

# PROJECT DOCUMENTATION

## Databases for Data Science - Group 8

### Presentation link:

[https://www.canva.com/design/DAGHRFssKVM/ELLKibThYgD8gYcx4tD-Ow/edit?utm\\_content=DAGHRFssKVM&utm\\_campaign=designshare&utm\\_medium=link2&utm\\_source=sharebutton](https://www.canva.com/design/DAGHRFssKVM/ELLKibThYgD8gYcx4tD-Ow/edit?utm_content=DAGHRFssKVM&utm_campaign=designshare&utm_medium=link2&utm_source=sharebutton)

### Revision

Due to the data having a different schema from our works in part 1, the relational schema was revised. The revision is documented below.

#### Changes:

- Beneficiary(ID, name, address) → Beneficiary(id, name, address, city\_id) (city actually connects to beneficiary instead of request)
- Request(ID, numberOfVolunteers, priorityValue, startDate, endDate, registerByDate, beneficiaryID, cityID) → Request(id, beneficiary\_id, number\_of\_volunteers, priority\_value, start\_date, end\_date, register\_by\_date)
- Skill(name, description)
- AreaOfInterest(name) → Interest(name)
- Volunteer(ID, name, birthdate, email, address, readinessToTravel) → Volunteer(id, birthdate, city\_id, name, email, address, travel\_readiness)
- Application(ID, modifiedTime, validity, volunteerID, requestID)
- City(ID, name, geolocation)
- RequiredSkill(requestID, skillName) → Request\_skill(request\_id, skill\_name, min\_need, value)
- RequiredInterest(requestID, areaName) → None
- Appraisal(beneficiaryID, skillName, appraisalValue, minimumValue, importanceValue) → None
- VolunteerSkill(volunteerID, skillName)
- VolunteerRange(volunteerID, cityID)
- VolunteerInterest(volunteerID, interestName)

#### Notes:

- Mostly changes to match the dataset columns naming (eg. from VolunteerRange → Volunteer\_range).
- City actually has an association with Beneficiary instead of Request.
- The request interest is instead used as the request title instead of being a separate table.
- The request can be at multiple locations → Request\_location(Request\_id, city\_id)

#### Final revised relation schema:

- Beneficiary(id, name, address, city\_id)

- Request(id, beneficiary\_id, number\_of\_volunteers, priority\_value, start\_date, end\_date, register\_by\_date)
- Skill(name, description)
- Interest(name)
- Volunteer(id, birthdate, city\_id, name, email, address, travel\_readiness)
- Volunteer\_application(id, request\_id, volunteer\_id, modified, is\_valid)
- City(name, id, geolocation)
- Request\_skill(request\_id, skill\_name, min\_need, value)
- Skill\_assignment(volunteer\_id, skill\_name)
- Volunteer\_range(volunteer\_id, city\_id)
- Interest\_assignment(interest\_name, volunteer\_id)
- Request\_location(Request\_id, city\_id)

The final revised schema matches the data. The script for creating the database, creating tables, and populating the tables are included in the submission. Below are the documented results of the queries.

## A. BASIC

1. This query updates the 'title' column in the 'request' table so that it contains the starting date and the end date of each request. The date columns are formatted to DD/MM/YYYY. The concatenation operator ( || ) is used to append a formatted date range (from *start\_date* to *end\_date*) to the title of each request.

request		
Enter a SQL expression to filter results (use Ctrl+Space)		
Grid	123 id	ABC title
1	2	work with young needed (31/05/2022 to 01/06/2022)
2	3	guide and teach needed (22/07/2024 to 27/07/2024)
3	4	guide and teach needed (25/09/2024 to 25/09/2024)
4	5	work with elderly needed (01/03/2021 to 07/03/2021)
5	6	work with elderly needed (18/10/2024 to 18/10/2024)
6	7	organise activities needed (15/09/2023 to 18/09/2023)
7	8	help in crisis needed (05/06/2024 to 07/06/2024)
8	9	work with young needed (28/05/2023 to 01/06/2023)
9	10	work in team needed (25/03/2024 to 27/03/2024)
10	11	collect donations needed (14/11/2024 to 15/11/2024)

2. This SQL query is designed to find volunteers who have applied for requests and have valid applications and to count the number of skills they possess that match the skills required by the requests. The query uses two CTEs to first find the number of matching skills for each volunteer-application and then ensure that all valid applications are considered. It then selects relevant details from the Request and Volunteer tables, joining them with the processed data in the CTEs. The final output is ordered to show requests, ranked volunteers by their matching skills, and sorted names alphabetically within the same skill match count.

	request_id	volunteer_id	volunteer_name	matching_skills
1	1	230283-963X	Mikko Rossi	3
2	1	211099-910H	Anniina Saastamoinen	1
3	1	011074-9149	Henna Hartikainen	1
4	1	160903A941P	Johannes Jäntti	1
5	1	211074-9401	Matilda Tuominen	1
6	1	250681-919H	Tapio Rantala	1
7	1	210753-990T	Oona Kauppinen	0
8	2	190697-999B	Anton Ketola	6
9	2	270794-9576	Matias Helminen	5
10	2	220782-910B	Pauliina Männistö	5
11	2	200569-926L	Tanja Niemi	4
12	2	101003A9918	Kari Lampinen-Heikkinen	3
13	3	190697-999B	Anton Ketola	7
14	3	030693-935X	Kasper Kilpeläinen	6
15	3	100494-989U	Markku Saastamoinen	5
16	3	200958-9326	Ilona Nieminen	3
17	3	220782-910B	Pauliina Männistö	3

- This SQL query calculates how many volunteers are missing for each skill in each request and returns a list of these shortages. The query uses two CTEs to calculate the volunteer skill coverage for each request and identify where there are shortages. The final query selects and orders the necessary data to display the requests along with the skills that need more volunteers and the exact number of volunteers still needed.

	122 request_id	ABC skill_name	123 min_need	122 missing_volunteers
1	1	CommunicationAndMarketing	5	2
2	1	EventHosting	3	1
3	1	EventOrganization	2	1
4	2	CookingAndBaking	3	2
5	2	PublicPerformances	3	1
6	2	TeamGuide	4	2
7	3	DigitalCompetence	6	2
8	3	EventHosting	4	1
9	3	EventOrganization	6	3
10	3	Organizational	4	1
11	3	TrainPeople	6	2
12	4	DigitalCompetence	6	3
13	5	HealthCareOrFirstAid	5	1
14	5	PhotographyAndVideo	4	1

- The SQL query retrieves a list of requests along with their associated beneficiaries, ordered by the priority value of the requests in descending order and, within the same priority, by the registration date in descending order.

	122 request_id	122 beneficiary_id	ABC name	123 priority_value	register_by_date
1	27	8	Immigration	5	2024-11-16 13:30:00.000
2	226	9	Local Branch	5	2024-10-29 05:30:00.000
3	276	5	Homeless Shelter	5	2024-10-03 17:00:00.000
4	106	1	Hospital	5	2024-09-25 08:00:00.000
5	335	1	Hospital	5	2024-09-23 19:30:00.000
6	376	7	Event First-Aid	5	2024-09-13 05:29:00.000
7	344	1	Hospital	5	2024-08-02 00:30:00.000
8	366	7	Event First-Aid	5	2024-07-24 09:00:00.000
9	321	7	Event First-Aid	5	2024-07-21 02:29:00.000
10	201	6	Blood-Drive (PA)	5	2024-07-13 20:00:00.000
11	221	3	Elderly Care	5	2024-07-08 06:00:00.000
12	136	4	Youth Centre	5	2024-06-29 04:29:00.000
13	114	3	Elderly Care	5	2024-06-22 12:00:00.000
14	157	5	Homeless Shelter	5	2024-06-20 12:00:00.000
15	381	5	Homeless Shelter	5	2024-06-19 01:30:00.000
16	341	8	Immigration	5	2024-06-06 14:00:00.000
17	291	8	Immigration	5	2024-06-04 05:30:00.000
18	131	7	Event First-Aid	5	2024-05-25 15:29:00.000
19	8	6	Blood-Drive (PA)	5	2024-05-25 00:00:00.000

- This query will list the requests that are within the volunteer's range and either match at least two of their skills or have no skill requirements. Volunteer is joined with *Volunteer\_range* to get the cities each volunteer can work in. *Volunteer\_range* is joined with *Request\_location* to find requests in those cities. Request is joined with *Request\_skill* to get the skills required by each request. *Skill\_assignment* is joined to find if the volunteer has the required skills. The subquery counts the number of distinct skills required by the request that the volunteer possesses. The query filters that either the volunteer matches at least two required skills for the request or the request does not require any skills.

	volunteer_id	volunteer_name	request_id	priority_value	start_date	end_date	request_city_id
1	010469-981A	Pentti Heiskanen	1	1	2024-07-25 22:15:00.000	2024-07-28 18:00:00.000	687
2	010469-981A	Pentti Heiskanen	2	3	2022-05-31 04:15:00.000	2022-06-01 17:00:00.000	687
3	010469-981A	Pentti Heiskanen	5	2	2021-03-01 01:15:00.000	2021-03-07 18:45:00.000	72
4	010469-981A	Pentti Heiskanen	5	2	2021-03-01 01:15:00.000	2021-03-07 18:45:00.000	687
5	010469-981A	Pentti Heiskanen	6	4	2024-10-18 00:15:00.000	2024-10-18 15:00:00.000	687
6	010469-981A	Pentti Heiskanen	10	5	2024-03-25 21:15:00.000	2024-03-27 12:30:00.000	687
7	010469-981A	Pentti Heiskanen	11	4	2024-11-14 03:15:00.000	2024-11-15 15:45:00.000	687
8	010469-981A	Pentti Heiskanen	12	1	2020-03-19 21:45:00.000	2020-03-20 15:45:00.000	72
9	010469-981A	Pentti Heiskanen	12	1	2020-03-19 21:45:00.000	2020-03-20 15:45:00.000	687
10	010469-981A	Pentti Heiskanen	13	2	2023-09-09 02:00:00.000	2023-09-11 17:30:00.000	72
11	010469-981A	Pentti Heiskanen	14	1	2022-08-17 15:00:00.000	2022-08-18 08:00:00.000	687
12	010469-981A	Pentti Heiskanen	15	2	2024-12-26 16:00:00.000	2024-12-27 21:45:00.000	687
13	010469-981A	Pentti Heiskanen	17	0	2021-06-09 05:30:00.000	2021-06-09 16:45:00.000	72
14	010469-981A	Pentti Heiskanen	17	0	2021-06-09 05:30:00.000	2021-06-09 16:45:00.000	687
15	010469-981A	Pentti Heiskanen	18	2	2024-05-23 11:15:00.000	2024-05-23 18:00:00.000	72
16	010469-981A	Pentti Heiskanen	19	1	2022-06-02 13:45:00.000	2022-06-02 20:30:00.000	72
17	010469-981A	Pentti Heiskanen	19	1	2022-06-02 13:45:00.000	2022-06-02 20:30:00.000	687
18	010469-981A	Pentti Heiskanen	20	0	2024-07-26 03:45:00.000	2024-07-26 17:15:00.000	687
19	010469-981A	Pentti Heiskanen	21	2	2024-10-24 18:15:00.000	2024-10-29 11:15:00.000	687

- This query lists all the requests where the title matches each volunteer's area of interest and are still available to register. Since the *'interest\_name'* in the *'interest\_assignment'* table is not in the same format as *'title'* in the *'request'* table, the *'interest\_name'* is lowered and spaced; then the LIKE function is used to match those *interest\_name* and title. The filter *r.register\_by\_date >= CURRENT\_DATE* ensures that only current or future requests are considered, excluding any requests whose register-by date has passed.

	volunteer_id	request_id	beneficiary_id	interest_name	start_date	end_date	register_by_date
1	011095-974M	24	4	FoodHelp	2024-11-17 13:45:00.000	2024-11-17 22:45:00.000	2024-11-12 11:30:00.000
2	011095-974M	88	6	FoodHelp	2024-12-12 13:45:00.000	2024-12-12 21:15:00.000	2024-11-28 16:30:00.000
3	011095-974M	220	5	FoodHelp	2024-08-06 19:30:00.000	2024-08-09 11:15:00.000	2024-07-29 19:30:00.000
4	011095-974M	314	7	FoodHelp	2024-07-02 19:00:00.000	2024-07-07 12:30:00.000	2024-06-26 22:59:00.000
5	011095-974M	3	9	GuideAndTeach	2024-07-22 22:15:00.000	2024-07-27 17:30:00.000	2024-07-12 06:00:00.000
6	011095-974M	4	5	GuideAndTeach	2024-09-25 02:30:00.000	2024-09-25 18:30:00.000	2024-09-17 02:59:00.000
7	011095-974M	56	7	GuideAndTeach	2024-09-13 18:00:00.000	2024-09-14 11:00:00.000	2024-09-13 11:00:00.000
8	011095-974M	80	8	GuideAndTeach	2024-06-26 00:45:00.000	2024-06-29 21:00:00.000	2024-06-14 11:00:00.000
9	011095-974M	101	6	GuideAndTeach	2024-09-21 17:45:00.000	2024-09-22 11:30:00.000	2024-09-10 14:00:00.000
10	011095-974M	176	8	GuideAndTeach	2024-06-18 08:15:00.000	2024-06-19 20:30:00.000	2024-06-13 10:00:00.000
11	011095-974M	231	2	GuideAndTeach	2024-09-18 02:45:00.000	2024-09-20 14:15:00.000	2024-09-08 17:30:00.000
12	011095-974M	290	9	GuideAndTeach	2024-11-21 03:15:00.000	2024-11-22 17:45:00.000	2024-11-16 06:29:00.000
13	011095-974M	301	7	GuideAndTeach	2024-12-02 05:45:00.000	2024-12-03 19:30:00.000	2024-11-23 08:00:00.000
14	011095-974M	308	9	GuideAndTeach	2024-12-08 12:45:00.000	2024-12-09 19:15:00.000	2024-11-30 07:30:00.000
15	011095-974M	311	8	GuideAndTeach	2024-10-13 08:45:00.000	2024-10-14 23:30:00.000	2024-10-01 21:59:00.000
16	200958-9326	11	3	CollectDonations	2024-11-14 03:15:00.000	2024-11-15 15:45:00.000	2024-11-01 00:30:00.000
17	200958-9326	15	7	CollectDonations	2024-12-26 16:00:00.000	2024-12-27 21:45:00.000	2024-12-13 12:29:00.000
18	200958-9326	20	1	CollectDonations	2024-07-26 03:45:00.000	2024-07-26 17:15:00.000	2024-07-25 09:29:00.000
19	200958-9326	51	8	CollectDonations	2024-10-25 14:45:00.000	2024-10-26 20:45:00.000	2024-10-22 15:29:00.000

7. This query is designed to retrieve a list of requests along with the names and email addresses of volunteers who have applied for those requests, but only if the request is not located in the same city as the volunteer. Additionally, the results are ordered by the volunteers' travel readiness in descending order. Condition: *WHERE RL.request\_id IS NULL* filters the results to include only those requests that are not in the same city as the volunteer. This condition checks if the *request\_id* from the *Request\_Location* table is null, which indicates that no matching record was found.

	request_id	name	email
1	237	Kalervo Rinne	kalervo.rinne@dbcourse.cs.aalto.fi
2	143	Kalervo Rinne	kalervo.rinne@dbcourse.cs.aalto.fi
3	20	Kalervo Rinne	kalervo.rinne@dbcourse.cs.aalto.fi
4	82	Kalervo Rinne	kalervo.rinne@dbcourse.cs.aalto.fi
5	227	Kalervo Rinne	kalervo.rinne@dbcourse.cs.aalto.fi
6	151	Karoliina Kallio	karoliina.kallio@dbcourse.cs.aalto.fi
7	99	Karoliina Kallio	karoliina.kallio@dbcourse.cs.aalto.fi
8	100	Karoliina Kallio	karoliina.kallio@dbcourse.cs.aalto.fi
9	214	Karoliina Kallio	karoliina.kallio@dbcourse.cs.aalto.fi
10	51	Karoliina Kallio	karoliina.kallio@dbcourse.cs.aalto.fi
11	382	Karoliina Kallio	karoliina.kallio@dbcourse.cs.aalto.fi
12	117	Paavo Kuusisto	paavo.kuusisto@dbcourse.cs.aalto.fi
13	113	Paavo Kuusisto	paavo.kuusisto@dbcourse.cs.aalto.fi
14	88	Paavo Kuusisto	paavo.kuusisto@dbcourse.cs.aalto.fi
15	69	Paavo Kuusisto	paavo.kuusisto@dbcourse.cs.aalto.fi
16	34	Paavo Kuusisto	paavo.kuusisto@dbcourse.cs.aalto.fi
17	381	Paavo Kuusisto	paavo.kuusisto@dbcourse.cs.aalto.fi
18	220	Paavo Kuusisto	paavo.kuusisto@dbcourse.cs.aalto.fi
19	368	Paavo Kuusisto	paavo.kuusisto@dbcourse.cs.aalto.fi


8. This query orders the skills overall (from all requests) in the most prioritized to least prioritized (average the importance value)

	avg skill_name	avg_priority
1	Organizational	2.6223776224
2	EventOrganization	2.5960264901
3	HealthCareOrFirstAid	2.5735294118
4	PublicPerformances	2.5546875
5	DigitalCompetence	2.5454545455
6	MeetingPeople	2.515625
7	FinanceAndAccounting	2.5067567568
8	CookingAndBaking	2.472972973
9	CommunicationAndMarketing	2.437037037
10	TeamGuide	2.435483871
11	PhotographyAndVideo	2.4121621622
12	EventHosting	2.4074074074
13	Rescue	2.2887323944
14	TrainPeople	2.1893939394

9. This query aims to identify popular skills among volunteers. This query provides insights into which skills are in high demand among volunteers. This information can inform beneficiaries to better align with volunteer interests and expertise.


	skill_name	volunteer_count
1	DigitalCompetence	98
2	HealthCareOrFirstAid	95
3	Organizational	93
4	PublicPerformances	92
5	MeetingPeople	89
6	EventHosting	89
7	TrainPeople	88
8	CookingAndBaking	84
9	CommunicationAndMarketing	84
10	PhotographyAndVideo	83
11	EventOrganization	80
12	Rescue	80
13	TeamGuide	80
14	FinanceAndAccounting	79

10. This query aims to determine volunteer availability by age group. This query provides insights into the distribution of volunteer age demographics. This information can be useful for beneficiaries to tailor their program to specific age groups.

	 abc age_group ▼	123 volunteer_count ▼
1	18-30	50
2	31-45	51
3	46-60	51
4	Above 60	39

11. This query aims to identify requests requiring attention from the volunteers. Those requests are considered as urgent when their register date is within one week from the current date. This query helps prioritize requests that require immediate attention and action from volunteers and organizations involved in the matching process.

Note: this output is generated on 07 June 2024. The output varies according to the date when the query is executed.

	123 request_id ▼	 register_by_date ▼	abc beneficiary_name ▼
1	66	2024-06-08 23:59:00.000	Local Branch
2	371	2024-06-09 20:00:00.000	Youth Centre
3	176	2024-06-13 10:00:00.000	Immigration



12. This query aims to identify volunteers with a set of unique skills, ordered by the number of skills they possess. This information can be valuable for matching volunteers to requests that require a wide range of expertise.

	volunteer_id	volunteer_name	unique_skills_count
1	030474-969H	Johan Tolonen-Riikonen	13
2	130597-961E	Martti Kärkkäinen	13
3	271194-957D	Esa Laakso	13
4	190703A954E	Susanna Autio-Puranen	13
5	101156-956L	Terttu Keränen	13
6	231288-9916	Ilmari Korhonen	13
7	250162-950K	Helinä Kukkonen-Mäki	13
8	010573-901K	Juhani Pesonen	12
9	120855-9556	Mikael Harju	12
10	301198-9549	Johanna Karjalainen	12
11	100174-937K	Robert Laine	12
12	160161-981H	Vilho Puhakka	12
13	050677-9718	Mika Moilanen	12
14	180955-948M	Marianne Malinen	12
15	120198-990S	Marjo Salo-Linna	11
16	250684-9410	Teemu Eriksson	11
17	211259-994H	Helmi Salonen-Valkonen	11
18	190792-9111	Jarno Juntunen-Väisänen	11
19	031185-956K	Kaarina Leppänen	11

## B. ADVANCED

### a. Views:

- For this part, a view named **BeneficiaryStats** is created. The number of volunteers applied is counted in a subquery, and left joined on *request\_id*. The average for each beneficiary is then calculated. For the average age, we use the function AGE() to calculate the age in Year and days, then the Year is extracted, and average is calculated. For the number of volunteers needed, a simple average calculation on column *number\_of\_volunteers* is done.

	beneficiary_id	beneficiary_name	avg_volunteers_applied	avg_age_applied	avg_volunteers_needed
1	4	Youth Centre	8.3374613003	44.786377709	16.6439628483
2	6	Blood-Drive (PA)	7.8194444444	44.7118055556	16.6006944444
3	2	Food Bank	8.7675840979	45.7125382263	19.6177370031
4	9	Local Branch	7.6141732283	46.4842519685	19.688976378
5	7	Event First-Aid	7.7314285714	43.2714285714	19.0114285714
6	3	Elderly Care	8.352	43.712	17.788
7	1	Hospital	7.023715415	42.8300395257	20.7826086957
8	5	Homeless Shelter	8.3558282209	44.8006134969	17.7576687117
9	8	Immigration	8.4674220963	44.6260623229	19.8498583569

- Create a view with each city id, name, the number of requests at these cities, the number of volunteers operating in these cities, and the corresponding ratio between the number of volunteers and number of requests. This view is intended to show how supply is compared to demand in different cities. A view called **CityStats** is created for this part.

citystats Enter a SQL expression to filter results (use Ctrl+Space)						
	city_id	city_name	num_requests	num_volunteers	vtorratio	
1	72	Hailuoto	138	61	0.4420289855	
2	426	Liperi	102	55	0.5392156863	
3	504	Myrskylä	137	58	0.4233576642	
4	687	Rautavaara	166	55	0.3313253012	
5	704	Rusko	206	63	0.3058252427	
6	783	Säkylä	103	49	0.4757281553	
7	834	Tammela	127	56	0.4409448819	
8	886	Ulvila	101	59	0.5841584158	

## b. Trigger and Functions:

- For this part, a function **validate\_volunteer\_id()** is created. First, the length of the id is checked. Next, different parts of the id are extracted using the *substring()* method. The separator\_char is checked for validity. Finally, the concatenation between date of birth and the individual string is divided by 31 for the remainder. The remainder is then casted to ASCII using the case switch mechanic. The cases are constructed using the ASCII table and the official Finnish converting table.

Decimal	Hex	Char
64	40	@
65	41	A
66	42	B
67	43	C
68	44	D
69	45	E
70	46	F
71	47	G
72	48	H
73	49	I
74	4A	J
75	4B	K
76	4C	L
77	4D	M
78	4E	N
79	4F	O
80	50	P
81	51	Q
82	52	R
83	53	S
84	54	T
85	55	U
86	56	V
87	57	W
88	58	X
89	59	Y
90	5A	Z

Remainder	Control character	Remainder	Control character	Remainder	Control character
0	0	11	B	21	N
1	1	12	C	22	P
2	2	13	D	23	R
3	3	14	E	24	S
4	4	15	F	25	T
5	5	16	H	26	U
6	6	17	J	27	V
7	7	18	K	28	W
8	8	19	L	29	X
9	9	20	M	30	Y
10	A				

Remainder ranges from 0 to 30.

For remainder less than 10, the value is left as is and converted to string.

For the alphabet characters, compared to the ASCII table, the convert table does not use many characters, such as G, I, O, Q. The cases are created to account for every missing character increment.

- For this part, a function **update\_number\_of\_volunteers()** and a trigger **update\_nof\_volunteers** are created. The trigger calls the mentioned function for every updated row after every update on the *min\_need* column of *Request\_skill*. We know that the total number of volunteers needed for each request is calculated as the sum of unskilled volunteers (those without any skill requirements) and the minimum need for each required skill. Therefore, the new *number\_of\_volunteers* required is simply the difference between the new and the old *min\_need* for a required skill

### c. Transactions:

- Optional
- For this part, I create a transaction that if the modified field in relation *Volunteer\_application* is changed, it will check if the new modified date is later than *register\_by\_date* for the corresponding *request\_id* in relation *Request*. If the modified date is later then the application is invalid; otherwise, the application is valid.

Below is an example for the application of the transaction. Before running the transaction, the last modification of the application is on 22.06.2023 which is earlier than the *register\_by\_date* of the corresponding *request\_id* so *is\_valid* is true.

	<small>123</small> id	<small>123</small> request_id	<small>ABC</small> volunteer_id	modified	<input checked="" type="checkbox"/> is_valid
1	2	263	011095-974M	2023-06-22 13:16:35.237	[v]

	volunteers	<small>123</small> priority_value	start_date	end_date	register_by_date
1	17	0	2023-07-06 10:45:00.000	2023-07-07 07:15:00.000	2023-06-28 04:30:00.000

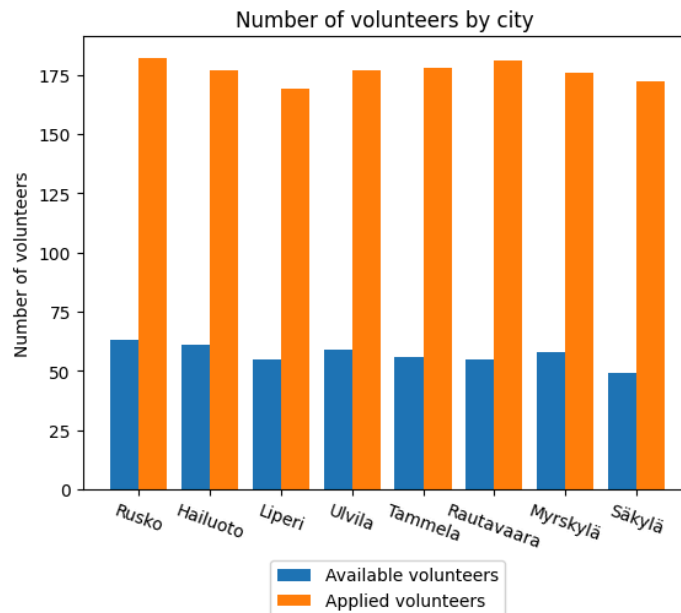
After running the transaction, I update the modified to *CURRENT\_TIMESTAMP* for testing, because modified is now later than the corresponding *register\_by\_date* of the corresponding *request\_id* then *is\_valid* is updated to false.

	<small>123</small> id	<small>123</small> request_id	<small>ABC</small> volunteer_id	modified	<input checked="" type="checkbox"/> is_valid
1	2	263	011095-974M	2024-06-08 11:54:39.967	[ ]

The reason for my choice of transaction is because this transaction is helpful and can prevent some certain accidents. For example, if a deadline for a request is approaching, some people might submit a low-quality draft or application and then modify it after the deadline. This transaction ensures that any such modifications immediately invalidate their application if it occurs after the deadline.

#### d. Analysis:

1. For this part, *volunteer\_range* is used to calculate the number of available volunteers, and *request\_location* with *request* and *volunteer\_application* are used to calculate the number of applied volunteers. These 2 tables (*applied* and *available*) are then joined together to create the final table. Then a double bar chart is graphed to illustrate the data.



2. For this, the scoring system involves 4 criterias: skill, range, travel readiness, and interest. Four views are created to calculate the 4 criterias' scores of the volunteers for each request. These views are then joined together to calculate the final score, with skill score is worth 70%, range score is worth 20%, travel readiness score is worth 5%, interest score is worth 5%. Since the title of request *work in team* does not match the interest *workInATeam*, 2 views *interest\_corrected* and *interest\_assignment\_corrected* to change from *workInATeam* to *workInTeam*.

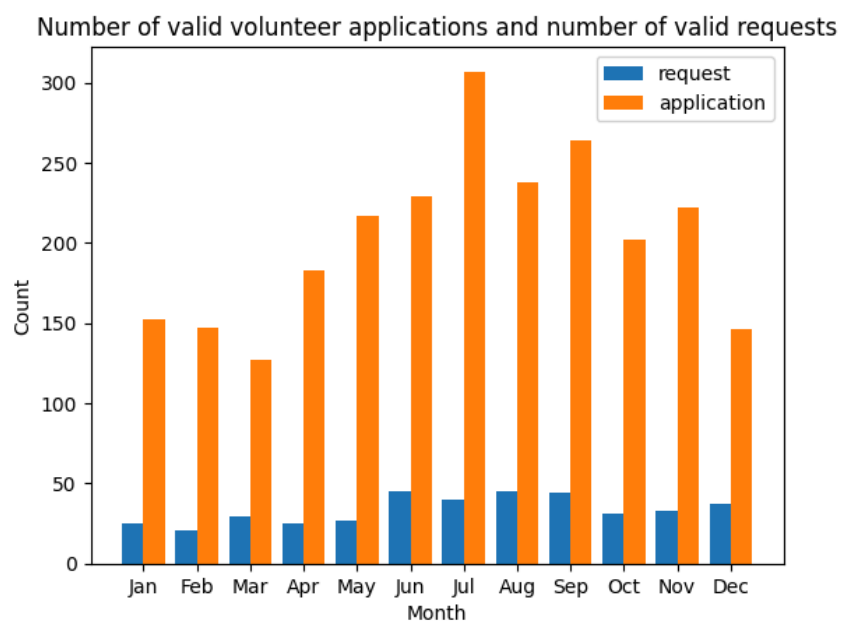
Reasoning for the criterias:

- Skill score: From the database, we cannot know how good a volunteer is at a certain skill. Hence, for example, for a request that requires *CommunicationAndMarketing* and *DigitalCompetence* a person who has both skills should have an advantage over a person who has only one.
- Range score: By using this formula, a volunteer who can cover more of the request's locations will be prioritized. Hence, it would be easier to assign volunteers to locations later.
- Travel readiness: In this criteria, 1200 minutes or 20 hours is chosen as a threshold. Anyone whose travel readiness is greater than 1200 will get 0%, and anyone with a travel readiness score between 0 and 1200 will have their score scaled, where 1200 corresponds to 0% and 0 corresponds to 100%.

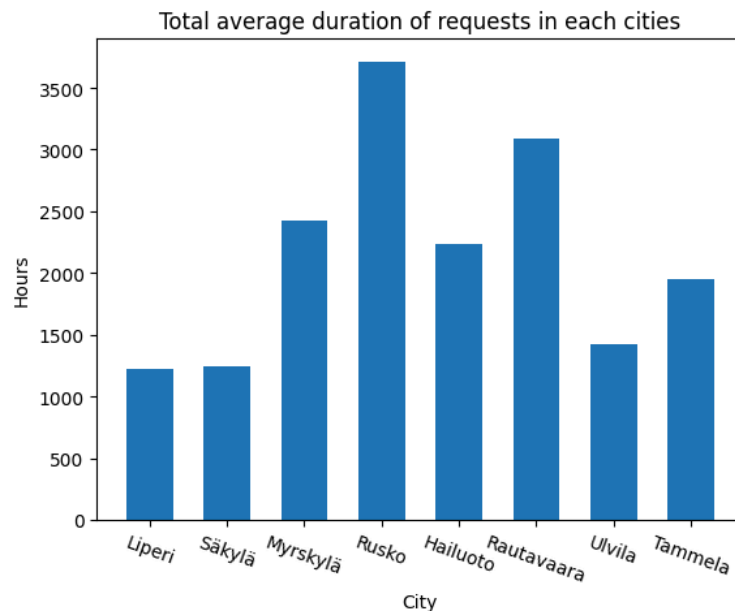
- Interest score: If a person was not interested in the request, he/she would not have applied. Hence, the interest score is only worth 5%.

	request_id	volunteer_id	final_score
0	1	230283-963X	77.294394
1	1	211074-9401	44.120617
2	1	011074-9149	39.121826
3	1	211099-910H	39.090909
4	1	250681-919H	32.771189
8	2	270794-9576	71.784089
9	2	220782-910B	59.593551
10	2	190697-999B	58.488969
11	2	200569-926L	47.436888
12	2	101003A9918	46.337514
13	3	190697-999B	95.010708
14	3	030693-935X	85.333333
15	3	200958-9326	62.354375
16	3	100494-989U	62.014792
17	3	220782-910B	39.028333
18	4	010573-901K	90.504413
19	4	231269-913S	76.943692
20	4	250684-9410	65.206332
21	4	311267-9044	26.805777

3. In this, requests and volunteer's applications are grouped by month according to their start date. Then a double bar chart is graphed to illustrate the number of valid volunteer applications and the number of valid requests. Then according to the graph, the trend and correlation are analyzed. Notably, when analyzing correlation of the number of applications, I split the graph at 'July' since the correlation is positive on one side and negative on the other.



4. For this I chose to analyze the total average duration (in hours) of requests associated with each city. The total hour of a request is calculated as the total hour of it divided by the number of its locations. Then these hours are grouped by cities and sum up.



## C. LEARNED LESSON

- Early preparation and work split can help simplify workflow and time management.
- Using common table expressions (CTE) makes SQL code more readable and organized than using subquery.
- Using LEFT JOIN instead of JOIN can help maintain all rows from the left table, even if there are no corresponding matches in the right table. This ensures that no data from the left table is omitted.
- Completing the documentation will significantly simplify the process of creating presentation slides.