -> 1-12~~
~~Person.gender - String -> male/female~~
~~age group - 32 bit Integer~~
~~Tag.name - String~~
~~message count - 64 bit Integer~~
~~Only return groups where number of messages is greater than 100~~

~~Sort:~~
~~1st: message count (descending)~~
~~2nd: Tag.name (ascending)~~
~~3rd: age group (ascending)~~
~~4th: Person.gender (ascending)~~

~~5th: month (ascending)~~
~~6th: Country.name (ascending).~~

~~**Limit:**~~

~~100~~

~~Query 3 - Tag evolution~~

~~**Description:**~~
~~Given a year and a month, find the Tags that were used in Messages during the given month of the given year, and the Tags that were used during the month after the given month of the given year.~~
~~For both months, compute the count of Messages that used each of the Tags.~~

~~**Parameters:**~~
~~year - 32 bit Integer~~
~~month - 32 bit Integer between 1 and 12~~

~~**Result:**~~
~~Tag.name - String~~
~~count_month1 - 32 bit Integer // occurrences of this Tag during year-month 1~~
~~count_month2 - 32 bit Integer // occurrences of this Tag during year-month 2~~
~~diff - 32 bit Integer // difference between occurrences of this Tag in month 1 and month 2~~
-

~~**Sort:**~~
~~1st: diff (descending)~~

~~2nd: Tag.name (ascending)~~

~~Query 4 - Popular topics in a country~~

~~**Description:**~~
~~Given a TagClass and a Country, find all the Forums created in the given Country, containing at least one Post with Tags belonging directly to the given TagClass.~~
~~The location of a Forum is identified by the location of the Forum's moderator.~~

~~**Parameters:**~~
~~class - 32 bit Integer~~
~~country - 32 bit Integer~~

~~**Results:**~~

~~Forum.id - 64 bit Integer~~
~~Forum.title - String~~
~~Forum.creationDate - DateTime~~
~~Forum-hasModerator->Person.id 64 bit Integer~~
~~count - 32 bit Integer~~

~~**Sort:**~~

~~1st count (descending)~~

~~2nd Forum.id (ascending)~~

~~**Limit:**~~

~~20~~

~~Query 5 - Top posters in a country~~

~~**Description:**~~
~~Find the most popular Forums for a given Country, where the popularity of a Forum is measured by the number of members that Forum has from the given Country.~~
~~For each member of the 100 most popular Forums, count the number of Posts they made in any of those (most popular) Forums.~~

~~Group persons by:~~
~~First, Person.id~~

~~Second, Person.firstName~~
~~Third, Person.lastName~~
~~Fourth, Person.creation date~~

~~**Parameters:**~~
~~country - 32 bit Integer~~

~~**Results:**~~
~~For each group:~~
~~Person.id - 64 bit Integer~~
~~Person.firstName - String~~
~~Person.lastName - String~~
~~Person.creation date - DateTime~~
~~post count - 32 bit Integer //posts created by that person in the top 100 forums~~
~~-~~

~~**Sort:**~~

~~1st post count (descending)~~

~~2nd Person.id (ascending)~~

~~**Limit:**~~

~~100~~

~~Query 6 - Most active Posters of a given Topic~~

~~**Description:**~~
~~Get Persons who have created a Message (Post or Comment) with a given Tag.~~
~~Each Person has a score, computed as follows:~~
~~Count of Messages with the given Tag PLUS count of Likes and Comments in reply of their Messages with the given Tag~~
~~The sum is weighted as follows:~~
~~Messages are multiplied by 1, Comments to Messages are multiplied by 2, Likes are multiplied by 10~~

~~**Parameters:**~~
~~Tag - 32 bit Integer~~

~~**Result:**~~
~~Person.id - 64 bit Integer~~
~~reply count - 32 bit Integer~~
~~like count - 32 bit Integer~~
~~post count - 32 bit Integer~~
~~score - 32 bit Integer~~

~~**Sort:**~~

~~1st score (descending)~~

~~2nd Person.id (ascending)~~

~~**Limit**~~
~~100~~

~~Query 7 - Most authoritative users on a given topic~~:

~~**Description**~~

~~Given a Tag, find all Persons that ever created a Message with the given Tag.~~
~~For each of these Persons compute their "authority score" as follows:~~
~~the sum of "popularity scores" of the Persons that liked any~~
~~of that Person's Messages with the given Tag.~~
~~A Person's "popularity score" is defined as the total number of likes on all of their Messages.~~

~~**Parameters:**~~
~~Tag - 32 bit Integer~~

~~**Result:**~~
~~Person.Id - 64 bit Integer~~
~~authority score - 32 bit Integer~~

~~**Sort:**~~

~~1st authority score (descending)~~

~~2nd Person.id (ascending)~~
-

~~**Limit:**~~

~~100~~

~~Query 8 - Related Topics~~

~~**Description:**~~

~~Find all Messages that have a given Tag.~~
~~Find the Tags attached to replies of these Messages, but only of those replies that do not have the given Tag.~~
~~Group the Tags by name, and get the count of replies in each group.~~

~~**Parameters:**~~
~~Tag - 32 bit Integer~~

~~**Result:**~~
~~For each group:~~
~~Tag.name - String~~
~~count - 32 bit Integer // The number of replies in which the tag appears~~

~~**Sort:**~~

~~1st count (descending)~~

~~2nd Tag.name (ascending)~~
-

~~**Limit:**~~

~~100~~

-

~~Query 9 - Forum with related Tags~~:

~~**Description:**~~

~~Given two TagClasses (tagClass1 & tagClass2), find Forums that contain at least one Post with a Tag from TagClass1 and at least one Post with a Tag from TagClass2 (direct children not transitive) – this may be the same Post.~~
~~Consider the Forums with a number of members greater than a given threshold.~~
~~For every such forum, count the number of Posts that have a Tag from TagClass1 (count1), and the number of posts that have a tag from TagClass2.~~

~~**Parameters:**~~
~~tagClass1 - 32 bit Integer~~
~~tagClass2 - 32 bit Integer~~
~~threshold - 32 bit Integer~~

~~**Return:**~~
~~Forum.id - 64 bit Integer~~
~~count1 - 32 bit Integer // Number of Posts with at least one tag belonging to tagClass1~~
~~count2 - 32 bit Integer // Number of Posts with at least one tag belonging to tagClass2~~
-

~~**Sort:**~~

~~1st count2 - count1 (descending)~~

~~2nd Forum.id (ascending)~~

~~**Limit:**~~

~~100~~

~~Query 10 - Central Person for a Tag~~

~~**Description:**~~

~~Given a Tag, find all Persons that are either interested in the Tag, or have written a Message (Post or Comment) with creation date after a given date and that has a given Tag. For each Person, compute the score as the sum of the following two aspects:~~

- ~~100, if the Person has this tag as their interest, or 0 otherwise~~

- ~~number of messages by this person with the given tag~~

~~**Parameters:**~~

- ~~Tag - 32 bit Integer~~
- ~~date - Date~~

~~**Result:**~~

- ~~Person.id - 64 bit Integer~~
- ~~score - 32 bit Integer~~
- ~~friends score - 32 bit Integer // The sum of the score of the Person's friends~~

~~Sort:~~

~~1st score + friends score (descending)~~

~~2nd Person.id (ascending)~~

~~Limit:~~

~~100~~

-

~~**NOTE:** 2nd September 2016,~~ add ~~date range to query to disincentive pre-computation of results~~

~~Query 11: Unrelated Replies~~

~~**Description**~~
~~Find those Persons of a given country that replied to any Message, such that the reply does not have any Tag in common with the Message. Consider only those replies not containing any word from a given blacklist. For each Person and valid reply, retrieve the Tags associated with the reply, and retrieve the number of likes on the reply.~~

~~**Group the results by:**~~
~~First, Person.id~~
~~Second, Tag.name~~

~~**Parameters:**~~
~~country - 32 bit Integer~~
~~blacklist - Array of Strings~~

~~**Result:**~~
~~For each group:~~
~~Person.id - 64 bit Integer~~
~~Tag.name - String~~
~~count_likes - 32 bit Integer // The count of Likes to replies with that Tag.~~
~~count_replies - 32 bit Integer // The count of replies with that Tag.~~
-

~~**Sort:**~~

~~1st count_likes (desdencing)~~

~~2nd Person.id (asdencing)~~

~~3rd Tag.name~~

~~**Limit:**~~

~~100~~

~~**NOTE: 14rth** September 2016. UPDATE QUERY to match against substrings instead of full word~~

~~Query 12 - Trending Posts:~~

~~**Description:**~~
~~Find all Messages created after a given date, that received more than a given number of likes~~

~~**Parameters:**~~
~~creationDate - Date~~
~~like_count - 32 bit Integer~~

~~**Result:**~~
~~Message.id - 64-bit Integer~~
~~Message.creationDate - DateTime~~
~~Creator.first_name - String // The first name of the post's creator~~
~~Creator.last_name - String // The last name of the post's creator~~
~~like_count - 32-bit Integer // The number of Likes the Post received~~

~~**Sort:**~~
~~1st like_count (descending)~~

~~2nd Message.id (ascending)~~

~~**Limit:**~~

~~100~~

~~Query 13 - Popular Tags per month in a country~~

~~**Description**~~
~~Find all Messages in a given Country, as well as their Tags.~~

~~Group the results by:~~
~~1st: message creation year~~
~~2nd: message creation month~~

~~For each group, find the 5 most popular Tags, where popularity is the number of Messages (from within the same group) where the Tag appears.~~

~~**Parameters:**~~
~~country - String~~

~~**Result:**~~
~~For each group, return:~~
~~Year - 32-bit Integer~~
~~Month - 32-bit Integer // from 1 to 12~~
~~popular_tags - [(Tag.Name - String , popularity - 32-bit Integer)] // sorted descending by popularity, then ascending by tag name~~
-

~~**Sort:**~~

~~1st Year (descending)~~

~~2nd Month (ascending)~~

~~**Limit:**~~

~~100~~

–

~~Query 14 - Top thread initiators~~

~~**Description**~~

~~For each person, count the number Posts they created in the time interval (~~begin,end~~), and the number of messages in each of their (transitive)~~ re
ply~~ trees.~~

~~When calculating message counts only consider messages created within the given time interval.~~

~~Return each person,~~ number~~ of Posts they created, and the count of all messages that appeared in the reply trees (including Post at tree root) they created.~~

~~**Parameters:**~~
~~begin - Date~~
~~end - Date~~

~~**Result:**~~
~~Person.id - 64-bit Integer~~
~~Person.first_name - String~~
~~Person.last_name - String~~
~~thread_count - 32-bit Integer // The number of threads initiated by that Person~~
~~message_count - 32-bit Integer // The number of messages created in all the threads this Person initiated~~
-

~~**Sort:**~~

~~1st message_count (descending)~~

~~2nd Person.id (ascending)~~

~~**Limit:**~~

~~100~~

~~Query 15: Social Normals~~

~~**Description**~~
~~Given a country, find all Persons of the country whose number of friends in the given country equals the (floor of) average number of friends that Persons of the given country have in the given country.~~

~~**Parameters:**~~
~~country - 32 bit Integer~~

~~**Result:**~~
~~Person.id - 64 bit Integer~~
~~count - 32 bit Integer // The number of friends~~

~~**Sort:**~~
~~1st Person.id (ascending)~~
~~**Limit:**~~

~~100~~

~~**Query 16 - Experts in Social Circle [DISCUSS PARAMETER SELECTION, VARIANCE DUE TO DIFFERENT DEPTHS, ETC.]**~~

~~**Description:**~~

~~Given a Person, find all other Persons that live in a given country and are connected to given person by a transitive path with length in range [min,max] through the knows relation.~~
~~**[DISCUSS: edge isomorphism semantics]**~~
~~For each of these Persons, retrieve all of their Messages (Posts & Comments) that contain at least one Tag belonging to a given TagClass (direct relation not transitive).~~
~~For each Message, also retrieve its Tags.~~

~~**Group:**~~
~~First, by Tag.name~~
~~Second, by Person.id~~

~~**Parameters:**~~
~~Person.id - 64 bit Integer~~
~~Country.name - String~~
~~TagClass.name - String~~
~~min_path_distance - 32 bit Integer~~
~~max_path_distance - 32 bit Integer~~

~~**Result:**~~
~~For each group, return:~~
~~Person.id - 64 bit Integer~~
~~Tag.name - String~~
~~message_count - 32 bit // number of Messages created by that Person containing that Tag~~

~~**Sort:**~~

~~1st post_count (descending)~~

~~2nd Tag.name (ascending)~~

~~3rd Person.id (ascending)~~

~~**Limit:**~~

~~100~~

~~Query 17 - Friend Triangles~~

~~**Description:**~~

~~Foor a given country, count all the distinct triples of persons such that A is friend of B, B is friend of C, and C is friend of A.~~

~~Distinct means that given a triple *a* in the result set R of all qualified triples, there is not a triple *b* in R such that |a U b| = 3~~

~~**Parameters:**~~
~~country - String~~

~~**Result:**~~
~~count - 32 bit Integer~~

~~**Sort:**~~

~~**Limit:**~~

~~Query 18 - How many persons have a given number of posts~~:

~~**Description:**~~
~~For each Person, count the number of Messages (Posts & Comments) they made.~~
~~Only consider messages with:~~
~~- length below the [length threshold]~~
~~- creationDate after [given date]~~
~~- any of the [given languages]~~

~~Group the results by~~
~~1: number of messages created~~

~~**Parameters:**~~
~~creationDate - Date~~

~~**Result:**~~
~~For each group:~~

~~message count - 32 bit integer~~
~~count - 32 bit integer // the number of Persons with that number of messages~~

~~**Sort:**~~

~~1st count (descending)~~

~~**Limit:**~~

~~Query 19 - Stranger's Interaction:~~

~~For all the Persons born after a certain date, find all the strangers they interacted with, where strangers are Persons that do not KNOW each other.~~
~~There is no restriction on the date that strangers were born.~~
~~Consider only strangers that are members of Forums tagged with tagClass1 (direct children not transitive) AND members of Forums tagged with~~
~~tagClass2 (direct children not transitive). It does not matter if these Tags are attached to the same Forum, or different Forums.~~
~~We define interaction as follows: a Person replies to a Message (Post or Comment) by another Person.~~
~~For each Person, count the number of strangers they interacted (directed) with and total number of times they interacted (directed) with them.~~

~~Parameters:~~
~~date - Date~~
~~tagClass1 - 32-bit Integer~~
~~tagClass2 - 32-bit Integer~~

~~Result:~~
~~Person.id - 64-bit Integer~~
~~strangers_count - 32-bit Integer~~
~~interaction_count - 32-bit Integer~~

~~Return Top 100~~
~~Sort~~
~~1st interaction_count (descending)~~
~~2nd Person.id (ascending)~~

~~Limit:~~

~~100~~

~~Query 20 - High level topics:~~

~~Description:~~
~~For all given TagClasses, count number of Messages that have a Tag that belongs to that TagClass or any of its children (all descendants through a transitive relation).~~

~~Paramters:~~
~~[TagClasses] - [String]~~

~~Return:~~
~~TagClass.name - String // The TaglClass of the root~~
~~post_count - 32 bit Integer~~

~~Sort:~~
~~1st post_count (descending)~~

~~2nd TagClass.name (ascending)~~

~~Limit:~~

~~100~~

~~Query 21 - Zombies in a country:~~

~~Description:~~
~~Find zombies within the given country, and return their zombie scores.~~
~~A zombie is a Person created before the given end_date and that has created between [0,1) messages per month, on average, during the time range between profile creation date and the given end_date~~
~~The number of months spans the time range from creation date of the profile to the end date and also includes partial months.~~
~~The zombie score of a person is the ratio between the number of likes received from zombies and the total number of likes received on that person's messages - only count likes received from profiles that were created before the given end date.~~

~~Parameters:~~
~~country - String~~
~~end_date - Date~~

~~Result:~~
~~Person.id - 64 bit Integer~~
~~zombie_like_count - 32 bit Integer~~
~~real_like_count - 32 bit Integer~~
~~zombie_score - 32 bit Floating point number~~

~~Sort:~~

~~1st zombie_score (descending)~~

~~2nd Person.id (ascending)~~

~~Limit:~~

~~100~~

~~**Query 22: International Dialog**~~

~~**Description:**~~

~~Consider all pairs of people (p1, p2) such that one is located in a city of Country X and the other is located in a city of Country Y.~~
~~For each city of Country X return the highest scoring pair.~~
~~The score of a pair is defined as the sum of the scores of the following kinds of interaction:~~
~~* p1 has created a reply Comment to at least one Comment or Post by p2 -> Score: 4~~
~~* p1 has created at least one Post or Comment that p2 has created a reply Comment to -> Score: 1~~
~~* p1 & p2 Know each other -> Score: 15~~
~~* p1 liked at least one Post or Comment by p2 -> Score: 10~~
~~* p1 has created at least one Post or Comment that was liked by p2 -> Score: 1~~
~~I.e., the maximum score a pair can obtain is: 4 + 1 + 15 + 10 + 1 = 31~~

~~**Parameters**~~
~~countryX - String~~
~~countryY - String~~

~~**Result:**~~
~~Person1.id - 64-bit Integer~~
~~Person2.id - 64-bit Integer~~
~~CityX.name - String~~
~~score - 32-bit Integer~~

~~**Sort**~~

~~1st score (descending)~~

~~2nd Person1.id (ascending)~~

~~2nd Person2.id (ascending)~~

~~**Limit:**~~

~~**Query 23: Holiday Destinations**~~

~~**Description:**~~
~~Count the messages all residents of country X have written abroad grouped by month and country.~~
~~A message was written abroad if the country the message was written in is different than the country of the person it was written by.~~

~~Group results:~~
~~First: name of destination Country~~
~~Second: month in which people traveled there~~

~~**Parameters:**~~
~~country - String~~

~~**Result:**~~
~~For each group:~~
~~message_count: 32-bit Integer // The number of messages in each group~~
~~Country.name: String // The name of the destination country~~
~~month: 32-bit Integer~~

~~**Sort:**~~

~~1st message_count (descending)~~

~~2nd Country.name (ascending)~~

~~3rd month (ascending)~~

~~**Limit:**~~

~~100~~

~~**Query 24: Messages By Topic And Continent**~~

~~**Description:**~~

~~Find all Messages tagged with a Tag from the given TagClass (non-transitive). Count all messages and their likes grouped by continent, year, and month.~~

~~Group all the Messages:~~
~~First: year of publication~~
~~Second: month of publication~~
~~Third: continent~~

~~**Parameters:**~~
~~TagClass - String~~

~~**Result:**~~
~~For each group:~~
~~message_count 32 bit Integer~~
~~like_count 32 bit Integer~~
~~year 32 bit Integer~~
~~month 32 bit Integer~~
~~continent String~~

~~**Sort:**~~

~~1st year (ascending)~~

~~2nd month (ascending)~~

~~3rd continent name (descending)~~

~~**Limit:**~~

~~100~~

~~Q25 – Weighted paths~~

~~**Description:**~~

~~Given two Persons, find all (unweighted) shortest paths between these two Persons, in the subgraph induced by the Knows relationship.~~
~~Then, for each path calculate a weight. The nodes in the path are Persons, and the weight of a path is the sum of weights between every pair of consecutive Person nodes in the path.~~
~~The weight for a pair of Persons is calculated such that every reply (by one of the Persons) to a Post (by the other Person) contributes 1.0, and every reply (by ones of the Persons) to a Comment (by the other Person) contributes 0.5.~~
~~Only consider messages that were created in a forum that was created within the timeframe [startDate,endDate]~~
~~Return all the paths with shortest length, and their weights.~~

~~**Parameter**~~

~~Person.id: ID // person 1~~
~~Person.id: ID // person 2~~
startDate~~: Date~~
endDate~~: Date~~

~~**Result (for each result return)**~~

~~[Person.id] [ID] // Identifiers representing an ordered sequence of the Persons in the path weight 64-bit Float~~

~~**Sort**~~

~~1st weight (descending) // The order of paths with the same weight is unspecified~~